Instituto Superior de Engenharia de Lisboa
Licenciatura em Engenharia Informática e Multimédia
Codificação de Sinais Multimédia

2º Semestre de 2018/2019

O ficheiro com o relatório do trabalho (em formato PDF) e os ficheiros de código implementado devem ser encapsulados num ficheiro ZIP com o número do grupo e submetidos no Moodle até 19 de Maio.

Este trabalho explora os princípios básicos da norma JPEG (modo sequencial) para compressão de imagens com perdas. Neste trabalho apenas serão exploradas as imagens em níveis de cinzento.

1. A biblioteca URLlib tem um conjunto de funções que permite aceder a conteúdos que estejam alojados em servidores. Existem vários servidores com imagens de teste, por exemplo
   https://www.ece.rice.edu/˜ wakin/images/
   ou
   https://homepages.cae.wisc.edu/˜ ece533/images/
   Faça download do ficheiro da lena através do seguinte código:

   ```
   import urllib
   urllib.request.urlretrieve("https://homepages.cae.wisc.edu/~ece533/images/lena.bmp","lena.bmp")
   ```

2. A biblioteca fftpack contém as transfomação de coseno (DCT) e coseno inversa (IDCT). Pode com estas construir a DCT2D e IDCT2D, como ilustrado abaixo.

   ```
   from scipy.fftpack import dct, idct
   X_kl = dct(dct(x_mn.T, norm='ortho').T ,  norm='ortho')
   x_mn = idct(idct(X_kl.T, norm='ortho').T ,  norm='ortho')
   ```

   Visualize cada uma das funções de base para a DCT2D, admitindo blocos de 8×8.

3. Construa uma função (codificador) que para cada bloco de 8×8 da imagem original efectue a DCT bidimensional. Use por exemplo a instrução. Veja a imagem com o conjunto dos blocos após a transformada.
   Construa uma função (descodificador) que faça a DCT inversa.
   Verifique que a imagem é igual à original.

4. Construa uma função (codificador) que para cada bloco de 8×8 de coeficientes da transformação efectuada faça a divisão pela matriz de quantificação (tabela K1 no anexo da norma) multiplicada por um factor de qualidade q (ver pág. 230 do livro "Tecnologias de Compressão Multimédia").
   Veja a imagem com o conjunto dos blocos após a quantificação.
   Construa uma função (descodificador) que realize a operação inversa da quantificação.
   Junte estas funções às já realizadas e verifique para diferentes factores de qualidade qual a SNR e veja a imagem descodificada.

5. Construa uma função (codificador) que faça a codificação diferencial dos coeficientes DC após a quantificação.
   Construa a função inversa para o descodificador.

6. Construa uma função (codificador) que faça crie um array com a indexação em zig-zag dos coeficientes AC após a quantificação e crie um array com os pares (zero run length, nonzero value).
   Construa a função inversa para o descodificador.

7. Junte estas funções às já realizadas e veja a imagem descodificada.

8. Construa uma função que dados os arrays das alíneas anteriores use as tabelas do código de Huffman (tabela K3 e K5) e grave num ficheiro a sequência de bits correspondente. (não é necessário usar o formato JFIF)

9. Construa uma função que leia o ficheiro gravado e retorne os arrays com os coeficientes AC e DC.

10. Junte estas funções às já realizadas e veja a imagem descodificada.
    Para diferentes factores de qualidade meça a relação sinal-ruído e a taxa de compressão obtida. Represente um gráfico onde se apresente a taxa de compressão em função do SNR.

11. No mesmo gráfico compare o seu compressor de imagem com outros existentes para várias qualidades, por exemplo use:

```
from PIL import Image
x = Image.open("lena.tiff")
x.save("lena_c.jpeg", "JPEG", quality=50)
```

12. O relatório deve conter uma descrição breve das funções realizadas e uma tabela com todos os resultados da SNR, taxa de compressão, tempo de compressão e descompressão.