CCITT SGXV
Working Party XV/1
Specialist Group on Coding for Visual Telephony.

Document # 381

September 1988

SOURCE: Norway

TITLE: Coding with Directional Transforms as alternative to DCT in RM5.

1. Introduction.

The main purpose of this document is to use a new class of transforms as alternatives to DCT. The new transforms will be named Directional Transforms (DT). The idea behind using DT is, that the picture content shall determine which one out of several - 16 used here - transforms will describe the picture most efficiently.

In order to decide which transform to use, we need to find the orientation of the data block. This determines which one out of 16 transforms will be used. If the direction is horizontal or vertical, or if the data does not have a distinct orientation, DCT is used. The details on the DT and on how to find the directions are given in the APPENDIX.

The rest of the coding is tried kept as close as possible to RM5.

2. Results.

Figure 1 gives a measure related to the advantage of DT over DCT. It shows the square sum of the energy in transform coefficients in falling order. It is shown both for DCT and DT. The statistics is obtained from the Claire sequence. The information from Figure 1 is, that the DT consentrates more energy in fewer coefficients. This is exactely what we want to obtain with transform coding. We have estimated the overall saving of bits to code transform coefficients to be approximately 20 % using DT instead of DCT. For one dimensional structures like edges the saving is much higher.

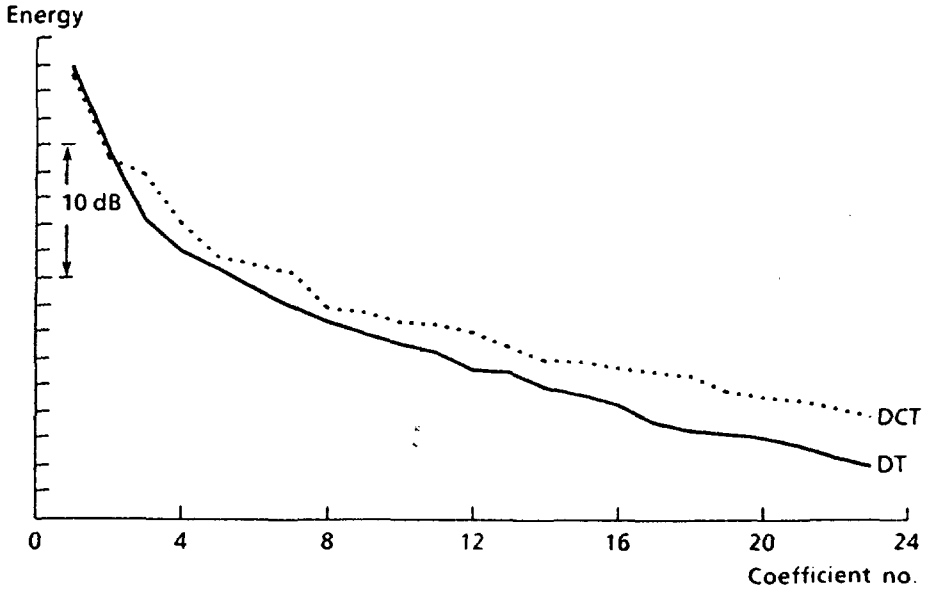

Figure 1. Example of energy distribution over transform coefficients for DT and DCT. The statistics is obtained from 165 pictures of the Claire sequence.

An equally - or even more - important gain, however, is the reduced quantization noise and "cleaner" reconstruction around edges. This is of great importance for the subjective quality.

It is our feeling that the use of DT still has more potential for compression. The knowledge of orientations may for example be used both to limit the number of motion vectors and switching of loop filter.

3. Conclusion.

The use of Directional transforms for coding of natural images has been introduced. The saving in bits is estimated to be approximately 20 %. In addition there is a considerable gain in subjective quality through cleaner edges. It is therefore proposed that this type of transform is included as a supplement to RM5.

Furthermore, the potential of the method has not yet been fully exploited More work both on the transform and how to use it is needed.

APPENDIX

Definition of directional transform and how to obtain the direction.

