

Chapter 2

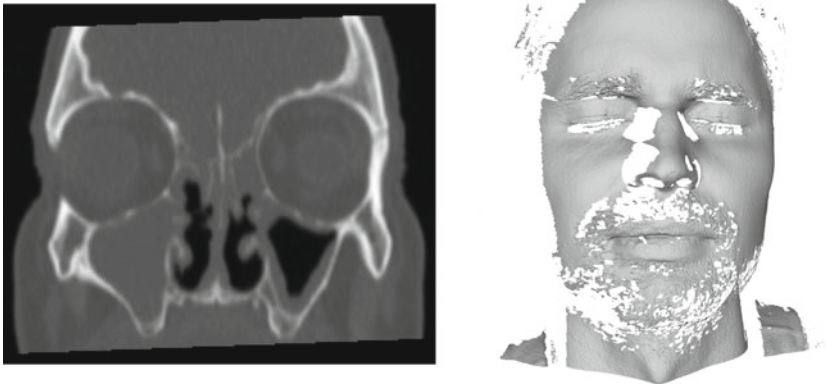
Statistical Shape Models (SSMs)

The automated extraction of objects from two-dimensional or three-dimensional image data is a complicated task that is not easy to solve. This is especially the case in medical image segmentation problems where the goal often is to extract highly variable anatomical structures from relatively low-quality image data [59]. The image data thereby originates for example from approaches that produce 2D images of the desired anatomy, like conventional X-ray imaging, or from newer approaches that produce 3D volumetric images of the anatomy under consideration, like e.g. *Computed Tomography* (CT) scans or *Magnetic Resonance Imaging* (MRI). Some exemplary segmentation problems will be discussed in Chap. 4. They include the automated extraction of the paranasal sinuses (Sect. 4.1) and the bones in the human knee (Sect. 4.2), which are both needed for surgical planning, but also the extraction of faces from range scan data (Sect. 4.3) is an increasingly popular application as it is needed for real time facial animation [79] or face recognition [20].

For problems of the above-mentioned kind, the available image or range information often does not suffice in order to extract the desired structures due to artifacts in the data, missing data, or poor contrast [59]. As a consequence, high-level information is needed in order to replace the missing information and to distinguish between correct and erroneous information. Two examples can be seen in Fig. 2.1. Without anatomical knowledge it is hard to identify the paranasal sinuses in the CT slice in Fig. 2.1a. However, it is easy to fill the holes of the disturbed range scan of a human face in Fig. 2.1b as all humans have a clear understanding of how a face should look like. Likewise, a medical expert (e.g. a surgeon) has no problem outlining the paranasal sinuses as he or she possesses the required anatomical knowledge.

Now, the question is how this high-level knowledge can be integrated into automatic object extraction approaches. This is where the so-called *Statistical Shape Models* (SSMs) come into play: When many hand-segmentations of the desired object class (e.g. the paranasal sinuses) or many complete range scans of the desired object class (e.g. human faces) are available, one can use this information in order to extract objects from the same object class in newly acquired data.

There exist many different variants of statistical shape models. For an extensive overview we refer the reader to the review article by Heimann and Meinzer [60].



(a): CT cross-section of a human head. (b): Disturbed range scan of a human face.

Fig. 2.1 Exemplary CT cross-section (a) and range scan (b) of a human head

In the following sections, we will first concentrate on the so-called *Point Distribution Model* (PDM) that has been introduced by Cootes et al. in [32] as it is probably the best-known method in the area of statistical shape models [60]. Afterwards, in Sect. 2.3, we will discuss an extension of the PDM that works with implicit, level set-based shape representations, and we will deal with the problem of rigidly aligning the training shapes in Sect. 2.4. In Sect. 2.5, we will address the problem of limited training data, and in Sect. 2.5 we will finally mention some previous approaches by other researchers who tried to address this fundamental problem.

2.1 Definition of Shape

In the introduction to this chapter, we have explained that we are interested in introducing high-level information into object extraction problems by modeling the statistical shape properties of a particular object class. So, before we can continue, we need to define what we mean when we speak about the *shape* of an object. A common general definition of the term *shape* is as follows [64, p. 1, Definition 1.1]:

Shape is all the geometrical information that remains when location, scale and rotational effects are removed from an object.¹

This definition has been originally coined by Kendall [70] back in 1977.

More specifically, when we refer to the geometrical information of an object, we mean the silhouette of an object, i.e. the geometric outline of an object in 2D or 3D [37]. This geometric outline can be represented in a variety of different ways,

¹Reprinted from [64, p. 1, Definition 1.1], © 2016 John Wiley and Sons Ltd.

like, e.g. splines [18] or fourier descriptors [122]. However, the two most popular representations are polygons (in 2D or equivalently polygon meshes in 3D) [60] and the zero level set of a higher-dimensional level set function that has been introduced in Sect. 1.4.1 [40].

A two-dimensional shape $\tilde{C}$ that is represented by a polygon is completely defined by an ordered list of 2D points

$$\tilde{C} = (x_1, y_1, \dots, x_r, y_r), \quad (2.1)$$

together with the conventions that each point (x_i, y_i) is connected by a straight line to its successor (x_{i+1}, y_{i+1}) and that the point (x_r, y_r) is connected by a straight line to the point (x_1, y_1) . This last convention may optionally be dropped in order to allow *open* shapes. Similarly, a three-dimensional shape $\tilde{C}$ that is represented by a polygon mesh is completely defined by an ordered list of 3D points

$$\tilde{C} = (x_1, y_1, z_1, \dots, x_r, y_r, z_r), \quad (2.2)$$

together with a set of connectivity rules that define which points (x_i, y_i, z_i) have to be connected by straight lines in order to obtain a surface that is composed out of many polygons of a specific type (e.g. triangles or quadrilaterals). As in Eqs. (2.1) and (2.2) the shape $\tilde{C}$ is directly defined by a list of points, we refer to these representations as *explicit* shape representations. Some examples can be seen in Fig. 2.2.

In contrast to the above-mentioned *explicit* shape representations, the zero level set of a higher-dimensional level set function is called an *implicit* shape representation as the shape C is indirectly obtained as the set of points for which a higher-dimensional embedding function $\Phi : \Omega \rightarrow \mathbb{R}$ equals zero. This has been defined in Eq. (1.32) as

$$C = \{\vec{x} \mid \Phi(\vec{x}) = 0\},$$

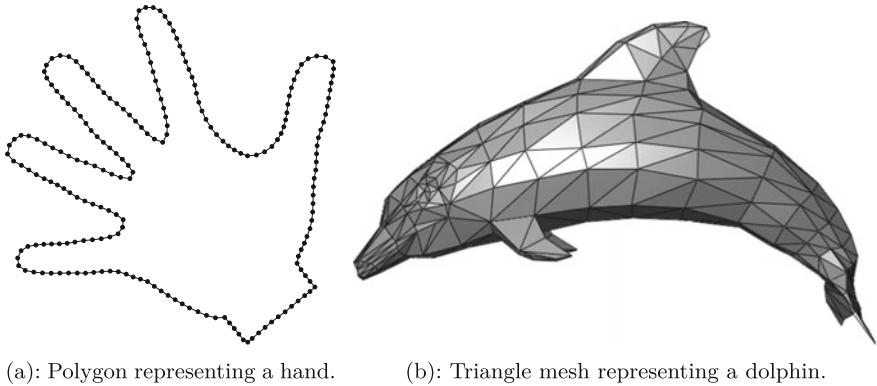


Fig. 2.2 Exemplary explicit shape representations of a hand (a) and a dolphin (b). *Source of subfigure (b):* Wikipedia (public domain)

where $\vec{x} \in \mathbb{R}^d$ and $\Omega \subset \mathbb{R}^d$. The variable d denotes the dimension of the object so that typical values for d are 2 or 3. As already mentioned in Sect. 1.4.1, it is common to use the signed distance function as the higher-dimensional embedding function for the shape C . However, other functions are also possible [41]. For an exemplary depiction of such an implicit shape see Fig. 1.2a.

2.2 An Explicit Linear Parametric Statistical Shape Model

Now that we have defined the term shape and provided two methods to represent a shape, we continue by introducing the *Point Distribution Model* (PDM) by Cootes et al. In order to construct the PDM, one needs first of all a set of (generally hand-labeled) training shapes $\{\tilde{C}_1, \dots, \tilde{C}_n\}$ that are given in the explicit shape representation discussed above. All these shapes must belong to the same shape class, which means that all training samples represent different shapes of the same object, e.g. a hand (c.f. Fig. 2.3). Additionally, all the training shapes $\tilde{C}_i$ have to be in correspondence to each other. This involves that all training shapes must be represented by the same amount of points and that all points have to be located at corresponding positions along all training shapes. Furthermore, we demand that all training shapes are aligned to each other with regard to rotation, translation, and scale as this information does not belong to the shape information of a specific shape class according to the definition in Sect. 2.1. How this alignment can be achieved will be discussed in Sect. 2.4. An example of 40 aligned training shapes for the hand shape class, where each shape

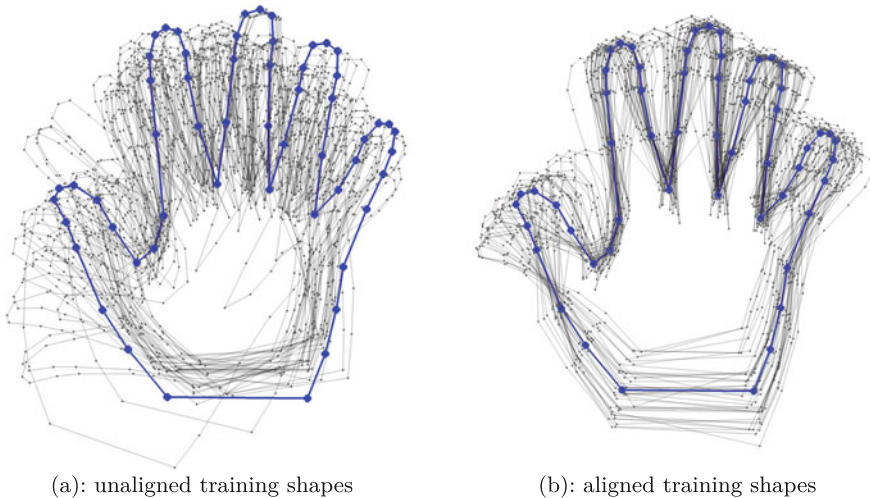


Fig. 2.3 Mean shape (blue) overlaid with the training shapes before (a) and after (b) they have been rigidly aligned with regard to rotation, translation, and scale. The shapes originate from the database that is described in [123]

is represented by 56 points, can be seen in Fig. 2.3b. The shapes originate from the database that is described in [123].

