Multilayer Document Compression Algorithm *†

Hui Cheng and Charles A. Bouman
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285
{hui, bouman}@ ecn.purdue.edu

Abstract

In this paper, we propose a multilayer document compression algorithm. This algorithm first segments a scanned document image into different classes such as text, images and background, then compresses each class using an algorithm specifically designed for that class. Two algorithms are investigated for segmenting documents: a general purpose image segmentation algorithm called the trainable sequential MAP (TSMAP) algorithm, and a rate-distortion optimized segmentation (RDOS) algorithm. Experimental results show that the multilayer compression algorithm can achieve a much lower bit rate than most conventional algorithms such as JPEG at similar subjective distortion levels. We also find that the RDOS method produces more robust segmentations than TSMAP by eliminating misclassifications which can sometimes cause severe artifacts.

1 Introduction

Document images differ from natural images because they often contain well defined regions with distinct characteristics, such as text, line graphics, images and background. Typically, text and line graphics need high spatial resolution, but can tolerate low color resolution. On the other hand, images need high color resolution, but not high spatial resolution. Because of the spatial variation, traditional compression algorithms, such as JPEG, which assume the input image to be spatially homogeneous, tend to perform poorly on document images.

Several compression algorithms [1, 2] exploit the spatial variation of document images by using a 3-layer (foreground/mask/background) representation proposed in the ITU's Rec. T.44 for mixed raster content (MRC). The foreground layer consists of colors of text and line graphics, and the background layer contains images and background. The mask is a bi-level image which determines, for each pixel in the reconstructed image, if the foreground or the background color should be used.

In this paper, we propose a multilayer document compression algorithm. This algorithm first segments 8×8 blocks of pixels into different classes, such as text, image and background. Then, each class is compressed using an algorithm specifically designed for that class. Two segmentation algorithms are used for the multilayer compression algorithm: a general purpose image segmentation algorithm called the trainable sequential MAP (TSMAP) algorithm [3], and a rate-distortion optimized segmentation (RDOS) algorithm. The RDOS method has several advantages. First, RDOS produces more robust segmentations. Intuitively, misclassifications which cause severe artifacts are eliminated because all possible coders are tested for each block of the image. In addition, RDOS allows us compute a segmentation which represents a best trade-off between the bit rate and distortion.

The RDOS algorithm is similar in concept to techniques used by Ramchandran and Vetterli [4] and Effros and Chou [5]. However, our approach is different from previous ones in that we switch among different types of coders, instead of among sets of parameters for the same type of coder. In particular, we use a coder optimized for text representation that can not be represented as a DCT transform, VQ or KL transform coder. Another distinction is that different coders for our method use somewhat different distortion measures. This is motivated by the fact that perceived quality for text, image and background is different.

2 Multilayer Compression Algorithm

The multilayer compression algorithm proposed in this paper segments each 8×8 block of pixels into four classes: Image blocks, Two-color blocks, One-color blocks and Other blocks. Ideally, Image blocks should be from a region containing either continuous-tone or halftone image data, Two-color blocks should be from text or line graphics, and One-color blocks should be from uniform background regions. In addition, some regions of text and line graphics can not be accurately represented by Two-color blocks. For example, thin lines bordered by regions of two different colors require a minimum of three or more colors for accurate representation. We assign these problematic blocks to the Other blocks class.

Each of the four classes corresponds to a specific coding algorithm which is optimized for that document region. The details of each compression algorithm are discussed below. The Two-color blocks and One-color blocks, use color quantization as a preprocessing step to coding. Importantly, different classes use different color palettes for the quantization since this improves the quality without significantly increasing bit rate. In all cases, we use the binary splitting algorithm of [6] with 255 colors or less.

*The work was supported by Xerox IMPACT Imaging and Xerox Foundation.

