**Chapter 6: The Discrete Cosine Transform**

1. **Introduction**

The Discrete Cosine Transform (DCT) [28] has been used in most of the international standards for image compression over the last decade. It produces a high level of decorrelation in the image data presented to it and can be implemented using a fast algorithm. The forward and inverse DCTs are shown in equations 6.1 and 6.2 respectively :

** (6.1)

 (6.2)

Where *N, M* are block side lengths,

is the image pixel intensity at *i,j*,

are the transform coefficients at *k,l*,

and ,

.

The DCT shown in equations 6.1 and 6.2 is orthonormal and perfectly reconstructing, provided the coefficients are represented to an infinite precision. The coefficients of the DCT are always quantised for image compression, but the DCT is very resistant to quantisation errors due to the statistics of the coefficients it produces. The coefficients of a DCT are usually linearly quantised by dividing by a predetermined quantisation step.

The DCT is applied to image blocks *N*x*N* pixels in size (where *N* is usually *2n*) over the entire image. The size of the blocks used are an important factor since they determine the effectiveness of the transform over the whole image. If the blocks are too small then the image is not effectively decorreated but if the blocks are too big then local features are no longer exploited.

The tiling of any transform across the image leads to artifacts at the block boundaries. The DCT is associated with blocking artifact since the JPEG standard suffers heavily from this at higher compressions. However the DCT is protected against blocking artifact as effectively as possible, without interconnecting blocks, since the DCT primitives/basis functions all have a zero gradient at the edges of their blocks. This means that only the DC level significantly effects the blocking artifact and this can then be targeted.

Ringing is a major problem in DCTs. When edges occur in an image the DCT relies on the high frequency components to make the image shaper. However these high frequency components persist across the whole block and although they are effective at improving the edge quality they tend to ‘ring’ in the flat areas of the block. A typical example is shown in figure 6-1. This ringing effect becomes worse, when larger blocks are used, but larger blocks are better in compression terms, so a trade off is usually established. Neither of the new methods proposed in this chapter eliminates ringing, but the problem is reduced compared to JPEG.

The limited coefficient quad-tree DCT (LQT-DCT), discussed in chapter 4, reduces ringing and will be compared to the new methods in this chapter. It can be argued that having no ringing is more visually pleasing than obtaining a better rate distortion performance.



Figure 6-1. Diagram of edge improved with high frequency components to demonstrate ringing.

The JPEG standard has been around since the late 1980’s and has been an effective first solution to the standardisation of image compression. Although JPEG has some very useful strategies for DCT quantisation and compression, it was only developed for low compressions (~20:1). The 8x8 DCT block size was chosen for speed (which is less of an issue now, with the advent of faster processors) not for performance. The JPEG standard will be briefly explained in this chapter to provide a basis to understand the new DCT related work.

This chapter contains two new and different approaches to using the DCT. The first uses a variable number of DCT coefficients and a quad-tree structure. The aim of this variable coefficient DCT is to establish what is best, a quad tree with a highly truncated DCT (chapter 4), a full DCT method where all the coefficients are used, or somewhere in between. Secondly, a DCT method is presented that optimally quantises the coefficients of a DCT using a fixed block size, using the theory from chapter 5.

**6.2 JPEG**

The JPEG (Joint Photographic Experts Group) standard [10] has been around for some time and is the only standard for lossy still image compression. There are quite a lot of interesting techniques used in the JPEG standard and it is important to give an overview of how JPEG works. There are several variations of JPEG, but only the ‘baseline’ method will be discussed here.

JPEG uses an 8x8 DCT, as previously shown in equations 6.1 and 6.2. The DCT is implemented using fast algorithms, such as those proposed by Feig [51], which only requires 54 multiplies, 464 adds and 6 binary shifts to compute the 8x8 DCT (in comparison to at least 4096 multiplies and 4096 adds that it usually requires). To improve the precision of the DCT the image is ‘zero shifted’, before the DCT is applied. This converts a 0 🡪 255 image intensity range to a -128 🡪 127 range, which works more efficiently with the DCT.

After the DCT coefficients have been calculated, they are quantised by dividing them by a quantisation step, which is different for each coefficient. The JPEG quantisation table is shown in table 6-1 for the luminance component of the image.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 6-1. Quantisation table for JPEG luminance component, as specified by the JPEG standard.

