BACHELORARBEIT

Conforming Bisection of Simplicial Triangulations in 3D

ausgeführt am

Institut für
Analysis und Scientific Computing
TU Wien

unter der Anleitung von

Univ.Prof. Dr. Dirk Praetorius

und

Dipl.-Ing. Michael Innerberger

durch

Iris Feldhammer
Matrikelnummer: 01525686
Öfnerstraße 68
8990 Bad Aussee

Wien, am 6. Oktober 2022

Danksagung

An dieser Stelle möchte ich mich bei allen Personen bedanken, die mich beim Verfassen dieser Arbeit unterstützt haben. Insbesondere gilt mein Dank meinen Betreuern Univ. Prof. Dirk Praetorius und Dipl.-Ing. Michael Innerberger für Ihre intensive Betreuung und all die investierte Zeit. Außerdem bedanke ich mich bei meinen Eltern, die mir dieses Studium ermöglicht haben und mir immer mit einem offenen Ohr zur Seite standen.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Bachelorarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 6. Oktober 2022

Iris Feldhammer

Contents

1 Introduction

The aim of this work is to introduce refinement rules for three-dimensional triangulations as well as to analyse their properties and limitations. Of the investigated properties, the most important one is shape-regularity of the resulting triangulations, i.e., how degenerated the resulting simplices are. We focus on the three-dimensional case and only consider conforming bisection. The main result of this thesis is the analysis of the refinement rule *oldest edge bisection* in Chapter 3. An implementation of this rule reveals that it does not produce shape-regular triangulations.

In Chapter 2, we begin with an introduction to triangulations and their refinement. We give the definitions of important notions concerning refinement, such as the definition of a conforming triangulation and a refinement rule. Moreover, we introduce a refinement algorithm that bisects an initial triangulation using a given refinement rule. This algorithm always produces conforming triangulations. However, it does not always terminate for all refinement rules or arbitrary triangulations. Thus, one has to ensure termination of the algorithm. In the subsequent chapters, we introduce different refinement rules which can be used for three-dimensional triangulations. For each rule, we investigate whether it leads to termination of the refinement algorithm. Additionally, there may be conditions imposed on the initial triangulation to ensure that the algorithm terminates.

In Chapter 3, we introduce a refinement rule called *oldest edge bisection*. This rule always bisects the edge that has been part of the element the longest. We show that this rule ensures termination of the refinement algorithm given in Chapter 2. However, it is unclear if this rule produces shape-regular triangulations. To check this, numerical experiments are performed. Additionally, it is tested whether different tie-breakers for edges of the same "age" during the refinement process influence the shape-regularity of the triangulation. The experiments reveal that *oldest edge bisection* does not produce shape-regular triangulations and that tie-breakers do not influence shape-regularity. To analyse this further, we introduce the definition of a patch index and show that mesh refinement can only act according to the patch index. With this result, we can see why the tie-breakers do not influence the resulting triangulation.

In Chapter 4, we analyse the properties of a refinement rule called *longest edge bisection*. With this rule, the longest edge of an element is considered as its refinement edge. Since the length of an edge is not a unique property, we introduce two ways to ensure uniqueness of the refinement edge. We show that using longest edge bisection with one of these additional rules also leads to termination of the refinement algorithm. Lastly, we give an overview of further results regarding longest edge bisection, especially with respect to shape-regularity.

In Chapter 5, we present another refinement rule called *newest vertex bisection*, which is defined for n-dimensional triangulations. This rule may not terminate for arbitrary triangulations. However, if the initial triangulation fulfils a certain condition, it terminates; see [Ste08]. This poses the question, whether it is possible to modify the rule such that termination is ensured for all arbitrary triangulations. For two dimensions, this was shown in [KPP13]. In [Sch17], the refinement rule was modified to ensure termination for arbitrary refinements in three dimensions. This is done by imposing an ordering of edges in the initial triangulation. We introduce the rule proposed in [Sch17] and all relevant definitions. We also take a closer look at the different types of elements that appear when using this refinement rule. Finally, we give an overview of the strategy that was used to show termination and note that the proof is similar to that of oldest edge bisection.

2 Notation and Preliminaries

The first section of this chapter introduces some notation and definitions regarding triangulations and their refinement. In the second section, a refinement algorithm is presented. The definitions and notations of this chapter are taken from [Ste08], which is formulated for n dimensions, and from [Sch17], which focuses on the three-dimensional setting.

2.1 Refinement of Triangulations

Definition 2.1 (n-Simplex)**.** Let $n, m \in \mathbb{N}$ with $2 \leq n \leq m$. We say that the $(n+1)$-tupel $T = [x_0, \ldots, x_n]$ is an n-simplex in $\mathbb{R}^m$, if $x_0, \ldots, x_n \in \mathbb{R}^m$ do not lie on an $(n-1)$-dimensional hyperplane. By $\overline{T} := \operatorname{conv}\{x_0, \ldots, x_n\}$, we denote the corresponding physical domain, which, by definition, is convex, compact and non-degenerated. The vectors x_i are called vertices of T. We denote by $\mathcal{V}(T) := \{x_0, \ldots, x_n\} \subseteq \mathbb{R}^m$ the set of vertices of T. We say that $S = [x'_0, \ldots, x'_k]$ is a k-subsimplex of T, if S is a k-simplex in $\mathbb{R}^m$ and $\{x'_0, \ldots, x'_k\} \subseteq \mathcal{V}(T)$. The 1-subsimplices are called edges of T. We denote by $\mathcal{E}(T)$ the set of edges of T. The $(n-1)$-subsimplices are called faces of T. We denote by $\mathcal{F}(T)$ the set of faces of T. Since edges $E \in \mathcal{E}(T)$ and faces $F \in \mathcal{F}(T)$ are simplices, we note that the physical domains $\overline{E}, \overline{F} \subseteq \mathbb{R}^m$ are well-defined.

Remark. Note that we have defined an n-simplex as a tupel (and not a set) of vectors, i.e., the order of the vertices is fixed.

Definition 2.2 (Triangulation)**.** Let $\Omega \subset \mathbb{R}^n$ be an open set. Then, $\mathcal{T}$ is called a triangulation of Ω if

- $\mathcal{T}$ is a finite set of n-simplices T in $\mathbb{R}^n$,
- $\mathcal{T}$ covers the closure of Ω, i.e., $\overline{\Omega} = \bigcup_{T \in \mathcal{T}} \overline{T}$,
- and the simplices in $\mathcal{T}$ are mutually essentially disjoint, i.e., for all $T, T' \in \mathcal{T}$ with $\overline{T} \neq \overline{T'}$ it holds that $|\overline{T} \cap \overline{T'}| = 0$, where $|\cdot|$ denotes the n-dimensional Lebesgue measure.

We denote the set of all vertices in a triangulation $\mathcal{T}$ by $\mathcal{V}(\mathcal{T}) := \bigcup_{T \in \mathcal{T}} \mathcal{V}(T)$, the set of all edges by $\mathcal{E}(\mathcal{T}) := \bigcup_{T \in \mathcal{T}} \mathcal{E}(T)$ and the set of all faces by $\mathcal{F}(\mathcal{T}) := \bigcup_{T \in \mathcal{T}} \mathcal{F}(T)$. The n-simplices $T \in \mathcal{T}$ are also called elements of the triangulation $\mathcal{T}$.

There exist various ways of generating triangulations consisting of n-simplices. In this work, we start with a given triangulation and aim to refine it by dividing elements into smaller pieces. We only consider refinement via bisection. Essentially, this means that an element is split in half resulting in two new elements. An exemplary bisection step of a single element $T = [x_0, x_1, x_2, x_3]$ is shown in Figure 2.1.

Definition 2.3 (Bisection)**.** Let $T = [x_0, \dots, x_n] \in \mathcal{T}$. The edge $r(T) := [x_0, x_n] \in \mathcal{E}(T)$ between the first and last vertex is called the refinement edge of T. When bisecting T, the midpoint $y = (x_0 + x_n)/2$ of the refinement edge $r(T)$ becomes a new vertex. This splits the element T into two sons $s_1(T)$ and $s_2(T)$, where $\mathcal{V}(s_1(T)) = \{x_0, \dots x_{n-1}, y\}$ and $\mathcal{V}(s_2(T)) = \{x_1, \dots, x_n, y\}$. Bisection of T results in a new triangulation $\mathcal{T}' = (\mathcal{T} \backslash \{T\}) \cup \{s_1(T), s_2(T)\}$.

During bisection, the refinement edge is split in half, turning it into two new edges $[x_0, y]$ and $[y, x_n]$. Additionally, another $n-1$ new edges are created to connect the vertex y with the other vertices of T.

Remark. Note that we only gave the sets of vertices of the son elements but not the elements itself. There are multiple ways the vertices in $s_1(T)$ and $s_2(T)$ can be arranged and we do not want to fix a specific order yet.

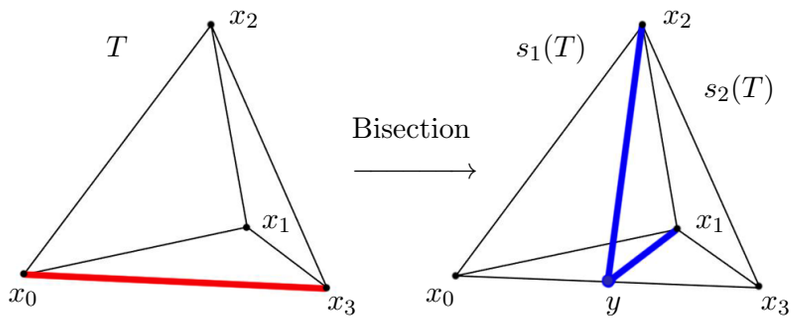


Figure 2.1: Bisection of an element $T = [x_0, x_1, x_2, x_3]$ leading to the sons $s_1(T)$ and $s_2(T)$ with $\mathcal{V}(s_1(T)) = \{x_0, x_1, x_2, y\}$ and $\mathcal{V}(s_2(T)) = \{x_1, x_2, x_3, y\}$. On the left, the refinement edge $[x_0, x_3]$ is coloured in red. On the right, after bisection, the new vertex y and the two new edges $[x_1, y]$ and $[x_2, y]$ are coloured in blue.

Remark. It is worth noting that bisection ensures that

$$|s_1(T)| = |s_2(T)| = \frac{|T|}{2} \tag{2.1}$$

for the corresponding volume.

We are especially interested in triangulations that are conforming in the sense of the next definition.

Definition 2.4 (Conforming Triangulation)**.** A triangulation $\mathcal{T}$ of $\Omega \subset \mathbb{R}^n$ is called conforming, if, for all $T, T' \in \mathcal{T}$ with $T \neq T'$, the intersection $\overline{T} \cap \overline{T'}$ is either empty or a lower-dimensional subsimplex of both simplices, i.e., there exists $2 \leq k \leq n$ and a k-tupel $S = [z_0, \ldots, z_k]$ such that $\mathcal{V}(S) = \mathcal{V}(T) \cap \mathcal{V}(T')$.

