Available online at www.sciencedirect.com

ScienceDirect

Comput. Methods Appl. Mech. Engrg. 376 (2021) 113666

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

Multipatch Isogeometric Analysis for electrophysiology: Simulation in a human heart

Michele Bucelli[a,*], Matteo Salvador[a], Luca Dede'[a], Alfio Quarteroni[a,b]

[a] *MOX, Department of Mathematics, Politecnico di Milano, P.zza Leonardo da Vinci 32, 20133 Milan, Italy*
[b] *Mathematics Institute, EPFL, Av. Piccard, CH-1015 Lausanne, Switzerland*

Received 6 July 2020; received in revised form 24 December 2020; accepted 31 December 2020
Available online xxxx

Abstract

In the framework of cardiac electrophysiology for the human heart, we apply multipatch NURBS-based Isogeometric Analysis for the space discretization of the Monodomain model. Isogeometric Analysis (IGA) is a technique for the solution of Partial Differential Equations (PDEs) that facilitates encapsulating the exact representation of the computational geometry by using basis functions with high-order continuity. IGA features very small numerical dissipation and dispersion when compared to other methods for the solution of PDEs. The use of multiple patches allows to overcome the conventional limitations of single patch IGA, thanks to the gained flexibility in the design of the computational domain, especially when its representation is quite involved as in bioengineering applications. We propose two algorithms for the preprocessing of CAD models of complex surface and volumetric NURBS geometries with cavities, such as atria and ventricles: our purpose is to obtain geometrically and parametrically conforming NURBS multipatch models starting from CAD models. We employ those algorithms for the construction of an IGA realistic representation of a human heart. We apply IGA for the discretization of the Monodomain equation, which describes the evolution of the cardiac action potential in space and time at the tissue level. This PDE is coupled with suitable microscopic models to define the behavior at cellular scale: the Courtemanche–Ramirez–Nattel model for the atrial simulation, and the Luo–Rudy model for the ventricular one. Numerical simulations on realistic human atria and ventricle geometries are carried out, obtaining accurate and smooth excitation fronts by combining IGA with the multipatch approach for the geometrical representation of the computational domains, either surfaces for the atria or solids for the ventricles.

Keywords: Isogeometric Analysis; Multipatch NURBS; Cardiac electrophysiology; NURBS; Mesh generation

1. Introduction

The heart is a muscular organ whose contraction pumps blood into the circulatory system, delivering oxygen and nutrients and allowing all the cells to perform their physiological activities [1–3]. The mechanical contraction of the heart is triggered by the electrical excitation of the cardiac tissue, starting from the right atrium at the sinoatrial node (SAN) and propagating to the left atrium and to the ventricles [4,5]. Since cardiovascular diseases are a very common cause of death worldwide, understanding heart function in both physiological and pathological conditions

* Corresponding author.
E-mail addresses: michele.bucelli@polimi.it (M. Bucelli), matteo1.salvador@polimi.it (M. Salvador), luca.dede@polimi.it (L. Dede'), alfio.quarteroni@polimi.it (A. Quarteroni).

https://doi.org/10.1016/j.cma.2021.113666

is crucial. In computational medicine, mathematical modeling and numerical simulations are used to provide a quantitative representation of the heart function, thus giving a very valuable contribution in this field [6–8].

Indeed, the electrical activation of the heart can be accurately described with mathematical models expressed in terms of partial differential equations (PDEs). A very common model for cardiac electrophysiology is the Monodomain equation [6,9–11], which describes the evolution in time and space of the transmembrane potential in a continuous domain. This PDE needs to be coupled with an ionic model describing the transmembrane ionic currents that determine the electrical activation of cardiac cells. Among the different ionic models proposed in literature [12–15], we consider the Luo–Rudy (LR) model [16] for the ventricles and the Courtemanche–Ramirez–Nattel (CRN) [17] model for the atria.

The complexity of electrophysiological models poses a significant challenge for their mathematical analysis and numerical approximation [7,18–20]. A fine space resolution is indeed required to capture the sharp propagating excitation fronts and suitable methods are needed to control numerical dissipation and dispersion. The use of the Finite Element Method (FEM) in the space discretization of either the Bidomain model [21,22] or the Monodomain equation requires highly refined meshes to capture the propagating front without introducing spurious oscillations. This entails a very large number of degrees of freedom (DOFs) and consequently a large computational cost [23].

In this paper, numerical simulations of the cardiac electrophysiology of the human heart are performed by means of multipatch Isogeometric Analysis (IGA) in the framework of the Galerkin method [24–29]. IGA is a methodology for the space discretization of PDEs that in many circumstances yields an exact representation of the computational geometry. Thanks to the high-order continuity of NURBS basis functions [30], it allows to solve PDEs accurately and efficiently, without necessarily requiring fine meshes while featuring limited numerical dispersion and dissipation [27,31–33]. These aspects make IGA better suited than FEM for wave propagation problems and PDEs featuring solutions with coherent structures, as is the case of cardiac electrophysiology models.

IGA has been applied so far to cardiac electrophysiology in [10,34,35] using the single patch approach, i.e. describing the computational domain by means of a single NURBS-parametrized geometry, i.e the so called patch. In particular, in [10,35], the results obtained by using high-order NURBS basis are compared with those coming from the C^0-continuous basis functions typical of FEM. These works highlight the superior accuracy per degree-of-freedom of high-order continuous NURBS in IGA, that also enables a better control of numerical dispersion errors, a drawback that potentially affects instead FEM solutions of the monodomain equation. However, the accuracy of the solution obtained by IGA is affected by a parametrization of the domain featuring significant distortions [27,36,37]. If we consider geometries with multiple cavities or complex topologies, such as those in a realistic representation of the heart, it is crucial to represent the domain by means of multiple patches. This approach is known as multipatch technique [27,38] and it allows reproducing a complex geometry by means of multiple patches with mesh elements with much better quality than in the single patch case, besides allowing for more flexibility in the definition of the computational domain.

Although CAD software can natively handle NURBS-parametrized geometries, CAD models may not allow the strong imposition of continuity of the solution across the interface of adjacent patches, at least in a seamless manner. Indeed, to impose this specific condition, IGA requires such patches to be conforming, both geometrically and parametrically. For this reason, we propose two algorithms to convert a multipatch surface representation of the computational domain into an IGA suitable one, which allows the strong imposition of continuity. The first algorithm addresses patch non-conformity issues in a surface geometry, while the second one addresses the issue of constructing a volumetric parametrization starting from a description of the domain boundary. These algorithms are applied to the Zygote Solid 3D heart model [39], resulting in an IGA reconstruction of a realistic human heart. In this way we obtain a domain description ready for the application of multipatch IGA to the space discretization of differential problems, including but not limited to the electrophysiology model considered in this paper.

The paper is structured as follows. In Section 2, we provide a brief description of the electrophysiology of the human heart. In Section 3 we present NURBS-based IGA and the multipatch. In Section 4 we describe a preprocessing pipeline to convert complex CAD models with cavities into suitable multipatch computational models, and apply it to obtain a realistic representation of both atria and ventricles. In Section 5 we introduce the Monodomain equation along with the mathematical method used for its numerical discretization. In Section 6 we discuss the numerical results. Finally, in Section 7 we draw some conclusions.

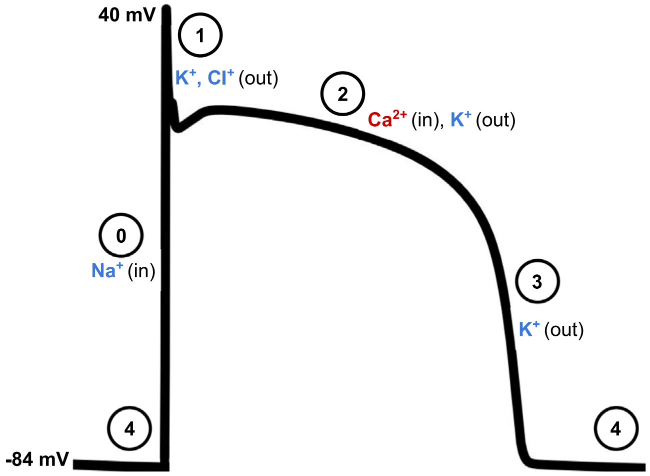


Fig. 1. Example of ventricular action potential with its phases: 0 (depolarization or upstroke), 1 (peak or notch), 2 (plateau), 3 (repolarization or recovery), 4 (resting). Intracellular calcium concentration Ca^{2+} plays a major role in heart contraction [40].

2. Cardiac electrophysiology

The human heart is a hollow muscular organ that pumps blood through the circulatory system, providing the body with oxygen and nutrients and assists with the removal of the metabolic wastes produced by cellular reactions [1–3]. The main driver of blood flow through the different chambers of the heart and through the circulatory system is the active contraction of the heart muscle. Such contraction is initiated by an electrical excitation of cardiac cells [6,41], which are able to activate themselves independently of nervous stimuli, unlike skeletal muscle cells.

At the microscopic scale, the electrical activation is due to an action potential (AP), which evolves in time according to the following phases (as depicted in Fig. 1) [6,42]:

- Phase 0: myocytes, i.e. cardiac cells, undergo a rapid depolarization due to the opening of the fast Na^+ channels. The difference of potential across the cellular membrane (referred to as transmembrane potential) is quickly (in about 2 ms) going from its resting value (about $-84\,\mathrm{mV}$) to slightly positive values thanks to the I_{Na} membrane current.
- Phase 1: inactivation of the fast Na^+ channels. The transient net outward current causing the small downward deflection of the action potential is due to K^+ and Cl^- ions carried by the I_{to1} and I_{to2} currents, respectively.
- Phase 2: plateau created by a balance between the inward movement of Ca^{2+}, the I_{Ca} current through L-type calcium channels, and the outward movement of potassium ions K^+, contributing to the I_{Ks} current through the slow delayed rectifier potassium channels.
- Phase 3: rapid repolarization phase of the action potential, when the L-type Ca^{2+} channels close while the K^+ channels are still open. This ensures a net outward current, corresponding to negative change in membrane potential, thus allowing more types of K^+ channels to open.
- Phase 4: resting phase. K^+ channels close and the transmembrane potential remains at the resting value until it is stimulated again by an external electrical activity.

We conclude that AP propagation is made possible by several ion channels opening and closing and resulting in ionic currents through the cellular membrane. Such currents cause variations in the concentration of calcium ions in the intracellular space, that in turn initiate the contraction of the myocytes [3], so that the electrical activation drives the mechanical one.

AP propagates from one cell to its neighbors through gap junctions, i.e. intercellular low-resistance ionic channels. This allows the stimulus to travel through the whole tissue from one cell to another. Cardiomyocytes are arranged in sheets of fibers. The gap junctions are located for the most part, although not exclusively, in the longitudinal direction of the fibers, resulting in a preferential propagation of the electric signal along the fibers [6].

Focusing on the macroscopic scale, as shown in Fig. 2, the electric signal originates at the sinoatrial node (SAN), a portion of tissue located at the junction between the superior vena cava and the right atrium. The SAN

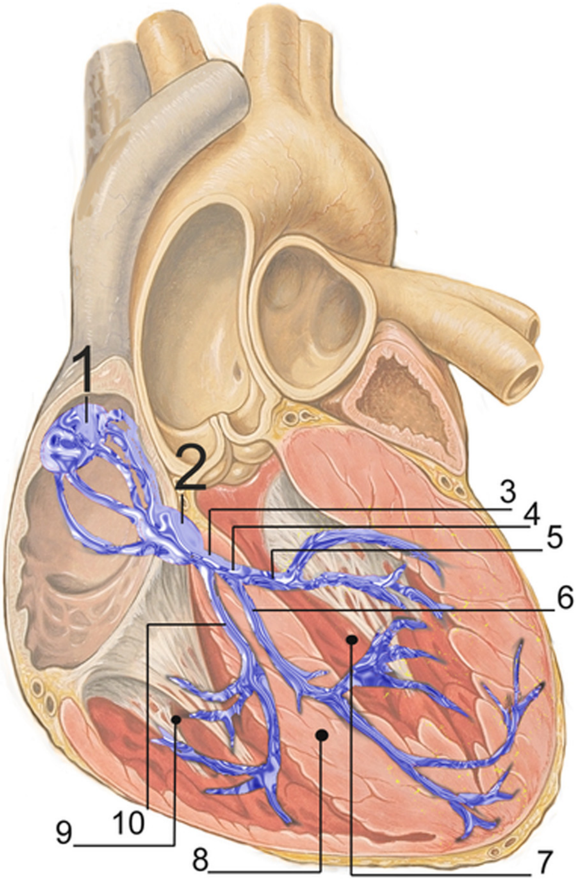


Fig. 2. Heart conduction system. 1. SAN. 2. AV node. 3. Bundle of His. 4. Left bundle branch. 5. Left posterior fascicle. 6. Left anterior fascicle. 7. Left ventricle 8. Ventricular septum. 9. Right ventricle. 10. Right bundle branch.
Source: Taken from [43].

is made of self-exciting cells and serves as the natural pacemaker of the heart. The AP propagates from the SAN through the right atrium and to the left atrium through a few preferential conduction sites (Bachmann's bundle first, then the anterior and posterior septum and the coronary sinus musculature) [4]. The activation front reaches the atrioventricular (AV) node, located at the junction between the interventricular septum and the interatrial septum. The cells composing this node have a slow conduction velocity, that causes a delay between the atrial and ventricular activation. The AV node conducts the electric signal through the atrioventricular septum and it activates the fibers of the bundle of His and of the Purkinje network, composed of cells specialized for the conduction. These fibers transmit then the action potential to the ventricular walls.

3. Multipatch IGA

IGA [24–28,44–46] is a framework for the spatial approximation of PDEs based on the isogeometric paradigm: that is using as basis functions the same functions that are used for the parametrization of the computational domain. The choice of such functions is driven by the design of the geometry.

Let $\Omega \subset \mathbb{R}^d$ be an open, bounded computational domain, defined as the union of $\mathrm{N_p}$ disjoint subdomains $\Omega_k, \mathrm{k} = 1, \dots, \mathrm{N_p}$, referred to as patches. More formally:

$$\overline{\Omega} = \bigcup_{k=1}^{N_p} \overline{\Omega}_k \quad \text{with} \quad \Omega_i \cap \Omega_j = \varnothing \quad i \neq j.$$

Each patch Ω_k is the image of a parametric domain $\widehat{\Omega} \subset \mathbb{R}^{\widehat{\mathrm{d}}}$, with $\widehat{\mathrm{d}} \leq \mathrm{d}$ (typically $\widehat{\Omega} = [0,1]^{\widehat{\mathrm{d}}}$) through a mapping in the form

$$\boldsymbol{\varphi}_\mathrm{k}(\boldsymbol{\xi}) = \sum_{\mathrm{i}=1}^{\mathrm{n_k}} \widehat{R_\mathrm{i}^\mathrm{k}}(\boldsymbol{\xi}) \boldsymbol{B}_\mathrm{i}^\mathrm{k} \qquad \mathrm{i} = 1,\dots,\mathrm{n_k},\ \mathrm{k} = 1,\dots,\mathrm{N_p}, \tag{1}$$

where $\widehat{R_\mathrm{i}^\mathrm{k}}(\boldsymbol{\xi})$ are suitable basis functions defined in the parametric domain and $\boldsymbol{B}_\mathrm{i}^\mathrm{k} \in \mathbb{R}^\mathrm{d}$ are points in the physical space known as control points.

In NURBS-based IGA, functions $\widehat{R_\mathrm{i}^\mathrm{k}}$ are $\widehat{\mathrm{d}}$-dimensional NURBS basis functions. They are tensor-product piecewise rational functions, whose definition involves the partition of each dimension of the parametric domain in sub-intervals, referred to as knot spans. The extremes of knot spans are known as knots, and the images of knot spans through the mapping Eq. (1) are referred to as elements. For more details on the definition of NURBS functions and their use for the parametrization of domains, we refer to [30,47].

