Quality Control in Lossy Compression of Digital Images

Viresh Ratnakar
ratnakar@cs.wisc.edu
Computer Sciences Department
University of Wisconsin
Madison, WI 53706

August 18, 1995

1 Introduction

The storage and transmission media of today's computers are heavily populated with digital images and video streams. Apart from entertainment and scientific applications, digital images are now an integral part of most documents, databases, and World Wide Web pages. The price associated with this multimedia revolution is that of the tremendous storage and bandwidth requirements. A single full-color image typically occupies more than one megabyte. Digital video, where every second of "action" uses about 15-25 still images, require even higher storage space and transmission bandwidth. This has necessitated the use of efficient compression strategies for digital images and video. Lossless compression, where the decompressed image will be the same as the original image, typically offers compression ratios under 2:1, which is insufficient for most applications. Lossy compression, where the decompressed image may be slightly different from the original, offers a reasonable solution to the storage and bandwidth problem for most applications. A nice overview of digital images and their compression can be found in [Jai89].

Lossy compression can compress an image to any extent—the greater the compression, the greater the degradation of the decompressed image as compared to the original. The tradeoff between compression and quality depends upon by the particular compression scheme used and the characteristics of the image being compressed. Figure 1 shows the typical nature of the quality-compression curve. The dark line shows the best quality achievable for a given compressed size. The shaded region is sub-optimal in terms of the quality-compression tradeoff. With most compression schemes, it is possible to tune some parameters so as to achieve a quality-compression point anywhere on or under the curve.

Our work is concerned with efficiently exploiting quality-compression tradeoffs in various image compression techniques, developing new compression techniques which offer flexibility in choosing image quality in application-specific ways, and studying the effect of image quality on various applications.

2 Quality of lossy-compressed images

The notion of *quality* of an image as compared to an original, is intuitively apparent, but hard to quantify. Many different quality measures exist, most of them being based on *distortion* (mean-squared error

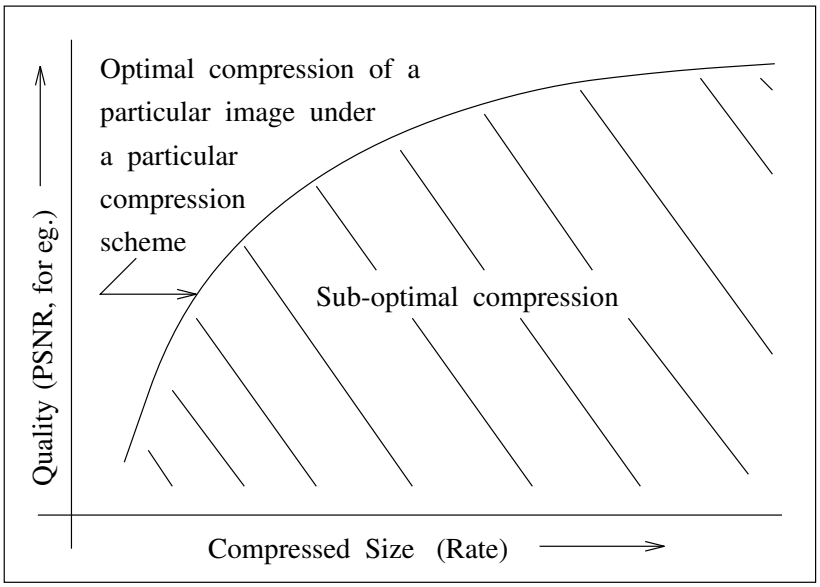

Figure 1: Quality-Compression tradeoff in lossy compression.

between pixel values in the original image and the copy of it that has undergone lossy compression). Each measure has its limitations, especially when comparing the quality of an image compressed using different techniques. Peak-Signal-to-Noise-Ratio (PSNR), and perceptually weighted PSNR are the most commonly used distortion-based measures. Distortion is easy to compute and is usually an adequate quality measure when comparing the results of compression on a particular image using a particular technique at different compression ratios.

