

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**



Nguyễn Ngọc Khang

**NGHIÊN CỨU THUẬT TOÁN PHÂN TÁCH
VẾT NÚT TRONG ẢNH BỀ MẶT THÀNH CÁC
DẠNG ĐƠN GIẢN**

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Cơ kỹ thuật

HÀ NỘI – 2022

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**



Nguyễn Ngọc Khang

**NGHIÊN CỨU THUẬT TOÁN PHÂN TÁCH
VẾT NÚT TRONG ẢNH BỀ MẶT THÀNH CÁC
DẠNG ĐƠN GIẢN**

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY

Ngành: Cơ kỹ thuật

Cán bộ hướng dẫn: TS. Đinh Trần Hiệp

HÀ NỘI – 2022

NGHIÊN CỨU THUẬT TOÁN PHÂN TÁCH VẾT NÚT TRONG ẢNH BỀ MẶT THÀNH CÁC DẠNG ĐƠN GIẢN

Nguyễn Ngọc Khang

Khóa QH-2018-I/CQ, ngành Cơ kỹ thuật

Tóm tắt đồ án tốt nghiệp:

Phát hiện, giám sát và bảo trì cơ sở hạ tầng một vấn đề hết sức cần thiết ngày nay. Trong đó vết nứt cũng mang đến những mối nguy hiểm khó lường vì vậy việc tìm ra chúng là hết sức cần thiết. Hiện nay để đảm bảo sự an toàn cho người lao động trong việc tìm kiếm nên việc phát hiện vết nứt được thực hiện bởi các thuật toán xử lý ảnh nhằm tăng tốc độ và độ chính xác cũng như tính thuật tiện. Trong giới hạn của đồ án tốt nghiệp này, em sẽ giới thiệu và áp dụng các thuật toán phân cụm, các thuật toán tìm đặc trưng quan trọng của vân tay (minutiae) để tìm điểm giao của các vết nứt phức tạp từ đó cải thiện độ chính xác khi tìm kiếm vết nứt. Đồ án cũng đóng góp một giao diện Matlab gắn nhãn điểm giao một cách dễ dàng và sử dụng chúng để đánh giá hiệu suất của các thuật toán tìm điểm giao.

Từ khóa: tìm kiếm điểm giao, GUI, DBSCAN, Crossing Number, Bwmorph,

LỜI CAM ĐOAN

Tôi xin cam đoan những kết quả và đóng góp trong đề án này được trình bày một cách chính xác và trung thực, tất cả các tài liệu tham khảo, công trình nghiên cứu của người khác được sử dụng trong đề án đều được ghi rõ nguồn, được liệt kê tại danh mục các tài liệu tham khảo của đề án.

Những kết quả trên bài báo cáo hoàn toàn không sao chép của người khác, không tự bịa đặt về kết quả. Nếu như những gì tôi nói trên đây là trái sự thật, tôi xin chịu hình thức kỷ luật cao nhất của nhà trường.

Hà nội, ngày 25 tháng 11 năm 2022

Sinh viên

Nguyễn Ngọc Khang

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành đến thầy Đinh Trần Hiệp đã trang bị cho em những kiến thức, kỹ năng cơ bản cần có để hoàn thành đề tài nghiên cứu này. Cảm ơn thầy vì luôn tận tình chỉ bảo, đưa ra ý kiến để bài báo cáo của em trở nên hoàn thiện hơn.

Em cũng xin chân thành cảm ơn đến các thầy cô trong khoa Cơ học kỹ thuật và Tự động hóa đã tạo điều kiện tốt nhất để cho em được hoàn thành đề tài.

Em xin gửi lời cảm ơn đến các tác giả của các bài báo, nghiên cứu, trang web có liên quan đã giúp em có thêm nhiều kiến thức và có thêm cơ sở để hoàn thiện đồ án này.

Tuy nhiên trong quá trình nghiên cứu đề tài, do kiến thức chuyên ngành còn hạn chế nên em vẫn còn nhiều thiếu sót khi tìm hiểu đánh giá và trình bày về đề tài. Rất mong nhận được sự quan tâm, góp ý của các thầy/cô giảng viên bộ môn để đề tài của em được đầy đủ và hoàn chỉnh hơn.

Hà nội, ngày 25 tháng 11 năm 2022

Sinh viên

Nguyễn Ngọc Khang

MỤC LỤC

MỞ ĐẦU	1
1. Tính cấp thiết của đề tài	1
2. Ý nghĩa khoa học và thực tiễn	1
3. Đối tượng và phương pháp nghiên cứu	1
4. Nội dung đồ án tốt nghiệp	2
CHƯƠNG 1 CÁC THUẬT TOÁN TÌM ĐIỂM GIAO	3
1.1. Tổng quát hóa về điểm giao và các thuật toán tìm điểm giao	3
1.2. Thuật toán tìm điểm giao đối với vết nứt có độ rộng 1 pixel	3
1.2.1. BWMORPH	3
1.2.2. DBSCAN	4
1.2.3. OPTICS	7
1.2.4. Crossing Number	8
1.3. Thuật toán tìm điểm giao đối với vết nứt có độ rộng lớn hơn 1 pixel	10
1.3.1. Run Representation	10
1.3.2. Square-based	12
1.4. So sánh các thuật toán	13
CHƯƠNG 2 GIAO DIỆN MATLAB PHỤC VỤ GÁN NHÃN DỮ LIỆU	15
2.1. Sự cần thiết của việc gán nhãn dữ liệu	15
2.2. Quy trình xử lý dữ liệu, xuất dữ liệu của GUI	18
2.2.1. Quá trình chuẩn bị dữ liệu	18
2.2.2. Quá trình xử lý dữ liệu	18
CHƯƠNG 3 DỮ LIỆU VÀ CÁC THÔNG SỐ KIỂM THỬ	23
3.1. Các tập dữ liệu	23
3.1.1. Tập dữ liệu CrackTree200	23
3.1.2. Tập dữ liệu CRKWH100	23
3.1.3. Tập dữ liệu CrackLS315	24
3.1.4. Dữ liệu được sử dụng	24
3.1.5. Xử lý dữ liệu	25
3.2. Các thông số đánh giá	31
3.2.1. Các thông số đánh giá dành cho vết nứt khác 1 pixel	31

3.2.2. Các thông số đánh giá cho vết nứt có độ rộng 1 pixel	32
CHƯƠNG 4 KẾT QUẢ THỰC NGHIỆM	36
4.1. GUI.....	36
4.2. Kết quả của các thuật toán	38
4.2.1 Kết quả của thuật toán có độ rộng 1 pixel.....	38
4.2.2. Kết quả của thuật toán có độ rộng khác 1 pixel	47
KẾT LUẬN	56
TÀI LIỆU THAM KHẢO	57

Danh mục bảng biểu

Bảng 1. 1: Các trường hợp xảy ra của thuật toán Crossing Number.....	9
Bảng 4. 1: Kết quả đánh giá định lượng hàm Bwmorph.....	39
Bảng 4. 2: Kết quả định lượng của thuật toán Crossing Number.....	41
Bảng 4. 3: Kết quả định lượng của thuật toán DBSCAN.....	43
Bảng 4. 4: Kết quả định lượng của thuật toán OPTICS	45
Bảng 4. 5: Kết quả định lượng của cả tập dữ liệu có độ rộng 1 pixel trên các thuật toán	46
Bảng 4. 6: Kết quả định lượng của thuật toán Square based.....	50
Bảng 4. 7: Kết quả định lượng của thuật toán Run Representation	54
Bảng 4. 8: Kết quả trung bình của các thuật toán trên tập dữ liệu vết nứt có độ rộng khác 1 pixel.....	54

Danh mục hình ảnh

Hình 1. 1: Sự hình thành của điểm giao	3
Hình 1. 2: Các hoạt động của BWMORPH.....	4
Hình 1. 3: Minh họa thuật toán DBSCAN [2].....	4
Hình 1. 4: Các loại dữ liệu của DBSCAN.....	6
Hình 1. 5: Các điểm lân cận quanh pixel P	8
Hình 1. 6: Các trường hợp xảy ra của thuật toán Crossing Number	9
Hình 1. 7: Ma trận 3x3 biểu thị các trường hợp xảy ra	10
Hình 1. 8: Mã hóa theo chiều ngang và chiều dọc [6].....	11
Hình 1. 9: Mô tả hòa động của thuật toán Square based	13
Hình 2. 1: Những nút nhấn có trong giao diện GUI.....	16
Hình 2. 2: Giao diện chính của GUI.....	19
Hình 2. 3: Phóng to và chọn điểm giao	19
Hình 2. 4: Điểm giao tạm thời	20
Hình 2. 5: Xác nhận lại điểm giao.....	20
Hình 2. 6: Tọa độ được lưu sau khi gán nhãn	21
Hình 2. 7: Lưu dữ liệu và chuyển qua ảnh tiếp theo	21
Hình 2. 8: Lưu đồ thuật toán	22
Hình 3. 1: Tập ảnh CrackTree200	23
Hình 3. 2: Tập ảnh CRKWH100	24
Hình 3. 3: Tập ảnh CrackLS315.....	24
Hình 3. 4: Khoảng cách nhỏ giữa các vết nứt đơn	25
Hình 3. 5: Quá trình dẫn nở vết nứt.....	26
Hình 3. 6: Dẫn nở vết nứt theo hình chữ nhật	27
Hình 3. 7: dẫn nở vết nứt theo hình thoi và hình chữ thập.....	28
Hình 3. 8: Điểm giao tìm được từ các nhân khác nhau	29
Hình 3. 9: Vết nứt sau khi lấp đầy khoảng trống.....	30
Hình 3. 10: Phân loại các thông số đánh giá	32
Hình 4. 1: Giao diện trực quan của GUI	36
Hình 4. 2: Quá trình chọn, xác nhận và lưu điểm giao.....	37
Hình 4. 3: Quá trình chuyển đến ảnh tiếp theo.....	38

Hình 4. 4: Kết quả của hàm Bwmorph	39
Hình 4. 5: Nhận diện thừa điểm giao	40
Hình 4. 6: Kết quả của thuật toán Crossing Number.....	41
Hình 4. 7: Nhận diện đầy đủ điểm giao khi chúng ở khoảng cách gần nhau	42
Hình 4. 8: Kết quả của thuật toán DBSCAN	43
Hình 4. 9: Nhận diện thiếu điểm giao.....	44
Hình 4. 10: Kết quả của thuật toán OPTICS	45
Hình 4. 11: Nhận diện sai điểm giao	46
Hình 4. 12: Kết quả của thuật toán Square based.....	49
Hình 4. 13: Nhận diện được 2 điểm giao nhưng chỉ được tính là một điểm giao	49
Hình 4. 14: Chỉ nhận diện được 1 điểm giao.....	50
Hình 4. 15: Nhận diện sai điểm giao	50
Hình 4. 16: Kết quả của thuật toán Run Representation	52
Hình 4. 17: Quá trình xác định điểm giao của thuật toán Run Representation	53
Hình 4. 18: Không nhận diện được và nhận diện sai điểm giao	53
Hình 4. 19: Các trường hợp nhận diện sai và không nhận diện được điểm giao	55

MỞ ĐẦU

1. Tính cấp thiết của đề tài

Trong những năm gần đây, xử lý ảnh được các nhà khoa học nghiên cứu mà phát triển mạnh mẽ để có thể ứng dụng vào đời sống thực tế. Xử lý ảnh cũng xuất hiện trong nhiều ngành nhiều nghề. Trong lĩnh vực giải trí, xử lý ảnh giúp người dùng có những bức ảnh đẹp hơn, cải thiện những bức ảnh có chất lượng kém. Trong y học, xử lý ảnh giúp phát hiện những khối u sớm, kiểm tra lỗi thiết bị y tế. Trong công nghiệp, xử lý ảnh giúp kiểm tra nhãn trên sản phẩm, hướng dẫn robot hay kiểm tra bao bì, đọc mã vạch hay phát hiện những sản phẩm lỗi. Ở một số lĩnh vực giám sát như các bãi gửi xe (giám sát số lượng xe vào ra) hay bảo trì các công trình kiến trúc xây dựng trở nên dễ dàng hơn.

Với sự phát triển đó nhiều thuật toán phát hiện vết nứt đã được phát triển dựa trên các mô hình học máy, học sâu, cùng với đó là những kết quả vô cùng khả thi. Tuy nhiên đối với những vết nứt có cấu trúc phức tạp lại không có kết quả tốt như những vết nứt đơn giản hay những vết nứt đơn. Dựa vào điều đó một ý tưởng về việc tách vết nứt phức tạp thành các vết nứt đơn được đưa ra dựa trên điểm giao của những vết nứt đơn. Từ đó sẽ giúp cải thiện kết quả của các mô hình học máy từ nay về sau.

2. Ý nghĩa khoa học và thực tiễn

Đánh giá, giám sát mức độ xuống cấp và bảo trì cơ sở hạ tầng là một công việc yêu cầu độ chính xác cao để có thể sửa chữa kịp thời tránh gây ra những tai nạn không đáng có và tất nhiên máy tính sẽ làm nhiệm vụ này tốt hơn, an toàn hơn so với việc đánh giá thủ công của con người. Sau nhiệm vụ thu thập dữ liệu về sự xuống cấp của cơ sở hạ tầng thì việc dùng các thuật toán để phát hiện vết nứt đang là một đề tài được nhiều người quan tâm. Ngày nay, các thuật toán xử lý ảnh đã được các chuyên gia phát triển và nâng cấp để ngày càng trở nên chính xác hơn.

3. Đối tượng và phương pháp nghiên cứu

Trong đồ án tốt nghiệp này, nghiên cứu tập trung chủ yếu vào bài toán khảo sát và cải thiện đầu ra của việc phát hiện vết nứt trong xử lý ảnh. Nhiệm vụ của đề tài là khái quát cơ sở lý thuyết, kế thừa các nghiên cứu đã có, đề xuất giải pháp cho việc cải thiện chất lượng thuật toán phát hiện vết nứt. Đóng góp của đồ án này là đưa ra một giao diện gán nhãn dữ liệu điểm giao để có kết quả đánh giá hiệu suất của các thuật toán nhận diện điểm giao, đây là một giao diện được thiết kế trên Matlab vừa chưa từng có trước đây.

Từ đó một loạt các thuật toán tìm điểm giao của vết nứt được đưa ra, từ thuật toán có hiệu suất thấp đến thuật toán có hiệu suất cao. Tất cả chúng đều được đánh giá bằng dữ liệu đã được gán nhãn trước đó bằng giao diện trên Matlab

4. Nội dung đề án tốt nghiệp

Đề án được trình bày trong 4 chương cung cấp cái nhìn tổng thể và những đóng góp của đề tài, cơ sở lý thuyết, quy trình, kết quả tiến hành thực nghiệm và so sánh với các thuật toán với nhau.

Chương 1 nêu ra một loạt các thuật toán tìm điểm giao từ các hàm có sẵn, các thuật toán phân cụm, các thuật toán sử dụng cho dấu vân tay cũng được áp dụng cho việc tìm điểm giao của vết nứt.

Chương 2 nêu ra đóng góp chính của đề án này đó chính là giao diện gán nhãn dữ liệu được phát triển dựa trên Matlab. Chương này sẽ làm nổi bật những ưu điểm và tính cần thiết của một giao diện gán nhãn dữ liệu.

Chương 3 giới thiệu các tập dữ liệu được sử dụng trong đề án này, các thông số kiểm thử để đánh giá hiệu suất của các thuật toán. Có hai nhóm thông số, một dành cho các thuật toán sử dụng hình ảnh vết nứt có độ rộng 1 pixel, những vết nứt có độ rộng khác 1 pixel sẽ được đánh giá bởi nhóm thông số còn lại.

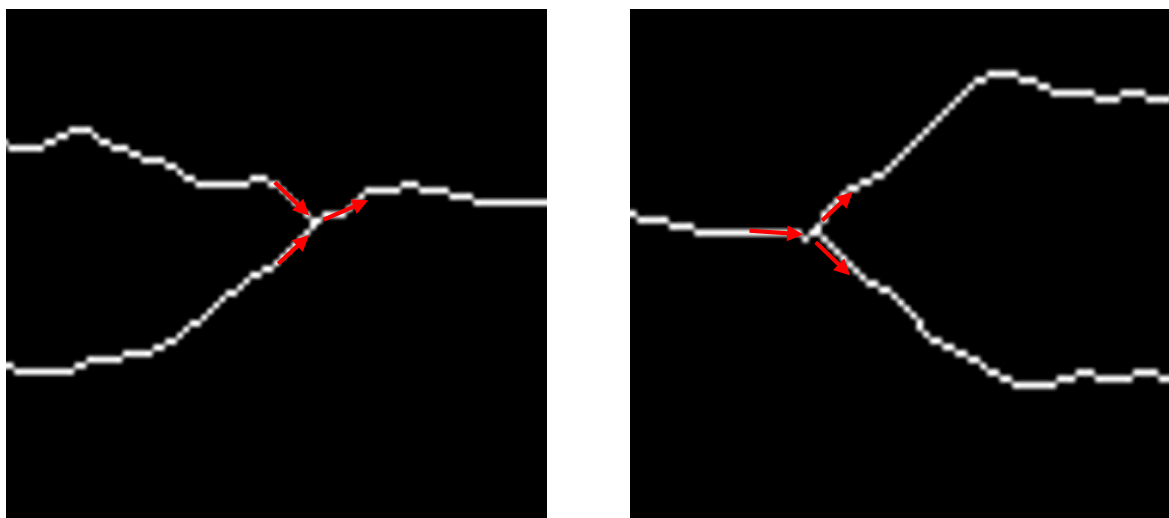
Chương 4 thể hiện kết quả của giao diện gán nhãn dữ liệu của hiệu suất thực của các thuật toán tìm điểm giao, cùng với đó là những tồn tại và cách khắc phục đối với từng thuật toán.

CHƯƠNG 1

CÁC THUẬT TOÁN TÌM ĐIỂM GIAO

1.1. Tổng quát hóa về điểm giao và các thuật toán tìm điểm giao

Vết nứt tồn tại dưới rất nhiều hình dạng và cấu trúc, từ đơn giản tới phức tạp. Đối với các vết nứt phức tạp sẽ chứa rất nhiều điểm giao. Điểm giao là giao điểm của từ 3 đường vết nứt đơn trở lên tạo thành, nó đánh dấu sự phân tách của một vết nứt đơn hoặc quy tụ của nhiều vết nứt đơn, như được thể hiện ở hình dưới đây.



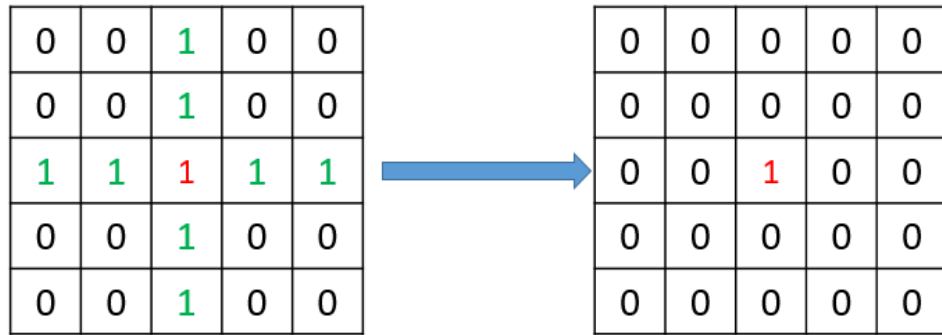
Hình 1. 1: Sự hình thành của điểm giao

Hiện nay, không có một thuật toán tìm điểm giao của vết nứt cụ thể nào, đa số là dựa trên các thuật toán phân cụm hoặc dựa trên các thuật toán tìm các điểm minutiae của những đối tượng có cấu trúc tương đương vết nứt như dấu vân tay. Điểm giao có thể tìm từ những ảnh nhị phân có độ rộng vết nứt không đồng đều, cũng có thể được tìm từ những ảnh có độ rộng 1 pixel. Chúng đều có những ưu nhược điểm, tùy thuộc vào ứng dụng trong nhiều bài toán khác nhau.

1.2. Thuật toán tìm điểm giao đối với vết nứt có độ rộng 1 pixel

1.2.1. BWMORPH

Đây là một hàm có sẵn và được phát triển bởi Matlab [1], có chức năng tìm các điểm phân nhánh của các đối tượng có cấu trúc phân nhánh ứng với đối số đầu vào là 'branchpoints'. Hàm này sẽ loại bỏ mọi pixel xung quanh điểm giao như được thể hiện trong hình dưới đây.

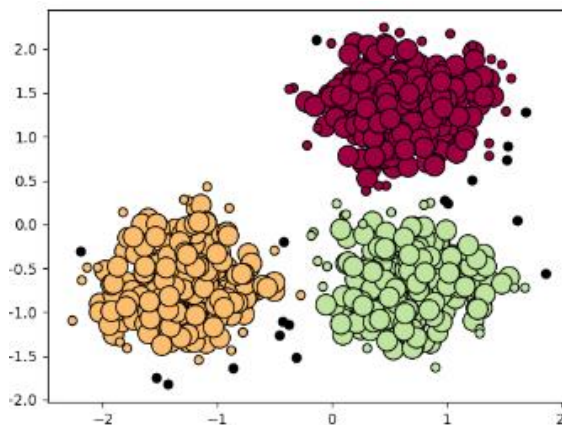


Hình 1. 2: Các hoạt động của BW-MORPH

Ngoài ra BW-MORPH còn rất nhiều đối số khác nhau với nhiều ứng dụng trong lĩnh vực xử lý ảnh. Trong đề án tốt nghiệp này một đối số khác cũng sẽ được sử dụng đến, đó là ‘skel’ có chức năng loại bỏ các pixel trên ranh giới của các đối tượng mà không cho phép các đối tượng bị tách rời. Hay nói với một cái tên dễ hiểu đó là đưa các đối tượng về dạng có độ rộng 1 pixel, đó cũng là cái tên được sử dụng trong bài viết này. Ứng dụng của nó sẽ được làm rõ hơn ở những mục tiếp theo.

1.2.2. DBSCAN

Khi biểu diễn các điểm dữ liệu trong không gian chúng ta sẽ thấy rằng thông thường các vùng không gian có mật độ cao sẽ xen kẽ với các vùng không gian có mật độ thấp. Nếu phải dựa vào mật độ để phân chia thì khả năng rất cao những tâm cụm sẽ tập trung vào những vùng không gian có mật độ cao trong khi biên sẽ rơi vào những vùng không gian có mật độ thấp.



Hình 1. 3: Minh họa thuật toán DBSCAN [2]

DBSCAN là viết tắt của Density – based spatial clustering of applications with noise [2] là một thuật toán cơ sở để phân nhóm dựa trên mật độ. Nó có thể phát hiện ra các cụm có hình dạng và kích thước khác nhau từ một lượng lớn dữ liệu chứa nhiễu.

Trước khi tìm hiểu về thuật toán DBSCAN chúng ta xác định một số định nghĩa mà thuật toán này sử dụng.

