Introdução ao Processamento de Imagem Digital (MO443/MC920)

Prof. Alexandre Xavier Falcão

Material de 2005

1 Objetivos

Este curso aborda os conceitos e técnicas básicas em processamento de imagem digital com o objetivo de preparar o aluno para cursos mais avançados, tais como análise de imagens, visão computacional, compressão de imagens, processamento de imagens de sensoriamento remoto, processamento de vídeo digital, e computação de imagens médicas.

2 Ementa

- Fundamentos de Imagem Digital
- Transformações Geométricas
- Interpolação e registro de imagens
- Transformações Radiométricas e realçe
- Convolução e Correlação
- Filtragem no Domínio Espacial
- Transformada de Fourier
- Filtragem no Domínio da Freqüência
- Morfologia Matemática
- Transformada Imagem-Floresta
- Segmentação de Imagens

3 Bibliografia

- E.R. Dougherty & R.A. Lotufo. Hands-on Morphological Image Processing, SPIE Press, 2003.
- A.S. Glassner. Principles of Digital Image Synthesis, Vols. 1 e 2, Morgan Kaufmann, 1995.
- R.C. Gonzalez & R. E. Woods. Processamento de Imagens Digitais, Edgard Blücher, 2000.
- R. C. Gonzalez & R. E. Woods. Digital Image Processing, Addison-Wesley, 1993.
- P. Soille. Morphological Image Analysis: Principles and Applications, Springer, 1999.
- A. Rosenfeld & A. C. Kak. Digital Picture Processing Vols. 1 e 2, Academic Press, 1982.
- J. Serra. Image Analysis and Mathematical Morphology, vol. 1, Academic Press, 1982.
- A.X. Falcão e N.J. Leite, Fundamentos de Processamento de Imagem Digital, Apostila em
  www.ic.unicamp.br/~cpg/material-didatico/MO815/9802.

1 Imagem Digital

Uma imagem digital $\hat{I}$ é um par $(D_I, \vec{I})$, onde D_I é um conjunto de pontos do Z^n (domínio da imagem), denominados spels (*space elements*), e $\vec{I}$ é um mapeamento vetorial que associa a cada spel p em D_I um conjunto $\{I_1(p), I_2(p), \ldots, I_k(p)\}$ de valores escalares, associados com alguma propriedade física. O valor de n refere-se à dimensão da imagem e o valor de k ao número de bandas.

2 Imagem em tons de cinza

Uma imagem $\hat{I} = (D_I, I)$ em tons de cinza (e.g. foto, imagem de ultrasom, fatia tomográfica) e bidimensional ($D_I \subset Z^2$) possui apenas uma banda I ($k = 1$), onde os spels são chamados pixels (*picture elements*).

A imagem é portanto uma matriz de tamanho $N \times M$ pixels (N linhas e M colunas). Sua representação vetorial relaciona o índice i a cada pixel $p = (x, y)$ por:

$$i = x + M * y \quad (1)$$

$$x = i\%M \quad (2)$$

$$y = i/M \quad (3)$$

Os valores $I(p)$ de cada pixel p são obtidos por amostragem e quantização de uma função contínua $I_c(x, y)$ que descreve a propriedade física correspondente em uma dada região do espaço. No caso de uma foto temos o brilho, e no caso de uma tomografia de Raios-X, temos a densidade do tecido.

Valores altos são apresentados na tela como pixels claros e valores baixos como pixels escuros.

2.1 Amostragem e Resolução Espacial

Cada pixel é amostrado a intervados (Δ_x, Δ_y) (e.g. $\Delta_x = \Delta_y = 1mm$). Quanto menor for o intervalo de amostragem para uma mesma região do espaço, maior será a resolução espacial da imagem. Observe que neste caso, o tamanho $N \times M$ da imagem também é maior, mas se uma imagem tem mais pixels que outra, não implica que tenha maior resolução.

2.2 Quantização e Resolução Radiométrica

Os valores de $I_c(x, y)$ amostrados são quantizados em 2^b níveis de cinza, onde b é chamado profundidade da imagem em bits (e.g. $b = 8$, profundidade de 8 bits). Quanto menor o intervalo de quantização, maior é a resolução radiométrica da imagem.

2.3 Histograma

O histograma de uma imagem cinza $\hat{I}$ é uma função $h(l)$ que produz o número de ocorrências de cada nível de cinza $0 \leq l \leq 2^b - 1$ na imagem. Ele representa a distribuição de probabilidade

dos valores dos pixels. O histograma é normalizado em $[0, 1]$ quando dividimos $h(l)$ pelo número $N \times M$ de pixels da imagem.

2.4 Operações matemáticas

Operações lógicas e aritméticas entre imagens são equivalentes às mesmas operações realizadas pixel a pixel, entre os valores dos pixels.

3 Imagem Multidimensional

Uma imagem $\hat{I} = (D_I, I)$ em tons de cinza e multidimensional define domínio de amostragem $D_I \subset Z^n$, para $n > 2$. Por exemplo, uma seqüência espacial de fatias tomográficas é uma imagem tridimensional ($n = 3$), e uma seqüência espacial e temporal de fatias tomográficas é uma imagem tetradimensional ($n = 4$). No primeiro caso, os spels são chamados de voxels (*volume element*).

O intervalo de amostragem ao longo do eixo temporal define a resolução temporal da imagem. Quanto menor o intervalo, maior é a resolução.

4 Imagem Multibanda

Uma imagem $\hat{I} = (D_I, \vec{I})$ é multibanda para $|\vec{I}| = k > 1$ (e.g. imagens de sensoriamento remoto, imagens coloridas).

No caso do sensor *thematic mapper* (TM) do LandSat5, por exemplo, as bandas $k = 1, 2, 3, \ldots, 7$ correspondem respectivamente a imagens cinza obtidas nos comprimentos de onda do azul, vermelho, verde, infravermelho próximo, infravermelho médio, infravermelho termal, e infravermelho médio. O intervalo de amostragem define a resolução espectral.

No caso de uma foto colorida temos $k = 1, 2, 3$ correspondendo aos componentes vermelho, verde e azul. Observe que o vídeo colorido é uma imagem multidimensional e multibanda.

5 Exercícios

1. Desenhe o histograma de uma imagem cujos valores dos pixels são $10, 1, 0, 3, 9, 2, 2, 3, 4, 5, 6, 5, 6, 7, 8, 9, 3, 4, 7, 6, 5$.

Deste ponto em diante, o curso será restrito ao caso bidimensional.

1 Cor

A cor é o resultado da percepção da luz (comprimento de onda de 0.4–0.7μm) que incide na retina em células foto-receptoras, denominadas cones. Existem três tipos de cones, sensíveis a diferentes cores de azul, vermelho e verde. A maioria das cores visíveis pelo olho humano pode ser representada pela combinação de luzes monocromáticas nos comprimentos de onda do azul, vermelho e verde. O olho humano percebe cerca de 30 níveis de cinza e 7 milhões de cores. Ele é mais sensível ao verde, depois ao vermelho, e menos ao azul, porém percebe mais variações de azul, depois de vermelho e menos de verde.

Uma cor pode ser decomposta em três componentes independentes: intensidade (I), matiz (M), e saturação (S). A intensidade é responsável pela sensação de brilho, a matiz pela sensação de cor (comprimento de onda), e a saturação pelo grau de pureza da cor.

Imagens coloridas são armazenadas em três componentes primários formando um espaço de cor.

2 Espaço RGB (Thomas Young, 1773-1829)

O espaço RGB é formado pela nossa sensação da soma ponderada do *red* (R), *green* (G) e *blue* (B), os quais geram a maioria das cores visíveis. Seu espaço complementar CMY é formado pelo cyan (C=255-R), magenta (M=255-G) e yellow (Y=255-B).

O espaço RGB é usado para mostrar imagens coloridas na tela do computador, enquanto o espaço CMY é usado em impressoras.

3 Espaço HSV

O espaço HSV representa a matiz, a saturação e o brilho. Como os componentes primários são descorrelacionados, melhoramentos na imagem através de transformações radiométricas aplicadas à saturação (S) e/ou ao brilho (V) não afetarão a matiz (H).

Conversão de RGB 24 bits para HSV real:

Entrada: Image $\hat{I} = (D_I, \vec{I})$, onde $\vec{I} = \{R, G, B\}$.
Saída: Imagem $\hat{I}' = (D_{I'}, \vec{I'})$, $D_{I'} = D_I$, $\vec{I'} = \{H, S, V\}$.
Auxiliares: Matrizes reais R', G', e B', e variáveis reais min, max e $delta$.

1. Para todo pixel $p \in D_I$ faça
2. $R'(p) \leftarrow R(p)/255.0$, $G'(p) \leftarrow G(p)/255.0$, e $B'(p) \leftarrow B(p)/255.0$.
3. $max \leftarrow \max\{R'(p), G'(p), B'(p)\}$ e $min \leftarrow \min\{R'(p), G'(p), B'(p)\}$.
4. $V(p) \leftarrow max$.
5. $delta \leftarrow max - min$.

6. Se $delta = 0.0$ então

7. $S(p) \leftarrow 0$ e $H(p) \leftarrow nil$,

8. no caso contrário,

9. $S(p) \leftarrow delta/max$.

10. Se $B'(p) = max$ então

11. $H(p) \leftarrow 4.0 + (R'(p) - G'(p))/delta$,

12. no caso contrário,

13. Se $G'(p) = max$ então

14. $H(p) \leftarrow 2.0 + (B'(p) - R'(p))/delta$,

15. no caso contrário,

16. $H(p) \leftarrow (G'(p) - B'(p))/delta$.

17. $H(p) \leftarrow H(p) * 60.0$.

18. Se $H(p) < 0.0$ então $H(p) \leftarrow H(p) + 360.0$.

Note que $H(p) = nil$ ou $0.0 \leq H(p) \leq 360.0$, $0.0 \leq S(p) \leq 1.0$, e $0.0 \leq V(p) \leq 1.0$, são valores reais.

Conversão de HSV real para RGB 24bits:

Entrada: Imagem $\hat{I'} = (D_{I'}, \vec{I'})$, $D_{I'} = D_I$, $\vec{I'} = \{H, S, V\}$.
Saída: Image $\hat{I} = (D_I, \vec{I})$, onde $\vec{I} = \{R, G, B\}$.
Auxiliares: Matrizes reais R', G', e B', variáveis reais a, b, c, e d, e variável inteira i.

1. Para todo pixel $p \in D_{I'}$ faça

2. Se $H(p) = nil$ então

3. $R'(p) \leftarrow V(p)$, $G'(p) \leftarrow V(p)$, e $B'(p) \leftarrow V(p)$,

4. no caso contrário,

5. Se $H(p) = 360.0$ então $H(p) \leftarrow 0.0$.

6. $H(p) \leftarrow H(p)/60.0$.

7. $i \leftarrow (int)H(p)$.

8. $a \leftarrow H(p) - i$.

9. $b \leftarrow V(p) * (1.0 - S(p))$.

10. $c \leftarrow V(p) * (1.0 - (S(p) * a))$.

11. $d \leftarrow V(p) * (1.0 - (S(p) * (1.0 - a)))$.

12. Verifique i:

13. Caso $i = 0$, então $R'(p) \leftarrow V(p)$, $G'(p) \leftarrow d$, $B'(p) \leftarrow b$.

14. Caso $i = 1$, então $R'(p) \leftarrow c$, $G'(p) \leftarrow V(p)$, $B'(p) \leftarrow b$.

15. Caso $i = 2$, então $R'(p) \leftarrow b$, $G'(p) \leftarrow V(p)$, $B'(p) \leftarrow d$.

16. Caso $i = 3$, então $R'(p) \leftarrow b$, $G'(p) \leftarrow c$, $B'(p) \leftarrow V(p)$.

17. Caso $i = 4$, então $R'(p) \leftarrow d$, $G'(p) \leftarrow b$, $B'(p) \leftarrow V(p)$.

18. Caso $i = 5$, então $R'(p) \leftarrow V(p)$, $G'(p) \leftarrow b$, $B'(p) \leftarrow c$.

19. $R(p) \leftarrow (int)(255 * R'(p))$, $G(p) \leftarrow (int)(255 * G'(p))$, e $B(p) \leftarrow (int)(255 * B'(p))$.

4 Espaço IHS (W. Pratt)

Para todo pixel $p \in D_I$, $0 \leq R(p) \leq 1$, $0 \leq G(p) \leq 1$, e $0 \leq B(p) \leq 1$ temos:

$$
\begin{aligned}
I(p) &= \frac{R(p) + G(p) + B(p)}{3} && (4)\\
V_1(p) &= \frac{R(p) + G(p) + 2B(p)}{\sqrt{6}} \\
V_2(p) &= \frac{R(p) + 2G(p)}{\sqrt{6}} \\
H(p) &= \arctan\left(\frac{V_2(p)}{V_1(p)}\right) && (5)\\
S(p) &= \sqrt{V_1(p)^2 + V_2(p)^2} && (6)
\end{aligned}
$$

onde $S(p) \geq 0.0$, $I(p) \leq 1.0$, e $0 \leq H(p) \leq 2\pi$. A inversa é dada por:

$$
\begin{aligned}
V_1(p) &= S(p)\cos(H(p)) \\
V_2(p) &= S(p)\sin(H(p)) \\
R(p) &= 12I(p) - 4.9V_1(p) - 2.45V_2(p) && (7)\\
G(p) &= -6I(p) + 2.45V_1(p) + 2.45V_2(p) && (8)\\
B(p) &= -3I(p) + 2.45V_1(p) && (9)
\end{aligned}
$$

5 Espaço YCbCr (Vídeo Digital)

Para todo pixel $p \in D_I$, $0 \leq R(p) \leq 255$, $0 \leq G(p) \leq 255$, e $0 \leq B(p) \leq 255$ temos:

$$Y(p) = 0.257R(p) + 0.504G(p) + 0.098B(p) + 16 \quad (10)$$
$$Cr(p) = 0.439R(p) - 0.368G(p) - 0.071B(p) + 128 \quad (11)$$
$$Cb(p) = -0.148R(p) - 0.291G(p) + 0.439B(p) + 128 \quad (12)$$

onde $0 \leq Y(p) \leq 255$, $0 \leq Cb(p) \leq 255$, e $0 \leq Cr(p) \leq 255$.

$$R(p) = 1.164(Y(p) - 16) + 1.596(Cr(p) - 128) \quad (13)$$
$$G(p) = 1.164(Y(p) - 16) - 0.813(Cr(p) - 128) - 0.392(Cb(p) - 128) \quad (14)$$
$$B(p) = 1.164(Y(p) - 16) + 2.017(Cb(p) - 128) \quad (15)$$

6 Display de Imagens

Note que em todas as conversões acima entre espaços de cores, o uso de valores reais é fundamental para não haver perdas. No entando, o *display* das imagens RGB requer valores inteiros de 0 a 255 para cada componente. Portanto, devemos garantir que os valores de R, G e B estarão neste intervalo antes do *display*.

Imagens em tons de cinza também podem ser apresentadas com cores, através do uso de tabela de cores. Neste caso, o valor de cinza é o índice da cor correspondente na tabela.

7 Histograma de Cor

O histograma de uma imagem RGB de 24 bits é uma função $h(c)$ que produz o número de ocorrências de cada cor c, considerando todas as 2^{24} (16.777.216) ocorrências de (r, g, b), $0 \leq r, g, b \leq 255$.

Na prática, o histograma de cor costuma ser quantizado em um número bem menor de cores, e.g. 64 cores. Neste caso, o espaço RGB é subdividido em $4 \times 4 \times 4$ regiões cúbicas, e todos os pixels com cor em uma dessas regiões soma 1 no *bin* (índice) correspondente do histograma.

8 Exercício

1. Escreva um algoritmo para calcular o histograma de cor de 216 bins a partir de uma imagem RGB.

2. Implemente as rotinas de conversão de RGB para HSV, e vice-versa, vistas nesta aula.

1 Introdução à Topologia Digital

Topologia digital é o estudo de propriedades de objeto em imagem digital, as quais não são afetadas por transformações geométricas, exceto aquelas que envolvem junção ou separação de partes do objeto.

1.1 Relação Binária

Uma relação binária R aplicada a um conjunto X é um subconjunto do produto cartesiano $X \times X$.

Uma relação binária é dita reflexiva se $(a, a) \in R$, para todo $a \in X$, simétrica se $(a, b), (b, a) \in R$, para todo $a, b \in X$, e transitiva se $(a, b), (b, c) \in R$ implica que $(a, c) \in R$, para todo $a, b, c \in X$. Neste caso R é dita de equivalência.

1.2 Métrica

Uma função d de distância entre pixels é uma métrica se:

$$d(p, q) \geq 0 \ (d(p, q) = 0, \text{ se } p = q), \tag{16}$$
$$d(p, q) = d(q, p), \tag{17}$$
$$d(p, r) \leq d(p, q) + d(q, r), \tag{18}$$

onde $p = (x_p, y_p)$, $q = (x_q, y_q)$, e $r = (x_r, y_r)$ são três pixels da imagem. As métricas mais usadas são:

- Euclideana: $d(p, q) = ((x_p - x_q)^2 + (y_p - y_q)^2)^{1/2}$,
- City-block: $d(p, q) = |x_p - x_q| + |y_p - y_q|$,
- Chessboard: $d(p, q) = \max\{|x_p - x_q|, |y_p - y_q|\}$.
- Chamfer: $d_{a,b}(p, q) = a * \max\{|x_p - x_q|, |y_p - y_q|\} + (b - a) * \min\{|x_p - x_q|, |y_p - y_q|\}$, onde a, b são constantes (e.g. $a = 5$ e $b = 7$).

1.3 Relação de Adjacência e Grafos

Uma relação de adjacência A é uma relação binária entre pixels, normalmente invariante à translação. Porém, A pode depender de propriedades locais da imagem, tais como cor e gradiente, e portanto, ser variante à translação. Dizemos que $A(p)$ é o conjunto dos pixels adjacentes ao pixel p de acordo com A. Isto é, $q \in A(p)$ é o mesmo que $(p, q) \in A$. Uma relação de adjacência leva, portanto, à definição de um grafo $G = (D_I, A)$ para a imagem $\hat{I} = (D_I, I)$. Exemplos:

- $(p, q) \in A$ se $d(p, q) \leq \rho$, onde d é distância Euclideana e ρ é um escalar,
- $(p, q) \in A$ se $q - p \in \{(-1, -1), (1, -1)\}$,

- $(p, q) \in A$ se $|x_p - x_q| + |y_p - y_q| \leq 1$ e $|I(p) - I(q)| \leq l$, onde l é um limiar de brilho.

Observe que $\rho = 1$ define vizinhança-4 (i.e. os pixels compartilham uma aresta), $\rho = \sqrt{2}$ define vizinhança-8 (i.e. os pixels compartilham um vértice ou uma aresta), e $\rho = \sqrt{5}$ faz com que pixels mais distantes sejam vizinhos no grafo. Esta relação é simétrica e invariante à translação. Note também que o segundo exemplo está relacionado com a definição de elemento estruturante planar usada em morfologia matemática, e portanto uma relação de adjacência pode ser assimétrica.

No grafo definido por A, um caminho π é uma seqüência de pixels adjacentes $< p_1, p_2, \ldots, p_n >$, onde $(p_i, p_{i+1}) \in A$, $i = 1, 2, \ldots, n-1$. O pixel p_1 é a origem $org(\pi)$ e $p_n = dst(\pi)$, e o caminho π é dito trivial se $\pi =< p_1 >$.

1.4 Relação de Conexidade

Um pixel q é conexo a um pixel p se existir um caminho de p a q no grafo definido por A. Dizemos que dois pixels são conexos-4 (conexos-8) se forem ligados por caminhos cujos pixels adjacentes são vizinhos-4 (vizinhos-8). Note que esta conexidade é simétrica, mas de uma forma geral a conexidade pode ser assimétrica.

1.5 Componente conexo

Um componente conexo na imagem $\hat{I} = (D_I, I)$ é um subconjunto de D_I, onde todos os pares (p, q) de pixels são conexos (i.e. existe um caminho de p a q e um caminho de q a p, que não necessariamente são os mesmos). Esses componentes são facilmente rotulados por busca em largura, no caso de conexidades simétricas. Observe que a adjacência pode ser assimétrica e a conexidade ainda assim ser simétrica.

Algoritmo geral de rotulação de componentes conexos com conexidade simétrica:

Entrada: Imagem cinza $\hat{I} = (D_I, I)$, relação de adjacência A, e limiar t.
Saída: Imagem rotulada $\hat{L} = (D_I, L)$, onde $L(p) = 0$ inicialmente.
Auxiliares: FIFO Q e variável inteira $l = 1$.

1. Para todo pixel $p \in D_I$, tal que $L(p) = 0$, faça
2. $\quad L(p) \leftarrow l$ e insira p em Q.
3. $\quad$ Enquanto $Q \neq \emptyset$ faça
4. $\quad\quad$ Remova p de Q.
5. $\quad\quad$ Para todo $q \in A(p)$, tal que $L(q) = 0$ e $|I(p) - I(q)| \leq t$, faça
6. $\quad\quad\quad L(q) \leftarrow L(p)$ e insira q em Q.
7. $\quad l \leftarrow l + 1$.

1.6 Objeto

Um objeto na imagem $\hat{I} = (D_I, I)$ é um subconjunto de D_I formado por um ou mais componentes conexos. Uma borda de objeto é um conjunto de pixels do seu interior que possui ao menos um pixel adjacente no exterior. Um objeto pode ser representado por suas bordas ou pelos pixels que compõem seu interior.

O número de componentes conexos NC, o número de buracos NB, o número de contornos internos, e o número de contornos externos são exemplos de propriedades topológicas de objeto. Essas propriedades podem ser usadas como descritores para análise. Um exemplo é o número de Euler definido por $NC - NB$.

2 Exercícios

1. Seja A uma relação de adjacência definida para todo par (p, q) de pixels que satisfaz $q - p \in \{(-1, -1), (1, 1), (-2, 3), (3, -5)\}$. Descreva o conjunto dos pixels q adjacentes ao pixel $p = (20, 30)$.

2. Dê um exemplo de conexidade simétrica e adjacência assimétrica em imagens binárias.

3. Considere um objeto definido pelo conjunto de pixels $X = \{(10, 10), (10, 11), (12, 11), (9, 12)\}$. Quantos componentes conexos-4 e quantos components conexos-8 este objeto possui? Quais são esses componentes e o número de Euler em cada caso?

4. Quais pixels estão a uma distância de city-block menor que 7 do pixel $(5, 5)$?

1 Transformações Geométricas

Uma transformação geométrica 2D é uma função que leva um ponto em outro ponto no espaço R^2. Neste curso, estamos interessados em transformações de rotação, translação e escalonamento aplicadas a pixels (pontos do Z^2).

A aplicação direta de uma transformação geométrica sobre os pixels de uma imagem leva a valores reais, que quando discretizados, podem gerar "buracos" na imagem transformada. A abordagem correta, portanto, requer a aplicação da transformação inversa seguida de reamostragem dos valores dos pixels. Esses conceitos são o objetivo desta aula.

No caso de imagens coloridas, a reamostragem se aplica para cada componente de cor separadamente.

1.1 Translação

A translação (t_x, t_y) aplicada a um ponto (x, y), e sua inversa são dadas em coordenadas homogêneas por:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{19}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \tag{20}$$

1.2 Rotação

A rotação θ no sentido horário aplicada a um ponto (x, y), e sua inversa são dadas em coordenadas homogêneas por:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{21}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \tag{22}$$

As equações acima são facilmente obtidas se considerarmos (x, y) e (x', y') em coordenadas polares: $(x, y) = (r, \alpha)$ e $(x', y') = (r, \alpha + \theta)$:

$$x' = r\cos(\theta + \alpha) \tag{23}$$

$$y' = r\sin(\theta + \alpha) \tag{24}$$

$$x = r\cos(\alpha) \tag{25}$$

$$y = r\sin(\alpha). \tag{26}$$

Desenvolvendo as Equações 23 e 24 temos

$$x' = r\cos(\alpha)\cos(\theta) - r\sin(\alpha)\sin(\theta) \quad (27)$$
$$y' = r\cos(\alpha)\sin(\theta) + r\sin(\alpha)\cos(\theta), \quad (28)$$

e substituindo as Equações 25 e 26 temos

$$x' = x\cos(\theta) - y\sin(\theta) \quad (29)$$
$$y' = x\sin(\theta) + y\cos(\theta). \quad (30)$$

Para evitar que a imagem rotacione em torno do eixo z (regra da mão direita), que penetra na tela, mais sim em torno do seu centro, ela deve ser inicialmente transladada de $(-M/2, -N/2)$ para a origem, rotacionada, e depois transladada de volta de $(M'/2, N'/2)$ de modo que todas as coordenadas de pixel sejam positivas. Observe que uma forma de garantir coordenadas positivas é definir a imagem final com tamanho $D \times D$, onde $D = \sqrt{N^2 + M^2}$. Neste caso, a transformação direta fica $T(D/2, D/2) \cdot R(\theta) \cdot T(-M/2, -N/2)$ e a inversa fica $T(M/2, N/2) \cdot R(-\theta) \cdot T(-D/2, -D/2)$, onde T é translação e R é rotação.

1.3 Escalonamento

A transformação de escalonamento (s_x, s_y) aplicada a um ponto (x, y), e sua inversa são dadas em coordenadas homogêneas por:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (31)$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1/s_x & 0 & 0 \\ 0 & 1/s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \quad (32)$$

Note que a imagem aumenta de tamanho para valores de escalonamento maiores que 1 e reduz de tamanho para valores entre 0 e 1, e que valores negativos refletem a imagem em torno do eixo correspondente.

