Chapter 3

Methods for vessel reconstruction and modeling

In this chapter, the focus is directed towards the methods used in the model building process. First, a snake formulation for curves and surfaces with B-splines is explained. Deformable models of one and two dimensions are separately described. Deformable models are used in this thesis to detect, track the vessels in coronary image sequences and perform $3D$ reconstruction. The problem to assign the external forces to the control points and the sensitivity to low level feature detector is introduced. Then, a survey of the statistical techniques used in this thesis is discussed. Statistical learning, dimensional reduction and distance measurement shall be used later to build the coronary model. The chapter finishes with a review of current tracking schemes in computer vision.

3.1 Physic based deformable models. Snakes

Using the well known principles of the newton mechanics, the general idea of deformable models in computer vision consist on the construction of a potential field from the image feature of interest where the model adapts itself to the feature minimizing its energy. In order to organize image features in semantic units, a global segmentation technique based on snakes [50] has important advantages.

- The method allows for incorporating high-level knowledge constraining the domain of solutions when doing image segmentation. The high-level knowledge introduced into the image processing is in terms of general constraints such as smoothness, continuity [50], closeness and guide by an approximate model [82]. These constraint regularize the problem of image feature organization and guide the process of finding a unique solution as a function of initial conditions.

- It is based on physical models theory. Snakes [50, 17, 82, 106] have the advantage of being dynamic and elastic curves with a physical interpretation. Snakes

represent physics-based models defined from the theory of elasticity and Newton Mechanics.

- Any solution using deformable models in computer vision is formulated using the concept of forces and equilibrium. Energy principles are used to deform the elastic curve under external forces attracting the curve towards the image features of interest; while internal forces take charge of imposing the constraint on the shape of the snake. Therefore, shape changes are driven by known dynamic laws. Elastic properties of physical nature as tension and stiffness are attributed to the deformable models to control its deformation. Additionally, mass and dissipation can be included to determine the dynamics of the models. The snake deforms as a physical body where the deformation is smooth and without discontinuities.

Recently, active contour models (snakes) have become not only a standard segmentation technique to organize image features but to deal with many other problems in computer vision. Some of these problems are:

- Segmentation of subjective contours [50], ultrasound cardiac images [91], satellite images [83], coronary arteries segmentation [54, 106], etc.. In [72] a statistical approach is incorporated into snakes for segmentation and tracking purposes. Other application of deformable contour models, to segment structures in $2D$ images can be found in [11, 12, 23].

- $3D$ object's boundary reconstruction of objects - of anatomical $3D$ object's boundary [84, 25, 24].

- Temporal tracking: indoor scenes[16, 9], facial features [56, 36], cardiac SPECT images [4, 5].

- Elastic matching of brain atlases in MR images [29, 31].

- Non-rigid motion reconstruction of anatomical organs (e.g. left ventricle) in SPECT images [5].

From the items above it is clear that deformable models have an increasing use in medical image analysis. Using snakes, one can interpolate image data representing objects in images as long as a good snake initialization is given.

Given the Computer Vision problems of segmenting and tracking objects in images, snakes begin to deform from an initial position trying to adjust to image features (e.g. edge points) under smooth and continuous deformations. It is deformed as a result of the influence of local forces derived from some feature of interest, while this deformation remains smooth due to the effect of internal forces.

These forces are represented as elastic curves with associated energy. External energy is defined as a function of the curve distance to the image features of interest. Internal energy depends on the smoothness and continuity of the model shape. The segmentation by snakes is defined as an energy-minimization problem. The snake deforms as close as possible to the image features of interest minimizing its external energy, while keeping its shape as smooth as possible minimizing its internal energy.

Representing parametrically the position of the snake as $\mathbf{Q}(u) = (x(u), y(u))$, the energy functional of the snake is written as follows:

$$E_{snake} = \int [E_{int}(\mathbf{Q}(u)) + E_{ext}(\mathbf{Q}(u))]du \tag{3.1}$$

where $E_{int}(\mathbf{Q}(u))$ is the internal energy and $E_{ext}(\mathbf{Q}(u))$ is the external energy. Snake energy (3.1) is minimised by Euler-Lagrange equation yielding:

$$-\frac{d}{du}(\alpha(u)\mathbf{Q}^{'}(u)) + \frac{d^2}{du^2}(\beta(u)\mathbf{Q}^{''}(u)) + \nabla E_{ext}(\mathbf{Q}(u)) = 0 \tag{3.2}$$

where $\alpha(u)$ and $\beta(u)$ are weight functions controlling the relative importance of the internal deformations. The external forces forming E_{ext} makes the snake to approach and lock on image features (minimizing the external energy). To define the external energy, a detector of image features is applied to build a potential map as a function of the distance to the extracted image features [82, 50]. Usually, feature map is generated as an edge/crest/valley image map [50, 83, 43] and the snake deforms on a potential field constructed as a distance map to the extracted image features. Temporal tracking is also achieved using the minimized snake in one frame as the initial one for minimizing in the next frame. Snakes can be generalized to n-dimensions.

3.2 Deformable models implementation. B-Snakes.

3.2.1 Numerical representation

Many different implementations can be used for the snakes. Kass et al. [50] proposes a point-based implementation and Menet et al. [64] a B-spline based snake. For the vessel structures, a B-spline [6, 17] based representation offers many interesting advantages.

1. Low degree curves: high processing efficiency and numerical stability. The lower the polynomial degree, the lower the complexity of the related algorithms in space and time.

2. The polynomial degree, once defined, is constant and independent of the number of points to interpolate.

3. Continuity: a k-degree spline has continuity up to order $k-1$.

In our case we have chosen to represent the snake model by polynomial of cubic degree. Using polynomial functions of cubic degree is enough to represent any vessel shape. Piecewise polynomials curves and a set of few control points are very important from the point of view of computational efficiency. However, it is necessary the use of some criteria to translate the external forces along the spline to the control points [60]. A review of the B-spline properties follows.

3.2.2 B-spline representation of deformable models

Let u be an internal parameter of a model in $\mathbb{R}^n$ $\mathbf{Q}(u) = (x_1(u), x_2(u), \ldots, x_n(u))$, (usually $2 \leq n \leq 3$).

A B-spline curve representation has the form:

$$\mathbf{Q}(u) = \sum_{i=0}^{m} \mathbf{v}_i B_{i,p}(u) \tag{3.3}$$

Where $\mathbf{v}_i$ are $m+1$ control points and $B_{i,p}(u)$, $i = 0, \ldots, m$ are piecewise polynomial functions forming a basis for the vector space of all piecewise polynomial functions, these functions are of a desired degree (p) and continuity (for a fixed breakpoint sequence) [81]. The continuity of the model is determined by the basis functions. Hence the control points can be modified without altering the model continuity. $B_{i,p}(u)$, $i = 0, \ldots, m$ are called standard B-spline blending functions. The curve representation obtained using B-splines inherits all the properties of such basis. Eq. (3.3) can also, be expressed in matrix notation as a matrix product between a vector of B-spline functions and a vector of weights (control points). For example, each control point i (weight) can be written as a vector $\mathbf{v}_i = v_x, v_y$ for a curve laying in a plane. The blending functions are recursively defined. Let $\mathbf{U} = \{u_0, \ldots, u_m\}$ be a non-decreasing sequence of real numbers, i.e. $u_i \leq u_{i+1}, i = 0, \ldots, m-1$. The u_i are called knots and $\mathbf{U}$ represent the knot vector. The jth B-spline basis function of k degree $B_{j,k}(u)$ is defined as follows:

$$B_{j,0}(u) = \begin{cases} 1 & \text{if } u_j \leq u < u_{j+1}, \\ 0 & \text{otherwise.} \end{cases}$$

$$B_{j,k}(u) = \frac{u - u_j}{u_{j+k} - u_j} B_{j,k-1}(u) + \frac{u_{j+k+1} - u}{u_{j+k+1} - u_{j+1}} B_{j+1,k-1}(u)$$

Properties of the B-spline curves (see also figures 3.1 and 3.2):

1. Local support property: $B_{j,k}(u) = 0$ if u is outside the interval $[u_j, u_{j+k+1})$. Local changes of snake curve do not affect the rest of the curve.
2. In any given knot span $[u_j, u_{j+1})$ at most $k+1$ of the $B_{j,k}$ are non zero, namely the functions $B_{j-k,k}, \ldots, B_{j,k}$
3. $B_{j,k}(u) \geq 0$ for all j, k, u
4. For an arbitrary knot span $[u_j, u_{j+1})$, $\sum_{i=j-k}^{j} B_{j,k}(u) = 1$
5. Controlled continuity: all derivatives up to order k of $B_{j,k}(u)$ exist in the interior of a knot span. At a knot, $B_{j,k}(u)$ is k times continuously differentiable.
6. B-splines can be either open or closed.

