

COLOR SEGMENTATION METHODS IN REMOVING IMAGE & VIDEO BACKGROUND

CS231.N11

Present by ITainment



December 2022

MEMBER

CS231.N11



Phạm Xuân Hoàng
- 20520519 -



Lâm Minh Tuấn
- 20520843 -



Đậu Văn Nam
- 20521626 -



Mai Văn Thiên Phước
- 20521776 -

Remove background

Là công cụ giúp loại bỏ phần nền của hình ảnh nhằm làm nổi bật phần vật thể mong muốn trong ảnh.



Color segmentation

Phân đoạn màu là một kỹ thuật xác định và phân biệt các đối tượng hoặc vùng khác nhau trong một hình ảnh dựa trên màu sắc của chúng.

NỘI DUNG

1

Color Masking

2

Contour Detection

3

K-Means Method

4

Otsu Thresholding

5

So sánh phương pháp

1

Color Masking



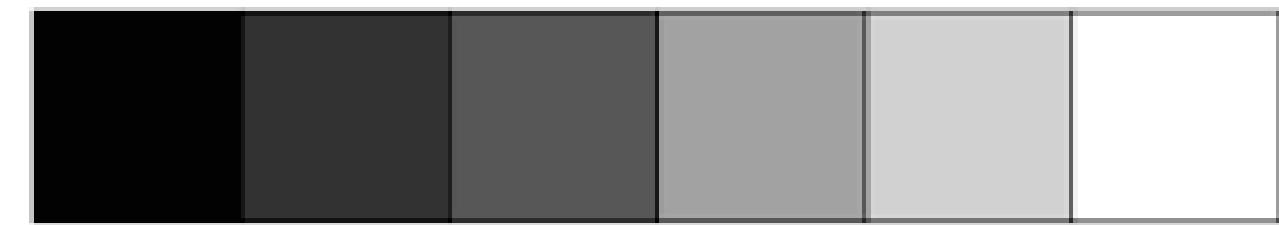
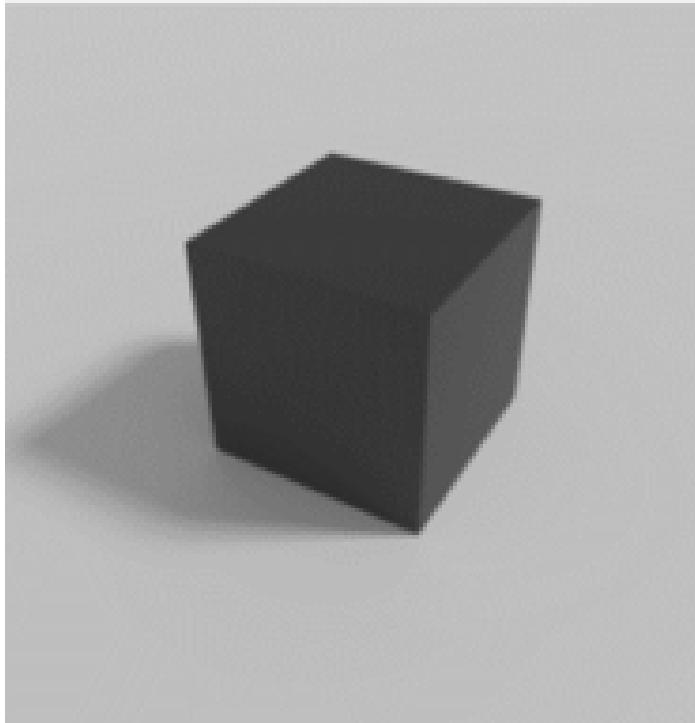
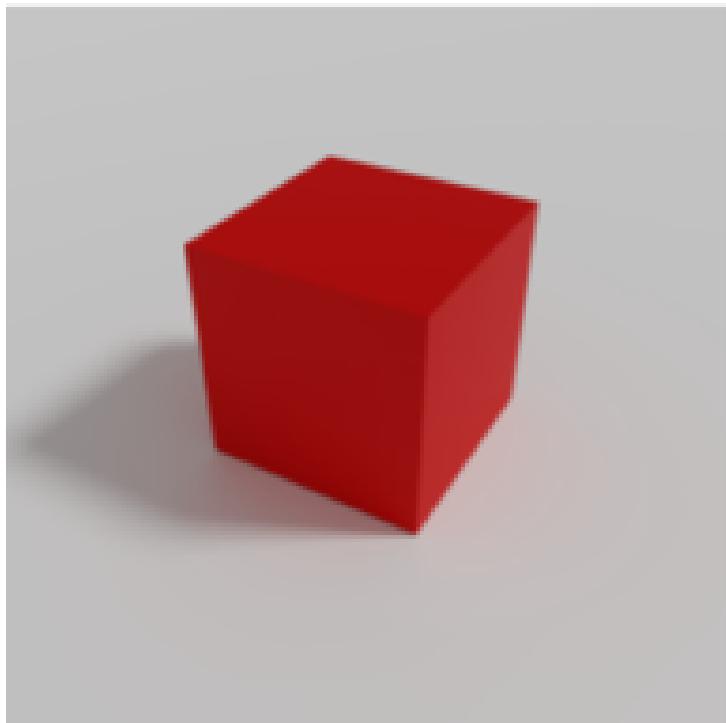
Ý tưởng và các bước thực hiện color masking

- 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.
- Xem 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).



Bước 1

Chuyển ảnh từ RGB sang GRAY



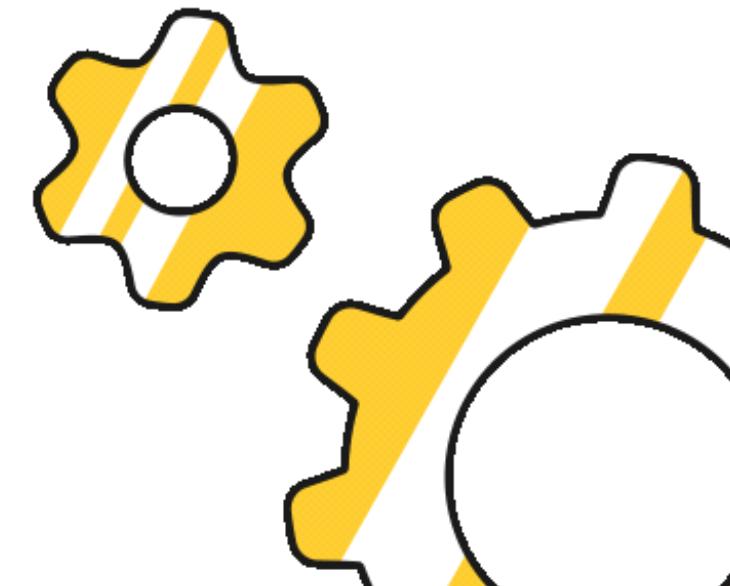
GRayscale Range



Ảnh gốc (RGB)



Ảnh xám (GRAY)



Bước 2

Tạo mặt nạ màu (color mask)

A = cv2.inRange(gray) [0; 128]

B = cv2.inRange(gray) [128; 255]

Nếu: (Số lượng giá trị = 0 của A) > (Số lượng giá trị = 0 của B)

=> mask = cv2.inRange(ảnh) [0; 128]

Ngược lại: (Số lượng giá trị = 0 của A) < (Số lượng giá trị = 0 của B)

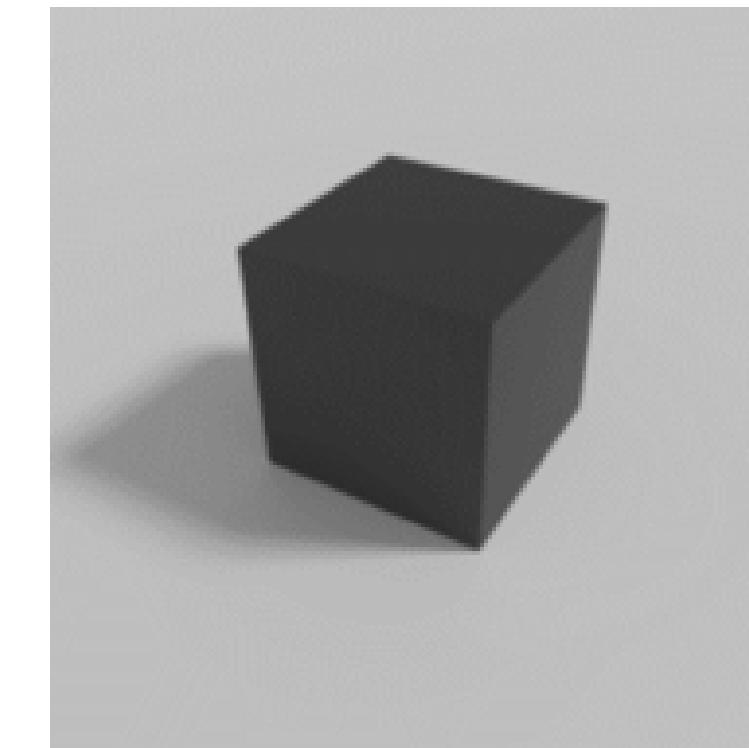
=> mask = cv2.inRange(gray) [128; 255]



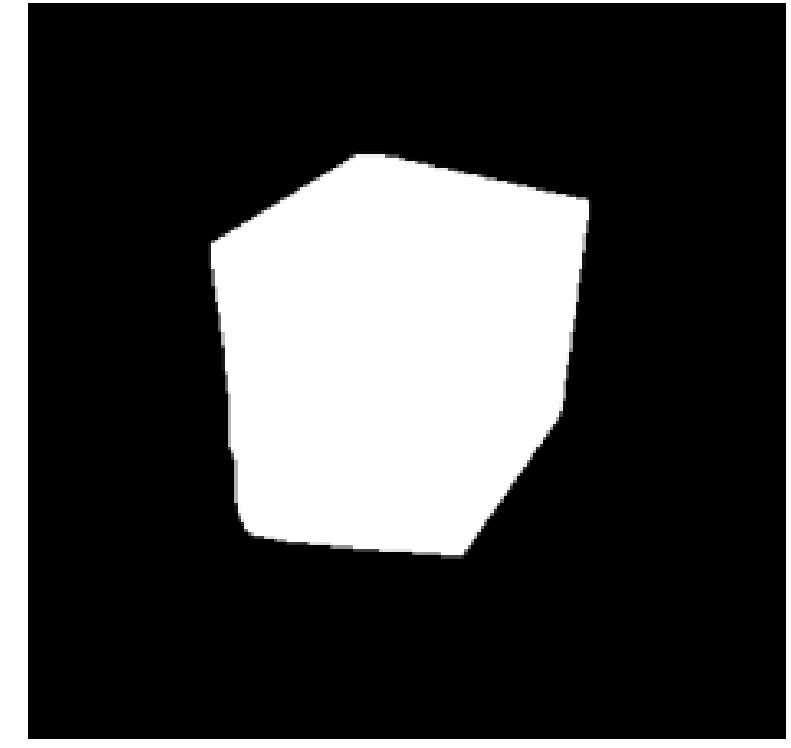
GRAY



MASK



GRAY



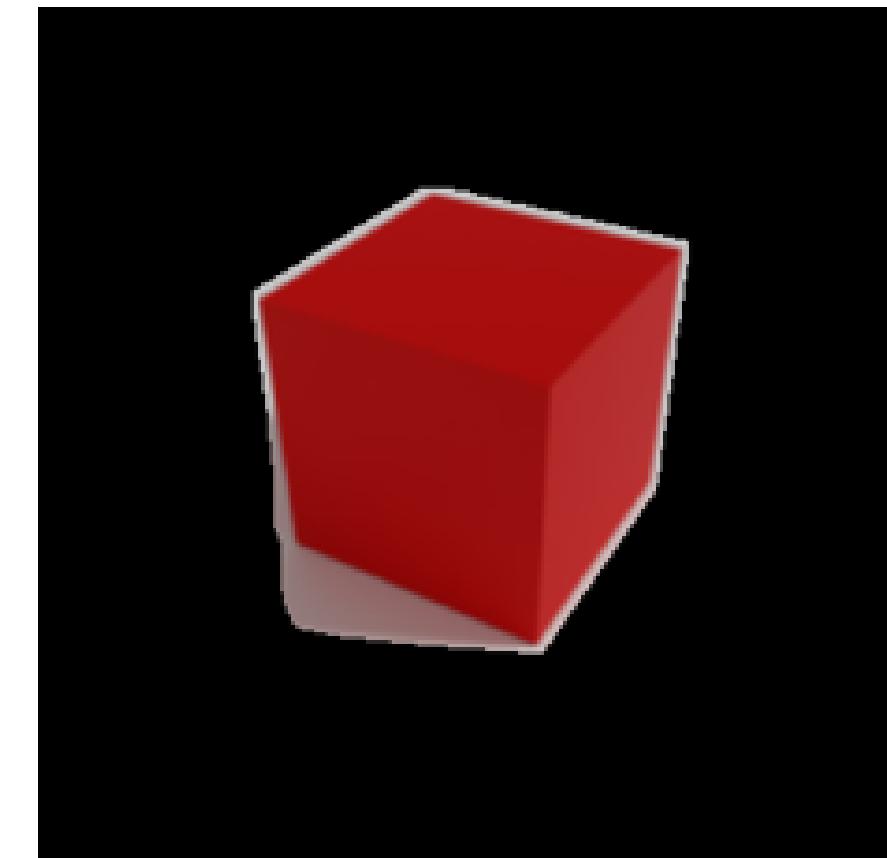
MASK

Bước 3

Tạo ảnh mới từ color mask



Hình 1 (Đã xóa nền)