†To appear in ICIP '99

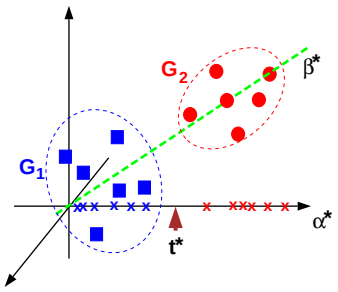


Figure 1: Minimal MSE thresholding. x's denote the projection of samples on α^*.

2.1 Compression of Image Blocks

Image blocks are compressed using JPEG with customized quantization tables. The luminance and chrominance components are handled separately using the following methods.

For the luminance component or Y component:

1. Luminance components of Image blocks are extracted and packed in raster order into a single image.
2. The new image is compressed using JPEG.

For the chrominance components or C_r and C_b:

1. Extract chrominance components from Image blocks, and form all 2×2 groups of blocks (i.e. 16×16 groups of pixels). The chrominance compression then depends on the number of Image blocks contained in each 2×2 group.
2. If the number of Image blocks in a 2×2 group equals
   1: Do not subsample the chrominance.
   2: Order the blocks and subsample the chrominance by 2 along the longest direction.
   3: Duplicate one Image block to fill in the missing block, and subsample by 2 along both directions.
   4: Subsample by 2 along both directions.
3. Pack the subsampled chrominance components to form a new image, and compress it using JPEG.

2.2 Compression of Two-color Blocks

Each Two-color block is represented by two indexed colors c_1, c_2 and a binary mask B. The algorithm for extracting two colors and computing the binary mask is described as follows. For an 8×8 block, if the algorithm can not extract two colors and a bilevel mask reliably, it will enlarge the block to 16×16 and extract the two colors and the mask from the large block.

1. Minimal MSE thresholding (see Fig. 1):
   (a) Project all colors onto the color axis α^* which has the largest variance among three axes.
   (b) For a threshold t on α^*, t partitions all colors into two groups. Let $E(t)$ be the mean squared error when colors in each group are represented by the mean color of that group. Find t^* which minimizes $E(t)$.
2. t^* partitions the block into two groups, G_1 and G_2. Let c_i be the mean colors of G_i, $i = 1, 2$. The binary mask B indicates the locations of G_1 and G_2. In B, use "0" to represent the group the brighter mean color.
3. Refinement. Find internal points of G_i, $i = 1, 2$, (an internal point means a point whose 8-nearest neighbors belong to the same class as the point does) and denote them as $\tilde{G}_i$.
   (a) If $|\tilde{G}_i| > 0$, re-set c_i to be the mean color of $\tilde{G}_i$.
   (b) If $|\tilde{G}_1| = 0$ or $|\tilde{G}_2| = 0$, and if the block size is 8×8, find the 16×16 block centering at the current block. Extract the two colors and a binary mask from the 16×16 block using the same algorithm (go to step 1).
4. Quantize colors extracted from all Two-color blocks.
5. Stack color indices to form two new images, and compress them using arithmetic coder.

The binary masks for each block are formed into a single binary image with the same height and width as the original document. Any block which is not a Two-color block is set to 0's in the binary mask, and the entire binary mask is encoded using a JBIG2 coder.

2.3 Compression of One-color Blocks

For One-color blocks, we first extract the mean color of each block. We then color quantize the mean colors of all One-color blocks. The color indices are entropy coded using an arithmetic coder. When reconstructing One-color blocks, smoothing is used among adjacent One-color blocks if their maximal differences along all coordinates are less than 12.

2.4 Compression of Other Blocks

Other blocks are compressed also compressed with JPEG. We use the same algorithm for compressing Image blocks, but apply the standard JPEG quantization table at quality 75.

3 TSMAP Segmentation Algorithm

The TSMAP segmentation algorithm is a general purpose segmentation algorithm proposed in [3]. We use TSMAP to first segment each block into Image blocks, Two-color blocks or One-color blocks. Other blocks are then selected from the Two-color blocks using the following post processing operation. We calculate the average distance (in rgb color space) of the boundary points (a boundary point is a point which is not an internal point of either G_1 or G_2) to the line determined by c_1 and c_2. If the average distance is larger than 45, then we re-classify the current block to an Other block. Also, if the total number of internal points of G_1 and G_2 is less than or equal to 8, then we re-classify the current block to a One-color block.

