A generalized finite element formulation for arbitrary basis functions: from isogeometric analysis to XFEM

David J. Benson, Yuri Bazilevs, Emmanuel de Luycker, Ming Chen Hsu, Colin Scott, Thomas Joseph Robert Hughe, Ted Belytschko

▶ **To cite this version:**

David J. Benson, Yuri Bazilevs, Emmanuel de Luycker, Ming Chen Hsu, Colin Scott, et al.. A generalized finite element formulation for arbitrary basis functions: from isogeometric analysis to XFEM. International Journal for Numerical Methods in Engineering, 2010, 83 (6), pp.765-785. 10.1002/nme.2864 . hal-01657907

HAL Id: hal-01657907
https://hal.science/hal-01657907

Submitted on 7 Dec 2017

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

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

Open Archive Toulouse Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in: http://oatao.univ-toulouse.fr/
Eprints ID: 18483

To link to this article: DOI:10.1002/nme.2864
URL:http://onlinelibrary.wiley.com/doi/10.1002/nme.2864/abstract

To cite this version:

Benson, David J. and Bazilevs, Yuri and De Luycker, Emmanuel and Hsu, Ming Chen and Scott, Colin and Hughe, Thomas Joseph Robert and Belytschko, Ted *A generalized finite element formulation for arbitrary basis functions : from isogeometric analysis to XFEM.* (2010) International Journal for Numerical Methods in Engineering, vol. 83 (n° 6). pp. 765-785. ISSN 1097-0207

Any correspondence concerning this service should be sent to the repository administrator: staff-oatao@listes-diff.inp-toulouse.fr

A generalized finite element formulation for arbitrary basis functions: From isogeometric analysis to XFEM

D. J. Benson[1,*,†], Y. Bazilevs[1], E. De Luycker[1], M.-C. Hsu[1], M. Scott[2], T. J. R. Hughes[2] and T. Belytschko[3]

[1]*Department of Structural Engineering, University of California, San Diego 9500 Gilman Drive, La Jolla, CA 92093, U.S.A.*
[2]*Institute for Computational Engineering and Sciences, The University of Texas at Austin, 201 East 24th Street, 1 University Station C0200, Austin, TX 78712, U.S.A.*
[3]*Department of Mechanical Engineering, 2145 North Sheridan Road, Northwestern University, Evanston, IL 60208-3111, U.S.A.*

SUMMARY

Many of the formulations of current research interest, including iosogeometric methods and the extended finite element method, use nontraditional basis functions. Some, such as subdivision surfaces, may not have convenient analytical representations. The concept of an element, if appropriate at all, no longer coincides with the traditional definition. Developing a new software for each new class of basis functions is a large research burden, especially, if the problems involve large deformations, non-linear materials, and contact. The objective of this paper is to present a method that separates as much as possible the generation and evaluation of the basis functions from the analysis, resulting in a formulation that can be implemented within the traditional structure of a finite element program but that permits the use of arbitrary sets of basis functions that are defined only through the input file. Elements ranging from a traditional linear four-node tetrahedron through a higher-order element combining XFEM and isogeometric analysis may be specified entirely through an input file without any additional programming. Examples of this framework to applications with Lagrange elements, isogeometric elements, and XFEM basis functions for fracture are presented.

KEY WORDS: isogeometric analysis; NURBS; shells; XFEM; generalized elements

*Correspondence to: D. J. Benson, Department of Structural Engineering, University of California, San Diego 9500 Gilman Drive, La Jolla, CA 92093, U.S.A.
†E-mail: dbenson@ucsd.edu

1. INTRODUCTION

Lagrange interpolation polynomials have been the standard choice for basis functions in finite element analysis from almost the very beginning of finite element research. They are the overwhelming choice today whether the preferred metric is their prevalence in commercial codes or in research articles. In the last few years, however, the research on different classes of basis functions has dramatically increased in an effort to circumvent the limitations of Lagrange polynomials.

Isogeometric analysis [1] uses basis functions from computer-aided design (CAD), including NURBS [1, 2], T-Splines [3, 4], and subdivision surfaces [5–7]. One major goal of this approach is direct design-to-analysis without the intermediate step of mesh generation. An exact representation of the geometry, e.g. via an IGES file, provides the definitions of the basis functions. Additional benefits may include higher-order accuracy [2, 8–14], C^1 or higher continuity, and increased robustness [15].

The extended finite element method (XFEM) [16–19] was developed to enhance the solution accuracy for particular classes of problems, such as fracture analysis, where the standard basis functions have difficulties in capturing the exact solution. The displacement field is typically enriched to include displacement fields approximating known classical solutions. The approach is general enough, however, to be applied to problems without classical solutions such as contact in multi-material arbitrary Lagrangian Eulerian (MMALE) calculations [20, 21].

The two broad classes of methods mentioned here are not intended to be exhaustive, but only a representative listing of the diversity of basis functions currently being investigated by the research community.

Traditionally, each choice of basis function is implemented in its own software. Equation solvers, material models, and boundary conditions must additionally be implemented even to solve simple problems, adding the burden of software infrastructure development to the basic research.

The objective of this paper is to present a framework that separates as much as possible the generation and evaluation of the basis functions from the analysis, resulting in a formulation that is readily implemented within the structure of a traditional finite element program and once implemented, permits the use of basis functions that need only to be defined through the input file. This separation facilitates

1. Direct design-to-analysis by permitting CAD software to provide both the exact geometry and the basis functions used to define it.
2. The rapid prototyping of elements for research by eliminating the need for implementing elements that are basis function specific or the additional infrastructure for material models and complicated boundary conditions, e.g. contact.
3. Teaching finite element methods in a hands-on computational environment that is not tied to a specific programming language.

The generalized element formulation discussed here is generalized in the sense that it is not tied to specific choices of basis functions, integration rules, or coordinates. *Elements ranging from a traditional linear four-node tetrahedron through a higher-order element combining XFEM and isogeometric analysis may be specified entirely through an input file without any additional programming.* Our implementation in LS-DYNA [22] of a generalized solid element and a generalized Reissner–Mindlin shell element [23] is intended to be illustrative of our approach, but not exhaustive. This element formulation can be implemented in virtually any finite element program,

and permits the extensive capabilities of mature production codes to be utilized in research. We anticipate that further generalized element capabilities will be added in the future.

2. THE GENERALIZED SOLID ELEMENT

2.1. Solid element kinematics

In the current configuration, the momentum equation for a solid body (Figure 1) is

$$\nabla \cdot \boldsymbol{\sigma} + \mathbf{b} - \rho \ddot{\boldsymbol{x}} = 0, \tag{1}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress, $\mathbf{b}$ is the body force, ρ is the density, and $\boldsymbol{x}$ is the current spatial coordinate. On a region Γ_h with an exterior normal $\boldsymbol{n}$, a traction boundary condition is imposed,

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{h}(\boldsymbol{x}, t), \tag{2}$$

and displacement boundary conditions are imposed on Γ_d,

$$\boldsymbol{u} = \boldsymbol{d}(\boldsymbol{x}, t). \tag{3}$$

The initial conditions are $\boldsymbol{x}(0) = \boldsymbol{X}$ and $\dot{\boldsymbol{x}}(0) = \dot{\boldsymbol{x}}(\boldsymbol{X})$.

After the standard arguments, the weak form, or principle of virtual work, is

$$\int_\Omega \rho \ddot{\boldsymbol{x}} \cdot \delta \boldsymbol{x} + \boldsymbol{\sigma} : \delta \boldsymbol{D} \, \mathrm{d}\Omega = \int_{\Gamma_h} \boldsymbol{h} \cdot \delta \boldsymbol{x} \, \mathrm{d}\Gamma + \int_\Omega \mathbf{b} \cdot \delta \boldsymbol{x} \, \mathrm{d}\Omega, \tag{4}$$

and the virtual rate of deformation is defined as

$$\delta \boldsymbol{D} = \frac{1}{2} \left(\frac{\partial \delta \boldsymbol{x}}{\partial \boldsymbol{x}} + \frac{\partial \delta \boldsymbol{x}}{\partial \boldsymbol{x}}^{\mathrm{T}} \right). \tag{5}$$

The domain is discretized into elements that are defined by nodes numbered $A = 1, N$. The kinematics are traditionally defined in terms of the basis functions, N_A, expressed in terms of the element parametric coordinates, $\boldsymbol{s}$, and the nodal values of the coordinates, velocities, and accelerations,

$$\boldsymbol{x}(\boldsymbol{s}, t) = \sum_{A=1,N} N_A(\boldsymbol{s}) \boldsymbol{x}_A(t). \tag{6}$$

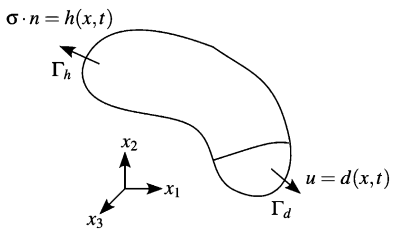

Figure 1. The generic boundary value problem in solid mechanics.

