TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ MÔN XỬ LÝ ẢNH SỐ**

**TRÍCH XUẤT CHỮ SỐ BẢNG SUDOKU**

*Người hướng dẫn*: **TS PHẠM VĂN HUY**

*Người thực hiện*: **NGUYỄN LÊ NGỌC HUY – 519H0295**

Lớp **: 19H50302**

Khoá  **: 23**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN/ĐỒ ÁN CUỐI KÌ MÔN XỬ LÝ ẢNH SỐ**

**TRÍCH XUẤT CHỮ SỐ BẢNG SUDOKU**

Người hướng dẫn: **TS PHẠM VĂN HUY**

Người thực hiện: **NGUYỄN LÊ NGỌC HUY**

Lớp **: 19H50302**

Khoá  **: 23**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Em xin chân thành cảm ơn thầy Phạm Văn Huy phụ trách phần lý thuyết và thầy Trịnh Hùng Cường phụ trách phần thực hành của bộ môn Xử lý ảnh số đã giúp đỡ em trong quá trình học tập và tiếp thu, cũng như phát triển kỹ năng và hoàn thành quá trình cũng như đồ án cuối kỳ này.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi và được sự hướng dẫn của TS Phạm Văn Huy;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Lê Ngọc Huy*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Đồ án này được thực hiện bao gồm 2 phần, tách và xử lý phối cảnh cho bảng số và xác định vị trí hàng cột của từng số, xuất file vị trí.

Phần code bao gồm các phần nhỏ như:

* Xác định toàn bộ bảng sudoku trên trang giấy
* Thay đổi phối cảnh bức hình để tập trung vào bảng sudoku
* Xác định lưới và ô của bảng sudoku
* Xác định vị trí của từng số
* Ghi lại bản đồ nhị phân
* Xuất file text

MỤC LỤC

[LỜI CẢM ƠN ii](#_Toc387692905)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN ii](#_Toc387692906)

[TÓM TẮT ii](#_Toc387692907)

[MỤC LỤC 2](#_Toc387692908)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 2](#_Toc387692909)

[CHƯƠNG 1 – MỞ ĐẦU 2](#_Toc387692910)

[1.1 Tiểu mục cấp 1 2](#_Toc387692911)

[1.1.1 Tiểu mục cấp 2 2](#_Toc387692912)

[1.1.1.1 Tiểu mục cấp 3 2](#_Toc387692913)

[1.1.1.2 Tiểu mục cấp 3 tiếp theo. 2](#_Toc387692914)

[1.1.2 Tiểu mục cấp 2 tiếp theo 2](#_Toc387692915)

[1.2 Nội dung của chương này 2](#_Toc387692916)

[CHƯƠNG 2 – TỔNG QUAN 2](#_Toc387692917)

[1.1 Trình bày công thức toán học 2](#_Toc387692918)

[1.2 Trình bày một hình vẽ, sơ đồ 2](#_Toc387692919)

[CHƯƠNG 3 – CƠ SỞ LÝ THUYẾT / NGHIÊN CỨU THỰC NGHIỆM 2](#_Toc387692920)

[3.1 Chèn bảng: 2](#_Toc387692921)

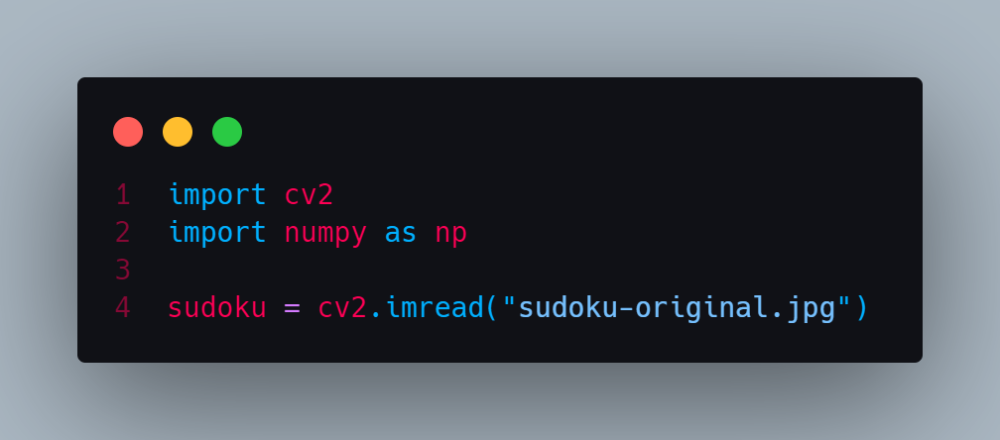
[3.2 Viết tắt 2](#_Toc387692922)