Conformity ensures that the intersection of two elements is a subsimplex that will be bisected the same way for both elements. An example for a conforming and a nonconforming triangulation is given in Figure 2.2.

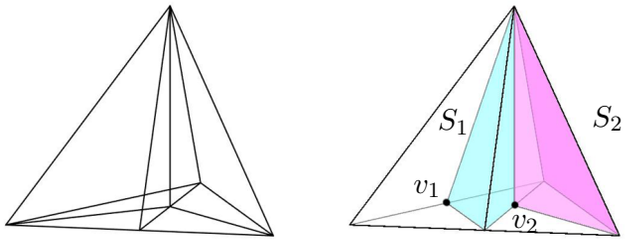


Figure 2.2: Example of a conforming triangulation on the left and a non-conforming triangulation on the right. On the right, two elements S_1 and S_2 are coloured. Considering the intersection of these elements $S_1 \cap S_2$, one can see that this is not a subsimplex of S_1. Therefore, the triangulation on the right is not conforming. Also, the vertex v_2 is not a vertex of the element S_1. This is called a hanging vertex. Connecting v_2 to the vertex v_1 of S_1, and thereby bisecting S_1, would result in a conforming triangulation.

Let $\mathcal{T}_0$ be a fixed initial triangulation of Ω. From now on, we will only consider triangulations $\mathcal{T}$ that are a refinement of $\mathcal{T}_0$. In the following, all elements are either part of the initial triangulation $\mathcal{T}_0$ or stem from bisecting an element $T_0 \in \mathcal{T}_0$ one or more times.

Definition 2.5 (Generation)**.** For elements $T_0 \in \mathcal{T}_0$, that are part of the initial triangulation, we define the generation as $\mathrm{gen}(T_0) := 0$. If an element T is bisected into sons $s_1(T)$ and $s_2(T)$, then $\mathrm{gen}(s_i(T)) := \mathrm{gen}(T) + 1$ for $i = 1, 2$.

Remark. Let T be an element that is generated by bisecting an initial element T_0 k times. Thus, there exists a sequence of elements $T_1, \ldots, T_k$, where T_j is a son of T_{j-1}, for all $j = 1, \ldots, k$, and $T_k = T$. Then, it holds that $\mathrm{gen}(T) = k$ if $|T| = 2^{-k}|T_0|$; see (2.1).

If we have an initial triangulation that is conforming, we are interested in maintaining this property also for its refinements.

Definition 2.6 (Edge patch)**.** Considering an edge $E \in \mathcal{E}(\mathcal{T})$, the set of all elements

containing E is called the edge patch of E:

$$\mathcal{T}(E) := \{T \in \mathcal{T} : \overline{E} \subset \overline{T}\}.$$

The edge patch $\mathcal{T}(E)$ is called suitable for compatible bisection or compatibly bisectable, if $\overline{E} = \overline{r(T)}$ for all $T \in \mathcal{T}(E)$, i.e., E is the refinement edge for all elements in the edge patch.

Definition 2.7 (Compatible Bisection)**.** Let $\mathcal{T}$ be a conforming triangulation and $E \in \mathcal{E}(\mathcal{T})$. The triangulation $\mathcal{T}'$ results from $\mathcal{T}$ by compatible bisection of the edge patch $\mathcal{T}(E)$, if $\mathcal{T}(E)$ is suitable for compatible bisection and

$$\mathcal{T}' = \big(\mathcal{T} \setminus \mathcal{T}(E)\big) \cup \big\{s_j(T) : T \in \mathcal{T}(E),\ j = 1, 2\big\}.$$

Thus, to form the refinement $\mathcal{T}'$, all elements in the edge patch $\mathcal{T}(E)$ are bisected, i.e., replaced by their sons. We note that the resulting triangulation is conforming.

There is usually more than one way to bisect an element and the choice of different refinement edges results in different triangulations. The following definition of a refinement rule allows us to specify the way we would like to bisect a given element.

Definition 2.8 (Refinement Rule)**.** Let $\mathbb{S} := \{T = [x_0, \dots, x_n] \subseteq \mathbb{R}^n : T \text{ is an } n\text{-simplex}\}$. A refinement rule $\mathfrak{R}$ is a mapping

$$\begin{aligned} \mathfrak{R} : \mathbb{S} &\to \mathbb{S}^2 \\ T &\mapsto (s_1(T), s_2(T)), \end{aligned}$$

where $s_1(T)$ and $s_2(T)$ are the sons of T resulting from a bisection step. Each simplex $T \in \mathbb{S}$ generates a unique binary tree with vertices in $\mathbb{S}$, called the bisection tree of T.

Remark. Note, that for each simplex $T \in \mathbb{S}$ its refinement edge $r(T) = [x_0, x_n]$ is already determined by the order of vertices in T. Thus, the vertices of its sons are also known. The refinement rule $\mathfrak{R}$ only has to define the order of the vertices in $s_1(T)$ and $s_2(T)$. In particular, $r(s_1(T))$ and $r(s_2(T))$ only depend on T and the choice of $\mathfrak{R}$.

Remark. We have defined a refinement rule $\mathfrak{R}$ as a function on the set of all simplices $\mathbb{S}$. In practice, knowledge of $\mathfrak{R}$ is only needed on the set of elements which are part of an initial triangulation $\mathcal{T}_0$ as well as the elements in their respective bisection trees.

Definition 2.9 (Uniform Refinement)**.** Let $\mathcal{T}$ be a triangulation with $\operatorname{gen}(T) = k$ for all $T \in \mathcal{T}$. Using a refinement rule $\mathfrak{R}$, we simultaneously bisect all elements in $\mathcal{T}$. The resulting (possibly non-conforming) triangulation $\mathcal{T}'$ is called a uniform refinement of $\mathcal{T}$. In $\mathcal{T}'$, all elements are of the same generation $k + 1$.

Remark. We have now seen two ways of refining a triangulation, namely compatible bisection and uniform refinement. When refining via compatible edge patch bisection, the edge that is considered has to be the refinement edge of all the elements it belongs to. This means it is bisected the same for all neighbouring elements. In particular, compatible edge bisection of a conforming triangulation always leads to a conforming triangulation. Unlike this, when uniformly refining a triangulation, all elements are bisected regardless of the way the refinement edges are arranged. We note that the uniform refinement of a conforming triangulation is not necessarily conforming, see Figure 5.6.

2.2 A Refinement Algorithm

In this section, we present an algorithm that refines an initial triangulation using a given refinement rule. It only uses compatible bisection steps. In particular, the resulting triangulations are conforming and the refinement rule does not create hanging vertices. We refer to the comments after Algorithm 1, but already note that Algorithm 1 may fail to terminate the outer loop (while the inner loop is always finite).

Algorithm 1 `refine`$(\mathcal{T}, \mathcal{M}, \mathfrak{R})$

1: $\mathcal{M} := \mathcal{M} \cap \mathcal{T}$
2: **repeat**
3: let $\mathcal{E}^*(\mathcal{M}) = \{r(T) : T \in \mathcal{M}\}$
4: **repeat**
5: let $\partial\mathcal{E}^*(\mathcal{M}) = \{E \in \mathcal{E}(\mathcal{T}) \setminus \mathcal{E}^*(\mathcal{M}) : \exists\, T \in \mathcal{T} : E = r(T) \text{ and } \mathcal{E}(T) \cap \mathcal{E}^*(\mathcal{M}) \neq \emptyset\}$
6: set $\mathcal{E}^*(\mathcal{M}) = \mathcal{E}^*(\mathcal{M}) \cup \partial\mathcal{E}^*(\mathcal{M})$
7: **until** $\partial\mathcal{E}^*(\mathcal{M}) = \emptyset$
8: let $\mathcal{R} = \{T \in \mathcal{T} : r(T) \in \mathcal{E}^*(\mathcal{M}) \text{ and } \mathcal{T}(r(T)) \text{ is suitable for compatible bisection.}\}$
9: set $\mathcal{T} = \big(\mathcal{T} \setminus \mathcal{R}\big) \cup \big\{s_j(T) : T \in \mathcal{R}, j = 1, 2\big\}$
10: set $\mathcal{M} = \mathcal{M} \setminus \mathcal{R}$
11: **until** $\mathcal{M} = \emptyset$

The input of Algorithm 1 consists of a conforming triangulation $\mathcal{T}$, a set of marked elements $\mathcal{M} \subseteq \mathcal{T}$ one wants to bisect, and a given refinement rule $\mathfrak{R}$. First, the set of marked edges $\mathcal{E}^*(\mathcal{M})$ is defined. It contains the refinement edges of all marked elements. An edge can belong to more than one element and it does not have to be the refinement edge for all of the elements it belongs to. This leads to the definition of the set $\partial\mathcal{E}^*(\mathcal{M})$ in line 4. It consists of all non-marked edges that are refinement edges for some elements T which have a marked edge. This means that, if an element's edge is marked and it is not the refinement edge of that element, the refinement edge belongs to the set $\partial\mathcal{E}^*(\mathcal{M})$. All the edges in $\partial\mathcal{E}^*(\mathcal{M})$ are marked additionally (line 5), i.e., added to the set of marked edges $\mathcal{E}^*(\mathcal{M})$. This inner loop always terminates, as there are only finitely many edges in the triangulation. The set $\mathcal{R}$ contains all elements which are suitable for compatible bisection, i.e., $\mathcal{R}$ contains all elements T, whose edge patch $\mathcal{T}(r(T))$ is compatibly bisectable (line 7). In line 9, the bisection step is carried out. All elements in $\mathcal{R}$ are refined and replaced by their son elements. For marked elements that have been bisected in this step, the marker is removed in line 10. The loop repeats for the remaining marked elements that have yet to be bisected.

Remark. In general, the outer loop of `refine` does not have to terminate. The triangulation $\mathcal{T}$ and the refinement rule could be given in a way that, at some point, there are no elements T with marked refinement edge such that $\mathcal{T}(r(T))$ is compatibly bisectable, i.e., $\mathcal{R}$ is empty before all marked elements have been bisected; see Figure 2.3 for a triangulation where this happens.

The following statement can be found in [Sch17, Remark 1.3.4].

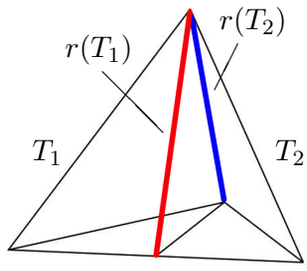


Figure 2.3: A triangulation $\mathcal{T} = \{T_1, T_2\}$ for which `refine` does not terminate. The refinement edge of T_1 is shown in red, the one of T_2 in blue.