Lagrange interpolation polynomials defined locally within each element are the usual choice for the basis functions. Their advantages are well known

1. The solution values at the nodes have the appealing physical interpretation of being the value of the solution at the node.
2. The compact support guarantees the sparseness of the global stiffness and mass matrices used in implicit solution algorithms.

Elements, in this traditional setting, define regions used for both numerical quadrature and generating the basis functions. The simplicity of the Lagrange polynomials encourages a strategy, where every choice of basis function is implemented separately as its own element type to optimize its efficiency. The boundaries of the elements also define the geometry used to evaluate surface tractions, contact, and other boundary conditions. Again, the simplicity of the basis functions encourages the implementation of each type of boundary condition independently to optimize efficiency. To obtain the necessary generality in our formulation, some efficiency is necessarily sacrificed; however, the savings in development time and the use of code that has been validated using a broad class of basis functions more than compensates for this in our opinion.

The basis functions are used in the same manner in the generalized elements as in traditional finite elements, but to emphasize that they are not necessarily interpolatory, the generalized coordinates are denoted by $\boldsymbol{q}_A$. The formulation is isoparametric, with the parametric coordinates $\boldsymbol{s}$. Components of vectors are denoted with lower-case indices (e.g. the ith component of $\boldsymbol{q}_A$ is q_{Ai}), and are summed over the range 1–3 unless explicitly stated otherwise.

$$\boldsymbol{x}(\boldsymbol{s},t) = \sum_{A=1,N} N_A(\boldsymbol{s})\boldsymbol{q}_A(t), \tag{7}$$

$$\boldsymbol{X}(\boldsymbol{s}) = \boldsymbol{x}(\boldsymbol{s},0) = \sum_{A=1,N} N_A(\boldsymbol{s})\boldsymbol{q}_A(0), \tag{8}$$

$$\boldsymbol{u}(\boldsymbol{s},t) = \boldsymbol{x}(\boldsymbol{s},t) - \boldsymbol{X}(\boldsymbol{s}). \tag{9}$$

where N is the number of nodes. The term *node* is retained for the lack of a better one, but there is no requirement that the generalized coordinates associated with a node represent a physical location on the body. In fact, for most of the basis choices of interest here, e.g. NURBS [24], the generalized coordinates do not even correspond to the locations of material points.

The structure of Equation (7) is identical to the structure of Equation (6), and on substitution into the weak form it immediately leads to a set of semi-discrete equations that have the same structure as found in traditional element formulations,

$$\sum_B \boldsymbol{M}_{AB}\ddot{\boldsymbol{q}}_B + \int_\Omega \boldsymbol{B}^{\mathrm{T}}{}_A\boldsymbol{\sigma}\,\mathrm{d}\Omega = \int_{\Gamma_h} \boldsymbol{h}N_A\,\mathrm{d}\Gamma + \int_\Omega \mathbf{b}N_A\,\mathrm{d}\Omega \quad \forall A, \tag{10}$$

where $\boldsymbol{M}$ and $\boldsymbol{B}$ are the usual definitions of the mass matrix and the discrete gradient operator, respectively.

2.2. Integration

Numerical quadrature is used to evaluate the integrals in the semi-discrete equations, but with no restriction to the Gauss quadrature. Special integration rules have been developed for XFEM

[16–19] and isogeometric NURBS elements [1], but their structure is the same as the Gauss quadrature; for any function $f(\boldsymbol{x})$, the integral is given by

$$\int_{\Omega} f(\boldsymbol{x})\,\mathrm{d}\Omega = \sum_{G} w^{G} f(\boldsymbol{s}^{G}) \det(\boldsymbol{J}^{G}), \tag{11}$$

where G indicates the quadrature point, $\boldsymbol{J} = \partial \boldsymbol{x}/\partial \boldsymbol{s}$ is the Jacobian, and $\boldsymbol{s}^{G}$ are the quadrature points in the parametric domain.

The volume integrals in Equation (10), when numerically integrated as in Equation (11), require the values of w_G, $N_A(\boldsymbol{s}^G)$ and $\partial N_A(\boldsymbol{s}^G)/\partial \boldsymbol{s}$. These values are constant during the analysis if the basis functions are fixed, allowing them to be calculated outside of the analysis code. The analysis code may then read them with the standard finite element input data and use the appropriate values in a loop over the integration points. Although the basis functions are usually fixed, exceptions do exist. For example, the XFEM enrichment functions change as a crack propagates [19]. The generalized element presented here, while applicable to linear fracture analysis with XFEM, must therefore be extended to handle the moving crack problem.

Although the concept of an element is not required, it is useful for maintaining computational efficiency. The assembly operation, A, is usually inefficient since it scatters values to random locations in memory, and in the case of an implicit solution method, it may use complicated sparse matrix data structures. By aggregating the terms from a subset of the integration points before the assembly, its total cost for the calculation is minimized. To promote computational efficiency, a generalized element, therefore, consists of a collection of integration points with a common set of nonzero basis functions. Each generalized element may therefore have its own unique integration weights with their corresponding values of basis functions and derivatives, and the integrals are evaluated as the sum over the integration points in exactly the same manner as any other element.

While elements with a variable number of nodes have been developed before [25], they have always had an underlying shape, class of basis functions, and class of integration rules. *Note that no implied geometry is associated with the generalized element other than the existence of a parametric coordinate system that has the same dimension as the physical coordinate system*. The parametric system is never explicitly defined, and it appears indirectly only through the numerical values of the derivatives of the basis functions at the integration points and the integration weights themselves.

Given the absence of information on the underlying geometry, determining an appropriate characteristic length for a stable time step size in a dynamics problem solved with explicit time integration is problematical. The maximum system frequency is obtained using the power iteration method developed by Benson [26] for MMALE methods.

2.3. Example basis functions

Two different classes of basis functions are presented to illustrate the generality of the methodology. It is easy to associate the properties of the Lagrangian interpolation polynomials with Equation (7) since their structure is the same.

The control points associated with NURBS basis functions [24], which define the geometry, are *not* material points in Ω. To illustrate this point, a NURBS curve is shown in Figure 2. At the ends of the curve, the basis functions are interpolatory and the end control points lie on the curve, but elsewhere, the control points may lie on either side of the curve. In two and three dimensions, the NURBS basis functions are interpolatory only at the corners of the NURBS patches.

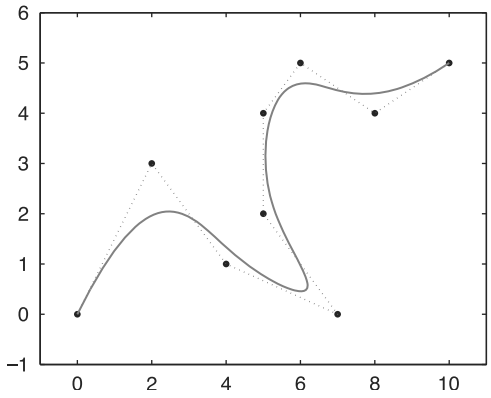

Figure 2. A NURBS curve in two dimensions. Note that the curve does not pass through the control points.

Another case, where not all the generalized coordinates correspond to material points is XFEM [16, 17]. For example, in applications involving fracture, regions of the XFEM mesh are enriched to have a form

$$\boldsymbol{u} = \sum_A N_A(\boldsymbol{s})\{\boldsymbol{u}_A + H(\boldsymbol{x})\mathbf{b}_A + \boldsymbol{u}^{\infty}_{K_I}(r,\theta)K_I\}, \tag{12}$$

where H is the jump function, $\boldsymbol{u}^{\infty}_{K_I}(r,\theta)$ are crack tip enrichment functions, and $\mathbf{b}_A$ and K_I are the generalized coordinates associated with them [19]. Note that only a subset of the nodes are enriched, and an enriched node may not use all of the available enrichment functions. A node with the maximum degree of enrichment has three degrees of freedom for $\boldsymbol{u}_A$ and three for $\mathbf{b}_A$. The fracture toughness, K_I, is treated as a global solution variable.