4 RDOS Algorithm

Although TSMAP usually results in satisfactory segmentations, it suffers from two major disadvantages which are shared among many other segmentation algorithms. First, it does not guarantee to avoid catastrophic misclassification. For example, even if only a few Two-color blocks

are misclassified as One-color blocks, the quality of the reconstructed image will drop significantly due to the broken lines and smeared text strokes caused by the misclassifications. In addition, the segmentation is usually computed independently of the bit rate and the quality desired by the user. This often causes inefficient use of bits and even artifacts in the reconstructed image. Since ultimately, each block is compressed with one coder, the optimal segmentation for compression should depend on both the image data and the properties of all coders.

Let y be the original image and x be the 8×8 block segmentation of y. The set of class labels for each block is $\mathcal{N}$, and the total number of blocks is L. Then, the rate-distortion optimized segmentation, x^*, is

$$x^* = \arg \min_{x \in \mathcal{N}^L} \{R(y|x) + R(x) + \lambda D(y|x)\}. \quad (1)$$

where $R(y|x)$ is the number of bits required to code y with segmentation x, $R(x)$ is the number of bits required to code x, $D(y|x)$ is the total distortion, and λ is a non-negative real number which is set by user which controls the trade-off between bit rate and distortion.

5 Computing RDOS

To compute the RDOS, we need to estimate the rate and the distortion for coding each block using each coder. For computational efficiency, we assume that the number of bites required for coding a block only depends on the image data and class labels of that block and the previous block in raster order. We also assume that the distortion can be computed independently from other blocks. Let x_i be the class label of the i-th block in raster order. Then, $x = \{x_0, x_1, \ldots, x_{L-1}\}$, and (1) can be rewritten as

$$x^* = \arg \min_{\{x_0,\ldots,x_{L-1}\}} \sum_{i=0}^{L-1} \{R_i(x_i|x_{i-1}) + R_x(x_i|x_{i-1}) + \lambda D_i(x_i)\} \quad (2)$$

where $R_i(x_i|x_{i-1})$ is the number of bits required to code block i as class x_i given x_{i-1}, $R_x(x_i|x_{i-1})$ is the number of bits needed to code x_i, and $D_i(x_i)$ is the distortion for coding block i with x_i.

Once we specify the expressions for $R_i(x_i|x_{i-1})$, $R_x(x_i|x_{i-1})$, and $D_i(x_i)$, then we may compute the exact solution to (2) using dynamic programming. For this work, we will assume that $R_x(x_i|x_{i-1})$ is constant. This is reasonable since the overhead for coding the segmentation is very small (typically ¡ 0.01 bits per pixel). In the follow sections, we will give approximate expressions for the rate and distortion resulting from each block classification.

5.1 One-color Blocks

For simplicity, the number of bits used for coding a One-color block is estimated with a first order approximation. Let μ_i be the color index of One-color block i. When x_i and x_{i-1} are all One-color blocks,

$$R_i(x_i|x_{i-1}) = -\log p_\mu(\mu_i|\mu_{i-1}),$$

where $p_\mu(\mu_i|\mu_{i-1})$ is the transition probability of color indices between adjacent blocks. When x_{i-1} is not a One-color block, we let $R_i(x_i|x_{i-1}) = -\log p_\mu(\mu_i)$. To estimate $p_\mu(\mu_i|\mu_{i-1})$ and $p_\mu(\mu_i)$, we assume that all blocks are One-color blocks, and compute the transition probabilities. In addition, the total squared error in YCrCb color space is used as the distortion measure of One-color blocks.

