IMAGE RECONSTRUCTION FOR DISCRETE COSINE TRANSFORM COMPRESSION SCHEMES

By

JEFFERY KERSEY OTTO

Bachelor of Science
Oklahoma State University
Stillwater, Oklahoma
1986

Master of Science
Oklahoma State University
Stillwater, Oklahoma
1988

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 1993

IMAGE RECONSTRUCTION FOR DISCRETE COSINE

TRANSFORM COMPRESSION SCHEMES

Thesis Approved:

C. M. Bauer

Thesis Advisor

Blayne E. Mayfield

Kevin A. Teague

Thomas C. Collins

Dean of the Graduate College

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to the individuals who made this undertaking possible. Without each and every one of them, I never would have finished this project. In particular, I wish to thank my friend, mentor, and major advisor, Dr. Charles Bacon, for his energetic assistance, intelligent guidance, constant and enthusiastic encouragement, and countless hours of critiquing and editing. Dr. Keith Teague deserves special recognition for his technical guidance at mid-stream, invaluable reviews, advice and overall strategy. I am also most grateful to my other committee members, Dr. Ronald Rhoten, and Dr. Blayne Mayfield.

I would like to thank Ms. Ann Roberts for her expeditious and discerning help in the research phase without which this project would have been much slower to start. I am also grateful to Scott and Margaret Cooper for editorial comments, and for providing support at every step on this journey. A great deal of appreciation goes to a long time co-conspirator, Mrs. Joanna Hensley for her friendship, moral support and empathic ear through the years. My friends Tim and Shelley, Todd and Kate, Tim and Lisa, and Chuck and Kristye always provided just the right amount of interest at just the right time.

Finally, I would like to thank my family for making this all possible. Thanks to my mother and father for giving me an interest in science and engineering (and not a little financial support), for the understanding and sustaining encouragement, and for making sure I kept the goal in sight. The rest of my family, Randy, Kris and Kent for their curiosity and unconditional enthusiasm. My daughter Carly provided me with the motivation for the final push. Most of all I would like to thank my wife Jana for her patience, understanding, support and determination. Truly, only the mistakes are mine.

TABLE OF CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

LIST OF SYMBOLS

$\Leftrightarrow$	A transform between the spatial domain and frequency domain.
$\otimes$	Convolution.
x, y	Spatial variables in the original image domain.
r, s	Spatial variables in the reconstructed image domain.
u, v	Frequency variables in the discrete cosine transform (DCT) domain.
ω_1, ω_2	Frequency variables in the Fourier domain.
ζ	Scaling factor.
$\delta()$	Dirac delta function.
S()	The spatial domain sampling function.
F()	The spatial continuous image function.
F()	The Fourier continuous image function.
$\mathcal{F}()$	The cosine domain continuous image function.
$F_s()$	The spatial sampled image.
$\mathbf{F_s()}$	The Fourier sampled image.
$\mathcal{F}_s()$	The cosine sampled image.
R()	The spatial reconstruction function.
R()	The Fourier reconstruction function.
$\mathcal{R}()$	The cosine reconstruction function.

CHAPTER 1

INTRODUCTION

Problem

During the past several years, digital imaging has become more popular. As the number of image-enabled applications grows, the demand for high quality images also grows. However, a problem exists for imaging applications running on today's hardware. A typical digital image requires a large amount of storage space. If a source image with a size of 8.5 by 11 inches is scanned (digitized) at 300 dots per inch, the resulting digital image is 2550 by 3300 pixels, or 8.4 million pixels. Depending on the number of colors needed, each pixel contains from 1 to 24 bits of color resolution. Today, a high quality color image has 24 bits (8 bits for each primary color; red, blue, and green). The resulting digital image has a size of just over 25 million bytes. Obviously, today's typical computer cannot store many of these images nor can the images be transmitted or processed rapidly.

Image compression is an area that has seen considerable interest for just this reason. The total size of the stored image can be reduced, saving storage requirements and transmission time. There have been several image compression methods proposed, each with different qualities and each with varying degrees of acceptance. Currently, no still frame image compression method offers a greater potential than the transform compression method; the most popular being the Joint Photographic Experts Group (JPEG) standard [JPEG Draft 92] which is based on the discrete cosine transform (DCT).

The JPEG algorithm, the transform and supporting coding, is CPU intensive, both in the compression stage and in the decompression stage. To make matters worse, once the original image is decompressed, the image must often be scaled in size to match a particular output device such as a video monitor. The scaling step is often an inaccurate approximation that involves errors from interpolation. Aliasing is often apparent if the scaling algorithm results in minifying the image. A similar problem is encountered under magnification, where pixelization occurs.

The process for reducing these distortions requires two steps: First, reconstruct the continuous image signal from the discrete image, and second, resample the new continuous image signal at desired output positions to match the output device characteristics. The decompression, reconstruction, and resampling steps all contribute to a longer than desired delay before the image can be viewed. Although the JPEG algorithm uses only the discrete image samples, the continuous image signal is needed for the scaling and resampling. The original discrete samples are usually thought of as a sampling of the original continuous image. A high quality scanning device or digital camera usually provides the first continuous to discrete operation.

A fast method to view a JPEG image is needed. However, the JPEG algorithm is intended to provide a high quality image but still maintain good compression. The tradeoff is between maintaining the high quality level of the original image, and providing a fast algorithm for decompression, reconstruction, and resampling.

Proposed Research

Previous research focuses on three main areas, image compression using transform coding, reconstruction functions, and scaling algorithms. Although the discrete cosine transform has been around for some time [Ahmed 74], and has been suggested as part of an image compression process, its use in a widely accepted international standard

is very recent. While there is reference to the possibility of combining these three steps using the Fourier transform [Wolberg 90], there is no research using the cosine transform. This is due to two main reasons. First, the JPEG algorithm is new, therefore few people have investigated its properties. Second, while Fourier analysis is often used to show the frequency properties of a particular reconstruction algorithm, the algorithm is usually implemented in the spatial domain for performance reasons and clarity. When a frequency domain study is performed in the literature, it is performed in the Fourier domain. The most likely reason seems to be that the Fourier transform and domain are clearly understood, and are a useful reference across many disciplines.

There are now several implementations of a JPEG decompression algorithm, but the reconstruction and scaling steps have been ignored, either because of performance reasons or because the original characteristics of the image are already matched to the output device and no reconstruction or scaling is required.

The ideas presented here describe a unique approach to this process. The proposed research is to combine the reconstruction with the scaling, and to operate directly on the compressed image data in the cosine-frequency domain before the decompression. In this manner, the reconstruction and scaling process will involve only the compressed image data, not the decompressed image data. This research determines if the operations required in the frequency domain are more complex and time consuming than those required in the time domain. As will be seen, the complexity of each operation is somewhat adjustable, depending on the interpolation and reconstruction algorithm used.

CHAPTER 2

BACKGROUND

Image Definitions

Pratt [Pratt 91] defines an image as a continuous, infinite extent field whose value represents some aspect of a scene. The values are often considered to represent some known physical property, such as luminance, absorption, reflection or some range of energy [Jain 89], [Pearson 91]. The data is not required to have such meaning, but it often helps in conceptualizing the process involved.

In an image sampling system, the sampled discrete image, $F_S(x,y)$, can be obtained by multiplying the continuous image, F(x,y), by a sampling function, S(x,y), such as the comb function, or a two-dimensional grid of Dirac delta functions, spaced Δx and Δy apart.

$$S(x,y) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \delta(x - j\Delta x, y - k\Delta y) \qquad (2\text{-}1)$$

where S(x,y) is the spatial sampling function based on a grid of Dirac delta functions.

The sampled image then becomes

$$F_s(x,y) = F(x,y)S(x,y) = F(x,y)\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\delta(x - j\Delta x, y - k\Delta y) \tag{2-2}$$

where F(x,y) is the continuous ideal image, and F_S(x,y) is the sampled image.

Moving the continuous image function, F(x,y), inside the summation, yields

$$F_s(x,y) = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}F(j\Delta x, k\Delta y) \times \delta(x - j\Delta x, y - k\Delta y) \tag{2-3}$$

where F(jΔx, kΔy) is determined at only the sample points jΔx, kΔy.

Fourier Analysis

The Fourier transform is a well known linear transform [Andrews 70], [Russ 92], [Goodman 68]. The Fourier transform decomposes a signal into an infinite set of coefficients of orthogonal complex waveforms. The Fourier transform is just one of a number of ways of decomposing the signal into coefficients of orthogonal waveforms. Any orthogonal transformation will provide a similar decomposition. In particular the cosine transform, which is an orthogonal transformation, has similar properties as the Fourier transform, and was originally derived from the Fourier transform [Ahmed 74].

If the Fourier transform of the continuous image, $\mathbf{F}(\omega_1, \omega_2)$, is convolved with the Fourier transform of the sampling function, $\mathbf{S}(\omega_1, \omega_2)$, the result,

$$\mathbf{F}_S(\omega_1,\omega_2) = \frac{1}{4\pi^2}\mathbf{F}(\omega_1,\omega_2) \otimes \mathbf{S}(\omega_1,\omega_2) \tag{2-4}$$

is the Fourier transform of the sampled image. This equation for the Fourier sampled image, $\mathbf{F}_S(\omega_1, \omega_2)$, from the Fourier continuous image reduces to

$$\mathbf{F}_S(\omega_1,\omega_2) = \frac{1}{\Delta x \Delta y} \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \mathbf{F}(\omega_1 - j\omega_{1S}, \omega_2 - k\omega_{2S}) \tag{2-5}$$

where ω_{1S}, and ω_{2S}, are the discrete sampling frequencies corresponding to the sample spacing Δx, and Δy, used in the sampling function.

Let R(x,y) be a reconstruction function. Then, in order to reconstruct the continuous image from the sampled image, the sampled image can be spatially interpolated using $F_S(x,y)$, and R(x,y), or it can be filtered with a reconstruction function using $\mathbf{F}_S(\omega_1,\omega_2)$, and $\mathbf{R}(\omega_1,\omega_2)$. where $\mathbf{R}(\omega_1,\omega_2)$ is the Fourier response of the reconstruction function. In this case the reconstructed image, $F_R(x,y)$, can be written as the convolution of the sampled image, $F_S(x,y)$, and the interpolation function, R(x,y), as

$$F_R(x,y) = F_S(x,y) \otimes R(x,y) \tag{2-6}$$

or, using the Fourier transform of this equation, we have

$$\mathbf{F}_R(\omega_1,\omega_2) = \mathbf{F}_S(\omega_1,\omega_2)\mathbf{R}(\omega_1,\omega_2). \tag{2-7}$$

Using the same properties as before in the continuous to sampled case with Equation 2-5, the transform of the reconstructed image becomes

$$\mathbf{F}_R(\omega_1,\omega_2) = \frac{1}{\Delta x \Delta y} \mathbf{R}(\omega_1,\omega_2) \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \mathbf{F}(\omega_1 - j\omega_{1S}, \omega_2 - k\omega_{2S}). \tag{2-8}$$

From [Pratt 91], and ignoring aliasing problems, the reconstructed image is equal to the ideal image if $\mathcal{R}(\omega_1,\omega_2)$ removes all the values except where j, k = 0.

Reconstruction

Image reconstruction is an area of research concerned with recreating an image after some spatial manipulation or spatial scale function has been applied to the original sampled image [Andrews 72], [Fant 86]. Many times an image starts as a one-to-one mapping of pixels in the sampled image to pixels on a display device. But, this one-to-one spatial mapping often changes, either because the original display device is no longer available, or the image is rendered under some different spatial mapping function. Magnification and minification are examples.

When the image is magnified, pixel duplication can be used to render the image [Andrews 76]. For example, if one were to render the image at half its normal resolution (doubling its physical size), one image pixel would be translated into 4 identical pixels (a 2 by 2 square). This process leads to a phenomenon known as pixelization.

Likewise, under minification, the simplest method is to drop pixels, or decimate the image. In a similar example, if the image were rendered at twice its normal physical resolution (making the image half its size), every other pixel value would be dropped. This simple process leads to aliasing, which is well documented in the signal processing area.

Thus, image reconstruction should render the image in a new coordinate system, while these artifacts, such as pixelization and aliasing [Heckbert 86]. There are two steps to the reconstruction stage. The first step is to establish a continuous image, and the second step is to resample the new continuous image at some interval that will minimize the distortions, while lending itself to the output grid. Once a continuous image is correctly produced, the resampling grid can be of any interval, without causing the distortion artifacts.

In order to recreate the continuous image signal, a continuous function is matched to two (or more) image sample points. The missing image values between the two points

are assumed to be equal to the continuous function. Once the continuous signal is recreated in this manner, the image signal can be sampled wherever a value is needed. In contrast, if the original sampled image is used, image data is available only at the original sampling interval. Fitting a function to a set of data points is similar to using a low pass filter on the data signal. Since, the continuous function smoothes the original image sample data. As expected, the resulting image under magnification will not exhibit the same pixelization artifacts, since these high frequency changes have been smoothed by the interpolation function.

Interpolation functions can be implemented as a convolution in the spatial domain, but this approach in not usually applied because of time requirements. Usually, interpolation is performed by evaluating the interpolation polynomial. Interpolation can also be performed as a multiplication in the frequency domain. In order to use this frequency approach, the image data and the spatial interpolation function must both be transformed into a particular frequency domain, then multiplied, and the inverse transformation applied to the result. This frequency filtering process also has significant time penalties, because the computational requirements for the forward and inverse transformation of the entire image are usually substantial. However, examination of the frequency filtering method proves to be useful when developing a spatial interpolation function. Ideal interpolation in the frequency domain is a simple rectangle function [Ratzel 80], [Parker 83]. The frequency rectangle function filters out frequencies higher than one-half the sampling frequency, while passing unchanged the frequencies less than this cutoff frequency. If this ideal frequency function is transformed to the spatial domain, the result is the sinc function.

Sinc Function

The sinc function is defined as

$$\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \tag{2-9}$$

and is an the ideal spatial interpolation function, corresponding to the ideal rectangle function in the frequency domain. While this function, shown in Figure 2-1, is the exact function needed for reconstruction, it has computational problems that prohibit its use.

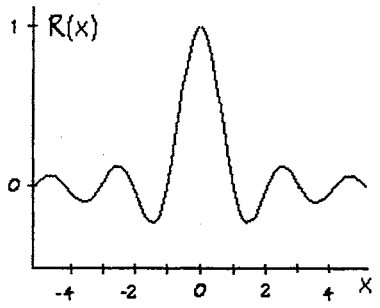


Figure 2-1.
The Sinc Function

From the graph of the sinc function, Figure 2-1, it is easy to see that it has significant energy in the side bands, which have infinite extent in the space domain. If this function is truncated in the space domain [Ratzel 80], [Parker 83], then the loss of the sideband energy produces ringing in the frequency domain. In Ratzel's comparisons, this method of truncating the sinc function proved to be inferior to some of the other interpolation functions like the cubic B-spline approach, which is discussed below.

Nearest Neighbor Interpolation.

This is the simplest method of interpolation. Since it represents the zero-order polynomial, it is computationally the quickest, but has corresponding accuracy drawbacks.

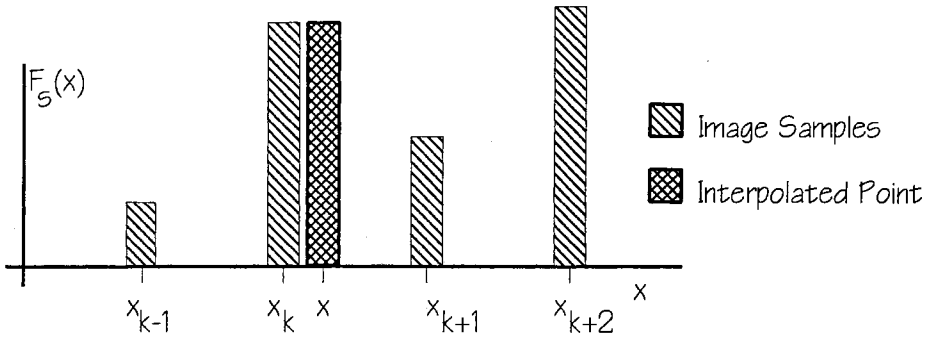


Figure 2-2. Nearest Neighbor Interpolation

As shown in Figure 2-2, each output pixel takes on the value of its nearest neighbor on the input grid. This is also called the point shift algorithm, and is defined as

$$F_R(x) = F(x_k), \quad \text{for } \frac{x_{k-1} + x_k}{2} < x \le \frac{x_k + x_{k+1}}{2}. \tag{2-10}$$

In the spatial domain, nearest neighbor interpolation can be achieved by convoluting the image with

$$\begin{aligned} R(x) &= 1, \quad 0 \le |x| < 0.5 \\ R(x) &= 0, \quad 0.5 \le |x|. \end{aligned} \tag{2-11}$$

This is a simple rectangle, as shown in Figure 2-3. It is also called the box filter, Fourier window, or the sample-and-hold function.

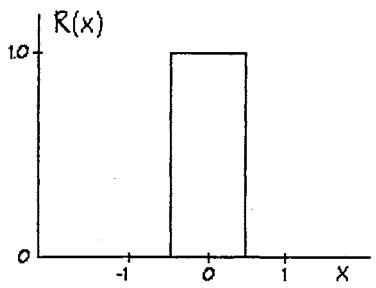


Figure 2-3.
The Rectangle Function

The nearest neighbor function results in pixelization in magnification, and when doing minification, the image is sampled, and produces aliasing artifacts. The image is also subject to position errors of up to one half of a pixel width (or height). For example, if the new sample falls between two of the original samples, the new pixel takes on the value of one of the original pixel values, but its location on the new grid is half way between the original points. This results in the new pixel position having a phase error of one half a pixel width. For minor scaling changes, nearest neighbor is usually considered adequate given its computational efficiency. For large scaling changes, nearest neighbor results in either aliasing or pixelization.

This method has seen much general use in the academic image processing field, and the commercial document imaging field. It is still used today where quick estimates are needed [Asal 86], [Eldon 90], such as in real time magnification. However, it is being replaced by more sophisticated algorithms.

Linear Interpolation

Linear interpolation uses a first-order polynomial function, as shown in Figure 2-4.

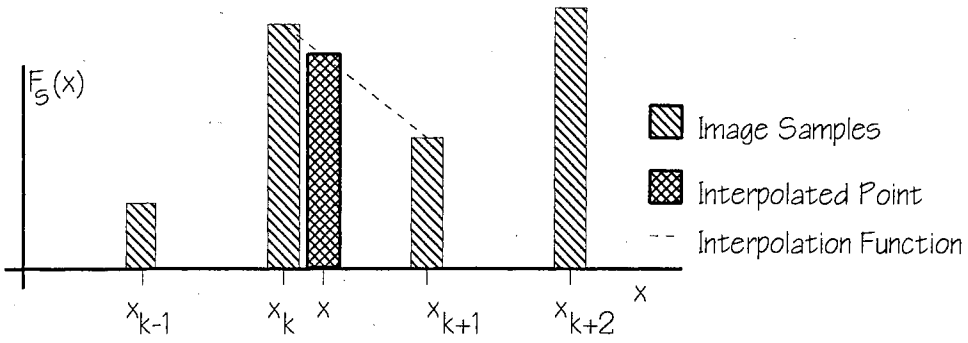


Figure 2-4. Linear Interpolation

For a linear interpolation function, the interpolated point lies on a straight line defined by the neighboring two image samples. The value of the function at the interpolated point is

$$F_R(x) = F(x_k) + \frac{x - x_k}{x_{k+1} - x_k}\left[F(x_{k+1}) - F(x_k)\right] \tag{2-12}$$

which is the equation for a line passing between the two sample points.

In the spatial domain, linear interpolation is achieved using convolution with a triangle function

$$\begin{aligned} R(x) &= 1 - |x|, \quad 0 \le |x| < 1 \\ R(x) &= 0, \quad 1 \le |x|. \end{aligned} \tag{2-13}$$

which is shown in Figure 2-5.

This interpolation function is a simple, but reasonable low pass filter in the frequency domain. It is useful to note the triangle function in Figure 2-5 is the convolution of two box functions.

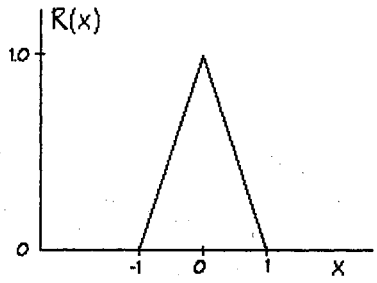


Figure 2-5.
The Triangle Function

Second Order Interpolation

If the triangle function is the convolution of two box functions, convolution with another box function yields a second order bell shaped wave form defined as

$$\begin{aligned} R(x) &= \tfrac{1}{2}(x+\tfrac{3}{2})^2, && -\tfrac{3}{2} \le x \le -\tfrac{1}{2} \\ &= \tfrac{3}{4} - x^2, && -\tfrac{1}{2} \le x \le \tfrac{1}{2} \\ &= \tfrac{1}{2}(x-\tfrac{3}{2})^2, && \tfrac{1}{2} \le x \le \tfrac{3}{2}. \end{aligned} \tag{2-14}$$

and shown in Figure 2-6.

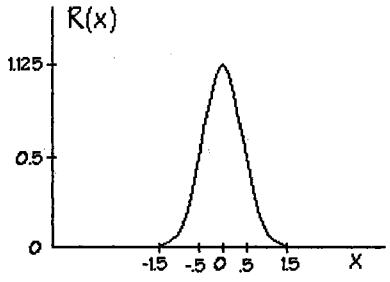


Figure 2-6.
The Bell Curve

If this process of convoluting rectangle functions is continued, the eventual result is a gaussian shaped bell curve, shown in Figure 2-7.

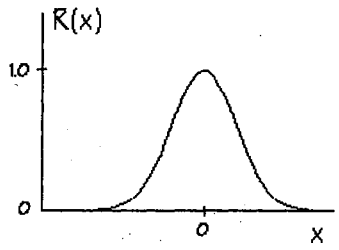


Figure 2-7.
The Gaussian Curve

However, there are a few more functions of interest before the convergence to this curve.

Second order polynomials are shown to be space variant [Schafer 73], [Abdou 82] and exhibit a phase distortion. In fact, this problem exists with all even number polynomials. Therefore, even powered polynomials are not used for interpolation. Further, higher order polynomials may not converge, so these are not used. However, higher order polynomials can be approximated by using a low order polynomial on repeated subintervals.

Cubic B-Spline Interpolation

If rectangle functions are continuously convolved together a B-spline will be created [Hou 87], [Lee 83]. Although the general n-degree B-spline is defined as $B_n = B_0 * B_{n-1}$, the cubic B-spline the most useful here. A B-spline of degree one is the triangle

function, degree two is the bell function, and degree three is the cubic B-spline, as shown in Figure 2-8.

