Project 6

Shape Representation

Md Zobaer Islam
PhD student, Electrical Engineering
Oklahoma State University
CWID: A20270547

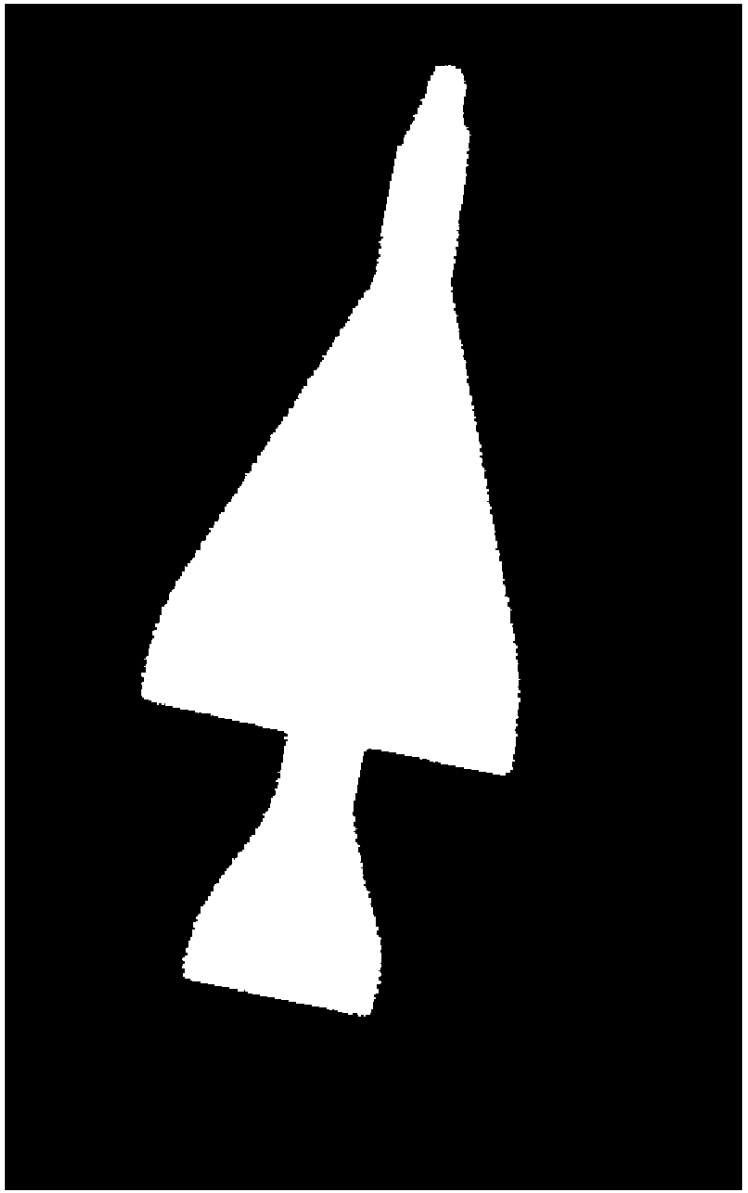
Part I

MATLAB code for boundary extraction and alignment

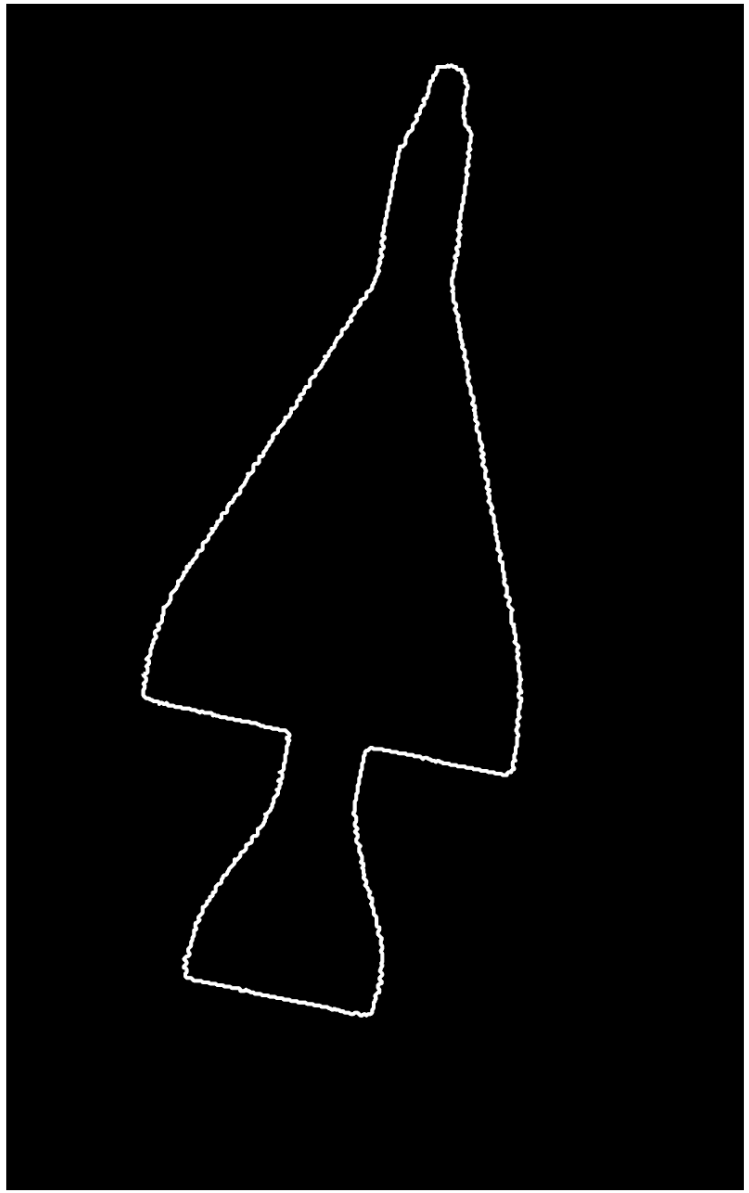
```
clc; clear; close all;f = 18;
I = imread('airplane.bmp'); s = size(I);
B = bwboundaries(I);  Bm = cell2mat(B);
figure(1);imshow(I);title('Original binary Image', fontsize=f);
boundary_im = zeros(s(1),s(2));
figure(2);imshow(boundary_im);hold on;
plot(Bm(:,2),Bm(:,1),'w',LineWidth=2);title('Part I.1(a). Boundary (extracted through bwboundaries())',fontsize=f);
x = Bm(:,2); y = Bm(:,1);
xm = mean(x); ym = mean(y);
cm = [xm, ym]; %centroid
c = [x, y];
%save('c.mat','c'); %uncomment this line to save data
cc = c - cm; %divergence
Cx = cov(cc); %covariance
[V, Cy] = eig(Cx);
A = V';
rot = [-1 0; 0 1]; %rotation matrix to make the rotation anti-clockwise
Arot = rot*A;
Y = (A * cc')';
Y = Y+cm; %offset
figure(3);imshow(boundary_im);hold on;
plot(Y(:,1),Y(:,2),'w',LineWidth=2);title('Part I.1(b). Vertically aligned boundary (clockwise rotation)',fontsize=f);
y2 = (Arot * cc')';
y2 = y2+cm; %offset
figure(4);imshow(boundary_im);hold on;
plot(y2(:,1),y2(:,2),'w',LineWidth=2);title('Part I.1(b). Vertically aligned boundary (anti-clockwise rotation)',fontsize=f);
%save('y2.mat','y2'); %uncomment this line to save data
```

Shape Boundary Extraction

Original binary Image

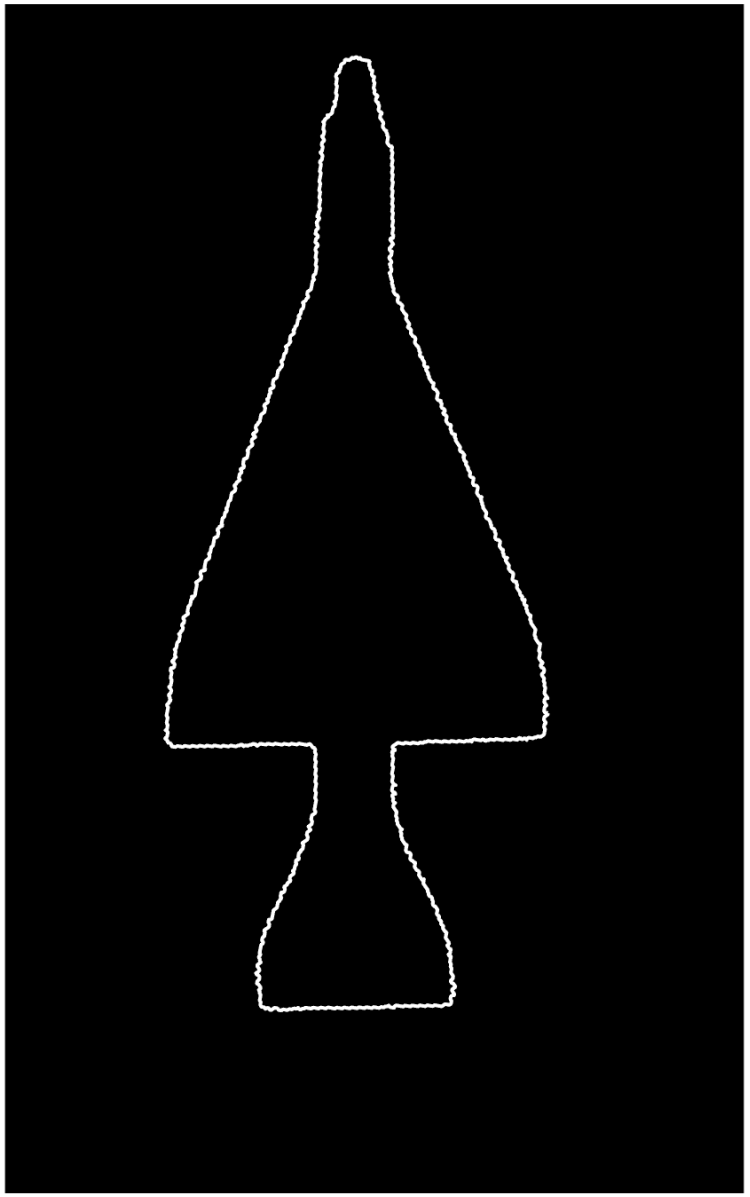


Part I.1(a). Boundary (extracted through bwboundaries())

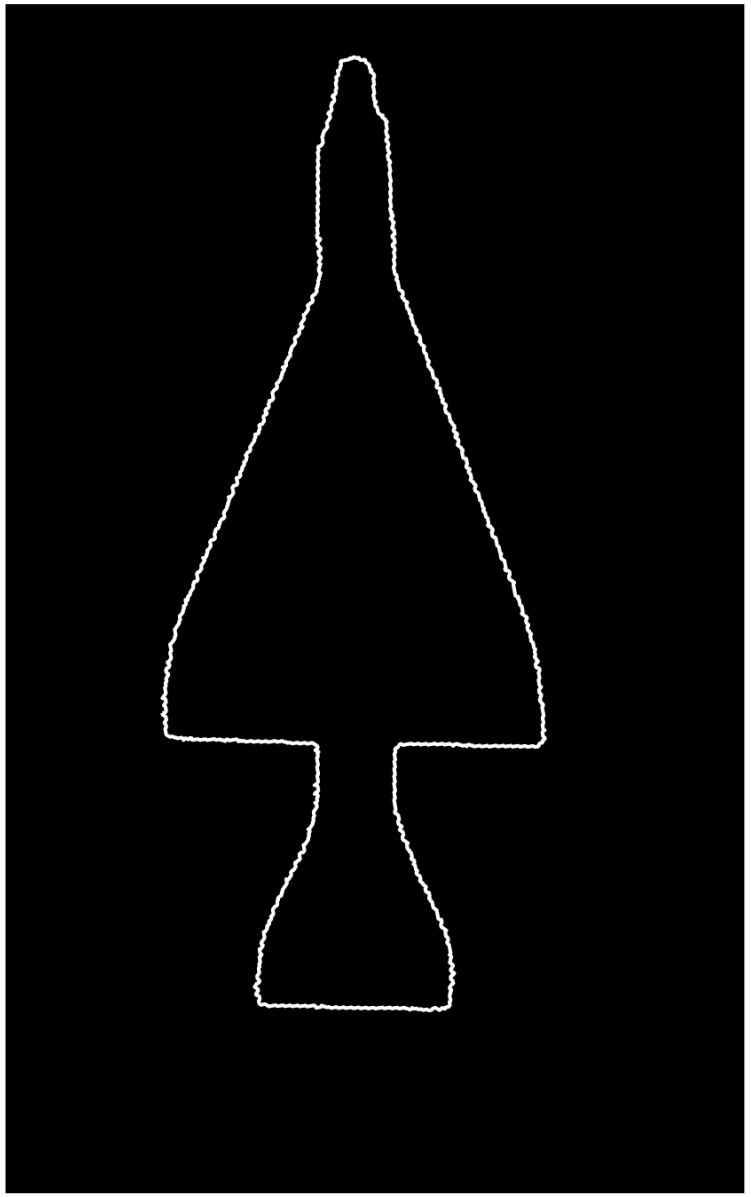


Hotelling Transform for shape alignment

Part I.1(b). Vertically aligned boundary (clockwise rotation)



Part I.1(b). Vertically aligned boundary (anti-clockwise rotation)



This is the required alignment

Discussion

1. More updated MATLAB function bwboundaries() with only the image as parameter was used to extract the boundary, instead of bdtraceboundary(), that requires several parameters.

2. After Hoteling transform, the rotation became clock-wise, thus although it was aligned vertically with higher variance in the vertical direction, it looked flipped when compared to the expected result in the project manual.

3. An additional rotation matrix rot = [-1 0; 0 1] was applied to make the rotation counter-clockwise.

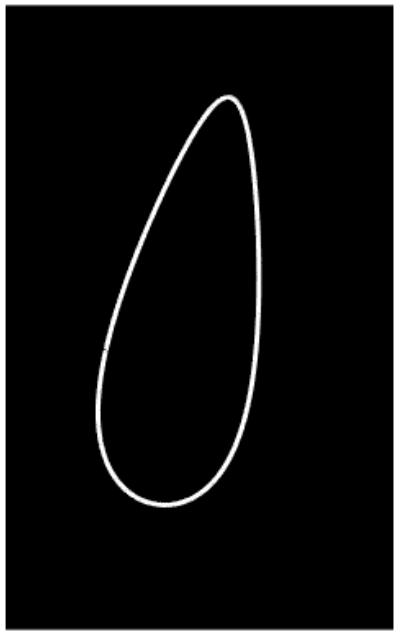
4. Co-ordinate data of original shape and aligned shape are saved as .mat files for further reuse in other .m files.

MATLAB code for Fourier Descriptor (with different orders P)

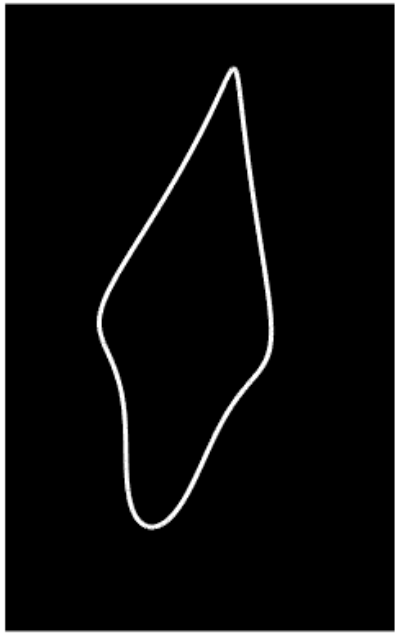
```
clc; clear; close all; f=20;
I = imread('airplane.bmp'); s = size(I);
y = importdata('y2.mat');
P = [2 5 10 15 20 30 40 45 50 60 75 85 100 150 200];
yc = y(:,1) + 1i*y(:,2);
fd = fft(yc);
for k = 1:length(P)
    fs = fd;
    fs(P(k)+1:length(fs)-P(k)) = 0;
    Sc = ifft(fs);
    Z = zeros(size(y));
    Z(:,1) = real(Sc); Z(:,2) = imag(Sc);
    boundary_im = zeros(s(1),s(2));
    if k<=5
        figure(1); subplot(1,5,k);
    end
    if k>=6 && k<=10
        figure(2); subplot(1,5,k-5);
    end
    if k>=11 && k<=15
        figure(3); subplot(1,5,k-10);
    end
    imshow(boundary_im);hold on;
    plot(Z(:,1),Z(:,2),'w',LineWidth=2);
    title(sprintf('P = %d',P(k)),FontSize=f);
end
```

Shape boundaries with Fourier descriptors
(with different orders P)

P = 2



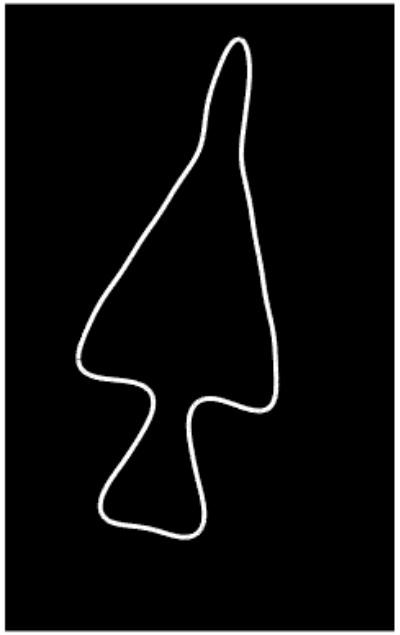
P = 5



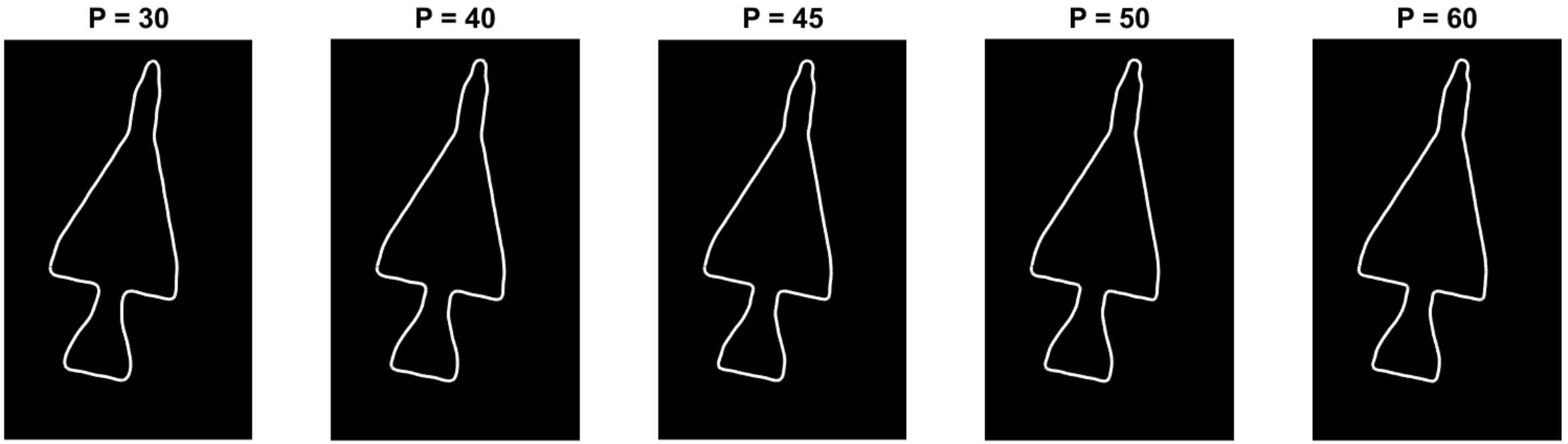
P = 10

P = 15

P = 20



Shape boundaries with Fourier descriptors
(with different orders P)



Shape boundaries with Fourier descriptors
(with different orders P)

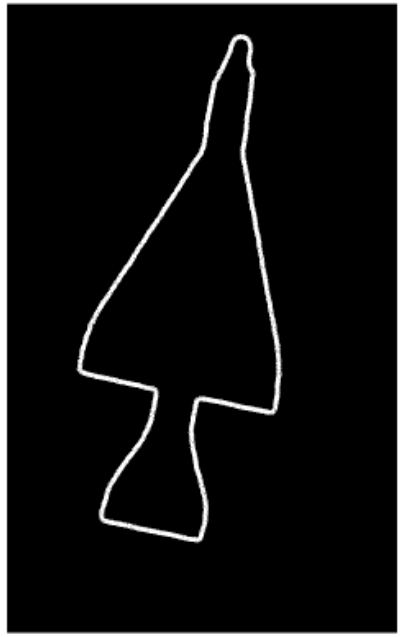
P = 75

P = 85

P = 100

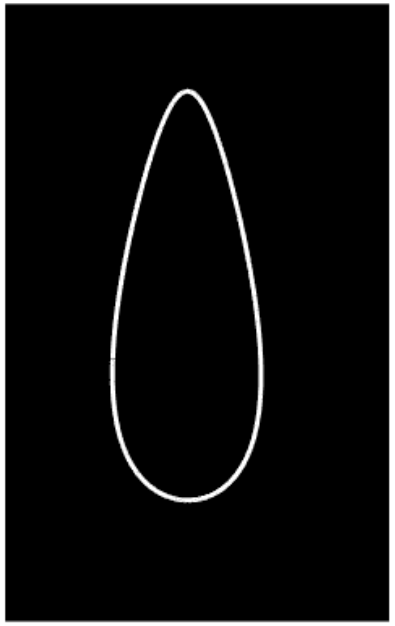
P = 150

P = 200

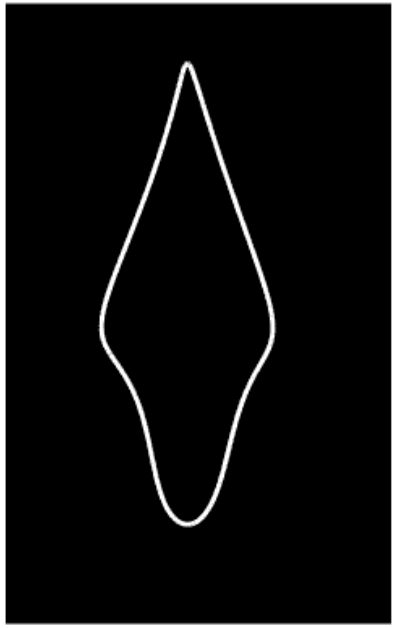


Shape boundaries with Fourier descriptors
(for aligned shape with different orders P)

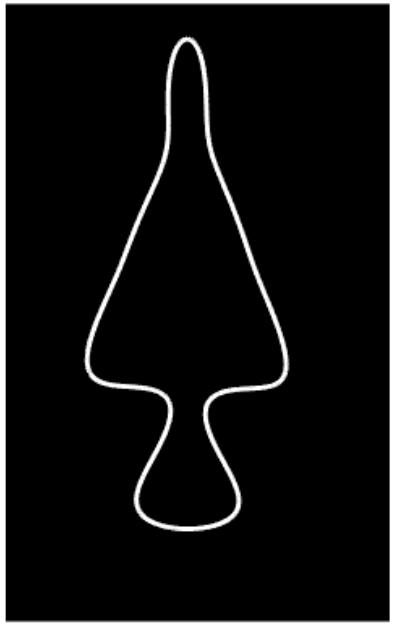
P = 2



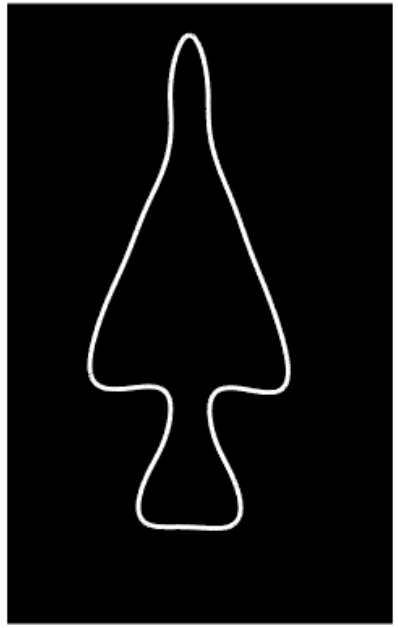
P = 5



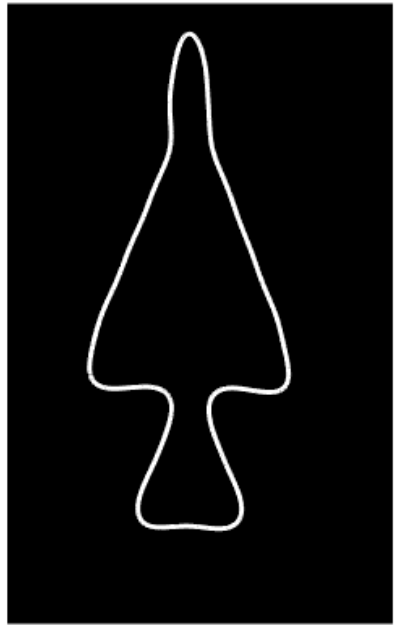
P = 10



P = 15

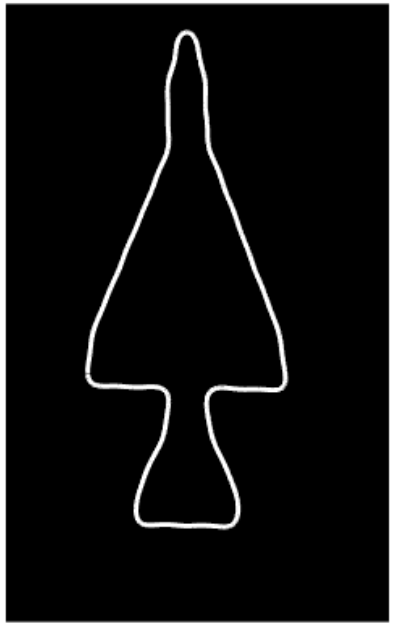


P = 20

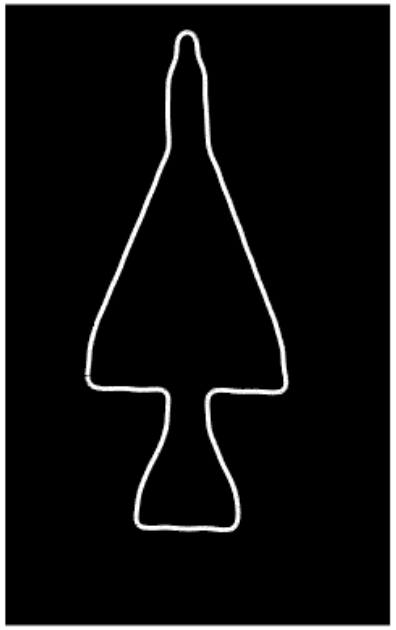


Shape boundaries with Fourier descriptors
(for aligned shape different orders P)

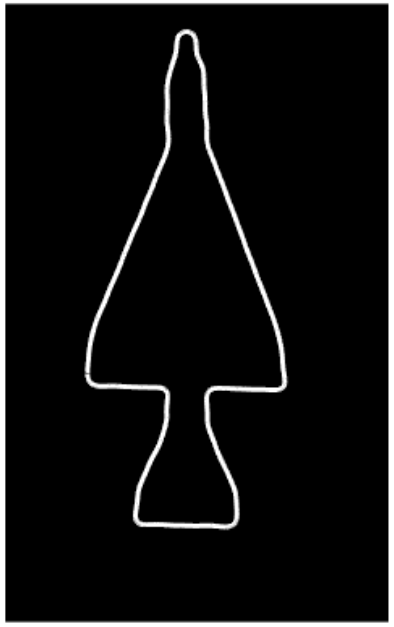
P = 30



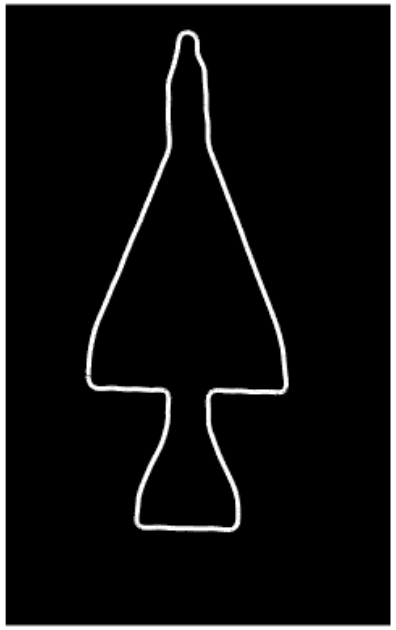
P = 40



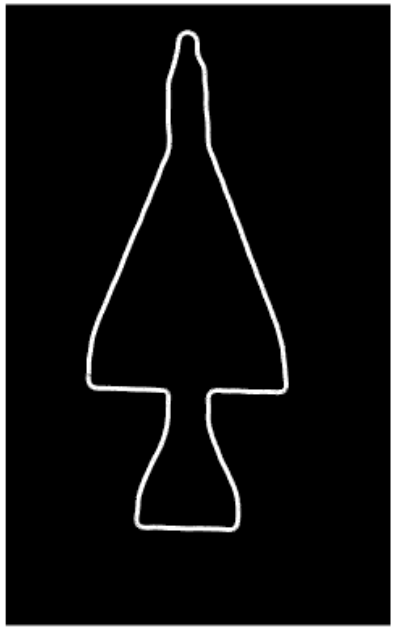
P = 45



P = 50

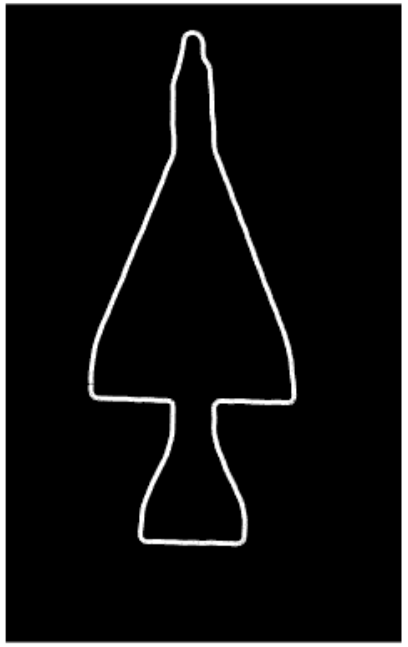


P = 60

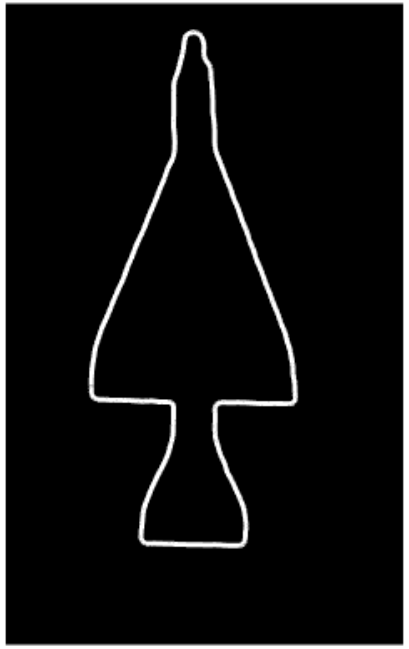


Shape boundaries with Fourier descriptors
(for aligned shape different orders P)

