From Medical Images to Fast Computational Models of Heart Electromechanics: An Integrated Framework towards Clinical Use

Oliver Zettinig[1,2], Tommaso Mansi[1], Bogdan Georgescu[1], Saikiran Rapaka[1], Ali Kamen[1], Jan Haas[3], Karen S. Frese[3], Farbod Sedaghat-Hamedani[3], Elham Kayvanpour[3], Ali Amr[3], Stefan Hardt[3], Derliz Mereles[3], Henning Steen[3], Andreas Keller[4,5], Hugo A. Katus[3], Benjamin Meder[3], Nassir Navab[2], and Dorin Comaniciu[1]

[1] Siemens Corporation, Corporate Technology, Imaging and Computer Vision, Princeton, NJ, USA
[2] Computer Aided Medical Procedures, Technische Universität München, Germany
[3] University Hospital Heidelberg, Department of Internal Medicine III - Cardiology, Angiology and Pneumology, Heidelberg, Germany
[4] Siemens AG, Healthcare Strategy, Erlangen, Germany
[5] Department of Human Genetics, Saarland University, Germany

Abstract. With the recent advances in computational power, realistic modeling of heart function within a clinical environment has come into reach. Yet, current modeling frameworks either lack overall completeness or computational performance, and their integration with clinical imaging and data is still tedious. In this paper, we propose an integrated framework to model heart electromechanics from clinical and imaging data, which is fast enough to be embedded in a clinical setting. More precisely, we introduce data-driven techniques for cardiac anatomy estimation and couple them with an efficient GPU (graphics processing unit) implementation of the orthotropic Holzapfel-Ogden model of myocardium tissue, a GPU implementation of a mono-domain electrophysiology model based on the Lattice-Boltzmann method, and a novel method to correctly capture motion during isovolumetric phases. Benchmark experiments conducted on patient data showed that the computation of a whole heart cycle including electrophysiology and biomechanics with mesh resolutions of around 70k elements takes on average 1min 10s on a standard desktop machine (Intel Xeon 2.4GHz, NVIDIA GeForce GTX 580). We were able to compute electrophysiology up to 40.5$\times$ faster and biomechanics up to 15.2$\times$ faster than with prior CPU-based approaches, which breaks ground towards model-based therapy planning.

1 Introduction

Cardiovascular diseases are a burden with high social, economic and healthcare impact. For instance, heart failure alone affects an estimated 2% of adults in the Western countries [8]. Dilated cardiomyopathy (DCM) is one of the most common causes of heart failure and the leading indication for heart transplantation in younger adults. Diagnosis and treatment of DCM is challenging and it is nowadays pivotal to elucidate the individual causes and disease stages of DCM to allow for appropriate risk stratification, outcome prediction, therapy planing and monitoring. Computational tools could assist physicians in multiple steps of the clinical workflow and could impact on future therapeutic strategies.

In the past decades, detailed computational models of heart electromechanics have been developed. On the one hand, computational models of cardiac electrophysiology (EP) have been proposed, from highly detailed ionic models to simplified Eikonal-based methods, see [3] for a review. Recent advances in numerical methods such as Lattice-Boltzmann algorithms for cardiac electrophysiology [13] have the potential to enable near real-time EP computation, thus paving the way to clinical applications. On the other hand, models of active and passive tissue behavior couple the electrophysiological signal with tissue biomechanics to compute cardiac motion. These models are traditionally solved using finite-element methods (FEM), but novel approaches for fast simulations are being proposed. Mass-spring methods offer real-time performance but fail to accurately capture myocardium tissue material properties [11]. Apart from transverse isotropic linear elasticity models [14], which become inaccurate for large deformations, the total Lagrangian explicit dynamics algorithm (TLED) [9] has drawn the community's attention, with efficient GPU implementations for fast biomechanical simulations [15]. However, at the best of our knowledge, efficient and accurate EP models have not yet been coupled with fast biomechanical frameworks for patient-specific heart modeling, and there is a lack of integration between image analysis and modeling tools into a unified framework.

