CENTROIDAL VORONOI TESSELLATIONS

JARED BURNS

Abstract. The Voronoi diagram and centroidal Voronoi tessellation (CVT) are defined and their properties explored. The Lloyd and MacQueen algorithms for determining a CVT from an ordinary Voronoi diagram are defined. In order to compare the efficiency of the two, a few stopping parameters including the energy functional are examined. Then, the data analysis of the computation time of Lloyd and MacQueen's algorithms given the same initial Voronoi diagram are presented and the differences discussed. Finally, a new hybrid method using the Lloyd and MacQueen algorithms as a template is constructed and shown to be more efficient then either method alone.

1. Introduction

1.1. **A Brief History.** According to Okabe in his book *Spatial Tessellations* [2], some of the first uses of the Voronoi diagram were recorded as early as the 17th century by the well-know philosopher Descartes. In his works, Descartes used weighted Voronoi diagrams to explain how matter is distributed throughout the solar system. Figure 1, a diagram of certain points on the Euclidean plane separated by lines that are equidistant from the closest two such points, is an example of Decartes' work.

Although Georgy Fedoseevich Voronoy (1868-1908) was not the first to study this special kind of decomposition of a metric space, it bears his name in honor of the advancements he made in the theory. He not only unequivocally defined the Voronoi diagram, he also spent time studing the general m-dimensional case of such diagrams, whereas many of his peers only examined the 2 or 3 dimensional cases. In honor of another one of his peers who also made significant advances to Voronoi theory, Voronoi diagrams are sometimes also referred to as a Dirichlet tessellations.

Since its conception, the Voronoi diagram has been used by anthropologists, crystallographers, ecologists, economists, and many others to model and explain structures, cultures, and markets. One interesting early example of Voronoi modeling is John Snow's analysis of the London Cholera outbreak in 1854. Snow used a crude Voronoi diagram to show that the outbreak of the cholera was due to contaminated water originating from a single water pump. After the handle from that pump was removed, the cholera outbreak soon ceased [5].

1.2. **Outline.** The Voronoi tessellation is also used in a wide scope of modern applications. From computational geometry to resource allocation to digital image compression and

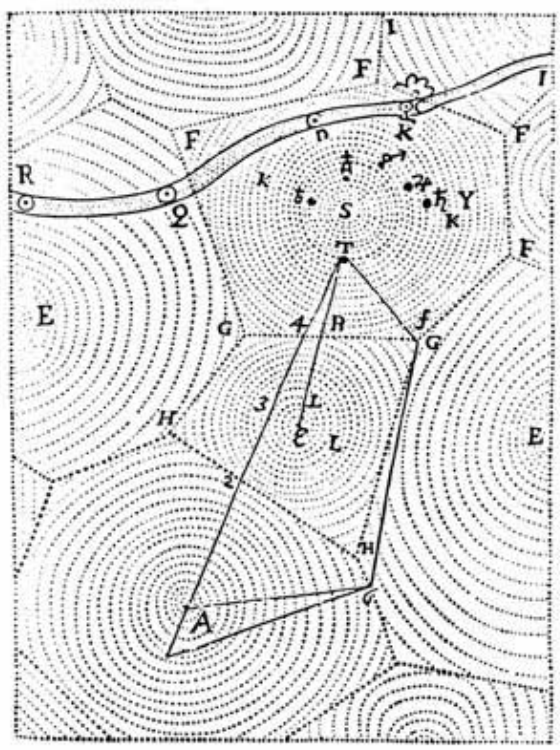

FIGURE 1. Descarte's Voronoi diagram [2](p. 7)

robot navigation, Voronoi diagrams are used to answer many questions. With the fairly recent advent of the computer age comes new possibilities in applying computationally intensive Voronoi algorithms to solve problems.

The main purpose of this paper is to explore the efficiency of different algorithms in determining a centroidal Voronoi tessellation given an initial Voronoi diagram. We are interested in deriving the CVT and some of the methods and mathematical necessities that come along with it. Thus, much of this paper is devoted to the basic concepts of the Voronoi diagram, the CVT, and computational methods necessary for the completion of the CVT algorithms.

We begin with probably the most basic and essential step in understanding the rest of this paper: the definition of the Voronoi diagram and the basic properties of such diagrams. From there, we look at the definition of the centroidal Voronoi tessellation and explore a few useful applications of the CVT. The next section of this paper is devoted to exploring the methods used to derive a CVT from a Voronoi diagram. In this section, we discuss the Lloyd and MacQueen methods in terms of their computational algorithms. After some final discussion on the nuances of the algorithms including stopping parameters and other methods necessary for the MacQueen and Lloyd method to work properly, we discuss some

specific data that was taken from the running of these algorithms. The first two sections contain computation time data for the two methods in a variety of conditions as well as a discussion on the development and effeciency of a new hybrid algorithm, formed from the strengths of the Lloyd and MacQueen algorithms.

2. Voronoi Diagrams

The purpose of the following section is to introduce the reader to Voronoi diagrams and a selection of their properties. If the reader is already familiar with Voronoi diagrams, he may wish to skip to the next section, which discusses the centroidal Voronoi tesellation (CVT).

2.1. **Definition of Voronoi Tessellations.**

In the 2-D case, a Voronoi diagram is a partition of the plane into n convex polytopes. Each partition contains one generator such that every point in the partition is closer to its own generator than any other generator. In general, the definition of the Voronoi diagram in n dimensional space is given as in Du's *Centroidal Voronoi Tessellations: Applications and Algorithms* [1]:

Definition 2.1 (Voronoi Tessellation)**.** Given a set of points $\{z_i\}_{i=1}^k$ belonging to the closed set $\overline{\Omega} \in \mathbf{R}^N$, the Voronoi region $\widehat{V}_i$ corresponding to the point z_i is defined by:

$$\widehat{V}_i = \{x \in \Omega \big| |x - z_i| < |x - z_j| \text{ for } j = 1, \ldots, k, j \neq i\}.$$

In addition, we include the definition of a tessellation.

Definition 2.2 (Tessellation)**.** Given an open set $\Omega \in \mathbf{R}^N$, the set $\{V_i\}_{i=1}^k$ is called a tessellation of Ω if $V_i \subset \overline{\Omega}$ for i=1, ...,k, $V_i \cap V_j = \emptyset$ for $i \neq j$, and $\cup_{i=1}^k \overline{V}_i = \overline{\Omega}$.

Here, Du chooses to define the Voronoi tessellations as a set composed of a collection of open sets. However, it is also possible for these tessellations to be defined as closed sets (as is done in Okabe's *Spatial Tessellations*). Some of the properties of Voronoi diagrams we discuss later may depend on a precise form of the definition. In such cases, a note will be made.

