MAC5701 - Tópicos em Ciência da Computação

Métodos de Estimação de Orientação de Faces

Aluno: Regis de Abreu Barbosa
regisb@ime.usp.br

Orientador: Carlos Hitoshi Morimoto
hitoshi@ime.usp.br

24 de junho de 2004

Sumário

1 Introdução

A pesquisa em reconhecimento de faces tem se destacado dentre as áreas de pesquisa de Visão Computacional, encontrando aplicações em diversas áreas, tais como:

- realidade virtual: mundos virtuais interativos, jogos, animação, teleconferência;
- sistemas inteligentes de vigilância: sistemas de controle de acesso, estacionamentos, supermercados, lojas de departamento, trânsito;
- interfaces avançadas com usuário: interfaces sociais, tradução de linguagens de sinais, sinalização em ambientes com muito ruído (como aeroporto e empresas metalúrgicas);
- codificação baseada em modelo: compressão de vídeo.

Dentro da área de reconhecimento de faces, encontram-se diversos tópicos de estudo, tais como: detecção de faces, estimação de orientação, expressões faciais, dentre outros. Este estudo se concentra no problema de estimar a orientação tridimensional da face, uma importante tarefa para o desenvolvimento de novos sistemas de interação humano-computador.

Atualmente existem muitos sistemas de estimação de orientação da face, mas a grande maioria requer algum tipo de sensor conectado à cabeça do usuário. Os novos estudos em orientação de faces têm se concentrado na execução da tarefa sem necessidade do usuário "vestir" nenhum componente de hardware, permitindo uma interação bastante natural com sistemas computacionais.

Neste estudo, serão apresentados dois métodos distintos para estimação da orientação da face, o primeiro baseado em um modelo geométrico da face, e o segundo baseado em um modelo foto-realístico construído a partir de pontos marcados em imagens do objeto. Ao final, faremos uma comparação entre as duas abordagens, apresentando as vantagens e desvantagens de cada um.

2 Fundamentos

Ao estimarmos a *orientação tridimensional da face*, estamos calculando a orientação do sistema de coordenadas da face presente em uma imagem com relação ao sistema de coordenadas universal centrado na câmera.

No plano da imagem da câmera temos a projeção da imagem da face, que pode ser uma projeção perspectiva (ponto é projetado em direção ao centro de coordenadas) ou ortográfica (ponto é projetado perpendicularmente ao plano da imagem).

Existem duas abordagens principais para determinar o orientação 3-D da face. A primeira é a *abordagem baseada em modelo*, onde a orientação é determinada utilizando um modelo geométrico tridimensional da face. O modelo é construído a partir dos pontos de controle, usualmente localizados em características faciais como canto dos olhos e da boca, nariz e narinas. Os pontos do modelo são, então, alinhados aos pontos de características faciais encontrados na imagem da câmera, permitindo determinar a orientação do modelo na imagem. O algoritmo POS/POSIT apresentado neste estudo é um exemplo desta abordagem.

Outra abordagem é conhecida como *abordagem baseada em aparência*. Nesta, a orientação da face é determinada a partir da aparência do objeto "face" na imagem, sem utilizar nenhum modelo geométrico explicitamente. A estimação consiste em determinar qual orientação tridimensional poderia resultar na aparência bidimensional observada. O método proposto por Cootes *et al.* [12], utilizando um modelo chamado *Active Appearance Model (AAM)* foi inicialmente proposto para ajuste de imagens a partir de modelos bidimensionais. Este modelo tem sido estendido permitindo a extração de um modelo tridimensional e, consequentemente, a orientação tridimensional da face a partir de AAM. Este é um exemplo de abordagem baseada em aparência e será também apresentada neste estudo.

3 POS e POSIT

Embora a face não seja um objeto rígido, o método proposto por DeMenthon e Davis [2] para determinar a orientação de um objeto rígido numa imagem tem sido também utilizado em algoritmos de estimação de orientação da face ([1]).

