CGL Technical Report, No. TP-2008-03, December 2008

Statistical Landmark Models
An ASM Approach

Panagiotis Perakis and Theoharis Theoharis
Computer Graphics Laboratory
Department of Informatics and Telecommunications
University of Athens, GREECE

Abstract This note aims at giving a brief introduction to the field of Statistical Shape Analysis. Basic techniques such as Procrustes analysis, tangent space projection, Principal Component Analysis and Active Shape Models (ASM) are presented. A biometric case study of 3D facial landmarks is subsequently demonstrated as an application, and a C++ implementation is presented.

Keywords landmarks, statistical shape analysis, Procrustes analysis, principal component analysis, active shape models, biometrics.

Version 1.1 Contact info: P. Perakis, `takis@antinoos.gr`.

Computer Graphics Laboratory
Department of Informatics and Telecommunications
University of Athens
15784 Ilisia
GREECE
http://graphics.di.uoa.gr

Contents

1 Introduction and Related Works

In a wide variety of disciplines it is of great practical importance to measure, describe and compare shapes of objects. We focus on the situation where the objects are summarized by points called landmarks. *Statistical shape analysis* is concerned with methodology for analyzing shapes in order to estimate population average shapes and the structure of shape variability. The foundation of statistical shape analysis was the pioneering work of Kendall (1984) and Bookstein (1986). Main contributions in shape analysis where the famous "Snakes" paper by Kass, Witkin and Terzopoulos (1988) [5] and subsequent papers published as "Active Shape Models: Smart Snakes" (1992) and "Active Shape Models: their training and application" by T. F. Cootes, C. J. Taylor, D. H. Cooper and J. Graham (1995) [2]. Although Snakes and Active Shape Models were rightfully claimed as they both are deformable models, contrary to Snakes, Active Shape Models (ASMs) have global constraints w.r.t. shape. These constraints are learned through observation, giving the model flexibility, robustness and specificity, as the model only can synthesize plausible instances w.r.t. the observations.

This note introduces the foundation of Active Shape Models, namely the statistical analysis of shapes, and presents a biometric case study of 3D facial landmark shapes.

2 Shapes and Landmarks

The word "shape" is very commonly used in everyday language, but what do we actually understand by the concept of shape? In this text we will adopt the definition by D.G. Kendall [4]:

> **Shape** *is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object.*

According to this, shape is, in other words, invariant to Euclidean similarity transformations. This is reflected in Fig. 1.

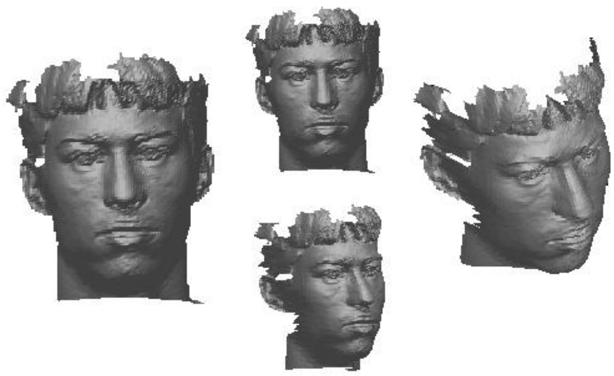


Figure 1: The same face shape under different Euclidean transformations.

Two objects have the same *shape* if they can be translated, scaled and rotated to each other so that they match exactly. Scale is sometimes considered a distinguishing characteristic.

Rigid shape *is all the geometrical information that remains when location and rotational effects are filtered out from an object.*

So, two objects have the same *size-and-shape* if they can be translated and rotated to each other so that they match exactly, i.e rigid shapes are rigid-body transformations of each other.

The next question that naturally arises is: How should one describe a shape? In everyday conversation, unknown shapes are often described as references to known shapes, e.g. "Italy has the shape of a boot". Such descriptions can obviously not easily be utilized in an algorithmic framework.

One way to describe a shape is by locating a finite number of points on the outline or other specific points. Consequently, the concept of a *landmark* is adopted. According to Dryden & Mardia [4]:

A **landmark** *is a point of correspondence on each object that matches between and within populations.*

Dryden & Mardia [4] also discriminates landmarks into categories:

Anatomical landmarks: *Points assigned by an expert that corresponds between organisms in some biologically meaningful way*, e.g. the corner of an eye.

Mathematical landmarks: *Points located on an object according to some mathematical or geometrical property*, e.g. a high curvature or an extremum point.

Pseudo-landmarks: *Constructed points on an object either on the outline or between anatomical or mathematical landmarks.*

Labeled landmarks: *Landmarks that are associated with a label (name or number), which is used to identify the corresponding landmark.*

Synonyms for landmarks include homologous points, interest points, nodes, vertices, anchor points, fiducial markers, model points, vertices, markers, key points etc.

2.1 Shape Space

A mathematical representation of an n-point shape in d dimensions could be to concatenate all point coordinates into a $k = nd$-vector and establish a *Shape Space* [4, 6, 1]. The *vector representation* for 3D shapes (i.e. $d = 3$) would then be:

$$\mathbf{x} = [x_1, x_2, ..., x_n, y_1, y_2, ..., y_n, z_1, z_2, ..., z_n]^T \quad (1)$$

where (x_i, y_i, z_i) represent the n landmark points.