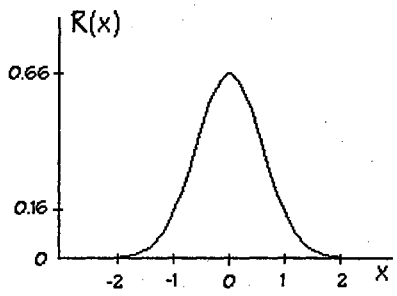


Figure 2-8.
The Cubic B-Spline

In interpolation applications, a B-spline is used to join data samples into a continuous function. Interpolation with a first degree B-spline (the triangle) is to join the data samples with straight lines (linear interpolation). Second order B-splines (bell functions) join the data samples with parabolas, with the span limited to three samples. Cubic B-splines are typically limited to matching four data samples, as shown in Figure 2-9.

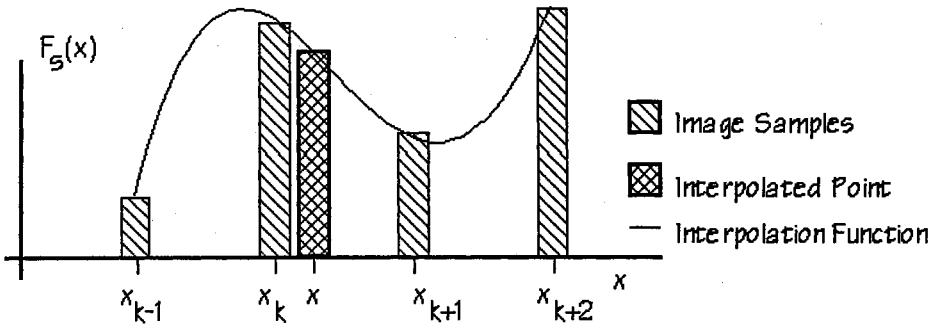


Figure 2-9. Cubic B-Spline Interpolation

Cubic B-splines are used in image interpolation because the first and second order derivatives are continuous, and they provide smoothing. The cubic B-spline is defined as

$$\begin{aligned} R(x) &= \tfrac{2}{3} + \tfrac{1}{2}|x|^3 - (x)^2, \quad 0 \le |x| \le 1 \\ &= \tfrac{1}{6}(2 - |x|)^3, \qquad 1 \le |x| \le 2 \end{aligned} \tag{2-15}$$

Cubic Convolution

Rifman and McKinnon [Rifman 74] originally suggested the cubic convolution algorithm as an approximation to the sinc function. It was developed at TRW to help in the reconstruction of Landsat digital images and is known as the TRW cubic algorithm.

It has been shown [Pratt 91] that if the digital image is band-limited and sufficiently sampled, then it can be reconstructed completely by using the ideal interpolation function from Equation 2-9,

$$\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x}. \tag{2-16}$$

The spatial sinc function in Equation 2-17 corresponds to the rectangle function in the frequency domain. The rectangle function is a low-pass or band-pass filter that does not introduce any distortions inside the passband.

However, Equation 2-17 is not a practical function to implement in the spatial domain, due to its infinite extent. In practice, the number of samples used to reconstruct one new sample point must be limited, and a limited approximation to the sinc function is needed. If the sinc function is limited to five points, there is a slope discontinuity at the end points, -2, and 2 [Park 82a]. This produces ripples in the frequency spectrum. Cubic convolution [Keys 81] is an attempt to eliminate these ripples by making the slope zero at the end points. This produces an approximation to the sinc function in the area of interest

(-2 < x < 2), while preserving the continuous end points. This cubic curve, which is actually a range of piecewise continuous functions, is given by

$$\begin{aligned} R(x) &= (\alpha+2)|x|^3 - (\alpha+3)|x|^2 + 1, && |x| < 1 \\ &= \alpha|x|^3 - 5\alpha|x|^2 + 8\alpha|x| - 4\alpha, && 1 \le |x| < 2 \\ &= 0 && \text{otherwise.} \end{aligned} \tag{2-17}$$

The parameter α is an adjustable variable. This parameter corresponds to the slope at x=1. The ideal sinc function has a slope of -1 at x=1, thus the parameter α is usually set to -1 to duplicate the ideal function. However, Park shows that a choice of -0.5 is actually a better choice than either the standard -1, or -0.75, which is also used at times. This choice of α gives a frequency response superior to nearest neighbor, and linear interpolation. This method can also be incorporated into an adaptable scheme, where the parameter α is based on local image statistics. In particular, for an image with edges as its main point of interest, Park suggests a value of -0.666 is an optimal choice for minimizing error.

Filter Choice

Several authors have studied various filters used in the reconstruction of discrete images [Maeland 88], [Mitchell 88], [Park 82a], [Park 82b], [Naiman 87]. Schreiber and Troxel [Schreiber 85] emphasized the importance of a perceptual based evaluation of the filters. In this study, which compared the nearest neighbor, truncated sinc, linear, B-spline, gaussian, and a sharpened gaussian, the results showed a subjective preference for the sharpened gaussian filter. This study is of particular interest because it emphasizes a real world approach. In the sharpened gaussian case, separable filter were cascaded together to form a combined sharpened gaussian function. The sharpening portion of this

filter is used to emphasize the high frequency edges in an image, mimicking the human eye. By using a separable filter, the authors kept the computations relatively short.

Coding

There are many different methods to encode image data. This research is concerned with the two types used in the JPEG DCT image compression scheme. The JPEG algorithm uses transform coding (DCT-based) to encode the image samples, and uses predictive coding to encode transform coefficients.

Predictive Coding

Predictive coding makes use of redundancy in the data. In image compression, redundancy is highly dependent on the type of image being compressed. For example, an image of a constant blue sky will be highly predictable, as will an image of a starfield. Of course, most images of interest are much more random. In predictive coding, the algorithm can either base its predictions on some fixed value, or on a adaptive value. Adaptive values commonly include the average pixel value, the previous pixel value, or a local average pixel value.

Most images contain some form of structure to them. For example, a natural scene image may contain a man-made building, a person, a tree, or other objects which are likely to contain similar colors and textures. Within that area of the image, the pixels may be highly predictable. However, when considered as a whole, the image may not contain elements that lead themselves to prediction. An adaptive algorithm [Arps 88] always makes more accurate predictions because of this. However, the tradeoff in algorithm complexity and speed may make a non-adaptive algorithm more suitable.

Transform Coding

The Fourier transform discussed above and other frequency space transforms, such as the cosine transform, are used in image processing for many different reasons. The use of transforms in image coding first began about 1970 [Chen 84], [Wintz 72]. However, because of the computational difficulties and resulting large delays, transform image coding did not receive much attention. Recently, due in part to advances in digital computers, there has been renewed interest in using frequency transforms for image compression and coding. It has proven to be an efficient means of image compression [Lohscheller 84]. The algorithm for computing the discrete cosine transform (DCT) has been an area of concentrated research over the last twenty years [Duhamel 90], [Chen 77], [Narasinha 78], [Ahmed 74], [Suehiro 86], [Lee 84] and [Vetterli 84]. Some of this research is directly applied to computing small (16 by 16, and 8 by 8) subblocks of two dimensional image transform data. The resulting speed improvements have helped make the DCT an acceptable solution for time-constrained image compression problems.

In a typical image coding algorithm, the original image is divided into small subblocks, 8 or 16 pixels per side. Each block undergoes a two-dimensional transformation, producing an equal size block of transform coefficients. Each block of coefficients is converted into an one-dimensional array, and then quantized and coded. Generally the low frequency components are quantized most finely, and the higher frequency coefficients are quantized more coarsely. Finally the quantized coefficients may be compressed further using some form of predictive coding, usually Huffman coding or arithmetic coding [Langdon 84], [Pennebaker 88].

The JPEG Draft

Background

The Joint Photographic Experts Group (JPEG) draft [JPEG Draft 92] is the accepted method for compression of natural scene images, and is likely to become an ISO standard. It includes four modes of compression; sequential, progressive, lossless, and hierarchical. In sequential encoding, each image pixel is encoded in a single left-to-right, top-to-bottom stream. Progressive encoding operates across the image in multiple scans, allowing the viewer to watch the restoration in multiple passes. The lossless encoding mode sacrifices compression ratio for image quality, guaranteeing an exact recovery of the compressed image. Finally, hierarchical encoding creates multiple images of differing resolutions, so that low resolution versions may be accessed without decompressing the full image at high resolution.

The lossy method includes the discrete cosine transform (DCT), to be formally defined below. The simplest process is called the baseline sequential process. The coding pathway shown in Figure 2-10 starts by grouping the image data into 8x8 blocks of a single color component. Each color component of the image is handled independently, and there can be up to 255 separate components. Each 8x8 block is then transformed into a set of 64 discrete cosine coefficients with the forward DCT (FDCT).

The coefficients are quantized using a predetermined quantization table, and are then prepared for the entropy encoder. The coefficients are passed through the one of two possible entropy encoders (either a Huffman encoder or an arithmetic encoder). Both again use a predetermined table (not specified in the JPEG draft).

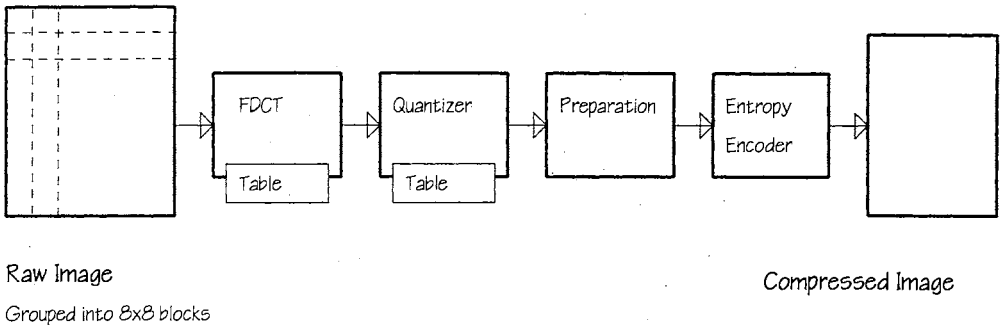


Figure 2-10. The JPEG Compression Process

The decoding steps in Figure 2-11 are essentially the inverse of the coding steps. The compressed image data is first passed through the entropy decoder. The coefficients are reorganized into the proper order (2-D), and the difference encoding is reversed. The data continues to the dequantization step, which converts the data back into DCT coefficients. Finally the inverse DCT reconstructs the 8x8 block of image data.

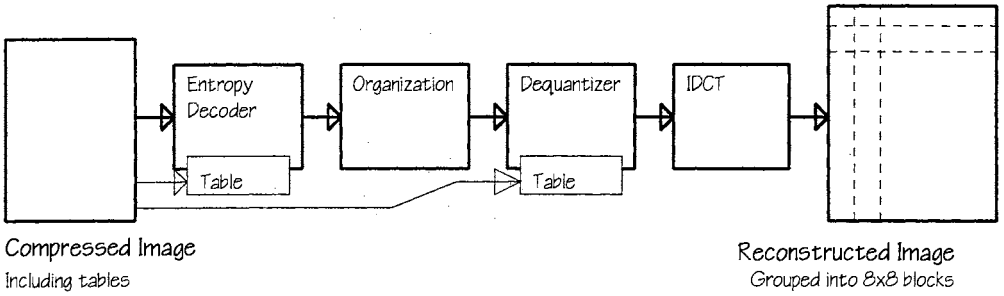


Figure 2-11. The JPEG Decompression Process

Forward and Inverse Discrete Cosine Transform

The FDCT used in the JPEG draft is:

$$\mathcal{F}(u,v) = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} F(x,y) \cos\left(\frac{u\pi}{16}(2x+1)\right) \cos\left(\frac{v\pi}{16}(2y+1)\right) \qquad (2\text{-}18)$$

and IDCT used in the JPEG draft is:

$$F(x,y)=\frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7}C_u C_v \mathcal{F}(u,v)\cos\left(\frac{u\pi}{16}(2x+1)\right)\cos\left(\frac{v\pi}{16}(2y+1)\right) \quad (2\text{-}19)$$

where:

$$C_u = C_v = \frac{1}{\sqrt{2}} \text{ for } u,v = 0$$
$$\text{and } C_u = C_v = 1, \text{ otherwise.}$$

Quantization and Dequantization

A single DCT coefficient, $\mathcal{F}(u,v)$, is quantized by a uniform quantization formula:

$$\mathcal{F}_q(u,v) = \text{round}\left(\frac{\mathcal{F}(u,v)}{Q(u,v)}\right). \quad (2\text{-}20)$$

The step size Q(u,v) comes from the quantization table, and rounding is done to the nearest integer. $\mathcal{F}_q(u,v)$ is the quantized DCT coefficient. Inside the decoding process, the normalization is removed by an inverse process:

$$\mathcal{F}_{uq}(u,v) = \mathcal{F}_q(u,v) \times Q(u,v) \quad (2\text{-}21)$$

where $\mathcal{F}_{uq}(u,v)$ is the unquantized DCT coefficient.

Data Preparation

The first (DC) coefficient of the block is encoded with the difference equation:

$$DIFF = DC - PRED \quad (2\text{-}22)$$

The PRED value is the unquantized value of the DC component of the preceding block, $\mathcal{F}_{uq}(0,0)$. The remaining AC components are arranged in a zig zag sequence as show in Table 2-1.

TABLE 2-1

ORGANIZATION OF COEFFICIENTS

0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

Compression Ratio and Image Quality

All frequency transform-based compression schemes can produce varying levels of image quality. Basically, the more high frequency components that are discarded in the quantization step, the better the compression, at the expense of the image quality. For DCT-based compression schemes, the levels shown in Table 2-2 have been found by [Wallace 91], [Leger 91] and [Mitchell 89] to be a consistent framework for image quality measurements.

TABLE 2-2

COMPRESSION VERSUS QUALITY

Bits Per Pixel	Subjective Quality
0.25 to 0.5	moderate to good, sufficient for some applications
0.5 to 0.75	good to very good, sufficient for many applications
0.75 to 1.5	excellent, sufficient for most applications
1.5 to 2.0	usually indistinguishable from the original, sufficient for the most demanding applications

From this information it is possible to predict the amount of compressed data needed for a given level of quality. For a quick example, a image 512 by 512 and 24 bits deep (6.3 Megabytes) could be reduced to 65 Kilobytes, while maintaining a moderate image quality. For the JPEG algorithm, this is an intermediate result. The DCT coefficients are further compressed using the zig zag organization and Huffman coding. According to [Mitchell 89], this allows for a ISDN network to display a recognizable image in less than one second. This assumes the decompression can be done in real time, which is 64 Kilobits per second for ISDN. An excellent quality image would be achieved in 5 to 10 seconds, and a visually indistinguishable image in 20 seconds.

CHAPTER 3

RESEARCH OUTLINE

Background

Traditional image reconstruction begins with a digitized image and includes the following steps: First, interpolation between the original image samples recreates the continuous image signal, then scaling of this continuous signal produces an image of the correct size. Finally resampling the scaled continuous image creates a new digitized image. The traditional method is shown in Figure 3-1a. Figure 3-1a includes the IDCT operation for later comparison.

This research examines the possibility of performing the interpolation and scaling operations in the frequency domain, rather than the spatial domain. This idea fits nicely with the JPEG image compression algorithm, because the JPEG algorithm uses the discrete cosine frequency transform to compress the image. The goals of this research are first to determine if this approach is feasible, and second, if feasible, to determine what benefits this approach can offer. The new frequency reconstruction method shown in Figure 3-1b. The new method uses the transformed image data and produces a scaled, resampled image stream by applying the reconstruction process to the DCT transform coefficients.

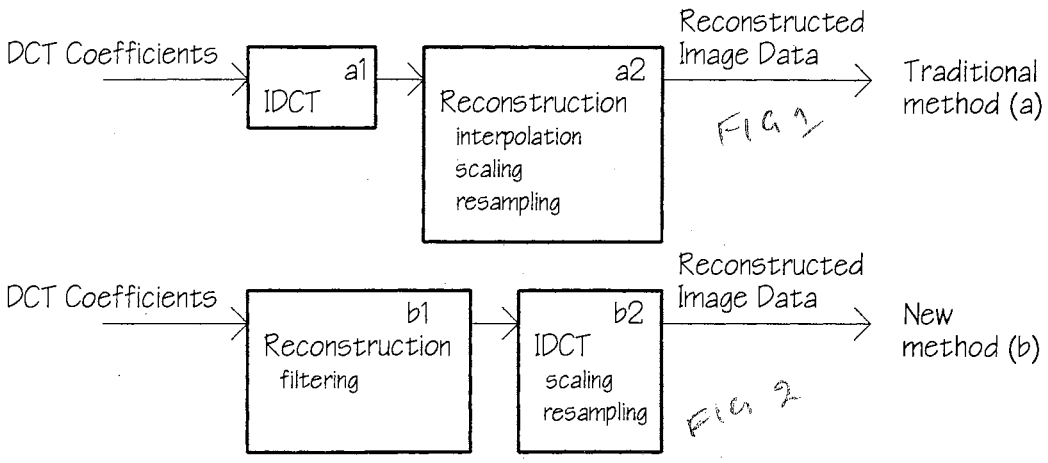


Figure 3-1. Reconstruction Methods: a) Traditional Method; b) New Method

If the reconstruction can be done in the frequency domain as a multiplication (filtering), rather than in the spatial domain as a convolution, the resulting reconstruction algorithm will be much faster. The new cosine domain-based reconstruction algorithm requires three related operations. They are filtering, scaling, and resampling. The filtering operation multiplies the DCT coefficients with the reconstruction kernel. The scaling operation adjusts the frequencies used in the IDCT operation. The resampling operation samples the scaled DCT basis frequencies at the new sample locations. Of the three steps in the process, the filtering operation is the only separate module. The scaling and resampling steps are integrated into the IDCT process.

Spatial Implementation

The first phase of the research covers the spatial implementations of the cubic, and the sharpened gaussian reconstruction functions. These two spatial functions are well documented in previous research literature, and were chosen because of their wide acceptance as high quality image interpolation functions. Although both functions are used in a two-dimensional manner, both are implemented in a one-dimensional fashion.

The separable implementation matches the previous research in the literature. A separable function operates in a two-pass method, the data is interpolated first the horizontal direction, then in a vertical direction. The result is a rectangular two-dimensional interpolation function. These spatial implementations, developed in Chapter 4, are needed for error analysis in later steps, as well as implementation guides during the frequency domain development presented in Chapter 5.

As part of the first phase to develop a suitable reconstruction process, the interpolation function will be combined with the scaling process. The interpolation function is used to interpolate between samples of the original image and the second function, the scaling function, scales the image to the correct size. The combined reconstruction function will contain an adjustable parameter, ζ, to specify the degree of scaling (either spatial magnification, $\zeta>1$, or spatial minification, $\zeta<1$) desired. Details of the spatial implementation is presented in Chapter 4.

Frequency Implementation

The second phase covers the frequency domain investigations, starting in the Fourier frequency domain, then extending the results to the discrete cosine frequency domain. The purpose of the Fourier work is to provide a basis of understanding and comparison, since little research of this nature is available in the cosine domain. The frequency phase of the research, presented in Chapter 5, produces two individual reconstruction filters (cubic and sharpened gaussian) which are implemented and documented in the discrete cosine domain, as well as the Fourier domain for reference. As part of the Fourier analysis, the ideal reconstruction filter is also examined. The ideal filter is used to calculate and predict the cutoff frequencies used to prevent aliasing. Using the ideal filter in the cosine domain, an ideal image of the correct size is generated and used as a basis for numerical error analysis.

The result of the frequency domain implementation phase is three filters, the combined cubic-scaling filter, the combined sharpened gaussian-scaling filter, and the ideal-scaling filter, all implemented and documented in the cosine domain. These functions are independent of any specific implementation which constrains the frequency range. In particular, they do not depend on the JPEG implementation of an 8x8 DCT block, which limits the possible frequencies to those represented by the 64 DCT frequency coefficients.

Application to the JPEG Data Stream

The third phase, presented in Chapter 6, demonstrates the use of these filters during JPEG decompression. The JPEG algorithm constraints the range of frequencies that are available to the reconstruction process because it uses an 8x8 block of coefficients. Only 64 discrete frequencies are allowed and the higher frequencies are set to zero. During this phase, two well known test images will be used; the Lena and mandrill images. A black and white reproduction of each image is shown in Figures 3-2 and 3-3.

Figure 3-2. The Lena Test Image

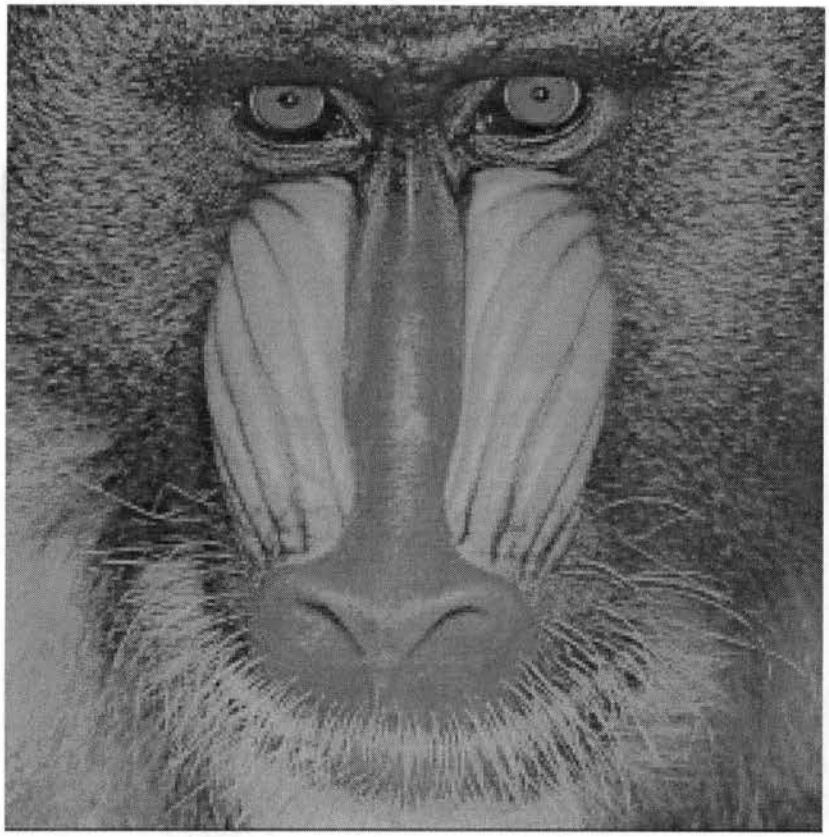


Figure 3-3. The Mandrill Test Image

Much of Chapter 6 is devoted to a discussion of differences (both subjective and quantitative) between the spatial and frequency versions of the images. This includes a discussion of the impact of filtering out the high frequencies (anti-aliasing) during the frequency reconstruction. This chapter includes a documentation of the implementation differences between the spatial and frequency research, primarily centering on the differences in circular and rectangular filters. Speed comparisons are also discussed for each process path. Finally, in the error analysis section of Chapter 6, the numerical errors are plotted for varying scaling factors, for both the Lena and mandrill images.

Evaluation

The research summary in Chapter 7 presents conclusions found in this research. The conclusions are largely subjective for two reasons. First, when applying the magnification and minification algorithms, the original correct image may not be available for any error analysis, although if possible, the ideal filter is used for this purpose. Secondly, there is no single quantitative measure that mimics the human visual system, the eventual receiver.

