Centroidal Voronoi Tessellation Algorithms for Image Compression, Segmentation, and Multichannel Restoration

QIANG DU

Department of Mathematics, Penn State University, University Park, PA 16802, USA

qdu@math.psu.edu

MAX GUNZBURGER

School of Computational Science, Florida State University, Tallahassee, FL 32306-4120, USA

gunzburg@csit.fsu.edu

LILI JU*

Institute for Mathematics and its Applications, University of Minnesota, Minneapolis, MN 55455-0436, USA

ju@math.sc.edu

XIAOQIANG WANG†

Department of Mathematics, Penn State University, University Park, PA 16802, USA

wang@math.psu.edu

Published online: 9 January 2006

Abstract. Centroidal Voronoi tessellations (CVT's) are special Voronoi tessellations for which the generators of the tessellation are also the centers of mass (or means) of the Voronoi cells or clusters. CVT's have been found to be useful in many disparate and diverse settings. In this paper, CVT-based algorithms are developed for image compression, image segmenation, and multichannel image restoration applications. In the image processing context and in its simplest form, the CVT-based methodology reduces to the well-known k-means clustering technique. However, by viewing the latter within the CVT context, very useful generalizations and improvements can be easily made. Several such generalizations are exploited in this paper including the incorporation of cluster dependent weights, the incorporation of averaging techniques to treat noisy images, extensions to treat multichannel data, and combinations of the aforementioned. In each case, examples are provided to illustrate the efficiency, flexibility, and effectiveness of CVT-based image processing methodologies.

Keywords: centroidal Voronoi tessellations, image compression, image segmentation, edge detection, image restoration, k-means

*Current address: Department of Mathematics, University of South Carlifornia, Columbia, SC 29208, USA.
†Current address: Institute for Applied Mathematics and its Applications, University of Minnesota, Minneapolis, MN 55455-0436, USA; wang@ima.umn.edu.

1. Introduction

The development of image processing techniques has brought many technological advances in communications, entertainment, security, medicine, and manufacturing [1, 13, 18]. Image processing includes the enhancement, restoration, coding, and understanding of images and is aimed not only at providing quality picto-

rial information and transmission efficiency, but also in assisting in machine reasoning and recognition. Naturally, the enhancement and restoration of an image often depends on a good understanding of the image itself. One of the oldest and yet still popular tools employed to analyze and understand images is provided by clustering [8, 18]. Broadly speaking, clustering is a commonly used technique for the determination and extraction of desired features from large data sets and for the determination of similarities and dissimilarities between elements in the data set [14, 19]. In the context of image processing, data sets take the form of one or more images.

The concept of centroidal Voronoi tessellations (CVT's) has recently received much attention in numerous applications, including computer graphics and image processing [9–11, 16, 20, 21]. CVT's can be viewed as being a very natural clustering strategy. In its simplest from, CVT-based clustering coincides with the well-known k-means clustering scheme. Applied to image segmentation problems, CVT's also fall within the class of thresholding segmentation methods. Among such methods, CVT has the distinct feature that as part of the CVT methodology, the threshold values are determined through an optimization procedure. This feature of the CVT methodology accounts for much of its effectiveness in the segmentation and other image processing settings. Furthermore, CVT's provide a general mathematical framework that allows for a natural means for developing substantial extensions, improvements, and enhancements of k-means clustering and other existing clustering and thresholding methods.

In this paper, we apply CVT-based clustering to the problems of image compression, segmentation, and multichannel restoration. The central task in image compression is to faithfully represent a given image using less data than that which describes the given image. In image segmentation, the central tasks are to divide an image into segments or clusters such that the members of each segment have like attributes and to determine the boundaries or edges separating the segments. In multichannel image restoration, one is given several incomplete versions of an image and the task is to try to recover the complete image. Of course, there have been many other approaches proposed for image compression, segmentation, and multichannel restoration; see, e.g., [1–8, 13, 18, 24, 26, 27, 30]. Compared to some existing methods (especially partial differential equation and variationally based methods), CVT-based image compression, image segmentation, and multichannel image restoration methods are considerably less

expensive to apply and are more flexible. As a result, certain tasks are greatly facilitated; these include segmentation into more than two segments, handling multichannel data, and the differential weighting of data.

2.   Image Compression

One can regard an image as a function u defined on a domain Ω in Euclidean space. The most familiar images correspond to Ω being two-dimensional rectangular domains, but all ideas and algorithms discussed in this paper are also applicable in higher dimensions and to non-rectangular images. The values of u represent some attribute of the picture, e.g., color or brightness.[1] Here, we consider digital images so that, after appropriate coordinate scalings, the function u can be defined over a set Ω of integral points, i.e., in two dimensions, the points $(x, y) = (i, j)$, where (i, j) are integer pairs that range over the image domain. In fact, we specialize, without any essential loss of generality, to two-dimensional rectangular images so that the domain of $u(i, j)$ can be defined by the index set $D = \{(i, j) \mid i = 1, 2, \ldots, I, \ j = 1, 2, \ldots, J\}$ for positive integers I and J.

One way to compress an image, i.e., to approximate an image by representing it using less data, is to replace the possibly many colors or brightness levels in an image by a fewer number of colors or brightness levels, respectively, chosen from a smaller set. Centroidal Voronoi tessellations (CVT's) can be an effective means for compressing images in this sense.

Not surprisingly, once a smaller set of replacement colors is chosen, the colors in the original image are replaced by the closest color in the replacement set. Thus, if $\mathcal{Z} = \{z_\ell\}_{\ell=1}^L$ denotes the set of replacement colors, then the color $u(i, j)$ found at the physical location (i, j) is replaced by a color $z_k \in \mathcal{Z}$ such that

$$|u(i, j) - z_k| \leq |u(i, j) - z_\ell| \quad \text{for } \ell = 1, \ldots, L .$$

Note that this comparison compares color values and not physical distances.

The remaining question is how does one choose the set $\mathcal{Z} = \{z_\ell\}_{\ell=1}^L$ of replacement colors? Here, we use an optimization criteria which we now describe.

Centroidal Voronoi tessellations as "energy" minimizers. For a given image u and for any set of L distinct replacement colors $\mathcal{W} = \{w_\ell\}_{\ell=1}^L$, we define the *CVT-energy* by

$$E(\mathcal{W}) \equiv \sum_{(i,j) \in D} \min_{\ell=1\ldots,L} |u(i, j) - w_\ell|^2. \quad (1)$$

In the CVT image compression method, the replacement colors are chosen so that the CVT-energy is minimized, i.e., the set of CVT replacement colors $\mathcal{Z} = \{z_\ell\}_{\ell=1}^L$ satisfy

$$E(\mathcal{Z}) = \min_{\mathcal{W}=\{w_\ell\}_{\ell=1}^L} E(\mathcal{W}),$$

where the minimum is with respect to all possible sets of replacement colors having cardinality L.

Geometric definition of centroidal Voronoi tessellations. The terminology "centroidal Voronoi tessellations" arises from another, equivalent definition of CVT's. Given an image u, let[2]

$$\mathcal{U} = \{u(i, j)\}_{(i,j)\in D}$$

denote the set of (not necessarily distinct) color values in the original image. Then, for any set of replacement colors $\mathcal{W} = \{w_\ell\}_{\ell=1}^L$, let

$$V_k = \{u(i, j) \in \mathcal{U} : |u(i, j) - w_k| \leq |u(i, j) - w_\ell|$$
$$\text{for } \ell = 1, \dots, L\}, \quad k = 1, \dots, L. \quad (2)$$

Note that there are a total of I^*J values for $u(i, j)$ and that V_k denotes the subset of those values that are closest to w_k than to any of the other w_ℓ's. The subset of color values V_k is called the *Voronoi cell* or *Voronoi cluster* corresponding to w_k and the set of subsets $\mathcal{V} = \{V_\ell\}_{\ell=1}^L$ is called a *Voronoi tessellation* or *Voronoi clustering* [23] of the set $\mathcal{U}$ of color values appearing in the original image. The replacement colors w_ℓ, $\ell = 1, \dots, L$, are called the *Voronoi generators*. Note that we have that $V_\ell \cap V_k = \emptyset$ if $k \neq \ell$ and that $\mathcal{U} = \cup_{\ell=1}^L V_\ell$ so that the set of subsets $\mathcal{V}$ is a non-overlapping covering of $\mathcal{U}$.