5.2 Two-color Blocks

For a Two-color block i, let $c_{i,0}$, $c_{i,1}$ be the two color indices, and let $b_{i,m,n}$ be the binary mask for block i where $0 \le m, n \le 7$. Then, the color of pixel (m, n) in block i is $c_{i,b_{i,m,n}}$. Let $p_j(c_{i,j}|c_{i-1,j})$ be the transition probability. If x_{i-1} is also a Two-Color block, the bits used for coding two color indices are approximated as $-\sum_{j=0}^{1} \log p_j(c_{i,j}|c_{i-1,j})$, If x_{i-1} is not a Two-Color block, we will use $p_j(c_{i,j})$ instead of $p_j(c_{i,j}|c_{i-1,j})$. We also assume that the number of bits for coding $b_{i,m,n}$ only depends on its four causal neighbors, denoted as $V = [b_{i,m-1,n-1}, b_{i,m-1,n}, b_{i,m-1,n+1}, b_{i,m,n-1}]^t$. Define $b_{i,m,n}$ to be 1, if $m < 0$ or $n < 0$ or $m > 7$ or $n > 7$. Then, the number of bits required to code the binary mask is approximated as $-\sum_{m=0}^{7}\sum_{n=0}^{7} \log p_b(b_{i,m,n}|V)$ where $p_b(b_{i,m,n}|V)$ is the transition probability from the four causal neighbors to the pixel (m, n). The probabilities $p_j(c_{i,j}|c_{i-1,j})$, $p_j(c_{i,j})$ and $p_b(b_{i,m,n}|V)$ are estimated for all 8×8 blocks whose maximal dynamic range along the three color axes is larger than 7.

The distortion measure used for the Two-color blocks is designed with following considerations. In a scanned image, pixels on the boundary of two color regions tend to have a color which is a combination of the colors of both regions. Since only two colors are used for a block, the boundaries between the color regions are usually sharpened. Although the sharpening generally improves the quality, it gives a large difference in pixel values on boundary points. On the other hand, if a block is not a Two-color block, a third color often appears on the boundary. Therefore, a desired distortion measure for Two-color coder should not excessively penalize the error caused by sharpening, but should produce a high distortion value, if more than two colors exist. This is especially important for distinguishing Two-color blocks from Other blocks. Also, desired Two-color blocks should have a certain proportion of internal points. Otherwise, the block usually comes from background or halftone background. To handle this case, we set the cost to the maximal cost, if the number of internal points is less than or equals to 8.

The distortion measure for the Two-color block is defined as follows. For a pixel (m, n) in block i, let $y_{i,m,n}$ be the original pixel value. We also define $I_{i,m,n}$ as an indicator function. $I_{i,m,n} = 1$, if (m, n) is an internal point. $I_{i,m,n} = 0$, if (m, n) is a boundary point. Then, if x_i is a Two-color block, and the number of internal points is greater than 8,

$$D_i(x_i) = \sum_{m=0}^{7}\sum_{n=0}^{7} \left[I_{i,m,n} \| y_{i,m,n} - c_{i,b_{i,m,n}} \|^2 + (1 - I_{i,m,n}) d^2(y_{i,m,n}; c_{i,0}, c_{i,1}) \right]$$

where $d(y_{i,m,n}; c_{i,0}, c_{i,1})$ is the distance between $y_{i,m,n}$ and the line determined by $c_{i,0}$ and $c_{i,1}$. If a color c is a combination of c_1 and c_2, c will be on the line determined by

c_1 and c_2. Then $d(c; c_1, c_2)$ is 0. However, if a third color does exist on a boundary point, $d(y_{i,m,n}; c_{i,0}, c_{i,1})$ tends to be large. If x_i is a Two-color block and the number of internal points is less than or equals to 8, set $D_i(x_i)$ to be $255^2 \times 64 \times 3$.

