Purdue ECE 47900 Image and Printing

Printer Forensics Final Report

Printer Scanned image features investigation using basic machine learning methods

Zuoqian Xu (left), Alexander Gokan(right)

Professor: Jan P. Allebach

Instructor: Zhi Li

04/30/2018

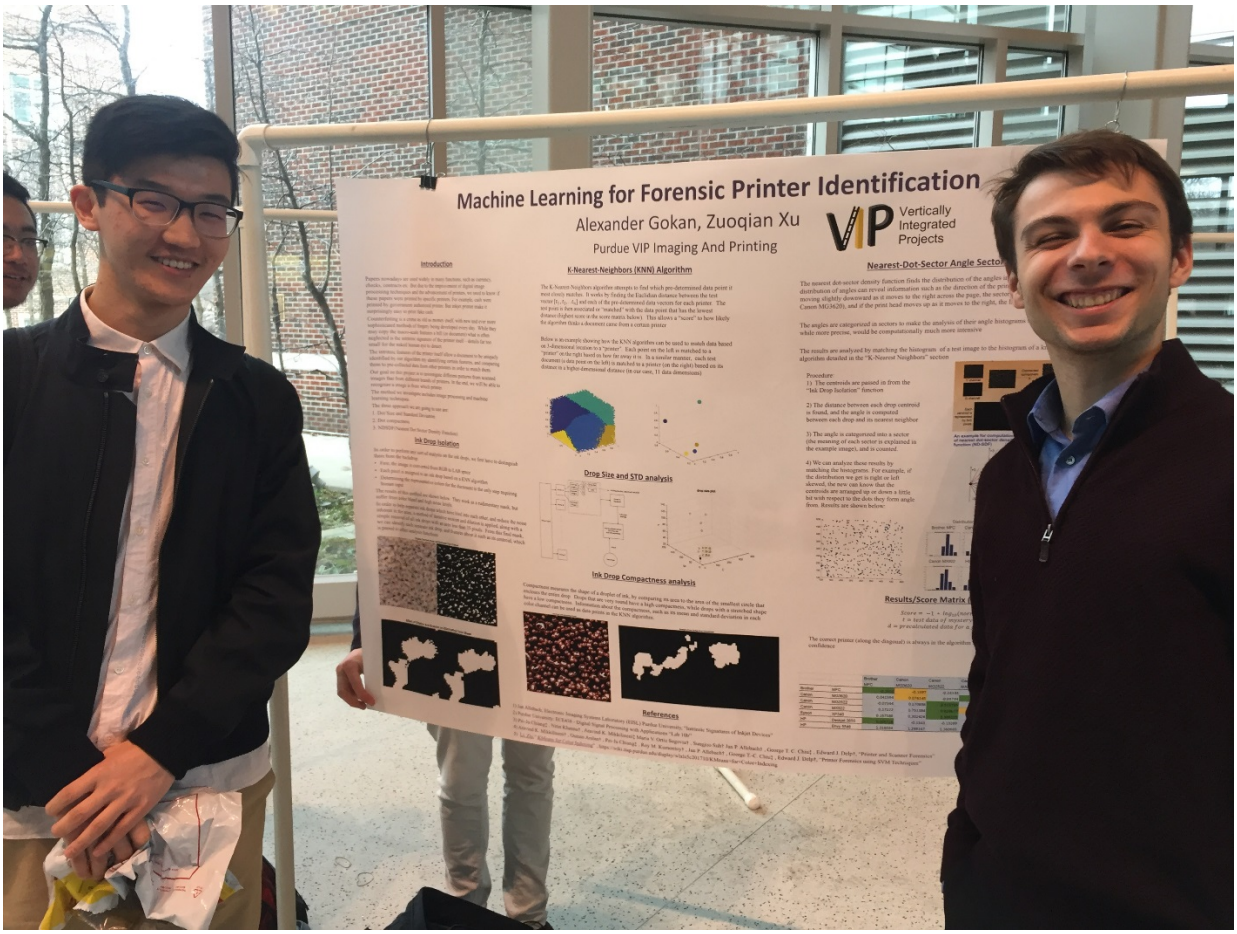


Table of Contents

Contents

Introduction

Papers nowadays are used widely in many functions, such as currency, checks, contracts etc. But due to the improvement of digital image processing techniques and the advancement of printers, we need to know if these papers were printed by specific printers. For example, cash were printed by government authorized printer. But inkjet printer make it surprisingly easy to print fake cash.

Counterfeiting is a crime as old as money itself, with new and ever more sophisticated methods of forgery being developed every day. While they may copy the macro-scale features a bill (or document) what is often neglected is the intrinsic signature of the printer itself – details far too small for the naked human eye to detect.

The intrinsic features of the printer itself allow a document to be uniquely identified by our algorithm my identifying certain features, and comparing them to pre-collected data from other printers in order to match them

Our goal on this project is to investigate different patterns from scanned images that from different brands of printers. In the end, we will be able to recognize the image is from which printer.

The method we investigate includes image processing and machine learning techniques.

The two approaches we are going to use are:

1. Dot Size and Standard Deviation matching

2. NDSDF (Nearest Dot Sector Density Function)

Forensics pre-approaches

Ink Drop Isolation:

In order to perform any sort of analysis on the ink drops, we first have to distinguish them from the backdrop.

- First, the image is converted from RGB to LAB space
- Each pixel is assigned to an ink drop based on a KNN algorithm
- Determining the representative colors for the document is the only step requiring human input

The results of this method are shown in Figure (1). They work as a rudimentary mask, but suffer from color bleed and high noise levels.

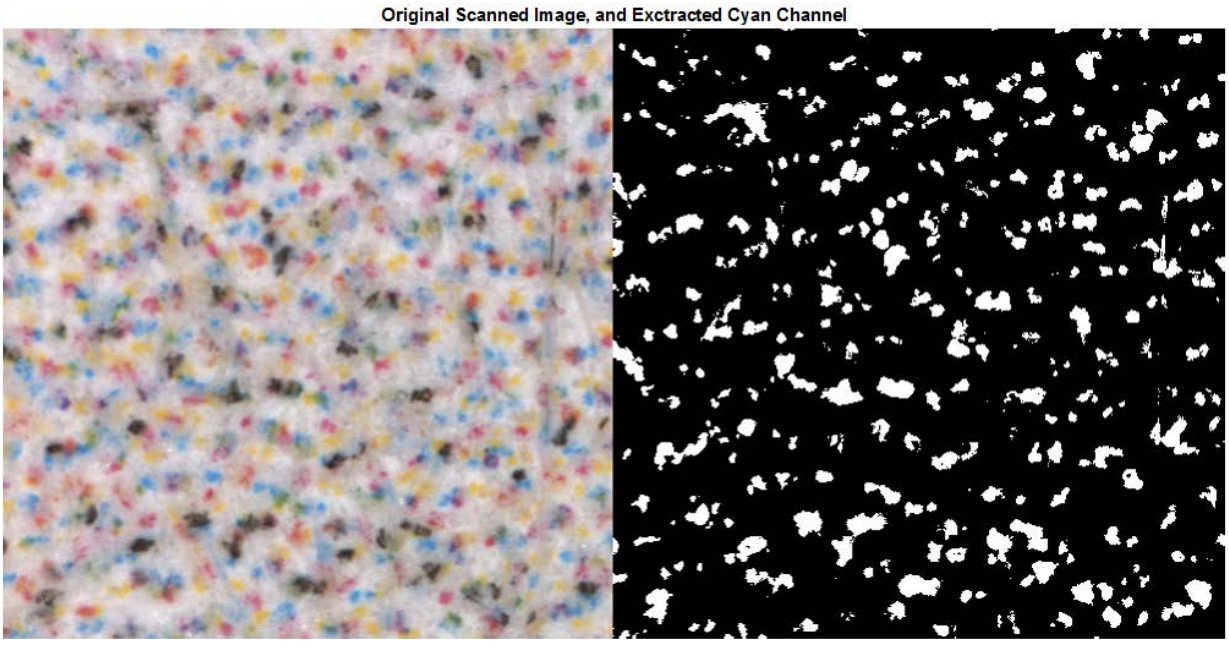


Figure (1) Ink Drop Isolation example

In order to help separate ink drops which have bled into each other, and reduce the noise inherent in the scan, a method of iterative erosion and dilation is applied Figure (2), along with a simple removal of all ink drops with an area less than 50 pixels. From this final mask, we can identify each separate ink drop, and features about it such as its centroid, which is passed to other analysis functions.

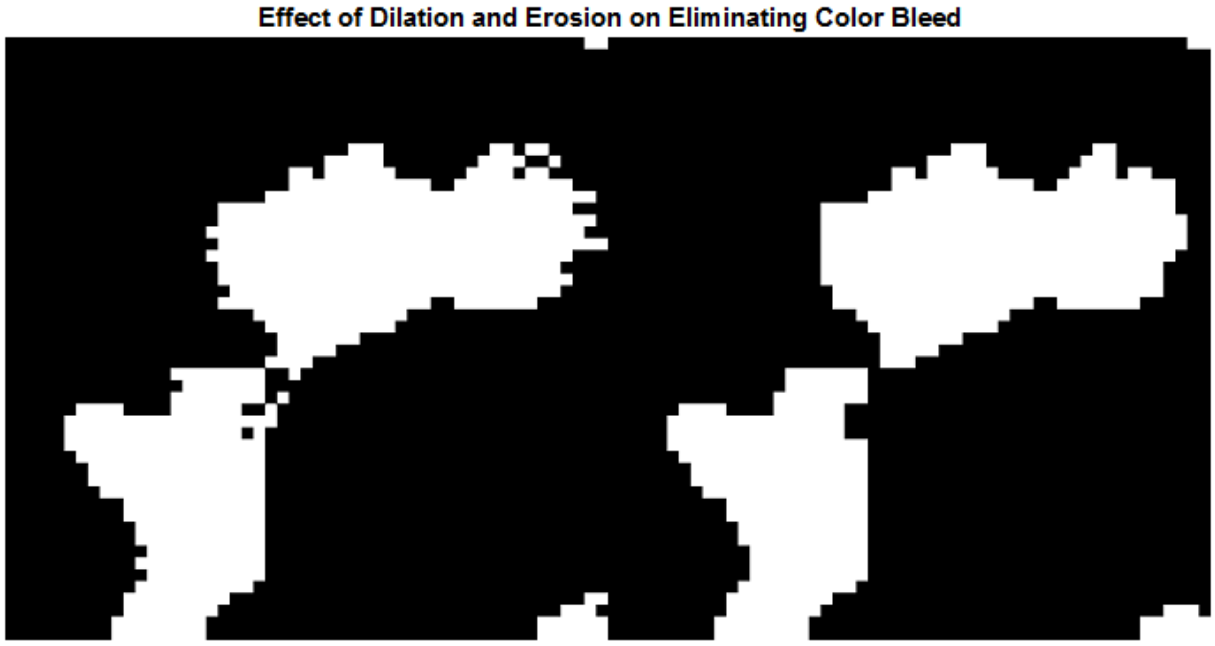


Figure (2) Effect of Dilation and Erosion Example

K-Nearest-Neighbors (KNN) Algorithm:

The K-Nearest-Neighbors algorithm attempts to find which pre-determined data point it most closely matches. It works by finding the Euclidian distance between the test vector $[\boldsymbol{t_1}, \boldsymbol{t_2}, \ldots \boldsymbol{t_n}]$ and each of the pre-determined data vectors for each printer. The test point is then associated or "matched" with the data point that has the lowest distance (highest score in the score matrix below). This allows

a “score” to how likely the algorithm thinks a document came from a certain printer

Figure (3) is an example showing how the KNN algorithm can be used to match data based on 3-dimensional location to a “printer”. Each point on the left is matched to a “printer” on the right based on how far away it is. In a similar manner, each test document (a data point on the left) is matched to a printer (on the right) based on its distance in a higher-dimensional distance (in our case, 11 data dimensions)

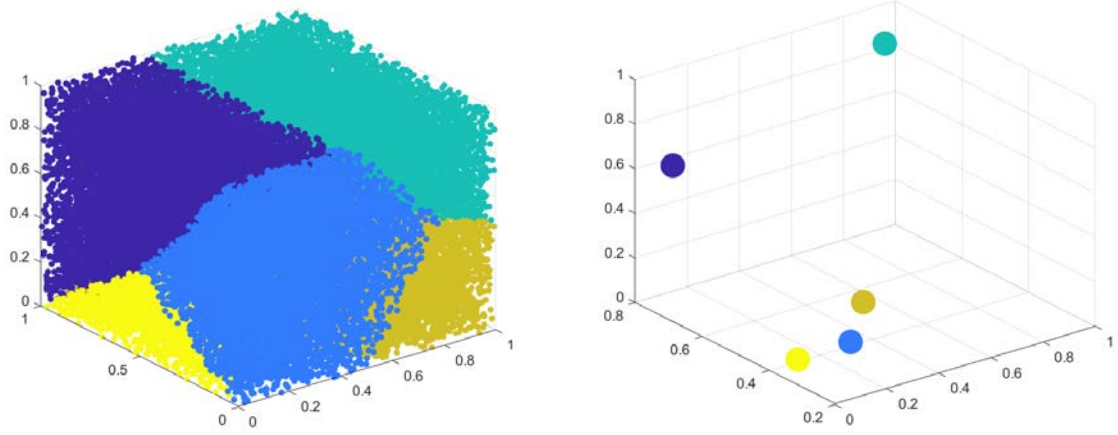


Figure (3) KNN algorithm example

Ink Drop Compactness analysis

Compactness measures the shape of a droplet of ink, by comparing its area to the area of the smallest circle that encloses the entire drop. Drops that are very round have a high compactness, while drops with a stretched shape have a low compactness. Information about the compactness, such as its mean and standard deviation in each color channel can be used as data points in the KNN algorithm. Example Compactness extraction shown in Figure (4).