Note, however, that the structure of Equation (7) implies the basis functions are the same in each coordinate direction, i.e.

$$\boldsymbol{u} = \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} = \sum_A N_A \begin{Bmatrix} q_{A1} \\ q_{A2} \\ q_{A3} \end{Bmatrix}, \tag{13}$$

which certainly holds for the displacement contribution from the generalized coordinates. The enrichment for the jump function also follows this format,

$$\boldsymbol{u}^H = \begin{Bmatrix} u^H_1 \\ u^H_2 \\ u^H_3 \end{Bmatrix} = \sum_A N_A H \begin{Bmatrix} b_{A1} \\ b_{A2} \\ b_{A3} \end{Bmatrix}, \tag{14}$$

but the crack tip enrichment, $\boldsymbol{u}^{\infty}_{K_I}(r,\theta)$ does not,

$$\boldsymbol{u}^{\infty}_{K_I} = \begin{Bmatrix} \boldsymbol{u}^{\infty}_{K_I 1} \\ \boldsymbol{u}^{\infty}_{K_I 2} \\ \boldsymbol{u}^{\infty}_{K_I 3} \end{Bmatrix} = \left(\sum_A N_A \frac{\sqrt{r}}{2\sqrt{2\pi}\mu} \begin{Bmatrix} \left(-\frac{1}{2}+k\right)\cos\left(\frac{\theta}{2}\right) - \cos\left(\frac{3\theta}{2}\right) \\ \left(+\frac{1}{2}+k\right)\sin\left(\frac{\theta}{2}\right) - \sin\left(\frac{3\theta}{2}\right) \\ 0 \end{Bmatrix} \right) K_I, \tag{15}$$

$$k = 3-4\nu, \tag{16}$$

where r and θ are a cylindrical coordinate system with its origin at the crack tip.

To incorporate the crack tip enrichment functions, two additional basis functions are introduced,

$$N^{\infty}_1 = \sum_A N_A \frac{\sqrt{r}}{2\sqrt{2\pi}\mu} \left\{ \left(-\frac{1}{2}+k\right)\cos\left(\frac{\theta}{2}\right) - \cos\left(\frac{3\theta}{2}\right) \right\}, \tag{17}$$

$$N^{\infty}_2 = \sum_A N_A \frac{\sqrt{r}}{2\sqrt{2\pi}\mu} \left\{ \left(+\frac{1}{2}+k\right)\sin\left(\frac{\theta}{2}\right) - \sin\left(\frac{3\theta}{2}\right) \right\}, \tag{18}$$

and the displacement field is enriched with a surplus of degrees of freedom,

$$\boldsymbol{u}^{\infty}_{K_I} = N^{\infty}_1 \begin{Bmatrix} u^{\infty}_{11} \\ u^{\infty}_{12} \\ u^{\infty}_{13} \end{Bmatrix} + N^{\infty}_2 \begin{Bmatrix} u^{\infty}_{21} \\ u^{\infty}_{22} \\ u^{\infty}_{23} \end{Bmatrix}. \tag{19}$$

The desired displacement field is

$$\boldsymbol{u}^{\infty}_{K_I} = N^{\infty}_1 \begin{Bmatrix} K_I \\ 0 \\ 0 \end{Bmatrix} + N^{\infty}_2 \begin{Bmatrix} 0 \\ K_I \\ 0 \end{Bmatrix}, \tag{20}$$

and it is obtained by equating the terms in Equations (19) and (20) to generate the constraint equations

$$u^{\infty}_{11} - u^{\infty}_{22} = 0, \tag{21}$$

$$u^{\infty}_{12} = 0, \tag{22}$$

$$u^{\infty}_{13} = 0, \tag{23}$$

$$u^{\infty}_{21} = 0, \tag{24}$$

$$u^{\infty}_{23} = 0. \tag{25}$$

Unlike most basis functions, those defined in Equation (12) use both the parametric coordinates $\boldsymbol{s}$ and the physical coordinates $\boldsymbol{x}$. The generalized formulation developed here requires a single

coordinate system for specifying the derivatives of the basis functions and the element integration. Two alternatives are available. First, the parametric coordinate system may be used, and the required derivatives are evaluated via the chain rule, e.g.

$$\frac{\partial N_A \boldsymbol{u}_{K_I}^{\infty}}{\partial \boldsymbol{s}} = \frac{\partial N_A}{\partial \boldsymbol{s}} \boldsymbol{u}_{K_I}^{\infty} + N_A \frac{\partial \boldsymbol{u}_{K_I}^{\infty}}{\partial \boldsymbol{x}} \cdot \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{s}}. \tag{26}$$

The second alternative is, perhaps, not as intuitive, but is equally valid; the derivatives with respect to the parametric coordinate system are transformed to the physical coordinates, and the derivatives are again evaluated using the chain rule,

$$\frac{\partial N_A \boldsymbol{u}_{K_I}^{\infty}}{\partial \boldsymbol{X}} = \frac{\partial N_A}{\partial \boldsymbol{s}} \cdot \frac{\partial \boldsymbol{s}}{\partial \boldsymbol{X}} \boldsymbol{u}_{K_I}^{\infty} + N_A \frac{\partial \boldsymbol{u}_{K_I}^{\infty}}{\partial \boldsymbol{X}}, \tag{27}$$

where $\boldsymbol{X} = \boldsymbol{x}(0)$ is used to emphasize that this transformation is applied only once at $t = 0$.

2.4. Solid element data requirements

The generalized element formulation requires two types of data. First, there is the data defining the parent generalized element: the number of integration points, the number of nodes, and for each integration point G, the integration weight, w^G, and for each node in the element, the values of the shape functions, $N_A(\boldsymbol{s}^G)$ and their derivatives, $\partial N_A(\boldsymbol{s}^G)/\partial \boldsymbol{s}$. Note that this data suffices for calculating the derivatives of the shape functions with respect to the spatial and material coordinates for any elements, since the Jacobian relating the spatial or material coordinates depends only on the $\partial N_A(\boldsymbol{s}^G)/\partial \boldsymbol{s}$ and the nodal coordinates. In the keyword input language of LS-DYNA [27], these data are specified using `*DEFINE_ELEMENT_GENERALIZED_SOLID`. Second, for each generalized element, the nodes must be specified along with the material number. These data are specified using `*ELEMENT_GENERALIZED_SOLID` in the LS-DYNA input file. The details may be found in the LS-DYNA manual [27].

Since the elements are defined solely through the input file, the values of the integration weights, the shape functions, and their derivatives can be calculated in any convenient manner. Almost any computer language that is comfortable and convenient for the researcher might be used; MATLAB is an obvious choice. In the example calculations, Fortran programs were written to generate the input files for convergence studies, and a T-Spline plug-in module was used with the CAD program Rhino for the bumper.

Note that an arbitrary number of parent generalized elements may be defined in a single file. For the isogeometric analyses presented later, each element has its own unique parent element. In contrast, for a standard Lagrangian element, each element in the calculation refers to a single parent element.

3. THE GENERALIZED SHELL ELEMENT

A Reissner–Mindlin formulation, for which a C^0-continuous discretization is sufficient, is summarized here; for greater detail see [23]. Having introduced the generalized element concept in the previous section, only the element formulation is summarized here, and the superscript ℓ refers to the local, lamina coordinate system. For an excellent review of shell theories and numerical formulations see Bischoff *et al.* [28].

The shell kinematics are based on the degenerated solid element approach developed by Hughes and Liu [29]. They are derived by starting with a solid element that has linear interpolation through the thickness coupled with the desired form of interpolation on the laminae. Pairs of control points on the upper and lower surfaces define a material fiber, $\boldsymbol{y}$, that remains straight during the deformation. The motion of the fiber is, therefore, a rigid body motion (although this contradicts the zero normal stress condition) that may be described in terms of three translational velocities and either two or three rotational velocities (the rotation about the axis of the fiber does not contribute to the deformation, but it is simpler computationally to use three angular velocities in the global coordinate system).

