



HAL
open science

Robust blood vessel surface reconstruction for interactive simulations from patient data

Ahmed Yureidini

► **To cite this version:**

Ahmed Yureidini. Robust blood vessel surface reconstruction for interactive simulations from patient data. Medical Imaging. Université des Sciences et Technologie de Lille - Lille I, 2014. English. NNT : . tel-01010973

HAL Id: tel-01010973

<https://theses.hal.science/tel-01010973>

Submitted on 21 Jun 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Thesis submitted to obtain the title of
Doctor of Philosophy

Doctoral School of Engineering Science
Field: Computer Science

Robust blood vessel reconstruction for interactive medical simulations

Prepared by Ahmed YUREIDINI at
Inria-Nancy Grand-Est

Defended on the 27th of March, 2014

Jury:

<i>Advisor:</i>	Stéphane COTIN	-	Inria/Université de Lille 1
<i>Advisor:</i>	Erwan KERRIEN	-	Inria/Université de Lorraine
<i>Reviewers:</i>	Elsa ANGELINI	-	Telecom ParisTech/Columbia University, School of Engineering
	Pascal HAIGRON	-	LTSI/Université de Rennes 1
<i>Examiners:</i>	Fernando BELLO	-	Faculty of Medicine, Department of Surgery & Cancer/Imperial College London
	Hervé DELINGETTE	-	Inria
<i>President:</i>			

ABSTRACT

Robust blood vessel surface reconstruction for interactive simulations from patient data

Abstract: In the context of interactive simulation, the lack of patient specific geometrical models remains one of the major limitations of simulators. Current commercial simulators proposed no or a limited number of cases. However, a vast literature on the subject has been introduced in the past twenty years. Nevertheless, the proposed methods are not adapted to an interactive context, especially when dealing with vascular networks.

In this work, we address the problem of blood vessel segmentation and reconstruction from 3DRA patient data. To this end, we propose two novel algorithms for segmentation and reconstruction. First, the vessel tree is built by tracking the vessel centerline. Our dedicated tracking process also extracts points on the vessel surface in a robust way. Second, those points are fitted by an implicit surface (a *blobby model*) that is iteratively refined. Tracking and reconstruction results are reported on synthetic and patient data. Simulations within an interventional tool navigation context showed that the resulting geometrical model complies with interactive simulation requirements: fast collision detection and prediction, topology information, smoothness and availability of differential quantities for contact response computation.

Keywords: Interactive medical simulation, Segmentation, Vessel tracking, Scattered data reconstruction, RANSAC, Blobby models

RÉSUMÉ

Reconstruction robuste des vaisseaux sanguins pour les simulations médicales interactives à partir de données patients

Résumé: Dans le cadre des simulations interactives, le manque de modèles géométriques reste une des limitations majeurs des simulateurs. Actuellement, les simulateurs commerciaux ne proposent pas ou, dans tout cas, un nombre limité de cas. Un grand nombre de travaux abordent cependant ce sujet tout au long de ces deux dernières décennies. Malgré une vaste littérature, les méthodes ne sont pas adaptées à un contexte interactif, plus particulièrement quand il s'agit des réseaux vasculaires.

Dans cette thèse, nous considérons le problème de la segmentation et la reconstruction des vaisseaux sanguins à partir de données patients en 3DRA. Pour ce faire, nous proposons deux nouveaux algorithmes, un pour la segmentation et un autre, pour la reconstruction. Tout d'abord, le réseau vasculaire est construit grâce à un algorithme de suivi de la ligne centrale des vaisseaux. De plus, notre procédure de suivi extrait des points à la surface des vaisseaux de manière robuste. Deuxièmement, ces points sont estimés par une surface implicite (un *blobby model*) qui est raffinée de façon itérative. Les résultats du suivi et de la reconstruction sont produits à partir de données synthétiques et réelles. Lors de la simulation de la navigation d'outils interventionnels, notre modèle géométrique remplit les exigences des simulations interactives: une prédiction et détection rapide des collisions, l'accès à l'information topologique, une surface lisse et la mise à disposition de quantités différentielles pour la résolution des contacts.

Mots-clés: Simulations médicales interactives, Segmentation, Suivi des vaisseaux, Reconstruction à partir de nuages de points, RANSAC, Blobby models

LIST OF PUBLICATIONS

Refereed conference papers

1. A. Yureidini, E. Kerrien, J. Dequidt, C. Duriez and S. Cotin. *Local implicit modeling of blood vessels for interactive simulation*. In *MICCAI - 15th International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 7510 of *Lecture Notes in Computer Science*, edited by N. Ayache, H. Delingette, P. Golland and K. Moria, pages 553–560, Springer, Nice, France, 2012b.
URL <http://hal.inria.fr/hal-00741307>
2. A. Yureidini, E. Kerrien and S. Cotin. *Robust RANSAC-based blood vessel segmentation*. In *SPIE Medical Imaging*, vol. 8314, edited by D. R. Haynor and S. Ourselin, page 8314M, SPIE Press, San Diego, CA, États-Unis, 2012a.
URL <http://hal.inria.fr/hal-00642003>
3. A. Yureidini, E. Kerrien and S. Cotin. *Reconstruction robuste des vaisseaux sanguins par surfaces implicites locales*. In *Orasis*, Praz-sur-Arly, France, 2011b.
URL <http://hal.inria.fr/inria-00579814>

Other conference papers

1. A. Yureidini, J. Dequidt, E. Kerrien, C. Duriez and S. Cotin. *Computer-based simulation for the endovascular treatment of intracranial aneurysms*. In *LIVIM Imaging Workshop*, Strasbourg, France, 2011a.
URL <http://hal.inria.fr/hal-00641990>

CONTENTS

List of publications	v
Contents	vi
List of Figures	x
List of Tables	xvii
I Introduction	1
1 Introduction	3
1.1 Context	3
1.2 Medical simulations	4
1.2.1 Endovascular procedure simulators	6
1.2.2 Challenges	6
1.3 Outline and contributions	9
2 Research Background	13
2.1 Boundary-based reconstructions	16
2.1.1 Discussion on geometrical models	17
2.2 Centerline-based reconstructions	18
2.2.1 Discussion on geometrical models	20
2.3 Model choice	20
2.4 Modality choice	21
II Blood Vessel Surface Reconstruction	23
3 Blood vessel segmentation	25
3.1 Introduction	26
3.2 RANSAC-based tracking (RBT) algorithm	26
3.3 Ray casting	28
3.4 Estimation step: cylinder estimation	29
3.4.1 Cylinder parametrization	29
3.4.2 Cylinder fitting	31
3.4.3 Geometrical filtering	32
3.5 Tracking a whole branch: Initialization, prediction step and stopping criteria	35
3.6 Tracking branches	37
3.7 Capturing aneurysms	38
3.8 Implementation details	39

3.8.1	Ray-casting	39
3.8.2	Interpolation	40
3.9	Validation protocol	47
3.9.1	Correspondence between trackings	47
3.9.2	Initial conditions	48
3.9.3	Centerline evaluation	49
3.9.4	Clinical data	49
3.9.5	Parametrization of the reference method: MHT	50
3.9.6	Parametrization of our algorithm: RBT	50
3.9.7	Evaluation measures	52
3.10	Results	54
3.11	Discussion	55
3.12	Conclusion	57
4	Blood Vessel Reconstruction	63
4.1	Introduction	64
4.2	Implicit formulation	66
4.2.1	Implicit function	66
4.2.2	Kernel choice	67
4.2.3	Parameters and locality	69
4.2.4	Distance function approximation	71
4.3	Energy formulation	73
4.3.1	Input data considerations	74
4.3.2	Test scenarios	75
4.3.3	Framework	77
4.3.4	Configurations	78
4.3.5	Preliminary discussion	80
4.3.6	Our energy-based framework	83
4.4	Blobby model refinement	84
4.4.1	Selection-Subdivision	85
4.4.2	Optimization	86
4.5	Implicit Modeling	86
4.5.1	Overall implicit modeling	86
4.5.2	Local implicit modeling	88
4.6	Using the local implicit models for simulation	90
4.7	Visualization	91
4.8	Experimental validation	91
4.8.1	Clinical data	91
4.8.2	Synthetic data	92
4.8.3	Parametrization	93
4.8.4	Evaluation measures	94
4.8.5	Validation protocol	94
4.9	Results	95
4.9.1	Reconstruction on synthetic data	95

4.9.2	Reconstruction on patient data	99
4.9.3	Visual assessment	99
4.9.4	Accuracy	99
4.9.5	Computation time	100
4.9.6	Compacity	101
4.10	Discussion	102
4.11	Conclusion	103
5	Applications: coil embolization simulation	107
5.1	Introduction	108
5.2	What do we need for a coil embolization simulation?	108
5.2.1	Time integration and numerical choices	109
5.2.2	Mechanical model	110
5.2.3	Geometrical model	114
5.3	Elementary step simulation	115
5.3.1	Free motion	115
5.3.2	Correction step	116
5.4	Overall simulation workflow	119
5.4.1	Collision detection	120
5.5	Experimental details	122
5.5.1	User-interaction	122
5.5.2	Data	123
5.5.3	Validation criteria	125
5.6	Results and discussion	127
5.7	Conclusion	130
III	Conclusion	133
6	Conclusion and Perspectives	135
6.1	Contributions	136
6.2	Perspectives	137
A	Appendix A	141
A.1	RANSAC-based Tracking InTerface (RTrackingIT)	141
A.1.1	Mayavi widget	142
A.1.2	Menu bar	142
A.1.3	Configuration section	142
A.1.4	Labeling section	143
B	Appendix B	145
B.1	Fomulae for a blob	145
B.2	Formulae for blobby models	146
B.2.1	Implicit function, gradient, Hessian matrix	146
B.2.2	Energies and their derivatives	147

References	149
Abbreviations	168

LIST OF FIGURES

1.1	Medical pipeline: first, the pathology is diagnosed (diagnosis); then it is treated (treatment) and finally, the physician follows the patient during its convalescence (prognosis). Medical simulations may help the physician to rehearse before the intervention (green arrow), to transfer his/her knowledge to residents (cyan arrow) and to predict possible complications after the intervention (purple arrow).	5
2.1	Two cut planes through 3DRA data exhibiting the Kissing Vessel (KV) issue: (left) A vessel runs along an aneurysm. (right) Three locally tangent vessels. .	15
2.2	Toy scenario of a problematic case for two consecutive simulation time steps. The catheter is represented as five connected points where red color is related to possible/detected collision, otherwise points are in black. All points remain inside the same vessel (left). All points lay inside the vasculature but in two different vessels (right).	15
3.1	Outline of RANSAC-Based Tracking (RBT) vessel tracking algorithm. (a) Cut plane through the original three-Dimensional (3D)Rotational Angiography (RA) data; (b) Parameter update: Starting from C_0 (green dot) and \vec{d}_0 (white arrow), the new center C (red dot) is found. $\vec{d} = \vec{d}_0$. Candidate points (white dots) are extracted at the vessel surface by casting rays in the volume, from C . Only the points around the displayed cut plane are shown. (c) N_d directions are tested for the cylinder axes (yellow arrows). One fitting cylinder is found using RANdom SAMple Consensus (RANSAC) , per axis. (d) Best cylinder found, together with its consensus set (green dots) and the final points used to set the cylinder height.	27
3.2	(left) A ray – of length four times the estimated vessel radius – is cast from inside the vessel lumen (blue dot). (center) The directional gradient along the ray is computed with central differences method. (right) In 3DRA data, the minimum along the ray characterizes the vessel wall.	27
3.3	Two radiation focus and their extracted points (blue) on vessel of width ρ : rays have length $\mathcal{L} = 3\rho$ (left) and $\mathcal{L} = 2\rho$ (right). A ray length of twice the local vessel radius is insufficient to capture points at the opposite wall when the ray-casting center is close to a vessel wall.	28
3.4	two-Dimensional (2D) representation of the proposed geometric criterion. The vessel lumen varies within a 2D -torus spanned with a cylinder (red dashed contour) and with minor radius r_t (green and blue circle). The circular hypothesis is insufficient for capturing the vessel cross-section (black dotted contour).	33

- 3.5 Influence of Powell refinement in the resulting cylinders. The corresponding cylinders for both centerlines produced without (middle) and with (right) Powell optimization on one vessel (left). Smooth transitions between cylinders are noticed at curved sections when using the refinement procedure. 34
- 3.6 Tracking a new vessel. 35
- 3.7 Influence of the step distance on the dense sampling. Resulting points for the prediction step $C = C^* + s \cdot h^* \cdot \vec{d}^*$: (top left) $s = 0.25$, (top right) $s = 0.5$, (bottom left) $s = 0.75$ and (bottom right) $s = 1$. Interpenetration of point-sets is almost inexistent for $s = 1$, while on the contrary a value of $s = 0.25$ exhibits point-sets covering the same area of the blood vessel. A step size ranging from $s = 0.5$ to $s = 0.75$ offers a good trade-off between interpenetration and distance between predictions. 36
- 3.8 Pseudo-automatic seed generation from a parent vessel (red). (From top to bottom) The first three rows exhibit instances of tracking initialized with seeds proposed by RBT (Fig. 3.6) for three different vessels width: 2mm (top), 1mm (center) and 0.6mm (bottom). (From left to right) First column displays the parent vessel and the target vessel lumen. Second column puts forth the corresponding seed. Third column shows the resulting point-set. The last row evinces the segmented vascular tree (left), a new user-defined seed – only the seed’s radius was manually modified – for the third-row case (middle) and the corresponding points for the tracking (right). 36
- 3.9 RBT robustness against rough tracking directions. Trackings of Fig. 3.8 – where seeds were automatically generated – were inspected. Directions were manually modified for the three former cases and the stem vessel (carotid) was included in the inspection : the largest vessel (1st row), a large vessel (2nd row), a medium size vessel (3rd row) and a small size vessel (4rd row). (From left to right) First column displays the parent vessel, except for the first row, and the target vessel lumen. Second column puts forth the modified direction of the seed. Third column shows the resulting point-set. The last column evinces the adaptation of cylinders to capture the vessel lumen. 37
- 3.10 Geodisation principle on one facet of an icosahedron. (left) First, the mid-points (green) of the facets (yellow) are computed. Then the facet is subdivided in 4 triangles. (right) Finally, the vertex of each triangle (yellow) are projected onto the unit sphere. 40
- 3.11 Four geodisation orders of an icosahedron from left to right: order 1, 2, 3 and 4. The white dots represent the vertices of the resulting tessellation. 40
- 3.12 A unit circle (green) of center C and radius ρ is discretized by $N_{px} = 2$ pixels along its diameter. The spatial discretization process of the circle starts by, first, computing spatial resolution $S = 2\rho/N_{px}$ in both directions. Finally, a mean filter is used for assigning to each pixel an intensity value (grayscale). In the end, an image is thus generated. Within the image, the ray-casting focus \mathcal{F} is allowed to evolve inside the circle and from which, rays are cast evenly sampled with length $\mathcal{L} = 2.5\rho$ 42

- 3.13 (Rows) Each row corresponds to one configuration in the interpolation experiment. The blue dot point is the center C of the blue unit circle \mathcal{C} . The red square indicates the ray-casting focus \mathcal{F} which extracts the green points. Experiments were run with $C = \mathcal{F} = (0,0)$: the unit circle was discretized with $N_{px} = 3$ (1st column) and $N_{px} = 7$ pixels (2nd column). In the last two columns, the center was placed at $(0.5, 0.4)$ and the ray-casting focus was positioned at $(-0.3, 0.4)$: the unit circle was sampled by $N_{px} = 3$ (3rd column) and $N_{px} = 7$ pixels (4th column). 43
- 3.14 **RBT** and **Multiple Hypothesis Tracking (MHT)** algorithms tracking a vessel (left). **RBT** (center) follows a different path than **MHT** (right). 48
- 3.15 Four points placed manually inside the vessel at the proximal end of the vessel (left) providing an initial tracking position, a tracking direction and an estimate of the vessel radius (right). 49
- 3.16 Four of the twenty vessels used for tuning the **RBT** parameters which exhibit kissing vessel issues (top left), fuzziness (top right), tortuosity (bottom left) and drops in image intensity (bottom right). 59
- 3.17 Top: (left) Maximum Intensity Projection of one patient data. Centerline delineation produced by **MHT** (middle) and **RBT** (right). Bottom: (left) When the **MHT** centerline (red) does not remain inside the vessel wall, it is cut – during the tracked length computation – before the problem arises (green). (middle) The whole **MHT** centerline (red) and the points on this centerline (cyan) selected for **ASSD** computation ($w = \bar{\rho}$). (right) The resulting points for **MHT** (red) and **RBT** (blue) involved in the **ASSD** computation. 59
- 3.18 Comparison of **MHT** (left) and our algorithm (right) with regard to a **Kissing Vessel (KV)** case on Patient 1: a blood vessel runs along the aneurysm sack. Our **RBT** algorithm successfully handles this case, while **MHT** is perturbed by the aneurysm lumen which misleads the tracking. 59
- 3.19 (left) Original image data for three kissing vessels (top) and a vessel running along an aneurysm (bottom). (right) The corresponding segmentation with **RBT** (same case as in Fig. 3.18). 60
- 3.20 (left) A tortuous branch of the posterior choroidal artery (0.6 – 0.7 mm of diameter) in Patient 3. (middle) **MHT** is misled by neighboring structures whereas (right) **RBT** tracks the tortuous vessel. 60
- 3.21 **MHT** deals well with drops in image intensity. (left) A cut plane along a vessel presenting drops in image intensity; (right top) **RBT** result: the tracking stops too early. (right bottom) The resulting centerline for **MHT**. 61

3.22 (Top) Influence of user-defined radius estimation bound on MHT tracking where underestimation of the vessel radius leads to an inaccurate outcome on Patient 5: (left) the upper bound is reached leading to <i>leakage issues</i> and (middle) computation with larger values avoid this problem. (right) Our RBT algorithm successfully tracks the vessel without resorting to this information. (Bottom) Plane views: the detection of the centerline for the MHT on a fuzzy portion of 0.8 mm of diameter, with saturation of the radius (left) and without saturation (middle), is disturbed whereas our algorithm detects correctly the centerline	61
3.23 Our RBT algorithm applied to Magnetic Resonance Angiography (MRA) data (left). The dense sampling produced on 37 vessels (middle) and the tracked vessels superimposed upon the original data (right).	62
4.1 Outline of the local reconstruction of the vessel walls. (a) Tracking results for two arteries on 3D RA data: Centerlines and extracted points at the arteries surface; (b) Choose one node for reconstructions: in this case, it represents a node at the bifurcation. (c) Neighboring nodes are selected with respect to (w.r.t) a topological distance of one. (d) Points belonging to neighboring nodes and the node of interest constitute the point cloud for local reconstruction. (e) Two blobs with width equal to the estimated radius serve as initial Blobby Models (BMs) . (f) Final reconstruction is achieved after fissioning one blob at a time. Thirty subdivisions were necessary to produce the resulting implicit surface.	66
4.2 Implicit function as a sum of 3 Gaussian kernels in (left) one-Dimension (1D) and (right) 2D . The implicit contour S is defined as the locus of points where $f(X; p) = T$	67
4.3 Kernels expression and shape.	68
4.4 Two blobs scalar field functions (solid lines), generated with Gaussian kernels, and their corresponding gradient functions (dashed lines). (left) Blobs with $\alpha_j \neq \rho_j$ exhibit highly different gradient profiles due to scaling factor α_j/ρ_j . (right) Blobs with $\alpha_j = \rho_j$ display similar height variation on their respective gradient functions, i.e. the scalar field shows a constant variation with disregard of blobs width.	69
4.5 Two scalar field functions representing the same surface at $T = 0.4$ and generated with 3 Gaussian kernels: (left) $\alpha = \{0.6715, 1.2006, 0.8681\}$ and $\rho = \{0.3722, 0.3258, 1.0863\}$ takes different values which exhibit high variations in the scalar field, while on the contrary, (right) the scalar field height is smoothed – i.e. even variations – by setting $\alpha = \rho = \{0.3722, 0.3258, 1.0863\}$	70
4.6 (left) A BM – based on Gaussian kernel – composed of three blobs: we set $\alpha_j = \rho_j$ and values were the same as in Fig. 4.2). (right) Local shape preservation is achieved when selecting two blobs within the BM (locality).	71

- 4.7 An interventional tool – composed of points A and B – interacting with the implicit surface S . (left) During simulation, two different T-values may be considered for predicting collisions: T and $T + \epsilon$. One defining the actual implicit surface S which is used as the constraint surface for computing contact forces. Another defining an implicit surface S_ϵ inside \mathcal{S} . No point A inside \mathcal{S} is in collision. A prediction zone is created between S and S_ϵ where any candidate point B is regarded as a candidate point for collision. (right) When a point B of the interventional tool is outside the surface, it is projected onto the surface – at location B' – along the implicit function gradient. 71
- 4.8 A **BM** estimating a circle of 2cm radius. (left) Blobs with $\alpha = \rho$ – and their corresponding contours at 0.1-level – composing the **BM**. (right) The associated scalar field. 73
- 4.9 Scalar field function generated with one negative height value for the Gaussian kernels: $\alpha = \{-1.2006, 0.8681\}$ and $\rho = \{0.3258, 1.0863\}$. The negative height produces a valley in the scalar field. Consider the line $y = 0$, ∇f gives the right direction for the recall force if $x > 0$, no force at all for $x = 0$ and on opposite, repulsive force for $x < 0$ 73
- 4.10 (left) Point-set extracted at the surface of a small and tortuous vessel with **RBT**. (right) For a better depiction, three **RBT** cylinders and their corresponding point-sets are displayed. Owing to the high curvature of the vessel, the point-set exhibits unbalanced density of points (top vs bottom points). Besides, point-set displays holes at its extremities. 75
- 4.11 (left) Input point-set of a synthetic aneurysm to be reconstructed from a **BM** composed of 4 blobs in red. (right) The noise-free point-set is displaced along the normal direction with a normal distribution (noise corruption). 75
- 4.12 Open noisy point-set which represents a vessel contour. (left) 6 points were placed outside the vessel so that outliers are mimicked. Outliers normal was randomly affected. (right) This time, 6 points were placed inside the vessel and their normal was randomly assigned. 76
- 4.13 Point-set of a synthetic aneurysm to be reconstructed from a **BM** composed of 4 blobs. Reconstruction fulfilled through successive subdivision of one blob using the \mathcal{E}_d (left) and $\mathcal{E}_d + 10^{-4}\mathcal{E}_n$ (right). The resulting **Implicit Contour (IC)** is supplied by the green curve while the centers of the blobs composing the final **BM** are located by the red crosses. 79
- 4.14 Same case as those presented in Fig. 4.13 but point-set is corrupted by noise. Reconstruction is fulfilled through successive subdivision of one blob using the \mathcal{E}_d (left), $\mathcal{E}_d + 10^{-3}\mathcal{E}_n$ (middle) and $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 5 \cdot 10^{-3}\mathcal{E}_a$ (right). The resulting **IC** is given by the green curve while the centers of the blobs composing the final **BM** are located by the red crosses. 80

- 4.15 Open noisy point-set which represents a vessel contour. 6 points were placed outside the vessel so that outliers are mimicked. Outliers orientation was randomly affected. Reconstruction fulfilled through successive subdivision of one blob using the $\mathcal{E}_d + 10^{-3}\mathcal{E}_n$ (left), $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c$ (middle) and $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c + 10^{-8}\mathcal{E}_a$ (right). The resulting **IC** is giving by the green curve while the centers of the blobs composing the final **BM** are located by the red crosses. 81
- 4.16 (left) Open noisy point-set which represents a vessel contour. 6 points were placed outside the vessel so that outliers are mimicked. Outliers orientation was randomly affected. Reconstruction fulfilled through successive subdivision of one blob using the $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c$ (middle), and $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c + 10^{-8}\mathcal{E}_a$ (right). The resulting **IC** is giving by the green curve while the centers of the blobs composing the final **BM** are located by the red crosses. 82
- 4.17 The elementary reconstruction step in 2D. (a) Points (black) are to be reconstructed with two blobs (red contour). (b) First, all blobs are fined tuned over the positions and then, over the widths. (c) Second, find the point farthest to the contour (cyan) and the blob closest to it (purple). (d) Third, replace this blob with two blobs. The first (black contour) is placed at the same position (black) and the second (cyan contour) is translated in the direction of the farthest point (cyan). Widths are calculated so that both blobs centered at the black point generate the same iso-surface as the fissioned blob (purple). (e) Fourth, the translated blob (cyan) is tuned over its center and width. (f) Finally, the **BM** is updated by integrating the recently tuned blob. 87
- 4.18 Implicit modeling of an aneurysm. The points $\{P_i\}$ are in red. (From left to right) Initialization with a single blob ; after the first minimization ; after 25 subdivisions ; final result (100 subdivisions) 89
- 4.19 Selection of a local **BM** during simulation. The current local **BM** surface used to solve the constraints at the tool tip is displayed in wire-frame for 4 simulation steps. The overall vessel surface is shown in transparent red. Discrepancies with the **BMs** might occur, due to this surface being simplified in order to minimize the CPU load devoted to visualization. 90
- 4.20 A tiny vessel stems from a portal vessel. In this situation, a tool point P using the local implicit surface \mathcal{S}_A (dashed contour) associated to node \mathcal{N}_A needs to attain a certain distance – i.e. a plane – to be associated to node \mathcal{N}_B in order to employ implicit surface \mathcal{S}_B (filled region). The plane position varies whether relying on the Euclidean distance – i.e. the Voronoi diagram – (left) or on a Power diagram (right) – i.e. the weighted Voronoi diagram. 91
- 4.21 Synthetic data: four convolution surfaces (blue) generated with point-based skeletons (red). 300 points evenly spaced where placed on the surface. 93
- 4.22 The maximal **Geometric Distance (GD)** value for 100 **BMs** reconstructed from \mathcal{S}_r sampling. For sake of clarity, values 1 and 2 for N_s were excluded from charts. The **BM** with the lowest figure was selected (see Fig. 4.23) and quantitatively further detailed in Table 4.6. However, we can see that all the shapes were very well modeled with no more than 20 blobs. 96

4.23	Fitting of 300 points (white) from initial BM s (red) and producing the blue surface.	96
4.24	Reference surface (blue) and the resulting BM s (purple) for maximal values in Table 4.7.	98
4.25	Geometrical model for patient 10 at different values for t_g : (From left to right) 0.2mm, 0.3mm and 0.5 mm. The top row displays the whole geometrical model while the bottom row presents their respective close-ups to the stem vessel. $t_g = 0.2$ mm presents variations in the stem vessel surface. On the other hand, $t_g = 0.5$ mm smoothes these variations. $t_g = 0.3$ mm displays a fair trade-off between smoothness and accuracy.	100
4.26	Visual assessment on patient 6 (from left to right): iso-surface from the raw data set ; iso-surface from the BM s (13956 models, 51272 blobs, $t_g = 0.3$, $N_s = 100$): much more vessels are visible; close-up showing smooth transitions between models ; close-up on a small artery branching onto the carotid: the connection is difficult to model	103
5.1	Forces acting on a beam element whose axis is defined by points 0 and 1 (left): axial forces F_1 and F_7 ; shearing forces F_2 , F_3 , F_8 and F_9 ; bending moments F_5 , F_6 , F_{11} and F_{12} ; and twisting moments (torques) F_4 and F_{10} . The local Frenet frame $\mathcal{F}_0 = [n_0, t_0, b_0]$ located at extremity 0 helps for expressing displacements induced by these twelve forces (right).	110
5.2	The medical device is modeled as a series of $N = 4$ beam elements $B^{(i)}/i = \{1, \dots, N\}$. (left) The mechanical device's stiffness matrix K – of dimension $(6(N+1) \times 6(N+1))$ – represented in a global frame has rank $6N$. (right) Meanwhile, the proximal extremity 0 in the mechanical device is fixed so that matrix K – of dimension $6N \times 6N$ and rank $6N$ – becomes invertible.	113
5.3	Unilateral contact formulation for one collision point α – pertaining to the interventional tool – of associated frame $\mathcal{F}_\alpha = [\vec{n}_\alpha, \vec{t}_\alpha]$. (left) Signorini's law: the closest peer of α on the vasculature is retrieved – i.e. α' . The gap $\delta_{\vec{n}_\alpha}$ between α and α' must be null to avoid interpenetration. Coulomb's law: The force exerted at α is decomposed into normal and tangential components. There is respectively sticking and slipping along \vec{t}_α when the resultant vector is inside the cone and over the cone.	118
5.4	From left to right and top to bottom: six simulation steps from 0.06–0.011 ms (time step 0.001 ms). Only collision detection for the tool tip is displayed. For each time step, the tip initial position (before correction) in magenta and the corrected position in black (superimposed in all cases aside for 0.06 ms) are shown. The normal vector (red), the tangential vector (blue), the binormal vector (green) and the linear representation of the surface (plane in light blue) at the corrected position are also displayed.	121
5.5	The silicon vascular network used for validation.	123

5.6	Capsule (left) and spiral (right) surfaces; and their corresponding skeleton (white). Both shapes presented respectively dimensions (width×height×depth) in mm: $11.9 \times 2.5 \times 2.5$ and $52.3 \times 39.9 \times 7.2$. A primitive was placed at each red dot composing the skeleton.	124
5.7	(left) Convolution surface for a line segment created with a Cauchy kernel. (middle) Projected vertices generated from a marching cubes algorithm. Their projection on the implicit surface is carried out via a Newton Raphson technique. (right) Points evenly spaced on the implicit surface that are associated (colored dots) to their closest node in the <i>centerline</i>	124
5.8	Tool tip trajectory for both Local Implicit Modeling (LIM) (left) and Triangular Mesh (TM) (right) models. Points utilized for computing \mathcal{L} : points leaving and not colliding with the surface are excluded (red). The first colliding point P_0 (green) and its normal vector define plane \mathbb{P} . The selected points \mathcal{P} (gray) are projected onto \mathbb{P} and the resulting positions are used for fitting line \mathcal{L} (top). Motion smoothness was quantified by computing the distance from the tip positions to the surface (bottom).	126
5.9	Simulation of catheter deployment inside a capsule-like shape. The collision is handled with a triangular mesh (left) and LIM (right). Even with a fine resolution mesh (12k triangles) artificial friction appears that eventually creates a loop in the catheter. LIM (12k blobs) enables the catheter to smoothly slide along the surface.	129
5.10	Numerical simulation at non interactive rates on a surface generated with LIM . The tip of the tool experiencing bumps at transitions between two implicit surfaces when sliding on the surface.	130
A.1	The overall window of RANSAC-based Tracking Interface (RTrackingIT) Graphical User Interface (GUI) . Four sections compose RANSAC-based Tracking Interface (RTrackingIT) : the menu bar (top), the mayavi widget (left), the configuration (middle) and the labeling sections (right).	141
A.2	RTrackingIT in action. Three instances of tracking with RBT combined with a triangulated mesh produced with Mayavi Isosurface filter. In the labeling section, vessels CID (red), ACA (cyan) ANE (lime) are respectively root, leaf and trial instances.	143

LIST OF TABLES

1.1	A brief summary of current endovascular procedure simulators and related works.	7
3.1	Parameter values for the RBT algorithm.	39

3.2	Outcomes for the 9 configurations. Each one was run 102060 times for different positions of the unit circle \mathcal{C} (ground truth) and the ray-casting focus. Five metrics were evaluated at two different scales of discretization of the unit circle: (All) all levels $N_{px} \in [2, \dots, 10]$ and (Low) discretizations ranging from 2 to 4 pixels (34020 instances per configuration). Each discretization scale computation may present its mean value μ and standard deviation σ . The first metric $\overline{RMS_{C^*}}$ measures the average root mean square error of the estimated center C^* to the center of unit circle C . Likewise, $\overline{RMS_{\rho^*}}$ informs about the average root mean square error of the estimated radius ρ^* to ground truth radius $\rho = 1$. $RMS_{\mathcal{C}^*}$ (respectively $RMS_{\mathcal{C}}$) computes the root mean square distance of the extracted point-set \mathcal{P}^* to the estimated circle \mathcal{C}^* (to the ground truth \mathcal{C}). Finally, the inlier rate p_{inl} qualifies how well the point-set \mathcal{P}^* complies with \mathcal{C} within a distance of $\rho/10$. Pink and green cells indicate respectively the worst and the best values for each column.	46
3.3	The number of excluded vessels per patient for both algorithms. Due to bad initialization, a certain number of vessels were excluded from the data-set. Similar figures were accounted for both algorithms.	49
3.4	RBT initial configuration: values tuned on twenty vessels from the data set.	51
3.5	Proposed ranges for RBT parameters according to the success rate scored in 20 vessels from the data-set. An optimal configuration is made up from this ranges.	51
3.6	Vascular tree of ten patients tracked with the initial (Table 3.4) and optimal (Table 3.5) configurations. Increase percentage in time and length between both configurations. In average, optimal values increase length and time tracking by 7% and 31% respectively.	52
3.7	Time, in minutes, and number of RBT instances needed to track a complete vascular tree. Overall, one cylinder computation took around 0.01 – 0.3 seconds.	55
3.8	Success rates of MHT and RBT on a data set of 10 patients. A total of 744 test vessels were used to evaluate both algorithms. The average success rate for MHT and for our algorithm is respectively 65% and 94%.	56
3.9	Success rate of MHT and RBT algorithms with tracking results classified as <i>medium</i> and <i>long</i> (M & L). The mean success rates for MHT and for our RBT algorithm are respectively 69% and 89%. Furthermore, the mean Average Symmetric Surface Distance (ASSD) , with a distance threshold of 3 voxels ($w = 3$) and the mean radius of the targeted vessel ($w = \bar{\rho}$), between the extracted centerlines by both methods is below one voxel (0.17-0.18 mm). The Common path Length (CL) corresponds to the total length of centerline used to compute ASSD on vessels successfully tracked by both methods. On average the total tracked lengths (Tracked Length (TL)) for MHT and RBT are respectively 778.6 mm and 1446.1 mm.	56
4.1	Comparative table of $\Phi(r)$ and $\phi(x)$ ($x = r^2$) kernel expressions of Fig. 4.3 with $r = P - C $	69

4.2	Two distances measures Hausdorff Distance (HD) and Average Symmetric Surface Distance (ASSD) – measured in pixels – and the resulting BMs capacity – i.e. the number of blobs (# Blobs) – for the aneurysm configurations. The first two rows belong to the noise-free point-set while the last two rows pertain to the noisy configuration.	80
4.3	Two distances measures Hausdorff Distance (HD) and Average Symmetric Surface Distance (ASSD) – measured in pixels – and the resulting BMs capacity – i.e. the number of blobs (# Blobs) – for the vessel. The first two rows belong to the configuration with 6 outliers outside the point-set and the last two rows pertain to that exhibiting 6 outliers inside the point-set.	82
4.4	Number of RBT instances per patient and the number of aneurysms segmented for 10 3DRA patient data.	92
4.5	Parameter values for the proposed vessel tracking and reconstruction algorithms.	94
4.6	Minimum (min), median (med) and maximum (max) distances (mm) computed between the original shape and the BM with the lowest geometric error to noiseless input points. Metrics are computed with Taubin's approximate Geometric Distance (TaGD) and the Geometric Distance (GD) . The initial BM was composed of N_{bi} . The reference surface was modeled by N_{br} blobs, meanwhile the resulting BM with the lowest GD presents N_b blobs. Four shapes were considered (see Fig. 4.23): arc (A), bifurcation (B), cylinder (C), open cylinder at extremities (Co).	97
4.7	Four shapes were considered: arc (A), bifurcation (B), cylinder (C), open cylinder at extremities (Co). All shapes have a radius of 1.0 mm. Gaussian noise was added along the normal to the surface to 300 points on the reference surface. The standard deviation σ varied from 0.025 to 0.25 mm (step 0.025), 300 tests per shape were run with $t_g = 3\sigma$ and other values set to values in Table 4.5. The minimum (min), median (med) and maximum (max) values of the error fit distribution d_{bm} (in mm) and computed with Taubin's approximate Geometric Distance (TaGD) and the Geometric Distance (GD) for the 300 noise configurations. For comparison with Table 4.6, the distribution of the resulting BM number of blobs N_b distribution is also given.	97
4.8	Distribution in % of the BMs according to the error of fit (d_{bm} , in voxel) in 4 classes (left column). v_s is the voxel size of the 3DRA data which ranges from 0.18 to 0.22 mm. 9 algorithm configurations were investigated depending on parameters t_g (targeted accuracy of fit) and N_s (maximum model complexity).	100
4.9	Time computation for 9 configurations grounded on the threshold accuracy $t_g = \{0.2, 0.3, 0.5\}$ (mm) and the number of subdivisions $N_s = \{30, 50, 100\}$ for each patient data. For each t_g configuration, the average computation time μ and its standard deviation σ were computed on all N_s configuration outcomes. Green and red colored cells respectively represent the lowest and the highest values for a t_g row. N_b gives the number of BMs modeling patient's vasculature.	101

4.10	Compacity assessment for 9 algorithm configurations depending on parameters t_g (targeted accuracy of fit) and N_s (maximum model complexity). A total number of 87564 BMs for 10 patients were considered. The total number of blobs N_b per configuration is provided. For a given t_g and a patient, compacity Γ was computed as the ratio between its average number of blobs – computed on all N_s – and the total amount of BMs for this patient. The average compacity Γ is supplied – as the mean of all compacity ratios at the same t_g	101
4.11	For each configuration, a triangulated surface was generated from the final geometrical model reconstructed with LIM . The number of triangles $\#\Delta$ for the triangulated surface and the number of blobs N_b composing the local implicit model were recorded. Distribution of the minimum (min), median (med) and maximum (max) ratio between $\#\Delta$ and N_b for all patients is given.	102
5.1	Parameter values for tracking (RBT) and reconstruction (LIM) algorithms.	125
5.2	Accuracy computation of the tool tip trajectory when interacting with a capsule shape modeled with LIM and TMs . N_p and N_u respectively represent the total number and the number of used tip positions. \mathcal{L} provides the line fitted to the trajectory and its corresponding Root Mean Square (RMS) error. Finally, statistical information – minimal, medial, 90th-percentile and maximal values – of the distance between tip positions and the surface during the slip motion.	127
5.3	Computation timings for collision detection on various deployment scenarios. For a comparable amount of primitives, collision detection on LIM outperforms collision detection on triangular meshes even whit the systematical use of state-of-the-art acceleration techniques. Thanks to the topological locality, the computational requirement for collision detection on LIM remains low even for large data-sets (e.g. more than 10k blobs).	128

Part I

Introduction

INTRODUCTION

1.1 Context

An estimated 17.3 million people died from **Cardiovascular diseases (CDV)** in 2008, representing 30% of all global deaths¹. Of these deaths, an estimated 7.3 million were due to coronary heart disease and 6.2 million were due to stroke². Globally, stroke is the second leading cause of death. It is a disease that predominantly occurs in mid-age and older adults who represent 85% of deaths due to stroke³. Around 9.4 million deaths each year can be attributed to high blood pressure. This includes 51% of deaths due to strokes and 45% of deaths due to coronary heart disease⁴. The number of people who die from **CDV**, mainly from heart disease and stroke, will increase to reach 23.3 million by 2030; this trend will let **CDV** and stroke to remain the leading causes of death. The pathological background for stroke may either be ischemic, i.e. occlusion of arteries supplying the brain due to a thrombus or blood clot; or haemorrhagic, i.e. bleeding, disturbances of the cerebral blood circulation. Among haemorrhagic strokes, we find intracerebral haemorrhage, i.e. bleeding from one of the brain arteries into the brain tissue; and **Subarachnoid Haemorrhage (SAH)**, i.e. arterial bleeding in the space between the pia mater and arachnoidea meninges.

Heart diseases and stroke cause billions of dollars in losses of national income each year in the world's most populous nations. Causes of this cost are: large numbers of premature deaths, ongoing disability in many survivors, impact on families/caregivers and on health services. To lessen these figures, two main streams are candidate solutions: first, prevention – i.e. education and food regulation laws – but it's difficult to put it in practice due to lack of political commitment worldwide; and second, improved health care, early detection and timely treatment is another effective approach for reducing the impact of vascular diseases. However, appropriate treatment is a challenging task due to the human morphology complexity and high level techniques are needed for this purpose.

For instance, many cerebrovascular pathologies like ischemic strokes, aneurysms⁵,

¹Global status report on noncommunicable diseases 2010. Geneva, World Health Organization, 2011

²Global atlas on cardiovascular disease prevention and control. Geneva, World Health Organization, 2011

³http://www.who.int/chp/steps/Section1_Introduction.pdf

⁴The global burden of disease: 2004 update. Geneva, World Health Organization, 2008

⁵A cerebral aneurysm is an area where a blood vessel in the brain weakens, resulting in a bulging or ballooning out

Arteriovenous Malformations (AVM)⁶, etc. are now often treated using **Interventional Neuroradiology (IN)** therapies which rely on the insertion and navigation of a catheter (long flexible thin tube) inside the vessels. Instead of open surgery, it allows to reach the lesion with surgical tools through a catheter. The treatment is delivered directly within the closed brain, using only image-based guidance.

The archetype of this high-level technique is the endovascular treatment of aneurysms, **also known as (a.k.a)** coil embolization or coiling, which involves the introduction of a catheter into a large blood vessel (Seldinger technique⁷). Typically it is inserted into the femoral artery (groin) and injected with a radio-opaque dye (contrast agent) that can be seen on live X-ray or fluoroscopy. A micro-catheter or a guide-wire is manipulated – placed inside the catheter along the length of the blood vessel – to reach the bulge area of the aneurysm. A small and thin wire (width of a human hair), i.e. a coil, made of soft platinum and presenting different shapes – e.g. a spring like shape – is inserted through the micro-catheter and when this latter has reached the collar (zone of insertion of the bulge on the artery), a coil is deployed. More than one coil is often packed into the aneurysm for preventing blood flow from entering it. Finally, the coil is detached from the guide-wire, e.g. an electrical current is used for this purpose when manipulating **Guglielmi Detachable Coil (GDC)**⁸.

Put another way, the dedicated skill of instrument navigation and the thorough understanding of vascular anatomy are critical to avoid devastating complications. Furthermore, these procedures require an intricate combination of visual and tactile feedback; a good comprehension of each individual pathology; and extensive training periods. Besides, recent studies have demonstrated that risks diminish with the skill of neuroradiologist (Singh et al. (2002)). Whereas no animal has a brain structure close to humans, classical curricula – such as training on cadavers – do not reflect a real experience since no blood flow comes into play.

Nowadays, a huge interest has arisen for educational simulations. These provide a boot strap between classroom learning and real-life clinical experience. Sophisticated simulations, similar to aviation courses, may rely on computerized mannequins providing the basis for medical simulations.

1.2 Medical simulations

According to the Society for Simulation in Healthcare⁹: *“Simulation is the imitation or representation of one act or system by another. Healthcare simulations can be said to have four main purposes – education, assessment, research, and health system integration in facilitating patient safety”*.

of part of the vessel wall (<http://www.aans.org>). Most cerebral aneurysms are present without any symptoms and are small in size (less than 10 millimeters in diameter). Smaller aneurysms may have a lower risk of rupture. A ruptured cerebral saccular aneurysm is the most common cause of SAH

⁶An AVM is a tangle of blood vessels in the brain or on its surface which bypasses normal brain tissue and directly diverts blood from the arteries to the veins

⁷http://en.wikipedia.org/wiki/Seldinger_technique

⁸It was invented by Italian interventional neuroradiologist Dr. Guido Guglielmi. (http://en.wikipedia.org/wiki/Guglielmi_detachable_coil)

⁹<http://ssih.org>

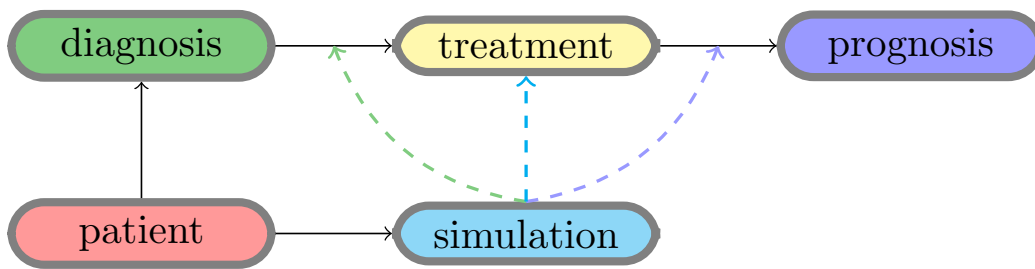


Figure 1.1: Medical pipeline: first, the pathology is diagnosed (diagnosis); then it is treated (treatment) and finally, the physician follows the patient during its convalescence (prognosis). Medical simulations may help the physician to rehearse before the intervention (green arrow), to transfer his/her knowledge to residents (cyan arrow) and to predict possible complications after the intervention (purple arrow).

It is true that simulation based learning is fast becoming an important tool to teach the integration of the knowledge of human anatomy and physiology with clinical skills in a safe environment and reproducible challenges. This approach allows for the early application and evaluation of techniques and skills in a less threatening environment before the patient factor comes into play. This increasing interest for medical simulations is due to several reasons: medical simulations may be used to train medical professionals to reduce accidents during surgery, prescription, and general practice; educational purpose in healthcare since the practitioner is free to make mistakes and to learn from them; big firms see medical simulations as a friendly way to disseminate techniques; Singh et al. (2002) showed – in an aneurysm coiling context – that the success of the intervention increases with the physician’s skill and consequently, the by-product is a reduction of time for operation and hospitalization.

Important considerations should be taken when designing a simulator for a specific application (Fig. 1.1). For the education purpose, the simulation must be *realistic* but not necessarily *predictive*, whereas the need for an absolute **Real-Time (RT)** execution is compulsory. In the area of *planning*, the simulation must be *accurate* enough to be predictive and its computation must be fast enough to be compatible with the medical pipeline but not necessarily **RT**. In contrast, when considering intra-operative guidance, the simulation must be also both accurate and **RT**. In fact, when referring to planning and intra-operative guidance, the simulation must take into account real patient data which is sometimes obtained in **RT** on the patient. Further interest on patient data simulators are found in the context of education where real cases offer real life challenges but above all, increase proficiency of trainees (Singh et al. (2002)).

In these three cases, the common denominator is models – for the interventional tools (i.e. guide-wire, catheter or micro-catheter) and the vasculature – complying to the application’s specific constraints. Besides, not only the modeling of virtual objects is a challenging task; other key-points such as user-interaction, interaction between virtual objects, and the patient-specific geometrical model availability bring about further problematics.

In the next sections, we briefly review current commercial simulators for **IN** for providing then a current state of challenges related to medical simulations in **IN**.

1.2.1 Endovascular procedure simulators

Many procedures start with a needle insertion into the vascular system but current commercial simulators skip this step to reduce complexity and build cost (Coles et al. (2011)). The navigation through the vascular anatomy is done using fluoroscopic guidance and tactile process, sensing small axial forces and torques at the fingertips while manipulating the interventional tools. Hereafter, we briefly survey current state-of-the-art on endovascular simulators. Table 1.1 recaps the existing products as well as works allowing their development.

Early prototypes for IN simulators were the Dawson-Kaufman for practicing angioplasty (Meglan (1996)) and the daVinci/ICard simulator (Anderson and Raghavan (1998); Anderson et al. (2002)). Mentice AB developed **Vascular Intervention Simulation Trainer (VIST)** from the called **Interventional Cardiology Training System (ICTS)** (Cotin et al. (2000)). Luboz et al. (2009) proposed a more recent simulator, the so-called **Vascular Surgical Platform (VSP)** device from Mentice for catheter and guide-wire manipulation coupled with a hydraulic system for palpation. Its purpose aimed at training the Seldinger Technique for catheter insertion, which covers the initial steps of introducing a guide-wire and catheter into the patient. In Li et al. (2001), NeuroCath is presented as a three component system: vascular extraction and modeling, instrument navigational simulations (further treated in Cai et al. (2003)), and human-computer interfaces. NeuroCath addresses the training and patient-specific planning of IN procedures and was further improved in Ma (2007). Lately, Wu et al. (2005) introduced **Real-time Endovascular Simulator (EVE)** as a neuroradiology training simulation framework (Wu et al. (2013)). Some features of the simulator include interactive fluid dynamics of blood flow (Wu et al. (2007)); and RT collision detection and collision response (Dequidt et al. (2007)). A branch of EVE was further developed by Inria under the **Simulation Open Framework Architecture (SOFA)** platform as an **Endovascular Embolization Simulator (EVE)** (Duriez et al. (2006)). Efforts were mainly focused on computation efficiency and improving contact response with the blood vessel surface (Dequidt et al. (2009)). In Wei et al. (2012), fluid dynamics of blood flow coupled to coiling were explored in near RT. Incidentally, commercial simulators such as CathLabVR¹⁰, ANGIO mentor¹¹, VIST¹², and COMPASS¹³ propose similar fluoroscopic guidance for interventional endovascular procedures. Last but not least, the HERMES project (Raspolti et al. (2005)), formerly created for coronary stent implants training, proposes soft-tissue modeling of the artery (Aloisio et al. (2006), catheter insertion (Aloisio et al. (2004)) based on the **Catheter Instruction System (CathI)** (Höfer et al. (2002); Rebholz et al. (2004)), and a mannequin on an operating table.

1.2.2 Challenges

Medical simulators have experienced a rapid development for the past 10 years but are still not enough advanced to tackle all the existing challenges related to computer-based

¹⁰CAE healthcare (<http://caehealthcare.com>)

¹¹Simbionix (simbionix.com).

¹²Mentice AB (mentice.com).

¹³Simsuite (medsimulation.com)

Simulator	Manufacturer	Works
Dawson-Kaufman	HT Medical	Meglan (1996)
daVinci/ICard	Intuitive Surgical	Lim et al. (1998) Anderson and Raghavan (1998) Anderson et al. (2002)
Vascular Surgical Platform (VSP)	Mentice AB	Luboz et al. (2009)
Vascular Intervention Simulation Trainer (VIST)		Cotin et al. (2000) (Interventional Cardiology Training System (ICTS))
NeuroCath		Nowinski and Chui (2001) Li et al. (2001); Cai et al. (2003) Ma (2007)
RT Endovascular Simulator (EVE)	SIM group	Wu et al. (2005, 2007, 2011) Cotin et al. (2005) Lenoir et al. (2006) Dequidt et al. (2007)
SOFA Endovascular Embolization Simulator (SOFAEVE)	Inria	Duriez et al. (2006) Dequidt et al. (2009) Wei et al. (2012)
HERMES	HERMES project	Raspolli et al. (2005) Höfer et al. (2002) (CathI) Rebholz et al. (2004) (CathI) Aloisio et al. (2004) Aloisio et al. (2006)

Table 1.1: A brief summary of current endovascular procedure simulators and related works.

simulations. An ideal medical simulator is supposed to provide interaction between the physician and the patient's anatomy in a clinical realistic manner. A bad simulator is worse than no simulator at all, and one that imparts bad habits is dangerous (Dawson (2006)). Realism means not only the visual aspect of virtual objects but also haptics which leads to modeling the mechanical properties, texture, color and even the appearance of real/soft objects.

The targeted application of this work is the aneurysm embolization simulation. Therefore, we orient related challenges toward this direction. Thereafter, we briefly review particular points in physical and geometrical modeling, as well as visualization and virtual object interaction to conclude by evaluation and validation of simulators.

Physical modeling

No matter in which aspect the problem is addressed, human body is complicated. The task is even more complex when its mechanical properties are needed to be known. More precisely, there exists a technical challenge when measuring kinematics, biomechanical, physiological and physical parameters. Furthermore, even if we measure these quantities, every human being will present different measures. Nowadays, patient-specific data is considered as a synonym of realism in numerical simulations (Gould et al. (2012)). However, realism is just merely subjective since it translates our impres-

sion about emulation of reality. In this domain, its quantification remains a difficult task. New trends lead to solutions addressing the problem as a comparison between virtual and clinical outcomes, followed by a correction of the simulation parameters but these solutions still reside at embryonic stage. Besides, it is always possible to improve realism by introducing a finer granularity in the modeling to detriment of computational efficiency.

No decision has a more profound impact on the eventual usefulness of a system than the concept of *physics based design* (Dawson (2006)). From a practical standpoint, a large part of the realism of a simulation, in particular for surgical simulation relies upon the ability to describe soft tissue response during the simulated intervention (Duriez (2013)). More specifically, for simulating interventional radiology interventions, the interventional instruments (needle, catheter, coils...) must also be modeled as deformable. Considerations about the availability of geometrical models are discussed in the following section.

Geometrical modeling

Despite a vast literature on the subject (Lesage et al. (2009)), patient-specific geometrical model recovery is still an active area of research and remains one of the major limitations of simulators. For instance, current commercial simulators such as ANGIO mentor and CathLabVR propose limited number of cases (respectively 100 and 30 cases) whereas VIST only provides a VIST Case-It as a segmentation integration tool for certifying *home-made* patient specific **three-Dimensional (3D)** reconstructions¹⁴. Effective simulation creates more than a sophisticated video game where the level of difficulty is fixed. To this point, the human arterial network puts forth extremely complex patient morphology and thus, complicated cases for treatment. Owing to this complexity, brain vasculature segmentation algorithms may fail at capturing pathologies, like cerebral aneurysms due to their small size and complexity of the neighboring arterial network; and tortuous and tiny vessels. This difficulty is stressed by the **IN** simulation context, particular requirements must be met by geometrical models such as, the availability of topological information (Li et al. (2012)). Procedural simulations must evolve from training to more elaborated stages such as planning and prognosis. For this purpose, there is a need to provide accurate representations of the vasculature to increase the fidelity and prediction of simulation when compared against real interventions. Every physician has the same 24-hours day, so procedure rehearsal will have to prove its worth to many skeptical users before it becomes a routine clinical event (Dawson (2006)).

Visualization

Challenging task for the visualization field are mainly rendering and the visual immersion. The rendering speed of one image is directly related to the complexity of the virtual scene and consequently, to interactivity; for example, zoom and camera rotation must be executed in **RT**, which could happen in the navigation of medical instruments and

¹⁴www.mentice.com/vist-case-it

other simulated processes. When deformations of the vascular model are also considered, a naive rendering refreshment of the whole vasculature may imperil this **RT** constraint. Therefore, an increasing need of dedicated rendering algorithms is ineluctable. Furthermore, efforts must be made to create high-quality visuals that effectively convey the real world to the user. Realism could be achieved in the field of data visualization by using appropriate rendering, including colors, textures, brightness, shadows, reflection, transparency, refraction, diffraction, etc.

Interaction

In the case of aneurysm coiling and most simulations, other than producing an appealing rendering, interactions between medical devices and anatomical structures are the core of the simulation. By this, we mean that detection of collisions; and response to deformation and collision are to be handled for all the objects composing the virtual scene. Once these steps managed, the final state of the virtual objects are visually and haptically rendered. Furthermore, it is sometimes compulsory to add a user-interaction to all these computations. Besides, the acute training of rod guidance and the response to the fine forces felt while advancing a rod is crucial for efficient **IN** procedures. An overexertion of force can have serious consequences and correct training to prevent this must be included in any **IN** training simulation (Coles et al. (2011)). Against this background, a (near or) **RT** computation is required which cannot always be guaranteed by modern calculators with both limited working frequency and memory. Otherwise, computation speed gain in collision detection and contact response computation demand optimized algorithms fully exploiting the actual hardware capabilities. Each of the above mentioned steps is an area of research by itself.

Evaluation and validation

Evaluation of procedural simulations is nontrivial. The availability and development of tactile interfaces is still in its infancy. The question of appropriate simulator metrics for the use of haptics remains open. Formal validation studies that focus on the use of haptics in medical simulation are scarce (Coles et al. (2011)). Haptic validation must fill this gap by fully understanding the perceptual mechanism of the human body.

1.3 Outline and contributions

In this work, we address the problem of the availability of geometrical models from patient data for **RT** or near **RT** computer-based simulations. More precisely, this work aims at providing accurate geometrical model that may be further used for planing or intra-operative guidance purposes. Against a coil embolization background, the proposed geometrical model alleviates computational burden during interactions between medical devices and the blood vessel surface.

We explore existing methods for capturing the vascular geometry whether reconstructing directly or segmenting the vasculature with regard to a **RT** simulation context (Chapter 2). After this brief review of state-of-the-art, we conclude that for capturing correctly the complex vasculature, the best geometrical model that fits our application

is implicit surfaces. However, methods providing such a model often relies on discrete grids for computation leading to local inaccuracies when considering a global reconstruction. At the end of our analysis of the current literature, we opt to capture locally the blood vessel as light models for computation efficiency (compact representation) and improving accuracy (precise description of the blood vessel surface). For that, we decide to rely on tracking algorithms which have the advantage of capturing locally the vessel surface while retrieving the vascular topology. This latter is capital for correctly handling and speeding up collision detections (Li et al. (2012); Wu et al. (2013)) since it provides a natural decomposition of the vasculature. Finally, we decompose the problem of geometrical reconstruction from image data in two steps: segmentation and then reconstruction.

The segmentation of the blood vessel surface relies on a novel tracking algorithm (Chapter 3). This paradigm uses cylinders to locally estimate the local vesselness (gravity center, local width and direction of the vessel) and points at the vessel surface are extracted via a robust stochastic filtering process. Therefore, our proposal provides a dense sampling of the vessel surface and a cylinder chain list as the centerline of vascular tree. The robustness, tracking capability and accuracy of our tracking procedures are compared against state-of-the-art **Multiple Hypothesis Tracking (MHT)** on 10 **3D Rotational Angiography (RA)** patient data.

The reconstruction algorithm (Chapter 4) provides an implicit surface representation for the previously extracted tree. For each cylinder on the centerline, a local reconstruction – driven by an energy minimization formulation – is fulfilled which provides an accurate implicit representation of the local blood vessel surface grounded upon the extracted points. Consequently, the blood vessel surface is reconstructed as tree of local implicit surfaces. The strength of the modeling process is shown on synthetic and real data.

In order to assess the efficiency of the proposed geometrical model, we applied our rigid geometrical model to the SOFA framework¹⁵ (Chapter 5). Synthetic as well as real data was modeled with our reconstruction algorithm, thus producing a dedicated geometrical model for **RT** simulation. Computation efficiency tests and realism assessment were carried out with the proposed geometrical model and a classical triangular meshes.

Finally, we derive our conclusion and discuss about future efforts to further the state of our research (Chapter 6). Among our perspectives, we explore new methods for our tracking algorithm to handle missing bifurcations automatically, the local reconstruction of the blood vessel surfaces directly from image data, the deformation of the tree of local implicit models and inherent discontinuity issues to our local modeling.

Our contributions are registered to two different research areas, namely the segmentation and the surface reconstruction fields. To this end, we introduce several improvements to the state-of-the-art literature.

Geometrical model suited to RT simulation. We propose a novel geometrical model for blood vessels which presents nice properties for computer-based simulations: smoothness, fast collision detection, geometric distance function approximation and its gradient (for the contact response direction and amplitude computation).

¹⁵<http://www.sofa-framework.org/>

RT Simulation validation. In addition, all these features were tested in real patient data and synthetic data against the well-established polyhedral model coupled with up-to-date collision detection algorithms.

Computation efficiency. The outcome reveals that the proposed model excels in computation efficiency and allows realistic interventional motion.

Novel robust tracking algorithm. Our tracking procedure improves upon the state-of-the-art **MHT**. A detailed analysis and comparison with **MHT** are performed to understand the robustness and accuracy of our proposal. Within this context, we extensively carry out both a quantitative and qualitative assessment on 744 vessels.

Improved modeling framework. In the context of surface reconstruction, we present a new framework for surface fitting driven by energy minimization. This framework improves upon similar works ([Muraki \(1991\)](#); [Bittar et al. \(1995\)](#) and [Tsingos et al. \(1995\)](#)) in the area through an automatic paradigm which resolves the problem based on geometrical criteria.

A novel geometrical model for blood vessels. We propose a novel geometrical model based on parametric implicit functions – namely **Blobby Models (BMs)** – which grant the ability to analytically express the blood vessel surface (closed-form expressions). Furthermore, our proposal’s ability to fine capture the vessel surface is easily regulated by modulating the number of parameters controlling the implicit function or its geometric precision. The by-product is a rigid model of the vascular tree which is presented as a tree of local implicit surface and well-suited for **RT** simulations.

RESEARCH BACKGROUND

*In this chapter, we provide a short review of existing geometrical model recovery algorithms. In this study, we consider blood vessel segmentation algorithms and visualization models. Our analysis is driven by interactive simulation constraints for assessing the characteristics of vascular models proposed in the literature. To this end, two classes of shape recovery algorithms are studied: those algorithms recovering the vascular tree from the centerline and their counterparts which directly recover the vessel boundaries. A brief summary of routinely employed modalities for **Interventional Neuroradiology (IN)** prognosis, detection and treatment of vascular diseases is also provided.*

The segmentation of vascular structures from **three-Dimensional (3D)** images, and subsequently the reconstruction, is particularly a challenging task. Segmentation consists of partitioning an image into an object – i.e. a structure of interest – and a background – i.e. the remainder of the image volume. The challenge in performing vessel segmentation is due to the sparseness of data, and the possible presence of irrelevant signal (other tissues, artifacts or noise). Furthermore, anatomical properties of vessels are highly variable in size, appearance, geometry and topology, specially in pathological cases such as aneurysms, stenoses, calcifications or **Arteriovenous Malformations (AVM)**. The choice of a segmentation method is usually closely related to three major concerns:

1. the modality considered for acquisition, since there exist several kinds of angiographic data with different quality levels and resolution, and consequently with heterogeneous types of artifacts;
2. the type of vessels being delineated – i.e. arterial or venous networks differ depending on organs and their surround;
3. and the clinical purpose.

In this chapter, we briefly review the state-of-the-art in segmentation and reconstruction algorithms for vascular networks. Following a globally similar classification presented in [Suri et al. \(2002\)](#), we decompose roughly our guideline in two families of approaches for blood vessel segmentation and reconstruction: boundary and centerline-based reconstructions. Recent surveys for lumen vessel segmentation are given in [Kirbas and Quek \(2004\)](#); [Lesage et al. \(2009\)](#). Attention is also paid to the relative performance of various surface modeling methods for vascular segmentation in [Bühler et al. \(2003\)](#); [Wu et al. \(2013\)](#). Together with the segmentation of the vessel anatomy, we also explore visualization techniques of vasculature [Preim and Oeltze \(2008\)](#).

For the remainder of this analysis, we drive our argumentation according to interactive simulation constraints and requirements. As stated in [Teschner et al. \(2005\)](#), high level of performances at interactive rates in numerical simulations may be obtained with suitable geometrical models. Henceforth, we also give importance to the resulting geometrical model or representation.

Recent works ([Dequidt et al., 2009](#)) and [Wu et al. \(2011\)](#) attest the feasibility of **Real-Time (RT)** or near **RT** simulators for **Interventional Neuroradiology (IN)** procedures. According to its future use, the geometrical model should meet the following characteristics to comply with interactive simulations:

1. collision detection needs to be accurate, fast and efficient without hampering the computation time ([Teschner et al. \(2005\)](#) and [Kockara et al. \(2007\)](#));
2. smoothness is compulsory to prevent jerky motions and to ensure realism ([Dequidt et al. \(2009\)](#));
3. for solving collisions, the model should facilitate a rapid computation of contact forces, as well as a prediction of possible collisions by supplying respectively the

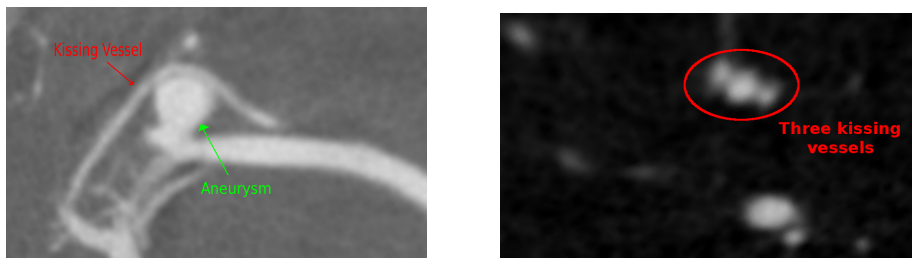


Figure 2.1: Two cut planes through 3DRA data exhibiting the **Kissing Vessel (KV)** issue: (left) A vessel runs along an aneurysm. (right) Three locally tangent vessels.

gradient and the distance to the surface (Teschner et al. (2005) and Kockara et al. (2007));

4. and the model must encode the topology information to disambiguate problematic cases (Jin et al. (2001); Wu et al. (2011) and Luboz et al. (2013)).

