SPECIAL ISSUE - REVIEW

An image-based modeling framework for patient-specific computational hemodynamics

**Luca Antiga · Marina Piccinelli · Lorenzo Botti ·
Bogdan Ene-Iordache · Andrea Remuzzi ·
David A. Steinman**

Abstract We present a modeling framework designed for patient-specific computational hemodynamics to be performed in the context of large-scale studies. The framework takes advantage of the integration of image processing, geometric analysis and mesh generation techniques, with an accent on full automation and high-level interaction. Image segmentation is performed using implicit deformable models taking advantage of a novel approach for selective initialization of vascular branches, as well as of a strategy for the segmentation of small vessels. A robust definition of centerlines provides objective geometric criteria for the automation of surface editing and mesh generation. The framework is available as part of an open-source effort, the Vascular Modeling Toolkit, a first step towards the sharing of tools and data which will be necessary for computational hemodynamics to play a role in evidence-based medicine.

L. Antiga (✉) · M. Piccinelli · L. Botti · B. Ene-Iordache ·
A. Remuzzi
Biomedical Engineering Department,
Mario Negri Institute for Pharmacological Research,
Villa Camozzi, Ranica, BG, Italy
e-mail: antiga@marionegri.it

L. Botti · A. Remuzzi
Industrial Engineering Department,
University of Bergamo, Bergamo, Italy

D. A. Steinman
Biomedical Simulation Laboratory,
Department of Mechanical and Industrial Engineering,
University of Toronto, Toronto, Canada

1 Introduction

Patient-specific computational hemodynamics appeared in the late 1990s with the work of a few researchers bridging image processing and computational fluid dynamics (CFD) [17, 19, 22, 30] and it is today pursued by an increasing number of groups around the world. The spread is motivated both by the growing interest in the study of hemodynamic variables and how they relate to disease processes, and by the improvements in the available computing resources, which now allow simulations to be run anywhere from personal computers to parallel clusters.

Despite the amount of work in this field has been growing steadily over the last years, the studies in which computational hemodynamic simulations are performed on large patient populations are still very limited. The ability of performing large-scale studies in a controlled way is a key requirement for the future of computational hemodynamics. In an evidence-based medicine perspective, it is only through the evaluation of the ability of some hemodynamic variables to predict disease or influence treatment in large populations that those hemodynamic variables, or some surrogate thereof, will become players in clinical research and clinical practice [18]. Without this necessary step, the usefulness of computational hemodynamics, although sought in principle, will remain unproven.

The likely cause of the paucity of studies on large patient populations is the fact that the process from image acquisition to simulation results is still based on several, operator-dependent steps, which involve image-segmentation, model

preparation, mesh generation and simulation. The lack of integrated workflows, with a few exceptions [10], makes the use of image-based computational hemodynamics for large-scale studies non trivial and encumbered by time-consuming operator-dependent tasks, which constitute a potential source of error whose influence is difficult to control and quantify. In this context, the availability of shared tools is in our opinion of primary importance. Open source software has proven itself in the last years to be a powerful resource in several fields of computing. In the scientific world, open source efforts have often lead to widespread adoption, standardization of methods, continuous quality checking, open code review, reproducibility of results and benchmarking. For this reason, we believe that the introduction of shared tools in the field of patient specific computational hemodynamics will lead to better reproducibility and comparability among studies, ultimately increasing the chance of generating results that contribute to clinical evidence.

To take a concrete step in the direction outlined above, we here present a framework aimed at providing a working solution for high-throughput computational hemodynamics. The techniques presented here are only a part of a larger effort that our groups have been undertaking in the past years towards the construction of shared and robust open source tools for geometric and hemodynamic vascular modeling (the Vascular Modeling Toolkit [2]). In this work, we will focus on the process that leads from imaging datasets to simulations. We will first present the general approach and the software tools on which our framework is based, and we will then proceed to describe the key components of the framework in some detail.

2 General approach and tools

Image-based modeling for computational hemodynamics in general requires three fundamental steps, namely image segmentation, model pre-processing and mesh generation. Our effort consists in the development of an integrated framework that allows these operations to be performed in a streamlined fashion. Decisions on the design of the framework were taken towards robustness and repeatability on one hand, and efficiency on the other. In more detail, we aimed at satisfying the following criteria:

- minimizing operator intensive tasks, while allowing the framework to retain the flexibility required for the application in diverse contexts;
- favoring high-level decisions, e.g., the specification of vascular segments through selection of fiducials at their endpoints followed by automated segmentation, over operator dependence, e.g., manual outlining or manual threshold-based segmentation criteria;

- reducing the number of free parameters to be specified at each step, so that the operator is faced with a few decisions that are easy to document.

From a software standpoint, the techniques presented in this paper are integrated within the Vascular Modeling Toolkit. The toolkit has been developed using a modular approach, by building specialized components capable to interact with each other to create complex processing pipelines. From an implementation standpoint, we followed a two-layer approach, in which computation-intensive algorithms are implemented in the C++ programming language, and algorithms are dynamically assembled to form processing pipelines using the Python programming language. Both layers are implemented using an object-oriented approach, at the advantage of modularity and maintainability.

The Vascular Modeling Toolkit relies on two major open-source efforts in the field of scientific visualization and medical image analysis. The Visualization Toolkit (VTK) [23] is an object-oriented toolkit for analysis and visualization of 3D data implemented in C++, whose classes are also accessible from three higher-level languages, namely Tcl, Python and Java. VTK provides data structures for images (vtkImageData), polygonal surfaces (vtkPolyData) and volumetric meshes (vtkUnstructuredGrid), among several others. From an algorithmic standpoint, VTK provides basic image processing tools, several algorithms for surface processing, such as smoothing, decimation, clipping, connectivity filters, and algorithms acting on volumetric meshes, as well as advanced visualization resources. The Insight Toolkit (ITK) [15] is a C++ toolkit for image processing, segmentation and registration, although it is growing beyond these boundaries. It is developed using a generic programming paradigm, with heavy use of C++ template constructs, which allows high modularity and code reuse with minimal impact on performance. Thanks to this approach, most algorithms can work unmodified for any spatial dimension and for multiple data types. Over the last years, ITK has become a standard resource in medical image analysis. State of the art algorithms are being contributed by the community and follow strict quality checking rules based on open code review and continuous testing.

The approach followed in the design of the Vascular Modeling Toolkit is to adopt the data structures provided by VTK, and to implement C++ algorithms in classes derived from base VTK algorithm classes. This way, new algorithms can coexist with native VTK algorithms in a processing pipeline. Using automated wrapping tools available in VTK, all algorithms are made available to the high-level layer implemented in the Python programming language. The same is true for ITK processing pipelines,

which are directly integrated within VTK derived classes and made available to the Python layer. The Python layer is in turn composed of classes that also act as individual command line scripts. These scripts, each implementing a specific functionality, can be piped at the command line to provide complete processing pipelines. This modular architecture allows users and developers to take advantage of the Vascular Modeling Toolkit at several levels, from the C++ algorithm to Python code to command line pipelines. This tries to answer the challenges posed by image-based modeling, which often requires operations to be assembled in a custom fashion depending with the imaging data and modeling problem at hand.

In the following sections, we will explore in detail our approach for image-based computational hemodynamics. Where not stated otherwise, the techniques have been implemented as components of the vascular modeling toolkit.

3 Image segmentation

Magnetic resonance (MR), three dimensional ultrasound (3D-US), computed tomography (CT) and rotational angiography (RA) scanners today available in clinical settings are capable of providing information about the anatomy of blood vessels with sub-millimetric resolutions. Such information is represented as a 3D array of grayscale intensities, which we will refer to as $I(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^3$ being the spatial coordinate, and it is usually provided by clinical scanners in DICOM format, which we directly use for importing data. Following this phase, image segmentation is performed, in order to extract the geometry of the vascular segments of interest which will later define the computational domain.

Image segmentation is a key aspect for computational hemodynamics, as it has been suggested that local geometry has a key influence on hemodynamic variables [31]. Rather than aiming at the automatic identification of entire vascular networks contained in the image [13], we leave the identification of the individual vascular segments of interest for a simulation to the operator through the selection of a few seed points on the image, and we then concentrate on the objective operator-independent determination of the location of the lumen boundary. This is justified by the fact that, in a typical computational hemodynamic problem, the need is to simulate flow in a subset of the vessels contained in the image, whose boundary has to be located in a reproducible way. In fact, more automated segmentation methods must later rely on model editing to prepare geometries for mesh generation [10, 13]. Our solution is to concentrate the effort required by the operator in the interpretation of the image content,

and rather to automate all the downstream operations as much as possible.