The choice of NURBS functions allows an easy integration with CAD software used in the design of the computational domain, since it usually handles NURBS geometries natively. Moreover, NURBS functions allow several refinement procedures that enrich the functional space used for the parametrization while preserving the exact geometry. Finally, NURBS feature useful approximation properties that make them suitable for the discretization of PDE problems [27,36,37,45], in particular with respect to numerical dissipation [47] and dispersion [31,48]. Specifically, controlling numerical dispersion is an important feature of NURBS-based IGA when approximating electrophysiological problems [10,35].

For the space approximation of PDEs, the Galerkin method is employed using in each patch Ω_k as basis functions the push-forward through $\boldsymbol{\varphi}_\mathrm{k}$ of the basis used in the domain parametrization:

$$R_\mathrm{i}^\mathrm{k}(\boldsymbol{x}) = \widehat{R_\mathrm{i}^\mathrm{k}}(\boldsymbol{\varphi}_\mathrm{k}(\boldsymbol{x}))\ .$$

Following [45,49], we assume that adjacent patches are conforming. This means that, if $\Gamma_\mathrm{kl} = \partial\Omega_\mathrm{k} \cap \partial\Omega_\mathrm{l} \neq \varnothing$ is the interface of patches Ω_k and Ω_l for some $\mathrm{k} \neq \mathrm{l}$, then:

- Γ_kl is either a single point or the image of an entire edge of the parametric domain, or else the image of an entire face (for volumetric patches, i.e. for $\widehat{\mathrm{d}} = 3$) of the parametric domain for both patches;
- if a basis function R_i^k in Ω_k has support on Γ_kl, then there exists a basis function R_j^l in Ω_l that coincides with R_i^k at the interface: $R_\mathrm{i}^\mathrm{k}|_{\Gamma_\mathrm{kl}} = R_\mathrm{j}^\mathrm{l}|_{\Gamma_\mathrm{kl}}$.

These assumptions imply that adjacent patches are geometrically and parametrically coincident at their interface. We define the function spaces

$$\begin{aligned} &V_\mathrm{h}^\mathrm{k} = \mathrm{span}\{R_\mathrm{i}^\mathrm{k},\ \mathrm{i} = 1,\dots,\mathrm{n_k}\} \qquad \mathrm{k} = 1,\dots,\mathrm{N_p}\ , \\ &V_\mathrm{h} = \{f \in C^0(\overline{\Omega})\ :\ f|_{\Omega_\mathrm{k}} \in V_\mathrm{h}^\mathrm{k} \quad \forall \mathrm{k} = 1,\dots,\mathrm{N_p}\}\ . \end{aligned}$$

Under the previous assumptions, it is possible to define a basis over the space V_h. For each basis function R_i^k define an index $G(\mathrm{i},\mathrm{k}) \in \mathbb{N}$ such that

$$\begin{aligned} &G(\mathrm{i},\mathrm{k}) = G(\mathrm{j},\mathrm{l}) \qquad \text{if } \ \mathrm{supp}(R_\mathrm{i}^\mathrm{k} \cap \Gamma_\mathrm{kl}) \neq \varnothing,\ \mathrm{supp}(R_\mathrm{j}^\mathrm{l} \cap \Gamma_\mathrm{kl}) \neq \varnothing,\ R_\mathrm{i}^\mathrm{k}|_{\Gamma_\mathrm{kl}} = R_\mathrm{j}^\mathrm{l}|_{\Gamma_\mathrm{kl}}, \\ &G(\mathrm{i},\mathrm{k}) \neq G(\mathrm{j},\mathrm{l}) \qquad \text{otherwise}, \end{aligned}$$

and let

$$\mathcal{I} = \{G(\mathrm{i},\mathrm{k}),\ \mathrm{i} = 1,\dots,\mathrm{n_k},\ \mathrm{k} = 1,\dots,\mathrm{N_p}\},$$

be the set of all indices $G(\mathrm{i},\mathrm{k})$. In this way, we assign the same index to two corresponding basis functions at the interface between two patches. Then, a basis for V_h is given by the functions

$$R_\mathrm{l}(\boldsymbol{x}) = \begin{cases} R_\mathrm{i}^\mathrm{k}(\boldsymbol{x}) & \text{if } \ \boldsymbol{x} \in \Omega_\mathrm{k} \text{ and } \mathrm{l} = G(\mathrm{i},\mathrm{k}) \\ 0 & \text{otherwise} \end{cases} \qquad \mathrm{l} \in \mathcal{I}\ .$$

The space V_h is used as trial and test space for the approximation of the differential models for electrophysiology that we will introduce in Section 5.2. The method achieves optimal order of convergence with respect to the mesh size parameter h, maximum diameter of all elements of all patches [36,45]. More details on the approximation properties of the spaces $V_{\mathrm{h}}^{\mathrm{k}}$ and V_{h} can be found in [36,37,45].

Following this approach, solutions on adjacent patches are coupled by prescribing the continuity of the solution across the interface, which can be achieved provided the adjacent patches match completely both geometrically and parametrically [27]. This requires adjacent patches to have the same knot vectors along their interfaces and to have coincident control points. The strong imposition of continuity across patch interfaces can be obtained also if the two patches are geometrically coincident at the interface, and the interface parametrization of one patch is obtained via h-refinement from that of the other [27]. A discontinuous Galerkin technique for the coupling of non-conforming adjacent patches is discussed in [50], while the possibility of imposing higher order continuity across patch interfaces is investigated in [29,38,51]. Moreover, in [52–55] techniques for the management of trimmed geometries and geometries obtained via boolean operations are presented.

The accuracy of the solution is hindered by the distortion of the geometrical map $\boldsymbol{\varphi}$ [36], quantified in terms of the determinant of its gradient $J = \det(\nabla\boldsymbol{\varphi})$. Because of the tensor product structure of the parametric space, representing multiple connected domains with a single patch causes large deformations in the geometrical map. Indeed, each patch can be seen as a continuous deformation of the parametric space (i.e. a cube). Therefore, the single patch representation of complex domains, for example domains with multiple cavities such as those involved in cardiac modeling, requires significant mesh deformations. In such cases, multiple patches allow to construct regular parametrizations, resulting in a gain in accuracy [27].

A detailed description of the methods and properties of NURBS-based IGA can be found in [26,27].

4. Generating multipatch NURBS geometries for heart modeling

In order to solve PDEs on NURBS multipatch geometries in the standard continuous Galerkin framework, the patches need to be conforming. More in general, the parametrizations of adjacent patches must be such that the basis functions used by one are a subset of those used by the other, so that the spaces spanned by those basis functions are contained one within the other. This can be achieved, for example, by applying the h-refinement procedure to one of the two patches, starting from a conforming parametrization. Moreover, if the computational domain is a volume, the NURBS parametrization should cover the whole volume, and not only its boundary surface. For these reasons, geometries coming from CAD modeling may not be ready to be used in numerical simulations directly.

Therefore, we propose two algorithms for the preprocessing of NURBS geometries. The first algorithm addresses non-conformity issues in a NURBS multipatch surface geometry. The second algorithm deals with the problem of constructing a NURBS volume starting from a surface representation of its boundary. In both cases, we will assume the original multipatch geometries to be continuous at the geometric level, i.e. to be composed by a set of G^0 surface patches. The two algorithms provide a continuous parametrization for the same geometry.

The manipulation of NURBS geometries is performed by means of the MATLAB/Octave NURBS package [56,57]. Pseudocode for the procedures described in this section can be found in the Appendix.

Both for atria and ventricles, the source geometry is a 3D model of the heart of a healthy, 21 year-old, 50th percentile U.S. male, created by the Zygote Media Group [39], reconstructed from magnetic resonance images. The original geometry is shown in Fig. 3.

4.1. Preprocessing of a realistic human atrial geometry

We consider a realistic human atrial geometry modeled as a NURBS multipatch surface. All patches are geometrically watertight at their interfaces. However some pairs of adjacent patches do not have the same parametrization at the interface, which is required to impose strong continuity of the solution. We face these issues in the mesh generation, producing a geometry composed of conforming patches. We distinguish two types of non-conformities:

- patch-level non-conformities: two adjacent patches only share a part of one side (Fig. 4(a));
- knot-level non-conformities: two adjacent patches share an entire side, but the interface is differently parametrized on the two patches (due to different knot vectors in the corresponding parametric directions; Fig. 6(a)).

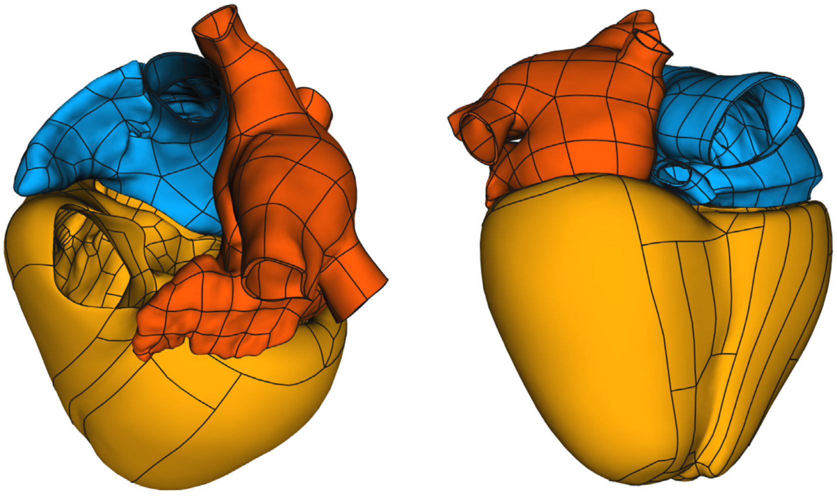


Fig. 3. Zygote heart model [39]. Black lines are the boundaries of the patches in the original geometry. Notice in particular that patches are not conforming, as evident from the figure on the right.

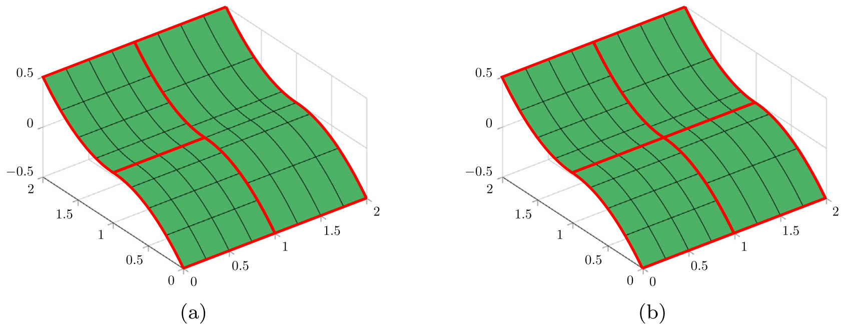


Fig. 4. Solution of a patch-level non-conformity; patch boundaries are shown in red. (a): initially, one side of the rightmost patch is shared with both the other patches; (b): a C^0 knot is inserted in the rightmost patch in correspondence of the boundary between the other patches, and then such a patch is split into two patches at that position. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

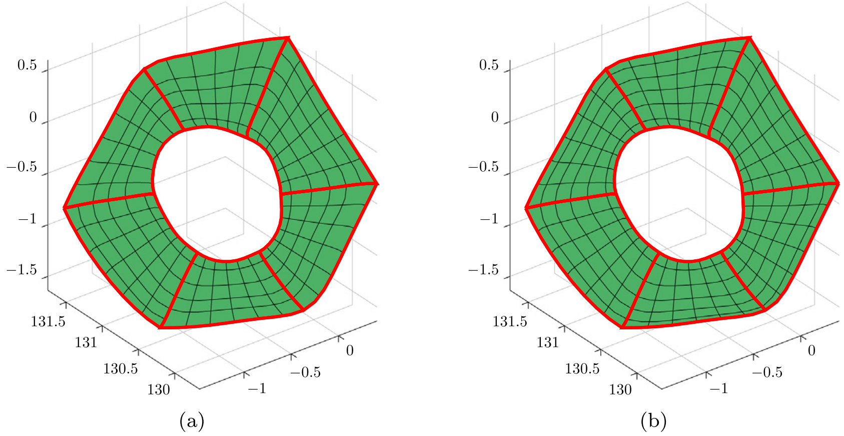
We assume that no patch is adjacent to itself along a portion of its boundary, so that non-conformities only occur between two distinct patches.

The following sections describe the algorithms to address non-conformities of the two types. Such algorithms manipulate the patches' knot vectors and control points in order to eliminate all non-conformities and convert the geometry into a conforming multipatch NURBS surface.

4.1.1. Detection of patch-level non-conformities

For a NURBS surface patch, assuming $\widehat{\Omega} = (0, 1)^2$, we refer to the four points corresponding to the parametric coordinates (0, 0), (0, 1), (1, 0) and (1, 1) as vertices, and to the four NURBS curves corresponding to the parametric equations $\xi = 0$, $\xi = 1$, $\eta = 0$ and $\eta = 1$ as sides. For our purposes, the non-conformity between two patches can be checked by counting the vertices that are shared by two patches, as described in Algorithm 1. We remark that this approach does not cover all the possible cases of non-conforming patches that may occur. It is however sufficient to deal with the Zygote model. For more details, see Appendix A.1.

To check if one vertex lies on one side of a patch (Algorithm 1, lines 7 and 9), the NURBS curve that parametrizes that side must be inverted. This requires the solution of a non-linear equation with Newton's method [30] and can be computationally demanding.



(a) (b)

Fig. 5. A group of patches before (a) and after (b) the removal of knot-level non-conformities. After the removal of non-conformities, all the patches have the same knot vector along the radial direction.

4.1.2. Solution of patch-level non-conformities

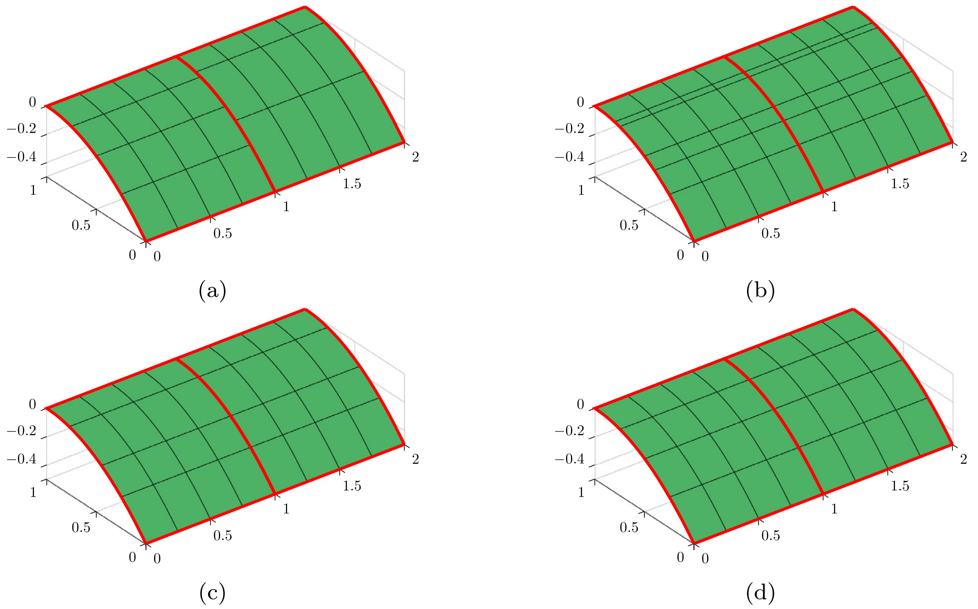
Once a patch-level non-conformity is detected, it can be removed by inserting a knot with multiplicity $p + 1$, where p is the polynomial degree of the patch, in correspondence of the hanging node, thus splitting one of the two patches into two smaller patches. This operation, shown in Fig. 4 and described in Algorithm 3, does not introduce any geometrical change in the geometry. To solve all patch-level non-conformities, we detect all pairs of non-conforming adjacent patches. Then, we solve non-conformities by the procedure described above and update the list of non-conforming pairs since the modification might have introduced further issues, repeating until all non-conformities have been solved. This procedure is described in Algorithm 4.

4.1.3. Solution of knot-level non-conformities

A knot-level non-conformity between two adjacent patches can be detected by comparing the knot vectors at the interface of the two patches. If the parametrizations of the interface on the two patches have the same orientation, then the patches are conforming if they have the same knot vectors. If the parametrizations have opposite orientation, then the patches are conforming if their knot vectors have the same size N and, by defining $\{\xi_\mathrm{i}^1\}_{\mathrm{i}=1}^\mathrm{N}$ and $\{\xi_\mathrm{i}^2\}_{\mathrm{i}=1}^\mathrm{N}$ as the knots of the two patches respectively, it holds that $\xi_\mathrm{i}^1 = 1 - \xi_{\mathrm{N-i+1}}^2$.