For example, in similar works [Ratzel 80, Ready 72] it is suggested that a numerical error evaluation, such as a minimum mean square error (MSE), may not be an appropriate choice when the actual receiver is the human eye. As an example, consider a small grayscale image with a range for each pixel from 0 to 255. If the value of 1 is added to every pixel value, the resulting MSE will be quite large, however the visual effect is not too disturbing. The same MSE results if 1 is added to every even numbered pixel in even numbered lines, and odd number pixels in odd numbered lines, and 1 is subtracted from all other pixels. This results in a checkerboard pattern that is more disturbing to the human visual system, yet yields the same MSE value.

Summary

Using the convolution and scaling properties of the discrete cosine frequency transform, this research implements and evaluates a new method of manipulating the transform data directly, before the final inverse transformation of the image data is accomplished. Three operations (filtering, scaling, and resampling) are combined into a single, adjustable reconstruction process.

CHAPTER 4

SPATIAL IMPLEMENTATION

Notation

Image Coordinates

This research uses three coordinate systems in the description of an image. It is useful to use three different sets of variables to help clarify these descriptions. First, the normal, unscaled image is described using x (horizontal) and y (vertical) dimensions, as shown in Figure 4-1. Each pixel represents one unit in these coordinates. For purposes of the JPEG DCT, this means that x and y are limited to the block size, N=8. The variables x and y then vary between 0 and 7.

In the discrete cosine frequency space, the frequency coordinates are u and v. When referring to the JPEG DCT algorithm, the DCT is limited to an 8 by 8 block, so u and v also vary from 0 to 7. In the Fourier frequency domain, the variables ω_1, and ω_2 are used to distinguish the Fourier frequency space from the cosine frequency space.

The third description of an image is the scaled IDCT output space (the reconstructed image space). This output space is given coordinates r and s. This coordinate system is the same as x and y when the scaling is 1. However, when the scaling is not 1, the new scaled space does not have the same range as the original 8 by 8 block. Instead, each block is scaled to a new size, described in terms of r and s. As indicated in Figure 4-1, the range or block size (N) is 8 for the JPEG DCT. The scaled (reconstructed) block size is M = round(Nζ), where ζ is the scaling factor discussed in Chapter 2.

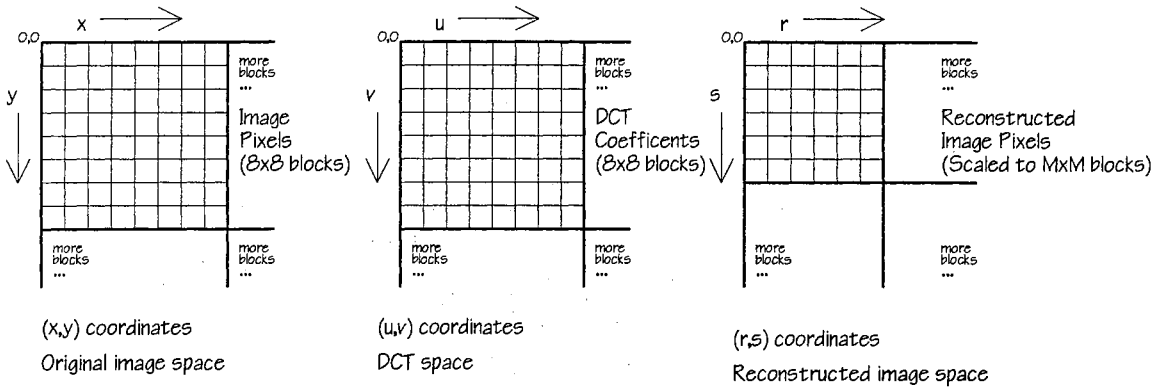


Figure 4-1. Coordinate Spaces

Interpolation

Interpolation functions use values of surrounding sample points to estimate the value of an intermediate interpolated point between the samples. Separable interpolation functions are implemented in a one-dimensional manner, and operate in a two-pass fashion. In this research, the spatial reconstruction functions are implemented in a separable fashion. The first pass interpolates horizontally and produces four intermediate points, the second pass interpolates in the vertical direction using the intermediate points and produces the final interpolated point. Separable functions are advantageous for speed of computation and simplicity during the implementation phase.

When the interpolated point lies directly on an original data sample, it is usually assumed the interpolated point will exactly match the original sample. However, in the case the sharpened gaussian, the interpolated point will not match the original data, but will produce a sharpened version of the data set instead. In the case of the cubic function, the new interpolated point will exactly match the original data.

Interpolation functions have the general form

$$g(x) = \sum_k c_k z(x - x_k), \tag{4-1}$$

where g(x) is the interpolating function applied to a sampled function f(x) (the sampled image), x_k corresponds to the sample data points, z is the interpolation function (i.e., the cubic function). The c_k's are parameters based on the sampled data, so that $g(x_k) = f(x_k)$ for each x_k. The sampling increment is assumed to be normalized to one.

Parametric Cubic Convolution

Conditions and Restrictions

The parametric cubic kernel, described in Chapter 2, contains four segments, (-2, -1), (-1,0), (0,1), and (1,2) [Rifman 74] [Park 82a] [Keys 81]. It is defined to be zero outside this interval. However, since the kernel must be symmetric to avoid phase shifts, the domain can be limited to (0,1) and (1,2), and reflected about the zero axis. With this simplification, the kernel has the form:

$$z(x) = \begin{cases} A_1x^3 + B_1x^2 + C_1x + D_1, & 0 < x < 1 \\ A_2x^3 + B_2x^2 + C_2x + D_2, & 1 < x < 2 \\ 0, & 2 < x. \end{cases} \tag{4-2}$$

where x is measured as an offset distance from the interpolated point.

By imposing additional conditions on the interpolation kernel, z(x), the coefficients in Equation 4-2 can be established. Since the value of the kernel function must be equal to the sampled data at the interpolated point when the offset is zero;

$$\begin{aligned} z(x) &= 1, \quad x = x_k \\ z(x) &= 0, \quad x \neq x_k \end{aligned} \tag{4-3}$$

thus, using x_k instead of x,

$$g(x_k) = \sum_k c_k z(x_k - x_k) \tag{4-4}$$

Since z(x)=0 unless x=0, and z(x)=1 at x=0, then $g(x_k) = c_k$. Combining this result with the requirement $g(x_k) = F(x_k)$, the end result is

$$c_k = F(x_k). \tag{4-5}$$

And:

$$\begin{aligned} 1 &= z(0) = D_1 \\ 0 &= z(1^-) = A_1 + B_1 + C_1 + D_1 \\ 0 &= z(1^+) = A_2 + B_2 + C_2 + D_2 \\ 0 &= z(2^-) = 8A_2 + 4B_2 + 2C_2 + D_2 \end{aligned} \tag{4-6}$$

Park [Park 82a] also imposes that the interpolation kernel, z(x), has a continuous derivative, z'(x), for all x. Keys [Keys 81] translates this to the cubic family as:

$$\begin{aligned} -C_1 &= z'(0^-) = z'(0^+) = C_1 \\ 3A_1 + 2B_1 + C_1 &= z'(1^-) = z'(1^+) = 3A_2 + 2B_2 + C_2 \\ 12A_2 + 4B_2 + C_2 &= z'(2^-) = z'(2^+) = 0. \end{aligned} \tag{4-7}$$

The first condition implies C_1=0. Putting the results of Equation 4-6 and Equation 4-7 in matrix form gives

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 8 & 4 & 2 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & 2 & 1 & 0 & -3 & -2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 12 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} A_1 \\ B_1 \\ C_1 \\ D_1 \\ A_2 \\ B_2 \\ C_2 \\ D_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \alpha \end{bmatrix} \tag{4-8}$$

This gives seven equations in eight unknowns plus the relation $A_2 = \alpha$. For this last equation, a range of values has been proposed for α. The value A_2 corresponds to the slope at x=1. Rifman [Rifman 74] originally proposed a value of A_2=-1, to match the sinc function. However, Keys [Keys 81] selects a value of -0.5, to make the Taylor series expansion of R(x) in Equation 2-17 match the interpolation function g(x) to as many terms as possible. The same value, -0.5, was used in this implementation. The result is a function based on the four surrounding points, the parameter α, and x, the offset from the interpolation point.

$$\begin{aligned} g(x) = &-\left[\alpha(c_{k+2} - c_{k-1}) + (\alpha+2)(c_{k+1} - c_k)\right]x^3 + \\ &\left[2\alpha(c_{k+1} - c_{k-1}) + 3(c_{k+1} - c_k) + \alpha(c_{k+2} - c_k)\right]x^2 - \\ &\left[\alpha(c_{k+1} - c_{k-1})\right]x + c_k \end{aligned} \tag{4-9}$$

Equation 4-9 can be rewritten as a reconstruction kernel, R(x), by removing the sample data points, and separating the two functions, the first from 0 to 1, the second from 1 to 2, then reflecting these two functions about the axis. This is the same as Equation 2-18.

$$
\begin{aligned}
R(x) &= (\alpha+2)|x|^3 - (\alpha+3)|x|^2 + 1, && |x| < 1 \\
&= \alpha|x|^3 - 5\alpha|x|^2 + 8\alpha|x| - 4\alpha, && 1 \le |x| < 2 \\
&= 0 && \text{otherwise.}
\end{aligned}
\tag{4-10}
$$

Sharpened Gaussian

Conditions and Restrictions

The sharpened gaussian function is a combination of three gaussian functions, one central and two side lobes. The two side lobes have a negative gain, and are displaced by the amount of scaling. The gains of the central and side lobes are arbitrary set to provide a pleasing visual result. This is a change from the cubic function, where the gain is determined by minimizing numeric difference between the sinc function. The sharpened gaussian is designed by subjective measures, where the cubic is designed by numeric measures.

Ratzel [Raztel 80] investigates the subjective quality of the image versus filter width. In particular, he studies the tradeoff between filter width in the frequency domain versus filter width in the spatial domain. One of the results of this paper is that a filter width given by σ=0.375 gives good results.

This value of $\sigma = 0.375$ is used in this implementation of the sharpened gaussian filter for the central and for the side lobes. However, the gains for the central and side lobes are different than the gains Schreiber [Schreiber 85] or Ratzel used. Schreiber starts out with the same gaussian parameters, however, he implements it in a discrete fashion. First he convolves the entire image with a sharpened filter given by

$$
\begin{bmatrix} 0 & -7 & 0 \\ -7 & 49 & -7 \\ 0 & -7 & 0 \end{bmatrix}. \tag{4-11}
$$

This removes most of the intersample interference, or dependence. After the entire image is sharpened in this manner, a two pass convolution, first horizontal, then vertical, is used as the interpolation step. The discrete interpolation filter he uses is

$$[\ldots\ 0\ 1\ 3\ 6\ 11\ 16\ 20\ 21\ 20\ 16\ 11\ 6\ 3\ 1\ 0\ \ldots]/135 \qquad (4\text{-}12)$$

which is his approximation to a central gaussian. The data is scaled to reduce the gain introduced by the interpolation and sharpening filters.

Schreiber results are used in this research, the same value for σ is used, but the two functions are combined into one, and a constant test image is used help fine-tune the gains of the central and the side lobes. The resulting filter is shown in Figure 4-2. It is identical to the previous sharpened gaussian filters, except it is a single continuous filter, and the adjusted gains used here eliminate the final scaling step done in Schreiber's work.

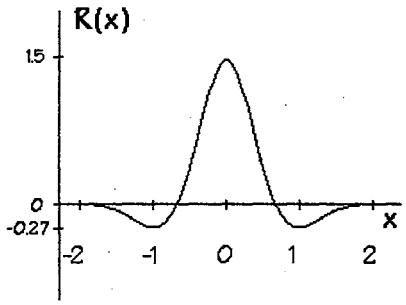


Figure 4-2.
Sharpened Gaussian

The central lobe is described as

$$\text{Central}(x) = \frac{1.5}{2.5066 \times 0.375} \exp\left(\frac{-0.5x^2}{(0.375)^2} \right) \tag{4-13}$$

and the two side lobes are

$$\text{Side}(x) = \frac{-0.27}{2.5066 \times 0.375} \exp\left(\frac{-0.5x^2}{(0.375)^2} \right). \tag{4-14}$$

These are consistent with the modified gaussian curves presented by Ratzel, and Schreiber. The final curve shown in Figure 4-2 is a result of adding the three curves together. The central gain, 1.5, and the side lobe gain, -0.27, were selected using a constant test image, to be discussed in the error analysis section.

Error Analysis

To test and adjust these functions and their parameters (particularly the gain of the sharpened gaussian function), different two-dimensional functions were used to simulate the image data. The first function used was a constant image (each pixel value equal to 100). The other artificial image used was a linearly varying image. Each pixel value was computed with the function

$$\text{Pixel Value} = (\text{Row Number}) + 10 \times (\text{Column Number}). \tag{4-15}$$

Using these functions as test data, the cubic function performed as expected, and needed no further modifications. However, the gains of the sharpened gaussian were adjusted to 1.5 and -0.27 to produce a subjective minimum distortion using the constant test image. In this research, there is a slight distortion introduced by round off errors. A four point window is used, which can lead to a phase distortion when centered the interpolated point lies on a sample data point. This problem is caused by the right most point of the window does not evaluate to zero. This is shown in Figure 4-3a. The four dark vertical lines in both figures represent the sample data points, x_{k-1}, x_k, x_{k+1}, and x_{k+2}.

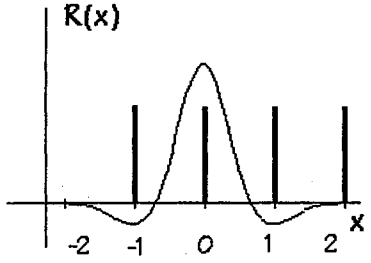


Figure 4-3a
On-Center Interpolation

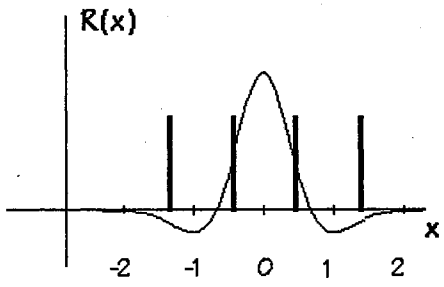


Figure 4-3b
Off-Center Interpolation

When the interpolated point is not centered on a data sample, x_k, as in Figure 4-3b, there is the balancing left sample, x_{k-1}. However, in Figure 4-3a, the right most sample point, x_{k+2}, does not evaluate to zero because the interpolation kernel, R(x), is not quite zero. This amount of distortion turns out to be significant. On the constant image test, this right most sample contributed a distortion of 3.6 percent. For separable horizontal and vertical implementations this is magnified because of the two pass implementation. The first pass contributes 3.6 percent, then the second pass adds another 3.6 percent. One method of compensating for this would be to decrease the width of the filter. To match Schreiber's work on spatial versus frequency filter width tradeoffs, the filter width was not changed. However, this idea could be subject to subsequent study. A typical interpolated (one dimensional) line is below in Figure 4-4, showing these ripple-like distortions. The odd pixel locations are the on-center values, the even are the off-center interpolated values.

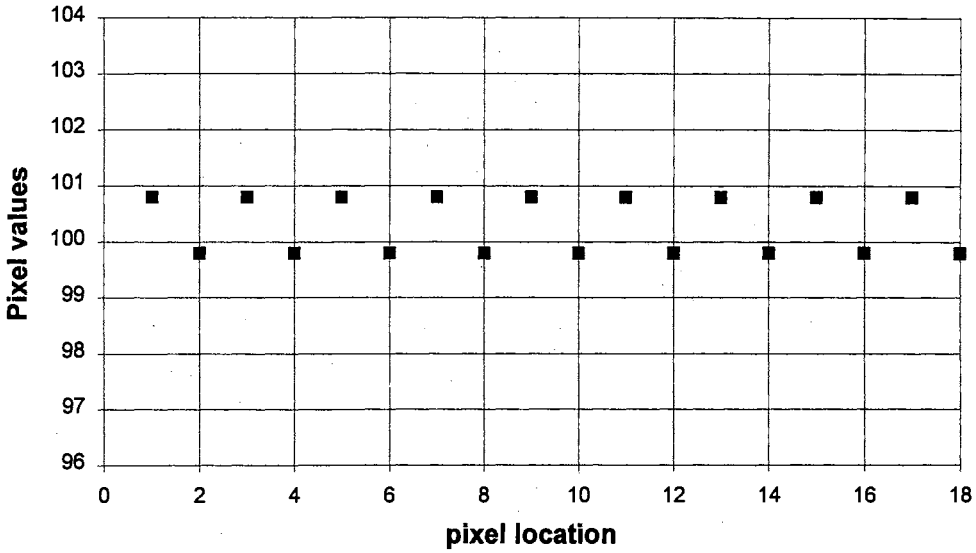


Figure 4-4. Sample Sharpened Gaussian Response

Using these values, a numerical estimate for the error can be computed. Both the on and off center interpolated are supposed to have a value of 100. The error can be measured against this artificial image.

$$\text{error} = \frac{\sqrt{(100 - 100.8)^2 + (100 - 99.8)^2}}{2} = 0.4123 \tag{4-16}$$

Summary

This chapter covers the first phase of the research, the spatial implementation of two interpolation kernels, the parametric cubic convolution, and the sharpened gaussian. This phase is important because it gives a method of comparing this research to other research in the literature. It will also be used later to compare the spatial implementation to the frequency implementation. Deviations from the literature include a one-pass

combined method, rather than the two-pass sharpen-interpolate scheme. Both functions perform 2 dimensional interpolations, first horizontally, then vertically.

Because of the on-center/off-center problem, the implementation of the sharpened gaussian function gave an error based on the scaling value. If the function is aligned (on-center) with a data sample, the error is about 3.5 percent greater than if the function is half way between the data samples (off-center).

CHAPTER 5

FREQUENCY IMPLEMENTATION

JPEG DCT Definition

The JPEG draft [JPEG Draft 92] specifies the forward and inverse discrete cosine transforms to be used for image compression and decompression. Recalling Equations 2-18 and 2-19, the forward (FDCT) and inverse (IDCT) transforms are defined as

$$\text{FDCT:} \quad \mathcal{F}_s(u,v) = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} F_s(x,y) \cos\left(\frac{u\pi}{16}(2x+1)\right) \cos\left(\frac{v\pi}{16}(2y+1)\right)$$

$$\text{IDCT:} \quad F_s(x,y) = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C_u C_v \mathcal{F}_s(u,v) \cos\left(\frac{u\pi}{16}(2x+1)\right) \cos\left(\frac{v\pi}{16}(2y+1)\right) \quad (5\text{-}1)$$

$$\text{where } C_u, C_v = \frac{1}{\sqrt{2}} \text{ for u or v} = 0; \quad C_u C_v = 1 \text{ otherwise.}$$

where an 8x8 block of image samples is assumed.

Scaling

Using the linear scaling property, the original image transform data is scaled to another size. By contracting the two axes in the frequency domain, the spatial axes are expanded. Likewise, by expanding the frequency axes, the spatial axes are contracted. This relationship [Jain 89] between a spatial function and its frequency transform is expressed as

$$F(\zeta x,\zeta y) \Leftrightarrow \frac{1}{\zeta^2}\mathcal{F}(\frac{u}{\zeta},\frac{v}{\zeta}). \tag{5-2}$$

where ζ is the scaling factor, and $\zeta > 1$ produces spatial magnification. The two axis could be scaled independently by different amounts, but this research makes use of a single scaling factor, corresponding to the scaling factor applied to the entire image.

By inserting this scaling property into the inverse discrete cosine process, the resulting spatial data is, in effect, scaled to the correct size, or sampling rate.

$$F_S(\zeta x,\zeta y) \Leftrightarrow \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7}\frac{C_u C_v}{\zeta^2}\mathcal{F}_S(u,v)\cos\left(\frac{\pi u}{16\zeta}(2x+1)\right)\cos\left(\frac{\pi v}{16\zeta}(2y+1)\right) \tag{5-3}$$

The impact of scaling the basis frequencies is illustrated in Figures 5-1 and 5-2. These figures show plots of the cosine terms in the summation

$$\sum_{u=0}^{7}\cos\left(\frac{\pi u}{16\zeta}(2x+1)\right) \tag{5-4}$$

for three different values of x and two different values of ζ. Figure 5-1 shows the case of ζ=1; the original, unscaled case.

As the scaling factor ζ is increased to 1.25 as in Figure 5-2, the frequency axis is effectively compressed. This inverse relationship between the spatial axis and the frequency axis is used to scale the image data in this research.

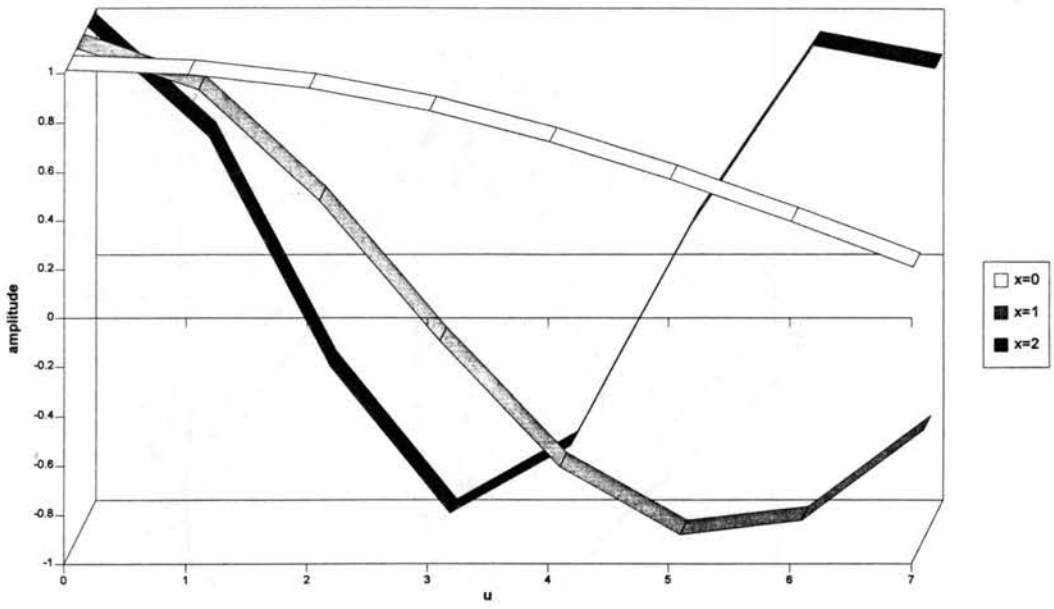


Figure 5-1. Unscaled Cosine Frequencies, ζ=1

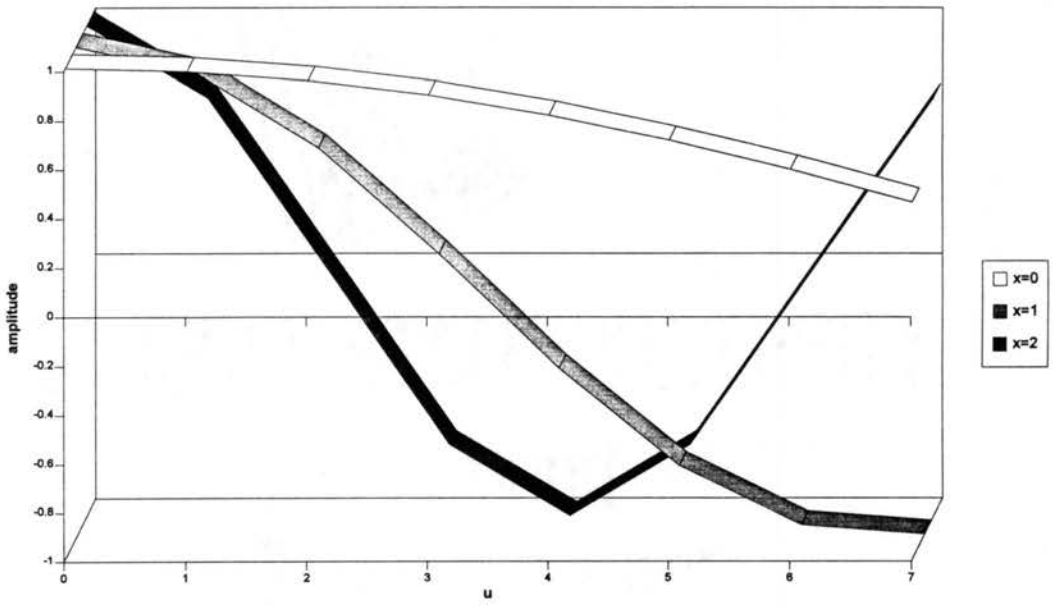


Figure 5-2. Scaled Cosine Frequencies, ζ=1.25.