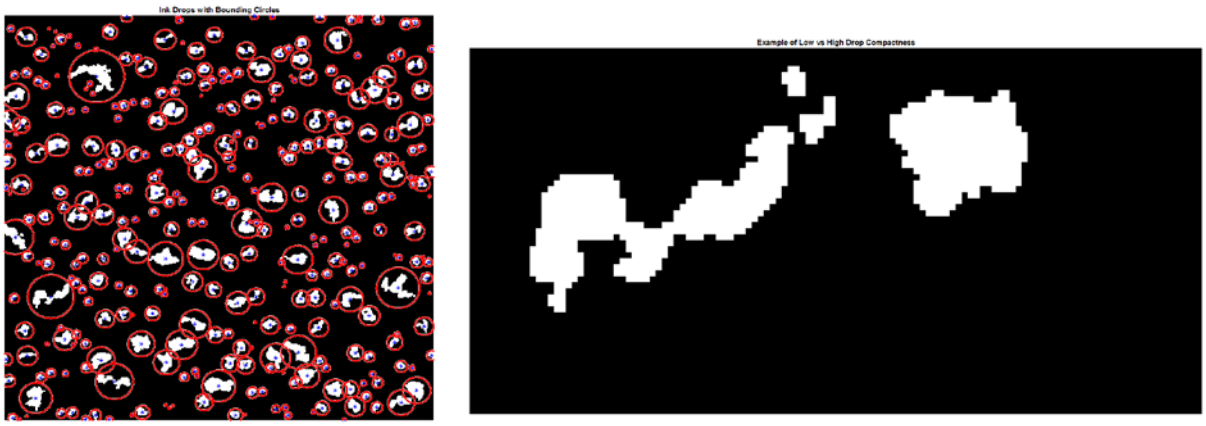


Figure (4) Compactness analysis example

Main feature Approaches

Drop Size and STD matching analysis:

Process of **Average drop size and standard deviation matching:**

1) Calculate the average drop size and drop standard deviation for C, M, Y, K colors of one scanned image.
2) Repeat step1, we implement the algorithm to the whole image base for single Printer type.
3) Create the database by export the data (drop size and STD) to excel file using MATLAB (Figure (5) (a) shows an example of extracted data).
4) Calculate the average of each color's Drop Size and STD and put them into 3D coordinate. (Figure (5) (b) shows an example of the coordinate)
5) Find the Drop Size and STD of the test image and match them with nearest Drop Size and STD in our database.

A Block Diagram explain this method shown in Figure (6)

24	15.08772	1.090032	20.5	1.375536	21.2093	1.242498
25	40.52941	0.687141	37.17857	1.001813	29.82051	0.89221
26	25.15789	0.919629	23.17857	0.791357	18.31579	0.926215
27	21.66667	1.056391	30.66667	0.897934	15.83333	1.133736
28	22.66667	0.759326	19.92857	1.418936	37.16667	1.01327
29	16.71429	1.767718	28.63636	0.783853	28.55556	0.919511
30	28.18182	0.902451	20.5	1.146521	32.16667	1.018887
31	16.41176	1.504467	24.83333	1.224878	28.09524	1.128305
32	33.72222	1.055745	18.36364	1.524806	32.05	1.58481
33	28.625	0.971406	38.23077	0.75351	23.5	0.895297
34	18.53333	1.108392	18.14286	0.888997	27.73913	0.988005
35	20.20588	1.12069	36	1.000086	26.33333	0.858973
36	13.88889	2.436975	29.16667	0.76463	20.875	1.188796
37	24.91667	0.989821	47.5	0.997393	19.57143	0.870049
38	16	2.028535	33.71429	0.883997	41	0.529442
39	28	0.901388	21.8	1.295171	25.8	0.890377
40	24.88889	0.954705	20.15385	1.110026	21.6	1.212397
41	13.72727	1.473646	9.916667	1.310895	99.66667	0.360413
42	26.88889	0.80978	23	1.129598	9	1.215893
43	17.77778	1.9675	35.55556	0.89649	31.16667	1.145309
44	22.125	1.324233	30.875	1.096501	26	1.176445
45						
46	Averages:					
47	Cds	Cstd	Mds	Mstd	Yds	Ystd
48	27.41969	1.286156	29.21972	1.208112	28.59704	1.124657
49						
50						
51						

Figure (5) (a)

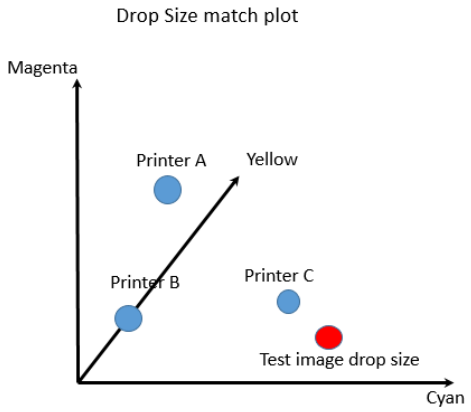


Figure (5) (b)

In Figure (5) (b), We find the average drop size of multiple different scanned images from specific printers and plot that point onto the coordinate system by Magenta, Yellow, and Cyan drop sizes

As the image shows, for example we have 3 printers A, B and C. All plotted on the coordinate by getting the average training drop size result.

After that, we calculate the drop size of a specific scanned image which we don't know its printer. Plot that point, and find out which printer represented dot it closest to. After this, we match the printer.

Note:

MATLAB CODE in Appendix (1)

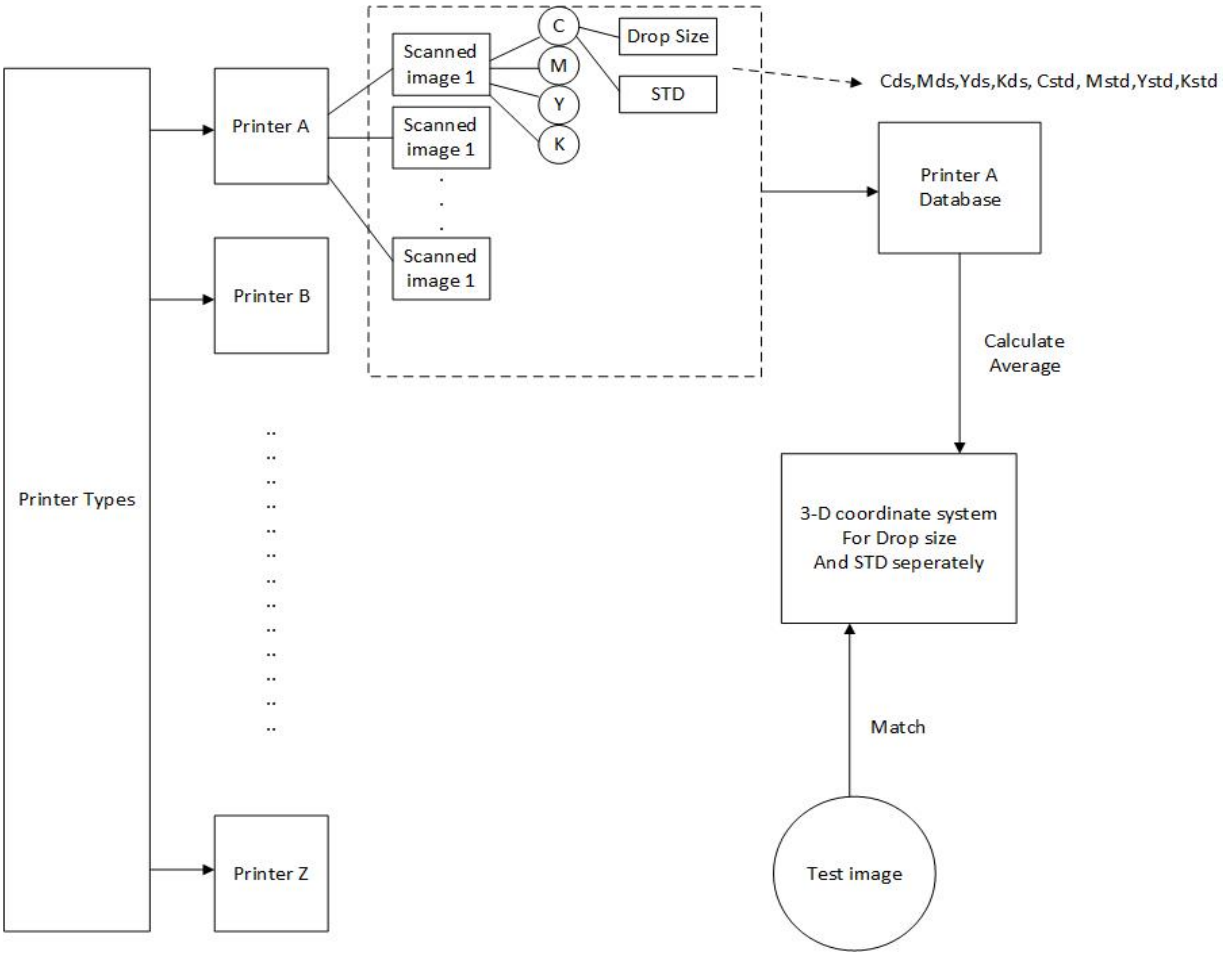


Figure (6) block diagram of DS&STD matching

Results:

The Image we tested on is all from new Scanned image that our instructor send us later. The images can be downloaded from here.

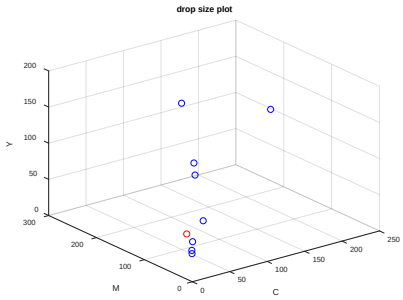
Printer Scanned image:

https://drive.google.com/open?id=1pLsyHmKnBEicdxce_e92ZtBkN0y1Olpx

Elements in the presented images:

1. Original image for testing:

2. 3D matching Coordinated system:



3. Output Drop Size extraction images:

4. Results he algorithm shows:

   D1112 with ds

   MFC with STD

Figure (7) (a-h) shows Example results for multiple test scanned images from different printers.

