Assignment 2 Report

Yue Hao, yhao3@gmu.edu

1 Summary of the two methods

Both methods use the same idea from paper [1], however there are few major differences which can be summarized in Table 1.

Table 1: Major Algorithmic Differences of the Two Methods

	Hedcuter	Voronoi
Initial Sites Distribution	Gaussian	Uniform
Alg. for Voronoi	Image Propagation	Fortune
Cell Representation	Discretized Points	Polygon
Metric of Displacement	Manhattan	Euclidean
Disk Color	Cell Avg. Color	Controid Color
Disk Radius	Cell Avg. Intensity	Cell Max Intensity

1.1 hedcuter method

1. Centroidal Voronoi Tessellation (CVT)

   The algorithm for generating CVT is summerized in Algorithm 1.

   Note there is an option in Algorithm 1 Line 5, that the method can also use the maximum Manhattan distance as a metric for displacement besides the average.

   Algorithm 2 shows that hedcuter collect n points that randomly spreaded w.r.t. the intensity of greyscale image as initial sites.

Algorithm 1: Centroidal_Voronoi_Tessellation(I,n,d_t)

```
Input  : $I$: a grayscale image
         $n$: the number of points to be collected
         $d_t$: a user defined threshold for the average displacement
Output: $C$: a collection of cells representing the CVT
         where $c.s$ is the 2D coordinate of site of cell $c$ in $C$
         and $c.P$ is a collection of 2D coordinates marking the coverage of the cell
1  $C = \{\}$
2  $P$ = SampleInitialPoints ($I$,$n$)
3  do
4      $V$ = Voronoi ($I$,$P$)
5      $P_c$ = Centroidal ($I$,$V$)
6      $d = \frac{\sum_{p,p_c|p\in P,p_c\in P_c} |p.x-p_c.x|+|p.y-p_c.y|}{|P|}$
7      $P := P_c$
8  while $d > d_t$;
9  for $p \in P$ do
10     $c.s = p$
11     $c.P$ = points in $V$ composing the cell that sited on $p$
12     $C = C \cup \{c\}$
13 end
14 return $C$
```

Algorithm 2: SampleInitialPoints(I,n)

```
Input  : $I$: a grayscale image w/ $I(p)$ being the intensity of pixel at $p$
         $n$: the number of points to be collected
Output: $P$: a set of 2D coordinates of collected points
1  $P = \{\}$
2  while $|P| < n$ do
3      $p$ = draw a coordinate uniformly random on image $I$
4      $i = I(p)$  $g$ = draw a random variable following a Gaussian distribution
5      if $i < g$ then
6          $P = P \cup p$
7      end
8  end
9  return $P$
```

2. Computing Voronoi Diagram

   The method uses a image wave propagtion algorithm to calculate voronoi diagram which is summerized in Algorithm 3.

```
Algorithm 3: Voronoi($I$,$P$)
Input  : $P$: a set of 2D coordinates
         $I$: a grayscale image $I$ w/ $I(p)$ being the intensity pixel at $p$
Output: $V$: a collection of cells representing the Voronoi Diagram,
         and $v.P$ is a collection of 2D coordinates marking the coverage of the cell
1  $H = \{\}$ (a heap data structure holding the points)
2  $D = \{\}$ (an image sized contrainer)
3  $R = \{\}$ (an image sized contrainer)
4  for $p_i \in P$ do
5      put $p_i$ in heap $H$ according to ColorDist ($I(p_i)$)
6      $D(p_i)$ = ColorDist ($p_i$)
7      $R(p_i) = i$
8  end
9  while $H$ is not empty do
10     $p$ = draw a point from the back of the heap $H$
11     for each point $p_n$ neighboring $p$ do
12         $d_n = D(p) +$ ColorDist($p_n$)
13         if $d_n < D(p)$ then
14             $D(p_n) = d_n$
15             $R(p_n) = R(p)$
16             put $p_n$ in heap $H$ according to ColorDist ($I(p_n)$)
17         end
18     end
19 end
20 for all points coordinates $p$ on image do
21     put all the points with the same $R(p)$ in the same voronoi cell $v_{R(p)}.P \in V$
22 end
23 return $V$
```

3. Stippling

   Finally, hedcuter method uses Algorithm 4 to generate the stippling disks.

```
Algorithm 4: Create_Disks($M$,$I$,$C$)
  Input  : $M$: an RGB color image w/ $M(p)$ being the RGB color of pixel at $p$
           $I$: a grayscale image $I$ w/ $I(p)$ being the intensity pixel at $p$
           $C$: a collection of cells representing the CVT
           where $c.s$ is the 2D coordinate of site of cell $c$ in $C$
           and $c.P$ is a collection of 2D coordinates marking the coverage of the cell
  Output: $D$: a collection of disks of the Stipples
           where $d.c$ is the 2D coordinates of the center of $d$ in $D$,
           $d.rgb$ is the RGB color and $d.r$ is the radius
1 $D = \{\}$
2 for $c \in C$ do
3     $d.c = c.s$
4     $d.rgb = \frac{\sum_{p \in c.P} M(p)}{|c.P|}$
5     $i = \frac{\sum_{p \in c.P} I(p)}{|c.P|}$
6     $d.r = \frac{100 * r}{i + 100}$ where $r$ is a default constant
7     $D = D \cup \{d\}$
8 end
9 return $D$
```

1.2 voronoi method

1. Centroidal Voronoi Tessellation (CVT)

   The algorithm for generating CVT is summerized in Algorithm 5, which is very similar to the one in hedcuter methods, the only difference are the cell representation, methods to create initial samplings, and distance metric.

   Algorithm 6 shows that hedcuter collect n points that randomly spreaded w.r.t. the intensity of greyscale image as initial sites.