1.4 Reamostragem

A reamostragem é o processo de estimação dos valores dos pixels usando um novo intervalo (Δ'_x, Δ'_y) de amostragem da função $\vec{I}$ de uma imagem $\hat{I} = (D_I, \vec{I})$. Esta transformação gera uma nova imagem $\hat{I}' = (D_{I'}, \vec{I'})$ com tamanho e resolução espacial diferentes.

1.4.1 Reamostragem de um único pixel

Suponha, por exemplo, que desejamos reamostrar o valor do pixel $r = (x', y')$, $y' = y$, entre os pixels $p = (x, y)$ e $q = (x + 1, y)$ de uma imagem $\hat{I} = (D_I, I)$ amostrada com intervalos (Δ_x, Δ_y). Se a função I variar linearmente ao longo do eixo x teremos:

$$I(r) \quad = \quad \frac{(\Delta_x - d) \times I(p) + d \times I(q)}{\Delta_x}, \tag{33}$$

onde d é a distância entre r e p. Estando $r = (x', y')$ entre quatro pixels mais próximos, $p = (x, y)$, $q = (x + 1, y)$, $u = (x, y + 1)$, e $v = (x + 1, y + 1)$, a Equação 33 é aplicada na horizontal; entre p e q gerando $I(r_1)$, $r_1 = (x', y)$, e entre u e v gerando $I(r_2)$, $r_2 = (x', y + 1)$; e depois na vertical; entre r_1 e r_2 gerando $I(r)$ (reamostragem ou interpolação bilinear).

Portanto, para gerar uma imagem $\hat{I}'$ por transformação geométrica de $\hat{I}$, para cada coordenada inteira (x', y') de um pixel de $\hat{I}'$, nós aplicamos a transformação inversa seguida de reamostragem bilinear, conforme descrito acima.

1.4.2 Reamostragem de uma imagem

Neste caso, podemos reamostrar a imagem de entrada $\hat{I}$, linha por linha usando a Equação 33, gerando uma imagem intermediária, e depois, coluna por coluna, usando a Equação 33, para gerar a imagem final $\hat{I}'$. A imagem $\hat{I}'$ terá dimensões $M' = \frac{M\Delta_y}{\Delta'_y}$ e $N' = \frac{N'\Delta_x}{\Delta'_x}$.

2 Exercícios

Note que os conceitos vistos nesta seção são facilmente estendidos para 3D e para imagens coloridas.

1. Implemente rotinas de rotação e escalonamento para imagens cinza.

2. Qual deve ser o tamanho mínimo da imagem gerada em cada caso, rotação e escalonamento, para não haver perda de pedaços da imagem original?

3. Desenvolva as matrizes de rotação em 3D ($R(\theta_x)$, $R(\theta_y)$, $R(\theta_z)$), usando a explicação vista para o caso 2D.

1 Transformações Radiométricas

Uma transformação radiométrica é um mapeamento de intensidade aplicado pixel a pixel de uma imagem $\hat{I} = (D_I, I)$, gerando uma imagem $\hat{I'} = (D_{I'}, I')$, $D_{I'} = D_I$, com brilho e contraste diferentes. O brilho está associado à intensidade de cinza e o contraste à variação de tons de cinza na imagem.

Essas transformações visam aumento de brilho e contraste, quando a imagem está muito escura, e redução de contraste, quando a capacidade do monitor de vídeo, por exemplo, é menor que a resolução radiométrica da imagem.

1.1 Variação de contraste linear

Sejam $[l_1, l_2]$, $l_1 \leq l_2$, e $[l'_1, l'_2]$ dois intervalos de cinza no conjunto de valores de I e I'. O aumento de contraste linear (stretching linear) é definido para cada pixel $p \in D_I$ por:

$$I'(p) = \begin{cases} l'_1, & \text{se } I(p) < l_1, \\ \frac{(l'_2 - l'_1)}{(l_2 - l_1)}(I(p) - l_1) + l'_1, & \text{se } l_1 \leq I(p) < l_2, \\ l'_2, & \text{se } I(p) \geq l_2. \end{cases} \tag{34}$$

Alguns casos particulares são:

- Normalização: $l'_2 = H$, $l'_1 = 0$, $l_1 = l_{\min}$, e $l_2 = l_{\max}$, onde $l_{\min}$ e $l_{\max}$ são os valores mínimo e máximo da imagem.
- Negativo: $l'_2 = l_{\min}$, $l'_1 = l_{\max}$, $l_1 = l_{\min}$, e $l_2 = l_{\max}$, onde $l_{\min}$ e $l_{\max}$ são os valores mínimo e máximo da imagem.
- Contraste & Brilho (width & level): $l'_2 = H$, $l'_1 = 0$, e $l_1 < l_2$, onde H é o maior valor de brilho que o monitor suporta (e.g. $H = 255$ para monitores de 8 bits), level é $\frac{l_1 + l_2}{2}$, e width é $l_2 - l_1$. O level altera o brilho e width altera o contraste.
- Limiarização (thresholding): $l'_2 = H$, $l'_1 = 0$, e $l_1 = l_2$.

Note que, apesar do stretching visar normalmente o aumento de contraste, este pode ser reduzido, caso H seja menor que $l_{\max} - l_{\min}$.

1.2 Variação de contraste exponencial

Existem dois casos de interesse:

- $I'(p) = l_{\max} \exp(\frac{I(p) - l_{\min}}{l_{\max} - l_{\min}}) - l_{\max}$ e
- $I'(p) = H \exp(\frac{-(I(p) - \mu)^2}{2\sigma^2})$.

O primeiro visa o aumento de contraste e brilho em todo intervalo $[l_{\min}, l_{\max}]$, e o segundo aumenta o contraste em torno de um valor μ (e.g. modificação da função gradiente para melhorar o desempenho de um algoritmo de segmentação).

1.3 Variação de contraste logaritmico

A visualização da imagem de magnitude da transformada de Fourier, por exemplo, requer normalmente uma redução de contraste do tipo

$$I'(p) = H \log(1 + \left|\vec{I}(p)\right|), \tag{35}$$

onde $\vec{I} = \{I_1, I_2\}$ contém a parte real I_1 e a imaginária I_2 do espectro.

1.4 Variações de brilho e contraste para imagens coloridas

Transformações radiométricas em imagens coloridas devem preservar a informação de matiz. Neste caso, as transformações acima podem ser aplicadas na imagem de brilho (ou de saturação) usando algum espaço descorrelacionado: HSV, IHS, ou YCbCr.

2 Exercícios

1. Exemplifique a imagem de um objeto com baixo contraste e alto brilho, e desenhe o seu histograma.

2. Aplique um aumento de contraste linear na imagem da questão anterior, mostre a imagem resultante e seu histograma.

3. Qual a diferença entre os histogramas de uma imagem clara, de uma imagem escura, com pouco contraste, e com muito contraste?

4. Uma função logística é dada por $I'(p) = \frac{H}{1+\exp(-(I(p)-\mu))}$. Que tipos de variação de contraste e brilho ela proporciona se $\mu = 0$ e $H < l_{\max}$?

5. Implemente uma rotina para aplicar variação de contraste linear em imagens RGB, usando as conversões RGB para HSV e HSV para RGB.

1 Transformações Radiométricas (cont.)

O histograma acumulado de uma imagem cinza $\hat{I}$ é uma função $h_a(l)$ que produz o valor acumulado do histograma $h(l)$ da imagem para cada nível de cinza $0 \leq l \leq L$ ($0 \leq l_{\min}$ e $L \geq l_{max}$).

$$h_a(l) = \sum_{l'=0}^{l} h(l') \tag{36}$$

As transformações radiométricas descritas abaixo exploram este conceito para histograma normalizado.

1.1 Aumento de contraste por equalização

Considere os valores dos pixels da imagem $\hat{I}$ normalizados entre 0 e 1, i.e. $0 \leq I(p) \leq 1$. A equalização $E(I)$ é uma transformação que satisfaz as seguintes condições:

- É bijetora e monotonicamente crescente em $[0, 1]$, e
- É limitada, $0 \leq E(I)\Big|_{I(p)} \leq 1$, para $0 \leq I(p) \leq 1$.

Esta transformação gera uma imagem $\hat{I}'$, tal que $0 \leq I'(p) \leq 1$ e $E^{-1}(I') = I$, da seguinte forma:

$$I'_c(p) = E(I_c)\Big|_{I_c(p)} = \int_0^{I_c(p)} h_c(l)dl \tag{37}$$

no caso contínuo, onde h_c é a densidade de probabilidade de I_c, e

$$I'(p) = E(I)\Big|_{I(p)} = \sum_{l=0}^{I(p)} h(l) = h_a(I(p)), \tag{38}$$

no caso discreto, onde h é o histograma normalizado de I. Note, portanto, que esta transformação utiliza os valores do histograma acumulado h_a.

A equalização visa uma distribuição de probabilidade uniforme para o brilho dos pixels de $\hat{I}'$. Observe que $\frac{dI'_c}{dl} = h_c(l)$, $0 \leq l \leq 1$, e que a probabilidade de $I_c(p) = E^{-1}(I'_c)\Big|_{I'_c(p)}$ está em um intervalo de largura dl em torno de $I_c(p)$ é $h_c(l)dl\Big|_{l=E^{-1}(I'_c)\Big|_{I'_c(p)}}$. Esta probabilidade também é igual a probabilidade de $I'_c(p)$ está em um intervalo de largura dI'_c em torno de $I'_c(p)$, que é $h'_c(p)dI'_c$. Portanto, a densidade de probabilidade $h'_c(l) = h_c(l)\frac{dl}{dI'_c}\Big|_{l=E^{-1}(I'_c)\Big|_{I'_c(p)}} = 1$, $0 \leq l \leq 1$ é uma densidade de probabilidade uniforme, e $h'(l)$ é a distribuição de probabilidade de I' que tende a ser uniforme.

No caso de imagens coloridas, a equalização é aplicada no componente de brilho (ou de saturação) da mesma forma que descrito na aula anterior.

Note que, a imagem final deve ser depois normalizada entre 0 e H, onde H é o brilho máximo do monitor, para fins de visualização.

1.2 Casamento de histogramas

Outra forma de modificar o histograma de uma imagem é casando seu histograma com o de outra. Esta transformação procura fazer com que duas imagens tenham o mesmo histograma (ou o mais parecido possível), e tem diversas finalidades. Por exemplo, o registro de imagens obtidas de uma mesma região em épocas diferentes, quando baseado na intensidade dos pixels, requer o casamento de histogramas como pré-processamento. O casamento de histogramas também pode ser realizado para melhorar o brilho e o contraste de uma imagem usando outra como referência.

O casamento do histograma de uma imagem $\hat{I}$ com o de uma imagem $\hat{J}$ gera uma $\hat{I}'$ da seguinte forma. Supõe-se que após equalização de I e de J, os histogramas resultantes são iguais e uniformes. Assim, a inversa da equalização $E_J(J)$ de J aplicada à equalização $E_I(I)$ de I, deve gerar uma imagem I' com histograma parecido com o de J.

$$I' = E_J^{-1}(E_I(I)) \tag{39}$$

O casamento entre imagens coloridas requer a transformação RGB para HSV de ambas imagens, o casamento entre os componentes de brilho, e a volta de HSV para RGB da imagem desejada.

2 Exercícios

1. Equalize a imagem cujos pixels têm valores $1, 2, 3, 2, 1, 4, 2, 4, 4$, mostrando o histograma acumulado e os valores finais.

2. Aplique o casamento de histogramas entre uma imagem, cujos valores são $2, 2, 2, 3, 4, 3, 4, 5, 6$, e a imagem da questão anterior.

3. Implemente rotinas de equalização e casamento de histogramas.

3 Convolução e Correlação

Sejam $f(x)$ e $g(x)$ duas funções reais [1], contínuas, limitadas e finitas em x (e.g. um sinal de voz, onde x é o tempo.). A convolução e a correlação entre elas são definidas como:

$$f(x) * g(x) = \int_{-\infty}^{+\infty} f(x')g(x-x')dx' \tag{40}$$

$$f(x) \odot g(x) = \int_{-\infty}^{+\infty} f(x')g(x+x')dx'. \tag{41}$$

Suponha, por exemplo, que

$$f(x) = \begin{cases} 2, & \text{se } |x| \leq 2, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{42}$$

$$g(x) = \begin{cases} 2x, & \text{se } 0 \leq x \leq 2, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{43}$$

A convolução $h(x) = f(x) * g(x)$ envolve quatro etapas.

1. Reflexão $g(-x')$ em x':

$$g(-x') = \begin{cases} -2x', & \text{se } -2 \leq x' \leq 0, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{44}$$

2. Deslocamento $g(x-x')$ de x em x':

$$g(x-x') = \begin{cases} -2x'+2x, & \text{se } x-2 \leq x' \leq x, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{45}$$

3. Multiplicação $f(x')g(x-x')$:

$$f(x')g(x-x') = \begin{cases} -4x'+4x, & \text{se } -2 \leq x' \leq x \text{ e } -2 \leq x < 0, \\ -4x'+4x, & \text{se } x-2 \leq x' \leq x \text{ e } 0 \leq x < 2, \\ -4x'+4x, & \text{se } x-2 \leq x' \leq 2 \text{ e } 2 \leq x \leq 4, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{46}$$

4. Integralização $h(x) = \int_{-\infty}^{+\infty} f(x')g(x+x')dx'$ (i.e. a área do produto $f(x')g(x-x')$ é o valor da convolução para cada coordenada x):

$$h(x) = \begin{cases} 2x^2+8x+8, & \text{se } -2 \leq x < 0, \\ 8, & \text{se } 0 \leq x < 2, \\ -2x^2+8x, & \text{se } 2 \leq x \leq 4, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{47}$$

[1] Se as funções fossem complexas, $f(x) \odot g(x) = \int_{-\infty}^{+\infty} f^*(x')g(x+x')dx'$, onde $f^*(x')$ é o complexo conjugado de $f(x')$.

A correlação é calculada de forma similar e representa a similaridade entre $f(x)$ e $g(x)$, medida usada em várias aplicações em processamento de imagens, quando estendida para 2D, tais como registro de imagens e estimação de movimento em vídeo.

Observe que a convolução obedece o princípio da superposição (distribuição e escalamento).

$$(af_1(x) + bf_2(x)) * g(x) \quad = \quad a(f_1(x) * g(x)) + b(f_2(x) * g(x)) \tag{48}$$

Este princípio é uma propriedade fundamental de sistemas lineares, onde $g(x)$ é a função de transferência do sistema (limitada e finita), e para toda entrada $f(x)$, limitada e finita, temos $f(x) * g(x)$ como resposta do sistema linear.

3.1 Convolução com a função impulso

A função impulso, ou delta de Dirac, é definida por:

$$\delta(x) \quad = \quad \begin{cases} \infty, & \text{se } x = 0, \text{ e} \\ 0, & \text{no caso contrário,} \end{cases} \tag{49}$$

$$\tag{50}$$

tal que

$$\int_{-\infty}^{+\infty} \delta(x')dx' = 1. \tag{51}$$

A convolução de uma função $f(x)$ com $\delta(x)$ é $f(x)$, com $\delta(x - m \cdot \Delta_x)$, $m = 0, 1, \ldots, M-1$, é $f(x - m \cdot \Delta_x)$, e com $\sum_{m=0}^{M-1} \delta(x - m \cdot \Delta_x)$ (trem de impulsos) é $\sum_{m=0}^{M-1} f(x - m \cdot \Delta_x)$ (função f repetida ao longo de x a cada intervalo Δ_x).

Note que podemos descobrir a função de transferência de um sistema linear aplicando um impulso como entrada.

3.2 Amostragem

O processo de amostragem de uma função $f(x)$, limitada e finita, a intervalos Δ_x pode ser modelado como

$$f(x) \cdot \sum_{m=0}^{M-1} \delta(x - m \cdot \Delta_x) = \sum_{m=0}^{M-1} f(m \cdot \Delta_x)\delta(x - m \cdot \Delta_x), \tag{52}$$

onde $\delta(x) = 1$, se $x = 0$, e 0 no caso contrário, é o impulso unitário, gerando M amostras $f(m \cdot \Delta_x)$, $m = 0, 1, \ldots, M-1$, de $f(x)$ espaçadas de intervalo Δ_x.

Isto é, um sinal discreto $I(x)$, $x = 0, 1, \ldots, M-1$, é um trem de impulsos de altura $f(m \cdot \Delta_x)$, $m = 0, 1, \ldots, M-1$, onde f é a função contínua amostrada (e.g. a corrente elétrica que representa um sinal de voz— neste caso, Δ_x é substituído por um intervalo de tempo Δ_t).

3.3 Convolução discreta

A convolução entre dois sinais discretos (finitos e limitados) é dada por:

$$H(x) = I(x) * J(x) = \sum_{x'=-\infty}^{+\infty} I(x')J(x - x'), \tag{53}$$

onde $x \in Z$.

Por exemplo:

$$I(x) = \begin{cases} 2, & \text{se } x \in \{-2, -1, 0, 1, 2\}, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{54}$$

$$J(x) = \begin{cases} 2x, & \text{se } x \in \{0, 1, 2\}, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{55}$$

Similarmente, a convolução $H(x) = I(x) * J(x)$ pode ser calculada em quatro etapas tais que:

$$J(x - x') = \begin{cases} -2x' + 2x, & \text{se } x' \in \{x - 2, x - 1, x\}, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{56}$$

$$H(x) = \begin{cases} \sum_{x'=-2}^{x} -4x' + 4x, & \text{se } x \in \{-2, -1, 0\}, \\ \sum_{x'=x-2}^{x} -4x' + 4x, & \text{se } x \in \{1, 2\}, \\ \sum_{x'=x-2}^{2} -4x' + 4x, & \text{se } x \in \{3, 4\}, \\ 0, & \text{no caso contrário.} \end{cases} \tag{57}$$

Observe que se $I(x)$ possui comprimento M_1 e $J(x)$ possui comprimento M_2, então $H(x) = I(x) * J(x)$ terá comprimento $M_1 + M_2 - 1$.

3.4 Filtragem por convolução discreta

Considere $I(x)$, o sinal discreto da Equação 54, e $J(x)$ um sinal discreto dado por

$$J(x) = \begin{cases} 1/3, & \text{se } x \in \{-1, 0, 1\}, \text{ e} \\ 0, & \text{no caso contrário.} \end{cases} \tag{58}$$

Podemos suavizar as variações abruptas que ocorrem em $I(x)$, para $x = -2$ e $x = 2$, calculando a convolução discreta $H(x) = I(x) * J(x)$. Observe que $J(x)$ atua como a função de transferência de um filtro linear discreto.

3.5 Extensão para imagens

No caso de imagens digitais, os resultados acima podem ser estendidos para:

$$H(x, y) = I(x, y) * J(x, y) = \sum_{y'=-\infty}^{+\infty} \sum_{x'=-\infty}^{+\infty} I(x', y')J(x - x', y - y'), \tag{59}$$

$$H(x, y) = I(x, y) \odot J(x, y) = \sum_{y'=-\infty}^{+\infty} \sum_{x'=-\infty}^{+\infty} I(x', y')J(x + x', y + y'), \tag{60}$$

onde $I(x, y)$ e $J(x, y)$ são os valores dos pixels na imagem $\hat{I}$ e na imagem $\hat{J}$. Observe que $J(x, y)$ é refletida em x' e em y', e depois deslocada da esquerda para direita e de cima para baixo durante a convolução.

4 Exercícios

1. Calcule o resultado da convolução apresentada na seção 3.4.
2. Implemente uma função para calcular a convolução entre duas imagens.
3. Apresente um kernel de convolução para detectar pontos de variação brusca em sinais discretos.
4. Calcule a convolução $g(x) * f(x)$ das funções contínuas apresentadas nesta aula para mostrar que o resultado é o mesmo que $f(x) * g(x)$.
5. Calcule a correlação $f(x) \odot f(x - 4)$, e interprete o resultado.

1 Filtragem no Espaço

Considere um sinal discreto e limitado $I(x)$, $x = 0, 1, \ldots, M_1 - 1$, com amostras espaçadas de Δ_x; e um mapeamento escalar e limitado $J(x)$, $x = 0, 1, \ldots, M_2 - 1$ (denominado kernel, máscara ou template). A filtragem linear de $I(x)$ por $J(x)$ pode ser calculada pela convolução discreta

$$H(x) \;=\; I(x) * J(x) = \sum_{x'=0}^{M-1} I(x')J(x - x'), \tag{61}$$

onde $x = 0, 1, \ldots, M-1$ e $M = M_1+M_2-1$, porque $H(x)$ é zero fora do intervalo $x \in [0, M-1]$. No caso de imagens, porém, a origem da máscara está normalmente no seu centro $(\frac{M_2}{2}, \frac{N_2}{2})$ e a imagem está deslocada de $(\frac{M_2}{2}, \frac{N_2}{2})$ para direita e para baixo com relação à origem da imagem resultante.

$$H(x, y) \;=\; \sum_{y'=0}^{N-1} \sum_{x'=0}^{M-1} I(x' - \frac{M_2}{2}, y' - \frac{N_2}{2})J(x - x' - \frac{M_2}{2}, y - y' - \frac{N_2}{2}), \tag{62}$$

onde $\hat{I} = (D_I, I)$, $|D_I| = N_1 \times M_1$, $\hat{J} = (D_J, J)$, $|D_J| = N_2 \times M_2$, $M = M_1 + M_2 - 1$, $N = N_1 + N_2 - 1$, e $\hat{H} = (D_H, H)$, $|D_H| = N \times M$.

Algoritmo para filtragem de imagens por convolução discreta:

Entrada: Imagem cinza $\hat{I} = (D_I, I)$, $D_I = \{(\frac{M_2}{2}, \frac{N_2}{2}), (\frac{M_2}{2} + 1, \frac{N_2}{2}), \ldots, (\frac{M_2}{2} + M_1 - 1, \frac{N_2}{2} + N_1 - 1)\}$ e máscara $\hat{J} = (D_J, J)$, $D_J = \{(-\frac{M_2}{2}, -\frac{N_2}{2}), \ldots, (0, 0), \ldots, (\frac{M_2}{2}, \frac{N_2}{2})\}$.
Saída: Imagem cinza $\hat{H} = (D_H, H)$, tal que $H(x, y) = I(x - \frac{M_2}{2}, y - \frac{N_2}{2}) * J(x + \frac{M_2}{2}, y + \frac{N_2}{2})$, $D_H = \{(0, 0), (0, 1), \ldots, (M - 1, N - 1)\}$, $M = M_1 + M_2 - 1$ e $N = N_1 + N_2 - 1$.

1. Calcule a reflexão $\hat{J'} = (D'_J, J')$ mapeando todo $(x, y) \in D_J$ para $(-x, -y) \in D'_J$ e $J'(-x, -y) \leftarrow J(x, y)$.
2. Calcule a relação de adjacência A, tal que $q \in A(p)$ se $q - p \in D'_J$.
3. Para todo pixel $p \in D_H$, faça
4. $\quad H(p) \leftarrow 0$.
5. $\quad$ Para todo pixel $q \in A(p)$, tal que $q \in D_I$, faça
6. $\quad\quad H(p) \leftarrow H(p) + I(q) * J'(q - p)$.

Note que o algoritmo acima funciona também para relações de adjacência assimétricas.

Melhoramentos na imagem podem ser realizados através de diferentes kernels e de diferentes tamanhos. Alguns exemplos são apresentados a seguir.

1.1 Suavização

Filtros de suavização (blurring) reduzem ruído de alta freqüência, mas borram as bordas da imagem.

- Filtro Média

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} \tag{63}$$

- Filtro Gaussiano

$$\frac{1}{16} * \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{64}$$

1.2 Realce

Filtros de realce aumentam o contraste nas bordas da imagem, mas podem amplificar o ruído.

- Gradiente de Sobel

$$S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{65}$$

  As máscaras S_x e S_y realçam bordas nas direções x e y, respectivamente, tal que S_x é usado para realçar bordas verticais e S_y para bordas horizontais. $I(x, y) * S_x(x, y)$ corresponde à derivada dI/dx e $I(x, y) * S_y(x, y)$ à dI/dy formando um vetor gradiente $\vec{G}(p) = dI(p)/dx \cdot \vec{i} + dI(p)/dy \cdot \vec{j}$ que indica a direção e o sentido de maior variação de brilho em torno de p. A magnitude $|\vec{G}(p)|$ do vetor gradiente é muito usada em segmentação.

- Gradiente de Roberts

  O gradiente de Roberts realça bordas nas direções diagonais ($45°$ e $-45°$), considerando pares de pixels em torno de $(x + 1/2, y + 1/2)$.

$$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \tag{66}$$

- Outros filtros direcionais

– Norte

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix} \tag{67}$$

– Nordeste

$$\begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix} \tag{68}$$

– Leste

$$\begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix} \tag{69}$$

– Sudeste

$$\begin{bmatrix} -1 & -1 & 1 \\ -1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{70}$$

– Sul

$$\begin{bmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{71}$$

– Sudoeste

$$\begin{bmatrix} 1 & -1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & 1 \end{bmatrix} \tag{72}$$

– Oeste

$$\begin{bmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix} \tag{73}$$

– Noroeste

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & -1 \\ 1 & -1 & -1 \end{bmatrix} \tag{74}$$

- Filtros Laplacianos

  Filtros laplacianos são não-direcionais e correspondem à derivada de segunda ordem $d^2I/dx^2 + d^2I/dy^2$.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} \tag{75}$$

- Sharpness

  Esses filtros realçam detalhes finos combinando o realce de bordas com a imagem original.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{76}$$

  Outros exemplos são:

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 17 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix} \tag{77}$$

1.3 Filtragem não-linear

- Filtro Mediana

  Considerando uma adjacência $A(p)$ em torno de cada pixel $p \in D_I$, ordena-se os pixels q adjacentes a p pelo valor crescente de $I(q)$, selecionando o pixel q' de valor $I(q')$ mediano e gerando uma nova imagem $\hat{J} = (D_I, J)$, onde $J(p) = I(q')$. Esta operação elimina ruídos do tipo speckle.