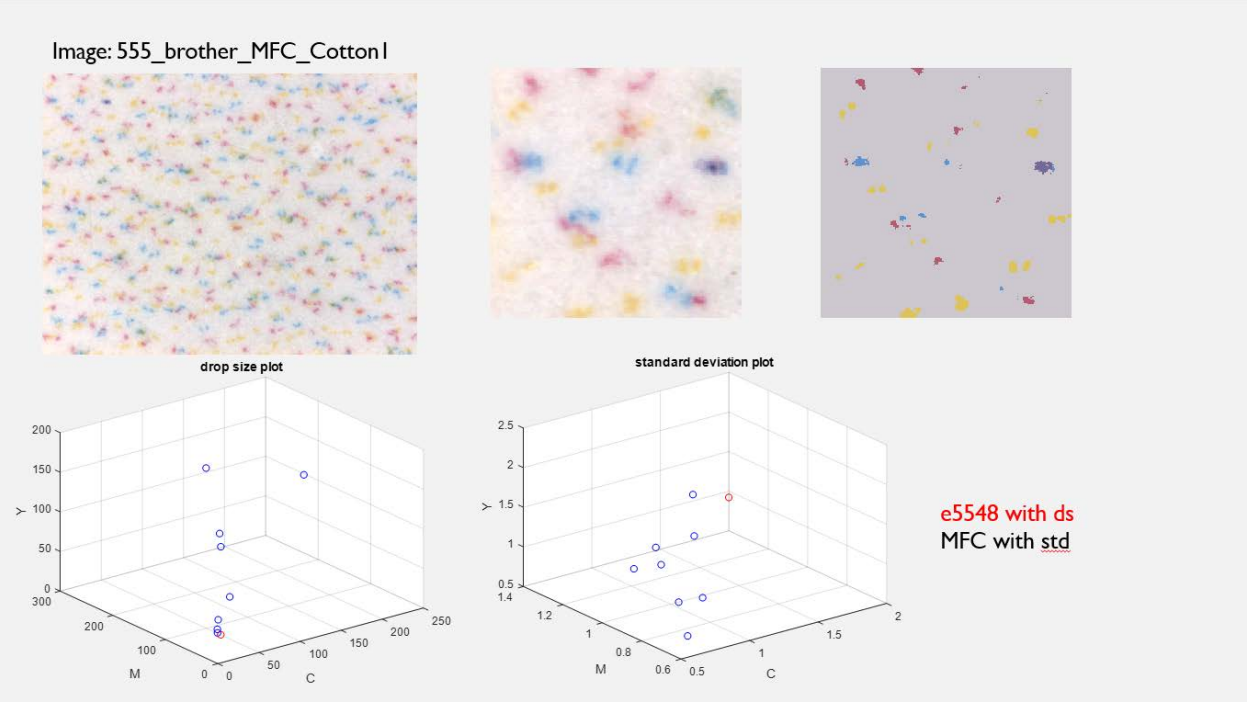


Figure (7) (a) Brother MFC scanned image matching results

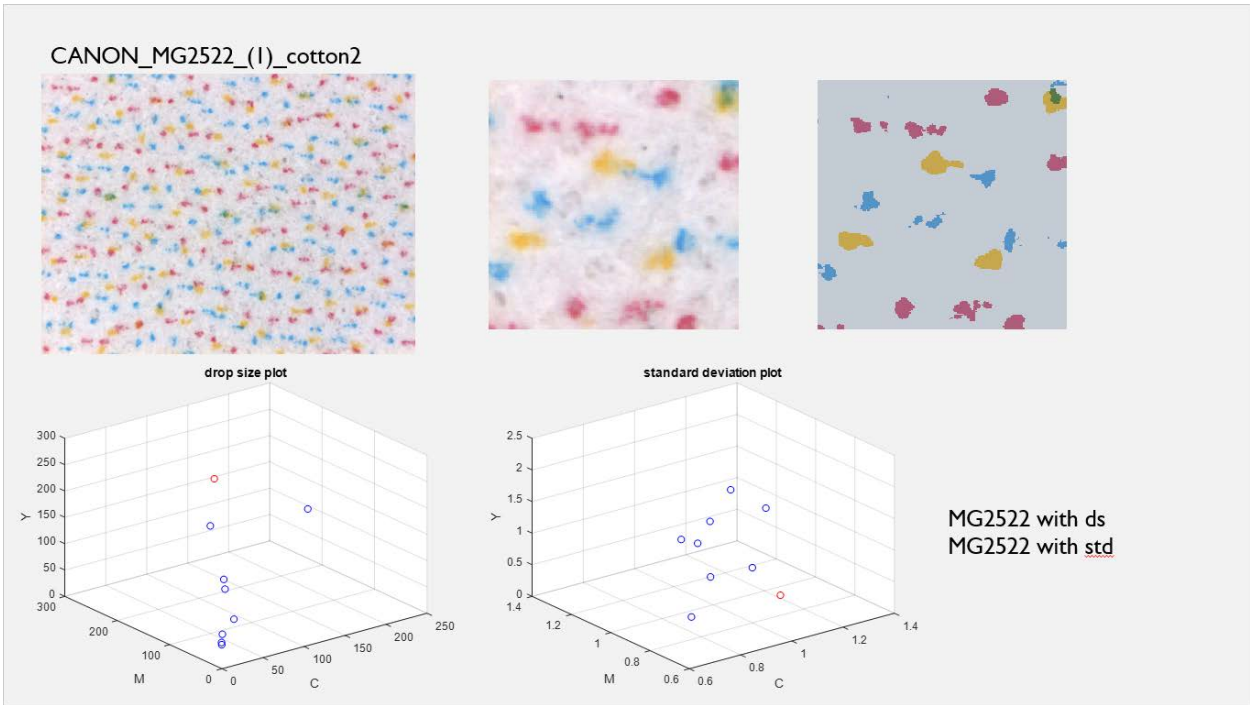


Figure (7) (b) CANON MG2522 scanned image matching results

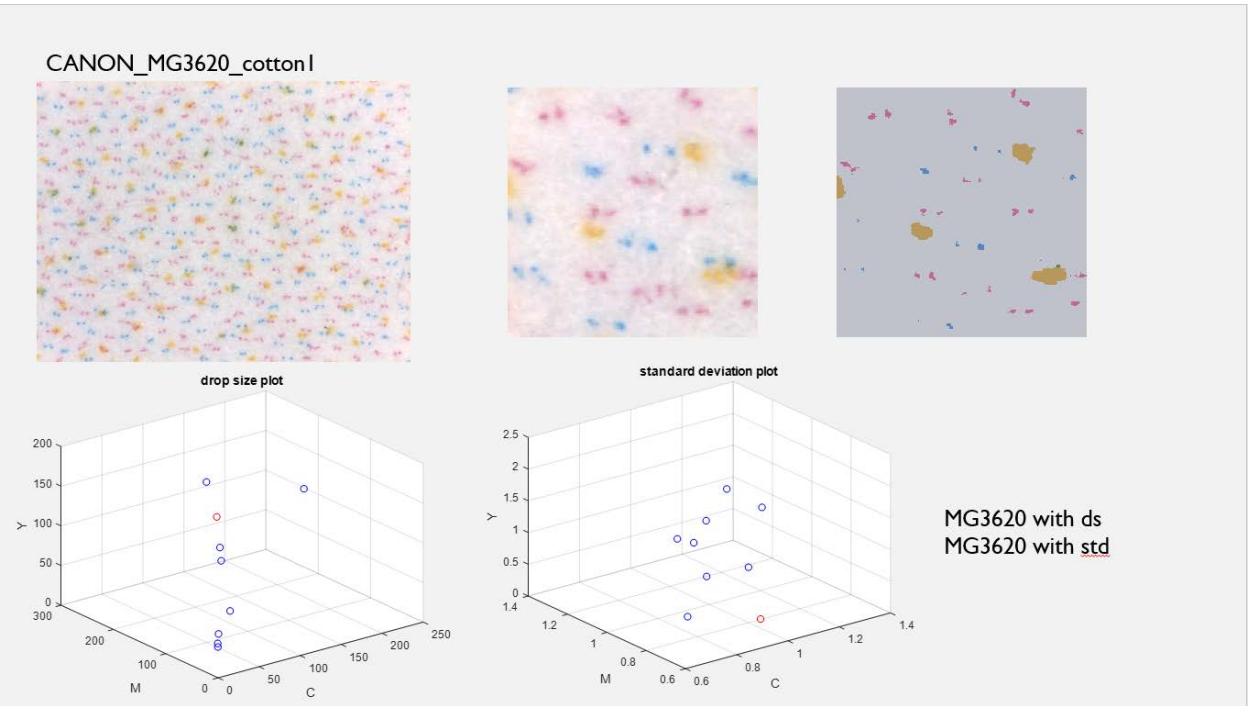


Figure (7) (c) CANON MG3620 scanned image matching results

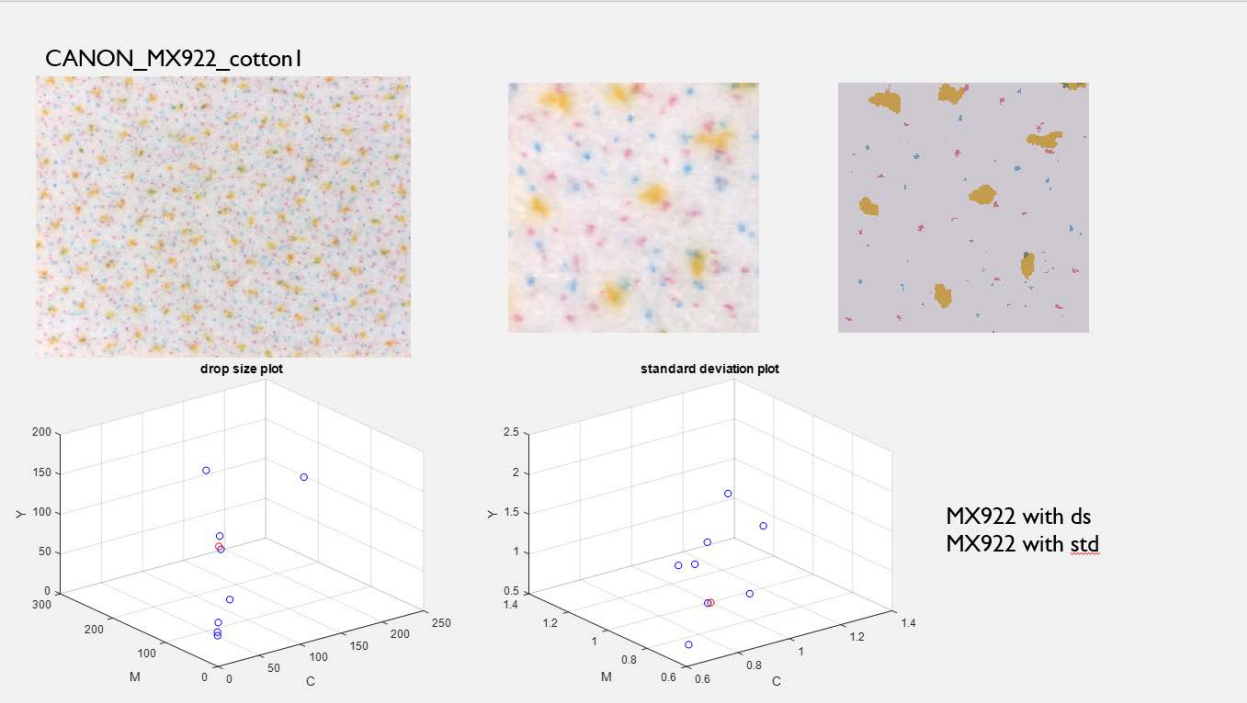


Figure (7) (d) CANON MX922 scanned image matching results

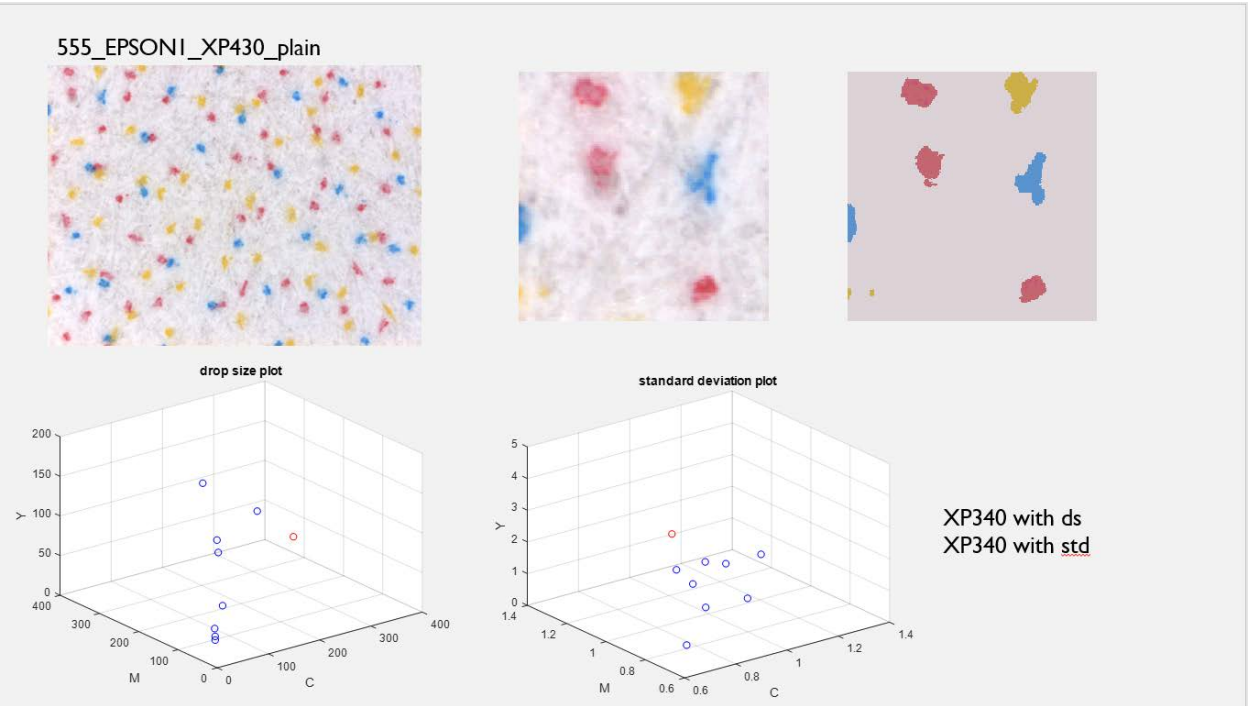


Figure (7) (e) EPSONI XP340 scanned image matching results

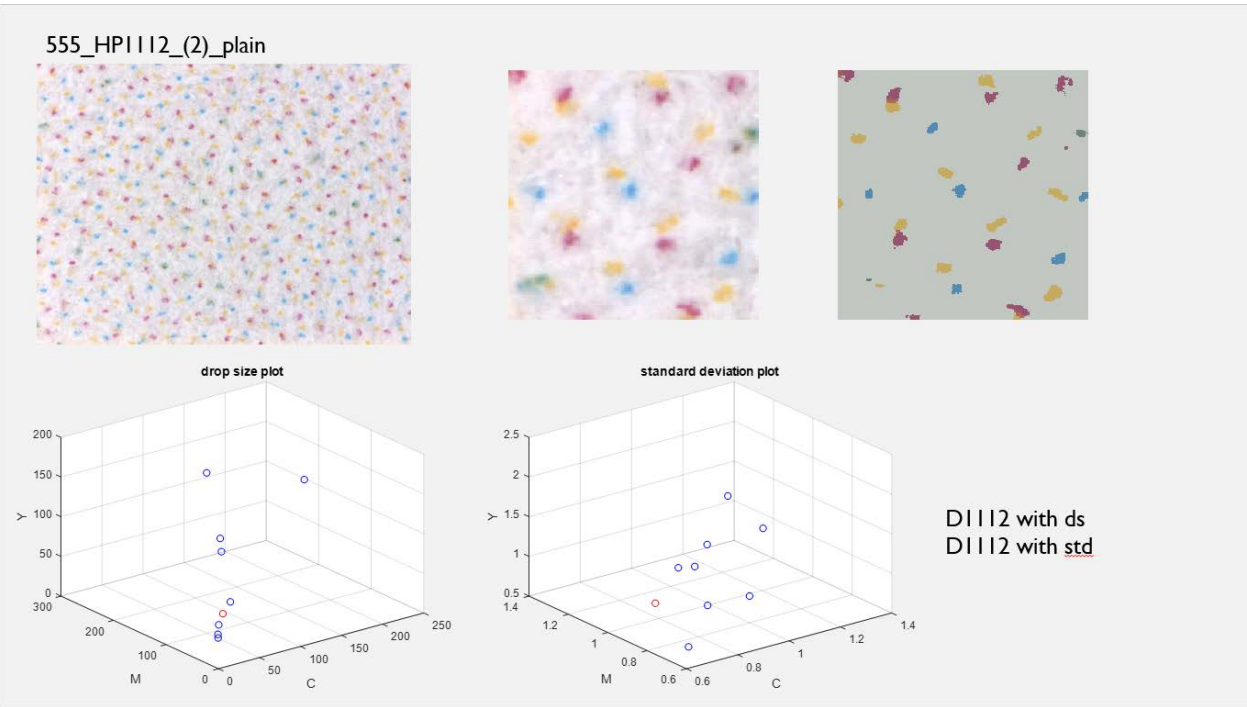


Figure (7) (f) HP1112 scanned image matching results

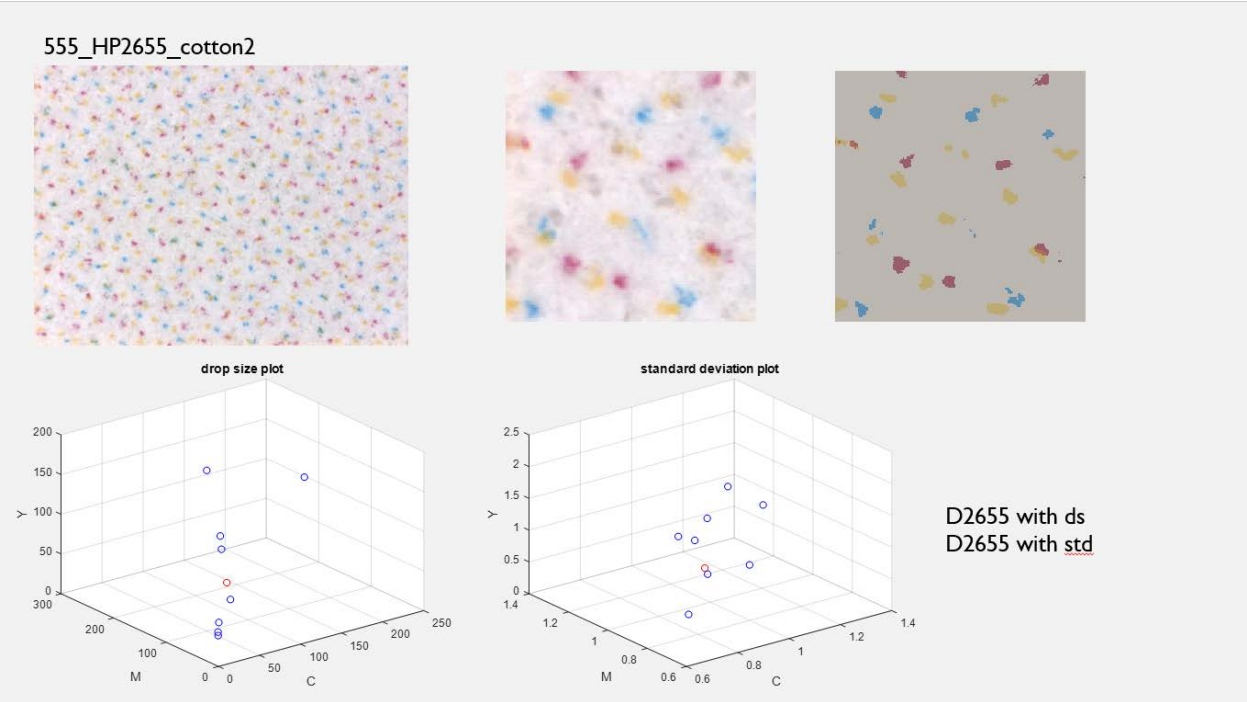


Figure (7) (g) HP2655 scanned image matching results

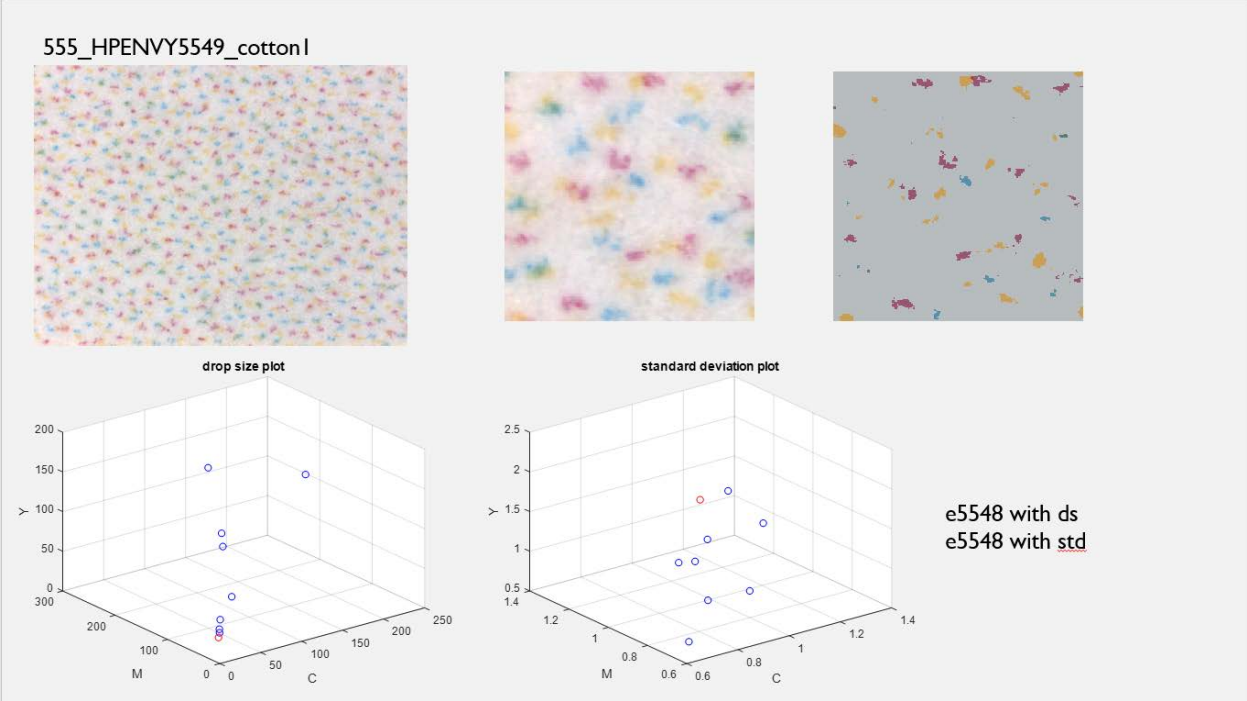


Figure (7) (h) HP envy5549 scanned image matching results

Matching Percentage and error analysis:

* STD match 100%

* DS match 87.5%

For the Drop Size result we have one wrong matching which is the HP envy5549 and Brother MFC. Figure (8) shows the two DS extracted image.

Figure (8)

If we take a close look to the drop sizes of three colors (in this case only Yellow, Cyan, and blue). We can see the drop sizes for the two printer's scanned image is visually close. The reason that cause this problem is normal, since the scanned images base we use only has at most 8 images for each printer. If we want a more precise results, we need more images to train the coordinate, adjust the printer represented dots again and again until the when we matching the closed dots, we have a precision good enough to distinguish them. But in conclusion, the algorithm is correct and can be used to perform printer forensics.