If a relationship between the distance in shape space and Euclidean distance in the original space can be established, then we have a *metric space*. This relationship is called a *shape metric*. A set of shapes actually forms a Riemannian manifold containing the shape object under consideration (*Kendall shape space*).

Often used shape metrics include the Haussdorf distance, the strain energy and the Procrustes distance. In the following we will use the celebrated *Procrustes distance*.

The squared Procrustes distance between two shapes, $\mathbf{x_1}$ and $\mathbf{x_2}$, is simply the sum of the squared point distances:

$$D_P^2 = |\mathbf{x_1} - \mathbf{x_2}|^2 = \sum_{j=1}^{n} [(x_{1j} - x_{2j})^2 + (y_{1j} - y_{2j})^2 + (z_{1j} - z_{2j})^2] \quad (2)$$

for shapes in a 3D original space, or:

$$D_P^2 = |\mathbf{x_1} - \mathbf{x_2}|^2 = \sum_{j=1}^{k} (x_{1j} - x_{2j})^2 \quad (3)$$

for the general case. It is a Euclidean metric in the $k = nd$ dimensional shape space.

The *centroid* of a shape is the center of mass (CM) of the physical system consisting of unit masses at each landmark. This is easily calculated as:

$$\mathbf{r_{CM}} = [x_{CM}, y_{CM}, z_{CM}]^T = [\frac{1}{n}\sum_{j=1}^{n} x_j, \frac{1}{n}\sum_{j=1}^{n} y_j, \frac{1}{n}\sum_{j=1}^{n} z_j]^T \quad (4)$$

in a 3D original space (i.e. $d = 3$).

The *centroid size* is used as a shape size metric:

$$S(\mathbf{x})^2 = \sum_{j=1}^{n} [(x_j - x_{CM})^2 + (y_j - y_{CM})^2 + (z_j - z_{CM})^2] \quad (5)$$

for shapes in an original 3D space (i.e. $d = 3$), or in the general case:

$$S(\mathbf{x})^2 = \sum_{j=1}^{n}\sum_{i=1}^{d} (x_{j,i} - x_{CM,i})^2 \quad (6)$$

The centroid size is the square root of the sum of squared Euclidean distances from each landmark $\mathbf{x_j}$ to the centroid $\mathbf{r_{CM}}$:

$$S(\mathbf{x})^2 = \sum_{j=1}^{n} |\mathbf{x_j} - \mathbf{r_{CM}}|^2 \quad (7)$$

in the original Euclidean space. The centroid size has the property that $2nS(\mathbf{x})^2$ equals the sum of the interlandmark distances.

2.2 Shape Alignment

To obtain a true representation of landmark shapes, location, scale and rotational effects need to be filtered out. This is carried out by establishing a common coordinate reference to which all shapes are aligned.

Alignment is performed by minimizing the *Procrustes distance*

$$D_P^2 = |\mathbf{x_i} - \mathbf{x_m}|^2 \tag{8}$$

of each shape $\mathbf{x_i}$ to the Mean Shape $\mathbf{x_m}$.

The Mean Shape $\mathbf{x_m}$ is the *Procrustes mean*:

$$\mathbf{x_m} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x_i} \tag{9}$$

of all example shapes $\mathbf{x_i}$, where N denotes the number of example shapes.

The alignment procedure is commonly known as Procrustes Analysis [2, 1, 6, 3], and is used to calculate the Mean Shape of example shapes. Although there are analytic solutions, a typical iterative approach is the following:

Algorithm 1: Procrustes Analysis

- Compute the centroid of each example shape
- Translate each example shape so that its centroid is at the origin (0,0,0)
- Scale each example shape so that its size is 1
- Assign the first example shape to the mean shape $\mathbf{x_m}$
- REPEAT
  - Assign the mean shape $\mathbf{x_m}$ to a reference mean shape $\mathbf{x_0}$
  - Align all example shapes to the reference mean shape $\mathbf{x_0}$ by an optimal rotation
  - Recalculate the mean shape $\mathbf{x_m}$
  - Translate the mean shape so that its centroid is at the origin (0,0,0)
  - Scale the mean shape so that its its size is 1
  - Align the mean shape $\mathbf{x_m}$ to the reference mean shape $\mathbf{x_0}$ by an optimal rotation
  - Compute the Procrustes distance of the mean shape $\mathbf{x_m}$ to the reference mean shape $\mathbf{x_0}$: $|\mathbf{x_0} - \mathbf{x_m}|$
- UNTIL Convergence (mean shape doesn't change much): $|\mathbf{x_0} - \mathbf{x_m}| < \varepsilon$

In cases where the size of the shape is of great importance, it is not filtered out by scaling shapes to unit size. In these cases shapes are considered rigid shapes and are aligned by just performing the translational and rotational transformations.

2.3 Shape Transformations

As we have already said, to obtain a true representation of landmark shapes, location, scale and rotational effects need to be filtered out, by bringing shapes to a common frame of reference. This is carried out by performing translational, scaling and rotational transformations. Notice that different approaches to alignment can produce different distributions of the aligned shapes.