The kinematics are, therefore, reduced to approximating a shell by a surface in space defined by the translational coordinates of a set of nodes, and the rotation of the fiber vectors attached to them. The current shell geometry is, therefore, expressed mathematically by

$$\boldsymbol{x}(\boldsymbol{s}) = \sum_A N_A(\boldsymbol{s}) \left(\boldsymbol{q}_A + \frac{h_A}{2} s_3 \hat{\boldsymbol{y}}_A \right), \tag{28}$$

where h is the thickness and $\hat{\boldsymbol{y}}_A$ is the unit fiber vector at node A. The coordinates $\boldsymbol{s}$ are from the parametric space, and, therefore, their ranges are defined by the appropriate knot vectors, and the third coordinate, $s_3 \in [-1, +1]$, is used with the standard linear interpolation functions through the thickness of the shell.

To simplify the notation, the dependence of the functions on the parametric coordinates is dropped, however, all are assumed to be evaluated at the integration point under consideration. Additionally, since the product of N_A and s_3 appears throughout the terms associated with the rotational degrees of freedom, the additional functions $\hat{N}_A = N_A s_3$ are introduced. For example, the expression for the current geometry is now written as

$$\boldsymbol{x} = \sum_A N_A \boldsymbol{q}_A + \frac{h_A}{2} \hat{N}_A \hat{\boldsymbol{y}}_A, \tag{29}$$

and the resulting Jacobian, $\boldsymbol{J}$, is therefore

$$\boldsymbol{J} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{s}} = \sum_A \frac{\partial N_A}{\partial \boldsymbol{s}} \boldsymbol{q}_A + \frac{h_A}{2} \frac{\partial \hat{N}_A}{\partial \boldsymbol{s}} \hat{\boldsymbol{y}}_A. \tag{30}$$

The velocity, $\dot{\boldsymbol{x}}$, is defined in terms of the translational velocity, $\dot{\boldsymbol{q}}_A$, and the angular velocity, $\boldsymbol{\omega}_A$, of the nodes, both in the global coordinate system,

$$\dot{\boldsymbol{x}} = \sum_A N_A \dot{\boldsymbol{q}}_A + \frac{h_A}{2} \hat{N}_A \boldsymbol{\omega}_A \times \hat{\boldsymbol{y}}_A. \tag{31}$$

Note the introduction of the angular velocity, $\boldsymbol{\omega}$, to update the orientation of the fiber vectors. The choice made here is also motivated by the simplicity of joining multiple non-smooth surfaces (e.g. a honeycomb structure). Using three rotational degrees of freedom introduces a singularity associated with the rotation about $\hat{\boldsymbol{y}}$ for smooth surfaces, which we address later. The test space, or virtual displacement, $\delta \boldsymbol{x}$, has the same structure as the velocity field,

$$\delta \boldsymbol{x} = \sum_A N_A \delta \boldsymbol{q}_A + \frac{h_A}{2} \hat{N}_A \delta \boldsymbol{\omega}_A \times \hat{\boldsymbol{y}}_A. \tag{32}$$

The basic degenerated solid formulation is modified in three ways, motivated by Belytshcko *et al.* [30, 31]. First, $\hat{\boldsymbol{y}}$ is replaced by $\boldsymbol{n}$, the unit normal in the shell laminae, throughout. The motivation for this simplification is to alleviate the artificial thinning that sometimes occurs with explicit time integration, which is caused by the scaling of the rotational inertias to maintain a large time step size. Additionally, the definition of a unique fiber direction for structures with intersecting shell surfaces is often problematical.

Second, in contrast to the standard formulations, which are focused on elements with linear basis functions, nothing is done to alleviate shear locking in the current formulation because we are interested in higher-order NURBS basis functions, where shear locking is generally not a significant problem. For lower degree NURBS, the generalization of $\bar{\boldsymbol{B}}$ for volumetric locking developed for isogeometric elements [32] may be modified for shear locking.

Finally, a corotational formulation [31] for the stress is used here instead of the Truesdell rate originally used by Hughes and Liu [29]. This choice was motivated by our desire to use the isogeometric shells for elastic-plastic materials, where the corotational formulation has been found to provide extra robustness.

The local corotational coordinate system for the stress is defined at the integration points using the invariant scheme of Hughes and Liu [29] with the convention that $\boldsymbol{e}_3^\ell$ is the direction normal to the reference surface. This coordinate system is derived from tangent vectors $\boldsymbol{t}_i$, $i=1,2$,

$$\boldsymbol{t}_i = \sum_A \frac{\partial N_A}{\partial s_i} \boldsymbol{q}_A, \tag{33}$$

and the normal vector,

$$\boldsymbol{p} = \boldsymbol{t}_1 \times \boldsymbol{t}_2, \tag{34}$$

$$\boldsymbol{n} = \boldsymbol{e}_3^\ell = \frac{\boldsymbol{p}}{\|\boldsymbol{p}\|}, \tag{35}$$

where $\|\cdot\|$ is the usual three-dimensional Euclidean norm.

3.1. Time integration

The explicit central difference method and implicit Newmark method are used in LS-DYNA to advance the solution in time, and either may be used with the element formulation presented here. Since the example calculations are explicit, the central difference method is briefly summarized. Note that this is not the 'central difference method' that is sometimes associated with descriptions of the Newmark method, and its stability is not independent of the damping. Using the lumped mass matrix, the accelerations are calculated from the assembled forces at time step n, and the velocities are updated to $n+\frac{1}{2}$,

$$\dot{\boldsymbol{q}}_A^{n+1/2} = \dot{\boldsymbol{q}}_A^{n-1/2} + \triangle t \ddot{\boldsymbol{q}}_A^n, \tag{36}$$

The translational velocities are integrated to give the current control point coordinates,

$$\boldsymbol{q}_A^{n+1} = \boldsymbol{q}_A^n + \triangle t \dot{\boldsymbol{q}}_A^{n+1/2}. \tag{37}$$

Given the absence of information on the underlying geometry, determining an appropriate characteristic length for a stable time step size in a dynamics problem solved with explicit time integration is problematical. The maximum system frequency is obtained using the power iteration method developed by Benson [26] for MMALE methods.

4. BOUNDARY CONDITIONS

At this time we have not developed generalized boundary conditions in the same format as the elements; however, they are certainly possible. Instead, we have chosen to rely on the existing capabilities within LS-DYNA to approximate the required boundary conditions to an adequate level of accuracy.

Displacement boundary conditions are imposed directly on the nodes in the standard finite element formulation. With basis functions that are not interpolatory, the introduction of a constraint equation, located at the material point where the displacement boundary condition is applied, $\boldsymbol{x}(\boldsymbol{s}_C)$, is often the most appropriate approach,

$$C_D(t) = \boldsymbol{u}(\boldsymbol{s}_D, t) + \boldsymbol{X}(\boldsymbol{s}_D) - \sum_A N_A(\boldsymbol{s}_D)\boldsymbol{q}_A(t) = 0, \tag{38}$$

where $\boldsymbol{u}(\boldsymbol{s}_D, t)$ is the specified displacement. This approach must be used with some care to avoid specifying too many constraints, a situation that may be avoided by using least squares to calculate the individual histories $\boldsymbol{q}_A(t)$. Selecting a set of points, $\mathscr{D}$, the objective function J is minimized at each time t,

$$J(t) = \frac{1}{2} \sum_{D \in \mathscr{D}} C_D^2(t), \tag{39}$$

to obtain $\boldsymbol{q}_A(t)$. Since the desired displacements are known, and independent of the solution, the least-squares solution can be evaluated before the actual analysis and the results specified in the input file.

Traction boundary conditions require the specification of the surface, Γ_h, where the boundary condition is applied. Since the generalized elements do not contain any explicit definition of their geometric boundaries, this specification must be performed separately. The geometry of Γ_h is approximated using bilinear quadrilateral *interpolation elements* so that all the existing boundary conditions (including contact) in LS-DYNA are immediately accessible. These elements are defined by *interpolation nodes*, which are material points on the element surface,

$$\boldsymbol{x}_B^I = \sum_A N_A(\boldsymbol{s}_B)\boldsymbol{q}_A, \tag{40}$$

where the superscript I indicates a variable that it is associated with an interpolation node or element. The interpolated surface geometry is used for both traction loads and contact. After the distributed loads are evaluated for the interpolated nodes and elements using the standard routines within LS-DYNA, the forces applied to the generalized coordinates are evaluated

$$\boldsymbol{F}_A = \sum_B N_A(\boldsymbol{s}_B)\boldsymbol{F}_B^I. \tag{41}$$

The same approach is used for body forces, where trilinear hexahedron interpolation elements are defined, and the body forces applied to them are distributed according to Equation (41).

