scc0251/5830— Prof. Moacir Ponti

PAE/monitores: Fernando Pereira dos Santos, Eduardo Carlos Souza

Trabalho 5: inpainting usando FFTs

Desenvolva seu código sem olhar o de colegas. Plágio não será tolerado.

Nesse trabalho você deverá implementar um método de inpanting[1], que é o processo de reconstruir partes deterioradas de imagens e vídeos.

Você deverá implementar usando `python 3` com as bibliotecas `numpy` e `imageio`.

Leia com calma as instruções de cada etapa.

Seu programa deverá permitir receber o nome base de três imagens: imagem original, imagem deteriorada, e máscara.

1. **Receber via teclado**:
   a) nome do arquivo da imagem original i: `imgo`;
   b) nome do arquivo da imagem deteriorada g: `imgi`;
   c) nome do arquivo da imagem com a máscara para realizar inpainting m: `imgm`;
   d) número de iterações para executar, T.

As imagens serão lidas no formato png.

Algoritmo Gerchberg-Papoulis com filtragem espacial

O algoritmo Gerchberg-Papoulis é utilizado amplamente em processamento de sinais e imagens, e parte do resultado de que sinais com banda limitada (ou seja, para os quais há um limite finito de frequências com amplitudes positivas), possui extensão temporal infinita. Assim, zeramos parte das frequências da imagem, e ao realizar a inversa, obtemos uma imagem na qual pixels de valor zero são extrapolados.

Considerando que a máscara m possui valor 0 nos locais em que a imagem é conhecida, e valor 255 nas regiões deterioradas, esse método é iterativo e é aplicado em imagens da seguinte forma:

1. $g_0 = g$
2. Obtem transformada de Fourier da máscara, $M = FFT(m)$
3. Para cada iteração $k = 1 \dots T$:
   a) Obtem transformada de Fourier da imagem $G_k = FFT(g_{k-1})$
   b) Filtra G_k, zerando coeficientes das frequências de G_k relativas a:

[1] `https://en.wikipedia.org/wiki/Inpainting`

i. maiores ou iguais a 90% do máximo da magnitude de M (remove em G_k as frequencias mais presentes na máscara), e

ii. menores ou iguais a 1% do máximo da magnitude de G_k (removendo em G_k as frequências com pouca influência) .

c) Obtem a transformada inversa da versão da imagem com frequência limitada (no passo anterior), i.e. $g_k = IFFT(G_k)$

d) Realiza convolução de g_k com um filtro de média 7×7 (para suavizar o resultado). OBS: essa operação também pode ser realizada antes da inversa, no domínio da frequência.

e) Renormaliza g_k de forma a garantir o intervalo $[0, 255]$

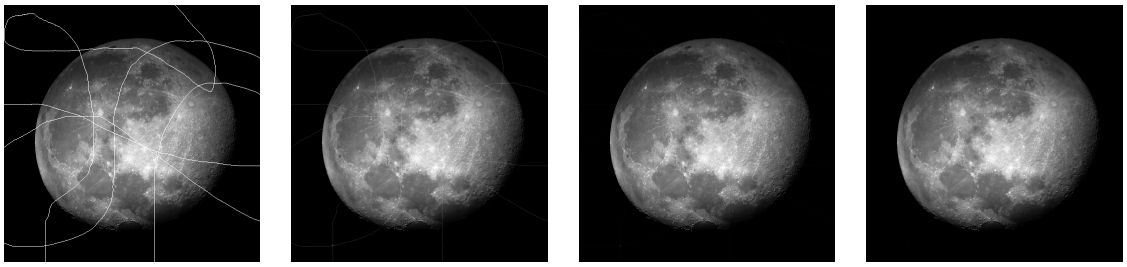
f) Insere os pixels conhecidos na estimativa k, de forma que:

$$g_k = (1 - m/255) \cdot g_0 + (m/255) \cdot g_k$$

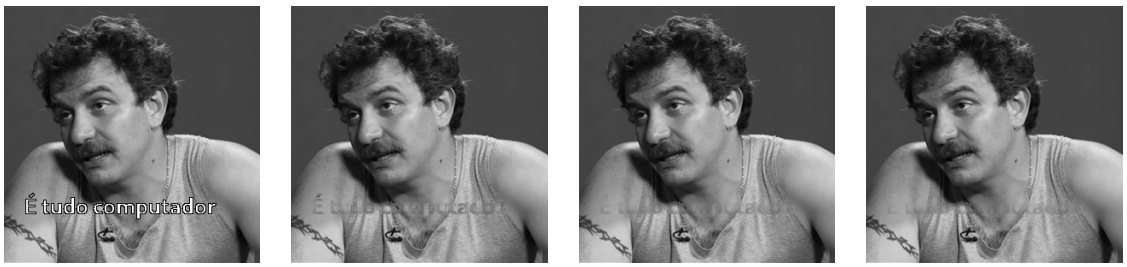
OBS: para FFT e IFFT, use `np.fft.fft2()` e `np.fft.ifft2()`, respectivamente.

Ao final das iterações, compara a imagem g_k com a imagem original i, usando $RMSE(g_k, i)$, exibindo o erro com 5 casas decimais.

Como referência, veja dois exemplos de inpainting por meio do algoritmo Gerchberg-Papoulis com filtragem.



F iteração 1 iteração 2 iteração 3



F iteração 1 iteração 2 iteração 3

Entrada e saída

Exemplo de entrada:

```
moon.png
moon-deta.png
moon-ma.png
10
```

Exemplos de saída: apenas o valor do RMSE entre a imagem original e a imagem restaurada, com 5 casas decimais

```
0.00499
```

Submissão e instruções

No sistema Run.Codes deve incluir apenas o arquivo `.py`

1. **É obrigatório comentar seu código**. Como cabeçalho insira seu nome, número USP, código da disciplina, ano/semestre e o título do trabalho. Haverá desconto na nota caso esse cabeçalho esteja faltando, além de descontos para falta de comentários.
2. **É obrigatório organizar seu código em funções**.