Visão e Processamento de Imagens - Avaliação única - Parte II

Preste atenção para as regras da prova

1- O fonte latex (.tex) da prova será disponibilizado para facilitar que você não tenha que copiar o enunciado das questões. Todas as questões devem ser respondidas no mesmo arquivo.
2- A prova é **individual**. É permitido a consulta a livros, apontamentos ou Internet, desde que devidamente referenciada. Não é permitida a consulta a colegas, amigos, família, cachorro, papagaio e etc.
3- A prova deve ser entregue diretamente no Paca, assim como todos os códigos e imagens devem ser entregues no mesmo arquivo comprimido. **Duração da prova: 14 dias**.
4- Cada questão vale 20 pontos (pois são apenas 3 questões) para a graduação e 15 pontos para a pós-graduação (pois são 4 questões).

Q1. Para fazer esta questão, leia primeiro o artigo abaixo:

- `http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-85.pdf`
- Faça um resumo do artigo de acordo com as indicações que deixei no paca (artigos sobre como fazer um resumo). TODO
- Implemente o método ELA (Error Level Analysis) em Python (apresente o **algoritmo** na prova e anexe o código em Python no arquivo zip).

  Abaixo o trecho onde o algoritmo efetivamente foi implementado. O código completo está em **source/ela.py**.

```
import os
from PIL import Image, ImageChops, ImageEnhance
from time import gmtime, strftime

# you can change the image directory here
HOAX_IMAGES = './HoaxImages/'

def check_image(img_path, scale=20.0, show=False):
    """ Compute the Error Level Analysis for the given
       image

    Save a copy of the given image changing its error
       rate, in our case 95%,
    and compute the difference between this image over
       the original. In addition, a
    scale is also applied to the final result for
       better viewing.

    References:
    http://blackhat.com/presentations/bh-dc-08/Krawetz/
       Whitepaper/bh-dc-08-krawetz-WP.pdf
```

```
        http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/
           EECS-2015-85.pdf
        """
        try:
            img = Image.open(img_path)
        except FileNotFoundError:
            print ("File '"+ img_path +"' not found.")
            return

        # check the image format for JPEG only
        if img.format is not 'JPEG':
            log(img_path + ' is not a JPEG file')
            return

        log("Image size "+ str(img.size[0]) + "x" + str(img
           .size[1]))
        short_file_name = os.path.splitext(img_path)[0]

        # save a copy of the image with a different
           inferior error rate
        resaved_path = short_file_name + '_resaved'
        img.save(resaved_path, 'JPEG', quality=95)
        resaved_img = Image.open(resaved_path)

        # compute the difference between the given image
           and the image
        # generated applying a scale to increase the
           brightness
        ela_img = ImageChops.difference(img, resaved_img)
        ela_img = ImageEnhance.Brightness(ela_img).enhance(
           scale)
        ela_img.save(short_file_name + '_ela.png')
        if (show):
            ela_img.show()

        os.remove(resaved_path)
```

– Teste seu algoritmo com as imagens que deixei no paca para este exercício. Quantas imagens seriam consideradas modificadas por esse método? Comente o resultado, comparando com a sua intuição.

Q2. Esta questão refere-se à transformada de Fourier.

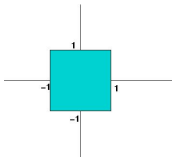
– Encontre a transformada de Fourier da função:

$$f(x) = \begin{cases} 7 & \text{if } -5 < x < 5 \\ 0 & \text{caso contrário} \end{cases}$$

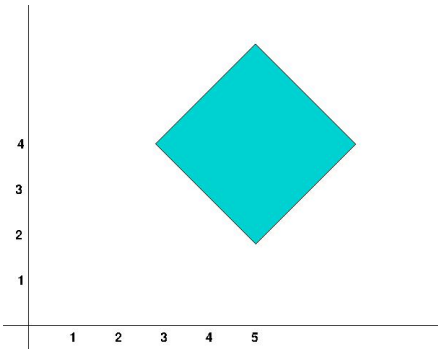
- Encontre a transformada de Fourier da função $g(x) = f(x)\cos\omega_0 x$, sabendo que a transformada de Fourier de $f(x)$ é dada por $F(\omega)$
- Ache a inversa da transformada de Fourier de $G(\omega) = 20\frac{\sin 5\omega}{5\omega}e^{-3\omega i}$
- Calcule a DFT do sinal $f = \{1, 3, 5, 3, 1\}$

Q3.

- Calcule (apresente os cálculos) dos descritores de Fourier das figuras 1a e 1b. Lembre-se que os pontos da borda do quadrado serão representados por pontos no plano de Argand-Gauss. Isto é, cada ponto no plano passa a ser um número complexo e a borda passa a ser um vetor de pontos complexos, como num sinal, mas com valores complexos.
- Para confirmar que seus cálculos estão corretos, implemente um programa em Python que receba como entrada um vetor de números complexos (que são as coordenadas das bordas) e retorne os descritores de Fourier do vetor de entrada. Você pode usar as funções fornecidas pela biblioteca NUMPY para facilitar a programação.
- Para reconstruir a curva, faça uma função que receba um vetor com os descritores de Fourier, um número N de descritores a serem usados e grafique os pontos num plano cartesiano (para fazer a mesma figura que fizemos nos slides das aulas 15 e 16.



(1a) Quadrado de lado 1



(1b) Quadrado de lado 3

Q4. Apenas para os alunos de pós-graduação

- Leia o artigo do Torre e do Poggio `ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-768.pdf` e faça um resumo de acordo com as indicações que deixei no paca (artigos sobre como fazer um resumo).
- O que é um problema mal-posto?
- O que é regularização?
- Qual a importância do teorema apresentado no artigo?
- O que são filtros de banda limitada? Qual a sua importância no artigo?
- Quais são os métodos de encontrar borda apresentados no artigo?