Note that one dimensional splines (curves) can have two or three dimensional control points depending on whether they lie in a plane or in a three dimensional space.

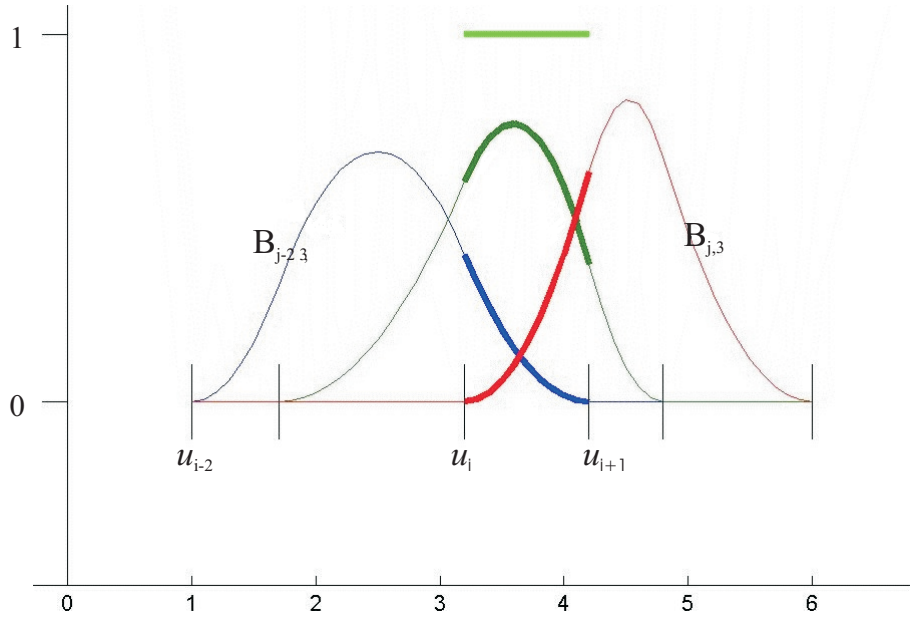


Figure 3.1: B-spline basis functions.

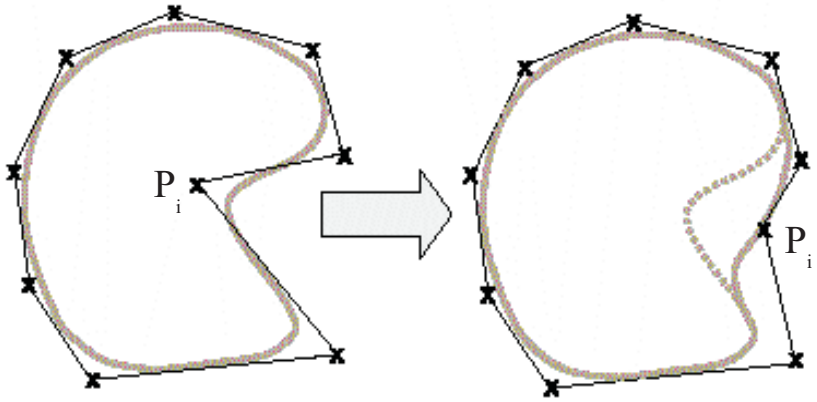


Figure 3.2: Change of a control point.

3.2.3 B-spline surfaces

The curve $\mathbf{Q}(u)$ is a vector valued function of one parameter. It is a mapping of a straight line segment into Euclidean two or three-dimensional space.

A surface is a vector-valued function of two parameters, u and v, and represent a mapping of a region $\Re \subset \mathbb{R}^2$, of the uv plane into Euclidean three-dimensional space. Thus it has the form $\mathbf{S}(u,v) = (x(u,v), y(u,v), z(u,v)), (u,v) \in \Re$.

There are many schemes for representing surfaces [49, 74, 73] They differ in the coordinate system and in the type of region $\Re$. Probably, the simplest method and the one most widely used in geometric modeling applications is the *tensor product* scheme.

It uses basis functions and geometric coefficients. The basis functions are bivariate functions of u and v, which are constructed as products of univariate basis functions.

The geometric coefficients are arranged in a (topologically) bidirectional, $n \times m$ net. Thus a tensor product surface has the form:

$$\mathbf{S}(u,v) = (x(u,v), y(u,v), z(u,v)) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_i(u) B_j(v) \mathbf{p}_{ij} \tag{3.4}$$

where $\mathbf{p}_{ij} = (x_{ij}, y_{ij}, z_{ij})$, $0 \leq u, v \leq 1$.

Note that the (u,v) domain of this mapping is a square (a rectangle in general). B-spline surfaces inherit the properties of univariate B-splines. While surfaces shall be considered as two dimensional spline it has to be clear that this is regarding the number of spline internal parameter and such dimension should not necessarily be the same as the control points dimensions.

3.2.4 Matrix representation for deformable models

Regarding a computer implementation it is interesting to give a unified matrix notation for B-splines curves and surfaces [17]. Using a matrix approach, usually one can compute constant blending matrices only once increasing the global performance of the related algorithms. Any parametric curve (like B-spline) can be noted in a matrix form. Knowing that a B-spline is a weighted sum of polynomials, the curve can be expressed as a product between a vector of polynomials and a vector of weights as follows:

Let $\mathbf{P}$ be a vector of $2D$ points

$$\mathbf{P} = \begin{pmatrix} \mathbf{p}_0 \\ \vdots \\ \mathbf{p}_m \end{pmatrix}$$

Noting $\mathbf{P} = (\mathbf{P}_x, \mathbf{P}_y)$ where

$$\mathbf{P}_x = \begin{pmatrix} x_0 \\ \vdots \\ x_m \end{pmatrix}$$

and

$$\mathbf{P}_y = \begin{pmatrix} y_0 \\ \vdots \\ y_m \end{pmatrix}$$

One can write

$$\mathbf{Q}(u) = (x(u), y(u)) = \begin{cases} x(u) = & \mathbf{F}^T\mathbf{P}_x, \\ y(u) = & \mathbf{F}^T\mathbf{P}_y \end{cases}$$

and more compact:

$$\mathbf{Q}(u) = \mathbf{F}^T\mathbf{P}$$

where

$$\mathbf{F}(u) = (F_0(u), F_1(u), \ldots, F_m(u))^T$$

When the elements $F_i(u)$ are the B-spline blending functions we have a B-spline curve.

For one-dimensional B-splines, Blake [17] uses a matrix notation well suited for snake computation. The recursive rules for generating the B-splines basis function can be converted into an algorithm by expressing each basis function as a sequence of polynomials $p_n(u)$ defined over the intervals $[u, u_{n+1})$. Since a B-spline basis $B_{n,d}$ is zero out of the interval $[u_n, u_{n+d})$, any B-spline function of order d can be expressed using just d polynomials $B^S_{n,d}$, one for each of the d spans where the basis function is not zero. The inductive form can be applied over each interval to obtain each $B^S_{n,d}$. The span is defined as any non-empty inter-knot interval to cope with the multiple knot case (considered as an empty inter-knot interval). If we consider that all the spans are unit length then the basis functions making up a spline are uniquely determined by the knot multiplicities $m_0, \ldots, m_L$ at the breakpoints.