Algorithm 5: Centroidal_Voronoi_Tessellation(I,n,d_t)

```
Input  : I: a grayscale image
         n: the number of points to be collected
         d_t: a user defined threshold for the average displacement
Output: C: a collection of cells representing the CVT
         where c.s is the 2D coordinate of site of cell c in C
         and c.V is a collection of 2D coordinates marking the boundary of the cell
         and c.E is a collection of edges marking the boundary of the cell
1  $C = \{\}$
2  $P$ = CreateInitialDistribution ($I$,$n$)
3  do
4      $V$ = VoronoiFortune ($I$,$P$)
5      $P_c$ = Centroidal ($I$,$V$)
6      $d = \frac{\sum_{p,p_c|p\in P,p_c\in P_c} \sqrt{(p.x-p_c.x)^2+(p.y-p_c.y)^2}}{|P|}$
7      $P := P_c$
8  while $d > d_t$;
9  for $p \in P$ do
10     $c.s = p$
11     $c.V$ = points in $V$ bounding the cell that sited on $p$
12     $c.E$ = edges in $V$ bounding the cell that sited on $p$
13     $C = C \cup \{c\}$
14 end
15 return $C$
```

Algorithm 6: CreateInitialDistribution(I,n)

```
Input  : I: a grayscale image w/ I(p) being the intensity of pixel at p
         n: the number of points to be collected
Output: P: a set of 2D coordinates of collected points
1  $P = \{\}$
2  while $|P| < n$ do
3      $p$ = draw a coordinate uniformly random on image $I$
4      $i = I(p)$  $g$ = draw a random variable uniformly random
5      if $i < g$ then
6          $P = P \cup p$
7      end
8  end
9  return $P$
```

2. Computing Voronoi Diagram

   This method uses the Fortune's algorithm which was fully discussed in lecture. For the sake of brevity, the pseudocode is neglected here.

3. Stippling

   The stippling technique is very similar to hedcuter method, where only minor difference is the color of disk is solely determined on the controid's color (shown in Alg. 7 Line 4), slightly different handling of radius (shown in Alg. 7 Line 6-7).

```
Algorithm 7: Stippling($M$,$I$,$C$)
Input  : $M$: an RGB color image w/ $M(p)$ being the RGB color of pixel at $p$
         $I$: a grayscale image $I$ w/ $M(p)$ being the intensity pixel at $p$
         $C$: a collection of cells representing the CVT
         where $c.s$ is the 2D coordinate of site of cell $c$ in $C$
         and $c.V$ is a collection of 2D coordinates marking the boundary of the cell
         and $c.E$ is a collection of edges marking the boundary of the cell
Output: $D$: a collection of disks of the Stipples
         where $d.c$ is the 2D coordinates of the center of $d$ in $D$,
         $d.rgb$ is the RGB color and $d.r$ is the radius
1  $D = \{\}$
2  for $c \in C$ do
3      $d.c = c.s$
4      $d.rgb = M(c.s)$
5      $P$ = sampling points in $c$
6      $i = \sum_{p \in P} I(p)$
7      $d.r = r * \frac{i}{max(i \text{ for all } c \in C)}$ where $r$ is a default constant
8      $D = D \cup \{d\}$
9  end
10 return $D$
```

2 Comparison of the two methods

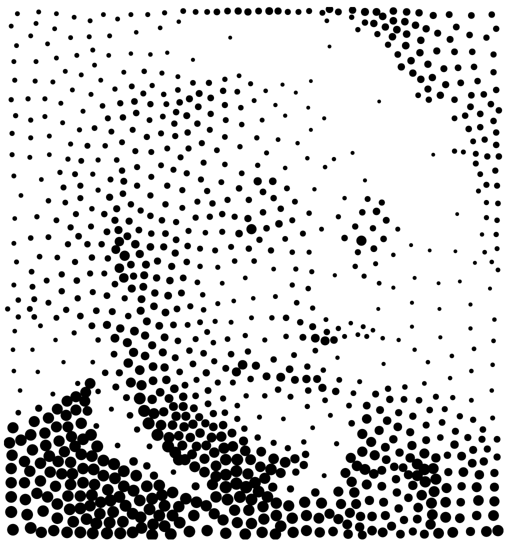
Unless otherwise specified, all the output in this section are using the following settings for both methods in Table 2, these parameters are tunes for the purpose of unifying the output for both algorithms (shown in question 1), and used as baseline for comparison the output with various parameters/input adjusted in the remaining questions.

Table 2: Baseline Parameters

	Hedcuter	Voronoi
Num of points	1000	1000
Threshold of Displacement	1.0	0.14
Default Radius	7	0,7
Color	Black	Black
Sub-pixels	1	5

For the hedcuter, GPU is NOT used and max iteration is set to ∞.

1. Do you get the same results by running the same program on the same image multiple times?


(a) 1st run

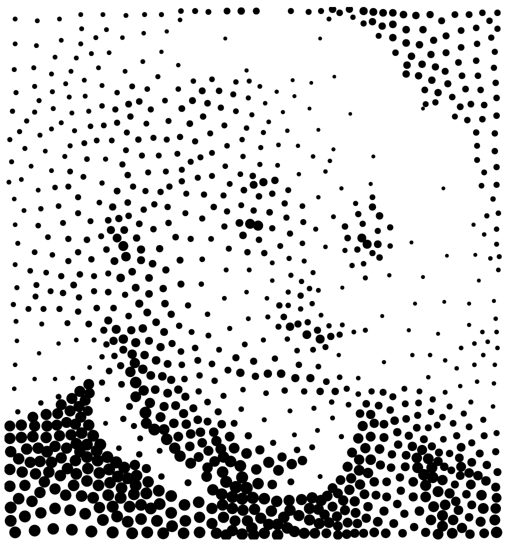
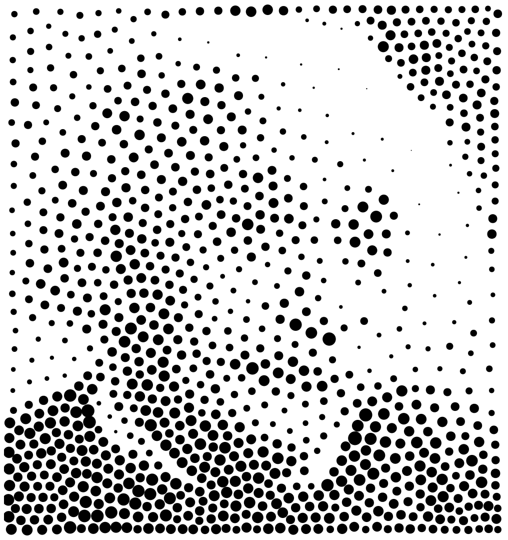

