Exact and efficient interpolation using finite elements shape functions

Gustavo H.C. Silva — Rodolphe Le Riche — Jérôme Molimard Alain Vautrin

École Nationale Supérieure des Mines de Saint-Étienne
158, cours Fauriel
F-42023 Saint-Étienne cedex 2

ABSTRACT. *The comparison of finite elements (FE) and experimental data fields have become ever more prevalent in numerical simulations. Since FE and experimental data fields rarely match, the interpolation of one field into the other is a fundamental step of the procedure. When one of the fields comes from FE, using the existing FE mesh and shape functions is a natural choice to determine mesh degrees of freedom at data point coordinates. This makes no assumptions beyond those already made in the FE model. In this sense, interpolation using element shape functions is exact. However, crude implementations of this technique generally display a quadratic computation complexity with respect to mesh size and number of data points, which is impractical when large data fields must be compared repeatedly. This document aims at assembling existing numerical procedures to improve the interpolation efficiency. With a combination of cross-products, bounding-boxes and indexing methods, the resulting algorithm shows linear computation cost, providing significant improvement in efficiency.*

RÉSUMÉ. *La comparaison entre des champs simulés par éléments finis (EF) et des données obtenues par mesures de champs devient une étape de plus en plus courante des simulations numériques. Quand un des champs vient d'un modèle EF, l'utilisation du maillage et des fonctions de forme EF pour déterminer des degrés de liberté aux coordonnées expérimentales est une approche naturelle. Elle n'introduit aucune autres hypothèses que celles de la simulation. En ce sens, nous qualifions cette interpolation d'exacte. Cependant, une implémentation directe de l'interpolation conduit à une complexité algorithmique qui varie de manière quadratique avec le nombre d'éléments et de points de mesure, ce qui n'est pas acceptable lorsque de grands champs de données doivent être comparés plusieurs fois. Cet article décrit comment, à travers une présélection des éléments par un maillage virtuel et l'utilisation de boîtes englobantes, la complexité algorithmique de l'interpolation EF peut rester linéaire.*

KEYWORDS: *identification, interpolation, finite elements, field measurements.*

MOTS-CLÉS : *identification, interpolation, éléments finis, mesures de champs.*

DOI:10.3166/EJCM.18.307-331

1. Introduction

The increasing use of finite elements has given rise to a series of applications requiring the comparison of data fields with finite elements results. These applications include mesh-interaction (Beckert, 2000; Van Loon *et al.*, 2006), geological topography (Fukushima *et al.*, 2005), visualization (Nikishkov, 2003; Rumpf, 1999), contact analysis (Faucher *et al.*, 2003), calibration of boundary conditions (Padmanabhan *et al.*, 2006) and material identification (Bledzki *et al.*, 1999; Bruno *et al.*, 2002; Cugnoni *et al.*, 2004; Genovese *et al.*, 2006; Kajberg *et al.*, 2004; Meuwissen *et al.*, 1998; Molimard *et al.*, 2005; Cardenas-Garcia *et al.*, 2006). Reviews of material identification techniques can be found in (Grediac, 2004; Amiot *et al.*, 2007; Avril *et al.*, 2007). In general, a finite elements solver provides information only at nodal or integration points, which may not always coincide available experimental data points. Hence, mapping one field of points onto the other becomes necessary.

For identification problems, the choice of mapping technique is fundamental, often the first of a long series of assumptions and decisions. Popular mapping methods include: generic approximation [1] and interpolation algorithms (Cressie, 1993; Mathworks, 2004; Kajberg *et al.*, 2004), node placement (Bruno *et al.*, 2002; Meuwissen *et al.*, 1998), mesh projection (Pagnacco *et al.*, 2007) and shape function interpolation. Amongst these, interpolation with finite element shape functions is selected as the most natural approach. It simplifies the identification decision chain, since it makes no assumptions beyond those already made in the FE model and requires no adjustment of interpolation intrinsic parameters (*i.e.* kernel width, polynomial degrees, etc). Interpolating nodal degrees of freedom at experimental data points leaves experimental data completely unaltered. Hence, no implicit filtering is introduced by interpolating experimental data. Since the FE mesh is also unaltered, the approach completely decouples the accuracy of the experiment and numerical model. However, in identification problems, each time a parameter changes, the FE model has to be compared to a constant data field, requiring the repeated mapping of 10^3 to 10^4 data points. Hence, it is very important that this mapping be carried out as efficiently as possible.

The mapping of each data point entails two steps: first, the element containing each data point is identified (this is the most computationally intensive step). Next, the coordinates of each data point are transformed into the finite element's reference coordinate system (hereafter this procedure is referred to as inverse mapping). The reference coordinates are used to compute the element's shape-function coefficients which completes the interpolation. Given the complexity of these operations, it is easy for implementations of the algorithm to be very time consuming. Hence, a variety of methods are discussed to improve the interpolation speed. Most of these methods can be found in a variety of contexts such as image processing (Bourke, 1989), mesh generation (Rassineux, 1994) and contact analysis (Faucher *et al.*, 2003). To the authors

1. Unlike interpolation, approximation points do not necessarily match experimental or FE data points.

knowledge, these methods have not been presented together in the context of material identification, which is the main objective of this article. The paper begins by describing some available mapping algorithms, justifying the choice of shape function interpolation. It continues by discussing possible combinations of available interpolation methods. A theoretical estimate of the computation cost, based on operations counting, is made for each combination. These are later bench-marked in a C++ test program, which is applied to plate with a hole problem.

Summary of notation

$e(p)$	Element containing p
$\epsilon(p)$	Neighborhood around point p
n_i^j	i^{th} node of set j
$\mathbf{n}^{mesh}$	Set of nodes in a finite elements mesh
$\mathbf{n}^{\epsilon(p)}$	Set of nodes in a neighborhood of p
p	A data point
$\mathbf{p}$	Set of data points
$\vec{v}$	A vector
v_i	The i^{th} scalar component of vector $\vec{v}$
$\vec{x}(p)$	Coordinates of point p in the global coordinate system
$\vec{\xi}(p)$	Coordinates of point p in an element's reference coordinate system

2. Overview of mapping techniques for comparing data fields

This section is an overview of approaches for comparing field measurements and FE model results. Experimental fields are corrupted by deterministic and random errors, but this article, which is primarily concerned with the numerical cost of field comparison, does not discuss these errors, although it is acknowledged they are of great importance for identification accuracy. Hence, throughout this article, experimental fields are assumed to be in the form of discrete data points (see Figure 1(a)). When interpolating nodal degrees of freedom, the interpolation accuracy is measured with respects to FE shape functions (see Figure 1(b)). An interpolated value $u_n(p)$ is

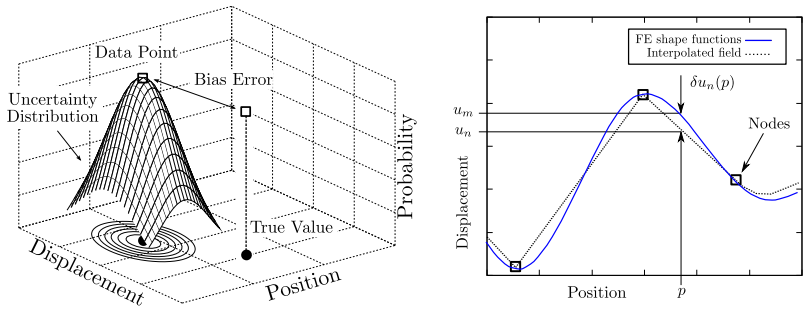
$$u_n(p) = u_m(p) + \delta u_n(p), \qquad [1]$$

where $u_m(p)$ is the mesh displacement and $\delta u_n(p)$ the interpolation error. In general this interpolation error is less important than the measurement uncertainty.

2.1. *Node placement and interpolation of experimental data*

In identification, coordinate mismatch problem is often avoided by forcing FE nodes and experimental points to coincide (Bruno *et al.*, 2002; Meuwissen *et*

al., 1998), a practice hereafter referred to as node placement. Node placement is achieved either by selecting only points with matching node and experimental coordinates, or by moving FE nodes to match experimental data points, or by acquiring experimental data at node coordinates. The three approaches are all valid solutions to the mapping problem but carry with them a series of technical challenges. In the first method, the selection of matching points cannot in general exploit all available experimental data, which makes the comparison sensitive to small positioning errors. In the second method, since the accuracy of FE models depend on node position, mesh distortion may introduce an undesired coupling between the position of data points and the accuracy of model. In the third method, node placement by acquisition at node coordinates requires either an non-evolving FE mesh or a new experimental acquisition each time the FE mesh evolves.