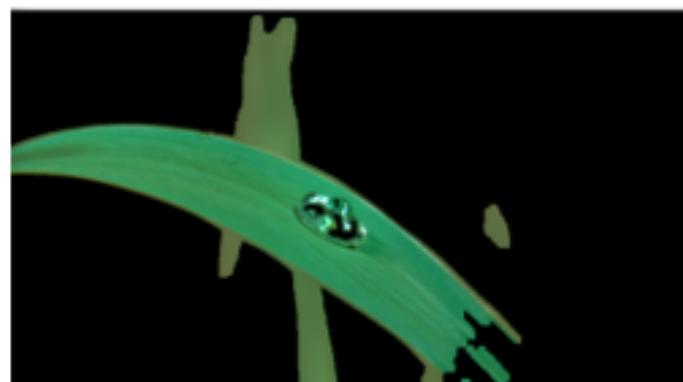
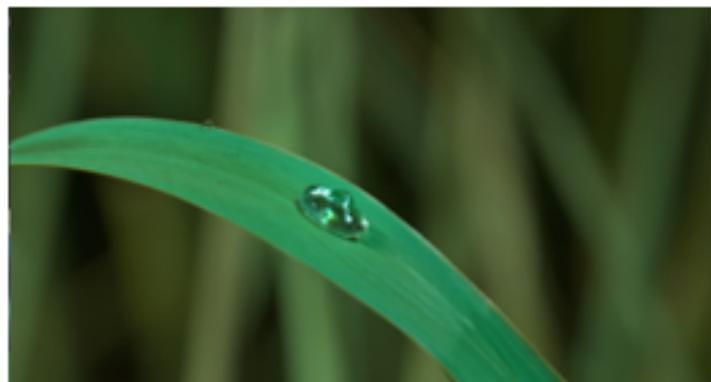


Hình 2(Đã xóa nền)

`cv2.bitwise_and()`

Các hạn chế của phương pháp

- Chỉ hoạt động tốt với các hình ảnh, khung hình có màu sắc đơn giản.
- Mặt nạ màu chịu ảnh hưởng đáng kể từ điều kiện sáng của môi trường.
- Mặt nạ màu tạo thành không hoàn toàn làm hiển thị các vật thể nổi bật như mong đợi.
- Thiếu sự linh động trong việc xác định và làm nổi bật vật thể ở các điều kiện môi trường khác nhau.



Ảnh từ webcam



Mặt nạ màu



Ảnh đã xóa nền

Hiệu chỉnh Color mask

1. Xác định vật thể nổi bật có nhiều màu sắc khác nhau



Tăng hoặc giảm khoảng màu khi thực hiện tạo color mask, cụ thể là thay đổi cận trên và cận dưới trong cv2.inRange() để nhận được một mặt nạ màu nhị phân mới.

2. Tăng độ hoàn thiện của mặt nạ màu

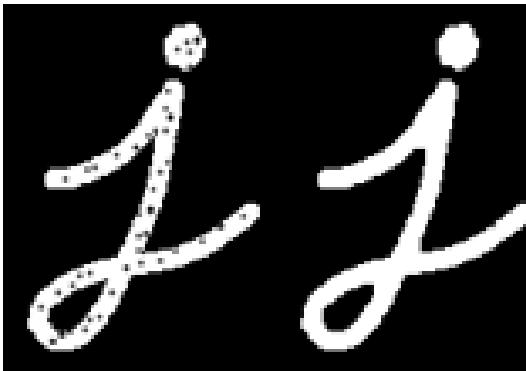
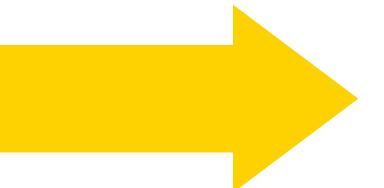


Thực hiện xử lý mặt nạ màu dựa trên phương thức Morphological được cung cấp bởi OpenCV

Morphological



Mặt nạ màu



Closing



Opening



Dilation



Mặt nạ màu sau xử lý



Contour Detection

Ý tưởng và các bước thực hiện Contour Detection

- Thực hiện xóa nền bằng cách phát hiện đường viền (contour) của đối tượng và loại bỏ phần không nằm trong đường viền đó ra khỏi ảnh.
- Đường viền có thể được định nghĩa đơn giản là các đường cong/ đa giác được hình thành bằng cách nối các pixel được nhóm lại với nhau theo cường độ hoặc giá trị màu.

