

# Processamento de Imagem

## 3 - Filtros Espaciais

Análise e Processamento de Imagem (M4031, M4094)

Processamento de Sinal e Imagem em Física Médica (F4012)

### [Conteúdo]

click on it

André R. S. Marçal

Departamento de Matemática  
Faculdade de Ciências, Univ. Porto (FCUP)  
<http://www.fc.up.pt/pessoas/andre.marcal>

versão 1.1 - 23 Fevereiro 2021

# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>3</b>
<b>2</b>	<b>Filtros espaciais lineares</b>	<b>4</b>
<b>3</b>	<b>Filtros espaciais não-lineares</b>	<b>5</b>
3.1	Filtros de ordem e de mediana . . . . .	5
3.2	Outros filtros não-lineares . . . . .	6
<b>4</b>	<b>Suavização e Redução de Ruído</b>	<b>7</b>
4.1	Ruído 'Sal e Pimenta' . . . . .	7
4.2	Linhas em falta . . . . .	10
<b>5</b>	<b>Deteção de contornos</b>	<b>11</b>
5.1	Contornos simples . . . . .	11
5.2	Contornos direcionais . . . . .	14
5.3	Detetores de cantos . . . . .	17
<b>6</b>	<b>Realce de contornos</b>	<b>17</b>
<b>7</b>	<b>Filtros espaciais no MATLAB</b>	<b>19</b>
7.1	Filtros lineares . . . . .	19
7.1.1	Definição do filtro . . . . .	19
7.1.2	Aplicação do filtro . . . . .	20
7.2	Filtros não-lineares . . . . .	20
7.3	Contaminação com ruído . . . . .	21
<b>8</b>	<b>Exercícios propostos</b>	<b>22</b>
8.1	APPENDIX - Exercices (in English) . . . . .	23

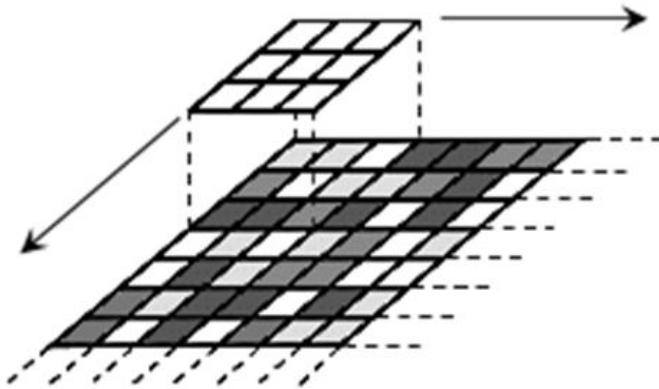


Figura 1: Esquema que ilustra o procedimento a seguir para aplicação de um filtro espacial (de  $3 \times 3$  pixels) a uma imagem digital.

## 1 Introdução

A utilização de filtros espaciais permite calcular novos valores para cada pixel ( $p_{ij}$ ) com base nos valores originais dos pixels de uma vizinhança em torno de  $p_{ij}$ . Normalmente a vizinhança é de tamanho reduzido, quando comparada com a imagem, e por simplicidade usa-se muitas vezes um quadrado com um número ímpar de pixels de lado (por exemplo  $3 \times 3$ ,  $5 \times 5$ ,  $7 \times 7$ , etc.). O processo está ilustrado na figura 1, para uma vizinhança de  $3 \times 3$  pixels. Para se calcular o novo valor de um dado pixel, coloca-se a janela do filtro (que define a vizinhança) centrada sobre esse pixel, e aplica-se a operação definida para o filtro. As operações, que actuam sobre os valores numéricos da imagem original, podem ser lineares ou não-lineares.

No caso ilustrado na figura 1, a janela do filtro (quadrado de  $3 \times 3$  pixels) está colocada sobre o pixel da 2<sup>a</sup> linha e 2<sup>a</sup> coluna da imagem. O novo valor desse pixel irá ser calculado com base nos valores numéricos dos 9 pixels da vizinhança (linhas de 1 a 3 e colunas de 1 a 3). A colocação da janela do filtro nas zonas centrais da imagem não causa problemas, uma vez que há sempre pixels na vizinhança. No entanto, nos pixels próximos das margens isso nem sempre acontece. Neste caso (janela de  $3 \times 3$  pixels), os problemas surgem para pixels na primeira e última linha/coluna. Por exemplo, colocando a janela do filtro sobre o pixel do canto superior esquerdo da imagem (linha 1, coluna 1), há apenas 4 elementos do filtro sobre a imagem, o que potencialmente impediria o cálculo do novo valor obtido por filtragem espacial. Uma forma que é frequentemente usada em para resolver este problema computacional é através da criação de uma margem suficientemente grande em torno da imagem (*Padding*), que é depois retirada. Essa margem pode tomar valores iguais ao das linhas / colunas mais próximas, o valor 0, ou outros valores a definir.

$p_{11}$	$p_{12}$	$p_{13}$	$p_{14}$	$p_{15}$	...
$p_{21}$	$p_{22}$	$p_{23}$	$p_{24}$	$p_{25}$	...
$p_{31}$	$p_{32}$	$p_{33}$	$p_{34}$	$p_{35}$	...
$p_{41}$	$p_{42}$	$p_{43}$	$p_{44}$	$p_{45}$	...
$p_{51}$	$p_{52}$	$p_{53}$	$p_{54}$	$p_{55}$	...
...	...	...	...	...	...

$a_{11}$	$a_{12}$	$a_{13}$
$a_{21}$	$a_{22}$	$a_{23}$
$a_{31}$	$a_{32}$	$a_{33}$

$p_{11}$	$p_{12}$	$p_{13}$	$p_{14}$	$p_{15}$	...
$p_{21}$	$p_{22}$	$p_{23}$	$p_{24}$	$p_{25}$	...
$p_{31}$	$p_{32}$	$p_{33}$	$p_{34}$	$p_{35}$	...
$p_{41}$	$p_{42}$	$p_{43}$	$p_{44}$	$p_{45}$	...
$p_{51}$	$p_{52}$	$p_{53}$	$p_{54}$	$p_{55}$	...
...	...	...	...	...	...

Figura 2: Aplicação de filtro espacial linear: (a) secção de imagem P, (b) filtro de  $3 \times 3$  pixels, (c) vizinhança relevante para calcular o novo valor do pixel  $p_{33}$ .

## 2 Filtros espaciais lineares

Consideremos uma imagem P, com intensidades  $p(i,j)$  (ou  $p_{ij}$ ) para o pixel na linha i e coluna j, e um filtro espacial linear A, com coeficientes  $a(i,j)$  (ou  $a_{ij}$ ). A aplicação do filtro linear A à imagem P consiste na operação de correlação ( $\otimes$ ), dando origem a uma imagem  $Q = P \otimes A$ . Por simplicidade, consideremos que o filtro A é quadrado com  $2K+1$  pixels de lado ( $K=1$  para um filtro de  $3 \times 3$  pixels,  $K=2$  para  $5 \times 5$ , etc.). A intensidade  $q(i,j)$  (ou  $q_{ij}$ ) do pixel em (i,j) de Q pode ser calculada a partir de (1).

$$q(i,j) = \sum_{m=-K}^K \sum_{n=-K}^K p(i+m, j+n) a(m+K+1, n+K+1) \quad (1)$$

O processo pode ser visto de uma forma mais simples recorrendo ao exemplo ilustrado na figura 2, onde se usa um filtro de  $3 \times 3$  pixels ( $K=1$ ). A figura mostra uma secção do canto superior esquerdo da imagem (a), o filtro (b) e a vizinhança a considerar para o cálculo do novo valor do pixel  $p_{33}$  (c). Para este caso, a equação (1) pode ser escrita como

$$q(3,3) = \sum_{m=-1}^1 \sum_{n=-1}^1 p(3+m, 3+n) a(m+2, n+2) \quad (2)$$

ou, de uma forma ainda mais simplificada,  $q_{33} = p_{22}a_{11} + p_{23}a_{12} + p_{24}a_{13} + p_{32}a_{21} + p_{33}a_{22} + p_{34}a_{23} + p_{42}a_{31} + p_{43}a_{32} + p_{44}a_{33}$ . Ou seja, o procedimento consiste em colocar o filtro sobre a imagem, centrado no pixel de interesse, e somar o produto dos valores numéricos de cada pixel da vizinhança pelos coeficientes do filtro.

O processo de aplicação de um filtro espacial linear pode, em alternativa, ser feito através da convolução (\*),  $Q = P * A$ . Na prática não há normalmente diferença entre a operação de filtragem efectuada via correlação ou convolução, uma vez que os filtros são em geral simétricos.