By scaling the frequencies in the discrete cosine transform, the image in the spatial domain is effectively scaled in the inverse manner.

As an example, consider the spatial magnification scaling process shown in Figure 5-3.

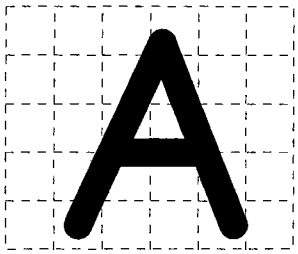


Figure 5-3a.
The Continuous Image.

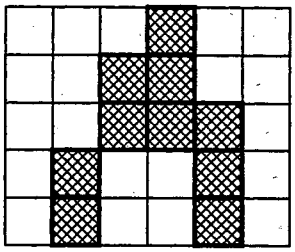


Figure 5-3b.
The Initial Sampled Image.

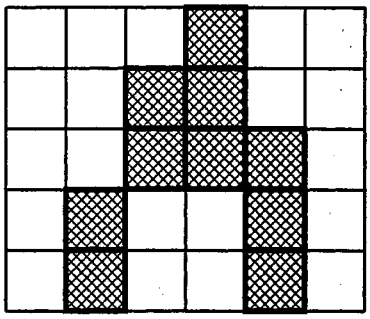
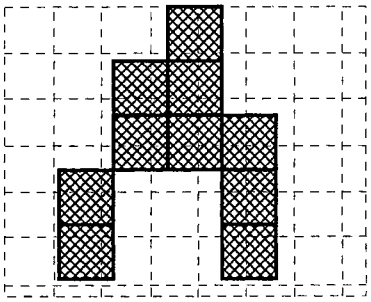


Figure 5-3c. The Scaled (Magnified) Sampled Image.



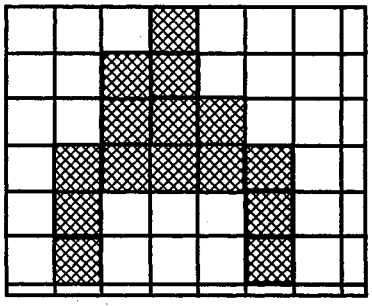


Figure 5-3d. Resampling the Scaled Image to Fit the Output Requirements.

Figure 5-3a and Figure 5-3b show the original continuous and sampled image for a 6x5 pixel array for a display device of a given size. The scaling step in Figure 5-3c scales

both spatial axes uniformly. In Figure 5-3b, the first sampling step, the image is sampled to 6x5 pixels, as it is in Figure 5-3c. However the size of each pixel is larger in Figure 5-3c than 5-3b. The image is still 6x5, but each unit has been scaled. In Figure 5-3d, the units are returned to the original size, but now the image is resampled to 7.5x6.25 pixels. The expansion by 25 percent in Figure 5-3 corresponds to the contraction of the frequency axis by 25 percent in Figure 5-2.

Each pixel in an image has frequency information associated with it. The edges of the letter in Figure 5-3a have very high frequencies, while the white areas have only DC frequency content. As the image in Figure 5-3c is scaled, the edges still retain the high frequencies, only the position (relative to the top left corner of the image) is changed. Likewise, as the FDCT is computed over a block of image samples, the positional relationship of the samples to the DCT basis frequencies is encoded in the resulting 8x8 DCT coefficients. By changing the basis frequencies during the IDCT process to match a new sampling grid, the positional information is interpreted as matching that new grid also.

Spatial and Fourier Domain Reconstruction

In order to resample the sampled image, the original image, F(x,y), can be convolved with a reconstruction kernel, R(x,y). Using the convolution property [Jain 89], the transform of the convolved functions is equal to the product of the transforms of the functions. This relationship can be written as

$$F(x,y) \otimes R(x,y) \Leftrightarrow \mathbf{F}(\omega_1,\omega_2)\mathbf{R}(\omega_1,\omega_2) \tag{5-5}$$

Recalling Equation 2-2,

$$F_S(x,y) = F(x,y)S(x,y). \tag{5-6}$$

The sampled image, $F_S(x,y)$, is the product of the continuous image, F(x,y), and the sampling function S(x,y). The sampled image, $F_S(x,y)$, is termed the ideal image, $F_I(x,y)$, if the sampling function used exactly matches the desired sampling function needed to match a given display device. The initial sampling function used to digitized the image is normally assumed to be different than the sampling function desired for the displaying the image on the output device.

The convolution of the sampled image, $F_S(x,y)$, and the spatial sampled reconstruction kernel, $R_S(x,y)$ is the sampled reconstructed image, $F_R(r,s)$.

$$F_R(r,s) = F_s(x,y) \otimes R_s(x,y) \tag{5-7}$$

Using the Fourier transform on Equation 5-7, but ignoring the change in variables from x,y to r,s gives

$$\mathbf{F}_R(\omega_1,\omega_2) = \mathbf{F}_S(\omega_1,\omega_2)\mathbf{R}_S(\omega_1,\omega_2) \tag{5-8}$$

where $\mathbf{F}_S(\omega_1,\omega_2)$ is the transform of the sampled image, $F_S(x,y)$, defined in Equation 2-5 as

$$\mathbf{F}_S(\omega_1,\omega_2) = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\mathbf{F}(\omega_1 - j\omega_{1S}, \omega_2 - k\omega_{2S}), \tag{5-9}$$

and the Fourier domain sampling frequencies, ω_{1S}, and ω_{2S} are defined as

$$\omega_{1S} = \frac{2\pi}{\Delta x}, \text{ and } \omega_{2S} = \frac{2\pi}{\Delta y}. \tag{5-10}$$

$\mathbf{R}_S(\omega_1,\omega_2)$ is the transform of the sampled reconstruction kernel.

If the original sampled image, $F_S(x,y)$, is bandlimited and sufficiently sampled, the spectrum of the reconstructed image, $\mathbf{F}_R(\omega_1,\omega_2)$, can be made equal to the spectrum of the ideal image, $\mathbf{F}_I(\omega_1,\omega_2)$ by using an ideal reconstruction function, $\mathbf{R}_I(\omega_1,\omega_2)$. In this case the spatial versions, $F_S(x,y)$ and $F_I(x,y)$, of the images can be made equal. To show this, recall Equation 5-6. The sampled image $F_S(x,y)$ is the product of the continuous image and the sampling function S(x,y). Using Equation 2-1, the sampling function is the summation of dirac delta functions, located on a grid spaced Δx and Δy apart,

$$S(x,y) = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\delta(x - j\Delta x, y - k\Delta y). \tag{5-11}$$

Using Equation 5-11 in Equation 5-6, and moving the continuous image function inside the summations, yields

$$F_s(x,y) = \sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}F(j\Delta x, k\Delta y)\times\delta(x - j\Delta x, y - k\Delta y). \tag{5-12}$$

Taking the continuous two dimensional Fourier transform of Equation 5-12 above gives $\mathbf{F}_S(\omega_1,\omega_2)$, the Fourier transform of the sampled image [Pratt 91].

$$\mathbf{F}_S(\omega_1,\omega_2) = \iint_{\infty} F_S(x,y)e^{-i(\omega_1 x+\omega_2 y)}dxdy \tag{5-13}$$

The Fourier transform of the sample image, $\mathbf{F}_S(\omega_1,\omega_2)$, can be expressed as the convolution of the Fourier transform of the continuous image and the Fourier transform of the sampling function.

$$\mathbf{F}_S(\omega_1,\omega_2)=\frac{1}{4\pi^2}\mathbf{F}(\omega_1,\omega_2)\otimes\mathbf{S}(\omega_1,\omega_2) \tag{5-14}$$

The Fourier transform of the spatial sampling function, Equation 2-1, is an infinite array of Dirac delta functions.

$$\mathbf{S}(\omega_1,\omega_2)=\frac{4\pi^2}{\Delta x\Delta y}\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\delta(\omega_1-j\omega_{1S},\omega_2-k\omega_{2S}). \tag{5-15}$$

Performing the convolution in Equation 5-14 as a multiplication in the spatial domain gives

$$\mathbf{F}_S(\omega_1,\omega_2)= \frac{1}{\Delta x\Delta y}\iint_{\infty}\mathbf{F}(\omega_1-\alpha,\omega_2-\beta)\times\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\delta(\alpha-j\omega_{1S},\beta-k\omega_{2S})d\alpha d\beta. \tag{5-16}$$

Using the sifting property of the delta function and combining the integration and summation:

$$\mathbf{F}_S(\omega_1,\omega_2)=\frac{1}{\Delta x\Delta y}\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\mathbf{F}(\omega_1-j\omega_{1S},\omega_2-k\omega_{2S}). \tag{5-17}$$

Finally, Equation 5-17 can be compared to the reconstructed image spectrum

$$\mathbf{F}_R(\omega_1,\omega_2)=\frac{1}{\Delta x\Delta y}\mathbf{R}_S(\omega_1,\omega_2)\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\mathbf{F}(\omega_1-j\omega_{Ss},\omega_2-k\omega_{2S}). \tag{5-18}$$

which is Equation 5-8, with Equation 5-9 used to expand the image spectrum.

The only difference is the presence of the transform of the sampled reconstruction function, $\mathbf{R}_S(\omega_1,\omega_2)$. Since by definition a reconstruction function filters out the image samples j, k≠0, the two spectrums, $\mathbf{F}_S(\omega_1,\omega_2)$, and $\mathbf{F}_R(\omega_1,\omega_2)$ are equal, except for the weighting of the reconstruction function. Differences will exist in the form of aliasing problems if either image is insufficiently sampled. In the discrete transform domain, the image spectrum is replicated at the sampling frequency. These replications can overlap if the sampling increments, Δx or Δy, are too large, resulting in aliasing errors.

In order to prevent the replicated spectrums from overlapping, the spatial sampling increment is chosen so that the region bounded by the cutoff frequencies, ω_{1C}, and ω_{2C}, is inside the region bounded by one-half of the sampling frequency:

$$\omega_{1C} \le \frac{\omega_{1S}}{2}, \text{ and } \omega_{2C} \le \frac{\omega_{2S}}{2} \tag{5-19}$$

or

$$\Delta x \le \frac{\pi}{\omega_{1C}}, \text{ and } \Delta y \le \frac{\pi}{\omega_{2C}}. \tag{5-20}$$

For image acquisition systems, this means the sampling increments must not be larger than one-half of the smallest detail in the image. In the case of the reconstruction, the higher frequencies (greater than one half the sampling frequency) can be filtered out during the calculation of Equation 5-17, by limiting the range of the summation terms to only include one spectrum.

Ideal Reconstruction

Ideal reconstruction changes the original sample spacing to match the desired sample spacing for the final output device. There are two simple reconstruction filters to be examined as represented in Figure 5-4. The first is a rectangular area enclosing the cutoff frequencies, ω_{1C}, and ω_{2C}. This is the simplest extension of the one-dimensional case. The second filter is a circle enclosing the same cutoff frequencies. The spatial versions of these filters are sinc functions and first order Bessel functions, respectively.

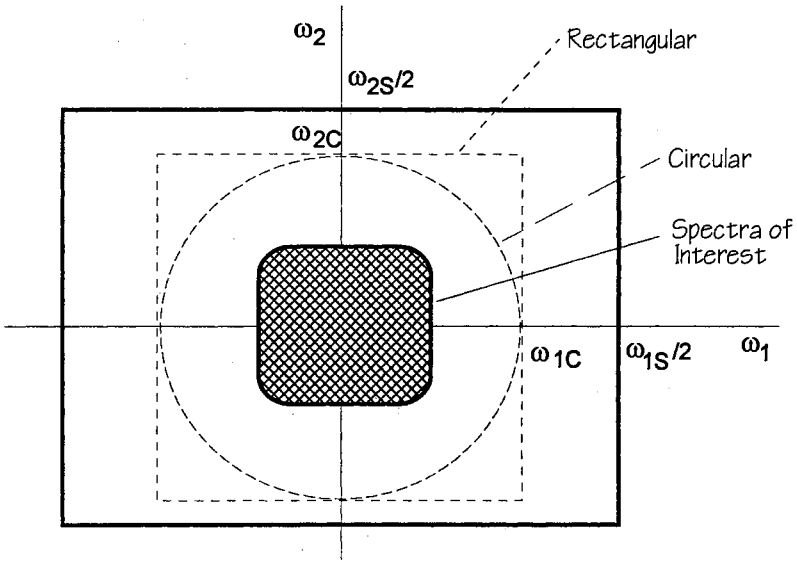


Figure 5-4. Reconstruction Filter Shapes

This research uses a circular reconstruction filter for the DCT frequency implementation in this chapter, because the offset from the DC coefficient in Table 2-1 is used to calculate the reconstruction function. However, the rectangular filter is examined in order to determine cutoff frequencies. Both the circular and rectangular filters have the same cutoff frequencies, but the rectangular filter is easier to describe mathematically and conceptualize.

The Fourier frequency response [Pratt 91] of the rectangular reconstruction function is

$$\begin{aligned} \mathbf{R}(\omega_1,\omega_2) &= K, \quad \text{for } |\omega_1| \le \omega_{1L}, \text{ and } |\omega_2| \le \omega_{2L} \\ \mathbf{R}(\omega_1,\omega_2) &= 0, \quad \text{otherwise} \end{aligned} \tag{5-21}$$

where ω_{1L} and ω_{2L} are between the cutoff frequencies and one half the sampling frequencies. The inverse Fourier transform of $\mathbf{R}(\omega_1,\omega_2)$ is

$$R(x,y) = \frac{K\omega_{1L}\omega_{2L}}{\pi^2} \frac{\sin(\omega_{1L}x)}{\omega_{1L}x} \frac{\sin(\omega_{2L}y)}{\omega_{2L}y}. \tag{5-22}$$

Equation 5-22 is not a practical filter to implement in the spatial domain because the sinc functions are infinite in extent. However, in the frequency domain, this function reduces to a scaling constant K.

Parametric Cubic Convolution

Used as an approximation to the ideal interpolation function in the spatial domain, the cubic kernel is a limited extent function design to match the sinc function at up to two points away from the center. It is described [Keys 81] as two functions, the first extending from zero to one, the second from one to two, reflected about the y axis. Equation 5-23 is the two dimensional form, with the offset from the central point weighting both axis equally.

$$R(x,y) = (\alpha+2)(x^2+y^2)^{\frac{3}{2}} - (\alpha+3)(x^2+y^2)+1,$$
$$\text{for } 0 \le \sqrt{x^2+y^2} \le 1$$

$$R(x,y) = \alpha(x^2+y^2)^{\frac{3}{2}} - 5\alpha(x^2+y^2)+8\alpha(x^2+y^2)^{\frac{1}{2}} - 4\alpha, \quad (5\text{-}23)$$
$$\text{for } 1 < \sqrt{x^2+y^2} \le 2$$

Taking the continuous Fourier transform of R(x,y) in Equation 5-23, and ignoring the limits of integration for now, these two functions are:

$$\mathbf{R}(\omega_1,\omega_2) = \iint_{\infty} [(\alpha+2)(x^2+y^2)^{\frac{3}{2}} - (\alpha+3)(x^2+y^2)+1]e^{-i(\omega_1 x+\omega_2 y)}dxdy,$$
$$\text{for } 0 \le \sqrt{x^2+y^2} \le 1$$

$$\mathbf{R}(\omega_1,\omega_2) = \iint_{\infty} [\alpha(x^2+y^2)^{\frac{3}{2}} - 5\alpha(x^2+y^2)+8\alpha(x^2+y^2)^{\frac{1}{2}} - 4\alpha]e^{-i(\omega_1 x+\omega_2 y)}dxdy, \quad (5\text{-}24)$$
$$\text{for } 1 < \sqrt{x^2+y^2} \le 2$$

Equation 5-24 is given as a reference for now, and is used later to compare the cosine frequency domain implementation of the parametric cubic convolution algorithm. The cosine frequency domain implementation of the parametric cubic convolution function is used in Chapter 6 during the reconstruction of JPEG compressed images.

Sharpened Gaussian

The sharpened gaussian function was developed is Chapter 4 and offers a balance between the computational problems inherent in a high order interpolation filter or spline,

and the pixelization problems of a low order function. Ratzel [Ratzel 80] compared this function to other common functions by visual methods but did not compare it to the cubic convolution or ideal reconstruction functions. The results of this comparison determined the best width of the spatial kernel compared to the width of the frequency kernel. Ratzel defined the spatial function as

$$R(x,y) = Ce^{-0.5(x^2+y^2)/\sigma^2} - Se^{-0.5((x^2+y^2)+1)/\sigma^2} - Se^{-0.5((x^2+y^2)-1)/\sigma^2} \tag{5-25}$$

where C and S are gains applied to the central and side lobes, respectively,

$$C = \frac{1.5}{2.5066\sigma}, \text{ and } S = \frac{0.27}{2.5066\sigma} \tag{5-26}$$

and

$$\sigma = 0.375. \tag{5-27}$$

As discussed in Chapter 4, the values of C and S are modified in this research by the central gain, 1.5, and the side gain, 0.27. The value of σ is also modified by the scaling factor, all in an effort to keep the filter width consistent with results in the literature.

The Fourier frequency domain counterpart for Equation 5-25 is

$$\mathbf{R}(\omega_1,\omega_2) = \iint_{\infty} [Ce^{-0.5(x^2+y^2)/\sigma^2} - Se^{-0.5((x^2+y^2)+1)/\sigma^2} - Se^{-0.5((x^2+y^2)-1)/\sigma^2}]e^{-i(\omega_1 x+\omega_2 y)}dxdy \tag{5-28}$$

using Equations 5-26 and 5-27 as definitions for the gains and σ. As with Equation 5-24, Equation 5-28 is provided as a reference for later cosine frequency domain implementations of the sharpened gaussian reconstruction function. The cosine domain versions of these reconstruction functions are used to reconstruct JPEG images.

Cosine Domain Reconstruction

Ideal Reconstruction

During the initial digitization of a continuous image, one sampling function is used to produce the image for compression and storage. During reconstruction, a different sampling function is often desired to produce a sampled image which matches the output device characteristics. No practical reconstruction function can exactly reproduce the original image data lost during the initial digitization, a reconstruction function can only interpolate between known image data samples to estimate the intermediate points. An ideal reconstruction function exactly reconstructs the original continuous image from the digitized version, and resamples this image to match the output device.

The ideal reconstruction function in the frequency domain is defined in Equation 5-21. It shows a scaling constant defined over a limited frequency range, and zero outside that range. This research is concerned with scaling images to different sizes. In order to define where to set cutoff frequencies, this section examines the ideal reconstruction case. The cutoff frequencies developed here will be applied to the cubic and sharpened gaussian reconstruction cases.

The first case to explore is magnification, the second is minification. The continuous image is assumed to be bandlimited and sufficiently sampled during digitization to prevent aliasing. However, as the image is reconstructed it is possible to reintroduce aliasing problems. Figure 5-5 illustrates a one-dimensional case of the

original sampled image spectra. The spectra is repeated each integer multiple of sampling frequency, ω_S.

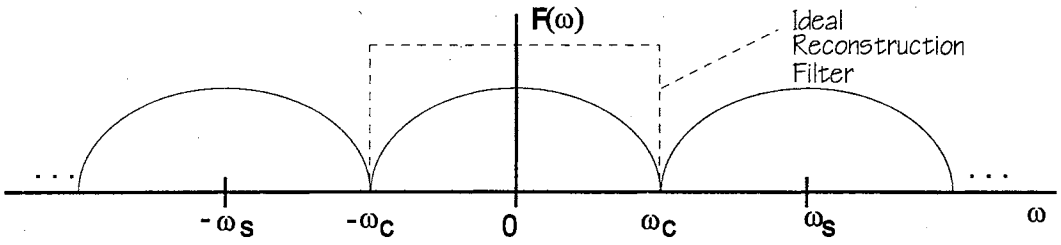


Figure 5-5. Original Spectra

During magnification, the new sampling rate is higher than the original sampling rate. Since the spectra is fixed (and bandlimited) at this point, the effect of the higher sampling rate is to spread out these repeated spectra. This is shown below in Figure 5-6.

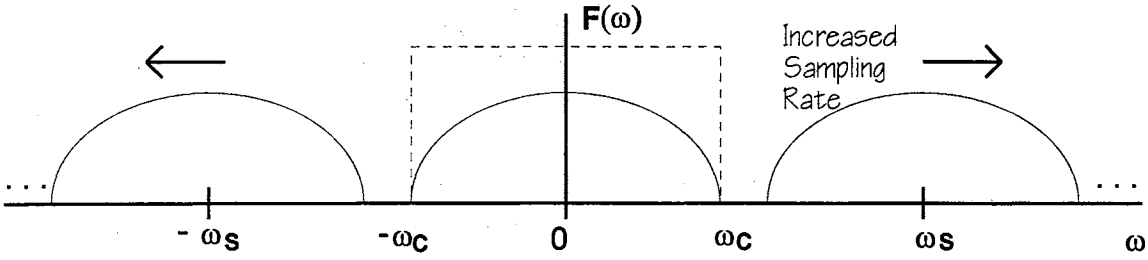


Figure 5-6. Increasing Sampling Rate

Clearly, in the case of magnification, no adjustment of the reconstruction filter cutoff frequencies is needed (at least the cutoff frequencies do not have to be restricted). In fact, the reconstruction filter can be relaxed to include higher frequencies, if they are available. Although this may seem to reduce aliasing, this is not an accepted method of anti-aliasing. If the image is not bandlimited, the individual spectra extend beyond ω_C. No matter how far the spectra are moved out by magnification, there will still be aliasing

problems. In fact, depending of the frequency content of the image, aliasing may become more apparent.