The quantisation steps are generated by multiplying the quantisation table values by a quality factor, depending on what compression is required. If the effects of the quantisation table are compared to the quantisation model used in chapters 2, 3 and 4, then the quantisation factor Q=~2 for this table. This is much lower, than the optimal for the model applied here, but this is to be expected, since this is the unmodified table (100% quality). Also the JPEG model uses a psycho-visual reason for quantising in the way it does. This is a failing in JPEG, since although psycho-visual reasoning is given for the quantisation table used, there is no conclusive evidence that it is the best quantisation table for an 8x8 block.

After the DCT coefficients have been quantised, the DC coefficients are DPCM coded (delta coding from chapter 2) and then they are entropy coded along with the AC coefficients. The quantised AC and DC coefficient values are entropy coded in the same way, but because of the long runs in the AC coefficient, an additional run length process is applied to them to reduce their redundancy.

The range of the DC coefficient is described by a 4 bit size symbol (SSSS) and additional bits are used to specify the exact coefficient within this range, as shown in table 6-2. The quantised DC coefficient below is coded as follows:

Cq=7 🡪 SSSS: 0011(3) Additional bits: 111

This method is similar to using a Huffman PDF for every quantised coefficient, but allows a much bigger variation in the coefficient size (efficiency for speed gain).

|  |  |  |
| --- | --- | --- |
| SSSS | Range of Cq | Additional Bits |
| 0 | 0 | 0 |
| 1 | -1, +1 | 1 |
| 2 | -3, -2, +2, +3 | 2 |
| 3 | -7, -6, -5, -4, +4, +5, +6, +7 | 3 |
| 4 | -15, -14, -13, -12, -11, -10, -9, -8, +8, +9, +10, +11, +12, +13, +14, +15 | 4 |
| . |  |  |
| . |  |  |
| 15 |  | 15 |

Table 6-2. Table showing SSSS symbol structure in JPEG

The AC coefficients are all examined in a zig-zag order as shown in figure 6-2. The run length in this zig-zag order is described by a RUN-SIZE symbol (RRRRSSSS). The RUN is a count of how many zeros occurred before the quantised coefficient and the SIZE symbol is used in the same way as it was for the DC coefficients, but on their AC counter parts. The two symbols are combined to form a RUN-SIZE symbol (since they are both 4 bits) and this symbol is then entropy coded. Additional bits are also transmitted to specify the exact value of the quantised coefficient. A size of zero in the AC coefficient is used to indicate that the rest of the 8x8 block is zeros (End Of Block or EOB). An example of the run length coding is shown in figure 6-3.



Figure 6-2. Diagram showing Zig-Zag ordering in 8x8 DCT block.



Figure 6-3. Demonstration of JPEG run length coding.

Before the compressed data is transmitted, a header is attached that describes the image details. These details include image size, JPEG method used, Quantisation tables, Huffman tables etc. This header can be transmitted multiple times in a single image to provide some error control, but only whole rows can be transmitted (not true sections of the image).

The JPEG performance on the Goldhill test image is shown in figure 6-4 and sections of Goldhill are shown at varying compressions in figure 6-5. There are three points to notice about JPEG.

* The compressor becomes unreliable at about 0.1-0.3 BPP and the images retrieved are very bad, as shown in figure 6-4.
* There are two types of artifacts produced by JPEG. Poor DC level transmission causes blocking artifacts. Ringing also occurs at lower compressions than the DC level artifact but is eventually completely replaced by the blocking artifact.
* The quality figure is not linearly related to the compression and so there is no precise control over the compression of JPEG.



Figure 6-4. A rate distortion performance graph for JPEG with the Goldhill test image.



1. Quality 100% (b) Quality 80% (c) Quality 60%

0.3035 RMSE 3.533924 RMSE 4.542825 RMSE

1. BPP 1.309 BPP 0.860 BPP



1. Quality 40% (e) Quality 20% (f) Quality 10%

5.303 RMSE 6.750RMSE 8.784 RMSE

1. BPP 0.396 BPP 0.240 BPP

Figure 6-5. Section of Goldhill compressed to different quality values using JPEG.

It can be seen that at high compression JPEG is not ideal, but the lower compressions are quite acceptable. The only real problem with low compression JPEG is that the compression ratio is not controllable.

The JPEG coding scheme is suitable for low compression ratios up to about 20:,1 but it breaks down at higher compression.

Extensions can be made to the JPEG standard and Sherlock and Monro [52] have made several advances.

