EE492 Senior Design II - Weekly Report 5

Group Number: May1634	Date: 2/11/16 - 2/18/16
Project Name: Studying cell behaviors in 3D microtissues using a LabChip	
Advisor: Long Que	
Client: Long Que	

The team

Role	Group Member
Group leader	Jonathan Yatckoske
Team Webmaster	Yaxiong Zhang
	Chun-Hao Lo
Team Communication Leader	Yuqian Hu
Team Key Concept Holder	Kaiyu Xu

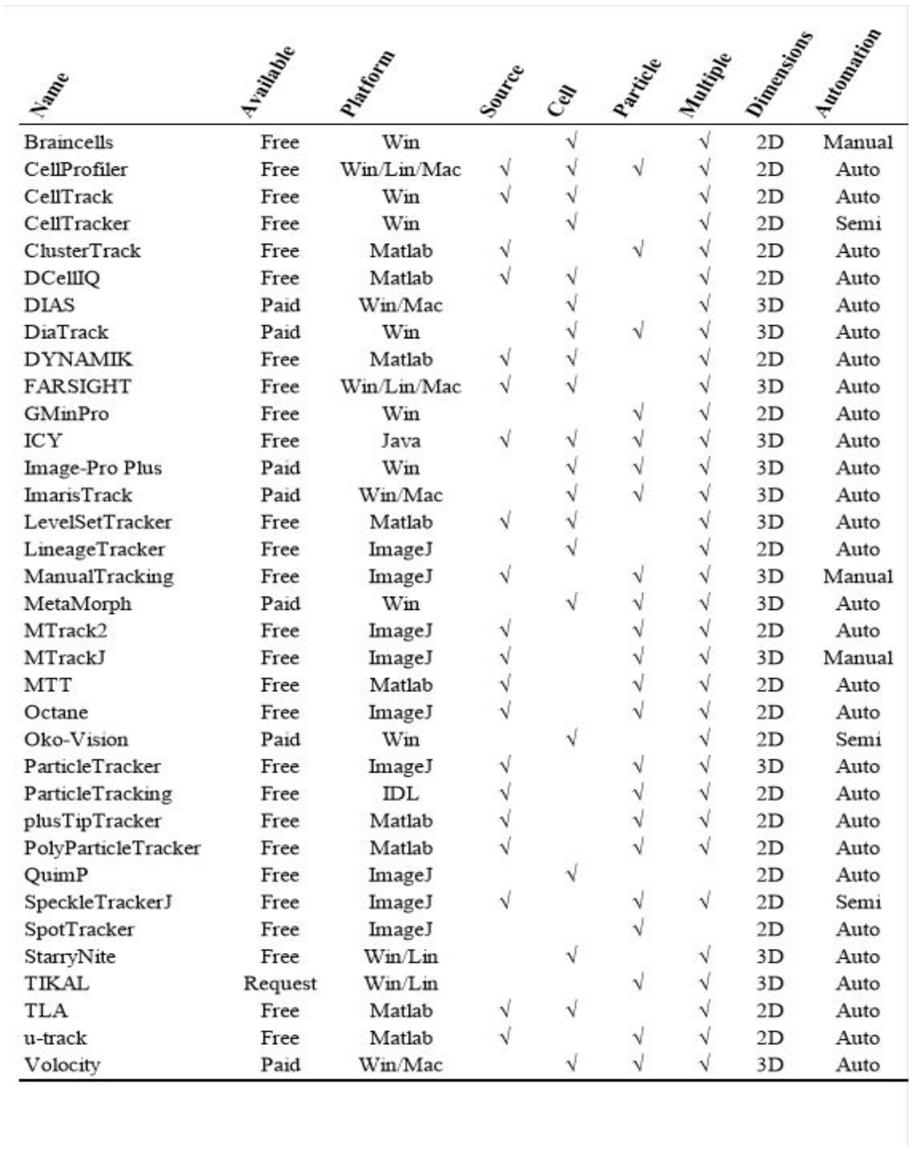
Attendance (meeting date: Feb. 19th 2016)

Jonathan Yatckoske	In person
Chun-Hao Lo	In person
Yaxiong Zhang	In person
Kaiyu Xu	In person
Yuqian Hu	In person

Accomplishments of past week

1. We tried to get the 3D plot from the lab however it seems that all the data we had in 3D is fixed in one particular z-plane. The confocal method we came up with might not work so we might need to work on something else.

