CENTRO UNIVERSITÁRIO IESB
Centro Universitário Instituto de Educação Superior de Brasília

Curso: Engenharia da Computação

Disciplina: Teoria da Informação e Codificação

Professor: Thiago Raposo Milhomem de Carvalho

Aluno: Diego Vieira Santos

Matrícula: 2012082035

Introdução

Nesta atividade iremos abordar o uso da transformada cosseno discreta (DCT) e como ela é utilizada para compressão de imagens com perda.

Discrete Cosine Transform 2D

A transformada discreta de cosseno (DCT) está bem relacionada com a transformada discreta de Fourier (DFT) porém ela usa apenas números reais. Ela é uma transformação linear que é separável, ou seja, na transformação em duas dimensões temos uma transformação DCT unidimensional performando ao lado de outra transformação DCT unidimensional em outra dimensão. A definição da DCT de duas dimensões para uma imagem A de entrada e uma imagem B de saída é:

$B_{pq} = a_p a_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} A_{mn} \cos\left(\frac{(\pi(2m+1)p)}{2M}\right) \cos\left(\frac{(\pi(2n+1)q)}{2N}\right)$, sendo que $0 \leq p \leq M-1$ e $0 \leq q \leq N-1$

Onde:

$$a_p = \left\{\frac{1}{\sqrt{M}}, p = 0\right\} \text{ e } \left\{\sqrt{\frac{2}{M}}, 1 \leq p \leq M-1\right\}$$

e

$$a_q = \left\{\frac{1}{\sqrt{N}}, q = 0\right\} \text{ e } \left\{\sqrt{\frac{2}{N}}, 1 \leq q \leq N-1\right\}$$

Sendo que M e N são as linhas e colunas da imagem A, respectivamente.

Ela é bastante utilizada na ciência e na engenharia para fazer compressões com perda, tanto de áudio, quanto de imagem, para o caso da compressão de imagens, ela muda o domínio da imagem para o domínio da frequência, como quase todas as imagens são compostas por informações de baixa frequência, essa transformada descarta os componentes de alta frequência, desta forma comprime a imagem.

Explicação do experimento

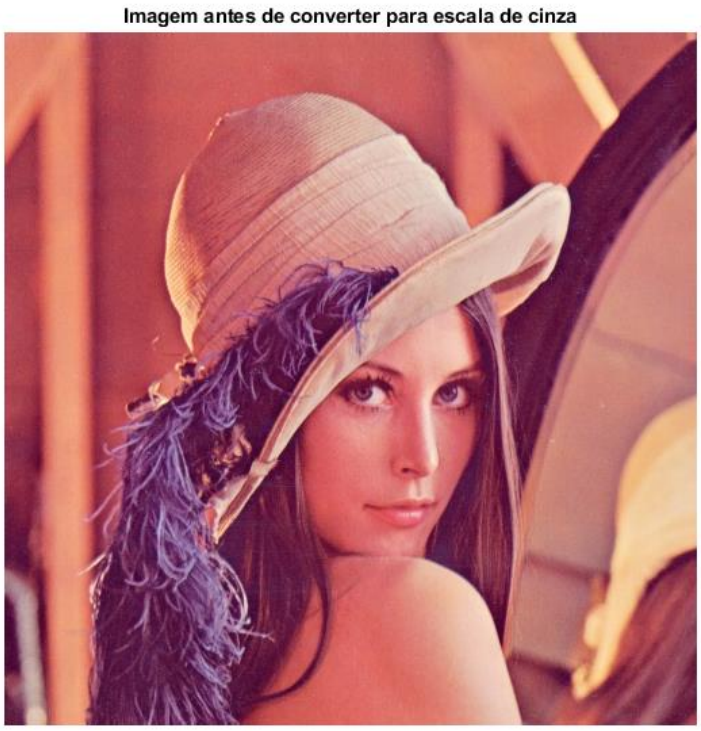
Este projeto foi feito inteiramente utilizando Matlab, através dele fizemos a compressão de duas imagens, a primeira sendo para testar o funcionamento da transformada discreta de cosseno e sua utilização para compressão de imagens e a segunda imagem foi utilizada para saber como a DCT se comporta para valores diferentes de taxa de compressão.

