Processamento de Imagem
6 - Operações Morfológicas

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 1.2 - 23 Março 2021

Conteúdo

1 Introdução

As operações morfológicas são usadas em processamento de imagem para tarefas de pré-processamento como redução de ruído e simplificação de formas, assim como para extração de informação sobre um objeto de interesse, como a área, perímetro e outras características relevantes. Numa primeira fase são consideradas operações morfológicas em imagens binárias, sendo depois os conceitos extendidos a imagens em escala de cinzento (1 banda) e coloridas (3 bandas).

Uma imagem pode ser definida como uma função I: $\Omega \rightarrow$ R, onde Ω é o conjunto de valores (x,y) correspondente à posição (linha,coluna), e R um conjunto de valores numéricos r (intensidades). Uma vez que se trata de uma imagem digital, as variáveis x,y,r são discretas, sendo habitual considerar inteiros não negativos para x e y. Ou seja, uma imagem digital é uma função I(x,y) com domínio Ω={1,2,...,N}×{1,2,...,M} (imagem de N×M pixels). No caso de uma imagem binária o contradomínio é o conjunto {0,1}, ou seja, cada pixel (x,y) da imagem toma o valor 1 ou 0 (Sim/Não ou ON/OFF).

Considerando a imagem binária I(x,y), é possível definir um conjunto A como A={ (x,y): I(x,y)=1 }. A é conjunto de pixels ON na imagem - objeto de interesse, enquanto que A^c={ (x,y): I(x,y)=0 } é o fundo.

2 Operadores elementares

Os dois operadores morfológicos elementares são Dilatação e Erosão, sendo os elementos base de outros operadores morfológicos mais elaborados. São considerados conjuntos A e E binários, sendo A o objeto e E o elemento estruturante. Em geral o elemento estruturante (E) é pequeno em comparação com o objeto (A), embora as definições sejam genéricas para quaisquer conjuntos A e E.

2.1 Dilatação

A dilatação de A por E (A$\oplus$E), ou soma de Minkowski, é definida por (1).

$$A \oplus E = \{(x+i, y+j) : (x,y) \in A; (i,j) \in E\} \tag{1}$$

O novo objeto, ou imagem binária, contém todos os pixels de A mais alguns que inicialmente não pertenciam a A, considerando que o elemento estruturante contém a origem (i,j)=(0,0). Na figura 1 é apresentado um exemplo da dilatação de um pequeno objeto (A), imagem do canto superior esquerdo, com um elemento estruturante (E) em forma de cruz (4-vizinhos). Para a aplicação da equação (1), considera-se que a origem do elemento estruturante, (i,j)=(0,0), é o elemento no centro da janela de 3×3 (figura 1, centro). Ou seja, o elemento estruturante contém os seguintes elementos (i,j)={ (0,-1); (-1,0); (0,0); (1,0); (0,1) }. O resultado dilatação de A por E (A$\oplus$E) é apresentado na figura 1, canto superior direito, podendo observar-se que a área do objeto é maior do que inicialmente, tendo aumentado 1 pixel para cima, baixo, esquerda e direita.

A aplicação da operação dilatação tem algumas semelhanças com o processo de filtragem espacial, na medida em que se pode usar uma janela deslizante, sendo possívelmente uma forma mais fácil de visualizar o impacto da dilatação em cada pixel da imagem (objeto) A. Colocando a janela que contém o elemento

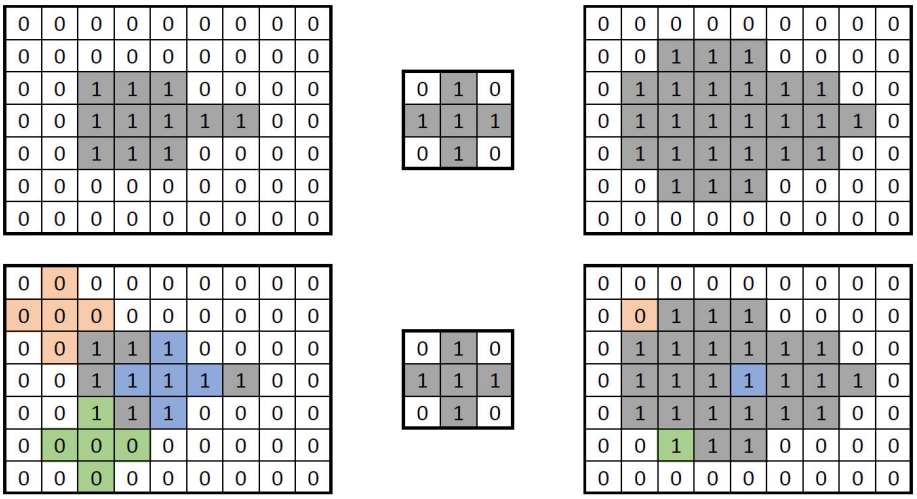

Figura 1: Imagem original A (esquerda), elemento estruturante E (centro) e resultado da operação dilatação A⊕E (direita).

estruturante (E) sobre um pixel em análise, o valor obtido para esse pixel será 1 desde que haja sobreposição entre um dos elementos de E com um elemento de A. No caso de todos os elementos de A cobertos por E serem 0, o pixel terá valor 0. A figura 1 (linha de baixo) mostra exemplos das várias situações possíveis: (1) um pixel com valor 1 em A que mantém o valor 1 após a dilatação (azul); (2) um pixel que passou de 0 para 1 (verde); e um pixel com valor 0 em A que mantém o valor 0 (laranja).

Uma outra forma de definir a dilatação é através da reunião de translações de A, considerando E. Ou seja, criam-se versões A_e deslocadas de A para cada elemento de E, e em seguida calcula-se a reunião (2).

$$A \oplus E = \bigcup_{e \in E} A_e \tag{2}$$

Para o exemplo da figura 1, considerando que o elemento estruturante tem 5 elementos, a dilatação pode ser calculada como a reunião da imagem binária correspondente ao objeto A com versões deslocadas 1 pixel para cima, baixo, esquerda e direita.

A dilatação é normalmente usada para tapar buracos e simplificar objetos, aumentando a sua área. A figura 2 mostra alguns exemplos de imagens obtidas por dilatação de uma imagem binária de uma águia (a), com 512×512 pixels. As imagens dilatadas apresentadas na figura 2 foram criadas com elementos estruturantes circulares de raio crescente: 3 pixels (b), 5 (c), 10 (d), 20 (e) e 30 pixels (f). À medida que o elemento estruturante aumenta, o objeto aumenta em área e torna-se mais simples no seu contorno.

A operação dilatação satisfaz algumas propriedades que podem ser úteis para a sua implementação computacional. É associativa, isto é A⊕(E⊕F)=(A⊕E)⊕F, comutativa - A⊕E=E⊕A, e invariante à translação.

