Fall 2017

# Voronoi Stippling Report

*Yongxin Wang* *Advance Algorithm Programming*

## 1 Summary of the two methods

Given an input image, the two mothods both try to generate an image filled with stipples which resembles the input image. To do this, the hedcutter and voronoi methods are following the four steps as below.

1. initialize sample points among the image, repeat from step2 to step4 until convergence
2. create voronoi diagram using the sample ponits
3. compute weighted centers for every voronoi region
4. move each sample point to its corresponding center

The difference between the hedcuter and voronoi methods are the different ways of generating voronoi diagram, which will be described in their individual sections. Then the two algorithms share the same procedure. In order to generate proper stipple image which resembles the original image, a method based the weighted centroidal voronoi diagram(CVT) is applied. In the process of computing weighted CVT, regions with higher values of (a density function) will pack generating points closer than regions with lower values[1]. The proposed methods use color intensity from the input image as , which in result packs more points around regions whose colors are closer to black while distributes fewer points around white regions. After several iterations this algorithm will generate an image with stipples which looks like the input image when configuration of parameters is good.

### 1.1 hedcuter method

The hedcuter method will compute voronoi diagram using the input image. The method is based on the property of voronoi diagram, which is that points inside the voronoi region is closer to its own site than all the other sites. For each sites, they propagate their labels to the four-connected neighbors and then those updated pixels will propagate labels to theirs neighbors and so on. This is processed in the manner of breadth-first search, which means at the end of each iteration every sites have reached out the same steps. Labels competing will happen when two sites have covered the same pixel and the site which is "closer" to such pixel will win. The hedcuter method will get the approximate voronoi diagram when there will be no update in terms of labels. The running time depends on three factors: the number of sites n; the resolution of the image (width, height) and the distribution of pixels. The lower bound is $O(width \times height)$ and the upper bound is $O(width \times height \times n)$.

### 1.2 voronoi method

The voronoi method uses Fortune's algorithm to generate voronoi diagram. Fortune's method is a sweep line algorithm and it can be computed in O(nlogn) time.

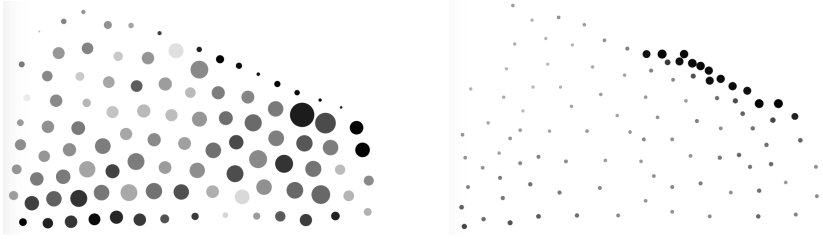
## 2 Comparison of the two methods

1. Output consistency: Voronoi method gives the same result every time given enough iteration, while hedcuter method's outputs vary. For the testing image, it seems that hedcuter will never converge using maximum distance termination condition, though it gives relatively good results after the first several iterations. The initial sample procedure and the number of iteration are factors of making the results differ from each other. They will affect the approximate voronoi diagram generated by hedcuter method.

(a) Hedcuter1 (b) Hedcuter2

Figure 1: Two results from Hedcuter on the same image

2. Distribution comparison. The distributions are not the same when the nubmer of the disk varies. When the number of disks drops, the voronoi's points evenly cover the image area while the hedcuter's points pack in the darker area. For the voronoi's method, the intensity information is easy to be filtered out when number of disks is low because there's a higher chance the intensity will be balanced out when the number of pixels in each region is high. While the hedcuter's method at the first step already takes intensity information into account. So they have different distribution when the number is low. Though when the number increases, the voronoi's method will focus on detailed region, where intensity information will take place.



(a) voronoi (b) hedcuter

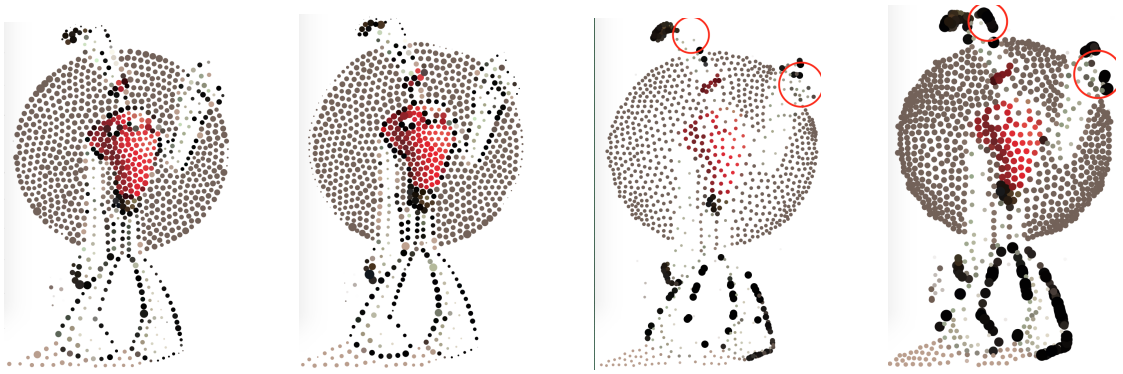
Figure 2: Left - voronoi's result; Right - hedcuter's result;

3. Speed comparison. Voronoi method is faster than hedcuter method when number of the disks increases. The voronoi's method can compute voronoi diagram in O(nlogn) time, while the hedcuter's method has an upper bound as $O(width \times height \times n)$. When the number of disks is high, the hedcuter's method tends to move to its upper bound because the chance that one pixel is updated increases when the number is high.

| #_of_disk | 100 | 1000 |
|---|---|---|
| voronoi | 4.34s | 1.61s |
| hedcuter | 1.09s | 5.82s |

(a) speed comparison

4. Their difference increases when the resolution of the image increases. When the resolution of the image is high, the hedcuter's method will capture more of the small_region details of the original image because it can propagate the information out. Though the voronoi's method will balance such small_region details with other sparse ares.