We are now going to introduce some useful vocabulary that we employ throughout the paper. See Figure 2 for an illustration of the following definitions:

- Voronoi generators - the set of points $\{z_i\}_{i=1}^k$ belonging to the closed set $\overline{\Omega} \in \mathbf{R}^N$ in a Voronoi diagram that are used to form the distinct Voronoi regions.
- A Voronoi Region - the convex area (in Euclidean space) that contains every point closest to a generator with respect to all the other generators.
- A Voronoi Edge is the line, half line, or line segment that corresponds to the points that connect exactly half way between two Voronoi Generators.

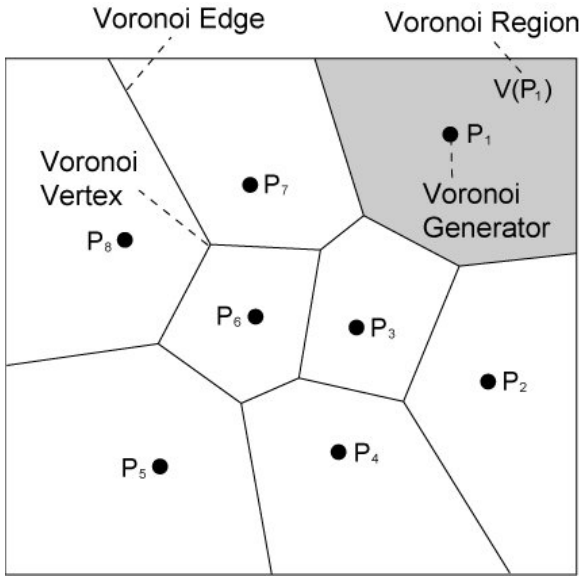

FIGURE 2. Graphical Definitions of Voronoi Pieces [6]

- A Voronoi vertex is the intersection of three (or more, depending on the restrictions of the given Voronoi Diagram) Voronoi Edges.

2.2. **Properties of Voronoi Diagrams.** Before we can go much further in dealing with Voronoi tessellations, it is important to understand the geometry of Voronoi diagrams.

Property 2.1. If the Voronoi polygon for the generator point p_i contains the point p, then p_i is the closest generator point from p. This is a direct result of the definition of a Voronoi tessellation.

Property 2.2. Given the set of generators $P = \{p_1, \ldots, p_n\} \in \mathbf{R}^N$, the Voronoi diagram of P is a unique tessellation of $\mathbf{R}^N$.

Property 2.3. (The Non-Cocircularity Assumption) Given the set of generators $P = p_1, \ldots, p_n \in \mathbf{R}^2$ (where $4 \leq n < \infty$), there can be no circle C, centered at a vertex, that contains more than 3 generator points on its circumference while having no other generator inside the area of the circle.

Property 2.4. For any vertex (q_i) in a Voronoi diagram, there exists a circle (C_i) that passes through at least three generator points, with no generator points in its interior. In the case of the Non-Coircularity Assumption, this simplifies to exactly three generator points on the edge of the circle. This circle happens to be the largest empty circle centered at the vertex q_i.

3. Centroidal Voronoi Diagrams

3.1. **Definition of Centroidal Voronoi Diagrams.** Up until this point, we have been exploring Voronoi diagrams and their properties. We now consider a diagram with a more restrictive definition: the centroidal Voronoi diagram. Recall the definition of the Voronoi tessellation, Definition 2.1, on which we now wish to place constraints in order to create the definition of the centroidal Voronoi tessellation.

The constraint for the centroidal Voronoi tessellation is simply that each Voronoi generator must be the mass centroid for its corresponding Voronoi region. Note Figure 5, for example. Although it may not be apparent at first glance, the second Voronoi graph has its generators in the exact place of the mass centroids of the Voronoi regions. Thus, it is a centroidal Voronoi diagram. Note that most of the time we will not be able to determine if the diagram is centroidal simply by inspection, but often CVT's look more organized than plain Voronoi tessellations.

Thus, our definition of a centroidal Voronoi diagram is the aforementioned definition of Voronoi tessellations with an extra constraint on the location of the generators.

Given the set of Voronoi regions $\{V_i\}_{i=1}^k$, the mass centroid c_i over a region with probability density $\rho(\mathbf{y})$ is defined as

$$c_i = \frac{\int_{V_i} \mathbf{y}\rho(\mathbf{y})\mathbf{dy}}{\int_{V_i} \rho(\mathbf{y})\mathbf{dy}}.$$

Note that the density function $\rho(\mathbf{y}) \geq 0$ and y is a vector in $\mathbf{R}^N$. Also, given k generators $\{z\}_{i=1}^k$, the following equality must hold in order for the Voronoi diagram to be considered a centroidal Voronoi tessellation:

$$z_i = c_i \text{ for } i = 1, \ldots, k.$$

It is interesting to note that centroidal Voronoi diagrams are not necessarily unique for a fixed density function and number of generators. That is, it is possible to have two or more different centroidal Voronoi tessellations for the same density function and number of generators. See Figures 3 and 4 for a simple two generator example of this non-uniqueness. Each has the same desnity function, domain, and each has two generators. Both diagrams in Figure 3 and 4 satisfy the conditions of a CVT.

An important characteristic of the centroidal Voronoi diagram is the density function, $\rho(\mathbf{y}) \in \mathbf{R}^N$. This function is responsible for "weighting" the CVT's generators. For instance, let us think for a moment of the density function and the CVT existing on the same contour map. A peak on the density function corresponds to an increased number of generators per unit area (or volume, etc.) at that same point in the CVT. Similarly, a valley corresponds to a decrease in the number of generators per unit area (or volume, etc).

