EXPERIMENT 2
ECE 6782
DIGITAL IMAGE PROCESSING

TEAM ZEHN

Jie Wang (jw4hr)
Akshat Verma (av2zf)

Goal :

Localize single bio-molecules in 3D space using superresolution microscopy with double helix point spread function

Summary of proposed methodology:

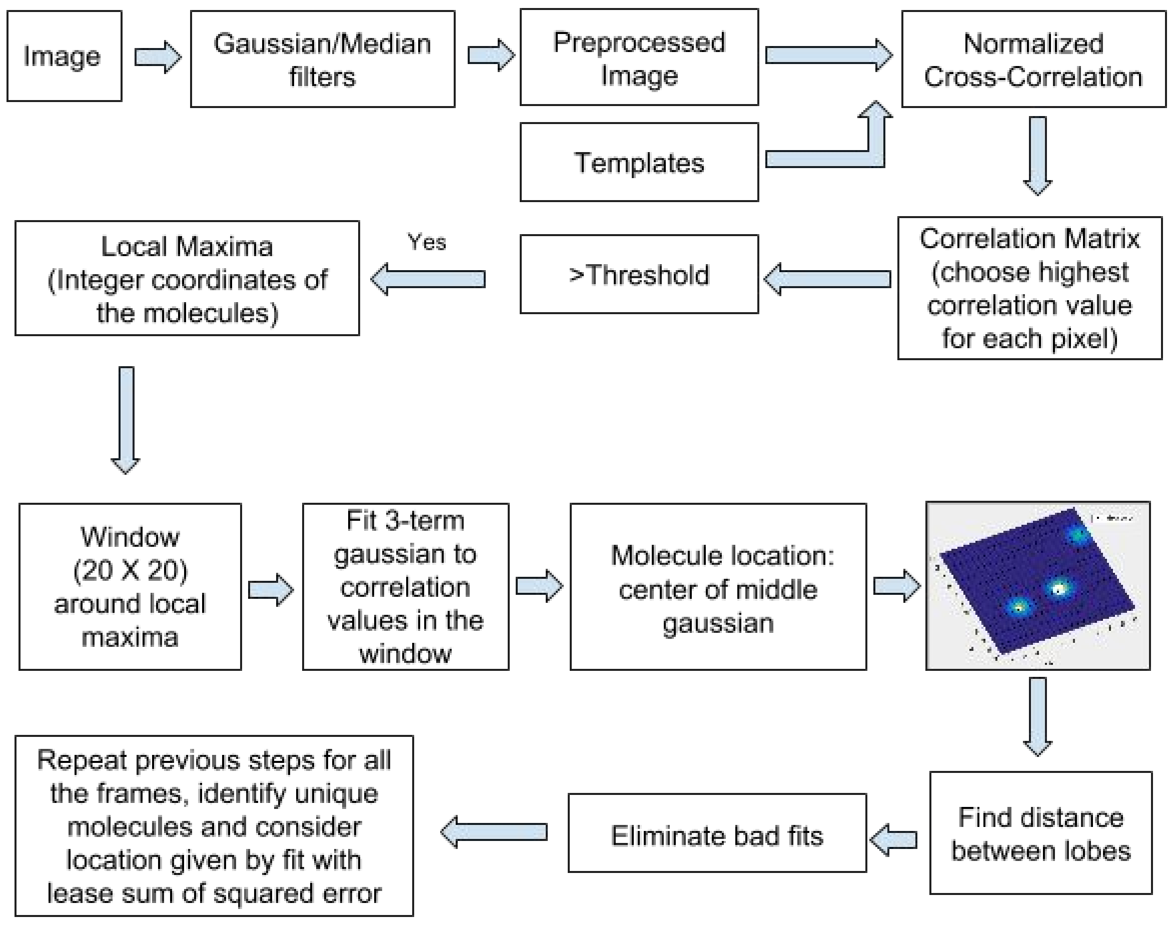
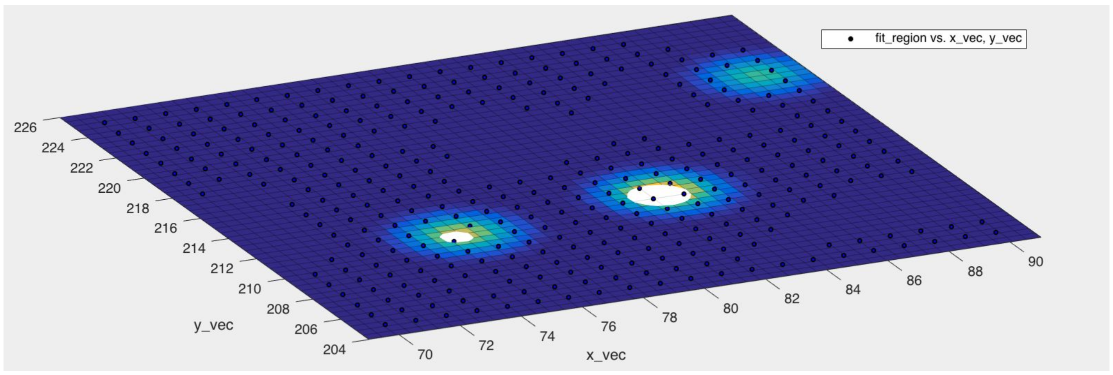


