TELECOM Bretagne

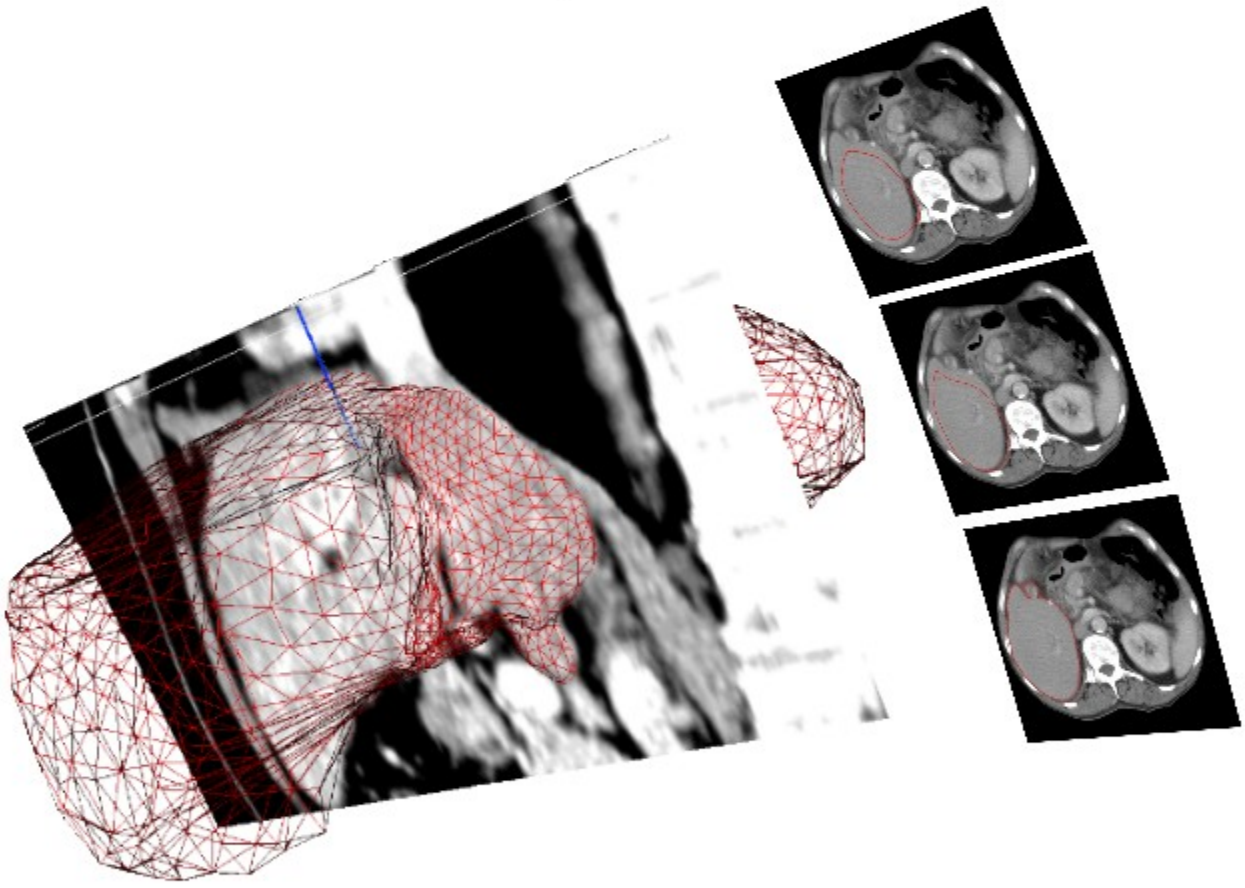
SISEA
Master Report

PHILIPS

Use of curvature features for Model Based Segmentation

Alberto Gomez

September 2009



Supervisor from
Télécom Bretagne

Valérie Burdin

Supervisors

Heinrich Schulz
Dr. Thomas Netsch

Acknowledgements

I thank all the people that have made my master thesis better, in any sense. I thank specially:

- Thomas Netsch, for his always challenging and motivating points of view, the constructive discussions, his guidance, his support and his help with this report; and Heinrich Schulz, for his help with all the paperwork, the programming environment, and his help during my PhD search.
- Valérie Burdin, for her trust, her kindness, and her guidance and help during my internship search and also my PhD search.
- The interns of Digital Imaging, for making my 6 months at Philips a very happy experience.
- The friends I met in the great city of Hamburg. They were a very important support in the difficult moments.
- My family, for their infinite love and patience.
- And last, but not less important, Dani, for teaching me that "Happiness is not real unless shared".

Abstract

Geometric features of surfaces such as ridges and valleys are being used for many applications in image processing, like object recognition, segmentation and shape matching [YBS05, ZGM09, SF04]. In this work, we investigated the computation and use of curvature features in model based segmentation [WPK$^+$] where a model, represented by a triangulated mesh, is adapted to an image containing the object of interest.

In particular, in liver CT images model based segmentation showed poor results at sharp regions of the object. Since curvature can be used to differentiate those regions, its potential to improve segmentation was explored. Therefore, methods to robustly calculate curvature features both in meshes and images were explored. Reliability of several methods was investigated and their suitability to establish correspondence of curvature in meshes and images studied.

Curvature principal directions were proven to constitute a promising feature due to robustness and invariance of their computation.

Contents

List of Figures

Chapter 1

Introduction

This document reports the work carried out at Philips Research (Hamburg, Germany) for my Master Thesis. I studied the Master SISEA in Image Processing at Télécom Bretagne (Brest, France) during the school year 2008-2009. This is a work on Medical Image Processing.

1.1 Philips Research

Royal Philips Electronics Inc., most commonly known as Philips is a Dutch electronics company. Philips is one of the largest electronics companies in the world. In 2007, its sales were €26.79 billion. The company employs 123,800 people in more than 60 countries.

Philips is organized in a number of sectors:

- Philips Consumer Lifestyle (formerly Philips Consumer Electronics and Philips Domestic Appliances and Personal Care)
- Philips Lighting
- Philips Healthcare (formerly Philips Medical Systems).

In parallel to these three bussines lines, there is Philips Corporate Technologies (PCT), which groups several organizations with a long history in contributing to the innovation in Philips and the development of new markets. Building on the resulting competencies, Corporate Technologies today creates and licenses options and Intellectual Property (IP), provides Research & Development (R&D) services and incubates new businesses.

Philips Research, the international research organization of Royal Philips, and an organization of PTC, is acting on a global basis with laboratories spread over Europe, North America and East Asia. Philips Research focuses on the areas of health and well-being.

The Healthcare program of Philips Research is structured into 5 focal areas:

- Medical Imaging Systems (MIS)
- Molecular Medicine (MOM)
- Monitoring and Treatment (MOT)
- Clinical Care Solutions (CCS)
- Emerging Markets Healthcare (EM-H)

Hamburg laboratory is divided into two teams:

- Tomographic Imaging Systems. This team is dedicated to research in hardware systems for Magnetic Resonance Imaging (MRI) and Computed Tomography (CT).
- Digital Imaging. This team is dedicated to research in medical image processing, and is the team that I was integrated in for my Masters Thesis.

1.2 Motivation

The detection and accurate segmentation of a particular structure in 3D gray scale images is of interest for many medical applications. In particular, the use of 3D models has proven to be a powerful tool for automatic segmentation. Generally, such methods adapt an initial 3D shape model to image structures under the constraint of preserving the model topology. In this document we describe a shape as an explicit triangulated surface. The adaptation of the shape is realized as an energy minimization process [WPK$^+$].

The adaptation has been proven to be robust, fast and accurate [KHL$^+$09]. However, we observed that the mesh adaptation demonstrated somehow poorer performance in the vicinity of sharp ridges of the shape model. The

objective of this work is to detect this regions and to study new features from the image and from the model that can improve the segmentation at ridges.

Sharp regions may be characterized by curvature features. In this document, we study curvature computation for ridge-like regions detection and the usability of curvature features in the Model Based Segmentation context (c.f. chapter 2).

1.3 Structure of the document

This document is structured in three parts. The first part corresponds to the bibliographic study done in the fields of differential geometry applied to triangular meshes and to images, active contours and Model Based Segmentation and ridge-like region detection. This part contains two chapters:

- Chapter 2 introduces the concept of Model Based Segmentation.
- All the mathematical basis required for ridge like regions detection are explained in chapter 3. In particular the same mathematical theory was adapted to discrete world in two different ways: for triangle meshes, as shown in section 3.1, and for 3D images in section 3.2.

The second part contains the study on how to merge results from meshes and from images and how they might be used to improve Model Based Segmentation, and constitutes the main contribution to the solution of the problem. This part corresponds to chapter 4.

The third part contains the conclusions of the study and an outlook on how this work might be continued. It contains two chapters:

- Chapter 5 analyzes the results obtained in previous chapters and summarize the main aspects of this work.
- Chapter 5.4 does an overview on the possibilities drawn from this study and what the future work might be.

There are five appendices containing supporting material:

- Appendix A is dedicated to explain the basics on differential geometry and curvature computation in real, continuous functions.

- Appendix B contains the definition of a new metric which is used for vector comparison.
- Appendix C contains technical details on how to obtain a local coordinate system in images and meshes.
- Appendix D gives a brief description on the visualization software that I have developed as a support for my research.
- Appendix E shows some experimental results that, for space reasons, have not been included in chapter 4.

Chapter 2

Model Based Segmentation

The aim of this chapter is to make a brief introduction to Model Based Segmentation (MBS). For more detailed explanations, refer to the Bibliography. The origins of active contours are referred in [KWT88]. For a detailed study on active shape models, refer to [CHTH94]. Most recent works on MBS can be found in [FRZ+05, EPH+08].

MBS is an automatic segmentation method that incorporates prior knowledge in a model, encoding high-level information such as shape and appearance. This information allows instances of the object to be robustly and accurately segmented.

Principle The principle of MBS assumes that structures of interest have repetitive form of geometry and appearance. Therefore, these characteristics may be statistically described and stored into an initial shape model. The key issue of MBS consists of adapting the initial model to image structures under the constraint of preserving model topology. These constraints are translated into an energy function which considers image and model information.

Energy function Model adaptation using shape constrained models is governed by a global energy function which is dependent on the image itself and on the model through two terms: an external energy and an internal energy term. The external energy tends to attract the model to image features such as a gradient or a specific gray value intensity whereas the internal energy ensures a good maintenance of model shape. The iterative procedure of model adaptation is performed by minimizing the sum of these regularization terms. They are weighted by a coefficient α which steers model deformation.

The global energy term is defined by

$$E = E_{ext} + \alpha E_{int} \tag{2.1}$$

where E_{ext} increases with the deviation of the model from the detected image features and E_{int} increases with the model deformation with respect to its initial shape.

Feature search The aim is to search for a target point $\tilde{x}_i$ per mesh vertex that optimizes a feature function along a search profile c_j defined by

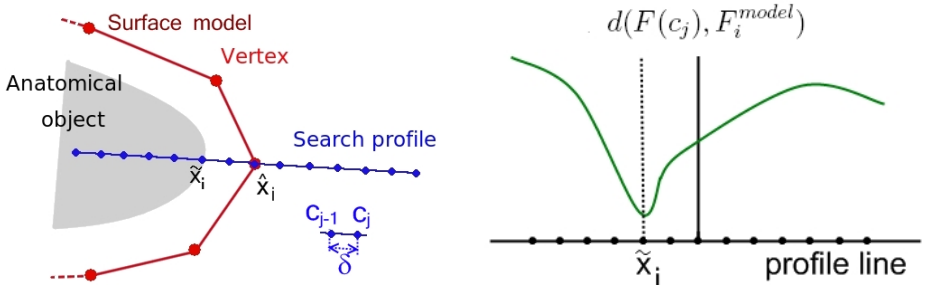
$$c_j = j\delta\vec{n}_i \tag{2.2}$$

where $j = -l \dots l$, resulting in $M = 2l + 1$ points along the profile with a sampling step δ. The search profile is centered in the vertex $\hat{x}_i$ and aligned with its normal $\vec{n}_i$. This is illustrated in figure 2.1(a).

The function that we want to minimize is in general some distance between the feature in the image and the feature stored in the model,

$$d(F) = d(F(c_j), F_i^{model}) \tag{2.3}$$

where $F(c_j)$ is a feature evaluated at the point c_j of the search profile, and F_i^{model} is the prior knowledge about that feature stored in the vertex i of the model (figure figure 2.1(b)). The definition of the distance depends on the feature.



(a) Feature search profile along the normal direction at the vertex $\hat{x}_i$, sampled from sampling point $-l$ to l with step size δ.

(b) Feature function example along the search profile.

Figure 2.1: Feature search profile along the normal direction in MBS

An iterative procedure is used to perform feature search along the line using sub-sample accuracy. In the first iteration the profile is centered in the vertex, once the best point $\tilde{x}_i$ is found for this iteration, the search profile is centered in $\tilde{x}_i$, l is set to 1 and the sample size δ is divided by half, therefore a new best point is found and the next iteration started until a minimum size δ_{min}.

Different formulations of feature evaluation functions $F(x)$ can be defined. For example gradient based feature function with a high response at object boundaries. The most promising points $\tilde{x}_i$, obtained from the feature search procedure are used for the calculation of the external energy.

External energy The external energy controlling the influence of image information on the adaptation process is given by

$$E_{ext} = \sum d(F(\tilde{x}_i), F_i^{model}) \tag{2.4}$$

So far we have only considered the contribution attached to the image information, the term that controls mesh deformation is the internal energy.

Internal energy The internal energy ensures that the adapted shape is close to the model restricting to a suitable distribution of the vertices. This is achieved by measuring the deviation of the adapted mesh vertices to those constituting the model. Penalizing deviations of the model shape to the reference shape, regularizes the image forces acting on it. Hence, only mesh shapes consistent with defined constraints are possible and the attraction of individual triangles to false image structures decreases and preserves shape similarity of all mesh vertices to the model vertices

$$E_{int} = \sum_{j,k \in edges(M)} \|(v_j - v_k) - T \cdot (\hat{v}_j - \hat{v}_j)\|^2 \tag{2.5}$$

where $(v_j - v_k)$ represents an edge of the adapted shape, and $(\hat{v}_j - \hat{v}_j)$ is an edge of the original model. T is the transformation (rotation, translation and homogeneous scaling) that minimizes the error between the adapted shape and the model. This way, the internal energy augments only if the adapted shape is different to the original model.

Chapter 3

Characterization of ridge-like regions through curvature analysis

In this Chapter we describe different methods that allow the characterization and detection of ridge-like regions in images and triangulated meshes. A ridge is a sharp region that extends along a line on the surface it lies on.

Notation In the document, we take the following conventions for notation:

- Scalars and points in the space are denoted by lower case letters (e.g the curvature k, the point p)
- Vectors are designed by lower case letters with an arrow (e.g. the gradient vector $\vec{g}$).
- Vector components are designed with subscripts (e.g. $\vec{n} = (n_x, n_y, n_z)$). Vector derivatives are also designed with subscripts, but they are not scalars (e.g. $\frac{\partial \vec{g}}{\partial x} = \vec{g}_x$).
- Matrices are designed by capital letters (e.g. the Hessian matrix, H)
- Functions are designed with capital letters (e.g. intensity function $I(x, y)$, or just I where there is no possibility of misunderstanding).
- A *normal* vector will stand for a vector which is orthogonal to a surface. A vector of unit length will be called a *normalized* vector.

- Directions are designed with normalized vectors unless otherwise indicated.

- When multiplying vectors, "$\times$" designates the cross product, and "$\cdot$" designates the dot product.

Mathematical characterization of ridge lines Intuitively, ridge lines are the curves on a surface along which the surface bends sharply [YBS05], producing crests and valleys (figure 3.1).

(a) Crest. (b) Valley.

Figure 3.1: Crests and valley in a surface

Although literature does not seem to converge to an unique description of a ridge [MvdEV96], the most common definition is the following : the sharp variation of the surface normals are described via extrema of the surface principal curvatures along their corresponding lines of curvature [LFM96]. For a more detailed description of the curvatures of a surface and their calculation in a continuous space, refer to Appendix A.

Using the definition from [MBF92], ridge lines are curves containing ridge points. At these points, one of the principal curvatures is locally maximal (in absolute value) along the maximal curvature direction. Ridge points may be detected from a simple thresholding to more complex methods like detection of zero-crossings of the "extremality" (i.e. the derivative of the curvature)[MBF92] , or analysis of first and second order derivatives of the curvature [LLSV99].

In all cases, ridge line detection requires first and second order derivatives computation. There exist well known methods for derivative estimation in images, that we describe in section 3.2. For meshes, we describe different methods that provide differential operators in section 3.1.

Considerations for discrete spaces We work with images and surface meshes, which are represented in a discrete space. This requires some important adaptations of the theory of differential geometry, as described in sections 3.1 and 3.2.

Differentiation is an ill-posed problem when applied to digital, sampled signals as opposed to smooth mathematical functions [MvdEV96]. Smoothing these signals regularizes this problem as showed in [FtHRKV]. Derivation is achieved by convolving the discrete function (e.g. the image) with the derivative of a Gaussian of width σ. This can also be understood in a more intuitive way as low-pass filtering the function to remove the high frequency variations due to discretization.

Redefinition of maximal and minimal curvature Curvature is a scalar magnitude that measures how bended a surface is at a point p, in a direction $\vec{\tau}$ tangent to the surface (c.f. Appendix A). Therefore, there is one curvature value, $k_{\vec{\tau}_i(p)}$, defined at p for each tangent direction $\vec{\tau}_i(p)$.

In the rest of the document, the convention from [OBS04], also used in [LLSV99, Lp00], for principal curvature classification will be used: we will call $k_1(p)$ the maximal curvature at p and $k_2(p)$ the minimal curvature, in absolute value:

$$\begin{array}{ll} \|k_1(p)\| \geq \|k_{\vec{\tau}_i}(p)\| & \forall i \\ \|k_2(p)\| \leq \|k_{\vec{\tau}_i}(p)\| & \forall i \end{array} \tag{3.1}$$

The directions, $\vec{\tau}_1(p)$ and $\vec{\tau}_2(p)$ associated to $k_1(p)$ and $k_2(p)$ are the principal (maximal and minimal, respectively) directions. For simplicity in the notation, we define the vectors $\vec{k}_1(p) = k_1(p)\vec{\tau}_1(p)$ and $\vec{k}_2(p) = k_2(p)\vec{\tau}_2(p)$.