The accuracy of this approximation of the distributed loading is clearly a function of the number of interpolation elements used to approximate the geometry. In the example isogeometric calculations, the number of elements in each direction is equal to order of the interpolation, p. A quadratic isogeometric shell element is divided into four interpolation elements with bilinear basis functions. The example calculations presented here that use these approximate boundary conditions are, therefore, accurate, but do not attain the maximum possible accuracy.

5. OUTPUT AND VISUALIZATION

The vast majority of finite element visualization software only display elements with linear shape functions. To be able to view the results of the computations, interpolation elements are defined that interpolate the output from the generalized elements (Figure 3). The interpolation elements are hexahedral or quadrilateral. For the case of subdivision surfaces with triangular topology, it is natural to make use of the interpolation triangle, which may be obtained by degenerating the quadrilateral. To obtain quality images from simulations using higher-order generalized elements, each generalized element should be subdivided into several multi-linear interpolation elements.

The state variables (e.g. the stress), generically denoted ϕ, are constant within the interpolation element. They are evaluated as a linear combination of the values at the integration points in the generalized elements,

$$\bar{\phi}^I = \sum_G \alpha^G \phi^G, \tag{42}$$

where the bar and superscript I indicate the mean value of ϕ in the interpolation element, α^G is the weighting coefficient, and the superscript G refers to the integration point in the generalized element.

An approximate L_2 projection was used to calculate the α^G for the example calculations presented later. Starting with the discrete approximation of the L_2 projection,

$$\sum_{B,G} N_A(\boldsymbol{s}^G) N_B(\boldsymbol{s}^G) w^G \det(\boldsymbol{J}^G) \phi_B = \sum_G N_A(\boldsymbol{s}^G) w^G \det(\boldsymbol{J}^G) \phi^G \quad \forall A, \tag{43}$$

the coefficient matrix on the left-hand side is replaced with its diagonal row-summed approximation,

$$\sum_G N_A(\boldsymbol{s}^G) w^G \det(\boldsymbol{J}^G) \phi_A = \sum_G N_A(\boldsymbol{s}^G) w^G \det(\boldsymbol{J}^G) \phi^G, \tag{44}$$

giving an easily evaluated expression for the value ϕ_A associated with the basis function N_A,

$$\phi_A = \sum_G W_A^G \phi^G, \tag{45}$$

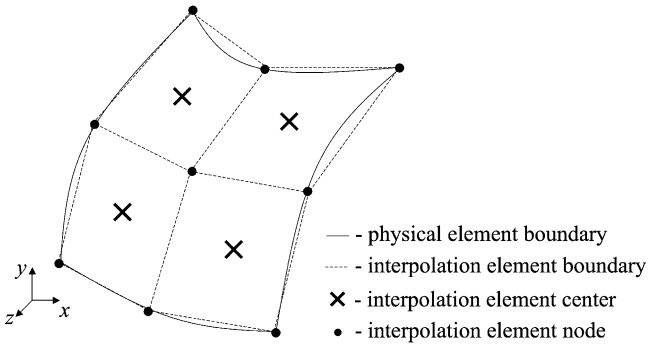

Figure 3. Interpolation element concept illustrated using a shell coming from rectangular topology.

$$W_A^G = \frac{N_A(\boldsymbol{s}^G) w^G \det(\boldsymbol{J}^G)}{\sum_{\hat{G}} N_A(\boldsymbol{s}^{\hat{G}}) w^G \det(\boldsymbol{J}^{\hat{G}})}. \tag{46}$$

The values of ϕ at the interpolation nodes are interpolated from the nodes of the generalized element,

$$\phi_A^I = \sum_{B,G} N_B(\boldsymbol{s}_A) W_B^G \phi^G. \tag{47}$$

The value of ϕ at the centroid of the interpolation element, which is written to the output database, is averaged from the values at the N interpolation nodes of the interpolation element. The expression for α^G simplifies to

$$\bar{\phi}^I = \frac{1}{N} \sum_{A,B,G} N_B(\boldsymbol{s}_A) W_B^G \phi^G. \tag{48}$$

$$\alpha^G = \frac{1}{N} \sum_{A,B} N_B(\boldsymbol{s}_A) W_B^G. \tag{49}$$

The interpolation element definition, provided in the Appendix, therefore, consists of three parts (1) the interpolation node definitions (that are defined independently of the interpolation element), (2) the interpolation element connectivity specified in terms of the interpolation nodes, and (3) the values of α^G.

6. EXAMPLE CALCULATIONS

The calculations presented here are presented to demonstrate the versatility of the generalized elements. All calculations were performed in double precision on a Dell laptop computer using a single core unless otherwise noted.

6.1. Quadratic Lagrange and isogeometric solid elements

The solid generalized elements are demonstrated by a rectangular bar with a length of 6 cm and a width of 4 cm, which strikes a rigid wall at an initial velocity of 300 m/s. Quarter symmetry is used to reduce the mesh to $1.5 \times 1.5 \times 6.0$ cm. The material is elasto-plastic with linear isotropic hardening. Its initial density is 1.71 g/cc, Young's modulus is 2.2 Mbar, the Poisson ratio is 0.3, the yield stress is 2×10^{-4} Mbar, and the linear plastic hardening modulus is 10^{-3} Mbar. The simulation terminates at 80 ms. The solution was advanced in time using explicit integration with the time step determined by the maximum system eigenvalue. Results using the standard 1-point brick elements, quadratic Lagrange polynomials, and quadratic NURBS are shown in Figure 4 and Table I summarizes the results.

The quadratic Lagrange elements are integrated with $3 \times 3 \times 3$ Gauss quadrature. The same quadrature was used as the 'full' integration rule for the NURBS elements, and the 'reduced' rule is $2 \times 2 \times 2$. As expected, the cost of an element scales in a roughly linear fashion with the number of integration points, and, therefore, the use of reduced integration with the NURBS elements is very worthwhile. It is important to note that none of the NURBS elements exhibited zero energy

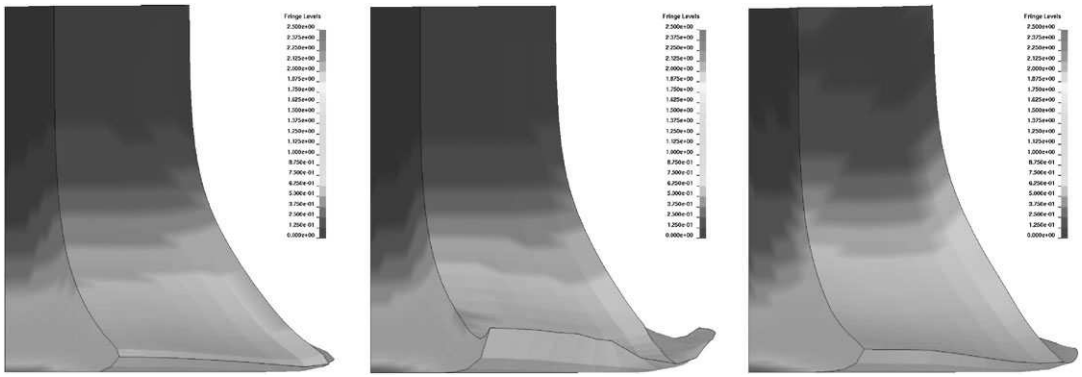

Figure 4. Contours of plastic strain for, from left to right, 2048 standard 1-point elements in LS-DYNA, 256 fully integrated quadratic Lagrange polynomial elements, and the 256 fully integrated quadratic NURBS elements.

modes with reduced integration. The recently proposed integration method [33] for NURBS should show an even greater improvement.

