Shimin Lei, Orian Churney, Nayiri Krzysztofowicz

ECE 6782/4501: Digital Image Processing

26 September 2018

**Methodology**

The first task was to count the number of bacteria in the image provided, ‘confocal.mat.’ The image was loaded into matlab and converted to grayscale, then different threshold levels were tried to see whether the individual bacteria shapes could be captured. There seemed to be several levels of grey that represented the individual bacteria, depending on the depth. When a small threshold (~0.1) was used, the peripheral bacteria shapes would be captured, but the middle bacteria was seen as one large blob. When a larger threshold was used (~0.6), the peripheral bacteria was erased but the middle blob was separated into distinct bacteria. Using either threshold resulted in dramatically undercounting the bacteria present. As a result, we thresholded the image twice, at two distinct thresholds that would allow us to capture the individual bacteria at two levels of brightness. While some bacteria were double-counted, after overlaying the two binary images from the different thresholds, not many bacteria overlapped, so we assumed the double-counting could offset the bacteria that we were never able to isolate and count. After using the two thresholds, our bacteria count closely matched what we had counted by hand.

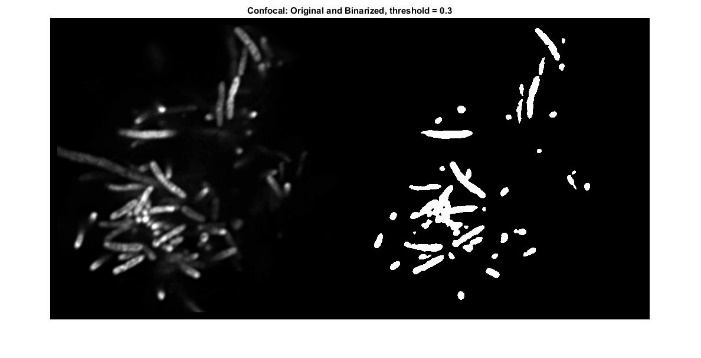
Next, the binary image was skeletonized to obtain the length of the bacteria. The sum of the white pixels of the skeleton image was found. This value would need to be paired with the magnification level of the bacteria image to compute the actual length of the bacteria, since the scale of the image was not provided.

The above process was repeated for the second image, latticeLightSheet.jpg. The overall structure of the image processing method remained the same, but at every step of the process we had to visually verify that the expected results were being produced, and modify any hard-coded values as we saw necessary. For example, for the lattice image a smaller lower bound on large object removal had to be set, as we found this produced a cleaner final binary image. Additionally, no further processing had to be done on the skeletonized image to remove branches, since there were virtually none to begin with.

The extra credit assignment was attempted, but never completed. The 3D.tif image was imported, and the stacks of the image were iterated through to obtain a 3D matrix. This was viewed in the MATLAB Volume Viewer app to get a better intuition of the data we were working with. However, we came up with no further processing that could be done on the image, not being sure about how to detect distinct objects in the third dimension.