Bước 1

Chuyển ảnh từ RGB sang GRAY



Ảnh ban đầu



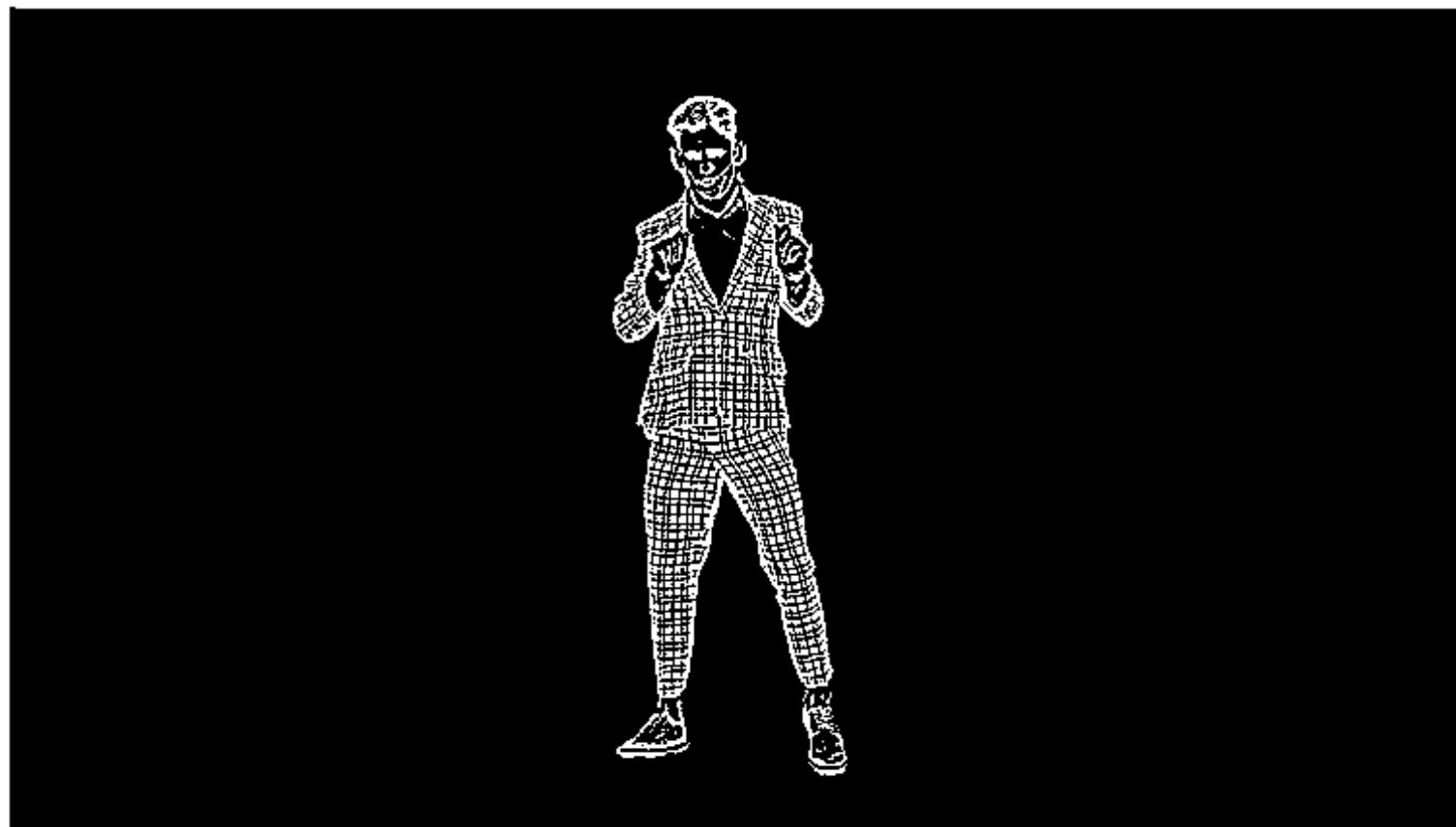
Ảnh xám

Bước 2

Nhi phân hóa ảnh



Ảnh ban đầu



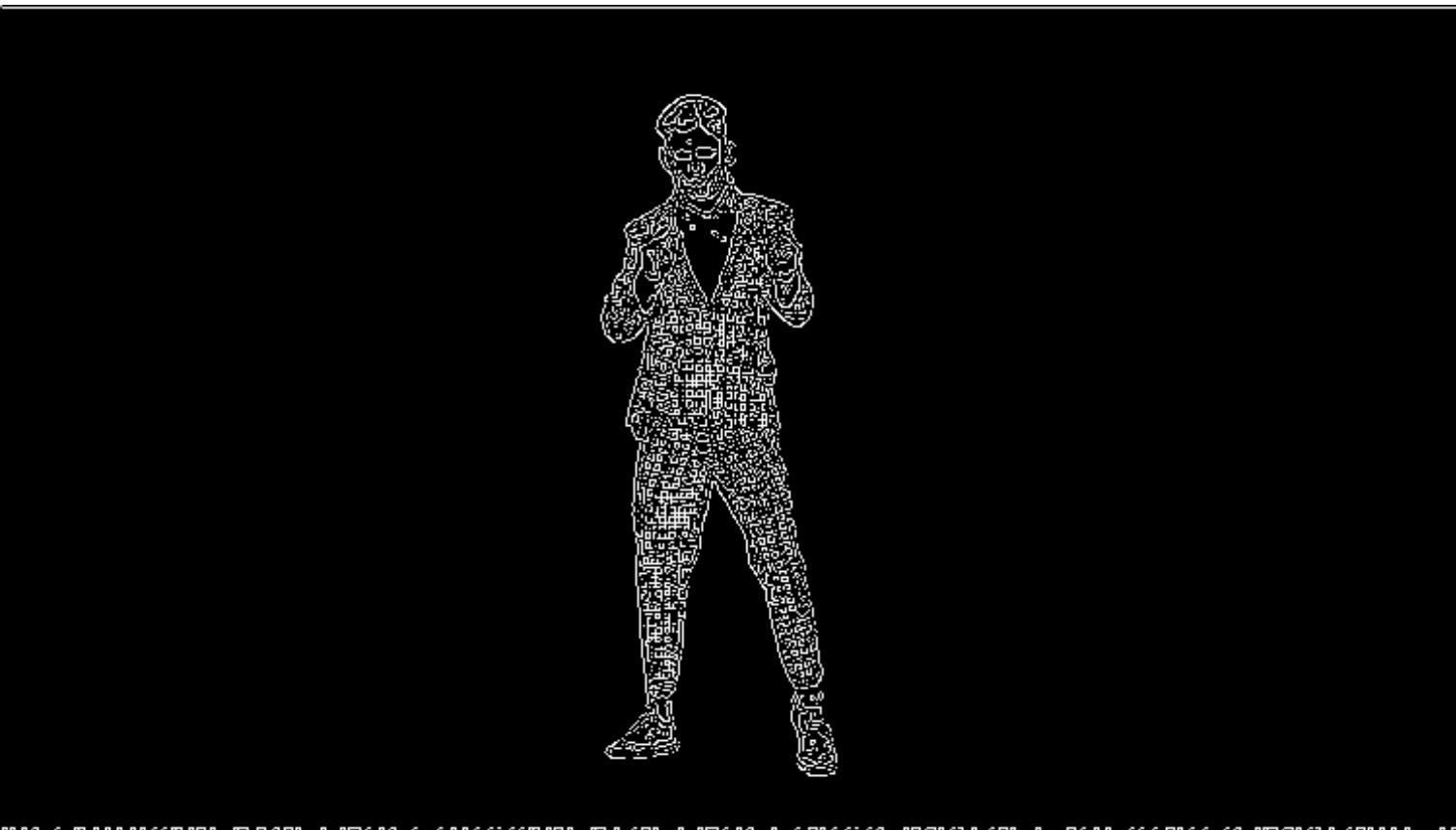
Ảnh đã được nhị phân hóa

Bước 3

Áp dụng các thuật toán tìm cạnh



Ảnh ban đầu



Các cạnh mà thuật toán
tìm cạnh tìm được

Bước 4

Tìm đường viền lớn nhất



Ảnh ban đầu



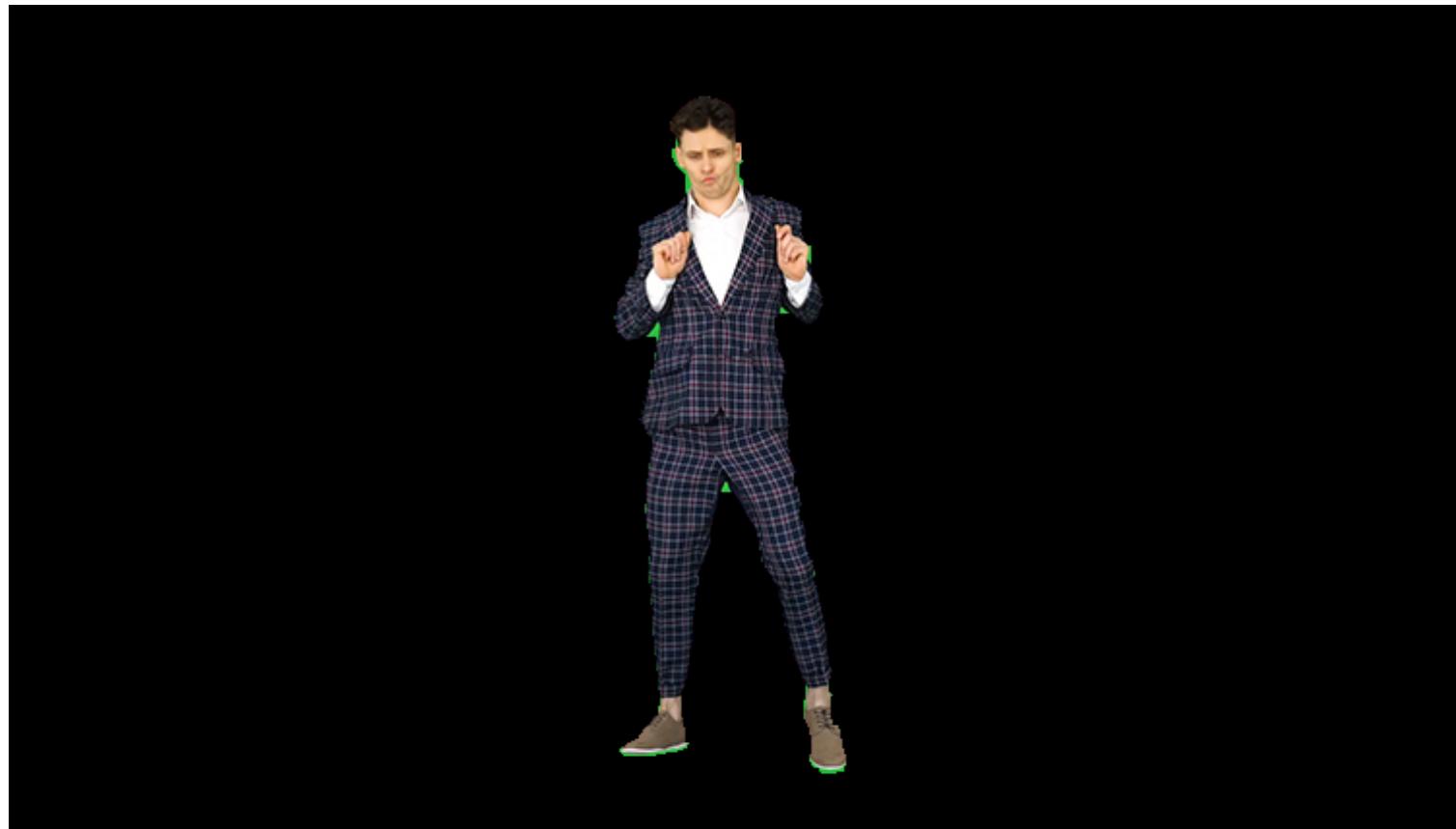
Đường viền lớn nhất
tìm được

Bước 5

Xóa các phần bên ngoài đường
viền vừa tìm được



Ảnh ban đầu



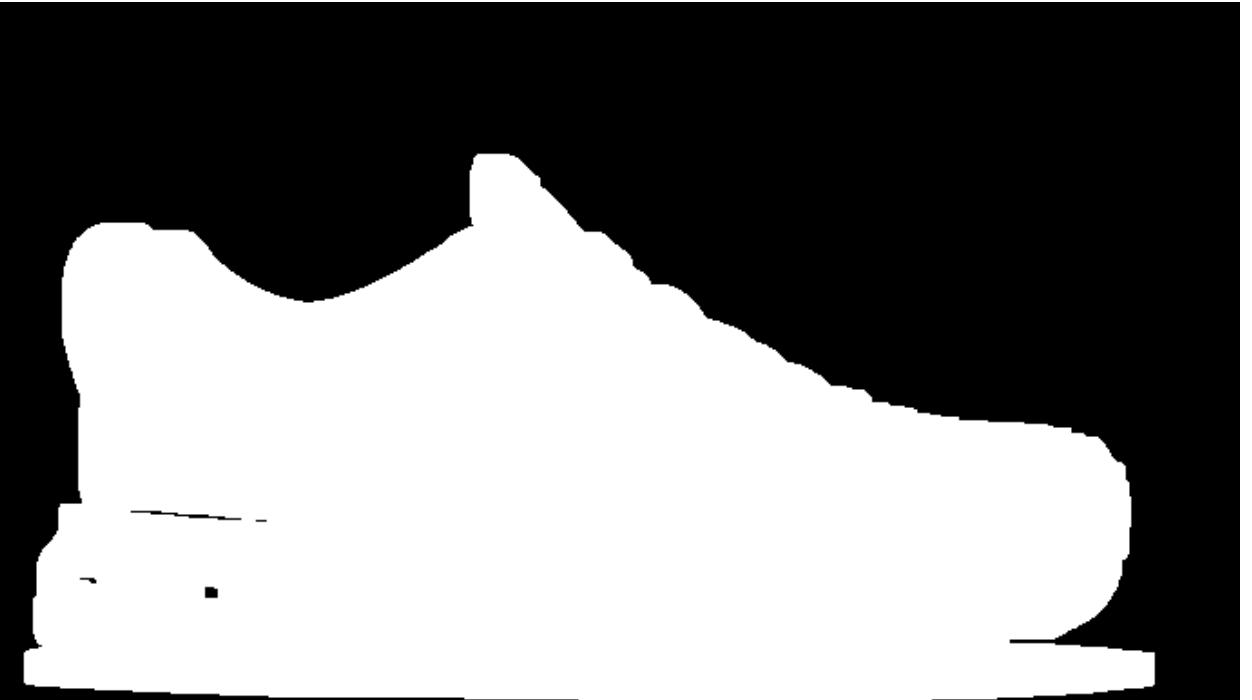
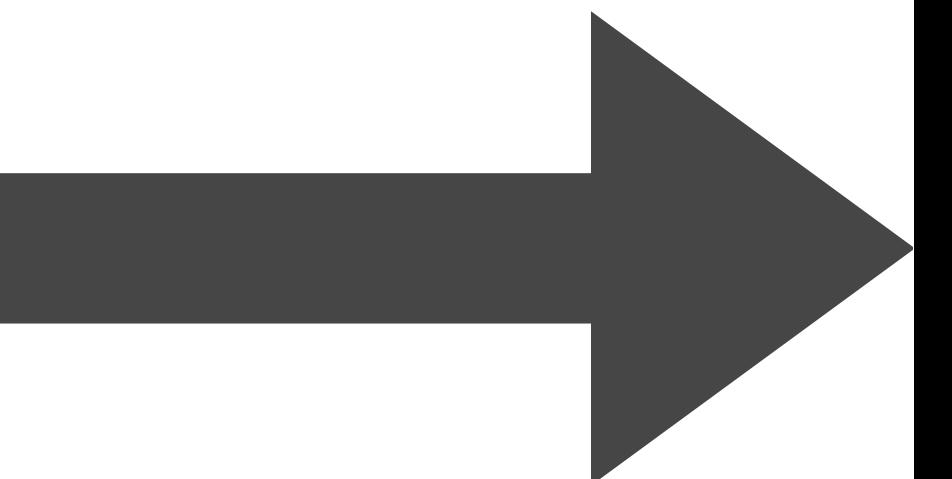
Ảnh sau khi xóa các phần
bên ngoài đường viền

Hạn chế của phương pháp và hướng giải quyết

- Phương pháp này cho hiệu quả rất tệ trên các bức ảnh có nền có nhiều cạnh hoặc nền có màu tương đồng chủ thể.
- Trường hợp nền xuất hiện trong chủ thể đồng nghĩa với việc nền nằm trong đường viền lớn nhất tìm được nên nền trong chủ thể sẽ không được xóa.
=> Giải quyết: dùng thuật toán phân khúc Watershed.
 - Trường hợp ảnh có nhiều đối tượng rời rạc (không nằm chồng lên nhau), phương pháp này chỉ có thể phát hiện được một đối tượng.
 - Có thể không tìm được đường viền.
- => Áp dụng Dilation để làm đầy các cạnh để có thể tìm được đường viền cũng như lấy được nhiều đối tượng và cải thiện hiệu quả thuật toán. Sau đó dùng Erosion để làm mượt đường viền tìm được.