3.1 Differential geometry operators on triangle meshes

3.1.1 Previous work on meshes

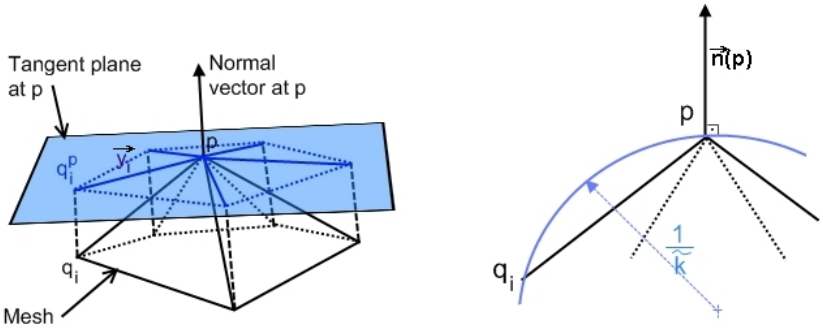
The surface models we work with are triangular meshes which consist of a list of points (vertex) and a list of triangles linking these points (topology). We want to compute principal curvatures and principal directions at every vertex.

For that, it becomes necessary to compute the principal curvature tensors of the surface (at each vertex). In [GI04] three ways of doing this are

compared. We have followed *The Normal Curvature Approximation Method*, which estimates the normal curvature in the direction of each edge passing through each peak[1]. This consists, for each edge, in the computation of the osculating circle that passes through the peak and its neighbor on that edge. From the family of circles verifying this condition, we choose the one that is orthogonal to the normal surface vector at the vertex. This yields a discrete differential operator valid for piecewise linear meshes [HPW05].

For each vertex p of the mesh, the Weingarten matrix W (c.f. Appendix A) gives the curvature in the direction of any tangent vector at each vertex. Following [GI04], we estimate this matrix to compute curvatures.

For each vertex p, the curvature $\tilde{k}$ is estimated in the direction of each edge going from p to each one of its neighbors q_i (figure 3.2(a)). This direction, $\vec{y}_i$, is the projection of the edge $\bar{pq}_i$ into the tangent plane at p. Then, the curvature for each direction is estimated as the curvature of the circle passing through p and q_i and orthogonal to the normal vector to the surface at p, $\vec{n}(p)$ [MDSB03], as shown in the figure 3.2(b).



(a) Estimation of tangent directions

(b) Estimation of the curvature in one tangent direction

Figure 3.2: Principal curvatures computation in meshes

This gives the following system of equations:

$$\tilde{k}_{\vec{y}_i}(p) = \vec{y}_i^T W(p) \vec{y}_i \tag{3.2}$$

[1]There are two other methods: The *Quadric Surface Approximation Method*, which has been proven to be equivalent to the one we use; and the *Adjacent-Normal Cubic Approximation Method*, which is more accurate but also more complex and requires more computational time.

It can be solved minimizing the expression $\|\tilde{k}_{\vec{y}_i} - \vec{y}_i^T W \vec{y}_i\|$. Then, the principal curvatures are the eigenvalues of W, and the principal directions are the associated eigenvectors.

3.1.2 Algorithmic details

Direction of the normal and curvature vectors The outer-pointing direction is taken as the positive orientation, for the normal vector $\vec{n}$. Regarding the principal curvature vectors, their orientation depends basically on the method used to compute the eigenvalues of W[2], i.e. only the direction is meaningful. For consistence, we impose, at every vertex, the orientation such that $\vec{k}_1 \times \vec{k}_2 = \vec{n}$.

Smoothing in meshes As we already mentioned at the beginning of this chapter, some smoothing is required to obtain the best curvature computation. The smoothing can be done in two ways:

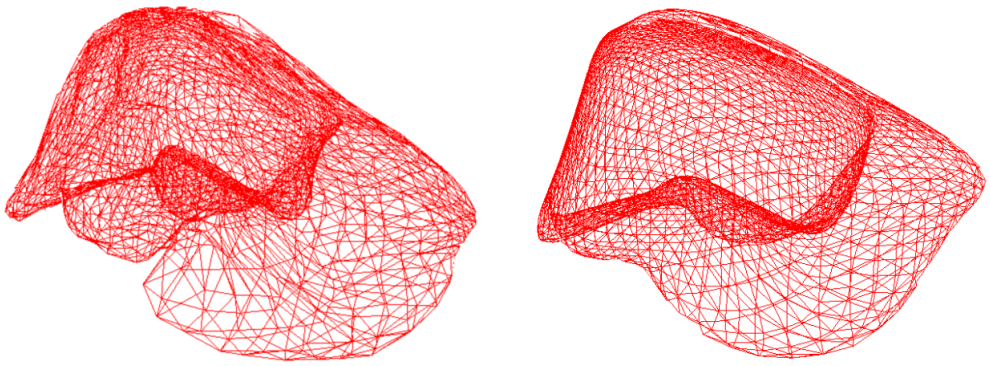
- Enlarging the neighborhood used for the curvature computation at each vertex. Taking into account vertex located in the n-ring ($n > 1$) helps the computation from being distorted for very local irregularities. Global fitting methods like in [KMW96, OBS04] use this approach.

- Smoothing the mesh prior to the curvature computation. This process reduces the sharpness of the surface and can be repeated for a higher degree of smoothing. Some smoothing methods have proven to retain desirable geometric features [DMSB99, HP04], but this is in general not the case.

The second smoothing technique is used for three reasons. First, using larger neighborhoods is computationally expensive. Second, with the second technique we can select the degree of smoothing in a finer scale, while with the first one we are restricted by the neighborhood size. And third, the meshes of interest are usually shape models, which have been generated already using a smoothing technique.

The degree of smoothing required depends on the sharpness of the surface with respect to the size and number of the triangles. To give an example on the degree of smoothing that is used in practice see figure 3.3(a). It shows a

[2]See the foot note in the Appendix B for a justification on this.



(a) Original mesh, obtained from a ground truth segmentation of a CT liver image.

(b) Smoothing version of the same mesh.

Figure 3.3: Triangulated meshes representing a liver meshes

mesh representing the surface of a liver. It has 2562 vertex and 5120 triangles. Figure 3.3(b) shows that same mesh after applying the smoothing.

3.2 Differential geometry operators on images

The application of differential geometry to 3D images uses discretized derivative operators [MvdEV96]. This operators are broadly used for image processing and well documented in the literature. Curvatures are obtained from first and second order derivatives of the intensity function $I(x, y, z)$. Several approaches exist to estimate the derivatives of the intensity function, like [Der87] used in [MBF92, MB92] or [MvdEV96].

3.2.1 An overview on curvature and ridge-like regions computation methods

There are two cases to be considered:

- Images where the surfaces are defined by isophotes (i.e. surfaces of constant intensity). This case includes binary images.
- Images where surfaces present variation in gray level. This is the most general case, and medical images are globally of this type; that is why

segmentation based in global thresholding of the intensity value is not feasible.

3.2.2 Images containing iso-surfaces

When the surface of an object in an image has a constant gray value, the Weingarten matrix W can be computed at each voxel. Then, curvatures are obtained by an eigen-analysis of W as we did for meshes (c.f. section 3.1).

The curvature of isophotes, i.e. the isophote curvature, $\kappa_{\vec{\tau}}$ is computed along the direction $\vec{\tau}$. For simplicity of the explanation, this curvature is defined in the 2D case and then extend it to 3D [MvdEV96]:

In the 2D case, the intensity function can be interpreted as the parametrization of a surface in $\mathbb{R}^3$. Let $\vec{g} = (I_x, I_y)$ be the gradient of I, which is oriented in the direction perpendicular to isophotes. Let θ be the orientation of the 2D gradient, $\theta = \arctan(\frac{I_y}{I_x})$[3]. Let $\vec{\tau}$ be the direction tangent to the curve. Then, $\vec{\tau}$ is orthogonal to $\vec{g}$, so

$$\vec{\tau} = \frac{(I_y, -I_x)}{\|\vec{g}\|} \tag{3.3}$$

and κ is the variation of θ along the direction $\vec{\tau}$

$$\kappa_{\vec{\tau}} = \frac{\partial \theta}{\partial \vec{\tau}} = \frac{2I_x I_y I_{xy} - I_y^2 I_{xx} - I_x^2 I_{yy}}{(I_x^2 + I_y^2)^{3/2}} \tag{3.4}$$

yields the expression,

$$\kappa_{\vec{\tau}} = -\frac{\vec{\tau}^T H \vec{\tau}}{\|\vec{g}\|} \tag{3.5}$$

where H is the Hessian matrix.

Comparing equation 3.5 with equation A.14, it is evident that $W = -\frac{H}{\|\vec{g}\|}$.

The same expression is obtained in [MB92] starting from a different point, as follows. Given that the gradient and the tangent vector are orthogonal (which is true for isophotes), i.e. $\vec{g} \cdot \vec{\tau} = 0$, we have

$$\frac{\partial(\vec{g} \cdot \vec{\tau})}{\partial s} = \frac{\partial \vec{g}}{\partial s} \cdot \vec{\tau} + \vec{g} \cdot \frac{\partial \vec{\tau}}{\partial s} = 0, \tag{3.6}$$

where ∂s is a differential arc length of the contour. We also know for the definition of curvature (c.f. appendix A) that

[3] If $I_x = 0$, then $\theta = \frac{\pi}{2}$

$$\frac{\partial \vec{\tau}}{\partial s} = k\vec{n} = k\frac{\vec{g}}{\|\vec{g}\|} \tag{3.7}$$

Applying the chain rule to the derivative of the gradient we get

$$\frac{\partial \vec{g}}{\partial s} = \frac{\partial \vec{g}}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial \vec{g}}{\partial y}\frac{\partial y}{\partial s} = H\vec{\tau} \tag{3.8}$$

Merging the results from equations 3.6, 3.7 and 3.8, the result is exactly the same as in equation 3.5:

$$\begin{gathered}
\tfrac{\partial \vec{g}}{\partial s}\cdot\vec{\tau} + \vec{g}\cdot\tfrac{\partial \vec{\tau}}{\partial s} = 0 \quad and \quad \tfrac{\partial \vec{g}}{\partial s}\cdot\vec{\tau} = \vec{\tau}\cdot\tfrac{\partial \vec{g}}{\partial s} = [\vec{\tau}]^T\cdot\left[\tfrac{\partial \vec{g}}{\partial s}\right] \quad in \quad matrix \quad notation \\
\vec{\tau}^T H\vec{\tau} + \vec{g}\cdot k\tfrac{\vec{g}}{\|\vec{g}\|} = 0 \\
\vec{\tau}^T H\vec{\tau} = -k\tfrac{\vec{g}\cdot\vec{g}}{\|\vec{g}\|} = -k\|\vec{g}\| \\
k = -\tfrac{\vec{\tau}^T H\vec{\tau}}{\|\vec{g}\|}
\end{gathered} \tag{3.9}$$

This conclusion can be extended to 3D. For that, the direction $\vec{\tau}$ is extended to any direction in the tangent plane, defined by (i.e. orthogonal to) the gradient direction. In 3D, curvatures are then computed as the eigenvalues of the W restricted to the tangent plane.

To project W into the tangent plane, local coordinate system is defined for each voxel, where one of the vectors of the local basis is aligned with the gradient (normal to the object) and the other two are in the tangent plane. This gives the transformation matrix

$$P = \begin{bmatrix} n_x & t_x^1 & t_x^2 \\ n_y & t_y^1 & t_y^2 \\ n_z & t_z^1 & t_z^2 \end{bmatrix} \tag{3.10}$$

where $\vec{n} = (n_x, n_y, n_z)$ is the normalized vector orthogonal to the surface (i.e., the gradient normalized: $\frac{\vec{g}}{\|\vec{g}\|}$), and $\vec{t^1}$ and $\vec{t^2}$ are two arbitrary normalized vectors, orthogonal to each other and to $\vec{n}$ (c.f. appendix C).

First, W is expressed in the local basis which is aligned with the normal vector $\vec{n}$. This yields $W^P = P^T W P$. Since the first vector in P is $\vec{n}$, the

2D Weingarten matrix in the tangent plane coordinates is obtained removing the first column from W^P

$$W^P = \begin{bmatrix} . & . & . \\ . & w_{00} & w_{01} \\ . & w_{10} & w_{11} \end{bmatrix} = \begin{bmatrix} . & . \\ . & W_{2D} \end{bmatrix} \tag{3.11}$$

Then, principal curvatures are the eigenvalues of W_{2D}, and principal directions are the associated eigenvectors. Once obtained, the inverse transform has to be applied to the eigenvectors to bring them to the 3D space again. Note that this approach meets the theoretical curvature values, which allows value matching with for example meshes.

3.2.3 General gray-level images

If the contours in the image do not have a constant gray value, the gradient at an edge point only approximates the normal to the surface and differential operators are not invariant. This motivated the second approach proposed in [MB92], where curvatures are obtained from curvatures in a hyper-surface defined on $\Re^4$; however, this approach requires some assumptions in the gray level distribution of the image which makes it less general. Nevertheless, the gradient is assumed to be a good enough estimator for the normal vector. Since curves are not isophotes any more, another approach is required. Theory is explained on 2D images and then extended to 3D.

In [MvdEV96], curvature is estimated from the Hessian matrix H of an image. Let $I(x, y)$ be the intensity function. Then a gradient-based local coordinate system is defined (note that the gradient approximates the direction normal to the contours):

$$\vec{t} = (I_y, -I_x) \qquad \vec{g} = (I_x, I_y) \propto \vec{n} \tag{3.12}$$

If $I(x, y)$ is represented as a parametrized surface in $\Re^3$, the gradient at any point points towards the ridges (with some exceptions, c.f. [MvdEV96]), and in the ridge it is aligned with the ridge line (unless the ridge is perfectly flat). From one side to the ridge to the other, the gradient reverses its direction.

In the case of a crest, on the ridge $\vec{t}$ is perpendicular to the ridge line. Therefore the intensity profile along $\vec{t}$ will be concave compared to the neighborhood of non-ridge points. Then, the second order derivative in that direction, $I_{\vec{t}\vec{t}}$, will have a local minimum at the ridge (a local maximum if it

is a valley instead of a crest). This means that the $I_{\vec{tt}}$ operator measures ridgeness. This operator can be computed as follows:

$$I_{\vec{tt}} = \frac{1}{\|\vec{t}\|^2}(\vec{t}\cdot\nabla)^2 I = \frac{I_y^2 I_{xx} - 2I_x I_y I_{xy} + I_x^2 I_{yy}}{(I_x^2 + I_y^2)} \tag{3.13}$$

Note that expression 3.13 is the same as expression 3.4 (for isophote curvature) but with a -1 factor and divided by $\|\vec{g}\|$.

In 3D, the tangent vector is actually a tangent plane. Therefore, the direction $\vec{t}$ has to be chosen:

1. In the plane normal to the local gradient (tangent plane).

2. In the direction in this plane for which second directional derivative is minimum.

Finding this direction requires a minimization process. This problem is equivalent to finding the eigenvector associated to the maximal eigenvalue of H_{2D} , which is the projection of H into the tangent plane, in the same way we did in the previous section. For convenience, we multiply by -1 the Hessian matrix. This way, the only difference with the isophotes case is a $\frac{1}{\|\vec{g}\|}$ factor.

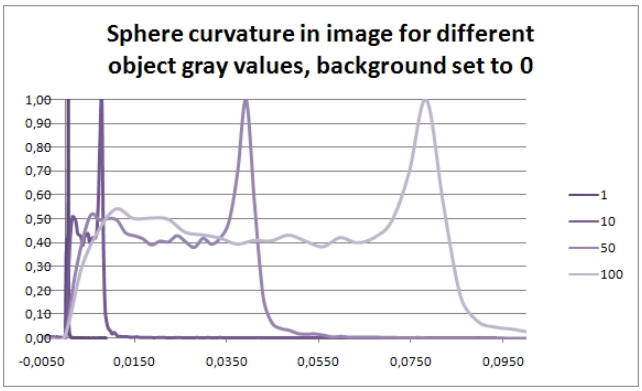
This approach in not invariant regarding transformations in gray values. This means that the same shape, represented in two images with different gray values will result in different curvature values. Moreover, for a given image containing different objects with a certain range of gray values, curvature value may be meaningless.

The derivative operation in an image is computed as the convolution of a derivative mask G (first or second derivative of the Gaussian) with the gray level value function, $I(x, y, z)$, which is a linear operation. This means that the curvature value for a given object will vary linearly with the gray value

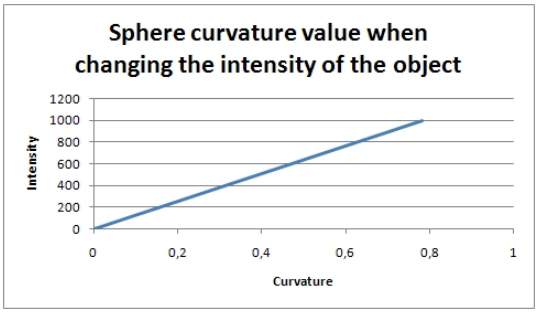
$$A \times (I(x, y, z) \otimes G) = (I(x, y, z) \times A) \otimes G, \tag{3.14}$$

where A is an arbitrary constant.

To verify this we compute curvature value in a binary sphere (constant curvature value), of radius $10mm$ (curvature value of $0.1\frac{1}{mm}$), and vary the gray level value (figures 3.4(a) and 3.4(b)).



(a) Curvature of the sphere for several intensity values



(b) Curvature value where the histogram is at its maximum (histogram peak) for different intensity values

Figure 3.4: Influence of the smoothing factor from the second order derivative in the curvature computation

This means that for a given object in an image, expected curvature value is only achieved for a particular gray value of the object. Then, curvature value is not sufficient for object detection (unless gray level distribution is a part of the prior knowledge, for example). Though, the curvature sign remains meaningful. We will use this information in the next sections, paying special attention to the zero crossings of the curvature value.