The peak plastic strain occurs at the center of the face of the bar striking the rigid wall (note that calculations were performed taking advantage of symmetry). The peak plastic strains are reported in Table I obtained from the integration points for the standard 1-point elements, and from the plastic strains projected from integration points in the generalized elements onto the interpolation elements. All the projection weights in the NURBS elements are positive and sum to 1.0, and, therefore, the projected values are bounded by the maximum and minimum values from the computational mesh, i.e. the reported maximum plastic strains are guaranteed to be less than, or equal, to the actual integration point values.

Comparing the peak plastic strains for the NURBS elements, it is almost uniformly true that the values for reduced integration are lower than for the cases with full integration. This result is counter-intuitive based on the response of standard finite elements, where reduced integration reduces both volumetric and shear locking, leading to larger deformations and plastic strains. NURBS basis functions do not experience locking to the same degree as Lagrange basis functions, and, therefore, reduced integration increases the deformations very little. Full integration, however, has integration points that are closer to the element boundary than reduced integration. Therefore, full integration has an integration point closer to the actual point of maximum plastic strain than reduced integration, and resolves the peak plastic strain for this particular problem better than reduced integration.

The total CPU times reported in Table I indicate relative trends, however, the performance was not optimized. For example, the rigid wall considers all the nodes in the mesh, including the nodes used in the interpolation elements. There is also a significant overhead associated with processing the interpolation elements and nodes for output. Despite these limitations, the generalized elements are competitive with the production 1-point element in LS-DYNA. Using only 256 quadratic isogeometric elements with reduced integration, the peak plastic strain is 2.42 after 3.9 CPU seconds, whereas the 2048 1-point elements give a peak plastic strain of 2.16 at a cost of 4.6 CPU seconds.

Table I. Bar impact results.

Type	Degree	Integration points	Nodes	Elements	$\bar{\varepsilon}^p_{\max}$	Time steps	CPU seconds
Lagrange	1	1	81	32	1.34628	314	7.325e−2
Lagrange	1	1	425	256	1.8642	872	3.726e−1
Lagrange	1	1	1225	864	2.04989	1500	1.535
Lagrange	1	1	2673	2048	2.16408	2136	4.6027
Lagrange	2	27	81	4	1.6764	609	0.58415
Lagrange	2	27	425	32	1.91551	1293	2.1039
Lagrange	2	27	1225	108	2.20551	2436	10.918
Lagrange	2	27	2673	256	2.34649	3370	35.509
NURBS	2	27	54	4	1.50409	229	0.10276
NURBS	2	27	160	32	1.93467	465	0.6906
NURBS	2	27	648	256	2.47937	954	9.2623
NURBS	2	8	54	4	1.57547	200	7.739e−2
NURBS	2	8	160	32	1.85617	432	0.28568
NURBS	2	8	648	256	2.41749	1051	3.878
NURBS	3	64	160	4	1.84722	304	0.38406
NURBS	3	64	648	32	2.16037	679	3.9282
NURBS	3	64	3400	256	2.40331	1911	80.538
NURBS	3	27	160	4	1.79937	293	0.2098
NURBS	3	27	648	32	2.02539	702	1.8819
NURBS	3	27	3400	256	2.32435	1974	37.472
NURBS	4	125	350	4	2.00626	393	1.5145
NURBS	4	125	1664	32	2.18698	985	18.332
NURBS	4	125	9800	256	2.71286	2546	385.86
NURBS	4	64	350	4	1.86798	380	0.7969
NURBS	4	64	1664	32	2.09956	1015	10.17
NURBS	4	64	9800	256	2.48212	2550	204.8

Conventional wisdom says the stable time step size decreases rapidly with an increase of the degree of the basis function. This is reinforced by the results using the Lagrange polynomial basis functions. For the same number of nodes, the quadratic elements have a time step size roughly half the size of the linear elements. In the case of the NURBS elements, however, the time step size decreases much more slowly as the degree increases.

6.2. T-Spline shell elements

The bumper test case demonstrates the potential of T-splines as a design through analysis technology. T-splines [34] are an important generalization of NURBS that also have many features in common with subdivision surfaces. They permit local refinement [35] and can model geometry of arbitrary topological genus. Importantly, from a design through analysis standpoint, they are analysis suitable from inception. This means that the geometric T-spline basis may be used in an isogeometric analysis setting without modification. Exploratory investigations using T-splines as a basis for isogeometric analysis indicate that the basis performs in a manner analogous to NURBS [4, 3] with fewer degrees of freedom.

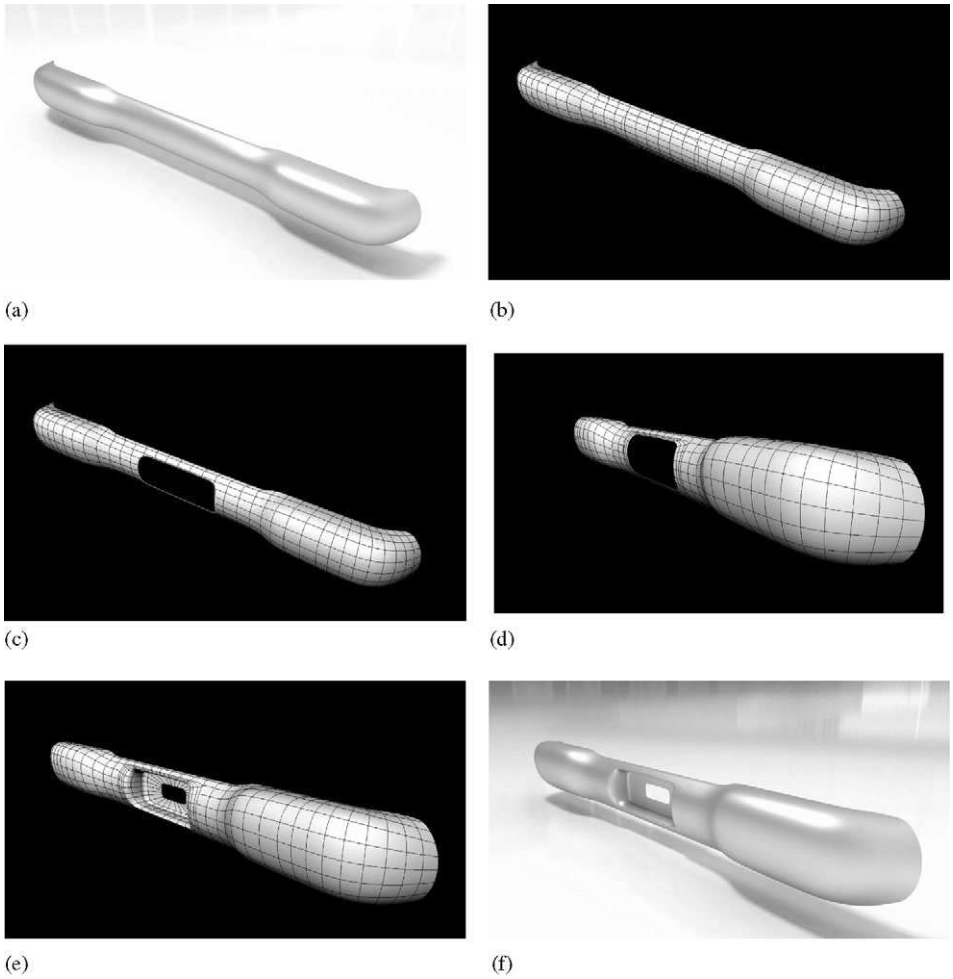

Figure 5. The design of an analysis suitable T-spline bumper model: (a) A cubic NURBS models the general shape of the bumper model; (b) a trimming curve is placed where additional geometric features are desired; and (c) the NURBS geometry upon application of the trimming curve. The application of trimming curves destroys the analysis suitable nature of the geometric NURBS basis; (d) the trimmed NURBS is converted into an untrimmed T-spline. The conversion process generates an unstructured T-mesh that matches the topology of the trimmed NURBS. The T-spline basis remains analysis suitable; (e) the final T-mesh for the bumper model; and (f) A rendering of the final T-spline. Notice the global smoothness of the geometry. The T-spline basis is G^1 continuous in the neighborhood of star points.

