Rochester Institute of Technology
RIT Digital Institutional Repository

Theses

1993

A reference guide to JPEG compression

Daniel P. Goodenow

Follow this and additional works at: https://repository.rit.edu/theses

Recommended Citation
Goodenow, Daniel P., "A reference guide to JPEG compression" (1993). Thesis. Rochester Institute of Technology. Accessed from

This Thesis is brought to you for free and open access by the RIT Libraries. For more information, please contact repository@rit.edu.

School of Printing Management and Sciences
Rochester Institute of Technology
Rochester, New York

Certificate of Approval

Master's Thesis

This is to certify that the Master's Thesis of

Daniel Philip Goodenow

With a major in Graphic Arts Publishing
has been approved by the Thesis Committee as satisfactory
for the thesis requirement for the Master of Science degree
at the convocation of
May, 1993

Thesis Committee:

Thesis Advisor

Graduate Program Coordinator

Director or Designate

A Reference Guide to JPEG Compression

by

Daniel P. Goodenow

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in the
School of Printing Management and Sciences in the College
of Imaging Arts and Sciences of the
Rochester Institute of Technology

February 1993

Thesis Advisor: Professor Frank Cost

Acknowledgments

First and foremost I would like to thank my parents, Gary and Shari, who not only provided the financial support for my studies, but also gave me the encouragement to pursue a career I enjoy. Secondly, I would like to thank my wife, Kathy, who put up with me during the many nights I stared into my computer trying to finish this project. Finally, I would like to thank professor's Frank Cost and Marie Freckleton who suggested ideas and helped to organize and proof read the material in this reference.

TABLE OF CONTENTS

Chapter 1 — Introduction

Footnotes 1–4

Chapter 2 — Theory & Background

Background 2–1
Image Blocking 2–2
Discrete Cosine Transform 2–3
Coefficient Quantization 2–5
Lossless Compression 2–8
Decompression 2–10
Color 2–12
Footnotes 2–14

Chapter 3 — Compression Considerations

Image Resolution 3–1
Image Frequency 3–1
Speed Requirements 3–2
Features 3–3
Intended Audience and Viewing Conditions 3–4
Output Devices 3–5
Compatibility 3–6
Footnotes 3–7

Chapter 4 — Description of Software

Radius ImpressIt 4–1
Adobe Photoshop 4–6
Footnotes 4–10

Chapter 5 — Objective Tests

Compression Factors 5–3
Image Frequency 5–6
Resolution 5–8
File Format 5–14
Sub-Sampling Ratios 5–15
Compression Cycles 5–18
Footnotes 5–23

Chapter 6 — Future Trends

PostScript Level 2 6–1
User Transparency 6–2
Additional User Controls 6–3
AC Prediction 6–4
Custom Quantization Tables 6–5
CIE Color Spaces 6–6
Footnotes 6–8

Bibliography

Appendix A — Original Images

Appendix B — Compression Comparisons

Appendix C — Sub-Sampling Comparisons

Appendix D — Data Tables

Appendix E — Image Size & Resolution vs. File Size

Appendix F — Equipment

Appendix G — Glossary

CHAPTER 1 — INTRODUCTION

The use of 24-bit color images in work produced on desktop computers is becoming increasingly more common. A quick look through any of the numerous magazines available on desktop publishing will quickly confirm that color is the "hot" topic of the day. The complexities involved with producing this color output range far and wide, however one of the biggest problems facing users today is the issue of storage space and file transmission (see Appendix E — File Size vs. Resolution). A brief glance at this table makes it immediately clear that even a small color image can generate a huge file. The problem with these files is that they occupy unacceptable amounts of expensive disk space while also bringing file transmission over phone lines to a virtual stand-still. According to Bruce Fraser of Publish magazine, ". . . modem transfer times are simply prohibitive: A 14-MB image would take more than five hours to transmit, even at 9,600 bps; at 2,400 bps, it would take close to a day"[1].

One of the first solutions which comes to mind for solving this problem is the use of some type of data compression technique. There are numerous compression schemes available which are capable of reducing file sizes, however the nature of continuous tone images prevent standard data compression schemes from achieving high compression ratios. Furthermore, the fact that continuous tone images can tolerate some degree of degradation during compression (known as "lossy" compression) without any visible difference leads to a need for a specialized compression technique.

To address this need, a committee known as the Joint Photographic Experts Group began work in the late 80's on a compression scheme specifically tailored to the needs of continuous tone images. The compression scheme which resulted from this work (JPEG compression) has become the de facto industry standard in the graphic arts world.

Although JPEG compression has been in existence in one form or another for over five years now, there is still a considerable lack of information available concerning its operation and proper use. Part of this problem results from the fact that the JPEG standard is not intended to be all-

encompassing (see Chapter 2 — Theory & Background, for a more thorough discussion). A far greater part of the mystery surrounding JPEG compression results from the conflicting information available in periodical literature. As an example of some of the misleading information being published, the author has assembled the following quotes concerning acceptable compression ratios.

- "For applications where accuracy is vital, a compression rate of 10 to 1 or 15 to 1 is recommended."[2]
- "Incredibly, even at compression ratios of 20:1, compressed images appear visually identical to the originals. Differences are noticeable only on close inspection."[3]
- "As a rule of thumb, for good-quality reproduction, you want to keep the compression under 20:1, though some images are hardly disturbed by even 30:1 or more."[4]
- "This method [JPEG] of compressing a very high-quality, high-resolution image is most advantageous when compressing an image at a ratio of 30-to-1. . . ."[5]
- "However, research shows that little quality is lost at a relatively high ratio such as 40:1, and that the loss is balanced against the saving of file space and the speed of operations."[6]

In addition to the confusing array of material concerning acceptable compression ratios, information regarding differences between input compression and absolute compression, time considerations, and other factors is either completely ignored or misleading. Sample images are usually limited to one or two selections and even these are only shown at the extreme ends of the compression scale. Searching through hard cover reference materials provides only detailed mathematical descriptions which are incomprehensible to an average user.

The purpose of this project is to serve as a reference work that can be utilized by users who are familiar with the basics of desktop color production and wish to effectively implement JPEG

compression. It includes all the basic background necessary for a new user to understand how JPEG compression works, as well as practical advice for effective implementation. The reference is divided into six chapters which cover different aspects of JPEG compression in varying degrees of detail. All readers should carefully study Chapters 3, 5, and 6 in order to understand the most important aspects of JPEG compression. Chapter 2 — Theory and Background is not necessary reading for an average user but is included for the sake of completeness and for those interested in the specifics of JPEG algorithms. Chapter 4 — Description of Software is also unnecessary for every user, however a quick look through this section will help those less familiar with JPEG compression understand the tests which are discussed in Chapter 5.

Footnotes

[1]Fraser, Bruce. Scan Handlers: *12 JPEG compression products that help you get a grip on large images.* Publish, April 1992. Page 56.

[2]Karney, James. *Calibration and compression for 24-bit color.* PC Magazine, January 1992. Page 274.

[3]Parascandolo, Salvatore. *ColorSqueeze, ImpressIt, and PicturePress.* MacUser, September 1991. Page 68.

[4]Dyson, Peter. *JPEG standard compression for still images.* Digital Media, January 1992. Page 23.

[5]Jarrett, Lynn. *Hot topic: compression of still images.* Digital Review, September 1991. Page 38.

[6]Schram, Laura. *Data compression: the key to storing images.* Presstime, March 1992. Page 52.

CHAPTER 2 — THEORY & BACKGROUND

The purpose of this section is to explain the basic fundamentals underlying JPEG compression in the most straight-forward terms possible. This information is included solely for the purpose of understanding the information and test results presented in the following chapters. Emphasis is not placed on the mathematics involved in the JPEG algorithms, nor on the exact ISO specifications which outline JPEG operation. Aside from the fact that this information is likely to confuse anyone without considerable mathematical background, there are already numerous books in existence which describe this process in great detail.

Background

One of the first things which must be understood is that JPEG compression is not one set standard that operates the same regardless of implementation. Rather than thinking of JPEG as a fixed tool which performs a certain operation, it should be considered a set of tools which can be used in different ways to achieve desired end results.

> JPEG is more than an algorithm for compressing images. Rather, it is an architecture for a set of image compression functions. It contains a rich set of capabilities that make it suitable for a wide range of applications involving image compression.[1]

Additionally, the JPEG standard does not completely spell out all of the functions necessary for software-independent decompression of the images.

> . . . JPEG is not a complete architecture for image exchange. The JPEG data streams are defined only in terms of what a JPEG decoder needs to decompress the data stream. Major elements are lacking that are needed to define the meaning and format of the resulting image.[2]

Finally, one must understand that even inside the boundaries that do exist for JPEG compression, there are numerous possible variations within each processing step. In this respect, not only is the combination of tools used variable, the tools themselves are also variable.

With all of these possible variations available, one may begin to wonder how any type of conformity can ever be achieved. This is indeed a valid question, and is addressed in the section on decompression in this chapter as well as the section on compatibility in Chapter 3 — Compression Considerations. The description of JPEG compression which appears in the following pages is a general outline of the steps involved in "lossy" JPEG compression. It is by no means intended to represent every possible variation of the JPEG algorithm, nor is it intended to cover the lossless modes of JPEG compression.

Image Blocking

JPEG compression works by breaking the image down into blocks which are eight pixels by eight pixels (see Figure 2-1). This eight-by-eight sampling rate remains constant regardless of image resolution or dimensions. Each of these blocks are independently processed in three distinct stages (see Figure 2-2). In order to understand how this process works, an eight-by-eight block from a typical grey-scale image will be processed through the entire JPEG algorithm.

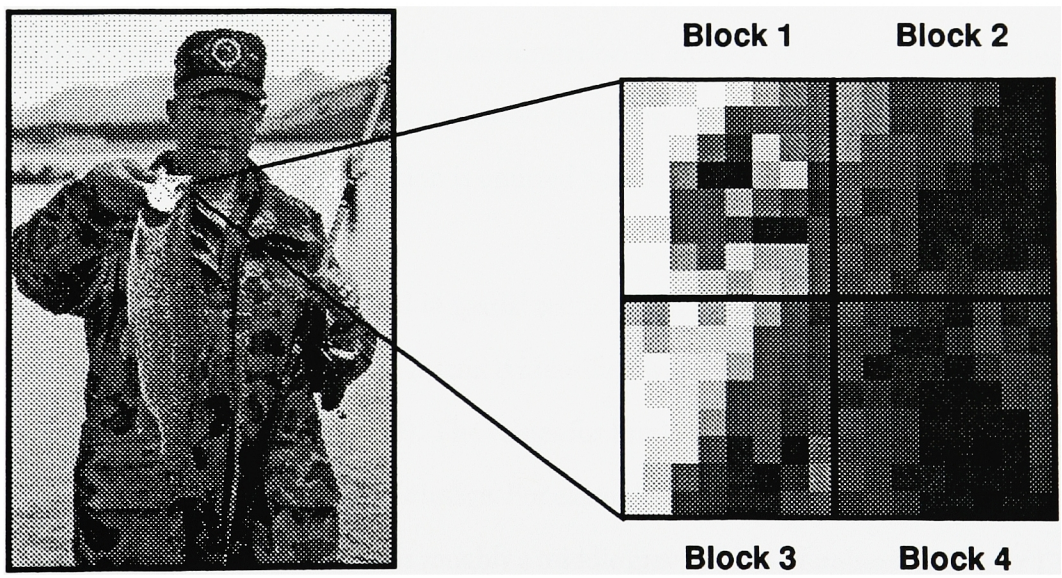


Figure 2–1

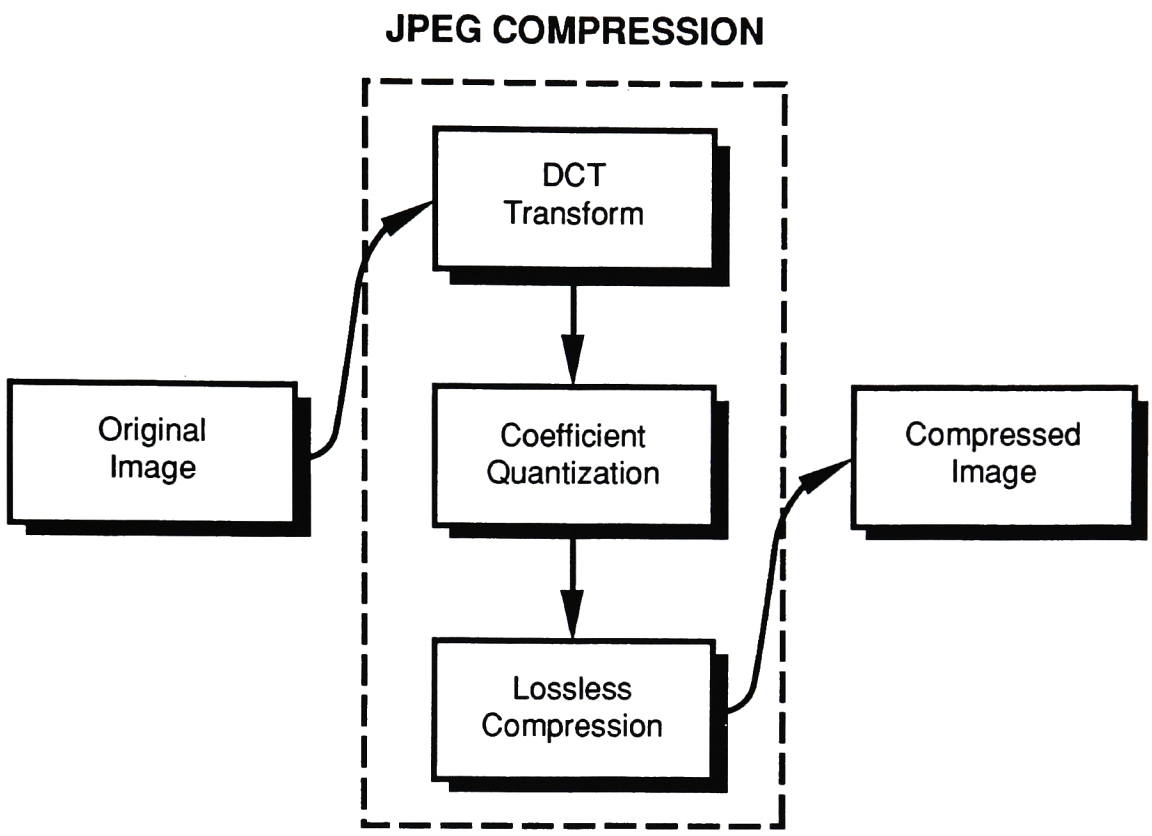


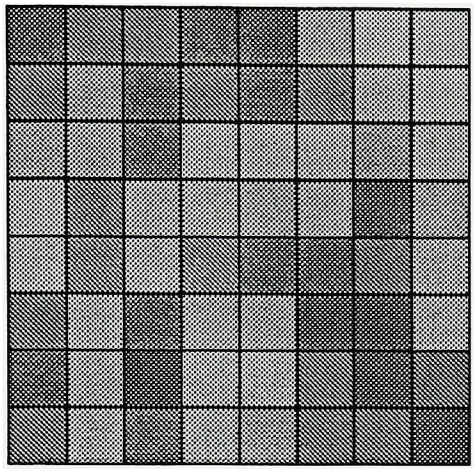
Figure 2–2

Discrete Cosine Transform

The first step in JPEG compression is the transformation of image data from the spatial domain to the frequency domain using a discrete cosine transform. While this process sounds rather intimidating (a mathematical description is omitted here for simplicity's sake) the end result is not very hard to comprehend.

Originally the image is described in spatial terms which can most accurately be described as follows: each and every point in the matrix is identified with an X coordinate, a Y coordinate and a luminosity value (see Figure 2-3). The values for luminosity range from 0 (black) to 255 (white) assuming an eight bit grey-scale image. For example the pixel which is seven over and six down from the upper left hand corner is roughly a middle grey and has a luminosity value of 136. Spatial representation of an image is the only method which appears visually correct to the human

eye, however it does not lend itself well to compression because of the random distribution of luminosity values.



X Values →

Y Values ↓

140	144	147	140	140	155	179	175
144	152	140	147	140	148	167	179
152	155	136	167	163	162	152	172
168	145	156	160	142	155	136	160
162	148	156	148	150	136	147	162
147	167	140	155	155	140	136	162
136	156	123	167	162	144	140	147
148	155	136	155	152	147	147	136

Luminosity Values

Figure 2–3

The purpose of the discrete cosine transform is to take the spatial information in the matrix and convert it into frequency information. The basic idea behind this transformation is that the most important data (*i.e.* the most frequent data) is represented by larger numbers placed in the upper left hand area of the matrix (see Figure 2-4). The number in the upper left hand corner is known as the DC coefficient and represents an average of the overall magnitude of the input matrix. The remaining 63 numbers in the matrix are known as AC coefficients. The AC coefficients closest to the DC coefficient represent the next most important information in the image, with relative importance (and absolute numerical value) continually decreasing as the numbers become farther away from the upper left hand corner. The end result of the discrete cosine transform is the concentration of important data into a compact area. This allows data

in the less important areas to be disregarded, which in turn allows greater levels of compression in the final stages.

Input Pixel Matrix

140	144	147	140	140	155	179	175
144	152	140	147	140	148	167	179
152	155	136	167	163	162	152	172
168	145	156	160	142	155	136	160
162	148	156	148	150	136	147	162
147	167	140	155	155	140	136	162
136	156	123	167	162	144	140	147
148	155	136	155	152	147	147	136

DCT Output Matrix

186	-18	15	-9	23	-9	-14	19
21	-34	26	-9	-11	11	14	7
-10	-24	-2	6	-18	3	-20	-1
-8	-5	14	-15	-8	-3	-3	8
-3	10	8	1	-11	18	18	15
4	-2	-18	8	8	-4	1	-7
9	1	-3	4	-1	-7	-1	-2
0	-8	-2	2	1	4	-6	0

Figure 2–4

Coefficient Quantization

Once the data has been converted into frequency information by the discrete cosine transform, the next step is to quantize the information. "Quantization is simply the process of reducing the number of bits needed to store an integer by reducing the precision of the integer"[3]. Quantization is carried out by dividing each number in the DCT matrix by a corresponding number in the quantization matrix and then rounding to the nearest integer. In other words, the value in the upper left hand corner of the DCT matrix is divided by the value in the upper left hand corner of the quantization matrix. This process continues on every corresponding set of cells until a new matrix, which will be referred to here as the post-quantized matrix, has been produced.

To illustrate this process a fictitious quantization matrix has been created and applied to the DCT matrix developed previously (see Figure 2-5). Several things are important to note concerning this process.

DCT Output Matrix

186	-18	15	-9	23	-9	-14	19
21	-34	26	-9	-11	11	14	7
-10	-24	-2	6	-18	3	-20	-1
-8	-5	14	-15	-8	-3	-3	8
-3	10	8	1	-11	18	18	15
4	-2	-18	8	8	-4	1	-7
9	1	-3	4	-1	-7	-1	-2
0	-8	-2	2	1	4	-6	0

Quantization Matrix

2	5	7	11	13	16	19	22
5	7	11	13	16	19	22	25
7	11	13	16	19	22	25	27
11	13	16	19	22	25	27	30
13	16	19	22	25	27	30	34
16	19	22	25	27	30	34	36
19	22	25	27	30	34	36	39
22	25	27	30	34	36	39	41

Post Quantization Matrix

93	-4	2	-1	2	-1	-1	1
4	-5	2	-1	-1	1	1	0
-1	-2	0	0	-1	0	-1	0
-1	0	1	-1	0	0	0	0
0	1	0	0	0	1	1	0
0	0	-1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Figure 2–5

First, the values in the quantization matrix start with very low numbers in the upper left hand corner and become increasingly large as they move away from this location. This reduces the accuracy of the numbers in the lower right hand corner, which were shown to be the least important in the previous section, and converts many of these values to zero. The significance of this will become more clear in the section on lossless compression, but suffice it to say here that the more zeros there are, the greater the amount of compression possible.

The second important fact to realize is that the values for the quantization matrix can be altered to allow for various levels of accuracy.

> The quantization values can be set individually for each DCT coefficient [all DC and AC coefficients], based on the visibility of the basis functions. If we measure the threshold for visibility of a given basis function—the coefficient amplitude that is just detectable by the human eye—we can divide (quantize) the coefficients by that value. . . . If we are willing to tolerate some visible artifacts in the reconstructed image, we might divide by a value larger than the visibility threshold.[4]

As larger values are used in the quantization matrix, the number of resulting zeros in the post-quantization matrix steadily increases. Although this ultimately results in higher levels of compression, it also leads to a greater degree of image degradation. This feature is what allows the end user of a JPEG product to select different levels of compression depending on their needs. It also allows developers of JPEG products to produce custom quantization tables that fit specific needs (see the section on Custom Quantization Tables in Chapter 6 — Future Trends for a more thorough discussion).

The final important fact to realize about quantization is that it is the only stage of JPEG compression where significant amounts of information are actually discarded.

> The FDCT [forward discrete cosine transform], quantization, dequantization, and IDCT [inverse discrete cosine transform] are the cause of the distortion in the images reconstructed by a JPEG lossy decoder. Arithmetic approximations in the integer arithmetic typically used in computing the FDCT and IDCT introduce a small amount of this distortion. The principal source of loss or distortion, however, is the quantization and dequantization of the coefficients.[5]

In other words, quantization is the major "lossy" component of JPEG compression. The other two stages, DCT transformation and lossless compression, merely reorganize the information into a different form.

Lossless Compression

The final stage in the JPEG algorithm is compression of the information in the post-quantized matrix into a more efficient form. This is carried out using three distinct steps which will be called coefficient transform, "zig-zag" arrangement, and data encoding. As with the discrete cosine transform, the mathematics involved are quite complex and of little concern to the end user. However, a basic understanding of the process will help explain how JPEG achieves such high levels of compression.

Coefficient transform involves changing the DC coefficient (the number representing the average overall magnitude of the input matrix) from an absolute value to a relative value. This is accomplished by subtracting the DC coefficient in a particular eight-by-eight block (block 2) from the coefficient of the preceding block (block 1). This value is then recorded as the new DC coefficient for block 2 (see Figure 2-6). Since adjacent blocks are very likely to be similar in a typical continuous tone photograph, the values of most DC coefficients are subsequently converted to relatively small numbers.

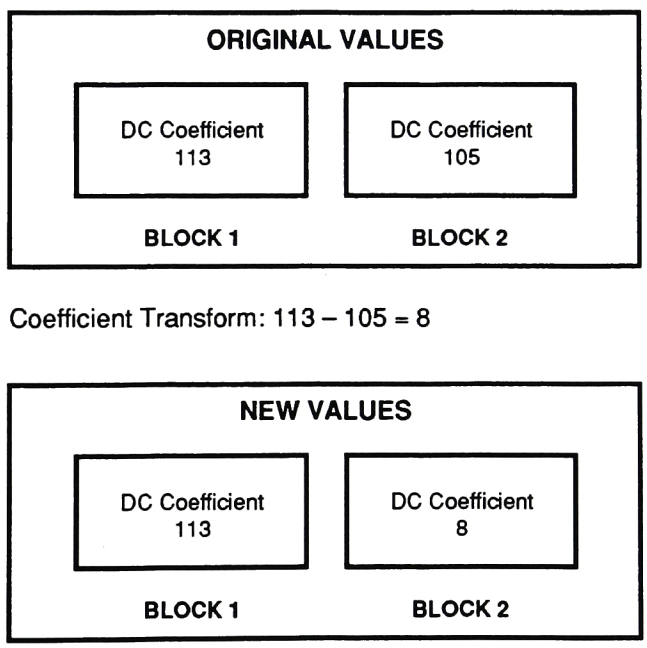


Figure 2–6

Once the DC coefficients in the post-quantization matrix have been altered using the coefficient transform, the AC coefficients are then re-arranged into a "zig-zag" sequence (see Figure 2-7). This sequence takes advantage of the fact that values in the matrix are more likely to be zero the farther away from the upper left hand corner they are located. By arranging the values so that large strings of zeros follow each other consecutively, the data can be efficiently compressed using a run-length coding scheme.

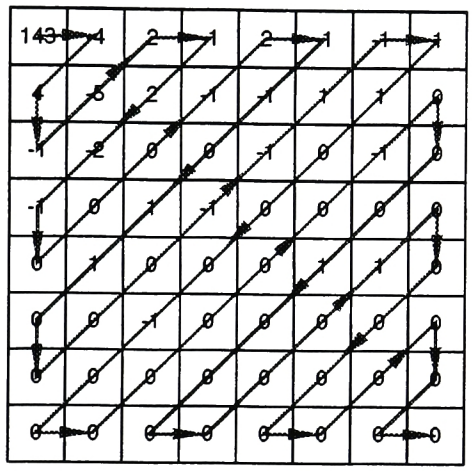


Figure 2–7

Run-length encoding is used to compress all of the zeros that occur in the final matrix. To understand how run-length encoding compresses the amount data required to represent an image, consider Figure 2-8 which shows a table containing sixteen zeros. Normally a computer would describe this information by recording values for each separate cell on the table. For example cell A1 would have a value of zero, cell A2 would have a value of zero, and so on all the way through cell B8. While this method is the only viable alternative for describing data that varies widely from cell to cell, it is not very efficient for data that follows a regular pattern. A far more efficient method is a run-length encoding scheme which counts the number of consecutive values and stores the information as a string. Using this method, the data in Figure 2-8 would be described by saying, "all numbers that fall between cell A1 and B8 have a value of zero". The

computer does not store data in the form of intelligible text strings such as the one above, however, it is easy to see that this general method of representing data takes up far less storage space than describing each cell individually.

	1	2	3	4	5	6	7	8
A	0	0	0	0	0	0	0	0
B	0	0	0	0	0	0	0	0

Figure 2–8

The final step in JPEG compression is known as entropy encoding and involves a combination of Huffman (run-length) encoding and arithmetic coding. Arithmetic coding statistically determines which numbers are most likely to occur and assigns these values lower numbers. Since these lower numbers require fewer bits to represent, the computer is able to record these values using less storage space.

Decompression

JPEG compression would be of little use if there wasn't some way to decompress the image at a later time. Luckily, the mathematical formulas and coding schemes used to compress the image can easily be reversed to decompress the image. The unfortunate part is that not all JPEG products work in exactly the same manner. This means that an image compressed with company ABC's JPEG algorithm may not necessarily decompress using company XYZ's JPEG algorithm.

Since the JPEG standard does not contain exact specifications for much of the compression/decompression process, the committee which developed the JPEG standard made provisions for including additional information with compressed files. This information, if included and properly identified, will allow files compressed with any JPEG-compliant algorithm to be decompressed by virtually any other JPEG-compliant algorithm.

The data contained in a JPEG-compressed file is made up of entropy-coded segments and marker segments.

> Entropy-coded segments contain the entropy-coded data [the actual image information], whereas marker segments contain header information, tables, and other information required to interpret and decode the compressed image data. Marker segments always begin with a "marker", a unique two-byte code that identifies the function of the segment.[6]

Marker codes for all of the allowable functions are clearly established in the ISO standards for JPEG compression (ISO DIS 10918-1) to insure that all manufacturers use the codes consistently. Without 100% consistent use of these codes, files compressed using one manufacturer's product have no hope of being decompressed using another manufacturer's product.

To support applications which require different levels of compatibility, the JPEG standard allows three different file formats for compressed data: interchange format for compressed image data, abbreviated format for compressed image data, and abbreviated format for table specification data. Brief definitions of these three formats are outlined in the list below.

1) Interchange Format: Contains all the information necessary to decompress an image regardless of the application. Most useful in applications where the image needs to be used in a variety of situations.
2) Abbreviated Format for Compressed Image Data: Omits some or all of the tables required to decompress an image. This format is most useful when the destination of the compressed image files is known and space saving requirements are critical.
3) Abbreviated Format for Table Specification Data: Defines the tables that are required in the abbreviated format for compressed image data.

The precise definitions for each of the above formats are outlined in sections 3.9.1, 3.9.2, and 3.9.3 of the JPEG specifications (ISO DIS 10918-1).

In addition to the information which is covered under the ISO specifications, many applications also include additional information in the compressed data to aid in decompression.

> Certain parameters that are needed by many applications are not part of the JPEG compressed data format and may therefore be needed in application-specific "wrappers" surrounding the JPEG data. Among these parameters are image aspect ratio, pixel shape, meaning of the image components, orientation of the image (vertical, horizontal, upside down, etc.) and the sample registration between the components.[7]

It is important to realize that the information in these "wrappers" is only recognizable by the application that created the compressed file and is thus useless in situations where that particular application is inaccessible.

Color

Methods described thus far have only addressed compression of grey-scale images. This was an intentional omission on the author's part in an effort to keep explanations as simple as possible. However, the real value of any image compression scheme is the ability to compress full-color images. Since a 24-bit RGB image with the same spatial resolution and physical dimensions of an equivalent 8-bit grey-scale image requires three times the storage space to describe, the need for compression becomes even greater. In order to accommodate this situation, there needs to be a method of applying the principles outlined above to color images.

Intrinsically, the most straight forward method is to break the image down into its individual red, green and blue channels and then treat each one as if it were a single grey-scale channel of varying luminosity. Once the image is represented in this form, the process which occurs during compression is exactly the same as the method outlined above. While this transformation makes the most sense at first glance, there is an even better method that allows far greater levels of compression.

Instead of separating into red, green and blue channels, the image is converted into the YUV color space.

> YUV provides a closer model for how the human eye perceives visual information. Chrominance has two components, U and V, which define the color of a given pixel. In most cases, the human eye is unable to detect the difference between the original image and an image that has some UV components deleted. On the other hand, the luminance component, which is Y, defines the relative intensity of a given pixel. The eye is much more sensitive to the Y component than the U and V components.[8]

This means that the chrominance channels (U and V) can be compressed using much larger quantization values, and in some cases can be completely ignored, since the resulting image degradation is difficult for the human eye to perceive. On the other hand, the more prominent luminance channel (Y), is compressed using smaller quantization values. The overall effect is that the visually important areas are compressed less while the unimportant areas are compressed more. Since the less important chrominance channels occupy two-thirds of the required storage space, the total amount of data compression available using this model increases substantially. For a more detailed look at this topic, see the section on Sub-Sampling Ratios in Chapter 5 — Objective Tests.

Footnotes

[1]Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 6.

[2]Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 6.

[3]Nelson, Mark. *The Data Compression Book.* M&T Publishing, 1991. Page 364.

[4]Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 35.

[5]Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 71.

[6]Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 98.

[7]Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 97.

[8]*Radius ImpressIt™ User's Manual.* Radius Inc., 1991. Page 31.

CHAPTER 3 — COMPRESSION CONSIDERATIONS

To effectively utilize JPEG compression in any work environment, one must consider numerous variables involved in the process. There is no "right answer" concerning acceptable compression ratios, acceptable speed rates, etc. — it all depends on the unique requirements of each individual or work place. To help determine the optimal settings in each particular environment, the following sections are included. Careful consideration of the issues discussed here will help make for a smooth and effective implementation of JPEG compression.

Image Resolution

One of the least talked about issues concerning JPEG compression is the resolution of the original image. It is rarely, if ever, mentioned in periodical literature that higher resolution images will tolerate greater absolute compression ratios than images scanned at lower resolutions. The reason behind this has to do with the way the JPEG algorithm works and is fully explained in the resolution section of Chapter 5 — Objective Tests. The important realization to make here is that different sets of guidelines need to be established for images of different resolutions.

Assume, for example, that subjective tests have determined a 10:1 absolute compression ratio is acceptable for a certain type of image in a given production environment. If the resolution at which the image is scanned is either increased or decreased, the absolute compression ratio which produces an equivalent result will also increase or decrease. If steps are not taken to compensate for this fact, the result will be wasted storage space at best and unacceptable image degradation at worst.

Image Frequency

Another one of the reasons it is impossible to come up with an overall acceptable compression ratio has to do with the variety of original images. Certain images which are low frequency (*i.e.* large areas of similar color and luminosity) can be compressed to very high levels with little noticeable image degradation, while high frequency images (*i.e.* numerous transitions between

color values and luminosity levels) will tolerate considerably less compression before image degradation becomes noticeable. This means that different settings and guidelines must be established for different image types in order to consistently produce acceptable results. For a further explanation of these considerations, see the section on image frequency in Chapter 5 — Objective Tests.

Speed Requirements

Depending on the particular situation at hand, compression and decompression speeds may be major considerations. If only one or two images need to be compressed each day, then the difference between a product that takes one minute and a product that takes five minutes is inconsequential. If hundreds of images are being compressed and decompressed each day at a large production facility, however, the difference between these two products is definitely a major concern. There are basically two choices available when choosing a JPEG compression utility; software-only packages and combination hardware/software packages.

The chief advantages of software-only packages are cost and ease of upgrade. A number of different products that fall into this category are commercially available in the $120.00 to $200.00 price range as of this writing. As with any software package, upgrades are typically available on a regular basis and can be easily installed without resorting to the expertise of a trained technician. This allows quick and inexpensive upgrades that keep computers in production and out of the dealer's repair shop.

The disadvantage to software-only packages is that they are relatively slow when compared to hardware/software packages. Since the JPEG algorithm is a computation-intensive process, compression and decompression times can become extremely slow on large files. This problem is particularly evident on older computers with slower CPU's that were not designed for the amount of computation demanded by JPEG compression. In situations where older computers are installed and/or speed is a major concern, hardware/software packages are recommended.

Hardware/software packages solve the speed problem by off-loading the computation to a separate card or chip.

> . . . hardware implementations of JPEG can be fast because the JPEG algorithm allows computations to take place in parallel. Once an image has been divided mathematically into sections, the sections can be independently compressed at the same time. . . . hardware compression beats software compression of the same method by about a factor of ten in most cases.[1]

In addition to hardware/software solutions that are designed exclusively to speed up JPEG compression, many new hardware alternatives (such as the ThunderStorm card by SuperMac) offer increased speed during JPEG compression as well as other computation-intensive tasks. These alternatives help justify the cost of the hardware by increasing throughput in numerous areas which slow down production.

The big disadvantage to hardware/software solutions is the high price tag that accompanies the hardware. Products that fall into this category average anywhere from $500.00 to $1000.00 at the time of this writing, although prices are constantly dropping as competition in the market increases. Another disadvantage of hardware solutions is that installation requires someone who is reasonably competent with computers. Although the installation process is quite simple in most cases, the mere thought of opening the case of a computer is enough to scare many users away. For these individuals, a hardware solution then requires the assistance of a trained technician, which in turn means even higher costs and longer down-time. A final disadvantage to hardware solutions is that there is generally no way to upgrade short of buying new hardware (*i.e.* spending another considerable sum of cash and going through the installation process again).

Features

Although all of the products available for JPEG compression accomplish the same basic result, there is a big difference between the features available in the various offerings. Luckily, this is one

area which the available periodical literature does a good job of covering (the April 1992 issue of Publish has an excellent article covering the various products currently on the market). Since there are already a number of articles published which cover these features, the topic will only be given a brief discussion here.

Basically, the features which are important to look for include the following:

1) Good range of available compression settings in sufficiently fine increments
2) Acceptable speed depending on individual requirements
3) Ability to save self-decompressing files
4) Preferably, though not necessarily, a preview window for viewing anticipated image degradation
5) *Adobe Photoshop* plug-in module if used in locations where *Photoshop* is the main production tool

Depending on the individual user's requirements and schedules, the relative importance of these features will vary. The important thing to keep in mind is that each environment will have different needs which must be fully evaluated before deciding on a particular JPEG product.

Intended Audience and Viewing Conditions

To a large extent, the acceptable image quality attainable with JPEG compression is determined by the intended audience and viewing conditions under which the images will be seen. If the end product is an expensive, high quality art book which will be scrutinized closely, the amount of acceptable image degradation is likely to be minuscule. However, if the end product is a picture of a cow on the side of a milk carton, the amount of acceptable image degradation will be quite a bit greater.