- Filtro Moda

  Considerando uma adjacência $A(p)$ em torno de cada pixel $p \in D_I$, calcula-se o histograma dos valores dos pixels q adjacentes a p, selecionando o valor $moda$ (valor $moda = I(q)$ de maior ocorrência) e gerando uma nova imagem $\hat{J} = (D_I, J)$, onde $J(p) = moda$. Esta operação é muito usada para eliminar pequenas regiões classificadas erroneamente em imagens rotuladas (e.g. mapas temáticos).

Outros exemplos são filtros morfológicos que veremos mais adiante.

No caso de imagens coloridas, a filtragem espacial pode ser aplicada em cada componente isoladamente. O valor máximo da magnitude do gradiente em cada componente gera, por exemplo, uma imagem de gradiente boa para segmentação.

2 Exercícios

1. Considere uma imagem 3×3 cujos valores dos pixels são $1, 2, 2, 3, 0, 1, 2, 2, 3$. Qual é o resultado da convolução discreta entre esta imagem e as máscaras de Roberts? Mostre a imagem de magnitude do vetor gradiente?

2. Considere a imagem de um quadrado branco (brilho 255) centrado no meio da imagem de fundo preto (brilho 0). Qual são os valores resultantes da convolução desta imagem com as máscaras de Sobel S_x e S_y em cada vértice do quadrado, em cada aresta do quadrado, e no interior do quadrado?

3. Escolha um filtro de suavização e um de realce, calcule a convolução discreta entre eles e interprete o resultado.

4. Qual é o resultado de uma filtragem mediana 3×3 aplicada à imagem da questão 1?

5. Implemente uma função para calcular filtragem mediana dada uma adjacência A.

1 Transformada de Fourier

Seja $f(x)$ uma função real e contínua, sua transformada de Fourier $\vec{F}(u) = \{F_{Re}(u), F_{Im}(u)\} = F_{Re}(u) + jF_{Im}(u)$ e a inversa são dadas por

$$\vec{F}(u) = \int_{-\infty}^{+\infty} f(x) \exp^{-j2\pi ux}\, dx \tag{78}$$

$$f(x) = \int_{-\infty}^{+\infty} \vec{F}(u) \exp^{j2\pi ux}\, du. \tag{79}$$

Note que, mesmo considerando apenas funções $f(x)$ reais (caso particular), a transformada é normalmente complexa, exceto quando $f(x)$ é uma função par (i.e. $f(x) = f(-x)$).

Alguns exemplos úteis são:

$$f(x) = \begin{cases} 1, & \text{se } |x| \leq x_o, \text{ e} \\ 0, & \text{no c.c.} \end{cases} \leftrightarrow \quad F(u) = 2x_o Sa(2\pi x_o u) \tag{80}$$

$$f(x) = 2u_o Sa(2\pi u_o x) \leftrightarrow \quad F(u) = \begin{cases} 1, & \text{se } |u| \leq u_o, \text{ e} \\ 0, & \text{no c.c.} \end{cases} \tag{81}$$

$$f(x) = \sum_{m=-\infty}^{+\infty} \delta(x - m\Delta_x) \leftrightarrow \quad F(u) = \tfrac{1}{\Delta_x} \textstyle\sum_{m=-\infty}^{+\infty} \delta(u - \frac{m}{\Delta_x}) \tag{82}$$

$$f(x) = \cos(2\pi u_o x) \leftrightarrow \quad F(u) = \tfrac{1}{2}\left[\delta(u - u_o) + \delta(u + u_o)\right] \tag{83}$$

$$\frac{d^{(n)}f}{dx}(x) \leftrightarrow \quad (2\pi uj)^n \vec{F}(u) \tag{84}$$

$$f(x) = \sin(2\pi u_o x) \leftrightarrow \quad \vec{F}(u) = \tfrac{j}{2}\left[\delta(u + u_o) - \delta(u - u_o)\right] \tag{85}$$

$$h(x) = f(x) * g(x) \leftrightarrow \quad \vec{H}(u) = \vec{F}(u)\vec{G}(u) \tag{86}$$

$$h(x) = f(x)g(x) \leftrightarrow \quad \vec{H}(u) = \vec{F}(u) * \vec{G}(u) \tag{87}$$

onde $Sa(\theta) = \frac{\sin(\theta)}{\theta}$, $g(x)$ é uma função real e contínua e os dois últimos exemplos são chamados teoremas da convolução no espaço e na freqüência, respectivamente.

1.1 Modulação em freqüência

Suponha que $F(u)$ é o espectro de freqüência de um sinal de voz, o qual está limitado em faixa $[-u_o, u_o]$ (i.e. $F(u) \neq 0$, se $|u| \leq u_o$, e $F(u) = 0$, no c.c.). Sua transmissão em um canal na freqüência u_p MHz, $u_p >> u_o$, requer que $f(t)$ seja multiplicado por uma portadora $\cos(2\pi u_p t)$ (ou $\sin(2\pi u_p t)$), o que pelas Equações 83 e 87 faz com que seu espectro seja deslocado para as freqüências u_p e $-u_p$ MHz:

$$\frac{1}{2}F(u - u_p) + \frac{1}{2}F(u + u_p). \tag{88}$$

Este processo, conhecido como modulação em freqüência, é utilizado em rádio FM analógica para transmitir simultaneamente vários canais de rádio a freqüências u_p diferentes.

1.2 Transformada de Fourier Discreta

Para facilitar, considere inicialmente $f(x)$ um sinal real, contínuo, par e limitado em faixa $[-u_o, u_o]$ (i.e. ilimitado no espaço). Se amostrarmos $f(x)$ a intervalos Δ_x, teremos pela combinação das Equações 82 e 87:

$$f_a(x) = f(x) \sum_{m=-\infty}^{+\infty} \delta(x - m\Delta_x) \leftrightarrow \quad F_a(u) = \tfrac{1}{\Delta_x} \textstyle\sum_{m=-\infty}^{+\infty} F(u - \frac{m}{\Delta_x}). \tag{89}$$

Isto é, o espectro de freqüência $F_a(u)$ do sinal amostrado é periódico com período $\frac{1}{\Delta_x}$.

Observe que podemos recuperar o sinal original a partir do espectro do sinal amostrado, ou melhor, de suas amostras no espaço. Basta multiplicar $F_a(u)$ por uma função $\Delta_x G(u)$, onde $G(u)$ é dada pela Equação 81. Pela Equação 86, esta operação equivale à convolução entre $f_a(x)$ e $\Delta_x g(x)$, onde $g(x) = 2u_o Sa(2\pi u_o x)$, que resulta em:

$$f(x) \;\; = \;\; 2\Delta_x u_o \sum_{m=-\infty}^{+\infty} f(m\Delta_x) Sa(2\pi u_o (x - m\Delta_x)) \tag{90}$$

Esta equação é conhecida como fórmula da interpolação, pois podemos utilizá-la com este propósito.

Observe também que se o intervalo Δ_x de amostragem fosse tal que $\frac{1}{\Delta_x} < 2u_o$, então o sinal original não poderia ser recuperado devido à superposição no espectro periódico do sinal amostrado (aliasing). Portanto, quanto menor for o intervalo de amostragem, maior será o intervalo $\frac{1}{\Delta_x}$ de repetição. Idealmente $\frac{1}{\Delta_x} \geq 2u_o$ para haver a recuperação do sinal original (Teorema de Nyquist).

Observe também que é computacionalmente inviável trabalhar com $f_a(x)$ ilimitada. Sua limitação $f'_a(x)$ no espaço entre $[\frac{-M\Delta_x}{2}, \frac{M\Delta_x}{2}]$ é obtida pela multiplicação de $f_a(x)$ por uma função $g(x) = 1$, se $|x| \leq \frac{M\Delta_x}{2}$, e 0 no c.c.. Pelas Equações 80 e 87, isto equivale à convolução de $F_a(u)$ com $G(u) = M\Delta_x Sa(\pi M \Delta_x u)$ gerando $F'_a(u)$ contínuo e periódico. Da mesma forma, é computacionalmente inviável trabalhar com um espectro contínuo. O espectro $F'_a(u)$ deve então ser amostrado a intervalos $\Delta_u = \frac{1}{M\Delta_x}$ gerando um espectro discreto e periódico $I_p(u)$. Pelas Equações 82 e 86, esta amostragem faz com que a inversa de $I_p(u)$ seja um sinal discreto e periódico $I_p(x)$, com período $M\Delta_x$.

A transformada de Fourier discreta $\vec{I}(u)$ de um sinal $I(x)$ provém dos coeficientes da série de Fourier discreta $\vec{I}_p(u)$ da seqüência periódica e discreta $I_p(x)$,

$$\vec{I}_p(u) \;\; = \;\; \sum_{x=0}^{M-1} I_p(x) \exp^{\frac{-j2\pi ux}{M}} \tag{91}$$

$$\vec{I}(u) \;\; = \;\; \begin{cases} \vec{I}_p(u), & u = 0, 1, \ldots, M-1 \\ 0, & \text{no c.c.} \end{cases} \tag{92}$$

$$I_p(x) \;\; = \;\; \frac{1}{M} \sum_{u=0}^{M-1} \vec{I}_p(u) \exp^{\frac{j2\pi ux}{M}} \tag{93}$$

$$I(x) \;\; = \;\; \begin{cases} I_p(x), & x = 0, 1, \ldots, M-1 \\ 0, & \text{no c.c.} \end{cases} \tag{94}$$

onde $x = 0\Delta_x, 1\Delta_x, \ldots, (M-1)\Delta_x$ no espaço e $u = 0\Delta_u, 1\Delta_u, \ldots, (M-1)\Delta_u$ na freqüência são representados de forma adimensional como $x = 0, 1, \ldots, M-1$ e $u = 0, 1, \ldots, M-1$. No caso de uma imagem $\hat{I} = (D_I, I)$ com $M \times N$ pixels teremos:

$$\vec{I}_p(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} I_p(x,y)\exp^{\left[-j2\pi(\frac{ux}{M}+\frac{vy}{N})\right]} \quad (95)$$

$$\vec{I}(u,v) = \begin{cases} \vec{I}_p(u,v), & u = 0,1,\ldots,M-1 \text{ e } v = 0,1,\ldots,N-1 \\ 0, & \text{no c.c.} \end{cases} \quad (96)$$

$$I_p(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} \vec{I}_p(u,v)\exp^{\left[j2\pi(\frac{ux}{M}+\frac{vy}{N})\right]} \quad (97)$$

$$I(x,y) = \begin{cases} I_p(x,y), & x = 0,1,\ldots,M-1 \text{ e } y = 0,1,\ldots,N-1 \\ 0, & \text{no c.c.} \end{cases} \quad (98)$$

Note que a imagem e a transformada de Fourier discreta iniciam em $(0,0)$ e vão até $(M-1, N-1)$. Portanto, a visualização do espectro no centro da imagem requer uma translação de $(-M/2, -N/2)$, e no caso da magnitude, temos ainda uma transformação radiométrica logaritmica como descrito na aula 6.

As freqüências digitais $\Omega_x = 2\pi u$ e $\Omega_y = 2\pi v$ em radianos por unidade de comprimento também são representadas como $\omega_x = \Omega_x\Delta_x$ e $\omega_y = \Omega_y\Delta_y$ em radianos. Neste caso, $u = \frac{1}{\Delta_x}$ e $v = \frac{1}{\Delta_y}$ equivalem à $\omega_x = \omega_y = 2\pi$. As freqüências u e v contínuas são substituídas por u/M e v/N discretas, $u = 0, 1, \ldots, M-1$ e $v = 0, 1, \ldots, N-1$. Estando a magnitude do espectro centrada na imagem $I(u,v)$, com $M \times N$ pixels, temos que $\frac{M}{2} = \frac{N}{2} = \pi$ (i.e. a imagem do espectro varia de $-\pi$ a π).

2 Exercícios

1. Demonstre os pares de transformadas contínuas apresentados no início da aula (A demonstração da Equação 82 é mais complicada, use somatórias de cossenos). A demonstração da Equação 84 pode ser feita facilmente a partir da fórmula da inversa. Lembre-se que $\exp^{j\theta} = \cos(\theta) + j\sin(\theta)$.

2. Se $\Delta_x = 1mm$ e $\Delta_y = 2mm$ e uma imagem possui $M \times N$ pixels com $M = 256$ e $N = 256$, então quais as freqüências u e v contínuas para $u = v = 128$ discretos?

1 Transformada de Fourier Discreta

Considere as imagens $\hat{I} = (D_I, I')$, com $N_1 \times M_1$ pixels, e $\hat{J} = (D_J, J')$, com $N_2 \times M_2$ pixels, e duas funções discretas $R_{MN}(x, y)$ e $R_{MN}(u, v)$, $M = M_1 + M_2 - 1$ e $N = N_1 + N_2 - 1$, tais que

$$R_{MN}(x,y) = \begin{cases} 1, & \text{se } x \in [0, M-1] \text{ e } y \in [0, N-1], \text{ e} \\ 0, & \text{no c.c.} \end{cases} \quad (99)$$

$$R_{MN}(u,v) = \begin{cases} 1, & \text{se } u \in [0, M-1] \text{ e } v \in [0, N-1], \text{ e} \\ 0, & \text{no c.c.} \end{cases} \quad (100)$$

Considere as extensões $I(x, y)$ de $I'(x, y)$ e $J(x, y)$ de $J'(x, y)$, onde zeros são acrescentados na horizontal e na vertical até $(M - 1, N - 1)$. Então, suas extensões periódicas $I_p(x, y)$ e $J_p(x, y)$, com períodos (M, N), e suas séries de Fourier discretas $\vec{I}_p(u, v)$ e $\vec{J}_p(u, v)$, devem ser tais que

$$I(x,y) = I_p(x,y)R_{MN}(x,y) = \begin{cases} I'(x,y), & \text{se } x \in [0, M_1-1] \text{ e } y \in [0, N_1-1], \text{ e} \\ 0, & \text{se } x \in [M_1, M-1] \text{ ou } y \in [N_1, N-1] \end{cases} \quad (101)$$

$$\vec{I}_p(u,v)R_{MN}(u,v) = \vec{I}(u,v), \quad \text{para } u \in [0, M-1] \text{ e } v \in [0, N-1]. \quad (102)$$

$$J(x,y) = J_p(x,y)R_{MN}(x,y) = \begin{cases} J'(x,y), & \text{se } x \in [0, M_2-1] \text{ e } y \in [0, N_2-1], \text{ e} \\ 0, & \text{se } x \in [M_2, M-1] \text{ ou } y \in [N_2, N-1] \end{cases} \quad (103)$$

$$\vec{J}_p(u,v)R_{MN}(u,v) = \vec{J}(u,v), \quad \text{para } u \in [0, M-1] \text{ e } v \in [0, N-1]. \quad (104)$$

onde $\vec{I}(u, v)$ e $\vec{J}(u, v)$ são as transformadas de Fourier discretas de $I(x, y)$ e $J(x, y)$.

Substituindo $\exp^{\frac{-j2\pi}{M}}$ por W_M e $\exp^{\frac{-j2\pi}{N}}$ por W_N temos

$$\vec{I}(u,v) = R_{MN}(u,v)\left[\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} I(x,y)W_M^{ux}W_N^{vy}\right] \quad (105)$$

$$I(x,y) = R_{MN}(x,y)\left[\frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} \vec{I}(u,v)W_M^{-ux}W_N^{-vy}\right] \quad (106)$$

1.1 Propriedades

1.1.1 Distributividade e escalamento

$$aI(x,y) + bJ(x,y) \leftrightarrow a\vec{I}(u,v) + b\vec{J}(u,v) \quad (107)$$

$$I(ax,by) \leftrightarrow \vec{I}\left(\frac{u}{a}, \frac{v}{b}\right) \quad (108)$$

Observe que a subamostragem $I(ax, by)$ é obtida para $a > 1$ e $b > 1$, e a superamostragem para $0 < a < 1$ e $0 < b < 1$. A primeira aproxima as repetições do espectro em freqüência (podendo ocasionar aliasing), enquanto a segunda afasta essas repetições.

1.1.2 Translação

A translação é redefinida como deslocamento circular de $I(x,y)$, ou translação da série $I_p(x,y)$. O mesmo sendo válido para o domínio da freqüência.

$$I_p(x+m,y+n)R_{MN}(x,y) \quad \leftrightarrow \quad W_M^{mu}W_N^{nv}\vec{I}(u,v) \tag{109}$$

$$W_M^{-mu}W_N^{-nv}I(x,y) \quad \leftrightarrow \quad \vec{I_p}(u+m,v+n)R_{MN}(u,v) \tag{110}$$

$$\tag{111}$$

1.1.3 Teorema da Convolução

A convolução discreta é redefinida como convolução circular (ou periódica),

$$I(x,y) * J(x,y) \quad = \quad R_{MN}(x,y)\left[\sum_{x'=0}^{M-1}\sum_{y'=0}^{N-1} I_p(x',y')J_p(x-x',x-y')\right] \tag{112}$$

$$\vec{I}(u,v) * \vec{J}(u,v) \quad = \quad R_{MN}(u,v)\left[\sum_{u'=0}^{M-1}\sum_{v'=0}^{N-1} \vec{I}_p(u',v')\vec{J}_p(u-u',v-v')\right]. \tag{113}$$

Observe que o resultado da convolução circular é essencialmente o mesmo da convolução discreta para $M \geq M_1 + M_2 - 1$ e $N \geq N_1 + N_2 - 1$. O teorema da convolução fica, portanto,

$$I(x,y) * J(x,y) \quad \leftrightarrow \quad \vec{I}(u,v)\vec{J}(u,v) \tag{114}$$

$$I(x,y)J(x,y) \quad \leftrightarrow \quad \frac{1}{(MN)^2}\vec{I}(u,v) * \vec{J}(u,v) \tag{115}$$

1.1.4 Teorema da Correlação

A correlação discreta é redefinida como convolução circular (ou periódica),

$$I(x,y) \odot J(x,y) \quad = \quad R_{MN}(x,y)\left[\sum_{x'=0}^{M-1}\sum_{y'=0}^{N-1} I_p(x',y')J_p(x+x',x+y')\right] \tag{116}$$

$$\vec{I}(u,v) \odot \vec{J}(u,v) \quad = \quad R_{MN}(u,v)\left[\sum_{u'=0}^{M-1}\sum_{v'=0}^{N-1} \vec{I}_p^*(u',v')\vec{J}_p(u+u',v+v')\right], \tag{117}$$

onde $\vec{I}_p^*(u',v')$ é o conjugado de $\vec{I}_p(u',v')$. Observe que o resultado da convolução circular é essencialmente o mesmo da convolução discreta para $M \geq M_1 + M_2 - 1$ e $N \geq N_1 + N_2 - 1$. O teorema da correlação fica, portanto,

$$I(x,y) \odot J(x,y) \quad \leftrightarrow \quad \vec{I}^*(u,v)\vec{J}(u,v) \tag{118}$$

$$\vec{I}(x,y)J(x,y) \quad \leftrightarrow \quad \frac{1}{(MN)^2}\vec{I}^*(u,v) \odot \vec{J}(u,v) \tag{119}$$

1.1.5 Rotação

Expressando $I(x,y)$ e $\vec{I}(u,v)$ em coordenadas polares $I(r,\theta)$, $x = r\cos(\theta)$, $y = r\sin(\theta)$, e $\vec{I}(r',\phi)$, $u = r'\cos(\phi)$ e $v = r'\sin(\phi)$, temos que

$$I(r,\theta+\alpha) \quad \leftrightarrow \quad \vec{I}(r',\phi+\alpha) \tag{120}$$

1.1.6 Separabilidade

A transformada $\vec{I}(u,v)$ de $I(x,y)$ pode ser separada em duas transformadas 1D, uma na horizontal e outra na vertical. O mesmo vale para a inversa.

$$\sum_{x=0}^{M-1}\left[\sum_{y=0}^{N-1} I(x,y)W_N^{vy}\right]W_M^{ux} = \sum_{x=0}^{M-1}\vec{I}(x,v)W_M^{ux} \tag{121}$$

$$\frac{1}{M}\sum_{u=0}^{M-1}\left[\frac{1}{N}\sum_{v=0}^{N-1}\vec{I_p}(u,v)W_N^{-vy}\right]W_M^{-ux} = \frac{1}{M}\sum_{u=0}^{M-1} I_p(u,y)W_M^{-ux} \tag{122}$$

Note que para cada valor de $\vec{I}(x,v)$, $x = 0,1,\ldots,M-1$, $v = 0,1,\ldots,N-1$, na Equação 121 temos que calcular N multiplicações de $I(x,y)$ por W_N^{vy}, $y = 0,1,\ldots,N-1$, e que para cada valor de $\vec{I}(u,v)$, $u = 0,1,\ldots,M-1$, $v = 0,1,\ldots,N-1$, temos que calcular M multiplicações de $\vec{I}(x,v)$ por W_M^{ux}, $x = 0,1,\ldots,M-1$. Portanto, a separabilidade já permite que a complexidade original seja reduzida de $O(M^2N^2)$ para $O(MN^2+NM^2)$. A transformada rápida de Fourier (FFT-*Fast Fourier Transform*) explora esta propriedade e a periodicidade das funções W_M^u e W_N^v para reduzir a complexidade para $O(MN\log_2^N + NM\log_2^M)$.

2 Exercícios

Demonstre todas as propriedades vistas nesta aula.

1 Algoritmo da Transformada Rápida de Fourier

Pela propriedade de separabilidade, o algoritmo 1D da transformada rápida de Fourier pode ser usado na horizontal e depois na vertical. Considere a somatória da transformada de Fourier 1D discreta

$$\vec{I}(u) = \sum_{x=0}^{M-1} I(x) W_M^{ux}, \quad \text{u=0,1,...,M-1} \tag{123}$$

Seja $M = 2^m$, e quando não for o caso, complete com zeros a função $I(x)$ para que M seja sempre uma potência de 2. Podemos substituir $M = 2M'$ e reescrever a somatória acima como

$$\vec{I}(u) = \sum_{x=0}^{2M'-1} I(x) W_{2M'}^{ux}, \quad \text{u=0,1,...,2M'-1} \tag{124}$$

Esta somatória pode ainda ser dividida nas somatórias dos termos pares e ímpares, mas o resultado só será válido para as $\frac{M}{2}$ primeiras amostras, $u = 0, 1, \ldots, M' - 1$, devido à subamostragem.

$$\vec{I}(u) = \sum_{x=0}^{M'-1} I(2x) W_{M'}^{ux} + \sum_{x=0}^{M'-1} I(2x+1) W_{M'}^{ux} W_{2M'}^{u}, \quad \text{u=0,1,...,M'-1.} \tag{125}$$

$$\vec{I}(u) = \vec{I}_{par}(u) + \vec{I}_{impar}(u) W_{2M'}^{u}, \quad \text{u=0,1,...,M'-1.} \tag{126}$$

Para calcular a outra metade $\vec{I}(u+M')$, $u = 0, 1, \ldots, M'-1$, usamos o fato que $W_{M'}^{u+M'} = W_{M'}^{u}$ e $W_{2M'}^{u+M'} = -W_{2M'}^{u}$ são periódicas.

$$\vec{I}(u+M') = \sum_{x=0}^{M'-1} I(2x) W_{M'}^{(u+M')x} + \sum_{x=0}^{M'-1} I(2x+1) W_{M'}^{(u+M')x} W_{2M'}^{(u+M')}, \quad \text{u=0,1,...,M'-1} \tag{127}$$

$$\vec{I}(u+M') = \vec{I}_{par}(u) - \vec{I}_{impar}(u) W_{2M'}^{u}, \quad \text{u=0,1,...,M'-1} \tag{128}$$

Portanto, uma transformada de Fourier de M amostras pode ser obtida dividindo-se a expressão em duas partes e calculando-se duas transformadas de $\frac{M}{2}$ amostras, $\vec{I}_{par}(u)$ e $\vec{I}_{impar}(u)$, as quais devem ser combinadas conforme as equações acima. Estamos reduzindo M^2 multiplicações para $2\frac{M^2}{4} = \frac{M^2}{2}$ multiplicações. Esta estratégia é repetida recursivamente até $M' = 1$ (final da recursão). O algoritmo terá complexidade $M \log_2^M$.