Problematic cases are situations where two vessels may happen to be locally tangent, or a vessel may run along a dense structure, e.g. an aneurysm or bone. Hereafter we refer to them as the **Kissing Vessel (KV)** issue. Fig. 2.1 displays sample images illustrating these cases. To understand the impact of **KV** cases on the simulation process, we must dive into an elementary simulation time step. During a simulation time step (Dequidt et al. (2009)), mechanical forces are applied to the catheter, modeled as a set of connected points, in a contact-free environment. Then the vessel geometry is added to detect points that crossed the vessel wall. Contact forces are then applied to these points before a new resolution loop is performed. No contact is detected at a point on the catheter if it is inside the vasculature at two consecutive simulation time steps although it could jump in between from two **KVs** as illustrated in Fig 2.2. Increasing the simulation step will slow down computation but may be useless when facing up to small gaps between **KVs**. In short, a simple inclusion/exclusion test is insufficient to detect the contact. Conversely, topology informs whether the catheter remains in the same branch or glide into a topologically far vessel.

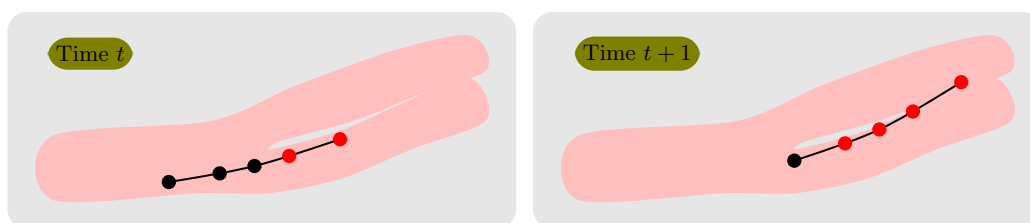


Figure 2.2: Toy scenario of a problematic case for two consecutive simulation time steps. The catheter is represented as five connected points where red color is related to possible/detected collision, otherwise points are in black. All points remain inside the same vessel (left). All points lay inside the vasculature but in two different vessels (right).

2.1 Boundary-based reconstructions

Boundary-based reconstruction techniques compute the vessel geometry directly from image data or segmentation results – e.g. a voxel soup. This kind of reconstruction is typical of deformable models. Among them, one of the most popular models is active contours. Active contours are curves or surfaces that deform within image data. They are classified as either parametric or geometric according to their representation and implementation (Xu et al. (2000)).

Parametric active contours or snakes have been used extensively over the last twenty years due to their simplicity (Kass et al. (1988); Klein et al. (1997); Mille and Cohen (2009); Mohan et al. (2010) and Wang et al. (2011)). The main advantage is that segmentation can be fulfilled without a great definition of the object boundary which can be computed iteratively (Suhuai Luo (2014)). Besides, they can capture both concave and convex features. However, they have two main limitations. First, when the targeted object and the initial contour differ greatly in size and shape, the model has to be dynamically reparametrized (Hegadi et al. (2010)). Second, one major downside of this model is that it has difficulty to deal with topological modifications such as splitting or merging (Delingette and Montagnat (2001)). To handle topological changes, McInerney and Terzopoulos (1999) introduced a sophisticated heuristic for reparametrizing a meshed contour/surface – namely T-snakes – in the area of retinal blood vessel segmentation. Klein et al. (1997) deformed a spline-based contour to detect coronary boundaries. Active contours were used by Mille and Cohen (2009) and Mohan et al. (2010) who proposed to combined shape-prior based snakes with minimal paths. The method relied on deformable cylindrical models which naturally fit to the boundaries while estimating the medial axis of vessels. A mesh-based active contour was used to smooth the resulting stack of cylinders and obtain the vascular tree segmentation. By the same token, Wang et al. (2011) used generalized cylinders coupled to a snake evolution scheme. Generalized cylinders were discretized into snaxels which in turn evolved according to vesselness energies.

Geometrical active contours or level set methods basic idea is to represent a contour as the zero level set of a higher dimensional function – the so-called level set function – the motion of the contour is formulated as the evolution of the level set function (Yan and Kassim (2006)). Level set frameworks based rely heavily on regular discrete grids for calculating their evolution (Gomes and Faugeras (1999) and Hegadi et al. (2010)) where the segmented object is encompassed by the zero level set. The main advantage of these methods is that they handle automatically topological changes. Due to this feature, these methods have been applied to the segmentation of cerebral pathologies (Hernandez and Frangi (2007); Scherl et al. (2007); Piccinelli et al. (2009); Bogunovic et al. (2011)) and blood vessel segmentation (Lorigo et al. (1999); Yan and Kassim (2006); Law and Chung (2007); Shang et al. (2011)). However, the computation of the level set evolution is time and memory consuming, and the segmentation may lead to false positives (Lingrand and Montagnat (2005)). For alleviating these issues, these methods are often initialized by a pre-processing step and when possible constrained with some prior. For example, in Manniesing et al. (2007) a level set was run twice. The first time, it provided

a rough estimation of the vasculature which served to compute the centerline. The second time, the level set used the centerline as a prior to further segment the vascular tree and avoid false positives. By the same token, [Frangi et al. \(1999\)](#); [Brian Mohr \(2012\)](#) and [Wang et al. \(2012\)](#) used the central vessel axis to constrain the evolution of the level set. In [Cebal et al. \(2005\)](#), a region growing segmentation and iso-surface extraction were used for initializing a deformable model for segmenting cerebral aneurysms and their neighboring vessels.

Discrete representations are heavily used in computer vision. Inherently, discrete representations are loose and allow to model complex geometries and arbitrary topology with no connectivity. Voxel soups (binary segmentation results) make up the most simple representation since they are closely related to the way data is represented – i.e. image or volume data are regular grids and the structure of interest is described by pixels/voxels composing it. In this context, morphological methods relying on basic operations (namely erosion, dilation, opening, closing), involving geometric patterns (structuring elements), allowed to segment 3D vessel vasculature ([Zana and Klein \(2001\)](#)) or served as pre-processing stages for other segmentation methods ([Alhonnoro et al. \(2010\)](#) and [Wu et al. \(2011\)](#)). High-level image processing techniques, based on these mathematical morphology operations, have been developed, e.g. watersheds have been employed for 3D vessel segmentation ([Passat et al. \(2007\)](#)). [Dufour et al. \(2013\)](#) presented an up-to-date survey on current mathematical morphology advances in 3D angiographic segmentation. Also exploiting the elementary information, one can find region growing approaches ([Masutani et al. \(1996\)](#); [Quek and Kirbas \(2001\)](#); [Flasque et al. \(2001\)](#); [Hoyos et al. \(2006\)](#) and [Carrillo et al. \(2007\)](#)). Region growing algorithms have small calculation complexity and high speed which make them a widely used method in medical image segmentation ([Gómez et al. \(2007\)](#); [Bock et al. \(2008\)](#); [Freiman et al. \(2009\)](#); [Alhonnoro et al. \(2010\)](#); [Jiang et al. \(2013\)](#)). The basic idea of traditional growth region is to collect pixels/voxels that have similar properties together to form a region. However, its performance depends largely on the position of seed points and growth conditions. Seed points are often manually supplied ([Freiman et al. \(2009\)](#)) but automated processes for vascular structures have been introduced in [Bock et al. \(2008\)](#) and [Jiang et al. \(2013\)](#).

2.1.1 Discussion on geometrical models

Boundary-based reconstructions algorithms often produce voxel soups or mesh-based structures of the vessel walls.

Voxel soups offer a loose representation of an object and could be interesting for predicting collisions during simulation via distance fields. The evaluation of distances and normals needed for collision and response is extremely fast and independent of the geometric complexity of the object ([Cornea et al. \(2007\)](#)). Collisions could be solved by embedding the binary mask in a discrete regular grid of the space. On the other hand, a mere estimation of the vessel boundaries is inconceivable in our context because a coarse staircase-shape representation leads to jagged surfaces and subsequently to unrealistic simulations. To alleviate this issue, interpolation methods can be employed in a post-processing stage which are discussed in the following paragraphs.

Other methods based on Delaunay triangulation, require an initial voxel soup or a centerline coordinate image, local radius and connectivity, provided pleasant results. Typically, these procedures work in a two step fashion. First, a pre-processing stage is necessary to reconstruct a rough approximation of the vasculature. These approaches routinely produce a coarse mesh. Here, topology problems, i.e. branches correspondence, are managed. Finally, this mesh is procedurally smoothed and refined using mesh-based subdivisions such as Catmull-Clark (Felkel et al. (2004); Wu et al. (2011); Hijazi et al. (2010)) which ensures a C^2 continuous surface almost everywhere. Other refinement algorithms were successfully employed for generating pleasant results. Ou and Bin (2005) used Loop subdivisions and Bornik et al. (2005) drove an iterative simplex mesh deformation based on Newtonian laws. The main advantages of subdivisions surfaces is that they yield smooth surfaces and simple heuristics.

Meshed representations owe their notoriety to an impressive power evolution of hardware graphics. Indeed, it appears that a vast literature on collision detection is grounded on meshed representations since most **Graphics Processing Units (GPUs)** are capable of handling millions of triangles for display. However, performing collision detection with a huge amount of objects still is an active area of research (Kockara et al. (2007); Avril et al. (2009)). In an **IN** simulation background, the interventional tools, e.g. guide-wire, are line-shaped and they tend to follow the valleys formed by successive edges, hence producing unwanted friction and jerky motions. Conferring smoothness – using subdivisions schemes – on polyhedral representations helps but at the expense of a high number of polygons which may put at risk the real-time performances.

Special consideration must be giving to the **KV** issue. In fact, there is no way to disambiguate this scenario while using voxel soups or meshed representations unless resorting to dedicated machinery. A major disadvantage of these representations is the lack of global topology information, at least without having recourse to a post-processing step. Even still, vessels may actually physically touch, leaving absolutely no hint of a gap between **KV** in the image data. Genuine topology retrieval algorithms cannot handle such case without further priors. Last but not least, the interventional tool may lie outside the vasculature during simulation. Whereas these representations allow differential estimations only on the *vessel boundary*, the gradient of the distance is required for finding a direction and a recall force in order to bring back the tool or part of it inside the vasculature while computing collision responses.

Implicit surfaces are smooth and provide high modeling capabilities when dealing with a high variety of topologies. Implicitation methods providing an implicit representation from voxel soups or polygonal surfaces have gained increasing interest (Preim and Oeltze (2008)). Implicit representations may enable fast collision detection through cheap inclusion/exclusion tests. To this end, many valuable methods employ convolution surfaces (Bloomenthal (1985); Oeltze and Preim (2005)), **Multi-level Partition of Unity (MPU)** implicits (Braude et al. (2007); Schumann et al. (2008)), and Poisson surfaces (Wu et al. (2010)). Nevertheless, the blood vessel surface is over-smoothed when using convolution surfaces. With this in mind, Schumann et al. (2008) proposed to use **MPUs** for a precise delineation of the vessel borders. However, **Bounding Volume Hierarchies (BVH)** are inefficient for detecting **KVs** and thereby, branches tended to merge. In that area,

Poisson surfaces may be resilient to these situations but point-sets provided with normals are mandatory. Therefore, a normal estimation from voxel soups is one possible solution which may however lead to inaccuracies during reconstruction.

2.2 Centerline-based reconstructions

A vessel can be seen as a 1D curve that is centered inside the vessel and coated with “skin”. Indeed, the vasculature centerline encodes significant vessel features: curvature, torsion, tortuosity and topology (Piccinelli et al. (2009)). Furthermore, the main interest for the centerline is that it reduces the problem to a one-dimensional clean and noiseless representation for blood vessels. Axis-based vascular segmentation algorithms provide an alternative means to kill two birds with one stone: vessel topology is extracted segment-by-segment while the vessel boundary associated with each segment is delineated (Wong and Chung (2007)).

Although, this representation of the vasculature seems natural, centerline extraction from image data or voxel soups is an extensive area of research and has bred a vast literature (Cornea et al. (2007)): topological thinning (Frangi et al. (1999); Bouix et al. (2005)), distance maps (Li et al. (2009); Mohan et al. (2010)), Voronoi diagrams (Piccinelli et al. (2009)) and tracking methods ((La Cruz et al., 2004; Tyrrell et al., 2007; Worz and Rohr, 2007; Friman et al., 2010)).

Tracking procedures follow the vessel centerline directly on image data. Overall, tracking methods perform from a given position on the centerline, a prediction step – i.e. the next candidate position on the centerline is predicted – and a correction step – i.e. the estimated positions of the centerline are recentered through model-based and/or image-based features.

Variants in the input have been required for different frameworks. In Wink et al. (2000); Wesarg and Firlle (2004); Li and Yezzi (2006) and Gülsün and Tek (2008), the user supplied two seed points and the tracking computed the minimal path between these two points; whereas Quek and Kirbas (2001) and Deschamps and Cohen (2002) propagated a wave that follows a manual path from one starting point in the root of the vascular tree. Others, like Tek et al. (2001); Hoyos et al. (2006); Carrillo et al. (2007); Manniesing et al. (2007) and Wong and Chung (2007), employed a single seed for initializing the tracking procedure which can be provided by a dedicated method (Tyrrell et al. (2007); Jiang et al. (2007); Zambal et al. (2008); Yedidya and Hartley (2008)). Besides, the start-point may be enriched with further information to constrain the vessel-axis search, e.g. a vessel radius estimate (Schaap et al. (2007)) and/or a tracking direction (Flasque et al. (2001); Worz and Rohr (2007); Yedidya and Hartley (2008); Friman et al. (2010)).

Model-based estimations employ geometric analysis to determine directly the vessel shape (center, radius and direction). For instance, Tyrrell et al. (2007) and La Cruz et al. (2004) fitted superellipsoids and cylinders to image data. Gaussian assumptions were successfully integrated for characterizing the vessel lumen as a cylindrical profile (Worz and Rohr (2007); Schaap et al. (2007); Jiang et al. (2007); Yedidya and Hartley (2008); Friman et al. (2010)), while the Hessian matrix of clustered voxels (inertia moments and

curvature), belonging to the vessel, mimicked a cylindrical model (Flasque et al. (2001); Hoyos et al. (2006) and Carrillo et al. (2007)).

Image-based methods exploit image features such as the gradient and Hessian of the image intensity. In this family, the correction step may be dissociated from the prediction step. Indeed, Wink et al. (2000); Tek et al. (2001); Wesarg and Firlle (2004); Gülsün and Tek (2008) and Zambal et al. (2008) used ray-casting grounded on the gradient to capture the vessel walls, coupled with medialness filter response to compute the vessel center along cross-sections. The prediction step may take different forms to compute the direction for translating the current position: Hessian matrix analysis (Aylward and Bullitt (2002); Wong and Chung (2007); Alhonnoro et al. (2010) and Bauer et al. (2010)), depth search (Zambal et al. (2008) and Friman et al. (2010)) or the reuse of the current estimated direction (Wink et al. (2000) and Jiang et al. (2007)).

Solutions for recovering the vessel walls while tracking the vessel axis have been presented in (Flasque et al. (2001); Carrillo et al. (2007)) where the vessel segmentation result is composed of clustered voxels belonging to the vessel of interest. Owing to its simplicity, another idea that have received much interest is ray-casting. In 2D, ray-casting was used for characterizing the vessel cross-section (Wink et al. (2000); Tek et al. (2001); Wong and Chung (2007); Gülsün and Tek (2008)). First, contour points are selected and then, instead of relying on a circular or elliptic model, they assume a regular, compact cross-sectional contour for estimating vesselness measures. The major advantage of ray-casting is that it reduces the detection problem to a 1-D analysis along the ray and in practice, it breeds fast computational algorithms. Threshold over the image intensity value (Wesarg and Firlle (2004)) or its gradient (Wink et al. (2000); Tek et al. (2001)) are the basis for probing points at the vessel boundaries. Nonetheless, ray-casting-based techniques do not prune candidates points at the lumen boundaries. Instead, they tend to minimize the centerline position estimation error while increasing the number of rays. An alternative to averaging is to use a filtering technique to prune candidate points, e.g. Tek et al. (2001) employed mean-shift filtering; Schaap et al. (2007) and Yedidya and Hartley (2008) used Kalman filters.

2.2.1 Discussion on geometrical models

A classical approach in this family of models is to represent the vessel walls as 2D cross-sectional contours sweeping along the centerline. A strong common form of extraction assumes that the vessel can locally be described as a tubular segment, and thereby the centerline curve serves to model the vessel surface as a generalized cylinder. For instance, Bloomenthal (1985), Oeltze and Preim (2005), Gerig et al. (1993) and Tian et al. (2006) considered the surface as a tree of generalized cylinders with a circular cross-section (disk) of varying radii. Following the same idea, Hahn et al. (2001) represented each position on the centerline as a cone. However, these assumptions result in unfaithful representations of the underlying vascular structures (Worz and Rohr (2007)), especially pathological structures whose cross-sections present irregular shapes.

More precise implicit representations of the vessel cross-section have been studied in Hong et al. (2012) and Kretschmer et al. (2013). The overall idea is to describe finely

the cross-sectional lumen contour and then sweep all contours along the centerline. As a result, the blood vessel surface maintains a 3D coherence while capturing the true variations of the vessel surface. The small downside of the approach is that only the segmented contour for each image slice is regarded. In contrast, Pizaine et al. (2011) performs interpolation in between with regard to patient data.

2.3 Model choice

Boundary-based reconstructions put forth algorithms representing the global vascular surface. These algorithms often produce voxel soups and meshed representation of the blood vessel surface. In this context, many sound algorithms exist for detecting collision (Kockara et al. (2007); Avril et al. (2009)). Furthermore, topology information – namely the centerline – can be retrieved from these models for KV issues disambiguation (Frangi et al. (1999); Cornea et al. (2007); Piccinelli et al. (2009); Wu et al. (2013)). However, one major downside of these representations resides on smoothness; whereas a precise contact response is required for ensuring high realistic motions. Despite a high number of subdivisions methods, they only consider geometric criteria for the refinement process such as Catmull-Clark, thus no precise description with respect to (w.r.t) the patient data is ensured. Besides, increasing the number of polyhedra assuredly increases smoothness but at the expense of a high computational collision detection.

Implicit representations of the vasculature authorizes cheap inclusion/exclusion tests during collision detection. Furthermore, implicit surfaces may provide at least C^1 continuous description of the blood vessel surface. High order continuity is desirable for handling friction. When the implicit representation provides at least first order continuity, methods following the gradient descent of the implicit function gradient can retrieve the medial-axis (Ma (2007)). However, manipulation of implicit surfaces brings about blending issues which can lead to merge branches in the presence of KVs. Controlling blending – while preserving continuity of the scalar field – has proven to be extremely complex when more than two objects are merging (Bernhardt et al. (2010); Gourmel et al. (2012)). In a boundary-based reconstruction context, constraining a level set framework with the centerline may avoid unwanted blending but it may also lead to under segmentation of the vasculature. Thereby, a correction step may be necessary.

In this area, centerline-based methods are well-suited since prior shapes increase robustness against KV issues (Worz and Rohr (2007); Wong and Chung (2007) and Friman et al. (2010)). However, tracking methods consider the vessel cross-section as circular which leads to over smoothed representations of the vascular tree (Preim and Oeltze (2008)). Nevertheless, local modeling while tracking the vessel cross-section seems to provide accurate descriptions for visualization and quantification (Wink et al. (2000); La Cruz et al. (2004); Friman et al. (2010); Hong et al. (2012); Kretschmer et al. (2013)). The resulting surface can be obtained through sweep surfaces of each cross-sectional contour along the centerline (Hong et al. (2012); Kretschmer et al. (2013)). Despite a precise description of the cross-sectional vessel lumen, these methods do not consider image data in between cross-sections since they often rely on generalized cylinders for reconstruction. Moreover, they provide a global representation of the vascular network.

In an interactive context, global representations often come hand in hand with space partitioning machinery. Since interventional tools are slender, their motion can be defined by that of longitudinal nodes. With this in mind, one can benefit from a local description to handle independently each node during collision detection. To this end, we believe that a tracking algorithm providing the centerline and a local description of the blood vessel surface is the best approach that fits our purpose. In this area, ray-casting algorithms provide such a representation. Nevertheless, the local description – i.e. point-sets – of the blood vessel surface is not exploitable during simulation. Implicit surfaces put forth sound features in an interactive context and consequently, we decided to fit implicits to these local point-sets. By this mean, the centerline produced with a tracking procedure is enriched with an implicit description of the local vessel surface. **BVH** machinery is not necessary since the centerline positions provides automatically such a space partitioning. Moreover, topological information encoded on the centerline may be used to handle **KV** issues.

2.4 Modality choice

Hereafter, we propose to review some major modalities in a clinical environment. Point in fact, we explore **two-Dimensional (2D) Digital Subtraction Angiography (DSA)**, **Computed Tomography Angiography (CTA)**, **Magnetic Resonance Angiography (MRA)** and **3D Rotational Angiography (RA)** in the area of aneurysm diagnosis, prognosis and treatment. We drive this analysis to expose the main reasons leading our work to focus on **3DRA** modality.

In the context of coil embolization, **2D DSA** was traditionally considered as gold standard (McKinney et al. (2008); Hiratsuka et al. (2008)). This technique uses fluoroscopy and iodine-based intra-vascular contrast material that is injected via a catheter into the femoral artery through the abdominal aorta and thoracic aorta, and into the carotid. A major limitation is the necessity of repeated injections of contrast material which leads to a higher radiation dose to the patient. Last but not least, the procedure requires highly skilled and experienced operators.

In clinical practice, **CTA** and **MRA** have been the most frequently used non invasive diagnostic technique for the detection of intra-cranial aneurysms. One major advantage over **DSA** is the generation of three-dimensional information on the geometries of intra-cranial arteries and cerebral aneurysms. However, compared to **DSA**, **CTA** and **MRA** have a lower spatial resolution.

Recently, there has been growing interest in utilizing **3DRA** for the detection of intra-cranial aneurysms (Anxionnat et al. (2001) and van Rooij et al. (2008)). When compared to **DSA**, **3DRA** uses lower contrast material and produces much less radiation dose to the patient (Pedicelli et al. (2007)). Tomycz et al. (2011) showed that **3DRA** provides additional information in the case of small aneurysms (e.g. 1 – 3 mm) for which the sensitivity of **CTA** and **MRA** is lesser. Besides, Tomycz et al. (2011) claimed that **3DRA** still is the main tool to disambiguate erroneous image interpretations. For instance, situations where part of vessels diagnosed, in **CTA** or **MRA**, as aneurysms appeared to be loop vessels in **3DRA**. **3DRA** is clearly of higher quality than **MRA** and **CTA**, and continues to

appeal to practitioners for the diagnosis, assessment, treatment and prognosis stages of intra-cranial aneurysms.

As a result, this work copes with the segmentation and reconstruction of blood vessels from **3DRA** image data. We ground our choice of this modality in the fact that **3DRA** patient data is often acquired before, during and after the neuroradiological intervention. Last but not least, it provides the most accurate representation of the vascular network among other angiographic modalities.

Part II

Blood Vessel Surface Reconstruction

BLOOD VESSEL SEGMENTATION

*In this chapter, we propose our blood vessel tracking algorithm that 1) detects the vessel centerline; 2) provides a local radius estimate; and 3) extracts a dense set of points at the blood vessel surface. This algorithm is based on a **RAN**dOm **S**Ample **C**onsensus (**RANSAC**) based robust fitting of successive cylinders along the vessel. Our method was validated against the **M**uLtiple **H**ypothesis **T**racking (**MHT**) algorithm on 10 patients **t**hree-**D**imensional (**3D**) **R**otational **A**ngiography (**RA**) data. Over 744 blood vessel of various sizes were considered for each patient. Our results demonstrated a greater ability of our algorithm to track small, tortuous and touching vessels (94% **S**uccess **R**ate (**SR**)), compared to **MHT** (65% **SR**). The computed centerline precision was below 1 voxel when compared to **MHT**. Moreover, our results were obtained with the same set of parameters for all patients and all blood vessels. Tracking results on **M**agnetic **R**esonance **A**ngiography (**MRA**) also exhibited the strength and robustness of our method.*

3.1 Introduction

In the former part of this work, we revisited the state-of-the-art techniques for accurately capturing the lumen variations and recovering the topology of the arterial network. At the end of our analysis, we opted to use a tracking procedure – coupled with an implicit modeling approach – to recover both topology and geometry.

Our aim is to recover a representation of the arterial network walls and its topology from **three-Dimensional (3D) Rotational Angiography (RA)** patient data. The segmentation task, however, remains difficult when dealing with tiny vessels, whose radius is about the image resolution, **Kissing Vessel (KV)** issues, where no image gradient is available since they are physically close, and the complex vascular morphology, which exhibits complex topology that loops in a highly random fashion. In this section, we propose a geometric method for retrieving the vessel properties. The algorithm can be classified as a tracking vessel-axis algorithm.

The overall work flow of our algorithm is described in (Sec. 3.2, Fig. 3.1). Our method exploits ray-casting techniques (Sec. 3.3) based on gradient intensity to extract a set of points capturing the vessel local shape (Fig. 3.1b). The ray-casting stage presents a set of candidate points which presents points corrupted with noise and outliers¹. Then, the candidate points are robustly, geometrically filtered using **RANdom SAMple Consensus (RANSAC)** (Fischler and Bolles (1981)) and a cylinder model (Sec 3.4, Fig. 3.1d). In this fashion, the cylinder robustly fitted furnishes a local estimate of the center, radius and direction of the vessel and in the end, a dense sampling of the vessel surface. Section 3.5 describes the usage of our **RANSAC-Based Tracking (RBT)** algorithm for tracking a single vessel and section 3.6 exemplifies the handling of bifurcations in order to segment the whole vascular tree. Besides, section 3.7 informs about the *tracking* of aneurysms while in section 3.8, technical details about the implementation of the ray casting procedure is given. **RBT**'s strength is quantitatively and qualitatively evaluated on 10 **3DRA** patient data (Sec. 3.9 and Sec. 3.10). Finally, we discuss (Sec. 3.11) and conclude (Sec. 3.12) about our work.

3.2 RANSAC-based tracking (RBT) algorithm

In general, by setting a start-point provided with a possible direction of the vessel centerline, tracking algorithms refine this position as the point with the highest “likelihood-of-being-center” in the plane perpendicular to this direction. The direction may be simultaneously determined (La Cruz et al. (2004); Worz and Rohr (2007); Tyrrell et al. (2007); Jiang et al. (2007)), retrieved from image/geometric features (Flasque et al. (2001); Aylward and Bullitt (2002); Hernandez and Frangi (2007); Carrillo et al. (2007); Wong and Chung (2007); Yedidya and Hartley (2008)), pruning directions (Schaap et al. (2007); Zambal et al. (2008); Friman et al. (2010)) or supplied by another method which provides orthogonal planes in the vessel lumen (Tek et al. (2001)). Then, they predict the next candidate point in this direction (Bühler et al. (2003); Lesage et al. (2009)). Thus, the tracking progresses by successive estimation and prediction steps.

¹Points that do not lie on the vessel of interest

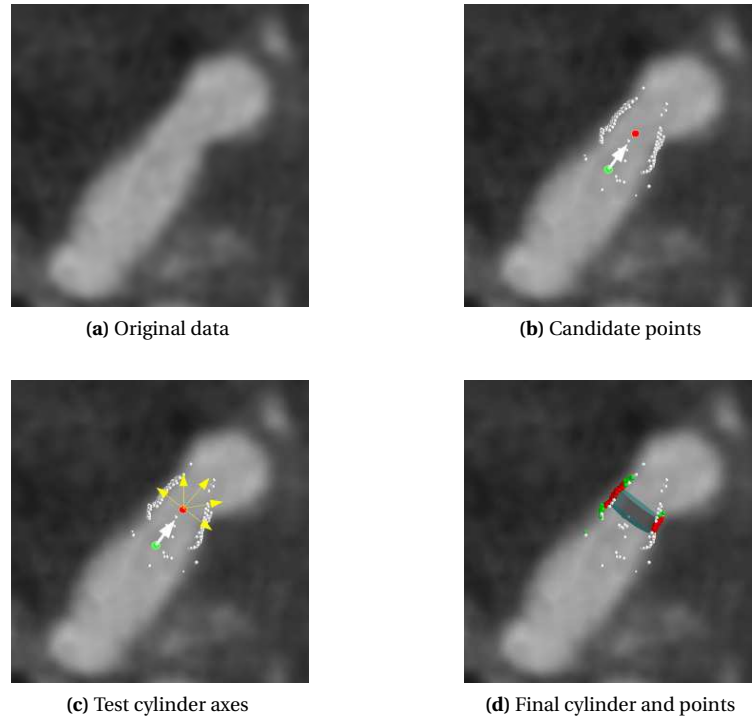


Figure 3.1: Outline of **RBT** vessel tracking algorithm. (a) Cut plane through the original **3DRA** data; (b) Parameter update: Starting from C_0 (green dot) and \vec{d}_0 (white arrow), the new center C (red dot) is found. $\vec{d} = \vec{d}_0$. Candidate points (white dots) are extracted at the vessel surface by casting rays in the volume, from C . Only the points around the displayed cut plane are shown. (c) N_d directions are tested for the cylinder axes (yellow arrows). One fitting cylinder is found using **RANSAC**, per axis. (d) Best cylinder found, together with its consensus set (green dots) and the final points used to set the cylinder height.

Our **RBT** algorithm also alternates between an estimation and a prediction step. The *estimation step* assumes that an estimate for the center C , axis direction \vec{d} and radius r of the cylinder are available. Candidate points are extracted at the vessel surface in the vicinity of C using a ray-casting procedure (Sec. 3.3). Then, a **RANSAC** estimation of the cylinder – that robustly fits this set of points – is performed (Sec. 3.4). The output of the estimation step is an updated cylinder of center C^* , axis direction \vec{d}^* , radius ρ^* , and height h^* . The set of points fitted by the estimated cylinder (inliers) is also furnished as an output. Finally, the *prediction step* considers C^* , \vec{d}^* , ρ^* and h^* to compute new values for C , \vec{d} and ρ for restarting the procedure (Sec. 3.5).

3.3 Ray casting

In **3DRA** data, the vessels are relative bright tubular-shaped objects, on a dark background. Let's consider a single ray cast from inside the vessel lumen (refer to Fig. 3.2). Then, along the ray, we compute the directional gradient. Looking closely, we observe that the minimum along the ray yields a location at the vessel border while the maximum exhibits a position on a neighboring structure. This simple technique avoids the usage of thresholds over the image intensity (Wesarg and Firlle (2004)) or its gra-

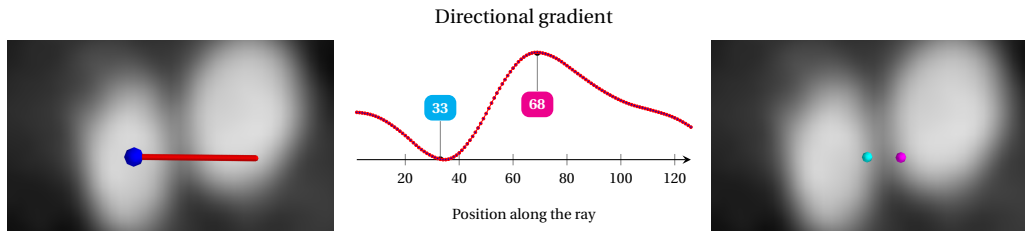


Figure 3.2: (left) A ray – of length four times the estimated vessel radius – is cast from inside the vessel lumen (blue dot). (center) The directional gradient along the ray is computed with central differences method. (right) In 3DRA data, the minimum along the ray characterizes the vessel wall.

dient (Wink et al. (2000); Tek et al. (2001)) and alleviates the problem of arterial signal variations. In other words, a ray-casting procedure combined with a minimum search along the directional gradient of each ray may provide points mostly at the boundaries of the vessel of interest. This extraction scheme is very similar to taking the points of minimal gradient along the columns of a **Bounded Spherical Projection (BSP)** image (Wong and Chung (2007)). As depicted in Fig. 3.1b, N_r rays, sampling evenly the space, cast from a point C , inside the vessel, yield points at the vessel borders and points inside the vessel. These latter positions exemplify *outliers* which might be whether inside the vessel, due to variations in the image intensity, or at neighboring structures. Nonetheless, an estimate of the vessel radius is compulsory to restrict the length of the ray. As pointed out by Gülsün and Tek (2008), large values of the ray length may produce strong boundary responses at locations which are outside the vessel. In this area, Carrillo et al. (2007) considered a maximal area of search of twice the radius of the vessel, in contrast to Wong and Chung (2007) who used 1.5 the current estimate of the vessel radius. However, situations where the radiation focus is close to a vessel wall, e.g. along an acute bend of the vessel, a length \mathcal{L} of twice the current vessel radius may be insufficient to recover points on the opposite wall (Fig. 3.3). In our case, to alleviate cases where the minimum along the ray yields positions on the neighboring structures or the ray-casting center is badly positioned at curved vessels, we bound the length of rays to 3 times the current

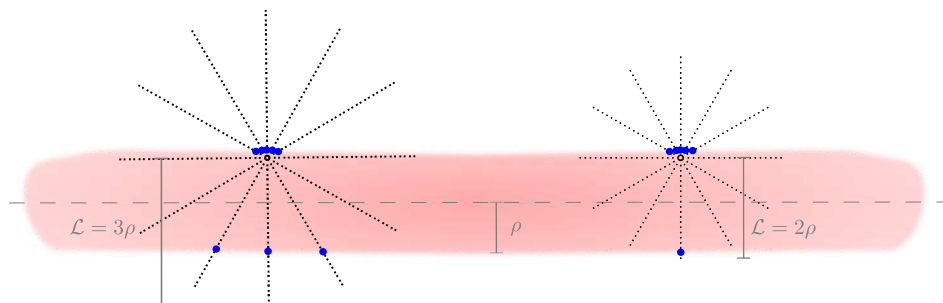


Figure 3.3: Two radiation focus and their extracted points (blue) on vessel of width ρ : rays have length $\mathcal{L} = 3\rho$ (left) and $\mathcal{L} = 2\rho$ (right). A ray length of twice the local vessel radius is insufficient to capture points at the opposite wall when the ray-casting center is close to a vessel wall.

vessel radius. This value is sufficient for **RBT** to adapt to abrupt changes in the vessel direction and width, i.e. pathologies and bifurcations, and to reduce the amount of outliers. Furthermore, **Worz and Rohr (2007)** and **Gülsün and Tek (2008)** used a maximum and minimum vessel radius to constrain the boundary search. Indeed, intensity variations may occur close to the vessel ridge (flow artifacts), hence we used a minimal bound to prevent points from agglomerating in the vicinity of the ray-casting focus. We deem that candidate points at the vessel wall should be at least at quarter of the current vessel radius estimate ($\mathcal{L}/12$). Similarly, maxima located further than 80% \mathcal{L} seem to indicate the presence of a neighboring structure (**KV** issue). Even though a closer minimum might correctly fit the vessel wall, the situation is avoided and the ray is discarded. There are obviously circumstances where this simple thresholding process discards inliers (see Fig. 3.3 where the most upper points are thus discarded) but in practice, the quality of the resulting point-set was consistent across a large panel of cases and enabled the subsequent **RANSAC** estimation to use a constant set of parameters (see Section 3.9 and 3.10).

In summary, **RBT** casts N_r rays of length \mathcal{L} from C . Then, the directional gradient along the ray is computed. Next, the minimum among the N_p points composing the ray is retrieved. Finally, points lying between $\mathcal{L}/12$ and $8\mathcal{L}/10$ make up the raw point-set \mathcal{P} (Alg. 1). Since variations in the image intensity are ubiquitous, the ray casting procedure may yield points inside the vessel lumen or points that do not belong to the vessel of interest, on both cases outliers. Unlike **Wink et al. (2000)** and **Gülsün and Tek (2008)** who constrained the ray casting directly on the image, we opted to filter the raw point-set according to a geometric criterion, namely **RANSAC**. This robust filtering technique is treated in the next section.

Algorithm 1 Tracking

Require: Cylinder (C, \vec{d}, ρ) .

- 1: Cast N_r rays of length $\mathcal{L} = 3\rho$ from C .
 - 2: Constitute the candidate points \mathcal{P} (minima along each ray and exclude points on extremities).
 - 3: $\{\mathcal{P}', C', \vec{d}', \rho'\} = \text{CYLINDERESTIMATION}(\mathcal{P}, N_d, \rho, p_{inl})$ $\triangleright \mathcal{P}'$: set of inliers.
 - 4: **if** (C', \vec{d}', ρ') is valid **then**
 - 5: $(C^*, \vec{d}^*, \rho^*) \leftarrow$ Refine parameters (Powell optimization).
 - 6: Cast N_r rays of length $\mathcal{L} = 3\rho$ from C^* . Keep inliers as \mathcal{P}^*
 - 7: Set h^* so as to encompass 75% of \mathcal{P}^* and update \mathcal{P}^* accordingly.
 - 8: **end if**
 - 9: **Stop** if (C', \vec{d}', ρ') is not valid or tracking turns back.
 - 10: **save** $(C^*, \vec{d}^*, \rho^*, h^*, \mathcal{P}^*)$
 - 11: Prediction of $(C = C^* + (h^*/2)\vec{d}^*, \vec{d} = \vec{d}^*, \rho = \rho^*)$. Go to step 1.
-

3.4 Estimation step: cylinder estimation

In this section, we describe the filtering process of points produced by our ray-casting procedure. First, we give a brief description about cylinders and their parametriza-

tion (Sec. 3.4.1). Second, robust cylinder fitting methodologies are presented and discussed (Sec. 3.4.2). Lastly, our geometric filtering approach is addressed (Sec. 3.4.3).

3.4.1 Cylinder parametrization

There are many definitions for the concept cylinder:

- *right circular cylinder* whose base is a circle and the centers of the sections form a straight line perpendicular to the base of the cylinder (Huysmans et al. (2005)). The length of the straight line is called height and the vector describing this line is designated by axis. When its height happens to be infinite, then it is called *infinite cylinder* and conversely, *finite cylinder* when the height is finite. For example, the infinite right circular cylinder whose axis is the z-axis is given by the equation on the xy-plane perpendicular to the z-axis: $(x/a)^2 + (y/a)^2 = 1$ where $a \in \mathbb{R}$ is the radius;
- *oblique cylinder* is a right circular cylinder whose top and bottom sections are displaced from one another, i.e. they are parallel but not orthogonal to the axis;
- *elliptic cylinder*, also called *cylindroid*, is a quadric surface and a generalization of the ordinary circular cylinder ($a=b$) (Tyrrell et al. (2007)): $(x/a)^2 + (y/b)^2 = 1$ where $(a, b) \in \mathbb{R}^2$;
- *generalized cylinders* are cylinders whose cross-sections can be any curve sweeping along an axis (a space curve) (Shafer and Kanade (1983)).

In this work, we choose the right circular cylinder to model the local shape of a blood vessel. It is represented by a center C , a unit-length direction \vec{d} , a radius ρ for the infinite version. An infinite cylinder can be described by 5 parameters, 4 parameters for the axis (any point sweeping along the axis is a valid center) and one for the radius. Fitting a cylinder to a set of points thus requires that at least five points – i.e. one point per parameter – are provided. To our knowledge, direct solutions to the case of 5 given points are unknown due to the lack of linear independent constraints (Christian Beder and Wolfgang Förstner (2006)). In circumstances where the cylinder axis is parallel to one coordinate plane, e.g. when it is vertical, the number of parameters decreases to 4, 3 parameters for the axis and one for the radius. A direct solution is proposed in Christian Beder and Wolfgang Förstner (2006). In case the cylinder axis is axis-aligned with one coordinate axis, e.g. the z-axis, the number drops to 3, two parameters for the position of the axis and one for the radius. This latter scenario can be seen as another problem, the one of finding a circle (center and radius) in a **two-Dimensional (2D)** plane. Christian Beder and Wolfgang Förstner (2006) resolved this problem by projecting three points onto the plane π orthogonal to the axis \vec{d} , which provides 3 **2D** points $\{(X_j, Y_j)\}_{j \in \{1, \dots, 3\}}$. Then the cylinder radius $\rho = \sqrt{c_x^2 + c_y^2 - u}$ and the center $C = (c_x, c_y)$ in π , are provided

by the following linear system:

$$\begin{pmatrix} 2X_1 & 2Y_1 & -1 \\ 2X_2 & 2Y_2 & -1 \\ 2X_3 & 2Y_3 & -1 \end{pmatrix} \begin{pmatrix} c_x \\ c_y \\ u \end{pmatrix} = \begin{pmatrix} X_1^2 + Y_1^2 \\ X_2^2 + Y_2^2 \\ X_3^2 + Y_3^2 \end{pmatrix} \quad (3.1)$$

A classical strategy to tackle the problem of fitting a cylinder to a set of points is to, first, determine the direction of the cylinder (3D problem) and then, the position and radius of the cylinder axis (2D problem).

In our segmentation context, we are interested in finite cylinders, hence an extra parameter corresponding to the height h is also required. Eberly (2008) proposed a formal representation of a finite cylinder:

$$(X - C)^T \frac{(I - \vec{d} \cdot \vec{d}^T)}{\rho^2} (X - C) = 1; \text{ with } |\vec{d} \cdot (X - C)| \leq h/2 \quad (3.2)$$

where I is the identity matrix.

3.4.2 Cylinder fitting

Our ray-casting procedure extracts at most one point per ray. Most of such points lie at the local vessel surface, even though this location is corrupted with noise. But a few points are outliers, either lying on the interface of a neighboring structure or thus emanating from inhomogeneities in the vascular signal intensity. Again, we desire to keep candidate points located at the vessel boundaries. Therefore, we may state the problem as finding two different populations: points considered as defining a vessel wall, the so-called inliers, and points lying elsewhere, designated as outliers. One applicable solution to our case, is to resort to a geometrical model since the blood vessels can be considered as tubes (Schaap et al. (2007); Carrillo et al. (2007); Worz and Rohr (2007); Tyrrell et al. (2007); Zambal et al. (2008) and Friman et al. (2010)). Indeed, our hypothesis involves that the vessel lumen can locally be assimilated to a cylinder of finite height. By considering this, the problem may be interpreted as finding the best cylinder that best fits the point-set. In a set of 3D points, one can find cylinders by surface fitting (Lukács et al. (1998)), i.e. a cylinder can be fitted using common nonlinear estimation techniques, e.g. the Levenberg-Marquardt algorithm (La Cruz et al. (2004)). Initial estimates are required, and the final results highly rely on the initial guess and the quality of the input data to overcome local minima. In our application where outliers are ineluctable, however, these methods are inconsistent with our purpose. In fact, classical techniques such as least squares (Eberly (2008)) – which optimize the parameters according to a specified cost function and all points – may fail since they have no internal mechanism for rejecting gross errors and instead, they tend to average it. Despite the fact that M-estimators have been introduced to counterbalance the light performances of least-squares, the outcomes heavily depend on the choice of the objective functions and without mentioning that in presence of outliers Least Median of Squares method is preferable but computationally expensive Zhang (1995).

In the field of primitive shape detection from range data, the extraction of cylinders is a common task (Vosselman et al. (2004) and Schnabel et al. (2007)). The two most widely

known robust estimators for this purpose are the Hough transform (Hough (1962)) and the RANSAC paradigm (Fischler and Bolles (1981)). Both have proven to be robust even in the presence of a high amount of outliers.

Owing to the discretization of the parameter space, a major drawback of the Hough Transform is its time and space complexity. In the case of a cylinder, proposed solutions demand a point-set provided with normals (Vosselman et al. (2004)) and the Hough transform possesses five dimensions in the parametric space but solutions for reducing the complexity have been proposed in (Rabbani and Heuvel (2005)). The idea is to decompose the problem in two stages (see Sec. 3.4.1). First, the axis is detected assuming that the cylinder normals form a circle on the Gaussian sphere (Gauss map). Second, by projecting all the point-set to the plane passing through the origin and orthogonal to the axis direction, the position and radius are computed as a regular 3D-Hough Transform.

The RANSAC paradigm is a stochastic approach. For estimating a cylinder, a minimal number of points – depending on the parametrization – are randomly selected which define an instance of a cylinder. The so constructed cylinder serves to count the number of points that are located within a certain distance r_t and then, suggest their compatibility. The random selection is repeated N_{\max} number of trials or until a user-defined threshold p_{inl} – used to imply that the correct cylinder has been found – is reached. RANSAC is usually combined with a refinement method and once it stops, the parameters and the set of compatible points are passed to a smoothing technique to compute an improved estimate of the parameters (Bolles and Fischler (1981)). In this context, Chaperon and Goulette (2001) used a similar framework as Rabbani and Heuvel (2005) but adapted to RANSAC. Nonetheless, the cylinder axis estimation from normals at input points may fail at curved sections for example. Also in our case, normals would be estimated as the image gradient at input points. Hence, such a procedure can lead to untrustful cylinder directions.

Though, both aforementioned robust techniques compute the direction and then, the complete cylinder. The determination of the cylinder direction remains a difficult task. Against this background, Lozano-Perez et al. (1987) explored the axis estimation, knowing the radius of the wanted cylinder and four to five points on the cylinder surface. Recently, Zambal et al. (2008) and Friman et al. (2010) offered an interesting strategy for finding a vessel direction that can be applied to our situation. They suggested to evenly sample the area of search – in front of the current tracking position and relative to the current vessel direction – and score all the directions for pruning them afterward. As a result, only a random sampling technique accommodates well with a pruning strategy of the cylinder directions. At this level, we opt to fit a cylinder using a similar strategy to Zambal et al. (2008) and Friman et al. (2010), and coupled with RANSAC for its robustness, simplicity and efficiency. In this fashion, one cylinder per direction is to be fitted but since the cylinder axis is provided, the next stage involves finding the center and radius from three points. This design authorizes a high computational efficiency of the cylinders parameters when complied to a pruning scheme. The main difference with the former proposals is that, instead of exhaustively scoring all directions by fitting a cylinder – which is cumbersome – we stop the process when a user-defined score is reached.

3.4.3 Geometrical filtering

Hereafter, we decompose the geometrical filtering in three parts. First, we describe the proceeding for fitting cylinders through **RANSAC**. Next, we talk about the stopping criteria involved in the cylinder fitting procedure. Finally, we present our paradigm for fine tuning the cylinders parameters.

RBT casts N_d directions, equi-distributed on a half-unit sphere from C^a . Each direction is associated by **RANSAC** to a cylinder knowing its direction following Alg. 2. **RBT** reviews the N_d directions with increasing angle^b, i.e. the angle formed **with respect to (w.r.t)** direction \vec{d} . Hereafter, we explain the way **RBT** filters the points bred by the ray-casting procedure for one direction \vec{d}_i .

^aThe plane passing by C and perpendicular to \vec{d} separates the two halves of the sphere. **RBT** uses the half sphere falling in direction \vec{d} .

^bOne can sort the directions according to their decreasing dot product with \vec{d} .

Algorithm 2 RANSAC

- 1: **function** RANSAC(\mathcal{P} , \vec{d}_i , p_{inl})
 - 2: Select randomly three points.
 - 3: Rotate the three points so that \vec{d}_i is parallel to the z-axis.
 - 4: Project the three points onto the xy-plane (2D problem).
 - 5: Compute the center and the radius.
 - 6: Compute the euclidean distance of \mathcal{P} to the cylinder and the inlier rate.
 - 7: Save the parameters if the inlier rate is improved.
 - 8: If the number of iterations N or the inlier rate respectively exceed N_{\max} or p_{inl} , then return the cylinder with the best inlier rate.
 - 9: If $N < N_{\min}$, then go to 2.
 - 10: **end function**
-

RANSACing cylinders First of all, **RANSAC** selects randomly three points (stochastic selection). These points are rotated so that the z-axis is aligned to direction \vec{d}_i (**Rabbani and Heuvel (2005)**). Then, equations provided in section 3.4.1, combined with the inverse rotation, supply the center C^* and the radius r^* in the 2D-plane passing by C and perpendicular to \vec{d}_i (cylinder characterization).

Later on, **RBT** computes the Euclidean distance from \mathcal{P} to the generated infinite cylinder and counts the number of *inliers*. In 2D, as displayed in Fig. 3.4, this can be seen either as only considering points located between two concentric – an interior (green) and an exterior (blue) – circles or within a distance r_t to the cylinder (red). The ratio of inliers **w.r.t** the cardinal of \mathcal{P} qualifies how well the cylinder fits the data. This stochastic procedure for selection, cylinder characterization and inliers count are repeated at least N_{min} iterations. We subject the fitting procedure to fulfill N_{min} iterations in order to explore a minimum number of candidate solutions. Once the cylinder found, it is considered as *valid* when its radius r^* is within $[\rho/2, 3\rho/2]$. Otherwise, the cylinder is labeled as *invalid*.

Stopping criteria As soon as a maximum number of iterations N_{\max} is reached or a threshold over the inlier rate p_{inl} is reached, the fitting stops. Once a cylinder presents

with an acceptable inlier rate above p_{inl} and is valid, the cylinder estimation procedure stops and returns the cylinder with no further exploration of the remaining directions. In case of failure, the next direction is treated. Up to this level, **RBT** prunes directions in the vicinity of direction \vec{d} (pruning strategy), however, a complete exploration of all directions may be sometimes necessary. Consequently, once all directions have been explored and no cylinder satisfies p_{inl} , the cylinder estimation algorithm returns the valid cylinder with the maximum inlier rate if it is at least $p_{inl}/2$ or otherwise, the algorithm returns that no valid cylinder could be found. Algorithm 2 recaps the cylinder fitting.

Algorithm 3 Cylinder estimation

Ensure: $\{\mathcal{P}', C', \vec{d}', \rho'\}$

- 1: **function** CYLINDERESTIMATION($\mathcal{P}, \vec{d}, \rho, p_{inl}$)
 - 2: Generate $\{\vec{d}_i\}_{i=1\dots N_d}$ and sort them according to their angle relative to \vec{d}
 - 3: **for all** $\{\vec{d}_i\}$ **do**
 - 4: $(\mathcal{P}_i, C_i, \rho_i) = \text{RANSAC}(\mathcal{P}, \vec{d}_i, p_{inl})$.
 - 5: If the $inliers_rate_i$ is above p_{inl} and $\rho_i \in [\rho/2, 3\rho/2]$, save the corresponding cylinder with its inliers $(\mathcal{P}', C', \vec{d}', \rho')$.
 - 6: If no cylinder is above p_{inl} , the best cylinder with $r_i \in [\rho/2, 3\rho/2]$ and presenting at least $p_{inl}/2$ is saved.
 - 7: If no cylinder is above $p_{inl}/2$, no valid cylinder is returned.
 - 8: **end for**
 - 9: **end function**
-

Refinement The above algorithm finds a valid cylinder and a point-set capturing the vessel surface. When in fact, the tracking direction was computed from a discrete set of directions and other parameters inferred from it, we noticed that when the chain list of cylinders is stirred, this lack of smoothness sometimes produced tracking errors (prematurely stop). As a result, the returned cylinder parameters (center, radius and direction) are refined, taking account of inliers, through Powell's optimization algorithm. The

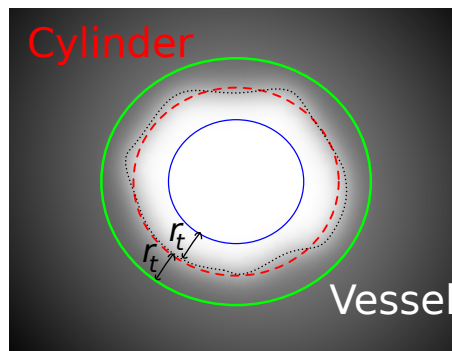


Figure 3.4: 2D representation of the proposed geometric criterion. The vessel lumen varies within a 2D-torus spanned with a cylinder (red dashed contour) and with minor radius r_t (green and blue circle). The circular hypothesis is insufficient for capturing the vessel cross-section (black dotted contour).

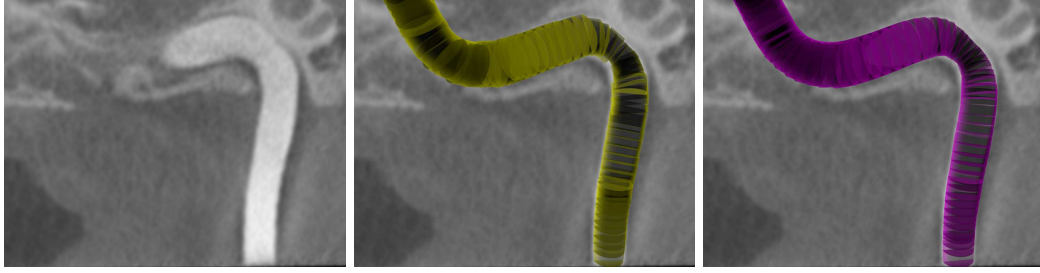


Figure 3.5: Influence of Powell refinement in the resulting cylinders. The corresponding cylinders for both centerlines produced without (middle) and with (right) Powell optimization on one vessel (left). Smooth transitions between cylinders are noticed at curved sections when using the refinement procedure.

centerline of a vessel tracked with a refinement procedure is, assuredly, smoother than without, specially at curved sections of the vessel (Fig. 3.5). Subsequently, the center C^* is refined as the median of the inliers point-set along the axis \vec{d}_i . Next, a new set of points is extracted by ray casting from the newly tuned cylinder center and the inliers points \mathcal{P}' are returned with the cylinder. Its height is adjusted so that 75% of \mathcal{P}' points are encompassed (Fig. 3.1d). The estimation algorithm is summarized in Algorithm 3 and the RBT parameters and their descriptions are condensed in Table 3.1.

3.5 Tracking a whole branch: Initialization, prediction step and stopping criteria

In general, RBT requires only one point inside the vessel lumen (Sec. 3.6). However, the user supplies a starting point C , a rough estimation of the radius r and a vessel direction estimate \vec{d} for the first tracking instance. In practice, the vascular tree is tracked by starting the procedure in the carotid or vertebral artery (the vascular tree stem vessels). The user defines a point inside the vessel and supplies an estimation of the local vessel radius. Most vessels irrigating the brain are vertical, the tracking starts in the upward direction. The parameters are refined by fitting a finite cylinder ($C^*, \vec{d}^*, \rho^*, h^*$) using the cylinder estimation algorithm described above. The next cylinder is predicted as ($C = C^* + s \cdot h^* \cdot \vec{d}^*, \vec{d} = \vec{d}^*, \rho = \rho^*$), where s is the step size. The cylinder estima-

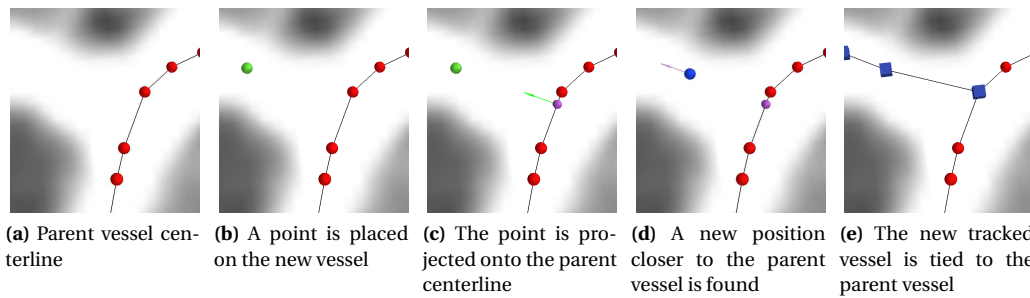


Figure 3.6: Tracking a new vessel.

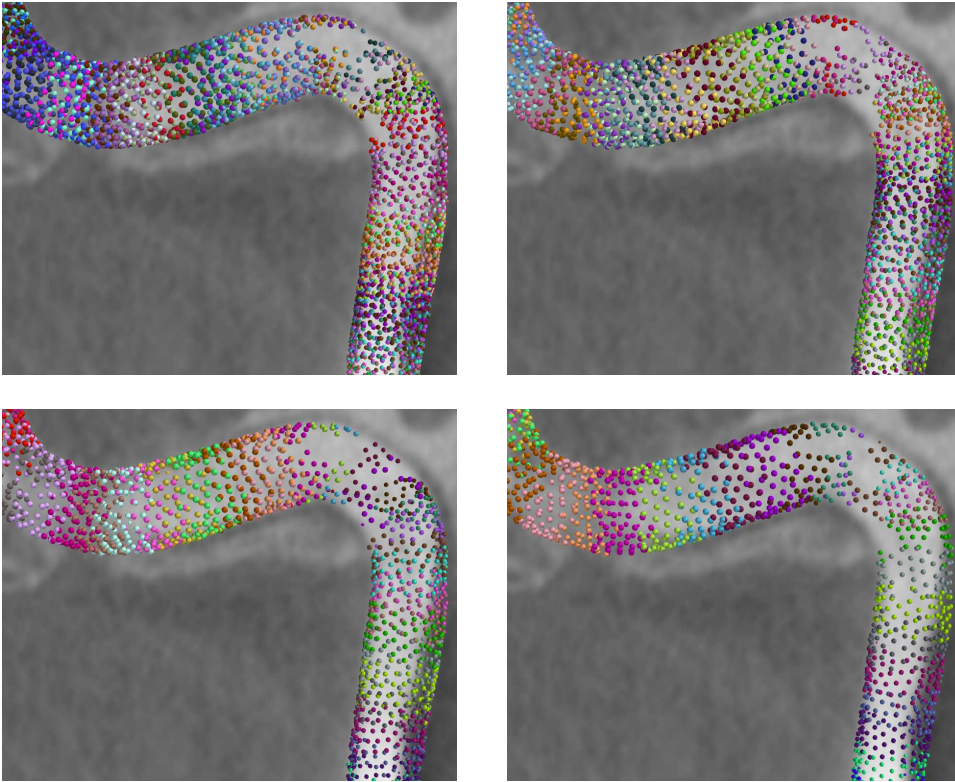


Figure 3.7: Influence of the step distance on the dense sampling. Resulting points for the prediction step $C = C^* + s \cdot h^* \cdot \vec{d}^*$: (top left) $s = 0.25$, (top right) $s = 0.5$, (bottom left) $s = 0.75$ and (bottom right) $s = 1$. Interpenetration of point-sets is almost inexistent for $s = 1$, while on the contrary a value of $s = 0.25$ exhibits point-sets covering the same area of the blood vessel. A step size ranging from $s = 0.5$ to $s = 0.75$ offers a good trade-off between interpenetration and distance between predictions.

tion algorithm is applied and thus the tracking progresses in the direction of the current cylinder direction. Fig. 3.7 displays the differences and influence of the step sizes on the resulting dense sampling of a vessel. We aim at producing point-sets exhibiting interpenetration so that holes, in the global dense sampling of the vessel, are limited. A step size ranging from 0.5-0.75 gave a sound trade-off between the step prediction and interpenetration. Heretofore, we chose a step size $s = 0.5$. **RBT** halts when the tracking turns back, i.e. the center of the new cylinder is closer than $\rho/10$ to the already collection of segments, or no valid cylinder is returned from the cylinder estimation procedure.

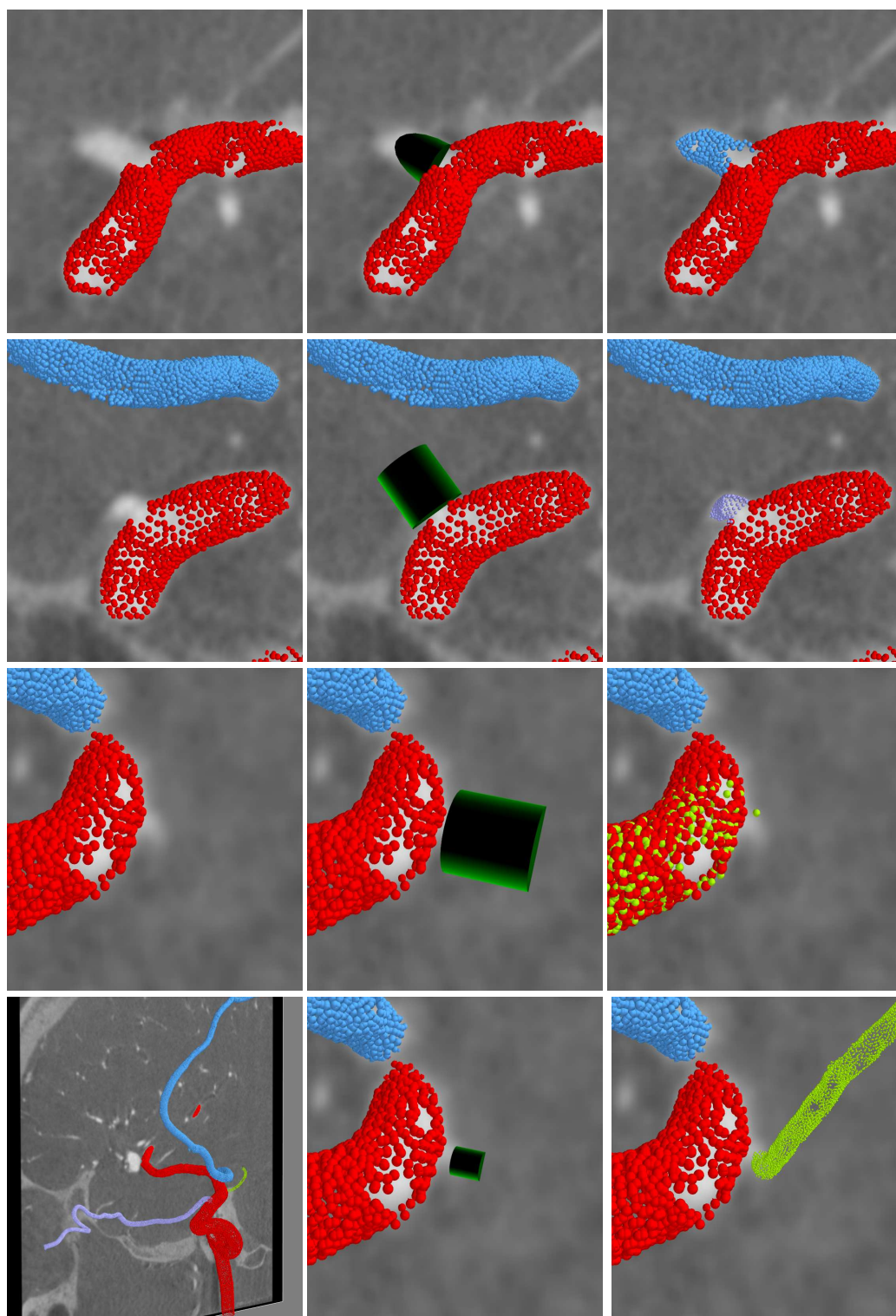


Figure 3.8: Pseudo-automatic seed generation from a parent vessel (red). (From top to bottom) The first three rows exhibit instances of tracking initialized with seeds proposed by RBT (Fig. 3.6) for three different vessels width: 2mm (top), 1mm (center) and 0.6mm (bottom). (From left to right) First column displays the parent vessel and the target vessel lumen. Second column puts forth the corresponding seed. Third column shows the resulting point-set. The last row evinces the segmented vascular tree (left), a new user-defined seed – only the seed’s radius was manually modified – for the third-row case (middle) and the corresponding points for the tracking (right).