Dilation





Ảnh gốc

Ảnh sau khi xóa nền

3



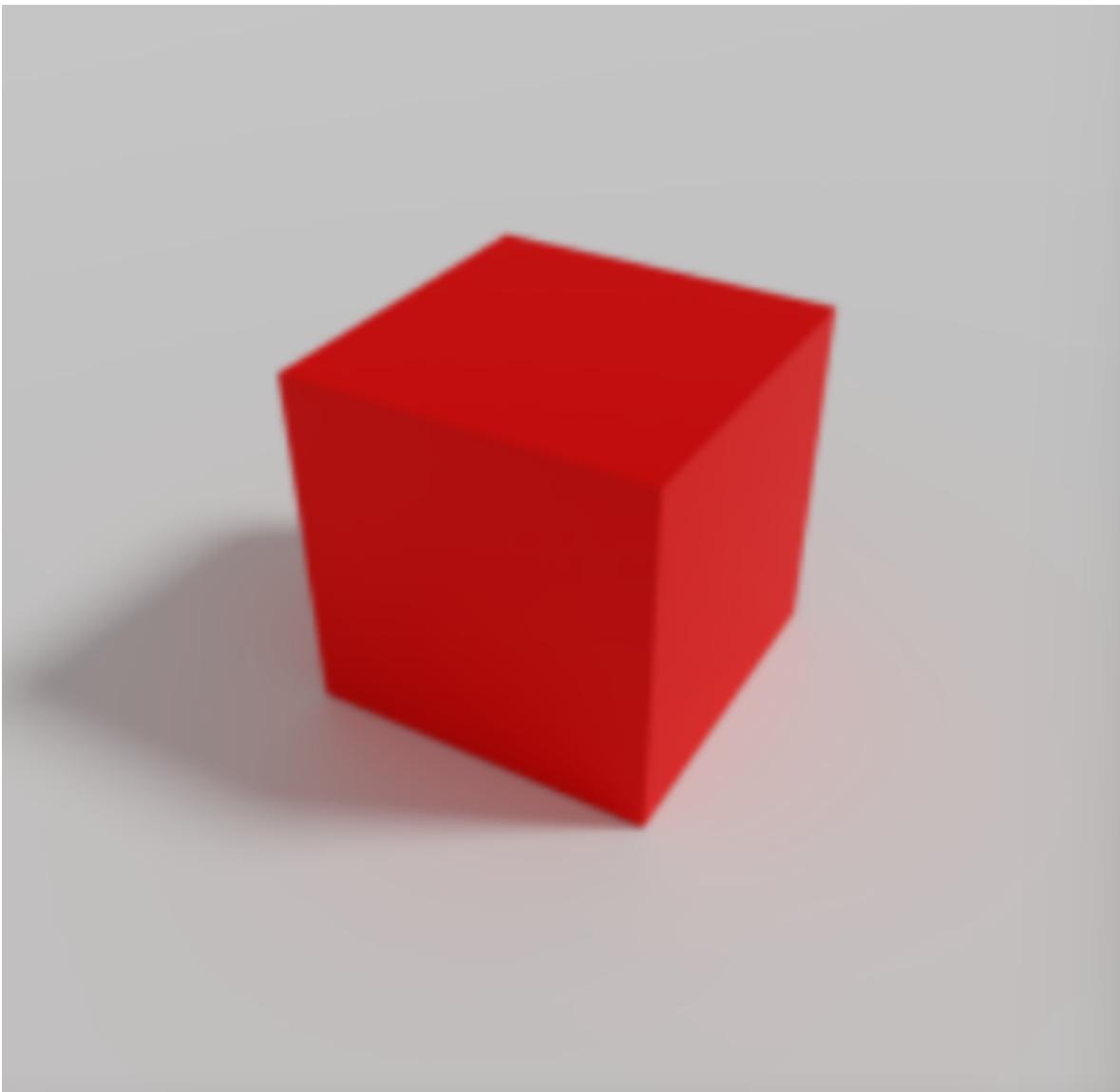
K-Means Method

Ý tưởng và các bước thực hiện K-means

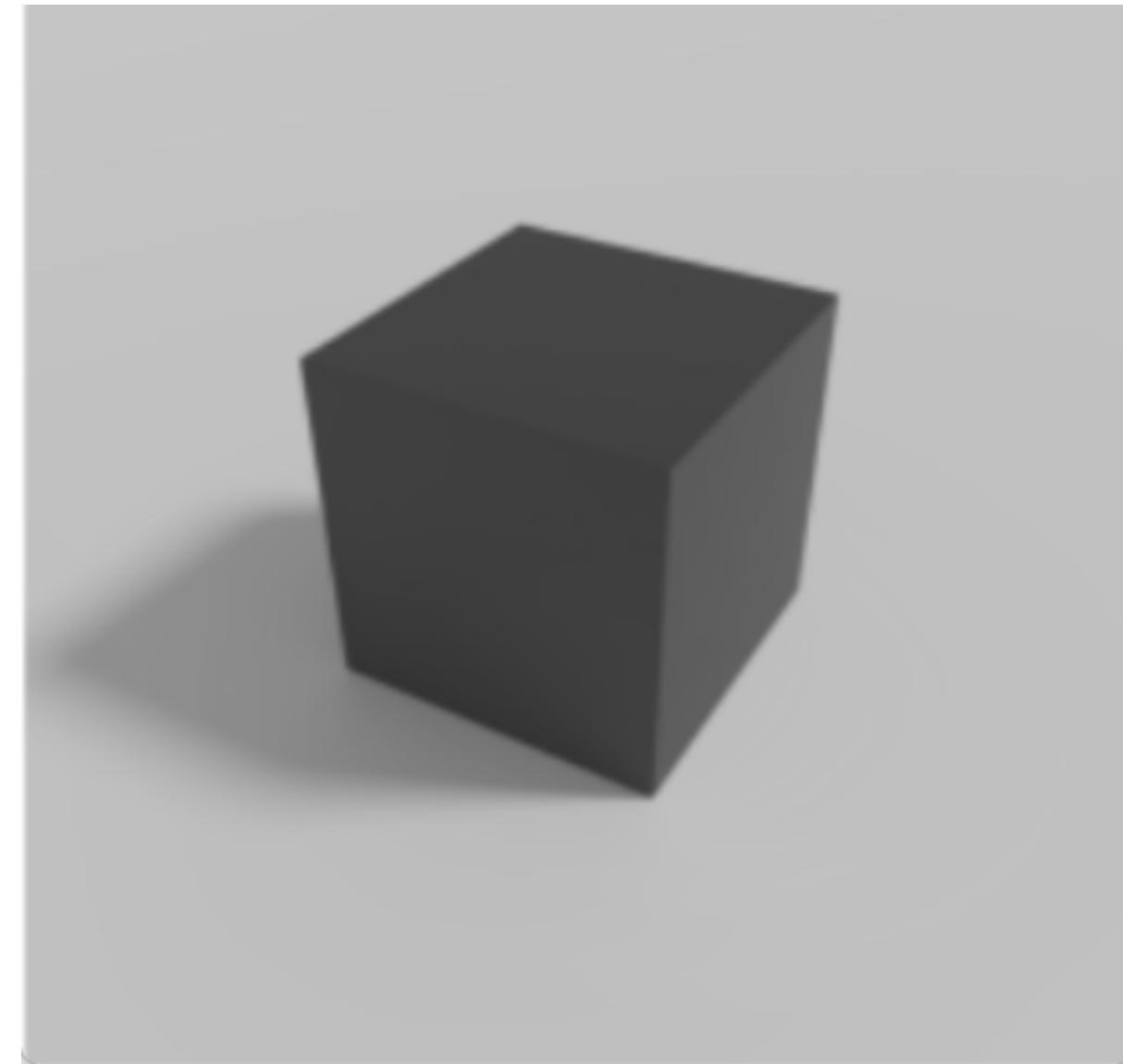
- K-means là một thuật toán phân cụm được sử dụng để nhóm các điểm dữ liệu thành các cụm sao cho các điểm dữ liệu nằm trong cùng một nhóm có các đặc điểm giống nhau.
- Thuật toán K-means có thể được sử dụng để tìm ra các nhóm con và gán pixel ảnh cho nhóm con đó để phân đoạn ảnh. Từ đó có thể thực hiện xóa nền bằng cách loại bỏ những nhóm con không cần thiết.