A B-spline curve $\mathbf{Q}(u)$ of order d ($d = 4$ for cubic splines) and m control points, is defined for $0 < u \leq N$, where $m = N$ for closed curves and $m = N + d$ for open ones (with appropriate variations where multiple knots are used to vary curve continuity): Over the span S_σ any spline function is a linear combination of the basis functions $B_{b_\sigma,d} \ldots B_{b_\sigma+d-1,d}$ where $b_\sigma = (\sum_{i=0}^{\sigma} m_i) - d$.

Then:

$\mathbf{Q}(u)^{[\sigma,\sigma+1)} = \mathbf{Q}(u)^\sigma = \sum_{i=b_\sigma}^{b_\sigma+d-1} \mathbf{p}_i B_{i,d}(u)$

For each span we can compute a $d \times d$ span matrix B^S_σ such that

$$\mathbf{Q}^\sigma(u+\sigma) = (1, u, \ldots, u^{d-1})B^S_\sigma \begin{pmatrix} \mathbf{p}_{b_\sigma} \\ \vdots \\ \mathbf{p}_{b_\sigma+d-1} \end{pmatrix}$$

Finally, one can compute the curve using the previously computed span matrices, and write the curve as follows:

$$\mathbf{Q}(u) = \mathbf{B}^T(u)\mathbf{P} = \mathbf{B}^T(u+\sigma)\mathbf{P} = \mathbf{u}\mathbf{B}^S_\sigma\mathbf{G}_\sigma\mathbf{P} \tag{3.5}$$

where $0 < u \leq 1, \mathbf{P}$ is a $m \times 1$ vector containing the control points, $\mathbf{B}^S_\sigma$ is a $d \times d$ standard B-spline matrix where the ith column corresponds to the polinomial coefficients

of the basis function $B_{b_\sigma+i-1,d}$ over the interval of span σ, $\mathbf{G}_\sigma$ is a $d \times m$ matrix that simply selects d consecutive control points.

$$(G_\sigma)_{ij} = \begin{cases} 1 & \text{if } i - b_\sigma = j \\ 0 & \text{otherwise} \end{cases}$$

and $\mathbf{u} = (1, u, \ldots, u^{d-1})$

For example, $\mathbf{G}_0$ and $\mathbf{G}_2$ for a cubic B-spline with nine control points should be as follow:

$$\mathbf{G}_0 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{G}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Given that:

$$\mathbf{u}^{'} = (0, 1, \ldots, (d-1)u^{d-2}) \quad and$$
$$\mathbf{u}^{''} = (0, 0, 2, \ldots, (d-2)(d-1)u^{d-3})$$

the derivatives of the function can be computed as

$$\mathbf{Q}^{'}(u+\sigma) = \mathbf{u}^{'}\mathbf{B}_\sigma^S\mathbf{G}_\sigma\mathbf{P} = \mathbf{B}^{'}\mathbf{P} \tag{3.6}$$

$$\mathbf{Q}^{''}(u+\sigma) = \mathbf{u}^{''}\mathbf{B}_\sigma^S\mathbf{G}_\sigma\mathbf{P} = \mathbf{B}^{''}\mathbf{P} \tag{3.7}$$

When considering a B-spline curve in a plane we get:

$$\begin{pmatrix} x \\ y \end{pmatrix} = (\mathbf{uu}) \begin{pmatrix} \mathbf{B}_\sigma^S\mathbf{G}_\sigma & 0 \\ 0 & \mathbf{B}_\sigma^S\mathbf{G}_\sigma \end{pmatrix} \begin{pmatrix} \mathbf{P}_x \\ \mathbf{P}_y \end{pmatrix} = \mathbf{BP}$$

where $\mathbf{P}_x, \mathbf{P}_y$ are the control points components in x and y respectively and $\mathbf{(uu)}$= $(1, u, \ldots, u^{d-1}1, u, \ldots, u^{d-1})$

The first and second derivatives can be expressed as follows:

$$\begin{pmatrix} x^{'} \\ y^{'} \end{pmatrix} = (\mathbf{u}^{'}\mathbf{u}^{'}) \begin{pmatrix} \mathbf{B}_\sigma^S\mathbf{G}_\sigma & 0 \\ 0 & \mathbf{B}_\sigma^S\mathbf{G}_\sigma \end{pmatrix} \begin{pmatrix} \mathbf{P}_x \\ \mathbf{P}_y \end{pmatrix}$$

$$\begin{pmatrix} x^{''} \\ y^{''} \end{pmatrix} = (\mathbf{u}^{''}\mathbf{u}^{''}) \begin{pmatrix} \mathbf{B}_\sigma^S\mathbf{G}_\sigma & 0 \\ 0 & \mathbf{B}_\sigma^S\mathbf{G}_\sigma \end{pmatrix} \begin{pmatrix} \mathbf{P}_x \\ \mathbf{P}_y \end{pmatrix}$$

The normal to the B-spline curve is given by:

$$\begin{pmatrix} n_x \\ n_y \end{pmatrix} = \begin{pmatrix} -y^{'} \\ x^{'} \end{pmatrix} = (\mathbf{u}^{'}\mathbf{u}^{'}) \begin{pmatrix} \mathbf{B}_\sigma^S\mathbf{G}_\sigma & 0 \\ 0 & \mathbf{B}_\sigma^S\mathbf{G}_\sigma \end{pmatrix} \begin{pmatrix} -\mathbf{P}_y \\ \mathbf{P}_x \end{pmatrix}$$

Note also that a surface $\mathbf{S}(u, v)$ can be described in a matrix form:

$\mathbf{S}(u,v) = [B_i(u)]^T[\mathbf{P}_{ij}][B_j(v)]$, where $[B_i(u)]^T$ is a $(1) \times (n+1)$ row vector of functions, $[B_j(v)]$ is a $(m+1) \times (1)$ column vector of functions and $[\mathbf{P}_{ij}]$ is a $(n+1) \times (m+1)$ matrix of control points forming the grid which define the surface. When $B_i(u)$ and $B_j(v)$ are univariate B-spline basis functions associated to two knot vectors $\mathbf{U}$ and $\mathbf{V}$ respectively, we obtain a B-spline surface.

3.2.5 Internal Energy

From mechanics one can consider the deformation energy of any elastic material and associate it to the snake curve. A suitable classification for such energy is in two kinds; stretching and bending. For any curve the stretching energy is proportional to the square of the first derivative and the bending is proportional to the square of the second derivative. $E_{stretching} \propto \mathbf{Q}_u^2(s)$ $E_{bending} \propto \mathbf{Q}_{uu}^2(s)$. Therefore, the total internal energy for the snakes can be written in terms of a sum of membrane energy, expressing the snake stretching, and of thin-plate energy, expressing the snake bending:

$$E_{int}(\mathbf{Q}(u)) = K_1\alpha(u)|\frac{d\mathbf{Q}(u)}{du}|^2 + K_2\beta(u)|\frac{d^2\mathbf{Q}(u)}{du^2}|^2 \tag{3.8}$$

The parameters of elasticity $\alpha(u)$ and $\beta(u)$ control the smoothness of the snake curve.

The concept can be extended to the two dimensional case as follows:

$$\begin{aligned} E_{int}(\mathbf{Q}) \propto \quad & \alpha_1|\tfrac{\partial\mathbf{Q}(u,v)}{\partial u}|^2 + \alpha_2|\tfrac{\partial\mathbf{Q}(u,v)}{\partial v}|^2 + \beta_{11}|\tfrac{\partial^2\mathbf{Q}(u,v)}{\partial u^2}|^2 + \\ & \beta_{12}|\tfrac{\partial^2\mathbf{Q}(u,v)}{\partial u\partial v}|^2 + \beta_{22}|\tfrac{\partial^2\mathbf{Q}(u,v)}{\partial v^2}|^2 \end{aligned} \tag{3.9}$$

where $\mathbf{Q}$ is the two dimensional spline (surface) with internal parameters u and v.

