SEPARABLE KARHUNEN LOEVE TRANSFORMS FOR THE WEIGHTED UNIVERSAL TRANSFORM CODING ALGORITHM

Hanying Feng *Michelle Effros*

Department of Electrical Engineering
California Institute of Technology
Mail Code 136-93
Pasadena, CA 91125, USA

ABSTRACT

The weighted universal transform code (WUTC) is a two-stage transform code that replaces JPEG's single, non-optimal transform code with a jointly designed collection of transform codes to achieve good performance across a broader class of possible sources. Unfortunately, the performance gains of WUTC are achieved at the expense of significant increases in computational complexity and larger codes. We here present a faster, more space-efficient WUTC algorithm. The new algorithm uses separable coding instead of direct KLT. While separable coding gives performance comparable to that of WUTC, it uses only 1/8 of the floating-point multiplications and 1/32 of storage of direct KLT. Experimental results included in this work compare the performance of new separable WUTC with both the WUTC and other fast variations of that algorithm.

Keywords: WUTC, separable coding, KLT

1. INTRODUCTION

Transform coding is one technique used to achieve the performance benefits associated with high dimensional coding at lower computational expense. In transform coding, high-dimensional data vectors are first sent through a transformation designed to decorrelate the vector components and then compressed using a collection of scalar quantizers.

One of the most common transforms used in transform coding for images and video is the discrete cosine transform (DCT). The DCT is a good decorrelating transform for "smooth," "natural" images. In the JPEG image coding standard, the DCT is combined with a collection of scalar quantizers. While the JPEG image coding standard allows the user to specify a new collection of scalar quantizers (in the form of a quantization matrix) for each image, many implementations of JPEG use a single default quantization matrix for all images. This matrix is scaled up and down to achieve a range of rates.

The JPEG algorithm's performance is hurt by the algorithm's rigidity. The optimal transform code is data dependent. The optimal collection of scalar transforms varies from data set to data set. The optimal decorrelating transform is a function of the data statistics. While the JPEG algorithm allows for variations in the scalar transforms it allows no such variation in the transform itself. Further, many implementations of the JPEG image coding standard do not take advantage of the allowed variation in the collection of scalar quantizers since the design of scalar quantizers is considered to be too computationally intensive to be included in the encoding procedure for each incoming image.

The Weighted Universal Bit Allocation (WUBA) algorithm [**?**] is a DCT-based transform code that replaces the single quantization matrix used in JPEG with a collection of quantization matrices. The quantization matrices are jointly and optimized off-line during a training procedure, thereby removing the quantization matrix design from the encoding procedure. The encoder chooses among the quantization matrices in its collection on an image by image or block-by-block basis. Since choosing among a collection of quantization matrices is much more computationally efficient than designing a new quantization matrix for each image, the resulting code gives a good computation / performance tradeoff. Further, since the collection of quantization matrices is fixed, quantization matrix description requires very little rate, since each quantization matrix may be described by its index in the list of quantization matrices.

While the WUBA algorithm goes a long way towards addressing the limitations of the JPEG algorithm, it too relies on the DCT, which is not an optimal transform for all possible data sets. In particular, while the DCT achieves good decorrelating performance on smooth, natural images, it achieves poor performance when used to decorrelated images with more high-frequency content, such as computer generated graphics or images containing text. For applications involving broader data sets, significant performance

THIS MATERIAL IS BASED UPON WORK SUPPORTED BY NSF CAREER AWARD NO. MIP-9501977, THE INTEL 2000 PROGRAM, AND THE POWELL FOUNDATION.

benefits may be achieved by using alternate transforms in place of the DCT.

The Karhunen Loeve Transform (KLT) is a good alternative to the DCT for image coding applications. The KLT, which is a transform composed of the normalized eigenvectors of the correlation matrix associated with a specific set of source statistics, is a data dependent transform that in some sense achieves the optimal decorrelation and energy compaction for image compression. Based on the theory of stochastic processes, this method can decorrelate data completely if the data is stationary. Yet because of its data-dependence, the KLT had traditionally not been very popular for image compression.