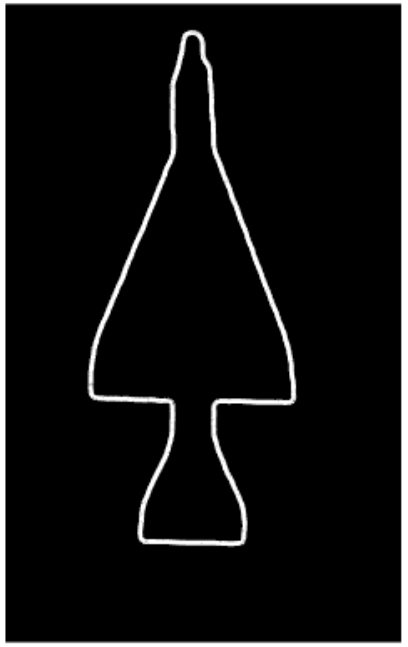
P = 75



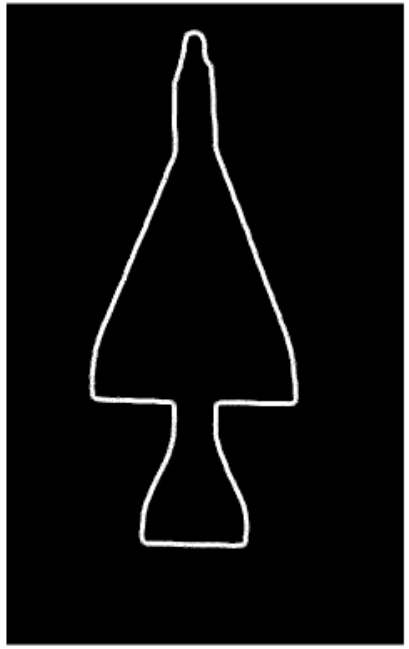
P = 85



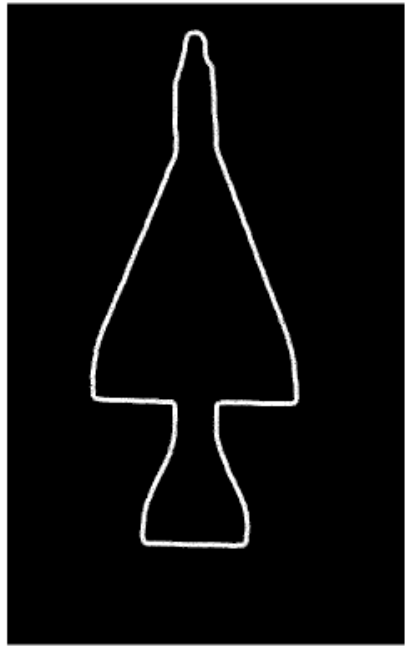
P = 100



P = 150



P = 200



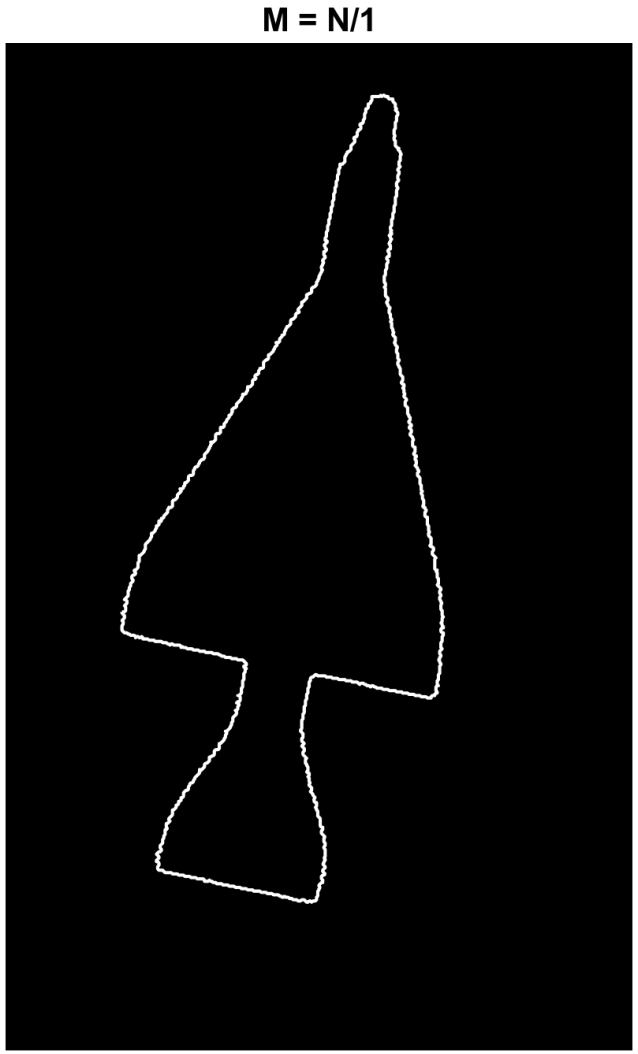
Discussion

1. Based on observation, P=50 gives sufficiently good representation of the shape and for P>50, there was no significant update in the shape.

2. The original number of DFT co-efficients were 1727, but only P=50, which is way less than 1727, is giving fairly good representation of the shape. This provides two benefits:
   a) Provides with a simpler parametric representation. Shape storage and computation on it become more efficient.
   b) More importantly, the lower dimensional representation makes the distance metric more intuitive and meaningful, making the comparison between shapes easier. This has application in image classification, pattern recognition and object detection from images.

MATLAB code for downsampled Fourier Descriptor

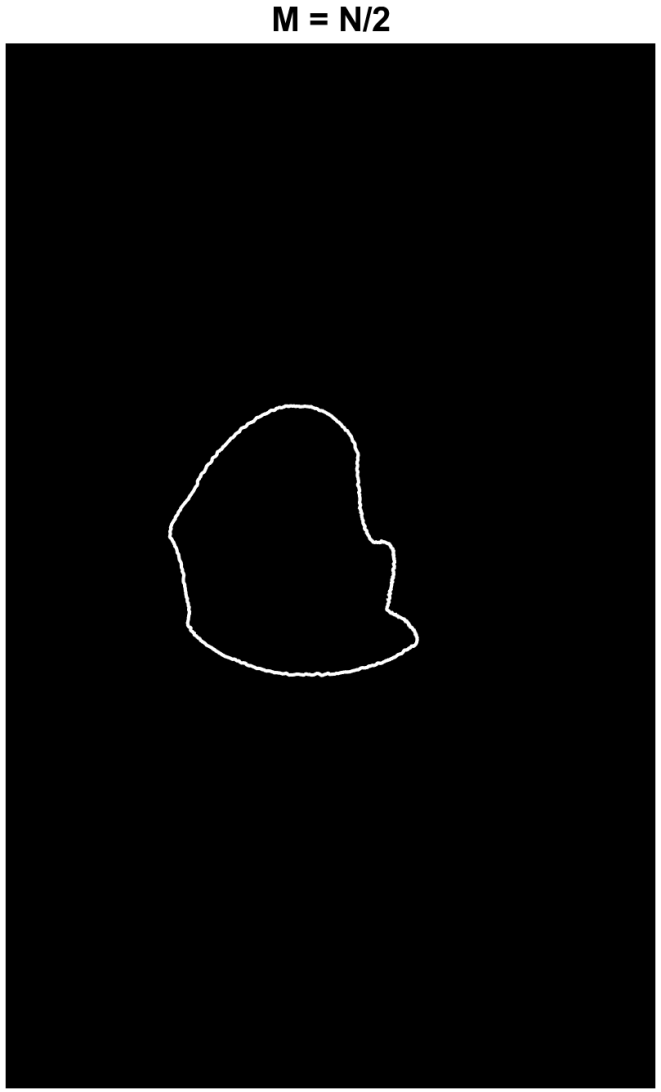
```
clc; clear; close all; f=20;
I = imread('airplane.bmp'); s = size(I);
y = importdata('c.mat');
N = length(y);
K = 1; %downsampling factor
%M = N/50; %number of points
yc = y(:,1) + 1i*y(:,2);
fd = fft(yc);
fs = downsample(fd,K);
Sc = (1/K)*ifft(fs);
Z = zeros(length(Sc),2);
Z(:,1) = real(Sc); Z(:,2) = imag(Sc);
boundary_im = zeros(s(1),s(2));
figure(1);imshow(boundary_im);hold on;
plot(Z(:,1),Z(:,2),'w',LineWidth=2);
title(sprintf('M = N/%d',K),FontSize=f);
```



With downsampling factor K = 1 (in M = N/K),
it worked as expected.

MATLAB code for downsampled Fourier Descriptor

```
clc; clear; close all; f=20;
I = imread('airplane.bmp'); s = size(I);
y = importdata('c.mat');
N = length(y);
K = 2; %downsampling factor
%M = N/50; %number of points
yc = y(:,1) + 1i*y(:,2);
fd = fft(yc);
fs = downsample(fd,K);
Sc = (1/K)*ifft(fs);
Z = zeros(length(Sc),2);
Z(:,1) = real(Sc); Z(:,2) = imag(Sc);
boundary_im = zeros(s(1),s(2));
figure(1);imshow(boundary_im);hold on;
plot(Z(:,1),Z(:,2),'w',LineWidth=2);
title(sprintf('M = N/%d',K),FontSize=f);
```



With downsampling factor K = 2 (in M = N/K), it showed a distorted boundary only. For higher K values, the results worsened and the boundary became smaller

Discussion

In downsampled fourier descriptor part, expected results were not achieved. I will just explain what I did in the code, if that brings some partial marks.

1. Image has been read using imread(). The boundary data were saved from the first task of part I. That data was read:

```
clc; clear; close all; f=20;
I = imread('airplane.bmp'); s = size(I);
y = importdata('c.mat');
N = length(y);
```

2. K is defined as M = N/K or K = N/M. K = 2 was chosen initially.

3. Complex numbers were formed for 1-D representation from boundary data: yc = y(:,1) + 1i*y(:,2);

4. DFT was taken: fd = fft(yc);

5. Before IDFT, downsampling was performed: fs = downsample(fd,K); Since K = 2, the number of DFT coefficients became half

6. IDFT is taken, with scaling factor 1/K = M/N: Sc = (1/K)*ifft(fs);

7. Real and imaginary parts of scaled IDFT was taken: Z = zeros(length(Sc),2); Z(:,1) = real(Sc); Z(:,2) = imag(Sc);

8. The black background was created and plotted: boundary_im = zeros(s(1),s(2)); figure(1); imshow(boundary_im); hold on;

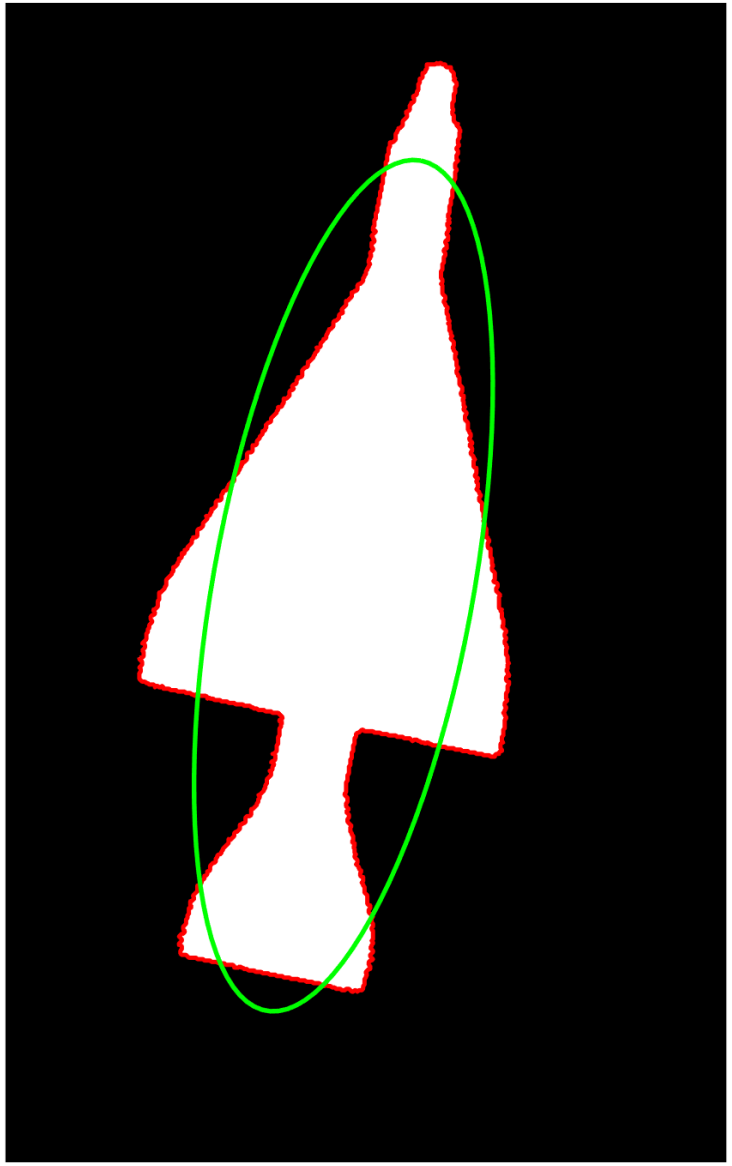
9. The reconstructed boundary was plotted: plot(Z(:,1),Z(:,2),'w',LineWidth=2);title(sprintf('M = N/%d',K),FontSize=f);

MATLAB code for ellipse approximation

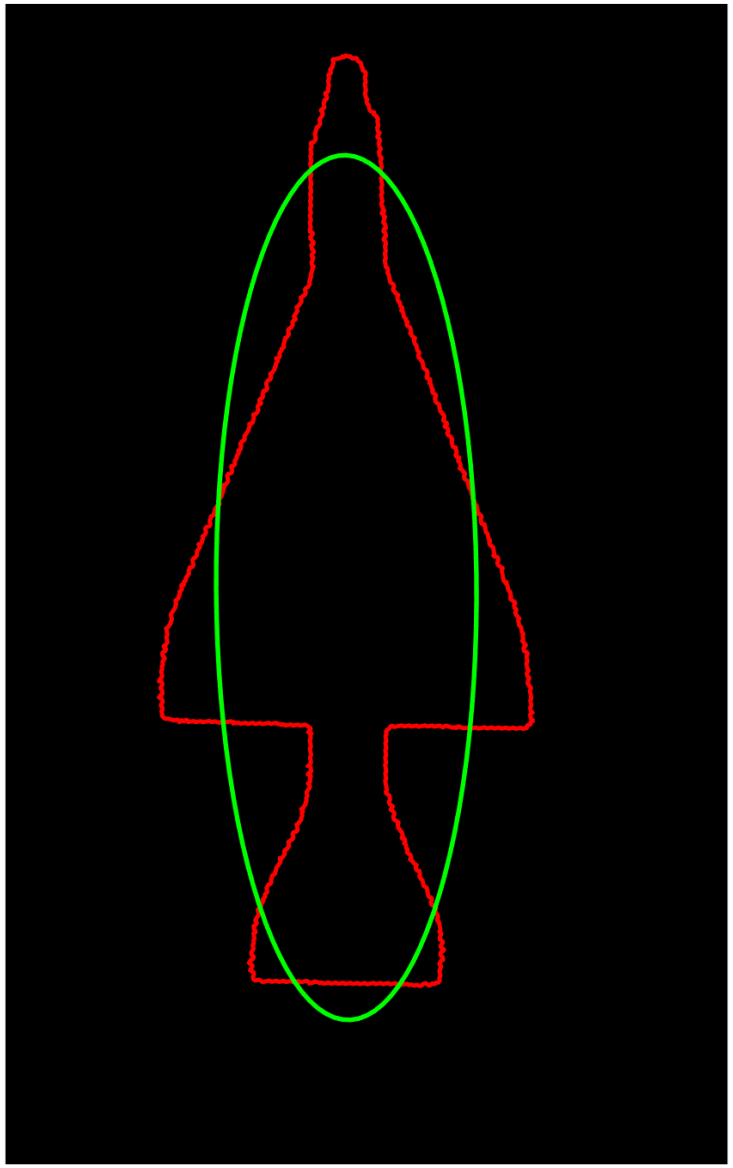
```
clc; clear; close all; f=18;
I = im2double(imread('airplane.bmp')); s = size(I);
figure(1);imshow(I);hold on;
title('Original Image, Boundary & Ellipse Approximation', fontsize=f);
y = importdata('c.mat');
yflip = fliplr(y);
[Z, A, B, alpha]=fitellipse(yflip','linear');
T(1,1)=Z(2,1);
T(2,1)=Z(1,1);
plot(y(:,1),y(:,2),'r',LineWidth=2.4);
plotellipse(T,B,A,-alpha);

%aligned
yalign = importdata('y2.mat');
yalignflip = fliplr(yalign);
boundary_im = zeros(s(1),s(2));
figure(2);imshow(boundary_im);hold on;
plot(yalign(:,1),yalign(:,2),'r',LineWidth=2.4);
title('Aligned boundary & Ellipse Approximation', fontsize=f);
[Z, A, B, alpha]=fitellipse(yalignflip','linear');
T(1,1)=Z(2,1);
T(2,1)=Z(1,1);
plotellipse(T,B,A,-alpha);
```

Results of Ellipse Approximation

Original Image, Boundary & Ellipse Approximation



Aligned boundary & Ellipse Approximation



Discussion

1. In the boundary data that were saved as .mat format, the order of x and y co-ordinates were reversed.Hence, filplr() is called to make them match with MATLAB image convention

```
y = importdata('c.mat');
yflip = fliplr(y);
```

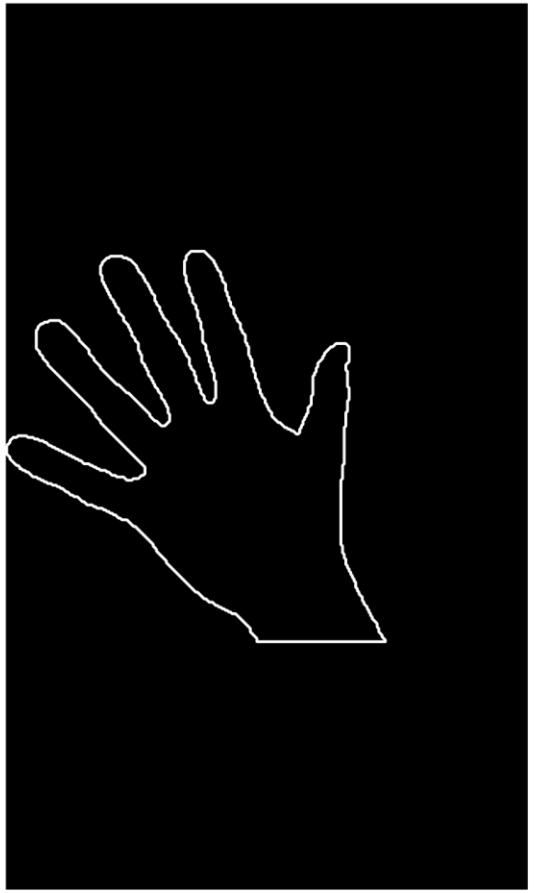
   However, later, necessary axis shifting has been performed again for plotting the ellipse, according to the given test_ellipse code.

2. There are many points in the boundary which are outside the ellipse. This is because the fitellipse() function did least squares fit, hence there is no guarantee that all boundary points will be inside the ellipse.

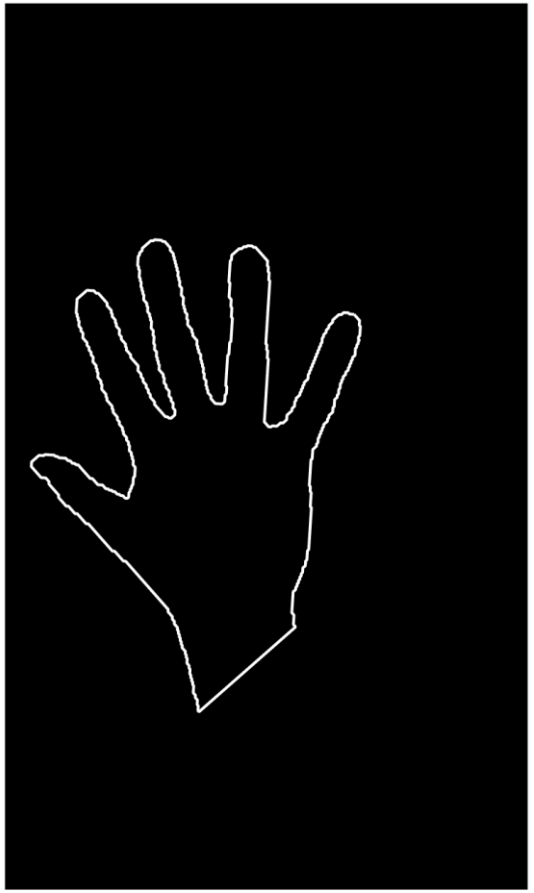
Bonus: try with another image (hand3.bmp)

Original binary Image

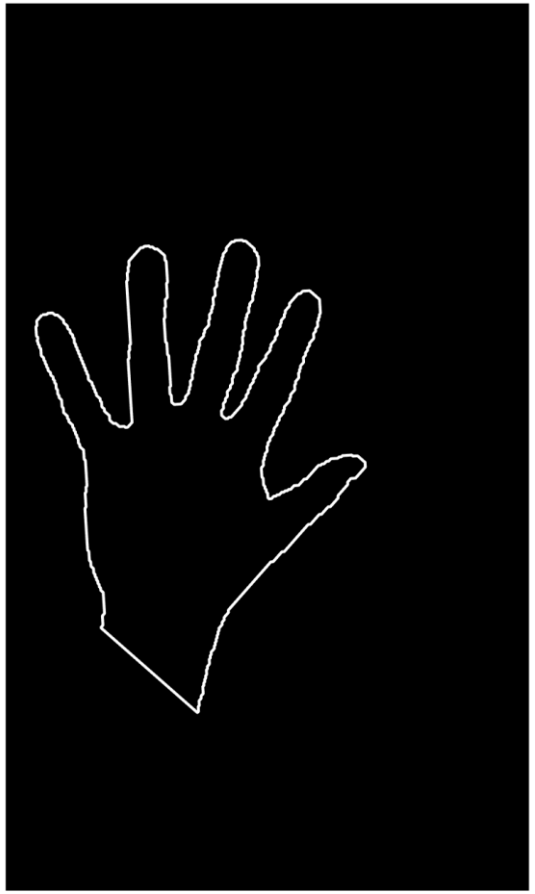
Boundary (extracted through bwboundaries())



Vertically aligned boundary (clockwise rotation)



Vertically aligned boundary (anti-clockwise rotation)

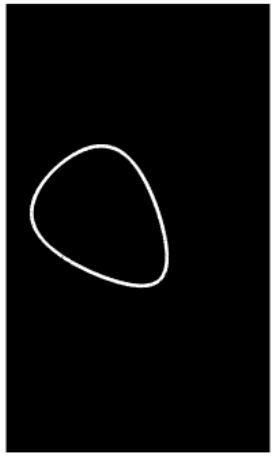


Bonus: try with another image (hand3.bmp)

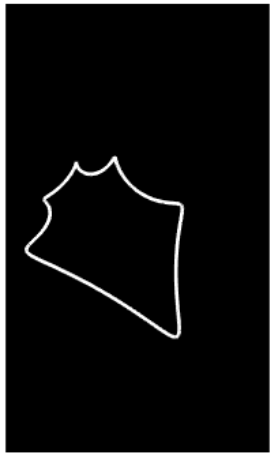
Fourier descriptor with different P

Original boundary size: 1167. P=30 is giving sufficiently good reconstruction

P = 2



P = 5



P = 10

P = 15

P = 20

P = 30

P = 40

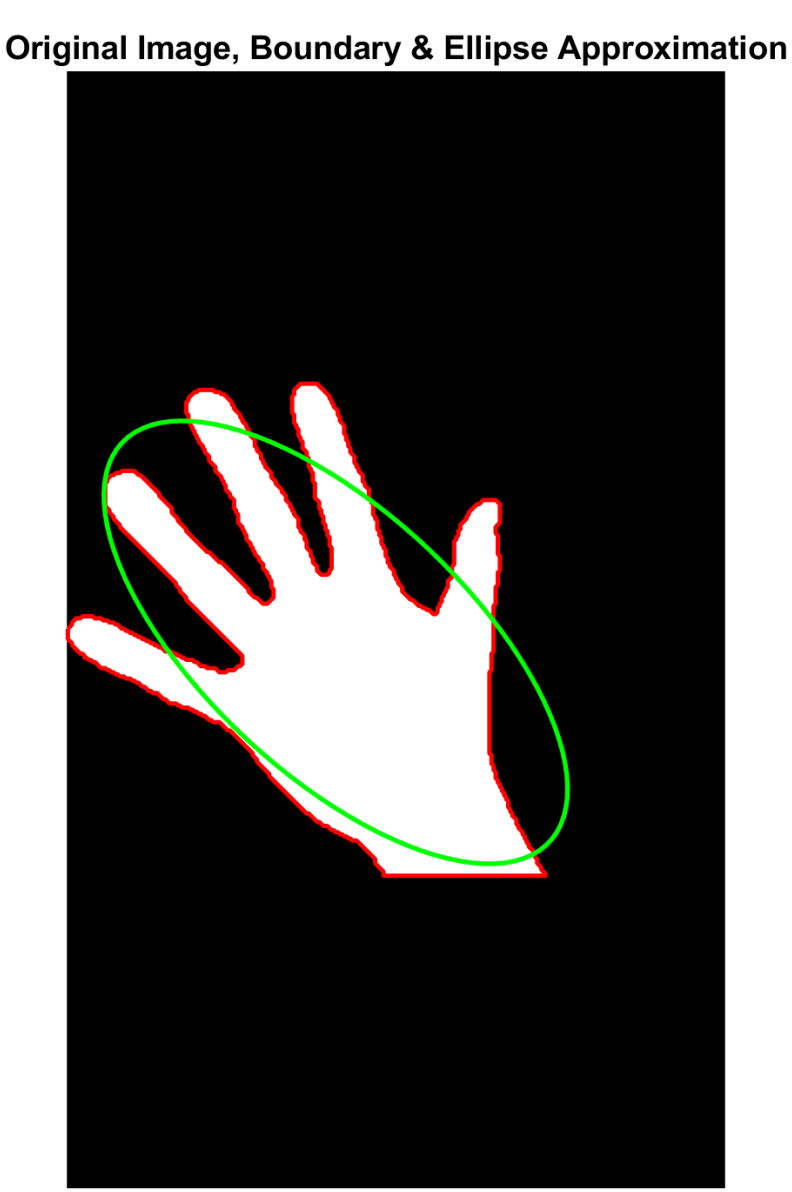
P = 45

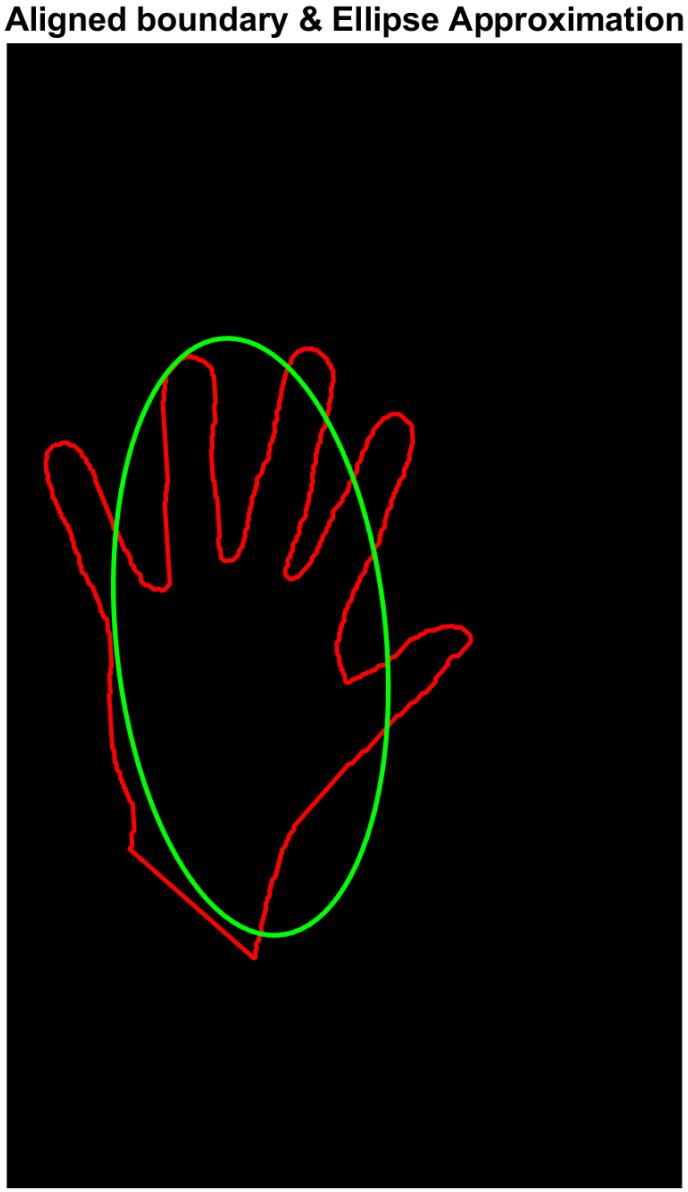
P = 50

P = 60

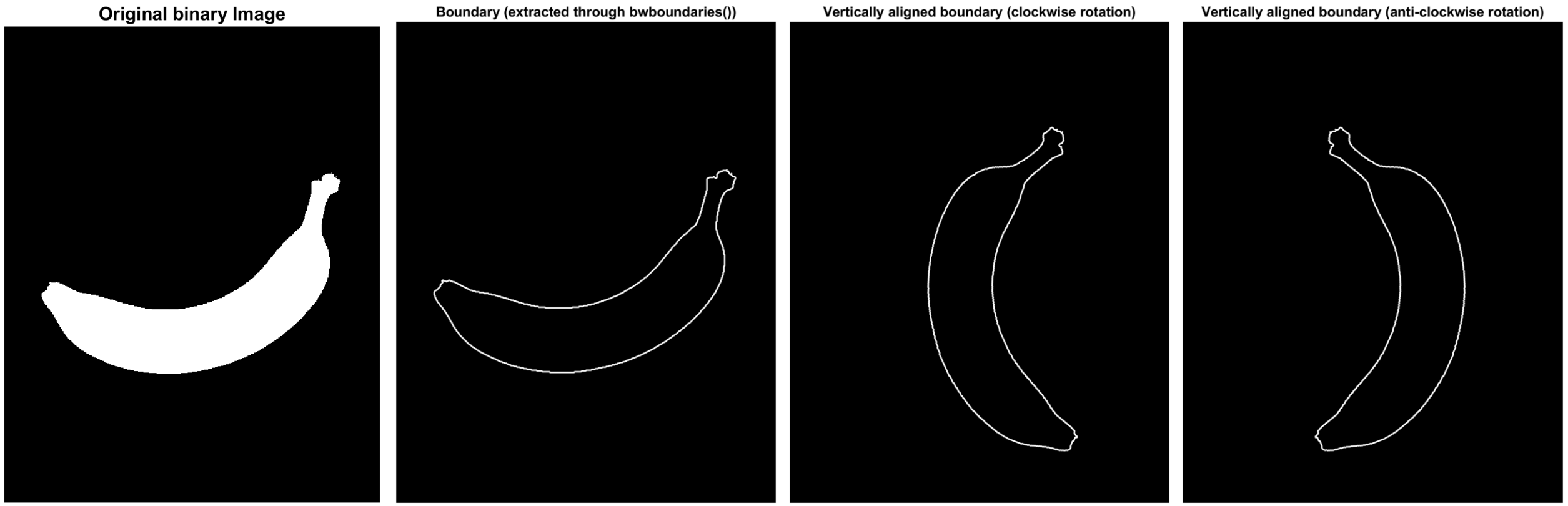
Bonus: try with another image (hand3.bmp)