Primeiramente, lemos a imagem e convertemos para a escala de cinza para que o algoritmo de compressão funcione sem erros e também para que fique mais nítido a visualização da recuperação desta imagem, depois utilizamos a transformada discreta do cosseno para obtermos a imagem no domínio da frequência, com isso definimos uma taxa de compressão e fazemos e utilizamos a transformada discreta do cosseno inversa em duas dimensões (IDCT2) para ter a imagem recuperada e comprimida.

Explicação do funcionamento das funções

Primeiro lemos a imagem através do comando:

```
X = imread("Lenna.jpg");
imshow(X);title("Imagem antes de converter para escala de cinza");
```



Agora convertemos a imagem para escala de cinza

```
X = double(rgb2gray(X))/255;
imshow(X);title("Imagem após conversão para escala de cinza");
```

Procedimentos para comprimir imagem

Aplicamos a transformada discreta para termos a imagem no domínio da frequência

```
X_dct = dct2(X);
```

Criamos uma matriz de zeros com o mesmo tamanho da imagem original para armazenar a imagem transformada

```
X_dct_modif = zeros(size(X_dct));
```

Definimos a quantidade de linhas e colunas

```
[L,C] = size(X_dct);
```

Agora temos uma variável que interfere no resultado do fator de compressão que será mostrado posteriormente, para esse caso foi dado um valor pelo roteiro

```
partial_dim = 0.05;
```

Quando utilizamos valores variados de compressão foi preciso utilizar uma fórmula para encontrar o valor de compressão que foi solicitado, essa fórmula é dada por

$$P_D = \sqrt{1 - C_t}$$

Em que P_D é equivalente a uma porcentagem das dimensões originais, e C_t é a taxa de compressão que desejamos encontrar para essa porcentagem de dimensões.

Agora definimos valores de linhas e colunas que equivalem a essa redução utilizando P_D

```
L_final = round(partial_dim*L) ; C_final = round(partial_dim*C);
```

Agora criamos uma matriz utilizando essas dimensões para definir os coeficientes que serão mantidos

```
X_dct_modif(1:L_final,1:C_final) = X_dct(1:L_final,1:C_final);
```

Procedimentos para recuperar imagem

Definição da quantidade de coeficientes que será usada para calcular o fator de compressão

```
qtd_coeficientes = L_final*C_final;
```

Cálculo do fator de compressão, foi utilizado apenas para conferir se a imagem está sendo comprimida pela taxa desejada, nesse caso a taxa foi de

```
fator_compressao = 1 - qtd_coeficientes/(L*C);
```

Aplicação da transformada inversa discreta do cosseno para recuperar a imagem

```
X_modif = idct2(X_dct_modif);
```

Mostrando a imagem recuperada após a compressão

```
imshow(X_modif);title("Imagem recuperada após compressão");
```

Após a compressão com taxa de 99%, na recuperação foi possível notar que a imagem está bem borrada, mas caso se saiba qual é a imagem original então será possível encontrar semelhança.

Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

Primeiro definimos a quantidade de linhas e colunas para poder fazer o calculo

```
[L,C] = size(X);
```

Agora fazemos um procedimento equivalente a seguinte expressão:

$$\mathrm{MSE} = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{(X(i,j) - Y(i,j))^2}{(X(i,j))^2}$$

Em que X é a imagem original e X_modif é a imagem comprimida e recuperada

```
num = 0;den = 0;
for i = 1 : L
    for j = 1 : C
        num = num + (X(i,j)-X_modif(i,j))^2;
        den = den + (X(i,j))^2;
    end
end
MSE = num / den
```

```
MSE = 0.0245
```

Agora fazemos um processo equivalente a seguinte expressão:

$$\mathrm{PSNR_{db}} = 10\log_{10}\left(\frac{65025}{\mathrm{MSE}}\right)$$

```
PSNR_db = 10*log10(65025/MSE)
```

```
PSNR_db = 64.2322
```

Agora iremos limpar o workspace para fazer o uso das técnicas em outra imagem com várias taxas de compressão

```
clear;
```

Compressão de imagens com perdas através de transformada

Carregando imagem

```
X = imread("img.jpg");
imshow(X);title("Imagem antes de converter para escala de cinza");
```