(b) 2nd run

Figure 1: Hedcuter


(a) 1st run

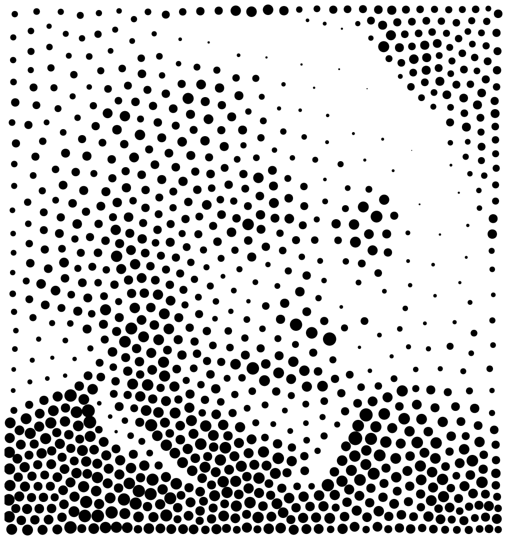

(b) 2nd run

Figure 2: Voronoi

I do NOT get the same results running hedcuter on the same image, due the the randomness introduced when sampling the initial sites.

I do get the same results running voronoi. Although the initial sites are also generated randomly, the random seed that voronoi uses is *boost::mt19937*, which creates a pseudo-random number generator. A numbers the random number produced will be the same every time the program is run.

2. If you vary the number of the disks in the output images, do these implementations produce the same distribution in the final image? If not, why?

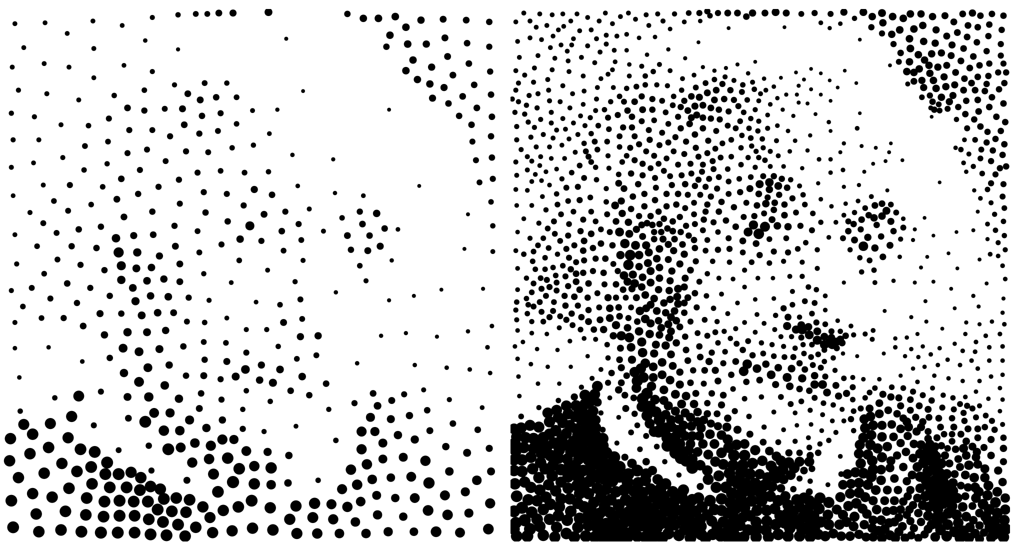

(a) 500 disks (b) 2000 disks

Figure 3: Hedcuter

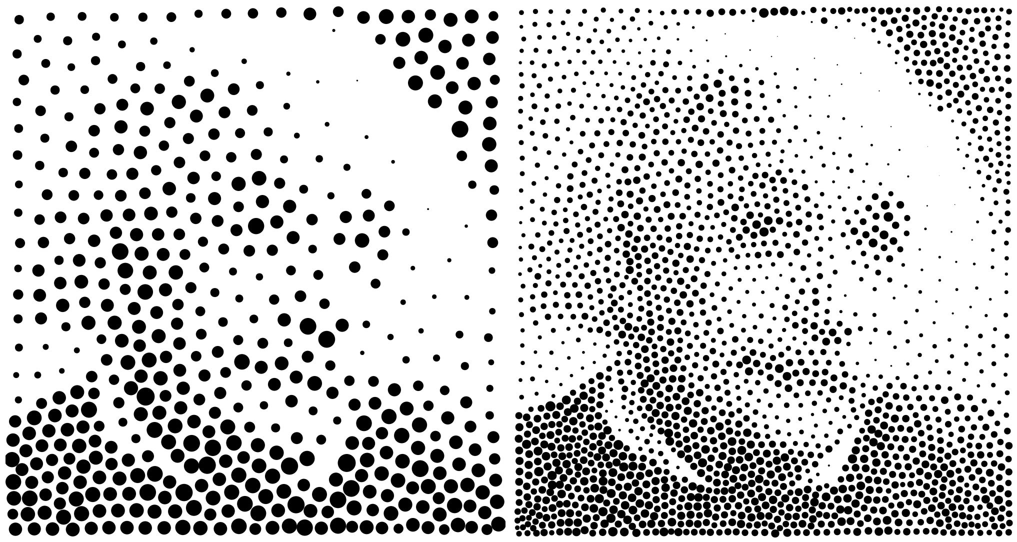
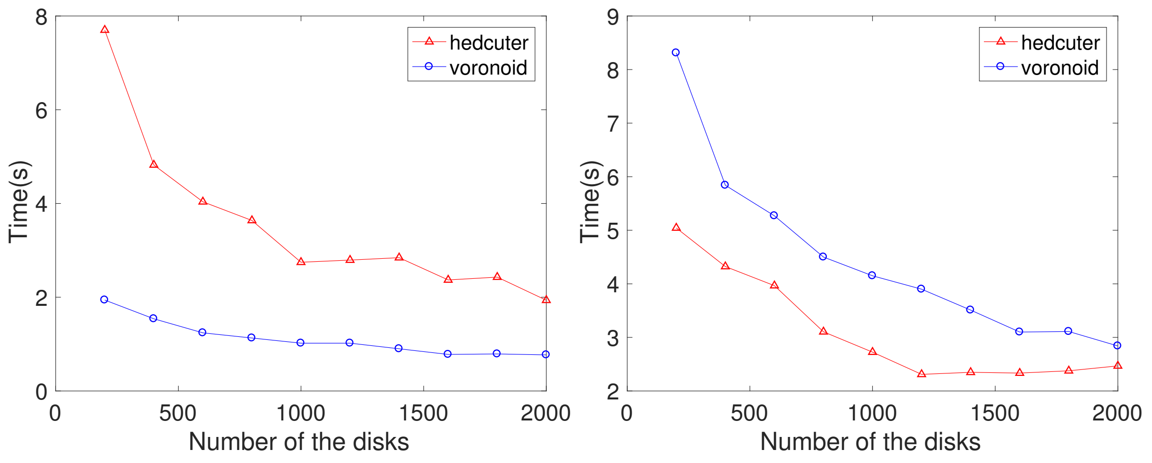