For the purpose of this report, we will refer to an image with width W, height H, and 8-bit pixel intensities in the range $[0 \ldots 255]$ as a $W \times H$ *image*. In an image I, the pixel intensity in row i and column j is referred to as $I[i, j]$. All the results and discussion are equally applicable to images with greater resolution, images with several color planes, and sequences of images.

Let I be a $W \times H$ image, $c(I)$ be a lossy-compressed copy of I and I' be the decompressed form of $c(I)$. Let $|c(I)|$ be the size of $c(I)$ in bits. Then, the compression ratio is $\frac{W \times H \times 8}{|c(I)|}$. It is common to measure the extent of compression in terms of *rate*, defined as

$$\frac{|c(I)|}{W \times H} \text{ bits per pixel (bpp).}$$

The *distortion* $\mathcal{D}(I, I')$ caused by compression is,

$$\mathcal{D}(I, I') = \frac{1}{W \times H} \sum_{\substack{i = 1 \ldots H \\ j = 1 \ldots W}} (I[i, j] - I'[i, j])^2.$$

Peak-Signal-to-Noise-Ratio is defined as,

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\mathcal{D}(I, I')} \text{ dB.}$$

The distortion $\mathcal{D}(I, I')$ may be calculated in a spatial frequency domain, with different weights attached to different frequencies, giving a weighted PSNR. The weights give more importance to distortion in the lower frequencies, which are perceptually more significant.

3 The need for quality control

It is clear that any compression technique should try to provide the best possible quality at any rate. However, most existing implementations of common compression techniques do not give the best possible quality, as the previous methods to do so were prohibitively expensive in terms of computation time. The gains in quality possible over what is delivered can be substantial, as was shown by us in [RL94].

Another desirable feature would be to allow the user to specify any desired rate, and compress an image to that rate, perhaps interactively, after examining the rate-quality tradeoff curve. Users should also be able to specify quality using a quality measure most useful for their needs, and get the least possible rate.

Consider a production-mode compression environment, where users can submit arbitrary images for compression, along with desired rate/quality specifications as above. It is crucial that the compression parameters be set automatically to best exploit the characteristics of the particular images being compressed. The images submitted may vary dramatically across users: what works well for one image may perform poorly on another, hence a production-mode compressor must be able to efficiently control quality for any image, rather than rely on defaults.

As another example, consider an image archive server from which remote clients can retrieve various images. The server will have a limited amount of diskspace, and the connecting network will provide a limited amount of bandwidth. It would be crucial to utilize these resources in the best possible manner, that is, to store the images in compressed form, and to do the compression in a way that would give the best possible quality at the available rate. Further, it would be important to provide browse capability to the clients, enabling them to quickly preview low-quality versions of the images, select some of the images of interest, and specify a desired quality or rate for retrieval of these images.

Scientific applications typically run classification software and other analytical tools on images acquired using remote sensing, microscopy, and various other methods. With the vast amount of image data being gathered, lossy compression provides an invaluable tool for storing it compactly. However, these applications need strict quality control. For example, it might be necessary to have distortion below a certain level, for the analyses to be useful. It would be very important to get the best possible compression, while keeping the distortion within the tolerance. For some applications, such as those running analyses on images, it is crucial that the distortion be bounded *everywhere* in the image, and not just in a mean, global sense. The effect of distortion on these applications merits further study in order to determine their tolerance to loss in image quality.

4 A brief overview of some compression techniques

In this section, we look at some of the current image compression techniques that we have worked with, or plan to work with. For each one of these, we will also list some of the parameters that can be set to vary rate and quality. We refer to these parameters as *Rate-Quality Knobs*.

4.1 Discrete Cosine Transform based techniques

The Discrete Cosine Transform (DCT) [ANR74] is perhaps the most popular tool in image compression today. The JPEG standard for still-image compression [Wal91, PM93], the MPEG-I and MPEG-II standards for video compression [MP91, Le 91], H.261 standard for video telephony [Lio91], etc., all employ the DCT to transform an image into components with different spatial frequencies. Since pixel intensities typically vary slowly among neighboring pixels, most of the image structure is captured in the low-frequency components.

