Belda, G.; Gargallo, A.; Roca, X. Local bisection for conformal refinement of unstructured 4D simplicial meshes. A: International Meshing Roundtable. "27th International Meshing Roundtable". Berlín: Springer, 2018, p. 229-247.
The final authenticated version is available online at https://doi.org/10.1007/978-3-030-13992-6_13

Local bisection for conformal refinement of unstructured 4D simplicial meshes

Guillem Belda-Ferrín, Abel Gargallo-Peiró* and Xevi Roca

Abstract We present a conformal bisection procedure for local refinement of 4D unstructured simplicial meshes with bounded minimum shape quality. Specifically, we propose a recursive refine to conformity procedure in two stages, based on marking bisection edges on different priority levels and defining specific refinement templates. Two successive applications of the first stage ensure that any 4D unstructured mesh can be conformingly refined. In the second stage, the successive refinements lead to a cycle in the number of generated similarity classes and thus, we can ensure a bound over the minimum shape quality. In the examples, we check that after successive refinement the mesh quality does not degenerate. Moreover, we refine a 4D unstructured mesh and a space-time mesh (3D + 1D) representation of a moving object.

1 Introduction

In the last three decades refinement of 2D and 3D unstructured simplicial meshes [1–14], based on red/green refinement [1–7] and bisection [8–14], has been shown to be a key ingredient on efficient adaptive loops. Although one could expect the same in 4D, a case of special interest for space-time adaption, this line of research has not been extensively explored.

For our space-time applications, we are interested in conformal bisection methods since they are really well suited to implement fast geometrical multi-grid conformal solvers. Moreover, bisection methods have ensured either a maximum number of generated similarity classes [11–13] or a minimum lower quality bound over the generated elements after successive refinements [8–10, 14]. Regarding 4D refinement, only a non-conformal local refinement method for pentatopic meshes has

Computer Applications in Science and Engineering, Barcelona Supercomputing Center, 08034 Barcelona, Spain.
*Corresponding author e-mail: abel.gargallo@bsc.es

been proposed [15]. Unfortunately, existent conformal 4D (nD) bisection methods with a bound over the number of generated similarity classes [11, 12] cannot be applied to general unstructured meshes.

The main contribution of this work is to propose a local bisection procedure, with a bound over the number of generated similarity classes, for conformal refinement of 4D unstructured simplicial meshes. Specifically, we propose a recursive refine to conformity procedure, in two stages, based on marking bisection edges on different priority levels (Sec. 3.1). The marking procedure allows classifying the pentatopes in different types (Sec. 3.2) and hence, determining different refinement templates (Sec. 4), in an analogous manner to the 3D bisection method proposed in [13].

The refinement method is composed of two stages (Sec. 4). Two successive applications of the initial stage of the bisection strategy (Sec. 4.1), based on the proposed element classification, ensure that any initial 4D unstructured simplicial mesh can be conformingly refined. After the two initial refinements our recursive refine to conformity strategy switches to the second stage (Sec. 4.2). This final stage is analogous to Maubach's algorithm, when it is successively applied to a single pentatope. Therefore, we can ensure a bound over the number of generated similarity classes. Thus, the minimum quality of the refined mesh is bounded, independently of the number of performed refinements. The main advantage and difference of our method when compared to Maubach's algorithm [11] is the first stage of the method, which allows the application of the method to any 4D unstructured simplicial mesh.

In all the examples (Sec. 5), we show that the proposed methodology leads to a periodic evolution of the minimum element quality (shape quality measure [16]) illustrating the lower bound of the quality through successive refinement. We first illustrate how to check that an implementation of the proposed method is valid by successively refining a pentatope. With our implementation, we show that the proposed bisection technique can be used to refine general unstructured 4D meshes. Finally, we also illustrate our application of interest, the refinement of a 4D mesh corresponding to a space-time representation, with varying resolution, of the temporal evolution of a 3D moving object.

2 Preliminaries

In this section, we state some preliminary notions required for the rest of this work. First, we detail how a pentatope in four dimensions is represented in the 2D plots of this paper. Second, we state the definition of bisection and finally, we introduce the strategy used in this work to refine a given mesh through edge bisection.

The element type considered in this work is the *pentatope* (4D simplex) which is defined as the convex hull of a set of 5 points $\{x_0, x_1, x_2, x_3, x_4\}$ in $\mathbb{R}^4$. To represent a given pentatope (4D) in the plane (2D), we focus on a perspective where the edges connecting the vertices have a minimal number of edge crossings. Herein, the pentatope $[x_0x_1x_2x_3x_4]$ is displayed plotting the edges of the tetrahedron $[x_0x_1x_2x_3]$ and the edges that connect the vertices of the tetrahedron $[x_0x_1x_2x_3]$ with the extra ver-

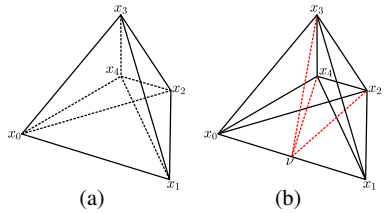

Fig. 1 (a) Three dimensional representation of a pentatope $[x_0x_1x_2x_3x_4]$, where the fifth vertex x_4 is plotted in $\mathbb{R}^3$ inside the tetrahedron $[x_0x_1x_2x_3]$. (b) Potential edges $[vx_2]$, $[vx_3]$ and $[vx_4]$ of the bisection of the pentatope $[x_0x_1x_2x_3x_4]$.

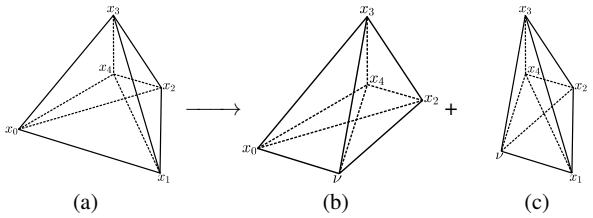

Fig. 2 Bisection of a pentatope (a) into two children (b) and (c).

tex x_4 located in the center of the tetrahedron, see Figure 1(a). This representation is used to display the edge marking procedure for bisection proposed in this work. The boundary of a pentatope is formed by 5 tetrahedra: the outer tetrahedron $[x_0x_1x_2x_3]$ and the four inner tetrahedra $[x_0x_1x_2x_4]$, $[x_0x_1x_4x_3]$, $[x_0x_2x_3x_4]$ and $[x_1x_2x_3x_4]$.

