Fall 2017

# Stippling Algorithm Report

*Anson Rutherford* *CS 633 - Computational Geometry*

## 1 Summary of the two methods

### 1.1 hedcuter method

1. The main function processes the arguments and sets up hedcut.
2. hedcut uses these arguments to place the initial points in darker areas of the original image via rejection sampling, and passes it to wcvt.
3. wcvt iteratively generates new images by calculating new Voronoi diagrams. This repeats until the maximum number of iterations is reached, or the necessary adjustments are minor enough to be below the given threshold. Each new image is computed via the following method:
    a. A new copy of the last image is virtually resized to a higher resolution.
    b. For each Voronoi cell in the previous iteration, calculate and store the position and weight of the cell (calculated based on the value of the corresponding pixel in the original image). Store each of these weight-position pairs in a heap. Store weight and cell id in two materials as well. Clear the coverage of each Voronoi cell.
    c. For each cell, from its central pixel, find pixels that belong within the cell by repeatedly checking neighbors (similar to floodfill), and labeling them.
    d. Once all pixels have been visited and labeled, add each pixel to the "coverage" of that cell. Now you have a new collection of Voronoi sites describing the full image.
    e. The new Voronoi sites are adjusted to be placed at the weighted center of their cell. The weighted center is calculated incorporating the original image. Darker portions of the Voronoi cell will have higher weight, pulling the center closer to them. Over several iterations, this results in the Voronoi sites "moving" and clustering in the darker areas of the image, producing the desired effect.
4. Export the resulting image. Each Voronoi site is sent as a disk either black or of the average color of all the pixels within the corresponding Voronoi cell. If the disk size is not uniform, the disk size is calculated according to the number of pixels within the cell, and the grayscale value of the cell's average color.

### 1.2 voronoi method

1. The passed parameters are used to set up initial sites, also using rejection sampling.

2. The Voronoi diagram of the sites is calculated using the plane sweep method described in class. Site and vector events are processed as the plane sweeps the sites, delivering the edges of the Voronoi cells associated with the sites.

3. Each Voronoi cell has its site moved to the weighted center of the cell. Like in the hedcuter method, the cell's center is shifted towards darker areas of the cell, moving sites into the darker portions of the image.

4. The above two steps are repeated until the displacement of the step is below the provided threshold. Displacement is calculated as the average of the euclidean differences between the original cell sites and cell centers.

5. The final svg file is rendered. If the stipples are dynamically sized, the radius is calculated according to the size and intensity of the Voronoi cell. All radii may be multiplied by a sizing factor. If color is used, the color used is the color at the stipple's location in the original image.

## 2 Comparison of the two methods

- The Voronoi method outputs much faster than the hedcuter method. I think this is because the hedcuter method is checking the maximum distance moved each iteration, as opposed to the average, which the Voronoi method checks. The maximum distance is subject to noise, and can lead to situations where the vast majority of stipples have essentially locked themselves in place, save for a few stragglers. Taking the average displacement means that those few undecided stipples won't settle, but it will save a large number of iterations.

- The Voronoi method seems to produce a fairly wide range of radii when it is allowed to, whereas hedcut tends to keep its stipples roughly the same size. Because of this, hecut tends to have awkwardly sparse areas

- Because the hedcut method takes the average of the pixels within its cell to get the color, it tends to be less splotchy. Using the Voronoi method with color produces an amount of color noise.

- Both programs select the initial cells randomly, so while I never observed it directly, I would assume both programs produce slightly different results each time they are run. However, I would guess that Voronoi would be more likely to have more intense differences, because it stops when the average movement weakens, as opposed to going too far like hedcut does.

- Darker images seem difficult to approximate with hedcut, either the stipples are too small, producing a grainy effect, or too large, causing everything to "smoosh" together. Voronoi does a much better job filling dark areas in with regular but separated stipples. It generally does better with larger areas of solid color.

- The number of disks generally does not drastically change the nature of the images, other than the "resolution", except for outside a reasonable range.

- For drawn images, such as "phoenix" and "klaymen" the differences between the two methods tend to be less severe. For photos, however, you can easily tell the difference, due to the reasons mentioned above.

- manually done hedcuts, like those used in the wall street journal, seem to exhibit qualities not present in these results, such as regular patterns of shapes (concentric circles/rows), as well as more stark transitions between shadows.

# 3 Improvement of hedcuter method

- Added size weight (arg=100) optional argument, which changes the way the size of the disks is generated. The lower the value is, the MORE the relative size of the Voronoi cell should change the disk's size.

- Added anneal rate (arg=1) and max weight (arg=1) optional arguments, which change when algorithm terminates. The annealing rate multiplies the maximum acceptable displacement after each iteration, slowly increasing it when the value is something slightly above 1. This basically allows the algorithm to accept "close enough" answers if it's been trying for long enough. The max weight is used to judge the displacement rate as something between the maximum and the average. The returned value will be max offset * max weight + avg offset * (1 - max weight).

- Added auto size optional argument, which changes the disk-size variable to incorporate the number of disks, and the size and general intensity of the image. An entirely black image of size 100x100 will have the disk-size variable multiplied by the square root of 100*100/disk-count. So if there are 2500 disks, radius will be set to 2, meaning if all disks are placed evenly, they will all touch exactly at the edges. Less dark images will have smaller disks, based on the assumption that there will necessarily be more space between the disks.