**6.3 Variable Coefficient DCT**

The variable coefficient DCT is an attempt to find the middle ground between the JPEG standard (high complexity DCT) and the methods used in chapter 4.

Other methods using the DCT, but with different block sizes and coding schemes, have been shown to give good fidelity over a wider range of compressions. Sherlock and Monro [52] developed a DCT compression technique, which extends the JPEG approach to larger sized blocks, and models quantisation coefficients in terms of position in the quantisation table and a desired compression ratio. Because the achieved compression ratio is not generally equal to the desired ratio, this approach does not allow specification of the exact compression required with an arbitrary image. This is the same problem encountered by JPEG and it makes the system difficult to use in some applications.

Hierarchical coding methods often use a simple function (or transform) to represent the image content within blocks. Often the aim of a hierarchical method is to distribute the coded error evenly across the image. The error can be spread across the image by several methods. These include thresholding the error in each block to be the same, or sorting through a list of errors associated with each block and splitting the block having the highest error. The error in a block can be estimated by using an edge image (which is highly correlated with error), or by coding and decoding to explicitly calculate the error. The error is calculated according to some error metric, the most frequently encountered being Mean Squared Error (MSE). These methods were discussed in chapter 4 and were not quite as effective as JPEG. Their effectiveness also varied from image to image depending on its content.

The distribution of detail in an image limits the performance of hierarchical methods. An image whose areas of detail are small and isolated can be coded more effectively by a complicated hierarchical structure, but as the total amount of image detail increases the overhead of a large hierarchical structure begins to outweigh its benefits. This implies that simple hierarchical structures (such as the quad-tree) can be applied successfully to a wide range of images, but are not optimal at the extremes of high and low detail.

Ideally a block-based image coder should be effective on arbitrary images. This is not usually the case and different coders are suited to images having different amounts of detail, as illustrated in figure 6-6.



Figure 6-6. Stylised diagram of the range of application of compression techniques.

In this section a coder is described that is effective across a wide range of images and compressions. It is progressive, allowing any image to be coded to a precisely specified compression ratio, and the error is evenly distributed across the image. This property is well suited to the effective coding of error images, which can be useful in video compression. It uses a simple quad-tree structure and a DCT based block coding approach similar to that used in JPEG, but using variable block sizes.

**6.3.1 Quantisation**

All lossy image compression techniques require a quantisation stage. The DCT produces a set of coefficients that are particularly easy to quantise, by using linear quantisation (chapter 2.3). Different quantisation steps (*q*) are used for different DCT coefficients, since the energy tends to be concentrated towards low frequencies, and heavy quantisation can be used to effectively 'ignore' some higher frequency coefficients (JPEG).

Unlike in the previous low complexity coders (described in chapters 3 and 4), the quantisation factor *(Q)* is described with a simple quadratic function and it is different for each coefficient. This approximation is similar to the shape used to generate the suggested JPEG quantisation table used by Sherlock and Monro [2] in their model of quantisation :

 (6.3)

where

*r* is the distance from the origin,

*Q* is the quantisation factor,

*cq*and *sq* are constants in the quantisation model.

The model is simple, but results show that quantisation is not critical in this method.

Although the coder uses several different block sizes, the same shape of quantisation table is used throughout. Since the range of the DCT coefficients changes with block size, it is necessary to scale the quantisation function in the usual way :

 (6.4)

where *N* is the block size of the DCT,

*q* is the quantisation step,

and *Q* is the quantisation factor.

**6.3.2 Overview of compressor**

In designing the proposed coder the following three features were of primary concern:

(a) the coder must be progressive (able to reach any compression ratio exactly),

(b) the error power must be evenly distributed over the image,

1. the rate distortion performance must be near optimal.

To produce a progressively coded image the coder must gradually add detail to the whole image, until the desired compression is reached. Use of fixed block size with a fixed quantisation scheme does not allow a progressive code; either the quantisation or the block size must be variable. Variable quantisation does not give predictable compression ratios (JPEG quality), but variable block size can be exploited in a more predictable way.

To segment the image a quad-tree structure was chosen for the following reasons:

1. Quad-trees have a simple structure, requiring very little overhead to store.
2. The blocks in the quad-tree are square and can have sides that are 2n in length, allowing the use of simple fast DCT algorithms (similar to FFT).
3. Complicated structures are not necessary, because the structure was only intended to extend the number of bits that could be applied to one area of the image. The major adaptive feature of the system comes from the variable number of coefficients used.