(a) Measurement uncertainty (b) Interpolation uncertainty

Figure 1. *(a) Experimental data always contains spatial and measurement uncertainties, both deterministic and random. This article assumes a point-wise recording of the measure. (b) The interpolation error is defined with respects to the FE shape functions*

Alternatively, the experimental measures may be interpolated at nodal coordinates without conducting new experimental acquisitions. Exploiting knowledge of the measurement system, it may be possible to preserve the field characteristics and change the post-processing of a measured field to obtain a coinciding discrete field. For instance, in digital image correlation, displacements are measured by assuming a deformation law (e.g. linear, quadratic) over a set of neighboring pixels (correlation windows) (Wang *et al.*, 2002). Hence, experimental data may be interpolated at node positions with the same deformation law for each correlation window without introducing an implicit filter. However, this can be very complex particularly when information about the process is not fully available. Otherwise, careless interpolation experimental data introduce fundamental assumptions on the shape of the field that may degrade information through implicit filtering.

2.2. *General interpolation of numerical results*

The *griddata* and *interp2* functions in the Matlab® software package (Mathworks, 2004) provides four techniques to obtain data $u(\mathbf{p})$ at arbitrary points $\mathbf{p}$ from a cloud of data $u(\mathbf{n})$: triangle-based and quadrilateral-based linear, cubic, nearest neighbor and biharmonic spline (B-spline) interpolation. All methods in *griddata* (except B-spline (Sandwell, 1987)) generate interpolation meshes via Delaunay triangulation (Barber *et al.*, 1996). Meshes for *interp2* are user-defined and must be expressed as a matrix of sorted node coordinates. These techniques are summarized by the equation

$$u(p) = \sum_{i=1}^{\text{no. nodes in } \epsilon(p)} W_i \left(\vec{x}(p), \vec{x}(\mathbf{n}^{\epsilon(p)}) \right) u(n_i^{\epsilon(p)}), \qquad [2]$$

which define the degrees of freedom at point p, $u(p)$, as weighted sum of neighboring data points $u(n_i^{\epsilon(p)})$. The nearest-node and linear interpolation algorithms are the fastest, but have discontinuous zeroth and first derivatives, respectively. The other two are continuous up to the second derivatives, but are significantly slower. By default, *griddata* does not account for fields with an internal discontinuity, such as a plate with a hole. The algorithm fills the hole with elements, thus introducing boundary effects on the interpolated data. Kriging (Cressie, 1993) is another interpolation approach popular in geo-statistics. It has the advantage of providing an estimate of the interpolation variance. However, it is computationally expensive (often requiring the solution of large linear systems). Another drawback of general interpolation is that it affects the data by introducing non-physical parameters to the comparison (*i.e.* kernel width, polynomial degrees, variogram length and scales in kriging).

2.3. *Methods based on finite elements shape functions*

With FE shape functions, N_j, the value $u(p)$ is estimated from the node values $u(\mathbf{n}^{e(p)})$ of the element containing the data point p, $e(p)$ (hereafter referred to as the owner element),

$$u(p) = \sum_{j=1}^{\text{no. nodes in } e(p)} N_j \left(\vec{\xi}(p) \right) u(n_j^{e(p)}). \qquad [3]$$

Generally, shape functions are written in the reference system of the element containing point p. Calculating the local coordinates of a point is not a trivial operation, requiring two steps. First, finding the owner element. Second, calculating the local coordinates of the point (an operation referred to as inverse mapping), which normally involves solving a multi-dimensional non-linear system of equations. The comparison of the FE model and data fields is performed either at the coordinates of the FE nodes or at the coordinates of data points. Continuity is guaranteed for the zeroth derivative, but is usually discontinuous for higher-order derivatives.

2.3.1. *Projection of data points onto the FE response space*

E. Pagnacco and D. Lemosse (Pagnacco *et al.*, 2007) describe a technique where the data field is approximated at the nodal coordinates of the finite element mesh. This method searches for node values $u(\mathbf{n}^{mesh})$ that best fit available data $u'(\mathbf{p})$. The approximation is defined as the projection of the data onto the finite element model space. The values of $u(\mathbf{n}^{mesh})$ are determined by solving a least-squares problem,

$$\min_{u(\mathbf{n}^{mesh})} \sum_{i=1}^{\text{no. points in } \mathbf{p}} \left[u'(p_i) - \sum_{j=1}^{\text{no. nodes in } e(p_i)} N_j \left(\vec{\xi}(p_i) \right) u(n_j^{e(p_i)}) \right]^2 . \qquad [4]$$

This procedure requires a number of data points greater than the number of finite elements nodes, and that the data field encompass a representative part of the finite element mesh. The computational cost involves determining the reference coordinates of each data point, and solving the least-squares problem. For applications where the comparison is performed multiple times over a non-changing finite element mesh, the interpolation of the data points has to be performed only once. This projection method may eliminate (filter) high-frequency information in the measurement (e.g. experimental noise).

2.3.2. *FE interpolation at data points*

Instead of interpolating measured data points at FE node coordinates, this article selects the interpolation of node values at the coordinates of the experimental points, leaving the experimental data unaltered. The strategy consists of simply evaluating Equation [3] for each data point, $p \in \mathbf{p}$. A special effort is made to improve the speed of determining the owner element. This approach allows for the computation of degrees of freedom at arbitrary points, and has no restriction in the distribution or the number of data points. The FE solution is completely independent of the position and size of the experimental data field, and since the same mesh is used for solving the model and interpolating the data, the resulting interpolated field is an exact representation of the FE solution. Also, similar to the previous technique, the interpolation coefficients can be computed only once for a non-changing mesh, thus saving time on applications that repeat the interpolation several times. Contrary to the previous technique, this approach does not perform implicit filtering.

3. Determination of the owner element

Interpolation using shape functions is accomplished in three steps: determining the element, which contains the data point (owner element), transforming the data point coordinates into the reference coordinate system (inverse mapping), and finally evaluating the finite elements shape functions to determine the degrees of freedom at that point (Equation [3]). The determination of the owner element is not required to be an independent step in the algorithm. However, this would entail inverse mapping

for every element in the mesh, and checking if the point did fall inside the element (Section 4). Since inverse mapping may be complex and numerically expensive, such an approach would be inefficient. Moreover, inverse mapping algorithms are not guaranteed to have a solution for points outside the element's domain, which jeopardizes the reliability of a two-step interpolation algorithm. Instead, a combination of simple tests is used to predetermine the owner element before inverse mapping.

3.1. *Element tests*

This section details both the cross-product and bounding-box tests. The term "element test" refers to any technique to determine whether a point lies inside or outside of an element.

3.1.1. *Cross-product test*

The cross-product test (Figure 2(a)) consists in a series of cross and dot products, which determines if a data point lies inside the intersection of the "vertex cones" of an element. This is similar to techniques described by (Bourke, 1989). If point p is inside the element, then for every node, $n \in \mathbf{n}^{e(p)}$, the vector $\vec{np}$ will lie inside the cone created by the two adjacent node vectors, $\vec{ni}$ and $\vec{nj}$. This condition is verified using the vectors $\vec{s_1}$ and $\vec{s_2}$,

$$\vec{s_1} = \vec{ni} \times \vec{np} \quad \text{and} \quad \vec{s_2} = \vec{np} \times \vec{nj}. \qquad [5]$$

These vectors will have same direction for internal points and opposite directions for external points. Hence, an internal point must satisfy the condition

$$\vec{s_1} \cdot \vec{s_2} \geq 0 \qquad [6]$$

for all nodes in the element. For external points, there is at least one vertex such that

$$\vec{s_1} \cdot \vec{s_2} < 0. \qquad [7]$$

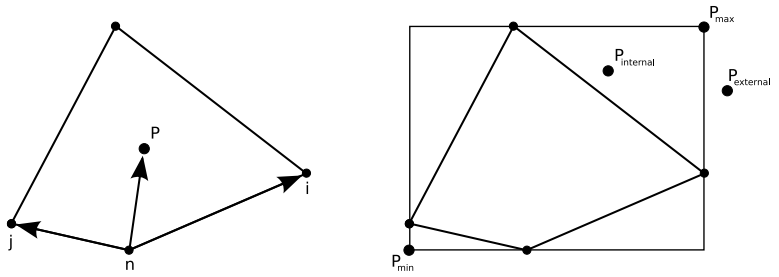
The cross-product test requires that a spatially ordered list of nodes be available for each element (this is standard in finite element mesh formats). This test is exact for elements with linear edges, and approximate otherwise.