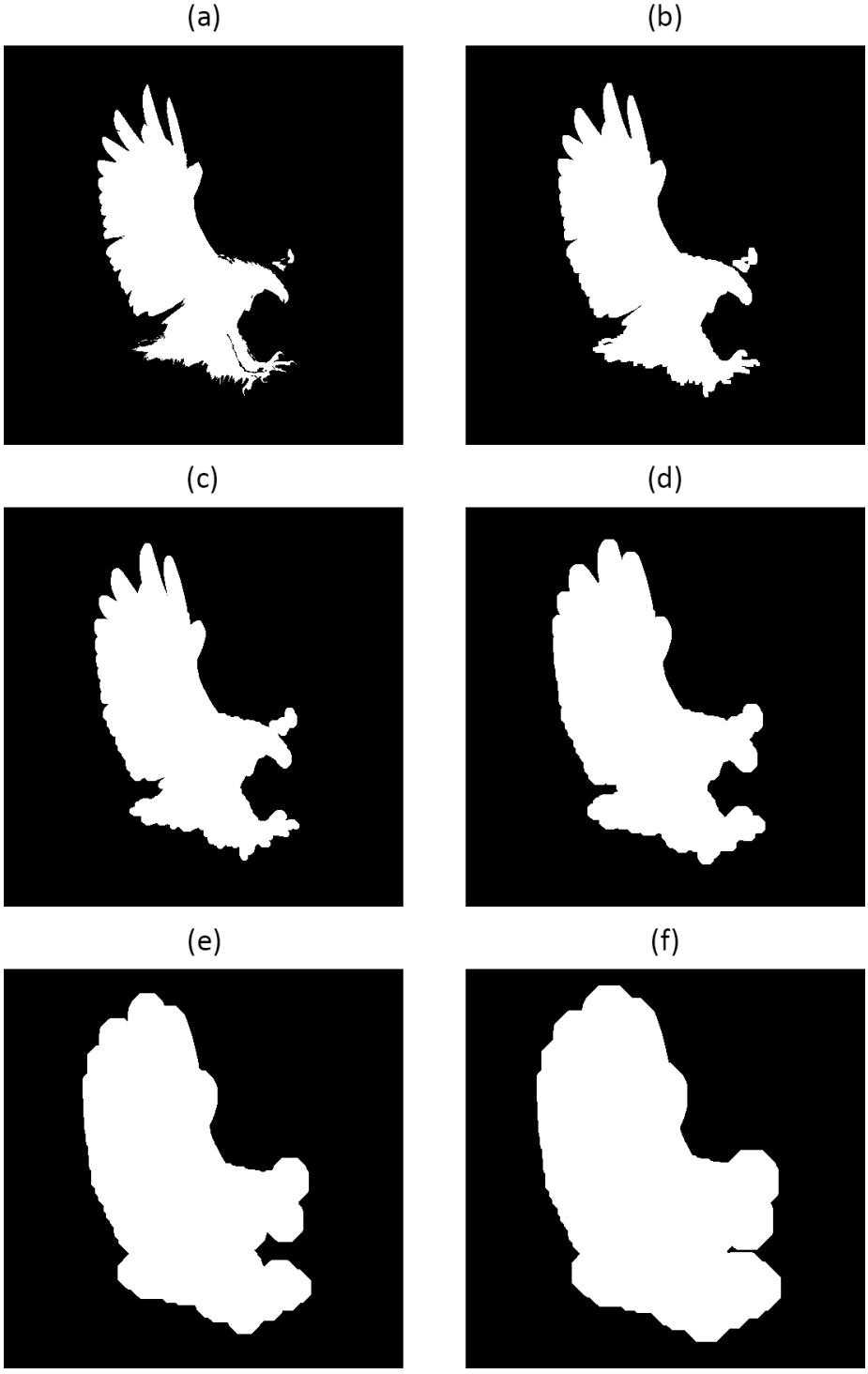

Figura 2: Exemplo de aplicação da operação dilatação a uma imagem binária de 512×512 pixels (a) com elemento estruturante circular de raio 3 (b), 5 (c), 10 (d), 20 (e) e 30 (f) pixels.

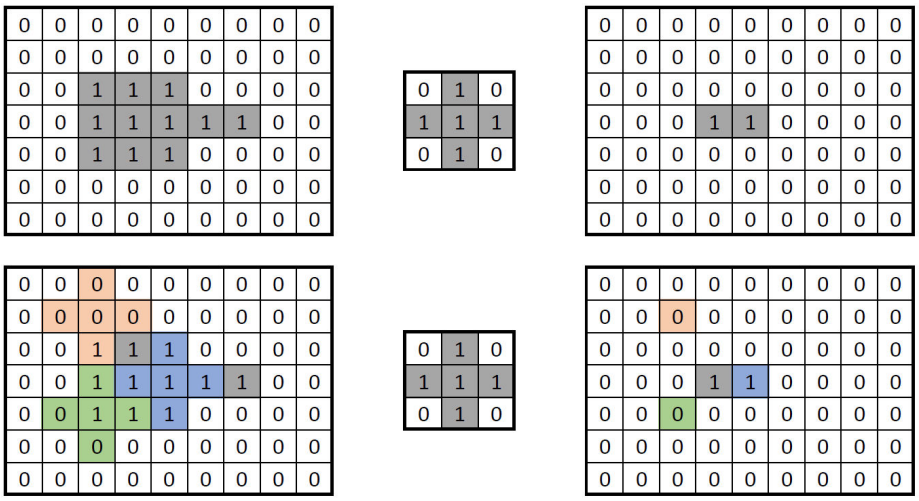

Figura 3: Imagem original A (esquerda), elemento estruturante E (centro) e resultado da operação dilatação A⊕E (direita).

2.2 Erosão

A erosão de A por E (A⊖E), ou subtração de Minkowski, é definida por (3).

$$A \ominus E = \{(x, y) : (x + i, y + j) \in A \quad \forall (i, j) \in E\} \tag{3}$$

O novo objeto, ou imagem binária, contém menos pixels do que A, excepto no caso extremo do elemento estruturante ter apenas 1 elemento. A figura 3 mostra um exemplo da dilatação de um pequeno objeto (A) com o elemento estruturante 4-vizinhos (E). O objeto que inicialmente tinha 11 elementos (pixels) foi reduzido a apenas 2 (figura 3, canto superior direito). Colocando o elemento estruturante (E) sobre um pixel em análise, o valor obtido para esse pixel será 1 apenas quando todos os pixels de A são 1 na zona de sobreposição. O processo é ilustrado na figura 1 (linha de baixo), onde são destacados 3 pixels. No primeiro caso (laranja), apenas 1 dos 5 pixels da zona de sobreposição entre A e E tem valor 1, pelo que o valor do pixel mantém-se a 0. No segundo caso (verde), 3 dos 5 pixels da zona de sobreposição entre A e E tem valor 1, havendo no entanto 2 pixels com valor 0 o que faz com que o valor a atribuir a esse pixel seja 0. Por último, é assinalado (a azul) um caso onde na zona de sobreposição não há qualquer 0 pelo que o valor obtido pela operação erosão é 1.

A erosão pode ser definida de uma outra forma, como a interseção de translações de A, considerando E. Ou seja, inicialmente criam-se versões A_e deslocadas de A para cada elemento de E, e depois calcula-se a interseção dessas imagens (4).

$$A \ominus E = \bigcap_{e \in E} A_{-e} \tag{4}$$

A erosão é normalmente usada para eliminar objetos isolados e pontas ou peninsulas, tendo no entanto o efeito de reduzir a área do objeto. A figura 4 mostra alguns exemplos de imagens obtidas por erosão da imagem binária da águia (a) com elementos estruturantes circulares de raio crescente: 3 pixels (b), 5 (c), 10 (d), 20 (e) e 30 pixels (f). À medida que o elemento estruturante