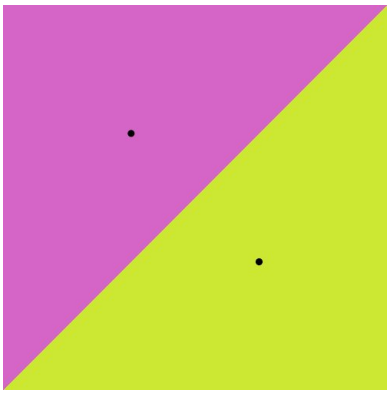

FIGURE 3. One version of the CVT

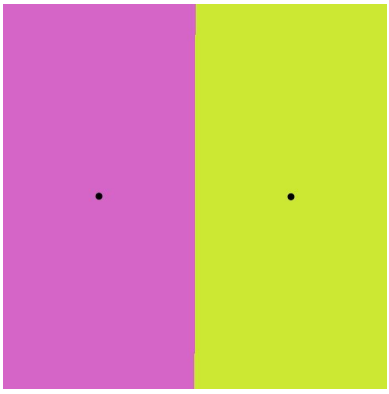

FIGURE 4. A second version of the CVT

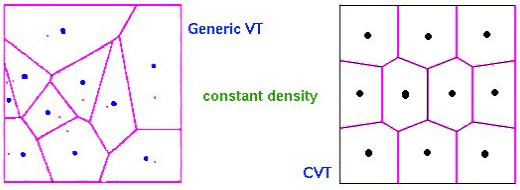

FIGURE 5. Regular Voronoi Diagram (left) and Centroidal Voronoi Diagram

3.2. **Application of Centroidal Tessellations: Data Compression.** Data compression is a technique of using fewer bits of memory to trasmit the same amount of information. The idea in most applications is to minimize the noticable loss of quality of the original information. In music, for example, there are many different methods of digitally storing music. The best, of course, are the ones that completely save all portions of the audio file. Naturally, these tend to be very large files. A data compression of a music file filters out most of the file that is essentially inaudable, thus using less memory with essentially the same sound as far as an average human can tell. We are now going to explore how

centroidal Voronoi tessellations can help us with something similar: image compression. The following example is in *Centroidal Voronoi Tessellations: Applications and Algorithms* [1].

Suppose a picture is made of 10^6 pixels. In computer graphics, each pixel is assigned a number that corresponds to its color. In this case, let that color be determined by a 24 bit number (so, each pixel requires 24 bits to "code" for its color). Thus, this uncompressed image, like an uncompressed music file, is very large. To be exact, it is 2.4×10^7 bits long. To reduce the amount of data, we could do one of two things: either reduce the number of pixels or reduce the amount of information needed to describe the colors. We are going to do the latter.

We decide to reduce the 24 bit color number to an 8 bit one. The question, then, is to decide how to approximate the old 24 bit color scheme with the new 8 bit one. An outline of a solution to this problem follows.

Let W be the set of admissable colors. Also, let $\rho(\mathbf{y})$ be the density function over the set W. We wish to choose k colors $\{z_i\}_{i=1}^k$ that will act as generators in W. If $\mathbf{y}$ is a color in the original picture and $\mathbf{y} \in W_i$, we replace $\mathbf{y}$ with z_i. Basically, we are creating a new set of colors. If the density function,$\rho(\mathbf{y})$, of the old color is found in a certain Voronoi region W_i, then that color is replaced with the new color (the Voronoi generator for that region, z_i).

In this algorithm, $\rho(\mathbf{y})$ denotes the number of times the combination of the basic colors $\mathbf{y}$ appears in the image. In this way, the 256 approximating colors (8 bits) can be chosen using the color density function $\rho(\mathbf{y})$. There are a number of ways to approximate $\{z_i\}_{i=1}^{256}$. All of the solutions we can generate for this problem are going to correspond to Voronoi spaces. But we are looking for a very close approximation of the original photo. To do this, we wish to find a Voronoi diagram that minimizes the distances between the old set of colors ($\mathbf{y_j}$) and the new approximated set (z_i). Thus, we create the energy functional

$$\varepsilon\big((z_i, V_i, i = 1, \ldots, k)\big) = \sum_{i=1}^{k} \sum_{\mathbf{y}_j \in V_i} \rho(\mathbf{y}) \mid \mathbf{y}_j - z_i \mid^2 . \tag{1}$$

If we minimize (1), we find the most approproate image compression we can using Voronoi theory (that is, we minimize the loss of data). In a proof in a later section, we will show that ε is minimized when the z_i's of the Voronoi regions are the mass centroids of their corresponding regions. In other words, the best compressed picture with k colors corresponds to the centroidal Voronoi tessellation over the set of admissible colors. If we wished to create the best compressed image we could using Voronoi diagrams, the solution would be as easy as computing the centroidal Voronoi diagram of the initial Voronoi diagram as described above.

4. The Lloyd and MacQueen Algorithms

There are numerous numerical methods for determining the centroidal Voronoi diagram given a specific probability density over the tessellation. The method we are first going to explore, MacQueen's method, is a probabilistic method that requires sets of random numbers according to the probability density function of the diagram. And the second method, Lloyd's method, is a deterministic algorithm that finds the centroid of a Voronoi region and averages the generator and the centroid to find a new generator.

4.1. **MacQueen's Algorithm.**

The following algorithm for Macqueen's Method is found in Ju, Du, and Gunburger's *Probabilistic Methods for Centroidal Voronoi Tessellations and Their Parallel Implementations* [1].

"Given a region Ω, and density function $\rho(x)$ defined for all $x \in \overline{\Omega}$ (the closed set), and a positive integer k,

(1) Choose an initial set of k points $\{z_i\}_{i=1}^k$ in Ω, e.g., by using a Monte Carlo method; set $j_i = 1$ for $i = 1, \ldots, k$;
(2) Determine a point y in Ω at random,e.g, by a Monte Carlo method, according to the probability density function $\rho(x)$;
(3) Find a z_i among $\{z_i\}_{i=1}^k$ that is the closest to y;
(4) Set
$$z_i \leftarrow \frac{j_i z_i + y}{j_{i+1}} \text{ and } j_i \leftarrow j_i + 1;$$
the new z_i, along with the unchanged $\{z_j\}, j \neq i$, form the new set of points $\{z_i\}_{i=1}^k$;
(5) If the new points meet some convergence criterion, terminate; otherwise, return to step 2" [1].

MacQueen's Method dictates that we use the density function of the region Ω to generate a random point $\mathbf{y}$ within Ω. Then, for whichever generator is closest, the average of that generator and the random point is taken. The new generator is the average. This whole process is then repeated as a weighted average with the new generators. After a set number of iterations or some stopping criteria is met, the new centroidal Voronoi diagram is formed from the values of the weighted averages.

The Monte Carlo method plays an important role in this algorithm. The Monte Carlo method is a general method that generates random inputs in the desired region and performs some computation on them based on where they fall in the region. In order to put MacQueen's algorithm to any practical use, we must first find a Monte Carlo method that satisfactorily chooses initial random numbers. For both parts (1) and (2) of MacQueen's algorithm, we choose to use the rejection method.

4.2. **Rejection Method.**

Random number generators, such as those used in programming languages C++ and Fortran, are generally of uniform density. Since we may deal with

non-uniform densities, we wish to find a Monte Carlo method capable of producing random numbers according to said non-uniform probability density function.