Ellipse approximation





Bonus: try with other images (banana2.bmp)

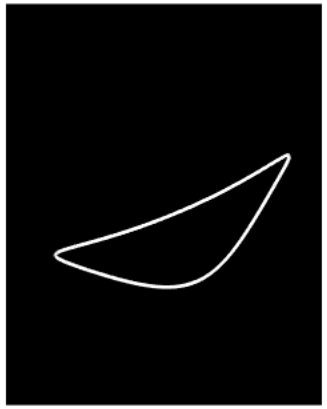


Bonus: try with other images (banana2.bmp)

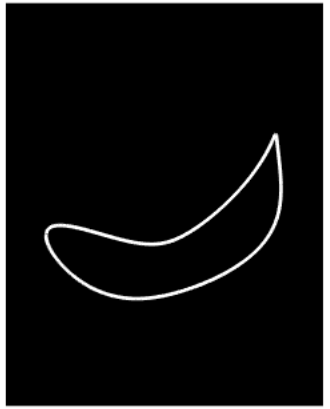
Fourier descriptor with different P

Original boundary size: 1009. P=45 is giving sufficiently good reconstruction

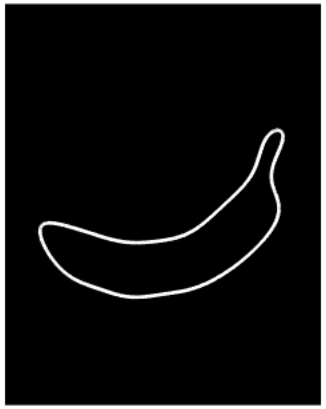
P = 2



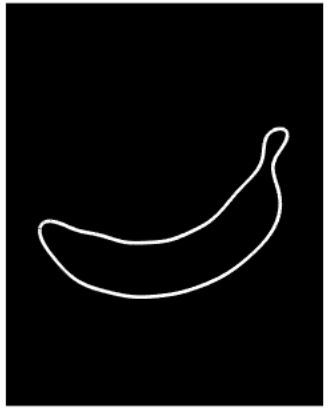
P = 5



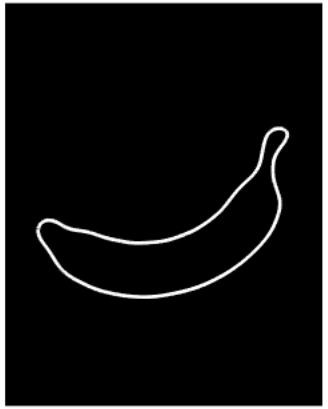
P = 10



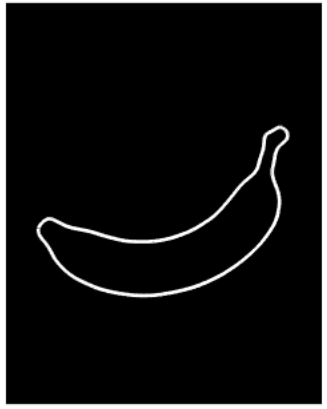
P = 15



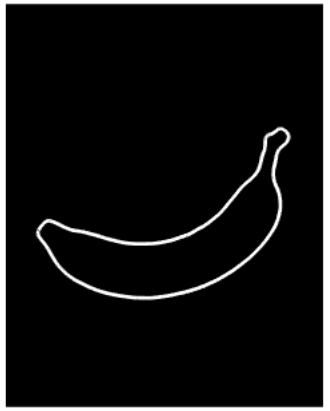
P = 20



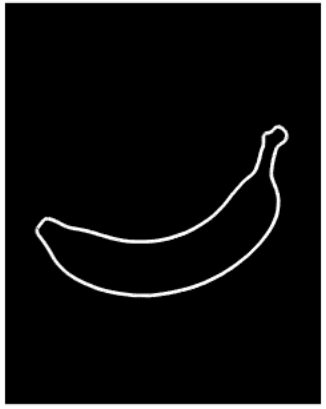
P = 30



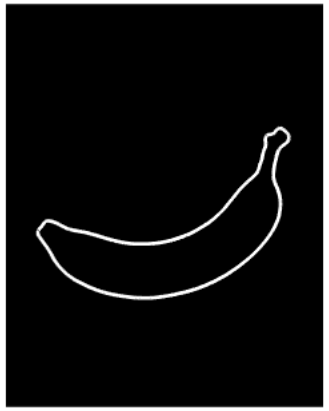
P = 40



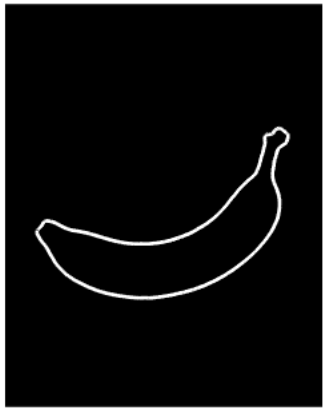
P = 45



P = 50



P = 60



Bonus: try with other images (banana2.bmp)

Ellipse approximation

It did not work, as shown in the picture. Higher order fit will be necessary, because linear fit is not generating any ellipse. This is happening because the shape of banana has no match with ellipse.

```
clc; clear; close all; f=18;
I = im2double(imread('banana2.bmp')); s = size(I);
figure(1);imshow(I);hold on;
title('Original Image, Boundary & Ellipse Approximation', fontsize=f);
y = importdata('c_banana.mat');
yflip = fliplr(y);
[Z, A, B, alpha]=fitellipse(yflip','linear');
T(1,1)=Z(2,1);
T(2,1)=Z(1,1);
plot(y(:,1),y(:,2),'r',LineWidth=2.4);
plotellipse(T,B,A,-alpha);

%aligned
yalign = importdata('y2_banana.mat');
yalignflip = fliplr(yalign);
boundary_im = zeros(s(1),s(2));
figure(2);imshow(boundary_im);hold on;
plot(yalign(:,1),yalign(:,2),'r',LineWidth=2.4);
title('Aligned boundary & Ellipse Approximation', fontsize=f);
[Z, A, B, alpha]=fitellipse(yalignflip','linear');
T(1,1)=Z(2,1);
T(2,1)=Z(1,1);
plotellipse(T,B,A,-alpha);
```

Command Window

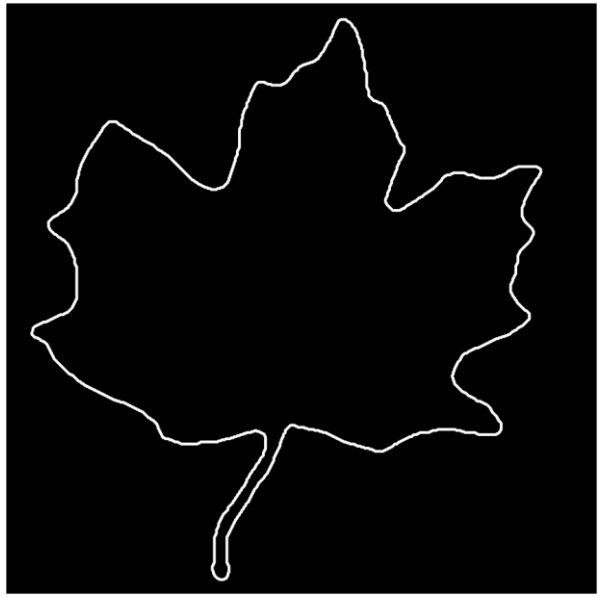
New to MATLAB? See resources for Getting Started.

```
Error using fitellipse>conic2parametric (line 294)
Linear fit did not produce an ellipse

Error in fitellipse>fitbookstein (line 166)
[z, a, b, alpha] = conic2parametric(A, bv, c);

Error in fitellipse (line 92)
        [z, a, b, alpha] = fitbookstein(x);

Error in partI_4 (line 7)
[Z, A, B, alpha]=fitellipse(yflip','linear');
```

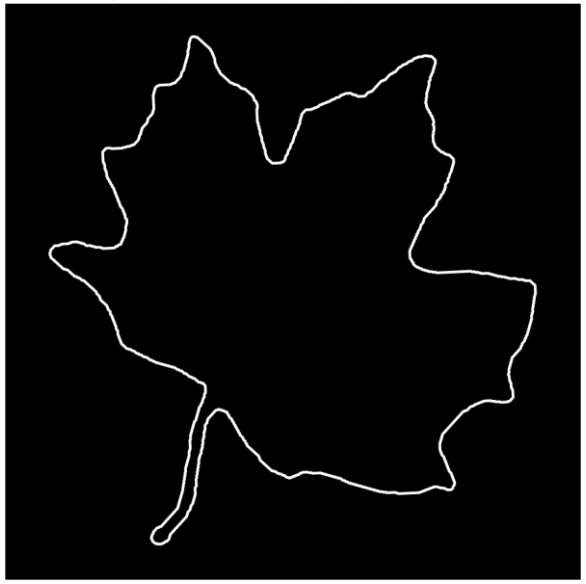
Bonus: try with another image (leaf.tif)

Original binary Image

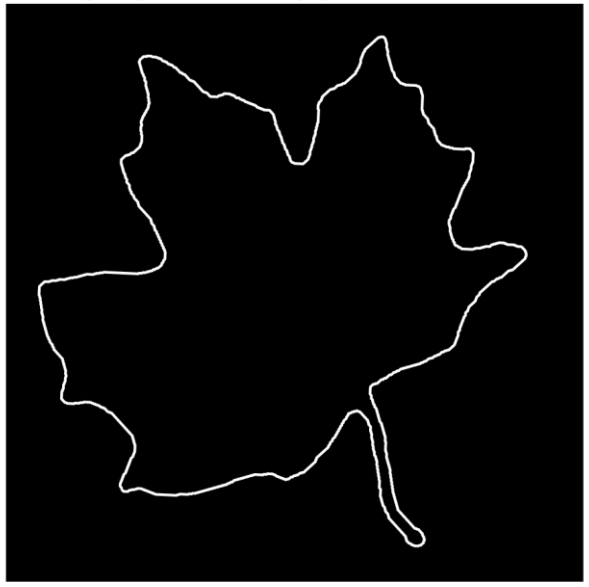
Boundary (extracted through bwboundaries())



Vertically aligned boundary (clockwise rotation)



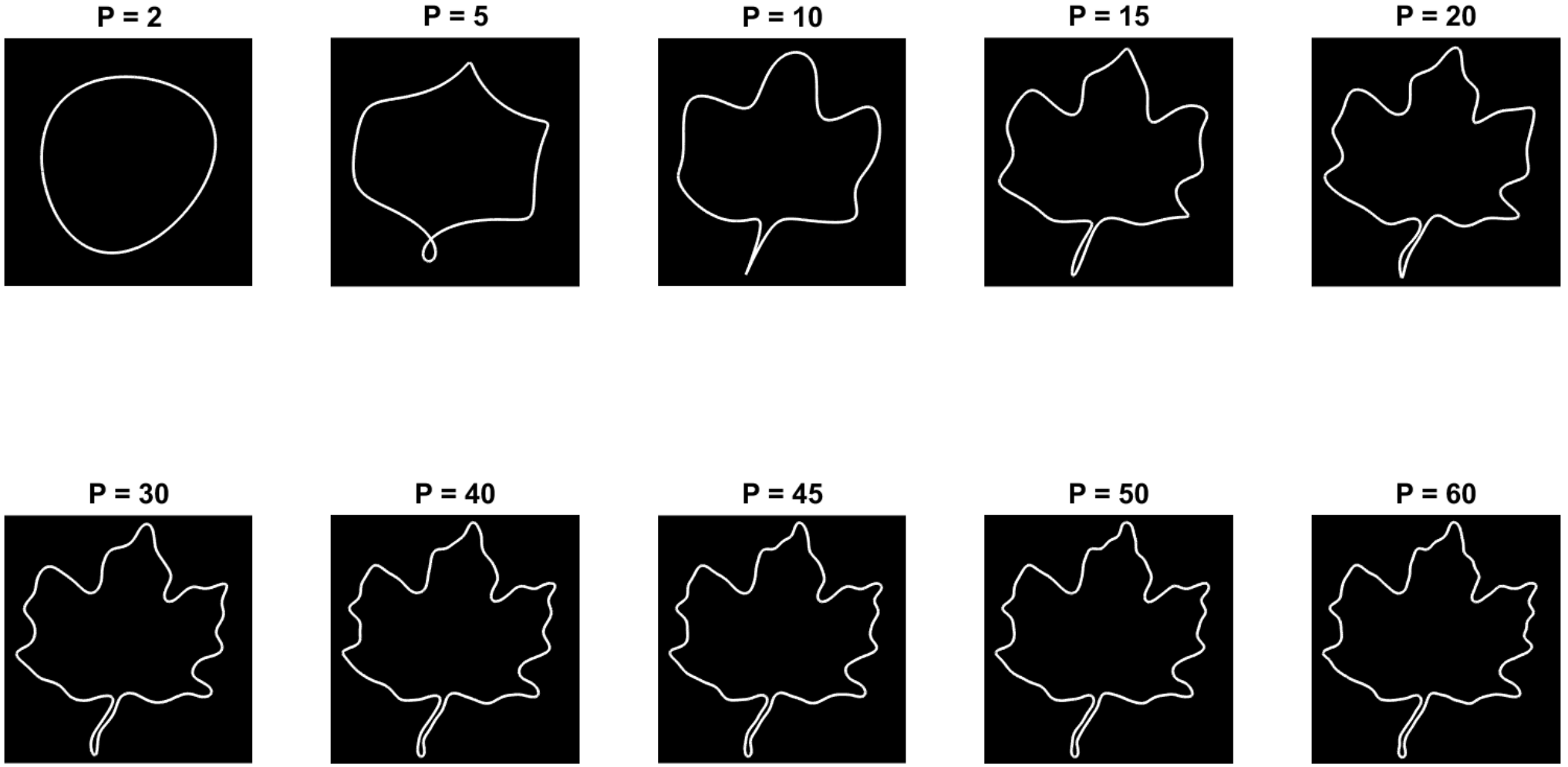
Vertically aligned boundary (anti-clockwise rotation)



Bonus: try with another image (leaf.tif)

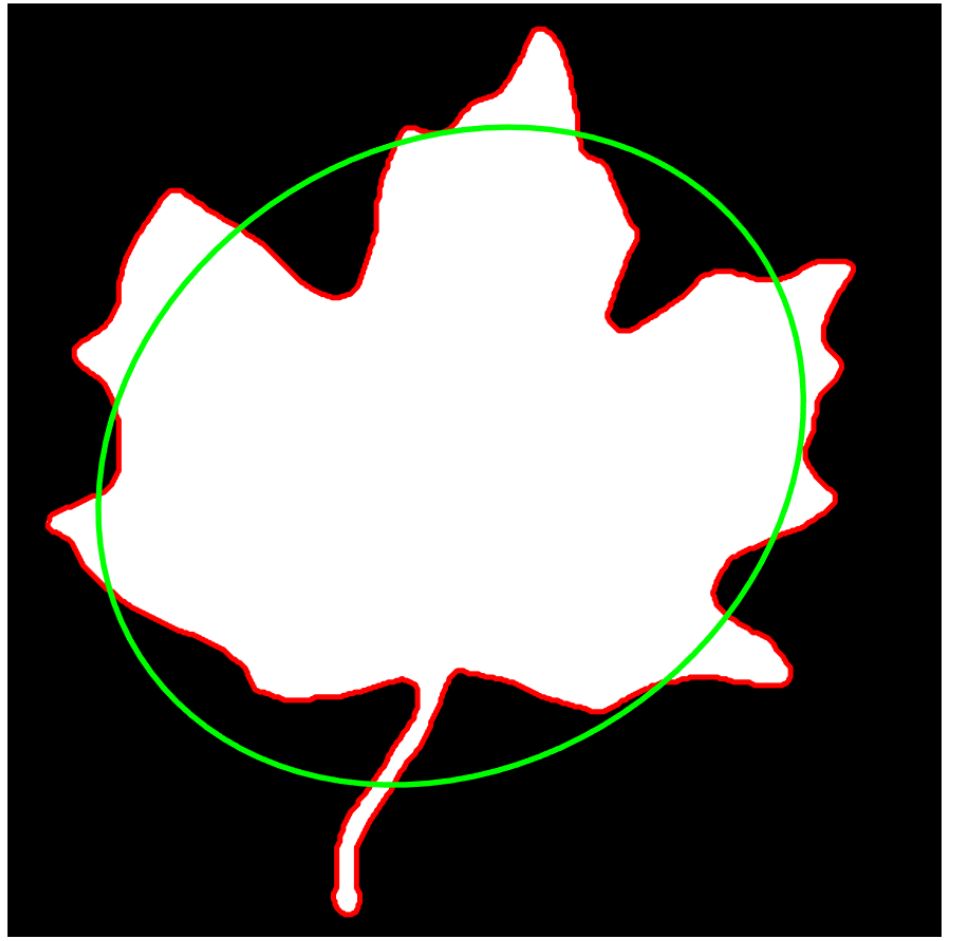
Fourier descriptor with different P

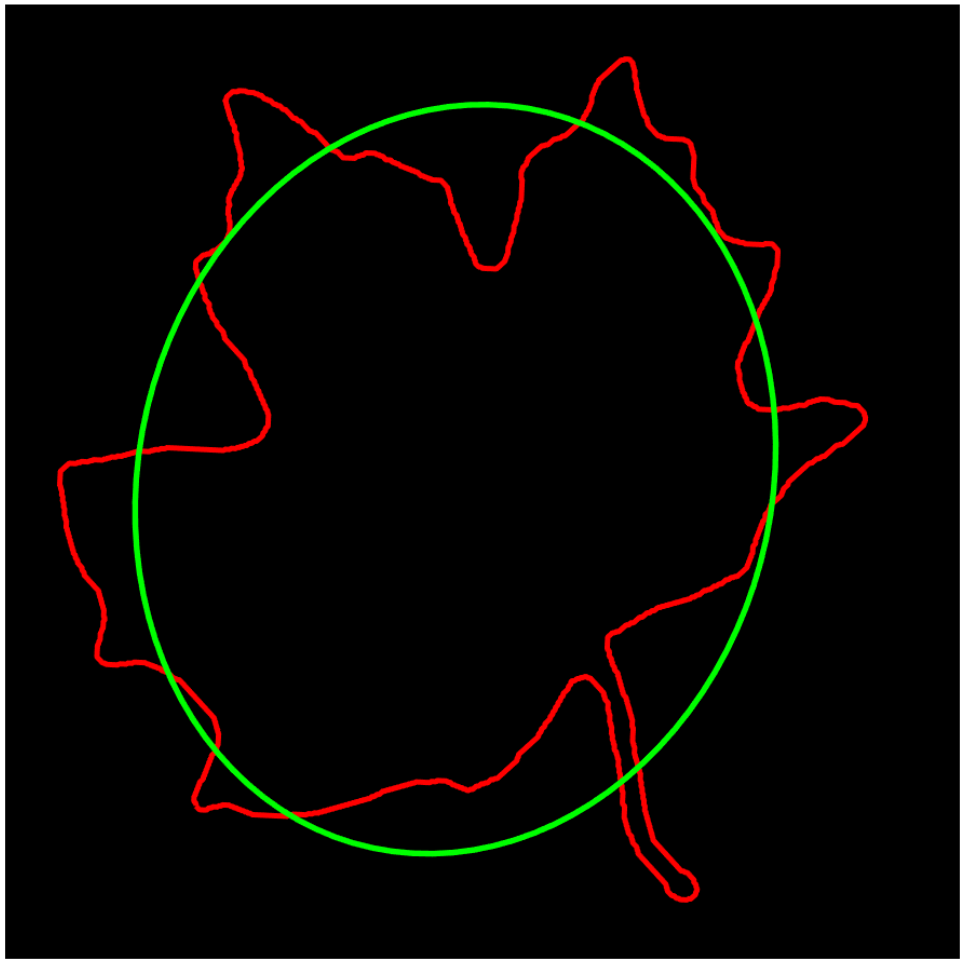
Original boundary size: 1900. P=40 is giving sufficiently good reconstruction



Bonus: try with another image (leaf.tif)

Ellipse approximation





Part II

II.1, II.2 and II.3 (MATLAB code)

The structuring element size was needed to be adjusted for different embryo images. This code is for em1.jpg.

For some images, the threshold for binarizing was needed to be changed too.

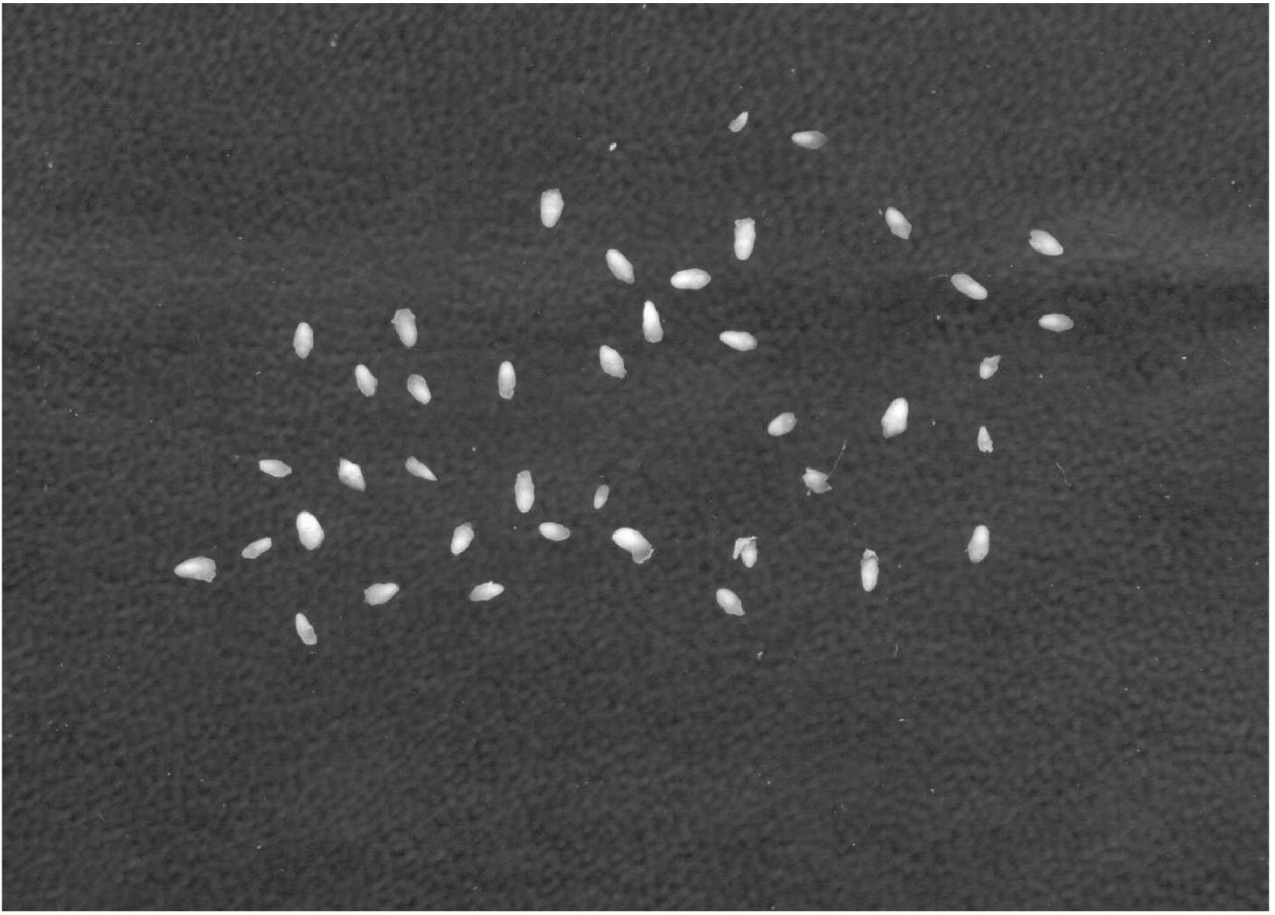
```
clc; clear; close all; f = 20;
I = imread("em1embryo.bmp");
s = size(I);
figure(1); imshow(I);title("Original image",fontsize=f);
J = imbinarize(I,0.5);
figure(2); imshow(J);title("Binarized image",fontsize=f);
E = strel("disk",3);
B = imerode(J,E);C = imdilate(B,E);
figure(3); imshow(C);
title("Binarized image after morphological processing",fontsize=f);
[L,num]=bwlabel(C,8);num
B = bwboundaries(C);
Bm = cell2mat(B);
figure(4);imshow(C);hold on;
boundary_im = zeros(s(1),s(2));
figure(4);plot(Bm(:,2),Bm(:,1),'r.',LineWidth=2);title('Boundaries marked',fontsize=f);
figure(5);imshow(C);hold on;title('Ellipse approximation',fontsize=f);
area = zeros(1,num);
pixcount = zeros(1,num);
for ii = 1:num
    A = zeros(s); A(find(L==ii))=1;
    pixcount(ii) = sum(A(:));
    B = bwboundaries(A);
    Bm = cell2mat(B);
    boundary_im = zeros(s(1),s(2));
    [Z, A, B, alpha]=fitellipse(Bm','linear');
    T(1,1)=Z(2,1);
    T(2,1)=Z(1,1);
    plotellipse(T,B,A,-alpha); hold on;
    area(ii) = pi*A*B;
end
area
avg_area_ellipse = mean(area)
pixcount
avg_area_pixel = mean(pixcount)
```

II.1, II.2 & II.3 (Results for em1.jpg)