Once detailed the representation of a given pentatope, we particularize the definition of bisection to 4D simplicial elements. In particular, for a given pentatope σ with vertices $[x_0x_1x_2x_3x_4]$, the element vertices are reordered so that the refinement edge is $[x_0x_1]$. Let v be the midpoint of $[x_0x_1]$. The bisection of σ by $[x_0x_1]$ corresponds to removing the element $[x_0x_1x_2x_3x_4]$ and generating two new elements by joining v with the tetrahedral faces $[x_0x_2x_3x_4]$ and $[x_1x_2x_3x_4]$.

We highlight that the tetrahedral face $[vx_2x_4x_3]$ is shared between the two children. This shared face has three inherited edges ($[x_2x_3]$, $[x_2x_4]$ and $[x_3x_4]$) and three new edges ($[vx_2]$, $[vx_3]$ and $[vx_4]$). We denote the new edges of the shared face as *potential edges* of the initial element. These potential edges are displayed in Figure 1(b) colored in red. This definition is required in Section 4.2 to characterize the proposed mesh refinement templates.

Finally, we introduce the algorithm proposed in this work to refine a given mesh by edge bisection. This algorithm uses a refine to conformity strategy similar to the 3D refinement method proposed in [13]. Given a marked mesh M and a set of elements to refine S, the mesh is refined according to Algorithm 1. In this algorithm, while there is not an empty set of elements to refine, Line 2, the mesh is refined as follows. In Line 3, the process `BisectPentatopes` bisects each pentatope in S:

Procedure 1 Refinement of a mesh ensuring conformity.

Input: Marked mesh M
Output: Marked mesh M'
1: **function** REFINETOCONFORMITY(M,S)
2: **if** $S \neq \emptyset$ **then**
3: $\bar{M} = \texttt{BisectPentatopes}(M,S)$
4: $S = \{\sigma \in M \,|\, \sigma \text{ has a hanging node}\}$
5: $M' = \texttt{RefineToConformity}(\bar{M})$
6: **else**
7: $M' = M$
8: **end if**
9: **end function**

$$\texttt{BisectPentatopes}(M,S) = (M \setminus S) \cup \bigcup_{\sigma \in S} \texttt{Bisect}(\sigma), \tag{1}$$

where `Bisect` performs the element bisection taking into account the element marks (refinement edge) and sets the proper marks to the two generated elements. In Sections 3 and 4 we will present the marking procedures proposed in this work for pentatopic meshes, and the marks that are assigned to the two children.

Following, in Line 4 the set of elements to refine in the next step is set as the elements with hanging nodes. In Line 5 the Algorithm `RefineToConformity` is called recursively. These recursive calls are continued until there are no more elements with hanging nodes in the mesh. We show that the marking processes presented in Sections 3 and 4 lead to a conformal mesh.

3 Edge marking and element classification for compatible refinement

In this section, we first present in Sec. 3.1 an edge marking process compatible between neighboring elements for conformal mesh refinement. Next, in Sec. 3.2 we present a classification of the elements of the mesh depending on the marks assigned to their edges.

3.1 Edge marking for compatible refinement

In this work, we use a marking procedure organized by levels to determine the priority of the bisection edges used during the element refinement. Following, we present a procedure to mark the edges of the pentatopes of a conformal mesh. These marks are devised to ensure that for a given face shared between two pentatopes, successive bisection of surrounding elements determines the same mesh from both sides of the shared face. Hence, this ensures mesh conformity along the bisection process.

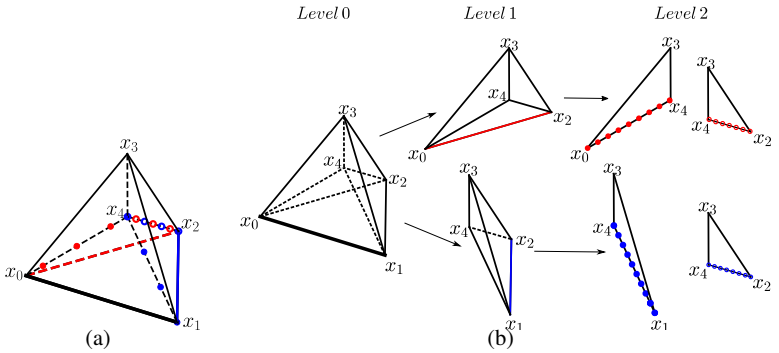

Fig. 3 (a) Marked pentatope and (b) marking diagram process at different levels.

We define three levels of marks in a pentatope. The level 0 features one edge, which corresponds to the refinement edge of the current pentatope. The level 1 features two edges, which correspond to the refinement edges of the two children of the first pentatope. Finally, the level 2 features four edges, which correspond to the refinement edge of the four grandchildren of the original pentatope.

Herein, to determine the marks assigned to each of the edges of the element, we prioritize the edges in terms of their length with a well-defined tie-breaking rule. For a given element we define its *consistent bisection edge* as the edge of longest length and lowest global index. The lowest global index is a tie-breaking rule that ensures that if there exist multiples edges with the same length, we select as the longest edge always the same one, independently of the order in which the edges are compared. In particular, with this tie-breaking rule we ensure that the edges of a common face between two adjacent pentatopes are marked in the same manner from the two of them. We remark that the longest edge is considered in the *consistent bisection edge* sorting rule since it is an heuristic to enforce better element quality. Nevertheless, it is not a key ingredient to ensure conformal mesh refinement. For instance, just by sorting the mesh edges using their global index would also lead to a valid consistent sorting rule.

