TRƯỜNG ĐẠI HỌC
Sư phạm Thành phố Hồ Chí Minh
HCMC University of Education

IMAGE SEGMENTATION WITH GRABCUT ALGORITHM

Nguyễn Xuân Tính, Phạm Nguyễn Thùy Trang, Lâm Phát Tài

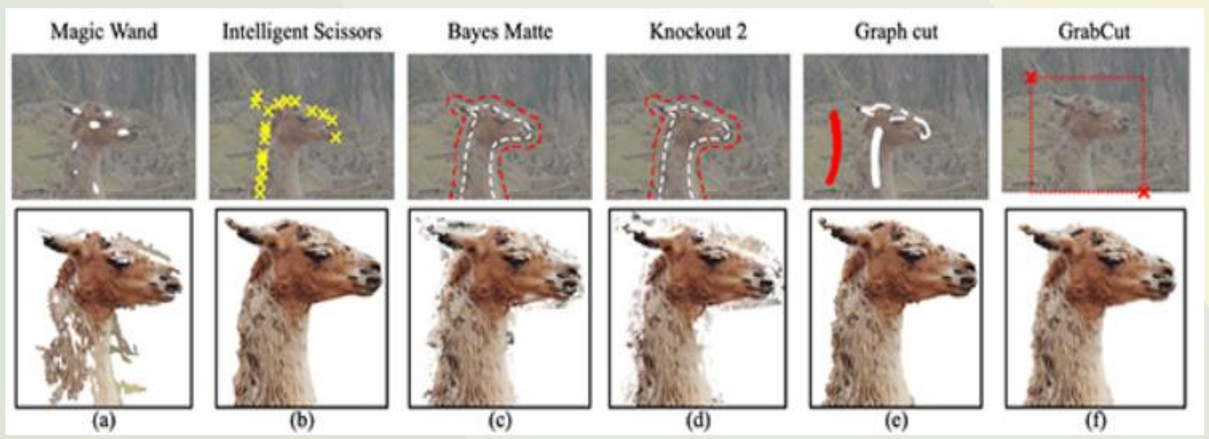
Abstract

Trong lĩnh vực xử lí ảnh, phân đoạn tiền cảnh và hậu cảnh là một vấn đề được mọi người hết sức quan tâm. Segmentation GrabCut giúp chúng ta có thể trích xuất được đối tượng chính và phụ trong bức ảnh của mình, nhờ đó người dùng và hơn nữa là máy tính có thể dễ dàng tiếp nhận được thông tin của bức ảnh. Một bức ảnh thông thường sẽ mang rất nhiều thông tin khác nhau, nhưng đa phần trong số đó chúng ta không sử dụng đến, do đó việc phân đoạn được tiền cảnh và hậu cảnh của bức ảnh sẽ nâng cao hiệu quả trong việc xử lí ảnh.

Introduction

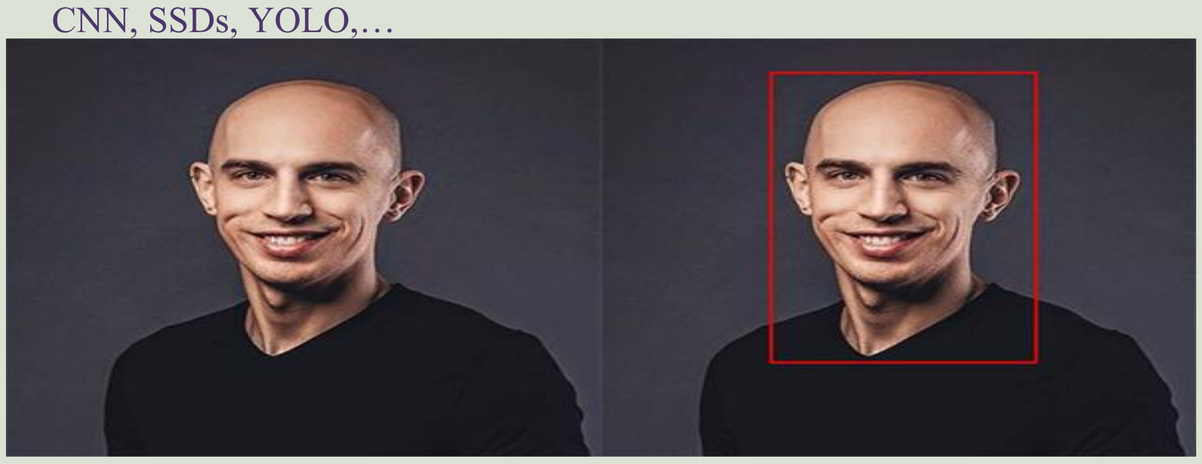
Segmentation GrabCut là một phương thức giúp chúng ta thực hiện trích xuất, phân đoạn tiền cảnh và hậu cảnh của một bức ảnh một cách chính xác. GrabCut tiếp nhận một bức ảnh đầu vào từ người dùng với một bounding box dùng để xác định vị trí của đối tượng trong bức ảnh mà chúng ta muốn phân đoạn hoặc một mask gần đúng với phân đoạn. Trong đề tài này, chúng em sẽ thực hiện phân đoạn ảnh bằng phương thức sử dụng bounding box, chúng em hi vọng rằng sẽ trích xuất được đối tượng mà mình mong muốn bằng cách vẽ một hình chữ nhật bao quanh đối tượng mà chúng em muốn trích xuất.