(a) 500 disks (b) 2000 disks

Figure 4: Voronoi

The disk distributions are different between hedcuter and voronoi, as hedcuter biases on darker region based on a Gaussian distribution, while voronoi is more of an uniform distribution. Vary the number of disks does not affect the distribution among both algorithm themselves as the distribution of random sampling is independent of how many samples your use.

3. If you vary the number of the disks in the output images, is a method faster than the other?

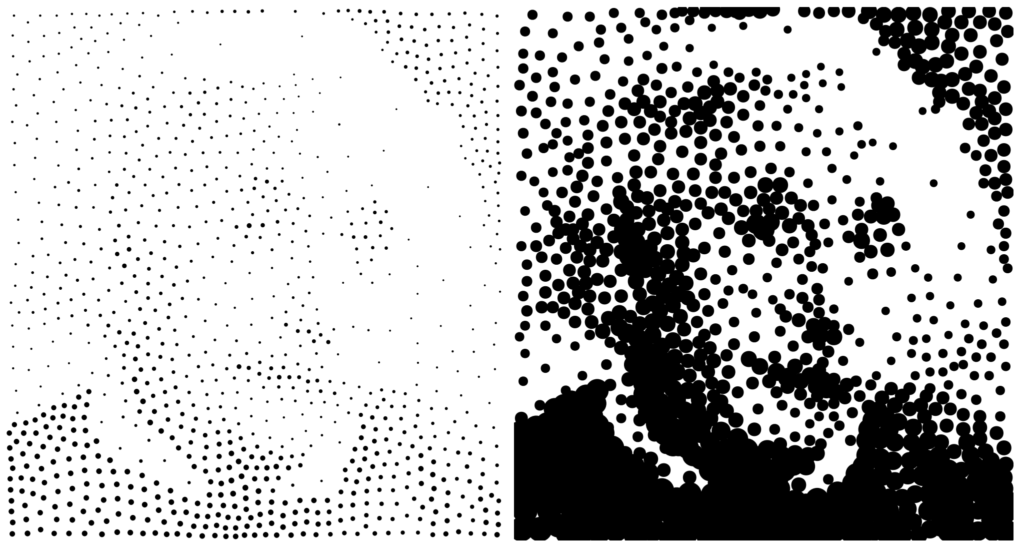


(a) Threshold: hedcuter = 1, voronoi = 0.5 (b) Threshold: hedcuter = 1, voronoi = 0.14

Figure 5: Running time vs number of disks.

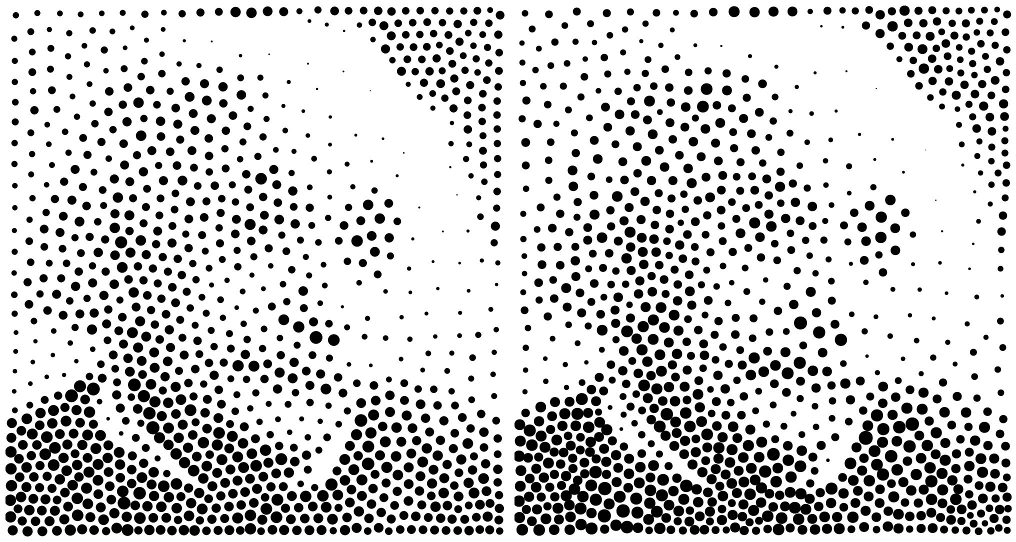
No method is strictly superior to the other in terms of running time. As it depends on other parameters besides the number of disks. However, both methods seem to require less time to converge for larger number of disks.