3.2.4 Discussion on the two approaches

The two approaches presented already result in two ridge seeking operators : $o_1 = -\frac{I_{\vec{n}\vec{n}}}{\|\vec{g}\|}$ (for images where contours have a constant gray value) and

$o_2 = -I_{\vec{n}\vec{n}}$. In a more general way, the operator $o_3 = -\frac{I_{\vec{n}\vec{n}}}{\|\vec{g}\|^\alpha}$ with $\alpha \in [0, 1]$ is investigated in [MvdEV96, LLSV99].

As presented in [MvdEV96], o_1 is insensitive to intensity transformations, which means that it is invariant for different gray level values, and gives the expected curvature values. This is desired for value matching. However, o_2 ignores spurious structures in the image background; the fact of dividing by the gradient shows up high curvature values in regions where the gradient is low. Finally, o_3 controls the trade-off between o_1 and o_2. For a α closer to 1, the range of real values of the curvature decreases, making more difficult to classify regions.

3.2.5 Algorithmic details

3.2.5.1 First order derivatives

Reliability of the normal vector estimation The normal vector estimation is robust even when the surface is not of constant gray level. Let $I(x, y)$ be the intensity function of a 2D image. $I(x, y)$ can also be considered as the parametrization of a 3D surface describing the gray level of the image. Then, a differential area of the surface is a square where the sides are $\frac{\partial I(x,y)}{\partial x}$ and $\frac{\partial I(x,y)}{\partial y}$. This means that the vectors $(1, 0, \frac{\partial I(x,y)}{\partial x})$ and $(0, 1, \frac{\partial I(x,y)}{\partial y})$ are in the tangent plane of the surface at (x, y). Then, the cross product of these two vectors is in the direction of the vector normal to the surface:

$$\vec{N}(x, y) = (1, 0, \frac{\partial I(x, y)}{\partial x}) \times (0, 1, \frac{\partial I(x, y)}{\partial y}) = (-\frac{\partial I(x, y)}{\partial x}, -\frac{\partial I(x, y)}{\partial y}, 1) \tag{3.15}$$

As the true image is in 2D, the projection of $\vec{N}$ on the XY plane gives the vector normal to the object in 2D: $\vec{N}_2D(x, y) = (-\frac{\partial I(x,y)}{\partial x}, -\frac{\partial I(x,y)}{\partial y})$. This is, as we wanted to prove, the gradient of the intensity function, times a -1 factor. This is extensible to any dimension, in particular to 3D.

Orientation of the gradient direction As indicated in the previous paragraph, there is factor of -1 that multiplies the normal vector. For this reason, to be consistent with the computation in meshes, this factor is applied to the gradient to produce a positive, outwards oriented normal vector.

Smoothing prior to the derivative computation For Hessian and gradient computation, derivative filters are used, but as the derivative operators

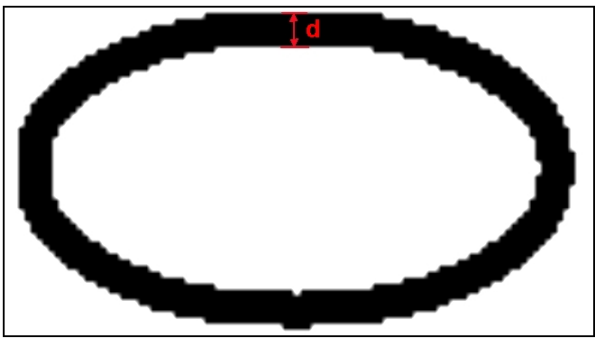


Figure 3.5: A slice of the synthetic image used for the smoothing study. The gap width d is highlighted in red. Dimensions are $104 \times 52 \times 52$ voxels with an isotropic spacing of $1mm$, which is of the order of the CT images we work with.

are sensible to noise, a smoothing step will improve the result [MBF92]. In practice, the derivative of a Gaussian (derivative and smoothing filter, implementation proposed by Deriche [Der87]) is used, to perform both smoothing and derivation in one step. This filter assures that the differentiation problem is well-posed in this context [FtHRKV]. The smoothing factor is controlled by the σ of the Gaussian.

The higher the order of the derivative is, the more sensible it becomes to noise. That is why the σ for the 2^{nd} order derivative is higher than for the 1^{st} order ($\sigma_2 > \sigma_1$).

To find the optimal smoothing for the gradient, we use a the synthetic image of the figure 3.5 (a 3D ellipsoid surrounded by a solid region). For the study, the width of the gap is modified to see the impact of the smoothing on the ability to detect narrow separations between objects.

To find the optimal smoothing , the metric defined in appendix B is used. The normal vector computed from the image (i.e. the gradient normalized) and the normal vector computed from the mesh are compared. It is assumed that the mesh is smooth enough to provide perfect normal vectors (figure 3.6).

Experiments are run for a gap width $d \in \{2, 4, 6, 8, 10, \infty\}mm$, and $\sigma_1 \in [1, 5]$. Figure 3.7 shows how for a $d > 4mm$, $\sigma_{1,optimal} \approx 2.5mm$. However, a $\sigma_1 \in (2, 4)$ can be used since variation within this range is almost null.

For $d \leq 4mm$, σ_1 has to be reduced to avoid structure overlapping due to smoothing. For that reason, σ_1 has to be lower than d, which means that

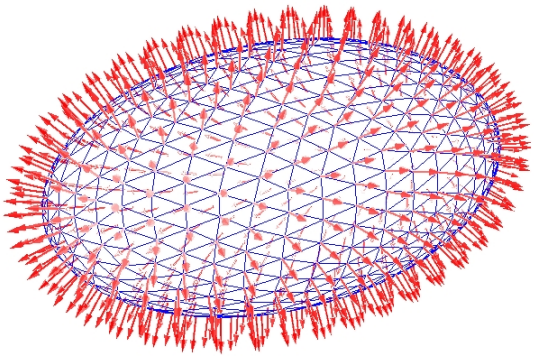
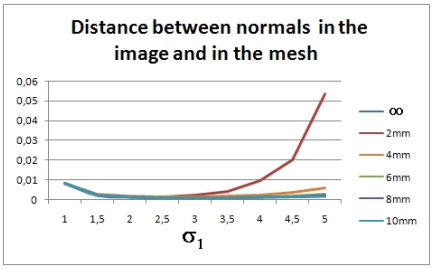
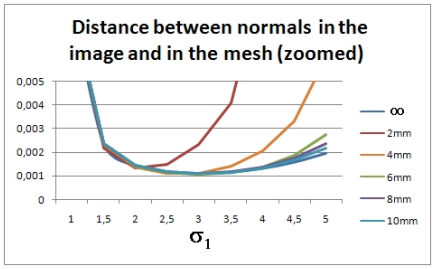


Figure 3.6: Normal vectors in the ellipsoid surface mesh.

for $d < 2mm$, vector matching is not feasible at this resolution (voxel size is $1 \times 1 \times 1mm$).



(a) Distance between normal vectors computed from the image and from the mesh for different gap widths and different σ_1 values.



(b) Distance between normal vectors computed from the image and from the mesh for different gap widths and different σ_1 values, zoomed in.

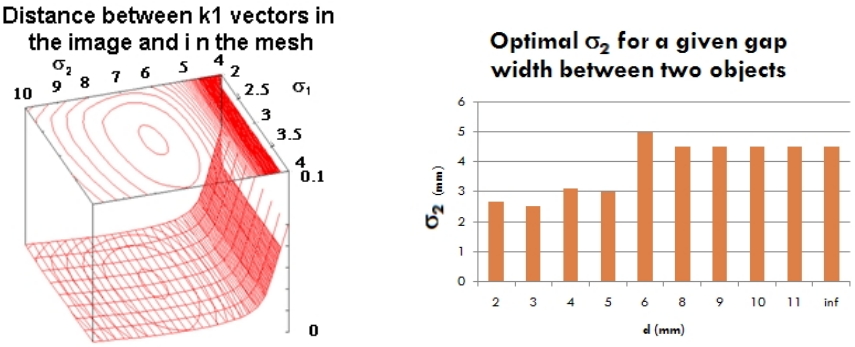
Figure 3.7: Influence of the smoothing in the gradient computation.

3.2.5.2 Second order derivatives

Gap between two objects and resolution Second order derivatives are more sensible to noise than first order's. This means that we need a higher σ for them. Also, the optimal σ_1 for an accurate gradient computation establishes a lower bound for σ_2. For the experiments in this section, the metric described in appendix B is used with the maximal principal direction, $\vec{k}_1$.

Metric value depends more in σ_2 than in σ_1, as shown in figure 3.8(a) for $d = \infty$. Therefore, the optimal σ_1 found in the previous section is used, then

σ_2 is modified.



(a) Distance between $\vec{k}_1$ computed from the image and from the mesh for different σ_1 and σ_2 values.

(b) Optimal σ_2 for different gap widths.

Figure 3.8: Influence of the smoothing in the gradient computation.

Figure 3.8(b) shows how for $d > 5mm$, $\sigma_{2,optimal} \approx 3mm$. If $d < 6mm$, σ_2 has to decrease for the same reason as for σ_1.

Note that the fact of using different σ in the first and the second order derivative introduces a small error in the curvature value when using the operator o_1 (c.f section 3.2.4).

3.2.6 Curvature sign in images

Another point one should pay attention to is the following: in meshes the normal vector has been defined as an outer-pointing vector. To be coherent with meshes, in images it has to be the same. However, the orientation of the gradient depends on whether the object is represented in a brighter or darker intensity than the background. Using typical derivative masks, as for example, in 2D along x and y:

$$\begin{bmatrix} -1 & 0 & 1 \\ -k & 0 & k \\ -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -k & -1 \\ 0 & 0 & 0 \\ 1 & k & 1 \end{bmatrix} \tag{3.16}$$

then if the object's intensity is higher than the background (as it is, for convention, in binary images), one can verify that the vector normal to the surface is an inward-pointing vector. Let our case be that, the sign has to be inversed to remain coherent with the curvature of the mesh.

3.3 Building up ridge lines out of curvatures

3.3.1 Overview on methods for ridge-lines detection

Once the differential operators are available, ridge points are detected from the principal curvatures of the surface. Ridge lines are the ensemble of these points. There are two cases: ridge line detection in meshes (section 3.3.2) and in volumetric gray scale images (section 3.3.3).

3.3.2 Ridge lines in meshes

Ridge lines detection in meshes uses the discrete differential operators described in section 3.1 to compute the principal curvatures at each peak. Then, all the solutions proposed start by local maxima and minima search of this quantity, which involves the derivative of the principal curvatures (the extremality [Thi96]). This yields a set of points marked as potential crest lines points.

Different approaches have been proposed to build up ridge lines from these candidate points. In [OBS04] the candidate points that are neighbors are connected. Then the strength of the each ridge line is measured by the integral of the maximal principal curvature along the line, and this value is used for thresholding "weak" crest lines.

In [SF04] the principal curvature of each peak is compared with its direct neighbors in the principal curvature direction to check if it is a local maxima or not, and this gives a set of crest points. The region made of this ensemble of crest points is grown (e.g. by means of morphological operations) to eliminate discontinuities. This yields a connected region which is skeletonized, again with morphological operations. This finally gives a set very fine connected lines on the edges of the triangulation. Advantages of this method are that it assures good positioning, line continuity and unicity of the crest lines.

In [YBS05] the crest points are detected as in [OBS04], but as in [SF04] they try to avoid crest line fragmentation. To do this, the 1-ring neighborhood of each candidate peak is inspected and if a criteria is satisfied then the candidate points are connected. Then the line strength is computed but this time operators of a superior order are used. This finally gives better results than in [OBS04] with a yet fast method.

In [HPW05] the important contribution is a smoothing stage in which extremalities are filtered to stabilize the computation and obtain high quality

crest lines. This consist in classifying triangles in *regular*, where the extremality computation is immediate, and *singular*, where neighbor triangles are used to compute the extremality. Then, a smoothing operator (e.g. a Laplacian operator as defined in [MDSB03]) is applied on the extremality values. Finally, these values are thresholded with the method proposed in [OBS04]. This solution produces high quality crest lines, because the smoothing stage reduces the noise and gives some "contextual" information.

A clear and complete summary and comparison of these methods can be found in [YBYS08].

In [ZGM09] an important improvement is introduced in the crest lines detection scheme: two types of feature are used, so-called *local* and *contextual* features (while the previous approaches remained local). Crest points are classified as in [SF04]. Then principal curvatures are computed on the K-neighborhood, and the values for neighbors at different distances are compared. This yields the "similarity function", that is exploited together with the classified points. This function reveals the difference between the vertex and its surrounding context, which makes the algorithm very robust with respect to noise and irregularities.

3.3.3 Ridge lines in 3D images

In volumetric images as well, ridge lines are obtained with the first, second and, for some methods, third order derivative of the intensity function.

In most cases [MBF92, MvdEV96, LLSV99], the "standard" approach is used: principal curvatures are computed from first and second derivatives of the intensity function, and extremality functions are computed from the derivative of the principal curvatures. The extremality is used then to determine the ridge point using a threshold.

This approach assumes that ridge lines have a constant gray value; if this is not true, the gradient at an ridge point does not approximate the normal to the surface and differential operators are not invariant. This motivated the second approach proposed in [MB92], where curvatures are obtained from curvatures in a hypersurface defined on $\mathbb{R}^4$ as we introduced in section 3.2.1.

Other solutions to this problem, based on Gaussian smoothing, have been proposed in [MvdEV96] with good results, or in [LLSV99] where several ridge lines detecting methods are evaluated, including a multi-scale Gaussian smoothing for regularization purposes prior to the detection.

Chapter 4

Exploiting curvature information for object detection

This chapter studies how curvature information is useful for object detection in an image having a certain prior knowledge on its shape or other features. This prior knowledge may be contained in a shape model (i.e. a triangulated mesh).

Curvature information consist on two different informations:

1. Curvature value, k, describes how much a surface is bended along a certain direction. Directions where maximal (k_1) and minimal (k_2) curvature are achieved, are called principal curvatures. The mean curvature (noted H) is the average of the curvature value along every direction of the surface (also defined as $H = \frac{k_1+k_2}{2}$); the Gaussian curvature is $K = k_1 k_2$.

2. Curvature direction is the direction in the tangent plane where curvature value is computed along.

In the following sections, both curvature value and direction are analyzed.

4.1 Curvature in meshes and binary images

4.1.1 Curvature value

When working with binary images, we use the o_1 operator described in 3.2.2 for curvature computation. This operator is invariant to gray value transfor-

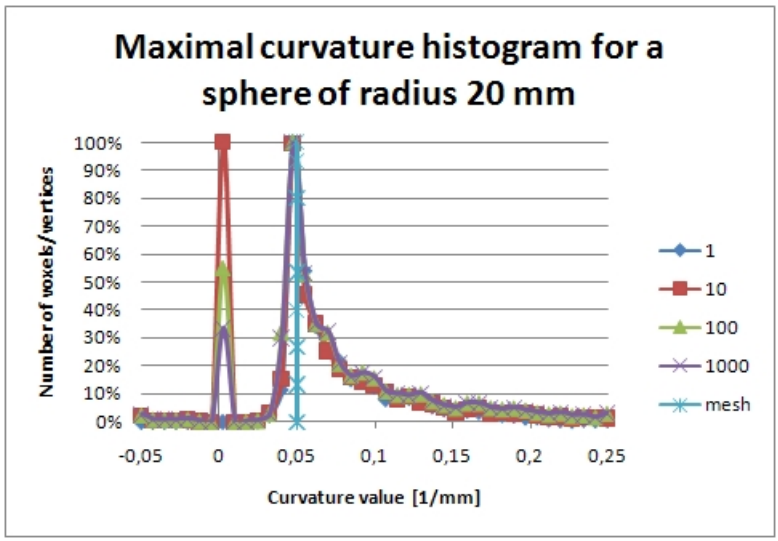


Figure 4.1: Curvature histogram for a binary sphere at different gray values. The background was set to 0 in all cases. The smoothing rate used was $\sigma_1 = 2$ and $\sigma_2 = 3$. The figure also shows the curvature computed for a mesh.

mations and thus the curvature value computed from the image approximates the value computed from meshes. However this value remains an approximation because:

1. The difference in the first order and second order derivatives smoothing $\sigma_2 \geq \sigma_1$ (c.f. section 3.2.5).

2. The smoothing of the mesh. In general, this smoothing modifies the shape of the mesh and as a consequence, curvature changes (c.f. section 3.1).

Accuracy of curvature value computation was tested in a sphere (constant curvature value), an ellipsoid and a ground truth segmentation of a liver.

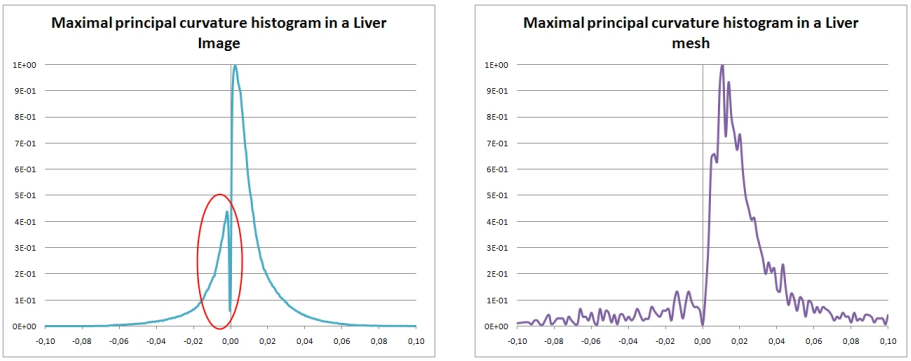
Figure 4.1 shows the goodness of the matching in the case of the sphere, where we have represented the curvature histogram. The sphere represented in a set of binary images, with gray value I ranging from 1 to 1000. The sphere had a radius of 20mm and a voxel size of $1 \times 1 \times 1mm$. In all cases the theoretical curvature value was achieved ($0.05\frac{1}{mm}$). However, there are many points with curvature values close to zero. This is due to the spurious

structures that appear in the image background when using the operator o_1 (c.f. section 3.2.4). The higher the gray value is, the less the spurious structures appear.