Nearest Dot Sector Density Function (NDSDF) Analysis:

Introduction:

The nearest dot-sector density function finds the distribution of the angles in that the ink drops are lined up in. The distribution of angles can reveal information such as the direction of the print head. For example, if the print head is moving slightly downward as it moves to the right across the page, the sector histogram will be left skewed (for example Canon MG3620), and if the print head moves up as it moves to the right, the histogram will be right skewed.

The angles are categorized in sectors to make the analysis of their angle histograms simpler. Matching the raw data, while more precise, would be computationally much more intensive.

The results are analyzed by matching the histogram of a test image to the histogram of a known printer, using the KNN algorithm detailed in the "K-Nearest Neighbors" section.

Procedures:

1) The centroids are passed in from the "Ink Drop Isolation" function. Figure (9) shows the example procedure of getting centroids.
2) The distance between each drop centroid is found, and the angle is computed between each centroids and its nearest neighbors.
3) The angle is categorized into a sector (the meaning of each sector is explained in the example image), and is counted. Sector example shown in Figure (10).

4) We can analyze these results by matching the histograms. For example, if the distribution we get is right or left skewed, the new can know that the centroids are arranged up or down a little bit with respect to the dots they form angle from.

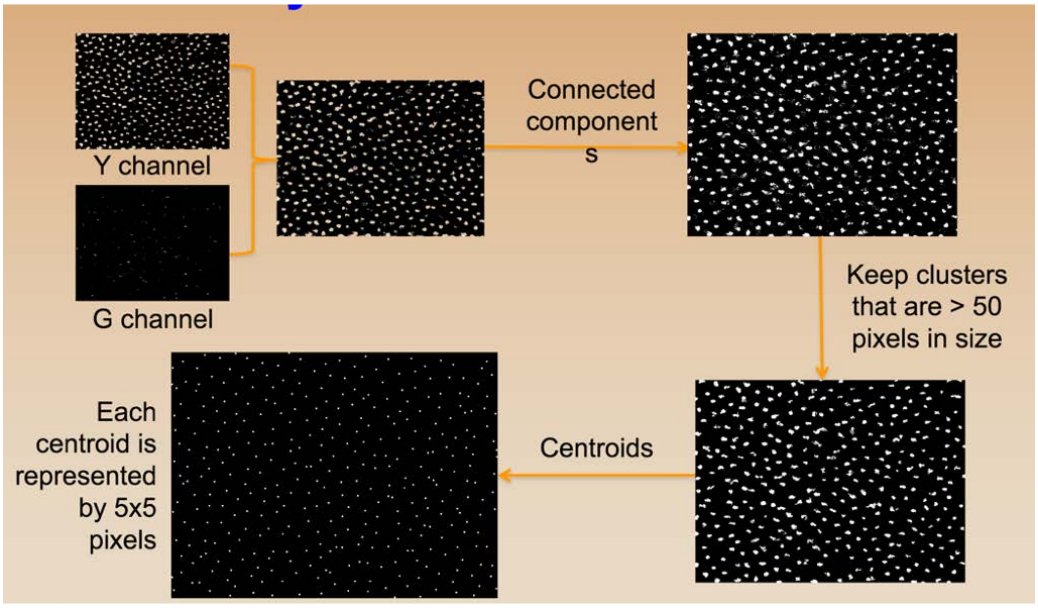
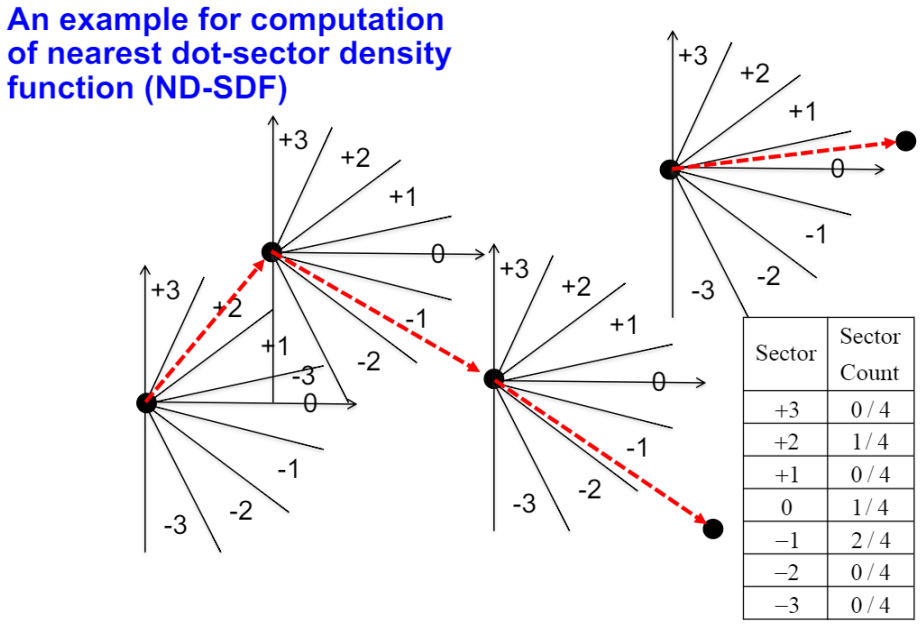


