Abstract

The goal of this paper is to introduce some tools developed in a code for fluid flow modeling in complex channels. Starting from 2D longitudinal images of bloodstreams, we reconstruct 3D vessels appropriate for flow simulation. Two mathematical tools are the key points for such a code, they are a level set function and a skeleton to describe the geometry. The integration of the geometry in the fluid mechanic code is simplified thanks to such a description of the geometry. This work leads to a stand-alone code capable to simulate 3D flows in a geometry reconstructed from an angiography.

1 Introduction

In the context of medical imagery, numerical 3D reconstruction is widely developed and is even integrated into the diagnostic process across already existing software. The 3D vessel reconstruction is a more specific application because of the branching problem which is also developed in the medical research area [1] [2] [3]. The aim of this article is to construct 3D bloodstreams from medical images but also to simulate flows in the reconstructed geometry. In this work, we are only interested in longitudinal view captures in order to focus on angiographies and select specific regions (for example a junction) where the vessels are nearly planar. Then, a 3D flow simulation in the reconstructed selected region is obtained and allows the analysis of mechanical stress on the fluid and on the geometry.

For this purpose, two mathematical tools are essential to describe the geometry, the skeleton of the geometry and the signed distance function to the boundary of the geometry. These tools allow a 3D modeling of the vessels but also a simple integration of the geometry into the fluid mechanic solver. The skeleton, also called centerline, is largely used in medical imagery. For example, Tozaki et al. [4] apply a 3D thinning algorithm to extract the vessel centerlines, which is used to classify the blood vessels. Kawata et al [5] use a graph description to extract the curvilinear centerlines in association with thresholding, elimination of small components and 3D fusion processes.

Section 2 is devoted to the description of these two mathematical objects and their interactions. The skeleton of vessels represents the centerline of vessels and gives access to a 2D description of the 2D views as well as to the 3D reconstruction of bloodstreams. The signed distance function to the boundary of the geometry is expressed thanks to the definition of the skeleton. This function is a level set function [6] whose zero level is the boundary of vessels and defines the geometry implicitly. Furthermore, this implicit representation of the geometry naturally fits into the fluid mechanic process. The fluid mechanics equations and numerical solver are also detailed in this second section.
Section 3 deals with successive steps which lead to the vessel reconstruction from the 2D image to the 3D geometry. Technical stages as contour detection, image enhancement and segmentation are first described. Techniques based on partial differential equations such as the Ginzburg-Landau equation and the Eikonal equation complete the scanline technique for contour detection and segmentation. The Eikonal equation is used again to define the skeleton which is the set of singular points of the solution of this equation. Details are given to construct the direction of the skeleton, which is useful for flow velocity information on the borders of images (inflow and outflow). Finally, we close this section with a concrete example of angiography detailing step by step the previous image process from the 2D view to the flow simulation.

2 Implicit Representation of Geometries

We work on images which represent 2D geometry and on their 3D modeling. In what follows, Ω will represent the implicit 3D geometry i.e. Ω denotes an open subset of $\mathbb{R}^3$ with boundary $\partial \Omega$.

The goal of this section is to describe the part of Ω included in a rectangular box $B$ from the data of a real function defined in the whole box. This function $\psi$ is negative inside Ω, positive outside,

$$\forall x \in \Omega, \quad \psi(x) < 0 \quad \text{and} \quad \forall x \notin \Omega, \quad \psi(x) \geq 0.$$  

This function vanishes on the boundary so that the zero level set of $\psi$ defines the boundary $\partial \Omega$. Such a function is classically called a level set function [6] and $\partial \Omega$ is the 0 level set function. We are only interested in the definition of the level set function restricted to the box $B$ corresponding to the extension of the 2D image of the geometry. The particularity of this work is to construct a 3D geometry with a single 2D view forcing assumptions on the geometry.

The 3D level set function will be constructed thanks to the skeleton of the 2D geometry.

2.1 Level Set Function and Skeleton

Starting from a 2D image of the geometry, the 3D domain Ω has to be defined by a level set function. The 2D view is assumed to be a longitudinal cut of a 3D geometrical object Ω. This object is assumed to be a set of channels centered on the longitudinal cut. This corresponds, for example, to channels engraved on a 2D face of a material or, locally, a junction of blood vessel. In such a case, the 2D view suffices to define the 3D geometry if furthermore we assume that we know the shape of engraving on normal cuts or we can assume, for blood vessel, circular shape of cuts of blood vessel. It is then useful to reduce the 2D view of channels to ”axes” of channels (1D channels) in order to define the 3D object by engraving it along the axes of channels. The mathematical tool to define such axes are skeleton. Mathematical details on skeleton are given in [7]. Hereafter, a brief introduction of such a tool is given.