The segmentation approach adopted in our framework is based on implicit deformable models [9, 24], which have the desirable property of being flexible with respect to the topology of the object to be segmented. The following main components have to be defined in order to specify the segmentation strategy:

1. the partial differential equation (PDE) describing the evolution of the implicit deformable model;
2. the initialization strategy, that define the initial conditions for the PDE;
3. eventual regularization strategies to adopt in case of noisy images;
4. the choice of the *feature image*, i.e., the input to the forcing terms of the PDE.

A description on how each component has been implemented in our framework follows.

3.1 Implicit deformable models

We refer to implicit deformable models as to deformable surfaces $S(t) : \mathbb{R}^2 \times \mathbb{R}^+ \to \mathbb{R}^3$ described by means of their 3D embedding $\phi(\mathbf{x}, t) : \mathbb{R}^3 \times \mathbb{R}^+ \to \mathbb{R}$, a scalar function defined such that $S(t)$ is its zero-level isosurface, i.e., $S(t) = \{\mathbf{x} | \phi(\mathbf{x}, t) = 0\}$. The technique used to track the temporal evolution of the embedding under image- and shape-based forces is known as *level sets method*. Following this method, the evolution of $\phi(\mathbf{x}, t)$ can be described by a PDE of the kind

$$\phi_t = -w_1 G(\mathbf{x})|\nabla\phi| + w_2 2H(\mathbf{x})|\nabla\phi| - w_3 \nabla P(\mathbf{x}) \cdot \nabla\phi \tag{1}$$

where the first term on the right-hand side represents surface inflation with a position-dependent speed given by $G(\mathbf{x})$, which can depend on image features or on shape constraints, the second term represents a smoothness constraint on the surface, $H(\mathbf{x}) = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$ being the mean curvature of the zero-level surface, and the last term represents advection of the surface by means of the vector field $\nabla P(\mathbf{x})$. Weights w_1, w_2 and w_3 regulate the influence of each term on surface evolution [24]. Equation 1 is solved using finite differences on the image grid, as briefly described in Sect. 3.5.

Choosing $P(\mathbf{x}) = |\nabla I|$ in Eq. 1 results in the advection of the surface towards the ridges of the image gradient magnitude, as shown in Fig. 1. This choice has the underlying assumption that the interface between two tissue types corresponds to the location of maximum change in image intensity. This is a reasonable assumption for acquisition modalities such as CTA, RA, contrast-enhanced and steady-state MRA (e.g., TrueFISP), in which the
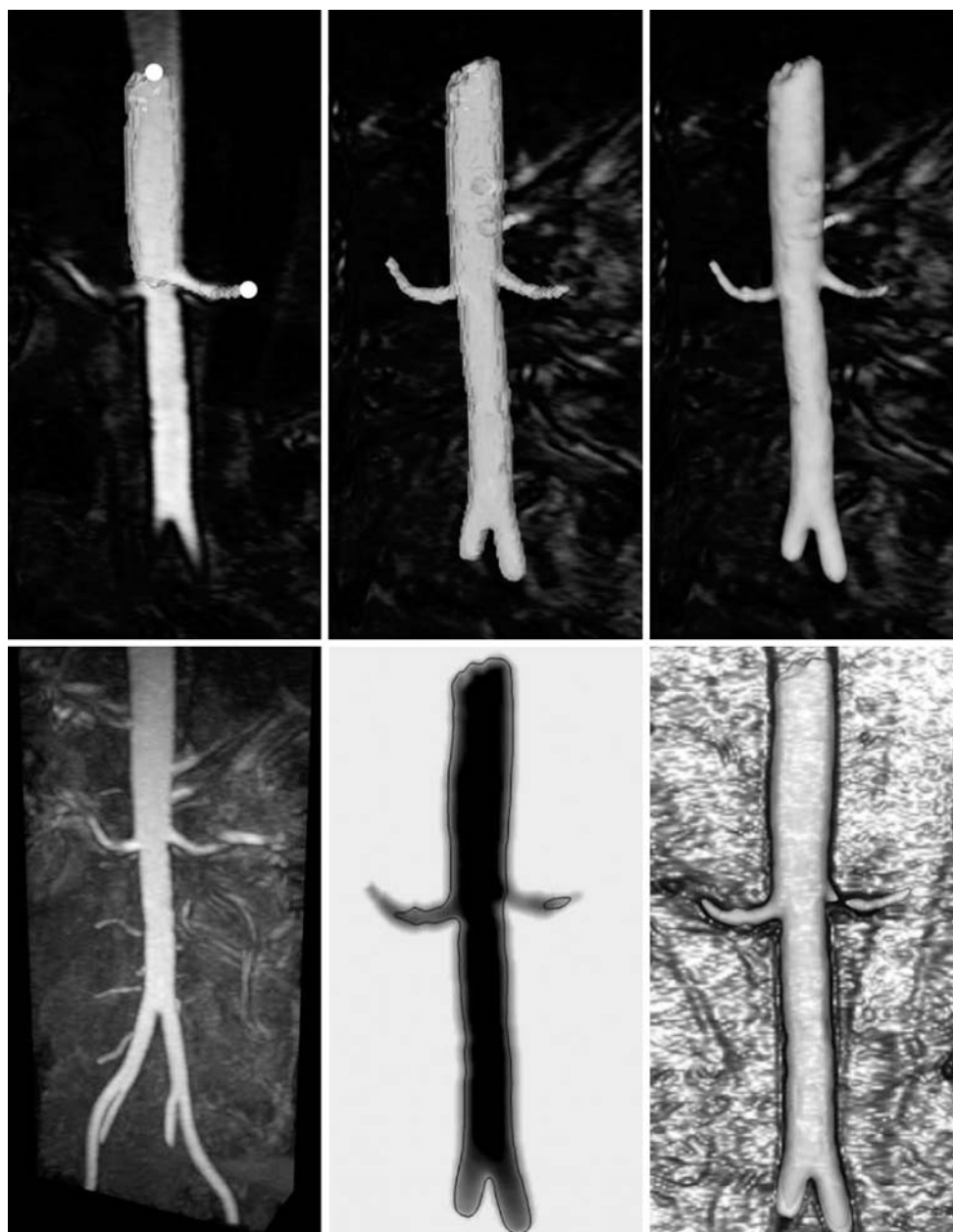
Fig. 1 Segmentation of an abdominal aorta MRA dataset. *Top row, left to right* interactive initialization of a branch through the specification of two seed points, one within the suprarenal aorta and the other within the right renal artery; complete initialization; result of segmentation after level set evolution. *Bottom row, left to right* maximum intensity projection of the dataset; a section of the final level set function ϕ with its zero level set; final model superimposed to a section of $|\nabla I|$ (darker intensity associated to higher values)

intensity of blood is given by the presence of a contrast agent or to T2/T1 weighting of blood. On the contrary, such advection field may not yield satisfactory results for images acquired using time-of-flight (TOF) or phase-contrast (PC) MRA sequences, in which signal from blood is linked to its velocity. In this latter case, the decrease in velocity in the vicinity of the vessel wall may lead to a decrease in image intensity which is not linked to the presence of an interface between blood and wall at that location. This effect affects PC–MRA images more

severely than TOF–MRA images, to the point that dedicated advection fields should be developed for the former, although this is not subject of the present work.

Relying on spatial variation in image intensity rather than on absolute intensity, e.g., in manual or automated thresholding approaches, confers greater robustness to the segmentation method, since the location of the lumen boundary is made independent to changes in the intensity of the lumen due to artifacts, contrast bolus dynamics or partial volume effects. An additional disadvantage of

intensity-based approaches is that, since the true lumen boundary corresponds to regions of steep changes in intensity, slight variations in the definition of a threshold lead to relatively large displacements of the location of the resulting lumen boundary. On the contrary, seeking the location corresponding to the ridges of the gradient magnitude of image intensity is a univocally defined criterion, although the problem is then shifted to capturing the ridges corresponding to the lumen boundary among all feature ridges contained in the image, which will be subject of the next Section.

In [5, 6], we proposed the use of Eq. 1 to segment vascular segments for computational hemodynamics. In particular, we described a three-stage procedure composed by centerline-based initialization, inflation ($w_1 = 1$, $w_2 = w_3 = 0$) towards the vessel edges and advection ($w_1 = 0$, $w_2 \geq 0$, $w_3 = 1$) to the gradient magnitude ridges. Although essentially robust, this approach suffered from operator-dependence induced by the inflation step, which was meant to bring the surface from the centerline towards the boundary of the vessel, within the capture radius of the advection term. In this work, we overcome this problem by skipping the inflation step thanks to a better initialization strategy, with which we directly generate an initial surface located within the capture radius of the advection term, as shown in Fig. 1 and described in the next Section. The first term in Eq. 1 is therefore unused in the present method, to the advantage of reproducibility.