In order to establish some type of common viewing environment which can be used to examine the quality of printed reproductions, certain standards have been set and accepted. An excellent article which covers the majority of these issues is contained in The Quality Control

Scanner, Volume 2, Number 4, ©1981 Graphic Arts Publishing Co., Miles and Donna Southworth. This article describes in detail the accepted ANSI standards for viewing numerous types of originals and prints so that communication between various viewers is based on common ground. It is highly recommended that users adhere to these accepted guidelines whenever judging print quality (including the prints contained in this reference). In cases where it is known that the final product will be viewed under conditions drastically different from those spelled out in the guidelines, analysis should take place in a controlled environment which most closely resembles the intended viewing conditions. If multiple parties are involved in making quality decisions in a case such as this, the viewing environment should be clearly outlined for all persons involved.

Output Devices

For a given level of print quality, the amount of acceptable compression varies depending on the type of output device utilized. Some devices cover up the artifacts of JPEG compression to a certain extent, while others exaggerate them. Unfortunately, it is economically unfeasible to include samples of all the various types of output available in this reference (or any reference for that matter). The important issue to be aware of is that each individual production method must be tested in order to determine acceptable compression ratios. If a certain set of guidelines have been established which work well on a continuous tone proofer, for example, it is highly unlikely that these same settings will produce the optimal results when running on an offset lithography press. Therefore, it is extremely important to establish guidelines concerning JPEG compression using the exact same equipment that will be used during actual production.

Compatibility

Even though all products which employ JPEG compression are based on the same standard, chances exist that a file compressed using one type of software cannot be decompressed using another type of software.

> "JPEG is an algorithm and a structure," explained Xing's Ambrosini. "There are a lot of variables that can be put in there. Sometimes someone says, 'You're JPEG compatible and we're JPEG compatible,' so we should be able to read their files. But we can't. Some people take the JPEG algorithm and do a new file format of their own. They change a couple of things, like headers, and all of the sudden you can't read their files. That's confusing the market right now."[2]

> The incompatibility issue "is a tragedy," said Storm's Krause. "There's nothing that creates [more] negative emotions about a new technology than when it doesn't do what you expect it to. JPEG is a documented, published, specified industry standard. Unfortunately, however, you have people who are trying to short thrift their investment requirements, so they are doing incomplete versions of the JPEG spec. They've created inferior products that aren't fully compliant."[3]

The implications of these incompatibilities depend on the particular situation with which each user is faced. If all images being compressed are for personal use or in-house production, the incompatibility issue is not a problem at all. However, if the compressed images are going to be sent to a number of different locations, the incompatibility issue becomes a serious problem. Currently, the best defense against this problem is the use of a JPEG program which creates self-decompressing files. These files contain extra code which allows them to be decompressed without the use of any additional software. This extra feature usually comes at the expense of a few extra kilobytes of storage space, however the trade-off is well worth it if compatibility problems are foreseen.

Footnotes

[1]Seiter, Charles. *Macworld Lab tests 8 JPEG products.* Macworld, March 1992. Page 150-151.

[2]Strothman, Jim. *Still-Image Compression Vendors Seek to Embed Technology in Applications.* Computer Pictures, March/April 1992. Page S-13.

[3]Strothman, Jim. *Still-Image Compression Vendors Seek to Embed Technology in Applications.* Computer Pictures, March/April 1992. Page S-13 – S-14.

Radius ImpressIt

To better understand the options available in a typical JPEG compression program, as well as test results in Chapter 5 — Objective Tests, the features available in *Radius ImpressIt* are thoroughly covered in this section. Some of the features available within this program are available in other JPEG products, although they may go by different names, while others are unique to *ImpressIt.* It is important to note that the interfaces of different JPEG compression programs are not controlled by any type of standard. The only standardization between programs has to do with the mathematics and coding involved during the compression/decompression process (for a more detailed look at this issue, see the compatibility section of Chapter 3 — Compression Considerations).

Figure 4-1 shows the Preview window available in *Radius ImpressIt.* This window allows the user to input the desired compression characteristics as well as view the expected effects of these settings. The following list explains each of these settings in greater detail.

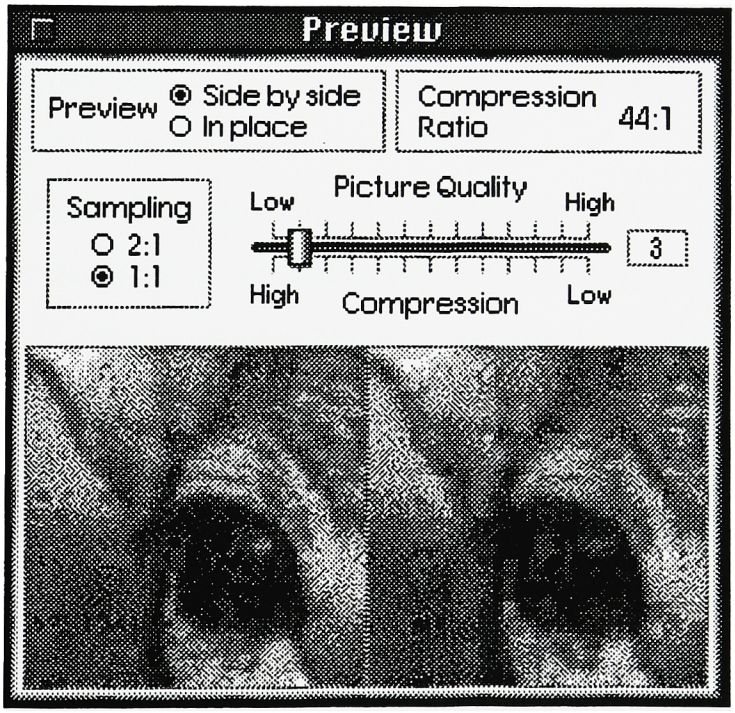


Figure 4–1

Preview: Controls whether the effects of the current settings will be viewed in two windows next to each other (*Side by side*) or in one window directly on top of the actual image (*In place*). Choosing *Side by side* results in an enlarged view of a small section of the original (as shown in Figure 4-1). Choosing *In place* allows the user to drag a small box over the surface of the original image and view the effects of compression in real time (not shown). Incidentally, the preview on an 24-bit color monitor is much more useful than the low resolution example provided here.

Compression Ratio: Predicts the expected absolute compression based on the area being sampled in conjunction with the current settings. This value will change depending on what section of the image is being sampled for the preview window and does not indicate the overall absolute compression that will actually occur.

Sampling: Determines the amount of chrominance information that will be discarded during compression. A setting of 1:1 indicates that no information will be discarded, while a setting of 2:1 indicates that every other row and column of chrominance information will be eliminated (see the section on sub-sampling ratios in Chapter 5 — Objective Tests for a more thorough description of this topic). Technically this term should be called sub-sampling and ratios should be indicated using three numbers (*i.e.* 2:1:1 instead of 2:1).

Picture Quality/Compression: Allows the user to select the amount of compression desired. Values for *ImpressIt* range from 1 (most compression, worst quality) to 25 (least compression, best quality). User can input values via sliding bar or directly type numbers in box at far right.

Figure 4-2 shows the dialog which appears when an image is ready to be saved. The information on the top half of this dialog is standard in any Macintosh application, while the information on the bottom is unique to *ImpressIt*. The following list explains the features which are unique to *ImpressIt*.

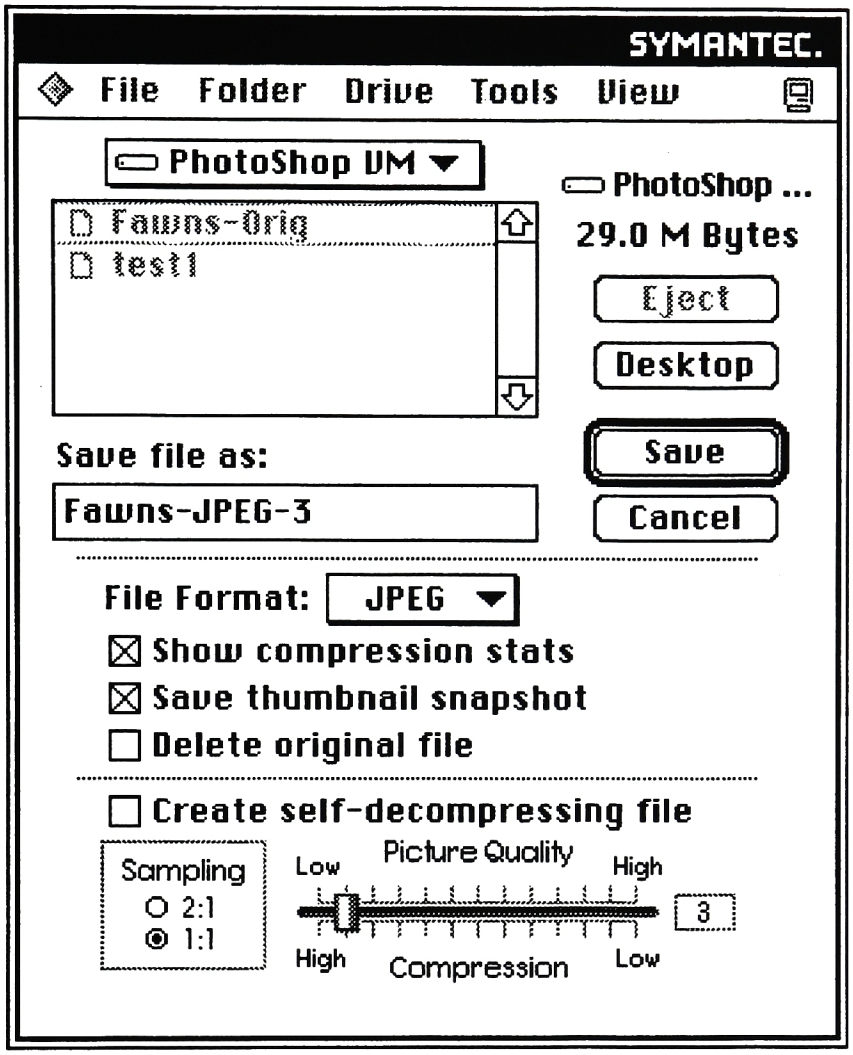


Figure 4–2

File Format: Determines the format used to save the current image. The default value for an original image is JPEG although TIFF and PICT are also available. Only files saved in JPEG format will use the settings supplied by the user to compress the image. Files saved in TIFF format cannot use JPEG compression, although they can be compressed using LZW (Lempel-Ziv-Welch) compression. LZW compression is a lossless process in which the image suffers absolutely no degradation and allows fixed absolute compression ratios in the neighborhood of 1.7:1. PICT format does not presently allow compression of any kind.

Show compression stats: Displays a window containing information about the compressed file after the compression process is completed. See Figure 4-3 for a complete description of this window.

Save thumbnail snapshot: Records a low resolution sketch of the image which can be used to preview images before decompression. This feature is very useful for determining the contents of a compressed file without spending all the time necessary for an actual decompression. According to the *ImpressIt* manual supplied by *Radius*, the inclusion of a thumbnail snapshot adds approximately 6 to 9 Kbytes to the compressed image.

Delete Original File: Automatically deletes the uncompressed file after completing the compression process. This saves the user the hassle of manually deleting the uncompressed file if it is no longer needed after compression.

Create self-decompressing file: Allows the compressed image to be decompressed on a computer which does not have a copy of *ImpressIt* software. This is a very useful feature for anyone who will be sending compressed images out to other users who may not own the same software. According to the *ImpressIt* manual, creating a self-decompressing file adds approximately 20 to 25 Kbytes to the compressed image.

Sampling and Picture Quality/Compression: These are the same features previously described in relation to Figure 4-1. If the settings for these options are altered during this step, the new settings will override the old setting from Figure 4-1.

Figure 4-3 shows the statistics window which appears after compression if the *Show compression stats* dialog was checked in Figure 4-2. This window displays some of the most important characteristics of the compression process and is very useful for collecting test data. A similar window which gives the same information for decompression can also be displayed after a file has been decompressed. The following list describes each of these statistics in greater detail.

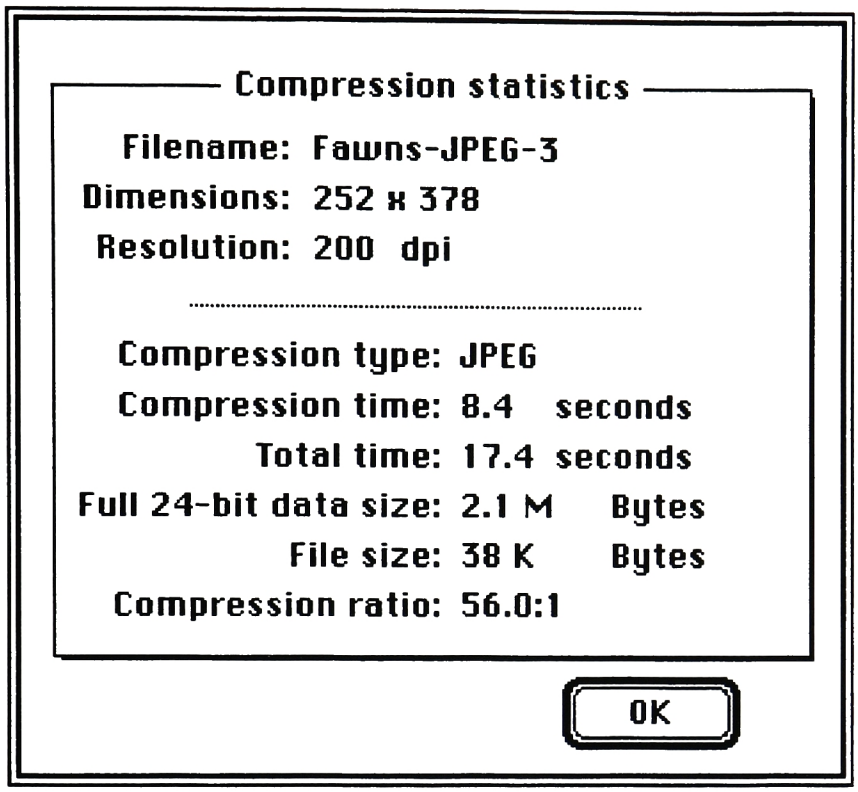


Figure 4–3

Filename: The name of the newly compressed file.

Dimensions: The physical dimensions of the image in screen pixels. Dividing these numbers by the screen resolution (normally 72 dpi) results in dimensions based in inches.

Resolution: Describes the spatial resolution of the original image in dots per inch (dpi).

Compression Types: Identifies the algorithm used during compression (JPEG or LZW).

Compression Time: The time required to compress the image using the current settings.

Total Time: The time required to compress the image plus generate thumbnail snapshots, self-decompressing files, etc.. Total time was not used during any stage of testing in this reference.

Full 24-bit data size: The storage space required by the image in its uncompressed form.

File size: The storage space required by the image after compression.

Compression ratio: The file size of the original image divided by the file size of the compressed image.

Adobe Photoshop

Although *Adobe Photoshop's* main purpose is typically for non-technical applications, it also contains some very powerful analytical tools. Through the use of a series of commands, *Photoshop* allows the collection of objective data concerning JPEG compression. This data, when used in conjunction with sample images, allows for a thorough look at the effects of JPEG compression under various circumstances.

The first step in comparing two images is the use of the *Difference* command available in *Photoshop*. "The *Difference* command subtracts the brightness value of the pixel in the Source 2 channel form the brightness value of the corresponding pixel in the Source 1 channel. A pixel with the resulting absolute brightness value (the absolute value means a negative value is treated as a positive value) is placed in the destination channel".[1] In order to better explain this process, a simple example is presented in Figure 4-4.

In the case of JPEG compression, this command allows the comparison of an original, uncompressed image with one that has been through one or more compression cycles (see Figure

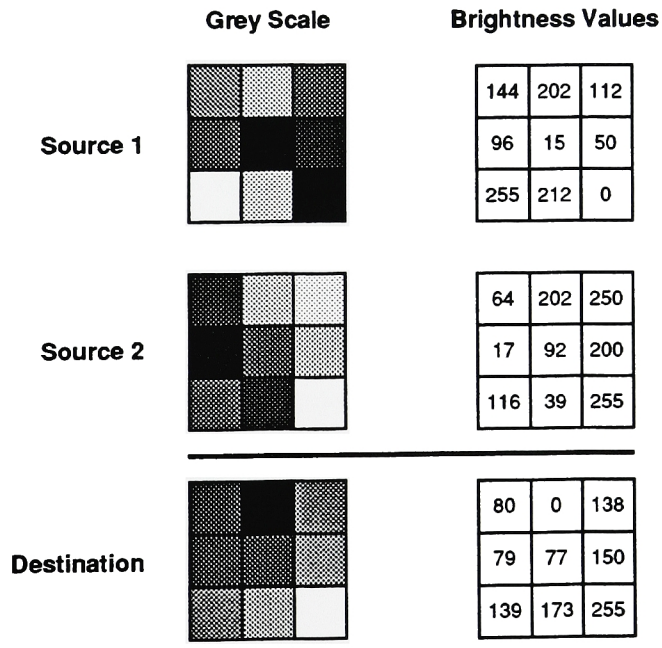


Figure 4–4

4-5). The new file that results from this comparison can then be evaluated using the Histogram command.

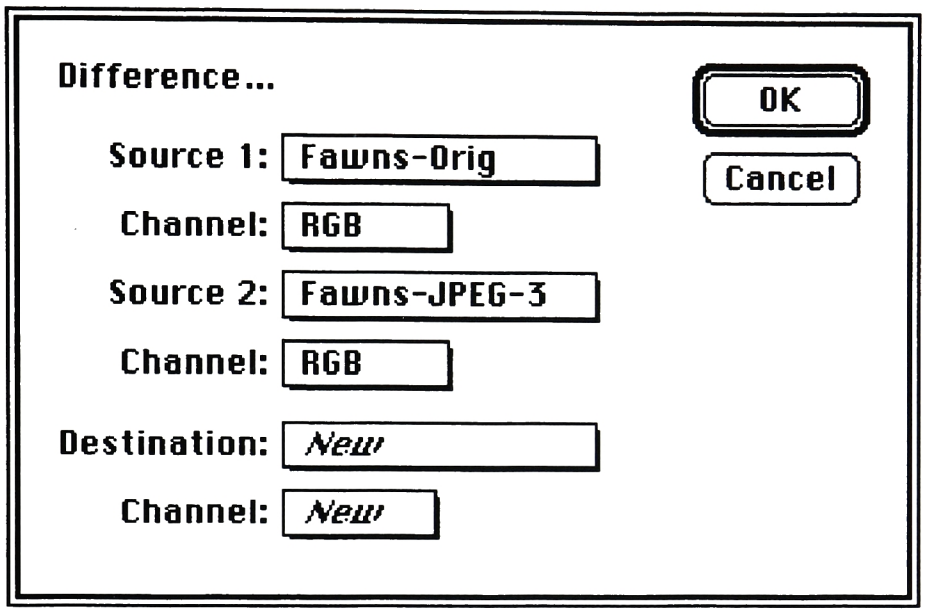


Figure 4–5

The Histogram command generates a graphical representation of the distribution of brightness values in an image (see Figure 4-6).

> The x-axis of the histogram represents the color values from black (0) at the far left to white (255) at the far right; the y-axis represents the total number of pixels with that value.
> The numerical values at the lower left of the Histogram dialog box display statistical information about the color values of the pixels:
> • The Mean is the average brightness value.
> • The standard deviation (Std Dev) represents how widely the values vary.
> • The Median value shows the middle value in the range of color values.
> • The Pixels value represents the total number of pixels in the image or selected area.
> . . . The values at the lower right of the dialog box change to display the color level (Level) of the point, from 0 to 255; the total number of pixels at that level (Count); and the percentage of pixels below that color level (Percentile).[2]

Histogram For A Normal Image

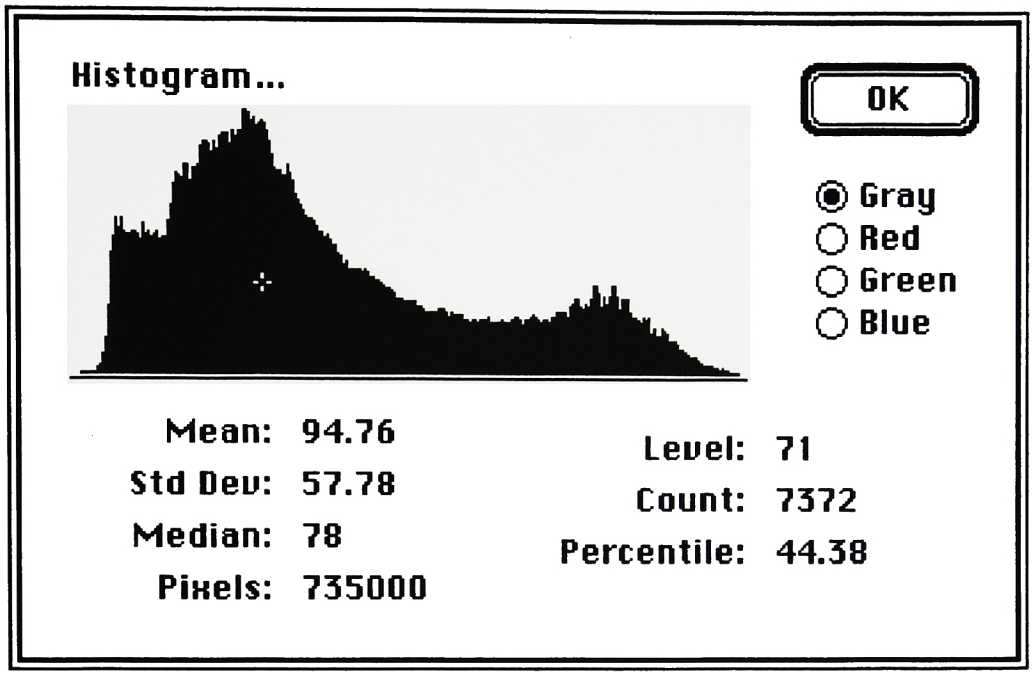


Figure 4–6

Employing this feature on the file generated using the *Difference* command, it becomes very easy to judge the effects of various JPEG settings. If a small degree of degradation is occurring, the majority of the bars in the histogram will be stacked towards the left side of the graph. Additionally the mean, median and standard deviation values will all remain quite small. As the degree of image degradation rises, the bars in the histogram will begin to spread to the right and the values for mean, median and standard deviation will increase. Figures 4-7 and 4–8 show histograms for the image "Fawns" when processed using two different compression factors to better illustrate this fact.

Histogram of Difference File at Compression Factor 24

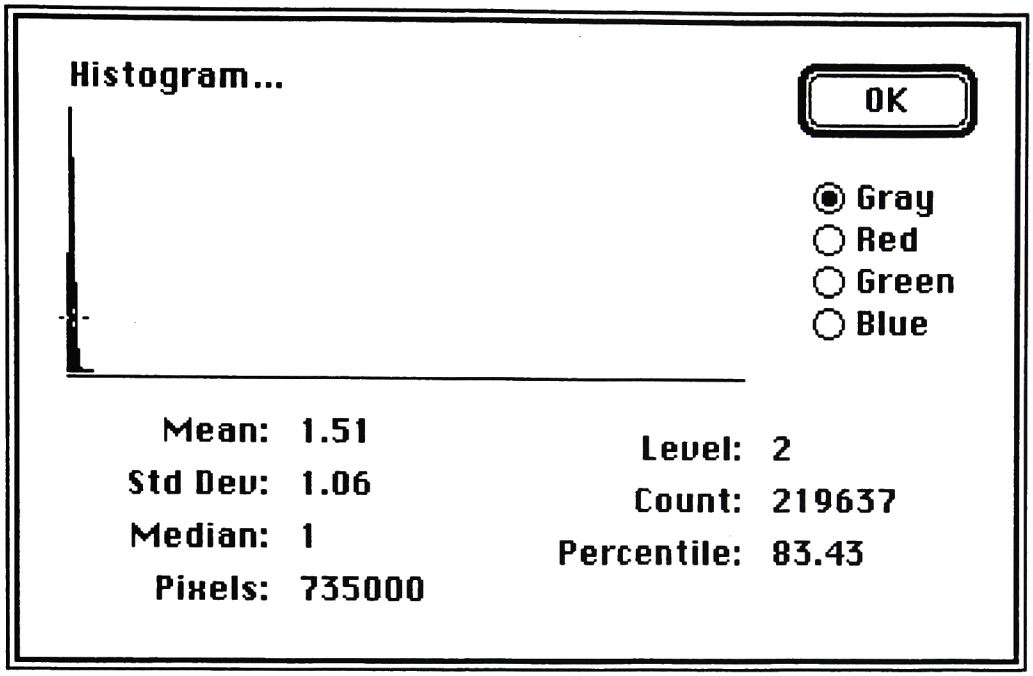


Figure 4–7

Histogram of Difference File at Compression Factor 3

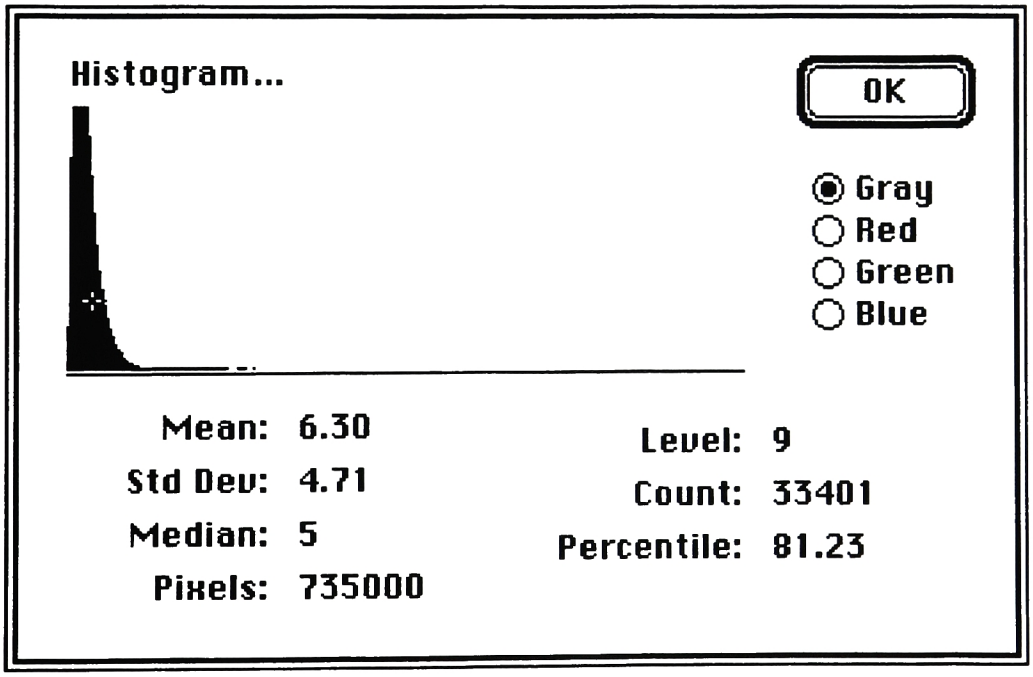


Figure 4–8

Footnotes

[1] *Adobe Photoshop® User's Guide.* Adobe Systems Incorporated, 1991. Page 149.

[2] *Adobe Photoshop® User's Guide.* Adobe Systems Incorporated, 1991. Page 155.

CHAPTER 5 — OBJECTIVE TESTS

In order to gain a better understanding of certain aspects of JPEG compression, numerous objective tests were conducted. The primary objective of these tests was to cover important topics which have been disregarded in the periodical literature. Each test has been conducted with a very specific purpose in mind and attempts to answer one or more of the major questions a typical user might have concerning JPEG compression.

Seven images were carefully selected and scanned for use in the objective testing (see Appendix A — Original Images). These images were chosen so that they spanned a large range of frequency levels and contained varying degrees of detail. By utilizing a broad cross section such as this, the various parameters of JPEG compression could be more fully explored. Traditional test targets and test images were intentionally avoided in this study because of the nature of JPEG compression. Since the JPEG algorithm was specifically designed to take advantage of defects in human sight, objective data from traditional test targets would likely result in misleading information.

All compression tests contained in the following sections were conducted using *Radius ImpressIt* software running on a Macintosh Quadra 700 (see Appendix F — Equipment, for a full list of equipment specifications). The hardware configuration was kept consistent throughout testing in order to eliminate unwanted sources of variability. *ImpressIt* software was chosen from the field of competing JPEG products because it offered a wide range of features which could be tested. Although the author has not conducted any research to support the fact, it is presumed that the results obtained with this software are fairly similar to those of other JPEG products on the market (with the likely exception of absolute time data).

The goal in this reference is to present data in a way which can be applied in any situation, not just one particular hardware and software configuration. For example, assume that tests show compression at level X takes three seconds and compression at level Y takes six seconds. The important fact to remember is not the absolute time, for this will vary from computer to computer

and software package to software package, but instead the relative time. Therefore, the relevant information from the test above is that compression at level X is twice as fast as compression at level Y. This information should more or less hold true no matter which type of software or hardware is being used.

Tests were performed using a variety of settings which compared variables including input compression factor, image frequency, image resolution, file format, sub-sampling ratio, and number of compression cycles. The default values for each of these settings are as follows unless specifically noted otherwise:

Resolution:	200 dpi
Dimensions:	3.5" x 5.25"
File Format:	TIFF
Sub-Sampling Ratio:	1:1:1
Compression Cycles:	1
Time:	Compression Time + Decompression Time (seconds)

The number of different images used to conduct each test varied from one to seven depending on the application and is clearly noted in the text for each test. The actual data collected from each test is recorded in Appendix D — Data Tables.

Graphs are included where necessary to show trends in the data which are difficult to perceive otherwise. All graphs are formatted in a similar manner and viewed from the same perspective in order to maintain consistency. One unusual characteristic of the three dimensional bar graphs is that the y-axis containing compression factors runs in opposite directions on every other graph. This non-standard formatting was necessary because of the nature of the data being plotted. If this formatting change had not been used, data in the first row of every other graph would have obscured all remaining data bars, rendering the graph useless.

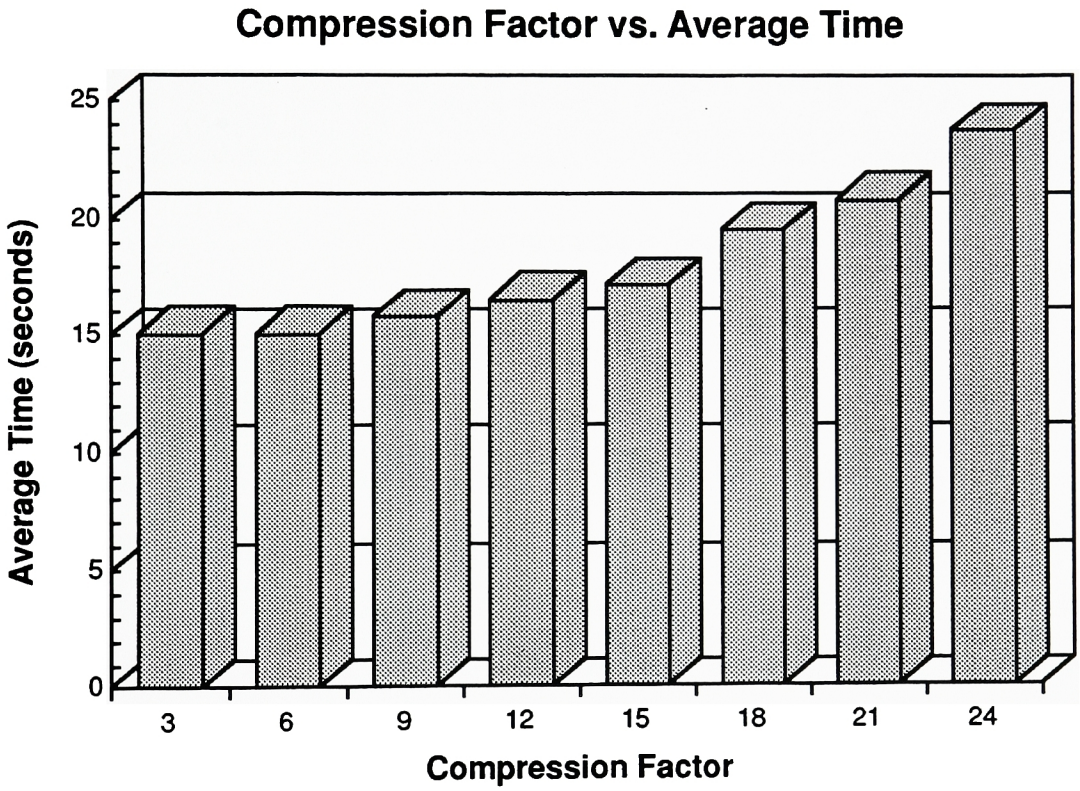
Compression Factors

One of the main advantages of the JPEG algorithm is the flexibility which allows users to choose among numerous levels of data compression via some type of input dialog. Each software manufacturer has implemented this feature using slightly different terminology, which has in turn resulted in a great deal of confusion for the end user. This author has personally seen at least half a dozen terms used for this dialog including picture quality, quality level, compression factor and numerous others. These input dialogs are typically adjusted using a sliding scale which is marked with different numbers and/or terms such as fair, good, excellent, etc. (see Chapter 4 — Description of Software, for an example of how this feature is implemented in *Radius ImpressIt*).

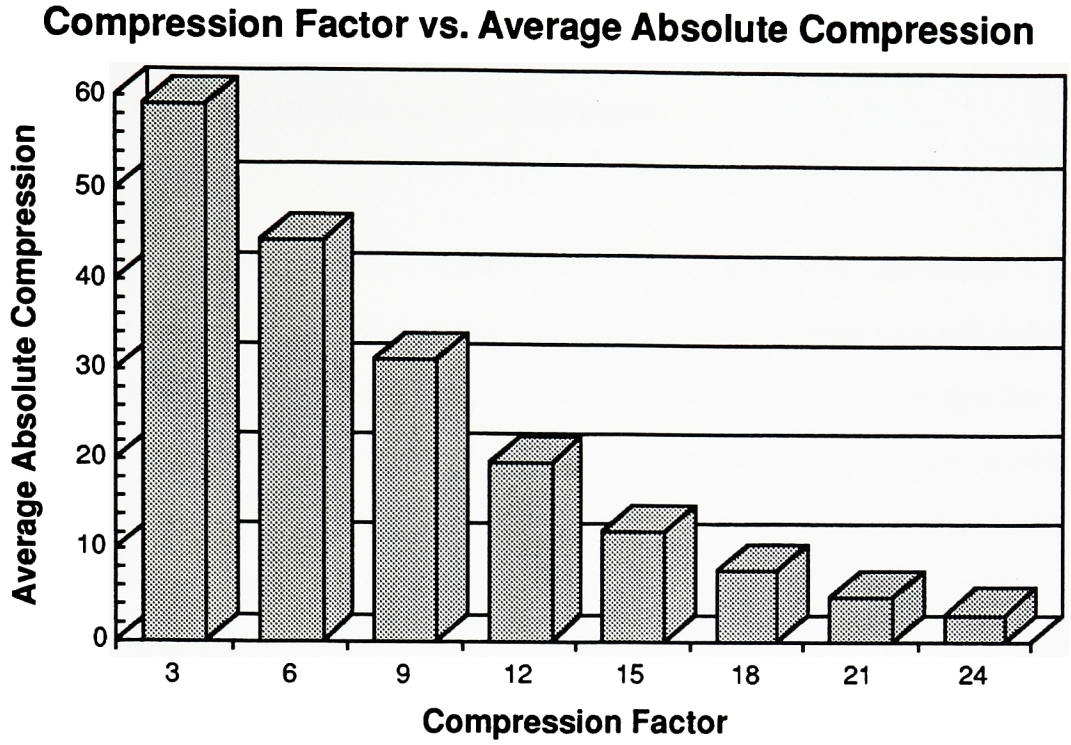
An important fact to realize about these terms is that they are not transferable from one type of software to another. For example the "good" setting on company X's software may result in an absolute compression ratio which is completely different than company Y's "good" setting. Furthermore, the numerical values which are assigned to these settings do not indicate the amount of absolute compression that will be achieved, although some software (including *ImpressIt*) gives the user an idea of how much absolute compression to expect.