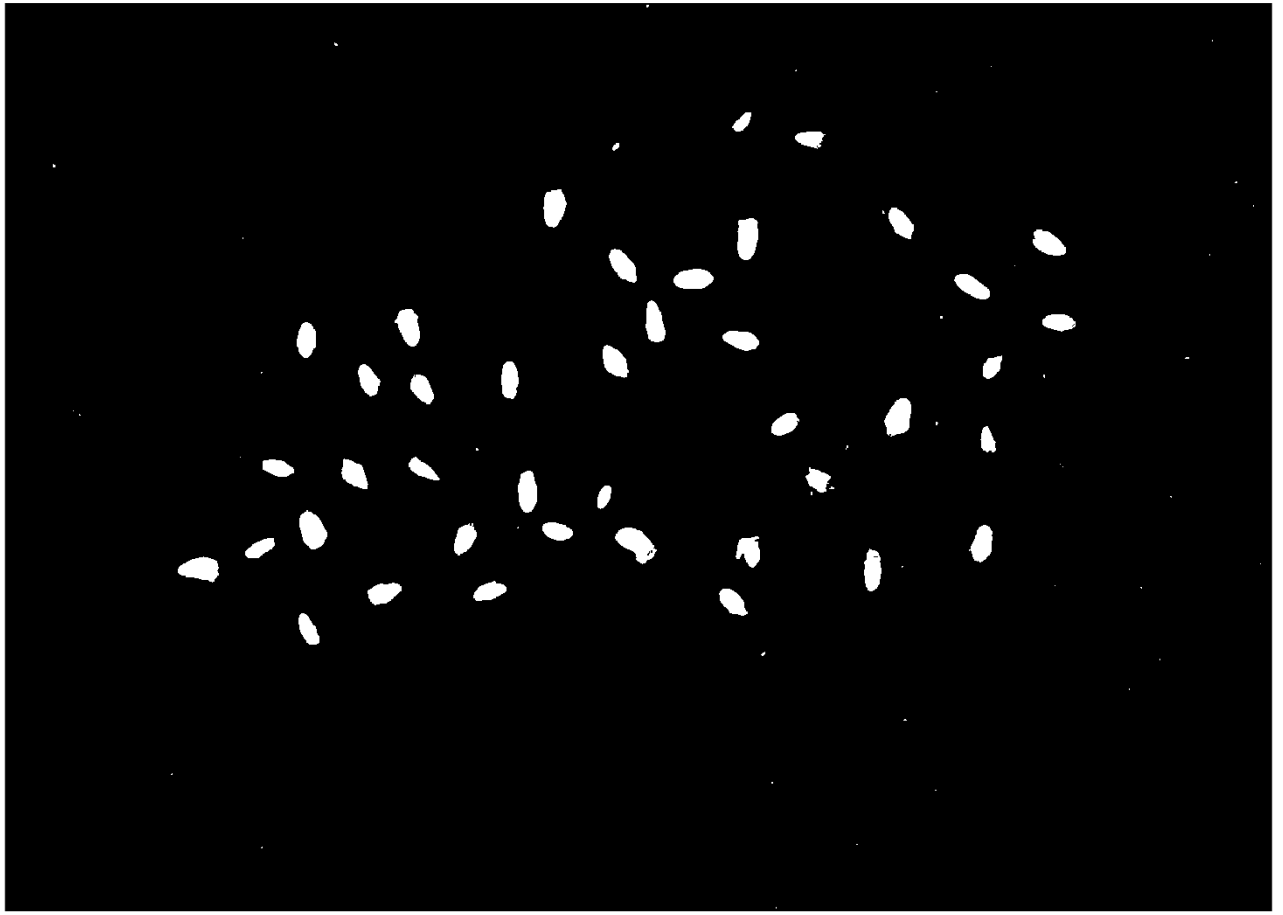
For best result, Threshold for binarizing: 0.5, structural element size = 3 (E = strel("disk",3);)
Number of embryos = number of connected components = 41

II.1, II.2 and II.3 (Results for em1.jpg)

Original image

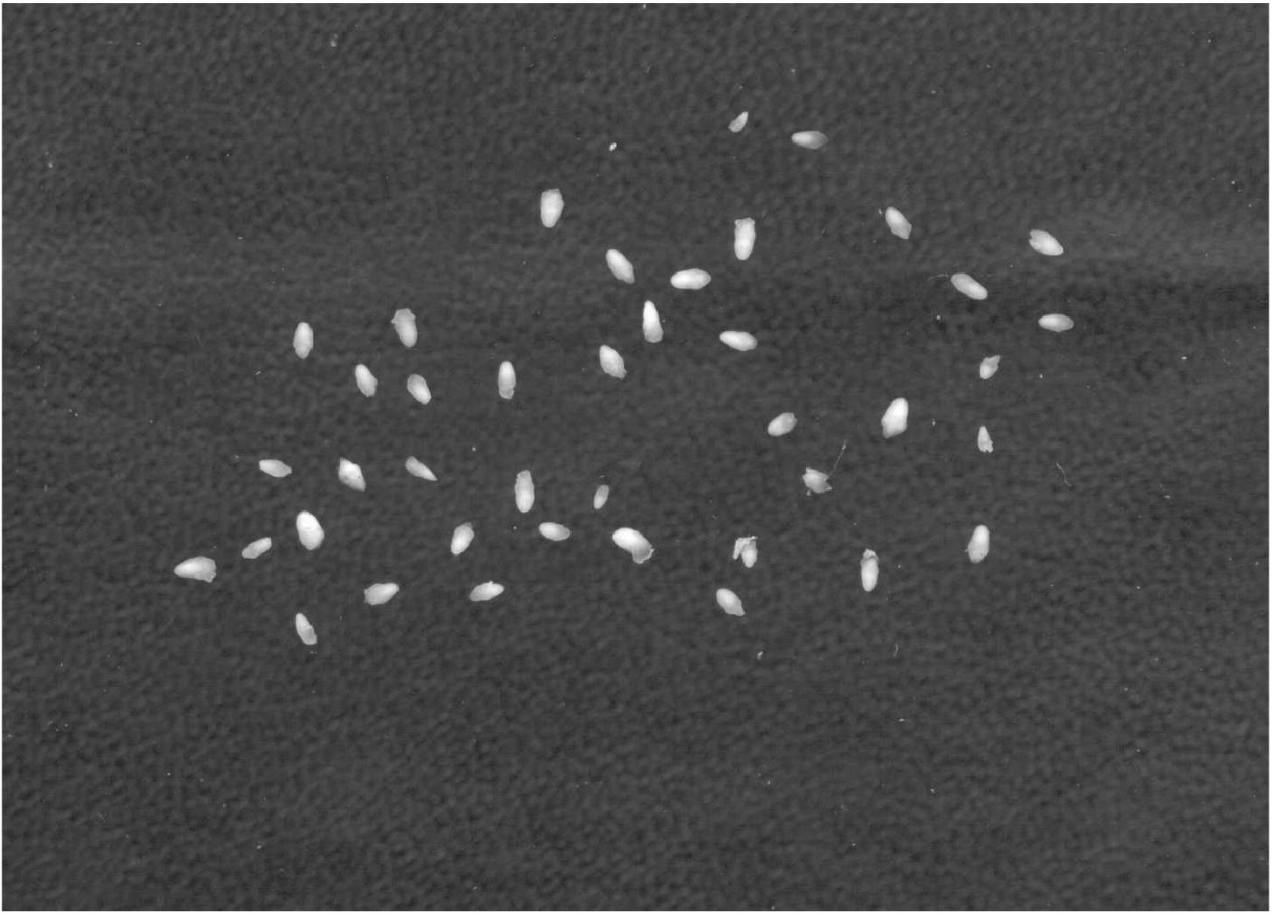


Binarized image

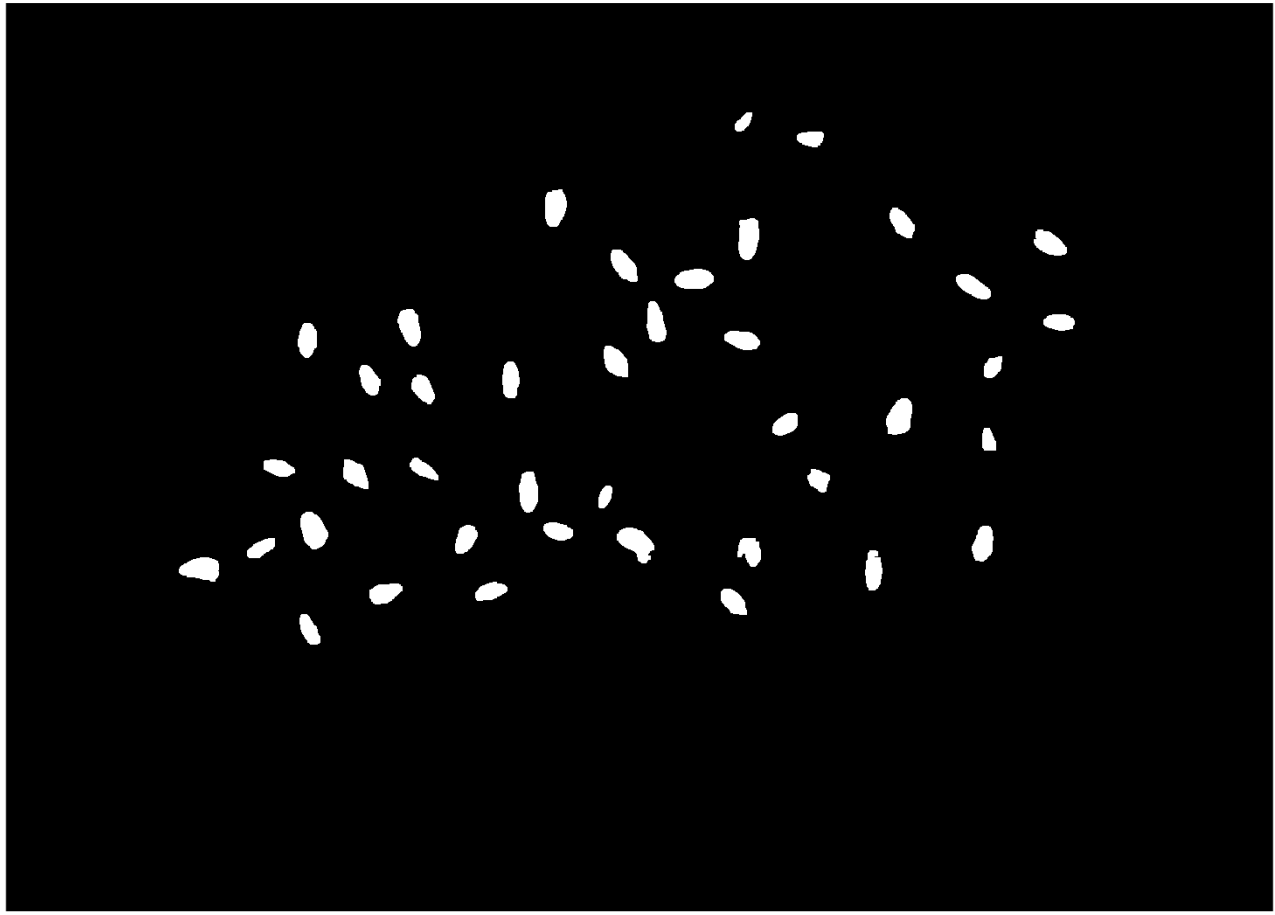


II.1, II.2 and II.3 (Results for em1.jpg)

Original image

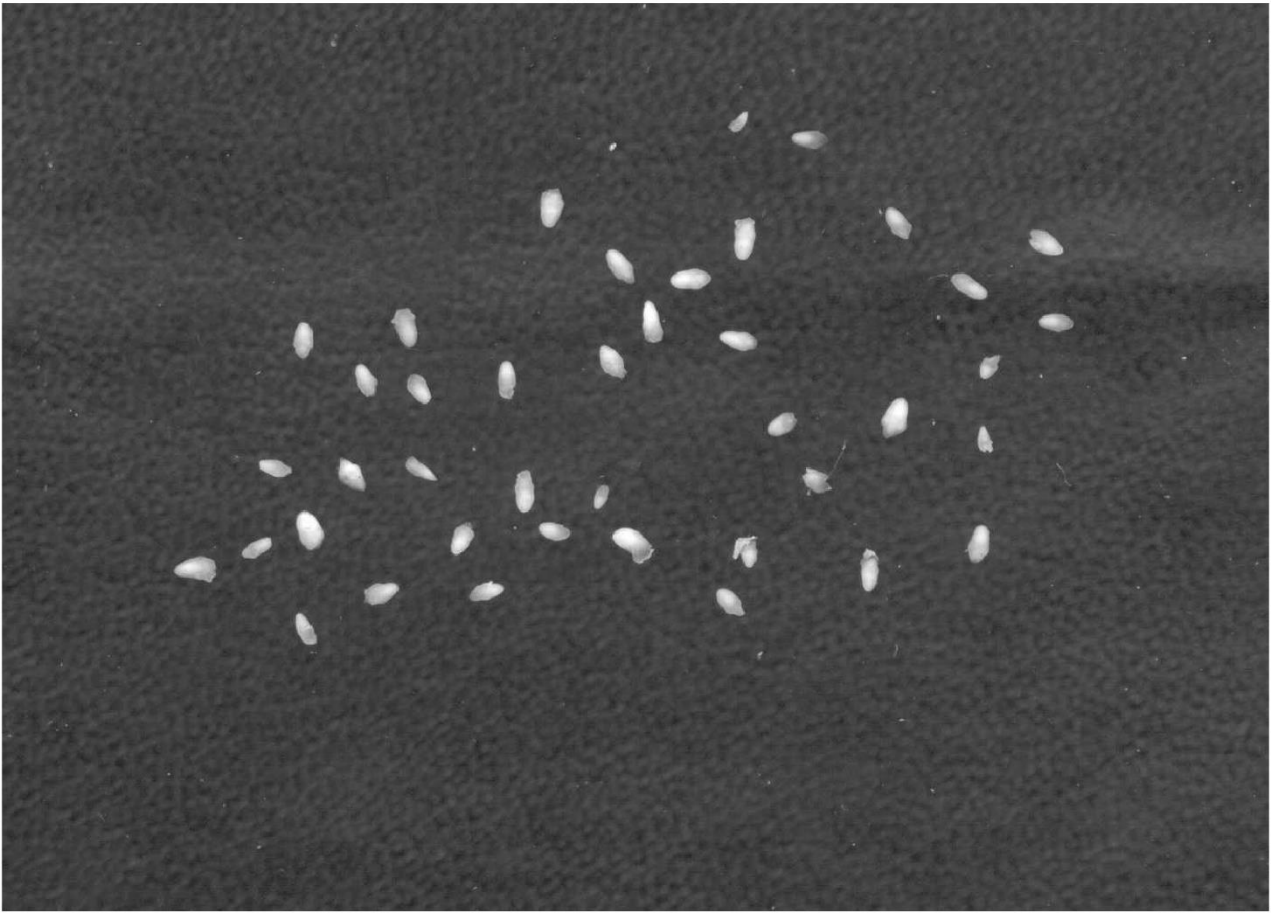


Binarized image after morphological processing

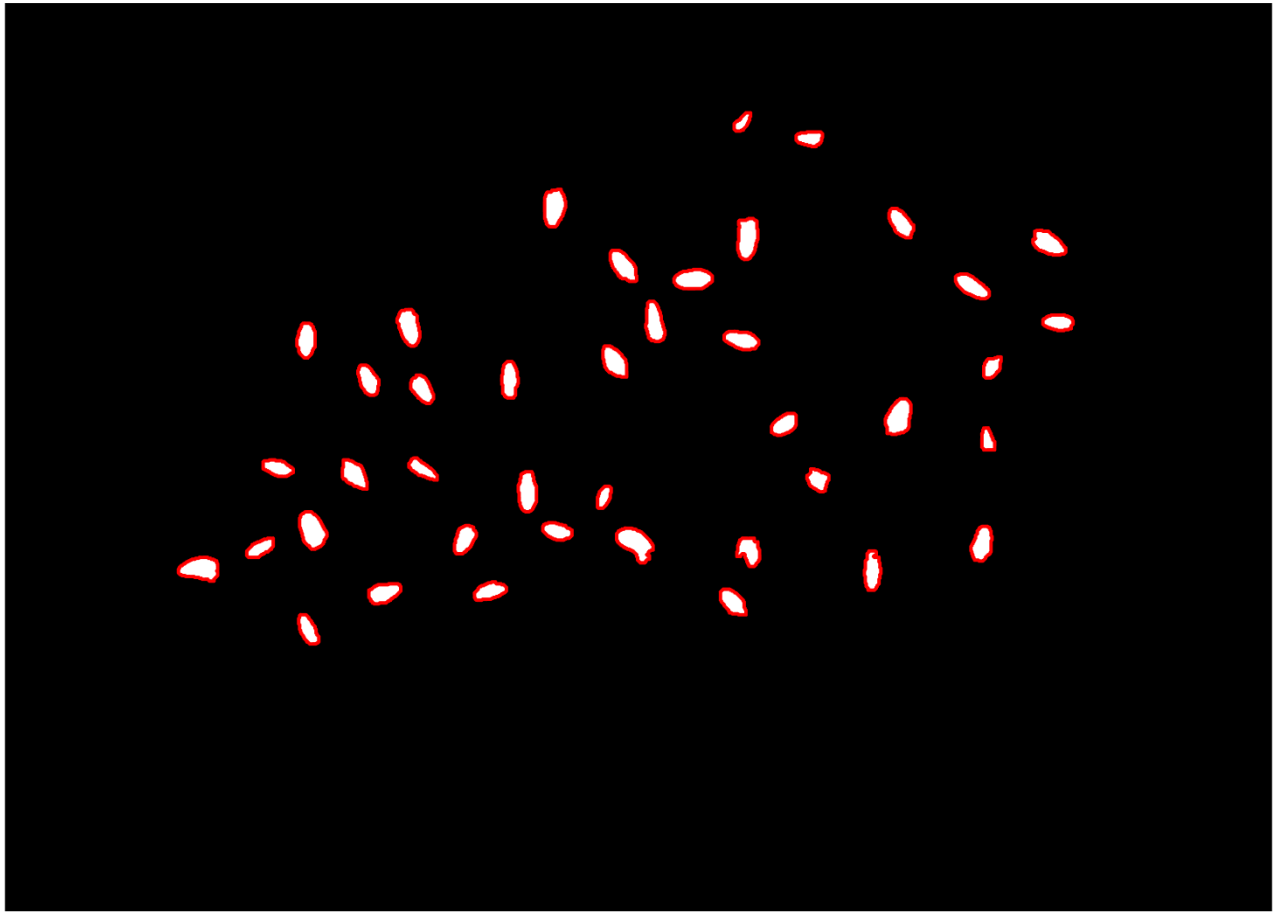


II.1, II.2 and II.3 (Results for em1.jpg)

Original image

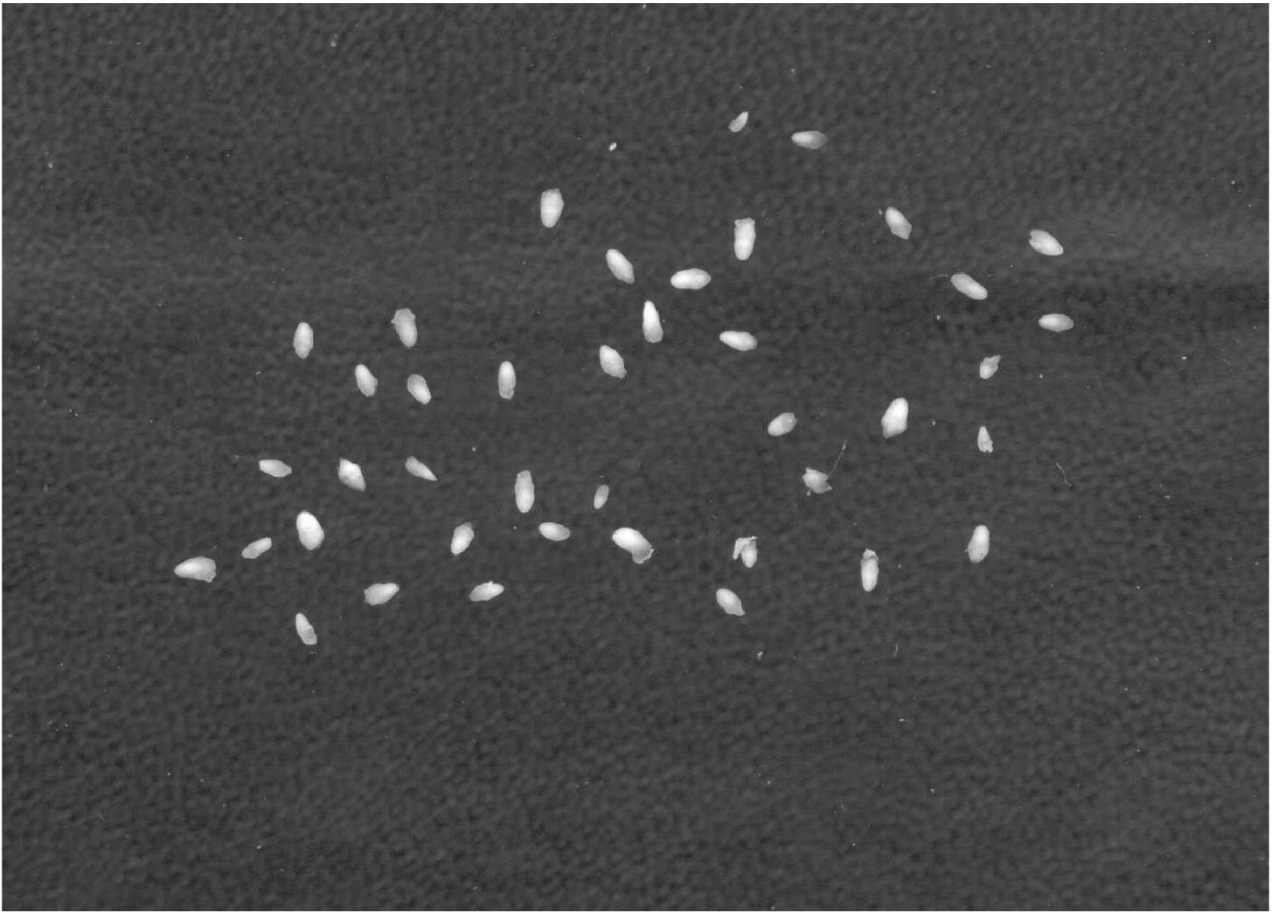


Boundaries marked

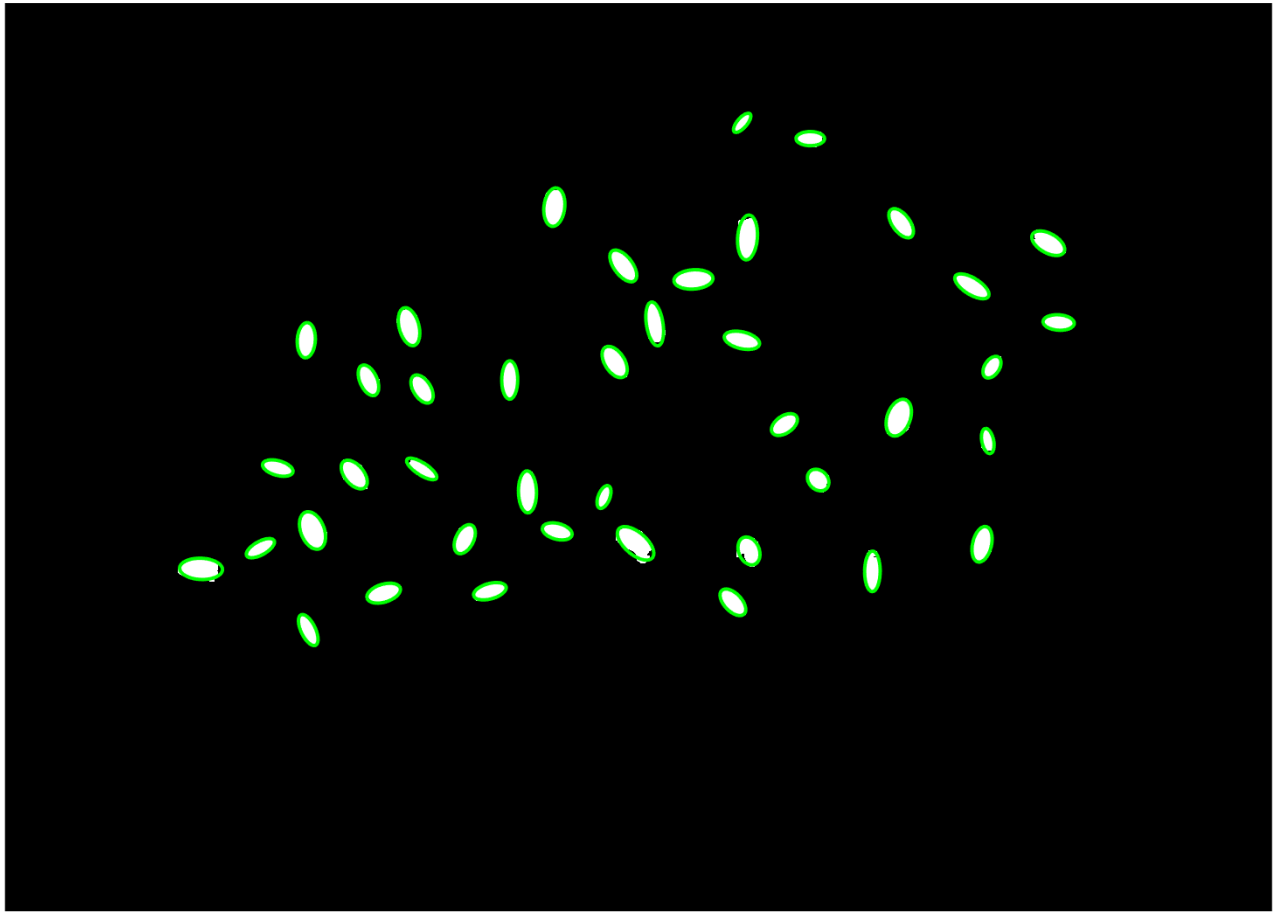


II.1, II.2 and II.3 (Results for em1.jpg)

Original image



Ellipse approximation



Number of embryos = number of connected components = 41

II.1, II.2 and II.3(Results for em1.jpg)

Average area in terms of ellipse approximation = 571.8785

```
num =

    41


area =

  Columns 1 through 13

  862.4139  397.5792  436.6270  601.5231  461.5697  872.1375  609.1825  532.3538  585.8029  727.9240  403.0188  518.9447  526.3171

  Columns 14 through 26

  489.4300  579.2844  714.4679  456.9979  761.2432  265.6476  650.9403  651.4661  926.4798  695.2019  712.5468  543.0625  571.7433

  Columns 27 through 39

  222.2412  564.0420  837.1984  475.3204  381.1836  425.9634  595.7555  836.7164  558.7206  607.5375  648.7347  284.9059  335.0198

  Columns 40 through 41

  663.0954  456.6763


avg_area =

  571.8785

fx >>
```

II.1, II.2 and II.3(Results for em1.jpg)

Average area in terms of pixel count = 606.7073

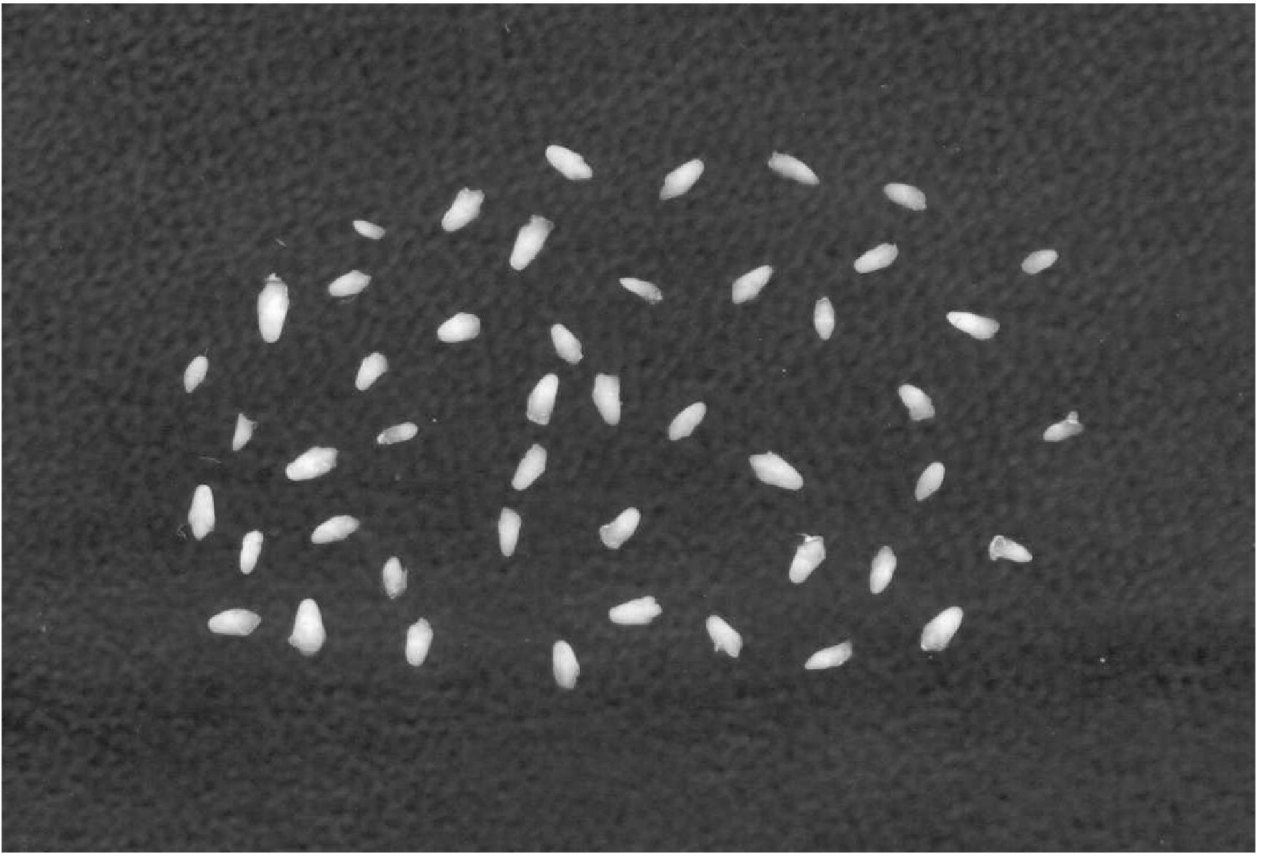
```
pixcount =

  Columns 1 through 21

   874   430   472   644   497   913   645   570   617   768   424   553   563   525   622   760   492   805   294   693   695

  Columns 22 through 41

   949   724   759   580   611   241   596   868   510   406   451   650   880   596   649   689   301   364   703   492


avg_area_pixel =

  606.7073
```

