

# DCT2D e IDCT2D NxN

- A DCT2D define-se como

$$F[u, v] = C[u]C[v] \sum_{m=0}^{N-1} x[m, n] \cos\left(\frac{(2m+1)u\pi}{2N}\right) \cos\left(\frac{(2n+1)v\pi}{2N}\right)$$

- A IDCT2D define-se como

$$f[m, n] = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C[u]C[v]F[u, v] \cos\left(\frac{(2m+1)u\pi}{2N}\right) \cos\left(\frac{(2n+1)v\pi}{2N}\right)$$

$$C[l] = \begin{cases} \frac{1}{\sqrt{N}}, & l = 0 \\ \sqrt{\frac{2}{N}}, & l \neq 0 \end{cases}$$

# DCT2D e IDCT2D 8x8

Utilizada no JPEG – codificação com perda, de imagem

## DCT2D 8x8

$$F[u, v] = \frac{1}{4} C[u] C[v] \sum_{m=0}^7 \sum_{n=0}^7 f[m, n] \cos\left(\frac{(2m+1)u\pi}{16}\right) \cos\left(\frac{(2n+1)v\pi}{16}\right)$$

## IDCT2D 8x8

$$f[m, n] = \frac{1}{4} \sum_{u=0}^7 \sum_{v=0}^7 C[u] C[v] F[u, v] \cos\left(\frac{(2m+1)u\pi}{16}\right) \cos\left(\frac{(2n+1)v\pi}{16}\right)$$

$$C[l] = \begin{cases} \frac{1}{\sqrt{2}}, & l = 0 \\ 1, & l \neq 0 \end{cases}$$



## DCT2D e IDCT2D: cálculo matricial

À semelhança do que se demonstra para a DFT2D tem-se a separabilidade da transformada:

$$Y = DCT2D[X] = CXCT^T,$$

sendo  $C$  o operador que calcula a DCT 1D.  $C$  é ortogonal:

$$C^T = C^{-1}, \quad CC^T = I, \quad C^TC = I.$$

Tem-se:

$$\begin{aligned} C^TYC &= C^T(CXC^T)C \\ &= (C^TC)X(CC^T) \\ &= X \end{aligned}$$

$$X = IDCT2D[Y] = C^TYC$$



## DCT2D: transformada separável

- Pode ser calculada à custa de duas DCT1D – é uma transformada separável
- Esta é a opção seguida na implementação em MATLAB



# Troço de código da função dct2

A função MATLAB **dct2** calcula a DCT2D, através de duas chamadas à **dct (1D)** que por sua vez utiliza a **fft (1D)**

```
function b=dct2(arg1,mrows,ncols)
% DCT2 Compute 2-D discrete cosine transform.
% B = DCT2(A) returns the discrete cosine transform of A.
% The matrix B is the same size as A and contains the
% discrete cosine transform coefficients.
% References:
%

[m, n] = size(arg1);
% Basic algorithm.
if ( nargin == 1 ),
    if ( m > 1 ) & ( n > 1 ),
        b = dct(dct(arg1).').';
        return;
    else
        (...)
```

# DCT1D: cálculo matricial

```
function dct_matrix(N)

Alpha      = zeros(N,N);
Alpha(1,:) = 1/sqrt(N);
Alpha(2:N,:) = sqrt(2/N);

[n,k] = meshgrid ( 0:1:N-1, 0:1:N-1 );
C = Alpha .* cos( (pi*(2.*n + 1).*k) / (2*N) );

% % ----- DCT1D -----
x = [1 2 3 4]';
y = C * x
Ex = sum ( x.* x )
Ey = sum ( y.* y )
% dct(x)
% % ----- DCT1D -----
.....
```

```
>> dct_matrix(4)
```

```
y =
```

```
5.0000
```

```
-2.2304
```

```
-0.0000
```

```
-0.1585
```

```
Ex = 30
```

```
Ey = 30.0000
```

```
>>
```

A DCT1D preserva a energia  
Transformada unitária

# DCT2D: cálculo matricial

```
% ----- DCT2D -----  
x = magic(N)  
y = C * x * C'  
% ----- IDCT2D -----  
z = C' * y * C  
  
Ex = sum(sum ( x.* x ))  
Ey = sum(sum ( y.* y ))  
% ----- DCT2D -----
```