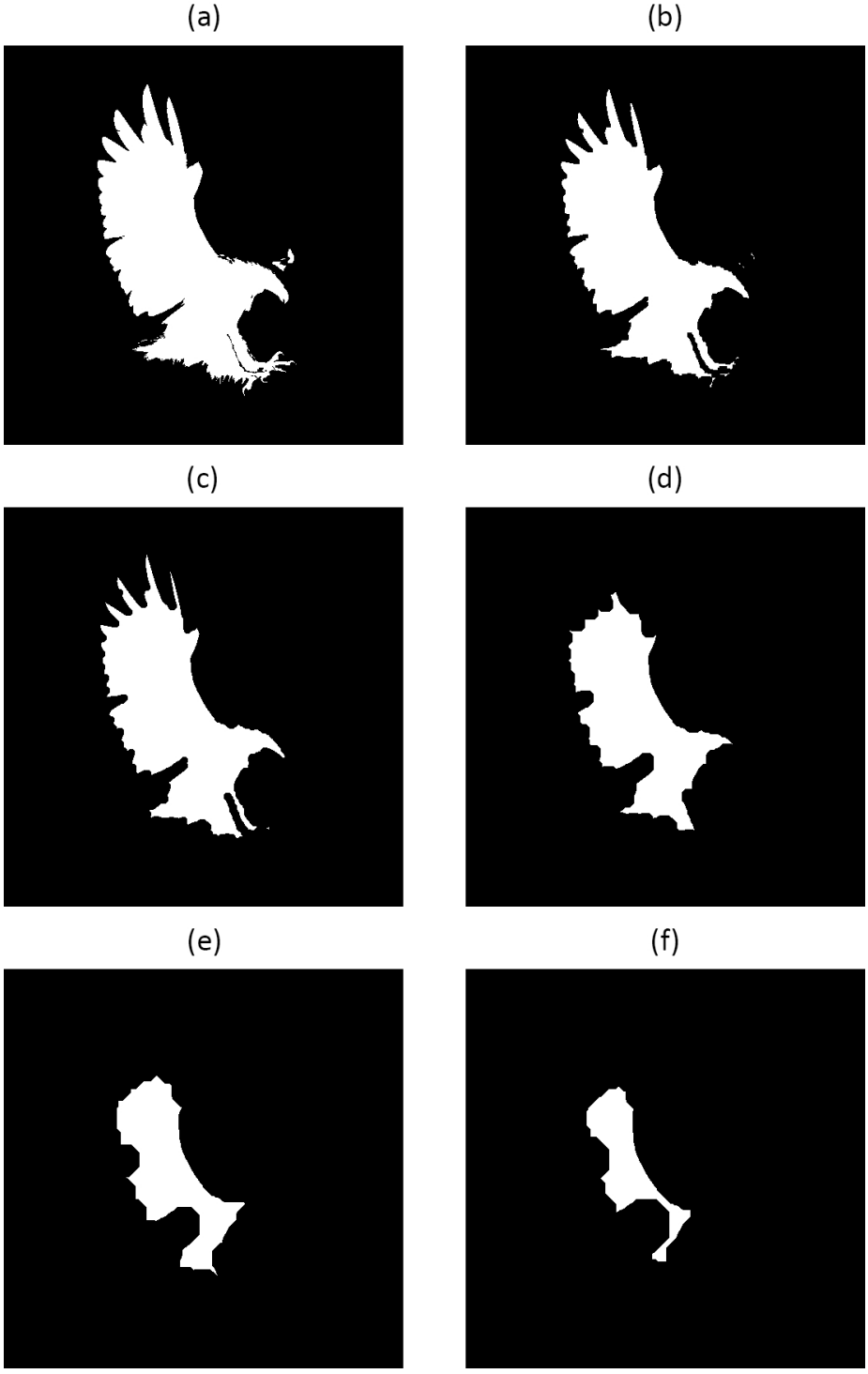

Figura 4: Exemplo de aplicação da operação erosão a uma imagem binária de 512×512 pixels (a) com elemento estruturante circular de raio 3 (b), 5 (c), 10 (d), 20 (e) e 30 (f) pixels.

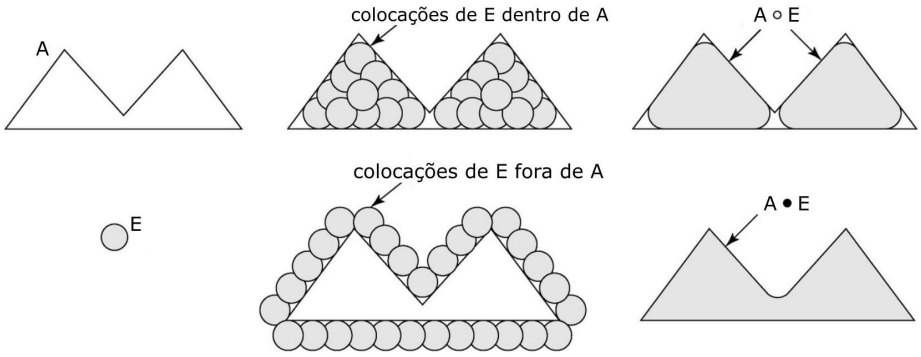

Figura 5: Representação esquemática das operações abertura (em cima) e fecho (em baixo) com elemento estruturante E, aplicadas a um objeto A.

aumenta, o objeto diminui em área, podendo eventualmente desaparecer. A operação erosão é invariante à translação mas, ao contrário da dilatação, não é comutativa nem associativa.

3 Abertura e fecho

A dilatação e a erosão são operações duais, não sendo no entanto operações inversas. Na verdade, a aplicação sucessiva destes operadores resulta numa versão simplificada do objeto (imagem binária), sendo relevante a ordem de aplicação dos operadores.

A operação abertura, de A por E (A∘E), conciste na aplicação de uma erosão seguida de uma dilatação (5) usando o elemento estruturante E. A operação fecho, de A por E (A•E), conciste na aplicação de uma dilatação seguida de uma erosão (6), usando o elemento estruturante E nas duas operações.

$$A \circ E = (A \ominus E) \oplus E \tag{5}$$

$$A \bullet E = (A \oplus E) \ominus E \tag{6}$$

O aplicação dos operadores abertura e fecho produz uma versão simplificada do objeto inicial, que depende da forma e tamanho do elemento estruturante. Em geral os resultados obtidos pelos operadores abertura e fecho são diferentes. O operador fecho é usado normalmente para tapar buracos e ligar zonas separadas, enquanto que o operador abertura é usado para eliminar pequenos objetos, peninsulas e outras extremidades.

O procedimento é ilustrade de forma esquemática na figura 5. A abertura de A por E (A∘E) conciste na colocação de E dentro de A em todas as posições possíveis, resultando numa versão igual ou menor a A em termos de área. No exemplo da figura 5 (em cima), perdem-se as zonas de cantos, uma vez que não é possivel colocar o elemento estruturante circular dentro de A de forma a conter os cantos. No caso da operação fecho de A por E (A•E), o resultado obtido é o complementar da região resultante de todas as colocação de E por fora de A figura 5 (em baixo). O objeto resultante da operação fecho é maior ou igual ao

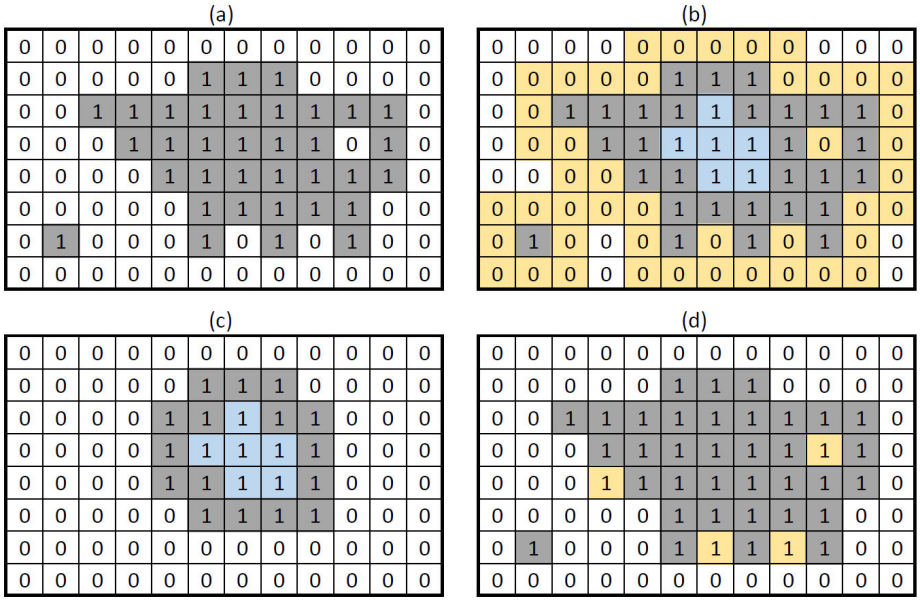

Figura 6: Exemplo de aplicação de operações abertura (c) e fecho (d) com elemento estruturante 8-vizinhos a uma imagem binária (a).

original. Neste caso é ligeiramente maior, devido a uma zona que não pode ser coberta por fora com o elemento estruturante circular.

A figura 6 mostra o efeito dos operadores abertura e fecho numa pequena imagem de teste (a), usando como elemento estruturante um quadrado de 3×3 pixels (8-vizinhos). Na figura 6(b) é apresentada a imagem original com os pixels resultantes da erosão representados a azul, e os pixels adicionais obtidos pela dilatação a amarelo. O resultado da operação abertura conciste na dilatação do objeto assinalado a azul, cujo resultado é apresentado na figura 6(c). Pode verificar-se que a nova imagem é uma versão simplificada da original, tendo-se perdidos os objetos soltos, extremidades e na verdade todas as zonas onde não é possível colocar um quadrado de 3×3. O resultado da operação fecho, que conciste na erosão do objeto assinalado a amarelo na figura 6(c), dá origem à imagem da figura 6(d). Pode observar-se que esta imagem é muito parecida com a original, havendo no entanto 4 novos pixels ON, assinalados a amarelo. A operação fecho tapou o buraco de 1 pixel (do lado direito) e cobriu a irregularidade na parte de baixo do objeto.