Figure (9) Procedure of getting centroids



Sector	Sector Count
+3	0 / 4
+2	1 / 4
+1	0 / 4
0	1 / 4
−1	2 / 4
−2	0 / 4
−3	0 / 4

Figure (10) sector explanation

In our results, we set the cluster size threshold filtering value to 50. And getting the results for centroids distributions. Note that the color channel that we are extracting from is Cyan channel only. Then, after we find the nearest dots for every dots in all the centroids, we connect them with lines. Finally, I also calculate the average change in angles for every pair of nearest dots. The results are shown in Figure (11) (a-h).

Note: MATLAB CODE in Appendix (2)

Printer: Brother MFC

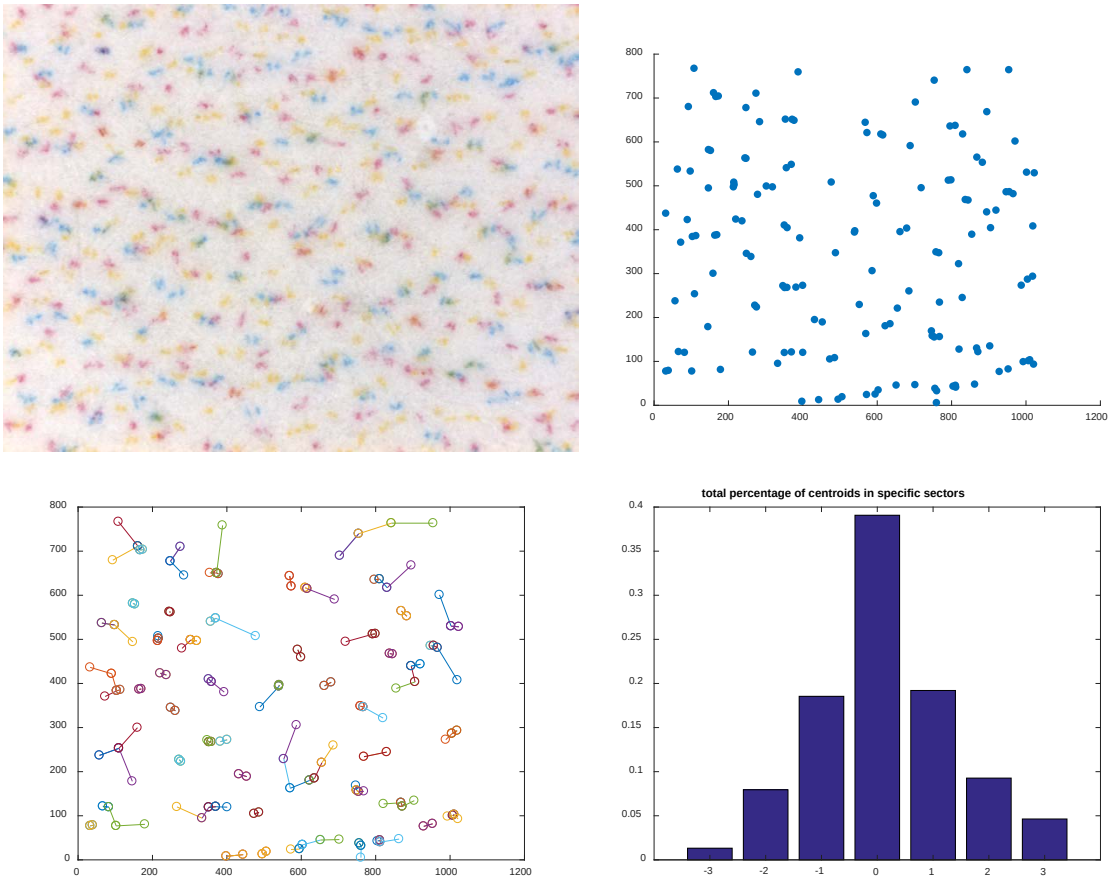


Figure (11) (a) 555_brother_MFC_Cotton1 test image results

Average change of angles: 42.1796

Brother MFC printer have pretty intensive cyan dots distribution. From the histogram and the trend lines shown in third plot. The 0, -1 and +1 sectors are most. Which means the angles between all the nearest pairs of centroids are close to horizontal with acute angle. And based on the average change of angles, 42 degree is not a big difference. So Brother MFC printer‘s ink drop has most of the nearest pairs lined up horizontally.

Printer: EPSON XP430

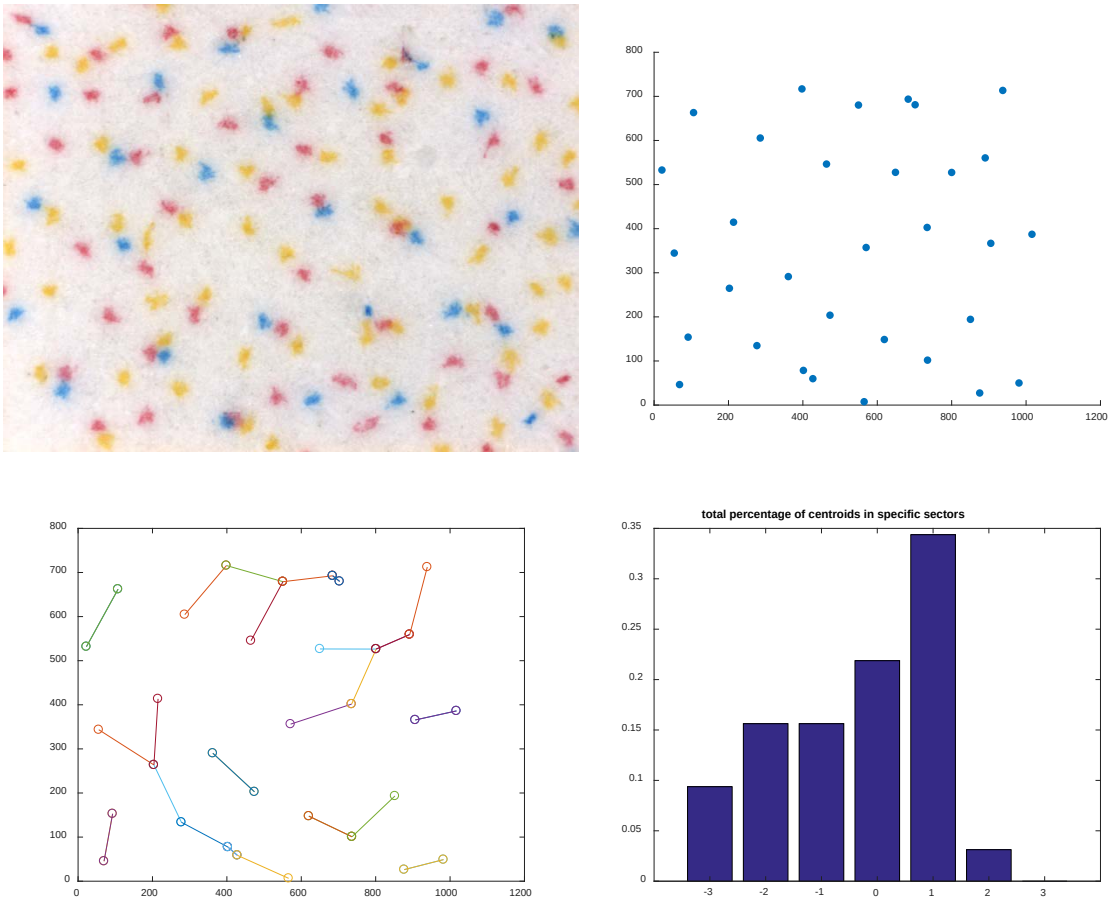


Figure (11) (b) 555_EPSON1_XP430_cotton1 test image results

Average change of angles: 47.2220

The ink drops characteristics for EPSON XP430 is that most of the pairs lined up in an acute angle contain both little negative and positive fluctuation. But due to the little amount of angles lie in sector +2 and +3, the overall trend will be lining downward with respect to the 0 sector.

Printer: HP1112

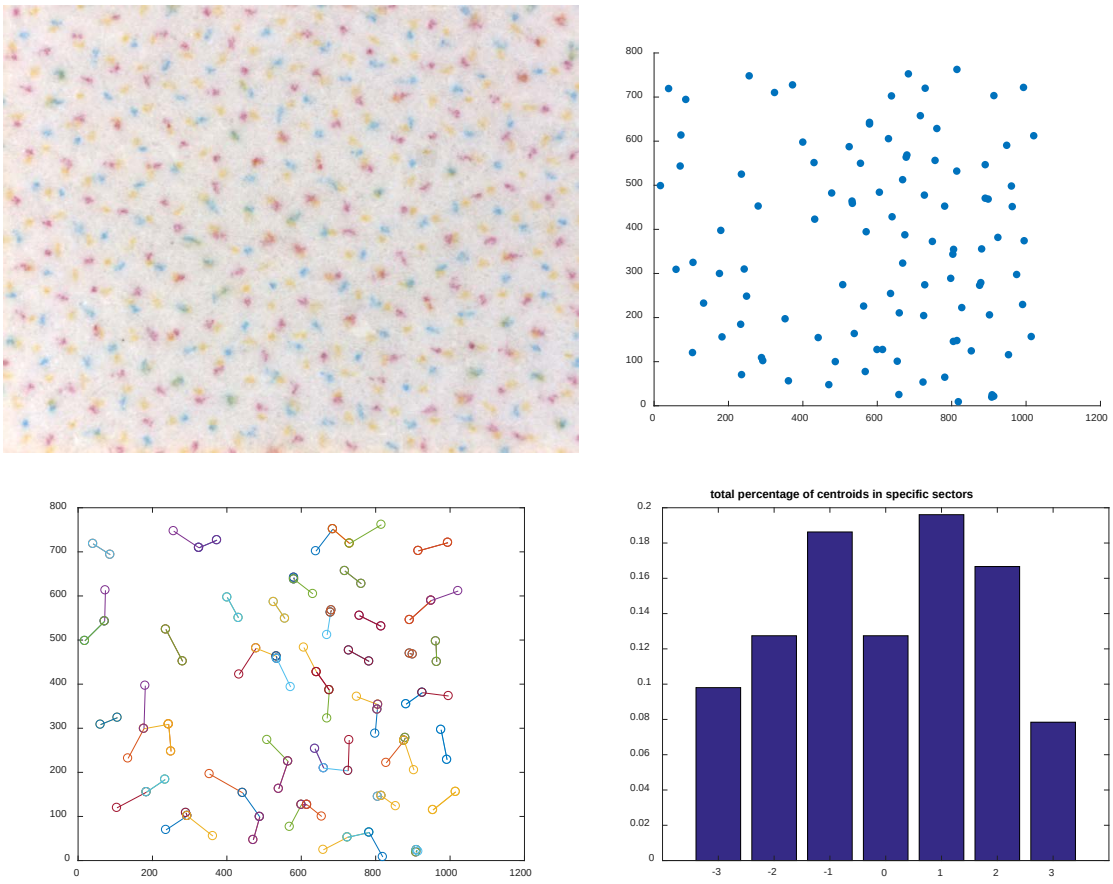


Figure (11) (c) 555_HP1112_(2)_cotton1 test image results

Average change of angles: 60.7320

The ink drops characteristic for HP 1112 is that most pairs lined up in obtuse angle. And the distribution of them is very random.