**Definition 2.1.** Let $\Omega_d$ be an open bounded set of $\mathbb{R}^d$, the skeleton of $\Omega_d$ is defined as the smaller set $S$ (in sense of inclusion) such that

$$\Omega_d = \bigcup_{x \in S} B_d(x, r(x)),$$

with maximal radius $r(x)$ where $B_d(x, r)$ denotes the euclidian ball of $\mathbb{R}^d$. That is, $B_d(x, r(x))$ is the biggest ball centered in $x$ included in $\Omega$. 

2
Remark 1. If \( x \) belongs to the skeleton \( S \), then the ball \( B_d(x, r(x)) \) is tangent to \( \partial \Omega_d \) in two points at least. On Figure 1, the skeleton is drawn with a dashed line and few tangent balls to \( \partial \Omega_d \) centered on the skeleton are represented.

As a matter of fact, if it is not the case the ball \( B_d(x, r(x)) \) can be included in a bigger one included in \( \Omega_d \).

We are then able to link the data of the skeleton and associated radius to the level set function of the geometry:

**Definition 2.2.** The following function \( \psi \) is associated with the skeleton:

\[
\forall y \in \mathbb{R}^d, \quad \psi(y) = \inf_{x \in S} \{ \| x - y \| - r(x) \},
\]

where \( \| . \| \) denotes the Euclidean norm of \( \mathbb{R}^d \). The function \( \psi \) is called the signed distance level set function to \( \partial \Omega_d \).

As a matter of fact, the distance of a point \( y \in \Omega_d \) to \( \partial \Omega_d \) is given by

\[
\| y - Py \| = \inf_{z \in \partial \Omega_d} \| y - z \|
\]

with \( Py \in \partial \Omega \).

If \( Py = Px \) for \( x \in S \), then \( \| x - Px \| = r(x) \) and

\[
\| y - Py \| = \| x - Px \| - \| x - y \| = r(x) - \| x - y \|.
\]

The negative distance (opposite of the distance) is defined for points of \( \Omega_d \). In the same way, the positive distance is defined for points outside \( \Omega_d \).
Remark 2. By virtue of the previous remark, if \( x \) belongs to the skeleton \( S \), then the gradient of the signed distance level set function \( \psi \) is singular in \( x \) and if \( x \in \Omega_d \) does not belong to the skeleton \( S \), the function \( \psi \) is smooth in \( x \). Then, if the function \( \psi \) is known inside \( \Omega_d \), the skeleton \( S \) can be identified as the singular points of the function \( \nabla \psi \) (\( \nabla \) denotes the gradient operator). On Figure defsk, the level set function \( \psi \) is represented and singularities on \( \nabla \psi \) are visible on the dashed line.

The strategy to construct the 3D geometry is then the following

1. Define the 2D level set function of the longitudinal cut containing the 2D geometry \( \Omega_2 \)
2. Construct the Skeleton as the set of singular points of the gradient of the 2D level set function
3. Construct the 3D level set function on a box \( B \) of \( \mathbb{R}^3 \) by the formula (1) with \( d = 3 \)

The first step of the algorithm is based on image processing tools in order to identify the region \( \Omega_2 \), then the 2D level set distance function is obtained by solving the Eikonal equation. This part of the work is detailed in section 3.

The second step of the algorithm is straightforward but taking the analysis a bit far allows the directions of skeleton to be determined. This point will also be developed in the next section.

The last step of the algorithm can offer variants. Indeed the Euclidian norm \( \| . \| \) appearing in (1) can be modified to take into account different shapes of sections of channels. Namely, the 2D-norm restricted to the normal plane of the direction of the skeleton defines the shape of channel sections. The choice of the Euclidian norm leads to a circular channel section, whereas another norm leads to different shapes which depend themselves on the direction chosen in the basis of \( \mathbb{R}^3 \). For example, by turning the direction around the skeleton, an \( L^1 \) norm instead of the Euclidian norm leads to twisted channels. On Figure 3, a branch of the channels is constructed with the Euclidian norm, the other with the \( L^1 \) norm and one is twisted.