Proposition 2.10. *Let $\mathcal{T}_0$ be an initial triangulation, $\mathfrak{R}$ a refinement rule and $\mathcal{T}$ a conforming refinement of $\mathcal{T}_0$. Let $\mathcal{M}$ be a set of marked elements such that* $\texttt{refine}(\mathcal{T}, \mathcal{M}, \mathfrak{R})$ *is well-defined, i.e., Algorithm 1 terminates. Then, for every $T \in \mathcal{M}$, it holds that*

$$\texttt{refine}\big(\mathcal{T}, \mathcal{M}, \mathfrak{R}\big) = \texttt{refine}\big(\texttt{refine}(\mathcal{T}, \mathcal{M}\backslash\{T\}, \mathfrak{R}), \{T\}, \mathfrak{R}\big),$$

where the right-hand side is well-defined.

3 Oldest Edge Bisection

In the previous section, we have given a general definition of the refinement rules. This section now gives an example for a possible refinement rule called oldest edge bisection. To determine the refinement edges in a triangulation, we first define an ordering of all edges. This ordering then determines how to bisect each element. Using this rule, we will be able to ensure that Algorithm 1 terminates.

Definition 3.1 ($\mathfrak{R}_{OE}$)**.** Let $\mathcal{T}_0$ be an initial triangulation where each edge $E_0 \in \mathcal{E}(\mathcal{T}_0)$ has a unique index $I(E_0) \in \mathcal{I} = \{1, \dots, \#\mathcal{E}(\mathcal{T}_0)\}$. The indices for refinements of $\mathcal{T}_0$ are defined inductively: Let $\mathcal{T}$ be a conforming refinement of $\mathcal{T}_0$ such that all edges $E \in \mathcal{E}(\mathcal{T})$ are already assigned an index $I(E)$. The refinement edge $r(T)$ of an element $T \in \mathcal{T}$ is then given as the edge with the smallest index,

$$r(T) = \operatorname*{arg\,min}_{E \in \mathcal{E}(T)} I(E).$$

For $E \in \mathcal{E}(\mathcal{T})$ let $\mathcal{T}(E)$ be compatibly bisectable. When $\mathcal{T}(E)$ is refined, let $E_1, \dots, E_N$ be the newly created edges. Let $\mathcal{N}_{\mathcal{T}} := \max_{E \in \mathcal{E}(\mathcal{T})} I(E)$. For the new edges, define $I(E_j) := \mathcal{N}_{\mathcal{T}} + j$, $j = 1, \dots, N$.

Remark. Using $\mathfrak{R}_{OE}$, the refinement edge of an element T is the edge E with the smallest index $I(E)$. This edge is also the one that has been part of T for the longest time, making it the "oldest" edge of the element, giving this rule the name oldest edge bisection.

Remark. In the definition of $\mathfrak{R}_{OE}$, the different indices for edges which are part of the initial triangulation $\mathcal{T}_0$ are assigned at random, with values from the set $\mathcal{I}$. However, the indices can also be assigned in such a way that they reflect a specific ordering of the edges, e.g., by length.

This refinement rule can be used to ensure the termination of Algorithm 1. Thus, when using $\mathfrak{R}_{OE}$ we can refine a given triangulation using only compatible bisection.

Proposition 3.2. *Let $\mathcal{T}_0$ be an initial triangulation, $\mathcal{T}$ a conforming refinement of $\mathcal{T}_0$ and $T \in \mathcal{T}$. Then the call* `refine`$(\mathcal{T}, \{T\}, \mathfrak{R}_{OE})$ *terminates.*

Proof. In general, the call `refine`$(\mathcal{T}, \mathcal{M}, \mathfrak{R}_{OE})$ terminates when each marked element $T \in \mathcal{M}$ has been bisected at least once. Therefore, we have to show that after a finite number of iterations the marked element T has been refined. Moreover, in each loop the set $\mathcal{R}$ has to be nonempty for the algorithm to continue. This means that there always has to be at least one element $\widetilde{T} \in \mathcal{T}$ with a compatibly bisectable edge patch $\mathcal{T}(r(\widetilde{T}))$. Initially, only the refinement edge of T is marked, i.e., added to the set of marked edges $\mathcal{E}^*(\mathcal{M})$ (line 3). If $\mathcal{T}(r(T))$ is a compatibly bisectable edge patch, we are able to bisect T and the algorithm terminates. Else, we have to mark the refinement edges of all elements in $\mathcal{T}(r(T))$. Again, if

one of those newly marked edges is not the refinement edge of all elements it belongs to, the refinement edges of those elements are also marked. We repeat this until all elements with a marked edge also have a marked refinement edge (line 5-6). The set of marked edges $\mathcal{E}^*(\mathcal{M})$ always contains an edge with a compatibly bisectable edge patch, namely the edge E with the smallest index $I(E)$. Since the smallest index is unique, this edge E is the refinement edge of all elements in $\mathcal{T}(E)$. Thus, the set $\mathcal{R}$ in line 8 is not empty and in each iteration of the outer loop at least one edge is bisected. During this bisection, new edges are generated which are assigned indices according to Definition 3.1. In particular, the indices of these newly generated edges are always larger than $\mathcal{N}_\mathcal{T}$. Thus, with each iteration of the loop, the oldest edge, i.e., the edge with the smallest index, is bisected and the overall values of the edge indices increase. There are only finitely many edges in the triangulation $\mathcal{T}$, hence there are only finitely many edges older than $r(T)$. Since the edges that are generated during bisection of $\mathcal{T}$ are always assigned higher indices, we have that after finitely many iterations $r(T)$ is the edge with the smallest index in $\mathcal{E}^*(\mathcal{M})$. Thus, we obtain that $\mathcal{T}(r(T))$ is a compatibly bisectable edge patch and the call $\texttt{refine}(\mathcal{T}, \{T\}, \mathfrak{R}_{OE})$ terminates. □

In the two-dimensional case, $\mathfrak{R}_{OE}$ produces the same triangulations as "newest vertex bisection". This is a strategy in which the refinement edge of an element is chosen as the one opposite to the newest vertex, the midpoint of the most recently bisected edge. We will look at this refinement rule in more detail in the next chapter.

Proposition 3.3. *For a triangulation $\mathcal{T}$ of $\Omega \subset \mathbb{R}^2$ and an element $T \in \mathcal{T}$ it holds that the refinement edge given by $\mathfrak{R}_{OE}$ is always the one opposite to the most recently generated vertex.*

Proof. Let $T := [x_0, x_1, x_2] \in \mathcal{T}$ be an element that is bisected along its refinement edge $r(T) = [x_0, x_2]$. The vertices of the sons of T are then given by $s_1(T) = \{x_0, x_1, y\}$ and $s_2(T) = \{x_1, x_2, y\}$, where y is the newly generated vertex. Since all newly generated edges contain the vertex y, the oldest edge in both $s_1(T)$ and $s_2(T)$ is the edge that does not contain y. The edge that does not contain the vertex y is also the edge opposite of that vertex. Therefore, the refinement edge of T using $\mathfrak{R}_{OE}$, which is given by the oldest edge of an element, coincides with the edge opposite to the newest vertex y. □

3.1 Numerical Experiments

In Proposition 3.2 we have shown that Algorithm 1 terminates when used with the refinement rule $\mathfrak{R}_{OE}$ from Definition 3.1. Since Algorithm 1 only uses compatible bisection, we also know that the resulting triangulation is always conforming. However, we do not know if oldest edge bisection produces shape-regular triangulations, as is given in the following definition.

Definition 3.4 (γ-shape-regular)**.** A triangulation $\mathcal{T}$ of $\Omega \subset \mathbb{R}^3$ is called γ-shape-regular for some constant $\gamma > 0$, if

$$\sigma(\mathcal{T}) := \max_{T \in \mathcal{T}} \frac{\mathrm{diam(T)}^3}{|T|} \leq \gamma < \infty$$

The following numerical experiments were performed to check for shape-regularity of triangulations produced with oldest edge bisection. According to Definition 3.1, during refinement using oldest edge bisection, all edges are given a unique index. First, all initial edges E_0 are given an index $I(E_0) \in \mathcal{I} = \{1, \dots, \#\mathcal{E}(\mathcal{T}_0)\}$. Edges that are newly generated during bisection are also assigned indices. This leads to the question, if the way these new indices are assigned, influences the shape-regularity of the resulting triangulation. Two different strategies of assigning indices were implemented.

- Strategy 1. The goal of the first strategy is to try to preserve shape-regularity by sorting the newly generated edges by length and assigning the lowest index to the longest edge. This way, the longer edges are the ones with the lower indices and therefore are bisected first.
- Strategy 2. The second strategy tries to reverse this idea. First, the lowest two indices are assigned to the two new edges that stem from bisecting the refinement edge. All other indices are sorted by length again, but assigning the lowest index to the shortest edge.

Algorithm 1 was implemented with an initial triangulation of Kuhn simplices, as can be seen in Figure 3.2. For every refinement step, about 5 percent of all elements were randomly chosen and marked for bisection. These elements were then bisected using the refinement rule $\mathfrak{R}_{OE}$. The marking of random elements and their subsequent refinement is then repeated until the number of elements in the triangulation reaches a given value. Finally, the shape regularity constant $\sigma(\mathcal{T})$ can be computed. The results can be seen in Figure 3.1.

3.2 Implementation

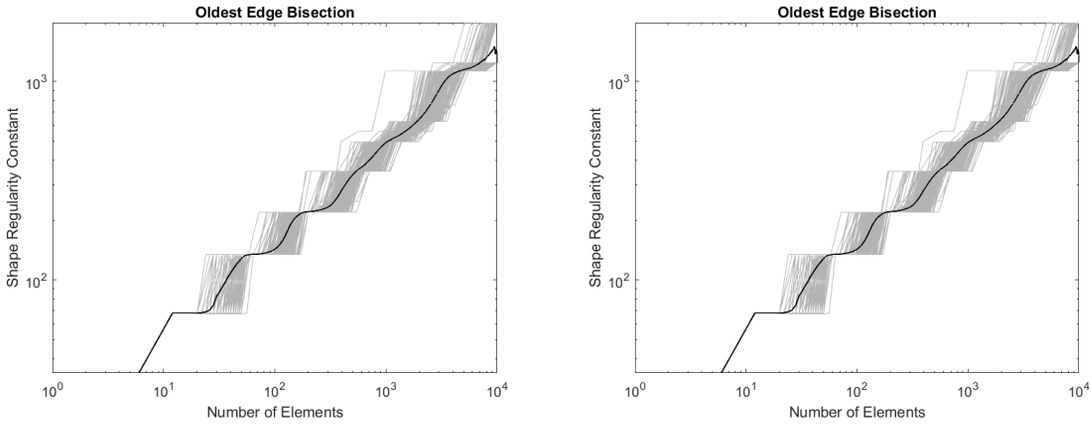
The following data structures were used.