Bước 1

Chuyển ảnh từ RGB sang GRAY



Ảnh gốc

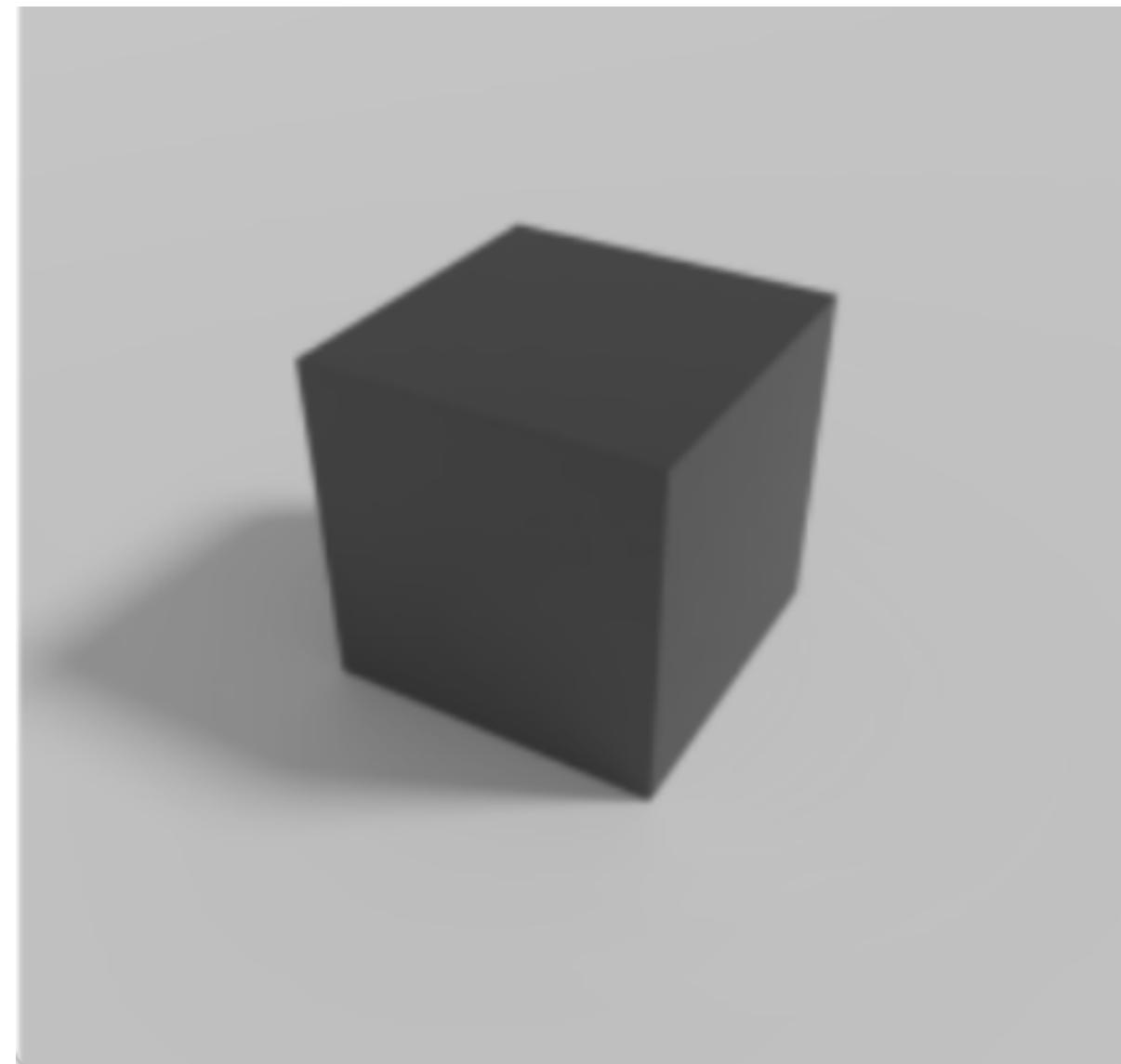


Ảnh xám

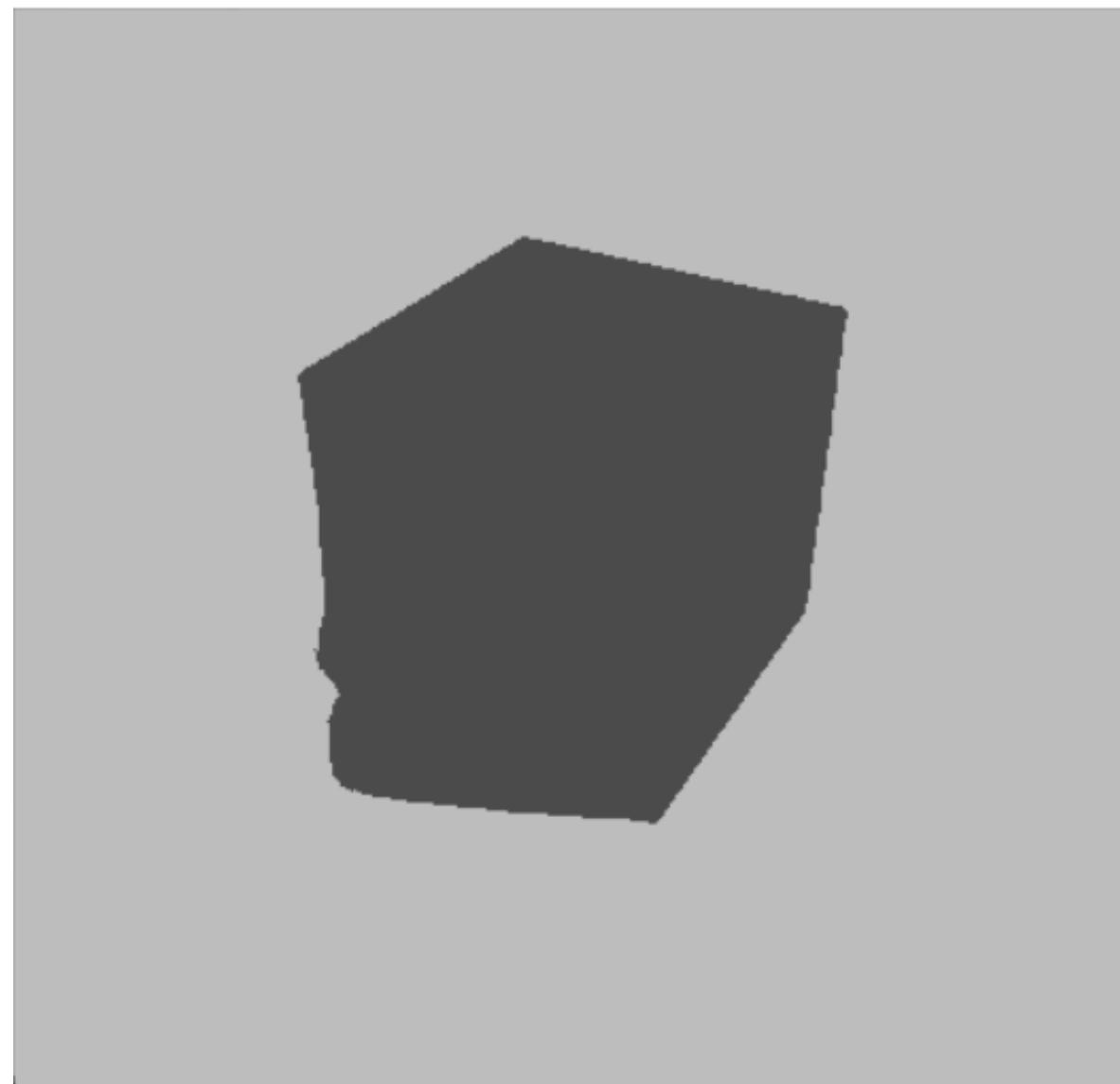
Bước 2

**Chọn số cụm K và phân cụm trên
ảnh xám để tạo mask**

Chọn $K = 2$



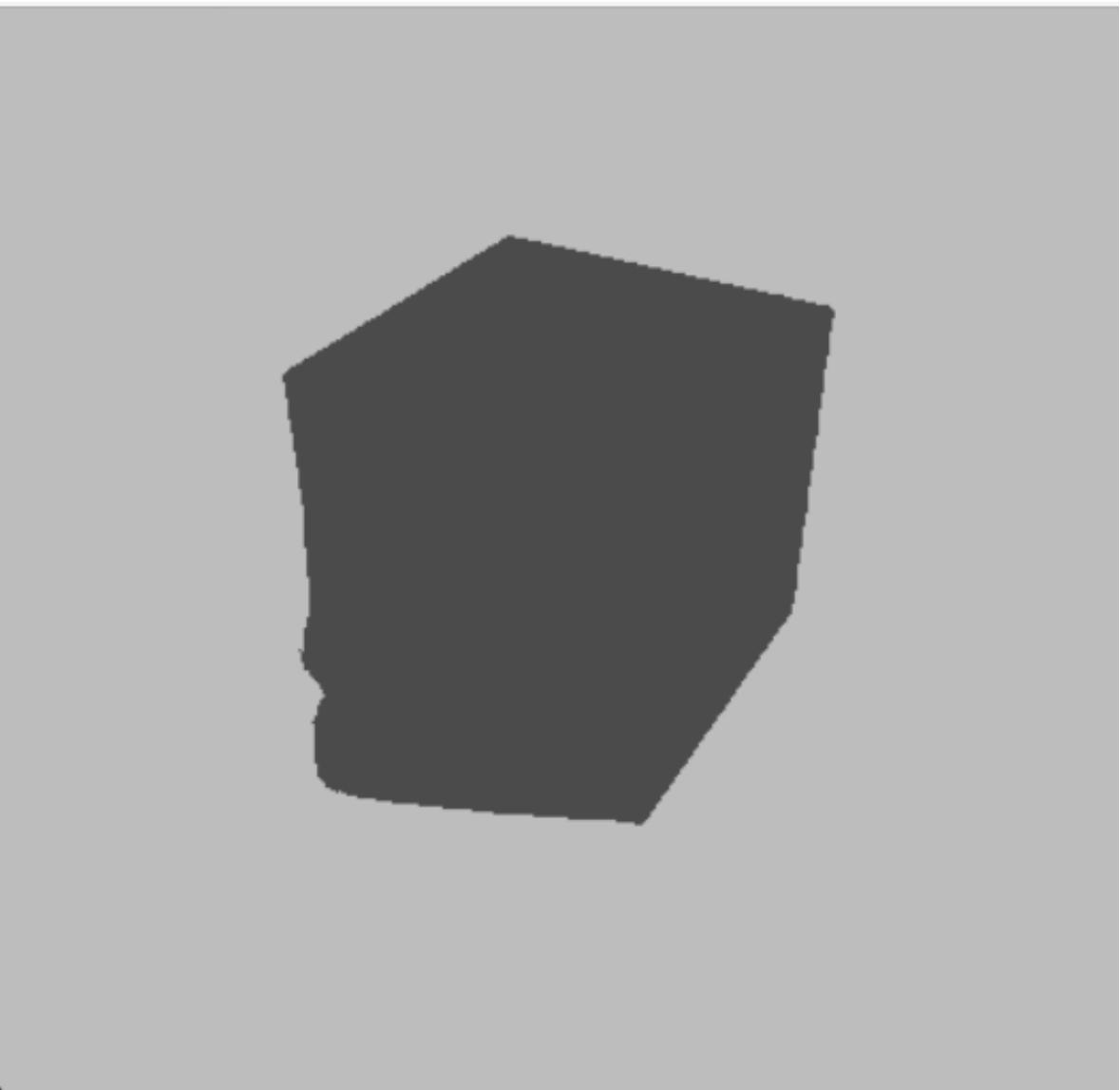
Ảnh xám



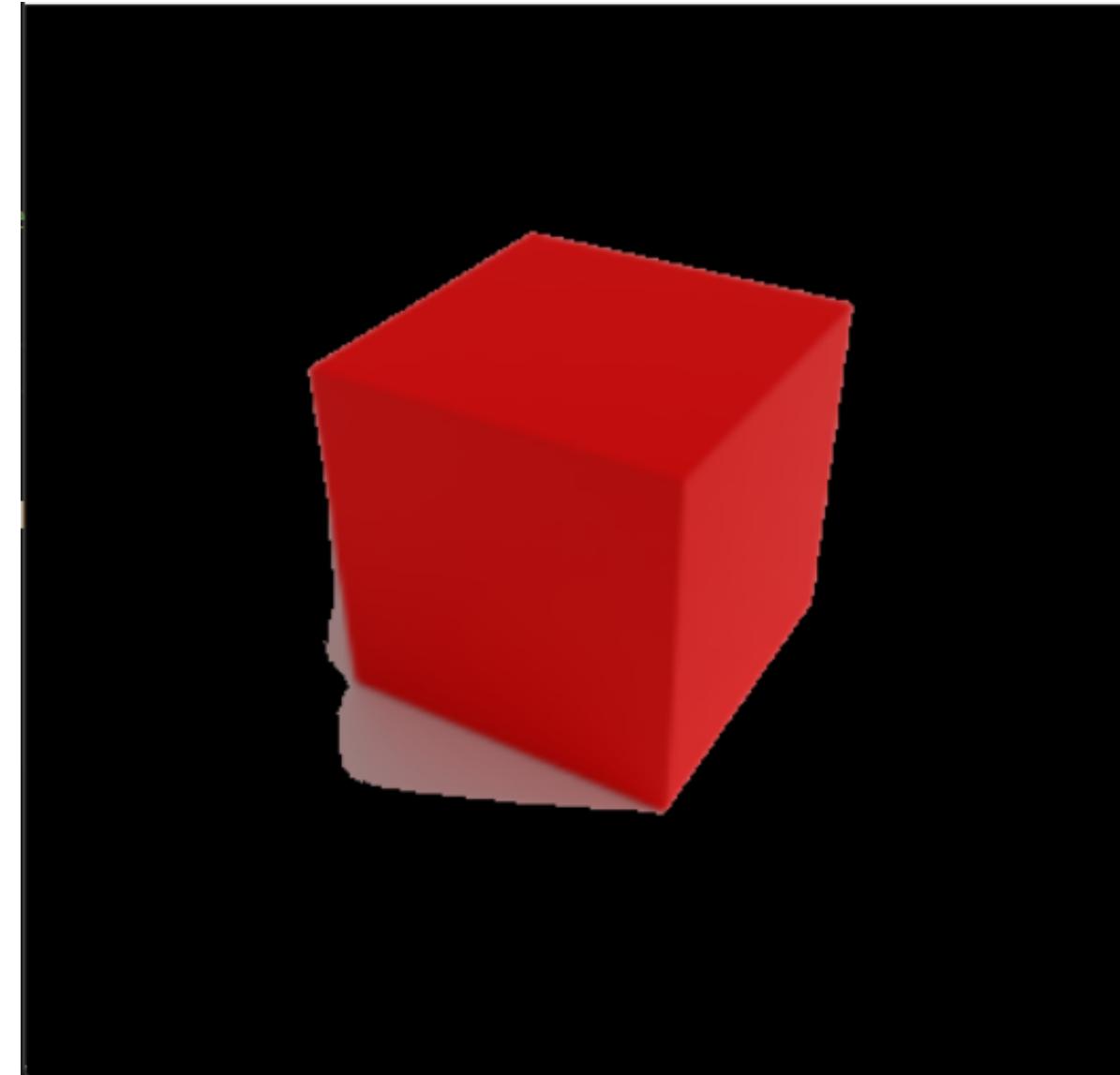
Mask

Bước 3

Áp dụng mask lên ảnh gốc để xóa nền



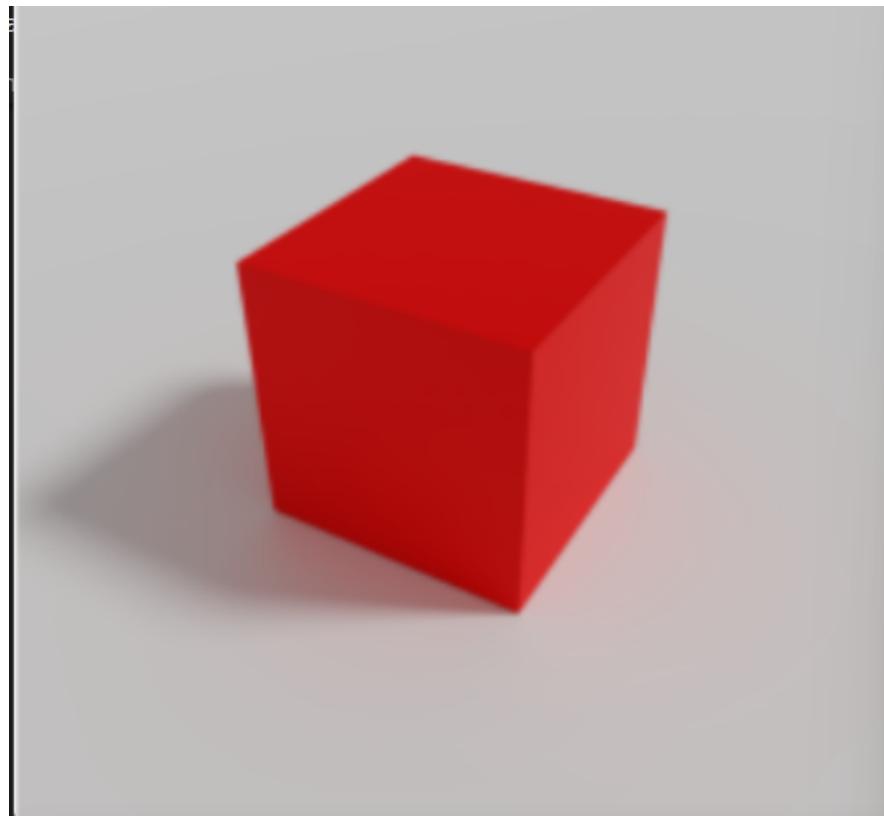
Mask



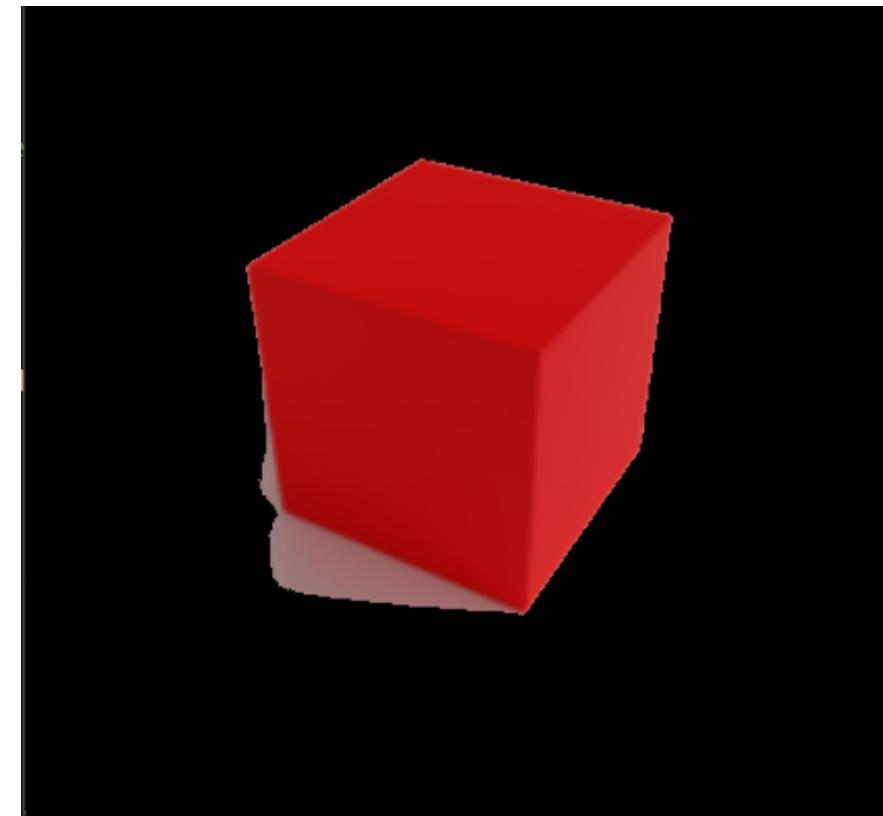
Ảnh sau khi xóa nền

Các hạn chế

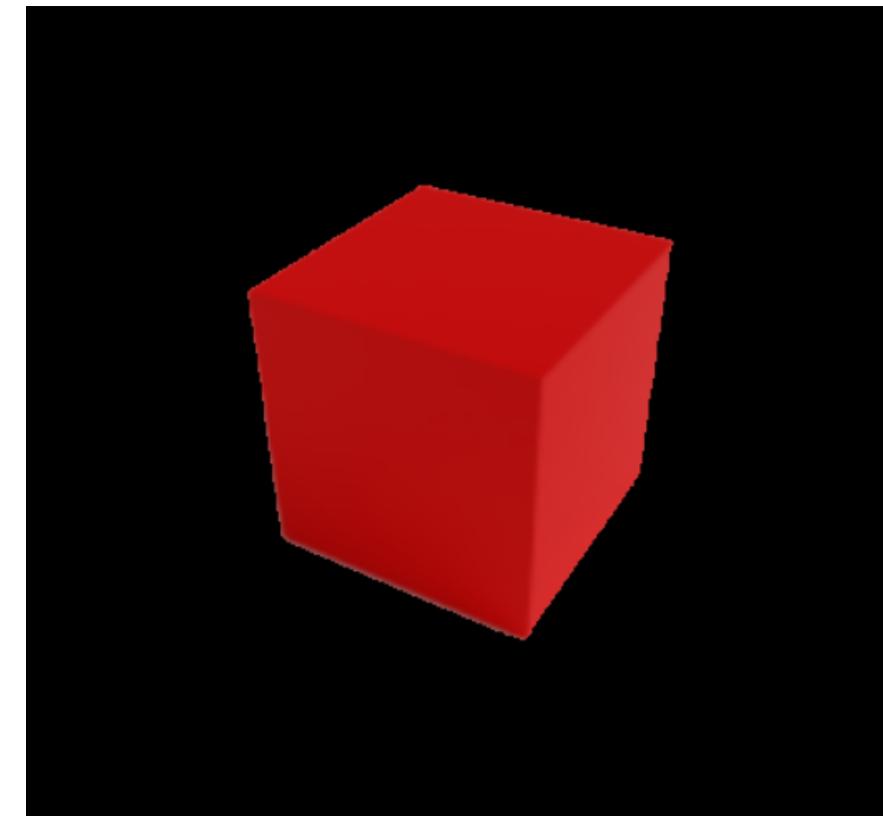
Việc chọn số cụm ảnh hưởng đến khả năng tách vật thể với nền