3.6 Tracking branches

A point is placed manually on the new vessel, roughly close to the bifurcation (green point Fig. 3.6b). The vector linking this point to the nearest point on the parent vessel centerline (purple point on Fig. 3.6c) provides the direction for the new vessel \vec{d} . The radius ρ is initialized with the value computed by the parent vessel tracking, at the purple point location. The manual position of the initial point inside the vessel branch is rough and might be either too far from the parent vessel (and thus it could create a discontinuity on the vessel sampling) or too close to the parent vessel (and thus potentially the parent vessel could be tracked again). This rough positioning is automatically tuned by placing the actual initial seed at a distance of 1.25ρ (blue point on Fig. 3.6d) along the estimated tracking direction \vec{d} . **RBT** is run with the new seed, the tracking direction \vec{d} and the radius ρ provided by the parent vessel. Finally, once the new vessel is tracked, the resulting centerline is tied to the parent vessel (blue squares on Fig. 3.6e).

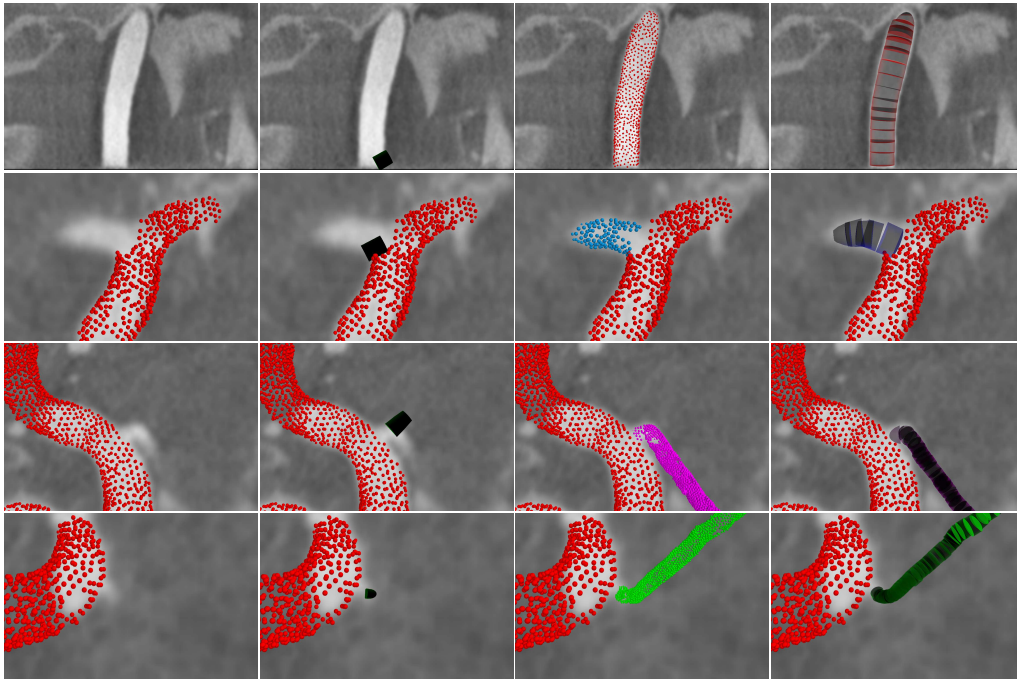


Figure 3.9: **RBT** robustness against rough tracking directions. Trackings of Fig. 3.8 – where seeds were automatically generated – were inspected. Directions were manually modified for the three former cases and the stem vessel (carotid) was included in the inspection : the largest vessel (1st row), a large vessel (2nd row), a medium size vessel (3rd row) and a small size vessel (4rd row). (From left to right) First column displays the parent vessel, except for the first row, and the target vessel lumen. Second column puts forth the modified direction of the seed. Third column shows the resulting point-set. The last column evinces the adaptation of cylinders to capture the vessel lumen.

The pseudo-automatic seed generation for furcations works fine for most cases. Fig. 3.9 and Fig. 3.8 display three instances of tracking for three different widths. Above all, **RBT** algorithm is robust to initialization concerning the tracking direction as depicted in Fig. 3.9. The first two rows in Fig. 3.8 display cases for medium and large width. In these two configurations, **RBT** successfully tracked the aimed vessel. Yet, in cases

where the ratio between parent and children vessels widths exceed 50%, the proposition for the local vessel width is largely overestimated which can lead to issues, such as the tracking of the parent vessel (Fig. 3.8, third row). In this circumstances, user intervention is mandatory for correcting this estimate (Fig. 3.8, last row). Nonetheless, these situations are rare and happen only with small vessels rooted on the carotid artery (red vessel in Fig. 3.8), elsewhere, no gross variations on the vessel width were noticed such that RBT had no problems.

3.7 Capturing aneurysms

Aneurysms clearly do not comply with our hypothesis of tubularity and our tracking algorithm may not provide an accurate description. Our main concern, when pondering aneurysms in our framework, is to obtain the topology and points at its surface. While dealing with volumetric data – such as **Magnetic Resonance Angiography (MRA)** and **Computed Tomography Angiography (CTA)** – with contrast enhanced vessels, Neugebauer et al. (2010) stated that a basic segmentation, like thresholding combined with a Connected Component Analysis, was sufficient to extract aneurysms. Generally speaking, 3DRA is of even greater quality than MRA and CTA. Due to good contrast, a simple ray-casting technique followed by a user-defined threshold (Wink et al. (2000)) may yield points at the aneurysm surface. Nevertheless, an open issue remains the location and number of ray-casting focuses when resorting to ray-casting-based procedures for capturing aneurysm. A more profound question arises when ray-casting focuses should be related to topology which is compulsory for producing compatible segmentation outcomes with RBT. In our framework, aneurysms are interactively captured: the user places one point C inside the aneurysm and sets the approximate distance from the point to the vessel wall ρ . Next, rays are cast from C using the same technique as described in Sec. 3.4 and thanks to a user-defined threshold over the gradient intensity, points are extracted. When the resulting point-set is correct, the user reiterates the transaction by placing another point. The reason for repeating this procedure is that aneurysms might have a very complex shape, specially saccular aneurysms, and one single ray-casting instance may be insufficient to capture the whole surface. For instance, Fig. 3.18 exhibits a *classical* saccular aneurysm whereas Fig. 3.17 put forth one formed by two lobes.

Until now, only points were extracted. The topology is inferred by connecting the centers for each transaction in the order they were created. The tracking direction \vec{d} and the height h are respectively given by the direction and distance between two consecutive centers. Then for each center, a cylinder is defined. The direction of the last transaction is set to its predecessor tracking direction. The resultant centerline is the ordered linked list of these cylinders and their corresponding point-sets. However, the user has the possibility to modify the proposed topology by merging centers and point-sets or changing the connection order of centers if needed. Finally, the generated centerline is tied to the closest position on the parent vessel centerline.

3.8 Implementation details

In this section, we discuss some technical details about the implementation of our algorithm. Important attention is paid to the ray casting procedure. More precisely, we focus on the interpolation methods and gradient computation along rays and the technique used for casting evenly sampled rays in the image data.

3.8.1 Ray-casting

During the ray-casting procedure, we want points at the vessel surface but relatively well-distributed in the vicinity of a point. Furthermore, the quantization of the direction space, during the cylinder estimation, shares the same needs. For this purpose, a geodesic shape is the most efficient way to cover a given volume of space. The **RBT** algorithm employs a geodesic dome construction algorithm. A geodesic dome is a triangulation of a Platonic solid or other polyhedron to produce a close approximation to a sphere (or hemisphere)². The only difference between the ray-casting procedure and the cylinder direction pruning is that for the former process is grounded on the geodisation of a unit sphere while the latter one, considers only the half of this sphere as candidate directions.

The geodisation heuristic starts from an icosahedron³ and subsequently, divides its edges. For a better understanding of this tessellation process, Fig. 3.10 illustrates a first subdivision of one facet of the icosahedron (yellow). First, the midpoints for each edge – constructed from the three vertices (white) – are computed. These three points (green) help to generate four triangles over the facet. Finally, each vertex of the triangle is projected onto the unit sphere. Generally speaking, the n -th tessellation frequency or geodisation order divides each edge of the polyhedron in 2^{n-1} edges and projects each edge onto the circumsphere (Fekete (1990)). For a given geodisation order 1,2,3 and 4 turns out to be respectively 12, 41, 161 and 654 vertices (Fig. 3.11). One ray for the ray-casting is cast in the direction from the center of unit-sphere and one vertex of the geodisation. **RBT** proceeds analogously for providing the N_d directions during the cylinder estimation step but restricted to half of the vertices.

² Weisstein, Eric W. "Geodesic Dome." From MathWorld—A Wolfram Web Resource. <http://mathworld.wolfram.com/GeodesicDome.html>

³ <http://gts.sourceforge.net>

Parameter	Description	Value
p_{int}	inlier rate threshold	70%
r_t	relative threshold w.r.t local cylinder radius	10%
N_r	number of rays thrown to extract points	162
N_d	number of axis direction tested	81
N_p	number of points per ray	128
N_{min}	minimum number of tests for RANSAC	220
N_{max}	maximum number of tests for RANSAC	500

Table 3.1: Parameter values for the **RBT** algorithm.

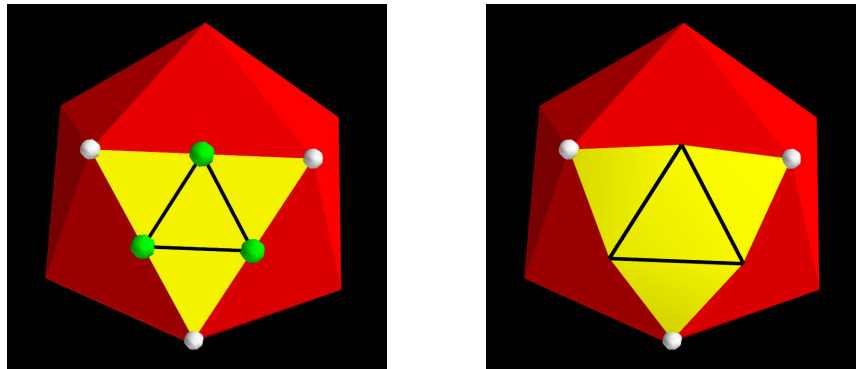


Figure 3.10: Geodisation principle on one facet of an icosahedron. (left) First, the midpoints (green) of the facets (yellow) are computed. Then the facet is subdivided in 4 triangles. (right) Finally, the vertex of each triangle (yellow) are projected onto the unit sphere.

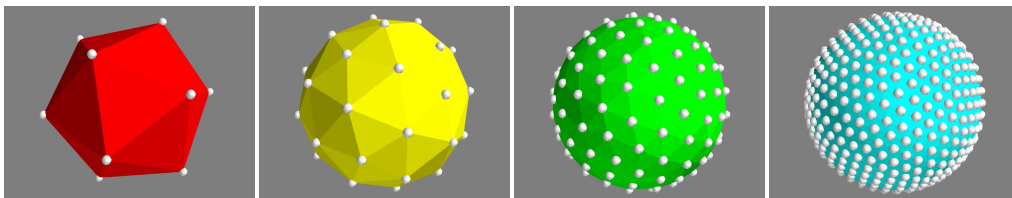


Figure 3.11: Four geodisation orders of an icosahedron from left to right: order 1, 2, 3 and 4. The white dots represent the vertices of the resulting tessellation.

3.8.2 Interpolation

RBT computes the directional gradient along a ray to find the vessel wall (characterized by the strongest minimum). Therefore, it needs to interpolate the image data during the ray-casting procedure. The quality of the resultant point-set and hence, the accuracy of the cylinder estimation are impacted by both the choice of the interpolation method and the way the gradient is computed. In this specific aspect, the computation of the minimum of the directional gradient can be approached in two different fashions. First, intensity values along the ray are extracted and then, the **one-Dimension (1D)** gradient along the ray is estimated with a finite difference method. Second, the **2D (or 3D)** gradient in the image is directly computed in the image (or volume) data and subsequently, an interpolation process is carried out to compute the directional gradient along the ray. The main differences between both techniques is the problem dimensionality for the directional gradient computation.

For the interpolation process, three well-known interpolation methods were tested: **Bi-Linear (L)**, **Bi-Cubic (C)** and **Bi-cubic Spline (BS)** interpolation. The finite differences were ensured by **Central Differences (CD)** and **Forward Difference (FD)** methods. Finally, the direct computation of the multidimensional gradient was entrusted to **Alvarez's Method (AM)** (Alvarez (1996)) which has the property to be rotational and scale invariant as well as unconditionally stable. Also **AM** was chosen for its fast computation when compared to a Gaussian-based computation. With this in mind, 9 configurations

were set for assessment. Six configurations for intensity values extraction through an interpolation method and coupled to a finite difference method and the search for the minimum along the ray (1 to 6 with in the following list) and 3 configurations for **AM**, followed by an interpolation process and the search for the minimum along the ray (1 to 9 in the following list):

1. **L + FD**: Intensity value extraction with Linear interpolation coupled to a Forward Difference method for the **1D** gradient computation;
2. **L + CD**: Intensity value extraction with Linear interpolation coupled to a Central Differences method for the **1D** gradient computation;
3. **C + FD**: Intensity value extraction with Cubic interpolation coupled to a Forward Difference method for the **1D** gradient computation;
4. **C + CD**: Intensity value extraction with Cubic interpolation coupled to a Central Differences method for the **1D** gradient computation;
5. **BS + FD**: Intensity value extraction with Bicubic Spline interpolation coupled to a Forward Difference method for the **1D** gradient computation;
6. **BS + CD**: Intensity value extraction with Bicubic Spline interpolation coupled to a Central Differences method for the **1D** gradient computation;
7. **AM + L**: Alvarez's Method for the **2D** gradient computation, then the directional gradient along the ray is computed through Linear interpolation;
8. **AM + C**: Alvarez's Method for the **2D** gradient computation, then the directional gradient along the ray is computed through Cubic interpolation;
9. **AM + BS**: Alvarez's Method for the **2D** gradient computation, then the directional gradient along the ray is computed through Bicubic Spline interpolation.

Synthetic data

In order to quantify the accuracy of each configuration, we mimicked our real data. During the cylinder estimation, **RBT** turns the **3D** problem into a **2D** problem by finding the center and radius of the cylinder for a known direction. Consequently, a vessel can be seen as a circle in **2D** whose center C and radius ρ are to be characterized. For this purpose, a known unit circle \mathcal{C} (green circle in Fig. 3.12) was used.

Without loss of generality, ρ was set to 1 for the entire experiments. The circle \mathcal{C} was discretized. An $N_{px} \times N_{px}$ image was generated which discretized the space from -5ρ to 5ρ in both directions (Fig. 3.12). The discretization procedure worked as follows: for each pixel on the image, a mean filter provided the intensity value (in grayscale) for each pixel. The spacial resolution of the image S in both directions was given by the number of pixels N_{px} sampling the diameter of circle \mathcal{C} and computed as $S = 2\rho / N_{px}$, where $N_{px} \in [2, 10]$ (with step of 1). N_{px} ensures the discretization level of the circle and also mimics vessels having widths ranging from 0.4 mm to 2mm given a usual **3DRA** resolution of 0.2 mm.

C was placed at $(0,0)$ and allowed to move within $[0,0.5S] \times [0,0.5S]$ (with step 0.1). Owing to the symmetry of the circle and the periodicity of digitization artifacts, only positive values were considered. In summary, one image was generated for each position of circle \mathcal{C} and at different levels of digitization. In the end, 324 images (9 cases for N_{px} and 36 for C) were thus generated.

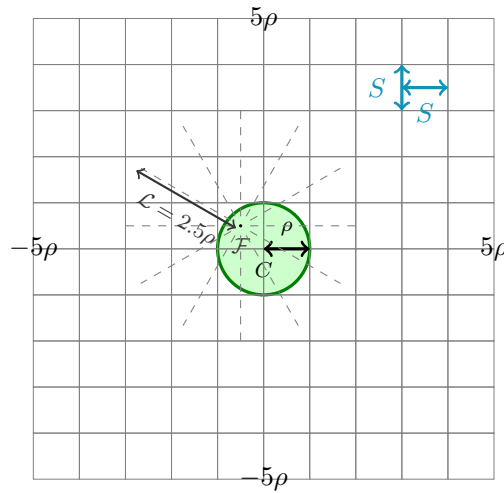


Figure 3.12: A unit circle (green) of center C and radius ρ is discretized by $N_{px} = 2$ pixels along its diameter. The spatial discretization process of the circle starts by, first, computing spatial resolution $S = 2\rho/N_{px}$ in both directions. Finally, a mean filter is used for assigning to each pixel an intensity value (grayscale). In the end, an image is thus generated. Within the image, the ray-casting focus \mathcal{F} is allowed to evolve inside the circle and from which, rays are cast evenly sampled with length $\mathcal{L} = 2.5\rho$.

Protocol

For one image in the data set, the ray-casting focus \mathcal{F} was authorized to move inside circle \mathcal{C} (green disk in Fig. 3.12). \mathcal{F} evolved from $-\rho$ to ρ in both direction with step size $\rho/10$ but the position was accounted if $|\mathcal{F} - C| < \rho$. From this location, $N_d = 128$ rays were evenly cast and $N_p = 128$ points sampled the ray along its length $\mathcal{L} = 2.5\rho$. Afterwards, the minimum along the directional gradient was computed according to one of the 9 techniques. The resulting point-set was fed to **RANSAC** for finding the center C^* and radius ρ^* of the estimated circle \mathcal{C}^* . The relative threshold r_t , for considering the points as inliers, in the fitting procedure was set to $10\%\rho$ and the inlier rate was fixed to 70%. Note that these values are the same as those used by **RBT** (compare with Tab. 3.1). As a result, 102060 instances were accounted per configuration.

Metrics

RANSAC supplies an estimate $\mathcal{C}^* = (C^*, \rho^*)$ of the reference circle $\mathcal{C} = (C, \rho)$ and a point-set \mathcal{P}^* . Firstly, we evaluated the precision of the 9 techniques with regard to their accuracy on the characterization of \mathcal{C} . For that, we used the **Root Mean Square (RMS)** computation. The precision on the estimation of the center C was computed as

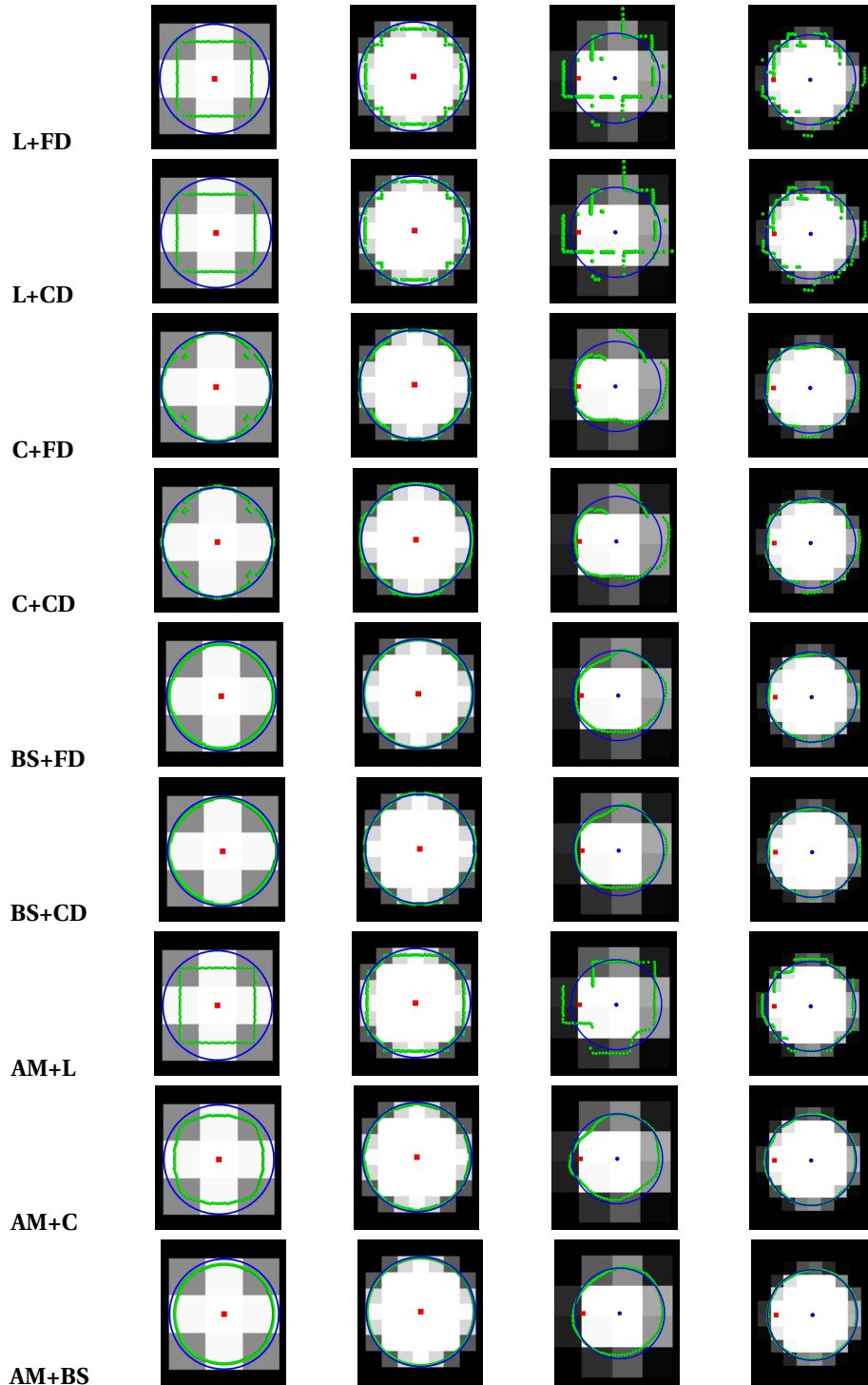


Figure 3.13: (Rows) Each row corresponds to one configuration in the interpolation experiment. The blue dot point is the center C of the blue unit circle \mathcal{C} . The red square indicates the ray-casting focus \mathcal{F} which extracts the green points. Experiments were run with $C = \mathcal{F} = (0, 0)$: the unit circle was discretized with $N_{px} = 3$ (1st column) and $N_{px} = 7$ pixels (2nd column). In the last two columns, the center was placed at $(0.5, 0.4)$ and the ray-casting focus was positioned at $(-0.3, 0.4)$: the unit circle was sampled by $N_{px} = 3$ (3rd column) and $N_{px} = 7$ pixels (4th column).

followed:

$$RMS_{C^*} = \sqrt{\frac{1}{N} \sum |C - C^*|^2} \quad (3.3)$$

and similarly, the RMS error made while estimating ρ :

$$RMS_{\rho^*} = \sqrt{\frac{1}{N} \sum (\rho - \rho^*)^2} \quad (3.4)$$

Secondly, we assessed the **RMS** distance of the extracted point-set \mathcal{P}^* with respect to the estimated circle \mathcal{C}^* :

$$RMS_{\mathcal{C}^*} = \sqrt{\frac{1}{N} \sum_{P^* \in \mathcal{P}^*} d^2(P^*, \mathcal{C}^*)} \quad (3.5)$$

and **w.r.t** the reference circle \mathcal{C} :

$$RMS_{\mathcal{C}} = \sqrt{\frac{1}{N} \sum_{P^* \in \mathcal{P}^*} d^2(P^*, \mathcal{C})} \quad (3.6)$$

where the squared distance between one point P^* of point-set \mathcal{P}^* and circle \mathcal{C} (alike for \mathcal{C}^*) was given by:

$$d^2(P^*, \mathcal{C}) = |P^* - C|^2 - \rho^2 \quad (3.7)$$

Finally, the quality of point-set \mathcal{P}^* was appraised by computing the inlier rate p_{inl} **w.r.t** the reference circle and within a distance of $\rho/10 = 0.1$.

Besides, we coupled the metrics with two statistical values: the mean and the standard deviation. Last but not least, we took 34020 cases from the 102060 instances – i.e. those that correspond to data created with $N_{px} = 2, 3, 4$ – and recompute the statistical values. The idea behind this approach was to provide a finer view on cases considered of low resolution (Low) while on the contrary, a global view of metrics was supplied with all the instances (All).

Paradigm choice

Table 3.2 recaps the outcomes on the accuracy of the circle estimation (RMS_{ρ^*} and RMS_{C^*}), i.e. radius and center computation from the extracted points, on the quality of the extracted points ($RMS_{\mathcal{C}^*}$ and $RMS_{\mathcal{C}}$), i.e. how far the points lie from the unit circle, and on the inlier rate p_{inl} , i.e. how well the points comply to the hypothesis of circularity, for the 9 configurations.

A word about circle estimation, it turned out that worst rates – i.e. gross error values – for the radius RMS_{ρ^*} and center RMS_{C^*} characterization were undoubtedly obtained with configuration **L + FD**. In contrast, the best figures – i.e. low error values – for RMS_{C^*} were shared with **BS + FD**, **BS + CD** and **AM + BS** configurations. Indeed, similar values were achieved with the common denominator: Bicubic Spline interpolation. Once again, **BS** interpolation is one of the reasons explaining the success of **BS + FD** and **BS + CD** when contending with the radius estimation. In contrast, **AM + BS** produced higher error values on approximating the real radius and especially at low resolution (Fig. 3.13 compare 1st column for configurations **BS + FD** and **AM + BS**). This first analysis unraveled that linear interpolation bred staircase point-sets of the unit circle – thus leading to

wrong parameter estimation (Fig. 3.13 1st and 2nd rows) – while on the contrary, Bicubic Spline interpolation handles nicely the changes in resolution but still dependent of the gradient computation (Fig. 3.13 5th, 6th and 9th rows).

The quality of the extracted points w.r.t the estimated circle $RMS_{\mathcal{C}^*}$ and to the reference circle $RMS_{\mathcal{C}}$ put clearly forth that linear interpolation prevails, independent of the resolution and gradient computation technique, at producing points far from a circular shape. Assuming that **BS** interpolation excels the two others, let focus our survey in the three paradigms using **BS** interpolation: **BS + FD**, **BS + CD** and **AM + BS**). Regardless of the resolution, similar figures for $RMS_{\mathcal{C}^*}$ were observed for these configurations. In other words, $RMS_{\mathcal{C}^*}$ values granted that the resulting points had a likely circular shape. Conversely, when considering $RMS_{\mathcal{C}}$ w.r.t the reference circle, **AM + BS** brought forth points far from the unit circle. This is the resultant of the fact that **AM + BS** had hard times, as aforementioned, at supplying correct points for radius estimation (see RMS_{ρ^*} in Table 3.2). As a result, **BS + FD** and **BS + CD** configurations excels with similar figures in this field. This latter statement supports our assumption that **BS** interpolation provides the best results among the three interpolation methods. Furthermore, when combined with **1D** gradient computation, **BS** interpolation yielded the best rates for circle estimation and point-set extraction quality. The inlier rate p_{inl} also sustained our claim that **BS** interpolation and **1D** gradient computation where **BS + FD** and **BS + CD** displayed highest p_{inl} values with same magnitude for standard deviations.

A final word on **BS + FD** and **BS + CD** configurations, **CD**-based paradigms performed better in all the assessed features – i.e. circle estimation, quality of the extraction and compliance of point-set to the hypothesis of circularity – than **FD**-based configurations.

In conclusion, the **BS + CD** scheme was implemented during the ray-casting procedure for its high accuracy in a low resolution context, as well as in high resolution, good compliance to our circular hypothesis and computational cost (inexpensive).

Scenario	$\overline{RMS_{C^*}}$		$\overline{RMS_{\rho^*}}$		$RMS_{\mathcal{C}^*}$				$RMS_{\mathcal{C}}$				p_{inl}			
	All	Low	All	Low	All		Low		All		Low		All		Low	
					μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
L + FD	0.309	0.528	0.180	0.307	0.190	0.182	0.368	0.222	0.156	0.100	0.267	0.100	0.681	0.142	0.538	0.084
L + CD	0.269	0.459	0.170	0.291	0.179	0.171	0.347	0.209	0.150	0.100	0.262	0.099	0.697	0.152	0.538	0.087
C + FD	0.108	0.180	0.080	0.135	0.083	0.074	0.149	0.097	0.080	0.053	0.136	0.056	0.894	0.121	0.759	0.101
C + CD	0.107	0.179	0.075	0.128	0.082	0.073	0.149	0.096	0.079	0.052	0.135	0.054	0.894	0.120	0.760	0.100
BS + FD	0.064	0.098	0.042	0.064	0.053	0.039	0.085	0.053	0.058	0.038	0.097	0.040	0.949	0.102	0.860	0.135
BS + CD	0.062	0.095	0.040	0.061	0.052	0.038	0.084	0.051	0.055	0.041	0.095	0.045	0.952	0.099	0.865	0.133
AM + L	0.168	0.282	0.082	0.134	0.114	0.101	0.211	0.124	0.114	0.081	0.205	0.081	0.808	0.168	0.616	0.096
AM + C	0.070	0.110	0.069	0.114	0.057	0.043	0.097	0.052	0.068	0.061	0.132	0.066	0.932	0.117	0.815	0.134
AM + BS	0.062	0.093	0.091	0.153	0.053	0.038	0.086	0.045	0.075	0.076	0.149	0.090	0.940	0.107	0.847	0.131

Table 3.2: Outcomes for the 9 configurations. Each one was run 102060 times for different positions of the unit circle \mathcal{C} (ground truth) and the ray-casting focus. Five metrics were evaluated at two different scales of discretization of the unit circle: (All) all levels $N_{px} \in [2, \dots, 10]$ and (Low) discretizations ranging from 2 to 4 pixels (34020 instances per configuration). Each discretization scale computation may present its mean value μ and standard deviation σ . The first metric $\overline{RMS_{C^*}}$ measures the average root mean square error of the estimated center C^* to the center of unit circle C . Likewise, $\overline{RMS_{\rho^*}}$ informs about the average root mean square error of the estimated radius ρ^* to ground truth radius $\rho = 1$. $RMS_{\mathcal{C}^*}$ (respectively $RMS_{\mathcal{C}}$) computes the root mean square distance of the extracted point-set \mathcal{P}^* to the estimated circle \mathcal{C}^* (to the ground truth \mathcal{C}). Finally, the inlier rate p_{inl} qualifies how well the point-set \mathcal{P}^* complies with \mathcal{C} within a distance of $\rho/10$. Pink and green cells indicate respectively the worst and the best values for each column.

3.9 Validation protocol

In this section, we show the strength, robustness and accuracy of our **RBT** algorithm when tracking and extracting the centerline in real data. A large number of methods have been evaluated qualitatively (Wink et al. (2000); Tek et al. (2001); Flasque et al. (2001); Wesarg and Firlre (2004); Carrillo et al. (2007); Schaap et al. (2007); Hernandez and Frangi (2007); Jiang et al. (2007)). In these works, detection, extraction or segmentation correctness have been visually determined. Other works present a quantitative evaluation of their performance (Li and Yezzi (2006); Tyrrell et al. (2007); Yedidya and Hartley (2008); Gülsün and Tek (2008)). Barely few algorithms have been compared to another (La Cruz et al. (2004); Worz and Rohr (2007); Wong and Chung (2007)) and only few algorithms in the literature have been subject to a standardized comparison (Schaap et al. (2009)). Benefiting from this latter, serious assessment of state-of-the-art Multiple Hypothesis Tracking (Friman et al. (2010)) (**Multiple Hypothesis Tracking (MHT)**) algorithm's performance, we quantify **RBT**'s strength and accuracy with respect to it. **MHT** is available on *MevisLab* software⁴ under the *TubularTracking* module. It achieved the highest score among 13 algorithms, both fully automatic as well as semi-automatic, in the *3D segmentation in the clinic: A Grand Challenge II - Coronary Artery Tracking competition* (Schaap et al. (2009)). Hereafter, we describe the main steps involved in the **MHT** proceeding.

Given a start-point provided with a tracking direction and an estimate of the vessel radius. The **MHT** algorithm proceeds in two phases that are iterated during the tracking: building the search tree and evaluating the tree to determine the next tracking step. In order to build the tree, **MHT** produces a number of predictions for the next vessel segment by placing trial positions, evenly spaced on a half-sphere, ahead of the current position (search step). Then, **MHT** scores all the trials according to a vesselness criterion. Next, positions that do not comply with a pruning threshold are discarded. Later on, each surviving position becomes the center of a new search step until a certain search-depth is attained. The resulting positions form a tree-like structure that keeps the most plausible hypotheses where vessels can be found. Finally, the evaluation of the search tree is fulfilled: leaves not surviving a termination threshold, are discarded; if leaves survived, bifurcations are detected – to be pursued later – and the tracking moves toward the leaf with the highest score, otherwise the tracking is terminated.

In the beginning, we intended to reconstruct the vascular tree using a multi-branch approach, however, we remarked that **MHT** provided wrong trackings which would certainly make the comparison task even more difficult. Consequently, we adopted a mono-branch tracking which corresponds to the same paradigm as **RBT**. When comparing tracking results, three principal questions arise: 1) how to handle furcations and branches of varied length? (see Sec. 3.9.1) 2) how to ensure that no tracking algorithm is favored during the essays, in particular by the initial conditions? (see Sec. 3.9.1) 3) how to evaluate the accuracy of a tracking instance? (see Sec. 3.9.1)

⁴<http://www.mevislab.de>

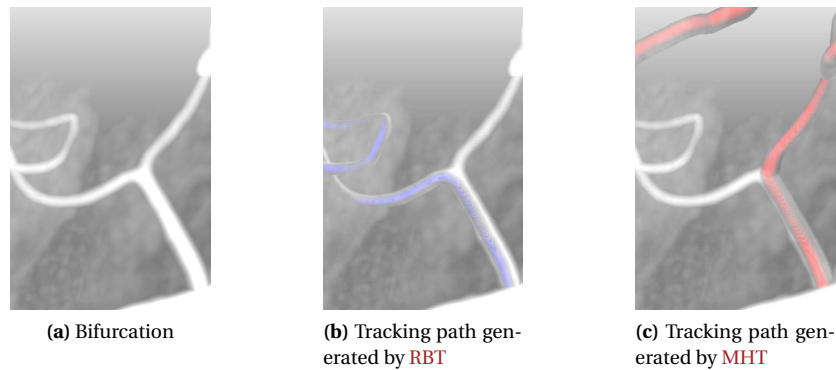


Figure 3.14: RBT and MHT algorithms tracking a vessel (left). RBT (center) follows a different path than MHT (right).

3.9.1 Correspondence between trackings

Vascular trees exhibit a complex morphology, i.e. kissing vessels and pathologies (aneurysms), and wherefore a direct comparison of tracking outcomes may lead to errors. Concerns are different when facing a particular type of tracking: multi-branch and mono-branch (one single vessel at a time). In a multi-branch context, for instance, tracking algorithms may track a different number of vessels from one seed point. Therefore, a tree matching procedure of tracking outcomes at furcations is necessary to disambiguate false and true positives, and to look for missing branches. In our case of single branch tracking, algorithms may follow different branches at bifurcations (Fig. 3.14). Here again, a matching procedure is compulsory to detect common portions for evaluation since errors may be introduced at furcations. A common denominator for matching trackings, in both situations, is furcations which beacon the end of common portions or *safe* comparisons. Along this philosophy, only sections in-between bifurcations or single branches presenting no furcations at their distal extremity were considered during the comparison of outcomes produced with MHT and RBT. We called hereafter these sections *vessels*.

3.9.2 Initial conditions

It is mandatory to set both algorithms in the same initial tracking conditions. It is important to note that MHT was not designed to track vessels of large width. Therefore, the stem vessels (carotid or vertebral arteries) of the vascular tree were excluded. Generally speaking, both algorithms are *interactive extraction methods* (Schaap et al. (2009)) since they use more than one point per vessel as input. With this consideration in mind, the algorithms were both initialized with the same information: an initial position, a tracking direction and a local estimate of the vessel radius. Four points were provided and placed manually on each *vessel* section in a reformatted cut plane (Fig. 3.15): one seed point C_0 was placed at the proximal end of the vessel; the vessel direction \vec{d}_0 was estimated by placing a second point along the vessel axis, a little distal from C_0 ; the ves-

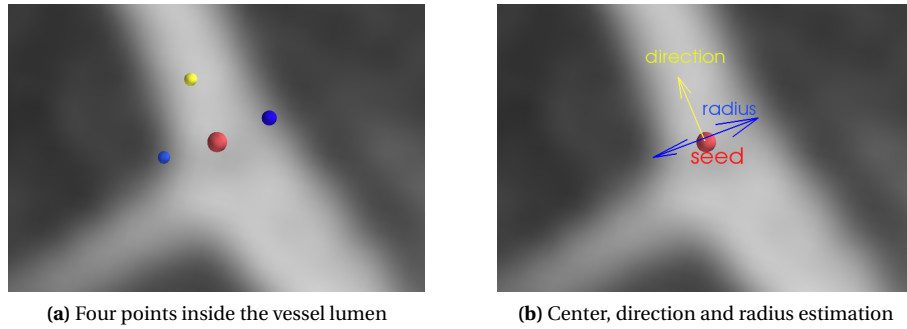


Figure 3.15: Four points placed manually inside the vessel at the proximal end of the vessel (left) providing an initial tracking position, a tracking direction and an estimate of the vessel radius (right).

Patient	1	2	3	4	5	6	7	8	9	10	Total
MHT	1	7	3	3	0	0	1	6	3	6	30
RBT	1	6	1	0	1	0	0	7	5	6	27
60	477										

Table 3.3: The number of excluded vessels per patient for both algorithms. Due to bad initialization, a certain number of vessels were excluded from the data-set. Similar figures were accounted for both algorithms.

sel radius r_0 was estimated as the distance between two diametrically opposed points placed on the vessel surface, in the cut plane showing the vessel cross-section around C_0 . Besides, when one of the algorithm did not start, the corresponding tracking for both algorithms was discarded from the evaluation set. Table 3.3 accounts the number of vessels per patient excluded from the validation set. It turned out that similar number of discarded vessels were counted for both algorithms (in total 30 for MHT and 27 for RBT).

3.9.3 Centerline evaluation

As well as former works, we intended to evaluate our tracking accuracy by considering the MHT results successfully tracked as the gold standard. At first, we quantified the performances of our method against MHT brought forward by the visual inspection. For this, we explored the success rate, i.e. the number of trackings that delineated correctly the vessels manually selected and used for validation (details are given in Sec. 3.9.4). The outcome of this visual analysis revealed that MHT produced results below our expectations in terms of tracking capability when compared to RBT. MHT could not act as a gold standard.

Previous works usually investigate two principal aspects of centerline evaluation: qualitative and/or quantitative. In essence, we followed the same outline. Besides the above visual inspection, we scanned the extraction ability, i.e. how much of the centerline can be extracted by both methods, and the accuracy, i.e. how accurately can our method locate the centerline with respect to MHT. Further details on these mea-

asures are given in Section 3.9.7. In this analysis, the vessel width assessment was not integrated since no validation, in this field, regarding MHT was carried out in previous works (Schaap et al. (2009) and Friman et al. (2010)).

3.9.4 Clinical data

A set of 10 patient data was used for validation. Each patient data set consisted of a 3DRA acquired on a vascular C-arm (Innova 4100, GE Healthcare) during the intra-arterial injection of the internal carotid artery. 3DRA volumes presented as a 512^3 isotropic voxel cubes, between 0.18-0.22 mm voxel size.

A *vessel* – section in-between bifurcations or branch with no further furcation – was included in the validation set if it could be tracked visually through successive cut planes in the 3DRA volume data. Despite the fact that RBT produces excellent results on the carotid artery, it was excluded because its radius is too large for the MHT algorithm. A total of 744 *vessels* and between 55-96 *vessels* per patient were examined.

3.9.5 Parametrization of the reference method: MHT

In one patient (Patient 1), we randomly selected 15 vessels of different sizes. The manual seed C_0 and direction \vec{d}_0 were used to track each branch (Fig. 3.15). Following a trial and error method, the parameters were tuned so that the algorithm produced satisfactory trackings in most of the vessels. A visual assessment was carried out, focused on a correct delineation of the targeted vessel centerline, for determining the pertinence of the parameters set. The best configuration was kept and used for validation.

MHT algorithm proved to be quite sensitive to the allowable bounds set onto the radius. To have MHT work on the most possible vessels, the radius was allowed to vary between 0.25 to 1.25 times the estimated initial radius r_0 .

The single hypothesis tracking option was set (one single vessel to track). The tracking step was set to 0.1 the radius of current tracking radius in the MHT, the maximum search angle angle between 2 tracking steps to 85° and the number of search angles was 5. A maximum of 2000 iterations were allowed and all other parameters initialized to default values.

3.9.6 Parametrization of our algorithm: RBT

Our aim was to find the best configuration for RBT, that is the parameters set that works in almost all cases. First of all, an initial candidate set of parameters was retrieved. The RBT parameters were tuned on the same 15 vessels as for MHT. The resulting centerlines as well as the points at the vessel surface were inspected visually for each configuration. When no errors – segmentation leakage or small tracked lengths – were observed, the configuration was taken into account. Among all the configurations, the best configuration was selected and this procedure led to the initial values in Table 3.4. A brief description of the parameters is also given in Table 3.1.

Secondly, following the same outline as mentioned previously, interest was paid to the optimality of these values. Since a visual assessment was also involved, we reduced our batch of study to 20 vessels (two per patient) from the data-set. Representative

Parameter	p_{inl} (%)	r_t (%)	N_r	N_d	N_p	N_{min}	N_{max}
Initial	70	10	162	81	128	220	500

Table 3.4: RBT initial configuration: values tuned on twenty vessels from the data set.

cases were chosen according to different criteria: length, possible perturbations which could mislead the tracking (fuzziness, kissing vessels, gaps and tortuosity as displayed in Fig. 3.16) and vessels width. Six parameters were considered in this evaluation: N_r , N_d , N_p , r_t , p_{inl} , and N_{min} . Only N_{max} was excluded and set to 500, except for N_{min} evaluation where it was set to 1000.

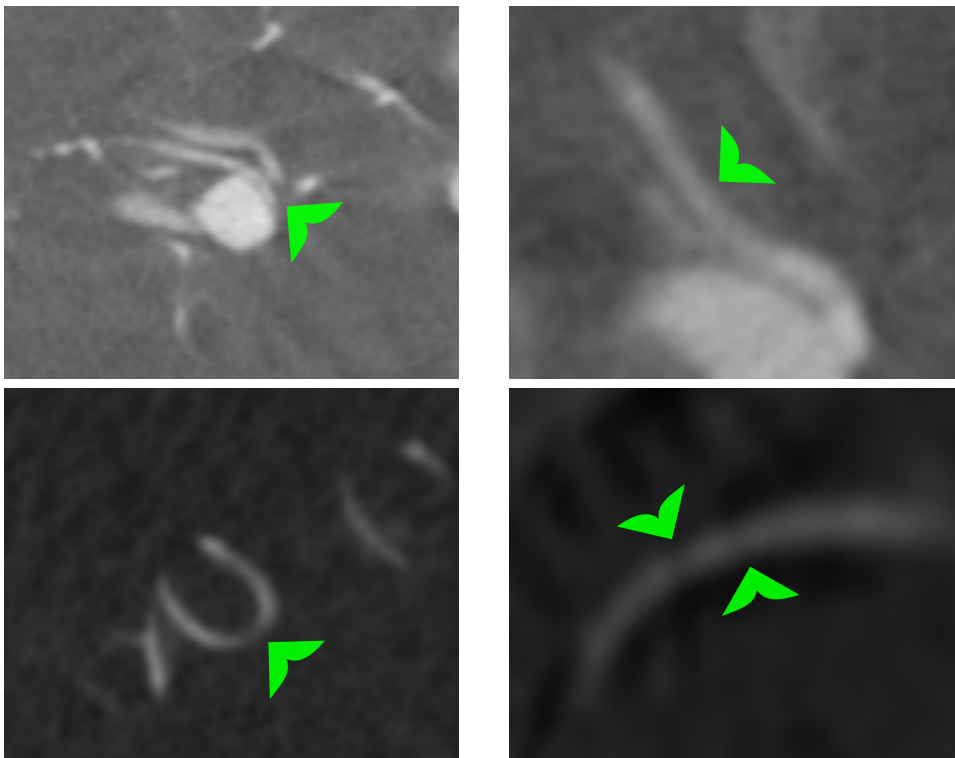


Figure 3.16: Four of the twenty vessels used for tuning the RBT parameters which exhibit kissing vessel issues (top left), fuzziness (top right), tortuosity (bottom left) and drops in image intensity (bottom right).

To find out which parameters have a significant impact on our algorithm's performance, the idea was to change one degree of freedom (one parameter) from a stable (initial) configuration and analyze its impact on the resulting centerline by visually inspecting and counting the number of successful trackings. A tracking was accounted as successful if the extracted centerline and points at the vessel surface delineated the targeted vessel from the seed point to the end of the vessel. Otherwise, trackings were considered as failures. One test case consisted of a tracking instance where five parameters were set to initial values in Table 3.4 and the remaining parameter took its values within a pre-computed set of values. This set was depended on the parameter of interest as follows: $N_r \in \{42, 162, 642\}$, $N_d \in \{21, 81, 321\}$, $N_p \in [32, 256]$ (step of 32), $r_t \in [3, 30]$

Parameter	p_{inl} (%)	r_t (%)	N_r	N_d	N_p	N_{min}	N_{max}
Proposed ranges	65-75	10-15	82-161	82-321	128-192	300-350	600-700
Optimal	65	12	162	321	128	300	600

Table 3.5: Proposed ranges for **RBT** parameters according to the success rate scored in 20 vessels from the data-set. An optimal configuration is made up from this ranges.

Patient	1	2	3	4	5	6	7	8	9	10	avg (%)
Time %	53	40	34	20	34	23	17	23	14	52	31
Length %	16	0	-2	4	14	-3	5	-2	13	23	7

Table 3.6: Vascular tree of ten patients tracked with the initial (Table 3.4) and optimal (Table 3.5) configurations. Increase percentage in time and length between both configurations. In average, optimal values increase length and time tracking by 7% and 31% respectively.

(step of 3), $p_{inl} \in [50, 100]$ (step of 5), and $N_{min} \in [25, 500]$ (step of 25). This evaluation on the success rate of **RBT** to track the 20 vessels with different configurations put forth the ranges in Table 3.5. Besides, we also noted that N_r , N_d and the p_{inl} had a significant impact on the computation time.

Finally, we evaluated the performance of the optimal configuration (Table 3.5) with that of the initial configuration (Table 3.4). For this study, we used the entire data-set and recomputed the success rate for both configurations. Moreover, the time and the tracked length were also considered in this assessment. The length was measured as the cumulative length of segments composing the detected centerline and the time represented the quantity in seconds needed to produce the result. Table 3.6 summarizes the increase in time and tracked length brought about by the optimal configuration *w.r.t* the initial configuration. The increase percentage in time and length was respectively 31% and 7%, in average. This raise in computation time was explained by the fact that the main difference between both configuration resides solely in the values for N_d and N_r . Note that N_{min} and consequently N_{max} , do not reflect this augmentation. For the test case with the twenty vessels, configuration with N_{min} set to 220 and 300 – N_{max} was set to 1000 iterations – put forth no significant differences in the computation time.

In final consideration, the performances produced with the initial configuration of parameters displayed similar results on the entire data-set for less computational effort. Henceforth, we favored the initial set of values (Tab. 3.4 or Tab. 3.1) for the remainder of the evaluation. Finally, **RBT** was run with the manually annotated C_0 , \vec{d}_0 and r_0 for each vessel and the aforementioned configuration.

3.9.7 Evaluation measures

Three different measures were applied to each *vessel* of the annotated data-set: success rate which quantifies the number of vessels successfully tracked by a particular method, the tracked length which accounts for the capability of an algorithm to track a vessel, and the accuracy which informs how different is the **RBT** centerline with respect to the reference method (**MHT**).

Success Rate (SR). The first experiment covered the qualitative assessment, we visually appraised the tracking success. A tracking was considered as successful if the resulting centerline remained inside the vessel of interest. Otherwise, the result was considered as a failure. A successful tracking was not necessary one that tracked the entire vessel. For instance, the first row of Fig. 3.17 depicts a situation where **RBT** tracked successfully the vessel and stopped at a correct location. Despite the fact that **MHT** tracked until the end of the vessel, it produced a segmentation leakage and the tracking was considered as a failure. This first evaluation resulted in a **Success Rate (SR)** in percent.

However, the first examination was tough for **MHT**. In some cases, rated as failures, the **MHT** successfully tracked the proximal section of the vessel, but astray trackings – or *leakage* – were commonly observed where neighboring dense structures locally misled the tracking (Fig.3.18). In contrast, **RBT** rather stopped than produced segmentation leakage. But this property of **RBT** produced a certain number of instances where only a small portion of the visually inspected vessel length was tracked.

Tracked Length (TL). In a second experiment, these **MHT** tracking results, considered as failures in a first place, were visually, manually cut before the problem occurred (refer to Fig. 3.17). No such cut was ever required for **RBT**. Then, according to the length of the detected centerline, results, both for **MHT** and **RBT**, were classified as: *short*, *medium*, *long* if the centerline was respectively correct along less than 33%, 33-66%, more than 66% of the visually estimated length of the targeted vessel. A second evaluation was therefore performed, recomputing the success rate, while only taking into account the *medium* and *long* trackings. The tracking considered as *short* were discarded since we aimed at reporting significant figures. Moreover, tracking lengths were computed and compared.

Accuracy (Ac). A third evaluation was carried out, focusing on a quantitative assessment of the centerline detection accuracy with respect to **MHT** centerline. For this study, all trackings – regardless of whether they were considered as failures or too short for both algorithms – were used for accuracy evaluation. The reason for reintroducing failures is that though **MHT** may go astray, it sometimes came back to the right vessel of interest and tracked until its end, as illustrated in the last row of Fig. 3.17.

Figure 3.17: Top: (left) Maximum Intensity Projection of one patient data. Centerline delineation produced by **MHT** (middle) and **RBT** (right). Bottom: (left) When the **MHT** centerline (red) does not remain inside the vessel wall, it is cut – during the tracked length computation – before the problem arises (green). (middle) The whole **MHT** centerline (red) and the points on this centerline (cyan) selected for **ASSD** computation ($w = \bar{\rho}$). (right) The resulting points for **MHT** (red) and **RBT** (blue) involved in the **ASSD** computation.

The accuracy measure was based on a point-to-point correspondence between the reference (**MHT**) and our evaluated centerline. To avoid including errors that may be due to leakage, for example, its computation required to find the centerline sections that were potentially common to both. To cope with this condition, a variant of the **Average**

Symmetric Surface Distance (ASSD) Schaap et al. (2009) was used. More precisely, we added a threshold condition on the distance. The **ASSD** measure translates the mean error between two centerlines (refer to the last row of Fig. 3.17).

Let $\mathcal{P} = \{P_i\}_{i=1,\dots,N}$ and $\mathcal{Q} = \{Q_j\}_{j=1,\dots,M}$ be respectively the set of points composing the centerline of **MHT** and **RBT**. Firstly, the closest point P_i in \mathcal{Q} is computed using the Euclidean distance $d(.,.)$:

$$d(P_i, \mathcal{Q}) = \min_j d(P_i, Q_j) \quad (3.8)$$

$d(P_i, \mathcal{Q})$ is accounted if it is less than a threshold distance w , designated as the restricted distance:

$$d_w(P_i, \mathcal{Q}) = \begin{cases} d(P_i, \mathcal{Q}) & \text{if } d(P_i, \mathcal{Q}) < w \\ 0 & \text{else} \end{cases} \quad (3.9)$$

Symmetrically, the same procedure is conducted on the other centerline leading to $d_w(Q_j, \mathcal{P})$, in order to find the closest point to Q_j . Lastly, both restricted distances are averaged: **MHT** selects $N' = |\{P_i / d(P_i, \mathcal{Q}) < w\}|$, where $|\{\cdot\}|$ is the cardinal, points on the **RBT** centerline and in turn, **RBT** selects $M' = |\{Q_j / d(Q_j, \mathcal{P}) < w\}|$ on the **MHT** centerline. The last row of Fig. 3.17 exemplifies the selected points for both centerlines. The **ASSD** was thereafter computed as follows:

$$ASSD = \frac{1}{N' + M'} \left\{ \sum_i^N d_w(P_i, \mathcal{Q}) + \sum_j^M d_w(Q_j, \mathcal{P}) \right\} \quad (3.10)$$

A major concern in measuring the centerline detection accuracy is to distinguish between correct and wrong portions of a tracked vessel. On one hand, **MHT** centerline is considered as the reference but we noticed that it might produce wrong trackings for certain portions of a vessel (Fig. 3.17 bottom row). Therefore, it is compulsory to select the correct portions for validation. On the other hand, **RBT**'s centerline detection is to be assessed and wrong portions are not to be discarded.

RBT does not produce false positives (no leakage) and it was extensively assessed (see below, Section 3.10 (SR)), as well as its tracking capability, in the previous experiments. Owing to these remarks, **ASSD** works well in our context since **RBT** suggests wrong portions produced by **MHT**. Therefore, the **RBT** precision validation turned out to be the accuracy of its centerline *w.r.t* a reference centerline that potentially contained false positives. The usage of different values for w allows us to verify that **RBT** centerline does not hold false positives, at least not in a significant rate. Lets consider a fixed value for $w = 3$ voxels. This threshold is not suited for small vessels (vessel width of 4-6 voxels) since portions on **RBT** centerline corrupted by leakage, or some part of it, are accounted during **ASSD** computation. In fact, a more adapted threshold to the vessel width is the average radius of the vessel $w = \bar{\rho}$. With an adaptive threshold, risks of accounting corrupted portions by leakage are dramatically reduced.

As a result, two **ASSD** measures were carried out with different distance thresholds : $w = 3$ voxels, which is suitable for large vessels, and $w = \bar{\rho}$, which is a more appropriate choice when considering the scale of vessels and avoid taking into account erroneous

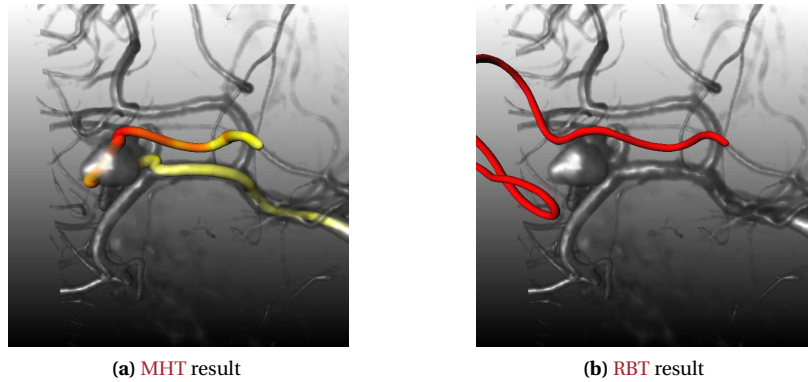


Figure 3.18: Comparison of **MHT** (left) and our algorithm (right) with regard to a **KV** case on Patient 1: a blood vessel runs along the aneurysm sack. Our **RBT** algorithm successfully handles this case, while **MHT** is perturbed by the aneurysm lumen which misleads the tracking.

centerlines at leaking portions on small vessels. For the latter measure, the estimation of the mean radius $\bar{\rho}$ of the targeted vessel was given by our algorithm, since it visually appeared more trustful than **MHT**'s estimation (see radius estimation issue with **MHT** on Fig. 3.22). Last but not least, if the comparison between both thresholds leads to similar figures and in addition, they are below the voxel size, we can assuredly conclude that **RBT** produces a centerline as accurate as **MHT**.

3.10 Results

Success Rate (SR). For the visual assessment, the overall performance of **RBT** algorithm was 94%, confirming its efficiency and strength. In contrast, the **MHT** algorithm reached a **SR** of 65%. Table 3.8 reports the respective **SR** for this experiment. This rather poor performance for **MHT** is the resultant of a lack of efficient mechanism for rejecting strong vesselness responses in unrelated vessels (kissing vessels). Leakages, due to neighboring structures, were the cause of many failures. On the contrary, the **RBT** procedure dealt well with this kissing vessel problem. Fig. 3.18 illustrates the difficulty for **MHT** to sometimes handle the kissing vessel problem and the robustness of **RBT** to such topological difficulties. The **RBT** success when striving with kissing vessels dwells simultaneously on two principles: first, the vesselness relies on the existence of a valid cylinder that locally fits the vessel and second, the cylinder should be relatively similar to its predecessor on the centerline. In contrast, **MHT** constructs a hypothesis tree where each branch represents a possible path. Then, it solely discriminates the hypothesis by thresholding the averaged score – i.e. vesselness response – of a branch. This pruning technique is insufficient to distinguish kissing vessels since neighboring structures may display high vesselness response, and subsequently, leading **MHT** to choose amiss paths. In other words, **MHT** disregards the information encoded on the already tracked centerline to improve robustness and only relies on thresholding in incoming vesselness information.

Tracked Length (Tracked Length (TL) & Common path Length (CL)). For the sec-

ond experiment, Table 3.9 summarizes the **SR** regarding the actual tracked length. In particular, short trackings that were accounted as successes in the visual assessment, were here discarded. As expected, **MHT** increased its performance up to 69%, since some trackings were previously rated as failures due to leakage issues, though they were correct on the proximal vessel section. On the contrary, **RBT** performance slightly dropped to 89%. This abatement could be mostly explained by **RBT** stopping prematurely due to drops in image intensity or to a high inlier rate threshold which was not suitable for some situations, as illustrated in Fig. 3.21. When both algorithms succeeded, on average, **RBT** went further ($CL = 725.5$ mm) than **MHT** ($CL = 680.9$ mm). Nonetheless, **MHT** definitively delivered a lesser number of successful trackings which impacted negatively the average tracked length per patient down to 778.6 mm against 1446.1 mm for **RBT**.

Accuracy (Ac). Finally, table 3.9 also reports quantitative **ASSD** measures. The average **ASSD** between both centerlines, with both 3 voxels and the mean radius of the targeted vessel, was below one voxel for all patients. This showed that the proposed method produced similar results as the **MHT**.

Time. As an illustration, we computed the time and the number of instances needed by **RBT** to track an entire vascular tree. The user manually placed a seed at the stem vessel of the vascular tree and branches were added as described in Section 3.6. For this purpose, a dedicated user interface was used which is further introduced in Appendix A. Furthermore, no aneurysms were introduced in the segmentation process. In average, it took roughly 30 minutes per patient to fulfill only the tracking computation (Table 3.7). No quantification was carried out on the time spent during user-interaction transactions. Based on our own experience, the user spends significant time depending on his/her experience and on the targeted vessel. Indeed, small vessels are more difficult

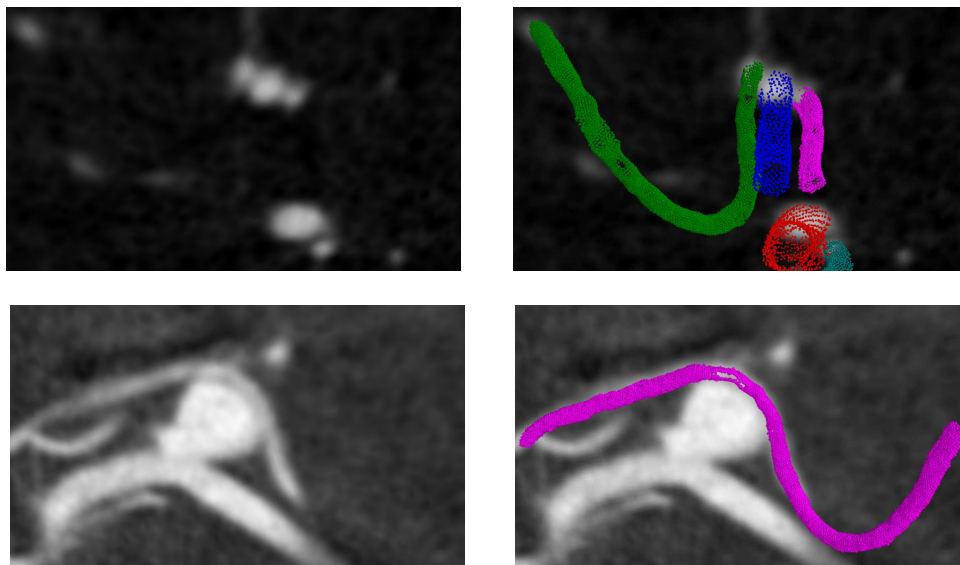


Figure 3.19: (left) Original image data for three kissing vessels (top) and a vessel running along an aneurysm (bottom). (right) The corresponding segmentation with **RBT** (same case as in Fig. 3.18).

Patient	1	2	3	4	5	6	7	8	9	10	Average
Time (min)	31.8	33.3	27.9	24.0	40.7	22.2	23.1	17.8	35.7	26.4	28.3
Number of instances	40	39	44	37	39	60	25	57	33	42	41.6

Table 3.7: Time, in minutes, and number of **RBT** instances needed to track a complete vascular tree. Overall, one cylinder computation took around 0.01 – 0.3 seconds.

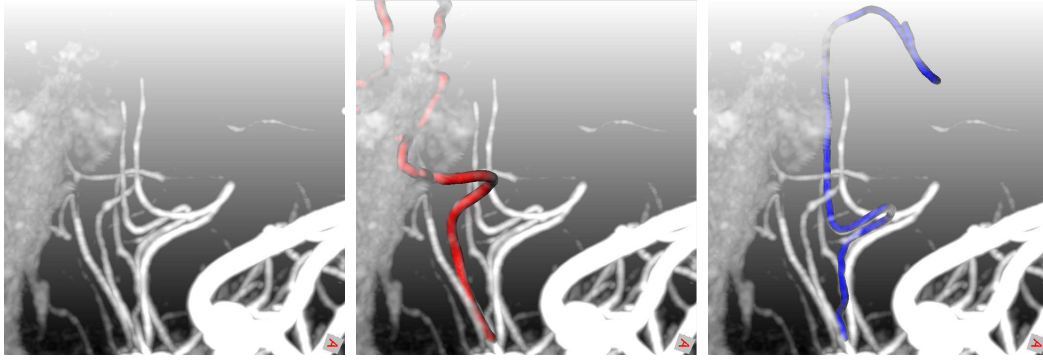


Figure 3.20: (left) A tortuous branch of the posterior choroidal artery (0.6 – 0.7 mm of diameter) in Patient 3. (middle) **MHT** is misled by neighboring structures whereas (right) **RBT** tracks the tortuous vessel.

to locate than large vessels.

3.11 Discussion

The results presented in the last section confirm our expectations that a bounded ray-casting procedure coupled with geometrical prior and a robust estimator allowed **RBT** to handle nicely the kissing vessel problem (Fig. 3.19). In contrast, the lack of a consistent pruning – while considering several strong vesselness responses – impedes **MHT** to strike these issues.

When handling vessels of varied diameters, **RBT** puts forth its robustness to tackle these scenarios with the same set of parameters. Technically speaking, **MHT** was not designed to handle vessels of different sizes since it is grounded on a finite set of filters. In other words, a large range of vessel widths demands a higher number of filters without mentioning a higher computational burden. In contrast, the computation time for **RBT** does not hold on the vessel size but in the complexity of the local morphology of the vessel. For instance, the tracking of the main stem vessel of Patient 4 (carotid artery) was run in 16 seconds for a tracked length of 165 mm while the tortuous vessel in Fig. 3.20 of the same patient was achieved in 69 seconds for a tracked length of 64 mm. Besides, **RBT** is robust to bad initialization on the vessel width which is not the case for **MHT**. A propos this limitation, the user-defined radius estimation bound could lead to a saturation of the radius estimation and ineluctably to a wrong estimation of the centerline (Fig. 3.22). On the contrary, **RBT** incorporates automatically radius variations along the vessel by considering the last estimated radius in the centerline to restrain the estimate and softly adjust to these changes.