Định nghĩa 1: Vùng lân cận epsilon (ε) của một điểm dữ liệu P được định nghĩa là tập hợp tất cả các điểm dữ liệu nằm trong bán kính epsilon xung quanh P . Kí hiệu tập hợp những điểm này là:

$$N_{eps}(P) = \{Q \in D : d(P, Q) \leq \varepsilon\} \quad 1.1$$

Trong đó D là tập hợp tất cả các điểm dữ liệu

Định nghĩa 2: Khả năng tiếp cận trực tiếp mật độ đã đề cập tới việc một điểm có thể tiếp cận trực tiếp với một điểm dữ liệu khác. Cụ thể là một điểm Q được coi là có thể tiếp cận trực tiếp bởi điểm P tương ứng với tham số epsilon và $minPts$ nếu như nó thỏa mãn 2 điều kiện:

- Q nằm trong vùng lân cận epsilon của P : $Q \in N_{eps}(P)$
- Số lượng các điểm dữ liệu nằm trong vùng lân cận epsilon tối thiểu là $minPts$: $|N_{eps}(Q)| \geq minPts$

Như vậy một điểm dữ liệu có thể tiếp cận được trực tiếp tới một điểm khác không chỉ dựa vào khoảng cách giữa chúng mà còn phụ thuộc vào mật độ các điểm dữ liệu trong vùng lân cận *epsilon* phải tối thiểu bằng *minPts*. Khi đó cùng lân cận được coi là có mật độ cao và sẽ được phân vào các cụm. Trái lại thì vùng lân cận sẽ có mật độ thấp. Trong trường hợp mật độ thấp thì điểm dữ liệu ở trung tâm được coi là không kết nối trực tiếp tới những điểm khác trong vùng lân cận và những điểm này có thể rơi vào bên ngoài của cụm hoặc là một điểm dữ liệu nhiễu không thuộc về cụm nào.

Định nghĩa 3: Khả năng tiếp cận mật độ liên quan đến cách hình thành một chuỗi liên kết điểm trong cụm. Cụ thể là một tập hợp chuỗi điểm $\{P_i\}_{i=1}^n \subset D$ mà nếu như bất kỳ một điểm P_i nào cũng đều có thể tiếp cận trực tiếp mật độ bởi P_{i-1} theo tham số *epsilon* và *minPts* thì khi đó ta nói điểm $P = P_n$ có khả năng kết nối mật độ tới điểm $Q = P_1$.

Trong thuật toán DBSCAN sử dụng 2 tham số chính đó là:

- *minPts*: là một ngưỡng số điểm dữ liệu tối thiểu được nhóm lại với nhau nhằm xác định một vùng lân cận *epsilon* có mật độ cao. Số lượng *minPts* không bao gồm điểm ở tâm.

- *Epsilon*: một giá trị khoảng cách được sử dụng để xác định vùng lân cận *epsilon* của bất kỳ điểm dữ liệu nào. Tức là nếu khoảng cách giữa hai điểm thấp hơn hoặc bằng *epsilon* thì chúng được coi là lân cận với nhau.

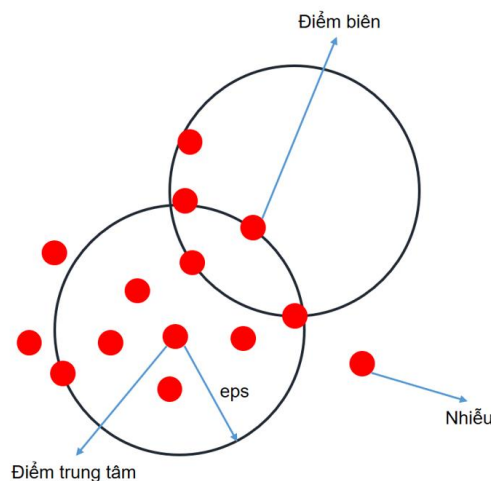
Hai tham số trên sẽ được sử dụng để xác định vùng lân cận *epsilon* và khả năng tiếp cận giữa các điểm dữ liệu lẫn nhau. Từ đó kết nối chuỗi dữ liệu vào chung một cụm.

Thuật toán này có 3 loại dữ liệu:

Điểm trung tâm: Một điểm là điểm trung tâm nếu nó có nhiều hơn *minPts* trong vòng *epsilon*.

Điểm biên: Một điểm có ít hơn *minPts* trong *epsilon* nhưng nó nằm trong vùng lân cận của điểm trung tâm.

Nhiều: một điểm không phải là điểm trung tâm hoặc điểm biên



Hình 1. 4: Các loại dữ liệu của DBSCAN

Quy trình [3] của thuật toán:

- **Bước 1**: Thuật toán lựa chọn một điểm dữ liệu bất kỳ. Sau đó tiến hành xác định các điểm trung tâm và điểm biên thông qua vùng lân cận *epsilon* bằng cách lan truyền theo liên kết chuỗi các điểm thuộc cùng một cụm.
- **Bước 2**: Đối với mỗi điểm trung tâm nếu nó chưa được gán cho một cụm, hãy tạo một cụm mới.
- **Bước 3**: Tìm đệ quy tất cả các điểm được kết nối mật độ của nó và gán chúng vào cùng một cụm với điểm trung tâm. Một điểm *a* và *b* được cho là liên kết mật độ nếu tồn tại một điểm *c* có đủ số điểm trong các điểm lân cận của nó và cả hai điểm *a* và *b* đều nằm trong khoảng cách *epsilon*. Đây là một quá trình xâu

chuỗi. Vì vậy, nếu b là lân cận của c , c là lân cận của d , d là lân cận của e , lần lượt là lân cận của a nghĩa là b cũng là lân cận của a .

- **Bước 4:** Lập lại các điểm chưa được duyệt còn lại trong tập dữ liệu. Nhưng điểm không thuộc về cụm nào là nhiều.

Xác định tham số: là một bước quan trọng và ảnh hưởng trực tiếp tới kết quả của các thuật toán. Đối với thuật toán DBSCAN cũng không ngoại lệ. Chúng ta cần phải xác định chính xác tham số cho thuật toán DBSCAN một cách phù hợp với từng bộ dữ liệu cụ thể, tùy theo đặc điểm và tính chất của phân phối của bộ dữ liệu. Hai tham số cần lựa chọn trong DBSCAN đó chính là $minPts$ và $epsilon$.

- $minPts$: theo quy tắc chung, $minPts$ tối thiểu có thể được tính theo số chiều D trong tập dữ liệu đó là $minPts \geq D+1$. Một giá trị $minPts=1$ không có ý nghĩa. Với $minPts \leq 2$, kết quả sẽ giống như phân cụm phân cấp. Do đó, $minPts$ phải được chọn ít nhất là 3. Tuy nhiên, các giá trị lớn hơn thường tốt hơn cho các tập dữ liệu có nhiễu và kết quả phân cụm thường hợp lý hơn. Theo quy tắc chung thì thường chọn $minPts = 2 \times \dim$. Trong trường hợp dữ liệu có nhiễu hoặc có nhiều quan sát lặp lại thì cần lựa chọn giá trị $minPts$ lớn hơn nữa tương ứng với những bộ dữ liệu lớn.

- $epsilon$: Giá trị $epsilon$ có thể được chọn bằng cách vẽ đồ thị k-distance. Đây là đồ thị thể hiện giá trị khoảng cách trong thuật toán k-Means. Những khoảng cách này trên đồ thị được sắp xếp theo thứ tự giảm dần. Các giá trị tốt của $epsilon$ là vị trí mà đồ thị này cho thấy xuất hiện một điểm khuy tay: nếu $epsilon$ được chọn quá nhỏ, một phần lớn dữ liệu sẽ không được phân cụm và được xem là nhiễu; trong khi đối với giá trị $epsilon$ quá cao, các cụm sẽ hợp nhất và phần lớn các điểm sẽ nằm trong cùng một cụm. Nói chung, các giá trị nhỏ của $epsilon$ được ưu tiên hơn và theo quy tắc chung chỉ một phần nhỏ các điểm nên nằm trong vùng lân cận $epsilon$.

1.2.3. OPTICS

OPTICS là viết tắt của cụm từ Ordering Points To Identify Cluster Structure [4]. Nó lấy cảm hứng từ thuật toán phân cụm DBSCAN. Nó bổ sung thêm 2 thuật ngữ vào các khái niệm về phân cụm DBSCAN [5].

Khoảng cách tâm: là giá trị bán kính tối thiểu cần thiết để phân loại một điểm nhất định làm điểm trung tâm. Nếu điểm đã cho không phải là điểm trung tâm, thì khoảng cách tâm là không xác định.

Khoảng cách khả năng tiếp cận: Nó được xác định đối với một điểm dữ liệu khác. Khoảng cách khả năng tiếp cận giữa một điểm p và q là giá trị lớn nhất của khoảng cách tâm của p và khoảng cách Euclide (hoặc một số chỉ số khoảng cách khác) giữa p và q . Lưu ý rằng khoảng cách khả năng tiếp cận không được xác định nếu q không phải là điểm trung tâm.

Kỹ thuật phân cụm này khác với các kỹ thuật phân cụm khác ở chỗ kỹ thuật này không phân đoạn dữ liệu thành các cụm một cách dễ dàng. Thay vào đó, nó tạo ra một hình ảnh hóa về khoảng cách khả năng tiếp cận và sử dụng hình ảnh hóa này để phân cụm dữ liệu.

Chi phí bộ nhớ: Kỹ thuật phân cụm OPTICS yêu cầu nhiều bộ nhớ hơn vì nó duy trì một hàng đợi ưu tiên để xác định điểm dữ liệu tiếp theo gần nhất với điểm hiện đang được xử lý về khoảng cách khả năng tiếp cận. Nó cũng đòi hỏi nhiều sức mạnh tính toán hơn vì các truy vấn hàng xóm gần nhất phức tạp hơn các truy vấn bán kính trong DBSCAN.

Ít tham số hơn: Kỹ thuật phân cụm OPTICS không cần suy trì tham số epsilon và chỉ được đưa ra trong mã giả ở trên để giảm thời gian thực hiện. Điều này dẫn đến việc giảm quá trình phân tích điều chỉnh thông số.

Kỹ thuật này không tách dữ liệu đã cho thành các cụm. Nó chỉ đơn thuần tạo ra một biểu đồ khoảng cách khả năng tiếp cận và dựa trên sự giải thích của lập trình viên để phân cụm các điểm cho phù hợp.

1.2.4. Crossing Number

Điểm giao được tìm bằng cách quét vùng lân cận của mỗi pixel trong hình ảnh bằng ma trận 3x3 [6], [7].

P4	P3	P2
P5	P	P1
P6	P7	P8

Hình 1. 5: Các điểm lân cận quanh pixel P

Phương pháp này được ưa chuộng hơn các phương pháp khác vì tính hiệu quả và tính đơn giản vốn có của nó. Phương pháp này liên quan đến việc sử dụng hình ảnh có độ rộng 1 pixel. Các điểm giao được trích xuất bằng cách quét vùng lân cận của mỗi pixel trong hình ảnh bằng ma trận 3x3. Một giá trị CN sau đó được xác định.

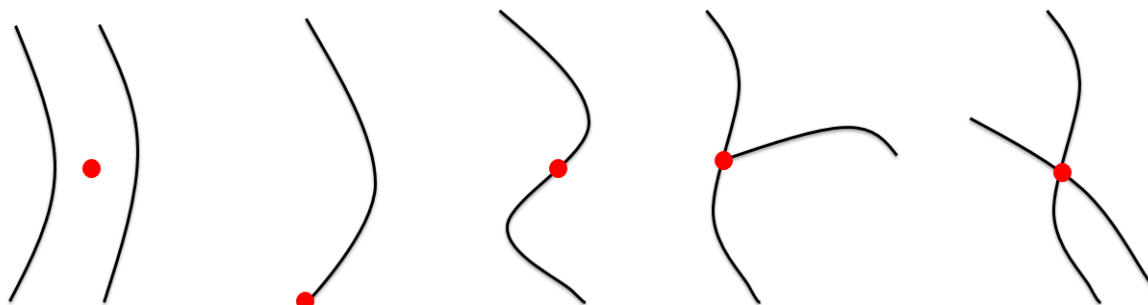
$$CN = 0.5 \sum_{i=1}^8 |P_i - P_{i-1}| \quad 1.2$$

Trong đó $P_9 = P_1$. Nó được định nghĩa là một cửa sổ tổng của sự khác biệt giữa các cặp pixel liên kề trong các pixel viền của ma trận 3x3. Sẽ có một số trường hợp xảy ra như sau:

Bảng 1. 1: Các trường hợp xảy ra của thuật toán Crossing Number

CN	Đặc tính
0	Điểm bị cô lập
1	Điểm kết thúc của một vết nứt
2	Điểm bất kỳ nằm trên vết nứt đơn
3	Điểm phân đôi
4	Điểm phân ba

Pixel bị cô lập là điểm pixel mà lân cận nó không có điểm pixel nào, nó thể là nhiễu. Điểm kết thúc của một vết nứt là pixel cuối cùng một đường vết, lân cận nó chỉ có một điểm pixel duy nhất. Điểm bất kỳ nằm trên vết nứt đơn là những điểm chiếm một lượng pixel lớn của vết nứt, nó chỉ có 2 pixel lân cận. Điểm phân đôi và phân ba là



Hình 1. 6: Các trường hợp xảy ra của thuật toán Crossing Number

điểm nói đến rất nhiều trong đồ án tốt nghiệp này, nó là điểm bắt nguồn của sự rẽ nhánh hoặc hội tụ từ 3 vết nứt đơn trở lên. Chúng được thể hiện cụ thể như hình dưới đây.

Để có một cái nhìn khái quát hơn về các cấu trúc này, hình dưới đây thể hiện các trường hợp trên trong ma trận 3x3.

1	
2	
3	
4	

Hình 1. 7: Ma trận 3x3 biểu thị các trường hợp xảy ra

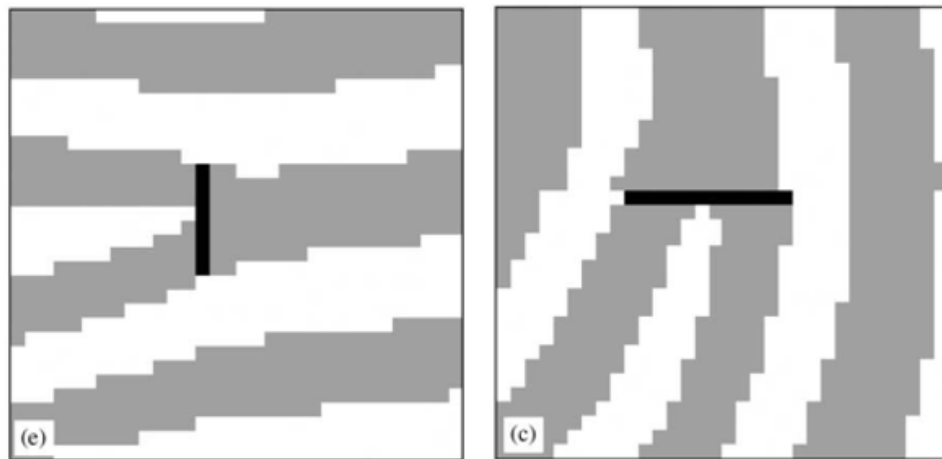
1.3. Thuật toán tìm điểm giao đối với vết nứt có độ rộng lớn hơn 1 pixel

Khác với các thuật toán tìm điểm giao trên vết nứt có độ rộng là 1 pixel, trong phần này điểm giao sẽ được trích xuất trực tiếp từ ảnh vết nứt ban đầu mà không cần bước xử lý để đưa về dạng 1 pixel. Điều này sẽ tiết kiệm được nhiều thời gian, tuy nhiên cũng tồn tại nhiều nhược điểm hơn so với những thuật toán xác định điểm giao trên vết nứt có độ rộng là 1 pixel.

1.3.1. Run Representation

Phương pháp này được giới thiệu trong [6], trích xuất nhanh các điểm minutiae của dấu vân tay dựa trên mã hóa độ dài chiều ngang và chiều dọc từ các hình ảnh nhị phân mà không cần quá trình đưa về dạng 1 pixel tốn kém về mặt tính toán [8], [9]. Một số ràng buộc hình học được giới thiệu để kiểm tra tính hợp lệ của các lần chạy đặc tính.

Cách đơn giản nhất để sử dụng hình ảnh nhị phân là chọn một giá trị ngưỡng và phân loại tất cả các pixel có giá trị trên ngưỡng này là màu trắng và tất cả các pixel khác là màu đen. Vấn đề là làm sao để chọn đúng ngưỡng. Trong nhiều trường hợp, việc tìm một ngưỡng tương thích với toàn bộ hình ảnh là rất khó và trong nhiều trường hợp thậm chí là không thể. Do đó, cần mã hóa hình ảnh thích ứng trong đó ngưỡng tối ưu được chọn cho mỗi cùng hình ảnh [10], [11]. Trong hình ảnh vết nứt nhị phân, các pixel trắng liên tiếp dọc theo đường quét được định nghĩa là một lần chạy. Nói chung, mã hóa thời lượng chạy của hình ảnh nhị phân là danh sách các pixel trắng chạy ngang liền nhau.



Hình 1. 8: Mã hóa theo chiều ngang và chiều dọc [6]

Năm trường hợp sau được xác định:

Trường hợp 1: Không có lần chạy liền kề nào cả trên dòng quét trước đó và dòng quét tiếp theo.

Trường hợp 2: Có hai lần chạy liền nhau trên dòng quét trước và sau.

Trường hợp 3: Có một lần chạy liền kề trên dòng quét trước trước đó hoặc tiếp theo.

Trường hợp 4: Có 2 lần chạy liền nhau trên dòng quét trước đó hoặc dòng quét tiếp theo.

Trường hợp 5: Có nhiều hơn 2 lần chạy liền nhau trên dòng quét trước đó hoặc dòng quét tiếp theo.

Điều kiện đầu tiên có nghĩa là đường chạy là một điểm pixel hoặc một đường cô lập với nhiều hơn 1 pixel. Trường hợp 2 có nghĩa là đường chạy là một phần của vết nứt đơn. Trường hợp thứ 3 có nghĩa là đường chạy là điểm kết thúc của vết nứt, điểm bắt đầu hoặc điểm kết thúc của vết nứt đơn. Trường hợp thứ 4 có nghĩa là hai lần chạy trên

dòng quét trước đó đang hợp nhất hoặc một lần chạy được tách thành 2 lần chạy trong dòng quét tiếp theo.

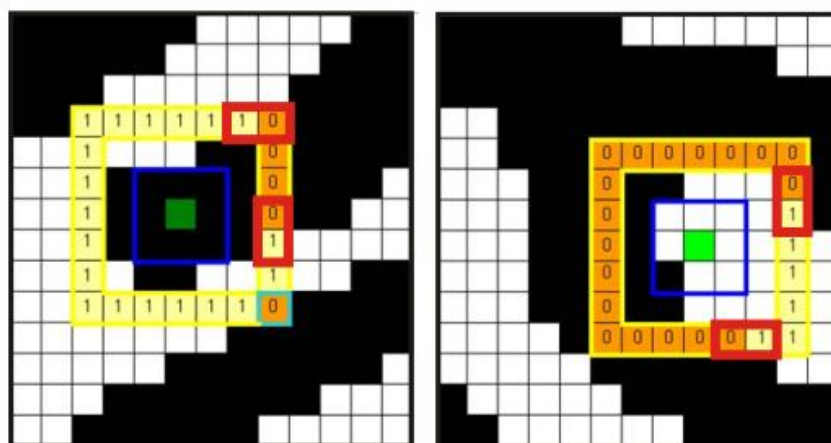
1.3.2. Square-based

Phương pháp này được đề xuất trong [12], [13] cũng là một phương pháp để trích xuất những điểm minutiae của dấu vân tay, trong đồ án tốt nghiệp này nó được sử dụng để tìm các giao điểm của vết nứt. Mỗi điểm minutiae của dấu vân tay có thể được định nghĩa là $M(x, y, t, \alpha)$ trong đó, x và y là tọa độ của điểm minutiae trong hình ảnh, t là loại điểm đặc biệt và α tương ứng với góc tạo bởi tiếp tuyến với đỉnh tương ứng tại điểm (x, y) . Sau khi xác định được cả 4 tham số trên khi đó mục tiêu tìm điểm giao sẽ đạt được.

Thuật toán được đề xuất dựa trên phương pháp phân tích hình ảnh dựa trên các đường dẫn bình phương. Mỗi pixel trong hình ảnh phải được phân loại là điểm phân nhánh hay điểm kết thúc hoặc có thể là những loại khác. Mẫu cường độ gần pixel ứng viên được nghiên cứu dọc theo một đường bình phương. Đặc biệt, số lần chuyển đổi mẫu giữa mức tối trung bình và mức trắng được cố định.

Việc lựa chọn đường vuông làm đường phân tích chỉ liên quan đến các khía cạnh triển khai: đường tròn có thể được coi là một đường lý tưởng. Mặt khác, xử lý hình ảnh dọc theo hàng và cột có thể nhanh hơn với việc đi theo các đường quét có hình dạng khác nhau (tức là hình tròn).

Ngay cả khi đường bình phương không thể được coi là thích ứng như đường tròn, phương pháp được trình bày vẫn hoạt động và nó duy trì tính độc lập quay đặc biệt của nó. Đó là bởi vì dự hiện diện của hai quá trình chuyển đổi gần như được đảm bảo trong trường hợp có các điểm minutiae, bất kể hướng của chúng. Phương pháp hiện tại phù hợp với hình ảnh thang xám. Trên thực tế, hình ảnh thang xám chất lượng cao có thể được lọc trước và tạo sinh học bằng các thuật toán thích ứng nổi tiếp trong tài liệu sinh trắc học. Ngoài ra, phương pháp được đề xuất có thể dễ dàng điều chỉnh cho phù hợp trực tiếp trên hình ảnh thang xám. Phương pháp được trình bày bao gồm các bước sau được lặp lại cho mỗi pixel của hình ảnh đầu vào. Do dấu vân tay cần trích xuất nhiều điểm minutiae, còn đối với vết nứt thì chỉ cần tìm điểm giao nên quá trình trích xuất sẽ đơn giản và gọn hơn.



Hình 1. 9: Mô tả hóa động của thuật toán Square based

Bước 1: Tạo mặt nạ 3x3 xung quanh pixel (x, y). tính giá trị trung bình, nếu mức trung bình lớn hơn 0.75 thì tạm thời coi là giao điểm.

Bước 2: tạo hình vuông chu vi P có kích thước WxW (W được đề xuất có giá trị là 7 và nó cũng được sử dụng trong bài viết này).

Bước 3: Tính toán giao hoán logic trong P.

Bước 4: Thuật toán sẽ tiếp tục nếu có lớn hơn 2 giao hoán logic

Bước 5: tính giá trị trung bình của các pixel trong P, nếu giá trị trung bình nhỏ hơn ngưỡng $I-K$ thì điểm đó được coi là giao điểm (K được đề xuất là 0.7).

1.4. So sánh các thuật toán

Thuật toán	Mô tả	Ưu điểm	Nhược điểm
BWMORPH	Là một hàm của matlab có chức năng tìm các điểm rẽ nhánh của vết nứt.	Là một hàm có sẵn nên tốc độ chạy tương đối nhanh chóng. Dễ dàng áp dụng	Các điểm giao được nhận diện sai và không đầy đủ
DBSCAN	Là một thuật toán phân cụm dựa trên mật độ, do mật độ tại điểm giao lớn hơn những vùng lân cận.	Độ chính xác tương đối cao. Có thể tìm điểm giao trên cả ảnh có độ rộng 1pixel và ảnh	Một số ảnh vết nứt phức tạp thì chưa nhận diện được đầy đủ điểm giao

		có độ rộng lớn hơn 1 pixel	
OPTICS	Là một thuật toán phân cụm giống DBSCAN. Tuy nhiên không giống DBSCAN ở chỗ giữ phân phát cụm cho bán kính vùng lân cận thay đổi. Phù hợp hơn để sử dụng trên các tập dữ liệu lớn hơn so với DBSCAN	Tuy giống DBSCAN nhưng nó cần ít tham số hơn	Độ chính xác không cao bằng DBSCAN. Yêu cầu nhiều bộ nhớ hơn.
Crossing Number	Điểm giao được tìm bằng cách quét vùng lân cận của mỗi pixel trong ảnh bằng ma trận 3x3.	Độ chính xác cao	Không xử lý được trên ảnh có độ dài lớn hơn một pixel. Tốc độ xử lý tương đối chậm
Run Representation	Là một thuật toán tìm nhanh các điểm chi tiết của dấu vân tay dựa trên mã hóa độ dài chiều ngang và chiều dọc được áp dụng đối với vết nứt.	Xử lý trực tiếp trên ảnh có độ rộng lớn hơn 1 pixel mà không cần đưa về dạng 1 pixel	Độ chính xác tương đối thấp. Nhận nhầm điểm giao nhiều. Không nhận diện được chính xác tọa độ điểm giao
Square-based	Phương pháp dựa trên hình vuông, giao hoán logic và giá trị các điểm pixel	Xử lý trực tiếp trên ảnh có độ rộng lớn hơn 1 pixel mà không cần đưa về dạng 1 pixel	Độ chính xác không được cao. Nhận diện nhầm nhiều điểm giao.

