XÓA PHÔNG NỀN ẢNH VÀ VIDEO SỬ DỤNG PHƯƠNG PHÁP MẶT NẠ MÀU

Mai Văn Thiên Phước
20521776

Xoá phông nền ảnh giúp chúng ta loại bỏ được những chi tiết không mong muốn, hoặc đơn giản là để có thể ghép hình ảnh mình vào những nền khác đẹp mắt hơn. Những phương pháp xóa nền cũng như các công cụ xóa nền ngày nay đều có khả năng xóa nền tốt, nhưng tốc độ xử lý lại chậm bởi quá trình thực hiện phức tạp. Các quy trình thực hiện phức tạp có thể cho ra kết quả xóa nền tốt ở ảnh nhưng đối với xóa nền video lại có nhiều hạn chế bởi lượng khung hình nhiều và liên tục, điều này đòi hỏi chúng ta cần phương pháp mới có hiệu suất cao và tính toán ít phức tạp hơn để có thể xóa nền cho video.

Báo cáo này trình bày phương pháp xóa nền ảnh dựa trên phân đoạn màu sắc thông qua việc tạo mặt nạ màu. Phương pháp đơn giản đi kèm với tốc độ xử lý nhanh chóng, dễ tiếp cận và cài đặt, thích hợp cho việc áp dụng xóa nền ảnh lẫn video.

I. GIỚI THIỆU

Thực hiện xóa nền vận dụng mặt nạ màu (color mask) nhằm cô lập, chỉnh sửa phần màu cần thiết của ảnh và loại bỏ phần màu còn lại khỏi ảnh và video.

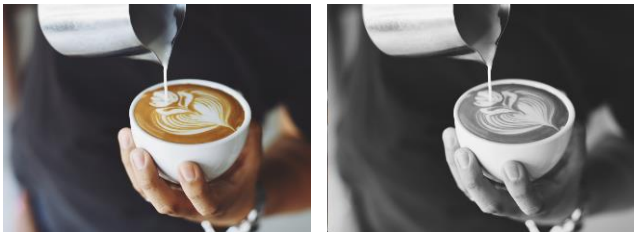
Mặt nạ màu giới hạn phạm vi màu được hiển thị bằng các sắc thái cụ thể. Dựa vào tác dụng đó, nếu coi nền (background) là phần cần ẩn đi và chỉ hiển thị phần còn lại không thuộc nền, chúng ta sẽ hiện thực hóa được xóa nền ảnh (remove background).

II. NỘI DUNG

Việc xây dựng mặt nạ màu thường được thao tác trực tiếp trên không gian màu RGB. Phần lớn những tấm ảnh lẫn video kĩ thuật số ngày nay đều có nhiều màu phức tạp. Trong quá trình xử lý video, các chuyển động của vật thể cũng như sự thay đổi liên tục của màu sắc trong mỗi khung ảnh của video tạo áp lực lớn cho phần cứng, dẫn tới sự suy giảm hiệu suất xử lý và tốc độ cho ra kết quả. Để giải quyết vấn đề này chúng ta cần biện pháp nhằm giải quyết sự phức tạp màu sắc của dữ liệu đầu vào.

Bước 1. Gray Scale

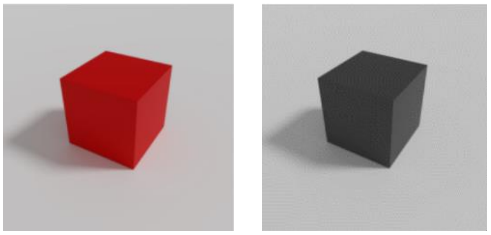
Thực hiện chuyển đổi các khung ảnh đầu vào từ không gian màu RGB sang không gian màu GRAY, giúp đơn giản hóa dữ liệu và loại bỏ sự phức tạp màu sắc của ảnh, là một biện pháp tốt cho vấn đề gặp phải.



Ảnh gốc (RGB) Ảnh xám (GRAY)

Hình 1

Để thực hiện tạo mặt nạ màu, chúng ta cần xác định được khoảng màu hiển thị, đây là yếu tố cần thiết để tạo nên sự phân đoạn màu sắc giữa vùng cần hiển thị và vùng cần loại bỏ.