When all above-mentioned prerequisites are met, the simplest way to model new shapes is to allow linear combinations of the training shapes [19]:

$$\vec{C}_{\text{bary}}(\vec{w}) = \sum_{i=1}^n w_i \vec{C}_i, \text{ with } \sum_{i=1}^n w_i = 1. \quad (2.3)$$

In Eq. (2.3), the vector $\vec{w} = (w_1, \dots, w_n)$ is called the parameter vector (or weight vector) of the model $\vec{C}_{\text{bary}}$. In this case, the parameters $(w_1, \dots, w_n)$ are known as normalized *barycentric coordinates* [119], and the parameters $(\frac{1}{n}, \dots, \frac{1}{n})$ denote the barycenter (or balance point) of the training shapes. The barycenter is identical to the sample mean of the training shapes:

$$\vec{C}_{\text{bary}}\left(\frac{1}{n}, \dots, \frac{1}{n}\right) = \sum_{i=1}^n \frac{1}{n} \vec{C}_i = \frac{1}{n} \sum_{i=1}^n \vec{C}_i = \bar{\vec{C}}. \quad (2.4)$$

This representation allows to quickly generate new shapes from a set of training shapes. However, it has the drawbacks that linear dependencies between the training shapes are not considered and that it contains no information about the likelihood of a modeled shape $\vec{C}_{\text{bary}}(\vec{w})$.

In order to consider the statistical distribution of the training shapes, it is often reasonable to assume that all shapes of a specific shape class are distributed according to a multivariate Gaussian distribution with mean $\vec{\mu} \in \mathbb{R}^{dr}$ and covariance $\Sigma \in \mathbb{R}^{dr \times dr}$, where d is the dimension of the training shapes (i.e. $d \in \{2, 3\}$) and r are the number of points on each shape (c.f Eqs. (2.1) and (2.2)). The Gaussian distribution of likely shapes $\vec{C}$ is then given as

$$\vec{C} \sim \mathcal{N}(\vec{\mu}, \Sigma) = \frac{1}{\sqrt{(2\pi)^{dr} |\Sigma|}} \exp\left(-\frac{1}{2}(\vec{C} - \vec{\mu})\Sigma^{-1}(\vec{C} - \vec{\mu})^T\right). \quad (2.5)$$

In Eq. (2.5), an estimate of the mean $\vec{\mu}$ is given by the sample mean $\bar{\vec{C}}$ from Eq. (2.4) and an estimate $\hat{\Sigma}$ of the covariance matrix Σ can be obtained as detailed in appendix D. Now, one can assign a probability to each shape of the model from Eq. (2.3) via

$$P\left(\vec{C}_{\text{bary}}(\vec{w})\right) = \frac{1}{\sqrt{(2\pi)^{dr} |\hat{\Sigma}|}} \times \exp\left(-\frac{1}{2}\left(\vec{C}_{\text{bary}}(\vec{w}) - \bar{\vec{C}}\right)\hat{\Sigma}^{-1}\left(\vec{C}_{\text{bary}}(\vec{w}) - \bar{\vec{C}}\right)^T\right). \quad (2.6)$$

The drawback of Eq. (2.6) is that it contains a lot of redundant information. As mentioned above, the sample covariance matrix has the dimensions $dr \times dr$. However,

only n training shapes have been used to estimate the sample covariance matrix, where typically $n \ll dr$. In this case, it follows from the construction of the sample covariance matrix that it has at most rank $n - 1$ [23, Eq. (4.26f)]. So, a multivariate Gaussian distribution with a dimension less or equal to $(n - 1)$ would suffice in order to model the data statistics.

This low-dimensional Gaussian distribution can be obtained by applying the so-called *Principal Component Analysis* (PCA) [66] to the training shapes. The PCA is mathematically defined as an orthogonal transformation that maps the data to a new coordinate system in which most variation of the data occurs along the first axis, the second-most variation occurs along the second axis, and so on. The axes of the new coordinate system can be obtained as the eigenvectors of the sample covariance matrix $\hat{\Sigma}$. They are called *principal axes* or also *main modes of variation* and we denote them as $\{\tilde{\vec{C}}_1, \dots, \tilde{\vec{C}}_m\}$. It is $\tilde{\vec{C}}_i \in \mathbb{R}^{dr}$ and $m \leq (n - 1)$ as the sample covariance matrix has at most $(n - 1)$ nonzero eigenvalues. The eigenvalues $\{\sigma_1^2, \dots, \sigma_m^2\}$ of the sample covariance matrix $\hat{\Sigma}$ correspond to the variances along the principal axes and are called *principal components*. They have to be sorted in descending order $\sigma_1^2 \geq \sigma_2^2 \geq \dots \geq \sigma_m^2$ in order to be in accordance with the definition of the PCA (i.e. the most variation of the data occurs along the first principal axis etc.).

By stacking the eigenvectors $\tilde{\vec{C}}_i$ of the sample covariance matrix on top of each other, one obtains a matrix

$$\tilde{\mathbf{C}} = \begin{pmatrix} \tilde{\vec{C}}_1 \\ \vdots \\ \tilde{\vec{C}}_m \end{pmatrix}, \quad (2.7)$$

the rows of which define the linear subspace that is spanned by the training shapes. We denote this subspace as *space of feasible shapes*. The representations of the training shapes $\vec{C}_i$ in this low-dimensional space of feasible shapes are given by the following coordinate transformation:

$$\vec{w}_i = (\vec{C}_i - \bar{\vec{C}}) \tilde{\mathbf{C}}^T, \quad (2.8)$$

and the distribution of the training shapes in the m -dimensional subspace $\tilde{\mathbf{C}}$ is defined by the following multivariate Gaussian distribution:

$$P(\vec{w}) = \frac{1}{\sqrt{(2\pi)^m |\tilde{\Sigma}|}} \exp \left(-\frac{1}{2} \vec{w} \tilde{\Sigma}^{-1} \vec{w}^T \right), \quad (2.9)$$

where $\tilde{\Sigma}$ is a diagonal matrix composed of the m most important eigenvalues $\{\sigma_1^2, \dots, \sigma_m^2\}$ of the sample covariance matrix $\hat{\Sigma}$:

$$\tilde{\Sigma} = \begin{pmatrix} \sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_m^2 \end{pmatrix}. \quad (2.10)$$

The inverse transformation that transforms the training shapes from the low-dimensional subspace back to the original shape space also exists and is given as

$$\tilde{C}_i = \bar{\bar{C}} + \vec{w}_i \tilde{C}. \quad (2.11)$$

The space of feasible shapes must not necessarily be spanned by all eigenvectors of the sample covariance matrix. When we recall the definition of the PCA, we remind us that the linear PCA subspace is constructed in such a way that the more axes are added to the subspace, the more variance in the data can be explained. So, when all eigenvectors of the sample covariance matrix are used to define the linear PCA subspace, all variance in the training data is represented in the subspace. However, those eigenvectors which have only small eigenvalues most likely correspond to noise in the training data which should not be represented in the space of feasible shapes. So, one typically chooses only the $m < n - 1$ most important eigenvectors to define the space of feasible shapes. An example of the shape space can be seen in Fig. 2.4. All training shapes from Fig. 2.3 have been projected with the help of Eq. (2.8) onto the $m = 2$ most important modes of variation. The estimated Gaussian weight distribution is also shown.

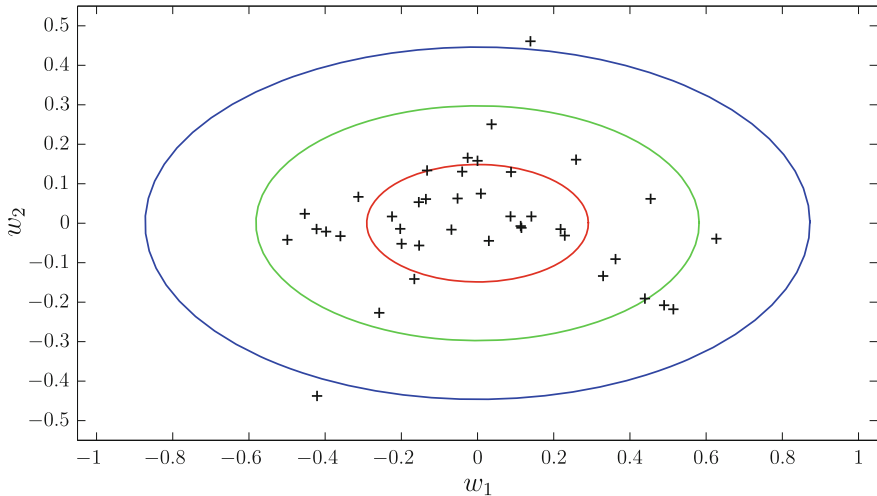


Fig. 2.4 Distribution of the training shapes (*black crosses*), projected onto the two main modes of variation. The ellipses at 1, 2, and 3 standard deviations of the corresponding Gaussian distribution are shown in *red, green, and blue*

In the case that fewer than available eigenvectors are used to define the shape space, Eq. (2.8) yields the orthogonal projection of the training shape C_i into the space of feasible shapes, i.e. the vector $\vec{w}_i$ denotes the point inside the low-dimensional space of feasible shapes which has the shortest euclidean distance to the original shape C_i :

$$\|(\vec{\bar{C}} + \vec{w}_i \vec{\bar{C}}) - \vec{C}_i\| \rightarrow \min . \quad (2.12)$$

There exist different criteria to determine which eigenvectors are important. Each eigenvector $\vec{\bar{C}}_i$ represents a certain amount of the total variance in the training shapes. It can be calculated with the help of the corresponding eigenvalue σ_i^2 to

$$\frac{\sigma_i^2}{\sum_{u=1}^{n-1} \sigma_u^2} , \quad (2.13)$$

where $\sum_{u=1}^{n-1} \sigma_u^2$ gives the total variance in the training shapes.

Consequently, one common way to define the shape space is to keep only as much eigenvectors until the accumulated variance of the m most important eigenvectors exceeds a fixed percentage (typically 90 to 98%) of the total variance in the training data [60]. The cumulative variance of the hand shapes can be seen in Fig. 2.5.

Another criterion is to search for an *elbow*, i.e. a characteristic drop, in the scree graph. In a scree graph, the abscissa gives the index i of the eigenvalue and the ordinate gives its value σ_i^2 . As the eigenvalues are given in descending order, the

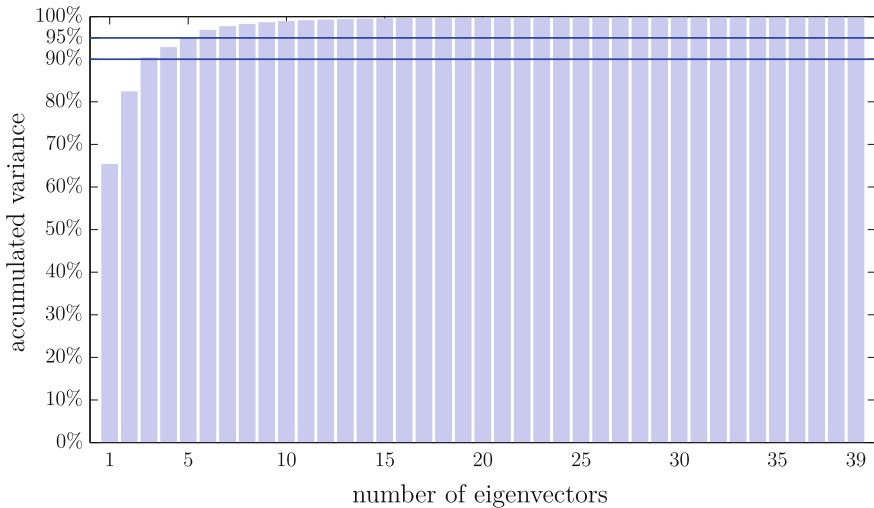


Fig. 2.5 Cumulative variance for a growing number of eigenvectors. Three eigenvectors are enough to account for 90% of the total variation and five eigenvectors capture roughly 95% of the total variation, respectively