[3.3 Trích dẫn 2](#_Toc387692923)

[3.3.1 Tài liệu tham khảo và cách trích dẫn 2](#_Toc387692924)

[3.3.2 Qui định của Khoa Công nghệ thông tin 2](#_Toc387692925)

CHƯƠNG 1 – CODE VÀ LÝ THUYẾT

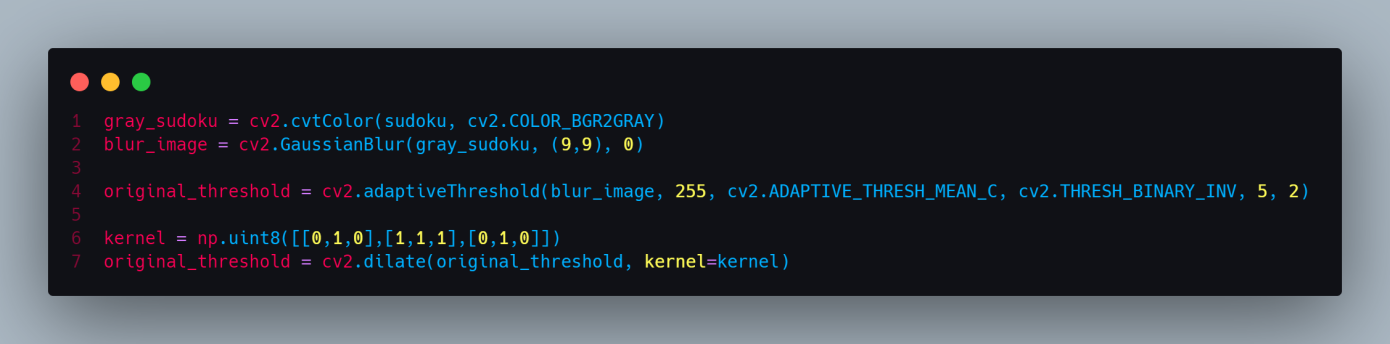
1.1 Import thư viện và hình ảnh vào file python

Hai thư viện được sử dụng nhiều nhất trong đoạn code bao gồm:

* Opencv-python: thư viện xử lý ảnh số thường dùng cho python
* Numpy: thư viện xử lý mảng thường dùng cho tính toán

Khởi đầu đoạn code với việc sử dụng imread của opencv để đưa hình ảnh vào biến sudoku

1.2 Tiền xử lý hình ảnh



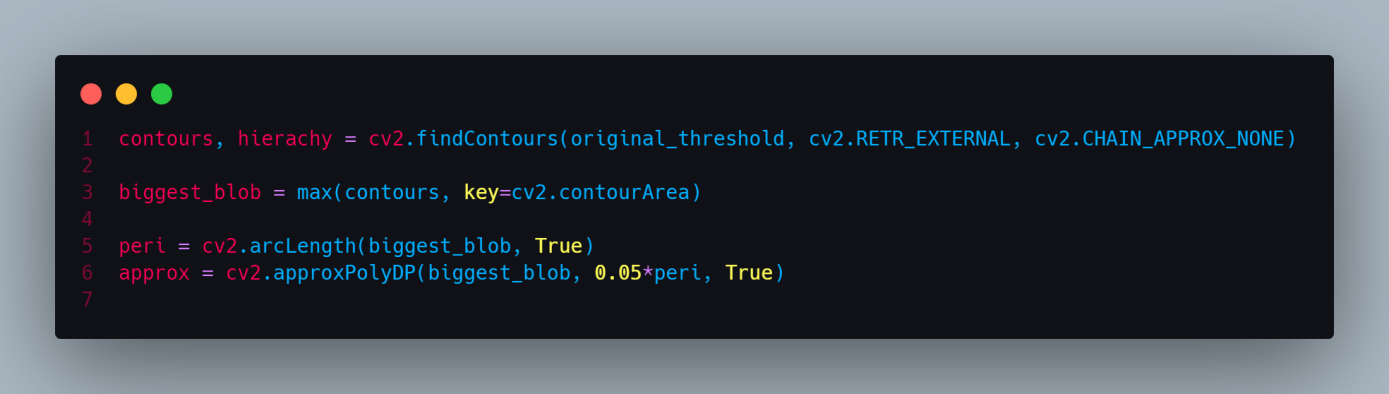
Bước tiếp theo sau khi đưa ảnh vào biến là chuyển nó thành dạng gray scale, dùng hàm cvtColor() của OpenCV