Translation to the centroid is performed by applying to the landmark points $\mathbf{x_j}$ the following transformation:

$$\begin{aligned} x'_j &= x_j - x_{CM} \\ y'_j &= y_j - y_{CM} \\ z'_j &= z_j - z_{CM} \end{aligned} \tag{10}$$

where $[x_{CM}, y_{CM}, z_{CM}]^T$ the centroid and $j \in [1..n]$.

Scaling to unit size is performed by applying to the landmark points $\mathbf{x_j}$ the following transformation:

$$\begin{aligned} x'_j &= \alpha x_j \\ y'_j &= \alpha y_j \\ z'_j &= \alpha z_j \end{aligned} \tag{11}$$

where $\alpha = 1/S(\mathbf{x})$ the scaling factor, $S(\mathbf{x})$ the shape's size and $j \in [1..n]$.

Rotation in a 3D original space is a little more complicated. We need to calculate a rotational transformation $R(\mathbf{x})$ so as to minimize the Procrustes distance $|R(\mathbf{x}) - \mathbf{x_0}|$ of the transformed shape $R(\mathbf{x})$ to a reference shape $\mathbf{x_0}$. The rotational transformation R can be expressed as a product of three rotations $R = R_{x,\theta} \cdot R_{y,\phi} \cdot R_{z,\psi}$. These can be expressed in a matrix form:

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{12}$$

$$R_{y,\phi} = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \tag{13}$$

$$R_{z,\psi} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$

After setting partial derivatives of $|R(\mathbf{x}) - \mathbf{x_0}|^2$ w.r.t each parameter to zero and some formal calculations, we have:

$$\theta = \tan^{-1}\left(\frac{S_{z0,y} - S_{y0,z}}{S_{y0,y} + S_{z0,z}}\right) \tag{15}$$

$$\phi = \tan^{-1}\left(\frac{S_{x0,z} - S_{z0,x}}{S_{z0,z} + S_{x0,x}}\right) \tag{16}$$

$$\psi = \tan^{-1}\left(\frac{S_{y0,x} - S_{x0,y}}{S_{x0,x} + S_{y0,y}}\right) \tag{17}$$

where:

$$\begin{array}{lll} S_{x0,x} = \sum_{j=1}^{n} x_{0j}x_j \ , & S_{x0,y} = \sum_{j=1}^{n} x_{0j}y_j \ , & S_{x0,z} = \sum_{j=1}^{n} x_{0j}z_j \ , \\ S_{y0,x} = \sum_{j=1}^{n} y_{0j}x_j \ , & S_{y0,y} = \sum_{j=1}^{n} y_{0j}y_j \ , & S_{y0,z} = \sum_{j=1}^{n} y_{0j}z_j \ , \\ S_{z0,x} = \sum_{j=1}^{n} z_{0j}x_j \ , & S_{z0,y} = \sum_{j=1}^{n} z_{0j}y_j \ , & S_{z0,z} = \sum_{j=1}^{n} z_{0j}z_j \ . \end{array}$$

So, the rotational transformation of every landmark point $\mathbf{x_j}$ gives:

$$\mathbf{x_j}' = R(\mathbf{x_j}) = R_{x,\theta}\left(R_{y,\phi}\left(R_{z,\psi}(\mathbf{x_j})\right)\right) \tag{18}$$

For the case of a 2D shape only the $R_{z,\psi}(\mathbf{x_j})$ transformation is applied.

2.4 Shape Variations

After bringing landmark shapes into a common frame of reference by applying Procrustes analysis and estimating the landmarks' Mean Shape, further analysis can be carried out for describing the shape variations. This shape decomposition is performed by applying Principal Component Analysis to the aligned shapes.

Due to size normalization of Procrustes analysis, all shape vectors live in a hyper sphere manifold in shape space, which introduces non-linearities if large shape scalings occur. Since PCA is a linear procedure all aligned shapes are at first projected to the tangent space of the Mean Shape. This way shape vectors lie in a hyper plane instead of a hyper sphere, and non-linearities are filtered out.

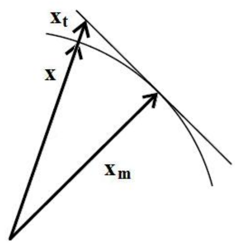


Figure 2: Tangent space projection $\mathbf{x_t}$ of shape vector $\mathbf{x}$ to the mean shape $\mathbf{x_m}$.

The tangent space projection (Fig. 2) linearizes shapes by scaling them with a factor α:

$$\mathbf{x_t} = \alpha\mathbf{x} = \frac{|\mathbf{x_m}|^2}{\mathbf{x_m}\cdot\mathbf{x}}\mathbf{x}$$

Aligned shape vectors form a distribution in the nd dimensional shape space. If points are not representing landmarks, then they will be totally uncorrelated, i.e. purely random. On the other hand, if points represent a certain class of shapes, then they will be correlated in some degree, and this fact will be exploited to reduce dimensionality.

