Programação orientada a objetos
Detector de borda

Conceitos importantes

- **Detector de bordas**

Uma das tarefas mais desafiadoras na área de Processamento Digital de Imagens é a detecção de bordas, pois muitos fatores influenciam a qualidade do resultado gerado, como iluminação, contraste, entre outros. Na literatura, diferentes detectores de bordas foram propostos, cada um deles apresentando vantagens e desvantagens em relação aos demais. Neste projeto será abordado o filtro de Sobel, que é um filtro espacial baseado no gradiente, que enfatiza regiões de alta frequência espacial na imagem (bordas).

A operação do filtro de Sobel consiste na convolução de dois *kernels* definidos, G_x e G_y, com a imagem. Cada um desses *kernels* gera uma resposta, enfatizando as bordas verticais e horizontais, respectivamente. Os *kernels* são apresentados a seguir.

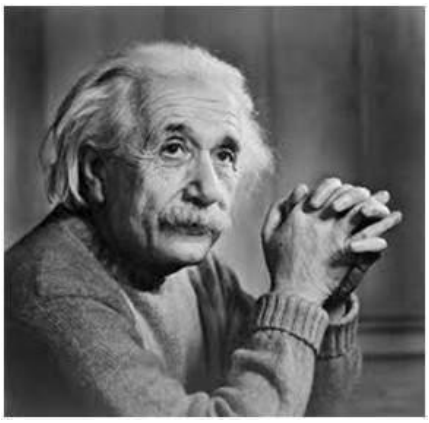
$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \qquad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

A convolução consiste em colocar o centro do kernel no pixel de interesse, multiplicar cada pixel vizinho pelo valor do kernel e somar o resultado. A convolução de G_x e G_y com a imagem original, A, gera duas novas imagens, E_x e E_y, respectivamente. A Figura 6 apresenta um exemplo da convolução entre a imagem original e os *kernels* G_x e G_y.

Por fim, o resultado final do filtro de Sobel é obtido com o cálculo da magnitude do gradiente ponto a ponto, combinando os resultados parciais, E_x e E_y, como

$$E = \sqrt{E_x^2 + E_y^2} \tag{5}$$

Os cálculos devem ser feitos em ponto flutuante, e no final deve ser feita a conversão para a escala 0 a 255, sendo que 255 corresponde ao maior valor calculado para *E*. A Figura 7 apresenta o resultado final do filtro Sobel, de acordo com a Equação 5.



(a)

(b)

(c)

Figura 6: exemplo da aplicação dos *kernels* G_x e G_y na imagem original. (a) imagem original A; (b) imagem E_x; (c) imagem E_y.

Figura 7: resultado final do filtro de Sobel.

Tarefas

Você deve agregar as novas funcionalidades, requisitadas neste projeto, à classe *Image* implementada no Projeto 02, de forma que os métodos anteriores estejam, também, disponíveis para utilização. **Algumas operações solicitadas neste projeto podem depender da implementação de funcionalidades desenvolvidas previamente.**

Sua implementação deve estar preparada para interpretar requisições de operações individualmente, bem como uma sequência delas, que devem ser aplicadas à imagem de entrada. Para maior esclarecimento dos detalhes descritos, ver a seção **Entrada**.

Você deve implementar, entre outros, o seguinte método:

- **sobel()** : que aplica o filtro de Sobel na imagem representada pelo objeto *Image*. O resultado deste método deve ser o produto da aplicação da Equação 5 devidamente convertido para valores inteiros conforme especificado, ou seja, as bordas, como apresentadas na Figura 7. Deve-se utilizar *zero-padding*. Este método deve retornar um novo objeto do tipo *Image*, com todas as informações pertinentes, e a imagem resultante deve ser salva em diretório especificado (ver seção **Entrada**).

Entrada

A passagem dos parâmetros de entrada do programa deve ser feita por linha de comando, seguindo o padrão especificado abaixo:

python <nome_programa>.py --imgpath <caminho_imagem_entrada> --op <operacao> [--outputpath <caminho_imagem_saida>]

A seguir, são descritos os parâmetros:

- **--imgpath** : consiste no caminho para a imagem, incluindo o nome;

- **--op** : consiste na operação a ser realizada individualmente, ou no conjunto de operações que devem ser realizadas, em sequência, na imagem original. Este parâmetro deve ser uma lista com um ou mais valores. Os possíveis valores da lista devem ser: *thresholding*, *sgt*, *mean*, *median*, *sobel*. Além disso, quando a operação escolhida possuir um parâmetro, este deve ser incluído também. Como exemplo, considere que a sequência de operações a ser aplicada sobre a imagem original seja, nesta ordem, *mean* → *sgt* → *sobel*. Então, deve ser utilizada a seguinte forma de *--op*:

  --op [mean --k 3 sgt --dt 1 sobel]

  Observe que tanto as operações quanto os valores para cada um dos parâmetros foi escolhido arbitrariamente, apenas para ilustração.

- **--outputpath** : consiste no diretório em que as imagens resultantes serão salvas; note que, diferente de *--imgpath*, este parâmetro **não** inclui o nome da imagem resultante, pois este deve estar definido no código. Por padrão, caso esta opção não seja utilizada pelo usuário, deve-se considerar o diretório corrente.

Saída

Seu programa deve salvar, quando pertinente, as imagens de saída no diretório especificado com o parâmetro *--outputpath*. É recomendado que o nome da imagem seja igual ao nome do método que a gerou, por exemplo, *sobel.pgm* quando a imagem resultante for gerada pela aplicação do filtro de Sobel. Além disso, como saída na tela, seu programa deve exibir o nome da imagem original e as operações aplicadas nela, na sequência requisitada. A impressão deve ser no estilo chave-valor, um par por linha, e para parâmetros relacionados à operação, quando houver, seus pares devem estar indentados logo após o nome da operação, com quatro espaços de recuo. Observe o padrão, seguindo o exemplo de entrada utilizado na seção **Entrada**:

```
image_name : <image_name>
op : mean
   k : 3
op : sgt
   dt : 1
op : sobel
```

Observações

- O código deve estar num arquivo Python simples (não notebook);
- **não é permitido o uso de pacotes de manipulação de imagens**;
- para a leitura dos parâmetros de entrada, é permitida a utilização do pacote *argparse*;
- lembre-se de organizar o seu código de forma apropriada, levando em conta as boas práticas de programação.

Material auxiliar

- http://proceedings.informingscience.org/InSITE2009/InSITE09p097-107Vincent613.pdf