MAC0317/MAC5920

Introdução ao Processamento de Sinais Digitais

Seção 3.7: Transformada DCT em blocos

Transformadas em Bloco 8×8

Dada uma matriz $A \in \mathcal{M}_{m,n}(\mathbb{C})$ onde m e n são múltiplos de 8, consideramos uma subdivisão em blocos 8×8, ou seja, definimos submatrizes $B^{(i,j)} \in \mathcal{M}_{8,8}(\mathbb{C})$ onde

$$A = \begin{pmatrix} B^{(0,0)} & B^{(0,1)} & \dots & B^{(0,q-1)} \\ B^{(1,0)} & B^{(1,1)} & \dots & B^{(1,q-1)} \\ \vdots & \vdots & \ddots & \vdots \\ B^{(p-1,0)} & \dots & \dots & B^{(p-1,q-1)} \end{pmatrix} \quad \text{e}$$

$$B^{(i,j)} = \begin{pmatrix} A_{8i,8j} & \dots & A_{8i,(8j+7)} \\ \vdots & \ddots & \vdots \\ A_{(8i+7),8j} & \dots & A_{(8i+7),(8j+7)} \end{pmatrix}$$

Em seguida definimos $\hat{B} = \mathrm{DCT\_BLK}(A)$ onde

$$\hat{B} = \begin{pmatrix} \hat{B}^{(0,0)} & \hat{B}^{(0,1)} & \dots & \hat{B}^{(0,q-1)} \\ \hat{B}^{(1,0)} & \hat{B}^{(1,1)} & \dots & \hat{B}^{(1,q-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{B}^{(p-1,0)} & \dots & \dots & \hat{B}^{(p-1,q-1)} \end{pmatrix}$$

e cada $\hat{B}^{(i,j)} = DCT(B^{(i,j)})$, que também é um bloco de 8×8.

In [11]:

```
# calcula a DCT 2D fazendo a DCT 1D das colunas e depois das linhas
def dct_2d(m):
    D1 = spfft.dct(m.T, norm='ortho')
    D2 = spfft.dct(D1.T, norm='ortho')
    return D2
# calcula a DCT da imagem
N = dct_2d(M)
# mapeia os valores para log
Nlog = np.log(1 + abs(N))
```

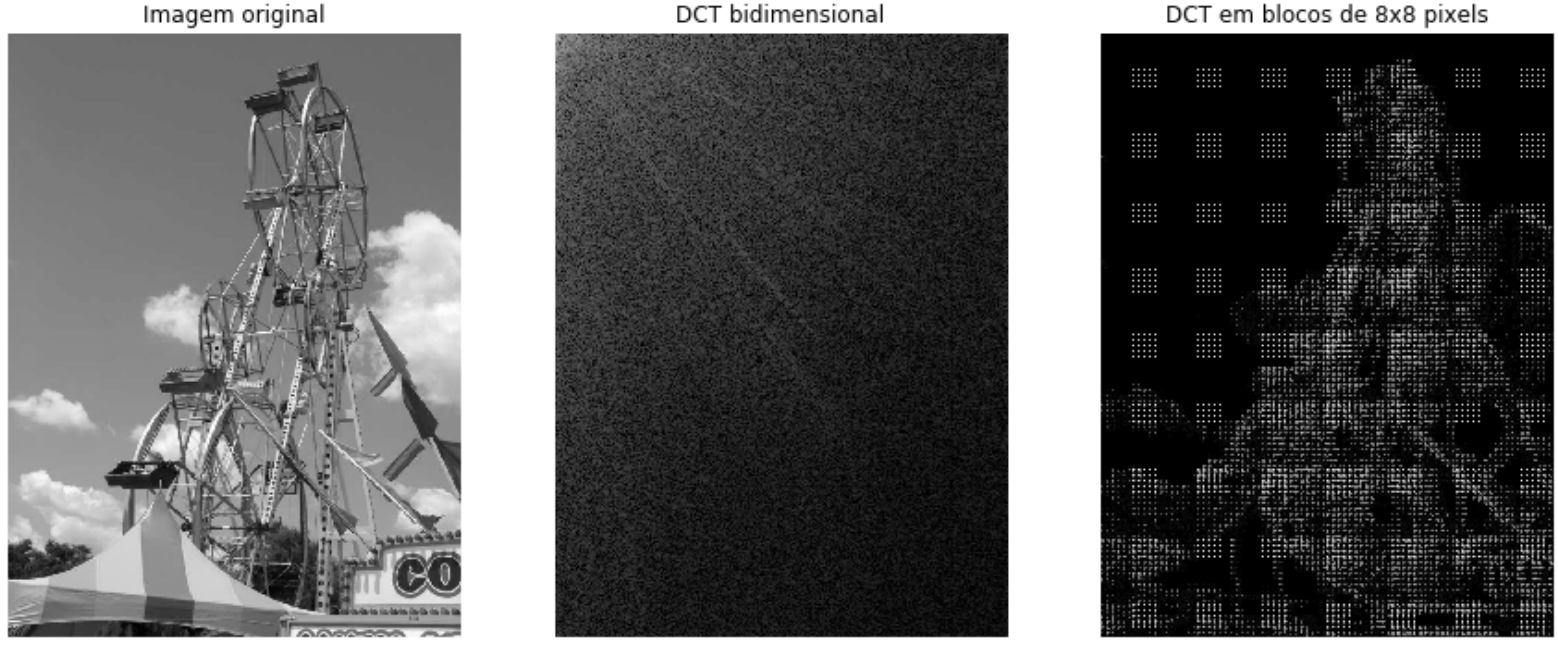
In [12]:

```
# calcula a DCT por blocos de 8x8
NBAK = np.empty(N.shape)

for j in range(0, My, divsize):
    for k in range(0, Mx, divsize):
        Mdiv = M[j : j+divsize, k : k+divsize]
        Ndiv = dct_2d(Mdiv)
        NBAK[j : j+divsize, k : k+divsize] = Ndiv

NBAKlog = np.log(1 + abs(NBAK))
```

