Universidade do Minho

Módulo 1 (anexo)

Explicação do caso de estudo

Convolução

A convolução é um operador matemático que, a partir de 2 funções *f()* e *g()*, produz uma terceira, *h()*. A função resultante, *h()*, é vista como uma versão modificada da função original, *g()*, mas ponderada (ou filtrada) pela função *f()*. h() é então a média pesada de g(), sendo os pesos dados por f(). Formalmente, a convolução é escrita como:

$$h(t) = (f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\delta\tau$$

A convolução é uma operação comummente utilizada em áreas como o processamento de sinal, visão por computador, engenharia electrotécnica e electrónica, entre outras. No contexto desta Unidade Curricular (UC) iremos aplicá-la a um problema de visão por Computador.

1. Sinais discretos unidimensionais

Se as funções consideradas forem discretas o integral apresentado acima converte-se num somatório. Na área da Visão por Computador as imagens são tipicamente sinais discretos pelo que é este o caso que nos interessa. A convolução discreta para o caso unidimensional (1D) é então definida como:

$$h[x] = (f * g)[x] = \sum_{i=-\lfloor n/2 \rfloor}^{\lfloor n/2 \rfloor} f[i + \lfloor n/2 \rfloor]g[x+i]$$

onde *n* é o número de elementos da função *f[]*.

A expressão acima assume que o filtro f[] está normalizado, isto é, que a soma de todos os seus elementos é 1. Em muitos casos tal não é verdade, sendo necessário dividir pelo somatório de todos os valores de f[]:

$$h[x] = (f * g)[x] = \frac{\sum_{i=-\lfloor n/2 \rfloor}^{\lfloor n/2 \rfloor} f[i + \lfloor n/2 \rfloor]g[x+i]}{\sum_{i=0}^{n} f[i]}$$

Nesta UC utilizaremos esta formulação que tem a vantagem de permitir utilizar inteiros como valores dos filtros. Evitaremos assim gerar *assembly* para operações em vírgula flutuante.

EXEMPLO 1

Seja o vector abaixo a função discreta *g[x]* com 10 valores:

x	0	1	2	3	4	5	6	7	8	9
g[x]	12	13	15	21	14	12	3	-5	-3	-1

Seja o vector *f[]*, com 3 valores (*n=3*), o filtro a aplicar a *g[x]*:

x	0	1	2
f[x]	1	1	1

Então para cada ponto de *g[x]* podemos calcular *h[x]=(f*g)[x]*. Note que na verdade não podemos calcular *h[x]* para *x=0* e *x=9*, pois tal implicaria aceder aos valores de *g[-1]* e *g[10]* que não pertencem ao domínio de *g[]*. Desta forma *h[0]* e *h[9]* são indefinidos.

Ilustremos o processo para *x=1* (relembrando que *n=3*, o número de elementos do filtro *f[]*). O operador matemático $\lfloor z \rfloor$ representação a operação *floor()*, isto é, devolve o maior inteiro menor do que *z*. Neste caso temos $\left\lfloor \frac{3}{2} \right\rfloor = 1$, logo

$$h[1] = \frac{\sum_{i=-1}^{1} f[i+1]g[1+i]}{\sum_{i=0}^{3} f[i]} = \frac{(f[0]g[0] + f[1]g[1] + f[2]g[2])}{3} = 13$$

Procedendo da mesma forma deverá ser claro que

$$h[2] = \frac{\sum_{i=-1}^{1} f[i+1]g[2+i]}{\sum_{i=0}^{3} f[i]} = \frac{f[0]g[1] + f[1]g[2] + f[2]g[3]}{3} = 16$$

O resultado final será portanto

x	0	1	2	3	4	5	6	7	8	9
g[x]	12	13	15	21	14	12	3	-5	-3	-1
h[x]	.	13	16	17	16	10	3	-2	-3	.

A figura seguinte apresenta os gráficos de *g[x]* e *h[x]*

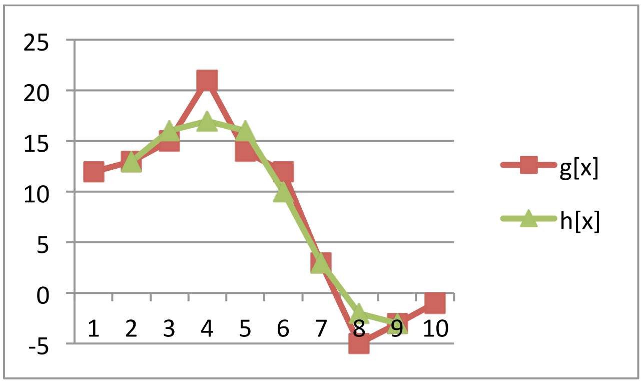

Ilustração 1- h[x] : convolução de f[x] com g[x]