Proposed Method



Tiến hành, chúng em đề xuất sử dụng thư viện OpenCV thông qua hàm cv2.grabCut. Để bắt đầu thực hiện GrabCut với OpenCV bằng phương thức sử dụng bounding box. Đầu tiên, chúng em sẽ truyền một ảnh đầu vào mà chúng em muốn xử lí, sau đó xác định một bounding box của đối tượng mà chúng em muốn phân đoạn trong bức ảnh, bounding box có thể được tạo ra bằng nhiều cách khác nhau, sau đây là một vài cách phổ biến nhất.

- Xem xét bức ảnh một cách thủ công và dán nhãn tọa độ (x, y) cho bounding box.
- Áp dụng a Haar cascade.
- Sử dụng HOG + Linear SVM để phát hiện đối tượng.
- Sử dụng các công cụ phát hiện đối tượng dựa trên deep learning như: Faster R-CNN, SSDs, YOLO,…

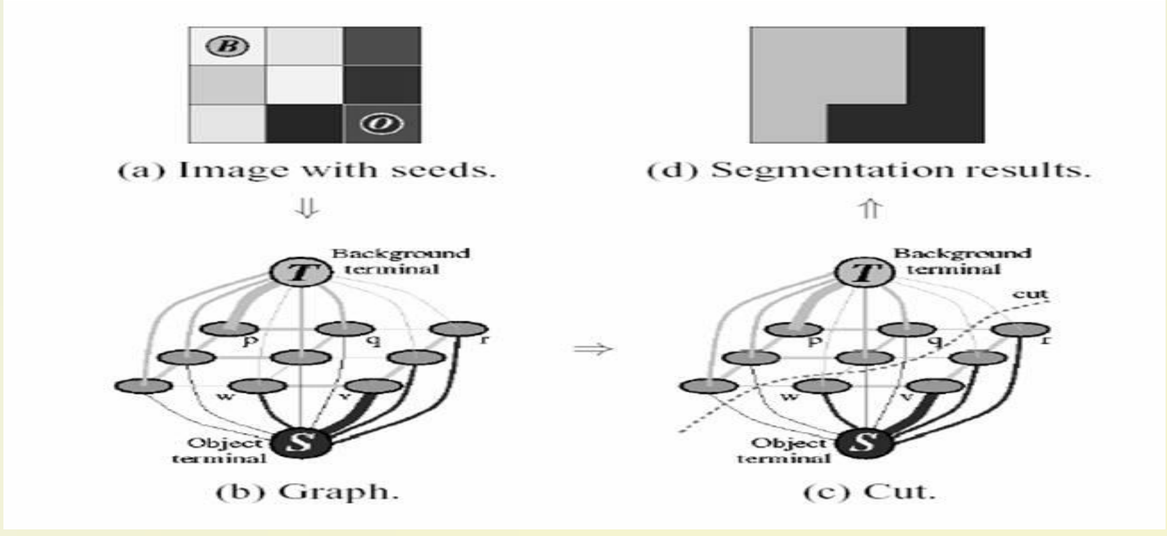


GrabCut dễ dàng kết hợp với các thuật toán trên miễn là các thuật toán đó tạo ra các bounding box.

