Octasection-based Refinement of Finite Element Approximations on Tetrahedral Meshes that Guarantees Shape Quality

Lance Endres, Petr Krysl[1,*]

[1] *University of California, San Diego,*
Structural Engineering, Mail code 0085,
9500 Gilman Dr, La Jolla, CA 92093-0085,
pkrysl@ucsd.edu

KEY WORDS: Finite element method, mesh refinement, octasection, tetrahedron, hierarchical adaptive approximation

SUMMARY

Adaptive refinement of finite element approximations on tetrahedral meshes is generally considered to be a non-trivial task. (We wish to stress that this paper deals with mesh *refinement* as opposed to *remeshing*.) The splitting individual finite elements needs to be done with much care to prevent significant deterioration of the shape quality of the elements of the refined meshes. Considerable complexity thus results, which makes it difficult to design (and even more importantly, to later maintain) adaptive tetrahedra-based simulation codes. An adaptive refinement methodology, dubbed CHARMS (Conforming Hierarchical Adaptive Refinement MethodS), had recently been proposed by Krysl, Grinspun, and Schröder. The methodology streamlines and simplifies mesh refinement, since conforming (compatible) meshes always result by construction. The present work capitalizes on these conceptual developments to build a mesh refinement technique for tetrahedra. Shape quality is guaranteed for an arbitrary number of refinement levels due to our use of element octasection based on the Kuhn triangulation of the cube. Algorithms and design issues related to the inclusion of the present technique in a CHARMS-based object oriented software framework (`http://hogwarts.ucsd.edu/~pkrysl/CHARMS`) are described.

INTRODUCTION

The generation of locally adapted conforming finite element meshes is a basic step in the construction of finite element approximations to partial differential equations. We would like to point the reader to the article by Arnold, Mukherjee, and Pouly [1] for an extensive literature review of the subject of the state-of-the-art local refinement of tetrahedral meshes.

Often the computational meshes result from the refinement of an initial coarse mesh. State of the art refinement techniques proceed by selecting individual finite elements for refinement, and bisecting or octasecting them. Since the mesh at that point is no longer conforming, special steps need to be taken to restore the conformity. As an example of a successful approach consider, for instance, the algorithm by Arnold et al.[1] which proposes to select additional elements for refinement and proceeds by bisection. This algorithm provably terminates. It also satisfies an additional requirement: the quality measure of the resulting tetrahedra does not degenerate irrespectively of the number of refinement levels.

Arnold's algorithm is relatively simple (compared to previously published approaches, for instance Reference [4]), but it is still complex compared to the algorithm we present here. Moreover, the present approach is an instance of a general methodology which has been shown to apply equally well in one, two, and three dimensions, and for large number of finite element types (triangles, quadrilaterals, hexahedra [2], or subdivision surfaces [2]).

1. ADAPTIVE BASIS FUNCTION REFINEMENT: CHARMS

We shall briefly describe the CHARMS (Conforming Hierarchical Adaptive Refinement MethodS) mesh refinement methodology. The reader is referred to Reference [3] for details.

We start by considering a set of linearly independent scalar basis functions $\mathcal{B} = \{\phi_i(x)\}$, supported on the finite elements in the mesh M, which span the finite element space $\mathcal{X}$,

$$\mathcal{X} = \left\{ v(x) : v(x) = \sum_i \phi_i(x) v_i \right\} , \tag{1}$$

with $x \in \mathbb{R}^n$, $v(x) \in \mathbb{R}^m$, $v_i \in \mathbb{R}^m$ ($n \geq 1$, $m \geq 1$), and $\phi_i(x) \in \mathbb{R}$. We propose to create a new, "refined" mesh, M', such that the set of functions $\phi'_i(x)$ constructed on mesh M' constitutes a basis for the *finer* space $\mathcal{X}'$, such that $\mathcal{X}'$ is related to the *coarser* space $\mathcal{X}$ by inclusion:

$$\mathcal{X} \subset \mathcal{X}' . \tag{2}$$

We shall assume that given a coarser mesh M with the associated basis functions $\phi_i(x)$ it is possible to construct a finer mesh M', and the associated basis functions $\phi'_i(x)$, such that the nesting relation (2) holds. By recursively extending the above argument, a *hierarchy* of approximation spaces $\mathcal{X}^{(j)}$ is obtained. The coarsest space in the hierarchy is $\mathcal{X}^{(0)}$, progressively finer spaces have increasing indices $\mathcal{X}^{(1)}$, $\mathcal{X}^{(2)}\ldots$, and this sequence may be either infinite or finite, resulting in the hierarchical nesting relation

$$\mathcal{X}^{(0)} \subset \mathcal{X}^{(1)} \subset \mathcal{X}^{(2)} \subset \ldots \subset \mathcal{X}^{(m)} ; \tag{3}$$

1.1. REFINEMENT EQUATION

The relation (3) makes it possible to apply the *refinement* (also known as the *multiresolution* or *dilation*) equation [7] to finite element mesh refinement. Since $\mathcal{X}^{(j)} \subset \mathcal{X}^{(j+1)}$, any basis function $\phi_i^{(j)}(x)$ on level j may be exactly resolved (reproduced) in the basis with finer resolution $\{\phi_i^{(j+1)}(x)\}$:

$$\phi_i^{(j)}(x) = \sum_k \beta_{ik}^{(j+1)} \phi_k^{(j+1)}(x) , \tag{4}$$

where $\beta_{ik}^{(j+1)}$ are the coefficients of the linear combination. We will usually attempt to construct the spaces $\mathcal{X}^{(j)}$ in such a way that the sum in (4) has a finite, and fairly small, number of terms.