Following, we detail the marking process for a given pentatope $[x_0x_1x_2x_3x_4]$. The process consists of three steps, illustrated in Figure 3(b):

1. Marked edge of level 0: *consistent bisection edge* of the pentatope $[x_0x_1x_2x_3x_4]$. In the bisection process, the marked edge of level 0 corresponds to the bisection edge of the element. In this work, the marked edges of level 0 are plotted with a thick black line, see the first column of Figure 3(b).
2. Marked edges of level 1: the two marked edges of level 1 are determined as the *consistent bisection edge* of the tetrahedra defined by $[x_1x_2x_3x_4]$ and $[x_0x_2x_3x_4]$. These two tetrahedra are indeed the opposite tetrahedral faces of the pentatope with respect to x_0 and x_1, respectively. These two tetrahedral faces are the faces of the original pentatope preserved in each child. The two marked edges of level 1 correspond to the bisection edges of the two children of the current element.

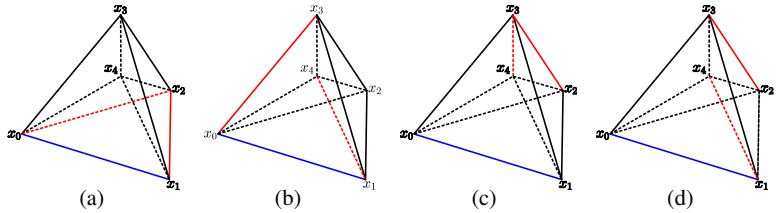

Fig. 4 Type of edge relations between the edges on level $l+1$ (red) and the edge of level l (blue): (a) P, (b) A, (c) O, and (d) M.

A particular configuration of the marked edges of level 1 is illustrated in the second column of Fig. 3(b). The marked edges of level 1 associated to the first and second node of the bisection edge are colored in red and blue, respectively.

3. Marking edges of level 2: the four marked edges of level 2 are determined as the *consistent bisection edge* of the opposite faces of the marked edges of level 1 in the tetrahedra $[x_1x_2x_3x_4]$ and $[x_0x_2x_3x_4]$. The four marked edges of level 2 correspond to the bisection edges of the four grandchildren of the current marked element. In the third column of Figure 3(b), we illustrate a particular configuration of the marked edges of level 2, coloring them with the same color of the associated marked edge of level 1. In addition, the edge associated to the first node of the marked edge of level 1 is plotted with fully colored circles, and the other edge is plotted with empty circles.

Figure 3(a) illustrates the resulting marked element for the test example of the marking procedure of Fig. 3(b). We highlight an edge, for instance $[x_2x_4]$ in Fig. 3(a), can have two marks once all the marks are displayed on the initial pentatope. These two marks indicate that this edge has been marked from both of the faces that remain after bisection. To differentiate them, we have used blue and red colors. After bisecting a marked pentatope, the marked edges of the two children have to be determined.

Remark 1 (Inheritance of marks). The marked edges of level 1 and 2 of the parent shift marks in the corresponding children and become the marked edges of level 0 and 1 of the two children, respectively. However, it is not straight-forward to determine the marked edges of level 2 from the parent marks. In Section 4 two methods are proposed to determine them.

3.2 Classification of marked pentatopes

In this section, we present a classification into different types of a pentatope resulting from the marking process detailed in Section 3.1. Several types of pentatopes are obtained depending on the marks assigned to their edges. Before detailing the

classification, we introduce four definitions that state how the marked edges of level $l+1$ are located with respect to the associated marked edge of level l for $l = 0, 1$.

We propose a classification for different pentatope types, according to the configuration of the marked edges at the different levels. This classification is an extension of the different tetrahedron types proposed in [13], where only two levels of marked edges are required. Figure 4 illustrates the four different configurations between two levels of marked edges, coloring the two marked edges of level $l+1$ with red color and the marked edge of level l with dark blue color:

- Type P (Planar): the two marked edges of level $l+1$ are coplanar with the marked edge of level l, *i.e.*, the three edges are connected defining a triangle. In Figure 4(a) an example of edges of type P is illustrated.
- Type A (Adjacent): each marked edge of level $l+1$ has a common vertex with the marked edge of level l but the two edges of level $l+1$ do not have any common vertex. In Figure 4(b) an example of edges of type A is illustrated.
- Type O (Opposite): the marked edges of level $l+1$ of the opposite faces of the marked edge of level $l+1$ do not intersect the marked edge of level l. In Figure 4(c) an example of edges of type O is illustrated. We highlight that a possible configuration of edges of type O is that the two edges of level $l+1$ are overlapped. For instance, the edge $[x_2x_3]$ could be the marked edge of level $l+1$ for the two faces opposite to the edge of level l.
- Type M (Mixed): the marked edges of level $l+1$ of just one of the opposite faces have a common vertex with one marked edge of level l. In Figure 4(d) an example of edges of type M is illustrated. We highlight that it is possible that the marked edges of level $l+1$ have a common vertex between them. For example, the marked edges of level $l+1$ could be $[x_1x_4]$ and $[x_4x_3]$.

Herein, in a pentatope we have marked edges of level 0, 1 and 2. We denote by α the edge type determined by how marked edges of level 1 are located with respect to the marked edge of level 0. Additionally, we denote by β and γ the edge relation type between the marked edges of level 2 and the marked edge of level 1. In this manner, a marked pentatope is classified into a type of the form $\alpha_{\beta\gamma}$.

In Figure 5, we illustrate three different types of marked pentatopes. First, Figure 5(a) illustrates a pentatope of type P_{PP}. In particular, the marked edge of level 0 (bisection edge) configures a triangular face together with the marked edges of level 1. Thus, the first index, α, of the element is P. Next, each marked edge of level 1 defines also a triangular face with the corresponding marked edges of level 2, determining β and γ equal to P. Hence, the element is of type P_{PP}.

Analogously, for the element illustrated in Figure 5(b) we detail the same process. For this element, α is A since the red and blue edges share a node with the bisection edge, but they do not share any node between them. In addition, β and γ are equal to P, since each of the blue and red edges determines a triangular face with the corresponding blue and red circled edges. Thus, this element is of type A_{PP}. Similarly, we can conclude that the element illustrated in Figure 5(c) is of type A_{AA}.