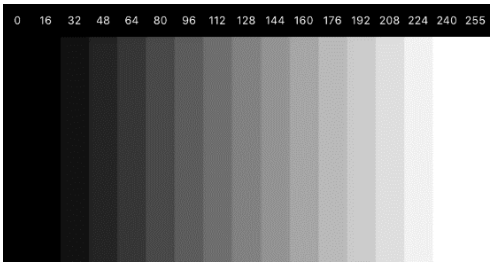
Ảnh gốc (RGB) Ảnh xám (GRAY)

Hình 2

Kết quả của xóa nền ảnh sẽ giữ lại những vật thể hoặc các yếu tố nổi bật của ảnh, và xóa đi phần còn lại, đây là yếu tốt tiền đề để xác định khoảng màu cần thiết cho mặt nạ màu. Quan sát hình 1 và 2, chúng ta có thể thấy, vật thể nổi bật trong cả 2 hình đều có sự khác biệt màu rõ rệt so với phần màu nền.

Bước 2. Tạo mặt nạ màu

Ở hình 1, vật thể nổi bật (ly coffee) sẽ có tông màu sáng và trắng (gần tới 255) và phần nền tối tương phản (tới gần 0) chiếm phần lớn các pixel của ảnh. Tương tự ở hình 2, vật thể nổi bật (khối lập phương) sẽ có tông màu tối và phần nền sáng tương phản chiếm phần lớn các pixel. Thông qua đó chúng ta có thể thiết lập hai khoảng màu phù hợp cho mặt nạ màu, một cho ảnh nền sáng và một cho ảnh nền tối.



Hình 3

Thực hiện tạo mặt nạ màu sử dụng OpenCV inRange() để cung cấp cho GRAY phạm vi màu thấp và cao cần được phát hiện trong hình ảnh để tạo mặt nạ. Hàm inRange() trả về một mảng bao gồm các phần tử bằng 0 (đen) nếu các phần tử của mảng nguồn nằm giữa các phần tử của hai mảng đại diện cho cận trên và cận dưới.

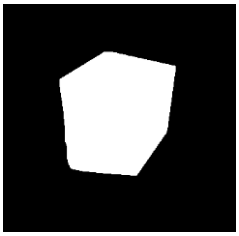
Quan sát hình 3 về sự thay đổi màu sắc trong không gian màu GRAY, ta xác định được phần màu tối (đen) sẽ có giá trị dao động từ 0 đến 128, và phần màu sáng (trắng) sẽ từ 128 đến 255. Vật thể tối sẽ nổi bật trong nền ảnh sáng, nên chúng ta cần mặt nạ màu trong khoảng sáng để để ẩn phần nền và ngược

lại, vật sáng sẽ nổi bật trong nền tối, cần mặt nạ màu trong khoảng tối. Từ đó ta xây dựng mô hình thuật toán đơn giản để phân loại ảnh có nền sáng và ảnh có nền tối như sau:

A = cv2.inRange(gray) $\in$ [0; 128]
B = cv2.inRange(gray) $\in$ [128; 255]
Nếu: (Tổng số giá trị = 0 $\in$ A > Tổng số giá trị = 0 $\in$ B)
=> mask = cv2.inRange(ảnh) $\in$ [0; 128]
Ngược lại: mask = cv2.inRange(gray) $\in$ [128; 255]

Hiện thực thuật toán, ta thu được kết quả như sau cho hình 1 và 2:


Mặt nạ màu cho hình 1


Mặt nạ màu cho hình 2

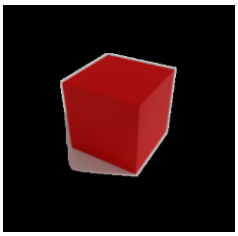
Hình 4

Bước 3. Xóa nền ảnh

Mặt nạ đã được tạo thành công, chúng ta sẽ đến với quá trình xóa bỏ nền của hình ảnh thông qua mặt nạ màu. Quan sát hình 4, chúng ta có thể nhận thấy được phần thông tin bị ẩn đi sẽ có màu đen, tương ứng với giá trị 0 (đen) trong khoảng màu GRAY [0:255], và phần nổi bật được hiển thị có giá trị bằng 255 (trắng). Chúng ta sẽ thay thế phần pixel có giá trị bằng 0 của mặt nạ vào vị trí tương ứng của nó trên ảnh gốc (hình 1 và 2), và kết quả thu được sẽ là phần vật thể nổi bật cần hiển thị.