Ở đây, mọi thứ bên ngoài bounding box sẽ được xem là hậu cảnh và bên trong bounding box sẽ được xem là tiền cảnh. Khi bounding box được vẽ nên thì tiền cảnh và hậu cảnh sẽ là nhãn cứng có nghĩa là chúng sẽ không thay đổi trong quá trình xử lí.

Tiếp theo, Gaussian Mixture Model (GMM) được sử dụng để tạo mô hình tiền cảnh và hậu cảnh. GMM sẽ học và tạo phân phối pixel mới, có nghĩa là các pixel không xác định có thể được gắn nhãn là tiền cảnh hoặc hậu cảnh tùy thuộc vào mối quan hệ của nó với các pixel được gắn nhãn cứng theo thống kê màu (nó tương tự như phân cụm).

Một đồ thị được xây dựng từ phân phối pixel này. Các node trong biểu đồ là pixel, có hai node được thêm vào là Source node và Sink node, các pixel tiền cảnh được kết nối với Source node và các pixel hậu cảnh được kết nối với Sink node. Trọng số giữa các pixel được xác định bằng thông tin về cạnh hoặc độ tương phản của pixel. Nếu có sự khác biệt lớn về màu sắc pixel, cạnh giữa chúng sẽ có trọng số thấp.



Sau đó, một thuật toán mincut được sử dụng để phân đoạn đồ thị, nó cắt đồ thị thành hai node riêng biệt là source node và sink node với hàm chi phí tối thiểu. Hàm chi phí là tổng của tất cả các trọng số của các cạnh bị cắt. Sau khi cắt, tất cả các pixel được kết nối với Source node trở thành tiền cảnh và những pixel được kết nối với Sink node trờ thành hậu cảnh (nền).

Quá trình này được tiếp tục cho đến khi phân loại được tiền cảnh và hậu cảnh.

Algorithm

Thuật toán GrabCut được xây dựng dựa trên GraphCut. Thuật toán GraphCut phân đoạn đối tượng từ một hình ảnh được cho bằng cách giảm thiểu năng lượng (energy minimization). Đối với mỗi điểm ảnh (pixel) z_i trong hình ảnh, có một giá trị độ mờ (opacity) $a_i \in \{0, 1\}$ (với 0 cho nền sau và 1 cho tiền cảnh). θ_f, θ_b lần lượt mô tả sự phân bố hình ảnh tiền cảnh và hình ảnh nền sau, cũng là biểu đồ tần suất (histogram) của hai loại giá trị pixel (điểm ảnh tiền cảnh hoặc điểm ảnh nền).

$$\theta = \{h(z;\ a), = 0, 1\}, \text{trong đó} \int h(z, a) = 1$$

GraphCut định nghĩa một hàm năng lượng (energy function) E là hàm mất (loss) của nhiệm vụ phân đoạn bằng cách sử dụng hình thức năng lượng "Gibbs". Mục đích để giảm thiểu E.

$$E(a, \theta, z) = U(a, \theta, z) + V(a, z)$$

Điều kiện data U đánh giá mức độ phù hợp của α và z dựa trên biểu đồ tần suất (histogram) nhất định θ (θf hoặc θb).

$$U(a, \theta, z) = \sum_i \log h\,(a_i, z_i)$$

Điều kiện smoothness V đánh giá độ mượt của ranh giới phân đoạn. Trong đó B(i, j) cao khi các giá trị của pixel i và pixel j giôn nhau. Tham khảo [1] để biết định nghĩa chi tiết về B(i, j).

$$V(a, z) = \gamma \sum_{(i,j) \in \text{img pixel}} [a_i \neq a_j]\, B_{(i,j)}$$

Nhiệm vụ phân đoạn sử dụng kỹ thuật Min Cut để tìm mức tối thiểu toàn cục nhằm giải quyết vấn đề tối ưu hóa.