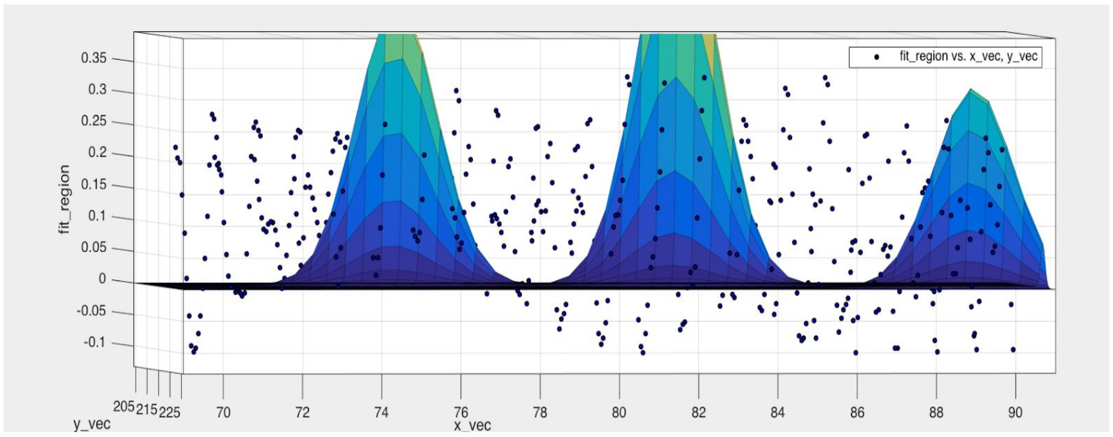
Image
Gaussian/Median filters
Preprocessed Image
Normalized Cross-Correlation
Templates
Correlation Matrix (choose highest correlation value for each pixel)
>Threshold
Yes
Local Maxima (Integer coordinates of the molecules)
Window (20 X 20) around local maxima
Fit 3-term gaussian to correlation values in the window
Molecule location: center of middle gaussian
Find distance between lobes
Eliminate bad fits
Repeat previous steps for all the frames, identify unique molecules and consider location given by fit with lease sum of squared error

Novelty of Approach:

Some of the preliminary ideas have been inspired from the work of Matthew D. Lew and Scott S. Hsieh -Automated, Robust Recognition and Extraction of the Double-Helix Point Spread Function in Fluorescence Microscope Images. However, overall approach is novel and no code has been adapted or referred.
The proposed methodology and the results have been discussed in details in the subsequent sections, but first will try to present novelty of our approach compared to the state of the art for this problem!