The refinement equation (4) is the key to our adaptivity approach: Given an approximation space $\mathcal{X}^{(j)}$ with basis $\mathcal{B}^{(j)}$, one can derive a custom space $\mathcal{X}$, with basis $\mathcal{B}$ whose resolution is locally finer, by replacing one or more parent basis functions from $\mathcal{B}^{(j)}$ by children functions from $\mathcal{B}^{(j')}$ on some levels $j' > j$. In this sense, the refinement equation may be viewed as two statements: Firstly, the *approximation properties* of the basis function set $\mathcal{B}$ preserve or enhance those of $\mathcal{B}^{(j)}$ if the children basis functions are substituted for the parent basis function. Secondly, the *continuity* of the children functions is greater than or equal to the continuity of the parent function (since any finite linear combination of C^k functions is C^k). Current mesh refinement techniques have to wrestle with continuity because the refinement operations are applied to isolated *pieces* of basis functions; CHARMS, on the other hand, achieves compatibility by construction.

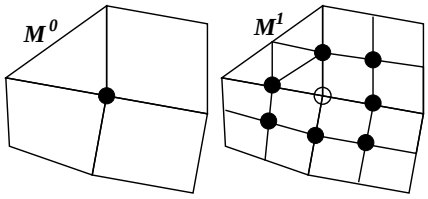

Figure 1. Illustration of Equation (5).

1.2. TRUE HIERARCHICAL BASIS

Equation (4) may be viewed also as a statement of equivalence: the left hand side (the coarse function) is equivalent to the right hand side (the set of finer functions) in the sense that if we combine both sides of (4) we get a linearly-dependent set. Furthermore, we may note that there is just one function too many in the combined set of the left and right hand side functions. Therefore, we could move all but one of the right hand side (finer) functions to the left hand side, obtaining

$$\phi_i^{(j)}(x) - \sum_{k \neq m} \beta_{ik}^{(j+1)} \phi_k^{(j+1)}(x) = \beta_{im}^{(j+1)} \phi_m^{(j+1)}(x) \ . \tag{5}$$

That is a statement of the same kind as before, an equivalence, but now the function that is being "reproduced" is on the finer level. Symmetry considerations dictate that the function to leave on the right hand side is the one whose node "stems" from the node of the coarse function. For instance, we show in Figure 1 the patch (one triangle and three quadrilaterals) that supports the function indicated by the filled circle (left). In the right hand part we show the refined patch, where the filled circles indicate functions on level $j+1$ on the left hand side of (5), and the empty circle stands for the single function on the right hand side ($\phi_m^{(j+1)}(x)$) (in other words, the filled circles indicate the linearly independent functions, the empty circle the one finer function that is dependent on the rest). We shall call the functions on level $j+1$ that were moved to the left hand side of (5) the ***detail functions*** of $\phi_i^{(j)}(x)$.

The above discussion may suggest to the reader a way of constructing the refined approximation spaces: To refine a function from the coarse space $\mathcal{B}$, we shall add its detail functions to $\mathcal{B}$ to obtain $\mathcal{B}'$. The resulting approximation will be dubbed the ***true hierarchical*** basis, and, indeed it is not a new concept in itself, since we get the well-known hierarchical basis discussed, for example, by Yserentant [9]. However, our use of this concept for selective refinement is novel.

1.3. QUASI-HIERARCHICAL BASIS

To formalize somewhat, we shall use the term ***active function*** for functions selected from a given basis set, and the symbol $\widehat{\mathcal{B}}^{(k)}$ will be used for the set of active functions from basis set $\mathcal{B}^{(k)}$, where $\mathcal{B}^{(k)}$ is the set of all basis functions on the mesh at level k. Now, the approximation basis function set $\mathcal{B}$ may be written as

$$\mathcal{B} = \bigcup_{j=0}^{\infty} \widehat{\mathcal{B}}^{(j)} \ , \tag{6}$$

that is as a union of active sets from all mesh levels. For the true hierarchical basis described above, the active set on level $j = 0$ includes all the basis functions defined on the initial mesh, and each active set on level $j > 0$ includes only the detail functions. Now we shall describe an alternative refinement strategy, the quasi-hierarchical basis.

As before we proceed from the refinement equation (4). In difference to the true hierarchical approximation, we shall interpret the refinement equation as a recipe for *replacing* coarse functions (left hand side) by "finer" (higher-resolution) functions (right hand side). Thus, the refinement will delete

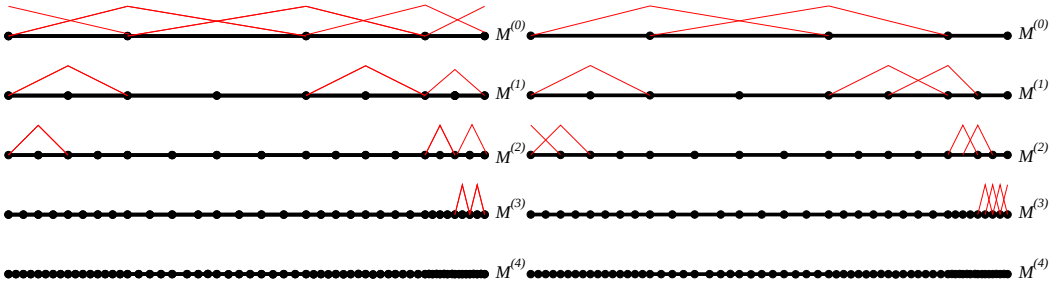

Figure 2. Comparison of the true hierarchical basis (left) with the quasi-hierarchical basis (right).