3.1.2. *Bounding-box test*

The bounding-box test creates an encompassing box around the finite element (Rassineux, 1994) and compares the coordinates of the data point with the bounding-box's two opposite corner points, p_{min} and p_{max} (see Figure 2(b)). The coordinates of any point inside the bounding-box must satisfy the inequality

$$x_i(p_{min}) \leq x_i(p_{internal}) \leq x_i(p_{max}) \qquad [8]$$

for all dimensions i. At least one of the inequalities will be false for an external point. The complexity of the test includes two parts: computing the boundaries of the bounding-box (later stored in memory), and testing if a data point lies inside the bounding-box. Since the bounding-box only approximates the geometry of a finite element, it is possible for a point to be inside the bounding-box, but outside the finite element ($p_{internal}$ in Figure 2(b)). Thus, in order to identify the owner element, it is necessary for the bounding-box to be used together with an exact test such as the cross-product. Section 3.3, discusses the advantages of scanning a list of finite elements using the bounding-box and cross-product tests together instead of a search algorithm using only cross-products. The algorithm first eliminates impossible owner elements with the computationally inexpensive bounding-box before checking actual owner elements with the more expensive cross-product test.



(a) Cross-Product Test (b) Bounding-Box Test

Figure 2. *(a) The cross-product test uses cross and dot products to check if a point lies inside all vertex cones of an element. (b) The bounding-box test uses a rectangular approximation of the element to quickly eliminate impossible owner elements (case of external points)*

3.2. *Searching the element list*

The default implementation for finding an owner element is to sequentially scan through a list of elements. Figure 3(a) shows two data points in a finite element mesh and their corresponding places in a sequential storage container. If a point lies near the end of the list, then element ownership tests must be executed for a large portion of the list. Guessing a viable initial point in the list requires knowledge of how the mesh was created, which is not always possible. The virtual mesh (Rassineux, 1994) is intended to limit the number of elements scanned. A virtual mesh must have two features: One, a computationally efficient way of determining a sub-region of the mesh containing data point (referred to as a virtual element). This is typically obtained through a regular paving of space. Two, its virtual elements, v, must store a list $\mathbf{e}^v$ of finite elements e that possibly share a segment of area,

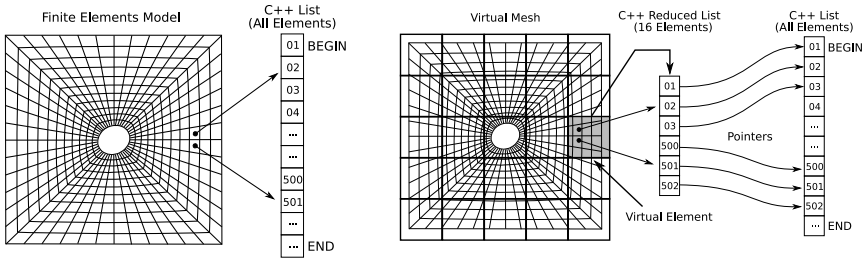
$$\mathbf{e}^v = \left\{ e \in \mathbf{E} \mid e \hat{\cap} v \neq \emptyset \right\}, \qquad [9]$$

where $\hat{\cap}$ is the operator "possibly intersects". This operator is implemented here by computing the virtual element index range for the two opposite diagonal points, $\gamma_i(p_{min})$ and $\gamma_i(p_{max})$ (see Figure 2(b) and Equation [10]). The list $\mathbf{e}^v$ is typically much smaller than the complete list of finite elements $\mathbf{E}$.

Figure 3(b) illustrates the simplest virtual mesh, made of regular rectangles in 2D. This virtual mesh is used for the experiments in this article. Computationally, retrieving the correct virtual element is very inexpensive. The i^{th} dimension index γ_i of the virtual element containing the data point p is resolved using only 3 operations,

$$\gamma_i = floor\left(\frac{x_i(p) - x_i}{dx_i}\right), \qquad [10]$$

where x_i is the origin of the virtual mesh grid and dx_i is the grid-step of the virtual mesh in the i^{th} direction. Once the virtual element has been retrieved, its reduced list of finite elements, $\mathbf{e}^v$, is scanned using element tests.



(a) No Virtual Mesh (b) With Virtual Mesh

Figure 3. *Examples of a finite element mesh and element containers. (a) With a sequential element list it is possible that interpolation points, although close in space, are far away in the container. (b) Virtual elements contain small lists of finite elements in their neighborhoods*

3.3. *Comparison of owner element search algorithms*

This section compares three algorithms used to find the owner element of a data point. Each algorithm is defined by a combination of element tests and an element retrieval technique (either sequential or based on a virtual mesh).

– Search technique 1 (ST1) sequentially scans the full list of finite elements using the cross-product test. The procedure is restarted at the beginning of the list for each data point.

– Search technique 2 (ST2) sequentially searches the full finite elements list using a combination of the cross-product and bounding-box tests. This algorithm is

essentially the same as the previous technique, except that the cross-product test is performed only if the data point is inside the finite element's bounding box. If the element fails the bounding box test, the algorithm moves to the next element in the list. Again, the idea is to eliminate impossible owner elements via the simpler bounding-box test.

– Search technique 3 (ST3) uses the cross-product and bounding-box tests to search small lists of finite elements obtained with the virtual mesh. For each data point, the algorithm retrieves the appropriate virtual element, and scans the list, e^v, using a test similar to ST2.

The efficiency of an element search algorithm is estimated by counting the average number of operations required to find the finite element containing each of N_p points in a uniform field. Operations are defined as basic math operators ($+ \quad - \quad \times \div$), comparisons ($= \neq < > \leqslant \geqslant$), assignments and standard mathematical functions in the C library. A uniform field is assumed to contain a fairly regularly distributed grid of data points throughout the specimen's surface. To the authors knowledge, this assumption is consistent with current full-field measurements techniques. For special applications where a large number of data points are concentrated in a small region (e.g. measuring localized plasticity), the conclusions of this section remain valid locally. A finite element model with a sequential list of N_e elements is illustrated in Figure 3. The search time for a data point belonging to the i^{th} element in this list is

$$\hat{T}_i = (i-1) \times \hat{T}_f + \hat{T}_s, \qquad [11]$$

where $\hat{T}_s$ (success) and $\hat{T}_f$ (failure) are the computation times (measured in number of operations) to determine that the data point does or does not belong to an element, respectively. For an average field, a point can belong to any element in the list with equal probability.

The time complexity $\hat{T}_c$, namely the average number of operations to determine the owner element of one point is

$$\hat{T}_c(N_e) = \frac{1}{N_e} \sum_{i=1}^{N_e} \hat{T}_i. \qquad [12]$$

The total search time $\hat{T}_t$ is an estimate of the average number of operations required to identify the owner element for N_p arbitrary data points including a preprocessing time $\hat{T}_b$ (required to initialize bounding boxes and the virtual mesh),

$$\hat{T}_t(N_e, N_p) = \hat{T}_b(N_e) + \hat{T}_c(N_e) \times N_p. \qquad [13]$$

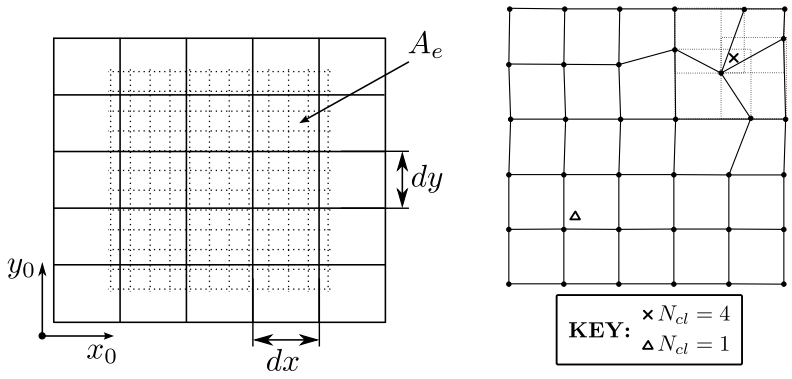
The objective is to find a combination of element tests resulting in the smallest total search time $\hat{T}_t$. An estimate of $\hat{T}_t$ was computed for ST1, ST2 and ST3 applied to meshes of QUAD4 elements and written as $\hat{T}_t|_{st1}$, $\hat{T}_t|_{st2}$ and $\hat{T}_t|_{st3}$, respectively.