In the case of minification, the filter cutoff frequencies do need to be restricted. This is shown in Figure 5-7. As the sampling rate decreases, the repeated spectra move in toward the center, causing the spectra to overlap, resulting in aliasing.

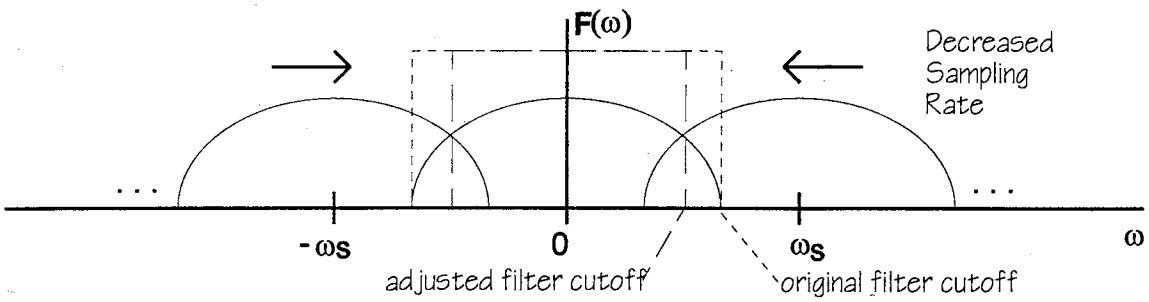


Figure 5-7. Decreasing Sampling Rate

By adjusting the cutoff frequency, ω_C, to one-half the new sampling frequency, ω_S, the aliasing problem is reduced (although not eliminated). If the new cutoff frequency is moved in further, aliasing can be further reduced, and eventually eliminated, at the cost of reducing the high frequencies inside the passband.

In order to find the new correct cutoff frequency, the new sampling frequency must be known. Recalling Equation 5-19 and 5-20,

$$\omega_{1C} \le \frac{\omega_{1S}}{2}, \text{ and } \omega_{2C} \le \frac{\omega_{2S}}{2} \tag{5-29}$$

and

$$\Delta x \le \frac{\pi}{\omega_{1C}}, \text{ and } \Delta y \le \frac{\pi}{\omega_{2C}}. \tag{5-30}$$

the new cutoff frequencies ω_{1C} and ω_{2C} are available from either the new sampling frequency, or the new sampling spacing.

These new cutoff frequencies are applied to the JPEG algorithm during reconstruction. During minification, the high frequency coefficients are set to zero, effectively filtering out the higher DCT frequencies. To get a better idea of the actual cutoff frequencies used during a JPEG reconstruction, it is useful to examine the JPEG algorithm more closely.

The algorithm starts with either 8 or 12 bits of precision (unsigned) per color component sample. These represent the magnitude of a particular chrominance component, or luminance field. These numbers are level shifted to be centered about zero, by subtracting 2^{P-1}, where P is the precision used. For 8 bits the shift is 128, for 12 bits the shift is 2048. These signed numbers (-128 to +127 and -2048 to +2047) are transformed into DCT coefficients using the FDCT algorithm in Equation 5-1, which may include a loss of precision. In order to achieve (lossy) compression, the DCT coefficients are then quantized using one of four tables. The two recommended tables are shown below (Table 5-1 and Table 5-2), one for the luminance channel, and one for the different chrominance components. Because the coefficients are rounded to the nearest integer after division, the quantizing tables reflect the relative importance (visually) of the frequencies for both luminance channels and chrominance channels. Using these tables as a heuristic guide, different cutoff frequencies can be set for the respective channels.

TABLE 5-1

JPEG LUMINANCE QUANTIZATION FACTORS

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

TABLE 5-2

JPEG CHROMINANCE QUANTIZATION FACTORS

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

The DCT coefficients for all channels are quantized (and later unquantized) using the above tables and Equations 2-20 and 2-21,

$$\mathcal{F}_q(u,v) = \text{round}\left(\frac{\mathcal{F}(u,v)}{Q(u,v)}\right), \text{ and } \mathcal{F}_{uq}(u,v) = \mathcal{F}_q(u,v) \times Q(u,v) \qquad (5\text{-}31)$$

where Q(u,v) is a quantization value from a table, $\mathcal{F}_q(u,v)$ is a quantized DCT coefficient, and $\mathcal{F}_{uq}(u,v)$ is the recovered (unquantized) DCT coefficient.

When performing the inverse operation, the data is first unquantized, then the shifted image values are generated using the IDCT in Equation 5-1, and the shifted values are level shifted back to their unsigned representation. This ignores the entropy coding, and other formatting operations, since this does not affect the data used in this research. Since the JPEG algorithm uses 8x8 DCT blocks, there is an obvious limitation on the high frequencies one block can represent. Each color component can also be sampled at a different rate. In order to locate which coefficients represent which frequencies, the respective contribution for each coefficient must be known. The coefficients are ordered as shown in Table 2-1. The higher transform frequencies of the JPEG algorithm are located in the lower right corner, and thus, are encountered last in the data stream. Using the FDCT portion of Equation 5-1,

$$\mathcal{F}_S(u,v) = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} F_s(x,y) \cos\left(\frac{u\pi}{16}(2x+1)\right) \cos\left(\frac{v\pi}{16}(2y+1)\right) \qquad (5\text{-}32)$$

and ignoring the input $F_S(x,y)$ for the moment, the transform frequency coefficient output, $\mathcal{F}_S(u,v)$, is the coefficient of a particular transform frequency set by the coordinates (u,v). The frequency referenced by (u,v) is not the Fourier frequency, but rather the cosine transform frequency. The difference is important when comparing these results to the Fourier transform (or any other transform). The high cosine frequencies in the lower right corner do not relate directly to the high frequencies in either the continuous Fourier, or the discrete Fourier. In fact, the high frequencies in one transform may include the low frequencies of another.

To find the actual frequency corresponding to a particular coefficient, the original sampling frequency must be known in order to relate the units. It is not necessary to assign units to the frequency term, all that is needed is a relative measure, using the value of one as a normalized, unscaled case. Since the frequency is related to the number of zero crossings for a particular transform (a generalization for the Fourier), the transform frequency is related in the same manner to the row and column indices of the JPEG algorithm. As the row (or column) indices increase, the number of zero crossings in that dimension increase linearly. Thus the product of the indices, divided by the entire space, is an accurate relative frequency term. Thus, the cutoff frequencies

$$u_C = \frac{u}{N} u_S, \text{ and } v_C = \frac{v}{N} v_S \tag{5-33}$$

are relative to the original sampling frequency, u_S and v_S. The original sampling frequencies, u_S and v_S, are assumed to have a value of one for simplicity in this research.

The new cutoff frequency can also be calculated directly from ζ, the scaling factor. Since the cutoff frequency is one-half the sampling frequency, the new cutoff frequency is the product of the scaling factor and the Nyquist frequency. Using Equation 5-29,

$$u_C \leq \zeta \times \frac{u_S}{2}, \text{ and } v_C \leq \zeta \times \frac{v_S}{2}. \tag{5-34}$$

For example, if the original sampling frequency is one, the Nyquist frequency is one-half. Is $\zeta \geq 1$, there is no change needed, assuming the original was sufficiently sampled and bandlimited. If $\zeta < 1$, the cutoff frequencies are reduced by that amount. Using a circular filter centered on the DC coefficient with ζ=0.25, only three (0, 1, and 2) low frequency

coefficients are non-zero in Table 2-3. Depending on the extent of aliasing allowable, higher frequencies (4, 3, and 5) can still be included.

Thus, as in the Fourier case, Equation 5-21, the cosine domain ideal reconstruction filter is a simple scaling constant defined over the passband, and zero elsewhere.

$$\begin{aligned} \mathcal{R}(u,v) &= K, \quad \text{for } |u| \le u_L, \text{ and } |v| \le v_L \\ \mathcal{R}(u,v) &= 0, \quad \text{otherwise} \end{aligned} \tag{5-35}$$

Parametric Cubic Convolution

Using Equation 5-23, the spatial definition for the cubic kernel, and Equation 5-1, the definition of the JPEG DCT, the corresponding cosine reconstruction filters are:

$$\begin{aligned} &\mathcal{R}(u,v) = \frac{1}{4} C_u C_v \times \\ &\quad \sum_{x=0}^{7}\sum_{y=0}^{7} [(\alpha+2)(x^2+y^2)^{\frac{3}{2}} - (\alpha+3)(x^2+y^2) + 1] \times \\ &\quad \cos\left(\frac{\pi u}{16}(2x+1)\right)\cos\left(\frac{\pi v}{16}(2y+1)\right) \\ &\text{for } 0 \le \sqrt{x^2+y^2} \le 1 \\ &\\ &\mathcal{R}(u,v) = \frac{1}{4} C_u C_v \times \\ &\quad \sum_{x=0}^{7}\sum_{y=0}^{7} [\alpha(x^2+y^2)^{\frac{3}{2}} - 5\alpha(x^2+y^2) + 8\alpha(x^2+y^2)^{\frac{1}{2}} - 4\alpha] \times \\ &\quad \cos\left(\frac{\pi u}{16}(2x+1)\right)\cos\left(\frac{\pi v}{16}(2y+1)\right) \\ &\text{for } 1 < \sqrt{x^2+y^2} \le 2 \end{aligned} \tag{5-36}$$

This is the cubic convolution reconstruction kernel used in the reconstruction of the JPEG image data. Equation 5-36 does not include the scaling and resampling portion of the

algorithm. This 8x8 array of DCT coefficients is used to multiply the image DCT coefficients, resulting in a cubic filtered 8x8 array of image coefficients.

Sharpened Gaussian

Using Equation 5-25, the spatial definition of the sharpened gaussian reconstruction function, and the definition of the JPEG DCT, the cosine domain reconstruction function is defined as:

$$\mathcal{R}(u,v) = \frac{1}{4} C_u C_v \times \sum_{x=0}^{7}\sum_{y=0}^{7} [Ce^{-0.5(x^2+y^2)/\sigma^2} - Se^{-0.5((x^2+y^2)+1)/\sigma^2} - Se^{-0.5((x^2+y^2)-1)/\sigma^2}] \times \cos\left(\frac{\pi u}{16}(2x+1)\right)\cos\left(\frac{\pi v}{16}(2y+1)\right) \quad (5\text{-}37)$$

The 8x8 DCT coefficient array produced from Equation 5-37 is used to filter the image data with the sharpened gaussian kernel.

Error Analysis

Ideal reconstruction in the spatial domain requires an infinite-order interpolation between the sample points. In the frequency domain, ideal reconstruction can be achieved if the image is bandlimited, sufficiently sampled, and a proper reconstruction filter is used. For image reconstruction systems applied to existing images, the continuous image has already been sampled at a predetermined increment. Often the original sampling increment is unknown and all that is available are the actual image samples. If the image is not sufficiently sampled for the limits of the passband, aliasing will result, and contribute to the total error. The amount of initial aliasing depends on the image, the original sampling increment, and the cutoff frequency used. This initial

aliasing is ignored in this research because the image is assumed to be correctly sampled when it is digitized.

For purposes of error analysis, the ideal image is used as a basis to measure against. The total error resulting from a non-ideal reconstruction filter is the difference between the ideally reconstructed image and the non-ideally reconstructed image. Relating the total error measured to the non-ideal filter, the total error measured has two components. The non-ideal passband response, E_p, resulting in a modification of the low and mid frequencies, and the non-ideal stopband response, E_S, resulting in aliasing or excessive high frequency reduction. These two components are shown in Figure 5-8.

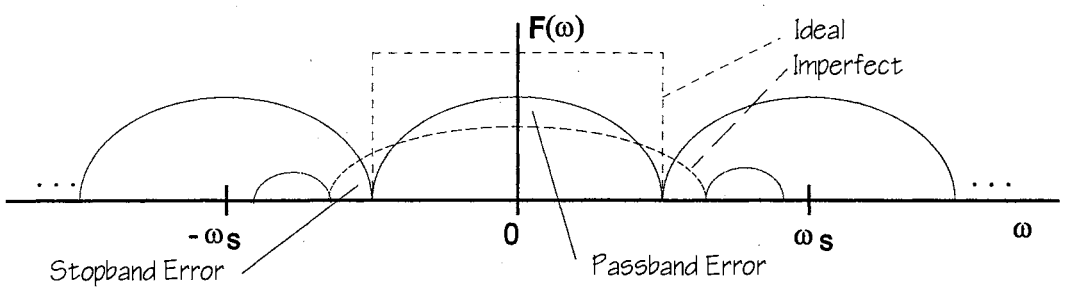


Figure 5-8. Components of the Total Error

The passband error affects frequencies less than the cutoff frequency, while the stopband error affects frequencies greater than the new cutoff frequency.

Using energy as an error estimation value, the energy of the ideally reconstructed image, is

$$\text{energy} = \sum_{\omega_1=-\infty}^{\infty} \sum_{\omega_2=-\infty}^{\infty} \left|\mathbf{F}_I(\omega_1,\omega_2)\right|^2 . \tag{5-38}$$

Since the ideal reconstruction filter stops at the Nyquist frequencies, the summation terms in Equation 5-38 can be limited to those frequencies,

$$\text{energy} = \sum_{\omega_1=-\frac{\omega_{1S}}{2}}^{\frac{\omega_{1S}}{2}} \sum_{\omega_2=-\frac{\omega_{2S}}{2}}^{\frac{\omega_{2S}}{2}} \left|\mathbf{F}_I(\omega_1,\omega_2)\right|^2 . \tag{5-39}$$

The passband error, E_P,of the non-ideal reconstruction filter is also limited to the new cutoff frequencies. The passband error can be written as the difference in the energy of the ideal, $\mathbf{F}_I(\omega_1,\omega_2)$, and non-ideal, $\mathbf{F}_R(\omega_1,\omega_2)$, reconstructed image spectra,

$$E_p = \sum_{\omega_1=-\frac{\omega_{1S}}{2}}^{\frac{\omega_{1S}}{2}} \sum_{\omega_2=-\frac{\omega_{2S}}{2}}^{\frac{\omega_{2S}}{2}} \left|\mathbf{F}_I(\omega_1,\omega_2)\right|^2 - \sum_{u=-\frac{\omega_{1S}}{2}}^{\frac{\omega_{1S}}{2}} \sum_{v=-\frac{\omega_{2S}}{2}}^{\frac{\omega_{2S}}{2}} \left|\mathbf{F}_R(\omega_1,\omega_2)\right|^2 . \tag{5-40}$$

The stopband error, E_S, is the energy in the non-ideally reconstructed image spectra, $\mathbf{F}_R(\omega_1,\omega_2)$, above the cutoff frequency ω_C,

$$E_s = \sum_{\omega_1=-\infty}^{\frac{-\omega_{1S}}{2}} \sum_{\omega_2=-\infty}^{\frac{-\omega_{2S}}{2}} \left|\mathbf{F}_R(\omega_1,\omega_2)\right|^2 + \sum_{\omega_1=\frac{\omega_{1S}}{2}}^{\infty} \sum_{\omega_1=\frac{\omega_{1S}}{2}}^{\infty} \left|\mathbf{F}_R(\omega_1,\omega_2)\right|^2 \tag{5-41}$$

Both the passband error, E_P, and the stopband error, E_S, are usually normalized by dividing by the total energy in the ideally reconstructed image as defined in Equation 5-39.

During ideal reconstruction only the passband aliasing is present. During non-ideal reconstruction, both passband and stopband aliasing are present. One method of reducing aliasing is to reduce the cutoff frequency below one-half the sampling frequency. However, as the cutoff frequency is reduce, more high frequency information is lost. The tradeoff is to allow some aliasing, and retain more of the high frequencies, or to eliminate the passband aliasing by reducing the cutoff frequency, and thus, more high frequencies.

The passband error is due both to passband aliasing (if present), and attenuation or amplification due to the non-ideal gain of the reconstruction filter. When considering the case of JPEG, there are other sources of error that should be included. JPEG is a lossy compression scheme, due to round-off errors in both the forward and inverse DCT, and quantization errors. Only the inverse DCT round-off error is part of the reconstruction stage. The inverse quantization step does not introduce any new errors. The quantization error is assumed to be entirely contained in the forward process. The error introduced in the cosine terms is more difficult to determine for all implementations. In this research, the cosine terms are computed with an accuracy of 19 digits. Because of this, the error resulting from inaccurate calculations of the cosine terms is negligible.

Although not part of the reconstruction process, the forward cosine calculations are also accurate enough not to be a factor. The calculation errors then can be assumed to lie entirely in the forward quantization step. Recalling Equation 5-31, the forward quantization of

$$\mathcal{F}_q(u,v) = \text{round}\left(\frac{\mathcal{F}(u,v)}{Q(u,v)}\right) \tag{5-42}$$

reduces the accuracy of the coefficient to two or three (in the higher frequency - luminance case) decimal places. Considering the DC case which is quantized into 16 steps and has a range from -16384 to +16384, the step size is about 2048. Therefore the maximum error will be half the step size, or 1024. Normalized, this is just over three percent. This is a useful maximum error due to quantization using the default tables, Table 5-1 and 5-2. These default tables were derived in this manner [JPEG Draft 92], and if the values are divided by two, the results are designed to be indistinguishable from the original source image.

CHAPTER 6

RESULTS

Background

The purpose of this research was to investigate the feasibility of using the discrete cosine transform (DCT) image data directly while performing basic operations on the image. Of particular interest was interpolation and the subsequent reconstruction of the scaled image using two of the proven interpolation filters, the cubic filter, and the sharpened gaussian filter. In order to investigate the properties of these two filters, a baseline image of the correct size was created with an ideal filter to measure errors against.

In the spatial implementation of this research, the JPEG DCT coefficients representing the image were uncompressed with the standard JPEG IDCT algorithm, described in Chapter 2. This decompression results in the original spatial sampled image, $F_S(x,y)$. The traditional approach to reconstruction spatially convolves the original spatial sampled image with a sampled reconstruction kernel, $R_S(x,y)$, such as the cubic or sharpened gaussian. The result is the reconstructed spatial image, $F_R(r,s)$. The spatial sampled reconstruction kernel, $R_S(x,y)$, is defined by Equation 4-10 for the cubic kernel, or Equations 4-13 and 4-14 for the sharpened gaussian kernel.

This research introduces a new frequency implementation of the reconstruction process, beginning with the JPEG compressed image. The frequency image is composed of several 8x8 DCT coefficient blocks, termed $\mathcal{F}(u,v)$. Each coefficient block of the image is multiplied with the reconstruction filter, $\mathcal{R}(u,v)$. This multiplication is the

equivalent of convolution in the spatial domain, and performs either the cubic or sharpened gaussian filtering. The cosine domain reconstruction filter $\mathcal{R}(u,v)$ is obtained by first computing the spatial reconstruction filter, R(x,y), and second computing the cosine reconstruction filter, $\mathcal{R}(u,v)$. The spatial cubic reconstruction filter is defined by Equation 5-23, and the spatial sharpened gaussian reconstruction filter is defined by Equation 5-25. Using the FDCT process in Equation 5-1, the cosine reconstruction filter, $\mathcal{R}(u,v)$ is computed. The IDCT process in Equation 5-3 is used to scale and resample the image.

Reconstruction

As indicated earlier, the JPEG DCT uses a value of 8 for the range or block size N. The scaled block size is M = round(Nζ), where ζ is the scaling factor. Recalling Equation 5-1, the JPEG draft gives the DCT definition as:

$$\text{FDCT: } \mathcal{F}_s(u,v) = \frac{1}{4} C_u C_v \sum_{x=0}^{7}\sum_{y=0}^{7} F_s(x,y)\cos\left(\frac{u\pi}{16}(2x+1)\right)\cos\left(\frac{v\pi}{16}(2y+1)\right)$$

$$\text{IDCT: } F_s(x,y) = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C_u C_v \mathcal{F}_s(u,v)\cos\left(\frac{u\pi}{16}(2x+1)\right)\cos\left(\frac{v\pi}{16}(2y+1)\right) \quad (6\text{-}1)$$

$$\text{where } C_u, C_v = \frac{1}{\sqrt{2}} \text{ for u or v} = 0; \quad C_u C_v = 1 \text{ otherwise.}$$

The image space (x,y) is replaced with the reconstruction space (r,s), giving a slightly more general definition of the DCT process, defined in Equation 6-2.

$$\text{FDCT: } \mathcal{F}_s(u,v) = \frac{1}{4} C_u C_v \sum_{x=0}^{7}\sum_{y=0}^{7} F_s(x,y)\cos\left(\frac{u\pi}{16}(2x+1)\right)\cos\left(\frac{v\pi}{16}(2y+1)\right)$$

$$\text{IDCT: } F_s(r,s) = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C_u C_v \mathcal{F}_s(u,v)\cos\left(\frac{u\pi}{16}(2r+1)\right)\cos\left(\frac{v\pi}{16}(2s+1)\right) \quad (6\text{-}2)$$

$$\text{where } C_u, C_v = \frac{1}{\sqrt{2}} \text{ for u or v} = 0; \quad C_u C_v = 1 \text{ otherwise.}$$

Equation 6-2 is used as the starting point for the new DCT process developed in this research, before the scaling and filtering operations are added.

Ideal Reconstruction

The ideal reconstruction filter for the IDCT process was developed to provide a baseline image to measure errors against, and to help establish the cutoff frequencies needed during reconstruction. Scaling is accomplished by adjusting the frequency axis in the opposite direction as the spatial axis with the scaling factor ζ. This is defined in Equations 5-3 and 6-3, where $F_S(r,s)$ is the reconstructed image.

$$F_S(r,s) = \frac{1}{4\zeta^2}\sum_{u=0}^{7}\sum_{v=0}^{7} C_u C_v \mathcal{F}_S(u,v)\cos\left(\frac{u\pi}{16\zeta}(2r+1)\right)\cos\left(\frac{v\pi}{16\zeta}(2s+1)\right) \quad (6\text{-}3)$$

Cosine Domain Reconstruction

The two spatial reconstruction filters, cubic and sharpened gaussian are applied to the JPEG DCT data in the following manner. First, the circular spatial filter, R(x,y), is calculated, then the FDCT is applied to the filter, resulting in the cosine domain version $\mathcal{R}(u,v)$. $\mathcal{R}(u,v)$ is multiplied with each JPEG DCT coefficient block, and the scaled-IDCT process is used to reconstruct the sampled image, $F_R(r,s)$.