Os filtros espaciais lineares permitem obter uma grande diversidade de resultados, dependendo da escolha dos coeficientes e também do tamanho do filtro. Em certos casos é conveniente usar filtros normalizados, com coeficientes  $a_{ij} \geq 0$  e com soma igual 1 ( $\sum_i \sum_j a_{ij} = 1$ ). Neste tipo de filtros a intensidade média

(a)	(b)	(c)	(d)																																				
<table border="1"> <tr><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	1	1	1	1	1	1	1	1	1	<table border="1"> <tr><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>2</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	1	1	1	1	2	1	1	1	1	<table border="1"> <tr><td>1/9</td><td>1/9</td><td>1/9</td></tr> <tr><td>1/9</td><td>1/9</td><td>1/9</td></tr> <tr><td>1/9</td><td>1/9</td><td>1/9</td></tr> </table>	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9	<table border="1"> <tr><td>0.1</td><td>0.1</td><td>0.1</td></tr> <tr><td>0.1</td><td>0.2</td><td>0.1</td></tr> <tr><td>0.1</td><td>0.1</td><td>0.1</td></tr> </table>	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.1
1	1	1																																					
1	1	1																																					
1	1	1																																					
1	1	1																																					
1	2	1																																					
1	1	1																																					
1/9	1/9	1/9																																					
1/9	1/9	1/9																																					
1/9	1/9	1/9																																					
0.1	0.1	0.1																																					
0.1	0.2	0.1																																					
0.1	0.1	0.1																																					

Figura 3: Filtros espaciais lineares de  $3 \times 3$  (a,b) e correspondentes versões normalizadas (c,d) para cálculo de média.

da imagem original é preservada, sendo no entanto alteradas as intensidades dos pixels localmente. Um filtro bastante usado é um filtro com os coeficientes todos iguais a 1, como o que é apresentado na figura 3(a). No entanto, a aplicação deste filtro sem normalização numa qualquer imagem produziria muito provavelmente uma imagem fortemente saturada, uma vez que o valor de cada pixel é substituído pela soma dos 9 pixels da vizinhança de  $3 \times 3$ . Através da normalização, o filtro passa a ter coeficiente com valor  $1/9$ , figura 3(c), sendo o valor de cada pixel substituído, no processo de filtragem espacial, pela média dos 9 vizinhos. Uma versão ligeiramente diferente é apresentada na figura 3(b,d). Este filtro de média considera uma contribuição ( $2x$ ) maior do pixel central em relação aos restantes 8 vizinhos. Estas operações de suavização são abordadas em maior detalhe na secção 4.

Há um outro grupo de filtros espaciais, com características diferentes, que são usados como detectores. Nesses filtros os coeficientes  $a_{ij}$  podem ser positivos, negativos e 0, sendo a soma dos coeficientes igual a 0 ( $\sum_i \sum_j a_{ij} = 0$ ). Os filtros espaciais para operações de deteção são abordados na secção 5.

### 3 Filtros espaciais não-lineares

Os filtros espaciais não-lineares actuam sobre a vizinhança escolhida (ex.  $3 \times 3$  ou  $5 \times 5$ ) através de uma operação não linear. Há vários exemplos de filtros espaciais não-lineares, sendo o filtro de mediana provavelmente o mais utilizado. Por simplicidade, consideremos que o filtro é quadrado com  $L$  pixels de lado.

#### 3.1 Filtros de ordem e de mediana

Um filtro de ordem (ou *rank filter*) substitui o valor de cada pixel pelo pixel correspondente a uma determinada ordem  $k$  ( $k=1, k=2, k=3$ , etc.) na vizinhança definida. Como casos particulares, consideram-se os filtros de máximo ( $k=1$ ), mínimo ( $k=L^2$ ) e mediana ( $k=(L^2+1)/2$ ). Por exemplo, um filtro de mediana de  $5 \times 5$  corresponde a um filtro de ordem 12 ( $k=(5^2+1)/2$ ).

A figura 4 mostra exemplos de aplicação de filtros não-lineares de  $3 \times 3$  a uma pequena imagem, com apenas 35 pixels ( $5 \times 7$ ), de 8-bits sem sinal (valores inteiros de 0 a 255). Inicialmente foi aplicado à imagem original (a) um filtro de mediana com *padding* 0, ou seja, foi considerado que os pixels da margem virtual em torno da imagem tem valor 0. Focando no pixel (2,2), com valor 120,

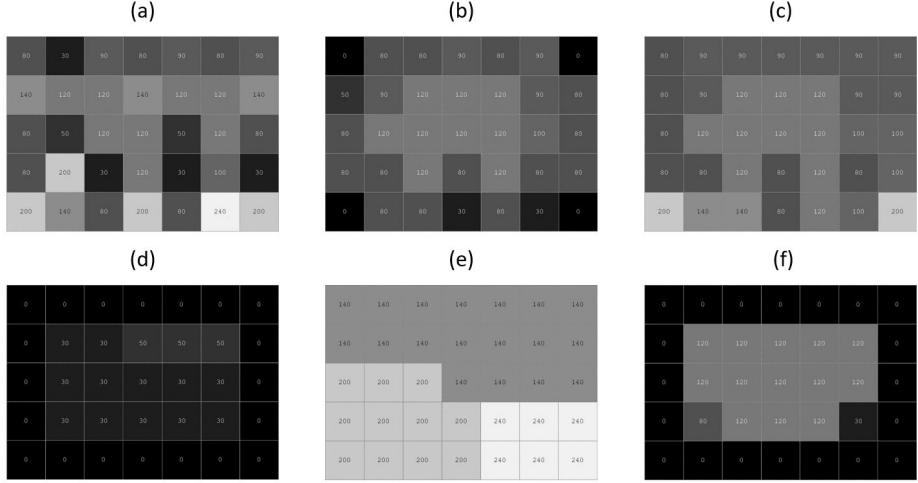


Figura 4: Exemplo de aplicação de filtros espaciais não-lineares de  $3 \times 3$ : imagem original (a), filtro de mediana com *padding* 0 (b) e simétrico (c), filtros de mínimo (d), máximo (e) e moda (f).

os 9 pixels da sua vizinhança de  $3 \times 3$  tem os seguintes valores (ordenados): 30, 50, 80, 80, 90, 120, 120, 120 e 140. Neste caso a mediana corresponde ao 5º valor da sequência, que é 90. Para o pixel (1,1), canto superior esquerdo, os pixels da vizinhança são: 0, 0, 0, 0, 0, 30, 80, 120 e 140, o que resulta numa mediana de 0. O resultado para a imagem de teste é apresentado na figura 4(b). Como se pode observar nesta imagem, o *padding* de 0 tem um impacto muito grande nas margens, em particular nos cantos. Uma alternativa mais interessante para o filtro de mediana é usar *padding* simétrico, onde os valores das linhas e colunas mais próximas são usados para a margem virtual (figura 4c).

A figura 4 mostra igualmente o resultado da aplicação de filtros de mínimo (d) e máximo (e), ambos com *padding* 0. Este *padding*, que é adequado para o filtro de máximo, tem um impacto muito adverso no mínimo, resultando em valores 0 para a primeira e última linha / coluna. Para o filtro de mínimo deve ser usado *padding* com valores constantes iguais ao máximo da escala (neste caso 255) para ter um efeito neutro no cálculo dos mínimos para cada pixel.

### 3.2 Outros filtros não-lineares

Há outros filtros espaciais não-lineares, tais como o filtro bilateral, usado para suavizar imagens sem perda de contornos, e o filtro de moda. A figura 4(f) mostra o resultado da aplicação de um filtro de moda de  $3 \times 3$  à imagem de teste (a). Uma vez que a vizinhança tem apenas 9 elementos, é frequente haver empates. Neste caso, em caso de empate é apresentado o valor mais baixo. Por exemplo, para o pixel (4,6), com valor 100 na imagem original (a), a vizinhança de  $3 \times 3$  contém 2 pixels com 30 e 2 pixels com 80, tendo no desempate sido selecionado o valor 30 (f). O filtro de moda é particularmente útil para simplificar imagens resultantes de um processo de classificação.

## 4 Suavização e Redução de Ruído

A suavização de uma imagem pode ser feita recorrendo a filtros espaciais lineares (2), por exemplo filtros de média como os indicados na figura 3(c,d), ou não-lineares como a mediana (3.1). Em ambos os casos a intensidade da suavização aumenta com o tamanho do filtro (vizinhança a considerar no cálculo da média ou mediana). Uma diferença importante no entanto é que o filtro de mediana preserva os contornos, enquanto que os filtros lineares de média tendem a esbater as zonas de transição entre tonalidades claras e escuras.

A figura 5 apresenta um exemplo de utilização de filtros espaciais para suavizar a imagem de teste 'Falkland' (a), de 8 bits (0-255) com  $512 \times 512$  pixels. Na coluna da direita são apresentados os resultados da aplicação do filtro linear de média (coeficientes todos iguais), numa vizinhança de  $3 \times 3$  (b),  $5 \times 5$  (d) e  $7 \times 7$  (f). A imagem filtrada tem menos detalhe que a imagem original, tornando-se cada vez mais suave ou esbatida, à medida que o tamanho do filtro aumenta. Isto é particularmente perceptível no telhado e na janela com grades, no canto inferior esquerdo da imagem. Um outro aspecto relevante é que os contornos se tornam esbatidos. Na coluna da esquerda são apresentados os resultados da aplicação do filtro de mediana, numa vizinhança de  $5 \times 5$  (c) e de  $7 \times 7$  (e). A principal diferença entre estas imagens e as produzidas pelos filtros de média é a forma como os contornos são preservados, o que pode ser observado por exemplo na transição entre o edifício e o céu, ou nas zonas escuras por baixo do telhado.