- A DCT2D preserva a energia
- Geralmente, concentra a energia num número reduzido de coeficientes
- Transformada unitária



```
>> dct_matrix(4)  
x =  
    16     2     3    13  
     5    11    10     8  
     9     7     6    12  
     4    14    15     1  
  
y =  
 34.0000    0.0000   -0.0000   -0.0000  
  0.0000    0.0000   13.5140    0.0000  
 -0.0000    3.3785    0.0000    2.9302  
 -0.0000   -0.0000   11.7206    0.0000  
  
z =  
 16.0000    2.0000    3.0000   13.0000  
  5.0000   11.0000   10.0000    8.0000  
  9.0000    7.0000    6.0000   12.0000  
  4.0000   14.0000   15.0000    1.0000  
  
Ex = 1496  
Ey = 1496
```

# Cálculo da DCT2D

Imagem DC

$$x1 = \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix}$$

$$\text{dct2}(x1) = \begin{matrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

Variação (frequência)  
apenas em coluna

$$x2 = \begin{matrix} 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \end{matrix}$$

$$\text{dct2}(x2) = \begin{matrix} 0 & 1.53 & 0 & 3.69 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

# Cálculo da DCT2D

Variação (frequência)  
apenas em linha

$$x3 = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$dct2(x3) =$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -1.53 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -3.69 & 0 & 0 & 0 \end{bmatrix}$$

Variação (frequência)  
em linha e em coluna

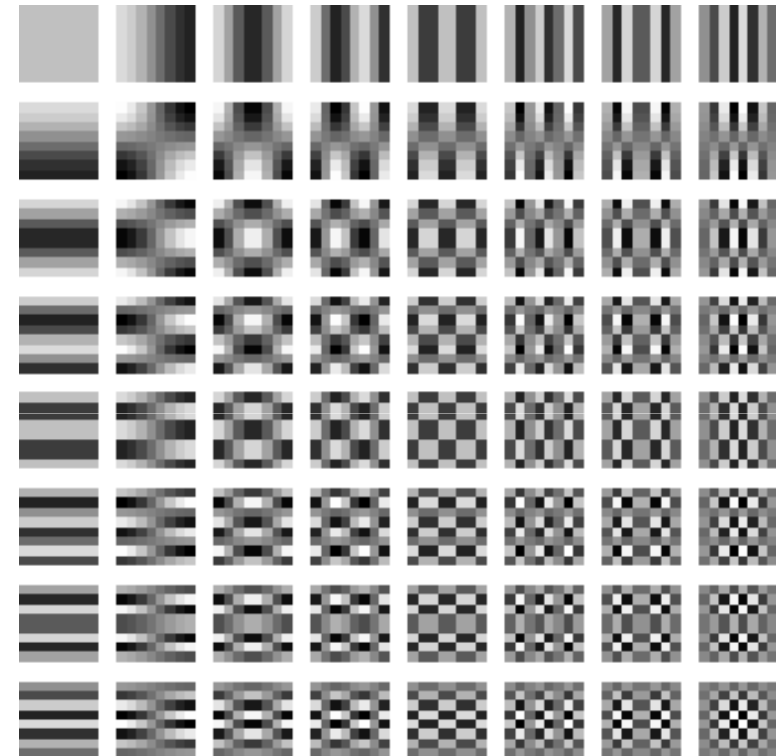
$$x4 = \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

$dct2(x4) =$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -0.58 & 0 & -1.41 \\ 0 & 0 & 0 & 0 \\ 0 & -1.41 & 0 & -3.41 \end{bmatrix}$$

# DCT 2D 8x8 – Imagens de base

```
pix=0;
N=8;
for v=0:N-1;
  for u=0:N-1;
    for y=0:N-1;
      for x=0:N-1;
        pix(v*(N+2)+1+y,u*(N+2)+1+x)=
          cos(((2*x+1)*u*pi)/(2*N))
            *cos(((2*y+1)*v*pi)/(2*N));
      end
    end
  end
end
mx=max(max(pix));mn=min(min(pix));
pix = pix-mn; pix = pix * 255 / (mx-mn);
colormap(gray); image(pix)
```



Qualquer imagem de resolução 8x8 pode ser obtida por combinação linear destas imagens de base. Os coeficientes da combinação linear são os coeficientes da DCT2D.