Internal energy can be expressed in a convenient matrix form known as stiffness matrix. Using the first and second derivatives of the splines formulated above the stiffness term is:

$$E_{int}(\mathbf{Q}) = \int_0^L [\alpha(u)\mathbf{Q}_u(u)^2 + \beta(u)\mathbf{Q}_{uu}(u)^2]du \tag{3.10}$$

Considering $\alpha(u)$ and$\beta(u)$ as constants:

$$E_{int}(\mathbf{Q}) = \alpha\int_0^L \mathbf{Q}_u(u)^2du + \beta\int_0^L \mathbf{Q}_{uu}(u)^2du \tag{3.11}$$

Using (3.6) and (3.7) we get:

$$\begin{aligned} E_{int}(\mathbf{Q}) &= \alpha \int_0^L \mathbf{P}^T \mathbf{B}'^T(u)\mathbf{B}'(u)\mathbf{P}du + \beta \int_0^L \mathbf{P}^T \mathbf{B}''^T(u)\mathbf{B}''(u)\mathbf{P}du \\ &= \alpha \mathbf{P}^T \tfrac{1}{L} \textstyle\sum_{\sigma=0}^{L-1} \mathbf{G}_\sigma^T \mathbf{B}_\sigma^{ST} \int_0^1 \mathbf{u}'^{\mathbf{T}} \mathbf{u}' du \; \mathbf{G}_\sigma \mathbf{B}_\sigma^S \mathbf{P} + \\ &\quad \beta \mathbf{P}^T \tfrac{1}{L} \textstyle\sum_{\sigma=0}^{L-1} \mathbf{G}_\sigma^T \mathbf{B}_\sigma^{ST} \int_0^1 \mathbf{u}''^{\mathbf{T}} \mathbf{u}'' du \; \mathbf{G}_\sigma \mathbf{B}_\sigma^S \mathbf{P} \\ &= \alpha \mathbf{P}^T \tfrac{1}{L} \textstyle\sum_{\sigma=0}^{L-1} \mathbf{G}_\sigma^T \mathbf{B}_\sigma^{ST} \boldsymbol{ß}' \; \mathbf{G}_\sigma \mathbf{B}_\sigma^S \mathbf{P} + \\ &\quad \beta \mathbf{P}^T \tfrac{1}{L} \textstyle\sum_{\sigma=0}^{L-1} \mathbf{G}_\sigma^T \mathbf{B}_\sigma^{ST} \boldsymbol{ß}'' \mathbf{G}_\sigma \mathbf{B}_\sigma^S \mathbf{P} \\ &= \mathbf{P}^T \tfrac{1}{L} \textstyle\sum_{\sigma=0}^{L-1} \mathbf{G}_\sigma^T \mathbf{B}_\sigma^{ST} (\alpha \boldsymbol{ß}' + \beta \boldsymbol{ß}') \mathbf{P} \end{aligned} \tag{3.12}$$

being $\boldsymbol{ß}' = \int_0^1 \mathbf{u}'^{\mathbf{T}} \mathbf{u}' du$ and $\boldsymbol{ß}'' = \int_0^1 \mathbf{u}''^{\mathbf{T}} \mathbf{u}'' du$.

For a B-spline of order 4, $\boldsymbol{ß}'$ and $\boldsymbol{ß}''$ take the form:

$$\boldsymbol{ß}' = \int_0^1 \begin{pmatrix} 0 \\ 1 \\ 2u \\ 3u^2 \end{pmatrix} (0\ 1\ 2u\ 3u^2) du = \int_0^1 \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 2u & 3u^2 \\ 0 & 2u & 4u^2 & 6u^3 \\ 0 & 3u^2 & 6u^3 & 9u^4 \end{pmatrix} du$$

$$\boldsymbol{ß}'' = \int_0^1 \begin{pmatrix} 0 \\ 0 \\ 2 \\ 6u \end{pmatrix} (0\ 0\ 2\ 6u) du = \int_0^1 \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 12u \\ 0 & 0 & 12u & 36u^2 \end{pmatrix} du$$

$$\boldsymbol{ß}' = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & \frac{4}{3} & \frac{3}{2} \\ 0 & 1 & \frac{3}{2} & \frac{9}{5} \end{pmatrix} \quad \boldsymbol{ß}'' = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 6 \\ 0 & 0 & 6 & 12 \end{pmatrix}$$

In general, the matrices $\boldsymbol{ß}'$ and $\boldsymbol{ß}''$ can be expressed with the following forms:

$$ß'_{i,j} = \begin{cases} 0 & \text{if } i = 1 \text{ or } j = 1, \\ \frac{(i-1)(j-1)}{i+j-3} & \text{otherwise.} \end{cases}$$

$$ß''_{i,j} = \begin{cases} 0 & \text{if } i < 3 \text{ or } j < 3, \\ \frac{(2-3i+i^2)(2-3j-j^2)}{i+j-5} & \text{otherwise.} \end{cases}$$

3.2.6 External Energy of a snake

The external energy comes from the image itself trying to attract the curve toward a selected image feature. Usually, it is formulated as a potential energy. Other forces

can be considered additionally to provide a predefined behavior somewhere along the curve.

Potential energy of a snake

Let us consider a surface as a potential field and assign a high value to the field far from the features of interest of the image and lower values toward the feature. The work of external forces along the curve have to push the snake towards the image features of interest. These forces are associated to a potential $Pot(x,y)$ which, in general, is defined in terms of gradient module of the image $I(x,y)$ convolved by a Gaussian function $G(x,y)$, [50]:

$$E_{ext} = Pot(x,y) = -|\nabla(G(x,y) * I(x,y))|$$

or as a distance map of the edge points [25]:

$$E_{ext} = Pot(x,y) = d(\mathbf{x}), \;\; Pot(\mathbf{x}) = -e^{-d(\mathbf{x})^2} \tag{3.13}$$

where $d(\mathbf{x})$ denotes the distance between the pixel $\mathbf{x}$ and its closest edge point. The snake is moved by forces conforming the potential and tries to fall in a valley as if it was under the effect of gravity. In figure 3.3 a frame of a coronary angiography sequence is used as an example showing the image valleys and the potential energy build as a distance map to the detected feature, figure 3.4 shows a 3D view of the potential energy showed in 3.3(c). Note that image features can be of different type. In [43] a crease detection and a minimum distance map to the creases is used as a potential map combined with edge potential field.

Energy minimization algorithm

Given a B-spline representation for the snakes, the expression for the total energy is [64]:

$$\begin{aligned}
\mathbf{E_{total}} = \;& \tfrac{1}{2}\int \alpha\mathbf{Q}'^2(u)) + \beta\mathbf{Q}''^2(u) + \mathbf{F}_{ext}(\mathbf{Q}(u)))du \\
= \;& \textstyle\sum_{j=0}^{p}\{\tfrac{1}{2}\alpha[(\sum_{i=0}^{m} X_i B_i'(u_j)^2) + (\sum_{i=0}^{m} Y_i B_i'(u_j)^2)] \\
& + \tfrac{1}{2}\beta[(\textstyle\sum_{i=0}^{m} X_i B_i''(u_j)^2) + (\sum_{i=0}^{m} Y_i B_i''(u_j)^2)] + \mathbf{F}_{ext}(\mathbf{Q}(u_j))\}
\end{aligned} \tag{3.14}$$

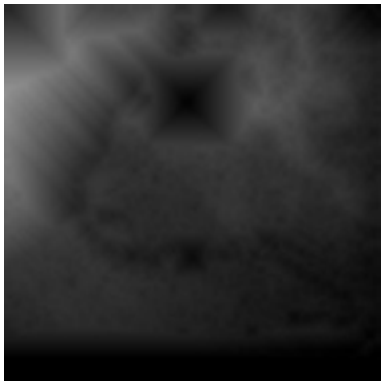
where the $\sum_{j=0}^{p}$ is the numerical integration.