As figuras 7 e 8 mostram as imagens obtidas pela aplicação dos operadores abertura (figura 7) e fecho (figura 8) à imagem binária da águia (a), com elementos estruturantes circulares de raio crescente: 3 pixels (b), 5 (c), 10 (d), 20 (e) e 30 pixels (f). No caso do operador abertura (figura 7), à medida que o elemento estruturante aumenta vão-se perdendo objetos soltos (por exemplo a asa esquerda) e peninsulas (penas da asa direita), resultando num objeto simplificado e menor que o original. Os resultados para o operador fecho (figura 8) são bastante diferentes. À medida que o elemento estruturante aumenta os objetos soltos e peninsulas vão-se juntando, resultando num único objeto, maior que o original e com uma forma mais simples (contornos menos recortados).

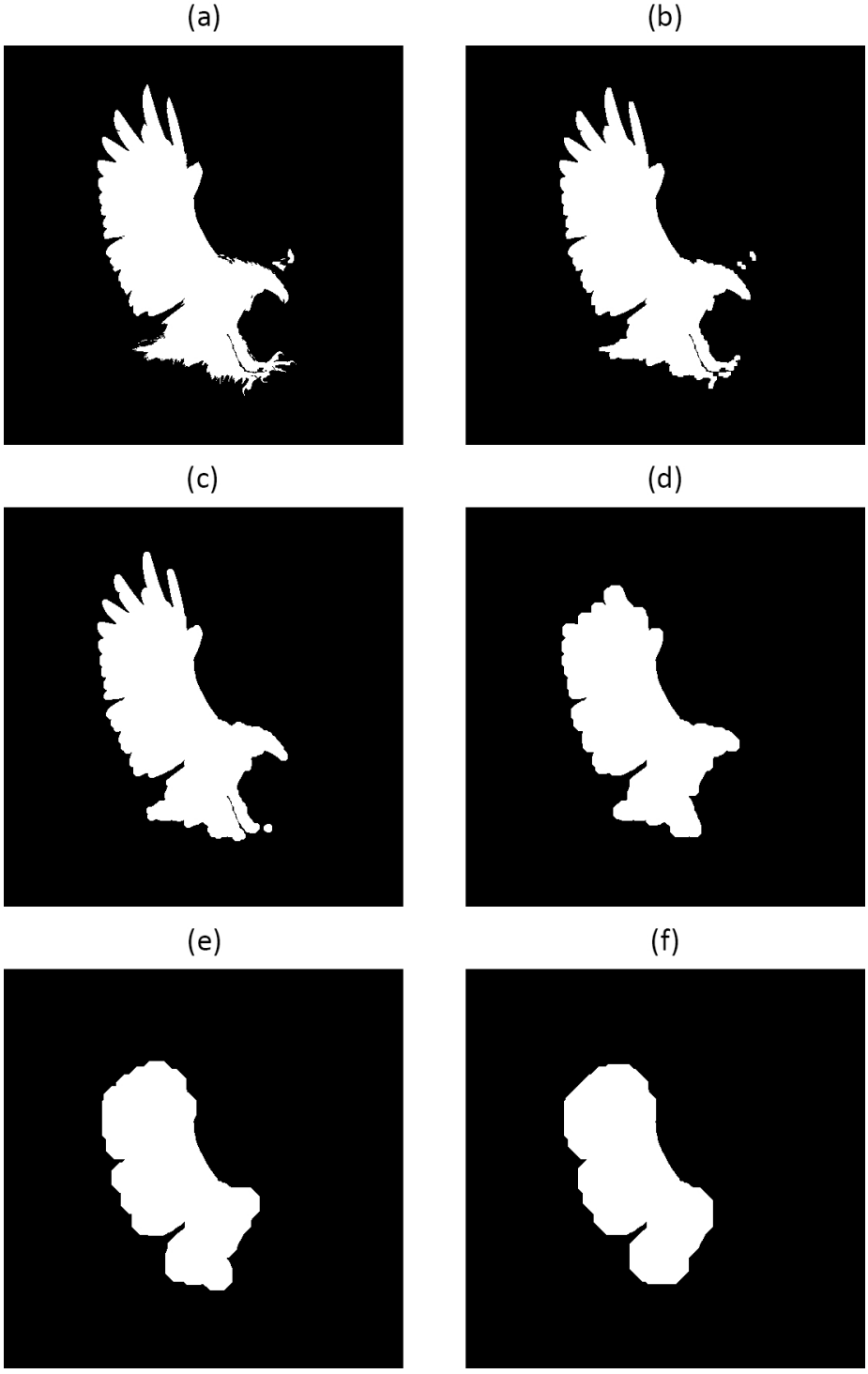

Figura 7: Exemplo de aplicação da operação abertura a uma imagem binária de 512×512 pixels (a) com elemento estruturante circular de raio 3 (b), 5 (c), 10 (d), 20 (e) e 30 (f) pixels.

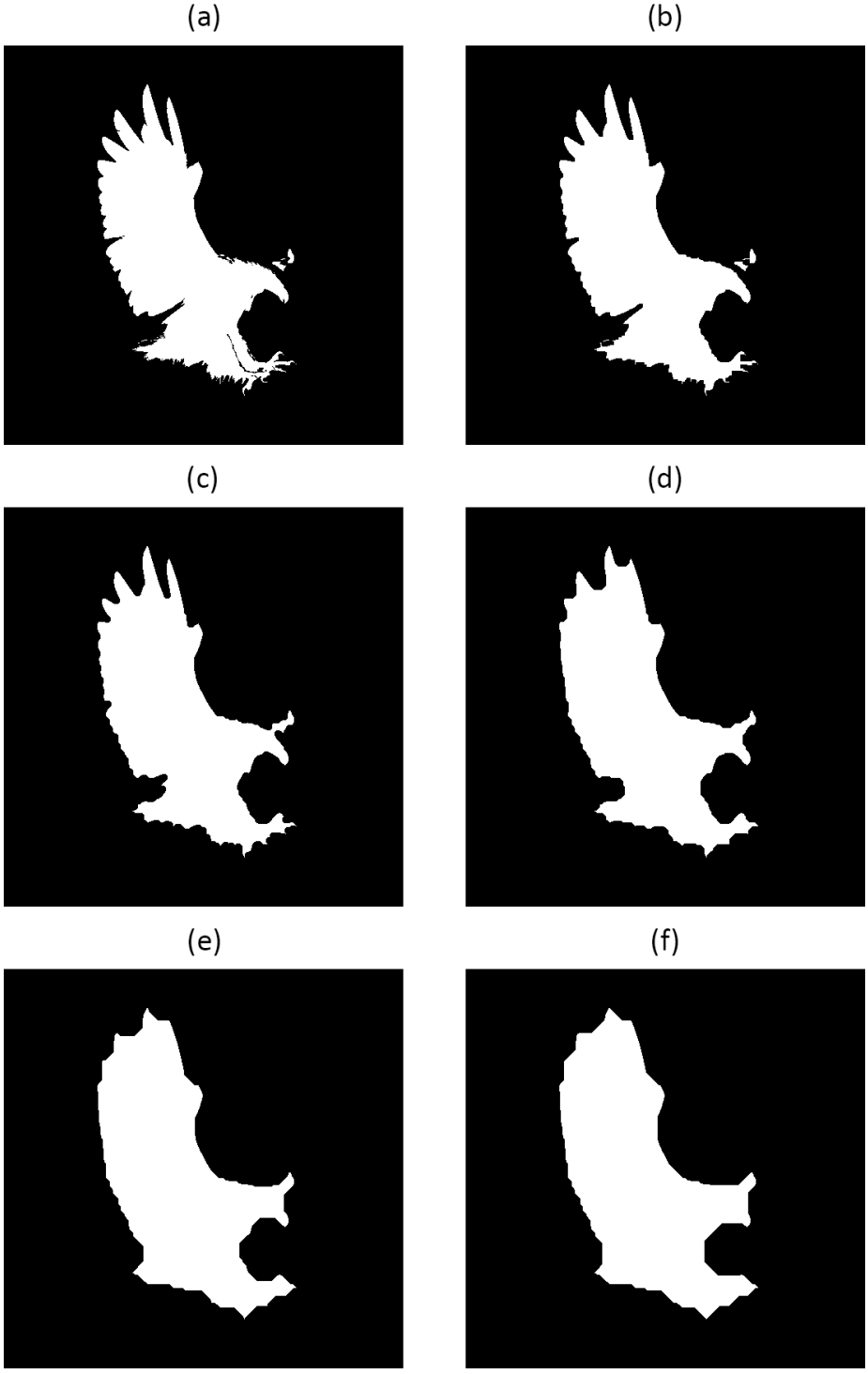

Figura 8: Exemplo de aplicação da operação fecho a uma imagem binária de 512×512 pixels (a) com elemento estruturante circular de raio 3 (b), 5 (c), 10 (d), 20 (e) e 30 (f) pixels.