$$\hat{\alpha} = argmin\ E(a, \theta, z)$$

GrabCut cải tiến hơn GraphCut chủ yếu ở hai khía cạnh. Đầu tiên, GrabCut thay thế biểu đồ bằng GMM (Mô hình Hỗn hợp Gaussian) trong điều kiện data U, GMM có thể được sử dụng cho không gian màu sắc. Thứ hai, tương tác với người dùng lặp đi lặp lại trong GrabCut giúp thuật toán đạt được hiệu quả tốt hơn. Có hai loại GMMs, một cho các điểm ảnh nền và một cho các điểm ảnh tiền cảnh, mỗi GMM có các thành phần K (thường là K = 5). Vì vậy:

1. Hàm năng lượng E trong GraphCut trở thành:

$$\mathrm{E}(a, k, \theta, \mathrm{z}) = \mathrm{U}(a, k, \theta, \mathrm{z}) + \mathrm{V}(a, z), \text{ trong đó } \mathrm{k} \in \{1, \ldots \mathrm{K}\}.$$

2. Điều kiện data U (α, k, θ, z) trở thành:

$$\mathrm{D}(a_i, k_i, \theta_i, \mathrm{z}_i) = -\log_{\pi}(a_i, k_i) + \frac{1}{2}\log\det\sum(a_i, k_i) + \frac{1}{2}[\mathrm{z}_i - \mu(a_i, k_i)]^T \sum(a_i, k_i)^{-1} [\mathrm{z}_i - \mu(a_i, k_i)]$$

$$\mathrm{U}(a, k, \theta, \mathrm{z}) = \sum_i \mathrm{D}(a_i, k_i, \theta_i, \mathrm{z}_i)$$

Trong đó $\mu(a_i, k_i)$ là trung bình của các giá trị và $\sum(a_i, k_i)$ là ma trận hiệp phương sai.

Điều kiện smoothness $\mathrm{V}(a, z)$ giống như trong GraphCut. Sau khi khởi tạo, Thuật toán GrabCut được thực hiện với các bước sau:

1. Gán các thành phần GMM cho các điểm ảnh: Áp dụng cụm "k-means" cho các điểm ảnh tiền cảnh và nền được phân đoạn tương ứng bằng tay của người dùng. Số lượng các cụm là K. Với mỗi điểm ảnh z_i ở tiền cảnh và nền tương ứng, gán k_i với giá trị $\mathrm{D}(a_i, k_i, \theta_i, \mathrm{z}_i)$ tối thiểu.

$$k_i = \operatorname{argmin} \mathrm{D}(a_i, k_i, \theta_i, \mathrm{z}_i)$$

2. Học các parameter của GMM từ các điểm ảnh z. Đối với các điểm ảnh tiền cảnh được gán cho k_i, $\mu(a_i, k_i)$ và $\sum(a_i, k_i)$. Áp dụng cùng một phương pháp xử lý cho các điểm ảnh nền.

