Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 283 (2015) 210–223
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Computational vademecums for the real-time simulation of haptic collision between nonlinear solids☆

David González[a,b], Icíar Alfaro[a,b], Carlos Quesada[a], Elías Cueto[a,b,*], Francisco Chinesta[c]

[a] *Aragon Institute of Engineering Research (I3A), Universidad de Zaragoza, Maria de Luna 3, E-50018 Zaragoza, Spain*
[b] *CIBER-BBN-Centro de Investigación Biomédica en Red en Bioingeniería Biomateriales y Nanomedicina, Zaragoza, Spain*
[c] *GEM UMR CNRS - Ecole Centrale de Nantes, 1 rue de la Noë, BP 92101, F-44321 Nantes cedex 3, France*

Received 12 June 2014; received in revised form 9 September 2014; accepted 23 September 2014
Available online 30 September 2014

Abstract

In this paper a novel strategy is presented for the real-time simulation of contact between non-linear deformable solids at haptic feedback rates. The proposed method is somehow related to the *Voxmap Pointshell* method for two deformable solids. Its novelty and crucial advantages over existing implementations of this algorithm come from the intensive use of *computational vademecums*. These are in essence a pre-computed solution of a parametric model in which every possible situation during the on-line phase of the method has been considered through the introduction of the appropriate parameters. Such a high-dimensional parametric model is efficiently solved by using Proper Generalized Decompositions (PGD) and stored in memory as a set of vectors. The paper presents a thorough description of the developed algorithm together with some examples of its performance.

Keywords: Real time; Contact; Model reduction; Proper Generalized Decomposition; Parametric models; Computational vademecums

1. Introduction

Computational contact mechanics [1] constitutes nowadays a very active field of research due to its inherent difficulty, associated to the highly non-linear character of its models. But when we deal with non-linear solids and, in addition, real-time response is required, the problem becomes extremely burdensome and has generated a plethora of

☆ This work has been partially supported by the Spanish Ministry of Science and Competitiveness, through grant number CICYT-DPI2011-27778-C02-01/02. Professor Chinesta is also supported by the Institut Universitaire de France. CIBER-BBN is an initiative funded by the VI National R&D&i Plan 2008–2011, Iniciativa Ingenio 2010, Consolider Program, CIBER Actions and financed by the Instituto de Salud Carlos III with assistance from the European Regional Development Fund.
* Corresponding author at: Aragon Institute of Engineering Research (I3A), Universidad de Zaragoza, Maria de Luna 3, E-50018 Zaragoza, Spain. Tel.: +34 876555253.
E-mail addresses: gonzal@unizar.es (D. González), iciar@unizar.es (I. Alfaro), cquesada@unizar.es (C. Quesada), ecueto@unizar.es (E. Cueto), francisco.chinesta@ec-nantes.fr (F. Chinesta).

http://dx.doi.org/10.1016/j.cma.2014.09.029
0045-7825/

publications searching for a good compromise between accuracy and time to response, see for instance [2–5], to name but a few of the available references.

Haptic peripherals have become very popular for *augmented* simulation in immersive environments, particularly for education purposes and games. They have been used notably in medical environments as an essential tool for the education in minimally invasive procedures [6–13] and for training of complex industrial processes. In particular, aircraft and automobile industries have recently incorporated virtual reality to the design process to see whether a mechanical part could be located at a particular position in the plane in order to facilitate manufacturing and maintenance processes [14–16].

The main difficulty associated to haptic peripherals (those with force feedback) is that they need, to provide with a realistic sense of touch, a feedback response in the order of 500 Hz to 1 kHz. What this means in practice is that, very much like some 25 frames per second are needed in cinemas to provide the spectator with a continuous sensation of movement, in haptic environments nearly one thousand simulations per second must be carried out to provide the user with a continuous sense of touch. 1 kHz is roughly the free-hand gesture frequency [17].

The reader will readily understand the difficulties associated with the simulation of non-linear solids (soft living tissues are frequently assumed to be hyperelastic, possibly with fiber reinforcement [18,19]) under such astringent requirements. If, in addition, the possibility of contact between deformable solids is taken into account, the 1 kHz constraint is even more difficult to fulfill.

Contact mechanics simulations under real-time restrictions have been tackled from a variety of approaches. One of the earliest and most popular is that of constructing Bounding Volume Hierarchies (BVH) [5], consisting in associating each node in a tree with a subset of the primitives (polygons, NURBS, etc.) defining the boundary of the object. Other approaches for real time include the use of stochastic contact detection [20], based on the assumption that the perceived realism of contact detection depends more of the real-time response than in the accuracy itself of the simulation.

One of the most popular families of real-time contact detection algorithms is based on the use of distance fields [21,4]. Distance fields (level sets) constitute a very convenient way of representing very intricate geometries for contact detection, but have been traditionally considered as non-apt for real-time contact simulation between deformable solids, since the distance field must be updated according to the deformation of one of the solids [5].

If we restrict ourselves to the problem of real-time simulation of hyperelastic solids, several approaches have been accomplished in the literature. Besides the obvious choice of considering a purely elastic material, but which renders very poor results in terms of visual realism in the presence of large strains, the first approaches considered multi-resolution methods [22,23]. Very popular in the last years are the methods based on the use of explicit finite element implementations [24], possibly based on parallelization by using general-purpose Graphics Processing Units (GPUs) [25,6,26]. Due to the inherent complexity of the objective, techniques based upon model order reduction methods have also recently reached some popularity. Among them, we can cite [27–32]. In general, all these last references are based on the use of Proper Orthogonal Decomposition (POD) techniques [33–35], known also as Principal Component Analysis (PCA). These techniques employ a statistical treatment of the results obtained for complete, *similar* problems to the one at hand to construct a set of global, Ritz-like, basis functions that are optimal, with respect to some measure, for the already solved, complete problems. They are then used for the problem under consideration in the hope that, if the modifications with respect to the original problems are small, they will be also a good choice for it.

This approach presents some drawbacks. For instance, the choice of *similar*, complete problems, after which to obtain the set of global basis, is not an easy task [36]. In addition, model reduction methods loose many of their advantages if we deal with non-linear problems, since they need for the reconstruction of the full tangent stiffness matrix in order to obtain a consistent linearization of the problem. Although several approaches have been developed to deal with this difficulty, no definitive response has been found [28,37,38].

As an alternative to POD techniques, Proper Generalized Decomposition (PGD) techniques can be seen as a sort of *a priori* model order reduction method, in the sense that no complete problem must be solved in order to obtain the set of global basis functions. The origin of PGD dates back to the so-called *radial loading* approximation of the LArge Time INcrements (LATIN) method [39]. It consisted, essentially, of a space–time separated representation of the solution that very much resembles that of POD, but obtained completely in an *a priori* fashion.

Some years later, F. Chinesta employed a similar separated representation for the solution of high-dimensional problems arising from the kinetic theory of complex, non-Newtonian flows [40,41]. Once both approaches have been identified as belonging to one single method for the model order reduction of PDEs, the field of application of PGD

has grown exponentially. Just to cite some of the most recent survey papers, the interested reader can consult, for instance, [42–48].

Based on PGD, the authors have developed a series of *computational vademecum* approaches [45] for different physical problems such as thermal control of industrial furnaces [49,50], dynamic, data-driven application systems (DDDAS) [51], shape optimization [52] or computational surgery [53], to name a few. In essence, a computational vademecum (from the latin *vade mecum*, "goes with me") is a sort of reference guide that helps engineers (or physicians) by abridging known solutions to given problems. One of the earliest and most known examples is that of Bernoulli [54]. This concept has been translated by the authors to nowadays computational mechanics by computing, off-line and once for life, parametric (and thus multi-dimensional) solutions for a given problem. This general, multi-dimensional solution (a sort of computational response surface or meta-model, if preferred) is then used (evaluated) on-line at very high feedback rates.