(a) voronoi-low-res (b) voronoi-high-res (c) hedcuter-low-res (d) hedcuter-high-res

Figure 3: resolution

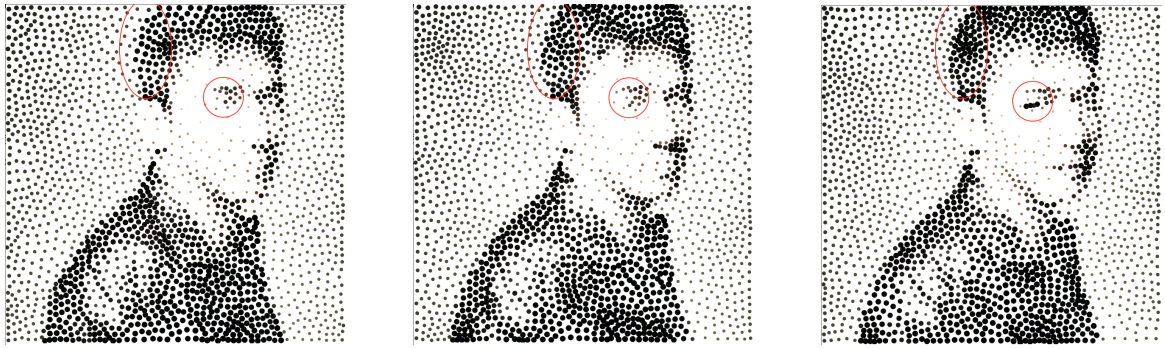
5. The outputs differ from the images created by artists. The artists uses lines instead of dots to represent line-property objects like hair and grid, while the program uses dots for all the objects. Also, the images created by the program have blurred borders.

(a) Artist (b) Voronoi (c) Hedcuter (d) Original

Figure 4: comparison with Wall Street Hedcut

## 3 Improvement of hedcuter method

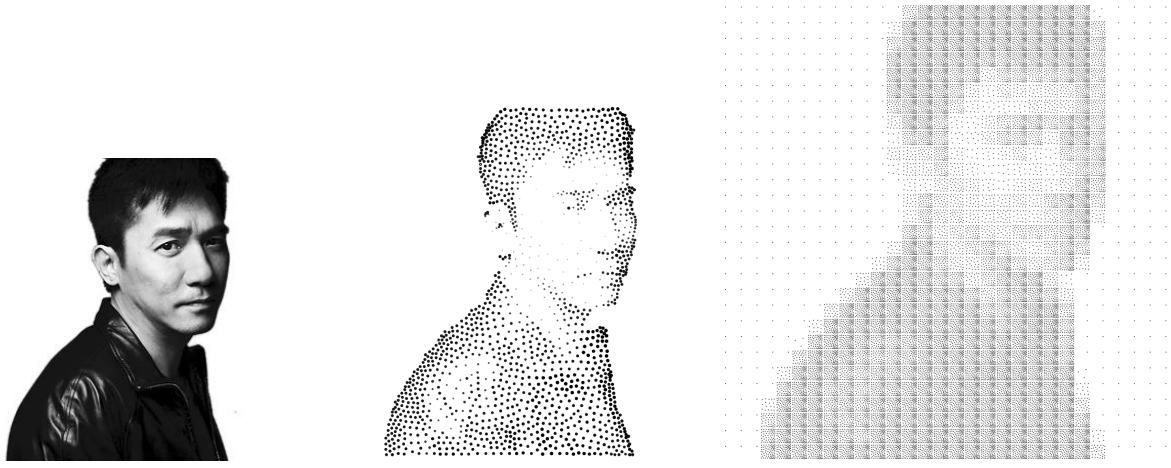
1. Decay. The program changes the propagation decay from 1.0 to 0.5, which leads to a more detailed representation of the original image. In the new method, the left eye shows up and the border of the head also becomes less blurred. When the decay rate is applied to propagation, the effect of the cell on farther pixels decreases. This in result makes the cell less involved with others and the details show up.



(a) decay: 1.5 (b) decay: 1.0 (c) decay: 0.5

Figure 5: Decay Demonstration

2. Pre-compute grid. This method uses the precompute method in the suggested paper, which construct the dot image by concatenate several 20-by-20 small dot images. The density of the small image is decided by the average image intenstiy in the corresponding image block in the input image.



(a) gray image (b) hedcut: 1500 (c) precompute

Figure 6: Precompute grid

3. Edge based hedcuter. The results in section2 show the weakness of hedcuter on discriminate between edges and non-edges. The proposed method thus makes an attempt to utilize the outcomes of canny edge detection. By putting more weight on non-edge pixels, the method appears to make some improvements on enhancing the visibility of edges on the examples below. The two lines circled in the final result(Right) shows the edges in the canny result(Left) that were not captured using hedcuter method(Middle) due to the closeness of intensity in the neighborhood.

(a) edge detection (b) hedcut (c) hedcut+edge

Figure 7: edge-present hedcuter