On the other hand, for any non-overlapping covering of $\mathcal{U} = \{U_\ell\}_{\ell=1}^L$ into L subsets, we can define the means or centroids of each subset U_ℓ as the color $\overline{w}_\ell \in U_\ell$ that minimizes

$$\min_{w_\ell \in U_\ell} \sum_{u(i,j)\in U_\ell} |u(i, j) - w_\ell|^2.$$

For a general Voronoi tessellation of $\mathcal{U}$, we have that $w_\ell \neq \overline{w}_\ell$ for $\ell = 1, \dots, L$, i.e., the color values that generate the Voronoi clustering are not the means or centroids of the corresponding clusters. *Centroidal Voronoi tessellations* of $\mathcal{U}$ are special Voronoi clusterings $\{V_\ell\}_{\ell=1}^L$ whose generators $\{z_\ell\}_{\ell=1}^L$ satisfy

$$z_\ell = \overline{z}_\ell \quad \text{for } \ell = 1, \dots, L,$$

i.e., the color values that generate the Voronoi clustering are also the means or centroids of the associated clusters.

By rewriting the CVT-energy in the cluster terminology as

$$E(w_1, \dots, w_L; U_1, \dots, U_L)$$
$$= \sum_{\ell=1}^L \sum_{u(i,j)\in U_L} |u(i, j) - w_\ell|^2,$$

we see that the two definitions for CVT's are connected since it can be shown (see, e.g., [9]) that among all possible non-overlapping coverings $\{U_\ell\}_{\ell=1}^L$ and all possible replacement color sets $\{w_\ell\}_{\ell=1}^L$, minimizers of the energy are CVT's, i.e., special Voronoi clusterings for which the generators of the clustering are also the means of the clusters.

As defined so far, a CVT is merely a *k-means* clustering [18]; such clusterings are widely used in image segmentation, e.g., in the Digital Image Processing package of the Mathematica software suite. However, as will be demonstrated in this paper, viewing k-means clusterings as CVT's enables simple paths to useful generalizations.

The relation between the CVT-energy and the number of generators. Given an image data set, then, according to the theory of quantization [12], we have, for the generators $\mathcal{Z} = \{z_\ell\}_{\ell=1}^L$ of a CVT clustering, that the CVT-energy satisfies

$$E_L = E(\mathcal{Z}) \approx C L^{-2/d} \quad \text{as } L \to \infty, \quad (3)$$

where d denotes the dimension of the image and C is a constant whose value depends on the dimension d and the distribution of colors within the data set. It is easy to see from (3) that

$$E_L - E_{L+1} \approx \frac{2}{d} C L^{-(2/d)-1} \quad \text{as } L \to \infty,$$

i.e., the CVT-energy reduces more and more slowly as the number of generators increase.

For data sets that do not cluster well, i.e., if the set is uniformly distributed in some sense, (3) can hold even for small values of L. However, if the data set clusters well, then it has been observed that even for small values of L, the CVT-energy E_L decays much faster than that indicated by (3); this behavior is known as the *elbowing effect*. The elbowing effect can be used to rationally measure the "goodness" of a CVT clustering and to choose an effective value for L, the number of

CVT generators: if the given data naturally clusters, E_L reduces quickly as L increases until the data is well clustered; further increases in L will then effect much smaller decreases in E_L.

In practice, the detection of when the addition of more generators becomes unproductive is more transparent if one monitors the indicator $I_L = E_L L^{2/d}$ instead of the CVT-energy E_L. From (3), we have that

$$I_L = E_L L^{2/d} \to C \quad \text{as} \quad L \to \infty \qquad (4)$$

so that the once I_L is detected to remain nearly constant as L increases, one knows that the given data is well clustered.

2.1. Algorithms for Determining CVT's

The special nature of CVT's require their construction, i.e., given a positive integer L and a digital image $\mathcal{U} = \{u(i, j)\}_{(i,j) \in D}$, one has to determine a set of replacement colors $\{z_\ell\}_{\ell=1}^L$ that are simultaneously the generators of a Voronoi clustering of the colors in the image and the means of the associated clusters. The following algorithm can be used to construct CVT's; see, e.g., [15, 28, 29] for details.

Algorithm 1. Given a positive integer L and a digital image $\mathcal{U} = \{u(i, j)\}_{(i,j) \in D}$, choose any L color values $\{z_\ell\}_{\ell=1}^L$ and determine the associated Voronoi clustering $\{V_\ell\}_{\ell=1}^L$.

1. For each cluster V_ℓ, $\ell = 1, \ldots, L$, determine the cluster mean $\bar{z}_\ell$.
2. Determine the Voronoi clustering associated with the color values $\{\bar{z}_\ell\}_{\ell=1}^L$.
3. If the Voronoi clusterings corresponding to $\{\bar{z}_\ell\}_{\ell=1}^L$ and $\{z_\ell\}_{\ell=1}^L$ are the same, exit the loop; otherwise, set $z_\ell = \bar{z}_\ell$ for $\ell = 1, \ldots L$ and return to Step 1.

It is easy to see that Steps 1 and 2 will result in a strict decrease in the energy except if local minimizer is reached. Thus, due to compactness, it is guaranteed that the algorithm will converge. The initial set of color values should be chosen so that none of the associated Voronoi clusters are empty. Note that, since a digital image is a finite data set, the algorithm terminates in a finite number of steps. However, it is often the case that a very good approximation to the final CVT configuration can be obtained in substantially fewer steps. For this reason, at each iteration, one should calculate the energy of the current configuration and terminate

when that energy is within some prescribed tolerance of the energy of the previous configuration. In our calculations, we terminate the iteration when the ratio of the absolute value of the change in the energy to the value of the current energy is less than $0.01L$.

Algorithm 1 does not transfer elements of the given data set $\mathcal{U}$ from one cluster to another until the end of each iteration step. When a point is transferred to a new cluster, the cluster mean will move closer to that point, something that Algorithm 1 misses until the iteration step is complete, i.e., until all points in the given data set are treated. The following accelerated version of Algorithm 1 works with a transfer test that directly measures the change in energy that occurs when a point moves from one cluster to another and then immediately re-determines the cluster means before moving to the next point in the given data set.

Algorithm 2. Given a positive integer L and a digital image $\mathcal{U} = \{u(i, j)\}_{(i,j) \in D}$, choose any L color values $\{z_\ell\}_{\ell=1}^L$ and determine the associated Voronoi clustering $\{V_\ell\}_{\ell=1}^L$.

1. For every $u(i, j) \in \mathcal{U}$,

    (a) evaluate the CVT energy for all possible transfers of $u(i, j)$ from its current cluster V_ℓ to any of the other clusters V_k, $k = 1, \ldots, L$, $k \neq \ell$;
    (b) if moving $u(i, j)$ from its current cluster V_ℓ to the cluster V_m most reduces the CVT energy, then

      (i) transfer $u(i, j)$ from cluster V_ℓ to cluster V_m;
      (ii) replace the colors z_ℓ and z_m by the means of the newly modified clusters V_ℓ and V_m, respectively.

2. If no transfers occurred, exit; otherwise, go to Step 1.

Algorithms 1 and 2 both result in a k-means clustering (in color space) of the digital image. Both are guaranteed to reduce the energy after every iteration, and they converge to a (local) minimizer of the CVT-energy. Each iteration of Algorithm 2 is more costly than those of Algorithm 1 since for Algorithm 2 one must determine the effect that each potential transfer has on the energy. On the other hand, an iteration of Algorithm 2 leads to a larger decrease in the energy than does an iteration of Algorithm 1, and thus a smaller number of iterations is required for Algorithm 2. A hybrid approach is also possible in which one starts with Algorithm 1 and then switches to Algorithm

2. Presumably, after several iterations of Algorithm 1, only a very few of the more expensive iterations of Algorithm 2 are needed to obtain accurate results.

The costs of both Algorithms 1 and 2 may be reduced at the price of increased storage. One recognizes that points that are close to their current cluster means will likely not be transferred to another cluster. Thus, for Algorithm 1, we do not need to consider every point $u(i, j) \in \mathcal{U}$ when we determine a new clustering; we only need to consider those points whose distances to their old cluster mean is larger than the average distance of all points in its cluster to the cluster mean. Similarly, for Algorithm 2, we do not need to loop over every point $u(i, j) \in \mathcal{U}$; we only need to loop over points whose distances to their current cluster mean are larger than the average distance of all points in its cluster to the cluster mean. The latter requires one to determine and store the average distance between points in a cluster and the mean of the cluster. However, such a strategy will roughly halve the cost of each iteration of Algorithm 2 because it is very cheap to calculate the average distances by storing the summation of all the distances and the number of points of each cluster. It is transparent that a suitable data structure that takes into account such stored information could lead to a much more efficient implementation [21].