The main difficulty with computational vademecums is that they give rise to high-dimensional problems. It is well-known that, for mesh-based methods such as finite elements or finite differences, high-dimensional problems suffer from the so-called *curse of dimensionality* [55], i.e., an exponential increase on the number of degrees of freedom with the number of dimensions of the space. This difficulty is at the very heart of the choice of PGD for such high-dimensional problems, since it assumes the solution as being expressed as a finite sum of separable functions.

In this paper a novel approach to the real-time simulation of contact between non-linear solids is presented. It is based on an intensive use of computational vademecums obtained off-line by means of PGD techniques. The developed technique can be seen as a particular instance of distance field methods, in which a general, high-dimensional distance field will be pre-computed off-line by taking the collision position as a parameter in the model. Thus, the most restrictive characteristic of distance field techniques will therefore be overcome, since there will be no need for distance field updating with the deformation of the solid.

In order to fully understand the developed methodology, a brief review of how to construct a computational vademecum by PGD techniques will be made in Section 2. The novel distance-field method will be presented in Section 3. In Section 4 examples of the performance of the proposed technique will be given. The paper is closed by some final remarks in Section 5.

2. A brief review of computational vademecums obtained by PGD techniques

As stated in the introduction, the main ingredient of the method here reported is the intensive use of computational vademecums [45]. In essence, the idea behind this method is to pre-compute off-line *every* possible situation (with the limitations that will be later clarified) that could be faced during the on-line phase of the simulation.

Consider, as a starting point, the case of a vademecum in which we would like to store the displacement field $\boldsymbol{u}(\boldsymbol{x})$ of a non-linear (here, hyperelastic) solid Ω under the action of a force (assumed for the sake of simplicity as unitary and always acting in the vertical direction) at any point $\boldsymbol{s}$ of its boundary region $\overline{\Gamma} \subset \Gamma_t \subset \Gamma = \partial\Omega$. This renders a problem defined in general in $\mathbb{R}^5$ ($\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{s})$), although if $\boldsymbol{s}$ is interpolated using nearest-neighbor interpolation, it can be seen as a one-dimensional parameter (the node in which the load is acting), rendering a problem in $\mathbb{R}^4$.

In the general case of an arbitrary force acting in any direction, the problem would be stated so as to find the displacement field for any physical point $\boldsymbol{x} \in \mathbb{R}^3$, for a load acting at any point of its boundary (a two-dimensional manifold in $\mathbb{R}^3$) and arbitrary values of its three components, thus $\boldsymbol{t} \in \mathbb{R}^3$. This renders a problem defined in dimension 8.

Consider now the weak form of the equilibrium equations (balance of linear momentum). Again, for the sake of simplicity, we omit inertia terms. For a detailed treatment of non-linear solid dynamics with the use of PGD and computational vademecums, the interested reader can consult [56]. Under these assumptions, the (doubly-) weak form of the problem, extended to the whole solid, Ω and the portion of its boundary which is accessible to load, $\bar{\Gamma} \subset \Gamma_t$, consists in finding the displacement $\boldsymbol{u} \in \mathcal{H}^1$ such that for all $\boldsymbol{u}^* \in \mathcal{H}_0^1$:

$$\int_{\bar{\Gamma}} \int_{\Omega} \nabla_s \boldsymbol{u}^* : \boldsymbol{\sigma}\, d\Omega\, d\bar{\Gamma} = \int_{\bar{\Gamma}} \int_{\Gamma_{t2}} \boldsymbol{u}^* \cdot \boldsymbol{t}\, d\Gamma\, d\bar{\Gamma} \tag{1}$$

where $\Gamma = \Gamma_u \cup \Gamma_t$ represents the boundary of the solid, divided into essential and natural regions, and where $\Gamma_t = \Gamma_{t1} \cup \Gamma_{t2}$, i.e., regions of homogeneous and non-homogeneous, respectively, natural boundary conditions. Here, $\boldsymbol{t} = -\boldsymbol{e}_k \cdot \delta(\boldsymbol{x} - \boldsymbol{s})$, where δ represents the Dirac-delta function and $\boldsymbol{e}_k$ the unit vector along the z-coordinate axis (we

consider here, as mentioned before, and for the ease of exposition, a unit load directed towards the negative z axis of reference).

The Dirac-delta term is then regularized, and approximated by a truncated series of separable functions in the spirit of the PGD method, i.e.,

$$t_j \approx \sum_{i=1}^{m} f_j^i(\boldsymbol{x}) g_j^i(\boldsymbol{s})$$

where m represents the order of truncation and f_j^i, g_j^i represent the jth component of vectorial functions in space and boundary position, respectively.

PGD techniques allow to efficiently construct the computational vademecum $\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{s})$ by constructing, in an iterative way, an approximation to the solution in the form of a finite sum of separable functions [43]. Let us assume that the method has converged to a solution, at iteration n of this procedure,

$$u_j^n(\boldsymbol{x}, \boldsymbol{s}) = \sum_{k=1}^{n} X_j^k(\boldsymbol{x}) \cdot Y_j^k(\boldsymbol{s}), \tag{2}$$

where the term u_j refers to the jth component of the displacement vector, $j = 1, 2, 3$ and functions $\boldsymbol{X}^k(\boldsymbol{x})$ and $\boldsymbol{Y}^k(\boldsymbol{s})$ represent the separated functions used to approximate the unknown field, obtained in previous iterations of the PGD algorithm. At this stage, the objective of PGD is to provide the solution with an improvement given by the $(n+1)$-th term of the approximation,

$$u_j^{n+1}(\boldsymbol{x}, \boldsymbol{s}) = u_j^n(\boldsymbol{x}, \boldsymbol{s}) + R_j(\boldsymbol{x}) \cdot S_j(\boldsymbol{s}), \tag{3}$$

where $\boldsymbol{R}(\boldsymbol{x})$ and $\boldsymbol{S}(\boldsymbol{s})$ are the sought functions that improve the approximation. In an equivalent manner, admissible variations of this displacement field will be given by

$$u_j^*(\boldsymbol{x}, \boldsymbol{s}) = R_j^*(\boldsymbol{x}) \cdot S_j(\boldsymbol{s}) + R_j(\boldsymbol{x}) \cdot S_j^*(\boldsymbol{s}).$$

As can be noticed, even if the original problem is linear, PGD needs for the solution of a non-linear problem, i.e., to determine a product of functions, see Eq. (3). To this end, any of the possible linearization procedures could be employed. Traditionally, the authors have employed the simplest one, a fixed point, alternating directions algorithms that, although not being optimal, works in general very well. This strategy is briefly described hereafter.