The overarching goal of this work is the development of methods for the estimation of personalized models of cardiac function in a clinical setting, starting with the introduction of an integrated but modular framework for the computation of patient-specific cardiac electromechanics. In particular, we propose: 1) an integrated pipeline for anatomical model generation, 2) a method to compute cardiac motion during the isovolumetric phases that is computationally efficient and 3) a GPGPU (general purpose graphics processing units) framework to solve cardiac electrophysiology and biomechanics efficiently.

2 Proposed Electromechanical Model

Four components constitute our complete heart model: After generating a patient-specific anatomical model from imaging data (Sec. 2.1), we compute cardiac electrophysiology over the cardiac cycle using the end-diastasis geometry (Sec. 2.2) and couple the calculated potentials with a cardiac biomechanics model (Sec. 2.3). A lumped model of cardiac hemodynamics is finally employed to compute biomechanical boundary conditions (Sec. 2.4).

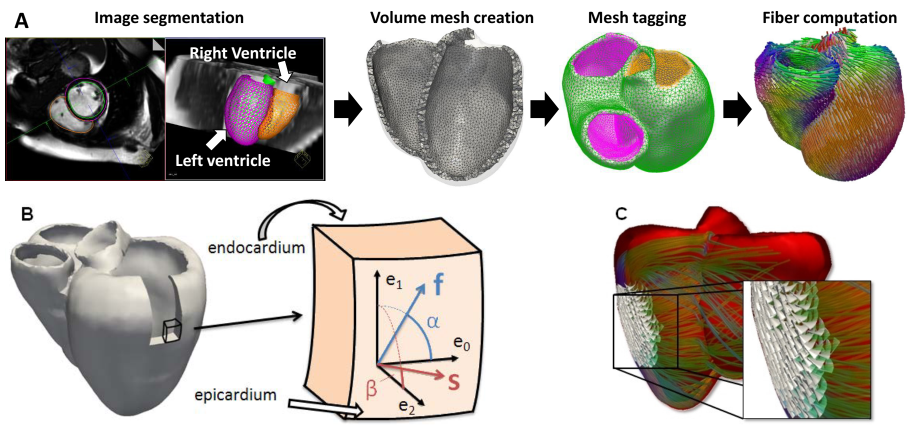


Fig. 1. A: Different steps of our automatic pipeline for the estimation of patient-specific anatomical models. See text for details. **B**: Definition of fiber directions **f** and sheet directions **s** in terms of angles α and β ($\mathbf{e_0}$ circumferential axis, $\mathbf{e_1}$ longitudinal axis, $\mathbf{e_2}$ radial axis). **C**: Fiber and sheet model computed on a patient-specific anatomy.

2.1 Integrated Pipeline for Cardiac Anatomy Modeling

Heart Morphology is automatically estimated, under expert guidance, from magnetic resonance images (MRI) using a database-guided machine-learning framework [16] (Fig. 1A). The algorithm yields triangulations of the endocardia and the epicardium, which are then fused together to form a closed surface of the biventricular myocardium, serving as basis for tetrahedral volume generation[1]. The facets are automatically tagged with labels according to the point-to-mesh distance between the volume mesh and the detected triangulations.

Rule-Based Model of Myocardium Fibers and Fiber Sheets. We calculate myocardium fiber architecture based on patient heart morphology as follows. Below the basal plane, the fiber elevation angle α (angle with respect to the short axis plane) and the sheet direction angle β (angle with respect to the outward radial axis) vary linearly from epi- to endocardium: $\alpha_{epi} = -70°$, $\alpha_{endo} = +70°$, $\beta_{epi} = +45°$, $\beta_{endo} = -45°$ [2] (Fig. 1B). Based on the geodesic distance to the endocardia (d_{endo}) and epicardia (d_{epi}), both angles are computed for each point of the volume mesh: $\alpha = (d_{epi}\,\alpha_{endo} + d_{endo}\,\alpha_{epi})/(d_{endo} + d_{epi})$, likewise for β. We then fix the fiber and sheet orientations around each valve (fibers are longitudinal around the aortic valve, tangential elsewhere, sheet normals are oriented towards the barycenter of the valves) and interpolate the local orthonormal basis from the basal plane to the valve, first by following the myocardium surface, then transmurally [10]. For orthonormality preservation, the interpolation is performed using the Log-Euclidean framework [1]. Fig. 1**AC** show the generated fiber and sheet directions in a DCM patient.