`coordinates` $\in \mathbb{R}^{N_V \times 3}$, where N_V is the number of vertices. Each row contains the three-dimensional coordinates of a vertex. The vertex in the i-th row is given the index i.

`elements` $\in \mathbb{N}^{N_T \times 4}$, where N_T is the number of elements. The i-th row stores the i-th element. Each row in `elements` contains the 4 indices of the vertices that belong to the element, where the vertex indices are given by the row number in `coordinates`.

`edges` $\in \mathbb{N}^{N_E \times 2}$, where N_E is the number of edges. The i-th row stores the i-th edge. Each row contains the two indices of the vertices belonging to the edge, where the vertex indices are given by the row number in `coordinates`.

`element2edges` $\in \mathbb{N}^{N_T \times 6}$ stores the edges that belong to an element. The i-th row stores the edges belonging to the i-th element. Each row contains the six indices of the edges belonging to the element, where the edge indices are given by the row number in `edges`. The first edge in each row vector is the refinement edge of the element, which is given by



(a) Strategy 1 (b) Strategy 2

Figure 3.1: Result of numerical experiments. After each refinement step, the number of elements in the triangulation and the shape-regularity constant $\sigma(\mathcal{T})$ of that triangulation $\mathcal{T}$ was computed. For each strategy, the whole refinement process was repeated 200 times with different randomly marked elements with predetermined seed (plotted in grey). The mean of all results is plotted in black. Both strategies create the same triangulations, see Corollary 3.7. Thus, the different strategies do not influence shape-regularity. We can see that the refinement rule $\mathfrak{R}_{OE}$ does not produce shape-regular triangulations.

the edge between the first and fourth vertex of the element.

`age` $\in \mathbb{N}^{N_E}$ stores global edge indices that show how long an edge has been part of the triangulation, i.e., the "age" of the edge. The index of the i-th edge is stored in the i-th row. At the beginning, the refinement edges are assigned the value 0, all other edges are assigned unique ascending indices. During bisection, newly generated edges are assigned higher indices depending on the strategy that is used.

The values in `age` correspond to the edge indices $I(E)$ from Definition 3.1. With the exception that in `age` all initial refinement edges are given the index 0 for implementational reasons. Because they are bisected first, their edge indices do not further influence any refinements.

During refinement, the marked elements are chosen randomly. To ensure repeatability, this was done by specifying seeds for the random number generator that marks elements. This guarantees that we are able to perform the same refinements with both strategies. In the k-th simulation, a vector of random numbers $\{k_i \mid i \in \mathbb{N}\}$ is generated with seed k. During a simulation, Algorithm 1 is called repeatedly, until the desired number of elements in the triangulation is reached. In the experiments, this was done until the number of elements reached 10000. For the ℓ-th run of Algorithm 1, the marked elements are then chosen randomly with seed k_ℓ.

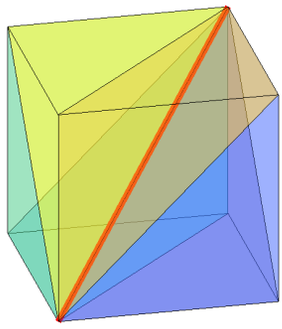


Figure 3.2: Initial triangulation of the unit cube $[0,1]^3$ into Kuhn simplices. The refinement edge is marked red.

3.3 Interpretation of Results

The numerical experiments have shown that the way indices are assigned to new edges during bisection of an edge patch does not influence the resulting triangulation. This means, that edges that were generated during bisection of the same edge patch are all part of different elements by the time they are the oldest edge of an element. Thus, when an edge is the oldest edge in an element, the element does not contain any edge that was generated during bisection of the same edge patch as the oldest edge. To explore this further, we give the definition of a patch index.

Definition 3.5 (Patch Index)**.** Let T be an element of a triangulation $\mathcal{T}$, $r(T)$ its refinement edge (and therefore the edge with the lowest index) and $\mathcal{P} = \mathcal{T}(r(T))$ the edge patch of the refinement edge. Furthermore, let $\mathcal{N}_\mathcal{T} = \max_{E\in\mathcal{E}(\mathcal{T})} \mathcal{I}(E)$ be the value of the highest edge index in $\mathcal{T}$. Let $E_1, \dots, E_N$ be the set of edges generated by bisection of the edge patch $\mathcal{P}$. For all edges $E_i, i = 1, \dots, N$, we define their patch index $p(E_i)$ as $p(E_i) = \mathcal{N}_\mathcal{T} + 1$. Thus, all edges generated by the bisection of $r(T)$ are assigned a patch index, which is given by the index of the youngest edge in the triangulation. Edges E that are part of the initial triangulation are assigned patch index $p(E) = 0$.

Proposition 3.6 (Three Types of Patch Index Distributions)**.** *Let $T \in \mathcal{T}$ be an element and denote the patch indices of its edges by $p(T) = [a\ b\ c\ d\ e\ f]$ with $a \le b \le c \le d \le e \le f$. For elements that appear during refinement with oldest edge bisection, $p(T)$ takes one of the following three forms:*

(i) $p(T) = [\,a\ a\ a\ a\ a\ a\,]$, *if* $\mathrm{gen}(T) = 0$,

(ii) $p(T) = [\,a\ a\ a\ b\ b\ b\,]$ *with* $a < b$, *if* $\mathrm{gen}(T) = 1$,

(iii) $p(T) = [\,a\ b\ b\ c\ c\ c\,]$ *with* $a < b < c$, *if* $\mathrm{gen}(T) \ge 2$.

The bisection steps that appear in the following proof are shown in Figures 3.3-3.5. Because of symmetry and for clarity, we only consider one son element for each bisection step.

Proof. Step 1: According to Definition 3.5, all edges $E \in \mathcal{E}(\mathcal{T}_0)$ in the initial triangulation $\mathcal{T}_0$ have the same patch index $p(E) = a = 0$. Thus, it holds that $p(T_0) = [\, a\ a\ a\ a\ a\ a\,]$ for all initial elements $T_0 \in \mathcal{T}_0$. According to Definition 2.5, for initial elements it holds that $\text{gen}(T_0) = 0$.

Step 2: Let T_0 be an initial element. Thus it is of the form $p(T_0) = [\, a\ a\ a\ a\ a\ a\,]$. When bisecting T_0, four new edges are generated, which all have the same patch index $b > a$. Two edges with patch index b lie on the former refinement edge and therefore are distributed to the two sons $s_1(T_0)$ and $s_2(T_0)$. The two edges connecting the new vertex y with the two vertices that are not part of the refinement edge are shared between sons. Hence, bisecting an element of the form $p(T_0) = [\, a\ a\ a\ a\ a\ a\,]$ always results in two elements $s_1(T_0)$ and $s_2(T_0)$ of the form $p(s_i(T)) = [\, a\ a\ a\ b\ b\ b\,]$, $i = 1, 2$. This is shown in Figure 3.3. Since $\text{gen}(T_0) = 0$, it holds that $\text{gen}(s_1(T_0)) = \text{gen}(s_2(T_0)) = 1$.

Step 3: Let T be an element with $\text{gen}(T) = 1$. Thus, it is of the form $p(T) = [\, a\ a\ a\ b\ b\ b\,]$. For T, there are three possible refinement edges, namely the edges with patch index a. During refinement, one of the edges with patch index a is bisected and each son element $s_1(T)$ and $s_2(T)$ contains three new edges with patch index c. Therefore, the sons of T are of the form $p(s_i(T)) = [\, a\ b\ b\ c\ c\ c\,]$, $i = 1, 2$. This is shown in Figure 3.4. It holds that $\text{gen}(s_1(T)) = \text{gen}(s_2(T)) = 2$.

Step 4: Let T' be an element with $\text{gen}(T') = k \geq 2$ of the form $p(T') = [\, a\ b\ b\ c\ c\ c\,]$. The refinement edge in T' is the edge with the smallest patch index a, since it is also the edge with the smallest edge index. During bisection of T', the newly generated edges receive the patch index d. Hence, bisecting T' results in two elements of the form $p(s_i(T')) = [\, b\ c\ c\ d\ d\ d\,]$, $i = 1, 2$. This is shown in Figure 3.5. Again, it holds that $\text{gen}(s_1(T')) = \text{gen}(s_2(T')) = k + 1$.

□

Corollary 3.7. *Let $\mathcal{T}$ be a triangulation that was refined using $\mathfrak{R}_{OE}$ and let $T \in \mathcal{T}$ with $\text{gen}(T) \geq 2$. Then, the refinement edge of T is given by the unique edge with the smallest patch index.* □

Remark. From Corollary 3.7, we can conclude that during refinement with $\mathfrak{R}_{OE}$, the order in which indices are assigned to newly generated edges does not influence the resulting triangulation.

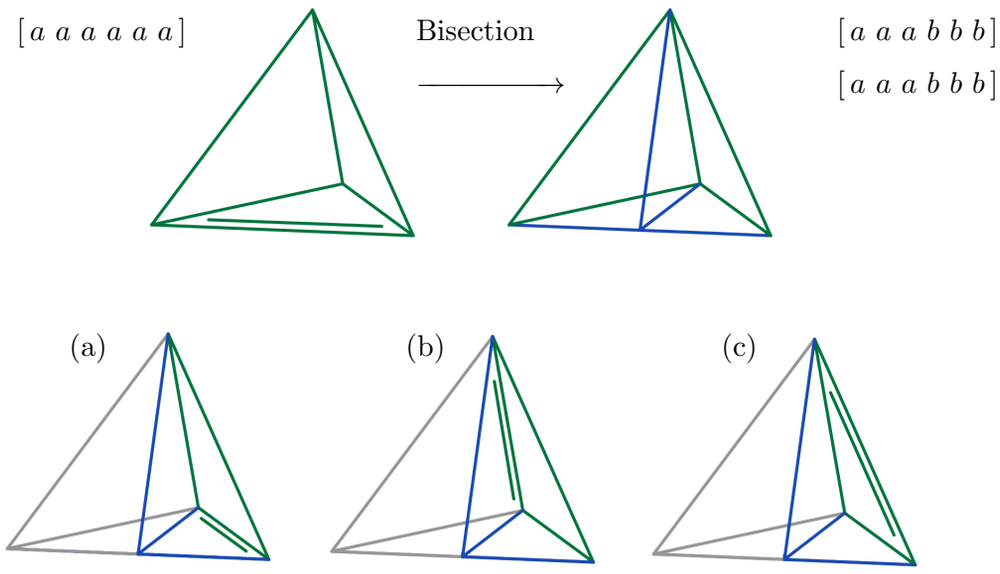


Figure 3.3: Refinement of an initial element. The refinement edges are indicated by a double line. All initial edges are coloured green and have patch index $a = 0$. After one bisection step, four new edges are generated (coloured blue), they all have a patch index of b with $b > a$. By symmetry, this sketch covers all possible cases. In both son elements, there are three possible candidates for refinement edges. For the right element, these three possibilities (a)-(c) are shown in the second row.

