# Uma Abordagem para Registro de Imagens

Prof. Alexandre Xavier Falcão

Aula 05 - Parte IV

## 1 Introdução

Em um contexto mais geral, sejam $\hat{I} = (D_I, \vec{I})$ (fixa) e $\hat{J} = (D_J, \vec{J})$ (móvel) duas imagens multi-dimensionais e multi-espectrais. Sejam $S_I \subset D_I$ e $S_J \subset D_J$ pontos dessas imagens tais que $S_I = MS_J$. Considere ainda que as características de imagem $\vec{I}$ e $\vec{J}$ podem ser diretamente usadas ou ainda serem transformadas por filtragem em novos vetores de características. O registro automático entre $\hat{I}$ e $\hat{J}$ pode ser dividido em três etapas: (i) Extração de pontos $S_J \subset D_J$, visto que $S_I = MS_J$, e de características de imagem para o *matching* entre esses pontos. Para simplificar, podemos assumir $\vec{I}$ e $\vec{J}$ nos pontos de $S_I$ e $S_J$, respectivamente. (ii) Estratégia de busca da transformação de *matching* $M$. (iii) Uma função critério $F(M, S_J, \hat{I}, \hat{J})$ usada na busca da transformação $M$. Para cada candidato $M$ durante a busca, avalia-se o valor de $F(M, S_J, \hat{I}, \hat{J})$. O registro pára quando $F(M, S_J, \hat{I}, \hat{J})$ é mínimo (ou máximo). Aplica-se $D_J = M^{-1}D_I$ para obter os valores $\vec{J}(p)$ para todo $p \in D_I$ por interpolação dos valores $\vec{J}(q)$ dos vizinhos $q$ de $M^{-1}p$ em $D_J$, gerando a imagem registrada $\hat{R} = (D_I, \vec{J})$.

A abordagem descrita nesta aula é baseada na técnica proposta na dissertação de mestrado abaixo, onde uma das principais contribuições é um algoritmo de busca por descendente de gradiente multi-escala.

Fernanda Favretto, Registro de imagens 3D do cérebro humano. IC-UNICAMP, CNPq, FAPESP07/53608-5, orientador: A.X. Falcão, Mar 2009.

## 2 Características de imagem

Filtros de realce de bordas (veremos mais adiante) e/ou os próprios valores das bandas espectrais em $\vec{I}$ e $\vec{J}$ podem ser usados como características de imagem. O registro com $S_J = D_J$ pode ser muito ineficiente, portanto a idéia é reduzir a avaliação de $F(M, S_J, \vec{I}, \vec{J})$ para poucos pontos em $S_J \subset D_J$. O conjunto $S_J$ pode ser composto por pixels de borda entre regiões homogêneas da imagem, pixels de mínimo (ou máximo) de brilho em várias escalas da imagem, pixels de um objeto segmentado na imagem, pixels de uma região de interesse contendo um objeto da imagem, etc.

# 3 Funções critério

Se esperamos que a imagem registrada $\hat{R} = (D_I, \vec{J})$ seja similar à imagem $\hat{I} = (D_I, \vec{I})$, então podemos usar como função critério o erro quadrático médio, que deve ser minimizado.

$$F(M, S_J, \hat{I}, \hat{J}) = \frac{1}{\mid S_J \mid} \sum_{\forall (p,q) \mid q \in S_J, p = Mq} \| \vec{I}(p) - \vec{J}(q) \|^2.$$

No caso particular, onde $\vec{I} = I$ e $\vec{J} = J$, podemos esperar alta dependência estatística entre $(S_I, I)$ e $(S_J, J)$, $S_I = MS_J$. Sejam, portanto, $L_1 = \max_{\forall p \in S_I}\{I(p)\}$ e $L_2 = \max_{\forall q \in S_J}\{J(q)\}$, $h_1(l_1)$ o histograma da subimagem $(S_I, I)$, $h_2(l_2)$ o histograma da subimagem $(S_J, J)$, e $h_{12}(l_1, l_2)$ o histograma conjunto das subimagens $(S_I, I)$ e $(S_J, J)$. A função critério pode ser a informação mútua, que deve ser maximizada.

$$F(M, S_J, \hat{I}, \hat{J}) = \sum_{\forall l_1 \in [0, L_1]} \sum_{\forall l_2 \in [0, L_2]} h_{12}(l_1, l_2) \log \frac{h_{12}(l_1, l_2)}{h_1(l_1) h_2(l_2)}.$$

Outra opção seria maximizar a correlação entre os histogramas. A função de correlação será vista mais adiante. Note que no caso multi-espectral, podemos maximizar a soma das informações mútuas entre as bandas correspondentes.