Để giảm bớt độ nhiễu của bức ảnh, bước tiếp theo là sử dụng GaussianBlur() cho bức ảnh, sau nhiều lần thử, ma trận 9x9 là phù hợp để khử nhiễu bức ảnh mà không làm mở đi phần lưới vốn đã không rõ của bức ảnh

Sử dụng adaptiveThreshold sau khi đã khử nhiễu để lấy ra bức ảnh dạng nhị phân, Adaptive được sử dụng là Mean và phương thức chia ngưỡng là Invert Binary để cho ra bức ảnh có lưới và số có màu trắng, vốn là màu tối ở bức ảnh gốc.

Không tránh được việc phân ngưỡng sẽ làm hình ảnh bị khuyết đi vài chỗ nên việc tiếp theo là dùng Dilation cho bức ảnh. Trước tiên cần tạo một kernel, do bảng sudoku bao gồm lưới hàng dọc và hàng ngang nên kernel có dạng chữ thập, khi dilate sẽ làm tăng độ rộng theo chiều ngang và chiều dọc mà không bị phồng nét ở các vị trí chéo.

1.3 Phân tách bảng sudoku

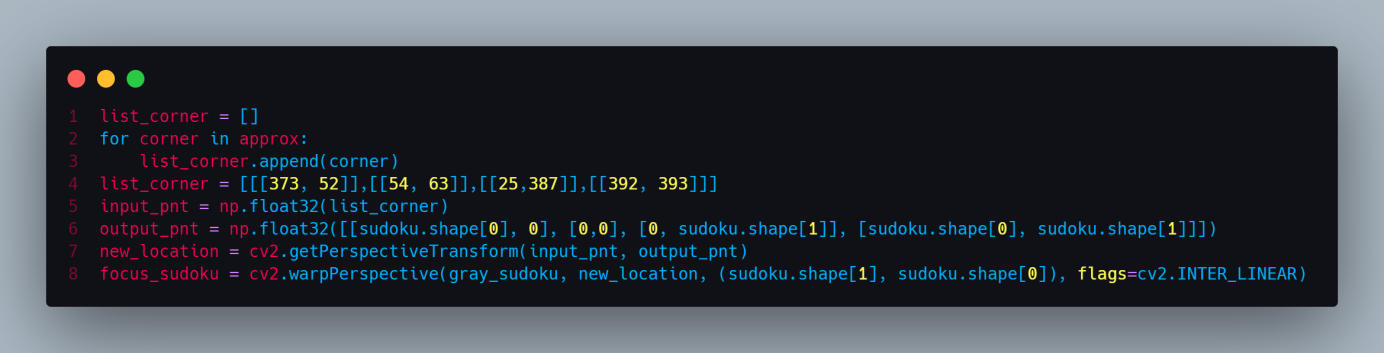


Để xác định được phần viền to nhất của bảng sudoku, ta sử dụng findContours của OpenCV, phần retrieval của hàm ta sử dụng external để contour bao bên ngoài toàn bộ phần khung. Approximate method của hàm ta sử dụng None để contour bao hết toàn bộ bảng.

Sau khi đã vẽ được contour, việc tiếp theo là sử dụng hàm max để xác định contour lớn nhất với key là contourArea để hàm chọn phần contour có diện tích bao phủ to nhất.

Ta tính chu vi đường viền bằng các sử dụng hàm arcLength()

Biến approx để lưu giá trị các góc, sử dụng hàm approxPolyDP để ước lượng hình dạng của đường viền, với epsilon là 0.05 \* chu vi, approx sẽ nhận được giá trị là 4 góc của hình vuông được ước lượng.



Tạo trước một list tên là list\_corner để lưu lại giá trị của biến approx, chỉnh sửa biến một chút để hình ảnh cân xứng cho phần xử lý phối cảnh.

