CENG466, Fall 2022, THE 3

1st Fırat Ağış
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
e2236867@ceng.metu.edu.tr

2nd Robin Koç
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
e2468718@ceng.metu.edu.tr

***Abstract*—**
***Index Terms*—**

I. Face Detection

For face detection, we propose an algorithm that use k-means to separate color groups, chooses the color group that is most likely to include skin colors and then uses morphological operations and convolution operation to detect areas that are most likely to include faces in the image. Our algorithm posses 3 distinct stages:

1) *Preprocessing:* Process the image to be more suitable for our algorithm and performs some processes to make the algorithm run a lot faster.
2) *K-Means:* Using k-means algorithm with randomized starting means to detect the pixels that are most likely to be in the color of skin tones.
3) *Context Based Postprocessing:* Using morphological operations and convolution operation to detect which clusters of the pixels that are found in the previous step to be the part of a face.

A. Preprocessing

resources/non-equal vs equal.jpg

Fig. 1. Difference in final results, on left, without histogram equalization and on right, with histogram equalization

1) We first read the relevant image. Shown in Row 1 of Figure 2.
2) (Optional) We perform histogram equalization to differentiate skin tones from similar colors in the image, most notably, the leaves in the 1_source.png and shirt and hair in 3_sources.png. Its effectiveness in demonstrated in Figure 1. Shown in Row 2 of Figure 2.
3) (Optional) We down-sampled the 2_source.png by a factor of 9 in order to complete the algorithm in a reasonable time.
4) We bit-sliced all images, eliminating the least significant bit. Colors that are differentiated by a single bit are most likely to be in the same color group, meaning this operation affected the quality of the k-means algorithm minimally while reducing the size of the color space by a factor of $2^3 = 8$. Shown in Row 3 of Figure 2.

B. K-Means

1) We performed k-means algorithm with randomized initial means, acquiring means and colors closest to that means.
2) We only took the colors whose means have the greatest red component using the knowledge that all faces are reddish in the context of color space.
3) We eliminated any colors whose red component is less then their blue or green component using the same logic as the previous step. Shown in Row 4 of Figure 2.

C. Context Based Postprocessing

1) (Optional) While processing 1_source.png and 3_sources.png, we used the assumption of faces being the central focus of images to eliminate pixels on the outer parts of the image. Shown in Row 1 of Figure 3.
2) We used opening operation with a 5x5 element to eliminate noise from the image. Shown in Row 2 of Figure 3.
3) We used closing operation with a 5x5 element to fix regions that are broken up during the previous step. Shown in Row 3 of Figure 3.
4) We used convolution with a elliptical mask, mimicking the shape of he face to determine locations where a face may lie. Shown in Row 4 of Figure 3.

Fig. 2. Face detection progress for preprocessing and k-means stages

5) We used thresholding to the result of the convolution, including the parts of the image that passed the treshholding, concluding the algorithm. Shown in Row 5 of Figure 3.

D. Design Process

After the implementation of basic k-means algorithm, execution time of the algorithm limited the speed of our further progress by trying different processes. For that reason, we implemented the preprocessing steps 3 and 4.

Then we looked at commonalities between regions that make up the face, which brought us the reasoning we used in the k-means steps 2 and 3.

After that, we experimented with histogram equalization to differentiate colors easier, giving us the results demonstrated in Figure 1 in the final product.

When we arrived at the final result of k-means, given in Row 4 of Figure 2, we determined the noise created by similarly colored regions to be a problem. To solve it, we implemented opening and closing mentioned in the postprocessing steps 2 and 3. For the noise not eliminated by these steps, around the edge of the image for 1_source.png, we implemented the postprocessing step 1.

Morphological operations didn't eliminated all non-face regions. To distinguish regions that are face shaped from others, we finally implemented the postprocessing steps 4 and 5.

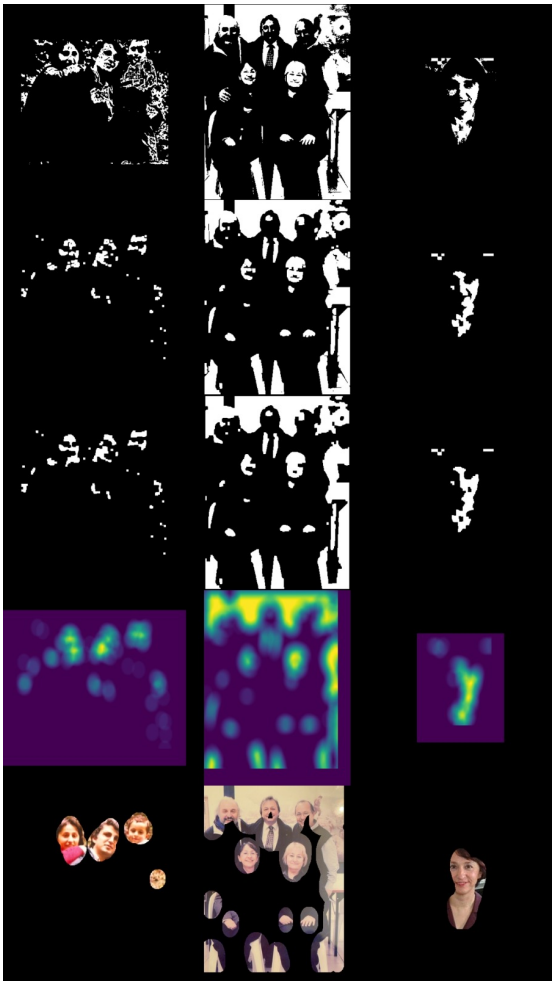

Fig. 3. Face detection progress for postprocessing stage

E. Results

Fig. 4. Original images and the end products

II. Pseudo-coloring

III. Dependencies

We used following libraries for the described reasons.

- **os:** Handling non-existent input or output paths.
- **math:** Performing square root operation.
- **numpy:** Executing array and matrix operations.
- **PIL:** Reading images and converting them to arrays.
- **mathplotlib:** Creating histograms as graphics and writing arrays as image files.