The solution shall be a set of control points $(X_i, Y_i), i = 0, \ldots, m$ that minimize the total energy, that is:

$$\forall l \in \{0, \ldots, m\} \begin{cases} \frac{dE}{dX_l} = 0 \\ \frac{dE}{dY_l} = 0 \end{cases}$$

(a)

(b)



(c)

Figure 3.3: Left coronary artery angiography (a), Vessel center detection (b), Potential field as a distance map (c).

Decoupling (3.14) with the equations above for each coordinate, X and Y:

$$\begin{aligned}
&\textstyle\sum_{j=0}^{p}[\alpha B_l^{'}(u_j)\sum_{i=0}^{m} X_i B_i^{'}(u_j) + \beta B_l^{''}(u_j)\sum_{i=0}^{m} X_i B_i^{''}(u_j) + \\
&B_l(u_j)\tfrac{d}{dx}F(\textstyle\sum_{i=0}^{m} X_i B_i(u_j) + \sum_{i=0}^{m} Y_i B_i(u_j))] = 0 \\
&\textstyle\sum_{j=0}^{p}[\alpha B_l^{'}(u_j)\sum_{i=0}^{m} Y_i B_i^{'}(u_j) + \beta B_l^{''}(u_j)\sum_{i=0}^{m} Y_i B_i^{''}(u_j) + \\
&B_l(u_j)\tfrac{d}{dy}F(\textstyle\sum_{i=0}^{m} X_i B_i(u_j) + \sum_{i=0}^{m} Y_i B_i(u_j))] = 0
\end{aligned} \tag{3.15}$$

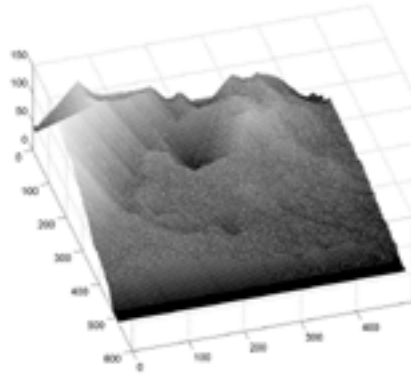


Figure 3.4: Topographic map of the potential image in fig. 3.3(c).

Changing the summation order we get:

$$\begin{aligned}
&\textstyle\sum_{i=0}^{m} X_i[\sum_{j=0}^{p} \alpha B_l^{'}(u_j)B_i^{'}(u_j) + \sum_{j=0}^{p} \beta B_l^{''}(u_j)B_i^{''}(u_j)+ \\
&\textstyle\sum_{j=0}^{p} B_l(u_j)\frac{d}{dx}F(\sum_{i=0}^{m} X_iB_i(u_j) + \sum_{i=0}^{m} Y_iB_i(u_j))] = 0 \\
&\textstyle\sum_{i=0}^{m} Y_i[\sum_{j=0}^{p} \alpha B_l^{'}(u_j)B_i^{'}(u_j) + \sum_{j=0}^{p} \beta B_l^{''}(u_j)B_i^{''}(u_j)+ \\
&\textstyle\sum_{j=0}^{p} B_l(u_j)\frac{d}{dy}F(\sum_{i=0}^{m} X_iB_i(u_j) + \sum_{i=0}^{m} Y_iB_i(u_j))] = 0, \\
&l \in \{0, \ldots, m\}
\end{aligned} \tag{3.16}$$

Writing (3.16) in a matrix form regarding the unknown coordinates $(\mathbf{X}, \mathbf{Y})$ of $m+1$ control points w e get:

$$\begin{cases} \mathbf{A}_b\mathbf{X} + H_x(x,y) = 0 \\ \mathbf{A}_b\mathbf{Y} + H_y(x,y) = 0 \end{cases} \tag{3.17}$$

Where $\mathbf{A}_b$ is the stiffness matrix (banded) for the B-snake.

$$\mathbf{A}_b = \begin{pmatrix} c_0 & d_0 & e_0 & 0 & 0 & a_0 & b_0 \\ b_1 & c_1 & d_1 & e_1 & 0 & 0 & a_1 \\ a_2 & b_2 & c_2 & d_2 & e_2 & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ e_{m-1} & 0 & 0 & a_{m-1} & b_{m-1} & c_{m-1} & d_{m-1} \\ d_m & e_m & 0 & 0 & a_m & b_m & c_m \end{pmatrix} \tag{3.18}$$

The matrix elements are:

$$\begin{aligned}
a_i &= \sum_{j=0}^{p} \alpha B_{i-2}^{'}(u_j)B_i^{'}(u_j) + \beta B_{i-2}^{''}(u_j)B_i^{''}(u_j) \\
b_i &= \sum_{j=0}^{p} \alpha B_{i-1}^{'}(u_j)B_i^{'}(u_j) + \beta B_{i-1}^{''}(u_j)B_i^{''}(u_j) \\
c_i &= \sum_{j=0}^{p} \alpha B_i^{'2}(u_j) + \beta B_i^{''2}(u_j) \\
d_i &= \sum_{j=0}^{p} \alpha B_{i+1}^{'}(u_j)B_i^{'}(u_j) + \beta B_{i+1}^{''}(u_j)B_i^{''}(u_j) \\
e_i &= \sum_{j=0}^{p} \alpha B_{i+2}^{'}(u_j)B_i^{'}(u_j) + \beta B_{i+2}^{''}(u_j)B_i^{''}(u_j)
\end{aligned} \tag{3.19}$$

and the ith elements of H_x and H_y are:

$$\begin{aligned}
[H_x]_i &= \sum_{j=0}^{p} B_i(u_j)\tfrac{d}{dx}F(\mathbf{Q}(u_j)) \\
[H_y]_i &= \sum_{j=0}^{p} B_i(u_j)\tfrac{d}{dy}F(\mathbf{Q}(u_j)) \\
i &= 0, \ldots, m
\end{aligned}$$

The equations in (3.17) can be solved in an iterative way:

$$\begin{aligned}
-\gamma(x_t - x_{t-1}) &= \mathbf{A}_b x_t + \tfrac{d}{dx}E_{ext}(x_{t-1} - y_{t-1}) \\
-\gamma(y_t - y_{t-1}) &= \mathbf{A}_b y_t + \tfrac{d}{dy}E_{ext}(x_{t-1} - y_{t-1}))
\end{aligned}$$

$$\mathbf{P}_t = (\mathbf{A}_b + \gamma I)^{-1}(\gamma \mathbf{P}_{t-1} + \nabla E_{ext}(\mathbf{Q}(u)_{t-1})) \tag{3.20}$$

In (3.20) the *damping parameter* γ (also called *Euler step size*) determines the rate of convergence of the minimization process. Matrix $\mathbf{A}_b$ is the stiffness matrix, $\mathbf{P}$ is the curve (control points) and E_{ext} is the external energy. In [81] Radeva generalizes the method for two dimensional snakes (deformable surfaces).

$$\mathbf{P}_t = (\mathbf{A}_u + \gamma I)^{-1}(\gamma \mathbf{P}_{t-1} + \nabla E_{ext}(\mathbf{Q}(u,v)_{t-1}))(\mathbf{A}_v + \gamma I)^{-1} \tag{3.21}$$

where $\mathbf{A}_u$ is the stiffness matrix for the u basis functions and $\mathbf{A}_v$ is the stiffness matrix for the v basis functions.

3.2.7 Critical issues in B-snakes: image feature detectors, external forces and control points

- The snake framework has a critical step when using the energy minimizing scheme applied to the segmentation problem in computer vision: the potential map is based on the output of a low level feature detection over the original image, and therefore, the whole method, being theoretically sound and correct, is highly dependent on the quality of the image feature detector used to build the potential field.