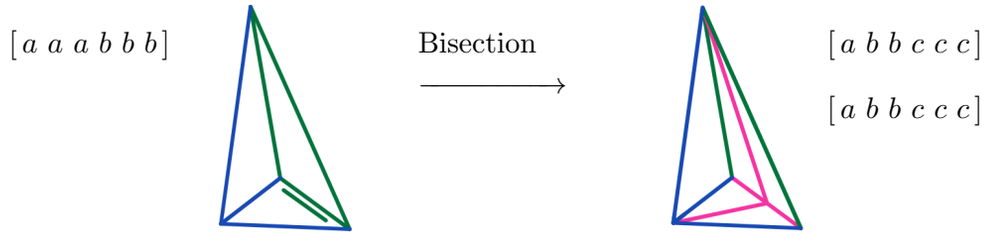


Figure 3.4: Second bisection step of an initial element as given in Figure 3.3. The green initial edges are assigned the patch index $a = 0$ and the newer blue edges have a patch index of b where $b > a$. After the second bisection step, the newly generated edges (coloured pink) have a patch index c where $c > b > a$. We can see that, after the second bisection step for the son elements, there is only one possible choice of refinement edge. By symmetry, this sketch covers all possible cases.

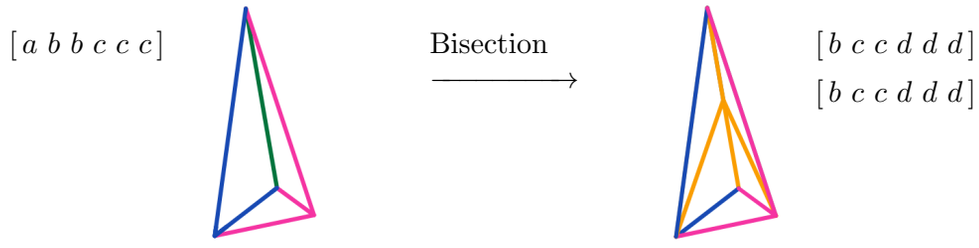


Figure 3.5: Third bisection step of an initial element as given in Figure 3.3. Here the newly generated edges (coloured orange) have a patch index of d. Both of the resulting son elements contain one blue edge with patch index b, two pink edges with patch index c and three orange edges with patch index d. Hence, they are again of the same type $[\, b\ c\ c\ d\ d\ d\,]$. By symmetry, this sketch covers all possible cases.

4 Longest Edge Bisection

This chapter discusses another variant of bisection, namely longest edge bisection. Using this rule, the longest edge of an element is chosen as its refinement edge.

Definition 4.1 (Refinement rule $\mathfrak{R}_{LE}$)**.** Using the refinement rule $\mathfrak{R}_{LE}$, the refinement edge of an element $T \in \mathcal{T}$ satisfies that

$$|r(T)| = \max_{E \in \mathcal{E}(T)} |E|.$$

Remark. In the case of $\mathfrak{R}_{LE}$, for an element T and its refinement edge $r(T) = [x_0, x_n]$, the order of the nodes x_0 and x_n is not relevant, since we are only interested in the length of the edge.

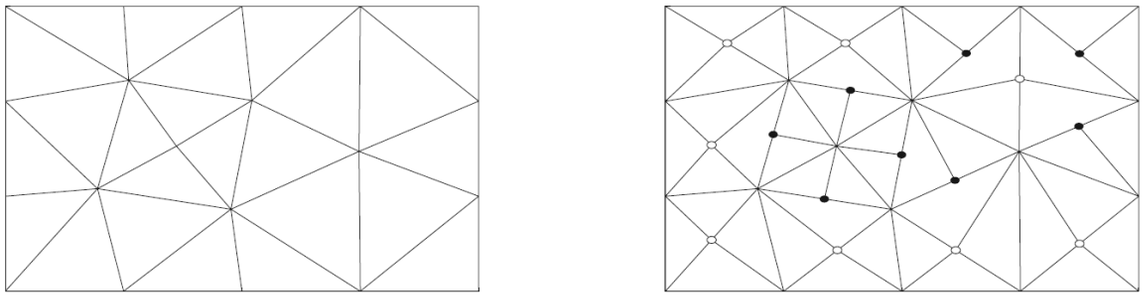


Figure 4.1: On the left, a conforming initial triangulation $\mathcal{T}_0$ is given. Uniform refinement of $\mathcal{T}_0$ using $\mathfrak{R}_{LE}$ results in the non-conforming triangulation on the right. Hanging vertices are coloured black [KPS16].

A summary of known results regarding longest edge bisection, as well as open problems, can be found in [KPS16]. In particular, we want to highlight the following results regarding shape-regularity. In the two-dimensional case, it was shown in [Adl83] that during the bisection process of a triangle using $\mathfrak{R}_{LE}$, the resulting triangles in its bisection tree fall into finitely many similarity classes. In the three-dimensional case, the analysis is more complex, thus most of the results regarding longest edge bisection are limited to the two-dimensional case. The shape-regularity of longest edge bisection in 3D is analysed numerically in [HKK14], but the mathematical proof of shape-regularity is still an open problem. However, numerical tests in [HKK14] indicate that their bisection algorithm using longest edge bisection produces shape-regular families of 3D-triangulations.

As we were able to prove for $\mathfrak{R}_{OE}$, we would like to show that Algorithm 1 terminates when used with the refinement rule $\mathfrak{R}_{LE}$. However, in general, longest edge bisection is non-unique if the simplex $T \in \mathcal{T}$ has multiple longest edges, i.e., $\#\{E \in \mathcal{E}(T) \mid |E| =$

$\max_{E' \in \mathcal{E}(T)} |E'| \} > 1$. Without a unique refinement edge, it is possible that the set $\mathcal{R}$ in line 8 of Algorithm 1 is empty, as we cannot always find an edge that is the refinement edge of all elements it belongs to. Thus, the call does not terminate. An example of such a case can be seen in Figure 4.2. We propose two ways of ensuring uniqueness of the refinement edge and prove that this leads to the termination of Algorithm 1.

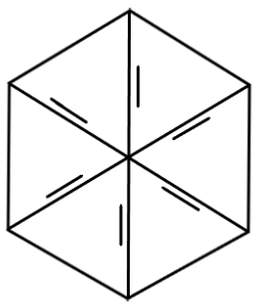


Figure 4.2: A conforming two-dimensional triangulation $\mathcal{T}$ in which the refinement edge of each element is marked with a line. In $\mathcal{T}$, the longest edge of an element is not unique and, thus, the refinement edge can be chosen such that there is no edge patch that is suitable for compatible bisection. Hence Algorithm 1 does not terminate for this triangulation.

The first way of ensuring termination is similar to the method described in Chapter 3.

Definition 4.2 ($\mathfrak{R}'_{LE}$)**.** When using the refinement rule $\mathfrak{R}_{LE}$, define edge indices as given in Definition 3.1. If for some element $T \in \mathcal{T}$ it holds that

$$\#\{E \in \mathcal{E}(T) \mid |E| = \max_{E' \in \mathcal{E}(T)} |E'| \} > 1,$$

i.e., T has multiple longest edges, the edge with the smallest index is chosen as the unique refinement edge.

Proposition 4.3. *Let $\mathcal{T}_0$ be a conforming initial triangulation, $\mathcal{T}$ a conforming refinement of $\mathcal{T}_0$ and $T \in \mathcal{T}$. Then, the call* `refine`$(\mathcal{T}, \{T\}, \mathfrak{R}'_{LE})$ *terminates.*

Proof. The proof follows from arguments similar to that of Proposition 3.2. To show termination of the call `refine`$(\mathcal{T}, \{T\}, \mathfrak{R}'_{LE})$, at some point the edge patch $\mathcal{T}(r(T))$ has to be compatibly bisectable. Furthermore, in each iteration of the outer loop in Algorithm 1, the set $\mathcal{R}$ has to be non-empty. Thus, in each iteration there has to be at least one marked edge with a compatibly bisectable edge patch. First, the refinement edge of T is marked. If $r(T)$ has a compatibly bisectable edge patch, we are able to bisect our marked element T and the call terminates. Otherwise, there has to be at least one element that contains the edge $r(T)$ but does not have $r(T)$ as refinement edge, i.e., that element has an edge that is longer than $r(T)$ or it has an edge of the same length as $r(T)$ but with a smaller index. We mark the refinement edges of all such elements. Again, now there might exist elements with a marked edge that is not their refinement edge, thus we repeat this step until all elements with a marked edge also have a marked refinement edge. To be able to continue with the call, the set of marked edges $\mathcal{E}^*(\mathcal{M})$ now has to contain an edge with a

compatibly bisectable edge patch. This is ensured by $\mathfrak{R}'_{LE}$. Of all the edges in $\mathcal{E}^*(\mathcal{M})$, we first choose the edge with the greatest length. If there are multiple longest edges, we choose the one with the smallest index, calling it E. If there was an edge that is part of the same element as E and is longer or has the same length and a smaller index than E, it would be the refinement edge of that element. However, then it would have also been marked and thus be part of $\mathcal{E}^*(\mathcal{M})$. Thus, E has to be the refinement edge of all elements it belongs to and we can always find an edge in $\mathcal{E}^*(\mathcal{M})$ with a compatibly bisectable edge patch. Therefore, the set $\mathcal{R}$ in line 8 is not empty and in each iteration of the outer loop at least one edge is bisected. Since we always choose the longest edge with smallest index as the refinement edge for bisection (which halves the edge's length), at some point, $r(T)$ will be the geometrically longest edge with smallest index in $\mathcal{E}^*(\mathcal{M})$. Then, we obtain that $\mathcal{T}(r(T))$ is a compatibly bisectable edge patch and the call $\texttt{refine}(\mathcal{T}, \{T\}, \mathfrak{R}'_{LE})$ terminates. □

The second way of ensuring termination uses the location of edges.

Definition 4.4 ($\mathfrak{R}''_{LE}$)**.** Let $\mathcal{T}$ be a triangulation of $\Omega \subset \mathbb{R}^n$ that is bisected using the refinement rule $\mathfrak{R}_{LE}$. Let $T \in \mathcal{T}$ be an element such that $\#\{E \in \mathcal{E}(T) \mid |E| = \max_{E' \in \mathcal{E}(T)} |E'|\} = k$, i.e. T has k edges $E_1, \dots, E_k$ of the same maximal length. Let $m_{E_1} = (x_1^1, \dots, x_n^1), \dots, m_{E_k} = (x_1^k, \dots, x_n^k)$ be the coordinates of the midpoints of these longest edges. Then, the unique refinement edge is chosen as the longest edge with the lexicographically smallest coordinates of its midpoint.