After bisecting the element, the bisection edge $[x_0x_1]$ is split into two edges, $[x_0v]$ and $[vx_1]$, and thus this edge is not present in any of the children. However, the two

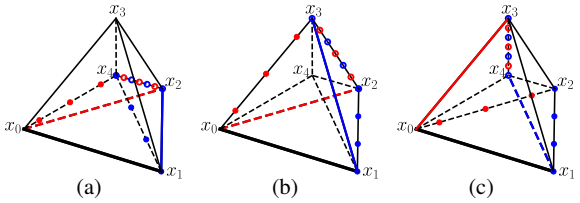

Fig. 5 Three different types of marked pentatopes: (a) P_{PP}, (b) A_{PP}, and (c) A_{AA}.

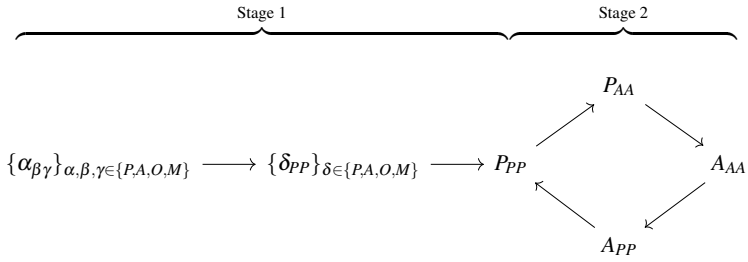

Fig. 6 Refinement process for a pentatope of type $\alpha_{\beta\gamma}$, where $\alpha,\beta,\gamma\in\{P,A,O,M\}$.

adjacent tetrahedral faces to this edge are preserved. Specifically, the face $[x_0x_2x_3x_4]$ is inherited by the child that preserves node x_0 of the bisection edge, and the face $[x_1x_2x_3x_4]$ is inherited by the child that preserves node x_1. Following we detail which marks of the parent pentatope are preserved after its bisection and how these marks are inherited by the two children.

Remark 2 (Inheritance of element type). Hence, after bisecting a marked pentatope of type $\alpha_{\beta\gamma}$, where α,β,γ can be $\{P,A,O,M\}$, the obtained children inherit the marked edges of level 1 and 2 of the parent. These marks become the marked edges of level 0 and 1 of the children, see Remark 1. Thus, one child inherits the edge relation type β and the other child the relation type γ. However, the marked edges of level 2 are not determined. Depending on the edges that are selected to be the marked of level 2, the type of element of the children will be $\beta_{\beta_1\beta_2}$ and $\gamma_{\gamma_1\gamma_2}$, where $\beta_1,\beta_2,\gamma_1,\gamma_2\in\{P,A,O,M\}$.

4 A refinement algorithm for 4D unstructured simplicial meshes with bounded number of similarity classes

In this section, we detail a new procedure composed by two stages for refinement of any 4D unstructured simplicial mesh. Given a mesh, we first mark it using the procedure stated in Section 3.1, and following, we classify the elements into the different types stated in Section 3.2. Next, given a marked element to be refined, the bisec-

Procedure 2 Element refinement with global mesh conformity

Input: Pentatope σ, set of marked edges m_σ, descendant level k.
Output: Pentatopes σ_1 and σ_2, set of marked edges m_{σ_1} and m_{σ_2}, descendant level k'

1: **function** BISECT(σ , m_σ, k)
2: $\sigma_1, \sigma_2 =$ `BisectPentatope`(σ, m_σ)
3: $m_{\sigma_1}, m_{\sigma_2} =$ `inheritMarksFromFather`(m_σ)
4: **if** $k < 2$ **then**
5: $m_{\sigma_1}, m_{\sigma_2} \leftarrow$set marked edges of level 2 using Stage 1 from Sec. 4.1
6: **else**
7: $m_{\sigma_1}, m_{\sigma_2} \leftarrow$set marked edges of level 2 using Stage 2 from Sec. 4.2
8: **end if**
9: $k' = k+1$
10: **end function**

tion of this element is performed according to Algorithm 2 and the diagram in Fig. 6. Algorithm 2 is used as bisection procedure in the `BisectPentatopes` function, Eq. (1), from the mesh refinement strategy `RefineToConformity` presented in Algorithm 1.

The two stages bisect a given marked element according to the bisection edge, Line 2 from Alg. 2, and following, according to Remarks 1 and 2, the marked edges of level 0 and 1 of the children are determined from the marked edges of level 1 and 2 of the parent, Line 3. The difference between the two stages is the process to set the marked edges of the children. The marks determine the type of the generated element and at the same time, how the children will be bisected through successive refinement.

The first two times that the element is bisected (Stage 1), Line 4 of Alg. 2, the marks of level 2 of the children are determined using Sec. 4.1. A child generated with one application of Stage 1 is of type δ_{PP}, being δ any edge relation type. No element enters to Stage 2 before two refinements, and once it is bisected twice in Stage 1, in Section 4.1 we show that it is of type P_{PP}. Then, from the second refinement and on, Line 7, Stage 2 is activated, see Section 4.2. In Section 4.3 the properties of the two-stage method are presented.

4.1 Stage 1: refinement from any unstructured marked mesh to P_{PP} elements

Stage 1 determines the marked edges of level 2 following the ideas of the marking strategy presented in Section 3.1. From the marking diagram presented in Figure 3(b) we observe that two of the edges of the triangular face of the third column remain unmarked in the parent. After the element is bisected, these edges are still present in the tetrahedral faces of the children. These edges can be enforced to be the marked edges of level 2 of the children. This decision is consistent by construction between adjacent elements since it is performed on the face shared between these

elements. This approach to determine the marked edges of level 2 of the children leads to a conformal refinement procedure.

