Technische Universität München
Fakultät für Mathematik

Automatic segmentation of Carotid Artery in freehand Ultrasound

Paulo Laurent Waelkens

München
2011

Abstract

In this work we present a fully automatic algorithm for the segmentation of carotid artery walls in tracked 2D freehand ultrasound datasets.
The algorithm can be divided into the following steps:

1. a modified version of the Hessian matrix based tubular detection algorithm proposed by Frangi *et al.* The image derivatives are defined in a more suitable way for ultrasound images and the vesselness measure is modified to consider the physical characteristics of ultrasound based imaging.

2. a simple centerline detection algorithm

3. a variational level-set segmentation algorithm applied to all frames simultaneously, using GAC, localized statistics, and a 3D smoothness correction term

We evaluate the algorithm on X carotid US datasets taken from different people and of different overall quality.

Contents

Chapter 1

Introduction

The purpose of this work is to expand the usefulness of current medical imaging tools.

Medical ultrasound (US) devices are much more widely available than X-ray/CT or MRT devices. Examinations with ultrasound devices are also cheaper, and physicians have imediate access to the medical images, unlike with X-ray and MRT imaging modalities. Alas, in contrast to many US devices, CT and MRT imaging yields 3D image datasets, which are much more useful for some diagnostics. Fortunately it is possible to artificially create 3D US image datasets by tracking the position of 2D images and arranging them together, interpolating all points for which there is no data. This is the idea behind the so called tracked freehand ultrasound imaging modality.

We believe medical ultrasound imaging has a lot of potential, thus the algorithm proposed in this thesis was developed specifically with ultrasound datasets in mind.

Many vascular surgeons use ultrasound devices to examine the carotid arteries for the presence of atheromatous placque and anatomical changes in this vessel‘s lumen. Still, in serious cases (e.g. when a endarterectomy is being considered) a CT angiography of the carotid arteries (which gives a nice 3D image dataset where the vessel lumen and calcifications in the vessel walls are clearly visible) is usually made. This procedure is invasive and the radiocontrast materials needed for it are not particularly healthy, there are risks of serious allergic reactions to the contrast material and of kidney failure. If tracked freehand ultrasound is made good enough, the CT angiography and all its unpleasantness might be completely avoided for carotid examinations. Which is why we developed an algorithm to segment the internal blood vessel walls (thus, also segmenting the lumen, like the CT angiography) of the carotid artery.

In particular, we assume that the algorithm runs over a 3D image dataset reconstructed from tracked 2D freehand ultrasound images of the carotid artery region. We assume no imaging artifacts arise through the creation of the 3D image dataset.

1.1 US carotid works

Zahalka and Fenster [8] propose a semi-automatic carotid wall segmentation algorithm for 3D US datasets. For each slice, a seed point is manually set. From this seed point rays are extended at every 5°. Along each ray, potential edge points are classified in 5 classes. A point is chosen as edge point based on its intensity classification and on the proximity of potential edge points from the surrounding rays. Once all edge points are chosen, a Geometrically deformable model (GDM, Miller *et al* (1991), [9]) is initialized and optimized with the use of an internal and external force.
Mao *et al* [10] propose a similar semi-automatic algorithm. A notable difference, is that in their work they propose the use of global statistics for each slice.
Chen *et al* [11] used a balloon model, adapted to US imaging characteristics, to segment 3D carotid US datasets.
Fenster *et al* [12] developed a reconstruction and visualization tool for tracked 2D US images. They propose an extension to their framework to include automatic (or semi-automatic) segmentation algorithms, a potential application for this thesis.
Many publications regarding US carotid segmentation propose methods to measure the IMT (intima media thickness) from longitudinal (in the direction of the blood flow) non tracked 2D US images. This is a lot simpler than 3D segmentation but is still clinically useful. Loizou *et al* [15] give a short overview of current approaches to IMT segmentation and propose a snakes based, semi-automatic segmentation approach, with previous despeckling of the image through the use of a linear scaling filter [14]. Loizou *et al* [16] also evaluate different snakes algorithms to obtain the IMT segmentation.

1.2 Some interesting tubular structure detection approaches

Of particular interest for this thesis is the tubular structure detection algorithm proposed by Frangi [7]. The algorithm gives a 'vesselness' value to all points in a 3D image dataset. The vesselness value is a function of the eigenvalues of the Hessian matrix of the image data. The eigenvalues of the Hessian matrix correspond to the second derivatives of the image data in the direction of the associated eigenvector. The partial derivatives of the image data are obtained through the convolution of the corresponding partial derivatives of a 3D Gaussian Kernel with the image data. The scale of the tubular structure being detected is associated to the standard deviations of the Gaussian Kernel.
This thesis proposes some adaptions to this algorithm for 3D ultrasound datasets.
Another very common tubular structure detection approach is the structure tensor (sometimes referred to as matrix of second moments). The eigenvalues and eigenvectors of the structure tensor summarize the local distribution of the gradient vector. These 2 approaches, the scale space considerations and

modifications proposed in this work will be approached in detail in section 2.3.

1.3 Segmentation literature

There is a very large collection of image segmentation approaches presented in the literature. A general overview of ultrasound segmentation approaches is presented by Noble *et al* [4]. Kirbas and Quek [2] present a very thorough review of vessel segmentation approaches. An overview of 3D segmentation approaches for vessel lumen (i.e. interior) segmentation, especially for MRI and CT, is presented by Lesage *et al* [6].

In this thesis we will use the variational level–set framework with implicit representation of the level–set function proposed by Baust *et al* [1]. For the energy term we use the Geodesic Active Contour energy proposed by Caselles *et al* [**?**] and a local statistics approach, heavily inspired by the approaches proposed by Lankton *et al* [**?**]. Furthermore we add an energy term to guarantee a relatively smooth vessel wall mesh. Here we use ideas inspired by the work of Mori *et al* [**?**].

The chosen segmentation algorithm, the justification for its choice and particular considerations with regard to the specific tracked ultrasound setting are approached in detail in section 2.5.