Việc thay thế các giá trị 0 của mặt nạ lên ảnh gốc có thể hiểu theo một hướng khác, đó là thay thế phần pixel của ảnh gốc và vị trí tương ứng trên pixel có giá trị bằng 255, đều sẽ cho ra cùng một kết quả. Để thực hiện phương pháp này, chúng ta sẽ sử dụng OpenCV bitwise_and(), tương ứng với việc sử dụng thuật toán AND với từng pixel tương ứng của ảnh. Trên quy tắc đầu ra sẽ lần phần chung của cả 2 ảnh, thực hiện thuật toán AND giữa ảnh gốc(hình 1,2) và mặt nạ (hình 4) bằng lệnh bitwise_and() sẽ giữ nguyên pixel trên ảnh gốc có vị trí tương ứng với các pixel có giá trị bằng 255 trên mask, và ngược lại sẽ thay thế pixel có vị trí tương ứng với pixel của mask có giá trị bằng 0.


Hình 1 (Đã xóa nền)


Hình 2(Đã xóa nền)

Hình 5

Bước 4. Xóa nền video

Hiện thực hóa phương pháp mặt nạ màu bằng ngôn ngữ lập trình Python với thư viện OpenCV. Chương trình hoạt động trên môi trường được cung cấp bởi Anaconda.

+ Áp dụng xóa nền cho video webcam:


Ảnh từ webcam


Ảnh đã xóa nền

Hình 6

Quan sát hình ảnh từ webcam, nhận thấy ảnh thu được trong môi trường thiếu sáng, chủ thể xuất hiện trong ảnh được xem là vật thể nổi bật và được giữ lại trong mask, ngược lại phần nền tối xung quanh đã bị xóa đi.

Quan sát hình 7, ta có thể thấy ảnh hưởng đáng kể của độ sáng môi trường lên mặt nạ màu, khiến cho phần mặt nạ đã ẩn đi nhiều pixel cần thiết, dẫn tới kết quả thu được sau khi xóa nền không như mong đợi. Để giải quyết vấn đề này, chúng ta xử lý nhị phân cho mặt nạ màu(mask) trước khi thực hiện xóa nền.


Ảnh từ webcam


Mặt nạ màu


Ảnh đã xóa nền

Hình 7

Để có kết quả tốt nhất cho quy trình xử lý nhị phân, chúng ta sẽ thực hiện phương thức Morphological được cung cấp bởi OpenCV. Chúng ta sẽ sử dụng phương thức closing và dilation cho mặt nạ màu.


Closing


Dilation

Hình 8

Đầu tiên, chúng ta sử dụng phương thức closing để đồng hóa các pixel có giá trị bằng 0 có các pixel xung quanh bằng 255 đều trở thành 255. Sau đó sử dụng Dilation nhằm làm tăng vùng trắng của mặt nạ màu, khiến cho vật thể hiển thị rõ nét hơn.


Mặt nạ màu ban đầu


Mặt nạ màu sau xử lý

Hình 9

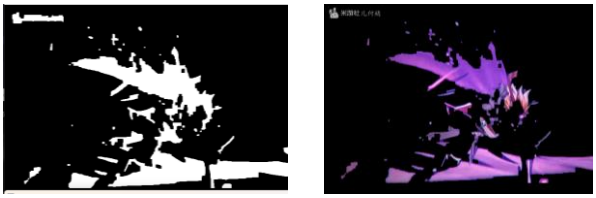
+ Áp dụng xóa nền cho video hoạt ảnh (animation):

Ảnh từ video Mặt nạ màu Ảnh đã xóa nền

Hình 10

Quan sát kết quả thu được từ hình số 10, có thể thấy sau khi thực hiện các phương thức Morphological đã khiến cho mặt nạ màu và kết quả thu được sau khi xóa nền đã tương đối đạt được mong đợi.

Tuy nhiên ở nhiều phân đoạn màu có màu sắc sặc sỡ hoặc nhiều vùng có các pixel sáng nằm rải rác sẽ làm tăng độ nhiễu của kết quả cuối cùng.



