# Computer Vision Laboratories

## Project 1

EE/CPE 428 - 03

Computer Vision

Winter 2023

Group 4

Students: Chris Miglio, Eitan Klass, Nathan Jaggers

Project Due: 01/19/23

Instructor: Dr. Zhang

# Part A

## Assignment

Use two images for the following steps. One is the group picture you just took. For the second one, find a landscape image from the Internet (the image should be of at least 800x600 pixels).

1. Read and display the image.
2. Convert the image to grayscale image (In MATLAB use rgb2gray()).
3. Find the maximum and minimum intensity values in the grayscale image, and their corresponding spatial coordinates. If multiple locations are found, you only need to show one spatial location.
4. Record the size of the grayscale image file in bytes and note the quality of the image; consider the overall scene and small details.
5. Write your own code to reduce the resolution of the original grayscale image. Display the image and note the image quality. Determine the lowest resolution that allows acceptable viewing. Record and compare the size of this image to that of the original image.

## Procedure

Many of the steps taken to complete this part of the lab were derived from lecture material and the Intro to Matlab lecture during the first lab. Images were read and displayed using the built-in Matlab functions imread() and imshow(). Images were also grayscaled using the Matlab function rgb2gray().

To find the minimum and maximum intensities and their coordinates, the functions min(), max(), and find() were used. Min and max were used twice because one use of it with a single matrix argument will result in a row vector of maxes or mins from each column. Using it twice will find the extremes of the entire matrix. We also stored the indices of the max and min so we could find the coordinates of those values in the image. An alternative approach to finding the coordinates is using the find() function to test the image for the first occurrence of max or min and saving the resulting row and column.

The grayscale image was saved and once put in the directory as another file, we can see information on its size. Using dir() and getting the .bytes attribute of the class we can observe and save the size of the image.

To reduce the resolution of the image, we skip rows and columns of the image and write it back as a new image. That way we decrease the density of pixels and reduce the image size.

**Results**

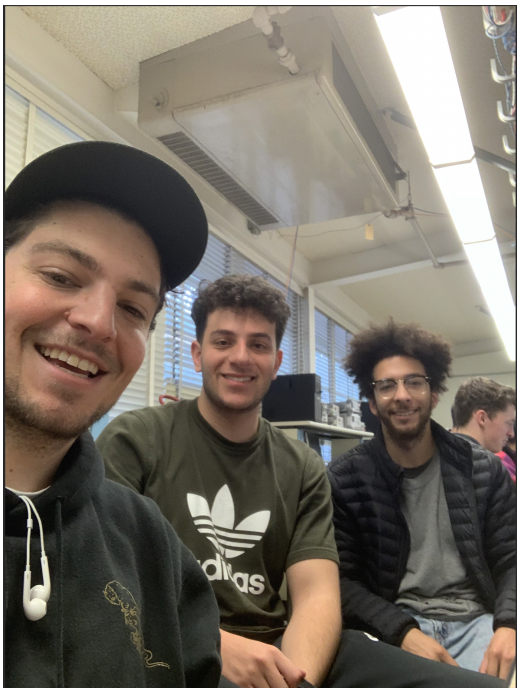

Figure 1: Original picture that was taken

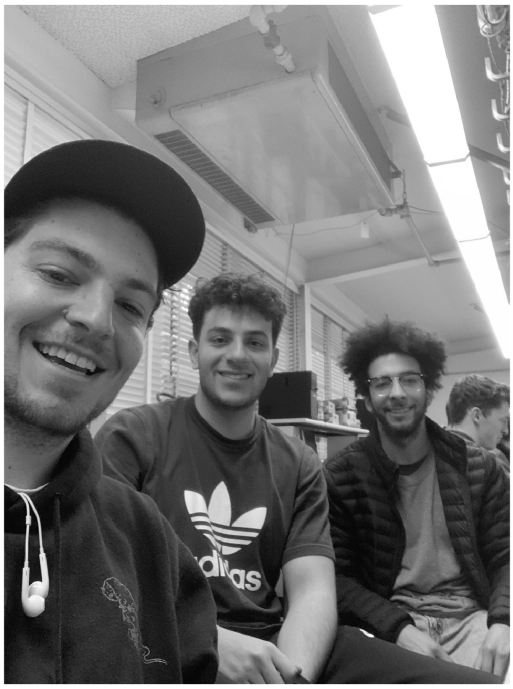

Figure 2: Grayscale image of original picture

In Figure 2, you can clearly still see and recognize all members of the group. You can pick up the little details easily like the Adidas letters on Eitans shirt and edge between Chris' hair and his hat. This is when the image is at 726647 bytes.

Figure 3: Reduced quality image of original picture

Figure 3 is last acceptable size appropriate for viewing but with a clearly degraded quality is at 5370 bytes. This new image is 135 times smaller but certainly at a cost of information. The Adidas lettering is unrecognizable at this point and it is very difficult to see Nathan's face. Overall you can still certainly tell it is a picture of the lab group but the fine details are hard to see.

Figure 4:  Original Sunflower image

Figure 4:  Grayscale Sunflower image


Figure 5 & 6:  Max and Min Intensities for Sunflower

Here in Figure 4 you can clearly see the and recognize the sunflower. You can distinguish its petals from each other and even see all the seeds at the center of the flower very well. When

comparing Figure 4 to the max and min intensity locations, in Figures 5 & 6 we can visually confirm those are some of the brightest and darkest spots in the image. The image in Figure 4 has a size of 233035 bytes

Figure 7: Reduced Sunflower Image

In Figure 7 we can still recognize the sunflower however it is very difficult to appreciate the fine details as we were able to see before. We lose clarity in the flowers in the background as well as the petals and seeds in the flower up front. The size of this image is 6121 bytes and is 38 times smaller than the original grayscale image.