### 4.1 Ruído 'Sal e Pimenta'

Um caso em que o filtro de mediana é particularmente eficiente é na redução de ruído 'Sal e Pimenta'. Este tipo de ruído consiste na presença de pixels com valores extremos localizados aleatoriamente na imagem. A figura 6 apresenta 2 versões da imagem de teste 'Falkland' com ruído 'Sal e Pimenta' a 5 % (a) e 15 % (b). Neste caso, uma vez que a imagem é de 8 bits sem sinal (0-255) os pixels contaminados tem valores 255 (Sal) ou 0 (Pimenta). Para este tipo de ruído, os filtros espaciais lineares (média) não são eficazes uma vez que os pixels contaminados afectam negativamente os pixels vizinhos. Ao contrário, o filtro de mediana é extremamente eficaz uma vez que os valores extremos (0 e 255) não são normalmente em número suficiente para serem selecionados como mediana.

O desempenho dos filtros espaciais pode ser observado na figura 6, onde são apresentados resultados da aplicação de filtros de  $3 \times 3$  de média (c,d) e mediana (e,f) às 2 versões contaminadas da imagem 'Falkland'. Para o filtro de média, as imagens resultantes continuam a ter a presença de ruído, sendo muito diferentes da imagem original sem ruído (figura 5a). Ao contrário, o filtro de mediana de  $3 \times 3$  produz imagens visualmente quase iguais à original. No primeiro caso (contaminação de 5%) o resultado é quase perfeito (e), havendo no segundo caso (contaminação de 15%) alguns pixels que mantém valores extremos (f). A utilização de janelas maiores (por exemplo de  $5 \times 5$ ) para os filtros não se traduziria em grandes diferenças, sendo apenas provavelmente corrigidos alguns dos pixels ainda contaminados em (f).

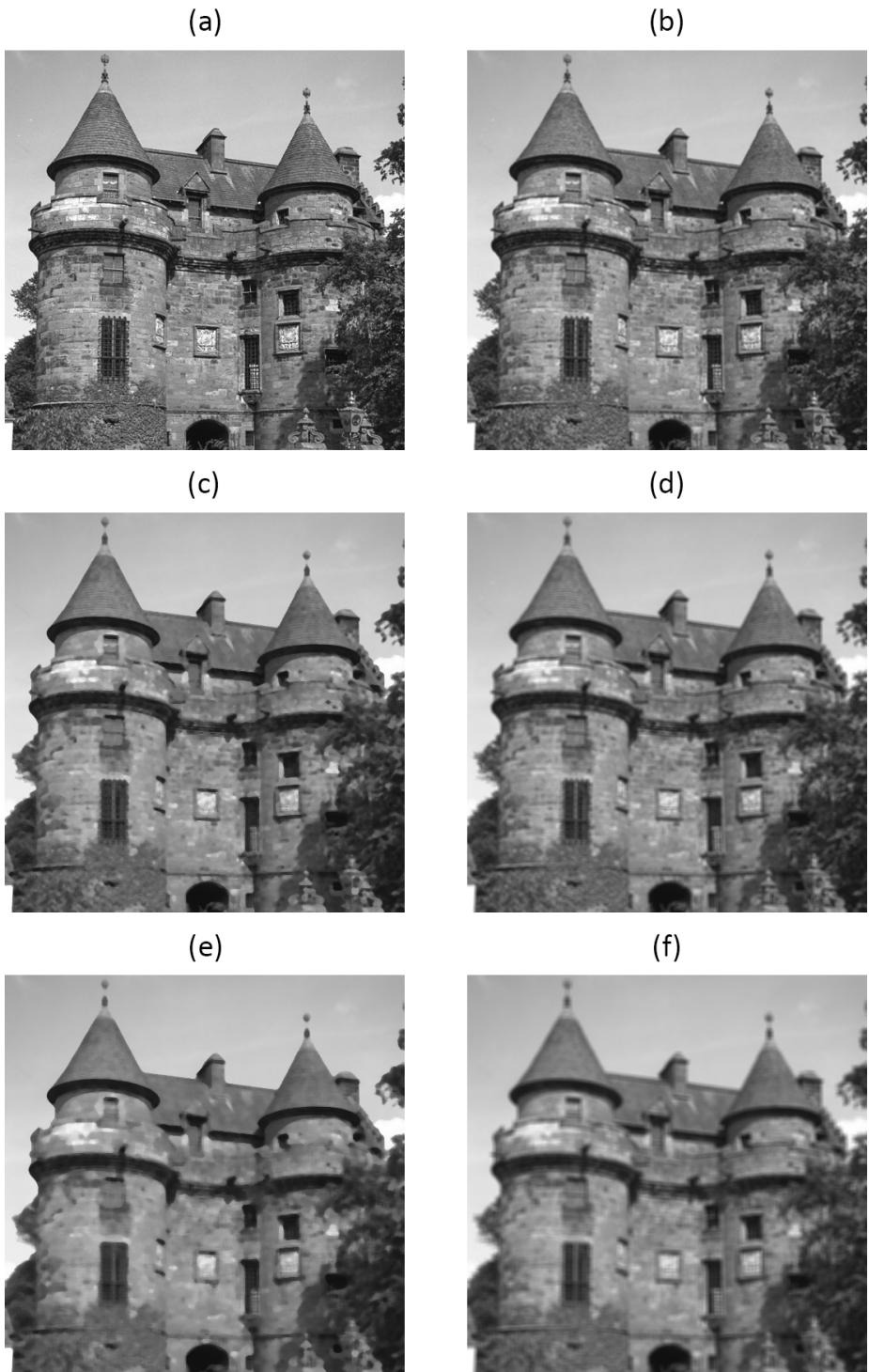


Figura 5: Utilização de filtros espaciais para suavizar imagem de teste 'Falkland', de 8 bits (0-255) com  $512 \times 512$  pixels: imagem original (a); filtros de média de  $3 \times 3$  (b),  $5 \times 5$  (d) e  $7 \times 7$  (f); filtros de mediana de  $5 \times 5$  (c) e  $7 \times 7$  (e).