5.3 JPEG Blocks

For a JPEG block i (an Image block or an Other block),

$$R_i(x_i|x_{i-1}) = R_i^l(x_i|x_{i-1}) + R_i^c(x_i|x_{i-1})$$

where $R_i^l(x_i|x_{i-1})$ and $R_i^c(x_i|x_{i-1})$ are the bits for coding the luminance and the chrominance, respectively. Let $\alpha_i^d(x_i)$ be the quantized DC coefficients of the luminance using the quantization table of class x_i, and $\alpha_i^a(x_i)$ as the vector of the quantized AC coefficients. Then, assuming fixed Huffman tables,

$$R_i^l(x_i|x_{i-1}) = r_d[\alpha_i^d(x_i) - \alpha_{i-1}^d(x_{i-1})] + r_a[\alpha_i^a(x_i)].$$

where $r_d(\cdot)$ is the number of bits used for coding the difference between two consecutive DC coefficients, and $r_a(\cdot)$ is the number of bits used for coding AC coefficients. The formula of calculating $r_d(\cdot)$ and $r_a(\cdot)$ is specified in the JPEG standard. Notice that when x_{i-1} is also a JPEG class, $R_i(x_i|x_{i-1})$ is the exact number of bits required for coding the luminance. If x_{i-1} is not a JPEG class, we assume the previous quantized DC value to be 0.

Since the two chrominance components are subsampled 2×2, we approximate the number of bits for coding the chrominance $R_i^c(x_i|x_{i-1})$, as follows. Let j be the index of the 16×16 block which contains block i. Also, let $\beta_{j,k}^d(x_i)$ be the quantized DC coefficient of the k-th chrominant component of 16×16 block j of class x_i, and $\beta_{j,k}^a(x_i)$ be quantized AC coefficients. Then,

$$\begin{aligned} R_i^c(x_i|x_{i-1}) &= \frac{1}{4}\sum_{k=0}^{1}\{r'_d[\beta_{j,k}^d(x_i) - \beta_{j-1,k}^d(x_i)] \\ &\quad + r'_a[\beta_{j,k}^a(x_i)]\}, \end{aligned}$$

where $r'_d(\cdot)$ and r'_a are the number of bits for coding the DC, and the AC coefficients of the chrominance, respectively. Notice that we split the bits used for coding the chrominance equally among the four 8×8 blocks of the original image, and assume that the classes of the chrominance blocks j and $j-1$ are x_i.

The total squared error in YCrCb is used as the distortion measure for JPEG blocks. As with the rate, each 16×16 chrominance block's distortion is equality split among the four 8×8 blocks of the original image.

6 Compression Results

For our experiments, 30 document images are scanned at 400 dpi. A large portion of the 30 images contain halftone background and have ghosting artifacts caused by printing on the reverse side of the page. These images are used without pre-processing. For TSMAP, we use the parameters extracted from 20 grayscale images scanned at 100 dpi [3]. These training images are manually segmented into 3 classes: text, image and background.

For comparison, after an image is compressed using the TSMAP segmentation, we will adjust λ to achieve the same bit rate with RDOS. Fig. 2 shows both the resulting segmentations of a test image using both TSMAP and RDOS. The test image is compressed 207:1 (0.116 bpp) with RDOS, 200:1 (0.120 bpp) with TSMAP, and 132:1 (0.182 bpp) using JPEG. Portions of the reconstructed images compressed using different algorithms are shown in Fig. 3. We see that in both segmentations, most of the regions are segmented correctly. Text lines, captions and even page numbers are located accurately. But many Two-color blocks are misclassified as One-color blocks in the TSMAP segmentation, This causes severe artifacts shown in Fig. 3 (d). This type of misclassification did not occur in the RDOS segmentation. In all cases, the multilayer compression algorithm using either TSMAP or RDOS outperform JPEG significantly.

7 Conclusion

In this paper, we propose a spatially adaptive compression algorithm for document images which we call multilayer compression. This algorithm first segments a scanned document image into different classes and compresses each class with an algorithm specifically designed for that class. Experimental results show that this compression algorithm can achieve much lower bit rates than conventional compression algorithms, such as JPEG, at similar distortion levels. Two segmentation algorithms are used in the compression algorithm: the TSMAP algorithm and the RDOS algorithm. We found that the RDOS algorithm is more robust than the TSMAP algorithm. The RDOS can eliminate the catastrophic misclassifications, such as misclassifying a Two-color block to a One-color block.