The **RBT** algorithm reached its limits when struggling with drops in image intensity (Fig. 3.21). Incidentally, **RBT** explores the vessel surface within a limited area. In-

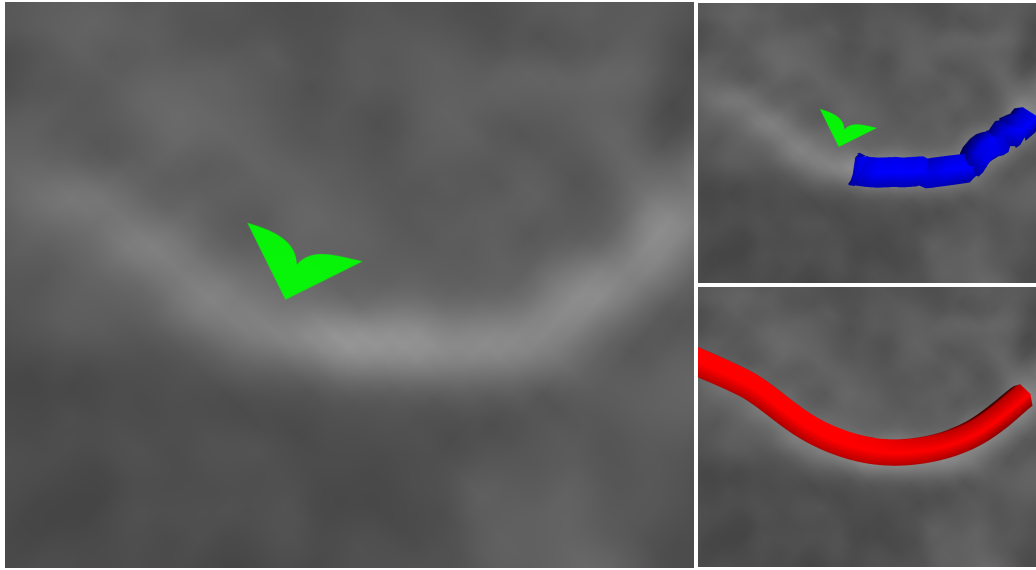


Figure 3.21: **MHT** deals well with drops in image intensity. (left) A cut plane along a vessel presenting drops in image intensity; (right top) **RBT** result: the tracking stops too early. (right bottom) The resulting centerline for **MHT**.

Patient	1	2	3	4	5	6	7	8	9	10	Average
Number of vessels	59	57	75	69	82	80	55	95	76	96	74.4
MHT %	67	72	66	74	60	59	48	67	74	62	65
RBT %	97	90	99	81	100	96	85	92	96	99	94

Table 3.8: Success rates of **MHT** and **RBT** on a data set of 10 patients. A total of 744 test vessels were used to evaluate both algorithms. The average success rate for **MHT** and for our algorithm is respectively 65% and 94%.

stead, the **MHT** algorithm coped especially well with gaps thanks to its search-depth scheme for inspecting further positions in the centerline.

One of the other reasons we identified as potentially impeding **MHT** performances is the fixed tracking step that it uses. The increase in resolution in **3DRA** implies that we try to track very small vessels (0.5 mm in diameter or less). Such vessels present with a very high tortuosity and potentially acute bends Fig. 3.20. **RBT** defines the tracking step as the quarter of the height of the last fitted cylinder. But since the height is adapted to the longitudinal extension of the inlier point set, fitted cylinders are shorter in twisted and bent portions of the blood vessels, thus naturally adapting the tracking step.

Last but not least, the reference method was very sensitive to bad initialization since some instances in the validation data-set did not breed any tracking or the tracking started inside neighboring structures. Conversely, **RBT** was as yet robust to bad initialization when the initial radius was scaled to the targeted vessel as illustrated in past Section 3.5 (Fig. 3.8). Then whence the possibility to manually add a new branch with just one user-defined point.

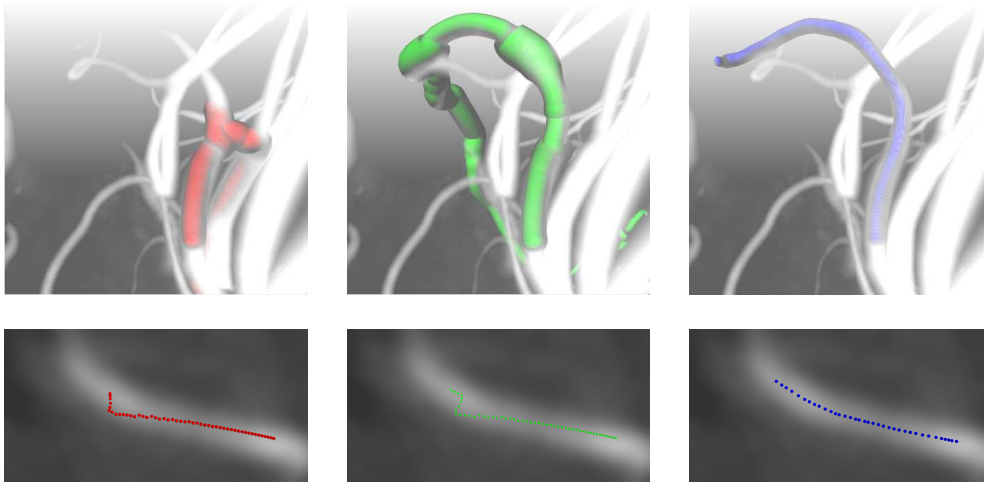


Figure 3.22: (Top) Influence of user-defined radius estimation bound on **MHT** tracking where underestimation of the vessel radius leads to an inaccurate outcome on Patient 5: (left) the upper bound is reached leading to *leakage issues* and (middle) computation with larger values avoid this problem. (right) Our **RBT** algorithm successfully tracks the vessel without resorting to this information. (Bottom) Plane views: the detection of the centerline for the **MHT** on a fuzzy portion of 0.8 mm of diameter, with saturation of the radius (left) and without saturation (middle), is disturbed whereas our algorithm detects correctly the centerline

Patient	1	2	3	4	5	6	7	8	9	10
MHT										
M & L (%)	75	71	73	75	65	65	57	67	77	63
TL (mm)	765.2	601.5	758.7	726.5	818.7	1086.9	555.4	906.1	688.1	879.0
CL (mm)	620.7	523.3	697.5	512.4	810.2	1074.1	389.0	834.1	581.8	766.3
RBT										
M & L (%)	94	79	97	77	99	95	82	84	89	91
TL (mm)	1120.4	1063.6	1383.3	1097.6	1752.7	2373.6	833.9	2121.3	1420.6	1294.0
CL (mm)	662.9	540.5	731.2	515.9	915.2	1164.7	403.6	914.2	644.7	782.6
ASSD										
$w = 3$ (mm)	0.20	0.21	0.16	0.17	0.16	0.16	0.18	0.19	0.16	0.20
$w = \bar{\rho}$ (mm)	0.20	0.20	0.16	0.17	0.16	0.16	0.19	0.19	0.16	0.20

Table 3.9: Success rate of **MHT** and **RBT** algorithms with tracking results classified as *medium* and *long* (M & L). The mean success rates for **MHT** and for our **RBT** algorithm are respectively 69% and 89%. Furthermore, the mean **ASSD**, with a distance threshold of 3 voxels ($w = 3$) and the mean radius of the targeted vessel ($w = \bar{\rho}$), between the extracted centerlines by both methods is below one voxel (0.17-0.18 mm). The **CL** corresponds to the total length of centerline used to compute **ASSD** on vessels successfully tracked by both methods. On average the total tracked lengths (**TL**) for **MHT** and **RBT** are respectively 778.6 mm and 1446.1 mm.

3.12 Conclusion

The general context of our work is interventional neuroradiology. While most previous works present segmentation results on **Computed Tomography Angiography (CTA)** or **Magnetic Resonance Angiography (MRA)** data, we only addressed the segmentation of **3DRA** volume. Indeed, **3DRA** is the modality of choice for interventional radiologists that routinely acquire **3DRA** data before, during and after the treatment. While **3DRA** is arguably of a higher quality than **CTA** and **MRA**, which would supposedly ease the segmentation task, noise and artifacts are still present. Moreover, even in **3DRA**, and as

in **MRA** and **CTA**, our aim remains the segmentation of vessels whose radius downs to the voxel size.

State-of-the-art **MHT** vessel segmentation algorithm (Friman et al. (2010)) was used as a reference for validation. The **MHT** performances were below our expectations, when applied to **3DRA**. This supports our claim that **3DRA** presents with specific issues regarding blood vessel segmentation: improved blood vessel visibility implies more kissing vessels, and vessels with a much stronger tortuosity.

In this chapter, we proposed a novel algorithm for blood vessel segmentation, called **RBT**, which builds a vessel tree structure in two steps. First, an estimation step which comprehends a ray-casting procedure coupled with a **RANSAC**-based cylinder fitting process to handle outliers in candidate points at the local vessel surface. Second, a prediction step of the next point on the centerline which adapts the tracking to the local morphology of the vessel.

A large part of the work was dedicated to the validation framework and effort to fairly compare **RBT** against **MHT**. A considerable lot of effort was dedicated to manually editing **MHT** results, visual assessment and user-interaction. It is worth pointing out that 744 vessels on 10 **3DRA** patient data were manually annotated. Three measures were presented in our centerline evaluation pipeline: **SR**, centered on successful tracking, **TL & CL**, concentrated on the extraction capability of both algorithms and **ASSD**, focused on the accuracy of **RBT w.r.t** the **MHT** centerline. The results demonstrated the strength of the proposed method as well as its ability to accurately detect centerlines. In particular, **RBT** proved to be able to capture very complex vascular topologies and to be very robust to the kissing vessel issue. In addition, it tracked very tortuous tiny blood vessels. As such, it improves upon previous blood vessel tracking algorithms.

Besides, **RBT** is also able to continue vessel tracking past bifurcations and is also very robust to poor initialization when the radius estimation is well-scaled. After tracking a vessel, a child vessel is segmented by providing a seed point on the parent vessel. This point is connected to its closest point on the centerline of the parent vessel, which provides an initial direction for the tracking. The initial radius is the same as that of the parent vessel at the connecting point. This paradigm to add new branches did not work with **MHT**, that required tighter initial parameter values. Furthermore, **RBT** was even robust enough for us to use the same set of parameters (aside from center, direction and radius) for all patients and vessels. These characteristics of our method minimize user interaction, allow for segmentation of vessels of interest to the user and counterbalance the fact that **RBT** does not automatically handles bifurcations. We would even argue that it is a clinical advantage since the radiologist is often interested in only a part of the vasculature (the one related to the pathology), that he/she would probably like to choose and control. Anyway, user interaction would be necessary, even with a so-called fully automated procedure, to correct for potential errors. Our expectations were proved in practice by developing a dedicated user interface allowing the user to easily segment a vascular tree with one click per branch (refer to Appendix. A for a brief overview).

The proposed segmentation algorithm seems to be easily applicable to other modalities by taking in consideration a pre-processing step with regard to the targeted modality, e.g. a skull removal process for **CTA** and **MRA**. This technique was also employed



Figure 3.23: Our **RBT** algorithm applied to **MRA** data (left). The dense sampling produced on 37 vessels (middle) and the tracked vessels superimposed upon the original data (right).

by **Friman et al. (2010)** when applying **MHT** to **CTA** data. Preliminary outcomes showed a promising behavior of our algorithm on **CTA** and **MRA** (Fig. 3.23).

A path to improve **RBT** relies on a recognized strength of **MHT**. **MHT** allows the tracking to locally degrade the fitting score, as long as a raise occurs. As a result, **MHT** resists to local drops in the density along a vessel. However, this strategy prevents **MHT** from stopping per se as it too greedily follows hints of vesselness on the data. Our method stops as soon as no cylinder is found with the minimum percentage of inliers and may occasionally stops prematurely. In order to increase the robustness of our proposal, we need to integrate an exploration phase during the tracking procedure. A difficult compromise would have to be found between our stopping criteria and a greedy **MHT** exploration.

A final word should be said on the segmentation characteristic of **RBT**. It is a tracking algorithm since the vessel centerlines are extracted, building a vessel tree with little manual interaction. But, as **MHT**, the local vessel radius is also estimated. However, **RBT** goes further since it also provides a set of points that reliably lie on the local vessel surface. These points provide a rather dense sampling of the vessel surface. The centerline produced by **RBT** serves as input data for our reconstruction algorithm which locally reconstructs the blood vessel surface. This dedicated algorithm is treated extensively in the next chapter.

BLOOD VESSEL RECONSTRUCTION

*In the context of computer-based simulation, contact management requires an accurate, smooth, but still efficient surface model for the blood vessels. In this chapter, we propose a new blood vessel surface model provided by a novel paradigm, namely **Local Implicit Modeling (LIM)**. **LIM** locally reconstructs the blood vessel surface thanks to information provided by its centerline and a dense sampling of the vessel surface (point-set). In other words, **LIM** locally performs the point-set reconstruction by fitting a skeleton-based implicit function – i.e. **Blobby Model (BM)** – through energy minimization, alternating with an original selection and subdivision scheme of **BM** primitives (iterative process). The idea of **LIM** is to enrich the blood vessel centerline by attaching a **BM** to a point on the centerline that locally reconstructs the vessel surface. The resulting blood vessel surface is supplied as a tree of local implicit surfaces generated by **BMs**. Our proposal is very efficient for simulation and was shown to provide a compact representation and a sub-voxel approximation of the vessel surface on 10 patients.*

4.1 Introduction

In the previous chapter, we explored a new paradigm for retrieving the topology of the vasculature and a local scattered representation of the blood vessel surface. This was achieved through a **RANSAC-Based Tracking (RBT)** procedure which uses cylinders as priors for the vessel shape. However, this scattered representation is insufficient for our simulation purpose. Indeed, a point-set representation should be further treated to ensure key features during simulation: realism, fast collision detection and contact response computation.

From a geometrical stand point of view, realism – during simulation – is ensured by an accurate and smooth representation of the blood vessel surface. Topology should also be available e.g. for **Kissing Vessel (KV)** issues disambiguation. Furthermore, the geometrical model – during simulation – should guarantee a fast and robust collision detection: access to in/out test and differential quantities for contact response computation such as the normal to the surface and the gradient of the distance function (Dequidt et al. (2007)).

In this chapter, we present a **Local Implicit Modeling (LIM)** technique for reaching these goals. Surface reconstruction from scattered data with implicit functions has proven effective in dealing with noisy and/or incomplete data (Carr et al. (2001); Ohtake et al. (2006); Kazhdan et al. (2006) and Mullen et al. (2010)). Despite a large number of sound methods for reconstructing scattered data with implicit functions, no direct method is applicable to our situation. Among all models, skeleton-based implicit surfaces were chosen for their compelling characteristics in our context. Implicit surfaces generated by skeletons are particularly well-suited for modeling smooth free-form shapes in a very compact way (Bittar et al. (1995)). Besides, they present a nice feature designated as *locality* which means that objects that are far apart should have no – or at least negligible – influence on each other (Dekkers et al. (2004)). Last but not least, skeletal surfaces also confer high shape modeling flexibility (Bittar et al. (1995)).

One of the most common approaches for reconstruction consists in deforming a surface or a volume in order to fit an input point-set. The idea has been widely used and developed in late 20 years. Following the seminal work of Muraki (1991) in surface reconstruction from scattered data, LIM defines the surface fitting – or model fitting – as an energy minimization problem, alternating with a model refinement as in Tsingos et al. (1995), but improving over this latter work. The idea of LIM is to provide an implicit, local representation of the blood vessel surface for each point on the centerline (Fig. 4.1). As a result, our algorithm requires that the vessel centerline is extracted as a tree (Fig. 4.1a), not necessarily dense, and that a local vessel radius estimate is available at each point on this centerline.

Our aim is to fit each local point-set \mathcal{P} with an implicit surface generated by a point-set skeleton (Fig. 4.1f). Within this vista, we explore theoretical and technical choices in our implicit formulation for satisfying simulation requirements (Sec. 4.2 and 4.3). Furthermore, each local model refinement is the resultant of an accuracy pursuit during reconstruction (Sec. 4.4). Besides, the resulting local models are organized under the same tree structure as the point centerline (Sec. 4.5). Solving contact constraints at each

point of the interventional tool, is performed using the appropriate local implicit model as the blood vessel surface (Sec. 4.6).

Three main reasons led our proceeding to propose a LIM paradigm. First, a straightforward reconstruction of the vasculature with implicit functions brings about an endemic issue, namely blending issues (Oeltze and Preim (2005); Schumann et al. (2008)). Whereas blending is a nice property of implicit functions, KVs tend to merge in a single surface. To control the blending, local strategies were adopted so that the scalar field is only modified at desired zones (Bernhardt et al. (2010); Gourmel et al. (2012)). Despite this procedure handles objects that are physically in contact (no space in-between), it remains difficult to be fulfilled when more than two objects are merging; user-interaction is required for tuning the blending function and thus avoiding extra artifacts. In the particular case of blood vessel reconstruction, Hong et al. (2012) proposed to reconstruct vessels' cross-sections with generalized cylinders and to use a global blending function to reconstruct the whole vascular tree. This approach presents an endemic issues when dealing with collisions. Indeed, the interventional tool may jump from one vessel to another between two simulation steps (see Chapter 2, Fig. ??). Second, direct collision detection with a complete geometrical model will be time consuming and not efficient. Efficiency may be gained by introducing extra machinery – like state-of-the-art Bounding Volume Hierarchies (BVH) (Teschner et al. (2005)) but these techniques are not well-suited to tree-like structures since the bounding volume is a poor approximation of the enclosed vessels, but at the finest resolution. As a by-product, a small ratio of the bounding volume is occupied by the vessel. Increasing the depth helps but induces a loss of efficiency. This latter remark leads us to our third argument. Even if we can perform fast inclusion/exclusion test (whether a point is inside or outside the surface) this procedure does not resolve the KV issue during simulation. One way to tackle this problem is to introduce the topology into play. Besides, topological information finds its capital gain during collision detection and its management (Li et al. (2012)). As a result, we follow the seminal idea of Bittar et al. (1995) who locally reconstruct organs by means of the medial-axis. Such a strategy confers LIM to handle undesired blending and improve collision detection efficiency during simulation.

For sake of clarity, we define a *node* as a given position on the vascular tree centerline. For instance, the red dot in Fig. 4.1b represents a *node* equipped with a local point-set \mathcal{P} and a cylinder (center, radius, direction). We also define the *topological distance* \mathcal{T} as the lowest number of consecutive positions on the centerline that separate two nodes. As displayed in Fig. 4.1c, the *topological distance* from the bifurcation node (central node) to its closest neighbors (colored nodes) is 1.

Fig. 4.1 displays the main steps in the reconstruction process. LIM's central idea is the local reconstruction of the blood vessel surface. For a given node – or reconstruction node – in the centerline, we aim at providing an implicit surface that locally depicts the blood vessel surface. The extent of this local depiction should be long enough to correctly handle collisions. Section 4.2 details the mathematical formulation of the implicit function and discusses it. At first time, we only consider the reconstruction from a local point-set \mathcal{P} associated to a reconstruction node. The resulting implicit surface – for this node – is the white surface in Fig. 4.1f (further described in Sections 4.3 and 4.4). In fur-

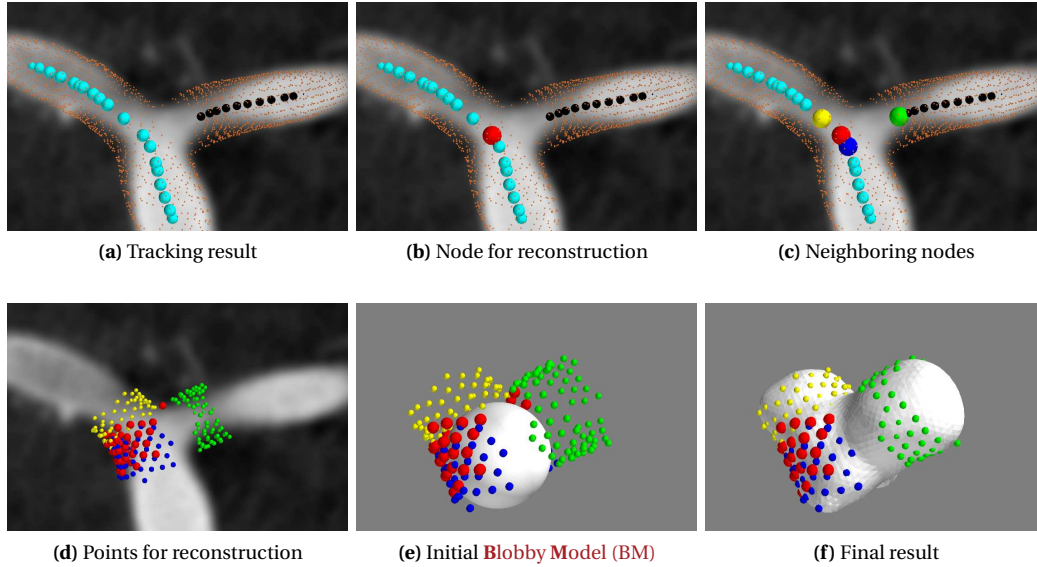


Figure 4.1: Outline of the local reconstruction of the vessel walls. (a) Tracking results for two arteries on **three-Dimensional (3D) Rotational Angiography (RA)** data: Centerlines and extracted points at the arteries surface; (b) Choose one node for reconstructions: in this case, it represents a node at the bifurcation. (c) Neighboring nodes are selected **with respect to (w.r.t)** a topological distance of one. (d) Points belonging to neighboring nodes and the node of interest constitute the point cloud for local reconstruction. (e) Two blobs with width equal to the estimated radius serve as initial **BMs**. (f) Final reconstruction is achieved after fissioning one blob at a time. Thirty subdivisions were necessary to produce the resulting implicit surface.

ther sections (Sec. 4.5), we detail the proceeding for recovering the whole geometrical model of the vascular tree.

4.2 Implicit formulation

4.2.1 Implicit function

From a mathematical standpoint, an implicit iso-surface generated by a point-set skeleton is expressed as the T-level set \mathcal{S} of a function f , a sum of implicit spheres:

$$f(X; p) = \sum_{j=1}^{N_b} \alpha_j \Phi \left(\frac{|X - C_j|}{\rho_j} \right) \quad (4.1)$$

where $X \in \mathbb{R}^3$, $T \in \mathbb{R}^+$ is the iso-surface threshold, $\{\alpha_j\}$ are weights, and $\{C_j\}$ is the point-set skeleton. Each implicit sphere $\#j$ is defined by a symmetric spherical function, centered on C_j , of width ρ_j and following a radial profile. This profile – or kernel – is a function $\Phi: \mathbb{R} \rightarrow \mathbb{R}^+$, rapidly and monotonically decreasing to 0 (Sherstyuk (1999)). $p^T = \{C_j^T, \rho_j\}_{j=1 \dots N_b}$ is the vector – of dimension $4N_b$ – gathering centers and widths of the N_b implicit spheres composing the implicit function. For a depiction of this formulation, let's consider a Gaussian kernel $\Phi(x) = \exp(-\frac{x^2}{2})$ and three blobs. Fig. 4.2 illustrates the

resulting implicit function and its associated T-level \mathcal{S} in **one-Dimension (1D)** and **two-Dimensional (2D)**.

Blinn (1982) presented this family of functions as a generalization of quadrics surfaces. Subsequently, Muraki (1991) was the first to use a Gaussian-based model in the context of object reconstruction. Such models were called differently depending on the kernel used (Sherstyuk (1999)). Following the seminal work of Muraki (1991), we shall use the terms *blob* for an implicit sphere, and **Blobby Models (BMs)** as a generic name for the implicit models.

4.2.2 Kernel choice

Two families of kernels exist: compact and infinite support kernels. The former drops to zero at a certain threshold distance while the latter decreases to 0 at infinity. In our application, we are interested in continuous and continuously derivable kernels since the implicit function f depends entirely on the continuity and derivability of the kernel; recall that we aim at providing the gradient of the approximated distance function at any point in space. Owing to their definition – scalar fields drop to 0 after a certain distance – compact support kernels present nice features: computation efficiency and ray-casting rendering affinity (Kanamori et al. (2008); Rouiller (2011)). However, the gradient vanishes after a certain distance which implies that no trustful gradient of the distance function is available there and subsequently, no recall force can be provided during simulation. On the other hand, infinite support kernels exhibit extra computational burden when evaluating the function value and need more sophisticated algorithms for rendering, e.g. Marching cubes algorithm. Nonetheless, the gradient is available at any point in space, thus providing the sought force during simulation. Since all kernels exhibit a similar bell-shaped profile, one can imagine to take advantage of both types of kernels – namely compact and infinite support kernels – and thus, provide a double representation of the same underlying implicit function.

Sherstyuk (1999) presented a comparative study of kernels applied to convolution surfaces and focused on computational efficiency. Point-based implicit surfaces,

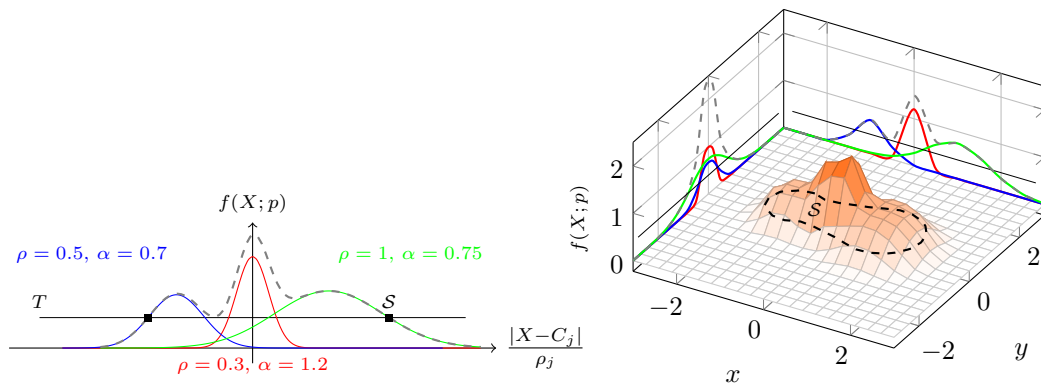


Figure 4.2: Implicit function as a sum of 3 Gaussian kernels in (left) 1D and (right) 2D. The implicit contour S is defined as the locus of points where $f(X; p) = T$.

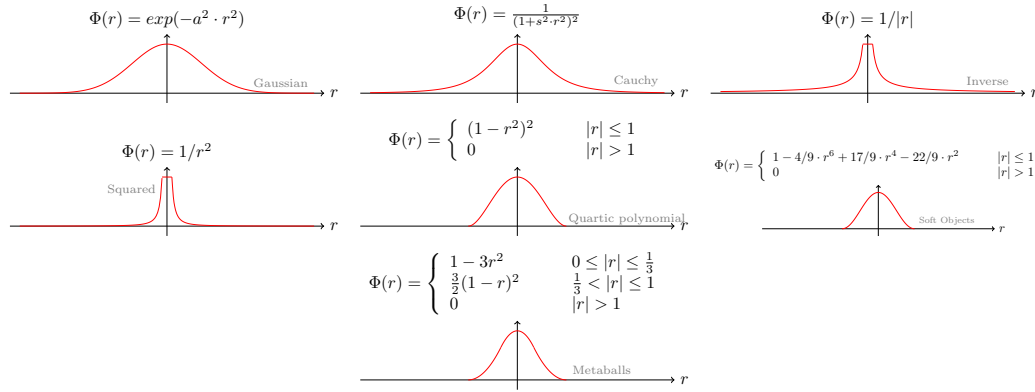


Figure 4.3: Kernels expression and shape.

namely **BMs**, can be seen as a particular case of convolution surfaces. Put another way, **BMs** are skeleton-points or point-based primitives when one refers to convolution surfaces (Bloomenthal and Shoemake (1991) and Bittar et al. (1995)). In this study, five different kernels were revisited within a point primitive context: Gaussian, Cauchy, Inverse, Squared, polynomial – i.e. Quartic polynomial, Soft objects and Metaballs – kernels (Fig. 4.3). Inverse and Squared kernels exhibit singularities at $r = 0$ leading also to discontinuities in their derivatives. Aside from these kernels, polynomial, Gaussian and Cauchy kernels are at least \mathcal{C}^1 . As a matter of fact, Sherstyuk (1999) concluded that Gaussian kernel – in the context of point primitives – is the most expensive kernel for function evaluation. Cauchy kernel turned out to be the second most expensive. In contrast, polynomial kernels were the cheapest among these five kernels. In conclusion, Cauchy kernel is the best choice when considering infinite support kernels, meanwhile polynomial kernel remains the best choice with regard to finite support kernels and computation efficiency.

A final word on kernels, let's dissect the usage of kernels $\Phi(r)$ – introduced in Fig. 4.3 – in Eq. 4.1. Let r be equal to $|P - C|$. We are interested in kernels derivative **w.r.t** P which is needed during the collision handling (contact force computation). Besides, the same reasoning applies to C which is required during the optimization process. When deriving this equation, one can observe that a singularity exists at $P = C$ ($r = 0$):

$$\frac{\partial}{\partial P} \Phi(r) = \Phi'(r) \frac{\partial}{\partial P} r = \Phi'(r) \left(\frac{P - C}{|P - C|} \right) \quad (4.2)$$

To avoid singularities, one can define an auxiliary function ϕ as $\phi(r^2) = \Phi(r)$. In this case, its derivative **w.r.t** P turns out to be:

$$\frac{\partial}{\partial P} \phi(r^2) = 2\phi'(r^2)(P - C) \quad (4.3)$$

Table 4.1 compares both $\Phi(r)$ and $\phi(r^2)$ expressions and their respective derivatives **w.r.t** P . For the remainder of this work, we use the form $\phi(r^2) = \Phi(r)$ where neither ϕ nor ϕ'

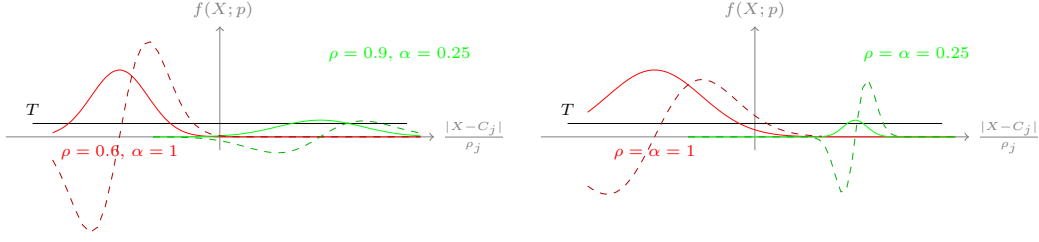


Figure 4.4: Two blobs scalar field functions (solid lines), generated with Gaussian kernels, and their corresponding gradient functions (dashed lines). (left) Blobs with $\alpha_j \neq \rho_j$ exhibit highly different gradient profiles due to scaling factor α_j/ρ_j . (right) Blobs with $\alpha_j = \rho_j$ display similar height variation on their respective gradient functions, i.e. the scalar field shows a constant variation with disregard of blobs width.

present with any singularities at 0. As a result, the new implicit function f is defined as:

$$f(X; p) = \sum_{j=1}^{N_b} \alpha_j \phi \left(\frac{|X - C_j|^2}{\rho_j^2} \right) \quad (4.4)$$

where ϕ is either one of Gaussian, Cauchy, Quartic polynomial or Soft objects kernels.

Kernel		$\Phi(r)$	$\phi(x) (x = r^2)$	$\Phi'(r)$	$\phi'(x)$
Gaussian		$\exp(-a^2 r^2)$	$\exp(-a^2 x)$	$-2a^2 \exp(-a^2 r^2)$	$-a^2 \exp(-a^2 x)$
Cauchy		$\frac{1}{(1+s^2 r^2)^2}$	$\frac{1}{(1+s^2 x)^2}$	$-\frac{4s^2 r}{(1+s^2 r^2)^3}$	$-\frac{2s^2}{(1+s^2 x)^3}$
Inverse		$1/r$	$1/\sqrt{x}$	$-\frac{1}{r^2}$	$-\frac{1}{2\sqrt{x^3}}$
Square		$1/r^2$	$1/x$	$-\frac{2}{r^3}$	$-\frac{1}{x^2}$
Quartic polynomial	$ r \leq 1$ $r > 1$	$(1 - r^2)^2$ 0	$(1 - x)^2$ 0	$-4r(1 - r^2)$ 0	$-2(1 - x)$ 0
Soft objects	$r \leq 1$ $r > 1$	$1 - 4/9 \cdot r^6 +$ $17/9 \cdot r^4 - 22/9 \cdot r^2$ 0	$1 - 4/9 \cdot x^3 +$ $17/9 \cdot x^2 - 22/9 \cdot x$ 0	$2/9 \cdot r(-12 \cdot r^4 +$ $34 \cdot r^2 - 22)$ 0	$2/9(-12 \cdot x^2 +$ $34 \cdot x - 22)$ 0
Metaballs	$0 \leq r \leq \frac{1}{3}$ $\frac{1}{3} < r \leq 1$ $r > 1$	$1 - 3r^2$ $\frac{3}{2}(1 - r)^2$ 0	$1 - 3x$ $\frac{3}{2}(1 - \sqrt{x})^2$ 0	$-6r$ $-3(1 - r)$ 0	-3 $-\frac{3}{\sqrt{x}}(1 - \sqrt{x})$ 0

Table 4.1: Comparative table of $\Phi(r)$ and $\phi(x) (x = r^2)$ kernel expressions of Fig. 4.3 with $r = |P - C|$.

4.2.3 Parameters and locality

In the previous section, we described in details the kernel function ϕ . Eq. 4.4 puts forth five parameters to be tuned per blob (height, width and the center). Now, we analyze the role of these parameters. More precisely, we study the usage of 2 parameters: the height and the width. This study is encouraged by a computational reason since each blob is controlled by 5 degrees of freedom in 3D. Let's dissect the expression of the implicit function and its associated gradient function:

$$\nabla f(X; p) = 2 \sum_{j=1}^{N_b} \left(\frac{\alpha_j}{\rho_j} \right) \cdot \frac{(X - C_j)}{\rho_j} \cdot \phi' \left(\frac{|X - C_j|^2}{\rho_j^2} \right) \quad (4.5)$$

Height α_j and width ρ_j are dual in their role with regard to the implicit function. However, redundancy in parameters can be dismissed by setting $\alpha_j = \rho_j$. By this mean, the gradient vector is independent of α_j , thus vector $\overrightarrow{C_j X}$ is normalized **w.r.t** the size of blobs, i.e. a unique parameter, the width ρ_j . Moreover, the scalar field benefits from a constant variation driven by the width. Fig. 4.4 depicts the gradient normalization of two blobs when setting $\alpha_j = \rho_j$, meanwhile blobs with $\alpha_j \neq \rho_j$ put forth high variations on their respective gradient profiles.

Incidentally, the same impact is noticeable in the derivative of f **w.r.t** C_j :

$$\frac{\partial}{\partial C_j} f(X; p) = -2 \frac{\alpha_j}{\rho_j} \cdot \frac{(X - C_j)}{\rho_j} \cdot \phi' \left(\frac{|X - C_j|^2}{\rho_j^2} \right) \quad (4.6)$$

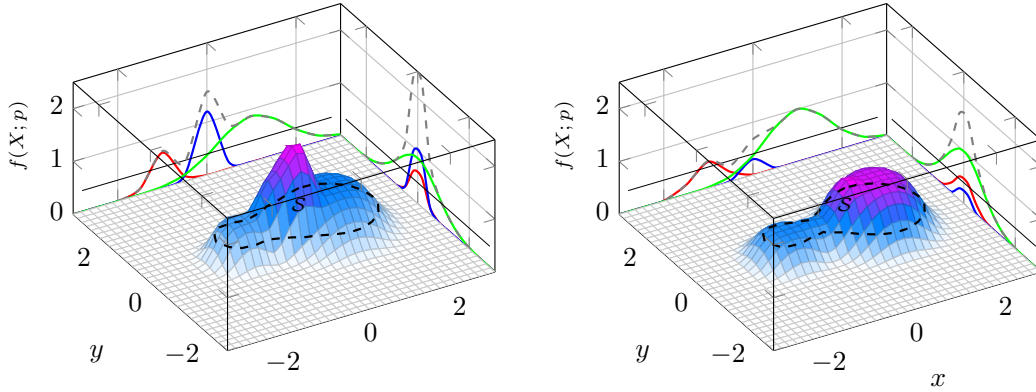


Figure 4.5: Two scalar field functions representing the same surface at $T = 0.4$ and generated with 3 Gaussian kernels: (left) $\alpha = \{0.6715, 1.2006, 0.8681\}$ and $\rho = \{0.3722, 0.3258, 1.0863\}$ takes different values which exhibit high variations in the scalar field, while on the contrary, (right) the scalar field height is smoothed – i.e. even variations – by setting $\alpha = \rho = \{0.3722, 0.3258, 1.0863\}$.

A tangible example is the contour \mathcal{S} generated with $\alpha_j \neq \rho_j$ and $\alpha_j = \rho_j$ in Fig. 4.5. In contrast to the formulation with $\alpha_j = \rho_j$, the scalar field $\alpha_j \neq \rho_j$ is hilly which hinders the work during the parameter optimization process. A practical effect due to this redundancy dismissal is the algorithm stabilization during the fine tuning of parameters. Owing to the aforementioned reasons, we set $\alpha_j = \rho_j$ for the remainder of this work. Therefore, the implicit function f is written as:

$$f(X; p) = \sum_{j=1}^{N_b} \rho_j \phi \left(\frac{|X - C_j|^2}{\rho_j^2} \right) \quad (4.7)$$

Eq. 4.7 presents mathematically a nice property of **BMs**, namely, locality. Locality may be translated by the fact that any subset of blobs may still be a good local approximation of the **BM**. This feature of **BMs** is displayed in Fig. 4.6, where a **BM** – composed of 3 blobs – may represent a certain contour composed of two parts: a main shape on the right with a small bump on the left (Fig. 4.6 left); one can locally preserve the shape on the right – in the vicinity of the selected blobs – by selecting the rightmost blobs (Fig. 4.6 right).

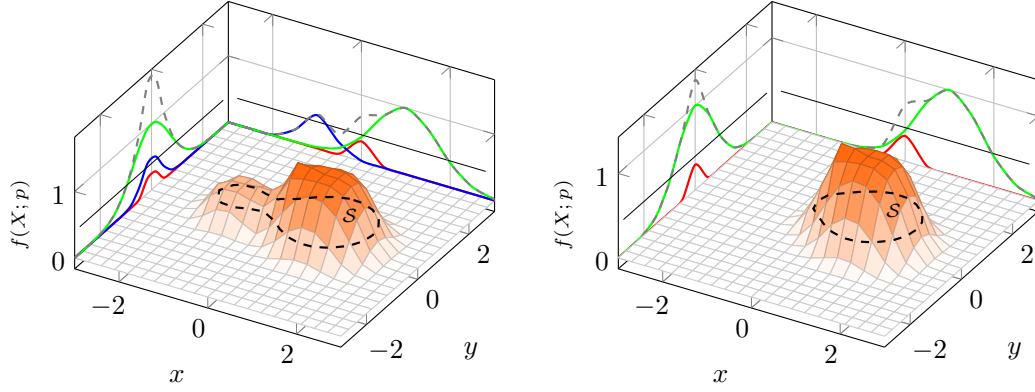


Figure 4.6: (left) A BM – based on Gaussian kernel – composed of three blobs: we set $\alpha_j = \rho_j$ and values were the same as in Fig. 4.2). (right) Local shape preservation is achieved when selecting two blobs within the BM (locality).

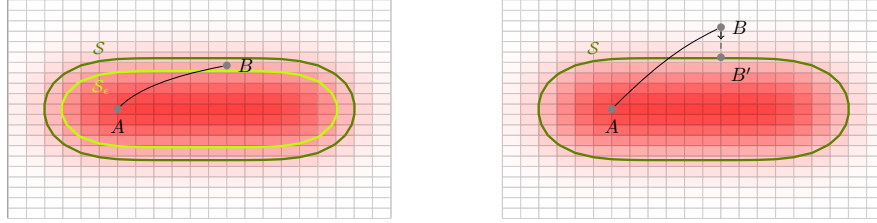


Figure 4.7: An interventional tool – composed of points A and B – interacting with the implicit surface S . (left) During simulation, two different T-values may be considered for predicting collisions: T and $T + \epsilon$. One defining the actual implicit surface S which is used as the constraint surface for computing contact forces. Another defining an implicit surface S_ϵ inside \mathcal{S} . No point A inside \mathcal{S} is in collision. A prediction zone is created between S and S_ϵ where any candidate point B is regarded as a candidate point for collision. (right) When a point B of the interventional tool is outside the surface, it is projected onto the surface – at location B' – along the implicit function gradient.

4.2.4 Distance function approximation

In the previous section, we studied the advantages of setting $\alpha_j = \rho_j$ and in the end, we proposed a new formulation of the implicit function and its gradient:

$$f(X; p) = \sum_{j=1}^{N_b} \rho_j \phi \left(\frac{|X - C_j|^2}{\rho_j^2} \right) \quad (4.8)$$

$$\nabla f(X; p) = 2 \sum_{j=1}^{N_b} \frac{(X - C_j)}{\rho_j} \cdot \phi' \left(\frac{|X - C_1|^2}{\rho_1^2} \right) \quad (4.9)$$

In this section, we analyze our implicit formulation from a practical standpoint in the context of collision detection and contact handling. In our particular simulation context, in order to help predict collisions, and have the function give a valid contact force direction, we would ideally like to have $f \equiv \mathcal{D}(\cdot, \mathcal{S})$ where $\mathcal{D}(\cdot, \mathcal{S})$ is the geometric distance function to the T-level set surface \mathcal{S} .

As illustration of collision detection and contact handling, consider the case of an interventional tool composed of points A and B (Fig. 4.7). Two different situations are possible: no collision (Fig. 4.7 left) or a collision (Fig. 4.7 right) has been detected.

First, points A and B are inside the vessel surface \mathcal{S} . For predicting collisions, one can define a prediction zone within which any point is regarded as a possible candidate for collision; for instance, point B is a candidate. The prediction zone can be seen as the zone encompassed by two surfaces, namely \mathcal{S} and \mathcal{S}_ϵ . Their desired distances to the vessel surface should be respectively $\mathcal{D}(\cdot; \mathcal{S}) = 0$ and $\mathcal{D}(\cdot; \mathcal{S}_\epsilon) = -\epsilon$.

Second, once point B is outside the vessel surface \mathcal{S} , following the gradient of \mathcal{D} , one can bring back this point to its closest peer B' on the vessel surface through a gradient descent strategy such as Newton-Raphson and accordingly, compute the required recall force to apply at the interventional tool. As a matter of fact, we can rely on f if it verifies $f = g(\mathcal{D})$; with g a monotonous function: both the point pre-selection for collision and the outside point reprojection process are unconstrained. Nevertheless, our model doesn't ensure such a relationship. For instance, consider a point-set describing a circle, and we look for a **BM** whose 0.1-level produces the sought representation. As depicted in Fig. 4.8 (left), a **BM** composed of blobs distributed around the center of the circle may be sufficient to obtain the sought \mathcal{S} . However, if we take a look at the scalar field of the resulting **BM**, one observes that the scalar field is no longer representative of the distance function inside \mathcal{S} (Fig. 4.8 right). Indeed, the scalar field presents a valley around the center circle which is not noticeable during the 0.1-level set computation. But the sought relation remains valid in a narrow band around the T-level set. Our aim is to enlarge this zone as much as possible for enhancing collision prediction and contact force computations.

Now, let's consider the case of a single blob. Against this context, the implicit function f is clearly monotonously related to the distance function and therefore, the implicit function gradient is aligned with the distance function gradient. Meanwhile, the prediction zone is vast, authorizing the extensive usage of an approximate geometric distance to the surface such as Taubin (1991) (further details are given in Sec. 4.4.1). Although in the case of one blob, our implicit formulation guarantees desired features for simulation, these features are not respected when considering more than one blob as aforementioned. In practice, we follow a strategy by letting a first **BM** captures roughly the point-set through placing blobs on the medial axis; and then capturing the details (Bittar et al. (1995)). In next chapters, we outline our paradigm for ensuring the required features for predicting collisions and handling collisions.

A final word on the choice $\rho_j = \alpha_j$, note that negative weights $\{\alpha_j\}$ allow implicit spheres to generate concavities in their neighborhood (Muraki (1991) and Bittar et al. (1995)). If we considered concavities in our formulation, assuredly more flexibility in the fitting process would be granted, but the distance function approximation would hardly or not be obtained with the proposed formulation. Fig. 4.9 illustrates the effect of a negative weight on the scalar field. A valley is formed in the scalar field leading to a carved surface. However, the gradient of the implicit function is no more to be trusted to indicate the direction of the recall force. Another side effect of allowing negative weights would be the extra computational burden due to redundancy, a larger range for $\{\alpha_j\}$

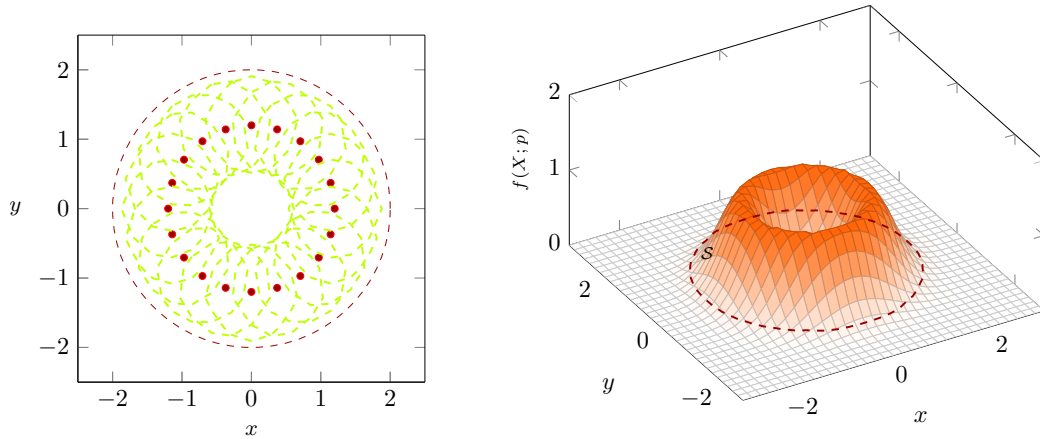


Figure 4.8: A BM estimating a circle of 2cm radius. (left) Blobs with $\alpha = \rho -$ and their corresponding contours at 0.1-level – composing the BM. (right) The associated scalar field.

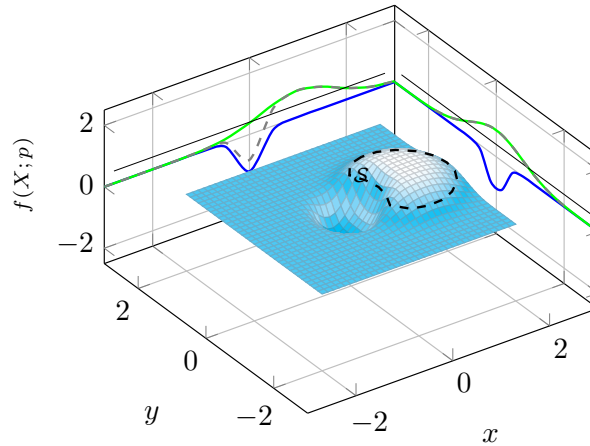


Figure 4.9: Scalar field function generated with one negative height value for the Gaussian kernels: $\alpha = \{-1.2006, 0.8681\}$ and $\rho = \{0.3258, 1.0863\}$. The negative height produces a valley in the scalar field. Consider the line $y = 0$, ∇f gives the right direction for the recall force if $x > 0$, no force at all for $x = 0$ and on opposite, repulsive force for $x < 0$.

would ineluctably increase this burden during optimization.

4.3 Energy formulation

Against an object reconstruction background, [Muraki \(1991\)](#) proposed a method driven by an energy minimization. The energy translated the fitting problem of BMs to range data. Later on, a similar methodology was utilized by [Tsingos et al. \(1995\)](#) and [Bittar et al. \(1995\)](#). Likewise, we opt to employ an energy-based formulation to fit a surface to N_p points $\mathcal{P} = \{P_i\}_{1 \leq i \leq N_p}$.

The idea of an energy-based framework is to translate the fitting problem as a sum of external and internal constraints. In other words, the global energy \mathcal{E} is a weighted sum

of constraints, for example, $\mathcal{E} = \alpha\mathcal{E}_1 + \beta\mathcal{E}_2 + \gamma\mathcal{E}_3$ where α , β and γ are positive hyper-parameters balancing the interaction of constraints \mathcal{E}_1 , \mathcal{E}_2 and \mathcal{E}_3 . For computing the BM that minimizes a certain \mathcal{E} , one needs to derive positions and widths of each blob which requires computing $\frac{\partial}{\partial C_j}\mathcal{E}$ and $\frac{\partial}{\partial \rho_j}\mathcal{E}$, and feed them to a gradient-based minimization algorithm, e.g. gradient descent.

Before diving into the energy minimization process, we make a detour by discussing about input data at our disposal.

4.3.1 Input data considerations

We assume that our input point-sets do not present outliers – or at least a small amount of outliers – and are unoriented. Nevertheless, orientation for point-sets may be obtained thanks to image gradient in 3DRA patient data. Input data may however present holes (missing data) and noise which makes the reconstruction task more difficult (Fig. 4.10). In practice, input data for LIM is provided by RBT.

1. **Noise.** Input points are corrupted by noise since no segmentation algorithm provides noise-free results. Accordingly, smoothing constraints must be utilized for minimizing the impact of noise on the quality of the reconstruction. One classical solution is to use the squared algebraic distance – i.e. the quadratic error minimization between the implicit surface and the input point-set – as proposed in Muraki (1991). Another classical approach for imposing smoothness consists in relying on curvature or area of the implicit surface (Lempitsky (2010) and Mullen et al. (2010)).
2. **Missing data.** Implicit functions are well-suited for interpolating missing data. Without a notion of inside/outside for the fitting process, blob placement may lead to ill-posed configurations and consequently, to wrong reconstructions (Muraki (1991)). In the case where data are missing, blobs can easily *slide into these holes*. Once a blob has slid, ambiguity is introduced in the fitting process. To alleviate this issue, a constraint can be defined to align the normal to the implicit surface with the normal provided at input points (Muraki (1991)). Literature puts forth another way for addressing this problem, namely, guarantee a cohesion between blobs. This latter solution may be qualified as an internal constraint while the former is an external constraint which requires that input points are provided with normals (Kazhdan et al. (2006)). However, normal-based algorithms rely on a consistent orientation of the normals to perform correctly. Unless reliable normals are provided, finding such an orientation has been recognized to be an ill-posed problem when the sampling is sparse and noisy (Alliez et al. (2007)) which is definitively our case (Fig. 4.10). In our case, one can estimate the normal orientation at input points by considering the 3DRA image gradient at input points.

We review here five classical energy terms used in the literature and introduce two novel terms in order to alleviate the aforementioned issues with input data. Our review is based on the evaluation of their performance on the following test scenario.

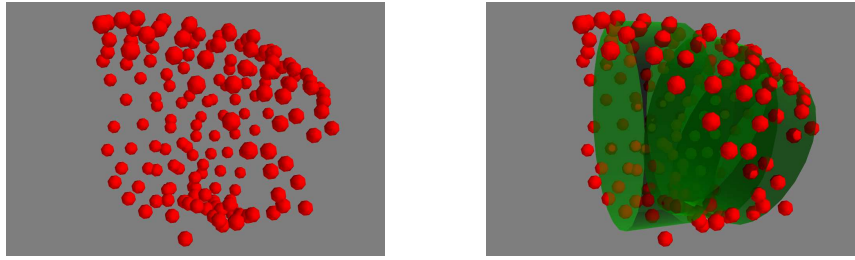


Figure 4.10: (left) Point-set extracted at the surface of a small and tortuous vessel with RBT. (right) For a better depiction, three RBT cylinders and their corresponding point-sets are displayed. Owing to the high curvature of the vessel, the point-set exhibits unbalanced density of points (top vs bottom points). Besides, point-set displays holes at its extremities.

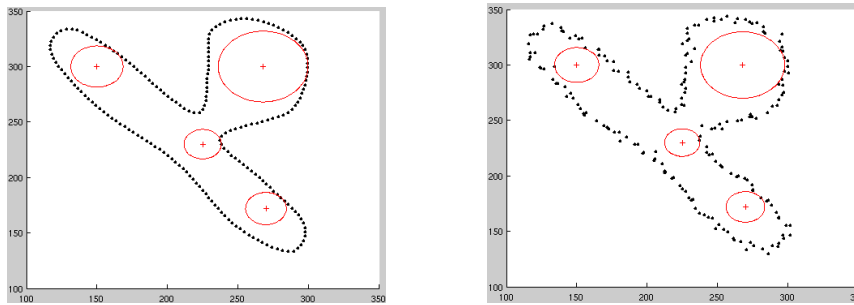


Figure 4.11: (left) Input point-set of a synthetic aneurysm to be reconstructed from a BM composed of 4 blobs in red. (right) The noise-free point-set is displaced along the normal direction with a normal distribution (noise corruption).

4.3.2 Test scenarios

Henceforth, we describe the synthetic data at our disposal, initialization conditions and the optimization framework used for assessing the impact of several energy terms in the reconstruction outcome. In order to facilitate the study of energy constraints, we mimicked our input data as 2D input point-set. Thereafter, we detail the different shapes and the fitting process used for producing the associated implicit contours in Fig. 4.13, Fig. 4.14, Fig. 4.15 and Fig. 4.16.

Data

Two different shapes were considered: an aneurysm (closed contour) and a vessel (open contour at extremities). Furthermore, we analyzed the 2D resulting point-set – for each shape – in a noise-free and noisy configuration.

Aneurysm. This synthetic case was produced for assessing reconstructions from closed point-sets with concavities. We first used a 2D manually created aneurysm point-set which was composed of 200 points (Fig. 4.11 left). Normals were computed using finite differences. Second, we considered the same reconstruction case but this time, from noisy input data (Fig. 4.11 right). Points were displaced along their normal direction

using a normal distribution (null mean and 2 pixels for the standard deviation) and the noise-free normal for each point was used when required.

Vessel. Besides, we intended to show the strength of each energy term when dealing with holes or open point-sets which reflected our input data (Fig. 4.10). To this end, we employed an open noisy point-set of a vessel contour – composed of 200 points and 30 pixels wide – to which we added 6 extra points outside the vessel walls so that outliers were mimicked (Fig. 4.12 left). Following the same guideline as for previous reconstruction cases, Gaussian noise (null mean and 0.5 pixels for the standard deviation) was added to the point-set. The normal was computed from the noise-free point-set using finite differences. A second test point-set was built, where the same 6 extra points were placed inside the corrupted point-set vessel (Fig. 4.12 right).

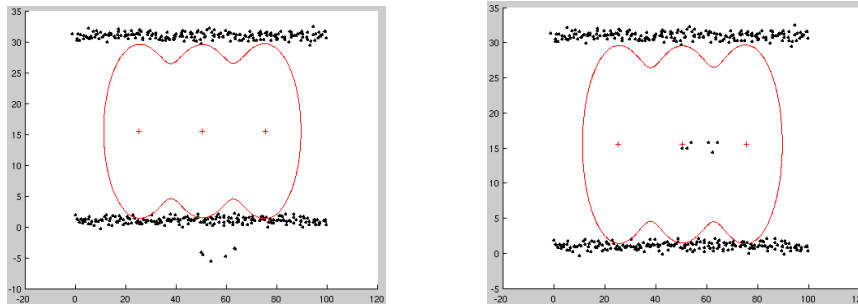


Figure 4.12: Open noisy point-set which represents a vessel contour. (left) 6 points were placed outside the vessel so that outliers are mimicked. Outliers normal was randomly affected. (right) This time, 6 points were placed inside the vessel and their normal was randomly assigned.

Initialization

Depending on the shape of the aimed point-set, the **BM** was composed of 4 blobs – for the aneurysm point-set (Fig. 4.11) – and of 3 blobs (Fig. 4.12) – for the noisy vessel point-sets – placed inside the point-set.

Optimization framework

The optimization framework alternates between **BM** fitting and blob fission. First, the initial **BM** is fine tuned over all centers and widths. Then, the point farthest from the surface, P_i^* , is computed. Next, the blob, contributing the most to the square function value $f(P_i^*)^2$ at this point, is split. Further details about P_i^* selection and the subdivision of its associated blob, are respectively given in Section 4.4.1 and 4.4.2. Up to here, the **BM** refinement strategy described herein is similar to our proposal in Section 4.4. Since the number of blobs may dramatically increase due to fissioning, we opt for reducing the number of blobs to be fine tuned at each iteration. With this in mind, the set of blobs \mathcal{M}_i^* , responsible for 90% of $f(P_i^*)^2$, is afterward retrieved. Then, thanks to the locality property of **BMs**, \mathcal{M}_i^* is fine tuned with a representative neighborhood around P_i^* . This latter point-set gathers the points P_i whose \mathcal{M}_i presents at least one of the

blobs in \mathcal{M}_i^* . Finally, the subdivision process is stopped when no significant drop on the energy value is attained as proposed in [Tsingos et al. \(1995\)](#). The same optimization process was used for all test scenarios. Furthermore, normals were provided with data and used only when required (not all energy terms need this information).

Evaluation criteria

The **Implicit Contour (IC)** was discretized in N_q points $\mathcal{Q} = \{Q_k\}_{1 \leq k \leq N_q}$ and compared to the N_p points of \mathcal{P} . To this end, we used the following metrics – expressed in pixels – based on the distance from a point A to a discrete contour (point-set) $\mathcal{B} = \{B_k\}$:

$$d(A, \mathcal{B}) = \min_k \|A - B_k\| \quad (4.10)$$

The first one, the **Hausdorff Distance (HD)** was defined as:

$$HD(\mathcal{P}, \mathcal{Q}) = \max\left\{ \max_{1 \leq i \leq N_p} d(P_i, \mathcal{Q}), \max_{1 \leq k \leq N_q} d(Q_k, \mathcal{P}) \right\} \quad (4.11)$$

which is a classical metric for comparing two point-sets. But the **HD** is sensitive to gross localized errors. Subsequently, we completed **HD** with the **Average Symmetric Surface Distance (ASSD)** ([Schaap et al. \(2009\)](#)):

$$ASSD(\mathcal{P}, \mathcal{Q}) = \frac{1}{N_p + N_q} \left(\sum_{i=1}^{N_p} d(P_i, \mathcal{Q}) + \sum_{k=1}^{N_q} d(Q_k, \mathcal{P}) \right) \quad (4.12)$$

which is more representative of what is visually perceived. Note that this measure was introduced in Section 3.9.7 to evaluate the precision of the centerline location in **RBT**. Furthermore, these metrics were computed with the noise-free point-set \mathcal{P} and the resulting **IC** \mathcal{Q} , even when considering noisy configurations. For all configurations, 200 points equi-distributed at the **IC** were used for discretization.

4.3.3 Framework

Remind that our input data are composed of noisy points and may present missing data (not necessarily at extremities) at the vessel surface. These two issues – missing and noisy input data – have been recurrently addressed in the literature. Therefore, we review classical energy terms which may fit to our purpose. First, we revisit two energy terms proposed in [Muraki \(1991\)](#). Then, we introduce two other energy terms from the literature since the former two constraints were insufficient to produce sound outcomes in our case.

Data attachment

The squared algebraic distance to the surface, \mathcal{E}_d , translates the raw problem of fitting a surface to scattered data. It is defined as:

$$\mathcal{E}_d = \frac{1}{N_p} \sum_i (T - f(P_i; p))^2 \quad (4.13)$$

Normal alignment

The normal alignment of the implicit surface with the input point-set normals is expressed through:

$$\mathcal{E}_n = \frac{1}{N_p} \sum_i \left| n_i - \left(-\frac{\nabla f(P_i; p)}{|\nabla f(P_i; p)|} \right) \right|^2 \quad (4.14)$$

where $\nabla f(P_i; p)$ is the implicit function gradient evaluated at P_i . It translates the compliance of the unit normal to the implicit surface with the unit normal n at the input points. It forces the normalized gradient of the implicit surface at P_i to coincide with the normal vector provided at this point.

Cohesion between blobs

To this point, no internal energy was introduced. In the presence of missing data, it turns out that one can try to enforce a certain cohesion between blobs to impede them from falling apart and thus generating extra contours/surfaces. In this area, physics bring about sound models for cohesion, namely the Van der Waals force. In our case, blobs can be seen as particles located at $\{C_j\}$ and with radii $\{\rho_j\}$. Hereafter, we define the Van der Waals energy as:

$$\mathcal{E}_c = \frac{1}{N_b(N_b - 1)} \sum_{j \neq k} \left(\frac{\rho_j}{|C_j - C_k|} \right)^2 \quad (4.15)$$

This term imposes an attractive force between neighboring blobs. Note that \mathcal{E}_c is an internal energy which relies only on the **BM** parameters.

Surface smoothing

A side issue of fitting a surface with implicit spheres is the so-called *blobby effect*; that is the wavy contour/surface bred when a small number of blobs are employed, for instance, the parent vessel of Fig. 4.14 (left) depicts this issue. Since the interventional tool often glides along the vessel wall, a *blobby* contour/surface may introduce jerky motion which hinders the simulation realism. To alleviate this issue, one can introduce internal constraints acting directly on the T-level set.

Four smoothness constraints – such as the classical squared Laplacian minimization – were explored in [Lempitsky \(2010\)](#). Among these internal energies, an efficient term based on the implicit surface area was revisited which improves upon classical regularization constraints:

$$\mathcal{E}_a = \frac{1}{N_p} \sum_i \left(\frac{\partial^2}{\partial x^2} f(P_i; p) \right)^2 + \left(\frac{\partial^2}{\partial y^2} f(P_i; p) \right)^2 + \left(\frac{\partial^2}{\partial z^2} f(P_i; p) \right)^2 \quad (4.16)$$

It smoothes the surface according to the minimal area criterion. Note, however, that this term cannot be considered as purely internal with regard to the implicit surface since it is evaluated at input points.

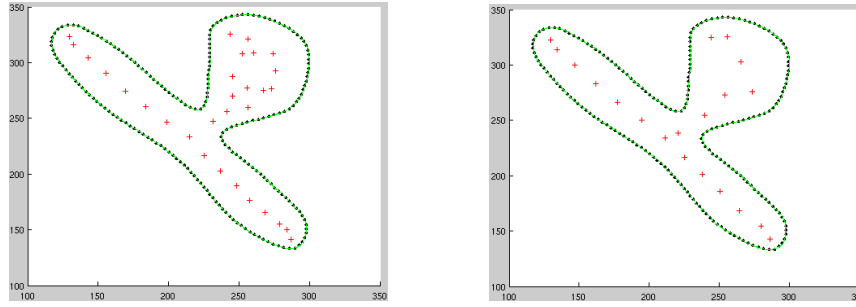


Figure 4.13: Point-set of a synthetic aneurysm to be reconstructed from a **BM** composed of 4 blobs. Reconstruction fulfilled through successive subdivision of one blob using the \mathcal{E}_d (left) and $\mathcal{E}_d + 10^{-4}\mathcal{E}_n$ (right). The resulting **IC** is supplied by the green curve while the centers of the blobs composing the final **BM** are located by the red crosses.

4.3.4 Configurations

We distinguished two reconstruction situations: reconstructions from closed (aneurysm) and open (vessel) contours. Both configurations presented a global energy of the form $\mathcal{E} = \mathcal{E}_d + \alpha\mathcal{E}_n + \beta\mathcal{E}_c + \gamma\mathcal{E}_a$. For each configuration, energy term weights were tuned so as to produce the best outcomes. Moreover, weights were chosen so that energy values were of the same magnitude. Several reconstruction trials with different weights were performed; the best result was chosen according to its visual correctness, its **HD** and **ASSD** scores. Besides, the optimization process was stopped when the drop on the energy value was respectively below 10^{-3} and $5 \cdot 10^{-3}$ for the closed and open contour.