The main distinction to be aware of in this area is the difference between the value entered into the dialog box, which will be referred to here as the compression factor, and the resulting absolute compression. Absolute compression is calculated by dividing the original file size by the compressed file size and expressing the result as a number. For example if the original file size is 1000K and the compressed file size is 100K, then the absolute compression is expressed as 10 (or alternatively 10:1). Absolute compression can be calculated on the same basis regardless of the specific software used and therefore stands as the most appropriate method for relating test results. All tests contained in this reference are specifically based on absolute compression for this reason.

Tests run in this series were designed to show the variation in time requirements and absolute compression using the range of compression factors available with *ImpressIt* software. This software allows input compression factors ranging from 1 (most compression-worst quality) to 25 (least compression-best quality). Eight equally spaced compression factors were chosen (3, 6, 9, 12, 15, 18, 21, and 24) and applied to all seven original images. The results of these tests are recorded in the Absolute Compression and Time section of Appendix D — Data Tables. Additionally, the results for all seven images were averaged and plotted on Graphs 5-1 and 5-2.



Graph 5–1



Graph 5–2

These tests indicate that the time necessary to compress an image increases as the level of quality required goes up (or as the amount of compression goes down). For the seven sample images used here, the average total time for compression factor 24 was 58% longer than the total time at compression factor 3. An interesting feature to note concerning this data is that the time requirements do not follow a linear path, but rather a gradually sloping exponential curve (see Graph 5-1).

Additionally, these tests indicate that the amount of absolute compression achieved using the various compression factors also follows a exponential curve (see Graph 5-2). This curve, however, shows that the average absolute compression at compression factor 3 is 1800% larger than that obtained at compression factor 24. It is interesting to note when comparing these two graphs that decreasing compression factors not only result in greater absolute compression, but

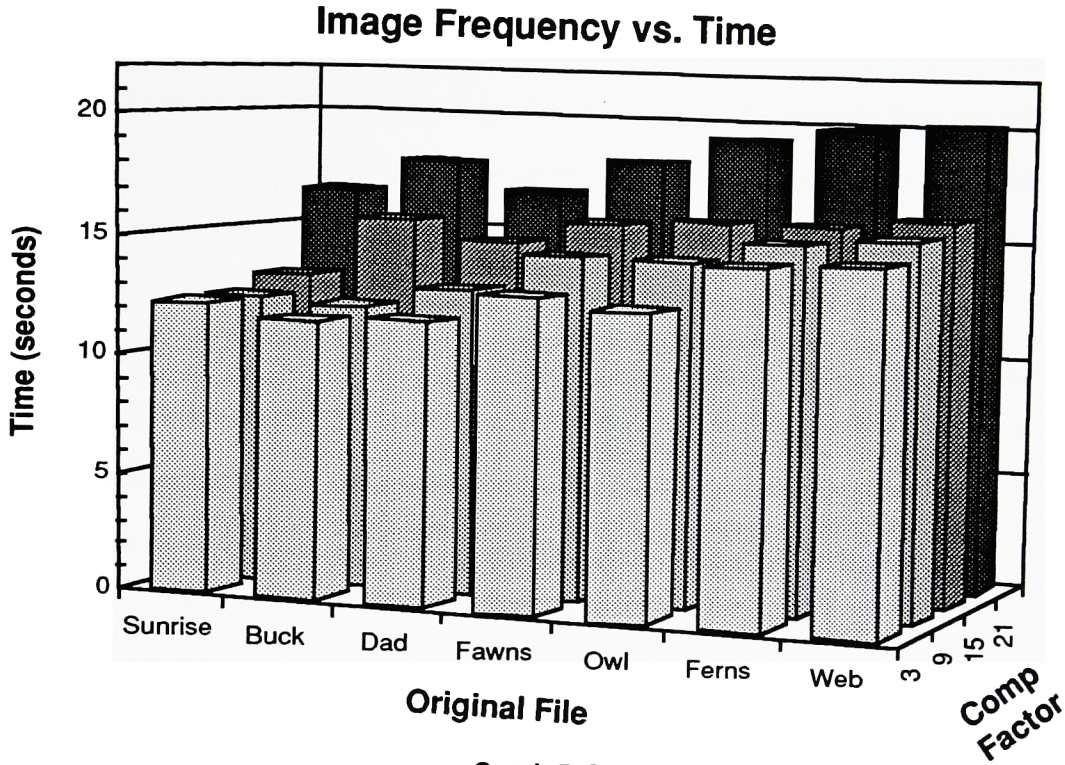
also in significant time savings. The catch to all of this, of course, is that image quality is constantly degrading as absolute compression ratios become higher.

Image Frequency

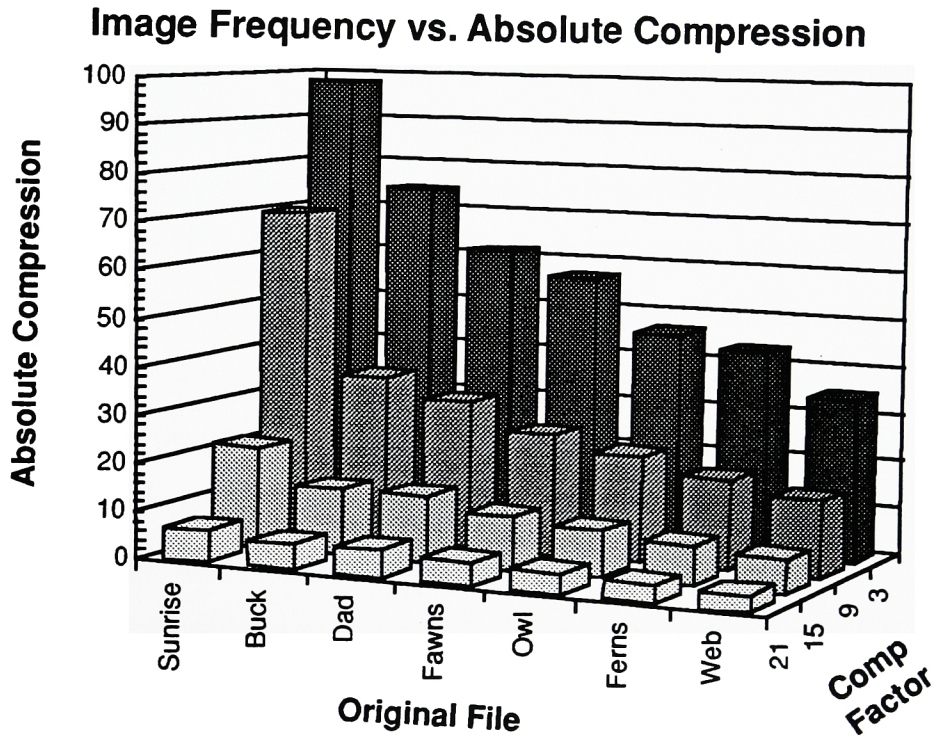
Image frequency is "the rate at which brightness of an image changes from light to dark"[1]. What this means in layman's terms is that images with rapid variations between light and dark areas have higher frequencies than those which contain large uniform areas. The images that were selected for use in this reference were specifically chosen so that they covered a wide range of frequency levels. This is an important aspect to consider when testing JPEG software because images of various frequencies react differently when compressed. The images used in this reference are arranged as follows according to frequency:

1) Sunrise — Lowest Frequency
2) Buck
3) Dad
4) Fawns — Medium Frequency
5) Owl
6) Ferns
7) Web — Highest Frequency

The purpose of the tests run in this series was to determine the amount of time and absolute compression variation that occurred as a result of different image frequencies. All seven test images were compressed and decompressed at four different compression factors (3, 9, 15, and 21). The results of these tests are recorded in the Absolute Compression and Time section of Appendix D — Data Tables. In addition, the results for all seven images are plotted on Graphs 5-3 and 5-4.



Graph 5–3



Graph 5–4

Test results indicate that the variation in total compression/decompression times differ only slightly between images of varying frequency. Comparison between the highest and lowest frequency images shows that total compression/decompression time is 20% longer for the high frequency image at compression factor 21 and 22% longer at compression factor 3.

Absolute compression, on the other hand, differs greatly between images of varying frequency. Comparison between the highest and lowest frequency images shows that absolute compression is 97% greater for the low frequency image at compression factor 21 and 187% greater at compression factor 3.

In the case of absolute compression, percentage differences at compression factor 3 are roughly twice those obtained at compression factor 21. This is a somewhat unusual finding from the author's perspective, since it intrinsically seems that the relative variations due to image frequency should be uniform (on a percentage basis) no matter which compression factor is used.

Resolution

Image resolution has a significant impact on many aspects of JPEG compression. It is fairly easy to realize that as resolution increases, file size will also increase and thus lead to longer compression and decompression times. The assumption is typically made that this time increase will follow a path parallel to the increasing file size (for an explanation of how file size varies with resolution, see Appendix E — Image Size, Resolution and File Size). Additionally, it is assumed that absolute compression at a fixed input compression factor will remain constant, given a fixed image, regardless of the resolution. Finally, many people believe that the amount of acceptable compression decreases as resolution increases because higher resolution images have more detail. Every one of the preceding three assumptions can be shown to be incorrect.

The reason these assumptions do not hold true has to do with the way the JPEG algorithm works. Since JPEG compression always breaks an image down into blocks which are eight pixels by eight pixels, the unit area which is being sampled becomes smaller and smaller as resolution

goes up. As an example, see Figure 5-1 which simulates an image recorded at two different resolutions.

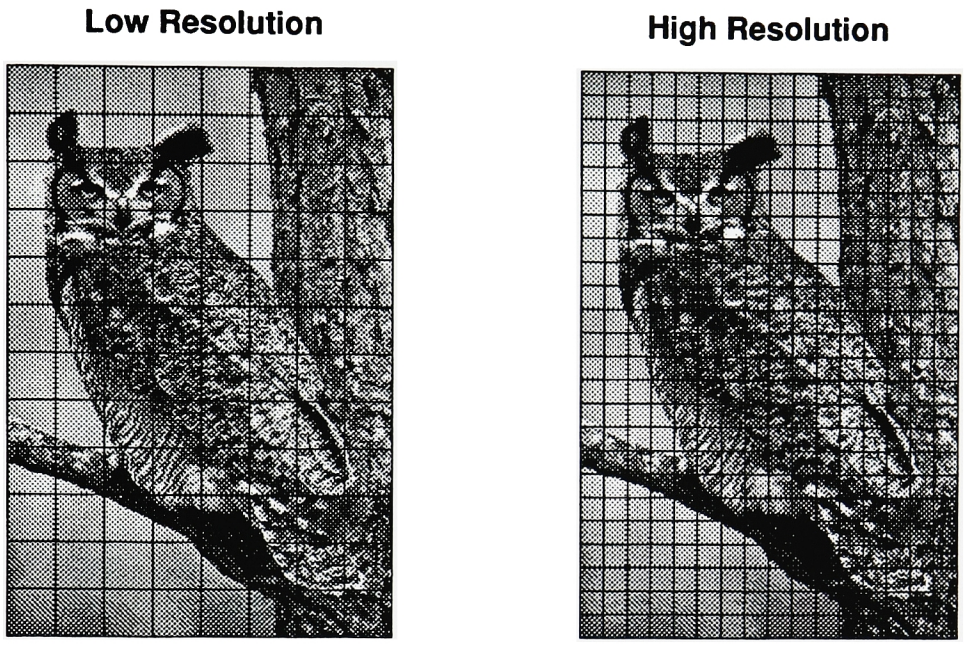
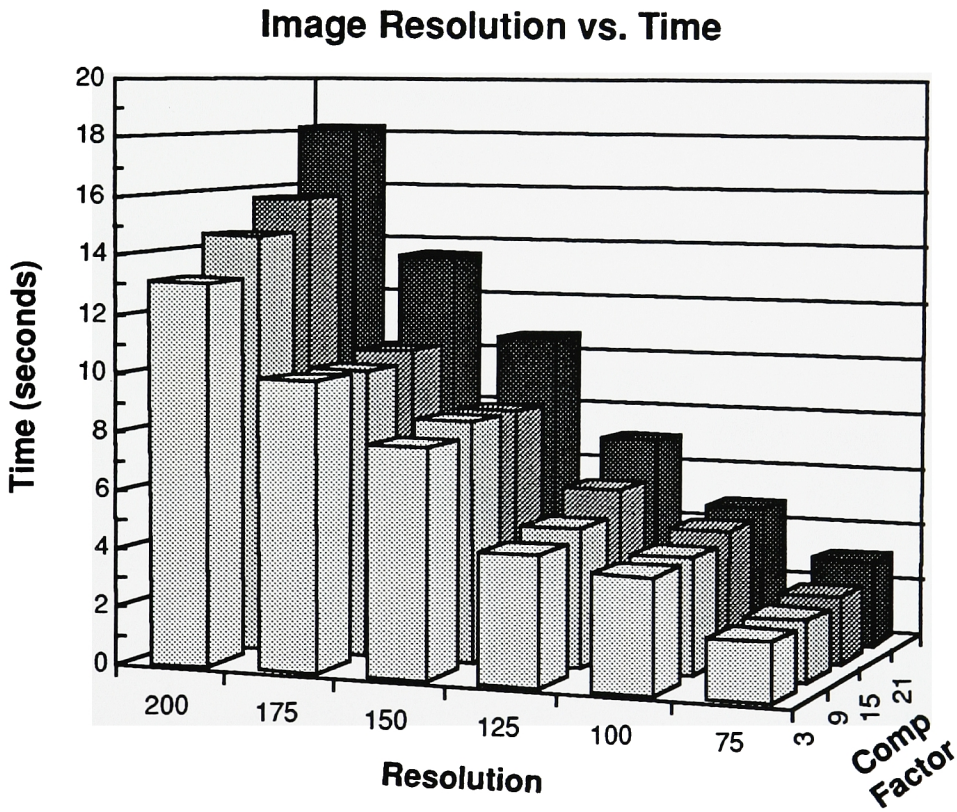


Figure 5–1

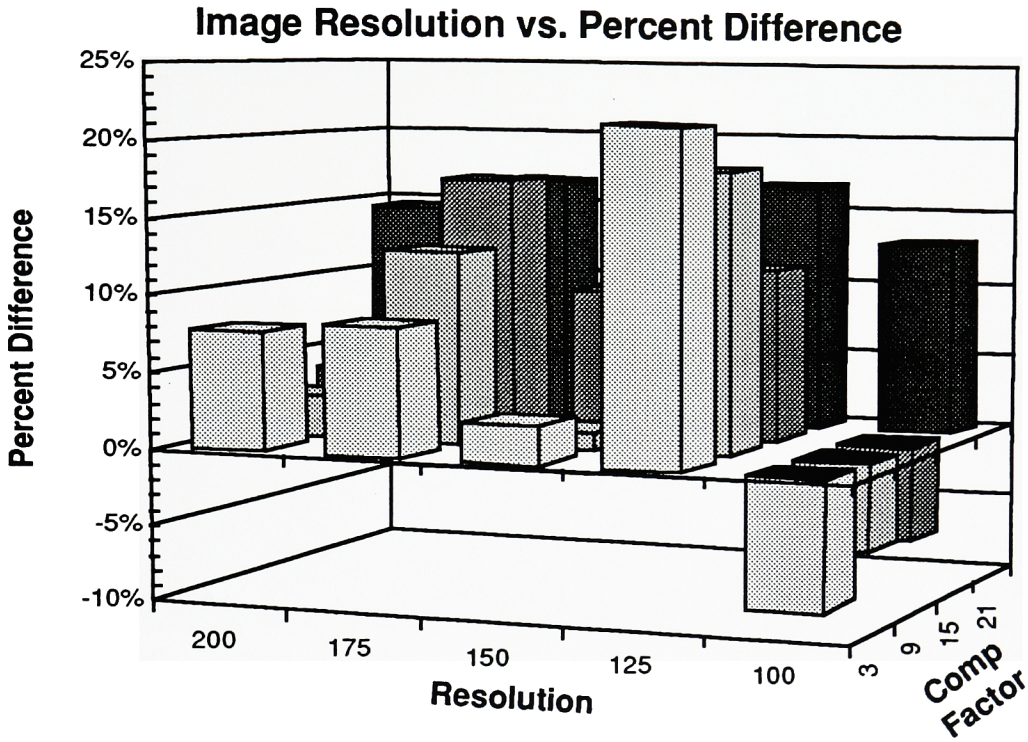
In the low resolution image on the left, each block represents an eight by eight matrix that JPEG would average when compressing. The likelihood of significant luminosity variations across large areas such as these are very high, which in turn means decreased JPEG performance. The image on the right, however, is stored at a higher resolution and each eight by eight sampling covers a smaller portion of the entire image. As this sampling area continues to decrease in size, the likelihood of large luminosity variation drops and the likelihood of blocks which contain uniform pixels increases. Since JPEG works most effectively on blocks which contain uniform pixels, this leads to faster relative compression/decompression times as well as higher absolute compression ratios. Additionally, as this sampling area becomes smaller and smaller, the chance that the human eye will be able to recognize the image degradation also decreases dramatically. In fact, if the resolution were high enough that an eight pixel by eight pixel block was too small

for the human eye to perceive as anything but a single point, JPEG could compress the block to the maximum possible level with no apparent image degradation.

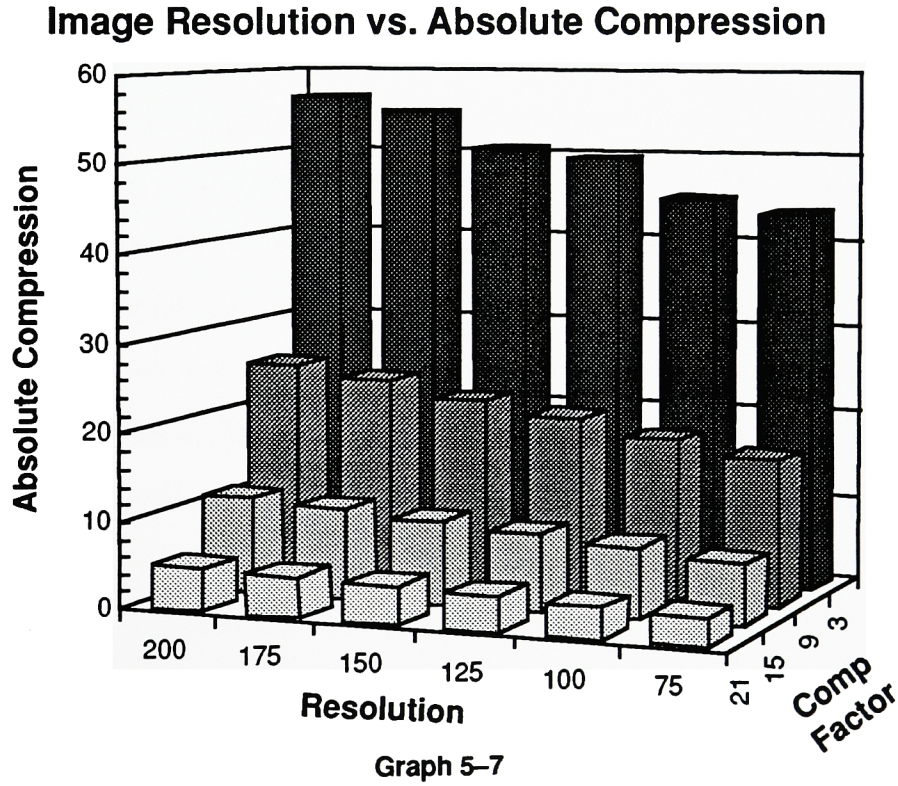
Tests intended to prove the preceding points were performed using a single medium frequency image (Fawns), which was scanned at six different resolutions (75 dpi, 100 dpi, 125 dpi, 150 dpi, 175 dpi, and 200 dpi). Since the purpose of these test sequences was to show differences due to varying resolution only, the author felt that the use of only one image was justifiable. The data which resulted from these tests is recorded in the Resolution section of Appendix D — Data Tables. In addition, the results of the tests are plotted on Graphs 5-5, 5-6, 5-7 and 5-8.



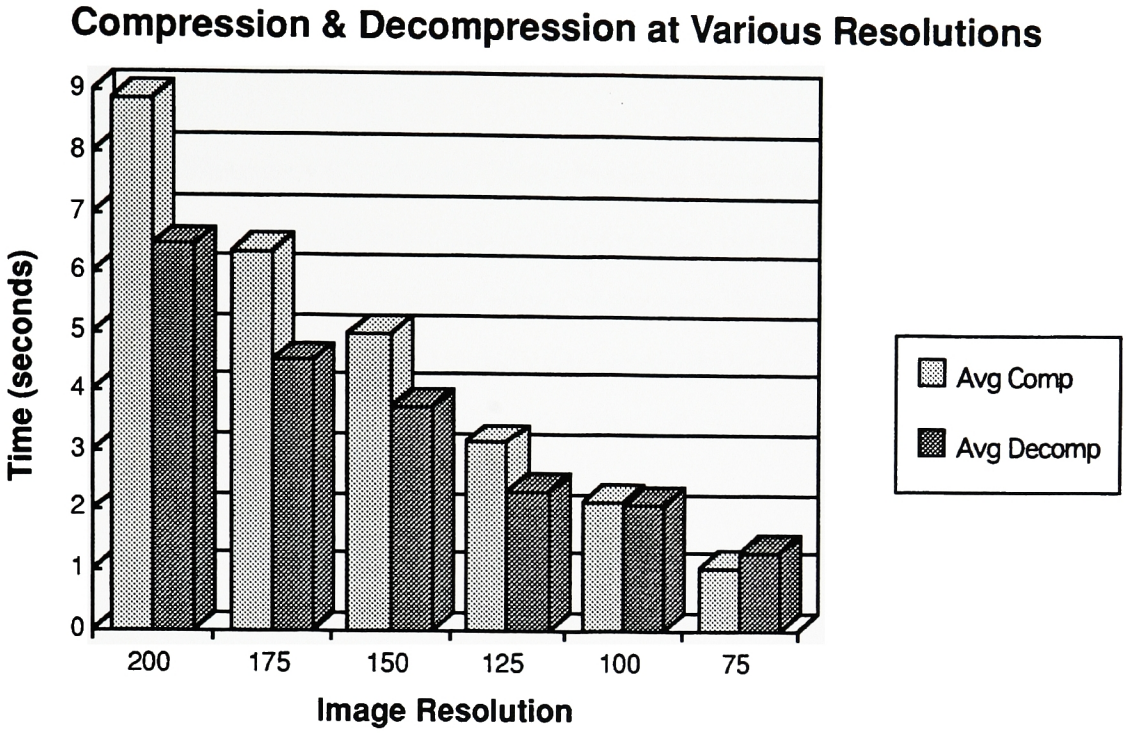
Graph 5–5



Graph 5–6



Graph 5–7



Graph 5–8

Graph 5-5 depicts the total time required to compress and decompress an image at various resolutions. While this graph gives a good visual representation of the fact that higher resolution files require more time, it is not very effective at proving the points stated above. Graph 5-6 does a much better job of proving the time component of this theory, although it requires some special explanation to thoroughly understand.

The predicted time value was calculated under the false assumption that the increase in time required by a higher resolution image should be proportionally related to the increase in file size. For example the predicted time to compress and decompress a 200 dpi image would be four times as great as the time required for a 100 dpi image, since the file size of the 200 dpi image is four times greater. Using this false assumption, predicted times were calculated for each of the five highest resolutions based on the time required for the 75 dpi image. After the tests had been

run, the actual values were recorded and compared to the predicted values using the following formula:

Percent Difference = (1 - (Actual Value / Predicted Value)) x 100

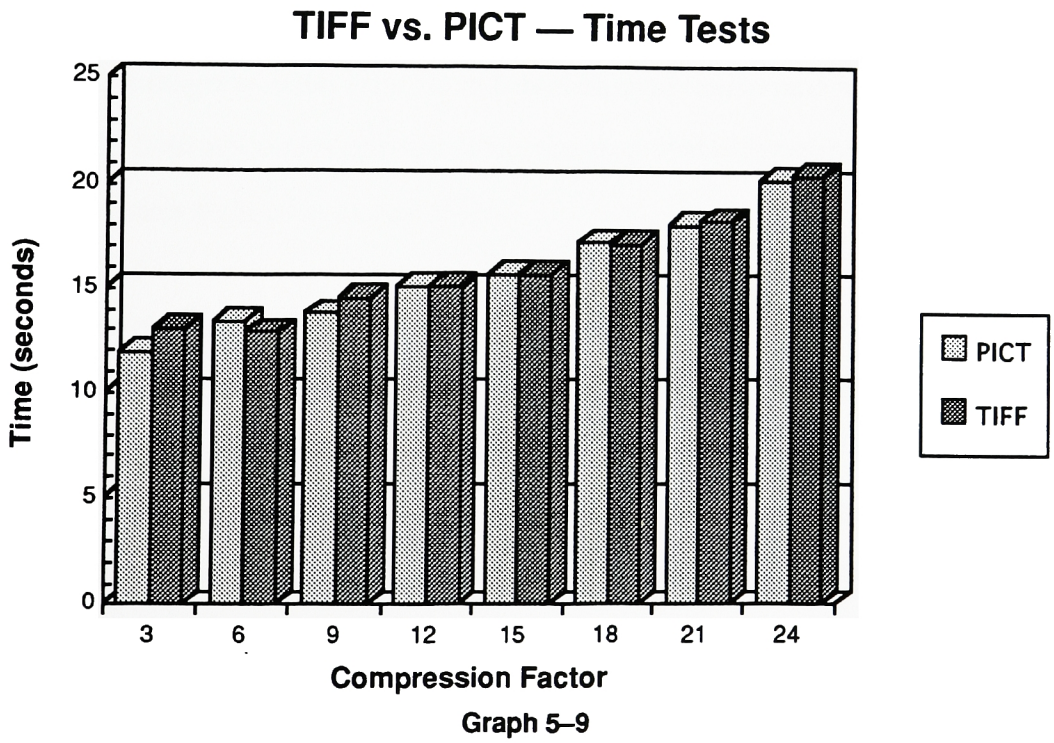
Positive values for percent difference indicate that the actual time required for compression and decompression is less than the predicted value and vice-versa for negative values. Values of zero indicate that the predicted value and actual value are exactly the same. Since the clear majority of values in Graph 5-6 fall on the positive side of the scale, it is clear that relative compression speed does increase as resolution becomes higher.

In the case of absolute compression ratios, it is very easy to prove similar results without going through nearly as much explanation. Because absolute compression ratio is calculated relative to the original file size (Absolute Compression Ratio = Original File Size / Compressed File Size), the increase in original file size due to increasing resolution is automatically compensated for. Therefore, a direct comparison of absolute compression ratios versus image resolution can be made. Graph 5-7 clearly shows that at any fixed input compression factor, absolute compression increases as image resolution increases.

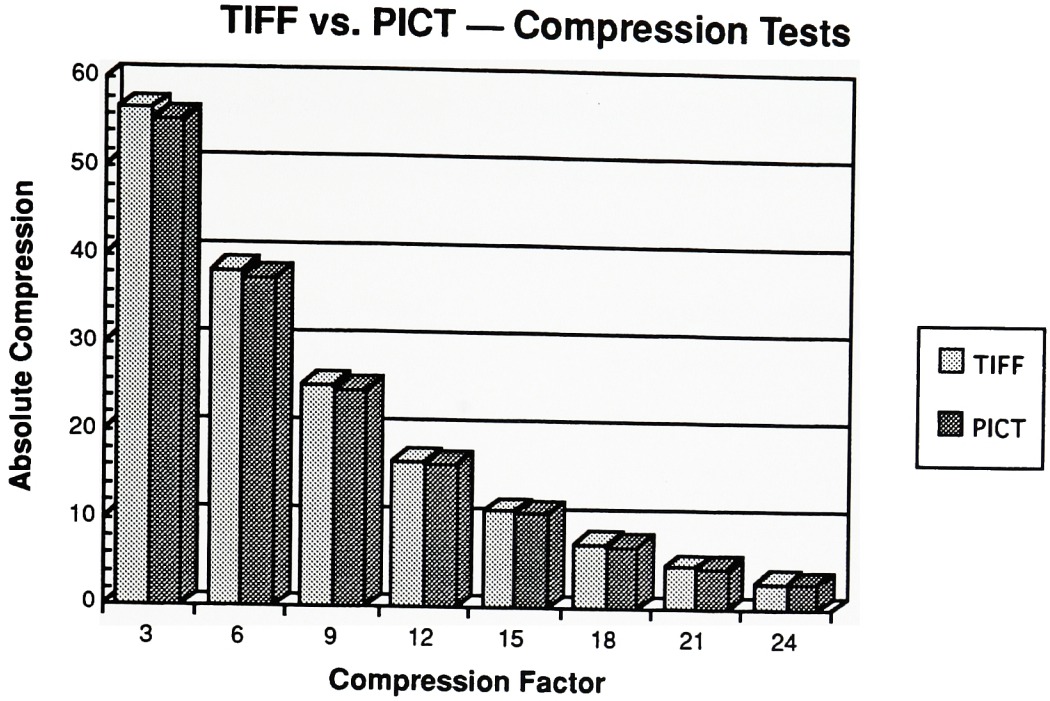
One last interesting point was unexpectedly discovered during resolution testing. In the initial tests conducted on all seven images at 200 dpi, average compression time was 37% longer than decompression time (standard deviation of 5.45%). It was expected that this relationship was constant regardless of file size until the tests conducted for resolution showed otherwise. Graph 5-8 shows that the relationship between compression time and decompression time varies with file size. With higher resolution images, and thus larger file sizes, compression time is noticeably longer than decompression time. However, as resolution and file size decrease, the relationship begins to even out until eventually decompression actually takes longer.

File Format

An issue which is frequently ignored when testing JPEG products is the difference between various file formats. Often tests are run on a variety of images saved in different file formats, but never on the exact same image saved in various formats. The author knows from previous experience that an image saved in TIFF format and PICT format occupy roughly the same amount of storage space. Therefore, there is really no advantage to saving the file in any particular format as far as storage space is concerned. The purpose of the tests in this section was to determine if any advantage could be gained during JPEG compression by saving the original file in a specific file format. Since *ImpressIt* software is limited to compressing files which are saved in TIFF or PICT format, testing was conducted using only these two options. *Adobe Photoshop* was used to open the original TIFF file (Fawns) and save it as a PICT file. Tests were then run comparing compression/decompression time as well as absolute compression and recorded in the File Format section of Appendix D — Data Tables. Additionally, the results of these tests are plotted on Graphs 5-9 and 5-10 below.



Graph 5–9



Graph 5–10

These graphs make it quite clear that neither file format has any significant advantage over the other as far as JPEG compression is concerned. Total compression/decompression time averaged 1.3% longer with the TIFF file and absolute compression ratio was 2.1% greater with the PICT file. Both of these insignificant differences are probably due to the fact that the original PICT file requires slightly less storage space.

Sub-Sampling Ratios

Many JPEG products, *ImpressIt* included, offer an option to sample the pixel data at different ratios. As mentioned in Chapter 2 — Theory and Background, the human eye perceives luminance to be more important than chrominance when distinguishing details. Since JPEG compression typically works in the YUV color space (where U and V describe chrominance and Y describes luminance), there exists the option to selectively disregard some of the chrominance

information and increase performance. The amount of information that is disregarded is designated by a term known as sub-sampling ratio. For example, if the sub-sampling ratio is 2:1:1, then every other row and column of U and V information is discarded. On the other hand, if the sub-sampling ratio is 1:1:1, then no information is discarded (see Figure 5-2).

Sub-Sampling Ratio = 1:1:1

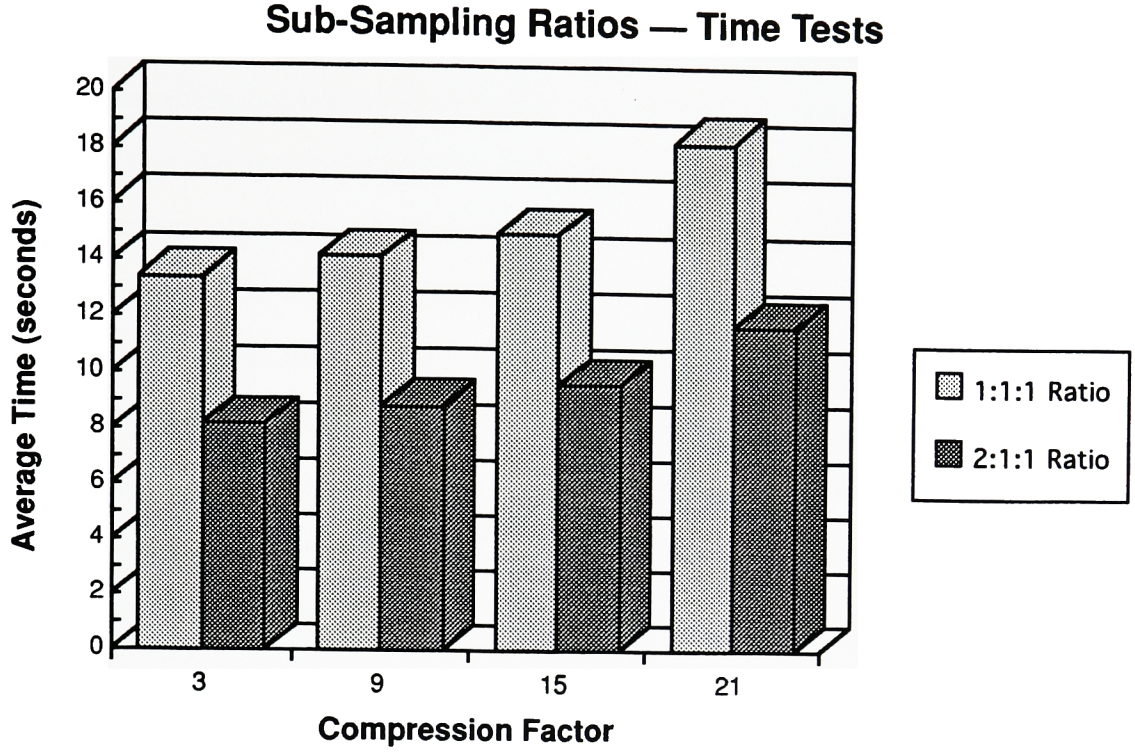
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV
YUV	YUV	YUV	YUV	YUV	YUV	YUV	YUV

Sub-Sampling Ratio = 2:1:1

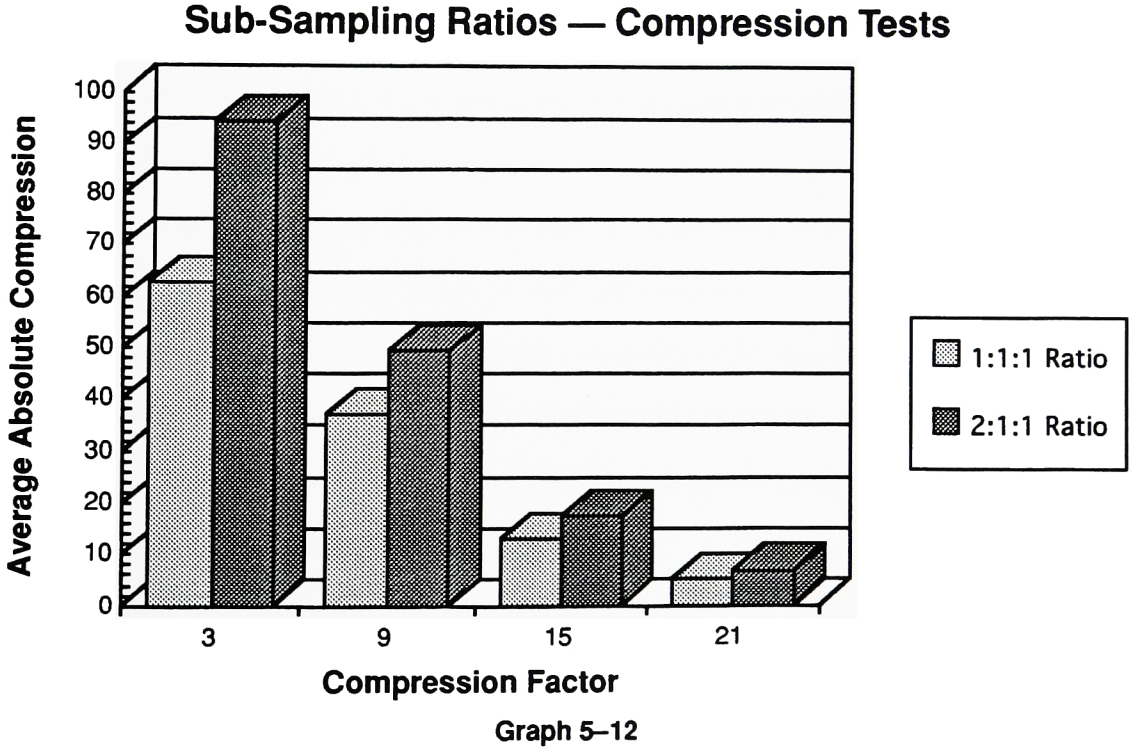
YUV	Y	YUV	Y	YUV	Y	YUV	Y
Y	Y	Y	Y	Y	Y	Y	Y
YUV	Y	YUV	Y	YUV	Y	YUV	Y
Y	Y	Y	Y	Y	Y	Y	Y
YUV	Y	YUV	Y	YUV	Y	YUV	Y
Y	Y	Y	Y	Y	Y	Y	Y
YUV	Y	YUV	Y	YUV	Y	YUV	Y
Y	Y	Y	Y	Y	Y	Y	Y

Figure 5–2[2]

Tests for this section were conducted using three files (Sunrise, Fawns, and Web) at four compression factors (3, 9, 15, and 21). Since *ImpressIt* is limited to 1:1:1 and 2:1:1 sub-sampling, these were the only combinations which were tested. Data for these tests is recorded in the sub-sampling ratio section of Appendix D — Data Tables. Graphs 5-11 and 5-12 graphically depict the time and absolute compression advantages of 2:1:1 sub-sampling. Additionally, Appendix C — Sub-Sampling Ratio Comparisons, presents comparisons of the actual images compressed using different sub-sampling ratios.



