

Projeto

August 22, 2021

Compressão de Imagens em JPEG e Grayscale Através da Segmentação e da Transformada Discreta do Cosseno de Fourier

August 22, 2021

1 Compressão de Imagens

[1]

1.1 O Formato JPEG

1.2 A Transformada do Cosseno Discreta de Fourier (DCT)

1.3 Usando DCT na Compressão de Imagens JPEG em Grayscale

2 Setup Inicial

2.1 Bibliotecas Usadas no Projeto

```
[19]: import numpy as np
import matplotlib.pyplot as plt
#Funções que implementam a transformada discreta do cosseno de fourier
from scipy.fftpack import dct, idct
import cv2 # Biblioteca usada para importar imagens
from math import floor
from IPython.display import Latex
```

2.2 Funções Auxiliares

O módulo `scipy.fftpack` fornece funções apenas para transformadas de uma dimensão, a função abaixo converte uma função de transformada discreta de fourier unidimensional para bidimensional.

```
[20]: def transform2D(img,function):
    return np.transpose(function(np.transpose(function(img,norm = "ortho"),norm = "ortho")))
```

Função responsável por segmentar uma imagem e aplicar uma função sobre os segmentos de tamanho `size X size` da imagem.

```
[21]: def segmentation(img,size,function=transform2D,function2 =dct):
    copy = np.zeros(img.shape)

    for i in range(copy.shape[0]//size):
```

```

        for j in range(copy.shape[1]//size):
            copy[i*size:(i+1)*size,j*size:(j+1)*size] = function(img[i*size:
↪ (i+1)*size,j*size:(j+1)*size],function2)

    return copy

```

A função abaixo é uma função auxiliar que é utilizada para o melhor entendimento da função **thresholdingPC**. Ela retorna o index que representa **n%** da matriz que representa a imagem quando esta é convertida para um array unidimensional.

```

[22]: def percent(img,n):
        return floor((img.shape[0]*img.shape[1])*(n/100))

```

Função responsável por filtrar através do método thresholding de segmentação, a imagem no domínio da frequência. Esta função iguala a zero os valores de frequência que estão abaixo de um certo limiar, que é calculado através do enésimo valor de maior magnitude da matriz que representa a imagem.

```

[23]: def thresholdingPC(img,pc):
        imgCopy = img.copy()
        sortedCts = np.sort(abs(imgCopy.ravel()))
        threshold = sortedCts[-percent(imgCopy,pc)]
        imgCopy[abs(imgCopy) < threshold] = 0;
        return imgCopy

```

Função auxiliar para a função **thresholdingMean**. Ela recebe um segmento de uma imagem **img** que já passou pela transformada de fourier, e baseado no critério dado pela função **function** calcula o limiar (threshold) para filtrar os coeficientes redundantes e igualá-los a zero.

```

[24]: def filterImg(img,function):
        imgCopy = img.copy();
        threshold = function(np.abs(imgCopy))
        #print(threshold)
        imgCopy[abs(imgCopy) < threshold ] = 0
        return imgCopy

```

Função que filtra os segmentos da imagem, aplicando a média dos valores da matriz de tamanho **size X size** como método estatístico para calcular o limiar (threshold)

```




[25]: def thresholdingMean(img,size):
        imgCopy = img.copy()
        imgCopy = segmentation(imgCopy,size,filterImg,np.mean)
        return imgCopy;

```

Função que é um compilado das funções acima. Ela basicamente recebe uma imagem JPEG e algumas informações adicionais e executa a compressão desta. O passo a passo de como ela funciona será explicado com mais detalhes futuramente.

```
[26]: def compress(img, thresholding = thresholdingMean, argThres = 8, seg = 8, save = False, path = "imgCMP.jpg"):
    compImg = img.copy()
    compImg = segmentation(compImg, seg)
    compImg = thresholding(compImg, argThres)
    compImg = segmentation(compImg, seg, function2 = idct)
    if(save):
        cv2.imwrite(path, compImg)
    return compImg
```

3 Compressão de Imagens Na Prática

```
[27]: path = "../BANCO-DE-IMAGENS/img6.jpg"
imgBGR = cv2.imread(path)
imgRGB = cv2.cvtColor(imgBGR, cv2.COLOR_BGR2RGB)
imgGRAY = cv2.cvtColor(imgBGR, cv2.COLOR_BGR2GRAY)
img = np.mean(img, -1)
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # Converte a imagem para grayscale
 (Preto e branco)
```

```
[28]: plt.imshow(imgRGB)
cv2.imwrite("generated-files/imgBGR.jpg", imgBGR)
```