There are a number of ways to do this. One of the simpler yet computationally intensive ways is to use the following formula:

$$X = \frac{\int_a^x \rho(s)ds}{\int_a^b \rho(s)ds},$$

where we are generating numbers in the interval (a,b) with density function $\rho(\mathbf{s})$. Note that s in this case is a generalized coordinate as the density function should span the spatial dimensions of the tessellation. Evaluating two integrals for each and every random number can be quite taxing on computer memory, especially when the density function is complex. Since we are interested in employing this algorithm using computational means, we will now look at a different procedure: the rejection method described in [1].

The rejection method is a probabilistic procedure that takes in two random numbers generated from uniform density and produces either one number that satisfies the non-uniform density or nothing at all. Suppose our two randomly generated numbers are u and v. We want to test to see if one of these random numbers fits into our new non-uniform density function. Choosing both u and v to be bounded over $[0, 1]$, let $X = a + (b - a)v$. We can think of this relation as a kind of parameterization that stretches the random number $v \in [0, 1]$ to the random number $X \in [a, b]$. At this point, we have a random number over the interval we want, but we do not know if X fits our non-uniform density requirements. Using probabilistic means, the rejection method states that we set up the following inequality using the second random number:

$$u < \frac{\rho(X)}{(\max_{x\in[a,b]} \rho(x))}. \tag{2}$$

If inequality (2) holds true, then we let the random number $x = X$. Otherwise, we pick two more random numbers and begin again until the inequality does hold.

A pictorial example of this method is shown in Figure 6 and Figure 7. Note that the density function $\rho(x)$ is normalized so that its maximum value is equal to one. This method is completely dependent on the fact that the smaller the value of $\rho(x)$ (the left and right portions of Figure 7), the less likely it will be for a random variable u (chosen over [0,1]) to be in the interval $[0,\rho(x)]$. This direct correlation gives us the random generators over the non-uniform density we are seeking.

The careful reader may question the effeciency of the rejection method, because in its implementation, we may end up throwing out many iterations worth of data that do not satisfy inequality (2). However, in general, even with the "wasted" iterations, the rejection method is on average less computationally intensive than the integrating method.

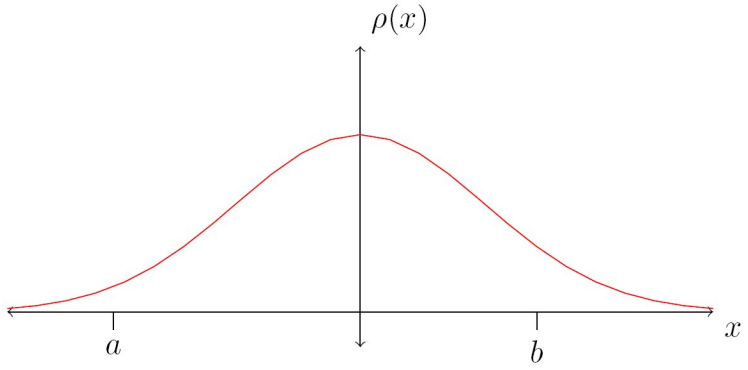

FIGURE 6. A Bell Density Curve

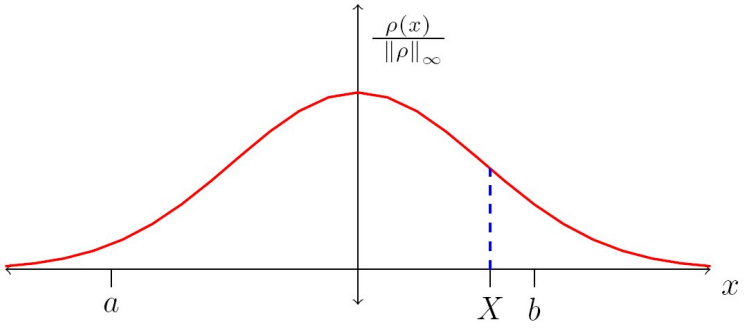

FIGURE 7. New, normalized density curve used to create tbe new set of random points

Since we are most interested in the two-dimensional cases, we now wish to generalize the rejection method for 2-D cases. The rejection method for two dimensional domains is very similar to that of the one dimensional case. We begin with a density function $\rho(x, y)$ (defined over our domain) and a rectangle $D = [a, b] \times [c, d]$ that encloses the domain in which we are interested (i.e. for our domain $\Lambda \subset \mathbf{R}^2$, $\overline{\Lambda} \subset D$). Once again, since the random points are picked over the interval $[0, 1]$, we have two formulas to "stretch" them over the interval of the enclosing rectangle of our domain: $X = a + (b - a)v$ and $Y = c + (d - c)u$. Once again, our randomly generated points are u and v.

We now know that the point (X, Y) is in the enclosing rectangle D, but we do not yet know if it is in our domain $\overline{\Lambda}$. We can check that now; so if $(X, Y) \notin \bar{\Lambda}$, we simply pick two more random variables u and v and begin again. Otherwise, exactly analogous to the

1-D case, we pick another random point over [0,1], say z. If

$$z < \frac{\rho(X,Y)}{\max_{(x,y)\in\overline{\Lambda}} \rho(x,y)},$$

we have found our random point: $(x,y) = (X,Y)$. Otherwise, we begin again from the beginning and choose two more random points u and v. Similarly, this method can be generalized for domains in $\mathbf{R}^3$ or even higher dimensions.

Now recall that MacQueen's algorithm requires a Monte Carlo method for two of its steps. We choose for both steps to use the method just described, the rejection method, which effectively chooses a random point in the tessellation given a known probability density over said tessellation. We will also use the rejection method, although to a lesser extent, in the next CVT method, Lloyd's algorithm.

4.3. **Lloyd's Algorithm.** Another method for finding the centroidal Voronoi diagrams for a given region Ω and density function $\rho(x,y)$ is Lloyd's Algorithm. In this algorithm, an initial Voronoi diagram is made with the desired number of generators over the region Ω. Then, the centroids of the Voronoi regions are computed and a new Voronoi diagram is created using the centroids as the generator points. This is repeated for a given number of iterations or until some stopping criterion on the new generators are met. The following is a more precise algorithm as stated in *Probabilistic Methods for Centroidal Voronoi Tessellations and Their Parallel Implementations* [1]:

"Given a region Ω, and density function $\rho(x)$ defined for all $x \in \overline{\Omega}$, and a positive integer k,

(1) Choose an initial set of k points $\{z_i\}_{i=1}^k$ in Ω, e.g., by using a Monte Carlo method;
(2) Construct the Voronoi sets $\{V_i\}_{i=1}^k$ associated with $\{z_i\}_{i=1}^k$;
(3) Determine the mass centroids of the Voronoi sets $\{V_i\}_{i=1}^k$; these centroids form the new set of points $\{z_i\}_{i=1}^k$;
(4) If the new points meet some convergence criterion, terminate; otherwise, return to step 2" [1].