Convertendo para escala de cinza

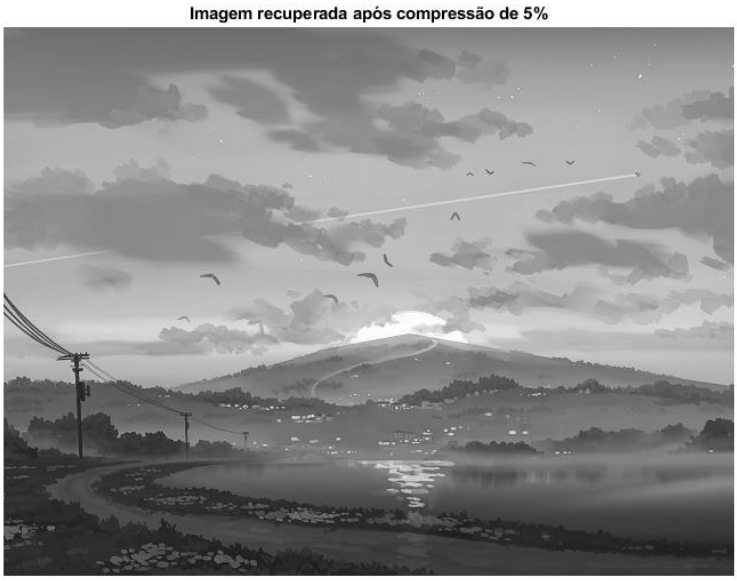
```
X = double(rgb2gray(X))/255;
imshow(X);title("Imagem após conversão para escala de cinza");
```

Criando vetores para gerar gráfico de PSNR versus taxa de compressão

```
vet_psnr = zeros(1,9);
vet_taxa = [.05 .1 .15 .2 .5 .8 .85 .9 .95];
pd = sqrt(1-vet_taxa);
```

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 5%

```
partial_dim = pd(1);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 5%");
```



Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

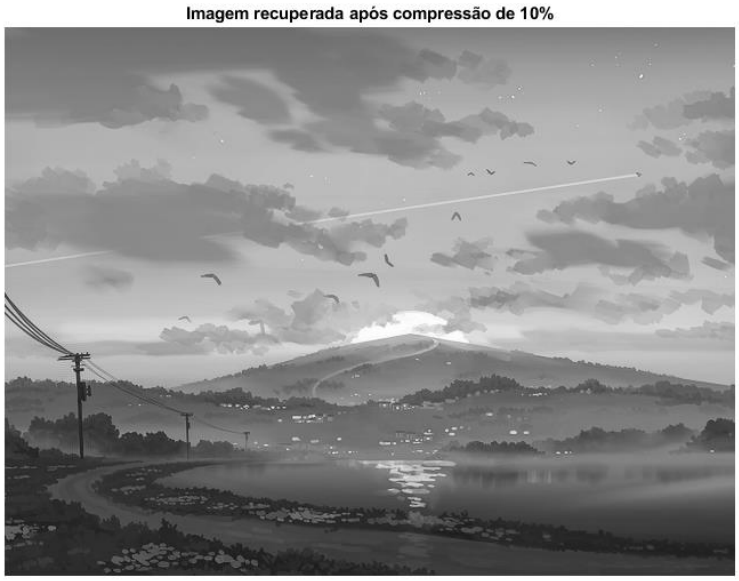
```
vet_psnr(1) = PSNR_db;
```

Avaliação subjetiva

Com essa taxa de compressão é quase que imperceptível a mudança entre a imagem recuperada e a original

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 10%

```
partial_dim = pd(2);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 10%");
```



Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

```
vet_psnr(2) = PSNR_db;
```

Avaliação subjetiva

Ainda está com uma taxa muito baixa, desta forma é imperceptível a diferença entre a imagem original e a imagem recuperada

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 15%

```
partial_dim = pd(3);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 15%");
```

Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

```
vet_psnr(3) = PSNR_db;
```

Avaliação subjetiva

Dessa imagem para a anterior é imperceptível ver diferença, talvez o fato da imagem ter uma resolução de 640x480 esteja relacionado a esse fato.