References

[1] L. Bottou, P. Haffner, P. G. Howard, P. Simard, Y. Bengio, and Y. LeCun, "High quality document image compression with 'DjVu'," *Journal of Electronic Imaging*, vol. 7, no. 3, pp. 410–425, July 1998.

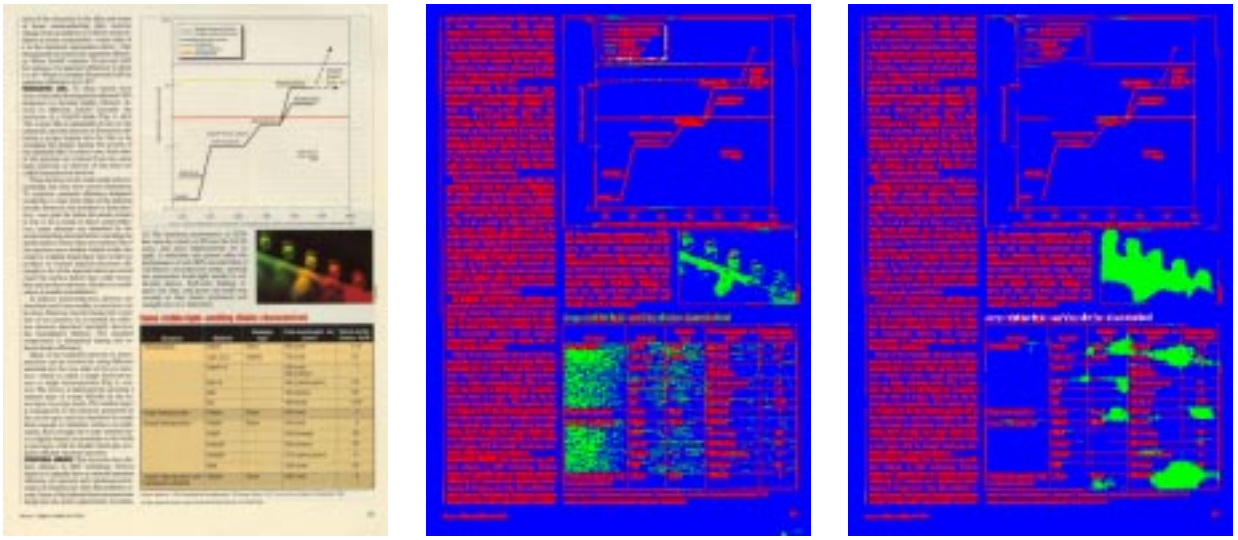
[2] R. L. de Queiroz, R. Buckley, and M. Xu, "Mixed raster content (MRC) model for compound image compression," *Proc. IS&T/SPIE Symp. on Electronic Imaging, Visual Communications and Image Processing*, vol. 3653, Februray 1999, San Jose, CA, pp. 1106–1117.

[3] H. Cheng and C. A. Bouman, "Trainable context model for multiscale segmentation," *Proc. of IEEE Int'l Conf. on Image Proc.*, vol. 1, October 4-7 1998, Chicago, IL, pp. 610–614.

[4] K. Ramchandran and M. Vetterli, "Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility," *IEEE Trans. on Image Processing*, vol. 3, no. 5, pp. 700–704, September 1994.

