Active Appearance Model Construction: Implementation notes

Nikzad Babaii Rizvandi*, Prof.Wilfried Philips, Dr.Aleksandra Pizurica

Image Processing and Interpretation Group (IPI)
TELIN Department, Gent University
St-Pietersnieuwstraat 41, B-9000 Gent, Belgium
** E-mail: nikzad@telin.UGent.be*
philips@telin.UGent.be
sanja@telin.UGent.be
http://telin.ugent.be/ipi

Active Appearance Model (AAM) is a powerful object modeling technique and one of the best available ones in computer vision and computer graphics. This approach is however quite complex and various parts of its implementation were addressed separately by different researchers in several recent works. In this paper, we present systematically a full implementation of the AAM model with pseudo codes for the crucial steps in the construction of this model.

Keywords: Shape Model, Texture Model, Active Appearance Model, Pseudo-Code Implementation.

1. Introduction

Model-based approaches to interpret variable-shape images have received a great deal of attention in recent years. These approaches analyze different variations of an object using a series of the object images in a training set and finally assign a model to the object variations. One of the famous Model-based methods is Active Appearance Model(AAM) of Cootes et al. in.[1] This model introduces a joint statistical model for the shape and gray-level texture appearance. An AAM model is constructed from a group of input images and some landmarks around the desired objects. These points include important features and regions such as fingers in a model of hand. To build the model every images is warped in order to collect each corresponding landmarks to the same position. The warped images, free from the final shape, are processed by Principle Component analysis(PCA). The fact the suitable features are corresponding together, independent from the shape, concludes the constructed model is better than the first model with-

out warping. In addition, gray values of images are sampled using the mean shape and delaunay triangles and construct a texture model using another PCA. Considering a weighting matrix, the built shape and texture models are combined together and produce the final model with another PCA. This model is used as a basis to learn a multi-variate regression matrix. From implementation aspect, AAM is a complex approach which needs several complex and time-consuming algorithms. The wrong implementation not only fails to reach to the suitable results but also wastes much time for running and debugging. Several recent works[1–6] address some parts of these algorithms separately but this paper aims to unit these implementation notes and present several pseudo codes for appearance model construction of the AAM implementation. Fig 1 shows the stages of the appearance model construction and their relationships.

2. Primary Shape Model

The first step in the appearance model construction is to form the shape model using the landmarks positions of several images in the training set. There are three steps to build the primary shape model which are:

2.1. *Procrustes analysis*

Procrustes analysis is a form of statistical shape analysis used to analyze a collection of shapes. In this case, the shapes are equalized by removing the translational, rotational and scaling components. Fig 2 shows a brief pseudo code to align shapes $X = (x_1, y_1, x_2, y_2, ..., x_n, y_n)$ and $Z = (z_1, w_1, z_2, w_2, ..., z_n, w_n)$.

2.2. *Shape Alignment*

The shapes of the objects in the training set are different in scaling, rotation and translation. In order to construct a stable shape model, these differences must be removed. The standard approach is to align all shapes to the mean shape and continue this procedure till the mean shape does not change in two consequent iterations (Fig 3).

2.3. *PCA and shape model order reduction*

PCA is a linear orthogonal transformation transmitting data set to a new coordinate system as the first greatest variance in data is assigned to the first dimension (or the first principle component), and so on. The principal

components are the eigenvectors of the covariance matrix. If Σ_s is the covariance matrix of the training shapes, we have $\Sigma_s * P_s = P_s * \Lambda_s$ where Λ_s denotes a diagonal matrix of eigenvalues

$$\Lambda_s = \begin{pmatrix} \lambda_{s,1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_{s,2N_t} \end{pmatrix} \tag{1}$$

corresponding to the eigenvectors in the column of P_s.
The major advantages of using PCA on a collection of data is to decrease the data set dimensions by keeping the low order principle components, corresponding to the greatest variances, and removing the high order components. To reduce the model parameters, the eigenvalues ($\lambda_{s,i}$) are sorted in a descending order. The first t modes of variations are elected to describe $SV \times 100$ of shape variations where t is the first eigenvalue satisfying the equation $\sum_{i=1}^{t} \lambda_{s,i} \geq SV \times \sum_{i=1}^{2n} \lambda_{s,i}$. A common value for SV is .95, so 95 percent of the shape variations are described by this model parameters. Fig 4 shows a pseudo code to implement a PCA and order reduction. If $\overline{X}$, $P_{s,r}$ and $X_{aligned}$ denote the mean shape, the reduced shape eigenvectors and the aligned shapes in the training set, respectively, then the shape model parameters will be : $b_s = P'_{s,r} * (X_{aligned} - \overline{X})$ where $P'_{s,r}$ is the transpose of $P_{s,r}$ and is equal to $P^{-1}_{s,r}$.