Este método, baseado em técnicas de álgebra linear, é composto por dois algoritmos, POS e POSIT. Ele assume um modelo do objeto e a detecção de pelo menos 4 pontos não-coplanares do objeto na imagem. O modelo do objeto consiste de um conjunto de pontos que determinam a localização de pontos de características do objeto identificáveis na imagem. Dentre os métodos de detecção de características faciais, temos a transformada de Karhunen-Loève, também chamada Análise de Componentes Principais (PCA), proposta por Turk e Pentland [10].

O primeiro algoritmo, conhecido como POS (Pose from Orthography and Scaling), faz uma estimação aproximada da posição do objeto na imagem utilizando produtos de uma matriz por dois vetores. A matriz contém informações de pontos do modelo do objeto, e os vetores contém informações das coordenadas destes pontos na imagem. Como resultado, temos dois vetores que, normalizados, constituem as primeiras linhas da matriz de rotação, e suas normas correspondem ao fator de escala da projeção, que provê o vetor de translação.

O segundo algoritmo consiste das mesmas operações, aplicadas com os pontos da imagem obtidos no passo anterior. A cada iteração obtemos uma estimação mais precisa da orientação do objeto. Por isso, este segundo algoritmo é chamado POSIT (POS com ITerações).

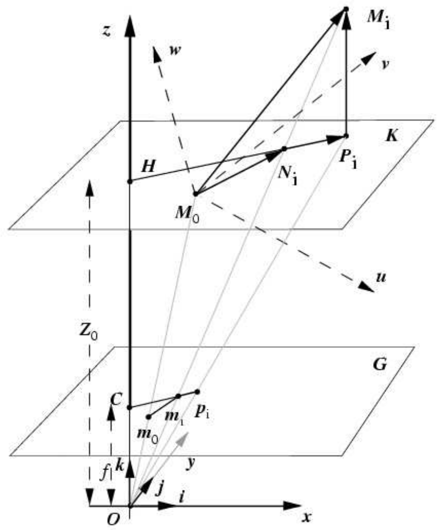

Figura 1: Sistema de Coordenadas da Câmera e Objeto

3.1 Notação

A câmera tem centro de projeção O e os seus eixos de coordenadas são representados pelos vetores unitários $\mathbf{i}$ e $\mathbf{j}$ correspondendo às linhas e colunas da imagem, e $\mathbf{k}$ correspondendo ao eixo óptico. A imagem está a uma distância f do centro de projeção O.

O objeto tem pontos de características $M_0, \ldots, M_i, \ldots, M_n$. O sistema de coordenadas do objeto é centrado em M_0 e seus eixos são u, v, w. As coordenadas (U_i, V_i, W_i) do ponto M_i são conhecidas no sistema de coordenadas do objeto, e desconhecidas no sistema de coordenadas da câmera.

A projeção perspectiva dos pontos M_i no plano da imagem nos fornece os pontos m_i com coordenadas (x_i, y_i).

3.2 Definição do problema

O objetivo deste método é encontrar a matriz de rotação, que permitirá transformar vetores definidos no sistema de coordenadas do objeto em vetores definidos no sistema de coordenadas da câmera, e o vetor de translação do objeto.

Verificamos que o produto interno $< \mathbf{M_0M_i}, \mathbf{i} >$, ambos definidos no sistema de coordenadas do objeto, resulta na projeção do vetor $\mathbf{M_0M_i}$ sobre o vetor unitário $\mathbf{i}$. O mesmo se aplica à $\mathbf{j}$ e $\mathbf{k}$.

As linhas da matriz de rotação $\mathbf{R}$ consistem, então, dos vetores $\mathbf{i}$, $\mathbf{j}$ e $\mathbf{k}$ expressos no sistema de coordenadas do objeto.

$$\mathbf{R} = \begin{bmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{bmatrix} \quad (1)$$

Observe que o vetor $\mathbf{k}$ pode ser obtido do produto vetorial entre $\mathbf{i}$ e $\mathbf{j}$.