The cosine domain filter is defined as

$$\mathcal{R}(u,v)=\frac{1}{4}C_u C_v \sum_{x=0}^{7}\sum_{y=0}^{7} R(x,y)\cos\left(\frac{u\pi}{16}(2x+1)\right)\cos\left(\frac{v\pi}{16}(2y+1)\right). \tag{6-4}$$

where R(x,y) is the spatial filter, either the cubic, or the sharpened gaussian. The spatial version of the reconstructed and scaled image, $F_R(r,s)$, is

$$F_R(r,s)=\frac{1}{4\zeta^2}\sum_{u=0}^{7}\sum_{v=0}^{7} C_u C_v \mathcal{F}_{\mathcal{R}}(u,v)\cos\left(\frac{u\pi}{16\zeta}(2r+1)\right)\cos\left(\frac{v\pi}{16\zeta}(2s+1)\right) \tag{6-5}$$

where $\mathcal{F}_R(u,v)$, the reconstructed DCT coefficient block, is defined as

$$\mathcal{F}_{\mathcal{R}}(u,v)=\mathcal{F}_S(u,v)\mathcal{R}(u,v). \tag{6-6}$$

$\mathcal{F}_S(u,v)$ is the FDCT of the original sampled image given by Equation 6-2.

Parametric Cubic Convolution

The cubic interpolation filter is developed in Chapter 4 and Chapter 5. The spatial version of the cubic reconstruction filter, R(x,y), is defined in Equation 5-23. Using Equation 5-23 and Equation 6-4, $\mathcal{R}(u,v)$ can be calculated. This cosine frequency cubic filter, $\mathcal{R}(u,v)$, in Equation 6-4 is shown in Figure 6-1.

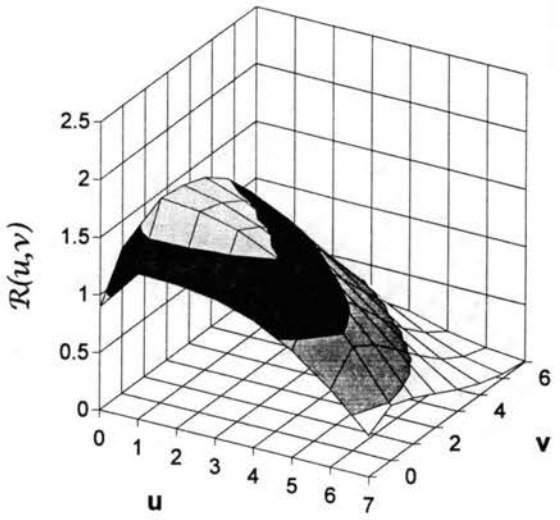


Figure 6-1a. Three dimensional view of the DCT Cubic Filter

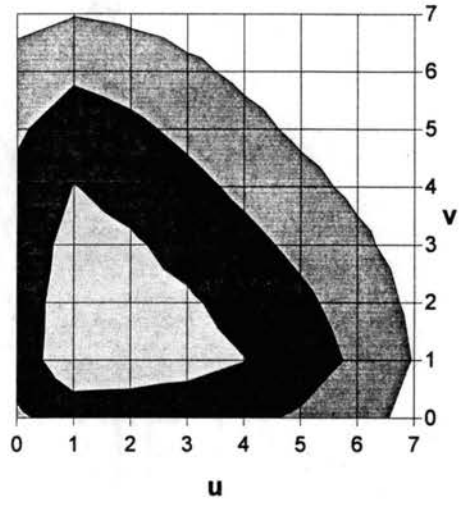


Figure 6-1b. Top view of the DCT Cubic Filter

From Figure 6-1, it is easy to see that the cubic filter, $\mathcal{R}$(u,v), attenuates the higher frequencies of a JPEG DCT block as expected, and thus can be described as a low pass filter.

Sharpened Gaussian

The spatial version of the sharpened gaussian function, R(x,y), is developed in Chapter 4. Chapter 5 includes development of the frequency version of this filter. Equation 5-25 gives the definition of R(x,y).

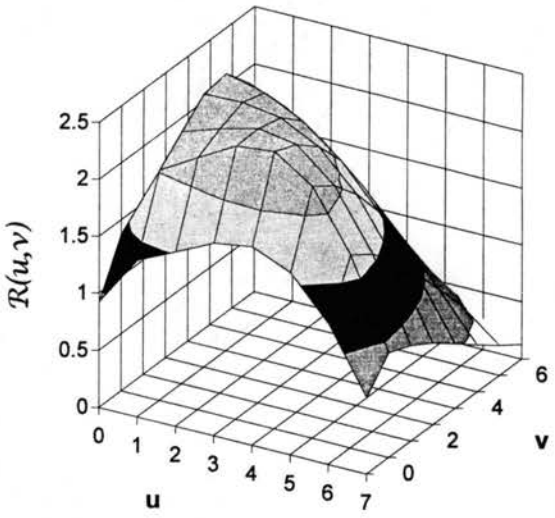


Figure 6-2a. Three dimensional view of the DCT Sharpened Gaussian Filter

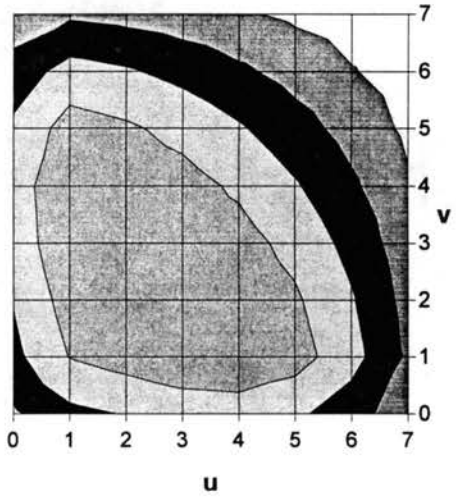


Figure 6-2b. Top view of the DCT Sharpened Gaussian Filter

Using Equation 5-25 with Equation 6-4, the cosine frequency version, $\mathcal{R}(u,v)$, of the sharpened gaussian filter can be calculated. The sharpened gaussian filter $\mathcal{R}(u,v)$, is shown in Figure 6-2. As in the case of the cubic function, the sharpened gaussian filter is a one pass filter, based on the radial distance from the interpolated point. The sharpened gaussian is slightly different than the cubic filter shown in Figure 6-1, but is also essentially a low-to-mid pass filter.

Implementation Differences

In this section, the differences between the spatial implementation and the cosine domain implementation are examined. In the figures below, a simple image represented by a pulse function (Figure 6-3) is filtered and reconstructed with a scaling factor of one. The cubic and the sharpened gaussian functions in both the DCT domain and in the spatial domain are applied and the percent error is evaluated. The spatial version of the cubic function has no errors, since that is how the cubic function is defined for a scaling value of one.

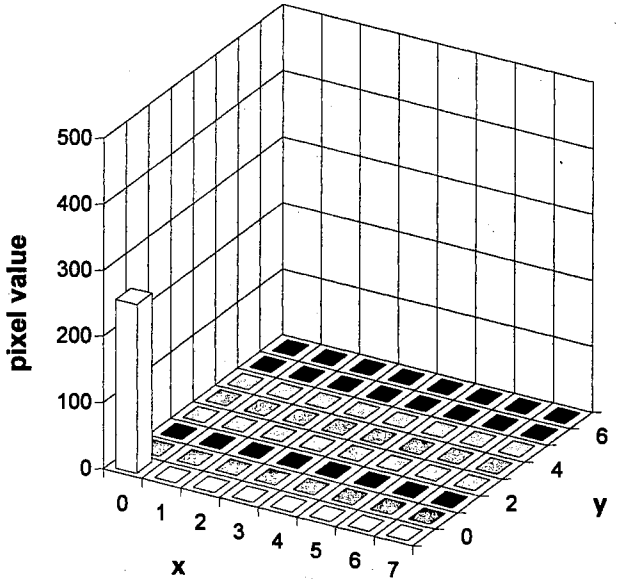


Figure 6-3.
The Pulse Function as the Original Image

Figure 6-4 shows the pulse function after reconstruction (ζ=1) with the new DCT-based method. Notice that the pixel value of the reconstructed pulse function extends past the original value (255). If this reconstruction process were part of an imaging application, the pixel value would be subjected to clipping or normalization. In either case, the resulting percent error would not be as severe. In the test images processed in this research, the clipping method produced a more visually pleasing image, because of the sharpening effect of the spurious noise, and the overall brightening of the image. Test images processed with normalization reduced both the percent error at the pulse location, and the spurious noise introduced by reconstruction, as well as reducing the apparent brightness of the image.

The pulse function reconstructed with the traditional spatial method did not have any measurable errors. The cubic spatial function exactly reconstructs the original data when the scaling factor is one, because the new interpolated points coincide with the original sample points.

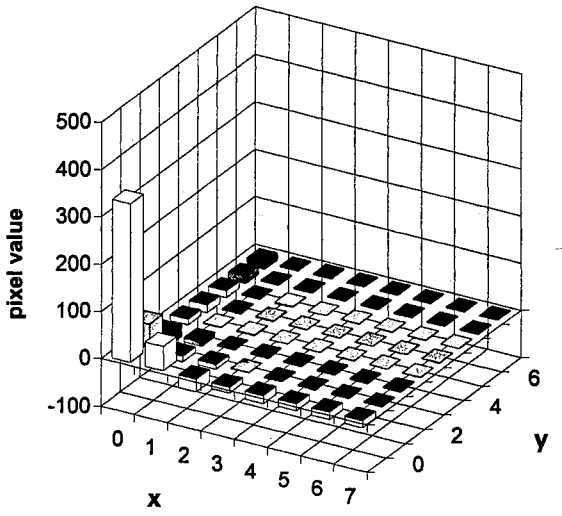


Figure 6-4a. The Pulse
Function after DCT Cubic Filtering

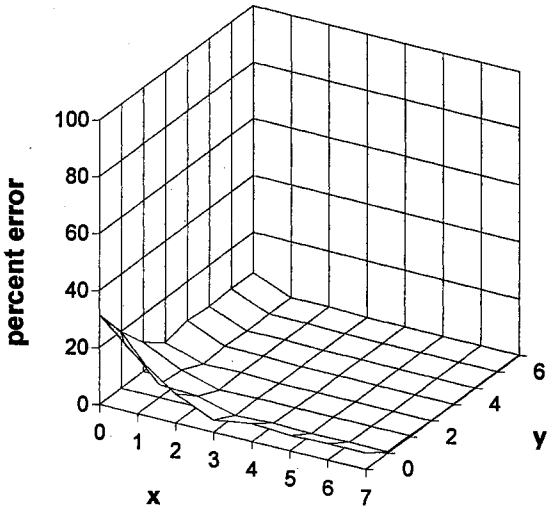


Figure 6-4b. Pulse Function
Percent Error for DCT Cubic Filtering

In Figure 6-5a, the reconstructed pulse function is shown, this time after processing with the DCT-based sharpened gaussian function. When compared to Figure

6-4a, the sharpened gaussian function obviously introduces more distortion. However, the purpose of the sharpened gaussian function is to produce a pleasing visual result, not to produce a numerically correct result. Figure 6-5b shows the percent error for the reconstructed pulse, before clipping or normalization. The maximum percent error, near 100 percent, is reduced after limiting the range of the pixel value. As in the case of the cubic function, clipping the values produced a visually pleasing image with the Lena and mandrill test images. However, the images that were processed with normalization after the sharpened gaussian function were subjectively better than the normalized images reconstructed with the cubic function. By clipping the values in the sharpened gaussian image, the spurious noise is not sufficiently reduced, and begins to interfere with the image.

Figure 6-6a shows the pulse function after sharpened gaussian reconstruction in the spatial domain using the traditional approach. The pulse value in Figure 6-6a has a greater percent error than the pulse value in the DCT case, but the noise introduced is reduced to the sharpening range of the function. Subjectively, there was little apparent difference in the quality of the two sharpened images. These subjective evaluations should only be used a heuristic guide for later research, there was no attempt to employ outside observers, and the evaluation was not done under ideal conditions.

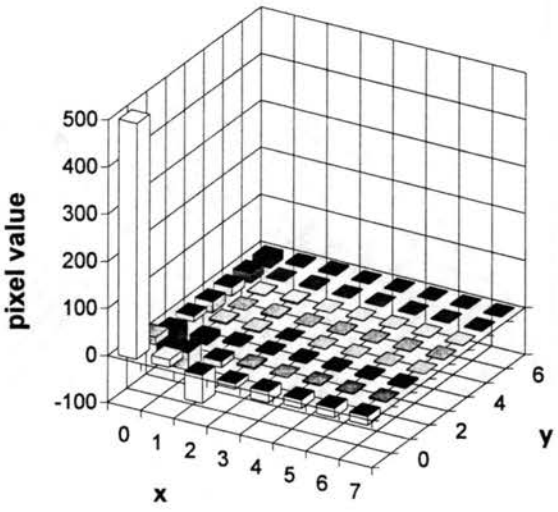


Figure 6-5a. The Pulse Function
after DCT Sharpened Gaussian Filtering

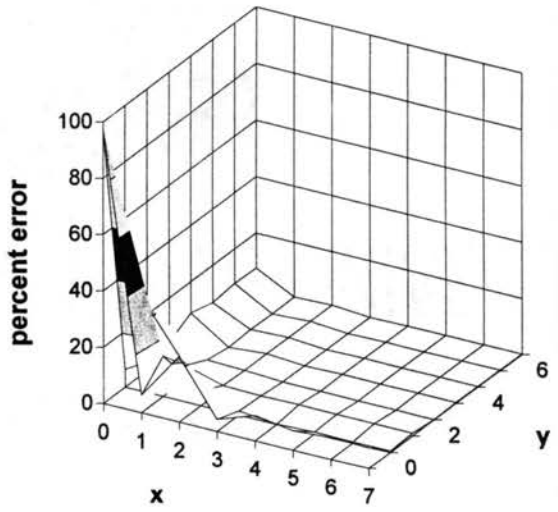


Figure 6-5b. Pulse Function Percent
Error for DCT Sharpened Gaussian Filtering

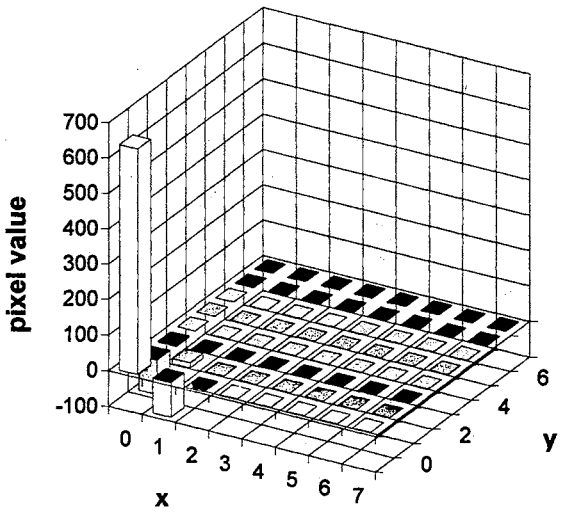


Figure 6-6a. The Pulse Function
after Spatial Sharpened Gaussian Filtering

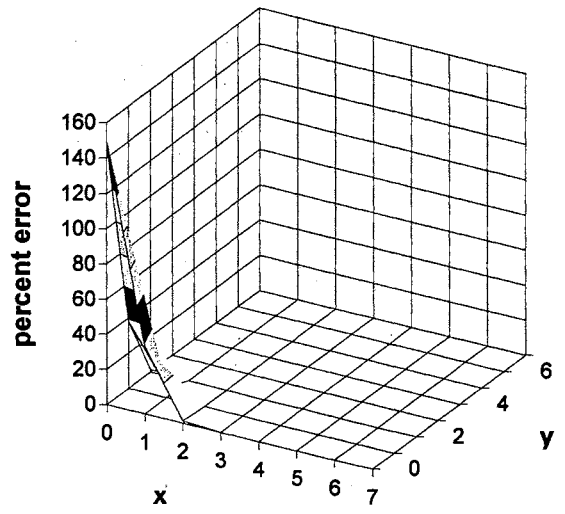


Figure 6-6b. Pulse Function Percent
Error for Spatial Sharpened Gaussian Filtering

As mentioned previously, both reconstruction filters can be described as low-mid pass filters that attenuate the high frequencies. It is useful to explore how this attenuation results in errors. The effect of this truncation, and the overall high frequency attenuation can be seen in the next set of figures where $\zeta=1$. These figures also serve to help explore the high frequency truncation introduced by minification. During minification, $\zeta<1$, the cutoff frequency is reduced to avoid aliasing problems.

Figure 6-7a shows the DCT transform coefficients produced by the pulse depicted in Figure 6-3. After the 8x8 block of DCT coefficients are calculated, the high frequencies are set to zero, simulating the low-pass filter, and the reduced cutoff frequency used during minification. The radial distance is used in these figures as the cutoff frequency. The second figure, Figure 6-7b, shows the resulting percent error, measured against the original pulse in Figure 6-3. Figure 6-7b clearly shows the ringing in the spatial domain as a result of truncating in the frequency domain. For this first cutoff frequency, the resulting maximum error is approximately one percent, which is below the JPEG three percent threshold before errors are considered noticeable.

Figures 6-8 through 6-11 follow the same description. The cutoff frequency is reduced by steps to a radial distance of 1, in Figure 6-11. As the cutoff frequency is reduced in the remaining figures, the maximum error grows quickly, and is centered at the original pulse location. The rest of the block shows an increasing percent error, but it is relatively small compared to the maximum. Since the high frequencies in the pulse test image are located at the origin, that is the location of the resulting errors as the high frequencies are truncated. This is to be expected since by truncating the high frequencies, the resulting filter smoothes the data over the entire block. By truncating the high frequencies during minification, the resulting image is smoothed in this manner. Thus the effect of the sharpening filters is reduced during minification, and may not be worth the computation. The minification-sharpening tradeoff is not pursued in this research, but would prove interesting for future study.

