Project 1 Report

Resolution Reduction and Image Labeling

CPE428-01,02 - Computer Vision

Prepared for: Professor Xiaozheng Zhang
Prepared By: The “A” Team; Nikhil Patolia, Alec Hardy

Date Submitted: Monday, January 18, 2020

Table of Contents

Part A

Reading and Displaying the image

We began this project by finding the images to take for this project. For one of the images, we took a picture of our group and for the other we found a landscape image of a wave in the ocean. We first began by reading in the images using “imread”. We then created a figure for each image and displayed it using “imshow”. These images are shown below as Figure 1 and Figure 2.

Figure 1. Landscape original image. Color

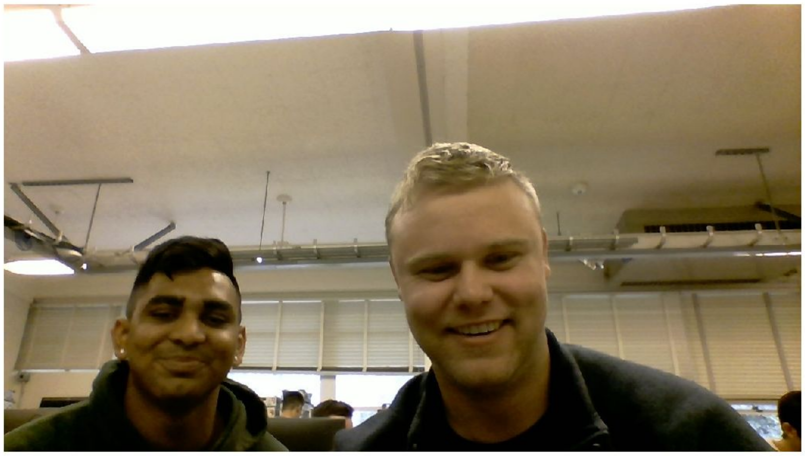

Figure 2. Group original image. Color

Obtaining Grayscale Image

To create a grayscale image, we simply took the image and used the function “rgb2gray” to convert our current image to a new grayscale image. The grayscale sample of our landscape image from Figure 1 is shown below as Figure 3.

Figure 3. Landscape original image. No resolution change. 742x990

Obtaining Maximum and Minimum Pixel Intensity Value.

To get the max brightness pixel (the lightest color pixel), we used "max(max(img_g))". This got the max brightness pixel's row and column but since there could be multiple pixels with the same max brightness and the spec only asks for the first instance, we run the line "max1 = max1(1);" to get the first occurence. We did this for both images and followed the same process for getting the min brightness value(darkest pixel).

```
Image 1: Max Pixel Val: 252 (0.988); Location: (500, 125)
         Min Pixel Val:   0 (0.000); Location: (711, 1)

Image 2: Max Pixel Val: 255 (1.000); Location: (420, 7)
         Min Pixel Val:   0 (0.000); Location: (697, 117)
```

Reducing the Resolution of the Image

To reduce the resolution of the images that we were presented, we decided to get a sample of the image, reducing the total number of pixels in the image to ¼ of the original size. We did this by getting the pixels that resided in every other row and every other column. The code that accomplished this was "halved_mat = mat_input(1:2:end,1:2:end);". We reduced the resolution of each image a total of 6 times and kept them all the same size, allowing the true effect of our actions to be seen. The images are found in Appendix A as Figure 3 - Figure 16. With each reduction in resolution, we can see that the image becomes more pixelate and at around the 4th reduction for the landscape image and the 5th reduction for the group image, we can see that the subject of the pictures is indeterminable.

```
Original Image 1 grayscale (UINT8)  size: 734580 bytes
Reduced Quality Image 1 (Halved 5x) size: 744 bytes
Original Image 2 grayscale (UINT8)  size: 921600 bytes
Reduced Quality Image 2 (Halved 5x) size: 920 bytes
```

Part B

Applying a Threshold

For Part B of the lab, we were given a grayscale picture of bacteria and were told to separate the background from the subject(the bacteria). We did this by setting a threshold of 100, done with "BW = im17 > 100;" which sets all pixels that are above 100 to white and those below to black.

Computing the Total Area

After the background was separated from the bacteria, we were able to get the total area of all the bacteria pixels by using the line "area = sum(~BW(:));".

Total area of bacteria: 4083 pixels

Labelling the Bacteria

After obtaining a binary image of the bacteria, we were able to use bwlabel and "groups = regionprops(labels, 'Centroid', 'FilledArea');" to further separate each bacteria from each other. This gave us a list with each bacteria in it and with that we were able to label the image by placing a text on each of the connected components. In total, there were 21 bacteria.

