VISÃO POR COMPUTADOR

2º Semestre **Ano lectivo 2016/2017**

Trabalho Prático Nº 2

FILTRAGEM e DETECÇÃO DE ARESTAS

FILTROS PASSA-BAIXO (*Filtragem de ruído*) e de DETECÇÃO DE ARESTAS

A filtragem é uma técnica de processamento de sinal/imagem que modifica e/ou melhora a informação contida na imagem, realçando ou removendo características da imagem. Em geral a filtragem linear é uma operação de vizinhança, na qual um pixel da imagem resultado é obtido processando os pixeis na vizinhança do pixel correspondente na imagem origem (normalmente por meio de uma convolução ou correlação). Todos os filtros a ser implementados neste trabalho existem na "Image Processing Toolbox" do Matlab. No entanto o objectivo deste trabalho é permitir a aprendizagem da implementação de filtros lineares por meio de convolução.

Diferenciação Numérica

Considere-se a função real $f(x)$ da qual se conhecem N amostras efectuadas em pontos equidistantes $x_1, x_2, \ldots, x_N$, de modo a que $x_i = x_{i-1} + h$ com $h>0$.

Representando a função $f(x)$ nos pontos $x+h$ e $x-h$ através de uma série de Taylor obtem-se:

$$f(x+h)=f(x)+hf'(x)+1/2\ h^2 f''(x)+O(h^3)$$
$$f(x-h)=f(x)-hf'(x)+1/2\ h^2 f''(x)+O(h^3)$$

onde $O(h^3)$ representa o erro de truncatura.

Subtraindo as equações apresentadas e resolvendo em ordem a $f'(x)$ obtem-se a equação para a primeira derivada de $f(x)$:

$$f'(x)=(f(x+h) - f(x-h))/2h+O(h^2).$$

De modo análogo, somando as equações obtem-se a equação para a segunda derivada de $f(x)$:

$$f''(x)=(f(x+h)-2f(x)+f(x-h))/ h^2+O(h)$$

Primeiras derivadas

$$f'_i=(f_{i+1} - f_{i-1})/2h+O(h^2)$$
$$f'_i=(-f_{i+2} +8f_{i+1} -8f_{i-1} +f_{i-2})/12h+ O(h^4)$$

Segundas derivadas

$$f''_i=(f_{i+1} -2f_i +f_{i-1})/ h^2+O(h)$$
$$f''_i=(-f_{i+2} +16f_{i+1} -30f_i +16f_{i-1} -f_{i-2})/12h^2+ O(h^3)$$

As equações apresentadas são chamadas de *equações de diferenciação central* uma vez que calculam as derivadas em x_i com base em amostras de intervalos simétricos centrados em x_i.

As utilização de intervalos assimétricos origina solucões de diferenciação em avanço (forward) ou em atraso (backward), resultando para as primeiras derivadas as seguintes equações:

$$f'_i=(f_{i+1} - f_i)/h+O(h)$$

$$f'_i=(f_i - f_{i-1})/h+O(h)$$

Exemplo de Máscaras de Diferenciação:

Laplaciano *(segunda derivada : aproxima a estimação da soma das 2ªs derivadas)*

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Sobel *(primeira derivada : aproxima o cálculo do gradiente numa imagem)*

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

As duas máscaras apresentadas detectam as derivadas horizontais e verticais respectivamente.

Prewitt *(primeira devivada : idêntico ao Sobel, com a diferença de que ambas as linhas contribuem com o mesmo peso)*

$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Roberts *(primeira derivada assimétrica)*

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Trabalho a realizar :

Parte I:

1. Efectuar a leitura de uma qualquer imagem existente em disco. (Ex: imagem RGB "cameraman.tif").
   Para converter uma imagem s cores numa imagem em níveis de cinzento utilizar a função *rgb2gray*.
2. Adicionar à imagem diferentes tipos de ruído (separadamente, ou seja não se deve aplicar o ruído sequencialmente à mesma imagem—o ruído é sempre aplicado à imagem original).
   a) ruído gaussiano com:
      --Média zero e 2 valores de variância;
      --Média 10, e 2 valores de variância;
   b) ruído 'salt & pepper' com densidade 0.1 e 0.5.
   c) ruído 'speckle' com variância 0.8.
      Utilizar a função *imnoise* disponível no MATLAB.
3. Efectuar a filtragem de cada uma das imagens utilizando o filtro de média, o filtro gaussiano e o filtro de mediana. Aplicar os filtros separadamente.