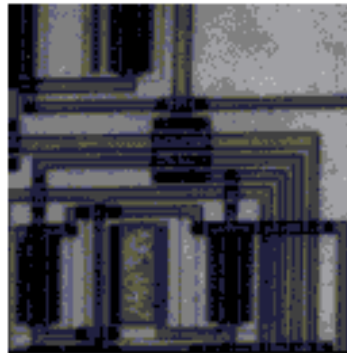
# DCT2D: concentração de energia

No MATLAB:

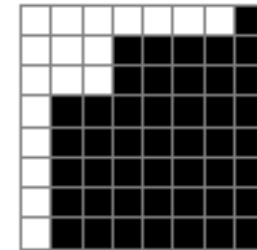
```
>> dctdemo
```

- Indicada para a codificação com perda
- Usada na norma JPEG

Original Circuit Image



DCT coefficients



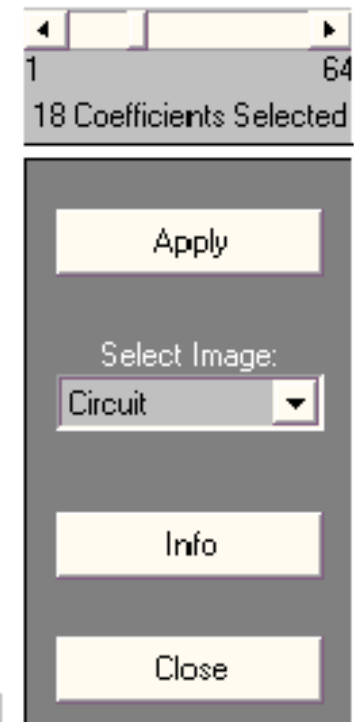
Reconstructed Image



Error Image



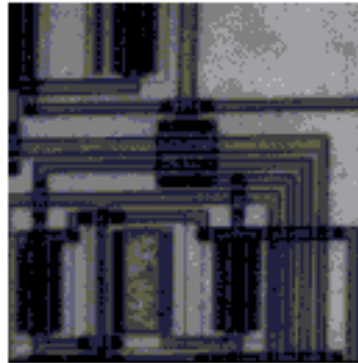
The MSE (with images normalized) is 0.000328 .



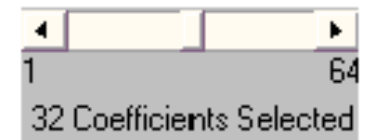
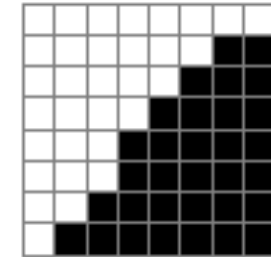
A MATLAB GUI window for the DCT2D demo. It features a slider at the top with values 1 and 64, and a label '18 Coefficients Selected'. Below the slider are three buttons: 'Apply', 'Info', and 'Close'. A 'Select Image:' dropdown menu is set to 'Circuit'.

# DCT2D: concentração de energia

Original Circuit Image



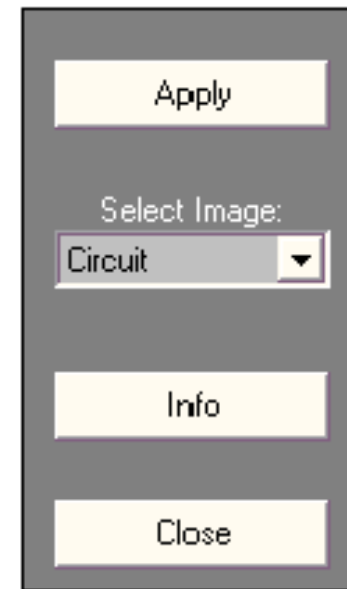
DCT coefficients



Reconstructed Image



Error Image



The MSE (with images normalized) is 0.000119.