Printer: HP2655

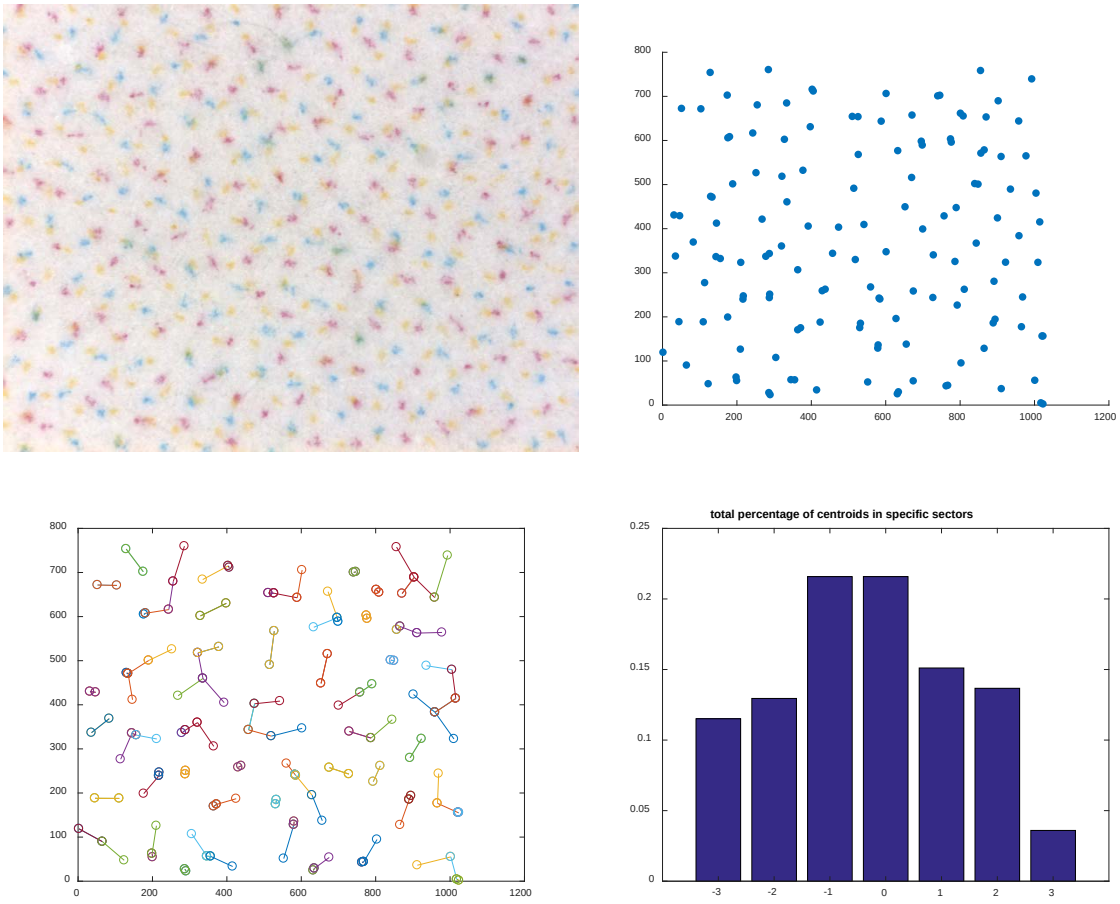


Figure (11) (d) 555_HP2655_cotton1 test image results

Average change of angles: 54.3961

For this printer, the results are close to HP1112 printer since they are from the same HP brand. But the slight difference is that more ink drop pair lined up horizontally with most of them in sector 0 and -1.

Printer: HP envy 5549

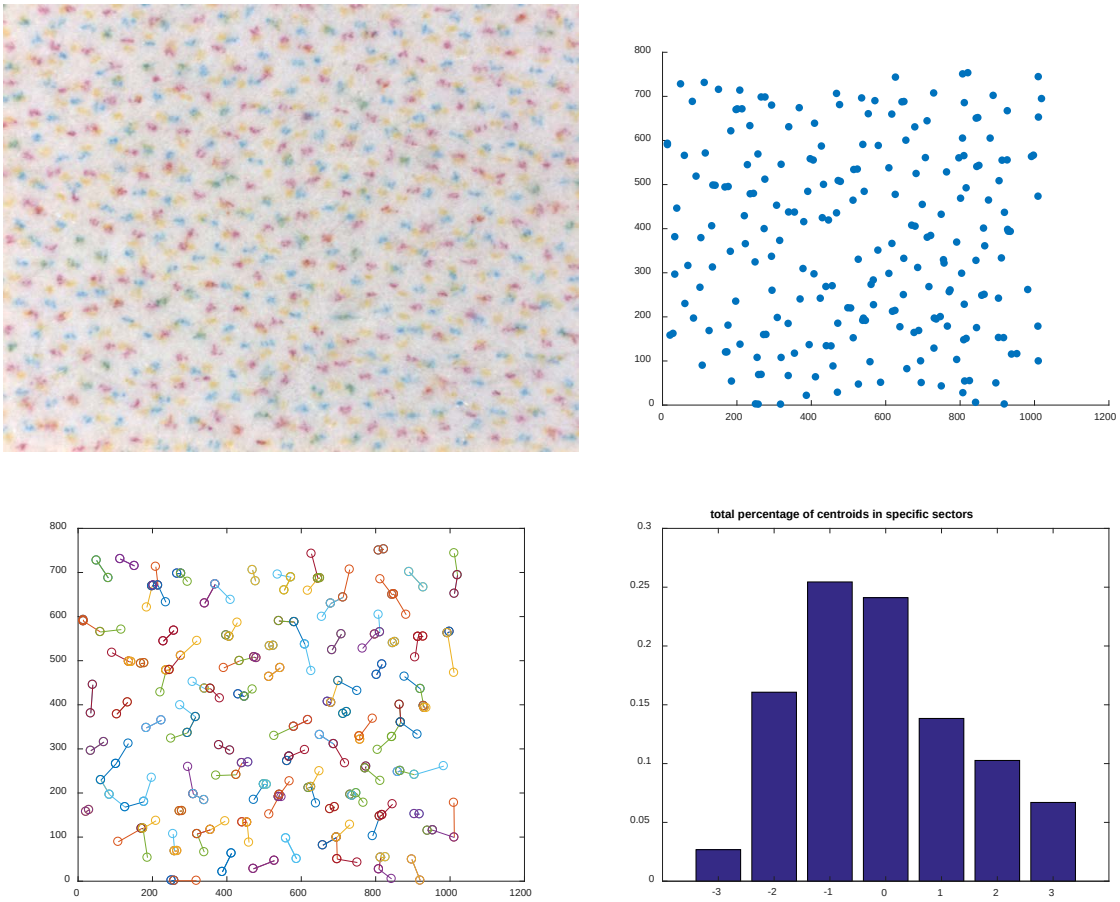


Figure (11) (e) 555_HPENVY5549_cotton1 test image results

Average change of angles: 49.8220

Again, same brand HP printer will give a pretty close result. But if we look at the detailed difference, this envy5549 have more ink drop pairs lined up horizontally. The difference of angles is the smallest in this printer type among HP brand.

Printer: CANNON MG2522

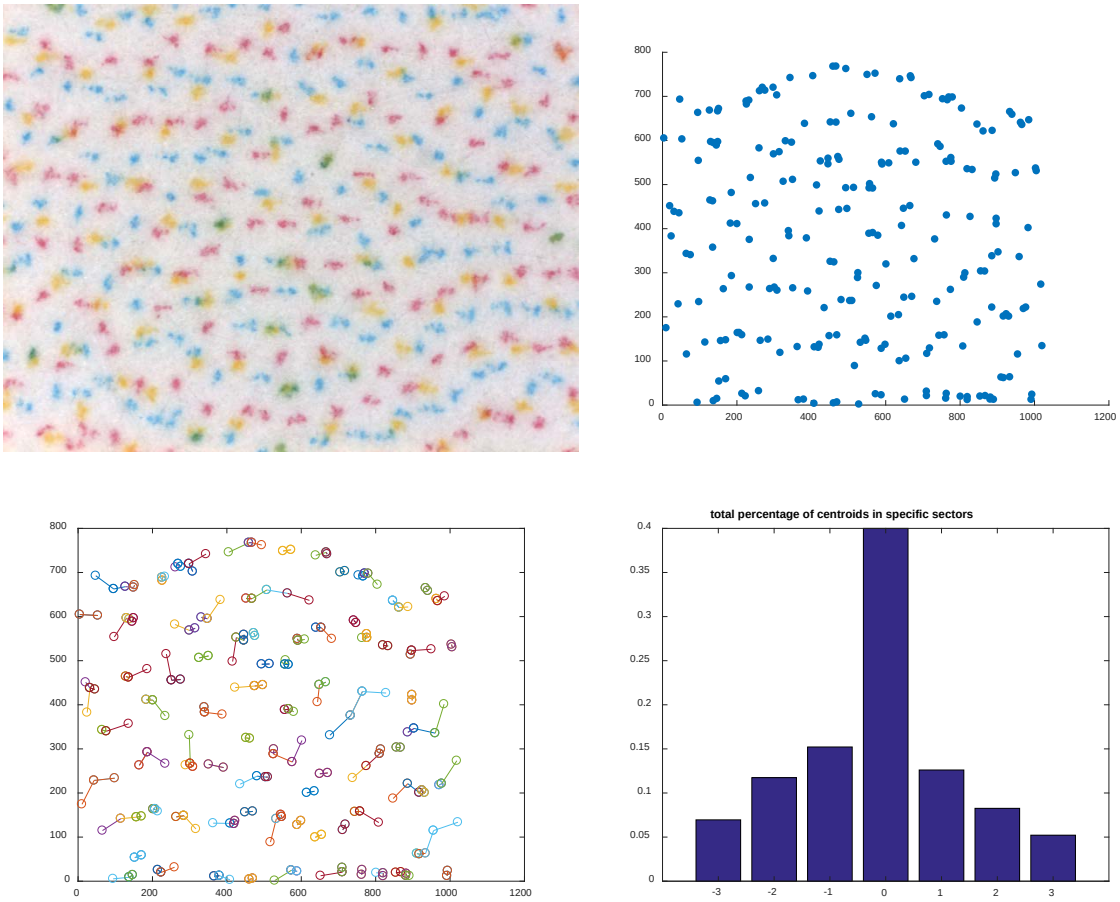


Figure (11) (f) CANON_MG2522_(1)_cotton1 test image results

Average change of angles: 48.5013

For this CANNON printer, after we extract all the centroid drops, we find that many centroids still positioned really close to each other. We can directly see from the second plot that most centroids lined up horizontally and even form an actual line. From the histogram we can also see that most pairs form angle that in sector 0.