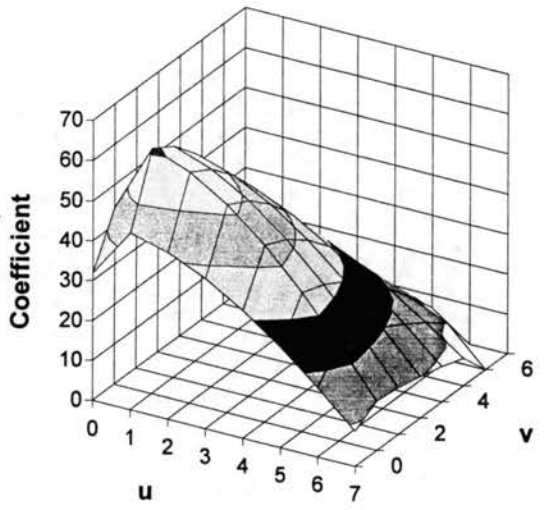


Figure 6-7a. DCT of Pulse,
Truncated Above Radial Distance 9

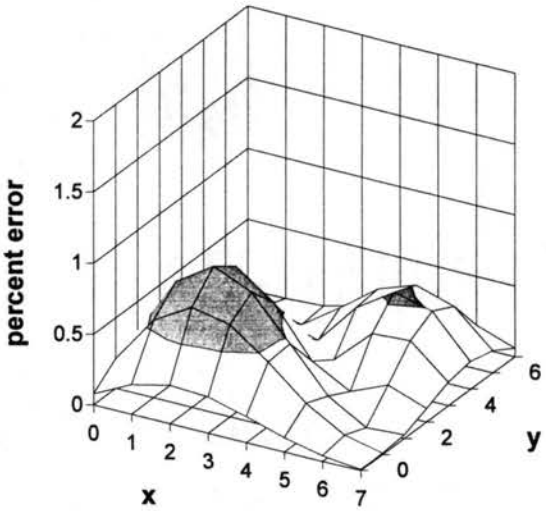


Figure 6-7b. Percent Error for
Truncation Above Radial Distance 9

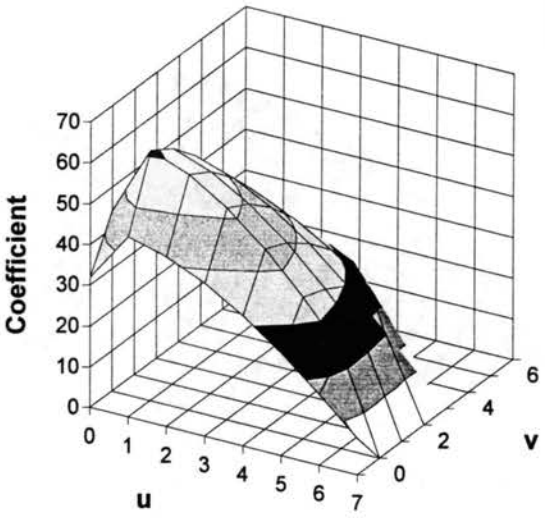


Figure 6-8a. DCT of Pulse,
Truncated Above Radial Distance 7

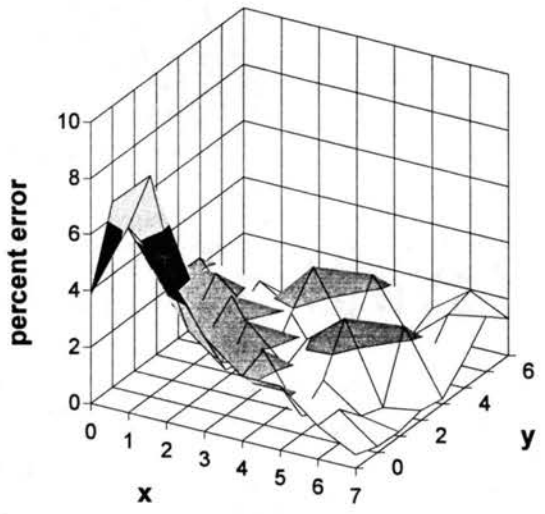


Figure 6-8b. Percent Error for
Truncation Above Radial Distance 7

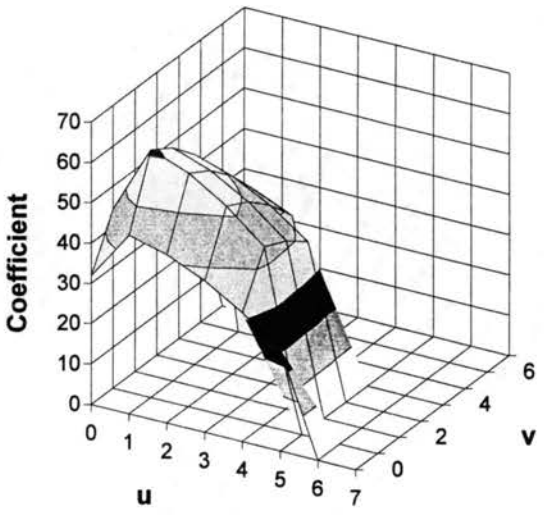


Figure 6-9a. DCT of Pulse,
Truncated Above Radial Distance 5

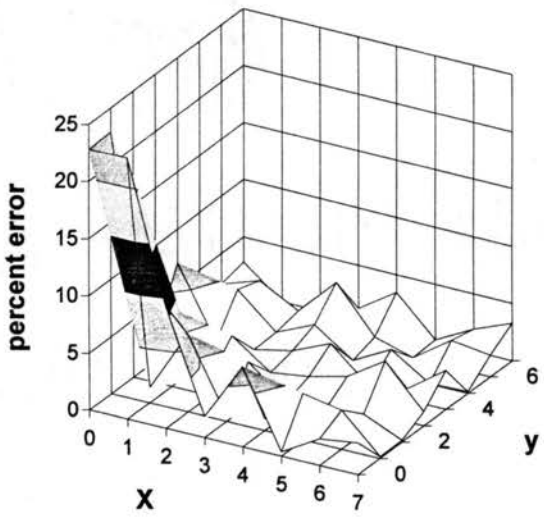


Figure 6-9b. Percent Error for
Truncation Above Radial Distance 5

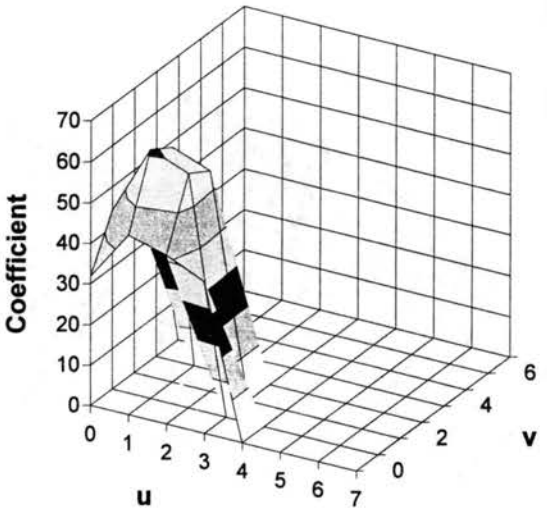


Figure 6-10a. DCT of Pulse,
Truncated Above Radial Distance 3

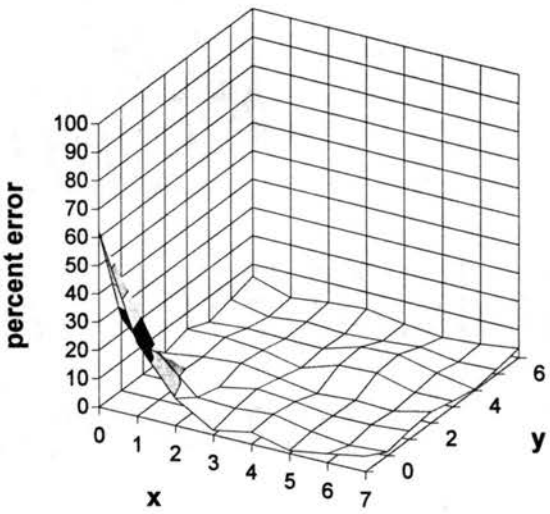


Figure 6-10b.Percent Error for
Truncation Above Radial Distance 3

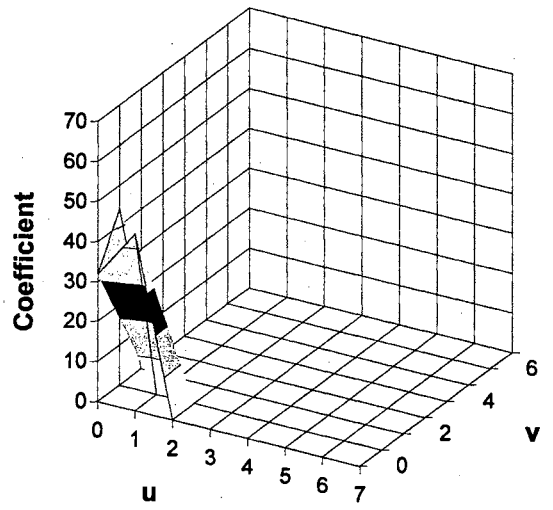


Figure 6-11a. DCT of Pulse,
Truncated Above Radial Distance 1

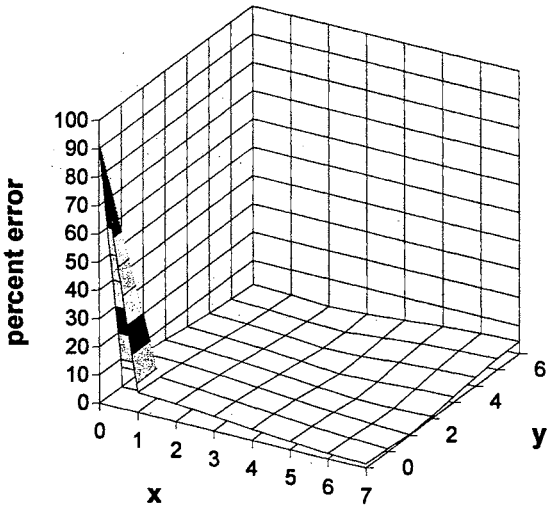


Figure 6-11b. Percent Error for
Truncation Above Radial Distance 1

Rectangular and Circular Filters

During reconstruction, a convolution is performed with the reconstruction kernel (filter) and the image data. The traditional spatial method performs this convolution in the spatial domain with a rectangular kernel. In the new method, the convolution is performed as a multiplication in the DCT domain with a circular filter. This difference may seem to be trivial, but on closer examination these two similar cases turn into two different filters. In Figure 6-12, the two cases are shown in the spatial domain. First, the rectangular case receives a positive (smoothing) weight from the inner rectangle, and a negative (sharpening) weight from the area between the two rectangles. Outside the outer rectangle, the weights are zero. Likewise, the circular filter receives a positive smoothing weight inside the inner circle, and a negative sharpening weight between the two circles.

As Figure 6-12 shows, the rectangular filter is influenced by 16 points, while the circular is influenced by at most 12 points. Because of this, the rectangular cubic filter will be more of a sharpening filter and less of a smoothing filter when compared to its circular counterpart. However, before that can be assumed for all cases, the actual weighting of each point must be considered. Note that if the radial distance for the circular filter is increased slightly, the four corner points left out previously will be included, and the circular filter will be influenced by all 16 samples, although at a different weighting than the rectangular case.

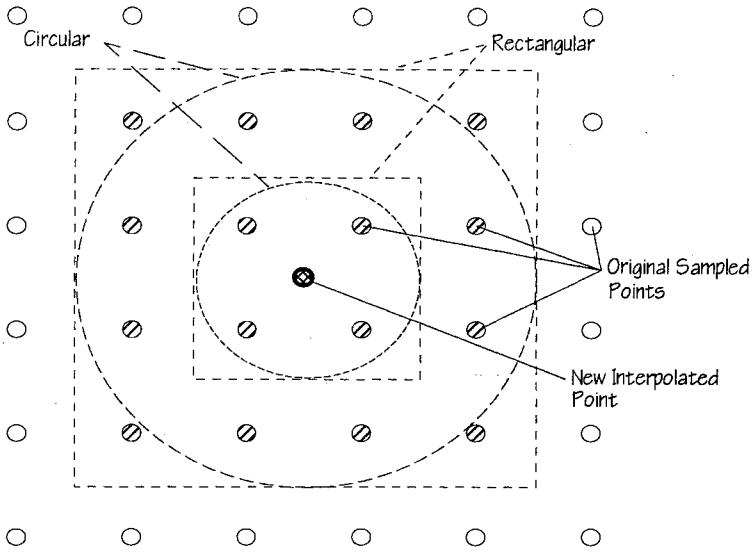


Figure 6-12. Rectangular and Circular Filters

Results

Speed Comparisons

In Figure 6-13, the two different methods of reconstruction are shown. The traditional method is shown in Figure 6-13a, and the new reconstruction process developed in this research is shown in Figure 6-13b. The primary difference between the two methods is when the reconstruction activity is performed relative to the IDCT operation. If the reconstruction is done after the inverse discrete cosine transform (IDCT) is computed, the spatial image is processed using the traditional method. If the reconstruction is done before the IDCT, the DCT coefficients are processed by the new method.

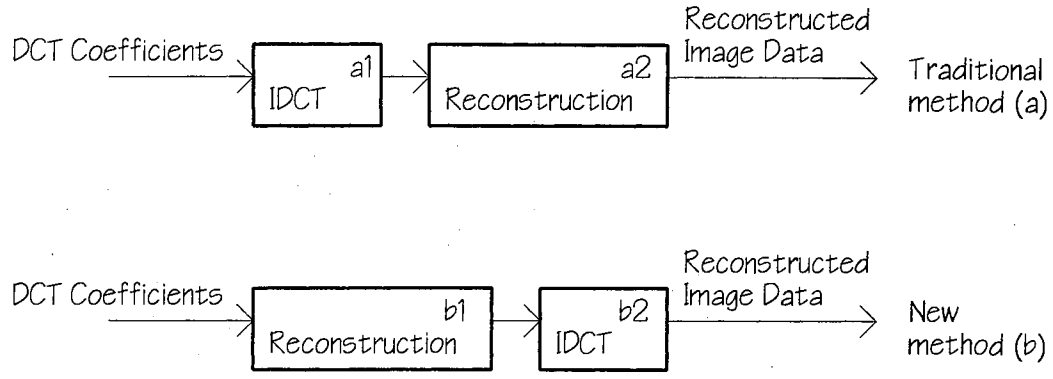


Figure 6-13. Reconstruction Methods: a) Traditional Method; b) New Method

Consider the algorithms in the two IDCT blocks (a1 and b2) in Figure 6-13. If the IDCT algorithms are the same and equally optimized, the speed of these blocks is directly related to the amount of data processed. For a standard JPEG algorithm, this means that for every new pixel produced, the algorithm processes the entire scaled DCT block once. In block a1 of the figure, the input block size is 8x8 and the output is 8x8. Thus 64 passes over 64 DCT coefficients, or 4096 operations are required. In block b2, the reconstruction has been completed and the input block size is 8x8, but the output block size is $8\zeta x8\zeta$. For block b2, this means $64\zeta^2$ passes over the 8x8 DCT coefficients, or $4096\zeta^2$ operations. This gives a sense of the difference in the amount of computation required for the IDCT blocks in the two methods. All things being equal, if it takes 1 time unit to perform the IDCT operation in block a1, it will take ζ^2 units to perform it in the new IDCT block b2, ignoring the $M = N\zeta$ round off.

The two reconstruction blocks in Figure 6-13 are labeled a2 and b1. The traditional method, a2, takes an 8x8 block of image pixels, and processes to an $8\zeta x8\zeta$ block of image pixels. The new reconstruction block, b1, takes an 8x8 block of DCT coefficients, and processes it to an 8x8 block of DCT coefficients. If we stop here, we can say that if the scaling factor is less than one (minification), the IDCT process is

improved by a factor of ζ^2 (from the IDCT operation). If the scaling factor is greater than one (magnification), the same factor of a power of 2 slows down the new IDCT process. In the reconstruction blocks, if the scaling is less than one, the speed of the traditional method is improved by ζ^2, while if the scaling is greater than one, the power of 2 slows down the speed of the traditional method.

However, before reaching any final conclusions, the two reconstruction blocks should be examined in more detail. In the traditional spatial case, a2, in order to generate one new pixel, the cubic convolution function must be evaluated five times (four times horizontal, and once vertical) with four points on each curve. To compute the cubic function, there are 5 multiplies and 4 adds in the center, where the inner two points are evaluated. There are 9 multiplies and 3 adds in the outer function, where the sharpening occurs. This equals 28 multiplies, and 14 adds per intermediate point generated. Since there are 5 intermediate points per new pixel, this gives 20 intermediate points generated for each pixel, for a total of 560 multiplies and 280 adds per new pixel generated.

In the frequency reconstruction case, b1, if the image is sufficiently large, the effect of computing the filter can be ignored, since it is done only once, before processing begins. The reconstruction is a multiply of each 8x8 DCT block. Each coefficient (and thus each new pixel) is one multiply, with no adds, compared to 560 multiplies and 280 adds in the traditional method. This is a dramatic difference, but should be considered a best guess only, since the actual implementation can vary these results. Several optimized algorithms [Duhamel 90][Chen 77][Lee 84][Narasinha 78][Suehiro 86][Vetterli 84] exist for IDCT evaluation as well as cubic function evaluation. Table 6-1 presents the average times to perform the reconstruction of the Lena image, using the cubic filter and the sharpened gaussian filter. Times for both the traditional spatial method, and the new cosine method are given. All of the times in Table 6-1 include reading the compressed image off the local disk, decompression, reconstruction, and writing the final image back to the local disk. The traditional method includes the time to

decompress the original JPEG image using a non-optimized IDCT algorithm. The JPEG IDCT definition is given in Equation 6-1.

TABLE 6-1

RECONSTRUCTION TIMES

	Reconstruction Time (seconds)[a]				
	ζ=0.50	ζ=0.75	ζ=1.00	ζ=1.25	ζ=1.50
Spatial Cubic[b]	41	84	301	707	1205
Spatial Cubic	976	1019	1236	1642	2140
Spatial Sharpened Gaussian[b]	292	706	1156	1967	2823
Spatial Sharpened Gaussian	1227	1641	2091	2902	3758
Cosine Cubic	237	516	944	1447	2090
Cosine Sharpened Gaussian	237	517	945	1449	2095

[a]For the Lena image (512x512), on a 80486 33MHz computer.
[b]Reconstruction time not including the IDCT process.

If an optimized IDCT is used, the reconstruction times for the spatial algorithms would be less. However, the spatial reconstruction time, not including the IDCT process, would be the same. For scaling values less than one, the new method is significantly faster. This is due to the reduced number of IDCT operations needed. For scaling values greater than one, the difference is not as large and diminishes, because the time to evaluate the expanded IDCT grows faster than the time to evaluate the spatial reconstruction curves.

Error Analysis

The method used for determining the error was to compute both the ideal and the reconstructed version of the test image at a given scale factor. Then a pixel-by-pixel comparison was performed, with the error being measured as a percentage of the total

pixel range. Both the spatial and the frequency interpolated images were rounded in the same manner, and both were subjected to clipping at 255, with negative values set to zero. The method used earlier in Figures 6-7b through 6-11b measures the response of the algorithms to a known mathematical function, a pulse at location zero. That method compares the results of the two filters without subjecting them to any clipping.

The individual component pixel percent error is:

$$\text{error} = \frac{\sqrt{\left[P_{\text{Ideal}} - P_{\text{Interpolated}}\right]^2}}{255}. \tag{6-7}$$

This was computed for each pixel and each color component (red, green, blue), and averaged to find the final interpolation percent error. In the case of positional errors, this was averaged over each block of the entire image, then plotted based on pixel position in the scaled block. The percent error was also computed for the DC image for comparison. The DC version was computed by only using the first coefficient (the DC coefficient) during the block reconstruction. This DC coefficient is the average of the 8x8 block.

Figure 6-14 shows the percent error averaged over the entire Lena image. The actual numbers given for percent error are somewhat misleading. The numbers indicate an error of about 3.5 percent for the DC coefficient image. According to the JPEG specification, 3 percent is regarded as the minimum difference the human eye can detect. However, the image reconstructed with just the DC coefficient is very noticeably distorted. Likewise, the other two images have noticeable imperfections. These numbers do give an idea of the relative distortion between the two different methods. As

expected, the cubic filter outperforms the sharpened gaussian filter when comparing average numerical errors.

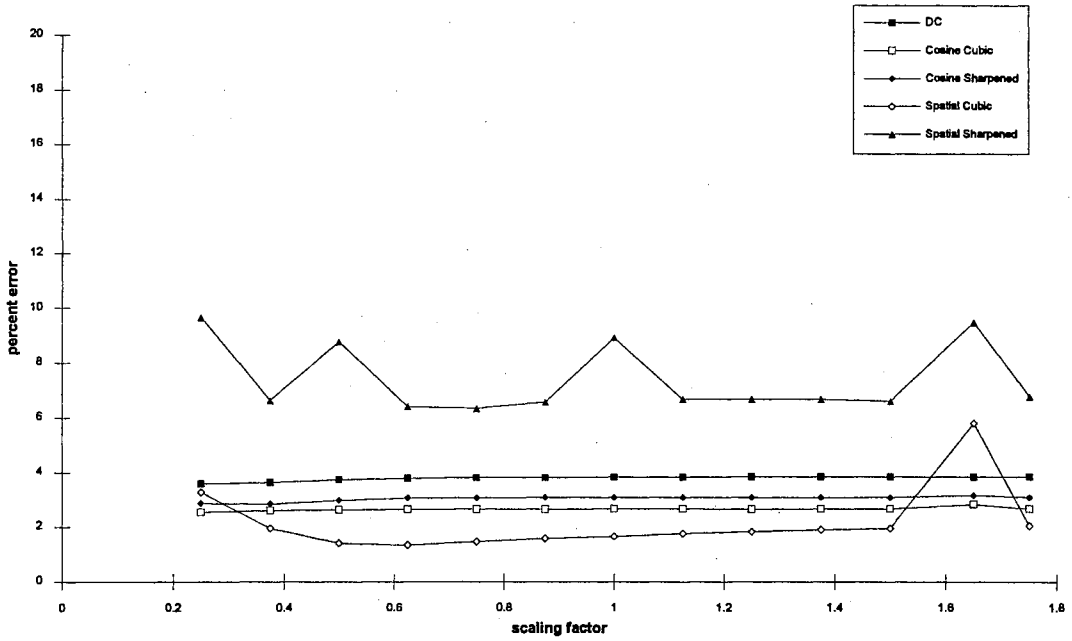


Figure 6-14. Total Error versus Scaling Factor for the Lena Image

Figure 6-15 shows the same error measures computed for the mandrill image. The mandrill image has more high frequencies than the Lena image. As expected, the DC mandrill image has a larger average error than the DC version of Lena.

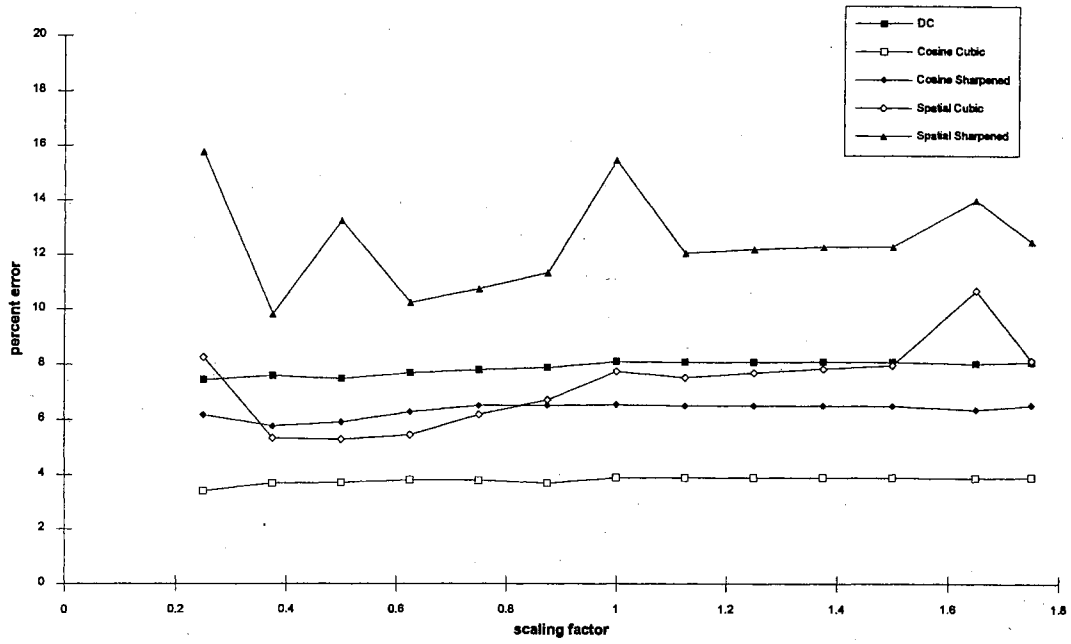


Figure 6-15. Total Error versus Scaling Factor for the Mandrill Image

In both cases (cosine cubic and cosine sharpened gaussian), the major objectionable visual distortion is the loss of high frequency information. Both of the reconstruction filters serve as low-to-mid pass filters because they smooth the data. This smoothing is related to the power of the interpolation function. For example, the cubic function can match a quadratic curve, but smoothes the higher frequencies. The high frequencies corresponding to those abrupt changes are lost.

This gives a good indication of where the visual artifacts can be expected. Numerically, even the highest distortion within a block is not bad (usually about 3 percent -- or in the just-noticeably-different range). If the entire block had a uniform error distribution, the images would be much better visually, even at the higher percent error. However, since each block has this error pattern, with the high frequencies having a higher than average distortion, the blocking is noticeable when viewing areas containing strong sharp lines that curve through several blocks, as in parts of the Lena image. Figure 6-16 shows the Lena image (ζ=1) after reconstruction in the cosine

domain using the cubic filter. The blocking artifacts are particularly noticeable along the mirror edge, and the shoulder edge. Figure 6-17 shows the mandrill image (ζ=1) after cubic cosine reconstruction. No blocking artifacts are visible because there are no high frequency edges that span several blocks.

Figure 6-16. The Cosine Cubic Lena

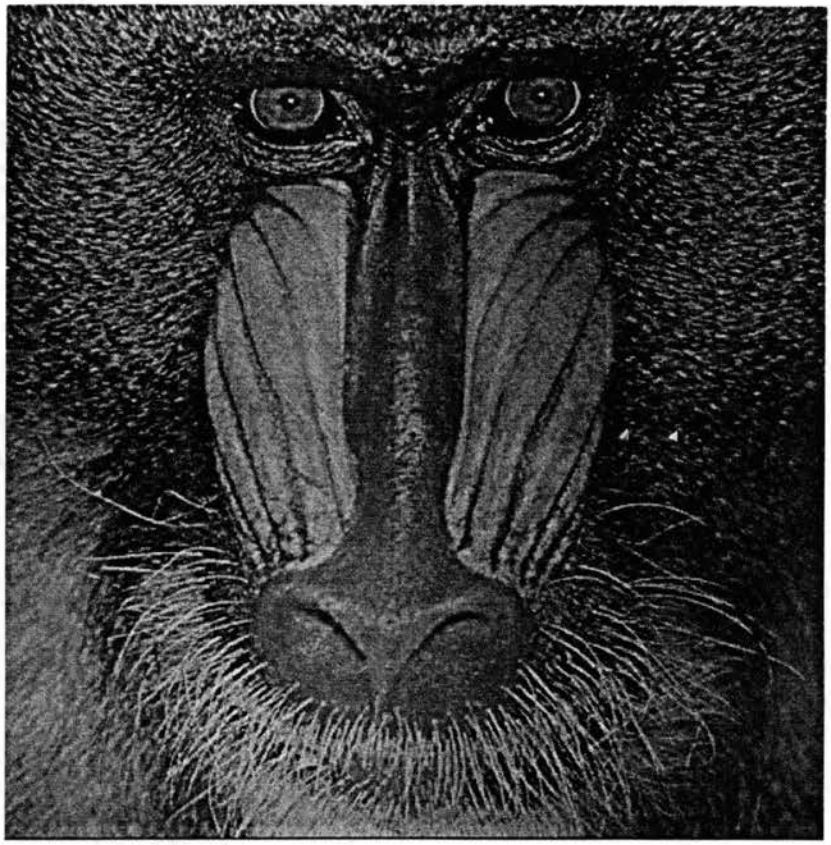


Figure 6-17. The Cosine Cubic Mandrill

Figure 6-18 shows the Lena image (ζ=1) after cosine domain reconstruction using the sharpened gaussian filter. Although the same blocking artifacts are present, the high frequencies are slightly enhanced compared to the cubic version. Both the cubic and sharpened gaussian versions are enhanced compared to the original, Figure 3-2. Again, the blocking artifacts are a result of working with the JPEG 8x8 DCT block, not a result of performing the reconstruction in the DCT domain.