This allows the information in the image to be selectively discarded or stored with varying degrees of accuracy: the higher frequencies are less important visually, and contain the bulk of the noise introduced by the image-capturing process [RY90]. Further, DCT has the nice property of being close to the Karhunen-Loeve-Hotelling transform (KLH), in that the different frequency components are nearly uncorrelated, which allows them to be compressed independently [RY90].

The most commonly used variant of DCT is the 2-dimensional DCT on 8×8 image blocks, which we shall refer to as DCT, hereafter. Our work on DCT-based compression is easily extendible to other block sizes as well. Let f be an 8×8 image block. We refer to individual pixel intensities in f as $f[i, j]$, for $i, j = 0 \ldots 7$. Applying the DCT to f results in an 8×8 block $\hat{f}$ of 64 coefficients that represent f as a linear combination of the 64 orthonormal DCT basis blocks. The lowest spatial frequency coefficient is $\hat{f}[0, 0]$, while higher frequencies in horizontal and vertical directions are captured by the coefficients $\hat{f}[u, v]$ with greater values of v and u, respectively. The Inverse Discrete Cosine Transform (IDCT), converts $\hat{f}$ back to f. The equations governing DCT and IDCT are:

$$\text{DCT:} \quad \hat{f}[u,v] = \frac{1}{4}C(u)C(v)\sum_{i=0\ldots7}\sum_{j=0\ldots7} f[i,j]\cos[\frac{(2i+1)u\pi}{16}]\cos[\frac{(2j+1)v\pi}{16}] \tag{1}$$

$$\text{IDCT:} \quad f[i,j] = \frac{1}{4}\sum_{u=0\ldots7}\sum_{v=0\ldots7} C(u)C(v)\hat{f}[u,v]\cos[\frac{(2i+1)u\pi}{16}]\cos[\frac{(2j+1)v\pi}{16}] \tag{2}$$

Where,

$$C(x) = \begin{cases} 1/\sqrt{2} & \text{for } x = 0 \\ 1 & \text{otherwise.} \end{cases}$$

The DCT-based image compression process typically consists of the following steps [RY90].

1. A given image I is divided into 8×8 blocks. To each image block f, the DCT is applied to get an 8×8 block $\hat{f}$ of DCT coefficients.

2. An 8×8 block of integers Q, called the quantization table, is used to *quantize* the coefficients in $\hat{f}$ to form the block $\hat{f}_Q$ of quantized coefficients. Quantization is defined as:

$$\hat{f}_Q[i,j] = \hat{f}[i,j] /\!\!/ Q[i,j], \;\; 0 \le i,j \le 7,$$

   where $/\!\!/$ represents division followed by rounding to the nearest integer.

3. The quantized blocks are entropy-coded to exploit similarities across blocks. Quantization typically sets most of the high frequency coefficients to zero, which allows the quantized blocks to be compactly stored. For example, in JPEG, the block $\hat{f}_Q$ can be coded using Huffman Coding [Huf52] of non-zero values and run-lengths of zeros [PM93]. The sequence of these compressed blocks forms the compressed image.

The decompression process reverses these steps as follows:

1. Each entropy-coded block is decoded to get the corresponding block of quantized coefficients.

2. Dequantization is done on each block $\hat{f}_Q$ of quantized coefficients to construct the block $\hat{f}'$, as follows:

$$\hat{f}'[i,j] = \hat{f}_Q[i,j] \cdot Q[i,j], \;\; 0 \le i,j \le 7.$$

3. The IDCT is applied to $\hat{f}'$ to get the decompressed image block f'. These decompressed blocks form the decompressed image I'.

The lossiness of the compression is essentially because of the quantization step ($\hat{f} \to \hat{f}_Q$), as in general,

$$\hat{f}'[i,j] = \hat{f}_Q[i,j] \cdot Q[i,j] = (\hat{f}[i,j] /\!\!/ Q[i,j]) \cdot Q[i,j] \neq \hat{f}[i,j].$$

This causes differences in pixel values between the original image block f and its approximation, the decompressed image block f'.