In the end, in this algorithm, the skeleton plays a major role, it allows a representation of the 2D geometry as well as the 3D geometry, modulo assumptions of the geometry.

The level set representation of the 3D geometry \( \Omega = \Omega_3 \) is then suitable for flow simulations in this geometry. The flow simulations computed in the box \( B \) and the domain \( \Omega \) appear in fluid mechanic equations across the level set function \( \psi \). This is detailed hereafter.

### 2.2 Fluid Flows in 3D Geometries

We are interested in incompressible Newtonian fluid flows. The Navier-Stokes equations govern the flow. In order to consider biffuid flows another level set function \( \phi \) is introduced to distinguish
Figure 3: Twisted geometry.

each phase. Furthermore, the surface tension forces arising at the interface of fluids are modeled and gravity or other external forces can be applied to the full domain.

We further assume that the flow is isothermal and fluids are homogeneous. Besides, densities and viscosities are constant within each phase of the fluid. The governing equations are then expressed by

\[
\rho \left( \frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} \right) - \nabla \cdot (\eta D \vec{u}) + \nabla p = \vec{F} \quad \forall (t, \vec{x}) \in \mathbb{R}^+ \times \Omega,
\]

together with the incompressibility condition:

\[
\nabla \cdot \vec{u} = 0 \quad \forall (t, \vec{x}) \in \mathbb{R}^+ \times \Omega,
\]

where the field \( \vec{u} = (u, v, w) \) is the velocity, \( p \) the pressure, \( \rho \) the density, \( \eta \) the viscosity, \( \vec{F} \) any body force detailed hereafter and \( D \vec{u} = (\nabla \vec{u} + \nabla^T \vec{u})/2 \). Moving interfaces can be handled with level set method - introduced by Osher and Sethian in [8] (see also [9] and [10]) - and we here use the approach of Sussman, Smereka and Osher [11] for incompressible two-phase flows. In addition, the interface between the two fluids is captured by advecting the level set function \( \phi \) (whose sign defines the phase of fluids) with the flow velocity \( \vec{u} \):

\[
\frac{\partial \phi}{\partial t} + \vec{u} \cdot \nabla \phi = 0 \quad \forall (t, \vec{x}) \in \mathbb{R}^+ \times \Omega.
\]

Note that densities and viscosities have to take the characteristic values of each phase depending on the sign of \( \phi \). As mentioned above, the forces occurring on the fluids are gravity and surface tension and are written:

\[
\vec{F}_\sigma = \rho \vec{g} + \sigma \kappa \delta(\phi) \vec{n},
\]

where \( \vec{g} \) is the gravity acceleration, \( \sigma \) is the surface tension coefficient, \( \vec{n} \) is the unit normal to the interface, \( \kappa \) is the mean curvature of the interface and \( \delta(\phi) \) is the Dirac measure localized on the fluid interface. This formulation of the surface tension has been used by Unverdi and Tryggvason [12] and Brackbill, Kothe and Zemach [13]. The unit normal to the interface is classically obtained via \( \phi \) by the formula:

\[
\vec{n} = \left. \frac{\nabla \phi}{|\nabla \phi|} \right|_{\phi=0}
\]

as well as the mean curvature of the interface:

\[
\kappa = \left. \nabla \cdot \left( \left. \frac{\nabla \phi}{|\nabla \phi|} \right|_{\phi=0} \right) \right|_{\phi=0}
\]
If we note $H$ the Heavyside function, a simple way to compute the Dirac measure is

$$\delta(\phi)\vec{n} = \nabla H(\phi),$$

which is approached thanks to a smooth regularization of the Heavy-side function.

Finally, these equations are extended to the box $B$ and flows are forced to arbitrary small values by penalization where $\psi$ is positive. The no-slip condition for fluids along boundaries of channels ($\partial\Omega$) is imposed. Boundary conditions have to be added for the velocity field at the inlets and outlets of channels on $\partial B$. For simplicity, we introduce Dirichlet boundary conditions:

$$\rho \left( \frac{\partial \vec{u}}{\partial t} + \vec{u}.\nabla \vec{u} \right) - \nabla.(2\eta D\vec{u}) + \frac{1}{\varepsilon} H(-\psi)\vec{u} + \nabla p = \rho \vec{g} + \sigma \kappa \nabla H(\phi), \quad \forall (t, \vec{x}) \in \mathbb{R}^+ \times B \quad (8)$$