[1] www.cgal.org

2.2 Cardiac Electrophysiology Model

The Mitchell-Schaeffer mono-domain EP model is solved using the recently proposed Lattice-Boltzmann LBM-EP method. The computational domain in LBM-EP is a 3D Cartesian grid, where a level-set representation of the patient anatomy is used to ensure correct boundary conditions. Fiber architecture is rasterized on the grid to cope with tissue anisotropy (see [13] for details). In this work, cardiac EP over the heart cycle is computed using the end-diastasis geometry and mapped back to the volume mesh for the biomechanics computation.

2.3 Cardiac Biomechanics

Cardiac biomechanics is computed by solving the dynamics equation $\mathrm{M}\ddot{\mathbf{u}}+\mathrm{C}\dot{\mathbf{u}}+\mathrm{K}\mathbf{u} = \mathbf{F}_a + \mathbf{F}_p + \mathbf{F}_b$, where $\ddot{\mathbf{u}}$, $\dot{\mathbf{u}}$ and $\mathbf{u}$ gather the accelerations, velocities and displacements of the mesh nodes, M is the mass matrix, K the internal elastic stiffness matrix and C the (Rayleigh) damping matrix. As described below, force vectors $\mathbf{F}_a$, $\mathbf{F}_p$ and $\mathbf{F}_b$ model active stress, ventricular pressure and mechanical boundary conditions respectively. The dynamics system is solved using the finite-element method (FEM) on linear tetrahedra meshes with an Euler implicit time-stepping. The resulting linear system $\Xi\mathbf{u} = \mathbf{F}$ is solved using the conjugate gradient method.

Passive Stress. The orthotropic Holzapfel-Ogden (HO) model [4] is used to compute tissue biomechanics. The strain-stress energy writes:

$$\begin{aligned}\Psi = \frac{a}{2b} exp\left[b(I_1-3)\right] + \sum_{i=f,s} \mathcal{H}_\delta(I_{4i}-1)\frac{a_i}{2b_i}\left\{exp\left[b_i(I_{4i}-1)^2\right]-1\right\} \\ + \frac{a_{fs}}{2b_{fs}}\left[exp(b_{fs}I_{8fs}^2)-1\right] + D_1(J-1)^2\end{aligned} \tag{1}$$

where the a_k's and b_k's are material constants, J is the Jacobian determinant of the deformation gradient F, $J = \det(\mathrm{F})$, D_1 is a parameter equivalent to the bulk modulus and the I_k's are invariants of the right Cauchy-Green deformation tensor $\mathrm{C} = \mathrm{F}^T\mathrm{F}$. In the previous equation, the subscripts f and s denote fiber and sheet directions. $\mathcal{H}_\delta(\cdot)$ is the logistic function, a smooth approximation of the Heaviside step function employed here for increased numerical stability.

For efficient computation, Eq. (1) is expressed according to the Multiplicative Jacobian Energy Decomposition (MJED) formulation [7]: $\Psi = \sum_k f^k(J)g^k(\tilde{I})$, where $\tilde{I} = [I_1, I_{4f}, I_{4s}, I_{8fs}]$. The nodal force $\mathbf{F}_i$ and the edge stiffness matrices K_{ij} are then defined by $\mathbf{F}_i = -(\partial\Psi/\partial\mathbf{x}_i)^T$ and $\mathrm{K}_{ij} = \partial^2\Psi/(\partial\mathbf{x}_i\partial\mathbf{x}_j)$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are two connected nodes. Closed form expressions of $\partial f^k(J)/\partial\mathbf{x}_i$, which do not involve calculating C^{-1}, are available [7]. Deriving the $g^k(\tilde{I})$ requires calculating their first and second derivative with respect to C, which can be easily calculated through the identities (6.1), (6.4) and (6.10) in [4].

Active Stress. The active contraction forces $\mathbf{F}_a$ are computed using the model proposed in [14], which expresses the active Cauchy stress tensor σ in terms of the action potential. Depolarization and repolarization times, triggering myocyte contraction and relaxation, are obtained from the EP model. The main parameters of that model are σ_0, the maximum contraction that can be reached by a cell, k_{ATP} and k_{RS}, the ATP binding and release rates.