CHƯƠNG 2

GIAO DIỆN MATLAB PHỤC VỤ GÁN NHÃN DỮ LIỆU

2.1. Sự cần thiết của việc gán nhãn dữ liệu

Ng . Đây là thuật ngữ chỉ cách giao tiếp của người dùng với các thiết bị máy tính thông qua các thao tác với chữ viết hay hình ảnh, hay vì sử dụng các chương trình phức tạp. GUI là giao diện cho phép người dùng tương tác với các thiết bị máy tính thông qua hình ảnh mà không cần gõ lệnh. Hiện nay, GUI có mặt trong mọi thứ, từ hệ điều hành, chương trình trên máy tính đến các ứng dụng trên điện thoại. Nếu không có GUI, người dùng sẽ cần nhập tất cả các lệnh dưới dạng văn bản, điều này sẽ mất thời gian.

Trước khi đi vào cụ thể giao diện GUI được giới thiệu trong đồ án này hãy cùng tìm hiểu một số khái niệm cơ bản về chủ đề gán nhãn dữ liệu.

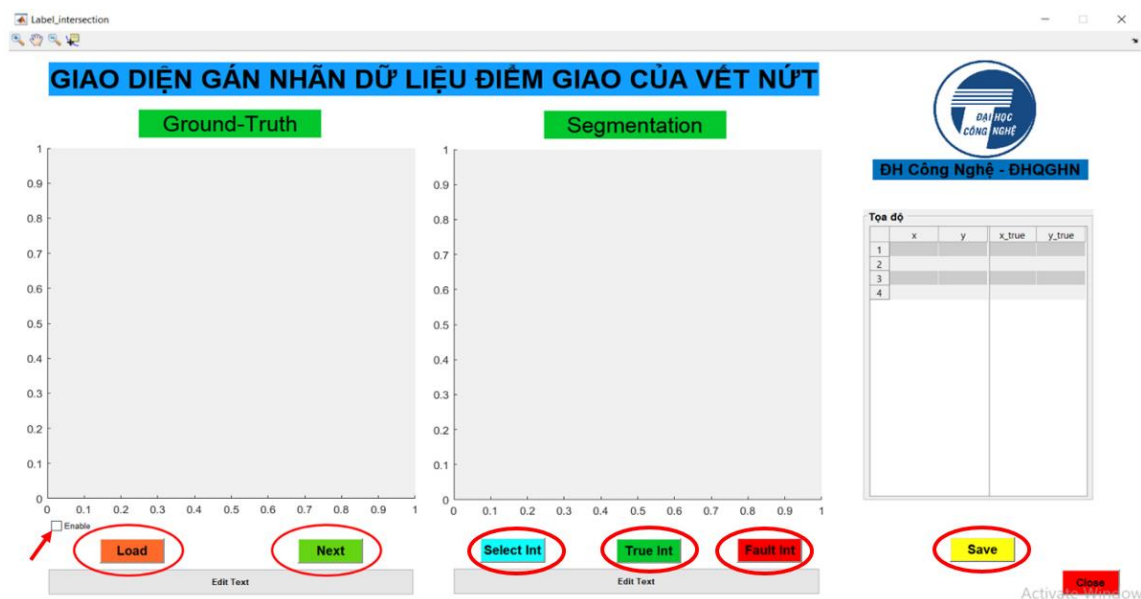
Gán nhãn dữ liệu hoặc chú thích dữ liệu là một phần của giai đoạn tiền xử lý khi phát triển mô hình học máy (ML). nó yêu cầu nhận dạng dữ liệu thô (tức là hình ảnh, tệp văn bản, video), sau đó thêm một hoặc nhiều nhãn vào dữ liệu đó để chỉ định ngữ cảnh của nó cho các mô hình, cho phép mô hình học máy đưa ra dự đoán chính xác. Quá trình này thường bao gồm công việc do con người thực hiện để quá lý thủ công và trong một số trường hợp, có sự trợ giúp của máy tính. Các loại nhãn được xác định trước bởi người thiết kế và được chọn để cung cấp cho các mô hình máy học về những gì được thể hiện để dạy mô hình từ các ví dụ này.

Gán nhãn dữ liệu là một bước quan trọng trong việc phát triển mô hình ML hiệu suất cao. Mặc dù việc ghi nhãn dữ liệu có vẻ đơn giản nhưng không phải lúc nào cũng dễ thực hiện. Do đó, nhiều người phải xem xét nhiều yếu tố và phương pháp để xác định các tiếp cận tốt nhất để gán nhãn. Vì mỗi phương pháp ghi nhãn dữ liệu đều có ưu điểm và nhược điểm nên cần đánh giá chi tiết về độ phức tạp của nhiệm vụ cũng như quy mô, phạm vi và thời gian của dự án.

Lợi ích và thách thức của việc gán nhãn dữ liệu: gán nhãn dữ liệu cung cấp cho người dùng, nhóm, chất lượng và khả năng sử dụng cao hơn. Cụ thể:

Thân thiện với người dùng: Cung cấp cho người dùng một giao diện trực quan hấp dẫn, dễ dàng sử dụng. Tất cả mọi người đều có thể sử dụng một cách dễ dàng bởi người dùng không cần ghi nhớ các lệnh. Do dữ liệu được biểu diễn dưới dạng ký hiệu, hình dạng và biểu tượng, người dùng có thể dễ dàng phân loại và điều hướng các tùy chọn. Để có được các chức năng của chúng, người dùng chỉ nhận nhấp và chúng.

Đơn giản - dễ tiếp cận: Thông qua các nút nhấn, các biểu diễn đồ họa khác của dữ liệu, giúp thông tin có thể được nhận dạng nhanh hơn so với văn bản. Đặc biệt là đối với



Hình 2. 1: Những nút nhấn có trong giao diện GUI

những người không phải là lập trình viên.

Giao diện thu hút hấp dẫn: Từng nút nhấn hay từng cửa sổ đều có thể thiết kế với nhiều màu sắc linh hoạt, bắt mắt.

Phản hồi trực quan tức thì: Sau một thao tác cụ thể, kết quả của thao tác đó sẽ được hiển thị trực tiếp ngay sau đó.

Dự đoán chính xác hơn: việc ghi nhãn dữ liệu chính xác đảm bảo chất lượng tốt hơn trong các thuật toán học máy, cho phép mô hình đào tạo và mang lại kết quả đầu ra mong đợi.

Khả năng sử dụng dữ liệu tốt hơn: việc gán nhãn dữ liệu cũng có thể cải thiện khả năng sử dụng của các biến dữ liệu trong một mô hình. Ví dụ: bạn có thể phân loại lại một biến phân loại thành một biến nhị phân để làm cho nó dễ hiểu hơn cho một số mô hình. Tổng hợp dữ liệu theo cách này có thể tối ưu hóa mô hình bằng cách giảm số lượng biến mô hình hoặc cho phép đưa vào các biến kiểm soát. Cho dù bạn đang sử dụng dữ liệu để xây dựng mô hình thị giác máy tính thì việc sử dụng dữ liệu chất lượng cao là ưu tiên hàng đầu.

Ghi nhãn dữ liệu không phải là không có những thách thức của nó. Đặc biệt, một số thách thức phổ biến nhất là:

Đặt tiền tốt thời gian: mặc dù việc ghi nhãn dữ liệu là rất quan trọng đối với các mô hình học máy, nhưng nó có thể tốn kém của về tiền bạc và thời gian. Nếu một doanh nghiệp sử dụng các tiếp cận tự động hơn, các nhóm kỹ sư sẽ vẫn cần thiết lập các đường dẫn dữ liệu trước khi xử lý dữ liệu và việc dán nhãn thủ công hầu như luôn tốn kém và mất thời gian.

Để mắc lỗi do con người: các phương pháp ghi nhãn này cũng có thể xảy ra lỗi do con người, điều này có thể làm giảm chất lượng của dữ liệu. Điều này dẫn đến việc xử lý và lập mô hình dữ liệu không chính xác. Kiểm tra đảm bảo chất lượng là điều cần thiết để duy trì chất lượng dữ liệu.

Các trường hợp sử dụng ghi nhãn dữ liệu: thị giác máy tính: một lĩnh vực AI sử dụng dữ liệu đào tạo để xây dựng mô hình thị giác máy tính cho phép phân đoạn hình ảnh và tự động hóa danh mục, xác định các điểm chính trong hình ảnh và phát hiện vị trí của các đối tượng.

Dữ liệu được gán nhãn không những cung cấp cho quá trình đào tạo mô hình của các thuật toán ML mà nó còn là công cụ để đánh giá độ hiệu quả của các mô hình đó cũng như các thuật toán tự động có độ chính xác như thế nào so với các dữ liệu được con người gán nhãn bằng tay.

Hiện nay, vết nứt vẫn luôn là một vấn đề nhức nhối trong lĩnh vực xử lý ảnh. Vết nứt thì tồn tại muôn hình vạn trạng từ đơn giản đến phức tạp. Nhiều vết nứt phức tạp sẽ tạo sự khó khăn trong việc nghiên cứu và phân tích, đó là một vấn đề cản trở hiệu quả và sự phát triển của các thuật toán liên quan đến vết nứt sau này. Nên cần phải phân tách các vết nứt phức tạp thành vết nứt đơn để dễ dàng sử dụng cho các ứng dụng của ML, DL. Những vết nứt có cấu trúc phức tạp là do có nhiều vết nứt đơn xuất hiện chồng chéo nhau tạo nên nhiều giao điểm. Để tách được chúng ra thành các vết nứt đơn cần phải xác định được vị trí của những giao điểm đó. Các thuật toán tìm điểm giao như đã nói ở Chương 1 sẽ giải quyết những vấn đề này. Tuy nhiên những thuật toán này đa số không phải được sử dụng dành riêng cho vết nứt nên vẫn còn tồn tại rất nhiều hạn chế và sai sót như điểm giao tìm được chưa đúng tọa độ, chưa tìm được đầy đủ điểm giao... Chính vì vậy cần gán nhãn dữ liệu các điểm giao của vết nứt để đánh giá tính đúng đắn của các thuật toán phát hiện điểm giao, từ đó có những biện pháp để cải thiện các thuật toán đó. Ngoài ra việc gán nhãn dữ liệu điểm giao cũng có một vai trò rất quan trọng trong việc cung cấp dữ liệu chính xác cho các mô hình học máy, học sâu.

Theo hiểu biết của tôi, hiện nay không có phần mềm hay giao diện nào được thiết kế để phục vụ cụ thể cho mục đích gán nhãn dữ liệu điểm giao này.

2.2. Quy trình xử lý dữ liệu, xuất dữ liệu của GUI

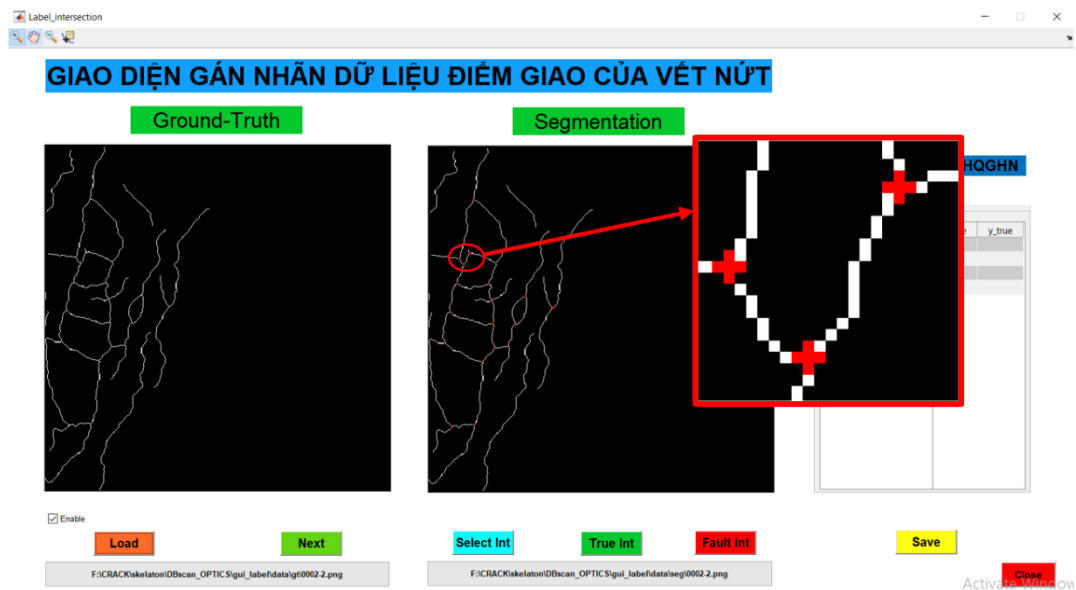
2.2.1. Quá trình chuẩn bị dữ liệu

Trước khi sử dụng giao diện để gán nhãn dữ liệu cần một số điều cần phải chuẩn bị trước để cho quá trình ghi nhãn thuận tiện và dễ dàng hơn. Dữ liệu cần chuẩn bị bao gồm 3 tập, thứ nhất là tập ảnh ‘gt’ chứa những ảnh vết nứt được nhận diện bởi con người với vết nứt là những pixel màu trắng, phần còn lại màu đen là các pixel nền, kích thước, định dạng tùy thuộc vào tập dữ liệu mà mình có. Thứ hai là tập ảnh ‘seg’ cũng chứa những ảnh vết nứt được nhận diện bởi con người tương ứng với tập ‘gt’, tuy nhiên nó đã trải qua một quá trình xử lý, đó là các điểm giao đã được tìm thông qua một thuật toán tìm giao ở bên trên. Ở đây tôi đã sử dụng thuật toán Crossing Number vì nó có hiệu suất tốt nhất cũng như được sử dụng trên dữ liệu vết nứt có độ rộng 1 pixel. Các điểm giao được phát hiện trước đó đã được ký hiệu là một pixel màu đỏ, sở dĩ có tập dữ liệu này là để việc gán nhãn được nhanh chóng, chính xác và đầy đủ hơn. Bởi vì trong quá trình ghi nhãn, đối với những bức ảnh có cấu trúc phức tạp, mật độ điểm giao dày đặc nên việc bỏ lỡ hoặc ghi lại điểm giao là việc rất dễ xảy ra. Có một điều cần chú ý là những cặp ảnh tương đương nhau ở hai tập dữ liệu trên phải cùng tên và cùng định dạng. Cuối cùng là tập dữ liệu ‘txt’, đây là tập chứa những file có định dạng txt, trong các file đó là tập hợp các tọa độ điểm giao đối với từng ảnh khi được gán nhãn xong.

2.2.2. Quá trình xử lý dữ liệu

Sau khi chuẩn bị xong tất cả các dữ liệu cần thiết, quá trình ghi nhãn dữ liệu sẽ được bắt đầu. Với đầu vào là ảnh Ground-Truth được chuẩn bị trong tập ‘gt’ sẽ được hiển thị ở cửa sổ Ground-Truth. Khi chọn ảnh từ tập ‘gt’ thì một ảnh tương ứng ở tập ‘seg’ cũng sẽ tự động được hiển thị ở cửa sổ Segmentation. Ảnh được chọn bằng cách kích chuột vào nút nhấn ‘Load’ sau đó cửa sổ chứa các tập ảnh sẽ hiện ra và khi đó chỉ cần chọn ảnh mà người dùng muốn. Nếu cần gán nhãn cả tập dữ liệu thì chúng ta nên chọn ảnh đầu tiên trước vì sau khi nhấn nút ‘Next’ ảnh liền ngay sau ảnh đầu tiên sẽ được chọn và cứ như thế cho đến khi hết tập dữ liệu. Điều này giúp người dùng không bị bỏ lỡ ảnh nào trong quá trình gán nhãn. Trước khi tải ảnh lên nếu không kích chuột vào ô ‘Enable’ thì ảnh Segmentation sẽ không được hiển thị trong cửa sổ ‘Segmentation’. Như đã nói ở trên, cửa sổ Segmantation sẽ hiển thị ảnh mà điểm giao đã được nhận diện bằng

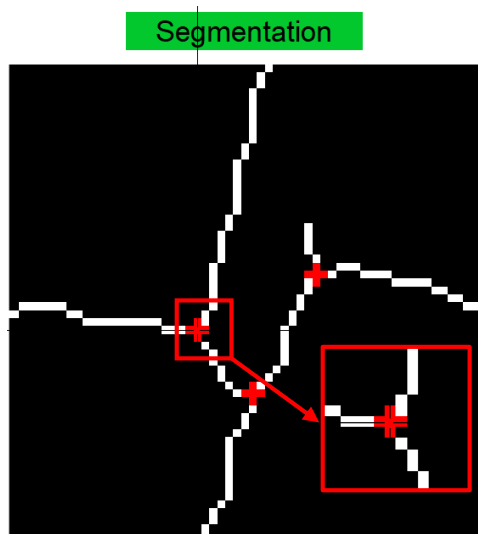
các dấu chấm đỏ như ảnh dưới đây. Bên dưới mỗi cửa sổ sẽ là địa chỉ của ảnh đang được hiển thị ở cửa sổ đó



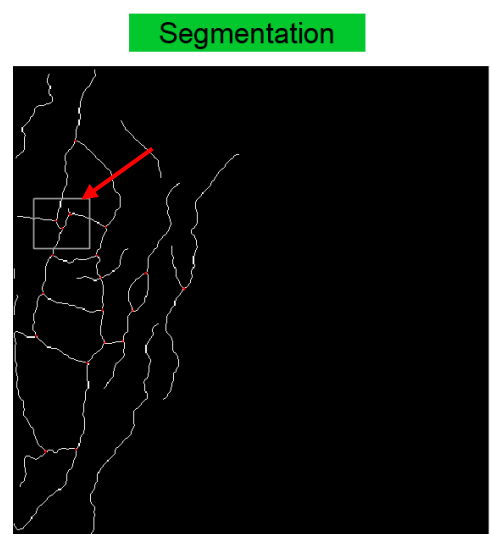
Hình 2. 2: Giao diện chính của GUI

Khi ảnh đã được chọn xong thì khi đó con trỏ chuột luôn ở trạng thái có thể phóng to, chỉ cần kéo thả chuột vào vị trí có điểm giao thì vị trí đó sẽ được phóng to lên đến khi nào nó đủ lớn để có thể thuận tiện cho việc gán nhãn thì dừng lại. Để gán nhãn hãy kích chuột vào nút nhấn ‘Select Int’, con trỏ sẽ ở chế độ sẵn sàng gán nhãn sau đó chỉ cần đưa con trỏ đến vị trí được coi là điểm giao và kích chuột.

Kích chuột vào điểm giao

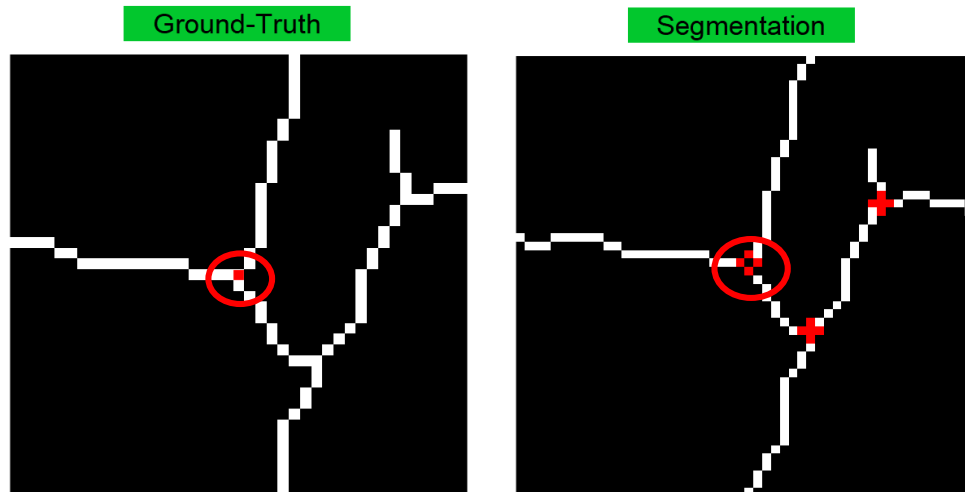


Phóng to vị trí có điểm giao



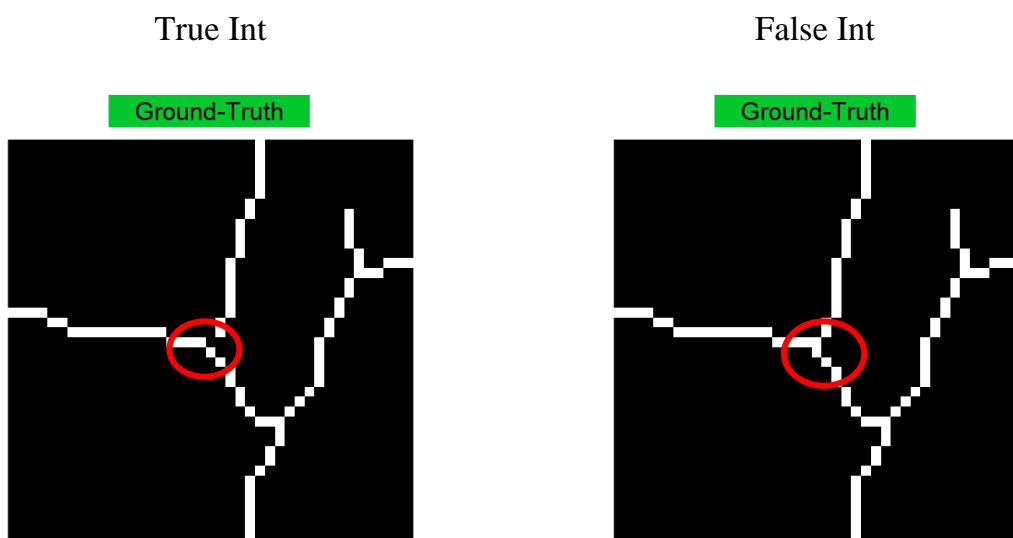
Hình 2. 3: Phóng to và chọn điểm giao

Khi đó điểm giao vừa được chọn bên cửa sổ ‘Segmentation’ sẽ được chuyển từ màu đỏ sang màu đen, đánh dấu rằng vị trí đó đã được gán nhãn, tránh việc gán nhãn lại. Bên cửa sổ Ground-Truth vị trí đó cũng được tự động phóng to lên và tạm thời vị trí đó sẽ được ký hiệu là một pixel màu đỏ, nó sẽ được đổi màu nếu chúng ta xác nhận nó điểm giao hoặc không phải điểm giao. Được thể hiện như hình vẽ dưới đây.