Proposition 4.5. *Let $\mathcal{T}_0$ be a conforming initial triangulation, $\mathcal{T}$ a conforming refinement of $\mathcal{T}_0$ and $T \in \mathcal{T}$. Then, the call* $\texttt{refine}(\mathcal{T}, \{T\}, \mathfrak{R}''_{LE})$ *terminates.*

Proof. The proof is similar to that of Proposition 4.3. To ensure termination of Algorithm 1 using $\mathfrak{R}''_{LE}$, we need to know that the edge patch of $r(T)$ will be compatibly bisectable. Also, for the call to continue, the set of marked edges $\mathcal{E}^*(\mathcal{M})$ always has to contain an edge with a compatibly bisectable edge patch. Such an edge has to be the refinement edge of all the elements it is part of. To find it, we look at the longest edges in $\mathcal{E}^*(\mathcal{M})$. If there is a unique longest edge E in $\mathcal{E}^*(\mathcal{M})$, it has to be the refinement edge of all elements in its patch. If an element in the patch had a different refinement edge, it would have to be longer than E, since all the refinement edges of the patch are also in the set $\mathcal{E}^*(\mathcal{M})$. However, then E would not be the longest edge of the set anymore. Therefore, the edge patch of E is suitable for compatible bisection. If there are multiple longest edges in $\mathcal{E}^*(\mathcal{M})$, the refinement edge, according to $\mathfrak{R}''_{LE}$, is given by the longest edge with the lexicographically smallest coordinates of its midpoint. This edge then is the unique refinement edge, since no two midpoints can have the same coordinates. Since bisection halves the length of the refinement edge, at some point, $r(T)$ will be the longest edge with the lexicographically smallest coordinates of its midpoint. □

5 A Refinement Rule for Arbitrary Three-Dimensional Triangulations

We have seen in the previous chapters that the algorithm `refine` does not terminate for arbitrary triangulations and refinement rules. In this chapter, we will introduce two refinement rules. The first section analyzes a rule which leads to the termination of `refine` under a special condition on the initial triangulation $\mathcal{T}_0$. This raises the question of whether there exists a refinement rule which assures the termination of `refine` without a restriction on the initial partition. That this is indeed possible for $n = 2$ was shown in [KPP13]. For the three-dimensional case, such a result was presented in [Sch17] and we will analyze this refinement rule in the second section.

5.1 Refinement of Triangulations with an Initial Ordering

In this section, we will present a refinement rule, which is known as newest vertex bisection, and examine its properties. It was introduced for n-dimensional simplices in [Mau95] and [Tra97]. Further analysis of the n-dimensional case was given in [Ste08], which is what we will present in this section.

Definition 5.1 (Type)**.** In the following, each simplex $T = [x_0, \dots, x_n]$ will be assigned a type $\gamma(T) \in \{0, \dots, n-1\}$, denoted by $T = [x_0, \dots, x_n]_\gamma$.

Definition 5.2 (Refinement Rule $\mathfrak{R}_{NVB}$)**.** Let $T = [x_0, \dots, x_n]_\gamma$. Then, the son elements of T are defined as

$$\begin{aligned} s_1(T) &:= [x_0, y, x_1, \dots, x_\gamma, x_{\gamma+1}, \dots, x_{n-1}]_{(\gamma+1) \bmod n}, \\ s_2(T) &:= [x_n, y, x_1, \dots, x_\gamma, x_{n-1}, \dots, x_{\gamma+1}]_{(\gamma+1) \bmod n}, \end{aligned}$$

where $y = (x_0 + x_n)/2$ is the newly generated node being the midpoint of $r(T) = [x_0, x_n]$.

Remark. In Definition 5.2, the cases $\gamma(T) = 0$ and $\gamma(T) = n - 1$ require additional explanation.
If $\gamma(T) = 0$, the subsequence $(x_1, \dots, x_\gamma)$ should be read as void, resulting in

$$\begin{aligned} s_1(T) &= [x_0, y, x_1, \dots, x_{n-1}]_1, \\ s_2(T) &= [x_n, y, x_{n-1}, \dots, x_1]_1. \end{aligned}$$

If $\gamma(T) = n - 1$, the subsequence $(x_{\gamma+1}, \dots, x_{n-1})$ should be read as void, resulting in

$$\begin{aligned} s_1(T) &= [x_0, y, x_1, \dots, x_{n-1}]_0, \\ s_2(T) &= [x_n, y, x_1, \dots, x_{n-1}]_0. \end{aligned}$$

Example. We focus on the 3D case. Let $T = [x_0, x_1, x_2, x_3]_\gamma$. Depending on its type $\gamma(T)$, bisecting T using $\mathfrak{R}_{NVB}$ yields the sons

- $\gamma(T) = 0$:
  $s_1(T) = [x_0, y, x_1, x_2]_1, \quad s_2(T) = [x_3, y, x_2, x_1]_1,$
- $\gamma(T) = 1$:
  $s_1(T) = [x_0, y, x_1, x_2]_2, \quad s_2(T) = [x_3, y, x_1, x_2]_2,$
- $\gamma(T) = 2$:
  $s_1(T) = [x_0, y, x_1, x_2]_0, \quad s_2(T) = [x_3, y, x_1, x_2]_0.$

Remark. Note that the type of an element T determines the refinement edge of its sons. For these son elements, we again know the type and, hence, the way its sons are refined. Thus, in a way, by defining a type $\gamma(T)$ the refinement rule is already "built-in", i.e., knowing the type of an element, we also know how it has to be refined. An example can be seen in Figure 5.1.

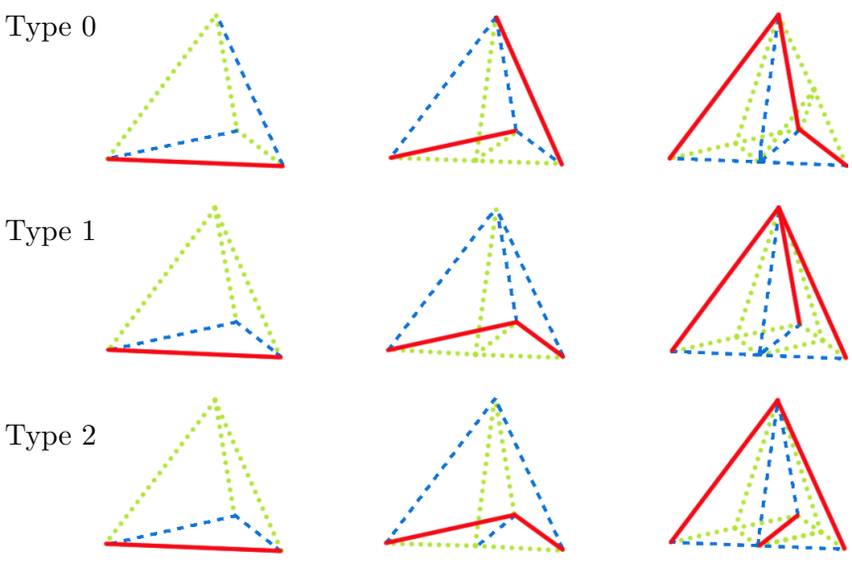


Figure 5.1: Refinement of elements of type 0, 1 and 2. The colouring of the edges symbolizes the order in which they are bisected. Red edges are the refinement edges of the first bisection, blue-dashed edges are bisected in the second refinement step and green-dotted edges in the third step. In the first row, we start with an element of type 0, its sons then both have type 1 and another uniform refinement step leads to four elements of type 2. In the second and third row, we can see how the refinement steps look if we start with an element of type 1 or 2. Note that the difference between type 1 and type 2 is the choice of refinement edges in the second step.

Definition 5.3 (Partner Element)**.** Rearranging the order of the vertices in an element $T = [x_0, \dots, x_n]_\gamma$, we get the corresponding partner element

$$T_P := [x_n, x_1, \dots, x_\gamma, x_{n-1}, \dots, x_{\gamma+1}, x_0]_\gamma \,,$$

which results in the same sons as T, when refined via $\mathfrak{R}_{NVB}$. Thus, T and its partner element T_P generate the same bisection tree.

Example. Let $T = [x_0, x_1, x_2, x_3]_\gamma$ be an element in a three-dimensional triangulation, then its partner element is

$$T_P = \begin{cases} [x_3, x_2, x_1, x_0]_0 & \text{if } \gamma = 0, \\ [x_3, x_1, x_2, x_0]_1 & \text{if } \gamma = 1, \\ [x_3, x_1, x_2, x_0]_2 & \text{if } \gamma = 2. \end{cases}$$

By carrying out a refinement step of $\mathfrak{R}_{NVB}$, one can see that both elements T and T_P lead to the same sons.

Definition 5.4 (Neighbours)**.** Two elements $T, T' \in \mathcal{T}$ are called neighbours, or neighbouring elements, if $\overline{T} \cap \overline{T'} \in \mathcal{F}(\mathcal{T})$.

Definition 5.5 (Reflected Neighbours)**.** Two neighbouring elements $T, T' \in \mathcal{T}$ are called reflected neighbours, if $\gamma(T) = \gamma(T')$ and if the ordered sequences of vertices of T, or of its partner element T_P, coincide with that of T' in all but one position.

It can be shown that $\mathfrak{R}_{NVB}$ terminates under the following conditions on the initial triangulation $\mathcal{T}_0$. We will refer to this as initial ordering condition (IO):

(IO1) $\mathcal{T}_0$ is conforming and all elements are of the same type γ.

(IO2) For all neighbouring elements $T = [x_0, \dots, x_n]_\gamma$ and $T' = [x'_0, \dots, x'_n]_\gamma$ in $\mathcal{T}_0$ there holds: If $\overline{r(T)} \subseteq \overline{T} \cap \overline{T'}$ or $\overline{r(T')} \subseteq \overline{T} \cap \overline{T'}$, they are reflected neighbours. Otherwise, the pair of neighbouring sons of T and T' are reflected neighbours.

In two dimensions, one can always find an initial ordering and assignment of types of elements such that condition (IO) is satisfied. This is shown in [BDD04, Lemma 2.1]. For $n > 2$ this is not the case. As seen in [Sch17, Lemma 1.7.14], there exist triangulations for which no such initial ordering is possible. However, in [Ste08, Appendix A] a method is proposed which can refine conforming triangulations in such a way that they satisfy condition (IO). Finally, with the condition (IO) being fulfilled, one can show that every uniform refinement is conforming, see [Ste08, Theorem 4.3]. This observation is key to prove that Algorithm 1 terminates.