These search times are functions of the number of elements N_e, interpolation points N_p, as well as mesh-specific geometric parameters, $\overline{N}_{cl}$, $\overline{N}_k$ and $\overline{N}_{ve}$. The quantity $\overline{N}_{cl}$ is a measure of mesh distortion, specifically the average number of finite element bounding-boxes that will claim an arbitrary data point (see Figure 4(b)). $\overline{N}_{ve}$ is the average number of virtual elements possessing a reference to the same finite element. $\overline{N}_k$ is the average number of finite elements in a virtual element. Results are summarized in the following table:

Table 1. *Computation times for the compared element search algorithms*

Time	Cross-Product	Bounding-Box	Virtual Mesh
$\hat{T}_b$	0	$20N_e$	$(5N_e + 7) + \left[20 + \overline{N}_{ve} + 12\right] N_e$
$\hat{T}_f$	37.5	2.5	Does Not Apply
$\hat{T}_s$	60	4	Does Not Apply

Technique	Computation Time
ST1	$\hat{T}_t\|_{st1} = 18.75N_eN_p + 41.25N_p$
ST2	$\hat{T}_t\|_{st2} = 1.25N_eN_p + 20.75\overline{N}_{cl}N_p + 110.5N_p + 20N_e$
ST3	$\hat{T}_t\|_{st3} = (\overline{N}_{ve} + 37)N_e + \left[1.25\overline{N}_k + 20.75\overline{N}_{cl} + 116.5\right] N_p + 7$



(a) (b) Example of N_{cl}

Figure 4. *(a) Superposition of a finite elements mesh (dotted lines) and a virtual mesh. A_e and $dx \times dy$ are the areas of a bounding-box and a virtual element, respectively. (b) Examples of N_{cl}, the number of bounding boxes which claim a data point*

Details for the operations count and an approximate relationship between $\overline{N}_k$ and $\overline{N}_{ve}$ is developed in (Silva *et al.*, 2007b). This relationship is summarized hereafter. Let the a coefficient α, be the ratio of the average bounding-box area of a finite element $\overline{A}_e$ to the area of a virtual element $dx \times dy$,

$$\alpha = \frac{\overline{A}_e}{dx \times dy}. \qquad [14]$$

The approximation considers the limiting behavior of $\overline{N}_k$ and $\overline{N}_{ve}$ as $\alpha \to 0$ and $\alpha \to \infty$, while accounting for mesh to virtual mesh offsets (see Figure 4(a)),

$$\overline{N}_k \approx \frac{1}{\alpha} + 1, \qquad [15]$$

$$\overline{N}_{ve} \approx \alpha + 1. \qquad [16]$$

If α is very large (*i.e.* very small virtual elements) the majority of the virtual elements will contain a reference to only one finite element, leading to a decrease in the time complexity $\hat{T}_c$ of the search algorithm. However, since there are more virtual elements, the preprocessing time $\hat{T}_b$ will offset the advantage gained by the smaller $\hat{T}_c$. The ratio α^* represents the best compromise between $\hat{T}_c$ and $\hat{T}_b$. It is obtained by expressing $\hat{T}_t|_{st3}$ in Table 1 with in terms of α (using Equations [15] and [16]) and solving for a zero of the derivative,

$$\frac{\partial \hat{T}_t|_{st3}(\alpha^*)}{\partial \alpha} = 0 \quad \Rightarrow \quad \alpha^* = \sqrt{\frac{1.25 N_p}{N_e}}. \qquad [17]$$

The result, α^*, is proportional to $\sqrt{N_p}$ and inversely proportional to $\sqrt{N_e}$. This is intuitive if we consider a situation where there is only one data point to interpolate (*i.e.* $N_p = 1$ thus $N_e >> N_p \Rightarrow \alpha^* << 1$), the optimum grid step produces a single virtual element containing all finite elements in the model. This is because the computational cost of scanning the entire finite element's list only once is smaller than the cost of populating a large number of virtual elements. The converse is also true. For a large number of data points (*i.e.* $N_p >> N_e \Rightarrow \alpha^* >> 1$) the resulting virtual mesh grid is very fine (*i.e.* an average of one finite element per virtual element). In this case the computational cost of populating a fine virtual mesh is compensated by only having to search one finite element per data point.

Figure 5 demonstrates some of the basic features of the different $\hat{T}_t$ curves. All functions are linear with respect to N_p. The slope of ST1 is clearly the largest, followed by ST2 and ST3, which stays nearly flat. In contrast, the offset of ST3 is the largest, followed by ST2, while ST1 has no offset. These offsets are due to the preprocessing time $\hat{T}_b$ required by the virtual mesh and the bounding box calculations. Next, pairs of element tests are compared while varying all parameters. Figure 6 is a comparison of ST1 and ST2. The curves represent iso-lines where $\hat{T}_t|_{st2} = \hat{T}_t|_{st1}$.

ST2 is more efficient than ST1 for $N_p > 12$ as long as $\overline{N}_{cl}$, the average number of elements that claim a data point, is lower than 80% of the number of elements, N_e. This condition is satisfied for all but very distorted meshes. The extra bounding-box build time $\hat{T}_b$ is compensated by the decreasing slope in the $\hat{T}_t$ vs. N_p curve. Figure 7 is a comparison of ST2 and ST3. The figure shows that for any reasonable mesh, $N_e \geq 30$ and $N_p \geq 50$, it is advantageous to use a virtual mesh. In Section 5 we validate these conclusions experimentally, using a C++ implementation of the different search techniques.

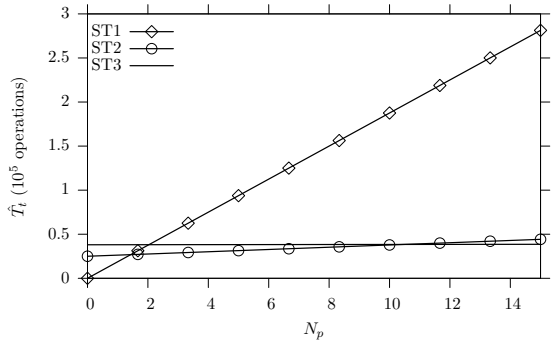


Figure 5. *Number of operations vs. data points for methods ST1, ST2 and ST3 ($N_e = 10000$, $\overline{N}_{cl} = 10$ and $\alpha = 0.05$)*

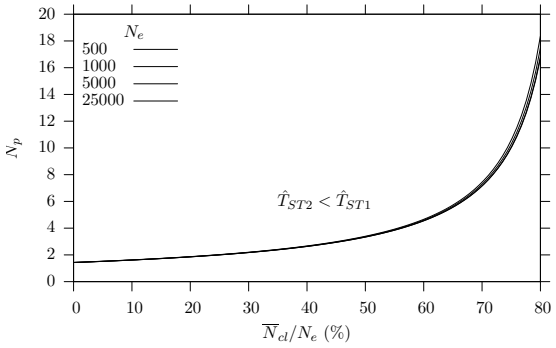


Figure 6. *Comparison of ST1 and ST2. The curves represent iso-lines where $\hat{T}_t|_{st2} = \hat{T}_t|_{st1}$. The regions above the curve are such that $\hat{T}_t|_{st2} < \hat{T}_t|_{st1}$. Notice that N_e does not affect the curves to a significant degree*