2. Coming up with skeleton code for 3D plotting.
Plan A：
Work on some software that could give 3D plot for cell movement. Below are some possible approaches.



Name	Available	Platform	Source	Cell	Particle	Multiple	Dimensions	Automation
Braincells	Free	Win		√		√	2D	Manual
CellProfiler	Free	Win/Lin/Mac	√	√	√	√	2D	Auto
CellTrack	Free	Win	√	√		√	2D	Auto
CellTracker	Free	Win		√		√	2D	Semi
ClusterTrack	Free	Matlab	√		√	√	2D	Auto
DCellIQ	Free	Matlab	√	√		√	2D	Auto
DIAS	Paid	Win/Mac		√		√	3D	Auto
DiaTrack	Paid	Win		√	√	√	3D	Auto
DYNAMIK	Free	Matlab	√	√		√	2D	Auto
FARSIGHT	Free	Win/Lin/Mac	√	√		√	3D	Auto
GMinPro	Free	Win			√	√	2D	Auto
ICY	Free	Java	√	√	√	√	3D	Auto
Image-Pro Plus	Paid	Win		√	√	√	3D	Auto
ImarisTrack	Paid	Win/Mac		√	√	√	3D	Auto
LevelSetTracker	Free	Matlab	√	√		√	3D	Auto
LineageTracker	Free	ImageJ		√		√	2D	Auto
ManualTracking	Free	ImageJ	√		√	√	3D	Manual
MetaMorph	Paid	Win		√	√	√	3D	Auto
MTrack2	Free	ImageJ	√		√	√	2D	Auto
MTrackJ	Free	ImageJ	√		√	√	3D	Manual
MTT	Free	Matlab	√		√	√	2D	Auto
Octane	Free	ImageJ	√		√	√	2D	Auto
Oko-Vision	Paid	Win		√		√	2D	Semi
ParticleTracker	Free	ImageJ	√		√	√	3D	Auto
ParticleTracking	Free	IDL	√		√	√	2D	Auto
plusTipTracker	Free	Matlab	√		√	√	2D	Auto
PolyParticleTracker	Free	Matlab	√		√	√	2D	Auto
QuimP	Free	ImageJ		√			2D	Auto
SpeckleTrackerJ	Free	ImageJ	√		√	√	2D	Semi
SpotTracker	Free	ImageJ			√		2D	Auto
StarryNite	Free	Win/Lin		√		√	3D	Auto
TIKAL	Request	Win/Lin			√	√	3D	Auto
TLA	Free	Matlab	√	√		√	2D	Auto
u-track	Free	Matlab	√		√	√	2D	Auto
Volocity	Paid	Win/Mac		√	√	√	3D	Auto

As we want to simplify our work we are considering functions in Matlab that works both for tracking cell and plotting in 3D. In this way, function LevelSetTracker might be the way to go.

