Two-stage color palettization for error diffusion

Niloy J. Mitra and Maya R. Gupta

[a]Dept. of Electrical Engineering, Stanford University, Stanford, CA, 94025 USA

ABSTRACT

Image-adaptive color palettization chooses a decreased number of colors to represent an image. Palettization is one way to decrease storage and memory requirements for low-end displays. Palettization is generally approached as a clustering problem, where one attempts to find the k palette colors that minimize the average distortion for all the colors in an image. This would be the optimal approach if the image was to be displayed with each pixel quantized to the closest palette color. However, to improve the image quality the palettization may be followed by error diffusion. In this work, we propose a two-stage palettization where the first stage finds some $m << k$ clusters, and the second stage chooses palette points that cover the spread of each of the M clusters. After error diffusion, this method leads to better image quality at less computational cost and with faster display speed than full k-means palettization.

Keywords: color palettization, color clustering, multi-level halftoning

1. INTRODUCTION

Color palettization is the problem of choosing a representative palette of a reduced number of colors to display an image. Palettization reduces the storage requirements as well as the necessary buffer for displaying images. These benefits are particularly important in low-cost displays, video-phones, video-conferencing, interactive games, etc. Palettization consists of designing a palette and mapping pixels into the palette. Palette design may be universal or image-adaptive. Universal palettes aim to represent all images (or all application-specific images) well. Image-adaptive palettes aim to represent a particular image as well as possible. After a palette is chosen, each pixel is mapped to a palette color. Pixels may be simply quantized to the closest palette color, or, to improve visual quality, the image may be halftoned using the colors in the palette.

In this paper we present a new two-stage algorithm for designing an image adaptive palette. Our algorithm takes into account the multi-level error diffusion halftoning that often follows palettization by choosing palette colors that capture the spread of a few color clusters. Furthermore, the efficiency of color palettization algorithms can be very important, and we show that the proposed technique is more computationally efficient than full k-means clustering while creating better images.

We consider some of the possible negative effects of palettization and discuss some of the prior work in this area. In section 2 we present our algorithm, discuss the theoretics of why it should result in good palettized images, and present some experiments and results.

1.1. Effects of palettization

The major issues in palettized image quality are false contouring, flat regions, color shifts, and vanishing colors. False contouring may appear if two spatially adjacent colors are consistently mapped to different palette colors when there is no real edge. Smooth regions and areas of slowly varying color should appear smooth but not 'flat'. Flat regions occur when an area of little color variance (like sky) is mapped almost solely to one palette color, and thus ends up appearing flat to the viewer.

Color shifts occur when an area has a sustained bias in the approximation error, and thus the perceived color value of the reproduced area is different enough from the original to cause viewer dismay. Color shifts are very noticeable in hue, due to human visual sensitivity to hue shifts.[1]

Further author information: (Send correspondence to Maya R. Gupta)
Maya R, Gupta: E-mail: guptama@stanford.edu, Telephone: 1 650.346.6324

Even palettes which optimally minimize the average distortion to the original image colors may miss colors that appear rarely in the original image, leading to a phenomenon called, 'vanishing colors'. Consequently, objects with infrequent color values may be very badly reproduced.

When the final stage is a halftoning stage, smooth regions may be represented by a variety of palette colors which average to the original color for the region. Due to the eye's sensitivity to contrast, a halftone will appear smoother if the palette colors are close in value, not far. For example, creating a mid-level gray by alternating light gray and dark gray appears smoother than reproducing mid-level gray with white and black pixels. In particular, the closer palette colors are in luminance values, the smoother we expect a halftone region to appear to the human eye, as the visual system is most sensitive to luminance contrasts at high spatial frequencies.[1]

1.2. Prior Art

A good review of the palettization literature before 1998 may be found in The Color Image Processing Handbook.[2]

A traditional approach to palettization is to cluster the colors in the image into k clusters, and then choose the cluster centroids as the k palette colors. Clustering pixels into k groups aims to minimize the total or average distance between pixels within clusters:

$$\min \sum_j \sum_i Dist(x_i, x_j) I_{ij}$$

Where $x \in R^d$ is a finite set of d-dimensional points, and the function I_{ij} is an indicator function equal to one if and only if x_i and x_j are members of the same cluster, and zero otherwise. The distance measure used in clustering is often a Euclidean or weighted Euclidean (Mahalanobis) distance.