II.1, II.2 and II.3(Results for em2.jpg)

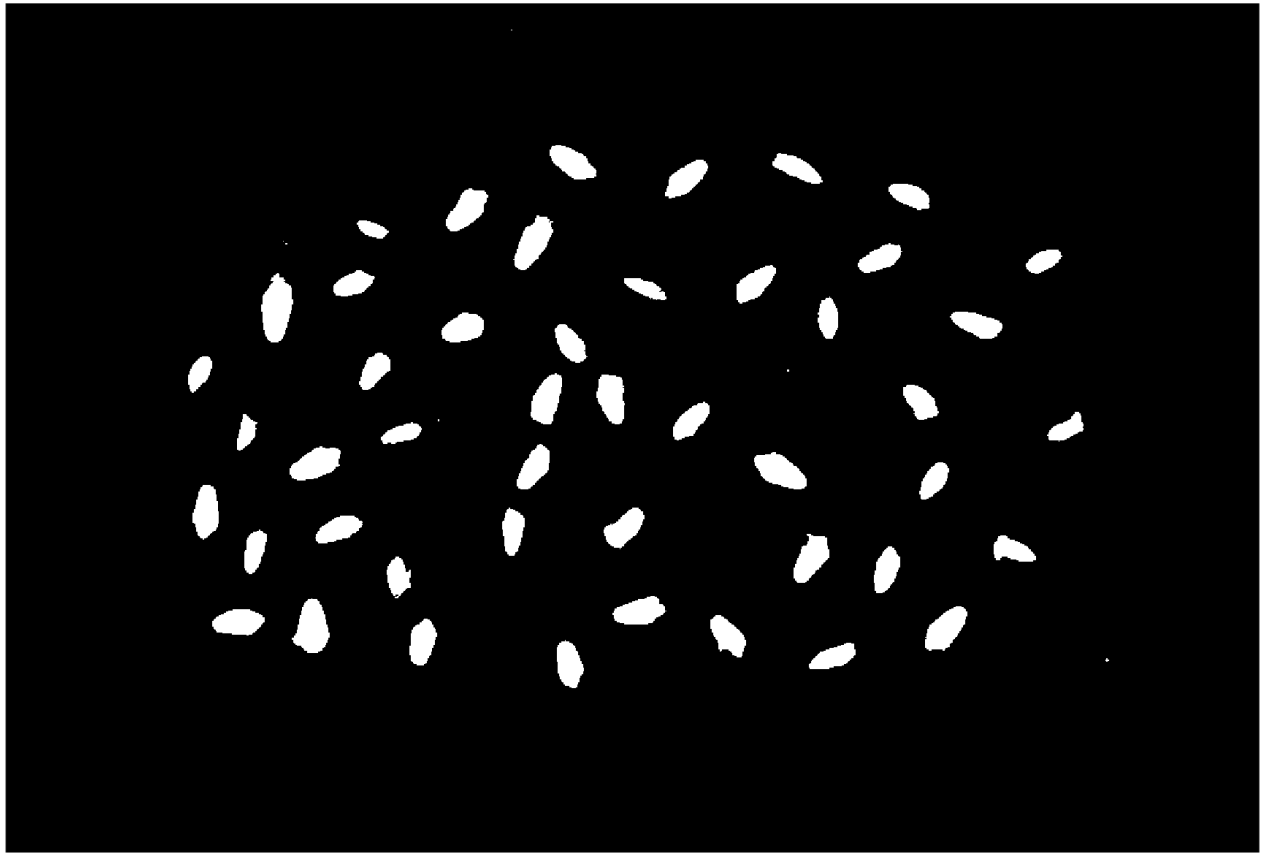
For best result, Threshold for binarizing: 0.5, structural element size = 3 (E = strel("disk",3);)
Number of embryos = number of connected components = 47

II.1, II.2 and II.3 (Results for em2.jpg)

Original image

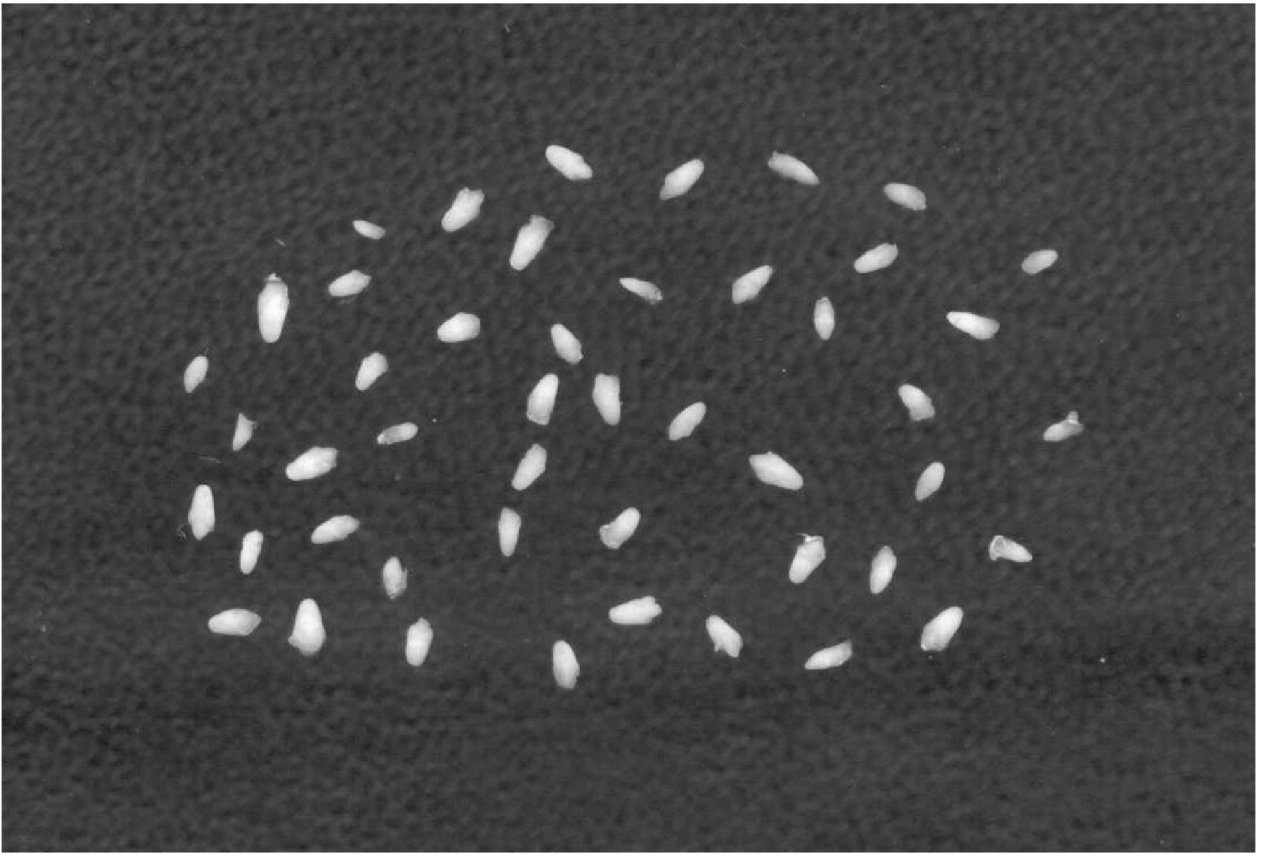


Binarized image

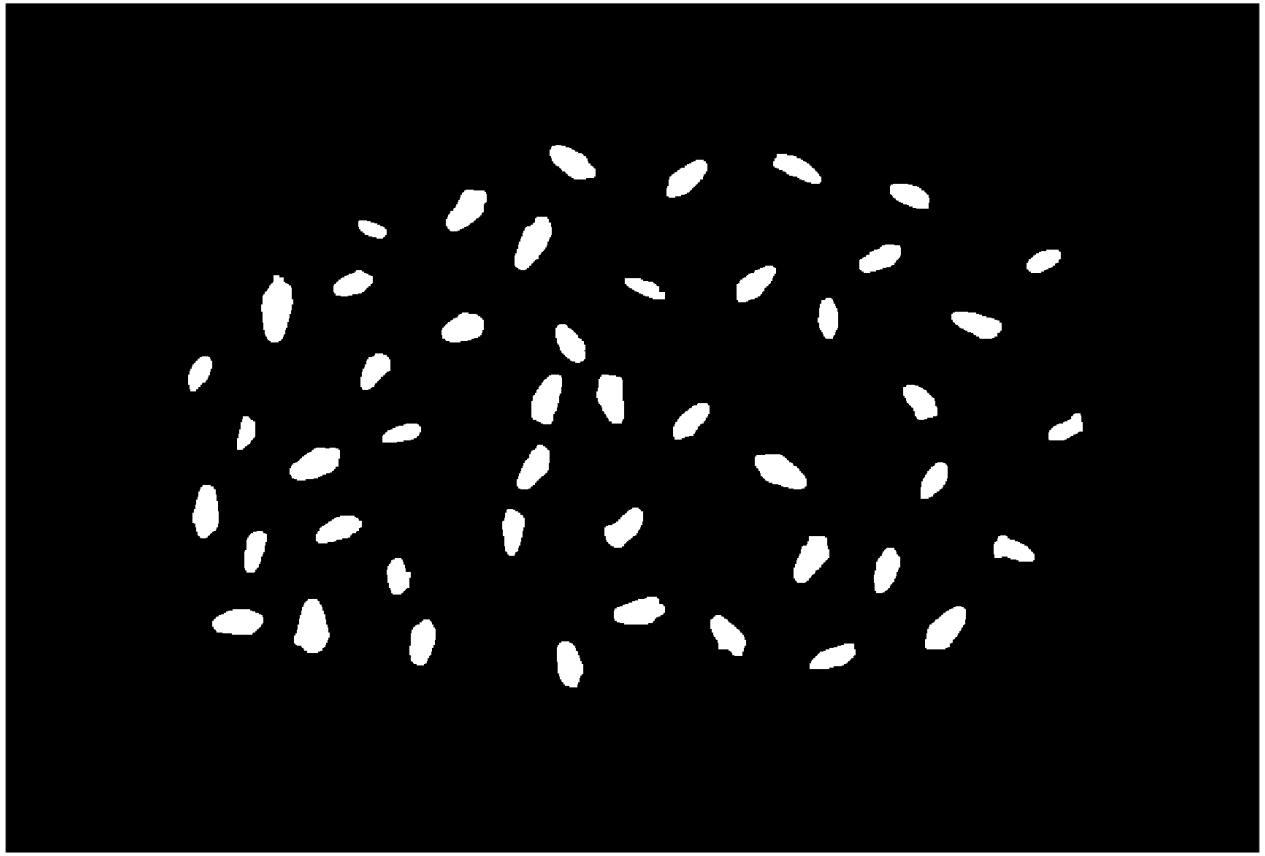


II.1, II.2 and II.3 (Results for em2.jpg)

Original image

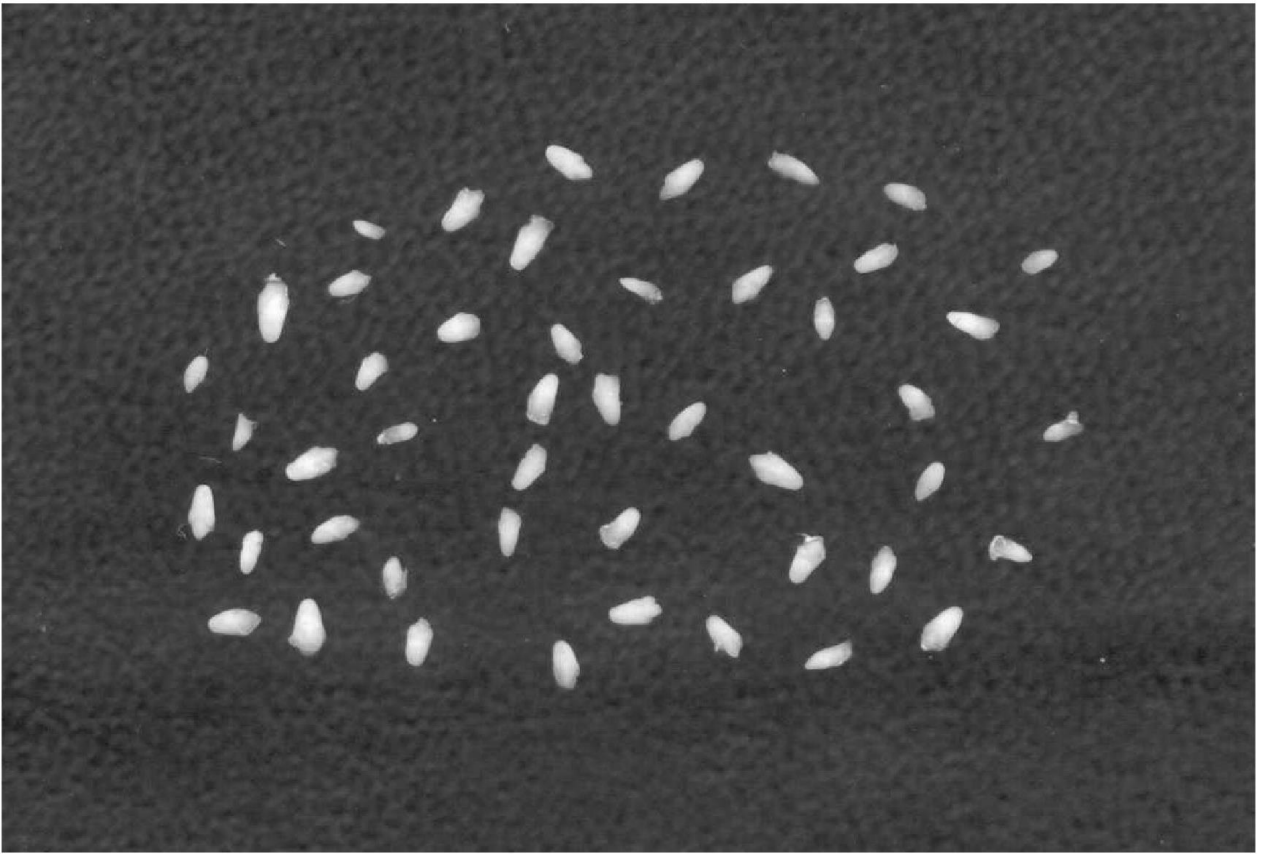


Binarized image after morphological processing

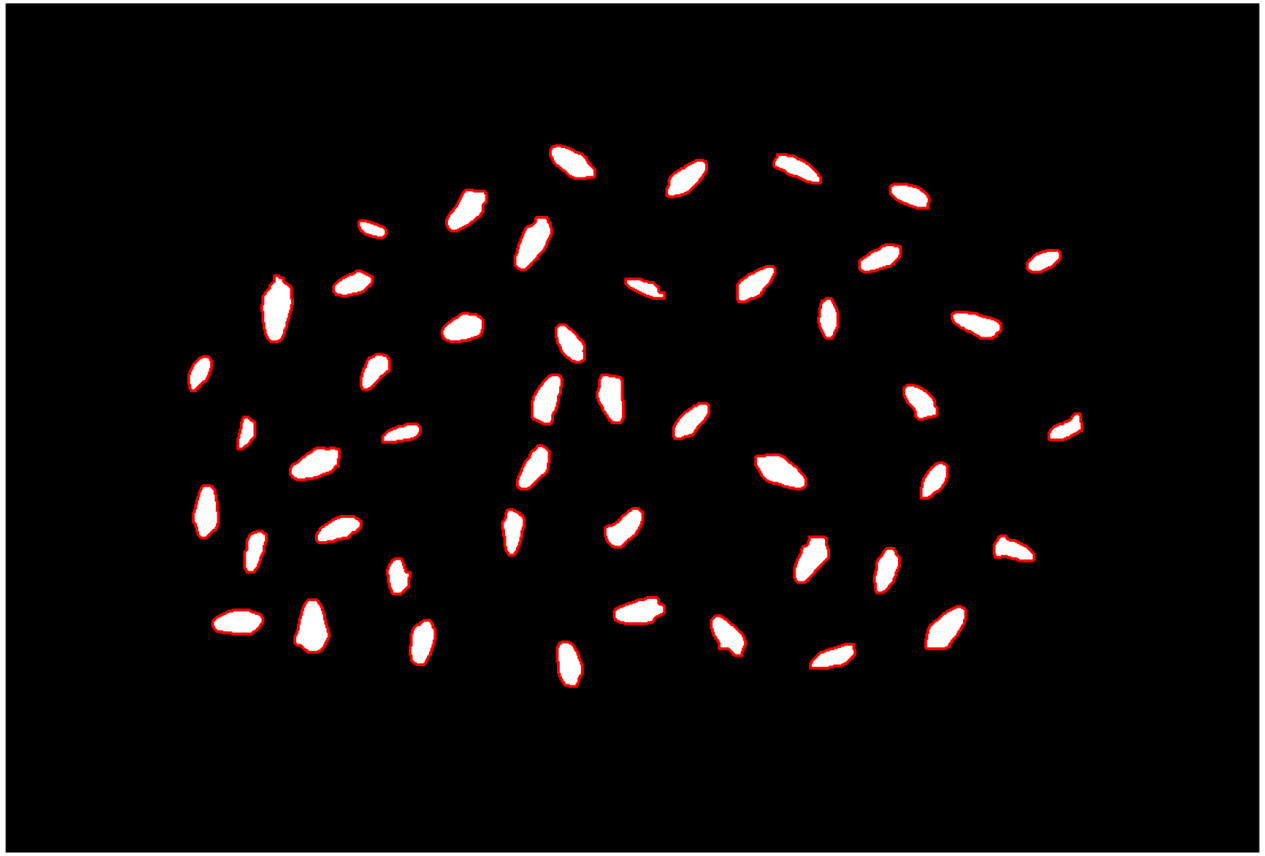


II.1, II.2 and II.3 (Results for em2.jpg)

Original image

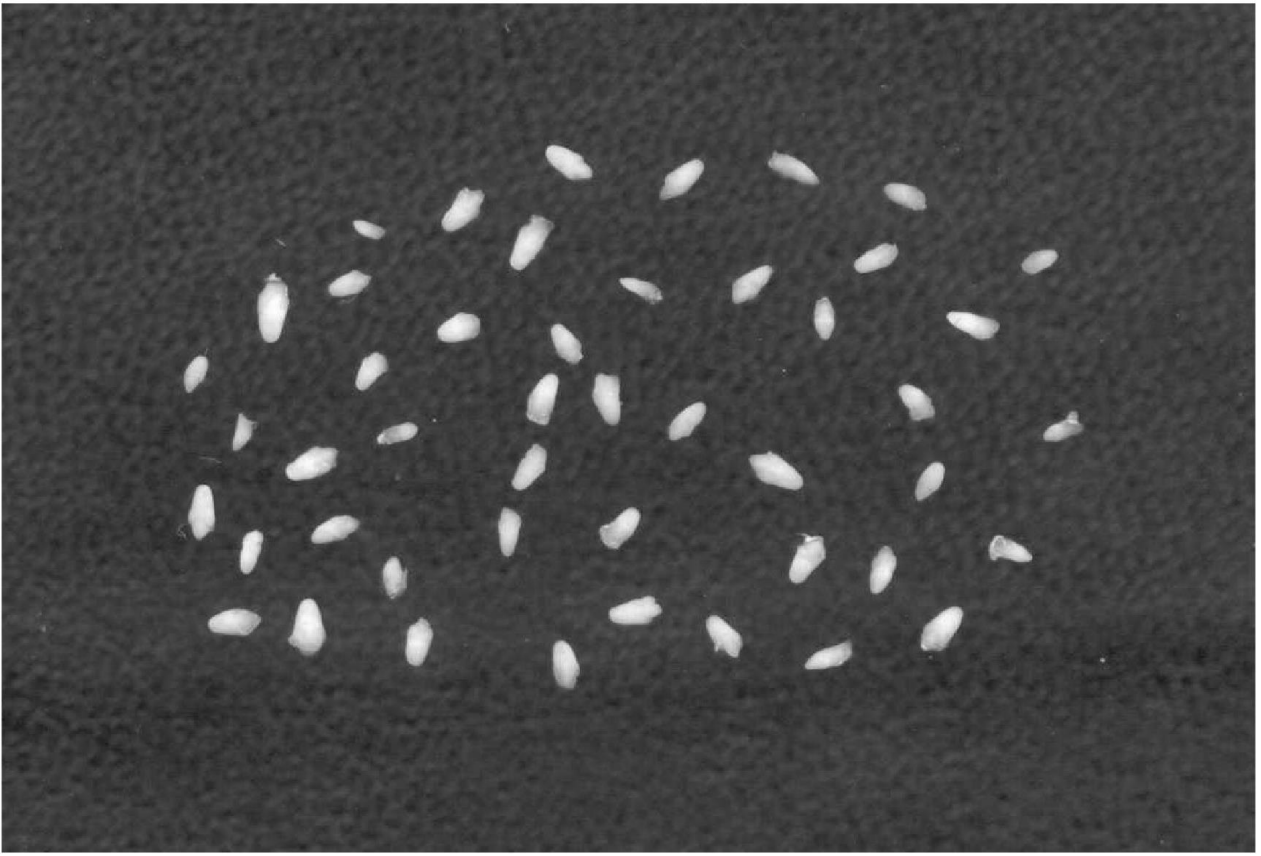


Boundaries marked

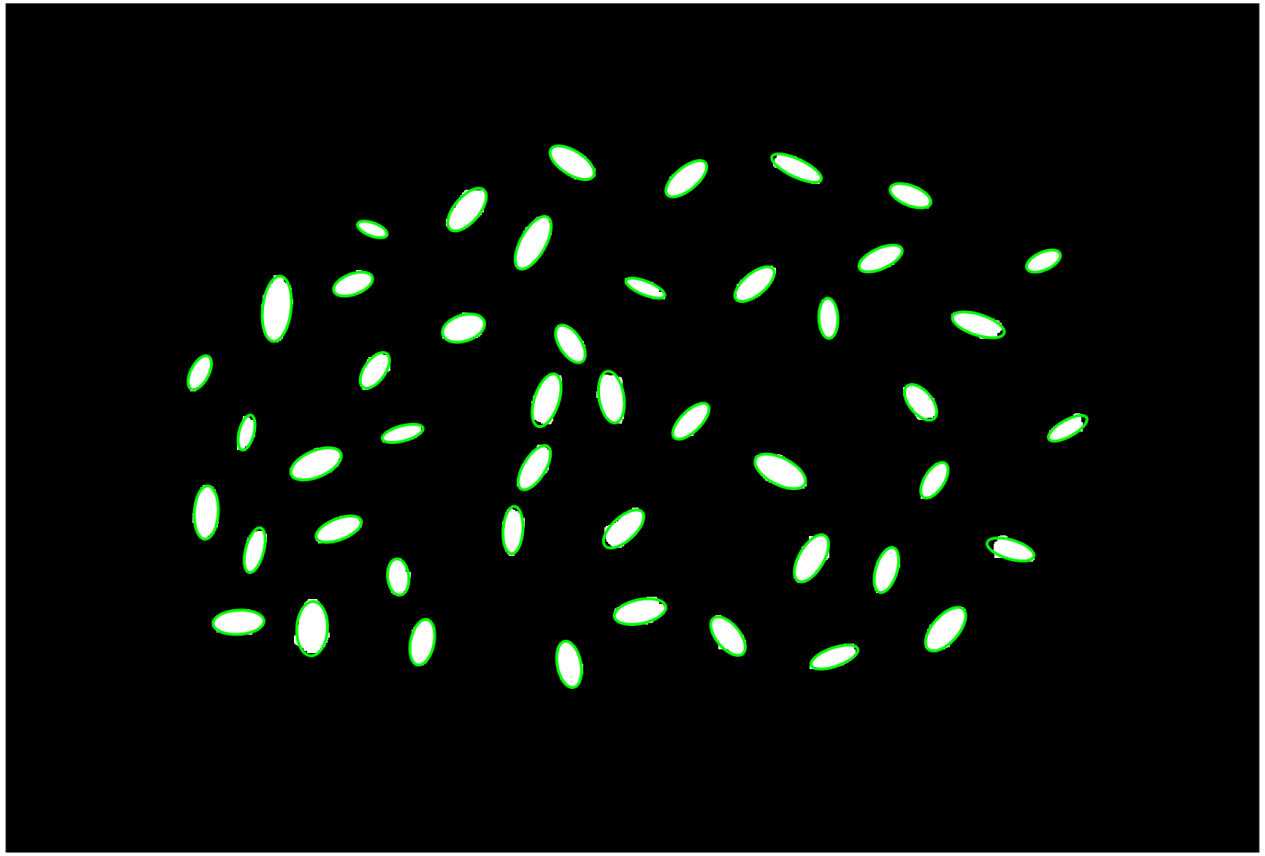


II.1, II.2 and II.3 (Results for em2.jpg)

Original image



Ellipse approximation



II.1, II.2 and II.3(Results for em2.jpg)

Average area in terms of ellipse approximation = 627.11

```
num =

    47


area =

   1.0e+03 *

  Columns 1 through 13

    0.4022    0.7701    0.7434    0.3220    0.5024    1.1242    0.8487    1.0067    0.6076    0.5070    0.2484    0.5217    0.3838

  Columns 14 through 26

    0.4645    0.6571    0.6840    0.7877    0.5663    0.8885    0.6653    0.8124    0.7147    0.5560    0.6691    0.7797    0.7101

  Columns 27 through 39

    0.7550    0.3388    0.6714    0.5770    0.6573    0.6272    0.8845    0.5849    0.7872    0.5507    0.4603    0.5706    0.6010

  Columns 40 through 47

    0.5141    0.5803    0.4772    0.8422    0.6762    0.5571    0.3896    0.4276


avg_area =

  627.1100
```

II.1, II.2 and II.3(Results for em2.jpg)

Average area in terms of pixel count=661.9574

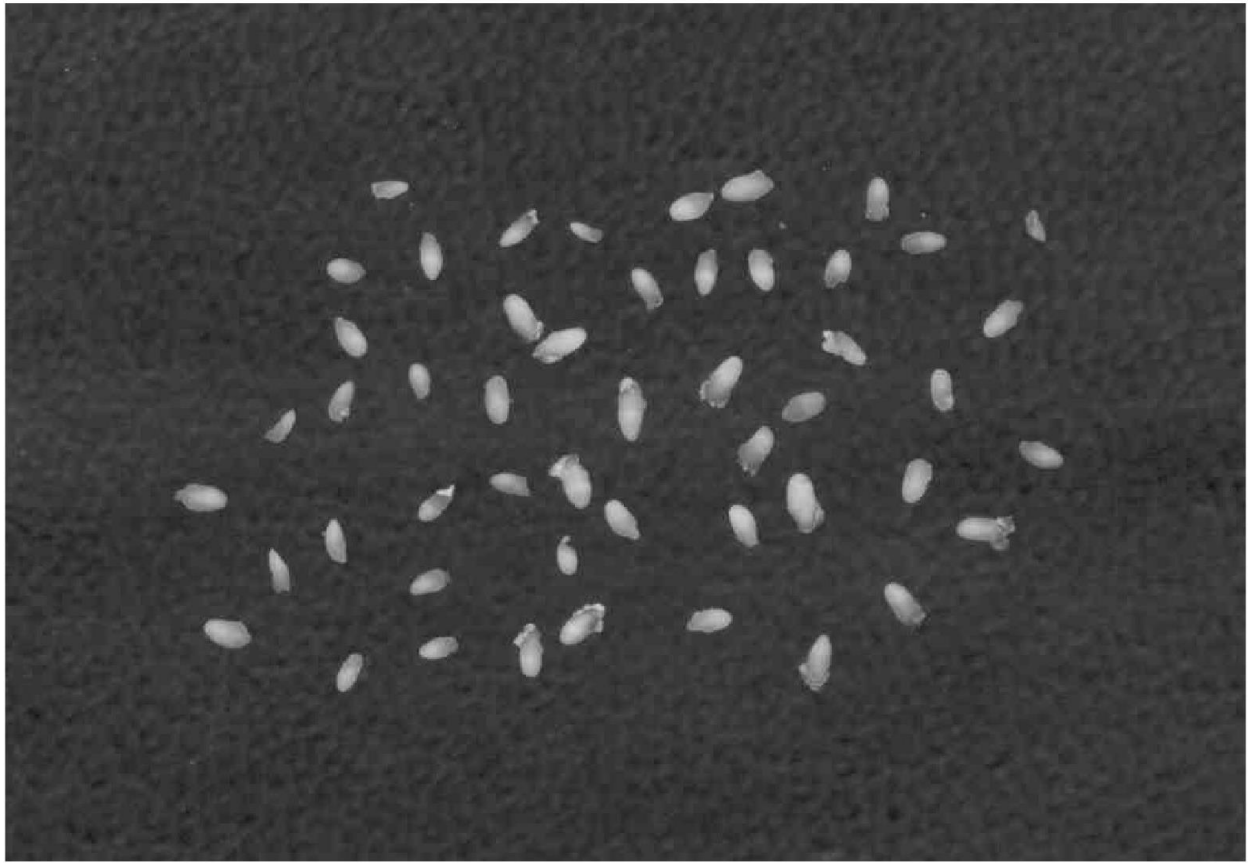
```
pixcount =

  Columns 1 through 10

         435         814         789         333         533        1180         895        1041         648         543

  Columns 11 through 20

         273         550         413         495         702         723         827         596         938         698

  Columns 21 through 30

         836         761         594         712         811         723         797         377         719         617

  Columns 31 through 40

         691         665         935         613         825         585         499         609         641         551

  Columns 41 through 47

         610         514         887         703         551         424         436


avg_area_pixel =

  661.9574
```