1. The Directional Transform.

A correlation description of the connection between neighbouring picture elements is useful to understand the difference between DT and DCT.

Assume that we have two points m and n, as indicated. The distance between the points is d, which may be decomposed into d1 and d2, depending on the orientation θ. The correlation between the points may be defined in different ways:

$$\varrho(m,n) = \varrho_0^{\,(|d1(\theta)| + e\cdot|d2(\theta)|)} \tag{1}$$

$$\varrho(m,n) = \varrho_0^{\,\sqrt{d1(\theta)^2 + (e\cdot d2(\theta))^2}} \tag{2}$$

d1 and d2 are measured in pixels. ϱ_0 is a correlation constant which depends on the picture material and sampling density. The parameter e indicates that the correlation may be different in the two directions.

Equation (1) results in correlation that is separable along the two directions d1 and d2. Separable correlation results in separabel transforms which give computational savings if the directions are along the sampling directions of the image. Equation (2) gives a more realistic model where the correlation depends on the real distance rather than the sum of the two components.

We may then define a correlation matrix $\underline{K}$ defining the correlation between each point:

$$\underline{K} = \begin{vmatrix} 1 & \varrho(m,n) \rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow\rightarrow & \text{m} \\ \varrho(m,n) & 1 & & & & & & & & & & & \\ \downarrow & & 1 & & & & & & & & & & \\ & & & 1 & & & & & & & & & \\ \downarrow & & & & 1 & & & & & & & & \\ & & & & & 1 & & & & & & & \\ & & & & & & 1 & & & & & & \\ & & & & & & & 1 & & & & & \\ & & & & & & & & 1 & & & & \\ & & & & & & & & & 1 & & & \\ & & & & & & & & & & 1 & & \\ & & & & & & & & & & & 1 & \\ & & & & & & & & & & & & 1 \end{vmatrix} \text{n}$$

A set of base vectors or transform functions may be obtained by solving the eigenvalue equation:

$$\underline{K} \cdot \underline{A} = \Lambda \cdot \underline{A} \tag{3}$$

$\underline{A}$ is a set of n·m vectors over all the n·m picture elements.

If we use equation (1) with e=1 and let $\varrho_0 \rightarrow 1$, the resulting eigenvectors $\underline{A}$ are the basis vectors of the Cosine transform.

In the method described here, I use eigenvectors resulting from using a value of e of typically 0.2 and for a set of different values of θ. I have used 16 different values that cover the range 0 - 180 deg. We call the resulting set of eigenvectors $\underline{A}(\theta)$.

We have therefore produced a number of different sets of transform vectors. Each set is specially suited to describe image blocks where the picture content has a certain orientation.

For the horizontal and vertical directions I have chosen to use the Cosine

transform which fulfills the requirement of coding one-dimensional structures with few coeffisients. Another reason for this choice is, that blocks without a special orientation are best described by the DCT. The DCT therefore acts as both directional transform in the horizontal and vertical directions and as a sort of default transform that is used when no particular orientation has been found. This will also be dealt with in the next section.

2. Finding the direction.

An essential part of the present method is to find the main direction of the data in a square block. Besides, we want to know how "well oriented" the data block is. By that I mean how well the data can be described with a one-dimensional function. I call this the quality of the direction - Q. In this section I will show how information on both the aspects may be obtained from a set of calculated block parameters.

Assume that the intensity over the block is represented by a continuous scalar field:

$$Y(x;y) \tag{4}$$

The gradient of this field is:

$$\nabla Y = \partial Y/\partial x \cdot \underline{i} + \partial Y/\partial y \cdot \underline{j} \tag{5}$$

Let θ represent a direction. We then define a vector:

$$\underline{R} = \cos\theta \cdot \underline{i} + \sin\theta \cdot \underline{j} \tag{6}$$

We define the direction of the block data as the value θ_0 that optimizes $(\nabla Y \cdot \underline{R})^2$ over the block. We requiere that the first derivative of the expression with respect to the direction is zero. This gives us the optimal direction. The double derivative of that same expression tells how sharp this maximum is - which again means how equally oriented the gradient field is over the block.