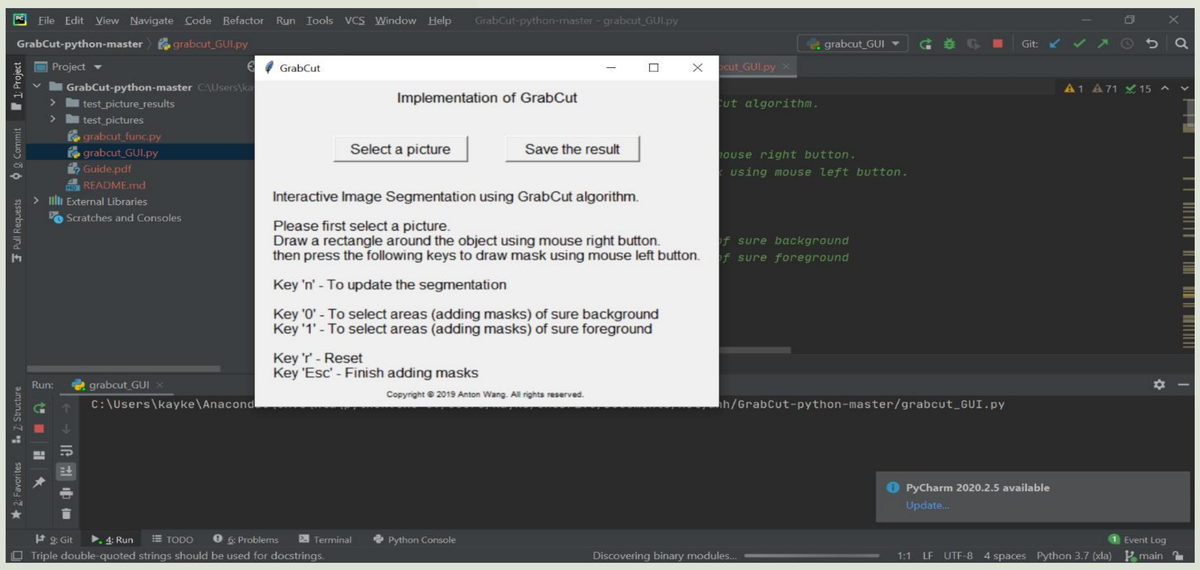
$$\theta = \operatorname{argmin}$$

3. Xây dựng đồ thị và tiến hành Min Cut.

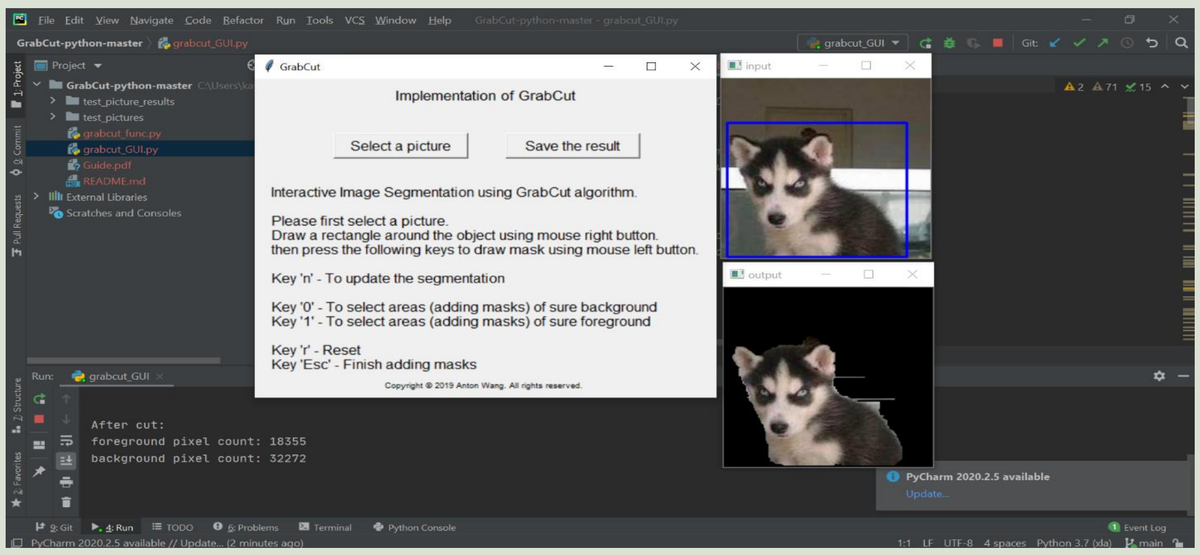
$$\min \mathrm{E}(a, k, \theta, \mathrm{z})$$