Exemplo: Suponha uma seqüência $I(x) =< I(0), I(1), \ldots, I(7) >$ com $M = 8$ amostras. A primeira divisão separa esta seqüência nas amostras pares $I_0(x) =< I(0), I(2), I(4), I(6) >$ e nas ímpares $I_1(x) =< I(1), I(3), I(5), I(7) >$, $x = 0, 1, 2, 3$, cujas transformadas $\vec{I}_0(u)$ e $\vec{I}_1(u)$, $u = 0, 1, 2, 3$, são combinadas da seguinte forma.

$$\vec{I}(u) = \vec{I}_0(u) + \vec{I}_1(u) W_8^{u}, \quad \text{u=0,1,2,3.} \tag{129}$$

$$\vec{I}(u+4) = \vec{I}_0(u) - \vec{I}_1(u) W_8^{u}, \quad \text{u=0,1,2,3.} \tag{130}$$

A segunda divisão separa $I_0(x)$ em pares $I_{00}(x) =< I(0), I(4) >$ e ímpares $I_{01}(x) =< I(2), I(6) >$, $x = 0, 1$, e $I_1(x)$ em pares $I_{10}(x) =< I(1), I(5) >$ e ímpares $I_{11}(x) =< I(3), I(7) >$, $x = 0, 1$, cujas transformadas $\vec{I}_{00}(u)$, $\vec{I}_{01}(u)$, $\vec{I}_{10}(u)$ e $\vec{I}_{11}(u)$, $u = 0, 1$, são combinadas da seguinte forma.

$$\vec{I}_0(u) = \vec{I}_{00}(u) + \vec{I}_{01}(u)W_4^u, \quad \text{u=0,1.} \tag{131}$$
$$\vec{I}_0(u+2) = \vec{I}_{00}(u) - \vec{I}_{01}(u)W_4^u, \quad \text{u=0,1.} \tag{132}$$
$$\vec{I}_1(u+4) = \vec{I}_{10}(u) + \vec{I}_{11}(u)W_4^u, \quad \text{u=0,1.} \tag{133}$$
$$\vec{I}_1(u+6) = \vec{I}_{10}(u) - \vec{I}_{11}(u)W_4^u, \quad \text{u=0,1.} \tag{134}$$

A terceira e última divisão separa $I_{00}(x)$ em par $I_{000}(x) =< I(0) >$ e ímpar $I_{001}(x) =< I(4) >$, $x = 0$; $I_{01}(x)$ em par $I_{010}(x) =< I(2) >$ e ímpar $I_{011}(x) =< I(6) >$, $x = 0$; $I_{10}(x)$ em par $I_{100}(x) =< I(1) >$ e ímpar $I_{101}(x) =< I(5) >$, $x = 0$; e $I_{11}(x)$ em par $I_{110}(x) =< I(3) >$ e ímpar $I_{111}(x) =< I(7) >$, $x = 0$; cujas transformadas $\vec{I}_{000}(u) = I(0)$, $\vec{I}_{001}(u) = I(4)$, $\vec{I}_{010}(u) = I(2)$, $\vec{I}_{011}(u) = I(6)$, $\vec{I}_{100}(u) = I(1)$, $\vec{I}_{101}(u) = I(5)$, $\vec{I}_{110}(u) = I(3)$, $\vec{I}_{111}(u) = I(7)$, $u = 0$, são combinadas da seguinte forma.

$$\vec{I}_{00}(u) = I(0) + I(4)W_2^u, \quad \text{u=0.} \tag{135}$$
$$\vec{I}_{00}(u+1) = I(0) - I(4)W_2^u, \quad \text{u=0.} \tag{136}$$
$$\vec{I}_{01}(u+2) = I(2) + I(6)W_2^u, \quad \text{u=0.} \tag{137}$$
$$\vec{I}_{01}(u+3) = I(2) - I(6)W_2^u, \quad \text{u=0.} \tag{138}$$
$$\vec{I}_{10}(u+4) = I(1) + I(5)W_2^u, \quad \text{u=0.} \tag{139}$$
$$\vec{I}_{10}(u+5) = I(1) - I(5)W_2^u, \quad \text{u=0.} \tag{140}$$
$$\vec{I}_{11}(u+6) = I(3) + I(7)W_2^u, \quad \text{u=0.} \tag{141}$$
$$\vec{I}_{11}(u+7) = I(3) - I(7)W_2^u, \quad \text{u=0.} \tag{142}$$

Note que o valor de x na seqüência original para identificar a amostra $I(x) = \vec{I}_{b_1b_2b_3}(u)$, $u = 0$, pode ser obtido revertendo a ordem do índice binário $b_1b_2b_3$ para $b_3b_2b_1$ e convertendo o resultado para decimal. Isto é, índice 000 equive à $x = 0$, índice 001 equive à $x = 4$, índice 010 equive à $x = 2$, índice 011 equive à $x = 6$, índice 100 equive à $x = 1$, índice 101 equive à $x = 5$, índice 110 equive à $x = 3$, e índice 111 equive à $x = 7$.

Portanto, o algoritmo deve aplicar a ordem reversa dos bits para reordenar a seqüência de amostras $< I(0), I(1), I(2), I(3), I(4), I(5), I(6), I(7) >$ do sinal original em $< I(0), I(4), I(2), I(6), I(1), I(5), I(3), I(7) >$, e depois combinar duas as duas conforme às equações acima, seguindo a ordem da volta da recursão.

2 Exercícios

1. Repita o mesmo raciocício para a transformada inversa e implemente o algoritmo para calcular a FFT 1D direta e inversa.

2. Teste seu algoritmo na transformada do cosseno.

3. Implemente as transformadas FFT 2D direta e inversa em função do algoritmo para os casos 1D.

1 Filtragem na Freqüência

Pelo teorema da convolução, se $J(x,y)$ é a função de transferência de um filtro linear cujo resultado da filtragem é $I(x,y) * J(x,y)$, então esta operação no domínio da freqüência equivale ao produto das transformadas $\vec{I}(u,v)\vec{J}(u,v)$. Isto significa que, dependendo do tamanho da máscara de convolução, pode ser mais vantagem calcular a FFT de I e de J, multiplicar os espectros de freqüência, e depois calcular a FFT inversa do resultado.

Outra forma de explorar o teorema da convolução é o projeto de filtros no domínio da freqüência. Sabemos que regiões de borda e outras transições abruptas de cinza correspondem a componentes de alta freqüência, enquanto as baixas freqüências representam regiões mais homogêneas na imagem original. Neste contexto, filtros no domínio da freqüência podem ser de quatro tipos: passa-baixas, rejeita-faixa, passa-faixa, e passa-altas freqüências. Os extremos variam da suavização da imagem ao realce de bordas.

Vamos estudar o caso $J(u,v)$ real. Isto é, filtros que não modificam a fase da imagem original (*zero-phase-shift filters*).

1.1 Filtragem ideal

No caso ideal temos como filtros passa-baixas e passa-altas, respectivamente:

$$L(u,v) = \left\{ \begin{array}{ll} 1, & \text{se } D(u,v) \leq D_l \\ 0, & \text{no c.c.} \end{array} \right\} \tag{143}$$

$$H(u,v) = \left\{ \begin{array}{ll} 1, & \text{se } D(u,v) \geq D_h \\ 0, & \text{no c.c.} \end{array} \right\} \tag{144}$$

onde $D_l > 0$ e $D_h > 0$ definem as freqüências de corte, e $D(u,v) = (u^2 + v^2)^{1/2}$. Note que para $0 < D_l < D_h$, $L(u,v) + H(u,v)$ é um filtro rejeita-faixa $[D_l, D_h]$, e para $0 < D_h < D_l$, $L(u,v)H(u,v)$ é um filtro passa-faixa $[D_h, D_l]$.

Muito embora esses filtros possam ser usados para simulação no computador, eles não podem ser implementados com componentes eletrônicos. A variação abrupta na freqüência também gera um efeito *ringing* (falsas bordas) no espaço. Uma alternativa é o filtro de *Butterworth*, que possui uma variação mais suave em torno das freqüências de corte.

1.2 Filtros de Butterworth

Os filtros de Butterworth de ordem $n > 0$, passa-baixas e passa-altas, são:

$$L(u,v) = \frac{1}{1 + 0.414\left[D(u,v)/D_l\right]^{2n}} \tag{145}$$

$$H(u,v) = \frac{1}{1 + 0.414\left[D_h/D(u,v)\right]^{2n}} \tag{146}$$

Note que para $n = 1$, $L(u,v)$ e $H(u,v)$ caem para $\sqrt{2}/2$ de seus valores máximos em $D(u,v) = D_l$ e $D(u,v) = D_h$, respectivamente.

1.3 Geração de máscaras espaciais a partir de especificações na freqüência

Considere $G(u,v)$ o espectro em freqüência de um filtro linear como uma função real e simétrica. Sua inversa será a máscara espacial $g(x,y)$ real e simétrica. Para facilitar, suponha que o número de amostras é o mesmo em ambas direções, i.e. $M = N$. É muito conveniente projetar $G(u,v)$ na freqüência, mas implementá-lo por convolução espacial usando uma máscara $g'(x,y)$ com poucos coeficientes. A máscara $g'(x,y)$ é uma restrição de $g(x,y)$, tal que $g'(x,y) = g(x,y)$ para $-N/2 < -n/2 \leq x,y \leq n/2 < N/2$, e $g'(x,y) = 0$ para $-N/2 < -n/2 > x,y > n/2 < N/2$. Esta restrição faz com que haja um erro e de aproximação entre o filtro $G(u,v)$ projetado e o implementado $G'(u,v)$.

$$e^2 = \sum_{u=-N/2}^{N/2} \sum_{v=-N/2}^{N/2} |G'(u,v) - G(u,v)|^2 \tag{147}$$

O objetivo desta seção é mostrar como obter os coeficientes de $g'(x,y)$ com erro quadrático e^2 mínimo.

O espectro $G'(u,v)$ com $(N+1)^2$ elementos pode ser representado em um vetor coluna $\mathbf{G'}$, cujos valores $G(i)$ são gerados variando $v = -N/2, \ldots, N/2$, $u = -N/2, \ldots, N/2$, $i = (u+N/2)(N+1)+(v+N/2)$, de forma que uma coluna de $G'(u,v)$ é copiada após a outra para $\mathbf{G'}$. O mesmo procedimento pode ser aplicado ao espectro $G(u,v)$ para gerar o vetor coluna $\mathbf{G}$ e à máscara $g'(x,y)$ para gerar o vetor coluna $\mathbf{g'}$ com $(n+1)^2$ elementos, cujos valores $g'(k)$ são gerados variando $y = -n/2, \ldots, n/2$, $x = -n/2, \ldots, n/2$, $k = (x+n/2)(n+1)+(y+n/2)$. Assim, a transformada de Fourier de $\mathbf{g'}$ pode ser expressa na forma matricial

$$\mathbf{G'} = \mathbf{w} \cdot \mathbf{g'}, \tag{148}$$

onde $\mathbf{w}$ é uma matriz de $(N+1)^2$ linhas e $(n+1)^2$ colunas, cujos valores $w(k,i) = \frac{1}{N+1} \exp^{\frac{-j2\pi}{N+1}(ux+vy)}$ da coluna k e linha i são gerados linha por linha, variando $y = -n/2, \ldots, n/2$, $x = -n/2, \ldots, n/2$, $v = -N/2, \ldots, N/2$, $u = -N/2, \ldots, N/2$, e calculando os índices $k = (x+n/2)(n+1)+(y+n/2)$ e $i = (u+N/2)(N+1)+(v+N/2)$. A minimização de e^2 implica que

$$e^2 = (\mathbf{G'} - \mathbf{G})(\mathbf{G'} - \mathbf{G}) = \|\mathbf{w}\mathbf{g'} - \mathbf{G}\|^2 \tag{149}$$

$$\frac{\delta e^2}{\delta \mathbf{g'}} = 2\mathbf{w}^* (\mathbf{w}\mathbf{g'} - \mathbf{G}) = 0 \tag{150}$$

$$\mathbf{g'} = (\mathbf{w}^*\mathbf{w})^{-1}\mathbf{w}^*\mathbf{G} \tag{151}$$

onde $\mathbf{w}^*$ é o conjugado transposto (matriz de $(n+1)^2$ linhas e $(N+1)^2$ colunas) de $\mathbf{w}$ e $(\mathbf{w}^*\mathbf{w})^{-1}\mathbf{w}^*$ é chamada a inversa generalizada de *Moore-Penrose*. Note que, basta expressar o filtro projetado $G(u,v)$ na forma vetorial $\mathbf{G}$, calcular $\mathbf{g'}$, e depois colocá-lo na forma $g'(x,y)$.

2 Exercícios

1. Projete filtros passa-faixa e rejeita-faixa usando os filtros passa-baixas e passa-altas de Butterworth.

2. Implemente uma função para calcular a filtragem de $I(x, y)$ pela máscara $J(x, y)$ no domínio da freqüência.

3. Implemente uma função para calcular uma máscara $g'(x, y)$ a partir de uma especificação $G(u, v)$ em freqüência.

4. Calcule $g'(x, y)$ para os filtros passa-baixas e passa-altas de Butterworth e compare os resultados da filtragem espacial com a filtragem em freqüência para uma imagem de entrada qualquer.

5. Considere as imagens das máscaras de Sobel e do filtro Gaussiano 3×3. Insira 253 zeros na horizontal e na vertical, e calcule a FFT de todas elas para visualizar seus espectros de magnitude com resolução 256×256 amostras. Que tipos de filtros em freqüência essas máscaras representam?

1 Restauração de Imagens

Técnicas de restauração visam recuperar a imagem original a partir de uma imagem degradada, usando o conhecimento sobre a natureza da degradação— a qual pode ser determinística ou aleatória.

As degradações mais comuns ocorrem durante a aquisição da imagem. Alguns exemplos são:

1. Uma imagem de sensoriamento remoto com degradação gerada pela turbulência atmosférica, causada por variações de temperatura que desviam os raios de luz.

2. Uma imagem de microscopia ótica adquirida fora de foco.

3. Uma imagem fotográfica "borrada", devido ao movimento relativo entre a câmera e o objeto.

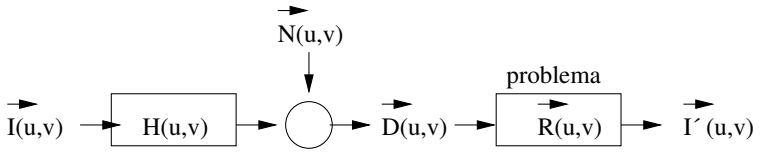
A estratégia básica da restauração de imagens é modelar o processo de degradação e aplicar o processo inverso. Vamos fazer isto usando nossos conhecimentos sobre sistemas lineares.

2 Modelos de degradação

Um modelo simples, mas bastante eficaz, assume que a imagem foi degradada por um filtro linear e invariante ao deslocamento, seguido de ruído aditivo.

Sejam $H(u, v)$ uma função real e simétrica de transferência do filtro (também chamada de *Point Spread Function* em analogia à resposta a um impulso de luz), $\vec{N}(u, v)$ o espectro em freqüência de um ruído aditivo, $\vec{I}(u, v)$ o espectro da imagem original, e $\vec{D}(u, v)$ o espectro da imagem degradada (ver Figura 1).

$$\vec{D}(u, v) = \vec{I}(u, v)H(u, v) + \vec{N}(u, v). \tag{152}$$



O problema, portanto, consiste em encontrar o filtro $\vec{R}(u,v)$ que aproxima $\vec{I'}(u,v)$ do espectro $\vec{I}(u,v)$ da imagem original.

$$\vec{I'}(u,v) \;=\; \vec{D}(u,v)\vec{R}(u,v). \tag{153}$$

Se a restauração for perfeita, então $\vec{I'}(u,v) = \vec{I}(u,v)$, significando que

$$\vec{R}(u,v) \;=\; \frac{1}{H(u,v) + \frac{\vec{N}(u,v)}{\vec{I}(u,v)}}. \tag{154}$$

Este modelo pode ainda ser simplificado usando as técnicas abaixo.

2.1 Filtragem inversa

Assume-se que $\vec{N}(u,v) = 0$, e portanto, $\vec{R}(u,v) = R(u,v) = \frac{1}{H(u,v)}$ é uma função real e simétrica. Neste caso, uma atenção especial deve ser dada quando $H(u,v) = 0$ para algumas freqüências (u,v). Como normalmente, $H(u,v)$ tem características de passa-baixas, é comum assumir que

$$R(u,v) \;=\; \begin{cases} \frac{1}{H(u,v)}, & u^2 + v^2 \leq w^2 \\ 1, & \text{no c.c.} \end{cases} \tag{155}$$

para algum raio w.

2.2 Filtragem de Wiener

Assume-se que $I(u,v), D(u,v)$, e $N(u,v)$ são processos estocásticos (campos aleatórios) estacionários e que as densidades espectrais, $\mathcal{S}_I(u,v)$ e $\mathcal{S}_N(u,v)$, da imagem original e do ruído são conhecidas. O Filtro de Wiener (filtro de mínimos quadrados) minimiza o erro quadrático médio $E[(\vec{I'}(u,v) - \vec{I}(u,v))^2]$ gerando

$$R(u,v) \;=\; \frac{|H(u,v)|^2}{|H(u,v)|^2 + \frac{\mathcal{S}_N(u,v)}{\mathcal{S}_I(u,v)}} \frac{1}{H(u,v)}. \tag{156}$$

A relação ruído-sinal $\frac{\mathcal{S}_N(u,v)}{\mathcal{S}_I(u,v)}$ é normalmente substituída por uma constante (e.g. 10%), quando desconhecemos a natureza estatística do problema. Note que se $\frac{\mathcal{S}_N(u,v)}{\mathcal{S}_I(u,v)} = 0$, a filtragem de Wiener fica igual à filtragem inversa, e que a aproximação acima usada para tratar casos onde $H(u,v) = 0$ também se aplica aqui. A questão agora é como encontrar $H(u,v)$.

2.3 Modelando $H(u, v)$

Voltando aos exemplos mais comuns de degradação, $H(u, v)$ pode ser modelada da seguinte forma:

1. Turbulência atmosférica

   Neste caso, $H(u, v)$ pode ter a forma de uma distribuição de impulsos aleatórios, caso o tempo de exposição para aquisição da imagem seja curto, ou pode ter a forma de uma Gaussiana para tempos de exposição prolongados. O primeiro caso é mais complexo e o segundo resulta em

   $$H(u, v) = \exp\left[-c(u^2 + v^2)^d\right], \tag{157}$$

   onde c e d são constantes, porém c depende do tipo de turbulência e d pode ser encontrada experimentalmente (e.g. $d = 5/6$ ou 1).

2. Imagem fora de foco

   $$\begin{aligned} H(u, v) &= \frac{J_1(r + w)}{r + w} \\ J_1(r + w) &= \left(\frac{r + w}{2}\right)\left[1 + \ldots + \frac{(-1)^n}{n!(n + 1)!}\left(\frac{r + w}{2}\right)^{2n} + \ldots\right], \end{aligned} \tag{158}$$

   onde $J_1(r + w)$ é uma função de Bessel de primeira ordem, deslocada de $w > 0$ para ter valor máximo na origem, e $r^2 = u^2 + v^2$.

3. Imagem "borrada" pelo movimento

   O movimento relativo entre a câmera e o objeto pode ser modelado no domínio espacial como

   $$D(x, y) = \int_{-T/2}^{T/2} I(x - x'(t), y - y'(t))dt, \tag{159}$$

   onde T é o tempo de exposição da câmera, $x'(t)$ e $y'(t)$ são os deslocamentos dos pontos da cena ao longo de x e y em função do tempo t. No domínio da freqüência temos

   $$\vec{D}(u, v) = \vec{I}(u, v)\int_{-T/2}^{T/2} \exp[-j2\pi(ux'(t) + vy'(t))]dt. \tag{160}$$

   Assumindo, por exemplo, movimento uniforme do objeto na direção x com velocidade constante V, $x'(t) = Vt$ e $y'(t) = 0$,

   $$\vec{D}(u, v) = \vec{I}(u, v)\frac{\sin(\pi uVT)}{\pi uV}. \tag{161}$$

   Isto é, $H(u, v) = T\frac{\sin(\pi uVT)}{\pi uVT}$.

1 Introdução à morfologia matemática

A morfologia matemática é a parte do processamento de imagem não-linear que tem por objetivo extrair características da imagem associadas à geometria dos objetos. A morfologia matemática foi desenvolvida inicialmente por Georges Matheron e Jean Serra na década de 60, para imagens binárias utilizando a teoria de conjuntos. Posteriormente, ela foi estendida para imagens em tons cinza (funções) utilizando a teoria de reticulados, onde uma imagem é vista como a superfície de um relevo.

Nosso objetivo neste curso é apresentar apenas uma introdução à morfologia matemática.

1.1 Elemento estruturante

Uma transformação morfológica consiste essencialmente da comparação da imagem com outra menor, cuja geometria é conhecida, denominada elemento estruturante.

Um elemento estruturante planar é um conjunto de coordenadas de pixel. Por exemplo, o elemento cruz é definido por $E = \{(0,0),(-1,0),(1,0),(0,-1),(0,1)\}$. Uma transformação morfológica requer uma operação não-linear entre a imagem e o elemento estruturante, o qual desliza sobre a imagem de forma similar à convolução discreta. Neste sentido, o elemento estruturante planar define uma relação de adjacência do tipo $(p,q) \in A$ se $q - p \in E$.

Um elemento estruturante não-planar é um par (E,V) que consiste de um conjunto de coordenadas de pixel E e um conjunto de valores V associados a cada coordenada, assim como uma imagem. Por exemplo, $V = \{2,1,1,1,1\}$ para o caso do elemento cruz. Este tipo de elemento é usado apenas em operações com imagens em tons de cinza. Neste caso, o elemento estruturante pode ser visto como uma máscara de convolução, muito embora a operação seja outra. No caso particular, onde todos valores em V são zero, o elemento estruturante se torna planar.

1.2 Dilatação e Erosão

A dilatação e a erosão são as duas transformações morfológicas básicas, as quais são combinadas para gerar várias outras. Elas envolvem as seguintes operações com conjuntos de coordenadas de pixel.

- Translação

  Um conjunto A transladado de $t = (x_t, y_t)$ é um conjunto $A^t = \{p + t : \forall p \in A\}$.

- Reflexão

  Um conjunto A refletido é um conjunto $A^r = \{q = -p : \forall p \in A\}$.

1.2.1 Dilatação

Considere $\hat{I} = (D_I, I)$ uma imagem binária e o conjunto $U_I \subset D_I$ formato pelos pixels $p \in D_I$, tais que $I(p) = 1$. A dilatação $\hat{I} \oplus E$ de $\hat{I}$ por um elemento estruturante planar E resulta em

uma imagem binária $\hat{J} = (D_J, J)$, onde

$$U_J \;=\; \{t : (E^r)^t \cap U_I \neq \emptyset\}. \tag{162}$$

Isto é, U_J é formado por todos os deslocamentos t tais que o elemento estruturante refletido e transladado superpõe os pixels com valor 1 na imagem em pelo menos um pixel. Note que $J(p) = 1$ se $p \in U_J \subset D_J$ e zero no caso contrário.

Como não se trata de uma convolução, temos apenas uma analogia, alguns autores não refletem o elemento estruturante.
Exemplo:

Sejam $U_I = \{(1,1),(2,1)\}$ e $E = \{(0,0),(0,1)\}$, então $E^r = \{(0,0),(0,-1)\}$ e $(E^r)^t = \{(x_t, y_t),(x_t, y_t - 1)\}$. Se $t = (0,0)$, $(E^r)^t \cap U_I = \emptyset$; se $t = (1,1)$, $(E^r)^t \cap U_I = \{(1,1)\}$; etc. Ao final teremos, $U_J = \{(1,1),(2,1),(1,2),(2,2)\}$.

Observe que os objetos representados por pixels com valor 1 na imagem ficam mais "gordos" e que pequenos buracos podem ser fechados com a dilatação.

Se $\hat{I} = (D_I, I)$ for uma imagem em tons de cinza, a dilatação $\hat{I} \oplus (E, V)$ de $\hat{I}$ por um elemento estruturante não-planar (E, V) resulta em uma imagem em tons de cinza $\hat{J} = (D_J, J)$ onde

$$J(p) \;=\; \max_{\forall t \in E}\{I(p-t) + V(t)\}, \tag{163}$$

para todo $p \in D_J$ e $p - t \in D_I$. Neste caso, a imagem fica mais clara e somem pequenas regiões escuras.

1.2.2 Erosão

A erosão $\hat{I} \ominus E$ de uma imagem binária $\hat{I}$ por um elemento estruturante planar E resulta em uma imagem binária $\hat{J} = (D_J, J)$, onde

$$U_J \;=\; \{t : (E^r)^t \subseteq U_I\}. \tag{164}$$

Isto é, U_J é formado por todos os deslocamentos t tais que o elemento estruturante refletido e transladado está inteiramente contido no conjunto dos pixels com valor 1 da imagem de entrada.
Exemplo:

Sejam $U_I = \{(1,1),(2,1),(1,2),(2,2)\}$ e $E = \{(0,0),(0,1)\}$, então $E^r = \{(0,0),(0,-1)\}$ e $(E^r)^t = \{(x_t, y_t),(x_t, y_t - 1)\}$. Se $t = (0,0)$, $(E^r)^t = \{(0,0),(0,-1)\}$; se $t = (1,1)$, $(E^r)^t = \{(1,1),(1,0)\}$; ...; se $t = (1,2)$, $(E^r)^t = \{(1,2),(1,1)\}$; etc. Ao final teremos $U_J = \{(1,2),(2,2)\}$.

Observe que os objetos representados por pixels com valor 1 na imagem ficam mais "magros" e que pequenos componentes com valor 1 somem com a erosão.

Se $\hat{I} = (D_I, I)$ for uma imagem em tons de cinza, a erosão $\hat{I} \ominus (E, V)$ de $\hat{I}$ por um elemento estruturante não-planar (E, V) resulta em uma imagem em tons de cinza $\hat{J} = (D_J, J)$ onde

$$J(p) \;=\; \min_{\forall t \in E}\{I(p-t) - V(t)\}, \tag{165}$$

para todo $p \in D_J$ e $p - t \in D_I$. Neste caso, a imagem fica mais escura e somem pequenas regiões claras.

1.3 Algoritmo genérico para dilatação/erosão

Observe que a dilatação e a erosão de imagens cinza por elemento estruturante não-planar incluem os casos de imagens binárias e elemento planar. Para facilitar, podemos também restringir o resultado ao domínio da imagem de entrada— i.e. $D_J = D_I$.

Algoritmo para dilatação:
Entrada: Imagem cinza $\hat{I} = (D_I, I)$ e elemento estruturante não-planar (E, V).
Saída: Imagem cinza $\hat{J} = (D_I, J) = \hat{I} \oplus (E, V)$.

1. Calcule a reflexão E^r mapeando todo $(x, y) \in E$ para $(-x, -y) \in E^r$ e $V(-x, -y) \leftarrow V(x, y)$.
2. Calcule a relação de adjacência A, tal que $q \in A(p)$ se $q - p \in E^r$.
3. Para todo pixel $p \in D_I$, faça
4. $\quad J(p) \leftarrow -\infty$.
5. $\quad$ Para todo pixel $q \in A(p)$, tal que $q \in D_I$, faça
6. $\quad\quad$ Se $(I(q) + V(q - p)) > J(p)$, então $J(p) \leftarrow I(q) + V(q - p)$.

