Processamento de Imagens usando Grafos

Prof. Alexandre Xavier Falcão

Segundo semestre de 2004

1 Transformada de watershed de marcadores rotulados

Uma abordagem para segmentação de k objetos em uma imagem $\hat{I} = (D_I, I)$ é assumir um conjunto $S \subset D_I$ de sementes, um rótulo $1, 2, \ldots, k$ para cada objeto, e um rótulo 0 para o fundo, de tal forma que todos os objetos (incluindo o fundo) possuem ao menos uma semente em S. Para facilitar, podemos assumir uma imagem inicial $\hat{L} = (D_I, L)$ onde $L(p) = -1$, se $p \notin S$, e $L(p)$ é igual ao rótulo de fundo ou objeto correspondente, se $p \in S$. A segmentação funcionará melhor em imagens de diferenças (do tipo magnitude de gradiente), onde as bordas dos objetos de interesse estão realçadas (mais claras) em relação a outras regiões da imagem. Vamos adotar o modelo mais geral $f_{\max}$, em vez de f_{peak}. A idéia é obter o resultado em $\hat{L}$ propagando os rótulos em L de tal forma que cada semente defina uma zona de influência composta pelos pixels mais conexos com esta semente, de acordo com $f_{\max}$, do que com qualquer outra.

A escolha da melhor relação de adjacência A vai depender da aplicação. Normalmente, vizinhança-8 é suficiente. Podemos adotar política FIFO, $h(q) = 0$, e diferentes funções de dissimilaridade $\delta(p, q)$.

$$\delta_1(p,q) = K(1 - \max_{\forall s \in S}\{\alpha(p,q)\}), \text{ onde} \quad (1)$$

$$\alpha(p,q) = \exp^{-\frac{(\frac{I(p)+I(q)}{2} - I(s))^2}{2\sigma^2}}.$$

$$\delta_2(p,q) = G(q), \quad (2)$$

onde K é um inteiro (e.g. máximo brilho em $\hat{I}$), σ é uma constante (e.g. $20 * K$), $G(q)$ pode ser a magnitude de um vetor gradiente (e.g. Sobel) ou um gradiente morfológico calculado no pixel q em uma imagem de afinidades $\hat{J} = (D_I, J)$ definida por:

$$J(q) = K \max_{\forall s \in S \setminus F}\{\exp^{-\frac{(I(q)-I(s))^2}{2\sigma^2}}\}, \quad (3)$$

onde F são sementes de fundo. Note que as funções $f_{\max}$ com $\delta_1(p, q)$ (grafo não-orientado) e $f_{\max}$ com $\delta_2(p, q)$ (grafo orientado) são suaves. Elas também diferem na resolução da imagem de diferenças, onde a primeira apresenta maior vantagem. Pois, a IFT usando $f_{\max}$ com $\delta_2(p, q)$ não inunda regiões com 1 pixel de espessura. Por outro lado, o algoritmo fica mais simples,

eficiente e não requer atualizações na fila de prioridades, pois todos os pixels são atingidos primeiro por um caminho ótimo.

Transformada de watershed de marcadores rotulados usando $f_{\max}$ com δ_1:

Entrada: Imagens $\hat{I} = (D_I, I)$, $\hat{L} = (D_I, L)$, e adjacência A.
Saída: Imagem $\hat{L} = (D_I, L)$ com rótulos propagados.
Auxiliares: Fila Q de prioridades com política FIFO, imagem $\hat{C} = (D_I, C)$ de custos, lista S de sementes e variável tmp.

1. Para todo pixel $p \in D_I$ faça
2. Se $L(p) = -1$ então $C(p) \leftarrow +\infty$,
3. Se não $C(p) \leftarrow 0$ e insira p em Q e em S.
4. Enquanto $Q \neq \emptyset$ faça
5. Remova um pixel p de Q cujo custo $C(p)$ é mínimo.
6. Para todo $q \in A(p)$, tal que $C(q) > C(p)$, faça
7. $tmp \leftarrow \max\{C(p), \delta_1(p, q)\}$, onde $\delta_1(p, q)$ requer S.
8. Se $tmp < C(q)$ faça
9. Se $C(q) \neq +\infty$, remova q de Q.
10. $C(q) \leftarrow tmp$, $L(q) \leftarrow L(p)$, e insira q em Q.

Transformada de watershed de marcadores rotulados usando $f_{\max}$ com δ_2:

Entrada: Imagens $\hat{G} = (D_I, G)$, $\hat{L} = (D_I, L)$, e adjacência A.
Saída: Imagem $\hat{L} = (D_I, L)$ com rótulos propagados.
Auxiliares: Fila Q de prioridades com política FIFO, imagem $\hat{C} = (D_I, C)$ de custos e variável tmp.

1. Para todo pixel $p \in D_I$ faça
2. Se $L(p) = -1$ então $C(p) \leftarrow +\infty$,
3. Se não $C(p) \leftarrow 0$ e insira p em Q.
4. Enquanto $Q \neq \emptyset$ faça
5. Remova um pixel p de Q cujo custo $C(p)$ é mínimo.

6. Para todo $q \in A(p)$, tal que $C(q) > C(p)$, faça

7. $tmp \leftarrow \max\{C(p), G(q)\}$.

8. Se $tmp < C(q)$ faça

9. $C(q) \leftarrow tmp$, $L(q) \leftarrow L(p)$, e insira q em Q.

2 Exercício

Muito embora a política FIFO tenda a dividir platôs de custo ótimo igualmente entre as raízes que os atingem, isso pode não acontecer em algumas situações. Modifique os algoritmos acima para que quando um pixel p, removido de Q, encontrar um vizinho q com custo $tmp = C(q)$, o desempate entre as respectivas raízes leve em conta as distâncias Euclideanas entre q e a raiz de p, e entre q e sua raiz atual.