Tiếp theo là lưu biến list\_corner vào một biến khác với định dạng mảng float32 và một biến khác chứa các điểm mà mình mong muốn điều chỉnh. Sử dụng shape của hình ảnh ban đầu để kéo giãn hình ảnh khi xử lý bằng kích thước với ảnh gốc

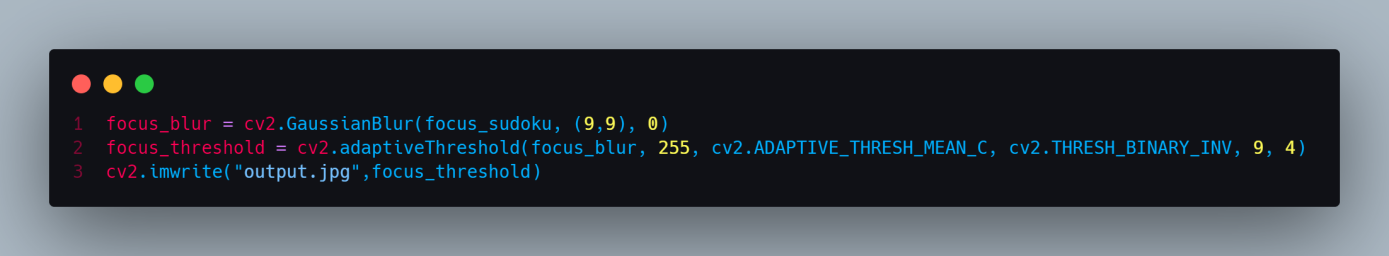
Bước cuối cùng là sử dụng hàm getPerspectiveTransform của OpenCV để tính toán phối cảnh mới sao cho hình ảnh bị kéo ra vẫn giữ nguyên nội dung

Hàm warpPerspective() sẽ sử dụng tính toán của hàm ở trên để xuất ra hình ảnh mới. Hình ảnh input được chọn sẽ là hình ảnh grayscale của bức hình gốc, khi đó qua hình ảnh mới sẽ không cần phải thay đổi lại hệ màu



Image 1. Hình ảnh sau khi được xử lý phối cảnh

1.4 Xử lý ảnh sau khi tách được bảng số và xuất file ảnh nhị phân



Sau khi được hình ảnh mới được phóng to ra của bảng sudoku, việc đầu tiên là sử dụng GaussianBlur() để khử nhiễu

Tiếp theo ta lại sử dụng adaptive threshold để đặt ngưỡng và lấy được ảnh nhị phân của bức hình mới

Sử dụng imwrite của OpenCV để lưu bức hình nhị phân lại, hoàn thành yêu cầu đầu tiên.

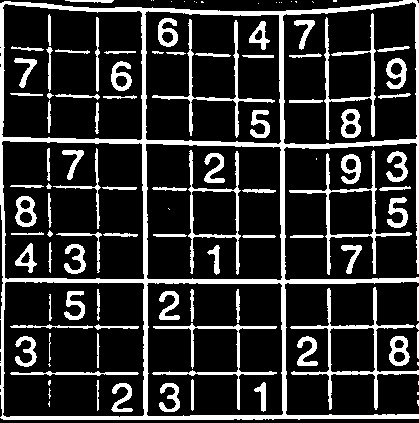
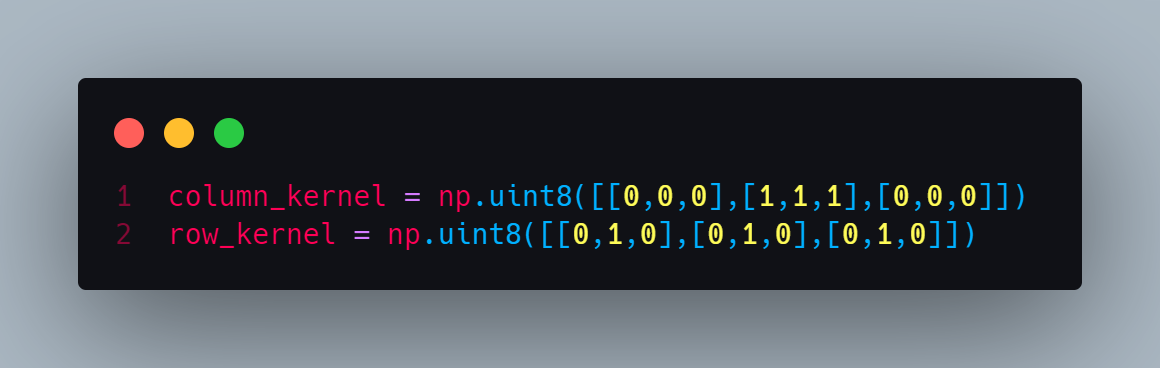


Image 2. Hình nhị phân được xuất thành file output.jpg

1.5 Nhận dạng lưới dọc và ngang



Trước khi bắt đầu nhận dạng, hai kernel được tạo nhầm mục đích dilate và erode theo hai hướng khác nhau, là hai ma trận chỉ có một cột hoặc dòng giữa có giá trị 1.