A erosão pode ser calculada de forma similar. Também é comum restringir os valores de J entre 0 e um valor máximo $2^b - 1$.

2 Exercícios

1. Calcule as imagens resultantes da dilatação e da erosão de $\hat{I} = (D_I, I)$, $D_I = \{(0, 0), (0, 1), (1, 1), (1, 2)\}$ e $I = \{0, 2, 2, 1\}$, pelo elemento $E = \{(-1, 0), (0, 0), (1, 0)\}$ cujos valores $V = \{1, 2, 1\}$.
2. Mostre que se $V = \{0, 0, 0\}$ e $I = \{0, 1, 1, 0\}$ na imagem anterior, a dilatação pelo algoritmo acima apresentaria o mesmo resultado da dilatação pela teoria de conjuntos.
3. Implemente o algoritmo acima para dilatação e para erosão.

1 Filtros morfológicos

As operações de dilatação e erosão podem ser combinadas para gerar vários filtros morfológicos, que podem ser usados para remover ruído, realçar bordas, e suavizar a imagem para a segmentação. Esses filtros se caracterizam por resultarem de uma operação não-linear entre uma imagem e um elemento estruturante, e pelas seguintes propriedades:

- monoticidade - O filtro Ψ preserva a relação de ordem entre as imagens cinza $\hat{I} = (D_I, I)$ e $\hat{J} = (D_J, J)$, onde $D_I = D_J$.

$$\hat{I} \leq \hat{J} \quad \Rightarrow \quad \Psi(\hat{I}) \leq \Psi(\hat{J}), \tag{166}$$

  onde $\hat{I} \leq \hat{J}$ significa que $I(p) \leq J(p)$, para todo pixel $p \in D_I$. No caso de imagens binárias, poderíamos também escrever $U_I \subseteq U_J \Rightarrow \Psi(U_I) \subseteq \Psi(U_J)$.

- idempotência - O filtro Ψ aplicado duas vezes à imagem gera o mesmo resultado de quando é aplicado uma única vez.

$$\Psi(\Psi(\hat{I})) \quad = \quad \Psi(\hat{I}). \tag{167}$$

1.1 Abertura e Fechamento

A dilatação de uma imagem binária por um elemento planar E fecha os buracos, mas "engorda" a figura. O fechamento morfológico por E corrige esta distorção. O fechamento "suaviza" a fronteira dos objetos fechando indentações, fecha buracos, e une components próximas. A abertura é a operação dual (i.e. a abertura equivale ao complemento do fechamento do complemento da imagem), que também "suaviza" a fronteira, eliminando protusões, remove componentes menores que o elemento estruturante, e quebra ligações finas entre componentes conexos. Observações similares se aplicam para imagens cinza (relevos) e elementos não-planares.

A abertura $\hat{I} \circ (E, V)$ e o fechamento $\hat{I} \bullet (E, V)$ de uma imagem $\hat{I}$ por um elemento (E, V) são definidas por:

$$\hat{I} \circ (E, V) \quad = \quad (\hat{I} \ominus (E, V)) \oplus (E, V) \tag{168}$$

$$\hat{I} \bullet (E, V) \quad = \quad (\hat{I} \oplus (E, V)) \ominus (E, V) \tag{169}$$

1.2 Filtros alternados seqüências

Filtros alternados seqüênciais resultam da aplicação alternada de aberturas (O de *opening*) e fechamentos (C de *closing*) morfológicos. Por exemplo,

$$CO(\hat{I}, (E, V)) \quad = \quad (\hat{I} \bullet (E, V)) \circ (E, V) \tag{170}$$

$$OC(\hat{I}, (E, V)) \quad = \quad (\hat{I} \circ (E, V)) \bullet (E, V) \tag{171}$$

$$COC(\hat{I}, (E, V)) \quad = \quad ((\hat{I} \bullet (E, V)) \circ (E, V)) \bullet (E, V) \tag{172}$$

$$OCO(\hat{I}, (E, V)) \quad = \quad ((\hat{I} \circ (E, V)) \bullet (E, V)) \circ (E, V) \tag{173}$$

Esses filtros também podem ser aplicados sucessivas vezes aumentando o tamanho do elemento estruturante a cada passo.

2 Gradiente morfológico

Como a erosão é uma operação anti-extensiva (a função resultante é menor que a original) e a dilatação é extensiva, bordas da imagem podem ser realçadas calculando-se o resíduo dessas operações.

$$\hat{G}_1 = \hat{I} - (\hat{I} \ominus (E, V)) \quad (174)$$
$$\hat{G}_2 = (\hat{I} \oplus (E, V)) - \hat{I} \quad (175)$$
$$\hat{G}_3 = (\hat{I} \oplus (E, V)) - (\hat{I} \ominus (E, V)) \quad (176)$$

As imagens resultantes são denominadas gradientes morfológicos e podem ser usadas na segmentação. Note que este tipo de gradiente é não-direcional.

3 Chapéu mexicano

Outra forma de realçar bordas (WTH de *white top-hat*) ou objetos escuros (BTH de *black top-hat*) na imagem é calculando o resíduo com relação à abertura e ao fechamento.

$$WTH(\hat{I}, (E, V)) = \hat{I} - (\hat{I} \circ (E, V)) \quad (177)$$
$$BTH(\hat{I}, (E, V)) = (\hat{I} \bullet (E, V)) - \hat{I} \quad (178)$$

O volume de resíduo para elementos estruturantes de diferentes tamanhos pode ser utilizado para descrever o conteúdo granulométrico da imagem (análise de textura por granulometria).

4 Transformada tudo-ou-nada

A transformada tudo-ou-nada (HMT de *hit-or-miss transform*) é normalmente usada para encontrar configurações específicas de pixels em imagens binárias. Não existe extensão da HMT para o caso de imagens cinza. Seja U_I o conjunto dos pixels com valor 1 em uma imagem binária $\hat{I} = (D_I, I)$. Sejam E_0 e E_1 dois elementos estruturantes planares com mesma origem. A idéia é que E_0^t deve indicar a configuração desejada dos pixels com valor zero na imagem para a translação t e E_1^t deve indicar a configuração desejada dos pixels com valor 1 na imagem para a translação t. A transformada tudo-ou-nada de $\hat{I}$ com E_0 e E_1 gera uma imagem binária $\hat{J} = (D_I, J)$ tal que U_J é definido por

$$U_J = \{t : (E_1^t \subseteq U_I) \text{ e } (E_0^t \subseteq U_I^c)\}, \quad (179)$$

onde U_I^c é o complemento do conjunto U_I com relação ao conjunto D_I. Ou seja, $J(t) = 1$ se a configuração dos pixels em torno de t satisfizer simultaneamente as configurações E_0^t e E_1^t.

Outra forma de calcular $\hat{J}$ é

$$\hat{J} = (\hat{I} \ominus E_1) \cap (\hat{I}^c \ominus E_0) \quad (180)$$

5 Exercícios

1. Aplique os conceitos acima em exemplos numéricos e gráficos envolvendo imagens binárias e cinza.

2. Implemente as funções acima.

1 Reconstrução morfológica

A reconstrução morfológica é uma operação monotônica e idempotente, que envolve duas imagens de entrada, uma máscara $\hat{J} = (D_J, J)$ e uma marcadora $\hat{I} = (D_J, I)$, e um elemento estruturante planar E. Esta operação usa o conceito de dilatação (ou erosão) geodésica. Por isso, alguns autores também classificam a reconstrução como uma transformação geodésica— i.e. uma transformação morfológica aplicada à imagem marcadora, cujo resultado é forçado a ser menor (ou maior) ou igual à máscara.

1.1 Dilatação e erosão geodésicas

A dilatação geodésica de uma marcadora $\hat{I}$ por um elemento E restrita a uma máscara $\hat{J} \geq \hat{I}$ é definida por

$$(\hat{I} \oplus E) \quad \wedge \quad \hat{J} \tag{181}$$

onde $\wedge$ calcula o valor mínimo pixel a pixel entre duas imagens.

A erosão geodésica de uma marcadora $\hat{I}$ por um elemento E restrita a uma máscara $\hat{J} \leq \hat{I}$ é definida por

$$(\hat{I} \ominus E) \quad \vee \quad \hat{J} \tag{182}$$

onde $\vee$ calcula o valor máximo pixel a pixel entre duas imagens.

Note que a erosão geodésica é a dual da dilatação geodésica— i.e. $((\hat{I}^c \oplus E) \wedge \hat{J}^c)^c = (\hat{I} \ominus E) \vee \hat{J}$. No caso binário, $\wedge$ e $\vee$ podem ser substituídos por $\cap$ (intersecção) e $\cup$ (união) entre conjuntos.

1.2 Reconstrução por dilatação e por erosão

A reconstrução por dilatação (ou reconstrução inferior) de $\hat{J}$ (máscara) a partir de $\hat{I}$ (marcadora) é uma imagem $\hat{R} = (D_I, R)$, $\hat{I} \leq \hat{R} \leq \hat{J}$, obtida por sucessivas dilatações geodésicas de $\hat{I}$ restritas à $\hat{J}$ até idempotência.

A reconstrução por erosão (ou reconstrução superior) de $\hat{J}$ (máscara) a partir de $\hat{I}$ (marcadora) é uma imagem $\hat{R} = (D_I, R)$, $\hat{I} \geq \hat{R} \geq \hat{J}$, obtida por sucessivas erosões geodésicas de $\hat{I}$ restritas à $\hat{J}$ até idempotência.

2 Operador conexo

Considere o relevo que representa uma imagem cinza $\hat{J}$. Um platô (*flat zone*) neste relevo é um componente conexo maximal, onde todos os pixels possuem o mesmo valor (mesma altitude). Um operador ψ é dito conexo se e somente se qualquer par de pixels pertencentes a um dado platô em $\hat{J}$ também pertencem a um mesmo platô em $\psi(\hat{J})$. A principal vantagem é que a operação conexa não cria falsas bordas, como a filtragem linear, apenas elimina bordas entre platôs.

Outra forma de visualizar uma operação conexa é através da decomposição por limiar. A decomposição de uma imagem $\hat{J}$ por limiar forma um conjunto T_J de imagens binárias $\hat{J}_l = (D_J, J_l)$, $l = 0, 1, \ldots, \max_{\forall p \in D_J}\{J(p)\}$, onde $J_l(p) = 1$ se $J(p) \geq l$, e 0 no caso contrário. Operadores conexos apenas eliminam ou unem componentes conexos deste conjunto.

Um platô é dito mínimo regional (máximo regional) se a intensidade dos pixels nos platôs vizinhos for estritamente maior (menor) que a intensidade no platô.

2.1 Reconstrução como uma operação conexa

Considere a decomposição por limiar T_J de $\hat{J}$ e os máximos regionais de uma imagem marcadora $\hat{I} \leq \hat{J}$. Esses máximos caem em alguns componentes conexos com valor 1 em T_J. Imagine a operação conexa que seleciona como *foreground* todos componentes-1 de T_J que contêm um máximo regional em $\hat{I}$ e pertencem a um nível l menor ou igual ao valor deste máximo, atribuindo 0 aos demais e gerando uma nova decomposição T_R. Note que, a imagem reconstruída $\hat{R}$ é a reconstrução inferior de $\hat{J}$ a partir de $\hat{I}$.

Agora considere a decomposição por limiar T_J de $\hat{J}$ e os mínimos regionais de uma imagem marcadora $\hat{I} \geq \hat{J}$. Esses mínimos caem em alguns componentes conexos com valor 0 em T_J. Imagine a operação conexa que seleciona como *background* todos componentes-0 de T_J que contêm um mínimo regional em $\hat{I}$ e pertencem a um nível l estritamente maior que o valor deste mínimo, atribuindo 1 aos demais e gerando uma nova decomposição T_R. Note que, a imagem reconstruída $\hat{R}$ é a reconstrução superior de $\hat{J}$ a partir de $\hat{I}$.

Genericamente, a seleção de componentes pode ser feita pela marcação de pixels com certa altitude. Todos componentes não selecionados mudam de categoria de *foreground* para *background*, ou vice-versa. Outro exemplo é a seleção por área. Imagine que são selecionados os componentes-1 de T_J com valor de área maior ou igual a um dado limiar. Esta operação conexa é denominada abertura por área (*area opening*). Sua operação dual é o fechamento por área (*area closing*), que seleciona os componentes-0 de T_J com valor de área estritamente maior que um dado limiar.

3 Filtragem por reconstrução

Operações de abertura e fechamento suavizam a imagem, mas borram as bordas. Uma forma de corrigir esta distorção é calculando a reconstrução morfológica. As seções seguintes apresentam alguns filtros por reconstrução e seus resíduos.

3.1 Abertura e fechamento

A abertura por reconstrução de $\hat{J}$ é a reconstrução inferior de $\hat{J}$ a partir de $\hat{I} = \hat{J} \circ E$. O fechamento por reconstrução de $\hat{J}$ é a reconstrução superior de $\hat{J}$ a partir de $\hat{I} = \hat{J} \bullet E$. Os resíduos dessas operações são denominados *top-hats* por reconstrução.

3.2 *H-domes* e *H-basins*

A imagem resíduo da reconstrução inferior de $\hat{J}$ a partir de $\hat{I} = \hat{J} - H$, onde H é um número inteiro positivo, é formada por domos denominados *H-domes*. Observe que para $H = 1$, os domos são os máximos regionais de $\hat{J}$. A imagem resíduo da reconstrução superior de $\hat{J}$ a partir de $\hat{I} = \hat{J} + H$, onde H é um número inteiro positivo, marca as bacias denominadas *H-basins*. Observe que para $H = 1$, a imagem resíduo é formada pelos mínimos regionais de $\hat{J}$.

3.3 Fechamento de bacias e abertura de domos

Considere uma imagem $\hat{J}$ com regiões escuras (bacias) em um fundo claro. Essas regiões podem ser eliminadas com a reconstrução superior de $\hat{J}$ a partir de $\hat{I}$, onde $I(p) = J(p)$ se p estiver na borda da imagem $\hat{J}$ ou $I(p) = \infty$, no caso contrário. Esta operação é denominada fechamento de bacias (ou "buracos"— *closing of holes*). As bacias são detectadas calculando-se o resíduo desta operação. A operação dual, que usa reconstrução inferior, é a abertura de domos (*removal of domes*).

3.4 Leveling

Todas operações acima envolveram um pré-processamento anti-extensivo ou extensivo para gerar $\hat{I}$. Em várias situações, porém, desejamos reconstruir uma imagem $\hat{J}$ a partir de uma imagem $\hat{I}$, onde $\hat{I}$ não é nem menor nem maior que $\hat{J}$. Um exemplo é quando $\hat{I}$ é obtida por filtragem seqüêncial alternada de $\hat{J}$. Neste caso, a operação *leveling* (nivelamento) pode ser aplicada seguindo as instruções abaixo.

1. Calcula $\hat{I}' = (\hat{I} \oplus E) \wedge \hat{J}$;
2. Encontra a reconstrução inferior $\hat{R}_i$ de $\hat{J}$ a partir de $\hat{I}'$;
3. Calcula $\hat{J}' = (\hat{J} \ominus E) \vee \hat{R}_i$;
4. Encontra a reconstrução superior $\hat{R}_s$ (nivelamento) de $\hat{R}_i$ a partir de $\hat{J}'$.

4 Exercícios

A implementação eficiente de todos operadores conexos acima será vista com a transformada imagem-floresta.

Aplique os conceitos acima em exemplos numéricos com imagens pequenas (ou sinais) que você mesmo vai criar. Visualize os resultados na forma gráfica no caso de sinais. Verifique a dualidade das operações.

1 Transformada Imagem-Floresta

Na aula de Introdução à Topologia Digital (aula 4) vimos que uma relação de adjacência A entre pixels define um grafo na imagem, onde os pixels são os nós, $(p, q) \in A$ é uma aresta entre pixels adjacentes e um pixel q é conexo a um pixel p se existir um caminho de p a q composto de pixels adjacentes no grafo. A Transformada Imagem-Floresta (IFT de *Image Foresting Transform*) explora esta representação para reduzir problemas de processamento de imagens baseados em conectividade em um problema de floresta de caminhos de custo mínimo (caminhos ótimos).

O custo de um caminho é calculado por uma função dependente da aplicação e com base em propriedades locais da imagem— tais como brilho, gradiente, e posição de pixel ao longo do caminho. Para uma função de custo adequada, relação de adjacência irreflexiva e um dado conjunto de pixels sementes, a IFT associa a cada pixel da imagem um caminho de custo mínimo, particionando a imagem em uma floresta de caminhos ótimos onde cada árvore tem como raiz um pixel semente e como nós os pixels da imagem mais "conexos" com a raiz do que com qualquer outra semente, em algum sentido apropriado. A IFT gera como resultado uma imagem anotada, onde cada pixel tem associado o predecessor no caminho ótimo, o custo deste caminho e o pixel raiz (ou algum rótulo associado à raiz).

Exemplos de problemas redutíveis a uma IFT são segmentação por transformada de watershed, segmentação baseada em conectividade *fuzzy*, filtragem e segmentação por reconstrução morfológica, segmentação por crescimento de regiões, segmentação por perseguição de borda, cálculo de caminhos geodésicos, transformadas de distância, representação por esqueletos multiescala, estimação de pontos de saliência em curvas, e cálculo da dimensão fractal multiescala de uma curva.

Observe que alguns casos não são facilmente relacionados com um problema de partição da imagem (e.g. reconstrução morfológica, perseguição de bordas, pontos de saliência). Porém, na maioria dos casos, a solução é obtida pela simples escolha de parâmetros da IFT seguida de um processamento local da imagem anotada, em tempo proporcional ao número de pixels. Este resultado é obtido com uma extensão do algoritmo de Dijkstra para múltiplas fontes e função de custo de caminho mais geral.

1.1 Exemplo simples

Suponha, por exemplo, a segmentação de um objeto em uma imagem 2D em tons de cinza, onde um pixel semente o é selecionado no interior do objeto e outro pixel semente o' é selecionado fora. Supondo que o objeto tem distribuição homogênea de brilho diferente do exterior, a função de custo de caminho pode ser o valor máximo das diferenças absolutas entre os valores dos pixels adjacentes ao longo do caminho. Podemos definir o valor de conectividade de um pixel p com relação ao objeto como o custo do caminho ótimo de o a p e com relação ao fundo como o custo do caminho ótimo de o' a p. Um pixel da imagem será classificado como pertencente ao objeto se sua conectividade com o objeto for maior do que sua conectividade com o fundo. Se a segmentação funcionar, a árvore de caminhos ótimos com raiz o representará o objeto desejado.

1.2 Funções de custo de caminho

Cada problema requer a escolha de uma função f, a qual associa um custo, que provêm de um conjunto V ordenado de valores com valor máximo denotado por $+\infty$, para cada caminho. Para garantir uma floresta de caminhos ótimos, esta função deve ser suave. Isto é, para todo pixel $q \in D_I$, onde D_I é o conjunto dos pixels de uma imagem $\hat{I}$, deve existir um caminho ótimo π terminando em q que deve satisfazer as condições estabelecidas em artigo submetido recentemente para a IEEE TPAMI
(http://www.math.wvu.edu/ kcies/SubmittedPapers/SS17DijkstraCharacterization.pdf).
O exemplo mais comum é a função de custo aditiva, a qual satisfaz

$$\begin{aligned} f_{sum}(\langle q \rangle) &= h(q), \\ f_{sum}(\pi \cdot \langle p, q \rangle) &= f_{sum}(\pi) + w(p, q), \end{aligned} \tag{183}$$

onde $(p, q) \in A$, π é qualquer caminho terminando em p, $h(q)$ é um custo inicial (*handicap cost*) fixo para qualquer caminho iniciando em q, e $w(p, q)$ é um peso não negativo associado ao arco (p, q).

Um outro exemplo é a função de custo $f_{\max}$, a qual satisfaz

$$\begin{aligned} f_{\max}(\langle q \rangle) &= h(q), \\ f_{\max}(\pi \cdot \langle p, q \rangle) &= \max\{f_{\max}(\pi), w(p, q)\}, \end{aligned} \tag{184}$$

onde $h(q)$ e $w(p, q)$ são fixos mas arbitrários.

De uma forma geral, os exemplos acima pertencem à classe de funções monotônicas-incrementais (MI), as quais satisfazem

$$\begin{aligned} f(\langle q \rangle) &= h(q), \\ f(\pi \cdot \langle p, q \rangle) &= f(\pi) \odot (p, q), \end{aligned} \tag{185}$$

onde $h(q)$ é arbitrário e $\odot : V \times A \to V$ é uma operação binária que satisfaz as condições

(M1) $x' \geq x \Rightarrow x' \odot (p, q) \geq x \odot (p, q)$,

(M2) $x \odot (p, q) \geq x$,

para quaisquer $x, x' \in V$ e qualquer $(p, q) \in A$, onde $\odot$ depende apenas do custo de π e não de qualquer outra propriedade de π.

Alguns operadores requerem funções mais gerais que as funções MI. Este é o caso da função f_{euc} usada em problemas que envolvem a transformada de distância Euclideana, a qual satisfaz

$$f_{euc}(\pi \cdot \langle p, q \rangle) = d^2(org(\pi), q), \tag{186}$$

onde d é a distância Euclideana entre dois pixels, $org(\pi)$ é o pixel inicial do caminho π. A suavidade de f_{euc}, porém, vai depender da relação de adjacência A adotada.

1.3 Pixels sementes

O conjunto $S \subseteq D_I$ de pixels sementes restringe a busca por caminhos ótimos que iniciam em S. Isto equivale a modificar a função f de custo para f^S, a qual satisfaz

$$f^S(\pi) \;=\; \left\{ \begin{array}{ll} f(\pi), & \text{se } org(\pi) \in S, \\ +\infty, & \text{no caso contrário.} \end{array} \right\} \tag{187}$$

No caso particular de funções MI, isto também é equivalente a definir $h(q) = +\infty$ para pixels $q \notin S$. Note que se f for MI, então f^S será MI e portanto suave. Infelizmente, isto não é necessariamente verdade se f for suave, mas não for MI. Por exemplo, f^S_{euc} com A igual à vizinhança-4 é suave para $|S| \leq 2$, mas não é suave para $|S| \geq 3$.

Observe que todas as raízes da IFT são pixels sementes, mas nem todas sementes se transformam em raízes da IFT, pois o custo de um caminho trivial $\langle q \rangle$, onde $q \in S$, pode ser maior que o custo de um outro caminho iniciado em S com término em q.

1.4 Imagem anotada

Um mapa P de predecessores é uma função que associa a cada pixel $q \in D_I$ ou outro pixel $p \in D_I$, $(p, q) \in A$, ou uma marca $nil \notin D_I$. No segundo caso, q é dito ser uma raiz do mapa. Uma floresta espalhada é um mapa de predecessores que não contém ciclos— i.e., aquele que leva todo pixel para nil em um número finito de iterações. Para qualquer pixel $q \in D_I$, a floresta P define um caminho $P^*(q)$ recursivamente como $\langle q \rangle$, se $P(q) = nil$, e $P^*(p) \cdot \langle p, q \rangle$ se $P(q) = p \neq nil$.

A IFT calcula essencialmente uma floresta P de caminhos ótimos— i.e. uma floresta espalhada onde $P^*(q)$ tem custo mínimo para todo $q \in D_I$. Para fins de eficiência, a IFT também gera um mapa de custos C e um mapa de raízes L, onde $C(q)$ é o custo do caminho ótimo até q e $L(q)$ é o pixel inicial deste caminho (ou algum rótulo associado a ele).

2 Exercícios

1. Mostre que toda função MI é suave.
2. Mostre que f^S_{euc} com vizinhança-4 é suave para $|S| \leq 2$.
3. Mostre que f_{euc} não é MI.

1 Algoritmos e estruturas de dados para a IFT

O algoritmo geral da IFT é essencialmente o Algoritmo de Dijkstra estendido para múltiplas fontes e funções de custo de caminho suaves. Note que podem existir várias florestas P de caminhos de custo mínimo que satisfazem um dado problema, apenas o mapa C de custos ótimos é único. Esta ambigüidade é parcialmente resolvida quando decidimos pelo caminho de menor custo que encontra um dado pixel primeiro. O algoritmo resultante é apresentado abaixo.

Algoritmo geral da IFT:
Entrada: Imagem $\hat{I} = (D_I, I)$, relação de adjacência A, e função f de custo de caminho suave.
Saída: Imagens $\hat{C} = (D_I, C)$ de custo, $\hat{P} = (D_I, P)$ de predecessores, e $\hat{L} = (D_I, L)$ de raízes.
Auxiliares: Fila Q de prioridade e variável c.

1. Para todo pixel $p \in D_I$ faça
2. $\quad C(p) \leftarrow f(\langle p \rangle)$, $P(p) \leftarrow nil$, e $L(p) \leftarrow p$.
3. $\quad$ Se $C(p) < +\infty$, insira p em Q.
4. Enquanto $Q \neq \emptyset$ faça
5. $\quad$ Remova um pixel p de Q cujo custo $C(p)$ é mínimo.
6. $\quad$ Para todo $q \in A(p)$, tal que $C(q) > C(p)$, faça
7. $\quad\quad c \leftarrow f(P^*(p) \cdot \langle p, q \rangle)$.
8. $\quad\quad$ Se $c < C(q)$ faça
9. $\quad\quad\quad$ Se $C(q) \neq +\infty$, remova q de Q.
10. $\quad\quad\quad C(q) \leftarrow c$, $P(q) \leftarrow p$, $L(q) \leftarrow L(p)$, e insira q em Q.

Aplicando a regra de desempate acima, inclusive para pixels distintos que são encontrados por caminhos ótimos de mesmo custo, aquele que entrou na fila Q primeiro é o primeiro a sair. Neste caso dizemos que a fila Q segue a política *First-In-First-Out* (FIFO) de desempate. Outra forma de resolver parcialmente o problema de ambigüidade das florestas ótimas é implementar a fila Q com política *Last-In-First-Out* (LIFO) de desempate. Este variante é apresentado abaixo.