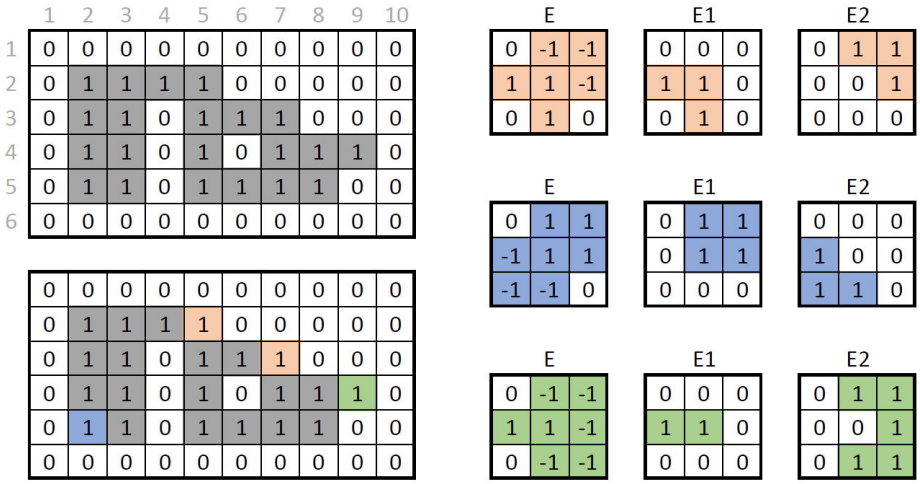

Figura 9: Aplicação do operador Hit-Miss a uma imagem binária (em cima à esquerda) com 3 elementos estruturantes diferentes (direita); e respetivos resultados - pixels coloridos sobre a imagem original (em baixo à esquerda).

4 Outros operadores morfológicos

Há vários outros operadores morfológicos para imagens binárias, a maior parte baseados na aplicação sucessiva dos operadores elementares (dilatação e erosão) e operações lógicas (interseção, reunião, complementar, etc.).

4.1 Operador Hit-Miss

O operador Hit-Miss ($\otimes$) permite detetar a presença de formas pré-definidas numa imagem binária, sendo definido através da equação (7), onde E um elemento estruturante composto, com componentes E_1 (presença) e E_2 (ausência). Os elementos estruturantes para presença/ausência não podem ter sobreposição, ou seja, um elemento não pode pertencer simultâneamente a E_1 e E_2.

$$A \otimes E = (A \ominus E_1) \cap (A^c \ominus E_2) \tag{7}$$

Uma forma compacta de representar o elemento estruturante composto é usando valores 1 (presença), -1 (ausência) e 0 (indiferente). A figura 9(direita) mostra exemplos de elementos estruturantes compostos (E) e as correspondentes componentes presença (E_1) e ausência (E_2). O primeiro caso (laranja) é um detetor de cantos NE, o segundo caso (azul) é um detetor de cantos sólidos SW, e o segundo caso (verde) é um detetor de peninsulas a E. O resultado para a uma imagem binária de teste é apresentado na figura 9 (em baixo à esquerda), onde são identificados com cores os pixels resultantes da aplicação do operador Hit-Miss com cada um dos elementos estruturante (em baixo à esquerda). É interessante observar que no caso do detetor de cantos NE (laranja) há 2 pixels identificados - coordenadas (2,5) e (3,7), mas o detetor de cantos SW (azul) deteta apenas o canto em (5,2). O canto em (5,5) não é detetado uma vez que o pixel (4,6) não pertence ao objeto (tem valor 0).

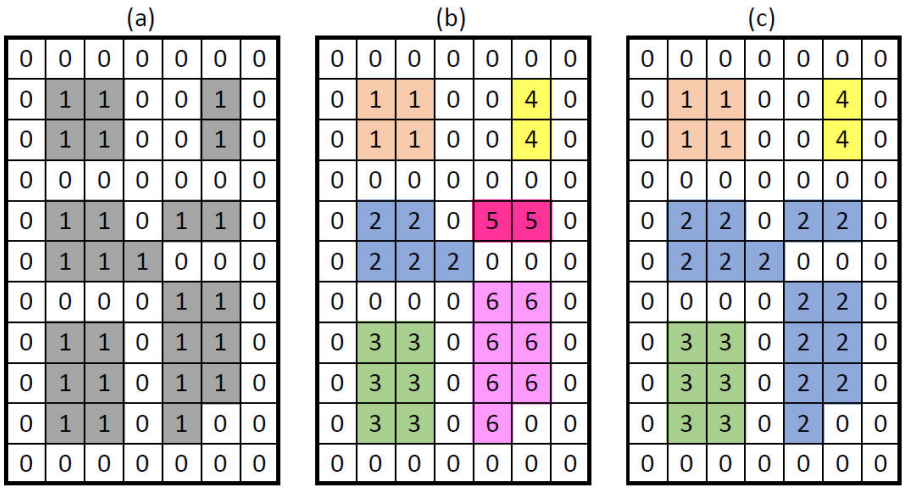

Figura 10: Exemplo de criação de uma imagem de etiquetas com componentes ligadas, obtida a partir de uma imagem binária (a), usando dois critérios de vizinhança: 4- vizinhos (b) e 8- vizinhos (c).

4.2 Componentes ligadas

Uma tarefa particularmente útil é a identificação de componentes ligadas numa imagem binária, que pode ser visto como a identificação de objetos distintos ou separados. Para se implementar esta operação, é necessário considerar um critério de vizinhança, que é usado para verificar se há ou não ligação entre 2 pixels ON da imagem binária. Habitualmente são considerados os 8-vizinhos (quadrado de 3×3) ou os 4-vizinhos (cruz com 5 elementos, como o elemento estruturante da figura 3). A identificação de componentes ligadas tem como *input* uma imagem binária e o tipo de vizinhança considerada, sendo o *output* uma imagem de inteiros com etiquetas associadas a cada objeto identificado. Os pixels de fundo mantém o valor 0, e aos restantes pixels é atríbuido o número do objeto a que pertencem.

A figura 10 mostra um exemplo de identificação de componentes ligadas numa pequena imagem de teste com apenas 11×7 pixels (a), para 2 critérios de vizinhança: 4-vizinhos (b) e 8-vizinhos (c). No primeiro caso foram obtidos 6 objetos, pelo que cada pixel ON na imagem binária toma um valor entre 1 e 6. Observando a imagem de etiquetas (figura 10b), é possível verificar que o pixel no centro, com coordenadas (6,4), pertence ao objeto 2 e toca no canto dos pixels de coordenadas (5,5) e (7,5) que pertencem aos objetos 5 e 6. No entanto, uma vez que a vizinhança considerada foi os 4-vizinhos, este contacto não é considerado vizinhança. Por outro lado, com 8-vizinhos esses pixels irão pertencer ao mesmo objeto (Nº 2), como se pode observar pelo resultado da figura 10(c).