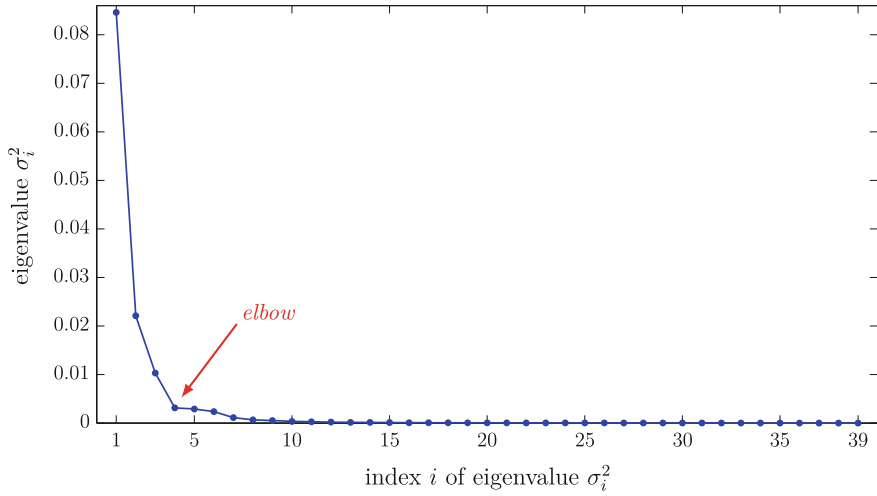


Fig. 2.6 Scree graph, showing the eigenvalues σ_i^2 in descending order of importance

scree graph is monotonically decreasing, where the decrease between the first few eigenvalues is typically larger than the decrease between the last few eigenvalues. Now, all eigenvalues left of and including the elbow are considered as important and the remaining eigenvalues are considered as unimportant. The basic idea of this is that the eigenvectors which correspond to noise in the training data are assumed to have approximately the same variance whereas the eigenvectors which correspond to meaningful variations are assumed to have different variances, respectively [66]. The scree graph for the hand shapes can be seen in Fig. 2.6.

Now, having identified the space of feasible shapes, the *Point Distribution Model* (PDM) by Cootes et al. is defined as

$$\vec{C}_{\text{PDM}}(\vec{w}) = \bar{\bar{C}} + \sum_{i=1}^m w_i \tilde{\bar{C}}_i, \quad (2.14)$$

i.e. all the modeled shapes $\vec{C}_{\text{PDM}}(\vec{w})$ are obtained as a linear combination of the m most important eigenvectors $\tilde{\bar{C}}_i$ of the sample covariance matrix (the *main modes of variation*) centered around the mean shape $\bar{\bar{C}}$. So, similar to the barycentric shape model in Eq. (2.3), the vector $\vec{w} = (w_1, \dots, w_m)$ is the parameter vector (or weight vector) of the model $\vec{C}_{\text{PDM}}(\vec{w})$ as it controls the influence that each main mode of variation has on the modeled shape. The shape variations that occur when varying the five most important shape weights are shown in Fig. 2.7.

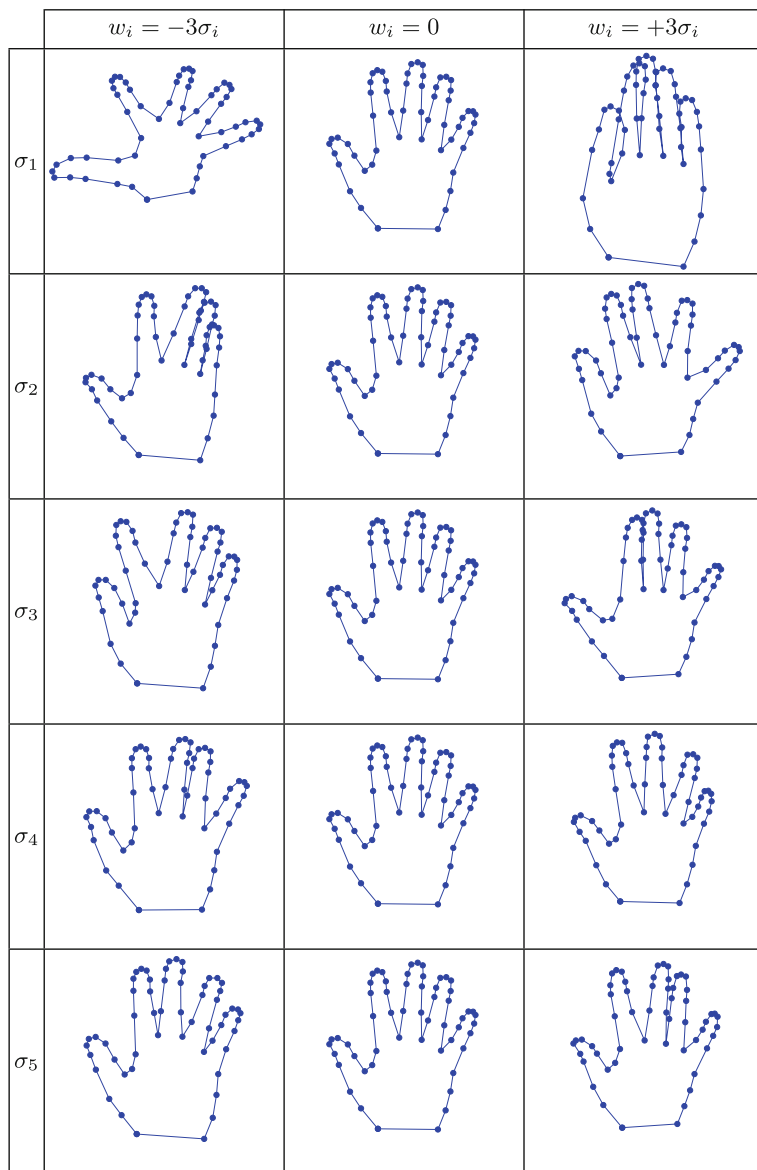


Fig. 2.7 Variation of the mean shape (*middle column*) by ± 3 standard deviations σ_i along the five most important eigenvectors (*left and right columns*)

In order to generate only likely shapes, i.e. shapes that correspond to the trained multivariate Gaussian distribution from Eq. (2.9), Cootes et al. additionally limit each weight w_i to be within three standard deviations of the Gaussian distribution:

$$-3\sigma_i \leq w_i \leq 3\sigma_i. \quad (2.15)$$

Another possibility is to constrain the weight vector $\vec{w}$ to lie inside the hyper-ellipsoid that defines the three standard-deviation boundary of the multivariate distribution [60]:

$$\sqrt{\sum_{i=1}^m \frac{w_i^2}{\sigma_i^2}} \leq 3. \quad (2.16)$$

Because of the facts that the shapes $\vec{C}_{\text{PDM}}(\vec{w})$ which are modeled by the PDM are obtained as linear combinations of the main modes of variation $\vec{\tilde{C}}_i$ and that a parameter vector (or weight vector) $\vec{w}$ is used to generate the shapes, the PDM is also called a *linear parametric statistical shape model* (linear parametric SSM). The construction of such a linear parametric SSM can be summarized as shown in Algorithm 2.1.

The m most important eigenvectors $\vec{\tilde{C}}_i$ of the sample covariance matrix together with the mean shape $\vec{\bar{C}}$ then define the low dimensional space of feasible shapes according to Eq. (2.14) and the corresponding eigenvalues σ_i^2 define the distribution of plausible shapes according to Eq. (2.9), respectively.²

2.3 An Implicit Linear Parametric Statistical Shape Model

Most of the currently used shape models are based on the PDM that has been defined in the last section [60]. This is most likely due to the fact that representing a shape as a set of points is very intuitive. Furthermore, the PDM is easy to understand and straightforward to implement. However, using points to describe a set of training shapes has also one big disadvantage: Dense point-correspondences need to be defined along the training shapes that are used to train the model. These dense point-correspondences may be determined manually or automatically. Yet, both approaches have their drawbacks. While the manual definition of point-correspondences is tedious and time-consuming, the automatic generation of point-correspondences is a difficult problem that is not easy to solve [60]. This contradicts the fact that wrong point-correspondences can cause strong artifacts in the shapes generated by the statistical shape model [98].

²Instead of explicitly setting up the covariance matrix, it is computationally more efficient to compute the *Singular Value Decomposition* (SVD) of the matrix $\vec{\tilde{C}}$, because the right-singular vectors of the matrix $\vec{\tilde{C}}$ are the eigenvectors of the matrix $\vec{\tilde{C}}^T \vec{\tilde{C}}$ [23, Chap. 4.6.3].

1. Start with a set of n rigidly-aligned (with regard to rotation, translation, and scale) training shapes, represented as dr -dimensional row vectors:

$$\{\vec{C}_1, \dots, \vec{C}_n\}, \text{ with } \vec{C}_i \in \mathbb{R}^{dr}. \quad (2.17)$$

2. Compute the sample mean of the training shapes:

$$\bar{\vec{C}} = \sum_{i=1}^n \vec{C}_i. \quad (2.18)$$

3. Subtract the sample mean from all training shapes in order to obtain a new set of zero-mean training shapes:

$$\{\check{\vec{C}}_1, \dots, \check{\vec{C}}_n\}, \text{ with } \check{\vec{C}}_i \in \mathbb{R}^{dr}. \quad (2.19)$$

4. Stack all mean-subtracted shape vectors $\check{\vec{C}}_i$ on top of each other in order to obtain the shape matrix $\check{\mathbf{C}} \in \mathbb{R}^{n \times dr}$, with $n \ll r$ and $d \in \{2, 3\}$:

$$\check{\mathbf{C}} = \begin{pmatrix} \check{\vec{C}}_1 \\ \vdots \\ \check{\vec{C}}_n \end{pmatrix}. \quad (2.20)$$

5. Estimate the sample covariance matrix $\hat{\Sigma} \in \mathbb{R}^{dr \times dr}$ of the mean-subtracted training shapes:

$$\hat{\Sigma} = \frac{1}{n-1} \check{\mathbf{C}}^T \check{\mathbf{C}}. \quad (2.21)$$

6. Compute the (at most) $n-1$ nonzero eigenvectors of the sample covariance matrix:

$$\{\check{\vec{C}}_1, \dots, \check{\vec{C}}_{n-1}\}, \text{ with } \check{\vec{C}}_i \in \mathbb{R}^{dr}. \quad (2.22)$$

Algorithm 2.1: Necessary steps to obtain an explicit linear parametric SSM.

So, in order to deal with this problem, Leventon et al. were the first to propose a statistical shape model which does not require any point-correspondences at all [77]. They chose to represent the training shapes C_i implicitly as the zero level set of a signed distance function Φ_i :

$$C_i = \{\vec{x} \mid \Phi_i(\vec{x}) = 0\}, \quad (2.23)$$

where the signed distance functions Φ_i are defined as in Eq. 1.33 (c.f. Sect. 1.4).