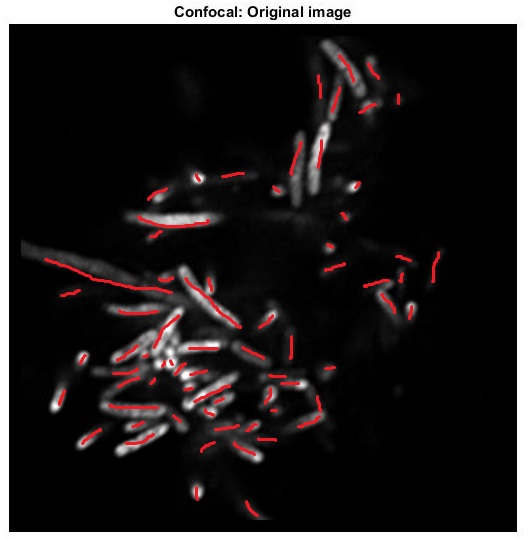
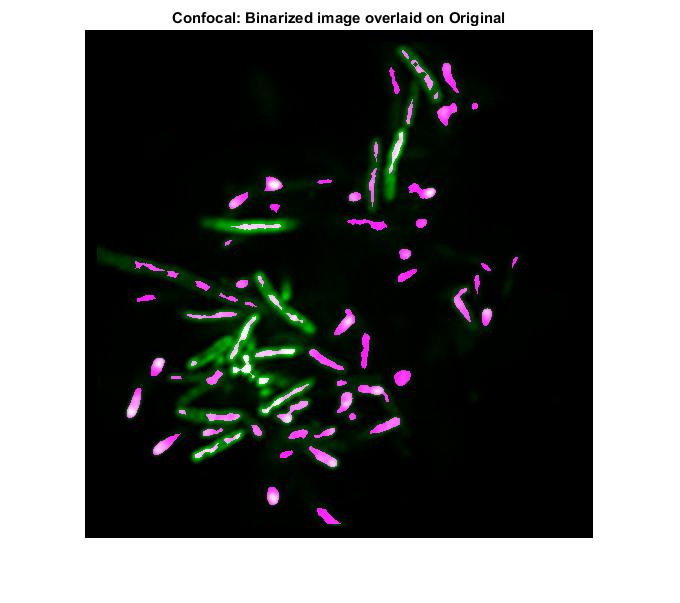
**Results**

Below are the bacteria thresholded from 0.1 to 0.8, in 0.1 increments. As the binary images show, each threshold isolates different  bacteria in the image.

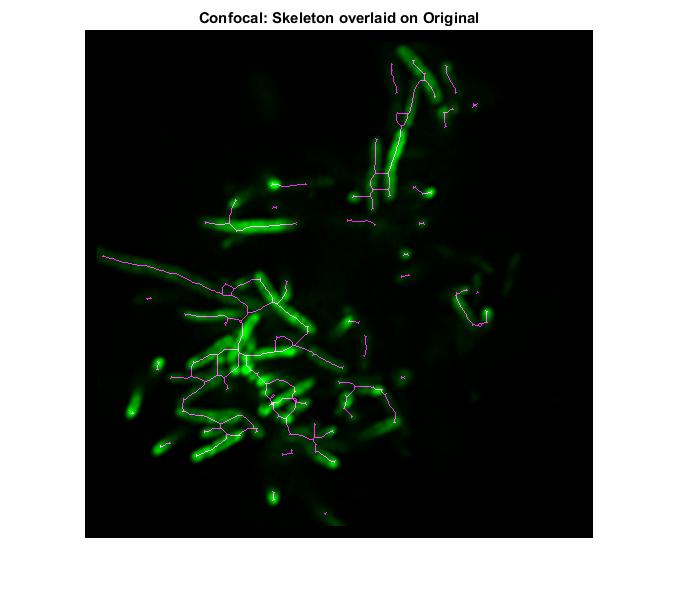
From these observations, the confocal image was thresholded from 0.1 to 1 in .001 increments, and the results minus large white objects were summed. In this way, as the small bacteria objects became visible in the image, they were added to the entire binary image. The results of this process are seen below, with the original and binary image displayed side-by-side, then overtop each other for comparison. The binary image (magenta) visually seemed to do an adequate job of capturing the count of the bacteria cells (green).



This process led to a bacteria count of 72. As a verification, the bacteria in the original image were counted, with the number totalling to around 70. The image below shows the bacteria we included in this count (red marks). The results were more accurate than we had hoped for, given how blurry and difficult to process the original image was.