O vetor de translação $\mathbf{T}$ é o vetor entre as origens dos sistemas de coordenadas da câmera e do objeto, ou seja, $\mathbf{OM_0}$. Se $\mathbf{M_0}$ é um ponto visível na imagem, então temos as coordenadas de $\mathbf{m_0}$ e o vetor T pode ser escrito como:

$$\mathbf{T} = \mathbf{OM_0} = \frac{Z_0}{f}\mathbf{Om_0} \quad (2)$$

3.3 Projeção Perspectiva e Projeção Escalar Ortográfica

A projeção perspectiva dos pontos M_i no plano G da imagem nos fornece os pontos m_i com coordenadas:

$$x_i = \frac{f}{Z_i}X_i \qquad \text{e} \qquad y_i = \frac{f}{Z_i}Y_i. \quad (3)$$

A projeção ortográfica escalar (*SOP, Scaled Orthographic Projection*) consiste de uma aproximação para a projeção perspectiva, assumindo mesma profundidade Z_i para os diferentes pontos M_i, como por exemplo, a profundidade Z_0. Ao invés de m_i teríamos p_i com coordenadas:

$$x_i^{'} = \frac{f}{Z_i}X_i \qquad \text{e} \qquad y_i^{'} = \frac{f}{Z_i}Y_i. \quad (4)$$

O ponto M_0 tem as mesmas coordenadas em SOP e na projeção perspectiva. Na figura 1 observamos a construção geométrica de SOP. Seja o plano K passando por M_0 paralelo ao plano G. O ponto Mi é projetado em K por uma projeção ortográfica, resultando no ponto P_i, que, por sua vez, é projetado no plano G por uma projeção perspectiva, resultando em p_i. Temos, então, o vetor m_0p_i paralelo a M_0P_i.

Podemos reescrever as coordenadas de SOP como:

$$x_i^{'} = sX_0 + s(X_i - X_0) = x_0 + s(X_i - X_0) \quad (5)$$

$$y_i^{'} = y_0 + s(Y_i - Y_0) \quad (6)$$

onde $s = \frac{f}{Z_0}$ é o fator de escala.

Esta equação mostra a proporcionalidade entre m_0p_i e M_0P_i.

3.4 Equações Fundamentais

Para definir a orientação do objeto, precisamos calcular as coordenadas de $\mathbf{i}$ e $\mathbf{j}$ no sistema de coordenadas do objeto para determinar a matriz de rotação, e a coordenada Z_0 para determinar o vetor de translação. Faremos isto utilizando as coordenadas de M_0M_i e as coordenadas (x_i, y_i) dos pontos m_0 e m_i da seguinte maneira:

$$\begin{aligned}
\mathbf{M_0M_i}\frac{f}{Z_0}\mathbf{i} &= x_i(1+\varepsilon_i) - x_0, & (7)\\
\mathbf{M_0M_i}\frac{f}{Z_0}\mathbf{j} &= y_i(1+\varepsilon_i) - y_0 & (8)
\end{aligned}$$

$$\text{onde } \varepsilon_i = \frac{1}{Z_0}\mathbf{M_0M_i}.k$$

A prova destas equações está em [2]. Os termos $x_i(1+\varepsilon_i)$ e $y_i(1+\varepsilon_i)$ são, na verdade, as coordenadas $x_i^{'}$ e $y_i^{'}$ dos pontos p_i, como podemos abaixo:

$$\begin{aligned}
\mathbf{M_0M_i} &= \mathbf{M_0P_i} + \mathbf{P_iM_i} & (9)\\
&= \frac{Z_0}{f}\mathbf{m_0p_i} + \mathbf{P_iM_i} & (10)
\end{aligned}$$