In general, the signed distance functions Φ_i are given as functions that map from a continuous domain $\Omega \subset \mathbb{R}^d$, with $d \in \{2, 3\}$, to the field of real numbers $\mathbb{R}$. However, in practical implementations, one typically uses sampled versions Φ_i^s of the continuous level set functions Φ_i . These functions $\Phi_i^s : \Omega^s \subset \mathbb{N}^d \rightarrow \mathbb{R}$ are defined on a discrete grid Ω^s , where $\Omega^s = \{1, \dots, j\} \times \{1, \dots, k\}$ for $d = 2$ and $\Omega^s = \{1, \dots, j\} \times \{1, \dots, k\} \times \{1, \dots, l\}$ for $d = 3$, respectively. See Fig. 1.2a for a 2D-example with $j = 512$ and $k = 366$.

Such two-dimensional discrete functions Φ_i^s can be represented as jk -dimensional row-vectors $\vec{\Phi}_i$ by successively appending the rows of the grid in a single vector:

$$\vec{\Phi}_i = (\Phi_i^s(1, 1), \dots, \Phi_i^s(j, 1), \dots, \Phi_i^s(1, k), \dots, \Phi_i^s(j, k)). \quad (2.24)$$

This can be extended to three-dimensional discrete functions Φ_i^s when we first vectorize each of the l two-dimensional slices to a jk -dimensional row-vector as explained above and then successively append the thus obtained vectors to a jkl -dimensional row-vector:

$$\vec{\Phi}_i = (\Phi_i^s(1, 1, 1), \dots, \Phi_i^s(j, 1, 1), \dots, \Phi_i^s(1, k, 1), \dots, \Phi_i^s(j, k, 1), \dots, \Phi_i^s(1, 1, l), \dots, \Phi_i^s(j, 1, l), \dots, \Phi_i^s(1, k, l), \dots, \Phi_i^s(j, k, l)). \quad (2.25)$$

1. Start with a set of n rigidly-aligned (with regard to rotation, translation, and scale) training shapes, implicitly represented as the zero level sets of discretely sampled signed-distance functions:

$$\{\Phi_1^s, \dots, \Phi_n^s\}, \text{ with } \Phi_i^s : \Omega^s \subset \mathbb{N}^d \rightarrow \mathbb{R}. \quad (2.26)$$

2. Convert the discrete functions Φ_i^s to jk - or jkl -dimensional row-vectors $\vec{\Phi}_i$ with the help of Eq. (2.24) (for $d = 2$) or Eq. (2.25) (for $d = 3$), respectively:

$$\{\vec{\Phi}_1, \dots, \vec{\Phi}_n\}, \text{ with } \begin{cases} \vec{\Phi}_i \in \mathbb{R}^{jk} & , \text{ if } d = 2 \\ \vec{\Phi}_i \in \mathbb{R}^{jkl} & , \text{ if } d = 3. \end{cases} \quad (2.27)$$

3. Compute the sample mean of the training shapes:

$$\bar{\vec{\Phi}} = \sum_{i=1}^n \vec{\Phi}_i. \quad (2.28)$$

4. Subtract the sample mean from all training shapes in order to obtain a new set of zero-mean training shapes:

$$\{\check{\vec{\Phi}}_1, \dots, \check{\vec{\Phi}}_n\}, \text{ with } \begin{cases} \check{\vec{\Phi}}_i \in \mathbb{R}^{jk} & , \text{ if } d = 2 \\ \check{\vec{\Phi}}_i \in \mathbb{R}^{jkl} & , \text{ if } d = 3. \end{cases} \quad (2.29)$$

5. Stack all mean-subtracted shape vectors $\check{\vec{\Phi}}_i$ on top of each other in order to obtain the shape matrix $\check{\Phi}$, where $n \ll jk < jkl$:

$$\check{\Phi} = \begin{pmatrix} \check{\vec{\Phi}}_1 \\ \vdots \\ \check{\vec{\Phi}}_n \end{pmatrix}, \text{ with } \begin{cases} \check{\Phi} \in \mathbb{R}^{n \times jk} & , \text{ if } d = 2 \\ \check{\Phi} \in \mathbb{R}^{n \times jkl} & , \text{ if } d = 3. \end{cases} \quad (2.30)$$

6. Estimate the sample covariance matrix $\hat{\Sigma}$ of the mean-subtracted training shapes:

$$\hat{\Sigma} = \frac{1}{n-1} \check{\Phi}^T \check{\Phi}, \text{ with } \begin{cases} \hat{\Sigma} \in \mathbb{R}^{jk \times jk} & , \text{ if } d = 2 \\ \hat{\Sigma} \in \mathbb{R}^{jkl \times jkl} & , \text{ if } d = 3. \end{cases} \quad (2.31)$$

7. Compute the (at most) $n-1$ nonzero eigenvectors of the sample covariance matrix (see also footnote 2 on page 31):

$$\{\tilde{\vec{\Phi}}_1, \dots, \tilde{\vec{\Phi}}_{n-1}\}, \text{ with } \tilde{\vec{\Phi}}_i \in \mathbb{R}^{dr}. \quad (2.32)$$

Algorithm 2.2: Necessary steps to obtain an implicit linear parametric SSM.

Now, with the help of Eqs. (2.24) and (2.25), one is able to transform the implicitly represented training shapes from Eq. (2.23) into a vectorial representation. This enables us to build a statistical shape model with the help of the implicit training shapes, exactly as explained in Sect. 2.2.

So, the construction of an implicit linear parametric SSM can be summarized as shown in Algorithm 2.2.

Like for the explicit SSM from Sect. 2.2, the m most important eigenvectors of the sample covariance matrix $\tilde{\tilde{\Phi}}_i$ together with the mean shape $\tilde{\tilde{\Phi}}$ define the low dimensional space of feasible shapes according to

$$\vec{\Phi}_{\text{glob}}(\vec{w}) = \tilde{\tilde{\Phi}} + \sum_{i=1}^m w_i \tilde{\tilde{\Phi}}_i, \quad (2.33)$$

with $\vec{w} = (w_1, \dots, w_m)$, and the corresponding eigenvalues σ_i^2 define the distribution of plausible shapes according to Eq. (2.9), respectively. By inverting the steps which were necessary to vectorize the training shapes (Eq. (2.24) or Eq. (2.25), respectively), and by considering the level sets again as continuous functions, one obtains a shape model which is able to generate shapes that are implicitly represented through a higher-dimensional function $\vec{\Phi}_{\text{glob}}(\vec{w}) : \Omega \subset \mathbb{R}^d \rightarrow \mathbb{R}$:

$$\Phi_{\text{glob}}(\vec{w}) = \tilde{\tilde{\Phi}} + \sum_{i=1}^m w_i \tilde{\tilde{\Phi}}_i. \quad (2.34)$$

As for the PDM from Sect. 2.2, all modeled shapes are obtained as linear combinations of the main modes of variation $\tilde{\tilde{\Phi}}_i$ and a parameter vector (or weight vector) $\vec{w}$ is used to generate the shapes, hence the term *implicit linear parametric statistical shape model*. Explicit representations of the generated shapes can be obtained as the zero level set of the higher dimensional function $\Phi_{\text{glob}}(\vec{w})$:

$$C_{\text{glob}}(\vec{w}) = \left\{ \vec{x} \mid \vec{\Phi}_{\text{glob}}(\vec{w})(\vec{x}) = 0 \right\}. \quad (2.35)$$

A schematic overview of the training process can be seen in Fig. 2.8. In the depicted setup, the training set consists of four shapes, each having a spatial resolution of 3×3 pixels, and the two main modes of variation are used to define the linear SSM subspace.

An example for the just described implicit linear parametric statistical shape model can be seen in Figs. 2.9, 2.10, 2.11, 2.12, 2.13 and 2.14. The statistical shape model that is shown describes the variation of the outer bony boundary which surrounds the nasal cavity and the paranasal sinuses. The used training shapes are depicted in appendix A (green line). A more detailed description of the dataset follows later in Sect. 4.1.1.

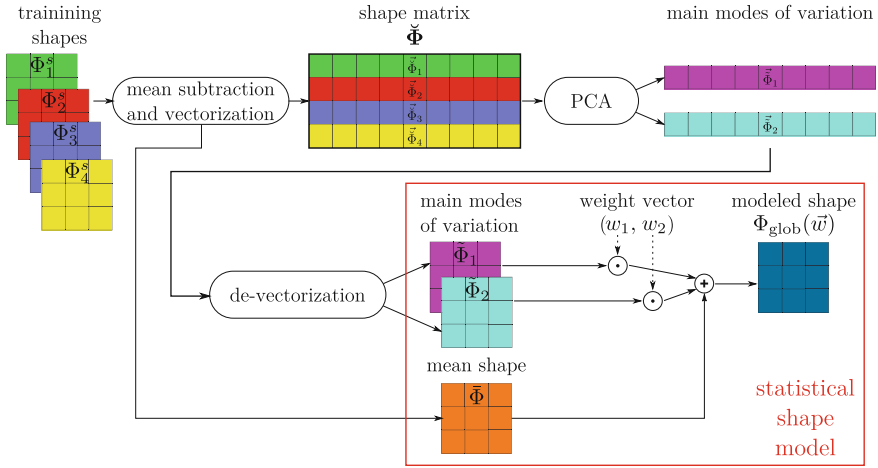


Fig. 2.8 Exemplary schematic overview of the training process of the implicit linear parametric statistical shape model from Eq. (2.34)

In Fig. 2.9, the first four main modes of variation are depicted (without addition of the mean shape). They nicely show where most of the variation occurs in the training shapes. It is clearly visible that the first mode of variation mainly encodes the variation in the frontal sinuses as these differ the most between different people (see also Fig. 4.2). The second mode of variation encodes mainly the lower expansion of the paranasal sinuses, the third allows to balance the expansion of the frontal sinuses, the fourth encodes amongst others the upper extension of the maxillary sinuses, and so on.

What can also be seen is that the maximum amplitude of the main modes of variation $\tilde{\Phi}_i$ quickly decreases with growing index i . This is due to the strong decrease of the corresponding eigenvalues σ_i^2 . The scree graph from Fig. 2.10 and the cumulative variance from Fig. 2.11 confirm this finding. It can be seen that the first five main modes of variation already account for more than 90% of the total variation which is inherent in the training data and the first ten main modes of variation account for more than 97% of the total variation, respectively.

In Fig. 2.12, one can see the shape variations that occur when the mean shape $\bar{\Phi}$ is varied by plus and minus three standard deviations along the five main modes of variation $\tilde{\Phi}_1, \dots, \tilde{\Phi}_5$. The level set functions that represent the modeled shapes are color-coded, where blue tones are assigned to negative values of the level set functions. The zero level sets are additionally shown as black curves.

The distribution of the training shapes, projected onto the first three main modes of variation, is shown in Figs. 2.13 and 2.14. It can be seen that a Gaussian distribution is a reasonable assumption to describe the distribution of the training shapes.