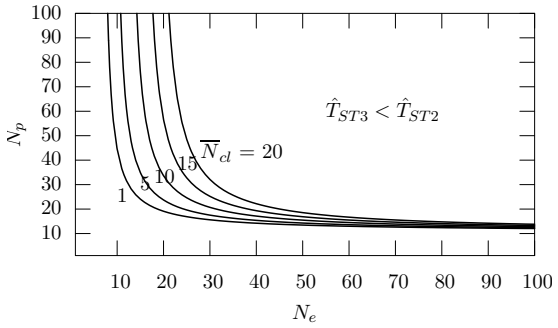


Figure 7. *Comparison of ST3 and ST2. The curves represent iso-lines where* $\hat{T}_t|_{st3} = \hat{T}_t|_{st2}$. *The curves use the optimum* α *and* $\overline{N}_{cl} = 1, 5, 10, 15, 20$

4. Inverse mapping

Finite elements shape coefficients N_j are generally known functions of an element's local coordinates, $\vec{\xi}$ (Equation [3]). Hence, to evaluate these coefficients it is often necessary to compute $\vec{\xi}(p)$ from $\vec{x}(p)$. Equation [18] defines a non-linear mapping function from the reference coordinates $\vec{\xi}$ to the real coordinates $\vec{x}$,

$$x(p) = \sum_{j=1}^{\text{no. nodes in } e(p)} S_j\left(\vec{\xi}(p)\right) x(n_j^{e(p)}). \qquad [18]$$

The coefficients S_j are also known functions of $\vec{\xi}$, and are the same as N_j for isoparametric elements. The opposite operation (*i.e.* calculating $\vec{\xi}(p)$ from $\vec{x}(p)$) is called inverse mapping.

Inverse mapping is a non-trivial operation, which needs to be performed numerically, although it is possible to invert the shape functions analytically in special cases. Whichever technique is applicable, the accuracy of the inversion can be verified as illustrated in Figure 8. If the inversion is successful the difference $\delta_\xi = \|\vec{\xi}^* - \vec{\xi}^T\|$ should approach machine precision. Since $\vec{\xi}^T$ is unknown, δ_ξ cannot be computed directly. Instead, by mapping $S : \vec{\xi}^* \rightarrow \vec{x}^*$, the value of $\delta_x = \|\vec{x}^* - \vec{x}\|$ is used as a measure of the inverse mapping error. For shape functions where the analytical solution of the inverse is unknown, δ_x is typically taken as the cost function to be minimized. Section 5 implements both analytic and iterative inverse mapping strategies for a QUAD4 element. The iterative approach uses a Newton-Raphson optimizer to minimize δ_x, by varying $\vec{\xi}$. the appendix Section 8 discusses the analytical results for QUAD4 elements, with more detail found in (Silva *et al.*, 2007b).

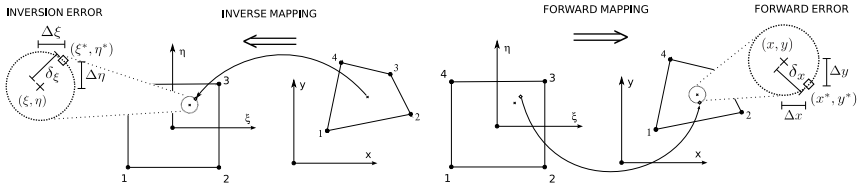


Figure 8. *An explanation of how to check the inverse mapping accuracy. (a) The inverse mapping function $S^{-1} : \vec{x} \rightarrow \vec{\xi}^*$ may contain errors $\delta_\xi = \|\vec{\xi}^* - \vec{\xi}^T\|$ caused by convergence accuracy, where $\vec{\xi}^T$ are the target coordinates. (b) The inversion error δ_ξ cannot be computed directly, since $\vec{\xi}^T$ is unknown. Instead, the accuracy of the inversion can be determined by mapping $\vec{\xi}^*$ to the real coordinate system $S : \vec{\xi}^* \rightarrow \vec{x}^*$ and computing the error $\delta_x = \|\vec{x}^* - \vec{x}\|$. If the inversion is successful, δ_x should approach zero*

5. Numerical experiments

The efficiencies of owner element search algorithms were estimated in Section 3.3 by counting operations. This assumes that other contributions to the algorithm's time complexity (memory allocation and different computation costs for integer and floating point arithmetic) are negligible. This section checks these assumptions by comparing the theoretical estimates with numerical experiments. The computation times are measured when interpolating data-point grids with varying number of points N_p over meshes with different numbers of elements N_e. The experiments are conducted with a C++ implementation of the interpolation algorithms, with an ABAQUS 6.4 finite element solver (ABAQUS Inc., 2003). The computer test-bed is a Toshiba satellite A60 with a Pentium 4 processor running on a GNU/Linux Debian 3.1 operating system (Hill *et al.*, 2005). The FE model is the open-hole plate specimen illustrated in Figure 9, see (Cardenas-Garcia *et al.*, 2006; Lin *et al.*, 1999). The model's lower edge is fully constrained, and the upper edge subjected to a 1000 kPa surface traction. The model and boundary conditions are based on an experiment conducted by the European Aeronautic Defense and Space Company (EADS) (Silva *et al.*, 2007a). The data points $\mathbf{p}$ are located on a grid of step s (Figure 9)[2]. The total processing time T_t is measured by an embedded C++ timer, which measures the time in seconds from the beginning of the interpolation procedure until all data points in the grid have been interpolated. The procedure is repeated with different mesh sizes, N_e, and grid steps, s.

Figure 10(a) shows the test results for the ST2 algorithm, which uses the bounding-box and cross-product tests to sequentially scan the entire list of finite elements. The

2. The data point grids encompass only a square window on the center of the model.

execution time T_t increases linearly with N_e and N_p when N_p and N_e are held constant, respectively. Notice that the slopes of T_t increase with N_e and N_p. The experimental behavior is in agreement with the $\mathcal{O}(N_e \times N_p)$ time complexity $\hat{T}_t|_{st2}$ in Section 3.3 (Table 1). Figure 10(b) shows the results for the ST3 algorithm, which uses a combination of the cross-product and bounding-box tests and a virtual mesh container. Clearly, the interpolation algorithm using the virtual mesh is more efficient than the previous algorithm. For large number of points ($N_e \geq 5000$), the virtual mesh reduces computation times by a factor of 10 to 60. Notice that T_t varies linearly with N_p and N_e, but unlike the previous case, the slopes of these curves are independent of N_e and N_p. These results match the $\mathcal{O}(\mathcal{A}N_e + \mathcal{B}N_p)$ in the time complexity $\hat{T}_t|_{st3}$ ($\mathcal{A}$ and $\mathcal{B}$ are functions of mesh distortion, but independent of N_e and N_p, see Table 1).

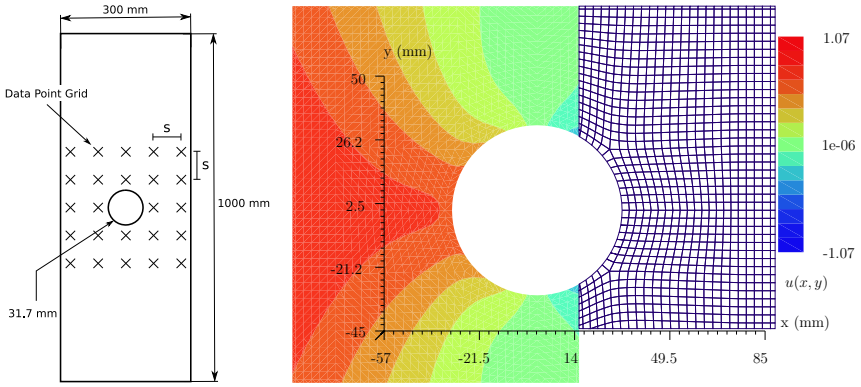
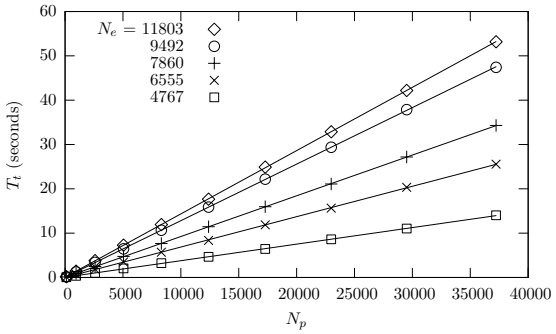