Knot-level non-conformities are removed simultaneously for a group of patches on which knot-level conformity must be enforced (see for example Fig. 5). This requires significantly less steps than removing non-conformities for two patches at a time and repeating. Let us assume to start from patch interfaces with matching orientations, so that conformity is achieved by imposing the same knot vector along the interface on all patches. A first naïve approach is to take the union of the interface knot vectors of all patches in the group, denoted by $\bar{\Xi}$, and insert in each patch the knots from $\bar{\Xi}$ that it does not already have. This procedure, shown in Fig. 6(b), does not introduce any geometrical change in the patches. However, in case there are two very close but distinct knots in two different patches, this results in extremely thin elements. In general, taking the union of knot vectors can lead to unevenly spaced knots, which is undesirable. For this reason, if two knots in $\bar{\Xi}$ are too close according to a prescribed tolerance, they are "merged" into their midpoint (Fig. 6(c)), that is their midpoint is inserted and the two knots are removed using the algorithm described in Appendix A.3. To decide whether two knots should be merged, their distance d in the physical space is checked against a threshold $d_\mathrm{th} = \varepsilon_\mathrm{th} h$: if $d < d_\mathrm{th}$, the knots are removed and their midpoint is inserted, otherwise they are kept as they are. In general, operations involving the removal of a knot introduce geometrical changes in the geometry, and whether such changes are acceptable or not should be manually assessed to properly tune the d_th parameter. A large value of d_th leads to more evenly spaced elements, at the price of more significant geometrical modifications. All these steps are reported in Algorithm 5. The case of adjacent patches where the parametrizations of the interface are oriented in a different way can be treated in a similar fashion. For further details, we refer to Appendix A.2 and Algorithm 5.

Additionally, it is sometimes desirable to obtain a mesh with evenly spaced elements. Indeed, the whole knot vector of the patch can be potentially replaced by a uniform knot vector of the same length by means of knot



(a)

(b)

(c)

(d)

Fig. 6. Steps for the solution of a knot-level non-conformity between two patches. Patch boundaries are shown in red. (a): initial non-conforming patches; (b): the knot vectors are joined, introducing very stretched elements; (c): close knots are replaced by their midpoint, improving the element shape; (d): optionally, the knot vectors are forced to be uniform. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

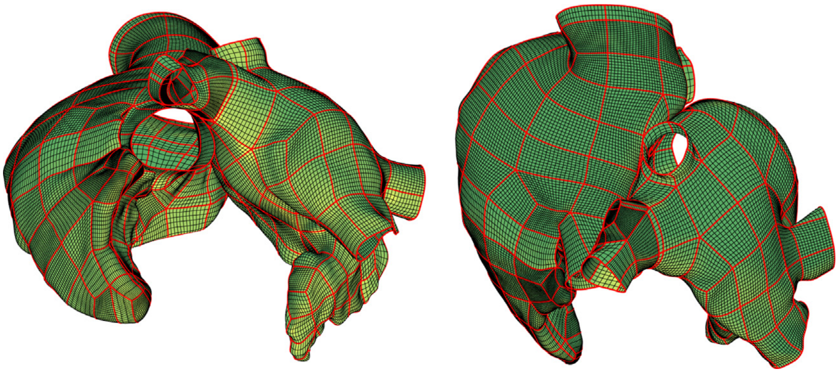


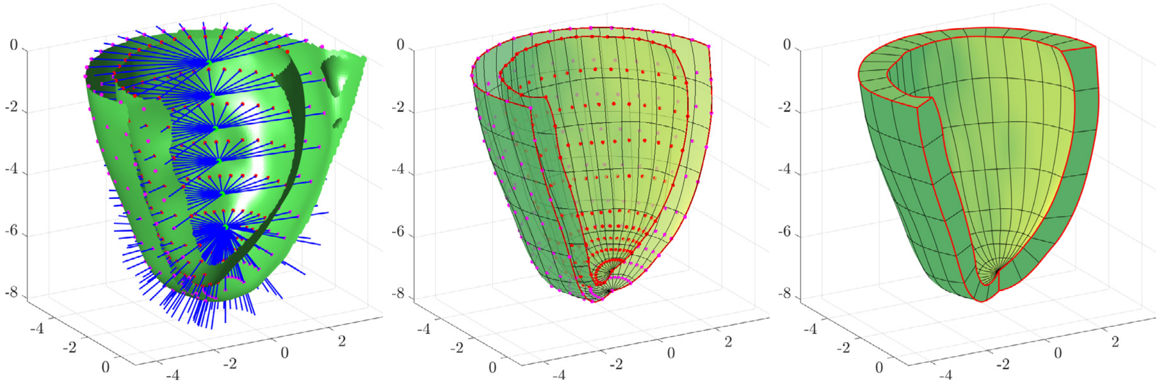
Fig. 7. The meshes for the two atria, at a coarse refinement level. Patch boundaries are shown in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

removal and insertion (Fig. 6(d)). As in the previous step, these operations can alter the geometry of the domain. However, while the removal of thin elements introduces small modifications which are generally acceptable, the redistribution of knots to obtain evenly spaced elements might introduce more significant geometrical changes. This should be checked manually, and this step should be skipped if the geometry is changed in an unacceptable way.

This operation is repeated on all non-conforming patches, until all knot-level non-conformities are solved.

4.1.4. Application to the human atrial geometries

We implement the above described algorithm in MATLAB relying on the NURBS package [56] for what concerns the treatment of the geometries. The resulting meshes after preprocessing are shown in Fig. 7. Our right atrium is comprised of 172 patches for a total of 162’234 elements, while our left atrium is comprised of 145 patches for a total of 121’671 elements. All the patches are parametrized by NURBS functions of degree 3.



(a) (b) (c)

Fig. 8. Reconstruction of the left ventricle patch. (a): half-lines and intersection points; green dots are the half-line origin points, blue lines are the half-lines, red and magenta dots are the intersection points between the half-lines and the endocardium and epicardium respectively; the green surface is the source geometry, sectioned for visualization. (b): fitted NURBS surface patches for endocardium and epicardium, alongside the points on which they were fitted; (c): NURBS volume patch for the left ventricle, obtained by linear interpolation of the endocardial and epicardial surface patches. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Moreover, for both atria, the internal and external surfaces were available, while the simulation was performed only on the external surface. The internal surfaces were manually eliminated using the CAD software SolidWorks [58].

4.2. Preprocessing of realistic human left and right ventricle geometries

We also preprocess a geometry representing human left and right ventricles. The source geometry is represented only in terms of its boundary (this is referred to as boundary representation, or B-rep), described by a set of non-conforming NURBS surface patches. Given the non-negligible thickness of the ventricles, we aim at representing them by one or more NURBS solid volume patches, obtained by filling the volume between the epicardial and endocardial surfaces.

The problem of moving from a surface representation of the boundary of a volume to a trivariate parametrization for that same volume is not trivial, and several automatic approaches have been proposed [46,59–63].

Our procedure is based on reconstructing in a semi-automatic way the source geometry, up to a small approximation error, with a parametrization independent of the original one. It is possible to impose patch conformity constraints to the resulting parametrization. In particular, we can impose a one-to-one correspondence of patches between the epicardium and the endocardium, that allows to construct in a trivial way the volume patches, and conformity between adjacent patches.

The method is outlined as follows. First, a structured grid of points lying on the surface of interest is obtained by a raycasting method. Then, NURBS surfaces are least-squares fitted on the grid of points. This is done both on the epicardium and on the endocardium. Finally, the epicardial and endocardial patches are connected to obtain the volume patches. The steps are shown for the left ventricle patch in Fig. 8.

This procedure still requires to choose manually the subdivision in patches of the geometry, and careful manual tuning for the placement of origin points and directions for raycasting, for the number of points to be used for fitting and for the knot vectors of the fitted surfaces. On the other hand, the grid of points could be provided by other sources, such as experimental measures, so that this approach can be used to construct analysis-suitable geometries in more general settings.

The following sections detail the individual steps of the above-mentioned procedure.

4.2.1. Computation of fitted points

The points at which surfaces are fitted are computed by a raycasting procedure: each of the needed points is obtained by intersecting a half-line, typically starting from the approximate centerline of the chamber being reconstructed and pointing in radial direction, with the source geometry.

The intersection of a half-line with a NURBS surface patch requires the solution of the non-linear system of equations:

$$\sum_{\mathrm{i}=1}^{\mathrm{n}} N_{\mathrm{i,p}}(\xi)\boldsymbol{P}_{\mathrm{i}} = \boldsymbol{o} + \eta\boldsymbol{r} \qquad \xi \in [0,1],\ \eta > 0\ , \tag{2}$$

where $\boldsymbol{o}$ and $\boldsymbol{r}$ are respectively the origin and direction of the half-line. Efficient iterative methods for the solution of Eq. (2) have been proposed [64,65], but the problem can become expensive, especially as the number of grid points and amount of patches increase.

A set of points is obtained for each patch we want to construct. For each patch, the origin points and directions of the half-lines are manually constructed so that the resulting intersection points form a grid without intersecting edges. To do so it is possible to exploit symmetries in the geometry to be reconstructed (for example, the approximate cylindrical symmetry of the left ventricle in our case). To enforce conformity between adjacent patches, the grids of intersection points used to fit those patches must share an interface of coincident points. This can be obtained by using the same half-line origins and directions for those points when computing the two grids.

Fig. 8(a) shows as an example the half-lines and intersection points that are obtained for the left ventricle endocardium.

4.2.2. *Fitting of NURBS surfaces*

The algorithm for the least-squares fit of a NURBS surface to a grid of points is taken from [30]. However, such an algorithm chooses knot vectors according to an optimality criterion based on the points to be fitted that aims at obtaining the least distorted parametrization. This would result in different fits at the interface for adjacent patches, that would end up being neither parametrically nor geometrically conforming.

For this reason, we choose to use uniform knot vectors for all patches, even if this is not optimal in the sense of [30], since the obtained result is sufficiently good in terms of distortion.

An example of a surface fitted on the points obtained from raycasting is shown in Fig. 8(b).

4.2.3. *Construction of NURBS volumes*

Once NURBS surfaces for the epicardium and endocardium are obtained, they can be connected with a volume (see Fig. 8(c)), with a straightforward generalization of the ruled surface construction algorithm described in [30], resulting in a linear interpolation between the two surfaces to be connected.

Since the surface patches are constructed to be geometrically and parametrically conforming, also the volume patches will be. The resulting geometry can be used as a computational domain for numerical simulations.

4.2.4. *Application to the ventricles' geometry*

The ventricles' geometry is reconstructed by five patches: left ventricle, right ventricle, septum and two junctions that connect the other three (see Fig. 9). This kind of subdivision allows us to have conforming patches while maintaining a small distortion of the elements.

In both ventricles, the half-line origin points are placed approximately along the centerline, and half-line directions are chosen radially. This allows to obtain a regular parametrization of both the endocardium and the epicardium, especially for the left ventricle, whose cross section is approximately circular. Due to the more complex shape of the right ventricle, its parametrization is slightly more irregular, in the sense that it leads to unevenly spaced elements, and it is more sensitive to the choice of half-line origins and directions. The polynomial degree used to parametrize the patches is equal to 2.

5. Mathematical models for cardiac electrophysiology

In this section we introduce the Monodomain equation and the ionic models, and we discuss their numerical discretization both in space and time.

5.1. *Monodomain equation and ionic model*

The evolution in space and time of the transmembrane potential in the cardiac tissue is described by means of the Monodomain model [6]. Let $\Omega \subset \mathbb{R}^3$ be the computational domain, $u(\boldsymbol{x}, t)$ the transmembrane potential at the

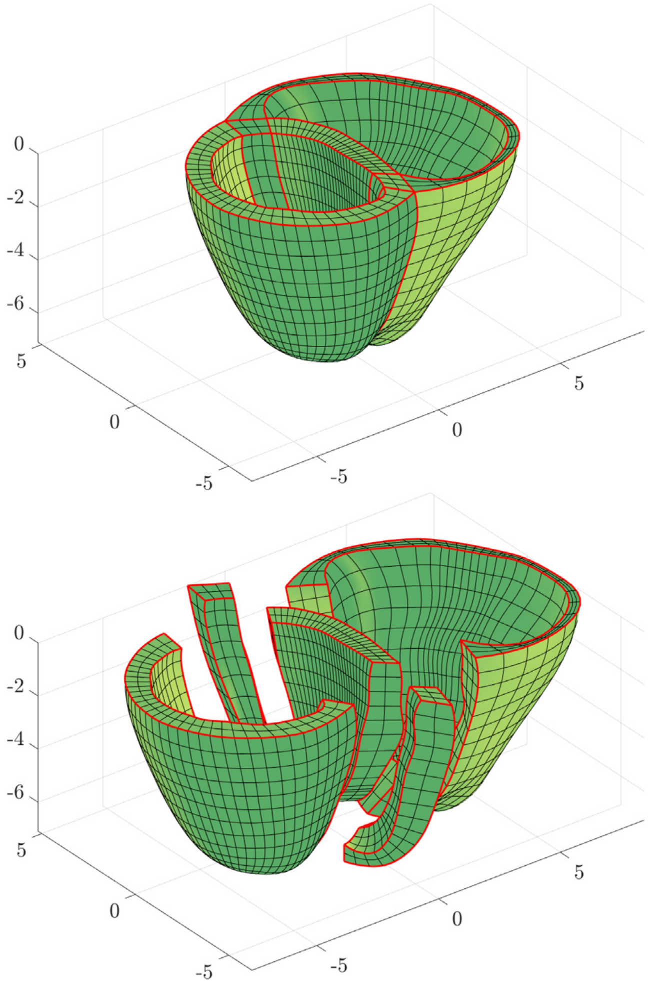


Fig. 9. Left and right ventricle geometry, at a coarse refinement level. Patch boundaries are shown in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

space coordinate $\boldsymbol{x} \in \Omega$ and at time $t \in (0, T)$, C_m the total membrane capacitance and χ the membrane surface to volume ratio. The Monodomain equation coupled with a generic ionic model reads:

$$\begin{cases} \chi C_\mathrm{m} \dfrac{\partial u}{\partial t} - \nabla \cdot (D \nabla u) + \chi I_\mathrm{ion}(u, \boldsymbol{w}, \boldsymbol{c}) = I_\mathrm{app} & \text{in } \Omega \times (0, T)\,, \\ \dfrac{\partial \boldsymbol{w}}{\partial t} = \boldsymbol{F}(u, \boldsymbol{w}) & \text{in } \Omega \times (0, T)\,, \\ \dfrac{\partial \boldsymbol{c}}{\partial t} = \boldsymbol{G}(u, \boldsymbol{w}, \boldsymbol{c}) & \text{in } \Omega \times (0, T)\,, \\ D \nabla u \cdot \boldsymbol{n} = 0 & \text{on } \partial\Omega \times (0, T)\,, \\ u = u_0 & \text{in } \Omega \times \{0\}\,, \\ \boldsymbol{w} = \boldsymbol{w}_0 & \text{in } \Omega \times \{0\}\,, \\ \boldsymbol{c} = \boldsymbol{c}_0 & \text{in } \Omega \times \{0\}\,, \end{cases} \tag{3}$$

where $\boldsymbol{n}$ is the outward directed unit vector normal to $\partial\Omega$, $C_\mathrm{m} = 0.001\,\mathrm{mF/cm^2}$, $\chi = 1000\,\mathrm{cm^{-1}}$. u_0, $\boldsymbol{w}_0$, $\boldsymbol{c}_0$ and I_app are the given initial conditions and the applied current respectively. System (3) defines the propagation in space and time of the transmembrane potential $u(\boldsymbol{x}, t)$ at the tissue level along with the ionic behavior, at the cellular level, by means of $\boldsymbol{w}$ and $\boldsymbol{c}$ [6]. We refer to $\boldsymbol{w}$ as gating variables, which represent fractions of open ionic channels, and to $\boldsymbol{c}$ as intracellular ionic concentrations. I_{ion}, $\boldsymbol{F}$ and $\boldsymbol{G}$ strictly depend on the selected ionic model. I_{ion}, i.e. the ionic current, models the multiscale effects from the cellular level up to the tissue one. $\boldsymbol{F}$ and $\boldsymbol{G}$ are suitably defined vector-valued functions. A variety of ionic models have been proposed in literature [12,13,15–17]. We consider here the LR model for the ventricles [16] and the CRN for the atria [17]. Both models reproduce the evolution of the intracellular calcium concentration, which is a crucial ionic species that triggers the mechanical activation of the cardiac tissue [3]. A comprehensive list of the existing ionic models can be found in [6]. The problem is closed with homogeneous Neumann boundary conditions for the transmembrane potential, to keep into account the assumption of having an electrically isolated domain, and initial conditions for both transmembrane potential and ionic model variables.