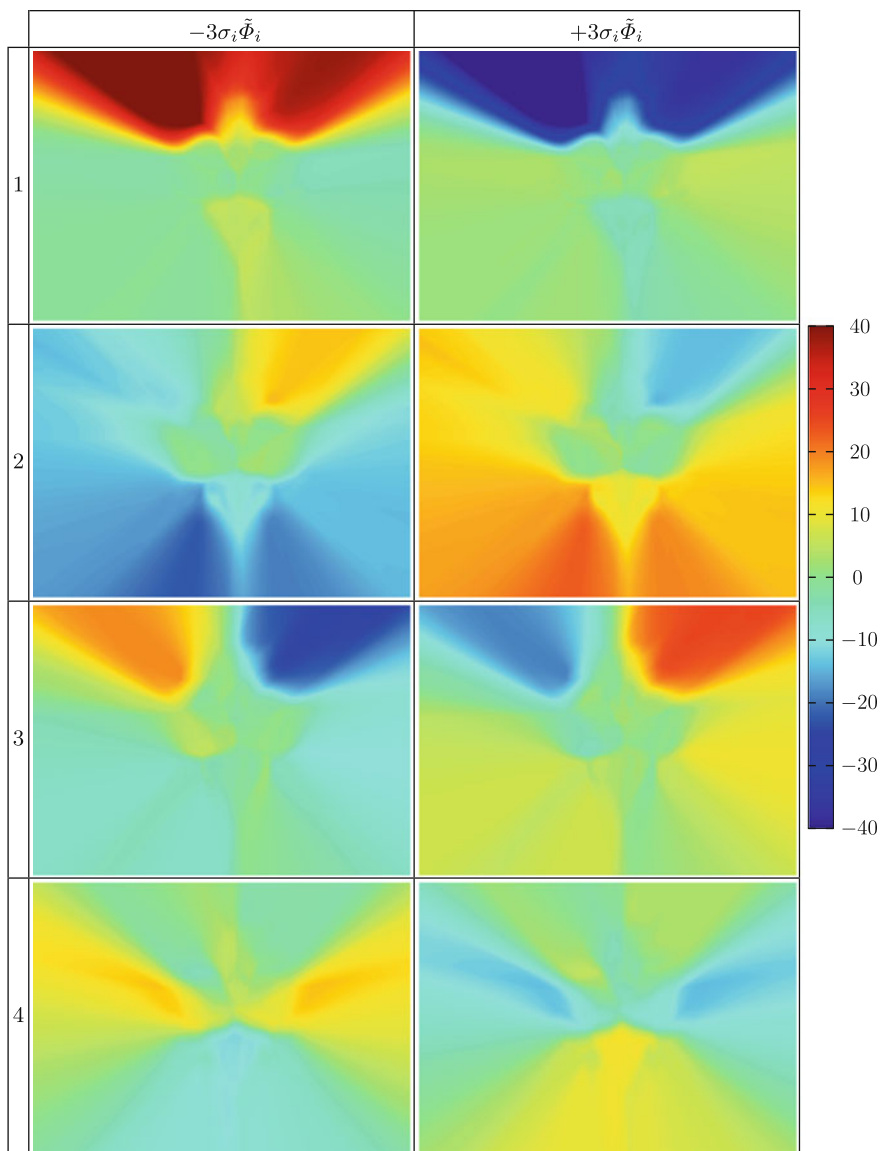


Fig. 2.9 First four main modes of variation of the implicit training shapes

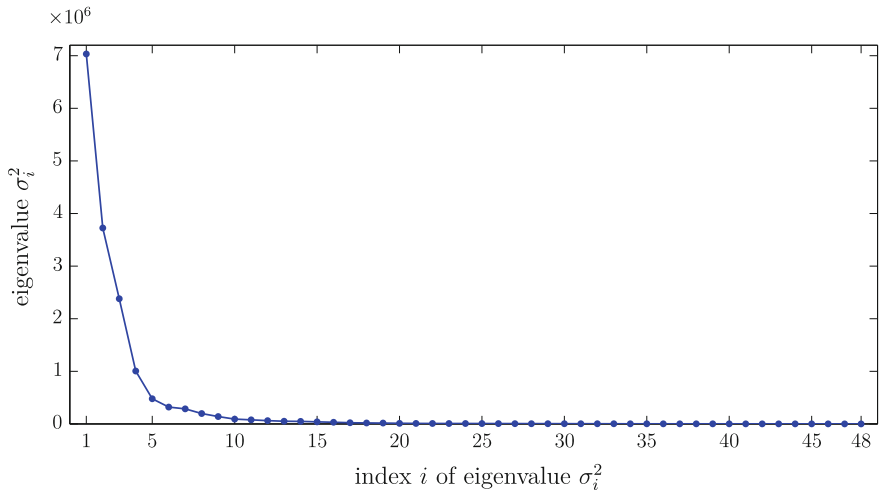


Fig. 2.10 Scree graph, showing the eigenvalues σ_i^2 in descending order of importance

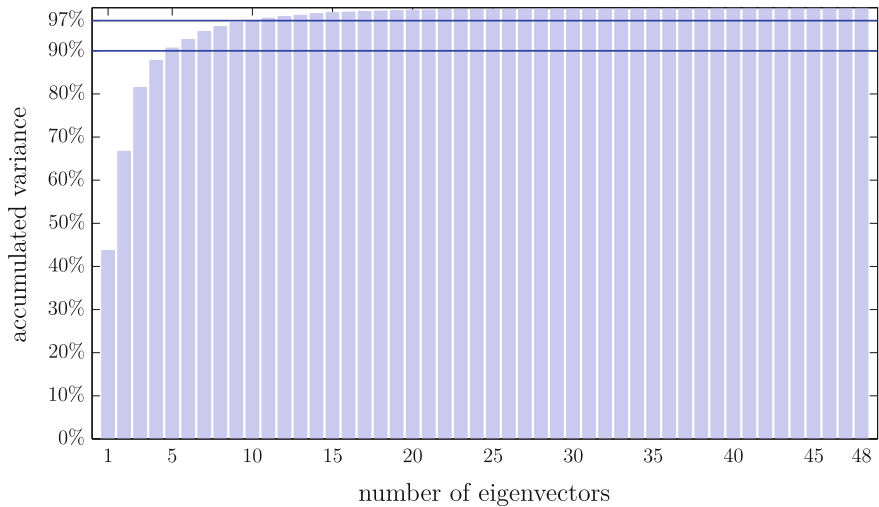


Fig. 2.11 Cumulative variance for a growing number of eigenvectors. Five eigenvectors are enough to account for 90% of the total variation and ten eigenvectors capture more than 97% of the total variation, respectively

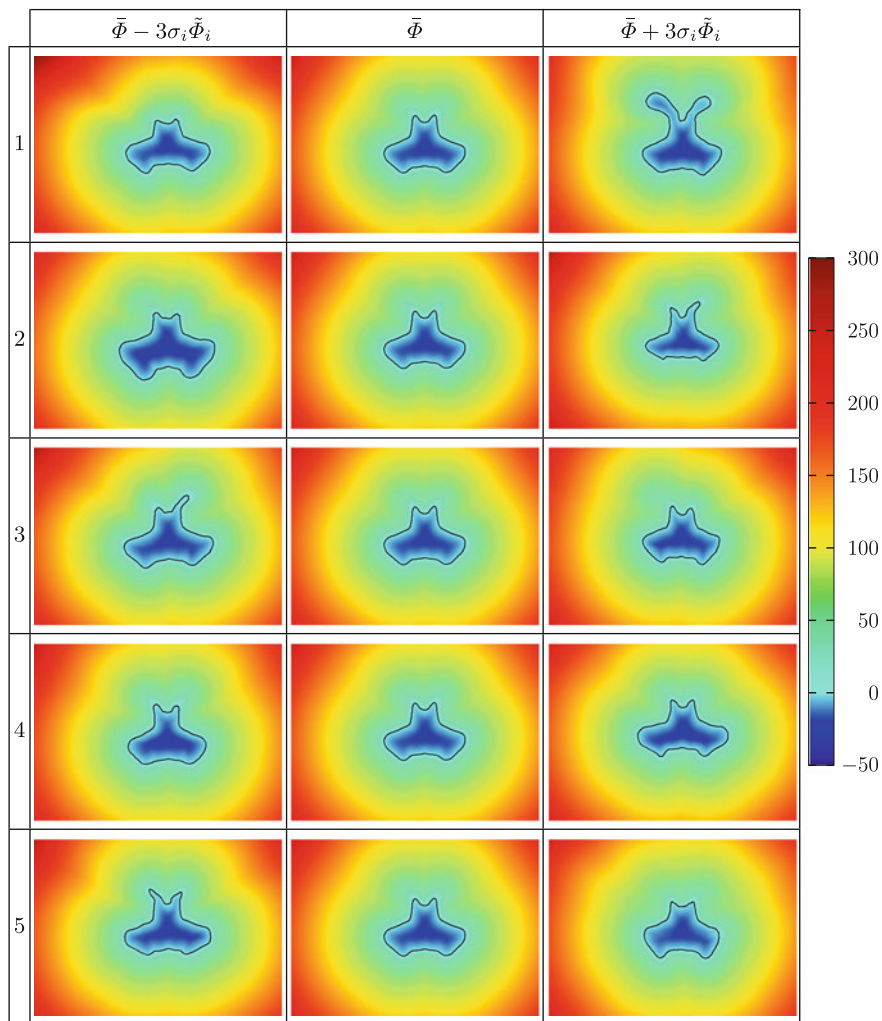


Fig. 2.12 Variation of the mean shape $\bar{\Phi}$ (middle column) by ± 3 standard deviations σ_i along the five main modes of variation $\tilde{\Phi}_i$ (left and right columns). The zero level set is shown as a black line

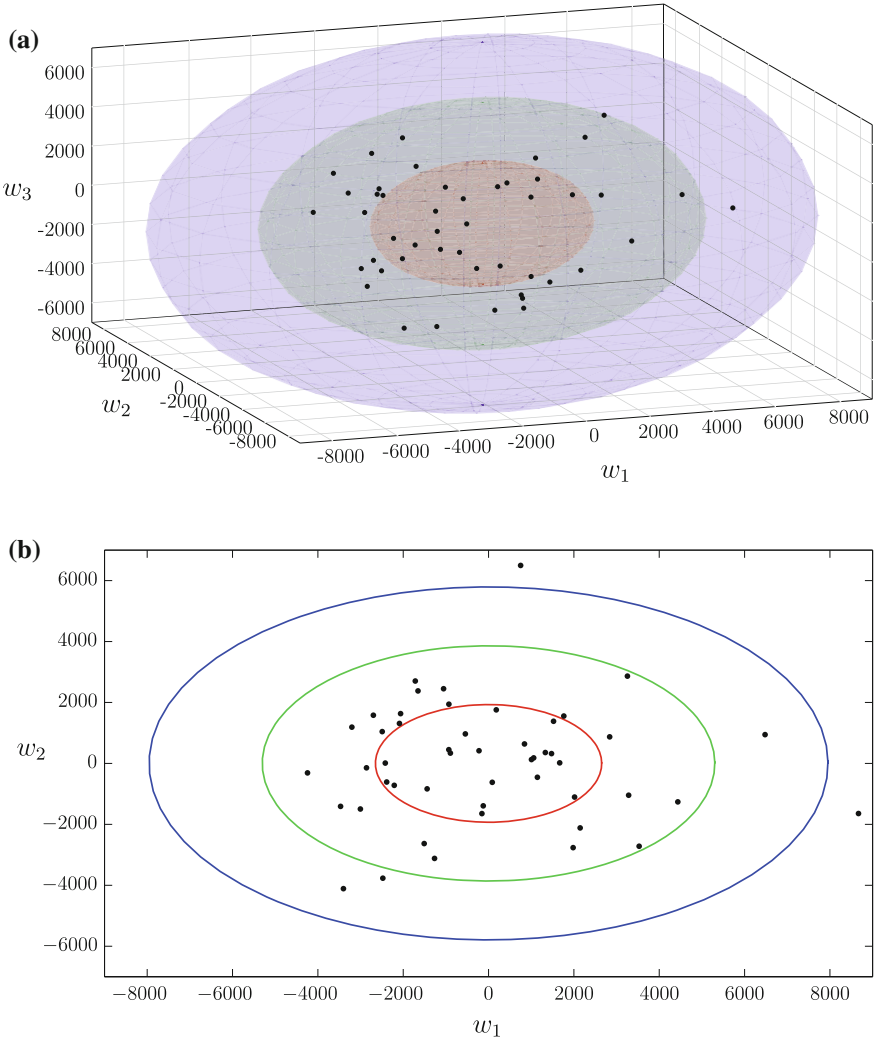


Fig. 2.13 **a** Distribution of the implicit training shapes (black dots), projected onto the first three main modes of variation. The ellipses at 1, 2, and 3 standard deviations of the estimated Gaussian distribution are shown in red, green, and blue, respectively. **b** Two-dimensional projection of the plot shown in (a) onto the first versus second main mode of variation

2.4 Rigid Shape Alignment

In the previous sections, we have demanded that the training shapes are aligned to each other with regard to rotation, translation, and scale. According to the definition in Sect. 2.1, this information does not belong to the shape information of a specific shape class.