Chapter 2

Algorithms

2.1 Chapter overview

This chapter presents the theoretical background of the algorithms used for our proposed automatic segmentation solution.
The first section briefly explains how all 3 algorithms (tubular structure detection, centerline extraction and vessel wall segmentation) are ment to work together, to obtain an automatic carotid artery segmentation.
In section 2.3 we present the the tubular structure detection algorithms mentioned in the introduction, the Frangi vesselness and the structure tensor algorithms, in detail. Then we do a short excursion on how image derivatives are usually defined, in particular how they are defined for Frangi‘s algorithm. Then we propose a different way to obtain image derivatives which leads to better results for tubular structure detection algorithms applied to US images of blood vessels.
Then we propose our modified version of Frangi‘s vesselness algorithm, using the new way to obtain image derivatives and the fact that the US signal response is much stronger in the direction parallel to the US beam direction than in the orthogonal directions to the beam.
In the next section we present a simple algorithm for centerline extraction, given the results obtained by the vesselness algorithm.
In the last section we present the variational level-set segmentation algorithm proposed for this application.

2.2 The complete setup

The automatic segmentation needs to know the range of vessel sizes (i.e. scales) it is ment to search for. More specifically, we must set a finite number of vessel radiuses the algorithm is going to run over (basically, the same algorithm will be run with different sizes of the smoothing mask). Automatic scale selection

algorithms, e.g. Lindeberg‘s approach [5], could probably replace this step.
The vesselness parameters and all the thresholding rules for centerline detection and segmentation have to be set somehow. A training data-set would be useful for this, but the hope is that the algorithms are robust enough to work with any halfway reasonable choice of these parameters. More on parameter setting in chapter **??**.
Once all parameters are set, the tubular structure detection algorithm is run over the 3D dataset, once for each of the chosen vessel radiuses. Each run calculates and stores:

- the vesselness measure for all points in the dataset for the given vessel radius choice
- the eigenvector associated to the smallest eigenvalue of the Hessian matrix, for the given radius size choice, for all points in the dataset

Then, for each point in the dataset, the maximum vesselness measure over all chosen vessel radiuses is selected and stored, its associated eigenvector and the chosen vessel radius are also stored. All other calculated data is discarded at this point.
At this point we have, for each point in the dataset:

- the largest vesselness measure for this point over all vessel radiuses contemplated
- the vessel radius for which the vesselness measure is the highest
- the eigenvector associated to the smallest eigenvalue of the Hessian matrix obtained for the chosen vessel radius

So, if the vesselness measure at a given point is large enough, we could assume that it belongs to the centerline of a tubular structure in the range of sizes we are contemplating. Furthermore, we would know the approximate direction of the centerline at this given point, because of the stored eigenvector and the approximate vessel radius at this point in the tubular structure.
These $(vesselness, chosenvesselradius, eigenvector)(x)$ for each point x in the dataset data structures are used as input for the centerline detection algorithm. Once the centerlines are obtained, based on the thresholding rules set at the beginning of the algorithm, we have:

- a list of centerlines, each centerline consisting of a number of voxels
- to each voxel in a centerline we have an associated vessel radius
- to each voxel we also have the approximate direction of the centerline

Storing the approximate direction of the centerline given by the calculated eigenvectors is not strictly necessary once we actually have the centerlines. But, it adds to the robustness of the algorithm, since the reconstruction of the volume

dataset might not be perfect.
The next step is to try a segmentation around all centerlines identified.
Up to this point, the quality of the 3D reconstruction was not such a big issue. Basically because the vesselness values are obtained from a smoothed version of the dataset, which allows for some imprecision.
However, gross imprecision is not acceptable for the segmentation step. Which is why we choose to run the segmentation algorithms in the original 2D US datasets used to obtain the 3D reconstruction.
We identify to which frame each centerline point belongs to and use the approximate direction of the centerline and the chosen vessel radius as parameters to draw an ellipse in the frame, centered around the centerline point. This ellipse is the initialization of the segmentation algorithm. Furhermore we limit the region of interest of the segmentation, stating that the contour which we are trying to segment is contained in a circular region, with radius given by $r_{vessel} * k_{ROIthreshold}$, where r_{vessel} is the chosen vessel radius and $k_{ROItreshold}$ is one of the constants set at the beginning. e.g. $k_{ROItreshold} = 2$.
Clearly, this segmentation setup will fall apart if the contour on the 2D US frame does not roughly correspond to an ellipse. This could happen if the centerline is orthogonal to the frame‘s normal vector, e.g. in longitudinal (in the direction of the blood flow) images. In this case, the segmentation would have to be run directly on the reconstructed 3D dataset, and the success of the segmentation would become contingent on an excellent 3D reconstruction.
None of the datasets used for experiments during the development of this thesis contain longitudinal images, so the chosen setup works fine.

2.3 Tubular structure detection algorithm

A precise definition of a tubular structure is:
Let there be a simple curve in 3D space, the *centerline.*
A tubular structure has a radius, r_x, associated to each point x of it‘s centerline. For each point x of the centerline, let there be a circle, C_x with radius r_x, belonging to the plane orthogonal to the tangent direction of the centerline at x. The tubular structure associated to the given centerline is defined as:

$$\bigcup\{closure(C_x)|x \in Image(centerline)\} \tag{2.1}$$

Intuitively one could think of a tubular structure as a garden hose and its interior.

2.3.1 The Frangi vesselness algorithm

The Frangi vesselness algorithm [7] is an algorithm to detect tubular structures of a given radius, r_{tub} , in 3D image datasets. More precisely, to detect groups of points with a broadly similar signal intensity whose boundary to points of notably different signal intensities is the boundary of a tubular structure.