4. Does the size (number of pixels), image brightness or contrast of image increase or decrease their difference?



(a) Large image size (b) Small image size

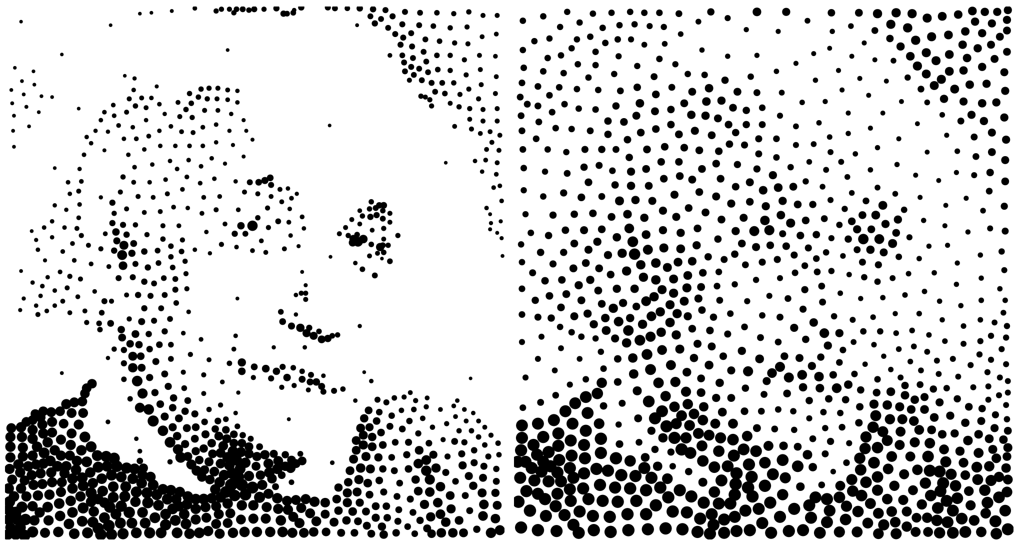
Figure 6: Hedcuter



(a) Large image size (b) Small image size

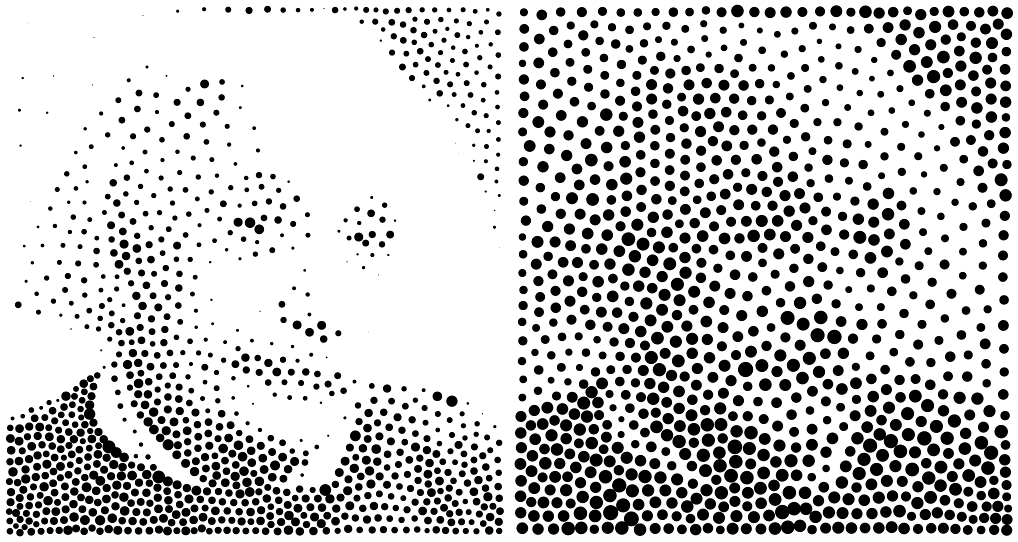
Figure 7: Voronoi

The sizes of the images have great impact on the size of the radius of the disks generated using hedcuter method, while voronoi method is consistent to the variance of image size. Because hedcuter calcuate the average intensity of the cell and this will impacted by the cell size, while voronoi uses the max intensity of the cell.



(a) High Brightness (b) Low Brightness

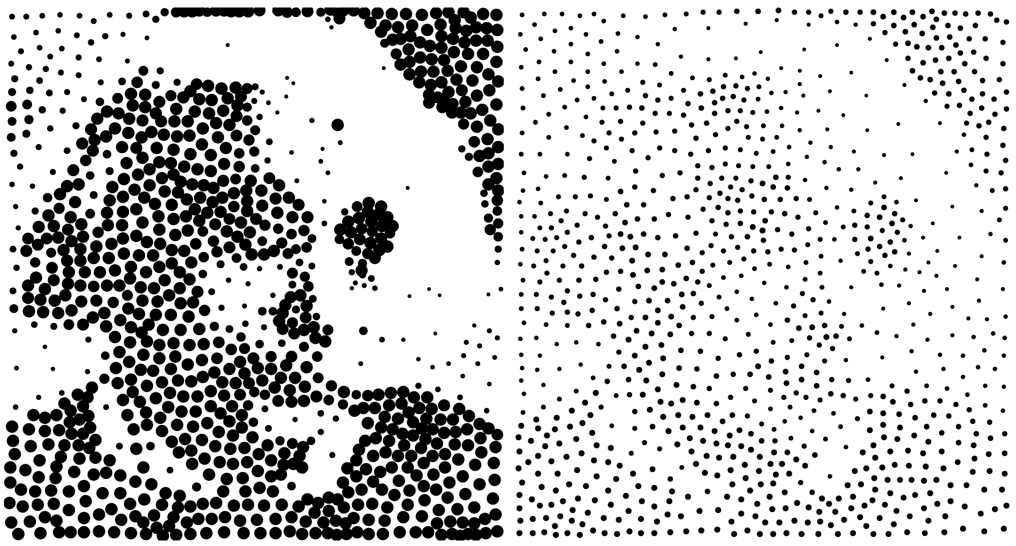
Figure 8: Hedcuter



(a) High Brightness (b) Low Brightness

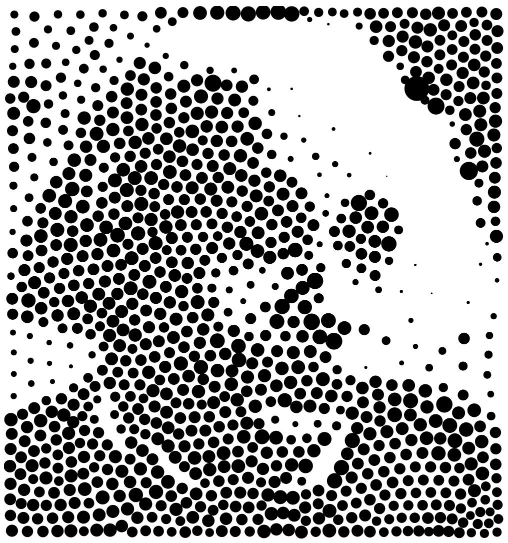
Figure 9: Voronoi

Both methods are affected greatly by the brightness of the image, as the darker pixels prevail the image. However, the hedcuter seems maintain a better consistency due to the way it calculate the disk radius.