2.1. Computation of $S(\boldsymbol{s})$ assuming $R(\boldsymbol{x})$ is known

In this case, following standard assumptions of variational calculus, we have

$$u_j^*(\boldsymbol{x}, \boldsymbol{s}) = R_j(\boldsymbol{x}) \cdot S_j^*(\boldsymbol{s}), \tag{4}$$

or, equivalently, $\boldsymbol{u}^*(\boldsymbol{x}, \boldsymbol{s}) = \boldsymbol{R} \circ \boldsymbol{S}^*$. The symbol "$\circ$" stands here for the so-called entry-wise, Hadamard or Schur multiplication for vectors. Once substituted into Eq. (1), gives

$$\int_{\bar{\Gamma}} \int_{\Omega} \nabla_s (\boldsymbol{R} \circ \boldsymbol{S}^*) : \mathbf{C} : \nabla_s \left(\sum_{k=1}^{n} \boldsymbol{X}^k \circ \boldsymbol{Y}^k + \boldsymbol{R} \circ \boldsymbol{S} \right) d\Omega d\bar{\Gamma} = \int_{\bar{\Gamma}} \int_{\Gamma_{t2}} (\boldsymbol{R} \circ \boldsymbol{S}^*) \cdot \left(\sum_{k=1}^{m} \boldsymbol{f}^k \circ \boldsymbol{g}^k \right) d\Gamma d\bar{\Gamma},$$

or, equivalently (we omit obvious functional dependencies)

$$\begin{aligned} \int_{\bar{\Gamma}} \int_{\Omega} \nabla_s (\boldsymbol{R} \circ \boldsymbol{S}^*) : \mathbf{C} : \nabla_s (\boldsymbol{R} \circ \boldsymbol{S}) d\Omega d\bar{\Gamma} = & \int_{\bar{\Gamma}} \int_{\Gamma_{t2}} (\boldsymbol{R} \circ \boldsymbol{S}^*) \cdot \left(\sum_{k=1}^{m} \boldsymbol{f}^k \circ \boldsymbol{g}^k \right) d\Gamma d\bar{\Gamma} \\ & - \int_{\bar{\Gamma}} \int_{\Omega} \nabla_s \left(\boldsymbol{R} \circ \boldsymbol{S}^* \right) \cdot \mathcal{R}^n d\Omega d\bar{\Gamma}, \end{aligned}$$

where $\mathcal{R}^n$ represents:

$$\mathcal{R}^n = \mathbf{C} : \nabla_s \boldsymbol{u}^n.$$

Since the symmetric gradient operates on spatial variables only, we have:

$$\int_{\bar{\Gamma}}\int_{\Omega}(\nabla_s \boldsymbol{R}\circ\boldsymbol{S}^*):\mathbf{C}:(\nabla_s \boldsymbol{R}\circ\boldsymbol{S})d\Omega d\bar{\Gamma}$$
$$=\int_{\bar{\Gamma}}\int_{\Gamma_{t2}}(\boldsymbol{R}\circ\boldsymbol{S}^*)\cdot\left(\sum_{k=1}^{m}\boldsymbol{f}^k\circ\boldsymbol{g}^k\right)d\Gamma d\bar{\Gamma}-\int_{\bar{\Gamma}}\int_{\Omega}\left(\nabla_s \boldsymbol{R}\circ\boldsymbol{S}^*\right)\cdot\mathcal{R}^n d\Omega d\bar{\Gamma}$$

where all the terms depending on $\boldsymbol{x}$ are known and hence all integrals over Ω and Γ_{t2} (the part of the natural boundary with non-homogeneous boundary conditions) can be computed to derive an equation to determine $\boldsymbol{S}(\boldsymbol{s})$.

2.2. *Computation of $R(\boldsymbol{x})$ assuming $S(\boldsymbol{s})$ is known*

Equivalently, in this case, we have

$$u_j^*(\boldsymbol{x},\boldsymbol{s})=R_j^*(\boldsymbol{x})\cdot S_j(\boldsymbol{s}),$$

which, once substituted into Eq. (1), gives

$$\int_{\bar{\Gamma}}\int_{\Omega}\nabla_s(\boldsymbol{R}^*\circ\boldsymbol{S}):\mathbf{C}:\nabla_s\left(\sum_{k=1}^{n}\boldsymbol{X}^k\circ\boldsymbol{Y}^k+\boldsymbol{R}\circ\boldsymbol{S}\right)d\Omega d\bar{\Gamma}=\int_{\bar{\Gamma}}\int_{\Gamma_{t2}}(\boldsymbol{R}^*\circ\boldsymbol{S})\cdot\left(\sum_{k=1}^{m}\boldsymbol{f}^k\circ\boldsymbol{g}^k\right)d\Gamma d\bar{\Gamma}.$$

In this case all the terms depending on $\boldsymbol{s}$ (load position) can be integrated over $\bar{\Gamma}$, leading to a generalized elastic problem to compute function $\boldsymbol{R}(\boldsymbol{x})$.

As mentioned before, even if fixed-point algorithms do not have guaranteed convergence properties, it is extremely unfrequent, in our experience, to find examples in which convergence is lost (see [42] and references therein). Other linearization strategies such as Newton–Raphson could also be equally employed [40,41].

This development assumes implicitly small strain measures. For general, hyperelastic constitutive laws, large strain tensors (usually the Green–Lagrange tensor $\boldsymbol{E}$) must be equally linearized. In the past, the authors have tackled this linearization problem within the PGD approach in two different ways. In [53] an explicit approach was developed that renders, in general, very good results without stability problems. In [57] an approach was developed based on the combined use of PGD and Asymptotic Numerical Methods (ANM). In this last approach, the solution $\boldsymbol{u}$ is expanded in terms of a power series of an arc-length parameter, providing a sort of continuation method in which there is no need of updating tangent stiffness matrices. Other approaches such as the Discrete Empirical Interpolation Method [58,37] could also be equally employed.

In general, following the aforementioned works, a strategy is followed in which enough terms in Eq. (2) are included so as to provide a norm of the residual below 10^{-5}. Sometimes, given the strong limitations imposed by haptic response, further simplification of the PGD representation of the solution is needed. In that cases, the number of terms are restricted to those giving an accuracy always above 10%, computed with respect to full FE solutions of the same problem.

In general, we refer the interested reader to the previous works in the field [57,53,56] for a thorough discussion on the relationship between accuracy and number of terms in the PGD representation of the solution.

3. A distance-field method based on the use of computational vademecums

The method here developed assumes that a computational vademecum $\boldsymbol{u}(\boldsymbol{x},\boldsymbol{s})$ has already been computed for the solids under consideration. As mentioned before, this multi-dimensional solution provides a general solution for the displacement field in the solids under an arbitrary load acting on $\bar{\Gamma}$.

The collision detection method assumes that one of the solids, say Ω_2, is modeled as a *pointshell* [4], i.e., a set of boundary points (assumed for simplicity identical to the boundary nodes of the solid model, although this is not strictly necessary) equipped with normals to the surface. As such, the algorithm is not symmetric, since the choice of which body is chosen to be equipped with the pointshell affects the final result of the simulation, albeit slightly. The other solid, Ω_1, is in turn equipped with a signed distance field, see Fig. 1.

In our implementation, following closely [4], at every haptic cycle, contact penalty forces are determined by querying the points of Ω_2 against the distance field associated to Ω_1. Traditionally, this approach has been considered

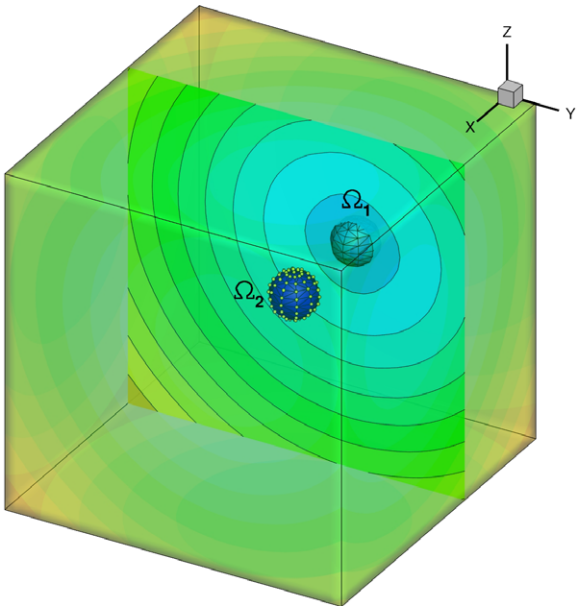

Fig. 1. Sketch of the collision detection algorithm. Solid Ω_1 is equipped with a distance field, represented in the figure, while solid Ω_2 is represented, for collision detection purposes, as a pointshell, i.e., a collection of boundary nodes and normals to the surface (not represented for simplicity).

not valid for haptic feedback requirements, if we deal with two deformable solids, see [5]. This limitation is due to the need of updating the distance field along with the deformation of Ω_1, which is not an easy task, even if the collision detection loop is not performed at every haptic cycle, as in [59]. In the approach here presented, there is no need for such an update, since a high-dimensional distance field is computed off-line that contains the distance fields for any deformed configuration of the solid.