If landmark points have a specific distribution we can model this distribution, by estimating a vector $\mathbf{b}$ of parameters that describes shapes deformations.

The approach according to Cootes & Taylor [1, 3] is as follows:

Algorithm 2: Principal Component Analysis

- Determine the mean shape.
- Determine the covariance matrix of the shape vectors.
- Compute the eigenvectors $\boldsymbol{\phi}_\mathbf{i}$, and corresponding eigenvalues λ_i of the covariance matrix, sorted in descending order.

By applying Procrustes analysis the Mean Shape is determined and example shapes are aligned and projected to Mean Shape's tangent space. Typically one would apply PCA on variables with zero mean.

The covariance matrix of N example shapes is calculated according to:

$$\mathbf{C_x} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x_i} - \mathbf{x_m})(\mathbf{x_i} - \mathbf{x_m})^T \tag{19}$$

If $\boldsymbol{\Phi}$ contains (in columns) the $k = nd$ eigenvectors $\boldsymbol{\phi}_\mathbf{i}$ of $\mathbf{C_x}$, by projecting aligned original example shapes to the eigenspace we uncorrelate them:

$$\mathbf{y} = \boldsymbol{\Phi}^T \cdot (\mathbf{x} - \mathbf{x_m}) \tag{20}$$

and the covariance matrix of projected example shapes:

$$\mathbf{C_y} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{y_i} - \mathbf{y_m})(\mathbf{y_i} - \mathbf{y_m})^T \tag{21}$$

becomes a diagonal matrix of the eigenvalues λ_i, so as to have:

$$\mathbf{C_x} \cdot \boldsymbol{\Phi} = \boldsymbol{\Phi} \cdot \mathbf{C_y} \quad , \quad \mathbf{C_y} = \boldsymbol{\Phi}^T \cdot \mathbf{C_x} \cdot \boldsymbol{\Phi} \tag{22}$$

The resulting transform is known as the *Karhunen-Loéve transform*, and achieves our original goal of creating mutually uncorrelated features.

To back project uncorrelated shape vectors to the original shape space, we can use:

$$\mathbf{x} = \mathbf{x_m} + \boldsymbol{\Phi} \cdot \mathbf{y} \tag{23}$$

Therefore, if $\boldsymbol{\Phi}$ contains (in columns) the p eigenvectors corresponding to the p largest eigenvalues, then we can approximate any example shape $\mathbf{x}$, using:

$$\mathbf{x'} \approx \mathbf{x_m} + \boldsymbol{\Phi} \cdot \mathbf{b} \tag{24}$$

where $\mathbf{b}$ is a p dimensional vector given by:

$$\mathbf{b} = \boldsymbol{\Phi}^T \cdot (\mathbf{x} - \mathbf{x_m}) \tag{25}$$

Vector $\mathbf{b}$ is nothing but the projection of $\mathbf{x}$ onto the subspace spanned by the p most significant eigenvectors of eigenspace (*principal components*). By selecting the p largest eigenvalues the mean square error between $\mathbf{x}$ and its approximation $\mathbf{x'}$ is minimum, since the last $k-p$ components are frozen to their respective mean values [7].

This way an *Active Shape Model* (ASM) is created [2, 1, 3].

The vector $\mathbf{b}$ defines the deformation parameters of the model. By varying the components of $\mathbf{b}$ we can create shape variations. By applying limits to each b_i:

$$b_i = \pm 3\sqrt{\lambda_i} \tag{26}$$

we can create marginal shape deformations, since each eigenvalue represents the data variance at the corresponding eigenspace axis [1, 7].

The number p of most significant eigenvectors and eigenvalues to retain (*modes of variations*), can be chosen so that the model represents a given proportion of the total variance of the data, i.e. the sum V_{tot} of all the eigenvalues.

$$\sum_{i=1}^{p} \lambda_i \geq f \cdot V_{tot} \tag{27}$$

Factor f represents the percentage of total variance incorporated into ASM. Least significant eigenvalues that are not incorporated, are considered to represent noise [1, 7].

2.5 Fitting Landmarks to the ASM

General purpose feature detection methods are not capable to identify and label the detected candidate landmarks. They can only locate mathematical landmarks according to some geometrical property of objects, i.e. high curvature or an extremum point. It is clear that some topological properties of the class of shape objects need to be taken under consideration. To address this problem we can use the ASM. Candidate landmarks, irrespectively of the way they are produced, have to be consistent with the corresponding ASM that represents the class of shapes. This is done by fitting a candidate landmark set to the ASM and checking the deformation parameters $\mathbf{b}$ to be inside certain margins.