4.3 Número de Euler

O número de Euler de uma imagem binária é o número total de objetos na imagem, menos o número total de buracos nesses objetos, considerando um

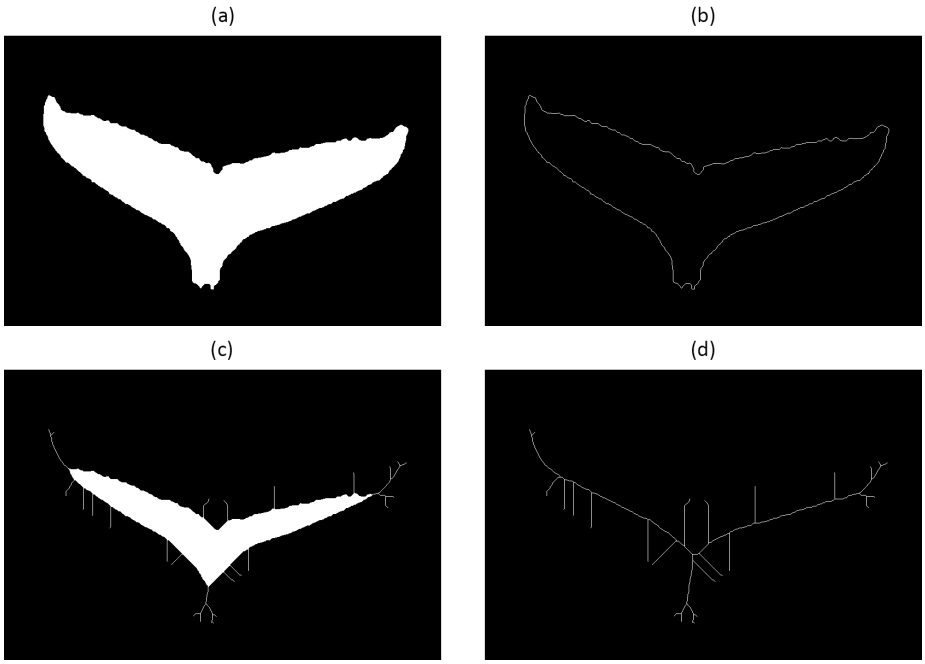

Figura 11: Exemplo de aplicação de operações morfológicas elaboradas: imagem binária original, de 531×800 pixels (a); operação *remove* (b); operação *skeleton* com parâmetro 40 (c) e Inf (d).

critério de vizinhança. No exemplo da figura 10 o número de Euler é 6 para 4-vizinhos e 4 para 8-vizinhos, uma vez que não há qualquer buraco. Por outro lado, no exemplo da figura 9, o número de Euler é 0 (1 objeto - 1 buraco), para qualquer das vizinhanças (4- ou 8- vizinhos).

4.4 Outras operações morfológicas

Há vários exemplos de operações morfológicas mais elaboradas, tendo por base os operadores elementares (dilatação e erosão) e operações lógicas, tais como: *remove* (retira), *skeleton* (esqueleto), *shrink* (mingar), *thin* (mais fino), etc.

A operação *remove* retira os pixels do interior de um objeto. A aplicação consiste em passar a 0 (OFF) os pixels com valor 1 (ON) cujos 4-vizinhos tem também valor 1 (ON). Um pixel mantém o valor 1 apenas no caso de haver pelo menos um dos seus 4-vizinhos com valor 0. O resultado é uma imagem binária com apenas a margem exterior do objeto inicial. A figura 11 mostra um exemplo de aplicação a uma imagem binária de uma baleia (a), sendo o resultado da operação *remove* (b) a zona de contorno do objeto original.

A operação *skeleton* (esqueleto) retira pixels das margens do objeto, mas sem deixar que o objeto se divida. A operação preserva o número de Euler. É necessário definir o valor para um parâmetro que controla o número de iterações, podendo ser Inf . Nesse caso o processo termina quando nõa é possível retirar mais pixels sem quebrar o objeto. A figura 11 mostra exemplos de aplicação operação *skeleton* com parâmetro 40 (c) e Inf (d).

5 Morfologia em imagens de cinzentos

As principais operações morfológicas em imagens binárias são extensíveis para imagens de níveis de cinzento. Na forma mais geral, o elemento estruturante é uma função de 2 dimensões, podendo ter uma maior diversidade de valores do que no caso de imagens binárias. Pode no entanto considerar-se o caso mais simples, com um elemento estruturante plano, que toma apenas um de 2 níveis (0 e 1). Nesse caso, a dilatação de A por E (A$\oplus$E) é definida por (8), e a erosão de A por E (A$\ominus$E) por (9).

$$(A \oplus E)(x,y) = max\{(x+i, y+j) : (x,y) \in A; (i,j) \in E\} \tag{8}$$

$$(A \ominus E)(x,y) = min\{(x+i, y+j) : (x,y) \in A; (i,j) \in E\} \tag{9}$$

Ou seja, colocando o elemento estruturante sobre um pixel em análise, o seu valor é substituído pelo máximo dos pixels cobertos pelo elemento estruturante, no caso da dilatação (8), e pelo mínimo no caso da erosão (9). A figura 12 mostra um exemplo de aplicação destes operadores a uma imagem de teste com um girassol (a), de 8-bits com 468×593 pixels, usando um elemento estruturante plano circular de raio 5 pixels. É possível observar que o resultado da dilatação (figura 12b) é uma imagem mais clara, em particular nas zonas de transição entre níveis claros/escuros, ficando praticamente inalterada em zonas homogéneas (por exemplo o céu). No caso da erosão (figura 12d), a imagem obtida é mais escura, particularmente nas zonas de diversidade de intensidades de cinzento, sendo igualmente preservadas as zonas homogéneas.

A definição das operações abertura e fecho é exatamente a mesma para imagens binárias e de níveis de cinzento. No caso da abertura (A$\circ$E) aplica-se primeiro uma erosão e depois uma dilatação (5), e no caso do fecho (A$\bullet$E) aplica-se uma dilatação seguida de uma erosão (6). A figura 12 mostra o resultado da aplicação destes operadores à imagem do girassol (a) com elemento estruturante plano circular de raio 5 pixels. A abertura (c) e fecho (e) são versão simplificadas da original, com valores menores no caso da abertura e maiores no caso do fecho. As variações de intensidades próximas, neste caso com raio de 5 pixels, são eliminadas, o que torna a imagem mais suave.

As operações dilatação e a erosão são a base do gradiente morfológico. Na sua forma geral, o gradiente morfológico (G) é a diferença entre a dilatação e a erosão (10), podendo igualmente ser definido o gradiente interno (11) e o gradiente externo (12). A figura 12(f) mostra o gradiente da imagem do girassol, onde se pode observar que são realmente detetados os contornos da imagem. Estes contornos são no entanto apresentados com alguma grossura, quando comparados com os obtidos através de filtros espaciais de deteção de contornos, uma vez que o elemento estruturante usado foi neste caso um circulo com 5 pixels de raio.

$$G(A,E) = A \oplus E - A \ominus E \tag{10}$$

$$G_i(A,E) = A - A \ominus E \tag{11}$$

$$G_e(A,E) = A \oplus E - A \tag{12}$$

Figura 12: Exemplo de operações morfológicas numa imagem de tons de cinzento. Imagem original, de 8-bits, com 468×593 pixels (a); dilatação (b), erosão (d), abertura (c) e fecho (e), com elemento estruturante circular de raio 5 pixels; gradiente morfológico (f) obtido a partir de (b) e (d).

Há outras operações morfológicas elaboradas, que tem por base as operações elementares. Por exemplo a transformação *tophat* consiste na subtração da abertura à imagem original, o que pode ser útil para normalização de um fundo com iluminação variável. Um outro exemplo são aplicações de granulometria, onde se pretende determinar o número e tamanho de objetos presentes numa imagem. A aplicação sucessiva de operações fecho, com elementos estruturantes circulares de tamanho crescente, permite identificar o número de objetos ciculares por tamanho, tendo por base a variação da área (ou volume) da imagem resultante em cada caso. Há igualmente técnicas de reconstrução morfológica que permitem simplificar uma imagem, na tentativa de extrair a informação essencial removendo o ruído. A reconstrução morfológica é um processo iterativo que usa uma imagem de marcadores, para além da imagem onde se pretende actuar e o elemento estruturante.

6 Morfologia em imagens a cores

As operações morfológicas podem igualmente ser extendidas a imagens a cores, embora não seja uma prática muito frequente. Uma possibilidade para imagens RGB é aplicação de uma dada operação a cada componente R, G, B, considerando cada uma como uma imagens de níveis de cinzento. Uma outra possibilidade é aplicar a operação morfológica à componente V da imagem a cores no modelo de cores HSV, mantendo inalteradas as componentes H e S.