Graph 5–11



Graph 5–12

Averaged results for all three files indicate that 1:1:1 sub-sampling takes 58% longer than 2:1:1 sub-sampling and delivers 30% less absolute compression. These gains are quite significant and could lead to dramatic storage space and time savings, especially in high volume operations. The advantages gained using this technique are somewhat offset by the fact that image quality degrades slightly. The real choice here is whether the best combination for a certain absolute compression is gained through a low compression factor and 1:1:1 sub-sampling, or a higher compression factor and 2:1:1 sub-sampling. Test images included in this reference indicate that the use of 2:1:1 sub-sampling results in considerably higher quality at a given absolute compression ratio (see pages C–4 and C–5 of Appendix C and compare the two images with absolute compression ratios of 98:1).

Compression Cycles

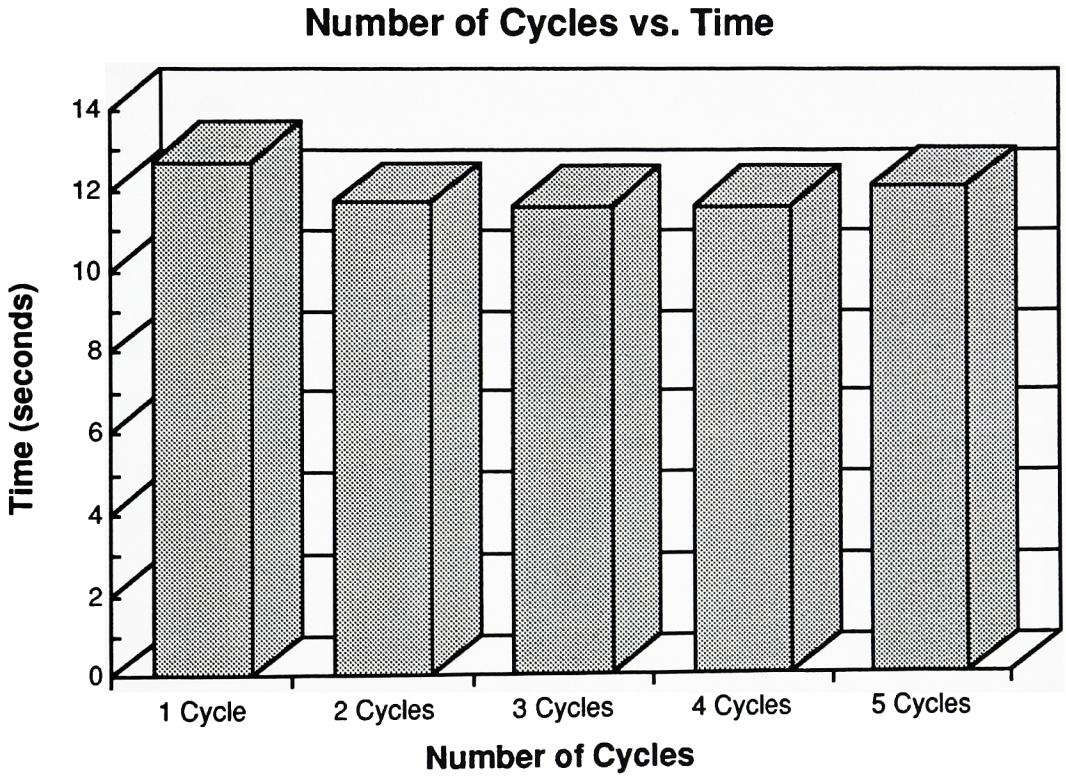
One of the most frequently asked questions the author has encountered concerning JPEG compression is the issue of multiple compression/decompression cycles. Many users are worried that each time an image goes through a compression/decompression cycle, the quality of the image will continue to decline. If this was indeed the case, users would be forced to consider the number of times an image might be compressed and decompressed when figuring out the initial acceptable compression factor. This would lead to a great many headaches, since it is usually impossible to predict how many times this cycle might occur.

In order to test the effects of multiple compression/decompression cycles, a medium frequency image (Fawns) was processed through five cycles at compression factor 3. Compression factor 3 was chosen because it represents the highest compression setting used in all tests, and thus would exaggerate any differences which occur.

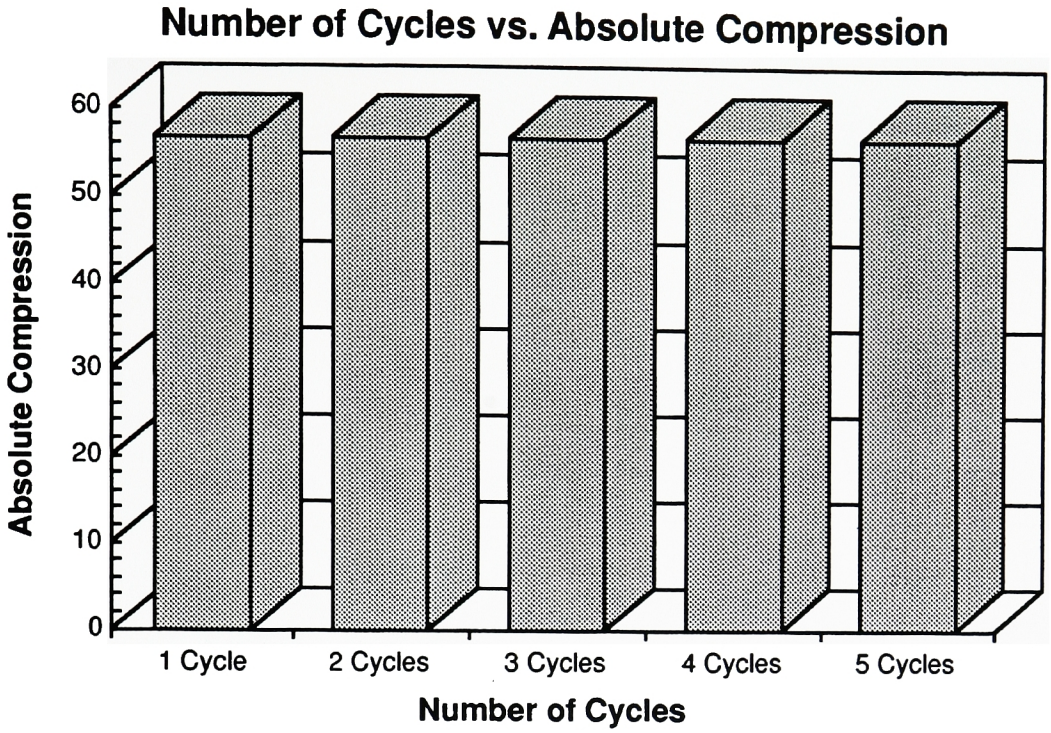
The first two tests conducted were designed to see if the amount of time required or absolute compression achieved varied after multiple cycles. If any changes occurred with either of these variables, one could be relatively certain that image quality was indeed decreasing (since

it wasn't remaining constant and it certainly wasn't increasing). The third set of tests used the difference command in *Adobe Photoshop* to compare the image that had been through one compression/decompression cycle with the images that had been through multiple cycles (see Chapter 4 — Description of Software, for a thorough explanation of how the difference command in *Adobe Photoshop* works).

Results for the first two tests are contained in the Compression Cycles section of Appendix D — Data Tables and also in Graphs 5-13 and 5-14 below. The results for the third set of tests are contained in the histograms of Figures 5-3 thru 5-6.



Graph 5–13



Graph 5–14

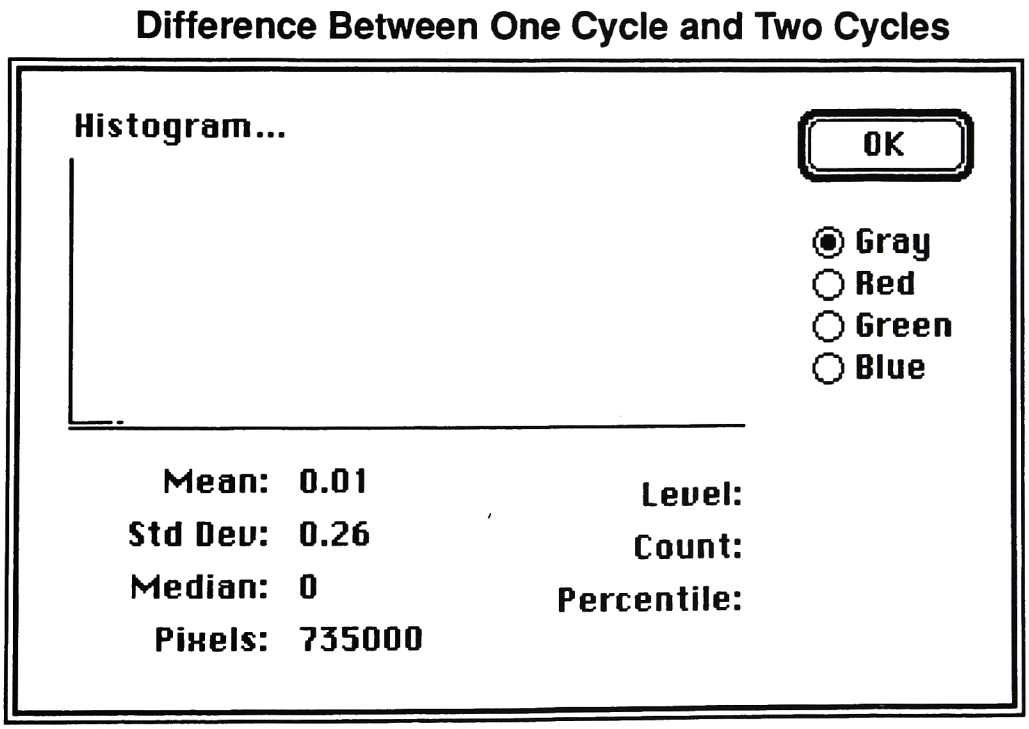


Figure 5–3

Difference Between One Cycle and Three Cycles

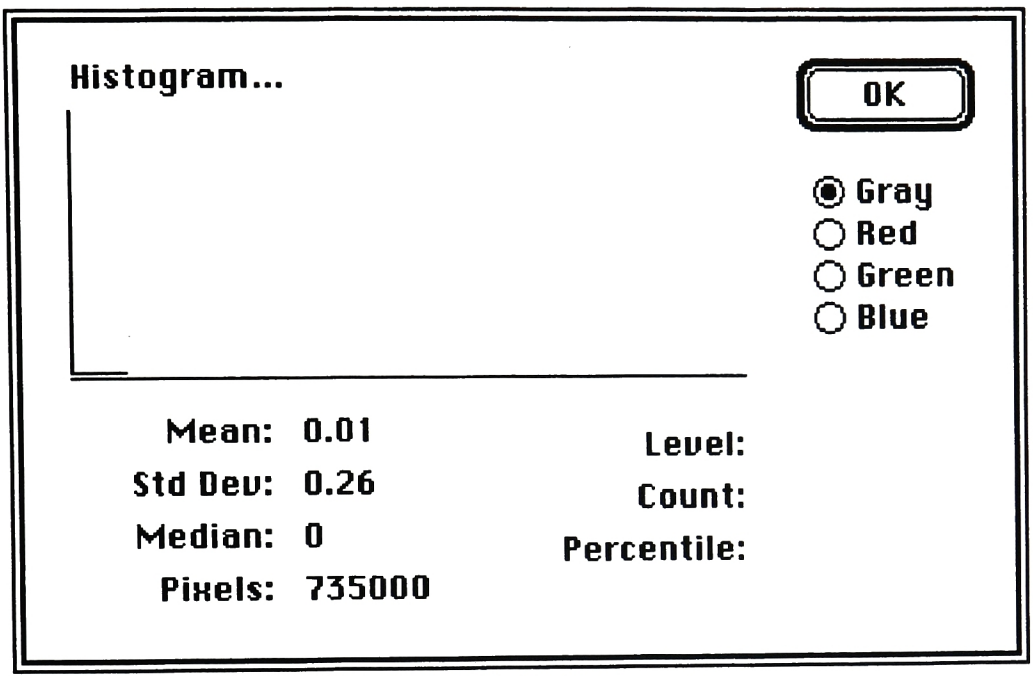


Figure 5–4

Difference Between One Cycle and Four Cycles

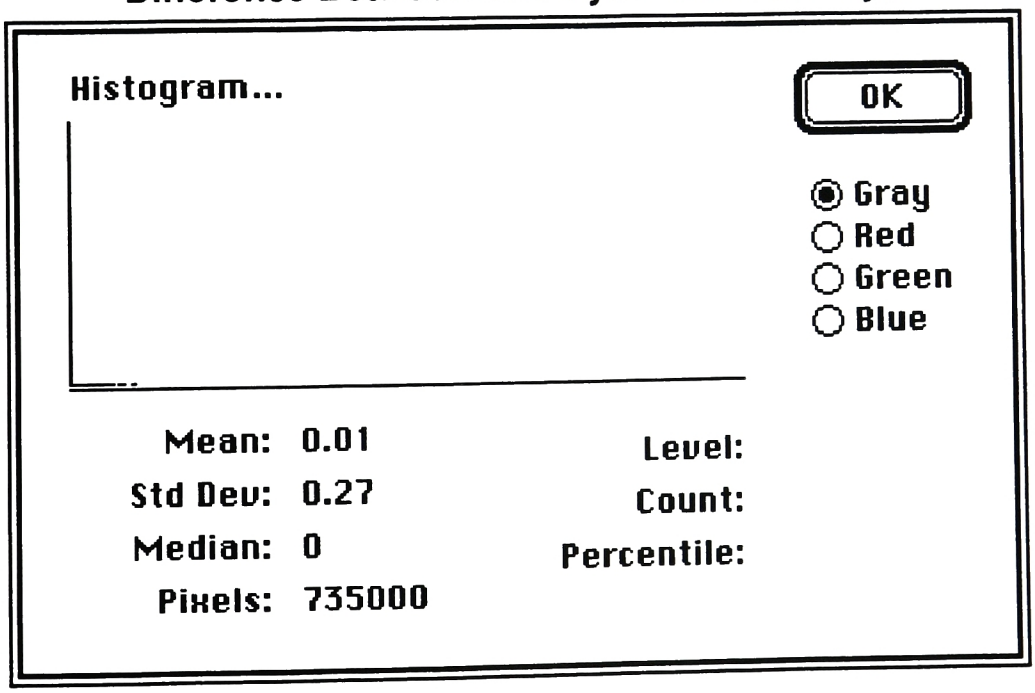


Figure 5–5

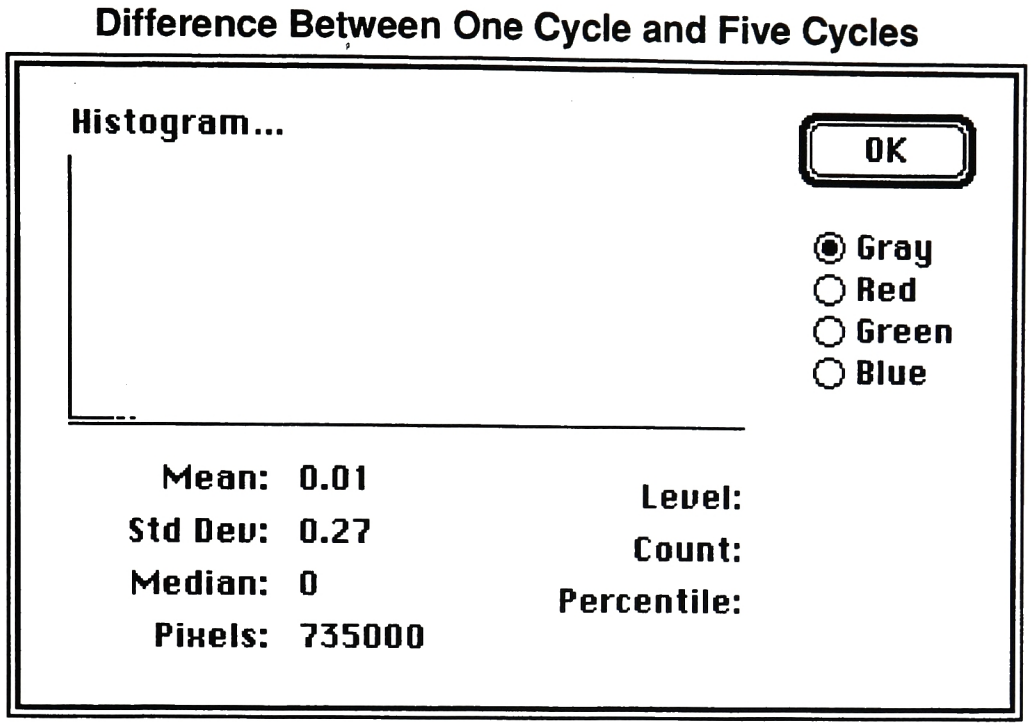


Figure 5–6

All of these tests lead to the conclusion that multiple compression/decompression cycles have very little effect on image quality. This statement, however, does come with a few qualifiers. First of all, the test image was compressed using the same compression factor each time which may, or may not, be a realistic expectation. The effects of cycling an image at many different compression ratios has not been tested here, although the author's guess is that the image will not degrade any lower than the state it was in at its highest absolute compression. Secondly, there were no changes made to the test image in between compression cycles. Any changes that are made to the image while decompressed, such as retouching, cutting and pasting, etc., will also suffer the effects of compression the next time the image is cycled.

Footnotes

[1]Baxes, Gregory. *Digital Image Processing.* Cascade Press, 1984. Page 23.

[2]*Radius ImpressIt™ User's Manual.* Radius Inc., 1991. Page 32.

With the use of color images becoming more widespread every day, it is inevitable that the presence of JPEG compression will continue to grow. The advantages JPEG offers to save disk space, speed up transmission over phone lines, and expedite the movement of files over networks guarantees that it will play a major role in the future of graphic arts and electronic publishing. Although the existing JPEG technology is performing quite well, there are many improvements in store for the future.

PostScript Level 2

One improvement in JPEG compression that is already beginning to show up in many of the software packages available is the ability to compress JPEG files into the encapsulated PostScript (EPS) format. "This allows compressed images to be scaled, cropped and rotated in graphics and layout programs and printed to Level 2 devices, where they are decompressed by the interpreter."[1]

The advantages of this system are twofold: 1) transmission time to a Level 2 output device is substantially reduced, freeing up the user's computer in much less time and 2) elements composed of fine details do not suffer the effects of image degradation. "When JPEG and EPS work together, JPEG handles continuous-tone color-image compression while PostScript instructions preserve details like fine lines and text".[2] It is important to note that this does not mean PostScript magically identifies the areas in a continuous-tone photograph which are composed of fine details and saves these areas from the effects of compression. The only areas that are preserved by Postscript are those elements which are described in terms of PostScript code to begin with, such as line illustrations and text. Areas of fine detail within the photograph will suffer the same amount of degradation no matter what format is used to save the compressed file.

User Transparency

According to the major manufacturers of JPEG products, one of the goals in the future is to make JPEG compression as unobtrusive as possible to the end user.

> "From the user's point of view, you want compression to be invisible," said Bill Krause, president and CEO of Storm Technology. . . . "We're working with all the major application developers and equipment manufacturers to get them to embed our technology in their scanners, cameras or printers on the I/O [input/output] side, and with software application developers like Adobe, Quark, Microsoft and Fractal Design".[3]

There are pitfalls involved with this scenario, however, because there still exist some serious compatibility problems between different JPEG products. For example, suppose an image has been transparently compressed with company X's algorithm in the background of Quark (in something other than the interchange format) and then sent to a printer that uses company Y's algorithm to decompress. The way technology stands now, there is a very good chance the decompression will generate an error. Storm Technology is attempting to overcome this problem by becoming a "clearing house" that will inspect all competitors products and give them the Storm "Seal of Approval" if they meet specifications.[4] This is one way of ensuring compatibility, however, no one knows whether this plan will ever gain wide-spread acceptance.

The other problem with transparent compression is that it is likely to go about setting compression ratios without any input from the user. As this reference has shown, there are numerous considerations which must be taken into account when developing a compression strategy. Unless the products which offer background compression allow the user a great deal of control over the process (which is highly unlikely since the whole idea is to make the technology transparent), this feature will be basically useless to anyone who demands high quality images. Hopefully the products that do not allow the kind of control professional users demand will at least contain a provision for disabling the background compression.

Additional User Controls

As JPEG compression stands today, the choices users have are generally limited to various compression factors and, in some cases, various sub-sampling ratios. In the future, JPEG products will incorporate enhanced feature sets which will allow users to fine-tune compression for specific situations. One example of this type of technology has already been developed by Storm Technology under the name JPEG++.

> It so happens that JPEG compression is very bad for text within an image; the edges are often blurry or jaggy. To combat this, Storm Technology developed a proprietary extension of JPEG, which it calls JPEG++. It allows the operator to select a rectangular portion of an image in which to preserve maximum quality. Within that rectangle, Storm uses only lossless compression techniques and throws away no data at all, so the compression ratio there is fairly low. However, in the rest of the image normal JPEG compression is applied.[5]

Other features along these lines are likely to continue showing up as manufacturers of JPEG products continue to strive for differentiation in the marketplace. One feature this author would like to see is a provision similar to JPEG++ which allowed the user to select multiple areas of the image and apply various compression factors to each. This would allow extremely high compression ratios in areas void of detail while still maintaining image integrity in important areas. The hitch in any sort of enhancement along these lines is that other JPEG products must be able to decompress files generated using the enhancement. Since these techniques are likely to be proprietary, as is the case with JPEG++, the advantage gained by the specialized function may be negated by the fact that no one else will be able to decompress the files generated.

Another area which contains possibilities for enhancement involves a trade-off between compression/decompression times and image quality.

> We see no reason why there could not be other improvements on the compression side. For example, all current products focus on fast compression as well as fast decompression. This symmetry is important in low-volume or do-it-yourself applications such as desktop publishing and office image archiving. But in systems that prepare images for mass distribution on CD-ROMS, it might pay to spend a long time on compression to fully optimize the quality/size tradeoff.[6]

It seems likely that there are many users who would be very interested in a product that allowed higher quality images, given a fixed absolute compression ratio, at the expense of added processing time.

AC Prediction

One of the first clues that an image has been compressed using a JPEG algorithm is the appearance of blocking artifacts throughout the image. These artifacts result from the image being broken down into individual eight-by-eight blocks and averaged when high quantization values are used. The artifacts appear as very distinct, hard edges which clearly show the blocking structure in the decompressed image (examples of this phenomenon can be observed in the images compressed at higher levels in Appendix B — Compression Comparisons).

The primary reason for these artifacts is that a significant number of the AC coefficients are transformed to zero when large quantization values are used. Therefore, when the image is decompressed the decoder does not have any information available and uses the overall average (the DC coefficient) to fill in the zero spaces. If zero spaces cover the majority of the available area in an eight-by-eight block, the resulting decompressed block appears as one continuous color. When this process is repeated on adjacent eight-by-eight blocks, the edges between each of the blocks become very noticeable and result in unacceptable image quality.

To help eliminate this problem, a process known as AC prediction has been developed, although it is currently not used in many of the JPEG products available. The basic idea behind AC prediction is that the unquantized DC coefficients of the eight surrounding neighbors nearest

each eight-by-eight block are used to predict zero AC coefficients. The mathematics involved are beyond the scope of this reference, but the end effect of the process is less blocking artifacts in the decompressed image.

> The predicted AC coefficients definitely help to remove blocking artifacts in smooth background areas of the image. However, the predictions often are wrong at the edges, and in any event must be suppressed wherever non-zero AC coefficient values are decoded.[7]

Custom Quantization Tables

As mentioned in Chapter 3 — Compression Considerations, one of the factors which determines the visibility of compression artifacts is the output device utilized. Since each device has different characteristics when it comes to compression, it only makes sense that quantization tables will be developed which are customized for the different devices. By utilizing custom tables such as this, areas which are normally sensitive to compression artifacts can be quantized using smaller values and vice-versa for areas which can tolerate greater levels of compression.

It is likely that these custom quantization tables will be incorporated into JPEG products in the form of pull down menus similar to those currently available for color matching functions. Since the JPEG standard already allows any number of different quantization tables to be utilized, this feature should not introduce any new incompatibility problems. One concern that will arise for an average user, however, is the question of which settings to use when the image is being sent to multiple locations. The answer to this question will depend on the individual circumstances and can only be determined after careful consideration of the factors which apply to all types of JPEG compression.

CIE Color Spaces

At the present time, most JPEG compression products convert RGB color values into YUV color values before compression. The reason for the this transformation is fully explained in the section on Sub-Sampling Ratios in Chapter 5 — Objective Tests, therefore, it is simply re-stated here that the YUV color model allows for greater compression ratios than the RGB color model. The problem with the YUV color model, however, is that uniform changes in the coordinates do not result in uniform color changes as perceived by the human eye. This means that a slight shift in coordinates resulting from the compression/decompression process may lead to large distortions in some areas while resulting in small distortions in other areas. Since this shift varies from area to area, it makes it extremely difficult for programmers of JPEG compression products to build effective algorithms and quantization tables.

CIE color models, such as CIELUV and CIELAB, help to solve this problem because they provide a relatively uniform perceptual response for a given shift in coordinates. The problem with these color models is that they require complex computation when being transformed from the RGB color model, whereas the transformation to the YUV color model is quite simple. The detriment to this complex computation is that it requires significantly more processing time and most manufacturers are currently marketing their products based on speed.

In the future, however, computers will continue to become faster and faster, until eventually the increased time required for the more complex computations becomes insignificant. When this occurs, it will no longer make sense to use the YUV color model and the use of CIELUV or CIELAB models will dominate the market. Since these models provide a more uniform and accurate color space to work with, decompressed images will be of higher quality (given a fixed absolute compression ratio) than those described in terms of YUV color spaces.

It is important to note that as far as JPEG algorithms are concerned, it makes absolutely no difference which color space is used. The mathematical operations which convert one color

space to another are completely independent of the JPEG process and are in no way controlled by any JPEG standards. The differences in quality which occur between various color spaces are related to the characteristics of the color spaces themselves, not the transformation which occurs in the JPEG algorithms.

Footnotes

[1]McManus, Neil. *JPEG-compressed images can print to PS Level 2 with EPS method.* MacWEEK, December 3, 1991. Page 30.

[2]Seiter, Charles. *Macworld Lab tests 8 JPEG products.* Macworld, March 1992. Page 150.

[3]Strothman, Jim. *Still Image Compression Vendors Seek to Embed Technology in Applicatioins.* Computer Pictures, March/April 1992. Page S-10 – S-12.

[4]Paraphrased from Strothman, Jim. *Still Image Compression Vendors Seek to Embed Technology in Applicatioins.* Computer Pictures, March/April 1992. Page S-10 – S-12.

[5]Dyson, Peter. *JPEG standard compression for still images.* Digital Media, January 1992. Page 23.

[6]Dyson, Peter. *JPEG standard compression for still images.* Digital Media, January 1992. Page 23.

[7]Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression.* Van Nostrand Reinhold, 1993. Page 263.

BIBLIOGRAPHY

- Fraser, Bruce. Scan Handlers: *12 JPEG compression products that help you get a grip on large images.* Publish, April 1992. Page 56.
- Karney, James. *Calibration and compression for 24-bit color.* PC Magazine, January 1992. Page 274.
- Parascandolo, Salvatore. *ColorSqueeze, ImpressIt, and PicturePress.* MacUser, September 1991. Page 68.
- Dyson, Peter. *JPEG standard compression for still images.* Digital Media, January 1992. Page23.
- Jarrett, Lynn. *Hot topic: compression of still images.* Digital Review, September 1991. Page 38.
- Schram, Laura. *Data compression: the key to storing images.* Presstime, March 1992. Page 52.
- Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 6.
- Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 6.
- Nelson, Mark. *The Data Compression Book.* M&T Publishing, 1991. Page 364.
- Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 35.
- Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 71.
- Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 98.
- Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1993. Page 97.
- *Radius ImpressIt™ User's Manual.* Radius Inc., 1991. Page 31.
- Strothman, Jim. *Still-Image Compression Vendors Seek to Embed Technology in Applications.* Computer Pictures, March/April 1992. Page S-13.

BIBLIOGRAPHY

- Strothman, Jim. *Still-Image Compression Vendors Seek to Embed Technology in Applications.* Computer Pictures, March/April 1992. Page S-13 – S-14.
- *Adobe Photoshop® User's Guide.* Adobe Systems Incorporated, 1991. Page 149.
- *Adobe Photoshop® User's Guide.* Adobe Systems Incorporated, 1991. Page 155.
- Baxes, Gregory. *Digital Image Processing.* Cascade Press, 1984. Page 23.
- *Radius ImpressIt™ User's Manual.* Radius Inc., 1991. Page 32.
- McManus, Neil. *JPEG-compressed images can print to PS Level 2 with EPS method.* MacWEEK, December 3, 1991. Page 30.
- Seiter, Charles. *Macworld Lab tests 8 JPEG products.* Macworld, March 1992. Page 150.
- Strothman, Jim. *Still Image Compression Vendors Seek to Embed Technology in Applicatioins.* Computer Pictures, March/April 1992. Page S-10 – S-12.
- Paraphrased from Strothman, Jim. *Still Image Compression Vendors Seek to Embed Technology in Applicatioins.* Computer Pictures, March/April 1992. Page S-10 – S-12.
- Dyson, Peter. *JPEG standard compression for still images.* Digital Media, January 1992. Page23.
- Dyson, Peter. *JPEG standard compression for still images.* Digital Media, January 1992. Page23.
- Pennebaker, William and Joan Mitchell. *JPEG Still Image Data Compression.* Van Nostrand Reinhold, 1993. Page 263.

APPENDIX A — ORIGINAL IMAGES

Images presented in this appendix represent the seven test images utilized in the objective tests. These images exist in their original form in the sense that they have not been through any compression cycles. All images were assembled in order of increasing frequency using *Quark XPress* and output on a Canon CLC 500 Laser printer at 200 dpi.

SUNRISE

BUCK

DAD

FAWNS

OWL

FERNS

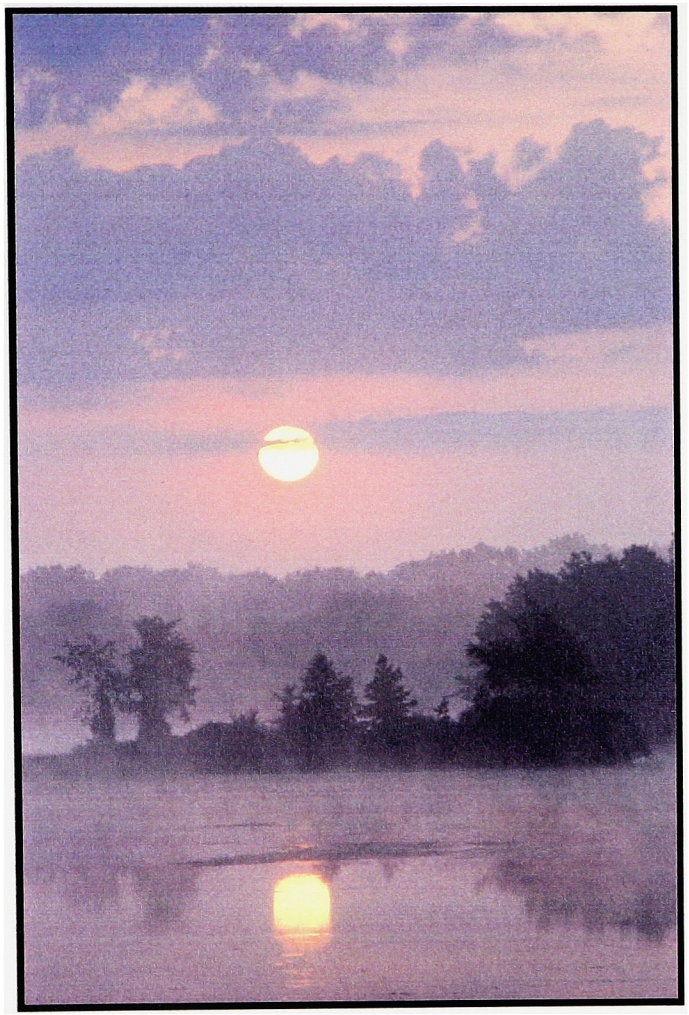
WEB

APPENDIX B — COMPRESSION COMPARISONS

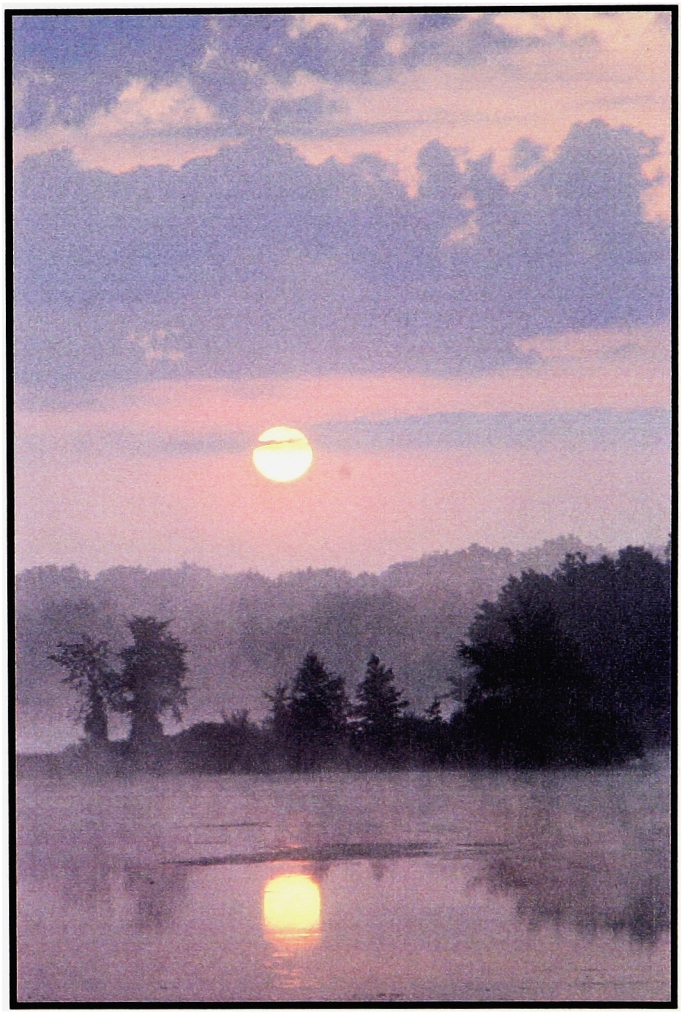
Images presented in this appendix represent the lowest frequency (Sunrise), medium frequency (Fawns), and highest frequency (Web) images utilized in the objective tests. Each image has been compressed using eight different compression factors (24, 21, 18, 15, 12, 9, 6, and 3) and arranged in order of increasing absolute compression. To effectively compare the compressed images with the original image compare under controlled 5000 K lighting.