One should note that the 'colors' in medical image datasets are usually a translation of some kind of signal intensity (as is the case in US images).
The algorithm gives a 'vesselness' value to all points in the 3D dataset. Ideally the points with the highest values will belong to a tubular structure‘s centerline. The farther away an interior point is from the tubular structure‘s centerline, the lower its value will be. Points outside the tubular structure should have even lower values.
The vesselness value at a point is a function of the eigenvalues of the Hessian matrix of the image dataset at the given point.
The second partial derivatives of the image data, needed for the Hessian matrix, are obtained through the convolution of the image data with the second partial derivatives of a discrete Gaussian kernel of standard deviation σ, $\sigma = f(r_{tub})$.
Alternatively, one could smooth the image data with the discrete Gaussian kernel, and take the derivatives of this smoothed dataset. More on this in 2.3.3.
The intuition is that, given an adequate scale, a centerline point, x, will have the following properties:

1. the curvature of the image data in the tangent direction of the centerline, v_1, is zero
2. if v_1, v_2, v_3 form an orthogonal basis of $\Re^3$, the curvature in the direction of v_2 and v_3 should be rougly equal and 'large'

It turns out, that the eigenvalues of the Hessian matrix of a function correspond to the second derivatives of the function in the direction of the associated eigenvectors, assuming normalized eigenvectors. See A for a detailed explanation.
Furthermore, if a point actually belongs to a tubular structure, the eigenvector associated to the smallest eigenvalue (in absolute value) will (roughly) correspond to the local tangent direction of the centerline.

To obtain the vesselness measure proposed by Frangi, we sort the eigenvalues

$$|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$$

and obtain the quantities

- $R_B = \frac{|\lambda_1|}{\sqrt{|\lambda_2 * \lambda_3|}}$, a blobness measure
- $R_A = \frac{|\lambda_2|}{|\lambda_3|}$
- $S = \lambda_1^2 + \lambda_2^2 + \lambda_3^2$, a relative significance measure

At a centerline point, the following conditions are expected to be satisfied:

1. $|\lambda_1| \approx 0$
2. $|\lambda_2| \approx |\lambda_3| >> |\lambda_1|$
3. $sign(\lambda_2) = sign(\lambda_3)$

Which means

1. the curvature in the direction of smallest curvature, which should correspond to the direction tangent to the centerline, is very small relative to the curvature in the directions of the other eigenvectors
2. the curvatures in the plane orthogonal to the centerline tangent should be broadly equal
3. obviously, these curvatures need to have the same sign:
    - positive means a convex structure on the orthogonal plane
    - negative a concave structure on the orthogonal plane

So, in an ideal case:

1. $R_B = 0$
2. $R_A = 1$
3. $S >> 0$
4. the curvature has the correct orientation (convex or concave)

Frangi then introduces one parameter for each of these relations. We call the parameters α, β and γ. The vesselness measure for a point x in 3D space is defined as:

$$\begin{aligned} V(x) &= 0 && \text{if curvature has wrong orientation} \\ V(x) &= (1 - exp(-\frac{R_A^2}{2\alpha^2})(exp(-\frac{R_B^2}{2\beta^2})(1 - exp(-\frac{S^2}{2\gamma^2})) && \text{otherwise} \end{aligned}$$

This algorithm can readily be extended to a so called 'multi-scale' approach. Here, the vesselness values obtained for a particular choice of the standard deviation of the Gaussian Kernel used to obtain the derivatives is compared to the vesselness values obtained for different choices of the standard deviation. This way it is possible to find tubular structures whose radius around the centerline changes for different points. Since the Gaussian kernel is chosen to obtain the derivatives, the comparison between values obtained at different scales is very straightforward. More on this in 2.3.3.

2.3.2 Structure Tensor

The structure tensor algorithm is an algorithm that can be used to detect tubular structures in 3D image datasets, as the Frangi filter. Basically, it evaluates the second moments of the flipped image data around every point p in the image and gives a measure based on the evaluated moments.
A probability distribution function $w(r)$, with support of $n + 1$ points, is given

(or chosen, e.g. discretization of a standard 3D Gaussian kernel). The first derivatives of the image data are obtained through the convolution of the image data with the derivatives of a discrete Gaussian kernel of standard deviation σ, $\sigma = f(r_{tub})$ More on this in 2.3.3.

Then the second moments of the flipped image data are evaluated at every point p in the image. Here, the second moments are defined as:

$$M_{xx}(p) = \sum_{r=0}^{n} w(r)(I_x(p-r))^2$$
$$M_{xy}(p) = \sum_{r=0}^{n} w(r)I_x(p-r) * I_y(p-r)$$
$$M_{xz}(p) = \sum_{r=0}^{n} w(r)I_x(p-r) * I_z(p-r)$$
$$M_{yy}(p) = \sum_{r=0}^{n} w(r)(I_y(p-r))^2$$
$$M_{yz}(p) = \sum_{r=0}^{n} w(r)I_y(p-r) * I_z(p-r)$$
$$M_{zz}(p) = \sum_{r=0}^{n} w(r)(I_z(p-r))^2$$

where $(I_x, I_y, I_z)(k)$ is the gradient vector at the point k obtained through the convolutions with the Gaussian kernel derivatives. Notice we 'flip' the indexes of the gradient. Intuitively we should evaluate $\sum_{r=0}^{n} w(r)(I_x(p+r))^2 = E_w((I_x(p))^2)$, but we don't. The reason we take the flipped image data, $I_k(p-r)$, is that this way we can evaluate M_{xx} and the other second moments through discrete convolutions, which is computationally convenient. It should be noted that if $w(r)$ is a symmetric distribution (which is usually the case), the flipping of the image data is not really necessary.

Finally, we make the eigenvalue decomposition of the structure tensor:

$$\begin{pmatrix} M_{xx} & M_{xy} & M_{xz} \\ M_{xy} & M_{yy} & M_{yz} \\ M_{xz} & M_{yz} & M_{zz} \end{pmatrix}$$

The meaning of the eigenvalues of this matrix is a lot less straightforward than the meaning of the eigenvalues of the Hessian matrix.

Basically, a large eigenvalue (in $|.|$) indicates that the gradient has changed a lot along the direction of the associated eigenvector.

Thus, for a tubular structure, we would have two relatively large eigenvalues, whose associated eigenvectors span the plane orthogonal to the centerline, and a relatively small eigenvalue associated to the direction of the centerline (approximately). One should notice the similarity to Hessian matrix based algorithms (like Frangi's algorithm) at this point.