Once collision has been detected, a force

$$\boldsymbol{F} = -k_c d\boldsymbol{n}$$

is applied to both solids, provided that d is trilinearly interpolated from the distance field accompanying Ω_1. k_C is the contact penalty stiffness. The minus sign assumes that normals to Ω_2 point towards the interior of the solid, although this is completely arbitrary. Finally, $\boldsymbol{n}$ represents the normal to Ω_2 in the deformed configuration. An equal force is applied to Ω_1, regardless of its geometry, since we have no information on its normals to the surface. This very simple algorithm preserves continuity in the force computation, essential to perceive haptic response as realistic.

For very slender solids, large penetrations could eventually lead to decreasing force values, provided that Ω_2 crosses the medial axis of Ω_1, see [4]. However, note that collision will be checked every millisecond. Except from very large solid velocities, this will likely never happen in fields such as computational surgery, for instance.

Remark 1. It is important to note that computational vademecums are designed for one single punctual load, or at least for a fixed number of loads. In contact problems the position of contact, and therefore the number of contacting nodes is a priori unknown. In the results presented in Section 4, where non-linear constitutive equations and strain measures are inherent, a predictor of the result is obtained by simply applying superposition, i.e., a linear combination of the contacting load predicted by the vademecums associated to each contacting node.

This very simple approach results to be enough for many applications such as computational surgery [38], since visual realism is kept (no artificial gain of volume, as in purely elastic approaches, is obtained) and the human sense of touch is not able to detect the lack of accuracy generated in the resulting load transmitted by the haptic peripheral.

If more accuracy is needed, a correction can be added to this prediction by employing *hyper reduction* methods [60–63]. In essence, the solution is projected onto the subspace spanned by the PGD modes of the solution (as in POD-based MOR) and the tangent stiffness matrix is evaluated only at a very limited number of elements of the mesh, typically some three times the number of modes considered. In our experiments, this approach has rendered

excellent results, albeit not always necessary. This same rationale could be applied to the updating of the pre-computed distance field, although our experience indicates that the force feedback produced by the predictor obtained by linear superposition provides sufficient accuracy. I general, the correction is not perceivable by the human sense of touch.

The use of the correction step could seem compromising real time responses. However, it is important to notice that (i) only some elementary stiffness matrices are assembled (the global matrix is not singular because the global stiffness matrix appears pre- and post-multiplied by the matrix containing the reduced basis) [27]; (ii) because the iteration starts from the prediction, that is in general close to the final solution, few correction iterations are needed; and (iii) only vector products are involved and all these can be computed simultaneously and in parallel by using GPUs, for example.

Even if the tests performed proved that surgical simulation applications do not need for the correction step, its consideration does not seem a great challenge, and it will be integrated in our future works to extend the applicability of this strategy to domains other than computational surgery.

3.1. Computation of the distance field

The main ingredient of this algorithm is the distance field associated to the solid Ω_1. Any method could be in principle valid for the computation of this field. The key novelty of the approach here presented is that a high-dimensional distance field

$$d = d(\boldsymbol{x}, \boldsymbol{s})$$

for every load location $\boldsymbol{s}$ is computed off-line and stored in memory, very much in the spirit of computational vademecums. Here, $\boldsymbol{x}$ represents, by an abuse of notation, the coordinates of points belonging to the prism in which the distance field is computed, see Fig. 1. This prism, in general, does not need to be much bigger than the solid Ω_1, see [5], to avoid storing a big number of nodal distance values. However, in the implementation here presented, the distance field must be such that *every* possible deformation state of Ω_1 must lie within the prism. Once computed off-line and once for life, this distance field is evaluated for a particular load position $\boldsymbol{s}$, that is, for any possible deformed configuration of the solid Ω_1, thus giving a standard 3-d field for each load position, without the need of further updating.

Our implementation interpolates trilinearly the three physical coordinates and by nearest neighbors the $\boldsymbol{s}$ coordinate. This leads actually to a 4-d distance field $d = d(x, y, z, s)$, where s represents the node number to which the load is associated. Test codes used Matlab's `griddedInterpolant` algorithm [64] for a very fast evaluation of the distance field at a query point. Results are given in Section 4 on the performance of this technique. The distance field at nodal position of the grid could be computed by employing fast marching algorithms [65] or by simply employing the `pdist` Matlab function.

Remark 2. The *brute force* computation of the distance field is a feasible option if the number of dimensions of the problem (i.e., the number of parameters of the solution) is kept relatively small. If this is not so, the number of distance values to compute grows exponentially with the number of dimensions. The authors are currently exploring the possibility of solving the fast marching equations [65] in parallel with the computation of the vademecum, Eqs. (2) and (3) in a PGD, separated way. In this manner, a vademecum could be obtained containing all the possible deformed configurations, each one attached to its distance field, both expressed in a separated form.

4. Numerical examples of performance

In what follows some examples have been selected to demonstrate the performance of the proposed method. Firstly, a Matlab [64] implementation of the algorithms was accomplished in order to check their validity. Once their performance had been assessed, a final implementation for the Geomagic Touch haptic device [66] was developed in order to test realism of the reaction forces, stability of the perceived touch, etc.

4.1. Contact between a beam and a rigid solid

To begin with, the simplest case of collision detection between a hyperelastic (Saint Venant–Kirchhoff) cantilever beam and a solid is considered. A squared cross-section beam, with $40 \times 40 \times 400$ mm, discretized with some 189 nodes, was considered. Young's modulus was assumed to be 209 000 MPa, while Poisson coefficient was set to 0.3.