$$\nabla \cdot \vec{u} = 0 \quad \forall (t, \vec{x}) \in \mathbb{R}^+ \times B$$

$$\vec{u} = 0 \text{ if } \psi \geq 0, \quad \forall (t, \vec{x}) \in \mathbb{R}^+ \times \partial B$$

$$\vec{u} = \vec{u}_b \text{ if } \psi < 0, \quad \forall (t, \vec{x}) \in \mathbb{R}^+ \times \partial B,$$

where $\varepsilon > 0$ the penalized parameter has to be chosen small in order to enforce the velocity to be small with $\varepsilon > 0$ outside $\Omega$. The forced flow $\vec{u}_b$ on boundaries of $B$ is defined thanks to the distance to the boundaries of channels across the value of $\psi$ modeling a Poiseille like flow. The scalar flow rate has to be specified by the user when processing the 2D image of the geometry. The construction of $\vec{u}_b$ is detailed in section 3.3 and uses the skeleton direction.

Then, a uniform cartesian grid is used to discretize the box $B$ and no meshes of the geometry $\Omega$ have to be developed. More precisely a staggered grid ”velocity pressure” is used. Classical tools of numerical analysis are used to choose the time scheme, viscous and penalized terms are implicated to solve the discrete approximation of (8) step by step in time. An augmented Lagrangian is used to take the incompressibility constraint into account.

Concerning the displacement of the interface, classical tools developed for the level set method are used: the level set function evolves in time by solving the transport equation (4). This transport equation is discretized in space with a Weno5 scheme [14] and redistanciation of the level set function $\phi$ is used when deformed. For microfluidics applications, where bifluid flows evolves with low Reynolds [15], a thinner grid for transport equation makes sense. A velocity interpolation on three times thinner grid is used.

We are now concerned with the imagery process and examples of flow simulation on the 3D reconstruction of an arterial route as well as bifluid flows driven by surface tension for microfluidic applications.

3 Results

We have developed a system providing models of bloodstreams from angiographies. First of all, we have to extract relevant information from images which can be degraded, mainly which lack contrast and are strongly noisy. This information represents the connected components of venous or arterial routes. To do this we must obtain a clean geometry, that is to say without artificial holes corresponding to a bad image interpretation. The field of contour detection methods is wide and includes techniques such as active contour models or snakes [16] and Mumford-Shah algorithm [17] [18]. In this work, a first step of image enhancement partially corrects the image, a second step is to overcome the remaining defects with PDE tools like the Ginzburg-Landau equation [19]. The goal is to be able to connect artificially interrupted interface. This technique
is a simple alternative to "inpainting" technique [20] based on order 4 PDE. From the connected components of this geometry, we construct its skeleton, because it allows us to derive a model, both in terms of geometrical and fluid mechanical informations (flow or pressure).

3.1 Image Enhancement

The image processing community has developed many enhancement algorithms, ranging from extremely simple crop dynamics to complex morphological operators. Most of these algorithms are intended to be used on general cases, which is not exactly our problem because our data is more structured.

Nevertheless, the first steps used to obtain a legible image are common. Increasing the contrast of an image can be done globally or locally, either from the histogram of the image (dynamic cropping for example) or taking into account the neighbors of each image pixel.

Moreover, we use some linear and non linear filter functions. For the linear part we use an unweighted average of neighbors:

$$I_f(x, y) = \frac{1}{k} \sum_{i=-n/2}^{n/2} H(i + n/2, j + n/2) \cdot I_i(x + i, y + j)$$ (9)

with $I_f$ the final image, $I_i$ the initial image, $H$ the filter and $n$ the filter size. This equation results in each image point by applying the discret convolution $I_f = H \otimes I_i$. The main quality of this filter is that it does not affect the overall intensity of the image. On the other side, we should note that edges of the image have to be considered separately: for a filter size $(n + 1) \times (n + 1)$ we have to add a frame thickness of size $n/2$.

For the non linear part, we mainly use the Tukey’s median filter (source of the Standard Median Filters or SMF), effective in an image made noisy by a pulse noise source, which is the case of medical imaging systems. The median filter behaves statistically as a lowpass filter. The algorithm in the discrete space, is simple: for each pixel in the image, it sorts its neighbors by increasing value and then we take the median value of pixels classified and assign it to the pixel. The pulse noise source dictate the choice of a cross-shaped mask, which creates less distortion while being more effective against isolated pixels than more conventional filters.