II.1, II.2 and II.3(Results for em3.jpg)

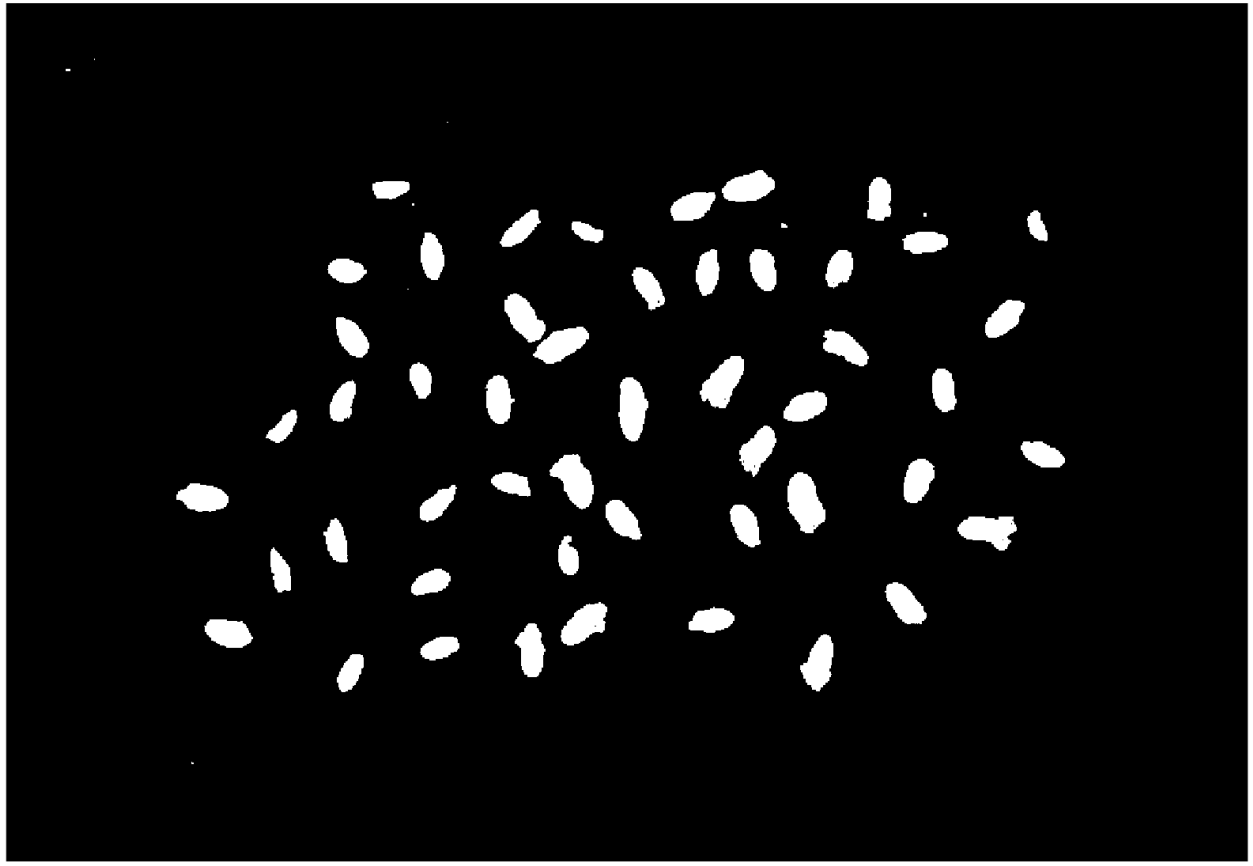
For best result, Threshold for binarizing: 0.35, structural element size = 3 (E = strel("disk",3);)
Number of embryos = number of connected components = 50

II.1, II.2 and II.3 (Results for em3.jpg)

Original image

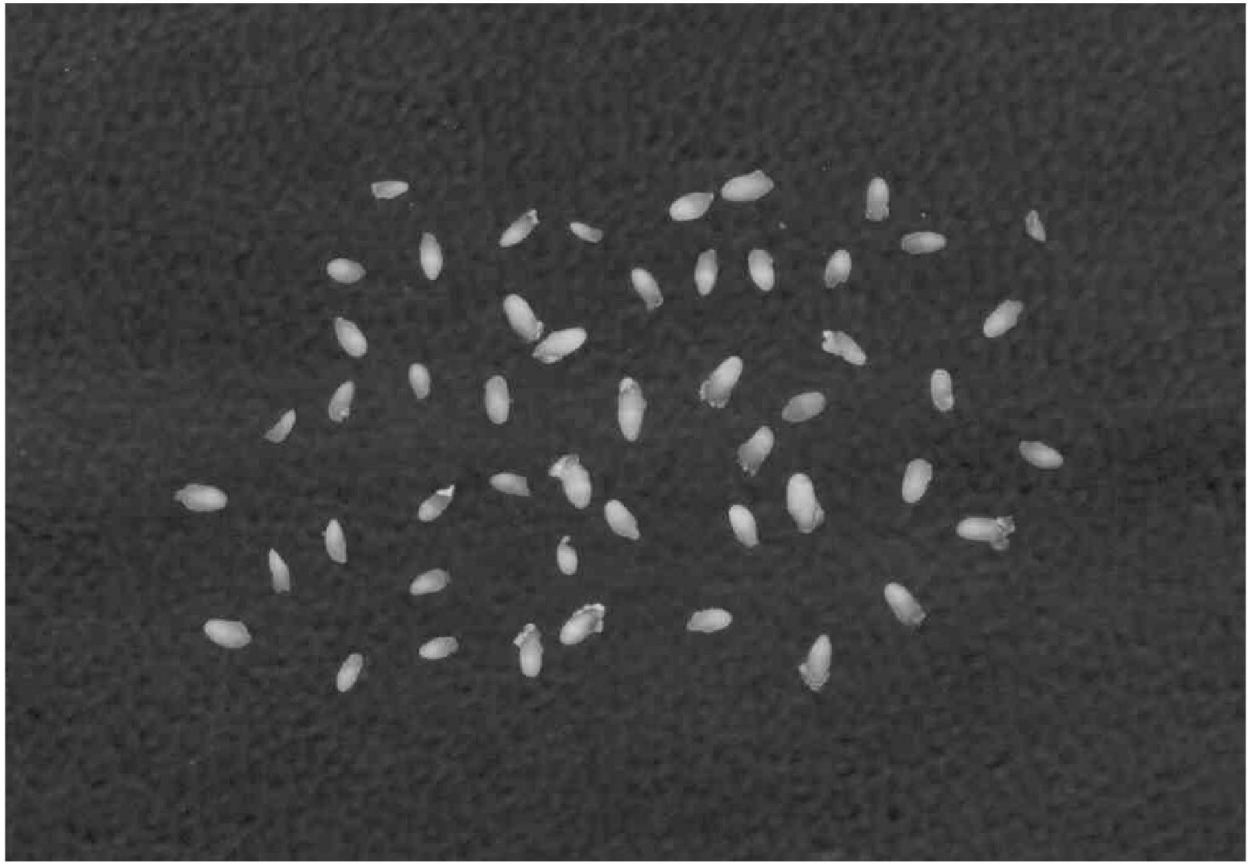


Binarized image

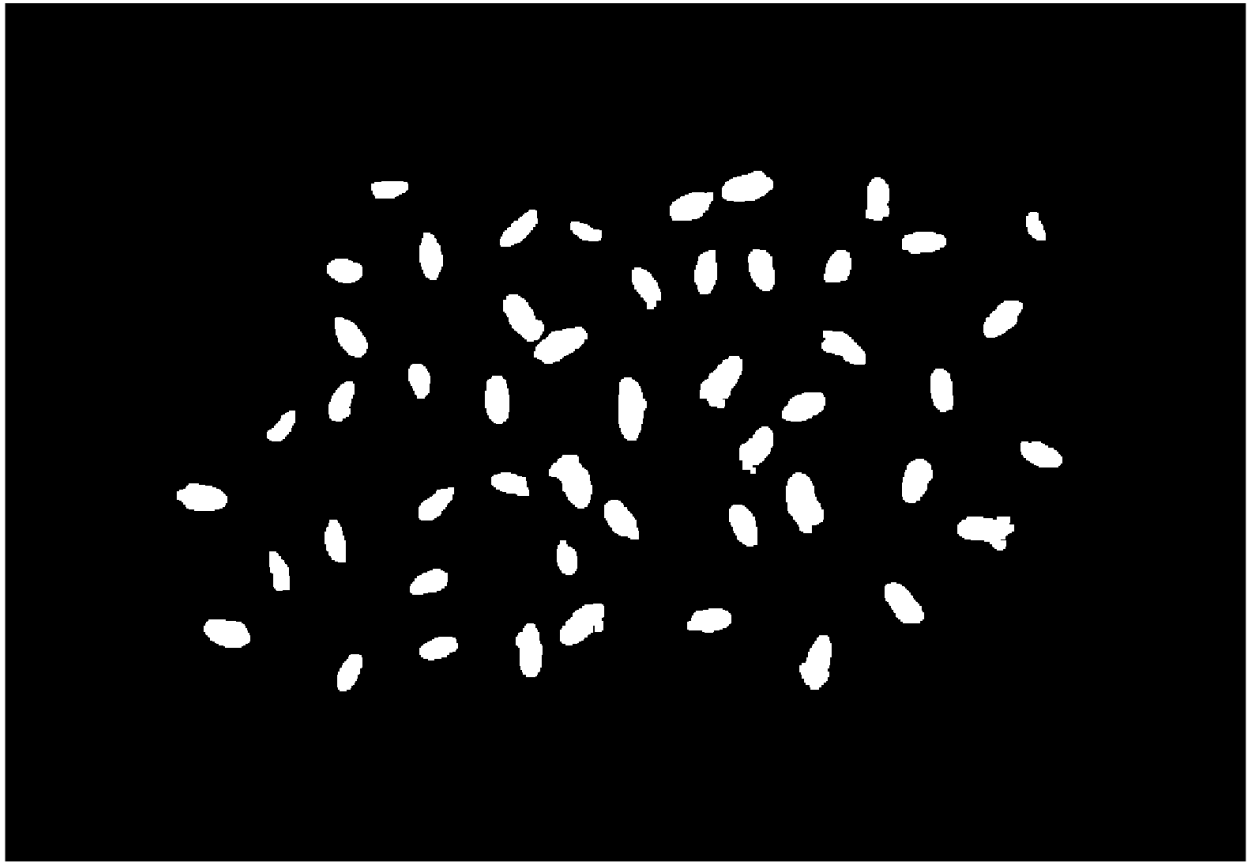


II.1, II.2 and II.3 (Results for em3.jpg)

Original image

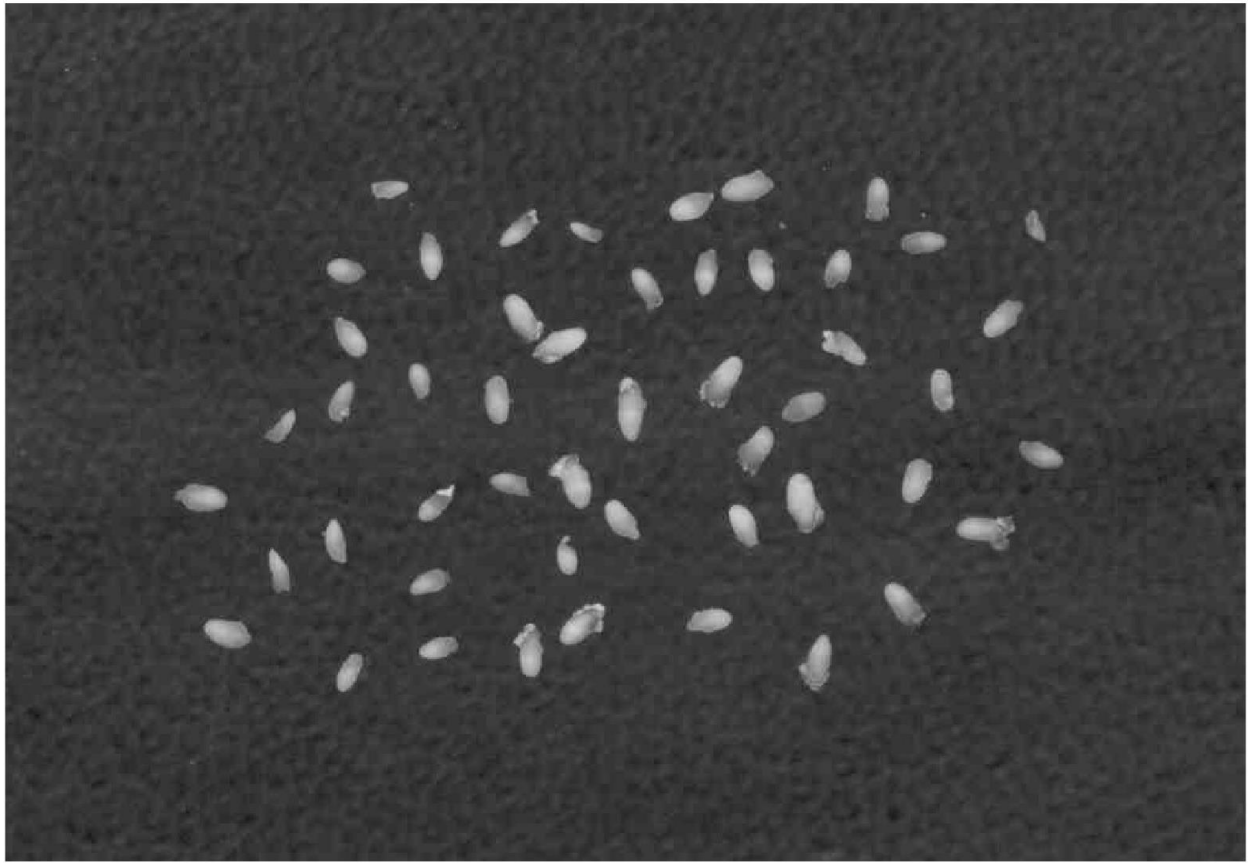


Binarized image after morphological processing

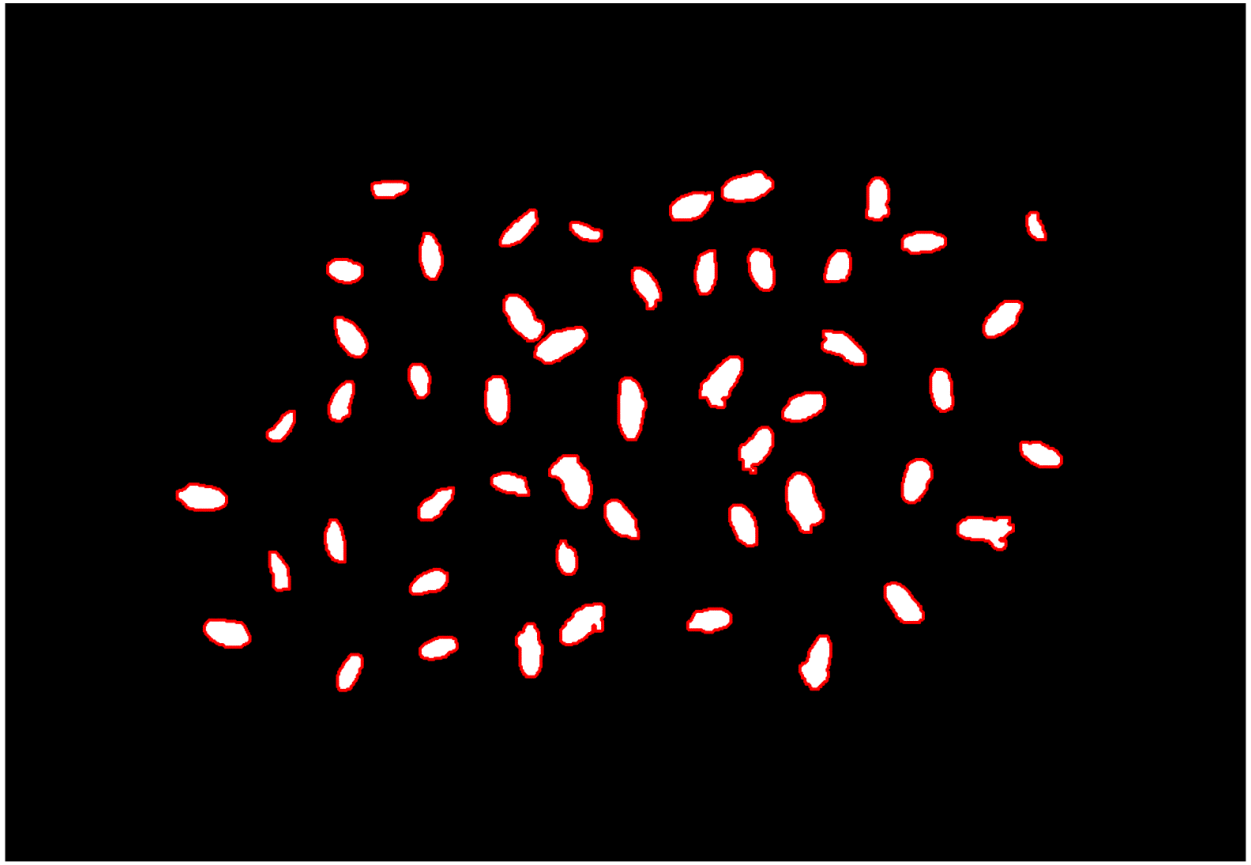


II.1, II.2 and II.3 (Results for em3.jpg)

Original image

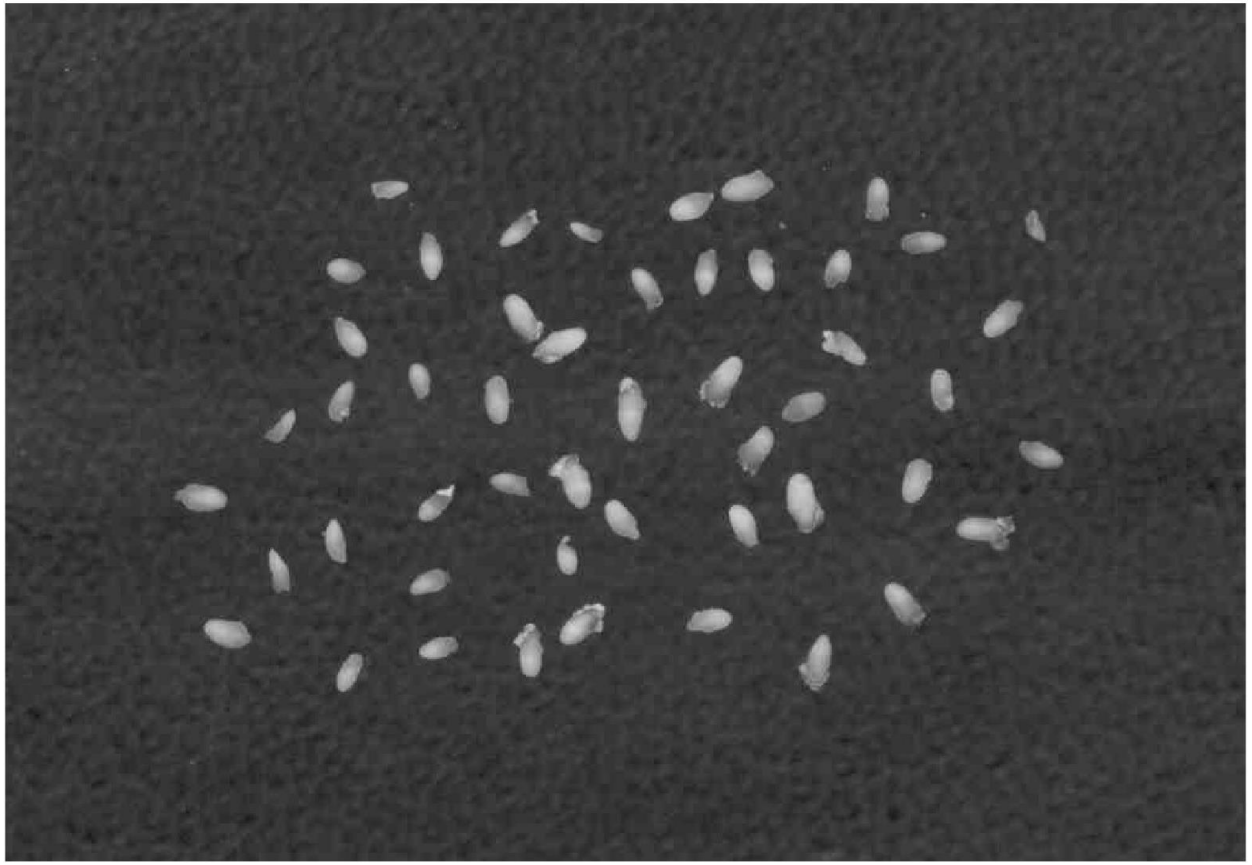


Boundaries marked

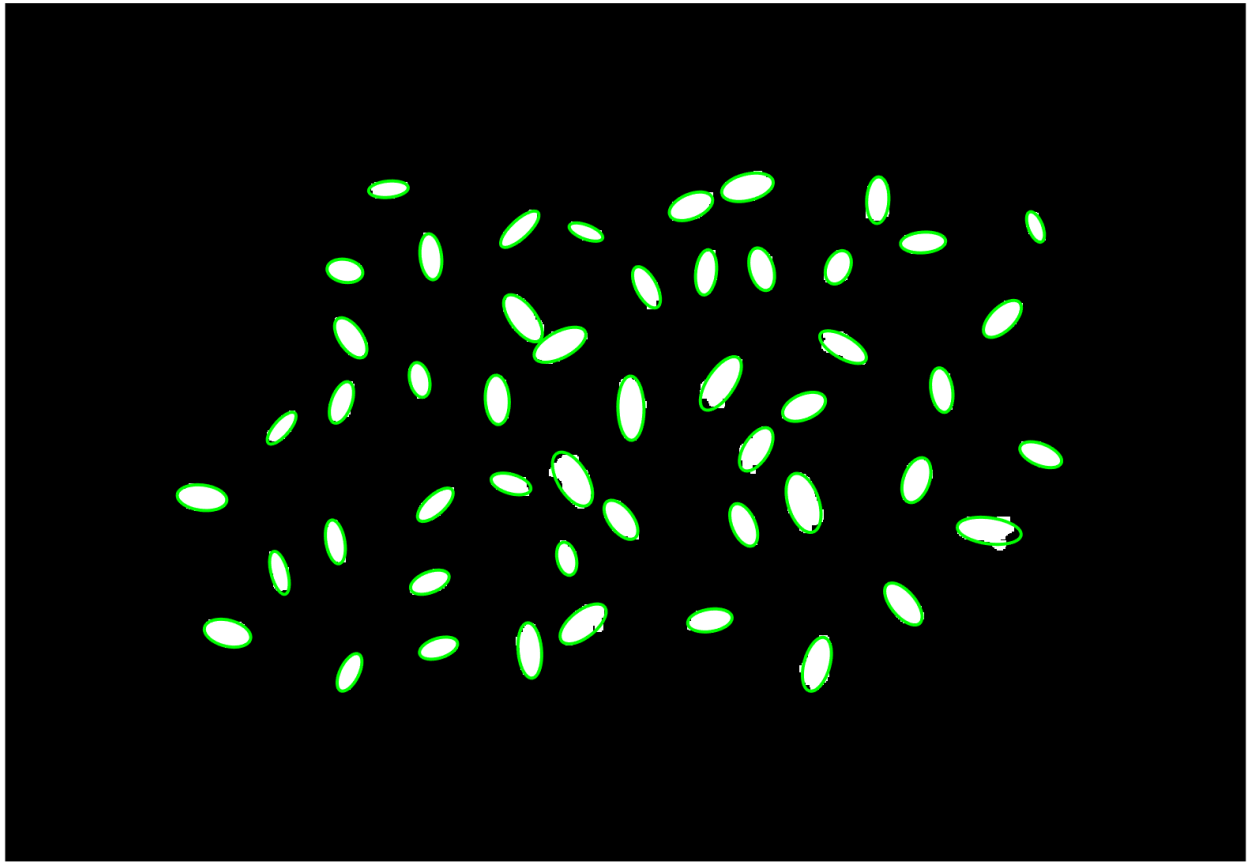


II.1, II.2 and II.3 (Results for em3.jpg)

Original image



Ellipse approximation



II.1, II.2 and II.3(Results for em3.jpg)

Average area in terms of ellipse approximation = 553.0282

```
num =

    50


area =

  Columns 1 through 13

  648.6203  639.2063  317.8572  377.2086  428.1041  425.6058  456.5791  549.5527  396.6888  332.8130  358.7587  431.4529  442.9092

  Columns 14 through 26

  510.2340  400.6314  608.2873  404.1117  458.8203  727.5094  670.0746  764.2800  931.4037  332.1888  749.9546  256.4686  560.7717

  Columns 27 through 39

  863.0504  478.7292  576.4929  513.5303  485.0168  863.8474  715.2541  541.3510  626.0824  535.9832  583.8213  978.0436  736.9673

  Columns 40 through 50

  597.6719  426.8664  531.3125  641.4502  480.5636  625.6301  500.5749  863.8029  577.3286  489.8754  238.0712


avg_area =

  553.0282
```

II.1, II.2 and II.3(Results for em3.jpg)

Average area in terms of pixel count= 585.2

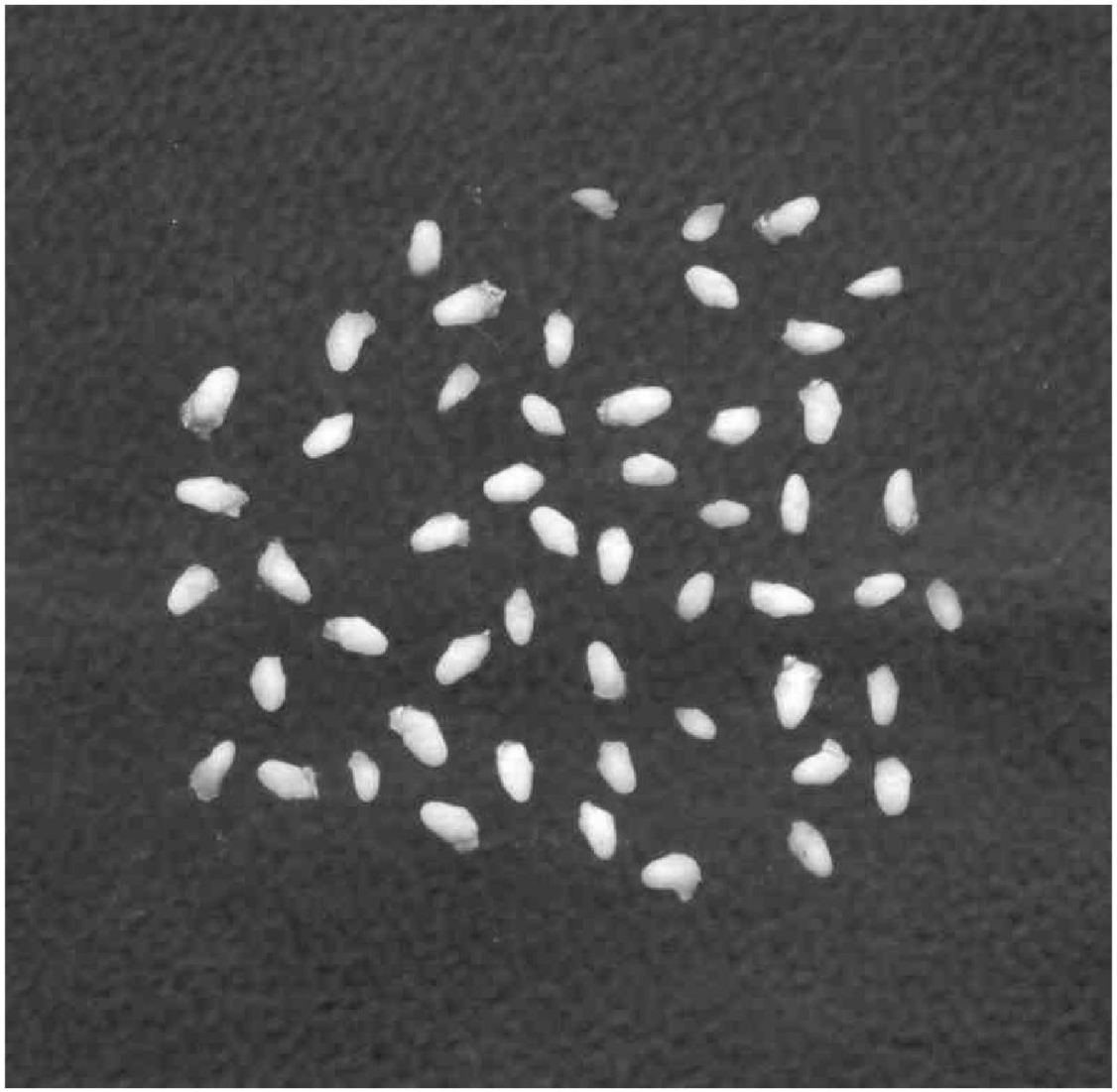
```
pixcount =

  Columns 1 through 10

         692         681         332         397         460         459         484         584         430         354

  Columns 11 through 20

         388         465         473         550         436         650         430         493         765         709

  Columns 21 through 30

         809         936         362         803         280         597         910         504         614         551

  Columns 31 through 40

         520         870         759         580         655         573         625        1021         766         629

  Columns 41 through 50

         458         557         678         520         665         537         844         618         529         258


avg_area_pixel =

  585.2000
```

II.1, II.2 and II.3(Results for em4.jpg)