The Weighted Universal Transform Code (WUTC) [?] is a two-stage algorithm that adds KLT transform coding to the WUBA algorithm. The resulting algorithm contains a collection of transform codes. Each transform code contains a KLT and a quantization matrix optimized for some subset of the training data. The transform codes in the collection are jointly and optimally designed off-line during a training procedure. As in the WUBA algorithm, the encoder chooses among the transform codes in its collection on an image by image or block-by-block basis. Since the collection of transform codes is fixed, the description of the chosen transform code requires very little rate, since each transform code may be described by its index in the list of transform codes.

The WUTC achieves significant gains over the DCT-based WUBA algorithm and the single-quantization matrix JPEG algorithm, yet this distortion-rate performance improvement is achieved at the expense of higher computational complexity and more prohibitive memory requirements. In particular, optimal implementation of the WUTC algorithm on 64-dimensional data vectors (8×8 data blocks as in the JPEG algorithm) uses a collection of 64-dimensional KLTs. Given an image containing N 8×8 data blocks and a KLT containing K transform codes, encoding a single image requires at least $K \times N \times 64 \times 64$ floating-point multiplications. Thus the computational complexity is very high. The storage complexity associated with the WUTC algorithm is also high. The dimension of each transform matrix is 64×64. So for the typical coder and decoder require $K \times 64 \times 64 = 4096K$ coefficients to describe the code's KLT matrices. (A typical value for K is 64.)

The goal of this work is to find alternative transform codes achieving the performance gains of the WUTC algorithm at a lower expense in computation and memory.

2. THE SEPARABLE WEIGHTED UNIVERSAL TRANSFORM CODING ALGORITHM

In this paper, we replace the 64-dimensional KLT of the WUTC algorithm with a separable pair of KLTs – one of which decorrelates 8×8 data block column by column, and the other of which decodes the data row by row. The combined pair of transforms is a sub-optimal transform that approximates the performance of direct KLT, with 1/4 of the floating point multiplications of the KLT. The transform is stored as a pair of 8×8 matrices, here denoted by L and R, where L performs the transformation on the columns and R performs the transformation on the rows. The transformation operation is then implemented as $Y = LXR$, where X is an 8×8 matrix of source coefficients and Y the corresponding 8×8 matrix of coefficients in the transform domain. Implementation of the matrix multiplications may be further stream-lined using the Winograd algorithm [?], as described in the Appendix. Using this approach, described briefly in the Appendix, the pair of matrix multiplications requires only $4 \times 8 \times 9 \times 2 = 576$ floating-point multiplications – roughly 1/8 those required per transformation using a 64-dimensional KLT.

The WUTC algorithm using a separable pair of KLTs is here called a separable WUTC (SWUTC). Designing an optimal SWUTC requires the joint design of a collection of optimal separable transform codes. The joint design algorithm is a variation on the optimal design algorithm described in [?], where the KLT design is replaced with a separable KLT design. The design algorithm uses an iterative descent technique. The initial collection of codes is chosen at random. Each iteration proceeds as follows.

1. Optimize the encoder for the decoder. Map each data vector in some training set to the transform code that reproduces that data vector with the best distortion-rate performance.

2. Optimize the decoder for the encoder. Redesign each transform and each quantization matrix to match the data that mapped to the associated transform code. The transform is designed using the KLT design algorithm on the row statistics for the design of R and the column statistics for the design of L. See the Appendix for a discussion of the transform design.

3. Optimize the entropy code. Redesign the entropy code used in describing the transform codes. The entropy-coded description length of each transform code should be chosen to match the negative log probability of that transform code.

The above three steps are iterated until convergence.

The SWUTC implemented using the Winograd algorithm reduces both the complexity and the memory required to implement the WUTC algorithm. A SWUTC with K transform codes uses $N \times [K \times (8 \times 68) + 32] = (544K + 32)N$ floating point multiplies for an image of N 8×8 data vectors as compared to the $N \times K \times 64 \times 64 = 4096KN$

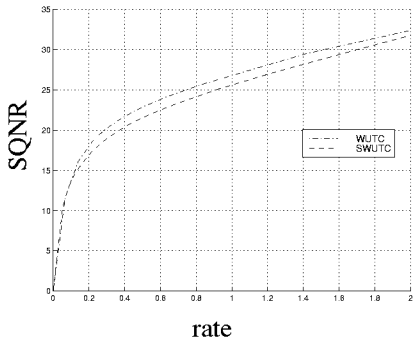

Figure 1: WUTC with the full KLT compared to WUTC with the separable KLT.

