

Remove background - Otsu and K-Means

HUYNH TIEN NAM, University of Science, VNU – HCMC

Remove background

Categories and Subject Descriptors: Computer Vision

Additional Key Words and Phrases: remove background, Otsu, Thresholding, K-Means, ...

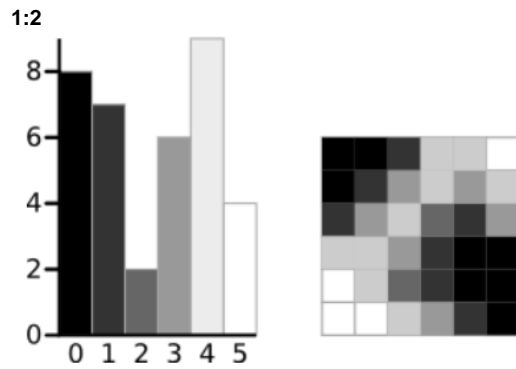
1. INTRODUCTION

- Bài toán xóa background có rất nhiều ứng dụng trong thực tiễn. Ví dụ như việc chỉnh sửa ghép ảnh, chụp ảnh sản phẩm minh họa để đưa lên các sàn thương mại điện tử. Bài toán này có thể xử lý được bằng rất nhiều phương pháp nhưng trong tài liệu này, ta sẽ sử dụng 2 phương pháp sử dụng ngưỡng Otsu và phân lớp K-Means.

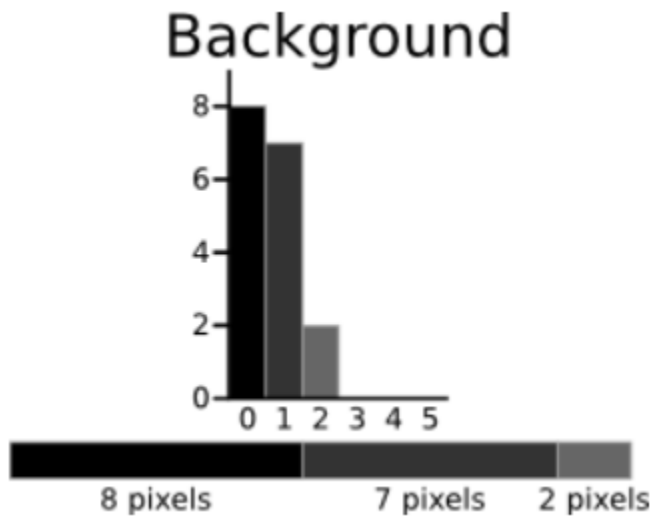
2. BACKGROUND

2.1 Otsu thresholding

- Ngưỡng ảnh được sử dụng để nhị phân hoá dựa trên cường độ pixels. Đầu vào những thuật toán ngưỡng ảnh thường là ảnh mức xám và một ngưỡng. Ảnh đầu ra là một ảnh nhị phân
- Nếu cường độ của một pixel trong ảnh đầu vào lớn hơn giá trị ngưỡng, pixel đầu ra tương ứng được đánh dấu là trắng (foreground) và nếu cường độ của một pixel trong ảnh đầu vào nhỏ hơn hoặc bằng giá trị ngưỡng, pixel đầu ra tương ứng được đánh dấu là đen (background).
- Các thuật toán tính ngưỡng toàn cục tự động thường có các bước sau:
 - i. Xử lý ảnh đầu vào.
 - ii. Lấy biểu đồ histogram của ảnh (sự phân phối của các pixels).
 - iii. Tính toán giá trị ngưỡng T.
 - iv. Thay thế pixel hình ảnh thành màu trắng ở những vùng đó, nơi mà pixels có cường độ lớn hơn T và thành màu đen trong các trường hợp ngược lại.
- Thông thường các thuật toán khác nhau thì khác nhau ở bước 3.
- Hãy tìm hiểu ý tưởng đằng sau cách tiếp cận của otsu. Phương pháp xử lý biểu đồ hình ảnh, phân đoạn đối tượng bằng cách giảm thiểu phương sai trên mỗi lớp. Thông thường, kỹ thuật này tạo ra kết quả thích hợp cho các hình ảnh bimodal. Biểu đồ của hình ảnh này chứa hai điểm hiển thị rõ ràng, đại diện cho các phạm vi khác nhau của các giá trị cường độ.
- Có 2 lựa chọn tìm ngưỡng dựa trên bài báo của Otsu. Cách đầu tiên là cực tiểu hoá các phương sai within-class được xác định bởi $\sigma_w^2(t)$, cách thứ hai là cực đại hoá phương sai between-class được xác định bởi $\sigma_b^2(t)$.
- 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:



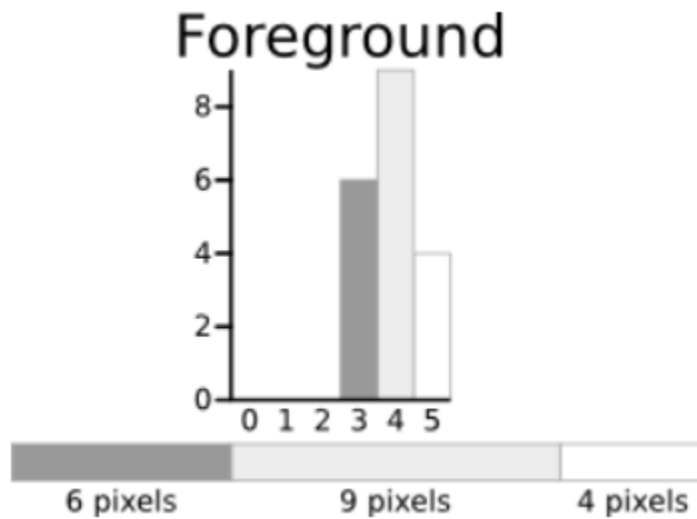
- Lựa chọn ngưỡng ban đầu bằng 3, ta tính toán trên 2 tập Background < 3 và Foreground >= 3 như sau:



Weight $W_b = \frac{8 + 7 + 2}{36} = 0.4722$

Mean $\mu_b = \frac{(0 \times 8) + (1 \times 7) + (2 \times 2)}{17} = 0.6471$

Variance $\sigma_b^2 = \frac{((0 - 0.6471)^2 \times 8) + ((1 - 0.6471)^2 \times 7) + ((2 - 0.6471)^2 \times 2)}{17}$
 $= \frac{(0.4187 \times 8) + (0.1246 \times 7) + (1.8304 \times 2)}{17}$
 $= 0.4637$



$$\text{Weight } W_f = \frac{6 + 9 + 4}{36} = 0.5278$$

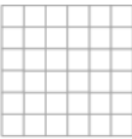
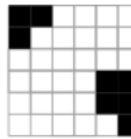
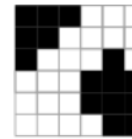
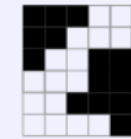
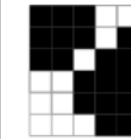

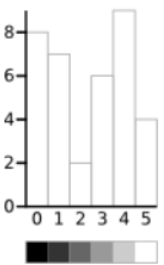
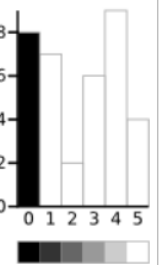
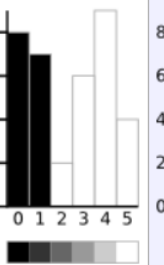
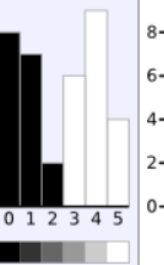
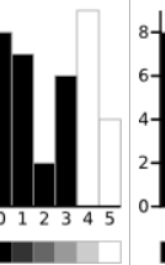
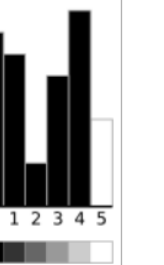
$$\text{Mean } \mu_f = \frac{(3 \times 6) + (4 \times 9) + (5 \times 4)}{19} = 3.8947$$

$$\begin{aligned} \text{Variance } \sigma_f^2 &= \frac{((3 - 3.8947)^2 \times 6) + ((4 - 3.8947)^2 \times 9) + ((5 - 3.8947)^2 \times 4)}{19} \\ &= \frac{(4.8033 \times 6) + (0.0997 \times 9) + (4.8864 \times 4)}{19} \\ &= 0.5152 \end{aligned}$$

$$\begin{aligned} \text{Within Class Variance } \sigma_W^2 &= W_b \sigma_b^2 + W_f \sigma_f^2 = 0.4722 * 0.4637 + 0.5278 * 0.5152 \\ &= 0.4909 \end{aligned}$$

- Giá trị cuối cùng này là " tổng của variances trọng số " cho giá trị ngưỡng 3. Tính toán này cần được thực hiện cho tất cả các giá trị ngưỡng có thể đạt 0 đến 5. Bảng dưới đây cho thấy kết quả cho các tính toán này. Cột được đánh dấu cho thấy các giá trị cho ngưỡng được tính ở trên.