Rate-Quality Knobs: The quantization table Q determines the quality and rate of compression. The greater the entries in Q, the less accurate the reproduction of the DCT coefficients, and hence, the poorer the quality. However, greater entries in Q quantize more coefficients to zero, enabling greater compression (smaller rate). There are 64 entries in Q, each one can (usually) be set to any integer between 1 and 255.

4.2 Fractal-based image compression

Fractal compression tries to exploit self-similarity in images [Bar93]. The idea is to find a contractive transformation[1] w that maps a $W \times H$ image I to itself ($w(I) = I$). This is usually done by dividing the image I into square blocks, and for each block, finding an approximately similar block on a shrunk version $s(I)$ of the same image. Specifically, for each block f, a block g on $s(I)$ is found, together with scalars s, o, and orientation[2] τ such that the difference between f and $s \cdot \tau(g) + o$ is minimized [Jac92]. The collection of these block-transformations forms a transformation w for the entire image I, such that $w(I) \approx I$.

The remarkable property of such transformations w is stated as the Collage Theorem [Bar93] which says that starting with any arbitrary $W \times H$ image I_0, repeated applications of w converge to a fixed image I' regardless of I_0, and that I' is an approximation of I. Thus, to compress an image, one simply needs to find the triple (s, o, τ) for each block f, and store it compactly, along with the location of the matching block g. The decompression process starts with any arbitrary image I_0, and repeatedly applies the transformation w by shrinking the image, and reconstructing every block f as $s \cdot \tau(g) + o$. After a few iterations, this process converges to the decompressed approximation I'.

Details on fractal compression, along with variations such as quadtree blocking, can be found in [Fis92] and [JFB92].

Rate-Quality Knobs: Rate is varied by storing s, o, and τ for each block with varying accuracy. For example, s might be constrained to to a fixed value, o to an integer between -128 and $+127$, and τ might be constrained to be the identity (i.e., no reorientation). Not all locations for the matching block g may be tried (say, by restricting g to be in a fixed region around f), in which case fewer bits would be needed to store the location of g. All these knobs can vary rate and quality, but the quality cannot be made higher than that determined by the extent of self-similarity in the image (there might not be *any* good matches for some image blocks!).

4.3 Vector Quantization

Vector Quantization (VQ), is perhaps the most general image compression technique. The idea is to split the image I into blocks or *vectors* of a certain kind (for example, 8×8 blocks, or rectangular tiles, or tiles of other shapes, etc.) and to represent these vectors by closest approximations from a small set of vectors, called the *codebook*. For example, if a codebook $C = \{c_i \mid i = 1 \ldots M\}$ of size M is used, then each vector

[1] A transformation is said to be *contractive* if the distance between any two transformed images is less than the distance between the images themselves, where *distance* is a metric on the space of images.

[2] There are eight different orientations that map a square onto a square: four possible rotations and a flip followed by four possible rotations.

v in the image would be stored in compressed form as the index i of the codebook vector c_i closest to v. Thus each vector can be stored using $\log_2 M$ bits (which can be reduced further by entropy coding). A comprehensive reference for VQ is [GG92].

For efficient use of VQ, it is very important that codebooks be reusable. Codebook design is usually done by a process called *training* which takes a number of images and a target size, and produces a codebook of that size that works well on all the images in the training set. The codebook needs only to be sent once to the decompressor, enabling it to decompress all the images compressed using that codebook.

Rate-Quality Knobs: The choice of a codebook size, and the codebook itself determine rate and quality of compression for an image.

Part I
Completed Work

We now present an overview of the work that has been completed. This can be classified into three broad areas:

1. DCT-based compression
2. Scientific applications
3. Fractal compression

5 Results in DCT-based compression

As described in section 4.1, the main Rate-Quality Knob is the quantization table. Most existing compressors use a default quantization table, and scale it up or down by a small factor to vary rate and quality. We did an extensive study on compressing a large number of images acquired from a wide variety of sources (satellite images, microscopy images from Molecular Biology, standard images used in compression literature). We used default tables for each image, and compared the results with those using *customized* tables obtained by searching over a large portion of the search space of all possible 8×8 tables. The results clearly demonstrated the need for using customized quantization tables, as for most images they offered substantial improvements in quality over scaled default tables. This work [RL94] can be found in Appendix A.