Curvature value distribution was also studied. Figures 4.2(a), 4.2(b), 4.3(c), 4.3(b), 4.4(b) and 4.4(c) demonstrate that the curvature distribution on the surface of both the binary image and the mesh match.

The 5% of the points with highest curvature in images was filtered to get the ridge lines. In meshes, the 25% was needed to obtain the same regions.

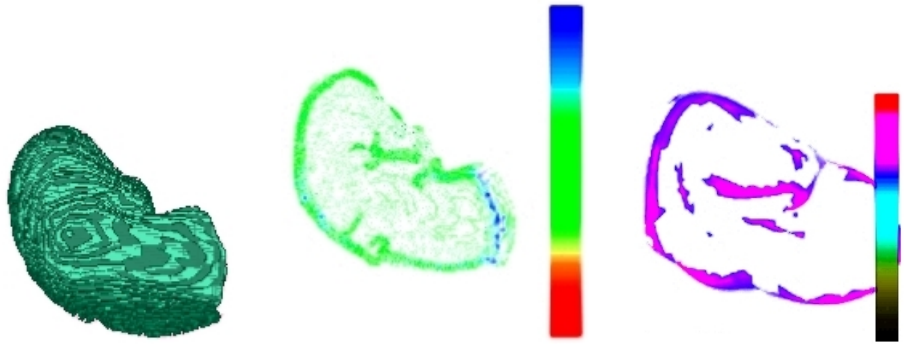
This is explained by a curvature histogram study. In the histogram of an image (figure 4.2(a)), there is a peak of negative curvature close to zero. This is due to spurious structures that appear in the background as mentioned in section 3.2.4. When removing this peak, by thresholding, no significant changes were noticed in the image. This peak is not present in meshes (figure 4.2(b)), and even for different smoothing factors, this peak represents the 20% of extra points that have to be removed in images.



(a) Curvature histogram for an image (b) Curvature histogram for a mesh

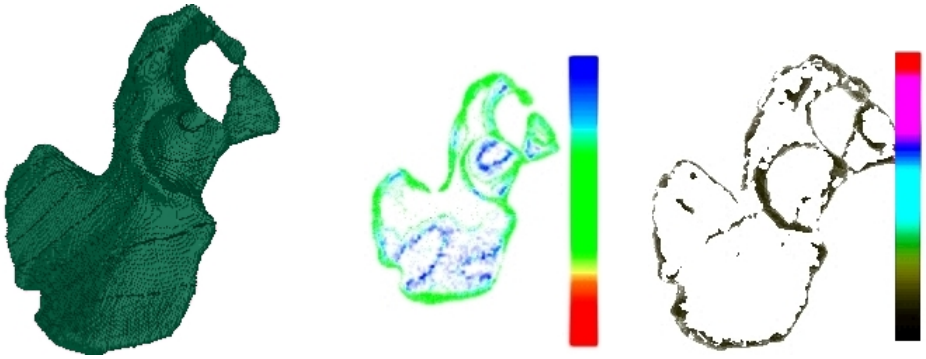
Figure 4.2: Liver curvature histogram comparison between meshes and images

The thresholded curvature images are shown in figures 4.3(c) and 4.3(b). Exactly the same thresholds have been used on a femur, also obtaining satisfactory results as in figures 4.4(b) and 4.4(c).



(a) Original liver binary image (b) Thresholded liver image (c) Thresholded liver mesh

Figure 4.3: Curvature computation in a liver



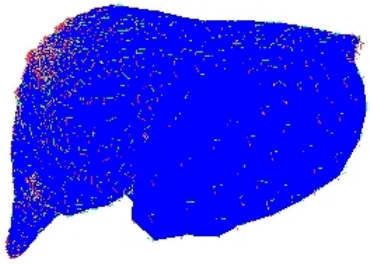
(a) Original femur binary image (b) Thresholded femur image (c) Thresholded femur mesh

Figure 4.4: Curvature computation in a femur

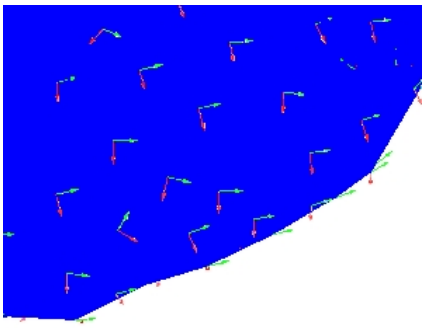
These results indicate that it is possible to directly compare in an absolute scale the curvature value between a binary image and a triangulated mesh.

4.1.2 Curvature directions

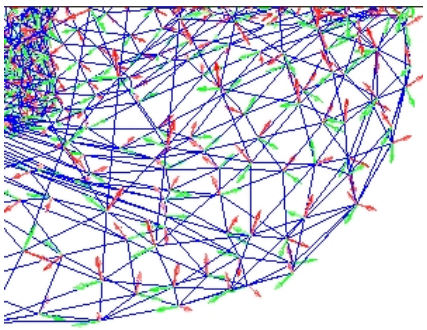
In this section we study the matching of principal directions between meshes and binary images. For this, we use the distance definition from the appendix B. Figure 4.5(a) to 4.5(c) show the principal directions computed on a mesh and on an image.



(a) Curvature directions in a liver binary image. Maximal curvatures are indicated with red vectors, and minimal curvature with green vectors.



(b) Detail of a ridge from the image.

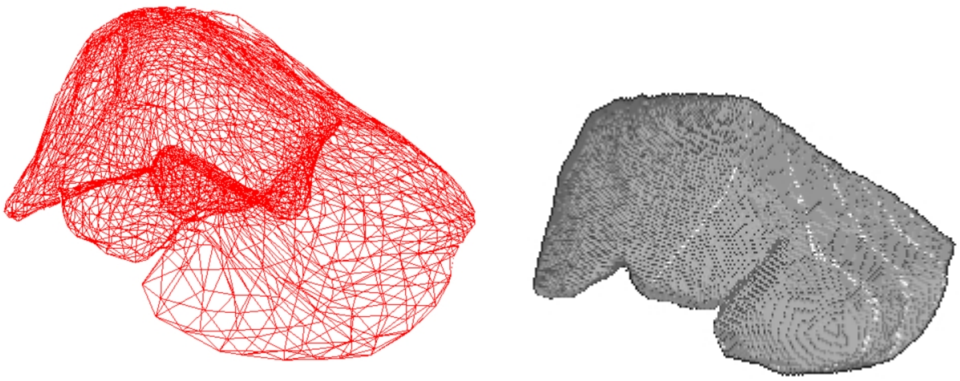


(c) Detail of a ridge from the mesh.

Figure 4.5: Principal curvature directions computed on a liver mesh and a liver binary image.

Principal curvature direction also showed good matching between binary images and meshes. We computed the distance between the maximal curvature direction in an ellipsoid and in a liver. The ellipsoid showed a high accuracy in the matching (approximately $1\,^{circle}$ of mean misalignment) except for the poles. In the poles, the ellipsoid is locally sphere-shaped, so principal directions are arbitrary because $k_{\vec{\tau_i}}$ is constant $\forall i$ (figures 4.7(a) and 4.7(b)).

In the liver, the matching resulted in a mean misalignment of $27.24\,^{circle}$ (figures 4.8(a) and 4.8(b)). The matching was done using a binary liver and a ground truth mesh (figure 4.6).

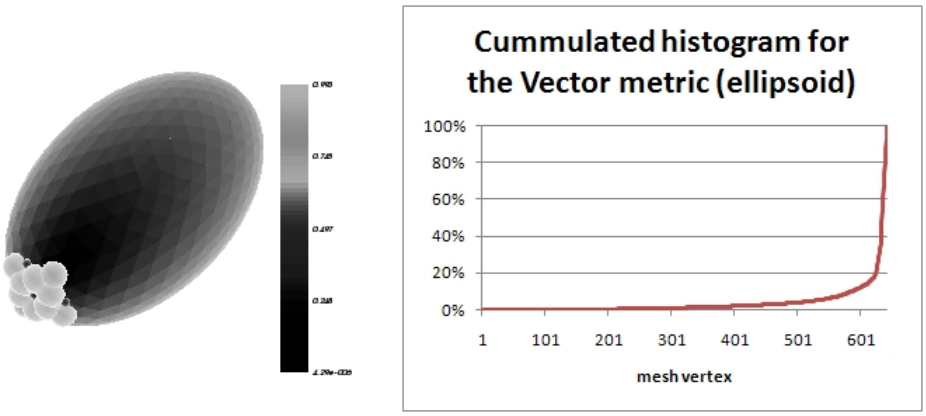


(a) Binary image of the liver.

(b) Ground truth segmentation of the liver

Figure 4.6: Binary image and mesh used for matching tests.

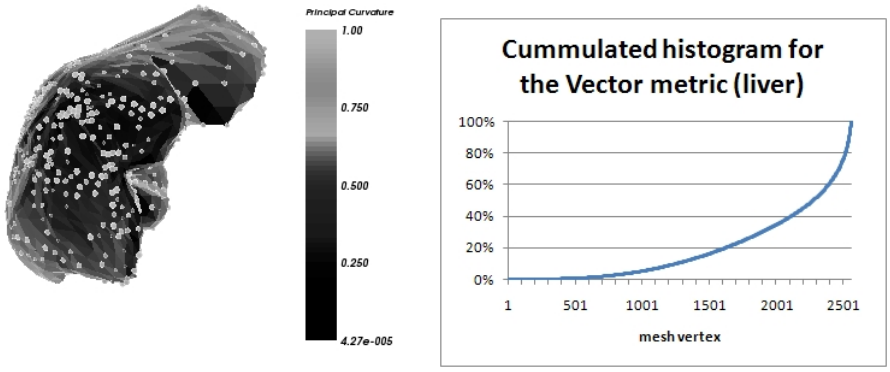
Although this mesh fits perfectly the liver image, it needs to be smoothed as explained in section 3.1. This explains why the error is located at sharp ridges and at regions where both curvatures are very similar or null (sphere-like or very flat).



(a) Distribution of the distance between maximal curvature directions from a mesh and a binary ellipsoid.

(b) Cumulated histogram of the distance in an ellipsoid

Figure 4.7: Vector metric in synthetic data.



(a) Distribution of the distance between maximal curvature directions from a mesh and a binary liver.

(b) Cumulated histogram of the distance in a liver

Figure 4.8: Vector metric in binarised real data

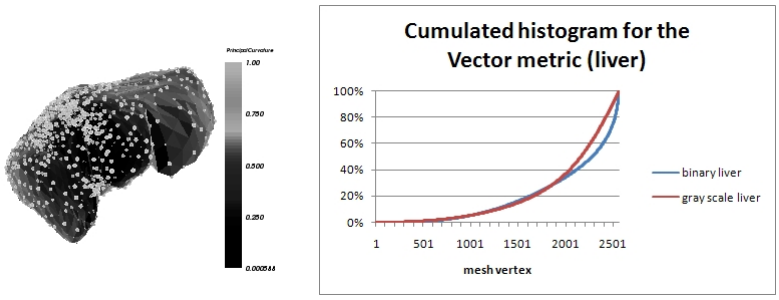
4.2 Curvature in meshes and gray level images

In this section we study the use of curvature values and curvature directions for matching a mesh with an image. The aim of this study is to show feasibility of the use of curvature features in the frame of MBS.

4.2.1 Using curvature directions as a feature

The geometry of the model represents, to some extent, the geometry of the object in the image to be segmented. In this section, we study the relationship between principal curvature directions from the model and from the image. We are particularly interested at the correspondence of curvature features at the ridge-like regions.

If the model represents exactly the object in the image, the vector metric defined in the appendix B is used as for binary images. Figure 4.9(a) shows that the metric has a higher value that in the binary case. Figure 4.9(b) shows the vector distance histogram compared to the vector distance for the binary image.



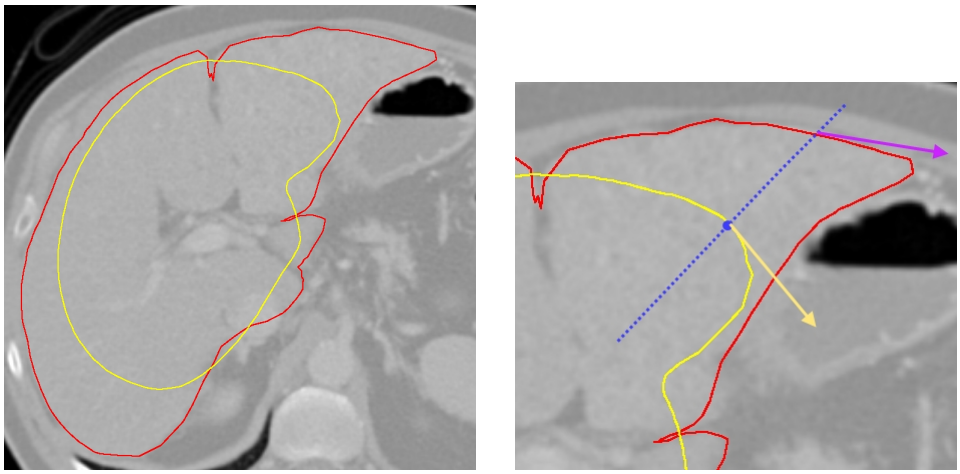
(a) Metric distribution on the mesh surface

(b) Cumulated histogram of the curvature value

Figure 4.9: Vector metric in real data.

If the model does not have exactly the same shape as the object in the image (which is usually the case), vectors do not match. Figure 4.10(a) shows some examples of a model and the image we want to segment. For visualization purposes, we have included the ground truth segmentation.

As depicted in figure 4.10(b), even if the model and the object have some geometrical similarity, curvature directions, specially in ridge-like regions, do not match in general. This results in poor performance when searching for the best matching point along the feature line in the MBS algorithm.



(a) Model (in yellow) of a liver and the corresponding ground truth segmentation (in red).

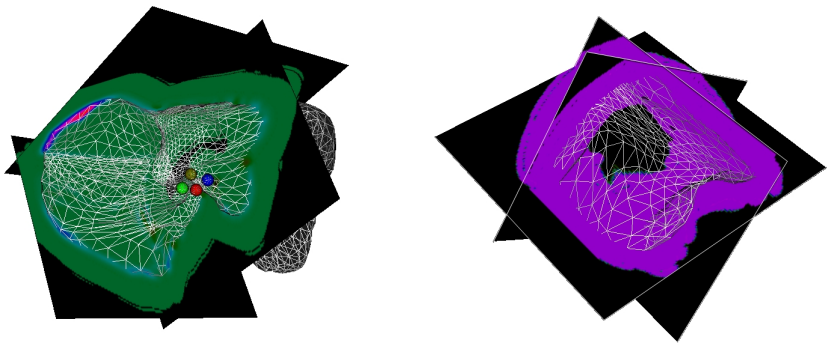
(b) Difference in the orientation of principal directions in the image and in the model.

Figure 4.10: Principal directions in a model and a real image.

4.2.2 Using curvature values as a feature

The direct matching of curvature value is not feasible because the only operator we can use in these images is o_2 (c.f. 3.2.4). This operator is not invariant to intensity value transformations, and therefore the curvature values do not match with the values obtained from the mesh.

However, some authors [MB92, MvdEV96] have used operators o_1 or o_3 in medical images. They found good results in angiography images of the vessels and in MRI[1] crane images. Our case differs from them both in the fact that image gradient is significantly lower at the surfaces (the contrast product in the vessels and the bones in the crane provide a higher contrast). Also, the CT resolution is smaller that MRI's, which imposes severe restrictions in the minimal smoothing required for our application (c.f. section 3.2). Finally, the fact of normalizing the Hessian matrix with the gradient results in a worsening of surface localization (c.f. section 3.2), as we verified in the case of the liver (figures 4.11(a) and 4.11(b)), even for binary images where surface is clearly defined.



(a) Maximal curvature value in a liver binary image, using the operator o_1.

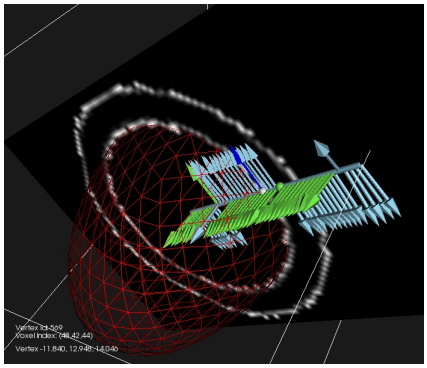
(b) Maximal curvature value in a liver binary image, using the operator o_2.

Figure 4.11: Comparison between curvature value distribution using o_1 and o_2 for curvature computation

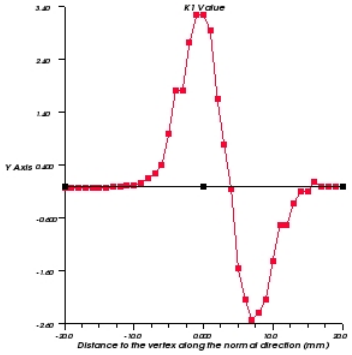
[1]Magnetic Resonance Imaging

4.2.2.1 Zero crossings of the curvature value

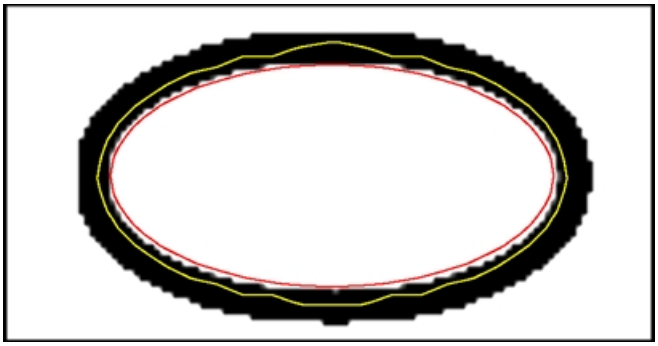
Even if curvature value is not meaningful, due to variations of the gray value along the surface, curvature sign might still be useful. Under the assumption that objects are brighter than the background, a concave surface will have positive curvature and a convex surface will have negative curvature (figure 4.12). This yields a zero crossing of the curvature value at the gap between both surfaces.