Mechanical Boundary Conditions. Two mechanical boundary conditions, accounting for force vectors $\mathbf{F}_b$, are considered. First, the effect of arteries and atria on ventricular motion is modeled by connecting the valve plane vertices (automatically defined as endocardium border vertices) to springs with anisotropic stiffness to allow longitudinal motion. Second, a pericardium constraint is added through the contact-based method proposed in [6].

2.4 Cardiac Hemodynamics

We mimic the four cardiac phases filling, isovolumetric contraction, ejection and isovolumetric relaxation by alternating endocardial boundary conditions. Ventricular pressure p is added to the dynamics system using the nodal forces $\mathbf{F}_p = p\,\mathbf{N}$, where $\mathbf{N}$ is the vector gathering the lumped area vectors $\mathbf{n}\,dS$ of the endocardial surface. During filling (ventricular blood flow $q > 0$) and ejection (initiated when p reaches the arterial pressure), we directly apply the respective atrial or arterial pressure, computed using a time-varying elastance model and a 3-element Windkessel model [5] respectively. In between, when both valves are closed (flow is reverted), we enable the following isovolumetric constraint.

To keep the ventricular volume V constant during isovolumetric phases, we propose an efficient projection-prediction method that aims at finding the pressure $\tilde{p}(t)$ which ensures $V(t+dt) = V_0$, dt is the time step. To this end, we first solve the dynamic system without constraint, computing new, unconstrained vertex positions $\hat{\mathbf{x}}(t+dt)$. Thereafter, we reformulate the system including an unknown corrective pressure $\lambda(t)$: $\Xi(\mathbf{x}(t+dt)-\mathbf{x}_0) = \mathbf{F}+\lambda\mathbf{N}$. Solving the system at $t+dt$ yields $(\mathbf{x}(t+dt)-\mathbf{x}_0) = (\hat{\mathbf{x}}(t+dt)-\mathbf{x}_0)+\lambda\Xi^{-1}\mathbf{N}$. The constrained system thus writes $\mathbf{x}(t+dt) = \hat{\mathbf{x}}(t+dt)+\lambda(t)\Xi^{-1}\mathbf{N}$ such that $V(t+dt) = V_0$.

As shown in [12], the Lagrangian coefficient λ is computed by solving a third-order polynomial. The vertices are then projected by applying displacements $u_p(t) = \lambda(t)\Xi^{-1}\mathbf{N}$. In a final step, we compute the corrected pressure $\tilde{p}(t) = p(t)+\lambda(t)$ and predict the pressure at the next time step by utilizing a second-order Taylor expansion scheme, $p(t+dt) = \tilde{p}(t) + dtd\tilde{p}/dt + dt^2 0.5 d^2\tilde{p}/dt^2$.

3 Fast GPU Implementation

Since the computational limitations of our model are predominantely present in the electrophysiological and biomechanical sub-models, we focused our efforts on the parallelization of these components, implemented using NVIDIA CUDA. In the following, we did not impose any simplifications of the mathematical formulations but rather aimed at reorganizing the underlying algorithm to fully exploit the parallel infrastructure.

Solving the dynamics equation using FEM involves computing nodal forces by accumulating the contributions of all elements sharing each node, e.g contributions of surrounding triangles for nodal pressure forces. In this paper, we propose an efficient adaptation of the parallel implementation strategy proposed in [15] to circumvent the inability of global random access accumulations on GPU devices. The key element of our method is the precomputed integer look-up texture *mapElements* of size $2 \times N_n \times V_{max}$, where valence V_{max} is the maximum number of elements connected to a node and N_n the number of nodes. It stores pairs (j, k) for any given node, k holding its local index within adjacent element j (e.g., $k \in [1, 4]$ for tetrahedra). A kernel *compute* is invoked across the N_e elements to perform the element-wise computation, storing its results into separate floating point textures T^k of size N_e each. A second kernel *accumulate* is invoked across N_n threads, looping over all V pairs (j, k) corresponding to the respective node and accumulating the element-wise contributions stored at the j-th positions of textures T^k. In contrast to [15], we need to manage only one texture instead of two, since indexing of *mapElement* only requires V_{max} and not the actual (non-constant) nodal valence, resulting in simpler code and additional speed-up through alignment with *accumulate* kernel threads. The higher memory demand is negligible in the light of current GPU memory sizes; for a mesh with 200k elements, $V_{max} = 42$, and $V_{avg} = 16$ (average valence), our look-up texture would require ca. 64.1 MB instead of about 24.4 MB.