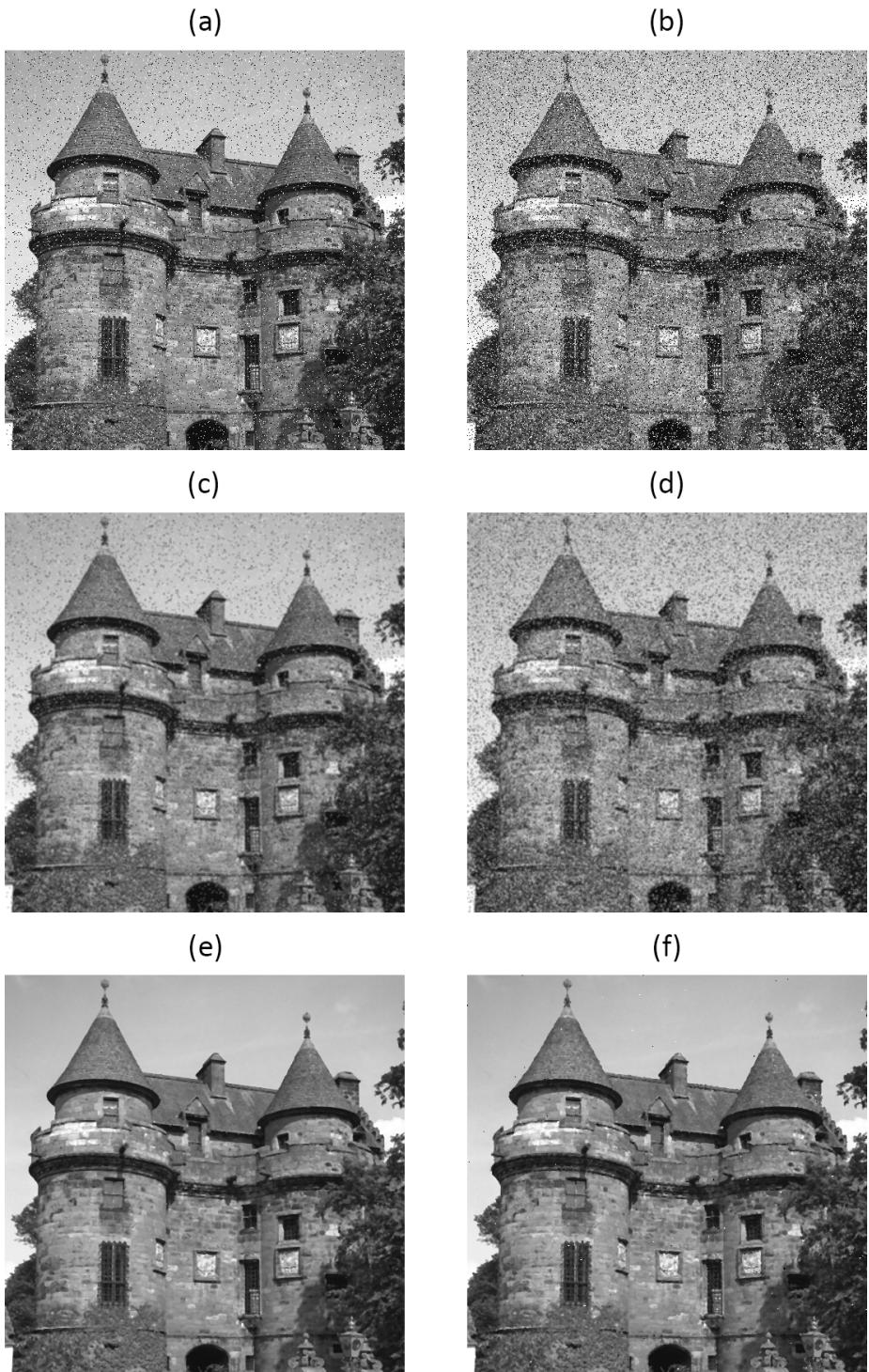


Figura 6: Exemplo de redução de ruído 'Sal e Pimenta' com filtros espaciais: imagens originais com 5% (a) e 10% (b) de pixels contaminados; resultados da aplicação de filtros de média (c,d) e de mediana (e,f), com janela de  $3 \times 3$ .

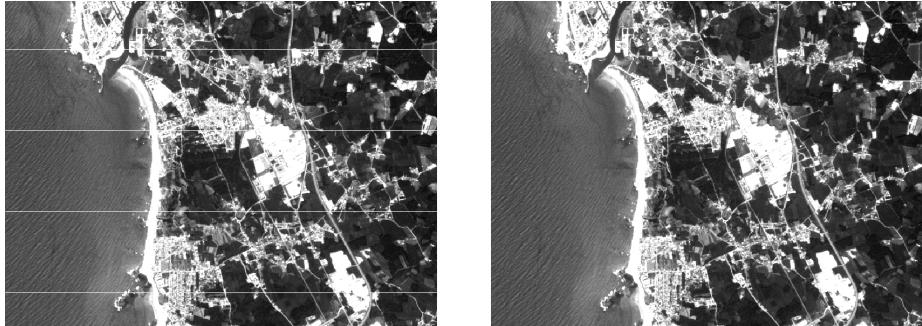


Figura 7: Imagem de satélite com linhas em falta (esquerda) e após correção com um filtro espacial linear (direita).

(a)	(b)	(c)	(d)																																				
<table border="1"> <tr><td>1</td><td>1</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	1	1	1	0	0	0	1	1	1	<table border="1"> <tr><td>1</td><td>2</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>2</td><td>1</td></tr> </table>	1	2	1	0	0	0	1	2	1	<table border="1"> <tr><td>1/6</td><td>1/6</td><td>1/6</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>1/6</td><td>1/6</td><td>1/6</td></tr> </table>	1/6	1/6	1/6	0	0	0	1/6	1/6	1/6	<table border="1"> <tr><td>1/8</td><td>1/4</td><td>1/8</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>1/8</td><td>1/4</td><td>1/8</td></tr> </table>	1/8	1/4	1/8	0	0	0	1/8	1/4	1/8
1	1	1																																					
0	0	0																																					
1	1	1																																					
1	2	1																																					
0	0	0																																					
1	2	1																																					
1/6	1/6	1/6																																					
0	0	0																																					
1/6	1/6	1/6																																					
1/8	1/4	1/8																																					
0	0	0																																					
1/8	1/4	1/8																																					

Figura 8: Filtros espaciais lineares de  $3 \times 3$  (a,b) e correspondentes versões normalizadas (c,d) para correção de linhas em falta.

## 4.2 Linhas em falta

Um outro tipo de ruído que pode ser removido com filtros espaciais é a existência de linhas em falta, que ocorre por exemplo em imagens recolhidas por satélites de observação da Terra. A figura 7 (esquerda) mostra um exemplo de uma imagem com uma linha em falta a cada 150 linhas. A utilização de um filtro de mediana poderia atenuar este problema, tendo no entanto um impacto negativo nas zonas não contaminadas da imagem, e também por aumentar ligeiramente as intensidades nas imediações das linhas em falta. Uma forma mais satisfatória de eliminar este tipo de ruído é aplicar um operador como os indicados na figura 8, que correspondem a substituir cada pixel da linha em falta pela média dos 6 vizinhos não contaminados mais próximos. No primeiro caso (figura 8c) é considerada uma média simples, e no segundo caso (figura 8d) uma média ponderada, onde os 2 pixels mais próximos (o de cima e o de baixo) tem um peso maior. Uma versão ainda mais simples seria ter apenas dois valores não nulos (0.5) nos vizinhos de cima e de baixo, o que corresponde a substituir a linha em falta pela média das linhas mais próximas (de cima e de baixo). A figura 7 (direita) mostra o resultado da aplicação do operador de média simples (figura 8c) apenas às linhas contaminadas na imagem original.

## 5 Deteção de contornos

Numa imagem podem ser identificados contornos, que correspondem às zonas da imagem onde a intensidade varia de forma abrupta. Por exemplo, a transição entre a praia (quase branco) e mar (cinzento escuros) na zona da esquerda da imagem de satélite de Árvore, Vila do Conde (figura 7), ou entre o céu e o telhado do Palácio de Falkland (figura 5a).

### 5.1 Contornos simples

Considerando uma imagem como uma função  $f(x,y)$  de variável contínua, onde  $x,y$  é a coordenada no plano (linha,coluna) e o valor  $f(x,y)$  é a intensidade de cinzento, é possível definir o Gradiente ( $\nabla$ ) através de (3), sendo  $\mathbf{i}$  e  $\mathbf{j}$  os vetores unitários nas direções linha / coluna. Pode igualmente definir-se o Laplaciano ( $\nabla^2$ ), através de (4).

$$\nabla f(x,y) = \frac{\partial f(x,y)}{\partial x} \mathbf{i} + \frac{\partial f(x,y)}{\partial y} \mathbf{j} \quad (3)$$

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \quad (4)$$

No caso de imagens digitais, as varáveis são discretas, pelo que estas operações tem de ser modificadas. É possível calcular versões simplificadas da 2<sup>a</sup> derivada (Laplaciano), considerando as variações no menor elemento (pixel) de uma imagem digital, em x (5) e y (6).

$$\frac{\partial^2 f(x,y)}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y) \quad (5)$$

$$\frac{\partial^2 f(x,y)}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y) \quad (6)$$

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \quad (7)$$

Usando as equações (5) e (6) em (4), resulta na equação (7), que pode ser usada para implementar o operador Laplaciano como um filtro espacial linear de  $3 \times 3$ . Este filtro, apresentado na figura 9(c), substitui o valor de cada pixel pela soma das diferenças para os 4-vizinhos mais próximos. Uma outra versão do operador Laplaciano, apresentada na figura 9(d), considera a soma das diferenças para os 8-vizinhos mais próximos.

Na figura 9 são apresentados igualmente os operadores Roberts (a,b), de apenas  $2 \times 2$  elementos. Os operadores Roberts e Laplaciano são detetores de contornos, tendo valores baixos (próximos de 0) em zonas da imagem uniformes, e valores absolutos altos (positivos ou negativos) em zonas com diversidade de intensidades. Os operadores Laplaciano, com 4 ou 8 vizinhos, são em geral preferidos aos operadores Roberts, uma vez que estes por considerarem apenas a diferença entre 2 pixels, estão muito susceptíveis a ruído e pequenas variações de intensidade na imagem.

Um aspeto a ter em conta é o formato dos dados, uma vez que o resultado da aplicação dos operadores Laplaciano, Roberts, ou outros detetores de contornos,

(a)	(b)	(c)	(d)																										
<table border="1"> <tr><td>1</td><td>0</td></tr> <tr><td>0</td><td>-1</td></tr> </table>	1	0	0	-1	<table border="1"> <tr><td>0</td><td>1</td></tr> <tr><td>-1</td><td>0</td></tr> </table>	0	1	-1	0	<table border="1"> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>-4</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> </table>	0	1	0	1	-4	1	0	1	0	<table border="1"> <tr><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>-8</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	1	1	1	1	-8	1	1	1	1
1	0																												
0	-1																												
0	1																												
-1	0																												
0	1	0																											
1	-4	1																											
0	1	0																											
1	1	1																											
1	-8	1																											
1	1	1																											

Figura 9: Filtros espaciais para deteção de contornos: operadores Roberts (a,b) e Laplaciano (c,d).

dão origem a valores positivos ou negativos (ou 0). Para imagens em formatos de valores positivos, por exemplo 8 bits (0-255) ou 16 bits sem sinal (0-65535), os valores negativos são truncados (passam a 0).