3.2 Colliding fronts initialization

The generation of an initial level set function $\phi_0 = \phi(\mathbf{x}, t_0)$ plays an important role in our framework. As stated above, our strategy is to let the operator interactively select the segments of interest and use deformable models to reproducibly identify the sub-voxel position of the lumen boundary. In this perspective, the goal of initialization is to produce the initial surface of selected branches already within the capture radius of the advection term in Eq. 1, with minimal input and at interactive speeds. Avoiding the inflation step during the evolution of the level set equation while retaining the ability of segmenting selected vascular segments goes in the direction of conferring robustness and flexibility to the modeling chain.

We here present a novel technique for level set initialization that satisfy the above requirements, here referred to as *colliding fronts*. The technique is based on the propagation of two independent wavefronts from two seeds, $\mathbf{s}_1$ and $\mathbf{s}_2$, interactively placed at the two ends of a vascular branch. The wavefronts travel with speed proportional to local image intensity, following the so-called eikonal equation [25]

$$|\nabla T| = I^{-1} \tag{2}$$

where T is the arrival time of the wavefront in each point of the domain and the right-hand side is the *slowness* of the wave in each point, as shown in Fig. 2. Equation 2 is customarily solved on the imaging grid by means of the fast marching method, which is an efficient method for the numerical approximation of the eikonal equation based on upwind finite differences [25]. Once the arrival times T_1 and T_2 have been computed from $\mathbf{s}_1$ and $\mathbf{s}_2$, the initial level set function can be defined as

$$\phi_0 = \nabla T_1 \cdot \nabla T_2 \tag{3}$$

Such function is negative where the two fronts are directed towards each other, which identifies the initial level set as the region between the two seed points. Eventual side branches are automatically excluded from the initialization, since the two fronts seep in side branches in the same direction from the parent vessel, and the function in Eq. 3 becomes positive, as shown in Fig. 2. In order to increase robustness, the initial volume is defined as the region for which ϕ_0 in Eq. 3 is negative and as the same time connected to the seeds, i.e., if there is a connected path between $\mathbf{s}_1$ and $\mathbf{s}_2$ such that ϕ_0 is always negative.

For high contrast between vessels and background, Eq. 3 already produces an initial surface located in close proximity of the boundary of the lumen. In fact, since the wavefronts propagate with speed proportional to image intensity, the lumen boundary is characterized by a steep decrease in speed, which makes both wavefronts align parallel to the direction of the boundary, thus leading ϕ_0 to be positive, as shown in Fig. 2. When the signal-to-background ratio is not high, the slow-down of fronts at the lumen boundary might not be sufficient for the fronts to assume a concurrent orientation. In this case, the selection of a threshold to constrain the propagation of the wavefronts outside a prescribed intensity range is required. It has to be noted that, provided that the threshold is specified in the range between the vessel intensity and the background, the final position of the surface after the level set evolution will not be affected by the choice of the threshold.

The use of wave propagation through the fast marching method opens up another simpler possibility for interactive initialization, to be used in case of objects of non tubular shape or for initializing entire vascular networks without branch selection. The approach is to select a group of seeds and a group of targets on the image, and to solve Eq. 2 from the seed points. The initial level set function is then $\Phi_0 = T - t_{\text{target}}$, where T is the solution of the eikonal equation, and t_{target} is the lowest eikonal solution among the targets. In other words, Φ_0 is the wavefront at the time when the first target has been reached. In particular for the initialization of large vascular segments, this approach

Fig. 2 Colliding fronts initialization along the abdominal aorta dataset. *Top, left* interactive selection of two seed points, one within the suprarenal aorta, on at the iliac bifurcation. *Top, right* contour lines of the solution of the eikonal equation T_1 from the suprarenal seed, and vectors showing the solution gradient ∇T_1, color coded according to the magnitude of the superior-inferior direction component (*red* for vectors pointing up, *blue* for vectors pointing down). *Bottom, left* initialized abdominal aorta, showing the selective initialization of the branch and the absence of side branches. *Bottom right* close-up of the two vector fields ∇T_1 and ∇T_2, showing counter-directed vectors within the lumen and co-directed vectors at the lumen boundary and within the renal artery; the black line is the zero level set of $\nabla T_1 \cdot \nabla T_2$, i.e., the trace of the initialized surface
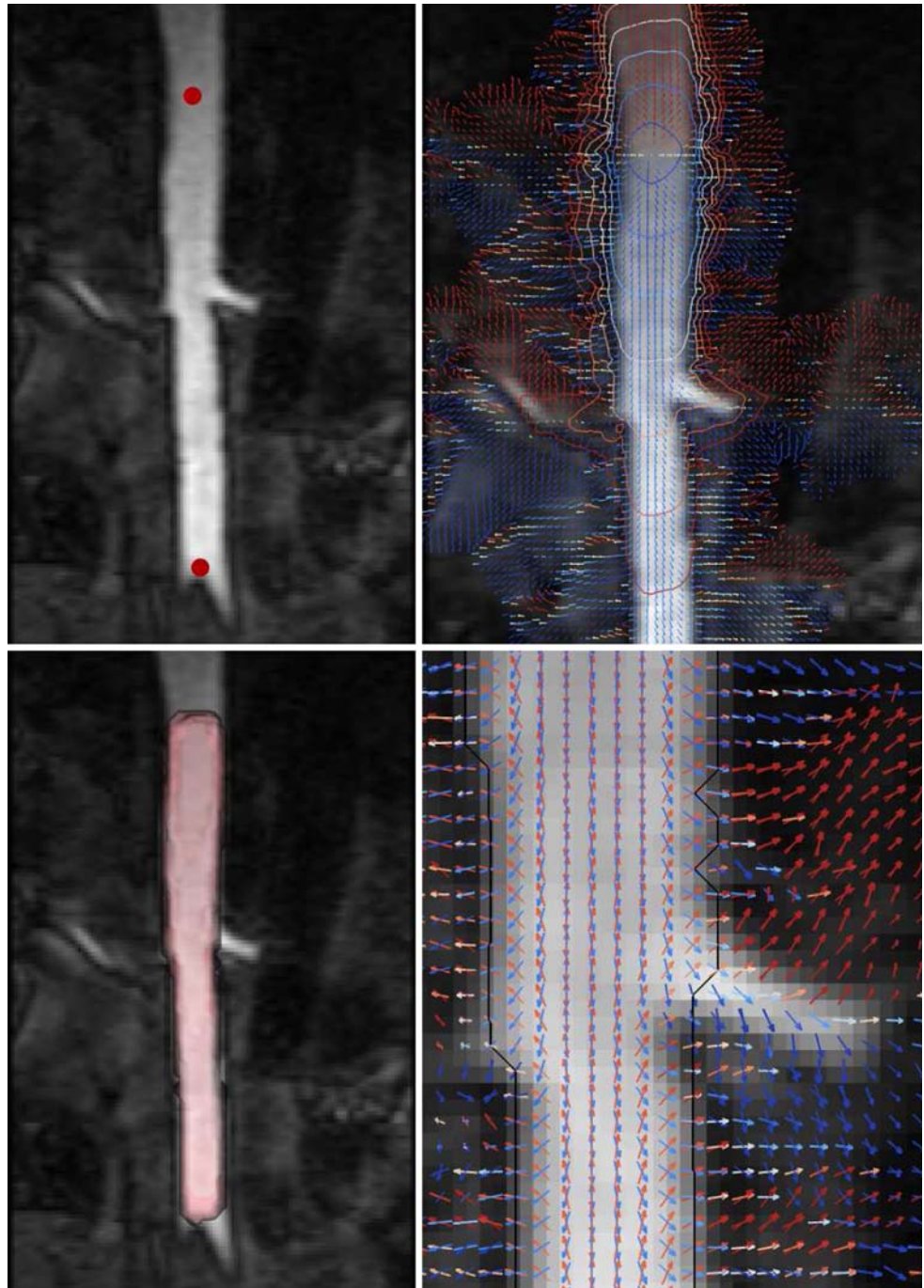
necessitates the specification of a threshold to constrain the propagation of the wavefront.

3.3 Regularization strategies

Although for datasets with good signal-to-background and signal-to-noise ratio the above initialization strategy and sole level set advection ($w_1 = w_2 = 0$, $w_3 = 1$) are sufficient, for datasets affected by noise or artifacts the introduction of regularization in the segmentation process

is required to produce a smooth segmentation. Clearly, regularization depends on the choice of at least one smoothing parameter, which potentially introduces operator-dependence.

In order to decrease the influence of noise in the segmentation process, noise can be suppressed using a Gaussian or other image smoothing filter. Alternatively, derivatives in the computation of the feature image can be performed by convolution of the image with a Gaussian derivative kernel, in lieu of finite differences. This is

equivalent to computing derivatives using finite differences on a subsampled version of the image after convolution with a Gaussian kernel. These two approaches can effectively suppress the influence of noise, but can also lead to corrupted segmentations, for instance by merging nearby vessels, or erasing small vessels. This is due to the fact that a Gaussian filter suppresses image features below the scale corresponding to its standard deviation.