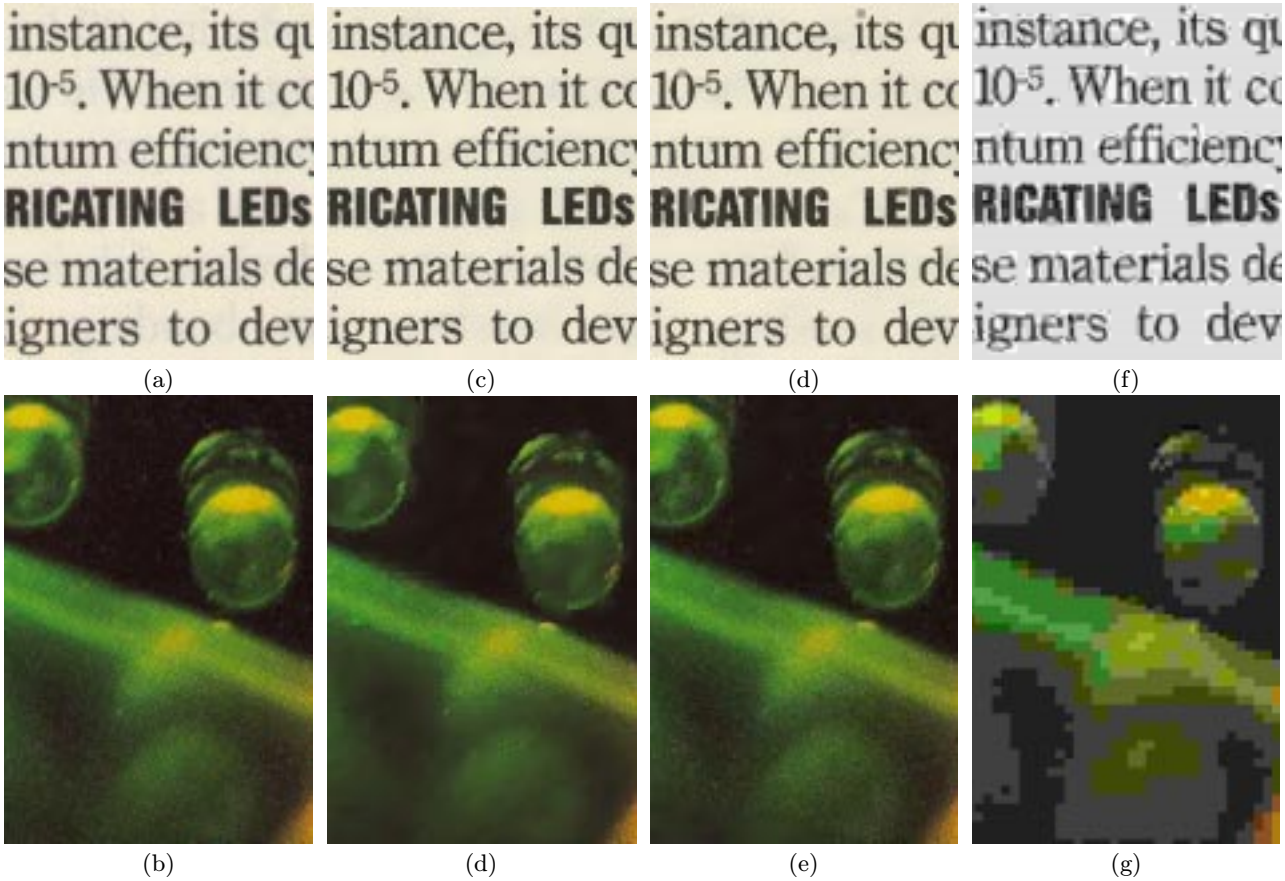
[5] M. Effros and P. A. Chou, "Weighted universal bit allocation: optimal multiple quantization matrix coding," *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, vol. 4, May 1995, Detroit, MI, pp. 2343–2346.

[6] M. Orchard and C. A. Bouman, "Color quantization of images," *IEEE Trans. on Signal Processing*, vol. 39, no. 12, pp. 2677–2690, December 1991.



(a) (b) (c)

Figure 2: Segmentation results. (a) original image. (b) RDOS segmentation with $\lambda = 0.003$. (c) TSMAP segmentation. Red, green, blue, white represent Two-color, Image, One-color, and Other blocks, respectively.



(a) (c) (d) (f)

(b) (d) (e) (g)

Figure 3: Comparison between the multilayer compression algorithm and JPEG. (a)-(b) portions of the original image. (c)-(d) portions of the reconstructed image compressed with the RDOS segmentation at 0.116 bpp ($\lambda = 0.003$). (d)-(e) portions of the reconstructed image compressed with the TSMAP segmentation at 0.120 bpp. (f)-(g) portions of the reconstructed image compressed by JPEG at 0.182 bpp