(a) Feature search line on a vertex of the ellipsoid



(b) Curvature profile along the the feature line. The zero crossing is located in the middle of the gap.

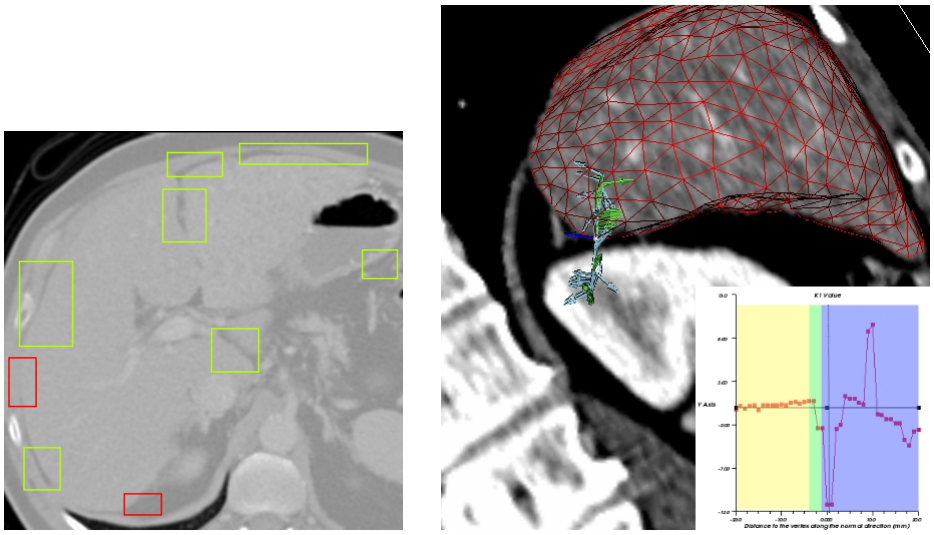


(c) Location of zero crossings (in yellow) of the maximal curvature value in a ellipsoid surrounded by a volume. The ellipsoid surface is highlighted in red.

Figure 4.12: Zero crossings of the curvature value in synthetic data

However, zero crossing location is more complex when working with real

data. The assumption of the curvature sign being equal in images and in meshes is valid only if, in the image, objects are darker than the surrounding material (otherwise, curvature sign changes). In medical images, this is mostly true for non deformable structures such as bones. In the case of soft tissues like liver, lungs, spleen, etc., this may not be true at every point of the surface. However, experimental results have shown concavity in the gray level profile in most cases (figure 4.13).



(a) Regions in a CT image where the gap between two objects is darker than the objects themselves (green). In red, regions where at this resolution there is no gap.

(b) Curvature profile in a border, where we are able to distinguish the zero crossing. In the curvature profile, liver is highlighted in yellow; the gap, in green; and the intestine in blue. Spurious zero crossings appear inside the intestine due to variations in the gray value.

Figure 4.13: Zero crossings of the curvature value in real data.

Focusing only in points where object is darker than the surrounding material, zero crossing of the curvature value are at the gap between two objects (figure 4.13(b)). If the object of interest is convex, the zero crossing goes from positive to negative and vice versa.

It is expected that the model will be convex in regions where the object of interest is also convex and vice versa. This would allow the detection of the right zero crossings of the curvature value from the image. As shown

in figure 4.14, this is true only in some cases, so this criteria is not robust enough.

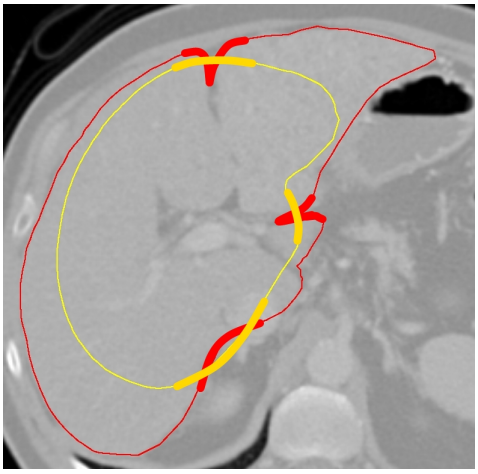


Figure 4.14: Highlight on the concavities and convexities of the liver where model and image do not match, at an arbitrary axial slice. The model is represented by the yellow contour, and the image ground truth segmentation is represented by the red contour.

Experiments have shown poor performance when detecting zero crossings of curvature, i.e. a visual inspection revealed that the location of zero crossings is often wrong. This is due to several reasons:

1. If the feature search line is very long, several structures are intersected and thus several zero crossings are found. Then an additional criteria is needed to solve the ambiguity.

2. Variation of gray value inside a single structure generates spurious zero crossings (figure 4.13(b)).

3. When two structures are very close, the gap between them may be too narrow with respect to the smoothing factor and the image resolution. Then, some zero crossings may be missed.

4. When two contiguous structures are relatively far away (i.e. the feature search line does not achieve the second structure), there will not be any zero crossing at all.

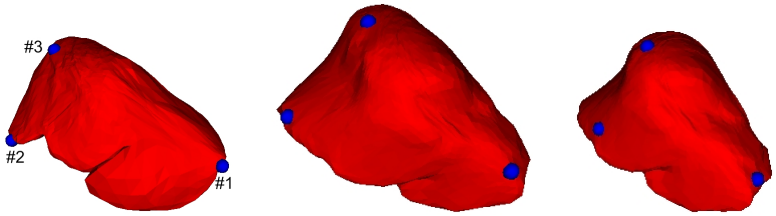
Additional criteria have been tested to help the decision process when trying to detect the right zero crossings[2] but none of them seemed to give robust results.

4.2.3 Exploiting curvature value profile

As shown in previous sections, there is no matching in curvature value and curvature directions computed for the mesh and the corresponding features computed from the image. Nevertheless, if these features match within different images of the same structure (i.e. the same organ in different patients), it would be possible to incorporate this knowledge to the model through a learning process.

In this section curvature value profile at corresponding points from different images is investigate. Each image in the training set has a ground truth segmentation of the liver, in the form of a triangulated mesh.

Profile repeatability was studied at three points of three different livers (figure 4.15). The profile line is centered at each point and oriented along the normal direction. Points #1 and #2 are located at ridge-like regions. Point #3 is located on a flat region of the surface (figure 4.15). The correspondence between each point of the three livers was done manually.



(a) Liver #1. (b) Liver #2. (c) Liver #3.

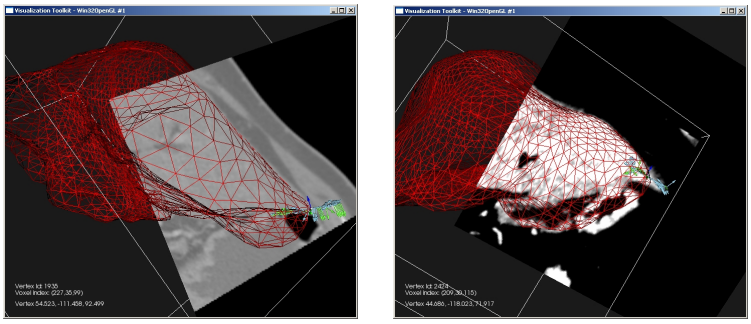
Figure 4.15: Oblique slice at a corresponding point #1 for three different liver images.

Point #1

This point is located at the sharpest region of the surface. Figure 4.16 shows the selected vertex (Point#1) of the mesh and an oblique slice with the profile

[2]These include: selecting the closest ZC, the ZC with highest gradient, the ZC where curvature vectors better match, etc.

line for each case.



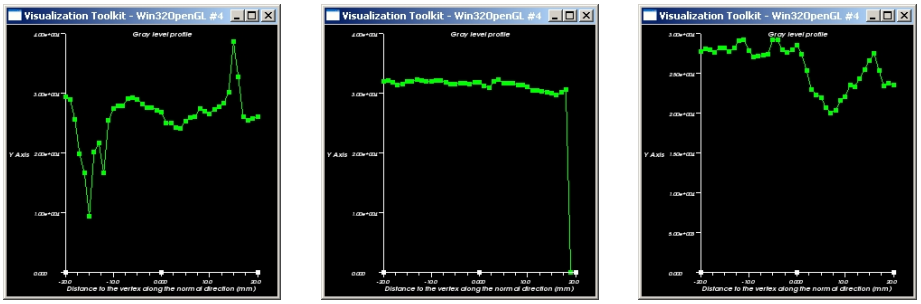
(a) Liver #1.

(b) Liver #2.

(c) Liver #3.

Figure 4.16: Oblique slice at a corresponding point #1 for three different liver images.

Regarding the gray level profile (figure 4.17), the correlation between the three livers is not very clear; in this case, gray level profile is not a useful feature.

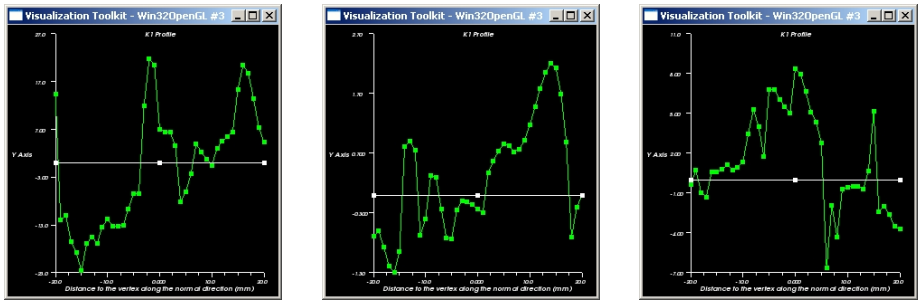


(a) Liver #1. (b) Liver #2. (c) Liver #3.

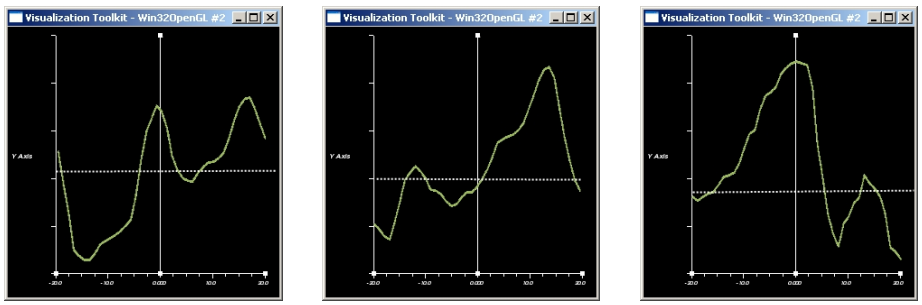
Figure 4.17: Gray level profile centered at point #1 along the normal direction, for three different liver images.

Comparing the three maximal curvature value profiles (figure 4.18), in the three cases curvature is high and positive at the border of the liver, and becomes negative outside of it. An smoothed version of the profiles is depicted in figure 4.18(d) to 4.18(f) to illustrate the visual correlation. In this case, matching is not evident because of the gray value variations in the inside of the liver.

In figure 4.16(b) the ground truth does not reach the real border of the liver, which explains why figure 4.18(b) is displaced with respect to figures 4.18(a) and 4.18(c). This means that curvature profile may help to improve segmentation at ridge-like regions.



(a) Liver #1. (b) Liver #2. (c) Liver #3.

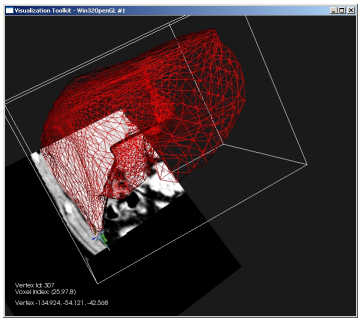


(d) Ideal curvature profile (e) Ideal curvature profile (f) Ideal curvature profile

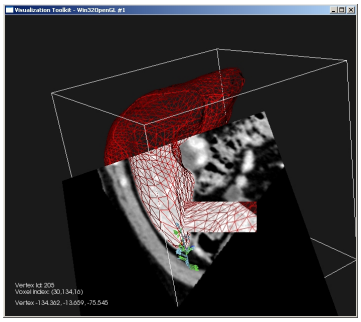
Figure 4.18: Maximal curvature profile centered at point #1 along the normal direction, for three different liver images.

Point #2

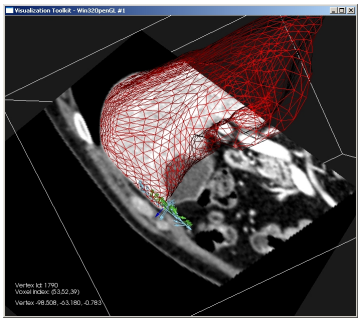
Figure 4.19 shows the location of the point #2 and the profile line for each liver. Figure 4.20 contains the gray level profile and figure 4.21 shows the maximal curvature profile.



(a) Liver #1.

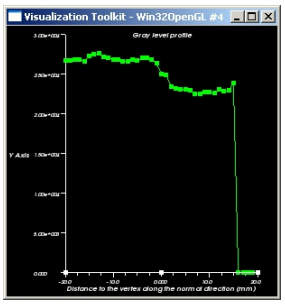


(b) Liver #2.

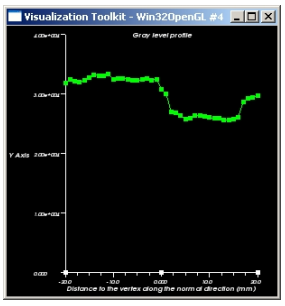


(c) Liver #3.

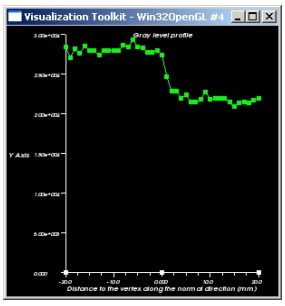
Figure 4.19: Oblique slice at a corresponding point #2 for three different liver images.



(a) Liver #1.

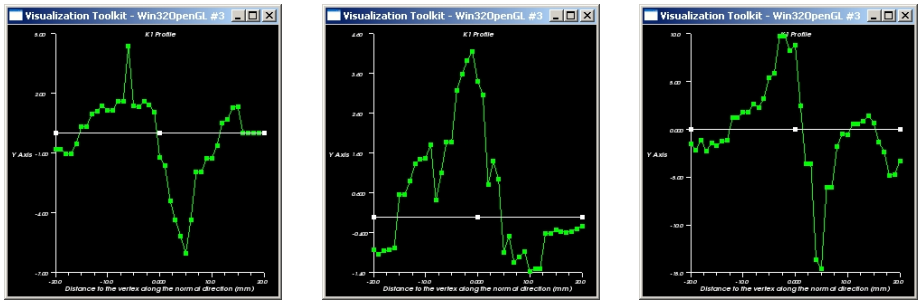


(b) Liver #2.

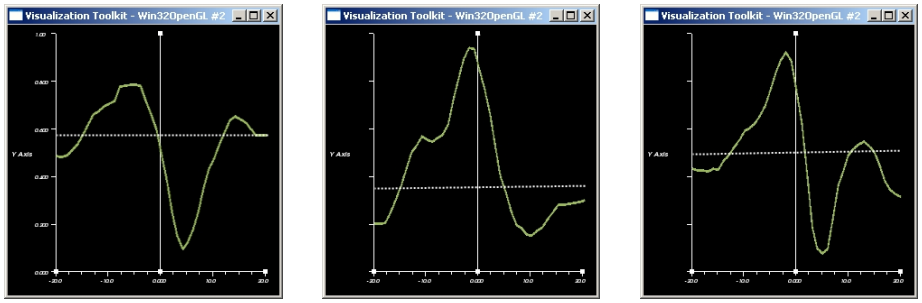


(c) Liver #3.

Figure 4.20: Gray level profile centered at point #2 along the normal direction, for three different liver images.



(a) Liver #1. (b) Liver #2. (c) Liver #3.



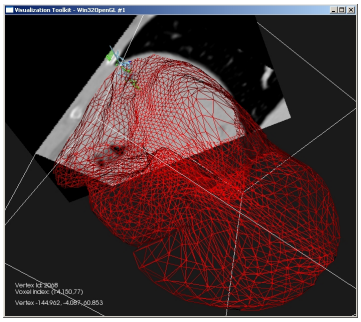
(d) Ideal curvature profile (e) Ideal curvature profile (f) Ideal curvature profile

Figure 4.21: Maximal curvature profile centered at point #2 along the normal direction, for three different liver images.

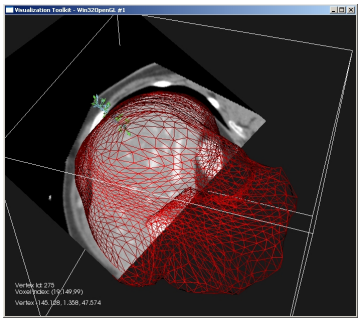
In this case, gray level profile matches between the three livers. Experiments show that when this happens, curvature value profile also matches, as depicted in figure 4.21.

Point #3

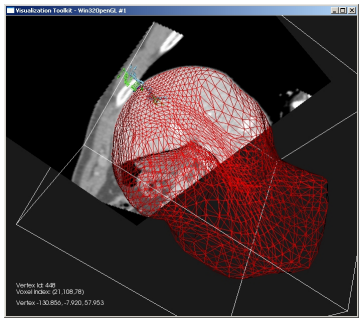
The last case evaluates a region of very low curvature (figure 4.22). Gray level profiles show high similarity (figure 4.23). The curvature value (figure 4.24) is very small and positive at the surface, followed by a high negative curvature from the surrounding object (the ribcage), and a very high curvature (the rib).



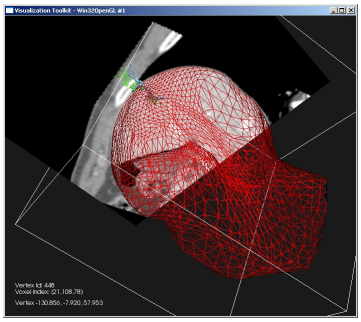
(a) Liver #1.



(b) Liver #2.