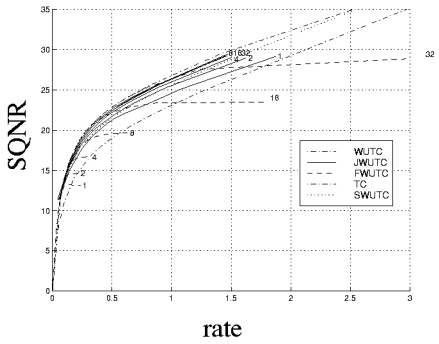

Figure 2: WUTC and a single optimal transform code (TC) are compared with fast alternatives. The FWUTC and JWUTC algorithms achieve complexity and memory reductions of a factor of $k/64$ relative to the complexity of a full WUTC by removing rows from the WUTC's transformation matrix. (Each curve is labeled by the associated value of k.)

floating point multiplies in a WUTC of the same size. Similarly, the SWUTC requires storage of $K \times 2 \times 8 \times 8 = 128K$ transform coefficients as opposed to the $4096K$ required for the WUTC. Thus the SWUTC reduces the complexity and memory of a WUTC by factors of approximately 8 and 32 respectively.

Since the statistics of the rows and columns of an image are typically similarly, memory may be further reduced by using the same transformation on the rows and the columns ($L = R$ in the above transform equation). The resulting code is called a single SWUTC (SSWUTC). The resulting code requires the same computational complexity as the SWUTC, but reduces the memory requirements by another factor of 2.

3. EXPERIMENTAL RESULTS

The following summary of results compares the distortion-rate performance of the SWUTC and SSWUTC algorithms to that of a variety of alternative transform coding algorithms. In each case, rate is measured as entropy and distortion is given by signal to quantization noise ratio (SQNR). Each system is designed using a 2048×2048 image as its training set. A different 2048×2048 image is used as the test set in all experiments. Each image is a different scanned page from the *IEEE Spectrum Magazine*, and both contain similar proportions of images and text. Each weighted universal code (e.g., the SWUTC, SSWUTC, WUTC, WUBA algorithms) uses a maximum of 64 codes per collection ($K = 64$) and allows a new code to be described for each 64-dimensional vector.

Figure 1 compares the performance of the WUTC and SWUTC algorithms. The SWUTC, which employs a suboptimal transform, achieves performance roughly 1 dB worse than that of the WUTC. Thus the move from optimal KLT to suboptimal separable KLT costs 1 dB in rate-distortion performance but yields an $8\times$ improvement in complexity and $64\times$ improvement in memory requirements.

Figure 2 compares the SWUTC to several alternate fast WUTC algorithms – namely the JWUTC and FWUTC algorithms described in [?]. The FWUTC and JWUTC algorithms achieve complexity and memory reductions to a factor of $k/64$ times the complexity of a full WUTC by removing rows from the WUTC's transformation matrix. (Each curve is labeled by the associated value of k.) In FWUTC, the number of rows per transform matrix is kept constant from transform to transform. In JWUTC, the number of rows per transform matrix is allowed to vary from transform to transform. As shown in Figure 2, the SWUTC gives far better performance than either the JWUTC or FWUTC of the same size (here labeled by "2"). From a complexity stand-point, the SWUTC is roughly comparable to the JWUTC and FWUTC labeled "8." While the SWUTC easily outperforms the FWUTC, it suffers a slight performance degradation when compared to the JWUTC of the same complexity, but this code requires $4\times$ the memory of the SWUTC.

Figures 3 and 4 compare the performance of SWUTC to SSWUTC and DCT-based codes respectively. In Figure 3, the SSWUTC suffers a further 1 dB performance degradation when compared to SWUTC but requires half as much memory. The DCT codes (JPEG, which uses a single quantization matrix and WUBA which uses up to 64 quantization matrices) give significantly poorer performance than either the WUTC or the SWUTC due to the use of the DCT.