Để nhận dạng được phần lưới ngang, code đã sử dụng hàm getStructuringElement() có chức năng lấy ra được hình dạng của phần tử cấu trúc, từ đó có thể sử dụng hàm morphologyEx(), hàm xử lý hình thái học của OpenCV, với phần tử cấu trúc là horizontal\_kernel đã được lấy ở trên và loại hình thái, ở đây đã sử dụng hình thái Open, là sự kết hợp của Erosion và Dilation

Tiếp theo là sử dụng dilate và erode để chỉnh lại các phần lưới đã được nhận diện, chúng được đưa vào bên trong vòng lặp để có thể kiểm soát được số lần sử dụng. Do mục tiêu là nối lại nếu lưới nhận dạng bị đứt rời cũng như không quan tâm tới việc nó sẽ kéo dài ra khỏi bức ảnh nên sử dụng kernel phù hợp với phần lưới là dọc hay ngang.

Sử dụng tương tự khi nhận diện lưới dọc. Về phần kích thước bên trong hàm getStructuringElement, đó là kích thước rút ra được từ nhiều lần thử và tùy theo khả năng xử lý ảnh của từng người và độ mờ của bức ảnh.

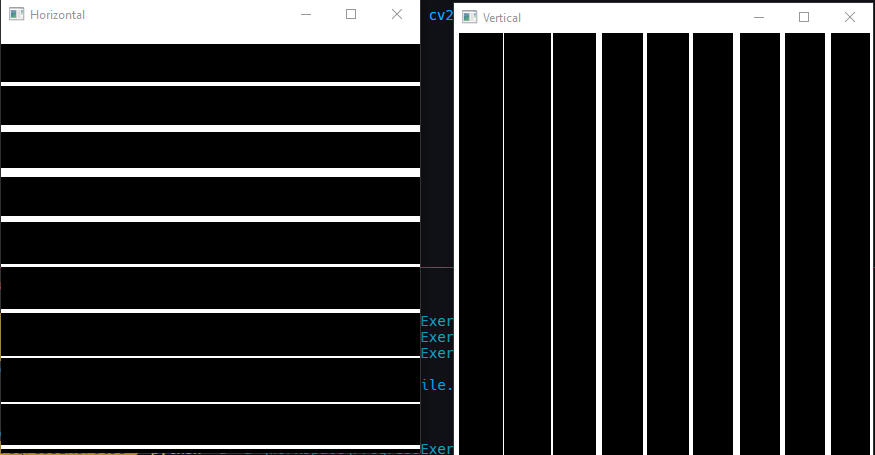
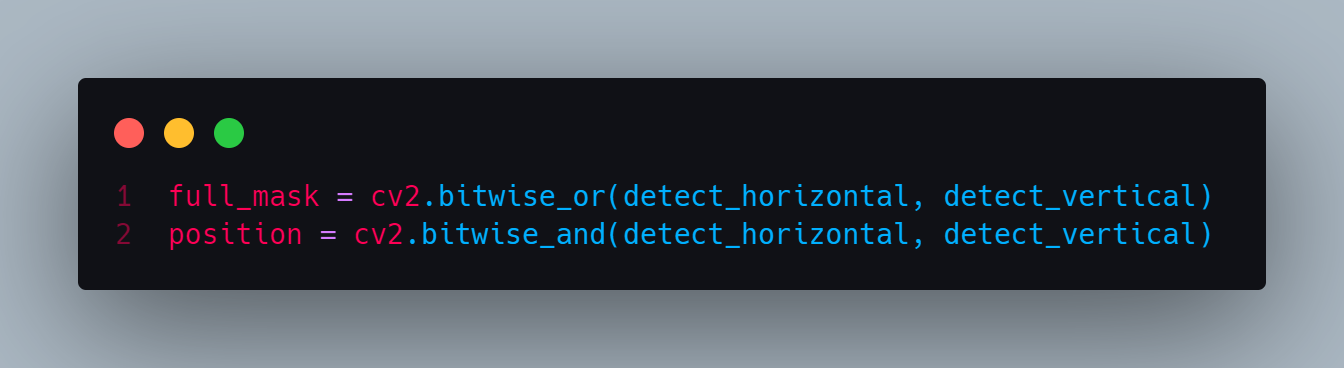