Our next step was to look for an efficient algorithm to design image-specific quantization tables to meet arbitrary rate/quality demands in the best possible ways. The previous work done in this area mostly used heuristic searches over all possible tables, using the entire JPEG compressor as a black-box to evaluate search points and decide search directions [MS93, WG93]. This is very expensive, and does not guarantee optimality. Other quantization table design strategies have relied on psycho-visual models [AP92, Wat93]. Our most important work is the development of the RD-OPT algorithm, which efficiently optimizes quantization tables for arbitrary images, to meet an arbitrary range of rate/quality specifications. A wide variety of distortion-based quality measures, including PSNR and weighted PSNR, can be used with RD-OPT. RD-OPT works by gathering DCT coefficient statistics for a given image, and uses these statistics in a novel way to build tables that predict the rate and quality resulting from any quantization table in terms of sums of coefficient-wise rates and qualities. It then runs a dynamic program to optimize rate against quality. This work [RL95] can

be found in Appendix C. We have implemented RD-OPT in C, and the implementation is being widely used for designing quantization tables.

In connection with the development of RD-OPT, we did a study of runlength coding of quantized DCT coefficients, that led to the idea of coefficient-wise decomposition of rate as used in RD-OPT. The study showed that the rate resulting from runlength coding is very close to the coefficient-wise sum of entropies of the various quantized coefficients. The results, along with analytical characterizations of rate, are presented in [RFVK94], included here as Appendix B.

6 Work done on scientific applications

We currently cater to the image compression needs of a wide variety of scientific users at the University of Wisconsin. These include Molecular Biologists with microscopy images, Soil Scientists with remote-sensed images of canopies, Geologists with images of crystals, and Space Scientists with satellite images.

The Soil Scientists are interested in running classification software on images, for instance, to classify canopy image pixels into those belonging to shaded/sunlit leaves, branches, sky, etc. We did a study to evaluate the possibility of using lossy-compressed images for these analyses. We compared the results of classification on the original images to those on the lossy-compressed images at various qualities, and found that the errors in classification can be tolerably low (and, more importantly, random) with compression ratios as high as 10:1. This represents a substantial saving of storage for these scientists. This work [RLNK95] is included here as Appendix D.

The Geologists are interested in using crystal images for studying the crystal structure, as well as identifying defects. For the former application, they were using expensive Fourier analysis software for image enhancement. We have developed a new enhancement/compression technique specifically designed for images of crystals. Since crystal images are periodic, the same pattern repeats over and over in the image. We use a simple algorithm to detect this periodicity in a crystal image, and to automatically divide the image into units that are copies of each other. These images typically contain a lot of noise (hence the need for enhancement). We assume this noise to be random, and create an *enhanced unit* by averaging the pixel intensities over all the units gleaned from the image, which eliminates the noise. The enhanced image can be stored in its compressed form by just storing the enhanced unit. We have carried tests on several images, and the results compare very well with those obtained using expensive Fourier analysis. This stresses the usefulness of designing application-specific compression strategies. This work has not yet been published.

7 Results in fractal compression

Since fractal compression is computationally expensive (because of expensive searches for matching blocks), and yet does not perform as well as other techniques like JPEG [JFB92], we tried to develop a hybrid scheme: We used a simple variant of fractal compression to compress images down to very low rates. The decompressed images were used as *predictors* in the hybrid scheme, and DCT was used to compress the error image (original minus predicted) at any desired rate. However, even the hybrid scheme could not beat JPEG in terms of PSNR. We also showed that for fractal compression to work well, the self-similarity in an image must be of a very specific kind. This was demonstrated by showing a totally self-similar image that could not be compressed well by fractal compression. This work [RFT94] can be found in Appendix E.