Algoritmo da IFT com política de desempate LIFO:
Entrada: Imagem $\hat{I} = (D_I, I)$, relação de adjacência A, e função f de custo de caminho suave.
Saída: Imagens $\hat{C} = (D_I, C)$ de custo, $\hat{P} = (D_I, P)$ de predecessores, e $\hat{L} = (D_I, L)$ de raízes.
Auxiliares: Fila Q de prioridade e variável c.

1. Para todo pixel $p \in D_I$ faça

2. $C(p) \leftarrow f(\langle p \rangle)$, $P(p) \leftarrow nil$, $L(p) \leftarrow p$, e insira p em Q.

3. Enquanto $Q \neq \emptyset$ faça

4. Remova um pixel p de Q cujo custo $C(p)$ é mínimo.

5. Para todo $q \in A(p)$, tal que $q \in Q$, faça

6. $c \leftarrow f(P^*(p) \cdot \langle p, q \rangle)$.

7. Se $c \leq C(q)$ então

8. Remova q de Q, faça $C(q) \leftarrow c$, $P(q) \leftarrow p$, e $L(q) \leftarrow L(p)$, e insira q em Q.

A principal diferença entre este último algoritmo e o anterior está na linha 8, onde $P(q)$ e $L(q)$ devem ser atualizados, e q deve ser removido e reinserido em Q, mesmo quando c é igual a $C(q)$. Com a política FIFO, qualquer conjunto conexo de pixels que poderiam ser encontrados por duas ou mais raízes por caminhos ótimos de mesmo custo serão particionados entre as respectivas árvores. No caso da poítica LIFO, esses pixels são associados a uma mesma árvore. A política LIFO é útil em algumas situações, como no cálculo do número de mínimos regionais de uma imagem, mas a política FIFO satisfaz melhor as expectativas do usuário com relação à partição da imagem (e.g. na segmentação). Infelizmente, ambas não resolvem por completo o problema de ambigüidade das florestas ótimas e regras extras de desempate devem ser implementadas em Q, ou podemos ainda definir f como uma função de custo lexicográfica.

1.1 Alguns variantes

Diversos variantes desses algoritmos podem ser adotados visando uma maior eficiência para determinadas operações de imagem. Os casos mais simples são a propagação simultânea de rótulos, a saturação da função de custo, e a busca por caminhos específicos.

No caso de funções suaves f^S restritas a um conjunto S de pixels sementes, onde cada semente p possui um rótulo $\lambda(p)$, podemos usar $L(p) \leftarrow \lambda(p)$ na linha 2 em ambos algoritmos e eles propagarão um mapa L de rótulos em vez de raízes. Este variante é muito comum em segmentação de imagens, quando desejamos atribuir um rótulo distinto para cada objeto (incluindo o fundo).

No caso de estarmos interessados em podar as árvores ficando com os nós de custo menor ou igual a um dado limiar, podemos evitar o cálculo da floresta completa fazendo com que a função f retorne $+\infty$ na linha 7 do algoritmo geral. Este variante pode ser útil, por exemplo, em métodos de segmentação de imagens por crescimento de regiões e no cálculo de caminhos geodésicos.

Quando desejamos encontrar um caminho ótimo que chega a um dado pixel (ou a um conjunto de pixels), podemos parar o cálculo quando este pixel (ou o primeiro pixel do conjunto) sai da fila na linha 5 do algoritmo geral. Este variante é útil no cálculo de caminhos geodésicos entre conjuntos de pixels e em algoritmos de perseguição de borda.

1.2 Fila de prioridade

A implementação mais fácil para a fila Q usa um *heap* binário. Neste caso os algoritmos acima terão complexidade $O(m + n \log n)$, onde $n = |D_I|$ é o número de nós (pixels) e $m = |A|$ o número de arcos.

Na maioria das aplicações, porém, podemos usar funções de custo de caminho com incrementos de custo inteiros e limitados a uma constante K ao longo do caminho. Isto permite a utilização da fila circular de Dial com $K + 1$ posições. Cada posição i, $i = 0, 1, \ldots, K$, deve armazenar uma lista duplamente ligada de todos os pixels p com custo $i = C(p)\%K$. Como sabemos o tamanho máximo $|D_I|$ do grafo, essas listas podem ser implementadas em uma única matriz A de ponteiros $A.next(p)$ e $A.prev(p)$ com $|D_I|$ elementos. Neste caso, os algoritmos da IFT terão complexidade $O(m+nK)$. Se a adjacência A definir um grafo esparso $m \ll n$, a IFT levará tempo proporcional ao número $n = |D_I|$ de pixels.

Note que a cada instante existe um valor mínimo $C_{\min}$ e um valor máximo $C_{\max}$ de custo para os pixels armazenados em Q. A diferença $C_{\max} - C_{\min} \leq K$ deve ser mantida para garantir a corretude da fila. Em algumas aplicações sabemos que os incrementos são inteiros e limitados, mas não conhecemos o valor de K. Neste caso, a fila circular inicia com um dado tamanho K, mas antes de inserir um novo pixel devemos verificar a necessidade de realocar ou não mais elementos para a fila.

O código em C com os algoritmos da IFT, FIFO e LIFO, a implementação da fila Q com realocação dinâmica, e alguns exemplos de operadores de imagem estão disponíveis em www.ic.unicamp.br/~afalcao/ift.html.

2 Exercícios

1. Implemente a IFT-FIFO com um *heap* binário e teste seu algoritmo com algumas funções de custo de caminho suaves.

2. Calcule o maior incremento K para a função de custo de caminho $f_{euc}(\pi \cdot \langle p, q \rangle) = d^2(org(\pi), q)$, onde d é a distância Euclideana e $org(\pi)$ é o pixel inicial do caminho π.

1 Algumas aplicações da IFT

Diversos operadores de imagem podem ser reduzidos a simples escolha de parâmetros da IFT e uma operação local sobre a imagem anotada. Neste curso vamos abordar apenas os operadores relacionados com filtragem e segmentação de imagens: mínimos regionais, transformada de watershed, reconstrução morfológica, e perseguição de bordas.

1.1 Mínimos regionais

Os mínimos regionais de uma imagem $\hat{I} = (D_I, I)$ podem ser calculados diretamente do mapa L de raízes, com a função f_{ini} descrita abaixo.

$$\begin{aligned} f_{ini}(\langle q \rangle) &= I(q), \text{para todo } q \in D_I, \\ f_{ini}(\pi \cdot \langle p, q \rangle) &= \begin{cases} f_{ini}(\pi), & \text{se } I(p) \leq I(q), \\ +\infty, & \text{no caso contrário.} \end{cases} \end{aligned} \tag{188}$$

Os mínimos regionais podem ser usados como sementes da IFT em algumas tarefas de segmentação. Com a política de desempate FIFO, um pixel será raiz da IFT se e somente se ele pertencer a um mínimo regional. Portanto, uma imagem binária dos mínimos regionais pode ser gerada associando 1 a pixels raízes e 0 aos demais. Com a política LIFO, vamos obter exatamente um pixel por mínimo regional. Neste caso, temos uma contagem direta do número de mínimos e a extensão desses mínimos na imagem é obtida podando as árvores com raíz r para escolher os pixels p com $I(p) = I(r)$.

1.2 Transformada de watershed

Considere uma imagem $\hat{I} = (D_I, I)$ onde $I(p)$ é a altitude dos pixels. A transformada clássica de watershed simula a inundação desta superfície por fontes de água colocadas uma em cada mínimo regional; e uma barreira (linhas de watershed) sendo erguida toda vez que águas provenientes de fontes distintas se encontram, impedindo assim que elas se misturem. Essas linhas podem ser obtidas direto do mapa L de raízes (bacias), usando a função f_{peak} (caso particular de $f_{\max}$) com $h(q) = I(q) + 1$ para todo pixel $q \in D_I$ e $w(p, q) = I(q)$.

$$\begin{aligned} f_{peak}(\langle q \rangle) &= h(q), \\ f_{peak}(\pi \cdot \langle p, q \rangle) &= \max\{f_{peak}(\pi), I(q)\}. \end{aligned} \tag{189}$$

Para obter linhas de watershed com espessura máxima de 2 pixels, classificamos como pertencentes à linha todos os pixels p com raíz $L(p) \neq L(q)$ para algum q vizinho-4 de p. Se atribuirmos um número inteiro distinto para cada mínimo, podemos obter linhas com espessura de 1 pixel, basta classificarmos como linha todos os pixels p com raíz $L(p) < L(q)$ para algum q vizinho-4 de p.

Com a política FIFO, todas as raízes da floresta serão mínimos regionais de $\hat{I}$, e obteremos um pixel por mínimo regional de $\hat{I}$ com a política LIFO. Note que a partição L reflete as zonas de influência desses mínimos. Se desejarmos uma divisão o mais igualitária quanto possível

de platôs do mapa C de custo alcançados por mais que um mínimo regional, então a política FIFO é a melhor opção. Este normalmente é o caso em segmentação por transformada de watershed como veremos mais adiante neste curso. Por outro lado, a posição da barreira no platô fica melhor definida com a política LIFO, que atribuirá o platô inteiro para o último mínimo a alcançá-lo.

Como o número de mínimos regionais é normalmente muito elevado, o mapa de raízes fica supersegmentado. Uma forma de resolver a supersegmentação é o uso de um conjunto S de marcadores (sementes) em um número bem menor que o de mínimos, de maneira que as raízes da floresta serão esses marcadores. Neste caso, a transformada de watershed com imposição de marcadores é obtida com função f_{peak} para $h(q) < I(q)$, se $q \in S$, e $h(q) = +\infty$ no caso contrário. Observe que apenas na política FIFO, podemos ter imposição de marcadores com $h(q) \leq I(q)$, se $q \in S$, e $h(q) = +\infty$ no caso contrário.

1.3 Reconstrução morfológica

Observe que o mapa C de custos da transformada clássica de watershed equivale ao resultado da reconstrução superior de $\hat{I}$ a partir de uma imagem marcadora $\hat{h} = (D_I, h)$. Na verdade, este resultado é estendido para qualquer imagem marcadora $\hat{h} > \hat{I}$ (i.e. $h(q) > I(q)$ para todo $q \in D_I$), inclusive com pixels sementes (i.e. $h(q) > I(q)$ para $q \in S$ e $h(q) = +\infty$ no caso contrário). No último caso, porém,não existe imposição de marcadores. Isto é, algumas sementes podem ser dominadas por outras e não resultarem em raízes da floresta. O mapa de raízes também equivale às bacias da transformada de watershed da imagem de custos $\hat{C} = (D_I, C)$ com fontes colocadas nas raízes da floresta. Este é um resultado importante que relaciona transformada de watershed com reconstrução morfológica.

Um variante interessante é a reconstrução superior local, a qual inunda uma ou mais bacias selecionadas por um conjunto S de pixels sementes com função f_{lrec}, com $h(q) > I(q)$.

$$\begin{aligned} f_{lrec}(\langle q \rangle) &= h(q), \text{ se } q \in S, \text{ e } +\infty \text{ no caso contrário,} \\ f_{lrec}(\pi \cdot \langle p, q \rangle) &= \begin{cases} f_{lrec}(\pi), & \text{se } f_{lrec}(\pi) > I(q), \\ +\infty, & \text{no caso contrário.} \end{cases} \end{aligned} \tag{190}$$

Note que a reconstrução só é aplicada nas bacias selecionadas. O restante dos pixels em C ficam com custo infinito. Se o objetivo for rotular apenas as bacias selecionadas, o resultado é obtido diretamente de L, mas se o objetivo for gerar uma imagem com menos bacias do que a original, então podemos gerar uma imagem $\hat{J} = (D_I, J)$, onde $J(p) = I(p)$, se $C(p) = +\infty$, e $J(p) = C(p)$, no caso contrário. Esta última operação é usada, por exemplo, para implementar o fechamento por área (*area closing*).

1.4 Perseguição de bordas

Considere o problema de perseguir a borda de um objeto em uma imagem $\hat{I} = (D_I, I)$ de uma marca inicial M_i a uma marca final M_f, onde essas marcas são conjuntos de pixels que cruzam a borda. Na literatura de processamento de imagens, as abordagens para resolver este

problema usam busca heurística em grafos e programação dinâmica. A solução via IFT adota como conjunto de sementes $S = M_i$, vizinhança-4 (ou 8) e uma função de custo de caminho f_{ctrack} (caso particular de f_{sum}) dada abaixo.

$$f_{ctrack}(\langle q \rangle) = \begin{cases} 0, & \text{se } q \in S, \\ +\infty, & \text{no caso contrário.} \end{cases} \quad (191)$$

$$f_{ctrack}(\pi \cdot \langle p, q \rangle) = f_{ctrack}(\pi) + (K - \max\{G(p,q) \cdot \eta(p,q), 0\}), \quad (192)$$

onde $G(p,q)$ é um vetor gradiente estimado no ponto médio do arco (p,q); $\eta(p,q)$ é o arco (p,q) rotacionado de 90 graus no sentido anti-horário; e K é um limite superior para $|G(p,q) \cdot \eta(p,q)|$. Observe que esta formulação leva em conta uma orientação para a borda, assumindo que o objeto é mais escuro dentro do que fora, por exemplo. Isto evita bordas com propriedades similares, mas orientação oposta.

A solução é o caminho ótimo $P^*(q)$, onde q é o primeiro pixel de M_f a sair da fila Q e sua origem $org(P^*(q)) \in M_i$. Podemos então usar o variante de busca por caminhos específicos para tornar o algoritmo mais eficiente.

2 Exercícios

1. As operações morfológicas acima usam essencialmente o conceito de erosão geodésica. Descreva as funções de custo de caminho para implementar o dual dessas operações.

2. Mostre alguns exemplos, usando a representação 1D do relevo de uma imagem, dos resultados das operações morfológicas acima.

3. Descreva um método de segmentação baseado no operador de perseguição de bordas.

4. Descreva um método de segmentação baseado na transformada de watershed.

5. Implemente uma biblioteca de funções com filtros morfológicos e transformadas de watershed usando os conceitos vistos nas últimas aulas e o código da IFT genérica disponível em www.ic.unicamp.br/˜afalcao/ift.html.

1 Introdução à segmentação de imagens

Segmentar uma imagem consiste em particioná-la em regiões de pixels relevantes para uma dada aplicação (i.e. objetos e fundo). A segmentação é uma das principais etapas na maioria das aplicações e representa um dos maiores desafios em processamento de imagens. A principal razão desta dificuldade está na falta de informação sobre os objetos nas imagens.

A segmentação consiste de duas tarefas básicas: identificação e delineamento. A identificação indica a localização aproximada do objeto e o delineamento extrai sua extensão na imagem. Seres humanos executam a primeira tarefa com relativa facilidade, mas o computador é capaz de executar a segunda com muito mais precisão do que os seres humanos. A dificuldade da máquina na identificação de objetos se deve a falta de uma descrição global dos objetos na forma de um modelo matemático, pois as decisões automáticas usam normalmente propriedades locais extraídas em torno dos pixels. Por exemplo, sabendo que o objeto de interesse tem a forma triangular, como incorporar esta informação no modelo de segmentação para que o algoritmo procure por objetos triangulares?

Métodos de segmentação podem ser classificados pelo tipo de representação que extraem, como baseados em borda ou em região. Métodos baseados em borda procuram extrair contornos fechados, onde um contorno fechado é um caminho de pixels 4- ou 8-adjacentes que separa o interior do exterior do objeto. Isto é, todo caminho 4-conexo vindo de dentro para fora, ou vice-versa, deve cruzar a borda em um pixel. Se o caminho for 8-conexo, então deve cruzar a borda em pelo menos um vértice de pixel. Algumas abordagens também definem orientação para o contorno. Isto é, se caminharmos ao longo da borda teremos sempre um dos lados (direito ou esquerdo) no interior e o outro no exterior do objeto. Métodos baseados em região extraem o conjunto de pixels que representa o interior do objeto, incluindo os pixels de fronteira. Algumas abordagens classificadas como híbridas utilizam estratégias baseadas em bordas e em região simultaneamente, independente da representação final.

A segmentação de uma imagem $\hat{I} = (D_I, I)$ baseada em região pode ser vista como um mapeamento que associa para todo pixel $p \in D_I$ um inteiro $L(p)$, denominado rótulo, cujo valor é diferente para cada objeto (incluindo o fundo). Neste caso, a segmentação é dita *hard* porque cada pixel p só pertence a um único objeto. Algumas abordagens estendem este conceito para *fuzzy*, onde cada pixel p pertence a todos os objetos com diferentes graus de pertinência. Por exemplo, calcula-se um grau de pertinência $0 \leq M(p, o) \leq 1$ do pixel p com o objeto o, tal que $\sum_{o'=1}^{k} M(p, o') = 1$ onde k é o número de objetos. Observe que também podemos entender uma borda de objeto como um conjunto de pixels em torno de uma fronteira e aplicarmos a abordagem *fuzzy*.

Métodos de segmentação também podem ser classificados como interativos ou automáticos. Métodos automáticos evitam a intervenção do usuário, mas nem sempre garantem o resultado desejado. Métodos interativos variam de técnicas manuais, onde o usuário pinta as regiões ou delineia as bordas dos objetos, a métodos que procuram minimizar o envolvimento e o tempo total do usuário na segmentação. Em abordagens interativas, uma sugestão é atribuir ao usuário a tarefa de identificação e ao computador a tarefa de delineamento.

1.1 Técnicas baseadas em região

A forma mais simples de particionar uma imagem é dividí-la em duas regiões, objeto e fundo, aplicando uma classificação pixel a pixel. Esta classificação pode ser baseada em uma única característica (e.g. o brilho) ou em um conjunto de características (e.g. brilho, gradiente, matiz) dos pixels. A classificação também se aplica a múltiplas regiões (objetos) que formam aglomerados (*clusters*) no espaço de características. Exemplos dessas técnicas são limiarização (*thresholding*), classificação estatística, redes neurais, e *clustering*. Entre essas, limiarização e clustering são as mais usadas para classificação de pixels.

A limiarização estabelece, por exemplo, um intervalo disjunto de brilho para cada região, e classifica o pixel cujo brilho está em um dado intervalo como pertencente à região correspondente. Sua extensão para múltiplas características, onde os intervalos se transformam em hipercubos, recebe o nome de método do paralelepípedo. Técnicas de clustering podem ser divididas entre particionais (e.g. algoritmo *k-means*) e hierárquicas (e.g. algoritmo *single-linkage*), sendo as hierárquicas divididas entre aglomerativas e divisivas. Técnicas particionais dividem o espaço de características em um número k de regiões satisfazendo um dado critério de dissimilaridade entre elas, enquanto os métodos hierárquicos tratam simultaneamente todos as possíveis partições. Métodos aglomerativos partem da partição onde cada pixel forma uma região para o caso onde todos os pixels formam a mesma região, e os métodos divisivos fazem o caminho inverso.

Um aspecto importante na segmentação de imagens é a conexidade entre pixels que pertencem a um mesmo componente. As técnicas mencionadas acima não exploram a conexidade, exceto alguns algoritmos de *clustering* que podem ser aplicados na imagem usando a relação de adjacência em vez de serem aplicados no espaço de características. A conexidade é explorada em técnicas de crescimento de regiões e técnicas que usam divisão e conquista de regiões conexas. O crescimento de regiões usa um conjunto de pixels sementes e um critério de parada. Em algumas abordagens, as regiões crescem a partir de sementes marcadas em um único objeto até atingirem o critério de parada (e.g. páre de crescer a região R se $I(q) < T_1$ ou $I(q) > T_2$, para $T_1 < T_2$, $q \in A(p)$, e $p \in R$), idealmente na fronteira do objeto com o fundo. Outras abordagens usam sementes em vários objetos (incluindo o fundo) e o critério de parada passa a ser definido pelo choque entre as regiões. Técnicas de divisão e conquista iniciam com a imagem representando uma única região R com um predicado $P(R)$ (e.g. 80% dos pixels possuem o mesmo brilho) associado, e aplicam divisões e uniões sucessivas. Uma divisão de R em regiões menores é a plicada sempre que $P(R)$ for falso. Uma união de duas regiões vizinhas é aplicada sempre que o predicado da região resultante for verdadeiro. O processo simplifica a imagem à medida que obtém regiões com predicado verdadeiro e pára quando nenhuma divisão e união forem mais possíveis. Ao final aplica-se uma limiarização para completar a segmentação.

1.2 Técnicas baseadas em borda

A abordagem mais simples é por classificação de pixels. Aplica-se um filtro de suavização, seguido de um realce de bordas (e.g. magnitude de gradiente, Laplaciano) e depois uma classificação binária (borda/fundo) para decidir quais pixels pertencem à borda de um objeto. Esta

abordagem deixa "buracos" na borda. Critérios locais e globais são aplicados para unir pixels que pertencem a uma mesma borda (técnicas de *edge linking*). Critérios locais buscam por segmentos próximos a cada segmento que possam pertencer a mesma borda (e.g. limiarização por histeresis) e os critérios globais assumem que segmentos de uma mesma borda satisfazem uma dada equação (e.g. Transformada de Hough).

Uma outra forma de abordar o problema é evitar a binarização da imagem de bordas realçadas, transformar a imagem em um grafo, e aplicar um algoritmo de busca por caminhos ótimos no grafo, onde cada caminho é um segmento de borda. Técnicas baseadas em busca heurística usando o algoritmo A^* e programação dinâmica são as mais populares. Essas técnicas costumavam impor restrições topológicas e geométricas para a borda e não consideravam todos os arcos de modo que nem sempre guarantiam uma solução. Como veremos mais adiante, a IFT generaliza essas técnicas eliminando esses problemas.

Um aspecto negativo nas abordagens acima é a falta de informação global sobre o objeto no modelo de segmentação. Métodos baseados em contornos deformáveis procuram resolver o problema no framework de Equações Parciais e Diferenciais (PDE). A idéia é partir de um contorno inicial que deforma-se para minimizar um funcional de energia, o qual deve ser mínimo quando o contorno adere à borda do objeto. Na maioria das técnicas, porém, a informação relevante para extrair o objeto não é incorporada no funcional de energia e o método falha na segmentação. Por exemplo, o funcional assume que o contorno é suave quando existem pontos de alta curvatura. Como resultado o contorno não adere às identações e protusões da borda.

1.3 Técnicas Híbridas

Existem várias formas de combinar técnicas baseadas em região com técnicas baseadas em borda. Uma idéia interessante é modelar a fronteira das regiões em abordagens de crescimento de regiões como um contorno deformável, fazendo com que o funcional de energia influencie no crescimento das regiões. O crescimento de regiões também poderia ser aplicado para inicializar o contorno deformável próximo à borda desejada, ou podemos ainda aplicar dois contornos deformáveis de dentro para fora e de fora para dentro do objeto como sementes iniciais para o crescimento de regiões.

1 Segmentação por limiarização

Analisando o histograma de uma imagem cinza $\hat{I} = (D_I, I)$ percebemos que muitas vezes os objetos de interesse são representados por elevações separáveis por intervalos $[l_i, h_i]$, $i = 1, 2, \ldots, k$, de cinza disjuntos. Nessas situações, a segmentação por limiarização gera uma imagem $\hat{L} = (D_I, L)$, onde $L(p) = i$ se $l_i \leq I(p) \leq h_i$.

A única característica usada na limiarização é o brilho dos pixels. Múltiplas características são usadas quando os intervalos de brilho não são disjuntos. A extensão desta abordagem para $n > 1$ características— denominada método do paralelepípedo— requer a análise de um histograma n-dimensional, onde os objetos devem ser separáveis por hipercubos com n arestas e uma aresta de hipercubo é definida por um intervalo no eixo da característica correspondente.

O problema fica ainda mais complicado se a superfície de separação tiver forma mais complexa que a de um hipercubo. Técnicas de classificação estatística são aplicadas nessas situações, onde cada elevação no histograma é interpretada como uma aproximação da densidade de probabilidade do vetor de características no objeto correspondente. Outra alternativa seria aplicar uma transformada dual de watershed no espaço $\Re^n$ para separar as elevações. Observe que tanto a classificação estatística quanto a transformada de watershed requerem que o aumento no número de características seja acompanhado de um aumento considerável no número de amostras que contribuem para o histograma n-dimensional. Porém, o número de amostras (pixels da imagem) é fixo. Por outro lado, essas amostras no espaço $\Re^n$ de características formam aglomerados (*clusters*) e os objetos podem ser identificados pela separação desses aglomerados usando técnicas de *clustering*.

Para simplificar, vamos considerar nesta seção problemas envolvendo a segmentação da imagem em duas classes: objeto e fundo. Este é o caso, por exemplo, da segmentação de imagens de texto, imagens de células, e imagens de realce de bordas.

1.1 Limiarização ótima

Considere uma imagem cinza $\hat{I} = (D_I, I)$ com $0 \leq I(p) \leq L-1$ para todo $p \in D_I$, duas classes de interesse, C_1 e C_2, representando objeto e fundo, e as seguintes definições.

- P_1 é a probabilidade *a priori* de ocorrer C_1.
- P_2 é a probabilidade *a priori* de ocorrer C_2.
- $p_1(l)$ é a densidade de probabilidade do nível de cinza l em C_1.
- $p_2(l)$ é a densidade de probabilidade do nível de cinza l em C_2.
- $p(l)$ é a densidade de probabilidade do nível de cinza l na imagem.

Portanto,

$$p(l) \quad = \quad P_1 p_1(l) + P_2 p_2(l), \tag{193}$$

e pela regra de decisão de Bayes, um pixel p deve ser classificado como classe 1, se $P_1 p_1(l) > P_2 p_2(l)$, e como classe 2, se $P_1 p_1(l) < P_2 p_2(l)$, ficando a igualdade a critério da implementação. Esta regra minimiza a probabilidade de erro assumindo como limiar de decisão $P_1 p_1(l) = P_2 p_2(l)$.