To make the coder progressive, it was decided to assemble each DCT block adaptively. Each block is processed in much the same manner as in JPEG, i.e. runs of zeros followed by coefficients, until the end of the block is declared by a marker. The essential difference between JPEG and the new method is that the coding of a block can be terminated at any point after the first (DC) coefficient.

By gradually increasing the bit budget available to the coder, the image is progressively built up throughout the coding process. The image coder assigns newly available bits to the areas of highest error within the image. The regions, where the extra bits are assigned change throughout the coding process, as the error is slowly reduced. This leads to an evenly distributed error across the image and to near-optimal rate-distortion performance.

**6.3.3 Quad-tree structure**

The quad-tree structure was used in this coder to allocate extra bits in regions of high error without changing the quantisation. To decide, where to place the additional bits a list of the errors in each quad-tree block is maintained throughout the coding process (see Figure 6-7, section 6.3.6). This allows the block with greatest error to be found and improved in fidelity, at any point during coding. The improvement is achieved by increasing, by a fixed amount, the number of bits used to code this block. The improved block can then be placed at the appropriate point in the error list and the next most important block coded. The process of improving one block at a time continues until the available bit budget is exhausted.

The number of bits to be allocated at each step is arbitrary, but it was found empirically that about 40 bits (~ 10 coefficients) was close to optimal. Fewer bits than 40 did not reduce the block error enough to consistently displace it from the top of the error list, and more than 40 bits caused the error to be more coarsely distributed across the image. However, in either case the rate distortion performance of the system was only marginally affected.

The error metric used in our system was squared error. Squared error was used rather than Mean Squared Error (MSE), because different block sizes are compared to each other in the list of errors. This effectively means that the MSE of the whole image is minimised, by evenly distributing the error power across the image. Unfortunately MSE is not always a good measure of perceived image quality, and improvements could be possible if a suitable psychovisually based metric were available.

**6.3.4 Entropy Coding**

The entropy coding of the quantised DCT coefficients is broadly similar to the JPEG approach. The differences will now be discussed .

The quad-tree structure does not permit DPCM coding of the DC coefficient. Therefore the DC coefficient is coded by a simple linear quantisation.

The AC coefficients were coded in the same way as JPEG, i.e. Huffman coding Run-Size symbols. However, the Huffman tables suggested in the JPEG standard were not appropriate for use in this system, because of the different quantisation and variable block sizes. Instead Huffman tables were determined from a probability density function generated from a large set of images.

**6.3.5 Intelligent Block Splitting**

It has been seen how the quad-tree structure determines which block to improve. It is also necessary to decide how to make the improvement, i.e. where to allocate the additional bits used to improve a particular block. The choice to be made is as follows :

(1) use the additional bits to increase the number of DCT coefficients used in the larger block, or

1. split the larger block into four smaller ones and use the bits already allocated to the larger block, plus the additional bits, to code the 4 smaller blocks.

The error metric is evaluated for (1) and (2) above, and the option with the least error is chosen.

The above calculation may seem expensive, but it must be remembered that for a given section of the image, each size of block has its DCT calculated only once. If the error metric is squared error, the truncation error of a DCT is easy to calculate for progressive truncation of the DCT.

The algorithm to compress an image is shown in figure 6-7.

1. The coder is initialised by dividing the image into 32x32 pixel blocks. Each block is coded using the first 3 non-zero coefficients of the DCT. The squared error associated with each block is calculated, and these are formed into a sorted list.
2. The block having the greatest error (i.e. the one at the top of the list) is improved by increasing the number of DCT coefficients used until a fixed additional number of bits (~40) has been added (as explained in section 6.3.3). The resultant squared error is calculated.
3. This block is divided into four equally sized smaller blocks. The total number of bits needed in step (2) to code the original block is divided equally between the four smaller blocks. This is done by simultaneously adding 1 coefficient to each smaller block until the bit budget is reached. The resultant squared error of the four smaller blocks is calculated.
4. The block is either kept in its improved form (from step 2) or split into 4 smaller blocks (as in step 3) depending upon which yields the smallest total squared error.
5. The resultant block or blocks are re-inserted into the sorted list at the appropriate points.
6. Steps 2 to 5 are repeated, eroding the error across the image until the target bit budget / compression ratio is reached.



Figure 6-7. Diagram of the block sorting and decision process for the variable coefficient DCT.