1:4

Threshold	T=0	T=1	T=2	T=3	T=4	T=5
						
						
Weight, Background	$W_b = 0$	$W_b = 0.222$	$W_b = 0.4167$	$W_b = 0.4722$	$W_b = 0.6389$	$W_b = 0.8889$
Mean, Background	$M_b = 0$	$M_b = 0$	$M_b = 0.4667$	$M_b = 0.6471$	$M_b = 1.2609$	$M_b = 2.0313$
Variance, Background	$\sigma_b^2 = 0$	$\sigma_b^2 = 0$	$\sigma_b^2 = 0.2489$	$\sigma_b^2 = 0.4637$	$\sigma_b^2 = 1.4102$	$\sigma_b^2 = 2.5303$
Weight, Foreground	$W_f = 1$	$W_f = 0.7778$	$W_f = 0.5833$	$W_f = 0.5278$	$W_f = 0.3611$	$W_f = 0.1111$
Mean, Foreground	$M_f = 2.3611$	$M_f = 3.0357$	$M_f = 3.7143$	$M_f = 3.8947$	$M_f = 4.3077$	$M_f = 5.0000$
Variance, Foreground	$\sigma_f^2 = 3.1196$	$\sigma_f^2 = 1.9639$	$\sigma_f^2 = 0.7755$	$\sigma_f^2 = 0.5152$	$\sigma_f^2 = 0.2130$	$\sigma_f^2 = 0$
Within Class Variance	$\sigma_W^2 = 3.1196$	$\sigma_W^2 = 1.5268$	$\sigma_W^2 = 0.5561$	$\sigma_W^2 = 0.4909$	$\sigma_W^2 = 0.9779$	$\sigma_W^2 = 2.2491$

Within Class Variance $\sigma_W^2 = W_b \sigma_b^2 + W_f \sigma_f^2$ (as seen above)

$$\begin{aligned}
 \text{Between Class Variance } \sigma_B^2 &= \sigma^2 - \sigma_W^2 \\
 &= W_b (\mu_b - \mu)^2 + W_f (\mu_f - \mu)^2 \quad (\text{where } \mu = W_b \mu_b + W_f \mu_f) \\
 &= W_b W_f (\mu_b - \mu_f)^2
 \end{aligned}$$

Threshold	T=0	T=1	T=2	T=3	T=4	T=5
Within Class Variance	$\sigma_W^2 = 3.1196$	$\sigma_W^2 = 1.5268$	$\sigma_W^2 = 0.5561$	$\sigma_W^2 = 0.4909$	$\sigma_W^2 = 0.9779$	$\sigma_W^2 = 2.2491$
Between Class Variance	$\sigma_B^2 = 0$	$\sigma_B^2 = 1.5928$	$\sigma_B^2 = 2.5635$	$\sigma_B^2 = 2.6287$	$\sigma_B^2 = 2.1417$	$\sigma_B^2 = 0.8705$

2.2 K-Means:

- Trong thuật toán K-means clustering, chúng ta không biết nhãn (label) của từng điểm dữ liệu. Mục đích là làm thế nào để phân dữ liệu thành các cụm (cluster) khác nhau sao cho dữ liệu trong cùng một cụm có tính chất giống nhau.

- Tóm tắt thuật toán:

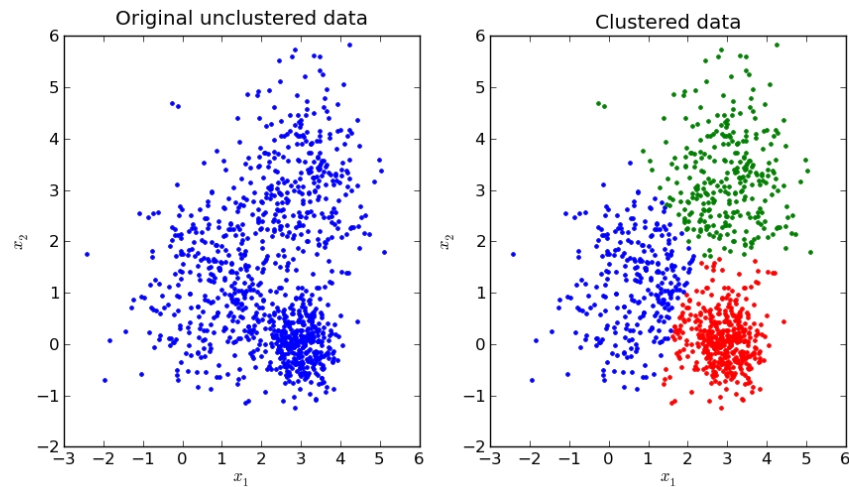
Đầu vào: Dữ liệu X và số lượng cluster cần tìm K.