Fitting a set of points $\mathbf{y}$ to the ASM $\mathbf{x}$ is done by minimizing the Procrustes distance in a simple iterative approach, adapted from Cootes & Taylor [1]:

Algorithm 3: Landmark Fitting

- Translate $\mathbf{y}$ so that its centroid is at the origin (0,0,0).
- Scale $\mathbf{y}$ shape so that its size is 1
- REPEAT
  - Align $\mathbf{y}$ to the mean shape $\mathbf{x_m}$ by an optimal rotation.
  - Compute the Procrustes distance of $\mathbf{y}$ to the mean shape $\mathbf{x_m}$: $|\mathbf{y} - \mathbf{x_m}|$
- UNTIL Convergence (Procrustes distance doesn't change much).
- Project $\mathbf{y}$ into tangent space of $\mathbf{x_m}$.
- Determine the model deformation parameters $\mathbf{b}$ that match to $\mathbf{y}$:

$$\mathbf{b} = \mathbf{\Phi}^T \cdot (\mathbf{y} - \mathbf{x_m})$$

- Accept $\mathbf{y}$ as a member of the shape's class, if $\mathbf{b}$ satisfies certain constraints.

Notice that scaling is not applied when we need to retain shape size.

We consider a landmark shape as plausible, if it is consistent with marginal shape deformations. Let's say that certain b_i satisfy the condition:

$$|b_i| \leq 3\sqrt{\lambda_i}$$

then the candidate landmark shape belongs to the class with probability $Pr(\mathbf{y})$, where:

$$Pr(\mathbf{y}) = \frac{\sum \lambda_i}{V_{tot}} \tag{28}$$

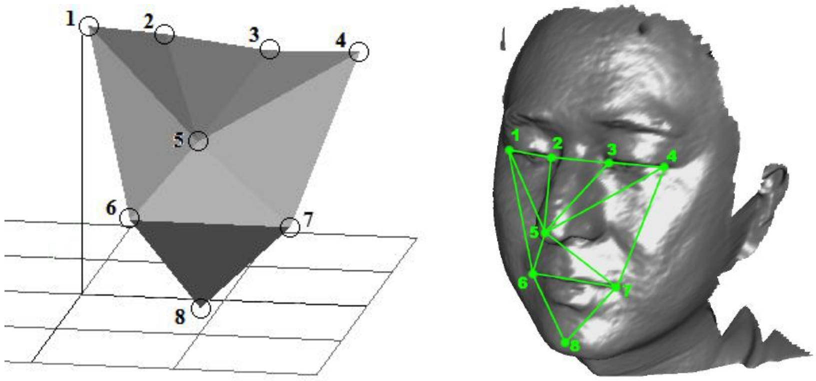
and V_{tot} is the sum of all the eigenvalues, that represents the total data variance.

If $Pr(\mathbf{y})$ exceeds a certain threshold limit, the landmark shape is considered plausible, otherwise it is rejected as a member of the class. Other criteria of declaring a shape as plausible can also be applied [1, 3].

3 3D Facial Landmarks

This section describes the case study of creating an ASM of landmark points on 3D facial objects. We used a set of 8 anatomical (labeled) landmarks: right eye outer corner (1), right eye inner corner (2), left eye inner corner (3), left eye outer corner (4), nose tip (5), mouth right corner (6), mouth left corner (7) and chin tip (8) (Fig. 3).

To create our training examples dataset, we used 150 frontal faces with neutral expressions, which were manually annotated. Specifically, regarding faces there is a great variability in the visibility of landmarks according to pose changes. For this reason frontal face scans were used. This consideration has a minor effect to the 3D model. Although faces with neutral expressions were chosen, these facial expressions affect positions of mouth corners (landmarks 6 & 7). Therefore, when one needs to incorporate them into model, faces with expressions have to be included in the training dataset.



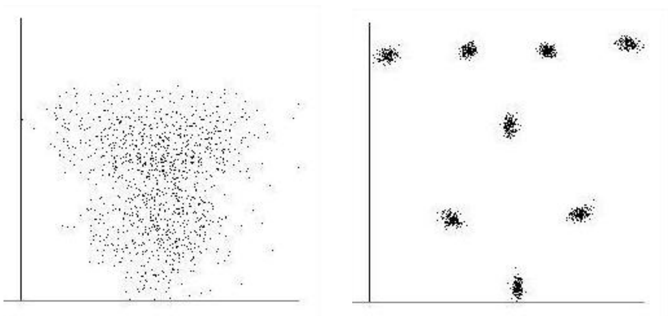
(a) (b)

Figure 3: Landmark models: (a) Landmark model as a 3D object; (b) Landmark model overlaid over a 3D facial scan.

Furthermore, placing landmarks is a subjective task, so, in a strict annotating procedure, intra- and inter-annotator variability studies should be carried out (also called *repeatability* and *reproducibility* studies). This is accomplished by letting a set of operators annotate the dataset several times each. At each annotation the image order should be randomized to remove ordering-bias. From these sets, landmarking variances between and within annotators are estimated and the maximum likelihood landmark set is selected [6].

3.1 Alignment

To retain the actual landmark shape variation, Procrustes analysis and tangent space projection have been carried out on the example shapes. Since for our purposes the size of the shape is of great importance, it is not filtered out by scaling shapes to unit size.