A figura 13 mostra um exemplo de aplicação de operadores morfológicos a uma imagem a cores (a), no formato RGB de 24-bits. Foi usado um elemento estruturante circular de raio 5 pixels para produzir imagens resultantes da dilatação (b), erosão (d), abertura (c) e fecho (e). As imagens de abertura (c) e fecho (e) são versões simplificadas da original, à semelhança do que ocorre para imagens de cinzentos (figura 12). Nas zonas heterogéneas, a imagem abertura é mais escura do que a original e a imagem fecho mais clara, sendo de uma forma geral preservadas as cores da imagem original. No entanto, para as operações morfológicas dilatação (a) e erosão (d) isso não acontece. No caso da erosão, as alterações de cor são facilmente perceptíveis, por exemplo na aura verde em torno das pétalas do girassol (d) que não existia na imagem original (a), e que resulta da mistura das cores amarela (pétalas) e azul (céu). Uma alternativa mais eficaz no que se refere à preservação das cores originais, é a aplicação dos operadores morfológicos no modelo HSV (apenas à componente V). A figura 13(f) mostra o resultado da erosão aplicada à imagem do girassol via modelo HSV. Nesta imagem erodida, as cores da imagem original são melhor preservadas, sendo no entanto o processo de erosão igualmente eficaz. Um local onde se pode confirmar este facto é na pétala em baixo do lado esquerdo, onde há uma ligeira mancha escura na imagem original. A erosão com elemento estruturante circular de raio 5 pixels faz com que essa pequena zona escura se torne bastante maior, sendo o resultado nessa zona da imagem praticamente igual na erosão via modelos RGB (d) e HSV (f).

Figura 13: Exemplo de operações morfológicas numa imagem a cores. Imagem original RGB, de 24-bits, com 468×593 pixels (a); dilatação (b), erosão (d), abertura (c) e fecho (e), com elemento estruturante circular de raio 5 pixels; erosão aplicada no modelo HSV (f), apenas à componente V.

7 Operações Morfológicas no MATLAB

7.1 Elemento estruturante

A definição de um elemento estruturante é um ponto essencial para a maioria das operações morfológicas. A forma habitual de o fazer no MATLAB é através da função `strel`, podendo no entanto ser feito através de funções e operadores básicos, como se ilustra de seguida, para um elemento estruturante quadrado de lado 3 pixels.

```
>> EE = strel('square',3);
>> whos EE
Name      Size      Bytes      Class      Attributes
EE        1x1       102        strel
>> EE
EE =
strel is a square shaped structuring element with properties:
    Neighborhood:  [3×3 logical]
   Dimensionality:  2
>> EE.Neighborhood
ans =
     1     1     1
     1     1     1
     1     1     1
>> EEQ3 = strel(logical(ones(3));
```

tipo e tamanho

mostrar o conteúdo

componente de EE

forma alternativa

Podem ser usadas outras formas *standard* de elementos estruturantes com a função `strel`, para além de `'square'`, tais como: `'diamond'`, `'rectangle'`, `'line'`, `'octagon'` e `'disk'`. Há ligeiras diferenças em termos de parâmetros de *input* para os vários tipos. Apresentam-se alguns exemplos em seguida.

```
>> EE1 = strel('diamond',2);
>> EE1.Neighborhood
     0     0     1     0     0
     0     1     1     1     0
     1     1     1     1     1
     0     1     1     1     0
     0     0     1     0     0
>> EE2 = strel('rectangle',[3 5]);
>> EE3 = strel('line',8,45);
>> EE3.Neighborhood
     0     0     0     0     1
     0     0     0     1     0
     0     0     1     0     0
     0     1     0     0     0
     1     0     0     0     0
>> EE4 = strel('octagon',6);
```

corresponde a janela de 5×5 pixels

comprimento e ângulo

janela de 13×13 pixels

Neste último caso (`'octagon'`) o parâmetro numérico corresponde aproximadamente ao raio do circulo envolvente do octógono (em pixels), que tem de ser múltiplo de 3. No caso da linha (`'line'`), os 2 parâmetros de *input* são o

comprimento (aproximado) em pixels e o ângulo em graus. O formato `'disk'`, que é um dos mais frequentemente utilizados, na sua forma mais simples usa apenas 1 valor numérico, que corresponde ao raio do disco, em pixels. No entanto, em certos casos pode ser útil usar um segundo parâmetro numérico que controla a precisão da identificação do disco. Colocando o valor 0 neste segundo parâmetro, a forma do disco é mais perfeita, mas o tempo de execução é maior.

```
>> EEd4a = strel('disk',4);
>> EEd4b = strel('disk',4,0);
```

raio 4 pixels pixels

maior precisão

7.2 Erosão, Dilatação, Abertura e Fecho

As operações morfológicas elementares, assim como as operações Abertura e Fecho, são implementadas de forma muito simples no MATLAB, através das funções `imerode` (Erosão), `imdilate` (Dilatação), `imopen` (Abertura) e `imclose` (Fecho). Para todas estas funções são usados 2 parâmetros de *input*: (1) a imagem onde se pretende operar e (2) o elemento estruturante. Alguns exemplos:

```
>> IMe = imerode(IM,EE1);
>> IMd = imdilate(IM,EE1);
>> IMa = imopen(IM,EE1);
>> IMf = imclose(IM,strel('square',5));
```

imagem original

elemento estruturante

Estas funções podem ser aplicadas a imagens binárias, de cinzento e de cor RGB. Neste último caso é aplicada a operação morfológica escolhida a cada uma das componenetes R,G,B. Para aplicação via modelo de cor HSV é necessário converter a imagem de RGB para HSV, aplicar a operação à componente V e converter de volta para o modelo de cor inicial (RGB).

7.3 Componentes Ligadas

A partir de uma imagem binária (`IB`) pode ser criada uma imagem de etiquetas usando a função `bwlabel`. A imagem de etiquetas tem valores 0 para os pixels com valor 0 em `IB`, e valores inteiros a partir de 1 correspondendo ao objeto distinto a que pertencem, considerando uma dada vizinhança. A função pode ter 1 ou 2 parâmetros de *input* - a imagem onde se vai actuar, e o tipo de vizinhança (por *default* 8, ou seja 8-vizinhos). Tem também 1 ou 2 parâmetros de *output* - a imagem de etiquetas e o número de objetos, como se ilustra nos exemplos em seguida.

```
>> ET = bwlabel(IB);
>> ET = bwlabel(IB,4);
>> [ET,NoOb] = bwlabel(IB);
```

imagem original

4 vizinhos

Nº de objetos

Há várias tarefas relevantes que podem ser feitas com esta imagem de etiquetas, que apesar de não serem específicas de operações morfológicas, são apresentadas em seguida. Uma delas é a vizualização da imagem de etiquetas, que em geral deve ser feita com uma tabela de cores, para facilitar a discriminação visual dos vários objetos. Por exemplo, com a tabela de cor prism:

```
>> imshow(ET,prism)
```

Uma outra tarefa muito útil é a separação de cada objeto, numa imagem binária distinta. Por exemplo, a extração de imagens binárias para os objetos com etiquetas número 4 e 7 pode ser feita através de:

```
>> ET = bwlabel(IB);
>> OBJ4 = (ET==4);
>> OBJ7 = (ET==7);
```

imagem de etiquetas

pixels com etiqueta 4

Ou de uma forma mais genérica, pode ser criada uma matriz tridimensional (ou imagem multi-banda), em que cada plano (ou banda) corresponde a uma imagem binária com cada um dos objetos identificados. Em seguida o código a utilizar para esse procedimento.

```
>> [ET,NoOb] = bwlabel(IB);
>> for f=1:NoOb
>>    OBJ(:,:,f)=(ET==f);
>> end
```

Nº de objetos

para cada objeto

matriz tridimensional

É fácil de imaginar que agora se poderia aplicar uma sequência de tarefas de processamento a cada uma das imagens binárias extraidas de forma automática da imagem original, e colocadas na matriz `OBJ`.