We calculated **normalized cross correlation** of preprocessed image with the templates and computed correlation matrix considering best matching template for each pixel. Then we computed **local maxima** in the correlation matrix which are **above a threshold**. This gives location of molecules in terms of integer coordinates, since the highest correlation values will corresponding to the molecule locations. This is because, both the lobes will match with the templates if the template center is at the molecule location. To achieve super localization, we consider 20 X 20 **windows** around thresholded local maxima in the correlation matrix of image with the best matching template for that local maxima location. Now, we fit a **3-term Gaussian model** to the correlation values in this **window**, using nonlinear least square fit. The middle gaussian will have the highest peak and the peak corresponds to the molecule location. The **center of middle gaussian** gives the x,y location of molecule while the index of template gives the z location.The two other gaussians on the sides will have lower peaks and these peaks correspond to the two lobe centers.





Unlike other methods which try to fit 2-term gaussian model to intensity values, we used 3-term gaussian on correlation values instead of intensity. This is because intensity values are likely to have a lot of noise, while correlation values which are measure of similarity between image and template would be better.

We also eliminated bad fits by considering the lobe distance, as expected lobe distance should be between 10 and 20 pixels. We repeated the above steps for all the frames, identified unique molecules by considering the proximity of their location and considered the location given by the fit with least sum of squared error in the final results.

Discussion of proposed methodology:

1. Preprocessing Steps

- **Gaussian filter**
  A Gaussian filter is a filter whose impulse response is a Gaussian function (or an approximation to it). Gaussian filters have the properties of having no overshoot to a step function input while minimizing the rise and fall time. In this case, it is easy for us to find the biomolecules, which have been filtered to Gaussian-like spots.
  F = fspecial('Gaussian',[6,6],0.5);
  This function returns a rotationally symmetric Gaussian low-pass filter of size [6,6] with standard deviation sigma (0.5).
  So the filtered image could be I (*I= imfilter(I, F, 'same');*), which shows a image with several max intensity locations(lobes around biomolecules) and some background noise.

- **Median filter**
  A Median filter is a nonlinear filtering technique, often used to remove noise. Such noise reduction is a typical pre-processing step to improve the results of such as edge detection on an image.
  I = medfilt2(I);
  Median filter preserves edges of the information of the image while removing noise.

2. Template Matching

To match each template with the biomolecules in the image stack, we have following steps to follow:

- **Normalized Cross-correlation**
  For this step, we used normalized cross-correlation as a measure of similarity of two images as a function of the lag of one relative to the other.
  First, the images are normalized. This is done at every step by subtracting the mean and dividing by standard deviation. That is, the cross-correlation of a template t(x,y), with an image f(x,y) is
  where n is the number of pixels in t(x,y) and f(x,y) , $\overline{f}$ is the average of f and

$$\frac{1}{n}\sum_{x,y}\frac{(f(x,y)-\overline{f})(t(x,y)-\overline{t})}{\sigma_f\sigma_t}$$

  σ_f is standard deviation of f.

  We calculated cross-correlation of image with all the templates and computed the correlation matrix with the correlation value corresponding to the best template match for each pixel

- **Computing Local Maxima**

  We calculated local maxima in the correlation matrix and considered maxima above a threshold of 0.6. Then, we constructed 20 X 20 regions across each thresholded maxima and retained just the maxima with highest value in each region.These final maxima locations give the x-y integer locations of the biomolecules. The z- location is given by the index of best matching template for that maxima.

3. Curve Fitting and Super Localization

- **3 term Gaussian fitting**

  The next step is to fit a 3 term gaussian curve,
  f(x,y) = a1.*exp(-((x-b1).^2/2)-((y-c1).^2/2))+
  a2.*exp(-((x-b2).^2/2)-((y-c2).^2/2))+a3.*exp(-((x-b3).^2/2)-((y-c3).^2/2))
  using nonlinear least square fit to the correlation values in each 20 X 20 region around the retained local maxima.
  For this, we used the correlation values in the matrix constructed by correlating image with the template which had the highest correlation for retained local maxima in a particular 20 X 20 region. The initial starting point for the least square fit is given by the location of local maxima and x,y coordinates 7 pixels to either side of the maxima.