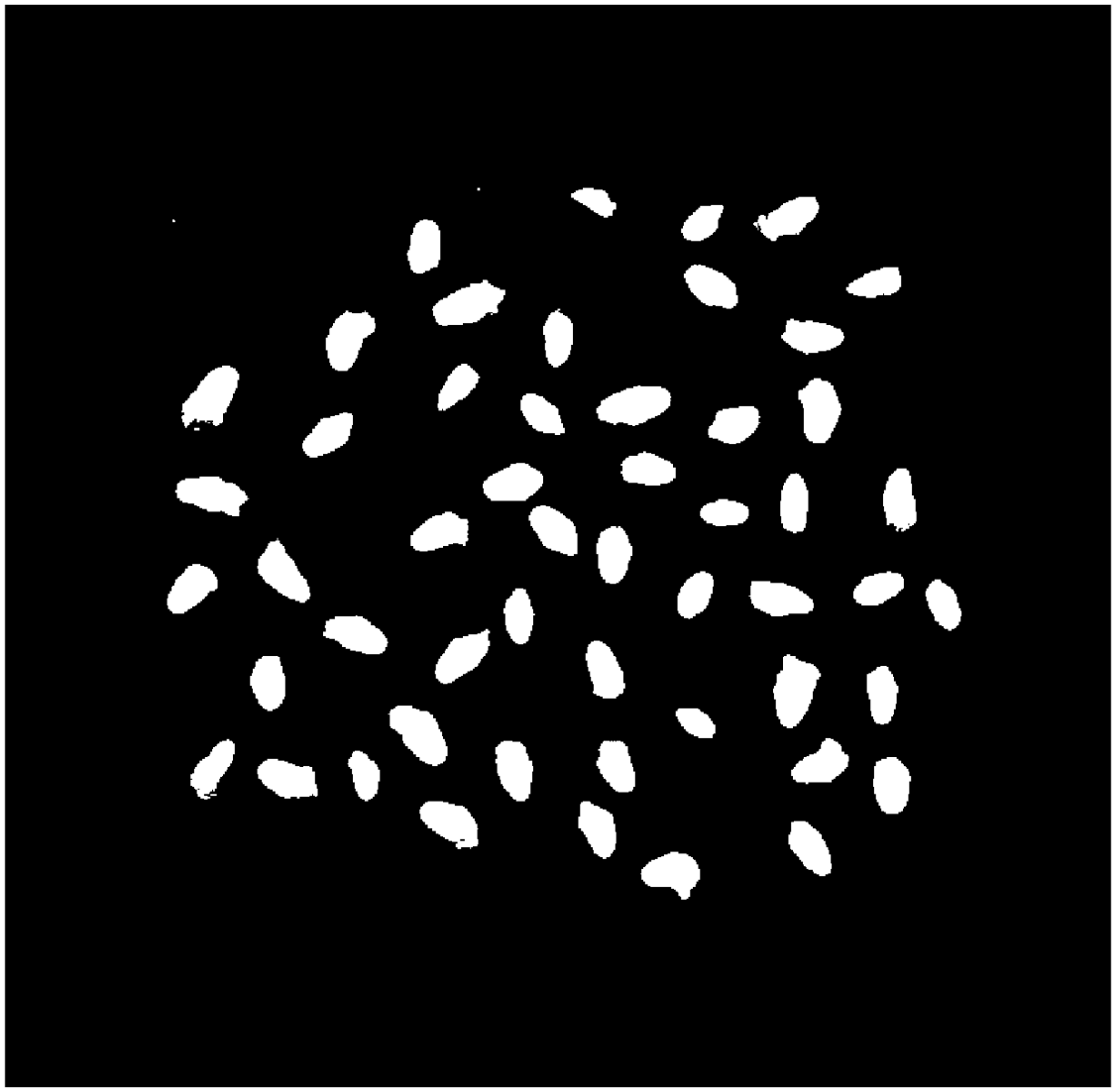
For best result, Threshold for binarizing: 0.5, structural element size = 1 (E = strel("disk",1);)
Number of embryos = number of connected components = 52

II.1, II.2 and II.3 (Results for em4.jpg)

Original image

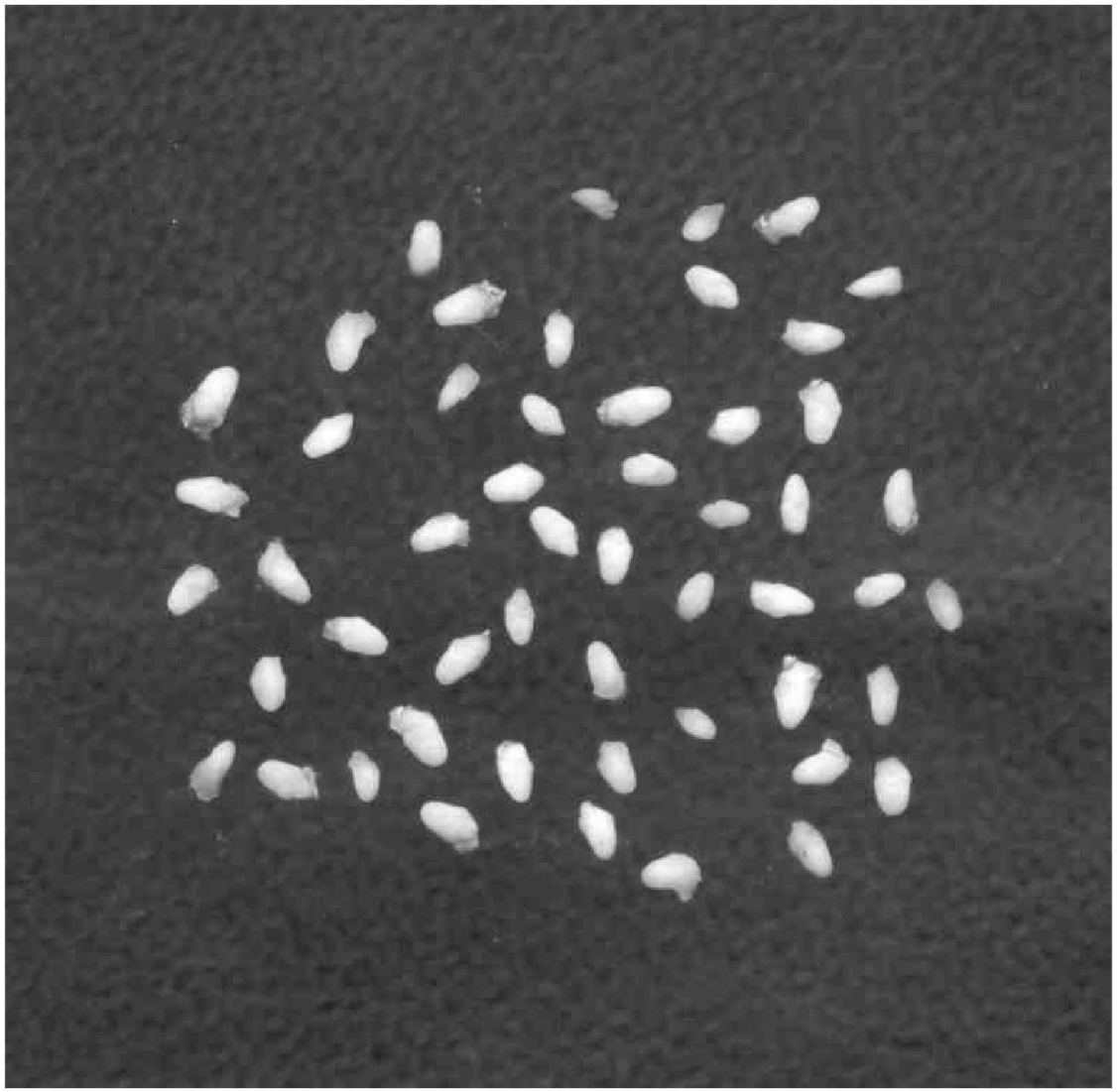


Binarized image

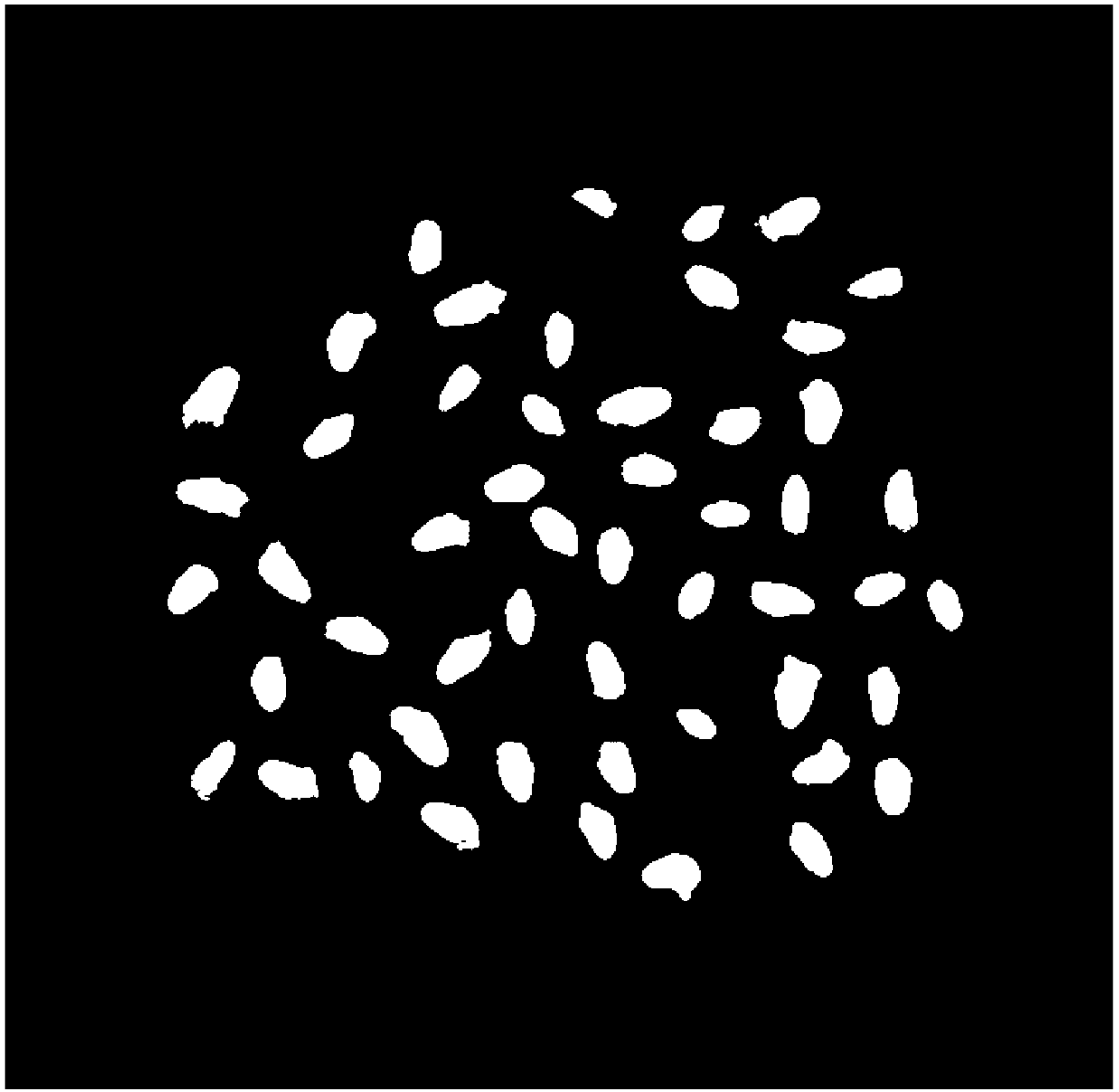


II.1, II.2 and II.3 (Results for em4.jpg)

Original image

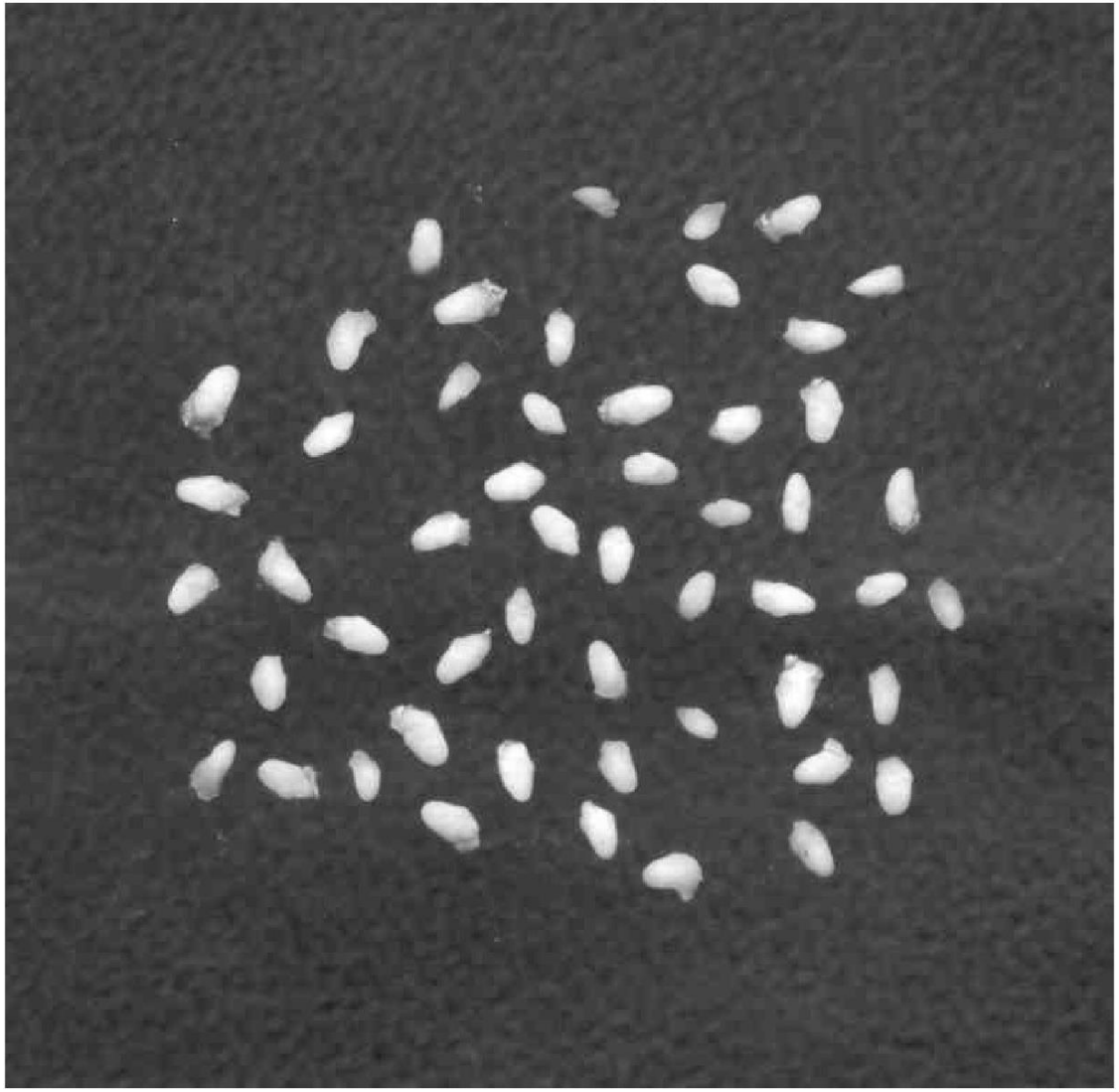


Binarized image after morphological processing

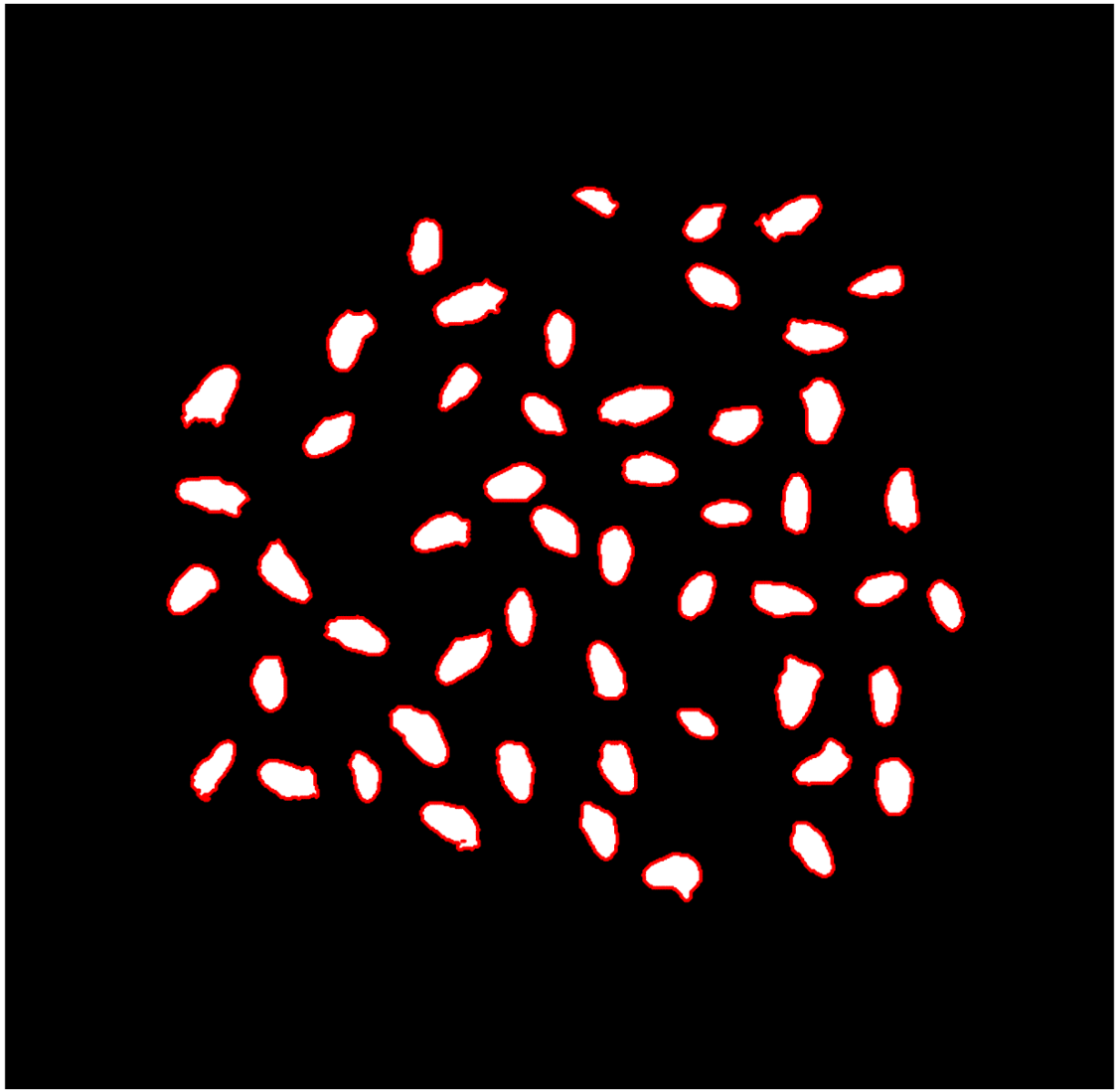


II.1, II.2 and II.3 (Results for em4.jpg)

Original image

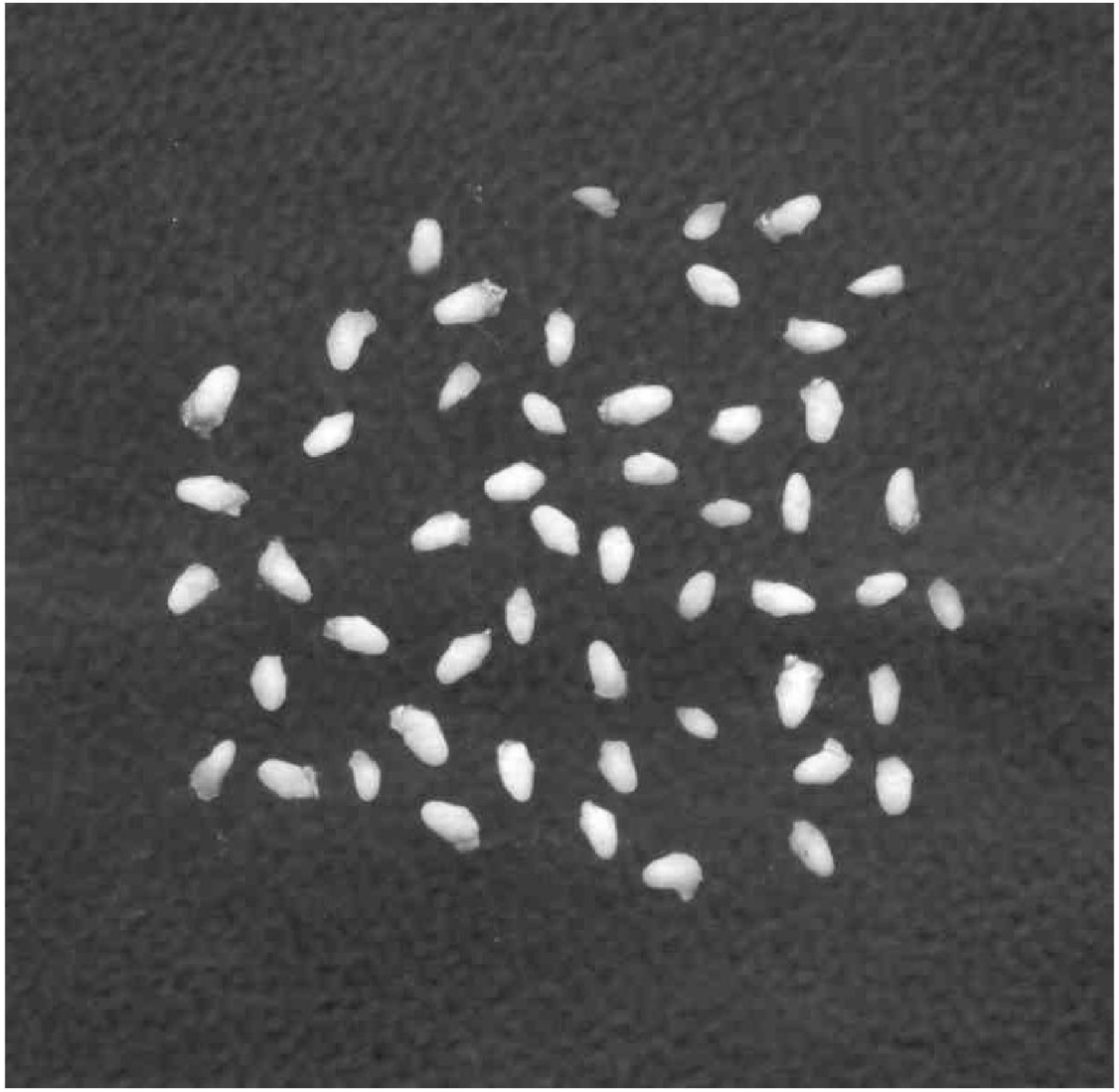


Boundaries marked

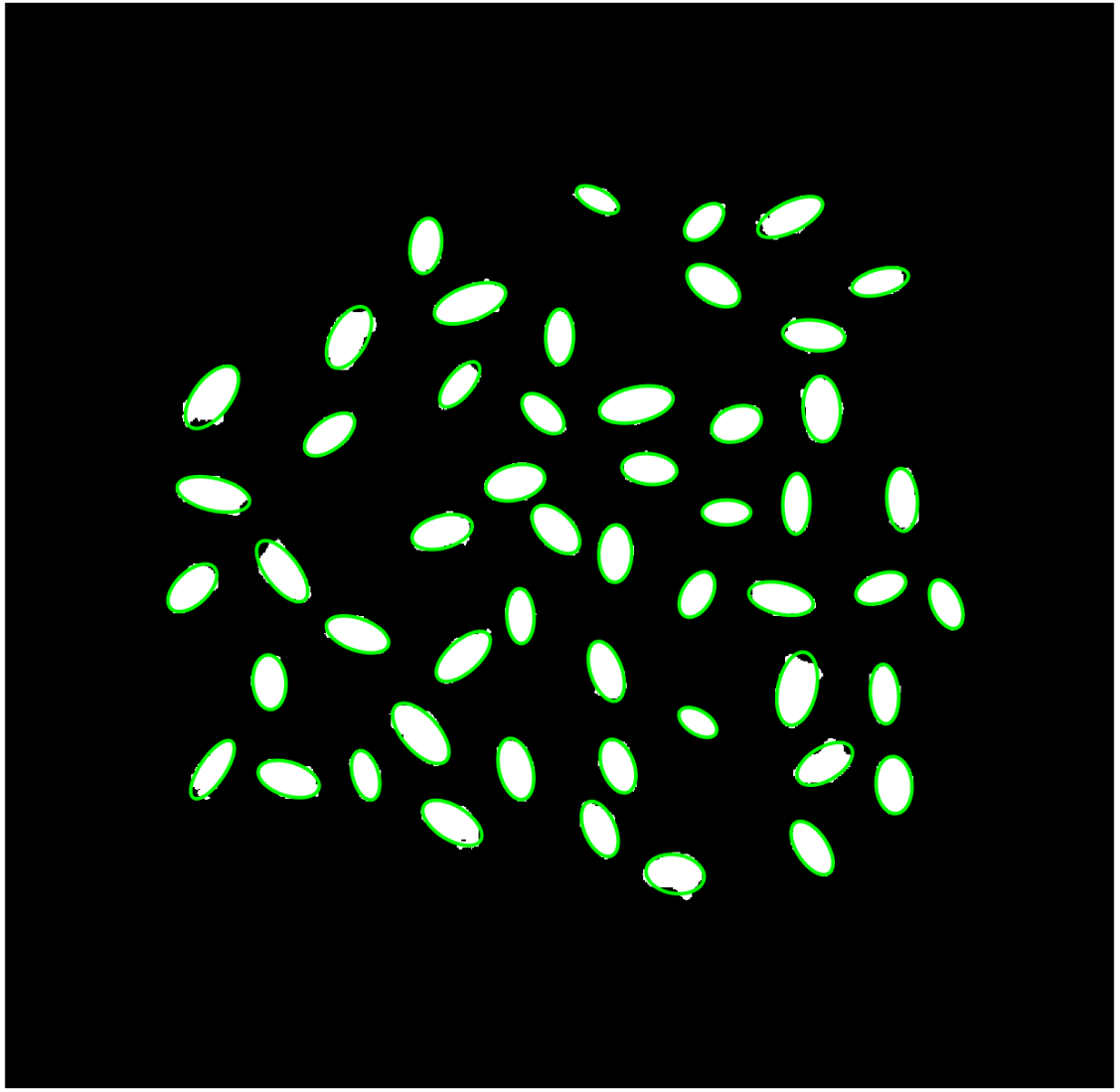


II.1, II.2 and II.3(Results for em4.jpg)

Original image



Ellipse approximation



II.1, II.2 and II.3 (Results for em4.jpg)

Average area in terms of ellipse approximation = 743.0014

```
num =

    52


area =

   1.0e+03 *

  Columns 1 through 13

    0.7717    0.9356    1.0638    0.6824    0.7096    0.9308    0.8325    0.6865    0.8145    0.9493    0.5162    1.0582    0.6796

  Columns 14 through 26

    0.7830    0.8774    1.0022    0.8262    0.5343    0.8183    0.8213    0.5890    0.5605    0.7760    0.6069    0.3710    0.7142

  Columns 27 through 39

    0.7675    1.0173    0.7435    0.7022    0.6606    0.8924    0.3858    0.5752    0.4850    0.7605    0.4721    0.6896    0.8021

  Columns 40 through 52

    0.8191    1.1237    0.6457    0.7421    0.7289    0.8029    0.9600    0.5727    0.5781    0.6608    0.8068    0.7536    0.5766


avg_area =

  743.0014
```

II.1, II.2 and II.3 (Results for em4.jpg)

Average area in terms of pixel count= 778.0577

```
pixcount =

  Columns 1 through 10

         801         971        1081         700         749         924         875         731         854         975

  Columns 11 through 20

         546        1102         720         810         929        1047         871         550         861         866

  Columns 21 through 30

         629         601         822         644         389         754         803        1074         788         742

  Columns 31 through 40

         703         908         418         615         517         807         510         730         847         829

  Columns 41 through 50

        1142         688         781         774         817         996         597         618         698         853

  Columns 51 through 52

         789         613


avg_area_pixel =

  778.0577
```

II.4 (MATLAB code)

PCA shape alignment has been performed to align all the boundaries of the embryos.