4. Tương tác với người dùng. Lặp lại các bước trên.

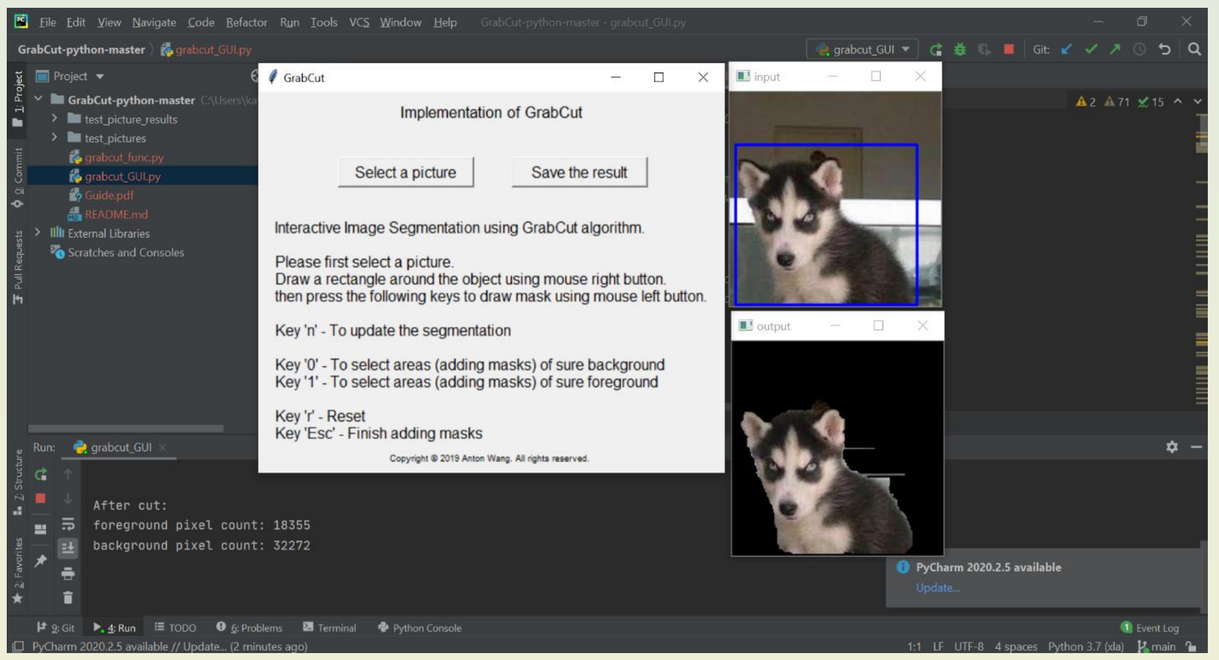
Experiment



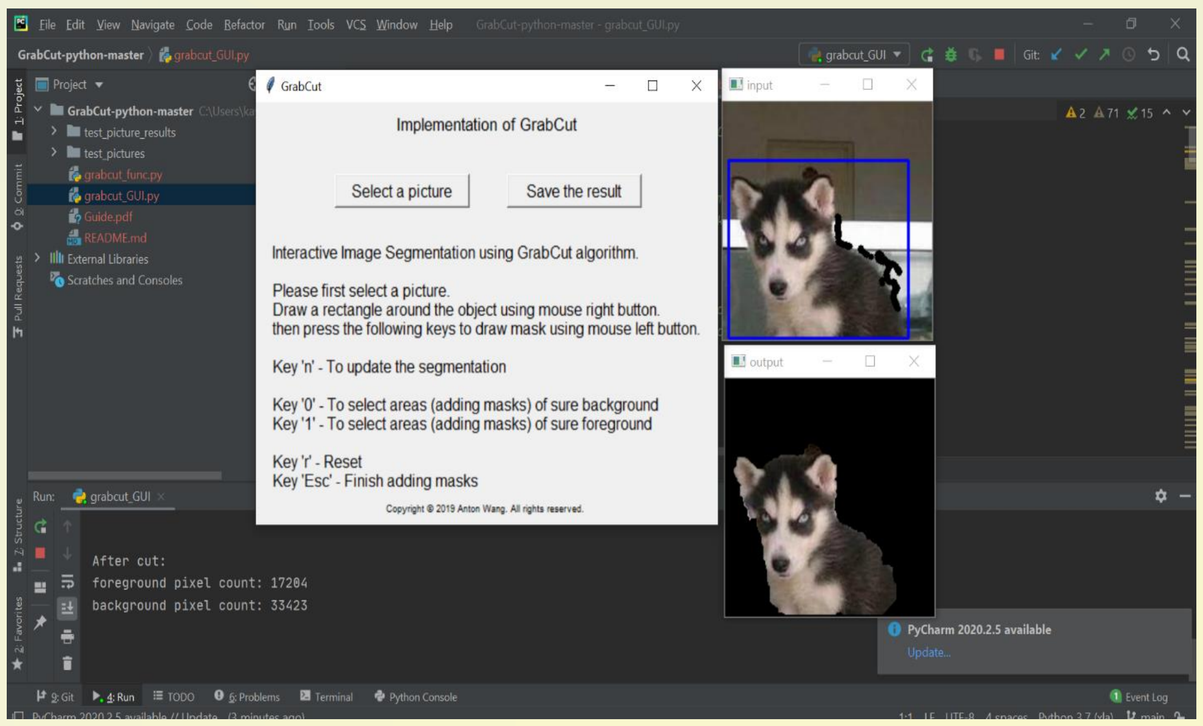
Trước tiên, nhấn nút "Select a picture" và chọn ảnh. Các hình ảnh mẫu được lưu trữ trong thư mục .\GrabCut\test_pictures. Ví dụ, chọn .\GrabCut\test_picture\dog2.jpg. Sau đó, hai cửa sổ sẽ bật lên hiển thị hình ảnh đầu vào và hình ảnh đầu ra.