Tomando o produto interno $< \mathbf{M_0M_i}, \mathbf{i} >$, do lado direito teremos a soma dos produtos internos $< \mathbf{m_0p_i}, \mathbf{i} >$ que resulta na coordenada x do vetor $\mathbf{m_0p_i}$, que é dada pela diferença $x_i^{'} - x_0$, e $< \mathbf{P_iM_i}, \mathbf{i} >$ que resulta em zero, pois os vetores são ortogonais entre si. Temos, então:

$$\begin{aligned}
\mathbf{M_0M_i}.\mathbf{i} &= \frac{Z_0}{f}\mathbf{m_0p_i}.\mathbf{i} + \mathbf{P_iM_i}.\mathbf{i} & (11)\\
&= \frac{Z_0}{f}(x_i^{'} - x_0) + 0; & (12)\\
\mathbf{M_0M_i}.\frac{f}{Z_0}\mathbf{i} &= x_i^{'} - x_0 & (13)
\end{aligned}$$

3.5 Algoritmo POS e POSIT

As equações fundamentais (7) podem ser reescritas como:

$$\begin{aligned}
\mathbf{M_0M_i}.\mathbf{I} &= x_i(1+\varepsilon_i) - x_0, & (14)\\
\mathbf{M_0M_i}.\mathbf{J} &= y_i(1+\varepsilon_i) - y_0 & (15)
\end{aligned}$$

$$\text{onde } \mathbf{I} = \frac{f}{Z_0}\mathbf{i} \text{ e } \mathbf{J} = \frac{f}{Z_0}\mathbf{j}$$

Se tivermos um valor para ε_i, então os únicos valores desconhecidos das equações acima são os valores de **I** e **J**. Ao calcularmos e normalizarmos estes vetores obtemos **i** e **j**, e a norma de **I** e **J** correspondem à terceira coordenada do vetor de translação, T_z.

Como o valor de ε_i é desconhecido, inicialmente atribuímos zero. Neste caso, estamos assumindo uma projeção ortográfica do objeto, e portanto obteremos somente uma aproximação de **i** e **j**. Estes cálculos correspondem ao algoritmo POS (*Pose from Orthography and Scaling*).

Esta aproximação nos permite computar valores mais exatos de ε_i, pois obtemos **k** a partir do produto vetorial de **i** e **j**, e assim, iterativamente, aplicamos o algoritmo POS obtendo sempre valores mais precisos de **i**, **j** e T_z. Este cálculo corresponde ao POSIT (POS com ITerações), já que consiste em iterações do POS, convergindo para um valor mais exato da orientação do objeto.

Nos experimentos feitos por DeMenthon e Davis [2], aproximadamente 5 iterações eram suficientes para se obter uma orientação precisa do objeto. Como podemos ver, entre as vantagens deste algoritmo estão a sua eficiência, pois requer poucas operações aritméticas, tornando-o viável para uso em tempo-real; a facilidade de implementação, utilizando poucas linhas de código; e o fato de não requerer uma estimação da orientação inicial.

O uso do POSIT para estimação da orientação de faces, porém, apresenta como desvantagem o fato deste algoritmo ser aplicado a objetos rígidos, enquanto que a face é um objeto não-rígido. Chang *et al.* [1] mostraram como este algoritmo pode ser estendido, possibilitando o seu uso na determinação de orientação de faces e expressões faciais. Os seus experimentos mostraram-se robusto e com alta porcentagem de convergência, provando sua viabilidade na estimação de orientação de faces.

4 Active Appearance Models

Nesta seção, apresentaremos uma abordagem baseado em aparência para interpretação de imagens. Este método, proposto por Cootes *et al.* [12] consiste na extração de um modelo chamado *Active Appearance Model*, ou AAM, a partir de imagens foto-realistas do objeto em análise, no caso, a face. O modelo inclui informações de forma e aparência.

A idéia de usar imagens foto-realistas para compor o modelo já foi utilizada por Turk e Pentland em Análise de Componentes Principais, PCA [10]. Embora PCA tenha se mostrado apropriado e eficiente na detecção de faces através das *eigenfaces*, este método apresenta problemas com variação de orientação e expressão da face. Veremos na próxima seção que AAM utiliza PCA, mas inclui no modelo mais informações que contribuem na detecção e extração de outras características do objeto presente na imagem.