Closed contour

A first reconstruction was performed on the noise-free point-set of Fig. 4.13 with two energy configurations. The left side exhibits the reconstruction using only \mathcal{E}_d meanwhile, the right figure shows the outcome for $\mathcal{E} = \mathcal{E}_d + \alpha\mathcal{E}_n$ ($\alpha = 10^{-4}$).

A second reconstruction was executed on the aneurysm noisy point-set as depicted in Fig. 4.14. Once again, former configurations – \mathcal{E}_d and $\mathcal{E}_d + \alpha\mathcal{E}_n$ – were run in addition to $\mathcal{E}_d + \alpha\mathcal{E}_n + \gamma\mathcal{E}_a$ combination. In this situation, α and γ were respectively set to 10^{-3} and 10^{-5} . Note that α is increased in the noisy configuration so as to enforce regularization on the normals.

Table 4.2 sums up both configurations **w.r.t HD** and **ASSD**, as well as the number of blobs required for producing the final **IC** in Fig. 4.13 and Fig. 4.14.

Open contour

By the same token, we used two cases for validation. Nevertheless, only reconstruction from noisy point-sets were regarded. For both reconstruction cases, the energy weights were $\alpha = 10^{-3}$, $\beta = 10^{-7}$ and $\gamma = 10^{-8}$. A first case involved the open vessel with 6 outliers outside the shape (Fig. 4.15). In this situation, we evaluated three energy configurations: $\mathcal{E}_d + \alpha\mathcal{E}_n$, $\mathcal{E}_d + \alpha\mathcal{E}_n + \beta\mathcal{E}_c$ and $\mathcal{E}_d + \alpha\mathcal{E}_n + \beta\mathcal{E}_c + \gamma\mathcal{E}_a$.

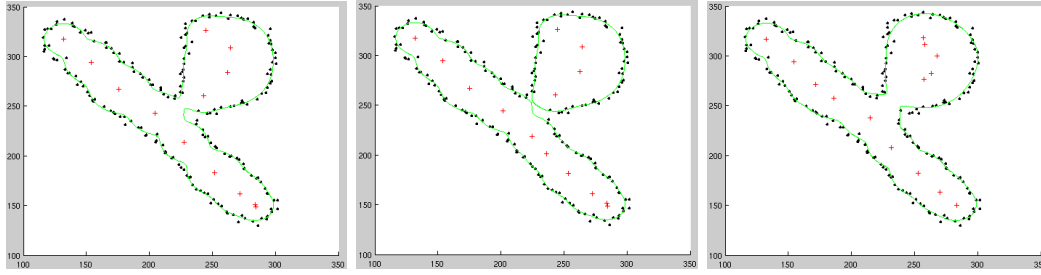


Figure 4.14: Same case as those presented in Fig. 4.13 but point-set is corrupted by noise. Reconstruction is fulfilled through successive subdivision of one blob using the \mathcal{E}_d (left), $\mathcal{E}_d + 10^{-3}\mathcal{E}_n$ (middle) and $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 5 \cdot 10^{-3}\mathcal{E}_a$ (right). The resulting IC is given by the green curve while the centers of the blobs composing the final BM are located by the red crosses.

Configuration	HD	ASSD	# Blobs
Fig. 4.13 left	2.00	0.91	30
Fig. 4.13 right	2.08	0.94	20
Fig. 4.14 middle	224.32	2.07	14
Fig. 4.14 right	8.89	1.76	14

Table 4.2: Two distances measures **Hausdorff Distance (HD)** and **Average Symmetric Surface Distance (ASSD)** – measured in pixels – and the resulting BMs compacity – i.e. the number of blobs (# Blobs) – for the aneurysm configurations. The first two rows belong to the noise-free point-set while the last two rows pertain to the noisy configuration.

In the same way, the second reconstruction case referred to the open vessel but equipped with 6 outliers inside the contour as illustrated in Fig. 4.16. With this intention, only two energy configurations were appraised since we aiming at showing the action of \mathcal{E}_c and \mathcal{E}_a on the IC. In other words, we compared the outcomes for $\mathcal{E}_d + \alpha\mathcal{E}_n + \beta\mathcal{E}_c$ and $\mathcal{E}_d + \alpha\mathcal{E}_n + \beta\mathcal{E}_c + \gamma\mathcal{E}_a$.

Together with Table 4.2 for the closed contour experiments, Table 4.3 encompasses figures for HD, ASSD and the number of blobs composing the final BMs in Fig. 4.15 Fig. 4.16.

4.3.5 Preliminary discussion

Fig. 4.13 (left) displays the final IC produced with \mathcal{E}_d when considering the noise-free aneurysm point-set. This energy constraint – as expected – works well in a noise-free context demonstrating the sub-pixel accuracy obtained, i.e. 0.91 pixels for ASSD as presented in Table 4.2. The same configuration in a noisy background performed well but it may be sensitive to variations in the point-set density as depicted in Fig. 4.14 around the neck of the aneurysm. Moreover, \mathcal{E}_d was coupled to \mathcal{E}_n – in Fig. 4.13 (right) – leading to a more compact BM (smaller number of blobs) due to the alignment induced on the implicit contour w.r.t to input point-set orientation. Table 4.2 evinces this compacity for a similar outcome with regard to HD and ASSD.

In the case of the aneurysm noisy point-set, the $\mathcal{E}_d + \alpha\mathcal{E}_n$ configuration introduced a visible effect, namely discontinuity around the neck of the aneurysm which was translated by a large value of HD in contrast to a lesser score of ASSD (Table 4.2). This discon-

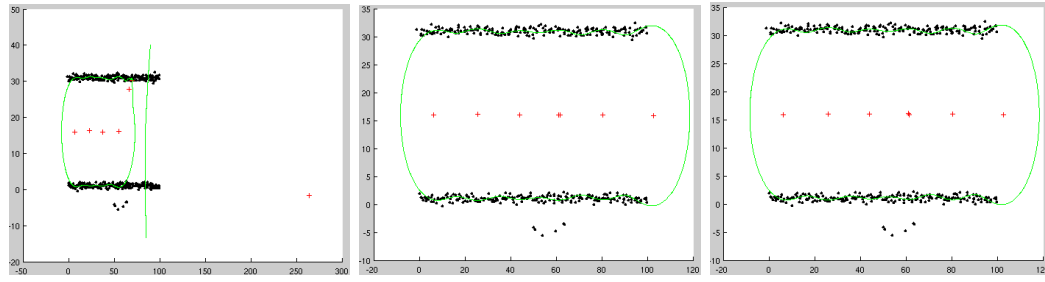


Figure 4.15: Open noisy point-set which represents a vessel contour. 6 points were placed outside the vessel so that outliers are mimicked. Outliers orientation was randomly affected. Reconstruction fulfilled through successive subdivision of one blob using the $\mathcal{E}_d + 10^{-3}\mathcal{E}_n$ (left), $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c$ (middle) and $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c + 10^{-8}\mathcal{E}_a$ (right). The resulting IC is giving by the green curve while the centers of the blobs composing the final BM are located by the red crosses.

tinuity might be the by-product of the fact that normals were no more consistent with the input points which led the minimization algorithm to wrong outcomes. Although one can evoke a problem of weighting in a noisy reconstruction context, configuration $\mathcal{E}_d + 10^{-3}\mathcal{E}_n$ gave birth to another problem when considering missing data and a small amount of outliers – as illustrated in Fig. 4.15 (left) – that is extra iso-contours. More precisely, the contour was split in two since outlier points could not be reached from inside the vessel walls. As a matter of fact, the minimization algorithm tried to take into consideration outliers and their orientation thus leading to expelling blobs and creating an extra surface (leakage). A final consideration, \mathcal{E}_d and \mathcal{E}_n were strongly biased in practice by even a small amount of outliers (Fig. 4.15). As a result, we needed to introduce regularization terms.

Against this background, our expectations were confirmed when coupling \mathcal{E}_c to $\mathcal{E}_d + \alpha\mathcal{E}_n$ since blobs stayed inside the shape (Fig. 4.15 middle). On the contrary, when outliers were inside the vessel walls with the same configuration (Fig. 4.16 left), two ICs were bred, thus leading to a high value for HD (Table 4.3). Indeed, the IC splitting was provoked by two facts, namely input data weighting in the data attachment term and normal vectors inconsistency. During the optimization process, points force equally the IC to pass through them and align the implicit function gradient with their normals. Normals inform of the IC interior/exterior which, in the presence of outliers inside the shape, induces the optimization algorithm to cleave the IC for minimizing both constraints. Note that alike situations respect the cohesion constraint, thus suppressing its effect.

To alleviate this issue in Fig. 4.16 (left), \mathcal{E}_a comes into play. The difference in the outcome was flagrant at first sight (Fig. 4.16 right) and second, HD and ASSD put forth lower scores in favor of \mathcal{E}_a configuration (Table 4.3). \mathcal{E}_a prevented the creation of a second IC which demanded higher energy. Besides, \mathcal{E}_a was applied to the aneurysm noisy point-set and coupled to \mathcal{E}_d and \mathcal{E}_n (Fig. 4.14 right). The benefit of this energy on the resulting IC was a higher smoothness and more resilience to noise. These characteristics were highlighted around the aneurysm neck and on the bulge where the IC was much rounder when compared to other configurations. Last but not least, when dealing with an open contour exhibiting outliers placed outside the vessel, similar outcomes were

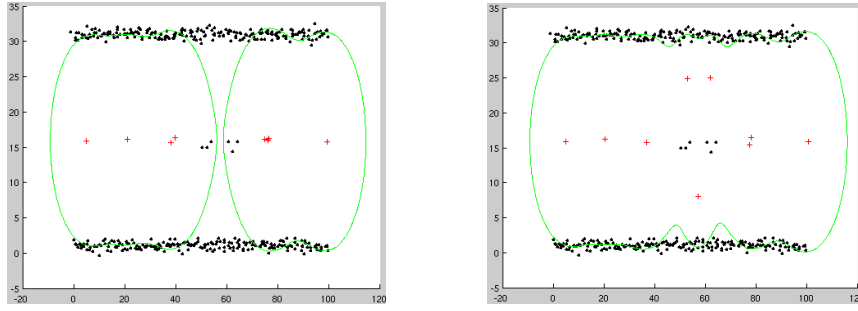


Figure 4.16: (left) Open noisy point-set which represents a vessel contour. 6 points were placed outside the vessel so that outliers are mimicked. Outliers orientation was randomly affected. Reconstruction fulfilled through successive subdivision of one blob using the $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c$ (middle), and $\mathcal{E}_d + 10^{-3}\mathcal{E}_n + 10^{-7}\mathcal{E}_c + 10^{-8}\mathcal{E}_a$ (right). The resulting IC is giving by the green curve while the centers of the blobs composing the final BM are located by the red crosses.

Configuration	HD	ASSD	# Blobs
Fig. 4.15 middle	23.93	5.15	7
Fig. 4.15 right	23.93	5.13	20
Fig. 4.16 left	1004.00	6.18	8
Fig. 4.16 right	22.05	4.69	9

Table 4.3: Two distances measures **Hausdorff Distance (HD)** and **Average Symmetric Surface Distance (ASSD)** – measured in pixels – and the resulting BMs compacity – i.e. the number of blobs (# Blobs) – for the vessel. The first two rows belong to the configuration with 6 outliers outside the point-set and the last two rows pertain to that exhibiting 6 outliers inside the point-set.

observed – as those bred by configuration $\mathcal{E}_d + \alpha\mathcal{E}_n + \beta\mathcal{E}_c$ – when this configuration was combined with \mathcal{E}_a (Fig. 4.15 right). Table 4.3 evinces quantitatively similar measures for **HD** and **ASSD** for both configurations.

As a result, four energy terms may be needed in our application when dealing with noisy point-sets and missing data, that is to say $\mathcal{E} = \mathcal{E}_d + \alpha\mathcal{E}_n + \beta\mathcal{E}_c + \gamma\mathcal{E}_a$. However, the number of constraints may be further reduced. Indeed, the constraint \mathcal{E}_n was not reliable when point-set normals were inconsistent and in particular, didn't influence the result (either positively or negatively) when normals were computed from the **3DRA** image gradients. Consequently, we eliminated this constraint without altering the efficiency of the other constraints, this is to say a global energy : $\mathcal{E} = \mathcal{E}_d + \alpha\mathcal{E}_c + \beta\mathcal{E}_a$.

A final word on \mathcal{E}_d . It draws the implicit function via the algebraic distance whereas the implicit surface and the point-set may present a low algebraic error; at some points where a gross geometric error exists, especially at locations where the implicit function gradient varies little. Instead, one can imagine employing a geometric criterion that approximates the geometric distance of the input point-set to the surface. By this mean, a much accurate fitting of the implicit function may be achieved. Against this background, **Taubin (1991)** provides the approximate distance from a point X to the T-level set \mathcal{S} :

$$\text{dist}(X, \mathcal{S})^2 \approx \frac{(T - f(X, p))^2}{|f(X, p)|^2} \quad (4.17)$$

With this in mind, we tried a novel energy term based on this approximate geometric

distance:

$$\mathcal{E}_g = \frac{1}{N_p} \sum_i \text{dist}(P_i, \mathcal{S})^2 \quad (4.18)$$

which translates the geometric distance relation between data points and the T-level set \mathcal{S} . Nonetheless, it uprose that \mathcal{E}_g becomes unstable when large distances between points and the implicit function were considered. In other words, it happened when the gradient of the implicit function became small, leading to situations where blobs were placed outside the targeted object due to bad estimation of the geometric distance. Indeed, Taubin's approximation formula is a priori only valid within a narrow band of the T-level set of the implicit.

4.3.6 Our energy-based framework

The current energy formulation seems to tackle the expected challenges with input data. Yet, one can address several critiques regarding energy terms. First, the internal energy for cohesion \mathcal{E}_c tended to agglomerate blobs (Fig. 4.15 middle and Fig. 4.16 left). Accordingly, it is also compulsory to introduce a repulsive force to counterbalance the attractive force so that compacity is ameliorated. In this context, [Szeliski et al. \(1993\)](#) proposed a potential term presenting both attractive, at long range, and repulsive, at close range, forces in a particle system modeling framework. This is the Lennard-Jones potential function which has the following expression when considering particles of equal sizes:

$$\Phi_{LJ}(r) = \left[\frac{\sigma}{r} \right]^n - \lambda \left[\frac{\sigma}{r} \right]^m \quad (4.19)$$

λ is a constant; r is the distance between particle centers C_j and C_k ; σ is the finite distance at which the inter-particle potential is zero ($\lambda = 1$). Furthermore, n and m are physical constants; e.g. in physics, n and m are respectively equal to 12 and 6 with $\lambda = 2$. The former term stands for the repulsive action whereas the latter stands for the attractive term. Within a physical context, Φ_{LJ} models the attractive Van der Waals force and the repulsive Pauli force. However, the proposed energy only considered particles of similar sizes.

In order to take into account blobs of different sizes, the energy term was modified to suit our purpose as follows:

$$\mathcal{E}_{LJ} = \frac{1}{N_b(N_b - 1)} \sum_{j \neq k} \left(\frac{s\sqrt{\rho_j \rho_k}}{|C_j - C_k|} \right)^{12} - 2 \left(\frac{s\sqrt{\rho_j \rho_k}}{|C_j - C_k|} \right)^6 \quad (4.20)$$

In our case, σ should be representative of the pair of blobs influence zone, that is to say their own ρ_j . However, the question arises as whose width to pick up. Furthermore, this distance should vary according to both widths since the inter-particle action is symmetric. For preserving it, we use the geometric mean between both blobs width. Consequently, we set σ to be s times the geometric mean, hence each term is minimal (with value -1) for $|C_j - C_k| = s\sqrt{\rho_j \rho_k}$, being repulsive for blobs closer than this distance, and attractive for blobs further away. It imposes some cohesion between neighboring blobs to avoid leakage – or expelling blobs – where data points are missing, while preventing blobs from accumulating within the model.

Second, the expression of \mathcal{E}_a corresponds to the sum of squared diagonal elements of the Hessian matrix of f . Put another way, \mathcal{E}_a is the minimization of the curvature along privileged directions. Instead of restraining the curvature minimization process to some directions, one can extend this to the implicit function curvature. For triangular meshes, mean curvature estimation has brought a vast number of methods (Surazhsky et al. (2003)). In contrast, a closed form computation at any point in space from an implicit formulation exists and is given as follows (Goldman (2005)):

$$\kappa(P) = \frac{\nabla f^t H_f \nabla f - |\nabla f|^2 \text{trace}(H_f)}{2|\nabla f|^3} \quad (4.21)$$

where H_f is the Hessian matrix of f , computed at point P . Instead of \mathcal{E}_a , we propose to minimize the square mean curvature $\kappa(P)$ as follows:

$$\mathcal{E}_\kappa = \frac{1}{N_p} \sum_i \kappa(P_i)^2 \quad (4.22)$$

This curvature dependent energy is a classical energy term used in active contours frameworks (Kass et al. (1988) and Angelini et al. (2005)) where κ is mostly provided by approximation formulae (Williams and Shah (1992)). In this work, we propose to use the closed-form which is well-suited for implicit surfaces. \mathcal{E}_κ reduces the wavy effect that could stem from modeling a tubular shape with implicit spheres.

In conclusion, we propose to combine these 3 energy terms: $\mathcal{E} = \mathcal{E}_d + \alpha \mathcal{E}_{LJ} + \beta \mathcal{E}_\kappa$. Behind the rather classical form given above for the energy terms, it is important to notice that the whole energy is known under a closed-form expression. As a consequence, closed-form expressions were derived for its gradients w.r.t the BM parameters $\{\rho_j\}$ and $\{C_j\}$. Equations are given in Appendix B.

In the next section, we describe in details the BM optimization and refinement which correspond to steps in-between Fig. 4.1e and Fig. 4.1f.

4.4 Blobby model refinement

Muraki (1991) presented an automatic method for generating an implicit shape description from range data, i.e. scattered points supplied with normal vectors. The model was iteratively refined by subdividing some blobs into two new ones, and optimizing their parameters according to the energy function. However, with no clue about which blob to select for subdivision, Muraki (1991) made exhaustive trials composed of: select a blob, split it in two new blobs and fine tune their parameters. At the end of this process, the best trial leading to the minimal energy was kept. No efficient heuristic was proposed to select a blob for fissioning, leading to a very expensive computational process. Moreover, since kernels of infinite support were employed, the reconstruction process was not localized. Adding a new blob to the BM modified the shape everywhere, even in areas that were already reconstructed well.

Tsingos et al. (1995) introduced a semi-automatic paradigm in this context. Likewise, blobs were progressively subdivided to refine the surface. The presented selection criterion was based on locality: measuring the contribution of each blob to the data attachment term (\mathcal{E}_d) in a user-defined window, and choosing the main contributor for

fissioning. Initialization was interactively performed by the user who had to define an initial set of blobs and a number of slightly overlapping windows. User input is not an option in our context where thousands of **BMs** are handled (see Section 4.8). Moreover, we experimentally noted that this technique was prone to favor small blobs, thus focusing on details, before dealing with areas roughly approximated by one large blob. This behavior is caused by this selection mechanism using the algebraic distance to the implicit surface. [Bittar et al. \(1995\)](#) addressed the initialization and refinement issue by proposing to select the position and width of blobs according to a pre-computed medial axis of the object to reconstruct. In other words, blobs were placed along the medial axis and initialized with information encoded in it, e.g. the width of the object at the targeted location. A new blob was placed on unused locations in the medial axis. The location in the medial axis was selected according to the measure of \mathcal{E}_d at data points located within an spherical window, i.e. the influence range of blobs only available with compact support kernels. In our case, this heuristic may be reduced to place blobs along the centerline of vessels. However, this procedure may fail or at least demand significant effort at capturing details on the vessel surface. A better representation of the blood vessel surface may intuitively be obtained by first placing blobs along the centerline and then, placing new ones for capturing details, e.g. in the vicinity of the desired area to be captured. To our knowledge, no heuristic for selection have been proposed based on a geometrical criterion.

4.4.1 Selection-Subdivision

To correct the poor performance of the selection mechanism based on the algebraic distance ([Tsingos et al. \(1995\)](#)), our criterion relies upon the geometric distance approximation proposed by [Taubin \(1991\)](#). The idea is to increase the number of blobs where the **BM** worst estimates the input points. In essence, details need a finer representation, thus an increase number of blobs. This paradigm relies upon three steps. First, we find the point in the input point-set which is farthest to the surface. Next, we find the blob that is for the most responsible for its estimation. Finally, this blob is replaced with two new blobs. Henceforth, we describe in details our selection-subdivision heuristic.

The point P_{i^*} farthest to the surface is such that:

$$i^* = \arg \max_{1 \leq i \leq N_p} \frac{|T - f(P_i; p)|}{|\nabla f(P_i; p)|} \quad (4.23)$$

Next, the blob $\#j^*$ where:

$$f_j(P_{i^*}) = \rho_j \phi \left(\frac{|P_{i^*} - C_j|^2}{\rho_j^2} \right) \quad (4.24)$$

whose iso-surface is the closest to P_{i^*} , is selected, according to Taubin's distance (see [Fig 4.17c](#) for depiction):

$$\#j^* = \arg \max_{1 \leq j \leq N_b} \frac{|T - f_j(P_{i^*})|}{|f_j(P_{i^*})|} \quad (4.25)$$

Note that these criteria – based on Taubin's distance – are valid in large areas because we set $\alpha_j = \rho_j$ in the definition of f whose expression provides a **BM** function

graph without bumps. The subdivision step then replaces this blob with two new ones (Fig 4.17d). Their width ρ'_{j^*} is chosen such that two blobs, centered on C_{j^*} , of width ρ'_{j^*} would have the same iso-surface as one blob centered on C_{j^*} , with width ρ_{j^*} (the formula depends on the kernel). That is to find r such that:

$$\rho_{j^*} \phi \left(\frac{r^2}{\rho_{j^*}^2} \right) = T \quad (4.26)$$

Then find ρ'_{j^*} so that:

$$2\rho'_{j^*} \phi \left(\frac{r^2}{\rho'_{j^*}{}^2} \right) = T \quad (4.27)$$

where r is the Euclidean distance from C_{j^*} to the iso-surface.

Incidentally, the cohesion energy \mathcal{E}_{LJ} may become large **w.r.t** other energy terms, when two blobs occupy the same location. Therefore, the optimization process tends to expel one of the blobs, thus suppressing the effect of the subdivision process. As a rule of thumb, the first new blob is centered on C_{j^*} , while the second is translated by $\rho_{j^*}/10$ towards P_{i^*} . This heuristic copes with singularities with \mathcal{E}_{LJ} and guides the process towards better fitting P_{i^*} .

4.4.2 Optimization

Such a gradual subdivision procedure may lead to a dramatic increase in the number of blobs, and hence the size of the optimization problem. The locality of the kernel ϕ allows us to focus the optimization onto the newly created pair of blobs (Fig 4.17e). More exactly, only the new blob, that is slightly misplaced, is optimized; the other blobs remain constant. By this way, we intend to reflect the impact of adding a new blob only in the newly created blob and not on the already fitted surface. The energy is minimized using Polak-Ribiere conjugate gradient (PR) algorithm, taking advantage of the closed-form expressions of both the energy and its gradients. Partial derivatives of energy **w.r.t** changes in width and positions are given in Appendix B. A single minimization loop consists in one PR minimization over the center (3 variables), followed by one on the width (1 variable). In practice, a maximum of 5 loops proved sufficient.

4.5 Implicit Modeling

In this section, we explain the overall implicit modeling algorithm (Alg. 4). To this end, we decompose the process in two phases: the overall implicit modeling and the **Local Implicit Modeling**.

4.5.1 Overall implicit modeling

The vascular tree segmentation result (**RBT**) is input to our algorithm as a tree of centerlines. Every centerline node $\{\mathcal{N}_l\}_{1 \leq l \leq L}$ is equipped with a cylinder $\mathcal{C}_l = (O_l, r_l, \vec{d}_l)$ (where O_l , r_l and \vec{d}_l are respectively the cylinder center, radius and direction) – that locally estimates the vessel radius and direction – and a sampling of the blood vessel surface \mathcal{P}_l^0 . The idea is to enrich each centerline node with a local implicit representation of

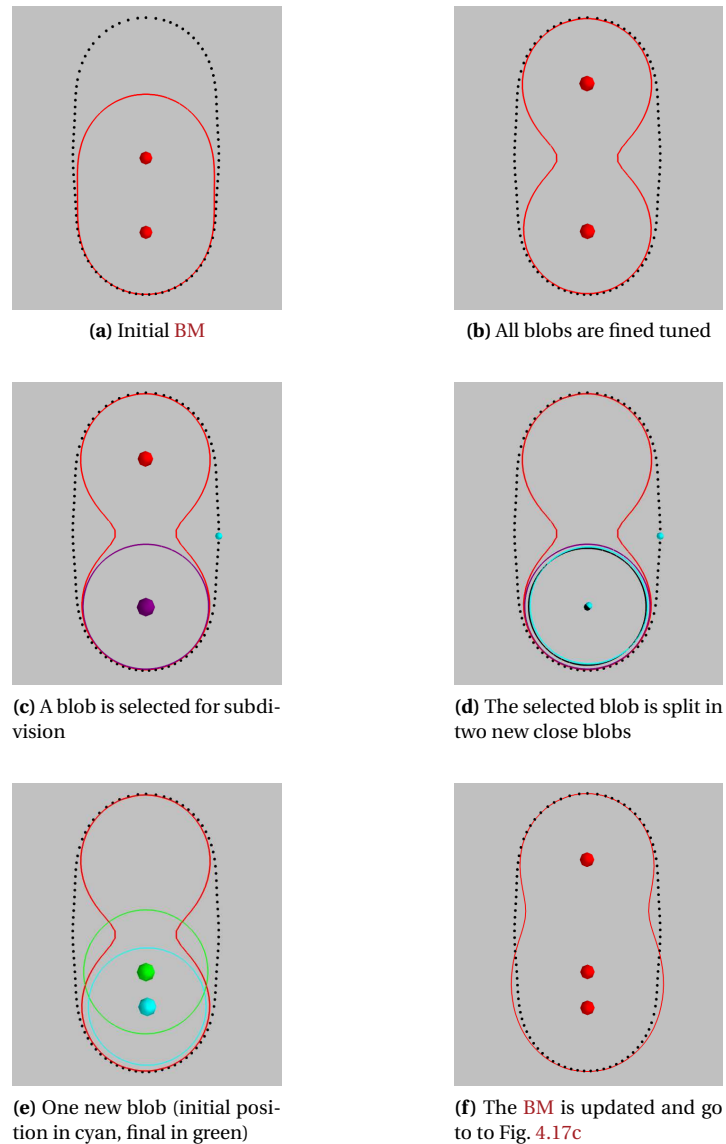


Figure 4.17: The elementary reconstruction step in 2D. (a) Points (black) are to be reconstructed with two blobs (red contour). (b) First, all blobs are finely tuned over the positions and then, over the widths. (c) Second, find the point farthest to the contour (cyan) and the blob closest to it (purple). (d) Third, replace this blob with two blobs. The first (black contour) is placed at the same position (black) and the second (cyan contour) is translated in the direction of the farthest point (cyan). Widths are calculated so that both blobs centered at the black point generate the same iso-surface as the fissioned blob (purple). (e) Fourth, the translated blob (cyan) is tuned over its center and width. (f) Finally, the BM is updated by integrating the recently tuned blob.

the local blood vessel surface. Next section addresses the production of these local implicit surfaces. In order to ensure smooth transitions between adjacent BMs (Fig. 4.1d), the local data point \mathcal{P}_l^0 sets are concatenated so as to overlap with their neighbors, e.g. neighboring nodes typically located at a topological distance $\mathcal{T} = 2$ are used. We thus equip each node \mathcal{N}_l with a point-set $\mathcal{P}_l = \bigcup \mathcal{P}_k$ so that $\mathcal{T}(\mathcal{N}_l, \mathcal{N}_k) \leq 2$.

The main steps are given in Algorithm 4. First, the algorithm places two blobs – which constitute the initial BM – at reconstruction node on the centerline. These two blobs are placed along the vessel direction \vec{d}_l so that an elongated shape is formed, and the radius ρ_l is scaled so it matches the vessel radius r_l (Fig. 4.1e and Algorithm 4.2). That is:

$$\rho_l \phi \left(\frac{r_l^2}{\rho_l^2} \right) = T \quad (4.28)$$

Second, an implicit surface which fits \mathcal{P}_l is produced through subdivision and fine tuning of the initial BM blobs. Once the whole vasculature is locally reconstructed, the final BMs are tied to their respective point on the centerline (Alg. 4.4). As a result, each node \mathcal{N}_l encompasses a cylinder (O_l, r_l, \vec{d}_l) , a point-set \mathcal{P}_l and a BM \mathcal{B}_l , thus describing the local blood vessel surface in its vicinity. In this algorithm, each local BM is treated independently of others, thus enabling parallel computation.

Algorithm 4 Overall implicit modeling

Require: $\{\mathcal{N}_l\}$, \mathcal{T} , N_s and t_g .

- 1: **for all** \mathcal{N}_l **do**
 - 2: **Initialization:** \mathcal{B}_l : place 2 blobs with diameter adapted to the vessel width, along the vessel direction. \mathcal{P}_l : concatenate local data points.
 - 3: **Fitting process:** $\mathcal{B}_l = \text{LIM}(\mathcal{B}_l, \mathcal{P}_l, N_s, t_g)$ (Alg. 5)
 - 4: **Centerline enrichment:** Associate \mathcal{B}_l to \mathcal{N}_l .
 - 5: **end for**
-

4.5.2 Local implicit modeling

In this section, we describe the mechanism for generating a final implicit surface which fits each one of the $\{\mathcal{P}_l\}$ point-sets – as defined in the previous section – from an initial BM (Algorithm. 4.2). This process involves selection (Alg. 5.3), subdivision (Alg. 5.4) and optimization (Alg. 5.5) stages which are repeated until a number of subdivisions N_s or threshold accuracy t_g is attained (Alg. 5.6) The following three steps are thus applied on the initial BM:

1. A first energy minimization is performed over the full BM (Fig 4.17a-4.17b), with a single minimization step, i.e. all centers (6 parameters) then all widths (2 parameters) of its two blobs (Alg. 5.2) (repeated 5 times).
2. The subdivision process is applied, as described in Section 4.4.1 (Fig. 4.17c-4.17f and Alg. 5.3-5.5). The process is stopped when a maximum number N_s of subdivisions is reached, or the distance between P_{i^*} – i.e. the point farthest from the surface – and blob $\#j^*$ drops below a threshold t_g (Alg. 5.6).

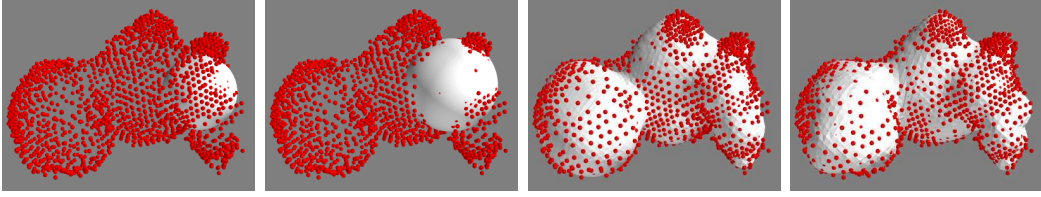


Figure 4.18: Implicit modeling of an aneurysm. The points $\{P_i\}$ are in red. (From left to right) Initialization with a single blob ; after the first minimization ; after 25 subdivisions ; final result (100 subdivisions)

3. The model is fine tuned by a single energy minimization step over the full **BM** ($3N_b$ parameters for the centers and then N_b parameters for the widths), repeated 5 times as in step 1 (Alg. 5.7).

Algorithm 5 Local Implicit Modeling

Require: Centerline node = Cylinder and point-set \mathcal{P} .

- 1: **function** LIM($\mathcal{B}, \mathcal{P}, N_s, t_g$)
 - 2: **Global energy minimization:** Single minimization over all centers and then over all widths of \mathcal{B} (5 loops).
 - 3: **Selection:** find the point farthest to the surface P_{i^*} and then the blob $\#j^*$ closest to it (Eq. 4.23).
 - 4: **Subdivision:** replace it with two new blobs (\mathcal{B} is updated). The first is centered at C_j and the second is displaced towards P_{i^*} .
 - 5: **Optimization:** The translated blob is to be optimized, the others remain constant, through a single energy minimization step (5 loops).
 - 6: **Stopping criteria:** Stop the process if the maximum number of subdivisions N_s is reached or the distance between P_{i^*} and blob $\#j^*$ is below a threshold accuracy t_g . Else repeat from step 3.
 - 7: **Final optimization:** Fine tune over all centers and then all widths of \mathcal{B} (as in step 2, 5 loops).
 - 8: **end function**
-

Redundant blobs are either ejected far away from the surface during the optimization over the centers, or their widths are reduced to almost zero during the minimization over the widths. A simple clean-up procedure is applied after each single or full minimization step: blobs relatively far from the node position (O_l), and blobs whose widths are below a certain distance are removed from the **BM**. In combination with the subdivision process, this clean-up enables the algorithm to simulate large blob displacements, which is hardly possible with the local minimization, thereby adding robustness to poor initialization: the clean-up procedure removes blobs meanwhile the split process adds blobs back at another place. Figure 4.18 illustrates the essential steps of this fitting algorithm and its capacity to model complex shapes, such as aneurysms, even from rough initialization (one blob at the aneurysm entrance).

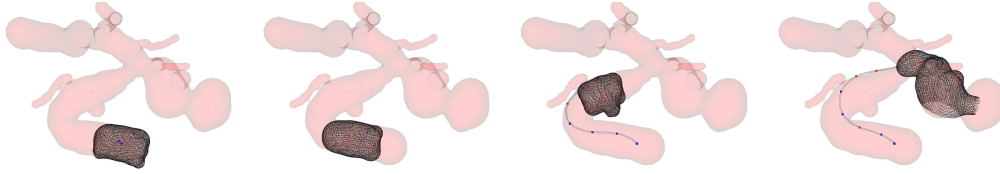


Figure 4.19: Selection of a local BM during simulation. The current local BM surface used to solve the constraints at the tool tip is displayed in wire-frame for 4 simulation steps. The overall vessel surface is shown in transparent red. Discrepancies with the BMs might occur, due to this surface being simplified in order to minimize the CPU load devoted to visualization.

4.6 Using the local implicit models for simulation

Most of the interventional tools are slender and their motion can be defined by that of longitudinal nodes. During a simulation step, contact with the vessel surface must be detected and solved for each point on the tool. Our first approach for solving collisions was to link each tool point to its closest point on the centerline and use the associated local BM as the surface constraint. In order to take into account the topology and avoid jumps into KVs, only the neighbors of the current centerline node are considered as candidates to update the surface constraint during the motion of the tool (see Fig. 4.19). However, this approach may fail at situations where tiny vessels stem from large vessels.

For sake of clarity, let's consider two nodes of a vascular tree as depicted in Fig. 4.20 (left). A tool point P sliding on the surface of the largest vessel becomes associated to node \mathcal{N}_A when facing the tiny vessel at time t . Then P becomes unable to reach node \mathcal{N}_B and continue its smooth motion. Indeed, point P during simulation is treated with regard to the implicit surface (\mathcal{S}_A) of \mathcal{N}_A . Since at $t + 1$, P lies outside \mathcal{S}_A as in Fig. 4.20, it is projected back onto it due to contact force computation. This projection is repeated indefinitely unless P crosses the median plane – i.e. the locus of points X defining the plane passing through point m :

$$d(X, O_A) = |X - O_A| = d(X, O_B) = |X - O_B| \quad (4.29)$$

A solution to this issue is to translate this frontier – along the segment $[O_A O_B]$ – proportionally to the two implicated nodes radii, r_A and r_B . It turns out to be the same as considering the power diagram – also known as (a.k.a) weighted Voronoi diagram – of the current associated node and the candidate node. In this particular case, the power diagram consists of two half-planes separated by a line where both circles have equal power:

$$d^*(X, C_A) = |X - O_A| - r_A^2 = d^*(X, O_B) = |X - O_B| - r_B^2 \quad (4.30)$$

The locus of points X respecting Eq. 4.30 are represented by the cyan line in Fig. 4.20 (right). In summary, the frontier position – for relaying a tool point from one node to another – is dependent of the vessel width.

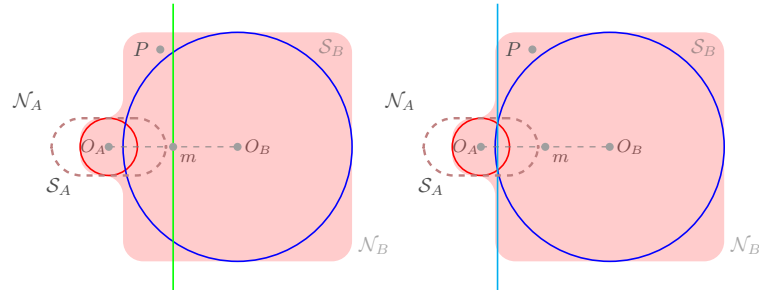


Figure 4.20: A tiny vessel stems from a portal vessel. In this situation, a tool point P using the local implicit surface \mathcal{S}_A (dashed contour) associated to node \mathcal{N}_A needs to attain a certain distance – i.e. a plane – to be associated to node \mathcal{N}_B in order to employ implicit surface \mathcal{S}_B (filled region). The plane position varies whether relying on the Euclidean distance – i.e. the Voronoi diagram – (left) or on a Power diagram (right) – i.e. the weighted Voronoi diagram.

The interest of the weighted Voronoi diagram is to keep the planar decomposition of the space which generates polyhedral cells. Indeed, these latter are helpful for visualization as described in the next section.

4.7 Visualization

We have to show visual results of our model. However, our model is adapted to simulation, but not to visualization. We translated the above **BM** selection procedure to give a visual impression of the result in the next section: each local iso-surface was first extracted with marching cubes, and was then cut by the radical planes separating the current centerline point from each of its neighbors, i.e. intersect the surface with the cell associated to the current node in the power diagram. No blending was performed between adjacent **BMs**. The final surface presents as a stack of individual surfaces.

4.8 Experimental validation

In this section, we evaluate the proposed geometrical model bred with **LIM**. To this end, we first present the patient data at our disposal and the synthetic data specially created for exhibiting the strength of our **LIM** algorithm. After this short introduction to our input data, we explain our proceedings for parametrizing our algorithm as well as technical choices taken for this evaluation. Afterwards, we describe the evaluation measures used for appraising robustness, modeling capability and efficiency of our proposal. Finally, we present the experiments carried out for this purpose.

4.8.1 Clinical data

Patient data was the same as that used to validate **RBT** in Chapter 3. The set of 10 patient data thereto was used for validation. Each patient data set consisted of a **3DRA** acquired on a vascular C-arm (Innova 4100, GE Healthcare) during the intra-arterial injection of the internal carotid artery. **3DRA** volumes presented as a 512^3 isotropic voxel cubes, between 0.18-0.22 mm voxel size.

Input data was provided by **RBT** algorithm which was first run on the carotid artery (the stem vessel of the vascular tree) and subsequently, all vessels were tracked. A total of 379 instances of **RBT** were needed and between 26-56 instances per patient were examined. Moreover, a total of 11 aneurysms were segmented with the algorithm described in Chapter 3.7. Table 4.4 reports in detail these values for each patient.

4.8.2 Synthetic data

We intended to provide cases close to those found in clinical data. For that, four **3D** shapes were considered which mimicked possible reconstruction instances: arc (A), bifurcation (B), capsule (C) and capsule opened at extremities (Co). In order to provide a smooth and continuous representation of these shapes, shapes were generated using convolution surfaces from point-based skeletons with Cauchy kernel (see Fig. 4.21 for a depiction). Point-based skeletons were created as follows:

- A.** The arc was created using the following formula which gave the skeleton point location:

$$C = C_0 + r * \begin{pmatrix} \cos(t_0 + t \cdot t_1) \\ \sin(t_0 + t \cdot t_1) \\ 0 \end{pmatrix} \quad (4.31)$$

where $r = 4$, $t_0 = 0$, $C_0 = \vec{0}$, $t_1 = \pi$ and $t \in [0, 1.0]$ (step 1/100). A total of 100 blobs of width $\rho = 0.2$ were necessary for generating the convolution surface in Fig. 4.21a.

- B.** The bifurcation was the resultant of two arcs summation: the first was computed with $r = 4$, $t_0 = -\pi/6$, $C_0 = \vec{0}$, $t_1 = 8\pi/9$, $t \in [0, 1.0]$ (step 1/100) and 100 blobs of width $\rho = 0.2$; and the second generated with $r = 4$, $t_0 = \pi/6$, $C_0 = [2, -4, 0]^T$, $t_1 = \pi/3$ and $t \in [0, 1.0]$ (step 1/100) and 100 blobs of width $\rho = 0.15$. Blending issues were limited as follows: the two arcs were separately polygonalized; their corresponding meshes were subsequently merged thanks to MeshLab¹; and finally, the widths of both skeletons were fine tuned so that the vertices of the resulting mesh were correctly approximated by the convolution surface (Fig. 4.21b).

- C, Co.** The capsule skeleton was provided by a line segment along the x-axis from -4 to 0. 11 blobs of width $\rho = 0.2$ were placed along the line segment to produce the shape in Fig. 4.21c and Fig. 4.21d.

¹First, the meshes (visible layers) were flattened and then, we applied a Poisson surface reconstruction filter with octree depth of 10, the solver divide was set to 9, and the remaining parameters were set to default. This software is available under <http://meshlab.sourceforge.net>

Patient	1	2	3	4	5	6	7	8	9	10	Total
Instances	32	39	23	25	36	56	26	51	26	34	348
Aneurysms	1	2	1	1	1	3	1	0	1	0	11

Table 4.4: Number of **RBT** instances per patient and the number of aneurysms segmented for 10 **3DRA** patient data.

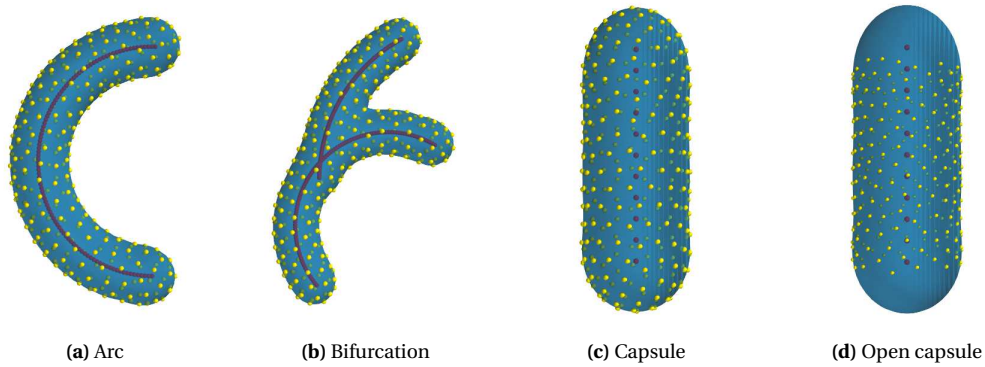


Figure 4.21: Synthetic data: four convolution surfaces (blue) generated with point-based skeletons (red). 300 points evenly spaced where placed on the surface.

For all shapes, an iso-value of 0.1 was used and the distance from the surface to one skeleton point was 1.0 mm. This distance is representative of a medium vessel width in our clinical data-set.

For providing input data to LIM, we employed an equi-distributed point-based sampling technique (Witkin and Heckbert (1994)). This way, 300 points were placed for each shape (Fig. 4.21) which served as a scattered representation of each input shape – thus breeding input point-set \mathcal{P} . For the open capsule, we placed 500 points on the surface and once all the points homogeneously placed, a bounding box ($C_{min} = [-4.1, -1.6, -1.6]^T$ and $C_{max} = [0, 1.6, 1.6]^T$) was utilized for clipping to 300 points and providing an open scattered representation (Fig. 4.21d). This synthetic data was used to evaluate the Local Implicit Modeling algorithm (Alg. 5).

4.8.3 Parametrization

The algorithm parameters were tuned on one node of the carotid artery of one patient. The tuning process was executed on this node (initial BM fitting its own and neighbor points), and thereafter the resulting parameters were used for all both patient and synthetic data. The energy weights were tuned so the energy values are of the same order of magnitude. For Lennard-Jones energy, s was set to 2 so that the natural distance between two blobs is twice the geometric mean of their width. The iso-level value T was chosen to be approximately half the voxel size. To ensure smooth transitions, a topological distance $\mathcal{F} = 2$ was used for generating the local point-sets \mathcal{P}_l for each node.

We chose “Cauchy” kernel for its computational efficiency and infinite support. All results were produced with following Cauchy kernel: $\phi(x) = (1 + x^2/5)^{-2}$ (dividing factor 5 normalizes the kernel such that $\phi''(1) = 0$). Note that our method is not kernel-dependent, and was successfully used with the computationally expensive Gaussian kernel.

During the clean-up procedure, blobs further than 20 mm from the node (5 times the diameter of the largest artery), and blobs whose width was below 0.02 mm (10% of

Parameter	Description	Value
α	hyper-parameter for the Lennard-Jones energy	10^{-5}
β	hyper-parameter for the mean curvature-based energy	10^{-3}
s	controls the repulsive distance in Lennard-Jones energy	2
T	iso-level value	0.1
N_s	maximum number of subdivisions	100
t_g	accuracy threshold (distance of a point to the surface)	0.3 mm

Table 4.5: Parameter values for the proposed vessel tracking and reconstruction algorithms.

the voxel size) were removed from the **BM**. Indeed, since we set the weight of each blob equal to its width, blobs whose width is below T do not generate any iso-surface (per se the maximum function value is $\rho_j < T$). Values are summarized in Table 4.5.

4.8.4 Evaluation measures

In order to quantify the error of fit, we would ideally have to compute the geometric distance between the ground truth surface \mathcal{S}_r and the implicit surface produced by **LIM**, hereafter the test surface \mathcal{S}_t . We first sampled \mathcal{S}_t using the same sampling technique as for synthetic data (**Witkin and Heckbert (1994)**), into a point set \mathcal{Q} . In the synthetic case, \mathcal{S}_r is available. For each point Q in \mathcal{Q} , we computed its **Geometric Distance (GD)** to \mathcal{S}_r by projecting Q onto \mathcal{S}_r resulting in Q' . We thus define:

$$GD(Q, \mathcal{S}_r) = |Q - Q'| \quad (4.32)$$

However, no \mathcal{S}_r surface is available with patient data and we have to resort to other means of evaluating the error of fit. We here took advantage of **Taubin's approximate Geometric Distance (TaGD)**:

$$TaGD(\mathcal{S}_t, P) = \frac{|T - f(P; p)|}{|\nabla f(P; p)|} \quad (4.33)$$

to qualify how close the implicit surface \mathcal{S}_t – defined as the T -isosurface of function f – fit an input point P .

Simple statistics (min, max, mean) on **TaGD** values could be used to quantify the error of fit onto a whole input point-set \mathcal{P} . However, \mathcal{P} is noisy, and contains a few outliers, such that the mean value of **TaGD** could be biased. We resorted to computing a robust statistic: the error of fit d_{BM} for a given **BM** is defined as the 90th percentile of **TaGD** over input point-set \mathcal{P} . The same error of fit is used with **GD** on synthetic data. Note that this criterion is very strict and much harder to meet than the usual mean or even median value.

A different distance is used for synthetic data and patient data. In order to validate the use of **TaGD**, we also computed it on synthetic data and compared it with ground truth criterion **GD**.

4.8.5 Validation protocol

We aim at providing figures characterizing compacity, accuracy and robustness to noise of **LIM**. To this end, we decomposed this assessment in two parts. First, we investigated

the accuracy, resilience to noise and compacity of LIM when applied to synthetic data where ground truth is available. Second, our LIM algorithm was applied to patient data. In this second study, we first visually assessed the outcomes produced by LIM. Next, we evaluated its accuracy w.r.t the threshold accuracy t_g and the number of subdivisions N_s .

In one hand, N_s influences the compacity of the resulting BM since it controls the maximum number of subdivisions. The by-product of this is also an impact on the capability of LIM to capture complex shapes. Put another way, a lower number of blobs authorizes simpler shapes while on the contrary, a raise in the number of blobs allows for more complex shapes. On the other hand, t_g has a direct impact on the accuracy of the reconstruction since it stops the reconstruction process when all points are at a distance threshold to the BM. We also provided figures for time computation on the entire data-set. Last but not least, we quantified the compacity of our model.

4.9 Results

4.9.1 Reconstruction on synthetic data

In a first step, we considered synthetic data – i.e. the four shapes described in 4.8.2 – as ground-truth. Hereafter, we designate the four shapes as the *reference surfaces* \mathcal{S}_r . Surface reconstruction was performed under two situations: noise-free reconstruction and reconstruction from noisy scattered data.

Noise-free reconstruction

We observed the evolution of N_s and how it affected the resulting BM. This study aimed at providing figures for LIM compacity and accuracy. t_g was set to 10^{-3} mm which is a sufficient value not to be responsible for stopping the reconstruction process. N_s took values ranging from 1 to 100 and for each value of N_s a BM was produced. Fig. 4.22 exhibits the d_{bm} for these BMs. Finally, we selected the BM presenting the lowest figure – among the 100 BMs – regarding the GD (corresponding N_s values are given in Fig. 4.22). Fig. 4.23 displays the selected BMs for each reference surface. By the same token, Table 4.6 recaps the minimum and maximum distances for these BMs, as well as the N_s providing such result from N_{bi} initial blobs.

Robustness to noise

We virtually corrupted noise-free data with noise in order to provide significant figures which translated LIM robustness to noise and accuracy w.r.t \mathcal{S}_r . Gaussian noise with standard deviation varying from 0.025 to 0.25 mm (step 0.025) was used for corrupting \mathcal{S}_r dense sampling whose positions were accordingly translated along the normal direction to the surface, measured at each noise-free data point P . The resulting noisy points constituted \mathcal{P} which was fed to LIM. t_g was set to three times the noise standard deviation and N_s was set to 100. 1K points (\mathcal{Q}) sampled homogeneously the resulting BM – obtained with the method described in Witkin and Heckbert (1994) – and were utilized for computing both GD and TaGD. Reconstruction, sampling of the final BM and

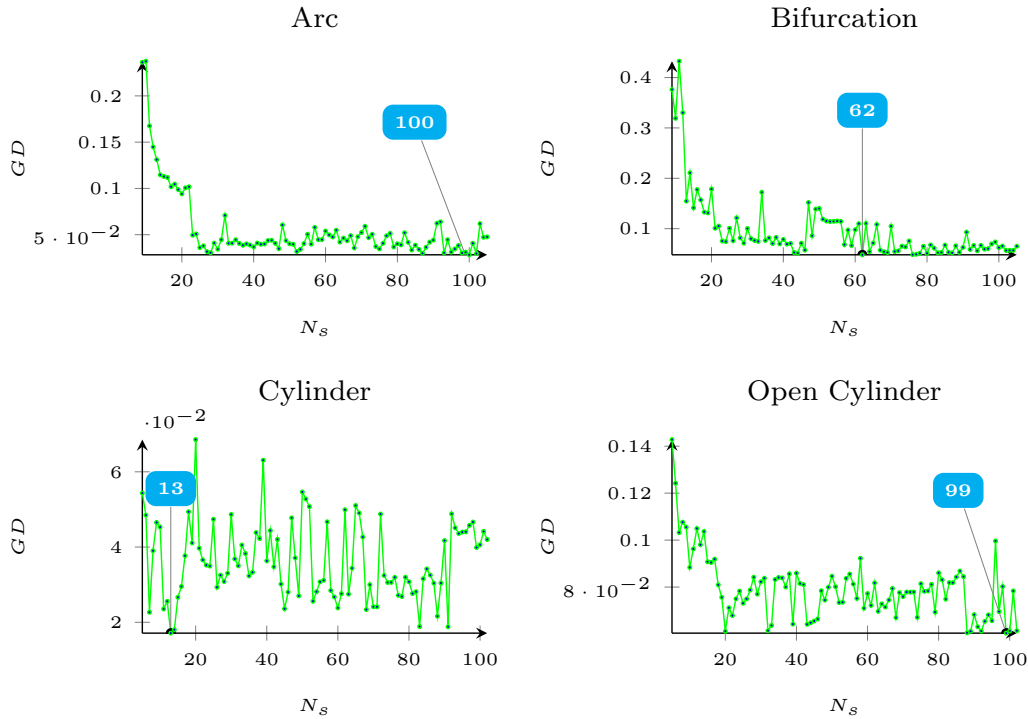


Figure 4.22: The maximal GD value for 100 BMs reconstructed from \mathcal{S}_r sampling. For sake of clarity, values 1 and 2 for N_s were excluded from charts. The BM with the lowest figure was selected (see Fig. 4.23) and quantitatively further detailed in Table 4.6. However, we can see that all the shapes were very well modeled with no more than 20 blobs.

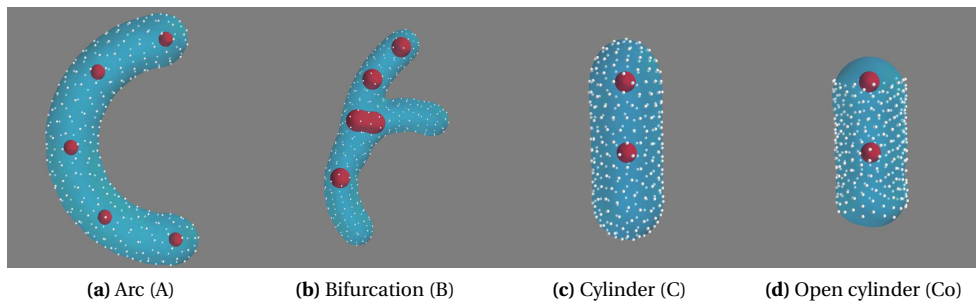


Figure 4.23: Fitting of 300 points (white) from initial BMs (red) and producing the blue surface.

computation of both distances were repeated 300 times for each reference surface, more precisely 30 times per noise standard deviation. For assessing the geometric precision, we measured the fit error d_{bm} (in mm) with both the GD and TaGD. In practice, d_{bm} was computed between \mathcal{S}_r and \mathcal{Q} for a given noise level.

Table 4.7 reports the minimum, median and maximum of the d_{bm} distribution for the 300 tests, as well as the corresponding distribution of the N_s carried out.

Shape		A	B	C	Co
TaGD	<i>min</i>	10^{-5}	$2 \cdot 10^{-4}$	$2 \cdot 10^{-5}$	$1 \cdot 10^{-4}$
	<i>med</i>	0.007	0.01	0.006	0.016
	<i>max</i>	0.027	0.051	0.016	0.063
GD	<i>min</i>	10^{-4}	$2 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$
	<i>med</i>	0.007	0.01	0.006	0.016
	<i>max</i>	0.028	0.048	0.017	0.060
N_b		100	62	13	99
N_{bi}		5	5	2	2
N_{br}		101	202	11	11

Table 4.6: Minimum (min), median (med) and maximum (max) distances (mm) computed between the original shape and the BM with the lowest geometric error to noiseless input points. Metrics are computed with **Taubin's approximate Geometric Distance (TaGD)** and the **Geometric Distance (GD)**. The initial BM was composed of N_{bi} . The reference surface was modeled by N_{br} blobs, meanwhile the resulting BM with the lowest GD presents N_b blobs. Four shapes were considered (see Fig. 4.23): arc (A), bifurcation (B), cylinder (C), open cylinder at extremities (Co).

Shape		A	B	C	Co
d_{bm} TaGD	<i>min</i>	0.016	0.046	0.015	0.038
	<i>median</i>	0.064	0.079	0.041	0.103
	<i>max</i>	0.146	0.211	0.164	0.175
d_{bm} GD	<i>min</i>	0.016	0.048	0.015	0.039
	<i>median</i>	0.063	0.078	0.042	0.119
	<i>max</i>	0.174	0.334	0.189	0.233
N_b	<i>min</i>	8	12	5	5
	<i>median</i>	15	15	7	9
	<i>max</i>	102	15	89	83

Table 4.7: Four shapes were considered: arc (A), bifurcation (B), cylinder (C), open cylinder at extremities (Co). All shapes have a radius of 1.0 mm. Gaussian noise was added along the normal to the surface to 300 points on the reference surface. The standard deviation σ varied from 0.025 to 0.25 mm (step 0.025), 300 tests per shape were run with $t_g = 3\sigma$ and other values set to values in Table 4.5. The minimum (min), median (med) and maximum (max) values of the error fit distribution d_{bm} (in mm) and computed with **Taubin's approximate Geometric Distance (TaGD)** and the **Geometric Distance (GD)** for the 300 noise configurations. For comparison with Table 4.6, the distribution of the resulting BM number of blobs N_b distribution is also given.

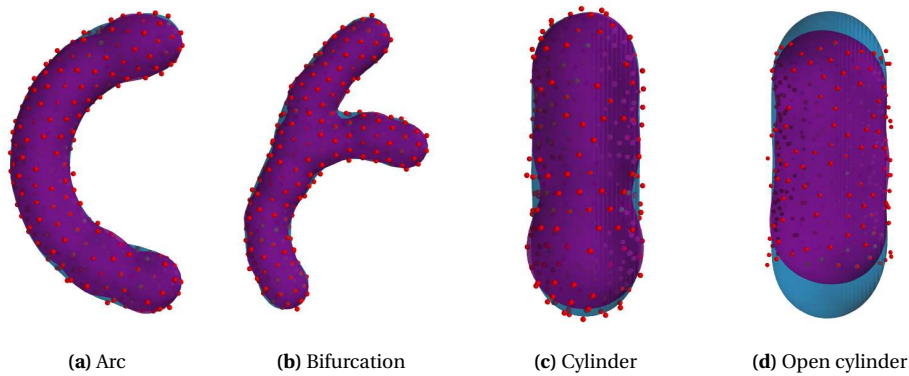


Figure 4.24: Reference surface (blue) and the resulting BMs (purple) for maximal values in Table 4.7.

Discussion

Table 4.6, our local implicit modeling framework confers to LIM a high robustness to bad initialization. As a result, a rough initialization of LIM with two blobs per node were enough when contending with real patient data. LIM can capture complex shapes from a few number of blobs. For instance, 5 blobs were necessary to initialize our modeling procedure for capturing a bifurcation composed of 202 blobs as illustrated in Fig. 4.23. In contrast to other point-based approaches – such as **Radial Basis Function (RBF)** (Buhmann (2003)) – which usually treat the problem of centers collocation by using a complete set/pruned version of point data (Turk and O'brien (2002); Mouat and Beatson (2002); Carr et al. (2003) and Macêdo et al. (2009)) or utilizing a dedicated approach to provide them (Samozino et al. (2006); Chen and Lai (2007) and Gelas et al. (2007)). Then, they adjust the implicit function weights to input data; we let the BM centers move which confers LIM robustness to bad initialization.

In a noise-free context, LIM displays a precision ranging from $0.01 \cdot 10^{-3}$ mm (Table 4.6) which represents $0.05 \cdot 10^{-3}$ -3% of the shape thickness (2mm). In average, 50% of input points lies at less than $0.01 \cdot 10^{-3}$ mm from the resulting implicit surface. A palpable increase in both distance maxima is noticed in a noisy context. In general, these values remain below the maximal noise standard deviation (0.25 mm) as Table 4.7 summarizes. Fig. 4.24 shows visually good surfaces which correspond to the resulting BMs for maximal values in Table 4.7. A word about N_b , Fig. 4.22 evinces that after a certain N_s , the precision of the fitting is degraded. For example, consider the open cylinder case with $N_s = 18$ (with value 0.061) against $N_s = 97$ (with value 0.060 in Table 4.6). Indeed, LIM attains an optimal N_b which is suited to the shape complexity. Although lower figures can be achieved, it implies to increase N_b which may hinder accuracy and incidentally, compactity. Besides, LIM will need more than one iteration to propagate and correct this deterioration. Note that LIM possesses a mean to counterbalance this problem through t_g .

When noise can be characterized (assuming a Gaussian model), LIM produces – in average – compact representations where a smaller number of blobs are required for the

same reconstruction complexity. For instance, the average ratio between the number of blobs producing the lowest GD values in Table 4.6 with the median number of blobs in Table 4.7 is 5. This feature of LIM confers to the final BM a low d_{bm} since it prevents the reconstruction process from pursuing levels of detail drowned in noise. It turns out that low values for N_b in Table 4.7 were the by-product of an adapted t_g for the shape complexity. For instance, the bifurcation required between 12 to 15 blobs. We believe that LIM used a lesser N_b since input points covered a larger area when compared to other cases. Consequently, LIM had more space for placing blobs without producing small fluctuations on the resulting implicit surface which could produce large fit errors. In which case, LIM may demand more blobs for propagating this error.

To evince the correctness of TaGD, we used both GD and TaGD for appraisal. It turns out that figures with both metrics were always similar for minimum and median values. Nevertheless, TaGD tends to slightly underestimate GD when further away from the surface but still of the same order of magnitude. With this in mind, we reckoned that TaGD could be used for computing d_{bm} for evaluating LIM results on real patient data.

4.9.2 Reconstruction on patient data

We also applied the LIM algorithm to real data. We used the same hyper-parameters as before (see Table 4.5). Only N_s and t_g remained to be fixed. We let $N_s = \{30, 50, 100\}$ and $t_g = 0.2, 0.3, 0.5$ mm vary for recording their impact on the accuracy of the reconstruction, the computation time and compacity of the model. Thereby, 9 algorithm configurations were investigated depending on parameters t_g and N_s .

4.9.3 Visual assessment

For providing a triangulated surface of our geometrical model, we used the method presented in Section 4.7. For this purpose, a volumetric representation of each local BM was produced (on a grid of the same voxel size as the 3DRA data) and used to drive a Marching Cubes algorithm. Finally, each triangulated surface was cut with regard to its neighbors and assembled as a stack of individual surfaces. Accordingly, a triangulated surface was computed for each algorithm configuration. A sample result on patient 10 for $N_s = 100$ is shown on Fig. 4.25. One can observe the impact of t_g on the volumetric representation: for $t_g = 0.2$ mm, variations on the main vessel surface are visible; in contrast, $t_g = 0.5$ mm smoothes them; $t_g = 0.3$ mm displays a good trade-off between accuracy and smoothness.

4.9.4 Accuracy

As already stated, ground-truth for comparing the resulting local BMs with the vascular surface is unavailable. Consequently, we only relied on the TaGD – validated on synthetic data – and used it for computing d_{bm} . We looked at how t_g impacted the reconstruction quality by including 4 classes for d_{bm} over all gathered patients: $d_{bm} < 0.5 \cdot v_s$, $0.5 \cdot v_s < d_{bm} < v_s$, $v_s < d_{bm} < 2 \cdot v_s$ and $2 \cdot v_s < d_{bm}$; where v_s is the voxel size. We used v_s because it was different for each patient and is related to the point extraction error.