There are many ways to measure if these properties are being fulfilled, for example:
If we sort the eigenvalues: $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$, we could define

- a measure of relative significance: $S = |\lambda_2| + |\lambda_3|$
- a blobness measure: $R_B = \frac{|\lambda_1|}{\sqrt{|\lambda_2 * \lambda_3|}}$
- $R_A = \frac{|\lambda_2|}{|\lambda_3|}$

At a centerline point, the following conditions are expected to be satisfied:

1. $|\lambda_1| \approx 0$
2. $|\lambda_2| \approx |\lambda_3| >> |\lambda_1|$

So, in an ideal case:

1. $S >> 0$
2. $R_B = 0$
3. $R_A = 1$

We could define a measure for a point x in 3D space as:

$$STMeasure(x) = S * R_A * exp(-R_B)$$

We could make this measure more flexible by inserting some parameters to quantify the relative significance of S, R_A and R_B. Clearly, any measure that gets larger for $S \to \infty$, $R_B \to 0$ and $R_A \to 1$ would do the trick.
Like the Frangi algorithm, this algorithm can readily be extended to a so called 'multi-scale' approach. Here, the measure values obtained for a particular choice of the standard deviation of the Gaussian Kernel used to obtain the derivatives is compared to the measure values obtained for different choices of the standard deviation. This way it is possible to find tubular structures whose radius around the centerline changes for different points. Since the Gaussian kernel is chosen to obtain the derivatives, the comparison between values obtained at different scales is very straightforward. More on this in 2.3.3.
As mentioned above, the interpretation of the eigenvalues of the structure tensor is not very straightforward, in stark contrast to the interpretation of the eigenvalues of the Hessian matrix, see A . Which is why we chose only to consider Hessian matrix algorithms for tubular structure detections in our new algorithm.

2.3.3 Image derivatives and scale space representations

2.3.3.1 On image derivatives

Images can be viewed as discrete functions, obtained through discrete sampling of some (real) function defined in a continuous domain. In a strict sense, all differentiable functions need to be continuous. Discrete functions are obviously not continuous, thus not differentiable in the usual sense.

Regardless, a measure of local variation and of local structure of discrete functions is clearly useful. This has led to a large amount of definitions of 'derivatives' for discrete functions. In the image analysis and computer vision communities in particular, some definitions of image derivatives have become particularly widespread. Very common examples are:

1. convolutions with the Sobel, Prewitt and Roberts' cross operators and many others based on finite differences masks with small support, e.g. 27 voxel mask for 3D Sobel
2. convolutions with partial derivatives of multi-dimensional (2D, 3D) discrete Gaussian kernels with any desired standard deviation σ and usually much larger support, e.g. 250^3 voxel mask for $\sigma = 50$

This work proposes another image derivative definition.

The possibility to control the size of the derivative mask, as in item 2, is an obvious improvement on the flexibility of algorithms to deal with the same image at different resolutions, or to highlight features too large for fixed mask size approaches. It also permits the detection of features of different sizes in a unified and consistent manner.

2.3.3.2 Scale space representation

The Frangi algorithm and many other 'multi-scale' algorithms propose the use of partial derivatives of Gaussian kernels to obtain image derivatives at any desired scale. The reason for their choice of the Gaussian kernel is that it is the only function which satisfies a number of special properties, usually referred to as the *scale space axioms.* Let $f(x, y, z)$ be the discrete signal. And let $g(x, y, z; t)$ be a smoothing function, with scale parameter t. The (linear) *scale space representation* of f is given by:

$$L(x, y, z; t) = f(x, y, z) * *g(x, y, z; t)$$

where $**$ denotes convolution.

At this point, it should be noted that to obtain an image derivative at a given scale, t, one can apply a simple finite differences mask (e.g. Sobel operator) to the smoothed image, $L(x, y, z; t)$. Convolutions satisfy the associative and commutative properties. So, smoothing and applying a finite differences mask on an image or convolving the image with a partial derivative of the smoothing mask are equivalent, assuming the derivative of the smoothing mask was obtained by using the same finite differences schema.

The Gaussian kernel is the only smoothing function satisfying the following properties:

1. linearity
2. shift invariance

3. semi-group structure:
   $g(x, y, z; t_1) * *g(x, y, z; t_2) = g(x, y, z; t_1 + t_2)$
4. the associated cascade smoothing property:
   $L(x, y, z; t_2) = L(x, y, z; t_1) * *g(x, y, z; t_2 - t_1)$, with $t_2 > t_1$
5. existence of an infinitesimal generator A, such that
   $\partial_t L(x, y, z; t) = (AL)(x, y, z; t)$
6. non-enhancement of local extrema
7. rotational symmetry
8. scale invariance:
   $\hat{g}(\omega_x, \omega_y, \omega_z; t) = \hat{h}(\frac{\omega_x}{\varphi(t)}, \frac{\omega_y}{\varphi(t)}, \frac{\omega_z}{\varphi(t)})$
   for some function h and some function φ
9. positivity
10. normalization

Where $g(x, y, z; t)$ is a gaussian kernel with diagonal correlation matrix $\Sigma = t * Identity$ and mean $\mu = 0$.
These properties where set with the following problem in mind:
One would like to analyse a sequence of gradually smoother versions of the same image. The smoothing should gradually dampen local features. After each smoothing only coarser features remain. Thus it should be possible to detect local features before smoothing and gradually coarser features after some degree of smoothing. Furthermore, it should be possible to compare features detected at different levels of smoothing in a consistent way. Also, the smoothing should not introduce artefacts into the images.
No assumptions are made on the features one wants to detect. So no kind of *a priori* knowledge is assumed.
The idea of these assumptions if to mimic how the biological vision system works.
These are very general assumptions in the sense that no assumptions are made on the features. But they are also very restrictive in the sense that it should be possible to compare features detected at *any* level of smoothing in a consistent manner, e.g. comparing the image of a same blade of grass seen from a meter away or seen from the moon should not be an issue.

2.3.3.3 Carotid US image particularities

Our current setting deals with a different problem:
We want to find tubular structures in US images of the carotid artery. So we know the shape of the features we want to detect. We also have a range of possible sizes for the blood vessel diameter: in our images' case from 70 to 95 pixels.