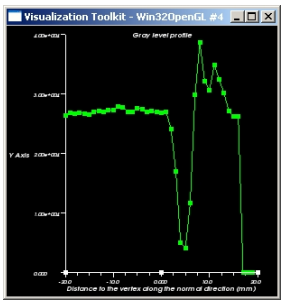


(c) Liver #3.

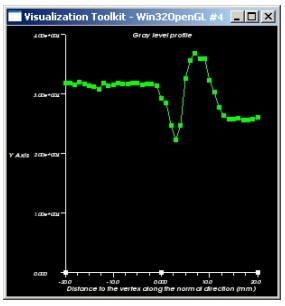


(d) Liver #3.

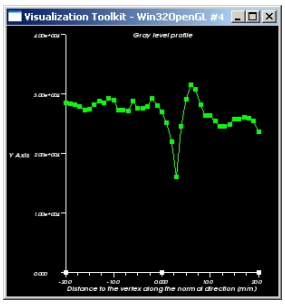
Figure 4.22: Oblique slice at a corresponding point #3 for three different liver images.



(a) Liver #1.

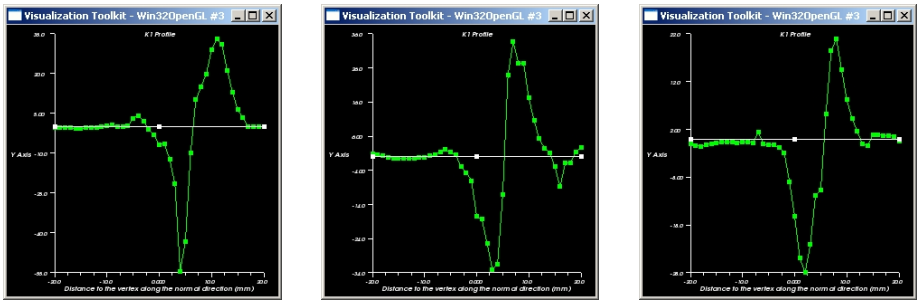


(b) Liver #2.



(c) Liver #3.

Figure 4.23: Gray level profile centered at point #3 along the normal direction, for three different liver images.



(a) Liver #1. (b) Liver #2. (c) Liver #3.

Figure 4.24: Maximal curvature profile centered at point #3 along the normal direction, for three different liver images.

4.2.4 Exploiting curvature vector profile

Vectors can be also computed along the feature profile. This produces a 3D profile, which is difficult to visualize and analyze. A component-by-component analyze does not give any usable correlation.

Another way to obtain a 1D profile function from vectors is to represent the distance (defined as in Appendix B) between the curvature direction from the mesh and the curvature direction computed from the image (at each point in the profile).

These profiles do not show any exploitable correlation neither. The full data for vectorial profile, for the three points defined in section 4.2.3 can be checked in appendix E.

Chapter 5

Discussion and further work

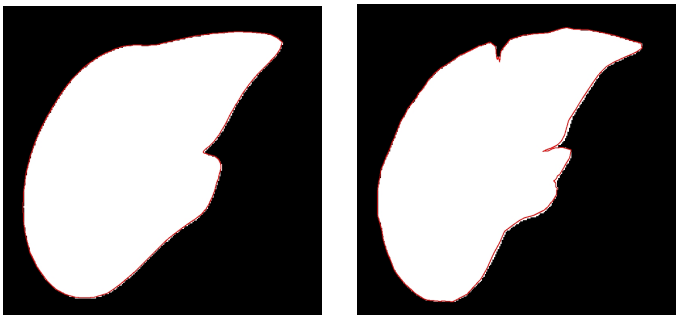
This chapter explains how curvature features may be used for matching a triangulated mesh to an image in the framework of MBS. Four scenarios are considered:

1. Matching a mesh and an image that represent the same surface.
2. Matching a mesh and an image when the mesh approximates the surface represented in the image.
3. Establishing correspondence between different meshes that represent the same object.
4. Initializing the model by registration.

Also, an overview on the future work can be found at the end of this chapter.

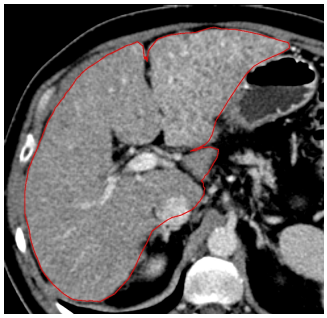
5.1 Mesh and Image represent the same surface

When a triangulated mesh and an image represent the same object, there is a correlation between principal curvatures and principal directions of the mesh and the image. The ground truth segmentation of a liver image (figure 5.1(c)) has been binarized (figure 5.1(b)) and then smoothed (figure 5.1(a)). Accuracy in the matching of curvature value and direction was compared among the three cases. (c.f. table 5.1).



(a) Smoothed binary liver and mesh

(b) Non smoothed binary liver and mesh



(c) gray-level liver and ground truth segmentation

Figure 5.1: Slice of three different 3D liver images, with the mesh highlighted in red.

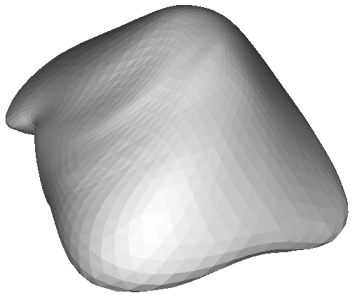
		binary liver		gray scale
		smth	no smth	liver
k value	Mean	46.50%	75.02%	201.17%
	Std. Dv.	39.32%	46.32%	156.45%
k direction	Mean	9.07°	27.24°	45.10°
	Std. Dv.	11.35°	28.84°	33.04°

Table 5.1: Error of curvature matching. Curvature computed from the mesh (values and directions) is used as reference. For simplicity of the interpretation, vector metric has been translated to angle deviation in degrees. Curvature value is expressed in relative error, $\frac{\|k_{image}-k_{mesh}\|}{\|k_{mesh}\|}$.

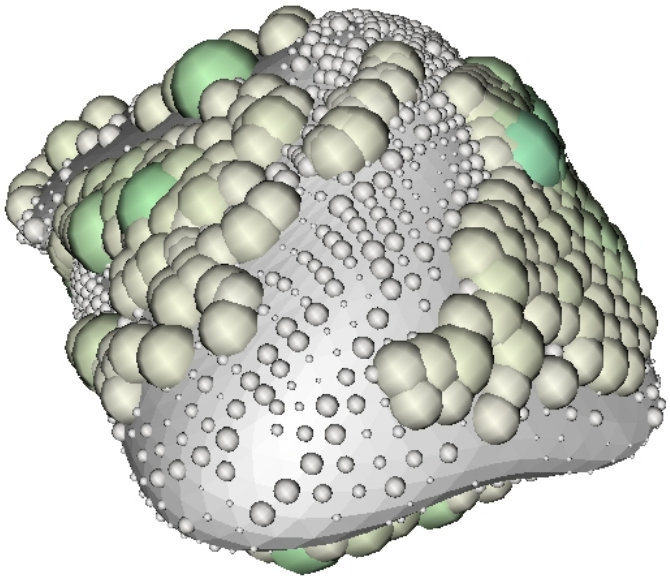
In general, vectors are a more robust feature. Curvature values showed

correlation only in some regions of the object. There are two kind of areas where error is particularly high:

- In flat areas the curvature value is low, which results in a high relative error even if the absolute error is very small (figure 5.2).



(a) Original liver.

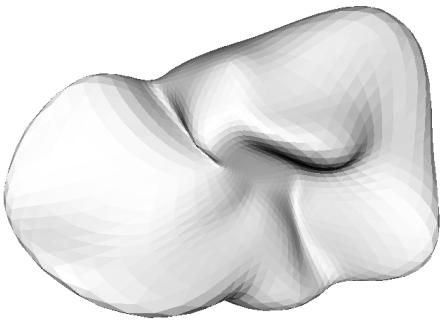


(b) Original liver with the curvature value error highlighted with spheres. The size of the spheres is proportional to the error.

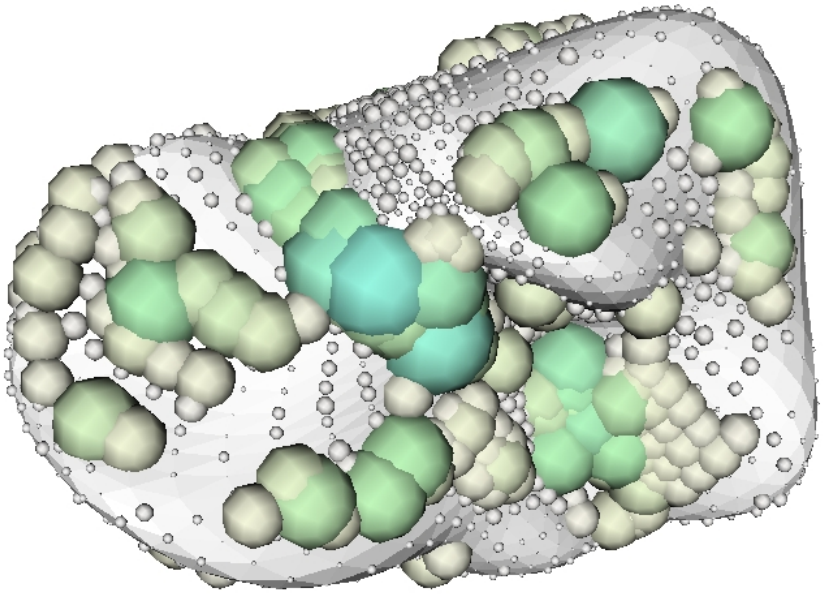
Figure 5.2: Regions where curvature value error is higher in the case of a binary image and a smooth mesh (Front view of the liver).

- Due to smoothing of the image, regions which are concave and narrow

may collapse (figure 5.3).



(a) Original liver.



(b) Original liver with the curvature value error highlighted with spheres. The size of the spheres is proportional to the error.

Figure 5.3: Regions where curvature value error is higher in the case of a binary image and a smooth mesh (back view of the liver).

The closest-neighbor interpolation[1] is used to compute the curvature in the image at the position of a vertex of the mesh. Closest neighbor is a

[1]The curvature value and direction for every point in the search profile and at the position of the mesh vertices has been computed using a closest neighbor interpolation.

simple but inaccurate interpolation method that may introduce a significant error. To compensate for this error, table 5.2 shows the results when the best matching value is selected from a neighborhood of the vertex of $3 \times 3 \times 3$ voxels.

		binary liver		gray scale
		smth	no smth	liver
k value	Mean	28.23%	44.15%	73.32%
	Std. Dv.	19.74%	32.62%	64.95%
k direction	Mean	3.77°	15.00°	22.95°
	Std. Dv.	5.20°	16.76°	22.24°

Table 5.2: Error of curvature matching. To compensate for the closest neighbor interpolation error, we take the best point in a neighborhood of $3 \times 3 \times 3$ voxels.

For a smooth mesh, and an image representing an object of the same shape, curvature is proven to have a high correlation between meshes and images, specially in convex ridge-like regions. Improving the interpolation method may significantly improve the error rate.

5.2 The mesh approximates the surface represented in the image

If the mesh and the image do not represent the same object, but the mesh is a model that approximates the shape of the expected object in the image, direct matching of curvature values or directions is in general not feasible because of the difference in the geometry.

If the distribution of curvature features is repeatable among different images in a data set, it is possible to build an statistical model and merge the value with the geometrical model. In other words, the model can learn the feature distribution from a training set, and then try to find this feature during the segmentation. In particular, we are interested at curvature features at ridge-like regions. Experiments point that curvature value distribution may

For a given point p of the mesh, in world coordinates, the voxel v whose center is closer to p is chosen. This can introduce a significant error in the estimation that depends on the shape and the image.

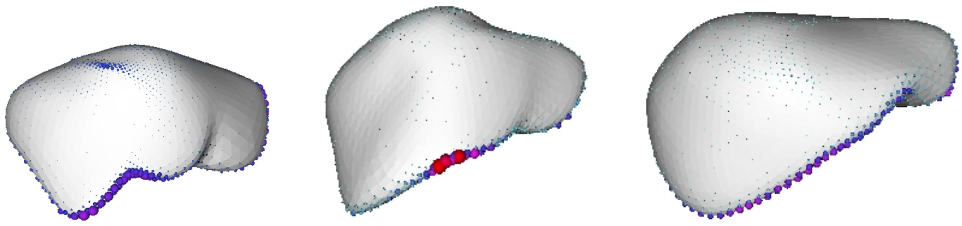
be more efficient that other features, like gray level profile, at these regions, as showed in section 4.2.

Furthermore, ridge-like regions can be very reliably determined in meshes, enabling the labeling of the models prior to the training stage. The adaptation algorithm could use specific features at the labeled points to improve the segmentation. However, this results must be further investigated.

5.3 Establishing correspondence

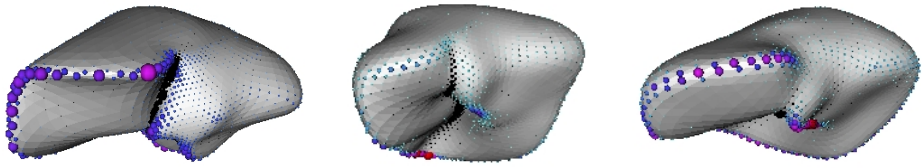
An important challenge in the model building stage of MBS (and in image processing in general) is establishing correspondence between shapes [Dav02]. Correspondence is needed in particular for building an statistical model from a training set [CHTH94]. The training set consists on a set of images and their associated ground truth segmentation.

Experiments show that ridge lines are located at corresponding regions for different shapes. Figures 5.4 and 5.5 show how three different livers present ridge lines at corresponding regions. The ridge lines where obtained by simple thresholding of the maximal curvature value.



(a) Liver #1 (b) Liver #2 (c) Liver #3

Figure 5.4: Correspondence between ridge lines in three liver models



(a) Liver #1 (b) Liver #2 (c) Liver #3

Figure 5.5: Correspondence between ridge lines in three liver models (II)

5.3.1 Initialization of the model by registration

The matching of curvature values presented in section 4.1 suggests the feasibility for rigid alignment (rotation and translation) of the mesh with the binary image. This alignment consists on the minimization of the following expression:

$$\min_{T} \sum_{i} \|k^{M}_{p_i} - o_1(T(I(x,y,z)))|_{p_i}\|^2 \tag{5.1}$$

Where T is the rigid transformation we are looking for; $k^{M}_{p_i}$ is the curvature of the vertex p_i of the mesh; and $o_1(T(I(x,y,z)))|_{p_i}$ is the operator o_1 applied to the transformed image and evaluated at point p_i.

As curvature value is $k = \frac{1}{radius}$, we can also allow scaling in the registration introducing a $\frac{1}{r}$ factor in the expression 5.1:

$$\min_{T,r} \sum_{i} \|k^{M}_{p_i} - \frac{1}{r} \times o_1(T(I(x,y,z)))|_{p_i}\|^2 \tag{5.2}$$

This can be also used for model initialization on a gray scale image, although accuracy of the method has to be determined.

Initialization is an important task in the MBS framework where the model has to be positioned close to the object of interest. Otherwise, the adaptation will be deviated and not captured by the object.

5.4 Further work

The use of curvature features in the MBS framework may involve:

- Establishing correspondence of the different meshes from the training set.
- Labeling the ridge-like regions of the model.
- Determination of the initial model position by registration.
- Use of curvature features during the adaptation process.

The integration of curvature features into the MBS was out of the scope of this work. The use of curvature shows large potential to improve MBS in particular at ridge-like regions where MBS currently often fails.

Appendix A

Elements of Differential Geometry

This Appendix introduces some basic concepts in Differential Geometry. For a more extensive reference, see for example [dC76].

Differential geometry is a branch of mathematics that uses the tools of calculus and linear algebra to study geometry. In this document, we are interested at the geometry of surfaces.

A.1 Curves in the space

Let $\alpha(t)$ be the parametrization of a curve. We would like to express the curve at any point in terms of a basis of $\Re^3$. At each point, we can build the orthonormal vectors $\vec{T}(t)$, $\vec{N}(t)$ and $\vec{B}(t)$. $\vec{T}$ is the tangent to the curve. $\vec{N}$ is the normal to the curve. $\vec{B}$, called the binormal, is perpendicular to the other two.

We will see that the derivatives of these vectors can be expressed as linear functions of the curvature, k, and the torsion, τ.

Let $\alpha(t)$ be the parametrization of a curve of unit speed. In that case, the tangent vector at a point t_0 is $\vec{T} = \frac{\partial \alpha}{\partial t}$ and $\left\|\vec{T}\right\| = 1$. Then we define the curvature of $\alpha(t)$ as:

$$k(t) = \left\|\vec{T}'(t)\right\| \tag{A.1}$$

The unit normal vector is defined as:

$$\vec{N}(t) = \frac{\vec{T}'(t)}{k(t)} \tag{A.2}$$

And the binormal:

$$\vec{B}(t) = \vec{T}(t) \times \vec{N}(t) \tag{A.3}$$

And we can define the torsion, which measures the extent to which the curve is twisting out of the plane in which it lies, as:

$$\tau(s) = -\left\langle \vec{B}'(t), \vec{N}(t) \right\rangle \tag{A.4}$$

Together, the quantities $\vec{T}, \vec{N}, \vec{B}, k, \tau$ are collectively called the Frenet-Serret Apparatus. The relationship between them is expressed in the Frenet-Serret Theorem:

$$\begin{array}{lccc} \vec{T}' = & & k(t)\vec{N}(t) & \\ \vec{N}' = & -k(t)\vec{T}(t) & & +\tau(t)\vec{B}(t) \\ \vec{B}' = & & -\tau(t)\vec{N}(t) & \end{array} \tag{A.5}$$

These expressions allow the computation of the curvature and the torsion at any point. A more practical way to compute curvature is to use the Fundamental Forms, as described in the following section.