Pode-se ver que *h[x]* é uma versão suavizada de *g[x]*. Os pontos onde *g[x]* tem variações mais bruscas ou valores mais extremos (*x=4* e *x=8*) são suavizados. Um filtro do tipo de *f[x]* aqui apresentado é designado por filtro passa-baixo, na medida em que elimina as altas frequências (ou variações) do sinal filtrado. Este tipo de filtros são frequentemente utilizados para eliminar ruído no sinal a processar.

O exemplo apresentado permite também ilustrar que para calcular *h[x]* o filtro *f[]* é aplicado a *g[]* numa janela centrada em *x* e com largura *n*, sendo *n* o número de elementos de *f[]*.

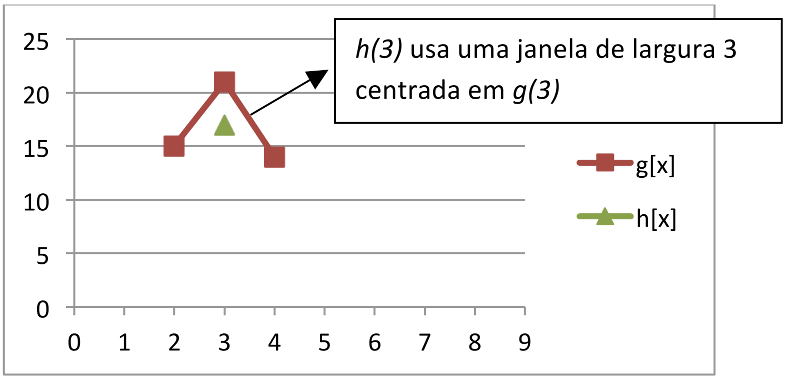

Ilustração 2 - janela de g[x] usada para calcular h[3]

FIM EXEMPLO 1

---

2. *Sinais discretos bidimensionais*

Uma imagem, *I[][]*, é um sinal bidimensional (2D) com largura (W) e altura (H). Pode ser convolvida com um filtro *f[]* 2D com largura U e altura V. Neste caso a expressão da convolução 2D é dada por

$$h[y][x] = (f * I)[y][x] = \frac{\sum_{i=-\lfloor U/2 \rfloor}^{\lfloor U/2 \rfloor} \sum_{j=-\lfloor V/2 \rfloor}^{\lfloor V/2 \rfloor} f[j + \lfloor V/2 \rfloor][i + \lfloor U/2 \rfloor] I[y + j][x + i]}{\sum_{i=0}^{U} \sum_{j=0}^{V} f[j][i]}$$

Esta expressão é apenas uma generalização da expressão anterior. *[y][x]* identifica o pixel da linha *y* e coluna *x* da imagem *I*. Note que agora para calcular *h[y][x]* é utilizada uma janela bidimensional centrada em *I[y][x]* com dimensões UxV, isto é, as dimensões do filtro *f[][]* – ver figura abaixo.