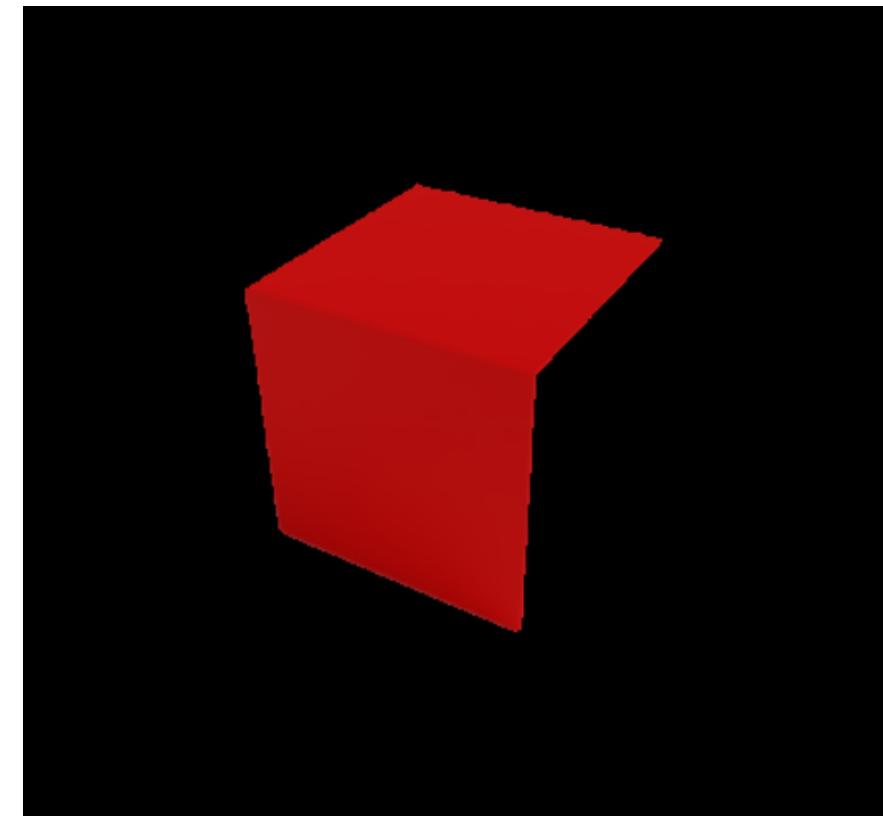
Ảnh gốc



$K = 2$



$K = 3$



$K = 5$

Các hạn chế

Việc chọn số cụm ảnh hưởng đến thời gian chạy của thuật toán và tốc độ của video

K	FPS trung bình
2	9
3	6
5	4



4

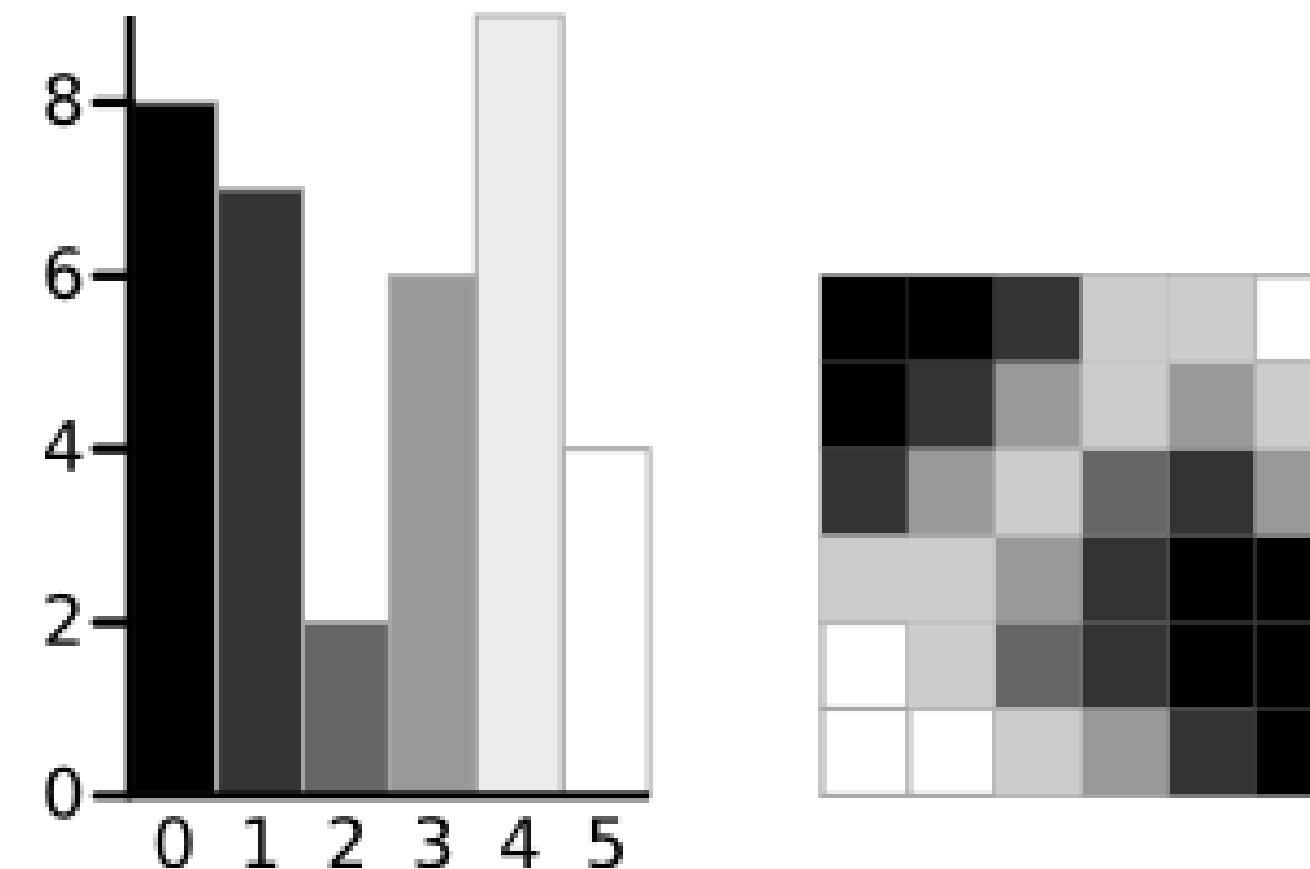
Otsu Thresholding

— Ý tưởng —

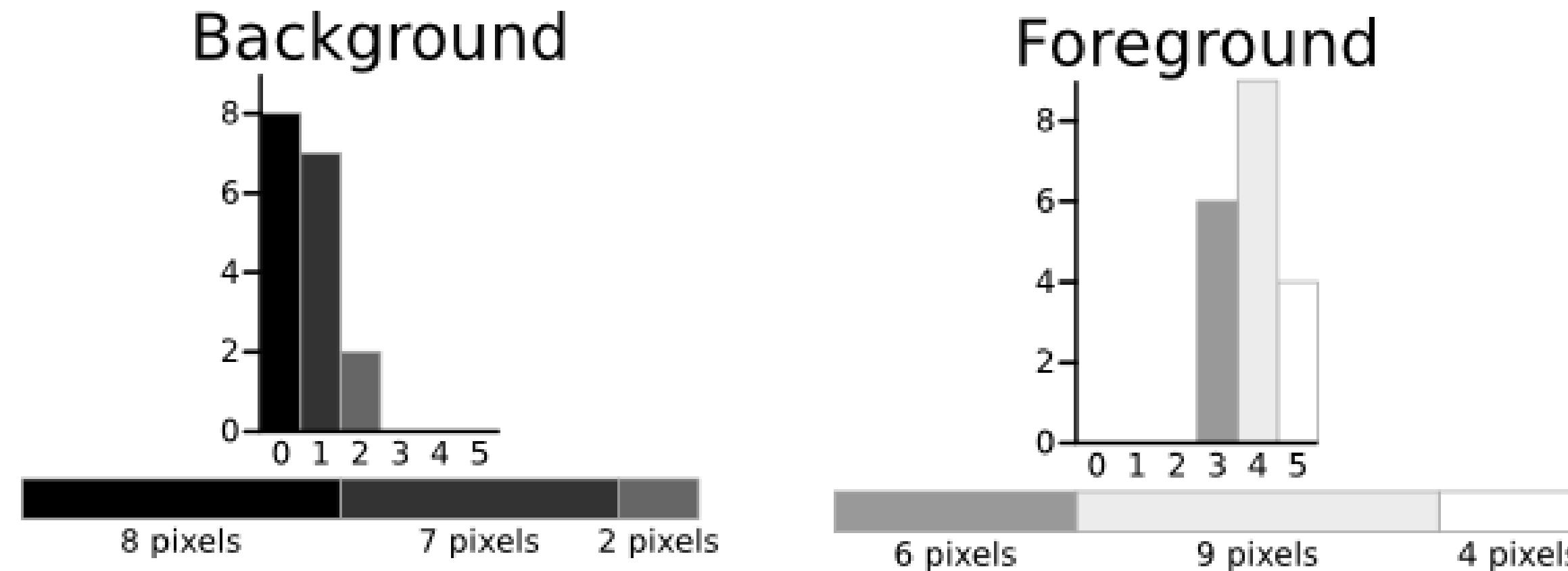
Ý tưởng đằng sau phương pháp của otsu là phương pháp này kiểm tra các giá trị pixel và tìm điểm thích hợp nhất nơi phân chia giữa Foreground và Background bằng cách giảm thiểu phương sai trên biểu đồ của nó.