Printer: CANNON MG3620

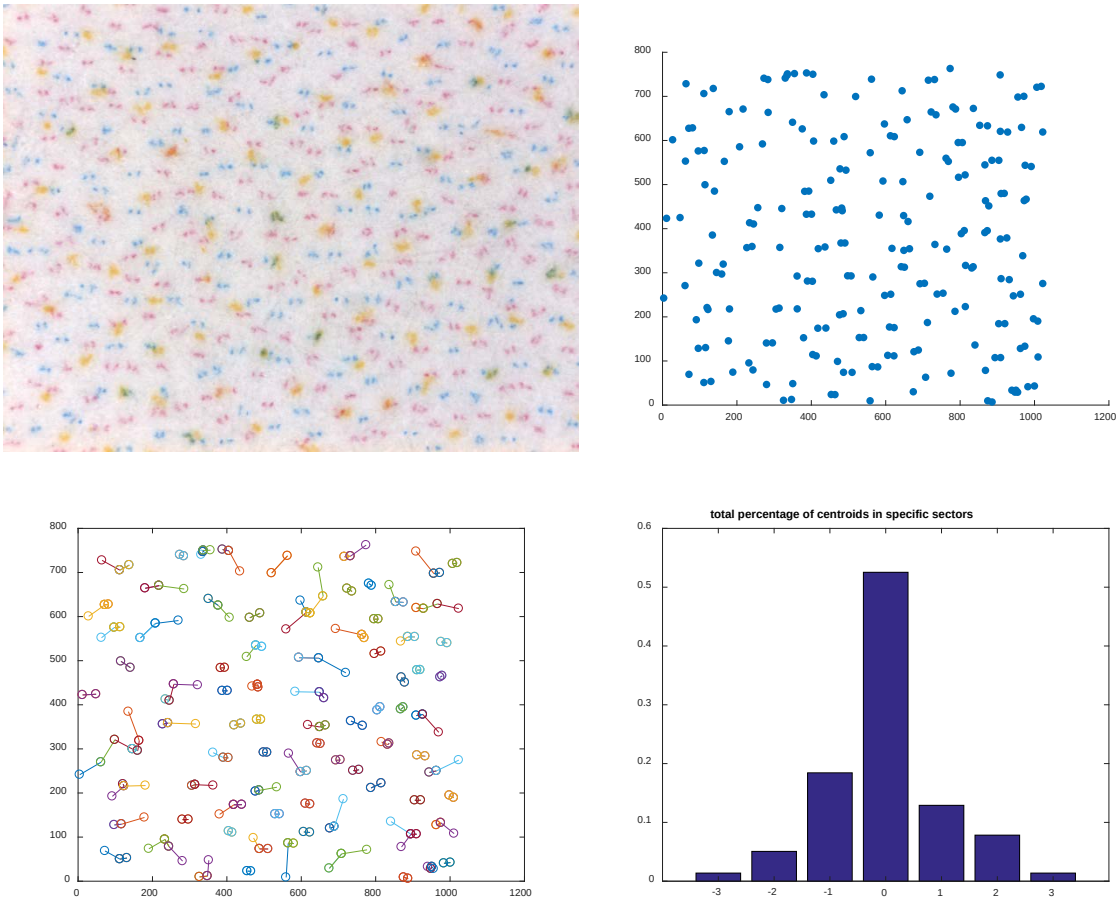


Figure (11) (g) CANON_MG3620_cotton1 test image results

Average change of angles: 32.6904

Again, same CANNON printer brand. But this printer is the most special one we can see so far. The ink drops distribute nearly in horizontal 0 degree line. Form the histogram we can see most pairs are in sector 0. Compare to MG 2522, the nearest centroid pairs in other sectors than sector 0 decrease. And From the average angle change, 32 degree is a very acute angle.

Printer: CANNON MX922

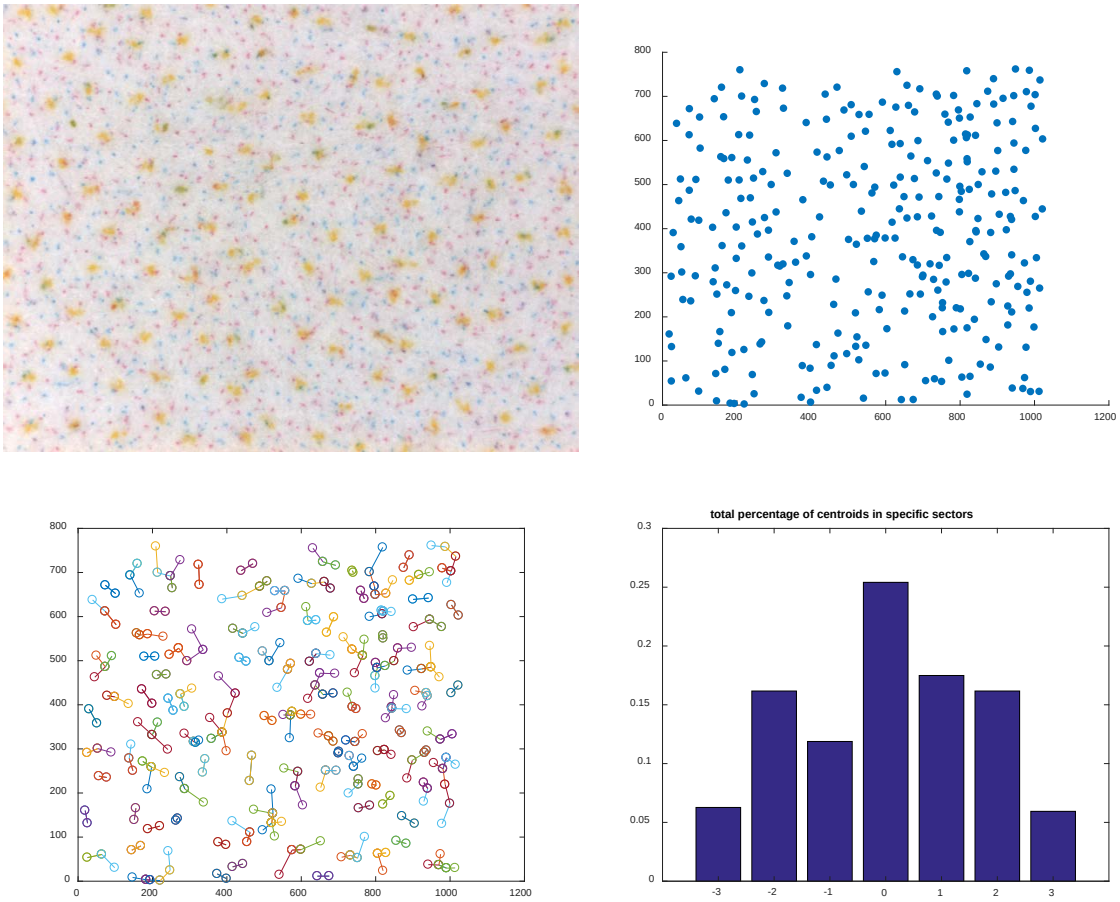


Figure (11) (h) CANON_MG3620_cotton1 test image results

Average change of angles: 54.8238

Another CANNON brand printer. The interesting part is this printer is no like the two CANNON printer we see above. This one has its ink drop distribution more randomly. Most pairs lined up in obtuse angle.

Conclusion:

The main features I investigate this semester are DS&STD matching and NDSDF method. The first one more depend on quantity of image for training and quality of image for database set up. As I mentioned earlier, if we have more scanned image available for us to modify the position of printer represent dots in the 3D forensics coordinate, our results should be better. In the future, this matching method can be used to determine the printer sources of an unknown image. But the problem is we can only distinguish the image if they are close enough to the images in our database. And again to set up a complete and perfect forensics database we need thousands of pictures in the future.

The NDSDF approach is more like a feature recognition of ink drop image. We can only use this method on ink drops. The NDSDF gives good features of a particular printer, we can investigate the distribution of any kinds of ink drop from any kinds of printers. But the problem of this method is if we use it on huge amount of printers, the replication of features can't be avoid. Maybe we have complete different printer type from different brand but the ink drop feature looks the same. For example HP2655 and CANNON MX922. The distributions for them are both pretty random, so as a result we can't really tell the difference between them.

These two approach are pretty basic methods for printer forensics. From my results, the data I generated will connect with my teammate Alex's algorithms. For detailed forensics using data scoring method, refer to Alex Gokan's report. But finally, this research use many machine learning approaches like KNN and

clustering. We all learned and became familiar with machine learning and hope this new skill will help us create more progress in the future.

References

1) Jan Allebach, Electronic Imaging Systems Laboratory (EISL) Purdue University, “Intrinsic Signatures of Inkjet Devices”

2) Purdue University: ECE438 - Digital Signal Processing with Applications “Lab 10b”

3) Pei-Ju Chiang, Nitin Khanna, Aravind K. Mikkilineni Maria V. Ortiz Segovia, Sungjoo Suh† Jan P. Allebach , George T. C. Chiu‡ , Edward J. Delp, “Printer and Scanner Forensics”

4) Aravind K. Mikkilineni, Osman Arslan, Pei-Ju Chiang, Roy M. Kumontoy, Jan P. Allebach , George T.-C. Chiu, Edward J. Delp, “Printer Forensics using SVM Techniques”

5) Li, Zhi,” KMeans for Color Indexing”, https://wiki.itap.purdue.edu/display/wlxls5c201710/KMeans+for+Color+Indexing

Appendix (1)

MATALB CODE data writer:

```
srcFiles = dir('U:\Desktop\ECE 479 Image and Printing\printer
forensics\database\555_diff_printer_paper\HP Deskjet2655\*.bmp');
for i = 1 : length(srcFiles)
    filename = strcat('U:\Desktop\ECE 479 Image and Printing\printer
forensics\database\555_diff_printer_paper\HP Deskjet2655\',srcFiles(i).name);
    im = imread(filename);
    colors = set_colors('HP Deskjet2655');%put folder of image here
    blur_factor = 0.05; %changed
    im = im(1:200,1:200,:); %changed
    [Cds,Cstd,Mds,Mstd,Yds,Ystd,Kds,Kstd] =
get_drop_size(im,colors,blur_factor);
    output = [Cds,Cstd,Mds,Mstd,Yds,Ystd,Kds,Kstd];
    filename = 'HP Deskjet2655';
    start_range = strcat('A',int2str(i));
    end_range = strcat('H',int2str(i));
    range1 = strcat(start_range,':');
    range = strcat(range1,end_range);
    xlswrite(filename,output,range);
end
toc;
```