D expresses the conductivity of the tissue and can be used to represent its electric anisotropy, accounting for the presence of cardiac fibers. Assuming axisymmetric conductivity and denoting by $\boldsymbol{a}_{\mathrm{l}}(\boldsymbol{x})$ the local direction of the fibers, the tensor can be computed as

$$D(\boldsymbol{x}) = \sigma_{\mathrm{t}}(\boldsymbol{x})I + (\sigma_{\mathrm{l}}(\boldsymbol{x}) - \sigma_{\mathrm{t}}(\boldsymbol{x}))\boldsymbol{a}_{\mathrm{l}}(\boldsymbol{x})\boldsymbol{a}_{\mathrm{l}}(\boldsymbol{x})^T \,,$$

where $\sigma_{\mathrm{l}} = 0.012\,042\,\mathrm{S\,cm^{-1}}$ and $\sigma_{\mathrm{t}} = 0.001\,761\,\mathrm{S\,cm^{-1}}$ are the conductivities in the fiber direction and in the direction orthogonal to fibers, respectively.

The Monodomain equation is a simplified version of the Bidomain model [6,34], derived under the assumption that the intracellular and extracellular conductivities of the Bidomain model have equal anisotropy ratio. This model is widely used in literature [6,66–68]. The advantage of the Monodomain equation resides in its reduced complexity in comparison with the Bidomain model, which is expressed as either a parabolic–elliptic or parabolic–parabolic system of two PDEs. As a consequence, the Monodomain equation is much less demanding from the computational viewpoint than the Bidomain model. Moreover, the system to be solved for the Monodomain equation is significantly better conditioned than the one of the Bidomain model. The results obtained from the Monodomain equation are accurate both in physiological conditions and pathological cases. Indeed, this equation is employed to perform whole heart electromechanical simulations in patient-specific cases [69]. It is also effectively used to study different cardiac diseases, such as potentially lethal arrhythmias in patients with ischemic cardiomyopathy [66,70,71]. The complexity of the Bidomain model might be required for the heart-torso coupling [72], where the extracellular potential could be used to compute electrocardiograms (ECGs). In our framework, as already pointed out, we are interested in the electrophysiological behavior of the heart. For all these reasons, we used the Monodomain equation in our numerical simulations. Deeper analysis and comparisons of the two different models are carried out for example in [6,9,20].

5.2. Space discretization

The weak formulation of the Monodomain equation, see e.g. [11], is here discretized in space using multipatch IGA. We look for an approximation u_{h} of $u \in V = H^1(\Omega)$ in Eq. (3) in the finite dimensional space $V_{\mathrm{h}} \subset V$, so that

$$u_{\mathrm{h}}(\boldsymbol{x}, t) = \sum_{\mathrm{j}=1}^{\mathrm{N_h}} U_{\mathrm{j}}(t) R_{\mathrm{j}}(\boldsymbol{x}) \,,$$

where $U_{\mathrm{j}}(t)$, $\mathrm{j} = 1, \ldots, \mathrm{N_h}$ are suitable time-dependent control variables and $R_{\mathrm{j}}(\boldsymbol{x})$, $\mathrm{j} = 1, \ldots, \mathrm{N_h}$ are NURBS-based basis functions, as illustrated in Section 3. Then we approximate the first equation of Eq. (3) by:

$$\chi C_{\mathrm{m}} \sum_{\mathrm{j}=1}^{\mathrm{N_h}} \frac{\partial U_{\mathrm{j}}}{\partial t} \left(R_{\mathrm{j}},\ R_{\mathrm{i}}\right) + \sum_{\mathrm{j}=1}^{\mathrm{N_h}} U_{\mathrm{j}}\, a(R_{\mathrm{j}},\ R_{\mathrm{i}}) + \chi\left(I_{\mathrm{ion}}(u_{\mathrm{h}}, \boldsymbol{w}, \boldsymbol{c}),\ R_{\mathrm{i}}\right) = \left(I_{\mathrm{app}},\ R_{\mathrm{i}}\right) \qquad \forall \mathrm{i} = 1, \ldots, \mathrm{N_h} \,, \tag{4}$$

where $(\cdot, \cdot)$ denotes the L^2 scalar product and

$$a : V \times V \to \mathbb{R} \,, \qquad a(\varphi, \psi) = \int_{\Omega} D\nabla\varphi \cdot \nabla\psi \; d\Omega \,.$$

Eq. (4) can be rewritten as a system of ODEs:

$$\chi C_{\mathrm{m}} M \frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{U}(t) + K\boldsymbol{U}(t) + \chi \boldsymbol{I}_{\mathrm{ion}}(\boldsymbol{U}(t), \boldsymbol{w}(t), \boldsymbol{c}(t)) = \boldsymbol{I}_{\mathrm{app}}(t) \,, \tag{5}$$

where $\boldsymbol{U}(t)$ denotes the vector of entries U_j, M is the mass matrix of entries $M_\mathrm{ij} = (R_\mathrm{j}, R_\mathrm{i})$, K is the stiffness matrix of entries $K_\mathrm{ij} = a(R_\mathrm{j}, R_\mathrm{i})$ and $\boldsymbol{I}_\mathrm{ion}(\boldsymbol{U}(t), \boldsymbol{w}(t), \boldsymbol{c}(t))$ is a vector of entries $(\boldsymbol{I}_\mathrm{ion})_\mathrm{i} = (I_\mathrm{ion}(u_\mathrm{h}(t), \boldsymbol{w}(t), \boldsymbol{c}(t)), R_\mathrm{i})$.

For the computation of the gating variables $\boldsymbol{w}$ and ionic concentrations $\boldsymbol{c}$ we use Gaussian integration [10,23,73]. We aim at computing

$$(\boldsymbol{I}_\mathrm{ion}(t))_\mathrm{i} = (I_\mathrm{ion}(u_\mathrm{h}(t), \boldsymbol{w}(t), \boldsymbol{c}(t)), R_\mathrm{i}) = \int_\Omega I_\mathrm{ion}(u_\mathrm{h}(t), \boldsymbol{w}(t), \boldsymbol{c}(t)) R_\mathrm{i} \, d\Omega \,.$$

We employ a Gaussian quadrature rule of order $p+1$, where p is the polynomial degree of the basis functions, to approximate the integral above. We denote by N_e the number of elements in Ω. $\{K_\mathrm{j}\}_{\mathrm{j}=1}^{\mathrm{N_e}}$ are the elements, $\{\boldsymbol{x}^\mathrm{j}_\mathrm{m}\}_{\mathrm{m}=1}^{\mathrm{s}}$ are the Gaussian quadrature points on the jth element, with associated weights $\{\omega_\mathrm{m}\}_{\mathrm{m}=1}^{\mathrm{s}}$, with s being the number of quadrature points per element. Using the introduced notation we get:

$$\begin{aligned}(\boldsymbol{I}_\mathrm{ion}(t))_\mathrm{i} &= \int_\Omega I_\mathrm{ion}(u_\mathrm{h}(t), \boldsymbol{c}(t), \boldsymbol{w}(t)) R_\mathrm{i} \, d\Omega = \sum_{\mathrm{j}=1}^{\mathrm{N_e}} \int_{K_\mathrm{j}} I_\mathrm{ion}(u_\mathrm{h}(t), \boldsymbol{w}(t), \boldsymbol{c}(t)) R_\mathrm{i} \, d\Omega \approx \\ &\approx \sum_{\mathrm{j}=1}^{\mathrm{N_e}} \sum_{\mathrm{m}=1}^{\mathrm{s}} I_{ion}(u_\mathrm{h}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t), \boldsymbol{w}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t), \boldsymbol{c}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t)) R_\mathrm{i}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}) \omega_\mathrm{m}. \end{aligned} \tag{6}$$

Therefore, for the purpose of computing the ionic current term in the discrete weak formulation Eq. (5), it is sufficient to compute the values of $\boldsymbol{w}$ and $\boldsymbol{c}$ at the Gauss points. This can be done by solving at each quadrature point the following ODE problem:

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{w}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t) = \boldsymbol{F}(u_\mathrm{h}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t), \boldsymbol{w}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t)) & t \in (0, T)\,, \\ \dfrac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{c}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t) = \boldsymbol{G}(u_\mathrm{h}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t), \boldsymbol{w}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t), \boldsymbol{c}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, t)) & t \in (0, T)\,, \\ \boldsymbol{w}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, 0) = \boldsymbol{w}_0(\boldsymbol{x}^\mathrm{j}_\mathrm{m})\,, \\ \boldsymbol{c}(\boldsymbol{x}^\mathrm{j}_\mathrm{m}, 0) = \boldsymbol{c}_0(\boldsymbol{x}^\mathrm{j}_\mathrm{m})\,. \end{cases}$$

The solution of the ionic model reduces to the solution of a set of independent ODEs, one for each quadrature node of the mesh.

Alternative approaches to the discretization of the ionic model are based on applying a Galerkin method to the ionic model equations as well, and are known as nodal interpolation methods [10,23,74,75]. With respect to such methods, Gaussian integration saves computational resources by avoiding the interpolation of the ionic variables, at the price of a higher memory storage due to the need to save ionic variables at each quadrature node.

IGA has been applied to the discretization of the Monodomain or Bidomain model in [10,34,35], highlighting the good properties of the method in terms of accuracy of the action potential propagation [10] and numerical dispersion [35]. However, the use of a single patch in those cases limits the applicability of the method to complex, realistic geometries. Such limit can be overcome by using multiple patches.

5.3. Time discretization

For the discretization of time derivatives, both for the Monodomain and ionic model equations, we use Backward Differentiation Formulas (BDF) of order 2, suitable for the solution of stiff differential equations [75]. We treat the ionic current $\boldsymbol{I}_\mathrm{ion}$, i.e. the only non linear term in the Monodomain equation, using a semi-implicit scheme. We also use suitable second order extrapolations (denoted in the following with the subscript ext). Denoting with the superscript k+1 the solution variables at the (k+1)th timestep, the resulting, fully discrete problem is the following: at each timestep,

1. Solve $\forall \mathrm{m} = 1, \dots, \mathrm{s}$ and $\forall \mathrm{j} = 1, \dots, \mathrm{N_e}$:

$$\alpha \frac{\boldsymbol{w}^{\mathrm{j,k+1}}_{\mathrm{h,m}} - \boldsymbol{w}^{\mathrm{j,k+1}}_{\mathrm{BDF,m}}}{\Delta t} = \boldsymbol{F}(u^{\mathrm{j,k+1}}_{\mathrm{m,ext}}, \boldsymbol{w}^{\mathrm{j,k+1}}_{\mathrm{ext}})\,,$$

$$\alpha \frac{\boldsymbol{c}^{\mathrm{j,k+1}}_{\mathrm{m}} - \boldsymbol{c}^{\mathrm{j,k+1}}_{\mathrm{BDF,m}}}{\Delta t} = \boldsymbol{G}(u^{\mathrm{j,k+1}}_{\mathrm{m,ext}}, \boldsymbol{w}^{\mathrm{j,k+1}}_{\mathrm{ext}}, \boldsymbol{c}^{\mathrm{j,k+1}}_{\mathrm{ext}})\,,$$

in which $u^{\mathrm{j,k+1}}_{\mathrm{m,ext}}$ denotes the evaluation of the extrapolated numerical solution for the transmembrane potential at the quadrature point $\boldsymbol{x}^{\mathrm{j}}_{\mathrm{m}}$;

2. compute the ionic current as in (6):

$$I^{\mathrm{k+1}}_{\mathrm{ion}} = \sum_{\mathrm{j=1}}^{\mathrm{N_e}} \sum_{\mathrm{m=1}}^{\mathrm{s}} I_{\mathrm{ion}}(u^{\mathrm{j,k+1}}_{\mathrm{m,ext}}, \boldsymbol{w}^{\mathrm{j,k+1}}_{\mathrm{h,m}}, \boldsymbol{c}^{\mathrm{j,k+1}}_{\mathrm{h,m}}) R_{\mathrm{i}}(\boldsymbol{x}^{\mathrm{j}}_{\mathrm{m}}) \omega_{\mathrm{m}}\,;$$

3. solve the fully discrete Monodomain equation:

$$\chi C_{\mathrm{m}} M \alpha \frac{\boldsymbol{U}^{\mathrm{k+1}} - \boldsymbol{U}^{\mathrm{k+1}}_{\mathrm{BDF}}}{\Delta t} + K\boldsymbol{U}^{\mathrm{k+1}} = \boldsymbol{I}_{\mathrm{app}} - \chi \boldsymbol{I}_{\mathrm{ion}}\,.$$

We remark that, in the above equations, the quantities with subscripts BDF and ‘ext’ are linear combinations of the solution at previous timesteps, representing the known terms in the discrete time derivative and the extrapolation from previous timesteps respectively. Therefore, these quantities are known when computing the solution at timestep $\mathrm{k}+1$.

Although we do not expect the time discretization to be of order 2, due to the splitting of ionic and monodomain equations, the use of BDF of order 2 increases the accuracy in time. This allows to obtain accurate results with a larger timestep than the one required by lower order time discretization schemes, such as implicit Euler, without additional computational costs [76].

6. Numerical results

We report the numerical results of our electrophysiological simulations on both atria and ventricles.

All the numerical simulations require the reconstruction of fibers. This can be achieved by interpolating diffusion tensor imaging (DTI) data on the computational mesh [6,67]. However, in case DTI data are not available or too noisy, it is necessary to reconstruct the fiber orientations through suitable rule-based algorithms [77–79]. For each geometry, we discuss the used algorithm for the development of suitable fiber fields.

We have developed our code in the framework of the C++ IGA library Isoglib. We use the HPC infrastructure made available at MOX, Department of Mathematics, Politecnico di Milano.