Plan B:
We find some previous work on cells tracking and measurement by using spatiotemporal images analysis. The essence of the following code might also help us get what we want.

```
%-------------------------------------------------------------------
-----
% This is a simulation demo to show multiple trackpaths for tracking the
% blood cell motion. In this demo, the diameter of the blood cell is 16
% pixels, so we use 9 trackpaths(in row 23) and the distance between each
% trackpath is 8 pixels(in row 25).
% Written by Yuan Chen,Nanjing university of aeronautic and
astronautic(2010)
%----------------------------------------------------------------------
-----
clear all;
%% read the avi file, and calculate the mean image;
display('Reading and processing demo3.avi file...')
video=aviread('demo3.avi');          % read the avi file;
video = {video.cdata};               % exracte the avi data(matrix);
z = 0;
for i=1:length(video);               % change 'color' video to gray;
   video{i}=rgb2gray(video{i});
   z = z + double(video{i});
end
z = z/length(video);                 % the mean image;
[mz,nz] = size(z);
[mu1,mu2,v1x,v1y,v2x,v2y] = eigfunction(z,7,5);  % calculate the eigen
values and eigen vectors of the mean image;

%% the trackpaths generation;
N = 9;                              % THE number of trackpath can be set
within [5,65];NOTE that,larger N will consume
                                    % much more processing times,we
suggest that N = 9, and no larger than 16;
t = 64/(N-1);                       % the distance between each
trakcpath;
T = 64:t:128;                       % T are the radius of the
trackpaths(the center(148,150)in the mean image z);
for k = 1:length(T)
   tmp = zeros(size(z));
   for i = 1:138
      for j = 1:300
         if sqrt((148-i)^2 + (150-j)^2 ) < T(k)+1 && sqrt((148-i)^2 +
(150-j)^2 ) >= T(k);
            tmp(i,j) = 255;
         end
      end
   end
   tmp = im2bw(tmp);
   tmp = bwmorph(tmp,'thin',inf);
   trackpath{k} = tmp;
end
display([num2str(N),' trackpaths are selected to generate ',num2str(N),'
ST images...Processing them may take several minutes'])
```

```
%% ST image generation, processing and traces extraction;
for i=1:size(trackpath,2)                              % the number of
the generated trackpath;
   map = [];
   [xx,yy] = startpoint(trackpath{i});                   % finding the
starting point of a trackpath;
   ind_trackpath = ord_line_indx(trackpath{i},xx,yy);     % order the
trackpath points;
   for j = 1:size(video,2)                             % ST image
generation;
       line = video{j}(ind_trackpath);
       map = [map;line];
   end
   map = map';
   Map{i} = map;                                      % store the ST
image;
   display(['processing ST image',int2str(i)])
   [F,Theta,angle] = J4(map,2.5);                       % raw ST image
enhanced by Jacob filter;
   R = nsf(F);                                        % applying noise
supression function;
   K = angfilter(R,Theta,angle,20);                      % applying
orientation filter function;
   th = graythresh(K);                                 % threshold for
cell tracked trackpath;
   if th <= 0.01;                                     % threshold for none
cell tracked trackpath;
       th = 0.01;
   end
   bw = im2bw(K,th);
   thin = bwmorph(bw,'thin',inf);                        % thinning
extracted traces;
   trace{i} = bwareaopen(thin,10);                       % remove noise
trace if its length smaller than 10 pixels;
end

%% calculate the trackpaths coordinate and grayscale;
h = fspecial('gaussian',7);
for i = 1:length(trackpath)                            % denoise;
   Map_denoise{i} = conv2(Map{i},h,'same');
   Map_denoise{i} = conv2(Map_denoise{i},h,'same');
end
% -------find out the coordinate(X_trackpath,Y_trackpath) and grayscale
of the trackpaths;
```

```
for i = 1:length(trackpath)
    [xx,yy] = startpoint(trackpath{i});                     % find the start
coordinate of a trackpath;
    ind_trackpath = ord_line_indx(trackpath{i},xx,yy);     % search the
trackpath's point from the start point orderly;
    for j = 1:length(ind_trackpath)
        [X_trackpath(j),Y_trackpath(j)] =
ind2sub([mz,nz],ind_trackpath(j)); % change the index to (X,Y)
coordinate;
    end
    Trackpath_value{i}(:,1) = X_trackpath';
    Trackpath_value{i}(:,2) = Y_trackpath';
    Trackpath_value{i}(:,3) = 0;
    [Y_trace,X_trace] = find(trace{i}==1);               % find
extracted trace coordinate;
    for k = 1:length(Y_trace)                              % store the
grayscale of the trace;
        Trackpath_value{i}(Y_trace(k),3) =
Map_denoise{i}(Y_trace(k),X_trace(k));
    end
    clear X_trackpath;clear Y_trackpath;
end
% -------backward mapping the extracted traces to the trackpaths;