AAM foi inicialmente proposto com o objetivo de alinhar modelos de face bidimensionais a uma imagem de face dada como entrada para o algoritmo. Por se tratar de um modelo bidimensional, este método a princípio não objetivava a estimação da orientação tridimensional da face. Um outro modelo, 3DMM

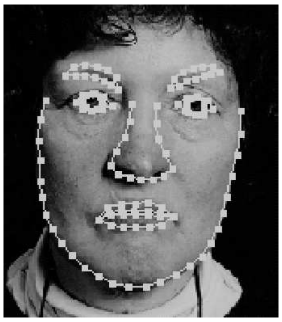

Figura 2: Exemplo de imagem marcada com 122 pontos de controle

(*3-Dimensional Morphable Model*), embora tridimensional, era lento e, consequentemente, inviável para sistemas de tempo real. Xiao *et al.* desenvolveram uma variante tridimensional do AAM, chamada *Combined 2D+3D Active Appearance Model* [15], que combinava a vantagem da tridimensionalidade de 3DMM com a eficiência do AAM. Esta variante possibilitou o uso do AAM na estimação de orientação de face e rastreamento do olhar [17].

4.1 Modelo de Forma e Aparência

Os modelos em AAM são uma combinação de um modelo de variação da forma com um modelo de variação da aparência. O conjunto de treinamento é composto por imagens com marcas de pontos de controle colocados em posições chaves do imagem, descrevendo as características principais da forma do objeto.

Com este conjunto de treinamento, podemos gerar um modelo estatístico de variação da forma. Os pontos de controle são alinhados e representados por um vetor $\mathbf{x}$ em cada imagem. Este vetor descreve a forma do objeto. Aplicando PCA a estes dados do conjunto de treinamento, podemos aproximar cada um dos dados por:

$$\mathbf{x} = \overline{\mathbf{x}} + \mathbf{P}_s \mathbf{b}_s \tag{16}$$

onde $\overline{\mathbf{x}}$ é a forma média, ou seja, média de todos os vetores $\mathbf{x}$ das imagens do conjunto de treinamento, $\mathbf{P}_s$ é um conjunto formado pelos auto-vetores ortogonais obtidos a partir dos dados $\mathbf{x}$ das imagens por PCA, e $\mathbf{b}_s$ é um conjunto de parâmetros da forma.

Para construir um modelo de variação da aparência em níveis de cinza, utiliza-se um algoritmo de triangulação, onde uma imagem é modificada de forma que seus pontos de controle coincidam com os pontos da forma média $\overline{\mathbf{x}}$.

Fazemos, então, uma amostra dos níveis de cinza, $\mathbf{g}_{im}$, da forma normalizada em toda a região coberta pela forma média. Para minimizar o efeito da variação de iluminação, as amostras são normalizadas por um fator de escala α e nível de cinza β:

$$\mathbf{g} = (\mathbf{g}_{im} - \beta\mathbf{1})/\alpha \tag{17}$$

Os valores de α e β são dados por $\alpha = \mathbf{g}_{im}.\overline{\mathbf{g}}$ e $\beta = (\mathbf{g}_{im}.\mathbf{1})/n$

Aplicando PCA aos dados normalizados, obtemos:

$$\mathbf{g} = \overline{\mathbf{g}} + \mathbf{P}_g\mathbf{b}_g \tag{18}$$

onde $\overline{\mathbf{g}}$ é vetor médio dos níveis de cinza normalizados, $\mathbf{P}_g$ é um conjunto formado pelos auto-vetores ortogonais obtidos a partir dos dados $\mathbf{g}$ das imagens por PCA, e $\mathbf{b}_g$ é um conjunto de parâmetros dos níveis de cinza.

Os vetores $\mathbf{b}_s$ e $\mathbf{b}_g$ caracterizam a forma e a aparência de cada imagem de exemplo. Dado que haja correlação entre a variação da forma e a variação dos níveis de cinza, podemos aplicar PCA a estes dados de forma conjunta.