- Another question comes from the distribution of the forces along the curve. When using a B-spline, one need a way to map the forces to the control points to deform the curve. We decided to use the active basis function as weights for the external forces for the control points involved in each span.
- Finally, there are the inherent errors always present in the discrete formulation the snake energy minimizing scheme.

The snake is highly affected by the quality of a specific image feature detector. Given the problem of detection of vessel structures, we found that one of the best performing detectors on our application is represented by the level set based crease detector [59]. It offers a very good image feature response (see fig.3.3) although still could be affected by the cleaning steps on short segments in the feature map. Cleaning the the short and isolated detector responses yields additional discontinuities along the linear structure, as a result one can loose precision in detecting significant parts of the object of interest. The immediate consequence is a possible distraction of the snake to near, false image fatures, due to image noise or close objects. To avoid these problems we propose a statistical vessel learning for an optimal feature detection. A survey on the statistical basis used to this approach follows.

3.3 Introducing statistical elements into the snake framework

The introduction of a probabilistic approach in this framework is aimed to:

1. obtain an optimal representation of image features using feature learning and statistical descriptors
2. reduce the dimension of the space for the image feature representation

3.3.1 Low-rank modeling in signal processing

A key problem in statistical signal processing is that of feature selection, which refers to a process whereby a data space is transformed in a feature space that, in theory, has exactly the same dimension as the original data space in order to optimize the selection. However, it would be desirable to design the transformation in such a way that the data can be represented by a reduced number of effective features and yet retain most of the intrinsic information content of the input data. As the volume of data involved in image processing is usually high, this dimensional reduction keeping as much information as possible offers advantages; from the increment of the processing speed up to the reduction/elimination of noise. In image signals, where there are a lot of conditioning variables for computer analysis, the statistical approach allows the needed flexibility.

Suppose a vector of a data sequence $\mathbf{x}$ coming from a wide sense stationary process with zero mean and correlation matrix R. Let $\boldsymbol{\gamma_1}, \boldsymbol{\gamma_2}, \ldots, \boldsymbol{\gamma_p}$ be the eigenvectors associated with the p eigenvalues of the matrix R. The vector $\mathbf{x}$ may be expanded as a linear combination of these eigenvectors as follows:

$$\mathbf{x} = \sum_{i=1}^{p} c_i \boldsymbol{\gamma_i} \tag{3.22}$$

The coefficients of the expansion are zero-mean, uncorrelated random variables defined by the inner product:

$$c_i = \boldsymbol{\gamma_i^T} \mathbf{x}_i \qquad i = 1, 2, \ldots, p \tag{3.23}$$

The data representation of (3.23) is exact in the sense that it involves no loss of information. But, usually there are some eigenvalues with a very small value. Such eigenvalues can be considered as explaining noise and hence can be discarded. Retaining the information in the first, say $k < p$ eigenvalues, we may define an approximate reconstruction of the original data keeping a predefined percentage of the data variance, see property (3.31). A dimensional reduction is carried out as follows:

$$\hat{\mathbf{x}} = \sum_{i=1}^{k} c_i \boldsymbol{\gamma_i} \qquad k < p \tag{3.24}$$

A reconstruction of the approximation of $\hat{\mathbf{x}}$ can be achieved using a set of p coefficients $c_i, \;\; i = \{1, \ldots, p\}$ defined in terms of the data vector $\mathbf{x}$, see (3.23). The rank of $\hat{\mathbf{x}}$ is $p < k$ being k the rank of $\mathbf{x}$, so the data model defined in (3.23) is referred as a *low-rank model*. The coefficients c_i can be viewed as a representation of the original data in a dimensional reduced space. The incurred error in the reconstruction of $\mathbf{x}$ due to the fact that $\hat{\mathbf{x}}$ is of lower rank is defined as:

$$\mathbf{e}(n) = \mathbf{x} - \hat{\mathbf{x}} \tag{3.25}$$

hence, using (3.22) and (3.24) we get:

$$\mathbf{e}(n) = \sum_{i=p+1}^{k} c_i \boldsymbol{\gamma_i} \tag{3.26}$$

The mean square error is therefore as follows:

$$\begin{aligned} \epsilon \quad &= \mathbf{E}[\|\mathbf{e}(n)\|^2] = \mathbf{E}[\mathbf{e}(n)^T \mathbf{e}(n)] \\ &= \mathbf{E}[\textstyle\sum_{i=p+1}^{k} \sum_{j=p+1}^{k} c_i^T c_j \boldsymbol{\gamma_i^T} \boldsymbol{\gamma_j}] \\ &= \textstyle\sum_{i=p+1}^{k} \sum_{j+1}^{k} \mathbf{E}[c_i^T c_j] \boldsymbol{\gamma_i^T} \boldsymbol{\gamma_j} = \sum_{i=p+1}^{k} \lambda_i \end{aligned} \tag{3.27}$$

which confirms that the data reconstruction defined by (3.24) is a good one, provided that the eigenvalues $\lambda_{p+1}, \ldots, \lambda_k$ are very small.

3.3.2 Dimensional reduction in computer vision

The first steps in statistical computer vision and pattern recognition using Principal Component Analysis (PCA) [61] are found in [109]. They use PCA technique to extract relevant information and learn faces from images. The underlying idea is to

find a new base for a set of correlated random variables where they can be represented as not correlated. Such a base is known as Principal Components. In other words, the principal components are the standardized linear combination of the original variables with maximal variance. The use of PCA in this work is mainly for searching for a few optimal linear combinations to reduce the dimensionality of the space. The optimization is in the sense of keeping the set of variables explaining almost all the variance of the data and discarding the less important ones. This technique is widely used in one dimensional signal processing [46], where the variables explaining a low percentage of the variance are regarded as noise and, hence, discarded.

3.3.3 Principal component analysis

Definition: A linear combination $\mathbf{1}^T\mathbf{x}$ is called Standardized Linear Combination (SLC) if $\sum_i l_i^2 = 1$ [61].

The first objective of principal component analysis is to seek the SLC of a set of random variables which has maximal variance. More generally, principal component analysis looks for a few linear combinations which can be used to summarize the data, loosing in the process as little information as possible.

Definition: If $\mathbf{x}$ is a random n-dimensional vector with mean μ and covariance $\mathbf{\Sigma}$, then the **principal component transformation** is the transformation

$$\mathbf{x} \rightarrow \mathbf{y} = \mathbf{\Gamma}^T(\mathbf{x}-\boldsymbol{\mu}) \tag{3.28}$$

where $\mathbf{\Gamma}$ is an orthogonal matrix, $\mathbf{\Gamma}^T\mathbf{\Sigma}\mathbf{\Gamma} = \mathbf{\Lambda}$ is a diagonal matrix with diagonal elements $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p \geq 0$. The eigenvalues of $\mathbf{\Sigma}$ are positive because $\mathbf{\Sigma}$ is positive definite. The ith principal component of $\mathbf{x}$ may be defined as the ith element of the vector $\mathbf{y}$, namely as:

$$\mathbf{y_i} = \boldsymbol{\gamma}_i^T(\mathbf{x}-\boldsymbol{\mu}) \tag{3.29}$$

where $\boldsymbol{\gamma}_i^T$is the ith column of $\mathbf{\Gamma}$. The function y_p may be called the last principal component of $\mathbf{x}$. Figure 3.6 gives a pictorial representation in two dimensions of the principal components for $\boldsymbol{\mu} = \mathbf{0}$.

Theorem: No standard linear combination of $\mathbf{x}$ has a variance larger than λ_1, the variance of the first principal component [61].

3.3.4 Variance of the intermediate components

Theorem: If $\alpha = \mathbf{a}^T\mathbf{x}$ is a standard linear combination of $\mathbf{x}$ which is uncorrelated with the first k principal components of $\mathbf{x}$, then the variance of α is maximized when α is the $(k+1)th$ principal component of $\mathbf{x}$.