# 4 Busca por descendente de gradiente multi-escala

A função $F(M, S_J, \hat{I}, \hat{J})$ varia na verdade com os parâmetros da matriz $M$. Seja, portanto, $\hat{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$ um vetor com $n$ parâmetros e $F(\hat{\theta}(t))$ o valor da função critério para uma dada instância $\hat{\theta}(t)$ deste vetor de parâmetros. Sabemos que o gradiente descendente parte de um vetor inicial e atualiza este vetor a cada iteração, seguindo a direção descendente do gradiente $\nabla F(\hat{\theta})$ com passo $\eta > 0$ pequeno.

$$\hat{\theta}(t) = \hat{\theta}(t-1) - \eta \nabla F(\hat{\theta}) \tag{1}$$

O cálculo de $\nabla F(\hat{\theta})$ pode ser feito por aproximação discreta, quando não conhecemos a forma de $F$:

$$\nabla F(\hat{\theta}) = \left( \frac{\delta F}{\delta \theta_1}, \frac{\delta F}{\delta \theta_2}, \ldots, \frac{\delta F}{\delta \theta_n} \right) \tag{2}$$

$$\frac{\delta F}{\delta \theta_i} = \frac{F(\theta_1, \ldots, \theta_i + \Delta_i, \ldots, \theta_n) - F(\theta_1, \ldots, \theta_i - \Delta_i, \ldots, \theta_n)}{2\Delta_i} \tag{3}$$

onde $\Delta_i > 0$. Outra forma de caminhar na direção descendente do gradiente é obter para cada parâmetro $\theta_i$, $i = 1, 2, \ldots, n$, o deslocamento $\Delta_i^* \in \{\Delta_i, 0, -\Delta_i\}$ que corresponde ao valor mínimo (ou máximo) de $F(\theta_1, \ldots, \theta_i + \Delta_i^*, \ldots, \theta_n)$, e se deslocar seguindo a regra:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + (\Delta_1^*, \Delta_2^*, \ldots, \Delta_n^*) \tag{4}$$

Uma desvantagem deste método é a possibilidade do gradiente descendente ficar preso em um ótimo local não satisfatório. Uma forma de evitar este problema é uma variação do método, que adiciona a técnica de busca pelo gradiente descendente em múltiplas escalas do espaço de parâmetros.

Considere agora o vetor de parâmetros

$$\hat{\theta}(t-1) = (\theta_1(t-1), \theta_2(t-1), \ldots, \theta_n(t-1))$$

no instante $t-1$ e o conjunto de $m$ escalas de valores para $\Delta_i(j) > 0$, onde $j = 1, 2, \ldots, m$, para cada parâmetro $\theta_i$.

Podemos encontrar o deslocamento de cada escala

$$\Delta^*(j) = (\Delta_1^*(j), \Delta_2^*(j), \ldots, \Delta_n^*(j))$$

onde $\Delta_i^*(j) \in \{-\Delta_i(j), 0, \Delta_i(j)\}$ tal que

$$\begin{aligned} & F(\theta_1(t-1), \ldots, \theta_i(t-1) + \Delta_i^*(j), \ldots, \theta_n(t-1)) = \\ \min_{i=1,2,\ldots,n} \{ & F(\theta_1(t-1), \ldots, \theta_i(t-1) - \Delta_i(j), \ldots, \theta_n(t-1)); \\ & F(\theta_1(t-1), \ldots, \theta_i(t-1), \ldots, \theta_n(t-1)); \\ & F(\theta_1(t-1), \ldots, \theta_i(t-1) + \Delta_i(j), \ldots, \theta_n(t-1))\} \end{aligned} \tag{5}$$

e então escolher aquele

$$\hat{\Delta}^* \in \{\hat{\Delta}^*(1), \hat{\Delta}^*(2), \ldots, \hat{\Delta}^*(m)\}$$

que minimiza a função $F$, e se deslocar seguindo a regra:

$$\hat{\theta}(t) \;=\; \hat{\theta}(t-1) + \hat{\Delta}^* \mid F(\hat{\theta}(t-1) + \hat{\Delta}^*) = \min_{j=1,2,\ldots,m} \{F(\hat{\theta}(t-1) + \hat{\Delta}^*(j))\} \tag{6}$$

Por exemplo, caso se busque a direção de deslocamento do parâmetro 3 com escala 2 que minimize $F$, deve-se calcular $F$ nas direções positiva ($d_{+1}$), negativa ($d_{-1}$) e sem deslocamento ($d_0$), onde

$$\begin{aligned} d_{-1} &= F(\theta_1(t-1), \theta_2(t-1), \theta_3(t-1) - \Delta_3(2), \ldots, \theta_{n-1}(t-1), \theta_n(t-1)) \\ d_0 &= F(\theta_1(t-1), \theta_2(t-1), \theta_3(t-1), \ldots, \theta_{n-1}(t-1), \theta_n(t-1)) \\ d_{+1} &= F(\theta_1(t-1), \theta_2(t-1), \theta_3(t-1) + \Delta_3(2), \ldots, \theta_{n-1}(t-1), \theta_n(t-1)) \end{aligned}$$

e escolher a direção $\{-1, 0, +1\}$ que minimize $F$.