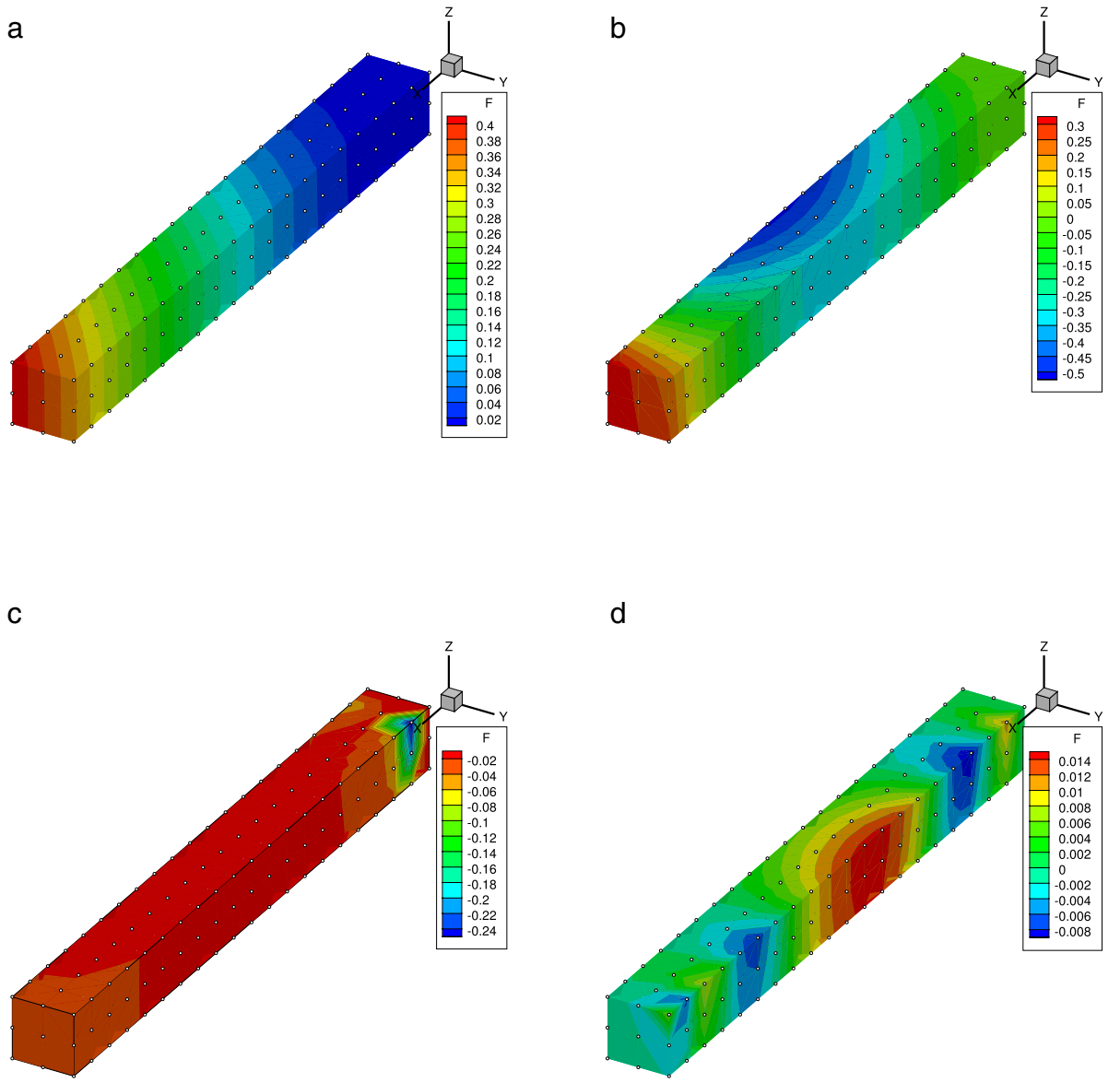

Fig. 2. First two modes of the beam model, depending on the physical coordinates $\boldsymbol{x}$ (top) and load position $\boldsymbol{s}$ (bottom).

The beam is actuated by a vertical load of 10^6 N, that can be applied at any point of the boundary of the beam. If nearest-neighbor interpolation for the position of the load was assumed, a total of 170 different positions of this load could be encountered during the on-line phase of the simulation.

During the off-line phase of the calculation, a multidimensional solution is obtained for the displacement field of the beam, whose two first pairs of modes are shown in Fig. 2. Details of the PGD procedure for the solution of this parametric problem can be found at [57] or [53]. A total of 81 pairs of functions $\boldsymbol{X}_i$ and $\boldsymbol{Y}_i$ (see Eq. (2)) were employed in this example, i.e., $n = 81$. Tests with only some 25 pairs also gave satisfactory results, with no perceptible loss of accuracy for the human eye. The off-line computation of this example takes roughly less than half an hour on a standard laptop.

Over this general solution $\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{s})$, a four-dimensional distance field $d(\boldsymbol{x}, \boldsymbol{s})$ is constructed. In this case, although the obvious choice would have been to equip the rigid solid with the distance field, to demonstrate the capabilities of the method the inverse solution has been preferred. Therefore, the distance field was attached to the deformable solid, the beam. The mesh to construct this distance field contained $42 \times 26 \times 61 \times 170$ nodes (i.e., more than eleven millions

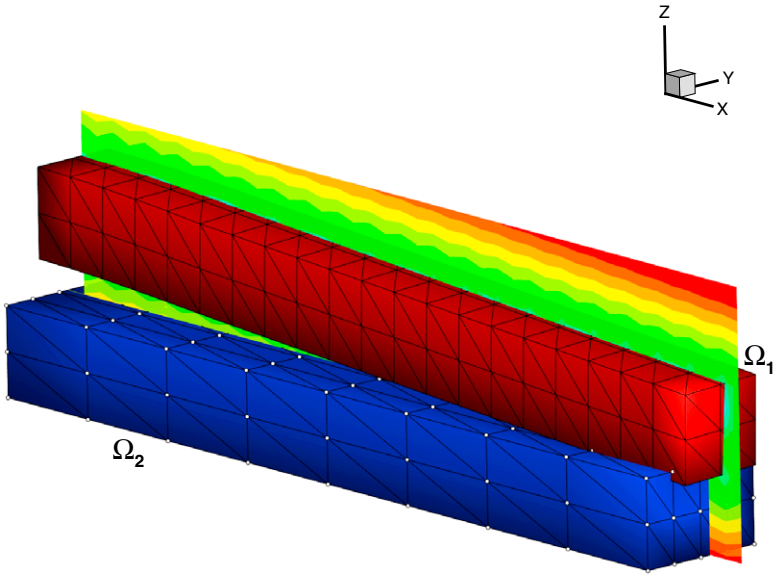

Fig. 3. Collision between a beam and a rigid block for one of the 170 possible load locations in the model. The beam (red), the block (blue) and a slice of the distance field are depicted. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of degrees of freedom) along x, y, z and s (load position, considered as a one-dimensional array of nodal locations) directions, respectively.

A rigid block is also considered with which the beam gets into contact, see Fig. 3. All the possible load cases were tested and the time necessary to detect contact stored. On a Macbook Pro laptop equipped with an Intel Core i7 processor running at 3 GHz (8Gb DDR3 RAM at 1600 MHz) and running Matlab [64] the results indicated that the time to contact detection was $0.0006223 \pm 1.02122E-07$ seconds, well within the strong requirements of haptic contact feedback.

4.2. Contact between two hyperelastic beams

The fact that in the previous example a rigid solid was considered has no special implications on the complexity of the algorithm. If a deformable–deformable contact between two beams is now considered, the numbers remain roughly the same, the complexity of the algorithm depending essentially on the number of nodes of the pointshell and the number of degrees of freedom considered for the distance field. Both vademecums are essentially the same, so that the computer cost of the off-line computations remains the same of the previous example.

The algorithm here presented is so powerful that it can be implemented on an html page running javascript, for instance, and still provide visual perception of interactivity when the beam is touched with the mouse on the screen, see Fig. 4. Contact iterations were limited to run in less than 100 ms, but the typical time to find collision was around 40 ms, still under the typical 25 frames per second of films.

The implementation of the same problem on a HP ProBook 6470b laptop (Intel Core i7, with 8 Gb DDR3 PC3-12800 SDRAM running Ubuntu) equipped with a Geomagic Touch [66] haptic device also gave excellent results, with no noticeable jumps in the perceived contact force, see Fig. 5.

4.3. Contact between two Stanford bunnies

In this example two Stanford bunnies were forced to contact each other in order to test the capabilities of the proposed technique. From the original STL file, a tetrahedral finite element model composed by 4285 nodes for each bunny was constructed, see Fig. 6. The total number of nodes in the pointshell (blue bunny in Fig. 6) was 3733. Bunnies were assumed to be modeled by a Kirchhoff–Saint Venant hyperelastic law with $E = 2.0 \cdot 10^7$ MPa and $\nu = 0.3$.