In the implimentation of this algorithm, we use a discrete method for determining the centroids in step (3) based on the number of pixels used in the Voronoi diagram. We will now look at the specific set-up for the two dimensional case. Given n pixels $i = 1, \ldots, n$ with position $\mathbf{p}_i$ that belong to the Voronoi region V_j, the center off mass C_j for Voronoi region V_j is given by:

$$C_j = \frac{\sum_{i=1}^n \rho(\mathbf{x}_i)\mathbf{x}_i}{\sum_{i=1}^n \rho(\mathbf{x}_i)}.$$

In this way, we tessellate over the entire region of our Voronoi diagram and find the centers of mass of each Voronoi region.

5. Stopping Parameters

Since both the MacQueen and Lloyd methods are iterative, we are now going to discuss how and when to stop these algorithms. There are a number of ways to set the stopping parameters for a program that determines a CVT. One is simply to set a number of iterations. Another involves testing the distance between each Voronoi generator and its corresponding centroid. Both of these methods are easy parameters to implement, but for our purposes they do not give enough information about the CVT's being derived.

The stopping parameter we choose to implement for comparing the Lloyd and MacQueen methods involves the minimization of the energy functional. This section explores two different stopping parameters and offers a proof proof that the CVT is the lowest energy Voronoi diagram.

5.1. **The Distance Stopping Parameter.**

We outline the distance stopping parameter in the following algorithm:

(1) Find the distance $d_i = \mid z_i - c_i \mid$; so d_i is the distance between each generator $\{z_i\}_{i=1}^k$ and its corresponding mass centroid $\{c_i\}_{i=1}^k$;
(2) Set a tolerance ϵ;
(3) Run through iterations of Lloyd's algorithm until $\max_{1 \leq i \leq k} d_i < \epsilon$, in which case stop.

Using this stopping criteria, Figures 8, 9, and 10 were produced (using Lloyd's method) with tolerances of 1, 10, and 50 units respectively (units are arbitrary and related to image pixels). For comparison, Figure 11 was made by simply iterating (again with Lloyd's method) 100 times. Note that the Voronoi diagram in Figure 11 is very similar to that of Figure 8 even though the image in Figure 8 was only iterated 14 times. Lloyd's method produces less than linear convergence toward the CVT; thus, for many practical purposes, the Voronoi diagram in Figure 8 is as good a CVT as the Voronoi tessellation in Figure 11.

One of the goals of this paper is to compare the efficiency of the Lloyd and MacQueen algorithms in computing a CVT. The distance stopping parameter, however, cannot aid us as, given a set tolerance with MacQueen's method, the question soon arises as to when to stop iterating. In MacQueen's algorithm, we randomly choose one coordinate per iteration, and one generator gets changed. Examining the distance between the new generator and old generator tells us very little about the entire Voronoi diagram. It is possible because of the random nature of MacQueen's algorithm that one iteration be within the set tolerance while the next is much bigger than said tolerance. Thus, a minimum distance between generators of sequential iterations does not produce similar Voronoi diagrams using the Lloyd and MacQueen algorithms.

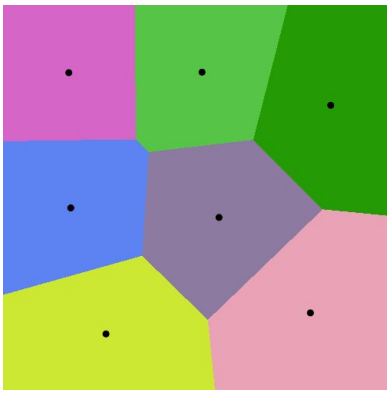

FIGURE 8. CVT produced with Lloyd's algorithm with a stopping parameter of 1 unit

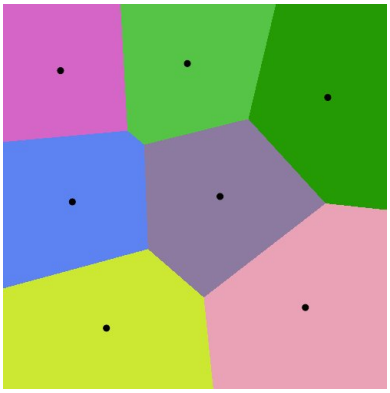

FIGURE 9. CVT produced with Lloyd's algorithm with a stopping parameter of 10 units

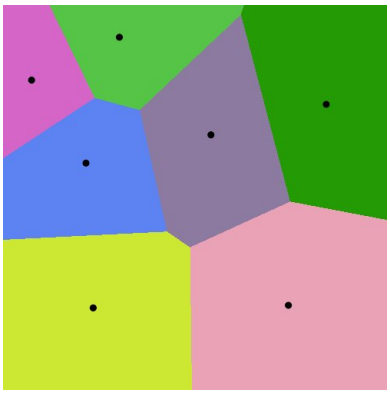

FIGURE 10. CVT produced with Lloyd's algorithm with a stopping parameter of 50 units

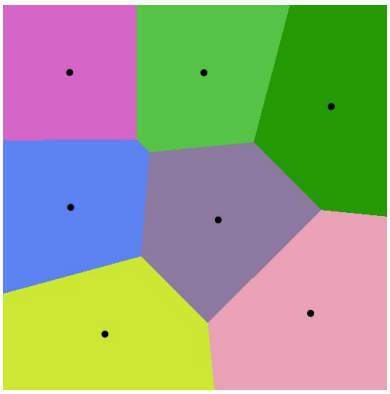

FIGURE 11. CVT produced with Lloyd's algorithm with 100 iterations

5.2. **Minimization of the Energy Functional as a Stopping Parameter.** We are interested in forming a stopping parameter that will stop both Lloyd's method and MacQueen's method at the same Voronoi graph. Once this is accomplished, it would then be a fairly trivial matter to compare the efficiency of the two methods. The stopping parameter we choose is dependent on the energy functional

$$\varepsilon\big((z_i, V_i, i = 1, \ldots, k)\big) = \sum_{i=1}^{k} \sum_{\mathbf{y}_j \in V_i} \rho(\mathbf{y}) \mid \mathbf{y}_j - \mathbf{z}_i \mid^2 .$$

Recall that $\rho(\mathbf{y})$ is the probability density function in the open set $\Omega \in \mathbf{R^N}$, where Ω is as defined in Definition 2.1. A centroidal Voronoi diagram is produced when the functional $\varepsilon\big((z_i, V_i, i = 1, \ldots, k)\big)$ is minimized. We will prove this momentarily, but for now we are simply interested in its usefulness as a stopping parameter. The energy functional behaves as a Cauchy sequence. That is,

$$\lim_{z_i, V_i \to z*_i, V*_i} \Delta\varepsilon\big((z_i, V_i, i = 1, \ldots, k)\big) = 0,$$

where $z*_i$,$V*_i$ are the centroidal Voronoi generators and the centroidal Voronoi regions, and $\Delta\varepsilon\big((z_i, V_i, i = 1, \ldots, k)\big)$ is the change in the energy functional between two different iterations of the sets $\{z_i\}_{i=1}^k$, $\{V_i\}_{i=1}^k$.