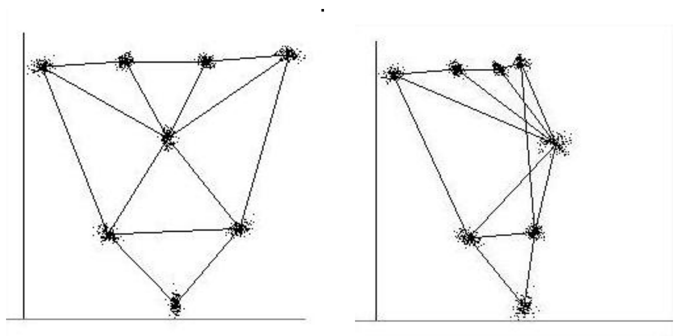


(a) (b)

Figure 4: Landmark alignment: (a) Unaligned Landmarks; (b) Aligned Landmarks.

Unaligned landmark points (Fig. 4 (a)) are aligned by applying Procrustes analysis (Fig. 4 (b)). Point clouds of aligned landmarks seem to have a multivariate

Gaussian distribution in 3D space. The axes of the Gaussians are analogous to the three standard deviations.



(a) (b)

Figure 5: Landmark Mean Shape estimation: (a) Landmark Cloud & Mean Shape at 0°; (b) Landmark Cloud & Mean Shape at 60°.

The centroid of each landmark cloud coincides with the corresponding landmark point of the Mean Shape (Fig. 5 (a),(b)).

3.2 Statistical Analysis

Point clouds of aligned landmarks represent landmark "movements" in 3D space. Looking at the correlation matrix we see that these "movements" are highly correlated (Fig. 6). Black squares denote negative correlation values, white, positive correlation values and mean gray, zero correlation. Note that shape vectors are presented in a $(x_1, x_2, ..., x_8, y_1, y_2, ..., y_8, z_1, z_2, ..., z_8)$ manner.

The main diagonal of the covariance matrix contains the variances of each shape vector component:

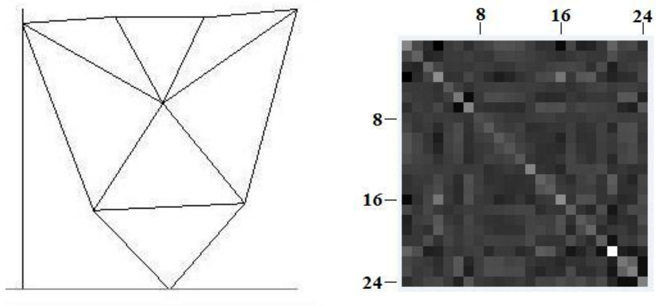
$$var(x_i) = \frac{1}{N-1} \sum_{k=1}^{N} (x_{k,i} - x_{m,i})^2 \tag{29}$$

and non diagonal values the covariances between any two components:

$$covar(x_i, x_j) = \frac{1}{N-1} \sum_{k=1}^{N} (x_{k,i} - x_{m,i})(x_{k,j} - x_{m,j}) \tag{30}$$

where $\mathbf{x_m}$ is the mean shape, $\mathbf{x_k}$ any example shape and N the examples number. The covariance matrix is symmetrical about the main diagonal, since $covar(x_i, x_j) = covar(x_j, x_i)$.

The values of the covariance indicates the strength of each relationship, and the sign whether the relationship is positive or negative. If the value is positive, the two components increase together. If it is negative, then if one component increases in one direction the other increases in the opposite direction (decreases). Notice that division is done by $N - 1$ since we are using an example dataset which is a representation of the entire population where division by N can properly been used.



(a) (b)

Figure 6: Statistical Analysis: (a) Landmark Mean Shape; (b) Correlation Matrix.

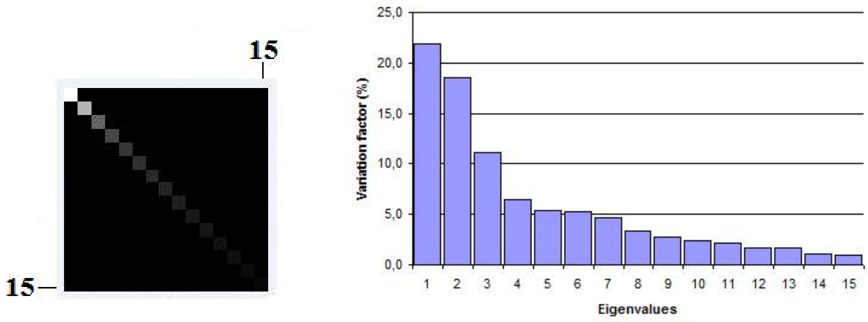
Consider the black square (1,4); it represents the correlation between (x_1, x_4), which are the x coordinates of left and right eye outer corners. We can conclude that they are negatively correlated; when right eye moves right, left eye moves left and vice versa. Consider the black square (6,7); it represents the correlation between (x_6, x_7), which are the x coordinates of mouth left and right corners. We can conclude that they are also negatively correlated; when mouth right corner moves right, left corner moves left and vice versa. Black squares (24,22) and (24,23) represent the correlation between (z_8, z_6) and (z_8, z_7), which are the z coordinates of chin tip versus mouth left and right corners. We can see a negative correlation, which means that chin tip and mouth corners "move" in opposite directions on z-axis. This is indicated in the third mode of variations (Fig. 10). Consider the gray squares of line (5); they represent the correlation of x,y,z coordinates of the nose tip with the other landmarks. We can conclude that the nose is mostly not correlated with any other landmark, because of the same gray color of the corresponding squares. It is the most robust facial landmark point. White square (21,21) represents the variance of (z_5), which is the z coordinate of the nose tip. We can see that this is the maximum variance, which is also indicated in the second mode of variations (Fig. 9).