Hình 2. 4: Điểm giao tạm thời

Sau khi chọn điểm giao, chúng ta có thêm 2 lựa chọn đại diện cho 2 nút nhấn là ‘True Int’ và ‘False Int’. ‘True Int’ là để xác nhận pixel vừa chọn là điểm giao còn ‘False Int’ là để loại điểm vừa chọn. Nếu là ‘True Int’ thì pixel vừa được chọn bên cửa sổ Ground-Truth sẽ bị mất đi hay chuyển từ màu đỏ thành màu đen, cũng là để xem trước vết nứt sau khi được tách ra thành các vết nứt đơn sẽ trông như thế nào. Nếu là ‘False Int’ thì điểm màu đỏ đó sẽ biến mất hay từ màu đỏ thành màu trắng hay chính là đưa vị trí vết nứt đó về trạng thái ban đầu như hình vẽ dưới đây.



Hình 2. 5: Xác nhận lại điểm giao

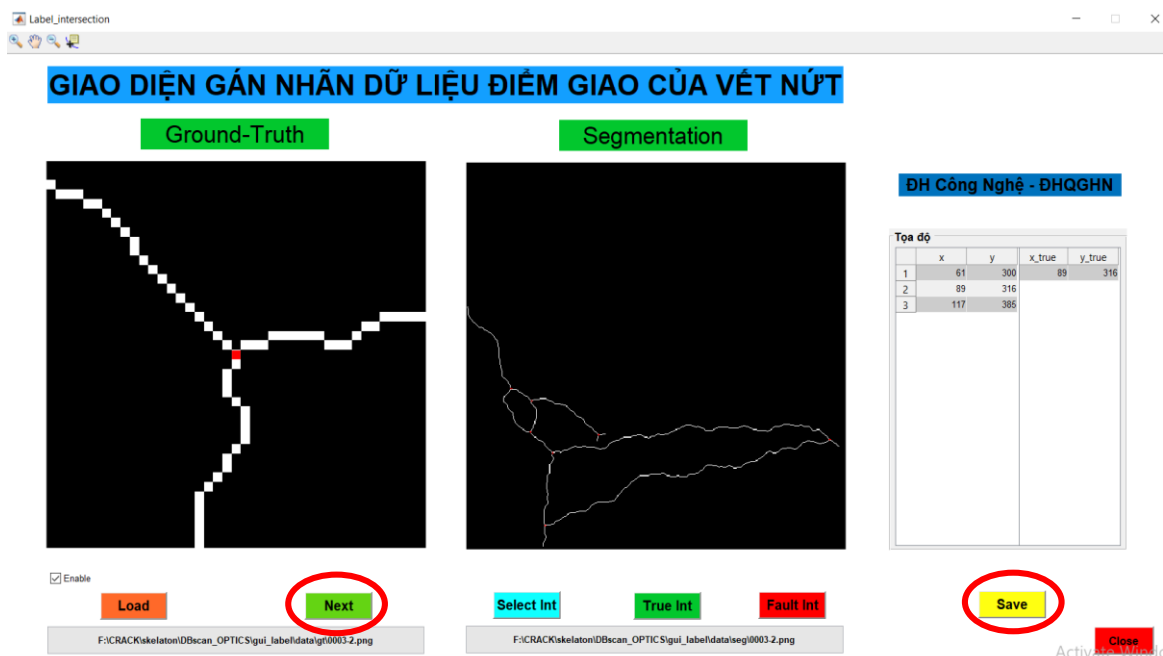
Sau khi xác nhận xong tọa độ được công nhận là điểm giao sẽ được ghi vào file định dạng txt và được hiển thị trong x_true, y_true của cửa sổ ‘Tọa độ’. Ngược lại nếu không phải điểm giao sẽ lập tức bị loại và nó chỉ có thể được hiển thị trong tọa độ tạm thời x, y của cửa sổ ‘Tọa độ’ như được thể hiện trong hình 14.

Tọa độ				
	x	y	x_true	y_true
1	47	167	54	175
2	47	178		
3	54	175		

Hình 2. 6: Tọa độ được lưu sau khi gán nhãn

Từ hình 14 ta có thể thấy hàng 1 và 2 là tọa độ của False Int, hàng 3 là tọa độ của True Int thì nó được xuất hiện trong x_true, y_true.

Sau một quá trình ghi nhãn dữ liệu, nếu điểm giao trong một ảnh đã được gán nhãn hết hoàn toàn người dùng tiếp tục kích chuột vào nút ‘Save’, toàn bộ tọa độ x_true, y_true sẽ được ghi vào file định dạng txt. Để chuyển sang gán nhãn cho ảnh tiếp theo người dùng hãy kích vào ‘Next’. Nhưng nên lưu ý, trước khi kích vào ‘Next’ hãy chắc chắn rằng chúng ta đã lưu dữ liệu của ảnh trước đó.

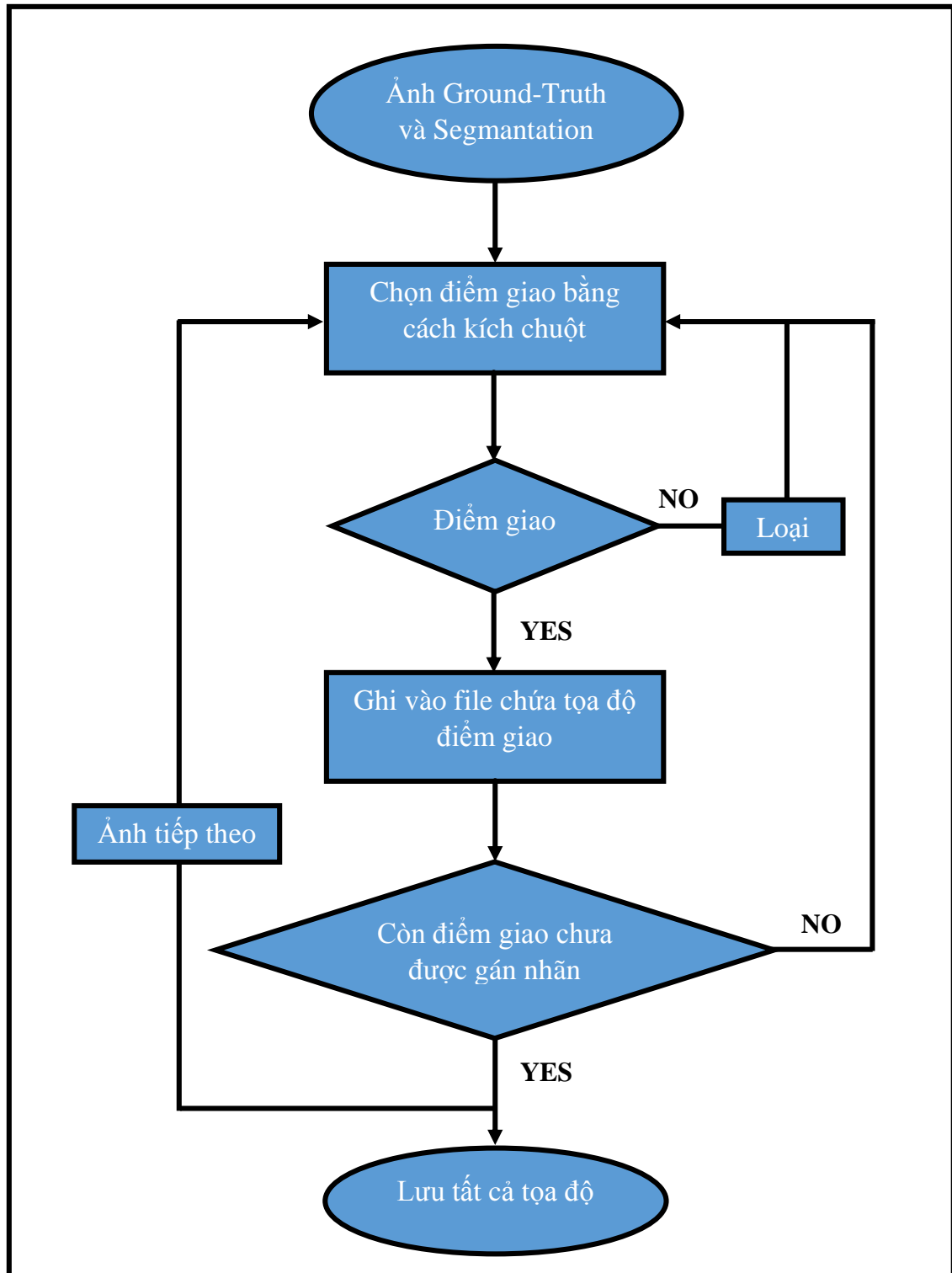


Hình 2. 7: Lưu dữ liệu và chuyển qua ảnh tiếp theo

Đây là một giao diện đơn giản, dễ dàng sử dụng, phù với mọi người kể cả những người chưa có nhiều khái niệm về gán nhãn dữ liệu nói riêng và xử lý ảnh nói chung.

Người dùng chỉ cần sử dụng chuột: kích chuột, kéo thả chuột, là hoàn toàn có thể gán nhãn dữ liệu mà mình mong muốn.

Dưới đây là lưu đồ thuật toán của quá trình gán nhãn dữ liệu.



Hình 2. 8: Lưu đồ thuật toán

CHƯƠNG 3

DỮ LIỆU VÀ CÁC THÔNG SỐ KIỂM THỬ

3.1. Các tập dữ liệu

3.1.1. Tập dữ liệu CrackTree200

Tập dữ liệu CrackTree200 chứa 206 hình ảnh vỉa hè trên nền đường nhựa với kích thước cố định là 800x600 pixel. Vết nứt có độ rộng nhỏ và dài, cấu trúc từ đơn giản đến phức tạp. Tập dữ liệu bao gồm tập ảnh Ground-truth với vết nứt là các pixel màu trắng, phần màu đen là nền, độ rộng vết nứt đều được để ở độ rộng 1 pixel.

Ảnh gốc



Ảnh Ground-Truth

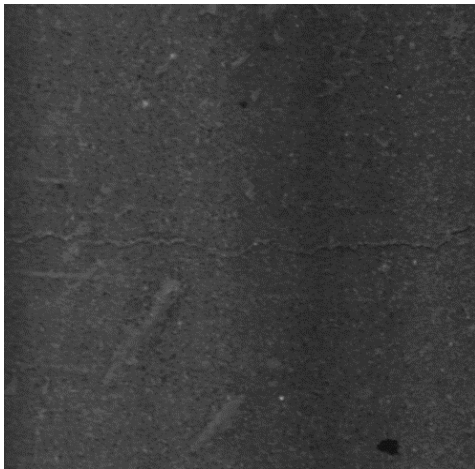


Hình 3. 1: Tập ảnh CrackTree200

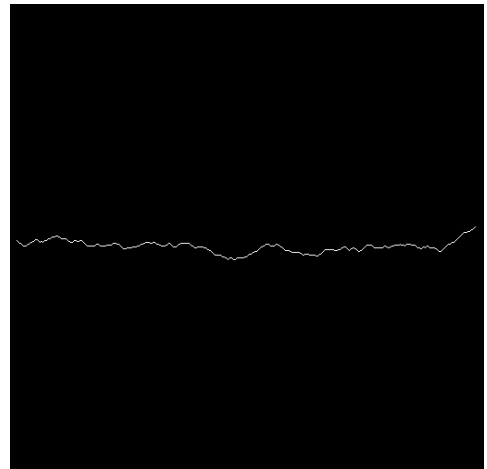
3.1.2. Tập dữ liệu CRKWH100

CRKWH100 bao gồm 100 hình ảnh thang xám hình ảnh mặt đường có kích thước 512x512 pixel được chụp bởi một máy ảnh line-array dưới ánh sáng có thể nhìn thấy được. Vết nứt của tập dữ liệu có cấu trúc từ mỏng đến dày, từ nông đến sâu, vết nứt phân bố từ đơn giản đến phức tạp. Tập dữ liệu cũng bao gồm tập ảnh Ground-truth chứa vết nứt có độ rộng 1 pixel.

Ảnh gốc



Ảnh Ground-Truth

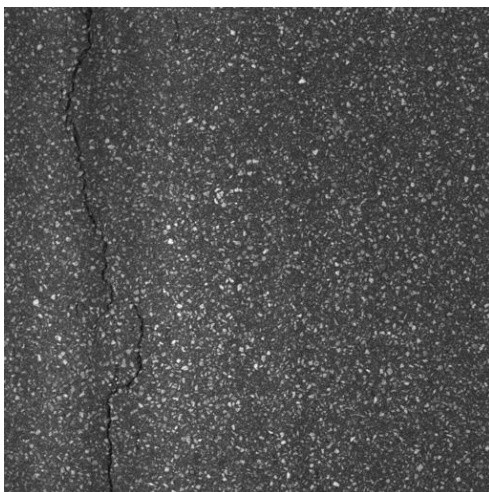


Hình 3. 2: Tập ảnh CRKWH100

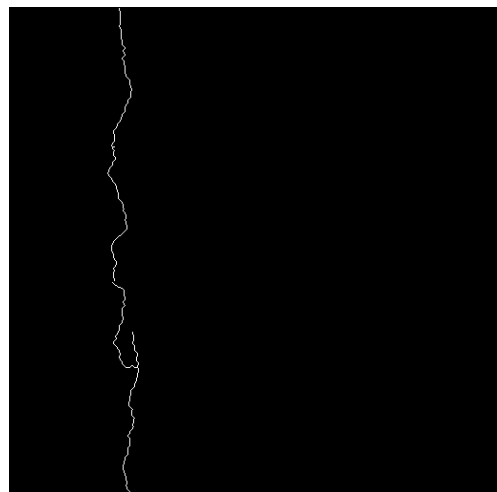
3.1.3. Tập dữ liệu CrackLS315

CrackLS315 chứa 315 hình ảnh mặt đường được chụp dưới ánh sáng laser, được chụp bởi một máy ảnh line – array và có kích thước 512x512 pixel. Tập dữ liệu chứa những vết nứt có cấu trúc đa phần là mỏng, nhỏ từ nông đến sâu, độ phức tạp đa số là đơn giản. Tập ảnh cũng chứa các ảnh Ground-truth có độ rộng là 1 pixel.

Ảnh gốc



Ảnh Ground-Truth



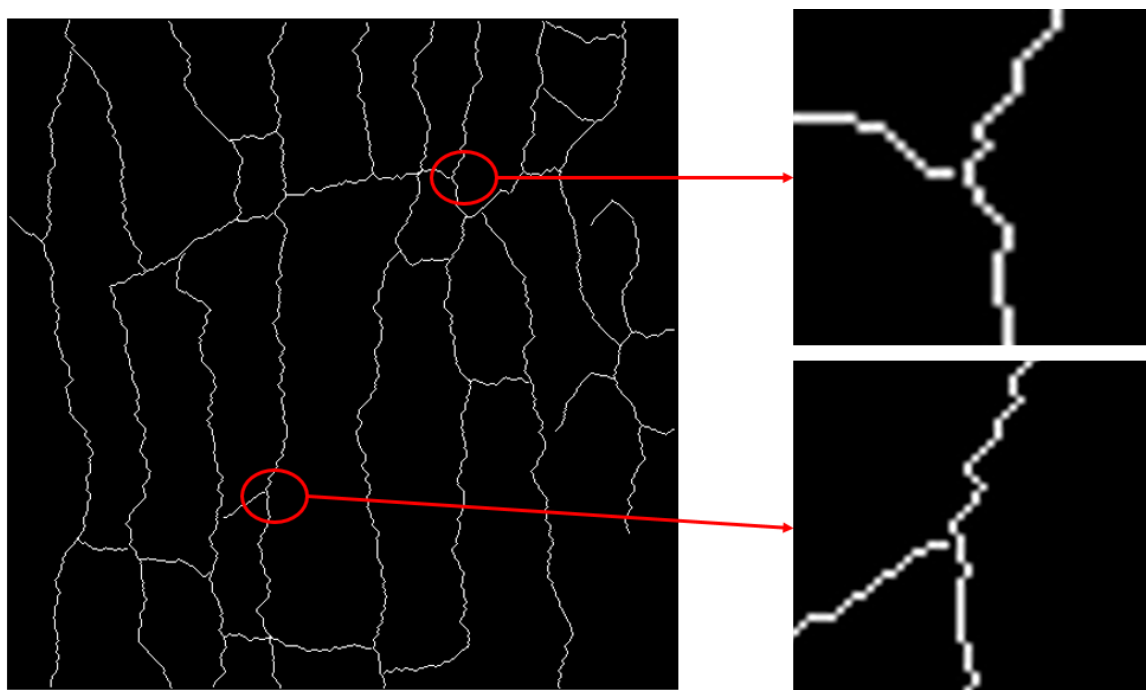
Hình 3. 3: Tập ảnh CrackLS315

3.1.4. Dữ liệu được sử dụng

Từ 3 tập dữ liệu trên chọn ra 86 ảnh vết nứt có cấu trúc phức tạp để tiến hành gán nhãn và tìm điểm giao bằng phương pháp tự động. Bao gồm 15 ảnh từ CRKWH100, 32 ảnh từ CrackLS315 và 39 ảnh từ CrackTree200. Trong đó có tổng số 982 điểm giao được gán nhãn và như vậy trung bình mỗi ảnh có khoảng 12 điểm giao.

3.1.5. Xử lý dữ liệu

Do có nhiều ảnh vết nứt, ở một vị trí cụ thể bằng mắt thường có thể nhìn thấy đó là một điểm giao nhưng thực tế lại không phải hoàn toàn như vậy. Giữa chúng vẫn tồn tại một khoảng cách rất nhỏ, thường là 1 đến 3 pixel như hình 21. Điều đó khiến cho người dùng bị lầm tưởng và ảnh hưởng đến kết quả nhận diện điểm giao cũng như là gán nhãn điểm giao. Để lấp đầy những khoảng trống nhỏ đó chúng ta có thể áp dụng phép biến đổi hình thái cơ bản Dialate (giãn nở).



Hình 3. 4: Khoảng cách nhỏ giữa các vết nứt đơn

Trong phép giãn nở các đường biên của một đối tượng trong ảnh được cộng thêm các pixel. Dialate giúp nối lại các đối tượng riêng lẻ khi khoảng cách của chúng là nhỏ, không liên mạch trên ảnh, giúp những điểm giao thực sự là điểm giao khi được phóng to. Điều đó vừa giúp dễ dàng trong việc gán nhãn dữ liệu và trong việc tìm điểm giao bằng phương pháp tự động.

Về mặt toán học, chúng ta có thể biểu diễn phép toán này theo cách sau:

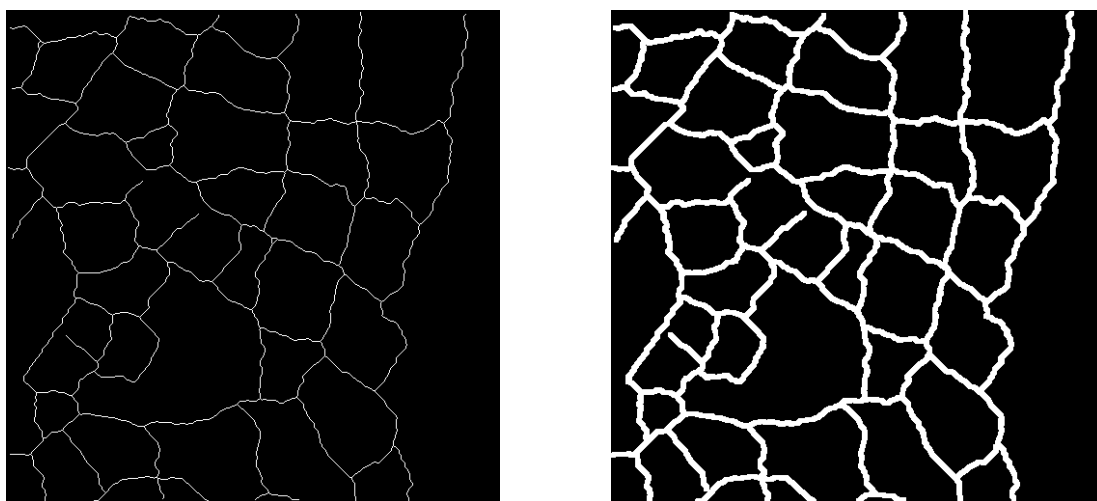
ở đây: A: Hình ảnh đầu vào được mã hóa

B: Cấu trúc phần tử hạt nhân

Một hình ảnh nhị phân điển hình bao gồm pixel 1 (255) và pixel 0. Nhân có thể là một tập hợp cơ của hình ảnh đầu vào hoặc không phải tập hợp con của hình ảnh đầu vào. Để nghĩ về điều này một cách toán học về ma trận, chúng ta có thể có:

Yếu tố cấu trúc: kích thước và hình dạng của phần tử cấu trúc xác định số lượng pixel nên được thêm vào hoặc xóa khỏi các đối tượng trong hình ảnh. Nó là một ma trận của 1 và 0. Pixel trung tâm của hình ảnh được gọi là điểm gốc. Nó chứa một hình ảnh A với một nhân (B), có thể có bất kỳ hình dạng hoặc kích thước nào, thường là hình vuông hoặc hình tròn. Ở đây hạt nhân B có một điểm neo xác định. Nó là trung tâm của hạt nhân. Trong bước tiếp theo, hạt nhân được phủ lên hình ảnh để tính toán các giá trị pixel tối đa. Khi tính toán hoàn tất, hình ảnh được thay thế bằng một mô neo ở trung tâm. Các vùng sáng hơn tăng kích thước làm tăng kích thước hình ảnh.

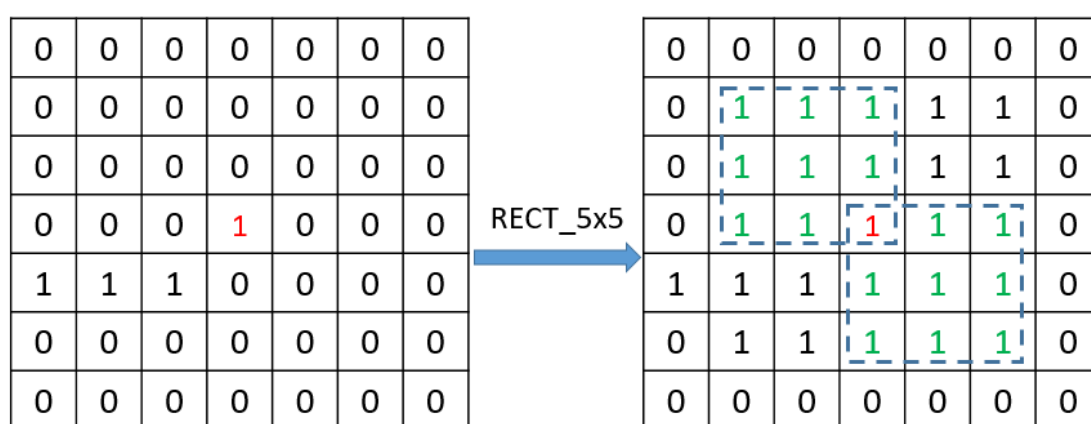
Ví dụ, kích thước của đối tượng tăng lên trong bóng râm màu trắng, kích thước của một đối tượng có màu đen sẽ tự động giảm xuống.