We can now set our stopping criteria to some value based on the Cauchy sequence. If we run both MacQueen and Lloyd's method until a chosen value for $\Delta\varepsilon\big((z_i, V_i, i = 1, \ldots, k)\big)$ is reached, we should have two Voronoi graphs that are essentially identical. At the very least, since the energy functional is a Cauchy sequence, both of their values for total energies are equally far away from the minimum energy. Using this value as a stopping criteria, we can then compare the two methods.

Note that there are a few potential problems with using the energy functional as a stopping parameter. The first crops up because the centroidal Voronoi diagram is not

unique, as discussed in section 3.1. If we desired to compare the efficiency of the MacQueen and Lloyd algorithms as they approached same CVT, the energy stopping parameter would require more specific criterion than we have so far given. For our purposes, however, we will not distinguish between one CVT and another.

Another possible problem occurs when either the Lloyd or MacQueen algorithm stops short of approaching the CVT at what we will call a "false bottom." A false bottom occurs when the Cauchy sequence has local minima. Since we are iterating Voronoi diagrams little by little, there is a chance our algorithm will happen upon a local minima in the energy functional before it reaches the absolute minimum. In such a case, the MacQueen or Lloyd algorithms could take thousands or even millions of iterations to push the diagram out of the local extrema. In testing the two methods, we will simply avoid any diagrams that seem to have such tendencies.

5.3. **Minimization of the Energy Functional.** We now wish to prove that the minimization of the energy function occurs at a centroidal Voronoi tessellation. The following proof follows closely to that given in Du, Faber, and Gunzenburger's *Centroidal Voronoi Tessellations: Applications and Algorithms.* [1]

Theorem 5.1. *Let $\rho(\mathbf{y})$ be the probability density function on the open set $\Omega \subseteq \mathbf{R^N}$ over which is found the sets of k points $\{\mathbf{z_i}\}_{i=1}^k$ and k regions $\{V_i\}_{i=1}^k$ that tessellate Ω. The functional*

$$F\big((z_i, V_i, i = 1, \ldots, k)\big) = \sum_{i=1}^{k} \int_{\mathbf{y} \in V_i} \rho(\mathbf{y}) \mid \mathbf{y} - \mathbf{z_i} \mid^2 d\mathbf{y} \tag{3}$$

is minimized when the set over Ω is a Voronoi tessellation with $\{\mathbf{z_j}\}_{j=1}^k$ as the centroids of the Voronoi regions $\{\widehat{V}_j\}_{j=1}^k$.

Proof. We begin by looking at a specific variation of F with respect to a single generator point, $\mathbf{z}_j$:

$$F(\mathbf{z}_j + \epsilon\mathbf{v}) - F(\mathbf{z}_j) = \int_{\mathbf{y} \in V_i} \rho(\mathbf{y}) \mid \mathbf{y} - \mathbf{z}_i - \epsilon\mathbf{v} \mid^2 \mid \mathbf{y} - \mathbf{z_i} \mid^2 d\mathbf{y}.$$

The only restriction on $\mathbf{v}$ is that $z_j + \epsilon\mathbf{v} \in \Omega$. If we then divide both sides by $\epsilon\mathbf{v}$ and take the limit as $\epsilon \to 0$, we have,

$$\lim_{\epsilon \to 0} \frac{F(\mathbf{z}_j + \epsilon\mathbf{v}) - F(\mathbf{z}_j)}{\epsilon\mathbf{v}} = \lim_{\epsilon \to 0} \int_{\mathbf{y} \in V_i} \frac{\rho(\mathbf{y}) \mid \mathbf{y} - \mathbf{z}_i - \epsilon\mathbf{v} \mid^2 \mid \mathbf{y} - \mathbf{z_i} \mid^2}{\epsilon\mathbf{v}} d\mathbf{y} \tag{4}$$

The left side of the equation is simply the derivative of the functional. We are interested in the point when the functional is minimized, which happens when $|F'(\mathbf{z}_j)|=0$. So, we

want to concentrate on the right hand side of equation 4. After doing a bit of vector algebra on the integrand of equation 4, we get

$$\lim_{\epsilon \to 0} \frac{|\int_{\mathbf{y} \in V_i} \rho(\mathbf{y}) 2\epsilon \mathbf{v} \cdot (-\mathbf{y} + \mathbf{z_j} + \epsilon \mathbf{v}) d\mathbf{y}|}{|\epsilon \mathbf{v}|} = 0. \tag{5}$$

Taking this limit gives us

$$\int_{\mathbf{y} \in V_i} \rho(\mathbf{y})(-\mathbf{y} + \mathbf{z_j}) d\mathbf{y} = -\int_{\mathbf{y} \in V_i} \rho(\mathbf{y}) d\mathbf{y} + \int_{\mathbf{y} \in V_i} \rho(\mathbf{y}) \mathbf{z_j} d\mathbf{y} = 0,$$

which leads to a familiar result:

$$\mathbf{z}_i = \frac{\int_{\mathbf{y} \in V_j} \mathbf{y} \rho(\mathbf{y}) d\mathbf{y}}{\int_{\mathbf{y} \in V_j} \rho(\mathbf{y}) d\mathbf{y}}.$$

Since this is the exact definition of a mass centroid, we see that the arbitrary $\mathbf{z}_j$ we have chosen is the centroid of our region V_i, and thus the points $\{\mathbf{z_j}\}_{j=1}^k$ are all centroids of their corresponding regions $\{V_i\}_{i=1}^k$. We now want to show that this arbitrary set of points and their corresponding regions are actually the Voronoi generators and regions. To do this, we are going to hold the points $\{\mathbf{z_i}\}$ fixed and compare the values of the functional given by the Voronoi tessellation $\{\widehat{V}\}_{j=1}^k$ and another tessellation $\{V\}_{i=1}^k$ as given in equation 3. The value of the functional for the Voronoi tessellation is

$$F\big((z_i, \widehat{V}_j, j = 1, \ldots, k)\big) = \sum_{j=1}^{k} \int_{\mathbf{y} \in \widehat{V}_j} \rho(\mathbf{y}) \mid \mathbf{y} - \mathbf{z}_j \mid^2 d\mathbf{y}. \tag{6}$$