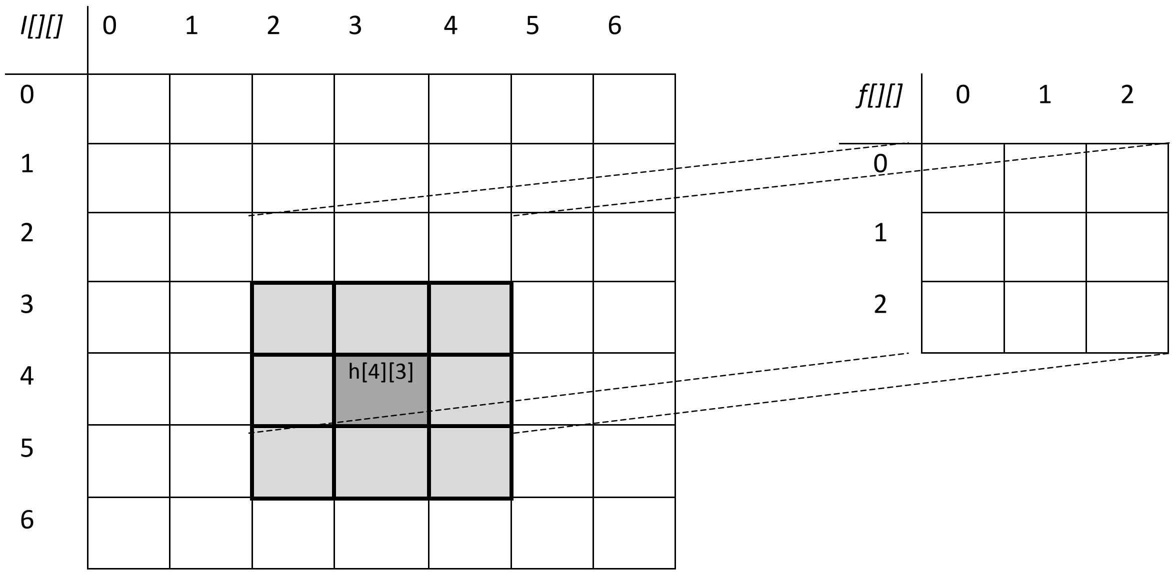

Ilustração 3 - janela de I[y][x] usada para calcular h[4][3]: é usada uma janela com a dimensão 3x3, isto é, a mesma dimensão do filtro f[][]

3. *Resultados*

Os filtros apresentados acima calculam o valor médio da janela atribuíndo o mesmo peso a todos os pixels abrangidos por essa janela. São designados por *box filters*. Um tipo de filtro passa-baixo muito comum é o filtro de Gauss. Este tem a particularidade de atribuir mais peso aos pixeis mais próximos do pixel central da janela; a forma da curva de um filtro de Gauss é portanto um sino, com o máximo no centro da janela. Será este tipo de filtro que iremos usar ao longo desta Unidade Curricular.

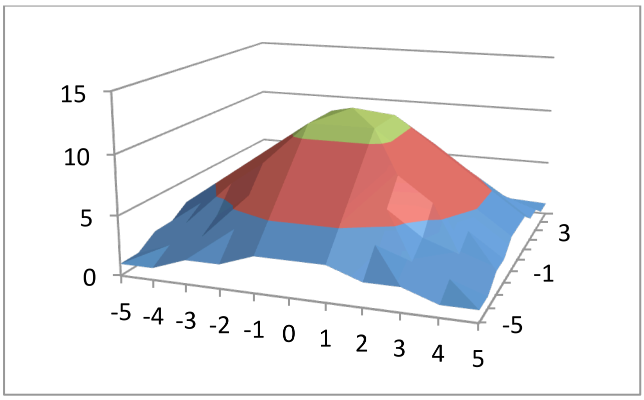

Ilustração 3 - Curva típica de um filtro de Gauss não normalizado

Todos os filtros apresentados nas secções anteriores tinham largura 3. A largura do filtro pode ser qualquer. No caso dos filtros de Gauss discretos esta largura é usualmente um valor ímpar e os filtros são quadrados, isto é, têm largura e altura iguais (U=V). Será também esta a norma ao longo desta UC. Filtros com maior largura têm menor frequência de corte, isto é, eliminam ruído com frequências menores; em contrapartida a imagem resultante será mais suavizada, isto é, exibe menor contraste.

Nas figuras abaixo exibem-se várias imagens do Professor Brian Kernighan: sem ruído, com ruído adicionado digitalmente e o resultado de aplicar um filtro de Gauss com largura 3, 7, 15 e 23.

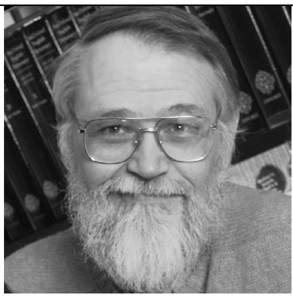	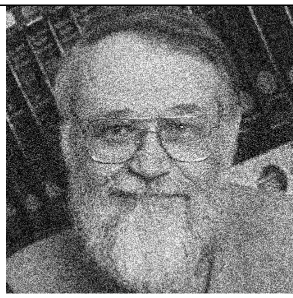	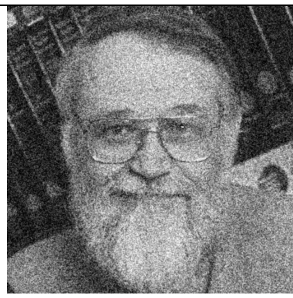
Ilustração 4 - Kernighan Imagem Original	Ilustração 5 - Kernighan c/ Ruído aleatório	Ilustração 6 - Kernighan Gauss U=3
		
Ilustração 7 - Kernighan Gauss U=7	Ilustração 8 - Kernighan Gauss U=15	Ilustração 9 - Kernighan Gauss U=23