A aplicação de operadores Laplaciano é ilustrada na figura 10, usando a imagem de teste 'Falkland' (a), de 8 bits (0-255) com  $512 \times 512$  pixels. A aplicação do operador Laplaciano com 4-vizinhos, apresentado na figura 9(c), dá origem à imagem da figura 10(b). A imagem é muito escura, uma vez que a maior parte dos pixels tem valores 0 ou muito baixos (em zonas homogéneas), e os valores negativos são truncados para 0. A utilização do mesmo operador mas com sinais trocados (1 por -1 e -4 por 4) dá origem à imagem da figura 10(c). Apesar destas duas imagens serem visualmente parecidas, a verdade é que os pixels com valores positivos numa destas imagens tem valor 0 na outra. Como em geral pretende-se detetar contornos sem preocupação sobre o sinal (positivo ou negativo), as duas imagens (b,c) podem ser combinadas numa só, através da soma das 2 imagens, o que corresponde ao valor absoluto (módulo) da operação Laplaciano (figura 9d). Uma outra alternativa é apresentar o resultado com a origem (0) deslocada para um valor intermédio na escala de cinzento, sendo os valores positivos mais claros e os negativos mais escuros do que este nível de referência. A figura 9(e) mostra o resultado do operador Laplaciano com 4-vizinhos apresentado desta forma. Por último, a figura 10(f) apresenta o resultado (módulo) do operador Laplaciano com 8-vizinhos. Comparando esta imagem com o resultado para o operador com 4-vizinhos (d), é possível verificar que os contornos aparecem mais destacados ou perceptíveis.

É possível observar nos resultados da figura 10, em particular nas imagens (d) e (f), que os detectores de contornos são extremamente sensíveis a pequenas variações de intensidade. Para além dos contornos mais relevantes da imagem original (ex. edifício/céu, janelas, etc.), são detetadas pequenas variações de intensidade, por exemplo na superfície do telhado, ou nas paredes do edifício.

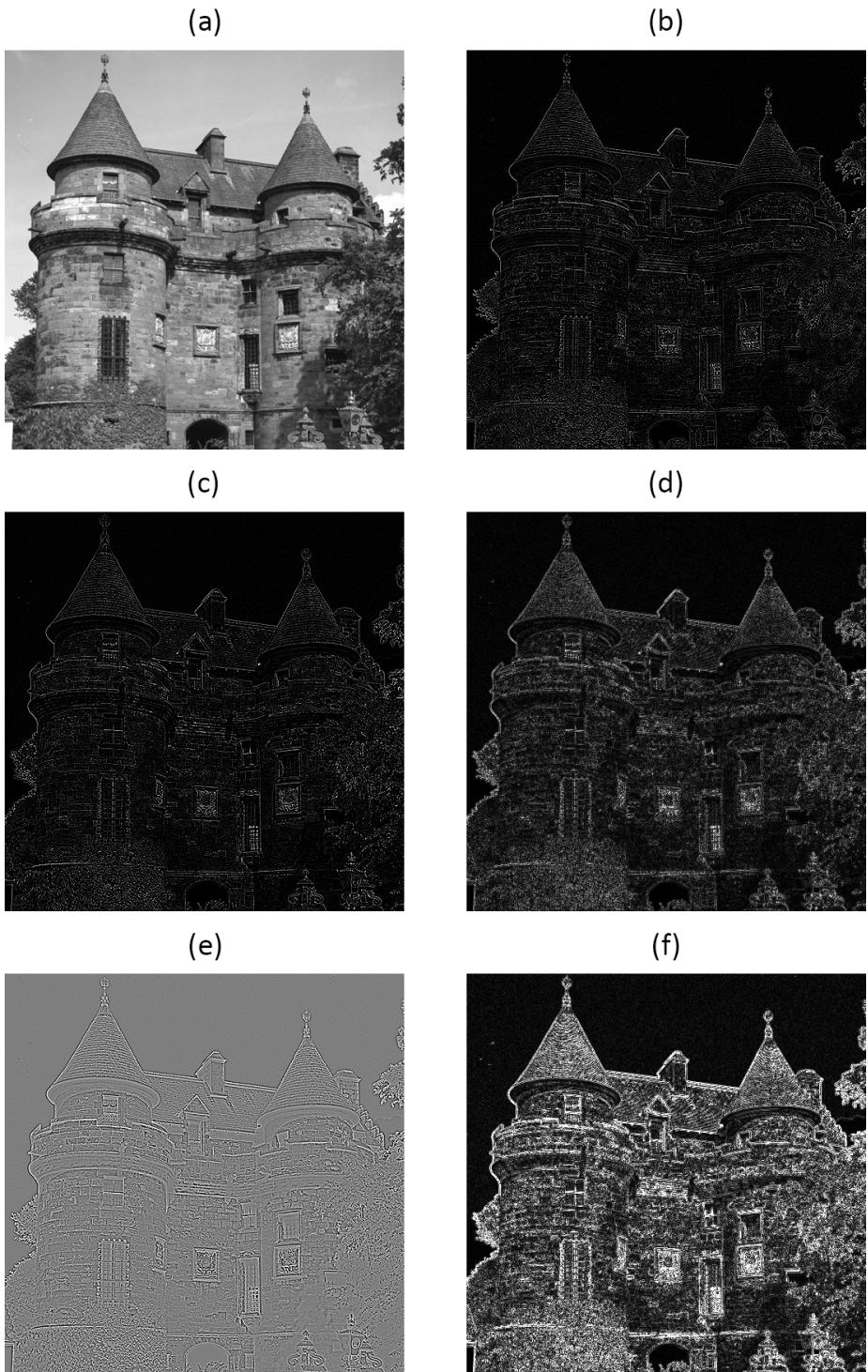


Figura 10: Aplicação de operadores Laplaciano à imagem do Palácio de Falkland  
 (a): operadores 4-vizinhos simples (b, c); origem deslocada para incluir valores positivos e negativos (e); valores absolutos para 4-vizinhos (d) e 8-vizinhos (f).

(a)	(b)	(c)	(d)																																				
<table border="1"> <tr><td>1</td><td>1</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>-1</td><td>-1</td><td>-1</td></tr> </table>	1	1	1	0	0	0	-1	-1	-1	<table border="1"> <tr><td>-1</td><td>0</td><td>1</td></tr> <tr><td>-1</td><td>0</td><td>1</td></tr> <tr><td>-1</td><td>0</td><td>1</td></tr> </table>	-1	0	1	-1	0	1	-1	0	1	<table border="1"> <tr><td>-1</td><td>-1</td><td>-1</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	-1	-1	-1	0	0	0	1	1	1	<table border="1"> <tr><td>1</td><td>0</td><td>-1</td></tr> <tr><td>1</td><td>0</td><td>-1</td></tr> <tr><td>1</td><td>0</td><td>-1</td></tr> </table>	1	0	-1	1	0	-1	1	0	-1
1	1	1																																					
0	0	0																																					
-1	-1	-1																																					
-1	0	1																																					
-1	0	1																																					
-1	0	1																																					
-1	-1	-1																																					
0	0	0																																					
1	1	1																																					
1	0	-1																																					
1	0	-1																																					
1	0	-1																																					
(e)	(f)	(g)	(h)																																				
<table border="1"> <tr><td>1</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>-1</td></tr> <tr><td>0</td><td>-1</td><td>-1</td></tr> </table>	1	1	0	1	0	-1	0	-1	-1	<table border="1"> <tr><td>0</td><td>-1</td><td>-1</td></tr> <tr><td>1</td><td>0</td><td>-1</td></tr> <tr><td>1</td><td>1</td><td>0</td></tr> </table>	0	-1	-1	1	0	-1	1	1	0	<table border="1"> <tr><td>1</td><td>2</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>-1</td><td>-2</td><td>-1</td></tr> </table>	1	2	1	0	0	0	-1	-2	-1	<table border="1"> <tr><td>-1</td><td>0</td><td>1</td></tr> <tr><td>-2</td><td>0</td><td>2</td></tr> <tr><td>-1</td><td>0</td><td>1</td></tr> </table>	-1	0	1	-2	0	2	-1	0	1
1	1	0																																					
1	0	-1																																					
0	-1	-1																																					
0	-1	-1																																					
1	0	-1																																					
1	1	0																																					
1	2	1																																					
0	0	0																																					
-1	-2	-1																																					
-1	0	1																																					
-2	0	2																																					
-1	0	1																																					

Figura 11: Filtros espaciais para deteção de contornos direcionais: operadores Prewitt horizontais (a,c), verticais (b,d) e obliquos (e,f); operadores Sobel horizontal (g) e vertical (h).