Our implementation of the HO model is formulated in a total Lagrangian framework, allowing for the precomputation of variables and the parallel execution of nearly all calculations. From the rest state, we precompute tetrahedron shape vectors $\mathbf{D}_i$ as the cross product of two opposing edges respectively [7], and the tensors of fiber, sheet, and cross-sheet directions ($\mathbf{f} \otimes \mathbf{f}$, $\mathbf{s} \otimes \mathbf{s}$, $\mathbf{f} \otimes \mathbf{s} + \mathbf{s} \otimes \mathbf{f}$). At each time step, the deformation gradient F is written as $\sum_{i=0}^{4} \mathbf{x}_i \, \mathbf{D}_i$, allowing to compute the right Cauchy-Green deformation tensor C and the invariants $I_1, I_{4f}, I_{4s}, I_{8fs}$ [4]. By using the definitions in Sec. 2.3 and our general GPU strategy explained above, we compute forces (tetrahedron/vertex look-up table) and edge stiffness matrices (tetrahedron/edge look-up table) as required by the implicit integration scheme. Our general strategy was also applied for fast computation of active stress, pressure forces and mechanical boundary conditions.

Finally, since Lattice-Boltzmann methods are inherently node-wise, their implementation on GPGPU architectures is straightforward. A kernel that handles the stream and collide procedures of the LBM algorithm is employed to compute the potential at each node of the Cartesian grid.

4 Experiments and Results

4.1 Benchmarking of Computational Performance

To evaluate the performance of our framework, we ran the entire simulation on one representative patient case with a different number of mesh elements.

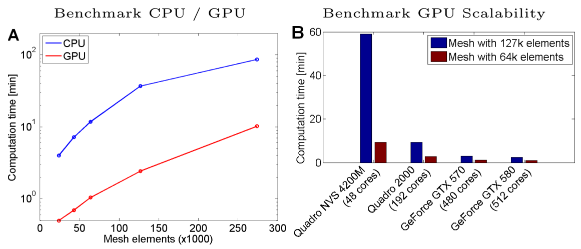


Fig. 2. Whole-heart simulation benchmarks. **A**: Comparison between CPU and GPU runtimes for meshes with various sizes of the same patient (EP computed on GPU for both experiments) **B**: GPU runtimes for two different meshes on various graphics cards, showing our method scales well with the number of CUDA cores.

Table 1. Comparison of measured (EF_m) and computed (EF_c) ejection fractions, and measured (SV_m) and computed (SV_c) stroke volumes (in ml) for 5 cases

Pat.	EF_m	EF_c	SV_m	SV_c
1	41%	36%	106	91
2	14%	12%	40	33
3	27%	23%	67	56
4	15%	17%	71	80
5	34%	30%	85	66

Fig. 2A shows the computation times for meshes with 24k, 43k, 64k, 127k and 274k tetrahedra. The time step of our Euler implicit scheme was set to $1\,ms$, we computed one full heart cycle of 0.8 s with a numerical threshold of $10^{-2}\,mm$ for the conjugate gradient solver. The general dynamics system parameters include a mass density of $\rho = 1.07\,g/ml$ and Rayleigh damping coefficients of 10^4 for mass and 10^{-1} for stiffness. Passive tissue parameters were set as in [4], and σ_0 was $150\,kPa$. We conducted our experiments on a system with an 16-core Intel Xeon 64-bit CPU at 2.4 GHz and an NVIDIA GeForce GTX 580 graphics card.