Đầu ra: Các center M và label vector cho từng điểm dữ liệu Y.

1. Chọn K điểm bất kỳ làm các center ban đầu.
2. Phân mỗi điểm dữ liệu vào cluster có center gần nó nhất.
3. Nếu việc gán dữ liệu vào từng cluster ở bước 2 không thay đổi so với vòng lặp trước nó thì ta dừng thuật toán.
4. Cập nhật center cho từng cluster bằng cách lấy trung bình cộng của tất cả các điểm dữ liệu đã được gán vào cluster đó sau bước 2.

1:5

5. Quay lại bước 2.



3. METHODOLOGY

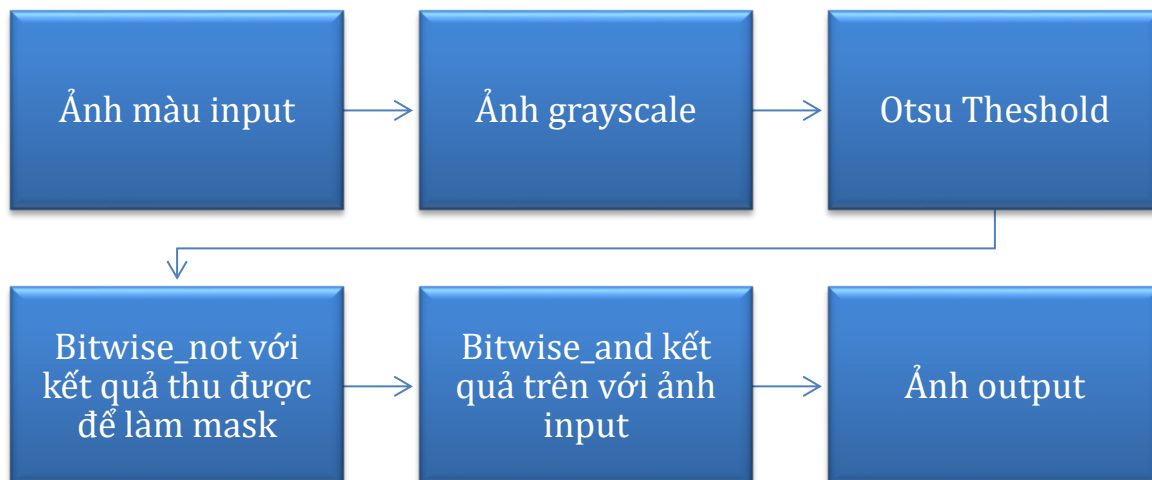
Input: Ảnh màu tùy ý.

Output mong muốn: Ảnh màu phần foreground, phần background bị xoá.

Otsu Thresholding:
- Với kết quả mặc định của thresholding, pixel có cường độ lớn hơn ngưỡng được xem là foreground còn ngược lại là background.



- Tuy nhiên các bước xử lý trên chỉ cho kết quả đúng khi foreground có màu sáng hơn background.
- Vậy với trường hợp pixel ở foreground có cường độ nhỏ hơn cường độ pixel ở background thì sao ?