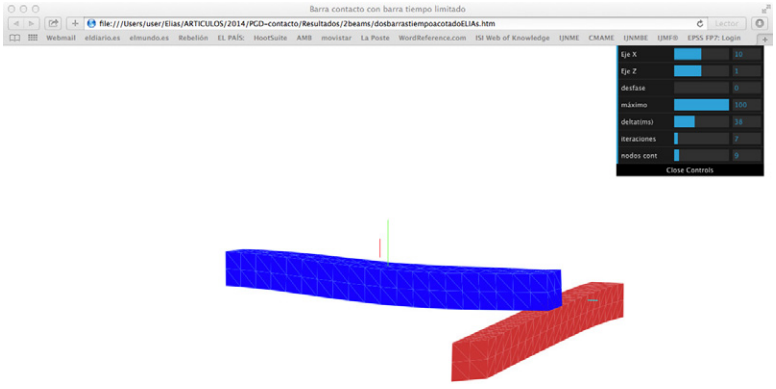

Fig. 4. Collision between two beams running on an html web page with embedded javascript. The red stroke indicates the position of the load actuating the blue beam. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

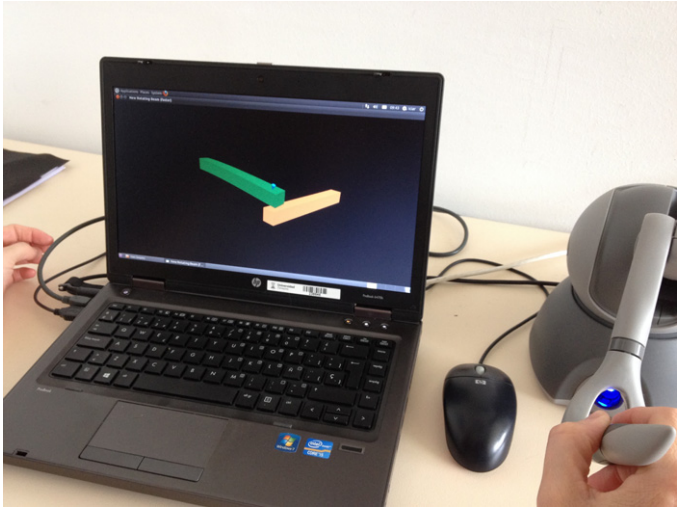

Fig. 5. The two-beam problem actuated by a Geomagic Touch haptic device.

The vademecum for each bunny – they are identical for both bunnies – contained 143 modes. The cost of the off-line computation of these vademecums was more than three hours on a Mac Pro computer running Matlab [64] and equipped with a 6-core Intel Xeon ES CPU (although no parallelization was accomplished) at 3.5 GHz and with 16 GB RAM, 1867 MHz DDR3.

The red bunny is actuated by the user, provoking a rigid-solid displacement until contact occurs with the blue one. Three different positions were tested. At position number 1, no collision was detected, and the total time employed by the proposed algorithm was 0.0007148 s. At position number 2, 36 contacting nodes were detected, while the total time employed for their location was 0.001611 seconds. Finally, at position number three, 427 nodes got into contact, and in this last case 0.0012811 seconds were necessary to detect them.

The deformed configuration of the bunnies after contact is depicted in Fig. 7.

Again, for complex, detailed models of intricate geometry, the proposed method is able to render feedback at rates on the order of 1 kHz, proving the validity of the assumptions made so far.

5. Conclusions

In this paper a method for the real-time detection of collision between deformable solids has been presented. Although inspired by existing algorithms, the novelty of the proposed technique lies in the use of *computational*

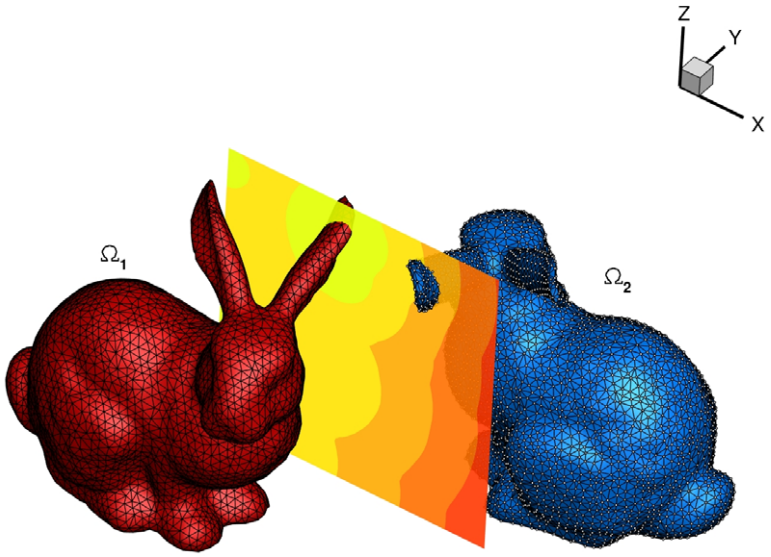

Fig. 6. Collision between two Stanford bunnies. A slice of the distance field is also shown.

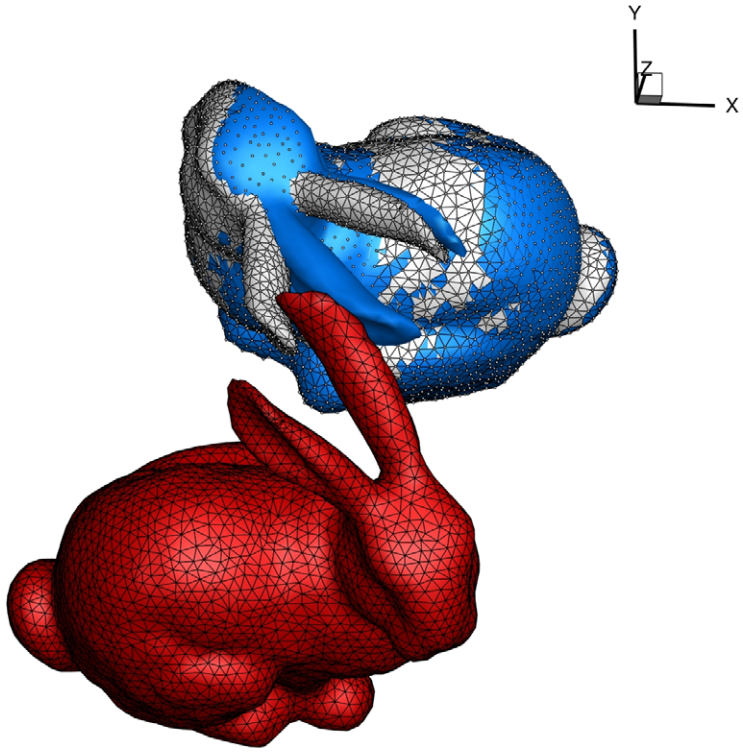

Fig. 7. Deformed configuration of the bunnies after contact. Only the undeformed configuration of the blue bunny is represented, for simplicity, in white. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

vademecums, i.e., pre-computed solutions that are exploited on-line at very high feedback rates. Here, the proposed method has been successfully tested on a Geomagic Touch device needing for 1 kHz feedback rates.

These extremely efficient vademecums are previously computed with the help of Proper Generalized Decomposition strategies, in order to alleviate the burden associated with the curse of dimensionality, i.e., the exponential increase of the number of degrees of freedom with the number of dimensions (parameters) of the problem.