In [13]:

```
fig, ax = plt.subplots(1,3, figsize=(15,6))
ax[0].imshow(M, cmap='gray');ax[0].axis("off")
ax[0].set_title("Imagem original")
ax[1].imshow(Nlog / np.amax(Nlog), cmap='gray');ax[1].axis("off")
ax[1].set_title("DCT bidimensional")
ax[2].imshow(NBAKlog / np.amax(Nlog), cmap='gray');ax[2].axis("off")
ax[2].set_title("DCT em blocos de 8x8 pixels")
fig.suptitle("Figura 3.12: Imagem original, DCT e DCT em blocos de 8x8 pixels");
plt.show()
```

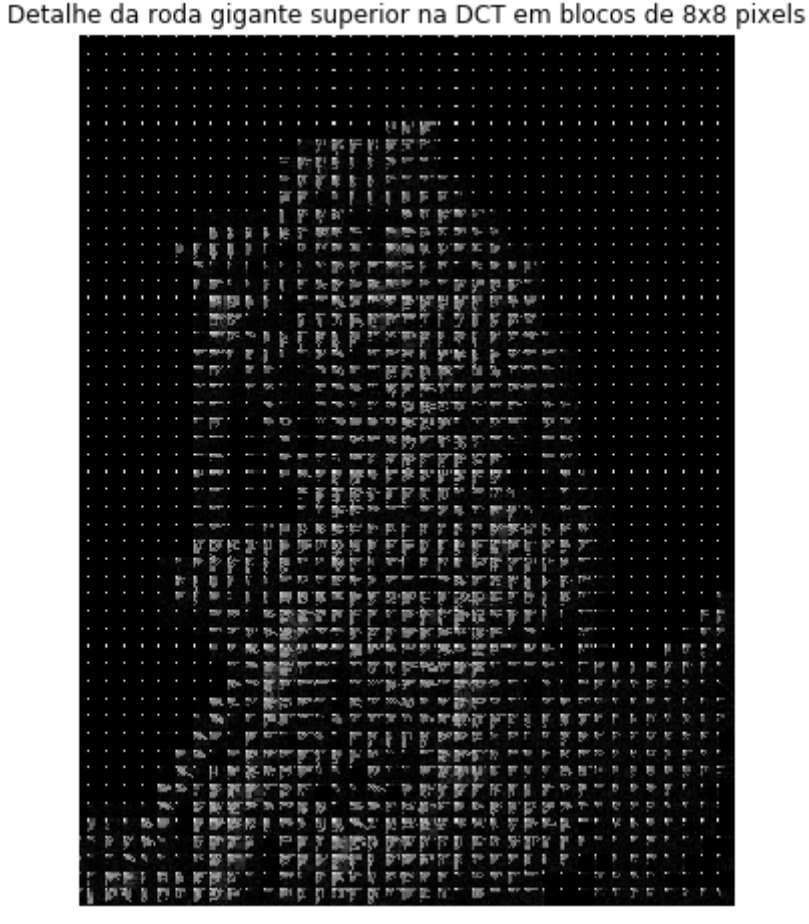
Figura 3.12: Imagem original, DCT e DCT em blocos de 8x8 pixels



Observação: Na DCT em blocos é possível perceber uma espécie de silhueta da figura original: isso se deve principalmente porque cada bloco 8×8 contém as componentes dc, que refletem o nível de energia do trecho correspondente da imagem.

In [14]:

```
plt.figure(figsize=(10,8));plt.axis("off")
plt.imshow(np.log(1+abs(NBAK[0:400,300:600]))/np.amax(Nlog), cmap='gray')
plt.title("Detalhe da roda gigante superior na DCT em blocos de 8x8 pixels")
plt.show()
```



DCT em blocos agrupados por frequência

Outra organização possível da DCT em blocos é agrupar os coeficientes de todos os blocos por pares de frequência $k, l = 0, 1, \ldots, 7$. Nesse caso, a matriz resultante possui 64 blocos $\tilde{B}^{(k,l)}$ de tamanho $\frac{m}{8} \times \frac{n}{8}$:

$$\tilde{B} = \begin{pmatrix} \tilde{B}^{(0,0)} & \tilde{B}^{(0,1)} & \cdots & \tilde{B}^{(0,7)} \\ \tilde{B}^{(1,0)} & \tilde{B}^{(1,1)} & \cdots & \tilde{B}^{(1,7)} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{B}^{(7,0)} & \cdots & \cdots & \tilde{B}^{(7,7)} \end{pmatrix}$$

onde

$$\left(\tilde{B}^{(k,l)}\right)_{i,j} = \left(\hat{B}^{(i,j)}\right)_{k,l}.$$

In [16]:

```
fig, ax = plt.subplots(1,2, figsize=(10,6))
ax[0].imshow(NOVOlog / np.amax(Nlog), cmap='gray')
ax[0].set_title("DCT em Blocos Reagrupados por frequência")
ax[1].imshow(np.log(1 + abs( NOVO[0:Ny//divsize, 0:Nx//divsize] )) / np.amax(Nl
og), cmap='gray')
ax[1].set_title("Detalhe da componente DC ($k=l=0$)")
fig.suptitle("Figura 3.13: DCT em blocos reagrupados por frequência e Componente
s DC")
plt.show()
```

Figura 3.13: DCT em blocos reagrupados por frequência e Componentes DC