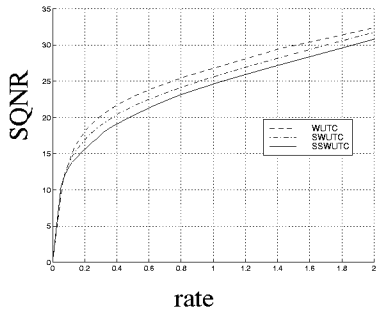

Figure 3: WUTC with SWUTC and SSWUTC.

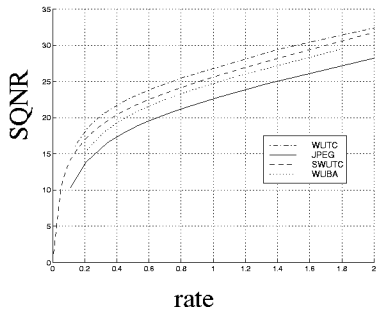

Figure 4: Separable WUTC compared to DCT-based transform codes.

4. APPENDIX

Separable KLT Design

It is tempting to believe that the original data should be used to train the column transform matrix L and that the transform row transform matrix R should be designed on the data given by the rows of LX rather than the rows of X. In reality, however, both L and R may be simultaneously designed from the original data. This is observation results from the unitary nature of the transform matrices. In particular, if $L = [L_1^t L_2^t \cdots L_8^t]^t$ (that is L_i is the ith row of matrix L, the rows of LX are $L_1X, L_2X, ..., L_8X$, and the autocorrelation matrix of these rows is given by

$$\begin{aligned}
&[(L_1X)^t(L_1X)] + ... + [(L_8X)^t(L_8X)] \\
&= [X^t(L_1^tL_1)X] + ... + [X^t(L_8^tL_8)X] \\
&= X^t(L^tL)X = X^tX
\end{aligned}$$

(since L unitary implies $L^tL = I$). Thus, the auto-correlation matrix of the rows of the transformed data equals the auto-correlation of the rows of the original data.

An introduction of Winograd algorithm

The Winograd algorithm is a fast algorithm for reducing the number of multiplications required for an inner product. Let $a = [a_1, a_2, ..., a_{2n}]$ and $b = [b_1, b_2, ..., b_{2n}]$ be two vectors of dimension $2n$. Direct calculation of the inner product $ab^t = \sum_{i=1}^{n} a_i b_i$ requires $2n$ multiplications. This number can be reduced as follows. First calculate the "pre-inner products" $a^* = \sum_{i=1}^{n} a_{2i-1}a_{2i}$ and $b^* = \sum_{i=1}^{n} b_{2i-1}b_{2i}$. Notice that $ab^t = \sum_{i=1}^{n}[(a_{2i}+b_{2i-1})(a_{2i-1}+b_{2i}) - a^* - b^*$. While this does not represent a savings for multiplying two vectors, it may yield a savings if the above vectors appear in matrix multiplications where each vector may be used more than 1 time but a^* and b^* need be calculated only once. For example, consider multiplying an $m \times 2n$ matrix $A = [A_1^t, A_2^t, \cdots, A_m^t]^t$ by the $2n \times k$-matrix $B = [B_1, B_2, \cdots, B_k]$. Evaluating the pre-inner-product of all the row vectors of A and all the column vectors of B, requires $m \times n + k \times n$ multiplications. Using the Winograd algorithm requires $m \times k \times n$ more multiplications. So we need $m \times k \times n + m \times n + k \times n$ multiplications in total as compared with the $m \times k \times 2n$ multiplications needed to calculated the multiplication directly. When m or k is very large, this savings can greatly reduce the number of multiplications.

Since the transform matrices in SWUTC are fixed, we can evaluate the pre-inner-products of the transform matrices in advance. Then, when we send the data through a SWUTC with K transform codes, we only need to evaluate the pre-inner-products of the data matrix columns once and the pre-inner products of the (transformed) data matrix rows K times. This reduces the number of multiplications to $K \times (8 \times 68) + 32$ multiplications per 64-dimensional data vector.