## 5.2 Contornos direcionais

Uma alternativa aos operadores de contornos descritos em (5.1) é usar detetores de contornos direcionais, sendo os operadores Prewitt, de  $3 \times 3$  elementos, dos mais usados. Na figura 11 são apresentados operadores Prewitt horizontais (a,c), verticais (b,d) e obliquos (e,f). À semelhança dos operadores Laplacianos, a aplicação dos operadores Prewitt dá origem a valores positivos e negativos, pelo que tem de ser tida em conta a estrutura de dados da imagem. Os operadores Sobel são também bastante usados para a deteção de contornos direcionais, sendo apenas ligeiramente diferentes dos operadores Prewitt. A figura 11 mostra um operador Sobel de  $3 \times 3$  horizontal (g) e um vertical (h).

O funcionamento dos operadores Prewitt é ilustrado através de um exemplo com uma pequena imagem com 35 pixels ( $5 \times 7$ ), de 8-bits sem sinal (figura 12a). Na verdade esta é uma pequena secção de uma imagem maior, com apenas 3 níveis de cinzento utilizados (30, 80 e 140). A imagem tem uma zona com valores baixos (30) no canto superior direito, uma zona com valores altos (140) no canto superior esquerdo, e uma zona de valores intermédios (80) na parte de baixo. As 3 zonas formam entre si fronteiras, ou contornos, com características distintas: um contorno vertical entre as regiões de intensidade 30 e 140, um contorno horizontal entre as regiões de intensidade 30 e 80, e um contorno obliquio, a  $45^\circ$  em relação às linhas, entre as regiões de intensidade 80 e 140.

Inicialmente foi aplicado o operador Prewitt horizontal apresentado na figura 11(c), à imagem original. Em zonas uniformes, o resultado desta operação é 0. Por exemplo, para o pixel (2,2) o resultado é  $-140-140-140 +0+0+0 +140+140+140 = 0$ . Em zonas de contorno vertical, também se obtém 0. Por exemplo, para o pixel (2,5) o resultado é  $-140-30-30 +0+0+0 +140+30+30 = 0$ . Por outro lado, em zonas de contorno horizontal obtém-se um valor diferente de 0. Por exemplo, para o pixel (3,6) o valor obtido é  $-30-30-30 +0+0+0 +80+80+80 = 150$ . O resultado da aplicação deste operador é apresentado na

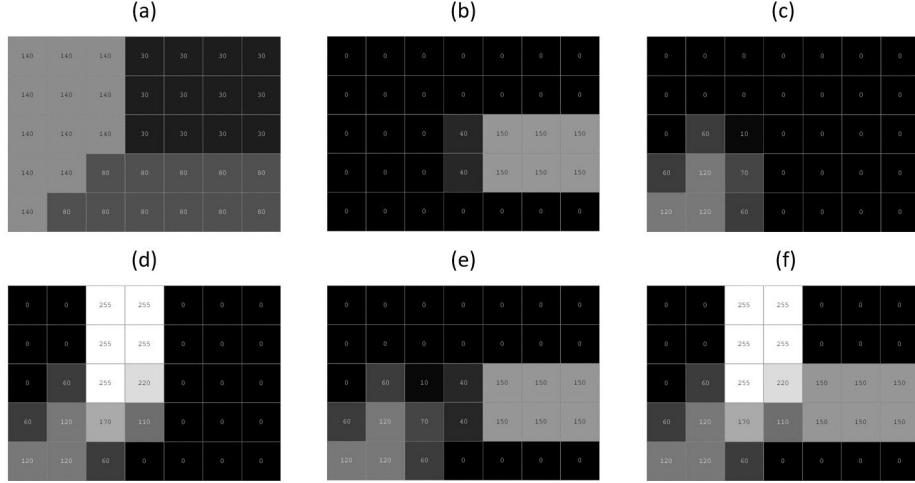


Figura 12: Exemplo de aplicação de filtros espaciais para deteção de contornos direcionais: imagem original (a), resultado de cada operador Prewitt horizontal (b,c), contornos verticais (d), horizontais (e) e todos os contornos (f).

figura 12(b). Convém referir que não foi necessário considerar *padding*, uma vez que a imagem apresentada é uma secção de uma imagem maior, em que os contornos continuam com as mesmas características. Caso o resultado da aplicação do filtro espacial dê origem a valores negativos, eles são truncados para 0. Por exemplo, para o pixel (4,2) o resultado seria  $-140-140-140+0+0+0+140+80+80=-120$ , que é truncado a 0. A utilização do outro operador Prewitt horizontal (figura 11a), dá origem a uma outra imagem com contornos horizontais, apresentada na figura 12(c). Cada uma destas imagens apresenta os contornos horizontais da imagem original, com intensidades crescentes ou decrescentes num dado sentido. Somando as duas imagens obtém-se uma imagem com todos os contornos horizontais, sem distinção de sentido de aumento de intensidade, como a que é apresentada na figura 11(e).

No caso dos operadores Prewitt verticais, o operador da figura 11(b) não detecta qualquer pixel, tendo a imagem resultante todos os pixels a 0. O resultado obtido com o outro operador Prewitt vertical é apresentado na figura 12(d), correspondendo igualmente a todos os contornos verticais da imagem. Poderia também aplicar-se os operadores Prewitt obliquos (figura 11e,f), e as suas versões invertidas, caso se pretendesse encontrar contornos com essas direções. A partir de 2 imagens com contornos em direções perpendiculares, é possível obter uma imagem com todos os contornos, por exemplo através da operação máxima. O resultado para a sub-imagem de teste é apresentado na figura 12(f).

A utilização de filtros espaciais lineares para deteção de contornos direcionais é ilustrada usando a imagem de teste 'Falkland' (figura 9a). A figura 13 mostra o resultado da aplicação dos operadores Prewitt horizontais (a,c); verticais (b,d) e obliquos (e). É possível observar que os contornos dos telhados das torres aparecem apenas numa das imagens de contornos verticais. Os contornos do lado esquerdo em (d), correspondendo a um decréscimo de intensidade (céu - telhado) ao longo da linha, e os contornos do lado direito em (b).

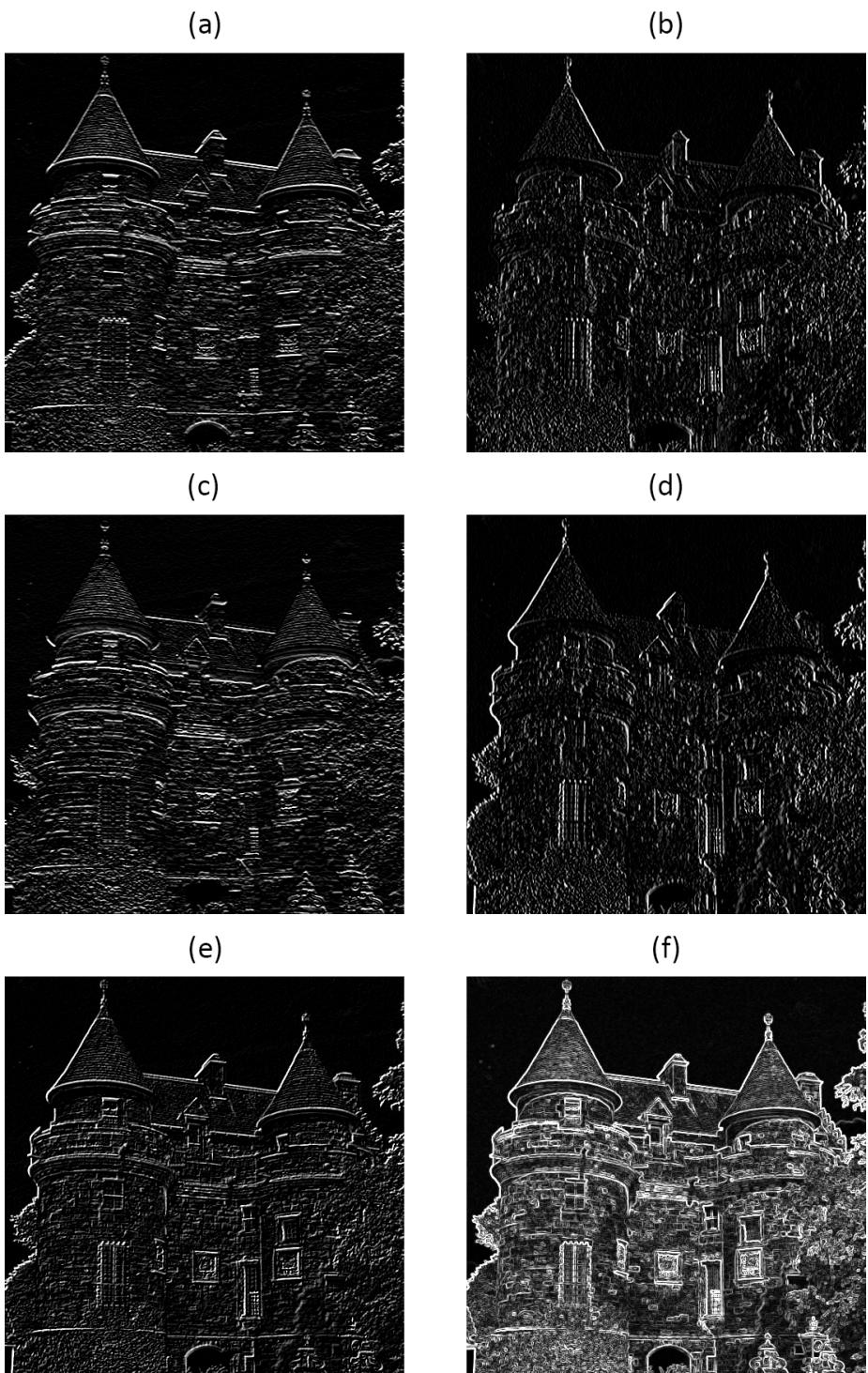


Figura 13: Deteção de contornos na imagem Falkland com operadores Prewitt horizontais (a,c); verticais (b,d) e obliquos (e); imagem com todos os contornos (f) obtida a partir de (a-d).