(deactivate) coarse functions, replacing them in a lossless manner by finer functions, and unrefinement will delete (deactivate) fine functions and activate coarse functions. Clearly, if this type of refinement is applied globally, all coarse functions are replaced by fine functions and the common form of finite element approximation is recovered, i.e. an interpolating partition of unity is obtained; on the other hand, if the refinement is graded, transition needs to be made from coarse to fine functions, and in the transition regions the resulting approximation resembles the true hierarchical basis in that the basis function do not necessarily add up to unity. This is illustrated in Figure 2, where on the left we show the true hierarchical basis, compared to the quasi-hierarchical basis on the right. Even though we replace coarse functions instead of augmenting them with finer functions, the resulting approximation basis is still written in the form of Equation (6), the only change being that the active sets $\widehat{\mathcal{B}}^{(j)}$ may now include not only the detail functions, but all the functions from the refinement set of the coarse function.

1.4. APPROXIMATION ON CHARMS-REFINED MESHES

Both the quasi-hierarchical and the true hierarchical approximation may be expressed in exactly the same way, and in fact often it is possible to devise refinement sequences that yield precisely the same spans for both basis types. This is not surprising, since both basis types follow from the refinement equation, and as shown in the next section, the construction of both types of bases may be expressed as the activation or deactivation of basis functions from the conceptual hierarchy of nested meshes. The equivalence of the quasi-hierarchical and the hierarchical basis allows us to write the finite element approximation in the unified way

$$u_h(x) = \sum_{j:\widehat{\mathcal{B}}^{(j)}\neq\emptyset} \left(\sum_{i:\phi_i^{(j)}(x)\in\widehat{\mathcal{B}}^{(j)}} \phi_i^{(j)} u_i^{(j)} \right) , \tag{7}$$

where $\widehat{\mathcal{B}}^{(j)}$ is the set of active functions on level j, and $u_i^{(j)}$ are the nodal parameters. Evidently, it would be possible to express the finite element approximation with a single sum, as usual, but the above form makes it clear that the basis functions "live" on different refinement levels. It remains to formulate algorithms for the construction of the active sets, and that is the goal of the next section.

2. FORMAL STATEMENT OF THE REFINEMENT AND UNREFINEMENT ALGORITHMS

It is of advantage in applications if the active functions constitute a *basis*, i.e. if they are linearly independent. In order to make the present work self-contained, we re-phrase here the algorithms

enunciated in Reference [3], but without proofs. These algorithms activate and deactivate functions from the nesting hierarchy, and guarantee the linear independence of the active function set.

For the finite element meshes we consider in this paper, no basis function support is entirely enclosed by the support of another basis function, i.e.,

$$\text{supp}\left[\phi_k^{(j)}\right] \not\subseteq \text{supp}\left[\phi_i^{(j)}\right] \quad \text{for} \quad k \neq i \text{ and } \forall j \geq 0 \ . \tag{8}$$

This is an important tool in our proofs [3]: Consider two functions on the same level; because of the above property, the refinement set of either must contain at least one function not present in the refinement set of the other.

There are many possible algorithms for building adapted bases, and here we present algorithms based upon three **rules**:

1. The refining/unrefining of a function on level j may affect that function or any of its children on level $j+1$; no other function may be involved.
2. A function on level $j+1$ ($j+1 \geq 1$) may be refined only when all its parents on level j have been refined.
3. A function on level j may be unrefined only if (a) it was previously refined and (b) all its children on level $j+1$ are not refined.

If the basis functions are completely supported by one ring of elements around a node, then rules 2 and 3 enforce the common rule of one-level-difference refinement of neighbors; this rule has been applied to finite element meshes, as well as to spatial datastructures such as quadtrees and octrees in various contexts (graphics, mesh generation, spatial searches, etc.) [8, 6].

We now show that if the (un)refinement is applied atomically, i.e. either it is entirely executed or not at all, then linear independence of the adapted basis is guaranteed. Furthermore, our algorithms ensure that the refinement step is *lossless*; the span of the resulting set includes the span of the original set. (Unrefinement is not lossless, in general: some information is always going to be lost, since the goal of unrefinement is to decrease the span of the approximation space.)

2.1. QUASI-HIERARCHICAL BASIS

Let us begin by exploring the quasi-hierarchical refinement strategy. We first describe the refinement operation, and show that it preserves the linear independence requirement and is lossless; we then describe the unrefinement operation, and show that it too preserves the linear independence requirement.