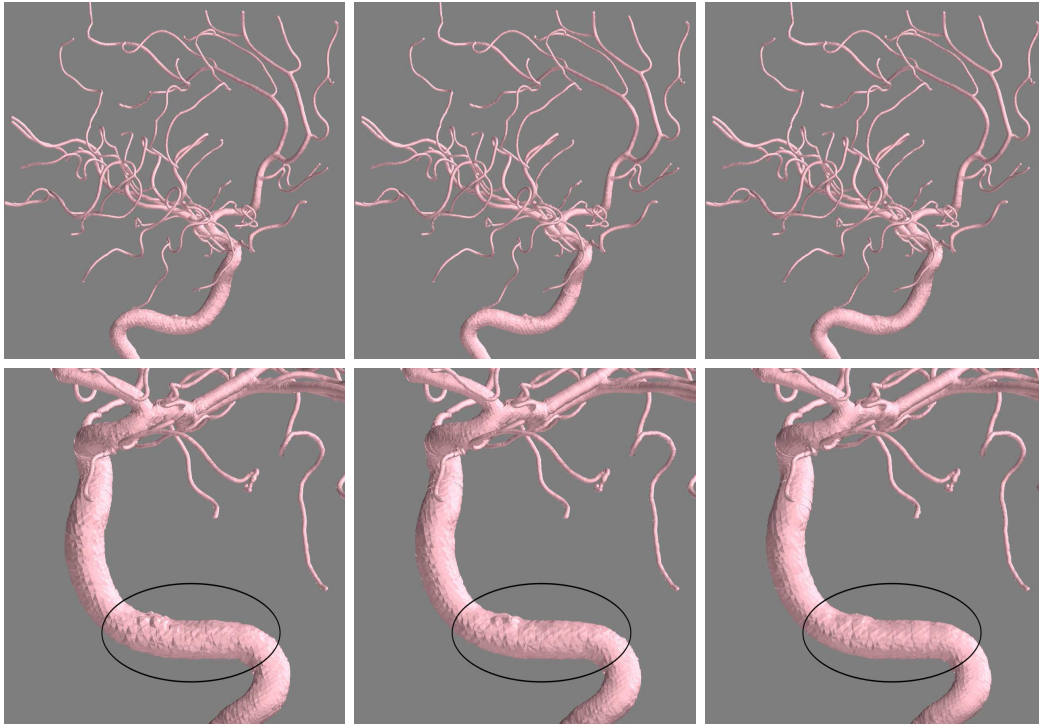


Figure 4.25: Geometrical model for patient 10 at different values for t_g : (From left to right) 0.2mm, 0.3mm and 0.5 mm. The top row displays the whole geometrical model while the bottom row presents their respective close-ups to the stem vessel. $t_g = 0.2$ mm presents variations in the stem vessel surface. On the other hand, $t_g = 0.5$ mm smooths these variations. $t_g = 0.3$ mm displays a fair trade-off between smoothness and accuracy.

t_g	0.5mm			0.3mm			0.2mm		
N_s	100	50	30	100	50	30	100	50	30
$d_{bm} < 0.5 \cdot v_s$	51.37	51.37	51.37	88.65	88.63	88.54	97.15	96.96	96.58
$0.5 \cdot v_s < d_{bm} < v_s$	47.67	47.64	47.61	11.00	10.96	10.97	2.60	2.73	3.01
$v_s < d_{bm} < 2 \cdot v_s$	0.84	0.85	0.86	0.25	0.28	0.34	0.15	0.20	0.26
$2 \cdot v_s < d_{bm}$	0.12	0.14	0.16	0.10	0.12	0.15	0.09	0.11	0.14

Table 4.8: Distribution in % of the **BMs** according to the error of fit (d_{bm} , in voxel) in 4 classes (left column). v_s is the voxel size of the **3DRA** data which ranges from 0.18 to 0.22 mm. 9 algorithm configurations were investigated depending on parameters t_g (targeted accuracy of fit) and N_s (maximum model complexity).

Table 4.8 reports the distribution, in percents, of d_{bm} measured on all 87564 **BMs** (for each configuration).

4.9.5 Computation time

For each configuration, i.e. given t_g and N_s , we measured the time – in minutes and seconds (min:sec) – required for reconstructing a patient. For a given t_g , we compared all N_s configurations against their time magnitudes. Since time magnitudes for N_s configurations were similar for a given patient, we exploited the average computation time

Patient			1	2	3	4	5	6	7	8	9	10
t_g	0.2	μ (min:sec)	27:59	23:30	15:36	11:14	15:37	32:46	17:40	25:27	12:27	18:22
		σ (min:sec)	05:14	04:12	02:58	01:10	01:25	04:15	03:12	01:21	00:47	01:02
(mm)	0.3	μ (min:sec)	16:25	13:58	10:30	06:29	10:32	22:43	11:44	16:42	08:45	12:42
		σ (10^{-4} sec)	4.7	6.9	8.5	0.8	6.1	13.2	7.3	1.4	3.7	1.9
	0.5	μ (min:sec)	09:09	07:59	06:56	03:20	05:49	12:40	07:01	08:10	04:48	06:14
		σ (10^{-4} sec)	1.5	3.3	6.2	0.4	5.3	4.6	2.9	1.0	3.2	0.6
N_b			9341	8307	4630	5260	8930	14042	5487	13866	6961	10740

Table 4.9: Time computation for 9 configurations grounded on the threshold accuracy $t_g = \{0.2, 0.3, 0.5\}$ (mm) and the number of subdivisions $N_s = \{30, 50, 100\}$ for each patient data. For each t_g configuration, the average computation time μ and its standard deviation σ were computed on all N_s configuration outcomes. Green and red colored cells respectively represent the lowest and the highest values for a t_g row. N_b gives the number of **BMs** modeling patient’s vasculature.

t_g (mm)		0.2	0.3	0.5
N_b	$N_s = 30$	766021	532831	325799
	$N_s = 50$	810056	546509	330852
	$N_s = 100$	857332	563953	339302
\bar{N}_b (for one patient)		81114	54776	33198
σ_{N_b} (for one patient)		4570.3	1562.1	683.3
Γ		9.5	6.4	3.9

Table 4.10: Compacity assessment for 9 algorithm configurations depending on parameters t_g (targeted accuracy of fit) and N_s (maximum model complexity). A total number of 87564 **BMs** for 10 patients were considered. The total number of blobs N_b per configuration is provided. For a given t_g and a patient, compacity Γ was computed as the ratio between its average number of blobs – computed on all N_s – and the total amount of **BMs** for this patient. The average compacity Γ is supplied – as the mean of all compacity ratios at the same t_g .

μ and its standard deviation σ . Table 4.9 accounts for these outcomes for all t_g configurations.

4.9.6 Compacity

First, we aimed at surveying the influence of t_g and N_s on the total number of blobs N_b composing all 10 patient implicit models. Therefore, we counted N_b for all 9 reconstruction configurations on the data-set. The number of **BMs** (number of nodes) needed to represent the 10 patients, were fixed by the tracking algorithm (87654 **BMs**). N_b didn’t depend much on N_s (see Table 4.10 first row). Therefore, we computed compacity Γ : for a given t_g and a patient, it was furnished as the ratio between its average number of blobs – computed on all N_s – and the total amount of **BMs** for this patient. The average compacity Γ is supplied – as the mean of all ratios compacity ratios at the same t_g . Table 4.10 summarizes these figures regarding the influence of t_g on the model compacity.

Second, we focused our surveying on the compacity of the proposed geometrical model **w.r.t** a triangular model. A rule of thumb to measure the compacity of our model, is to count the number of primitives used. For each patient, we compared the total number of blobs N_b to the number of triangles $\#\Delta$ in a vessel iso-surface mesh of a similar visual quality (see e.g. Fig. 4.26, left). Triangulated surfaces generated for the visual

t_g (mm)	N_s	$\frac{\#\Delta}{N_b}$		
		min	med	max
0.2	30	52	66	82
	50	48	62	79
	100	44	58	77
0.3	30	83	109	142
	50	78	107	141
	100	73	104	138
0.5	30	171	244	318
	50	166	240	314
	100	159	234	308

Table 4.11: For each configuration, a triangulated surface was generated from the final geometrical model reconstructed with LIM. The number of triangles $\#\Delta$ for the triangulated surface and the number of blobs N_b composing the local implicit model were recorded. Distribution of the minimum (min), median (med) and maximum (max) ratio between $\#\Delta$ and N_b for all patients is given.

inspection were also considered for this assessment. Table 4.11 reports the minimum, median and maximum distribution of ratios between the $\#\Delta$ and N_b for all patients and configurations.

4.10 Discussion

When visually assessing outcomes produced with LIM, these demonstrate the strength of our proposal at producing geometrical models presenting very good vessel resolution and smooth transitions between neighboring BMs (Fig. 4.25). Furthermore, LIM confers the possibility to capture tiny vessels, as well as different blood vessel sizes and widths; without mentioning its modeling capability at capturing complex topology. As depicted in the last row of Fig. 4.25, LIM possesses the ability to fine tune the detail level encoded in the geometrical model through one single parameter, i.e. the threshold accuracy t_g . Besides, a visual comparison between a geometrical model produced with a classical Marching Cubes algorithm and LIM reveals that more vessels are visible with our proposal (see Fig. 4.26 top row). Furthermore, vessels are correctly represented when considering KV issues, in other words, no blending issues (see Fig. 4.26 bottom left). In addition, LIM correctly modeled bifurcations which exhibited smooth transitions despite gaps between stem and branching vessels.

As a matter of fact, two parameters primarily control LIM: the number of subdivisions N_s and the threshold accuracy t_g . Together with synthetic data, LIM is capable of reducing the impact of noise on the final outcome. Table 4.8 shows that 99% of BMs present sub-voxel d_{bm} for all configurations. Besides, the lower t_g is set, the more BMs are below 0.5 voxel regardless N_s value. In essence, t_g regulates the precision of the BM w.r.t input data. Conversely, N_s has very little impact on this control.

Speaking about computation time, LIM takes between 3-33 min to model one patient. This time is dependent on the patient morphological complexity and the aimed

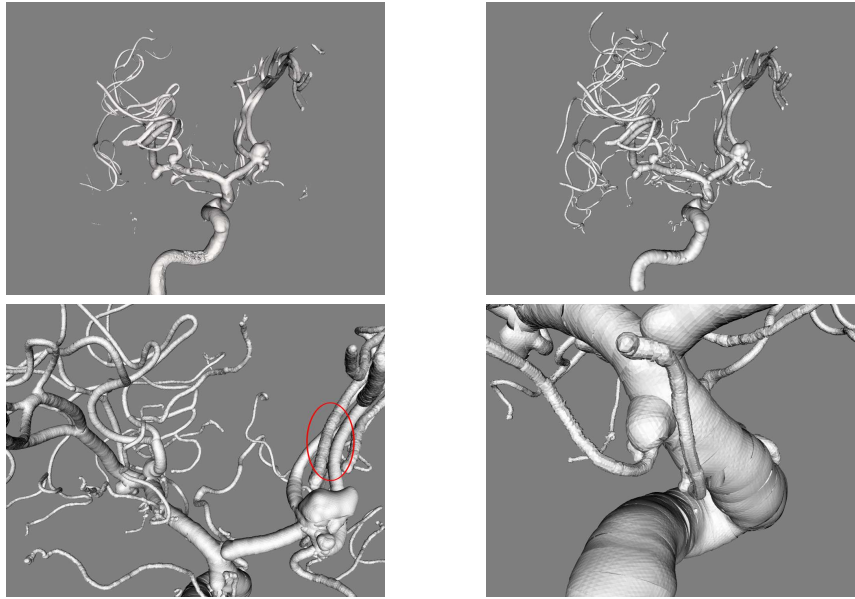


Figure 4.26: Visual assessment on patient 6 (from left to right): iso-surface from the raw data set ; iso-surface from the **BMs** (13956 models, 51272 blobs, $t_g = 0.3$, $N_s = 100$): much more vessels are visible; close-up showing smooth transitions between models ; close-up on a small artery branching onto the carotid: the connection is difficult to model

precision (t_g). For instance, patient 4 presented the lowest time figures among all patients (Table 4.9), meanwhile patient 3 possess the lowest N_b . This evinces that for a given **LIM** parametrization, the time for reconstruction is merely driven by patient morphology. As a rule of thumb, we can say that – for a given patient – time computation doubles when considering a certain t_g configuration to its lower counterpart. Besides, computation time for t_g equals 0.3 and 0.5mm displays similar figures regardless the value for N_s . On the contrary, we believe that $t_g = 0.2\text{mm}$ shows remarkable variations on time scores **LIM** modeling noise at this precision level (see Table 4.10), hence it directly impacts the time spent on optimization. We deem that the best trade-off between time computation and accuracy for our data-set is $t_g = 0.3\text{mm}$. Note that these figures were generated with 3 cores and it may be further reduced by using multi-cores frameworks.

In average, **LIM** implicit models – for all 87564 **BMs** – produced similar number of blobs N_b regardless N_s and compacity ranges from 4 to 9.5 blobs as Table 4.10 recaps. A noticeable difference in N_b is observed for $t_g = 0.2\text{mm}$. Since this value is closed to voxel size, **LIM** was driven to model also the noise. When comparing a triangular model with our implicit model, the ratio $\# \Delta / N_b$ ranged from 44 to 318, with an average of 134 (Table 4.11). Ratios are slightly similar for a given t_g regardless N_s . In conclusion, N_s has very little influence on the result. Therefore, we chose $N_s = 100$ to confer **LIM** high modeling capabilities without restrictions on the number of blobs. The best trade-off between compacity and precision remains for $t_g = 0.3\text{mm}$ with 6.4 blobs and a ratio of 100 triangles to 1 blob per **BM**.

4.11 Conclusion

In this chapter, we proposed a novel reconstruction method called **LIM** and dedicated to blood vessel modeling which produces a suited implicit model for numerical simulations. In practice, **LIM** relied on **RBT** input data but many other valuable methods can provide this initial structure (Lesage et al. (2009)). Generally speaking, **LIM** demands a set of points, located on the local blood vessel surface, should be associated with each point on the centerline as input data. However, should the vessel surface be available as a mesh; selecting the vertices in the vicinity of each point on the centerline should also provide such a point-set. In this case, our method could be seen as a mesh implicitization. **LIM** aims at enriching this centerline with an implicit surface fitting the blood vessel surface local dense sampling. This implicit surface satisfies simulation requirements: it provides a continuous scalar field, representative of the distance function for predicting collisions; and the implicit function gradient is aligned with the distance function gradient so as to give a valid recall force outside the implicit surface. For retrieving such an implicit surface, **LIM** proceeds in three steps. First, local data points are concatenated according to a topological distance $\mathcal{T} = 2$ and an initial **BM** composed of two blobs is created. Second, for each node on the centerline, the **BM** is iteratively fissioned and fine tuned so as it fits the local point-set. During the fitting process, centers are allowed to move, thus authorizing **LIM** to be particularly resilient to bad initialization. Third, the resulting **BM** is associated to its corresponding node on the centerline. Finally, the resulting set of local implicit surfaces is supplied under the same tree structure.

Results on synthetic and patient data showed the strength of **LIM** when dealing with noise and a small amount of outliers. To this end, we used the fit error d_{bm} based on two different distances: **Geometric Distance (GD)** and **Taubin's approximate Geometric Distance (TaGD)**. Besides, the minimum, median and maximum statistical values were also employed. First, we evaluated **LIM** on typical shapes such as curved and straight vessel portions; and bifurcations. Within this context, **LIM** exhibited its robustness against bad initialization – 2-5 blobs were utilized as initial **BM** during the reconstruction process. Our assessment was centered on robustness to noise, accuracy and compactness. The **LIM** algorithm produced watertight surfaces. When dealing with noisy point-sets, it was able to create – in average – compact representations which presented a fit error below noise level. In a second stage, **LIM** was applied to real patient data on which we evaluated the visual quality, accuracy, computation time and compactness of the resulting **BMs**. For this purpose, we let t_g and N_s vary leading to 9 configurations. Note that the ten patients were reconstructed thanks to 87564 **BMs**. These outcomes displayed smooth transitions and high modeling capabilities at reconstructing tortuous, tiny vessels; bifurcations and aneurysms. Besides, our local reconstruction approach allowed to handle nicely **KV** issues and thus avoid unwanted blending. Moreover, 99% of all **BMs** presented a fit error below voxel-size for a small amount of blobs (4 to 9.5) per **BM**. In a final assessment, we compared our proposal outcomes with classical triangulated surfaces for each patient and it turns out that **LIM** model was a very compact representation of the vascular tree (in average the ratio $\# \Delta / N_b$ was 134).

Previously, ten patient specific implicit models were generated using an infinite support kernel, i.e. a Cauchy kernel. Nevertheless, the proposed modeling method is generic and can make usage of compact support kernels. A double representation of the same underlying implicit function brings about a tremendous interest for collision detection and management. The usage of the compact kernel-based representation is preferred for inside/outside tests and for handling collisions when the interventional device is inside or close to the vasculature; meanwhile the infinite support kernel-based representation provides a mean to compute recall forces far outside the vasculature – e.g. after a sudden and strong motion whereas the implicit function gradient is null with the compact support kernel. Besides, a compactly supported kernel-based geometrical model may be easily obtained from an infinite support kernel representation. Indeed, one can easily translate blobs width into the new kernel domain and then resort to a single tuning iteration of all widths.

Despite good transitions on the final implicit model, discontinuities between models are present. These discontinuities are not kernel dependent, but are mostly sensitive to inhomogeneities in the point-set density. Although discontinuities are below voxel size, we believe they prohibit using our model for blood flow simulation using **Computational Fluid Dynamics (CFD)** at the present time.

A word on \mathcal{E}_κ constraint which is an external energy term in our current implementation. More precisely, it is computed at each data point:

$$\mathcal{E}_\kappa = \frac{1}{N_p} \sum_i \kappa(P_i)^2 \quad (4.34)$$

whereas its continuous counterpart is calculated over the entire implicit surface leading to a purely internal energy term:

$$\mathcal{E}_\kappa = \int \kappa(A)^2 dA \quad (4.35)$$

It turns out that \mathcal{E}_κ evaluation of Eq.4.34 is scale invariant. For illustration, consider the case of a single blob centered at the origin and fitting a unit circle which in turn, is sampled by $\{P_i\}$ points. The mean curvature of this blob is expressed as follows:

$$\kappa = -\frac{1}{r} \quad (4.36)$$

where r is the constant distance from points $\{P_i\}$ to the blob center. The fact of changing the blob's width has no impact on the evolution of \mathcal{E}_a value. That is because the points are fixed and not located on the T-level set.

With this in mind, a correct computation of \mathcal{E}_κ may demand a discretization or equi-distributed sampling of the implicit surface so that the integral term is better approximated. To this end, one can resort to similar sampling techniques as proposed in **Witkin and Heckbert (1994)**. This energy presents a double interest. First, \mathcal{E}_κ smoothes the surface according to the minimal area criterion when correctly computed. Indeed, surface area minimization is analogous to the square of mean curvature minimization (**Joshi and Séquin (2007)**). Incidentally, it turns out that this energy term has the same minimizer as the Willmore energy since its minimization is reduced to the square of mean

curvature minimization over the surface (Joshi and Séquin (2007)):

$$\begin{aligned} \text{Willmore Energy} &= \int \frac{1}{4} (\kappa_1 - \kappa_2)^2 dA & (4.37) \\ &= \int \left(\frac{\kappa_1 + \kappa_2}{2} \right)^2 dA - \int \kappa_1 \kappa_2 dA \end{aligned}$$

$$\begin{aligned} \text{Bending Energy} &= \int (\kappa_1^2 + \kappa_2^2) dA & (4.38) \\ &= \int 4 \left(\frac{\kappa_1 + \kappa_2}{2} \right)^2 dA - \int 2\kappa_1 \kappa_2 dA \\ &= \int 4\kappa^2 dA - \int 2\mathbb{G} dA \\ &= \int \kappa^2 dA + C & (4.39) \end{aligned}$$

where κ_1, κ_2 are the principal curvatures; \mathbb{G} is the Gaussian curvature; and C is a topological constant when the surface topology does not change during optimization. Second, it may lessen discontinuities introduced by inaccuracies due to bad calculation during the minimization process of \mathcal{E}_κ .

It is worth to point out that a lot of effort was dedicated at providing sound energy constraints with closed-form expressions. Moreover, we improved upon previous works (Muraki (1991); Bittar et al. (1995) and Tsingos et al. (1995)) since – to our knowledge – no subdivision process relying on a geometric criterion was introduced. A path for improving LIM points toward a correct computation of energy constraints, together with a method for better ensuring smooth transitions between BMs; and a proposal for the clean-up procedure as an energy constraint. In the next chapter, we quantify the efficiency of our implicit model created with LIM and compare it against classical meshed geometrical models routinely used in a real-time simulation context.

APPLICATIONS: COIL EMBOLIZATION SIMULATION

In this chapter, we review the simulation framework presented in [Dequidt et al. \(2007\)](#) and further detailed in [Dequidt et al. \(2009\)](#). Within this framework, we apply our dedicated geometrical model to an interactive simulation context. For this purpose, we consider validation of the computational efficiency and realistic behavior against synthetic data as well as clinical data. Furthermore, we quantify the increase in performance and in realism with the proposed model when compared to triangulated surfaces.

5.1 Introduction

In previous chapters of this work, we described a series of algorithms to model the arterial vasculature from patient data, as a tree of local implicit surfaces.

A large part of simulation realism, in particular for surgical simulation, relies upon the ability to describe soft tissue response during the simulated intervention. This formulation is unquestionably related to the nature and the purpose of simulation. In our application of coil embolization, we believe that realism is essentially grounded on a sound mechanical formulation of the interventional tools, the smoothness and accuracy of geometrical models for soft tissue and above all, the interaction between these two constituents. In this chapter, we aim at showing the strength and the affinity of the proposed implicit model in the interactive simulation context of aneurysm coil embolization. For this purpose, we compare our dedicated surface model with a classical polyhedral surface, namely a **Triangular Mesh (TM)**, within the deformation framework presented in both [Dequidt et al. \(2007\)](#) for convolution surfaces/implicit surfaces and [Dequidt et al. \(2009\)](#) for the case of polyhedral models. Both paradigms are available under the state-of-the-art deformation platform **Simulation Open Framework Architecture (SOFA)**¹ and are clearly not part of our contributions.

First of all, we expose the needs for a coil embolization simulation (Section 5.2). In section 5.3, generally speaking, an elementary step during simulation is distilled. Next, we describe the general outline of the deformation framework (Section 5.4). Later on, we describe the data at our disposal and the experiments we carried on (Section 5.5). Finally, we discuss about our outcomes (Section 5.6) and conclude (Section 5.7).

5.2 What do we need for a coil embolization simulation?

In the context of interactive coil embolization simulation, a user is integrated in the simulation process who may change the course of it. Indeed, the physician manipulates interventional tools such as guide-wires, coils and catheters; and he/she reacts to the behavior and shape of the tool during the simulation when navigating through the vasculature, as seen in fluoroscopy. This reaction is very difficult to model but is a crucial input. The calculation thus has to be interactive. Against this interactive background, it is often admitted that only a few milliseconds are available for computation so as to have *the feeling of interactivity*. For instance, refreshment rates for visualization should be at least 25 Hz; and for haptic feedback, a computation frequency of 300 Hz (1 kHz when possible) should at least be met ([Duriez \(2013\)](#)). The simulation must be realistic or accurate but above all, the simulation needs to adapt – in real-time – to the patient’s data and to take into account the user’s feedback. The interaction between the geometrical model and the interventional tool is ensured by the deformation framework. Therefore, the simulation realism is translated by two facts: the geometrical model for blood vessels should be as accurate as possible and the deformation of the simulated medical devices should allow similar motion to that observed in a real intervention (e.g. gliding, looping, etc). No deformation of the vascular model is intended since our pro-

¹<http://www.sofa-framework.org/>

posed geometrical model currently does not handle deformations (perspectives in this area are provided in Chapter 6). Besides, deformation of the vasculature assuredly improves realism but also introduces extra computational burden. As a result, the vasculature is considered as a rigid object for the remainder of this work. Despite this lack, we demonstrate the effectiveness and efficiency of our implicit model in a interactive context. Regarding the mechanical device deformations – in our simulations – we aim at modeling the global behavior of interventional devices, e.g. catheter, coil, guide, etc. In conclusion, the simulation is composed of two models: one for the mechanical devices and another for the vasculature.

5.2.1 Time integration and numerical choices

Before speaking about a plausible mechanical model, we briefly describe numerical choices concerning time integration utilized in the **SOFA** platform. Deformable models are often modeled as second order mechanical systems with non-linear equations. Moreover, these models are often constrained by complex boundary conditions. In our case, user interaction comes into play and consequently, we do not have a perfect control of these boundary conditions and variable simulation parameters such as gesture delays (virtuality against reality) and intensity of solicitations. To alleviate this issue, implicit integration schemes provide unconditional stability at the expense of solving a large system of equations. Indeed, such schemes support constant and large time steps while maintaining the computations as simple as possible. As a result, an integration approach based on implicit Euler and coupled with a linearization per time step is employed.

Let's consider Newton's second law in the case of a generic dynamic deformable model:

$$\mathbb{M}(u(t))\dot{v} = \mathbb{P}(t) - \mathbb{F}(u, v, t) + H^T \lambda \quad (5.1)$$

where $u \in \mathbb{R}^n$ is the vector of generalized **Degrees of Freedoms (DoFs)**, $\mathbb{M}(u) : \mathbb{R}^n \mapsto \mathcal{M}^{n \times n}$ is the inertia matrix, $v \in \mathbb{R}^n$ is the velocity vector. \mathbb{F} represents internal forces applied to the simulated object depending on the current state and \mathbb{P} gathers external forces (user inter-action). $H^T \lambda \in \mathbb{R}^n$ is the vector gathering the m constraint forces. H^T is the matrix containing the constraint directions and $\lambda \in \mathbb{R}^m$ is the vector gathering the constraint forces intensities. $\mathbb{M}(u)$ and $\mathbb{F}(u, v)$ are derived from the physics-based deformable model.

We aim at providing a linear time-stepping implicit integration of Eq. 5.1. To this end, let's consider the time interval $[t_i, t_f]$ whose length is $h = t_f - t_i$. Therefore, Eq. 5.1 becomes:

$$\mathbb{M}(v_f - v_i) = \int_{t_i}^{t_f} (\mathbb{P}(t) - \mathbb{F}(u, v, t)) dt + hH^T \lambda \quad (5.2)$$

$$u_f = u_i + \int_{t_i}^{t_f} v dt \quad (5.3)$$

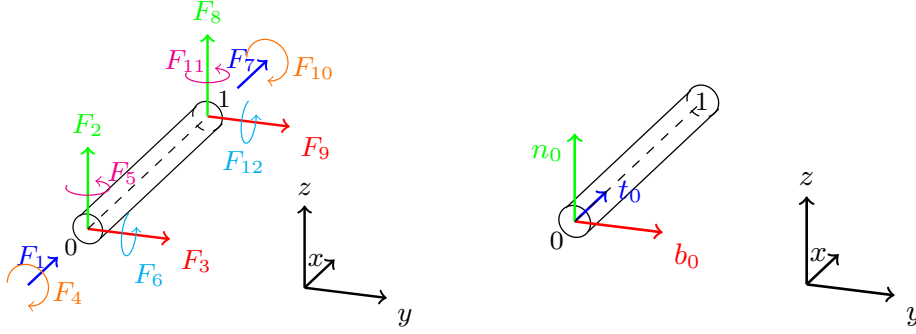


Figure 5.1: Forces acting on a beam element whose axis is defined by points 0 and 1 (left): axial forces F_1 and F_7 ; shearing forces F_2 , F_3 , F_8 and F_9 ; bending moments F_5 , F_6 , F_{11} and F_{12} ; and twisting moments (torques) F_4 and F_{10} . The local Frenet frame $\mathcal{F}_0 = [n_0, t_0, b_0]$ located at extremity 0 helps for expressing displacements induced by these twelve forces (right).

To evaluate integrals, the following implicit Euler integration scheme is adopted:

$$\mathbb{M}(v_f - v_i) = h(\mathbb{P}(t_f) - \mathbb{F}(u_f, v_f, t_f)) + hH^T \lambda_f \quad (5.4)$$

$$u_f = u_i + hv_f \quad (5.5)$$

But still, \mathbb{F} is a non-linear function, subsequently, we calculate its first order Taylor series expansion:

$$\mathbb{F}(u_i + du, v_i + dv) = F_i + \frac{\partial \mathbb{F}}{\partial u} du + \frac{\partial \mathbb{F}}{\partial v} dv \quad (5.6)$$

where F_i is the value of function \mathbb{F} at time t_i . We set $du = u_f - u_i = hv_f$ (Eq. 5.5) and $dv = v_f - v_i$. Next, we replace them in Eq. 5.6:

$$\mathbb{F}(u_f, v_f) = F_i + \frac{\partial \mathbb{F}}{\partial u} h(dv + v_i) + \frac{\partial \mathbb{F}}{\partial v} dv \quad (5.7)$$

Finally, we substitute Eq. 5.7 in Eq. 5.4 and obtain the following linear approximation:

$$\underbrace{\left(\mathbb{M} + h \frac{\partial \mathbb{F}}{\partial v} + h^2 \frac{\partial \mathbb{F}}{\partial u} \right)}_{\mathbf{A}} \underbrace{dv}_{\mathbf{x}} = \underbrace{-h^2 \frac{\partial \mathbb{F}}{\partial u} v_i - h(F_i - P_f)}_{\mathbf{b}} + hH^T \lambda_f \quad (5.8)$$

where P_f is the value of function \mathbb{P} at time t_f . Before solving the linear system $\mathbf{Ax} = \mathbf{b} + hH^T \lambda_f$, we need to find the intensity of the constraint forces at time t_f – i.e. λ_f . The mechanism for retrieving this unknown is provided in Section 5.3.

5.2.2 Mechanical model

Against a coil embolization background, medical devices are wire-like structures which behave like deformable rods or tubes. Interventional devices have a common denominator, their cross-section is negligible when compared to the device length. When this applies, interventional devices meet sound models grounded on beam theory. With this

in mind, the catheter, guide-wire and coil models consist of a series of non-linear deformable beam elements. As Fig. 5.1 illustrates, a beam element is a straight bar of uniform cross-section, length l , cross-section area A , having two extremities – designated by 0 and 1 in Fig. 5.1 – and capable of resisting axial forces F_1 and F_7 ; shearing forces F_2, F_3, F_8 and F_9 ; bending moments F_5, F_{11}, F_6 and F_{12} about the two principal axes in the plane of its cross-section, and twisting moments or torques F_4 and F_{10} about its centroidal axis (Przemieniecki (1985)). Axial, and shearing forces produce 3 translations x_1, x_2 and x_3 along the local Frenet frame $\mathcal{F}_0 = [n_0, t_0, b_0]$ on the extremity where they are exerted. By the same token, bending and twisting moments produce 3 rotations with respect to (w.r.t) the extremity – i.e. θ_1, θ_2 and θ_3 . More precisely, a beam element possesses twelve DoFs – i.e. 6 translations and 6 rotations due to both extremities – which are gathered in vector $X = [x_1, x_2, x_3, \theta_1, \theta_2, \theta_3, x_4, x_5, x_6, \theta_4, \theta_5, \theta_6]^T$ and expressed according to \mathcal{F}_0 . The variation of these twelve quantities between two simulation steps t_i and t_f – e.g. the positions of the beam – represents the displacement vector δX induced to the beam:

$$\delta X = X^{(t_f)} - X^{(t_i)} \quad (5.9)$$

For the remainder of this chapter, we call *node* the extremity or common extremities of the N beams composing the medical device (Fig. 5.2 left). Let $F^{(i)} = [F_1^{(i)}, \dots, F_{12}^{(i)}]^T$ and $\delta X^{(i)} = [x_1^{(i)}, \dots, x_{12}^{(i)}]^T$ be respectively the vector gathering all the forces acting on a single beam $i = 1, \dots, N$ and the vector encompassing the displacement originated in the same beam. Note that forces $\{F^{(i)}\}$ represent the internal forces derived from the physics-based model in Eq. 5.8 but expressed in their corresponding frame \mathcal{F}_i . Beam theory assumes that deformations remain *small* at the level of each element. When it is the case, the relation between the forces and displacements in the local frame is stated as follows:

$$\delta F^{(i)} = K^{(i)} \cdot \delta X^{(i)} \quad (5.10)$$

where $K^{(i)}$ is the stiffness matrix of beam i . $K^{(i)}$ matrix is symmetric and composed of four 6x6 matrices (Przemieniecki (1985) and Duriez (2013)):

$$K^{(i)} = \begin{bmatrix} K_{00}^{(i)} & K_{01}^{(i)} \\ K_{10}^{(i)} & K_{11}^{(i)} \end{bmatrix} \quad (5.11)$$

with matrix $K_{01}^{(i)} = K_{10}^{(i)T}$. Hereafter, we give the expressions for matrices $K_{00}^{(i)}$, $K_{10}^{(i)}$, and $K_{11}^{(i)}$:

$$K_{00}^{(i)} = \frac{E}{l} \begin{bmatrix} A & & & & & \\ 0 & \frac{12I_z}{l^2(1+\Phi_y)} & & & & \\ 0 & 0 & \frac{12I_y}{l^2(1+\Phi_z)} & & & \\ 0 & 0 & 0 & \frac{GJ}{E} & & \\ 0 & 0 & \frac{-6I_z}{l(1+\Phi_y)} & 0 & \frac{(4+\Phi_z)I_y}{1+\Phi_z} & \\ 0 & \frac{6I_z}{l(1+\Phi_y)} & 0 & 0 & 0 & \frac{(4+\Phi_z)I_y}{1+\Phi_z} \end{bmatrix} \quad \text{symmetric} \quad (5.12)$$

$$K_{10}^{(i)} = \frac{E}{l} \begin{bmatrix} -A & & & & & \\ 0 & \frac{-12I_z}{l^2(1+\Phi_y)} & & & & \\ 0 & 0 & \frac{-12I_y}{l^2(1+\Phi_z)} & & & \\ 0 & 0 & 0 & \frac{-GJ}{E} & & \\ 0 & 0 & \frac{-6I_y}{l(1+\Phi_z)} & 0 & \frac{(2-\Phi_z)I_y}{1+\Phi_z} & \\ 0 & \frac{6I_z}{l(1+\Phi_y)} & 0 & 0 & 0 & \frac{(2-\Phi_y)I_z}{1+\Phi_y} \end{bmatrix} \quad \text{anti-symmetric} \quad (5.13)$$

$$K_{11}^{(i)} = \frac{E}{l} \begin{bmatrix} A & & & & & \\ 0 & \frac{12I_z}{l^2(1+\Phi_y)} & & & & \\ 0 & 0 & \frac{12I_y}{l^2(1+\Phi_z)} & & & \\ 0 & 0 & 0 & \frac{GJ}{E} & & \\ 0 & 0 & \frac{6I_z}{l(1+\Phi_y)} & 0 & \frac{(4+\Phi_z)I_y}{1+\Phi_z} & \\ 0 & \frac{-6I_z}{l(1+\Phi_y)} & 0 & 0 & 0 & \frac{(4+\Phi_z)I_y}{1+\Phi_z} \end{bmatrix} \quad \text{symmetric} \quad (5.14)$$

with $G = \frac{E}{2(1+\nu)}$; where E is the Young's modulus and ν is the Poisson's ratio; I_y and I_z are the cross-section moments of inertia; $\Phi_y = \frac{12EI_y}{GA_{sy}l^2}$ and $\Phi_z = \frac{12EI_z}{GA_{sz}l^2}$ denote shear deformation parameters with A_{sy} and A_{sz} the shear area in the y and z directions. These matrices relate the **DoFs** with the exerted forces over one beam on its local frame. Consequently, we need to describe the deformations undergone by the whole tool.

Accordingly, we need a matrix relationship between the variation of the position in the local coordinate system δX and the variation of the node position in the global coordinate system δQ . This relationship is expressed by the matrix equation:

$$\delta X^{(i)} = \Lambda_{(i)} \delta Q^{(i)} \quad (5.15)$$

where $\Lambda_{(i)}$ is a diagonal matrix allowing the change of basis between the local and global coordinate systems.

As a result, the linearization of the element force-displacement equation is obtained in the global frame:

$$\delta \mathbb{F}^{(i)} = [\Lambda_{(i)}^T K^{(i)} \Lambda_{(i)}] \delta Q^{(i)} \quad (5.16)$$

where $\delta \mathbb{F} = [\delta \mathbb{F}^{(1)}, \dots, \delta \mathbb{F}^{(N)}]^T$ is the vector encompassing the associated forces. Likewise as Eq. 5.10 locally relates forces $\{F^{(i)}\}$ to beams displacements $\{\delta X^{(i)}\}$ in their respective frames $\{\mathcal{F}_i\}$, it is mandatory to define a common reference frame for all beam elements so that local displacements are referred to their vector force in this global coordinate system. Indeed, internal forces \mathbb{F} are added to external forces \mathbb{P} which are defined in a common reference frame (Eq. 5.8). The idea is to only work in a global coordinate system by using both the node position before deformation and the computed variation of its position after deformation. The sought relation expressing \mathbb{F} w.r.t displacements δQ in a global coordinate system is written as follows:

$$\delta \mathbb{F} = K \delta Q \quad (5.17)$$

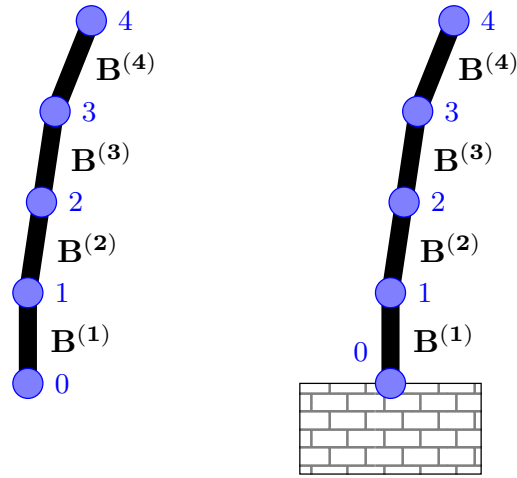


Figure 5.2: The medical device is modeled as a series of $N = 4$ beam elements $B^{(i)}/i = \{1, \dots, N\}$. (left) The mechanical device's stiffness matrix K – of dimension $(6(N + 1) \times 6(N + 1))$ – represented in a global frame has rank $6N$. (right) Meanwhile, the proximal extremity 0 in the mechanical device is fixed so that matrix K – of dimension $6N \times 6N$ and rank $6N$ – becomes invertible.

With this in mind, the stiffness matrix K of all beam elements is of the form:

$$K = \begin{bmatrix} \Lambda_0^T K_0 \Lambda_0 & 0 & \dots & \dots & 0 \\ \Lambda_1^T K_1 \Lambda_1 & 0 & \dots & \dots & \vdots \\ 0 & \Lambda_1^T K_1 \Lambda_1 & 0 & \dots & \vdots \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \dots & 0 & \Lambda_{N-2}^T K_{N-2} \Lambda_{N-2} & 0 \\ 0 & \dots & \dots & \Lambda_{N-1}^T K_{N-1} \Lambda_{N-1} & \vdots \end{bmatrix} \quad (5.18)$$

and of dimension $6(N + 1) \times 6(N + 1)$ and rank $6N$ – i.e. not invertible.

Note that there are forces summation at beam connections (common extremities) which is translated by the interpenetration of individual stiffness beam matrices (colored squares) in Eq. 5.18 and intuitively observed at nodes of two beams (Fig. 5.2 left).

One important feature of this model is that computation is performed segment by segment between only two local coordinate systems and then, deformations of the mechanical tool are retrieved automatically in a common frame.

A final word, the interventional tool is described by a curve which in turn is discretized in N beams as already stated. For computing an overall physics-based behavior, N is consequential which may hinder the interactive context. One can think about a possible trade-off between N and the realistic behavior. However, realism is our main concern. One important player in this area, is the interaction between the tool and the

blood vessel surface – i.e. the contact response. In order to leverage computation efficiency and contact response accuracy, each beam is in turn discretized in M points. As a result, the interventional tool is composed of $N \times M$ points for the collision detection phase and of N beams for the physical behavior computation. When one of these M points collides, the resulting force exerted at this point is distributed to its corresponding beam extremities (0 and 1). By this mean, a small number of beams presenting this technique are required for ensuring similar outcomes when compared to a classical beam approach.

5.2.3 Geometrical model

In this section, we review some of the interesting features for the models involved in this application: our **Blobby Model (BM)**, presented in Chapter 4; and a classical **Triangular Mesh (TM)**. Our stand point for this theoretical comparison, is the main characteristics of both models, in particular, applied to the specific context of coil embolization simulation. With this in mind, five points are to be revisited: accuracy of the object representation, efficiency for collision detection in a real-time context, smoothness and continuity of the model, higher derivatives computation and topology availability.

First of all, **Blobby Models (BMs)** are continuous representations of a desired object and whose analytical expression is available. In contrast, **TMs** are discrete models that parametrically represent a surface. As a result, one major difference between both models is smoothness. **BMs** – depending on the selected kernel – may be C^∞ continuous, while **TMs** are basically C^0 continuous but they may be promoted to C^2 continuous surfaces almost everywhere through a Catmull-Clark technique. One downside is that these techniques pursue a geometric criterion for interpolation without considering image data.

In one hand, **BMs** have proven to achieve sub-voxel accuracy for a relatively small number of blobs (Chapter 4). On the other hand, **TMs** are just an approximation of the true geometry and need to increase the number of triangles – to the detriment of the model compactness – to improve precision in the representation.

Regarding collision detection, implicit surfaces, and so **BMs**, supply an easy way to test whether a point is inside or outside the surface by probing the function value sign at the point position. In contrast, more sophisticated machinery is necessary when handling collision detection for triangular meshed objects but for which, a vast literature in this field is at hand (Teschner et al. (2005) and ?). Incidentally, **Bounding Volume Hierarchies (BVH)** have proven to be among the most efficient data structures in this area. Beyond that, these techniques may benefit from hardware acceleration and dedicated architectures (Avril et al. (2009)). Nevertheless, we deem that these methods may fail when dealing with **Kissing Vessels (KVs)**. Indeed, **BVH** only encodes the object geometry which – in this case – may be insufficient for detecting collisions since vessels are physically in contact.

Another key point to be highlighted is the availability of certain differential quantities, e.g. the normal. **BMs** provide the surface normal through the normalized gradient to the surface and **TMs** estimate the normal by computing the plane passing through

the triangle vertices. In the first case, normals are continuously, smoothly defined over the surface while in the second case, normals – for a patch of the true object covered by a triangle – turns out to be unique. When normal refinement is required, this ineluctably falls into the interpolation domain.

Our **Local Implicit Modeling (LIM)** procedure also provides the topology of the reconstructed vascular tree. More precisely, the topology encoding is automatically integrated during the tracking procedure with **RANSAC-Based Tracking (RBT)**. Whereas **TMs** exclude the topology in the parametrical representation, dedicated algorithms for retrieving the topology information and more precisely for centerline extraction (**Piccinelli et al. (2009)**) from meshed surfaces, exist (**Cornea et al. (2007)**). However, we believe that user inter-action is still required for correcting wrong outcomes, specially when dealing with **KVs**.

5.3 Elementary step simulation

In our case, the coil embolization simulation can be seen as two objects: one deformable, i.e. the medical device or tool; and the other rigid, i.e. the vascular geometry or patient data. For the sake of clarity, we rewrite Eq. 5.8 as Eq. 5.19 which – without loss of generality – applies to both objects:

$$\underbrace{\left(\mathbb{M} + h \frac{\partial \mathbb{F}}{\partial v} + h^2 \frac{\partial \mathbb{F}}{\partial X} \right)}_{\mathbf{A}} dv = \underbrace{-h^2 \frac{\partial \mathbb{F}}{\partial X} v_i - h(F_i - P_f) + hH^T \lambda}_{\mathbf{b}} \quad (5.19)$$

$$\begin{aligned} A_1 dv_1 &= b_1 + hH_1^T \lambda \\ A_2 dv_2 &= b_2 + hH_2^T \lambda \end{aligned} \quad (5.20)$$

As already stated, λ is unknown and can be seen as Lagrange Multipliers that place constraints on both objects so that a certain physical behavior is obtained. To solve these equations, we consider one simulation time step as follows: the tool is only driven by user supplied forces to a particular position – the so-called *free motion*; and then, the geometry is taken into account during the mechanical behavior computation of collisions between the tool and the vasculature. Up to this point, we have a set of potential contact spots $\alpha = \{\alpha_j\}_{1 \leq j \leq N_\alpha}$ and their associated Frenet frames \mathcal{F}_α . Once the collisions are identified, contact responses of the tool are computed – the so-called *correction step*.

5.3.1 Free motion

The mechanical tool is subject to forces \mathbb{P} induced from user interaction. To evaluate the new position of nodes, we first compute the velocities complying to Eq. 5.20 with no constraint – i.e. with $\lambda = 0$:

$$\begin{aligned} dv_1^{free} &= A_1^{-1} b_1 \\ dv_2^{free} &= A_2^{-1} b_2 \end{aligned} \quad (5.21)$$

In order to compute Eq. 5.21, matrix A needs to be invertible. However, it turns out that one needs to compute the force variation $\delta \mathbb{F}$ w.r.t the variation of the node position δX which involves the stiffness matrix K (Eq. 5.10 and 5.17). As aforementioned, matrix K is

however not invertible. For this reason, we fix/neglect the first beam's proximal extremity displacements so the first six columns/rows of matrix K – due to its symmetry – are removed (Fig. 5.2 right). This approach leads to an invertible version of K . Accordingly, forces are applied. Beyond that, displacements in free motion are thus computed from Eq. 5.7:

$$\begin{aligned} X_1^{free} &= X_{i_1} + h(v_{i_1} + dv_1^{free}) \\ X_2^{free} &= X_{i_2} + h(v_{i_2} + dv_1^{free}) \end{aligned} \quad (5.22)$$

5.3.2 Correction step

Once the user action on the mechanical device is accounted (free motion), the geometrical model is added to the simulation for considering possible interpenetration of objects and thus, correct the mechanical model behavior **w.r.t** the vasculature (correction step). The correction is addressed in two parts.

First, collision/interpenetration detection is performed and the way it is carried out, is dependent upon the targeted geometrical model. Recall that a beam is made of N beams which in turn are discretized in M points. Therefore, $N \times M$ points are possible candidates for collision. Function \mathbb{H} transposes the forces exerted on the M points to their corresponding beam extremities. Besides, the idea is to have a fixed mapping over time so that an interpolation procedure is avoided per iteration and collision. In practice, matrix H of Eq. 5.19 encodes this constant mapping. In essence, when the point α_j happens to be in collision/interpenetration with the blood vessel surface, the associated **BM** to this point can inform about its position **w.r.t** the surface by evaluating the implicit function value at α_j . On the contrary, **TM** relies only on dedicated techniques for speeding up the collision detection – for instance, in our simulation process, **BVH** are employed for spatial partitioning – since all the triangles may be inspected for one contact point.

Second, forces are applied to the mechanical device for correcting its position with regard to its contact responses and mechanical properties.

Hereafter, we briefly introduce the laws behind the contact formulation and then, we detail the paradigm for solving contacts.

Contact formulation

In our case of a rigid body and a deformable model, the contact is modeled as a unilateral constraint applied to contact point α . By unilateral constraint, we mean that there exists a surface through which α cannot pass. The α point is free to move in any direction on one side of the surface, but cannot pass through the surface. It is mathematically expressed as the inequality provided by the signed distance function from the contact point to the surface. Generally speaking, for each contact point α of the interventional tool, a peer point α' on the vasculature exists – i.e. the closest. Let $f_{\vec{n}_\alpha}^{(1)}(\alpha)$ be the force applied to the mechanical tool at α in the direction of the contact \vec{n}_α by the blood vessel surface. By the same token, we define $f_{\vec{n}_\alpha}^{(2)}(\alpha')$ as the force applied to the blood vessel surface at α' by the interventional tool. The action/reaction principle

gives: $f_{\vec{n}_\alpha}^{(1)}(\alpha) = -f_{\vec{n}_\alpha}^{(2)}(\alpha')$. For the sake of simplicity, we use $\lambda_{\vec{n}_\alpha} = f_{\vec{n}_\alpha}^{(1)}(\alpha)$. The gap between both objects at α is $\delta_{n_\alpha}(\alpha) = \vec{\alpha}' \cdot \vec{n}_\alpha$ and further shorten as $\delta_{\vec{n}_\alpha}$.

Hereafter, two friction laws are reviewed for complying with a realistic simulation.

Signorini's law In a few words, this law prohibits two objects to interpenetrate. The condition of contact are given for contact point α as:

$$0 \leq \delta_{\vec{n}_\alpha} \perp \|\lambda_{\vec{n}_\alpha}\| \geq 0 \quad (5.23)$$

\perp stands for a complementary relation – between the distance $\delta_{\vec{n}_\alpha}$ and the contact force $\lambda_{\vec{n}_\alpha}$ – imposing that one value must be null. In other words, the contact force is null if the points are not strictly in contact. Note that Signorini's law simulates a friction-less response along \vec{n}_α (Fig. 5.3). In other words no sticking is accounted. Dynamic problems often use a velocity formulation of this law, only valid during the time of contact:

$$0 \leq \dot{\delta}_{\vec{n}_\alpha} \perp \lambda_{\vec{n}_\alpha} \geq 0 \quad \text{if} \quad \delta_{\vec{n}_\alpha}(t) = 0 \quad (5.24)$$

because finding the associated acceleration is even harder since it is undefined at the time of contact.

Coulomb's friction law In contrast to Signorini's law, Coulomb's law fills in the lack of friction and for that, it describes the macroscopic behavior in the tangential contact space. The reaction force is included in a cone which height and direction is given by the normal component. If the reaction force is strictly included inside the cone, objects stick together, otherwise, the reaction force is on the cone's border and objects slip along the tangential direction \vec{t} . In this last case, the friction force must be directed along the direction of motion. The angle $\mu \in \mathbb{R}$ is a simulation-dependent parameter:

$$\begin{aligned} \dot{\delta}_{\vec{t}_\alpha} = \vec{0} &\Rightarrow \|\lambda_{\vec{t}_\alpha}\| < \mu \|\lambda_{\vec{n}_\alpha}\| && \text{(stick)} \\ \dot{\delta}_{\vec{t}_\alpha} \neq \vec{0} &\Rightarrow \lambda_{\vec{t}_\alpha} = -\mu \|\lambda_{\vec{n}_\alpha}\| \frac{\dot{\delta}_{\vec{t}_\alpha}}{\|\dot{\delta}_{\vec{t}_\alpha}\|} = -\mu \|\lambda_{\vec{n}_\alpha}\| \vec{t}_\alpha && \text{(slip)} \end{aligned} \quad (5.25)$$

Considerations. During **three-Dimensional (3D)** motion, the tangential direction is unknown. The only known fact is that the tangential force and the tangential velocity are opposite along a certain direction (to be found) – the problem is non-linear. Signorini's law and Coulomb's law are valid in multi-contact scenarios. For applying both laws to every contact we have to consider their coupling and their mechanical foundations. In other words, force exertion to a point on the interventional tool induces displacements in its neighboring points. Besides, when a contact response is computed, points not producing collisions may drive others to a collision state – i.e. new constraints in the resolution process – afterwards. This latter remark points out that the addressed problem is combinatorially complex.

For the sake of brevity, we simplify the expression of the relative displacement as follows:

$$\delta_{n_\alpha} = \mathbb{A}_{n_\alpha}(X_1, t) - \mathbb{A}_{n_\alpha}(X_2, t) \quad (5.26)$$

where \mathbb{A}_{n_α} is a mapping applied on the contact point for positions X_1 and X_2 of both colliding objects.

Same as before, we reduce the expression of the relative displacement velocity to:

$$\dot{\delta}_\alpha = \mathbb{H}_\alpha(X_1, t) v_1(t) - \mathbb{H}_\alpha(X_2, t) v_2(t) \quad (5.27)$$

that accounts for the velocity along \vec{n}_α and \vec{t}_α at contact point α . \mathbb{H}_α corresponds to the direction of contact based on the positions at the beginning of the time step. Furthermore, we suppose that this matrix does not change during the contact response process, leading to the constant matrix H of Eq. 5.19.

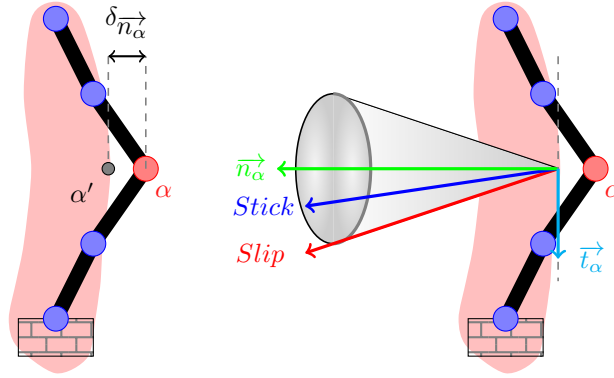


Figure 5.3: Unilateral contact formulation for one collision point α – pertaining to the interventional tool – of associated frame $\mathcal{F}_\alpha = [\vec{n}_\alpha, \vec{t}_\alpha]$. (left) Signorini's law: the closest peer of α on the vasculature is retrieved – i.e. α' . The gap $\delta_{\vec{n}_\alpha}$ between α and α' must be null to avoid interpenetration. Coulomb's law: The force exerted at α is decomposed into normal and tangential components. There is respectively sticking and slipping along \vec{t}_α when the resultant vector is inside the cone and over the cone.

Solving contacts

Herein, we expose how the laws of contact (Eq. 5.23) and friction (Eq. 5.25) are solved while taking into account the mechanical behavior. We introduced Eq. 5.21 and 5.22 for node position and velocity computation. These translated the user's action exerted on the mechanical device which in turn moved freely in space. Indeed, this process was fulfilled under the assumption that no correction was required ($\lambda = 0$).

Now, we are interested in finding the corrective forces to be applied at contact points. To this end, we compute the corrective motions in Eq. 5.20 with b_1 and b_2 set to null:

$$\begin{aligned} dv_1^{cor} &= hA_1^{-1} H_1^T \lambda \\ dv_2^{cor} &= hA_2^{-1} H_2^T \lambda \\ X_1^{cor} &= X_{i_1} + h(v_{i_1} + dv_1^{cor}) \\ X_2^{cor} &= X_{i_2} + h(v_{i_2} + dv_1^{cor}) \end{aligned} \quad (5.28)$$

Until now, no link with the free motion was made. Thereby, the correction phase and the free motion must be coupled. Note that the free motion may produce a certain number of contact points which present different interpenetration distances. Such

distances must be minimized so that interpenetrations are eliminated (Signorini's law). Accordingly, the constraint laws are linearized as follows:

$$\delta = \underbrace{\mathbb{A}_\alpha(X_1^{free}) - \mathbb{A}_\alpha(X_2^{free})}_{\delta^{free}} + hH_1 dv_1^{cor} + hH_2 dv_2^{cor} \quad (5.29)$$

and gathering Eq. 5.29 and 5.28 in a single equation²:

$$\delta = \delta^{free} + h^2 \underbrace{[H_1 A_1^{-1} H_1^T + H_2 A_2^{-1} H_2^T]}_{\mathbf{w}} \lambda \quad (5.30)$$

When only considering Signorini's law, Eq. 5.30 leads to a **Linear Complementary Problem (LCP)**. Together with Signorini's law, this problem becomes a **Non-Linear Complementary Problem (NLCP)** when coupled to Coulomb's law. Besides, the same relation stands for a velocity-based approach which is more stable for discrete collision algorithms (grounded on interpenetration detection):

$$\dot{\delta} = \underbrace{H_1 v_1^{free} - H_1 v_2^{free}}_{\delta^{free}} + \mathbf{w} \lambda \quad (5.31)$$

Accordingly, λ is obtained through a Gauss-Seidel algorithm dedicated to **NLCP** – i.e. involving contact frictions. Finally, the corrective motions are recomputed and applied:

$$\begin{aligned} X_{1,t+h} &= X_1^{free} + h dv_1^{cor} & \text{with} & \quad dv_1^{cor} = A_1^{-1} H_1^T \lambda \\ X_{2,t+h} &= X_2^{free} + h dv_2^{cor} & \text{with} & \quad dv_2^{cor} = A_2^{-1} H_2^T \lambda \end{aligned} \quad (5.32)$$

5.4 Overall simulation workflow

Three steps are required to solve Eq. 5.19. First, we compute displacement X^{free} in free motion from Eq. 5.21. Second, we perform collision detection leading to α contact spots. For each α , the interpenetration distance δ_j^{free} is evaluated and its matrix H_j is updated according to the direction of the contact. Third, we compute λ by solving Eq. 5.31 (or Eq. 5.30). Nevertheless, some values for λ may indicate that not all constraints are necessarily required. Redundant constraints (for which the corresponding value in λ is negative) are deactivated. This is the so called *status method*. At this point, we find the sought corrective motions through Eq. 5.32. However, some contacts may not satisfy the interpenetration constraint since we use a linear approximation of the local shape at the point contact. That is, the plane defined by the contact point α and the the normal vector \vec{n}_α (Fig. 5.3 and Fig. 5.4). As a consequence, collision detection is performed on the new configuration to check if some contact constraints are still violated (lines 20 to 25 in Alg. 6). When it is the case, a new evaluation of H is carried out, and new corrective motions are computed. This iterative procedure is repeated until all current contacts are solved. Practically speaking, the number of iterations is dependent upon the application and the number of contacts. Up to this point, all contacts initially detected are solved. However, as was previously stated, it is likely that new contacts appear while

²which represents the *Delassus* operator

solving them. This behavior is typical of any collision response algorithm. In this case, all contacts are solved within a given time step (lines 9 to 26 in Alg. 6), rather than the next time step. Checking for new contacts within the same time step adds a computational overhead but ensures a more consistent configuration at the beginning of the following time step.

Algorithm 6 Simulation pipeline

```

1: Find  $X^{free}$ 
2:  $contact = ()$ ,  $done = true$ 
3: for  $j = 1 \rightarrow N_\alpha$  do
4:   if DETECTCOLLISION( $\alpha_j(X^{free})$ ) then
5:      $contact += (H_j, \delta_j^{cor})$ 
6:      $done = false$ 
7:   end if
8: end for
9: while  $!done$  do
10:  repeat
11:    Solve  $\delta = \delta^{free} + w\lambda$  ▷ // I) Constraints deactivation using status method
12:     $done = true$ 
13:    if  $\exists j / (\lambda_j < 0)$  then
14:      Remove from contact:  $contact_k / (\lambda_k = \min(\lambda_j))$ 
15:       $done = false$ 
16:    end if
17:  until  $done$ 
18:   $X_{t+h} = X^{free} + hdv^{cor}$  ▷ // II) Constraints activation
19:   $done = true$ 
20:  for  $j = 1 \rightarrow N_\alpha$  do
21:    if DETECTCOLLISION( $\alpha_j(X_{t+h})$ ) then
22:       $contact += (H_j, \delta_j^{cor})$ 
23:       $done = false$ 
24:    end if
25:  end for
26: end while

```

5.4.1 Collision detection

Local implicit model

A final word on collision detection, the collision detection consists in finding the intersection between the T-level set of the implicit function f and a point α within two time steps t and $t + 1$. This is to find from α the closest point α' on the surface \mathcal{S} . Technically speaking, one can employ a gradient descent approach such as a Newton-Raphson algorithm. For computation efficiency, we employ a heuristic grounded on **Taubin's approximate Geometric Distance (TaGD)** (Taubin (1991)):

$$d^*(\alpha, \mathcal{S}) = \frac{|f(\alpha)|}{|\nabla f(\alpha)|} \quad (5.33)$$

where $\nabla f(\alpha)$ is the implicit function gradient evaluated at α . With this in mind, the sought point is retrieved as follows:

$$\alpha' = -d^*(\alpha, \mathcal{S}) \frac{\nabla f(\alpha)}{|\nabla f(\alpha)|} \quad (5.34)$$

The above equation informs that α' is located at distance $d^*(\alpha, \mathcal{S})$ from α in the gradient descent direction. For using this estimation, we assume that points between two simulation steps move very little or that the time step is small. Moreover, we reckoned Taubin's distance was a good approximate of the geometric distance in Chapter 4.

Once α' computed, we compute a linear approximation of the surface, i.e. the plane passing through α' and perpendicular to vector $n_{\alpha'}$. This vector is provided by the normalized implicit gradient evaluated at point α' . From this point, we can easily retrieve the associated Frenet frame $\mathcal{F}_{\alpha'} = [n_{\alpha'}, t_{\alpha'}, b_{\alpha'}]$. Since this plane is only a valid approximation of the surface around α' , the corrected motion α^* may not lead to an actual position on \mathcal{S} . Under those circumstances, the above steps are reiterated.

From a mathematical vista, this method is close to the secant method algorithm to find the root of a function. Indeed, in convex cases, the distance between the corrected position of α^* and \mathcal{S} decreases through the successive iterations and finally, it leads to a point lying on the surface. Such method is proved to converge in convex cases.

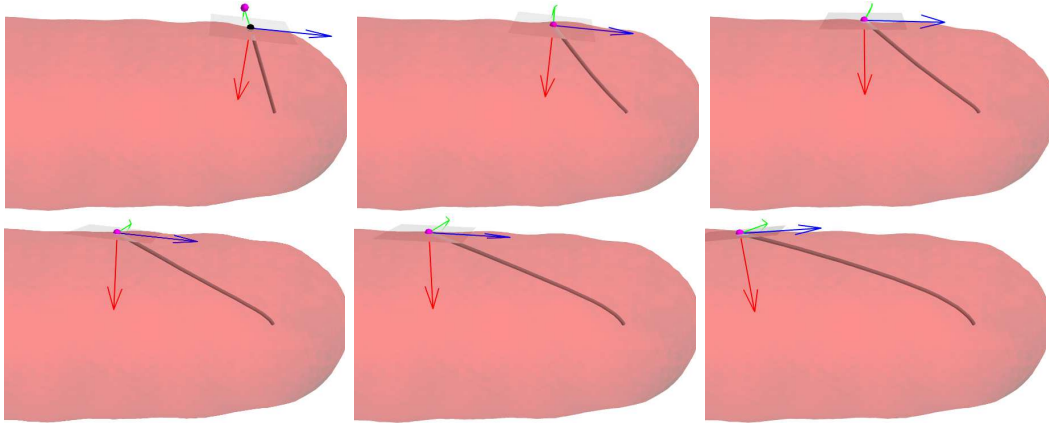


Figure 5.4: From left to right and top to bottom: six simulation steps from 0.06–0.011 ms (time step 0.001 ms). Only collision detection for the tool tip is displayed. For each time step, the tip initial position (before correction) in magenta and the corrected position in black (superimposed in all cases aside for 0.06 ms) are shown. The normal vector (red), the tangential vector (blue), the binormal vector (green) and the linear representation of the surface (plane in light blue) at the corrected position are also displayed.

Triangular mesh

For computing collision detection with a **TM** of the blood vessel surface, we used **BVH**, namely **Axis Aligned Bounding Box (AABB)**. Let a **AABB** $\mathcal{O} = \cup \mathbf{C}_{i=1 \dots N_c}$ be a collection of N_c cells which encompass the vasculature triangles – i.e. $\mathbf{C}_i = \{\Delta_j\}_{1 \leq j \leq N_i}$. Similar as for the implicit surface case, the idea is to determine the closest point α' on the **TM**. To this end, we look for cell $C_{i'}$ containing point α . We compute the Euclidean distance

$d(\alpha, \{\Delta'_j\})$ from α to all the triangles contained in this cell. A triangle – presenting a distance $d(\alpha, \Delta'_j)$ below an alarm distance d_a – is considered as a potential candidate for collision and afterwards, stored for further processing. These candidate triangles $\{\Delta'_j\}$ are a local representation of the vessel surface, and consequently, potential spots for placing mechanical constraints. As a result, constraints are placed at their vertices and their corresponding Frenet frames are computed. However, a single resultant force is built, by weighting all contributions forces of the neighboring $\{\Delta'_j\}$ triangles, and exerted on the closest triangle to α . These steps are repeated until no more contact violations are evaluated.

Note that this approach leads to an increase in the number of constraints per contact point which is cumbersome. Whereas no continuous description of the surface is available, it is mandatory to store triangles $\{\Delta'_j\}$ for preventing a tool point from leaving the vasculature. Indeed, one way to achieve this is by placing constraints on the local surface of the vessel. Nevertheless, if the distance $d(\alpha, \{\Delta'_j\})$ is above d_a , there is no mean to bring back the point to the surface without resorting to curve interpenetration analysis of the tool. In short, the choice of d_a is capital for avoiding these problematic situations and challenging to fix in advance.