Remark 3 (Refinement towards P_{PP} elements). Given an element of type $\alpha_{\beta\gamma}$ for $\alpha,\beta,\gamma \in \{P,A,O,M\}$, the application of two refinements of Stage 1 leads to elements of P_{PP}, see Figure 6.
To show this, we first focus on the initial refinement step. From Remark 2 the children will be $\beta_{\beta_1\beta_2}$ and $\gamma_{\gamma_1\gamma_2}$, where $\beta_1,\beta_2,\gamma_1,\gamma_2 \in \{P,A,O,M\}$ depend on how the marked edges of level 2 are located with respect to the marked edges of level 1. By construction (see Fig. 3(b)), the marked edges of level 2 have been chosen on the same triangular face of the corresponding marked edge of level 1. Thus, the new marked edges of level 2 are coplanar with the marked edges of level 1 for each child and their edge relation is of type P. Hence, by setting these edges as marked edges of level 2 of the children, we obtain two children of type β_{PP} and γ_{PP}, respectively. Applying this marking strategy again, the grandchildren of the original pentatope are of type P_{PP}.

Although the marking process is consistent between adjacent elements by construction and the marks of level 2 are chosen consistently with the marking process, following we analyze all the possible neighboring configurations between two marked elements to illustrate that the stated bisection procedure is conformal.

Remark 4 (Conformal refinement). Given two neighbor marked elements, when the shared face is bisected from the two sides, it is bisected by the same edge. That is, the interface between the children of the two elements is still conformal. We analyze three different configurations of the two elements:

- First, let us assume that both elements share a face that contains their *consistent bisection edge*. This edge must be the same for each one of the elements, since in particular, it is the *consistent bisection edge* of the face. Then, it is clear that they are refined by that edge and that the new interface is conformal.
- Second, let us assume that the shared face does not contain the *consistent bisection edge* in any of the two adjacent elements. Following the stated marking procedure, the shared tetrahedral face is marked in the second column of Figure 3(b), containing the marked edges of level 1. Thus, in the first refinement of the elements, the face is not refined and the interface is still conformal. Next, when we perform the second refinement, the shared face is refined by the same edge from the two elements, ensuring a conformal bisection.
- Finally, the third case to be analyzed is when the face contains the *consistent longest edge* of the pentatope in one element, but does not contain the *consistent longest edge* of the adjacent pentatope. After refining once the elements, the mesh is not conformal, since the face is bisected from one of the elements, but is not bisected from the other one. However, the element that has not bisected the initially shared face, does bisect it after the second refinement, since the *consistent longest edge* of the adjacent pentatope is specifically the *consistent longest edge* of the shared face, and thus it is marked in the level 1 of the second element. Hence, after two iterations the mesh is already conformal.

Procedure 3 Bisection of a simplex from Maubach [11].

Input: Tagged n-simplex σ .
Output: Tagged n-simplices σ_1 and σ_2.
1: **function** BISECTSIMPLEXMAUBACH(σ)
2: Set $d' = \begin{cases} d-1, & d>1 \\ n, & d=1 \end{cases}$
3: Create the new vertex $z = \frac{1}{2}(x_0 + x_d)$.
4: Set $\sigma_1 = ((x_0, x_1, \ldots, x_{d-1}, z, x_{d+1}, \ldots, x_n), d')$.
5: Set $\sigma_2 = ((x_1, x_2, \ldots, x_d, z, x_{d+1}, \ldots, x_n), d')$.
6: **end function**

In addition, in the three different presented configurations, the marks determined on the children are always compatible by construction. Analogously, the same reasoning follows for the case where two pentatopes share a triangular face.

4.2 Stage 2: conformal refinement of all-P_{PP} meshes

In this section, we present a conformal refinement algorithm with a bounded number of generated similarity classes for meshes composed uniquely by elements of type P_{PP}. This algorithm determines the second stage of the refinement method for any unstructured mesh presented in Section 4.

The procedure presented in this section is stated in terms of a cycle composed of four steps, presented in Fig. 6. In Fig. 7 the templates for the bisection and setting of the marked edges of the children are presented. Given an element of type P_{PP}, Fig. 7(a), this element is split into two P_{AA} elements setting their marks using the templates presented in Figs. 7(b) and 7(c). After that, the type P_{AA}, Fig. 7(d), is bisected into two A_{AA} types applying the templates of Figs. 7(e) and 7(f). Following, an element of type A_{AA}, Fig. 7(g), is bisected into two A_{PP} using the templates presented in Figs. 7(h) and 7(i). Finally, from the type A_{PP}, Fig. 7(j), we obtain again two P_{PP} types applying the templates of Figs. 7(k) and 7(l).

We highlight that in order to apply the templates of Figure 7 we need to reorder the vertices of a given P_{PP} element to match the canonical representation of Figure 7(a). Similarly, the two children in Figures 7(b) and 7(c) have to be reordered to obtain the canonical P_{AA} in Figure 7(d) and then apply the corresponding templates. This node reordering has to be performed after each bisection to locate the marks in the canonical representation of the templated fathers. In addition, we highlight that in Figures 7(d), 7(b) and 7(c) the marked edges of level 2 are assigned on potential edges (see definition in Sec. 2). Although those edges do not exist on the parent, they exist in the children and grandchildren, where they will be used to determine the bisection edge.

Next, in Remark 5 we detail that this templated refinement procedure is analogous to Maubach's algorithm [11] when applied successively to one element.