Another improvement to Algorithm 2 is possible by not comparing reductions in the CVT-energy for possible transfers to far away clusters [25]. Thus, in Step 1(a) of Algorithm 2, one would only consider clusters with means having a distance to the mean of the current cluster for the point $u(i, j)$ that is less than twice the distance from the point $u(i, j)$ itself to its own cluster mean. Implementing this strategy requires the computation and storing of the distances between cluster means. A similar strategy could be employed with Algorithm 1 and may be more effective in that case since, for Algorithm 2, one would have to do the additional computation of the distances between cluster means within the inner loop. This strategy is very useful in the case that the number of clusters L is very large since it effects a factor of $O(L^{-1})$ cost decrease.

We note that CVT-based image compression algorithms are sufficiently inexpensive so that one usually does not have to resort to the implementation of any cost-saving strategies. We also note that Algorithm 1 is easier to parallelize while Algorithm 2 is easier to generalize, e.g., to determine weighted CVT's; see Section 2.3.

2.2. *CVT-based Image Compression*

Figure 1 provides an example of the effectiveness of CVT-based image compression. The contouring effect seen in the middle image is not specific to CVT image compression; it results from the assignment step during which a slowly varying color distribution in the original image is mapped to a piecewise constant color distribution in the approximate image. The contouring effect may be removed by a simple dithering algorithm; see [9] for details.

CVT-based image compression is very effective for images having large, essentially uniform backgrounds. For example, consider the image on the left in Fig. 2 that was created by embedding the original image of Fig. 1 into a larger image with a nearly uniform background. The resulting image contains 256 shades of gray which can be represented by the integers 0 to 255. The background was created by randomly choosing, at each pixel, one of the 8 contiguous shades from 126 to 133. The gray shade density plot for the resulting image is given in Fig. 3 from which one clearly sees that the background shades dominate. The interesting part of the image, i.e., the upper left-hand corner, is relegated to shades with small densities. Clearly, one should be able to very well approximate the background with just one or two of the shades within the set of background shades; this is exactly what CVT-based image

Figure 1.   Left: the original image containing 256 shades of gray. Center: the CVT-approximate image containing 8 shades of gray. Right: the CVT-approximate image containing 8 shades of gray after dithering is applied to eliminate contouring.

Figure 2.   Left: the original image of Fig. 1 embedded in a nearly uniform background; there are 256 shades of gray used in the image. Right: the 8-shade CVT approximate image.
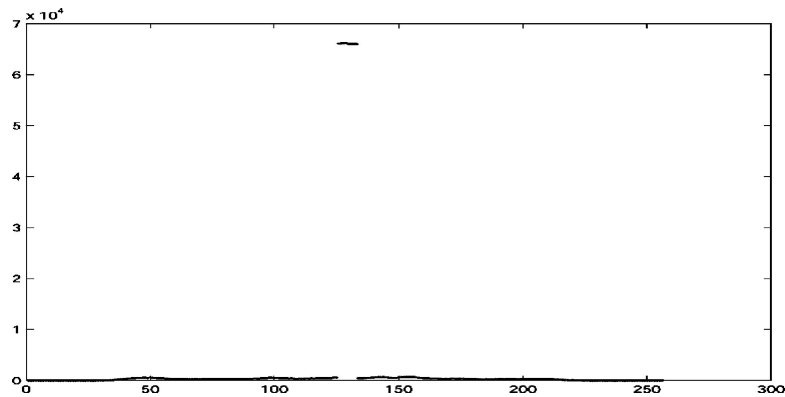

Figure 3.   The number of times each shade of gray appears in the image on the left of Fig. 2.

compression effects. The 8-shade CVT approximate image contains only two shades in the background (127 and 132) and 6 shades that are not in the background (49, 76, 101, 154, 177, 205) so that a sufficient number of shades are available to well approximate the interesting part of the picture; see the image on the right in Fig. 2. In contrast, a random choice (according to the density distribution of Fig. 3, suitably normalized) for the 8 replacement colors picks, in one realization, 7 shades within the set of background shades (126–129 and 131–133) and only 1 shade (50) that is not in the background. Further improvements that may be effected in compressing images with large, nearly uniform backgrounds are discussed in Section 2.3 where weighted CVT clustering is discussed.

Figure 4 provides another example of CVT-based image compression. As one expects, when one increases the number of replacement colors, the approximate image improves. One also sees that CVT-based image compression produces good approximate images with few approximate colors.

Examining a plot of the CVT-energy associated with the CVT-approximate images vs. the number of replacement colors L can be illuminating. For the case of Fig. 4, a plot of the normalized CVT-energy associated with the CVT-approximate images vs. the number of replacement colors L is provided in Fig. 5 (left). For comparison purposes, the hyperbolic curve $1/L$ is also plotted. We observe that the CVT-energy changes smoothly and also the elbowing effect discussed in Section 4, i.e., the fast reduction in the CVT-energy for small values of L (much faster than that for the hyperbolic curve) and the slower reduction for a larger number of colors. This behavior is even more apparent from Fig. 5 (right) which provides a plot of the indicator ratio I_L/I_1 (see (4)) vs. the number of replacement colors L; recall that for images that do not cluster naturally, this ratio should remain roughly unity. Instead, one observes from Fig. 5 (right) that as one increases the value of L, at first a large reduction in the value of I_L/I_1 occurs but that for $L > 5$ or so, that ratio remains roughly constant. One can then conclude that
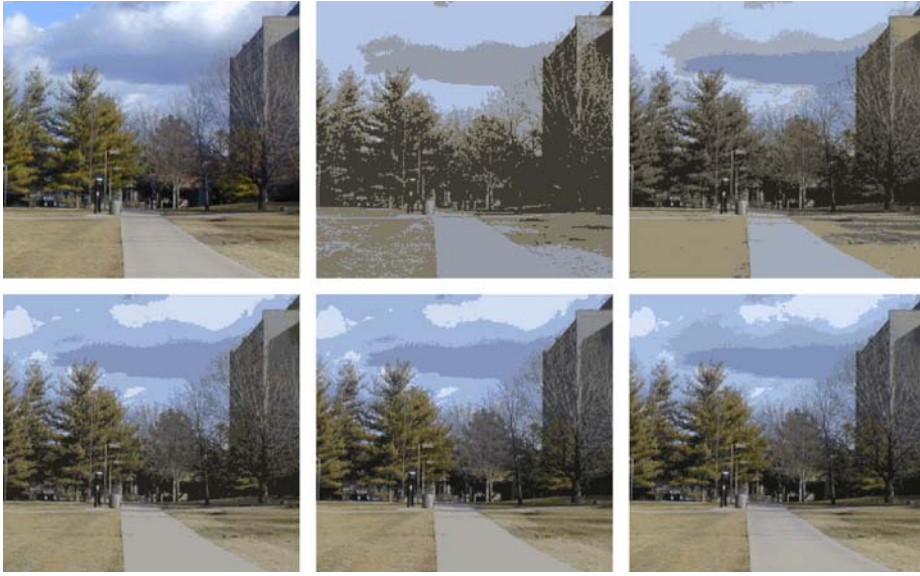
Figure 4.   The original image containing 1434 different colors is on the top left. The remaining 5 images from left to right and top to bottom) are CVT-approximate images containing 4, 8, 16, 32, and 64 colors, respectively.
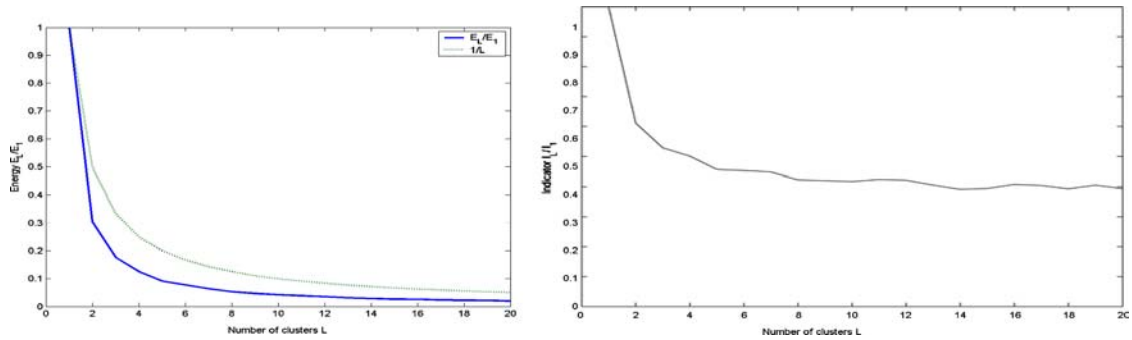

Figure 5.   Plots of normalized CVT-energy E_L/E_1 and $1/L$ (left) and the indicator ratio I_L/I_1 of the CVT-approximate images of Fig. 4 vs. the number of replacement colors L.

the data of the digital image of Fig. 4 clusters naturally into about 6 clusters.