6.1. Atria

6.1.1. Fiber generation

The fiber field on the atria is reconstructed following a rule-based method [10,35,76]. We solve the following Laplace–Beltrami problem:

$$\begin{cases} -\Delta_{\Omega}\varphi = f & \text{in } \Omega\,, \\ \nabla_{\Omega}\varphi \cdot \boldsymbol{n} = 0 & \text{on } \partial\Omega\,, \end{cases} \tag{7}$$

where ∇_{Ω} and Δ_{Ω} indicate the surface gradient operator and the Laplace–Beltrami operator respectively, whereas f is a scalar field satisfying the following compatibility condition:

$$\int_{\Omega} f \, d\Omega = 0\,.$$

Given the homogeneous Neumann boundary condition, the solution φ to Eq. (7) is defined up to a constant. Therefore, we add the constraint $\int_{\Omega} \varphi \, d\Omega = 0$, which is imposed with a Lagrange multiplier from the numerical viewpoint. Starting from φ, we can compute the direction of the fibers as the normalized gradient of the solution:

$$\boldsymbol{f}_0 = \frac{\nabla_{\Omega}\varphi}{|\nabla_{\Omega}\varphi|}\,.$$

Thanks to the homogeneous Neumann conditions, the resulting field is tangential to the boundary of the atria (corresponding to the cutting sections of the incoming and outgoing vessels and valves). This is consistent both

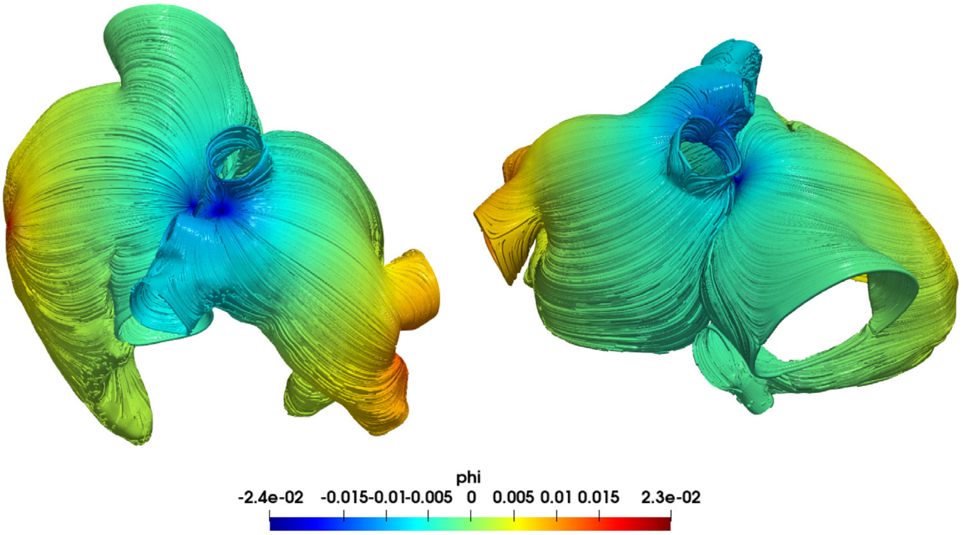


Fig. 10. The computed fiber field for the atria; the color is the solution of the Laplace–Beltrami problem Eq. (7). Function φ achieves its maximum and minimum in $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively. These two points correspond to the values $\varphi = 2.3 \cdot 10^{-2}$ and $\varphi = -2.4 \cdot 10^{-2}$ respectively.

with histological observations [80,81] and with other, more sophisticated rule-based methods [76]. The function f is defined as follows:

$$f(\boldsymbol{x}) = \begin{cases} 1 & \text{if } |\boldsymbol{x} - \boldsymbol{x}_1| < \delta \,, \\ -1 & \text{if } |\boldsymbol{x} - \boldsymbol{x}_2| < \delta \,, \\ 0 & \text{otherwise} \,, \end{cases}$$

where the two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are defined on each of the atria following [35]: on the left atrium, one point is located between the left pulmonary veins, the other is between the right pulmonary veins. On the right atrium, one point is located on the auricle, the other is on the top part, opposite to the tricuspid valve. The resulting fiber field is shown in Fig. 10.

6.1.2. *Electrophysiology*

Atria electrophysiology is simulated using the Monodomain equation coupled with the CRN ionic model. The simulation ran for 1 s, with a timestep $\Delta t = 10\,\mu\text{s}$. The mesh for the atria is comprised of 283'905 elements (162'234 for the right atrium, 121'671 for the left), corresponding to a mesh size of about $h = 0.0005\,\text{m}$. For the atrial simulation, we use two cluster nodes endowed with 20 cores each (Intel Xeon ES-2640 CPUs) and 128 GB of available RAM.

The linear system deriving from the discretization of the Monodomain equation is solved using the GMRES method preconditioned with the ILUT factorization [82,83], with a tolerance of 10^{-6} on the normalized residual.

The right atrium is stimulated in correspondence of the SAN by an applied current of 100 mA for 1 ms. This is meant to mimic its pacemaker role. The stimulation of the left atrium, on the other hand, is triggered according to the electric pattern of the right atrium, to simulate the muscular bundles that connect electrically the two atria (Bachmann's bundle, anterior and posterior septum, coronary sinus musculature [4]). Following [35], for each bundle, we check when the transmembrane potential reaches a suitable threshold ($u > 0\,\text{mV}$) at the junction between the bundle and the right atrium. When it happens, we apply a stimulus to the corresponding point on the left atrium. As for the left and right ventricles, stimuli, both on the right and on the left atrium, are applied as external currents of 100 mA with a duration of 1 ms. The locations of bundles and stimulation points are shown in Fig. 11.

In Fig. 12 we report the evolution of the transmembrane potential and intracellular calcium concentration. From Fig. 13(a) we can observe how the front propagates preferentially in the direction of the fibers, in accordance with the model. The wave front is captured smoothly and is not hindered by patch interfaces or collisions (Fig. 13(b)).

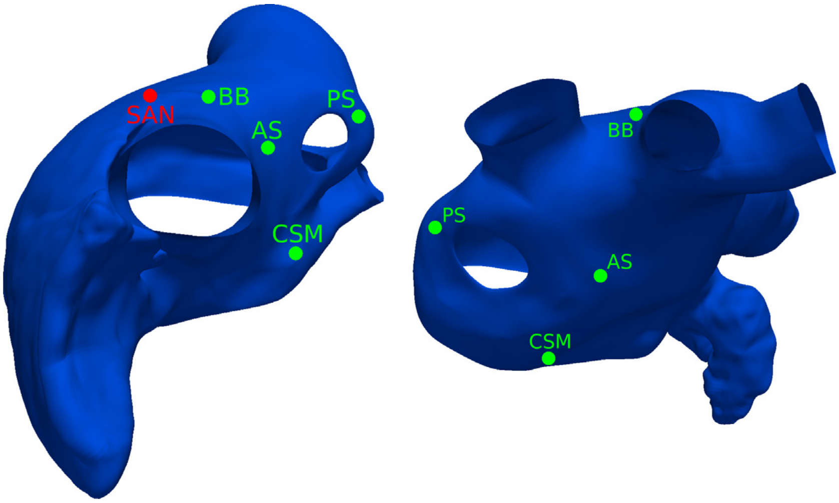


Fig. 11. Stimulation point corresponding to the SAN on the right atrium and position of the muscular bundles connecting the two atria: Bachmann's bundle (BB), anterior septum (AS), posterior septum (PS), coronary sinus musculature (CSM).

In Fig. 14 the evolution of the transmembrane potential at three points on the atria is reported. It can be seen how the action potential behaves in the same way at all the points, up to a time shift due to different activation times, reproducing correctly the atrial hump before the plateau phase.

6.2. *Left and right ventricles*

6.2.1. *Fiber generation*

The fibers on the left and right ventricles are generated according to the rule-based algorithm by Bayer et al. [77], and are reported in Fig. 15(a). The algorithm is based on solving four Laplace problems on the ventricular geometry, with different boundary conditions, whose solutions are depicted in Fig. 15(b).

6.2.2. *Electrophysiology*

Electrophysiological simulations on the left and right ventricles are modeled by the Monodomain equation coupled with the LR ionic model. The simulation ran for 800 ms, with a timestep $\Delta t = 10\,\mu\text{s}$. The mesh representing left and right ventricles is composed of 801'216 elements, corresponding to a mesh size of about $h = 0.0014\,\text{m}$. We use 48 cores on a single cluster node, endowed with Intel Xeon Platinum 8160 processors and a total amount of 1792 GB of available RAM.

The linear system resulting from the discretization is solved by the GMRES method and ILUT preconditioner [82,83], with a tolerance of 10^{-6}.

For the application of the stimuli, we qualitatively replicate the effect of the Purkinje network by applying a current at three points of the left ventricle endocardium and in one point of the right ventricle endocardium, as shown in Fig. 16. Each stimulus consists of an applied current of 100 mA for 1 ms.

In Fig. 17 we report the space evolution of transmembrane potential and calcium concentration on the left and right ventricles for different timesteps of the numerical simulation. In Fig. 18 we depict the evolution in time of the same two electrophysiological variables in two different points of the mesh. We can appreciate the detailed representation of all the action potential phases, including both the upstroke and the downstroke just after depolarization. Moreover, the calcium behavior in time follows the LR ventricular physiological dynamics, which is significantly different from the atrial ones, detected by the CRN ionic model.

6.3. *Comparison with the single patch case*

Given the complexity of the realistic geometries that we consider in this paper, it is not feasible to reconstruct them using a single NURBS patch, at least without considerable modeling efforts. Therefore, we cannot provide a rigorous and detailed comparison between the single patch and the multipatch approaches for this specific scenario.

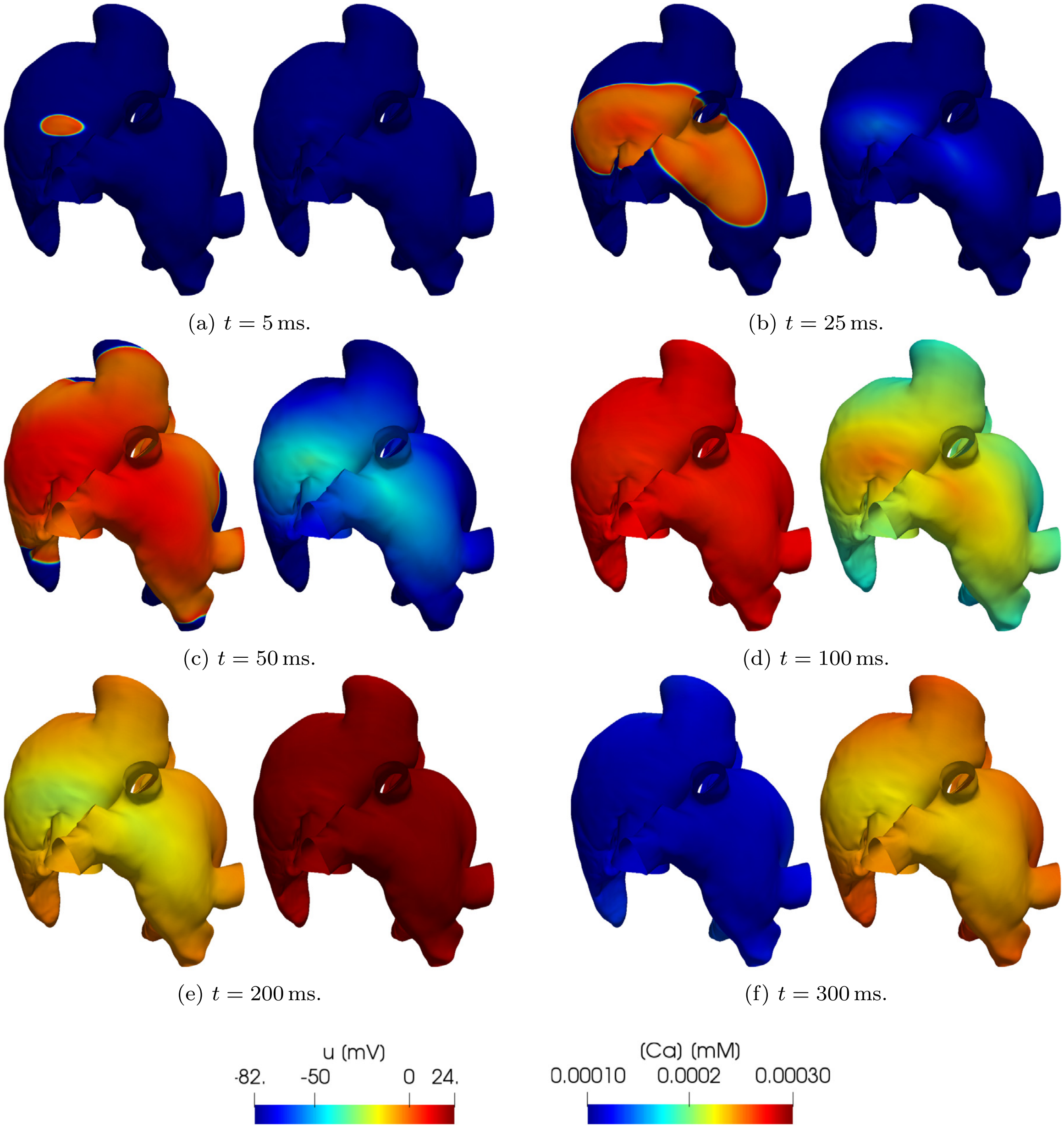
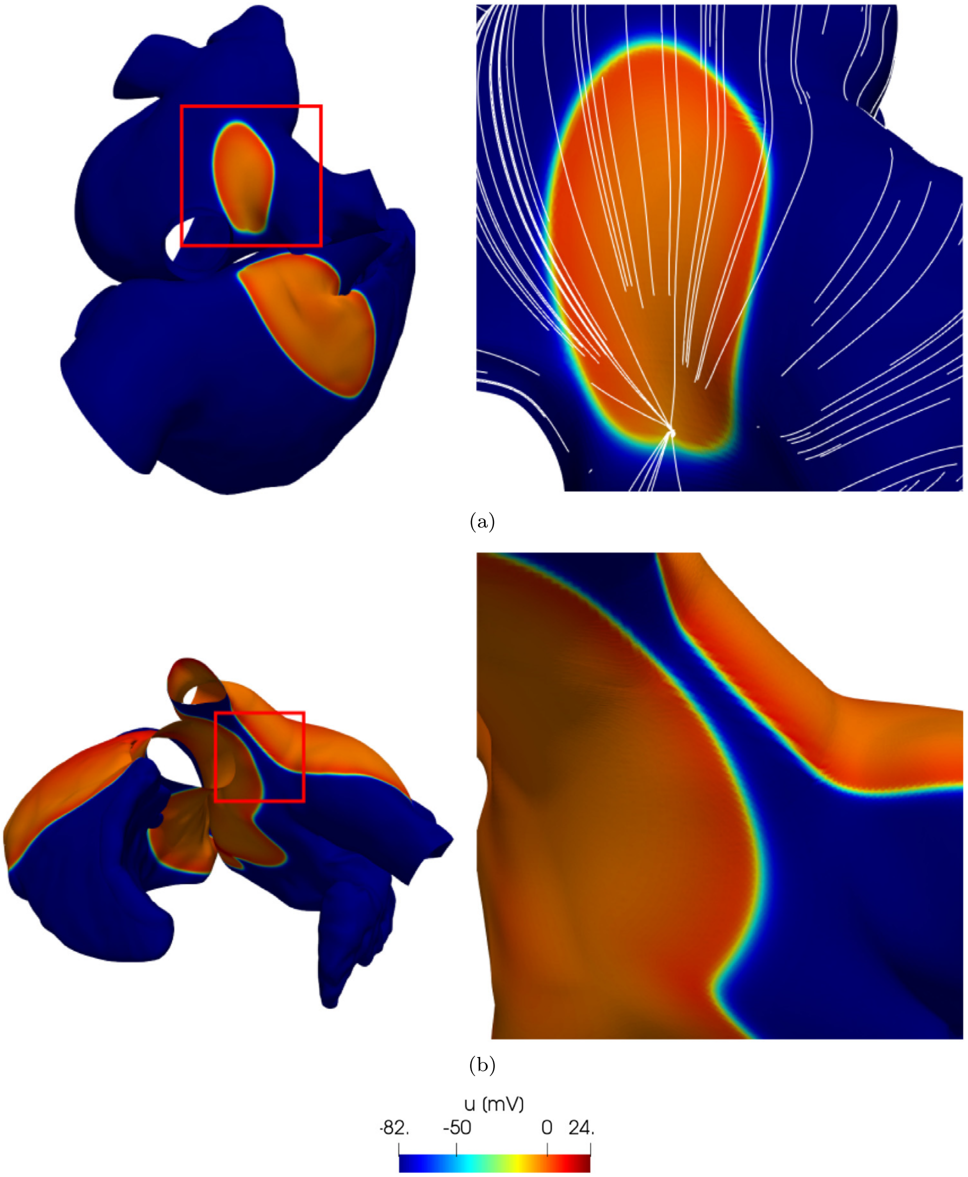

(a) $t = 5\,\text{ms}$.
(b) $t = 25\,\text{ms}$.
(c) $t = 50\,\text{ms}$.
(d) $t = 100\,\text{ms}$.
(e) $t = 200\,\text{ms}$.
(f) $t = 300\,\text{ms}$.

Fig. 12. Transmembrane potential (left) and intracellular calcium concentration (right) on the atria, computed using Monodomain equation coupled with the CRN ionic model.

In Fig. 19 we depict the mesh of a single patch idealized model of two ventricles (taken from [42]). We observe that the elements are necessarily heavily distorted, especially all over the interventricular septum. This distortion dramatically reduces the accuracy of the solution. In particular, it causes significant unphysical under- and overshoots in the transmembrane potential [42]. These oscillations do not appear in the multipatch case, although our geometries are realistic and much more complicated to model.