Figura 14: Deteção de contornos na imagem Falkland com operadores Prewitt horizontais (a,c); verticais (b,d) e obliquos (e); imagem com todos os contornos (f) obtida a partir de (a-d).

A figura 13(f) mostra a imagem com todos os contornos, obtida a partir das 4 imagens de contornos horizontais e verticais, através da operação máxima. Comparando esta imagem de contornos com a obtida pelos operadores Laplaciano (figura 10d,f), é possível observar que as linhas mais importantes, por exemplo os contornos exteriores do edifício, aparecem melhor destacados.

### 5.3 Detetores de cantos

Um caso particular de contornos que tem algum interesse é a deteção de cantos. Isso pode ser feito combinando resultados de operadores direcionais perpendiculares, ou através de filtros especificamente desenvolvidos para esse efeito. Um dos mais utilizados é o detetor de cantos Harris.

A figura 14 mostra um exemplo do uso do detetor de cantos de Harris numa imagem de um dos edifícios da FCUP. No total foram detetados 589 pontos de interesse (possíveis cantos) nesta imagem, sendo apresentados os 20 (esquerda) e os 200 (direita) mais importantes, como cruzes a verde nas imagens da figura 14.

## 6 Realce de contornos

Em certos casos não se pretende identificar ou detetar os contornos, mas sim realçá-los, de forma a permitir uma maior facilidade de interpretação visual da imagem. Uma forma de fazer realce de contornos consiste em aplicar um detetor de contornos, e somar a imagem resultante (com valores positivos e negativos) à imagem original. Desta forma, as zonas uniformes mantém-se inalteradas, e as zonas com contornos ficam com as diferenças aumentadas. Uma outra abordagem usa o facto do filtro de média atenuar os contornos. Inicialmente é calculada a diferença entre a imagem original e uma versão suavizada com um filtro de média. Esta imagem de diferenças, que tem valores não nulos (positivos e negativos) próximo das zonas de contornos, é somada à original, produzindo uma versão da imagem com contornos realçados.

A figura 14 mostra uma imagem da lua (a), cujos contornos foram realçados com um filtro Laplaciano (b) e via filtro de média de  $5 \times 5$  (c) e  $7 \times 7$  (d).

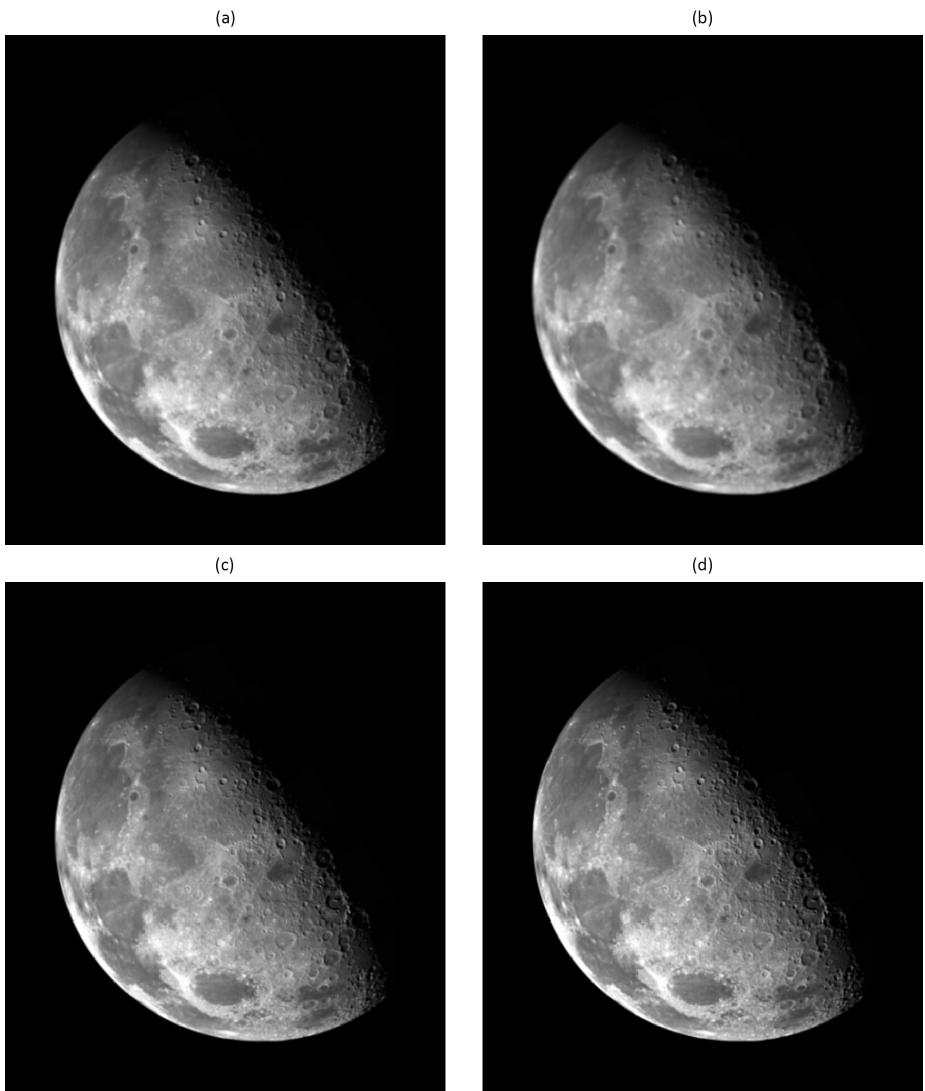


Figura 15: Exemplo de realce de contornos: imagem original (a); realce com filtro Laplaciano (b) e via filtro de média de  $5 \times 5$  (c) e  $7 \times 7$  (d).

## 7 Filtros espaciais no MATLAB

Nesta seção são apresentados alguns códigos MATLAB para a aplicação de filtros espaciais e outras tarefas relacionadas, como a contaminação de uma imagem com ruído aditivo.

### 7.1 Filtros lineares

A aplicação de filtros espaciais lineares é em geral feita em 2 fases - primeiro é definido o filtro (matriz com coeficientes) e em seguida é aplicado à imagem. A definição do filtro pode ser feita simplesmente criando e atribuindo valores a uma matriz, ou através da função `fspecial`, que permite definir filtros *standard*. A aplicação do filtro é feita através da função `imfilter`.

#### 7.1.1 Definição do filtro

Apresentam-se em seguida alguns exemplos de definição de filtros.

```
>> hLF = [ 1 1 1; 0 0 0; 1 1 1] criação da matriz de coeficientes
hLF =
    1     1     1
    0     0     0
    1     1     1
>> hPh = fspecial('prewitt') tipo de filtro standard
hPh =
    1     1     1
    0     0     0
   -1    -1    -1
>> hAv3 = fspecial('average',3) tamanho do filtro
hAv3 =
    0.1111    0.1111    0.1111
    0.1111    0.1111    0.1111
    0.1111    0.1111    0.1111
```

Os restantes operadores Prewitt e Sobel podem ser obtidos através das operações elementares multiplicação (\*) e transposta (`transpose` ou `'`), como exemplificado em seguida.

```
>> hSOBELh1 = fspecial('sobel');
>> hSOBELh2 = - hSOBELh1 trocar o sinal
    -1    -2    -1
     0     0     0
     1     2     1
>> hSOBELv1 = hSOBELh1' transposta
     1     0    -1
     2     0    -2
     1     0    -1
>> hSOBELv2 = - hSOBELv1;
```

Os operadores Laplaciano, descritos em [5.1](#), podem ser definidos atribuindo valores a uma matriz de  $3 \times 3$ , ou usando a função `fspecial`. No entanto, apenas

o operador da figura 9(c) pode ser obtido diretamente, podendo o operador da figura 9(d) obtido de forma indireta, como se exemplifica em seguida.

```
>> hL0 = fspecial('laplacian',0)
hL0 =
    0    1    0
    1   -4    1
    0    1    0
>> hL1 = fspecial('laplacian',1);
>> hL2 = hL0+2*hL1;
hL2 =
    1    1    1
    1   -8    1
    1    1    1
```

parametro que controla a forma do laplaciano

### 7.1.2 Aplicação do filtro

Uma vez definido o filtro, através da matriz com coeficientes ( $h$ ), a aplicação é feita usando a função `imfilter`, na forma mais simples como

```
>> IMf = imfilter(IMo,h);
```

imagem filtrada

Nesta forma de utilização da função `imfilter`, com apenas 2 parâmetros de *input*, não foi indicado qual o *padding* a usar, pelo que é considerada uma margem com valores 0 (*default*). Em alternativa pode ser colocado o tipo de *padding* como 3º parâmetro de *input*, por exemplo

```
>> IMf = imfilter(IMo,h,'symmetric');
```

padding simétrico

As outras possibilidades para o *padding* são '`'replicate'`', '`'circular'`' e '`'C'`', onde C é um valor numérico.