[28]: True



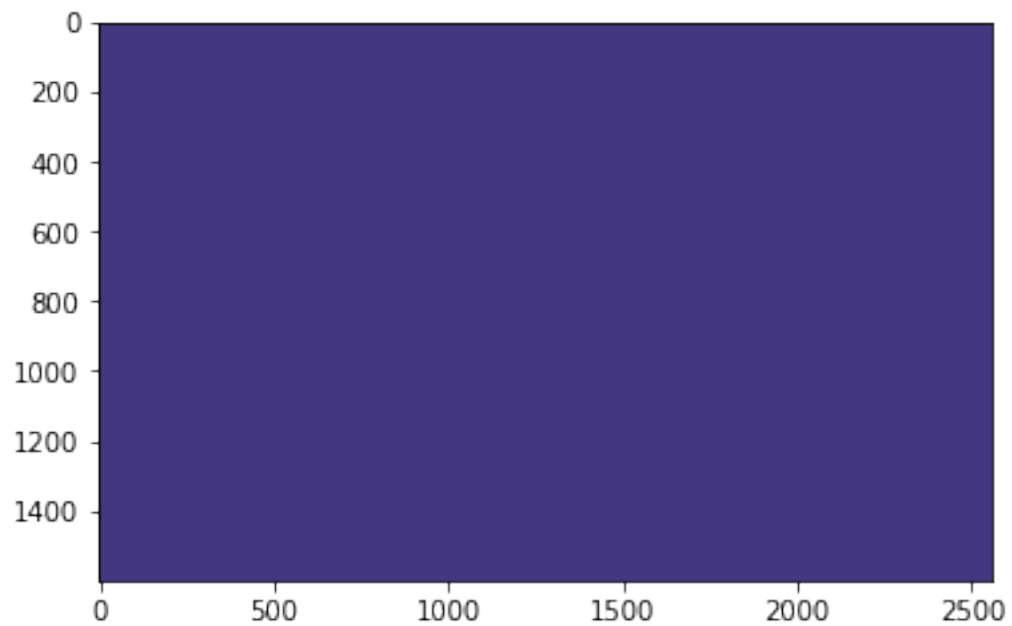
```
[29]: plt.imshow(imgGRAY,cmap = "gray")
      cv2.imwrite("generated-files/imgGRAY.jpg",imgGRAY)
```

[29]: True



```
[30]: img_dct = transform2D(imgGRAY,dct)
      plt.imshow(img_dct)
```

[30]: <matplotlib.image.AxesImage at 0x7ff5aa67abe0>



```
[31]: img_idct = transform2D(img_dct,idct)
      plt.imshow(img_idct,cmap = "gray")
```

```
[31]: <matplotlib.image.AxesImage at 0x7ff5aa6a7880>
```



```
[32]: imgCMP = segmentation(imgGRAY,8)
imgCMP = thresholdingMean(imgCMP,8)
#np.savetxt('img2.out', img2)
imgCMP = segmentation(imgCMP,8,function2 = idct)
#np.savetxt('img3.out', img3)
```

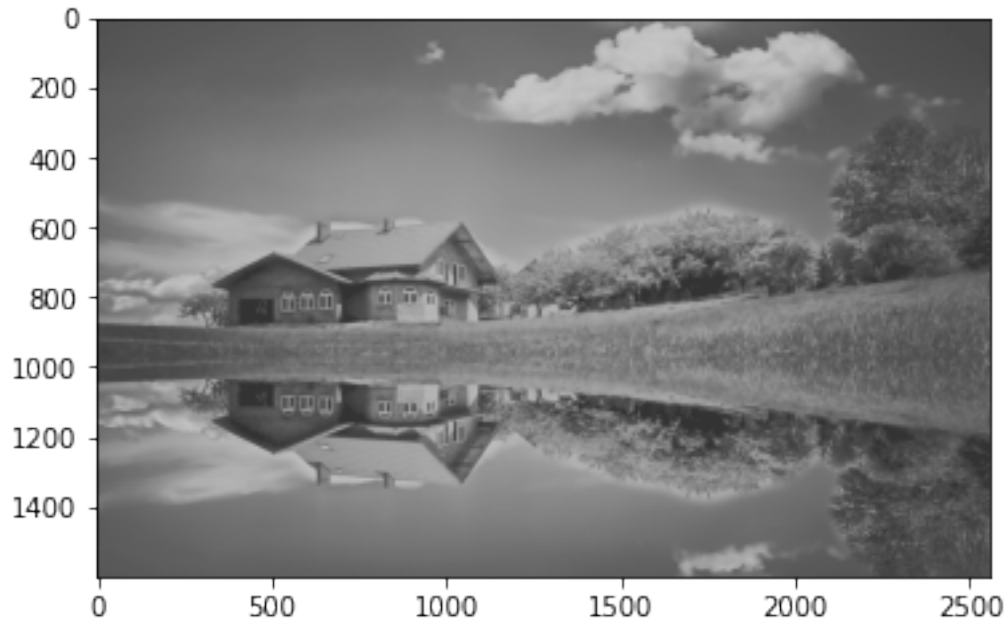
```
[33]: plt.imshow(imgCMP, cmap = "gray")
cv2.imwrite('generated-files/imgCMP.jpg',imgCMP)
```

[33]: True



```
[34]: test = compress(imgGRAY,thresholding = thresholdingPC, argThres = 2,save = True)
plt.imshow(test, cmap = "gray")
```

[34]: <matplotlib.image.AxesImage at 0x7ff5aa9435e0>



References

- [1] M. Kanaka Reddy , V. V. Haragopal and S. A. Jyothi Rani "Statistical Image Compression using Fast Fourier Coefficients" , Dec. 2016
- [2] J.Feydy "Part 6: Fourier analysis and JPEG compression", Feb.2019. [Online]. Available: http://www.jeanfeydy.com/Teaching/MasterClass_Radiologie/Part%20%20-%20JPEG%20compression.html
- [3] S. W. Smith, "The Scientist and Engineer's Guide to Digital Signal Processing", [Online]. Available: <http://www.dspguide.com/ch27/6.htm>
- [4] J. F. Neto, "Compressão Sem Perdas de Imagens Digitais", [Online]. Available: <http://www.dpi.inpe.br/~carlos/Academicos/Cursos/Pdi/SemPerdas.htm>
- [5] O. Hampiholi, "Image Compression — DCT Method", Mar.21. [Online]. Available: <https://towardsdatascience.com/image-compression-dct-method-f2bb79419587>
- [6] Johnsy,A. "2-D Discrete Cosine Transform", [Online]. Available: <https://www.imageprocessing.com/2013/03/2-d-discrete-cosine-transform.html>
- [7] S.Thayammal and D.Selvathi "A Review On Segmentation Based Image Compression Techniques", 2013