The actual minimization of the average distance is generally NP-hard; clustering algorithms, such as k-means (also called vector quantization, VQ, LBG, and Lloyd's algorithm) and the expectation maximization algorithm (EM) aim to find the minimal average distance, but may achieve only sub-optimal (local minima) solutions. Furthermore, these iterative clustering algorithms take a large, and variable, amount of computation. Tree-structured VQ is not as optimal as full search VQ but improves the computation time for pixel mapping.

Another approach in the literature is heuristic clustering, both top-down (divisive) and bottom-up (agglomerative). These methods include the popularity algorithm, median-cut algorithm, binary-splitting algrithm, variance-minimization algorithm, and greedy tree growing technique. Gevautz's octree algorithm[2] is another heuristic clustering technique but aims to minimize the maximum quantization error instead of minimizing the average error.

For palettization systems where the output image is the original image with each pixel mapped to the closest palette color, the traditional palettization method of choosing a palette that minimizes the average distortion over all the pixels is the appropriate goal. However, if the output image is the original image halftoned with the palette, then the goal changes. After halftoning, a palette with minimal distortion will lead to false contouring and false colors. For example, imagine a smooth sky region - traditional clustering will lead to much of the region being well-approximated by one particular sky palette color, but that will leave that region looking very flat.

Two papers have appeared which aim to design color palettes for use with halftoning. In Kolpatzik and Bouman's work,[3] a universal palette is designed for use with error diffusion by using sequential scalar quantization(SSQ), an approximation to vector quantization. Since the image will be halftoned with the palette, the work includes heuristics to ensure that their palette colors extend to the boundaries of the devices gamut.

In Ozdemir and Akarun's paper,[4] they recognize the issue of designing a color palette to minimize mean-squared error and then dithering which judiciously increases mean-squared error. They propose three fuzzy joint quantization and dithering algorithms to combat this contradiction. In two of the algorithms they introduce dither errr into the design of the palette. In the third algorithm they minimize an objective function with the goal of moving cluster centers out towards the convex hull of the color space to obtain a palette more suitable for dithering.

2. THE ALGORITHM

The proposed palettization algorithm consists of two stages:

Stage 1) For a K color palette, cluster the image colors into $M << K$ clusters. Index the clusters $\{C_1, C_2, ..., C_M\}$.

Stage 2) For each cluster C_m, assign it a number of palette colors P_m, based on the cluster size and/or the color variance within the cluster. For each cluster C_m, choose the P_m palette colors to cover the spread of the original image colors in that cluster.

Post-Palettization) Error diffuse the image using the palette of colors.

Thus the basic idea is to use fewer clusters but have palette colors capture the full spread of each cluster. In the next couple sections we consider the details.

2.1. Stage 1

The clustering in Stage 1 could be done by a variety of clustering algorithms. We explored using k-means (vector quantization) and EM clustering, but found that the EM algorithm, with its extra degrees of freedom and computation due to the modelling of clusters as gaussians, did not produce significantly better results.

The k-means clustering[5] algorithm is a clever use of Lloyd's optimality conditions.[6] Begin with an initial set of cluster centers (e.g. chosen randomly from a uniform distribution over the space) and then assign the each data point to the closest centroid. Then each cluster's center is redefined as the average of the points falling in that cluster. Then the points are possibly re-assigned to the closest new centroid. This two-step iterative process is continued until an equilibrium is arrived at, possibly a local optimum. K-means may lead to different clusterings depending on the choice of the initial centroids or distance metric.

We used a Euclidean distance for our clustering in the YCbCr space. The human visual system is more sensitive to high frequency luminance changes than high frequency chrominance changes,[1] and thus halftones with lower luminance variance of palette colors tend to look smoother. We hypothesized that doing the first stage of clustering only in the luminance domain (cluster on the Y component only) would lead to sub-optimal clusters but possibly halftones with less luminance variance.