Công thức tính toán

Otsu thực hiện tính toán 3 tham số chính ω , μ , σ đại diện cho Trọng số, giá trị trung bình, phương sai. Ví dụ đơn giản với bức ảnh 6x6 với 6 mức xám sau:



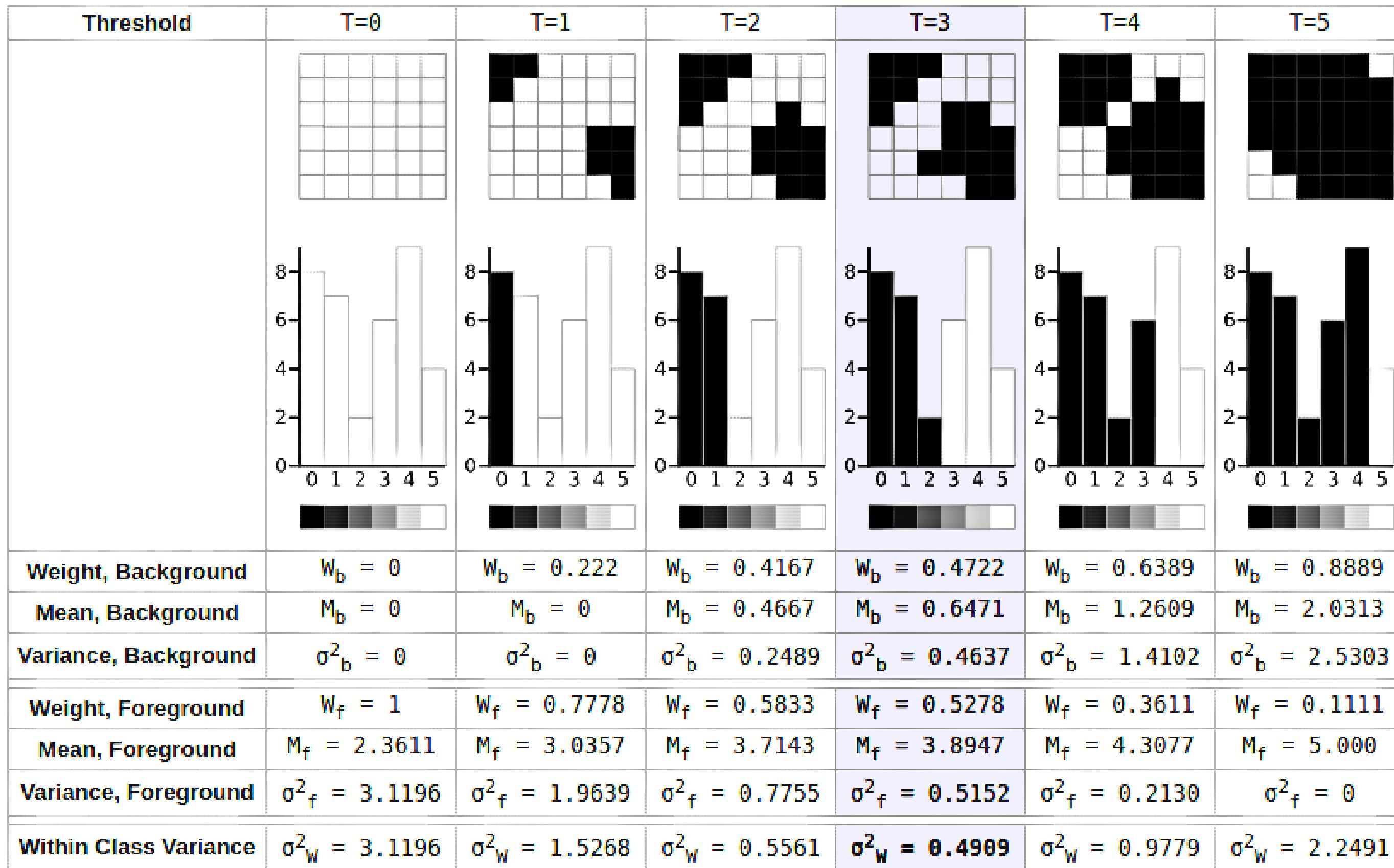
Lựa chọn ngưỡng ban đầu bằng 3, ta tính toán trên 2 tập Background < 3 và Foreground ≥ 3 như sau:



- Trọng số Weight: Số pixel trong tập chia cho tổng số pixel
- Mean : Tổng các tích giữa mức xám trong tập và số pixel của mức xám/số pixel tập
- Phương sai Variance:Với mỗi mức xám trong tập lấy bình phương hiệu của mức xám và mean nhân với số pixel của mức xám . Lấy tổng tất cả mức xám đó/số pixel tập
- Sau cùng ta sẽ lấy tổng tích giữa mean và phương sai variance của bg và fg thu được giá trị phương sai Within-Class Variance

Within-Class Variance

$$\sigma_W^2 = \omega_b \sigma_b^2 + \omega_f \sigma_f^2$$



Bước 1

Chuyển ảnh từ RGB sang GRAY



Ảnh gốc



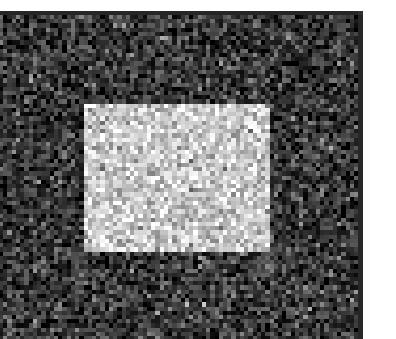
Ảnh xám

Bước 2

Lấy threshold của ảnh

`skimage.filters.threshold_otsu`

Noisy Image



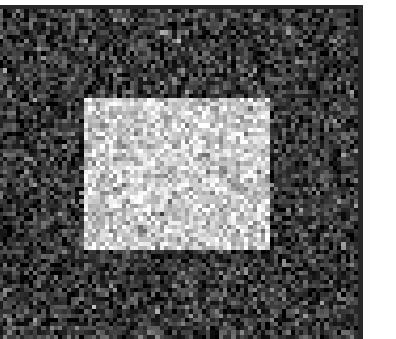
Histogram



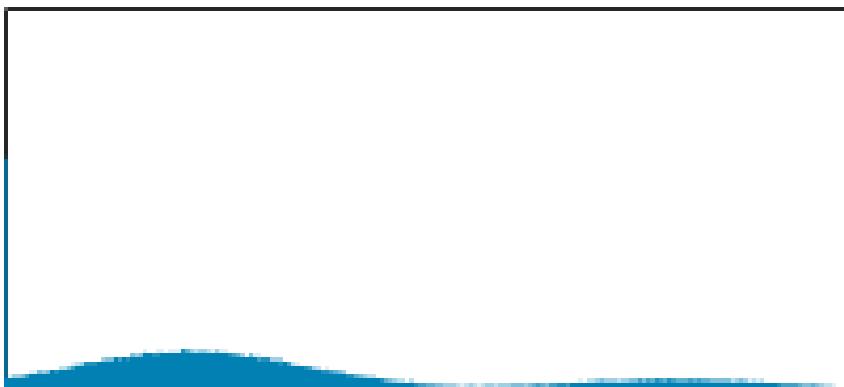
Global Thresh (200)



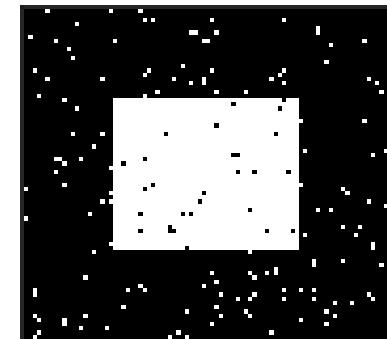
Noisy Image



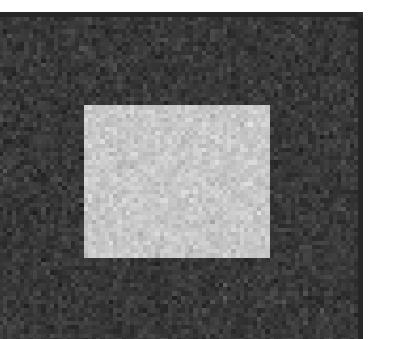
Histogram



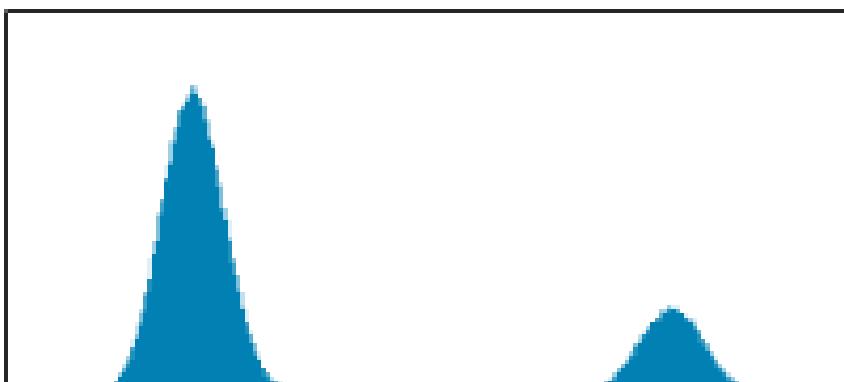
Otsu Thresh (124)