3.3.5 Sample principal components

Let $\mathbf{X} = (\mathbf{x_1}, \ldots, \mathbf{x_n})^T$ be a sample data matrix and $\mathbf{a}$ is a standardized vector. Then $\mathbf{Xa}$ gives n observations on a new variable defined as a weighted sum of the columns of $\mathbf{X}$. The sample variance of this new variable is $\mathbf{a}^T\mathbf{Sa}$ where $\mathbf{S}$ is the sample covariance

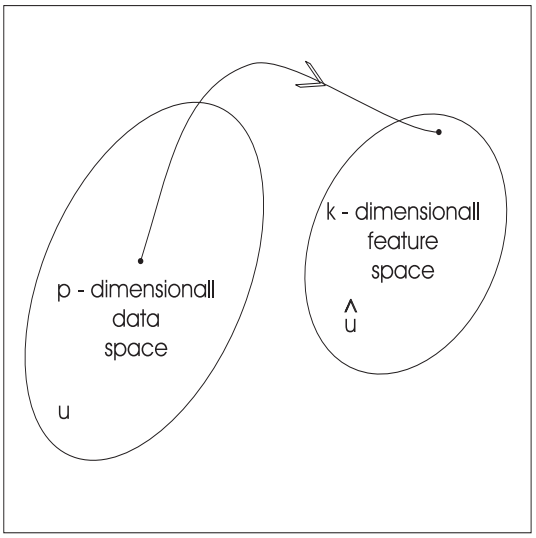


Figure 3.5: Dimensional Reduction ($k < p$).

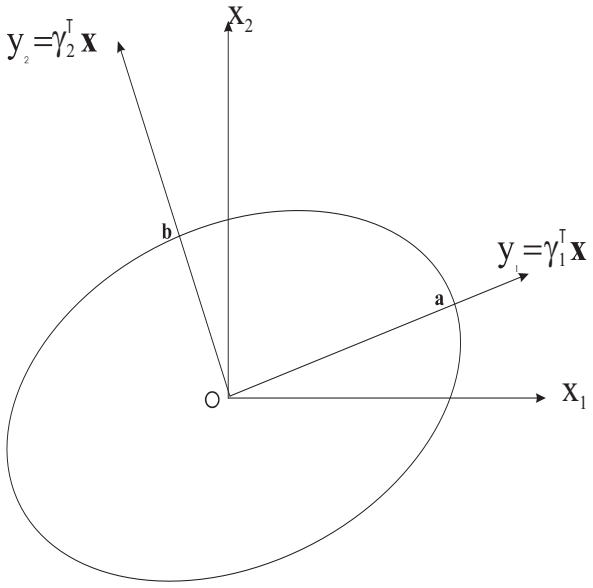


Figure 3.6: Ellipsoid $\mathbf{x}^T\mathbf{A}^{-1}\mathbf{x} = 1$.
Lines defined by $\mathbf{y_1}$and $\mathbf{y_2}$ are the first and second principal axes,
$\|\mathbf{a}\| = \sqrt{\lambda_1}, \|\mathbf{b}\| = \sqrt{\lambda_2}$.

matrix of $\mathbf{X}$. The first principal component is the standardized linear combination with largest variance and by direct analogy with (3.29), we get:

$$y_{(1)} = (X - \mathbf{1}\bar{\mathbf{x}}^T)\mathbf{g}_{(1)} \tag{3.30}$$

where $\mathbf{g}$ is the standardized eigenvector corresponding to the largest eigenvalue of $\mathbf{S}$. Similarly, the *ith* sample principal component is defined as $\mathbf{y}_{(i)} = (\mathbf{X} - \mathbf{1}\bar{\mathbf{x}})\mathbf{g}_{(i)}$. Putting the principal components together: $\mathbf{Y} = (\mathbf{X} - \mathbf{1}\bar{\mathbf{x}}^T)\mathbf{G}$, $(\mathbf{X} - \mathbf{1}\bar{\mathbf{x}}^T) = \mathbf{Y}\mathbf{G}^T$ since $\mathbf{G}$ is orthogonal.

Properties of principal components:

- The sum of the first k eigenvalues divided by the sum of all the eigenvalues

$$(\lambda_1 + \ldots + \lambda_k)/(\lambda_1 + \ldots + \lambda_p) \tag{3.31}$$

  represents the "proportion of total variation" explained by the first k principal components. The proportion of the total variation defined, gives a quantitative measure of the amount of information retained in the reduction from p to k dimensions.

- The principal components of a random vector are not scale invariant. If $p = 2$, considering $\Sigma = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}$, where $\rho > 0$, the larger eigenvalue is $\lambda_1 = \frac{1}{2}(\sigma_1^2 + \sigma_2^2) + \frac{1}{2}\Delta$, where $\Delta = \{(\sigma_1^2 - \sigma_2^2)^2 + 4\sigma_1^2\sigma_2^2\rho^2\}^{\frac{1}{2}}$, with eigenvector proportional to

$$(a_1, a_2) = (\sigma_1^2 - \sigma_2^2 + \Delta, 2\rho\sigma_1\sigma_2). \tag{3.32}$$

  When $\frac{\sigma_1}{\sigma_2} = 1$, the ratio $\frac{a_1}{a_2}$ given by (3.32) is unity. If $\sigma_1 = \sigma_2$ and the first variable is multiplied by a factor k, then for scale invariance we would like the new ratio $\frac{a_1}{a_2}$ to be k. However, changing σ_1 to $k\sigma_1$ in (3.32) shows that this is not the case.

- If the covariance matrix of $\mathbf{x}$ has rank $r < p$ then the total variation of $\mathbf{x}$ can be entirely explained by the first r principal components. If $\mathbf{\Sigma}$ has rank r, then the last $p - r$ eigenvalues of $\mathbf{\Sigma}$ are identically zero. Hence the result follows from the first property above.

- The vector subspace spanned by the first k principal components $1 \leq k < p$ has smaller mean square deviation from the population (or sample) variables than any other k-dimensional subspace. If $\mathbf{x} \sim (\mathbf{0}, \mathbf{\Sigma})$ and a subspace $H \subset \mathbf{R^P}$ is spanned by orthonormal vectors $\mathbf{h_{(1)}}, \ldots, \mathbf{h_{(k)}}$, then by projecting $\mathbf{x}$ onto this subspace we see that the squared distance d^2 from $\mathbf{x}$ to H has expectation $E(d^2) = E(\mathbf{x}^T\mathbf{x}) - \sum_{j=1}^{k} E(\mathbf{h_{(j)}^T x})$.

  Let $\mathbf{f_{(j)}} = \mathbf{\Gamma}^T\mathbf{h_{(j)}}, j = 1, \ldots, k$, we have

$$E(d^2) = tr\mathbf{\Sigma} - \sum_{j=1}^{k} E({\mathbf{f_{(j)}}}^T\mathbf{y}) = tr\mathbf{\Sigma} - \sum_{i=1}^{p}\sum_{j=1}^{k} \mathbf{f_{(ij)}^2}\lambda_\mathbf{i} \tag{3.33}$$

since the $\mathbf{f}_{(\mathbf{j})}$ are also orthonormal. It is minimized when $f_{ij} = 0, i = k+1, \ldots, p$ for each $j = 1, \ldots, k$; that is, when $\mathbf{f}_{(\mathbf{j})}$ span the subspace of the first k principal components.

- As a special case of the last property for $k = p - 1$, the plane perpendicular to the last principal component has a smaller mean square deviation from the population (or sample) variables than any other plane.