2.2. Stage 2

For the second stage, we considered various ways of assigning the number of palette colors P_m to each cluster C_m, including equal division, division based on cluster size, and division based on cluster variance. If a cluster contains a large variance of colors, then we expect it will need a lot of palette colors to cover its spread. On the other hand, the background of an image scene may have small color variance relative to other clusters, but contain half the image pixels, a key that that cluster is an important cluster to represent well. The more palette colors assigned to a cluster the smoother those colors will appear after halftoning.

Given a cluster of colors C_m and a number of palette colors P_m assigned to that cluster, we choose palette colors that represent the spread of the cluster. This palette design scheme is *non-iterative* and hence very fast. Unlike most of the palette quantization techniques, the aim here is to locally maximize the distance between palette colors within a color cluster, subject to the rate constraint of the color palette size.

The color palette is initiated with the centroid of the color cluster. For each subsequent palette color selection, the goal is to pick from the cluster's image colors the one that gives the maximum nearest neighbor with respect to the current palette:

For each cluster C_m containing N_m image colors $x_n \in C_m$ and assigned P_m palette colors, let the chosen palette colors be $\{p_1, p_2, ...p_{P_m}\}$ such that:

$$p_1 = \frac{1}{N_m} \sum_{x_n \in C_m} x_n$$

for $i = 2 : P_m$,

$p_i = x_n$ where x_n solves the following max-min problem:

$$\max_{x_n \in C_m} \quad \min_{p_y \in \{p_1, p_2, \ldots p_{i-1}\}} (x_n - p_y)^2$$

The process is repeated until all P_m palette colors have been chosen for every cluster C_m. This process can be performed in parallel for each of the M clusters. Note, we do not check whether a chosen palette color is close to a chosen palette color from another cluster. This may lead to a similar colors being stored in multiple palettes. This has been done to restrict the computational bound. However, this restriction can easily be lifted and better pixel-mapping may be attained at the cost of increased time requirements.

2.3. Error Diffusion

Error diffusion is a halftoning technique that can be viewed as a two-dimensional version of sigma-delta modulation. It was first proposed by Floyd and Steinberg in 1975.[7] Error diffusion quantizes each pixel, calculates the error of the quantization, and spatially passes a filtered version of this error to the neighboring pixels that have not yet been quantized. Many improvements have been attempted to the original algorithm, including different spatial filters for diffusing the error and adaptive thresholds for sharper edges.[8] However, most of these improvments have been oriented towards printed outputs with very few palette colors (generally 2-6 for printing).

3. COMPUTATIONAL COMPLEXITY

Let us assume that the image has a total of N pixels and we would like to create a palette of total P palette colors. Let the first level clustering lead to M clusters, $C_1, C_2, ...C_M$. Further, the m^{th}cluster C_m contains N_m image colors and a palette color budget of P_m. Then,

$$\sum_{m=1}^{M} N_m = N$$

$$\sum_{m=1}^{M} P_i = P$$

A complexity analysis of the *palette allocation* step can be done as follows:

For the m^{th} cluster the complexity is given by:

$$t(C_m) = O(1) + \sum_{t=2}^{P_m} [O(1) + (N_m - t)(t-1)] = O(N_m P_m)$$

Thus total complexity is given by:

$$T(N, P) = \sum_{m=1}^{M} O(N_m P_m)$$

Assume that each cluster contains an equal number of image colors and is assigned equal palette colors, then

$$N_m = \frac{N}{M} \quad \forall \ m$$

$$P_m = \frac{P}{M} \quad \forall \ m$$

then the palette design complexity (PDT) is:

$$T_{PDT}(N,P) = \sum_{m=1}^{M} O(N_m P_m) = O\left(\frac{NP}{M}\right)$$

and for the palette allocation time (PAT):

$$T_{PAT}(N) = \sum_{m=1}^{M} O(N_m) = N$$

Thus the total complexity is

$$T_{total}(N,P,M) = T_{PDT}(N,P) + T_{PAT}(N) + T_{clust}(N,M)$$

where $T_{clust}(N,M)$ is the clustering complexity from Stage 1, and depends on what clustering algorithm is used.