An alternative strategy for regularization is the use of the second term in Eq. 1 ($w_2 > 0$), which penalizes high curvature features in the resulting segmentation. This can be a very effective strategy when the advection term is strong, i.e., when the image contrast to background ratio is high. On the contrary, it can lead to sever shrinkage of the surface where the curvature term is stronger than the advection term. It is in fact known from differential geometry that any genus zero surface tends to a sphere if allowed to deform under mean curvature flow.

In order to obtain surface regularization while retaining vessel diameter, Lorigo et al. [21] have proposed to replace the mean curvature H appearing in the second term of Eq. 1 with the smallest principal curvature, computed as the smallest eigenvalue of the curvature tensor. In this case, regularization only results in the penalization of surface features with high minimal curvature. Considering that a vessel is a tubular object with low smallest principal curvature (zero for a straight vessel) and high largest principal curvature (the inverse of the radius of the vessel), the curvature term in this case acts by regularizing a vessel in the longitudinal direction, without shrinkage of the vessel diameter. Note that this method can fail to provide effective regularization or can generate unexpected artifacts in case of non-tubular shapes.

As a last regularization strategy, we briefly mention surface smoothing at a post-segmentation stage, e.g., through the use of a Taubin type non-shrinking smoothing filter [29] on the final triangulated surface. Since the filter is non-shrinking and preserves topology, it can be safely employed, although it can affect the shape of surfaces at a bifurcation apex or similar saddle-shaped features.

As usual, the perfect regularization strategy does not exist. The choice has to take into account the type of image, geometry and quantities of interest for the simulation. An important point to address in the execution of a large-scale study is the adoption of a consistent set of regularization parameters to use throughout the study, and the evaluation of sensitivity of results to the chosen parameters.

3.4 Feature images

As stated above, the computation of the feature image $|\nabla I|$ to be used as the advection field is critical to the present approach. The advection term expresses a vector field pushing the deformable surface from the two sides of each feature ridge. The way the feature image is computed determines the location where the deformable surface converges. In the previous section, we have already mentioned the use of convolution with Gaussian derivatives as a way of computing regularized derivatives as compared to finite differences. We also pointed out how regularization can lead to the loss of small vessels. In the remainder of this section we will address the problem of segmenting vessels with a diameter of a few pixels.

When the size of a vessel is comparable to the width of the point-spread function of the acquisition system, the shape of the intensity profile tends to be similar to the point-spread function itself. This means that the intensity profile of a small vessel tends to lose the plateau within the lumen and to assume a hat-like shape, with a single maximum within the lumen. This effect leads to underestimation of the diameter of small vessels, as the gradient magnitude is non-zero within the lumen. As a further problem, the availability of a few grid points across the lumen tends to degrade the quality of the approximation of partial derivatives computed using finite differences in the imaging grid. Since level sets are not topologically constrained, severe underestimation can lead to disappearance of entire tracts of vasculature.

The usefulness of image interpolation in improving the results of segmentation in these cases is usually moderate, since, for low resolutions, the effects of the point spread function of the acquisition system cannot be recovered from pure interpolation. In general an increase in sampling density can lead to a better approximation of partial derivatives and potentially avoid level set collapse in some cases, but it cannot compensate the effect of the point spread function on the signal. In the same direction, the prescription of anisotropic resolutions during acquisition aimed at achieving higher in-plane resolution at the expense of a lower resolution on the orthogonal direction is seldom a solution in these cases. The effect of the point spread function in the coarse direction can severely distort the shape of a vessel, and the apparent gain in in-plane detail actually derives from a partial-volume averaging in the out-of-plane direction, which can lead to suppression of vessels smaller than the slice thickness and further corrupt the shape of the remaining vessels.

In order to cope with the effect of low resolution and avoid collapse of vascular tracts, we here propose to approximate gradient magnitude using an upwind finite difference scheme during the computation of the feature image [4]. In the non-limiting assumption that vessels are brighter than the surrounding tissue, we compute image gradient magnitude $|\nabla I| \approx \|D_x I, D_y I, D_z I\|$ using the following approximation for partial derivatives

$$D_x I = \max\left(-\frac{I_{i+1} - I_i}{h_x}, \frac{I_i - I_{i-1}}{h_x}\right) \qquad (4)$$

where i is a voxel index in the x direction and h_x is the voxel spacing in the x direction (similarly for y and z). More intuitively, Eq. 4 expresses the computation of derivatives from a voxel only *towards* neighboring voxels having a lower intensity (if available), with the effect of reducing underestimation and avoiding level set collapse in cases where the vessel is represented by less than a few pixels across, as is the case for the renal arteries in Fig. 1. On the other hand, when voxel spacing is small compared to vessel diameter, the upwind approximation is close to the one provided by centered finite differences.

Although the above technique is effective in compensating the underestimation obtained for low resolutions, the accuracy of a CFD analysis on a model obtained from a low resolution scan is a question that should be addressed on a problem-specific basis. Certainly, the lower the resolution, the coarsest the level of detail that a CFD analysis would be able to assess, but a more specific characterization of this relationship depends on the domain of interest and on the problem being addressed. For instance, a straight tube necessitates of less geometric information than a complex saccular aneurysm for a CFD analysis to be considered accurate, and, similarly, accurate estimation of a pressure drop along a vascular segment necessitates of less geometric detail than accurate estimation of a wall shear stress map. Validation studies on physical or virtual phantoms on one hand, and patient studies based on repeated and multi-modality scans on the other, have to be encouraged for specific flow domains and quantities of interest.

3.5 Implementation notes

In our framework, we adopted the level set implementation provided by the Insight Toolkit. In particular, we make use of the sparse field level sets approach [32], by virtue of which only a minimum amount of voxel layers around the zero level set are tracked during the evolution. This way, computation of the solution on the whole imaging grid at each time step can be avoided, and the computing time depends on the actual extent of the zero level set surface. Originally implemented for isotropic voxel spacings ($h_x = h_y = h_z$), we extended the implementation to handle imaging data with anisotropic voxels, and contributed the code back to the Insight Toolkit. Similarly, we contributed the implementation of the colliding fronts initialization algorithm.

Once segmentation has been performed, the zero level set surface can be extracted using the Marching Cubes algorithm, as provided by the Visualization Toolkit. In the following, we will assume that a triangulated surface of the segmented vascular network is available.

4 Centerline computation

In order to streamline the process from segmented surfaces to simulation, it is required that a robust geometric characterization methodology is in place, so that operations can be performed robustly and with minimal or no user intervention. In our framework, geometric analysis is based on centerlines, which in turn rely on the concept of medial axis and its discrete counterpart, the Voronoi diagram [11]. The medial axis is defined as the locus of centers of spheres maximally inscribed inside an object, a sphere being maximally inscribed if not contained in a any other inscribed sphere. The envelope of all maximal inscribed spheres is the boundary surface of the object itself, therefore the (oriented) boundary surface and its medial axis are dual representations of the same shape [1]. In 3D, the medial axis is a non-manifold surface, i.e., not everywhere locally homomorphic to a 2D disk. A possible approximation to the medial axis for a discrete surface is the so-called embedded Voronoi diagram, that is defined by the portions of the Voronoi diagram lying inside the surface. The Voronoi diagram is defined by the boundaries of the regions of $\mathbb{R}^3$ closer to each surface point. As such, each point on the embedded Voronoi diagram is associated with a sphere touching the surface at at least four points and not inscribed in any other sphere, the discrete equivalent to a maximal inscribed sphere. Like the medial axis, the embedded Voronoi diagram is a non-manifold surface.