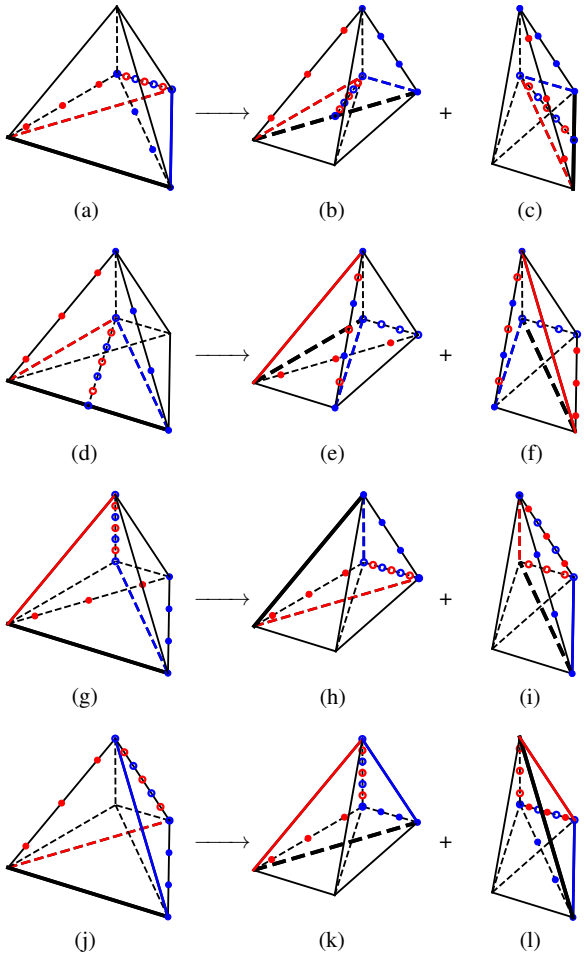

Fig. 7 Templates to perform the refinement cycle presented in Figure 6. (a)-(c) An element of type P_{PP} is bisected into two P_{AA}. (d)-(f) P_{AA} is bisected into two A_{AA}. (g)-(i) A_{AA} is bisected into two A_{PP}. (j)-(l) A_{PP} is bisected into two P_{PP}.

Maubach's algorithm cannot be applied in general to any given unstructured mesh as detailed in [11, 13]. Thus, finally in Remark 6, we analyze the conformity of the application of our approach for meshes composed of P_{PP} elements.

Remark 5 (Analogy to Maubach's algorithm). The refinement cycle in Fig. 6 performed using the templates presented in Fig. 7 is analogous to Maubach's algorithm [11] (see Alg. 3) when applied to a single pentatope. This analogy is interpreted as follows. Given a pentatope to bisect using Maubach's algorithm with a tag d, we consider as marked edge of level 0 the tagged edge. Next, we consider as marked edges of level 1 the tagged edges of the two children in the next application

Table 1 Permutations from the Maubach Algorithm 3 to canonical types in Figure 7

Canonical type	Tag in Algorithm 3	Permutation to obtain canonical representation
P_{PP}	$d=2$	$(0,2,1,3,4)$
P_{AA}	$d=1$	$(0,1,2,3,4)$
A_{AA}	$d=4$	$(0,4,2,3,1)$
A_{PP}	$d=3$	$(0,3,2,1,4)$

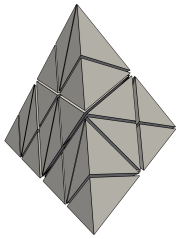

Fig. 8 Tetrahedral face of a pentatope of type P_{PP} after five refinements of the face .

of Maubach's algorithm. Analogously, we consider as marked edges of level 2 the tagged edges of the four grandchildren. Next, we find the permutation of the vertices $[x_0x_1x_2x_3x_4]$ to align the marks on the edges of the element with the canonical representation from Fig. 7. The obtained permutations are presented in Table 1.

Remark 6 (Conformal refinement for all-P_{PP} meshes). The refinement using Stage 2 of a marked mesh composed by elements of type P_{PP} leads to a conformal mesh. To illustrate the conformity of the refined mesh, we analyze two different cases:

- First, we analyze the case of the refinement of a single element. Since our method is analogous to Maubach's by Remark 5, it is also conformal when there is a single element successively refined, see details in [11, 13].
- Second, we analyze the conformity between the interface of adjacent elements of type P_{PP} with compatible marks. Extending the reasoning for tetrahedra in [13], it is sufficient to check if the bisection structure determined on a shared face is the same from both sides. Given a P_{PP} element to be refined, if we obtain the same refined mesh on all its tetrahedral faces we can ensure that the refinement of two adjacent P_{PP} is also conformal when using the `RefineToConformity` strategy. In particular, if we refine five times any of the five tetrahedral faces of a given P_{PP} the same tetrahedral mesh is obtained for all of them. This refined face mesh is illustrated in Figure 4.2 and is composed by 32 tetrahedra. The same reasoning follows for the case where two pentatopes share a triangular face.

Hence, if a pentatopic mesh can be marked with all elements as P_{PP}, then it can be conformingly refined using our analogy to Maubach's algorithm combined with the `RefineToConformity` strategy. This is the case when any given mesh is refined two times with Stage 1 in Sec. 4.1.

4.3 Properties of the method

In this section, we analyze the two main properties of the refinement procedure determined by Algorithm 2. Our refinement procedure requires as input a conformal unstructured 4D simplicial mesh. Given a set of elements to refine, the resulting mesh is a locally refined unstructured 4D simplicial mesh that is conformal and has a bounded number of generated similarity classes. These properties are discussed in the following remarks.

Remark 7 (Conformal refinement). The algorithm presented in Section 4 generates a conformal mesh. To show this, we take into account that this algorithm combines two refinement methods. The two first refinement steps in Stage 1 are performed by the algorithm presented in Sec. 4.1. After two refinements the elements are refined in Stage 2 according to the cycle in Fig. 6, see Sec. 4.2. In the worst case scenario, to prove conformal mesh refinement, all the elements of the initial mesh have to be twice refined at Stage 1. At this point, all the elements of the mesh are of type P_{PP} with compatible marks, as detailed in Remark 4. Then, the conformity of the refinement is ensured by Remark 6.

Remark 8 (Bounded number of generated similarity classes). The number of similarity classes produced by the repeated application of the cycle presented in Fig. 6 to an element is bounded by 1536. To prove this bound, we take into account that in the refinement scheme of Figure 6 it is required to perform two bisection steps before entering in the cycle. For each bisection, we generate at most two new similarity classes. Hence, from the given initial element, the bound of the similarity classes after the two first steps is $2 \cdot 2 = 4$. As highlighted in Remark 5 from Sec. 4.2, this second stage is analogous to Maubach's algorithm when applied to a single pentatope. In [13] it is proved that in 4D Maubach's algorithm has a sharp bound of 384 generated similarity classes for an element. Thus, the bound for the procedure of Figure 6 is $4 \cdot 384$, that is 1536.