5.5 Experimental details

Expected benefits of LIM from a simulation standpoint are related to the computation efficiency of the collision detection, the overall robustness of the collision treatment and improve realism. The evaluation of these benefits was conducted using synthetic test scenarios as well as models gathered from actual data such as a silicon phantom or patient data-sets. These scenarios are based on the same principle: a wire-like surgical tool is progressively deployed in a closed surface. During the deployment, the tool is touching, colliding and sliding on the surface; meanwhile the tool motion, simulation metrics and timings are recorded. The tool was modeled using serially-linked beams with 200 elements. The mechanical properties of the surgical tool were chosen to match tools encountered in actual interventional radiology procedures such as catheters (mainly composed of silicone rubber), guidewires and coils (which are both composed of titanium or alloys). Therefore the Young's modulus of tools ranges from 0.05 to 100 GPa. We chose a very stiff medical device of 100 GPa for the Young's Modulus and 0.48 for the Poisson's ratio. The time step was fixed to (1 ms), for all simulations, which is a plausible magnitude in an interactive context.

5.5.1 User-interaction

We opted for a simple scheme by imposing speed (50 mm/s) and direction at the fixed node of the mechanical device. By this way, tools were allowed to freely evolve within the geometrical model and slide according to the geometrical model surface.



Figure 5.5: The silicon vascular network used for validation.

5.5.2 Data

We present the input data that was at our disposal and the guideline followed for retrieving the geometrical models. Geometrical models were obtained from two sources: synthetic data and real data.

Synthetic data

We aimed at providing similar cases to those pertaining to real data. Blood vessels are locally cylindrical and may exhibit high curvature. As a result, we generated a cylindrical shape (*capsule*) and a *spiral* which mimicked both characteristics of the vasculature (Fig. 5.6). In order to provide a smooth and continuous representation of both shapes, we used convolution surfaces coupled with Cauchy kernels. The capsule skeleton or centerline (white line Fig. 5.6 left) was provided by a line segment (along the x-axis from 0 to 10). The skeleton of the spiral was constructed using the Archimedean formulation:

$$P = \begin{pmatrix} \rho \\ \theta \end{pmatrix} = \begin{pmatrix} 15.75 + (15.75 + \pi)\theta \\ \theta \end{pmatrix} \quad (5.35)$$

where points P of the skeleton were given in the x-y plane with $\theta \in [3.14, 540]$.

Triangular mesh The scalar value for both shapes was computed using a rectangular grid and then, a marching cubes algorithm was applied to the grid. For the capsule, we considered a highly detailed mesh (12K triangles), while in contrast, 3 level of details (2.74K, 11.3K and 45K triangles) were obtained for the spiral in order to test the influence of the number of triangles in the same shape complexity. Since the resultant vertices of the **TM** might not be on the surface, a Newton-Raphson projection technique was carried out. At the end of the projection step, vertices laid in average within a distance of 10^{-7} mm to the surface (Fig. 5.6 middle). Note that the projection step was done along the normal to the surface, and did not modify the mesh topology.

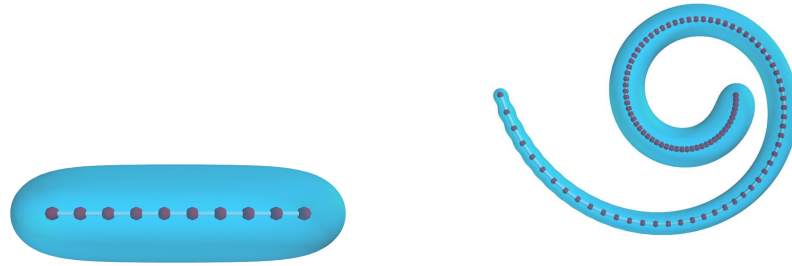


Figure 5.6: Capsule (left) and spiral (right) surfaces; and their corresponding skeleton (white). Both shapes presented respectively dimensions (width×height×depth) in mm: $11.9 \times 2.5 \times 2.5$ and $52.3 \times 39.9 \times 7.2$. A primitive was placed at each red dot composing the skeleton.

LIM We mimicked **RBT** output data in order to reconstruct both shapes with our **LIM** algorithm. Recall that **RBT** furnishes a stack of cylinders (center, direction, radius and height) and a dense sampling. For both capsule and spiral shapes, we used the above generated **TM**. We aimed at providing a dense and equi-distributed sampling of the shape of interest (mimicking **RBT** output dense sampling). For this purpose, the vertices of the generated triangulated surfaces were fed to a particle system (Witkin and Heckbert (1994)). A total of 3K particles were aimed. After 2000 iterations, the resulting position of particles was used as a dense sampling of the shape of interest. The skeleton became the **RBT** tracked centerline – directly encoding the topology – for both shapes. Every particle position was associated to its closest point on the centerline (Fig. 5.7). The direction was provided by the difference of positions between two consecutive points on the centerline and whose norm supplied a local estimate for the height. The radius was set to the distance between the center position and a closest point on the surface. Finally, both shapes were reconstructed with **LIM** algorithm using parameters in Table 5.1, a topological distance of 2 and Cauchy Kernels.

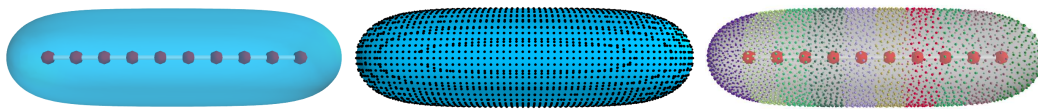


Figure 5.7: (left) Convolution surface for a line segment created with a Cauchy kernel. (middle) Projected vertices generated from a marching cubes algorithm. Their projection on the implicit surface is carried out via a Newton Raphson technique. (right) Points evenly spaced on the implicit surface that are associated (colored dots) to their closest node in the *centerline*.

Silicon phantom

A silicon phantom (Elastrat, Geneva) was used for representing the vascular network. The rigid phantom holds three aneurysms (Fig. 5.5). **3D Rotational Angiography (RA)** acquisition of the phantom was obtained in a C-Arm (Innova 4100, GE Healthcare) –

Parameter	Description	Value
<i>RBT</i>		
p_{inl}	inlier rate threshold	70%
r_t	relative threshold w.r.t local cylinder radius	10%
N_r	number of rays thrown to extract points	162
N_d	number of axis direction tested	81
N_p	number of points per ray	128
N_{min}	minimum number of tests for RANSAC	220
N_{max}	maximum number of tests for RANSAC	500
<i>LIM</i>		
α	hyperparameter for the Lennard-Jones energy	10^{-5}
β	hyperparameter for the mean curvature-based energy	10^{-3}
s	controls the repulsive distance in Lennard-Jones energy	2
T	isolevel value	0.1
N_s	maximum number of subdivisions	100
t_g	accuracy threshold (distance of a point to the surface)	0.3 mm

Table 5.1: Parameter values for tracking (*RBT*) and reconstruction (*LIM*) algorithms.

presenting as a 512^3 isotropic voxel cube with 0.2 mm voxel size – at the CHU Nancy, department of neuroradiology.

Triangular mesh The *3DRA* data was segmented with a classical algorithm: a first triangulated surface was extracted by a marching cube algorithm, the main connected component was kept, and the surface was refined by an active surface algorithm using the gradient norm as data energy term (*Lachaud and Montanvert (1999)*). The resulting triangulated surface was composed of 100k triangles.

LIM The *3DRA* volume was first tracked with *RBT* and then modeled with *LIM*. Configuration values for both algorithms are given in Table 5.1. A topological distance of 2 and Cauchy Kernels were chosen for reconstructing the silicon geometry. The tracked centerline was composed of 558 cylinders and 310K points which produced a geometrical model of 9.5K blobs.

5.5.3 Validation criteria

For assessing the efficiency of *LIM* against *TM* models, the assessment was threefold. First, we compared the simulation outcomes in terms of realism. To this end, the motion of medical devices were visually inspected. Second, we evaluated the theoretical trajectory – that the interventional tip should present – with those produced by both geometrical models. Third, we compared the time required for detecting collision.

Visual assessment

We considered the surface smoothness during the visual inspection of the tip trajectory. Indeed, the tip may slide on the surface with more or less difficulty depending on the surface quality. Roughness on the surface produces artificial friction which may induce

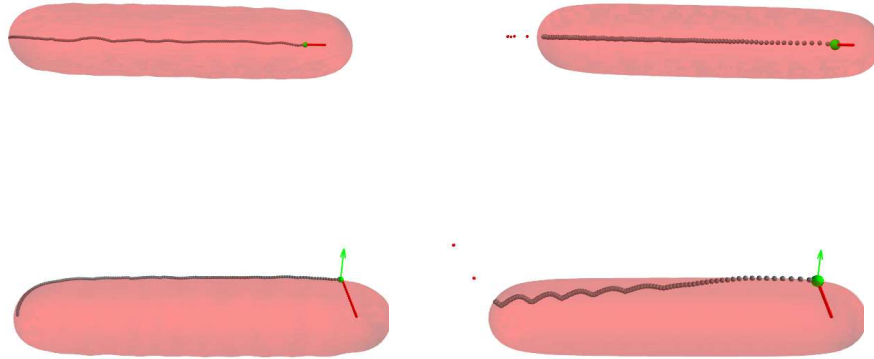


Figure 5.8: Tool tip trajectory for both LIM (left) and TM (right) models. Points utilized for computing \mathcal{L} : points leaving and not colliding with the surface are excluded (red). The first colliding point P_0 (green) and its normal vector define plane \mathbb{P} . The selected points \mathcal{P} (gray) are projected onto \mathbb{P} and the resulting positions are used for fitting line \mathcal{L} (top). Motion smoothness was quantified by computing the distance from the tip positions to the surface (bottom).

non-linear movements on the tool such as bending, kinetic energy accumulation and sudden damping.

Deployment accuracy

Together with the visual inspection, we compared the tip trajectory of the medical device when interacting with both LIM and TM models. Tip positions were stored during 2s. The direction of the tool being imposed, a theoretical trajectory was computed for the tip. In this assessment, only the capsule shape was used for its calculation simplicity. With this in mind, the tip should describe a path line \mathcal{L} when sliding along the surface. \mathcal{L} was found as follows. First, the first tip position, P_{t_0} , and the last, P_{t_N} , colliding with the surface are retrieved. Consecutive positions between P_{t_0} and P_{t_N} described the tip trajectory \mathcal{P} . It was compulsory to find the last position on the surface because the tool tip left the surface after a certain number of time steps when interacting with the TM (see Fig. 5.8). Next, the plane Φ defined by P_{t_0} and its normal \vec{n}_{P_0} – provided by the convolution surface – was calculated. \mathcal{P} was projected onto this plane. That is a view from above as depicted in Fig. 5.8 (top row) Finally, the corresponding projection \mathcal{P}^* was fed to a linear regression algorithm so that the equation of line \mathcal{L} and its associated **Root Mean Square (RMS)** error were computed.

A second assessment was centered on the distance from \mathcal{P} to the surface since the tip should slide on the surface. TaGD was used in this accuracy inspection. To this end, we provided statistical information about this distance – i.e. min, median, 90th-percentile and max values.

Computational Efficiency

Computational Efficiency was evaluated by measuring the computation time of the collision detection process using LIM. This computation time was compared with the time

Model			
LIM	N_p		329
	N_u		249
	\mathcal{L}		$\begin{pmatrix} 0.1903 \\ 0.2101 \\ 0.2323 \end{pmatrix} + \begin{pmatrix} -0.5050 \\ -0.5716 \\ -0.6466 \end{pmatrix} \cdot t$
	RMS error		$1.1 \cdot 10^{-5}$
	TaGD	min	$7.1 \cdot 10^{-6}$
median		0.006	
90th		0.019	
max		0.031	
TM	N_p		335
	N_u		191
	\mathcal{L}		$0.172 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - 0.5774 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \cdot t$
	RMS error		$1.6 \cdot 10^{-17}$
	TaGD	min	$3.5 \cdot 10^{-3}$
median		0.167	
90th		0.336	
max		0.465	

Table 5.2: Accuracy computation of the tool tip trajectory when interacting with a capsule shape modeled with LIM and TMs. N_p and N_u respectively represent the total number and the number of used tip positions. \mathcal{L} provides the line fitted to the trajectory and its corresponding **Root Mean Square (RMS)** error. Finally, statistical information – minimal, medial, 90th-percentile and maximal values – of the distance between tip positions and the surface during the slip motion.

required to detect collision with a **TM** at different level of details. The collision detection process with meshes was based on state-of-the-art techniques: broad phase spatial partitioning and **AABB** for the narrow phase (the reader may refer to [Teschner et al. \(2005\)](#) for an overview of collision detection techniques with meshes).

5.6 Results and discussion

The tip trajectory visually displayed frictionless behavior for **LIM** against the capsule shape. On the contrary, **TM** put forth jerky motions (Fig. 5.8 bottom). Table 5.2 encompasses figures quantifying the tool motion for both **LIM** and **TM** models. The trajectory with **TM** presented the the lowest **RMS** error but both remained very close to a path line. In a second stage, we accounted the distance between the tip position and the surface. It turns out that **LIM** configuration provided the best figures for **TaGD** in Table 5.2. Indeed, the tool tip described oscillatory motions when sliding on the **TM** at interactive rates whereas, simulation with **LIM** produce frictionless motions as depicted in Fig. 5.8).

Results for the computational efficiency of the collision detection process are displayed in Table 5.3 and illustrate that collision detection on **LIM** is significantly faster (about two orders of magnitude) than collision detection on triangular meshes. Even for dense data-sets, the topological locality of **LIM** allows to limit the computational

	# of primitives	Comp. time (msec)	# of contacts
Capsule (Mesh)	12k triangles	4.89	51
Capsule (LIM)	1k blobs	0.0232	50
Spiral (Mesh)	2.74k triangles	1.09	79
Spiral (Mesh)	11.3k triangles	1.84	80
Spiral (Mesh)	45k triangles	5.58	80
Spiral (LIM)	10k blobs	0.0219	80
Phantom (Mesh)	100k triangles	16.23	80
Phantom (LIM)	9.5k blobs	0.133	82

Table 5.3: Computation timings for collision detection on various deployment scenarios. For a comparable amount of primitives, collision detection on **LIM** outperforms collision detection on triangular meshes even whith the systematical use of state-of-the-art acceleration techniques. Thanks to the topological locality, the computational requirement for collision detection on **LIM** remains low even for large data-sets (e.g. more than 10k blobs).

burden of collision detection, making it suitable for interactive simulations.

Implicit modeling also offers greater accuracy for the simulation. During interventional radiology procedures, the navigation of the neurosurgical device involves touching contacts and sliding of the device against blood vessels. Contacts are usually defined as holonomic constraints between the tool and the surface. The constrained mechanical system is then solved using direct or iterative solvers. In order to be suitable for real-time simulations, two simplifications are usually performed: only a local part of the surface is considered and this local part of the surface is often linearly approximated. This local linear approximation of the surface is an infinite plane that the device must not cross. While this approach is fast and relevant for *bouncing* collisions, it does not provide accurate simulations when sliding contacts occur because the corrected motion computed by the solver, may be outside the surface (Fig. 5.4 top left). Strategies to solve that issue are first, to increase the quality of the approximation as a union of planes inducing more constraints added in the system or second, to update the local approximation of the surface by firing again the collision detection process. When dealing with **TMs**, the latter strategy is not relevant since the computation time that is needed is too high to target interactive simulations. The former strategy implies a heavier computation burden but is still compatible with interactive application [Dequidt et al. \(2009\)](#). The main drawback is that the approximation is defined as a union of planes. The used local surface constraint is thereby not smooth, leading to bumpy/friction motion when sliding on the surface. This gives the impression that the surface the device is sliding on, is not smooth. This phenomenon is displayed on Figure 5.9 (left) where a neurosurgical device was deployed in the capsule shape and contacts were frictionless. The tip of the device got stuck due to the artificial friction induced by the approximation of the contact surface. This drawback can be attenuated by increasing the mesh resolution but it significantly increases the time to perform the collision detection as well as the resolution of the system. When dealing with **LIM**, the time needed to perform collision detection is very small and therefore the second strategy (collision updates) is suitable for real-time applications. Figure 5.9 (right) illustrates the same simulation using **LIM** and collision

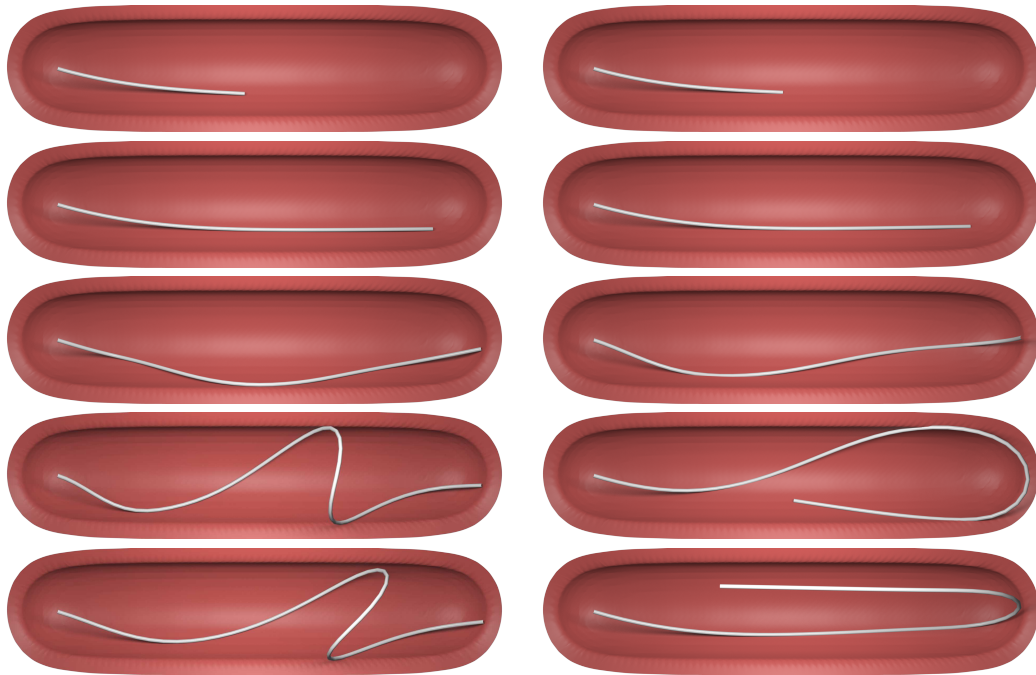


Figure 5.9: Simulation of catheter deployment inside a capsule-like shape. The collision is handled with a triangular mesh (left) and LIM (right). Even with a fine resolution mesh (12k triangles) artificial friction appears that eventually creates a loop in the catheter. LIM (12k blobs) enables the catheter to smoothly slide along the surface.

updates in the solver which produced a smooth motion without artificial friction.

Finally LIM also increases the robustness of the simulation. When using a discrete collision pipeline, collisions may be missed during two successive time-steps because simulated objects move too fast. Having a cheaper interior/exterior evaluation as well as an approximation of the geometric distance to the surface helps setting constraints to correct the simulation due to the missed collisions. While doable on triangular meshes, this approach is however costly and scarcely used. In the context of interactive simulation, the computation time should be close to **Real-Time (RT)** but the use of an iterative solver makes this constraint hard to reach. Thanks to the constraints robustness provided by the implicit modeling, a maximum number of iterations may be set for the iterative solver allowing to control the execution time. Even if the maximum number of iterations is reached and the error is not inferior to the desired threshold (possibly inducing points outside the surface), the aforementioned mechanism helps to reach a good solution (for the solver) at the next time-step. This feature is illustrated with the simulation used in figure 5.9 where when the catheter was very stiff (Young's modulus = 100 GPa), a really high number of iterations is required to ensure solver convergence when the surface was a TM (more than 200 iterations per simulation time step). Clamping the maximum number of iterations at 100, in order to have a computation time below 20 ms per simulation time step, made the simulation diverge and the catheter move outside the capsule. Using LIM and more generally, constraints that can be set with im-

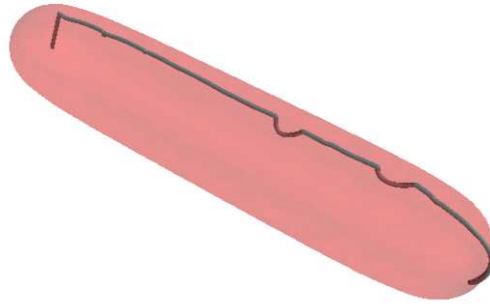


Figure 5.10: Numerical simulation at non interactive rates on a surface generated with LIM. The tip of the tool experiencing bumps at transitions between two implicit surfaces when sliding on the surface.

implicit surfaces, the same catheter remained inside and the solver was able to recover the simulation from a non-optimal solution at the previous time-step.

5.7 Conclusion

In this chapter, we described the deformation algorithms implemented in the platform SOFA within our local implicit model was tested. Within an interactive context, our proposal was compared against classical TMs. Both models were applied to situations where a medical device is interacting with them. More precisely, we assessed qualitatively and quantitatively the outcomes of these numerical simulations. First, we visually inspected the tool motion – i.e. smooth and realistic. Numerical simulations with the implicit model put forth slipping and smooth motions when the device was sliding along its surface. Conversely, outcomes on TMs evinced artificial frictions and situations where the medical device left the surface. A second assessment was carried out on the tool motion. For that, we considered the tip position during 2s when sliding along the surface of a capsule shape. Figures showed that both models produced similar outcomes when considering the trajectory of the tip. Nevertheless, the tool tip displayed jerky motions during navigation on a TM which was not the case with LIM. Finally, we evaluated computation time during the collision detection phase with both models. It turned out that the usage of LIM improved computation efficiency by two orders of magnitude when compared to collision detection with state-of-the-art BVH associated with TM.

Despite discontinuities in our model are present, no jerky motions were observed during the tests. We believe that in order to notice their impact on the tool motion, one has to employ small time steps below interactive rates. As a result, a better estimation of contacts is achieved which produces jumps at transitions between two BMs (see Fig. 5.10). Further discussion and possible solutions on this subject, are provided in the perspectives – at the end of this manuscript.

Numerical simulations on different scenarios showed the strength and adequacy of our model to interactive contexts. That is to say, realistic motions as those seen on fluoroscopy were achieved such as slide and loops. Major contributions of LIM during collision detection are cheap inclusion/exclusion evaluations, easiness of computing the distance to the surface and last but not least, the availability of a gradient direction re-

lated to the distance function gradient. This latter feature increases the robustness of the simulation, especially when objects are moving fast; and provides accurate constraints which stabilize the iterative solver and authorize to fix a maximum number of iterations for complying with an interactive context.

Part III

Conclusion

CONCLUSION AND PERSPECTIVES

*In this section, we conclude about our work and explore new perspectives. A brief summary of our main contributions reported along this manuscript is also provided. Four new paths of exploration are investigated: a multi-branch scenario for **RANSAC-Based Tracking (RBT)**, direct reconstruction from image data with **Local Implicit Modeling (LIM)**, deformations of the implicit model and reduction of discontinuities between local implicit surfaces.*

6.1 Contributions

In the context of **Interventional Neuroradiology (IN)** simulations, we addressed the problem of the availability of patient specific geometrical models for **Real-Time (RT)** or near **RT** computer-based simulations. We explored existing methods for capturing the vascular geometry with regard to an interactive simulation context. After this, we concluded that implicit surfaces fit the best a simulation background. However, methods providing such a model often rely on discrete structures for computation which may lead to local inaccuracies when considering a global reconstruction. Besides, a global description of the brain vasculature is inefficient when performing collision detection whereas only certain portions of the geometrical model are involved in this task. Furthermore, the availability of topological information is capital for correctly handling and speeding up collision detection (Li et al. (2012)) since it provides a natural decomposition of the vascular tree, specially when dealing with **Kissing Vessel (KV)** issues. Further considerations were surveyed when choosing an implicit model: compacity of the model, modeling capabilities (ability to capture complex shapes) and input data requirements. Despite the fact that methods based on implicit reconstruction from oriented scattered data present sound modeling capabilities and resilience to noise, they were discarded since the normal estimation at input data points – in our case – would be estimated from **three-Dimensional (3D) Rotational Angiography (RA)** image data. Unless reliable normals are provided, finding a consistent orientation of the normals has been recognized to be an ill-posed problem when sampling is sparse and noisy (Alliez et al. (2007)); two issues that are met in our case. Among the existing implicit models, we chose **Blobby Models (BMs)** for their compacity, locality (Dekkers et al. (2004)), and complex shape modeling capabilities (Muraki (1991); Bittar et al. (1995) and Tsingos et al. (1995)). To this point, we opted to capture locally the blood vessel to attain a precise representation and increase the computational efficiency during collision detection. For that, we relied on a two-folded framework: vessel tracking with **RANSAC-Based Tracking (RBT)** and local surface reconstruction with **Local Implicit Modeling (LIM)**.

In this work, we presented a novel tracking algorithm **RBT** which provides a local description of the blood vessel surface: cylinders – which encode the local vesselness – and a dense sampling of the vessel surface. State-of-the-art **Multiple Hypothesis Tracking (MHT)** vessel segmentation algorithm (Friman et al. (2010)) was used – as reference – for validation on 744 vessels of a 10 **3DRA** patient data-set. **MHT** is the leading algorithm in the Rotterdam Coronary Artery Algorithm Evaluations Framework¹. **RBT** algorithm was extensively evaluated on its tracking success, centerline extraction capability and its extraction accuracy **with respect to (w.r.t)** the **MHT** centerline. The outcomes showed the ability of **RBT** to precisely detect centerlines, its ability to capture very complex vascular topology and tiny tortuous blood vessels, and its robustness to handle **KVs** thanks to its **RANdom SAmples Consensus (RANSAC)**-based cylinder fitting process. As such, it improved upon **MHT** in all these areas: 89% of success rate (vs 69% for **MHT**), doubled tracked length **w.r.t** **MHT** and same accuracy as **MHT**.

Next, the segmented vascular tree with **RBT** was entrusted to our **LIM** procedure.

¹<http://coronary.bigr.nl/>

For each position on the centerline, a **BM** ensures a local description of the blood vessel surface. The resulting **BM** complies with simulation requirements. In essence, the implicit function approximates the distance function and its gradient – both features playing a major role during collision prediction, detection and handling. For providing such an implicit surface, the **BM** fitting process was expressed as an energy minimization problem. To this end, energy constraints in the literature were revisited by assessing their pertinence with regard to input data. At the end of this analysis, we proposed new formulations suited to our **BM** fitting context. Together with (Muraki (1991); Bittar et al. (1995) and Tsingos et al. (1995)), a subdivision heuristic was employed for increasing the number of blobs in the **BM** but improving upon these previous works. The novelty on this approach was to automatically choose blobs for fissioning, according to a geometric criterion. As a result, our **LIM** procedure first captures a rough estimation of input data and subsequently, captures details. **RBT** centerline was thereby enriched with **BMs**. Following a thorough evaluation on both synthetic and patient data, **BMs** were shown to precisely describe the vessel surface (sub-voxel precision). Such a scheme also provided compact representations. Moreover, **LIM** put forth smooth transitions at bifurcations and modeling capabilities at capturing tiny, tortuous vessels and aneurysms. Last but not least, **LIM** showed its strength when dealing with noise and a small amount of outliers.

Third, we applied **LIM** outcomes to an interactive context within the **Simulation Open Framework Architecture (SOFA)** platform. We aimed at assessing qualitatively and quantitatively **LIM** efficiency. To this end, we compared our model against a state-of-the-art configuration – i.e. a **Triangular Mesh (TM)** model coupled with **Axis Aligned Bounding Box (AABB)** for collision detection. On both synthetic and patient data, the catheter displayed realistic and smooth motions when interacting with **LIM**. On the other hand, jerky motions and artificial friction were noticed when sliding on **TMs**. Figures for the trajectory of the tip confirmed our observations where **LIM** provided smooth sliding meanwhile the scores of the **TM** configuration evinced oscillations induced in the catheter tip. Special features of the **LIM** implicit function conferred to our geometrical model an overwhelming computation efficiency dividing by two orders of magnitude the computation time for collision management when compared to **AABB**. Incidentally, it turned out that **LIM** induced robustness during the simulation process – especially when objects were moving fast – and stabilization in the iterative solver through precise constraints. All these features of **LIM** make it suited to interactive numerical simulations.

6.2 Perspectives

In this work, we presented an efficient framework for modeling the arterial network with application to interactive numerical simulations. Despite good behavior on several situations, we recognize that **RBT** and **LIM** can be further polished. A path to walk for improving our framework points toward four different directions: the automatic bifurcation handling of **RBT**, the direct reconstruction from image data, the integration of deformations of our geometrical model and the inherent problem of discontinuities be-

tween implicit surfaces.

Bifurcation handling

We recognize one weak point to our **RBT** procedure. It is true that tracking relies on manual interaction, while solid works on this area exist which handle bifurcations automatically such as [La Cruz et al. \(2004\)](#); [Tyrrell et al. \(2007\)](#); [Wong and Chung \(2007\)](#) and [Friman et al. \(2010\)](#). Nonetheless, the tracking with **RBT** is very fast, and physicians may find it more user-friendly to manually select the vessels of interest. This selection of vessels of interest is especially true in the context of **IN**; where neuroradiologists work on a specific part of the arterial network – that is, around the pathology. As a result, we believe that two modes should be available for **RBT**: single branch and multi-branch tracking.

In the case of multi-branch tracking, **RBT** needs to use an automatic process for detecting bifurcations. In the current state of our tracking procedure, this detection is hard to be achieved due to the fact that **RANSAC** is a single model detector. It means that in the presence of point-sets depicting two vessels, one single cylinder is retrieved. In the research field of shape retrieval from range data, one possible approach for fitting more than one model to data, is to sequentially use **RANSAC** ([Chaperon and Goulette \(2001\)](#); [Kanazawa and Kawakami \(2004\)](#); [Rabbani and Heuvel \(2005\)](#); [Schnabel et al. \(2007\)](#) and [Gallup et al. \(2010\)](#)): first, find one model fitting the point-set, then delete this points from the data-set and run again the fitting process. For implementations like in [Kanazawa and Kawakami \(2004\)](#) and [Gallup et al. \(2010\)](#), the number models must be known in advance. Nevertheless, experiments with our input data revealed that this heuristic fails at correctly fitting a second model. The reason for this failure is that point-sets extracted in the ray-casting stage are not dense. Increasing the number of rays does not enhance initial conditions for this method.

Parallel implementations of multi-model **RANSAC** exist ([Zuliani et al. \(2005\)](#)). Recently, J-linkage clustering ([Toldo and Fusiello \(2008\)](#)) was introduced as a method for fitting multiple instances of a model to data corrupted by noise and outliers. The algorithm is based on random sampling – as **RANSAC** – and a clustering method. First, the input data points are represented with feature vectors that indicate the set of random models consistent with every point. In this feature space, J-linkage clustering is used to group the points belonging to the same model. Results have shown better performance than **RANSAC** or Hough-based approaches.

In conclusion, **RBT** fitting process should evolve towards multi-model fitting for handling multi-branch trackings. High expectations for J-Linkage in this process are put.

Reconstruction from image data

We reckon that **LIM** relies heavily on point-sets and consequently, **LIM** reconstruction is dependent on the method precision and robustness of the point extraction method. A desired evolution for **LIM**, is the appealing idea that the local implicit surface fitting be made with regard to image data.

Two different scenarios may be conceivable. First, the resulting implicit function is fine tuned statically – i.e. over its widths without moving centers. In a second situation, **BMs** are authorized to change their blobs positions and widths, –i.e. a dynamic fitting.

In a static fitting case, [Gelas et al. \(2007\)](#) used compactly supported **Radial Basis Functions (RBFs)** distributed on a regular grid and coupled to a level-set formulation for medical segmentation. The **RBF** was employed for solving a differential equation expressing the level set evolution. By the same token, [Xie and Mirmehdi \(2011\)](#) used a level-set approach grounded on **RBFs** for segmentation but extended to **3D**. In a dynamic scenario, [Morse et al. \(2005\)](#) segmented the left-ventricular chamber of the heart with snakes. In this work, the contour was evenly discretized by a set of charged particles which in turn, where the centers for a **RBF**. The particles positions were advected through image-based and smoothing energies. This method also accounted for a user defined force to drive the segmentation process.

The work introduced in [Morse et al. \(2005\)](#) is the closest to our scenario for both static and dynamic fitting. In essence, the **BM** should be able to move *w.r.t* image forces. However, the main difference between our implicit formulation and that supplied for **RBFs** is the center placement. In our case, there is no direct formulation linking deformation at the surface with the **BM** parameters; meanwhile **RBFs** centers are given by the particles position in Morse’s method.

Computational Fluid Dynamics (CFD) community brings about a possible solution, namely particle level-set methods ([Foster and Fedkiw \(2001\)](#); [Enright et al. \(2002\)](#) and [Mihalef et al. \(2007\)](#)). These methods combine a level set function with a particle system for providing a dual representation of a surface. During evolution of the level set function, the particle system is employed as a sampling of the zero level set. Moreover, particles provide information about the motion characteristics of this surface. Besides, both descriptions update each other during evolution computation which leads to a precise and robust computation of the evolutionary equations. The idea of using particle systems to sample surfaces is not novel. [Szeliski et al. \(1993\)](#) introduced them for sampling triangulated surfaces and [Witkin and Heckbert \(1994\)](#) provided a sound particle system for implicit surfaces. There is a vast literature pursuing Witkin’s seminal work ([Levet et al. \(2006\)](#); [Nesme and Bouthors \(2006\)](#) and [Meyer et al. \(2007\)](#)).

To get back to **LIM**, one can use Witkin’s approach for sampling a **BM** in order to provide a better estimation of energy values, in particular the mean square curvature \mathcal{E}_κ , and use the particle system to control the **BM**. Indeed, Witkin provided formulae linking the implicit surface parameters and the particle system. With this in mind, one can think about a framework computing internal and external energy values and accordingly, deform the **BM** so that these energies are minimized.

This procedure may also alleviate a recognized weakness of our framework, the modeling of pathologies. At the current state, the user is requested to place seeds roughly at the medial axis of pathologies so that topology is retrieved from these seeds, as well as, points at the vessel surface. In the case of **LIM** working directly on the image, blob positions may directly inform about the pathology medial axis which can be estimated by applying a gradient descent strategy to particle positions along the implicit function gradient ([Ma \(2007\)](#)).

Geometrical model deformation

In Chapter 5, we applied a surface produced with LIM to an interactive simulation. In this context, the geometrical model was considered as rigid. Two different paths may be followed for taking into account deformations. Spatially and physically based deformations. The former technique deforms the whole space in which the objects are embedded (Jin et al. (2000) and Yoon and Kim (2006)). The latter produces very realistic deformations of objects by solving physically-grounded differential equations (Cani and Desbrun (1997) and Amrani et al. (2001)).

In our context of interactive simulation, deformations of the scalar field function may hinder LIM dedicated features for simulation. Indeed, this kind of techniques manipulates the implicit function scalar field with no guarantee of at least a C^1 -continuity (Jin et al. (2000)).

Physically-based deformations are preferable given our application field. Cani and Desbrun (1997) introduced a double representation technique for animating deformable implicit surfaces. For instance, an object provided as a mesh was coated with an implicit surface which was used during collision detection and contact response computation. Same as for particle level-set methods, the implicit surface was evenly sampled for computing collisions and numerical integration of forces. Once the contact response was computed with the implicit surface, the object was accordingly adapted. Visualization was achieved by meshing the particle positions. By the same token, Amrani et al. (2001) used a skeletal representation of implicits coupled to a particle system evolving on its zero level set. During simulation, particle positions are tracked by a mesh; and skeletal points are linked together by a mass-spring system. The animation process is decomposed in four steps: rigid motion of objects, collision detection, deformation of the surface in contact and accordingly, deformation of the skeleton.

Once again, particle systems seem to be an asset when dealing with deformations. Indeed, reconstruction from image data can also be seen as deforming a BM so that it fulfills minimal energy criteria. Let's consider the usage of particle systems within LIM. Generally speaking, deformations may be achieved through: first, detect interpenetration and particles involved; reaction forces are next applied to colliding particles; and finally, modify blobs properties according to particles motion.

During interactive simulation of deformable objects, visualization becomes crucial and cumbersome since objects need to be updated at near RT rates. Recent advances in point-based and Graphics Processing Unit (GPU) based techniques have proven to be a valuable alternative to polygonal meshes, especially when dynamic models are to be processed (Kobbelt and Botsch (2004)).

Discontinuities

Due to our local approach, discontinuities between local implicit surfaces are inevitable. We believe that image data reconstruction may lessen this issue but not efface it. Our beliefs are grounded on the fact that for a node on the centerline, LIM concatenates local data points to a certain topological distance. As a result, some data points are shared with neighboring nodes which happens to induce small differences in the resulting BM

within this common region. With this in mind, an ideal solution would be to locally re-fit/adjust **BMs** within a common portion. To this end, one can use the radical planes (Weighted Voronoi diagram) between consecutive nodes for defining this zone. Within it, blobs – of **BMs** participating to its modeling – are to be optimized and the other blobs remain fixed. An optimization over the widths may be sufficient to further lessen discontinuities. Along the same outline, one can use a particle system on the surface, bounded to this region. Subsequently, it is possible to evaluate the distance between surfaces in this portion and minimize it according to a geometrical or image-based criterion. Note that both procedures may also work at bifurcations.



APPENDIX A

Herein, we present our dedicated **Graphical User Interface (GUI)** for tracking with **RANSAC-Based Tracking (RBT)**, the so-called **RANSAC-based Tracking Interface (RTrackingIT)**. This GUI is composed of four parts: the mayavi window, the menu bar, the configuration and the labeling sections (see Fig. A.1).

A.1 RANSAC-based Tracking InTerface (RTrackingIT)

RTrackingIT is based on the open library **PyQt**¹ which provides a vast number of widgets for rapid prototyping. Furthermore, this software also relies on **Mayavi2**² which is a

¹<http://www.riverbankcomputing.co.uk/software/pyqt/intro>

²<http://docs.enthought.com/mayavi/mayavi/index.html#>

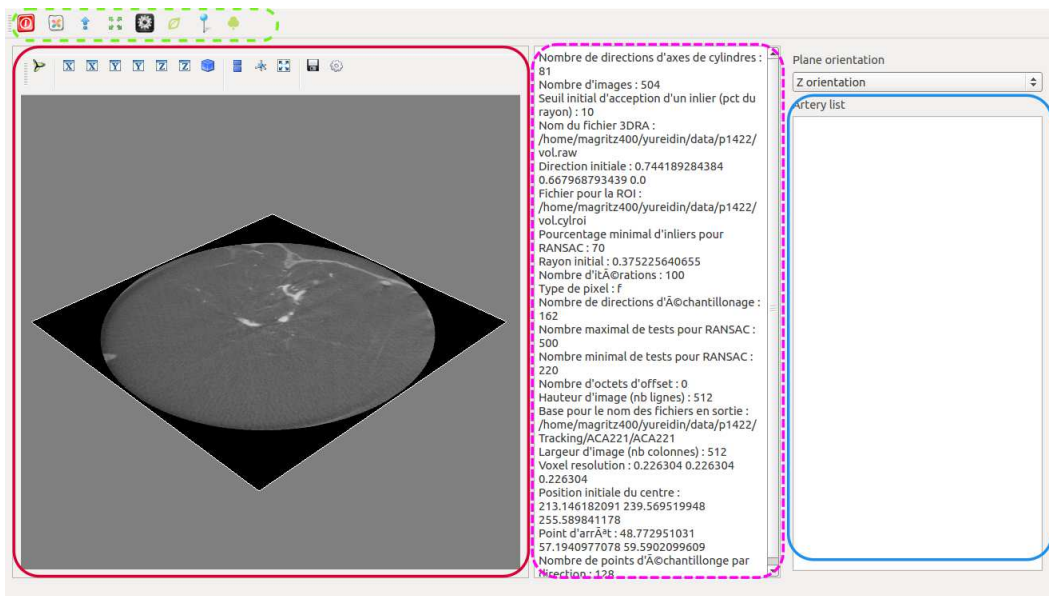


Figure A.1: The overall window of **RANSAC-based Tracking Interface (RTrackingIT)** GUI. Four sections compose **RTrackingIT**: the menu bar (top), the mayavi widget (left), the configuration (middle) and the labeling sections (right).

visualization software – entirely written in Python. The VTK library, originally written in C++, is entirely available under Mayavi2 which delivers ready-to-use filters in a few clicks. With this in mind, we first introduce the mayavi window.








A.1.1 Mayavi widget

Mayavi serves to visually inspect volumetric images and to place seed points for the tracking procedure. When clicking with the mouse wheel on the volume, a red axis appears, indicating its coordinate. This picking process helps **RTrackingIT** to define the position, radius and direction of the tracking seed. Furthermore, The versatility of Mayavi authorizes to use home-made python scripts. Generally speaking, operations executed with a mouse can be automatize by importing the python package of Mayavi.

For instance, run **RTrackingIT** and click on . A new window appears, right click over the first entry in the scene. Select **Filters**→**AddFilter**→**IsoSurface**. An iso-surface of the volumetric data is automatically created.


A.1.2 Menu bar

The menu bar is composed of several action buttons:

- exit **RTrackingIT** ;
- place the position of the seed , one single click on the volume is necessary;
- provide the direction of the seed , two points on the volume are required;
- set the radius of the seed ; this is provided by the Euclidean distance between two positions on the volumetric data;
- add a new leaf on the tree . Tracking is automatically run after adding the label of the leaf on the **Labeling section**. The configuration file loaded in the **Configuration section** is used for **RBT**;
- add an instance of tracking without adding to the tree . Similar to add a new leaf;
- provide the tree topology . It writes on the configuration section the labels of each tracking instance as: `Parent Children1 Children2 ...`

Note that all actions are also available on the windows menu under the names **File**, **Action** and **Tracking**.

A.1.3 Configuration section

The configuration section shows the text file with configuration arguments of parameters. **RTrackingIT** stores internally this configuration file which can be loaded by clicking on . In this section, all **RBT** parameters can be changed. One can manually modify this configuration file by directly writing in this section and then, save modifications to the selected configuration file. Otherwise, one can let **RTrackingIT** do it automatically.

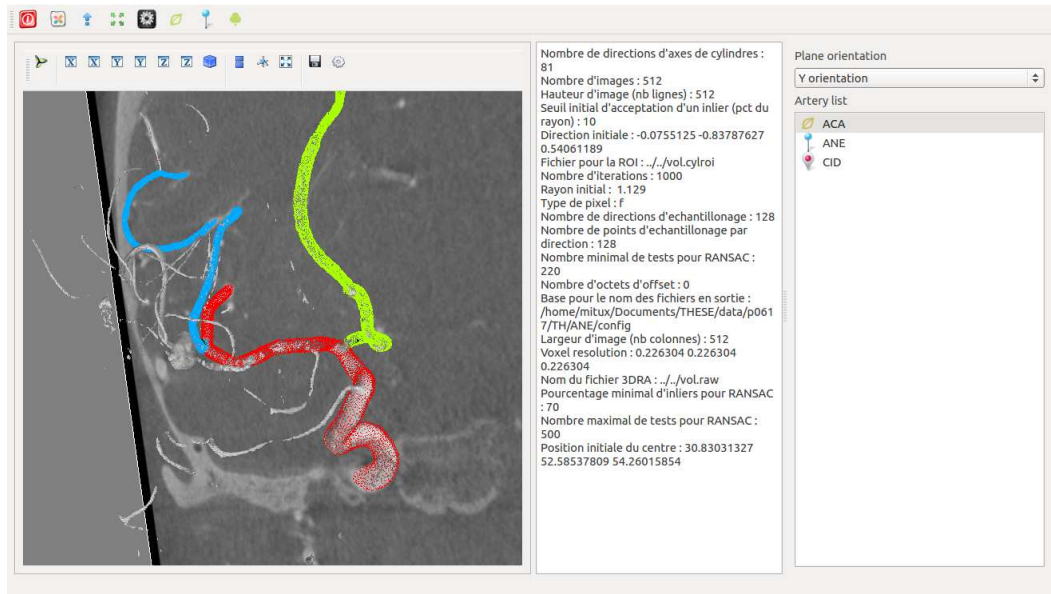





Figure A.2: RTrackingIT in action. Three instances of tracking with RBT combined with a triangulated mesh produced with Mayavi Isosurface filter. In the labeling section, vessels CID (red), ACA (cyan) ANE (lime) are respectively root, leaf and trial instances.

Indeed, every time an action on the menu bar is executed, it updates the values for a tracking instance.

A.1.4 Labeling section

Herein, a list of all tracked vessels is provided. A vessel instance can be root  and leaf  in the tree. Another class of vessel exists which was especially created for trials – i.e. those prefixed by . For depiction, Fig. A.2 illustrates this three types of tracking instances.

Right click on one label pop-ups filters for loading (View...), hiding (Hide...), showing (Show...), changing color (Color...) and deleting (Clear...) the blood vessel dense sampling (Glyphs), centerline (Medial Axis) and tracking seed (Seed).



APPENDIX B

In this appendix, the partial derivatives of **Blobby Models (BMs)** and energies \mathcal{E}_d , \mathcal{E}_c and \mathcal{E}_κ **with respect to (w.r.t)** changes in width (ρ_j) and positions C_j are derived. This derivatives are required in the fitting process described in Chapter 4.

Before starting to drive equations. We introduce some notations for the sake of clarity:

- 1st-order tensors are underlined once – i.e \underline{t} – and 2nd-order tensors are underlined twice – i.e. $\underline{\underline{v}}$. For instance, the null vector is denoted as $\underline{\underline{0}}$ and the identity matrix is denoted by $\underline{\underline{1}}$.
- scalars are zero-order tensors and not underlined $\mathbf{0}$.
- Einstein's summation convention is used.

B.1 Fomulae for a blob

A blob B – of center \underline{C} and width ρ – is defined at point P as:

$$B(\underline{P}) = \rho\phi(|\underline{\delta}|^2) \quad (\text{B.1})$$

where ϕ is the kernel function – considered as C^∞ – and $\delta = \frac{P-C}{\rho}$. Derivatives **w.r.t** scalar ρ and vectors \underline{C} and \underline{P} are to be calculated. For the sake of brevity, we use $B = B(\underline{P})$, $\phi = \phi(|\underline{\delta}|^2)$ and their corresponding derivatives. $\underline{\underline{H}}$ denote the hessian matrix of B .

Let's derive useful derivatives before presenting results for Eq. B.1.

- $\underline{\delta}$

$$\partial_\rho \underline{\delta} = -\frac{1}{\rho} \underline{\delta} \quad (\text{B.2}) \quad \partial_{\underline{C}} \underline{\delta} = -\frac{1}{\rho} \underline{\underline{1}} \quad (\text{B.3}) \quad \partial_{\underline{P}} \underline{\delta} = \frac{1}{\rho} \underline{\underline{1}} \quad (\text{B.4})$$

- $|\underline{\delta}|^2$ $\partial_\rho |\underline{\delta}|^2 = -\frac{2}{\rho} |\underline{\delta}|^2$ (B.5) $\partial_{\underline{C}} |\underline{\delta}|^2 = -\frac{2}{\rho} \underline{\delta}^2$ (B.6) $\partial_{\underline{P}} |\underline{\delta}|^2 = \frac{2}{\rho} \underline{\delta}^2$ (B.7)

- ϕ

$$\partial_\rho \phi = \phi' \partial_\rho |\underline{\delta}|^2 = -\frac{2}{\rho} \phi' |\underline{\delta}|^2 \quad (\text{B.8}) \quad \partial_{\underline{C}} \phi = -\frac{2}{\rho} \phi' \underline{\delta} \quad (\text{B.9}) \quad \partial_{\underline{P}} \phi = \frac{2}{\rho} \phi' \underline{\delta} \quad (\text{B.10})$$

- $B = \rho\phi$

$$\partial_\rho B = \phi + \rho\partial_\rho\phi = \phi - 2\phi'|\underline{\delta}|\partial_{\underline{C}}B = \rho\partial_{\underline{C}}\phi = -2\phi'\underline{\delta} \quad \partial_{\underline{P}}B = \rho\partial_{\underline{P}}\phi = 2\phi'\underline{\delta} \quad (\text{B.11}) \quad (\text{B.12}) \quad (\text{B.13})$$

- Hessian matrix

$$\underline{\underline{H}} = \partial_{\underline{P}}\partial_{\underline{P}}B = 2\partial_{\underline{P}}(\phi'\underline{\delta})$$

Note that

$$\partial_{\underline{P}}(\phi'\underline{\delta}) = \underline{\delta}(\partial_{\underline{P}}\phi')^t + \phi'\partial_{\underline{P}}\underline{\delta}$$

where

$$\partial_{\underline{P}}\phi' = \frac{2}{\rho}\phi''\underline{\delta}$$

then we have

$$\underline{\underline{H}} = \frac{2}{\rho}\left(2\phi''\underline{\delta}\underline{\delta}^t + \phi'\underline{\underline{1}}\right) \quad (\text{B.14})$$

From this, we can deduce that

$$\text{tr}(\underline{\underline{H}}) = \frac{2}{\rho}(2\phi''|\underline{\delta}|^2 + 3\phi')$$

We define matrix $\underline{\underline{\gamma}} = \underline{\underline{H}} - \text{tr}(\underline{\underline{H}})\underline{\underline{1}}$, further employed

$$\underline{\underline{\gamma}} = \underline{\underline{H}} - \text{tr}(\underline{\underline{H}})\underline{\underline{1}} = \frac{4}{\rho}\left[\phi''(\underline{\delta}\underline{\delta}^t - |\underline{\delta}|^2\underline{\underline{1}}) - \phi'\underline{\underline{1}}\right] \quad (\text{B.15})$$

B.2 Formulae for blobby models

A **BM** is a sum of N_b blobs. We use subscripts for indicating blobs in the **BM**: blob $\#i \in \{1, \dots, N_b\}$ is defined by its center \underline{C}_i and its width ρ_i . The following notations are used for point \underline{P} :

$$\bullet \underline{\delta}_i = \frac{\underline{P} - \underline{C}_i}{\rho_i} \quad \bullet \phi_i = \phi(|\underline{\delta}_i|^2) \quad \bullet B_i = \rho_i\phi_i$$

Note that $j \neq i$:

$$\partial_{\rho_j}\underline{\delta}_i = \underline{\underline{0}} \quad (\text{B.16}) \quad \partial_{\underline{C}_j}\underline{\delta}_i = \underline{\underline{0}} \quad (\text{B.17})$$

B.2.1 Implicit function, gradient, Hessian matrix

The implicit function for a **BM** is defined as:

$$\mathcal{B} = \sum_{i=1}^{N_b} B_i \quad \mathcal{B} = \sum_{i=1}^{N_b} \rho_i\phi_i \quad (\text{B.18})$$

whose gradient is provided by (B.13):

$$\underline{\underline{\mathcal{G}}} = \sum_{i=1}^{N_b} \partial_{\underline{P}}B_i \quad \underline{\underline{\mathcal{G}}} = 2 \sum_{i=1}^{N_b} \phi'_i\underline{\delta}_i \quad (\text{B.19})$$

Using (B.14), its Hessian matrix is:

$$\begin{aligned}\underline{\underline{\mathcal{H}}} &= \sum_{i=1}^{N_b} \partial_{\underline{P}} \partial_{\underline{P}} B_i \\ \underline{\underline{\mathcal{H}}} &= 2 \sum_{i=1}^{N_b} \frac{1}{\rho_i} \left(2\phi_i'' \delta_i \delta_i^t + \phi_i' \underline{\underline{1}} \right)\end{aligned}\quad (\text{B.20})$$

and using (B.15) with $\underline{\underline{\Gamma}} = \underline{\underline{\mathcal{H}}} - \text{tr}(\underline{\underline{\mathcal{H}}}) \underline{\underline{1}}$, one obtains:

$$\underline{\underline{\Gamma}} = 4 \sum_{i=1}^{N_b} \frac{1}{\rho_i} \left[\phi_i'' \left(\delta_i \delta_i^t - |\delta_i|^2 \underline{\underline{1}} \right) - \phi_i' \underline{\underline{1}} \right]\quad (\text{B.21})$$

B.2.2 Energies and their derivatives

We use three energy terms:

- the data attachment term (T is the iso value, and $\{\underline{P}_k\}_{k \in \{1, \dots, N_p\}}$ are the N_p points to fit)

$$\mathcal{E}_d = \frac{1}{N_p} \sum_k (T - \mathcal{B}(\underline{P}_k))^2 \quad (\text{B.22})$$

For the sake of brevity, we denote by a semicolon (;), a quantity evaluated at point \underline{P}_k . For instance, we have:

$$\underline{\delta}_{n;k} = \frac{\underline{P}_k - \underline{C}_n}{\rho_n} \quad (\text{B.23}) \quad \mathcal{B}_{;k} = \mathcal{B}(\underline{P}_k) \quad (\text{B.24})$$

Therefore, we write

$$\mathcal{E}_d = \frac{1}{N_p} \sum_k (T - \mathcal{B}_{;k})^2 \quad (\text{B.25})$$

\mathcal{E}_d has the following derivatives:

$$\partial_{\rho_n} \mathcal{E}_d = -\frac{2}{N_p} \sum_{k=1}^{N_p} [T - \mathcal{B}_{;k}] \left(\phi_{n;k} - 2\phi'_{n;k} |\underline{\delta}_{n;k}|^2 \right) \quad (\text{B.26})$$

$$\partial_{\underline{C}_n} \mathcal{E}_d = \frac{4}{N_p} \sum_{k=1}^{N_p} [T - \mathcal{B}_{;k}] \phi'_{n;k} \underline{\delta}_{n;k} \quad (\text{B.27})$$

- the Lennard-Jones energy:

$$\mathcal{E}_{LJ} = \left(\frac{r_m}{r} \right)^{12} - 2 \left(\frac{r_m}{r} \right)^6 = \left(\frac{r_m^2}{r^2} \right)^3 \left[\left(\frac{r_m^2}{r^2} \right)^3 - 2 \right] \quad (\text{B.28})$$

where r is the distance between two blob centers. In our case, we set $r_m^2 = \rho_i \rho_j$ and we introduce a scaling factor s . As a result, we have:

$$\mathcal{E}_{LJ} = \frac{1}{N_b(N_b-1)} \sum_{i \neq j} \left(\frac{s^2 \rho_i \rho_j}{|\underline{C}_i - \underline{C}_j|^2} \right)^3 \left[\left(\frac{s^2 \rho_i \rho_j}{|\underline{C}_i - \underline{C}_j|^2} \right)^3 - 2 \right] \quad (\text{B.29})$$

We denote $\sigma_{ij} = \frac{s^2 \rho_i \rho_j}{|\underline{C}_i - \underline{C}_j|^2}$, and we obtain :

$$\mathcal{E}_{LJ} = \frac{1}{N_b(N_b-1)} \sum_{i \neq j} \sigma_{ij}^3 \left(\sigma_{ij}^3 - 2 \right) \quad (\text{B.30})$$

\mathcal{E}_{LJ} has the following derivatives:

$$\partial_{\rho_n} \mathcal{E}_c = \frac{12}{N_b(N_b-1)} \frac{1}{\rho_n} \sum_{\substack{i=1 \\ i \neq n}}^{N_b} \sigma_{ni}^3 (\sigma_{ni}^3 - 1) \quad (\text{B.31})$$

$$\partial_{\underline{C}_n} \mathcal{E}_c = -\frac{24}{N_b(N_b-1)} \sum_{\substack{i=1 \\ i \neq n}}^{N_b} \frac{\sigma_{ni}^2}{s^2 \rho_n \rho_i} (\sigma_{ni}^3 - 1) (\underline{C}_n - \underline{C}_i) \quad (\text{B.32})$$

- the mean square curvature:

$$\mathcal{E}_\kappa = \frac{1}{N_p} \sum_{k=1}^{N_p} \kappa_{;k}^2 \quad (\text{B.33})$$

with derivatives:

$$\begin{aligned} \partial_{\rho_n} \mathcal{E}_\kappa &= \frac{2}{N_p} \sum_{k=1}^{N_p} \frac{\kappa_{;k}}{|\underline{\mathcal{G}}_{;k}|^3} \left[-\frac{4}{\rho_n} \left(2\phi''_{n;k} |\underline{\delta}_{n;k}|^2 + \phi'_{n;k} \right) \underline{\mathcal{G}}_{;k}^t \underline{\Gamma}_{;k} \underline{\delta}_{n;k} - \frac{1}{\rho_n} |\underline{\mathcal{G}}_{;k}|^3 \kappa_{;k} \right. \\ &\quad - \frac{4}{\rho_n^2} \left(\phi'''_{n;k} |\underline{\delta}_{n;k}|^2 + \phi''_{n;k} \right) \left(\underline{\mathcal{G}}_{;k}^t \underline{\delta}_{n;k} \right)^2 \\ &\quad + \frac{4}{\rho_n^2} \left(\phi'''_{n;k} |\underline{\delta}_{n;k}|^2 + 2\phi''_{n;k} \right) |\underline{\delta}_{n;k}|^2 |\underline{\mathcal{G}}_{;k}|^2 \\ &\quad \left. + \frac{6}{\rho_n} \kappa_{;k} |\underline{\mathcal{G}}_{;k}| \left(2\phi''_{n;k} |\underline{\delta}_{n;k}|^2 + \phi'_{n;k} \right) \left(\underline{\mathcal{G}}_{;k}^t \underline{\delta}_{n;k} \right) \right] \quad (\text{B.34}) \end{aligned}$$

$$\begin{aligned} \partial_{\underline{C}_n} \mathcal{E}_a &= \frac{2}{N_p} \sum_{k=1}^{N_p} \frac{\kappa_{;k}}{|\underline{\mathcal{G}}_{;k}|^3} \left\{ -\frac{8}{\rho_n} \phi''_{n;k} \left(\underline{\mathcal{G}}_{;n}^t \underline{\Gamma}_{;k} \underline{\delta}_{n;k} \right) \underline{\delta}_{n;k} - \frac{4}{\rho_n} \underline{\Gamma}_{;k} \underline{\mathcal{G}}_{;n} \right. \\ &\quad - \frac{8}{\rho_n^2} \left[\phi'''_{n;k} \left(\underline{\mathcal{G}}_{;k}^t \underline{\delta}_{n;k} \right)^2 - \left(\phi'''_{n;k} |\underline{\delta}_{n;k}|^2 + 2\phi''_{n;k} \right) |\underline{\mathcal{G}}_{;k}|^2 \right] \underline{\delta}_{n;k} \\ &\quad - \frac{8}{\rho_n^2} \left(\underline{\mathcal{G}}_{;k}^t \underline{\delta}_{n;k} \right) \underline{\mathcal{G}}_{;k} \\ &\quad \left. + \frac{6}{\rho_n} \kappa_{;k} |\underline{\mathcal{G}}_{;k}| \left[2\phi''_{n;k} \left(\underline{\mathcal{G}}_{;k}^t \underline{\delta}_{n;k} \right) \underline{\delta}_{n;k} + \phi'_{n;k} \underline{\mathcal{G}}_{;k} \right] \right\} \quad (\text{B.35}) \end{aligned}$$

with

$$\underline{\delta}_{n;k} = \frac{P_k - \underline{C}_n}{\rho_n} \quad (\text{B.36})$$

$$\phi_{n;k} = \phi(|\underline{\delta}_{n;k}|^2) \quad (\text{B.37})$$

$$\phi'_{n;k} = \phi'(|\underline{\delta}_{n;k}|^2) \quad (\text{B.38})$$

$$\phi''_{n;k} = \phi''(|\underline{\delta}_{n;k}|^2) \quad (\text{B.39})$$

$$\phi'''_{n;k} = \phi'''(|\underline{\delta}_{n;k}|^2) \quad (\text{B.40})$$

$$\underline{\mathcal{G}}_{;k} = 2 \sum_{i=1}^{N_b} \phi'_{i;k} \underline{\delta}_{i;k} \quad (\text{B.41})$$

$$\underline{\Gamma}_{;k} = \sum_{i=1}^{N_b} \frac{4}{\rho_i} \left[\phi''_{i;k} (\underline{\delta}_{i;k} \underline{\delta}_{i;k}^t - |\underline{\delta}_{i;k}|^2 \underline{\underline{1}}) - \phi'_{i;k} \underline{\underline{1}} \right] \quad (\text{B.42})$$

$$\underline{\mathcal{C}}_{;k} = \frac{\underline{\mathcal{G}}_{;k}^t \underline{\Gamma}_{;k} \underline{\mathcal{G}}_{;k}}{|\underline{\mathcal{G}}_{;k}|^3} \quad (\text{B.43})$$

REFERENCES

- [Alhonnoro et al., 2010] T. Alhonnoro, M. Pollari, M. Lilja, R. Flanagan, B. Kainz, J. Muehl, U. Mayrhauser, H. Portugaller, P. Stiegler and K. Tscheliessnigg. *Vessel Segmentation for Ablation Treatment Planning and Simulation*. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2010*, vol. 6361 of *Lecture Notes in Computer Science*, edited by T. Jiang, N. Navab, J. Pluim and M. Viergever, pages 45–52, Springer Berlin Heidelberg, 2010, ISBN 978-3-642-15704-2.
URL http://dx.doi.org/10.1007/978-3-642-15705-9_6
- [Alliez et al., 2007] P. Alliez, D. Cohen-Steiner, Y. Tong and M. Desbrun. *Voronoi-based variational reconstruction of unoriented point sets*. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, SGP '07, pages 39–48, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007, ISBN 978-3-905673-46-3.
URL <http://dl.acm.org/citation.cfm?id=1281991.1281997>
- [Aloisio et al., 2004] G. Aloisio, L. Barone, M. Bergamasco, C. A. Avizzano, L. T. De Paolis, M. Franceschini, A. Mongelli, G. Pantile, L. Provenzano, M. Raspolli et al.. *Computer-based simulator for catheter insertion training*. *Studies in health technology and informatics*, vol. 98, page 4, 2004.
- [Aloisio et al., 2006] G. Aloisio, L. T. De Paolis and L. Provenzano. *A training simulator for the angioplasty intervention with a web portal for the virtual environment searching*. In *Proceedings of the 5th WSEAS International Conference on Signal Processing, Robotics and Automation*, ISPR'06, pages 135–140, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 2006, ISBN 960-8457-41-6.
URL <http://dl.acm.org/citation.cfm?id=1365904.1365928>
- [Alvarez, 1996] L. Alvarez. *Images and PDE's*. In *ICAOS '96*, vol. 219 of *Lecture Notes in Control and Information Sciences*, edited by M.-O. Berger, R. Deriche, I. Herlin, J. Jaffré and J.-M. Morel, pages 3–14, Springer Berlin Heidelberg, 1996, ISBN 978-3-540-76076-4.
URL http://dx.doi.org/10.1007/3-540-76076-8_112
- [Amrani et al., 2001] M. Amrani, F. Jaillet, M. Melkemi and B. Shariat. *Simulation of Deformable Organs with a Hybrid Approach*. 2001.
- [Anderson and Raghavan, 1998] J. Anderson and R. Raghavan. *A vascular catheterization simulator for training and treatment planning*. *Journal of Digital Imaging*, vol. 11, no. 1, pages 120–123, 1998, ISSN 0897-1889.
URL <http://dx.doi.org/10.1007/BF03168278>
- [Anderson et al., 2002] J. Anderson, C.-K. Chui, Y. Cai, Y. Wang, Z. Li, X. Ma, W. Nowinski, M. Solaiyappan, K. Murphy, P. Gailloud et al.. *Virtual reality training in interventional radiology: the Johns Hopkins and Kent Ridge Digital Laboratory experience*. In *Seminars in Interventional Radiology*, vol. 19, pages 179–186, 2002.