We define centerlines as minimal action paths on top of the Voronoi diagram, where the action field is the inverse of the radius of maximal inscribed spheres [6]. This method links two locations on the Voronoi diagram with a path minimizing the line integral of the action path, which results in centerlines that maximize their minimal distance to the boundary surface. Operatively, centerlines are computed by first solving the eikonal equation from a seed point on the Voronoi diagram domain

$$|\nabla T| = R(\mathbf{u})^{-1} \qquad (5)$$

where $R(\mathbf{u})$ is the maximal inscribed sphere radius field defined on the Voronoi diagram, $\mathbf{u}$ is its parametric space, and T is the solution representing the arrival times of a wavefront traveling on the Voronoi diagram with speed equal to $R(\mathbf{u})$, as shown in Fig. 3. Once T has been computed, a centerline is generated by following the path of steepest descent of T on the Voronoi diagram from each target point, that is, by solving the following trajectory equation
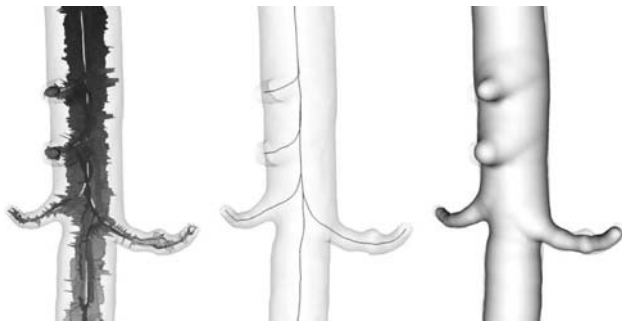
Fig. 3 *Left* embedded Voronoi diagram for the abdominal aorta dataset, grayscale-coded with the solution of the eikonal equation T seeded at the top-most section; centerlines traced from the downstream sections are shown superimposed to the Voronoi diagram. *Center* resulting centerlines. *Right* envelope of maximal inscribed spheres along the computed centerlines

$$\frac{d\mathbf{c}}{ds} = -\nabla T \qquad (6)$$

where s is the centerline parametric space taken as its curvilinear abscissa. Being defined on the Voronoi diagram, each centerline point is associated with the corresponding maximal inscribed sphere radius. The envelope of maximal inscribed spheres along centerlines produces a *canal surface*, or tube, which is inscribed inside the vessel and tangent to it at discrete locations, as shown in Fig. 3. Since they are associated to the deeper Voronoi structures, such defined centerlines are not sensitive to small displacements in surface points, which are instead known to affect the peripheral portions of the Voronoi diagram. Furthermore, the centerline computation problem is well-posed even in regions where the vascular domain doesn't resemble a cylinder, such as at bifurcations or inside aneurysms, and this confers robustness to the algorithm.

5 Model pre-processing

The polygonal surface resulting from segmentation is in general not directly usable for generating a computational mesh. Typically, inlet and outlet sections of the model are not flat and no boundary indicators are available for prescription of boundary conditions. Furthermore, in case fully developed Dirichlet boundary conditions have to be imposed at inlet or outlet boundaries, circular sections need to be generated in order for Poiseuille or Womersley velocity profiles to be specified analytically. Last, it is desirable to impose boundary conditions a few diameters upstream or downstream the region of interest for the flow to develop before it enters the computational domain of interest. These problems are overcome through the automated connection of cylindrical flow extensions to all inlets

and outlet sections of the model. Such sections can be generated by interactively cutting the original surface or by using centerlines to define the cutting planes in a more automated fashion. In any case, the direction of flow extensions must be robust with respect to the actual orientation of the cutting planes, and must rather reflect the orientation of the vessel as it approaches the boundary sections. For this purpose, we employ the orientation of centerlines within the last diameter before the boundary sections to define the normal to the flow extensions.

The shape of flow extensions must provide a smooth transition between the model boundary sections, in general non circular, and circular sections of equivalent area, in order to avoid artifactual flow disturbances in the transition between the flow extensions and the flow domain. We achieve this by generating a template cylindrical flow extension and then warping its first half by means of a thin plate spline transformation, computed in such a way to bring the cylinder circular section onto the model boundary section. The thin plate spline algorithm [8], aims at finding the deformation induced by the transformation between two sets of corresponding landmarks that minimizes bending energy. This is achieved through the use of radial basis function interpolation with a kernel of the form $R^2 \log R$ corresponding to the minimization of the bending energy of a steel plate subject to boundary constraints [8]. In our framework, we use the implementation of the thin plate spline algorithm provided within the VTK library. Results of the automated application of flow extensions is shown in Figs. 4 and 5.

By adjusting the strategy with which landmarks are prescribed for the thin plate spline algorithm, this methodology could be easily extended to the need of flow extensions with a curved axis, needed in the case where a vessel presents a strong curvature near the end and the flow domain of interest is close to the boundary section. However, it is desirable, that a sufficient segment of the upstream geometry is available for segmentation, in order to reproduce the flow patterns entering the domain of interest in a patient-specific way.

6 Mesh generation

6.1 Surface remeshing

Once the shape of the model has been finalized, a computational mesh must be generated for the numerical solution of the partial differential equations describing blood flow. Our approach for automated mesh generation is to generate a high quality triangulated surface mesh and to then fill the volume using a combination of tetrahedral and prismatic elements.

Fig. 4 *Left* model after the automated addition of flow extensions, color-coded with the distance to centerlines. *Center* result of surface remeshing using distance to centerline as sizing function and $R_{el} = 0.4$. *Right* cutout of the volume mesh, showing element size grading that closely follows surface element grading and adaptive 2-element prismatic boundary layer
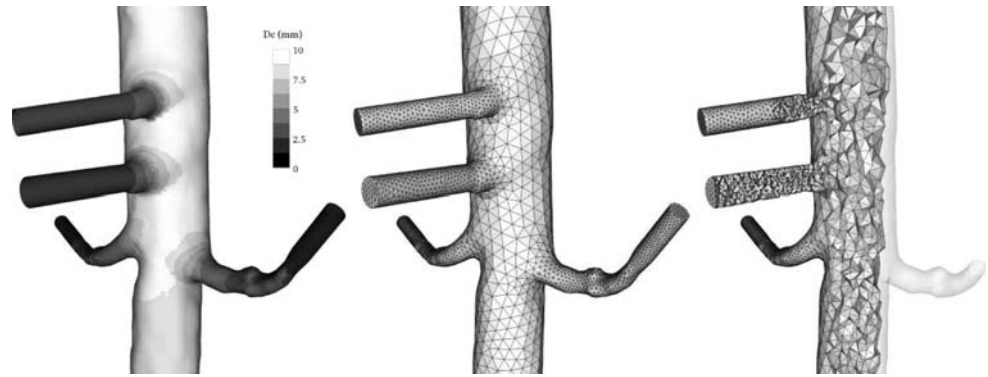

Fig. 5 Image-based model and mesh of a cerebral aneurysm located in the distal internal carotid artery, acquired from RA. *Left* sizing function based on distance to centerlines D_c. *Center* sizing function based on centerline maximum inscribed sphere radius R_c. *Right* mesh resulting from the use of R_c as sizing function, demonstrating the capability of the technique of handling non-tubular shapes
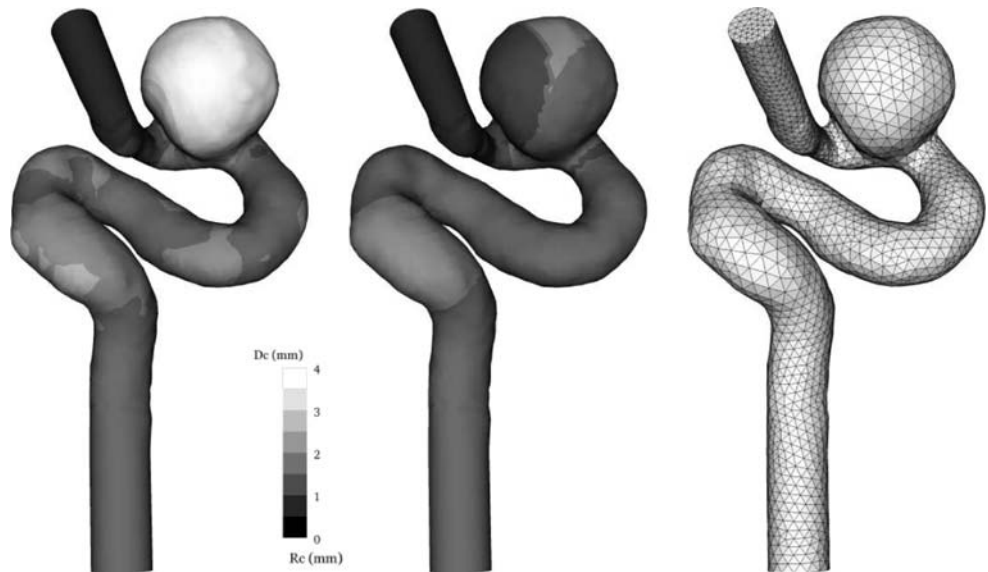
For the present work, we adopted a mesh improvement approach to surface remeshing, where we assume that the input surface mesh is a good polygonal approximation of the continuous geometry. In fact, this may not be true for a surface generated from level sets in case the geometric features are on the order of a voxel in size, since those features end up being linearly interpolated with large triangles. In this case, surface mesh density can be first increased using a smooth subdivision scheme. In particular, we employ Loop's subdivision scheme [20], provided by VTK, which yields a C2 continuous surface when mesh density tends to infinity, under assumptions of regularity of the starting mesh. Taking the input surface mesh as the reference geometry has the advantage of avoiding the conversion of the input triangulation to a CAD-like parametric representation, which could in principle introduce a regularization of the input geometry which is not directly controllable from the user side.