Mặt nạ màu Ảnh đã xóa nền

Hình 11

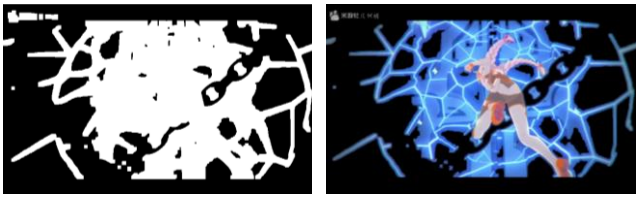
Quan sát hình 11, có thể dễ dàng nhận ra hai vấn đề trong khung hình này:

- Thứ nhất là hiệu quả của phương thức closing và dilation được thiết lập ở phần (a) vẫn chưa đủ, vì gặp tình huống ảnh có độ phức tạp màu cao với sự đan xen giữa các mảng màu sáng và tối trên vật thể nổi bật sẽ dẫn đến kết quả không như mong đợi, điều này yêu cầu chúng ta phải tiếp tục cải thiện closing và dilation bằng cách gia tăng số lần xử lý của hai phương thức này.
- Thứ hai là sự hiển thị của các pixel không cần thiết cho sự hiển thị của vật thể nổi bật, các pixel này nằm rải rác bên ngoài vùng trắng chính của mặt nạ màu, tạo nên hiện tượng nhiễu cho kết quả.

Opening

Hình 12

Để giải quyết vấn đề này chúng ta nên sử dụng thêm phương thức opening (Hình 12), đối lập với việc xử lý các pixel nằm bên trong vùng trắng như closing, opening sẽ giúp chúng ta ẩn đi nhưng pixel trắng nằm rải rác bên ngoài vùng trắng chính.

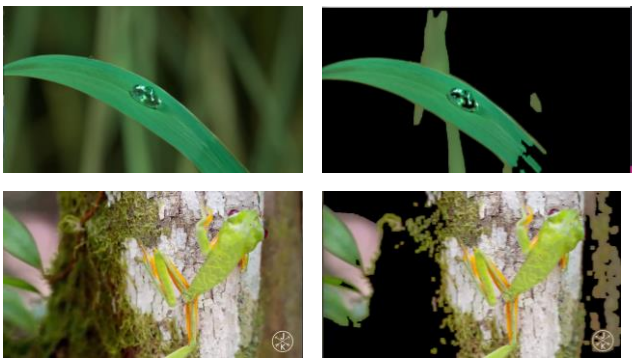


Mặt nạ màu Ảnh đã xóa nền

Hình 13

Sau khi thực hiện cài đặt cả ba phương thức xử lý Morphological, kết quả mới thu được đã được cải thiện rất nhiều.

+ Áp dụng xóa nền cho video ngoại cảnh:



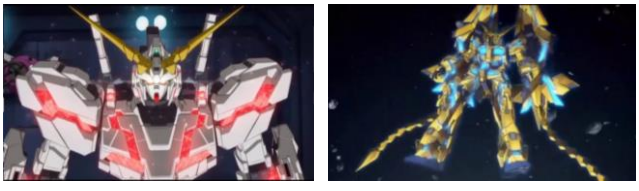
Ảnh ban đầu Ảnh đã xóa nền

Hình 14

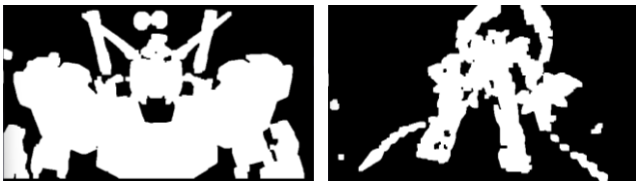
Quan sát hình 14, đối với video được quay từ ngoại cảnh, môi trường có độ sáng cao và màu sắc phức tạp gây nhiều tác động lên việc tạo mặt nạ màu không tốt, phần vật thể nổi bật chưa được xác định đúng, từ đó thu được kết quả xóa nền không như mong đợi.

Bước 5. Nhận xét

Phương pháp xóa nền ảnh sử dụng mặt nạ màu (color masking) là ý tưởng dựa trên phân đoạn màu sắc (color segmentation) có cấu trúc đơn giản mang lại hiệu suất xử lý cao khi xử lý các video kĩ thuật số. Phương pháp này hoạt động rất tốt đối với ảnh đơn giản và các video hoạt ảnh (hình 15), nhưng đồng thời cũng rất yếu khi gặp các ảnh có nhiều màu sắc phức tạp, ảnh ngoại cảnh và rất nhạy cảm với ánh sáng từ môi trường (hình 14).



Ảnh ban đầu



Mặt nạ màu

Ảnh sau khi xóa nền

Hình 15

III. THỰC HIỆN THAY THẾ NỀN CHO ẢNH ĐÃ XÓA NỀN

Thực hiện thay thế phần nền bị xóa đi bằng ảnh nền mới: thay thế các pixel có giá trị bằng 0 của ảnh sau khi xóa nền bằng cái pixel tại vị trí tương ứng của ảnh nền mới, từ đó thu được kết quả ảnh thay nền.