A.2 Surfaces in the space

The curvature of a surface may be studied through the variation of the vector normal to that surface at each point. For this, we define the Fundamental Forms:

1. The First Fundamental Form contains the intrinsic properties of a surface, and it is defined by the inner product on the tangent space of a surface in three-dimensional space. Let $\mathbf{t}_s^1$ and $\mathbf{t}_s^2$ be two tangent vectors at the point $s \in S$:

$$\mathbf{I}(\mathbf{t}_s^1, \mathbf{t}_s^2) = < \mathbf{t}_s^1, \mathbf{t}_s^2 > \tag{A.6}$$

And if we use this operator with $d\mathbf{p} = (\mathbf{p}_u, \mathbf{p}_v)$:

$$\mathbf{I}(d\mathbf{p}, d\mathbf{p}) = d\mathbf{p}^2 = E\mathbf{p}_u^2 + 2F\mathbf{p}_u\mathbf{p}_v + G\mathbf{p}_v^2 \tag{A.7}$$

This can be written in matrix form, for any two tangent vectors as:

$$\mathbf{I}(\mathbf{t}_s^1, \mathbf{t}_s^2) = (\mathbf{t}_s^1)^T \begin{bmatrix} E & F \\ F & G \end{bmatrix} \mathbf{t}_s^2 \tag{A.8}$$

2. The Second Fundamental Form is a quadratic form on the tangent plane of a surface in the three dimensional space, and it is defined by:

$$\mathbf{II}(\mathbf{t}_s^1, \mathbf{t}_s^2) = - < \nabla_{\mathbf{t}_s^1} \mathbf{N}, \mathbf{t}_s^2 > = < \mathbf{N}, \nabla_{\mathbf{t}_s^1} \mathbf{t}_s^2 > \tag{A.9}$$

Where $\nabla_{\mathbf{t}_s^1} \mathbf{N}$ is the derivative of the normal vector in the direction of $\mathbf{t}_s^1$. Again, if we use this operator with $d\mathbf{p} = (\mathbf{p}_u, \mathbf{p}_v)$:

$$\mathbf{II}(d\mathbf{p}, d\mathbf{p}) = e p_u^2 + 2 f p_u p_v + g p_v^2 \tag{A.10}$$

This can be written in matrix form, for any two tangent vectors as:

$$\mathbf{II}(\mathbf{t}_s^1, \mathbf{t}_s^2) = (\mathbf{t}_s^1)^T \begin{bmatrix} e & f \\ f & g \end{bmatrix} \mathbf{t}_s^2 \tag{A.11}$$

A.3 Curvatures

In a point of a surface, the curvature is defined with it corresponding direction; as there are infinite possible directions in the tangent plane, there are infinite curvature values at a single point. Then, we define the following curvatures:

- Principal curvatures, k_1 and k_2, and the associated principal directions $\vec{k}_1$ and $\vec{k}_2$. These are the maximal and minimal curvatures and their directions.

- Mean curvature, H, is the average of all the curvatures at one point. $H = \frac{k_1 + k_2}{2}$.

- Gaussian curvature, K, defined as $K = k_1 k_2$.

For more details in how to compute these curvatures from the fundamental forms, see next section.

A.4 Weingarten matrix

The curvature of a surface may be studied through the variation of the vector normal to that surface at each point. The derivative of the normal vector in the direction of the vectors tangent to the surface at each point is called the Weingarten Map. Let $\mathbf{p}(u,v) : \Re^2 \rightarrow \Re^3$ be a parametrization of the surface S. For computing this Weingarten Map, we use the First and Second Fundamental Forms.

Then, the normal curvature in a tangent direction $\vec{\tau}$ (i.e. the curvature of the section of the surface by a plane containing the normal vector and the vector $\vec{\tau}$) is:

$$k_n(\vec{\tau}) = \mathbf{II}(\vec{\tau},\vec{\tau}) \tag{A.12}$$

And in the case when $\vec{\tau}$ is not a unit vector (general case),

$$k_n(\vec{\tau}) = \frac{\mathbf{II}(\vec{\tau},\vec{\tau})}{\mathbf{I}(\vec{\tau},\vec{\tau})} \tag{A.13}$$

Which, in matrix form, and for the tangent vector $d\mathbf{p}$, yields:

$$\begin{aligned} k_n(\vec{\tau}) &= \begin{bmatrix} \mathbf{p}_u \\ \mathbf{p}_v \end{bmatrix}^T \begin{bmatrix} E & F \\ F & G \end{bmatrix}^{-1} \begin{bmatrix} e & f \\ f & g \end{bmatrix} \begin{bmatrix} \mathbf{p}_u \\ \mathbf{p}_v \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{p}_u \\ \mathbf{p}_v \end{bmatrix}^T \frac{1}{EG-F^2} \begin{bmatrix} eG-fF & fG-gF \\ fE-eF & gE-fF \end{bmatrix} \begin{bmatrix} \mathbf{p}_u \\ \mathbf{p}_v \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{p}_u \\ \mathbf{p}_v \end{bmatrix}^T W \begin{bmatrix} \mathbf{p}_u \\ \mathbf{p}_v \end{bmatrix} \end{aligned} \tag{A.14}$$

Where W is the Weingarten matrix. The eigenvalues of this matrix are the principal curvatures, and the associated eigenvectors are the principal directions.

Appendix B

Distance metric when using vectors

We describe in this section a metric establishing a distance definition between two vectors. The distance is defined in such a way that two vectors are more similar the smaller the distance between them is.

A vector is characterized by a direction and an orientation. We are not interested at the length of the vectors so we assue that vectors are normalized.

Distance $d(a, b)$ is a measure of how close or how far two elements a and b, $a, b \in G$ are according to some metric definition. In this work, $a, b \in \ell(\vec{i}, \vec{j}, veck)$, where $\ell(b_i)$ is the vector space generated by the basis b_i, and $\vec{i}, \vec{j}, \vec{k}$ is the canonical base of $\mathbb{R}^3$.

For mathematical consistency, the distance definition has to assure that:

1. $d(x, y) \geqslant 0$ (non-negativity).
2. $d(x, y) = 0$ if and only if $x = y$ (identity of indiscernibles).
3. $d(x, y) = d(y, x)$ (symmetry).
4. $d(x, z) \leqslant d(x, y) + d(y, z)$ (triangle inequality).

Then we propose two definitions for distance between normal vectors:

$$d_1(\vec{a}, \vec{b}) = \frac{1 - \langle \vec{a}, \vec{b} \rangle}{2} = \frac{1 - \cos(\theta_{\vec{a},\vec{b}})}{2} \tag{B.1}$$

$$d_2(\vec{a}, \vec{b}) = 1 - \|\langle \vec{a}, \vec{b} \rangle\|^2 = 1 - \cos^2(\theta_{\vec{a},\vec{b}}) = \sin^2(\theta_{\vec{a},\vec{b}}) \tag{B.2}$$

Where $\theta_{\vec{a},\vec{b}}$ is the angle between both vectors. Note tht we alway consider the smallest angle between both. Both definitions verify the 4 conditions.

First definition d_1 takes into account direction and orientation, while d_2 ignores orientation given the direction. In practice, we usethis metric to compute distance between a principal curvature direction computed from the image and a principal curvature direction computed from the image. In that case, we want to ignore orientation (which may depend on the method used to compute eigenvectors of the Weingarten matrix[1]). Therefore, we use the second definition.

A different metric, which does not verify the conditions of distance, is used in [ZGX05] for registration based in vectors information.

[1]For example, the eigenvectors may be computed: in a several step process which begins with a Hessenberg decomposition, followed by a Schur decomposition, as done by GNU Octave (www.octave.org); or using the EISPACK routine RS, which in turn calls TRED2 to reduce A to tridiagonal form, followed by TQL2, to find the eigensystem decomposition of the matrix, as done with the VNL library for C++ (vnl: Numerics Library) following the method from Golub and van Loan [GvL96]. In some practical cases we found that with both methods eigen values were identical but eigenvectors were in the same direction but opposite orientation.

Appendix C

Computation of local coordinate system

We need to compute a local coordinate system where one axis is aligned with the gradient (vector orthogonal to the surface) and the other two define the tangent plan at a voxel. Given the normal vector $\overline{g} = (g_x, g_y, g_z)$, then one normal vector $\overline{v}$ in the tangent plane $\pi \equiv g_x x + g_y y + g_z z = 0$ verifies:

$$\begin{aligned} g_x v_x + g_y v_y &= 0 \\ v_x^2 + v_y^2 &= 1 \end{aligned} \tag{C.1}$$

Where we did $v_z = 0$ because we have one degree of freedom. That yields:

$$\begin{aligned} v_x &= \frac{g_y}{\sqrt{g_x^2 + g_y^2}} \\ v_y &= \frac{-g_x}{\sqrt{g_x^2 + g_y^2}} \end{aligned} \tag{C.2}$$

If g_x and g_y are simultaneously zero (i.e. the gradient is $(0, 0, g_z)$), then a vector in the tangent plan is simply any normal vector where $v_z = 0$, for example $\overline{v} = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0)$.

Finally, the third vector of our basis, $\vec{w}$, is a vector orthogonal to $\vec{g}$ and $\vec{v}$, this is, $\vec{w} = \vec{g} \times \vec{v}$.

Appendix D

Visualization tools

Despite the several visualization applications available at the beginning of this work, some visualization functionalities were still missing. Along this proyect, several visualization tools have been developped as they were needed.

These visualization tools have been developped in C++, making use of the Insight Toolkit (ITK[1]), the Visualization Toolkit (VTK[2]) and the Prima library from Philips Research.

The code developped is built as a C++ library, and the aim is to make it usable from other modules in a way as easy way as possible. In the following sections, we describe the different modules available in the library.

The imput data format is *.vtk for meshes and *.ics (Philips private format) for 3D images.

D.1 SimpleVisualizer

SimpleVisualizer provides basic visualization capabilities for displaying triangular meshes and 3D images. It consists in a set of methods that can add objects to a visualization window. The objects that can be added are:

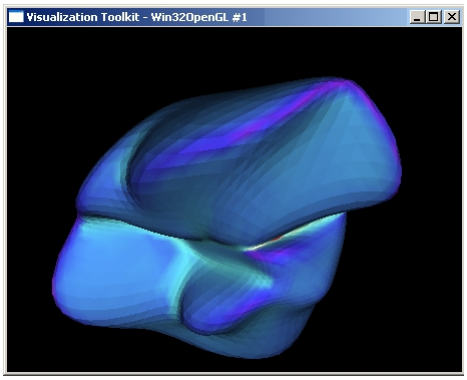
- Meshes. In VTK file or as a Model object (Philips specific data type). A scalar field may be specified for vertex coloring (figure D.1(a)).
- Volumes. In BVolume format (Philips specific data type). Minimal and maximal intensity values may be specified for only displaying a range of

[1] www.itk.org
[2] www.vtk.org

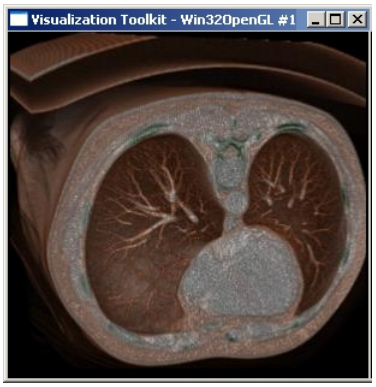
intensity. The rendering technique used is a ray cast rendering (figure D.1(a)).

- Vectors. A set of vectors and the points where they are placed required. Instead of a list of points, the user can also specify a mesh ant its vertex are taken as the points where vectors are located (figure D.1(c)).

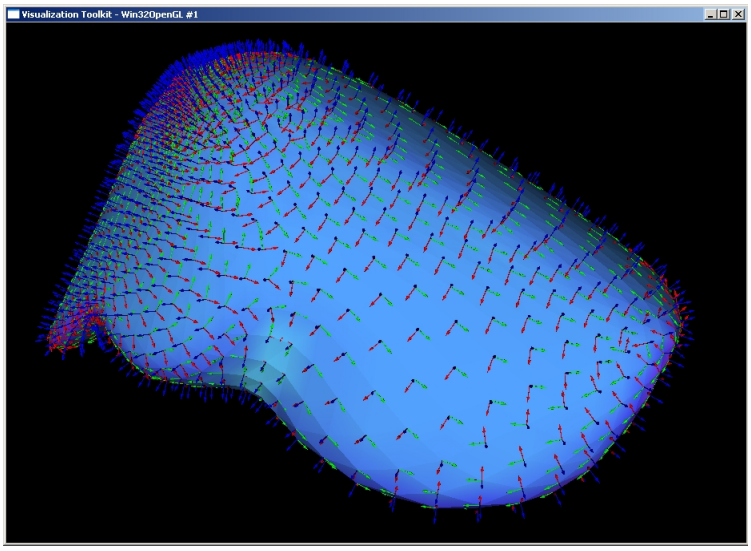
And they can be displayed independently or in the same window.



(a) Mesh plot, colored with curvature values.



(b) Volume plot.



(c) Normal and principal vectors on the mesh surface.

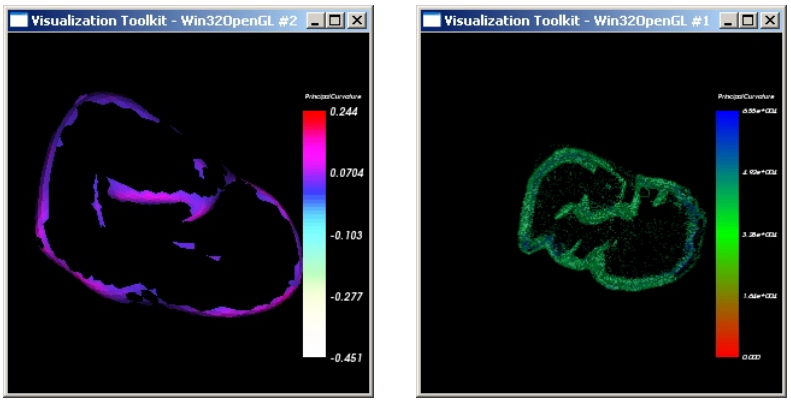
Figure D.1: SimpleVisualizer graphical interface.

D.2 ThresholdVisualizer

ThresholdVisualier consist on two classes (one for meshes and one for volumes) that provide interactive thresholding in the viewer.

For meshes (figure D.2(a)), a scalar field is used to associate a value with each vertex (e.g. curvature value). Then the user can select the range that will be displayed, and change it dinamically. This allows selection of vertex of high value only, low value only, etc. The thresholding is done by moving the mouse over the image. It is also posible to switch between the thresholding mode and the normal mode, where the standard vtk interactor is activated and the mouse events control the rotation, position, zoom, etc..

For images (figure D.2(b)), the scalar field is simply the gray level. The curvature image can then be thresholded in the same way as for meshes, and only the voxels where the gray level is in the threshold range are rendered.



(a) Threshold on a liver mesh.

(b) Threshold on a binary liver image.

Figure D.2: ThresholdVisualizer graphical interface.

D.3 SlicesVisualizer

This module provides methods for interactive visualization of three orthogonal slices of a volume. The user can translate the slices to any location within the volume bounds with the mouse motion over the display. Each slice is chosen by pressing one of the three mouse buttons.

There is also a oblique slice mode, and mouse motion rotates the slice instead of translating it. This provides oblique slices visualization. In the current version of the module, only the axial view can be rotated.

The user can also associate a triangulated mesh to the display and visualize the mesh object together with the slices. This makes easier the understanding of curvature distribution on a certain plane with respect to the object, as depicted in figure D.3.

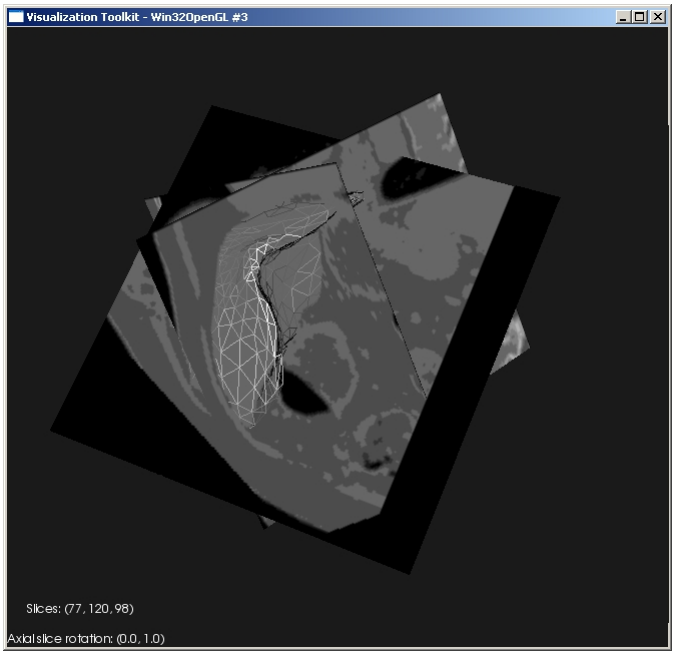


Figure D.3: SlicesVisualizer interface

D.4 FeatureExplore

FeatureExplore (figure D.4) provides methods for exploring a certain image feature along a feature line centered in a vertex of the model that is interactively selected by the user. The interface consist in a main window, where the user interacts, and a set of side windows.

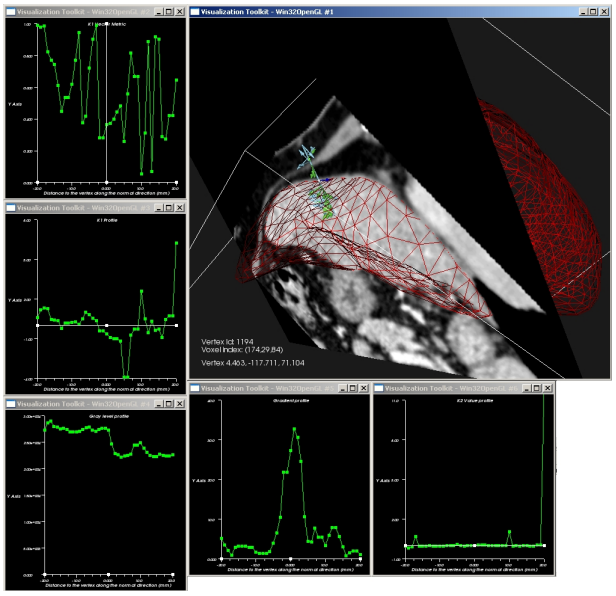


Figure D.4: FeatureExplore graphical interface.