In general, a single patch approach allows to obtain a higher order of global continuity of the basis functions. Conversely, the multipatch approach described in this work achieves only C^0 global continuity, since we only impose C^0 continuity at patch interfaces. The downside is that accuracy per degree-of-freedom may be reduced and numerical dispersion may increase [10,35], even if in our computational study we did not observe appreciable effects.


(a)

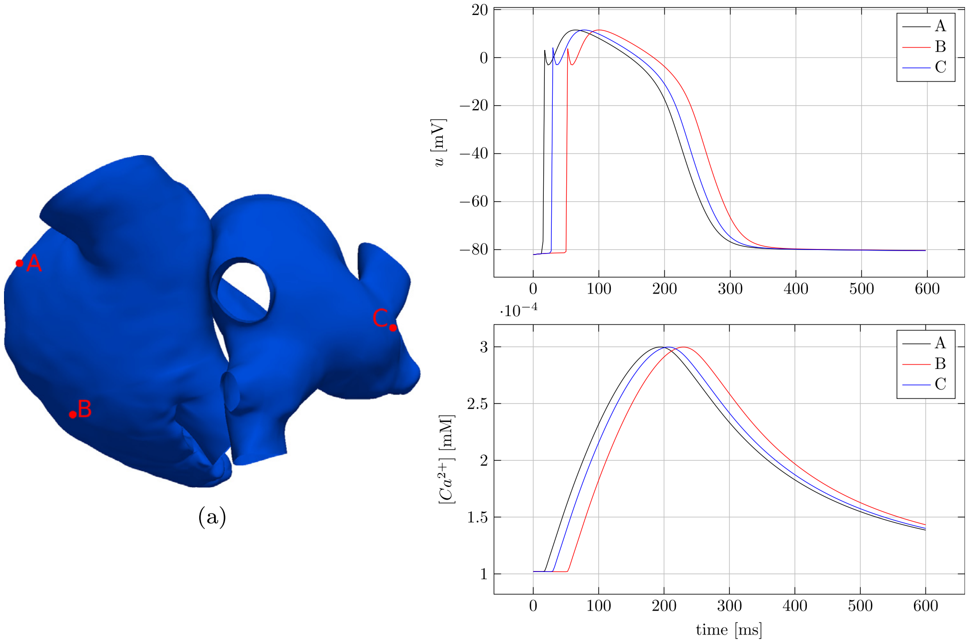
(b)

Fig. 13. (a): close-up of the electrophysiological simulation in the left atrium, near the location of Bachmann's bundle; the color is the transmembrane potential, whereas white lines are the fibers; (b): close-up of the electrophysiological simulation in the left atrium, highlighting the collision between different fronts. We can also appreciate the smoothness of the transmembrane potential wave propagation.

6.4. Simulating the full heart

Computational models for the electrophysiology of the whole heart can be found in [69,76,84,85]. We show here the results obtained by combining the simulation of the atria with the one of the ventricles. For the sake of simplicity, we do not consider the outflow tracts, nor the papillary muscles in the geometrical reconstruction of the ventricles, as done in Section 6.2. Indeed, these cardiac structures account for minor effects on electrophysiological simulations without rhythm disorders.

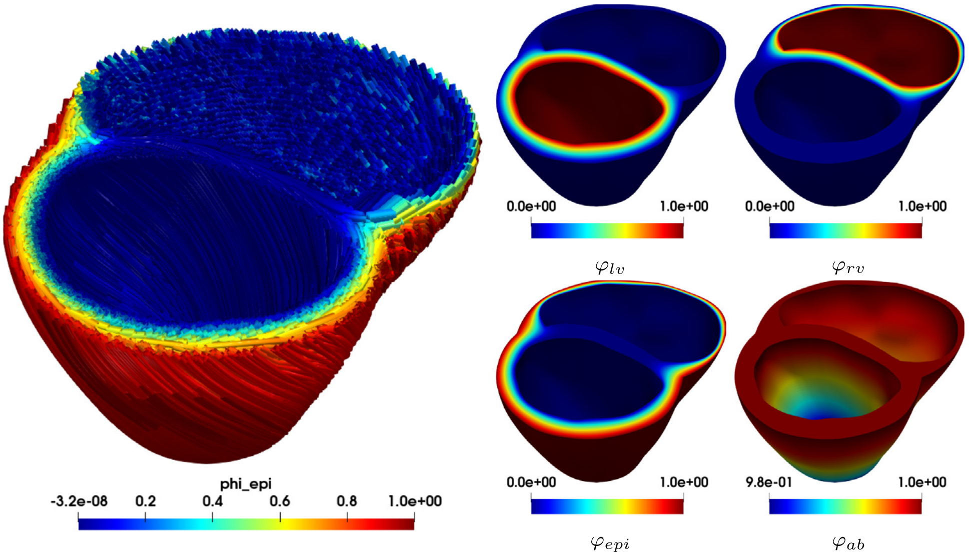
The simulations on the atria and the ventricles are run separately, by properly timing the stimuli to replicate the delay between the activation of the atria and of the ventricles (due to the effect of the atrioventricular node and of the Purkinje network, which are not modeled explicitly). This is achieved by applying the stimuli on the ventricles



(a)

(b)

Fig. 14. Evolution of both transmembrane potential and intracellular calcium concentration at three points on the atria, according to the Monodomain and CRN models. The three points, shown in (a), were chosen sufficiently far from the SAN and from the interatrial bundles on the left atrium, to minimize the effect of the applied currents.



(a)

(b)

Fig. 15. (a): computed fiber field for the left and right ventricles; (b): solutions of the Laplace problems involved in the definition of the fiber field.

approximately 200 ms after applying them on the atria. The results of the two simulations are then combined during post-processing. This is consistent with the fact that atria and ventricles are electrically isolated [6], and that the

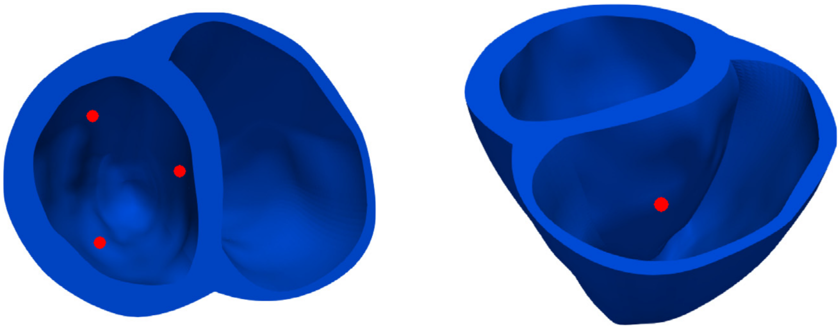
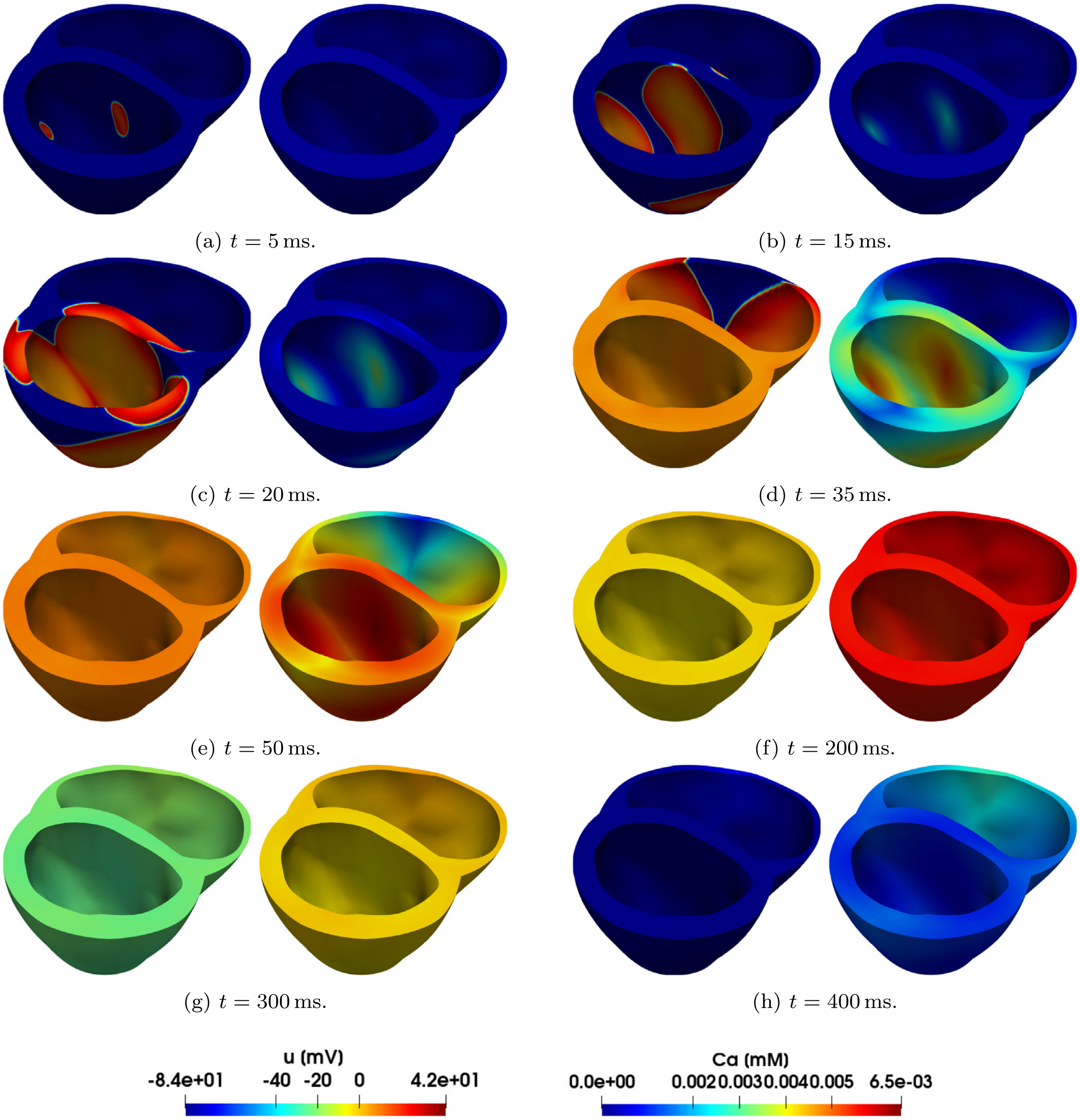


Fig. 16. Stimulation points on the left and right ventricles.



(a) $t = 5\,\text{ms}$.

(b) $t = 15\,\text{ms}$.

(c) $t = 20\,\text{ms}$.

(d) $t = 35\,\text{ms}$.

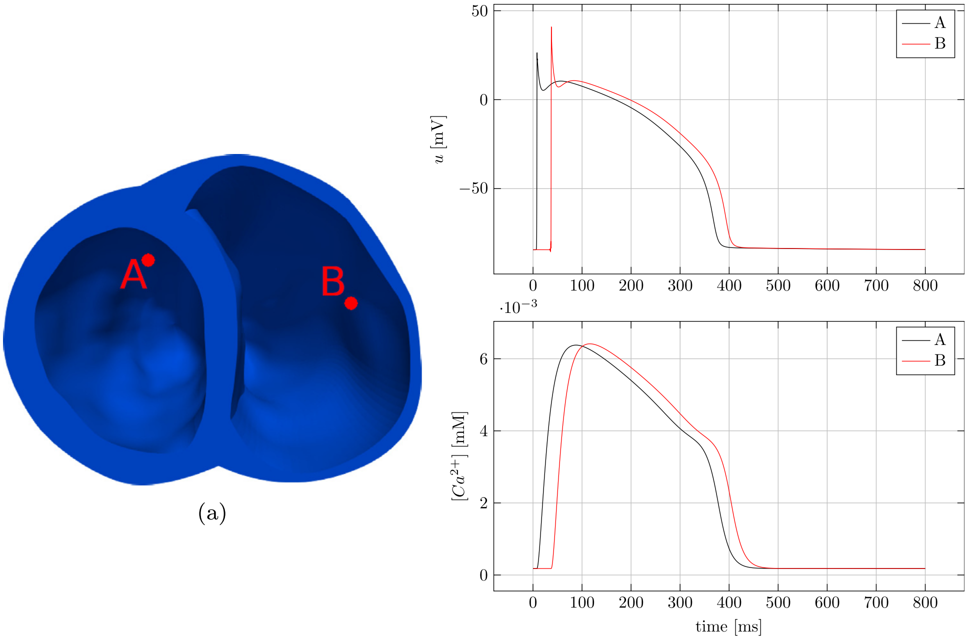
(e) $t = 50\,\text{ms}$.

(f) $t = 200\,\text{ms}$.

(g) $t = 300\,\text{ms}$.

(h) $t = 400\,\text{ms}$.

Fig. 17. Transmembrane potential (left) and intracellular calcium concentration (right) on the left and right ventricles, computed using Monodomain equation coupled with the LR ionic model.



(a)

(b)

Fig. 18. Evolution of both transmembrane potential and intracellular calcium concentration at two points on the left and right ventricles, according to the Monodomain and LR models. The two points, shown in (a), were chosen sufficiently far from the stimulation points, to minimize the effect of the applied currents.

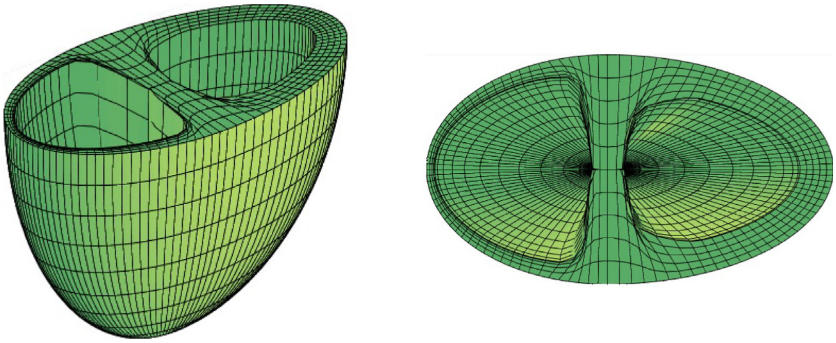


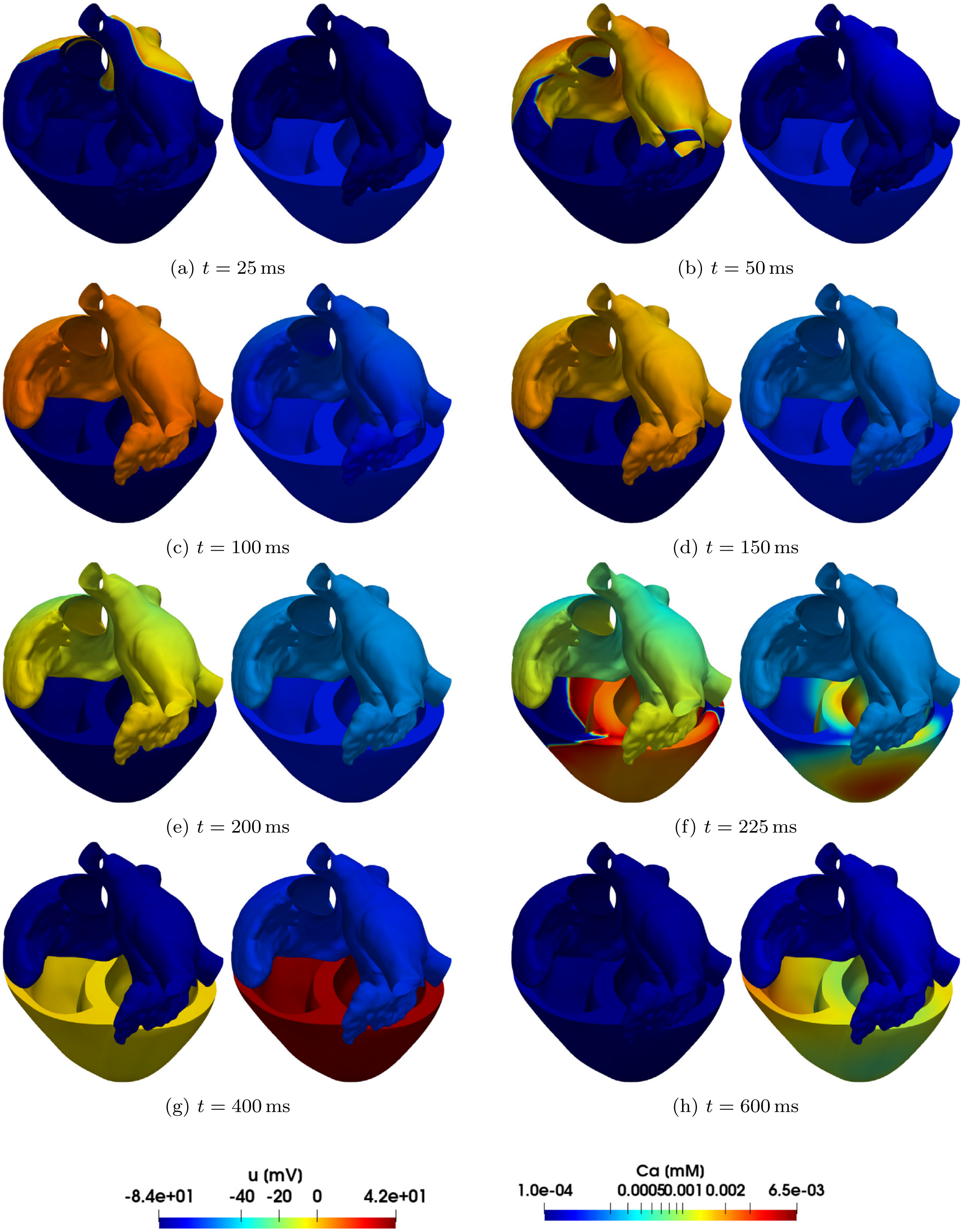
Fig. 19. A single patch idealized model for the two ventricles, at a coarse refinement level. The elements near the interventricular septum are heavily distorted.
Source: Taken from [42].

activation is propagated from atria to ventricles through the cardiac conduction system, specifically through the atrioventricular node and the Purkinje network.