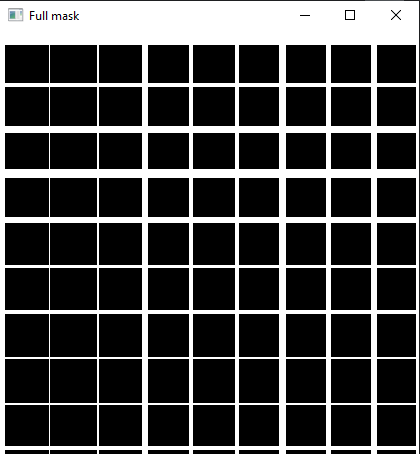
Image 3. Kết quả của nhận diện lưới

1.6 Xác định vị trí số



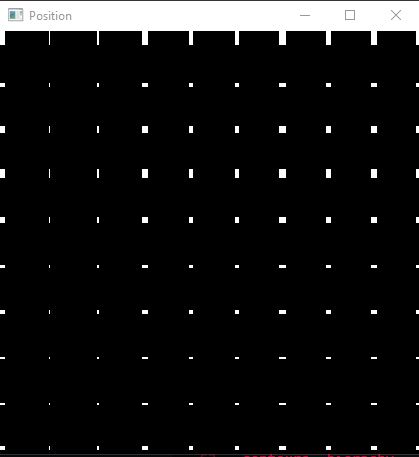
Khi có đầy đủ hai phần lưới, ta có hai cách để xác định vị trí của số dựa theo việc sử dụng hàm bitwise.

Với bitwise\_or:



Ta sẽ có hình ảnh đầy đủ của lưới sudoku, việc tiếp theo có thể sử dụng contour và bounding box để xác định vị trí của ô, sau đó lại tiếp tục dùng contour và bounding box để tìm tọa độ của số và xác định nó thuộc box nào

Với bitwise\_and:



Khi sử dụng or, ta có thể tìm ra được các giao điểm của từng ô, trong phần code sẽ đi theo hướng sử dụng giao điểm này.

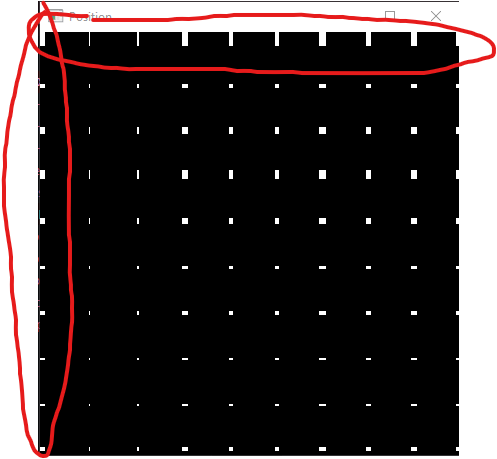


Sau khi đã có được giao điểm của các phần lưới. Ta có thể sử dụng kết hợp contour và bounding box để tìm được tọa độ của từng giao điểm, sau đó dùng hai list trống để lưu lại các tọa độ x và y này.



Sau khi đã có được x và y của từng giao điểm, việc tiếp theo cần làm đó là sử dụng sort để sắp xếp x và y lại, tuy nhiên x và y sẽ đi theo cặp và không phải theo thứ tự nên việc sort này sẽ làm mất các cặp x và y của giao điểm.

Mục đích của việc sắp xếp này để có thể lọc ra những tọa độ x và y trùng nhau khi đó chúng ta sẽ chỉ còn lưu lại các giá trị ở đầu của hai lưới