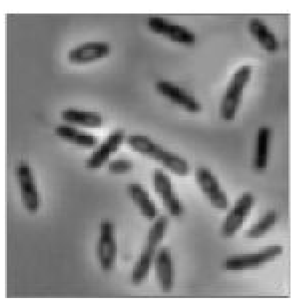

Figure 17. Original Image of Bacteria.

Figure 18. Labelled Image of Bacteria.

Computing Individual Bacteria Areas

When we separated the bacteria into their own list items with regionprops, we added the information of the filledArea which easily allowed us to calculate the area of each bacteria. We have shown the results below and as we can clearly see, all of these bacteria do not belong to the same family since some (such as #7 and #21) are much smaller than the others.

```
Area of bacteria #1 is 237
Area of bacteria #2 is 172
Area of bacteria #3 is 159
Area of bacteria #4 is 198
Area of bacteria #5 is 182
Area of bacteria #6 is 236
Area of bacteria #7 is 53
Area of bacteria #8 is 156
Area of bacteria #9 is 247
Area of bacteria #10 is 229
Area of bacteria #11 is 288
Area of bacteria #12 is 176
Area of bacteria #13 is 241
Area of bacteria #14 is 161
Area of bacteria #15 is 182
Area of bacteria #16 is 249
Area of bacteria #17 is 200
Area of bacteria #18 is 198
Area of bacteria #19 is 199
Area of bacteria #20 is 235
Area of bacteria #21 is 85
```