4. EXPERIMENTS

We compare our algorithm to full k-means for 24 bit color images reduced to 8 bit palettes. We used three images, *Hector*, which has a large critical face area, the *Flowers* image which contains a wide assortment of color regions, and the *Girl* image, which contains subtle wallpaper patterns, difficult stuffed animal texture, and skin and clothing features on the small girl.

We experimented with a number of different choices for each step of the algorithm. We found consistently good results with the following choices:

To select a 256 color palette:

Step 1) Use k-means to cluster the image colors into 10 clusters.

Step 2) Assign each cluster 20 palette colors.

Step 3) Distribute the remaining 56 colors amongst the ten clusters proportional to the product of the cluster size and variance.

Step 4) For each cluster, select the assigned number of palette colors using the method specified in the section on Stage 2.

We processed the two-stage algorithm and the control experiments in YCbCr space. We present results using the YCbCr space for the first step, and another set of results using only the Y component for the first-stage clustering. For all tests, after selecting the palette, the original image was error diffused with a Floyd-Steinberg filter using the selected palette.

5. RESULTS AND DISCUSSION

We show the full images and zoomed highlights in the next few pages. For each image, the top left image is the original, the top right image is the two-stage palettization with both stages performed in the YCbCr colorspace. The bottom left image is the full 256 cluster k-means result, and at the bottom right is the two-stage with the clustering (first stage) performed only on the Y component.

A soft copy of the paper may be downloaded from *www.stanford.edu/~guptama/researchpapers.html* and the full color images viewed on a monitor. In general, the results are significantly visually better at 75 dpi for the two-stage palettization than for the full clustering. We present quantitative results and discuss the visual differences.

Mean-squared error is not an appropriate metric for halftoned images because the whole point of halftoning is to *not* quantize each pixel to its nearest value, and thus not to minimize the mean-squared error. Instead, for an objective metric we use the sCIELab metric proposed by Zhang and Wandell,[9] which has been shown to correlate well with human perception of halftone image quality.[10] The sCIELab metric is a spatial extension of the CIELab Δ E metric. We use parameters for a standard CRT under standard office viewing conditions. We present results in Table 1.

	Full 256 k-means	Two-Stage	Two-Stage, Y-only first stage
Hector	3.63	1.06	1.17
Girl	2.36	1.09	1.26
Flowers	3.22	1.40	1.63

Table 1: Table of sCIELab mean error

5.1. A palette prepared for halftoning

Note that by trying to minimize the optimal clustering criteria, as k-means attempts to do, the palette is not optimized for halftoning. The desired result of $k - means$ is that most colors are very close to a palette color, but then in regions of smooth gradation all the image colors map to the same palette color and with little error, causing the region to look flat. As we move through the smooth region finally the colors change enough to map to a new palette color, creating a false contour look. Since the image colors are so close to palette colors there are only very small errors, which do not add up enough to put the error diffusion into effect, and thus there is less halftone alternation of colors that visually is perceived as smooth gradation. Furthermore, since the next palette color belongs to an entirely different cluster of colors, it may be quite far away, and thus appear in contrast to the surrounding flat color region.

The two-stage algorithm is optimized for error diffusion or dither because the palette colors have been chosen around a few clusters of colors. Thus most image colors have a small error, and it only takes a small error adding up or a small gradation to reach another close palette color from the same cluster. Thus the contrast of palette colors is usually small, creating a smooth appearance.

These differences yielded by the two-stage algorithm are clear in the results. In the close-up of Hector's face, the full-clustered image has false contours where the skin should appear as a smooth gradient, and the gray patches around the eye appear in contrast at 75 dpi. The two-stage images appear much smoother. In the girl's image we focus in on the texture of the blue rhinocerous. At 75 dpi, the full clustering result shows disturbing false contouring, whereas the two-stage results show smooth halftoned gradations.