1:6

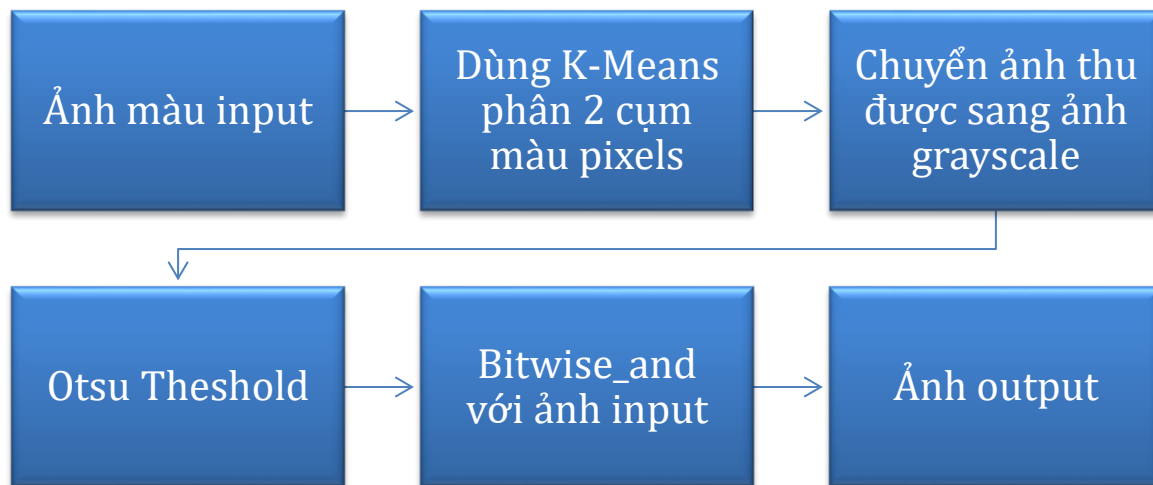
- Bitwise_not sau khi cho chạy thuật toán otsu thresholding để đảo kết quả nhận được sau khi otsu thresholding để nhận được kết quả đúng.

3.2 K-Means + Otsu

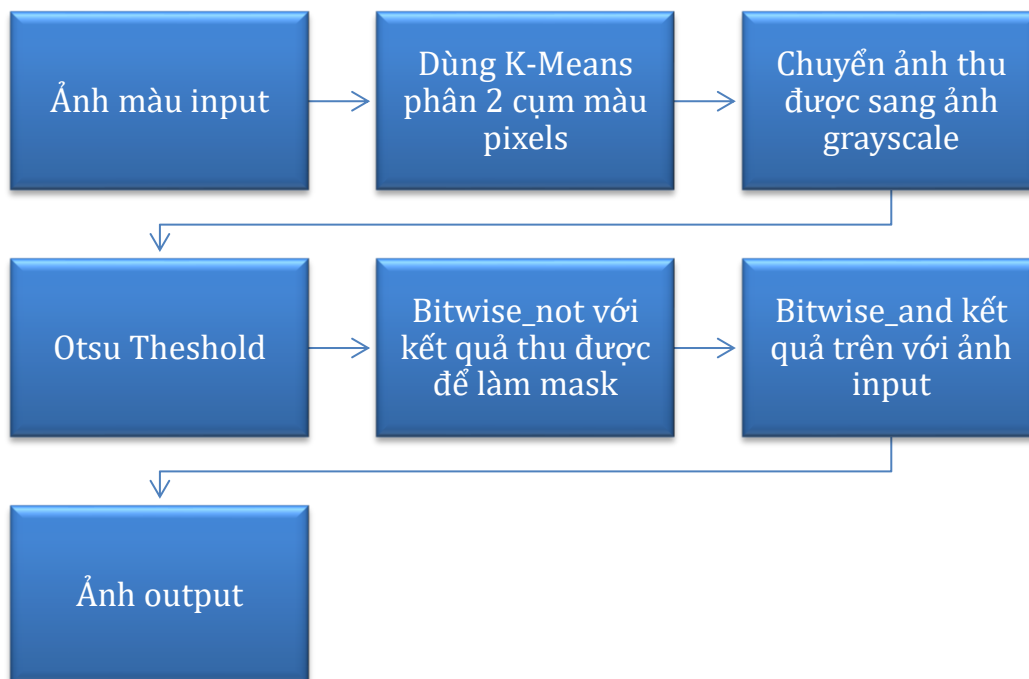
- Do có sử dụng Otsu để xử lý nên cũng chia ra 2 trường hợp như Otsu.

- Pixel ở foreground có cường độ lớn hơn cường độ pixel ở background.

- Cơ sở kết hợp: Từ ảnh màu input, ta dùng K-Means để phân 2 cụm màu pixels. Histogram thu được từ ảnh kết quả lúc này sẽ có 2 cột ở 2 giá trị pixel. Đây là một điều kiện rất lý tưởng để sử dụng Otsu để tìm ra threshold phân chia. Khi này ta chuyển ảnh màu thu được sau khi dùng K-Means chuyển sang ảnh mức xám và dùng otsu. Từ kết quả thu được sau khi dùng Otsu ta lấy nó làm mask và áp vào ảnh input để thu được kết quả cuối cùng.