2.1.1. Refinement The ***refinement set*** of the coarser function $\phi_i^{(j)}$ supported by the mesh on level $j+1$ is denoted by $\mathcal{R}^{(j+1)}[\phi_i^{(j)}]$, and contains exactly those basis functions that contribute to the right-hand side of the refinement equation with a non-zero coefficient:

$$\mathcal{R}^{(j+1)}[\phi_i^{(j)}] = \left\{ \phi_k^{(j+1)} \mid \beta_{ik}^{(j+1)} \neq 0 \right\} \ . \tag{9}$$

If $\phi_k^{(j+1)}$ belongs to $\mathcal{R}^{(j+1)}[\phi_i^{(j)}]$, we say that "$\phi_i^{(j)}$ *is a* ***parent*** *of* $\phi_k^{(j+1)}$," and "$\phi_k^{(j+1)}$ *is a* ***child*** *of* $\phi_i^{(j)}$."

Given an initial basis function set, $\mathcal{B}$, which contains $\phi_i^{(j)}$, and satisfies the linear independence requirement, we choose to produce another function set $\mathcal{B}'$ by deactivating $\phi_i^{(j)}$ and activating all of its children $\mathcal{R}^{(j+1)}[\phi_i^{(j)}]$. We refer to this algorithm, which maps $\mathcal{B}$ to $\mathcal{B}'$, as *quasi-hierarchical refinement.*

We claim that quasi-hierarchical refinement (a) preserves the linear independence requirement and (b) is lossless. See Reference [3] for proofs.

2.1.2. Unrefinement Given an initial basis function set, $\mathcal{B}$, that satisfies the linear independence requirement, and given a previously refined $\phi_i^{(j)}$, not in $\mathcal{B}$ and eligible for unrefinement under rule 3, we choose to produce another function set $\mathcal{B}'$ by activating $\phi_i^{(j)}$ and deactivating those children of $\phi_i^{(j)}$ which have no other currently refined (i.e. inactive) parent. More concisely, the members of the following set are deactivated:

$$\left\{ \phi_m^{(j+1)} \in \mathcal{R}^{(j+1)}[\phi_i^{(j)}] \quad \wedge \quad \forall\, r \neq i \;\; \phi_m^{(j+1)} \in \mathcal{R}^{(j+1)}[\phi_r^{(j)}] \rightarrow \phi_r^{(j)} \in \widehat{\mathcal{B}}^{(j)} \right\} \; . \tag{10}$$

We refer to this algorithm, which maps $\mathcal{B}$ to $\mathcal{B}'$, as *quasi-hierarchical unrefinement.* We claim that quasi-hierarchical unrefinement preserves the linear independence requirement. (See Reference [3] for proofs.)

2.2. HIERARCHICAL BASIS

We have completed our overview of quasi-hierarchical refinement, which treats refinement as the *replacement* of coarse-level functions by finer-level functions. Let us turn to an alternative strategy for constructing adapted bases: hierarchical refinement treats refinement as the *addition* of finer-level "detail functions" to an unchanged set of coarse-level functions. Before we continue, let us formalize the concepts of a detail function and a detail (function) set that had been introduced above.

Definition: Given a function $\phi_i^{(j)}$, construct the set of all functions $\phi_k^{(j+1)} \in \mathcal{R}^{(j+1)}[\phi_i^{(j)}]$ such that they vanish at the location x_i of node i

$$\mathcal{D}^{(j+1)}[\phi_i^{(j)}] = \left\{ \phi_k^{(j+1)} \mid \phi_k^{(j+1)} \in \mathcal{R}^{(j+1)}[\phi_i^{(j)}] \text{ and } \phi_k^{(j+1)}(x_i) = 0 \right\} \; . \tag{11}$$

The set $\mathcal{D}^{(j+1)}[\phi_i^{(j)}]$ is the ***detail set*** of $\phi_i^{(j)}$. Functions that belong to at least one detail set are called ***detail functions***.

2.2.1. Refinement Given an initial basis function set, $\mathcal{B}$, which contains $\phi_i^{(j)}$, and satisfies the linear independence requirement, we choose to produce a refined set $\mathcal{B}'$ by activating $\mathcal{D}^{(j+1)}[\phi_i^{(j)}]$. We refer to this algorithm, which maps $\mathcal{B}$ to $\mathcal{B}'$, as *hierarchical refinement.* As proved in Reference [3], the hierarchical refinement (a) preserves the linear independence requirement and (b) is lossless.

2.2.2. Unrefinement Given an initial basis function set, $\mathcal{B}$, that satisfies the linear independence requirement, and given a previously refined $\phi_i^{(j)}$, also in $\mathcal{B}$ and eligible for unrefinement (rule 3), we choose to produce another function set $\mathcal{B}'$ by deactivating functions $\phi_m^{(j+1)} \in \mathcal{D}^{(j+1)}[\phi_i^{(j)}]$ that are used to refine only the function $\phi_i^{(j)}$, and not any other function. We refer to this algorithm, which maps $\mathcal{B}$ to $\mathcal{B}'$, as *hierarchical unrefinement.* It is easy to show that hierarchical unrefinement preserves the linear independence requirement [3].

3. DIVISION OF TETRAHEDRA

As explained in the previous section, CHARMS proceed by constructing basis functions with increased spatial resolution and then combining the finer functions with the coarse functions or replacing the coarse functions with finer functions. In order to support the finer functions, CHARMS needs to divide a coarse tetrahedron into smaller tetrahedra.