5.2 Refinement of Arbitrary Initial Triangulations

In this section, we will present a refinement rule which ensures the termination of the refinement algorithm from Section 2.2 for arbitrary initial triangulations $\mathcal{T}_0$. For two-dimensional

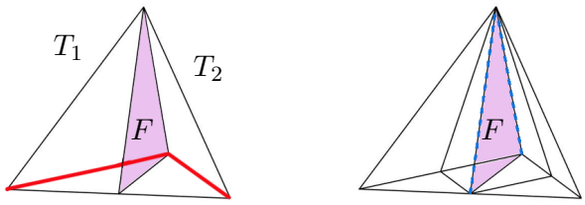


Figure 5.2: On the left, the face F is part of two elements T_1 and T_2, which have refinement edges marked in red. On the right, after uniform bisection, the face F contains the refinement edge of a son of T_1 as well as the refinement edge of a son of T_2, both coloured blue and dashed. The refinement edge of F is therefore not unique.

triangulations such a refinement rule was presented in [KPP13]. We will focus on the three-dimensional case, where a refinement strategy was given in [Sch17, Definition 1.8.4]. It is based on an algorithm presented in [AM99, Section 3] and also uses the refinement rule $\mathfrak{R}_{NVB}$ from the previous section. We begin by giving some additional definitions.

Definition 5.6 (Refinement Edges of Faces)**.** Let $\mathcal{T}$ be a conforming triangulation of $\Omega \subset \mathbb{R}^3$ and $F \in \mathcal{F}(\mathcal{T})$. Let T_1 and T_2 be two elements in $\mathcal{T}$ with $\overline{T_1} \cap \overline{T_2} = \overline{F}$. In the case that F is part of the boundary of $\mathcal{T}$ and thus only part of one element T_1, define $T_1 = T_2$. Let $\mathfrak{R}$ be a refinement rule, and let $s_1(T_i)$ and $s_2(T_i)$ be the sons of the element T_i, $i = 1, 2$, given by $\mathfrak{R}$. The set of refinement edges $r(F)$ of F then depends on the refinement edges of T_1 and T_2 and their sons.

- If $r(T_i) \in \mathcal{E}(F)$, then $r(T_i) \in r(F)$.
- If $r(T_i) \notin \mathcal{E}(F)$ and $r(s_1(T_i)) \in \mathcal{E}(F)$, then $r(s_1(T_i)) \in r(F)$.
- If $r(T_i) \notin \mathcal{E}(F)$ and $r(s_2(T_i)) \in \mathcal{E}(F)$, then $r(s_2(T_i)) \in r(F)$.

Remark. Note that, according to this definition, there are possibly multiple refinement edges of a face $F \in \mathcal{F}(\mathcal{T})$. If F is part of two elements T_i, $i = 1, 2$, they do not necessarily both lead to the same refinement edge for F. An example for a situation where a face does not have a unique refinement edge is given in Figure 5.2.

In the following, we are only interested in the case that a face F has a unique refinement edge, as this ensures that F will be refined in the same way for both of the elements it is part of. This motivates the following definition.

Definition 5.7 (Conformingly Marked)**.** A conforming triangulation $\mathcal{T}$ together with a refinement rule $\mathfrak{R}$ is conformingly marked, if each face $F \in \mathcal{F}(\mathcal{T})$ has a unique refinement edge, i.e., $\#r(F) = 1$.

The following refinement rule uses an ordering on the edges of the inital triangulation as well as the refinement rule $\mathfrak{R}_{NVB}$ from Section 5.1. An example of refinement via this new rule $\mathfrak{R}_{NVB'}$ is given in Figure 5.3.

Definition 5.8 (Refinement rule $\mathfrak{R}_{NVB'}$)**.** Let $\mathcal{T}$ be a triangulation of $\Omega \subset \mathbb{R}^3$. The refinement rule $\mathfrak{R}_{NVB'}$ is defined as follows.

- Choose a linear ordering on the set of edges $\mathcal{E}(\mathcal{T})$, such that every edge $E \in \mathcal{E}(\mathcal{T})$ is given an unique index in $\{1, \dots, \#\mathcal{E}(\mathcal{T})\}$.
- For $T \in \mathcal{T}$, set $r(T)$ as the edge with the lowest index in $\mathcal{E}(T)$. For $F \in \mathcal{F}(T)$, set $r(F)$ as the edge with the lowest index in $\mathcal{E}(F)$.
- Let $T = [x_0, x_1, x_2, x_3] \in \mathcal{T}$ with $r(T) = [x_0, x_3]$. Let $F_1 = [x_{i_0}, x_{i_1}, x_{i_2}]$, $F_2 = [x_{j_0}, x_{j_1}, x_{j_2}] \in \mathcal{F}(T)$ be the faces that do not contain $r(T)$, i.e., $i_0, i_1, i_2 \in \{0, 1, 2\}$ and $j_0, j_1, j_2 \in \{1, 2, 3\}$. Thus, $r(F_1) = [x_{i_0}, x_{i_2}]$ and $r(F_2) = [x_{j_0}, x_{j_2}]$ are the refinement edges of F_1 and F_2 respectively. The first generation children of T are then defined as

$$s_1(T) = [x_{i_0}, y, x_{i_1}, x_{i_2}]_1, \;\; s_2(T) = [x_{j_0}, y, x_{j_1}, x_{j_2}]_1,$$

  where $y = (x_0 + x_3)/2$.
- The sons $s_1(T)$ and $s_2(T)$ are then further bisected using $\mathfrak{R}_{NVB}$.

Remark. Note that the ordering of edges in $\mathcal{T}$ defines the refinement edge for each element. Thus, unlike in $\mathfrak{R}_{NVB}$, we do not start with elements T_0 with type $\gamma(T_0) \in \{0, 1, 2\}$. In fact, we will soon see that this strategy leads to different types of elements in the given triangulation. However, after one refinement step, all son elements are defined to have type 1, which means we are then able to continue with the refinement rule $\mathfrak{R}_{NVB}$.

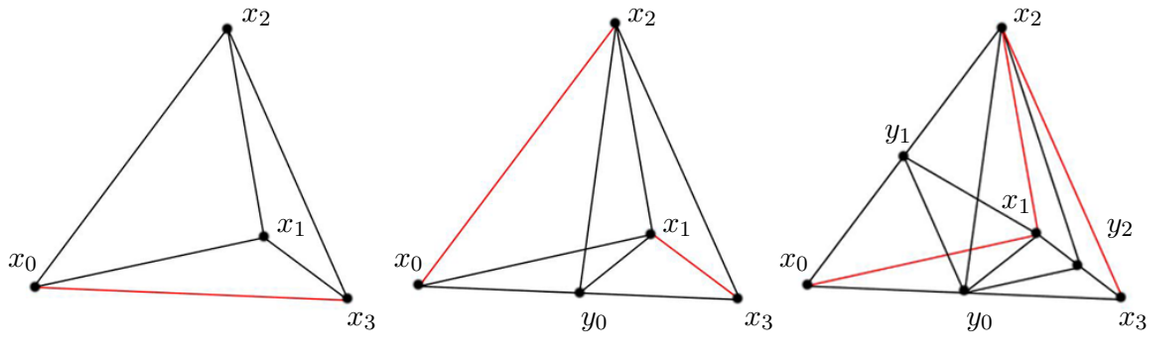


Figure 5.3: Refinement of an element via $\mathfrak{R}_{NVB'}$. Edges coloured red are refinement edges. Thus, for the given element, the edge $[x_0, x_3]$ received the lowest index during the ordering step. The faces that do not contain the refinement edge are given by $F_1 = [x_0, x_1, x_2]$ and $F_2 = [x_1, x_2, x_3]$.

Lemma 5.9. *After three uniform refinements of $T \in \mathcal{T}$ using the refinement rule $\mathfrak{R}_{NVB'}$, every edge $E \in \mathcal{E}(T)$ has been refined exactly once.*

Proof. Let $T = [x_0, x_1, x_2, x_3]$ and thus $r(T) = [x_0, x_3]$. Let $F_1 = [x_{i_0}, x_{i_1}, x_{i_2}]$, $F_2 = [x_{j_0}, x_{j_1}, x_{j_2}] \in \mathcal{F}(T)$ be the faces that do not contain $r(T)$, i.e., $i_0, i_1, i_2 \in \{0, 1, 2\}$ and $j_0, j_1, j_2 \in \{1, 2, 3\}$. Furthermore, let $r(F_1) = [x_{i_0}, x_{i_2}]$ and $r(F_2) = [x_{j_0}, x_{j_2}]$ be the refinement edges of F_1 and F_2, respectively. The first generation children of T are then defined as

$$s_1(T) = [x_{i_0}, y_0, x_{i_1}, x_{i_2}]_1, \ s_2(T) = [x_{j_0}, y_0, x_{j_1}, x_{j_2}]_1,$$

where $y_0 = (x_0 + x_3)/2$. The two elements $s_1(T)$ and $s_2(T)$ are bisected via $\mathfrak{R}_{NVB}$. This results in the four elements

$$[x_{i_0}, y_1, y_0, x_{i_1}]_2, \ [x_{i_2}, y_1, y_0, x_{i_1}]_2, \ [x_{j_0}, y_2, y_0, x_{j_1}]_2 \text{ and } [x_{j_2}, y_2, y_0, x_{j_1}]_2,$$

where $y_1 = (x_{i_0} + x_{i_2})/2$ and $y_2 = (x_{j_0} + x_{j_2})/2$. Note that y_1 and y_2 can denote the same vertex, if $s_1(T)$ and $s_2(T)$ have the same refinement edge. The four refinement edges of the next step are $[x_{i_0}, x_{i_1}]$, $[x_{i_2}, x_{i_1}]$, $[x_{j_0}, x_{j_1}]$ and $[x_{j_2}, x_{j_1}]$, which are all part of the inital element T. The edges $[x_0, x_3]$, $[x_{i_0}, x_{i_2}]$ and $[x_{j_0}, x_{j_2}]$ have already been bisected in the previous steps. Therefore all edges in T have been refined exactly once. □

Remark. Looking at the proof of Lemma 5.9 we can see that in the first three refinement steps of $\mathfrak{R}_{NVB'}$ only edges that were already a part of the given triangulation $\mathcal{T}$ are selected as refinement edges. Therefore, the ordering on the edges of $\mathcal{T}$ does not just define the refinement edge of elements in $\mathcal{T}$ but determines the refinement edges of the first three bisection steps.

The ordering of edges leads to finitely many types of elements that can appear in the inital triangulation. We obtain four different types, 0, 1, AO and OO depicted in Figure 5.4, where the types 0 and 1 correspond to the types we already know from the previous section. These different types were presented in [AM99]. They can best be described by the choice of the two refinement edges for the second bisection step.

- Type 0
  Both edges intersect with the refinement edge of the first bisection but not with each other.
- Type 1
  Both edges intersect with the refinement edge of the first bisection and with each other.
- Type AO
  One of the edges intersects with the refinement edge of the first bisection and the edges also intersect with each other.
- Type OO
  Both edges do not intersect with the refinement edge of the first bisection, i.e., they are the same edge.