The main display shows a triangulated mesh. The user can select a vertex with the mouse. For the selected vertex, the main display show several objects (figure D.5):

- The feature search line: a line in the direction of the vector normal to the esh at the vertex.

- The feature slice: a slice of the volume along a plane that contains the normal vector and the maximal curvature direction.

- The image curvature directions: represented with vectors, at each evaluation point of the feature search line.

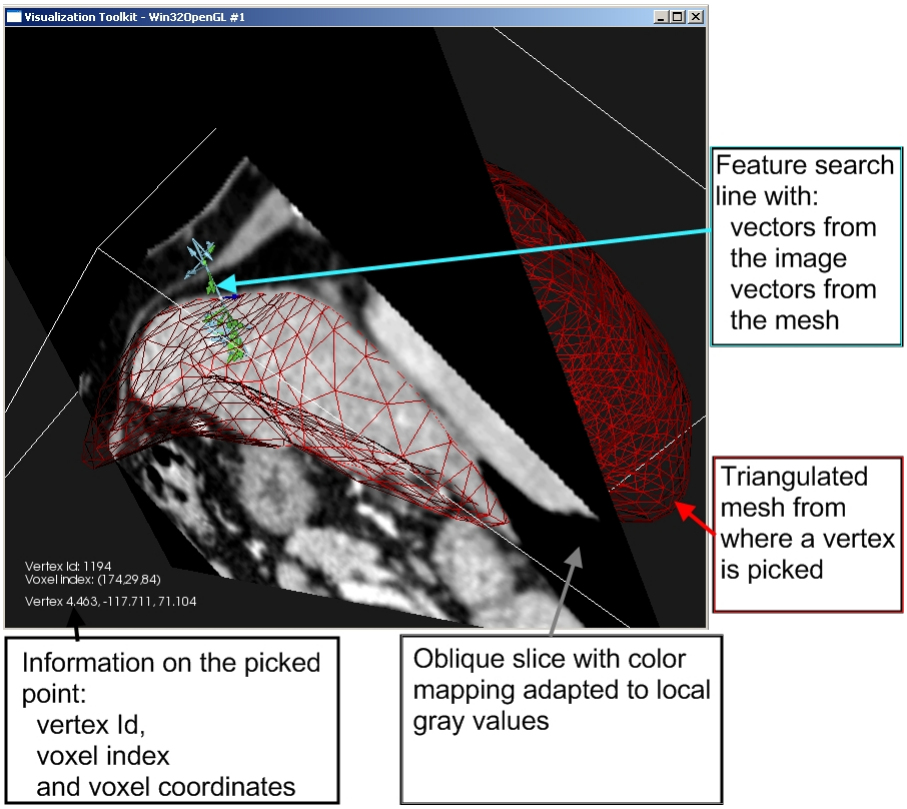


Figure D.5: FeatureExplore main window

The length of the feature search line and the sampling rate of it (i.e. the distance between to points of the line where metric is evaluated) are specified by the user.

The side windows display different metrics evaluated along the current feature search line displayed in the main window. Example of available metrics are curvature value, principal directions distance (as defined in Appendix B), gray level profile, gradient profile, etc.

The side windows present a number of features that can be easily modifies in the code. The default layout is shown in figure D.6.

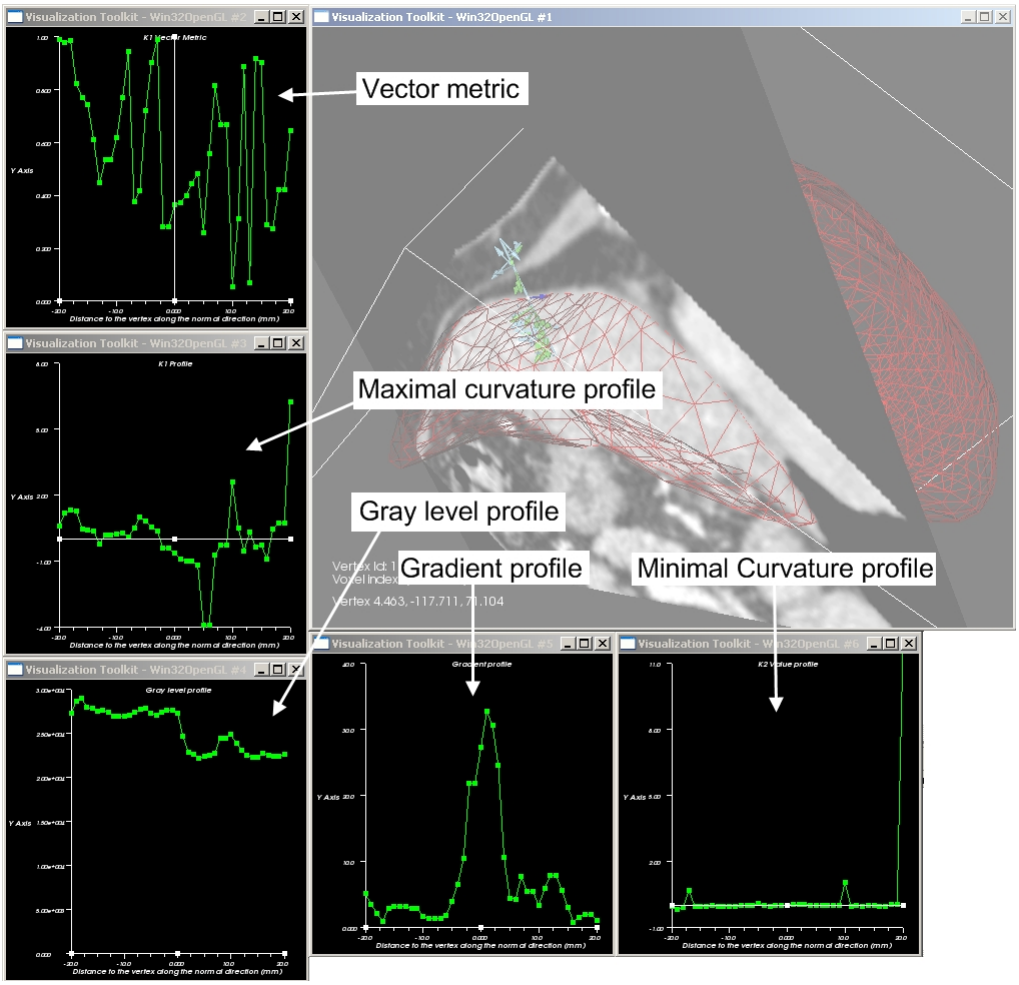


Figure D.6: FeatureExplore side windows

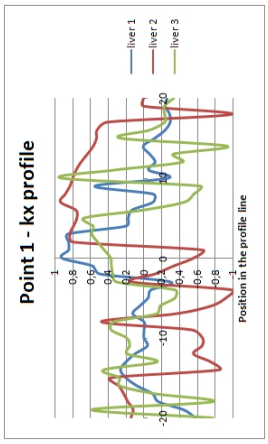
Appendix E

Full profile data experiments

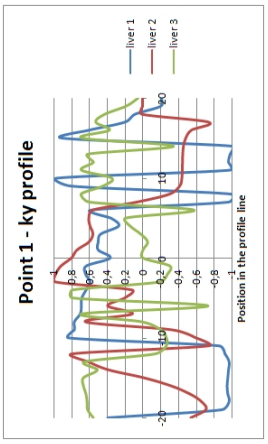
In this Appendix, different features are explored along a profile line. They are evaluated at three corresponding points of three different livers (c.f. section 4.2.3).

E.1 Component-by-component profile

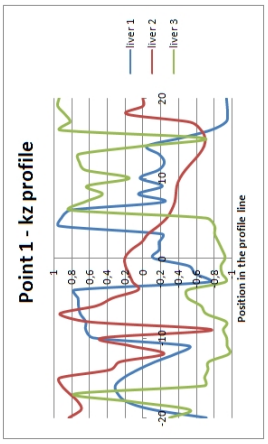
The graphs in this section show the for maximal curvature direction profile, comparing component-by-component the vectors at corresponding points of three different livers.



(a) x component of maximal curvature direction

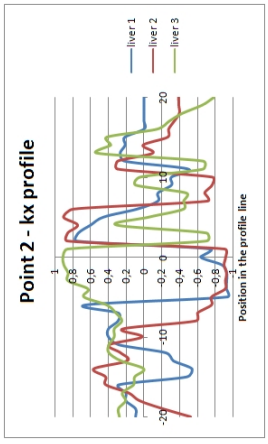


(b) y component of maximal curvature direction

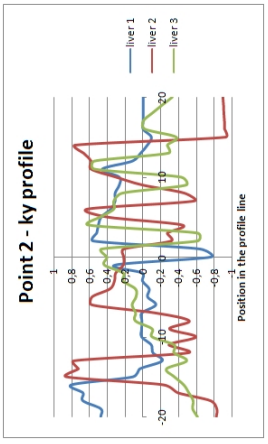


(c) z component of maximal curvature direction

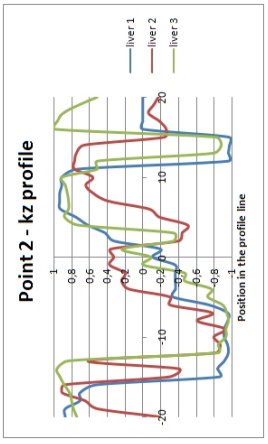
Figure E.1: Curvature direction profile for point #1



(a) x component of maximal curvature direction

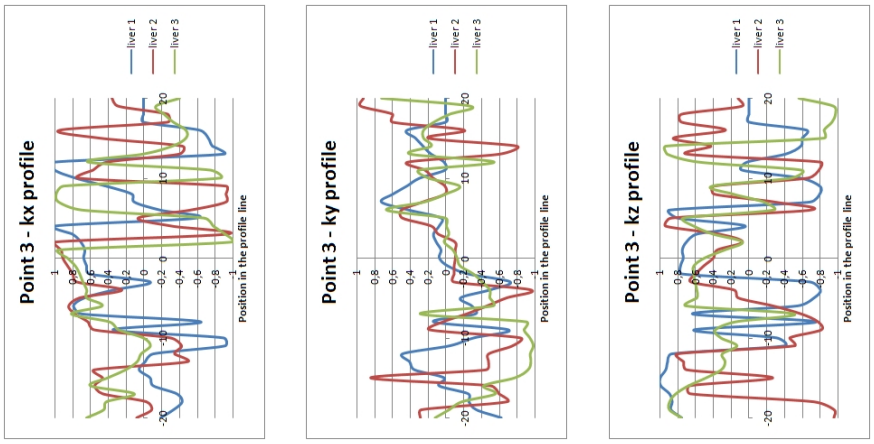


(b) y component of maximal curvature direction



(c) z component of maximal curvature direction

Figure E.2: Curvature direction profile for point #2

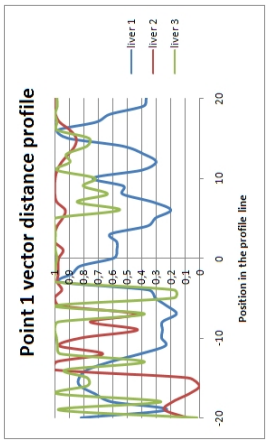


(a) x component of maximal curvature direction (b) y component of maximal curvature direction (c) z component of maximal curvature direction

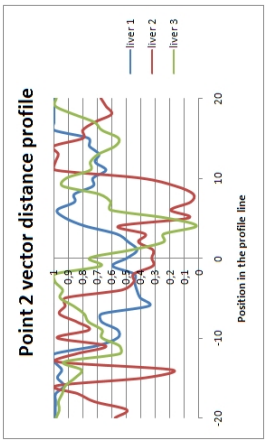
Figure E.3: Curvature direction profile for point #3

E.2 Vector distance profile

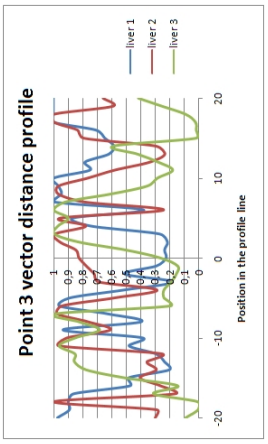
The graphs in this section show the profile of the distance (defined as in Appendix B, definition 2) of the maximal curvature direction for three different livers at three different points.



(a) vector distance profile for point #1



(b) vector distance profile for point #2



(c) vector distance profile for point #3

Figure E.4: Curvature direction profile using vector distance metric

Index

Bibliography

[CHTH94] T.F. Cootes, A. Hill, C.J. Taylor, and J. Haslam. The use of active shape models for locating structures in medical images. *Image and Vision Computing*, 12:355–366, 1994.

[Dav02] R. H. Davies. *Learning shape: Optimal models for analysing natural variability*. PhD thesis, Department of Imaging Science and Biomedical Engineering, Stopford building, University of Manchester, 2002.

[dC76] M. do Carmo. *Differential geometry of curves and surfaces*. Prentice-Hall, 1976.

[Der87] Rachid Deriche. Using canny's criteria to derive a recursively implemented optimal edge detector. *International Journal of Computer Vision*, pages 167–187, 1987.

[DMSB99] Mathieu Desbrun, Mark Meyer, Peter Schrder, and Alan H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. *Proceedings of ACM SIGGRAPH*, pages 317–324, 1999.

[EPH+08] O. Ecabert, J. Peters, H.Schramm, C. Lorenz, J. von Berg, M.J. Walker, M. Vembar, M.E. Olszewski, K. Subramanyan, G. Lavi, and J. Weese. Model-based segmentation of the heart in ct images. *IEEE Transactions on Medical Imaging*, 27:1189–1201, 2008.

[FRZ+05] Daniel Freedman, Richard J. Radke, Tao Zhang, Yongwon Jeong, D. Michael Lovelock, and George T.Y. Chen. Model-based segmentation of medical imagery by matching distributions. *IEEE Transactions on Medical Imaging*, 24, 2005.

[FtHRKV] L.M.J. Florack, B.M. ter Haar Romeny, J.J. Koenderink, and M.A. Viergever. On scale and the differential structure of images. *Image and Vision Computing*.

[GI04] Jack Goldfeather and Victoria Interrante. A novel cubic-order algorithm for approximating principal direction vectors. *ACM Transactions on Graphics*, 23:46–63, 2004.

[GvL96] G. Golub and C. van Loan. *Matrix computations*. The Johns Hopkins University Press, 1996.

[HP04] Klaus Hildebrandt and Konrad Polthier. Anisotropic filtering of non-linear surface features. *Computer Graphics Forum*, 23:391–400, 2004.

[HPW05] Klaus Hildebrandt, Konrad Polthier, and Max Wardetzky. Smooth feature lines on surface meshes. *Eurographics Symposium on Geometry Processing*, 2005.

[KHL+09] Tobias Klinder, Patrick Hein, Cristian Lorenz, Hans Barschdorf, Thomas Blaffert, and SebastianDries. Liver modeling, segmentation and tracking. Technical report, Philips Research Hamburg, 2009.

[KMW96] J.T. Kent, K.V. Mardia, and J.M. West. Ridge curvaes and shape analysis. *The British Machine Vision Conference*, pages 43–52, 1996.

[KWT88] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: active contour models. *International Journal of Computer Vision*, pages 321–331, 1988.

[LFM96] Richard Lengagne, Pascal Fua, and Olivier Monga. Using crest lines to guide surface reconstruction from stereo. *Proceedings of the 13th International Conference on Pattern Recognition (ICPR96)*, 1:847–850, 1996.

[LLSV99] Antonio M. Lpez, Felipe Lumbreras, Joan Serrat, and Juan J. Villanueva. Evaluation of methods for ridge and valley detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:327–335, 1999.

[Lp00] Antonio M. Lpez. *Multilocal methods for ridge and valley delineation in image analysis*. PhD thesis, Universitat Autonoma de Barcelona, 2000.

[MB92] Olivier Monga and Serge Benayoun. Using partial derivatives of 3d images to extract typical surface features. *IEEE Proceedings of the Third Annual Conference of Al, Simulation and Planning in High Autonomy Systems Theme: Integrating Perception, Planning and Action*, pages 225–236, 1992.

[MBF92] Olivier Monga, Serge Benayoun, and Olivier D. Faugeras. From partial derivatives of 3d density images to ridge lines. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 354–359, 1992.

[MDSB03] Mark Meyer, Mathieu Desbrun, Peter Schrder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *Proceedings of Visualization and Mathematics III*, pages 35–37, 2003.

[MvdEV96] J.B Antoine Maintz, Petra A. van den Elsen, and Max A. Viergever. Evaluation of ridge seeking operators for multimodality medical image matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:353–365, 1996.

[OBS04] Yutaka Ohtake, Alexander Belayaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Transactions on Graphics*, 23:609–612, 2004.

[SF04] Georgios Stylianou and Gerald Farin. Crest lines for surface segmentation and flattening. *IEEE Transactions on Visualization and Computer Graphics*, 10:536–544, 2004.

[Thi96] Jean-Philippe Thirion. The extremal mesh and the understanding of 3d surfaces. *International Journal of Computer Vision*, 19:115–128, 1996.

[WPK+] J. Weese, V. Pekar, M. Kaus, C. Lorenz, and P. Rsch. Shape constrained deformable models for 3d medical image segmentation. *Information Processing in Medical Imaging: 17th International Conference, IPMI 2007*.

[YBS05] Shin Yoshizawa, Alexander Belayaev, and Hans-Peter Seidel. Fast and robust detection of crest lines on meshes. *Symposium on solid and physical modelling*, pages 227–232, 2005.

[YBYS08] Shin Yoshizawa, Alexander Belayaev, Hideo Yokota, and Hans-Peter Seidel. Fast, robust, and faithful methods for detecting crest lines on meshes. *Computer Aided Geometric Design*, 25:545–560, 2008.

[ZGM09] Mao Zhihong, Cao Guo, and Zhao Mingxi. Robust detection of perceptually salient features on 3d meshes. *Visual Comput*, 25:289–295, 2009.

[ZGX05] Xiahai Zhuang, Lixu Gu, and Jianfeng Xu. Medical image alignment by normal vector information. *Computational Intelligence and Security*, pages 890–895, 2005.