5 Results

In this section, we present several results to illustrate the features and the applicability of the presented refinement scheme. In all the examples, we plot the minimum and maximum shape quality [16] in each refinement step of Alg. 1. To visualize the results we intersect each 4D mesh with a hyperplane to obtain a 3D cut that can be visualized. In Section 5.1, we refine an equilateral pentatope with two different initial marking configurations to illustrate that the similarity classes are bounded. In Section 5.2, we refine an unstructured 4D mesh to capture a hypersphere and, finally, in Section 5.3 we refine a simplicial mesh on a hypercube to capture a moving sphere.We highlight that in all the presented examples it has been explicitly checked that the generated meshes are conformal after the applied

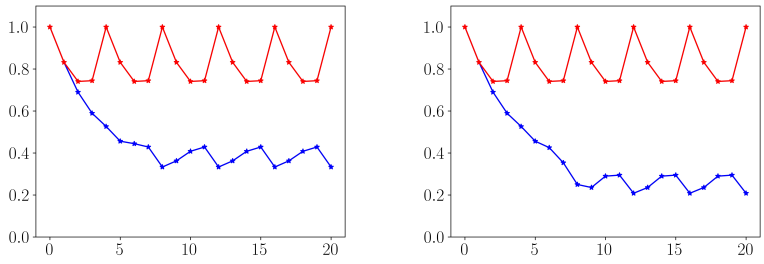

Fig. 9 Quality versus the number of iterations of the `RefineToConformity` algorithm applied to an equilateral pentatope marked as (a) P_{AA} and (b) A_{PO} type. The blue (red) line corresponds to the minimum (maximum) of the element shape quality at each iteration.

`RefineToConformity` strategy by checking that the only boundary faces of the mesh are on the boundary of the domain.

5.1 Bounded quality: iterative refinement of one pentatope

In this example, we check that our implementation of the refinement algorithm does not lead to degenerated elements after successive refinement of a given pentatope type. We enforce an equilateral pentatope to be marked as P_{AA} and a second equilateral pentatope to be marked as A_{PO}. Then, both pentatope types are globally refined 20 times. Figure 9 shows the minimum and maximum element quality at each refinement step. We observe that the minimum quality (vertical axis) decreases on the first refinement steps (horizontal axis) until a minimum value is reached. Then, the minimum and maximum qualities start to cycle every 4 refinement steps. This is an indicator of the bound of the number of generated similarity classes.

5.2 4D unstructured mesh: refining an extruded sphere octant

This example shows that the proposed refinement scheme can be applied to unstructured 4D pentatopic meshes. To this end, we generate an unstructured 4D mesh of a 3D sphere octant, of radius 1 and centered in the origin, extruded one unit along the fourth dimension. Then, we successively refine those elements that intersect a hypersphere of radius $1/4$ and centered in the origin. To generate the 4D mesh, we first generate an unstructured 3D mesh of the sphere octant composed by 40 nodes and 95 elements, see Figure 10(a). Then, we embed two copies of the 3D mesh points in the 4D space by setting the fourth coordinate to 0 and 1, respectively. Finally,

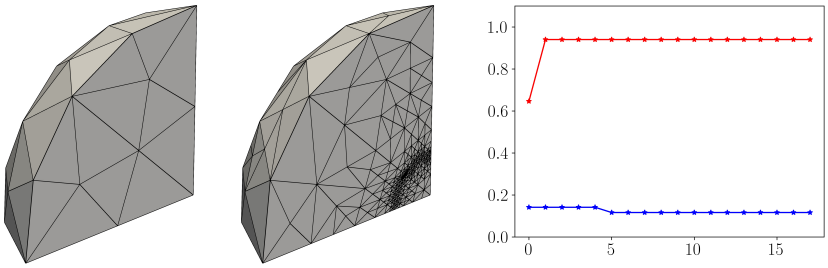

Fig. 10 A slice with the hyperplane $t = 0$ of the 4D simplicial mesh is illustrated for (a) the initial configuration and (b) after 17 iterations of `RefineToConformity`. (c) Minimum (blue) and maximum (red) element quality for each refinement step.

these 80 points are reconnected, using the implementation of the Delaunay algorithm provided by QHull [17], to obtain the unstructured 4D mesh. After applying 17 times the `RefineToConformity` algorithm, we obtain a 4D mesh composed by 8072909 elements and 433887 nodes. Figure 10(b), shows the tetrahedral mesh that corresponds to the boundary of the 4D pentatopic mesh at the base of the extrusion along the fourth dimension. Figure 10(c) shows the quality at each refinement step, where we can observe a lower quality bound is constant at value 0.11.

5.3 Space-time mesh: refining a sphere moving along the z-axis

Finally, we illustrate our application of interest, the refinement of a 4D mesh corresponding to a space-time representation, with varying resolution, of the temporal evolution of a 3D moving object. We consider a sphere of radius $1/5$ centered in the origin that moves along the z-axis from 0 to 1 with constant velocity 1. We generate an initial mesh on the hypercube $[0,1]^4$ composed by 24 pentatopes using Freudenthal-Kuhn algorithm [1–3]. Next, we apply 25 times the algorithm `RefineToConformity` to refine those elements that intersect the 4D sphere extrusion that represents the moving sphere. The final 4D mesh is composed by 5233296 pentatopes and 251457 nodes and it is illustrated in Figure 11. Figures 11(a)-11(c) show three slices of the mesh at $t = 0$, $t = 1/2$ and $t = 1$, respectively. We can observe that each one of the slices on t shows different positions of the moving sphere, from the initial point (0,0,0) at $t = 0$ to the final point $(0,0,1)$ at $t = 1$. In contrast with these three slices, in Figure 11(d) we show an slice of the mesh at $x = 0$. In the closest quadrilateral face of Fig. 11(d) we observe the path of the sphere on the surface of dimension 2 defined by the axis z and t at $x = y = 0$. In this quadrilateral face, we can see that the center of the sphere describes a straight line going from the lower left corner $(0,0,0,0)$ up to the top right corner $(0,0,1,1)$.