2.3.   *Weighted CVT's*

The examples of Section 2.2 are all for the k-means clustering version of CVT-based image compression. In this section, we begin to show how the CVT approach can be generalized to treat more challenging images. We note that in some situations, it is desirable to change the definition of the CVT-energy so that the contributions from each of the clusters are weighted. This allows, for example, for a given data point to be included in a large cluster, somewhat analogous to the gravitational situation wherein a large planet has greater attraction. Such a change in the definition

of the CVT-energy gives rise to *weighted CVT's*. Applied to a digital image, weighted CVT's can let color generators focus on selected details of the image and not be overwhelmed by other colors.

To define weighted CVT's, we amend the definition of the CVT-energy to allow for cluster-dependent weights, i.e., we let

$$E(w_1, w_2, \ldots, w_L; U_1, \ldots, U_L)$$
$$= \sum_{\ell=1}^{L} \lambda_\ell \sum_{u(i,j) \in U_\ell} |u(i, j) - w_\ell|^2,$$

where λ_ℓ, $\ell = 1, \ldots, L$, are positive weighting factors. The case $\lambda_\ell = 1$ for all ℓ leads to the situation already discussed. In general, λ_ℓ is allowed to depend on factors such as the cardinality $|U_\ell|$ of the subset

U_ℓ, the within cluster variance, etc. [17], but not on the individual members within a cluster. The concept of center of mass is not affected by the introduction of weights. However, due to the presence of the weights $\{\lambda_\ell\}$, the minimization of the weighted CVT-energy leads to weighted Voronoi tessellations [23]. In the current context, we choose the weights λ_ℓ to be only a function of $|U_\ell|$. In particular, the choice $\lambda_\ell = 1/|U_\ell|$ has a statistical meaning: the weighted CVT-energy is then given by the summation of the average deviation of each cluster. Instead, one could select the power function $\lambda_\ell = |U_\ell|^\alpha$ for values other than 0 and −1. Different choices for the cluster weights result in different CVT clusterings. If $\alpha < 0$, larger clusters will tend to attract points from smaller ones; this may result in an interesting phenomenon: large clusters eating small ones. If $|\alpha|$ is too large, one may end up with only a single cluster, a situation that should be avoided. On the other hand, if $\alpha > 0$, smaller clusters will tend to attract points from larger ones and clusters will be prevented from becoming too large.

Before providing examples of the use of weighted CVT's for image compression, we give an algorithm (that is a direct generalization of Algorithm 2) for their construction.

Algorithm 3. Given a positive integer L and a digital image $\mathcal{U} = \{u(i, j)\}_{(i,j) \in D}$, choose any L color values $\{z_\ell\}_{\ell=1}^L$ and determine the associated Voronoi clustering $\{V_\ell\}_{\ell=1}^L$ and the associated weights $\{\lambda_\ell\}_{\ell=1}^L$, possibly dependent on $|V_\ell|$, the cardinality of the subset V_ℓ.

1. For every point $u(i, j) \in \mathcal{U}$,
   - (a) evaluate the weighted-CVT energy for all possible transfers of $u(i, j)$ from its current cluster V_ℓ to any of the other clusters $V_k, k = 1, \ldots, L$, $k \neq \ell$;
   - (b) if moving $u(i, j)$ from its current cluster V_ℓ to the cluster V_m most reduces the weighted-CVT energy, then
     - (i) transfer $u(i, j)$ from cluster V_ℓ to cluster V_m;
     - (ii) if the number of points in cluster V_ℓ is less than a preset value, delete that cluster and transfer all its elements to the cluster V_n which results in the smallest total weighted-CVT energy;
     - (iii) replace the colors z_ℓ, z_m, and, if necessary, z_n by the means of the newly modified clusters V_ℓ, V_m, and V_n, respectively.

2. If no transfers occurred, exit; otherwise, go to Step 1.

Note that in Step 1(b)-(ii), the total energy may increase for some choices of cluster weights. Since generally a cluster containing too few points may not be of great interest, we delete it whenever it occurs. By deletion, the number of clusters is nonincreasing. Furthermore, for a fixed number of clusters, the total energy is decreasing. Thus, Algorithm 3 is guaranteed to converge to a local minimizer of the weighted energy function.

Examples of weighted CVT image compression are given in Fig. 6 for three choices of the exponent α. For
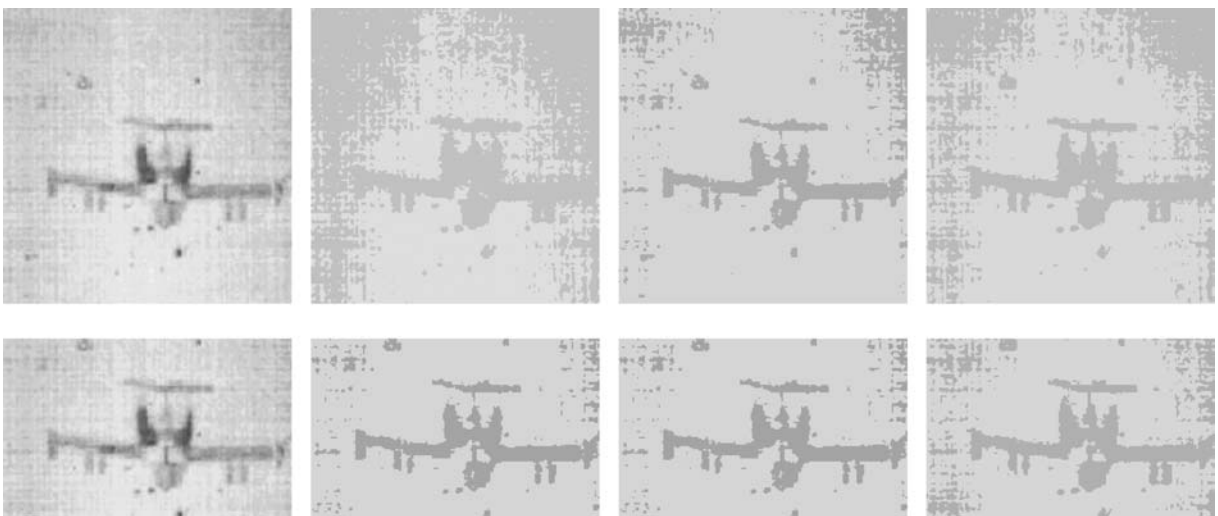

Figure 6.   The left column contains the original images, with the bottom one being part of the top one. The remaining three columns correspond to weighted CVT-based compressed images with weight function $\lambda_\ell = |U_\ell|^\alpha$ with, from left to right, $\alpha = 0$, 1, and 2.

these examples, we only consider clusterings into two clusters. Our objective is to select a value of α such that the resulting two-color weighted CVT image is better than the non-weighted CVT image determined with $\alpha = 0$. First, let us examine the first row of Fig. 6 for which there are clear differences in the results for different values of α. The left-most image is the original image. The second image is the non-weighted, two-color CVT image determined using $\alpha = 0$. The third and fourth images are weighted CVT images determined using $\alpha = 1$ and $\alpha = 2$, respectively. In the original image, the airplane is made up of mostly dark shades while the background contains light and intermediate shades. Varying α can affect how the intermediate shades are assigned to generators. In the non-weighted image, part of the background (corresponding mostly to the intermediate shades in the original image) is clustered together with the airplane to produce the large cluster. Using the weight $\alpha = 1$, more of the background is assigned to the background cluster, i.e., the smaller cluster for $\alpha = 0$ now attracts more points having an intermediate shade and becomes the larger cluster. Then, for $\alpha = 2$, the roles are reversed; the cluster corresponding to the darker shade of the airplane takes points away from the cluster corresponding to the lighter shade of the background. It seems clear that the image corresponding to $\alpha = 1$ is the best one since, relative to the other two compressed images, there is a much better separation of the airplane and background.

We now examine the second row of Fig. 6. The left-most original image is part of the image above it, with the remaining images corresponding to using the same values of α as the images in the first row. Now, in the original image, the background is much more uniform than that for the original image of the first row so that now there is less possibility of having background colors assigned to the generator representing the airplane. As a result, in this case, the weighted-CVT image for $\alpha = 1$ is almost the same as the non-weighted one; increasing the value of α from 0 to 1 does not appreciable change the clustering. However, for $\alpha = 2$, the clusters are sufficiently changed so that a visible difference in the CVT-weighted image can be observed.

Systematic means for choosing α or, for that matter, weight functions in general, have not yet been developed. However, the weighting flexibility allowed within the CVT methodology may prove to be very useful.