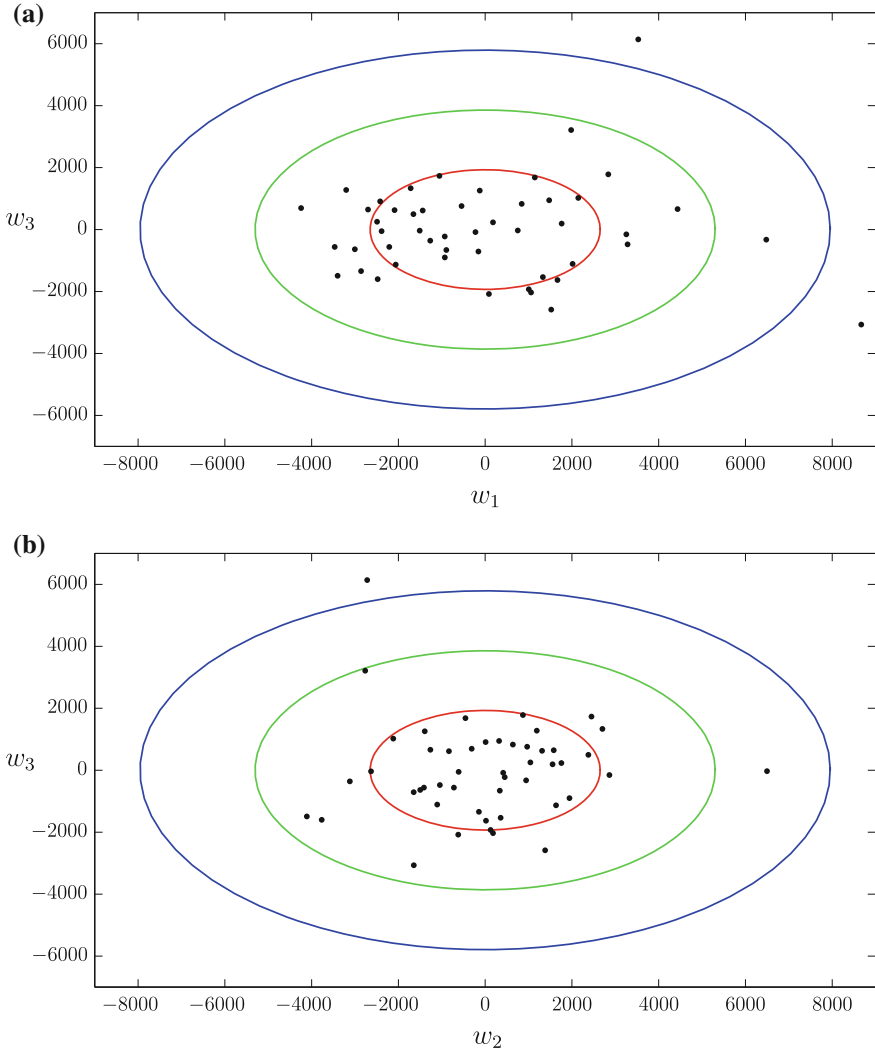


Fig. 2.14 Two-dimensional projections of the plot shown in Fig. 2.13a onto the first versus third (a) and second versus third (b) main mode of variation

So, in the following we will discuss how such a *rigid alignment* can be obtained. For this purpose, we will first define the so-called similarity transformation in Sect. 2.4.1, then we discuss the rigid alignment of two shapes in Sect. 2.4.2, and finally in Sect. 2.4.3 we give an algorithm for aligning multiple shapes.

2.4.1 Similarity Transformation

We remember the explicit shape representation from Eq. (2.1) or from Eq. (2.2), respectively:

$$\vec{C} = \begin{cases} (x_1, y_1, \dots, x_r, y_r) & , \text{ if } d = 2 \\ (x_1, y_1, z_1, \dots, x_r, y_r, z_r) & , \text{ if } d = 3. \end{cases} \quad (2.36)$$

This explicit representation can also be denoted by an $r \times d$ -dimensional matrix $\mathbf{X}$ when we write the individual points (x_i, y_i) or (x_i, y_i, z_i) row-wise one above the other:

$$\mathbf{X} = \begin{pmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_r \end{pmatrix}, \quad (2.37)$$

with $\vec{x}_i = (x_i, y_i)$ for $d = 2$ or $\vec{x}_i = (x_i, y_i, z_i)$ for $d = 3$, respectively.

The matrix $\mathbf{X}$ is also called a *configuration matrix* [64, Definition 2.1]. Now, the *Euclidean similarity transformations* of a configuration matrix $\mathbf{X}$ are defined as all the matrices $\mathbf{X}'$ that can be obtained by translating, rotating, and isotropically scaling the matrix $\mathbf{X}$ [64, Definition 3.2]:

$$\mathbf{X}' = \beta \mathbf{X} \Gamma + \vec{1}_r^T \vec{\gamma}, \quad (2.38)$$

where $\beta \in \mathbb{R}^+$ is a positive scale factor, $\Gamma \in SO(d)$ is an orthogonal rotation matrix, $\vec{\gamma} \in \mathbb{R}^d$ is a d -dimensional translation vector, and $\vec{1}_r$ is a r -dimensional row-vector whose entries are all equal to one.

2.4.2 Rigid Alignment of Two Shape Configurations

In order to rigidly align a shape configuration $\mathbf{X}^{(2)}$ to a shape configuration $\mathbf{X}^{(1)}$ with regard to rotation, translation, and scale, we need to determine the Euclidean similarity transformation that minimizes the squared Euclidean distance between both configurations:

$$\left[\hat{\beta}, \hat{\Gamma}, \hat{\vec{\gamma}} \right] = \arg \min_{\beta, \Gamma, \vec{\gamma}} \|\mathbf{X}^{(1)} - \beta \mathbf{X}^{(2)} \Gamma - \vec{1}_r^T \vec{\gamma}\|^2 \quad (2.39)$$

1. Compute the centroid of each configuration $\mathbf{X}^{(k)}$, $k \in \{1, 2\}$, as

$$\bar{\mathbf{x}}^{(k)} = \frac{1}{r} \sum_{i=1}^r \vec{x}_i^{(k)}. \quad (2.40)$$

2. Center the configurations via

$$\begin{aligned} \tilde{\mathbf{X}}^{(k)} &= \mathbf{X}^{(k)} - \bar{\mathbf{1}}_r^T \bar{\mathbf{x}}^{(k)} \\ &= \begin{pmatrix} \vec{x}_1^{(k)} \\ \vdots \\ \vec{x}_r^{(k)} \end{pmatrix} - \begin{pmatrix} \bar{x}^{(k)} \\ \vdots \\ \bar{x}^{(k)} \end{pmatrix}. \end{aligned} \quad (2.41)$$

3. Compute the sample covariance matrix of the mean-subtracted configurations $\tilde{\mathbf{X}}^{(1)}$ and $\tilde{\mathbf{X}}^{(2)}$ (Eq. (D.16)):

$$\begin{aligned} \hat{\Sigma} &= \frac{1}{r-1} (\tilde{\mathbf{X}}^{(1)})^T \tilde{\mathbf{X}}^{(2)} \\ &= \frac{1}{r-1} \sum_{i=1}^r (\vec{x}_i^{(1)})^T \vec{x}_i^{(2)}. \end{aligned} \quad (2.42)$$

4. Compute a *Singular Value Decomposition* (SVD) of the sample covariance matrix [23, Chap. 4.6.3]:

$$\hat{\Sigma} = \mathbf{U} \Lambda \mathbf{V}^T, \quad (2.43)$$

with $\mathbf{U}, \mathbf{V} \in O(d)$ being orthogonal matrices and Λ is a $d \times d$ diagonal matrix of positive elements, the so-called *singular values*.

5. Obtain the rotation estimate via

$$\hat{\Gamma} = \mathbf{V} \begin{pmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \det(\mathbf{U}\mathbf{V}) \end{pmatrix} \mathbf{U}^T, \quad (2.44)$$

where $\det(\mathbf{V}\mathbf{U}) \in \{-1, +1\}$.

6. Estimate the scale factor via

$$\hat{\beta} = \frac{\text{tr}(\hat{\Sigma} \hat{\Gamma})}{\frac{1}{r-1} \text{tr}((\tilde{\mathbf{X}}^{(2)})^T \tilde{\mathbf{X}}^{(2)})}, \quad (2.45)$$

where $\text{tr}(\mathbf{A})$ gives the trace of a square matrix $\mathbf{A}$.

7. Obtain the translation estimate via

$$\hat{\gamma} = \bar{\mathbf{x}}^{(1)} - \hat{\beta} \bar{\mathbf{x}}^{(2)} \hat{\Gamma}. \quad (2.46)$$

Algorithm 2.3: Necessary steps to perform a full ordinary Procrustes analysis.

This means, we need to estimate a scale factor $\hat{\beta}$, a rotation matrix $\hat{\Gamma}$, and a translation vector $\hat{\gamma}$. This problem is also known as *full ordinary Procrustes analysis* [64, Definition 7.1]. A solution to problem (2.39) can be obtained with the approach³ in Algorithm 2.3 ([64, result 7.1] or [47, Chap. 9.4.1]).

2.4.3 Rigid Alignment of Multiple Shape Configurations

In general we have more than two shape configurations that need to be rigidly aligned to each other. So, given $n \geq 2$ shape configurations, e.g. the training shapes for the PDM from Sect. 2.2, Eq. (2.39) modifies to

$$\begin{bmatrix} \hat{\beta}_1, \dots, \hat{\beta}_n \\ \hat{\mathbf{r}}_1, \dots, \hat{\mathbf{r}}_n \\ \hat{\gamma}_1, \dots, \hat{\gamma}_n \end{bmatrix} = \arg \min_{\substack{\beta_1, \dots, \beta_n \\ \mathbf{r}_1, \dots, \mathbf{r}_n \\ \gamma_1, \dots, \gamma_n}} \frac{1}{n} \sum_{i=1}^n \sum_{j=i+1}^n \left\| \left(\beta_i \mathbf{X}^{(i)} \mathbf{r}_i + \vec{1}_r^T \vec{\gamma}_i \right) - \left(\beta_j \mathbf{X}^{(j)} \mathbf{r}_j + \vec{1}_r^T \vec{\gamma}_j \right) \right\|^2 \quad (2.47)$$

This problem is also known as *full generalized Procrustes analysis* [64, Definition 7.4]. A solution to problem (2.47) can be obtained with the simple Algorithm 2.4 [34].