Bước 1. Tạo mặt nạ màu

Thực hiện xóa nền ảnh, với đầu ra là ảnh đã xóa nền cùng với mặt nạ màu (mask).

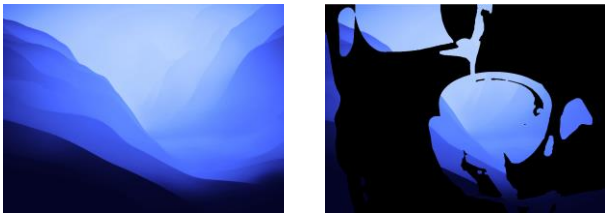
Ảnh ban đầu Mặt nạ màu Ảnh đã xóa nền

Hình 16

Bước 2. Chèn phông nền mới vào mặt nạ màu

Sử dụng lệnh OpenCV cv2.bitwise_or() với đầu vào là ảnh nền mới, lấy mặt nạ màu là mask đã thu được từ bước 1, thực hiện thuật toán OR với các pixel tương ứng của mask ban đầu.

Các pixel của ảnh nền mới ứng với vị trí của các pixel có giá trị bằng 255 (trắng) trên mask sẽ được giữ nguyên, và ứng với vị trí bằng 0(đen) sẽ bị thay thế.



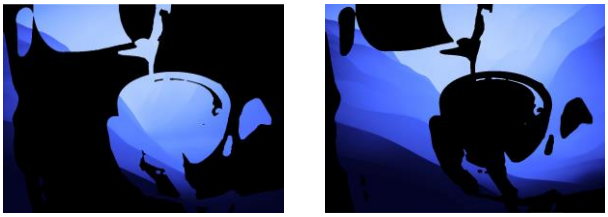
Ảnh nền mới Sau khi dùng cv2.bitwise_or()

Hình 16

Bước 3. Tạo mặt nạ màu mới

Sử dụng lệnh OpenCV cv2.bitwise_xor() với đầu vào là kết quả thu được sau khi thực hiện bước 2(-> rest) để tiến hành XOR với phần ảnh nền mới.

Khi thực hiện XOR, các pixel có giá trị bằng nhau ở rest và ảnh nền mới sẽ bị thay bằng 0, ngược lại ở những vị trí mà các pixel có giá trị khác nhau sẽ được thực hiện phép toán OR. Theo góc nhìn logic, chúng ta có thể thấy rest là phần mask mới đã được thay pixel ở các vị trí thuộc vật thể nổi bật thành phần ảnh nền mới, nhưng mong muốn của chúng ta là giữ nguyên phần pixel ở khu vực này và thay các pixel ở khu vực ngoài thành các pixel của ảnh nền, vì thế việc thực hiện lệnh XOR để thu được kết quả đối nghịch của rest là phương hướng chính xác.



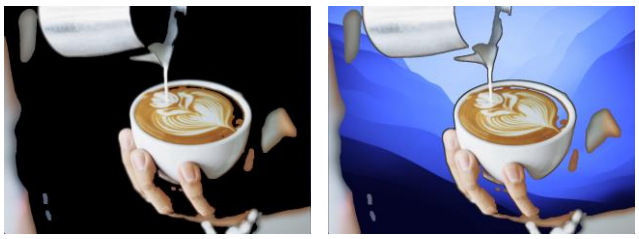
rest Sau khi dùng cv2.bitwise_xor()

Hình 17

Bước 4. Tạo ảnh thay nền

Sử dụng lệnh OpenCV cv2.bitwise_or() với đầu vào là ảnh đã xóa nền và kết quả thu được từ sau khi thực hiện cv2.bitwise_xor() ở bước 3 (-> No_rest).

Thực hiện lệnh OR với hai ảnh sẽ thay thế toàn bộ phần pixel có giá trị bằng 0 của nhau. Phần pixel có giá trị bằng 0 của ảnh sau khi xóa nền sẽ được thay bằng phần pixel tương ứng của No_rest, cũng có thể hiểu rằng phần pixel có giá trị bằng 0 trên No_rest được thay thế bằng pixel tương ứng của ảnh đã xóa nền với, từ đó thu được kết quả cuối cùng: Ảnh thay nền!



Ảnh xóa nền Ảnh thay nền

Hình 18