3.   Image Segmentation and Edge Detection

Image segmentation is the process of identifying the parts of an image that have a common attribute, e.g., that are roughly the same color or have the same brightness, and to also identify edges, i.e., the boundaries between the different segments of the image. The segmentation is done on the physical picture. CVT clustering in color space of an image can be easily and effectively used for image segmentation and edge detection.

Recall that the CVT clustering of an image is done in color space. Suppose that an image represented in color space by $u(i, j)$ for $(i, j) \in D$ is partitioned into the CVT clusters $\{V_\ell\}_{\ell=1}^{L}$. In physical space, this corresponds to the segmentation of the image into the L segments $\{D_\ell\}_{\ell=1}^{L}$, where

$$D_\ell = \{(i, j) : u(i, j) \in V_\ell\}.$$

Edges can be detected by seeing if neighboring points belong to a different cluster, i.e., $(i, j) \in D_\ell$ is an edge point if one of its neighboring points belongs to a different segment D_k, $k \neq \ell$. Equivalently, (i, j) is an edge point of the segment D_ℓ if $u(i^*, j^*) \notin V_\ell$, where (i^*, j^*) denotes one of the neighbors of (i, j). Thus, the key to CVT-based image segmentation and edge detection is the construction of the CVT of an image by the methods discussed in Section 2.

Figure 7 provides an example of CVT-based image segmentation and the corresponding edges of the segments for a grayscale image of some relatively simple geometric shapes. Figure 8 provides the CVT-based image segmentation and edge detection into 2, 3, and 4 segments for a bone tissue image.

We also computed CVT-based segmentations of the bone tissue image of Fig. 8 into higher numbers of segments. A plot of the normalized CVT-energy E_L/E_1 and the indicator ratio I_L/I_1 vs. the number of segments L are provided in Fig. 9. We see that for this particular image, the best segmentation occurs at around 4 segments. Using fewer segments results in unlike points being clustered together so that using a larger number of segments is useful for identifying more details in an image. However, having too many segments may result in the separation of features that one may want to have clustered together.

The indicator I_L can again be used to provide an automatic way for choosing the number of segments that balances these two competing objectives. For example, one could use the the following segmentation

Figure 7.   Left: the original image. Center: the CVT-based segmentation into two segments. Right: the edges detected via CVT-based image segmentation superimposed on the original image.
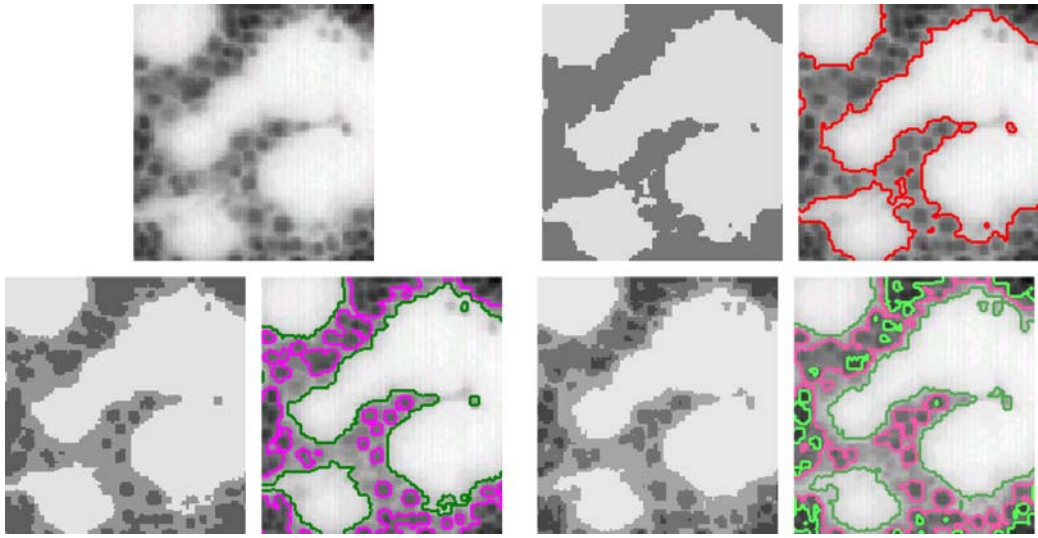

Figure 8.   On the top-left is an original grayscale bone tissue image. The left of each pair of images shows a CVT-based segmentation and the right of each pair shows the edges detected via CVT-based image segmentation superimposed on the original image. The three image pairs are for segmentations into 2 (top-right), 3 (bottom-left), and 4 (bottom-right) segments.

algorithm that includes a simple way to automatically choose the number of segments.

Algorithm 4. Given a digital image $\mathcal{U} = \{u(i, j)\}_{(i,j)\in D}$, choose small integers L_0 and dL. Set $L = L_0$ and compute the clustering of $\mathcal{U}$ into L segments using any CVT algorithm.

1. Set $\tilde{L} = L + dL$ and compute the CVT clustering of $\mathcal{U}$ into $\tilde{L}$ segments.
2. Compute $\epsilon = |I_{\tilde{L}} - I_L|/\mathcal{I}_{L_0}$.
3. If ϵ is within some prescribed tolerance, exit the loop; otherwise, set $L = \tilde{L}$ and return to step 1.

Of course, more sophisticated algorithms of this type can also be designed.

3.1. Smoothing

Images often contain substantial amounts of noise or have segments with highly irregular edges. To treat such cases, CVT-based image segmentation and edge detection can be combined with *averaging techniques*. From the beginnings of the development of image processing theory and algorithms, averaging has been a standard tool [22]. For digital images, an effective averaging formula is given by

$$\hat{u}(i, j) = \frac{\sum_{(\tilde{i},\tilde{j})\in B_r} u(i - \tilde{i}, j - \tilde{j})\, g(\tilde{i}, \tilde{j})}{\sum_{(\tilde{i},\tilde{j})\in B_r} g(\tilde{i}, \tilde{j})}, \quad (5)$$

where $g(\cdot, \cdot)$ is an averaging kernel and $B_r = B_r(i, j) = \{(\tilde{i}, \tilde{j}) \mid (i - \tilde{i})^2 + (j - \tilde{j})^2 < r^2\}$ is
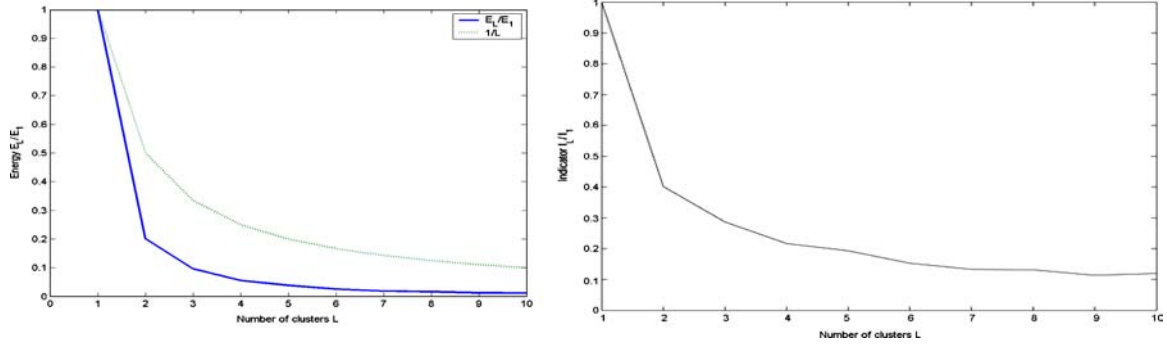
Figure 9. The normalized CVT-energy E_L/E_1 (left) and the indicator ratio I_L/I_1 (right) vs. the number of segments L for CVT-based segmentations of the bone tissue image of Fig. 8.
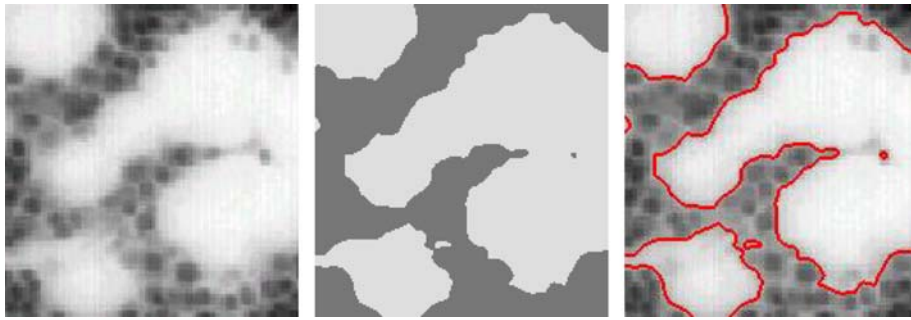

Figure 10. Left: an original grayscale bone tissue image. Middle: a CVT-based segmentation into two segments with subsequent averaging (with $\sigma = 2$ and $\beta = 0.05$). Right: the corresponding edges superimposed on the original image.

an averaging ball of radius r. The averaging formula (5) is an example of smoothing via discrete convolution with a kernel. The averaging radius r is typically chosen so that the integral of the averaging kernel $g(x, y)$ over the ball B_r is some prescribed large fraction of the integral of that kernel over the whole plane. For example, for the Gaussian kernel $g(x, y) = e^{-(x^2+y^2)/\sigma^2}$, if one wants the ratio of the integral of $g(x, y)$ over the ball of radius r and its integral over the whole plane to equal $(1 - \beta)$, $0 < \beta < 1$, then the averaging radius $r = -\sigma^2 \ln \beta$.