The compression in fractal techniques results from self-similarity in an image. The requirement that the self-similarity be expressible as a contractive transformation comes only to enable decompression using

convergent iterations as described in the Collage Theorem (see Section 4.2). We have developed an image compression technique called *spatial prediction* that exploits self-similarity but does not require contractivity. The compressor processes image blocks in raster order. For each block, a good match is found in the already encoded portion of the image. If no match is found that meets the desired quality requirement, then the block is encoded using DCT. Otherwise, the block is encoded by storing the location of the matching block and the triple (s, o, τ) that describes the matching transformation (as in fractal compression, see Section 4.2). The decompressor also processed blocks in raster order, and can decompress both predicted blocks (applying the appropriate transformation to the matching block which is in the part of the image that has already been decompressed) and DCT-coded blocks. Spatial prediction works better than JPEG at extremely low bit rates, as the decompressed images show far fewer compression artifacts. At higher rates, JPEG outperforms spatial prediction. This work [FPR95] is included here as Appendix F.

Part II
Future Work

In this part of the report, we discuss some of the issues we plan to deal with in the rest of our thesis.

DCT-based compression: There are some improvements to be made in RD-OPT. Another Rate-Quality Knob in DCT-based compression is the ability to arbitrarily set some coefficients in some blocks to zero. This allows smaller quantization table entries, as most of the compression results from the presence of zeros in quantized coefficient blocks. We would like to incorporate this zeroing capability in RD-OPT. We are also working on implementing a graphical user interface to RD-OPT, whereby users will be presented with the rate-quality curve for their images, allowing them to interactively choose any point on the curve.

For some applications, such as image classification, it would be useful to guarantee bounds on distortion everywhere in the image, and not just in the mean sense. We plan to develop quantization strategies to meet such *uniform quality* demands. The idea would be to cluster similar image blocks together and use one quantization table for each cluster, such that every block has bounded distortion.

Vector quantization: This will be the primary area to explore for our future work. There are several problems associated with VQ codebooks. While there has been considerable work done on developing training algorithms [LBG80, GG92], VQ has not gained popularity as an efficient compression technique. We feel that for VQ to be efficiently used, there need to be techniques to efficiently maintain and use a collection of codebooks. The compressor and decompressor would share a certain number of codebooks, created using a training algorithm. For a given new image to be compressed to meet certain rate/quality specifications, the first consideration would be whether one of the existing codebooks would meet those specifications (the problem of *codebook selection*). If not, perhaps it might be useful to expand one of the existing codebooks (the problem of *codebook evolution*). Otherwise, a new codebook must be created. These individual problems, along with the decisions listed above, are issues that we plan to address.

Effects of lossy compression on scientific applications: We plan to conduct tests similar to those described in Appendix D on various other applications. Classification, feature detection, etc. are some of the common analytical applications used in the scientific community. To explore the promising possibility of using lossy-compressed images in these applications, one needs to study the effect of lossy compression on the results given by these various applications.

8 Conclusions

We have shown the need for efficient quality control in lossy compression of digital images. For DCT-based compression techniques, we have developed the RD-OPT algorithm to efficiently solve the quality control problem. Similar techniques need to be developed for other compression techniques such as VQ. Further, a complete VQ compression environment needs to be developed which efficiently solves not only the codebook training problem, but also the codebook selection and evolution problems. The feasibility of using lossy compression with accurate quality control in scientific applications needs to be demonstrated, as scientists are usually wary of "losing" any part of their data.

References

[ANR74] Ahmed, N., Natarajan, T., and Rao, K. R. Discrete Cosine Transform. *IEEE Trans. Computers*, C-2390-3, Jan. 1974.

[AP92] Ahumada Jr., A. J. and Peterson, H. A. Luminance-Model-Based DCT Quantization for Color Image Compression. *Human Vision, Visual Processing, and Digital Display III, B. E. Rogowitz, ed. (Proceedings of the SPIE)*, 1992.

[Bar93] Barnsley, M. F. *Fractals Everywhere.* Academic Press Professional, Cambridge, MA, second edition, 1993.

[Fis92] Fisher, Y. Fractal Image Compression, 1992. SIGGRAPH '92 Course notes.