Solutions for the displacement field of different solids under arbitrary load locations are thus obtained and employed to compute four-dimensional distance fields to the boundary of the object. Collision with the second solid is detected by checking the position of its boundary nodes against this distance field. Boundary nodes lying at negative distance regions are then penalized to force them to avoid interpenetration.

The main advantage of the proposed method over existing algorithms is that it avoids the need of updating the distance field with the deformation of the reference solid. Since *any* deformation of the solid has been parametrized previously by PGD and stored efficiently as a set of one-dimensional vectors, with the distance function associated to each deformed configuration, no need for on-line updating of the distance field results. This allows for a more detailed mesh for the computation of the distance field or, equivalently, for a more detailed geometric approximation of the pointshell in the second solid.

So far only collision detection has been accomplished within this algorithm, although we strongly believe that a generalization of the proposed technique is possible for frictional contact. This constitutes nowadays our main effort of research, whose results will be presented elsewhere.

References

[1] P. Wriggers, Computational Contact Mechanics, Wiley, 2002.

[2] Jernej Barbič, Doug James, Time-critical distributed contact for 6-DoF haptic rendering of adaptively sampled reduced deformable models. In D. Metaxas and J. Popovic, Symposium on Computer animatioN 2007: ACM SIGGRAPH/eurographics Symposium Proceedings, pages 171–180, 1515 broadway, New York, NY 10036-9998 USA, 2007. ACM SIGGRAPH; Eurog Assoc, assoc computing machinery. Symposium on Computer Animation, San Diego, CA, AUG 03-04, 2007.

[3] D.L. James, Bd-tree: Output-sensitive collision detection for reduced deformable models, ACM Trans. Graph. 23 (2004) 393–398.

[4] Jernej Barbič, Doug L. James, Six-dof haptic rendering of contact between geometrically complex reduced deformable models, IEEE Trans. Haptics 1 (1) (2008) 39–52.

[5] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.-P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, P. Volino, Collision detection for deformable objects, Comput. Graph. Forum 24 (1) (2005) 61–81.

[6] Z.A. Taylor, M. Cheng, S. Ourselin, High-speed nonlinear finite element analysis for surgical simulation using graphics processing units, IEEE Trans. Med. Imaging 27 (5) (2008) 650–663.

[7] Z.A. Taylor, O. Comas, M. Cheng, J. Passenger, D.J. Hawkes, D. Atkinson, S. Ourselin, On modelling of anisotropic viscoelasticity for soft tissue simulation: Numerical solution and GPU execution, Med. Image Anal. 13 (2) (2009) 234–244. Includes special ection on functional imaging and modelling of the heart.

[8] Herve Delingette, Nicholas Ayache, Hepatic surgery simulation, Commun. ACM 48 (February) (2005) 31–36.

[9] Yi-Je Lim, Suvranu De, Real time simulation of nonlinear tissue response in virtual surgery using the point collocation-based method of finite spheres, Comput. Methods Appl. Mech. Engrg. 196 (2007) 3011–3024.

[10] U. Meier, O. Lopez, C. Monserrat, M.C. Juan, M. Alcaniz, Real-time deformable models for surgery simulation: a survey, Comput. Methods Programs Biomed. 77 (3) (2005) 183–197.

[11] Stéphane Cotin, Hervé Delingette, Nicholas Ayache, in: Hans Hagen (Ed.), Real-Time Elastic Deformations of Soft Tissues for Surgery Simulation, in: IEEE Transactions on Visualization and Computer Graphics, vol. 5, IEEE Computer Society, 1999, pp. 62–73.

[12] Iciar Alfaro, David Gonzalez, Felipe Bordeu, Adrien Leygue, Amine Ammar, Elias Cueto, Francisco Chinesta, Real-time in silico experiments on gene regulatory networks and surgery simulation on handheld devices, J. Computat. Surgery 1 (1) (2014) 2194–3990.

[13] S. Niroomandi, I. Alfaro, D. Gonzalez, E. Cueto, F. Chinesta, Real-time simulation of surgery by reduced-order modeling and x-fem techniques, Internat. J. Numer. Methods in Biomed. Eng. 28 (5) (2012) 574–588.

[14] Aiert Amundarain, Diego Borro, Alex García-Alonso, Jorge Juan Gila, Luis Mateya, Joan Savall, Virtual reality for aircraft engines maintainability, Mécanique et Industries 5 (2) (2004) 121–127.

[15] Ming Wan, William A. McNeely, Quasi-static approximation for 6 degrees-of-freedom haptic rendering, in: Proceedings of the 14th IEEE Visualization 2003 (VIS'03), VIS '03, IEEE Computer Society, Washington, DC, USA, 2003, p. 34.

[16] Mikel Sagardia, Thomas Hulin, Fast and accurate distance, penetration, and collision queries using point-sphere trees and distance fields, in: ACM SIGGRAPH 2013 Posters, SIGGRAPH '13, ACM, New York, NY, USA, 2013, p. 83:1.

[17] H. Delingette, N. Ayache, Soft tissue modeling for surgery simulation, in: N. Ayache (Ed.), Computational Models for the Human Body, in: Ph. Ciarlet (Ed.), Handbook of Numerical Analysis, Elsevier, 2004, pp. 453–550.

[18] G.A. Holzapfel, T.C. Gasser, A new constitutive framework for arterial wall mechanics and a comparative study of material models, J. Elasticity 61 (2000) 1–48.

[19] V. Alastrue, B. Calvo, E. Pena, M. Doblare, Biomechanical modeling of refractive corneal surgery, Tras. ASME J. Biomech. Eng. 128 (2006) 150–160.

[20] S. Uno, M. Slater, The sensitivity of presence to collision response, in: Proceedings of the 1997 Virtual Reality Annual International Symposium (VRAIS '97), VRAIS '97, IEEE Computer Society, Washington, DC, USA, 1997, p. 95.

[21] R. Bridson, S. Marino, R. Fedkiw, Simulation of clothing with folds and wrinkles. in: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '03, pages 28–36, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[22] Jung Kim, Changmok Choi, Suvranu De, Mandayam A. Srinivasan, Virtual surgery simulation for medical training using multi-resolution organ models, Internat. J. Med. Robotics Computer Assisted Surgery 3 (2) (2007) 149–158.