In addition to the color samples of the compressed images, black and white files which depict the difference between the original image and certain compressed images (compression factor 21, 15, 9 and 3) are included in pages B–14 through B–19. These files were produced in *Adobe Photoshop* according to the following procedure:

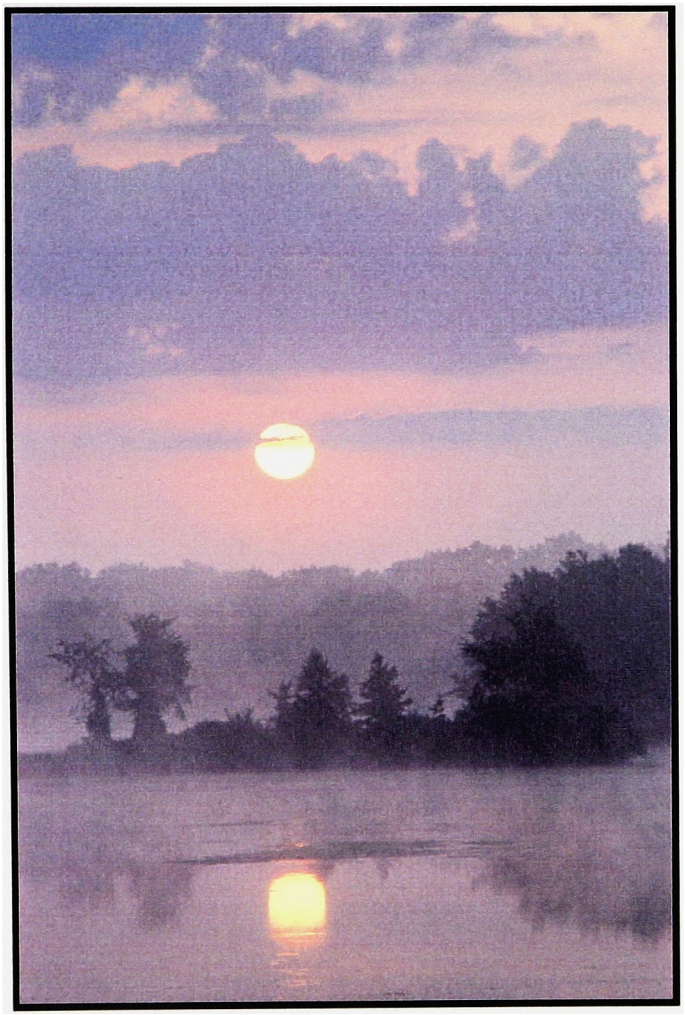
1) Utilize the difference command to compare the original file and the compressed file. Areas which contain no difference have a value of zero and appear black — the larger the difference becomes, the lighter the area becomes.
2) Apply a custom filter which multiplies the value of each pixel by ten in order to exaggerate differences for easier viewing.
3) Invert the entire image so that dark areas represent locations where change has occurred and vice-versa for light areas. This allows the viewer to discern various degrees of difference more easily.



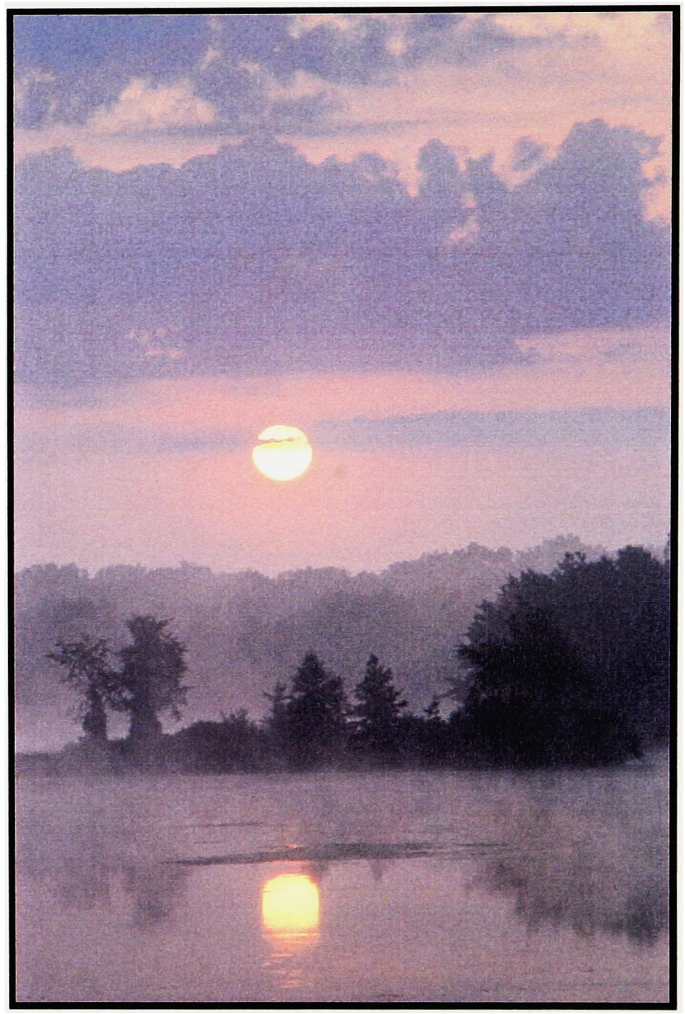
Original Size: 2,155KB Comp Factor: 21
Compressed Size: 320KB Absolute Comp: 7:1



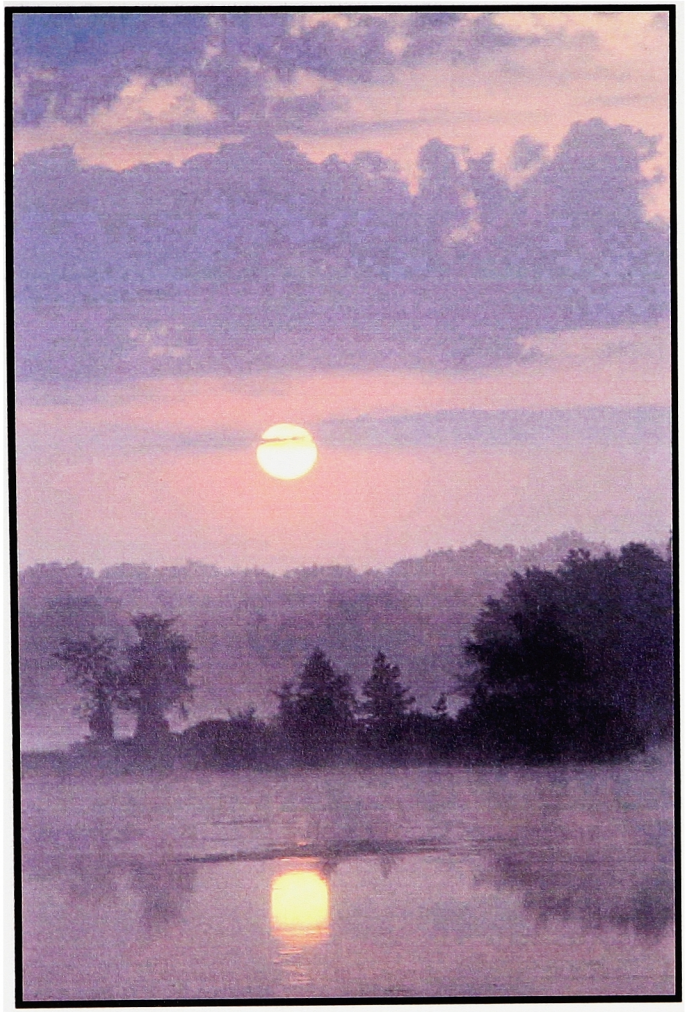
Original Size: 2,155KB Comp Factor: 24
Compressed Size: 605KB Absolute Comp: 4:1



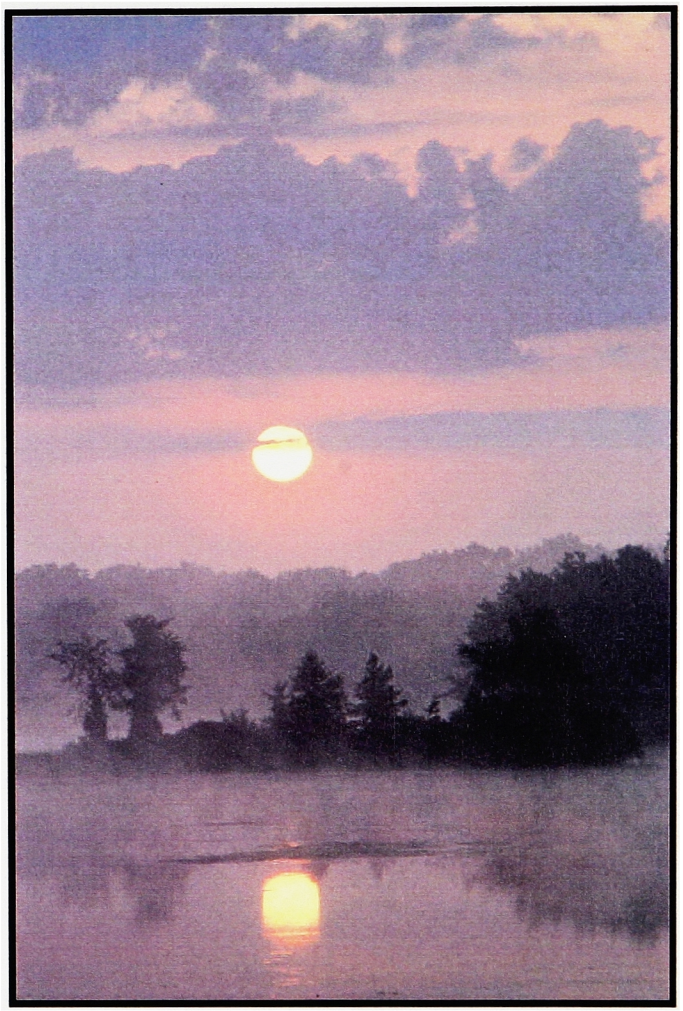
Original Size: 2,155KB Comp Factor: 15
Compressed Size: 101KB Absolute Comp: 21:1



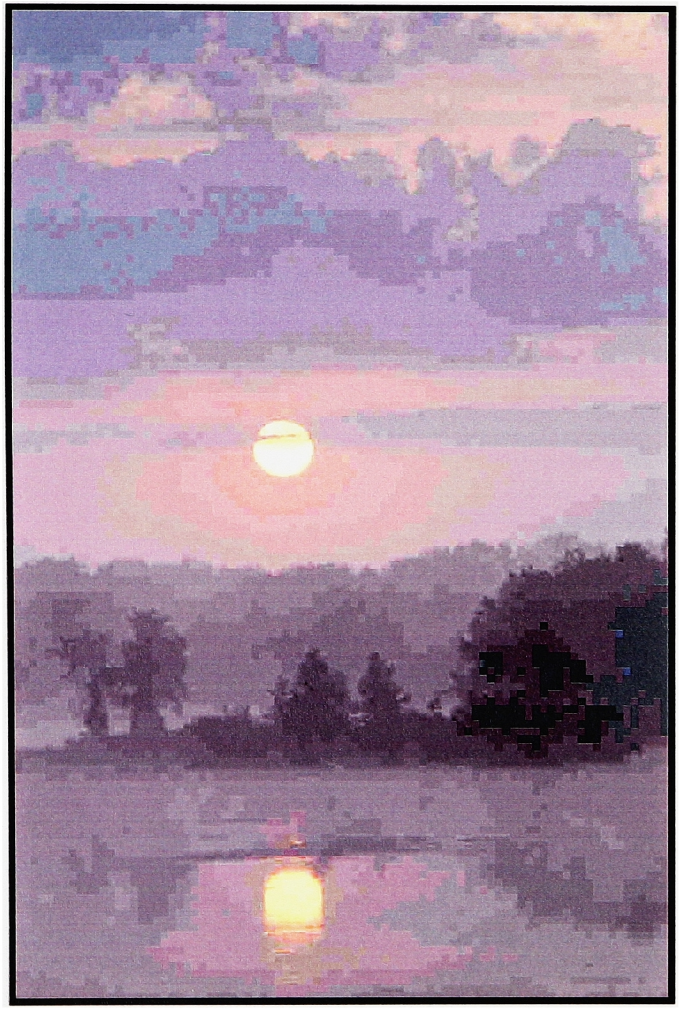
Original Size: 2,155KB Comp Factor: 18
Compressed Size: 177KB Absolute Comp: 12:1



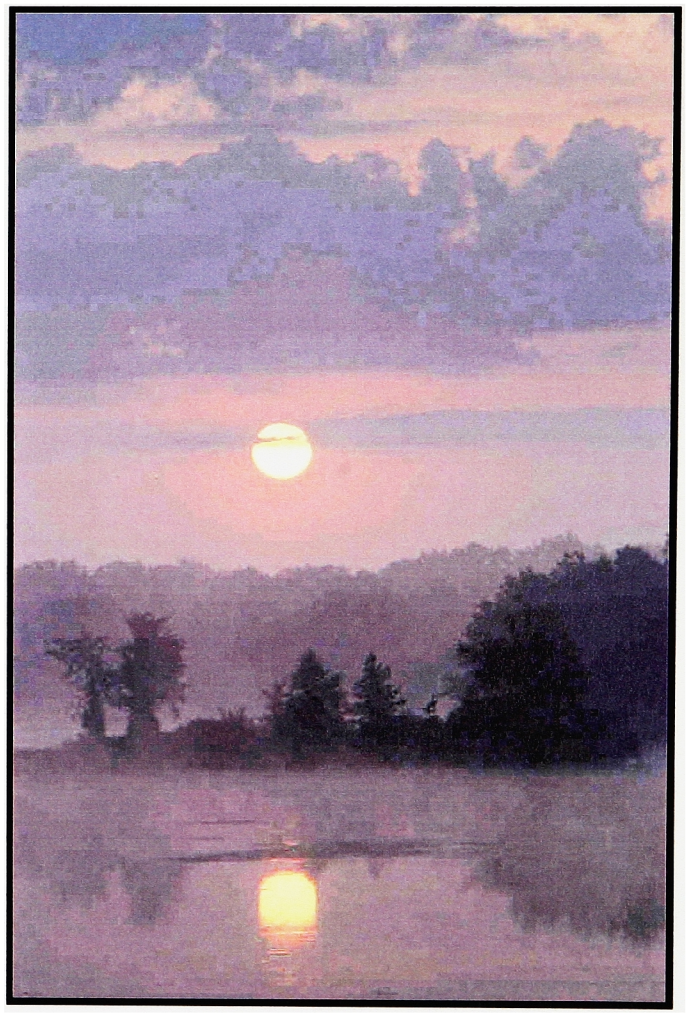
Original Size: 2,155KB Comp Factor: 9
Compressed Size: 31KB Absolute Comp: 70:1



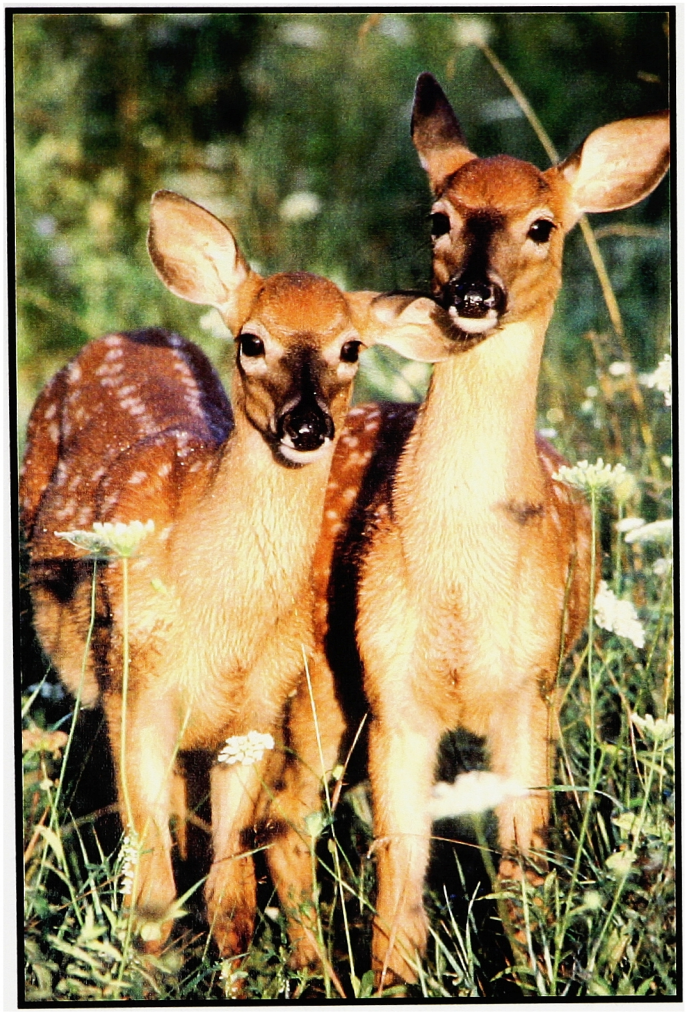
Original Size: 2,155KB Comp Factor: 12
Compressed Size: 53KB Absolute Comp: 41:1



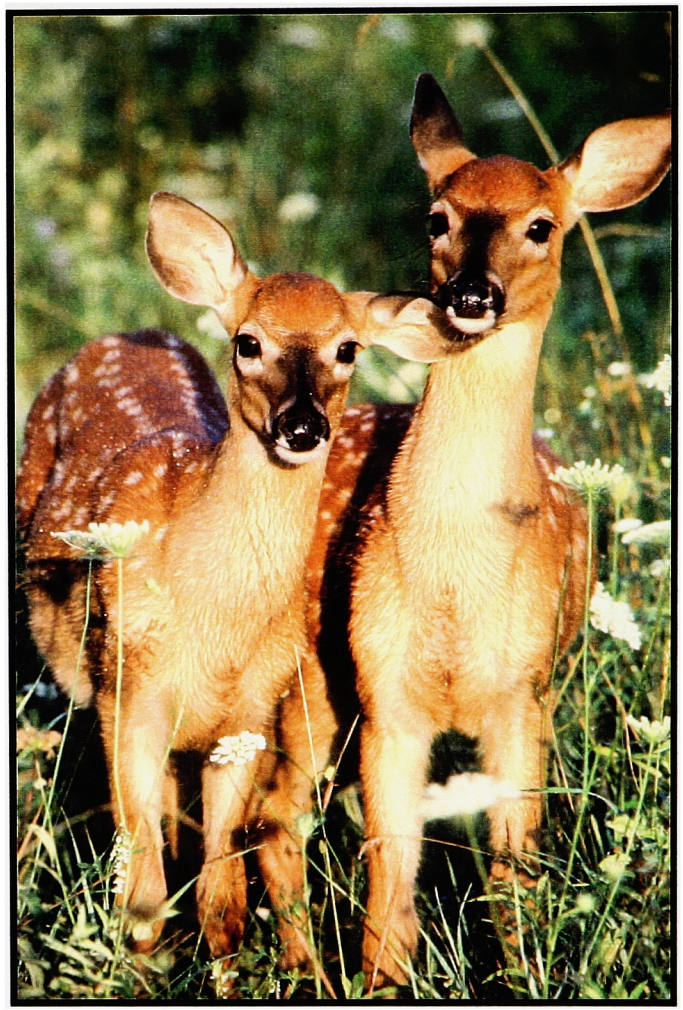
Original Size: 2,155KB Comp Factor: 3
Compressed Size: 22KB Absolute Comp: 98:1



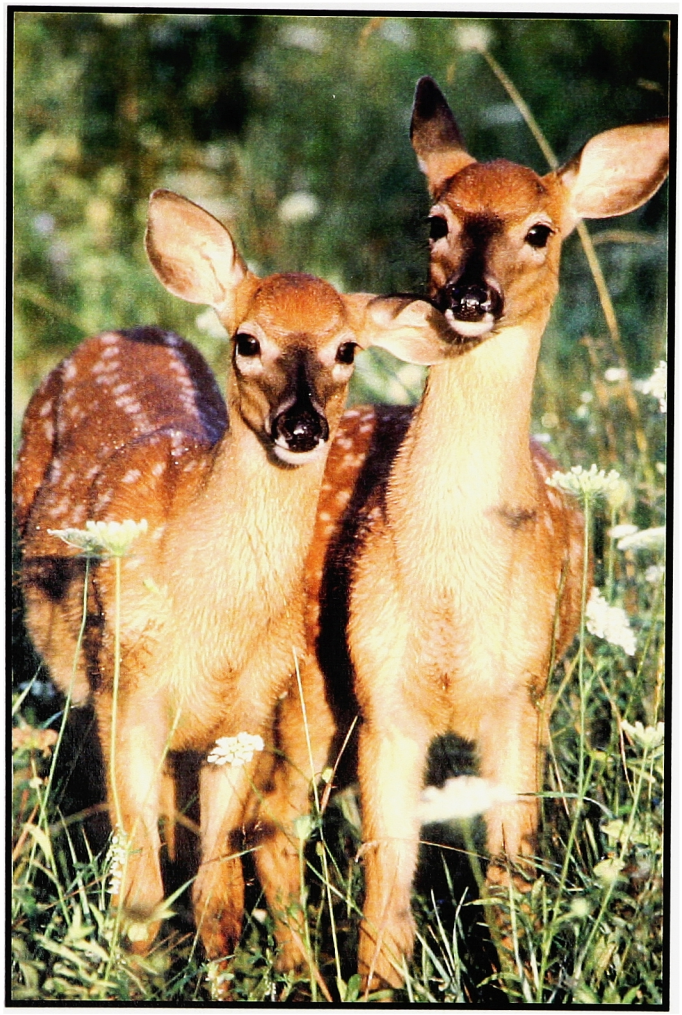
Original Size: 2,155KB Comp Factor: 6
Compressed Size: 24KB Absolute Comp: 90:1



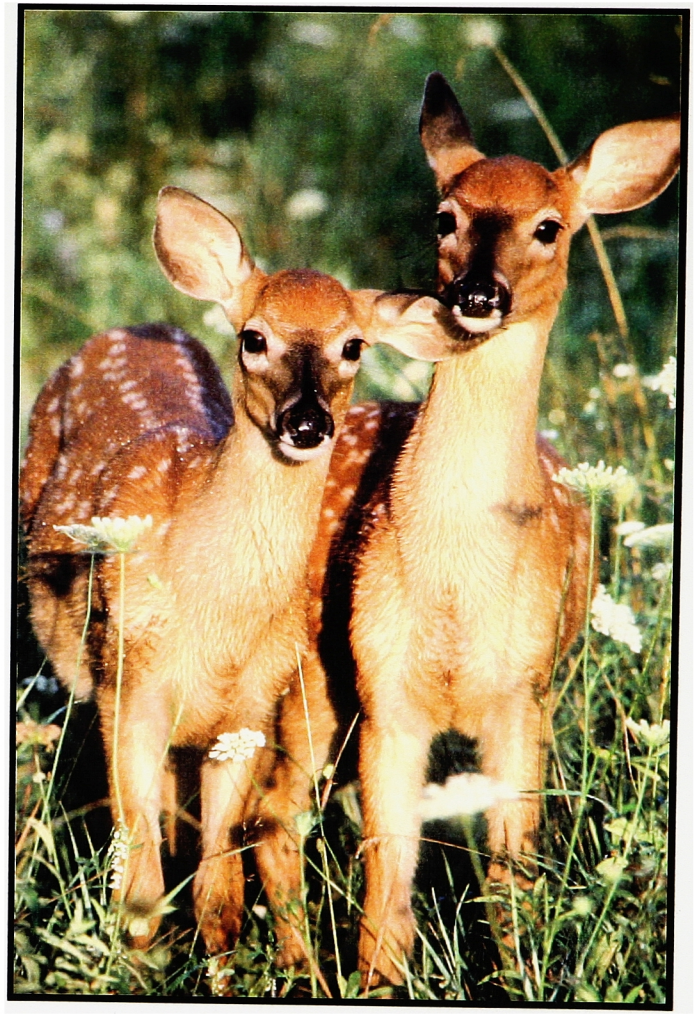
Original Size: 2,155KB Comp Factor: 21
Compressed Size: 439KB Absolute Comp: 5:1



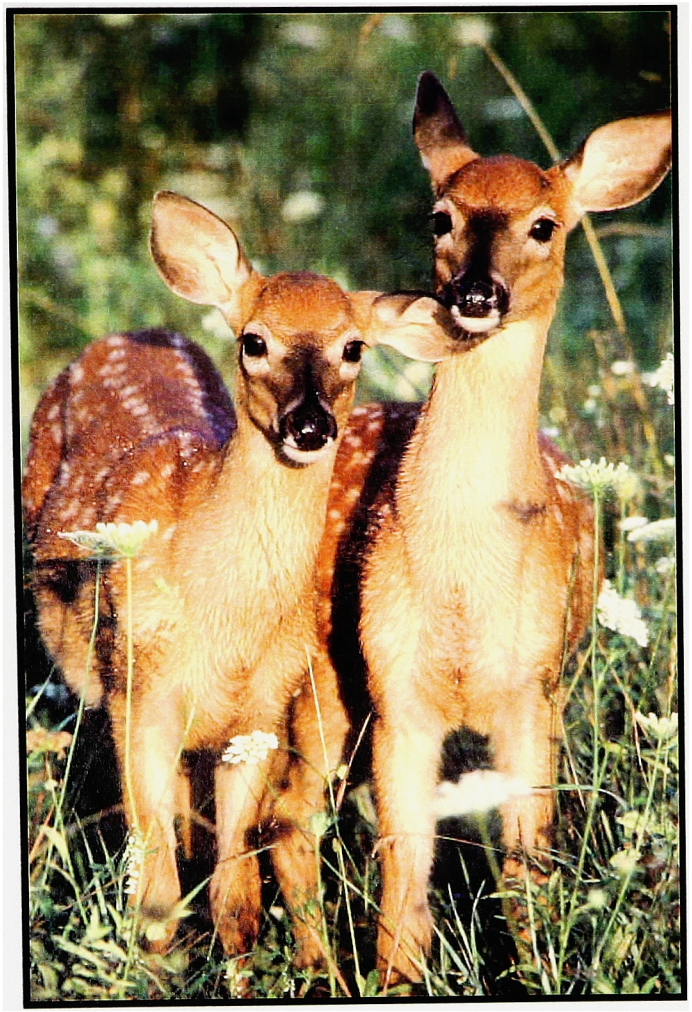
Original Size: 2,155KB Comp Factor: 24
Compressed Size: 703KB Absolute Comp: 3:1



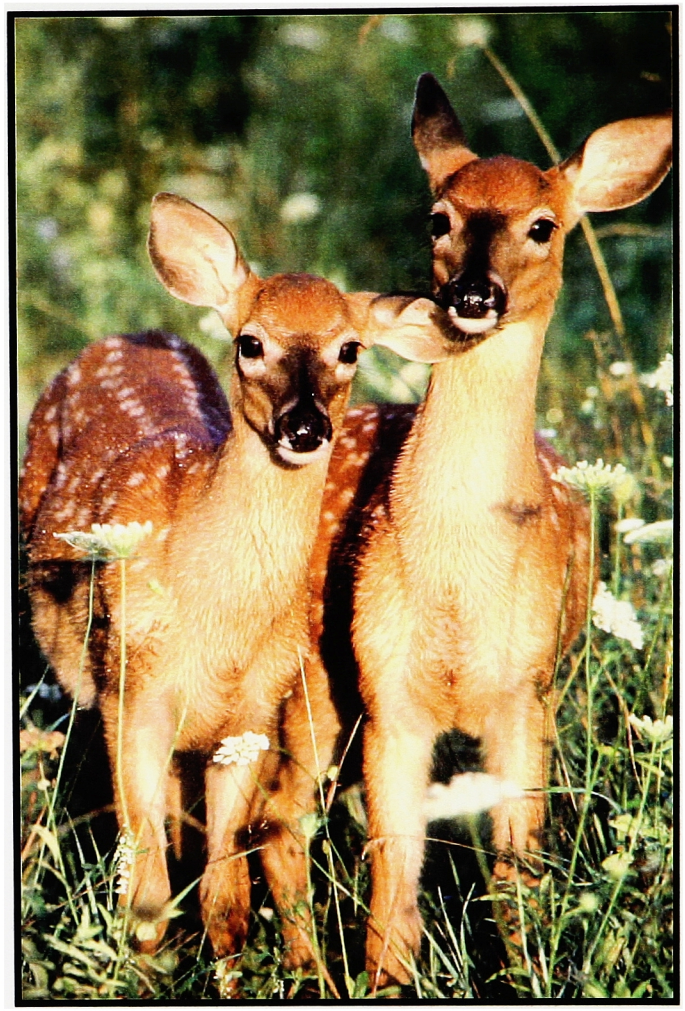
Original Size: 2,155KB Comp Factor: 15
Compressed Size: 194KB Absolute Comp: 11:1



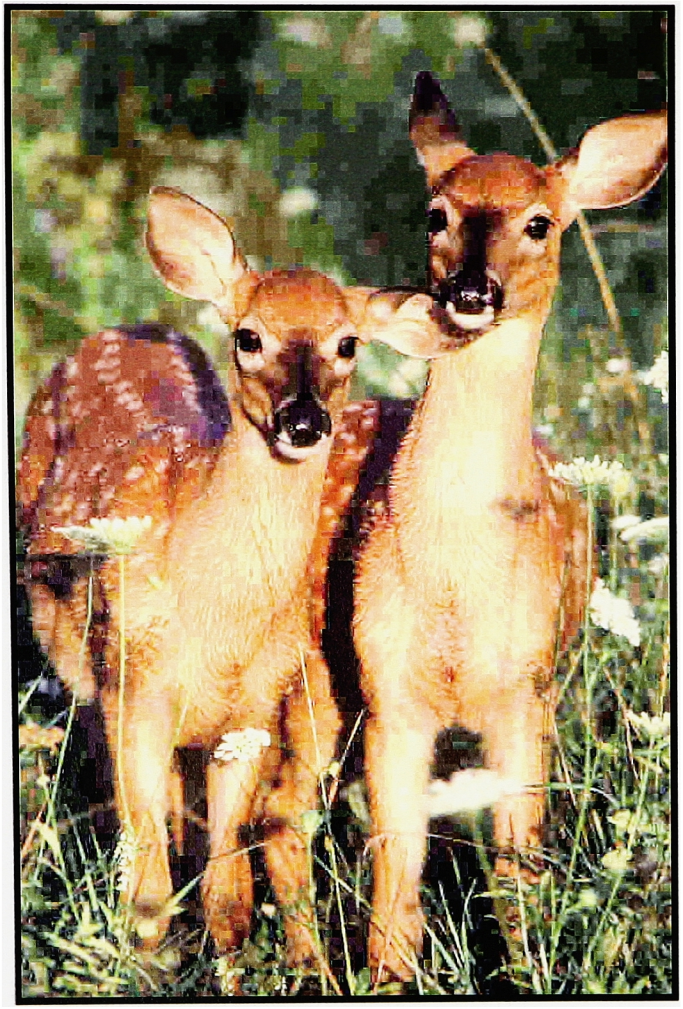
Original Size: 2,155KB Comp Factor: 18
Compressed Size: 292KB Absolute Comp: 7:1



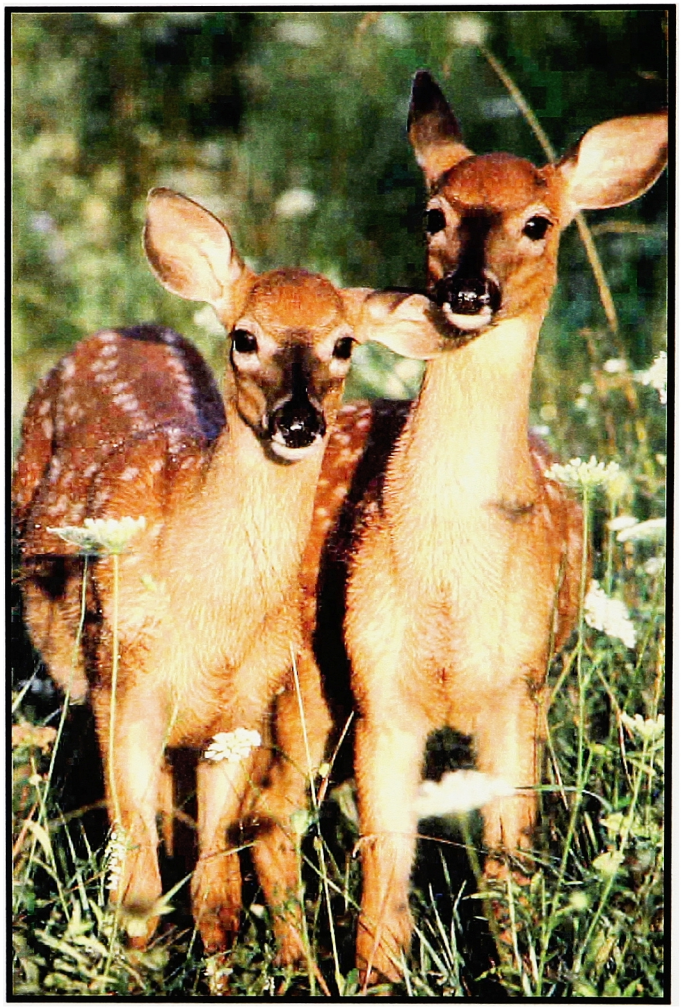
Original Size: 2,155KB
Compressed Size: 85KB
Comp Factor: 9
Absolute Comp: 25:1



Original Size: 2,155KB
Compressed Size: 129KB
Comp Factor: 12
Absolute Comp: 17:1



Original Size: 2,155KB
Compressed Size: 38KB
Comp Factor: 3
Absolute Comp: 57:1



Original Size: 2,155KB
Compressed Size: 56KB
Comp Factor: 6
Absolute Comp: 39:1

Original Size: 2,155KB Comp Factor: 21
Compressed Size: 639KB Absolute Comp: 3:1

Original Size: 2,155KB Comp Factor: 24
Compressed Size: 944KB Absolute Comp: 2:1

Original Size: 2,155KB Comp Factor: 18
Compressed Size: 431KB Absolute Comp: 5:1

Original Size: 2,155KB Comp Factor: 15
Compressed Size: 295KB Absolute Comp: 7:1

Original Size: 2,155KB Comp Factor: 9
Compressed Size: 133KB Absolute Comp: 16:1

Original Size: 2,155KB Comp Factor: 12
Compressed Size: 200KB Absolute Comp: 11:1

Original Size: 2,155KB Comp Factor: 3
Compressed Size: 63KB Absolute Comp: 34:1

Original Size: 2,155KB Comp Factor: 6
Compressed Size: 90KB Absolute Comp: 24:1

File: Sunrise Comp Factor: 21
Compressed Size: 320K Absolute Comp: 7:1

File: Sunrise Comp Factor: 15
Compressed Size: 101K Absolute Comp: 21:1

File: Sunrise Comp Factor: 3
Compressed Size: 22K Absolute Comp: 98:1

File: Sunrise Comp Factor: 9
Compressed Size: 31K Absolute Comp: 70:1

File: Fawns Comp Factor: 21
Compressed Size: 439K Absolute Comp: 5:1

File: Fawns Comp Factor: 15
Compressed Size: 194K Absolute Comp: 11:1

File: Fawns Comp Factor: 9
Compressed Size: 85K Absolute Comp: 25:1

File: Fawns Comp Factor: 3
Compressed Size: 38K Absolute Comp: 57:1

File: Web Comp Factor: 21
Compressed Size: 639K Absolute Comp: 3:1

File: Web Comp Factor: 15
Compressed Size: 295K Absolute Comp: 7:1

File: Web Comp Factor: 9
Compressed Size: 133K Absolute Comp: 16:1

File: Web Comp Factor: 3
Compressed Size: 63K Absolute Comp: 34:1

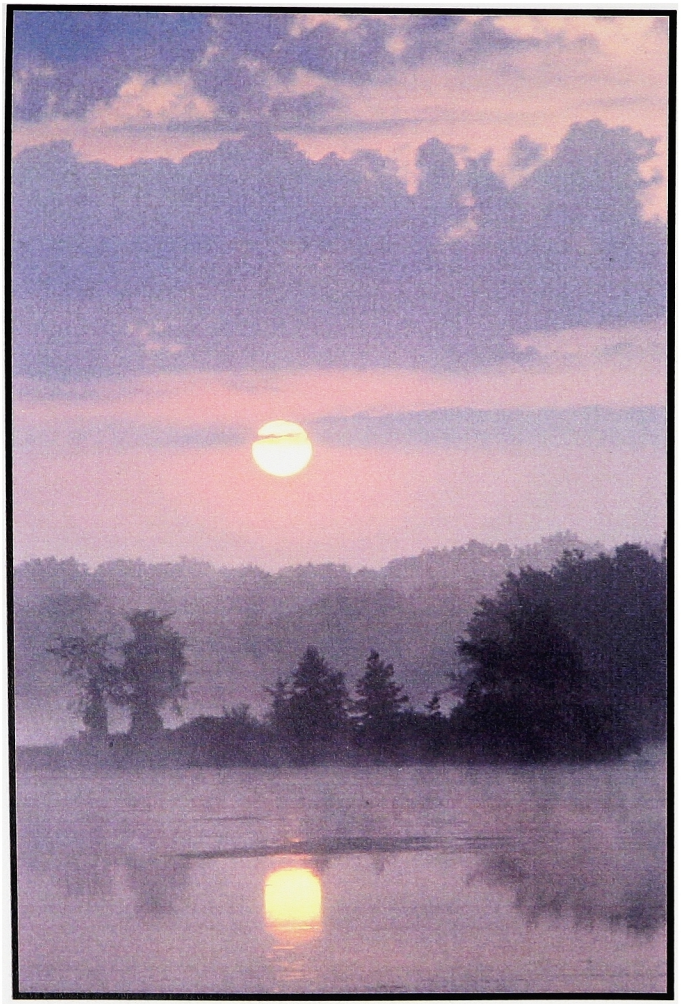
APPENDIX C — SUB-SAMPLING COMPARISONS

Images presented in this appendix represent the lowest frequency (Sunrise), medium frequency (Fawns), and highest frequency (Web) images utilized in the objective tests. These images have been compressed using four different compression ratios (21, 15, 9, and 3) and two different sub-sampling ratios (1:1 and 2:1). The layout is arranged so that images compressed with the same compression factor and different sub-sampling ratios are placed side by side for direct comparison. As is the case with images in Appendix B — Compression Comparisons, all viewing should be conducted under standard 5000 K lighting conditions.