For the anatomical model construction, detection and tracking was performed in less than $4\,s$ per frame. Preparation times for a mesh with medium resolution (64k) amounted to $64.4\,s$ for tetrahedral mesh generation and $16.8\,s$ for anatomy model computation. It should be noted that the anatomical model is built only once. We are able to calculate the electrophysiology for one full heart cycle in around $3\,s$ on a Cartesian grid of $1.5 \times 1.5 \times 1.5\,mm$, which is up to $40.5\times$ faster than a CPU implementation of the same algorithm (in total between two and three orders of magnitude faster than current FEM-based approaches with meshes of comparable resolution). The most significant runtimes correspond to the biomechanics component of our framework. Here, we gained a mean speed-up factor of 10.6 (std. dev. 2.8) for different mesh resolutions. For the mesh with 64k elements, the simulation only required $62\,s$, and even for the mesh with highest resolution (274k), an entire heart cycle could be computed in 10 min and 12 seconds. The overall runtime from image to model was 2 min and 31 seconds for the mesh with 64k elements (Fig. 2A). We also conducted scalability benchmarks on graphics cards with 48, 192, 480 and 512 CUDA cores with two different meshes (Fig. 2B). Excluding the experiment on our mobile system (Quadro NVS 4200M) due to its different architecture, our results indicate that the whole-heart computation time is linearly dependent on the number of available cores, with higher scalability as the mesh size increases.

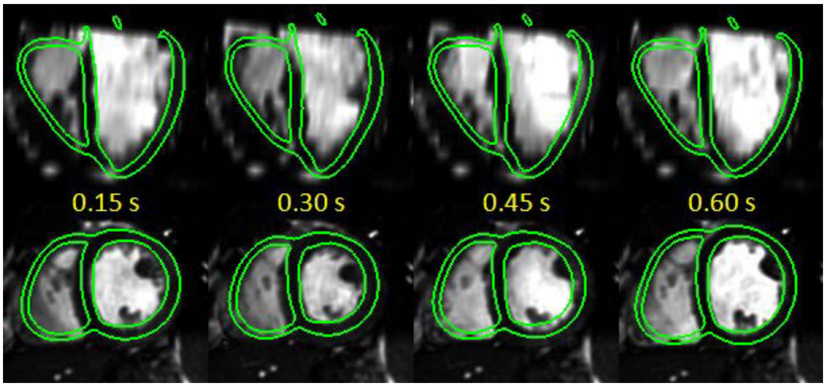
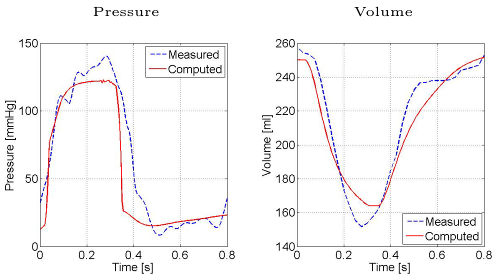


Fig. 3. Personalization results for the patient with largest contraction. Long-axis (top) and short-axis (bottom) slices showing MRI images and personalized model at various timesteps throughout one heart cycle.

Fig. 4. Personalization results for the patient with largest contraction. **Left panel** shows pressure curve and **right panel** volume curve for the left ventricle, indicating a good match.



4.2 Model Personalization in Dilated Cardiomyopathy

We illustrate our framework on five anonymized datasets of DCM patients with subnormal to severely abnormal ejection fractions (Tab. 1). The datasets consisted of cine MRI sequences and heart catheter pressure measurements of the left ventricle and the aorta. We first manually estimated the Windkessel parameters using the aortic pressure measurements and the aortic flow obtained from the images by matching pressure curves. Thereafter, we manually optimized parameters of the biomechanical model (active stress σ_0, boundary conditions) to match the cardiac motion and ejection fraction as shown in the images. Tissue parameters were kept to their default values [4]. As one can see from Fig. 3, we were successful in modeling realistic cardiac motion. The volume and pressure curves generated by our model (Fig. 4) qualitatively represented the measured values. Table 1 reports computed and measured ejection fractions and stroke volumes for all five patients, showing promising agreement. On average, computation time was 70 seconds to simulate a full heart cycle (mesh resolution around 70k), which is fast enough to allow user interaction with the model.