(a) High Contrast (b) Low Contrast

Figure 10: Hedcuter


(a) High Contrast

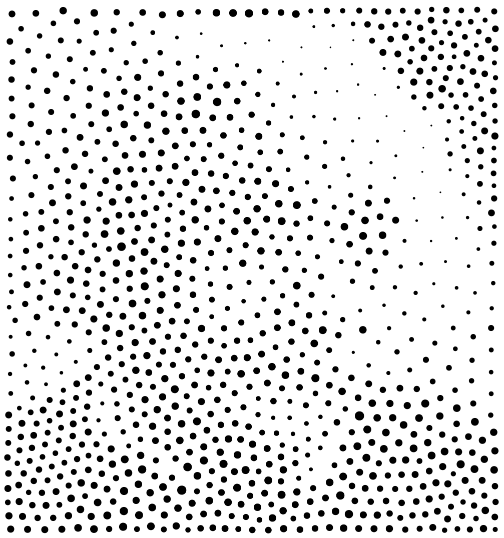
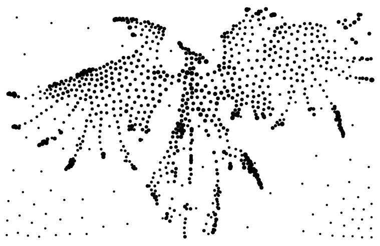
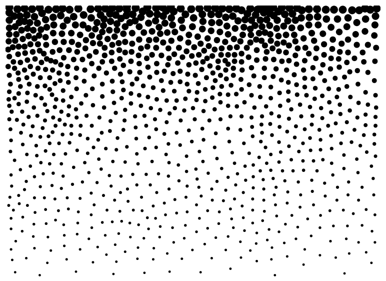

(b) Low Contrast

Figure 11: Voronoi

Contrast is not a major factor that would increase of decrease the difference between the two methods.

5. Does the type of image (human vs. machine, natural vs. urban landscapes, photo vs. painting, etc) increase or decrease their difference?


(a) Phoenix


(b) Gradient

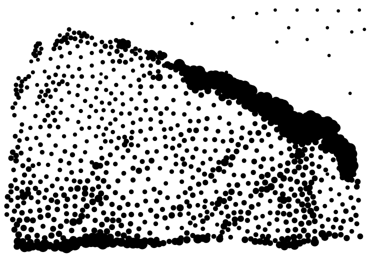
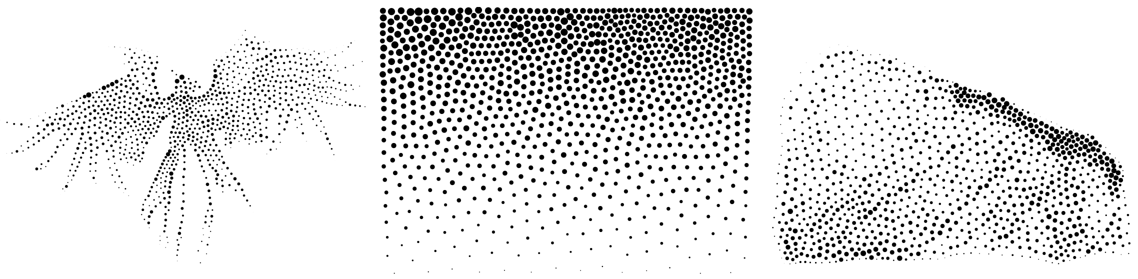

(c) Erinking

Figure 12: Hedcuter



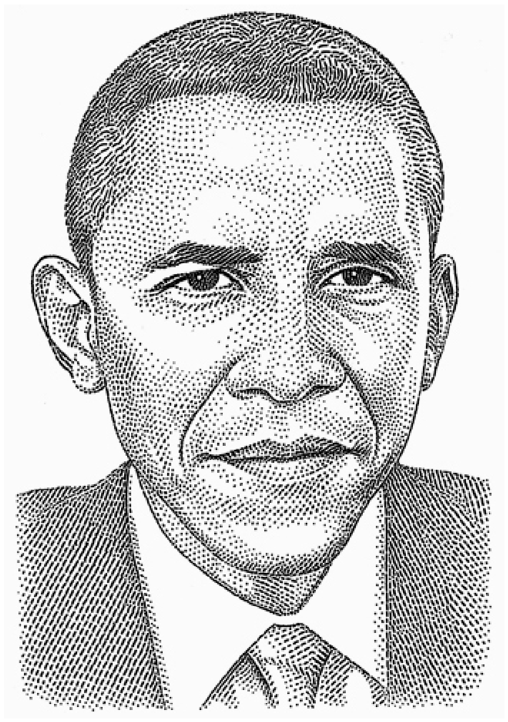
(a) Phoenix (b) Gradient (c) Erinking

Figure 13: Voronoi

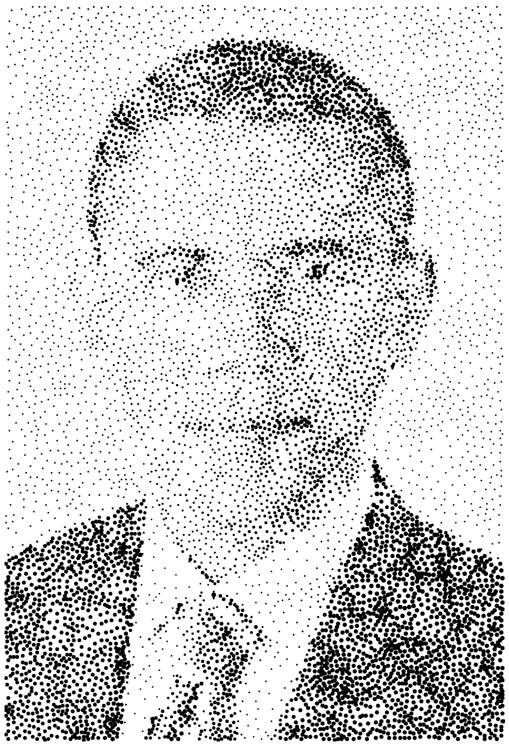
Based the observation on 12 and 13, different subjects in the image does not have much effect on the performance of the two methods.