Of course, we can combine all of these tools, in any order, and use do/undo operations. All of these tools are designed to ease the work of the next step in the extraction of connected components.

When contrast increasing does not suffice, tools based on partial differential equations are used. The goal of the next subsection is to introduce a tool based on unsteady Ginzburg-Landau partial differential equations able to connect interrupted interface by thickening the interface and able to slim artificial defaults of interface.

3.2 Ginzburg-Landau PDE

Starting from initial data corresponding to the weakly contrasted image, we let a PDE evolve for positive time leading to a more and more contrasted solution. The contrasted values are considered to be 0 and 1. The transition from two states 0 and 1 is set to a length parameter $L$. The Ginzburg-Landau (G-L) model [19] was firstly introduce in semiconductor physics
and electro-magnetism. In a PDE formulation with time evolution, the unknown function $u$ depending on time $t$ and position $x$ solves

$$\partial_t u(t, x) - L^2 \Delta u(t, x) = -u(u - \frac{1}{2})(u - 1), \quad x \in \Omega, \; t \in \mathbb{R}^+, \quad (10)$$

with an additional Neuman boundary condition if the domain (corresponding to the image) is bounded. Here, the laplacian $\Delta$ is defined with respect to the variable $x$. When time goes to infinity, the solution $u$ of (10) converges to a stationary contrasted solution with two states 0 and 1 with a smooth transition whose size is of order $L$. If $L$ is close to the pixel size, the solution $u$ for a very long time corresponds to a trivial contrast process with a threshold to $\frac{1}{2}$: if $u_0(x) > \frac{1}{2}$ then $u(x) = 1$. A modified system of (10) adapted to image processing consists in introducing a variable threshold $0 < \theta < 1$ in the equations:

$$\partial_t u(t, x) - L^2 \Delta u(t, x) = -u(u - \theta)(u - 1), \quad x \in \Omega, \; t \in \mathbb{R}^+, \quad (11)$$

The interest is double: firstly the variable $\theta$ makes it possible to choose a threshold adapted to the image and secondly regions where $u < \theta$ expands along time if $\theta > \frac{1}{2}$. This phenomenon is well known on the Ginzburg-Landau model when studying traveling waves. Then, for image processing, a thin interface (possibly interrupted) of values bigger than $\theta < \frac{1}{2}$ can be thickened.

An example of a thin interface is given on the following example. Thanks to iterative resolutions of (11) with adapted thresholds, the interface is thickened and made continuous.

![Figure 4: An image with an interface geometry (doppler of a carotid)](image)

Weakly contrasted image and G-L applications:

On figure 5, the Ginzburg-Landau is iterated for fixed time evolution. The first application extracts an explicit region of flow, then a second iteration with a different $\theta$ is used to slim artificial obstacle regions. Note that to solve the Ginzburg-Landau model (11) an Euler explicit scheme is used in time and a centered space discretization approaches the Laplacian operator. The classical CFL condition of heat equations due to explicit discretization is verified to ensure a stable approximation. Because $L$ is chosen of order pixel size, the stability condition is not restrictive and the explicit discretization is prefered.
3.3 Segmentation

Image segmentation algorithms can be classified into two categories: edge-based and region-based segmentation. Both techniques have pros and cons. The edge detection has simple control structures and regular operators such as image convolution but can produce nonconnected boundaries, so it is necessary to postprocess the result to obtain the final segmentation. In our solution, based on the scan line algorithm, we maintain as a natural consequence the connectivity of the boundaries. On the other hand, most region-based techniques depend on the selection of initial regions, manage complex control structures and region boundaries are often distorted, so that a merge step is needed to provide the final segmentation [21]. However, with good image enhancements and iterative applications of the Ginzburg-Landau model, as we have seen in the previous section, we have solved most problems associated with this approach, in particular those associated with holes in the outline.