- **Eliminating bad fits**

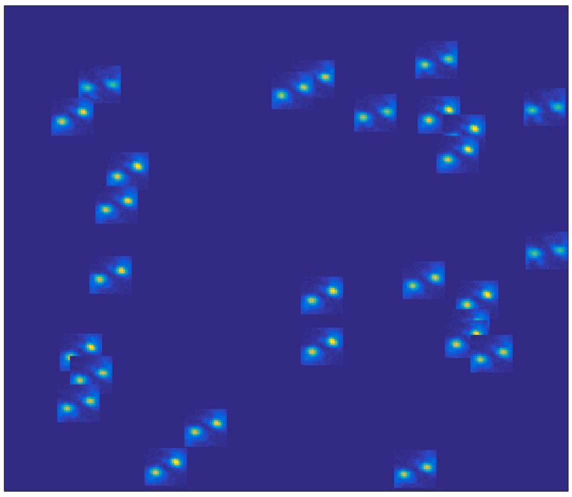
  We eliminated bad fits by considering the distance between centers of the two lower gaussian peaks (which correspond to the two lobes). If the lobe distance was less than 10 pixels or more than 20 pixels, it was considered as a bad fit.

- **Identifying unique molecules across frames**

  The preceding steps were repeated for all the frames and unique molecules were identified by considering the x,y,z location of the molecules. If the x,y locations of two molecules in different frames are within 5 pixels of each other and z (template index) is within 3 units , they are considered as same molecules. The location given by the least sum of squared error fit for a molecule is considered in the final results.

Results:

Reconstructed Image:



Tabular Results:

The table below shows the integral and super localized locations of biomolecules.
x -> Integer x-coordinate of molecule after local maxima computation
y -> Integer x-coordinate of molecule after local maxima computation
x* -> super localized x-coordinate of molecule after gaussian fit
y* -> super localized y-coordinate of molecule after gaussian fit
z -> Index of best matching template
separation -> separation distance between the two lobes
SSE -> sum of squared error in the gaussian fit

x	y	x*	y*	z	separation	SSE
30	205	33.256	208.35	14	16.661	8.9266
40	147	42.074	146.68	16	20.895	8.3818

43	46	46.911	44.161	11	12.791	11.282
46	137	48.266	138.36	17	8.5899	9.2854
55	256	54.802	258.14	11	12.978	10.447
58	176	56.883	175.33	14	13.474	12.946
59	206	56.038	202.83	20	12.73	8.5197
60	33	57.484	33.955	19	8.8059	6.9101
69	218	70.949	217.62	20	9.5954	11.505
80	215	82.278	212.13	20	12.525	7.0926
89	59	89.809	60.058	20	18.821	7.974
107	54	106	55.382	18	15.342	7.86
131	257	130.2	261.49	11	12.294	8.0027
144	51	142.68	59.295	18	16.348	6.4709
147	199	148.2	194.83	17	13.368	8.3696
155	151	155.94	151.11	19	12.453	12.575
157	224	156.73	222.34	20	8.1298	18.006
172	220	174.96	219.91	20	12.177	11.344
178	219	181.39	217.82	20	10.092	5.29
182	151	180.5	150.69	20	16.992	5.8392
185	37	178.85	35.821	20	18.2	9.982
186	231	186.84	230.5	16	16.509	5.4541
197	42	199.22	42.327	16	16.483	5.2266
212	36	206.45	43.16	16	12.728	12.637
225	96	224.62	97.19	18	17.832	15.119
246	77	244.22	79.54	20	15.817	11.924
247	195	251.96	195.85	16	11.396	10.681

Time taken:

The recorded time taken for all images was 69.809451 seconds.