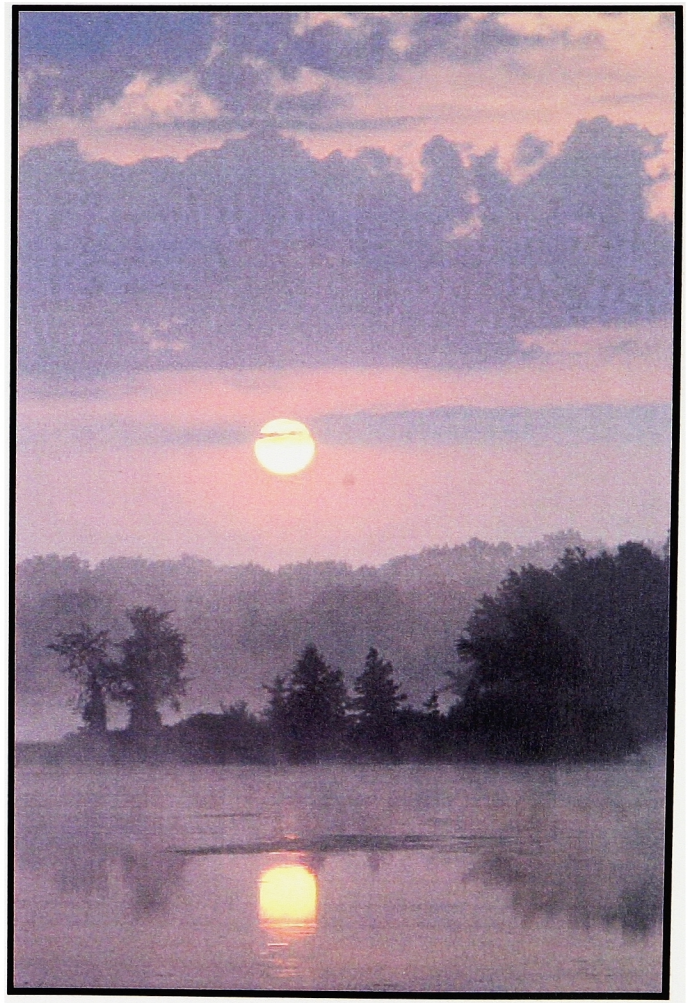
In addition to the color samples provided, histograms of the difference file for each image are included in pages C–14 through C-25. These difference files were created using *Adobe Photoshop*, as explained in Chapter 4 — Description of Software.

Comp Size: 320KB Comp Factor: 21
Sub-Sampling Ratio: 1:1 Absolute Comp: 7:1

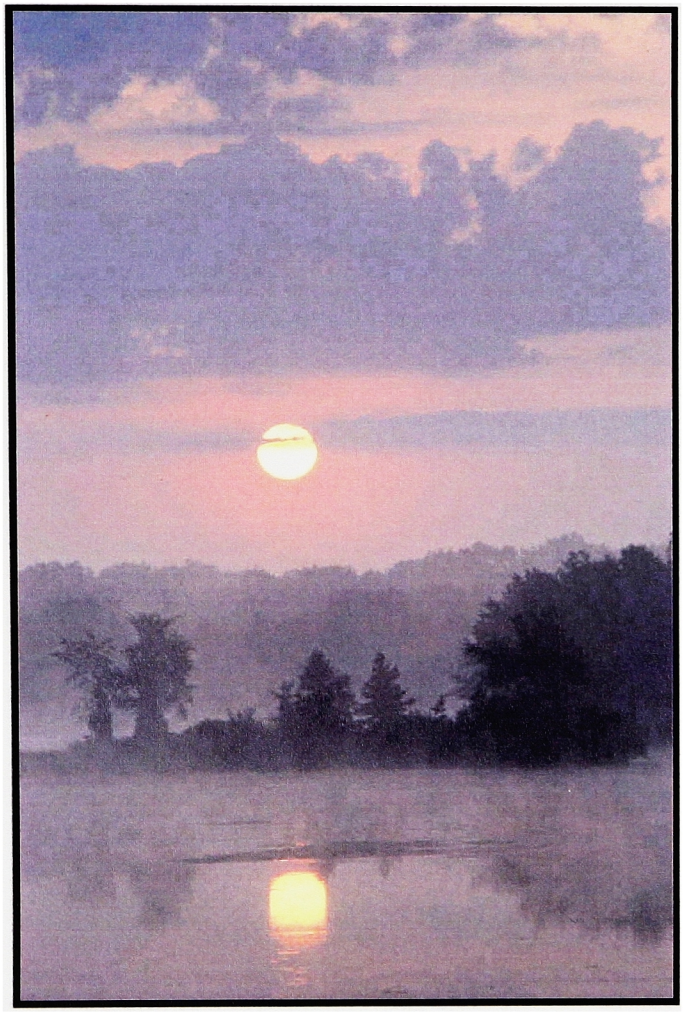
Comp Size: 223KB Comp Factor: 21
Sub-Sampling Ratio 2:1 Absolute Comp: 10:1



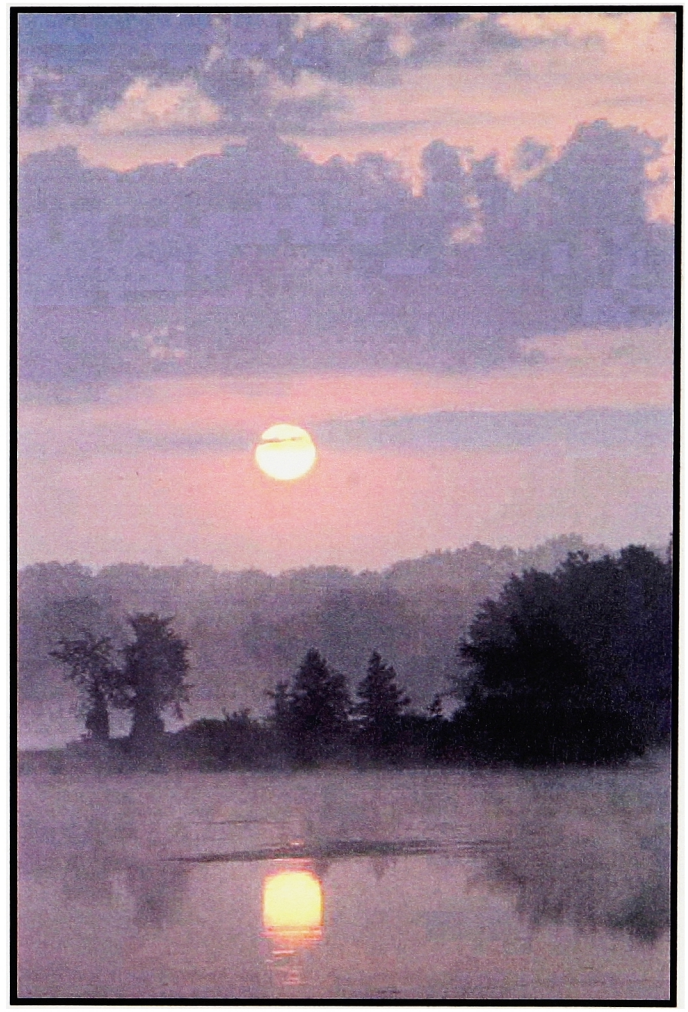
Comp Size: 101KB Comp Factor: 15
Sub-Sampling Ratio: 1:1 Absolute Comp: 21:1



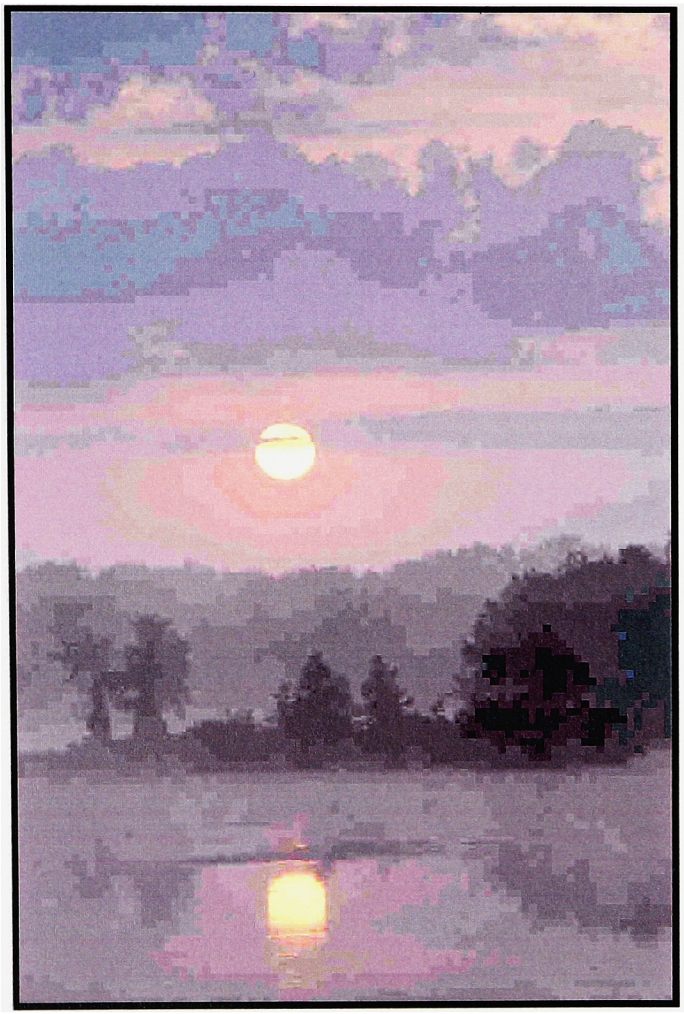
Comp Size: 75KB Comp Factor: 15
Sub-Sampling Ratio: 2:1 Absolute Comp: 29:1



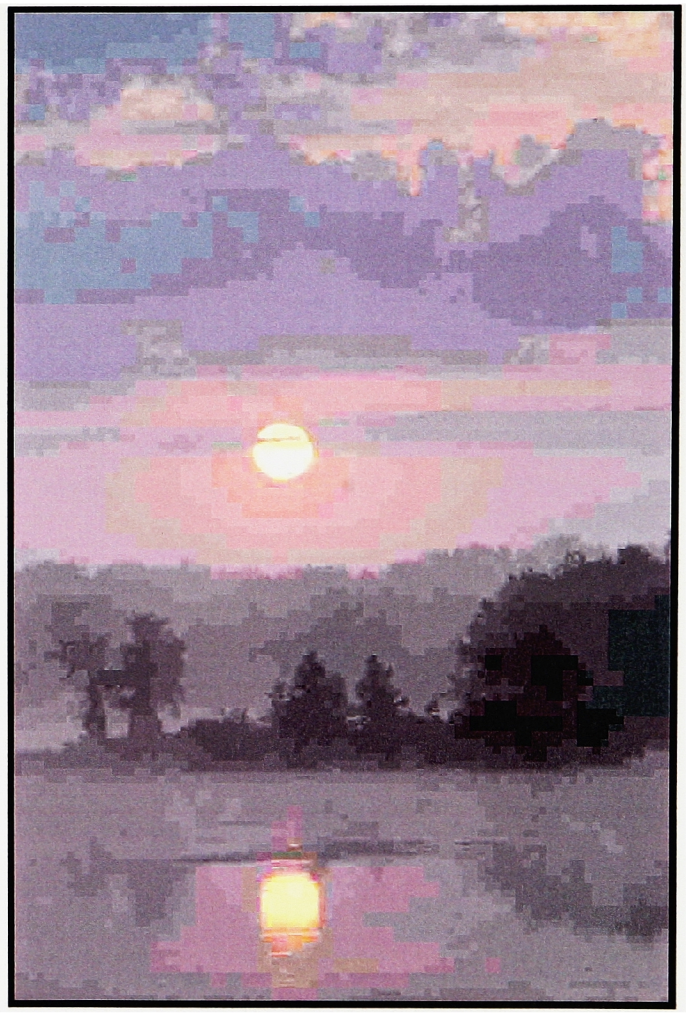
Comp Size: 31KB Comp Factor: 9
Sub-Sampling Ratio: 1:1 Absolute Comp: 70:1



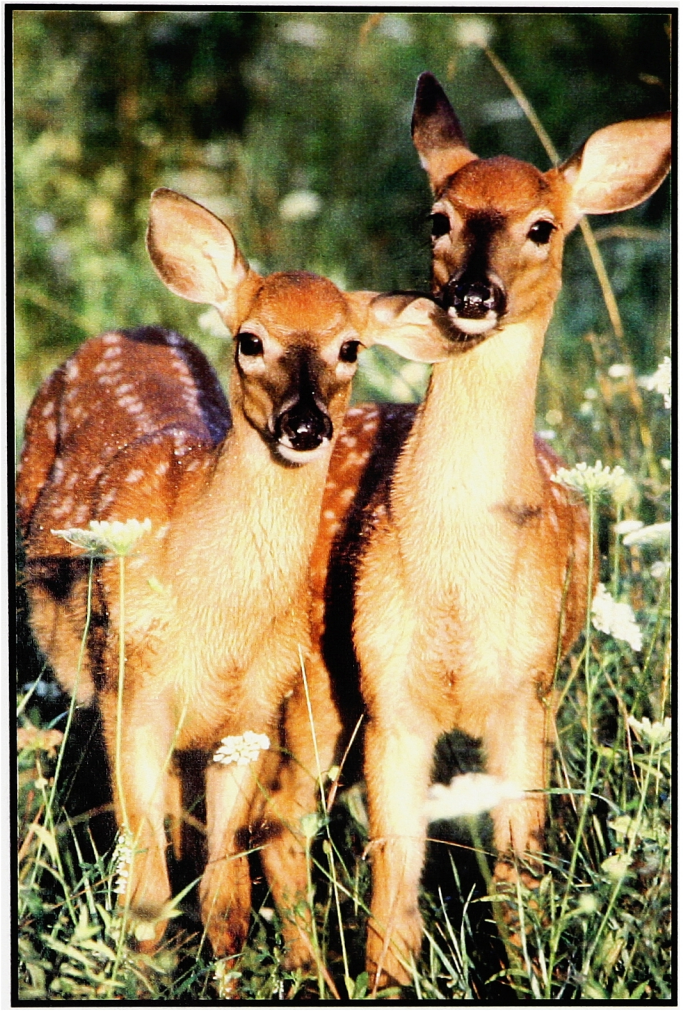
Comp Size: 22KB Comp Factor: 9
Sub-Sampling Ratio 2:1 Absolute Comp: 98:1



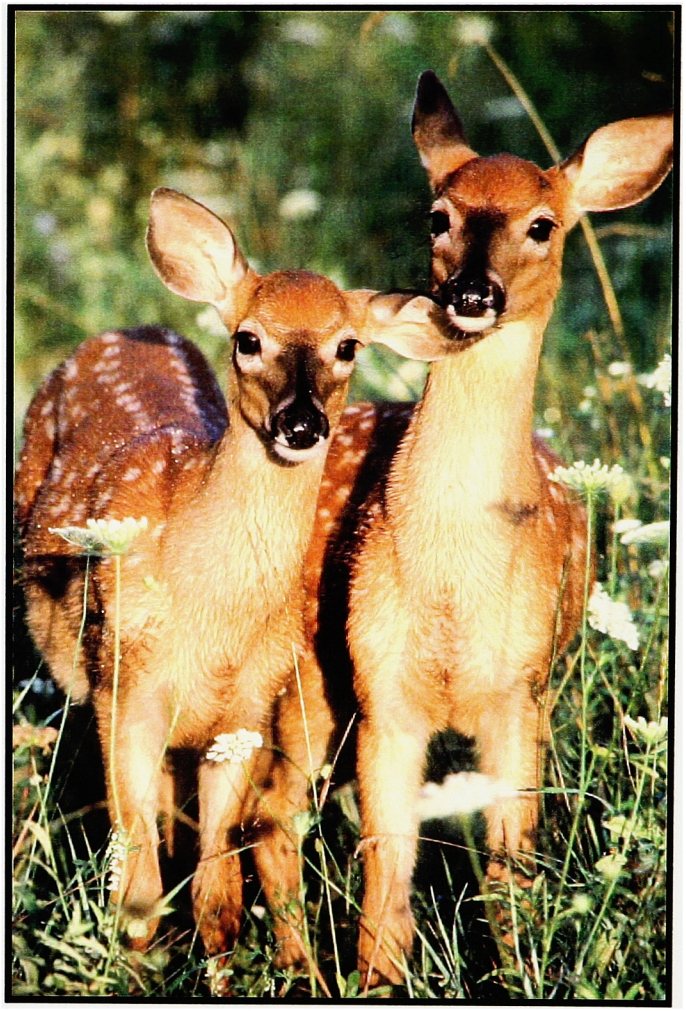
Comp Size: 22KB Comp Factor: 3
Sub-Sampling Ratio: 1:1 Absolute Comp: 98:1



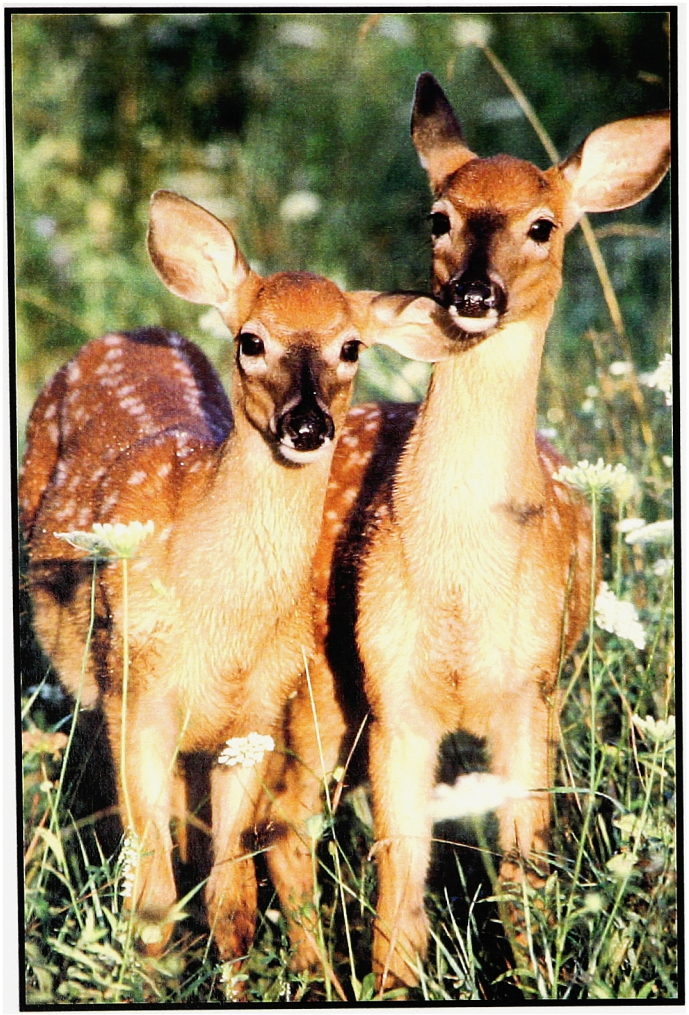
Comp Size: 13KB Comp Factor: 3
Sub-Sampling Ratio: 2:1 Absolute Comp: 166:1



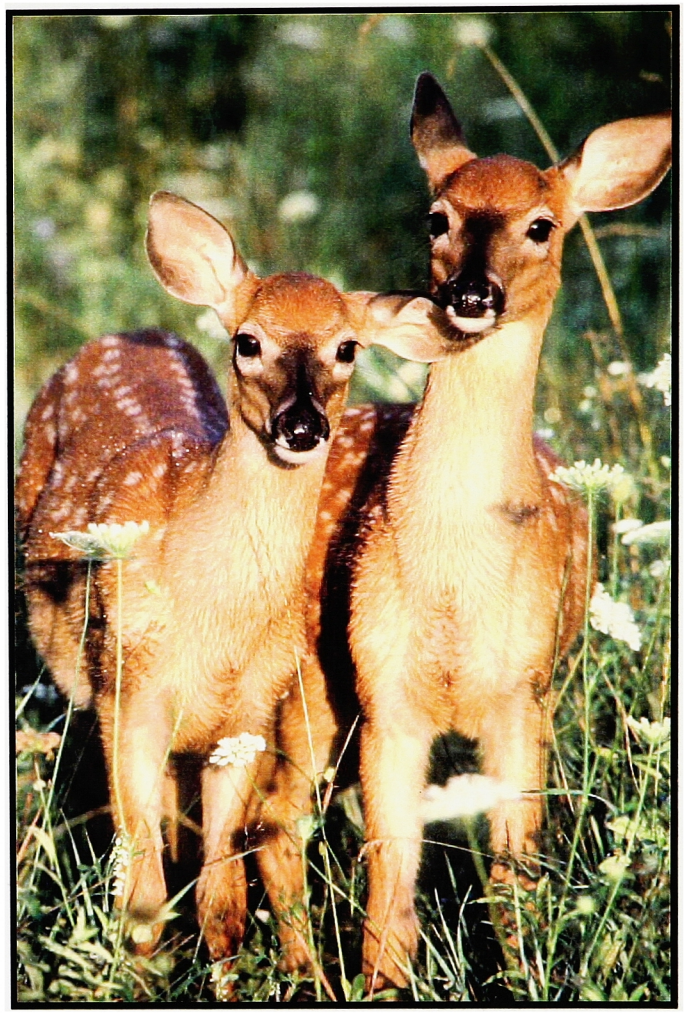
Comp Size: 439KB Comp Factor: 21
Sub-Sampling Ratio: 1:1 Absolute Comp: 5:1



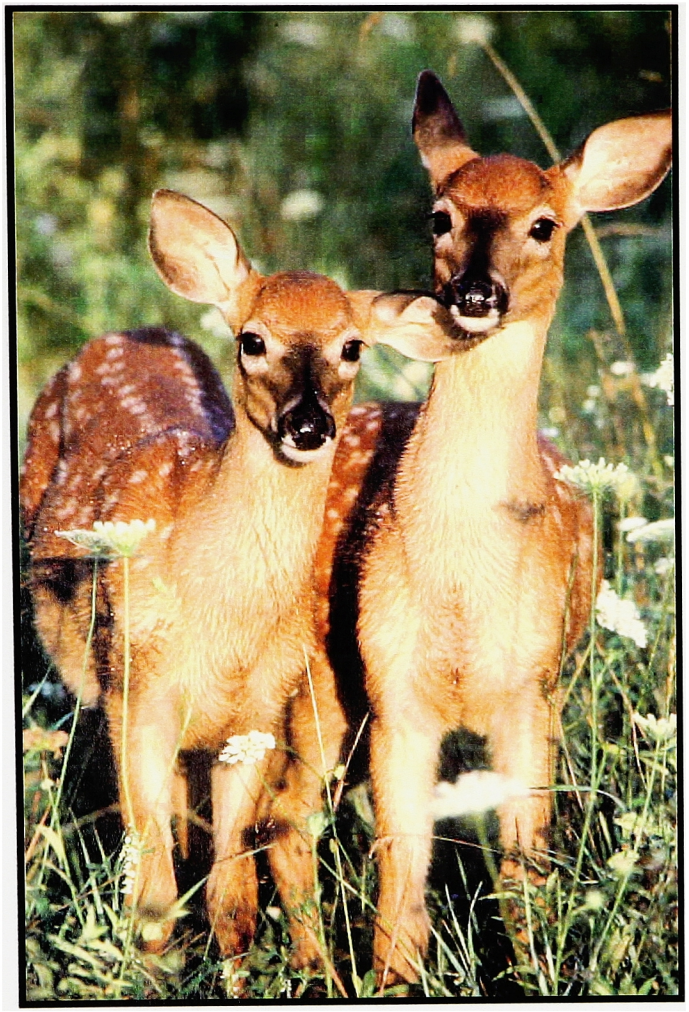
Comp Size: 325KB Comp Factor: 21
Sub-Sampling Ratio 2:1 Absolute Comp: 7:1



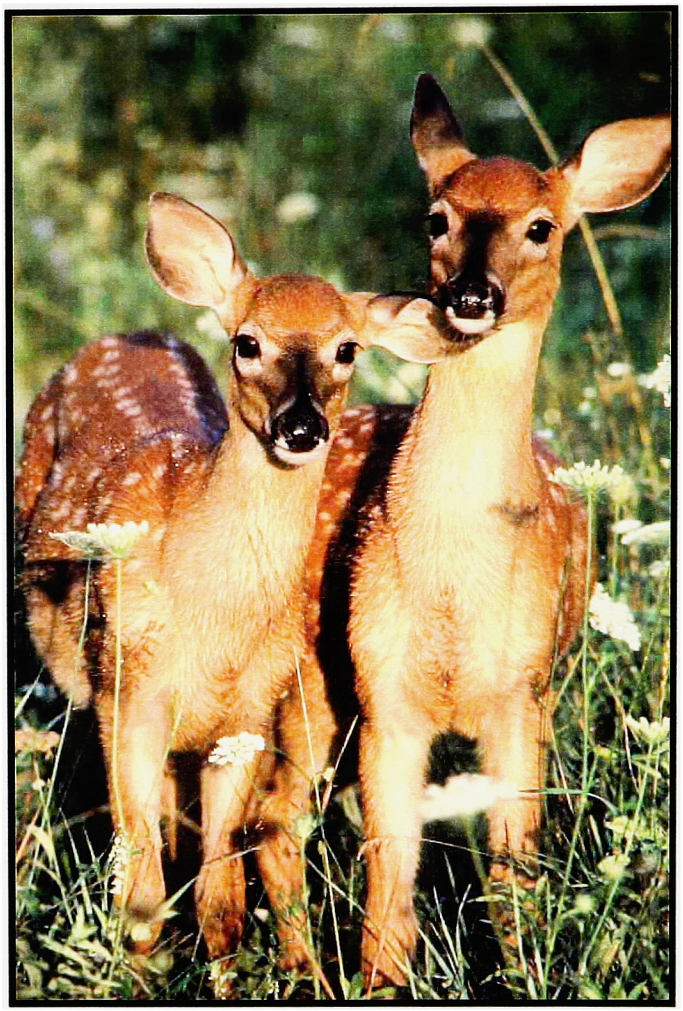
Comp Size: 194KB Comp Factor: 15
Sub-Sampling Ratio: 1:1 Absolute Comp: 11:1



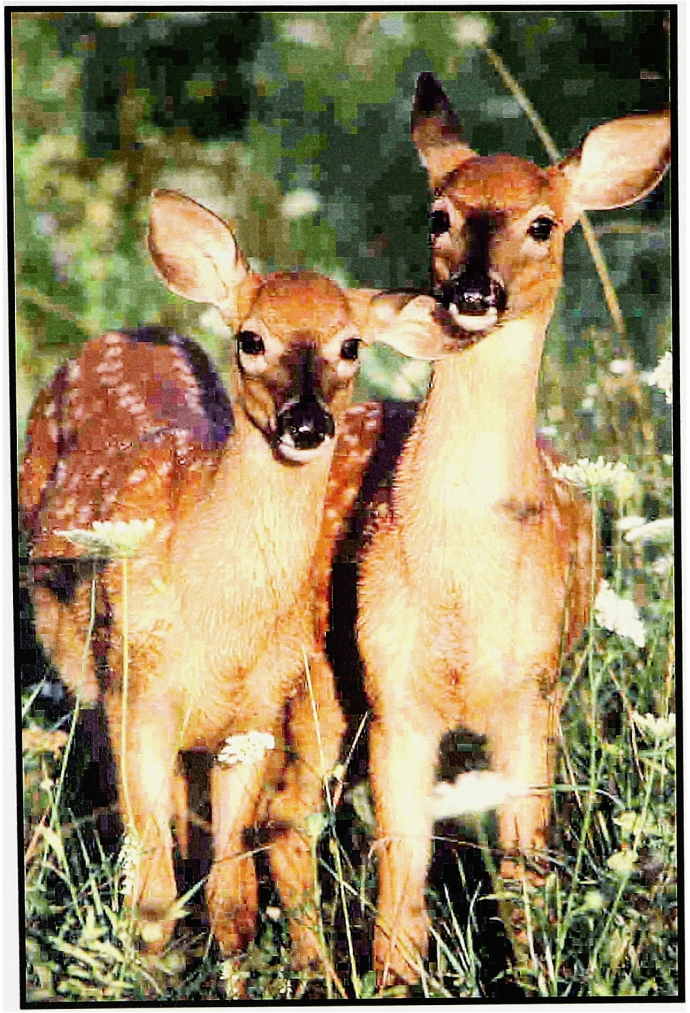
Comp Size: 146KB Comp Factor: 15
Sub-Sampling Ratio: 2:1 Absolute Comp: 15:1



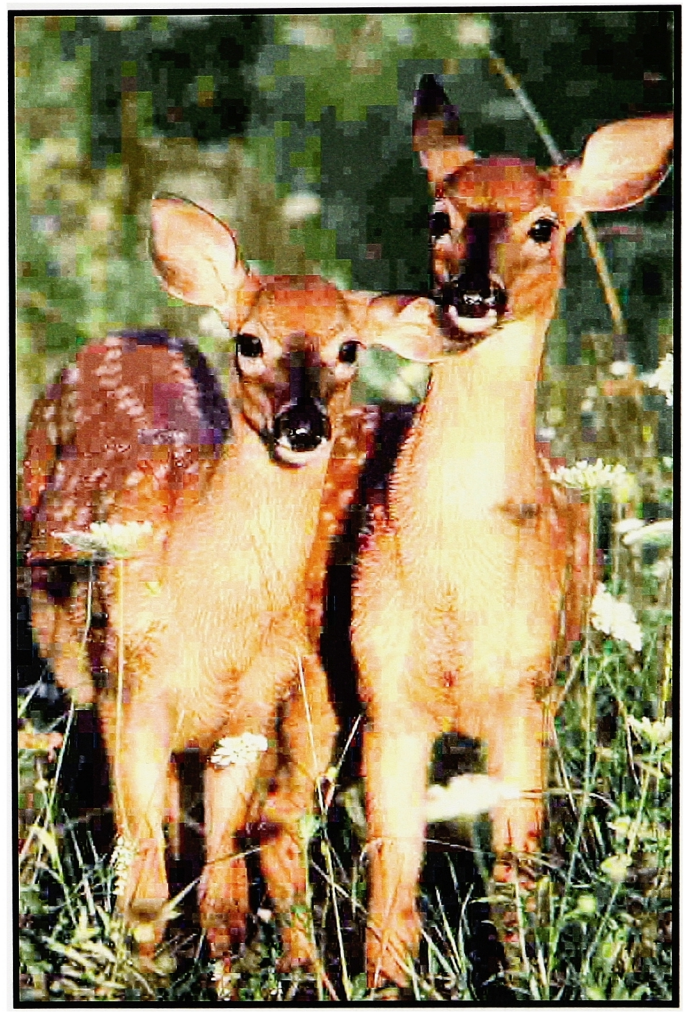
Comp Size: 85KB Comp Factor: 9
Sub-Sampling Ratio: 1:1 Absolute Comp: 25:1



Comp Size: 66KB Comp Factor: 9
Sub-Sampling Ratio 2:1 Absolute Comp: 33:1



Comp Size: 38KB Comp Factor: 3
Sub-Sampling Ratio: 1:1 Absolute Comp: 57:1



Comp Size: 28KB Comp Factor: 3
Sub-Sampling Ratio: 2:1 Absolute Comp: 77:1

Comp Size: 639KB Comp Factor: 21
Sub-Sampling Ratio: 1:1 Absolute Comp: 3:1

Comp Size: 500KB Comp Factor: 21
Sub-Sampling Ratio 2:1 Absolute Comp: 4:1

Comp Size: 245KB Comp Factor: 15
Sub-Sampling Ratio: 2:1 Absolute Comp: 9:1

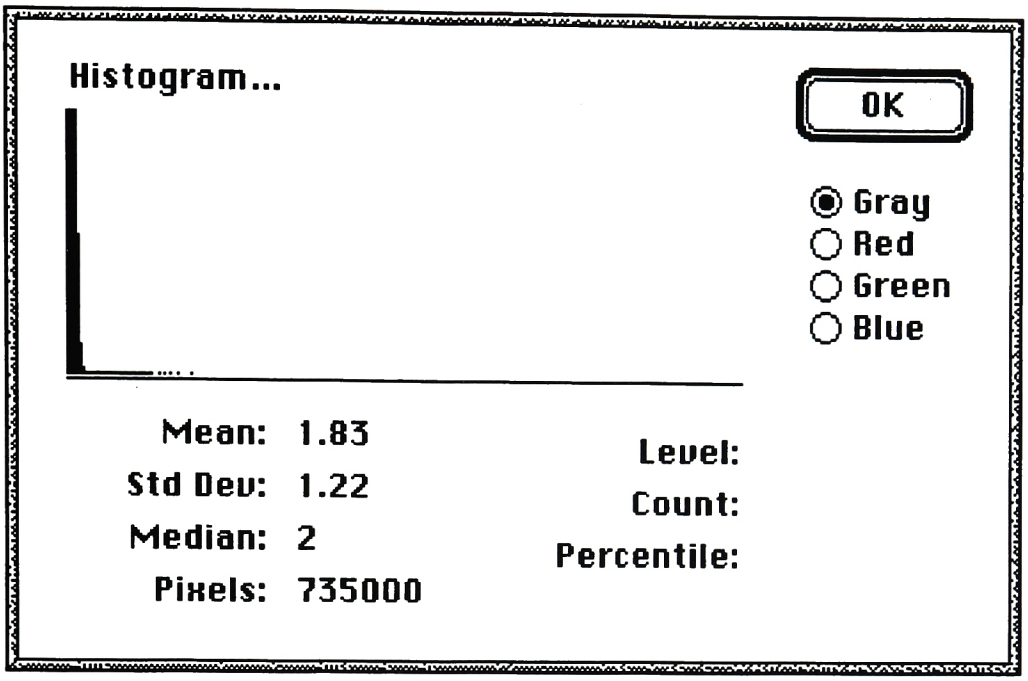
Comp Size: 295KB Comp Factor: 15
Sub-Sampling Ratio: 1:1 Absolute Comp: 7:1