# Part B

## Assignment

The main goal of this assignment is to determine the number and area of the bacteria in the given grayscale image (electronic version of the image can be found on Canvas).

1. Apply a threshold to the image to separate the bacteria from the background. Display the grayscale image and the binary image. Compute the total area of the bacteria in the image.
2. Label pixels belonging to each bacterium with a unique label. Display the labeled image.
3. Compute the area of each bacterium in the binary image. Would you say all the bacteria belong to the same family?

## Procedure

To analyze the grayscale bacteria image in MATLAB, the bacteria image was first loaded into the MATLAB IDE using the imread() function. The grayscale image was then converted to a binary image using the thresholding technique. The threshold was set to a value of 100 and a boolean expression set all values above this threshold to a value of 1 (white) and all values below this threshold to a value of 0 (black). This image was then displayed in the MATLAB IDE.

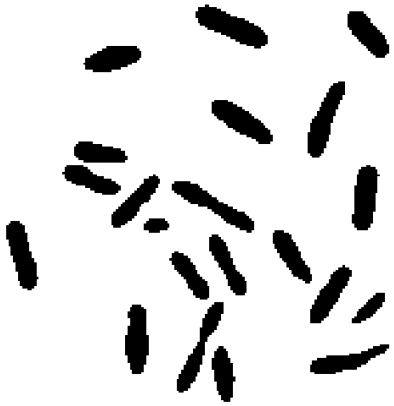

Figure 4: Binary Image of the bacteria image using the threshold of 100

A calculation was then performed to determine the area of all bacteria in the image. Using the bwarea() function, a total count of the number of black pixels was totaled representing the total area of the bacteria in the image. Each bacteria was then labeled using the bwlabel() function built into MATLAB. This function returns an image with the same size as the original image, but displays each connected component with a unique integer value that represents a different color. This picture is then visualized with the imshow() function.

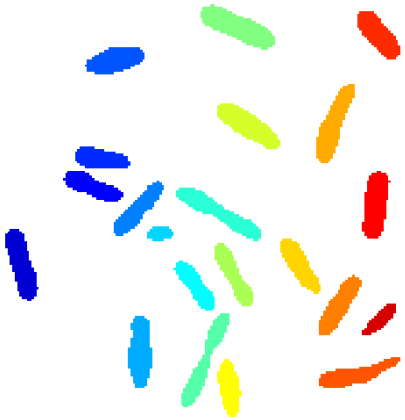

Figure 5: Bacteria image displayed with unique integer value representing the different connected components or bacteria.

Using the same bwlabel() function provided a count for the number of connected components, returning a value of 21 bacteria. Performing a manual count provided the same

value. Finally to obtain the area of each connected component, the function regionprops() was used. This function returned a structure containing information on the number of pixels in each connected component.

## Results

Performing the grayscale to binary image conversion is displayed in Figure 4. This image shows the bacteria in black and the background in white based on the set threshold value of 100. The bwarea() function returned a value of 4142 pixels, indicating that the bacteria make up a total of 4141 pixels in the total of 31684 pixels in the entire image. The bwlabel() returned the image in Figure 5 showing each connected component as a unique integer value or color. The bwlabel() also returned the total number of bacteria–connected components–with a value of 21, which matched the manual count. Finally, the pixel area of each bacteria–connected component–was calculated using the regionprops() function. This function returned a scalar number of pixels in the connected region.

Table 1: Area of each bacteria

| | |
|---|---|
| 1 | 237 |
| 2 | 172 |
| 3 | 159 |
| 4 | 198 |
| 5 | 182 |
| 6 | 236 |
| 7 | 53 |
| 8 | 156 |
| 9 | 247 |

| | |
|---|---|
| 10 | 229 |
| 11 | 288 |
| 12 | 176 |
| 13 | 241 |
| 14 | 161 |
| 15 | 182 |
| 16 | 249 |
| 17 | 200 |
| 18 | 198 |
| 19 | 199 |
| 20 | 235 |
| 21 | 85 |

This table displays the count of each bacteria and the pixel count associated with the specific bacteria.

# Teamwork

Chris

Worked on Part B of the lab outside of class and did exercise in python to see how the development process differs from Matlab. Wrote up the content of the Part B section of the lab.

Eitan

Provided the code that is in the appendix. Provided the group picture results as well as the captions for the results

Nathan

Set up the document and wrote out the content of Part A of the lab. Included Sunflower pictures and analysis.

# Appendix

```
% Part 1

% Read and display the image of the group
pic = imread('ee428project1pic.jpg');
imshow(pic);

% Convert the image to grayscale
grPic = rgb2gray(pic);
imshow(grPic);

% Min and max intensity values in the grayscale image
[minVal, minIndex] = min(min(grPic));
[maxVal, maxIndex] = max(max(grPic));
[minRow, minCol] = find(grPic == minVal, 1);
[maxRow, maxCol] = find(grPic == maxVal, 1);

% Size in bytes
fileinfo = dir('grPic')
filesize = fileinfo(1).bytes

% Reduced quality of image
reduced = grPic(1:20:end, 1:20:end);
figure;
imshow(reduced);

% Part 2

% Display the original image
bacteriaOriginal = imread("bacteria.bmp");
imshow(bacteriaOriginal);

% Threshold process
threshold = 100;
tImg = bacteriaOriginal < threshold;
imshow(tImg)

% Compute the total area
area = bwarea(tImg)
disp(area)

% Label pixels
labeledImage = bwlabel(tImg);
imshow(labeledImage, [])

% Get the area of each bacterium
for i = 1 : max(labeledImage(:))
   area = bwarea(labeledImage == i)
end
```