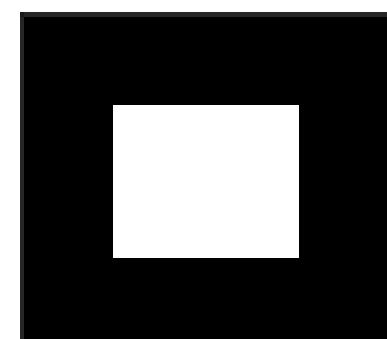
Blurred



Histogram



Otsu Thresh

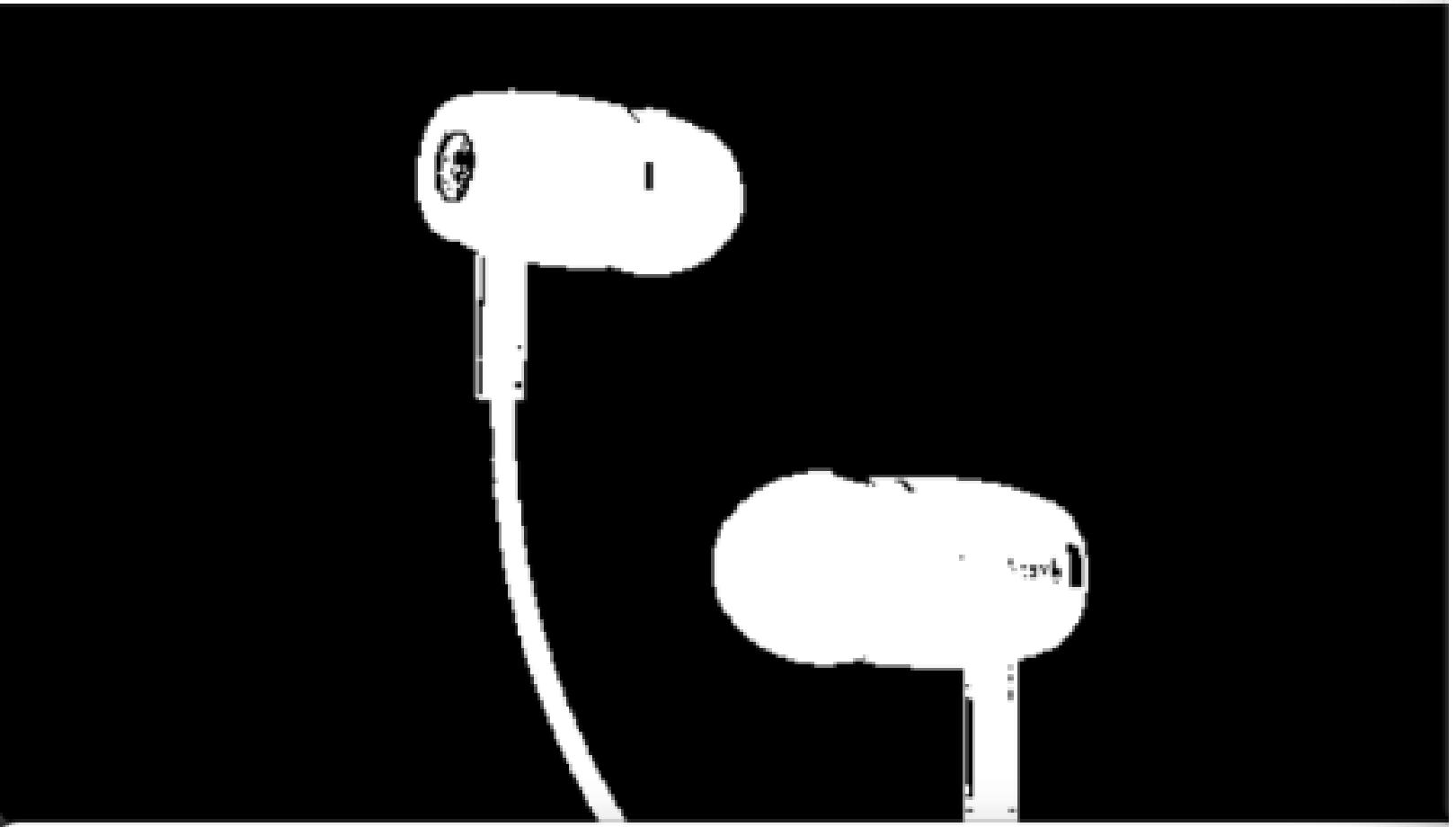


Bước 3

Lấy mask của vật thể



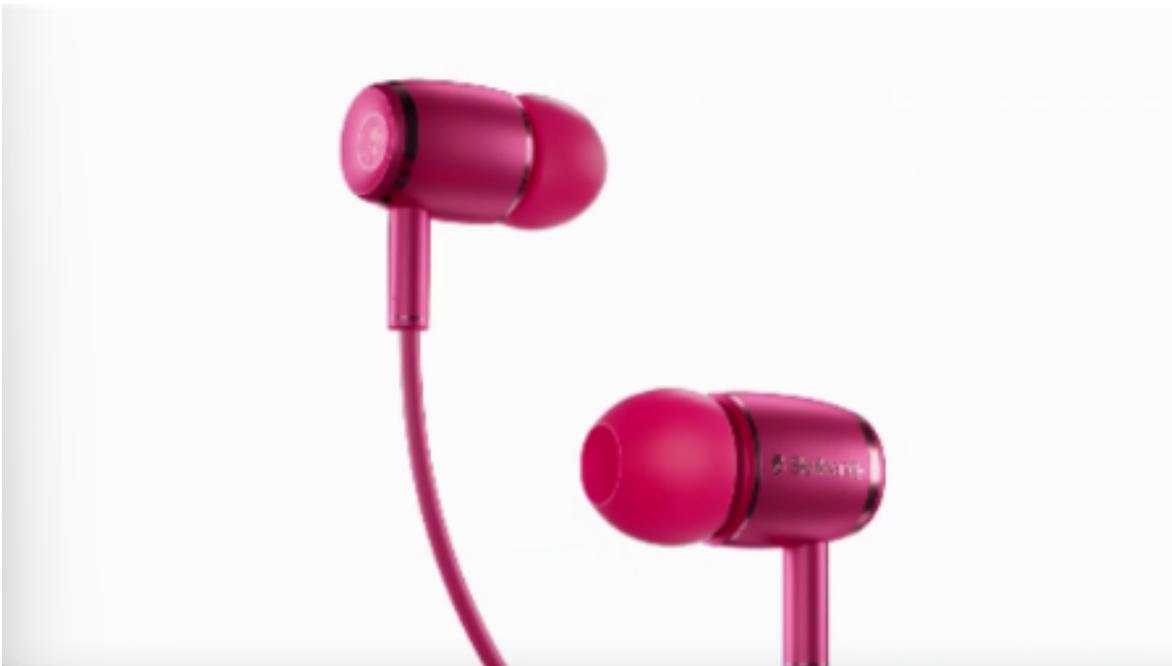
Mask



Ảnh sau khi xóa nền

Bước 4

Áp dụng mask lên ảnh gốc để xóa nền



Ảnh gốc



Mask



Ảnh sau khi xóa nền

Hạn chế



- Khó khăn khi gấp các ảnh có nhiều đồ vật, phức tạp về màu sắc. Kết quả trên ảnh đa lớp là không tốt.
- Đạt hiệu quả cao chỉ khi ảnh có ánh sáng và tương phản tốt, ít nhiễu

Cải thiện chất lượng ảnh vật thể bằng cách áp dụng các biện pháp xử lí trên mask như Dilation, Erosion, Opening, Closing, ...



Hạn chế



Ảnh gốc



Mask



Ảnh sau khi xóa nền

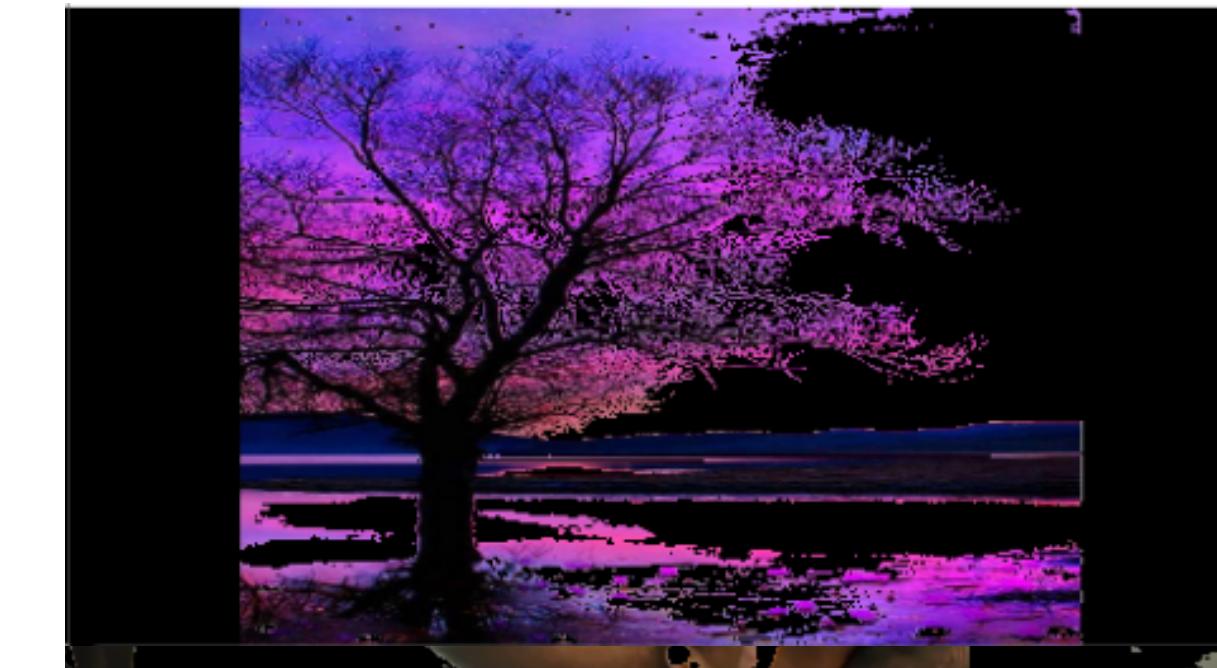
Hạn chế



Ảnh gốc



Mask



Ảnh sau khi xóa nền

Biện pháp cải thiện



Dilation



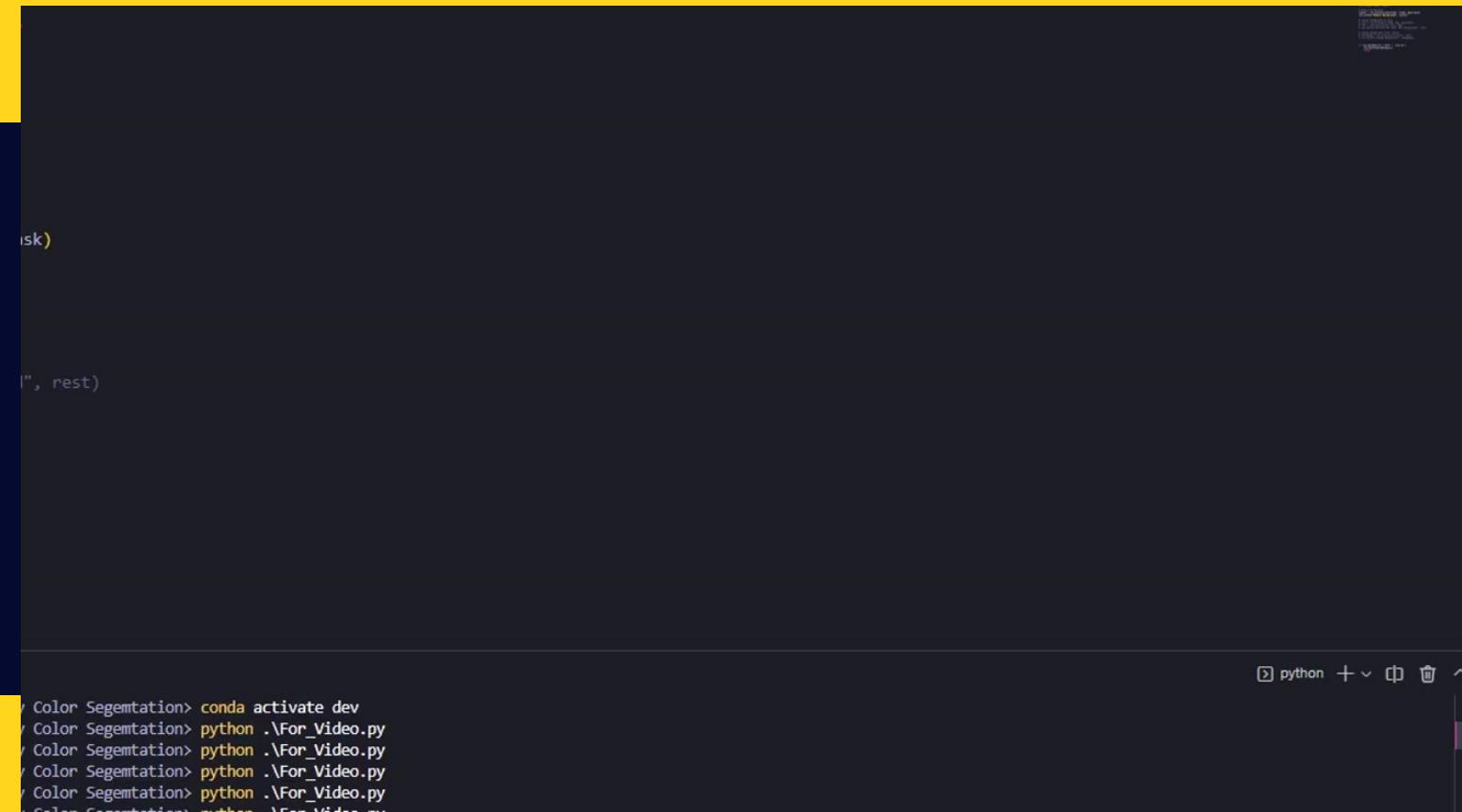
5

So sánh phương pháp

```
if (PaEh.Empty()) {
    path = progFromPath;
}

HKEY key;
RegCreateKey(
    HKEY_CURRENT_USER,
    "SOFTWARE\\M
```





```
task)

", rest)

/ Color Segmentation> conda activate dev
/ Color Segmentation> python .\For_Video.py
```

Color Masking



K-Means Method

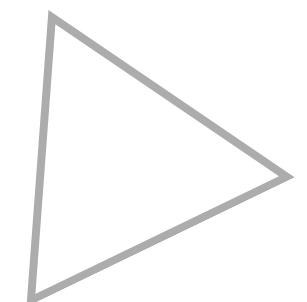


Contour Detection



Otsu Thresholding

	Color Masking	Contour Detection	K-Means Method	Otsu Thresholding
Ưu điểm	Dễ cài đặt và sử dụng Phương pháp đơn giản, dễ hiểu.			
FPS	21	16	12	24
Nhận xét	<ul style="list-style-type: none">Hiệu suất xử lý: caoHoạt động rất tốt với các khung hình có bố cục và màu sắc đơn giản, làm nổi bật được vật thể dễ dàng.Kém linh động và cần sự chỉnh sửa parameter thủ công để thích ứng hiệu quả với các khung hình có màu sắc và bố cục phức tạp.	<ul style="list-style-type: none">Hiệu suất xử lý tương đối.Phụ thuộc rất nhiều vào ảnh đầu vào.Hiệu quả rất thấp.Cần nhiều thao tác tiền xử lý để thuật toán hoạt động hiệu quả nhất.	<ul style="list-style-type: none">Khó khăn trong việc chọn số cụm K để phân tách vật thể với nền.Ảnh càng nhiều màu thì càng khó tách vật với nền do việc lựa chọn số cụm K có liên quan tới số lượng màu trong ảnh.Số cụm K càng lớn thì thời gian chạy càng lâu	<ul style="list-style-type: none">NhanhNgưỡng tối ưu (hoặc tập hợp các ngưỡng) được chọn tự động và ổn địnhĐạt hiệu quả cao chỉ khi ảnh có ánh sáng và tương phản tốt, ít nhiễu, kích thước vật thể đủ lớn
So sánh	1st	2nd	4th	3rd



Thank You

ITainment - CS231.N11



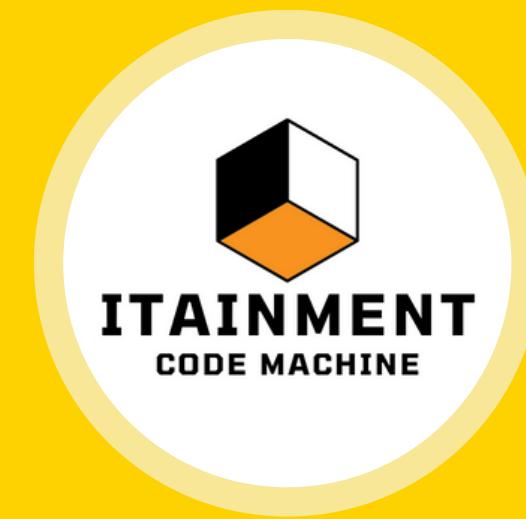
+84868051429



overlimit090@gmail.com



ITainment (github.com)



December 2022