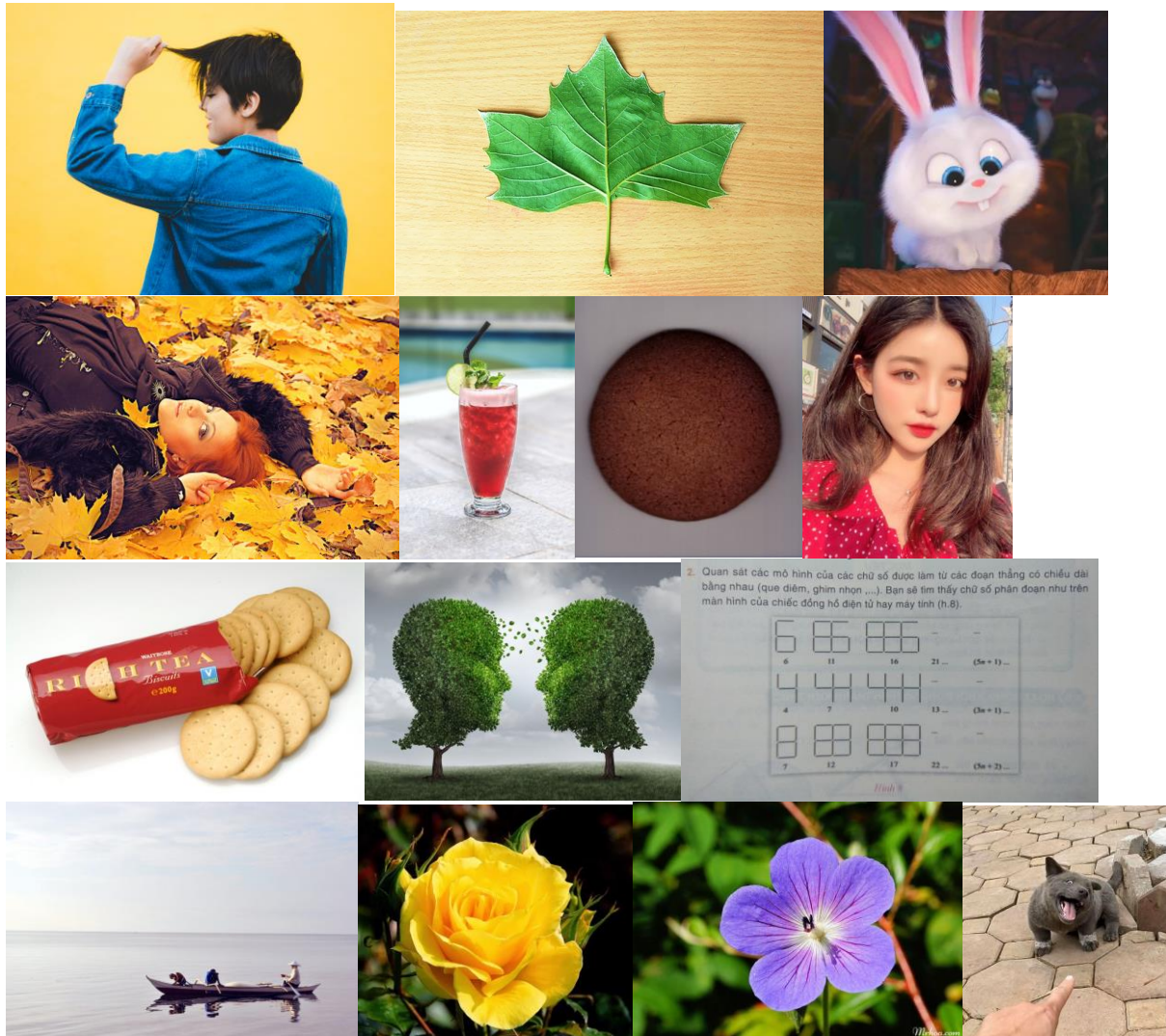
- Pixel ở foreground có cường độ nhỏ hơn cường độ pixel ở background.