7.4 Outras operações morfológicas

O operador Hit-Miss é implementado através da função `bwhitmiss`, que recebe como input a imagem binária onde se pretende actuar (`IB`), e uma matriz onde são identificadas as imposições de pertença (1), ausência (-1), e indiferença (0). Em seguida um exemplo com o elemento estruturante (`EE`) usado no primeiro caso da figura 9 (laranja).

```
>> EE = [0 -1 -1 ; 1 1 -1 ; 0 1 0];
>> Ihm = bwhitmiss(IB,EE);
```

O número de Euler de uma imagem binária (`IB`) é obtido a partir da função `bweuler`, de acordo com a sintaxe indicada nos exemplos abaixo. O segundo parâmetro de *input* define o critério de vizinhança, sendo o valor por omissão (*default*) 8, o que corresponde a uma janela de 3×3.

```
>> NE = bweuler(IB);
>> NE = bweuler(IB,4);
```

default, 8 vizinhos

4 vizinhos

As operações morfológicas mais elaboradas, apresentadas em 4.4 e ilustradas no exemplo da figura 11, podem ser implementadas no MATLAB através da função `bwmorph`. Alguns exemplos em seguida.

```
>> IB1 = bwmorph(IB,'remove');
>> IB2 = bwmorph(IB,'skel',40);
>> IB3 = bwmorph(IB,'skel',Inf);
>> IB4 = bwmorph(IB,'shrink');
```

retira

esqueleto com 40 iterações

Infinito, ou seja o máximo possível

tornar mais fino

8 Exercicios

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

A1 - Imagem binária com objetos de interesse

Pretende-se ter uma função que crie uma imagem binária correspondente aos objetos de interesse numa dada imagem. A função deverá ter como *inputs* o nome do ficheiro imagem (formato standard) e um 2º parâmetro (`graf`) para controlar se deve (`graf`=1) ou não (outros valores) ser apresentada uma janela gráfica com resultados. Se a imagem original for RGB, deverá ser convertida para uma versão a níveis de cinzento. Depois deverá ser aplicada binarização por *Thresholding* Global e implementado um critério de decisão para escolher `IB` entre essa versão binária (`IBo`) ou a complementar (`IBc=1-IBo`), por forma a que os pixels de valor 1 (ON) estejam predominantemente no centro da imagem. Se `graf`=1, deverá ser apresentada uma janela (`subplot` de 2×2) com: (1) imagem original, (2) versão cinzento (pode ser igual), (3) `IBo` e (4) `IB`.

Exemplo de utilização: `IB=FuncaoA1('RobotC.tif',1);`

A2 - Imagem com etiquetas

Uma função para criar uma imagem de etiquetas para objeto distintos, cosiderando uma vizinhança de 8. A função deverá ter como *inputs* o nome do ficheiro imagem (formato standard) e um 2º parâmetro (`graf`) para controlar se deve (`graf`=1) ou não (outros valores) ser apresentada uma janela gráfica com resultados. Inicialmente deverá ser chamada a função A1 para se obter uma imagem binária `IB`, onde se identificarão posteriormente os vários objetos separados. Se `graf`=1, deverá ser apresentada uma janela (`subplot` de 1×3) com: (1) imagem original, (2) `IB` e (3) imagem de etiquetas (`IEt`), com indicação do número de objetos identificado.

Exemplo de utilização: `IEt=FuncaoA2('RobotC.tif',1);`

A3 - Imagem com maior objeto na zona central

Uma função para criar uma imagem binária com o maior objeto presente numa imagem. A função deverá ter como *inputs* o nome do ficheiro imagem (formato standard) e o parâmetro `graf`. Inicialmente deverá ser chamada a função A1 para se obter uma 1ª imagem binária (`IB1`). Deverá igualmente chamar-se a função A2 para se obter uma imagem de etiquetas (`IEt`). Depois, deverá ser criada uma imagem binária (`IB2`) com o maior objeto presente em `IEt`, e uma outra (`IB3`) com o maior objeto de `IEt` que não toca na margem da imagem (faixa de 2 pixels a toda a volta). Se `graf`=1, deverá ser apresentada uma janela (`subplot` de 2×2) com a imagem original, `IB1`, `IB2` e `IB3`.

Exemplo de utilização: `[IB2,IB3]=FuncaoA3('RobotC.tif',1);`

B1 - Visualização de operações morfológicas

Crie uma função MATLAB que receba como *input* o nome de um ficheiro imagem (formato standard, imagem de cinzento ou RGB) e apresente uma figura (`subplot` de 3×4) com a dilatação, erosão, abertura e fecho, usando elementos estruturantes circulares de raio 3, 5 e 10 pixels.

Exemplo de utilização: `FuncaoB1('Aguia.jpg');`

B2 - Visualização de operações morfológicas (binárias)

Prepare uma versão modificada de B1 que, recebendo como *input* o nome de um ficheiro imagem (formato standard, imagem de cinzento ou RGB), chama a função A1 para obter uma imagem binária (`IB`), e apresenta 2 janelas gráficas. A primeira, com a imagem original e `IB`. A segunda com a dilatação, erosão, abertura e fecho de `IB`, com elementos estruturantes circulares de raio 3, 5 e 10 pixels (`subplot` de 3×4) .

Exemplo de utilização: `FuncaoB2('Aguia.jpg');`

C - Videos com operações morfológicas

Implemente uma função MATLAB para criar videos com dilatação, erosão, abertura e fecho de uma imagem binária, usando elementos estruturantes circulares de raio crescente (de 1 a 50). A função deverá receber como *input* o nome de um ficheiro imagem (formato standard, imagem de cinzento ou RGB), chamando a função A1 para obter uma imagem binária (`IB`), Cada vídeo deverá ter 51 imagens (*frames*): a imagem inicial (`IB`) e as obtidas pelas operações morfológicas.

Exemplo de utilização: `FuncaoC('Aguia.jpg');`
Ficheiros criados: `Dilatacao.avi` ; `Erosao.avi` ; `Abertura.avi` ; `Fecho.avi`

8.1 APPENDIX - Exercices (in English)

This Appendix proposes some exercises for MATLAB that can be used to consolidate the Image Processing concepts presented.

A1 - Binary image with objects of interest

The goal is to have a function that creates a binary image corresponding to the objects of interest in a given image. The function should have as inputs the image filename (standard format) and a 2nd parameter (`graf`) to control whether (`graf`=1) or not (other values) a graphic window with results is displayed. If the original image is RGB, it must be converted to a grayscale version. Then, a binary image should be produced by Global Thresholding, and a decision criterion should be implemented to choose `IB` between this binary version (`IBo`) or the complementary one (`IBc=1-IBo`), so that pixels of value 1 (ON) are predominantly in the center of the image. If `graf`=1, a window (2×2 `subplot`) should be displayed with: (1) original image, (2) grayscale version (can be equal), (3) `IBo` and (4) `IB`.

Example of use: `IB=FunctionA1('RobotC.tif',1);`

A2 - Image with labels

A function to create an image of labels for different objects, considering a neighborhood of 8. The function should have as inputs the image filename (standard format) and a 2nd parameter (`graf`) to control whether (`graf`=1) or not (other values) a graphic window with results is displayed. Initially, function A1 should be called to obtain a binary image `IB`, where the separate objects will be identified later. If `graf`=1, a window (1×3 `subplot`) should be displayed with: (1) original image, (2) `IB` and (3) image of labels (`IEt`), with an indication of the number of objects identified.

Example of use: `IEt=FunctionA2('RobotC.tif',1);`

A3 - Image with largest object in the central zone

A function to create a binary image with largest object present in the imagem. The function should have as inputs the image filename (standard format) and the parameter (`graf`). Initially, function A1 should be called to obtain a 1st binary image (`IB1`). Function A2 should also be called to obtain an image of labels (`IEt`). Then, a binary image (`IB1`) must be created with the largest object present in (`IEt`), and another one (`IB3`) with the largest object that does not touch the edge of the image (2 pixel range all around). If `graf`=1, a window (2×2 `subplot`) should be displayed with the original image, `IB1`, `IB2` and `IB3`.

Example of use: `[IB2,IB3]=FunctionA3('RobotC.tif',1);`

B1 - Visualization of morphological operations

Create a MATLAB function that receives as input the name of an image file (standard format, grayscale or RGB image) and presents a figure (3×4 `subplot`) with dilation, erosion, opening and closing, using circular structuring elements of radius 3, 5 and 10 pixels.

Example of use: `FunctionB1('Aguia.jpg');`

B2 - Visualization of morphological operations (binary)

Prepare a modified version of B1 that receives as input the name of an image file (standard format, grayscale or RGB image), calls function A1 to obtain a binary image (`IB`), and presents 2 graphic windows. The first, with the original image and (`IB`). The second with the dilation, erosion, opening and closing of `IB`, with circular structuring elements of radius 3, 5 and 10 pixels (3×4 `subplot`).

Example of use: `FunctionB2('Aguia.jpg');`

C - Videos of morphological operations

Implement a MATLAB function to create videos with dilation, erosion, opening and closing of a binary image, using circular structuring elements of increasing radius (from 1 to 50). The function should receive as input the name of an image file (standard format, grayscale or RGB image), calling function A1 to obtain a binary image (`IB`). Each video must have 51 images (frames): the initial image (`IB`) and those obtained by the morphological operations.

Example of use: `FunctionC('Aguia.jpg');`
Files created: `Dilation.avi` ; `Erosion.avi` ; `Opening.avi` ; `Closing.avi`