Individual Contributions:

We worked on most of the steps together, however just to highlight some of the individual contributions as required.

Akshat : Overall Approach, Template Matching, Super localization(Gaussian fitting), Report (partly)
Jie: Preprocessing, Identifying unique molecules, Image reconstruction, Report(partly), Experimenting with other ideas

Code:

main.m

```
clc;
clear;
close all;
load('TheTemplates.mat');

fname = 'TheImages.tif';

info = imfinfo(fname);
num_images = numel(info);

F = fspecial('Gaussian',[6,6],0.6 ); %Gaussian filter
% Initialising the local maxima finder object and setting parameters
hLocalMax = vision.LocalMaximaFinder;
hLocalMax.MaximumNumLocalMaxima = 100;
hLocalMax.NeighborhoodSize = [5 5];

% location array contains the x,y location of peaks/local maxima in all the frames
location = zeros(100,6,100);

tic;
```

```
for k = 1:100 % Loop over all the frames

   I = imread(fname, k); % Read image
   I = imfilter(I, F, 'same'); % Image after Gaussian
   I = medfilt2(I); % Image after median filter
   [m,n] = size(I);
   % Initialise matrix to find maximum correlation between image and template for
each frame
   maxCor = ones(m+20-1,n+20-1)*(-1);

   % Initalise matrix to store index of best matching template for each frame
   maxTemplate = zeros(m+20-1,n+20-1);

   for i = 1:20  % Loop over all the templates
      T_i = template(i,:,:);
      T = squeeze(T_i); % Get the individual template matrix

      I_T = normxcorr2(T,I); % Perform normalized cross correlation of template with
image

      prevMax = maxCor;

      % Updating the maximum correlation matrix if correlation between
      % current template and image is better than previous correlation
      maxCor = max(maxCor,I_T);

      % Saving the index of best matching template till now for every pixel
      nz = find(maxCor - prevMax);
      maxTemplate(nz) = i;

   end

   % Choosing threshold to find local maxima peaks which are above
   % threshold
   hLocalMax.Threshold = 0.6;

   % Calculating local maxima in the correlation matrix
   loc = step(hLocalMax, maxCor);
   loc = double(loc);
   loc(loc(:,1)>258 | loc(:,2)>264 | loc(:,1) < 21 | loc(:,2) < 21  ,:) = [];
   [row,col] = size(loc);

   % We only want one local maxima in each window of correlation matrix,
   % so retaining the highest maxima in each window and removing others.
```

```
    sorted_loc = sortrows(loc,[1 2]);
    init_sorted = sorted_loc;
    for p = 1:row-1

        if (abs(sorted_loc(p,1)-sorted_loc(p+1,1)) < 10) &&
(abs(sorted_loc(p,2)-sorted_loc(p+1,2)) < 10)
            if maxCor(sorted_loc(p,1),sorted_loc(p,2)) >
maxCor(sorted_loc(p+1,1),sorted_loc(p+1,2))
                sorted_loc(p+1,1) = sorted_loc(p,1);
                sorted_loc(p+1,2) = sorted_loc(p,2);
                sorted_loc(p,1) = 0;
                sorted_loc(p,2) = 0;
            else
                sorted_loc(p,1) = 0;
                sorted_loc(p,2) = 0;
            end

        end
    end
    sorted_loc( ~any(sorted_loc,2), : ) = [];
    [row,col] = size(sorted_loc);

    % Creating windows around the retained local maxima
    width = 10;
    window = (2*width+1)^2;

    % Fitting a triple gaussian to correlation values in the window around each
retained local maxima
    % Center of middle gaussian gives the molecule location while center of
    % the other two gaussians give the lobe center location
    for p = 1:row
        x = sorted_loc(p,1);
        y = sorted_loc(p,2);
        T_i = template(maxTemplate(x,y),:,:);
        T = squeeze(T_i);
        I_T = normxcorr2(T,I);
        x_vec=reshape(repmat((x-width:x+width)',1,2*width+1),window,1);
        y_vec=reshape(repmat((y-width:y+width)',1,2*width+1)',window,1);
        region = maxCor(x-10:x+10,y-10:y+10);
        fit_region = reshape(region,window,1);
        [fitresult, gof] =
createFit(double(x_vec),double(y_vec),double(fit_region),double(x),double(y));
        x_center = fitresult.b2;
        y_center = fitresult.c2;
        lobe_distance = sqrt((fitresult.b1-fitresult.b3)^2+(fitresult.c1-fitresult.c3)^2);
        location(p,1,k) = x;
        location(p,2,k) = y;
        location(p,3,k) = x_center;
        location(p,4,k) = y_center;
        location(p,5,k) = maxTemplate(x,y);
        location(p,6,k) = lobe_distance;
        location(p,7,k) =  k;
        location(p,8,k) = gof.sse;

    end
end

C = permute(location,[1 3 2]);
C = reshape(C,[],size(location,2),1);
C( ~any(C,2), : ) = [];
sortedC = sortrows(C,[1 2 5]);

% Eliminating bad fits based on lobe distance
sortedC(sortedC(:,5) < 10 | sortedC(:,5) > 20 ,:) = [];
[row,col] = size(sortedC);

% Idemtifying unique mocules across frames and retaining just one location
% of molecule (the location given by lowest sum of squared error fit)
for p = 1:row-1
    if (abs(sortedC(p,1)-sortedC(p+1,1)) <= 5) && (abs(sortedC(p,2)-sortedC(p+1,2)) <=
5) && (abs(sortedC(p,5)-sortedC(p+1,5)) <= 5 )
          if sortedC(p,8) > sortedC(p+1,8)
              sortedC(p,:) = 0;
          else
              sortedC(p+1,:) = sortedC(p,:);
              sortedC(p,:) = 0;
          end
    end
end

sortedC( ~any(sortedC,2), : ) = [];
[row,col] = size(sortedC);


toc;
```

```
% Reconstructing Image
temp = ones(258,264);
for i = 1:row
   x=sortedC(i,1);
   y=sortedC(i,2);
   temp((x-10):(x+9),(y-10):(y+9)) = template(sortedC(i,5),:,:);
end
imagesc(temp)
set(gca,'XTick',[])
set(gca,'YTick',[])
saveas(gcf,'reconstructed','png')
```