1:7

4. EXPERIMENTAL RESULTS

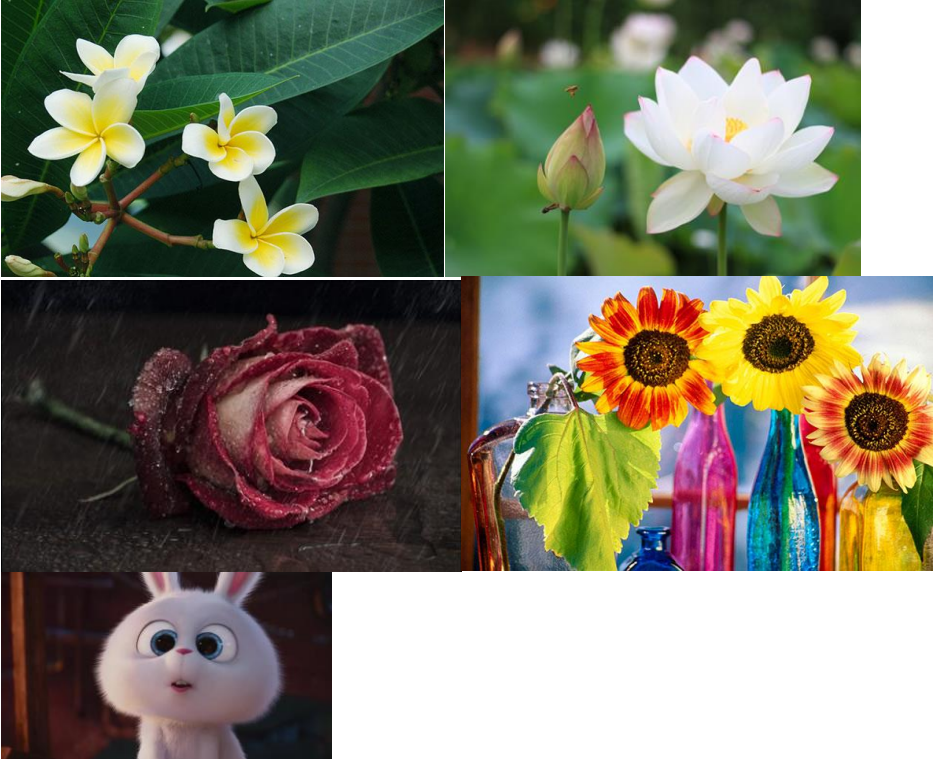
Dataset







1:8



1:9



Results

STT	Otsu	K-Means + Otsu
1		
2		

1:10

3



4



5



1:11

6



7



8

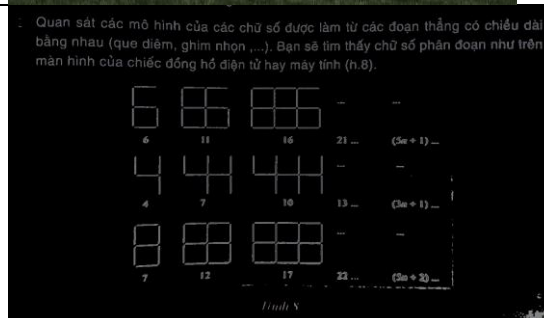
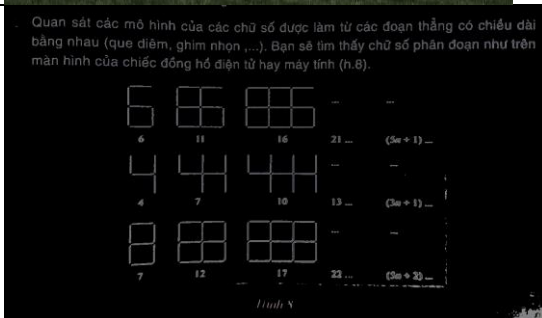


1:12

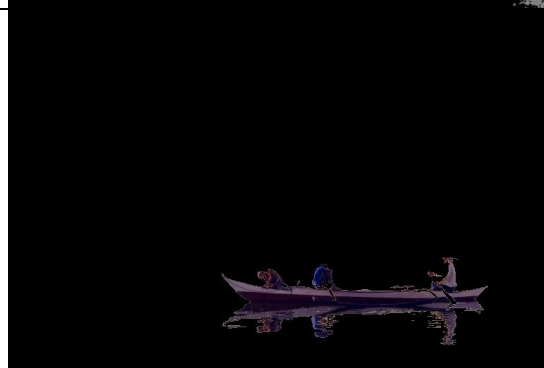
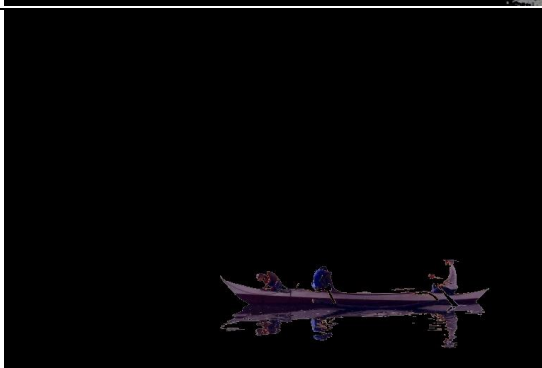
9