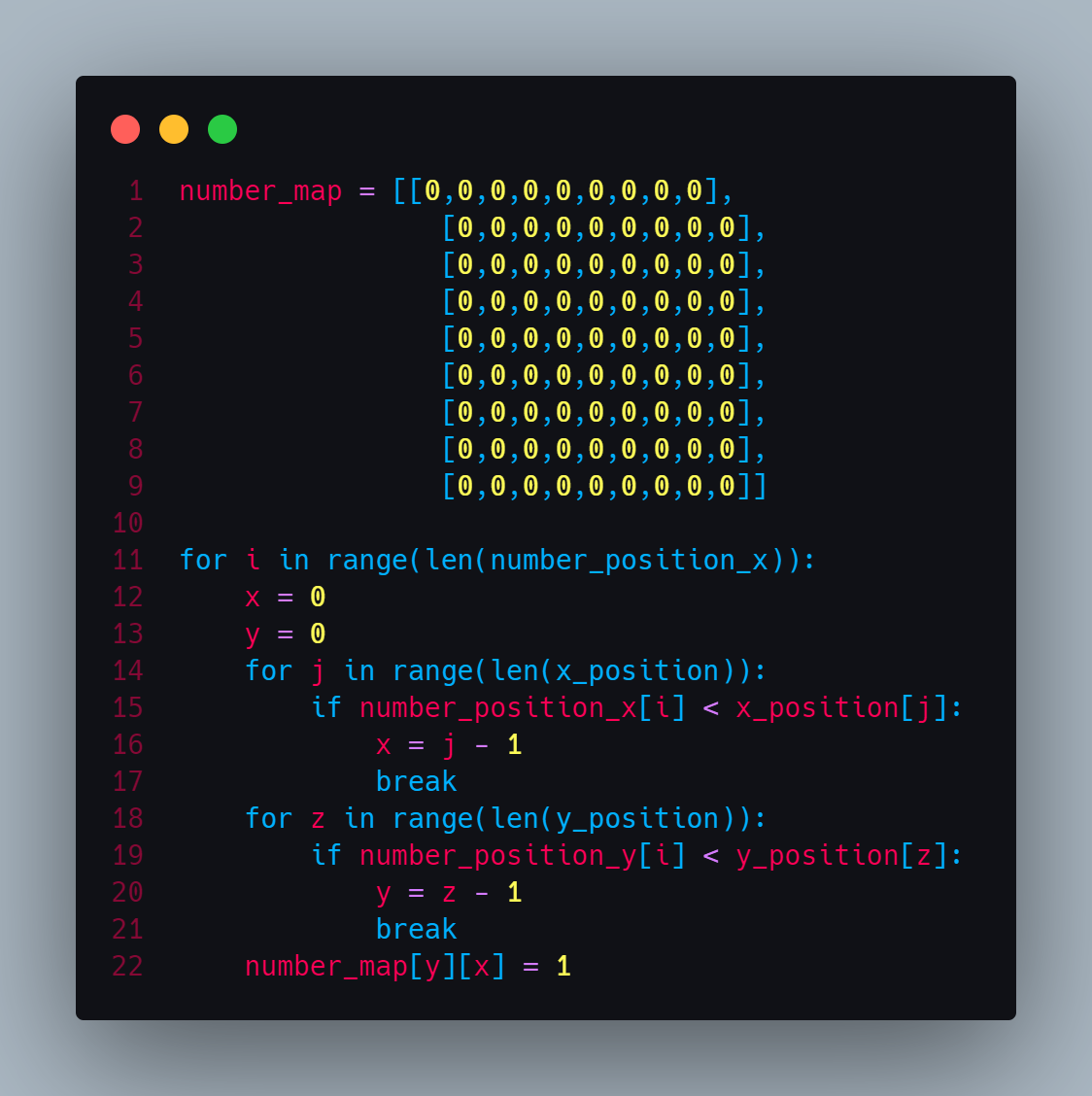


Khi tìm được tọa độ của số, ta sẽ có thể so sánh tọa độ hai cột của đó nhỏ hơn ở cột nào thì sẽ tìm ra được ô của nó ở vị trí ô kề bên.



Tiếp theo là việc xử lý hình ảnh trước khi tiếp tục sử dụng contour và bounding box để có thể tìm được vị trí con số. Khi duyệt trên list các contour thì code sử dụng hàm boundingRect() để tìm tọa độ ngay tại thời điểm đó, kèm theo thông số chiều dài và chiều rông. Trước khi xác định được giá trị của hình chữ nhật bao xung quanh một số là bao nhiêu, code đã được chạy với hàm rectangle() để dò được giá trị width và height của số. Sau đó, thay vì vẽ hình ảnh, hàm vẽ được được thay thế bằng việc thêm vào hai list có sẵn hai tọa độ của countor được tìm thấy.

Hai list này khá đặc biệt là các index trùng nhau sẽ là một cặp tọa độ của chữ số nên việc sắp xếp lại hai list sẽ gây ra xáo trộn các tọa độ.