[FPR95] Feig, E., Peterson, H., and Ratnakar, V. Image Compression Using Spatial Prediction. *Proc. Inter. Conf. Acoustics, Speech and Signal Processing*, May 1995.

[GG92] Gersho, A. and Gray, R. M. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, 1992.

[Huf52] Huffman, D. A. A Method for the Construction of Minimum Redundancy Codes. *Proc. IRE.*, 40(9):1098–101, Sept. 1952.

[Jac92] Jacquin, A. Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations. *IEEE Transactions on Image Processing*, 1:18–30, 1992.

[Jai89] Jain, A. K. *Fundamentals of Digital Image Processing.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[JFB92] Jacobs, E. W., Fisher, Y., and Boss, R. D. Image compression: A study of the iterated transform mathod. *Signal Processing*, 29(3):251–263, December 1992.

[LBG80] Linde, Y., Buzo, A., and Gray, R. M. An algorithm for vector quantizer design. *IEEE Trans. Commun.*, COM - 28:84—95, Jan. 1980.

[Le 91] Le Gall, D. MPEG: A Video Compression Standard for Multimedia Applications. *Commun. ACM*, 34(4), April 1991.

[Lio91] Liou, M. Overview of the px64 kbit/s Video Coding Standard. *Commun. ACM*, 34(4), April 1991.

[MP91] MPEG I draft: Coding of Moving Pictures and associated audio for digital storage, 1991. Document ISO/IEC-CD-11172.

[MS93] Monro, D. M. and Sherlock, B. G. Optimum DCT Quantization. *Proceedings of Data Compression Conference*, pages 188–194, 1993.

[PM93] Pennebaker, W. B. and Mitchell, J. L. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, New York, 1993.

[RFT94] Ratnakar, V., Feig, E., and Tiwari, P. Fractal Based Hybrid Compression Schemes. *Proceedings of SPIE's VCIP*, 1994.

[RFVK94] Ratnakar, V., Feig, E., Viscito, E., and Kalluri, S. Runlength encoding of quantized DCT coefficients. *IBM RC 19693 (87318) 8/5/94 (Also in Proceedings of SPIE '95)*, 1994.

[RL94] Ratnakar, V. and Livny, M. Performance of Customized DCT Quantization Tables on Scientific Data. *Science Information Management and Data Compression Workshop Proceedings, NASA Conference Publication 3277*, pages 1–8, Sept 1994.

[RL95] Ratnakar, V. and Livny, M. RD-OPT: An Efficient Algorithm For Optimizing DCT Qunatization Tables. *Proceedings of Data Compression Conference (Also, Technical Report 1257, Dept of Computer Sciences, UW-Madison)*, pages 332–341, 1995.

[RLNK95] Ratnakar, V., Livny, M., Norman, J. M., and Kucharik, K. Classification on compressed images with bounded loss. *International Geoscience and Remote Sensing Symposium Proceedings*, July 1995.

[RY90] Rao, K. R. and Yip, P. *Discrete Cosine Transform: Algorithms, Advantages, Applications.* Academic Press, Inc, San Diego, California, 1990.

[Wal91] Wallace, G. K. The JPEG Still Picture Compression Standard. *Commun. ACM*, 34(4), April 1991.

[Wat93] Watson, A. B. DCT quantization matrices visually optimized for individual images. *Human Vision, Visual Processing, and Digital Display IV, B. E. Rogowitz, ed. (Proceedings of the SPIE)*, 1993.

[WG93] Wu, S. and Gersho, A. Rate-constrained picture-adaptive quantization for JPEG baseline coders. *Proc. Inter. Conf. Acoustics, Speech and Signal Processing*, 5:389—392, April 1993.

Part III

Appendix

A Performance of Customized DCT Quantization Tables on Scientific Data

B Runlength Encoding of Quantized DCT Coefficients

C RD-OPT: An Efficient Algorithm for Optimizing DCT Quantization Tables

D Classification on Compressed Images with Bounded Loss

E Fractal Based Hybrid Compression Schemes

F Image Compression Using Spatial Prediction