We are now going compare the integrands of both Equations (3) and (6). We know from previous study of Voronoi tessellations that a point $\mathbf{y}$ belongs to a Voronoi region $\widehat{V}_m$ only if $|\mathbf{y} - \mathbf{z}_m|^2 < |\mathbf{y} - \mathbf{z}_{j \neq m}|^2$ for generators $\{\mathbf{z}_j\}_{j=1}^k$, where $1 \leq m \leq k$. Since the tessellation $\{V_i\}_{i=1}^k$ is not a Voronoi tessellation of our given Ω, it is necessary that the following inequality involving the integrands of Equations (3) and (6) hold:

$$\rho(\mathbf{y})|\mathbf{y} - \mathbf{z}_j|^2 < \rho(\mathbf{y})|\mathbf{y} - \mathbf{z}_i|^2.$$

Thus,

$$F\big((z_i, \widehat{V}_j, j = 1, \ldots, k)\big) < F\big((z_i, V_i, i = 1, \ldots, k)\big),$$

and F is minimized only when the tessellation $\{V_i\}_{i=1}^k$ is chosen to be the Voronoi tessellation $\{\widehat{V}_j\}_{j=1}^k$ with corresponding generator points $\{\mathbf{z}_j\}_{j=1}^k$.

□

6. Convergence and Efficiency

We are now going to examine how well the Lloyd and MacQueen algorithms derive centroial Voronoi Tessellations. We begin with a short discussion on the iterative convergence of the two methods by examining emperical data. We then compare the computation time of the two methods (for our purposes the shorter an algorithm takes, the more efficient it is) and discuss the differences between the two.

6.1. **Iterative Convergence of the MacQueen and Lloyd Algorithms.** Table 1 shows the average energy and maximum time for a one-dimensional MacQueen's method over the interval (-1,1) given a constant density function. In Table 1, as the number of iterations becomes greater, the energy converges at a smaller rate. As we can see, the first 20,000 iterations changes the value of the energy by about 17×10^{-5}, creating a significant digit 5×10^{-5}. The rest of the 3,180,000 iterations did very little more comparatively. They did not even create another significant digit to the right of the decimal point. Table 2 reveals similar truths about Lloyd's method. Basically, the same number of iterations produces fewer significant figures the closer to the CVT the graph becomes. Understandably, for applications that require a very high approximation of the CVT, these methods may not be adequate as they could take a very long time to compute such a CVT.

When constructing a method to compare the two methods, simply adding code to compute the energy of the preexisting CVT coded algorithm adds unwanted computation time. We are only interested in the computation time of the Lloyd or MacQueen algorithm itself. Fortunately, computing the energies of a CVT derived using Lloyd's method is a trivial matter. Since both the Lloyd algorithm and the energy functional require numerical iteration over Voronoi regions, we simply add a few lines of code adding negligible computation time that allow for the computation of the energy. This method adds a small but trivial amount of time to the computation time for Lloyd's method. Finding the energies of a CVT derived using MacQueen's algorithm is a lot more taxing on MacQueen computation time. For all comparisons between methods made in this section, only the MacQueen algorithm time (without computing the energy) is shown. In particular for the data shown here, we make two seperate programs for MacQueen's method: one that tests energies and one that does not. The one that tests CVT energies outputs the energies and the iterations corresponding to those energies. Once we know the iteration of a particular energy, we can then input the set number of iterations into the second program, which returns the computation time.

Note the discrepancy between the energy values in Table 1 and Table 2. This difference is a result of how the dimensions of the Voronoi diagram are measured. The diagrams whose values correspond to Table 1, for example, are measured over the interval [-1,1] as stated earlier. The diagrams whose values correspond to Table 2, however, are measured over the interval [0,500]. Recall that the energy functional as given by Equation (3) is a

1

TABLE 1. Maximum Run times and energies using MacQueen's method over tessellations with a constant probability density.

No. Iterations	Average Energy	Maximum time (s)
0	22.514E-5	0.00
200,000	5.888E-5	2.93
800,000	5.658E-5	11.59
3,200,000	5.475E-5	46.34

TABLE 2. Energies using Lloyd's method over tessellations with a constant probability density.

No. Iterations	Average Energy
1	8.89587E+08
10	5.49157E+08
25	5.29632E+08
1,000	5.28842E+08

function dependent on distance. Since the diagram corresponding to Table 2 has Voronoi regions covering much larger distances, it also has much bigger corresponding energies.

One thing that affects the convergence of MacQueen's method is the initial set of generators. That is, the success of MacQueen's method is dependent on the beginning placement of the Voronoi generators of its initial iteration. Thus, the better the Monte Carlo method is at distributing random points according to the probability density, the better MacQueen's method will converge. As Ju, Du, and Gunzburger said of the MacQueen method, "One observation that can be gleamed from a detailed examination of the results of the computational experiments (although not definitively from the data given in the tables) is that the energy of the final set of points is closely correlated to the energy of the corresponding initial set of points, i.e., in general, the smaller the initial energy, the smaller the final energy" [1].

6.2. **Run times of MacQueen and Lloyd Algorithms.** Table 3 and Table 4, when looked at simultaneously, compare the difference in computation time between Lloyd's method and MacQueen's method to reach a specific energy. In this case, since it is much easier and takes much less computation time to compute the energy for Lloyd's method, the energy is based on 10,000,000 interations of Lloyd's method. Since very little computation time is lost when finding the energy using Lloyd's method, the Lloyd energy was compared with the MacQueen until it was less than or equal the MacQueen value. Notice that since Lloyd's method computes a lot more in one iteration than MacQueen's, we often get slightly different energy values for Lloyd's method. Though they have different energies, we are guaranteed that the energy we are searching for in Lloyd's method is between the

TABLE 3. Maximum Run times and energies using MacQueen's method.

Density Function	Generators	No. Iterations	Ave. Energy	Max time (s)
1	10	10,000,000	1.069E9	7.248
	20	10,000,000	5.474E8	10.201
	100	10,000,000	1.085E8	36.334
$e^{-(y+x)^2}$	20	10,000,000	3.010E8	13.857
	100	10,000,000	6.215E7	40.033
$e^{-10*(y^2+x^2)}$	20	10,000,000	1.121E7	47.980
	100	10,000,000	2.657E6	73.364

TABLE 4. Maximum Run times and energies using Lloyd's method.