3. Primary Texture Model

A complete model of appearance not only includes the object shape specifications but also must involve the gray value of the object texture. The following describes the procedure to construct the texture model.

3.1. *Delaunay triangulation of the mean shape*

The first step to construct the texture model is determining which gray values must be used because only the pixels including the object are necessary. A suitable basis to make this is to utilize the position information of the landmarks. The standard solution is to divide the object into a combination of triangles by delaunay triangulation and then use the gray values inside these triangles.[3] The major problem is when the mean shape is not convex the delaunay triangles contain all inside and outside pixels.[7] So the texture includes some pixels of background that is not desirable.

3.2. *Gray-value Image Sampling*

This step samples the pixels inside the mean shape to obtain the corresponding pixels in the object textures and finds the related pixels in the textures of the training set by delaunay triangles. A brief algorithm description is in Fig 5.

3.3. *Texture alignment*

Within the object there are usually some variations in gray values because of different illumination intensities. Since the goal is to build a stable model without these unwanted effects, these variations must be eliminated. The common method is to align all textures to the standardized mean texture, with zero mean and unit variance, and continue this procedure till the differences between the standardized mean texture in two following iterations is less than a threshold. The complete procedure is explained in Fig 6.

3.4. *PCA and texture model order reduction*

The difference between this section and the section 2.3 is the way to calculate the covariance matrix. If N_g and N are the number of texture pixels and the number of shapes in the training set, so the covariance matrix will have $N_g * N_g$ dimensions. When $N_g \gg N$, calculating the covariance matrix, and therefore the texture eigenvectors and eigenvalues, is computationally expensive. The idea is to calculate the covariance between the textures and then convert it to the covariance between the pixels(Fig 7). After model reduction, the model parameters can be calculated by $b_g = P'_{g,r} * (G_{aligned} - \overline{G})$ where $\overline{G}$, $P_{g,r}$ and $G_{aligned}$ are the mean texture, the reduced texture eigenvectors and the aligned textures.

4. Combine Shape and Texture Models

Both b_s and b_g models should be merged to form an unit model including both texture and shape variabilities and keeping the correlation between them. Since the nature of shape and texture are different, some weighting are necessary. In the absence of these weighting, spread of the points in the space will be undesirable.[6] A simple weighting matrix is a diagonal matrix:

$$W = \omega I \tag{2}$$

where I is identity matrix, $w = \frac{\Sigma \tilde{\lambda_g}}{\Sigma \tilde{\lambda_s}}$ and $\tilde{\lambda_g}$ and $\tilde{\lambda_s}$ are eigenvalues of b_s and b_g, respectively. The merged model is a simple column vector as $b = (W * b_s, b_g)^T$.

4.1. *PCA and combined model order reduction*

To eliminate correlation between shape and texture parameters, another PCA should be performed on the combined data $b = P_c * c$ where P_c is the eigenvector matrix and c is the combined model parameters. The same as the section 2.3, the order of model is reduced after calculating the PCA.

5. Final Shape and Texture models

The final shape and texture models are calculated with the following equations:[1,4]

$$\begin{aligned} X &= \overline{X} + P'_s * W^{-1} * P_{c,s} * c' \\ G &= \overline{G} + P'_g * P_{c,g} * c' \end{aligned} \tag{3}$$

where c' is the reduced version of c and $P_c = (P_{c,s}, P_{c,g})^T$. These two equations are the basis to calculate the regression matrix in the next level.

6. Conclusion

The goal of this paper is to explain the implementation procedure of constructing shape and texture models in Active Appearance Model. For this purpose, necessary algorithms have been explained in details and described in several pseudo codes.