Step 4 in Algorithm 2.4 is needed in order to prevent the mean from shrinking, rotating or drifting [34]. This would happen, because Eq. (2.47) defines an under-determined system of equations that has no unique solution if the mean is not constrained. Algorithm 2.4 usually converges quickly so that already one iteration provides a very good alignment result. A set of hand shapes that have been rigidly aligned with Algorithm 2.4 can be seen in Fig. 2.3b.

Algorithm 2.4 is simple and it works in most cases, however its convergence has not been formally proven. Another slightly more difficult algorithm with investigated convergence bounds can be found in [64, chap. 7.4.1]. For two-dimensional shapes there also exists a closed-form solution for the mean configuration from Eq. (2.47) so that only step 5 of Algorithm 2.4 needs to be executed once in order to obtain the rigidly aligned shapes.

³The construction in step 5 of Algorithm 2.3 ensures that $\hat{\Gamma} \in SO(d)$, i.e. $\hat{\Gamma}$ is a rotation matrix (an orthogonal matrix with determinant 1). In the case that reflections should be also allowed, i.e. $\hat{\Gamma} \in O(d)$ (an orthogonal matrix with determinant 1 or -1), Eq. (2.44) reduces to $\hat{\Gamma} = \mathbf{V}\mathbf{U}^T$ [64, Chap. 7.2.1].

The here presented approach works for the implicit shape representation from Eq. (1.32) as well when at least $r = 2$ (for $d = 2$) or $r = 3$ (for $d = 3$)⁴ explicit corresponding points are additionally available on the implicit shapes. These points can e.g. be manually selected [99]. Other approaches try to align the implicit shapes by maximizing the mutual overlap of the interior regions (blue region in Fig. 1.2a) [130] or by minimizing the sum of squared differences between the signed distance functions [95] with the help of variational methods as introduced in Sect. 1.3.

1. Rigidly align each shape configuration $\mathbf{X}^{(i)}$, $2 \leq i \leq n$, to the first shape configuration $\mathbf{X}^{(1)}$ with the help of Algorithm 2.3 in order to obtain initial estimates for $\hat{\beta}_i$, $\hat{\Gamma}_i$, and $\hat{\gamma}_i$, $2 \leq i \leq n$.
2. Set $\hat{\beta}_1 = 1$, $\hat{\Gamma}_1 = \mathbf{I}_d$, and $\hat{\gamma}_1 = \vec{0}_d$, where $\mathbf{I}_d$ is the $d \times d$ identity matrix and $\vec{0}_d$ is a d -dimensional row-vector containing only zeros.

repeat

3. Compute the mean of the similarity transformed shape configurations

$$\bar{\mathbf{X}} = \frac{1}{n} \sum_{i=1}^n \mathbf{X}^{(i)'} = \frac{1}{n} \sum_{i=1}^n \left(\beta_i \mathbf{X}^{(i)} \Gamma_i + \vec{1}_r^T \vec{\gamma}_i \right). \quad (2.48)$$

4. Rigidly align the mean configuration $\bar{\mathbf{X}}$ to the first shape configuration $\mathbf{X}^{(1)}$ with the help of Algorithm 2.3 in order to obtain an adjusted mean $\bar{\mathbf{X}}'$.
5. Rigidly align each similarity transformed shape configuration $\mathbf{X}^{(i)'}$, $1 \leq i \leq n$, to the adjusted mean configuration $\bar{\mathbf{X}}'$ with the help of Algorithm 2.3 in order to update the estimates for $\hat{\beta}_i$, $\hat{\Gamma}_i$, and $\hat{\gamma}_i$, $1 \leq i \leq n$.

until convergence

Algorithm 2.4: Simple procedure to perform a full generalized Procrustes analysis.

2.5 Problem: Limited Training Data

The statistical shape models from Eqs. (2.14) and (2.34) can provide an important means in the solution of segmentation problems or in the reconstruction of incomplete shapes. However, one major problem of statistical shape models is to obtain a sufficient amount of training data. Heimann and Meinzer have nicely summarized the problem of limited training shapes in [60, p. 546]:

The power of a statistical model rises and falls with the quantity of available training data. In case of 3D SSMs, this quantity is almost always too low, ...⁵

⁴The 2D similarity transformation has 4 degrees of freedom, and the 3D similarity transformation has 7 degrees of freedom, respectively.

⁵Reprinted from [60, p. 546], © 2009, with permission from Elsevier.

The reason why there are almost always too few training shapes is that they have to be manually extracted from given image data which is a tedious and time-consuming task. For example, the manual extraction of the paranasal sinuses from CT data takes about 900 min for a dataset which consists of 98 slices, each having a resolution of 512×512 pixels [128].

The limited training data is a problem since for the above mentioned statistical shape models the assumption has been made that all plausible shapes can be described by linear combinations of the training shapes. Now, for a limited number of training shapes, this assumption is too restrictive. As can be seen in Eqs. (2.22) and 2.23 the dimension of the shape space cannot exceed the number of the training shapes minus one. The actual dimension will most likely be even smaller because the maximum dimension is only achieved for linearly independent training shapes (compare also the scree graphs from Figs. 2.6 and 2.10). So, for a small number of training shapes, the dimension of the shape space will most probably be too small to capture the full amount of variation inside a specific shape class. Much effort has therefore been spent on making statistical models more flexible.

In the following, we will discuss the most important previous contributions by other researchers which aim at obtaining more flexible SSMs. There are basically three different approaches to address the problem of limited training data. The approaches can be divided in three categories, but their boundaries are fluid [33]:

1. Artificially enlarge the shape variations which are described by the model.
2. Relax the model-constraints and locally refine the segmentation result.
3. Partition the model and describe the individual segments independently.

An overview of the approaches discussed below can be seen in Table 2.1. The approaches [22, 27, 39, 108, 109] are based on the variational level set image seg-

Table 2.1 Overview of existing approaches which aim at obtaining more flexible SSMs. Explicit approaches are given in black and implicit approaches in blue

Artificial enlargement of shape variations	Relaxation of model-constraints	Model partitioning
Cootes and Taylor [28, 31]	Cootes and Taylor [29]	Blanz and Vetter [19]
Cootes and Taylor [29]	Wang and Staib [133, 135]	de Bruijne et al [24]
Wang and Staib [134]	Shen et al. [117, 118]	Roberts et al. [104]
de Bruijne et al. [24]	Cootes and Taylor [30]	Davatzikos et al. [43]
Shen et al. [117, 118]	Weese et al. [136]	Zhao et al. [141]
Taron et al. [125]	Chen et al. [27]	Nain et al. [92]
Loog [80]	Bresson et al. [22]	Rousson and Paragios [108]
Koikkalainen et al. [72]	Shang and Dössel [116]	Knothe [71]
Lüthi et al. [83]	Rousson et al. [109]	Amberg et al. [11, 82]
Jud and Vetter [67, 68]	Cremers et al. [39]	
	Rousson and Paragios [108]	

mentation framework. We will discuss them extensively later in Sect. 5.2. The other approaches will be shortly discussed in the following.

2.5.1 Artificial Enlargement of Shape Variations

An early attempt to artificially enlarge the shape variations has been made by Cootes and Taylor [28, 31]. Instead of considering each shape as a rigid object, they assigned them additional elastic properties described by a stiffness matrix. Cootes and Taylor then performed a PCA on the stiffness matrix in order to obtain the resonant modes of vibration of each shape. This leads to a so-called *vibrational model* for each shape that is of the same form as the PDM from Eq. (2.14). However, the variation modes of these vibrational models are defined by the eigenvectors of the stiffness matrix and not by the training shapes. One can use the vibrational models in order to enlarge the number of training shapes by creating a set of vibrationally-deformed training shapes from each real training shape. The extension of the training set by these vibrationally-deformed training shapes leads to an increase of the estimated sample covariance along the resonant modes of vibration, thus enhancing the flexibility of the trained shape model in these directions.

Later, Cootes and Taylor simplified their approach by directly modifying some elements of the sample covariance matrix of the training shapes [29]. They proposed to add a constant value to the diagonal elements, representing the variance of a certain point, and to those off-diagonal elements that represent the covariance of neighboring points. The extra variance gives the individual points more freedom to move while the additional covariance also ensures that neighboring elements tend to move together. Consequently, smooth shape variations are favored.

A very similar approach has also been proposed by Wang and Staib [134]. They additionally introduced a weighting factor that tunes the influence of the smooth shape variations on the total variation. When no training samples are available their formulation relies completely on the smooth shape variations, and with an increasing number of training samples it iteratively adapts to purely trained shape-driven variations. Another very similar approach has been proposed by de Bruijne et al. [24]. In their approach the components of the artificial deformations are decoupled so that there exist no dependencies between different spatial dimensions. The decoupling has the effect that one does not need to establish the full covariance matrix, which has very large memory requirements especially for 3D shapes. The approach by de Bruijne et al. works best when the considered object class is very elongated in one spatial dimension in relation to the other dimensions.

All above-mentioned approaches make use of a synthetic covariance matrix that enhances the shape variations by additionally allowing smooth deformations. A different approach has been made by Shen et al. [117, 118]. They introduced the so-called *Active Focus Deformable Statistical Shape Model* (AFDSM). The AFDSM can *focus* on important structures by multiplying the subset of boundary points that define those structures with weighting factors greater one. By doing this, the vari-

ance of the training shapes is increased in the direction of the important structures. After PCA, this results in a modified shape space where the main modes of variation correspond to variations of the important structures. The structures can be determined manually or they can be chosen to those boundary points which correlate with strong image information (e.g. large gradient magnitudes). Subsequently, Koikkalainen et al. [72] modified this approach by smoothly deforming randomly selected local patches of the training shapes with the help of a so called *deformation sphere* [84] in order to enlarge the variations in the training set.

Other Approaches have been proposed by Taron et al. [125] and Loog [80]. Taron et al. [125] tried to include the uncertainties, that remain after establishing the point correspondences, into the model building process. This is achieved by modeling the uncertainties of the control points as a multivariate Gaussian distribution and generating new, artificial shapes by sampling from this distribution. Loog [80] proposed to fill the training shapes' covariance matrix only with those elements that correspond to the k nearest neighbors of each point and to fill the other entries with the help of a maximum entropy approach. By doing this, they model only the variations between nearby points and assume the other points as independent as possible.⁶

All so far discussed approaches rely on the PDM from Eq. (2.34). Recently, Lüthi et al. [83] proposed an extension of the approaches from [29] and [134] to those shape models where the modes of variation are given by continuous functions instead of discrete vectors, i.e. that are of the same form as the model from Eq. (2.34).⁷ For this purpose, they proposed to describe the variability of the shape model and the extra variability as a combined Gaussian process, which is a multivariate Gaussian distribution of infinitely many random variables, and to use the Nyström approximation in order to obtain a low-rank approximation of this process. By doing this, they obtain a new set of continuous eigenfunctions that again represent the trained shape variations as well as the extra flexibility. Subsequently, Jud and Vetter [67, 68] extended this approach by multiplying the covariance function of the shape model's Gaussian process by a covariance function which decreases exponentially with growing distance between two points on the training shapes. This has the effect that distant shape parts are decoupled which also leads to more flexibility in the model. In fact, the basic idea behind the approach by Jud and Vetter, to decouple distant parts of the training shapes, is very similar to our idea which we are going to present in Chap. 3. However, the benefit of our approach is that globally consistent deformations can still be modeled whereas they are completely eliminated in the approach by Jud and Vetter.