Primeiramente, concatenamos os vetores $\mathbf{b}_s$ e $\mathbf{b}_g$ da seguinte maneira:

$$\mathbf{b} = \begin{pmatrix} \mathbf{W}_s\mathbf{b}_s \\ \mathbf{b}_g \end{pmatrix} = \begin{pmatrix} \mathbf{W}_s\mathbf{P}_s^T(\mathbf{x} - \overline{\mathbf{x}}) \\ \mathbf{P}_g^T(\mathbf{g} - \overline{\mathbf{g}}) \end{pmatrix} \tag{19}$$

onde $\mathbf{W}_s$ é uma matriz diagonal de pesos de cada parâmetro da forma, permitindo que comparemos unidades de distância em $\mathbf{b}_s$ com unidades de intensidade dadas pelos níveis de cinza em $\mathbf{b}_g$. Em [12] vemos como calcular esta matriz.

Em seguida, aplicamos PCA às amostras $\mathbf{b}$, obtendo:

$$\mathbf{b} = \mathbf{Q}\mathbf{c} \tag{20}$$

onde $\mathbf{Q}$ são os auto-vetores e $\mathbf{c}$ os parâmetros da aparência que controlam tanto a forma quanto os níveis de cinza do modelo.

A partir da linearidade do modelo, podemos expressar a forma e os níveis de cinza diretamente em função de $\mathbf{c}$:

$$\mathbf{x} = \overline{\mathbf{x}} + \mathbf{P}_s\mathbf{W}_s\mathbf{Q}_s\mathbf{c} \text{ e } \mathbf{g} = \overline{\mathbf{g}} + \mathbf{P}_g\mathbf{Q}_g\mathbf{c} \tag{21}$$

onde

$$\mathbf{Q} = \begin{pmatrix} \mathbf{Q}_s \\ \mathbf{Q}_g \end{pmatrix} \tag{22}$$

4.2 Ajuste do modelo

A interpretação da imagem em AAM é um problema de otimização no qual queremos minimizar a diferença entre uma nova imagem e uma imagem sintetizada pelo modelo de aparência. Em outras palavras, dado o vetor de diferença $\delta\mathbf{I} = \mathbf{I_i} - \mathbf{I_m}$, onde $\mathbf{I_i}$ é o vetor de níveis de cinza da imagem e $\mathbf{I_m}$ é o vetor de níveis de cinza dos valores atuais dos parâmetros do modelo, queremos minimizar a sua magnitude $\Delta = |\delta\mathbf{I}|^2$, variando os parâmetros $\mathbf{c}$ do modelo.

Queremos, então, encontrar uma relação entre $\delta\mathbf{I}$ e $\delta\mathbf{c}$, que é o erro nos parâmetros do modelo. O algoritmo para minimizar Δ é construído através desta relação, que, por simplificação, é assumido uma relação linear dada por:

$$\delta\mathbf{c} = \mathbf{A}\delta\mathbf{I} \tag{23}$$

Para modelar pequenos deslocamentos na posição bidimensional, escala e orientação, colocamos elementos extras no vetor $\delta\mathbf{c}$: (s_x, s_y, t_x, t_y), onde $s_x = s\cos(\theta)$, $s_y = s\sin(\theta)$, s é o fator de escala, θ é a orientação e (t_x, t_y) é a translação.

Seja um vetor $\mathbf{c}_0$ inicial correspondendo aos parâmetros do modelo de aparência de uma imagem dada, este será sucessivamente deslocado por $\delta\mathbf{c}$, obtendo novos parâmetros $\mathbf{c} = \mathbf{c}_0 + \delta\mathbf{c}$. Utilizando $\mathbf{c}$ em (21), obtemos os parâmetros da forma $\mathbf{x}$ e níveis de cinza $\mathbf{g}_m$. Utilizamos, então, $\mathbf{x}$ para "entortar" a imagem dada e obter uma amostra normalizada $\mathbf{g}_s$. A amostra do erro é dada, então, por $\delta\mathbf{g} = \mathbf{g}_s - \mathbf{g}_m$.