Furthermore we know the images will have the particular characteristics of US, namely:

- speckle
- a direction dependency on signal strength:
  The signal response is much stronger in the direction parallel to the US beam direction than in the orthogonal directions to the beam
- shadowing

We should also consider anatomical features of the region around the carotid artery. We can observe rather abrupt changes in the tissue. I.e. the signal response of relatively (e.g. 20 pixels/voxels away) close neighbor points has no visible correlation to the signal response of the contemplated point. This becomes especially problematic near the carotid bifurcation area.
Especially because of the direction dependency in signal strength and the abrupt changes in signal response, the Gaussian kernel convolution based derivatives do not yield convenient results. The basic problem is that the Gaussian kernel is far too wide. More precisely, points very far away from the center of the kernel have too high weights. So signal intensities at voxels distant from the contemplated voxel have a far too large influence in the derivative values. It is mainly this limitation that we want to correct.
In section 3.1 we show how much better our prosed algorithm is for finding vessel centerline points compared to the (Gaussian kernel using) Frangi algorithm.

2.3.4 New image derivative definition

The Gaussian kernel proved to be too wide to yield good results. Regardless, we need to measure derivatives somehow, and at different scales in a flexible way. So we propose a different smoothing function to replace the Gaussian kernel. Once we have this function, we take a discrete sampling of it as our smoothing mask.
This function should satisfy some general requirements to be an acceptable replacement for the Gaussian kernel here

- linearity
- shift invariance
- non-enhancement of local extrema
- rotational symmetry
- positivity
- normalization
- at least continuous second partial derivatives in all spatial directions

Special attention should be given to the rotational symmetry property. The algorithm should be able to find tubular structures with any desired orientation equally well. So, we must not favor any particular direction. Where that not the case, one would not be stuck with large 3D masks. The function should also satisfy some very specific properties, to be especially useful for our algorithm. The function should be tailor-made to yield good second partial derivative masks, since we want to obtain a good Hessian matrix of the image data. We are particularly interested in the shape of the masks used to obtain ∂_{xx}, ∂_{yy} and ∂_{zz}. We design our new masks to obtain good responses in ideal US vessel images.

- the second partial derivatives should have broadly the same shape as those of the Gaussian kernel, i.e. ∂_{xx}, ∂_{yy} and ∂_{zz} should be reasonable measures of curvature, and ∂_{xy}, ∂_{xz} and ∂_{yz} should be reasonable measures of saddle-pointness in the diagonal directions
- the interior of the blood vessels in US images has a relatively uniform signal intensity distribution. Thus, ∂_{xx} *et al* should assign approximately the same weights for all points inside the vessel
- the area around the vessel lumen which we can assume to belong to the vessel wall should not be very thick. This is exactly where the Gaussian kernel flounders.
- the function should be as simple as possible!

Given all these constraints, we experimented with a class of functions of the form:

$$new_g(x, y, z; t) = k * exp(-P(x, y, z; t)) \tag{2.2}$$

where k is a positive normalizing constant and P is a polynomial. Notice how the Gaussian kernel is a special case in this class of functions.
Any function in this class satisfies:

- linearity
- shift invariance
- positivity
- normalization
- at least continuous second partial derivatives in all spatial directions

Rotational invariance demands that $P(x, y, z; t) = P(\sqrt{x^2 + y^2 + z^2}; t)$.
Although it is not strictly necessary, we restrict P to polynomials of even powers. That the second partial derivatives should have broadly the same shape as those of the Gaussian kernel leaves a lot of room for choice. For our application we chose P as polynomial of degree 12, only with even powers. The normalization

is done numerically. We used the following function:
Let $r := \sqrt{x^2 + y^2 + z^2}$

$$\begin{aligned} new_g(x, y, z; t) =& k \cdot exp(-0.5(c_{12}(\frac{r \cdot 0.408}{t})^{12} + c_{10}(\frac{r \cdot 0.408}{t})^{10} + && (2.3) \\ & c_8(\frac{r \cdot 0.408}{t})^8 + c_6(\frac{r \cdot 0.408}{t})^6 + c_4(\frac{r \cdot 0.408}{t})^4 + && (2.4) \\ & c_2(\frac{r \cdot 0.408}{t})) && (2.5) \\ & && (2.6) \end{aligned}$$

with

$$\begin{aligned} c_{12} &= 1 \\ c_{10} &= 409.6 \\ c_8 &= 128 \\ c_6 &= 42.66 \\ c_4 &= 16 \\ c_2 &= 8 \end{aligned}$$

obtained through numerical fitting, more details in Appendix B. t corresponds to the vessel radius.
Using exponentials of polynomials of other degrees is clearly possible. Once the choice of the degree is made, a parameter fitting should be made to ensure that ∂_{xx}, ∂_{yy} and ∂_{zz} assign approximately the same weights for all points inside the vessel, i.e. ∂_{xx}, ∂_{yy} and ∂_{zz} should be 'flat' around the middle point of the mask. Once this fitting is made, one can observe that, the higher the degree of the polynomial chosen, the smaller the support of the positive terms in ∂_{xx}, ∂_{yy} and ∂_{zz}, i.e. the smaller the width of the area around the lumen assumed to belong to the vessel wall.

2.3.5 Vesselness algorithm adapted to US images

We propose a Hessian matrix based tubular detection algorithm. The image derivatives needed for the Hessian matrix are obtained through the convolution of the image data with the new smoothing function proposed in section 2.3.4. The eigenvalues of the Hessian matrix are calculated, and sorted:

$$|\lambda_1| \le |\lambda_2| \le |\lambda_3|$$

It is convenient to notice that here the Hessian matrix is real and symmetric (and Hermitian), so all its eigenvalues and eigenvectors are real. The eigenvector associated to λ_1 is obtained indirectly, by finding the eigenvector associated to λ_3 with the power method, then finding the eigenvector associated to λ_2 through the power method, with starting iteration vector orthogonal to λ_3's

eigenvector. Or through some standard algorithm...
As mentioned previously, the US signal response is much stronger in the direction parallel to the US beam direction than in the orthogonal directions to the beam. Thus, at a blood vessel, the curvature measured in the direction of the US beam should be larger than the curvature measured in the direction orthogonal to this direction and to the centerline.
The approximate direction of the US beam, d_{beam} is assumed known for all points in the volume. Which is not at all unreasonable, since we assume a tracked ultrasound probe.
Ideally, the direction of the centerline is given by the eigenvector associated to λ_1. The direction of largest measured curvature, given by λ_3's eigenvector should be roughly parallel to the US beam. Otherwise the contemplated point is not in the center of a blood vessel.
So our vesselness measure is defined by 4 relations:

1. $dir = | < \frac{eigenvector_{\lambda_3}}{|eigenvector_{\lambda_3}|}, \frac{d_{beam}}{|d_{beam}|} > |$ a direction component
2. $S = |\lambda_2| + |\lambda_3|$ a measure of relative significance
3. a blobness measure: $R_B = \frac{|\lambda_1|}{\sqrt{|\lambda_2 * \lambda_3|}}$
4. $R_A = \frac{|\lambda_2|}{|\lambda_3|}$

At a centerline point, the following conditions are expected to be satisfied:

1. $|\lambda_1| \approx 0$
2. $|\lambda_2| \approx |\lambda_3| >> |\lambda_1|$
3. $sign(\lambda_2) = sign(\lambda_3)$
4. $dir = 1$

Which means

1. the curvature in the direction of smallest curvature, which should correspond to the direction tangent to the centerline, is very small relative to the curvature in the directions of the other eigenvectors
2. the curvatures in the plane orthogonal to the centerline tangent should be broadly equal
3. obviously, these curvatures need to have the same sign:
   - positive means a convex structure on the orthogonal plane
   - negative a concave structure on the orthogonal plane
4. the largest curvature measured should be in the direction parallel to the US beam

So, in an ideal case:

1. $R_B = 0$
2. $R_A = 1$
3. $S >> 0$
4. the curvature has the correct orientation (convex or concave)
5. $dir = 1$

The modified vesselness measure for a point x in 3D space can be defined as:

$$\begin{array}{ll} V(x) = 0 & \text{if curvature has wrong orientation} \\ V(x) = dir \cdot S \cdot R_A \cdot exp(-R_B) & \text{otherwise} \end{array}$$

We could introduce two parameters, α and β to make it more flexible.

$$\begin{array}{ll} V(x) = 0 & \text{if curvature has wrong orientation} \\ V(x) = \alpha \cdot dir \cdot S \cdot R_A \cdot exp(-R_B \cdot \beta) & \text{otherwise} \end{array}$$

The exact choice of the measure function is a bit arbitrary... Any measure that gets larger for $S \to \infty$, $R_B \to 0$ and $R_A \to 1$ would do the trick.
Since we have chosen not to use the Gaussian kernel to obtain the image derivatives, extending the algorithm to deal with various scales simultaneously is not completely straightforward.
We define an ideal vessel for each scale, obtain all second partial derivatives at the given scale and normalize all partial derivative masks, so that the results over all contemplated scales are the same for the ideal vessel. It is not as elegant as the approach with the Gaussian kernel, but it was a tradeoff we considered more than acceptable, given the great increase in precision at identifying the vessel centerlines.
Our experiments show that the greatest improvement upon Frangi‘s algorithm is given by the choice of the new smoothing function. The beam direction component does not lead to a major improvement at points near the center of the lumen. But, it minimizes false positives in image points outside the vessel lumen.

2.4 Centerline extraction

After running the tubular detection algorithm, points that probably belong to a tubular structure‘s centerline have a relatively high vesselness measure. But, they are not organized into any kind of structure. Sometimes in the image analysis field the term structure is replaced by *skeleton*.
In our case, we want to know the centerlines of all the tubular structures in the range of vessel radius sizes we have chosen.
This way we have the knowledge of where in our 3D dataset the relevant tubular

structures are, which gives us valuable information for the segmenation algorithm.
Furthermore, having knowledge of the structures present in an image dataset is essential for successful image registrations, e.g. registration of a tracked 2D US dataset with a CT Angiography dataset of the same patient. Multi-modal registration is beyond the scope of this work, but we must mention that it is an extremely relevant topic for medical imaging applications.

2.4.1 Algorithm overview

Basically we want to separate our points of interest (the ones with a high vesselness measure) into different groups of points that 'belong together', i.e. points in a group should describe the centerline of a tubular structure.
Two adjacent points that:

1. have a relatively high vesselness measure
2. are close enough to one another
3. share approximately the same estimated centerline direction (given by λ_1's eigenvector, see appendix A for details)

Should belong to the same centerline and are thus grouped together.
Our algorithm can be divided into the following steps:

1. seed point selection (details in section 2.4.3)
2. centerline evolution from each seed point (section 2.4.4)
3. merging of centerlines belonging to the same structure (section 2.4.5)
4. a step to deal with the complications which might arise in the carotid bifurcation area (section 2.4.6)

Since the algorithm is ment to run automatically, all the parameters and thresholding rules are already set when the algorithm is executed.

2.4.2 Global variables and data structures

Each centerline point has the following attributes:

- the position in the dataset (e.g. (x, y, z) coordinates)
- the direction given by the eigenvector associated to the Hessian matrix‘s smallest eigenvalue (in $|\cdot|$) for this point
- the vesselness value for this point

Each centerline has the following attributes:

- an ordered list of centerline points. The first point in the list is at an extremity of the centerline, wlog the beginning, the last point is at the other extremity, the end of the centerline. All points in between roughly follow the evolution of the centerline given by the estimated centerline direction (λ_1's eigenvector at the given point)
- optional attributes would be the highest vesselness value of the listed centerline points and the number of points in the centerline. These attributes would only serve for thersholding purposes (e.g. the centerline cannot be too short)

We also define a number of global variables:

- a 3D matrix of binary values, $B_{centerline}$, which associates a binary value to each point in the dataset. If a given point is associated to a centerline, the binary value is set to 1
- a 3D matrix of binary values, $B_{intersection}$, which associates a binary value to each point in the dataset. If a given point is associated to two or more centerlines, the binary value is set to 1

2.4.3 Seed point selection

We need to automatically select a number of initial centerline points in the 3D dataset from which centerlines can be grown from, we call them *seed* points. These points should have a high chance of belonging to an actual centerline of a structure in the vessel radius size range we are contemplating. Thus, we only choose points with a relatively high vesselness measure. More specifically we set a threshold for the smallest vesselness value considered acceptable. Furthermore, the points should be spread out over the dataset so that all tubular structures of the contemplated size range have a seed point associated to them. Also, since we will grow centerlines from the seed points, it is pointless to choose seed points which are too close to another. Taking these requirements into consideration, we propose the following schema to find our seed points:

1. find the voxel with highest vesselness measure in the dataset, x_{max}
2. take the eigenvector of λ_1 associated to this point, v_{max}
3. consider the plane with normal vector v_{max}, $plane_{max}$
4. now construct two sequences of planes, given by:

$$above_{n_a} = plane_{max} + \beta \cdot n_a$$

$$below_{n_b} = plane_{max} - \beta \cdot n_b$$

with $\beta \leq 1$ and n_a and n_b beloning to subsets of the natural numbers, chosen such that all constructed planes have points belonging to the dataset. The distance between the planes,β, and the subsets n_a and n_b are parameters of the algorithm

5. find local vesselness maximums in each of the planes. The minimal distance between the local maximums is one of the parameters of the algorithm
6. set the seed point vesselness threshold:

$$seed_{treshold} = vesselness(x_{max}) \cdot k_{seed}$$

   where $1 > k_{seed} \geq 0$ is a parameter of the algorithm

7. take all the local maximums obtained that are above the set threshold. These points will be the *seed* points of the algorithm
8. mark all seed points as centerline points

2.4.4 Centerline evolution

We create a centerline for each of the seed points, with the seed point being the only point in the centerline‘s centerline points list.
We then go through all seed points (one at a time to avoid simultaneous access to $B_{centerline}$ and $B_{intersection}$) and evolve centerlines from them.
Let

$$fit_{threshold} = vesselness(x_{max}) \cdot k_{fit}$$

where $1 > k_{fit} \geq 0$ is a parameter of the algorithm.
And let v_{seed} be the eigenvector associated to the seed point, x_{seed}.
We propose the following centerline evolution schema:
First the evolution in the direction given by v_{seed}

1. set $d_{current} = v_{seed}$, the centerline direction of the current point
2. set $x_{current} = x_{seed}$, the current centerline point
3. define the centerline candidate points near the current point:

   - we consider the 3x3x3 voxel box around the current point, so we would have 26 candidate points
   - of these 26 points, we only consider those above the plane containing $x_{current}$ with normal $d_{current}$
   - i.e. of these 26 points we only consider those satisfying:

$$< (x_{candidate} - x_{current}), d_{current} > \quad > 0$$

4. among the candidate points, choose the one, $x_{chosen\ candidate}$, that maximizes the following fit function

   $$fit(x_{candidate}) = | < d_{current}, v_{candidate} > | \cdot vesselness(x_{candidate})$$

   where $v_{candidate}$ is the eigenector associated to the point $x_{candidate}$

5. if $fit(x_{chosen\ candidate}) > fit_{threshold}$ we can try to evolve our centerline! So we:

   - check if $x_{chosen\ candidate}$ is already marked as a centerline point
   - if $x_{chosen\ candidate}$ is not marked as a centerline point we
     - add $x_{chosen\ candidate}$ to the end of the centerline points list
     - set $d_{current} = sign(< d_{current}, v_{chosen\ candidate} >) \cdot v_{chosen\ candidate}$ where $v_{chosen\ candidate}$ is the eigenvector associated to $x_{chosen\ candidate}$ and $sign(\cdot)$ is the sign function
     - mark $x_{chosen\ candidate}$ as a centerline point
     - set $x_{current} = x_{chosen\ candidate}$
     - start over from step 3
   - if $x_{chosen\ candidate}$ is marked as a centerline point we
     - add $x_{chosen\ candidate}$ to the end of the centerline points list
     - mark $x_{chosen\ candidate}$ as an intersection point
     - and stop

6. if $fit(x_{chosen\ candidate}) \leq fit_{threshold}$ we stop

We do not want the evolution to suddenly change direction, which is why we set the new current direction by:

$$d_{current} = sign(< d_{current}, v_{chosen\ candidate} >) \cdot v_{chosen\ candidate}$$

i.e., we force the new evolution direction to have a positive scalar product with the evolution direction of the previous iteration.

Then we do the evolution in the direction given by $-v_{seed}$. The only differences in this direction are:

- we initialize $d_{current} = -v_{seed}$
- if we add a point to the centerline point list, we add it at the beginning instead of at the end of the list

With this setup we obtain an ordered list of centerline points for each centerline, as desired.
The only setback at this point is that we may have a large number of small centerlines. In the next section we propose a way to connect them.

2.4.5 Centerline merging and handling of forkings

If two centerlines intersect at some point, it is clear that they belong to the same structure. At an intersection there are two possibilities

1. the two centerlines can be joined into one longer centerline
2. we have a branching, e.g. a bifurcation

The previous step of the algorithm marked all intersections between centerlines, so all we need to do is investigate all intersection points. For an intersection point we

1. find all the centerlines which have the intersection point in their centerline points lists
2. if only two centerlines are found
    - check if the intersection point is at an extremity of the centerline points list for each of the centerlines
    - if yes
        - to keep the ordered list of centerline points property, we need to connect the last point of the first list with the first point of the second list
        - thus, if needed, revert the order of the centerline points lists in the centerlines, so that the intersection point is the last point in the first list and the first point in the second
        - append the second list at the end of the first list
        - stop
3. if this point was reached, we have a branching at this intersection point
4. in this case, do not connect the centerlines, i.e. do nothing

We could obviously connect the centerlines at branching points, but we choose not to do this. If branchings become particularly troublesome to segment, one could store the branching points and use different segmentation approaches in their vicinity.

2.4.6 Complications

The sections of the carotid artery which are usually screened by a physician are the common carotid artery up to the bifurcation into the external and internal carotid arteries.