Hình 3. 5: Quá trình dẫn nở vết nứt

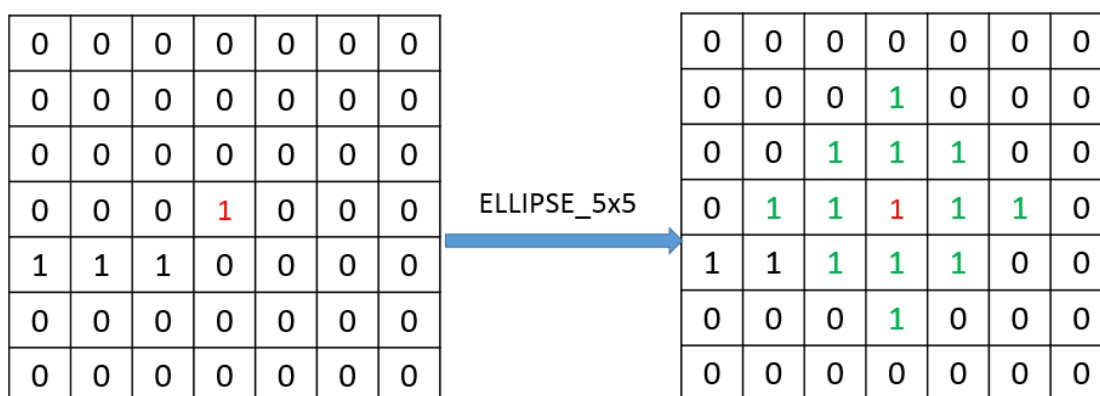
Vì vậy làm thế nào chúng ta có thể thu nhỏ hoặc mở rộng hình ảnh? chúng ta chỉ cần thực hiện phép tích chập trên hình ảnh đầu vào của chúng ta với hạt nhân. Nhân có thể có bất kỳ hình dạng nào, nhưng thông thường, nó là hình vuông. Một điều quan trọng cần nhớ là hạt nhân được định nghĩa đối với điểm neo thường được đặt ở điểm ảnh trung tâm.

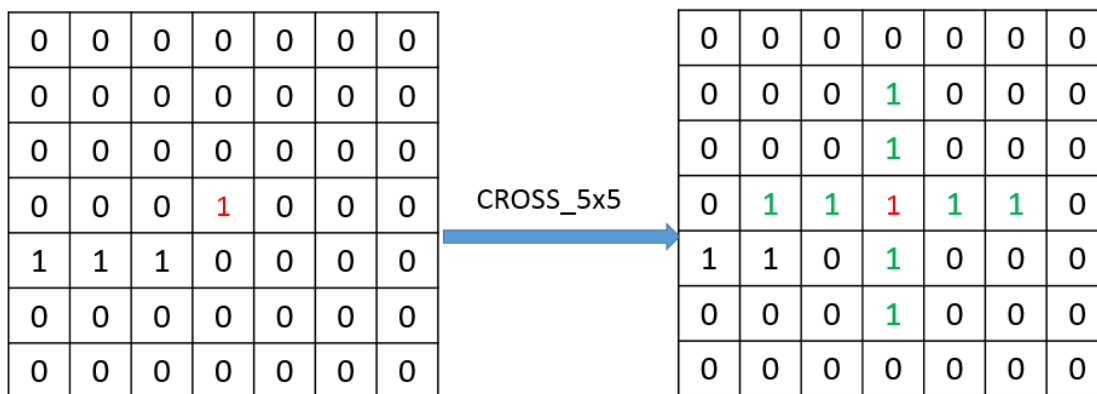
Trong quá trình là dẫn nở, chúng ta sẽ có thêm một số khái niệm về nhân, bao gồm: nhân hình chữ nhật, nhân chữ thập, nhân hình thoi. Tùy thuộc vào mỗi ứng dụng chúng ta có thể sử dụng nhân khác nhau ứng với các ma trận có kích thước 3x3, 5x5, 7x7.



Hình 3. 6: Dẫn nở vết nứt theo hình chữ nhật

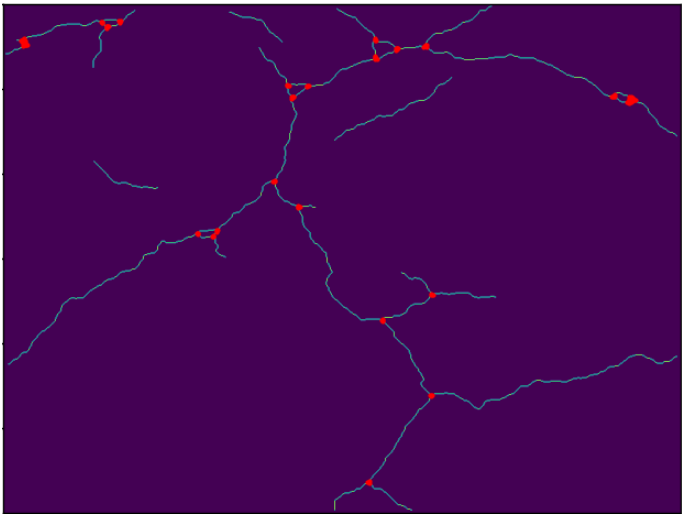
Hình trên thể hiện kết quả khi chúng ta sử dụng nhân hình chữ nhật có kích thước 5x5. Với pixel trung tâm có giá trị là 1, nó sẽ được giãn nở ra các pixel xung quanh nó với kích thước giãn nở tối đa là $\sqrt{8}$ pixel đối với nhân 5x5. Tương tự nếu nhân là 3x3 thì kích thước giãn nở sẽ là $\sqrt{2}$ pixel, nếu nhân là 7x7 thì kích thước giãn nở sẽ là ... Ngoài ra chúng ta có thể sử dụng nhân hình thoi hay hình chữ thập như hình dưới đây:

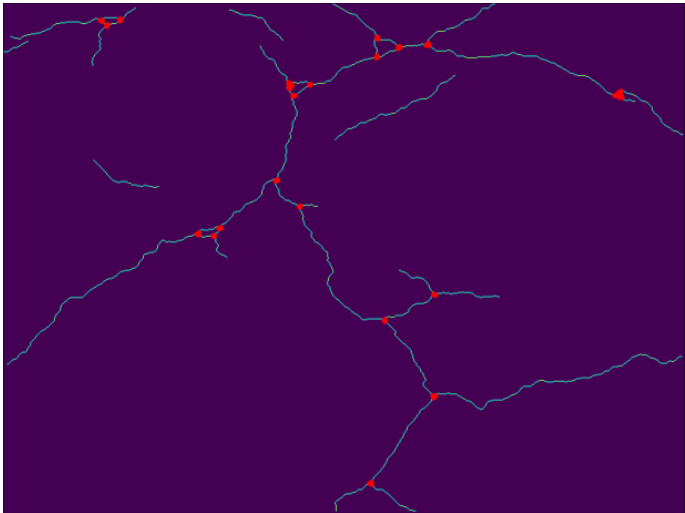
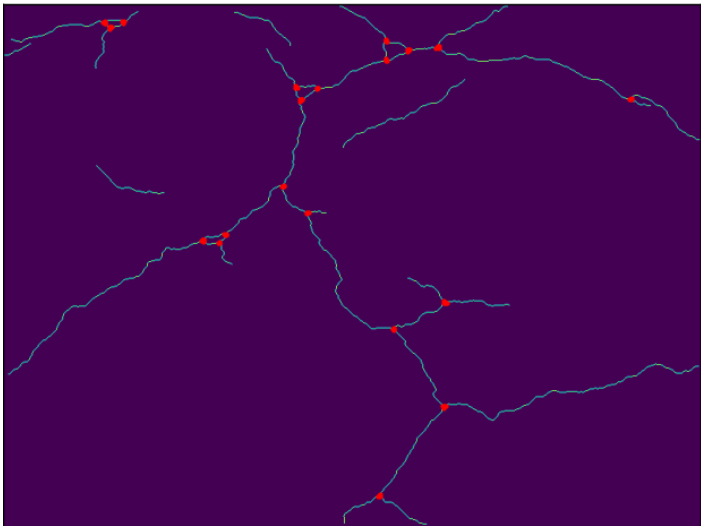




Hình 3. 7: dẫn nở vết nứt theo hình thoi và hình chữ thập

Do những khoảng trống giữa các đối tượng vết nứt trong có độ lớn giao động từ 1-3 pixel nên chúng ta sẽ sử dụng nhân là ma trận có kích thước 5x5.

Hình dáng phần tử cấu trúc	Kết quả	Số điểm giao
Hình chữ nhật (5x5)		28

Hình Elip (5x5)		24
Hình chữ thập (5x5)		20

Hình 3. 8: Điểm giao tìm được từ các nhân khác nhau

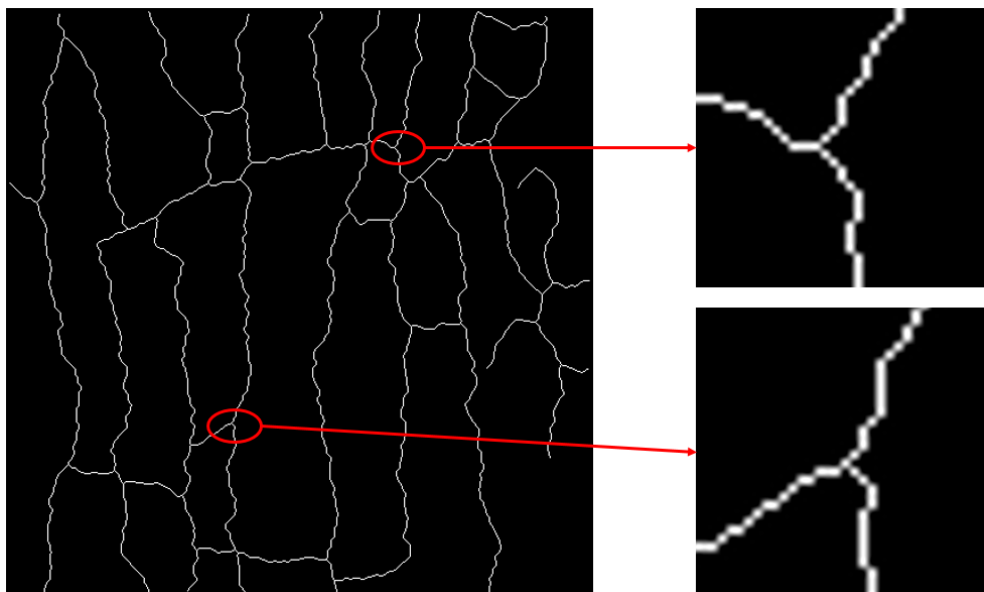
Từ hình trên cho thấy, hình dáng phần tử cấu trúc hình chữ nhật có thể giúp phát hiện được nhiều điểm giao hơn (28 điểm giao) so với hai hình dạng cấu trúc còn lại (20 đối với hình chữ thập và 24 đối với hình elip). Chính vì vậy trong đồ án tốt nghiệp này hình chữ nhật có kích thước 5x5 được sử dụng để nối các phần tử gần nhau trong ảnh vết nứt.

Sau quá trình giãn nở để nối các đối tượng với nhau thì độ rộng của vết nứt cũng từ đó mà tăng lên. Để phục vụ cho quá trình gán nhãn dữ liệu và quá trình phát hiện điểm giao tự động đối với những thuật toán chỉ dùng cho vết nứt có độ rộng là 1 pixel, những vết nứt được giãn nở đó phải được đưa về dạng có độ rộng 1 pixel. Để làm điều đó chúng ta sẽ sử dụng đối số ‘skel’ như đã nói ở mục 1.2.1 hoặc sử dụng chức năng ‘Skeletonize’. Skeletonize làm giảm các đối tượng nhị

phân thành các biểu diễn rộng 1 pixel. Điều này có thể hữu ích cho việc trích xuất đối tượng hoặc đại diện cho cấu trúc liên kết của một đối tượng. Skeletonize hoạt động bằng cách thực hiện các chuyển tiếp liên tiếp của hình ảnh. Trên mỗi đường chuyển, các pixel đường viền được xác định và loại bỏ với điều kiện chúng không phá vỡ khả năng kết nối của đối tượng tương ứng.

Skeletonize [14] hoạt động bằng cách thực hiện các bước liên tiếp của hình ảnh, loại bỏ các pixel trên đường viền đối tượng. Điều này tiếp tục cho đến khi không thể loại bỏ thêm pixel nào nữa. Hình ảnh tương quan với một mặt nạ gán cho mỗi pixel một số trong phạm vi $[0..255]$ tương ứng với mỗi mẫu có thể có trong số 8 pixel lân cận của nó. Sau đó, một bảng tra cứu được sử dụng để gán giá trị 0, 1, 2 hoặc 3 cho các pixel, những giá trị này sẽ bị loại bỏ một cách có chọn lọc trong các lần lặp lại.

Skeletonize [14] sử dụng cấu trúc dữ liệu octree để kiểm tra vùng lân cận $3 \times 3 \times 3$ của một pixel. Thuật toán tiến hành bằng cách quét lặp đi lặp lại hình ảnh và loại bỏ các pixel ở mỗi lần lặp cho đến khi hình ảnh ngừng thay đổi. Mỗi lần lặp lại bao gồm hai bước: đầu tiên, một danh sách các ứng cử viên để loại bỏ được tập hợp; sau đó các pixel từ danh sách này được kiểm tra lại tuần tự, để duy trì khả năng kết nối của hình ảnh tốt hơn. Phương pháp [] được thiết kế để sử dụng trên hình ảnh 3-D và được chọn tự động cho những hình ảnh đó. Kết quả cuối cùng của quá trình này là:



Hình 3. 9: Vết nứt sau khi lấp đầy khoảng trống

Những khoảng trống khi trước đã được lấp đầy sau 2 quá trình Dilate và Skeletonize.

3.2. Các thông số đánh giá

Một điểm giao được tìm bằng phương pháp tự động là đúng nếu dữ liệu được gán nhãn xác định một điểm giao cùng loại trong vùng lân cận của nó.

3.2.1. Các thông số đánh giá dành cho vết nứt khác 1 pixel

Đây là các thông số đánh giá hiệu suất của các thuật toán sử dụng hình ảnh có độ rộng khác 1 pixel, vì điểm giao chúng tìm được nó tồn tại dưới nhiều dạng và không thể đánh giá bằng các phương pháp thông thường. Chính vì vậy các thông số được đề xuất trong [12] được sử dụng trong phần này của đề án.

- Index of minutiae revelation ($I_{Revealed}$) là tỷ số giữa các điểm giao đúng vị trí được xác định bằng phương pháp tự động ($N_{CorrectAuto}$) với các điểm giao đúng vị trí được trích xuất bởi con người (N_{Man}) hay chính là dữ liệu được gán nhãn trước đó.

$$I_{Revealed} = \frac{N_{CorrectAuto}}{N_{Man}} \quad 3.2$$

- Index of positional correctness ($I_{PositionCorrect}$) là tỷ số giữa các điểm giao đúng vị trí được xác định bằng phương pháp tự động ($N_{CorrectAuto}$) và các điểm giao được trích xuất bằng phương pháp tự động (N_{Auto}).

$$I_{PositionCorrect} = \frac{N_{CorrectAuto}}{N_{Auto}} \quad 3.3$$

- Index of correctness ($I_{Correct}$) là tỷ lệ giữa hiệu các điểm giao đúng vị trí ($N_{CorrectAuto}$) và sai vị trí ($N_{Exchanged}$) với các điểm giao được trích xuất bằng phương pháp tự động (N_{Auto}).

$$I_{Correct} = \frac{N_{CorrectAuto} - N_{Exchanged}}{N_{Auto}} \quad 3.4$$

3.2.2. Các thông số đánh giá cho vết nứt có độ rộng 1 pixel

Điều quan trọng là phải đánh giá hiệu suất của các thuật toán phân loại để sử dụng một cách đáng tin cậy các thuật toán này trong sản xuất để giải quyết các vấn đề trong thế giới thực. Các thước đo hiệu suất trong các mô hình phân loại học máy được sử dụng để đánh giá mức độ hoạt động của các mô hình phân loại học máy trong một ngữ cảnh nhất định. Các chỉ số hiệu suất này bao gồm: Accuracy, Precision, recall, F1-score [15]. Bởi vì nó giúp chúng ta hiểu được điểm mạnh và hạn chế của các thuật toán này khi đưa ra dự đoán trong các tình huống mới, hiệu suất của mô hình là điều cần thiết cho việc học máy. Trước tiên chúng ta hãy tìm hiểu những thuật ngữ: True Positive, False Positive, True Negative, False Negative.

		PREDICTED	
		Positive	Negative
ACTUAL	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Hình 3. 10: Phân loại các thông số đánh giá

- *True Positive (TP)*: đo lường mức độ mà thuật toán dự đoán đúng về lớp positive. Nghĩa là, thuật toán dự đoán rằng trường hợp là positive và trường hợp thực sự là positive. Những positive thực sự có liên quan khi chúng ta muốn biết có bao nhiêu mặt positive mà thuật toán của chúng ta dự đoán chính xác. Ví dụ trong một bài toán phân loại nhị phân với các lớp “A” và “B”, nếu mục tiêu của chúng ta là dự đoán chính xác lớp “A”, thì true positive sẽ là số trường hợp của lớp “A” mà thuật toán đã dự đoán chính xác như lớp “A”. Lấy một ví dụ trong thế giới thực, nếu mô hình được thiết kế để dự đoán liệu một email có phải là thư rác hay không, thì một kết quả true positive sẽ xảy ra khi thuật toán dự đoán chính xác rằng một email là thư rác. Tỷ lệ true positive là tỷ lệ phần trăm của tất cả các trường hợp được phân loại chính xác là thuộc một lớp nhất định. Các true positive rất quan trọng vì chúng cho biết thuật toán của chúng ta hoạt động tốt như thế nào trên các trường hợp positive.

- *False Positive (FP)*: Kết quả false positive xảy ra khi thuật toán dự đoán rằng một đối tượng thuộc về một lớp mà nó thực sự không thuộc về lớp đó. False Positive có thể là vấn đề vì chúng có thể dẫn đến việc ra quyết định không chính xác. Ví dụ, nếu một thuật toán chuẩn đoán y tế có tỷ lệ false positive cao, nó có thể dẫn đến việc bệnh nhân phải điều trị không cần thiết. Kết quả false positive có thể gây bất lợi cho các thuật toán phân loại vì chúng làm giảm độ chính xác tổng thể của thuật toán. Có một số cách để đo lường kết quả false positive, bao gồm cả tỷ lệ false positive. Tỷ lệ false positive là tỷ lệ của tất cả các ví dụ negative được dự đoán là positive. Mặc dù kết quả false positive có vẻ như có hại cho thuật toán, nhưng trong một số trường hợp, chúng có thể là mong muốn. Ví dụ, trong các ứng dụng y tế, thông thường tốt hơn là bạn nên thận trọng và có một vài kết quả false positive hơn là bỏ lỡ hoàn toàn một chuẩn đoán. Tuy nhiên, trong các ứng dụng khác chẳng hạn như lọc thư rác, kết quả false positive có thể rất tốn kém. Vì vậy, điều quan trọng là phải xem xét cẩn thận những đánh đổi liên quan khi lựa chọn giữa các thuật toán phân loại khác nhau.

- *True Negative (TN)*: True negative là kết quả mà thuật toán dự đoán chính xác là negative. Ví dụ, nếu thuật toán đang dự đoán một người có mắc bệnh hay không, thì giá trị true negative sẽ là khi thuật toán dự đoán rằng người đó không mắc bệnh và họ thực sự không mắc bệnh. True negative là một trong những thước đo được sử dụng để đánh giá thuật toán phân loại đang hoạt động tốt như thế nào. Nói chung, một số lượng true negative cao cho thấy rằng mô hình đang hoạt động tốt. True negative được sử dụng cùng với false negative, true positive và false positive để tính toán nhiều loại chỉ số hiệu suất như accuracy, precision, recall và F1 score. Mặc dù true negative cung cấp thông tin chi tiết có giá trị về hiệu suất của thuật toán phân loại, nó nên được giải thích trong ngữ cảnh của các số liệu khác để có được bức tranh toàn cảnh về độ chính xác của mô hình.

- *False Negative (FN)*: False negative xảy ra khi một thuật toán dự đoán một trường hợp là negative khi nó thực sự là positive. False negative có thể rất tốn kém, đặc biệt là trong lĩnh vực y học. Ví dụ, nếu xét nghiệm tầm soát ung thư dự đoán rằng một bệnh nhân không ung thư khi họ thực sự bị ung

thư, thì điều này có thể dẫn đến bệnh tiến triển mà không điều trị. False negative cũng có thể xảy ra trong các lĩnh vực khác, chẳng hạn như bảo mật hoặc phát triển gian lận. Trong những trường hợp này, false negative có thể dẫn đến việc ai đó được cấp quyền truy cập hoặc phê duyệt một giao dịch mà lẽ ra không được phép. False negative thường nghiêm hơn false positive và vì vậy điều quan trọng là phải tính đến chúng khi đánh giá hiệu suất của thuật toán phân loại.

- *Precision*: Là thước đo tỷ lệ các nhãn được dự đoán positive có thực sự đúng. Precision còn được gọi là giá trị dự đoán positive. Precision được sử dụng cùng với recall để đánh đổi kết quả false position và false negative. Precision bị ảnh hưởng bởi sự phân bố lớp. Nếu có nhiều mẫu hơn trong nhóm thiểu số, thì precision sẽ thấp hơn. Precision có thể được coi là thước đo độ chính xác hoặc chất lượng. Nếu chúng ta muốn giảm thiểu false negative, chúng ta sẽ chọn một thuật toán có độ chính xác cao. Ngược lại, nếu chúng ta muốn giảm thiểu false positive, chúng ta sẽ chọn một thuật toán có recall cao. Precision chủ yếu được sử dụng khi chúng ta cần dự đoán loại position và có chi phí lớn hơn liên quan đến false positive so với false negative chẳng hạn như trong chuẩn đoán y tế và lọc thư rác. Precision là thước đo hữu ích cho sự thành công của dự đoán khi các lớp rất mất cân bằng.

$$\frac{TP}{FP + TP} \quad 3.5$$

Từ công thức trên, bạn có thể nhận thấy rằng giá trị dương sai sẽ ảnh hưởng đến độ chính xác.