Results are reported in Fig. 20, where we display the evolution of both the transmembrane potential and the calcium concentration over a realistic four chambers representation of the human heart, considering one full cardiac cycle. Gating and ionic variables are described using a different ionic model according to the specific chamber at hand, which leads to a time development of all electrophysiological variables specialized on the different parts of the heart, as shown in both Figs. 14 and 18.

7. Conclusions

In this work, we apply multipatch IGA to the spatial approximation of the Monodomain model on realistic computational models of human atria and ventricles. We propose different techniques for the preprocessing of CAD



(a) $t = 25\,\text{ms}$

(b) $t = 50\,\text{ms}$

(c) $t = 100\,\text{ms}$

(d) $t = 150\,\text{ms}$

(e) $t = 200\,\text{ms}$

(f) $t = 225\,\text{ms}$

(g) $t = 400\,\text{ms}$

(h) $t = 600\,\text{ms}$

Fig. 20. Transmembrane potential (left) and calcium concentration (right) for the whole heart, obtained by combining the atrial and ventricular simulations during post-processing. The color scale for the calcium concentration is logarithmic, in order to accommodate the different orders of magnitude predicted by the CRN and the LR ionic models. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

geometries that we employ to construct multipatch parametrizations of realistic cardiac geometries. These techniques result in IGA conforming multipatch surface and volumetric representations of the computational domain, that can then be used for the discretization of differential problems modeling cardiac physiology. Then, we perform numerical simulations of cardiac electrophysiology both with atria and ventricles, using the Courtemanche–Ramirez–Nattel and the Luo–Rudy ionic models respectively to reproduce the physiological electrical behavior both at a macroscopic scale and at the level of single cardiomyocytes. In both cases we use suitable rule-based algorithms to reconstruct the cardiac muscular fibers. By inspecting our numerical results, we find that the high mesh quality and regularity that can be obtained by the multipatch IGA technique, together with the high-order continuity featured by NURBS basis functions, result in smooth excitation fronts and an accurate solution of the electrophysiology problem, thus improving the control of numerical dispersion enabled by IGA (which was formerly highlighted in [10,35] for the single patch case). For this reason multipatch IGA on complex domains proves to be more suitable than FEM for wave propagation problems involving multiple time and space scales.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 740132, iHEART — An Integrated Heart Model for the simulation of the cardiac function, P.I. Prof. A. Quarteroni).

Appendix. Algorithms for geometry preprocessing

We report the pseudocode for the algorithms introduced in Section 4.

A.1. Patch-level non-conformities

The following algorithm detects patch-level non-conformities, if any, between two patches. This is done by counting the vertices that two patches share and the vertices that one patch may have on the sides of the other patch. We remark that this approach is not fully general. In particular, the case of a single patch that has two sides in contact is not covered, and we assume that such situation does not occur in the CAD model. Moreover, we do not deal with the case of two patches sharing two adjacent vertices and only a portion of the side connecting them, although that situation can be treated similarly by suitably splitting the patches. That case does not occur in the Zygote model. Therefore, we assume that if two patches share two adjacent vertices they are either conforming or not adjacent, and require no modifications.

Algorithm 1: Detection of adjacent patches that are not conforming at the patch level

```
procedure DETECTPATCHLEVELNONCONF(p1, p2)
Input p1, p2 - the two patches
Output result - one of VALID, NON_CONFORMING

    extract vertices of both patches
    if the patches have two common, adjacent vertices then
        result = VALID
```

```
   else if the patches have one common vertex then
      P = common vertex
      if p2 has a vertex on one side of p1 then
         result = NON_CONFORMING
      else if p1 has a vertex on one side of p2 then
         result = NON_CONFORMING
      else
         result = NOT_ADJACENT
      end if
   else
      if p1 has a vertex on one side of p2 and
         p2 has a vertex on one side of p1 then
         result = NON_CONFORMING
      else if p2 has two vertices on one side of p1 then
         result = NON_CONFORMING
      else if p1 has two vertices on one side of p2 then
         result = NON_CONFORMING
      else
         result = VALID
      end if
   end if
end procedure
```

Since we aim at manipulating sets of patches to enforce their conformity, it is useful to retrieve the set of non-conforming patches among them. Algorithm 2 performs that by taking as input two sets of patches `p1` and `p2`, and returning all pairs of patches, one from `p1` and one from `p2`, that are non-conforming at the patch level.

Algorithm 2: List all non-conforming pairs of patches in a set

```
procedure listPatchLevelNonConf(p1, p2)
Input p1, p2 - two sets of patches
Output pairs - a set of pairs [i,j] such that p1[i] and p2[j]
are patch-level non-conforming
   n1 = length(p1)
   n2 = length(p2)
   pairs = [ ]
   for i = 1, ..., n1 do
      for j = 1, ..., n2 do
         if detectPatchLevelNonConf(p1[i],p2[j]) == NON_CONFORMING then
            insert [i,j] into pairs
         end if
      end for
   end for
end procedure
```

To solve a non-conformity between two patches, we need to split one of the two patches in correspondence of hanging nodes. For simplicity, the following algorithm deals only with the case of the two patches sharing a vertex and `patch1` having a vertex on the side of `patch2`. The symmetric case and the cases of non-conformity without shared vertices detected by Algorithm 1 are handled in the same way. In the pseudocode that follows, we denote by `p.knots[i]` the knot vector of patch `p` along the `i`th parametric direction (with `i` either 1 or 2). Assume to have a procedure `splitPatch(patch,direction)` that returns the patches obtained by splitting `patch` in correspondence of the already existing knots with multiplicity $p + 1$, if any, along the parametric direction `direction`.

Algorithm 3: Solve a patch-level non-conformity between two patches

```
procedure solvePatchLevelNonConf(patch1, patch2)
Input patch1, patch2 - the two patches
```

```
    Output patches - set of resulting patches
    Require patch1 and patch2 have a common vertex
    Require patch1 has a vertex on one side of patch2 (the symmetric
    case is the same; other cases are similar)
      p = polynomial degree of patch2
      direction2 = parametric direction of the interface in patch2
      A = common vertex to patch1 and patch2
      B = hanging vertex

      xi = coordinate of B on the side of patch2
      insert xi into patch2.knots[direction2] with multiplicity p+1

      [patch2a, patch2b] = splitPatch(patch2,direction2)
      patches = [patch1, patch2a, patch2b]
    end procedure
```

Finally, the following algorithm deals with the solution of all non-conformities between patches in a set, by iteratively solving non-conformities between pairs of patches using Algorithm 3. For efficiency, at each step we check only the modified patches for new non-conformities (Algorithm 4, lines 8–14). It is possible that the algorithm cannot solve all non-conformities. For example, if there is a loop of patches in which a single non-conformity is present, the algorithm is not able to eliminate such non-conformity in a finite number of steps. The same holds true if such configuration is generated during the execution of the algorithm. For this reason, we force the algorithm to stop after a prescribed maximum number of iterations `nIterMax`. In that case, manual modifications are needed to eliminate the non-conformity issues. This situation does not occur with the Zygote heart model.

Algorithm 4: Detect and solve patch-level non-conformities

```
 1: procedure PatchLevelNonConformities(P)
    Input P - the set of patches
    Output modifies P solving patch-level non-conformities

 2:   NC = listPatchLevelNonConf(P, P)
 3:   nIter = 0
 4:   while  nIter < nIterMax and length(NC) > 0  do
 5:      Pnew = solvePatchLevelNonConf(P[NC[1][1]],P[NC[1][2]])
 6:      remove patches P[NC[1][1]] and P[NC[1][2]] from P
 7:      insert patches Pnew into P
 8:
 9:      remove all pairs containing either NC[1][1] or NC[1][2]
         from NC
10:
11:      NCnew1 = listPatchLevelNonConf(P, Pnew)
12:      NCnew2 = listPatchLevelNonConf(Pnew,Pnew)
13:
14:      append NCnew1 to NC
15:      append NCnew2 to NC
16:
17:      nIter = nIter + 1
18:   end while
19:
20:   if nIter < nIterMax then
21:      algorithm is successful
22:   else
23:      algorithm has failed, manual intervention needed
24:   end if
25: end procedure
```

A.2. Knot-level non-conformities

The following procedure eliminates knot-level non-conformities within a group of patches that should be conforming, as shown in Fig. 5 and discussed in Section 4.1.3. The algorithm also deals with possible different orientation of patch interfaces, by tracking the relative orientation of patches in the variable `orientationFlag`. In order to check whether or not two patches have the same orientation along their interface, it is enough to check the parametric coordinates of the vertices they share (e.g. if the vertices of coordinates $(0, 0)$ and $(1, 0)$ of one patch coincide with those of coordinates $(0, 1)$ and $(1, 1)$ respectively on the other patch, they have the same interface orientation; if the correspondence is inverted, the parametrizations of the interface on the two patches have opposite orientation).

Assume to have a routine `reverse(vector)` that returns the given vector in reverse order. The procedure `nrbkntset` is defined in Appendix A.3.

Algorithm 5: Removal of a knot-level non-conformity in a group of patches

```
procedure SolveKnotLevelNonConf(patches,sides,epsilon,h)
Input patches - the set of patches containing knot-level non-
conformities, ordered so that adjacent patches are adjacent in the set
Input sides - for each patch the parametric side corresponding to
the interface
Input epsilon - the threshold parameter for joining knots
Input h - the desired element size
Output modifies patches solving non-conformities

    n = length(patches)
    Xi = patches[1].knots[sides[1]                                ▷ union of all knot vectors
    orientationFlag = 1

    for i = 2, ..., n do
        if patches i and i-1 have different interface orientation then
            orientationFlag = 1 - orientationFlag
        end if

        if orientationFlag == 1 then
            Xi = union of Xi and patches[i].knots[sides[i]]
        else
            Xi = union of Xi and (1 - patches[i].knots[sides[i]])
        end if
    end for

    if patches 1 and n are adjacent then                ▷ first and last patch, in case of a loop
        if patches 1 and n have different interface orientation then
            orientationFlag = 1 - orientationFlag
        end if

        if orientationFlag == 1 then
            Xi = union of Xi and patches[1].knots[sides[1]]
        else
            Xi = union of Xi and (1 - patches[1].knots[sides[1]])
        end if
    end if

    for each knot span [xi1,xi2] in Xi do
        if |xi1 - xi2| < epsilon * h then
            xiNew = 0.5 * (xi1 + xi2)
            remove xi1 and xi2 from Xi
            insert xiNew into Xi
        end if
    end for

    Eta = reverse(1 - Xi)
    orientationFlag = 1
    for i = 1, 2, ..., n do
        if patches i and i-1 have different interface orientation then
            orientationFlag = 1 - orientationFlag
        end if

        if orientationFlag == 1 then
            patches[i] = nrbkntset(patches[i],sides[i],Xi)
        else
            patches[i] = nrbkntset(patches[i],sides[i],Eta)
        end if
    end for
end procedure
```

A.3. Utility routines for knot vector manipulation

It may be necessary to remove a knot from a patch, for instance if that knot gives rise to excessively thin elements that would hinder the quality of the numerical solution. In general, the removal of a knot cannot be carried out without altering the geometry of the patch [30]. However, in case there are multiple knots that are relatively close, the geometrical changes are small and can be acceptable.

Assume to have defined the following routines:

- `hRefineMatrix(knots1, knots2, p)`, to construct the h-refinement matrix T, that maps control points before knot insertion onto control points after the insertion of one or more knots [26];
- `pseudoInverse(A)`, to compute the least-squares inverse of a matrix.

Then, the removal of a knot can be carried out using the following algorithm:

Algorithm 6: Removal of a knot from a NURBS surface patch

```
procedure NRBKNTREM(patch, side, i)
    Input patch - the NURBS patch
    Input side (1 or 2) - the parametric direction to remove the knot
    from
    Input i - the index of the knot to be removed
    Output modifies patch removing the knot

        p = polynomial degree of patch
        newKnots = patch.knots[side]
        remove ith element from newKnots
        T = hRefineMatrix ( newKnots, patch.knots[side], p )

        patch.knots[side] = newKnots
        patch.ctrlPoints = pseudoInverse(T) * patch.ctrlPoints
end procedure
```

The following algorithm can be applied to simultaneously insert and remove multiple knots from a NURBS surface patch.

Algorithm 7: Multiple knot insertion and removal for a NURBS surface patch

```
procedure NRBKNTSET(patch, side, newKnots)
    Input patch - the patch to be modified
Input side (1 or 2) - the parametric side to be modified
Input newKnots - the desired knot vector
Output modifies patch imposing the desired knot vector

    for each knot xi in newKnots not in patch.knots[side] do
        insert xi into patch.knots[side]
    end for

    for each knot xi in patch.knots[side] not in newKnots do
        patch = nrbkntrem(patch, side, xi)
    end for
end procedure
```

References

[1] R.A. Harrington, J. Narula, Z.J. Eapen, Hurst's the Heart, MacGraw-Hill, 2011.

[2] A.M. Katz, Physiology of the Heart, Wolters Kluwer, 2011.

[3] R. Klabunde, Cardiovascular Physiology Concepts, Lippincott Williams & Wilkins, 2011.

[4] S. Sakamoto, T. Nitta, Y. Ishii, Y. Miyagi, H. Ohmori, K. Shimizu, Interatrial electrical connections: The precise location and preferential conduction, J. Cardiovasc. Electrophysiol. 16 (2005) 1077–1086.

[5] J. Sundnes, G.T. Lines, X. Cai, B.F. Nielsen, K. Mardal, A. Tveito, Computing the Electrical Activity in the Heart, volume 1, Springer, 2007.

[6] P. Colli Franzone, L.F. Pavarino, S. Scacchi, Mathematical Cardiac Electrophysiology, Springer, 2014.

[7] A. Quarteroni, T. Lassila, S. Rossi, R. Ruiz-Baier, Integrated heart-coupling multiscale and multiphysics models for the simulation of the cardiac function, Comput. Methods Appl. Mech. Engrg. 314 (2017) 345–407.