Our algorithm for surface remeshing belongs to the class of explicit remeshing algorithms, in which an initial triangulation $\mathcal{T}_0$ is progressively modified in order for the surface mesh to conform to imposed quality and sizing criteria [14, 28]. Mesh improvement is performed through three basic topological operations, namely edge collapse, edge split and edge flip, acting on surface triangles, and one geometric operation, point relocation, acting on point position. Target triangle area is locally defined as a sizing function $A(\mathbf{u})$ on the nodes of $\mathcal{T}_0$, where $\mathbf{u}$ is a point in the parameterization of $\mathcal{T}_0$. The four atomic operations are iteratively applied to the mesh $\mathcal{T}$ until the following criteria are met:

- the Frobenius aspect ratio of each triangle, defined as $f = \frac{e_1^2 + e_2^2 + e_3^2}{4\sqrt{3}A}$ where e_i is the length of triangle edge i and A is triangle area, is as high as possible and higher than a threshold (by default set to 1.2);
- local triangle area $A(\mathbf{u})$ conforms the sizing function specified on $\mathcal{T}_0$;
- triangle neighborhoods have a valence (i.e., the number of vertices connected to a vertex through a triangle edge) as close as possible to 6.

Whenever a new point is inserted in $\mathcal{T}$ or a point relocation is performed, the point is projected back on $\mathcal{T}_0$ by means of a fast octree locator. The same locator is used

whenever a value of $A(\mathbf{u})$ defined on $\mathscr{T}_0$ is needed for a triangle on $\mathscr{T}$. The sizing function $A(\mathbf{u})$ can be automatically defined on the basis of the geometry of $\mathscr{T}_0$, for instance by setting it proportional to the surface mean curvature. This solution has the advantage of refining the mesh around small-scale features, but suffers from sensitivity to the presence of small-scale noise on the surface, which potentially leads to artifactual over-refinements. We instead choose to define $A(\mathbf{u})$ on the basis of local vessel size, with the rationale that a vessel should have the same number of elements around, or across, the lumen, irrespective of its local diameter.

We adopt the distance of surface points to centerlines, $D_c(\mathbf{u})$, as a robust surrogate of local vessel size. For each surface point, we compute its distance to the centerline point that yields the minimum tube function value among all centerline points. This tube-based approach is superior to simply computing the distance to the closest point on centerlines, since the latter approach would tend to underestimate local size where the size of vessels change abruptly, such as in the case of stenoses or T-shaped bifurcations. Finding the centerline point that yields the minimal tube function can be performed efficiently by considering that a tube $\mathbf{c}_T$ is a line in a 4-dimensional space $\mathbf{c_T}(s) = (x(s), y(s), z(s), ir(s))$, where s is the line parameter, x, y and z the spatial coordinates of the centerline, r its radius and i the imaginary unit [3]. The tube function evaluated at a point $\mathbf{p} = (x_p, y_p, z_p, 0)$ is then

$$f(\mathbf{p}) = \min_{s \in [0,L]} \{\|\mathbf{c}_T(s) - \mathbf{p}\|\} \tag{7}$$

For a piecewise linear representation, Eq. 7 has to be evaluated for every linear segment $[s_{i+1}, s_i]$ to find $\tau \in [0,1]$ such that the tube function is minimal at $s = s_i + \tau(s_{i+1} - s_i)$. This is done analytically by extending the expression for the closest point to a line in Euclidean geometry to the 4-dimensional geometry of the tube function

$$\tau = \frac{(\mathbf{p} - \mathbf{c}_T(s_i)) \cdot (\mathbf{c}_T(s_{i+1}) - \mathbf{c}_T(s_i))}{\|\mathbf{c}_T(s_{i+1}) - \mathbf{c}_T(s_i)\|} \tag{8}$$

Once the distance to centerlines has been defined at every node on $\mathscr{T}_0$, it can be manipulated and used as a sizing function for the surface remeshing algorithm. In our framework, we compute the local target area field as

$$A(\mathbf{u}) = \frac{\sqrt{3}}{4} * (R_{\mathrm{el}} * D_c(\mathbf{u}))^2 \tag{9}$$

where R_{el} is a constant factor expressing the ratio between the nominal edge length and the distance to centerlines, as shown in Fig. 4. In practice, R_{el} is the only free parameter in the surface remeshing procedure, and it quantifies the length of the nominal triangle edge relative to the vessel

size. Since it's a relative quantity, it can be chosen once for all in a population study.

The above approach is robust but suffers from the presence structures that are not well represented by centerlines, such as saccular aneurysms, on whose surface $D_c(\mathbf{u})$ could assume artifactually high values. A simple workaround is to set a upper threshold to the admissible values of $D_c(\mathbf{u})$. However, this introduces an absolute mesh sizing parameter which may imply model-specific decisions, and impede automatic meshing in a population study. An alternative, more robust, approach is to replace $D_c(\mathbf{u})$ with $R_c(\mathbf{u})$, i.e., the radius of the maximal inscribed sphere at the centerline point with the minimal tube function. In other words, after finding the point with the minimal tube value, we get its inscribed sphere radius instead of computing the distance to it. The definition of $A(\mathbf{u})$ is then the same as in Eq. 9 with $D_c(\mathbf{u})$ replaced by $R_c(\mathbf{u})$. This way, the target area doesn't increase in those regions that deviate from a tubular shape, such as saccular aneurysms as shown in Fig. 5, but it instead depends on the size of the tube where those regions originate.

6.2 Volume meshing

Volume mesh generation is performed on the basis of the surface mesh produced at the previous step, as the latter is left unchanged and the interior is filled with volumetric elements. The mesh is composed of two parts, the tetrahedral interior and the prismatic or tetrahedral boundary layer. We will now present them in this order.

Tetrahedral mesh generation in the Vascular Modeling Toolkit is performed by incorporating the functionality offered by the Tetgen mesh generator [27], a tool for constrained Delaunay and high-quality tetrahedralization of piecewise linear complexes. The algorithms provided by Tetgen are provided in code that was directly embedded within a VTK class, which provides seamless integration with the rest of the framework and achieves complete automation. We refer to [26] for in-depth description of the theory and algorithms behind the tool. For our purposes, it is sufficient to mention that Tetgen is capable of generating meshes conforming to a boundary triangulation and satisfying the Delaunay criterion (i.e., no vertex falls inside the circumsphere of any tetrahedron). In addition, mesh quality is ensured through the specification of a bound on the target ratio of the radius of the element circumsphere to the length of the shortest element edge.

Sizing of the tetrahedral mesh can be provided through the specification of target tetrahedron edge size at a set of points in space. In our case, we express target volume at each node of the input surface mesh, setting it proportional to the local triangle edge length on the surface mesh, as shown in Fig. 4. The proportionality constant can be

chosen once close to 1. In practice, dealing with tubular geometries, and in general with concave bounding surfaces with non-negligible curvature everywhere, it is advisable to keep the proportionality constant lower than one (e.g., 0.8), in order to penalize the generation of elongated tetrahedra at the boundary surface.

Since the behavior of the solution close to the wall is of great interest for hemodynamics, as it is directly linked to wall shear stress and derived quantities, it is in general desirable that the mesh has a high element density near the wall, and that elements are aligned with the local orientation of the boundary surface in that region. In practice, this is realized by generating one or more layers of prismatic elements having the boundary triangles as their base and protruding into the domain along the surface normal direction (for solvers not supporting prismatic element types, the boundary layer can then be tetrahedralized). The thickness of the boundary layer is expressed as a fraction of the local edge length, which is in turn dependent on the local vessel size, consistently with the fact that the size of the physical shear layer is expected to scale with the vessel size, as shown in Fig. 4.

The concrete advantage of the generation of such boundary layer depends on the specific problem at hand and on the density of the internal tetrahedral mesh. When using an adaptive mesh refinement strategy as presented in the next section, it is expected that resolving the boundary layer in the direction normal to the surface will limit the number of elements tagged for refinement within the shear layer, therefore increasing efficiency. A systematic evaluation of such effect is beyond the scope of this paper and will be subject of future work.

As detailed in the next section, our solution strategy is based on P2-P1 finite elements. For this task we adopt isoparametric quadratic finite elements, namely 10-noded quadratic tetrahedra and 18-noded quadratic prisms. The conversion of the elements from linear to quadratic is performed by introducing additional nodes using a regular subdivision scheme. Once subdivided, midside nodes of boundary triangles are instead projected onto the initial surface $\mathcal{T}_0$ using the octree locator, so to obtain a proper high-order representation of the surface.