**6.3.6 Experiment: Vary the quantisation constants  and  to find the optimal shape of the quantisation tables.**

The method described in the previous section was applied to the Goldhill test image at three different compressions. At each fixed compression the quantisation constants ( and ) were varied to find their optimal values.

The results of these experiments are shown in Figure 6-8 at 0.2, 0.2667, and 0.4 BPP. Examination of Figure 6-8 shows that for all three compression ratios the lowest error is obtained when  is zero. This implies constant quantisation. Because the curves of MSE versus  for  are relatively flat (see Figure 6.8), and the minimum does not occur at a consistent  value, we recommend a mid-range value of  as being a reasonable compromise over all compression ratios. It was found that the choice ,  worked well over a wide range of images. This is an interesting results since it applies roughly the same quantisation as the unmodified JPEG quantisation table but constant over all the coefficients.



Figure 6-8. Diagrams of varying quantisation factors (*sq* and *cq*) for the Goldhill image at (a) 0.4 BPP (b) 0.267 BPP and (c) 0.2 BPP

**6.3.7 Experiment: Compare the variable coefficient DCT with normal and extended JPEG.**

Figure 6-9 shows rate-distortion curves comparing the variable coefficient DCT (using the optimal values of  and ) with the normal and the extended JPEG approach using fixed block sizes [52].

The variable coefficient DCT shows an improvement over standard JPEG at all compressions but the extended JPEG is still more efficient. Although this indicates that the variable coefficient DCT is not as effective as the extended JPEG there are still certain advantages to it. The new method can reach any compression exactly and also spreads the reconstruction error more evenly across the image, as shown in figure 6-10. Distributing the error evenly across the image allows the compressor to allocate the compression more effectively. Figures 6-11 and 6-12 shows the corresponding recovered images for comparison.

Figure 6-9. Rate-distortion curves comparing the variable coefficient DCT with normal and extended JPEG.



Figure 6-10. Error images at 0. 2BPP associated with fragment of Goldhill image, for (a) new approach and (b) extended JPEG approach



Figure 6-11. Variable Coefficient DCT at 0.2 BPP on Goldhill.



Figure 6-12. Extended JPEG at 0.2 BPP on Goldhill

**6.4 Optimal Quantisation**

The principle investigated in chapter 5 can be used to great effect with the DCT as discused by Bethel et al. [4]. It is first necessary to investigate some problems introduced by using a DCT with the method. In this case the DCT is applied to a fixed NxN tiling of the image surface. The blocks are either 4x4, 8x8 or 16x16, so that the fast Cosine transform can be applied to speed up the compression process.

The coefficients generated by the DCT need to be quantised. In this case a linear quantiser was appropriate since multiple sets of quantisation need to be applied in the method and anything else would be too slow. The aim of the compressor is then to produce one quantisation table which can be used to quantise DCT blocks all over the image. This quantisation table should produce the optimal MSE for the exact number of bits required in the DCT coder.

The coefficients need to be losslessly compressed after quantisation to produce the final compressed file. In the process to produce the multiple rate distortion curves it was necessary to apply a wide range of quantisation and a lot of data was produced. To avoid the overhead of losslessly compressing each data item the theoretical entropy was used to estimate the number of bits associated with a particular quantisation. The final set of quantised coefficients were compressed with an arithmetic compressor since the entropies indicated that a lot of symbols has less that one bit/symbol.

The additions to the method in chapter 5 will be discussed in more detail in the following section and the results for 4x4, 8x8 and 16x16 blocks will be compared to each other to establish the best block to use.

**6.4.1 Generation of the Rate Distortion Curves for the Equal Gradient method applied to the DCT.**

The equal gradient method described in chapter 5 discusses in detail how to finding the optimal number of bits or compression that should be applied to each coefficient set. The method uses a set of rate distortion curves (one for each coefficient set) but it does not deal with how the rate distortion curves are generated.

The application to the DCT uses a linear quantisation method as in equation 6.5 :

 (6.5)

where *cq* is the quantised coefficient,

*c* is the unquantised coefficient,

and *q* is the quantisation step.