Comp Size: 118KB Comp Factor: 9
Sub-Sampling Ratio 2:1 Absolute Comp: 18:1

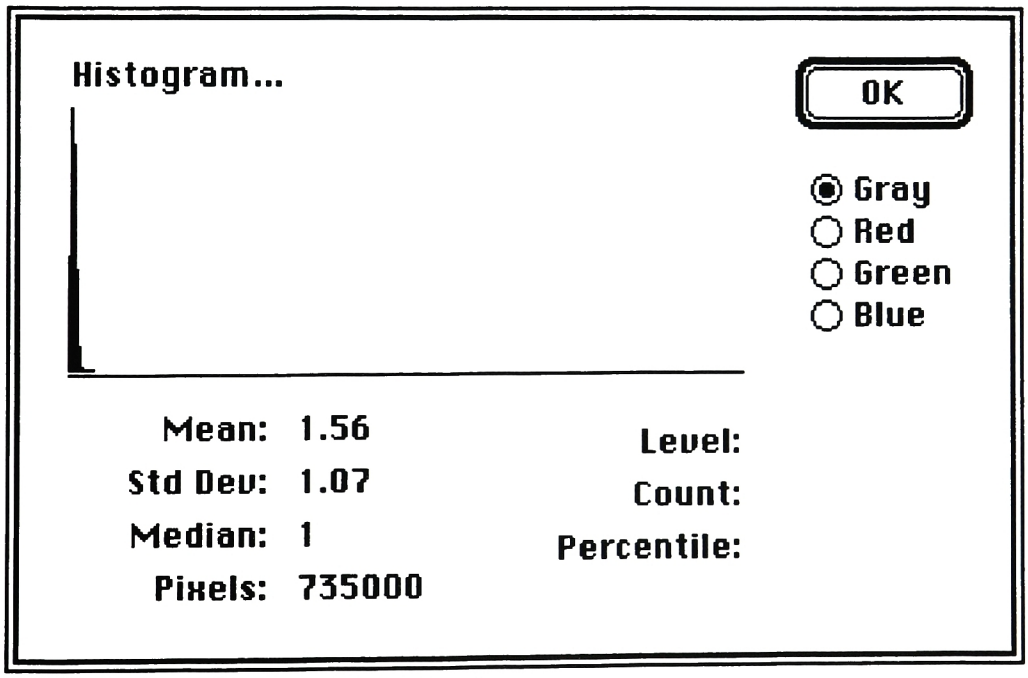
Comp Size: 133KB Comp Factor: 9
Sub-Sampling Ratio: 1:1 Absolute Comp: 16:1

Comp Size: 53KB Comp Factor: 3
Sub-Sampling Ratio: 2:1 Absolute Comp: 41:1

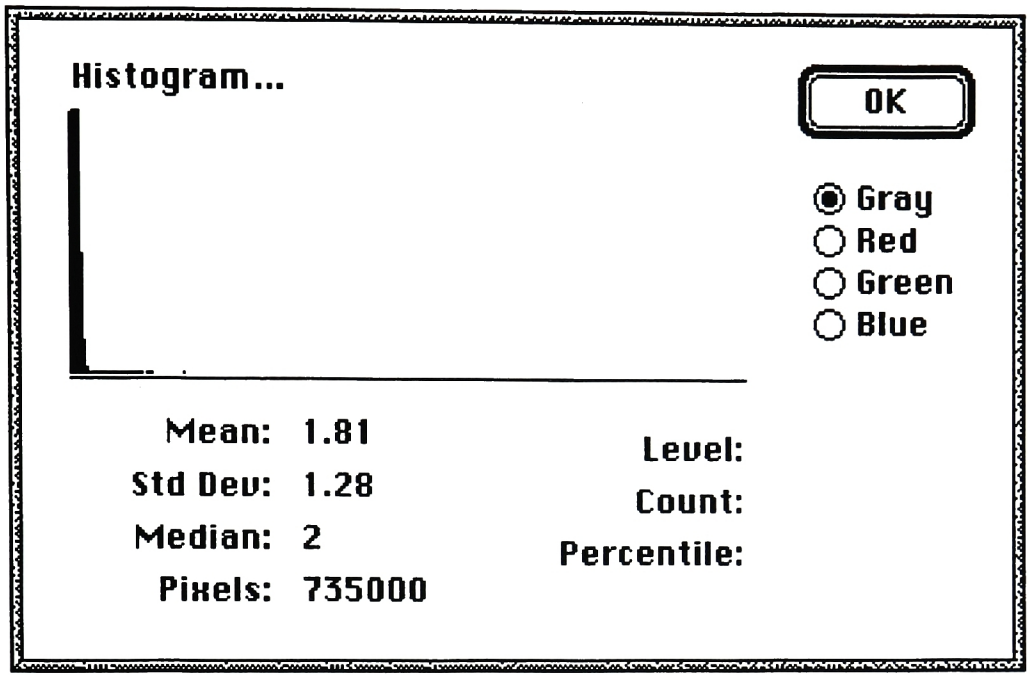
Comp Size: 63KB Comp Factor: 3
Sub-Sampling Ratio: 1:1 Absolute Comp: 34:1



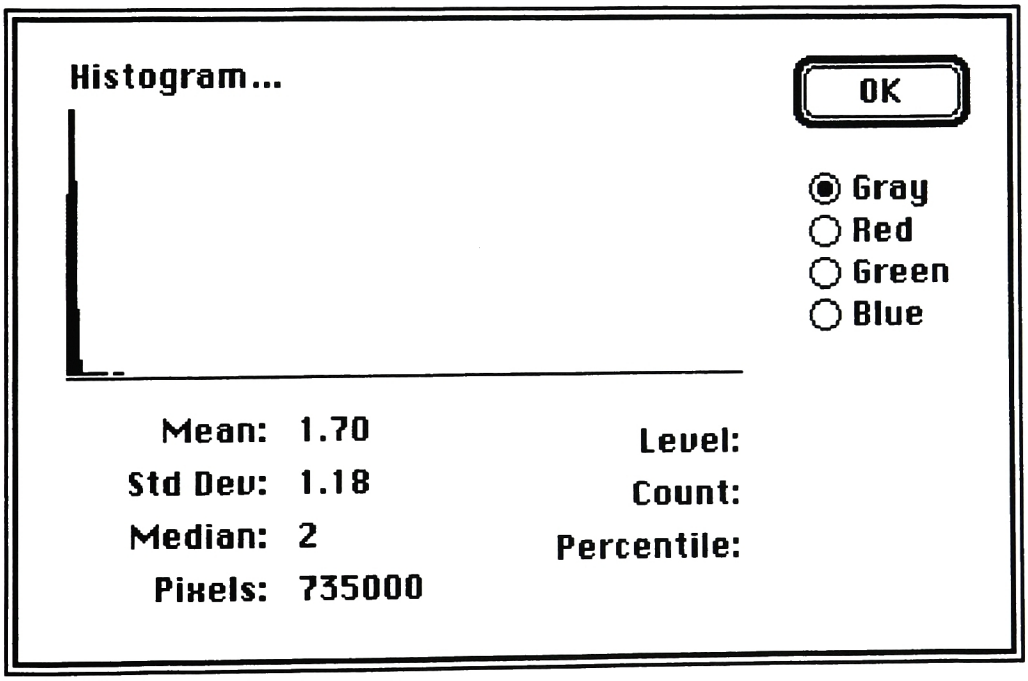
File:	Sunrise	Comp Factor:	21
Sub-Sampling Ratio:	2:1	Absolute Comp:	10:1



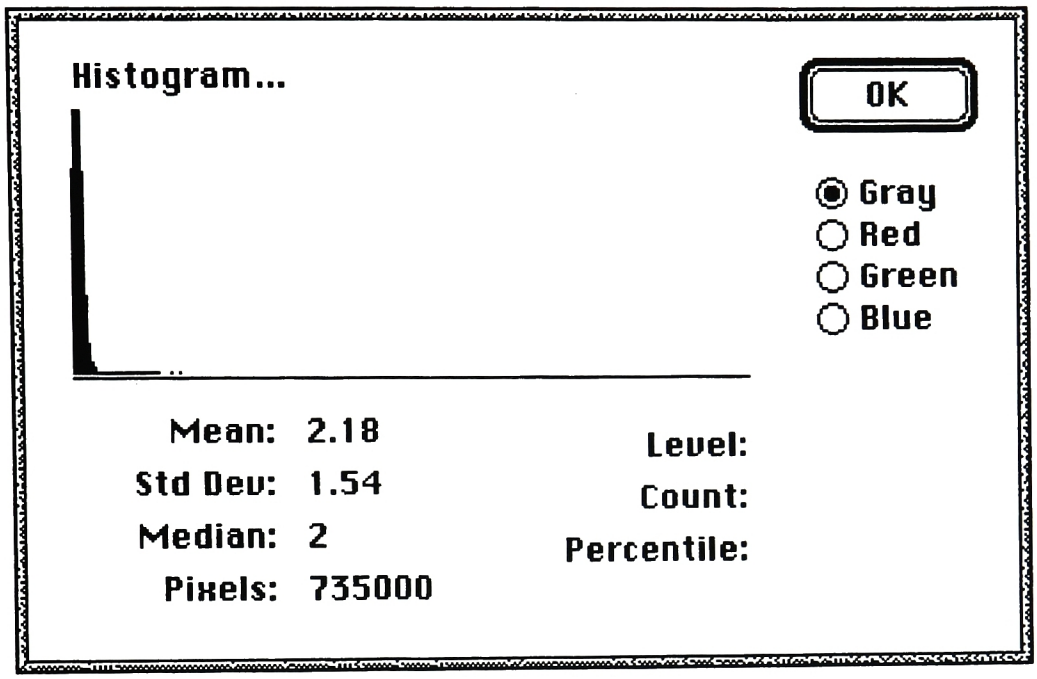
File:	Sunrise	Comp Factor:	21
Sub-Sampling Ratio:	1:1	Absolute Comp:	7:1



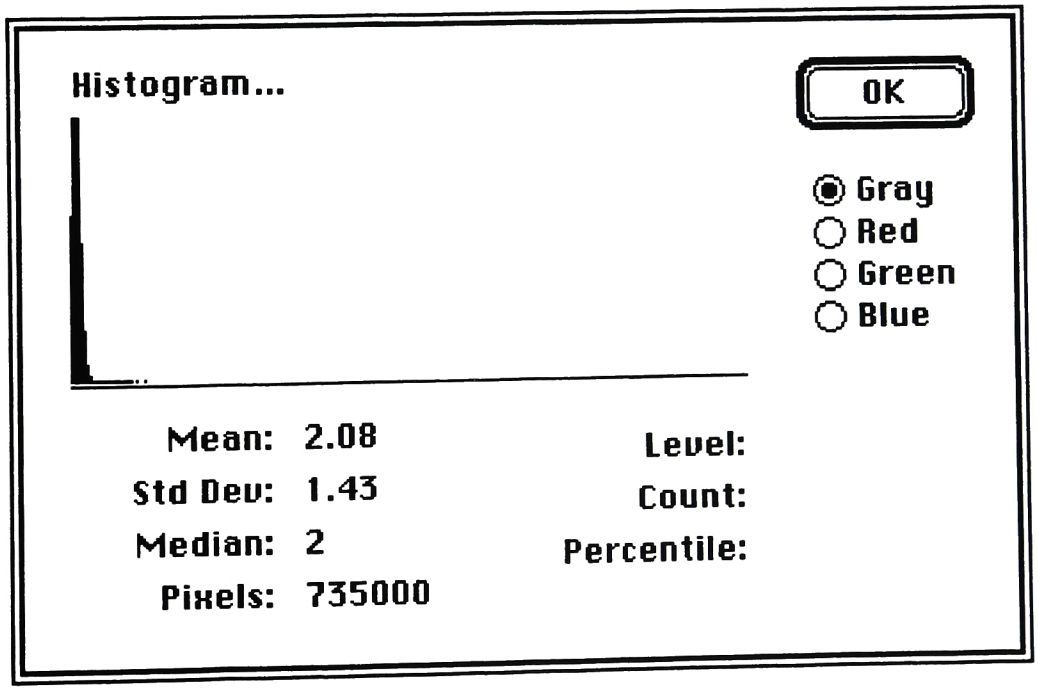
File:	Sunrise	Comp Factor:	15
Sub-Sampling Ratio:	2:1	Absolute Comp:	29:1



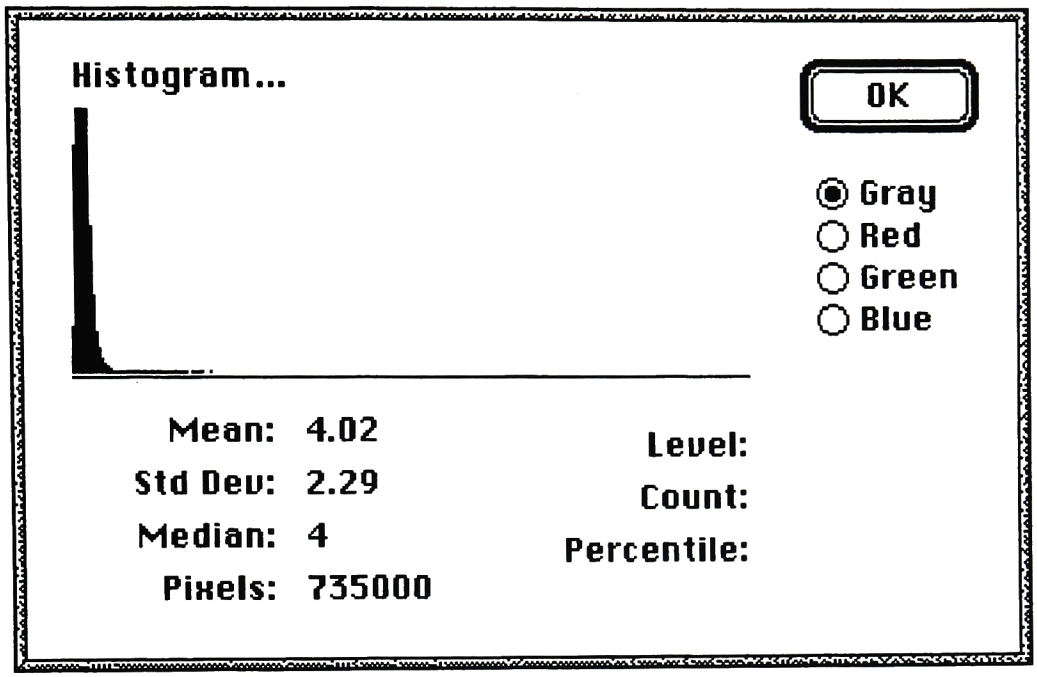
File:	Sunrise	Comp Factor:	15
Sub-Sampling Ratio:	1:1	Absolute Comp:	21:1



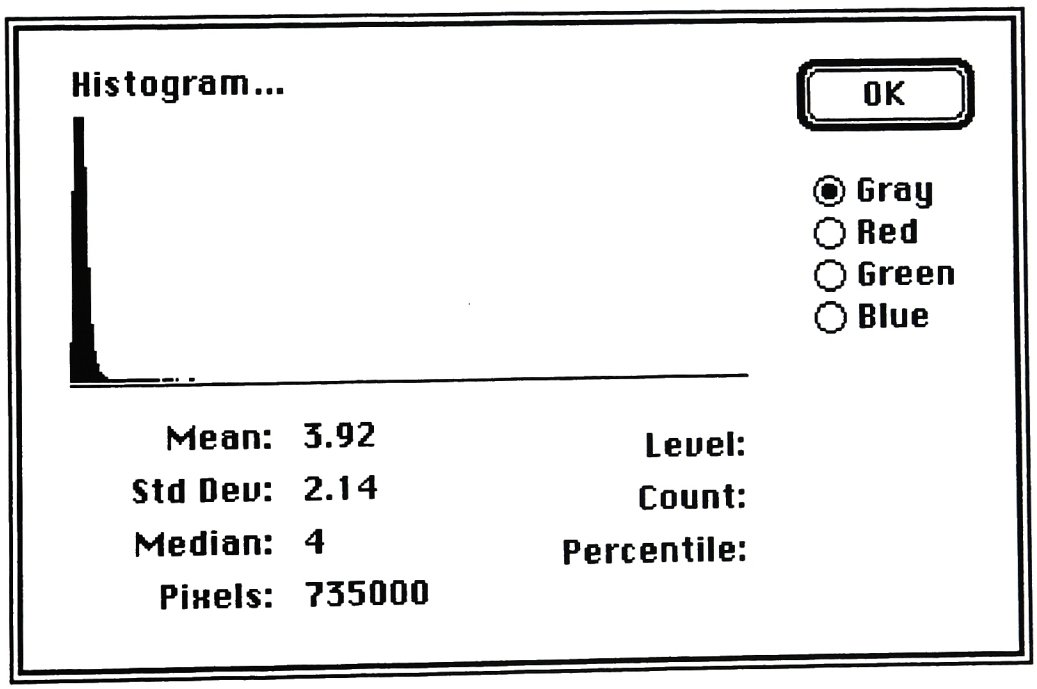
File: Sunrise
Sub-Sampling Ratio: 2:1
Comp Factor: 9
Absolute Comp: 98:1



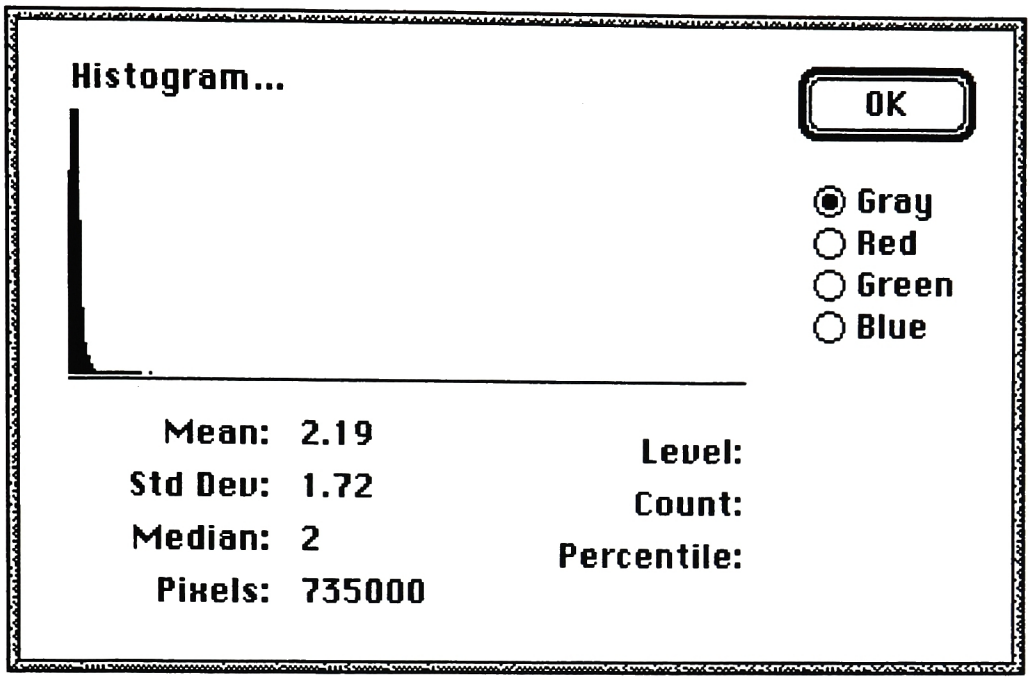
File: Sunrise
Sub-Sampling Ratio: 1:1
Comp Factor: 9
Absolute Comp: 70:1



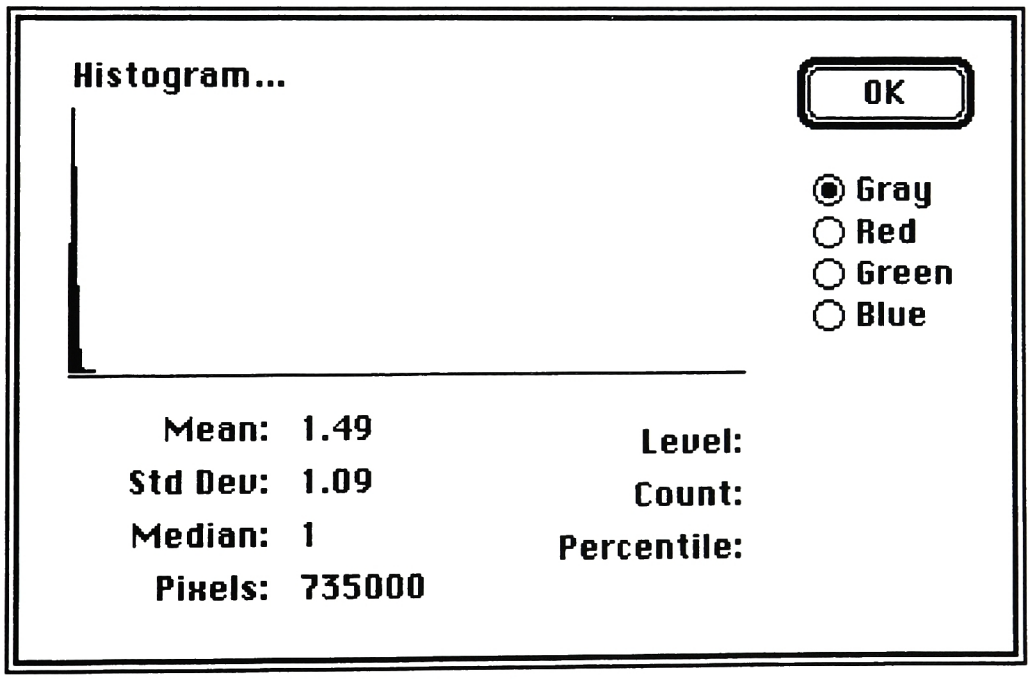
File:	Sunrise	Comp Factor:	3
Sub-Sampling Ratio:	2:1	Absolute Comp:	166:1



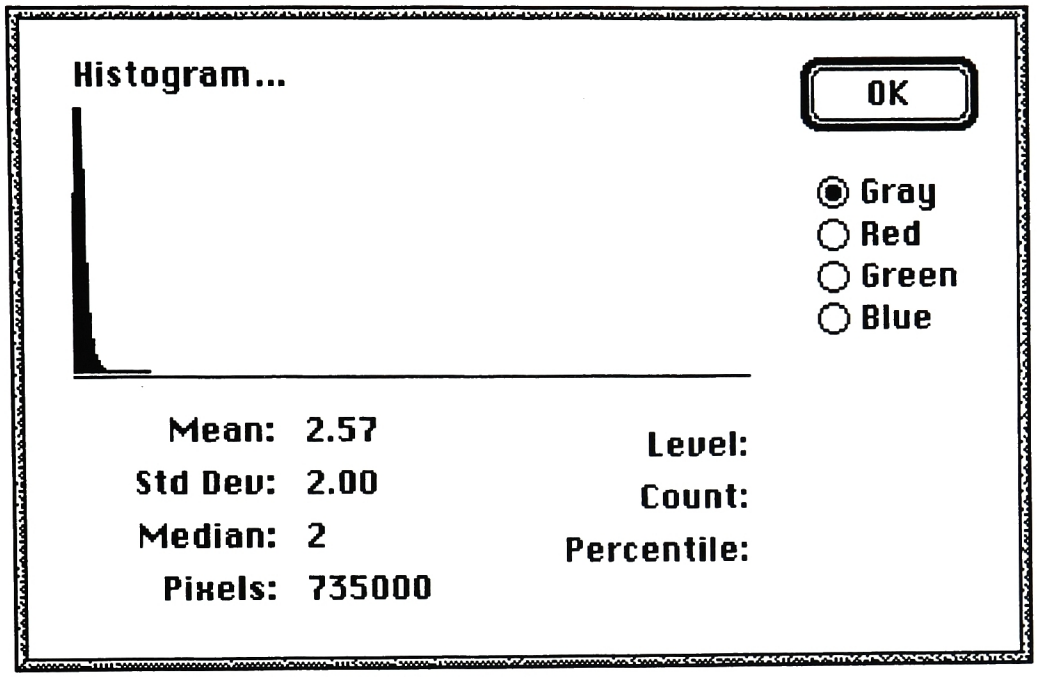
File:	Sunrise	Comp Factor:	3
Sub-Sampling Ratio:	1:1	Absolute Comp:	98:1



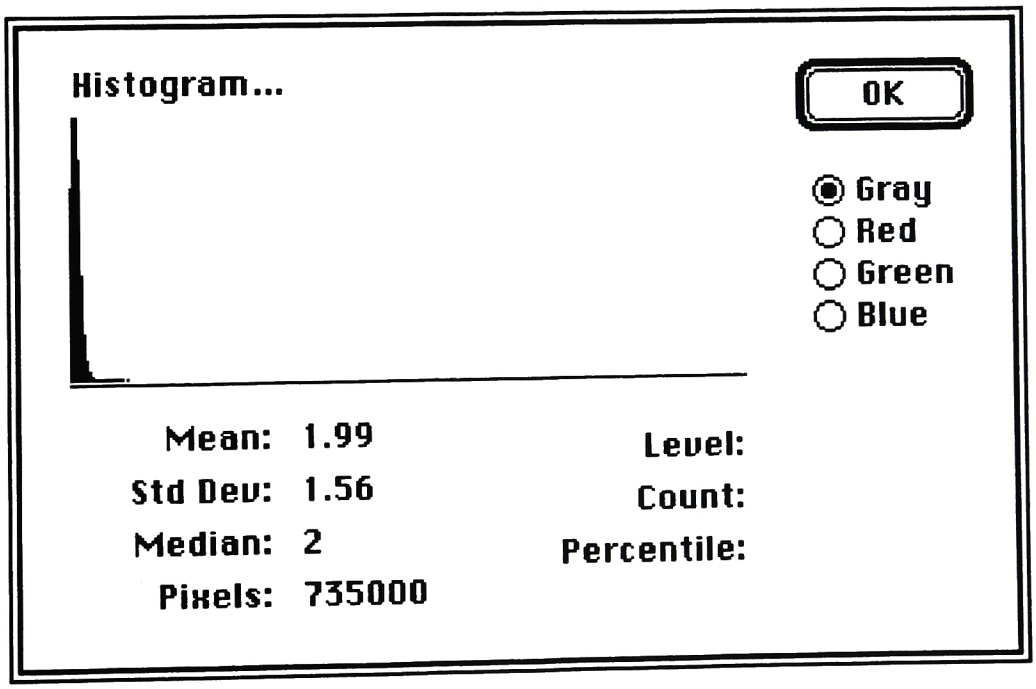
File:	Fawns	Comp Factor:	21
Sub-Sampling Ratio:	2:1	Absolute Comp:	7:1



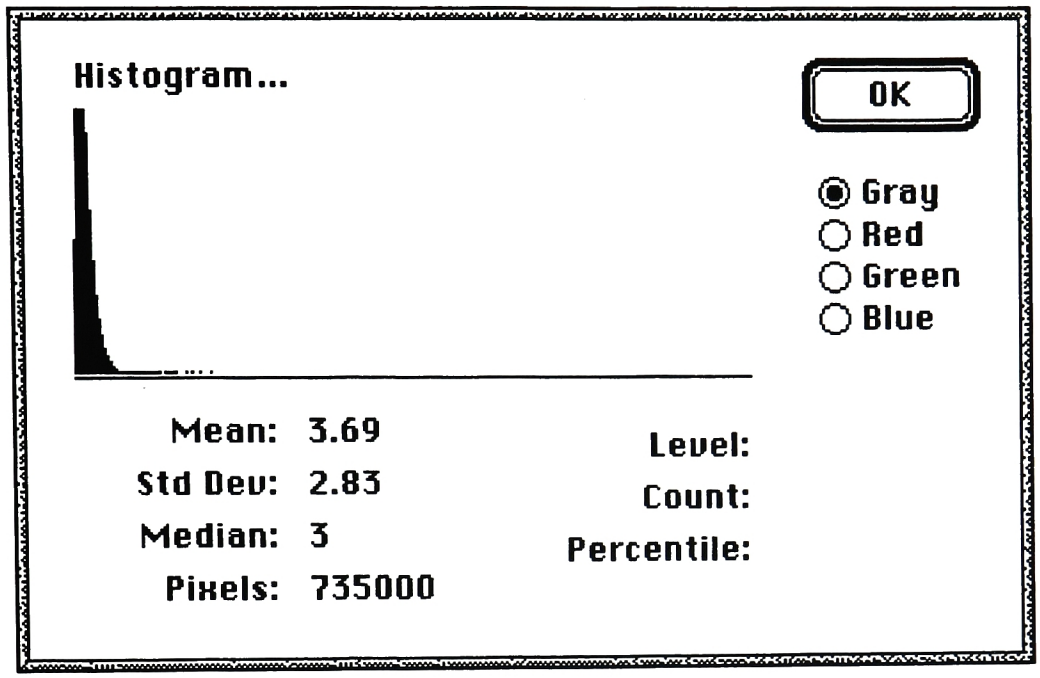
File:	Fawns	Comp Factor:	21
Sub-Sampling Ratio:	1:1	Absolute Comp:	5:1



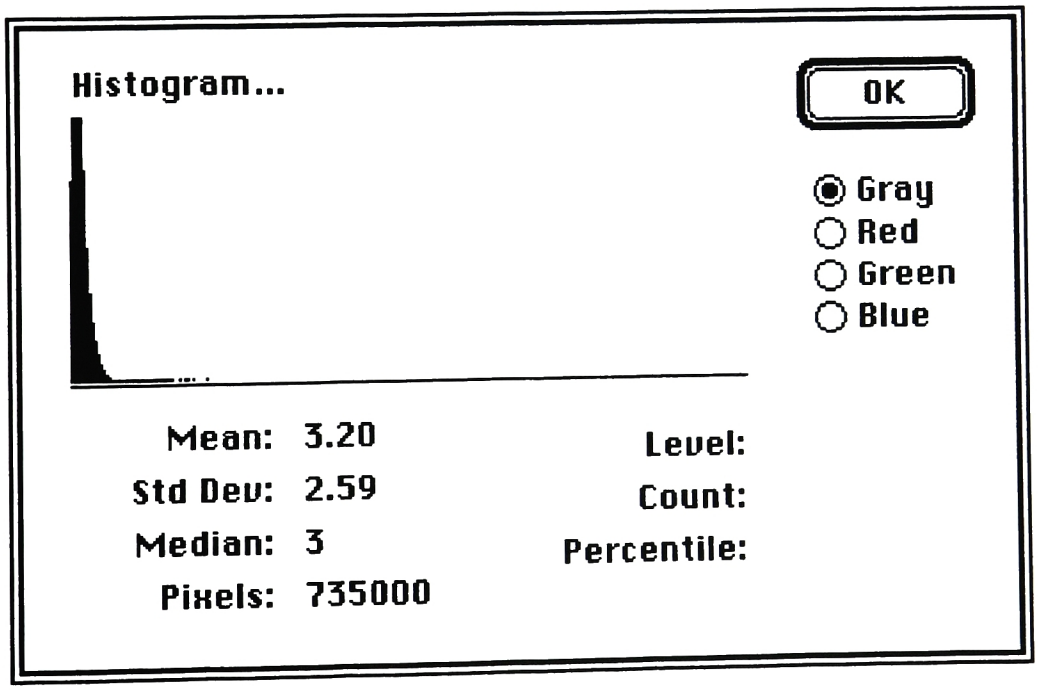
File: Fawns
Sub-Sampling Ratio: 2:1
Comp Factor: 15
Absolute Comp: 15:1



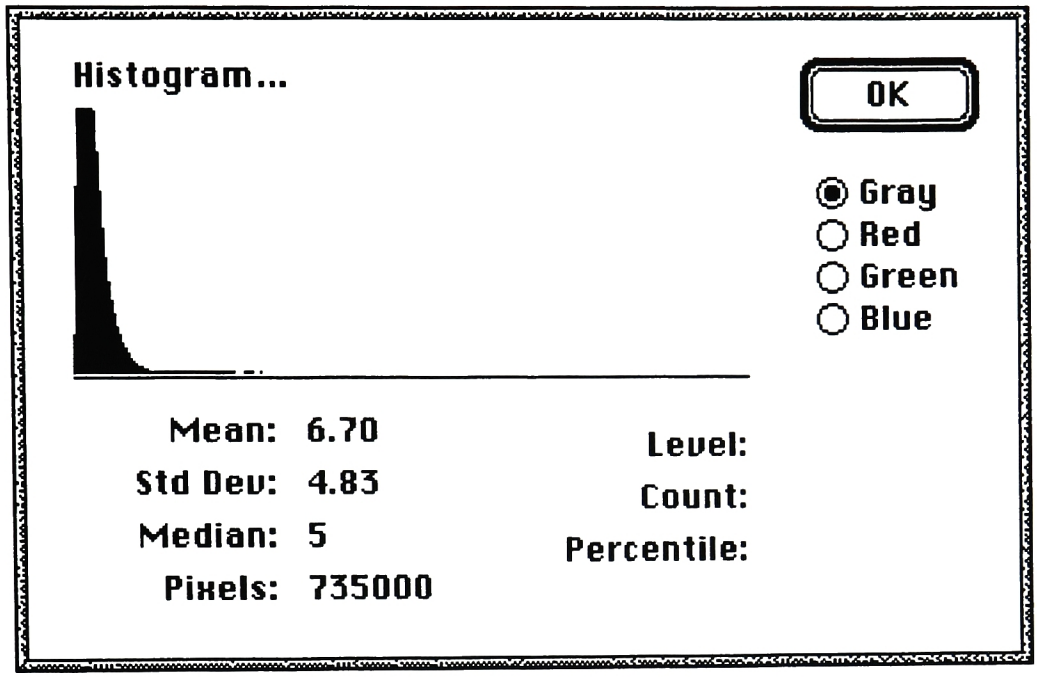
File: Fawns
Sub-Sampling Ratio: 1:1
Comp Factor: 15
Absolute Comp: 11:1