- [Angelini et al., 2005] E. Angelini, Y. Jin and A. Laine. *State of the Art of Level Set Methods in Segmentation and Registration of Medical Imaging Modalities*. In *Handbook of Biomedical Image Analysis*, edited by J. Suri, D. Wilson and S. Laxminarayan, Topics in Biomedical Engineering International Book Series, pages 47–101, Springer US, 2005, ISBN 978-0-306-48607-4.
URL http://dx.doi.org/10.1007/0-306-48608-3_2
- [Anxionnat et al., 2001] R. Anxionnat, S. Bracard, X. Ducrocq, Y. Troussel, L. Launay, E. Kerrien, M. Braun, R. Vaillant, F. Scomazzoni, A. Lebedinsky and L. Picard. *Intracranial Aneurysms: Clinical Value of 3D Digital Subtraction Angiography in the Therapeutic Decision and Endovascular Treatment*. *Radiology*, vol. 218, no. 3, pages 799–808, 2001, pMID: 11230659.
URL <http://pubs.rsna.org/doi/abs/10.1148/radiology.218.3.r01mr09799>
- [Avril et al., 2009] Q. Avril, V. Gouranton and B. Arnaldi. *New trends in collision detection performance*. In *VRIC'09 Proceedings*, vol. 11, edited by S. R. . A. Shirai, page 53, Laval, France, 2009.
URL <http://hal.archives-ouvertes.fr/hal-00412870>
- [Aylward and Bullitt, 2002] S. Aylward and E. Bullitt. *Initialization, noise, singularities, and scale in height ridge traversal for tubular object centerline extraction*. *IEEE Trans. Med. Imaging*, vol. 21, no. 2, pages 61–75, 2002, ISSN 0278-0062.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.8435>
- [Bauer et al., 2010] C. Bauer, T. Pock, E. Sorantin, H. Bischof and R. Beichel. *Segmentation of interwoven 3d tubular tree structures utilizing shape priors and graph cuts*. *Medical Image Analysis*, vol. 14, pages 172–184, 2010.
- [Bernhardt et al., 2010] A. Bernhardt, L. Barthe, M.-P. Cani et al.. *Implicit Blending Revisited*. *Comput. Graph. Forum*, vol. 29, no. 2, pages 367–375, 2010.
URL <http://hal.inria.fr/inria-00450722>
- [Bittar et al., 1995] E. Bittar, N. Tsingos and M.-P. Cani. *Automatic Reconstruction of Unstructured 3D data : Combining Medial Axis and Implicit Surfaces*. In *Eurographics*, vol. 14 of 3, edited by F. Post and M. Göbel, pages 457–468, Blackwell Publishers, Maastricht, Pays-Bas, 1995, published under the name Marie-Paule Gascuel.
URL <http://hal.inria.fr/inria-00537543>
- [Blinn, 1982] J. F. Blinn. *A Generalization of Algebraic Surface Drawing*. *ACM Trans. Graph.*, vol. 1, no. 3, pages 235–256, 1982, ISSN 0730-0301.
URL <http://doi.acm.org/10.1145/357306.357310>
- [Bloomenthal, 1985] J. Bloomenthal. *Modeling the mighty maple*. *SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pages 305–311, 1985, ISSN 0097-8930.
URL <http://doi.acm.org/10.1145/325165.325249>
- [Bloomenthal and Shoemake, 1991] J. Bloomenthal and K. Shoemake. *Convolution surfaces*. *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pages 251–256, 1991, ISSN 0097-8930.
URL <http://doi.acm.org/10.1145/127719.122757>

- [Bock et al., 2008] S. Bock, C. Käijhnel, T. Boskamp and H.-O. Peitgen. *Robust vessel segmentation*. pages 691,539–691,539–9, 2008.
URL <http://dx.doi.org/10.1117/12.768555>
- [Bogunovic et al., 2011] H. Bogunovic, J. M. Pozo, M. C. Villa-Uriol, C. B. L. M. Majoie, R. van den Berg, H. A. F. Gratama van Andel, J. M. Macho, J. Blasco, L. San Román and A. F. Frangi. *Automated segmentation of cerebral vasculature with aneurysms in 3DRA and TOF-MRA using geodesic active regions: An evaluation study*. Medical Physics, vol. 38, no. 1, pages 210–222, 2011.
URL <http://scitation.aip.org/content/aapm/journal/medphys/38/1/10.1118/1.3515749>
- [Bolles and Fischler, 1981] R. C. Bolles and M. A. Fischler. *A RANSAC-based approach to model fitting and its application to finding cylinders in range data*. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2, IJCAI'81*, pages 637–643, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1981.
URL <http://dl.acm.org/citation.cfm?id=1623264.1623272>
- [Bornik et al., 2005] A. Bornik, B. Reitinger and R. Beichel. *Simplex-Mesh Based Surface Reconstruction and Representation of Tubular Structures*. In *Bildverarbeitung für die Medizin 2005*, edited by H.-P. Meinzer, H. Handels, A. Horsch and T. Tolxdorff, Informatik aktuell, pages 143–147, Springer Berlin Heidelberg, 2005, ISBN 978-3-540-25052-4.
URL http://dx.doi.org/10.1007/3-540-26431-0_30
- [Bouix et al., 2005] S. Bouix, K. Siddiqi and A. Tannenbaum. *Flux driven automatic centerline extraction*. Medical Image Analysis, vol. 9, no. 3, pages 209–221, 2005.
- [Braude et al., 2007] I. Braude, J. Marker, K. Museth, J. Nissanov and D. E. Breen. *Contour-based surface reconstruction using MPU implicit models*. Graphical Models, vol. 69, no. 2, pages 139–157, 2007.
- [Brian Mohr, 2012] C. P. Brian Mohr, Saad Masood. *Accurate Lumen Segmentation and Stenosis Detection and Quantification in Coronary CTA*. MICCAI Workshop "3D Cardiovascular Imaging: a MICCAI segmentation Challenge", 2012.
URL <http://coronary.bigr.nl/stenoses/results/resultsmethod.php?id=273&type=testing&subs=public&sinfo=no&vendor=all>
- [Bühler et al., 2003] K. Bühler, P. Felkel and A. L. Cruz. *Geometric Methods for Vessel Visualization and Quantification - A Survey*. In *Geometric Modelling for Scientific Visualization*, edited by H. M. G. Brunnert, B. Hamann, pages 399–421, Springer, 2003, ISBN ISBN: 3-540-40116-4.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.12.5367>
- [Buhmann, 2003] M. Buhmann. *Radial Basis Functions: Theory and Implementations*. Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2003, ISBN 9781139435246.
URL <http://books.google.fr/books?id=TRMf53opzlsC>

- [Cai et al., 2003] Y. Cai, C. Chui, X. Ye, Y. Wang and J. H. Anderson. *{VR} simulated training for less invasive vascular intervention*. *Computers & Graphics*, vol. 27, no. 2, pages 215 – 221, 2003, ISSN 0097-8493.
URL <http://www.sciencedirect.com/science/article/pii/S0097849302002789>
- [Cani and Desbrun, 1997] M.-P. Cani and M. Desbrun. *Animation of Deformable Models Using Implicit Surfaces*. *IEEE Transactions on Visualization and Computer Graphics*, vol. 3, no. 1, pages 39 – 50, 1997, published under the name Marie-Paule Cani-Gascuel.
URL <http://hal.inria.fr/inria-00537529>
- [Carr et al., 2001] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum and T. R. Evans. *Reconstruction and representation of 3D objects with radial basis functions*. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 67–76, ACM, New York, NY, USA, 2001, ISBN 1-58113-374-X.
URL <http://doi.acm.org/10.1145/383259.383266>
- [Carr et al., 2003] J. C. Carr, R. K. Beatson, B. C. McCallum, W. R. Fright, T. J. McLennan and T. J. Mitchell. *Smooth Surface Reconstruction from Noisy Range Data*. In *Proceedings of the 1st International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, GRAPHITE '03, pages 119–ff, ACM, New York, NY, USA, 2003, ISBN 1-58113-578-5.
URL <http://doi.acm.org/10.1145/604471.604495>
- [Carrillo et al., 2007] J. Carrillo, M. Hoyos, E. Dávila and M. Orkisz. *Recursive tracking of vascular tree axes in 3D medical images*. *International Journal of Computer Assisted Radiology and Surgery*, vol. 1, no. 6, pages 331–339, 2007, ISSN 1861-6410.
URL <http://dx.doi.org/10.1007/s11548-007-0068-6>
- [Cebal et al., 2005] J. R. Cebal, M. A. Castro, S. Appanaboyina, C. M. Putman, D. Millan and A. F. Frangi. *Efficient pipeline for image-based patient-specific analysis of cerebral aneurysm hemodynamics: technique and sensitivity*. *IEEE Trans. Med. Imaging*, vol. 24, no. 4, pages 457–467, 2005.
- [Chaperon and Goulette, 2001] T. Chaperon and F. Goulette. *Extracting cylinders in full 3D data using a random sampling method and the Gaussian image*. In *Vision Modeling and Visualization Conference (VMV'01)*, pages 35 – 42, 2001.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.23.5998>
- [Chen and Lai, 2007] Y.-L. Chen and S.-H. Lai. *A Partition-of-Unity Based Algorithm for Implicit Surface Reconstruction Using Belief Propagation*. In *Shape Modeling and Applications, 2007. SMI '07. IEEE International Conference on*, pages 147–155, 2007.
- [Christian Beder and Wolfgang Förstner, 2006] Christian Beder and Wolfgang Förstner. *Direct solutions for computing cylinders from minimal sets of points*. In *9th European Conference on Computer Vision (ECCV'06)*, vol. 3954 of LNCS, pages 135–146, 2006.
URL http://dx.doi.org/10.1007/11744023_11

- [Coles et al., 2011] T. R. Coles, D. Meglan and N. W. John. *The Role of Haptics in Medical Training Simulators: A Survey of the State of the Art*. *EEE Trans. Haptics*, vol. 4, no. 1, pages 51–66, 2011, ISSN 1939-1412.
URL <http://dx.doi.org/10.1109/TOH.2010.19>
- [Cornea et al., 2007] N. Cornea, D. Silver and P. Min. *Curve-Skeleton Properties, Applications, and Algorithms*. *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 3, pages 530–548, 2007, ISSN 1077-2626.
URL http://coewww.rutgers.edu/www2/vizlab/NicuCornea/publication/1stpaper_download/tvcg06.pdf
- [Cotin et al., 2000] S. Cotin, S. Dawson, D. Meglan, D. Shaffer, M. Ferrell, R. Bardsley, F. Morgan, T. Nagano, J. Nikom, P. Sherman et al.. *ICTS, an interventional cardiology training system*. *Studies in health technology and informatics*, pages 59–65, 2000.
- [Cotin et al., 2005] S. Cotin, C. Duriez, J. Lenoir, P. Neumann and S. Dawson. *New Approaches to Catheter Navigation for Interventional Radiology Simulation*. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2005*, vol. 3750 of *Lecture Notes in Computer Science*, edited by J. Duncan and G. Gerig, pages 534–542, Springer Berlin Heidelberg, 2005, ISBN 978-3-540-29326-2.
URL http://dx.doi.org/10.1007/11566489_66
- [Dawson, 2006] S. Dawson. *Procedural Simulation: A Primer I*. *Radiology*, vol. 241, no. 1, pages 17–25, 2006.
URL <http://radiology.rsna.org/content/241/1/17.short>
- [Dekkers et al., 2004] D. Dekkers, K. van Overveld and R. Golsteijn. *Combining CSG modeling with soft blending using Lipschitz-based implicit surfaces*. *The Visual Computer*, vol. 20, no. 6, pages 380–391, 2004, ISSN 0178-2789.
URL <http://dx.doi.org/10.1007/s00371-002-0198-3>
- [Delingette and Montagnat, 2001] H. Delingette and J. Montagnat. *Topology and Shape Constraints on Parametric Active Contours*. *Computer Vision and Image Understanding*, vol. 83, no. 2, pages 1–10, 2001.
URL <http://hal.archives-ouvertes.fr/hal-00682936>
- [Dequidt et al., 2007] J. Dequidt, J. Lenoir and S. Cotin. *Interactive contacts resolution using smooth surface representation*. In *Proceedings of the 10th international conference on Medical image computing and computer-assisted intervention, MICCAI'07*, pages 850–857, Springer-Verlag, Berlin, Heidelberg, 2007, ISBN 3-540-75758-9, 978-3-540-75758-0.
URL <http://dl.acm.org/citation.cfm?id=1775835.1775951>
- [Dequidt et al., 2009] J. Dequidt, C. Duriez, S. Cotin et al.. *Towards interactive planning of coil embolization in brain aneurysms*. In *MICCAI 2009*, vol. 5761 of *LNCS*, pages 377–385, Springer-Verlag, 2009.
URL <http://hal.inria.fr/inria-00430867>

- [Deschamps and Cohen, 2002] T. Deschamps and L. Cohen. *Fast extraction of tubular and tree 3D surfaces with front propagation methods*. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 1, pages 731–734 vol.1, 2002, ISSN 1051-4651.
- [Dufour et al., 2013] A. Dufour, O. Tankyevych, B. Naegel, H. Talbot, C. Ronse, J. Baruthio, P. Dokládál and N. Passat. *Filtering and segmentation of 3D angiographic data: Advances based on mathematical morphology*. *Medical Image Analysis*, vol. 17, no. 2, pages 147–164, 2013.
- [Duriez, 2013] C. Duriez. *Simulation temps-réel d'interventions médicales impliquant des déformations et des interactions mécaniques entre les tissus et les outils (Manuscrit en anglais)*. Hdr, Université des Sciences et Technologie de Lille - Lille I, 2013.
URL <http://tel.archives-ouvertes.fr/tel-00785118>
- [Duriez et al., 2006] C. Duriez, S. Cotin, J. Lenoir and P. Neumann. *New approaches to catheter navigation for interventional radiology simulation I*. *Computer Aided Surgery*, vol. 11, no. 6, pages 300–308, 2006.
URL <http://informahealthcare.com/doi/abs/10.3109/10929080601090623>
- [Eberly, 2008] D. Eberly. *Fitting 3D data with a cylinder*. Online, February, 2008.
URL <http://www.geometrictools.com/>
- [Enright et al., 2002] D. Enright, S. Marschner and R. Fedkiw. *Animation and Rendering of Complex Water Surfaces*. *ACM Trans. Graph.*, vol. 21, no. 3, pages 736–744, 2002, ISSN 0730-0301.
URL <http://doi.acm.org/10.1145/566654.566645>
- [Fekete, 1990] G. Fekete. *Rendering and managing spherical data with sphere quadtrees*. In *Proceedings of the 1st conference on Visualization '90, VIS '90*, pages 176–186, IEEE Computer Society Press, Los Alamitos, CA, USA, 1990, ISBN 0-8186-2083-8.
URL <http://dl.acm.org/citation.cfm?id=949531.949560>
- [Felkel et al., 2004] P. Felkel, R. Wegenkittl and K. Buhler. *Surface Models of Tube Trees*. In *Proceedings of the Computer Graphics International, CGI '04*, pages 70–77, IEEE Computer Society, Washington, DC, USA, 2004, ISBN 0-7695-2171-1.
URL <http://dx.doi.org/10.1109/CGI.2004.93>
- [Fischler and Bolles, 1981] M. A. Fischler and R. C. Bolles. *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. *Commun. ACM*, vol. 24, no. 6, pages 381–395, 1981, ISSN 0001-0782.
URL <http://doi.acm.org/10.1145/358669.358692>
- [Flasque et al., 2001] N. Flasque, M. Desvignes, J.-M. Constans and M. Revenu. *Acquisition, segmentation and tracking of the cerebral vascular tree on 3D magnetic resonance angiography images*. *Medical Image Analysis*, vol. 5, no. 3, pages 173–183, 2001.
URL <http://hal.archives-ouvertes.fr/hal-00805904>

- [Foster and Fedkiw, 2001] N. Foster and R. Fedkiw. *Practical Animation of Liquids*. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 23–30, ACM, New York, NY, USA, 2001, ISBN 1-58113-374-X.
URL <http://doi.acm.org/10.1145/383259.383261>
- [Frangi et al., 1999] A. Frangi, W. Niessen, R. Hoogeveen, T. Van Walsum and M. Viergever. *Model-based quantitation of 3-D magnetic resonance angiographic images*. *Medical Imaging*, IEEE Transactions on, vol. 18, no. 10, pages 946–956, 1999, ISSN 0278-0062.
- [Freiman et al., 2009] M. Freiman, J. Frank, L. Weizman, E. Nammer, O. Shilon, L. Joskowicz and J. Sosna. *Nearly automatic vessels segmentation using graph-based energy minimization*. 2009.
- [Friman et al., 2010] O. Friman, M. Hindennach, C. Kühnel and H.-O. Peitgen. *Multiple hypothesis template tracking of small 3D vessel structures*. *MedIA*, vol. 14, no. 2, pages 160 – 171, 2010.
URL http://www.mevis-research.de/~ola/publications/Friman_MedIA2010.pdf
- [Gallup et al., 2010] D. Gallup, J.-M. Frahm and M. Pollefeys. *Piecewise planar and non-planar stereo for urban scene reconstruction*. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1418–1425, 2010, ISSN 1063-6919.
- [Gelas et al., 2007] A. Gelas, J. Schaerer, O. Bernard, D. Friboulet, P. Clarysse, I. E. Magnin and R. Prost. *Radial Basis Functions Collocation Methods for Model Based Level-Set Segmentation*. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, vol. 2, pages II – 237–II – 240, 2007, ISSN 1522-4880.
- [Gerig et al., 1993] G. Gerig, T. Koller, G. Székely, C. Brechbühler and O. Kübler. *Symbolic description of 3-D structures applied to cerebral vessel tree obtained from MR angiography volume data*. In *Information Processing in Medical Imaging*, pages 94–111, Springer Berlin / Heidelberg, 1993.
URL <http://dx.doi.org/10.1007/bfb0013783>
- [Goldman, 2005] R. Goldman. *Curvature formulas for implicit curves and surfaces*. *Comput. Aided Geom. Des.*, vol. 22, no. 7, pages 632–658, 2005, ISSN 0167-8396.
URL <http://dx.doi.org/10.1016/j.cagd.2005.06.005>
- [Gomes and Faugeras, 1999] J. Gomes and O. Faugeras. *Reconciling Distance Functions and Level Sets*. Tech. Rep. RR-3666, INRIA, 1999.
URL <http://hal.inria.fr/inria-00073006>
- [Gómez et al., 2007] O. Gómez, J. González and E. Morales. *Image Segmentation Using Automatic Seeded Region Growing and Instance-Based Learning*. In *Progress in Pattern Recognition, Image Analysis and Applications*, vol. 4756 of *Lecture Notes in Computer Science*, edited by L. Rueda, D. Mery and J. Kittler, pages 192–201, Springer Berlin Heidelberg, 2007, ISBN 978-3-540-76724-4.
URL http://dx.doi.org/10.1007/978-3-540-76725-1_21

- [Gould et al., 2012] D. Gould, N. Chalmers, S. Johnson, C. Kilkenny, M. White, B. Bech, L. Lonn and F. Bello. *Simulation: Moving from Technology Challenge to Human Factors Success*. CardioVascular and Interventional Radiology, vol. 35, no. 3, pages 445–453, 2012, ISSN 0174-1551.
URL <http://dx.doi.org/10.1007/s00270-011-0266-z>
- [Gourmel et al., 2012] O. Gourmel, L. Barthe, M.-P. Cani, B. Wyvill, A. Bernhardt, M. Paulin and H. Grasberger. *A Gradient-Based Implicit Blend*. ACM Trans. Graph., 2012.
URL <http://hal.inria.fr/hal-00753246>
- [Gülsün and Tek, 2008] M. A. Gülsün and H. Tek. *Robust Vessel Tree Modeling*. In *Proceedings of the 11th international conference on Medical Image Computing and Computer-Assisted Intervention - Part I, MICCAI '08*, pages 602–611, Springer-Verlag, Berlin, Heidelberg, 2008, ISBN 978-3-540-85987-1.
URL http://dx.doi.org/10.1007/978-3-540-85988-8_72
- [Hahn et al., 2001] H. K. Hahn, B. Preim, D. Selle and H. otto Peitgen. *Visualization and Interaction Techniques for the Exploration of Vascular Structures*. In *Visualization, 2001. VIS '01*, pages 395–402, 2001.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.3933>
- [Hegadi et al., 2010] R. Hegadi, A. Kop and M. Hangarge. *A Survey on Deformable Model and its Applications to Medical Imaging*. IJCA, Special Issue on RTIPPR, , no. 2, pages 64–75, 2010, published By Foundation of Computer Science.
- [Hernandez and Frangi, 2007] M. Hernandez and A. F. Frangi. *Non-parametric geodesic active regions: Method and evaluation for cerebral aneurysms segmentation in 3DRA and CTA*. Medical Image Analysis, vol. 11, no. 3, pages 224–241, 2007.
- [Hijazi et al., 2010] Y. Hijazi, D. Bechmann, D. Cazier, C. Kern and S. Thery. *Fully-automatic Branching Reconstruction Algorithm: Application to Vascular Trees*. Shape Modeling and Applications, International Conference on, vol. 0, pages 221–225, 2010.
- [Hiratsuka et al., 2008] Y. Hiratsuka, H. Miki, I. Kiriya, K. kikuchi, S. takahashi, I. Matsubara, K. Sadamoto and T. Mochizuki. *Diagnosis of Unruptured Intracranial Aneurysms: 3T MR Angiography versus 64-channel Multi-detector Row CT Angiography*. Magnetic Resonance in Medical Sciences, vol. 7, no. 4, pages 169–178, 2008.
- [Höfer et al., 2002] U. Höfer, T. Langen, J. Nziki, F. Zeitler, J. Hesser, U. MÄijller, W. Voelker and R. MÄd'nnner. *CathI – catheter instruction system*. In *CARS 2002 Computer Assisted Radiology and Surgery*, edited by H. Lemke, K. Inamura, K. Doi, M. Vannier, A. Farman and J. Reiber, pages 101–106, Springer Berlin Heidelberg, 2002, ISBN 978-3-642-62844-3.
URL http://dx.doi.org/10.1007/978-3-642-56168-9_17
- [Hong et al., 2012] Q. Hong, Q. Li and J. Tian. *Implicit Reconstruction of Vasculatures Using Bivariate Piecewise Algebraic Splines*. IEEE Trans. Med. Imaging, vol. 31, no. 3, pages 543–553, 2012, ISSN 0278-0062.

- [Hough, 1962] P. Hough. *Method and Means for Recognizing Complex Patterns*. U.S. Patent 3.069.654, 1962.
- [Hoyos et al., 2006] M. Hoyos, P. Orlowski, E. Piatkowska-Janko, P. Bogorodzki and M. Orkisz. *Vascular Centerline Extraction in 3D MR Angiograms for Phase Contrast MRI Blood Flow Measurement*. International Journal of Computer Assisted Radiology and Surgery, vol. 1, no. 1, pages 51–61, 2006, ISSN 1861-6410.
URL <http://dx.doi.org/10.1007/s11548-006-0005-0>
- [Huysmans et al., 2005] T. Huysmans, J. Sijbers and B. Verdonk. *Parametrization of Tubular Surfaces on the Cylinder*. Journal of WSCG, vol. 13, no. 3, pages 97–104, 2005.
- [Jiang et al., 2013] H. Jiang, B. He, D. Fang, Z. Ma, B. Yang and L. Zhang. *A region growing vessel segmentation algorithm based on spectrum information*. Comput Math Methods Med, vol. 2013, no. 743870, page 9, 2013.
- [Jiang et al., 2007] Y. Jiang, A. Bainbridge-Smith and A. Morris. *Blood vessel tracking in retinal images*. Proceedings of Image and Vision Computing, pages 126–131, 2007.
- [Jin et al., 2000] X. Jin, Y. Li and Q. Peng. *General constrained deformations based on generalized metaballs*. Computers & Graphics, vol. 24, no. 2, pages 219 – 231, 2000, ISSN 0097-8493.
URL <http://www.sciencedirect.com/science/article/pii/S0097849399001569>
- [Jin et al., 2001] X. Jin, C.-L. Tai, J. Feng and Q. Peng. *Convolution Surfaces for Line Skeletons with Polynomial Weight Distributions*. Journal of Graphics Tools, vol. 6, no. 3, pages 17–28, 2001.
URL <http://www.tandfonline.com/doi/abs/10.1080/10867651.2001.10487542>
- [Joshi and Séquin, 2007] P. Joshi and C. H. Séquin. *Energy Minimizers for Curvature-Based Surface Functionals*. CAD Conference, Waikiki, Hawaii, pages 607–617, 2007.
URL <http://graphics.berkeley.edu/papers/Joshi-EMC-2007-06/>
- [Kanamori et al., 2008] Y. Kanamori, Z. Szego and T. Nishita. *GPU-based Fast Ray Casting for a Large Number of Metaballs*. Computer Graphics Forum, vol. 27, no. 2, pages 351–360, 2008, ISSN 1467-8659.
URL <http://dx.doi.org/10.1111/j.1467-8659.2008.01132.x>
- [Kanazawa and Kawakami, 2004] Y. Kanazawa and H. Kawakami. *Detection of planar regions with uncalibrated stereo using distribution of feature points*. In *In British Machine Vision Conference*, pages 247–256, 2004.
- [Kass et al., 1988] M. Kass, A. Witkin and D. Terzopoulos. *Snakes: Active contour models*. International Journal of Computer Vision, vol. 1, no. 4, pages 321–331, 1988, ISSN 0920-5691.
URL <http://dx.doi.org/10.1007/BF00133570>

- [Kazhdan et al., 2006] M. Kazhdan, M. Bolitho and H. Hoppe. *Poisson surface reconstruction*. In *Proceedings of the fourth Eurographics symposium on Geometry processing, SGP '06*, pages 61–70, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2006, ISBN 3-905673-36-3.
URL <http://dl.acm.org/citation.cfm?id=1281957.1281965>
- [Kirbas and Quek, 2004] C. Kirbas and F. Quek. *A review of vessel extraction techniques and algorithms*. *ACM Comput. Surv.*, vol. 36, no. 2, pages 81–121, 2004, ISSN 0360-0300.
URL <http://doi.acm.org/10.1145/1031120.1031121>
- [Klein et al., 1997] A. Klein, F. Lee and A. Amini. *Quantitative coronary angiography with deformable spline models*. *Medical Imaging, IEEE Transactions on*, vol. 16, no. 5, pages 468–482, 1997, ISSN 0278-0062.
- [Kobbelt and Botsch, 2004] L. Kobbelt and M. Botsch. *A Survey of Point-based Techniques in Computer Graphics*. *Comput. Graph.*, vol. 28, no. 6, pages 801–814, 2004, ISSN 0097-8493.
URL <http://dx.doi.org/10.1016/j.cag.2004.08.009>
- [Kockara et al., 2007] S. Kockara, T. Halic, K. Iqbal, C. Bayrak and R. Rowe. *Collision detection: A survey*. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 4046–4051, 2007.
- [Kretschmer et al., 2013] J. Kretschmer, C. Godenschwager, B. Preim and M. Stamminger. *Interactive Patient-Specific Vascular Modeling with Sweep Surfaces*. *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, no. 12, pages 2828–2837, 2013, ISSN 1077-2626.
- [La Cruz et al., 2004] A. La Cruz, M. Straka, A. Kochl, M. Sramek, E. Groller and D. Fleischmann. *Non-Linear Model Fitting to Parameterize Diseased Blood Vessels*. In *Proceedings of the conference on Visualization '04, VIS '04*, pages 393–400, IEEE Computer Society, Washington, DC, USA, 2004, ISBN 0-7803-8788-0.
URL <http://dx.doi.org/10.1109/VISUAL.2004.72>
- [Lachaud and Montanvert, 1999] J.-O. Lachaud and A. Montanvert. *Deformable meshes with automated topology changes for coarse-to-fine 3D surface extraction*. *MedIA*, vol. 3, no. 2, pages 187–207, 1999.
URL <http://www.sciencedirect.com/science/article/pii/S1361841599800061>
- [Law and Chung, 2007] M. W. K. Law and A. C. S. Chung. *Weighted Local Variance-Based Edge Detection and Its Application to Vascular Segmentation in Magnetic Resonance Angiography*. *IEEE Trans. Med. Imaging*, vol. 26, no. 9, pages 1224–1241, 2007.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.136.1718>
- [Lempitsky, 2010] V. Lempitsky. *Surface extraction from binary volumes with higher-order smoothness*. In *CVPR 2010*, pages 1197–1204, 2010.
URL <http://research.microsoft.com/apps/pubs/default.aspx?id=79956>

- [Lenoir et al., 2006] J. Lenoir, S. Cotin, C. Duriez and P. Neumann. *Interactive physically-based simulation of catheter and guidewire*. *Computers & Graphics*, vol. 30, no. 3, pages 416–422, 2006, ISSN 0097-8493.
URL <http://www.sciencedirect.com/science/article/pii/S0097849306000641>
- [Lesage et al., 2009] D. Lesage, E. Angelini, I. Bloch et al.. *A review of 3D vessel lumen segmentation techniques: Models, features and extraction schemes*. *MedIA*, vol. 13, no. 6, pages 819–845, 2009.
URL http://perso.telecom-paristech.fr/~bloch/papers/MedIA_David2009.pdf
- [Levet et al., 2006] F. Levet, X. Granier and C. Schlick. *Fast Sampling of Implicit Surfaces by Particle Systems*. In *Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on*, pages 39–39, 2006.
- [Li and Yezzi, 2006] H. Li and A. Yezzi. *Vessels as 4D Curves: Global Minimal 4D Paths to Extract 3D Tubular Surfaces*. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW '06. Conference on*, pages 82–82, 2006.
- [Li et al., 2009] H. Li, A. Yezzi and L. Cohen. *3D multi-branch tubular surface and centerline extraction with 4D iterative key points*. In *MICCAI 2009*, vol. 5761 of *LNCS*, pages 1042–1050, 2009.
URL <http://www.ceremade.dauphine.fr/~cohen/mypapers/HuaLiMICCAI09.pdf>
- [Li et al., 2012] S. Li, J. Guo, Q. Wang, Q. Meng, Y.-P. Chui, J. Qin and P.-A. Heng. *A Catheterization-Training Simulator Based on a Fast Multigrid Solver*. *Computer Graphics and Applications, IEEE*, vol. 32, no. 6, pages 56–70, 2012, ISSN 0272-1716.
- [Li et al., 2001] Z. Li, C.-K. Chui, J. H. Anderson, X. Chen, X. Ma, W. Hua, Q. Peng, Y. Cai, Y. Wang and W. L. Nowinski. *Computer Environment for Interventional Neuroradiology Procedures*. *Simulation & Gaming*, vol. 32, no. 3, pages 404–419, 2001.
URL <http://sag.sagepub.com/content/32/3/404.abstract>
- [Lim et al., 1998] H. L. Lim, B. R. Shetty, C. K. Chui, Y. P. Wang and Y. Y. Cai. *Real-time interactive surgical simulator for catheter navigation*. 1998.
URL <http://dx.doi.org/10.1117/12.309454>
- [Lingrand and Montagnat, 2005] D. Lingrand and J. Montagnat. *Levelset and B-Spline Deformable Model Techniques for Image Segmentation: A Pragmatic Comparative Study*. In *Image Analysis*, vol. 3540 of *Lecture Notes in Computer Science*, edited by H. Kalviainen, J. Parkkinen and A. Kaarna, pages 25–34, Springer Berlin Heidelberg, 2005, ISBN 978-3-540-26320-3.
URL http://dx.doi.org/10.1007/11499145_4
- [Lorigo et al., 1999] L. Lorigo, O. Faugeras, W. Grimson, R. Keriven, R. Kikinis and C.-F. Westin. *Co-dimension 2 Geodesic Active Contours for MRA Segmentation*. In *Information Processing in Medical Imaging*, vol. 1613 of *Lecture Notes in Computer Science*, edited by

- A. Kuba, M. Saamal and A. Todd-Pokropek, pages 126–139, Springer Berlin Heidelberg, 1999, ISBN 978-3-540-66167-2.
URL http://dx.doi.org/10.1007/3-540-48714-X_10
- [Lozano-Perez et al., 1987] T. Lozano-Perez, W. Grimson and S. White. *Finding cylinders in range data*. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4, pages 202–207, IEEE, 1987.
- [Luboz et al., 2009] V. Luboz, C. Hughes, D. Gould, N. John and F. Bello. *Real-time Seldinger technique simulation in complex vascular models*. *International Journal of Computer Assisted Radiology and Surgery*, vol. 4, no. 6, pages 589–596, 2009, ISSN 1861-6410.
URL <http://dx.doi.org/10.1007/s11548-009-0376-0>
- [Luboz et al., 2013] V. Luboz, J. Kyaw-Tun, S. Sen, R. Kneebone, R. Dickinson, R. Kitney and F. Bello. *Real-time stent and balloon simulation for stenosis treatment*. *The Visual Computer*, pages 1–9, 2013, ISSN 0178-2789.
URL <http://dx.doi.org/10.1007/s00371-013-0859-4>
- [Lukács et al., 1998] G. Lukács, R. Martin and D. Marshall. *Faithful least-squares fitting of spheres, cylinders, cones and tori for reliable segmentation*. In *Computer Vision – ECCV’98*, vol. 1406 of *Lecture Notes in Computer Science*, edited by H. Burkhardt and B. Neumann, pages 671–686, Springer Berlin Heidelberg, 1998, ISBN 978-3-540-64569-6.
URL <http://dx.doi.org/10.1007/BFb0055697>
- [Ma, 2007] X. Ma. *Latest development of an interventional radiology training simulation system: neurocath*. In *Proceedings of the 1st international conference on Digital human modeling*, ICDHM’07, pages 684–693, Springer-Verlag, Berlin, Heidelberg, 2007, ISBN 978-3-540-73318-8.
URL <http://dl.acm.org/citation.cfm?id=1784074.1784156>
- [Macêdo et al., 2009] I. Macêdo, J. a. P. Gois and L. Velho. *Hermite Interpolation of Implicit Surfaces with Radial Basis Functions*. In *Proceedings of the 2009 XXII Brazilian Symposium on Computer Graphics and Image Processing*, SIBGRAP ’09, pages 1–8, IEEE Computer Society, Washington, DC, USA, 2009, ISBN 978-0-7695-3813-6.
URL <http://dx.doi.org/10.1109/SIBGRAP.2009.11>
- [Manniesing et al., 2007] R. Manniesing, M. A. Viergever and W. J. Niessen. *Vessel Axis Tracking Using Topology Constrained Surface Evolution*. *IEEE Trans. Med. Imaging*, vol. 26, no. 3, pages 309–316, 2007.
- [Masutani et al., 1996] Y. Masutani, K. Masamune and T. Dohi. *Region-growing based feature extraction algorithm for tree-like objects*. In *Visualization in Biomedical Computing*, vol. 1131 of *LNCS*, pages 159–171, Springer-Verlag, 1996.
URL <http://www.ut-radiology.umin.jp/people/masutani/PDF/masutani-VBC96.pdf>

- [McInerney and Terzopoulos, 1999] T. McInerney and D. Terzopoulos. *Topology adaptive deformable surfaces for medical image volume segmentation*. Medical Imaging, IEEE Transactions on, vol. 18, no. 10, pages 840–850, 1999, ISSN 0278-0062.
- [McKinney et al., 2008] A. McKinney, C. Palmer, C. Truwit, A. Karagulle and M. Teksam. *Detection of aneurysms by 64-section multidetector CT angiography in patients acutely suspected of having an intracranial aneurysm and comparison with digital subtraction and 3D rotational angiography*. AJNR Am J Neuroradiol, vol. 29, no. 3, pages 594–602, 2008.
- [Meglan, 1996] D. Meglan. *Making surgical simulation real*. SIGGRAPH Comput. Graph., vol. 30, no. 4, pages 37–39, 1996, ISSN 0097-8930.
URL <http://doi.acm.org/10.1145/240806.240811>
- [Meyer et al., 2007] M. Meyer, B. Nelson, R. M. Kirby and R. Whitaker. *Particle Systems for Efficient and Accurate High-Order Finite Element Visualization*. IEEE Transactions on Visualization and Computer Graphics, vol. 13, no. 5, pages 1015–1026, 2007, ISSN 1077-2626.
URL <http://dx.doi.org/10.1109/TVCG.2007.1048>
- [Mihalef et al., 2007] V. Mihalef, D. Metaxas and M. Sussman. *Textured Liquids based on the Marker Level Set*. Comput. Graph. Forum, vol. 26, no. 3, pages 457–466, 2007.
- [Mille and Cohen, 2009] J. Mille and L. Cohen. *Deformable tree models for 2D and 3D branching structures extraction*. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 149–156, 2009, ISSN 2160-7508.
URL <http://liris.cnrs.fr/julien.mille/doc/mmbia09.pdf>
- [Mohan et al., 2010] V. Mohan, G. Sundaramoorthi and A. Tannenbaum. *Tubular Surface Segmentation for Extracting Anatomical Structures From Medical Imagery*. IEEE Trans. Med. Imaging, vol. 29, no. 12, pages 1945–1958, 2010, ISSN 0278-0062.
- [Morse et al., 2005] B. S. Morse, W. Liu, T. S. Yoo and K. Subramanian. *Active contours using a constraint-based implicit representation*. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, ACM, New York, NY, USA, 2005.
URL <http://doi.acm.org/10.1145/1198555.1198655>
- [Mouat and Beatson, 2002] C. Mouat and R. K. Beatson. *RBF Collocation*. Tech. Rep. UCDSMS2002/3, University of Canterbury, 2002.
URL http://www.rbfsrus.com/pdfs/mouat_beatson_rbf_pde/mouat_beatson_rbf_pde.pdf
- [Mullen et al., 2010] P. Mullen, F. de Goes, M. Desbrun, D. Cohen-Steiner and P. Alliez. *Signing the unsigned: Robust surface reconstruction from raw pointsets*. In *Eurographics Symp. on Geometry Proc.*, pages 1733–1741, 2010.
URL <http://hal.inria.fr/inria-00502473>

- [Muraki, 1991] S. Muraki. *Volumetric shape description of range data using Blobby Model*. SIGGRAPH Comput. Graph., vol. 25, pages 227–235, 1991.
URL <http://doi.acm.org/10.1145/122718.122743>
- [Nesme and Bouthors, 2006] M. Nesme and A. Bouthors. *Dynamic Triangulation of Implicit Surfaces: towards the handling of topology changes*. Rapport de recherche RR-6128, INRIA, 2006.
URL <http://hal.inria.fr/inria-00132537>
- [Neugebauer et al., 2010] M. Neugebauer, V. Diehl, M. Skalej and B. Preim. *Geometric Reconstruction of the Ostium of Cerebral Aneurysms*. In VMV, pages 307–314, 2010.
- [Nowinski and Chui, 2001] W. Nowinski and C.-K. Chui. *Simulation of interventional neuro-radiology procedures*. In *Medical Imaging and Augmented Reality, 2001. Proceedings. International Workshop on*, pages 87–94, 2001.
- [Oeltze and Preim, 2005] S. Oeltze and B. Preim. *Visualization of vasculature with convolution surfaces: method, validation and evaluation*. IEEE Trans. Med. Imaging, vol. 24, no. 4, pages 540–548, 2005.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.71.1739>
- [Ohtake et al., 2006] Y. Ohtake, A. G. Belyaev and H.-P. Seidel. *Sparse surface reconstruction with adaptive partition of unity and radial basis functions*. Graphical Models, vol. 68, no. 1, pages 15–24, 2006.
- [Ou and Bin, 2005] S. Ou and H. Bin. *Subdivision method to create furcating object with multi-branches*. The Visual Computer, vol. 21, no. 3, pages 170–187, 2005, ISSN 0178-2789.
URL <http://dx.doi.org/10.1007/s00371-005-0280-8>
- [Passat et al., 2007] N. Passat, C. Ronse, J. Baruthio, J. P. Armspach and J. Foucher. *Watershed and multimodal data for brain vessel segmentation: Application to the superior sagittal sinus*. Image Vision Comput., vol. 25, no. 4, pages 512–521, 2007, ISSN 0262-8856.
URL <http://dx.doi.org/10.1016/j.imavis.2006.03.008>
- [Pedicelli et al., 2007] A. Pedicelli, M. Rollo, G. Di Lella, T. Tartaglione, C. Colosimo and L. Bonomo. *3D rotational angiography for the diagnosis and preoperative assessment of intracranial aneurysms: preliminary experience*. Radiol Med (Torino), 2007.
- [Piccinelli et al., 2009] M. Piccinelli, A. Veneziani, D. A. Steinman, A. Remuzzi and L. Antiga. *A Framework for Geometric Analysis of Vascular Structures: Application to Cerebral Aneurysms*. IEEE Trans. Med. Imaging, vol. 28, no. 8, pages 1141–1155, 2009.
URL <http://www.engineering.uiowa.edu/~raghavan/images/chung/04915792.pdf>
- [Pizaine et al., 2011] G. Pizaine, E. Angelini, I. Bloch and S. Makram-Ebeid. *Vessel geometry modeling and segmentation using convolution surfaces and an implicit medial axis*. In *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*, pages 1421–1424, 2011, ISSN 1945-7928.

- [Preim and Oeltze, 2008] B. Preim and S. Oeltze. *3D Visualization of Vasculature: An Overview*. In *Visualization in Medicine and Life Sciences*, edited by L. Linsen, H. Hagen, B. Hamann, G. Farin, H.-C. Hege, D. Hoffman, C. R. Johnson, K. Polthier and M. Rumpf, Mathematics and Visualization, pages 39–59, Springer Berlin Heidelberg, 2008, ISBN 978-3-540-72630-2.
URL http://www.medvis-book.de/MedVisBookMaterial/Chapter14_Vessels/VesselVisOverview.pdf
- [Przemieniecki, 1985] J. Przemieniecki. *Theory of Matrix Structural Analysis*. Dover books on engineering, DOVER PUBN Incorporated, 1985, ISBN 9780486649481.
URL <http://books.google.fr/books?id=Pb8qTzqOKbAC>
- [Quek and Kirbas, 2001] F. Quek and C. Kirbas. *Vessel extraction in medical images by wave-propagation and traceback*. Medical Imaging, IEEE Transactions on, vol. 20, no. 2, pages 117–131, 2001, ISSN 0278-0062.
- [Rabbani and Heuvel, 2005] T. Rabbani and F. V. D. Heuvel. *EFFICIENT HOUGH TRANSFORM FOR AUTOMATIC DETECTION OF CYLINDERS IN POINT CLOUDS*. Workshop "Laser scanning 2005", 2005.
URL <http://www.isprs.org/proceedings/xxxvi/3-w19/papers/060.pdf>
- [Raspolli et al., 2005] M. Raspolli, C. Avizzano, G. Facenza and M. Bergamasco. *HERMES: an angioplasty surgery simulator*. In *Eurohaptics Conference, 2005 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2005. World Haptics 2005. First Joint*, pages 148–156, 2005.
- [Rebholz et al., 2004] P. Rebholz, C. Bienek, D. Stsepankou and J. Hesser. *CathI – Training System for PTCA. A Step Closer to Reality*. In *Medical Simulation*, vol. 3078 of *Lecture Notes in Computer Science*, edited by S. Cotin and D. Metaxas, pages 249–255, Springer Berlin Heidelberg, 2004, ISBN 978-3-540-22186-9.
URL http://dx.doi.org/10.1007/978-3-540-25968-8_28
- [Rouiller, 2011] O. Rouiller. *Real time rendering of skeletal implicit surfaces*. Master's thesis, Technical University of Denmark, DTU Informatics, E-mail: reception@imm.dtu.dk, Asmussens Alle, Building 305, DK-2800 Kgs. Lyngby, Denmark, 2011, supervised by Associate Professor Jakob Andreas Bærentzen, jab@imm.dtu.dk.
- [Samozino et al., 2006] M. Samozino, M. Alexa, P. Alliez and M. Yvinec. *Reconstruction with Voronoi Centered Radial Basis Functions*. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing, SGP '06*, pages 51–60, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2006, ISBN 3-905673-36-3.
URL <http://dl.acm.org/citation.cfm?id=1281957.1281964>
- [Schaap et al., 2007] M. Schaap, I. Smal, C. Metz, T. Walsum and W. Niessen. *Bayesian Tracking of Elongated Structures in 3D Images*. In *Information Processing in Medical Imaging*, vol. 4584 of *Lecture Notes in Computer Science*, edited by N. Karssemeijer and B. Lelieveldt, pages 74–85, Springer Berlin Heidelberg, 2007, ISBN 978-3-540-73272-3.
URL http://dx.doi.org/10.1007/978-3-540-73273-0_7

- [Schaap et al., 2009] M. Schaap, C. Metz, T. van Walsum, A. van der Giessen, A. Weustink, N. Mollet, C. Bauer, H. BogunoviÄ, C. Castro, X. Deng, E. Dikici, T. O'Donnell, M. Frenay, O. Friman, M. H. Hoyos, P. Kitslaar, K. Krissian, C. KÄijhnel, M. A. Luengo-Oroz, M. Orkisz, Å. Smedby, M. Styner, A. Szymczak, H. Tek, C. Wang, S. K. Warfield, S. Zambal, Y. Zhang, G. P. Krestin and W. Niessen. *Standardized Evaluation Methodology and Reference Database for Evaluating Coronary Artery Centerline Extraction Algorithms*. MedIA, vol. 13/5, pages 701–714, 2009.
URL <http://www.sciencedirect.com/science/article/pii/S1361841509000474>
- [Scherl et al., 2007] H. Scherl, J. Hornegger, M. PrÄijmmer and M. Lell. *Semi-automatic level-set based segmentation and stenosis quantification of the internal carotid artery in 3D {CTA} data sets*. Medical Image Analysis, vol. 11, no. 1, pages 21 – 34, 2007, ISSN 1361-8415.
URL <http://www.sciencedirect.com/science/article/pii/S1361841506000764>
- [Schnabel et al., 2007] R. Schnabel, R. Wahl and R. Klein. *Efficient RANSAC for Point-Cloud Shape Detection*. Computer Graphics Forum, vol. 26, no. 2, pages 214–226, 2007.
- [Schumann et al., 2008] C. Schumann, M. Neugebauer, R. Bade et al.. *Implicit Vessel Surface Reconstruction for Visualization and CFD Simulation*. IJCARS, vol. 2, no. 5, pages 275–286, 2008.
URL http://www.mevis-research.de/~schumann/pdf/schumann_2008_ijcars.pdf
- [Shafer and Kanade, 1983] S. Shafer and T. Kanade. *The Theory of Straight Homogeneous Generalized Cylinders and A Taxonomy of Generalized Cylinders*. In *Proceedings of the 1983 DARPA Image Understanding Workshop*, pages 210–218, 1983.
- [Shang et al., 2011] Y. Shang, R. Deklerck, E. Nyssen, A. Markova, J. de Mey, X. Yang and K. Sun. *Vascular Active Contour for Vessel Tree Segmentation*. Biomedical Engineering, IEEE Transactions on, vol. 58, no. 4, pages 1023–1032, 2011, ISSN 0018-9294.
- [Sherstyuk, 1999] A. Sherstyuk. *Kernel Functions in Convolution Surfaces: A Comparative Analysis*. The Visual Computer, vol. 15, no. 4, pages 171–182, 1999.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.4267>
- [Singh et al., 2002] V. Singh, D. Gress, R. Higashida, C. Dowd, V. Halbach and S. Johnston. *The learning curve for coil embolization of unruptured intracranial aneurysms*. AJNR Am J Neuroradiol, vol. 23, no. 5, pages 768–71, 2002.
- [Suhuai Luo, 2014] J. L. Suhuai Luo, Xuechen Li. *Review on the Methods of Automatic Liver Segmentation from Abdominal Images*. Journal of Computer and Communications, , no. 1-7, 2014.
- [Surazhsky et al., 2003] T. Surazhsky, E. Magid, O. Soldea, G. Elber and E. Rivlin. *A comparison of Gaussian and mean curvatures estimation methods on triangular meshes*. In

- Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, vol. 1, pages 1021–1026 vol.1, 2003, ISSN 1050-4729.
- [Suri et al., 2002] J. S. Suri, K. Liu, L. Reden and S. Laxminarayan. *A review on MR vascular image processing: skeleton versus nonskeleton approaches: part II*. IEEE Transactions on Information Technology in Biomedicine, vol. 6, no. 4, pages 338–350, 2002.
- [Szeliski et al., 1993] R. Szeliski, D. Tonnesen and D. Terzopoulos. *Modeling surfaces of arbitrary topology with dynamic particles*. In *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR '93., 1993 IEEE Computer Society Conference on*, pages 82–87, 1993, ISSN 1063-6919.
- [Taubin, 1991] G. Taubin. *Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation*. PAMI, vol. 13, pages 1115–1138, 1991.
URL <http://mesh.brown.edu/taubin/pdfs/Taubin-pami91.pdf>
- [Tek et al., 2001] H. Tek, D. Comaniciu and J. P. Williams. *Vessel Detection by Mean Shift-Based Ray Propagation*. In *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA'01)*, MMBIA '01, pages 228–, IEEE Computer Society, Washington, DC, USA, 2001, ISBN 0-7695-1336-0.
URL <http://dl.acm.org/citation.cfm?id=882464.882802>
- [Teschner et al., 2005] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrman, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser and P. Volino. *Collision Detection for Deformable Objects*. Computer Graphics Forum, 2005.
URL <http://hal.inria.fr/inria-00394479>
- [Tian et al., 2006] X. Tian, G. Han, M. Chen and Z. Situ. *Skeleton-Based Surface Reconstruction for Visualizing Plant Roots*. In *Artificial Reality and Telexistence—Workshops, 2006. ICAT '06. 16th International Conference on*, pages 328–332, 2006.
- [Toldo and Fusiello, 2008] R. Toldo and A. Fusiello. *Robust Multiple Structures Estimation with J-Linkage*. In *Proceedings of the 10th European Conference on Computer Vision: Part I, ECCV '08*, pages 537–547, Springer-Verlag, Berlin, Heidelberg, 2008, ISBN 978-3-540-88681-5.
URL http://dx.doi.org/10.1007/978-3-540-88682-2_41
- [Tomycz et al., 2011] L. Tomycz, N. Bansal, C. Hawley, T. Goddard, M. Ayad and R. Mericle. *"Real-world" comparison of non-invasive imaging to conventional catheter angiography in the diagnosis of cerebral aneurysms*. Surg Neurol Int, vol. 2, 2011.
- [Tsingos et al., 1995] N. Tsingos, E. Bittar and M.-P. Cani. *Implicit Surfaces for Semi-Automatic Medical Organ Reconstruction*. In *Computer Graphics Internat. (CGI'95)*, pages 3–15, 1995.
URL <http://hal.inria.fr/inria-00537538>
- [Turk and O'brien, 2002] G. Turk and J. F. O'brien. *Modelling with Implicit Surfaces That Interpolate*. ACM Trans. Graph., vol. 21, no. 4, pages 855–873, 2002, ISSN 0730-0301.
URL <http://doi.acm.org/10.1145/571647.571650>

- [Tyrrell et al., 2007] J. Tyrrell, E. di Tomaso, D. Fuja et al.. *Robust 3-D modeling of vasculature imagery using superellipsoids*. IEEE Trans. Med. Imag., vol. 26, no. 2, pages 223–237, 2007.
- [van Rooij et al., 2008] W. van Rooij, M. Sprengers, A. de Gast, J. Peluso and M. Sluzewski. *3D rotational angiography: the new gold standard in the detection of additional intracranial aneurysms*. AJNR Am J Neuroradiol, vol. 29, no. 5, pages 976–9, 2008.
- [Vosselman et al., 2004] G. Vosselman, B. G. Gorte, G. Sithole and T. Rabbani. *Recognising structure in laser scanner point clouds*. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 46, no. 8, pages 33–38, 2004.
- [Wang et al., 2012] C. Wang, R. Moreno and Ö. Smedby. *Vessel Segmentation Using Implicit Model-Guided Level Sets*. MICCAI Workshop "3D Cardiovascular Imaging: a MICCAI segmentation Challenge", 2012.
URL <http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-89844>
- [Wang et al., 2011] Y. Wang, A. Narayanaswamy and B. Roysam. *Novel 4-D Open-Curve Active Contour and curve completion approach for automated tree structure extraction*. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1105–1112, 2011, ISSN 1063-6919.
- [Wei et al., 2012] Y. Wei, S. Cotin, J. Dequidt, C. Duriez, J. Allard and E. Kerrien. *A (Near) Real-Time Simulation Method of Aneurysm Coil Embolization*. In *Aneurysm*, edited by Y. Murai, pages 223–248, InTech, 2012, ISBN 978-953-51-0730-9.
URL <http://hal.inria.fr/hal-00736865>
- [Wesarg and Firlé, 2004] S. Wesarg and E. A. Firlé. *Segmentation of Vessels: The Corkscrew Algorithm*. In *Medical Imaging Symposium 2004. Volume 5370 of Proc. of SPIE. (2004) 1609-1620*, pages 1609–1620, 2004.
- [Williams and Shah, 1992] D. J. Williams and M. Shah. *A Fast Algorithm for Active Contours and Curvature Estimation*. CVGIP: Image Underst., vol. 55, no. 1, pages 14–26, 1992, ISSN 1049-9660.
URL [http://dx.doi.org/10.1016/1049-9660\(92\)90003-L](http://dx.doi.org/10.1016/1049-9660(92)90003-L)
- [Wink et al., 2000] O. Wink, W. J. Niessen and M. A. Viergever. *Fast Delineation and Visualization of Vessels in 3D Angiographic Images*. IEEE Trans. Med. Imaging, vol. 19, no. 4, pages 337–346, 2000.
- [Witkin and Heckbert, 1994] A. P. Witkin and P. S. Heckbert. *Using particles to sample and control implicit surfaces*. In *SIGGRAPH*, pages 269–277, 1994.
URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.27.2922>
- [Wong and Chung, 2007] W. Wong and A. Chung. *Probabilistic vessel axis tracing and its application to vessel segmentation with stream surfaces and minimum cost paths*. MedIA, vol. 11, pages 567–587, 2007.
URL http://www.cs.ust.hk/faculty/achung/media07_wong_chung.pdf

- [Worz and Rohr, 2007] S. . Worz and K. . Rohr. *Segmentation and Quantification of Human Vessels Using a 3-D Cylindrical Intensity Model*. *Trans. Img. Proc.*, vol. 16, no. 8, pages 1994–2004, 2007, ISSN 1057-7149.
URL <http://dx.doi.org/10.1109/TIP.2007.901204>
- [Wu et al., 2010] J. Wu, M. Wei, Y. Li, X. Ma, F. Jia and Q. Hu. *Scale-adaptive surface modeling of vascular structures*. *Biomed Eng Online*, vol. 9, page 75, 2010, ISSN 1475-925X.
- [Wu et al., 2013] J. Wu, Q. Hu and X. Ma. *Comparative study of surface modeling methods for vascular structures*. *Comp. Med. Imag. and Graph.*, vol. 37, no. 1, pages 4–14, 2013.
- [Wu et al., 2005] X. Wu, V. Pegoraro, V. Luboz, P. F. Neumann, R. Bardsley, S. Dawson and S. Cotin. *New approaches to computer-based interventional neuroradiology training*. *Proceedings of the Medicine Meet Virtual Reality (MMVR 13) conference*, pages 602–607, 2005.
- [Wu et al., 2007] X. Wu, J. Allard and S. Cotin. *Real-Time Modeling of Vascular Flow for Angiography Simulation*. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2007*, vol. 4791 of *Lecture Notes in Computer Science*, edited by N. Ayache, S. Ourselin and A. Maeder, pages 557–565, Springer Berlin Heidelberg, 2007, ISBN 978-3-540-75756-6.
URL http://dx.doi.org/10.1007/978-3-540-75757-3_68
- [Wu et al., 2011] X. Wu, V. Luboz, K. Krissian, S. Cotin and S. Dawson. *Segmentation and reconstruction of vascular structures for 3D real-time simulation*. *MEDIA*, vol. 15, no. 1, pages 22 – 34, 2011, ISSN 1361-8415.
URL <http://www.sciencedirect.com/science/article/pii/S1361841510000691>
- [Xie and Mirmehdi, 2011] X. Xie and M. Mirmehdi. *Radial Basis Function Based Level Set Interpolation and Evolution for Deformable Modelling*. *Image Vision Comput.*, vol. 29, no. 2-3, pages 167–177, 2011, ISSN 0262-8856.
URL <http://dx.doi.org/10.1016/j.imavis.2010.08.011>
- [Xu et al., 2000] C. Xu, J. Yezzi, A. and J. Prince. *On the relationship between parametric and geometric active contours*. In *Signals, Systems and Computers, 2000. Conference Record of the Thirty-Fourth Asilomar Conference on*, vol. 1, pages 483–489 vol.1, 2000, ISSN 1058-6393.
- [Yan and Kassim, 2006] P. Yan and A. A. Kassim. *Segmentation of volumetric {MRA} images by using capillary active contour*. *Medical Image Analysis*, vol. 10, no. 3, pages 317 – 329, 2006, ISSN 1361-8415, special Issue on The Second International Workshop on Biomedical Image Registration (WBIR'03).
URL <http://www.sciencedirect.com/science/article/pii/S1361841506000028>
- [Yedidya and Hartley, 2008] T. Yedidya and R. Hartley. *Tracking of Blood Vessels in Retinal Images Using Kalman Filter*. In *Digital Image Computing: Techniques and Applications (DICTA), 2008*, pages 52–58, 2008.

- [Yoon and Kim, 2006] S.-H. Yoon and M.-S. Kim. *Sweep-based Freeform Deformations*. Comput. Graph. Forum, vol. 25, no. 3, pages 487–496, 2006.
- [Yureidini et al., 2011a] A. Yureidini, J. Dequidt, E. Kerrien, C. Duriez and S. Cotin. *Computer-based simulation for the endovascular treatment of intracranial aneurysms*. In *LIVIM Imaging Workshop*, Strasbourg, France, 2011a.
URL <http://hal.inria.fr/hal-00641990>
- [Yureidini et al., 2011b] A. Yureidini, E. Kerrien and S. Cotin. *Reconstruction robuste des vaisseaux sanguins par surfaces implicites locales*. In *Orasis*, Praz-sur-Arly, France, 2011b.
URL <http://hal.inria.fr/inria-00579814>
- [Yureidini et al., 2012a] A. Yureidini, E. Kerrien and S. Cotin. *Robust RANSAC-based blood vessel segmentation*. In *SPIE Medical Imaging*, vol. 8314, edited by D. R. Haynor and S. Ourselin, page 8314M, SPIE Press, San Diego, CA, États-Unis, 2012a.
URL <http://hal.inria.fr/hal-00642003>
- [Yureidini et al., 2012b] A. Yureidini, E. Kerrien, J. Dequidt, C. Duriez and S. Cotin. *Local implicit modeling of blood vessels for interactive simulation*. In *MICCAI - 15th International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 7510 of *Lecture Notes in Computer Science*, edited by N. Ayache, H. Delingette, P. Golland and K. Moria, pages 553–560, Springer, Nice, France, 2012b.
URL <http://hal.inria.fr/hal-00741307>
- [Zambal et al., 2008] S. Zambal, J. Hladuvka, A. Kanitsar and K. Bühler. *Shape and Appearance Models for Automatic Coronary Artery Tracking*. 2008.
- [Zana and Klein, 2001] F. Zana and J. C. Klein. *Segmentation of vessel-like patterns using mathematical morphology and curvature evaluation*. Trans. Img. Proc., vol. 10, no. 7, pages 1010–1019, 2001, ISSN 1057-7149.
URL <http://dx.doi.org/10.1109/83.931095>
- [Zhang, 1995] Z. Zhang. *Parameter Estimation Techniques: A Tutorial with Application to Conic Fitting*. Rapport de recherche RR-2676, INRIA, 1995.
URL <http://hal.inria.fr/inria-00074015>
- [Zuliani et al., 2005] M. Zuliani, C. Kenney and B. Manjunath. *The multiRANSAC algorithm and its application to detect planar homographies*. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 3, pages III–153–6, 2005.

Abbreviations

Symbols | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **K** | **L** | **M** | **N** | **R** | **S** | **T** | **V** | **W**

Symbols

1D one-Dimension 40, 41, 45, 66, 67

2D two-Dimensional 21, 30, 32, 33, 40, 41, 66, 67, 75

3D three-Dimensional 8, 10, 14, 15, 17, 20–22, 25–28, 30, 31, 38, 40, 41, 47, 49, 50, 56, 57, 66, 69, 74, 82, 91, 92, 99, 100, 117, 124, 125, 136, 139

A

a.k.a also known as 4, 90

AABB Axis Aligned Bounding Box 121, 127, 137

Ac Accuracy 52, 55

AM Alvarez's Method 40, 41, 44, 45

ASSD Average Symmetric Surface Distance 53–57, 59, 77, 79–82

AVM Arteriovenous Malformations 4, 14

B

BM Blobby Model 11, 63, 66, 67, 70–73, 75, 76, 78–82, 84, 85, 87–91, 93–104, 106, 114, 116, 130, 136–140, 145, 146

BS Bi-cubic Spline 40, 41, 44, 45

BSP Bounded Spherical Projection 28

BVH Bounding Volume Hierarchies 18, 21, 65, 114, 116, 121, 130

C

C Cubic 40, 41

CathI Catheter Instruction System 6, 7

CD Central Differences 40, 41, 44, 45

CDV Cardiovascular diseases 3

CFD Computational Fluid Dynamics 105, 139

CL Common path Length 54, 56, 57

CTA Computed Tomography Angiography 21, 22, 38, 57, 58

D

DoF Degrees of Freedom 109, 111, 112

DSA Digital Subtraction Angiography 21

E

EVE Real-time Endovascular Simulator 6, 7

EVE Endovascular Embolization Simulator 6, 7

F

FD Forward Difference 40, 41, 44, 45

G

GD Geometric Distance 94–99, 104

GDC Guglielmi Detachable Coil 4

GPU Graphics Processing Unit 17, 140

GUI Graphical User Interface 141

H

HD Hausdorff Distance 77, 79–82

I

IC Implicit Contour 77, 79–82

ICTS Interventional Cardiology Training System 6, 7

IN Interventional Neuroradiology 4–6, 8, 9, 13, 14, 18, 136, 138

K

KV Kissing Vessel 15, 18, 20, 21, 26, 29, 59, 64, 65, 90, 102, 104, 114, 115, 136

L

L Linear 40, 41, 44

LCP Linear Complementary Problem 119

LIM Local Implicit Modeling 63–65, 74, 86, 89, 91, 93–95, 98, 99, 102–104, 106, 115, 122, 124–130, 135–140

M

MHT Multiple Hypothesis Tracking 10, 25, 47–50, 52–61, 136

MPU Multi-level Partion of Unity 18

MRA Magnetic Resonance Angiography 21, 22, 25, 38, 57, 58, 62

N

NLCP Non-Linear Complementary Problem 119

R

RA Rotational Angiography 10, 15, 21, 22, 25–28, 38, 41, 49, 50, 56, 57, 66, 74, 82, 91, 92, 99, 100, 124, 125, 136

RANSAC RANdom SAmples Consensus 25–27, 29, 31, 32, 42, 57, 136, 138

RBF Radial Basis Function 98, 138, 139

RBT RANSAC-Based Tracking 26–29, 32–42, 47–62, 64, 74, 75, 77, 86, 91, 92, 103, 115, 124, 125, 135–138, 141–143

RMS Root Mean Square 42, 44, 126, 127

RT Real-Time 5–11, 14, 129, 136, 140

RTrackingIT RANSAC-based Tracking Interface 141–143

S

SAH Subarachnoid Haemorrhage 3, 4

SOFA Simulation Open Framework Architecture 6, 7, 108, 109, 130, 137

SR Success Rate 25, 52–54, 57

T

TaGD Taubin's approximate Geometric Distance 94, 96, 97, 99, 104, 120, 126, 127

TL Tracked Length 52, 54, 56, 57

TM Triangular Mesh 108, 114–116, 121, 123–130, 137

V

VIST Vascular Intervention Simulation Trainer 6–8

VSP Vascular Surgical Platform 6, 7

W

w.r.t with respect to 20, 32, 33, 44, 51, 53, 57, 66, 68–70, 79, 80, 84, 86, 95, 101, 102, 111, 112, 115, 116, 136, 139, 145