It is possible to proceed with bisection or octasection. The crucial aspect to observe are the shape measures of the tetrahedra as the number of refinement levels goes to infinity. We assume that the initial mesh consists of elements with finite quality measure

$$\sigma_T = \frac{h_T}{\rho_T}, \quad \forall T \tag{12}$$

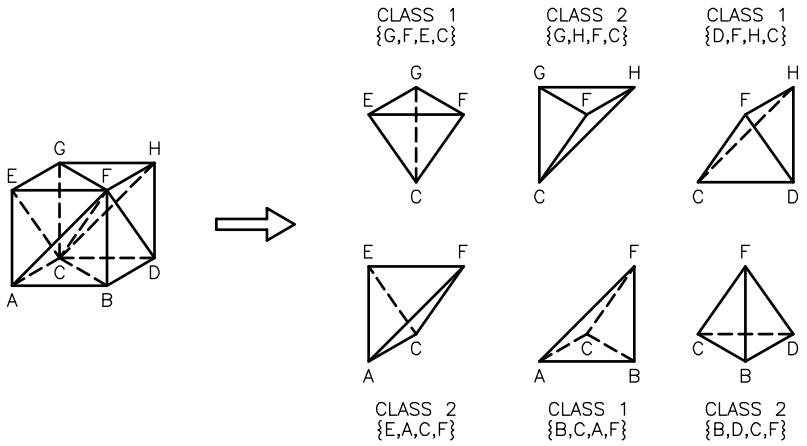

Figure 3. Kuhn's triangulation of a cube.

where h_T is the diameter of the tetrahedron T, and ρ_T is the radius of the inscribed sphere, and we wish to ensure that if tetrahedron $T_i^{(j)}$ at refinement level $j > 0$ results from division of a tetrahedron T on refinement level 0, the quality measure $\sigma_{T_i^{(j)}} < \infty$. In fact, we would like to achieve $\sigma_{T_i^{(j)}} < C\sigma_T$, where $C = O(1)$.

For our purpose bisection has some undesirable properties. In particular,

- the bisection pattern on a face shared by two tetrahedra is not unique: three cuts are possible. That would make it a necessity to communicate the refinement information among neighbors. Moreover,
- bisection has the potential to severely degrade the shape quality measure of the finer functions since the bisected element is not similar to its offspring.

Octasection avoids the problems with the need to communicate refinement on the shared faces, but the issue with the degrading shape quality is something that currently known approaches struggle with. In this section we wish to demonstrate how CHARMS circumvent this problem.

We show how the theoretical developments presented in Reference [5], where octasection is applied as a tool for *uniform* refinement, may be easily adapted to our setting, i.e. non-uniform, adaptive refinement.

Using octasection we can refine two adjoining tetrahedra with automatic matching of the refined faces. As remarked in [1], octasection alone cannot produce a conforming mesh for standard finite elements during refinement. However, this is not a concern here, because the use of CHARMS will ensure the construction of a conforming mesh a priori, even though the geometrical mesh (i.e. supports of the basis functions) *looks* non-conforming. That frees us to use solely octasection – there is no need to apply special rules between coarse and fine neighbors.

We wish to point out that we do not modify or enhance the theoretical developments of Ong [5]; rather, we combine CHARMS with Ong's theory to produce an adaptive strategy that is able to guarantee shape quality without necessitating restoration of compatibility or other complications.

3.1. Kuhn triangulation of the cube

The developments presented in Reference [5] are based on the so-called *Kuhn triangulation* of a cube. The Kuhn triangulation divides the cube into six tetrahedra as shown in Figure 3. The tetrahedra are of equal volume, and in each pair they are either similar to each other or reflections of each other. Ong

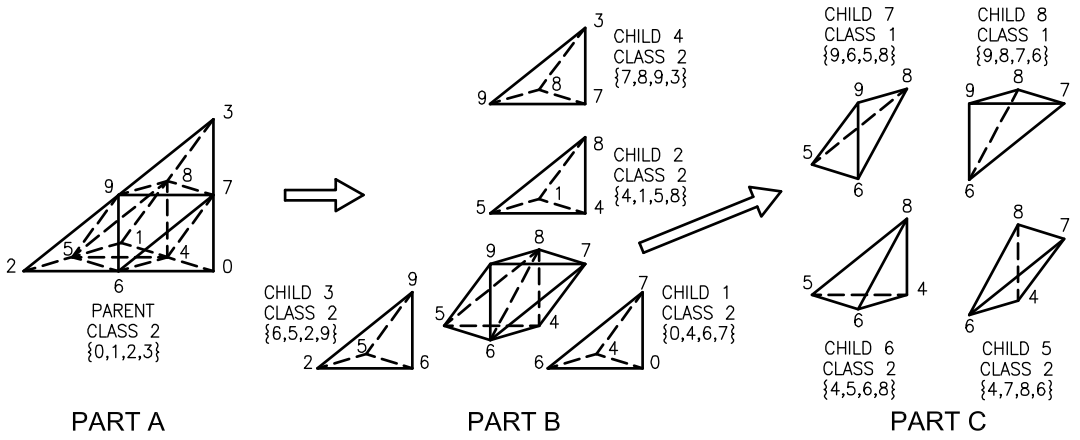

Figure 4. Ong's division of a class 2 tetrahedron.

uses the tetrahedra from the Kuhn triangulation of the cube as models for uniform refinement. We refer to these tetrahedra as class 1 and class 2. Class 2 tetrahedra are reflections of class 1 tetrahedra (and vice versa, of course) and all tetrahedra of the same class are similar.