O algoritmo para obter $\mathbf{A}$ consiste em deslocar randomicamente os parâmetros do modelo e salvar os valores de $\delta\mathbf{c}$ e $\delta\mathbf{g}$. A partir destes, podemos aplicar regressão linear multivariada múltipla para obter a relação:

$$\delta\mathbf{c} = \mathbf{A}\delta\mathbf{g} \tag{24}$$

Para refinar o vetor de parâmetros do modelo $\mathbf{c}$, este processo é repetido iterativamente até se obter convergência. Teremos, então, um vetor de parâmetros do modelo $\mathbf{c}$ que aplicado ao modelo de aparência ativa dada em (21), obtém uma aproximação ótima da imagem de entrada.

4.3 Combined 2D+3D AAM

Xiao *et al.* [15] mostraram que é possível um modelo de forma bidimensional gerar o mesmo conjunto de instâncias de um modelo de forma tridimensional. Para isto é necessário que o modelo 2D tenha seis vezes mais vetores de forma que o modelo 3D. Isto faz com que este modelo 2D tenha maior poder de representação, gerando inclusive instâncias que não são válidas no modelo 3D.

Para computar a forma 3D de um AAM, é necessário não somente uma, mas uma sequência de imagens $I^t(\mathbf{u})$ com $t = 0, ..., N$ correspondendo ao rastreamento da face. O algoritmo é composto de três passos.

O primeiro passo consiste de computar uma forma tridimensional do AAM. Isto é feito utilizando Decomposição de Valor Singular (SVD), decompondo a matriz composta pelos pontos de controle das imagens em um produto de matrizes correspondendo aos parâmetros e a pontos de um modelo tridimensional.

O segundo passo é responsável por eliminar instâncias do AAM que não são válidas em um modelo tridimensional, forçando o AAM a mover-se de maneira consistente com um modelo de forma 3D.

O último passo consiste em ajustar um AAM às imagens com relação à forma, aparência e parâmetros de normalização, respeitando os limites dados pelo passo anterior. Os detalhes deste algoritmo podem ser vistos em [15].

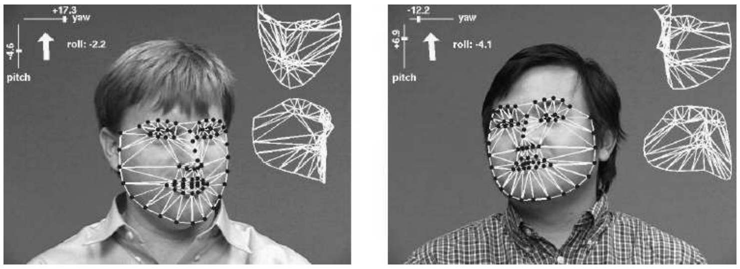

Figura 3: Resultados da estimação de orientação da face obtidos em [17]

Utilizando *Combined 2D+3D AAM*, Ishikawa *et al.* [17] descreveram um algoritmo para estimar a orientação da face. As informações da orientação são obtidas da matriz de parâmetros geradas pelo ajuste do modelo *Combined 2D+3D AAM*.

Um dos problemas gerados pela variação da orientação da face é a oclusão. O rastreamento de faces com oclusão geralmente falha quando se utiliza o algoritmo padrão do AAM. Os novos estudos em AAM tem procurado propor soluções para este problema [15], [16].

5 Conclusão

Neste estudo, foram apresentados dois métodos aplicados à estimação da orientação da face. No primeiro método, POS e POSIT, apresentamos um algoritmo eficiente baseado em um modelo geométrico rígido para estimação da orientação deste objeto. Embora a face não seja um objeto rígido, os resultados obtidos pelo uso de uma extensão deste algoritmo para estimação de orientação de faces [1] justificou nosso estudo, dado a eficiência, a facildade de implementação e o fato deste método não requerer nenhum pré-processamento para obtenção de uma estimativa inicial da orientação da face.

O segundo método apresentado, AAM, consistia de uma abordagem baseada em aparência, na qual sua aplicação inicial era o de ajuste bidimensional do modelo de aparência ativa em uma imagem de face dada. O poder de representação do AAM, porém, tem sido motivo de recentes estudos, sendo utilizado em diversas tarefas de visão computacional, tais como reconhecimento de face, estimação da orientação e reconhecimento de expressões faciais.