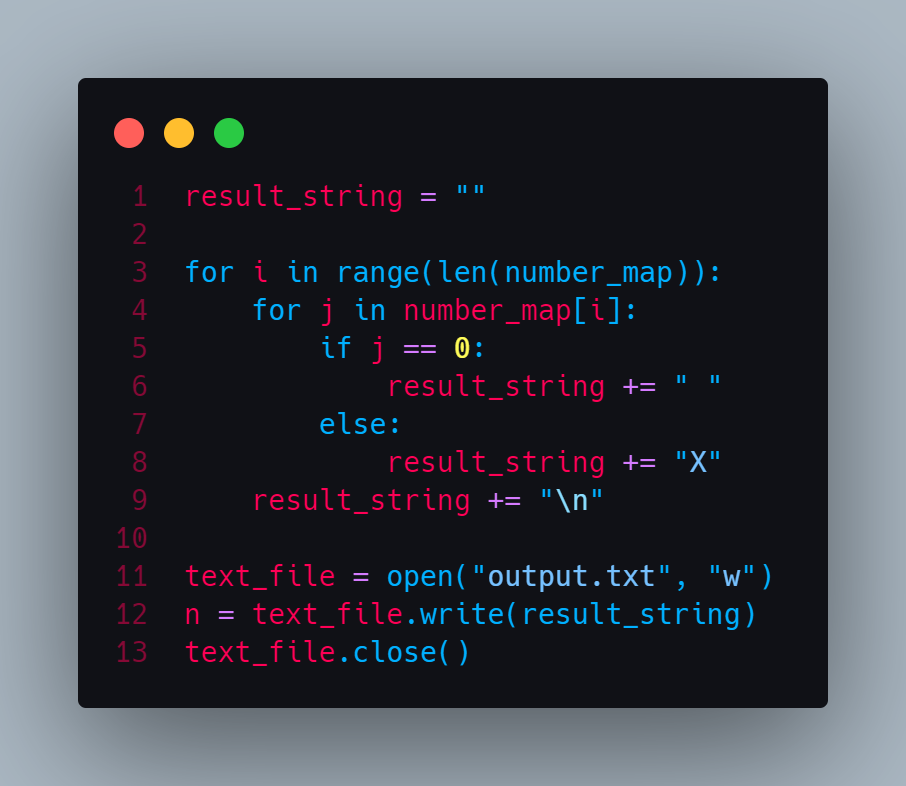


Trước khi bắt đầu chạy vòng lặp duyệt qua list tọa độ, một biến mang tên number\_map sẽ được tạo, biến này có kích thước là 9x9, tương đương với một bàn sudoku, khi xác định được cột và hàng của con số thì chỉ cần thay đổi ở index tương tự là có thể tìm ra được sơ đồ của trò chơi.

Sau khi đã tạo được biến map, vòng lặp đầu tiên sẽ được chạy với biến chạy là index của tạo độ bên trong biến number\_position\_x, tiếp theo hai vòng lặp con sẽ được chạy để dò vị trí theo trục x và trục y. Logic ở vòng lặp này là việc phát hiện tọa độ số nhỏ hơn tọa độ giao điểm tương đương với việc đã đi qua tọa độ giao điểm của con số đó 1 lần. Index lưu lại sẽ cần phải trừ đi 1 để ra được kết quả chính xác. Trong vòng lặp sẽ có lệnh break vì khi tiếp tục chạy, tất cả tọa độ phía sau sẽ đều lớn hơn tọa độ của con số.

Khi kết thúc hai vòng lặp trên trục, ta sẽ thay đổi giá trị bên trong biến map với số index trên biến lưu trùng với số hàng và cột của con số trên bảng trò chơi đó.

1.7 Lưu file và hoàn thành đồ án



Trước khi lưu thành file, một biến string sẽ được tạo để lưu giá trị ghi vào file.

Hai vòng lặp lồng nhau sẽ được sử dụng để duyệt qua mảng map hai chiều, khi đọc tới biến có giá trị là 1 sẽ thêm vào chuỗi một ký tự X, các số còn lại vòng lặp sẽ thêm vào ký tự “ “, khi hết một dòng, chuỗi sẽ thêm ký tự đặc biệt “\n” để chuỗi xuống dòng.

Tiếp theo là mở file tên output.txt, khi file này không có sẽ được code tự động ghi thêm và sử dụng hàm write để ghi chuỗi vào, sau cùng là đóng file lại.