[23] Doug L. James, Dinesh K. Pai, Multiresolution green's function methods for interactive simulation of large-scale elastostatic objects, ACM Trans. Graph. 22 (1) (2003) 47–82.
[24] Karol Miller, Grand Joldes, Dane Lance, Adam Wittek, Total lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation, Commun. Numer. Methods Eng. 23 (2) (2007) 121–134.
[25] Grand Roman Joldes, Adam Wittek, Karol Miller, Real-time nonlinear finite element computations on {GPU} – application to neurosurgical simulation, Comput. Methods Appl. Mech. Engrg. 199 (49–52) (2010) 3305–3314.
[26] Hadrien Courtecuisse, Jérémie Allard, Pierre Kerfriden, Stéphane P.A. Bordas, Stéphane Cotin, Christian Duriez, Real-time simulation of contact and cutting of heterogeneous soft-tissues, Med. Image Anal. 18 (2) (2014) 394–410.
[27] S. Niroomandi, I. Alfaro, E. Cueto, F. Chinesta, Real-time deformable models of non-linear tissues by model reduction techniques, Comput. Methods Programs Biomed. 91 (3) (2008) 223–231.
[28] Siamak Niroomandi, Iciar Alfaro, Elias Cueto, Francisco Chinesta, Model order reduction for hyperelastic materials, Int. J. Numer. Methods Eng. 81 (9) (2010) 1180–1206.
[29] S. Niroomandi, I. Alfaro, E. Cueto, F. Chinesta, Accounting for large deformations in real-time simulations of soft tissues based on reduced-order models, Computer Methods and Programs in Biomedicine 105 (1) (2012) 1–12.
[30] Z.A. Taylor, S. Ourselin, S. Crozier, A reduced order finite element algorithm for surgical simulation. in: Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE, pages 239–242, 31 2010–sept. 4 2010.
[31] Z.A. Taylor, S. Crozier, S. Ourselin, A reduced order explicit dynamic finite element algorithm for surgical simulation, IEEE Trans. Med. Imaging 30 (9) (2011) 1713–1721.
[32] Jernej Barbič, Doug L. James, Real-time subspace integration for St. Venant-Kirchhoff deformable models, ACM Trans. Graphics (SIGGRAPH 2005) 24 (3) (2005) 982–990.
[33] K. Karhunen, Uber lineare methoden in der wahrscheinlichkeitsrechnung, Ann. Acad. Sci. Fennicae, Ser. Al. Math. Phys. 37 (1946).
[34] M.M. Loève, Probability theory, in: The University Series in Higher Mathematics, 3rd ed., Van Nostrand, Princeton, NJ, 1963.
[35] E.N. Lorenz, Empirical Orthogonal Functions and Statistical Weather Prediction, Scientific Report Number 1, Statistical Forecasting Project, MIT, Departement of Meteorology, 1956.
[36] D. Amsallem, Ch. Farhat, An interpolation method for adapting reduced-order models and application to aeroelasticity, AIAA J. 46 (2008) 1803–1813.
[37] Saifon Chaturantabut, Danny C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (September) (2010) 2737–2764.
[38] Elias Cueto, Francisco Chinesta, Real time simulation for computational surgery: a review, Adv. Model. Simul. Eng. Sci. 1 (1) (2014) 11.
[39] P. Ladeveze, Nonlinear Computational Structural Mechanics, Springer, NY, 1999.
[40] A. Ammar, B. Mokdad, F. Chinesta, R. Keunings, A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids, J. Non-Newtonian Fluid Mech. 139 (2006) 153–176.
[41] A. Ammar, B. Mokdad, F. Chinesta, R. Keunings, A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. part ii: transient simulation using space–time separated representations, J. Non-Newtonian Fluid Mech. 144 (2007) 98–121.
[42] Francisco Chinesta, Pierre Ladeveze, Elias Cueto, A short review on model order reduction based on proper generalized decomposition, Arch. Comput. Methods Eng. 18 (2011) 395–404.
[43] F. Chinesta, A. Ammar, E. Cueto, Recent advances in the use of the Proper Generalized Decomposition for solving multidimensional models, Archives of Computational Methods in Engineering 17 (4) (2010) 327–350.
[44] D. Gonzalez, A. Ammar, F. Chinesta, E. Cueto, Recent advances on the use of separated representations, International Journal for Numerical Methods in Engineering 81 (5) (2010).
[45] F. Chinesta, A. Leygue, F. Bordeu, J.V. Aguado, E. Cueto, D. Gonzalez, I. Alfaro, A. Ammar, A. Huerta, PGD-based computational vademecum for efficient design, optimization and control, Archives of Computational Methods in Engineering 20 (1) (2013) 31–59.
[46] P. Ladeveze, J.-C. Passieux, D. Neron, The latin multiscale computational method and the proper generalized decomposition, Comput. Methods Appl. Mech. Engrg. 199 (21–22) (2010) 1287–1296.
[47] Ch. Heyberger, P.-A. Boucard, D. Neron, A rational strategy for the resolution of parametrized problems in the {PGD} framework, Comput. Methods Appl. Mech. Engrg. 259 (1) (2013) 40–49.
[48] Anthony Nouy, A priori model reduction through Proper Generalized Decomposition for solving time-dependent partial differential equations, Computer Methods in Applied Mechanics and Engineering 199 (23–24) (2010) 1603–1626.
[49] Ch. Ghnatios, F. Chinesta, E. Cueto, A. Leygue, A. Poitou, P. Breitkopf, P. Villon, Methodological approach to efficient modeling and optimization of thermal processes taking place in a die: Application to pultrusion, Composites A 42 (9) (2011) 1169–1178.
[50] Ch. Ghnatios, F. Masson, A. Huerta, A. Leygue, E. Cueto, F. Chinesta, Proper generalized decomposition based dynamic data-driven control of thermal processes, Comput. Methods Appl. Mech. Engrg. 213–216 (2012) 29–41.
[51] D. Gonzalez, F. Masson, F. Poulhaon, E. Cueto, F. Chinesta, Proper generalized decomposition based dynamic data driven inverse identification, Math. Comput. Simul. 82 (2012) 1677–1695.
[52] Amine Ammar, Antonio Huerta, Francisco Chinesta, Elías Cueto, Adrien Leygue, Parametric solutions involving geometry: A step towards efficient shape optimization, Comput. Methods Appl. Mech. Engrg. 268 (2014) 178–193.
[53] S. Niroomandi, D. González, I. Alfaro, F. Bordeu, A. Leygue, E. Cueto, F. Chinesta, Real-time simulation of biological soft tissues: a PGD approach, International Journal for Numerical Methods in Biomedical Engineering 29 (5) (2013) 586–600.
[54] C. Bernoulli, Vademecum des Mechanikers, Cotta, 1836.
[55] R.B. Laughlin, David Pines, The theory of everything, Proc. Natl. Acad. Sci. 97 (1) (2000) 28–31.
[56] David González, Elias Cueto, Francisco Chinesta, Real-time direct integration of reduced solid dynamics equations, Int. J. Numer. Methods Eng. 99 (9) (2014) 633–653.

[57] S. Niroomandi, D. Gonzalez, I. Alfaro, E. Cueto, F. Chinesta, Model order reduction in hyperelasticity: a proper generalized decomposition approach, International Journal for Numerical Methods in Engineering 96 (3) (2013) 129–149.
[58] M. Barrault, Y. Maday, N.C. Nguyen, A.T. Patera, An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations, Comptes Rendus Math. 339 (9) (2004) 667–672.
[59] Lenka Jeřábková, Torsten Kuhlen, Stable cutting of deformable objects in virtual environments using xfem, IEEE Comput. Graph. Appl. 29 (2) (2009) 61–71.
[60] D. Ryckelynck, A priori hyperreduction method: an adaptive approach, J. Comput. Phys. 202 (1) (2005) 346–366.
[61] D. Ryckelynck, Hyper-reduction of mechanical models involving internal variables, International Journal for Numerical Methods in Engineering 77 (1) (2008) 75–89.
[62] D. Ryckelynck, L. Hermanns, F. Chinesta, E. Alarcon, An efficient 'a priori' model reduction for boundary element models, Eng. Anal. Bound. Elem. 29 (8) (2005) 796–801.
[63] E. Lopez, E. Abisset, Ch. Ghnatios, Ch. Binetruy, F. Chinesta, Towards real time thermal homogenization of heterogeneous microstructures. in: 3rd International Conference on Computational Methods for Thermal Problems, 2014.
[64] MATLAB, version 8.1.0.604 (R2013a), The MathWorks Inc., Natick, Massachusetts, 2013.
[65] J.A. Sethian, Fast marching methods, SIAM Rev. 41 (1998) 199–235.
[66] Geomagic, OpenHaptics Toolkit. 3D systems - Geomagic solutions, 430 Davis Drive, Suite 300 Morrisville, NC 27560 USA, 2013.