6. Are the outputs of these stippling methods different the hedcut images created by artists (e.g. those from the Wall Street Journal)?

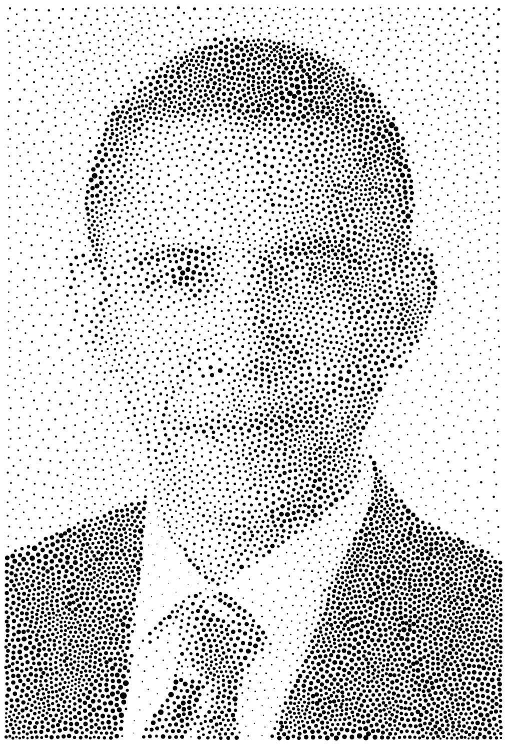
(a) Photo



(b) WSJ artist stippling



(c) Hedcuter



(d) Voronoi

Figure 14: Comparison of WSJ artist stippling vs outputs of these stippling methods.

Clearly, both methods cannot create a stippling on par with the human art work, especially the the distribution of dots. The artist will strengthen the edge of the object and the stippling shows better contrast. These are all based on the semantic understanding of the object. The computer generated is pure base on each independent pixels and has nothing to do with the semantics.

3 Improvement of hedcuter method

Improvement 1: Weighted Coloring

The ideas is the pixel the centroid has larger weight in determining the color of the disk than the pixels around the boundary. I employed the following weighted method to improve coloring in Algorithm 4 Line 4.

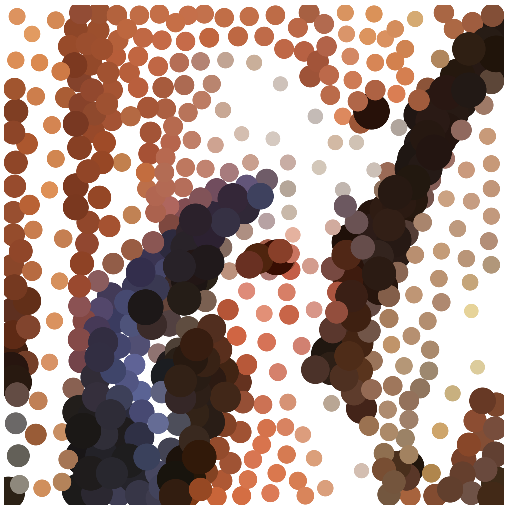
$$d.rgb = \frac{\sum_{p \in c.P} \xi_p \cdot M(p)}{\sum \xi_p}$$

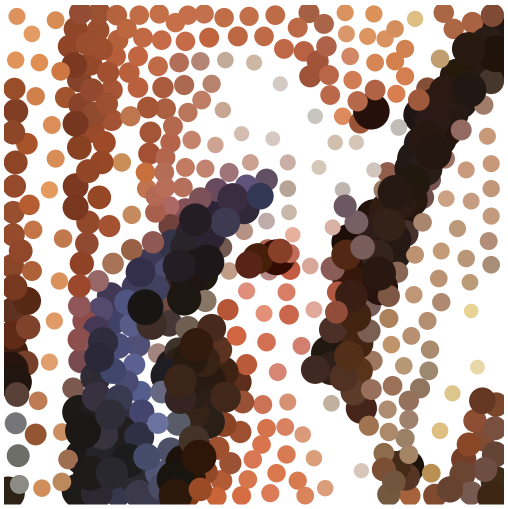
where

$$\xi_p \begin{cases} 1, & \text{if } p = p_c \\ \frac{1}{\text{euclidean\_dist}(p, p_c)}, & \text{otherwise} \end{cases}$$

given p_c is the centroid of the cell containing p.

(a) Input image: Lenna


(b) Disk color based on cell average


(c) Disk color based on weighted cell average

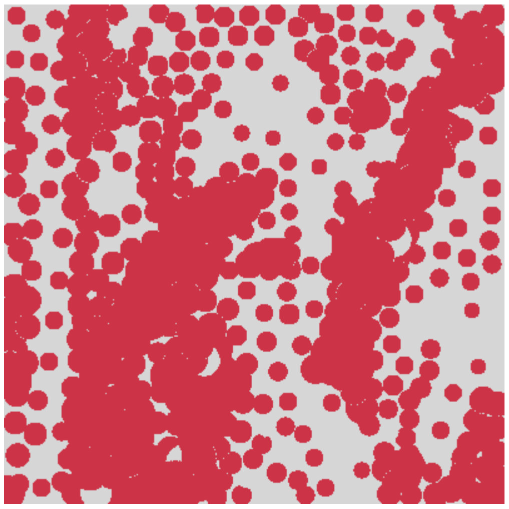

(d) Difference of (b) and (c)

Figure 15: Comparison of color based on average cell color vs. weight average cell color.

The results are shown in Figure 15, depending on the monitor, the color enhancement might be hard to tell. The different area of the two output are shown in 15 (d).