Primary Shape model:

1. Align shapes in the training set to the mean shape using Procrustes analysis.
2. Find the shape model by PCA analysis of aligned shapes.
3. Shape Model reduction.

Primary Texture model:

1. Find Delaunay triangles of the mean shape.
2. Remove Convex Hull from the Delaunay triangles.
3. Find textures of each object in the training set by sampling the mean shape.
4. Align each texture to the mean texture.
5. Find the texture model by PCA analysis of aligned textures.
6. Texture Model reduction.

Combined model:

1. Calculate a scaling matrix to mix shape and texture models.
2. Find the combined model by PCA analysis of the combined data.
3. Texture Model reduction.

Final Shape and Texture models:

1. Construct the final shape model from the combined model.
2. Construct the final texture model from the combined model.

Fig. 1. *The relationship between steps of Active Appearance Model Construction.*

References

1. G. T.F.Cootes and C.J.Taylor, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **23** (2001).

1. **Translation**
   - Calculate the center of mass for each shape:

$$\bar{x} = \frac{x_1 + \ldots + x_n}{n}, \bar{y} = \frac{y_1 + \ldots + y_n}{n}, \bar{z} = \frac{z_1 + \ldots + z_n}{n}, \bar{w} = \frac{w_1 + \ldots + w_n}{n}$$

   - Translate each shape to its center of mass:

$$(x', y') \mapsto (x - \bar{x}, y - \bar{y}), (z', w') \mapsto (z - \bar{z}, w - \bar{w})$$

2. **Scaling**
   - Calculate scaling for each shape using L2-norm:

$$ScaleX = \sqrt{x_1'^2 + y_1'^2 + x_2'^2 + y_2'^2 + \ldots + x_n'^2 + y_n'^2}$$

$$ScaleZ = \sqrt{z_1'^2 + w_1'^2 + z_2'^2 + w_2'^2 + \ldots + z_n'^2 + w_n'^2}$$

   - $X' = \frac{X'}{ScaleX}, Z' = \frac{Z'}{ScaleZ}$

3. **Rotation**
   - Construct $\underline{X}$ and $\underline{Z}$ as following

$$\underline{X} = \begin{pmatrix} x_1' & x_2' & \ldots & x_n' \\ y_1' & y_2' & \ldots & y_n' \end{pmatrix}^T, \underline{Z} = \begin{pmatrix} z_1' & z_2' & \ldots & z_n' \\ w_1' & w_2' & \ldots & w_n' \end{pmatrix}^T$$

   - Calculate Singular Value Decomposition (SVD) for $\underline{Z}^T * \underline{X}$ and find U, S, V.
   - The rotation matrix(R) to match $\underline{Z}$ to $\underline{X}$ is :

$$R = V * U^T = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

   - Calculate $Z_{rotate} = R * \underline{Z}$.

Fig. 2. *Procrustes Analysis between two shapes.*

1. Choose the first shape in the training set as the mean shape.
While
   3. Align other shapes to the mean shape by Procrustes analysis.
   4. Calculate the mean shape from the aligned shapes.

until $distance(MeanShape, LastMeanShape) < Threshold$
(A suggestion for the threshold is 10^{-3})

Fig. 3. *Shape alignment to the mean shape.*

1. Built the S' (S transpose)
   S' is a $2N_t * N$ matrix, where N_t is the number of landmarks and N is the number of shapes in the training set.
2. Calculate Covariance Matrix of S'.
   Covariance Matrix has $2N_t * 2N_t$ dimensions.
3. Calculate PCA on Covariance Matrix.
4. Sort eigenvalues and corresponding eigenvectors from the greatest eigenvalue to the smallest one.
5. Find t based on $\sum_{i=1}^{t} \lambda_{s,i} \geq SV \times \sum_{i=1}^{2n} \lambda_{s,i}$.
6. Keep the first t eigenvectors and eigenvalues as the Shape model.

Fig. 4. *Principle Component Analysis of the aligned shapes.*

2. B.Stegmann and D. D. Gomez, *A Brief Introduction to Statistical Shape Analysis*, tech. rep., Technical University of Denmark (March 2002).
3. M. B. Stegmann, Active appearance models: Theory, extensions and cases, Master's thesis, Technical university of Denmark (2000).
4. S. Zambal, 3d active appearance models for segmentation of cardiac mri data, Master's thesis, Technische Universitat Wien (2005).