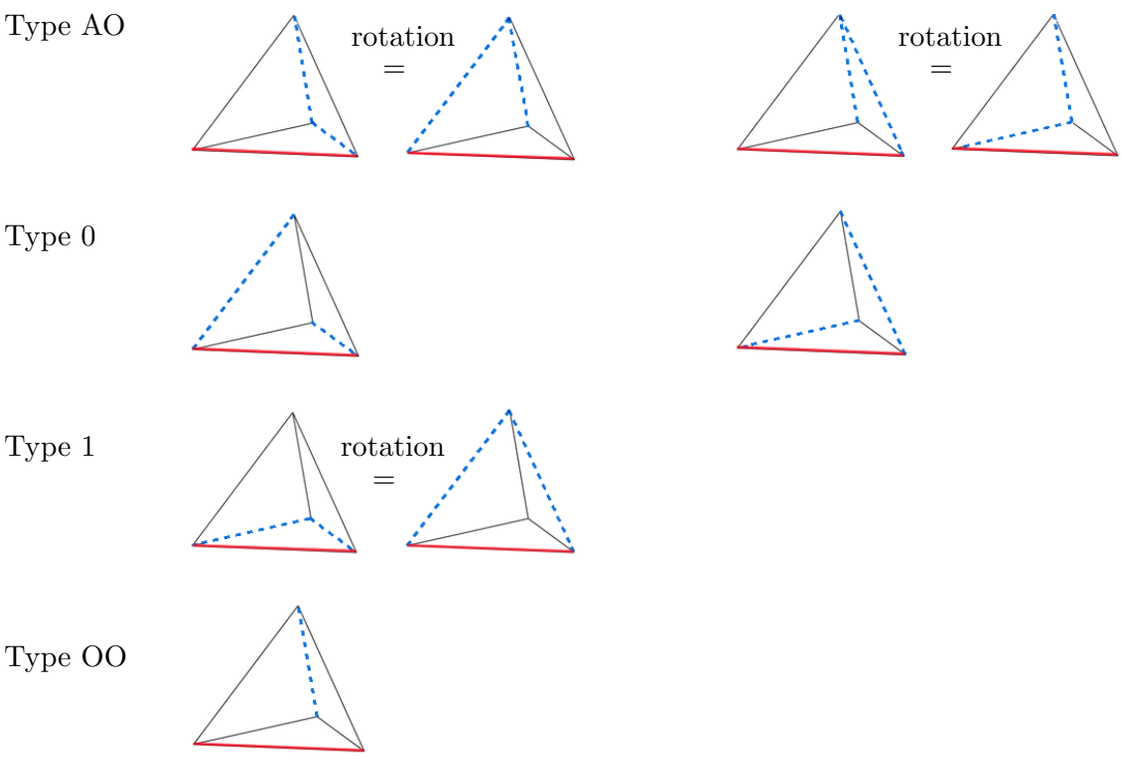


Figure 5.4: Different types of elements that can appear when ordering the edges in a given triangulation as is done in $\mathfrak{R}_{NVB'}$. The red edge is the refinement edge of the first bisection and the two blue dashed edges are the refinement edges of the second bisection. The three remaining edges are refined in the third bisection step.

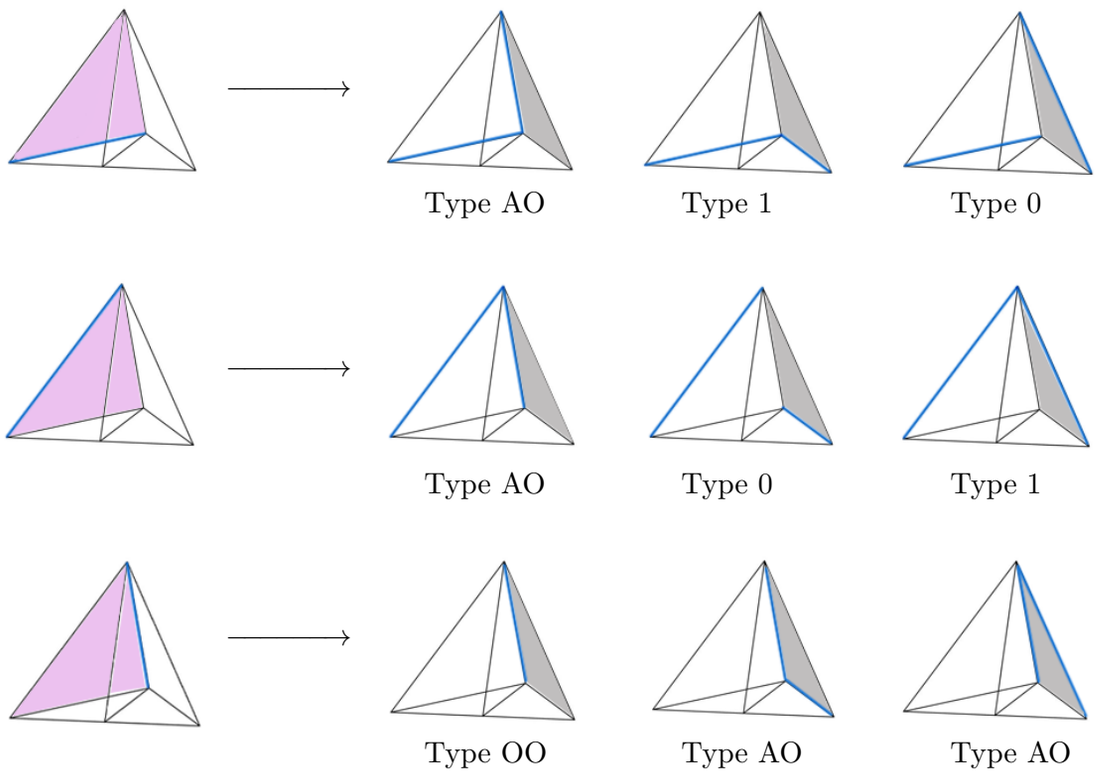


Figure 5.5: All possible combinations of refinement edges for the two sons of an element T. The refinement edges of the sons of T are given by the refinement edges of the faces that did not contain its refinement edge $r(T)$. For the left son, this face is coloured pink, for the son element on the right it is shown in grey.

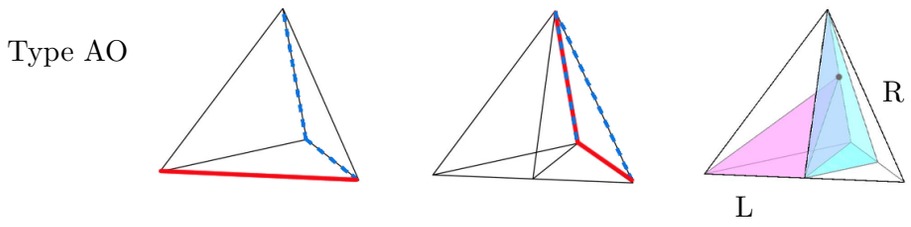


Figure 5.6: Uniform refinement of an element with type AO using $\mathfrak{R}_{NVB'}$. Edges coloured in red are next in line to be bisected, dashed edges coloured in blue are the refinement edges of the resulting son elements. After two bisection steps, we can see that the resulting triangulation is not conforming and produced a hanging node. On the right, two elements L and R of the triangulation are coloured. The intersection of L and R is not a subsimplex of R.

For $\mathfrak{R}_{NVB}$ with condition (IO), it is shown in [Ste08, Theorem 4.3] that every uniform refinement is conforming. Furthermore, for a refinement rule which ensures that every uniform refinement is conforming, one can show that Algorithm 1 terminates, see [Sch17, Proposition 1.7.11]. For $\mathfrak{R}_{NVB'}$, not every uniform refinement is conforming and an example for this can be seen in Figure 5.6. However, for $\mathfrak{R}_{NVB'}$, a weaker statement holds, namely that every uniform refinement of generation $3n$ with $n \in \mathbb{N}$ is conforming, see [Sch17, Corollary 1.8.8]. It is based on a lemma in [AMP00, Lemma 3.2] showing a similar property for their algorithm.

Using a different approach, in [Sch17, Section 1.8] it is shown that $\mathfrak{R}_{NVB'}$ terminates when used with Algorithm 1. Initially, a two-dimensional edge index is introduced. It is defined in such a way, that the refinement edge of an element is also the edge with the unique lexicographically smallest index. Using this index, one can reformulate $\mathfrak{R}_{NVB'}$ into a rule that resembles oldest edge bisection. With this reformulation, it is possible to show termination of $\mathfrak{R}_{NVB'}$, [Sch17, Proposition 1.8.14]. The proof is similar to that of Proposition 3.2 for oldest edge bisection. In addition, the triangulations resulting from refinement with $\mathfrak{R}_{NVB'}$ are also shown to be shape-regular, see [Sch17, Proposition 1.8.15].

Bibliography

[Adl83] Andrew Adler. On the bisection method for triangles. *Math. Comp.*, 40(162):571–574, 1983.

[AM99] Douglas N. Arnold and Arup Mukherjee. Tetrahedral bisection and adaptive finite elements. In *Grid generation and adaptive algorithms (Minneapolis, MN, 1997)*, volume 113 of *IMA Vol. Math. Appl.*, pages 29–42. Springer, New York, 1999.

[AMP00] Douglas N. Arnold, Arup Mukherjee, and Luc Pouly. Locally adapted tetrahedral meshes using bisection. *SIAM J. Sci. Comput.*, 22(2):431–448, 2000.

[BDD04] Peter Binev, Wolfgang Dahmen, and Ron DeVore. Adaptive finite element methods with convergence rates. *Numer. Math.*, 97(2):219–268, 2004.

[HKK14] Antti Hannukainen, Sergey Korotov, and Michal Křížek. On numerical regularity of the face-to-face longest-edge bisection algorithm for tetrahedral partitions. *Sci. Comput. Programming*, 90:34–41, 2014.

[KPP13] Michael Karkulik, David Pavlicek, and Dirk Praetorius. On 2D newest vertex bisection: optimality of mesh-closure and H^1-stability of L_2-projection. *Constr. Approx.*, 38(2):213–234, 2013.

[KPS16] Sergey Korotov, Ángel Plaza, and José P. Suárez. Longest-edge n-section algorithms: properties and open problems. *J. Comput. Appl. Math.*, 293:139–146, 2016.

[Mau95] Joseph M. Maubach. Local bisection refinement for n-simplicial grids generated by reflection. *SIAM J. Sci. Comput.*, 16(1):210–227, 1995.

[Sch17] Patrick Schön. *Scalable Adaptive Bisection Algorithms on Decomposed Simplicial Partitions for Efficient Discretizations of Nonlinear Partial Differential Equations*. Ph.d. thesis, Albert-Ludwigs-Universität Freiburg, Fakultät für Mathematik und Physik, 2017.

[Ste08] Rob Stevenson. The completion of locally refined simplicial partitions created by bisection. *Math. Comp.*, 77(261):227–241, 2008.

[Tra97] Christoph T. Traxler. An algorithm for adaptive mesh refinement in n dimensions. *Computing*, 59(2):115–137, 1997.