⁶The maximum entropy approach is necessary to obtain a valid covariance matrix. Simply setting all remaining elements to zero, which means to assume that the corresponding points are uncorrelated, does not result in a valid covariance matrix.

⁷In the work of Lüthi et al. smooth deformation fields are modeled instead of level set functions.

2.5.2 *Relaxation of Model-Constraints*

Another approach to cope with a limited amount of training data is to use the statistical shape model only to robustly find a coarse initial solution and to relax the model-constraints afterwards. This means that also those shapes are allowed in the final solution which cannot be accurately approximated by the statistical shape model. Like for the artificially enhanced shape models, an early attempt in this direction has again been made by Cootes and Taylor. After fitting their already artificially enhanced PDM from [29], they search along lines normal to the model boundary in order to find a new set of candidate points which match a previously trained intensity distribution. They then select those points from the candidate set as final segmentation result that minimize the residual error to the best model fit. Later, they extended their approach by learning the needed local offsets based on the observed gray level differences between the optimal segmentation result and the best global model fit [30]. Similar approaches have also been proposed by Shen et al. [117, 118] and Shang and Dössel [116]. They both search for candidate points in regions with large gradient magnitudes. Wang and Staib [133, 135] proposed a physical model-based image registration approach that incorporates a statistical shape model. In their approach, they first compute the best fit of the statistical shape model and then use the boundary points of the deformed model shape as landmarks in their image registration framework. At the landmark positions, they constrain the deformation vectors of their deformation field to match the vector difference between the points on the mean shape and the deformed model shape. Similar to Cootes and Taylor, they also mentioned that using the enhanced PDM from [134] could further improve the results.

In the previously mentioned approaches, the segmentation result is still bound to the subspace of feasible shapes during the model fitting process. The common approach is to iteratively estimate a shape candidate which is then projected back into the model space. The model-constraints are relaxed only after the last iteration of the adaptation process, when the best model fit has already been found. Weese et al. [136] proposed to relax the model-constraints not only after but already during the model fitting process. For this purpose, they used an energy function which consists of a weighted combination of an external energy and an internal energy. The external energy drives the shape towards nearby gradients. Simultaneously, the distances between neighboring points on the deformed shape are compared to the distances between neighboring points on the best model approximation. These distances should be equal for a deformed shape that resembles the modeled shape. So, the internal energy penalizes the squared difference of these two distances. Weese et al. minimized their energy with the help of the conjugate gradient method in order to obtain a segmentation result which captures the image boundaries as good as possible but which resides also close to the subspace of feasible shapes. Many level set approaches with relaxed model-constraints (e.g. [22, 27, 39, 108]) are based on the same idea. They will be discussed in Sect. 5.2.

2.5.3 *Model Partitioning*

All the above-mentioned approaches enhance the flexibility of statistical shape models. However, their main disadvantage is that they allow variations which cannot be explained by the training shapes. A third way to address the problem of limited training data, which does not introduce artificial variations, is to partition the shapes either spatially or in the frequency domain. The assumption behind this is that the individual partitions are likely to contain less variation than the complete shape so that their deformations should be easier to model with only a few training shapes. For example, Blanz and Vetter [19] built a statistical shape model of human faces. In order to increase the expressiveness of their model, they segmented the faces into four parts, namely the mouth, nose, eyes, and the rest. This means that the space of feasible shapes is divided into four subspaces and a model fit is obtained by searching four independent parameter vectors in these subspaces. The thus obtained four shape parts have afterwards been blended together with the help of an image stitching algorithm which yields a natural-looking complete face.

A different approach has been made by de Bruijne et al. [24]. They proposed an extension of statistical shape models that is especially tailored to tubular objects. It consists of decoupling the variations in the bending of the centerline from the variations in the diameter of the object. This is achieved by building two separate models so that one describes the bending of the object and the other describes the varying diameter. Both models are subsequently combined into one model by extracting the principal modes of variation from a matrix that contains the main modes of variation from both models, the centerline model and the diameter model, respectively. Another approach that is best suited for elongated objects has been made by Roberts et al. [104]. They proposed to describe the object with a global statistical shape model that is combined with a sequence of partially overlapping sub-models. Each of the sub-models is thereby trained by using only a sub-part of all available object points. An initial solution for the global model is obtained by minimizing the mean squared error to a few user-defined landmarks. The thus obtained solution is then used as a soft constraint in the determination of the local solutions. The sub-models are fitted to the data in a user-defined sequence, and the overlapping parts of the sub-models are used as additional soft constraints in the fitting of neighboring sub-models. As a result, the segmentation clearly depends on the sequence in which the different sub-models are adapted to the data.

A straightforward further development of partitioned statistical shape models are hierarchical statistical shape models. Davatzikos et al. [43] proposed two methods to obtain those models. The basic idea of their approaches is to use one global statistical shape model that captures the global deformations of an object class and many local statistical shape models that capture the local variations, respectively. In their first approach, they partition the shapes spatially into a collection of lined-up segments, each containing a subset of all points that define the object boundary. They then compute the gravity center of each segment and build a global PDM by considering only these center points. Afterwards, one local PDM is constructed for

each segment by considering only the points in the respective segment. Now, in order to approximate a shape with their hierarchical model, Davatzikos et al. first compute a fit of the global model, use this to determine the gravity centers of the sub-models, and then refine the global fit by computing local fits for all sub-models. Like in the approach by Blanz and Vetter, this approach leads to discontinuities at the segment borders so that Davatzikos et al. also compute a continuous blending in order to smoothly connect neighboring segments.

Because their first approach has been rather heuristically motivated, Davatzikos et al. additionally proposed a mathematically more sound hierarchical shape partition based on the wavelet transform. They used the discrete wavelet transform in order to divide the space-frequency domain in a number of bands that are arranged in a tree structure, i.e. the low-frequency bands have a broad spatial support whereas high-frequency bands get more and more localized. The Daubechies wavelets are used as basis functions since they have a wider support as e.g. Haar wavelets and thus provide a smoother transition between neighboring bands. The remaining spatial coupling among neighboring bands greatly reduces the discontinuities between them so that no additional blending is necessary. Now, similar to their heuristic approach, Davatzikos et al. independently model the wavelet coefficients in each frequency band by building one PDM for each band. With this model, an unknown shape can be approximated by computing the wavelet transform of the shape, projecting the wavelet coefficients into the individual PDM subspaces, and computing the inverse wavelet transform of the now shape-restricted coefficients. Nain et al. [92] extended the wavelet-based approach from Davatzikos et al. to 3D shapes by mapping the 3D shapes onto a sphere and then using spherical wavelets to divide them into space-frequency bands. In the model fitting process, they start by fitting the global PDM at the coarsest level of their wavelet tree and incrementally add the higher frequency bands when no better approximation can be obtained at the preceding level. This is supposed to increase the robustness of the fitting process against local minima, in contrary to the wavelet-based approach by Davatzikos et al. where all coefficients are used from the beginning of the fitting process.

Another hierarchical approach has been proposed by Knothe [71]. He extended the approach from [19] by using a Laplacian pyramid in order to divide the face shape variations into various frequency bands. In contrast to the wavelet-based approaches from [43] and [92], the frequency bands obtained from the Laplacian pyramid have intrinsically no local support. However, in practical applications, the variance of the high frequency components differs greatly from zero only in local regions of the shape (e.g. around the nose or the ears for face shapes) and is close to zero in the remaining regions. So, Knothe proposed to spatially decouple the high frequency components by thresholding the variance. Afterwards, a PCA is performed in each spatially decoupled frequency band and the resulting eigenvalues and eigenvectors are gathered into one combined PDM by sorting them with regard to decreasing eigenvalues. This construction leads to some global modes which control the overall shape and many modes with local support which control local face properties.

As mentioned above, shape inconsistencies can occur at the borders of the individual partitions when individual weight vectors are used for each partition and the

partitions have no overlapping support. Zhao et al. [141] tried to address this problem. As in the preceding approaches, the weight vectors for each sub-part of the shape are first determined individually. Subsequently, the individual weight vectors are connected by straight line segments so that a modeled shape is now represented by a weight curve. Each training sample can likewise be presented by a weight curve. In order to prevent shape inconsistencies, Zhao et al. then determine the sample curve with closest distance to the weight curve of the modeled shape and restrict the modeled shape to be *close* to the sample curve. This is achieved by applying an affine transformation that is constructed as a trade-off between small deformations of the weight curve and minimization of the mean squared distance to the nearest sample curve. By doing this, basically only those weight combinations are allowed which also occurred for one of the training shapes. This may prevent the problem of shape inconsistencies, however, this comes at the cost of loosing the ability to adapt to different training shapes in the individual partitions, which is one of the central parts of partitioned statistical shape models.

Recently, Amberg et al. [11, 82] proposed an approach that differs from all the above-mentioned approaches. Their approach is inspired by *local linear regression* and it is able to enhance the model flexibility while avoiding to explicitly partition the model either spatially or in the frequency domain. The basic idea is to fit the statistical shape model only to a local neighborhood around a given point on the target shape instead of fitting the model to the complete target shape. The thus obtained solution then determines the modeled shape in this particular point only. So, in order to obtain a fit of the full model, the local fitting procedure has to be repeated for all points on the target shape. The size of the local neighborhood to which the model is fitted thereby determines how strictly the modeled shape adheres to the global shape properties. When the neighborhood includes all contour points, one obtains a global model fit that is strictly restricted to the subspace of feasible shapes. On the other hand, when the neighborhood contains only the actually considered point, the shape constraints are completely relaxed. The major problem with this approach is that computing a fit of the full SSM in each contour point is computationally very demanding. In order to address the computational burden, Amberg et al. proposed to compute a full model fit only for a subset of the contour points and to smoothly interpolate between the local model fits. This reduces the computation time, however, it can also be regarded as re-introducing predefined segments into the model-fitting process.

In the following, we will present a new approach to enhance the model flexibility. We call it the *Locally Deformable Statistical Shape Model* (LDSSM). Similar to [11], the main idea of our LDSSM is to allow a different model approximation in each point of the underlying data domain. However, we propose a novel framework that iteratively adapts the parameters of the local model approximations in order to approximate a given target shape. So, our approach allows to obtain a locally optimal model approximation without the necessity for any predefined segments. Additional smoothness constraints on the local parameters thereby ensure a globally consistent solution. The iterative framework removes the need to compute a full SSM fit in each point and makes our approach directly applicable also for high-resolution 3D datasets.

From Global to Local Statistical Shape Priors
Novel Methods to Obtain Accurate Reconstruction
Results with a Limited Amount of Training Shapes
Last, C.

2017, XXI, 259 p. 84 illus., 64 illus. in color., Hardcover

ISBN: 978-3-319-53507-4