O algoritmo 1 formaliza a estratégia de busca Descendente de Gradiente em Múltiplas Escalas (MSGD de *Multi-Scale Gradient Descent*) que minimiza uma dada função $F$, sendo trivial sua modificação para a maximização de $F$.

Os parâmetros de entrada são o vetor inicial de parâmetros $\theta[i], i = 1, 2, \ldots, n$ e a matriz de deslocamentos $\Delta[j][i]$, $j = 1, 2, \ldots, m$ e $i = 1, 2, \ldots, n$ por parâmetro para cada escala. A função $F$ será implementada a parte, visto que é específica de cada problema. No algoritmo, eu vou apenas indicar o que ela deve calcular. A saída do algoritmo será o vetor $\theta^*[i]$, $i = 1, 2, \ldots, n$, que contém os parâmetros que minimizam o valor da função $F$.

Os comentários inserido no Algoritmo 1 fazem referência à definição matemática dada no início desta seção.

**Algorithm 1** – ALGORITMO DE BUSCA MSGD

INPUT: Vetor $\theta$ de parâmetros e matriz $\Delta$ de deslocamentos.
OUTPUT: Vetor $\theta^*$ com os parâmetros ótimos.
AUXILIARY: $\Delta^*$,$V_F^*$,$V_F$, $V_{F0}$, $V_{F1}$ e $V_{F2}$

1. $V_F^* \leftarrow F(\theta[1], \ldots, \theta[n])$ //$VF^* \leftarrow F(\hat{\theta}(t-1))$
2. **For** $i \leftarrow 1..n$ **do** $\theta^*[i] \leftarrow \theta[i]$ //$\hat{\theta^*}(t-1) \leftarrow \hat{\theta}(t-1)$
3. **do**
4. $\quad V_{F0} \leftarrow V_F^*$ // *valor inicial da iteração* $t-1$
5. $\quad$ **For** $i \leftarrow 1..n$ **do** $\theta[i] \leftarrow \theta^*[i]$ // $\hat{\theta}(t-1) \leftarrow \hat{\theta^*}(t-1)$
6. $\quad$ **For** $j \leftarrow 1..m$ **do**
7. $\quad\quad$ **For** $i \leftarrow 1..n$ **do**
8. $\quad\quad\quad V_F \leftarrow V_{F0}$ *e* $\Delta^*[i] \leftarrow 0$ // $\Delta_i^*(j) \leftarrow 0$
9. $\quad\quad\quad V_{F1} \leftarrow F(\theta[1], \ldots, \theta[i] + \Delta[j][i], \ldots, \theta[n])$
10. $\quad\quad\quad V_{F2} \leftarrow F(\theta[1], \ldots, \theta[i] - \Delta[j][i], \ldots, \theta[n])$
11. $\quad\quad\quad$ **If** $V_{F1} < V_F$ **then** $V_F \leftarrow V_{F1}$ *e* $\Delta^*[i] \leftarrow \Delta[j][i]$
12. $\quad\quad\quad$ **If** $V_{F2} < V_F$ **then** $\Delta^*[i] \leftarrow -\Delta[j][i]$
13. $\quad\quad V_F \leftarrow F(\theta[1] + \Delta^*[1], \ldots, \theta[n] + \Delta^*[n])$ // $F(\hat{\theta}(t-1) + \hat{\Delta}^*(j)$
14. $\quad\quad$ **If** $V_F < V_F^*$ **then**
15. $\quad\quad\quad V_F^* \leftarrow V_F$ //$F(\hat{\theta}(t-1) + \hat{\Delta}^*(j)$
16. $\quad\quad\quad$ **For** $i \leftarrow 1..n$ $\theta^*[i] \leftarrow \theta[i] + \Delta^*[i]$ //$\hat{\theta^*}(t) \leftarrow \hat{\theta}(t-1) + \hat{\Delta}^*(j)$
17. $\quad$ **While** $V_F^* < V_{F0}$

A técnica MSGD pode ser aplicada a vários problemas que requerem otimização, sendo necessária a instanciação dos parâmetros de entrada de acordo com o problema que se deseja solucionar.