Although the use of Gaussian kernels is natural, there are many other, simpler, averaging kernels that are effective in practice. For our computational experiments that involve averaging, we use

$$g(i, j) = \frac{\sigma}{2\pi(i^2 + j^2 + \sigma^2)^{\frac{3}{2}}}, \qquad (6)$$

where, as for the Gaussian kernel, $\sigma > 0$ may be thought of as a smoothing parameter; in practice, the choice for the value of σ is based on the level of noise in the image; the larger the amount of noise, the larger should σ be. For the averaging kernel (6), one finds

that $r = \frac{\sigma}{\beta}\sqrt{1 - \beta^2}$. Typically, one chooses $\beta \leq 0.05$ so that a good approximation for the averaging radius is simply $r \approx \sigma/\beta$.

Averaging may also be combined with weighted CVT's. Figures 11 and 12 provide two examples.

CVT-based image segmentation and averaging can be combined in either order:

1. average the original image first, then apply a CVT segmentation technique to the averaged image;
2. apply a CVT segmentation technique to the original image, then average the original image, but use the replacement colors as determined by the CVT segmentation technique to determine the colors in the final image.

For the examples given here, we use the second approach since computational experiments indicate that it is more effective. Figure 10 shows the original bone tissue image of Fig. 8 along with the images and edges that result from a two-color CVT segmentation followed by averaging, i.e., the second approach listed

Figure 11. Left: an original grayscale image of an airplane. Middle: a weighted-CVT-based segmentation into two segments (with weights $\lambda_\ell = |U_\ell|$, i.e., $\alpha = 1$) and with subsequent averaging (with $\sigma = 1.5$ and $\beta = 0.05$). Right: the corresponding edges superimposed on the original image.

Figure 12. Left: an original grayscale image of a house. Middle: a weighted-CVT-based segmentation into two segments (with weights $\lambda_\ell = |U_\ell|$, i.e., $\alpha = 1$) and with subsequent averaging (with $\sigma = 1.5$ and $\beta = 0.05$). Right: the corresponding edges.

above. The CVT/averaging results should be compared with the top-right pair of images in Fig. 8 for which no averaging is done. Clearly, the segments and edges for the CVT/averaging images of Fig. 10 are considerably smoother than the ones in the corresponding images in Fig. 8.

3.2. *Additional Examples of CVT-Based Image Segmentation*

We provide some additional examples of CVT-based image segmentation that serve to illustrate its efficiency, flexibility, and effectiveness.

In Fig. 13, we consider a "fuzzy" ball image. We show its CVT segmentation into two segments with and without applying an averaging step (as described in Section 3) to the original image. Another example of the CVT-segmentation of a noisy image is provided in Fig. 14. Here the original image is a simulated noisy minefield image which is then averaged before it is segmented into two segments. A final example of combining image averaging with CVT-based image segmentation is given in Fig. 15. The original image is non-symmetrically averaged using

the kernel

$$g(i, j) = \begin{cases} 1 & \text{if } |i - j| < 2 \\ 0 & \text{otherwise.} \end{cases}$$

The resulting averaged image and the CVT segmentation into two segments of the averaged image are given in Fig. 15. Further improvement can be effected by averaging the segmented image (see Fig. 15).

The final example is the "Europe-by-night" image. In Fig. 16, we show weighted CVT-based segmentations of the original image into two, three, and four segments. Note that we segment the whole "Europe-by-night" image, and not just the "easier" part. In particular, note that even the two-segment image does a very good job of capturing Iceland, a part of the original image that is difficult to capture. The two-segment image shows that CVT-based image segmentation does a good job of differentiating between land and sea masses. By increasing the number of segments, CVT-based image segmentation can also separately capture the areas that are lit up. Note that a large positive exponent is used for the weight functions in the segmentation into two segments, but negative exponents are used for segmentation into three and four segments. The original image contains a
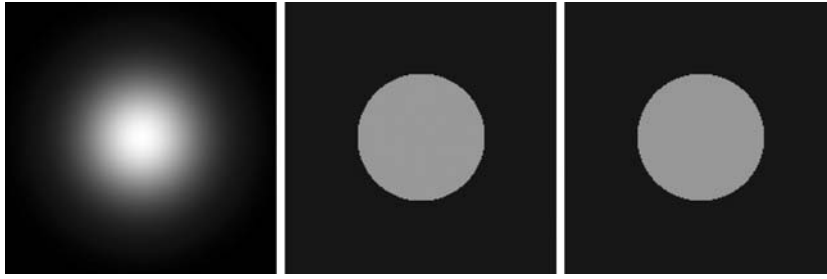
Figure 13.   Left: original image. Middle: CVT segmentation into two segments without averaging. Right: CVT segmentation into two segments after averaging the original image.
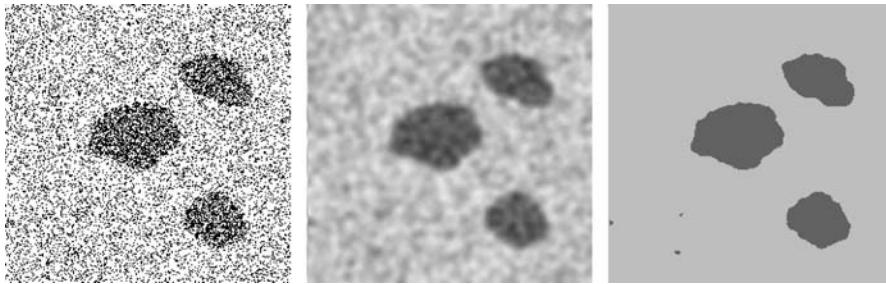

Figure 14.   Left: original image. Middle: Original image after averaging. Right: CVT segmentation into two segments after averaging the original image.

Figure 15.   Left: original image. Middle left: averaged image. Middle right: CVT segmentation into two segments of the averaged image. Right: average of the segmented image.

very dark sea mass, a slightly less dark land mass, and very bright lit areas. In the two-segment case, the large positive exponent is used to make sure that the dark land mass is clustered together with the lit up areas and not with the sea. On the other hand, when using three of more segments, we have the flexibility to differentiate between dark land masses and lit up areas, so that we do not want these to be clustered together; thus we switch to positive exponents in the weight functions. This example points out one of the strengths of CVT-based image segmentation: segmenting and image into more than two segments is hardly more difficult or costly than it is to segment it into just two segments.

4.   Reconstruction of Multichannel Images

We now consider the case for which one has in hand several versions of a picture, none of which contains all the information necessary to recover the complete image. The question is how the information contained in the different versions can be combined so as to recover the whole image? This is a natural task for CVT-based image processing.