A vantagem do Modelo de Aparência Ativa está na representação de objetos deformáveis, como é o caso da face, e extensões deste modelo tem possibilitado a localização destes objetos mesmo com oclusão e variação na orientação, o que torna esta técnica promissora nas tarefas relacionadas ao reconhecimento de faces em visão computacional.

Referências

[1] Chin-Chun Chang and Wen-Hsiang Tsai. *Determination of Head Pose and Facial Expression from a Single Perspective View by Successive Scaled Ortographic Approximations*, International Journal of Computer Vision 46(3), pp. 179-199, 2002.

[2] Daniel DeMenthon and Larry Davis. *Model-Based Object Pose in 25 Lines of Code*, International Journal of Computer Vision, 15, pp. 123-141, 1995.

[3] S. H. Joseph. *Optimal Pose Estimation in Two and Three Dimensions*, Computer Vision and Image Understanding Vol. 73 (2), pp. 215-231, 1999.

[4] Volker Krüeger, Sven Bruns, Gerald Sommer. *Efficient Head Pose Estimation with Gabor Wavelet Networks*, Proc. British Machine Vision Conference, Bristol, UK, Sept. 12-14, 2000.

[5] Volker Krüeger. *Gabor Wavelet Networks for Object Representation*, Technical Report TR 4245, University of Maryland, April 2001.

[6] Trevor Darrell, Baback Moghaddam, Alex P. Pentland. *Active Face Tracking and Pose Estimation in an Interactive Room*, Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, 1996.

[7] D. Gavrila. *The Visual Analysis of Human Movement: A Survey*, Computer Vision and Image Understanding, 73(1), pp. 82-98, 1999.

[8] R. S. Feris, J. Gemmell, K. Toyama and V. Krueger. *Facial Feature Detection Using a Hierarchical Wavelet Face Database*, Microsoft Research Technical Report MSR-TR-2002-05, 2002.

[9] Lixin Fan and Kah Kay Sung. *Model-based varying pose face detection and facial feature registration in video images*, Proceedings of the eighth ACM international conference on Multimedia, pp. 295 - 302, 2000.

[10] M. Turk and A. Pentland, *Eigenfaces for recognition*, J. Cognitive Neuroscience, vol. 3, pp. 71-86, 1994.

[11] Carlos H. Morimoto and Myron Flickner. *Real-Time Multiple Face Detection Using Active Illumination*, Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000.

[12] T.F. Cootes, G.J. Edwards and C.J. Taylor. *Active Appearance Models*, In Proc. 5th European Conference on Computer Vision, Freiburg, Germany, 1998.

[13] G. J. Edwards, C. J. Taylor, and T. Cootes. *Face recognition using the active appearance model.* In H.Burkhardt and B. Neumann, editors, 5th European Conference on Computer Vision, volume 2, pages 581–695. Springer, 1998.

[14] Jochen Heinzmann and Alexander Zelinsky. *3-D Facial Pose and Gaze Point Estimation using a Robust Real-Time Tracking Paradigm*, Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, pages 142–147, 1998.

[15] Jing Xiao, Simon Baker, Iain Matthews and Takeo Kanade. *Real-Time Combined 2D+3D Active Appearance Models*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June, 2004.

[16] Ralph Gross, Iain Matthews and Simon Baker. *Constructing and Fitting Active Appearance Models with Occlusion*, Proceedings of the IEEE Workshop on Face Processing in Video, June, 2004.

[17] T. Ishikawa, S. Baker, I. Matthews, and T. Kanade. *Passive Driver Gaze Tracking with Active Appearance Models*, tech. report CMU-RI-TR-04-08, Robotics Institute, Carnegie Mellon University, February, 2004.

[18] I. Matthews and S. Baker. *Active Appearance Models Revisited*, tech. report CMU-RI-TR-03-02, Robotics Institute, Carnegie Mellon University, April, 2003.