- *Recall*: Thể hiện khả năng dự đoán chính xác các mặt positive thực tế của thuật toán. Điều này không giống như precision đo lường có bao nhiêu dự đoán được thực hiện bởi các thuật toán thực sự là positive trong số tất cả các dự đoán positive được thực hiện. Ví dụ: nếu thuật toán học máy của bạn đang cố gắng xác định các đánh giá positive, thì recall sẽ là phần trăm trong số các đánh giá positive mà mô thuật toán học máy của bạn dự đoán chính xác là positive. Nói cách khác, nó đo lường mức độ tốt của thuật toán học máy của chúng ta trong việc xác định tất cả các positive thực tế trong số tất cả các

positive tồn tại trong một tập dữ liệu. Recall cao thuật toán học máy càng tốt trong việc xác định cả ví dụ positive và negative. Recall còn được gọi là độ nhạy hoặc tỷ lệ true positive. Recall cao cho thấy rằng thuật toán này rất tốt trong việc xác định các ví dụ positive. Ngược lại, recall thấp cho thấy thuật toán không giỏi trong việc xác định các ví dụ positive. Recall thường được sử dụng cùng với các số liệu hiệu suất khác chẳng hạn như precision và accuracy để có bức tranh toàn cảnh về hiệu suất của thuật toán. Về mặt toán học, nó đại diện cho tỷ lệ giữa giá trị true positive trên tổng số true positive và false negative.

$$\frac{TP}{FN + TP} \quad 3.6$$

- *F1-score*: đại diện cho độ chính xác của thuật toán dưới dạng một hàm của precision và recall. F1 là một chỉ số hiệu suất của thuật toán học máy mang lại trọng số bằng nhau cho cả precision và recall để đo lường hiệu suất của nó về độ chính xác, làm cho nó trở thành một thay thế cho các chỉ số độ chính xác. Nó thường được sử dụng như một giá trị duy nhất cung cấp thông tin cấp cao về chất lượng đầu ra của thuật toán. Đây là một thước đo hữu ích của mô hình trong các tình huống mà người ta cố gắng tối ưu hóa precision hoặc recall và kết quả là hiệu suất của thuật toán bị ảnh hưởng.

$$2 * precision \frac{recall}{precision + recall} \quad 3.7$$

- *Accuracy*: là số liệu phân loại hiệu suất của thuật toán học máy được định nghĩa là tỷ lệ giữa số true positive và số true negative cho tất cả các quan sát positive và negative. Nói cách khác, accuracy cho chúng ta biết tần suất chúng ta có thể mong đợi thuật toán học máy của mình dự đoán chính xác một kết quả trong tổng số lần nó đưa ra dự đoán.

$$\frac{TP + TN}{TP + FN + TN + FP} \quad 3.8$$

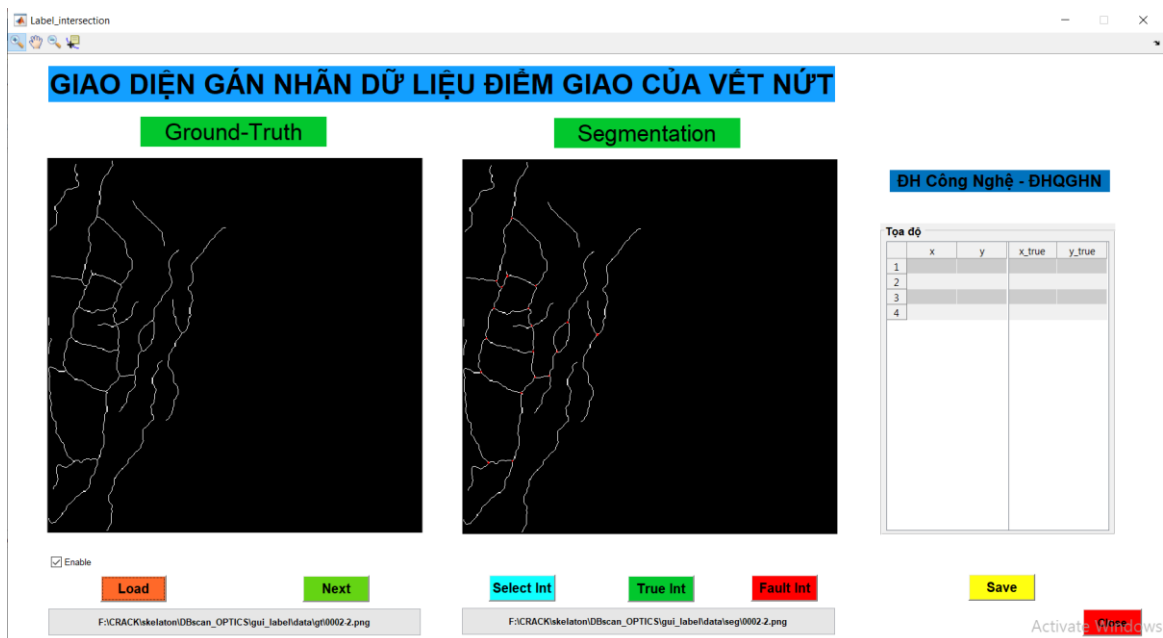
CHƯƠNG 4

KẾT QUẢ THỰC NGHIỆM

4.1. GUI

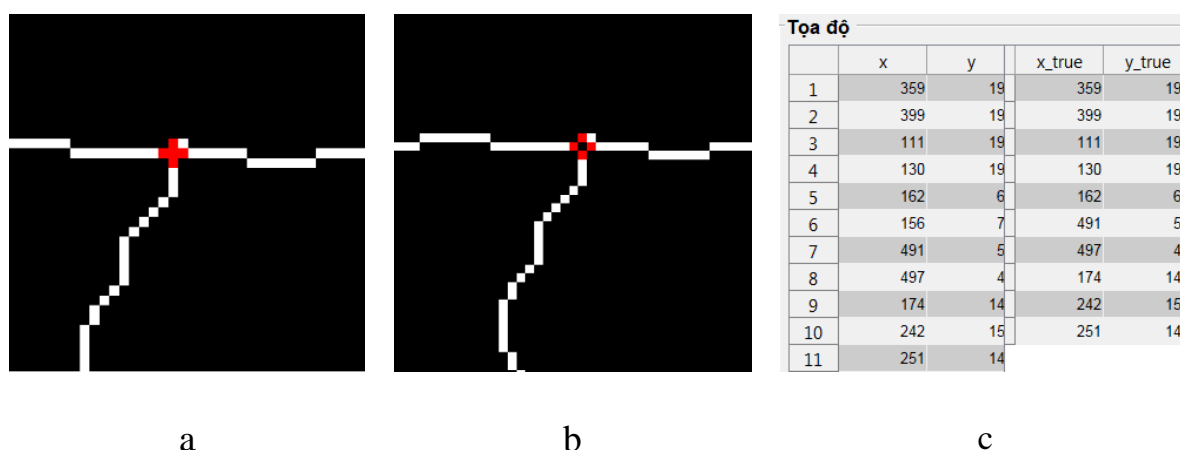
Trong đồ án tốt nghiệp này GUI được sử dụng là một giao diện vô cùng thân thiện với người dùng, từ việc phóng to thu nhỏ đến việc gán nhãn dữ liệu, tất cả đều được hiện một cách đơn giản chỉ với một giao tác là kích chuột hoặc kéo thả chuột. Vẫn chỉ là những thao tác bằng tay nhưng quá trình gán nhãn không quá lâu và thậm chí khi sử dụng GUI sẽ nhanh hơn rất nhiều so với việc không sử dụng nếu muốn có một tập dữ liệu để đánh giá kết quả của các thuật toán nhận diện điểm giao.

Sử dụng GUI đem lại tính trực quan và gần như không có sai sót về độ chính xác. Điểm giao được tìm trực tiếp trên GUI, nếu chọn sai thì vị trí đó cũng sẽ được thể hiện trực tiếp để người dùng có thể nhìn lại và xác nhận đó không phải là điểm giao. Sau mỗi lần chọn điểm giao trên GUI, người dùng đều được nhìn lại vị trí mình vừa chọn có thực sự là điểm giao hay không sau đó là xác nhận lại một lần nữa, điều này nâng cao độ chính xác, tránh việc gán nhãn sai dẫn đến việc phải gán nhãn lại, đặc biệt khiến quá trình đánh giá kết quả không chính xác.



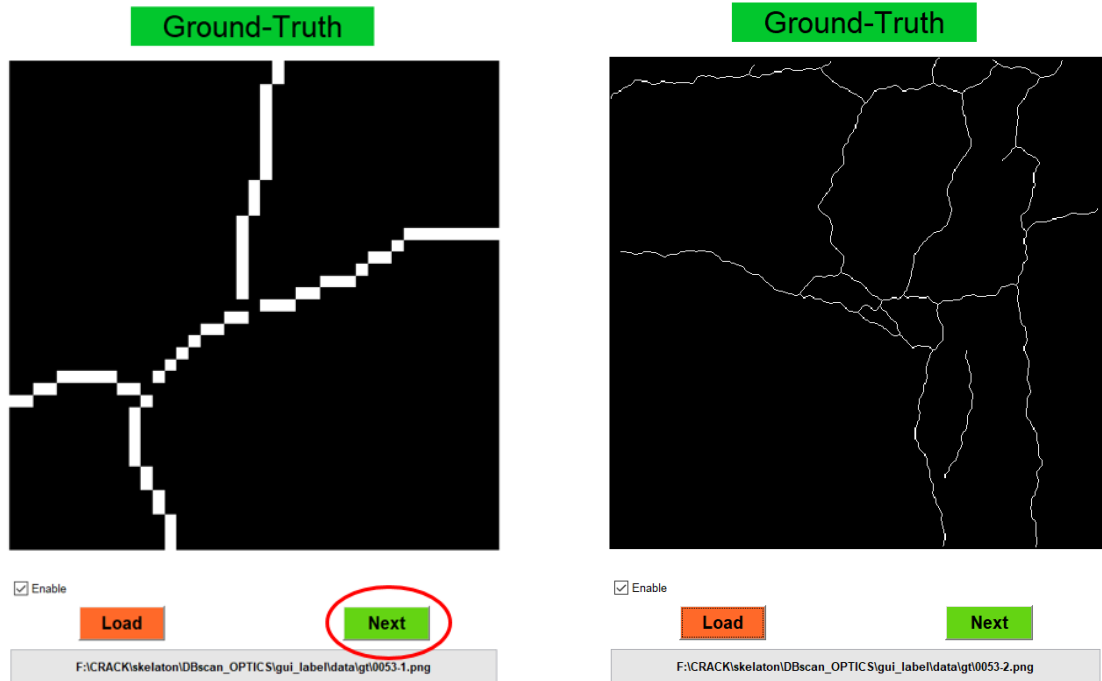
Hình 4. 1: Giao diện trực quan của GUI

Cùng với ảnh được xử lý trước khi đưa vào quá trình gán nhãn như đã nói ở mục 2.2 giúp cho quá trình gán nhãn dữ liệu được rút gọn thời gian đi rất nhiều và tăng cả về độ chính xác (Hình 4.2a). Vì sau mỗi lần chọn điểm giao vị trí đó sẽ được đánh dấu lại để tránh việc gán nhãn lại nhiều lần. Đặc biệt ngay từ đầu, những vị trí được coi là điểm giao đã được thể hiện bằng màu khác so với vết nứt, điều này giúp cho người dùng dễ nhìn dẫn đến việc chọn điểm giao nhanh hơn và chuẩn xác hơn (Hình 4.2b). Kết quả cuối cùng của quá trình gán nhãn là những pixel người dùng chọn trực tiếp trên GUI và được xác định là điểm giao thực sự sẽ được lưu thành file định dạng txt để phục vụ cho quá trình đánh giá (Hình 4.2c). Dữ liệu được lưu là tọa độ ứng với những pixel điểm giao đó, với kiểu dữ liệu này, người dùng có thể sử dụng để đánh giá kết quả mọi thuật toán, mọi thông số đánh giá, điển hình như các thông số được trình bày ở mục bên trên.



Hình 4. 2: Quá trình chọn, xác nhận và lưu điểm giao

Với sự linh hoạt này dữ liệu kết quả có thể được sử dụng cho nhiều ứng dụng đánh giá thuật toán nhận diện điểm giao với nhiều kiểu dữ liệu hơn nữa. Đối với một tập dữ liệu muốn gán nhãn, người dùng chỉ việc chọn ảnh từ tập dữ liệu đó một lần duy nhất, vì chỉ cần kích chuột vào “Next” trên GUI thì ảnh tiếp theo sẽ tự động được chọn, cứ như vậy cho đến khi ảnh cuối cùng trong tập dữ liệu được chọn. Điều này cũng giúp GUI giảm đáng kể thời gian chọn ảnh cũng như giúp quá trình gán nhãn trở nên tuần tự và lần lượt hơn, tránh việc gán nhãn lại ảnh đã từng gán nhãn rồi.



Hình 4. 3: Quá trình chuyển đến ảnh tiếp theo

4.2. Kết quả của các thuật toán

4.2.1 Kết quả của thuật toán có độ rộng 1 pixel

Trong mục này, đồ án tốt nghiệp sẽ trình bày kết quả của các thuật toán đối với vết nứt có độ rộng 1 pixel, bao gồm: Bwmorph, Crossing Number, DBSCAN, OPTICS.

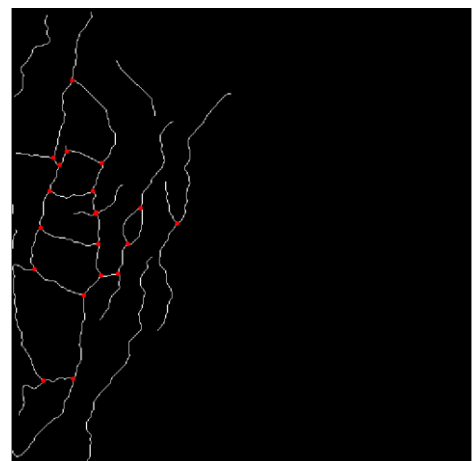
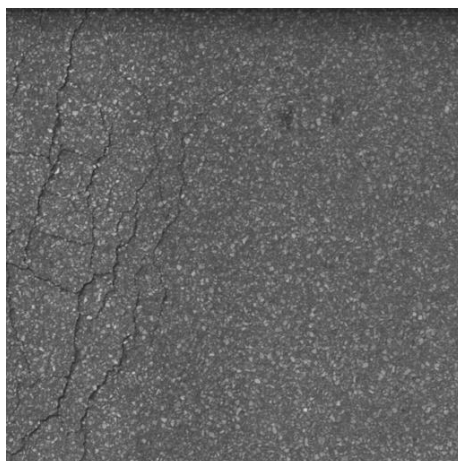
4.2.1.1. Bwmorph

Ảnh

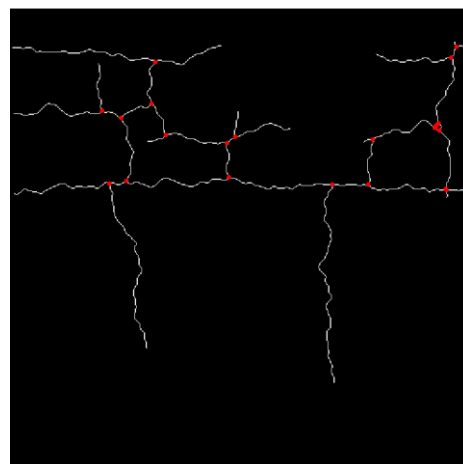
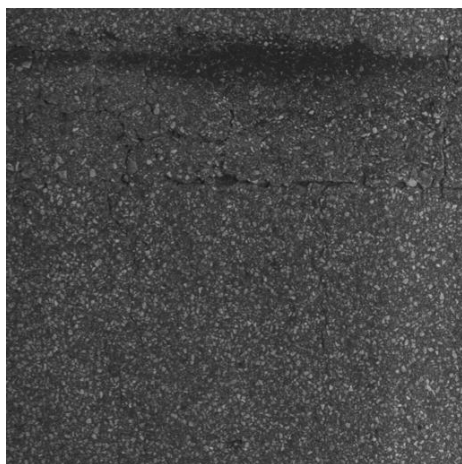
Ảnh gốc

Ảnh kết quả

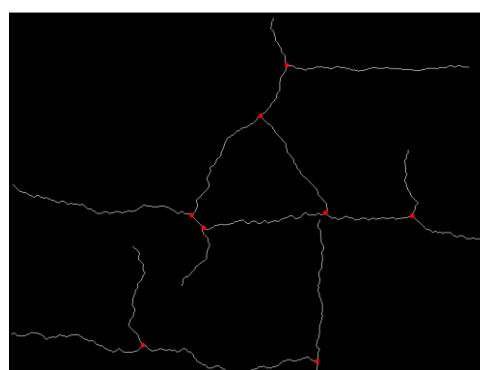
1



2



3



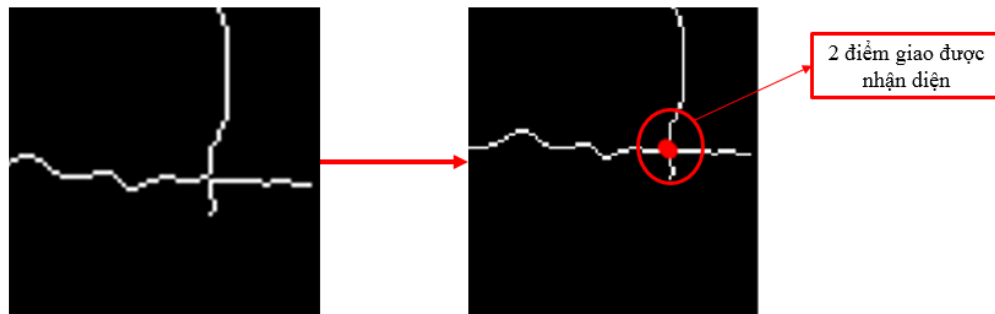
Hình 4. 4: Kết quả của hàm Bwmorph

Trong hình trên, vết nứt được thể hiện là những pixel màu trắng, những pixel đỏ là điểm giao được nhận diện bởi hàm bwmorph. Bằng mắt thường có thể thấy hàm được phát triển bởi Matlab này nhận diện chính xác hoàn toàn tất cả các điểm giao của vết nứt từ vết nứt phức tạp như Ảnh 1 đến vết nứt đơn giản hơn như Ảnh 3. Do hàm này nhận diện chính xác tọa độ của vết nứt nên chúng ta sử dụng các thông số Precision, Recall, F1-Score để đánh giá hiệu quả của nó.

Bảng 4. 1: Kết quả đánh giá định lượng hàm Bwmorph

Tên	Precision	Recall	F1-Score
Ảnh 1	0,9744	1,0000	0,9500
Ảnh 2	0,9545	1,0000	0,9130
Ảnh 3	1,0000	1,0000	1,0000

Kết quả từ Bảng 4.1 cho thấy hiệu suất thực sự của hàm bwmorph. Một kết quả thực sự cao, gần như là tuyệt đối trong tập dữ liệu được sử dụng. Kết quả của cả tập dữ liệu cũng rất vượt trội. Một số trường hợp Bwmorph nhận diện sai điểm giao (hình 4.5 – Vị trí một điểm giao nhưng đã nhận diện thành 2 điểm giao do tồn tại một pixel liền kề có đủ điều kiện trở thành một điểm giao).



Hình 4. 5: Nhận diện thừa điểm giao

4.2.1.2. Crossing Number

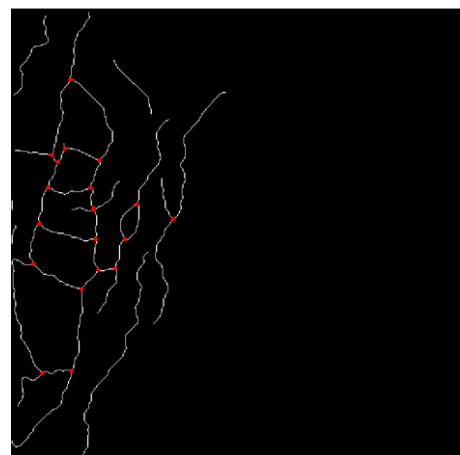
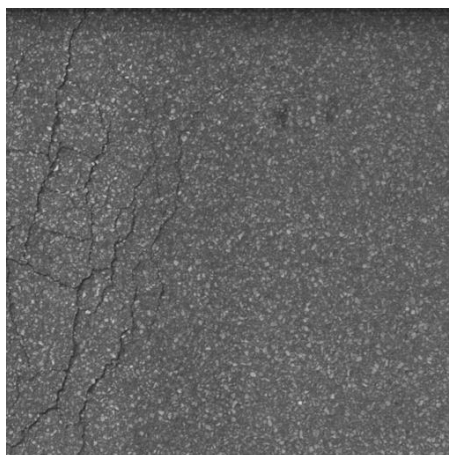
Một thuật toán có hiệu suất tương tự như hàm có sẵn của Matlab đó là thuật toán Crossing number. Tuy không phải hàm được phát triển bởi Matlab nhưng nó cũng mang lại một hiệu suất vượt trội như được thể hiện trong hình dưới đây.

Ảnh

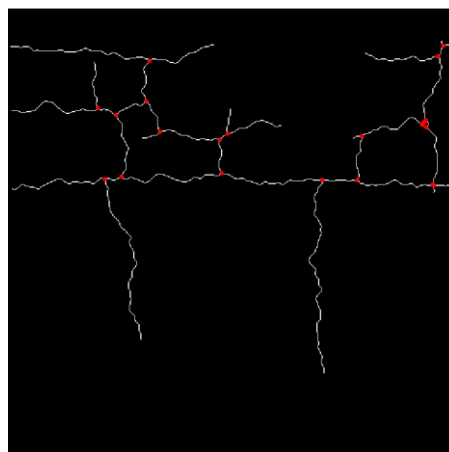
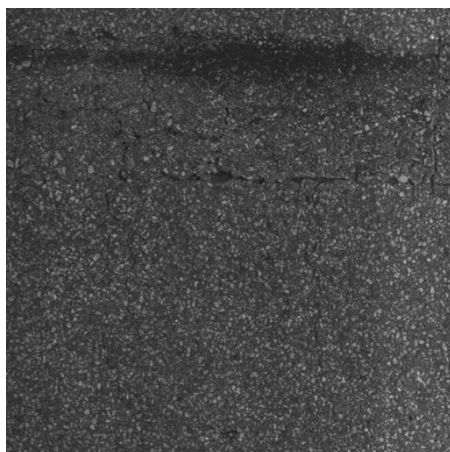
Ảnh gốc

Ảnh kết quả

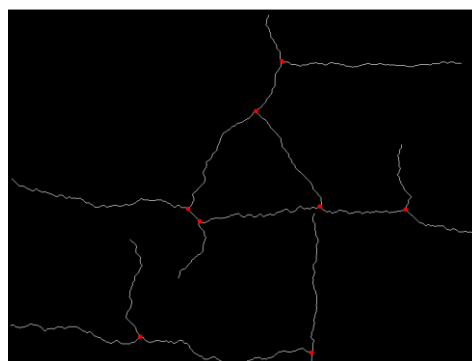
1



2



3



Hình 4. 6: Kết quả của thuật toán Crossing Number

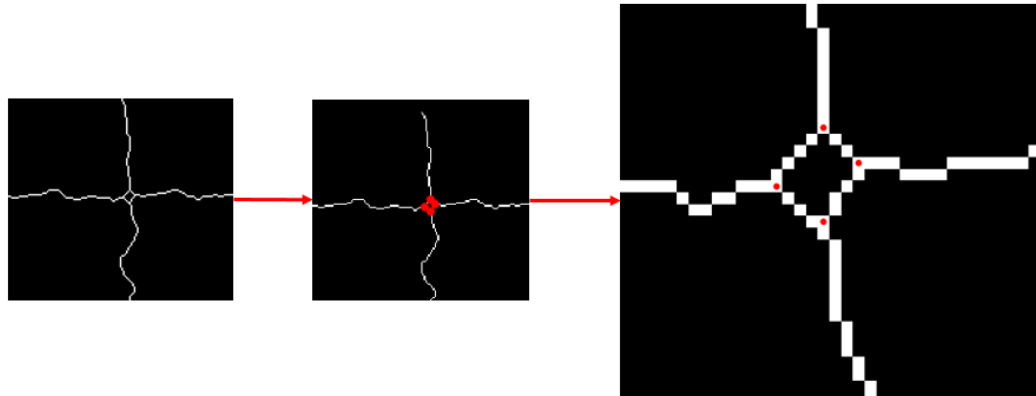
Cũng giống như hàm bwmorph thuật toán Crossing number cũng nhận diện chính xác vị trí điểm điểm nên nó cũng được đánh giá bởi các thông số Precision, Recall, F1-Score.

Bảng 4. 2: Kết của định lượng của thuật toán Crossing Number

Tên	Precision	Recall	F1-Score
Ảnh 1	0,9744	1,0000	0,9500
Ảnh 2	0,9545	1,0000	0,9130
Ảnh 3	1,0000	1,0000	1,0000

Có thể thấy hàm Bwmorph và thuật toán Crossing Number cho kết quả tương đương nhau, tuy nhiên do là hàm có sẵn bởi Matlab nên tốc độ của Bwmorph là vượt trội hơn. Crossing Number cũng tồn tại một số trường hợp khiến hiệu suất của việc nhận

diện điểm giao bị suy giảm như Bwmorph. Tuy nhiên trong những không gian hẹp tồn tại nhiều điểm giao cả 2 thuật toán trên đều phát hiện chính xác và đầy đủ như hình dưới đây.