5.2. Luminance only first stage

Clustering the colors of the first stage in YCbCr space led to only marginal improvement over doing the first stage clustering in only the Y, or luminance, dimension. The palette color alternations in the Y-only experiments often appear smoother as the colors being alternated have lower luminance variance (and the eye is more sensitive to luminance variance than chrominance variance), but the Y-only results may appear to have a little false contouring due to the sub-optimal clustering in Stage 1.

5.3. Hue shifts

Shifts in the hue of image objects can be visually disturbing. We found no percievable hue shifts with the two-stage algorithm. The k-means did present shifts in hue. The close-up of the flower image is of the colored reflection on the flower vase. K-means did not happen to pick up the right color in its palette, and the hue is shifted. The two-stage algorithms choose far fewer clusters in the first stage then normal palettization algorithms and thus are even more prone to missing entire color clusters. Yet, since the palette colors are chosen not just as the center of clusters but also as image colors that form farthest-neighbors within a cluster, image colors that are not at the center of a cluster are not forgotten. Thus the risk of a hue shift is minimized due to the greater color variance represented and the greater ease of forming halftone color alternations within a region to recreate a correct hue.

5.4. Checkerboarding and the error diffusion filter

Most error diffusion filters are optimized for use with very small palettes and printed output. Due to the selection of our palette colors, many image colors fall midway between two or more palette colors. That makes the halftone susceptible to mid-level density error diffusion artifacts such as checkerboarding. In fact, at very low resolutions checkerboarding is quite noticeable in the two-stage palettized images, but at 75 dpi (monitor resolution) the checkerboarding is not noticeable.

We used the Floyd-Steinberg error diffusion filter for all the results, but experiments with the Judice-Jarvis-Ninke filter, which shows less checkerboarding in print, also show fewer checker effects with the palettized images. However, the Judice-Jarvis-Ninke filter created more larger scale halftoning artifacts, which may be noticeable at 75 dpi.

6. CONCLUSIONS AND FUTURE WORK

The two-stage palettization algorithm we present is an efficient and effective way to create a palette for error diffusion palette systems with few negative palettization effects. We have discussed and shown that full clustering which aims to minimize the average distortion will not create as good a final halftoned palettized image.

Current directions include finding an effective but faster first stage clustering, determining an optimal number of clusters in the first stage, and investigating a more robust way to determine palette colors that represent the spread of each cluster in the second stage. We hope to present future results with synthetic images that highlight the issues in palettization.

6.1. Acknowledgments

The authors would like to thank Robert M. Gray for helpful comments and suggestions. Niloy J. Mitra acknowledges the generous support of the Stanford Graduate Fellowship program.

REFERENCES

1. B. Wandell, *Foundations of Vision*, Sinauer Associates, 1995.
2. S. Sangwine and R. Horne, *The Colour Image Processing Handbook*, Chapman and Hall, Great Britain, 1998.
3. B. Kolpatzik and C. Bouman, "Optimized universal color palette design for error diffusion," *Journal of Electronic Imaging* **4**(2), pp. 131–143, 1995.
4. D. Ozdemir and L. Akarun, "Fuzzy algorithms for combined quantization and dithering," *IEEE Trans. on Image Processing* **10**(6), pp. 923–931, 2001.
5. A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice-Hall, New Jersey, 1988.
6. A. Gersho and R. M. Gray, *Vector Quantization and Signal Processing*, Kluwer Academic Publishers, Massachusetts, 1992.
7. R. Floyd and L. Steinberg, "An adaptive algorithm for spatial gray scale," *Proc. of the Int. Symp. on Digital tech.* **36**, 1975.
8. H. Kang, *Digital Color Halftoning*, SPIE/IEEE Press, 1999.
9. B. Wandell, *www.stanford.edu/∼wandell*. This web page has links to papers and code for sCIELab metric.
10. X. Zhang, D. Silverstein, J. Farrell, and B. Wandell, "Color image quality metric s-cielab and its application on halftone texture visibility," *Proceedings of the IEEE CompCon* , 1997.