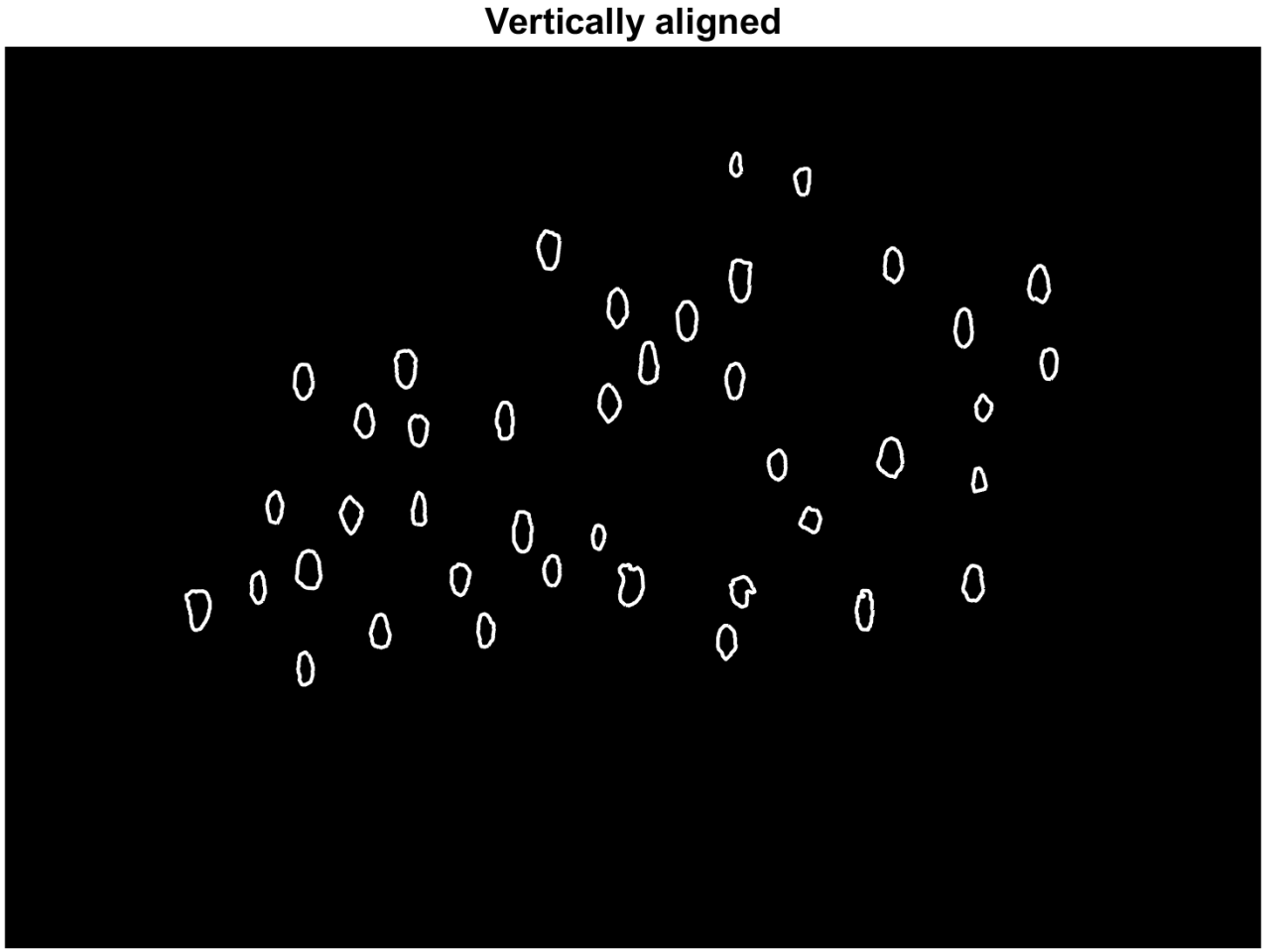
```
clc; clear; close all; f = 20;
I = imread("em1embryo.bmp");
s = size(I);
J = imbinarize(I,0.5);
E = strel("disk",3);
B = imerode(J,E);C = imdilate(B,E);
[L,num]=bwlabel(C,8);
B = bwboundaries(C);
Bm = cell2mat(B);
boundary_im = zeros(s(1),s(2));

figure(2);imshow(boundary_im);hold on;
for ii=1:num
    A = zeros(s); A(find(L==ii))=1;
    B = bwboundaries(A);
    Bm = cell2mat(B);
    x = Bm(:,2); y = Bm(:,1);
    xm = mean(x); ym = mean(y);
    cm = [xm, ym]; %centroid
    c = [x, y];
    cc = c - cm; %divergence
    Cx = cov(cc); %covariance
    [V, Cy] = eig(Cx);
    AA = V';
    Y = (AA * cc')';
    Y = Y+cm; %offset
    figure(2); plot(Y(:,1),Y(:,2),'w',LineWidth=2);title('Vertically aligned',fontsize=f);
end
```

II.4 (MATLAB code)

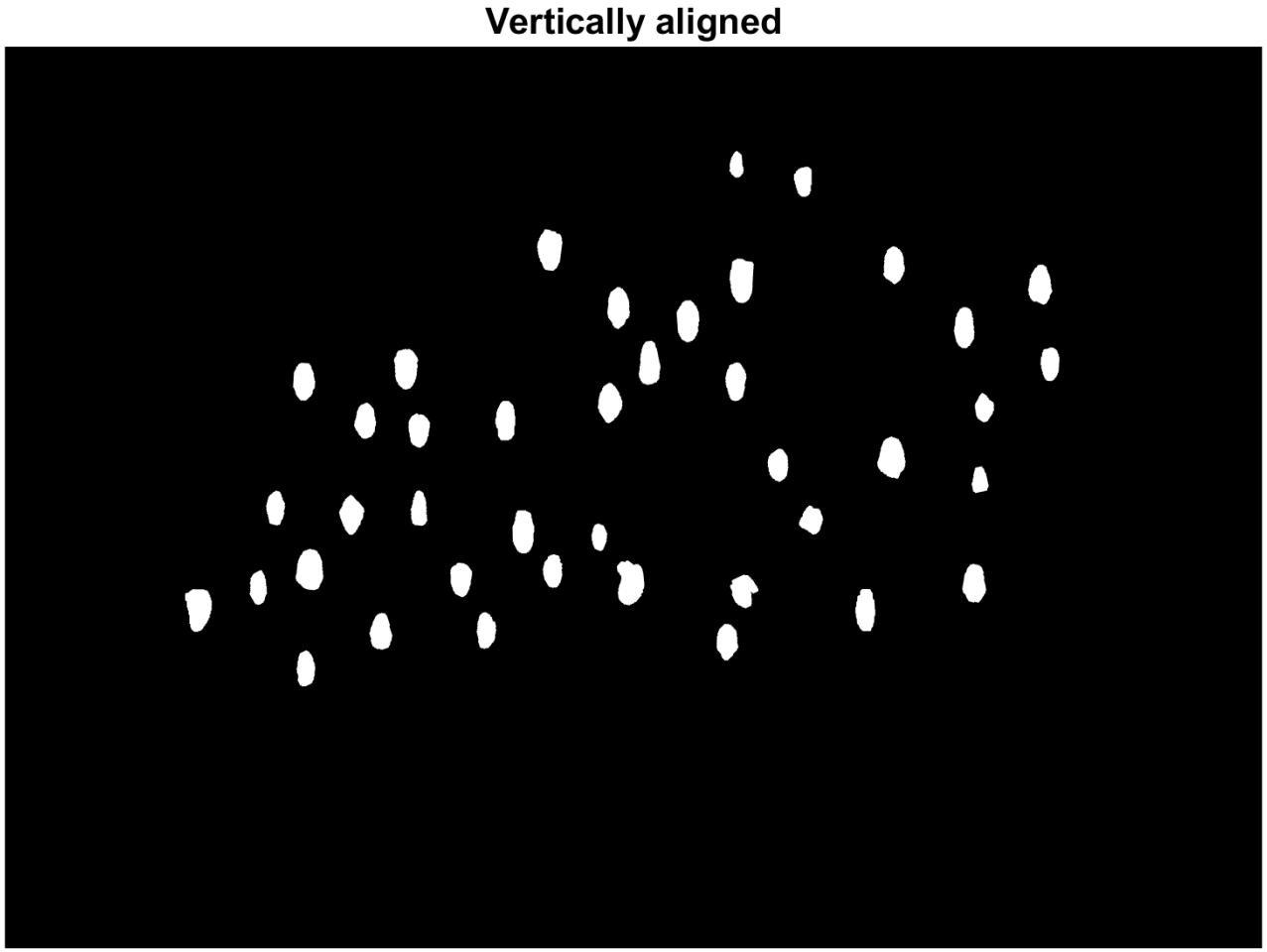
PCA shape alignment has been performed to align all the embryos (all pixels)

```
clc; clear; close all; f = 20;
I = imread("em1embryo.bmp");
s = size(I);
J = imbinarize(I,0.5);
E = strel("disk",3);
B = imerode(J,E);C = imdilate(B,E);
[L,num]=bwlabel(C,8);
B = bwboundaries(C);
Bm = cell2mat(B);
boundary_im = zeros(s(1),s(2));

figure(2);imshow(boundary_im);hold on;
for ii=1:num
    A = zeros(s); A(find(L==ii))=1;
    [xx yy] = find(L==ii);
    x = yy; y = xx;
    xm = mean(x); ym = mean(y);
    cm = [xm, ym]; %centroid
    c = [x, y];
    cc = c - cm; %divergence
    Cx = cov(cc); %covariance
    [V, Cy] = eig(Cx);
    AA = V';
    Y = (AA * cc')';
    Y = Y+cm; %offset
    figure(2); plot(Y(:,1),Y(:,2),'w',LineWidth=2);title('Vertically aligned',fontsize=f);
end
```

II.4 (em1.jpg)

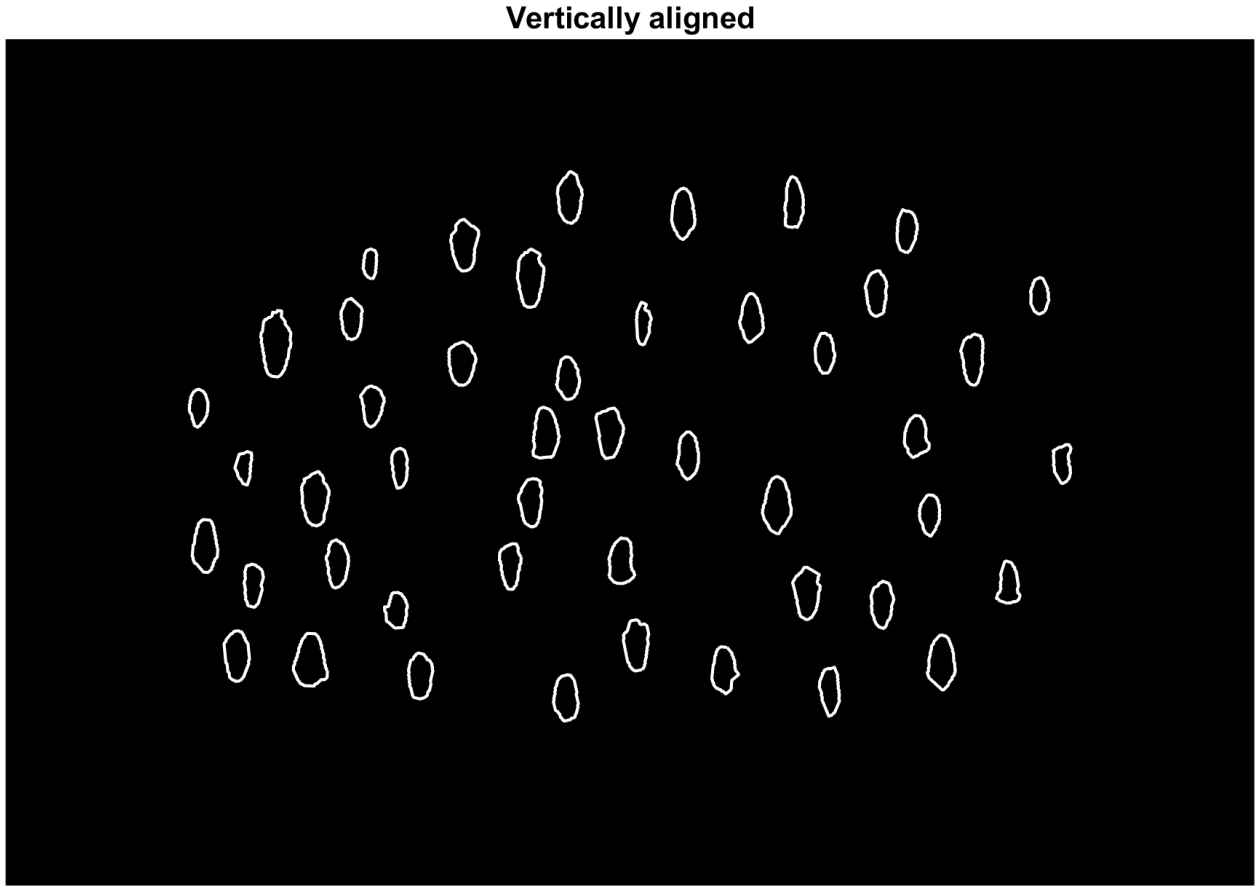


Boundaries

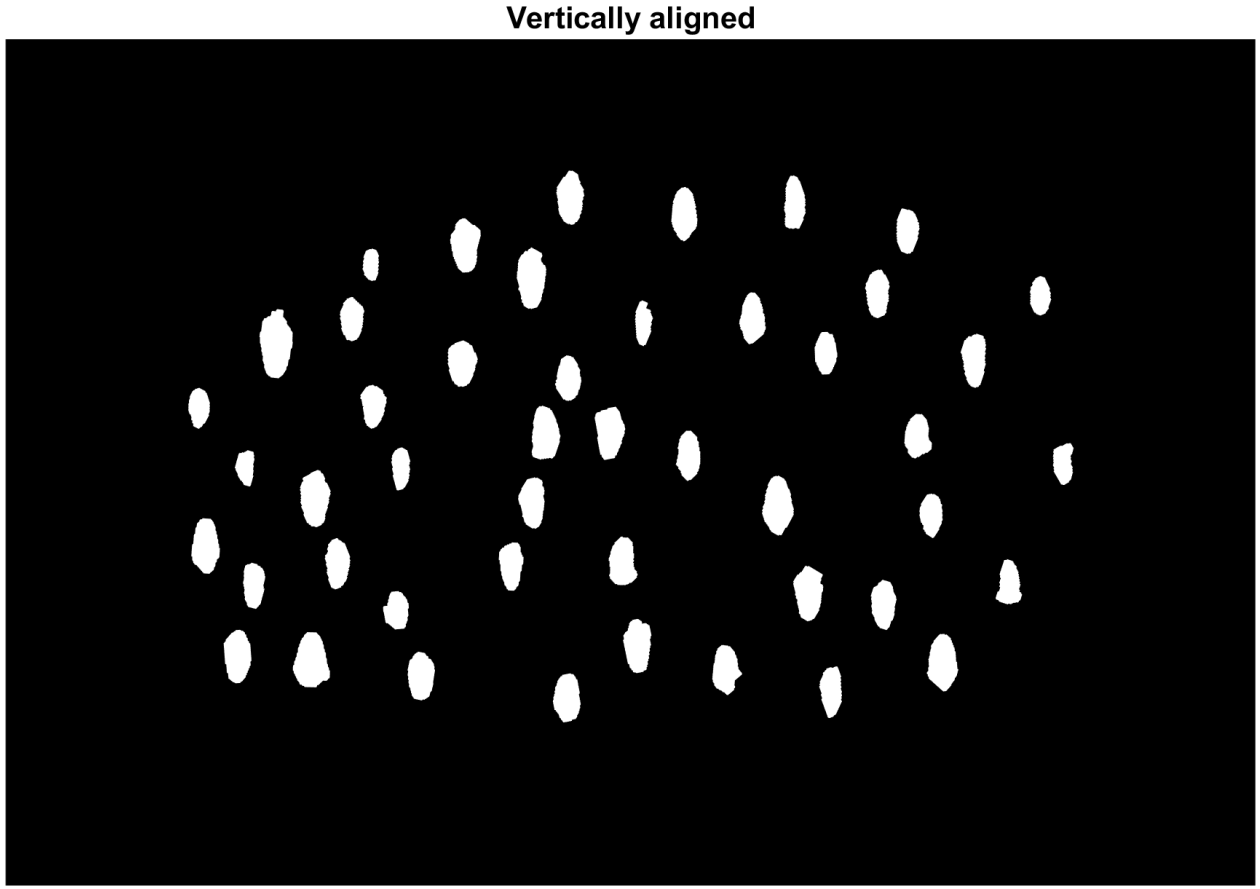


All pixels in embryos

II.4 (em2.jpg)

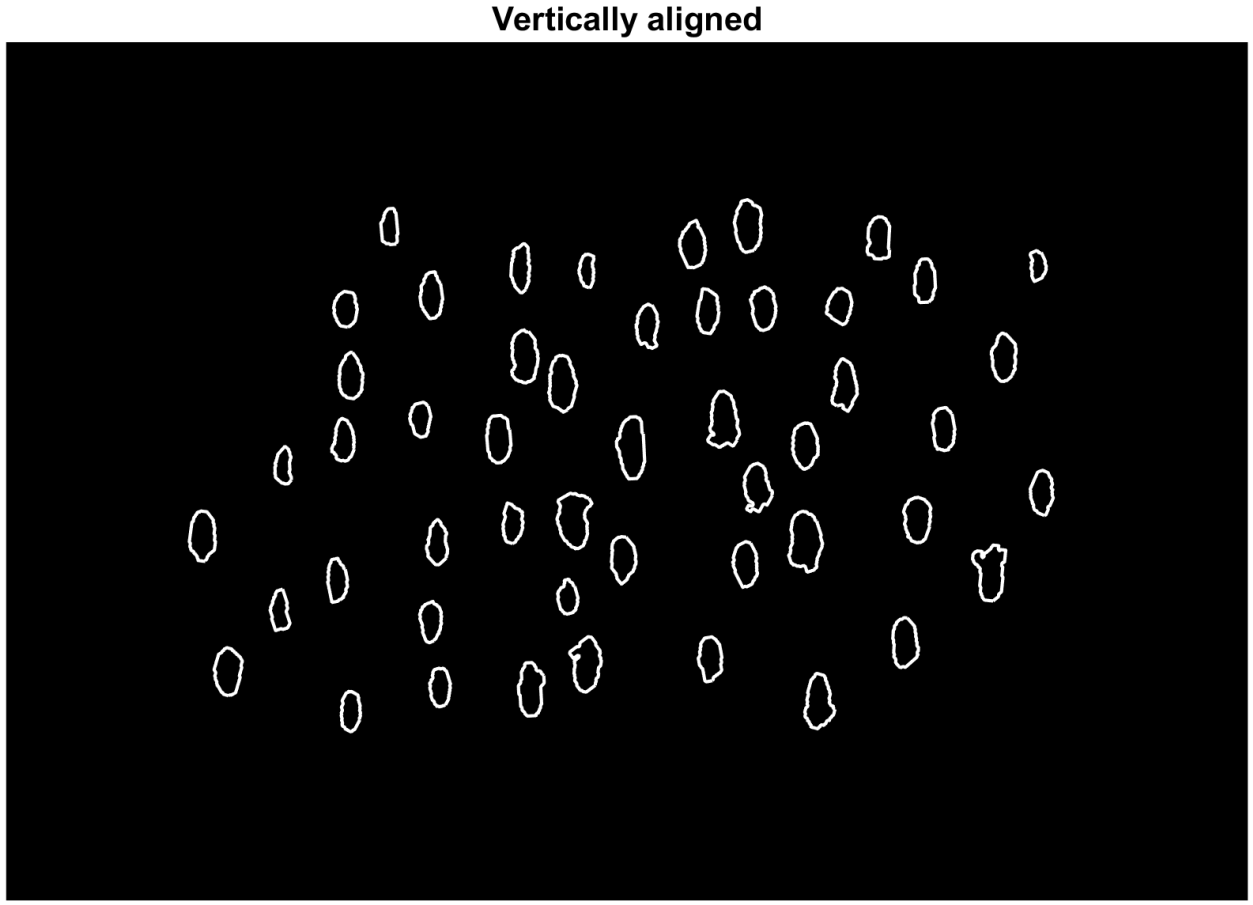


Boundaries

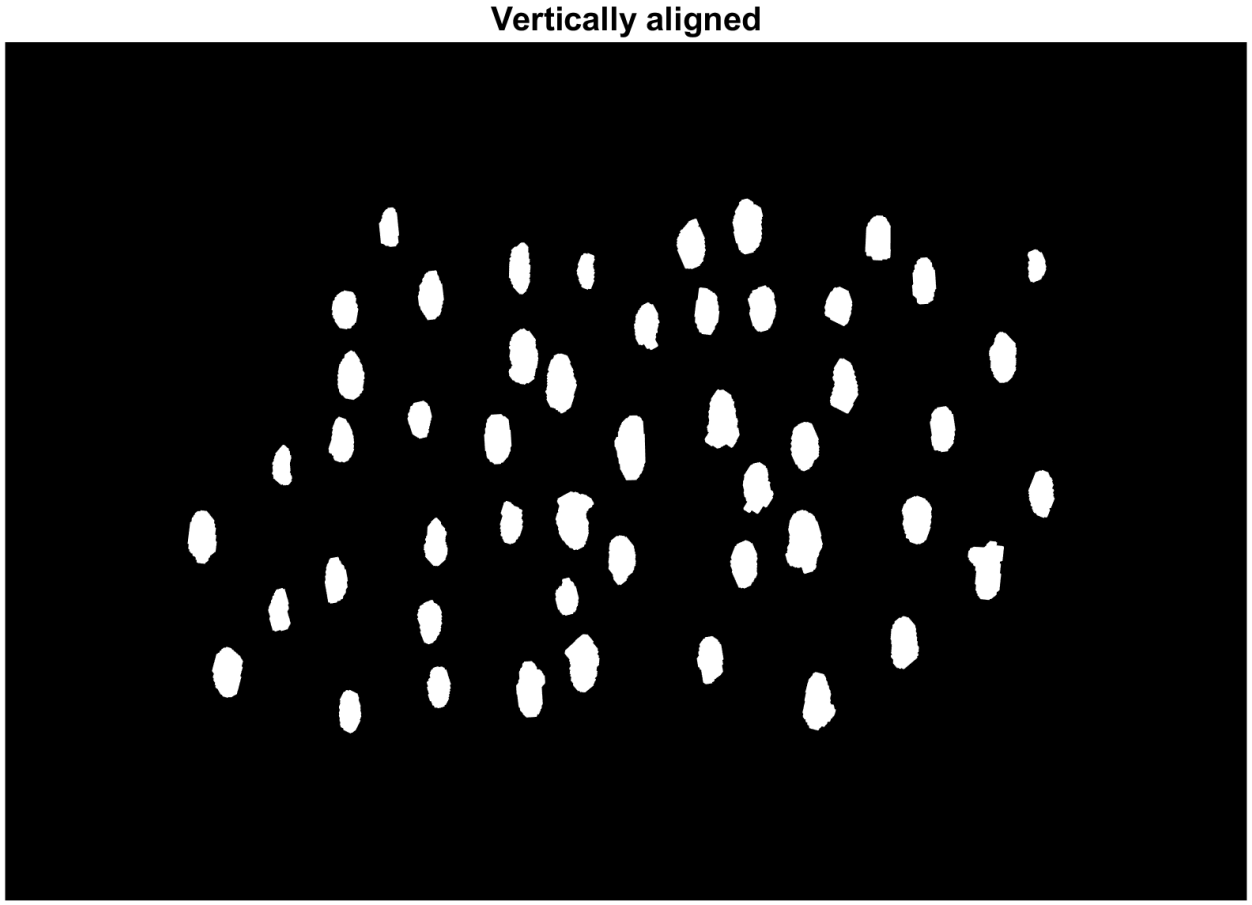


All pixels in embryos

II.4 (em3.jpg)

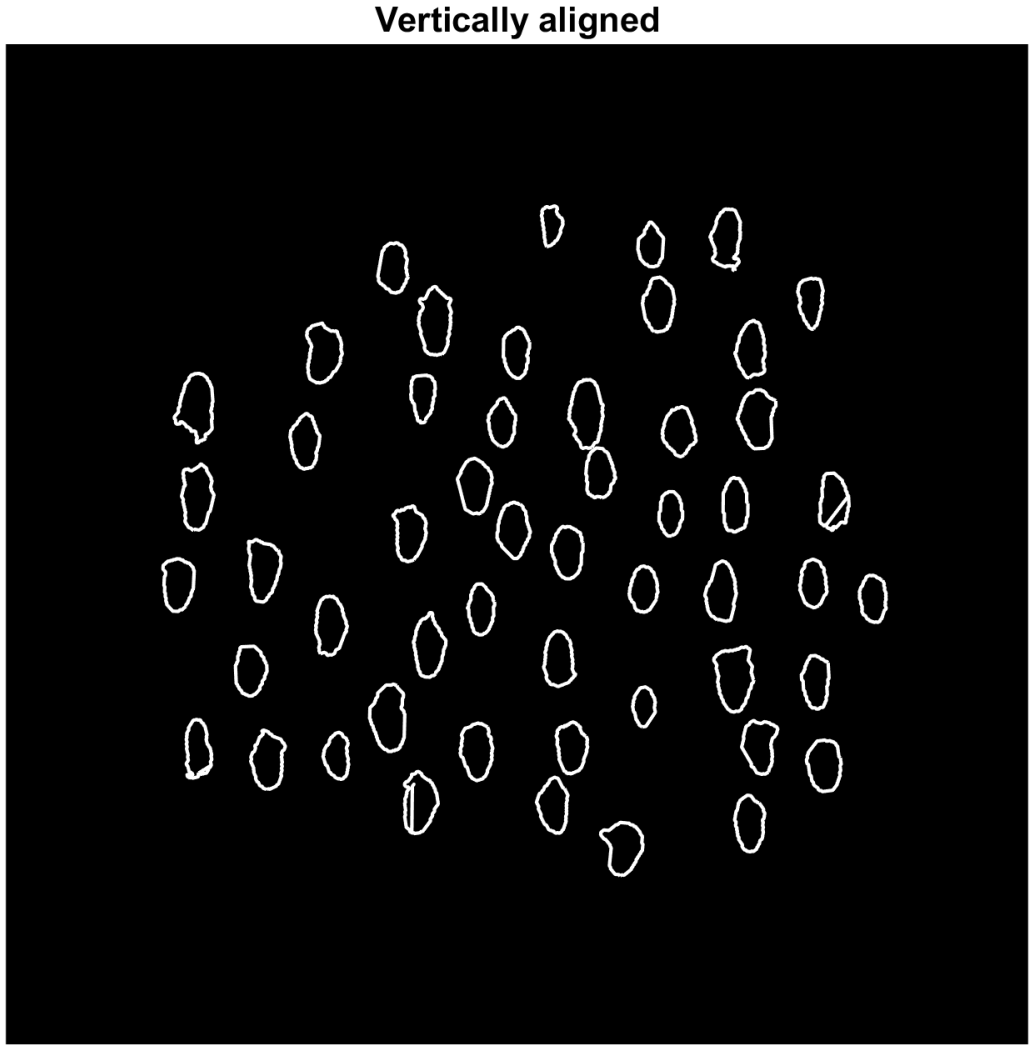


Boundaries

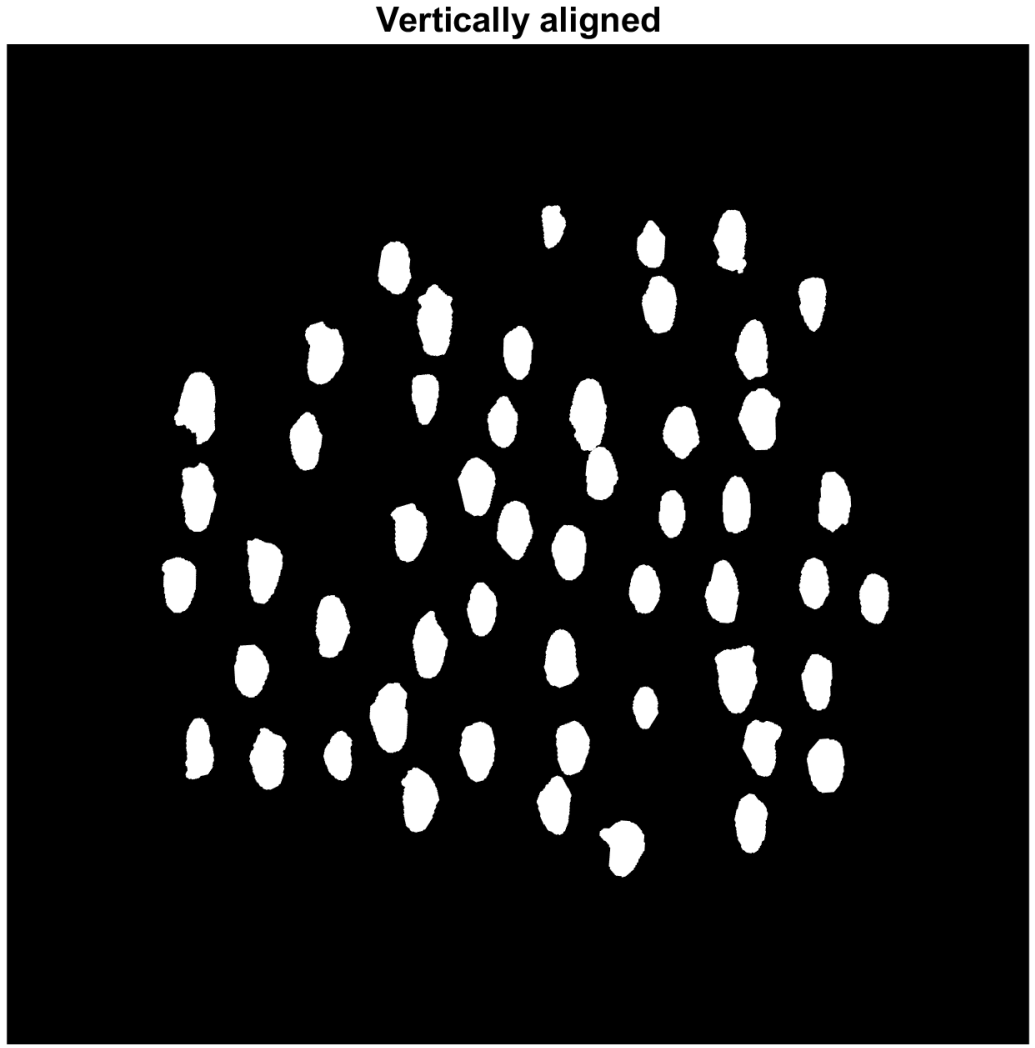


All pixels in embryos

II.4 (em4.jpg)



Boundaries



All pixels in embryos

Thanks