The model was generated using the T-spline plug-in for Rhino [36] and the back-end T-Tools libraries [37]. In Figure 5(a), a simple cubic NURBS is used to model the general shape of the bumper. It should be noted that NURBS are a very simple type of T-spline without T-junctions or star points. To add additional topological features to NURBS, it is common to use trimming curves. Figures 5(b) and (c) show the placement of a trimming curve and the resulting geometry once the trim is applied. In a design through analysis environment, the application of trimming curves

Figure 6. A mode of the T-Spline bumper model.

destroys the analysis suitable nature of the geometry. The geometric basis no longer describes the geometry and cannot be used directly in analysis. Using T-splines, however, it is possible to overcome the NURBS trimming problem by converting a trimmed NURBS into an untrimmed, watertight, and analysis suitable T-spline. The details of this process are described in [38]. The conversion process first modifies the topology of the T-mesh to accommodate any trimming curves. A fitting procedure is then used to match the T-spline surface to the trimming curve. Figure 5(d) shows the untrimmed T-spline that matches the original trimmed NURBS upon completion of the conversion process. The T-spline basis in the neighborhood of the star points is G^1 continuous. Everywhere else the T-spline basis is C^2 continuous as you would expect for a cubic T-spline. This untrimmed T-spline is now analysis suitable. Additional modeling generates the final bumper T-mesh and geometry shown in Figures 5(e) and (f).

The final model of the bumper consists of 876 cubic generalized shell elements with 705 control points. Assuming $E = 10^7$, $\nu = 0.3$, and the thickness is 1.0, the free–free eigenvalues were calculated. A mode, chosen primarily for its shape, is shown in Figure 6.

6.3. Isogeometric extended finite elements for linear fracture analysis

This problem demonstrates that the generalized element formulation is not restricted to spline-based analysis. All the details associated with the element implementation have been presented in previous sections. Additional results regarding the convergence rate and accuracy of the combination of the XFEM with isogeometric analysis will be presented in a future paper.

A linear fracture problem, using the extended finite element enrichment functions from Equation (12), consists of a square domain 6 in on a side. A crack runs from the left edge to the center of the square. The material is linearly elastic with a modulus of elasticity of 10^7 psi and a Poisson ratio of 0.3. Fifth degree NURBS basis functions were used to solve it on meshes ranging from 3×3 to 11×11 elements, with the center element containing the crack tip. Since the problem is two-dimensional, linear basis functions are used in the z direction. Gaussian quadrature with 10 points in each direction was used to evaluate the integrals. This integration rule is not optimal, but it was chosen to accurately integrate the products of the fifth degree NURBS polynomials with the high gradient crack tip enrichment function without introducing special purpose integration rules. Displacement boundary conditions were imposed corresponding to an exact solution of $K_I = 100$.

Since the NURBS basis functions are not exactly interpolatory on the boundary, specifying their displacement as the analytical value at their spatial coordinates does not provide an exact displacement boundary condition. Two different boundary condition procedures were tested to study their effect on the convergence rate of K_I. In the first approach, the displacement boundary conditions were directly applied to the control points. In the second approach, the displacement of the control points were chosen so that the prescribed displacement boundary conditions are met in a least-square sense using the objective function defined in Equation (39). As shown in Figure 7,

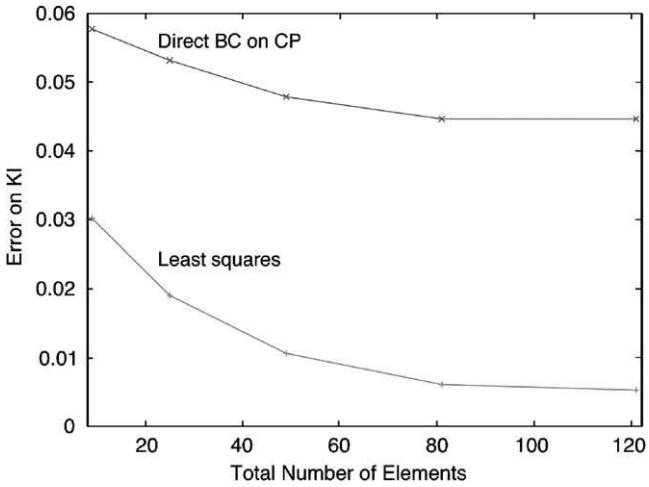

Figure 7. The error in the calculated value of K_I as a function of the number of elements and the method for imposing the boundary conditions.

the least-squares procedure is the most accurate, but in many engineering analyses, the simpler one may be more than adequate. On the finest mesh with the least squares boundary conditions, a value of $K_I = 99.49$ was obtained and the x and y strains are shown in Figure 8.

7. CONCLUSIONS

A generalized element formulation for solid and shell elements that separates the definition of the basis functions and the integration rule from the analysis was introduced. This separation facilitates (1) the development of a direct design-to-analysis capability, (2) rapid prototyping of elements for new classes of basis functions, and (3) teaching the finite element method without programming. While, perhaps, not optimal in either memory usage or speed, this approach provides a means of accessing the many advanced capabilities of robust, mature production codes for research, and state-of-the-art analyses.

Example calculations for Lagrange interpolation functions, NURBS, T-Splines, and XFEM demonstrated the versatility of the formulation. Other types of basis functions can obviously be used. The elements are defined entirely through the input file and no additional programming is required once the generalized element formulation is implemented. Any convenient means, e.g. MATLAB, may be used to calculate the required data.

Excellent accuracy was readily achieved by combining the XFEM approach to linear fracture analysis with the higher-order NURBS basis functions of isogeometric analysis. This was demonstrated with the generalized elements and did not require any finite element programming.

The feasibility of CAD programs to directly generate models suitable for analysis without an intermediate mesh generation step was demonstrated using T-Splines with the CAD program Rhino. Research on CAD basis functions are, therefore, immediately available for use in finite element analysis.

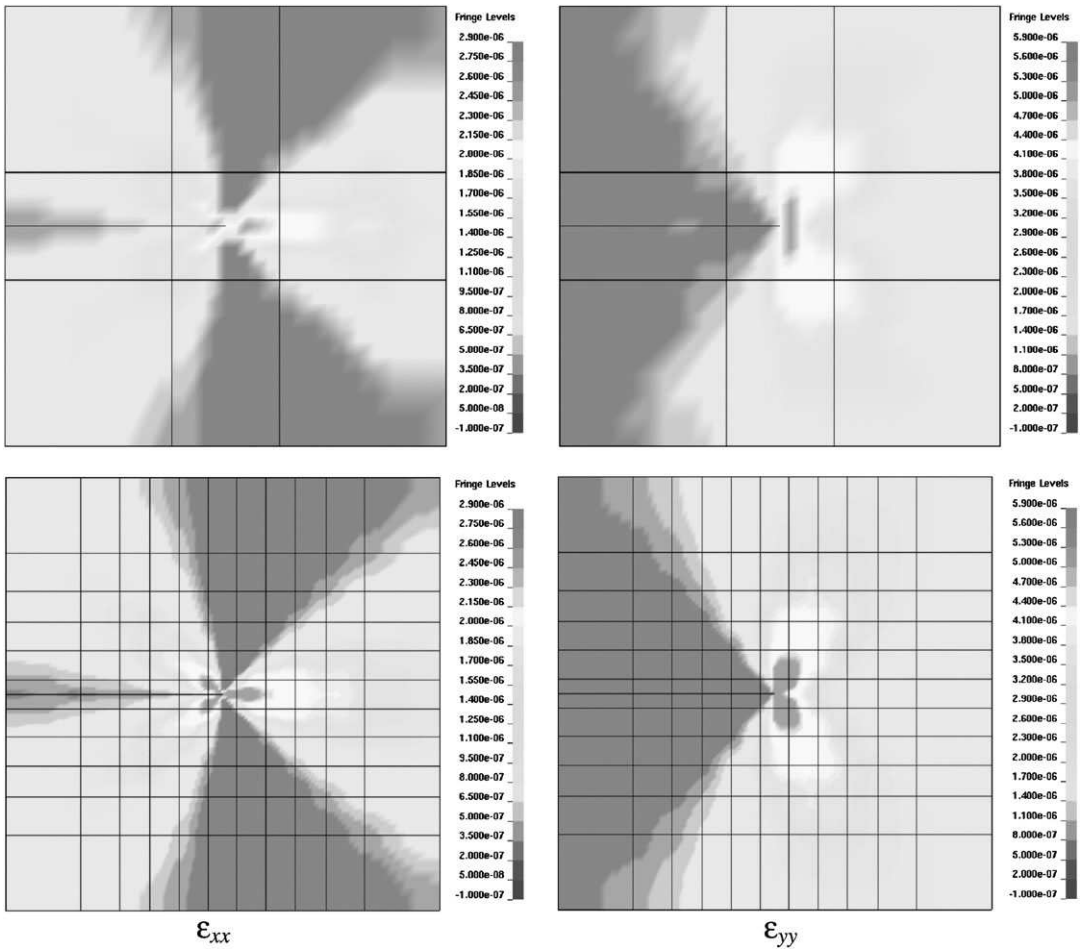

Figure 8. The x and y strains calculated using the XFEM with fifth degree NURBS basis functions. The top and bottom rows show the results calculated with 3×3 and 11×11 element meshes, respectively.