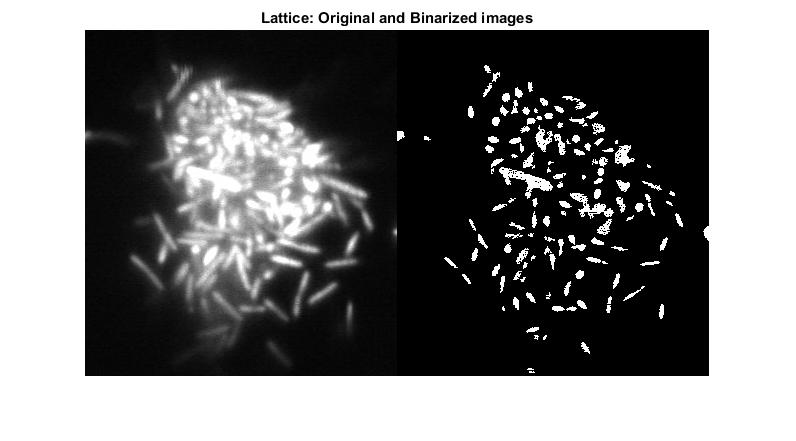
To compute cell length, the original image was thresholded then skeletonized. Below is the original confocal image (green) and overlaid skeleton (magenta). The pixel count was found to be 2265, which would need to be translated to an actual distance, given the microscope resolution for these photos.



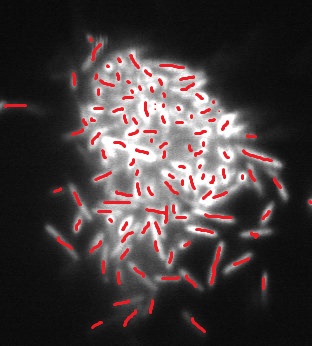
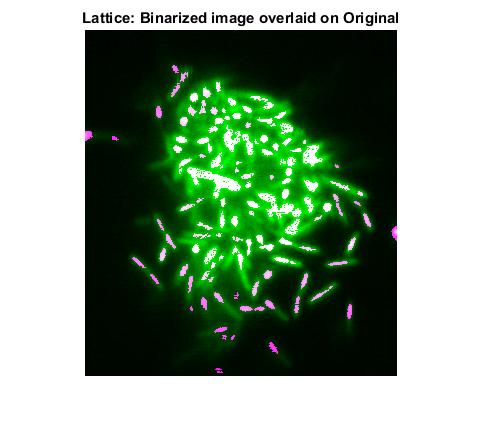
***\*\*\*What can you say about how the length of cells varies across the image?***

Our image processing method was tested on the second image, ‘latticeLightSheet.jpg.’ The method worked somewhat smoothly for the second image, requiring only minor tweaks such as adjusting threshold values and sizes of objects to be removed. The same process of summing the lattice image thresholded at different values was followed, yielding the image below.

What are the limitations of your method for the new  data?



The binary image overlaid on the original is seen below. This process yielded 125 bacteria counted, whereas counting by hand we got around 130 bacteria. Again, we were surprised by how accurately our processing captured the bacteria count.



To compute cell length, the lattice image was skeletonized, shown below. The length of the skeleton was found to be 2393 pixels, which again would need the microscope resolution to scale to a real length unit.



4. Extra credit (up to 15 points):  The 2D images used above are slices from a 3D stack.  Try your methods on the corresponding 3D lattice light sheet dataset (‘3D.tif’.) What  do you observe about cell length now? Not required.

****

**MATLAB Code**

For confocal.mat:

load('confocal.mat');

confocal = cluster1Processed5;

clear cluster1Processed5;

confocal=mat2gray(confocal);

figure, imshow(confocal), title('Confocal: Original image')

% this part initially uncommented to display confocal at different thresholds

% for i=.1:.1:.9

% confocalBin = imbinarize(confocal,i);

% confocalBin = bwareaopen(confocalBin, 15);

% figure, imshowpair(confocal,confocalBin,'montage'), title(['Confocal: Original and Binarized, threshold = ',num2str(i)]);

% end

%% CELL COUNT

confocalBin = imbinarize(confocal,0.99);

for i=0.07:.001:1

confocalBintemp = imbinarize(confocal,i);

confocalBintemp = confocalBintemp & ~bwareaopen(confocalBintemp, 400);

confocalBin = confocalBin | confocalBintemp;

end

confocalBin = imerode(confocalBin, strel('sphere',2));

confocalBin = bwareaopen(confocalBin, 20);

figure, imshowpair(confocal,confocalBin,'falsecolor'), title('Confocal: Binarized image overlaid on Original');

figure, imshowpair(confocal,confocalBin,'montage'), title('Confocal: Original and Binarized images');