Ong's uniform-refinement strategy divides a model tetrahedron into eight tetrahedra of equal volume [5]. We demonstrate the geometrical division of one model tetrahedron by octasection in Figure 4 on a tetrahedron of class 2. The midpoints of the edges are numbered and connected as shown in Part A. The four corners of the tetrahedra are then removed as shown in Part B, leaving an octahedron. Connecting points 8 and 6 with an edge divides the octahedron into the remaining four tetrahedra shown in Part C. The four tetrahedra from the octahedron form two pairs. Each pair consists of a tetrahedron and its reflection.

We have numbered the parent (tetrahedron being refined) and the children (tetrahedra resulting from refinement) as shown in Figure 4. This numbering yields consistently positive volume and ensures that the mappings between parent and child (and vice versa) are consistent.

3.2. Octasection strategy

The division of an arbitrary tetrahedron will be presented to clarify the division of a "reflected" child versus a "non-reflected" child and to segue to the shape qualities of the refined tetrahedron discussed in the next section.

Class 1 and class 2 tetrahedra must be refined differently to guarantee that the shape measures do not degrade with refinement. This is important as the octahedron that results during refinement produces two children that are "reflected" with respect to their parent. Therefore, these children must be refined differently than the other children.

With this in mind we examine the division of the arbitrary tetrahedron presented in Figure 5. We assume that this tetrahedron is of class 1 to demonstrate how the refinement differs from the refinement of a class 2 tetrahedron. (This assumption is arbitrary and an initial tetrahedron can be refined as class 1 or 2 as long as the children are refined consistent with its parent.)

We proceed as before, but, when the inner octahedron is divided, the 4-9 diagonal must be used instead of 6-8 diagonal which is used for a class 2 tetrahedra. (Compare Figures 4 and 5.) Moreover, because of the general shape of the tetrahedron, the resulting children are *not* similar to the parent.

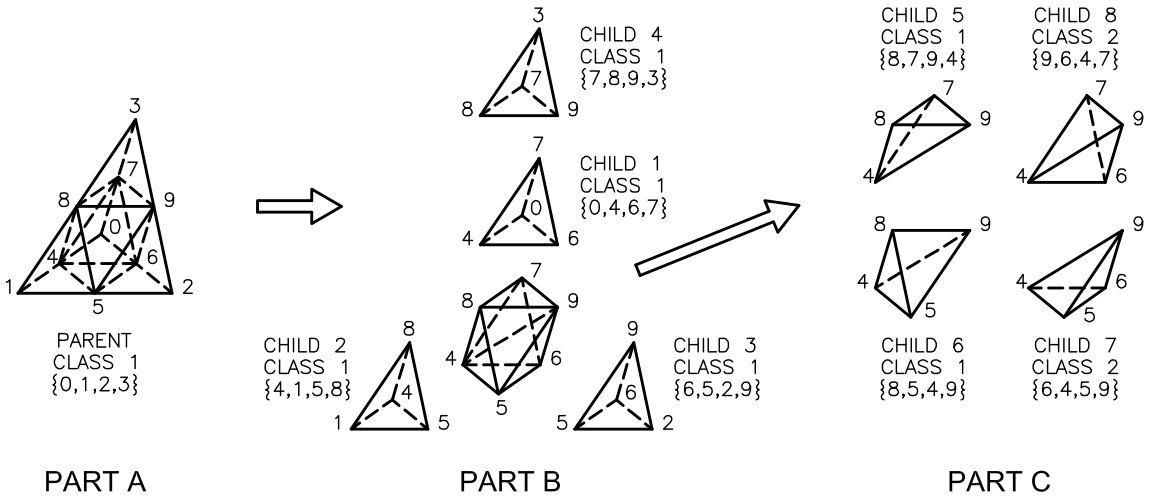

Figure 5. Division of an arbitrary tetrahedron of type class 1.

Ong has shown that these tetrahedra do not degrade any further with subsequent refinement [5]. The children are numbered as shown in Figure 5.

While it is not immediately obvious by examination, the "reflected" children are 7 and 8 [5]. This can be seen when the children are mapped to the parent (or by dividing one of Ong's model tetrahedra of class 1). In the general tetrahedron the children from the octahedron are not similar to the parent and therefore the "reflected" children are not actual reflections. We intend the use of reflection here to mean that the child would be a reflection of the parent if the model tetrahedra had been used for the initial refinement. More to the point, a reflected child is a child which must be refined using the opposite diagonal of the parent.

We have numbered the reflected children as 7 and 8 for both class 1 and 2 tetrahedra. This makes it easy to implement in code, simply use the opposite diagonal of the parent in refining the octahedron of children 7 and 8 and the same diagonal for the other children.

3.3. Shape Quality of Refined Meshes

The strategy adopted above is the one invoked by Ong. It insures that there are at most six types of tetrahedra during refinement (see Theorem 3.1 of Reference [5]). (Incidentally, it also happens to satisfy the labelled edge subdivision rule of Zhang [10].) However, the types of tetrahedra include the reflections, meaning that there are three pairs of shapes, each pair consisting of a tetrahedron and its reflection. One shape results from the removal of the corners of the tetrahedron and two more result from the octahedron division. The quality measure of tetrahedra which are reflections of each other are the same. Hence, the refinement of an arbitrary tetrahedron will result in at most three different values of quality measures of tetrahedra.