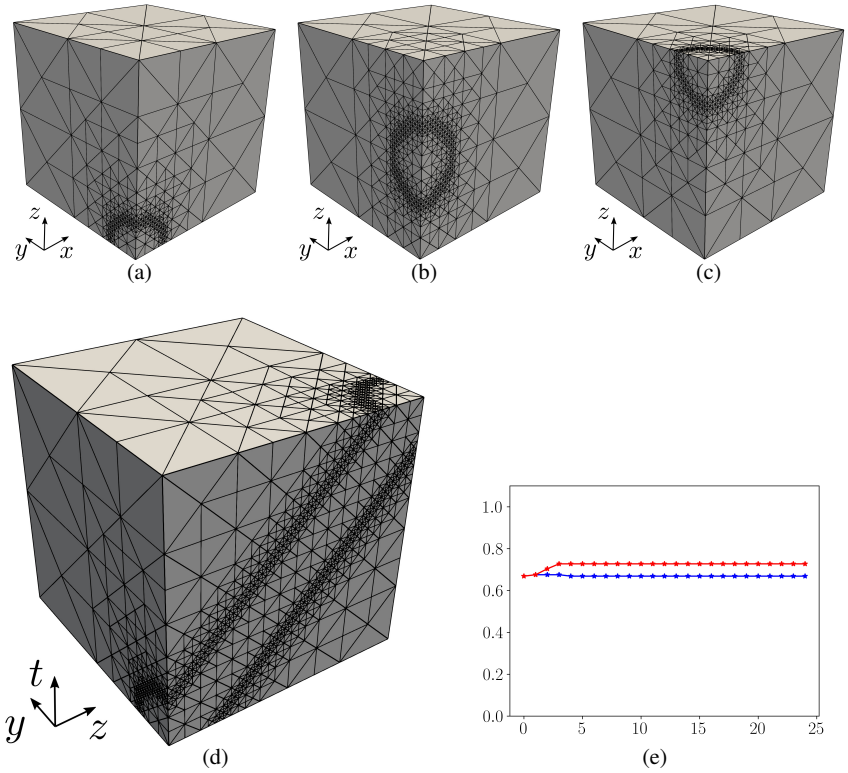

Fig. 11 Slice of the 4D simplicial mesh of the hypercube with the hyperplane: (a) $t = 0$, (b) $t = 0.5$, (c) $t = 1$ and (d) $x = 0$. (e) Minimum (blue) and maximum (red) element quality for each refinement step.

This is so since the sphere goes from $z = 0$ to $z = 1$ with constant velocity starting at $t = 0$ and finalizing at $t = 1$. Specifically, the location on the z-axis of the sphere is $z = t$.

6 Concluding remarks

In this work, we have presented a new refinement method via edge bisection for 4D pentatopic meshes. This method ensures that the mesh quality does not degenerate after successive refinements of a given element. To develop this method, we require to classify the elements of the mesh into different types in a similar fashion to [13]. Using the pentatope classification we provide four refinement templates to perform a cyclic bisection analogous to Maubach's method [11]. Combining two initializing refinements (Stage 1) with this templated refinement (Stage 2) we obtain a refine-

ment strategy that can be applied to any given pentatopic mesh. Using this method a finite number of similarity classes are generated when a given element is refined.

We apply the refinement scheme to different meshes to illustrate its features. First, we analyze that the mesh quality of the refinement of different element types does not degenerate. Second, we illustrate the applicability of the technique to refine unstructured 4D simplicial meshes. Finally, we analyze a space-time configuration of a sphere moving along an axis.

Acknowledgements This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 715546. This work has also received funding from the Generalitat de Catalunya under grant number 2017 SGR 1731. The work of X. Roca has been partially supported by the Spanish Ministerio de Economía y Competitividad under the personal grant agreement RYC-2015-01633.

References

1. H Freudenthal. Simplizialzerlegungen von beschrankter flachheit. *Ann. Math.*, pages 580–582, 1942.
2. H Kuhn. Some combinatorial lemmas in topology. *IBM Journal of research and development*, 4(5):518–524, 1960.
3. J Bey. Simplicial grid refinement: on freudenthal's algorithm and the optimal number of congruence classes. *Numerische Mathematik*, 85(1):1–29, 2000.
4. R Bank, A Sherman, and A Weiser. Some refinement algorithms and data structures for regular local mesh refinement. *Scientific Computing*, 1:3–17, 1983.
5. J Bey. Tetrahedral grid refinement. *Computing*, 55(4):355–378, 1995.
6. A Liu and B Joe. Quality local refinement of tetrahedral meshes based on 8-subtetrahedron subdivision. *Mathematics of Computation*, 65(215):1183–1200, 1996.
7. S Zhang. Successive subdivisions of tetrahedra and multigrid methods on tetrahedral meshes. *Houston J. Math*, 21(3):541–556, 1995.
8. MC Rivara. Algorithms for refining triangular grids suitable for adaptive and multigrid techniques. *International journal for numerical methods in Engineering*, 20(4):745–756, 1984.
9. E Bänsch. Local mesh refinement in 2 and 3 dimensions. *IMPACT of Computing in Science and Engineering*, 3(3):181–191, 1991.
10. A Liu and B Joe. Quality local refinement of tetrahedral meshes based on bisection. *SIAM Journal on Scientific Computing*, 16(6):1269–1291, 1995.
11. J Maubach. Local bisection refinement for n-simplicial grids generated by reflection. *SIAM Journal on Scientific Computing*, 16(1):210–227, 1995.
12. C Traxler. An algorithm for adaptive mesh refinement in *n* dimensions. *Computing*, 59(2):115–137, 1997.
13. D Arnold, A Mukherjee, and L Pouly. Locally adapted tetrahedral meshes using bisection. *SIAM Journal on Scientific Computing*, 22(2):431–448, 2000.
14. A Plaza and MC Rivara. Mesh refinement based on the 8-tetrahedra longest-edge partition. In *IMR*, pages 67–78, 2003.
15. M Neumüller and O Steinbach. A flexible space-time discontinuous galerkin method for parabolic initial boundary value problems. *Berichte aus dem Institut für Numerische Mathematik*, 2, 2011.
16. P. M. Knupp. Algebraic mesh quality metrics. *SIAM J. Numer. Anal.*, 23(1):193–218, 2001.
17. C Barber, D Dobkin, and H Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483, 1996.