Implementation of the DCT transform with application to the JPEG transform

1. DCT transform in 2-D

1.1. THEORETICAL BACKGROUND

Proposition 1. *The two dimensional DCT of $m \times n$ matrix A is the product*

$$\widehat{A} = C_m A C_n^T \tag{1}$$

where the matrix C_N has elements

$$C_N(k, r) = u_k \cos\left(\frac{\pi}{N} k \left(r + \frac{1}{2}\right)\right) \tag{2}$$

with $u_0 = \sqrt{\frac{1}{N}}$ and $u_k = \sqrt{\frac{2}{N}}$ for $k > 0$. The inverse DCT is computed by

$$A = C_m^T \widehat{A} C_n \tag{3}$$

1.2. MATLAB EXERCISES

The JPEG transform uses the DCT on blocks of 8×8 pixels, so it is profitable to precompute C_N for $N = 8$. In Matlab, write a script that compute C_8 using (2) and store the result as a variable named `C8` in a file with the same name. Be careful that Matlab starts its indices at 1.

Check that the transpose of C_8 is indeed the inverse of C_8 (if not, check your script)

2. JPEG encoding

2.1. BLOCK DIVIDE

The first task is to divide the grayscale images into blocks of 8×8 pixels. if the dimension of the image is not multiple of 8, you will have to pad the image with zeros. You can use any greyscale image of your choice.

2.1.1. Matlab exercise

Write a function that will split an image matrix into small blocks of size 8×8.

2.2. DCT BLOCK

Now use the C8 matrix to perform a block DCT on each of the split images.

2.3. QUANTIZATION

A quantization matrix for the JPEG encoding is

$$\mathtt{DCTQ} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 194 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 121 & 100 & 103 & 99 \end{bmatrix} \tag{4}$$

This quantization matrix can be multiplied by a scaling factor r that will affect the compression performance and the quality of the restored image after decompression. Let $T = r \times DCTQ$

The quantization step consists in replacing each element $x_{i,j}$ of an image block by

$$y_{i,j} = \text{round}\left(\frac{x_{i,j}}{T_{i,j}}\right) \tag{5}$$

2.3.1. Matlab exercise

Write a function that takes as argument a scaling factor r, the quantization matrix DCTQ and a block split DCT of an image to produce the quantized block split image.

3. JPEG decoding

3.1. RESCALING THE DATA BLOCKS

To recover the DCT (up to the quantization), we have to modify each block according to the formula

$$\boldsymbol{x_{i,j} = y_{i,j} \times T_{i,j}}$$

where T is the quantization matrix DCTQ multiplied by the scaling factor r.

3.1.1. Matlab exercise

Write a function to do the rescaling Do not forget to convert the input signal to double, as r is double. The output is double.

3.2. DCT BLOCK

The inverse DCT is computed with the same matrix C_N.The inverse DCT is given by

$$A = C_N^T \times \hat{A} \times C_N$$

3.2.1. Matlab exercise

Write a line of code that computes the inverse block DCT using the routine "blockDCT".

3.3. MERGING THE BLOCKS

We need now to merge the reconstructed split image into a conventional image. To do so, write a function that recombines the blocks. The output must be uint8.