createFit.m

```
function [fitresult, gof] = createFit(x, y, z , c_x , c_y)
%CREATEFIT(X,Y,Z)
%  Create a fit.
%
%  Data for 'untitled fit 1' fit:
%      X Input : x
%      Y Input : y
%      Z Output: z
%  Output:
%      fitresult : a fit object representing the fit.
%      gof : structure with goodness-of fit info.
%
%  See also FIT, CFIT, SFIT.



%% Fit: 'untitled fit 1'.
[xData, yData, zData] = prepareSurfaceData( x, y, z );

% Set up fittype and options.
ft = fittype( '(a1.*exp(-((x-b1).^2/2)-((y-c1).^2/2))) +
(a2.*exp(-((x-b2).^2/2)-((y-c2).^2/2)))+(a3.*exp(-((x-b3).^2/2)-((y-c3).^2/2)))',
'independent', {'x', 'y'}, 'dependent', 'z' );
opts = fitoptions( 'Method', 'NonlinearLeastSquares' );
opts.Display = 'Off';
opts.StartPoint = [0.882837605583052 0.913711681293018 0.558284923622275
c_x-7 c_x c_x-7 c_y-7 c_y c_y+7];

% Fit model to data.
```

```
[fitresult, gof] = fit( [xData, yData], zData, ft, opts );
```

References:

[1] Matthew D. Lew and Scott S. Hsieh, Department of Electrical Engineering, Stanford University - Automated, Robust Recognition and Extraction of the Double Helix Point Spread Function in Fluorescence Microscope Images

[2] www.wikipedia.com