Figure 6-18. The Cosine Sharpened Gaussian Lena

Figure 6-19 shows the mandrill image (ζ=1) after cosine domain reconstruction, using the sharpened gaussian filter. As with Figure 6-17, there are no blocking artifacts visible, and like Figure 6-18, the high frequencies are enhanced compared to the cubic version in Figure 6-17. The original mandrill image is shown in Figure 3-3.

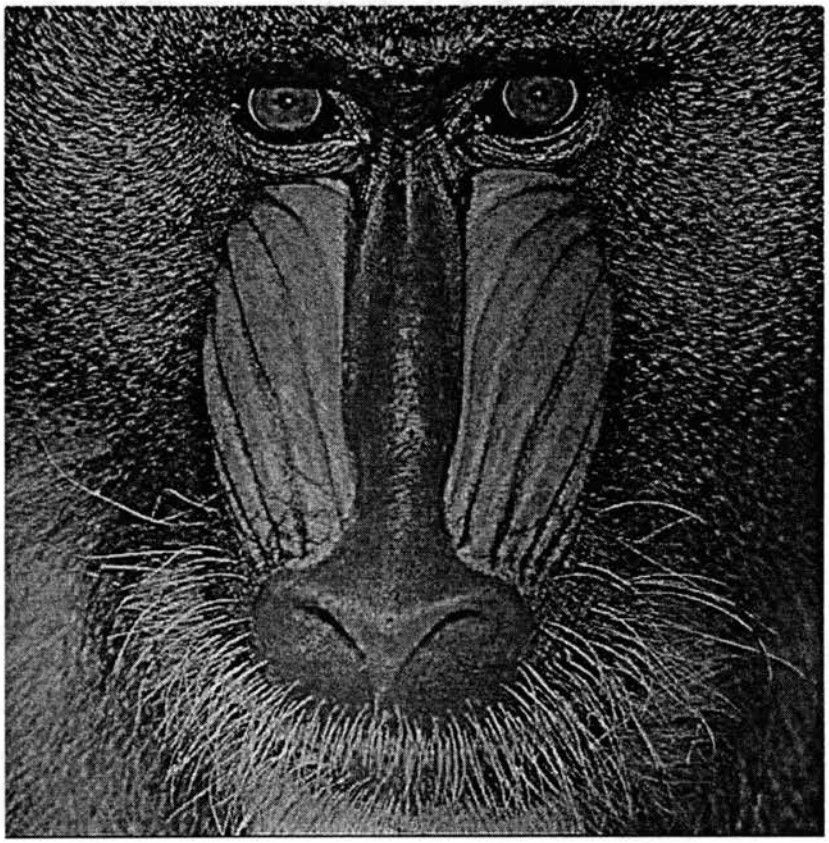


Figure 6-19. The Cosine Sharpened Gaussian Mandrill

Figure 6-20 shows a difference image for the Lena image (ζ=1). Figure 6-20 was generated by taking the difference between the ideal image and the cubic cosine image. Because the differences were too small to be visible, each pixel value was multiplied by 10. For reproduction purposes, the image was reversed, so the darker the pixel, the greater the error. The high frequency blocking artifacts are clearly visible in Figure 6-20. The low frequency background does not show the artifacts, which is as expected.

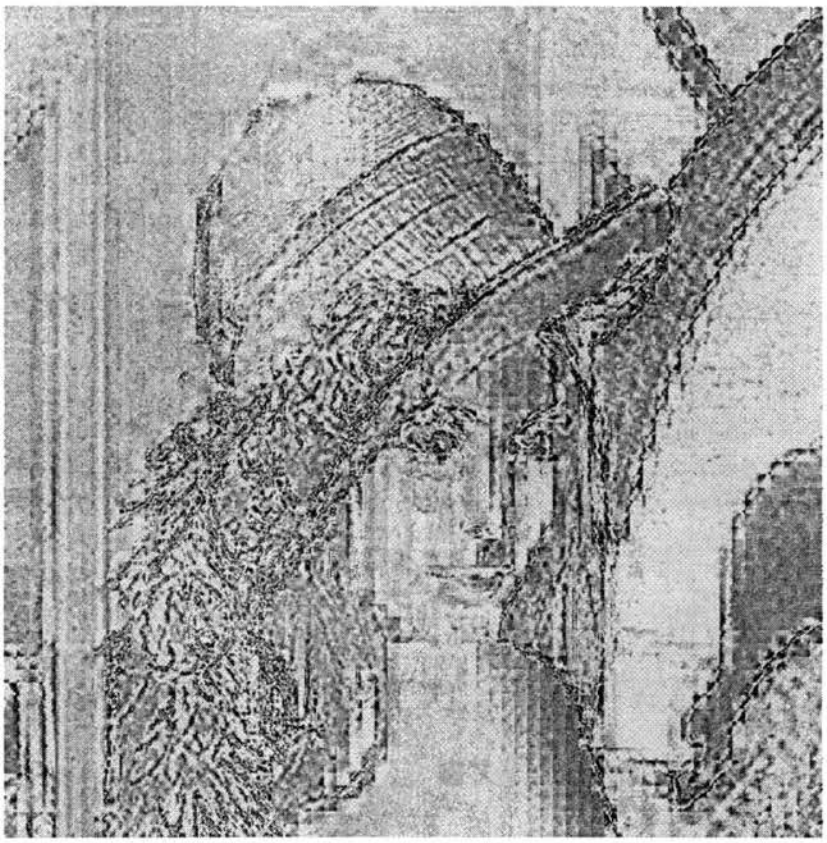


Figure 6-20. The Difference Image for Cosine Cubic Lena

The parts of the Lena image that do not contain strong inter-block edges, like the background, do not exhibit any noticeable blocking. The blocking was not noticeable in any region of the mandrill image, because the image is busy, and does not contain high frequency edges that span several blocks.

CHAPTER 7

SUMMARY

Problem

As the availability of higher-quality images increases, the need for mass storage of these images also increases. One method of increasing the capacity of mass storage devices is to compress the images before storage. The Joint Photographic Experts Group (JPEG) has proposed a method of compressing natural scene image data. The JPEG algorithm is based on a discrete cosine transform (DCT) that moves the spatial image data into the cosine frequency domain. The DCT coefficients can be quantized and compressed without adversely affecting the original image data.

However, the time needed to decompress the image back to a displayable format can be substantial. Once the image is decompressed, the image data frequently needs to be scaled to fit a particular display device or resolution. This last scaling step can be extremely time consuming if the reconstruction uses a high quality cubic, or gaussian curve to interpolate the data in two dimensions. This scaling delay can be a significant prohibiting factor in the use of high quality imaging systems.

Proposal Review

Traditional image reconstruction involves beginning with a digitized image, interpolating between the original image samples to recreate the continuous image signal,

scaling that continuous signal to the correct size, then resampling the scaled continuous image to create a new digitized image.

This research developed a method for performing the interpolation and scaling operations in the frequency domain, rather than the spatial domain. This idea fits nicely with the JPEG image compression algorithm, because the JPEG algorithm uses the discrete cosine frequency transform to compress the image. The goal of the research was first to determine if it was feasible, and second, if feasible, to determine what benefits this approach could offer.

If the scaling step can be done in the frequency domain as a multiplication, rather than in the spatial domain as a convolution, the resulting algorithm will be much faster. The new cosine domain-based reconstruction algorithm required three related operations. They are:

- filtering, which reconstructs the continuous image from the sampled image,
- scaling, which scales the continuous function to the new dimensions, and
- resampling, which samples the continuous function at the new locations.

The first phase of the research covered the spatial implementations of the cubic, and the sharpened gaussian reconstruction functions. These two spatial functions are well documented in previous research literature, and were chosen because of their wide acceptance as high quality image interpolation functions.

The second phase covered the frequency investigations. The frequency research starts in the Fourier domain, then extends the results to the discrete cosine domain. The purpose of the Fourier work was to provide a basis of understanding and comparison, since little research of this nature is available in the cosine domain. The frequency phase research produced two individual algorithms (cubic and sharpened gaussian) which were implemented and documented in the discrete cosine domain.

The third phase was the application of these algorithms to the JPEG data stream. There are some restrictions imposed on the algorithms by the JPEG standard, and this phase investigated the consequences of these restrictions.

Conclusions

This research has shown that performing the interpolation and scaling in the cosine domain is completely feasible. This method of reconstruction requires both the image and the reconstruction filter be implemented in the discrete cosine domain. Therefore, this method is particularly attractive for use with images compressed with the JPEG DCT algorithm. Otherwise, the computational cost of transforming the entire image to the cosine domain will likely be greater than the computational benefits produced by this method.

The new reconstruction method uses a precalculated filter, and the time to calculate the filter can be ignored if the image is of large enough size. While the speed-up of the new method is potentially large, the data observed in this research assumes the IDCT operation is a non-optimized version. If an optimized version of the IDCT were used in the traditional method, the observed speed-up would be smaller.

Using either method, a magnified image ($\zeta>1$) takes more time to produce and a minified image ($\zeta<1$) takes less time to produce. The computational time in both the traditional method and the new method are affected by the scaling factor, but not to the same degree. In the new method, the IDCT operation is slowed by ζ^2, while in the traditional method, the reconstruction operation is slowed by the same factor of ζ^2. Since the time to calculate the IDCT in the new method is less than the time to calculate the reconstruction curves in the traditional method, the new method is not influenced by the scaling factor to the same degree as the traditional method.

Depending on the image, there are visible artifacts that result from working in the JPEG DCT space. Since the DCT reconstruction filters are low-mid pass filters, the high frequencies are attenuated, as they are in the traditional method. However, the difference is that in the spatial domain, the entire image is processed. In the JPEG DCT domain, only an 8x8 block is processed at any one time. The result is that if the image has strong sharp lines (high frequencies) that span several blocks, as in the Lena image, the distortion introduced by high frequency loss (per block) may be noticeable. The distortion is also present in the spatially reconstructed image, and it can be numerically larger, however, the distortion is not patterned in blocks and is usually not apparent to the eye. This is the only case found where the cosine reconstructed images were visually different from the spatially reconstructed images. This type of distortion is a result of the 8x8 block imposed by the JPEG algorithm, and is not a result of operating in the frequency domain. Also, this distortion is image dependent. It is not apparent in the mandrill image, for example. Thus it is unlikely that the images produced by the new method and the traditional method will be distinguishable. Except in special cases (ζ=1), the new method resulted in numerically lower percent errors than the traditional method.

Unlike the traditional spatial case, the errors do not increase with the scaling factor. For both cosine filters, the percent errors for the new method varied less than one percent over a wide range of scaling factors. In the traditional method, the percent error increased with the scaling factor, and also exhibited significant fluctuations. Also in the traditional method, computational speed can be improved if the reconstruction curves are simpler (nearest neighbor, etc.). In the new method, the time to calculate the reconstruction filter is negligible compared to the rest of the process, so there is no computational benefit in using a simpler filter.

As expected, both methods introduced errors in the reconstructed image. Depending on the application and the image, the distortions (the loss of high frequencies) introduced may or may not be objectionable. For an average natural scene image, the

distortions are not objectionable. It is possible to perform an ideal reconstruction directly in the cosine domain, and bypass the reconstruction filter altogether. This special case of the new method has several advantages over the traditional method, and can offer improvements to the cosine reconstruction method. The speed of computation is improved because no cosine reconstruction filter is needed, and the resulting image has about 5 percent less error. However, the image may be prone to aliasing errors if the high frequencies are not removed in some manner.

As expected with the new method, pixelization was reduced, giving a better image when compared to nearest neighbor or other lower-order interpolation methods. Aliasing was not apparent (although it was not altogether eliminated) when viewing the images during minification, since the higher frequencies were reduced in this process.

Finally, the total percent error rates were much better than expected, usually within 3 percent (or less). The DCT based JPEG algorithm is lossy, and usually lies in this region of distortion (within 3 percent), although the quality is adjustable at the expense of the compression ratio.

Future Work

There are four areas of suggested future work; speed improvements to the current process, reducing reconstruction errors in the current process, evaluating reconstruction results, and improvements to the entire process.

The scaling and resampling portions of the new reconstruction method would be very useful for all imaging applications, except for the present requirement of using a non-optimized IDCT process. If the scaling and resampling capability case could be applied to one of the optimized IDCT algorithms, it would be possible to improve the speed of the entire process. This process could lead to fewer errors if applied to the subsampled components in a JPEG image.

Future study in the error reduction area should make use of a panel of viewers and evaluation should be carried out under controlled viewing conditions. Potential subjects for error reduction include clipping versus normalization tradeoffs, rectangular versus circular filters (or other possible filter shapes), and methods of interblock sampling when the sampling increment is larger than the block size.

Possible improvements to the process include cutoff frequency determination based on the frequency content of an individual block. Also, methods of varying individual block sizes during reconstruction would allow any size image to be reconstructed.

BIBLIOGRAPHY

[JPEG Draft 92] -, *Digital compression and coding of continuous-tone still images*, ISO/IEC DIS 10918-1, International Organization for Standardization, 1991, International Electrotechnical Commission, 1991.

[Abdou 82] Abdou, I. E. and K. Y. Wong, "Analysis of Linear Interpolation Schemes for Bi-Level Image Applications," IBM J. Res. Develop., vol. 26, no. 6, pp. 667-680, November 1982.

[Ahmed 74] Ahmed, N., T. Natarajan, and K. R. Rao, "Discrete Cosine Transform," IEEE Trans. on Computers, vol. C-23, pp. 90-93, Jan. 1974.

[Andrews 70] Andrews, H. C., *Computer Techniques in Image Processing,* Academic Press, New York, 1970.

[Andrews 72] Andrews, H. C., A. G. Tescher, and R. P. Kruger, "Image Processing by Digital Computer," IEEE Spectrum, vol. 9, no. 7, pp. 20-32, July 1972.

[Andrews 76] Andrews, H. C. and C. L. Patterson III, "Digital Interpolation of Discrete Images," IEEE Trans. Computers, vol. C-25, no. 2, pp. 196-202, February 1976.

[Arps 88] Arps, R. B., T. K. Truong, D. J. Lu, R. C. Pasco, and T. D. Friedman, "A multi-purpose VLSI chip for adaptive data compression of bilevel images,", IBM J. Res. Develop., vol. 32, no. 6, pp. 775-795, November 1988.

[Asal 86] Asal, M., G. Short, T. Preston, R. Simpson, D. Roskell, and K. Guttag, "The Texas Instruments 34010 Graphics System Processor," IEEE Computer Graphics and Applications, vol. 6, no. 10, pp. 24-39, October 1986.

[Duhamel 90] Duhamel, P., and C. Guillemot, "Polynomial Transform Computation of the 2-D DCT," Proc. IEEE ICASSP-90, pp. 1515-1518, Albuquerque, New Mexico, 1990.

[Chen 77] Chen, W. H., C. H. Smith, and S. C. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform," IEEE Trans. on Comm., COM-25, pp. 1004-1009, Sept. 1977.

[Chen 84] Chen, W. H., and W. K. Pratt, "Scene Adaptive Coder," IEEE Transactions on Communications, vol. COM-32, pp. 225-232, March 1984.

[Eldon 90] Eldon, J. A. and M. Sani, "Image Resampling," Advanced Imaging, vol. 5, no. 2, pp. 38-42, February 1990.

[Fant 86] Fant, K. M., "A Nonaliasing, Real-Time Spatial Transform Technique," IEEE Computer Graphics and Applications, vol. 6, no. 1, pp. 71-80, January 1986. See also "Letters to the Editor" in vol. 6, no. 3, pp. 66-67, March 1986 and vol. 6, no. 7, pp. 3, 8, July 1986.

[Goodman 68] Goodman, J. W., *Introduction to Fourier Optics*, McGraw-Hill Inc., New York, NY, 1968.

[Heckbert 86] Heckbert, P., "Survey of Texture Mapping," IEEE Computer Graphics and Applications, vol. 6, no. 11, pp. 56-67, November 1986.

[Hou 87] Hou, H. S. and H. C. Andrews, "Cubic Splines for Image Interpolation and Digital Filtering," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-26, pp.508-517, 1987.

[Jain 89] Jain, A. K., *Fundamentals of Digital Image Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[Keys 81] Keys, Robert G., "Cubic Convolution Interpolation for Digital Image Processing," IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-29, pp. 1153-1160, 1981.

[Langdon 84] Langdon, G., "An Introduction to Arithmetic Coding," IBM J. Res. Develop., vol. 28, pp. 135-149, March 1984.

[Lee 83] Lee, C. H., "Restoring Spline Interpolation of CT Images," IEEE Trans. Medical Imaging, vol. MI-2, no. 3, pp. 142-149, September 1983.

[Lee 84] Lee, B. G., "A New Algorithm to Compute the Discrete Cosine Transform," IEEE Trans. on Acoust., Speech and Signal Processing, vol. ASSP-32, no. 6, pp. 1243-1245, Dec. 1984.

[Leger 91] Leger, A., T. Omachi, G. K. Wallace, "JPEG Still Picture Compression Algorithm," Optical Engineering, vol. 30, no. 7, pp. 947-954, July 1991.

[Lohscheller 84] Lohscheller, H., "A subjectively adapted image communication system," IEEE transactions on Communications, vol. COM-32, pp. 1316-1322, December 1984.

[Maeland 88] Maeland, E., "On the Comparison of Interpolation Methods," IEEE Trans. Medical Imaging, vol. MI-7, no 3, pp.213-217, September 1988.

[Mitchell 88] Mitchell, D. P. and A. N. Netravali, "Reconstruction Filters in Computer Graphics," Computer Graphics, (SIGGRAPH '88 Proceedings), vol. 22, no. 4, pp. 221-228, August 1988.

[Mitchell 89] Mitchell, J. L., W. B. Pennebaker, and C. A. Gonzales "The Standardization of Color Photographic Image Data Compression," Digital Image Processing Applications, Proc. SPIE, Vol 1075, pp. 101-106, January 1989.

[Naiman 87] Naiman, A., A. Fournier, "Rectangular Convolution for Fast Filtering of Characters," Computer Graphics, vol. 21, no. 4, pp.233-242, July 1987.

[Narasinha 78] Narasinha, N. J., and A. M. Peterson, "On the Computation of the Discrete Cosine Transform," IEEE Trans. Communications, vol. COM-26, no. 6, pp. 966-968, Oct. 1978.

[Park 82a] Park, Stephen K. and Robert A Schowengerdt, "Image Reconstruction by Parametric Cubic Convolution," Computer Vision, Graphics, and Image Processing, vol. 23, pp. 258-272, 1983.

[Park 82b] Park, Stephen K. and Robert A Schowengerdt, "Image Sampling, Reconstruction, and the Effect of Sample-Scene Phasing," Applied Optics, vol. 21, no. 17, pp.3142-3151, September 1982.

[Parker 83] Parker, J. A., R. V. Kenyon, and D. E. Troxel, "Comparison of Interpolating Methods for Image Resampling," IEEE Trans. Medical Imaging, vol. MI-2, no. 1, pp. 31-39, March 1983.

[Pearson 91] Pearson, D., *Image Processing,* McGraw-Hill Book Company (UK) Limited, London, England, 1991.

[Pennebaker 88] Pennebaker, W. B., J. L. Mitchell, G. Langdon, Jr., and R. B. Arps, "An Overview of the Basic Principles of the Q-Coder Binary Arithmetic Coder," IBM J. Res. Develop., vol. 32, no. 6, pp. 717-726, Nov. 1988.

[Pratt 91] Pratt, W. K., *Digital Image Processing*, John Wiley and Sons, Inc., New York NY, 1991.

[Ratzel 80] Ratzel, J. N., "The Discrete Representation of Spatially Continuous Images," Ph.D. Thesis, Dept. of EECS, MIT, 1980.

[Ready 72] Ready, J., "Multispectral Data Compression through Transform Coding and Block Quantization," Ph.D. Thesis, Electrical Engineering, Purdue University, 1972.

[Rifman 74] Rifman, S. S. and D. M. McKinnon, "Evaluation of Digital Correction Techniques for ERTS Images - Final Report", Report 20634-6003-TU-00, TRW Systems, Redondo Beach, Calif., July 1974.

[Russ 92] Russ, J. C., *The Image Processing Handbook*, CRC Press, Inc., Boca Raton, FL, 1992.

[Schafer 73] Schafer, R W., and L. R. Rabiner, "A Digital Signal Processing Approach to Interpolation," Proc. IEEE, vol. 61, pp. 692-702, June 1973.

[Schreiber 85] Schreiber, W. F., and D. E. Troxel, "Transformation Between Continuous and Discrete Representations of Images: A Perceptual Approach," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. PAMI-7, no.2, pp. 178-186, March 1985.

[Suehiro 86] Suehiro, N. and M. Hatori, "Fast Algorithms for the DFT and other Sinusoidal Transforms," IEEE Trans. on Acoust., Speech and Signal Processing, vol. ASSP-34, no. 3, pp. 642-644, June 1986.

[Vetterli 84] Vetterli, M., and H. J. Nussbaumer, "Simple FFT and DCT Algorithms with Reduced Number of Operations," Signal Processing, Aug. 1984.

[Wallace 91] Wallace, G. K., "The JPEG Still Picture Compression Standard," Communications of the ACM, vol. 34, no. 4, pp. 31-44, April 1991.

[Wintz 72] Wintz, P., "Transform Picture Coding," Proc. IEEE, vol. 60, no. 7, pp. 809-820, July 1972.

[Wolberg 90] Wolberg, G., *Digital Image Warping*, IEEE Computer Society Press, Los Alamitos, Ca, 1990.

VITA

Jeffery Kersey Otto

Candidate for the Degree of

Doctor of Philosophy

Dissertation: IMAGE RECONSTRUCTION FOR DISCRETE COSINE TRANSFORM COMPRESSION SCHEMES

Major Field: Electrical Engineering

Biographical:

Personal Data: Born in Midland, Michigan, March 5, 1963, the son of Kenneth and Virginia Otto.

Education: Graduated from Jenks High School, Jenks, Oklahoma, in June 1981; attended University of Michigan from September 1981 to May 1984; received Bachelor of Science Degree in Electrical and Computer Engineering from Oklahoma State University in May 1986; received Master of Science Degree in Electrical and Computer Engineering from Oklahoma State University in May 1988; completed requirements for the Doctor of Philosophy in Electrical and Computer Engineering at Oklahoma State University in December, 1993.

Professional Experience: Engineering Manager, Imaging Products Division, TMS Inc., 1989 to present. Instructor and Research Assistant, Department of Electrical and Computer Engineering, Oklahoma State University, 1986 to 1991. Software Engineer, Seagate, 1987 to 1989. Diagnostics Engineer, Compaq Computer Corp., 1985 to 1986.

Professional Organizations: IEEE, ACM, Tau Beta Pi, NSPE