10



11



12



1:13

13



14



15



16



17



18



19



20

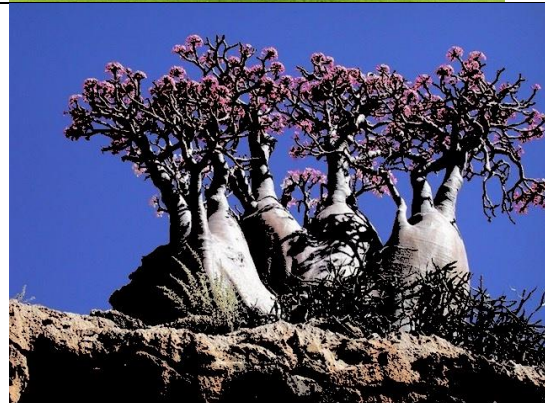


1:15

21



22












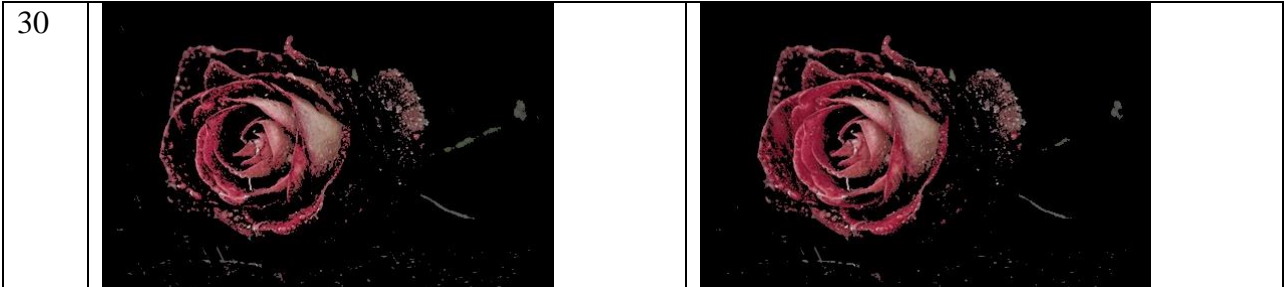
23



24



25		
26		
27		
28		
29		



5. DISCUSSION

- Qua các kết quả nhận được ở trên ta có thể thấy trong hầu hết các trường hợp thì việc xoá background bằng K-Means + Otsu hiệu quả hơn so với Otsu do kết quả của K-Means giữ lại được nhiều phần từ foreground hơn kết quả của Otsu.
- Ở cả 2 phương pháp, với ảnh chụp người trên nền đơn giản, màu background có màu gần với màu của da người nên khi xoá background thì màu da người bị xoá theo.
- Ảnh kết quả thu được từ K-Means + Otsu, phần foreground giữ được những phần vật thể sáng hơn (do vật thể phản chiếu ánh sáng) còn ảnh thu được từ Otsu thì không.
- Ảnh có background đơn giản và foreground gồm ít vật thể và có màu giống nhau, màu background và foreground tách bạch thì khi xài K-Means + Otsu cho kết quả rất tốt.

6. IMPROVEMENT

- Với phương pháp Otsu, ta sẽ điều chỉnh tham số nhận được sao cho không gian nhận foreground được tăng lên để có thể bắt được nhiều phần từ foreground mà ta mong muốn.
- Với phương pháp K-Means + Otsu thì ta tăng số lượng cụm k (bội số của 2) lên để đạt được kết quả tốt hơn.

7. CONCLUSION

- Có rất nhiều phương pháp có thể áp dụng để xoá background trong ảnh. Với Otsu thì thời gian cho ra kết quả nhanh nhưng chưa tốt. Với K-Means thì cho kết quả tốt hơn tuy nhiên thời gian chạy không được nhanh, k càng cao thì thời gian chạy càng lâu. Và cả 2 phương pháp trên không cho được kết quả tốt trên hình ảnh có foreground phức tạp hoặc background phức tạp. Hiện tại có các công cụ sử dụng hiệu quả hơn như Graph Cut, Deep Learning , ...

REFERENCES

- <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>
- <https://learnopencv.com/otsu-thresholding-with-opencv/>
- https://docs.opencv.org/4.x/d5/d38/group_core_cluster.html#ga9a34dc06c6ec9460e90860f15bcd2f88
- https://docs.opencv.org/4.x/d1/d5c/tutorial_py_kmeans_opencv.html