For i^{th} pixel in the mean shape and k^{th} shape in the training set

1. Find positions of the first Triangle $(x_1, y_1), (x_2, y_2), (x_3, y_3)$.
2. Calculate α, β and γ.

$$\beta = \frac{yx_3 - x_1y - x_3y_1 - y_3x + x_1y_3 + xy_1}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2}$$

$$\gamma = \frac{xy_2 - xy_1 - x_1y_2 - x_2y + x_2y_1 + x_1y}{-x_2y_3 + x_2y_1 + x_1y_3 + x_3y_2 - x_3y_1 - x_1y_2}$$

$$\alpha = 1 - (\beta + \gamma)$$

if $0 \leq \alpha, \beta, \gamma \leq 1$ then the i^{th} pixel is in this triangle:

4. Find positions of the corresponding triangle in the k^{th} shape, $(\acute{x}_1, \acute{y}_1), (\acute{x}_2, \acute{y}_2), (\acute{x}_3, \acute{y}_3)$
5. Find i^{th} pixel in the k^{th} shape by

$$\acute{x} = \alpha\acute{x}_1 + \beta\acute{x}_2 + \gamma\acute{x}_3$$

$$\acute{y} = \alpha\acute{y}_1 + \beta\acute{y}_2 + \gamma\acute{y}_3$$

6. keep the gray value of the point $(\acute{x}, \acute{y})$

else go to the next Triangle and then step 2

Fig. 5. *Shape sampling by Delaunay triangulation of the mean shape.*[3]

1. Choose the first texture as the mean texture

While

2. Standardize the Mean Texture to have zero mean and unit variance
   - *calculate zero-mean Mean texture*
   - *find the variance of the zero-mean Mean texture*
   - *divide the zero-mean Mean texture to the* $\sqrt{variance}$
3. Align each textures to the standardized mean texture
   - $\alpha = (current_texture) * (standard_mean_texture)$
   - $\beta = \frac{current_texture}{n}$
   - $current_texture = \frac{current_texture - \beta}{\alpha}$
4. Calculate the mean shape from the aligned textures

until ($Standard_Mean_Texture$ changing be less than a Threshold)
(A suggestion for the threshold is 10^{-5})

Fig. 6. *Texture Alignment to the mean Texture.*[3]

1. Built the whole texture matrix: G
   G is a $N_g * N$ matrix, where N_g is the number of texture pixels and N is the number of shapes in the training set.
2. Calculate Covariance Matrix of G' : $\Sigma_{g,temp}$.
   $\Sigma_{g,temp}$ *has $N * N$ dimensions*
3. Calculate PCA on $\Sigma_{g,temp}$
   $P_{g,temp}$ *and* $\Lambda_{g,temp}$ *are the eigenvectors and eigenvalues of* $\Sigma_{g,temp}$.
4. Convert $\Sigma_{g,temp}$ to the real covariance matrix: Σ_g[3,4,8]
   $P_g = G * P_{g,temp}$
   $\lambda_g = \lambda_{g,temp}$
   Normalize the columns of P_g : $P_{g,column(i)} = \frac{P_{g,column(i)}}{\sqrt{\lambda_{g,i}}}$
4. Sort eigenvalues and corresponding eigenvectors from the greatest eigenvalue to the smallest one.
5. find t' based on $\sum_{i=1}^{t'} \lambda_{g,i} \geq SV2 \times \sum_{i=1}^{2n} \lambda_{g,i}$.
6. Keep the first t' eigenvectors and eigenvalues as the texture model.

Fig. 7. *Fast Covariance Matrix Calculation.*[8]

5. E. Jones and S. Soatto, Layered active appearance models, in *The Tenth IEEE International Conference on Computer Vision*, (Beijing, China, 2005).
6. R.S.Schestowitz, *Project Background and Description*, tech. rep., University of Manchester (November 2003).
7. D. T.F.Cootes, C.J.Taylor and J.Graham, *Computer Vision and Image Understanding* **61**, 38 (1995).
8. T. F. Cootes and C. J. Taylor, *Statistical Models of Appearance for Computer Vision*, tech. rep., University of Manchester (December 2000).