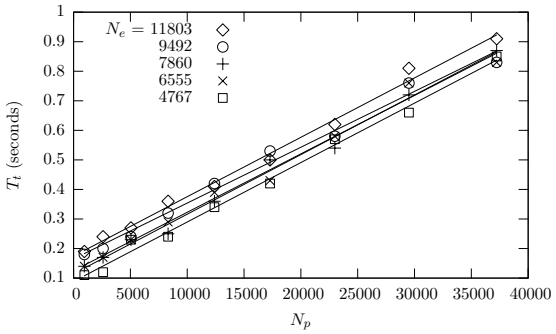
Figure 9. *(a) Test-Specimen Geometry: The model is an open-hole tensile test with the lower edge fully constrained and the upper edge subjected to a 1000 kPa surface traction. (b) Details of the finite element mesh ($N_e = 33,000$) with displacements in mm*

Next, Figures 11(a) and 11(b) show the recorded inverse mapping times for analytic and iterative inverse mapping algorithms. The iterative technique is a 2D Newton-Raphson minimization of δ_x. The optimizer has a stopping criterion of $|\delta_x| < 1e^{-5}$mm. Inverse mapping times for both techniques show a linear variation with N_p and no tendency with respects to N_e. The graphs show a scatter pattern because the inverse mapping cost depends on the position of data points inside an element (see appendix Section 8). These figures illustrate that inverse mapping can be performed very quickly in comparison to the owner element search. For the analytic solution, nearly 250,000 data points are interpolated in less than 1 CPU second. The iterative approach is significantly slower, but is still capable of interpolating 40,000 points in about 5 seconds. If the field comparison is performed repeatedly over a constant mesh, the index of the owner element and reference coordinates of each data

point can be saved and reused in each subsequent comparison. This practice further reduces interpolation time.



(a) ST2



(b) ST3

Figure 10. *Element search times (measured numerically)*

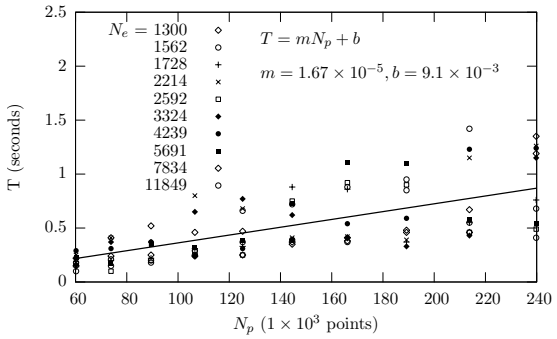
This interpolation technique is now applied to the construction and visualization of error maps. The plate with a hole (Figure 9) is composed of an orthotropic material of properties $E_{11} = 100$ GPa, $E_{22} = 36$ GPa, $G_{12} = 25$ GPa, $\nu_{12} = 0.54$ GPa. A "reference model" is solved with a fine mesh of approximately 33,000 elements and its nodal displacements are taken as the experimental data field see (Figure 9(b)). The error maps are produced by solving a FE model with the same material properties, but with a coarser mesh, $N_e = 2,500$. The coarser mesh is compared with the reference mesh using FE shape function interpolation, Matlab's *griddata* nearest neighbor approximation, linear and cubic interpolation. The processing time for each technique was 6, 10, 14 and 14 seconds, respectively [3].

3. The processing times include reading a matrix of data points and displacements for the reference and test meshes, interpolating the test mesh at reference coordinates, then writing the interpolated field onto a file.

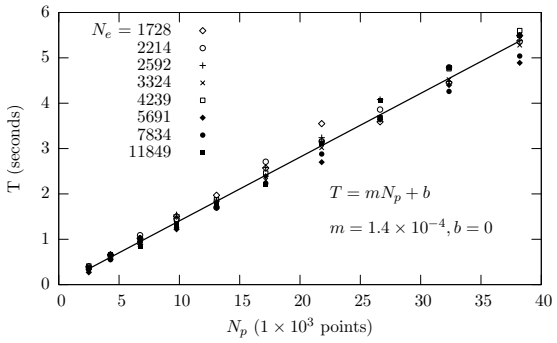
Since the test FE model has the same geometry, material properties and boundary conditions as the reference model, the difference between the test and reference meshes is only due to discretization and interpolation changes, which is small. Figure 12 shows error maps for all interpolation strategies of the normalized distance,

$$J(p) = \sqrt{\left(\frac{u(p)^{ref} - u(p)^{test}}{u_{max}^{ref} - u_{min}^{ref}}\right)^2 + \left(\frac{v(p)^{ref} - v(p)^{test}}{v_{max}^{ref} - v_{min}^{ref}}\right)^2}, \qquad [19]$$

where the data points p are nodal displacements of the reference mesh. Notice that all error maps are very small in scale, the smallest being FE shape function interpolation, followed by linear, cubic and nearest neighbor interpolation. Although the variation in scale for all techniques is small (*i.e.* $0.0034 \leq \Delta J_{scale} \leq 0.048$), their relative variation with respects to FE shape function interpolation is significant (*i.e.* $54.8\% \leq \Delta J_{scale} \leq 793\%$).

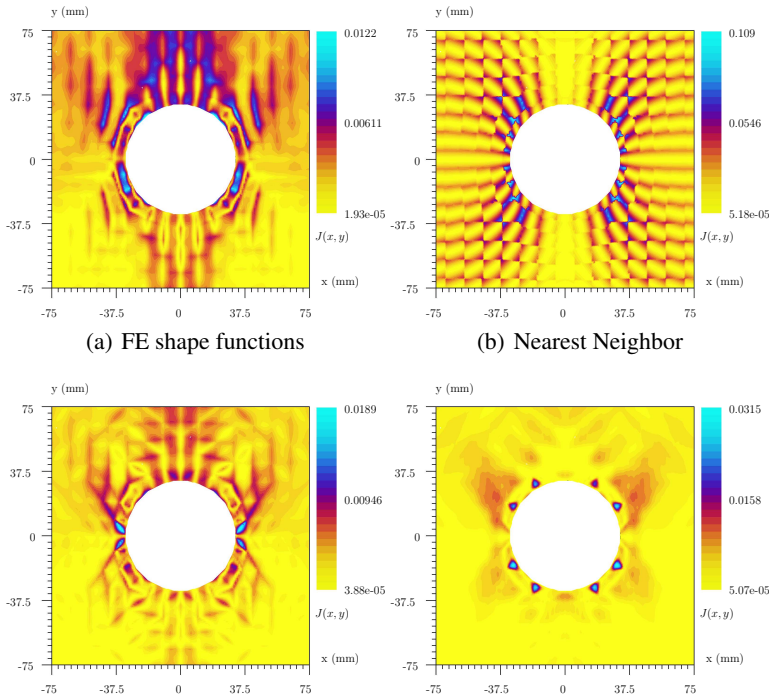


(a) Analytic inverse mapping



(b) Iterative inverse mapping

Figure 11. *Inverse mapping times using (a) analytic and (b) iterative inversion algorithms*



(a) FE shape functions (b) Nearest Neighbor

(c) Linear Interpolation (d) Cubic Interpolation

Figure 12. *Error maps $J(p)$ between a reference mesh of $N_e = 33,000$ and a mesh of $N_e = 2,500$ using (a) FE shape function interpolation, Matlab's (b) nearest neighbor approximation (c) linear and (d) cubic interpolation*

6. Conclusions

This article adapted and evaluated existing algorithms for mapping mesh information to arbitrary points using finite element shape functions. The algorithms consist of two parts: identification of the owner element, and mapping of interpolation points into the element's reference coordinate system. Interpolation is then a direct evaluation of the element's shape functions.