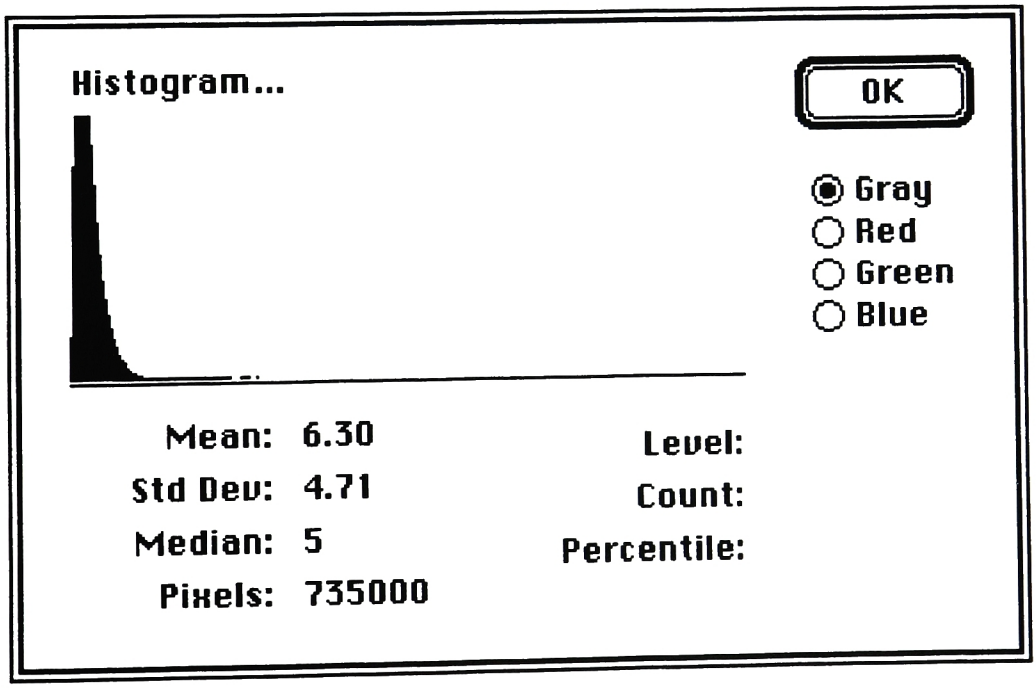
File: Fawns
Sub-Sampling Ratio: 2:1
Comp Factor: 9
Absolute Comp: 33:1



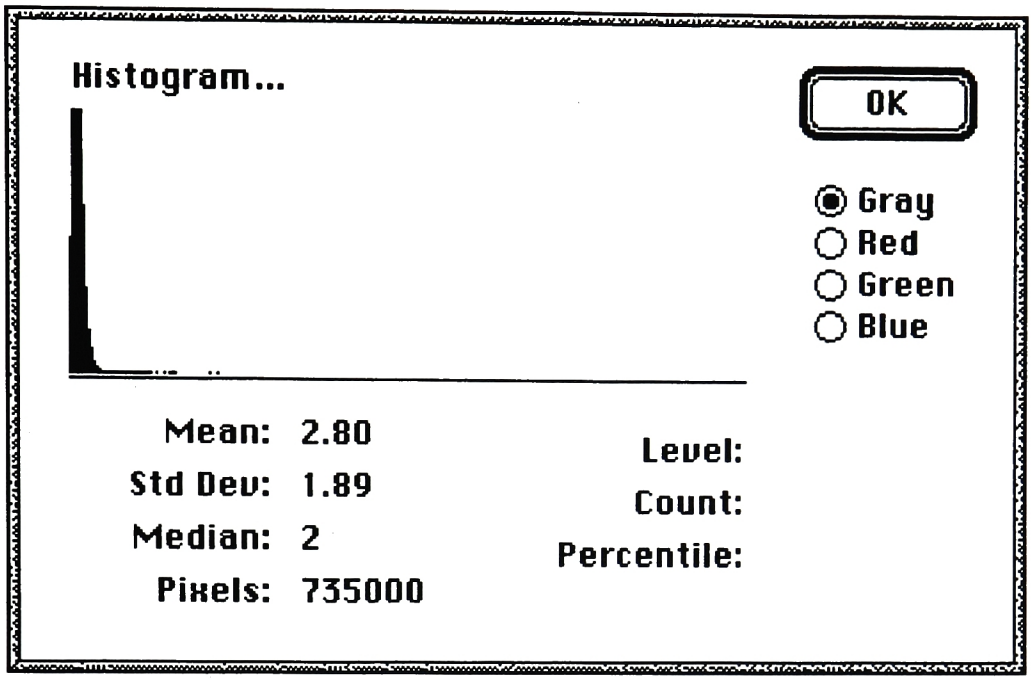
File: Fawns
Sub-Sampling Ratio: 1:1
Comp Factor: 9
Absolute Comp: 25:1



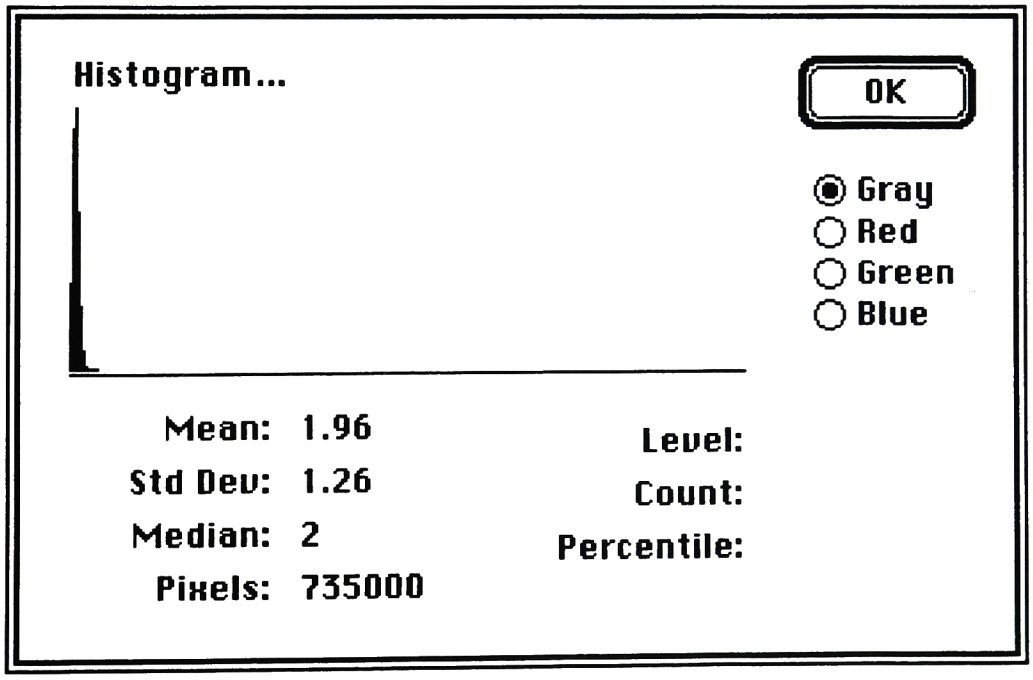
File: Fawns Comp Factor: 3
Sub-Sampling Ratio: 2:1 Absolute Comp: 77:1



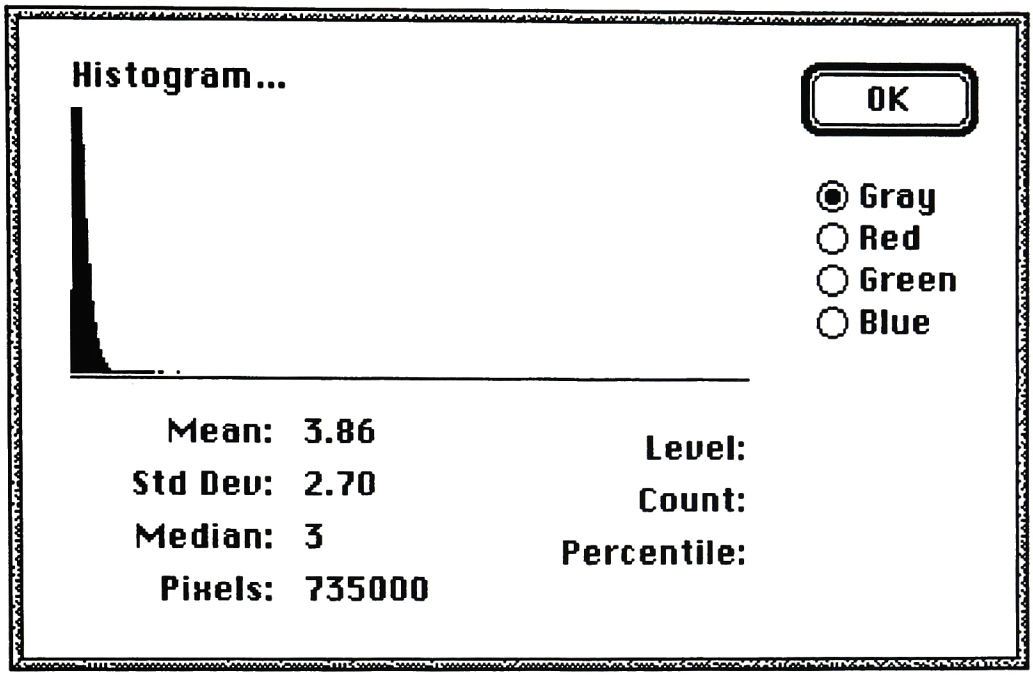
File: Fawns Comp Factor: 3
Sub-Sampling Ratio: 1:1 Absolute Comp: 57:1



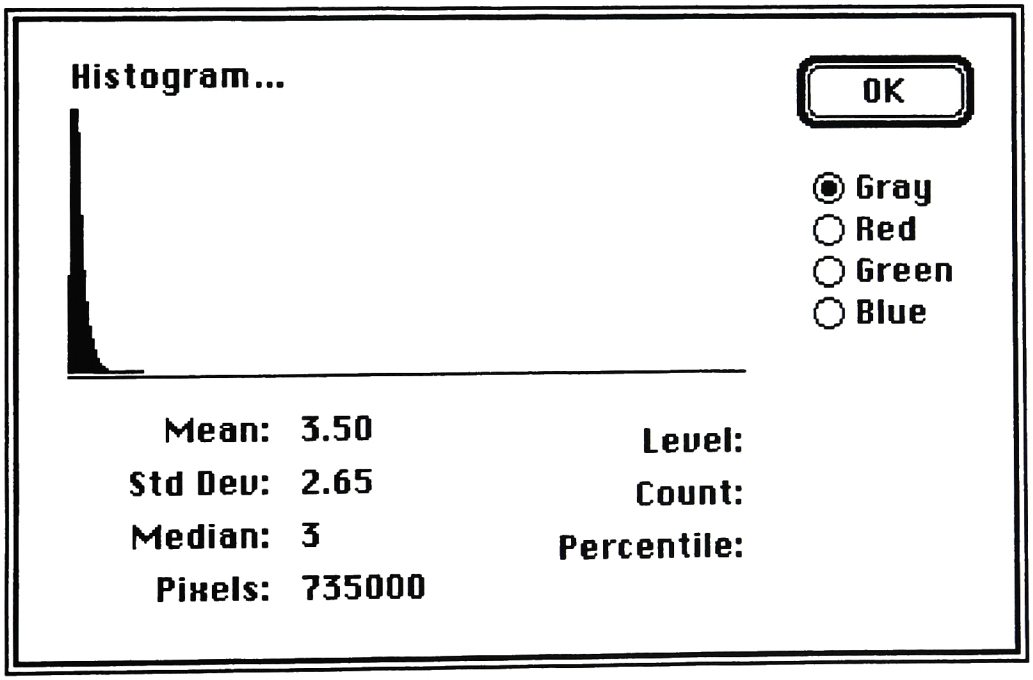
File:	Web	Comp Factor:	21
Sub-Sampling Ratio:	2:1	Absolute Comp:	4:1



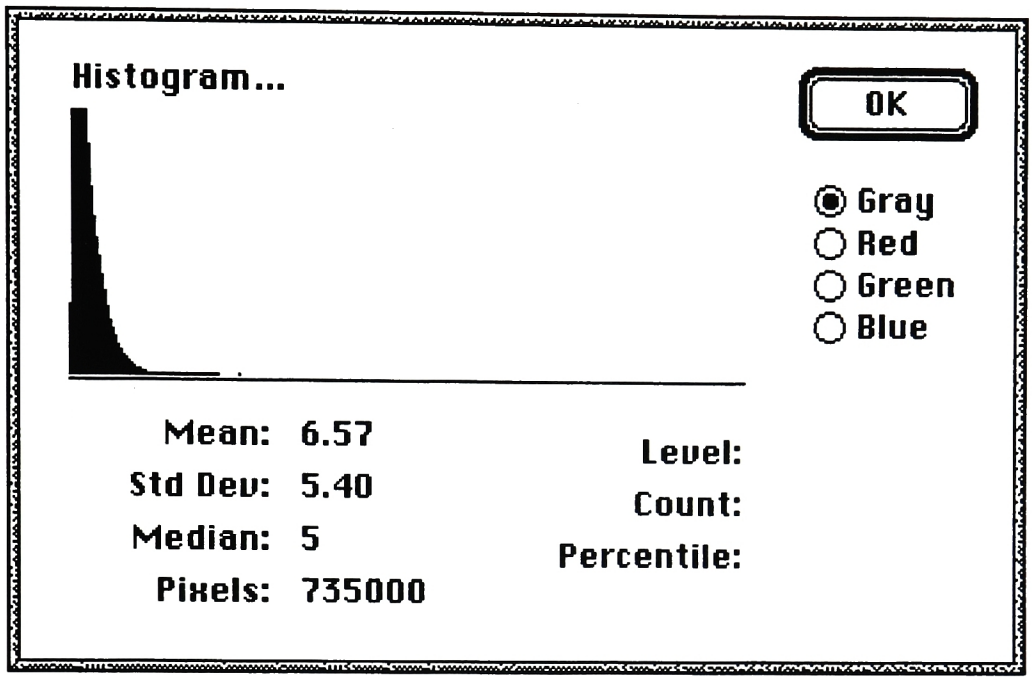
File:	Web	Comp Factor:	21
Sub-Sampling Ratio:	1:1	Absolute Comp:	3:1



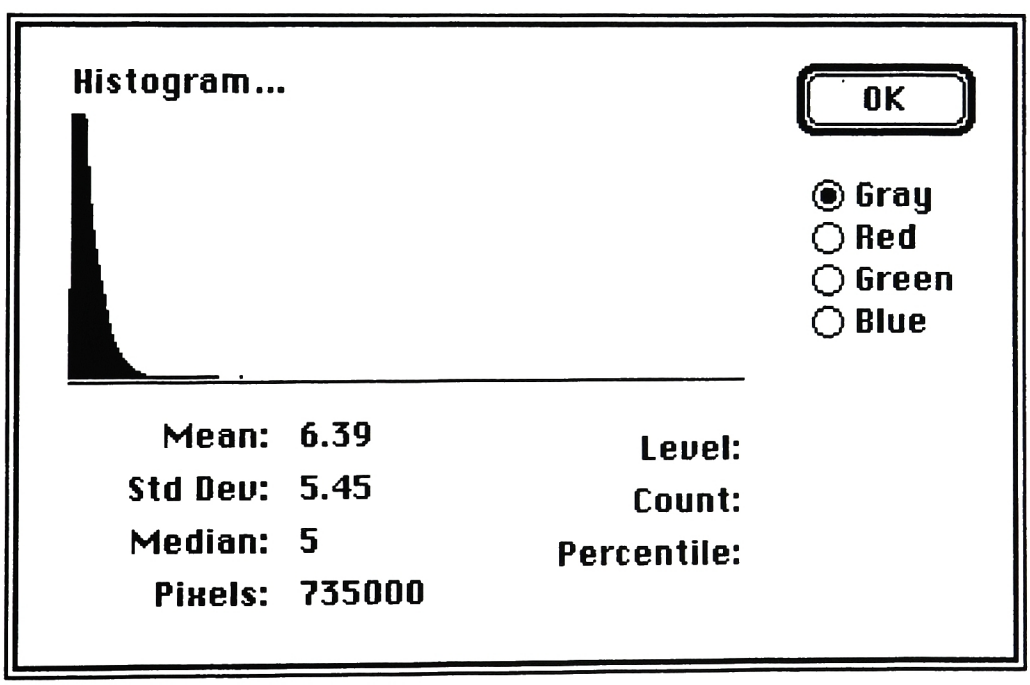
File:	Web	Comp Factor:	15
Sub-Sampling Ratio:	2:1	Absolute Comp:	9:1



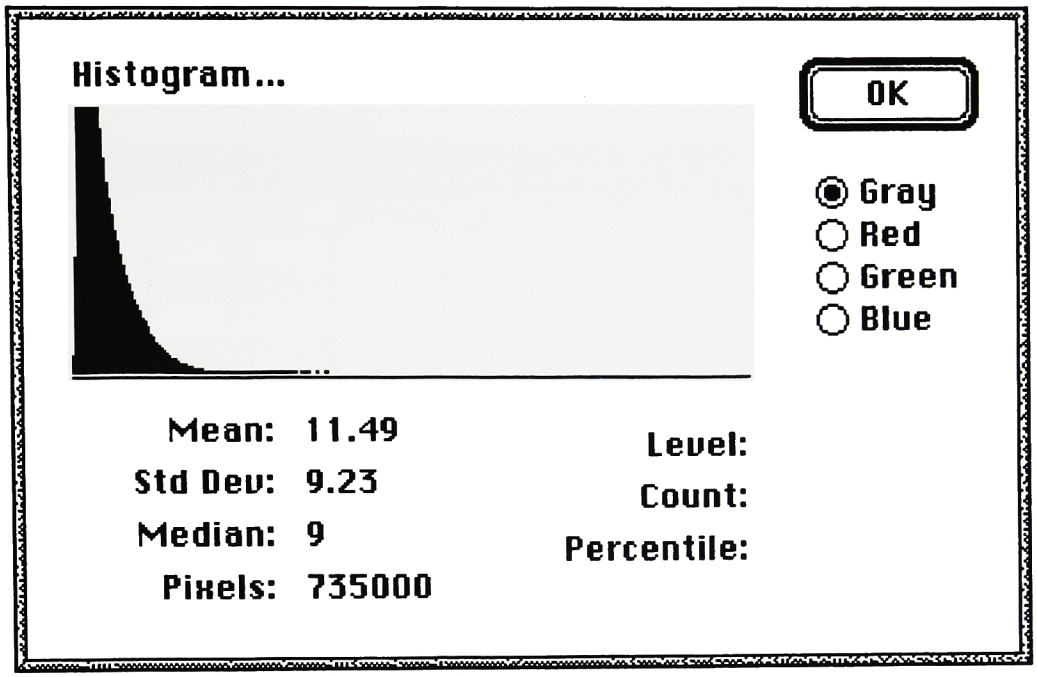
File:	Web	Comp Factor:	15
Sub-Sampling Ratio:	1:1	Absolute Comp:	7:1



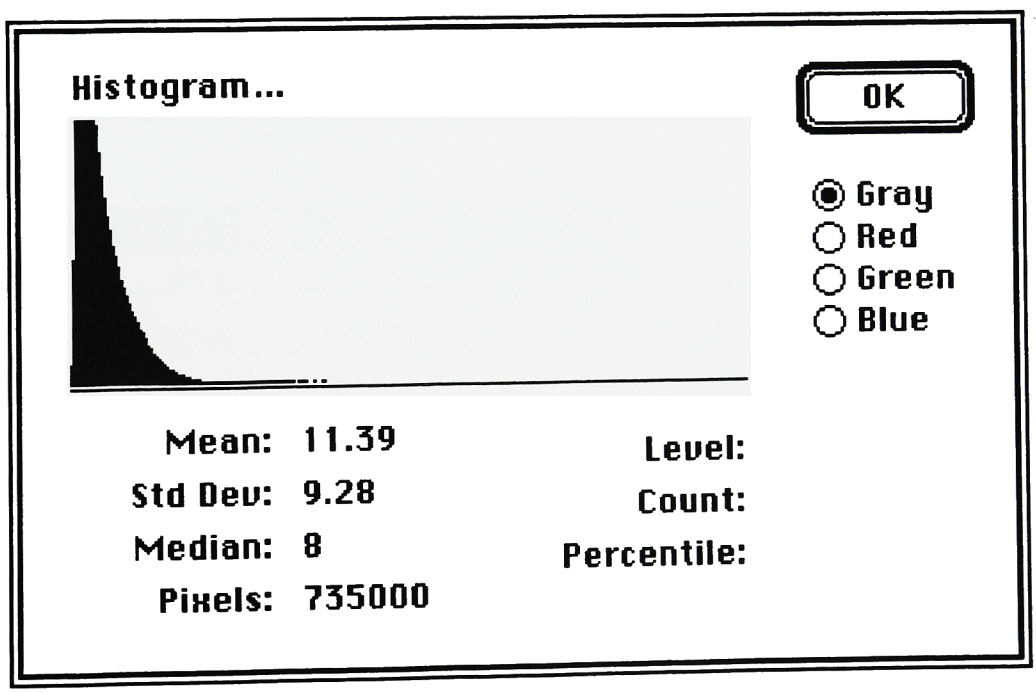
File:	Web	Comp Factor:	9
Sub-Sampling Ratio:	2:1	Absolute Comp:	18:1



File:	Web	Comp Factor:	9
Sub-Sampling Ratio:	1:1	Absolute Comp:	16:1



File:	Web	Comp Factor:	3
Sub-Sampling Ratio:	2:1	Absolute Comp:	41:1



File:	Web	Comp Factor:	3
Sub-Sampling Ratio:	1:1	Absolute Comp:	34:1

Absolute Compression & Time Data

SUNRISE TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.1	5.1	12.2	22	98.0
Level 6	7.1	5.0	12.1	24	89.8
Level 9	7.1	5.1	12.2	31	69.5
Level 12	7.1	5.4	12.5	53	40.7
Level 15	7.3	5.6	12.9	101	21.3
Level 18	9.2	6.8	16.0	177	12.2
Level 21	9.4	7.1	16.5	320	6.7
Level 24	11.3	8.9	20.2	605	3.6

BUCK TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.1	4.6	11.7	29	74.3
Level 6	7.1	4.6	11.7	41	52.6
Level 9	7.4	4.6	12.0	61	35.3
Level 12	8.5	5.3	13.8	97	22.2
Level 15	8.8	6.7	15.5	155	13.9
Level 18	9.3	6.9	16.2	246	8.8
Level 21	10.8	7.1	17.9	396	5.4
Level 24	11.4	8.6	20.0	676	3.2

DAD TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.1	4.8	11.9	35	61.6
Level 6	7.1	5.2	12.3	48	44.9
Level 9	7.1	5.8	12.9	69	31.2
Level 12	7.6	6.7	14.3	103	20.9
Level 15	8.3	6.4	14.7	156	13.8
Level 18	9.2	6.8	16.0	240	9.0
Level 21	9.8	6.9	16.7	371	5.8
Level 24	11.0	8.3	19.3	612	3.5

Absolute Compression & Time Data (cont.)

FAWNS TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.9	5.2	13.1	38	56.7
Level 6	7.8	5.1	12.9	56	38.5
Level 9	8.1	6.4	14.5	85	25.4
Level 12	8.7	6.4	15.1	129	16.7
Level 15	8.8	6.9	15.7	194	11.1
Level 18	9.8	7.3	17.1	292	7.4
Level 21	10.8	7.4	18.2	439	4.9
Level 24	11.4	9.0	20.4	703	3.1

OWL TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	8.0	4.7	12.7	47	45.9
Level 6	8.3	6.0	14.3	66	32.7
Level 9	8.4	6.1	14.5	96	22.4
Level 12	8.5	6.3	14.8	145	14.9
Level 15	8.9	7.0	15.9	221	9.8
Level 18	10.5	7.1	17.6	335	6.4
Level 21	11.1	8.2	19.3	523	4.1
Level 24	12.2	9.1	21.3	821	2.6

FERNS TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	8.5	6.2	14.7	50	43.1
Level 6	8.5	6.2	14.7	75	28.7
Level 9	8.6	6.8	15.4	114	18.9
Level 12	8.9	6.9	15.8	172	12.5
Level 15	9.0	6.9	15.9	257	8.4
Level 18	10.9	7.3	18.2	385	5.6
Level 21	11.3	8.5	19.8	586	3.7
Level 24	12.6	9.3	21.9	883	2.4

Absolute Compression & Time Data (cont.)

WEB TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	8.6	6.3	14.9	63	34.2
Level 6	8.7	6.2	14.9	90	23.9
Level 9	8.9	6.8	15.7	133	16.2
Level 12	8.9	7.0	15.9	200	10.8
Level 15	9.1	7.2	16.3	295	7.3
Level 18	11.0	7.5	18.5	431	5.0
Level 21	11.2	8.9	20.1	639	3.4
Level 24	12.9	9.5	22.4	944	2.3

Resolution Data

FAWNS 200 dpi — 2,155K

	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.9	5.2	13.1	38	56.7
Level 9	8.1	6.4	14.5	85	25.4
Level 15	8.8	6.9	15.7	194	11.1
Level 21	10.8	7.4	18.2	439	4.9

FAWNS 175 dpi — 1,652K

	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	5.9	4.0	9.9	30	55.1
Level 9	5.9	4.0	9.9	69	23.9
Level 15	5.9	4.4	10.3	157	10.5
Level 21	7.8	5.7	13.5	354	4.7

FAWNS 150 dpi — 1,214K

	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	4.3	3.5	7.8	24	50.6
Level 9	4.8	3.5	8.3	55	22.1
Level 15	4.8	3.5	8.3	125	9.7
Level 21	6.1	4.5	10.6	282	4.3

FAWNS 125 dpi — 845K

	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	2.8	1.6	4.4	17	49.7
Level 9	2.8	2.0	4.8	41	20.6
Level 15	2.9	2.8	5.7	95	8.9
Level 21	4.1	2.9	7.0	214	3.9

FAWNS 100 dpi — 541K

	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	2.1	1.8	3.9	12	45.1
Level 9	2.1	1.9	4.0	29	18.7
Level 15	2.2	2.2	4.4	67	8.1
Level 21	2.2	2.5	4.7	153	3.5

FAWNS 75 dpi — 305K

	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	0.9	1.1	2.0	7	43.6
Level 9	0.9	1.2	2.1	18	16.9
Level 15	0.9	1.4	2.3	43	7.1
Level 21	1.5	1.5	3.0	97	3.1

File Format Data

FAWNS PICT — 2,110K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.3	4.6	11.9	38	55.5
Level 6	8.2	5.2	13.4	56	37.7
Level 9	8.1	5.7	13.8	85	24.8
Level 12	8.7	6.4	15.1	129	16.4
Level 15	8.9	6.8	15.7	194	10.9
Level 18	10.2	7.0	17.2	292	7.2
Level 21	10.8	7.2	18.0	439	4.8
Level 24	11.3	8.9	20.2	703	3.0

FAWNS TIFF — 2,155K					
	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.9	5.2	13.1	38	56.7
Level 6	7.8	5.1	12.9	56	38.5
Level 9	8.1	6.4	14.5	85	25.4
Level 12	8.7	6.4	15.1	129	16.7
Level 15	8.8	6.9	15.7	194	11.1
Level 18	9.8	7.3	17.1	292	7.4
Level 21	10.8	7.4	18.2	439	4.9
Level 24	11.4	9.0	20.4	703	3.1

Sub-Sampling Ratio Data

SUNRISE — 2,155K					
1:1:1 RATIO	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.1	5.1	12.2	22	98.0
Level 12	7.1	5.4	12.5	53	40.7
Level 15	7.3	5.6	12.9	101	21.3
Level 24	11.3	8.9	20.2	605	3.6
2:1:1 RATIO					
Level 3	4.0	3.7	7.7	13	165.8
Level 12	4.3	3.8	8.1	22	98.0
Level 15	4.6	4.0	8.6	75	28.7
Level 24	5.7	4.8	10.5	223	9.7

FAWNS — 2,155K					
1:1:1 RATIO	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	7.9	5.2	13.1	38	56.7
Level 12	8.1	6.4	14.5	85	25.4
Level 15	8.8	6.9	15.7	194	11.1
Level 24	10.8	7.4	18.2	439	4.9
2:1:1 RATIO					
Level 3	4.6	3.9	8.5	28	77.0
Level 12	4.7	4.0	8.7	66	32.7
Level 15	4.8	4.8	9.6	146	14.8
Level 24	6.2	5.4	11.6	325	6.6

WEB — 2,155K					
1:1:1 RATIO	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
Level 3	8.6	6.3	14.9	63	34.2
Level 12	8.9	6.8	15.7	133	16.2
Level 15	9.1	7.2	16.3	295	7.3
Level 24	11.2	8.9	20.1	639	3.4
2:1:1 RATIO					
Level 3	4.8	3.5	8.3	53	40.7
Level 12	5.3	4.5	9.8	118	18.3
Level 15	5.6	5.0	10.6	245	8.8
Level 24	6.9	6.1	13.0	500	4.3

Compression Cycles Data

Fawn Compression Cycles — Compression Factor 3

	Comp Time	Decomp Time	Total Time	Comp File Size	Comp Ratio
1 Cycle	8.0	4.7	12.7	38	56.7
2 Cycles	7.1	4.6	11.7	38	56.7
3 Cycles	7.0	4.6	11.6	38	56.7
4 Cycles	7.0	4.6	11.6	38	56.7
5 Cycles	7.4	4.7	12.1	38	56.7

APPENDIX E — IMAGE SIZE & RESOLUTION vs. FILE SIZE

In order to help determine the storage requirements of 24-bit images of various sizes and resolutions, Figure E–1 is included. This chart covers most of the basic sizes and resolutions typically used in normal production. To determine storage requirements for file sizes and resolutions not contained in Figure E–1, the following formula can be used:

File Size (bytes) = (Brightness Resolution/8) x (Spatial Resolution)^2 x (Dimensions)

For example assume an image with the following characteristics:

Brightness Resolution = 24-bit

Spatial Resolution = 225 dots per inch (dpi)

Dimensions = 2.75 inches x 5.75 inches

File Size = (24/8) x (225)^2 x (2.75 x 5.75)

= 3 x 50,625 x 15.81

= 2,401,144 bytes

To convert this number to kilobytes, divide by 1,024

= (2,401,144) / 1,024 = 2,355 KB

To convert this number to megabytes, divide by 1,048,576

= (2,401,144) / 1,048,576 = 2.3 MB

FILE SIZE FOR 24-BIT COLOR IMAGES

	1" x 1"	2" x 3"	4" x 6"	5" x 7"	8" x 10"
75 dpi	16.5 KB	98.9 KB	395.5 KB	576.8 KB	1.3 MB
100 dpi	29.3 KB	175.8 KB	703.1 KB	1.0 MB	2.3 MB
150 dpi	65.9 KB	395.5 KB	1.5 MB	2.2 MB	5.1 MB
200 dpi	117.2 KB	703.1 KB	2.7 MB	4.0 MB	9.2 MB
300 dpi	263.7 KB	1.5 MB	6.2 MB	9.0 MB	20.6 MB
400 dpi	468.8 KB	2.7 MB	11.0 MB	16.0 MB	36.6 MB
600 dpi	1.0 MB	6.2 MB	24.7 MB	36.0 MB	82.4 MB
1000 dpi	2.9 MB	17.2 MB	68.7 MB	100.1 MB	228.9 MB

Figure E–1

Information in this Appendix is paraphased from *Scanned File Size*, ©1990 Linotype Company.

Apple Macintosh Quadra 700 / 8MB RAM

All tests were performed using a single Quadra 700 with a fixed configuration. This insured that time tests between different files and JPEG settings were being compared on an equal basis. This same computer was also used to layout and produce the entire reference.

Nikon LS–3510 AF Slide Scanner

The original 35mm transparencies were scanned at 700 dpi using the Nikon scanner in order to produce TIFF images which measured 3.5" x 5.25" at 200 dpi.

Canon CLC 500 Color Copier

All color pages in Appendices A, B, and C were output at 200 dpi on the same Canon CLC 500 with a *DiceNet* interface to the Macintosh. Settings were held constant for all output pages in order to reduce the number of variables affecting image quality.

44 MB SyQuest Drives / Cartridges

A number of different SyQuest cartridges and drives were used to transport images from one workstation to another. All files used to produce this reference comprised over 140 MB of storage space.

Radius ImpressIt v.1.1

Radius ImpressIt was chosen to run all tests contained in this reference because it offered a wide range of features to test. The software was allocated 4 MB of RAM as well as a 32 MB virtual memory hard disk partition. All files were compressed without saving a thumbnail snapshot or a self-decompressing file.

Adobe Photoshop v.2.0.1

Adobe Photoshop was used for file format changes, generating difference files, and producing histograms. In addition, *Photoshop* was used to prepare all images before layout into *Quark XPress.*

Microsoft Excel v.4.0

All data generated during testing was processed using *Microsoft Excel. Excel* was also used to produce the data tables in Appendix D as well as the charts in Chapter 5.

Adobe Illustrator v.3.0.1

Line art drawings and block diagrams in the body of the reference were produced using *Adobe Illustrator.*

Aldus PageMaker v.4.2

Aldus PageMaker was used to produce the entire body of the reference with the exception of Appendices A, B, and C.

Quark XPress v.3.1

Quark XPress was used to layout the color pages in Appendices A, B, and C because a *DiceNet* driver does not exist for *Aldus PageMaker.*

APPENDIX G — GLOSSARY

Absolute Compression: The original file size divided by the compressed file size. Typically, but not always, expressed as a ratio such as 10:1 or 100:1.

AC Coefficient: A term used to describe any number in a JPEG matrix except the one in the upper left hand corner (see DC Coefficient).

Arithmetic Coding: A compression scheme which identifies frequent terms and replaces them with smaller terms which require less storage space. Particularly useful when original data is composed of tightly grouped numbers of high magnitude.

Brightness Resolution: The number of tonal steps (or brightness steps) that can be achieved on a certain device. Brightness resolution is frequently referred to in terms of bits when discussing scanners and monitors. To determine the tonal range of a certain device, use the formula 2^x = available tonal steps; where x equals the number of bits. For example an 8-bit grey scale monitor can produce 256 (2^8 = 256) different values of grey. A 24-bit monitor is capable of producing 256 shades each of red, green and blue (3 different 8-bit channels) for a total of roughly 16.8 million colors.

Compression Factor: The value input by the user of a JPEG product which indicates the desired level of compression. Also known as picture quality, quality level, and numerous other terms.

DC Coefficient: A number used in JPEG compression to describe the average overall magnitude of the pixels in an given eight-by-eight matrix. This number is placed in the upper left hand corner of the DCT output matrix.

Discrete Cosine Transform (DCT): A mathematical equation used to transform information from the spatial domain into the frequency domain. Used in the first step of JPEG compression to allow for more efficient run length encoding.

Entropy Encoding: A data encoding scheme which takes advantage of statistical probabilities to compress information. In the case of JPEG compression, entropy encoding is carried out using either Huffman (run length) or arithmetic coding.

Frequency: In the case of images, frequency is used to describe the amount of variation between light and dark areas. High frequency images contain large variations between light and dark areas and vice-versa for low frequency images.

JPEG Compression: A compression scheme developed by the Joint Photographic Experts Group for use with continuous tone images. Expected to be adopted as the international standard for image compression in early 1993 (ISO standard 10918-1 and 10918-2).

Lossless Compression: A form of data compression in which all of the original information is retained. Useful with files such as office documents, spreadsheets, etc. where data integrity is of paramount importance. Typically able to achieve compression ratios on the order of 2:1, making its value somewhat limited in the case of continuous tone images.

Lossy Compression: A form of data compression in which some of the original information is discarded. Only useful in cases such as image compression where the loss of data is not critical. Lossy compression is able to produce compressed files many orders of magnitude smaller than those produced using lossless compression.

Luminosity: The brightness of an object or pixel. Luminosity ranges from 0 (black) to 255 (white) in the case of 8-bit grey scale images and 24-bit color images.

Matrix: A fancy, mathematical term for a table or chart.

RGB Color Model: A color model which describes all colors as a combination of varying amounts of red, green, and blue. All television sets and computer monitors are based on the RGB color model.

Run Length Coding: A compression scheme which identifies strings of numbers and replaces them with more efficient code. Particularly useful when data contains long strings of similar information.

Spatial Resolution: The amount of information per linear unit. Spatial resolution is frequently referred to in terms of dpi (dots per inch) when discussing input and output devices.

Quantization: The process of reducing the accuracy of a number through division or truncation so that it requires less storage space.

YUV Color Model: A color model which describes all colors in terms of one luminance channel (Y) and two chrominance channels (U and V). Images are converted into the YUV color space for JPEG processing to allow greater compression of the chrominance channels.