for i = 1:length(Trackpath_value)
    for j = 1:size(Trackpath_value{i},1)
        z(Trackpath_value{i}(j,1),Trackpath_value{i}(j,2)) =
Trackpath_value{i}(j,3);
    end
end
% figure,imshow(z,[]),title('The tracking state of trackpaths')

%% trackpaths alignment
k = round(length(T)/2);                                    % the centerline
trackpath;
% -------turn all trackpaths and all ST images to the same length(or size)
as the centreline trackpath and centreline ST image
% according to the eigen victor v1x,v1y;
for i = 1:length(T)
    p = findpoints(trackpath{k},trackpath{i},v1x,v1y);     % the
corresponding relationship between the k and the i trackpaths.
    P{i} = round(smooth(p));
    if i~=k
        for j = 1:length(P{i})
            trace_aligned{i}(j,:) = trace{i}(P{i}(j),:);
            Points_aligned{i}(j,:) = Trackpath_value{i}(P{i}(j),:);
        end
    else
        trace_aligned{i} = trace{i};
        Points_aligned{i} = Trackpath_value{i};
    end
end

% -------base trace generation;
B_trace = 0;
for i = 1:length(trace_aligned)
    B_trace = B_trace + trace_aligned{i};
end
% figure,imshow(B_trace),title('the result of traces mapping')
B_trace = im2bw(B_trace);
B_trace = bwmorph(B_trace,'dilate');
B_trace = bwmorph(B_trace,'thin',inf);
% figure,imshow(B_trace),title('Base-trace')

%%  fusion of aligned points in trackpaths to calculate the trajectory;
l = 1;
for j = 1:size(Points_aligned{1},1)           % the number of the aligned
points in the trackpath;
    Fusion_points = [];
    for i = 1:size(Points_aligned,2)           % the number of the
trackpaths;
        if Points_aligned{i}(j,3) > 0          % >0 means the cell is
tracked;
            X = Points_aligned{i}(j,1);        % X coordinate;
            Y = Points_aligned{i}(j,2);        % Y coordinate;
            V = Points_aligned{i}(j,3);        % grayvalue of (X,Y);
            Fusion_points = [Fusion_points,[X;Y;V]];
        end
    end
    if size(Fusion_points,2) > 2               % the cell is tracked by
three or more than three trackpths;
        X_fused(l,1) = mean(Fusion_points(1,:));% the fused position is
the center among the aligned points;
        Y_fused(l,1) = mean(Fusion_points(2,:));
        l = l+1;
    elseif size(Fusion_points,2) == 2          % the cell is tracked by
two trackpaths(Eq.(15)in the paper);
        x1 = Fusion_points(1,1); x2 = Fusion_points(1,2);
        y1 = Fusion_points(2,1); y2 = Fusion_points(2,2);
        v1 = Fusion_points(3,1); v2 = Fusion_points(3,2);
        [X_fused(l,1),Y_fused(l,1)] = fusion2p(x1,y1,x2,y2,v1,v2,0.4); %
the fused position calculation;
        l = l+1;
    elseif size(Fusion_points,2) ==1          % the cell is tracked by
only one trackpath;
        X_fused(l,1) = Fusion_points(1,1);     % no fusion is needed;
        Y_fused(l,1) = Fusion_points(2,1);
        l = l+1;
    end
end

   X_trajectory = round(X_fused);             % calculated trajectory;
   Y_trajectory = round(Y_fused);
   X_smooth = round(smooth(X_fused,10));       % smooth process to
prevent jagged trajectory;
   Y_smooth = round(smooth(Y_fused,10));
for i = 1:length(X_fused)
    z(X_trajectory(i),Y_trajectory(i)) = 255;
end
figure,imshow(z,[]);title(['Blood cell tracking by ',num2str(N), '
trackpaths(black line)'])
hold on;plot(Y_smooth,X_smooth,'r')

% -------the real trajectory of the cell;
x = 76:228;
y = 124*sin(pi/162*x+80.32);
y = round(-y+151);
hold on,plot(x,y);
h = legend('Tracked trajectory','Real trajectory',2);

%% calculating the error and variance;
for i = 1:length(X_trajectory)               % by finding the closest
points in the real trajectory to calculate the error;
    for j = 1:length(x)
        dis(j) = sqrt((X_trajectory(i) - y(j))^2 + (Y_trajectory(i) -
x(j))^2);
        Error(i) = min(dis);
    end
end
ES = [mean(Error),std2(Error)];              % mean error and standard
deviation;
display(['The Error and Variance of the tracked tracjectory by using
',num2str(N),' trackpaths are'])
```

```
display([num2str(ES(1)) ' pixels and ' num2str(ES(2)),' pixels'])
```

3. Prepare the presentation about our plans for 3D plotting.

Plan for coming week

1. Group meeting. Discuss what we had for 3D plotting code.
3. Give a brief presentation to advisor about what we have so far.

Pending issues

Perfect the code for 3D plotting.

Individual contributions

Jonathan Yatckoske	work on 3D plot coding
Chun-Hao Lo	website maintenance; work on coding
Yaxiong Zhang	website maintenance; lab work
Kaiyu Xu	Take down meeting notes
Yuqian Hu	work on weekly report; find thesis related paper

Individual hourly contributions

Name	Week Hours	Cumulative Hours
Jonathan Yatckoske	3.5	48.5
Chun-Hao Lo	3.5	41.5
Yaxiong Zhang	2.5	42.5
Kaiyu Xu	2	27
Yuqian Hu	2.5	39