The strategy for determining the owner element for each data point is critical to the numerical efficiency of the procedure. The use of a virtual mesh is recommended as an alternative to sequentially searching the entire list of finite elements. With this indexing technique, time complexity increases linearly with mesh size N_e and number of data points N_p, contrary to sequential techniques which show a quadratically growing cost. In addition to the virtual mesh, an efficient test for determining if a point belongs to an element has been analyzed. The recommended test is a combi-

nation of a fast approximate test based on bounding-boxes and an exact test based on cross-products. Inverse mapping has been discussed through the analytical example of bilinear quadrilateral elements, and numerically in general cases.

Finally, the different interpolation strategies have been implemented in C++, and applied to a finite element model of open-hole tensile test. As a typical example, the best strategy (virtual mesh, mixed test) interpolates over 40,000 data points in less than 1 second for a finite elements mesh of 10,000 elements on a Pentium 4, 2.8 GHz computer.

7. References

ABAQUS Inc., *ABAQUS/CAE user's manual : version 6.4.*, Pawtucket, RI : ABAQUS, 2003.

Amiot F., Hild F., Roger J. P., " Identification of elastic property and loading fields from full-field displacement measurements", *International Journal of Solids and Structures*, vol. 44, n° 9, p. 2863-2887, May, 2007.

Avril S., Pierron F., " General framework for the identification of constitutive parameters from full-field measurements in linear elasticity", *International Journal of Solids and Structures*, vol. 44, n° 14-15, p. 4978-5002, July, 2007.

Barber B. C., Dobkin D. P., Huhdanpaa H., " The quickhull algorithm for convex hulls", *ACM Trans. Math. Softw.*, vol. 22, n° 4, p. 469-483, 1996.

Beckert A., " Coupling fluid (CFD) and structural (FE) models using finite interpolation elements", *Aerospace Science and Technology*, vol. 4, n° 1, p. 13 - 22, January, 2000.

Bledzki A. K., Kessler A., Rikards R., Chate A., " Determination of elastic constants of glass/epoxy unidirectional laminates by the vibration testing of plates", *Composites Science and Technology*, vol. 59, n° 13, p. 2015-2024, October, 1999.

Bourke P., " Determining if a point lies on the interior of a polygon", *online tutorials*, Crawley, AU., November, 1989. Available http://local.wasp.uwa.edu.au/pbourke/geometry/insidepoly/index.html.

Bruno L., Furgiuele F. M., Pagnotta L., Poggialini A., " A full-field approach for the elastic characterization of anisotropic materials", *Optics and Lasers in Engineering*, vol. 37, p. 417-431, April, 2002.

Cardenas-Garcia J. F., Preidikman S., " Solution of the moire hole drilling method using a finite-element-method-based approach", *International Journal of Solids and Structures*, vol. 43, n° 22-23, p. 6751-6766, November, 2006.

Cressie N., *Statistics for spatial data*, 1st edn, Wiley-Interscience, New York, 15 January, 1993.

Cugnoni J., Gmur T., Schorderet A., " Identification by modal analysis of composite structures modelled with FSDT and HSDT laminated shell finite elements", *Composites Part A-Applied Science and Manufacturing*, vol. 35, n° 7-8, p. 977 - 987, 2004.

Faucher V., Combescure A., " A time and space mortar method for coupling linear modal subdomains and non-linear subdomains in explicit structural dynamics", *Computer Methods in Applied Mechanics and Engineering*, vol. 192, n° 5-6, p. 509-533, January, 2003.

Fukushima Y., Cayol V., Durand P., " Finding realistic dike models from interferometric synthetic aperture radar data: The February 2000 eruption at Piton de la Fournaise", *Journal of Geophysical Research-Solid Earth*, 23 March, 2005.

Genovese K., Lamberti L., Pappalettere C., " Mechanical characterization of hyperelastic materials with fringe projection and optimization techniques", *Optics and Lasers in Engineering*, vol. 44, p. 423-442, May, 2006.

Grediac M., " The use of full-field measurement methods in composite material characterization: interest and limitations", *Composites Part A: Applied Science and Manufacturing*, vol. 35, n° 7-8, p. 751-761, July, 2004.

Hill B. M., Harris D. B., Vyas J., *Debian GNU/Linux 3.1 Bible*, John Wiley & Sons, New York, 2005.

Kajberg J., Lindkvist G., " Characterisation of materials subjected to large strains by inverse modelling based on in-plane displacement fields", *International Journal of Solids and Structures*, vol. 41, n° 13, p. 3439-3459, February, 2004.

Lin S. T., Rowlands R. E., " Hybrid stress analysis", *Optics and Lasers in Engineering*, vol. 32, n° 3, p. 257-298, September, 1999.

Mathworks, *Matlab: The language of technical computing*, 7 edn, Mathworks, Natic, June, 2004.

Meuwissen M. H. H., Oomens C. W. J., Baaijens F. P. T., Petterson R., Janssen J. D., " Determination of the elasto-plastic properties of aluminium using a mixed numerical-experimental method", *Journal of Materials Processing Technology*, vol. 75, n° 1-3, p. 204 - 211, March, 1998.

Molimard J., Le Riche R., Vautrin A., Lee J., " Identification of the four orthotropic plate stiffnesses using a single open-hole tensile test", *Experimental Mechanics*, vol. 45, n° 5, p. 404-411, 2005.

Nikishkov G., " Generating contours on FEM/BEM higher-order surfaces using Java 3D textures", *Advances in Engineering Software*, vol. 34, n° 8, p. 469-476, August, 2003.

Padmanabhan S., Hubner J. P., Kumar A. V., Ifju P. G., " Load and boundary condition calibration using full-field strain measurement", *Experimental Mechanics*, vol. 46, n° 5, p. 569 - 578, October, 2006.

Pagnacco E., Moreau A., Lemosse D., " Inverse strategies for the identification of elastic and viscoelastic material parameters using full-field measurements", *Materials Science and Engineering: A*, vol. 452-453, p. 737-745, April, 2007.

Rassineux A., Maillage automatique tridimentionnel par une méthode frontale pour la méthode des éléments finis, PhD thesis, Université Henri Poincaré, Vandoeuvre les Nancy, France, 1994. (in French).

Rumpf M., " Recent numerical methods - A challenge for efficient visualization", *Future Generation Computer Systems*, vol. 15, n° 1, p. 43-58, September, 1999.

Sandwell D. T., " Biharmonic spline interpolation of GEOS-3 and SEASAT altimeter data", *Geophysical Research Letters*, vol. 12, n° 2, p. 139-142, February, 1987.

Silva G. H. C., Le Riche R., Molimard J., Vautrin A., " Aspects numériques de la comparaison éléments finis-mesures de champs: Application à l'identification de propriétés de matériaux", *Huitième colloque national en calcul des structures*, vol. 1, Lavoisier, Giens, p. 287-292, May, 2007a.

Silva G. H. C., Le Riche R., Molimard J., Vautrin A., Exact and efficient interpolation using finite element shape functions, Technical report, LTDS, St. Etienne, France, 4 January, 2007b. available https://hal.archives-ouvertes.fr/hal-00122640/en/.

Van Loon R., Anderson P. D., Van de Vosse F. N., " A fluid-structure interaction method with solid-rigid contact for heart valve dynamics", *Journal of Computational Physics*, vol. 217, n° 2, p. 806 - 823, SEP 20, 2006.

Wang Y., Cuitino A. M., " Full-field measurements of heterogeneous deformation patterns on polymeric foams using digital image correlation", *International Journal of Solids and Structures*, vol. 39, n° 13-14, p. 3777 - 3796, JUN-JUL, 2002.