Há outros características da função `imfilter` que podem ser controladas através de parâmetros de *input* adicionais. Por omissão (*default*), o processo de filtragem espacial é implementado através da correlação, mas pode ser alterado para convolução usando '`conv`'. O tamanho da imagem final (*output*) pode ou não ser igual à imagem original. No 1º caso, a margem criada para calcular os valores próximos das margens é eliminanda, enquanto que no 2º caso é mantida, sendo os parâmetros '`'same'`' (*default*) e '`'full'`' usados para controlar este factor.

## 7.2 Filtros não-lineares

Existe uma função para aplicar um filtro de mediana a um vetor (`median`). No entanto, a função não pode ser aplicada a matrizes, pelo que para imagens deve ser usada a função `medfilt2`. A função pode ter apenas 1 parâmetro de *input*, a imagem a filtrar, sendo nesse caso usado um *padding* com zeros (*default*). Caso se pretenda escolher o *padding*, deverá ser indicado no 2º parâmetro de *input*. Em seguida alguns exemplos de utilização.

```
>> IMf = medfilt2(IMo);
>> IMf = medfilt2(IMo,'symmetric');
>> IMf = medfilt2(IMo,255);
```

padding 0

padding simétrico

padding 255

Os filtros de ordem são aplicados através da função `ordfilt2`, que recebe 3 parâmetros de *input* na sua forma de utilização habitual: a imagem a filtrar, a ordem (numa sequência crescente) e uma matriz com a vizinhança. A função `true` é normalmente usada para definir uma região retangular, sendo semelhante à função `ones` mas criando uma matriz com variável do tipo lógico (1-bit). Em seguida apresentam-se alguns exemplos de utilização da função `ordfilt2`.

```
>> IM1 = ordfilt2(IM, 2, true(3));
>> IM2 = ordfilt2(IM, 1, true(3,5));
>> IM3 = ordfilt2(IM, 1, true(5), 'symmetric');
>> IM4 = ordfilt2(IM, 25, true(5));
```

A ordem considerada é numa sequência crescente dos vizinhos, pelo que o mínimo corresponde à ordem 1 e o máximo será o número de elementos da vizinhança, 25 no caso de IM4. A função `ordfilt2` aceita apenas 2 tipos de *padding*: com zeros (*default*) ou '`symmetric`'.

Surpreendentemente não há uma função para o filtro de moda implementada no MATLAB, sendo considerada como proposta de exercício (8).

### 7.3 Contaminação com ruído

O último ponto que se inclui nesta secção é a contaminação de uma imagem com ruído, uma vez que poderá ser útil experimentar o desempenho de vários filtros espaciais em imagens com ruído. A contaminação de uma imagem pode ser feita através da função `imnoise`, para diferentes tipos de ruído, como por exemplo sal & pimenta, Gaussiano, e *speckle*.

O ruído sal & pimenta é caracterizado por ocorrerem valores extremos (por exemplo 0 e 255 em imagens a 8 bits) localizados de forma aleatória na imagem. A sintaxe é a seguinte, sendo indicado um valor numérico para a fração de pixels contaminados (por *default* é 0.05, ou 5%).

```
>> IR1 = imnoise(IM, 'salt & pepper');
>> IR2 = imnoise(IM, 'salt & pepper', 0.1);
```

O ruído aditivo Gaussiano é caracterizado por 2 parâmetros: a média e a variância. Por omissão (*default*), a média é 0 e a variância 0.01. Alguns exemplos de contaminação de uma imagem com ruído Gaussiano:

```
>> IR3 = imnoise(IM, 'gaussian');
>> IR4 = imnoise(IM, 'gaussian', 0.1, 0.2);
```

O ruído multiplicativo do tipo *speckle* é caracterizado pela equação  $J=I+n*I$ , onde I e J são as imagens inicial e final, e n é ruído com distribuição uniforme com média 0 e variancia v (*default* 0.05). Alguns exemplos:

```
>> IR5 = imnoise(IM, 'speckle');
>> IR6 = imnoise(IM, 'speckle', 0.1);
```

## 8 Exercícios propostos

Nesta secção são propostos alguns exercícios para MATLAB que poderão ser usados para consolidar os conceitos de Processamento de Imagem apresentados.

### I - Filtros espaciais em imagens com ruído Sal & Pimenta

Crie uma função MATLAB que receba como *input* o nome de um ficheiro imagem (formato standard) e o nível de contaminação por ruído Sal & Pimenta (valor entre 0 e 1), e apresente uma figura com as seguintes imagens (num **subplot** de  $2 \times 3$ ):

- imagem original (IO)
- imagem contaminada com ruído Sal & Pimenta (IR)
- aplicação de filtros de média de  $3 \times 3$  e  $5 \times 5$  à imagem IR
- aplicação de filtros de mediana de  $3 \times 3$  e  $5 \times 5$  à imagem IR

Deverá igualmente ser apresentado no gráfico (títulos) uma medida quantitativa de erro para cada imagem (IR e imagens filtradas) em relação a IO, que poderá por exemplo ser o erro médio absoluto.

Exemplo de utilização: `NomeFuncao('RobotC.tif', 0.15)`

### II - Avaliação dos filtros espaciais para redução ruído

Escreva um script para avaliar de forma quantitativa o desempenho dos filtros de média e mediana em imagens contaminadas por ruído Sal & Pimenta e Gaussiano. Poderá usar imagens reais (ex. 'RobotC.tif') ou sintéticas (ex. Tabuleiro de Xadrez com iluminação variável), com vários níveis de contaminação e diferentes tamanhos de filtro. Os resultados deverão ser apresentados de uma forma clara e concisa.

### III - Filtro de Moda

Crie uma função MATLAB para aplicar um filtro de Moda a uma imagem *grayscale* (1 banda). Numa primeira versão, a função deverá receber 2 *inputs* - a imagem e o tamanho da janela do filtro (3 para  $3 \times 3$ , k para  $k \times k$ ), e ter com output a imagem filtrada. Nesta versão poderá considerar um *padding* com valores constantes iguais a 0. Uma versão mais sofisticada terá com 3º *input* o tipo de padding (0, C ou 'symmetric'), à semelhança de outros filtros do MATLAB.

Exemplo de utilização:

```
v1 - IMnova=FiltroModa(IM,5);  
v2 - IMnova=FiltroModa(IM,3,'symmetric');  
v2 - IMnova=FiltroModa(IM,3,10);
```

## 8.1 APPENDIX - Exercices (in English)

This Appendix proposes some exercises for MATLAB that can be used to consolidate the concepts of Image Processing - Spatial Filtering.

### I - Spatial filters in images with Salt & Pepper noise

Create a MATLAB function that receives as input an image file name (standard format) and the level of contamination by Salt & Pepper noise (value from 0 to 1), and presents a figure with the following images (in a  $2 \times 3$  subplot):

- original image (IO)
- image contaminated by Salt & Pepper noise (IN)
- result of the application of average filters of  $3 \times 3$  and  $5 \times 5$  to IN
- result of the application of median filters of  $3 \times 3$  and  $5 \times 5$  to IN

Quantitative error measurements should also be presented for each image (IN and filtered images), in the graph (titles), in relation to IO. For example the absolute mean error.

Example of use: `TestFilters('RobotC.tif', 0.15)`

### II - Evaluation of spatial filters for noise reduction

Write a script to quantitatively evaluate the performance of the average and median filters in images contaminated by Salt & Pepper noise and Gaussian noise. You can use real (e.g. 'RobotC.tif') or synthetic (e.g. Chess board with variable lighting) images, with various levels of contamination and different filter sizes. The results should be presented in a clear and concise manner.

### III - Mode Filter

Create a MATLAB function to apply a mode filter to a grayscale image (1 band). In a first version, the function should receive 2 inputs - the image and the size of the filter window (3 for  $3 \times 3$ , k for  $k \times k$ ), and output the filtered image. In this version you can consider a padding with constant values equal to 0. A more sophisticated version will have as a 3rd input the padding type (0, C or 'symmetric'), as in other MATLAB filters.

Example of use:

```
v1 = IMnew=ModeFilter(IM,5);  
v2 = IMnew=ModeFilter(IM,3,'symmetric');  
v2 = IMnew=ModeFilter(IM,3,10);
```