The region-based technique is used by starting from a given region selected by the user. This region grows by moving its boundary with unitary normal velocity as long as the region is inside the connex region to identify. We then solve the Eikonal equation:

\[
|\nabla T(x)| = \frac{1}{V(x)}, \quad x \in \Omega,
\]

\[
T(x) = 0, \quad x \in \Omega_0 \subset \Omega,
\]

where \(\Omega_0\) is the starting region and the variable \(T\) corresponds to the arrival time of the interface of the growing region. The normal velocity \(V\) is 1 where the image is white and 0 where the image is black. The identified connex region corresponds to points where the arrival-time is finite. The Eikonal equation is solved by the fast marching method introduced by James A. Sethian[10]. The information propagates outward from the interface and allows a local solver of the Eikonal equation. But the approximation is made efficient if the solution \(T\) is computed from low values to high values. The cost of the method relies on the fact that the minimal value of all predicted arrival times has to be computed. This minimal value is then consistent with the solution of the Eikonal equation. With adapted algorithms (heapsort, . . .), the resolution is obtained in a sufficiently short time for an image of 1 megapixel. Furthermore, several connected regions can be identified by starting with an adapted initial region.

For the edge detection, we explored another way to extract the connected components of these images by adjusting the scan line algorithm. The purpose of this method is to capture the geometry of the objects of the scene through the spatial and temporal consistency from the boundary of the object and from the scan line respectively.
Despite its apparent simplicity, the scan-line partitioning technique is quite robust against noise [22] [23] and it is well adapted for specific images from which the geometry is quite known, like angiographies.

In our application, first of all the user must define if the first pixel, the upper left point, is inside or outside the object. He must also define a strength threshold that corresponds to the value at which the scan line algorithm detects a boundary i.e. the passage from outside to inside or from inside to outside. Finally, he can choose the thickness of the border if appropriate. We maintain a data structure that helps the algorithm to solve special situations like borders which begin on the left edge of the image or nearly horizontal edges.

The experimental results are rather satisfactory but the method is extremely dependent on the choice of the edge strength threshold and creates false edges in the presence of high curvatures (the same conclusions are present in [22]). For all these reasons, for now we prefer to use the region-based technique that relies on the fast marching method, but we plan to merge the both tools in a near future.

3.4 Skeleton and flows

As detailed in section 3.3, the starting point to define the skeleton is to compute the distance function to the boundary of the region. Once again, we have to solve the Eikonal equation:

\[
|\nabla \psi(x)| = 1, \quad x \in \Omega
\]
\[
\psi(x) = 0, \quad x \in \partial \Omega \subset \Omega,
\]

This equation is solved again with the Fast Marching Method. The two-dimensional distance function \( \psi \) to the boundary is then known and the singular points of \( \nabla \psi \) have to be extracted. The criteria we have chosen determine the angles of multivalued \( \nabla \psi \) in the following way: the discrete gradients are evaluated with a decentered scheme and inner products are compared.

More precisely, \( \partial^+_x, \partial^-_x, \partial^+_y \) and \( \partial^-_y \) denotes the discrete decentered gradients:

\[
\partial^+_x u(x,y) = (u(x+h,y) - u(x,y))/h
\]
\[
\partial^-_x u(x,y) = (u(x,y) - u(x-h,y))/h.
\]

The whole inner products of different normalized decentered gradients are six in number and are evaluated to determine if the gradient is multivalued. The minimal inner product, corresponding to the most opposite directions, is compared to a threshold value. Under a threshold value of 0.4, the set of singular gradients constitutes a sufficient skeleton on most geometries. The skeleton is enriched for an upper threshold value.
More information can be extracted from the image. The direction of the 2D skeleton can be computed and will play a role to define a velocity field of reference. The two normalized directions \( (d_1 \text{ and } d_2) \) of decentered gradients corresponding to the minimal inner product are the two extrema directions of the multivalued gradient. These directions look at two different points of the 2D geometry wall. The bisectrix of the directions \( d_1 \) and \( d_2 \) is adapted to define a skeleton direction. The bisectrix is carried by \( d_1 + d_2 \) or \( d_1^\perp - d_2^\perp \) when \( d_1 + d_2 \) degenerates (opposite directions).

In order to determine the boundary condition, the user chooses a flow rate (or a pressure) where the geometry meets the image boundary. In the case of the Dirichlet boundary condition as defined in (8), the 3D velocity field \( \vec{u}_b \) has to be constructed at least on boundaries. In fact, we define a velocity field of reference in the neighborhood of vessels near boundaries. The velocity is based on the flow rate informed by the user, the skeleton and the skeleton direction. If \( \vec{d} \) is the inward normalized direction of the skeleton, \( f_r \) is the inflow rate associated with a vessel and \( r_s \) is the maximal radius of the skeleton in this region, then the velocity field of reference writes:

\[
\vec{u}_b = \frac{2 f_r \psi}{\pi r_s^2} (\psi + 2r_s)\vec{d}
\]