Suppose we have M channels, i.e., M images, $\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_M$, where $\mathcal{U}_m = (u_m(i, j))$, and we wish to represent the complete image using L colors $\mathcal{Z} = \{z_1, z_2, \ldots, z_L\}$. We now define the Voronoi cluster
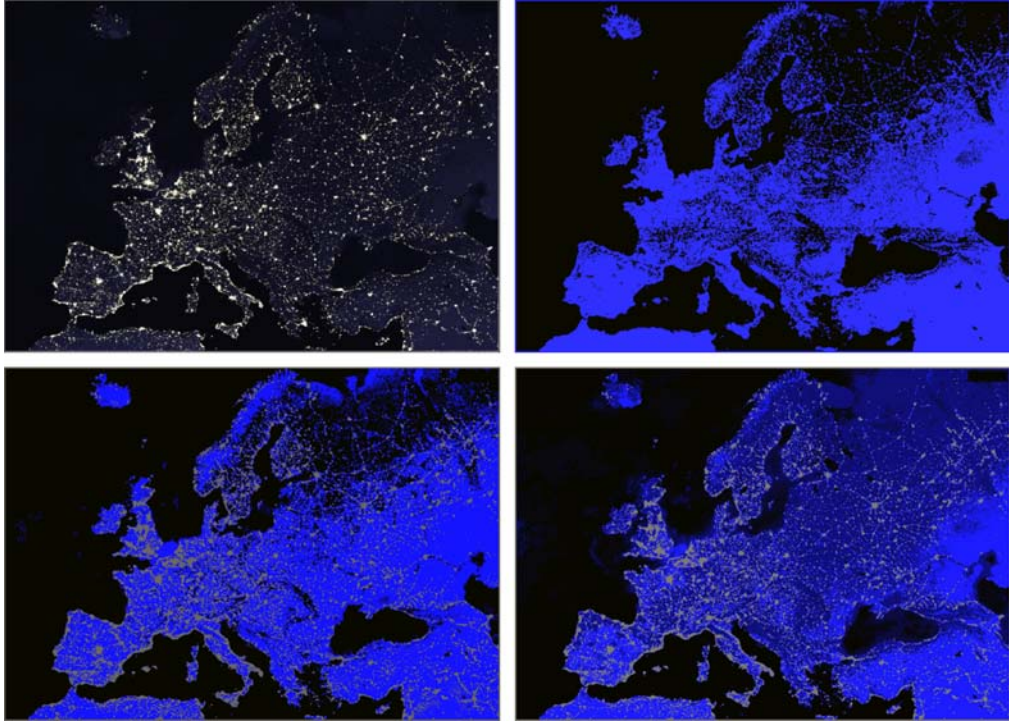
Figure 16. Top left: original "Europe-by-night" image. Top right: CVT segmentation into two segments with weights $\lambda_\ell = |U_\ell|^{10}$. Bottom left: CVT segmentation into three segments with weights $\lambda_\ell = |U_\ell|^{-3}$. Bottom right: CVT segmentation into four segments with weights $\lambda_\ell = |U_\ell|^{-2}$.

corresponding to the color z_ℓ by

$$V_\ell = \left\{ \{u_m(i, j)\}_{m=1}^M : \sum_{m=1}^M |u_m(i, j) - z_\ell|^2 \right.$$
$$\left. \leq \sum_{m=1}^M |u_m(i, j) - z_k|^2 \quad \text{for} \quad k = 1, \ldots, L \right\} \quad (7)$$

and we now define the mean or centroid of a Voronoi cluster to be the color that minimizes

$$\sum_{m=1}^M \sum_{u_m(i, j) \in V_\ell} |u_m(i, j) - z_\ell|^2. \quad (8)$$

As before, we define a CVT as a special Voronoi tessellation such that $z_\ell = \overline{z}_\ell$ for $k = 1, \ldots, L$.

The energy of the multiple channels is defined by

$$E(\mathcal{Z}) = \sum_{m=1}^M \sum_{\ell=1}^L \sum_{u_m(i, j) \in V_\ell} |u_m(i, j) - z_\ell|^2.$$

It can be shown that minimizers of the multichannel energy are CVT's of the multichannel data set $\{u_1, u_2, \ldots, u_M\}$ so that a multichannel CVT approximation of an image is found by minimizing that energy, i.e., by determining the set $\{z_\ell\}_{\ell=1}^L$ that minimizes the multichannel energy. We provide an algorithm for determining multichannel CVT's.

Algorithm 5. Given M versions $\mathcal{U}_m = \{u_m(i, j)\}_{(i, j) \in D}$, $m = 1, \ldots, M$, of a digital image and a positive integer L, choose any L color values $\{z_\ell\}_{\ell=1}^L$ and determine the associated Voronoi clustering $\{V_\ell\}_{\ell=1}^L$ defined by (7).

1. For each cluster V_ℓ, $\ell = 1, \ldots, L$, determine the cluster means $\{\overline{z}_\ell\}_{\ell=1}^L$ as defined in (8).
2. Determine the Voronoi clustering associated with the color values $\{\overline{z}_\ell\}_{\ell=1}^L$.
3. If the Voronoi clusterings corresponding to $\{\overline{z}_\ell\}_{\ell=1}^L$ and $\{z_\ell\}_{\ell=1}^L$ are the same, exit the loop; otherwise, set $z_\ell = \overline{z}_\ell$ for $\ell = 1, \ldots L$ and return to Step 1.

Figure 17 provides a word extraction example of multichannel CVT image processing.

Referring to Fig. 17, we see that sometimes, because the range of the colors in an image is relatively

Figure 17. The left three images are noisy and incomplete versions of the same image that is to be recovered. The right two images are the multichannel CVT-based approximate image using two replacement colors and its normalization.

Figure 18. The left and middle images are two noisy images of an airplane. The right image is a two-color multichannel CVT approximate image using a weight $v_1 = 5$ for the less noisy image on the left and a weight $v_2 = 1$ for the more noisy image in the middle.

limited, the resulting CVT generators of the two clusters may end up being "close" (in color space). In this case, we can redefine the range of the colors in the image so that, to the eye, the differences in the resulting generators is more obvious. A well-known method to do this, referred to as *normalization*, is to expand the range of colors in an image to the largest allowable. For example, suppose the possible shades of gray in an 8-bit grayscale image are labeled by the integers in $[0, 255]$. Suppose, however, that the shades of gray in a particular image only range over $[a, b] \subset [0, 255]$. If a and b are close, the eye may have difficulty differentiating between the two shades of gray appearing in a CVT (or any other kind of) approximate image. To ameliorate the situation, after we determine a CVT of the image, we redefine each shade of gray $u(i, j) \in [a, b]$ appearing in the image to $u(i, j) \leftarrow \text{Int}\big(255(u(i, j) - a)/(b - a)\big) \in [0, 255]$, where $\text{Int}(\cdot)$ is, say, the nearest integer function. In Fig. 17, the right-most image is a normalization of image to its left.

4.1. Weighted Multichannel CVT's

We can incorporate two different types of weights into the multichannel CVT-energy. We can assign a weight v_m, $m = 1, \ldots, M$, to each channel and, as before, we can also assign a weight λ_ℓ, $\ell = 1, \ldots, L$, to each cluster. If we only incorporate channel weights, we have that the multichannel CVT-energy is given by

$$E(\mathcal{Z}) \sum_{m=1}^{M} \sum_{\ell=1}^{L} = \sum_{u_m(i, j) \in V_\ell} v_m |u_m(i, j) - z_\ell|^2.$$

If we also use cluster dependent weights, the energy is then given by

$$E(\mathcal{Z}) \sum_{m=1}^{M} \sum_{\ell=1}^{L} = \sum_{u_m(i, j) \in V_\ell} v_m \lambda_\ell |u_m(i, j) - z_\ell|^2.$$

The channel weights v_ℓ can be chosen according to the quality of the image corresponding to each channel. The "clearer" the channel, e.g., the more complete or less noisy is a particular channel image, the larger the relative value of the corresponding weight. Selection of the cluster dependent weights λ_ℓ was discussed in Section 2.3. An algorithm for determining weighted, multichannel CVT's follows much the same lines as Algorithm 5.

Figure 18 provides a two-channel example for which one channel is less noisy than the other so that is assigned a higher channel weight in the CVT construction.

As for the single channel case, we can do averaging in order to smooth out images. For the multichannel case, averaging (with channel weighting) is based on discrete versions of the formula

$$\hat{u} = \frac{1}{MN} \sum_{m=1}^{M} \int_{\mathbb{R}^2} v_m u_m(x - \xi, y - \eta) g(\xi, \eta) \, d\xi d\eta,$$

where $N = \sum_{m=1}^{M} v_m$. We can also combine multichannel CVT with averaging in order to extract smoother edges. Figure 19 provides the results of combining averaging and multichannel CVT segmentation and edge detection for the example of Fig. 18.

Even after combining averaging with CVT segmentation, the resulting image, e.g., the image on the left

Figure 19.   Multichannel CVT-image segmentation (left) and edge detection (middle) with averaging for the two-channel noisy images of Fig. 18. The channel weights are the same as that for Fig. 18, $\beta = 0.05$, and $\sigma = 1.5$. The image on the right is the result of filtering the image on the left.

in Fig. 19, contains some noise. The following simple filtering procedure, which is applied after the clustered image is obtained, can make further inroads into the reduction of noise in a CVT-based image. First, for each $(i, j) \in D$, suppose the color at the point (i, j) in the physical image belongs to cluster V_ℓ in color space. Then, for a given filtering radius ρ, one determines how many points within a radius ρ of the point (i, j) belong to the same cluster V_ℓ; denote this number by $\mu_{i,j}$. (One can use percentages instead of raw counts.) If $\mu_{i,j}$ is less than a preset value, one views the color at the point (i, j) to be noise and then replaces the color at that point with the color corresponding to the cluster (in color space) that appears at the most points within the circle of radius ρ centered at (i, j). The image on the right in Fig. 19 is the result of applying this filtering procedure to the image on the left of that figure. We see that the filtered image has less noise.