[8] E. Vigmond, F. Vadakkumpadan, V. Gurev, H. Arevalo, M. Deo, G. Plank, N.A. Trayanova, Towards predictive modeling of the electrophysiology of the heart, Exp. Physiol. 94 (2009) 563–577.

[9] Y. Bourgault, C. Pierre, Comparing the bidomain and monodomain models in electro-cardiology through convergence analysis, 2010, HAL Archives-ouvertes.

[10] A.S. Patelli, L. Dede', T. Lassila, A. Bartezzaghi, A. Quarteroni, Isogeometric approximation of cardiac electrophysiology models on surfaces: An accuracy study with application to the human left atrium, Comput. Methods Appl. Mech. Engrg. 317 (2017) 248–273.

[11] A. Quarteroni, L. Dede', A. Manzoni, C. Vergara, Mathematical Modeling of the Human Cardiovascular System: Data, Numerical Approximation, Clinical Applications, Cambridge University Press, 2019.

[12] R.R. Aliev, A.V. Panfilov, A simple two-variable model of cardiac excitation, Chaos Solitons Fractals 7 (1996) 293–301.

[13] A. Bueno-Orovio, E.M. Cherry, F.H. Fenton, Minimal model for human action potentials in tissue, J. Theoret. Biol. 253 (2008) 544–560.

[14] A.L. Hodgkin, A.F. Huxley, A quantitative description of membrane current and its application to conduction and excitation in nerve, J. Physiol. 117 (1952) 500–544.

[15] J.M. Rogers, A.D. McCulloch, A collocation-Galerkin finite element model of cardiac action potential propagation, IEEE Trans. Biomed. Eng. 41 (1994) 743–757.

[16] C.H. Luo, Y. Rudy, A model of the ventricular cardiac action potential. Depolarization, repolarization, and their interaction, Circ. Res. 68 (1991) 1501–1526.

[17] M. Courtemanche, R.J. Ramirez, S. Nattel, Ionic mechanisms underlying human atrial action potential properties: Insights from a mathematical model, Am. J. Physiol. Heart Circ. Physiol. 275 (1998) H301–H321.

[18] P. Colli Franzone, L.F. Pavarino, A parallel solver for reaction-diffusion systems in computational electrocardiology, Math. Models Methods Appl. Sci. 14 (2004) 883–911.

[19] P. Colli Franzone, L.F. Pavarino, S. Scacchi, A numerical study of scalable cardiac electro-mechanical solvers on HPC architectures, Front. Physiol. 9 (2018) 268.

[20] P. Colli Franzone, L.F. Pavarino, B. Taccardi, Simulating patterns of excitation, repolarization and action potential duration with cardiac Bidomain and Monodomain models, Math. Biosci. 197 (2005) 35–66.

[21] Y. Bourgault, Y. Coudiere, C. Pierre, Existence and uniqueness of the solution for the bidomain model used in cardiac electrophysiology, Nonlinear Anal. RWA 10 (2009) 458–482.

[22] E. Vigmond, R. Weber dos Santos, A.J. Prassl, M. Deo, G. Plank, Solvers for the cardiac bidomain equations, Prog. Biophys. Mol. Biol. 96 (2008) 3–18.

[23] S. Krishnamoorthi, M. Sarkar, W.S. Klug, Numerical quadrature and operator splitting in finite element methods for cardiac electrophysiology, Int. J. Numer. Methods Biomed. Eng. 29 (2013) 1243–1266.

[24] A. Bartezzaghi, L. Dede', A. Quarteroni, Isogeometric analysis of high order partial differential equations on surfaces, Comput. Methods Appl. Mech. Engrg. 295 (2015) 446–469.

[25] A. Buffa, G. Sangalli, IsoGeometric Analysis: A New Paradigm in the Numerical Approximation of PDEs, Springer, 2016.

[26] J.A. Cottrell, T.J.R. Hughes, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement, Comput. Methods Appl. Mech. Engrg. 194 (2005) 4135–4195.

[27] J.A. Cottrell, T.J.R. Hughes, Y. Bazilevs, Isogeometric Analysis: Towards Integration of CAD and FEA, Wiley, 2009.

[28] L. Dede', A. Quarteroni, Isogeometric analysis for second order partial differential equations on surfaces, Comput. Methods Appl. Mech. Engrg. 284 (2014) 807–834.

[29] T.J.R. Hughes, G. Sangalli, Mathematics of isogeometric analysis: A conspectus, in: E. Stein, R. Borst, T.J.R. Hughes (Eds.), Encyclopedia of Computational Mechanics, second ed., American Cancer Society, 2017, pp. 1–40.

[30] L.A. Piegl, W. Tiller, The NURBS Book, Springer, 1997.

[31] L. Dede', C. Jäggli, A. Quarteroni, Isogeometric numerical dispersion analysis for two-dimensional elastic wave propagation, Comput. Methods Appl. Mech. Engrg. 284 (2015) 320–348.

[32] P. Gervasio, L. Dede', O. Chanon, A. Quarteroni, Comparing isogeometric analysis and spectral element methods: Accuracy and spectral properties, J. Sci. Comput. 83 (2020).

[33] T.J.R. Hughes, A. Reali, G. Sangalli, Duality and unified analysis of discrete approximations in structural dynamics and wave propagation: Comparison of p-method finite elements with k-method NURBS, Comput. Methods Appl. Mech. Engrg. 197 (2008) 4104–4124.

[34] L.A. Charawi, Isogeometric overlapping Schwarz preconditioners for the bidomain reaction–diffusion system, Comput. Methods Appl. Mech. Engrg. 319 (2017) 472–490.

[35] L. Pegolotti, L. Dede', A. Quarteroni, Isogeometric analysis of the electrophysiology in the human heart: Numerical simulation of the bidomain equations on the atria, Comput. Methods Appl. Mech. Engrg. 343 (2019) 52–73.

[36] Y. Bazilevs, L. Beirao da Veiga, J.A. Cottrell, T.J.R. Hughes, G. Sangalli, Isogeometric analysis: Approximation, stability and error estimates for h-refined meshes, Math. Models Methods Appl. Sci. 16 (2006) 1031–1090.

[37] L. Beirao da Veiga, A. Buffa, J. Rivas, G. Sangalli, Some estimates for h-p-k-refinement in isogeometric analysis, Numer. Math. 118 (2010) 271–305.

[38] A. Collin, G. Sangalli, T. Takacs, Analysis-suitable G^1 multi-patch parametrizations for C^1 isogeometric spaces, Comput. Aided Geom. Design 47 (2016) 93–113.

[39] Zygote 3D Human Anatomy Models, https://www.zygote.com.

[40] D. Bers, Excitation-Contraction Coupling and Cardiac Contractile Force, volume 237, Springer Science & Business Media, 2001.

[41] A.M. Scher, The sequence of ventricular excitation, Am. J. Cardiol. (1964) 287–293.

[42] M. Salvador, Isogeometric Analysis of Cardiac Electrophysiology: Application to the Whole Human Heart, Politecnico di Milano, 2018.

[43] Wikipedia, Electrical conduction system of the heart, https://en.wikipedia.org/wiki/Electrical_conduction_system_of_the_heart.

[44] A. Buffa, J. Rivas, G. Sangalli, R. Vàzquez, Isogeometric discrete differential forms in three dimensions, SIAM J. Numer. Anal. 49 (2011) 814–844.

[45] A. Buffa, R. Vàzquez, G. Sangalli, L. Beirao da Veiga, Approximation estimates for isogeometric spaces in multipatch geometries, Numer. Methods Partial Differential Equations 31 (2014) 422–438.

[46] V.P. Nguyen, C. Anitescu, S.P.A. Bordas, T. Rabczuk, Isogeometric analysis: An overview and computer implementation aspects, Math. Comput. Simulation 117 (2015) 89–116.

[47] J.A. Cottrell, T.J.R. Hughes, A. Reali, Studies of refinement and continuity in isogeometric structural analysis, Comput. Methods Appl. Mech. Engrg. 196 (2007) 4160–4183.

[48] V. Calo, Q. Deng, V. Puzyrev, Dispersion optimized quadratures for isogeometric analysis, J. Comput. Appl. Math. 355 (2019) 283–300.

[49] S. Kleiss, C. Pechstein, B. Jüttler, S. Tomar, IETI - Isogeometric tearing and interconnecting, Comput. Methods Appl. Mech. Engrg. 247–248 (2012) 201–215.

[50] U. Langer, A. Mantzaflaris, S.E. Moore, I. Toulopoulos, Multipatch discontinuous Galerkin isogeometric analysis, in: Isogeometric Analysis and Applications 2014, in: Lecture Notes in Computational Science and Engineering, vol. 107, 2015, pp. 1–32.

[51] M. Kapl, G. Sangalli, T. Takacs, An isogeometric C^1 subspace on unstructured multi-patch planar domains, Comput. Aided Geom. Design 69 (2019) 55–75.

[52] P. Antolin, A. Buffa, M. Martinelli, Isogeometric analysis on V-reps: First results, Comput. Methods Appl. Mech. Engrg. 355 (2019) 976–1002.

[53] P. Antolin, A. Buffa, R. Puppi, X. Wei, Overlapping multi-patch isogeometric method with minimal stabilization, 2019, arXiv:1912.06400.

[54] S. Kargaran, B. Jüttler, S. Kleiss, A. Mantzaflaris, T. Takacs, Overlapping multi-patch structures in isogeometric analysis, Comput. Methods Appl. Mech. Engrg. 356 (2019) 325–353.

[55] A. Nagy, D. Benson, On the numerical integration of trimmed isogeometric elements, Comput. Methods Appl. Mech. Engrg. 284 (2015) 165–185.

[56] NURBS package on Octave Forge, https://octave.sourceforge.io/nurbs/index.html.

[57] C. de Falco, A. Reali, R. Vàzquez, GeoPDEs: A research tool for isogeometric analysis of PDEs, Adv. Eng. Softw. 42 (2011) 1020–1034.

[58] SolidWorks - 3D CAD design software, https://www.solidworks.com/.

[59] H.A. Akhras, T. Elguedj, A. Gravouil, M. Rochette, Isogeometric analysis suitable trivariate NURBS models from standard B-Rep models, Comput. Methods Appl. Mech. Engrg. 307 (2016) 256–274.

[60] C. Heinrich, B. Simeon, E. Pilgerstofer, Swept volume parametrization for isogeometric analysis, in: Conference: Mathematics of Surfaces XIII, 2009.

[61] M. Hsu, C. Wang, A.J. Herrema, D. Schillinger, A. Ghoshal, Y. Bazilevs, An interactive geometry modeling and parametric design platform for isogeometric analysis, Comput. Math. Appl. 70 (2015) 1481–1500.

[62] G. Xu, B. Mourrain, R. Duvigneau, A. Galligo, Analysis-suitable volume parametrization of multi-block computational domain in isogeometric applications, Comput. Aided Des. 45 (2013) 395–404.

[63] Y. Zhang, Y. Bazilevs, S. Goswami, C.L. Bajaj, T.J.R. Hughes, Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow, Comput. Methods Appl. Mech. Engrg. 196 (2007) 2943–2959.

[64] W. Martin, E. Cohen, R. Fish, P. Shirley, Practical ray tracing of trimmed NURBS surfaces, J. Graph. Tools 5 (2000) 27–52.

[65] K. Qin, M. Gong, G. Tong, Fast ray tracing NURBS surfaces, J. Comput. Sci. Tech. 11 (1996) 17–29.

[66] D. Deng, H.J. Arevalo, A. Prakosa, D.J. Callans, N.A. Trayanova, A feasibility study of arrhythmia risk prediction in patients with myocardial infarction and preserved ejection fraction, Europace 18 (2016) 60–66.

[67] V. Gurev, T. Lee, J. Constantino, H. Arevalo, N.A. Trayanova, Models of cardiac electromechanics based on individual hearts imaging data, Biomech. Model. Mechanobiol. 10 (2011) 295–306.

[68] G. Plank, L. Zhou, J.L. Greenstein, S. Cortassa, R.L. Winslow, B. O'Rourke, N.A. Trayanova, From mitochondrial ion channels to arrhythmias in the heart: Computational techniques to bridge the spatio-temporal scales, Phil. Trans. R. Soc. 366 (2008) 3381–3409.

[69] C. Augustin, A. Neic, M. Liebmann, A. Prassl, S. Niederer, G. Haase, G. Plank, Anatomically accurate high resolution modeling of human whole heart electromechanics: A strongly scalable algebraic multigrid solver method for nonlinear deformation, J. Comput. Phys. 305 (2016) 622–646.

[70] H.J. Arevalo, F. Vadakkumpadan, E. Guallar, A. Jebb, P. Malamas, K.C. Wu, N.A. Trayanova, Arrhythmia risk stratification of patients after myocardial infarction using personalized heart models, Nature Commun. 7 (2016) 11437.

[71] D. Deng, A. Prakosa, J. Shade, P. Nikolov, N.A. Trayanova, Characterizing conduction channels in postinfarction patients using a personalized virtual heart, Biophys. J. 117 (2019) 2287–2294.

[72] M. Boulakia, S. Cazeau, M.A. Fernández, J.F. Gerbeau, N. Zemzemi, Mathematical modeling of electrocardiograms: A numerical study, Ann. Biomed. Eng. 38 (2010) 1071–1097.

[73] S. Götepke, E. Kuhl, Computational modeling of cardiac electrophysiology: A novel finite element approach, Internat. J. Numer. Methods Engrg. 72 (2009) 156–178.

[74] P. Pathmanathan, G. Mirams, J. Southern, J. Whiteley, The significant effect of the choice of ionic current integration method in cardiac electro-physiological simulations, Int. J. Numer. Methods Biomed. Eng. 27 (2011) 1751–1770.

[75] A. Quarteroni, Numerical Models for Differential Problems, Springer, 2017.

[76] R. Piersanti, P.C. Africa, M. Fedele, C. Vergara, L. Dede', A.F. Corno, A. Quarteroni, Modelling Cardiac Muscle Fibers in Ventricular and Atrial Electrophysiology Simulations, MOX Report, 29, 2020.

[77] J.D. Bayer, R.C. Blake, G. Plank, N.A. Trayanova, A novel rule-based algorithm for assigning myocardial fiber orientation to computational models, Ann. Biomed. Eng. 40 (2012) 2243–2254.

[78] R. Doste, D. Soto-Iglesias, G. Bernardino, A. Alcaine, R. Sebastian, S. Giffard-Roisin, M. Sermesant, A. Berruezo, D. Sanchez-Quintana, O. Camara, A rule-based method to model myocardial fiber orientation in cardiac biventricular geometries with outflow tracts, Int. J. Numer. Methods Biomed. Eng. (2019) 3185.

[79] S. Rossi, T. Lassila, R. Ruiz-Baier, A. Sequeira, A. Quarteroni, Thermodynamically consistent orthotropic activation model capturing ventricular systolic wall thickening in cardiac electromechanics, Eur. J. Mech. A Solids 48 (2014) 129–142, http://dx.doi.org/10.1016/j.euromechsol.2013.10.009.

[80] R.H. Anderson, A.C. Cook, The structure and components of the atrial chambers, EP Eur. 9 (6) (2007) vi3–vi9.

[81] S. Yen Ho, D. Sanchez-Quintana, J.A. Cabrera, R.H. Anderson, Anatomy of the left atrium: Implications for radiofrequency ablation of atrial fibrillation, J. Cardiovasc. Electrophysiol. 10 (11) (1999) 1525–1533.

[82] A. Quarteroni, R. Sacco, F. Saleri, P. Gervasio, Matematica Numerica, Springer, 2014.

[83] Y. Saad, Iterative Methods for Sparse Linear Systems, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003.

[84] B. Baillargeon, N. Rebelo, D. Fox, R. Taylor, E. Kuhl, The living heart project: A robust and integrative simulator for human heart function, Eur. J. Mech. A Solids 48 (2014) 38–47.

[85] A. Santiago, J. Aguado-Sierra, M. Zavala-Ake, R. Doste-Beltran, S. Gomez, R. Aris, J. Cajas, E. Casoni, M. Vàzquez, Fully coupled fluid-electro-mechanical model of the human heart for supercomputers, Int. J. Numer. Methods Biomed. Eng. 34 (2018) e3140.