The approach is readily extensible—adding a generalized beam element is obvious. Adding meta operations on the generalized elements, such as generalized projection methods (e.g. $\bar{\boldsymbol{B}}$ or $\bar{\boldsymbol{F}}$ [32]), is also feasible. There is no restriction of the generalized element method to solid mechanics. Stabilized methods for computational fluid dynamics are a natural extension of the current work.

ACKNOWLEDGEMENTS

This research was supported by NSF grant 0700204 (Dr Joycelyn S. Harrison, program officer) and ONR grants N00014-08-1-0992 and N00014-08-1-1191 (Dr Luise Couchman, contract monitor).

REFERENCES

1. Hughes TJR, Cottrell J, Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering* 2005; **194**:4135–4195.
2. Cottrell J, Reali A, Bazilevs Y, Hughes TJR. Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**:5257–5297.
3. Dörfel M, Jüttler B, Simeon B. Adaptive isogeometric analysis by local h-refinement with T-splines. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**:264–275.
4. Bazilevs Y, Calo V, Cottrell J, Evans J, Hughes TJR, Lipton S, Scott M, Sederberg T. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**:229–263.
5. Cirak F, Ortiz M, Schröder P. Subdivision surfaces: a new paradigm for thin shell analysis. *International Journal for Numerical Methods in Engineering* 2000; **47**:2039–2072.
6. Cirak F, Ortiz M. Fully C^1-conforming subdivision elements for finite deformation thin shell analysis. *International Journal for Numerical Methods in Engineering* 2001; **51**:813–833.
7. Cirak F, Scott MJ, Antonsson EK, Ortiz M, Schröder P. Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision. *Computer-Aided Design* 2002; **34**:137–148.
8. Bazilevs Y, Calo V, Zhang Y, Hughes TJR. Isogeometric fluid–structure interaction analysis with applications to arterial blood flow. *Computational Mechanics* 2006; **38**:310–322.
9. Bazilevs Y, Calo V, Cottrel J, Hughes TJR, Reali A, Scovazzi G. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. *Computer Methods in Applied Mechanics and Engineering* 2007; **197**:173–201.
10. Akkerman I, Bazilevs Y, Calo V, Hughes TJR, Hulshoff S. The role of continuity in residual-based variational multiscale modeling of turbulence. *Computational Mechanics* 2008; **41**:371–378.
11. Bazilevs Y, Hughes TJR. NURBS-based isogeometric analysis for the computation of flows about rotating components. *Computational Mechanics* 2008; **43**:143–150.
12. Bazilevs Y, Calo V, Hughes TJR, Zhang Y. Isogeometric fluid-structure interaction: theory, algorithms, and computations. *Computational Mechanics* 2008; **43**:3–37.
13. Wall WA, Frenzel MA, Cyron C. Isogeometric structural shape optimization. *Computer Methods in Applied Mechanics and Engineering* 2008; **197**:2976–2988.
14. Hughes TJR, Reali A, Sangalli G. Duality and unified analysis of discrete approximations in structural dynamics and wave propagation: Comparison of p-method finite elements with k-method nurbs. *Computer Methods in Applied Mechanics and Engineering* 2008; **197**:4104–4124.
15. Lipton S, Evans J, Bazilevs Y, Elguedj T, Hughes TJR. Robustness of isogeometric structural discretizations under severe mesh distortion. *Computer Methods in Applied Mechanics and Engineering* 2009; DOI: 10.1016/j.cma.2009.01.022.
16. Belytschko T, Black T. Elastic crack growth in finite elements with minimal remeshing. *International Journal for Numerical Methods in Engineering* 1999; **45**(5):603–620.
17. Moes N, Dolbow J, Belytschko T. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering* 1999; **46**(1):131–150.
18. Sukumar N, Moes N, Moran B, Belytschko T. Extended finite element method for three-dimensional crack modelling. *International Journal for Numerical Methods in Engineering* 2000; **48**(11):1549–1570.
19. Ventural G, Gracie R, Belytschko T. Fast integration and weight function blending in the extended finite element method. *International Journal for Numerical Methods in Engineering* 2009; **77**:1–29.
20. Vitali E, Benson DJ. An extended finite element formulation for contact in multi-material arbitrary Lagrangian–Eulerian calculations. *International Journal for Numerical Methods in Engineering* 2006; **67**(10):1420–1444.
21. Vitali E, Benson DJ. Contact with friction in multi-material arbitrary Lagrangian–Eulerian formulations using X-FEM. *International Journal for Numerical Methods in Engineering* 2008; **76**(6):893–921.
22. Hallquist JO. LS-DYNA theoretical manual. *Technical Report*, Livermore Software Technology Corporation, 1998.
23. Benson DJ, Bazilevs Y, Hsu MC, Hughes TJR. Isogeometric shell analysis: The Reissner–Mindlin shell. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**:276–289. Available at: http://www.sciencedirect.com/science/article/B6V29-4WC1126-1/2/b073c14a2534022b5eda8399974d6164.
24. Piegl L, Tiller W. *The NURBS Book* (*Monographs in Visual Communication*) (2nd edn). Springer: New York, 1997.
25. Hughes TJR. *The Finite Element Method*, *Linear Static and Dynamic Finite Element Analysis*. Dover: New York, 2000.

26. Benson DJ. Stable time step estimation for multi-material Eulerian hydrocodes. *Computer Methods in Applied Mechanics and Engineering* 1998; **167**:191–205.
27. Hallquist JO. LS-DYNA keyword user's manual. *Technical Report*, Livermore Software Technology Corporation, 2007.
28. Bischoff M, Wall MA, Bletzinger K, Ramm E. Models and finite elements for thin-walled structures. In *Encyclopedia of Computational Mechanics*, *Solids*, *Structures and Coupled Problems*, Stein E, de Borst R, Hughes TJR (eds), vol. 2, Chapter 3. Wiley: New York, 2004.
29. Hughes TJR, Liu W. Non-linear finite element analysis of shells: Part I. Three-dimensional shells. *Computer Methods in Applied Mechanics and Engineering* 1981; **26**:331–362.
30. Belytschko T, Lin JI, Tsay CS. Explicit algorithms for the non-linear dynamics of shells. *Computer Methods in Applied Mechanics and Engineering* 1984; **42**:225–251.
31. Belytschko T, Liu WK, Moran B. *Non-linear Finite Elements for Continua and Structures*. Wiley: West Sussex, England, 2001.
32. Elguedj T, Bazilevs Y, Calo V, Hughes TJR. $\bar{B}$ and $\bar{F}$ projection methods for nearly incompressible linear and non-linear elasticity and plasticity using higher-order NURBS elements. *Computer Methods in Applied Mechanics and Engineering* 2008; **197**:2732–2762.
33. Hughes TJR, Reali A, Sangalli G. Efficient quadrature for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**:301–313. Available at: http://www.sciencedirect.com/science/article/B6V29-4V75YNJ-1/2a88a102a5291e66436f127c234077e1e.
34. Sederberg T, Zheng J, Bakenov A, Nasri A. T-splines and T-NURCCSs. *ACM Transactions on Graphics* 2003; **22**(3):477–484.
35. Sederberg T, Cardon D, Finnigan G, North N, Zheng J, Lyche T. T-spline simplification and local refinement. *ACM Transactions on Graphics* 2004; **23**(3):276–283.
36. T-Splines, Inc. http://www.tsplines.com/products/tsplines-for-rhino.html.
37. T-Splines, Inc. http://www.tsplines.com/products/t-tools-libraries.html.
38. Sederberg T, Finnigan G, Li X, Lin H. Watertight trimmed NURBS. *ACM Transactions on Graphics* 2008; **27**(3).