MATALB CODE DS&STD matching:

```
MFC = xlsread('Brother MFC.xlsx','A5:F5');
MG3620 = xlsread('Canon MG3620.xlsx','A5:F5');
MG2522 = xlsread('CANON MG2522.xlsx','A11:F11');
MX922 = xlsread('Canon MX922.xlsx','A5:F5');
XP340 = xlsread('Epson XP340.xlsx','A11:F11');
D1112 = xlsread('HP Deskjet1112_Best.xlsx','A8:F8');
D2655 = xlsread('HP Deskjet2655.xlsx','A5:F5');
e5548 = xlsread('HP envy5548.xlsx','A5:F5');
```

```
Cds_db =
[MFC(1),MG3620(1),MG2522(1),MX922(1),XP340(1),D1112(1),D2655(1),e5548(1)];
Cstd_db =
[MFC(2),MG3620(2),MG2522(2),MX922(2),XP340(2),D1112(2),D2655(2),e5548(2)];
Mds_db =
[MFC(3),MG3620(3),MG2522(3),MX922(3),XP340(3),D1112(3),D2655(3),e5548(3)];
Mstd_db =
[MFC(4),MG3620(4),MG2522(4),MX922(4),XP340(4),D1112(4),D2655(4),e5548(4)];
Yds_db =
[MFC(5),MG3620(5),MG2522(5),MX922(5),XP340(5),D1112(5),D2655(5),e5548(5)];
Ystd_db =
[MFC(6),MG3620(6),MG2522(6),MX922(6),XP340(6),D1112(6),D2655(6),e5548(6)];



%implement algorithm to get data for the test image
im = imread('555_diff_printer_paper/HP envy5548/555_HPENVY5549_cotton1.bmp');
colors = set_colors('HP envy5548');%put folder of image here
blur_factor = 0.05;
im = im(1:200,1:200,:);
[Cds,Cstd,Mds,Mstd,Yds,Ystd,Kds,Kstd] = get_drop_size(im,colors,blur_factor);
output = [Cds,Cstd,Mds,Mstd,Yds,Ystd,Kds,Kstd];
figure(2);
scatter3(Cds_db,Mds_db,Yds_db,'b');
title('drop size plot');
xlabel('C');
ylabel('M');
zlabel('Y');
hold;
scatter3(output(1),output(3),output(5),'r');

figure(3);
scatter3(Cstd_db,Mstd_db,Ystd_db,'b');
title('standard deviation plot');
xlabel('C');
ylabel('M');
zlabel('Y');
hold;
```

```
scatter3(output(2),output(4),output(6),'r');
%distance in drop size
dist = zeros(1,8);
dist2 = zeros(1,8);
for i = 1:8
    d_Cds = (output(1) - Cds_db(i))^2;
    d_Mds = (output(3) - Mds_db(i))^2;
    d_Yds = (output(5) - Yds_db(i))^2;
    dist(i) = sqrt(d_Cds + d_Mds + d_Yds);
end
min = dist(1);
printer = 1;
for i = 2:8
    if dist(i) < min
        min = dist(i);
        printer = i;
    end
end
switch (printer)
    case 1
        disp('MFC with ds');
    case 2
        disp('MG3620 with ds');
    case 3
        disp('MG2522 with ds');
    case 4
        disp('MX922 with ds');
    case 5
        disp('XP340 with ds');
    case 6
        disp('D1112 with ds');
    case 7
        disp('D2655 with ds');
    case 8
        disp('e5548 with ds');
end
%distance in std
for i = 1:8
```

```
    d_Cds2 = (output(2) - Cstd_db(i))^2;
    d_Mds2 = (output(4) - Mstd_db(i))^2;
    d_Yds2 = (output(6) - Ystd_db(i))^2;
    dist2(i) = sqrt(d_Cds2 + d_Mds2 + d_Yds2);
end
min2 = dist2(1);
printer2 = 1;
for i = 2:8
    if dist2(i) < min2
        min2 = dist2(i);
        printer2 = i;
    end
end
switch (printer2)
    case 1
        disp('MFC with std');
    case 2
        disp('MG3620 with std');
    case 3
        disp('MG2522 with std');
    case 4
        disp('MX922 with std');
    case 5
        disp('XP340 with std');
    case 6
        disp('D1112 with std');
    case 7
        disp('D2655 with std');
    case 8
        disp('e5548 with std');
end



function [Cds,Cstd,Mds,Mstd,Yds,Ystd,Kds,Kstd] =
get_drop_size(im,colors,blur_factor)
    im = imgaussfilt(im,blur_factor);
    im_orig = im;
    im = rgb2lab(im);
```

```
    [M,N,D] = size(im);
    data = reshape(im,[M*N,D]);



    numColors = size(colors); numColors = numColors(1);
    centroids = colors;

    scores = assign_scores(data,centroids,numColors);

    clusters = assign_to_clusters(data,scores);

    final_im_rgb = create_final_image(im,clusters,centroids);
    final_im_rgb = uint8(final_im_rgb);

    subplot(1,2,1); imshow(im_orig);
    subplot(1,2,2); imshow(final_im_rgb);

    color_lists = create_color_lists(clusters,numColors);

    cyan_list = color_lists(:,1) + color_lists(:,7) + color_lists(:,5);
    magenta_list = color_lists(:,2) + color_lists(:,7) + color_lists(:,6);
    yellow_list = color_lists(:,3) + color_lists(:,5) + color_lists(:,6);
    black_list = color_lists(:,8);

    C_cc = bwlabel(reshape(cyan_list,[M,N,1])); num_cyan_drops =
max(max(C_cc)); num_cyan_px = sum(cyan_list);
    M_cc = bwlabel(reshape(magenta_list,[M,N,1])); num_magenta_drops =
max(max(M_cc)); num_magenta_px = sum(magenta_list);
    Y_cc = bwlabel(reshape(yellow_list,[M,N,1])); num_yellow_drops =
max(max(Y_cc)); num_yellow_px = sum(yellow_list);
    K_cc = bwlabel(reshape(black_list,[M,N,1])); num_black_drops =
max(max(K_cc)); num_black_pix = sum(black_list);

    [Cds,Cstd] = find_drop_sizes(C_cc);
    [Mds,Mstd] = find_drop_sizes(M_cc);
```

```
    [Yds,Ystd] = find_drop_sizes(Y_cc);
    [Kds,Kstd] = find_drop_sizes(K_cc);
end



function [drop_size,deviation] = find_drop_sizes(CC)
    numDrops = max(max(CC));
    [M,N] = size(CC);
    list = zeros(numDrops,1);
    counter = 1;
    for i = 1:M
        for j = 1:N
            if(CC(i,j) ~= 0)
                list(CC(i,j)) = list(CC(i,j)) + 1;
                counter = counter+1;
            end
        end
    end
    drop_size = mean(list);
    deviation = std(list) / drop_size;
end

function out = create_color_lists(clusters,numColors)
    [M,N] = size(clusters);
    out = zeros(M,numColors);
    for i = 1:M
       for k = 1:numColors
          if(clusters(i) == k)
             out(i,k) = 1;
          end
       end
    end
end

function final_im_rgb = create_final_image(im,clusters,centroids)
    [M,N,D] = size(im);
    [Mc,Nc] = size(clusters);
```

```
    final_im = zeros(Mc,D);
    for i = 1:Mc
       final_im(i,:) = centroids(clusters(i),:);
    end
    final_im = reshape(final_im,[M,N,D]);
    final_im_rgb = lab2rgb(final_im) .* 255;
end

function clusters = assign_to_clusters(data,scores)
    [M,N] = size(data);
    clusters = zeros(M,1);
    for i = 1:M
       [minVal,minIdx] = min(scores(i,:));
       clusters(i,1) = minIdx;
    end
end

function scores = assign_scores(data,centroids,numColors)
    [M,N] = size(data);
    scores = zeros(M,numColors);
    for i = 1:M
       for k = 1:numColors
           a = double(data(i,:)); b = double(centroids(k,:));
          scores(i,k) = color_dist(a,b);
       end
    end
end

function dist = color_dist(source,target)
    w = source(1) - target(1); w = double(w);
    x = source(2) - target(2); x = double(x);
    y = source(3) - target(3); y = double(y);
%     z = source(4) - target(4); z = double(z);
%     dist = w^2 + x^2 + y^2 + z^2;
    dist = w^2 + x^2 + y^2;
    dist = sqrt(dist);
end
```

```
function out = set_colors(printer)
    switch(printer)
        case 'Canon MG3620'
            cyan = [86,135,194];
            magenta = [190,105,148];
            yellow = [184,152,91];
            white = [192,194,203];
            green = [109,120,95];
            red = [187,93,66];
            blue = [95,102,142];
            black = [0,0,0];
        case 'Epson XP340'
            cyan = [90,147,206];
            magenta = [196,102,113];
            yellow = [204,175,72];
            white = [220,210,215];
            green = [108,147,125];
            red = [197,80,118];
            blue = [80,105,126];
            black = [0,0,0];
        case 'HP envy5548'
            cyan = [90,143,172];
            magenta = [153,86,116];
            yellow = [203,159,87];
            white = [183,188,190];
            green = [94,123,110];
            red = [166,70,67];
            blue = [82,74,96];
            black = [0,0,0];
        case 'Brother MFC'
            cyan = [96,147,207];
            magenta = [179,93,114];
            yellow = [222,195,89];
            white = [204,199,206];
            green = [99,127,96];
            red = [192,92,80];
```

```
        blue = [116,106,152];
        black = [57,43,45];
    case 'CANON MG2522'
        cyan = [84,148,199];
        magenta = [174,92,122];
        yellow = [199,167,77];
        white = [196,203,212];
        green = [89,122,73];
        red = [177,73,46];
        blue = [68,74,100];
        black = [0,0,0];
    case 'Canon MX922'
        cyan = [115,156,189];
        magenta = [196,115,141];
        yellow = [196,156,78];
        white = [206,201,209];
        green = [114,128,103];
        red = [174,85,58];
        blue = [128,114,158];
        black = [0,0,0];
    case 'HP Deskjet1112_Best'
        cyan = [83,139,179];
        magenta = [151,84,114];
        yellow = [198,171,96];
        white = [194,199,194];
        green = [104,135,123];
        red = [172,106,87];
        blue = [120,81,105];
        black = [0,0,0];
    case 'HP Deskjet2655'
        cyan = [92,144,182];
        magenta = [154,94,109];
        yellow = [189,171,116];
        white = [188,183,177];
        green = [102,132,107];
        red = [177,113,83];
        blue = [90,75,91];
        black = [0,0,0];
```

```
        otherwise
            error('invalid printer class');

    end

    out = [cyan;magenta;yellow;white;green;red;blue;black];
    out = rgb2lab(out ./ 255);
end
```

Appendix (2)

MATLAB CODE for Centroid generator:

```
clc; clear;

im = imread('555_diff_printer_paper\HP Deskjet2655\555_HP2655_cotton1.bmp');
colors = set_colors('HP Deskjet2655');%put folder of image here
blur_factor = 0.1;
im = imgaussfilt(im,blur_factor);
drop_size_cutoff = 50;
%min number of pixels to be considered a drop of ink
im_orig = im;
im = rgb2lab(im);

[M,N,D] = size(im);
data = reshape(im,[M*N,D]);
numColors = size(colors);
numColors = numColors(1);
C_centroids = colors;
scores = assign_scores(data,C_centroids,numColors);
clusters = assign_to_clusters(data,scores);

color_lists = create_color_lists(clusters,numColors);
color_lists = reshape(color_lists,[M,N,numColors]);
cyan_list = color_lists(:,:,1) + color_lists(:,:,7) + color_lists(:,:,5);
```

```
magenta_list = color_lists(:,:,2) + color_lists(:,:,7) + color_lists(:,:,6);
yellow_list = color_lists(:,:,3) + color_lists(:,:,5) + color_lists(:,:,6);
black_list = color_lists(:,:,8);

C_cc = bwlabel(cyan_list); num_cyan_drops = max(max(C_cc)); num_cyan_px =
sum(cyan_list);
%dilation stuff here
%dilation stuff here
C_dil = erode_dilate(C_cc,20,3,4);
C_cc_large = eliminate_small_drops(C_dil,drop_size_cutoff);
%setup ends up here
[C_centroids,C_rads,C_areas,C_drop_area] = get_circles(C_cc_large);

function out = erode_dilate(im,resize_scale,radius,num_iter)
    se = offsetstrel('ball',radius,radius);
    im = uint8(imbinarize(im)); im = imresize(im,resize_scale);

    for i=1:num_iter
        im = imerode(im,se);
    end
    for i = 1:num_iter/2
       im = imdilate(im,se);
    end

    out = imresize(im,1/resize_scale);
end

function ConnComp = eliminate_small_drops(ConnComp,drop_size_cutoff)
    drop_area = regionprops(ConnComp,'Area');
    drop_area = cat(1,drop_area.Area);
    [M,N] = size(ConnComp);

    %Eliminate ink drops below a certain size
    ConnCompSize = zeros(M,N);
    for i = 1:M
       for j = 1:N
            idx = ConnComp(i,j);
```

```
            if(idx ~= 0)
               c = drop_area(idx);
               if(c <= drop_size_cutoff)
                  c = 0;
               end
               ConnCompSize(i,j) = c;
            end
        end
    end
    ConnComp = bwlabel(ConnCompSize);
end

function [centroids,rads,areas,drop_area] = get_circles(ConnComp)
    numDrops = max(max(ConnComp));
    centroids = regionprops(ConnComp,'Centroid');
    centroids = cat(1,centroids.Centroid);

    bounds = regionprops(ConnComp,'BoundingBox');
    bounds = cat(1,bounds.BoundingBox);



    rads = zeros(numDrops,1);

    for i = 1:numDrops
        cr = ((bounds(i,3)/2)^2) + ((bounds(i,4)/2)^2);
        rads(i,1) = sqrt(cr);
    end
    areas = pi * (rads .^ 2);

    drop_area = regionprops(ConnComp,'Area');
    drop_area = cat(1,drop_area.Area);
end
```

MATLAB CODE for NDSDF:

```
close all;
```

```
clear;
% Nearest dot sector density functinon
%Start from the first dot location
%Cyan
C_centroids = xlsread('CANON_MX922_cotton1.xlsx');
C_len = length(C_centroids);
nearest_dots = zeros(C_len,2);
angles = zeros(C_len, 1);
index = 2;
[z, p1, p2, p3, n1, n2, n3] = deal(0);
for i = 1:C_len
    min_dist = 5000;
    first_dot = [C_centroids(i,1) C_centroids(i,2)];
    for j = 1:C_len
        if (j ~= i)
            next_dot = [C_centroids(j,1) C_centroids(j,2)];
            E_dist = sqrt((next_dot(1)- first_dot(1))^2 + (next_dot(2) -
first_dot(2))^2);
            if E_dist < min_dist
                min_dist = E_dist;
                index = j;
                nearest_dots(i,1) = C_centroids(index,1);
                nearest_dots(i,2) = C_centroids(index,2);
            end
        end
    end
    angles(i) = atand((nearest_dots(i,2) - first_dot(2))/(nearest_dots(i,1) -
first_dot(1)));
    if ((~isnan(angles(i))) && (angle(i) < -90))
        angles(i) = angles(i) + 180;
    elseif ((~isnan(angles(i))) && (angle(i) > 90))
        angles(i) = angles(i) - 180;
    end
end
%count clusters
for i = 1: C_len
    if ((angles(i) < 15) && (angles(i) > -15))
                z = z + 1;
```

```
            elseif ((angles(i) < 45) && (angles(i) > 15))
                p1 = p1 + 1;
            elseif ((angles(i) < 75) && (angles(i) > 45))
                p2 = p2 + 1;
            elseif ((angles(i) < 90) && (angles(i) > 75))
                p3 = p3 + 1;
            elseif ((angles(i) < -15) && (angles(i) > -45))
                n1 = n1 + 1;
            elseif ((angles(i) < -45) && (angles(i) > -75))
                n2 = n2 + 1;
            elseif ((angles(i) < -75) && (angles(i) > -90))
                n3 = n3 + 1;
    end
end
figure(1)
sectors = [p3/C_len p2/C_len p1/C_len z/C_len n1/C_len n2/C_len n3/C_len];
bar([-3 -2 -1 0 +1 +2 +3],sectors);
title('total percentage of centroids in specific sectors');
figure(2)
scatter(C_centroids(:,1),C_centroids(:,2),30,'filled');
figure(3)
for i = 1:C_len
    plot([C_centroids(i,1) nearest_dots(i,1)], [C_centroids(i,2)
nearest_dots(i,2)], 'o-');
    hold on;
end
angle_sum = 0;
count = 0;
for i = 1:(C_len - 1)
    for j = 2:C_len
        if (~isnan(angles(i)) && ~isnan(angles(j)))
        angle_sum = angle_sum + abs(angles(j) - angles(i));
        count = count + 1;
        end
    end
end
angle_change = angle_sum / count;
display(angle_change);
```