Hình 4. 7: Nhận diện đầy đủ điểm giao khi chúng ở khoảng cách gần nhau

4.2.1.3. DBSCAN

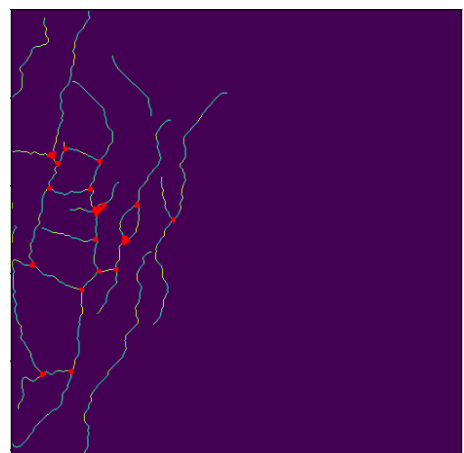
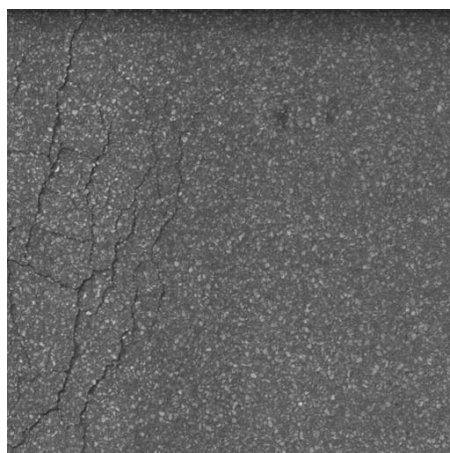
Tiếp theo là một thuật toán tìm điểm giao nữa đó là DBSCAN, do đối số đầu vào – số pixel tối thiểu được nhóm lại thành một cụm (*minPts*) là 4 nên mỗi điểm giao mà nó nhận diện được thường bao gồm tối thiểu là 4 pixel và có thể lên tới 8 hoặc 10 pixel. Chính vì vậy để đánh giá hiệu suất của hàm này một quá trình xử lý được thực hiện. Chúng sẽ được đối chiếu với dữ liệu đã được gán nhãn, nếu một pixel trong tối thiểu 4 pixel đó trùng khớp với giá trị đã được gán nhãn thì sẽ chọn pixel đó làm pixel trung tâm, ngược lại nếu không trùng khớp với giá trị vào được gán nhãn thì cả 4 pixel đó không phải là điểm giao. Dưới đây là kết quả của thuật toán này.

Ảnh

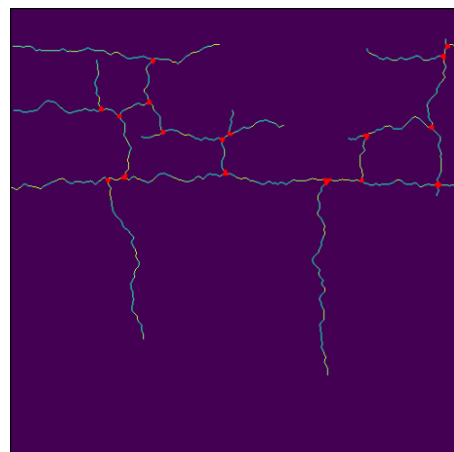
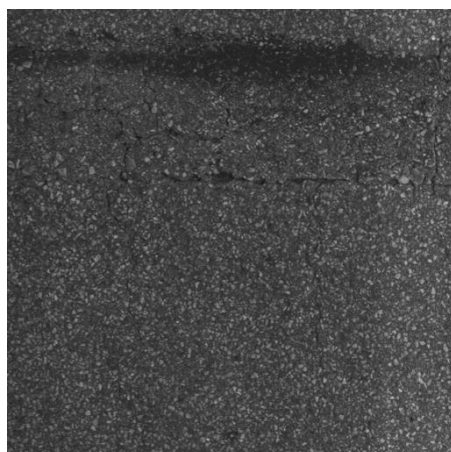
Ảnh gốc

Ảnh kết quả

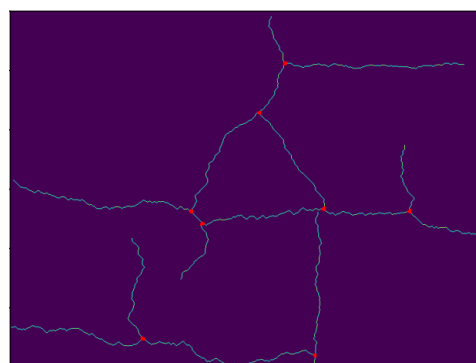
1



2



3



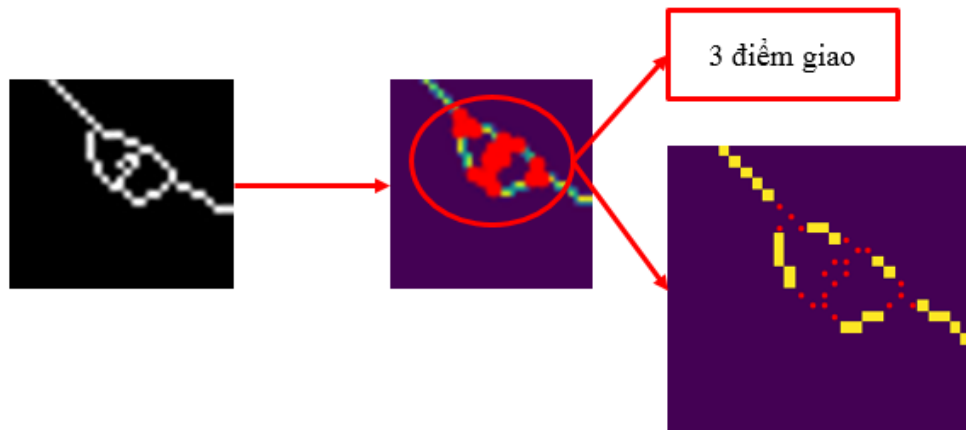
Hình 4. 8: Kết quả của thuật toán DBSCAN

Bằng mắt thường cũng có thể thấy được độ chính xác của thuật toán DBSCAN. Nó có thể nhận diện hoàn toàn chính xác hầu hết tất cả các điểm giao có trên vết nứt.

Bảng 4. 3: Kết quả định lượng của thuật toán DBSCAN

Tên	Precision	Recall	F1-Score
Ảnh 1	0,9596	0,9722	0,9474
Ảnh 2	1,0000	1,0000	1,0000
Ảnh 3	1,0000	1,0000	1,0000

Từ kết quả của Bảng 4.3 cho thấy những nhận xét bên trên là hoàn toàn chính xác, Thuật toán DBSCAN mang lại một hiệu suất vượt trội, có nhiều ảnh đạt giá trị chính xác tuyệt đối so với dữ liệu đã được gán nhãn. Tuy nhiên ở một số vị trí do có các điểm giao gần nhau thuật toán không nhận diện đầy đủ chúng do tính năng phân cụm như hình dưới đây. Vị trí được khoanh tròn màu đỏ thực chất có 4 điểm giao nhưng thuật toán chỉ nhận diện được 3 như hình bên dưới.



Hình 4. 9: Nhận diện thiếu điểm giao

4.2.1.4. OPTICS

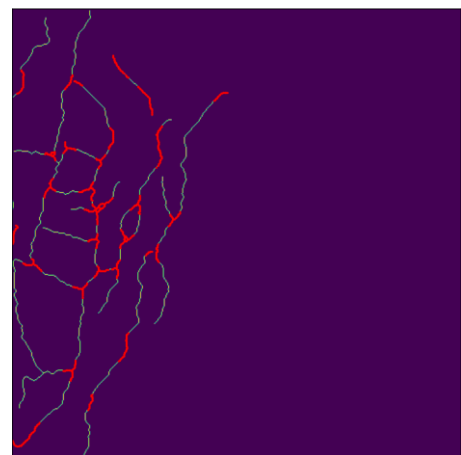
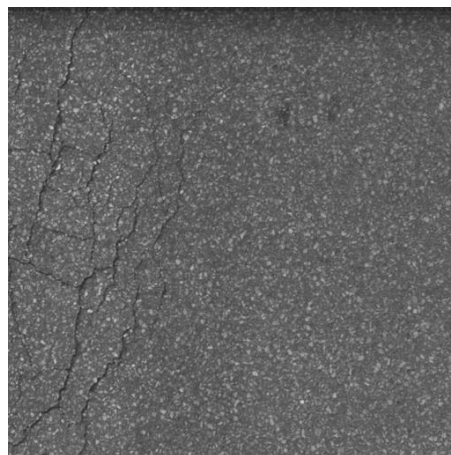
Một thuật khác được phát triển dựa trên thuật toán DBSCAN đó là OPTICS. Thuật toán này chứa các đối số đầu vào giống DBSCAN nhưng giá trị của chúng lại khác. Đối số đầu vào – số pixel tối thiểu được nhóm lại thành một cụm (*minPts*) là 16 và có thể nhiều hơn nữa. Do trong cụm đó sẽ chứa một pixel là điểm giao đúng vị trí nên cả cụm đó sẽ được coi là 1 điểm giao. Chính vì vậy một số cụm được coi là điểm giao chứa một số lượng lớn các pixel, điều này có thể dẫn đến hai hoặc nhiều điểm giao sẽ được coi là một cụm điểm giao hay những cụm có số lượng lớn pixel nhưng lại không chứa pixel điểm giao. Rất khó để tìm được một đối số đầu vào tốt có thể sử dụng cho từng ảnh nên chỉ có thể tìm đối số tốt nhất cho cả tập dữ liệu được sử dụng. Kết quả của thuật toán được thể hiện trong hình dưới đây.

Ảnh

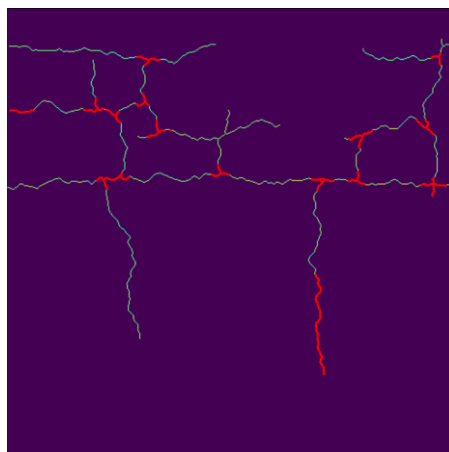
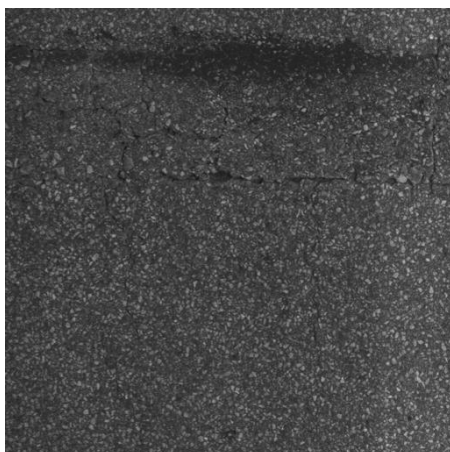
Ảnh gốc

Ảnh kết quả

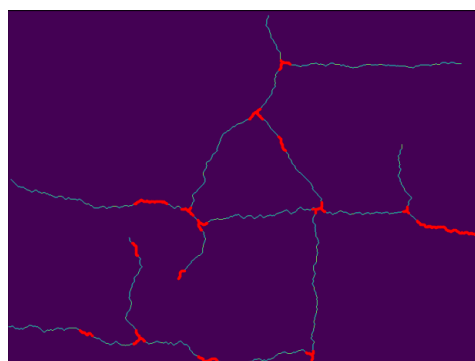
1



2



3



Hình 4. 10: Kết quả của thuật toán OPTICS

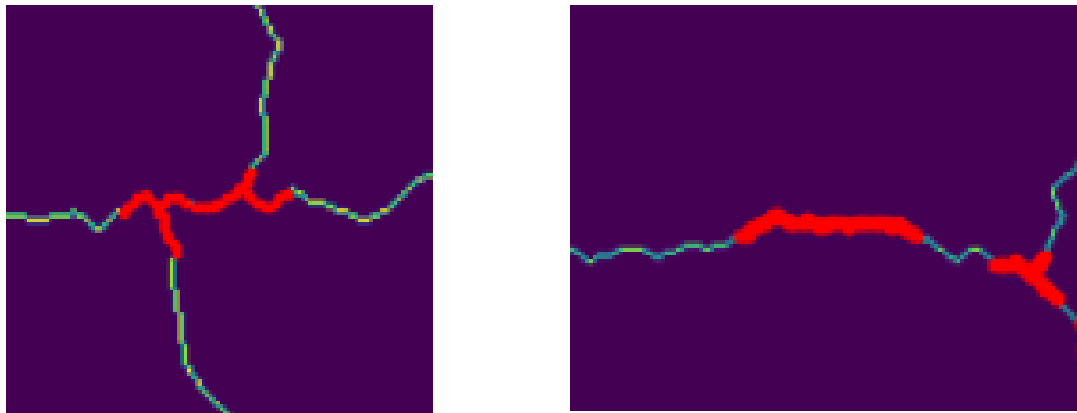
Hình trên cho thấy thuật toán không nhận diện đầy đủ những điểm giao có trong ảnh vết nứt cũng như nhận diện sai một số vị trí là điểm giao – là những đường vết nứt đơn dài. Do chúng ta vẫn coi cả một cụm điểm giao chứa nhiều pixel chỉ là một điểm giao nên chúng ta cũng có thể dùng các thông số dành cho ảnh vết nứt có độ rộng 1 pixel để đánh giá hiệu suất của nó.

Bảng 4. 4: Kết quả định lượng của thuật toán OPTICS

Tên	Precision	Recall	F1-Score
Ảnh 1	0,925	0,7931	0,854
Ảnh 2	0,8421	0,9062	0,873
Ảnh 3	0,9375	0,7178	0,8137

Bảng 4.4 cho thấy tuy hiệu suất của thuật toán không cao như những thuật toán đã nói trước đó nhưng nó cũng đem lại kết quả tương đối tốt. Thuật toán còn tồn tại rất nhiều trường hợp sai như nhận thiếu điểm giao hay nhận diện sai điểm giao như hình

minh họa dưới đây. Hình 4.11 bên trái cho thấy việc coi 2 điểm giao gần nhau được nhận diện là một điểm giao, điều này cũng có thể xảy ra đối với 3, 4 điểm giao gần nhau. Hình bên phải thể hiện việc điểm giao được nhận diện sai trên một vết nứt đơn.



Hình 4. 11: Nhận diện sai điểm giao

4.2.1.5. Tương quan kết quả giữa các thuật toán

Qua những kết quả của các thuật toán cho vết nứt có độ rộng 1 pixel, chúng ta có thể thấy được hiệu suất vượt trội của chúng trên những hình ảnh và kết quả định tính được nói ở trên. Nhằm mục đích so sánh các thuật toán với nhau và hiệu suất của chúng trên toàn bộ tập dữ liệu

Bảng 4. 5: Kết quả định lượng của cả tập dữ liệu có độ rộng 1 pixel trên các thuật toán

Tên thuật toán	Precision	Recall	F1-Score
Bwmorph	0,998594	0,983131	0,990474
Crossing number	0,998594	0,983131	0,990474
DBSCAN	0,986449	0,99231	0,995382
OPTICS	0,900474	0,776103	0,833023

Bảng 4.5 cho thấy kết quả trung bình của 4 thuật toán tìm điểm giao dành cho vết nứt có độ rộng 1 pixel trên tập dữ liệu 86 ảnh. Kết quả vẫn cho thấy sự vượt trội về hiệu suất, precision của 2 thuật toán bwmorph và Crossing Number lên tới 99.8%, Recall của DBSCAN cũng lên đến 99.2%. Cũng là DBSCAN có F1-Score lên tới 99.5%, cao nhất

trong các thuật toán trên, sau đó là Bwmorph và Crossing Number với F1 là 99%. Thấp nhất trong số các thuật toán trên là OPTICS với 83%.

Các thuật toán trên không chính xác hoàn toàn trên tất cả các ảnh của tập dữ liệu là do ở một số ảnh có cấu trúc phức tạp. Hai thuật toán nhận diện đầu đủ điểm giao chính xác là Bwmorph và Crossing Number nhưng chúng lại nhận diện quá số lượng điểm giao, có nghĩa là ngoài những điểm giao chúng nhận diện đầy đủ còn phát sinh một số điểm giao khác không được coi là điểm giao. Những pixel này nằm lân cận những pixel điểm giao và chúng cũng có những đặc tính như một điểm giao. Tuy nhiên khi xóa một trong những pixel đó vẫn có thể tách vết nứt phức tạp thành các vết nứt đơn vì vậy việc phát sinh một điểm giao lân cận một điểm giao khác là không cần thiết. Chính vì vậy khi gán nhãn dữ liệu, chỉ một trong những pixel đó được gán nhãn. Đối với những trường hợp như vậy chúng ta có gán nhãn lại tập dữ liệu hoặc chỉnh sửa lại thuật toán nhận diện điểm giao đó là loại các pixel điểm giao lân cận không cần thiết hoặc coi tất cả những pixel đó là một điểm giao duy nhất. Cũng trong tập dữ liệu đó tồn tại một số vết nứt có cấu trúc phức tạp, nhiều điểm giao có vị trí gần nhau nhưng không lân cận như trường hợp nói trên, chúng cách nhau từ 3-5 pixel vì vậy những thuật toán phân cụm như DBSCAN hay OPTICS sẽ gặp hạn chế vào những trường hợp như thế này. Những điểm giao gần nhau như vậy sẽ chỉ được coi là một cụm hay là một điểm giao. Chính vì điều đó số lượng điểm giao chính xác sẽ không được đầy đủ so với dữ liệu gán nhãn ảnh hưởng đến kết quả đánh giá hiệu suất của thuật toán. Trong một cụm đó cũng đã chứa tất cả những pixel mà được coi là điểm giao nhưng khi nhận diện và đánh giá chúng chỉ được coi là một điểm giao duy nhất. Điều đó có thể cải thiện bằng cách phân tích lại cụm chứa nhiều hơn một điểm giao đó để tách nó thành các điểm giao khác. Ngoài ra riêng ở thuật toán OPTICS nhiều vị trí không phải điểm giao nhưng được nhận diện là điểm giao. Một phần là do đối số đầu vào của thuật toán khá lớn, phần còn lại là do cấu trúc vết nứt phức tạp. Chúng ta có thể tìm cho mỗi ảnh một đối số tốt nhất với nó chứ không chỉ sử dụng một đối số duy nhất như hiện tại để cải thiện hiệu quả thuật toán nhưng điều đó cũng sẽ gặp trở ngại khi tập dữ liệu quá lớn.

4.2.2. Kết quả của thuật toán có độ rộng khác 1 pixel

Trong phần này 2 thuật toán tiếp theo sẽ được đánh giá hiệu suất của chúng. Tuy nhiên thông số đánh giá 2 thuật toán này sẽ khác với các thuật toán đã nói ở trên vì chúng là những vết nứt có độ rộng khác một pixel. Sở dĩ có điều này là vì 2 thuật toán Square – based và Run Representation không nhận diện chính xác vị trí của điểm giao mà nó chỉ nhận diện được đường viền xung quanh những pixel điểm giao đó. Chính

những điều đó các thông số như: $I_{Revealed}$, $I_{PositionCorrect}$ và $I_{Correct}$ được sử dụng để xác định hiệu suất của hai thuật toán đó. Chúng ta cũng không thể đưa vết nứt về dạng 1 pixel rồi sử dụng 2 thuật toán trên và sau đó đánh giá nó theo các thông số cho vết nứt có độ rộng 1 pixel. Do những thuật toán đó được phát triển dành riêng cho vết nứt có độ rộng khác 1 pixel, nếu áp dụng cho vết nứt 1 pixel kết quả nhận được có thể sai rất nhiều.

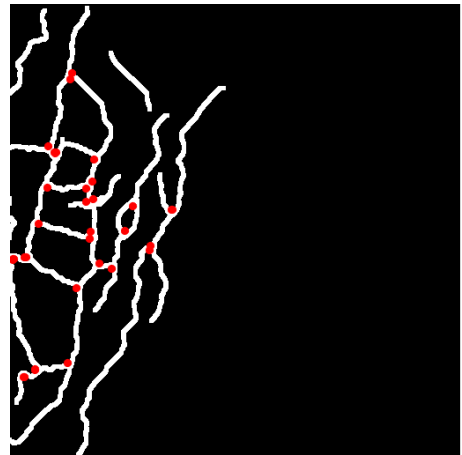
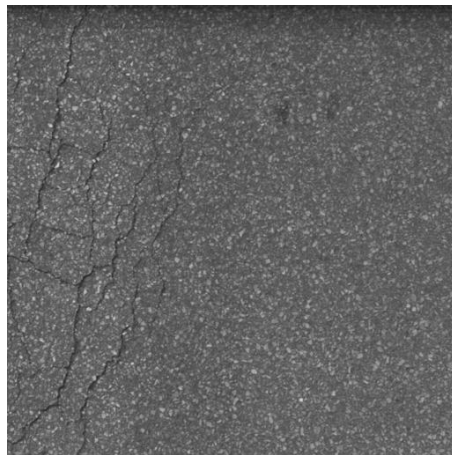
4.2.2.1. Square based

Ảnh

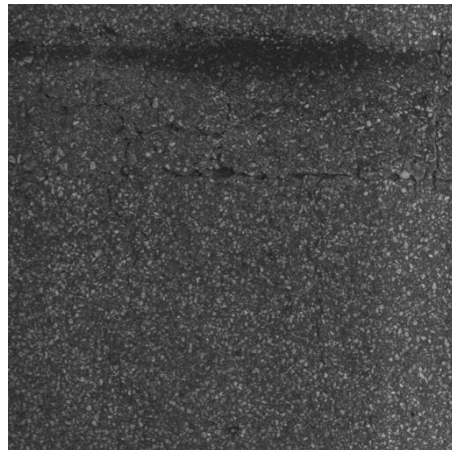
Ảnh gốc

Ảnh kết quả

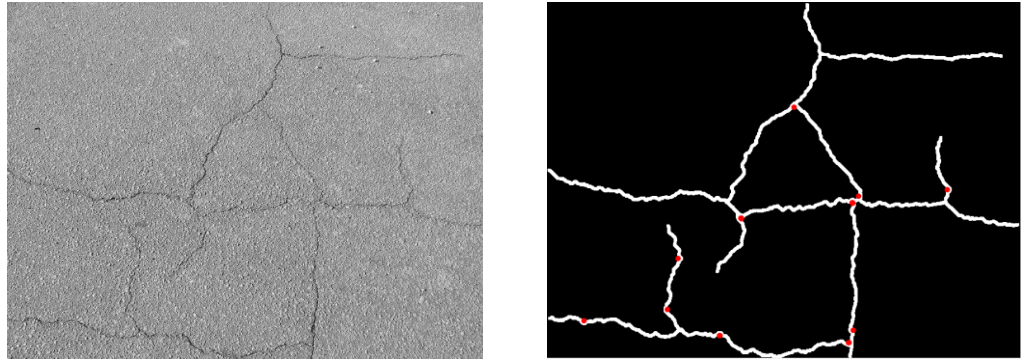
1



2

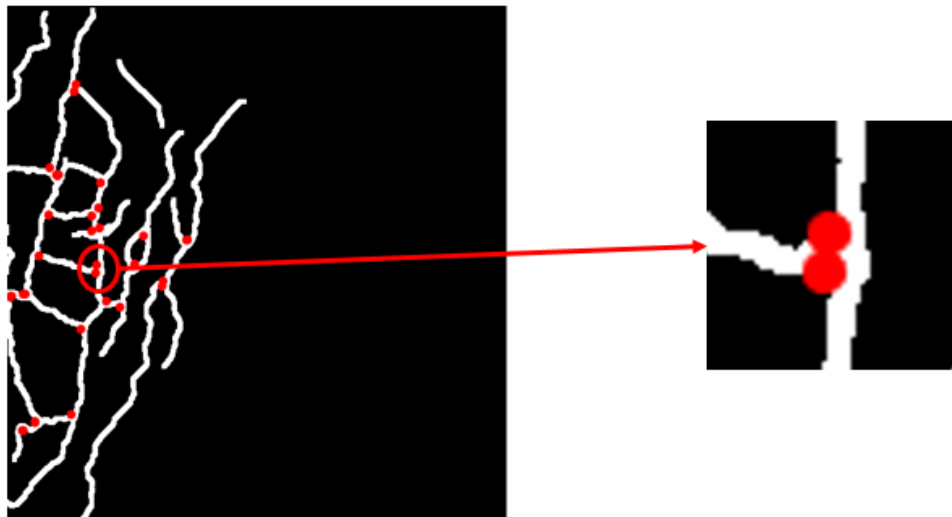


3



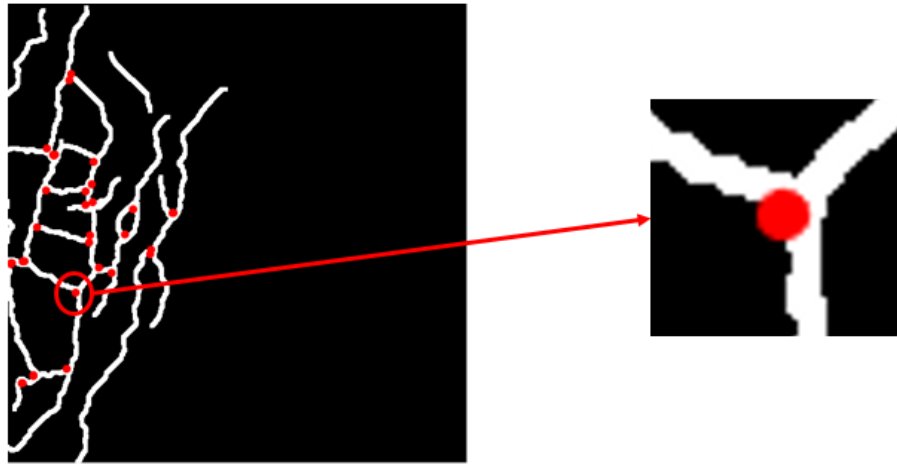
Hình 4. 12: Kết quả của thuật toán Square based