Vẽ một hình chữ nhật xung quanh đối tượng bằng cách sử dụng nút chuột phải. Sau đó, một hộp thư sẽ bật lên. Làm theo hướng dẫn của hộp thư. Nhấn phím "n", chờ vài giây và kết quả ban đầu sẽ được hiển thị trong cửa sổ "output".



Tiếp theo, nhấn phím "0" để vẽ đường kẻ (màu đen) cho nền hoặc nhấn phím '1' để vẽ đường kẻ (trắng) cho tiền cảnh. Hãy nhớ nhấn phím "n" khi vẽ xong. Và chờ đợi trong vài giây. Ở đây, ví dụ chúng ta sẽ vẽ đường kẻ đen cho nền.



Nhấn phím "Esc" khi hoàn tất tất cả các bước. Kết quả có thể được lưu trong \GrabCut\grabcut_output.png bằng cách nhấn nút "Save the result" trên giao diện. Nhấn phím "r" để reset quá trình phân đoạn. Vui lòng tham khảo ghi chú chương trình của .\GrabCut\grabcut_GUI.py và .\GrabCut\grabcut_func.py để có thêm thông tin hướng dẫn thực hiện. File .\GrabCut\grabcut_GUI.py chứa code xây dựng giao diện và file .\GrabCut\grabcut_func.py chứa hai class grabcut và GMM, dùng để xây dựng thuật toán GrabCut.

Conclusion

Trong đề tài này, chúng ta đã biết cách sử dụng thuật toán GrabCut để phân đoạn tiền cảnh và hậu cảnh trong một bức ảnh. Thuật toán này rất hữu ích trong việc giúp người dùng loại bỏ những phần dư thừa hoặc nhiễu khỏi các bức ảnh. Đối với lĩnh vực xử lí ảnh, đây là một thao tác vô cùng quan trọng vì một bức ảnh chứa quá nhiều thông tin dư thừa hoặc nhiễu sẽ ảnh hưởng xấu đến các quá trình xử lí tiếp theo của bức ảnh.

Lời sau cùng, chúng em xin gửi lời cảm ơn đến **Thầy Ngô Quốc Việt** đã giúp em và các bạn hoàn thành môn học Xử Lí Ảnh. Môn học này mang đến những kĩ năng quan trọng giúp chúng em có thể học tốt hơn trong các môn học tiếp theo.

References

Rother, C., Kolmogorov, V., & Blake, A. (2004, August). Grabcut: Interactive foreground extraction using iterated graph cuts. In ACM transactions on graphics (TOG) (Vol. 23, No. 3, pp. 309-314). ACM.

BLAKE, A., ROTHER, C., BROWN, M., PEREZ, P., AND TORR, P. 2004. Interactive Image Segmentation using an adaptive GMMRF model. In Proc. European Conf. Computer Vision.

BOYKOV, Y., AND KOLMOGOROV, V. 2003. Computing Geodesics and Minimal Surfaces via Graph Cut. In Proc. IEEE Int. Conf. on Computer Vision.

Github: *https://github.com/AntonotnaWang/GrabCut-python.git*