Note that in this formula, the skeleton direction at a point is the one corresponding to the nearest point of the skeleton. A more complicated formula of the velocity of reference takes into account a velocity direction which depends on the skeleton direction and the gradient of the distance function to the geometry \( (\nabla \psi) \). This direction is in the plane defined by \( \vec{d} \) and \( \nabla \psi \). The velocity direction is close to \( \vec{d} \) near the skeleton and close to \( \nabla \psi^\perp \) near the geometry. This is adapted to tubular geometry whose radius is far from constant. At last, the velocity field \( \vec{u}_b \) on the boundary is slightly modified by a factor out the outlet in order to verify the exact discrete flow equilibria on boundaries of \( B \) related to the incompressible constraint (8).2.

### 3.5 Applications

Our approach is validated by a program, at each stage of the methods presented in this article. Roughly speaking, the different numerical solvers are written in Fortran while the graphical interface and the visualisation engine are written in C++. For the latter, we rely on two libraries: wxWidget for the GUI and image processing and OpenGL or VTK for the 3D visualisation.

First of all, after loading the image and selecting a region of interest, we apply different filters to obtain a result which will allow our algorithms to work best. We recall that these tools are known but we specify that the user can use them in any order and at multiple times. He can also undo the last filter that has been applied.
After a step we call "binarization", i.e. the image rendering is in black and white, we select a seed region (the circle in the screen capture below) and apply our fast marching method to find the connected region of interest. The advantages of this method are numerous, because it allows us to extract a connected region and naturally eliminate regions of the image that are not relevant.

![Circular seed on left picture and expended area on right.](image1)

We then construct the skeleton of the selected arterial scan. For this purpose, we need to build the matrix of the distances, that is to say, label every point of the domain (every pixel of the image) with a number indicating its distance to the edge of the implicit geometry (negative inside, positive outside). Then we normalize these values to obtain a range of luminance. From this 2D level set function, we construct the skeleton reduced to points inside the geometry and separated by more than 3 pixels from the boundary of the geometry. From this, we build the 3D level set function.

![Signed distance function on left picture, Skeleton on right picture.](image2)

At this stage, we can define the inflow and the outflow (or boundary pressure) for each boundary section. As detailed in section 3.4, the 3D geometry is enriched with boundary conditions. In the following views the 3D vessel is introduced with the associated velocity field of reference near the boundary. This velocity field of reference is constructed with the flow data given by selecting the region by means of the graphical interface (see Figure 7).

The solutions of the flow with such a boundary condition are solved by the code detailed in section 2.2 and produce a complete velocity field in the carotid given in the example (see Figure 8).
4 Conclusion

With this work, we designed a user friendly code in order to extract geometries from medical imagery. These images were obtained, for example, from angiographies and dopplers. This code permits us to reconstruct the 3D geometries of blood vessels and to simulate fluid flows in them.

The whole software is developed from scratch with personal libraries, which allow perennial developments, adaptative solutions and fast scalability. This software enables the extaction of
pertinent data for flow simulation, such as velocity information. The 3D code for flow simulation is also developed with personal libraries allowing a strong scalability again.

Such work is made possible by the choice of simple and pertinent tools, in particular the skeleton and the level set functions defining the geometry. It is important to note that the introduction of such tools avoids meshing the flow domain and simplifies the flow mesh to a simple cartesian grid.

In addition, the study of the interaction between blood and vessels which are of great interest are facilitated by the Level Set representation of the geometry. Elastic models of geometries in fluid structures are well developed in the literature [24], [25], [26], [27].

The analysis of flow simulation in blood vessels provides a wide field of applications to this work. For further developments on blood vessel reconstruction, we are interested in 3D reconstruction from multiple transverse cuts. Such images are obtained from scanners and force us to connect the blood vessels from one cross-section to the next. Algorithms based on mass transfer problems are adapted for such a purpose in order to construct the skeleton and the associated radius of the geometry. Then, the geometry, constructed by level set methods, allows the use of the same fluid mechanic code. Nevertheless, improvements on this code are necessary, namely the development of adaptive mesh refinement techniques. Besides, the imagery libraries developed in this work will be a fundamental set of tools for the image processing of cross-sections.

Acknowledgement: This work has been supported by French National Research Agency (ANR) through COSINUS program (project CARPEINTER ANR-08-COSI-002).

References