The common, the internal and the external carotid arteries are assumed to be tubular structures, which is not unreasonable. Unfortunately, the same cannot simply be assumed for the bifurcation area. The bifurcation area has cross-sections around the centerline with roughly elliptical shapes, where the major

axis increases in size starting from the end of the common carotid artery up to the point where the external and internal arteries are reached.
Thus, at this specific area, the tubular structure detection algorithms will yield lower vesselness values than for tubular structures, because the curvature measured in the direction of the ellipse‘s major axis will be small, since we chose to use an isotropic smoothing function to obtain the image derivatives. These values could be so low, that no points in the bifurcation area are eligible to be centerline points. This would result in a missing piece in our carotid skeleton. We would have the following setting:

- we have centerlines for the common carotid up to almost the bifurcation area
- we have centerlines for the external and internal carotid arteries past the bifurcation area
- the ending points of the centerlines are known, since each centerline has an ordered list of centerline points
- there is a route through the vessel lumen connecting the ending points closest to the bifurcation area of the common and of the external carotid arteries
- there is a route through the vessel lumen connecting the ending points closest to the bifurcation area of the common and of the internal carotid arteries

We cannot be certain where exactly the bifurcation region will be in the dataset, so we try to find all the points at centerline extremities which could reasonably be connected together. To find these points we propose a simple schema:
Set a threshold for the distance between the ending points, $dist_{threshold}$. Set a threshold for the minimal alignment expected of the centerline directions at the ending points, $align_{threshold}$.
Take all combinations of two centerline ending points, end_1 and end_2

1. check if the distance between them is small enough:

$$||end_1 - end_2|| < dist_{threshold}$$

2. check if the centerline directions of the points are approximately aligned:

$$| < v_1, v_2 > | > align_{threshold}$$

3. check if the shortest route connecting the points is aligned with the centerline directions

$$| < (end_1 - end_2), v_1 > | > align_{threshold}$$

4. if all conditions where satisfied, add this pair of points to a candidate list

Given the list of candidate pairs, there are a number of ways to proceed:

- stick to the algorithm proposed in the previous section but using a lower $fit_{threshold}$ and try to evolve a centerline from one of the ending points in the direction pointing to the other to try to connect the two points
- use a height ridge/valley following algorithm as the one proposed by Aylward *et al*, [3], using the information provided by the gradient of the smoothed image (instead of by the second partial derivatives like the vesselness algorithm and possibly using a different smoothing kernel) to try to connect the two points
- find the curve connecting the ending points, γ, which minimises the energy term

  $$\int_{\gamma} img(\mathbf{r}) d\mathbf{r}$$

  where $img(\cdot)$ is the US signal strength (grayscale value) of a dataset point. This way a route connecting points with low US signal strength should be found. If all points on the route have a low enough US signal strength, the route is considered acceptable.

To keep the algorithm simple and because we believe tubular structure detection algorithms are a good idea for this application, we choose to use the first option proposed.
After all pairs have been investigated, we take all new succesfully evolved centerlines and run the centerline merging step described in section 2.4.5 again.

2.5 Vessel wall segmentation algorithm

Chapter 3

Experimental results

??

3.1 Tubular structure detection

Bibliography

[1]

[2] Cemil Kirbas and Francis K.H. Quek (2002) A Review of Vessel Extraction Techniques and Algorithms. ACM Comput. Surv. 36 number 2, 2004, pages 81–121

[3] Stephen R. Aylward and Elizabeth Bullitt (2002) Initialization, Noise, Singularities, and Scale in Height Ridge Traversal for Tubular Object Centerline Extraction. IEEE TRANSACTIONS ON MEDICAL IMAGING, VOL. 21, NO. 2, FEBRUARY 2002

[4] J. Alison Noble and Djamal Boukerroui (2006) Ultrasound Image Segmentation: A Survey. IEEE TRANSACTIONS ON MEDICAL IMAGING, VOL. 25, NO. 8, AUGUST 2006

[5] Lindeberg, T. (1998) Feature detection with automatic scale selection'. Int. Journal of Computer Vision 30(2), 77–116

[6] David Lesage, Elsa D. Angelini, Isabelle Bloch, Gareth Funka-Lea (2009) A review of 3D vessel lumen segmentation techniques: Models, features and extraction schemes. Medical Image Analysis 13 (2009) 819-845

[7] Alejandro F. Frangi (2001) Three-Dimensional Model-Based Analysis of Vascular and Cardiac Images. PhD Thesis, ISBN 90-393-2647-9.

[8] Abir Zahalka and Aaron Fenster (2000) An automated segmentation method for three-dimensional carotid ultrasound images. Phys. Med. Biol. 46 (2001) 1321-1342

[9] Miller J. V., Breen D. E., Lorensen W. E., OBara R. M. and Wozny M. J. (1991) Geometrically deformed models: a method for extracting closed geometric models from volume data. Proc. Computer Graphics SIGGRAPH91 Conf. (Las Vegas, NV) vol 25, pp 217-26

[10] Fei Mao, Jeremy Gill, Donal Downey and Aaron Fenster (2000) Segmentation of Carotid Artery in Ultrasound Images. Proceedings of the 22nd Annual EMBS international Conference, July 23–28,2d00, Chicago IL.

[11] Chaoquan Chen *et al* (2004) Segmentation of Arterial Geometry from Ultrasound Images Using Balloon Models. Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on : page 1319

[12] A. Fenster, B. Chiu, A. Landry, J.D. Spence, G.E Parraga (2006) 3D Ultrasound System for Analysis of Carotid Plaque Progression and Regression. Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference on : pages 1966–1970

[13] Julian Guerrero *et al* (2007) Real-Time Vessel Segmentation and Tracking for Ultrasound Imaging Applications. IEEE TRANSACTIONS ON MEDICAL IMAGING, VOL. 26, NO. 8, AUGUST 2007 1079

[14] Lee J-S (1980) Digital image enhancement and noise filtering by use of local statistics. IEEE Trans Pattern Anal Mach Intell PAMI 2(2):165-168

[15] C.P. Loizou *et al* (2007) Snakes based segmentation of the common carotid artery intima media. Med Bio Eng Comput (2007) 45:35-49

[16] C.P. Loizou *et al* (2007) An Integrated System for the Segmentation of Atherosclerotic Carotid Plaque. IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE, VOL. 11, NO. 6, NOVEMBER 2007

[17] Amr R. Abdel-Dayem and Mahmoud R. El-Sakka (2005) Carotid Artery Ultrasound Image Segmentation Using Fuzzy Region Growing. International Conference on Image Analysis and Recognition, ICIAR'2005, LNCS 3656, pp. 869–878, Springer-Verlag Berlin Heidelberg, September 2005, Toronto, Canada

Appendix A

Hessian eigenvalue decomposition

Sbrubbles

Appendix B

New smoothing function

Appendix C

Calculus of variations - Extremizing functionals