Over a discrete block of data we have to estimate the gradient of the intensity field from pixel values. The gradient components between the points (m,n) and (m+1,n+1) are approximated to:

$$\partial Y/\partial x \approx D1 = (Y(m+1,n+1)+Y(m+1,n)-Y(m,n+1)-Y(m,n))/2 \tag{7a}$$

$$\partial Y/\partial y \approx D2 = (Y(m+1,n+1)-Y(m+1,n)+Y(m,n+1)+Y(m,n))/2 \tag{7b}$$

The expression to be optimized is:

$$S = \sum_{1}^{N-1} \sum_{1}^{M-1} (D1\cdot\cos\theta + D2\cdot\sin\theta)^2 \tag{8}$$

We will need three different sums for the computation of direction and quality of that direction:

$$S1 = \sum_{1}^{N-1} \sum_{1}^{M-1} D1\cdot D2 \tag{9}$$

$$S2 = \sum_{1}^{N-1} \sum_{1}^{M-1} D1\cdot D1 \tag{10}$$

$$S3 = \sum_{1}^{N-1} \sum_{1}^{M-1} D2\cdot D2 \tag{11}$$

The derivatives of equation (8) are:

$$\partial S/\partial\theta = 1/2\cdot\sin 2\theta\cdot(S3-S2) + \cos 2\theta\cdot S1 \quad (12)$$

$$\partial^2 S/\partial\theta^2 = \cos 2\theta\cdot(S3-S2) - 2\cdot\sin 2\theta\cdot S1 \quad (13)$$

Requiering that $\partial S/\partial\theta = 0$ gives the solution for the optimal direction θ_0:

$$\theta_0 = 1/2\ \mathrm{Atan}(2S1/(S3-S2)) \quad (14)$$

For this value for θ, the double derivative has the value:

$$\partial^2 S/\partial\theta^2(\theta_0) = -\sqrt{(S2-S3)^2 + 4\ S1^2} \quad (15)$$

If the gradient field over the whole block is oriented in the same direction, it can be shown that equation (15) becomes -(S2+S3). As the quality criterion for the obtained direction θ_0, we therefore use:

$$Q = ((S2-S3)^2 + 4S1^2)/(S2+S3)^2 \quad (16)$$

The value of Q varies from 0 for uncorrelated gradient components to 1 when all the gradient components have the same orientation.

As already mentioned, the DCT is used as a default transform. The decision of which transform to use is made in three steps.

- If (S2 + S3) is below a certain limit we use DCT. This actually means that the whole block may be described by a DC component.
- If Q is below a certain limit we use DCT. θ_0 determines if we shall use the horizontal or vertical scanning of the DCT.
- When Q is above the limit, the DT corresponding to θ_0 is used.

The value of θ_0 is quantized. The outcome gives directly the number of the transform to be used for describing that particular block of data.

The directions are calculated both for the block to be coded and for the prediction. The difference in direction is VLC coded. An average of 7 bits pr. MB was used to code directions.

3. On complexity.

I will only deal with the additional number of additions and multiplications needed when we use DT and DCT.

The additional complexity comes from finding the direction and from computing the transform coefficients. No fast algorithm has been found for this, except that the coefficients may be grouped as symmetric and antisymmetric. This means that sums and differences between pixels may be calculated once, so that the subsequent summations to obtain the coefficients are performed over a matrix half the size of the data block.

Moreover, it should be kept in mind that the main advantage of DT is, that we need fewer coefficients to describe the block content. This means that it is sufficient to calculate approximately ten coefficients in a 8·8 matrix. This is taken into account in the numbers given in Table 1. For comparison, I have also included the number of operations needed for a full 8·8 DCT with fast algorithm, for a loop filter and for the motion compensation with the 3-step method.

Table 1. Number of operation of typical functions needed for DT coding.

	Additions/pixel	Multiplications/pixel
Finding direction for the data block:	6	3
Transform:	6	5
DCT 8·8:	6.5	4
Loop filter 3·3 taps:	6	1
MC with 3-step method:	25	