Appendix A - Images at Different Resolutions

Figure 3. Landscape original image. No resolution change. 742x990

Figure 4. Landscape image, resolution halved once. 371x495

Figure 5. Landscape image, resolution halved twice. 186x248

Figure 6. Landscape image, resolution halved thrice. 93x124

Figure 7. Landscape image, resolution halved 4x. 47x62

Figure 8. Landscape image, resolution halved 5x. 24x31

Figure 9. Landscape image, resolution halved 6x. 12x16

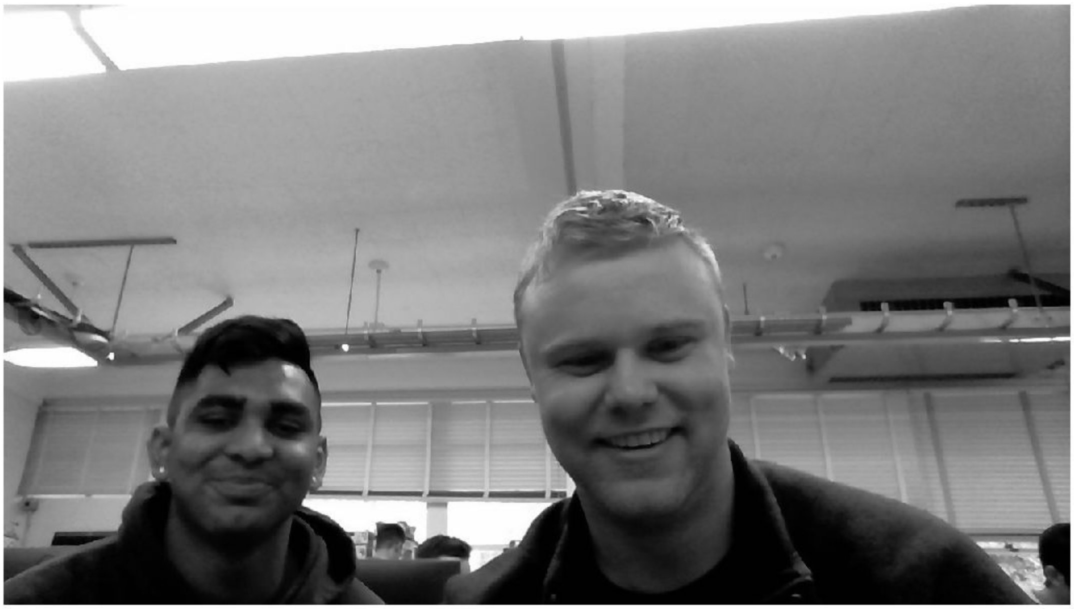

Figure 10. Group original image. No resolution change. 720x1280

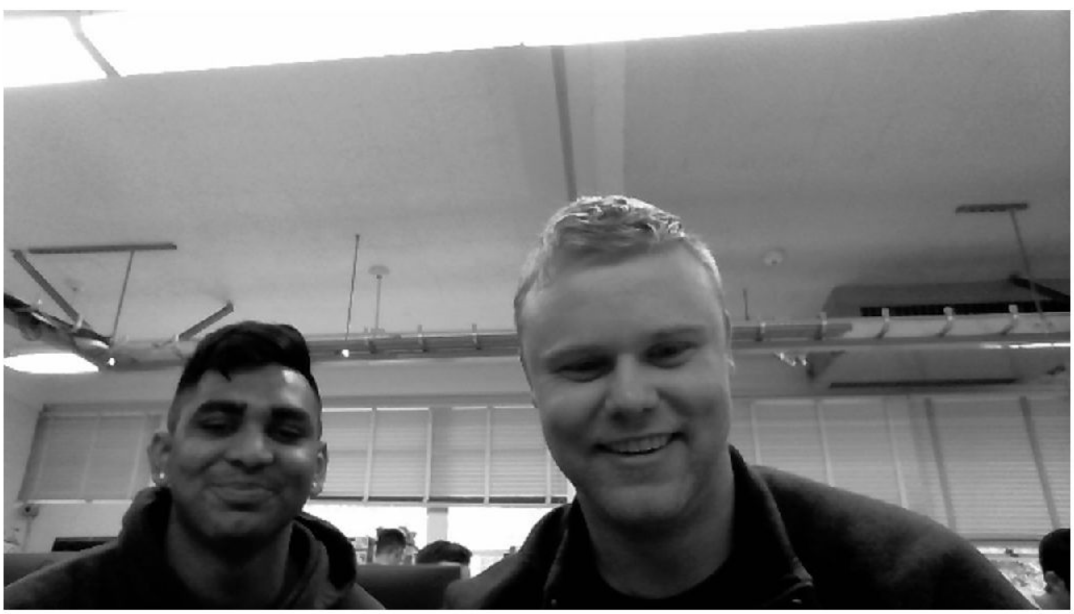

Figure 11. Group image, resolution halved once. 360x640

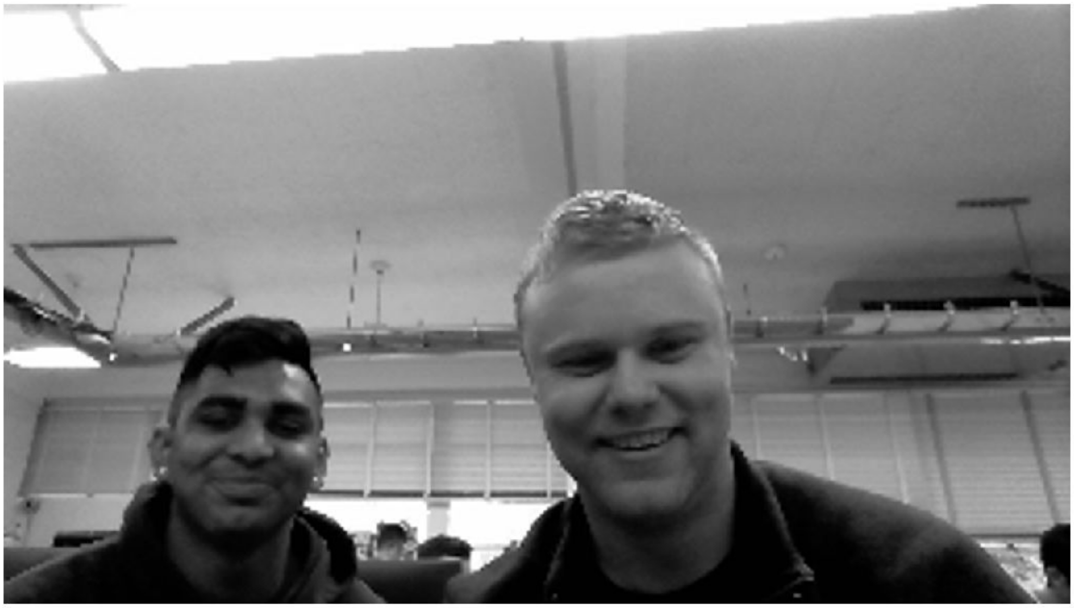

Figure 12. Group image, resolution halved twice. 180x320

Figure 13. Group image, resolution halved thrice. 90x160

Figure 14. Group image, resolution halved 4x. 45x80

Figure 15. Group image, resolution halved 5x. 23x40


Figure 16. Group image, resolution halved 6x. 12x20

Appendix B - MATLAB Code

```
% Read and Display the image
im1 = imread('landscape.jpg');
im2 = imread('group.jpg');

figure();
imshow(im1);
xlabel("Figure 1. Landscape original image. Color");

figure();
imshow(im2);
xlabel("Figure 2. Group original image. Color");

%Convert the image to grayscale
im1_g = rgb2gray(im1);
im2_g = rgb2gray(im2);

max1 = max(max(im1_g));
max1 = max1(1);
[Xmax1, Ymax1] = find(im1_g==max1);
max2 = max(max(im2_g));
max2 = max2(1);
[Xmax2, Ymax2] = find(im2_g==max2);

min1 = min(min(im1_g));
min1 = min1(1);
[Xmin1, Ymin1] = find(im1_g==min1);
min2 = min(min(im2_g));
min2 = min2(1);
[Xmin2, Ymin2] = find(im2_g==min2);

fprintf('Image 1: Max Pixel Val: %3d (%.3f); Location: (%d, %d)\n', max1, double(max1)/255, Xmax1(1),
Ymax1(1));
fprintf('         Min Pixel Val: %3d (%.3f); Location: (%d, %d)\n\n', min1, double(min1)/255, Xmin1(1),
Ymin1(1));

fprintf('Image 2: Max Pixel Val: %3d (%.3f); Location: (%d, %d)\n', max2, double(max2)/255, Xmax2(1),
Ymax2(1));
fprintf('         Min Pixel Val: %3d (%.3f); Location: (%d, %d)\n\n', min2, double(min2)/255, Xmin2(1),
Ymin2(1));

% Record filesize and comment on quality
fprintf('Image 1 grayscale (UINT8) size: %d bytes\n', size(im1_g,1)*size(im1_g,2));
fprintf('Image 2 grayscale (UINT8) size: %d bytes\n', size(im2_g,1)*size(im2_g,2))

% Reduce the quality of each image a bunch of times
img1_c1 = halve_mat(im1_g);
img1_c2 = halve_mat(img1_c1);
img1_c3 = halve_mat(img1_c2);
img1_c4 = halve_mat(img1_c3);
img1_c5 = halve_mat(img1_c4);