8. Appendix: Inverse mapping for QUAD4 elements

This section establishes a mathematical basis for the inversion technique presented in Section 4. Let the shape functions, Equation [18], of a 2D bi-linear quadrilateral element be rewritten as

$$\begin{aligned} x_0 = x - a_0 &= a_1\xi + a_2\eta + a_3\xi\eta \\ \text{and } y_0 = y - b_0 &= b_1\xi + b_2\eta + b_3\xi\eta, \end{aligned} \qquad [20]$$

such that $(\xi, \eta)\epsilon[-1, 1]^2$. The coefficients a_i and b_i are

$$\begin{array}{rclrcl} a_0 &=& \frac{1}{4}\left[(x_{n1}+x_{n2})+(x_{n3}+x_{n4})\right] & b_0 &=& \frac{1}{4}\left[(y_{n1}+y_{n2})+(y_{n3}+y_{n4})\right] \\ a_1 &=& \frac{1}{4}\left[(x_{n2}-x_{n1})+(x_{n3}-x_{n4})\right] & b_1 &=& \frac{1}{4}\left[(y_{n2}-y_{n1})+(y_{n3}-y_{n4})\right] \\ a_2 &=& \frac{1}{4}\left[(x_{n3}+x_{n4})-(x_{n1}+x_{n2})\right] & b_2 &=& \frac{1}{4}\left[(y_{n3}+y_{n4})-(y_{n1}+y_{n2})\right] \\ a_3 &=& \frac{1}{4}\left[(x_{n1}-x_{n2})+(x_{n3}-x_{n4})\right] & b_3 &=& \frac{1}{4}\left[(y_{n1}-y_{n2})+(y_{n3}-y_{n4})\right], \end{array}$$

where x_{ni} and y_{ni} are node coordinates.

Definition 1. *Geometrically admissible element*: A geometrically admissible element is such that the nodal arrangement in the real coordinate system does not contain any crossing segments, and has a non-zero area.

Proposition 1. *If $a_1 + a_3\eta \neq 0$, then for all geometrically admissible elements one of the solutions of the system*

$$\xi = \frac{x_0 - a_2\eta}{a_1 + a_3\eta} \quad \textit{and} \quad A\eta^2 + B\eta + C = 0, \qquad [21]$$

is an inverse mapping solution. The coefficients A, B and C are

$$\begin{array}{rcl} A &=& a_3b_2 - a_2b_3\ , \\ B &=& (x_0b_3 + a_1b_2) - (y_0a_3 + a_2b_1) \textit{ and} \\ C &=& x_0b_1 - y_0a_1. \end{array}$$

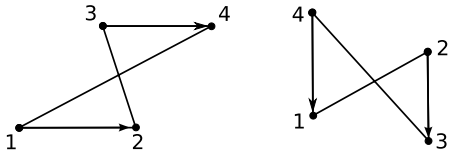


Figure 13. *If any two opposite node vectors are parallel and point to the same direction (a) $\vec{12}$ and $\vec{23}$ or (b) $\vec{23}$ and $\vec{41}$, the nodal configuration is geometrically inadmissible*

Proposition 2. *If $a_1 + a_3\eta = 0$ and $a_3 \neq 0$, then for all geometrically admissible elements,*

$$\xi = \frac{y_0 a_3 + a_1 b_2}{a_3 b_1 - a_1 b_3} \quad \textit{and} \quad \eta = \frac{-a_1}{a_3} \qquad [22]$$

is an inverse-mapping solution.

Proof. For all geometrically admissible elements such that $a_1 + a_3\eta = 0$, $a_1 \neq 0$ and $a_3 \neq 0$, we prove that $a_3 b_1 - b_3 a_1 \neq 0$, which guarantees [22] can be used without division by zero. Assuming without loss of generality that the nodes in the reference element are numbered counter-clockwise, if $a_3 b_1 - b_3 a_1 = 0$ then $a_3 b_1 = b_3 a_1$, where

$$\begin{aligned} a_3 b_1 &= [(y_{n2} - y_{n1}) + (y_{n3} - y_{n4})][(x_{n1} - x_{n2}) + (x_{n3} - x_{n4})], \\ b_3 a_1 &= [(y_{n1} - y_{n2}) + (y_{n3} - y_{n4})][(x_{n2} - x_{n1}) + (x_{n3} - x_{n4})]. \end{aligned}$$

Simplifying the expression yields,

$$(x_{n2} - x_{n1})(y_{n4} - y_{n3}) = (x_{n4} - x_{n3})(y_{n2} - y_{n1}),$$

$$\Rightarrow \frac{(y_{n4} - y_{n3})}{(x_{n4} - x_{n3})} = \frac{(y_{n2} - y_{n1})}{(x_{n2} - x_{n1})}.$$

Thus, the vectors $\vec{12}$ and $\vec{34}$ point to the same direction, which results in a geometrically inadmissible configuration (see Figure 13).

□

Proposition 3. *For all geometrically admissible elements, such that* $a_1 + a_3\eta = 0$ *and* $a_3 = 0$*,*

$$\xi = \frac{y_0 a_2 - b_2 x_0}{b_3 x_0 + a_2 b_1} \quad \textit{and} \quad \eta = \frac{x_0}{a_2} \tag{23}$$

is an inverse mapping solution.

Proof. The proof follows directly from substitution into Equation [20]. It shows that if there is a division by zero in Equation [23] the element is geometrically inadmissible.

Part 1: For all geometrically admissible elements, such that $a_1 + a_3\eta = 0$, and $a_3 = 0$, it is necessary that $a_2 \neq 0$. If $a_1 + a_3\eta = 0$, and $a_3 = 0$ then $a_1 = 0$, thus

$$\begin{gathered}(x_{n2} - x_{n1}) = (x_{n4} - x_{n3}) \text{ and } (x_{n1} - x_{n2}) = (x_{n4} - x_{n2}),\\ \Rightarrow x_{n3} = x_{n4} \text{ and } x_{n1} = x_{n2}.\end{gathered}$$

If $a_2 = 0$, then

$$\begin{gathered}x_{n3} + x_{n4} = x_{n1} + x_{n2},\\ \Rightarrow x_{n1} = x_{n2} = x_{n3} = x_{n4}.\end{gathered}$$

Hence, the resulting node coordinates are collinear, which is geometrically inadmissible.

Part 2: Assume that $a_2 \neq 0$ and that $b_1 a_2 + b_3 x_0 = 0$. Using $x_0 = x - a_0$, it follows that

$$b_3 x = b_3 a_0 - b_1 a_2, \text{ where}$$

$$\begin{gathered}b_3 = [(y_{n1} - y_{n2}) + (y_{n3} - y_{n4})]\\ b_3 a_0 = \tfrac{1}{4}[(y_{n1} - y_{n2}) + (y_{n3} - y_{n4})][x_{n1} + x_{n2} + x_{n3} + x_{n4}]\\ b_1 a_2 = \tfrac{1}{4}[(y_{n2} - y_{n1}) + (y_{n3} - y_{n4})][(x_{n3} + x_{n4}) - (x_{n1} + x_{n2})].\end{gathered}$$

case 1: $b_3 \neq 0$. Substituting $x_{n1} = x_{n2}$ and $x_{n3} = n_{n4}$ and simplifying yields,

$$\begin{gathered}[(y_{n1} - y_{n2}) + (y_{n3} - y_{n4})]x = [(y_{n1} - y_{n2}) + (y_{n3} - y_{n4})][x_{n1} + x_{n3}]\\ +[(y_{n2} - y_{n1}) + (y_{n3} - y_{n4})][x_{n3} - x_{n1}].\end{gathered}$$

Solving for x,

$$x = \frac{(y_{n3} - y_{n4})x_{n1} + (y_{n1} - y_{n2})x_{n3}}{(y_{n3} - y_{n4}) + (y_{n1} - y_{n2})}.$$

Defining α and β, where $\alpha + \beta = 1$ yields,

$$x = \alpha x_{n1} + \beta x_{n3} \; .$$

For any internal point it is necessary that both $\alpha > 0$ and $\beta > 0$. However, if $y_{n3} - y_{n4} > 0$ it follows that $y_{n1} - y_{n2} < 0$ for an element with no intersecting edges. Hence, α and β always have opposite signs for a valid element, thus the point x is not an internal point.

case 2: $b_3 = 0$. It follows that

$$\begin{gathered}
(y_{n3} - y_{n4}) + (y_{n1} - y_{n2}) = 0 \\
y_{n1} - y_{n2} = y_{n4} - y_{n3} \\
0 = (y_{n3} - y_{n4})x_{n1} + (y_{n1} - y_{n2})x_{n3} \\
0 = (y_{n2} - y_{n1})x_{n1} + (y_{n1} - y_{n2})x_{n3} \\
x_{n1} - x_{n3} = 0 \Rightarrow x_{n1} = x_{n3} \\
\Rightarrow x_{n1} = x_{n2} = x_{n3} = x_{n4}
\end{gathered}$$

The result is a collinear nodal arrangement, which is geometrically inadmissible. □