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 20%

```
partial_dim = pd(4);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 20%");
```

Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

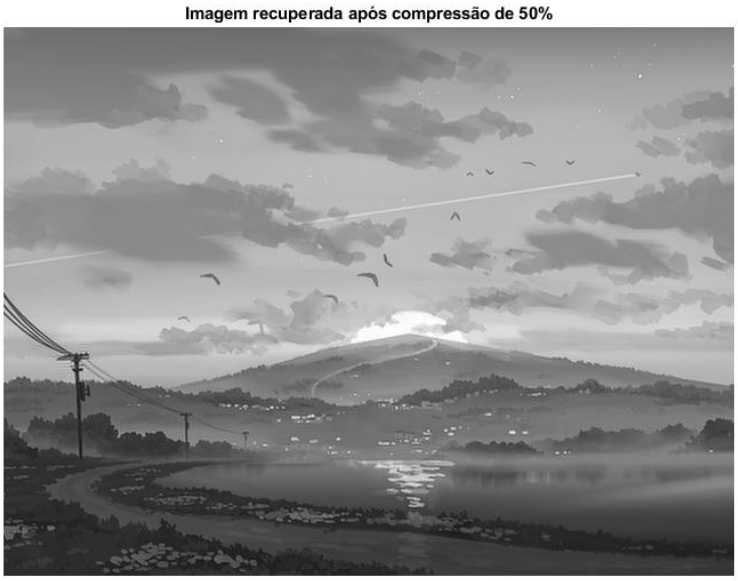
```
vet_psnr(4) = PSNR_db;
```

Avaliação subjetiva

Com essa taxa também é imperceptível a mudança pois a forma de recuperação é bem precisa.

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 50%

```
partial_dim = pd(5);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 50%");
```



Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

```
vet_psnr(5) = PSNR_db;
```

Avaliação subjetiva

Com essa taxa também já é possível a começar a perceber aliasing, mas é bem sutil e tem que olhar bem de perto.

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 80%

```
partial_dim = pd(6);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 80%");
```

Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

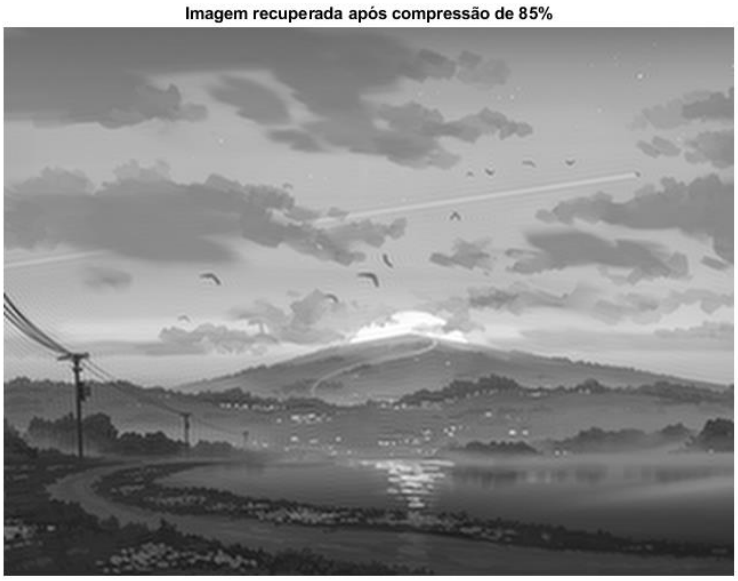
```
vet_psnr(6) = PSNR_db;
```

Avaliação subjetiva

Agora com essa taxa de compressão já é bem notória a diferença, já é possível ver bastante aliasing

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 85%

```
partial_dim = pd(7);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 85%");
```



Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

```
vet_psnr(7) = PSNR_db;
```

Avaliação subjetiva

Com essa taxa a imagem já está começando a embaçar e ficar com bastante aliasing mas ainda é possível ver semelhança entre ela e a original.

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 90%

```
partial_dim = pd(8);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 90%");
```

Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

```
vet_psnr(8) = PSNR_db;
```

Avaliação subjetiva

Com essa taxa a imagem já está bem embaçada, mas ainda é possível perceber a semelhança com a original.