It is necessary to generate a number of points on the rate distortion curves of each coefficient set in the DCT. These points need to be distributed correctly between the quantisation limits so that information is not wasted. The quantisation limits are *qn*  (which is the quantisation step that completely truncates the all the data) and *q0* (which is the quantisation step that integerises the already integer DCT (*q=1*)). Quantisation is not a linear process, but the compressions produced are more linearly spaced, if the quantisation steps are chosen to vary as in equation 6.6 :

 (6.6)

where *qi* is the quantisation step,

*i=0,1,2…n,*



and max is the maximum coefficient value in the set.

In this work *n*=10 was found to be satisfactory, by experiment, as described in chapter 5.

When the quantisation steps have been calculated they are applied to the appropriate DCT coefficient set and a set of quantised coefficients are generated. The error in each quantisation step is evaluated as the sum of the squares of the errors between the quantised and unquantised coefficients. The number of bits required to store the quantised coefficients set is calculated from the theoretical entropy (equation 6.7).

 (6.7)

where *pdd(x)* is the probability density distribution of symbol *x.*

The equal gradient method is then passed to a set of ‘support points’ to describe the rate distortion performance of each coefficient set. It returns the number of bits required to compress each coefficient set at the optimal point. Since this needs to be converted to quantisation steps to be useful, the data from the rate distortion curve calculations is used to linearly interpolate the number of bits to the correct quantisation.

**6.4.2 Experiment: Implement the optimal quantisation process for DCTs with different block sizes and compare.**

The process described in chapter 5 and the pervious section was implemented for the DCT at three different block sizes (4x4, 8x8 and 16x16). An arithmetic lossless coder was used to compress the quantised coefficients and it was initialised from a store of PDDs to improve its performance.

The rate distortion curves are shown in figure 6-13 for the three block sizes. It is clear that the 16x16 block size is better in rate distortion terms than either the 4x4 or 8x8 DCT. The 16x16 block size has the best performance since it combines the improved compression provided by a highly correlated transform with a sufficiently high spacial resolution. The sections of Goldhill shown in figure 6-14 demonstrate this but there is increased ringing artifact in the 16x16 blocks. This is to be expected with larger blocks but there is not enough degradation to warrant using the smaller block sizes which have much larger RMS (at 0.2 BPP).



Figure 6-13. Rate distortion performance curve for Goldhill using 4x4, 8x8 and 16x16 block size DCT.



1. 4x4 DCT (b) 8x8 DCT (c) 16x16 DCT

10.324 RMSE 8.789 RMSE 8.151 RMSE

Figure 6-14. Sections of Goldhill at 0.2 BP using 4x4(a), 8x8(b) and 16x16(c) block size DCT.

1. **Experiment: Compare JPEG to the 16x16 Optimal quantisation DCT.**

The 16x16 optimal quantisation DCT is compared to JPEG in figure 6-15 . It can be seen that the new DCT is better at all compressions. A section of Goldhill at 0.2 BPP compares JPEG to the 16x16 DCT in figure 6-16, this shows that JPEG still has substantially more ringing that the new method. Although the new method is far better than JPEG visually, it is still slightly blurred as shown in figure 6-17. This could be because more high frequencies are available to clear up the ringing, but not enough to make the image sharper.

Figure 6-15. Rate distortion performance curve comparing JPEG to the 16x16 optimally quantised DCT.



(a) JPEG (b) 16 x 16 DCT

Figure 6-16. Sections of Goldhill at 0.2 BPP comparing JPEG to the 16x16 optimally quantised DCT.



Figure 6-17. Goldhill at 0.2 BPP compressed using the 16x16 optimal quantised DCT.

**6.5 Summary**

In this chapter the existing JPEG standard has been reviewed and compared to two new methods that have been proposed. It is interesting to see, how the methods compare to each other and those proposed in chapter 4 as shown in Figure 6-18. This shows the rate distortion performance curves for the best implementation of the variable coefficient DCT, Optimally quantised DCT, the LQT-DCT and JPEG.

Figure 6-18. Rate distortion curves for all the effective DCT methods mentioned in this work.

The two new high complexity methods show an improvement on JPEG in rate distortion terms (optimally quantisation being the most effective), but their algorithms can only be implemented as fast as JPEG and they are still susceptible to ringing. The LQT-DCT is resistant to ringing and very fast (requiring a fraction of the calculations needed by a complex DCT method) but it does not compare favorably with JPEG rate distortion performance.

The use of image coders is not just limited by their RMS performance, there can be other factors which favour which coder to use. For instance in low bit rate video applications the LQT-DCT would be acceptable but at higher bit rates its could not compete with the high complexity methods (unless speed is important).