5.   Concluding Remarks

The examples provided in the paper indicate that CVT-based methodologies are effective at several image processing tasks; the main goal of this paper was to present the CVT-based image compression, segmentation, and multichannel reconstruction methodologies and to provide some examples of their use. We have not included any explicit comparisons with other approaches, but CVT-based methodologies seem to be very effective and are certainly much less costly to apply than, e.g., partial differential equation or variationally based methodologies. Furthermore, some tasks, e.g., segmentation into more than two segments, are easily and efficiently accomplished by the CVT-based methodologies.

Of course, this paper should be viewed as only a starting point for studies in CVT-based image processing; however, we believe that the examples provided in this paper justify further study of CVT-based method-

ologies. Certainly, work remains to be done in order to conclude that they provide a better means for treating image compression, segmentation, and/or multichannel reconstruction tasks. For example, the systematic selection of weight functions should be studied in order to make weighted CVT methods more "automatic." Further studies of how averaging and CVT segmentation should be combined are also called for, as is a more detailed consideration of multichannel CVT methods. Explicit comparisons with other approaches would also be desirable. Finally, it is possible that CVT-based methodologies can be of use for other image processing tasks, e.g., inpainting, so that studies in these directions are also called for. All of the above suggestions for further study are currently being pursued.

Acknowledgments

Notes

1. The function u may be viewed as a scalar-valued function, e.g., in case it represents a brightness or grayscale level, or as a vector-valued function, e.g., in case it represents a color in RGB or CMY space.
2. A tie breaking rule should be invoked in case equality holds in (2), i.e., in case $|u(i, j) - w_k| = |u(i, j) - w_n|$ for some particular k and n; for example, in this case, the color $u(i, j)$ could be assigned to V_k if $k < n$ and to V_n if $k > n$.

References

1. K. Castleman, *Digital Image Processing*, Prentice Hall: Englewood Cliffs, 1996.
2. T. Chan, B. Sandberg, and L. Vese, "Active contours without edges for vector-valued images," *J. Visual Comm. Image Rep.*, Vol. 11, pp. 130–141, 1999.

3. T. Chan, J. Shen, and L. Vese, "Variational PDE models in image processing," *Notices AMS*, Vol. 50, pp. 14–26, 2003.

4. T. Chan and L. Vese, "An efficient variational multiphase motion for the Mumford-Shah segmentation model," *Proc. 34th Asilomar Conf. on Signals, Systems, and Computers*, Vol. 1, pp. 490–494, 2000.

5. T. Chan and L. Vese, "Active contours without edges," *IEEE Trans. Image Process*, Vol. 10, pp. 266–277, 2001.

6. T. Chan and L. Vese, "Active contour and segmentation models using geometric PDE's for medical imaging," in *Geometric Methods in Bio-Medical Image Processing*, R. Malladi (Ed.), Springer: Berlin, 2002, pp. 63–75.

7. L. Cohen, E. Bardinet, and N. Ayache, "Surface reconstruction using using active contour models," in *Proc. SPIE Conf. on Geometric Methods in Computer Vision*, SPIE, Bellingham, 1993.

8. E. Diday and J. Simon, "Clustering analysis," in *Digital Pattern Recognition*, K. Fu (Ed.), Springer: Berlin, 1980, pp. 47–94.

9. Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi tessellations: Applications and algorithms," *SIAM Rev.*, Vol. 41, pp. 637–676, 1999.

10. Q. Du, M. Gunzburger, and L. Ju, "Meshfree, probabilistic determination of point sets and support regions for meshless computing," *Comput. Meths. Appl. Mech. Engrg.*, Vol. 191, pp. 1349–1366, 2002.

11. Q. Du, M. Gunzburger, and L. Ju, "Constrained centroidal Voronoi tessellations on general surfaces," *SIAM J. Sci. Comput.*, Vol. 24, pp. 1488–1506, 2003.

12. A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic: Norwell, 1991.

13. R. Gonzalez and R. Woods, *Digital Image Processing*, Prentice Hall: Upper Saddle River, 2002.

14. J. Hartigan, *Clustering Algorithms,*, Wiley Interscience: New York, 1975.

15. J. Hartigan and M. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Appl. Stat.*, Vol. 28, pp. 100–108, 1979.

16. A. Hausner, "Simulating decorative mosaics," in *Proc. 28th Annual Conference on Computer Graphics and Interactive Techniques*, 2001, pp. 573–580.

17. M. Inaba, N. Katoh and H. Imai, "Applications of weighted Voronoi diagrams and randomization to variance-based *k*-clustering: (extended abstract)," in *Proc. Tenth Ann. Symp. on Computational Geometry*, 1994, pp. 332–339.

18. A. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall: Englewood Cliffs, 1989.

19. A. Jain and R. Dubes, *Aglorithms for Clustering Data*, Prentice Hall: Englewood Cliffs, 1988.

20. L. Ju, Q. Du, and M. Gunzburger, "Probabilistic methods for centroidal Voronoi tessellations and their parallel implementations," *Paral. Comput.*, Vol. 28, pp. 1477–1500, 2002.

21. T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE Trans. Pattern Anal. Machl. Intel.*, Vol. 24, pp. 881–892, 2002.

22. N. Nilsson, *Artificial Intelligence: A New Synthesis*, Morgan Kaufmann: San Francisco, 1998.

23. A. Okabe, B. Boots, K. Sugihara, and S. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, Wiley: Chichester, 2000.

24. S. Osher, "Level set method: Applications to imaging science," CAM Report 02-43, UCLA, 2002.

25. S. Phillips, "Acceleration of *k*-means and related clustering algorithms," *ALENEX*, pp. 166–177, 2002.

26. B. Sandberg, T. Chan, and L. Vese, "A level-set and Gabor-based active contour algorithm for segmeting textured images," CAM Report 02-39, UCLA, 2002.

27. G. Sapiro, *Geometric Partial Differential Equations and Image Processing*, Cambridge University, Cambridge, 2001.

28. D. Sparks, "Algorithm AS 58: Euclidean cluster analysis," *Appl. Stat.*, Vol. 22, pp. 126–130, 1973.

29. H. Späth, *Cluster Dissection and Analysis, Theory, FORTRAN Programs, Examples*, Prentice Hall: Upper Saddle River, 1985.

30. L. Vese and T. Chan, "A multiphase level set framework for image segmentation using the Mumford and Shah model," *Inter. J. Comp. Vision.*, Vol. 50, pp. 271–293, 2002.

Qiang Du is a Professor of Mathematics at the Pennsylvania State University. He received his Ph.D. from the Carnegie Mellon University in 1988. Since then, he has held academic positions at several institutions such as the University of Chicago and the Hong Kong University of Science and Technology. He has published over 100 papers on numerical algorithms and their various applications. His recent research works include studies of bio-membranes, complex fluids, quantized vortices, micro-structure evolution, image and data analysis, mesh generation and optimization, and approximations of partial differential equations.

Max Guzburger is the Frances Eppes Professor of Computational Science and Mathematics at Florida State University. He received his Ph.D. degree from New York University in 1969 and has held positions at the University of Tennessee, Carnegie Mellon University, Virginia Tech, and Iowa State University. He is the author of five books and over 225 papers. His research interest include computational methods for partial differential equations, control of complex systems, superconductivity, data mining, computational geometry, image processing, uncertainty quantification, and numerical analysis.

Lili Ju is an Assistant Professor of Mathematics at the University of South Carolina, Columbia. He received a B.S. degree in Mathematics from Wuhan University in China in 1995, a M.S. degree in Computational Mathematics from the Chinese Academy of Sciences in 1998, and a Ph.D. in Applied Mathematics from Iowa State University in 2002. From 2002 to 2004, he was an Industrial Postdoctoral Researcher at the Institute of Mathematics and Its Applications at the University of Minnesota. His research interests include numerical analysis, scientific computation, parallel computing, and medical image processing.

Xiaoqiang Wang is a graduate student in mathematics at the Pennsylvania State University, working under the supervision of Qiang Du. Starting in September 2005, he will be an Industrial Postdoctoral Researcher at the Institute of Mathematics and its Applications at the University of Minnesota. His research interests are in the fields of applied mathematics and scientific computation. His work involves numerical simulation and analysis, algorithms for image processing and data mining, parallel algorithms, and high-performance computing.