Improvement 2: Disk Radius Based on Cell Area

I employed the following weighted method to improve coloring in Algorithm 4 Line 6.

$$d.r = \frac{r \cdot \mathrm{area}(c)}{\max_{c \in C} (\mathrm{area}(c))}$$

The results are shown in Figure 16

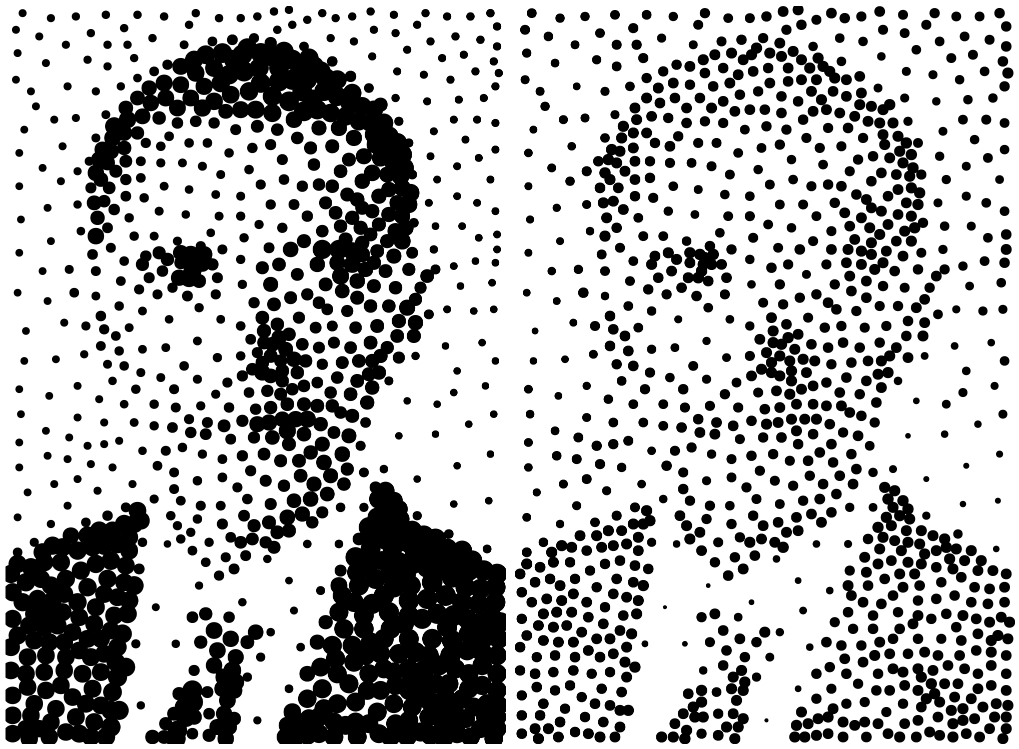

(a) Disk radius based on intensity (b) Disk radius based on cell area

Figure 16: Comparison of disk radius based on intensity vs. disk radius based on cell area

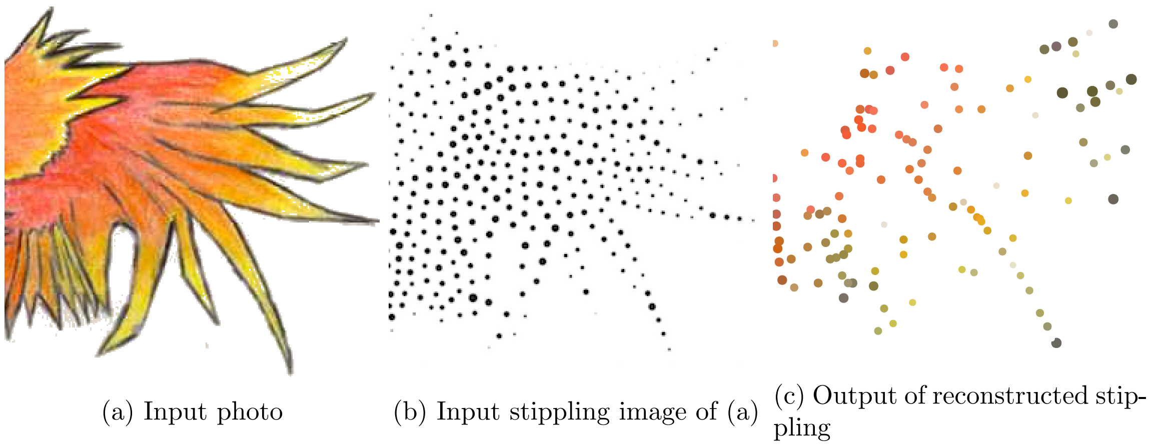
Improvement 3: Stippling Reconstruction

The idea is that we can obtain a stippling image created by artist, like the Wall Street Journal ones that are printed on a piece of paper, then we digitize it by extracting the stippling points (x, y) locations from a scanned copy. If we reconstruct a existing stippling, we can recreate many things like re-coloring based on an existing art work.

I implemented this functionality in the following step

1. collect all dark pixels in the scanned stippling (greyscale) image.
2. run a Density-based spatial clustering of applications with noise (DBSCAN)[2] on the extracted pixels locations, so that in the original image many dark pixels from the same disk are joined into one site.
3. build a voronoi diagram on the sites obtained in the previous step.

To show the result, I reconstructed a black and white stippling image and recolored it based on the reconstruction and the original photo. The results are shown in Figure 17 It is very difficult to obtain the original photo of a artist created stippling, so I just use voronoi method to generate a stippling then printed it in a image.



(a) Input photo (b) Input stippling image of (a) (c) Output of reconstructed stippling

Figure 17: Stippling reconstruction

4 Known Bugs and Limitations

The stippling reconstruction only works for small number of stippling points, mainly due to the computational complexity of the DBSCAN clustering algorithm.

References

[1] Adrian Secord. 2002. Weighted Voronoi stippling. In Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering (NPAR '02). ACM, New York, NY, USA, 37-43.

[2] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96), Evangelos Simoudis, Jiawei Han, and Usama Fayyad (Eds.). AAAI Press 226-231.