4. IMPLEMENTATION IN AN ADAPTIVE FRAMEWORK

The technique described in this paper has been implemented in a finite element software framework. The starting point is Equation (7), which despite its appearance does not differ very much from ordinary finite element expansions. The challenges described in Section 4 are minor. There is the

additional requirement of traversing through the levels of refinement. However, the mapping from one level to another is not expensive, and efficiency is not compromised.

Next we shall briefly address the last point, namely how to evaluate the basis functions and their derivatives.

4.1. Interlevel Mappings

For each child, a mapping between the child and parent (and vice versa) must be developed to map points between the refinement levels. For any nondegenerate tetrahedron there exists a mapping to another nondegenerate tetrahedron [5],

$$v = B\overline{v} + t \tag{13}$$

where

v is a vector in $\mathbb{R}^3$, a point in the parent tetrahedron;
$\overline{v}$ is a vector in $\mathbb{R}^3$, a point in the child;
B is a 3x3 invertible matrix;
t is a vector in $\mathbb{R}^3$;

The child is mapped to the parent's coordinates in such a way as to maintain the structure of the tetrahedron upon refinement. We have numbered the parent and the children in a consistent manner to easily implement this in a finite element framework. For example, the points 7, 8, 9, and 3, which are the vertices of child 4, get mapped to points 0, 1, 2, and 3 on the parent, respectively. The mapping of Equation (13) may be specialized to these four points (the corners) of the tetrahedron, thus yielding 12 equations from which to solve for B_{ij} and t_k.

The finite element framework that implements the present refinement evaluates the weak form integrals on the finest tetrahedra overlaying a given basis function. Thus, all basis functions need to be transformed into a single set of parametric coordinates. This is easily accomplished using Equations (13).

4.2. Refinement of a Randomly-shaped Tetrahedron

A tetrahedron with vertices at random locations in the unit cube has been refined uniformly seven levels deep to illustrate the bounds on the shape quality measures. The original tetrahedron shape quality measure is approx. 0.55. The first refinement introduces two more shape quality measure values, 0.44 and 0.63. Thereafter, no change in shape quality measures is observed. The resulting shape measures are shown in Figure 6.

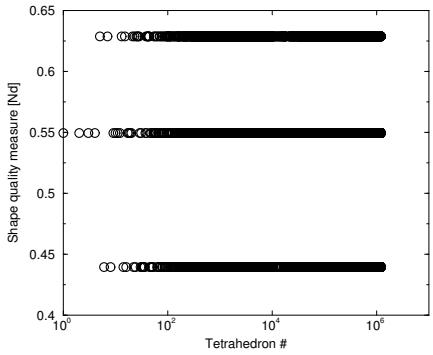

Figure 6. Shape quality measures of tetrahedra resulting from 7-level uniform refinement of a random initial tetrahedron.

4.3. Elastic deformation under a Punch

CHARMS was applied in the following example to the simulation of the displacement field in a block of isotropic elastic material under a rigid rectangular punch. One quarter of the symmetric geometry had been discretized with 86 tetrahedra, and subsequently adaptively (non-uniformly!) refined five levels deep, and the results and the computational grids are displayed in Figure 7. Shown are the tetrahedra that support the active basis functions. Notice also the dots that indicate nodes associated with a basis function.