7 CFD simulation

The methodology for automated mesh generation described in the previous Section has one potential limitation, the fact that the computational mesh is generated on the basis of the geometry, and not on the basis of the expected flow features. As an example, in the case of a stenotic vessel, the mesh would be dense at the stenosis throat, but the post-stenotic sections would likely be under-resolved with respect to the high-speed jet emanating from the stenosis. This would potentially limit the application of the present workflow for large-scale simulations, as each case should in theory be individually evaluated for potential problems linked to geometry-driven mesh generation. In order to overcome this limitation, we stress the importance of the adoption of an adaptive mesh refinement approach for the numerical approximation of the differential problem, in this case the incompressible Navier–Stokes equations. The automated geometry-driven mesh generation approach described in the previous Section is then complemented by a flow feature-driven mesh refinement strategy, therefore providing a robust approach to high-throughput computational hemodynamics. A solver should therefore fulfill the following requirements:

- to provide a simple and robust solution strategy, so to reduce the number of parameters to be chosen for the simulation and to reduce the need for monitoring during computation;
- to allow adaptive mesh refinement and coarsening at any timestep, so that the mesh can adapt to the flow features arising throughout the cardiac cycle;
- to exploit parallel computation, both for the computation on clusters and on now standard multi-core workstations.

While it is beyond the scope of this paper to present a CFD solver, we briefly summarize the features of the solver currently under development in our group. Based on a second-order velocity correction scheme in rotational form, the implementation relies on the libMesh library [16], which provides a finite elements infrastructure featuring parallel adaptive non-conformal mesh refinement and coarsening. Non-conformal mesh refinement, i.e., the possibility to refine an element without refining its neighbors thanks to the correct handling of hanging nodes, allows element aspect ratios to remain unchanged throughout the simulation, and makes it efficient to perform refinement and coarsening frequently during the cardiac cycle. Also, we argue that the adoption of prismatic boundary layers and adaptive mesh refinement overcomes the theoretical presence of a numerical boundary layer close to non-slip boundaries, whose thickness is a function of the local element size. As mentioned above, the solver will be presented in a separate publication and will made freely available as open source software.

8 Conclusions

In this work, we presented the building blocks of our framework for high-throughput computational

hemodynamics, which is already freely available within the open-source Vascular Modeling Toolkit. We hope that the advent of these and other tools will enable studies to be performed on a large-scale and will allow to lift the effort required in generating patient-specific models and to shift the focus on the underlying biological problems.

Although our framework potentially allows patient-specific hemodynamic variables to be evaluated in large-scale studies, new approaches for post-processing of hemodynamic simulations need to be developed for the identification of geometric and hemodynamic criteria to be employed in clinical research [3]. This is, in our opinion, one of the pressing issues in this field, and as such will constitute a focus of our future work.

Appendix: Experimental validation on a patient-specific carotid phantom

Phantom manufacturing

In order to validate the image-based modeling framework presented in this work, and more specifically the accuracy of the segmentation process, we performed a phantom experiment on a physical replica of the lumen of a human carotid bifurcation, generated from CT-based patient-specific model using stereo-lithographic rapid prototyping (RP) technology. Briefly, CTA images of a normal carotid bifurcation were acquired during routine examination of a patient with suspect stenosis at the contra-lateral side. Images were acquired on a multi-slice scanner (Light-Speed QX/i, GE, Fairfield, CT) with in-plane resolution of 0.25 mm and slice thickness of 1.25 mm. The luminal surface of the carotid bifurcation, depicted in Fig. 6 was obtained using the method described in Sect. 3. The model presented diameters of 7, 4.8 and 3.6 mm for the common, internal and external carotid arteries, each evaluated approximately 3 diameters away from the bifurcation.

The surface was exported in stereo-lithography interface format (STL) and transferred to a RP stereo-lithography system (SLA-3500, 3D Systems, CA, smallest feature size 0.30 mm, typical layer thickness 0.15 mm) for the generation of a physical model consisting of a hexahedron of $30 \times 30 \times 64$ mm in size made of transparent epoxidic resin containing a hollow representation of the vessel lumen, as shown in Fig. 6. Further details on the generation of RP phantom are reported in [31].

Phantom imaging

After being filled with a solution of saline and iodinated contrast agent, the carotid phantom was first imaged with the same CT scanner and contrast-enhanced CTA protocol used for the patient's acquisition. Successively, a Gd-enhanced MR acquisition was performed (Gyroscan NT 1.5T, Philips, Best, the Netherlands) during which two contrast-enhanced MRA series were acquired, one in axial orientation, one in coronal orientation, both of them with acquired in-plane resolution of 0.78 mm and slice thickness of 2 mm, interpolated by zero-padding in k-space to 0.39 mm in-plane and 1 mm slice thickness. The scanning parameters were chosen to closely follow clinical protocols at the scanning site, and the phantom was placed in the scanners so to mimic the patient's neck position and orientation. All imaging tests were performed at the Radiology Unit of the Ospedale San Paolo, Milan, Italy. Images were downloaded in DICOM format for further processing.

To obtain exact dimensions of the vessel lumen, the phantom was milled on a numerically driven milling machine using 0.5 mm thickness slices for a total length of 60 mm. Each section was digitally photographed using a fixed frame holding the camera after each cut. The acquired images were stored on a PC at a resolution of $2,800 \times 1,700$ pixels. Pixel coordinates of 60 points for each lumen profile were manually identified on each section using a general purpose image processing software (NIH Image, v 4.02). In-plane spatial coordinates for the points were then obtained through a scaling factor provided by a scale grid that was photographed together with the sections. The measurement points were then arranged in a 3D stack by locating each planar section according to the position of the milling machine. The signed distance between the measurement points and the original STL model used as input for the RP machine averaged -0.09 ± 0.06 mm (mean $\pm$ standard deviation), and ranged from -0.49 to 0.24 mm (positive values corresponding to measurement points located outside the STL model).

Validation

Three-dimensional (numerical) models of the carotid phantom were generated from the CT images and from the two axial and coronal MR series using the modeling framework presented in this paper, setting $w_1 = w_2 = 0$ and $w_3 = 1$ in Eq. 1. Two sets of segmentations were produced, one employing the customary centered finite difference approximation of the gradient magnitude for the computation of the advection field, and one using the upwind finite difference approximation presented in Eq. 4. No regularization strategy was adopted in either case.

Fig. 6 *Left* patient-specific model of a carotid bifurcation used as input for the RP system (see Appendix). *Right* picture of the physical phantom

Fig. 7 Models obtained from CT, MR axial and MR coronal acquisitions of the physical RP phantom (see Appendix). No smoothing strategies have been adopted during segmentation. Signed distance to the STL model used as input for the RP system is displayed. Positive values indicate that the segmented model is located outside the STL model

All surfaces resulting from segmentation were aligned onto the STL surface using the iterative closest point (ICP) algorithm [32]. Once registered, the signed shortest distance of each point of the segmented models to the STL surface was computed, taken positive for points located outside the STL model. In order to factor out the effect of RP manufacturing from the evaluation of accuracy, the signed shortest distance of each point of the segmented models to the measurement points was computed.

Results are summarized in Table 1, in which mean, standard deviation and range of the signed distance of each segmented model from the ground truth, taken in turn as both the surface given input to RP system (STL) and the physically measured points on the phantom (PTS). The models and their signed distance to the STL surface are depicted in Fig. 7.

The values show that the segmentation technique in general produces surfaces that are located in close vicinity to ground truth. The average distance is below the resolution of the acquisition system, and with the minimum and maximum errors ranging within the value of in-plane resolution. From the comparison among the three scanning protocols, CT showed to allow for the highest accuracy due to the superior resolution, followed by the axial MR scan
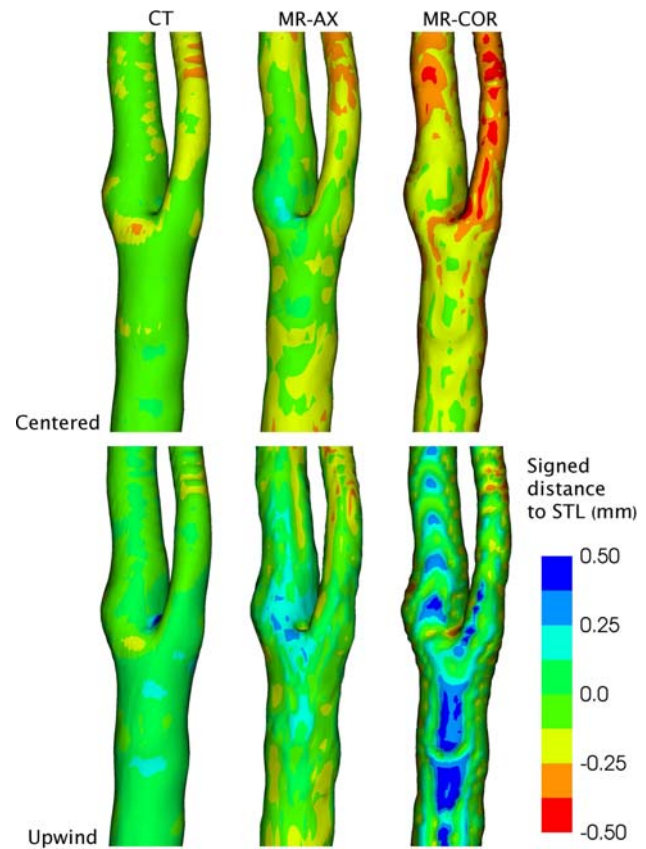
and the coronal MR scan. In particular, the accuracy in the coronal MR acquisition is hampered by the fact that coronal planes slice the carotid bifurcation longitudinally, which, due to the anisotropic resolution prescribed in the scanning protocol that leads the lumen to be spanned by a lesser number of voxels. Nevertheless, even the more severe underestimation of the lumen in this case, – 0.59 mm, is within the in-plane resolution, 0.78 mm, and well below the out-of-plane resolution, 2 mm. As for the comparison among the finite difference approximations of the gradient operator for the computation of the feature image, it can be noted that the centered finite difference approximation leads to increasing underestimation of the lumen (negative signed distance values) with decreasing cross-sectional resolutions. Such underestimation is uniform throughout the model, since it is not associated with an increase in the standard deviation of the error. On the other hand, such underestimation is not witnessed on average for the upwind approximation, at the price of a generally larger standard deviation and of possible