Supondo que $p_1(l)$ e $p_2(l)$ são distribuições Gaussianas com médias μ_1 e μ_2, e variâncias σ_1^2 e σ_2^2 conhecidas,

$$p_1(l) = \frac{1}{\sqrt{2\pi}\sigma_1} \exp\left[\frac{-(l-\mu_1)^2}{2\sigma_1^2}\right] \tag{194}$$

$$p_2(l) = \frac{1}{\sqrt{2\pi}\sigma_2} \exp\left[\frac{-(l-\mu_2)^2}{2\sigma_2^2}\right], \tag{195}$$

e que $0 \leq \mu_1 < \mu_2 \leq L-1$, os intervalos de brilho que identificam cada classe podem ser encontrados da seguinte forma.

$$\begin{aligned} P_1 p_1(T) &= P_2 p_2(T) \\ \frac{P_1}{\sqrt{2\pi}\sigma_1} \exp\left[\frac{-(T-\mu_1)^2}{2\sigma_1^2}\right] &= \frac{P_2}{\sqrt{2\pi}\sigma_2} \exp\left[\frac{-(T-\mu_2)^2}{2\sigma_2^2}\right] \\ \ln\left(\frac{P_1}{\sigma_1}\right) - \frac{(T-\mu_1)^2}{2\sigma_1^2} &= \ln\left(\frac{P_2}{\sigma_2}\right) - \frac{(T-\mu_2)^2}{2\sigma_2^2} \end{aligned}$$

Continuando o desenvolvimento chegamos à equação:

$$AT^2 + BT + C = 0, \tag{196}$$

onde

$$\begin{aligned} A &= (\sigma_1^2 - \sigma_2^2) \\ B &= 2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2) \\ C &= \left[\mu_2^2\sigma_1^2 - \mu_1^2\sigma_2^2 + 2\sigma_1^2\sigma_2^2 \ln\left(\frac{P_1\sigma_2}{\sigma_1 P_2}\right)\right], \end{aligned}$$

cujas raízes são

$$\begin{aligned} T' &= \frac{-B - \sqrt{B^2 - 4AC}}{2A} \\ T'' &= \frac{-B + \sqrt{B^2 - 4AC}}{2A}. \end{aligned}$$

Temos, portanto, 5 possibilidades.

- Caso 1: $B^2 - 4AC < 0$. O problema não tem solução. Isto ocorre, por exemplo quando $P_2 p_2(l) \ll P_1 p_1(l)$ para todo $l \in [0, L-1]$.
- Caso 2: $T' \in [0, L-1]$ e T'' não. Neste caso, $C_1 : [0, T']$ e $C_2 : [T'+1, L-1]$.

- Caso 3: $T'' \in [0, L-1]$ e T' não. Neste caso, $C_1 : [0, T'']$ e $C_2 : [T''+1, L-1]$.
- Caso 4: $T' < T''$ e $T', T'' \in [0, L-1]$. Se $T' < \mu_1$, então $C_1 : [T'+1, T'']$ e $C_2 : [0, T'] \cup [T''+1, L-1]$. Caso contrário, $C_1 : [0, T'] \cup [T''+1, L-1]$ e $C_2 : [T'+1, T'']$.
- Caso 5: $T = T' = T''$ e $T', T'' \in [0, L-1]$.
  - Se $B^2 - 4AC = 0$, então $T = \frac{-B}{2A} = \frac{\mu_2\sigma_1^2 - \mu_1\sigma_2^2}{\sigma_1^2 - \sigma_2^2}$. Neste caso, $C_1 : [0, T]$ e $C_2 : [T+1, L-1]$.
  - Se $\sigma_1 = \sigma_2 = \sigma$, então $A = 0$ e $BT + C = 0$ implica que $T = \frac{-C}{B}$. Isto é,

$$T = \frac{\mu_1 + \mu_2}{2} + \frac{\sigma^2 \ln(P_1/P_2)}{\mu_2 - \mu_1}. \quad (197)$$

    Observe que se $P_1 = P_2$, $T = \frac{\mu_1+\mu_2}{2}$.

1.2 Limiarização por modelo de aproximação

O único problema da abordagem anterior é que não conhecemos os valores de μ_1, μ_2, σ_1, σ_2, P_1 e P_2. Sabemos apenas que $P_1 + P_2 = 1$. Uma alternativa é aproximar as distribuições Gaussianas usando otimização numérica. Considerando $h(l)$ o histograma normalizado da imagem. Isto é, uma aproximação de $p(l)$. Temos que o erro quadrático médio é

$$E(P_1, \mu_1, \mu_2, \sigma_1, \sigma_2) = \frac{1}{L}\sum_{l=0}^{L-1} |p(l) - h(l)|^2. \quad (198)$$

Seja $v = \begin{bmatrix} P_1 \\ \mu_1 \\ \mu_2 \\ \sigma_1 \\ \sigma_2 \end{bmatrix}$. A derivada $\frac{dE}{dv} = \frac{2}{L}\sum_{l=0}^{L-1} |p(l) - h(l)| \begin{bmatrix} dp(l)/dP_1 \\ dp(l)/d\mu_1 \\ dp(l)/d\mu_2 \\ dp(l)/d\sigma_1 \\ dp(l)/d\sigma_2 \end{bmatrix}$. Conhecemos o histograma $h(l)$, e $p(l)$ é dada pela Equação 193, onde P_2 pode ser substituída por $(1 - P_1)$. As derivadas da Equação 193 com relação aos parâmetros são

$$dp(l)/dP_1 = p_1(l) - p_2(l) \quad (199)$$

$$dp(l)/d\mu_1 = P_1 p_1(l)\frac{(l - \mu_1)}{\sigma_1^2} \quad (200)$$

$$dp(l)/d\mu_2 = (1 - P_1)p_2(l)\frac{(l - \mu_2)}{\sigma_2^2} \quad (201)$$

$$dp(l)/d\sigma_1 = P_1 p_1(l)\frac{1}{\sigma_1}\left[\frac{(l - \mu_1)^2}{\sigma_1^2} - 1\right] \quad (202)$$

$$dp(l)/d\sigma_2 = (1 - P_1)p_2(l)\frac{1}{\sigma_2}\left[\frac{(l - \mu_2)^2}{\sigma_2^2} - 1\right], \quad (203)$$

onde $p_1(l)$ e $p_2(l)$ são dadas pelas Equações 194 e 195. A idéia é partir de um valor inicial $v^{(0)}$, para a iteração $t = 0$, e aplicar iterativamente a equação

$$v^{(t+1)} = v^{(t)} - \alpha \frac{dE}{dv}, \tag{204}$$

onde $0 < \alpha < 1$ é uma constante, até minimizarmos $E(P_1, \mu_1, \mu_2, \sigma_1, \sigma_2)$. Por exemplo, podemos usar o método de limiarização de Otsu para encontrar $v^{(0)}$.

Otsu utiliza uma função critério

$$J(T) = \frac{P_1(T)P_2(T)[\mu_1(T) - \mu_2(T)]^2}{P_1(T)\sigma_1^2(T) + P_2(T)\sigma_2^2(T)}, \tag{205}$$

onde

$$\begin{aligned}
P_1(T) &= \sum_{l=0}^{T} h(l) \\
P_2(T) &= 1 - P_1(T) \\
\mu_1(T) &= \frac{1}{P_1(T)} \sum_{l=0}^{T} lh(l) \\
\mu_2(T) &= \frac{1}{P_2(T)} \sum_{l=T+1}^{L-1} lh(l) \\
\sigma_1^2(T) &= \frac{1}{P_1(T)} \sum_{l=0}^{T} [l - \mu_1(T)]^2 h(l) \\
\sigma_2^2(T) &= \frac{1}{P_2(T)} \sum_{l=T+1}^{L-1} [l - \mu_2(T)]^2 h(l).
\end{aligned}$$

Variando T de 0 a $L-1$, o objetivo é encontrar o limiar T' para o qual $J(T')$ é máxima. Otsu usa este limiar para separar as classes $C_1 : [0, T']$ e $C_2 : [T' + 1, L - 1]$. No caso do modelo de aproximação, podemos adotar $v^{(0)} = \begin{bmatrix} P_1(T') \\ \mu_1(T') \\ \mu_2(T') \\ \sigma_1(T') \\ \sigma_2(T') \end{bmatrix}$.

Observe que as técnicas acima foram aplicadas para toda imagem. Nada impede que a imagem seja dividida em blocos de pixels e uma regra de limiarização seja definida para cada bloco.

1.3 Limiarização por reconstrução morfológica

Em muitas situações o histograma da imagem não é bimodal, mas o objeto de interesse é mais claro (ou mais escuro) do que os valores dos pixels em uma vizinhança ao redor do objeto. A reconstrução morfológica pode ser usada para transformar o histograma em bimodal, sendo o

objeto representado por uma elevação e o fundo por outra, e as técnicas acima são aplicadas para concluir a tarefa de segmentação.

Supondo que o objeto é mais escuro que sua vizinhança imediata, podemos aplicar uma IFT com uma semente no interior do objeto e função de custo de caminho f_{srec}

$$\begin{aligned} f_{srec}(\langle q \rangle) &= I(q), \text{ se } q \in S, \text{ e } +\infty \text{ no caso contrário.} \\ f_{srec}(\pi \cdot \langle p, q \rangle) &= \max\{f_{srec}(\pi), I(q)\}. \end{aligned} \tag{206}$$

O mapa de custos da IFT é o resultado desta reconstrução morfológica superior, onde o fundo ficará com intensidade igual ou maior que a da vizinhança do objeto.

1 Crescimento de regiões

Considere uma imagem $\hat{I} = (D_I, I)$ com k objetos $\{O_1, O_2, \ldots, O_k\}$, $\cap_{i=1}^{k} O_i = \emptyset$. Podemos assumir um conjunto S_i de pixels sementes selecionados em cada objeto e um rótulo $i = 1, 2, \ldots, k$ por objeto. Esta seleção pode ser automática, baseada em algum conhecimento *a priori*, ou manual. A idéia é crescer regiões conexas às sementes por agregamento de pixels adjacentes até que essas regiões definam os objetos. Assume-se certa homogeneidade no interior dos objetos. Este crescimento de regiões pode ser feito com ou sem competição entre as sementes.

1.1 Crescimento de regiões sem competição entre sementes

Nesta abordagem consideramos os objetos O_i como de interesse e separados dos pixels que constituem o fundo da imagem, $\cup_{i=1}^{k} O_i \subset D_I$. Um algoritmo simples é apresentado abaixo.

Algoritmo de crescimento de regiões sem competição entre sementes:

Entrada: Imagem cinza $\hat{I} = (D_I, I)$, conjuntos S_i, $i = 1, \ldots, k$, de sementes, relação de adjacência A, e função $f : D_I \rightarrow \{verdadeiro, falso\}$ que estabelece um critério de parada.
Saída: Imagem rotulada $\hat{L} = (D_I, L)$, onde $L(p) = i$ se $p \in O_i$, e $L(p) = 0$ se p pertence ao fundo.
Auxiliar: Conjunto Q.

1. Para todo pixel $p \in D_I$, se $p \in S_i$ então faça $L(p) \leftarrow i$ e insira p em Q. No caso contrário, faça $L(p) \leftarrow 0$.
2. Enquanto $Q \neq \emptyset$ faça
3. Remova p de Q.
4. Para todo $q \in A(p)$, tal que $L(q) = 0$, faça
5. Se $f(q) = falso$ então faça $L(q) \leftarrow L(p)$ e insira q em Q.

Poderíamos usar, por exemplo, $f(q) = false$ se $T_1 \leq I(q) \leq T_2$ para limiares $T_1 < T_2$, e verdadeiro no caso contrário. Poderíamos ainda adotar limiares diferentes por objeto. Uma desvantagem é que o critério para decidir se um pixel pertence a um dado objeto é local, dependendo apenas das propriedades do pixel ou de uma vizinhança do pixel. Uma abordagem mais interessante é definir uma força de conectividade de p com relação a um dado objeto O_i como o custo de um caminho ótimo do conjunto S_i até p. Por ser ótimo, o algoritmo deve considerar todos os possíveis caminhos de S_i até p em D_I, e portanto, as propriedades de seus pixels. Se esta força de conectividade for maior que um dado limiar T_i, então dizemos que $p \in O_i$. Esta abordagem é também conhecida como conectividade *fuzzy*-absoluta e pode ser

implementada por uma IFT com função de custo de caminho f_{afuz} (complementar de força de conectividade), por exemplo:

$$\begin{aligned} f_{afuz}(\langle q \rangle) &= 0, \text{ se } q \in S_i, \text{ ou } +\infty \text{ no caso contrário.} \\ f_{afuz}(\pi \cdot \langle p,q \rangle) &= \begin{cases} \max\{f_{afuz}(\pi), w(p,q)\}, & \text{se } f_{afuz}(\pi \cdot \langle p,q \rangle) \leq T_i, \text{ e} \\ +\infty, & \text{no caso contrário.} \end{cases} \end{aligned} \quad (207)$$

onde $w(p,q) = K * (1 - \alpha_i(p,q))$ é o complemento da afinidade entre a aresta (p,q) e o objeto O_i,

$$\alpha_i(p,q) = \exp\left[\frac{-(\frac{I(p)+I(q)}{2} - \mu_i)^2}{2\sigma_i^2}\right], \quad (208)$$

K é o valor máximo de I, (μ_i, σ_i) são média e desvio padrão das intensidades de pixel no objeto i.

Observe que se o objeto O_i for uma bacia em I, podemos calcular a reconstrução superior local usando uma IFT com função de custo f_{lrec} e única semente em S_i.

$$\begin{aligned} f_{lrec}(\langle q \rangle) &= h(q) > I(q), \text{ se } q \in S_i, \text{ e } +\infty \text{ no caso contrário,} \\ f_{lrec}(\pi \cdot \langle p,q \rangle) &= \begin{cases} f_{lrec}(\pi), & \text{se } I(p) > I(q), \\ +\infty, & \text{no caso contrário.} \end{cases} \end{aligned} \quad (209)$$

Neste caso, o crescimento de regiões tem como critério de parada $I(p) \leq I(q)$.

Uma questão a ser considerada é que um dado pixel pode satisfazer os critérios de similaridade de mais de um objeto. No algoritmo simples acima, uma vez rotulado um pixel p como pertencente a um dado objeto O_i, esta decisão não pode mais ser mudada. Mesmo que p tenha propriedades mais similares as de outro objeto O_j, $j \neq i$, cuja semente só chegou até p depois. Na abordagem *fuzzy*-absoluta, os pixels são avaliados por ordem decrescente de conectividade (e.g. usando a fila Q de prioridade da IFT) e se dois caminhos provenientes de S_i e S_j, $i \neq j$, chegarem a um mesmo pixel com o mesmo custo, o desempate pode adotar política $FIFO$, por exemplo. Esses casos, porém, configuram uma competição entre as sementes, mesmo entre aquelas que pertencem a um mesmo objeto. Isto nos permite, também, eliminar o limiar de conectividade T_i com os objetos, fazendo com que o critério de parada seja substituído pelo choque entre as regiões.

1.2 Crescimento de regiões com competição entre sementes

Considere agora $k+1$ objetos $\{O_0, O_1, O_2, \ldots, O_k\}$, $\cap_{i=0}^{k} O_i = \emptyset$, incluindo o fundo O_0, com um conjunto de sementes S_i em cada um deles. Isto é, $\cup_{i=0}^{k} O_i = D_I$. Um pixel p deve ser atribuído ao objeto cuja conectividade com p é máxima. Esta abordagem é denominada conectividade *fuzzy*-relativa e para que os resultados teóricos sejam garantidos, o método original requer um valor de afinidade único para qualquer aresta do grafo, independente do objeto mais afim. Esta restrição pode ser traduzida em uma função de custo de caminho suave no contexto da IFT. Por exemplo:

$$\begin{aligned} f_{rfuz}(\langle q \rangle) &= 0, \text{ se } q \in S = \cup_{i=0}^{k} S_i, \text{ ou } +\infty \text{ no caso contrário.} \\ f_{rfuz}(\pi \cdot \langle p,q \rangle) &= \max\{f_{rfuz}(\pi), w(p,q)\}, \end{aligned} \quad (210)$$

onde $w(p,q) = K * (1 - \alpha(p,q))$ é o complemento da afinidade máxima entre a aresta (p,q) e os objetos $\{O_0, O_1, \ldots, O_k\}$,

$$\alpha(p,q) = \max_{i=0,1,\ldots,k} \{\alpha_i(p,q)\}, \tag{211}$$

onde $\alpha_i(p,q)$ é dada pela Equação 208.

Outro exemplo de crescimento de regiões com competição entre sementes é a transformada de watershed. Ela se aplica quando os objetos são delimitados por elevações mais altas do que as elevações internas e externas. A implementação por uma IFT pode usar, por exemplo, a função f_{peak} com imposição de sementes.

$$\begin{aligned} f_{peak}(\langle q \rangle) &= 0, \text{ se } q \in S = \cup_{i=0}^{k} S_i, \text{ ou } +\infty \text{ no caso contrário.} \\ f_{peak}(\pi \cdot \langle p,q \rangle) &= \max\{f_{peak}(\pi), I(q)\}. \end{aligned} \tag{212}$$

Lembre-se que em todos exemplos com e sem competição de sementes que usam a IFT, o resultado da segmentação é obtido do mapa de rótulos L.

2 Exercícios

1. Apresente outros critérios de parada para o algoritmo simples de crescimento de regiões sem competição de sementes.

2. Apresente outras funções de custo de caminho suaves que possam ser utilizadas nos métodos baseados em conectividade *fuzzy*.

3. Verifique se a função de custo de caminho abaixo é suave.

$$\begin{aligned} f(\langle q \rangle) &= 0, \text{ se } q \in S = \cup_{i=0}^{k} S_i, \text{ ou } +\infty \text{ no caso contrário.} \\ f(\pi \cdot \langle p,q \rangle) &= \begin{cases} \max\{f(\pi), 1 - \alpha_i(p,q)\}, & \text{se } L(p) = i, \\ 1, & \text{no caso contrário.} \end{cases} \end{aligned} \tag{213}$$

1 Perseguição de bordas

Quando examinamos a intensidade dos pixels em torno da borda de um objeto em uma imagem cinza percebemos que existe uma incerteza com relação à posição exata da borda (abordagem *hard*). Poderíamos transferir nossa incerteza para um conjunto de pixels que forma uma faixa de largura variável em torno da borda e usar este conjunto para representá-la (abordagem *fuzzy*). Considerando k conjuntos S_i, $i = 1, 2, \ldots, k$, de pixels sementes selecionados sobre uma borda, tais que $S_1 = S_k$ possui um único pixel e os pixels em S_i, $i = 2, 3, \ldots, k-1$, pertencem a uma região pequena de incerteza (e.g. uma linha que cruza a borda, uma marca circular sobre a borda), poderíamos escolher por exemplo os n caminhos de menor custo de S_i para S_{i+1}, $i = 1, 2, \ldots, k-1$, (i.e. os n caminhos mais conexos) como parte de um segmento ótimo de borda *fuzzy*, ou um único caminho de menor custo como segmento ótimo de borda *hard*. Observe que a IFT permite ambas abordagens, mas vamos tratar aqui apenas o caso de contornos ótimos.

Uma borda é um contorno ótimo que passa pela seqüência de conjuntos S_i, $i = 1, 2, \ldots, k$, de pixels sementes. O cálculo de caminhos de custo mínimo de S_i até S_{i+1}, $i = 1, 2, \ldots, k-1$, pode ser feito com uma IFT usando adjacência-4 (ou 8) e função de custo de caminho f_{ctrack}.

$$\begin{aligned} f_{ctrack}(\langle q \rangle) &= h(q) \\ f_{ctrack}(\pi \cdot \langle p, q \rangle) &= f_{ctrack}(\pi) + (K - \max\{G(p,q) \cdot \eta(p,q), 0\}), \end{aligned} \tag{214}$$

onde $h(q) = 0$, se $q \in S_1$ na primeira iteração, ou $h(q)$ é o custo final do pixel q na iteração anterior, se $q \in S_i$ e $i > 1$, ou $h(q) = +\infty$ no caso contrário, independente da iteração; $G(p,q)$ é um vetor gradiente estimado no ponto médio do arco (p,q); $\eta(p,q)$ é o arco (p,q) rotacionado de 90 graus no sentido anti-horário; e K é um limite superior para $|G(p,q) \cdot \eta(p,q)|$. O vetor $G(p,q)$ deve ser tal que arcos sobre a borda orientada do objeto possuam valores baixos de custo e os demais valores altos.

Note que a cada iteração, o algoritmo pode parar o cálculo da IFT quando o último pixel de S_{i+1} sai da fila Q. O contorno final pode ser obtido dos respectivos $k-1$ mapas de predecessores $P_{k-1}, \ldots, P_1$, mas também é possível modificar o algoritmo da IFT para usar uma única imagem anotada e um mapa de predecessores auxiliar em todas iterações.

1.1 Aplicações

O método *live-wire* é um exemplo que utiliza esta abordagem de forma interativa. Para segmentar uma borda, o usuário seleciona a primeira semente (pixel em S_1) e o método calcula uma IFT em toda imagem, cuja única árvore terá como raiz a semente selecionada. Para cada posição subseqüente do cursor, o algoritmo mostra na tela o caminho de custo mínimo da raíz até esta posição. O usuário pode mover livremente o cursor sobre a imagem e verificar os caminhos ótimos. Quando o cursor se aproxima da borda desejada, este caminho gruda na borda e o usuário então seleciona a posição atual do cursor (pixel em S_2) para confirmar o segmento ótimo de borda. O processo se repete a partir do pixel selecionado até o usuário decidir fechar o contorno.

O algoritmo da IFT também pode ser modificado para ser executado de forma incremental. O método live-wire-on-the-fly usa este variante. Neste caso nós exploramos três propriedades do algoritmo de Dijkstra.

1. O algoritmo pode parar o cálculo quando o pixel q que define a posição do cursor sai da fila Q.

2. Neste instante, todos os caminhos com custo menor que $C(q)$ já foram calculados, definindo uma região de crescimento em torno da raíz (árvore de caminhos ótimos). Então, se o usuário mover o cursor para qualquer outro pixel desta região, basta mostrar o caminho ótimo correspondente na tela.

3. Quando o pixel q sai da fila Q, a fronteira da região de crescimento que ainda está na fila Q deve ser armazenada junto com os valores de custo e predecessor de cada pixel. Se o usuário move o cursor para fora da região, então o cálculo continua a partir dos pixels de fronteira até encontrar a nova posição do cursor.

O efeito desta otimização é que após selecionar um pixel sobre a borda, o usuário movimenta o cursor e já visualiza o caminho ótimo da semente até a posição atual do cursor, mesmo em imagens grandes.

Já no método 3D-live-wire para seqüências de imagens tomográficas (fatias), o usuário executa o live-wire em cortes ortogonais ao volume criado pelo empinhamento das fatias. Cada live-wire ortogonal gera pelo menos dois pixels sobre a borda do objeto nas fatias de forma tal que obtemos uma seqüência ordenada de sementes ($|S_i| = 1$, $i = 1, 2, \ldots, k$) sobre a borda em cada fatia. Em seguida, os contornos ótimos são calculados automaticamente fatia por fatia. O método é implementado com algumas restrições para tratar mudanças de topologia ao longo das fatias.

2 Exercícios

1. Apresente outras funções de custo de caminho para perseguição de bordas.

2. Implemente uma função para calcular o contorno ótimo que passa por uma seqüência de conjuntos de pixels sementes $\{S_1, S_2, \ldots, S_k\}$, $S_1 = S_k$ possui apenas um único pixel, usando uma única imagem anotada e um mapa de predecessores auxiliar para ir copiando os caminhos obtidos em cada iteração S_i para S_{i+1}, $i = 1, 2, \ldots, k-1$.

1 Segmentação por clustering

Técnicas de clustering têm sido usadas em problemas de reconhecimento de padrões de diversas áreas. Estamos interessados na aplicação dessas técnicas para segmentação de imagens. Neste caso, cada pixel $p \in D_I$ de uma imagem $\hat{I} = (D_I, I)$ tem associado um vetor de características de imagem, tais como brilho, variância e magnitude de gradiente [2]. Para um número n de características, cada pixel p é mapeado em um ponto no espaço $\Re^n$ de características. Técnicas de clustering se baseiam na premissa que regiões de interesse para segmentação formam aglomerados de pontos separáveis em $\Re^n$, e procuram identificar esses aglomerados particionando o espaço $\Re^n$ em k regiões $R_1, R_2, \ldots, R_k$ tais que $\cap_{i=1}^{k} R_i = \emptyset$ e $\cup_{i=1}^{k} R_i = D_I$. Se cada região R_i estiver associada a um único objeto O_i, $i = 1, 2, \ldots, k$, incluindo o fundo, então a partição obtida é a solução da segmentação. Caso contrário, quando um objeto possui múltiplas regiões de pixels com características distintas, temos ainda que associar quais regiões pertencem a quais objetos. Informações globais sobre o problema— tais como características das regiões, vizinhança entre regiões, e a forma gerada pela união de regiões— poderiam ser usadas para completar a segmentação.

O particionamento em regiões requer o cálculo de distância entre pontos do $\Re^n$. Esta distância pode ser traduzida em uma dissimilaridade $d(p, q)$ entre os pixels $p, q \in D_I$ que esses pontos representam. Por exemplo podemos ter em $\Re^2$,

$$d(p, q) = [I(p) - I(q)]^2 + [G(p) - G(q)]^2, \tag{215}$$

onde $G(p)$ é a magnitude do gradiente de Sobel calculado em p. Existem várias funções de dissimilaridade, mas elas normalmente satisfazem os axiomas métricos.