[L,n]=bwlabel(confocalBin); %n is total number

n

%% LENGTH

confocalBin = imbinarize(confocal,.1);

confocallength = bwmorph(confocalBin,'skel',Inf);

%figure, imshowpair(confocal,confocallength,'falsecolor'), title('skeleton')

%remove branches / noise and calculate length

%code from: https://www.mathworks.com/matlabcentral/answers/88284-remove-the-spurious-edge-of-skeleton

skel = confocallength;

B = bwmorph(skel, 'branchpoints');

E = bwmorph(skel, 'endpoints');

[y,x] = find(E);

B\_loc = find(B);

Dmask = false(size(skel));

for k = 1:numel(x)

D = bwdistgeodesic(skel,x(k),y(k));

distanceToBranchPt = min(D(B\_loc));

Dmask(D < distanceToBranchPt-1) = true;

end

confocallength = skel - Dmask;

%figure, imshowpair(confocal,confocallength,'montage'), title('skeleton')

l = sum(confocallength(:)) % print red lymphocyte skeleton length

figure, imshowpair(confocal,confocallength,'falsecolor'), title('Confocal: Skeleton overlaid on Original')

For latticeLightSheet.jpg:

lattice = imread('latticeLightSheet.jpg');

% figure, imshow(lattice), title('original')

lattice = mat2gray(lattice);

% this part originally uncommented to see lattice at different thresholds

% for i=.2:.1:.9

% latticeBin = imbinarize(lattice,i);

% latticeBin = bwareaopen(latticeBin, 20);

% figure, imshowpair(lattice,latticeBin,'montage'), title('Confocal: Original and Binarized');

% end

%% CELL COUNT

latticeBin = imbinarize(lattice,0.99);

for i=0.07:.01:1

latticeBintemp = imbinarize(lattice,i);

latticeBintemp = latticeBintemp & ~bwareaopen(latticeBintemp, 60);

latticeBin = latticeBin | latticeBintemp;

end

latticeBin = bwareaopen(latticeBin, 10);

figure, imshowpair(lattice,latticeBin,'montage'), title('Lattice: Original and Binarized images');

figure, imshowpair(lattice,latticeBin,'falsecolor'), title('Lattice: Binarized image overlaid on Original');

[L,n]=bwlabel(latticeBin); %n is total number

n % display number of bacteria

%% LENGTH

latticeLength = bwmorph(latticeBin,'skel',Inf);

figure, imshowpair(lattice,latticeLength,'falsecolor'), title('Lattice: Skeleton overlaid on Original')

% visually determined that additional processing of skeleton was unnecessary

l = sum(latticeLength(:)) % print red lymphocyte skeleton length

For 3D.tif:

close all

clc

clear tiff\_stack

tiff\_info = imfinfo('3D.tif'); % return tiff structure, one element per image

tiff\_slice = imread('3D.tif', 1);

tiff\_slice(tiff\_slice<=15000) = 0;

tiff\_slice(tiff\_slice>15000) = 1;

tiff\_stack = tiff\_slice; % read in first image

%concatenate each successive tiff to tiff\_stack

for ii = 2 : size(tiff\_info, 1)

tiff\_slice = imread('3D.tif', ii);

tiff\_slice\_mod = tiff\_slice;

tiff\_slice\_mod(tiff\_slice\_mod>0) = 0;

for i=5000:500:25000

temp = tiff\_slice;

temp(temp<=i) = 0;

temp(temp>i) = 1;

temp = medfilt2(temp, [9 9]);

temp = temp & ~bwareaopen(temp, 200);

temp = bwareaopen(temp,20);

tiff\_slice\_mod = tiff\_slice\_mod | temp;

end

tiff\_slice = tiff\_slice\_mod;

tiff\_stack = cat(3 , tiff\_stack, tiff\_slice);

end