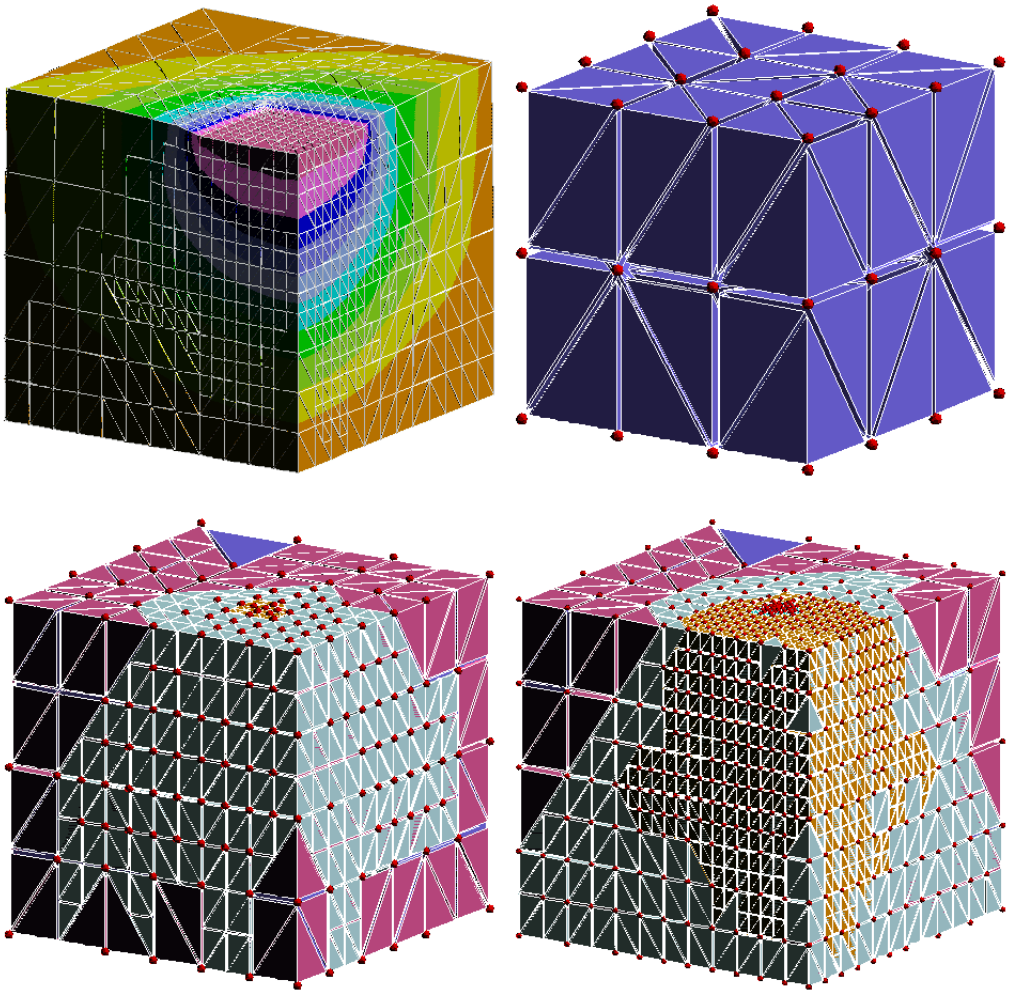

Figure 7. Left to right, top to bottom: The displacement contours displayed on the final refined model, the initial and intermediate refined models.

Figure 8 illustrates the shape quality measures (see Equation (12)) for the final mesh. As may be observed, the refinement results in a fairly small number of distinct quality measures: compare the original 15 shape quality measures with the approximately 37 distinct values in the refined mesh.

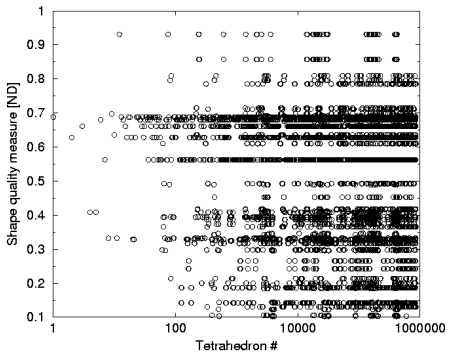

Figure 8. Shape quality measures of tetrahedra resulting from 5-level non-uniform, adaptive, refinement of the tetrahedral mesh of the block.

CONCLUSIONS

A general procedure for mesh refinement, CHARMS (which stands for Conforming Hierarchical Adaptive Refinement MethodS) [3], has been in this work adapted to approximations on tetrahedral finite element meshes. Octasection is being used to divide the finite elements in order to support the basis functions with increased resolution. CHARMS automatically ensures compatibility of the adapted basis, and also guarantees that a basis (i.e. a linearly independent basis set) is obtained for any refinement sequence. Using the theoretical results from Reference [5], the present approach guarantees that the quality measure of the divided elements is bounded for non-uniform, adaptive, refinement, irrespectively of the number of refinements. This is achieved by appropriately selecting one of the three possibilities of the division of the octahedron obtained after the corners of the input tetrahedron had been removed. The algorithm is fully described, and the procedure is illustrated with examples.

ACKNOWLEDGMENTS

The research reported here was supported in part by the NSF (DMS-9874082), and by the Hellman Fellowship 2001 (PK).

REFERENCES

1. D. N. Arnold, A. Mukherjee, and L. Pouly. Locally adapted tetrahedra meshes using bisection. *SIAM Journal on Scientific Computing*, 22(2):431–448, 2001.
2. E. Grinspun, P. Krysl, and P. Schröder. CHARMS: A simple framework for adaptive simulation. *ACM Trans. Graph.*, 21(3):281–290, 2002.
3. P. Krysl, E. Grinspun, and P. Schröder. Natural hierarchical refinement for finite element methods. *International Journal for Numerical Methods in Engineering*, 56(8):1109–1124, 2003.
4. A.W. Liu and B. Joe. Quality local refinement of tetrahedral meshes based on bisection. *SIAM Journal on Scientific Computing*, 16(6):1269–1291, 1995.
5. M.E.G. Ong. Uniform refinement of a tetrahedron. *SIAM Journal on Scientific Computing*, 15(5):1134–1144, 1994.
6. H. Samet. *Applications of spatial data structures*. Addison-Wesley, 2nd edition, 1995.
7. G. Strang. Wavelets and dilation equations: A brief introduction. *SIAM Review*, 31(4):614–627, 1989.

8. B. von Herzen and A. Barr. Accurate triangulations of deformed, intersecting surfaces. In *Proceedings of SIGGRAPH'87 conference*, pages 103–110, 1987.
9. H. Yserentant. Hierarchical bases. In *ICIAM91*. SIAM, 1991.
10. S. Zhang. *Mult-level Iterative Techniques*. Ph.d. thesis, Dept. of Mathematics, The Pennsylvania State University, University Park, PA, August 1988.