1. $d(p, q) \geq 0$, e se $d(p, q) = 0$, então $p = q$.
2. $d(p, q) = d(q, p)$.
3. $d(p, r) \leq d(p, q) + d(q, r)$.

As regiões R_i são definidas dinaminamente durante o algoritmo de clustering. Isto permite que a dissimilaridade entre dois pixels varie durante o algoritmo e seja uma medida mais global do que local. Alguns variantes definem $d(p, q)$ como a dissimilaridade entre regiões R_i e R_j, onde $1 \leq i, j \leq k$, $i \neq j$, $p \in R_i$ e $q \in R_j$, levando-se em conta as características de todos os pixels que pertencem a essas regiões no dado instante.

Os métodos de clustering mais comuns podem ser divididos em particionais e hierárquicos. Métodos particionais definem um valor fixo para k e maximizam um critério de dissimilaridade entre regiões, enquanto que métodos hierárquicos consideram todas as possibilidades para $1 \leq k \leq |D_I|$. Métodos hierárquicos aglomerativos partem da partição onde cada pixel forma uma região ($k = |D_I|$) para o caso onde todos os pixels formam uma mesma região ($k = 1$), e os métodos divisivos fazem o caminho inverso.

[2] No caso de imagens coloridas, temos ainda propriedades relacionadas com os componentes de cor que podem ser usados como características.

1.1 Clustering por partição

Métodos particionais iniciam com um conjunto $S = \{p_1, p_2, \ldots, p_k\}$ de pixels representativos de cada região R_i, $p_i \in R_i$, $i = 1, 2, \ldots, k$. Esses pixels podem ser eleitos com base em algum critério automático de escolha ou selecionados manualmente pelo usuário. Uma partição é obtida por associar cada pixel $p \in D_I \backslash S$ à região cujo representante é o mais próximo em S. Após particionar a imagem, novos representativos são eleitos com base em uma função critério f aplicada a cada região. Se os representativos eleitos por f forem diferentes dos que estão em S, o algoritmo substitui esses representativos em S e repete o particionamento. O algoritmo pára quando os pixels em S são reeleitos novamente por f. Um algoritmo exemplo é dado abaixo.

Clustering por partição:

Entrada: Imagem cinza $\hat{I} = (D_I, I)$, número de regiões k, conjunto $S = \{p_1, p_2, \ldots, p_k\}$ de pixels representativos, função de dissimilaridade d, e uma função critério f.
Saída: Imagem rotulada $\hat{L} = (D_I, L)$, onde $L(p) = p_i$ para um representativo $p_i \in R_i$, $i = 1, 2, \ldots, k$.
Auxiliares: Conjunto S' e conjuntos de regiões $\{R_1, R_2, \ldots, R_k\}$.

1. $S' \leftarrow \emptyset$ e $R_i \leftarrow \emptyset$ para $i = 1, 2, \ldots, k$.
2. Para todo pixel $p \in D_I$, faça
3. Se $p \in S$ então $L(p) \leftarrow p$. Caso contrário, $L(p) \leftarrow nil$.
4. Para todo pixel $p \in D_I$, tal que $L(p) = nil$,
5. Encontre $p_i \in S$ tal que $d(p, p_i) = \min_{\forall q \in S}\{d(p, q)\}$.
6. Faça $L(p) \leftarrow p_i$.
7. Para cada pixel $p_i \in S$ associe um único conjunto R_i, $i = 1, 2, \ldots, k$.
8. Para todo pixel $p \in D_I$,
9. Encontre $p_i \in S$ tal que $L(p) = p_i$ e insira p em R_i.
10. Para $i = 1, 2, \ldots, k$ eleja um novo representativo $\hat{p}_i \in R_i$ baseado no critério $f(R_i)$ aplicado a todos os pixels em R_i, e insira $\hat{p}_i$ em S'.
11. Se $S' = S$, então páre. Caso contrário, faça $S \leftarrow S'$ e volte para a etapa 1.

O único aspecto não esclarecido no algoritmo acima é a função critério f usada na linha 10. O algoritmo *k-medoids*, por exemplo, usa como representativo de cada região R_i, o pixel $\hat{p}_i \in R_i$ que minimiza a distância média entre ele e os demais em R_i. Isto é,

$$f(R_i) \;=\; \frac{1}{|R_i|} \sum_{\forall q \in R_i} d(\hat{p}_i, q) = \min_{\forall p \in R_i} \left\{ \frac{1}{|R_i|} \sum_{\forall q \in R_i} d(p, q) \right\}. \tag{216}$$

Já o algoritmo *k-means* escolhe como representativo o pixel que minimiza a distância quadrática média entre ele e os demais pixels em R_i. Neste caso, o pixel $\hat{p}_i \in R_i$ escolhido é o mais próximo do centróide de R_i. Observe que as funções critério usadas nesses dois algoritmos tendem a encontrar aglomerados hiperesféricos e compactos em $\Re^n$. Elas não se aplicam a aglomerados alongados por exemplo.

1.2 Clustering hierárquico

Métodos hierárquicos são normalmente mais eficientes do que os particionais, porém não revertem decisões tomadas durante a construção da hierarquia. Por exemplo, uma vez decidido que dois pixels pertencem a uma mesma região em um nível da hierarquia durante um algoritmo aglomerativo, eles pertencerão a esta mesma região deste nível em diante.

Métodos aglomerativos partem da situação onde cada pixel forma uma região R_i, $i = 1, 2, \ldots, k$ e $k = |D_I|$. A cada passo, avaliam uma função de dissimilaridade $d(R_i, R_j)$ para todo par (R_i, R_j), $i \neq j$, $i, j = 1, 2, \ldots, k$, e unem os pares de regiões com **menor dissimilaridade**, reduzindo o número k de regiões. Este processo guarda a hierarquia entre as regiões e é repetido até $k = 1$.

Métodos divisivos partem da situação onde todos os pixels formam uma única região. A cada passo, avaliam uma função de dissimilaridade $d(p, q)$ entre os pixels de cada região, e dividem as regiões separando os pixels com **maior dissimilaridade**, aumentando o número k de regiões. Este processo guarda a hierarquia entre as regiões e é repetido até $k = |D_I|$.

A premissa no clustering hierárquico é que a solução desejada estará em um nível da hierarquia. Muito embora os métodos divisivos tendam a obter regiões maiores que os aglomerativos, o custo computacional alto para fazer as primeiras divisões faz com que a maioria dos algoritmos publicados sejam aglomerativos.

No contexto de métodos aglomerativos, existem várias formas de definir a dissimilaridade $d(R_i, R_j)$ entre duas regiões como função da dissimilaridade entre seus pixels. Uma alternativa para obter aglomerados hiperesféricos e compactos é definir $d(R_i, R_j)$ como a média das dissimilaridades entre todos os pares de pixels (p, q), $p \in R_i$ e $q \in R_j$. O caso mais popular é o algoritmo *single-linkage* que define $d(R_i, R_j)$ como a menor dissimilaridade entre um pixel $p \in R_i$ e um pixel $q \in R_j$. Este critério é adequado para obter aglomerados alongados. Já no caso de aglomerados compactos e hiperesféricos, mas não bem separados, a escolha de $d(R_i, R_j)$ como a maior dissimilaridade entre um pixel $p \in R_i$ e um pixel $q \in R_j$ é mais adequada. Este é o caso do algoritmo *complete-linkage*.

Em particular, o algoritmo *single-linkage* tem uma relação estreita com algoritmos que constroem uma árvore de peso mínimo para um grafo de entrada (e.g. Algoritmos de Prim e de Kruskal), onde no nosso caso, os vértices são os pixels e as arestas são definidas por uma relação de adjacência simétrica entre os pixels. Se o grafo for completo (i.e. todos os pixels da imagem são considerados adjacentes entre si), o algoritmo é dito sem restrição de conexidade. Alta eficiência pode ser obtida com restrição de conexidade para relações de adjacência pequenas (e.g. vizinhança-8). Uma vez construída a árvore de peso mínimo, a segmentação hierárquica é obtida removendo arestas da árvore com peso maior que um dado limiar T, para T variando do peso de aresta mínimo ao máximo. Isto é, cada solução possível é uma floresta

de peso mínimo, onde as regiões R_i são árvores de peso mínimo. Esta hierarquia de regiões é conhecida como dendrograma. Se o valor de T for conhecido *a priori*, podemos modificar o algoritmo para gerar o resultado da segmentação em um mapa de rótulos. Um exemplo deste variante aplicado ao algoritmo de Prim é apresentado a seguir.

Segmentação de imagens como uma floresta de peso mínimo:

Entrada: Imagem cinza $\hat{I} = (D_I, I)$, função de dissimilaridade d, e um limiar T.
Saída: Imagem rotulada $\hat{L} = (D_I, L)$, onde $L(p) = p_i$ para um representativo $p_i \in R_i$, $i = 1, 2, \ldots, k$. Imagem de predecessores $\hat{P} = (D_I, P)$ representando a floresta de peso mínimo. Imagem de custos $\hat{C} = (D_I, C)$ onde $C(p)$ é o peso da aresta que liga o pixel p com seu predecessor $P(p)$.
Auxiliares: Fila de prioridades Q e variável c.

1. Para todo pixel $p \in D_I$, faça $C(p) \leftarrow +\infty$, $L(p) \leftarrow p$, $P(p) \leftarrow nil$, e insira p em Q.
2. Enquanto $Q \neq \emptyset$, faça
3. Remova p de Q tal que $C(p)$ é mínimo.
4. Se $P(p) = nil$ então $C(p) \leftarrow 0$.
5. Para todo $q \in A(p)$, tal que $q \in Q$, faça
6. $c \leftarrow d(p, q)$.
7. Se $c < C(q)$ e $c < T$, então
8. Remova q de Q, faça $C(q) \leftarrow c$, $P(q) \leftarrow p$, e $L(q) \leftarrow L(p)$, e insira q em Q.

Observe que o algoritmo acima é essencialmente o algoritmo de Dijkstra modificado para uma função de custo $f(\pi)$ não-suave, definida como o peso da última aresta no caminho π, ou 0 caso π seja trivial. Com memória adicional, poderíamos também modificar $d(p, q)$ para representar a dissimilaridade entre as regiões com raíz $L(p)$ e $L(q)$. Se removermos o teste $c < T$ da linha 7 teremos uma árvore de peso mínimo em P.

Uma solução parecida é usada por métodos que procuram reduzir a supersegmentação criada pela transformada clássica de watershed de uma imagem de gradiente através da união de bacias. Por exemplo, constrói-se um grafo de vizinhança entre as bacias, calcula-se uma árvore de peso mínimo para este grafo usando a dissimilaridade entre bacias vizinhas, e depois escolhe-se a floresta de peso mínimo que resolve a segmentação. A floresta pode resultar da limiarização do peso das arestas ou da escolha de sementes pelo usuário. Note que em uma árvore de peso mínimo só existe um caminho que liga dois nós. Se escolhermos uma semente s_1 com rótulo objeto, por exemplo, e outra s_2 com rótulo fundo, podemos partir a árvore em duas por remover a aresta de maior peso ao longo do caminho que liga s_1 a s_2. Nesta abordagem, porém, o mapa de predecessores P deve ser transformado em um grafo não orientado e conexo, onde existe um caminho entre qualquer par de nós.

2 Exercícios

1. Proponha e analise algumas funções de dissimilaridade e critério para o algoritmo de clustering por partição.

2. Proponha um algoritmo de clustering hierárquico usando a abordagem divisiva.

3. Proponha variações do algoritmo apresentado acima para segmentação como uma floresta de peso mínimo.

1 Detecção de bordas

Esta seção apresenta duas outras abordagens para detecção do contorno fechado que descreve uma borda de objeto, como alternativa a técnica de perseguição de borda apresentada na aula 23. A primeira faz uma classificação binária dos pixels em borda ou fundo, e os segmentos de borda resultantes desta operação são ligados para formar o contorno final. A segunda parte de um contorno inicial que se deforma segundo um funcional de energia até aderir à borda do objeto na situação de energia mínima.

1.1 Classificação binária

Uma suavização para reduzir ruído seguida de um realce de bordas gera a imagem inicial para a segmentação. Uma alternativa é aplicar a técnica de segmentação por limiarização (aula 21) quando a imagem de realce é a magnitude de um vetor gradiente. Outra alternativa interessante, adotada por Marr & Hildret, usa como imagem de realce o resultado da convolução da imagem original pelo **negativo** do operador Laplaciano de uma Gaussiana $g(x, y)$ de média zero e desvio padrão σ.

$$g(x, y) = \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right). \tag{217}$$

Substituindo $r^2 = x^2 + y^2$, o Laplaciano $-\nabla^2 g(r)$ é o negativo da derivada segunda de $g(r)$ com relação a r.

$$-\nabla^2 g(r) = \left(\frac{\sigma^2 - r^2}{\sigma^4}\right) \exp\left(\frac{-r^2}{2\sigma^2}\right). \tag{218}$$

Por exemplo, para $\sigma = 1$, podemos gerar a máscara de convolução $\begin{bmatrix} -0.37 & 0 & -0.37 \\ 0 & 1 & 0 \\ -0.37 & 0 & -0.37 \end{bmatrix}$. Na imagem de realce, os pixels classificados como borda são aqueles em que ocorre uma transição de valor positivo para negativo, ou vice-versa, com relação ao valor de pelo menos um pixel adjacente— use adjacência circular de raio σ, e quanto maior for σ mais grossa será a borda. Lembre-se que para evitar convolução com máscaras grandes podemos aplicar a filtragem em freqüência.

Uma vez classificados os pixels como borda ou fundo, a imagem binária resultante apresenta "buracos" nas bordas. Uma idéia interessante pode ser usar esses segmentos binários de borda como sementes da IFT e aplicar uma perseguição de bordas na imagem de realce. A seguir apresentamos duas outras técnicas de tratamento do problema.

1.1.1 Limiarização por histeresis

A limiarização por histeresis se baseia em dois limiares de classificação binária para a imagem de realce de bordas, um limiar inferior e outro superior. A regra para classificar um pixel como borda é ele ter realce ou maior que o limiar superior ou maior que o limiar inferior, desde que conexo a outro pixel cujo realce é maior que o limiar superior. Apesar de ser uma técnica

local e simples, a limiarização por histeresis elimina várias bordas espúrias reduzindo buracos, e tem sido adotada em alguns operadores clássicos, como o operador de Canny.

1.1.2 Transformada de Hough

O princípio básico da transformada de Hough é mapear os pixels classificados como borda para um espaço de parâmetros de uma dada equação. Pixels que pertencem a segmentos de borda e que satisfazem a esta equação para um dado conjunto de parâmetros devem formar um aglomerado de pontos no espaço de parâmetros. A localização deste aglomerado de pontos permite identificar na imagem de bordas quais pixels pertencem à borda de interesse, eliminando os segmentos espúrios. A própria equação pode ser usada para fechar os buracos entre os segmentos selecionados.

O caso mais simples é a identificação de segmentos de reta. Todos os pixels que satisfazem a equação de uma reta $y = a'x + b'$ são mapeados no ponto (a', b') do espaço de parâmetros $a \times b$. Como não conhecemos a' e b', a idéia é quantizar o espaço de parâmetros para possíveis valores de a e b, depois acumular em cada posição (a, b) o número de pixels e quais pixels satisfizeram a equação $y = ax + b$. Os pixels da reta, portanto, formarão um aglomerado mais alto neste histograma bidimensional em torno de (a', b').

Um problema, porém, é que a e b assumem valores infinitos para retas verticais. Isto requer uma mudança de parâmetros para expressar a equação da reta como

$$\rho = x \cos\theta + y \sin\theta, \tag{219}$$

onde ρ é a distância da reta à origem do espaço $x \times y$ (isto é, $\rho = |\vec{n}|$, onde $\vec{n}$ é ortogonal à reta) e θ é o ângulo entre o vetor $\vec{n}$ e o eixo x. Neste caso, os valores de ρ e θ são quantizados de 0 até o valor D da diagonal da imagem e de 0° a 90°, respectivamente. Uma matriz de acumulação $A(\rho, \theta)$ (histograma bidimensional) e uma lista de pixels $P(\rho, \theta)$ podem ser usadas para calcular a seguinte transformação.

Transformada de Hough para detecção de linhas:

Entrada: Imagem binária de bordas $\hat{I} = (D_I, I)$, onde $I(p) = 1$ indica borda e $I(p) = 0$ é fundo.
Saída: Matriz de acumulação $A(\rho, \theta)$ inicializada com zeros e lista de pixels $P(\rho, \theta)$ vazia.

1. Para cada valor de $\theta = 0, 1, \ldots, 90$ faça
2. Para todo pixel $p = (x_p, y_p) \in D_I$, tal que $I(p) = 1$, faça
3. $\rho \leftarrow round(x_p \cos\theta + y_p \sin\theta)$.
4. $A(\rho, \theta) \leftarrow A(\rho, \theta) + 1$.
5. insere p em $P(\rho, \theta)$.

Observe que as várias retas que passam por um mesmo pixel (x, y) viram senóides no espaço $\rho \times \theta$.

1.2 Contornos deformáveis

Contornos deformáveis são curvas paramétricas que se deslocam com o objetivo de minimizar uma função objetivo. O valor da função deve ser mínimo na borda do objeto a ser segmentado.

Entre os modelos de contornos deformáveis disponíveis na literatura, o modelo *Snakes* foi o primeiro proposto e é o mais citado.

1.2.1 Teoria

Uma *snake* é um contorno do tipo spline de continuidade controlada sob a influência de forças externas e forças internas. As forças internas atuam impondo suavidade ao contorno, enquanto as forças externas empurram a *snake* para características da imagem que se assemelhem a bordas ou segmentos de bordas. Pode ser definida no plano como um contorno paramétrico fechado composto de pontos, $\mathbf{v}(\mathbf{s}) = (\mathbf{x}(\mathbf{s}), \mathbf{y}(\mathbf{s}))$, onde $(\mathbf{x}, \mathbf{y}) \in \mathbf{R}^2$, $\mathbf{s} \in [0, 1]$ e $\mathbf{v}(\mathbf{0}) = \mathbf{v}(\mathbf{1})$. A equação geral da energia deste modelo é escrita em termos das Energias Interna e Externa:

$$E_{snake}(v(s)) = \int_0^1 E_{int}(v(s)) + E_{ext}(v(s))\, ds \tag{220}$$

onde,

$$E_{int}(v(s)) = \frac{\alpha(s)\|\frac{\partial v}{\partial s}\|^2 + \beta(s)\|\frac{\partial^2 v}{\partial s^2}\|^2}{2} \tag{221}$$

e

$$E_{ext}(v(s)) = E_{image}(v(s)). \tag{222}$$

O objetivo é encontrar a solução que minimiza $\mathbf{E}_{snake}$. O termo $\mathbf{E}_{int}$ representa a Energia Interna do contorno cujo diferencial gera a Força Interna que impõe regularidade ao contorno.

Os termos $\frac{\partial v}{\partial s}$ e $\frac{\partial^2 v}{\partial s^2}$ representam as derivadas de primeira e segunda ordem, respectivamente. O coeficiente $\alpha(s)$ é o termo que controla a "tensão" entre os pontos do contorno, enquanto $\beta(s)$ é responsável pela "elasticidade" do contorno. A Energia Externa é calculada com base no gradiente da imagem, para que a *snake* seja atraída para pontos de borda. A Força Externa que executa esta ação pode ser escrita como:

$$F_{ext}(v(s)) = -\|\nabla(G_\sigma * I(v(s)))\|^2 \tag{223}$$

onde ∇ representa o operador gradiente, e $\mathbf{G}_\sigma * \mathbf{I}(\mathbf{v}(\mathbf{s}))$ denota a imagem trabalhada por um filtro de suavização gaussiano com desvio padrão σ.

A força interna, quando atua sozinha, pode levar a curva a tornar-se um único ponto, e por isto deve ser balanceada pela força externa, resultante do diferencial da Energia Externa.

1.2.2 Discretização

Com base em (220), (221) e (223), deve-se achar o contorno $v(\mathbf{s})$ que minimiza o funcional de energia. Para resolver este problema, é usado o cálculo variacional para derivar um sistema dinâmico (uma equação diferencial parcial) e chegar no estado de energia mínima no equilíbrio, usando programação dinâmica para achar a posição da *snake* que minimiza a energia discretizada. Para facilitar a solução, os coeficientes $\alpha(s)$ e $\beta(s)$ são considerados constantes.

O cálculo do extremo do funcional

$$E_{snake}(v(s)) = \int_0^1 \frac{\alpha\|\frac{\partial v}{\partial s}\|^2 + \beta\|\frac{\partial^2 v}{\partial s^2}\|^2}{2} + F_{ext}(v(s))\, ds \tag{224}$$

resulta na equação de Euler-Lagrange

$$F_{ext}(v(s)) - \frac{\partial}{\partial s}\alpha\frac{\partial v}{\partial s} + \frac{\partial^2}{\partial s^2}\beta\frac{\partial^2 v}{\partial s^2} = 0 \tag{225}$$

que é igual a

$$\alpha\frac{\partial^2 v}{\partial s^2} - \beta\frac{\partial^4 v}{\partial s^4} - F_{ext}(v(s)) = 0 \tag{226}$$

A curva que minimiza o funcional de energia deve satisfazer a equação de Euler-Lagrange (226). Para tornar a solução dinâmica, o contorno $\mathbf{v}$ é tratado como uma função de $\mathbf{s}$ e tempo $\mathbf{t}$:

$$\frac{\partial v(s,t)}{\partial t} = \alpha\frac{\partial^2 v(s,t)}{\partial s^2} - \beta\frac{\partial^4 v(s,t)}{\partial s^4} - F_{ext}(v(s,t)) \tag{227}$$

Quando a *snake* estabiliza, $\frac{\partial v(s,t)}{\partial t} = 0$ e temos a solução de (226).

Para a discretização, o método de diferenças finitas é usado, sendo h um passo pequeno ou diferença de s, e

- $v_i = v(ih) = (x(ih), y(ih))$
- $\mathbf{F_{ext}} = (F_x, F_y)$

Aplicando as diferenças finitas para frente e para trás alternadamente,

$$\begin{aligned} f'(x) &= f(x+1) - f(x) \\ f'(x) &= f(x) - f(x-1) \end{aligned} \tag{228}$$

a equação (226) pode ser discretizada em:

$$\begin{aligned} &\alpha(v_i - v_{i-1}) - \alpha(v_{i+1} - v_i) \\ &\quad +\beta(v_{i-2} - 2v_{i-1} + v_i) - 2\beta(v_{i-1} - 2v_i + v_{i+1}) \\ &\qquad +\beta(v_i - 2v_{i+1} + v_{i+2}) + \mathbf{F_{ext}} = 0 \end{aligned} \tag{229}$$

Esta equação pode ser escrita na forma matricial:

$$\begin{aligned} \mathbf{A}x + \mathbf{F}_x &= 0 \\ \mathbf{A}y + \mathbf{F}_y &= 0 \end{aligned} \tag{230}$$

sendo

$$\mathbf{A} = \begin{pmatrix} A & B & C & 0 & 0 & 0 & \cdots & B \\ B & A & B & C & 0 & 0 & \cdots & C \\ C & B & A & B & C & 0 & \cdots & 0 \\ 0 & C & B & A & B & C & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & A & B & C \\ C & 0 & 0 & 0 & \cdots & B & A & B \\ B & C & 0 & 0 & \cdots & C & B & A \end{pmatrix} \tag{231}$$

onde

$$\begin{aligned} A &= 2\alpha + 6\beta \\ B &= -\alpha - 4\beta \\ C &= \beta \end{aligned} \tag{232}$$

A equação (230) pode ser resolvida iterativamente da seguinte forma (assumindo que $\mathbf{F}$ não muda em um passo de tempo):

$$\mathbf{A}x(t) + \mathbf{F}_x(t-1) = -\gamma(x(t) - x(t-1)) \tag{233}$$

$$\mathbf{A}y(t) + \mathbf{F}_y(t-1) = -\gamma(y(t) - y(t-1)) \tag{234}$$

onde γ é um intervalo de tempo pequeno. Após rearranjar os termos, temos:

$$\begin{aligned} x(t) &= (\mathbf{A} + \gamma\mathbf{I})^{-1}(\gamma x(t-1) - \mathbf{F}_x(t-1)) \\ y(t) &= (\mathbf{A} + \gamma\mathbf{I})^{-1}(\gamma y(t-1) - \mathbf{F}_y(t-1)). \end{aligned} \tag{235}$$

Usando as equações acima, $x(t)$ e $y(t)$ podem ser calculados iterativamente através de $x(t-1)$, $y(t-1)$, $\mathbf{F}_x(t-1)$, e $\mathbf{F}_y(t-1)$.

1.3 Algumas dificuldades de implementação

O modelo apresentado para contornos deformáveis apresenta algumas dificuldades de implementação.

1. Inicialização do contorno.

   O contorno inicializado em regiões homogêneas de brilho tende a se contrair, portanto o contorno deve ser inicializado fora do objeto de interesse. Mesmo assim, caso o contorno seja inicializado longe da borda de interesse, ele pode ficar preso em mínimos locais.

2. Estabilidade e convergência.

   O contorno pode oscilar em torno de bordas "fracas". A escolha dos parâmetros α e β é fundamental. Contornos com muita rigidez não aderem a identações e protusões da borda e contornos muito flexíveis tendem a passar por bordas "fracas".

3. Auto-cruzamento.

   Durante o movimento, o contorno pode formar "laços" cruzando a si próprio. Isto requer uma reamostragem dos pontos eliminando os laços.

4. Mudança de topologia.

   O modelo apresentado só trata o caso de uma única borda de interesse por vez.

A literatura de contornos deformáveis é extensa e consiste basicamente de trabalhos que procuram resolver as dificuldades supracitadas. Neste aspecto, métodos de segmentação baseados em região são mais simples de serem implementados.