img1_c6 = halve_mat(img1_c5);
figure();
imshow(im1_g);
```

```
xlabel("Figure 3. Landscape original image.  No resolution change. 742x990");
figure();
imshow(imresize(img1_c1, 2));
xlabel("Figure 4. Landscape image, resolution halved once. 371x495");
figure();
imshow(imresize(img1_c2, 4));
xlabel("Figure 5. Landscape image, resolution halved twice. 186x248");
figure();
imshow(imresize(img1_c3, 8));
xlabel("Figure 6. Landscape image, resolution halved thrice. 93x124");
figure();
imshow(imresize(img1_c4, 16));
xlabel("Figure 7. Landscape image, resolution halved 4x. 47x62");
figure();
imshow(imresize(img1_c5, 32));
xlabel("Figure 8. Landscape image, resolution halved 5x. 24x31");
figure();
imshow(imresize(img1_c6, 64));
xlabel("Figure 9. Landscape image, resolution halved 6x. 12x16");

%Next image
img2_c1 = halve_mat(im2_g);
img2_c2 = halve_mat(img2_c1);
img2_c3 = halve_mat(img2_c2);
img2_c4 = halve_mat(img2_c3);
img2_c5 = halve_mat(img2_c4);
img2_c6 = halve_mat(img2_c5);
figure();
imshow(im2_g);
xlabel("Figure 10. Group original image.  No resolution change. 720x1280");
figure();
imshow(imresize(img2_c1, 2));
xlabel("Figure 11. Group image, resolution halved once. 360x640");
figure();
imshow(imresize(img2_c2, 4));
xlabel("Figure 12. Group image, resolution halved twice. 180x320");
figure();
imshow(imresize(img2_c3, 8));
xlabel("Figure 13. Group image, resolution halved thrice. 90x160");
figure();
imshow(imresize(img2_c4, 16));
xlabel("Figure 14. Group image, resolution halved 4x. 45x80");
figure();
imshow(imresize(img2_c5, 32));
xlabel("Figure 15. Group image, resolution halved 5x. 23x40");
figure();
imshow(imresize(img2_c6, 64));
xlabel("Figure 16. Group image, resolution halved 6x. 12x20");
```

```
%% Part B

im17 = imread('bacteria.bmp');
figure();
imshow(im17);
xlabel("Figure 17. Original Image of Bacteria.");

BW = im17 > 100;
figure();
imshow(BW);
area = sum(~BW(:));

fprintf("Total area of bacteria: %d pixels\n", area);

labels = bwlabel(~BW);

groups = regionprops(labels, 'Centroid', 'FilledArea');
imshow(BW)
xlabel("Figure 18. Labelled Image of Bacteria.");
for num = 1:numel(groups)
    coordinate = groups(num).Centroid;
    text(coordinate(1), coordinate(2), int2str(num), 'Color', 'red');
    fprintf("Area of bacteria #%d is %d\n", num, groups(num).FilledArea);
End
```

```
function [ halved_mat ] = halve_mat( mat_input
)
%Returns a "compressed" copy of the input image
matrix
    halved_mat = mat_input(1:2:end,1:2:end);
end
```