By applying PCA we decompose shape variations by projecting to the eigenspace having an ordered basis of eigenvectors, where each shape component is ranked after the corresponding eigenvalue. This gives the components an order of significance. Each eigenvalue represents the variance in eigenspace axes which are orthogonal. Notice that correlation matrix of shape vectors in eigenspace has only diagonal elements, i.e. the eigenvalues (Fig. 7 (a)).

Modifying one component at a time we get the *principal modes of variations*. So, for each selected eigenvalue λ_i, we calculate the deformation parameter b_i within some limits $(\pm 3\sqrt{\lambda_i})$, and we get a corresponding mode of variations, which represents $f_i = \frac{\lambda_i}{V_{tot}}$ of the total shape variations in the dataset (Fig. 7 (b)).

We can see that the first mode, which is created by setting the deformation parameter values to $(b_1 = -3\sqrt{\lambda_1},\ b_1 = 0,\ b_1 = +3\sqrt{\lambda_1})$, captures the face shape (circular vs oval) and represents the 21.9% of total shape variations (Fig. 8).

We can also see that the second mode, which is created by setting the deformation

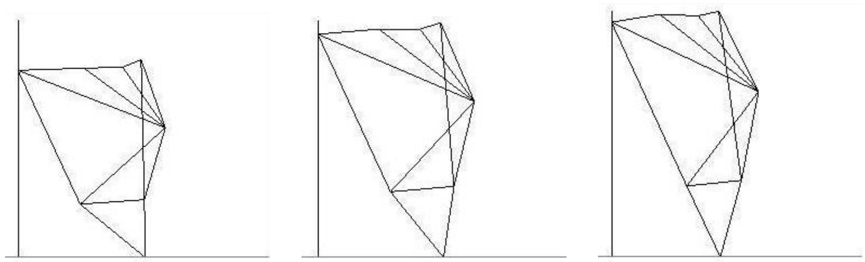


(a) (b)

Figure 7: Shape eigenvalues (a) and percentage of total variations they capture (b).

parameter values to ($b_2 = -3\sqrt{\lambda_2}$, $b_2 = 0$, $b_2 = +3\sqrt{\lambda_2}$), captures the nose shape (flat vs peaked) and represents the 18.6% of total shape variations (Fig. 9).

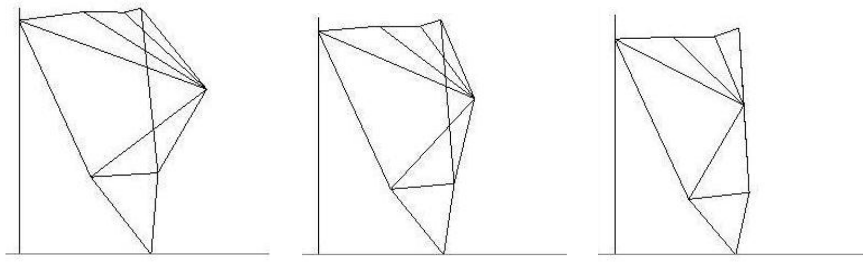
Finally, we can see that the third mode, which is created by setting the deformation parameter values to ($b_3 = -3\sqrt{\lambda_3}$, $b_3 = 0$, $b_3 = +3\sqrt{\lambda_3}$), captures the chin tip position (extruded vs intruded) and represents the 11.1% of total shape variations (Fig. 10).



(a) $b_1 = -3\sqrt{\lambda_1}$ (b) $b_1 = 0$ (c) $b_1 = +3\sqrt{\lambda_1}$

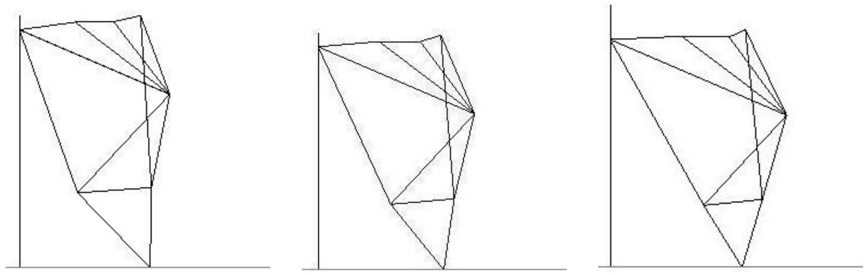
Figure 8: First mode of Mean Shape deformations (viewed at 60°).

The first three principal modes capture 51.6% of the total shape variations. We incorporated 15 eigenvalues (out of the total 24) in ASM, which represent 99.0% of total shape variations of the dataset. By selecting the 15 most significant eigenvalues and corresponding eigenvectors, each shape vector in the original 24-dimensional shape space is projected to a feature vector in an 15-dimensional *feature space*.