3.3.6 Distances and similarities

Definition: Let $\mathbf{P}$ and $\mathbf{Q}$ be two points where these may represent measurements $\mathbf{x}$ and $\mathbf{y}$ on two objects. A real-valued function $d(\mathbf{P}, \mathbf{Q})$ is a distance function if it has the following properties:

(**I**) symmetry, $d(\mathbf{P}, \mathbf{Q}) = d(\mathbf{Q}, \mathbf{P})$

(**II**) non-negativity, $d(\mathbf{P}, \mathbf{Q}) \geq 0$

(**III**) identification mark, $d(\mathbf{P}, \mathbf{P}) = 0$

For many distance function the following properties also hold:

(**IV**) definiteness, $d(\mathbf{P}, \mathbf{Q}) = 0$ if and only if $\mathbf{P} = \mathbf{Q}$

(**V**) triangle inequality, $d(\mathbf{P}, \mathbf{Q}) \leq d(\mathbf{P}, \mathbf{R}) + d(\mathbf{R}, \mathbf{Q})$

If (**I**) – (**V**) hold, d is called a metric. For some purposes, it is sufficient to consider distance functions satisfying (**I**) – (**III**), but we only consider distances for which (**I**) – (**V**) are satisfied unless otherwise mentioned. One would expect $d(\mathbf{P}, \mathbf{Q})$ to increase as dissimilarity or divergence between $\mathbf{P}$ and $\mathbf{Q}$ increases. Thus $d(\mathbf{P}, \mathbf{Q})$ is also described as a coefficient of dissimilarity even when it does not satisfy the metric properties (**IV**) and (**V**).

Frequently used distances:

Euclidean distance Let $\mathbf{X}$ be an $(n \times p)$ data matrix with rows $\mathbf{x}_1^T, \ldots, \mathbf{x}_n^T$. Then the Euclidean distance between the points $\mathbf{x}_i$ and $\mathbf{x}_j$ is d_{ij}, where

$$d_{ij}^2 = \sum_{k=1}^{p} (x_{ik} - x_{jk})^2 = \|\mathbf{x}_i - \mathbf{x}_j\|^2 \tag{3.34}$$

This distance function satisfies properties (**I**) – (**V**). It also satisfies the following properties:

1. *Positive semi-definite property:* Let $\mathbf{A} = -\frac{1}{2} d_{ij}^2$. Then $\mathbf{HAH}$ is p.s.d., where $\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^T$ is the centering matrix. This property is important when examining similarity coefficients.

2. d_{ij} is invariant under orthogonal transformations of the $\mathbf{x}s$.

3. The cosine law:
$$d_{ij}^2 = b_{ii} + b_{jj} - 2b_{ij} \tag{3.35}$$
where $b_{ij} = (\mathbf{x}_i - \bar{\mathbf{x}})^T (\mathbf{x}_j - \bar{\mathbf{x}})$ is the centered inner product between $\mathbf{x}_i$ and $\mathbf{x}_j$. Another useful identity for calculation purposes is given by
$$\sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}^2 = 2n \sum_{i=1}^{n} b_{ii} \tag{3.36}$$

Karl Pearson distance

When the variables are not commensurable, it is desirable to standardize (3.34); that is, we can use

$$d_{ij}^2 = \sum_{k=1}^{p} \frac{(x_{ik} - x_{jk})^2}{s_k^2} \tag{3.37}$$

where s_k is the variance of the kth variable. We shall call such standardized distance the "Karl Pearson" distance and denote it $\mathbf{K}^2$. The distance is then invariant under changes of scale. Another way to scale is to replace s_k by the range

$$\mathbf{R}_k = max_{ij}|x_{ik} - x_{jk}| \tag{3.38}$$

For cluster analysis one usually uses (3.37) except when the difference in scale between two variables is intrinsic, when one uses Euclidean distance.

Mahalanobis distance

The squared Mahalanobis distance between points $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined as:

$$\mathbf{D}_{ij}^2 = (\mathbf{x}_i - \mathbf{x}_j)^T \hat{\mathbf{\Sigma}}^{-1} (\mathbf{x}_i - \mathbf{x}_j) \tag{3.39}$$

The Mahalanobis distance underlies Hotelling's $\mathbf{T}^2$ test and the theory of discriminant analysis.

Properties:

1. Let $\mathbf{x} \sim (\boldsymbol{\mu}_1, \Sigma)$ and let $\mathbf{y} \sim (\boldsymbol{\mu}_2, \Sigma)$. Then $\mathbf{D}(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$ is a Mahalanobis distance between the parameters. It is invariant under transformations of the form: $\mathbf{x} \rightarrow \mathbf{A}\mathbf{x} + \mathbf{b}$, $\mathbf{y} \rightarrow \mathbf{A}\mathbf{y} + \mathbf{b}$, $\mathbf{\Sigma} \rightarrow \mathbf{A}\mathbf{\Sigma}\mathbf{A}^T$, where $\mathbf{A}$ is a non-singular matrix.

2. Let $\mathbf{x} \sim (\boldsymbol{\mu}, \Sigma)$. The Mahalanobis distance between $\mathbf{x}$ and $\boldsymbol{\mu}$, $\mathbf{D}(\mathbf{x}, \boldsymbol{\mu})$, is here a random variable.

3. Let $\mathbf{x} \sim (\boldsymbol{\mu}_1, \Sigma)$, $\mathbf{y} \sim (\boldsymbol{\mu}_2, \Sigma)$. The Mahalanobis distance between $\mathbf{x}$ and $\mathbf{y}$ is $\mathbf{D}(\mathbf{x}, \mathbf{y})$.

3.3.7 Factor Analysis

Factor analysis is a mathematical model which attempts to explain the correlation between a large set of variables in terms of a small number of underlying factors. A major assumption of factor analysis is that it is not possibly to observe these factors directly; the variables depend upon the factors but are also subject to random errors.

The Factor Model

Definition: Let $\mathbf{x}_{(p\times 1)}$ be a random vector with mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{\Sigma}$. Then we say that the k-factor model holds for $\mathbf{x}$ if $\mathbf{x}$ can be written in the form

$$\mathbf{x} = \mathbf{\Lambda}\mathbf{f} + \mathbf{n} + \boldsymbol{\mu} \tag{3.40}$$

where $\mathbf{\Lambda}_{(p\times k)}$ is a matrix of constants and $\mathbf{f}_{(k\times 1)}$ and $\mathbf{n}_{(k\times 1)}$ are random vectors. The elements of $\mathbf{f}$ are called *common* factors and the elements of $\mathbf{n}$ *specific* or *unique* factors. We suppose that:

$$E(\mathbf{f}) = \mathbf{0}, \qquad V(\mathbf{f}) = \mathbf{I}, \tag{3.41}$$

$$E(\mathbf{n}) = \mathbf{0}, \qquad C(n_i, n_j) = 0, i \neq j, \tag{3.42}$$

and

$$C(\mathbf{f}, \mathbf{n}) = \mathbf{0}. \tag{3.43}$$

Denote the covariance matrix of $\mathbf{n}$ by $V(\mathbf{n}) = \mathbf{\Psi} = diag(\Psi_{11}, \ldots, \Psi_{pp})$. Thus, all of the factors are uncorrelated with one another and further the common factors are each standardized to have variance 1. Note that $x_i = \sum_{j=1}^{k} \lambda_{ij} f_j + n_i + \mu_i, \quad i = 1, \ldots, p$, so that $\sigma_{ij} = \sum_{j=1}^{k} \lambda_{ij}^2 + \Psi_{ii}$. Thus, the variance of $\mathbf{x}$ can be split into two parts. First, $h_i^2 = \sum_{j=1}^{k} \lambda_{ij}^2$ is called the *communality* and represents the variance of x_i which is shared with the other variables via the common factors. In particular $\lambda_{ij}^2 = C(x_i, fj)$ represents the extent to which x_i depends on the jth common factor. On the other hand Ψ_{ii} is called the *specific* or *unique variance* and is due to the unique factor n_i. It explains the variability in x_i not shared with the other variables.

3.4 Summary

The theoretical fundamentals used for the computer vision related elements were presented. Each topic developed is an introduction to chapter 5 where the main results of this thesis regarding computer vision are presented. The deformable models (snakes) and the statistical elements introduced here will be enhanced and used in innovative ways to obtain the coronary model. The processes where deformable models are applied are the vessel detection and tracking from image sequences and $3D$ reconstruction of the coronary arteries from biplane image projections. The statistical components are incorporated to the deformable models for the vessel segmentation and tracking tasks.