Suponha por exemplo o registro entre imagens 2D usando o vetor $\hat{\theta} = \{\theta_1, \theta_2, \theta_3, \theta_4\}$ com os seguintes parâmetros da transformação $M$: $\theta_1$ é o ângulo de rotação em torno de $z$, $\theta_2$ é a translação em $x$, $\theta_3$ é a translação em $y$, e $\theta_4$ é um fator de escala $s_x = s_y$. Inicialmente $\hat{\theta} = \{0, 0, 0, 1\}$, o que corresponde à matriz identidade em $M$.

A matriz $\Delta_{m \times n}$ armazena as escalas dos vetores de gradiente que serão avaliadas para cada parâmetro. Um exemplo seria usar 3 escalas ($m = 3$): $\Delta = \begin{bmatrix} 10.0 & 10.0 & 10.0 & 0.10 \\ 5.0 & 5.0 & 5.0 & 0.05 \\ 1.0 & 1.0 & 1.0 & 0.01 \end{bmatrix}$ onde o índice da coluna, $i$, identifica o parâmetro $\theta_i$ e o índice da linha, $j$, identifica a escala. Por exemplo, o elemento $\Delta[1][2] = 10.0$ indica uma translação de 10 em $x$.

O algoritmo 2 define a função $F(S_j, M, \hat{I}, \hat{J})$ como erro quadrático médio, onde $M$ é substituído por quatro outros parâmetros usados para seu cálculo,

$$\theta, \Delta, i, dir,$$

cobrindo as possibilidades de cálculo de $F$ mostradas no algoritmo 1. O vetor $\theta$ de parâmetros; o vetor $\Delta$ de deslocamentos, que pode ser deslocamentos para cada parâmetro $i$ na escala $j$

(i.e., linha $\Delta[j]$ da matriz $\Delta[j][i]$) ou $\Delta^*$; o parâmetro $i \in \{0, 1, 2, \ldots, n\}$ que indica deslocamento apenas em $\theta[i]$, onde $i = 0$ indica que todos os parâmetros serão deslocados; e a direção $dir \in \{-1, 0, 1\}$ deste deslocamento, onde 0 indica sem deslocamento. O argumento $NULL$ é passado no lugar de $\Delta$ quando a matriz $M$ for calculada apenas com os valores em $\theta$, sem deslocamentos. Por exemplo, usamos $(\theta, \Delta[j], i, -1)$ para calcular $F(\theta[1], \ldots, \theta[i] - \Delta[j][i], \ldots, \theta[n])$, $(\theta, NULL, 0, 0)$ para calcular $F(\theta[1], \ldots, \theta[n])$ e $(\theta, \Delta^*, 0, 0)$ para calcular $F(\theta[1] + \Delta^*[1], \ldots, \theta[n] + \Delta^*[n])$ no algoritmo 1.

**Algorithm 2** – Função de Erro Quadrático Médio

| | |
|---|---|
| Input: | Conjunto de pontos $S_J$, vetor de parâmetros $\theta$, vetor de deslocamentos $\Delta$, parâmetro $i$, direção $dir$, e imagens $\hat{I}$ e $\hat{J}$ |
| Output: | Erro quadrático médio $E$ |
| Auxiliary: | Matriz $M$, vetor de parâmetros $\theta'$ após deslocamento |

1. **If** $\Delta = NULL$ **then** $M \leftarrow Transformacao(\theta)$
2. **Else**
3. ⎸ **If** $i = 0$ **then**
4. ⎸ ⎿ **For** $k \leftarrow 1..n$ **do** $\theta'[k] \leftarrow \theta[k] + \Delta[j][k]$
5. ⎸ **Else**
6. ⎸ ⎸ **For** $k \leftarrow 1..n$ **do** $\theta'[k] \leftarrow \theta[k]$
7. ⎸ ⎿ ⎸ $\theta'[i] \leftarrow \theta'[i] + dir \times \Delta[j][i]$
8. $M \leftarrow Transformacao(\theta')$
9. $E \leftarrow 0$
10. **For each** $q \in S_J$ **do** $p \leftarrow MS_j$ *e* $E \leftarrow E + \|\vec{I}(p) - \vec{J}(q)\|^2$
11. *Retorne* $E/ \mid S_J \mid$

A função $Transformacao(\theta)$ retorna a matriz resultante da rotação $\theta[1]$, translação $(\theta[2], \theta[3])$ e escala $\theta[4]$. Note, porém, que o mínimo de $F$ pode levar a uma variação de escala que reduz o tamanho do conjunto $S_I = MS_J$ ao caso trivial de um ou poucos pixels. Para evitar isso, podemos acrescentar restrições ao parâmetro $\theta[3]$. Por exemplo, $\theta[3] \geq 0.25$.