Table 1 Accuracy of image-based modeling evaluated on a rapid-prototyping phantom of a patient-specific carotid bifurcation

Segmentation accuracy

Feature image	CT vs STL	MR-AX vs STL	MR-COR vs STL	CT vs PTS	MR-AX vs PTS	MR-COR vs PTS
Centered	−0.08 ± 0.06 [−0.37, 0.15]	−0.17 ± 0.10 [−0.54, 0.21]	−0.24 ± 0.08 [−0.59, 0.02]	0.01 ± 0.06 [−0.30, 0.37]	−0.08 ± 0.11 [−0.53, 0.33]	−0.15 ± 0.08 [−0.42, 0.27]
Upwind	0.02 ± 0.07 [−0.25, 0.85]	−0.07 ± 0.12 [−0.50, 0.32]	−0.01 ± 0.15 [−0.41, 0.50]	0.11 ± 0.08 [−0.20, 0.83]	0.01 ± 0.13 [−0.45, 0.51]	0.08 ± 0.15 [−0.32, 0.68]

The table shows mean ± standard deviation and range of the signed distance between the segmented model and ground truth. Further details are provided in the Appendix

Centered and Upwind are the finite difference approximation of the gradient operator employed for the computation of the feature image $P(\mathbf{x}) = |\nabla I(\mathbf{x})|$ in Eq. 1

STL input model for stereolithography, PTS measurement points taken on the physical phantom, CT, MR-AX, MR-COR models generated from CT, MR axial and MR coronal phantom scans

overestimations of the lumen. This effect, also evident in Fig. 7, can be explained by the inferior smoothness of the upwind gradient operator. In fact, while centered finite differences are second-order accurate in the computation of first derivatives, the accuracy of upwind finite differences is only first-order. This can be obviated by the adoption of a regularization strategy as described in Sect. 3.3, which, for sake of clarity, has not been applied in this experiment.

References

1. Amenta N, Choi S, Kolluri R (2001) The power crust, unions of balls, and the medial axis transform. Comput Geometry Theory Appl 19(2–3):127–153
2. Antiga L, Steinman DA (2008) The Vascular Modeling Toolkit. http://www.vmtk.org. Last accessed August 9
3. Antiga L, Steinman DA (2004) Robust and objective decomposition and mapping of bifurcation vessels. IEEE Trans Med Imaging 23(6):704–713
4. Antiga L, Planken N, Piccinelli M, Ene-Iordache B, Huberts W, Remuzzi A, Tordoir J (2006) Geometric intra-subject variability of arm vessels assessed by MRA: a challenge for quantification and modeling of the vascular access for hemodialysis. In: 5th World congress of biomechanics, Munich, Germany
5. Antiga L, Ene-Iordache B, Caverni L, Cornalba GP, Remuzzi A (2002) Geometric reconstruction for computational mesh generation of arterial bifurcations from CT angiography. Comput Med Imaging Graph 26:227–235
6. Antiga L, Ene-Iordache B, Remuzzi A (2003) Computational geometry for patient-specific reconstruction and meshing of blood vessels from MR and CT angiography. IEEE Trans Med Imaging 22(5):674–684
7. Besl PJ, McKay ND (1992) A method for registration of 3D shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256
8. Bookstein FL (1989) Principal warps: thin-plate splines and the decomposition of deformations. IEEE Trans Pattern Anal Mach Intell 11(6):567–585
9. Caselles V, Kimmel R, Sapiro G (1997) Geodesic active contours. Int J Comput Vis 22(1):61–79
10. Cebral JR, Castro MA, Appanaboyina S, Putman CM, Millan D, Frangi AF (2005) Efficient pipeline for image-based patient-specific analysis of cerebral aneurysm hemodynamics: technique and sensitivity. IEEE Trans Med Imaging 24(4):457–467
11. Dey TK, Zhao W (2002) Approximate medial axis as a Voronoi subcomplex. In: Proceedings of the 7th ACM symposium of solid modeling applications, pp 356–366
12. Ene-Iordache B, Antiga L, Soletti L, Caverni L, Remuzzi A (2004) Validation of a 3D reconstruction method for carotid bifurcation models starting from angio CT images. In: 6th International symposium on computer methods in biomechanics and biomedical engineering, Madrid
13. Hernandez M, Frangi AF (2007) Non-parametric geodesic active regions: method and evaluation for cerebral aneurysms segmentation in 3DRA and CTA. Med Image Anal 11:224–241
14. Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W (1993) Mesh optimization. In: SIGGRAPH 93 conference proceedings
15. Ibanez L, Schroeder W, Ng L, Cates J (2005) The ITK software guide, 2nd edn. Kitware Inc., Albany
16. Kirk BS, Peterson JW, Stogner RH, Carey CF (2006) libMesh: a C++ library for parallel adaptive mesh refinement/coarsening simulations. Eng Comput 22(3):237–254

17. Krams R, Wentzel JJ, Oomen JA, Vinke R, Schuurbiers JC, de Feyter PJ, Serruys PW, Slager CJ (1997) Evaluation of endothelial shear stress and 3d geometry as factors determining the development of atherosclerosis and remodeling in human coronary. Aterioscler Thromb Vasc Biol17(10):2061–2065

18. Lieber BB, Siebes M, Yamaguchi T (2005) Correlation of hemodynamic events with clinical and pathological observations. Ann Biomed Eng 33(12):1695–1703

19. Long Q, Xu XY, Collins MW, Griffith TM, Bourne M (1998) The combination of magnetic resonance angiography and computational fluid dynamics: a critical review. Crit Rev Biomed Eng 26(4):227–274

20. Loop C (1987) Smooth subdivision surfaces based on triangles. Master's Thesis, University of Utah

21. Lorigo LM, Faugeras OD, Grimson WEL, Keriven R, Kikinis R, Nabavi A, Westin CF (2001) CURVES: curve evolution for vessel segmentation. Med Image Anal 5:195–206

22. Milner JS, Moore JA, Rutt BK, Steinman DA (1998) Hemodynamics of human carotid artery bifurcations: computational studies with models reconstructed from magnetic resonance imaging of normal subjects. J Vasc Surg 28(1):143–156

23. Schroeder W, Martin K, Lorensen WE (2005) The visualization toolkit, an object oriented approach to 3D graphics, 4th edn. Kitware Inc, Albany

24. Sethian JA (1999) Level set methods and fast marching methods, 2nd edn. Cambridge University Press, Cambridge

25. Sethian JA (1999) Fast marching methods. SIAM Rev 41(2):199–235

26. Si H, Gartner K (2005) Meshing piecewise linear complexes by constrained Delaunay tetrahedralizations. In:Proceeding of the 14th international meshing roundtable

27. Si H (2008) Tetgen, a quality tetrahedral mesh generator and three-dimensional delaunay triangulator. User's Manual, http://tetgen.berlios.de. Last accessed Sep 30, 2008

28. Surazhsky V, Gotsman C (2003) Explicit surface remeshing. In: Kobbelt L, Schroeder P, Hoppe H (eds) Eurographics symposium on geometry processing

29. Taubin G (1995) A signal processing approach to fair surface design. In: SIGGRAPH 95 conference proceedings, pp 351–358

30. Taylor CA, Draney MT, Ku JP, Parker D, Steele BN, Wang K, Zarins CK (1999) Predictive medicine: computational techniques in therapeutic decision making. Comput Aided Surg 4(5):231–247

31. Thomas JB, Milner JS, Rutt BK, Steinman DA (2003) Reproducibility of computational fluid dynamics models reconstructed from serial black blood MRI of the human carotid bifurcation. Ann Biomed Eng 31(2):132–141

32. Whitaker R (1998) A level set approach to 3D reconstruction from range data. Int J Comp Vis 29(3):203–231