Utilize a função *imfilter* disponível no MATLAB.
As máscaras de convolução podem ser obtidas através da função *fspecial*.
A função *medfilt2* realiza a filtragem de mediana.

a) Comparar o desempenho dos diversos filtros na presença dos diferentes tipos de ruído. Que diferença existe entre a aplicação do filtro linear de média e do gaussiano?

   Utilizar a função *improfile* para visualizar os valores de nível de cinzento ao longo de uma recta horizontal na imagem. Comparar os valores antes e depois da filtragem. Juntar os gráficos.

b) Qual é a influência do desvio padrão na definição da máscara de convolução do filtro gaussiano? Justifique.

   Utilizar a função *fspecial* para obter a máscara de filtragem associada a um determinado desvio padrão.

4. Implementar o ***filtro de mediana*** e comparar o seu funcionamento e desempenho com o filtro de mediana disponível no MATLAB.

FILTRO DE MEDIANA

O filtro de mediana é um filtro de vizinhança onde o valor de um pixel (x,y) é substituido pela *mediana* dos pixeis da sua vizinhança.

Algoritmo MED_FILTER

Considere-se **I** a imagem origem, $\mathbf{I_m}$ a imagem resultando e **n** um número ímpar.

Para cada pixel **I(x,y)** :

1 - Calcular a ***mediana*** *m*(x,y) dos valores de cinzento numa vizinhança *n*x*n* de (x,y), {**I(x+h,j+k)**,h,k∈{-n/2,n/2}}, representando n/2 uma divisão inteira.

2 - Atribuir $\mathbf{I_m(x,y)=m(x,y)}$.

O mediana de um conjunto de **k** valores corresponde ao valor central (ordem k/2) desse conjunto após o ordenamento dos seus valores.

Parte II:

Considere de novo a imagem original (sem as alterações efectuadas na Parte I):

5. Determinar os pixeis pertencentes a arestas utilizando as máscaras correspondentes aos operadores de : ***Sobel, Laplaciano, LoG (Laplaciano da Gaussiana) e Canny. Use fspecial*** para criar as máscaras do Sobel, o Laplaciano e o LoG**.** Para aplicar o Canny use a função ***edge***. Fazer as filtragens separadamente e não sequencialmente: os filtros são aplicados directamente às imagens originais. Fazer o "display" das imagens de módulo e de fase (nos casos em que isso se aplique). Nos casos em que o gradiente é calculado faça o display do vector do gradiente usando a função *quiver*. Tal display deve ser feito nos pixels onde o módulo do gradiente é localmente máximo. Para isso considere uma vizinhança de 3*3 em torno de cada pixel. Se o módulo do gradiente nesse pixel for maior que o módulo do gradiente nos 8 vizinhos, deve ser feito o traçado do vector.

6. Obtenha os contornos da imagem utilizando a função *edge* com as opções *Canny*, *log*, *Sobel* e *zerocross*. Use a informação de orientação determinada pelo "edge detector" para obter uma imagem da orientação dos contornos. Faça um histograma com as orientações dos contornos. Quantifique os 360º em 8 intervalos e faça corresponder a cada valor uma intensidade de níveis de cinzento. Compare os resultados.

Algoritmo de extracção de contornos baseado no operador SOBEL

1 - Após a prévia filtragem da imagem *I* por meio de um filtro passa-baixo, convolucionar a imagem filtrada I_f com as máscaras de SOBEL apresentadas (horizontal e vertical), obtendo as imagens I_x e I_y.

2 - Cálcular a amplitude do gradiente em cada pixel (*i,j*) da imagem através de:

$$G(i,j) = \sqrt{I_x^2 + I_y^2}$$

obtendo a imagem de amplitude do gradiente *G*.

3 – Defina um limiar δ. Marcar os pixeis da imagem que constituem os contornos da imagem através da binarização da imagem de acordo com:

G(i,j)>δ então O(i,j)=255;
senão O(i,j)=0;