5 Conclusion and Future Work

In this paper, we have presented an integrated patient-specific framework of computational heart electromechanics that is fast enough to be applied in clinical routine. Incorporating a very efficient LBM implementation of cardiac EP with the state-of-the-art Holzapfel-Ogden model for passive biomechanics, our framework yields medically expedient results and becomes applicable for clinical therapy planning due to the exploitation of massively parallel GPU architectures. Our framework has important potential applications, for instance it may enable physicians to plan cardiac interventions and compute predictors of therapy outcome *in silico*. While parameter adjustment and patient personalization is still done manually, more automatic methods are being investigated. Future work also includes the integration of length-dependent active forces and the development of efficient strategies for strong electromechanical coupling.

References

1. Arsigny, V., Fillard, P., Pennec, X., Ayache, N.: Log-Euclidean metrics for fast and simple calculus on diffusion tensors. MRM 56(2), 411–421 (2006)
2. Bayer, J., Blake, R., Plank, G., Trayanova, N.: A novel rule-based algorithm for assigning myocardial fiber orientation to computational heart models. ABME 40, 2243–2254 (2012)
3. Clayton, R., Bernus, O., Cherry, E., Dierckx, H., Fenton, F., Mirabella, L., Panfilov, A., Sachse, F., Seemann, G., Zhang, H.: Models of cardiac tissue electrophysiology: Progress, challenges and open questions. PBMB 104(1-3), 22 (2011)
4. Holzapfel, G.A., Ogden, R.W.: Constitutive modelling of passive myocardium: a structurally based framework for material characterization. Phil. Trans. R. Soc. A 367(1902), 3445–3475 (2009)
5. Kerckhoffs, R., Neal, M., Gu, Q., Bassingthwaighte, J., Omens, J., McCulloch, A.: Coupling of a 3D finite element model of cardiac ventricular mechanics to lumped systems models of the systemic and pulmonic circulation. ABME 35, 1–18 (2007)
6. Mansi, T.: Image-Based Physiological and Statistical Models of the Heart, Application to Tetralogy of Fallot. Ph.D. thesis, Mines ParisTech (2010)
7. Marchesseau, S., Heimann, T., Chatelin, S., Willinger, R., Delingette, H.: Fast porous visco-hyperelastic soft tissue model for surgery simulation: Application to liver surgery. PBMB 103, 185–196 (2010)
8. McMurray, J., et al.: ESC guidelines for the diagnosis and treatment of acute and chronic heart failure 2012. European Heart Journal 33(14), 1787–1847 (2012)
9. Miller, K., Joldes, G., Lance, D., Wittek, A.: Total lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. Communications in Numerical Methods in Engineering 23(2), 121–134 (2007)
10. Moireau, P.: Assimilation de données par filtrage pour les systèmes hyperboliques du second ordre-Applications à la mécanique cardiaque. Ph.D. thesis
11. Mosegaard, J., Herborg, P., Sorensen, T.: A GPU accelerated spring mass system for surgical simulation. Studies Health Tech. & Inf. 111, 342–348 (2005)
12. Promayon, E., Baconnier, P., Puech, C.: Physically-Based Deformations Constrained in Displacements and Volume. Computer Graphics Forum 15(3), 155–164 (1996)

13. Rapaka, S., Mansi, T., Georgescu, B., Pop, M., Wright, G.A., Kamen, A., Comaniciu, D.: LBM-EP: Lattice-boltzmann method for fast cardiac electrophysiology simulation from 3D images. In: Ayache, N., Delingette, H., Golland, P., Mori, K. (eds.) MICCAI 2012, Part II. LNCS, vol. 7511, pp. 33–40. Springer, Heidelberg (2012)
14. Sermesant, M., Delingette, H., Ayache, N.: An electromechanical model of the heart for image analysis and simulation. IEEE TMI 25(5), 612–625 (2006)
15. Taylor, Z., Cheng, M., Ourselin, S.: High-speed nonlinear finite element analysis for surgical simulation using graphics processing units. IEEE Transactions on Medical Imaging 27(5), 650–663 (2008)
16. Zheng, Y., Barbu, A., Georgescu, B., Scheuering, M., Comaniciu, D.: Four-chamber heart modeling and automatic segmentation for 3 cardiac CT volumes using marginal space learning and steerable features. IEEE TMI 27, 1668–1681 (2008)