Từ hình trên ta thấy đúng như những nhận xét bên trên, điểm giao nó đều nằm trên đường viền của vết nứt có độ rộng khác 1 pixel. Chính vì điểm giao tìm được nằm trên đường viền nên tại một khu vực là giao điểm của các vết nứt đơn sẽ có thể có hơn một điểm giao được tìm thấy nằm trên đường viền. Tuy nhiên chúng vẫn chỉ được coi là một điểm giao và chúng được thể hiện như hình dưới đây:



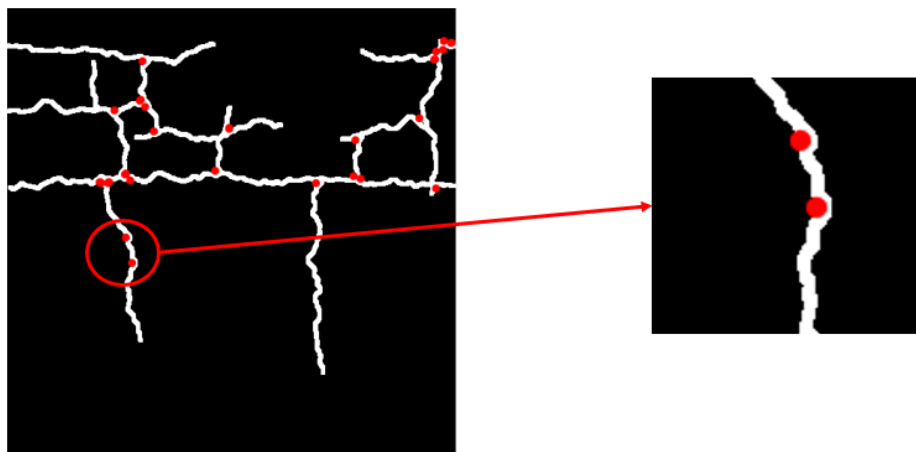
Hình 4. 13: Nhận diện được 2 điểm giao nhưng chỉ được tính là một điểm giao

Trên đây là trường hợp 2 pixel khác nhau được coi là điểm giao nằm trên viền của vết nứt nhưng chúng sẽ được coi là một điểm giao vì chúng cùng biểu diễn cho việc 2 vết nứt đơn giao nhau, chỉ là chúng nằm ở 2 vị trí đường viền khác nhau mà thôi. Ngoài trường hợp trên còn tồn tại các trường hợp khác như 3, 4 pixel khác nhau và chúng vẫn chỉ được coi là một điểm giao. Cũng có trường hợp 2, 3 vết nứt đơn giao nhau nhưng thuật toán chỉ nhận diện được một pixel đường viền duy nhất được coi là điểm giao.



Hình 4. 14: Chỉ nhận diện được 1 điểm giao

Đó là những trường hợp thuật toán nhận diện chính xác, bên cạnh đó còn khá nhiều vị trí mà thuật toán đã nhận diện sai do ở những vết nứt đơn có nhiều điểm cong nhỏ và thuật toán lầm tưởng rằng đó là vị trí giao của nhiều hơn một vết nứt đơn.



Hình 4. 15: Nhận diện sai điểm giao

Để có một cái nhìn tổng quan hơn chúng ta sẽ xem hiệu suất của thuật toán này thông qua các thông số đánh giá dành cho nó.

Bảng 4. 6: Kết quả định lượng của thuật toán Square based

Tên	$I_{Revealed}$	$I_{PositionCorrect}$	$I_{Correct}$
Ảnh 1	1	19/21	17/21
Ảnh 2	15/19	15/18	12/18
Ảnh 3	5/8	5/11	-1/11

Bảng 4.6 cho thấy hiệu suất thực sự của thuật toán. Thuật toán có thể nhận diện đầy đủ điểm giao như những gì dữ liệu gán nhãn có như Ảnh 1, tỷ lệ điểm giao được nhận diện chính xác và điểm giao được nhận diện cũng tương đối cao. Điều đó cho thấy những điểm giao nhận diện sai cũng chiếm một số lượng nhỏ. Tuy nhiên vẫn tồn tại một số ảnh (điển hình như Ảnh 3) có số lượng điểm giao sai nhiều hơn số lượng điểm giao đúng và tỷ lệ giữa điểm giao đúng và dữ liệu gán nhãn cũng không cao.

4.2.2.2. Run Representation

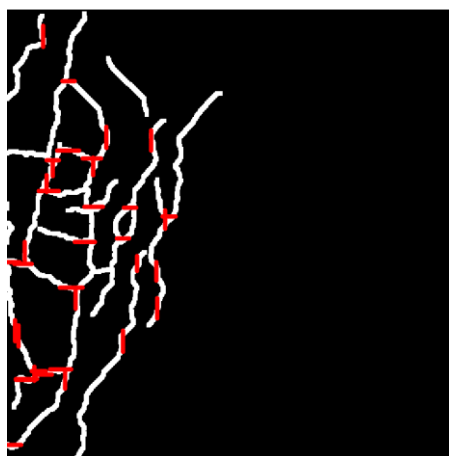
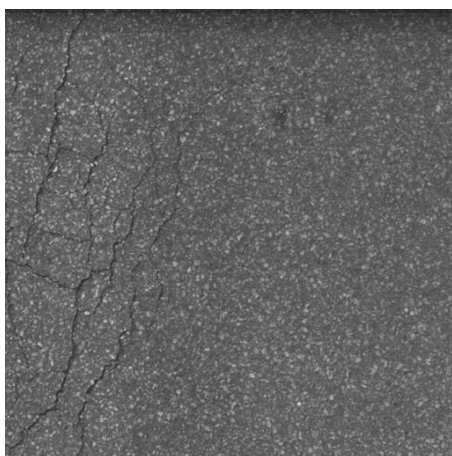
Một thuật toán nữa không dành cho vết nứt có độ rộng 1 pixel có là Run Representation. Nhưng điểm giao nó tìm được không còn nằm trên đường viền nữa mà nó tồn tại dưới dạng đường thẳng theo hướng thẳng đứng hoặc theo hướng nằm ngang như hình dưới đây.

Ảnh

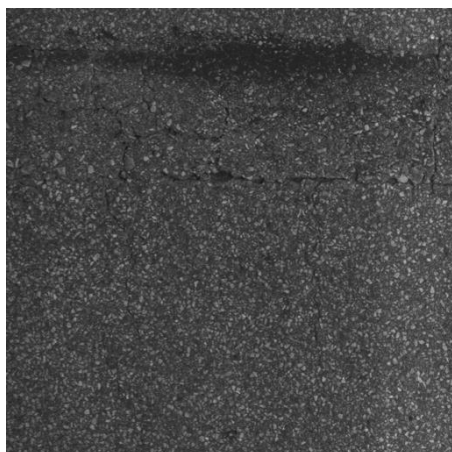
Ảnh gốc

Ảnh kết quả

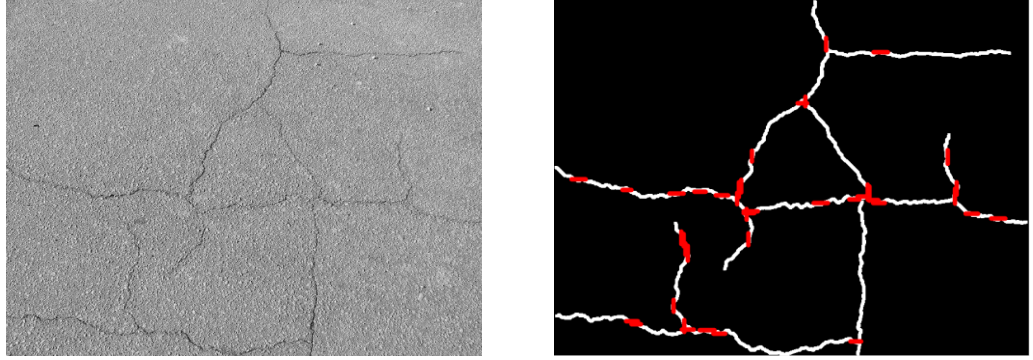
1



2



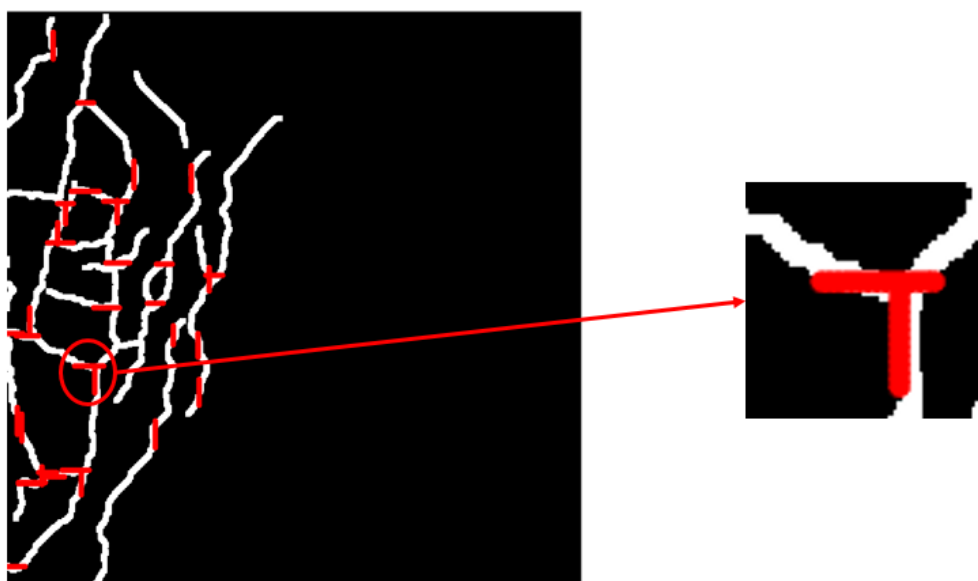
3



Hình 4. 16: Kết quả của thuật toán Run Representation

Hình 4.16 cho thấy kết quả của thuật toán Run Representation. Chỉ bằng mắt thường cũng có thể thấy được khá nhiều sai sót trong thuật toán này. Số lượng điểm giao nhận diện sai là khá lớn đối với cả 3 ảnh, điểm giao cũng không được nhận diện đầy đủ như trong ảnh 2, 3.

Khác với thuật toán Crossing Number hay DBSCAN ở thuật toán này các điểm giao được tìm bằng cách xác định vị trí rẽ nhánh hoặc tập tụ của các vết nứt đơn nhưng cũng giống ở chỗ không nhận diện chính xác được vị trí của pixel điểm giao. Ở thuật toán này những đường thẳng được tạo bởi các pixel liên kề nằm ngang hoặc nằm dọc sẽ được tìm thấy khi chúng là những đường khởi đầu cho việc rẽ nhánh hoặc kết thúc cho việc hội tụ của các vết nứt. Tại một số điểm giao nhất định sẽ có cả đường thẳng đứng và đường thẳng nằm ngang vì chúng đều thỏa mãn việc rẽ nhánh hoặc hội tụ nhưng chúng cũng chỉ được coi là kết quả của một điểm giao duy nhất. Còn một điểm khác nữa so với thuật toán Square - based ở thuật toán này tại một điểm giao chỉ có tối đa 2 đường thẳng xung quanh nó vì đường thẳng chỉ tồn tại dưới dạng ngang và dọc.



Hình 4. 17: Quá trình xác định điểm giao của thuật toán Run Representation

Do các vết nứt có cấu trúc hết sức phức tạp nên thuật toán này đã gặp khó trong quá trình nhận diện điểm giao. Chính vì điểm suất hiện nhiều vị trí nhận diện sai. Độ dài của các đường thẳng nằm ngang hay dọc cũng không thể là tốt với một giới hạn duy nhất mà sử dụng cho cả tập dữ liệu vết nứt. Bởi những lý do đó kết quả của thuật toán này không được tốt, nhận diện sai rất nhiều vị trí. Điều đó được thể hiện trong hình 4.18.



Hình 4. 18: Không nhận diện được và nhận diện sai điểm giao

Một số điểm giao không được nhận diện và cũng nhận diện sai một số lượng khá lớn các đường thẳng mà thuật toán này coi nó là vị trí rẽ nhanh hay hội tụ. Nhưng để biết được hiệu suất thực sự của thuật toán này chúng ta vẫn cần dựa vào các thông số để đánh giá nó.

Bảng 4. 7: Kết quả định lượng của thuật toán Run Representation

Tên	$I_{Revealed}$	$I_{PositionCorrect}$	$I_{Correct}$
Ảnh 1	14/19	14/27	1/19
Ảnh 2	12/19	12/35	-11/19
Ảnh 3	7/8	7/38	-31/8

Từ Bảng 4.7 có thể thấy được hiệu suất của thuật toán này không cao. Số lượng điểm giao nhận diện sai hầu hết đều nhiều hơn số lượng điểm giao nhận diện đúng (Ảnh 2, Ảnh 3). Tuy nhiên nó cũng không bỏ sót quá nhiều điểm giao đúng vì tỷ lệ điểm giao nhận diện chính xác và dữ liệu gán nhãn cũng khá cao (Ảnh 3).

4.2.2.3. *Trương qua kết quả giữa các thuật toán*

Dưới đây sẽ là kết quả của tập dữ liệu 86 ảnh được sử dụng trong đồ án này

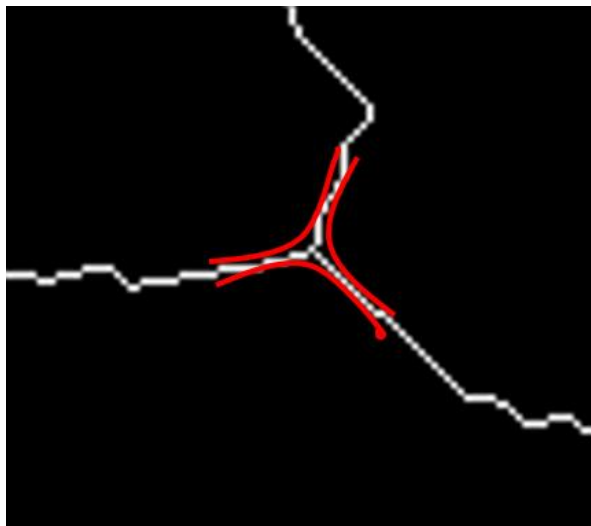
Bảng 4. 8: Kết quả trung bình của các thuật toán trên tập dữ liệu vết nứt có độ rộng khác 1 pixel

Tên thuật toán	$I_{Revealed}$	$I_{PositionCorrect}$	$I_{Correct}$
Run Representation	0,875764	0,315365	-0,36927
Square-based	0,809572	0,742297	0,484594

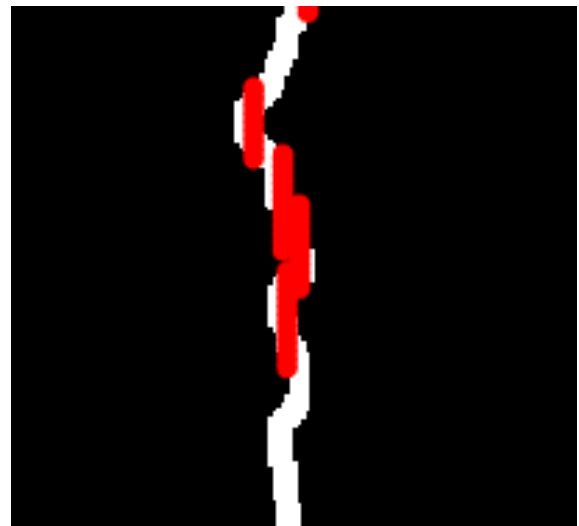
Tuy rằng hai thuật toán được sử dụng cho vết nứt có độ rộng khác 1 pixel được đánh giá theo các thông số khác so với các thuật toán dùng cho vết nứt có độ rộng 1 pixel nhưng bằng mắt thường và Bảng 4.8 bên trên có thể thấy hiệu suất của chúng cho sự chênh lệch nhất định. Trong hai thuật toán này thì Square-based có hiệu suất tốt hơn Run Representation tuy ở thông số tỷ lệ giữa điểm giao nhận diện chính xác bằng phương pháp tự động và điểm giao được xác định bằng dữ liệu gán nhãn của Run Representation lại cao hơn Square - based (87%). Thông số $I_{Correct}$ tốt hơn của Square-based cho thấy số điểm giao nhận diện sai của thuật toán Square-based là ít hơn.

Hai thuật toán phụ thuộc rất nhiều vào cấu trúc vết nứt và cấu trúc điểm giao. Đối với thuật toán Square-based nó dựa vào số lượng pixel vết nứt xung quanh một pixel được coi là vết nứt vì số lượng đó có thể nhiều lần xác định một pixel có phải là điểm giao hay không. Tại vị trí một số điểm giao như Hình 4.19a, khu vực đường viền của

điểm giao chứa những đường gấp khúc (do các vết nứt đơn giao nhau) với góc tương đối lớn. Chính vì vậy tại một pixel đường viền, số lượng những pixel lân cận trong phạm vi ma trận 7×7 sẽ không đủ điều kiện để khiến nó trở thành một điểm giao. Điều này cũng là một vấn đề đối với thuật toán Run Representation bởi điểm giao được tìm bằng những đường thẳng với số lượng pixel nhất định theo chiều dọc hoặc ngang. Cho nên những vết nứt đơn nằm gần như song song với chiều ngang hoặc chiều dọc thì rất khó tìm được vị trí mà chúng giao nhau vì khi đó điều kiện về số lượng pixel trong một đường thẳng sẽ phải tăng lên nhưng nó cũng khiến thuật toán nhận diện sai nhiều vị trí không phải điểm giao khác. Điều đó đã được thể hiện trong Hình 4.19b. Để cải thiện điều này có thể tăng thêm nhiều điều kiện để xác nhận một pixel là điểm giao hoặc kết hợp với các thuật toán phân cụm để có thể nhận diện chính xác vị trí điểm điểm giao và loại bỏ nhiều điểm giao được phát hiện sai. Một điều hạn chế nữa của hai thuật toán này đó chính là nó chỉ được áp dụng cho vết nứt có độ rộng lớn hơn 1 pixel.



a



b

Hình 4. 19: Các trường hợp nhận diện sai và không nhận diện được điểm giao

KẾT LUẬN

Nghiên cứu này đã đưa ra tổng quan về lý thuyết của các thuật toán tìm điểm giao từ các vết nứt phức tạp để cải thiện kết quả của các thuật toán tìm kiếm vết nứt nói riêng và xử lý ảnh nói chung.

Giao diện gắn nhãn dữ liệu được thiết kế trên Matlab đã giải quyết được vấn đề tính hiệu suất của các thuật toán tìm điểm giao và là bước đầu để phát triển các giao diện gắn nhãn dữ liệu vết nứt khác. Các tập dữ liệu CrackTree200, CRKWH100, CrackLS315 vừa dùng để gắn nhãn dữ liệu vừa sử dụng để áp dụng cho các thuật toán tìm điểm giao.

Các thuật toán tìm điểm giao đã để lại nhiều ấn tượng về hiệu suất cao như DBSCAN hay Crossing Number với kết quả tốt trên tập ảnh có độ rộng 1 pixel, tuy nhiên cũng còn một số hạn chế và có thể cải thiện được. Các thuật toán trên tập ảnh vết nứt có độ rộng khác 1 pixel chưa thể hiện kết quả tốt như mong đợi.

TÀI LIỆU THAM KHẢO

Website

- [1] [Online].Available:<https://www.mathworks.com/help/images/ref/bwmorph.html>.
- [2] [Online].Available:<https://scikitlearn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html>.
- [3] [Online].Available:<https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/>.
- [4] [Online].Available:https://en.wikipedia.org/wiki/OPTICS_algorithm#:~:text=Ordering%20points%20to%20identify%20the,Peter%20Kriegel%20and%20J%C3%B6rg%20Sander..
- [5] [Online].Available:<https://www.geeksforgeeks.org/ml-optics-clustering-explanation/>.
- [14] [Online].Available:https://scikitimage.org/docs/stable/auto_examples/edges/plot_skeleton.html.

Tiếng Anh

- [6] P. S. P. B. R. Bansal, "Minutiae Extraction from Fingerprint Images - a Review," Computer Science, 2011.
- [7] L. W. Y. a. J. A. K. Hong, "Fingerprint image enhancement: Algorithm and performance evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, 1998.
- [8] L. C. a. S. L. Zenzo, "Run-Based Algorithms for Binary Image Analysis and Processing," *IEEE Trans Pattern Analysis and Machine Intelligence*, vol. 18, pp. 83-88, 1996.
- [9] H. Y. H. S. C. J Hwan Shin, "Detecting fingerprint minutiae by run length encoding scheme," *Pattern Recognition*, vol. 39, pp. 1140-1154, 2005.
- [10] Z. Y. a. X. Qinghan, "An optimized approach for," *International Joint Conference on Neural Networks*, pp. 391-395, 2006.
- [11] N. Otsu, "A threshold selection method from gray level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, pp. 62-66, 1989.

- [12] M. V. Gamassi, "Fingerprint local analysis for high-performance minutiae extraction," *IEEE International Conference on Image Processing*, 2005.
- [13] R. B. Alibeigi.E, "Pipelined minutiae extraction from fingerprint images," *Canadian Conference on Electrical and Computer Engineering*, 2009.
- [15] T. H. & K. A, "Towards More Accurate Contactless Fingerprint Minutiae Extraction and Pose-Invariant Matching," *IEEE Transactions on Information Forensics and Security*, 2020.