Density Function	Generators	Average Energy	Max time (s)
1	10	1.067E9	2.248
	20	5.474E8	5.470
	100	1.080E8	10.254
$e^{-(y+x)^2}$	20	3.010E8	7.035
	100	6.189E7	10.349
$e^{-10*(y^2+x^2)}$	20	1.119E7	3.747
	100	2.646E6	12.185

energy of the current and the previous iteration. Thus, the current iteration of Lloyd's method is the first iteration whose energy is less than or equal to the energy we want it to reach.

6.3. **Some More Run Times with Different Energy Stopping Parameters.** Tables 5 and 6 are almost identical to Tables 3 and 4 with one exception: the target energies are different. This time, the target energies are based on only 1,000,000 iterations of MacQueen's method. Notice how this time, unlike the first set of data, MacQueen's method often takes less computation time to reach the target. This data indicates that there is a point at which MacQueen's method becomes more efficient than Lloyd's method. In particular, our data indicates that MacQueen's method pushes the Voronoi diagram closer to a CVT initially but Lloyd's approaches the CVT more quickly given a diagram sufficiently close to the CVT.

Thus, we are unable to answer the question of which method is more efficient without qualifiers. As we just discussed, Lloyd's method is more efficient than MacQueen's given certain conditions, and visa versa. The conditions, however, naturally lead us to consider a third algorithm, one possibly more efficient than either of the first two. A hybrid of the two methods, combining them when they are most effiecient, is discussed in the next section.

TABLE 5. Maximum Run times and energies using MacQueen's Method.

Density Function	Generators	No. Iterations	Ave. Energy	Max time (s)
1	10	1,000,000	1.070E9	.724
	20	1,000,000	5.577E8	1.041
	100	1,000,000	1.099E8	3.617
$e^{-(y+x)^2}$	20	1,000,000	3.032E8	1.377
	100	1,000,000	6.296E7	3.993
$e^{-10*(y^2+x^2)}$	20	1,000,000	1.132E7	5.056
	100	1,000,000	2.741E6	7.618

TABLE 6. Maximum run times and energies using Lloyd's Method.

Density Function	Generators	Average Energy	Max time (s)
1	10	1.070E9	1.904
	20	5.575E8	3.445
	100	1.095E8	8.0887
$e^{-(y+x)^2}$	20	3.0E8	10.161
	100	6.261E7	8.714
$e^{-10*(y^2+x^2)}$	20	1.129E7	3.825
	100	2.721E6	10.321

7. HYBRID ALGORITHM

Using the idea that MacQueen's method better approaches a CVT initially and Lloyd's method does better when the diagram is closer to the CVT, we create an algorithm that is a hybrid of the two methods. The new algorithm is very simple:

(1) Given an initial set of generators, run MacQueen's algorithm for a set number of iterations and store the final set of generators.
(2) Run Lloyd's algorithm with the product of step 1 until a desired stopping parameter is met.

Note that we do not intend to find the most efficient hybrid method; such a problem lies outside the scope of this paper. We simply wish to know if we can combine the two to create a more efficient hybrid method than either the MacQueen or Lloyd algorithms.

7.1. **Hybrid Runtimes.** The following sets of data were taken using the hybrid method defined previously. Note that the particular hybrid methods used in Tables 7 and 8 actually run faster than just the MacQueen and Lloyd algorithms in most instances.

Table 7

Maximum Run times and energies using a combination of MacQueen and Lloyd's method (first 10,000 iterations of MacQueen, then Lloyd until it reaches the desired energy).

Density Function	Generators	Average Energy	Max time (s)
1	10	1.067E9	1.950
	20	5.602E8	1.000
	100	1.079E8	10.820
$e^{-(y+x)^2}$	20	3.008E8	9.650
	100	6.180E7	6.465
$e^{-10*(y^2+x^2)}$	20	1.119E7	1.798
	100	2.637E6	8.088

Table 8

Maximum Run times and energies using another combination of MacQueen and Lloyd's method (first 100,000 iterations of MacQueen, then Lloyd until it reaches the desired energy).

Density Function	Generators	Average Energy	Max time (s)
1	10	1.067E9	1.534
	20	5.575E8	.664
	100	1.077E8	10.222
$e^{-(y+x)^2}$	20	3.006E8	10.161
	100	6.172E7	4.575
$e^{-10*(y^2+x^2)}$	20	1.119E7	.848
	100	2.631E6	5.992

8. Conclusion

Based on emperical evidence, the effectiveness of both the Lloyd and MacQueen algorithms depend on how close the current Voronoi diagram is to the centroidal Voronoi diagram. The evidence shows that MacQueen's algorithm was more efficient with Voronoi diagrams further from the CVT (energy wise) while Lloyd's algorithm was more efficient at closer energies.

This conclusion led us to propose a new, (generally) more efficient method: a hybrid of the MacQueen and Lloyd algorithms. In our experiments, the hybrid method took as much as 88 percent off the computation time of the MacQueen or Lloyd algorithms alone. The hybrid method, however, is not always more efficient than either of the other two. If the Voronoi diagram starts too close to the CVT, it would obviously be more efficient just to run Lloyd's algorithms rather than a mixture of the two. Other possibilities for the reason why the hybrid method is sometimes slower would make for an interesting area of future research.

Although we demonstrated a more effecient algorithm, one thing that is still unclear is how effecient of an algorithm we actually have. For future research, we may be interested in finding the optimal switching point for the hybrid method. That is, we may want to

find the most efficient point to stop running the MacQueen algorithm and begin running Lloyd's. If we are in need of effeciency in general, we may also want to come up with a mathematical model that shows what method to use (hybrid or otherwise) based on the density function. Combining these two possibilities for research would allow us to always run our algorithms most effeciently.

References

[1] Du, Qiang, Vance Faber, and Max Gunzburger. *Centroidal Voronoi Tessellations: Applications and Algorithms.* Society for Industrial and Applied Mathematics. (1999): 637-676.

[2] Okabe, Michiko, Barry Boots, and Sung Nok Chiu. *Spatial Tessellations : Concepts and Applications of Voronoi Diagrams.* New York: John Wiley & Sons, Incorporated, 2000.

[3] Wilson, Robin. *Introduction to Graph Theory.* New York: Academic Press, Incorporated, 1972.

[4] Bollobas, Bela. *Graph Theory: An Introductory Course.* New York: Springer-Verlag New York Incorporated, 1985.

[5] Weisstein, Eric W. "Voronoi Diagram." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/VoronoiDiagram.html

[6] Hiroyasu, Miki, Shimosaka. "Noise figure and minor triangulation DORONE." miki-lab.doshisha.ac.jp