Procedimentos para comprimir e recuperar imagem com taxa de compressão de 95%

```
partial_dim = pd(9);
X_modif = comp_and_rec(X,partial_dim);
title("Imagem recuperada após compressão de 95%");
```

Relação sinal ruído de pico da imagem

Valores de erro quadrático médio e relação sinal ruído de pico, respectivamente:

```
[MSE, PSNR_db] = rel_sinal_ruido(X,X_modif);
```

Salvando valores para fazer gráfico de relação sinal-ruído de pico versus taxa de compressão

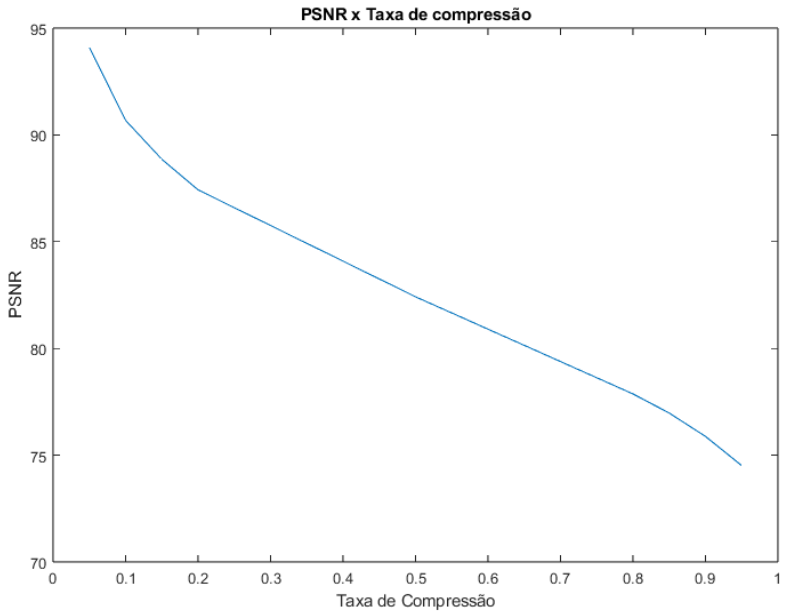
```
vet_psnr(9) = PSNR_db;
```

Avaliação subjetiva

Com essa taxa a imagem já está bem embaçada, fazendo com que se torne bem diferente da original, porém ainda dá pra notar alguns traços de semelhança.

Plotando gráfico de PSNR versus Taxa de Compressão

```
 plot(vet_taxa,vet_psnr);title("PSNR x Taxa de compressão");xlabel("Taxa de Compressão");ylabel("PSNR");
```



Avaliação subjetiva sobre o gráfico

Com esse gráfico foi possível notar que os pontos onde se há mais diferença nesta relação são nas extremidades, ou seja, caso a taxa de compressão seja muito pequena ou muito grande a curva é maior, tanto que entre 20% e 80% o gráfico é quase linear.

Conclusões gerais

O trabalho realizado foi bem interessante e satisfatório, com ele foi possível colocar em prática o estudo de processamento de imagens e foi possível entender melhor o funcionamento de algumas funções do Matlab e conhecer algumas funções novas.

Cálculo da relação sinal-ruído

```
function [MSE,PSNR_db] = rel_sinal_ruido(X,Y)
    [L,C] = size(X);
    num = 0;
    den = 0;
    for i = 1 : L
        for j = 1 : C
            num = num + (X(i,j)-Y(i,j))^2;
            den = den + (X(i,j))^2;
        end
    end
    MSE = num / den;
    PSNR_db = 10*log10(65025/MSE);
end
```

Procedimento para comprimir e recuperar imagem

```
function X_modif = comp_and_rec(X,partial_dim)
    X_dct = dct2(X);
    X_dct_modif = zeros(size(X_dct));
    [L,C] = size(X_dct);
    L_final = round(partial_dim*L) ; C_final = round(partial_dim*C);
    X_dct_modif(1:L_final,1:C_final) = X_dct(1:L_final,1:C_final);
    qtd_coeficientes = L_final*C_final;
    fator_compressao = 1 - qtd_coeficientes/(L*C);
    X_modif = idct2(X_dct_modif);
    imshow(X_modif);
end
```