(a) $b_2 = -3\sqrt{\lambda_2}$ (b) $b_2 = 0$ (c) $b_2 = +3\sqrt{\lambda_2}$

Figure 9: Second mode of Mean Shape deformations (viewed at 60°).



(a) $b_3 = -3\sqrt{\lambda_3}$ (b) $b_3 = 0$ (c) $b_3 = +3\sqrt{\lambda_3}$

Figure 10: Third mode of Mean Shape deformations (viewed at 60°).

4 C++ Implementation

The following C++ code is intended to create an ASM.

It reads a space delimited text file (`.lsv`) where shape vectors of training example shapes have been saved. It outputs a space delimited text file (`.ash`) where the ASM resulting data are saved.

The Shape Vectors file (`.lsv`) has a header in the first line:
`[nExamples nLandmarks nDimensions].`
Each next row has an example Shape Vector:
`[x1 x2 .. xn y1 y2 .. yn z1 z2 .. zn]`

The resulting Active Shape Model file (`.ash`) has a header:
`[nFeatures nLandmarks nDimensions].`
Next, ASM data follows. First row has the Mean Shape Vector:
`[x1 x2 .. xn y1 y2 .. yn z1 z2 .. zn].`
Second row has the Shape Variance Vector:
`[sx1 sx2 .. sxn sy1 sy2 .. syn sz1 sz2 .. szn].`
Next rows have the `nF` principal shape eigenvectors of `nK=nL*nD` length:
`[e1 e2 .. enK].`
Last row has the `nF` principal eigenvalues:
`[d1 d2 .. dnF].`

At run time, user is prompted to enter the following parameters:

```
Enter name of Input File:
Enter name of Output File:
Enter percentage of Shape variability:
Perform Shape Size Normalization? [0/1]:
```

The following code listing "ASM Training" gives an idea of the data structures and the procedure sequence used to perform an ASM training from an example shapes dataset.

Code Listing: ASM Training

```
void main(int argc,char *argv[])
{
    char ifilename[1024];      //Input file name
    char ofilename[1024];      //Output file name
    bool sizenorm; int snorm;  //Denotes Shape Size Normalization
    int shapevar;              //Shape Variations into ASM

    int nE;  //Examples Count
    int nL;  //Landmark Count
    int nD;  //Space Dimensions
    int nK;  //Shape Space Dimensions (nK=nL*nD)
    int nF;  //Selected Features Number

    double **vmat;     //Array of Shape Vectors [nE x nK]
    double **cmat;     //Array of Shape Centroids (CMs) [nE x nD]
    double **corrmat;  //Correlation Matrix [nK x nK]
    double *mshape;    //Mean Shape vector [nK]
    double *varshape;  //Mean Shape Variances [nK]
    double *eval;      //Eigenvalues vector [nK]
    double **evec;     //Eigenvectors matrix [nK x nK]

    /********* user input *********/
    printf("Enter name of Input File: "); scanf("%s",&ifilename);
    printf("Enter name of Output File: "); scanf("%s",&ofilename);
    printf("Enter percentage of Shape variability: "); scanf("%i",&shapevar);
    printf("Perform Shape Size Normalization? [0/1]: "); scanf("%i",&snorm);
    sizenorm = (snorm==1);

    /********* main code *********/
    readHeader(ifilename, nE, nL, nD);
    nK=nL*nD;                            //Shape Space Dimensions
    allocateMatMemory;                   //Memory allocation of Matrices & Vectors
    readVectors(ifilename, vmat, nE, nK);

    calcCentroids(cmat, vmat, nE, nL, nD);
    ProcrustesAnalysis(mshape, cmat, vmat, sizenorm, nE, nL, nD);
    TangentProjection(mshape, vmat, nE, nL, nD);
    calcCorrMat(corrmat, mshape, vmat, nE, nK);
    getVariances(varshape, corrmat, nK);
    calcEigenvaluesAndEigenvectors(corrmat, evec, eval, nK, true);
    nF = getFeatureNum(eval, nK, shapevar/100.0f);      //Dimension of feature space

    saveASM (ofilename, mshape, varshape, evec, eval, nF, nL, nD);

    deallocateMatMemory;                  //Memory deallocation of Matrices & Vectors
    printf("Finished creating ASM");
}
```

References

[1] T. Cootes and C. Taylor, *Statistical models of appearance for computer vision*, Tech. report, University of Manchester, Oct 2001.

[2] T. Cootes, C. Taylor, D. Cooper, and J. Graham, *Active shape models: Their training and application*, Computer Vision and Image Understanding (1995).

[3] T. Cootes, C. Taylor, H. Kang, and V. Petrovic, *Handbook of face recognition*, ch. Modeling Facial Shape and Appearance, pp. 39–63, Springer, 2005.

[4] I. Dryden and K. Mardia, *Statistical shape analysis*, Wiley, 1998.

[5] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contours models*, International Journal of Computer Vision (1988).

[6] M. Stegman and D. Gomez, *A brief introduction to statistical shape analysis*, Tech. report, Technical University of Denmark, Mar 2002.

[7] S. Theodoridis and K. Koutroumbas, *Pattern recognition*, 3d ed., Academic Press, 2006.