Hauptseminar Wissenschaftliches Rechnen 2016/17 (Rumpf)
Vortrag von Tobias Klas

2. Seminarvortrag: Tetraeder-Gitter-Verfeinerung

Ziel des Vortrags ist die Konstruktion eines Algorithmus der zu einer gegebenen Tetraeder-Triangulierung eine konsistente und stabile Verfeinerung erzeugt.

Definition 1 (C1)**.** Wir sagen eine Folge $(\mathcal{T}_m)_{m\in\mathbb{N}_0}$ von Triangulierungen erfüllt die *Verschachtelungseigenschaft*, falls für alle $T \in \mathcal{T}_k$ mit $k > 0$ genau ein $T' \in \mathcal{T}_{k-1}$ existiert, so dass $T \subseteq T'$ und jeder Eckpunkt von T ist entweder Eckpunkt von T' oder Mittelpunkt einer Kante von T'. Wir nennen T' *Vater* von T und T *Sohn* von T'.

Definition 2 (C2)**.** Eine Folge $(\mathcal{T}_m)_{m\in\mathbb{N}_0}$ von Triangulierungen heißt *konsistent*, falls für alle k der Schnitt $T_i \cap T_j$ für alle $T_i, T_j \in \mathcal{T}_k$ entweder leer ist, eine gemeinsame Ecke, eine gemeinsame Kante oder eine gemeinsame Fläche enthält.

Definition 3 (C3)**.** Wir nennen eine Folge $(\mathcal{T}_m)_{m\in\mathbb{N}_0}$ von Triangulierungen *stabil*, falls für jedes k

$$\delta(T) := \frac{h(T)}{2\rho(T)}$$

für alle $T \in \mathcal{T}_k$ einheitlich beschränkt ist. Hierbei ist $h(T)$ die Länge der längsten Seite des Dreiecks T und $\rho(T)$ der Radius des Innenkreises.

Definition 4. Eine Verfeinerungsprozedur, die stabile und konsistente Triangulierungen liefert, nennen wir *regulär*.

2D-Simplex Verfeinerungen:

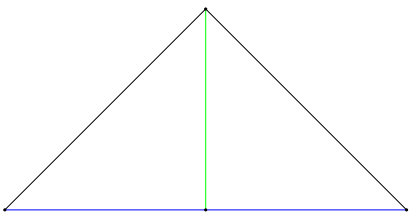


Abbildung 1: Grün-Verfeinerung

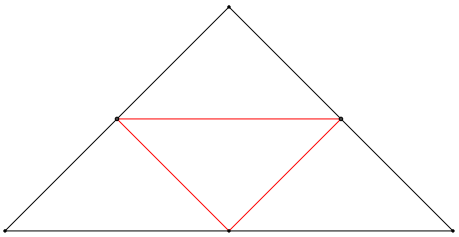


Abbildung 2: Rot-Verfeinerung

Problem hängender Knoten bei globaler Verfeinerung möglich. Ausweg:

- Dreiecke die eine gemeinsame Verfeinerungskante haben werden beide zur selben Zeit verfeinert.
- Ein Dreieck dessen Verfeinerungskante auf dem Rand liegt wird verfeinert.

Für eine globale Verfeinerung merken wir uns:

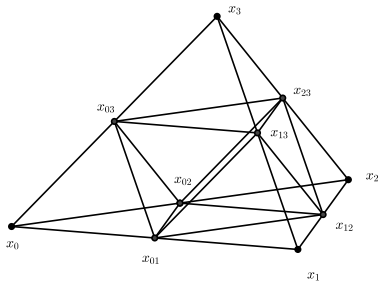
(C4) Irreguläre Simplices werden nie verfeinert, d.h. Simplices die aus einer irregulären Verfeinerung entstanden sind.

(C5) Ist $T \in \mathcal{T}_k \cup \mathcal{T}_{k+1}$, dann ist $T \in \mathcal{T}_l$ für alle $l \geq k$. Wird ein Simplex nicht von $\mathcal{T}_k$ nach $\mathcal{T}_{k+1}$ verfeinert, so wird es nie mehr verfeinert.

Im folgenden sei T stets ein Tetraeder. Unser Ziel ist es nun, T in acht volumengleiche Subtetraeder $T_1, ..., T_8$ zu zerlegen. Dazu sei ein Tetraeder durch seine vier Eckpunkte identifiziert. Also

$$T = [x_0, x_1, x_2, x_3],$$

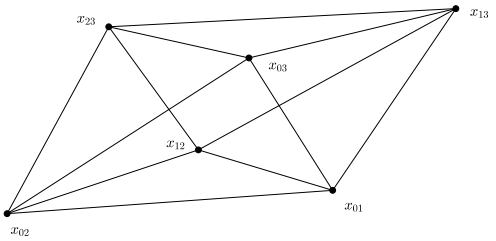
wobei $[x_0, ..., x_3]$ ein geordnetes 4-Tupel.

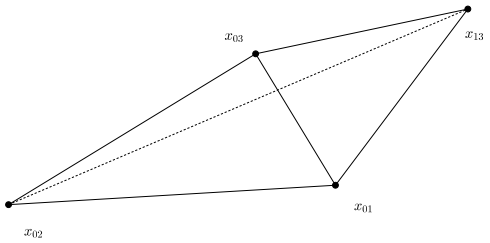


Für $0 \leq i, j \leq 3$, $i \neq j$, notieren wir den Mittelpunkt der Kante $[x_i, x_j]$ durch

$$\boxed{x_{ij} = \frac{1}{2}(x_i + x_j).}$$

Der Traedersetzt sich aus vier volumengleichen Tetraeder in den Ecken und einem Oktaeder zusammen. Dieser Oktaeder lässt sich wie folgt in vier volumengleiche Tetraeder zerlegen. Links ist der innere Oktaeder abgebildet, rechts ein Viertel des Oktaeders.





Das hier beispielhaft beschriebene Vorgehen lässt sich wie folgt in einem Algorithmus präzisieren. Diesen bezeichnen wir mit RRA.

```
1 RegularRefinement(T)
2 {
3 //divide(T) into subtetrahedra T_i with 1 ≤ i ≤ 8 given by
4    T_1 := [x_0, x_01, x_02, x_03]; T_5 := [x_01, x_02, x_03, x_13];
5    T_2 := [x_01, x_1, x_12, x_13]; T_6 := [x_01, x_02, x_12, x_13];
6    T_3 := [x_02, x_12, x_2, x_23]; T_7 := [x_02, x_03, x_13, x_23];
7    T_4 := [x_03, x_13, x_23, x_3]; T_8 := [x_03, x_12, x_13, x_23];
8 }
```

Definition 5. Zwei Tetraeder T_1, T_2 heißen *kongruent*, falls ein $c \neq 0$, ein Vektor $\mathbf{x} \in \mathbb{R}^3$ und eine orthogonale Matrix $Q \in \mathbb{R}^{3\times 3}$ existiert, so dass

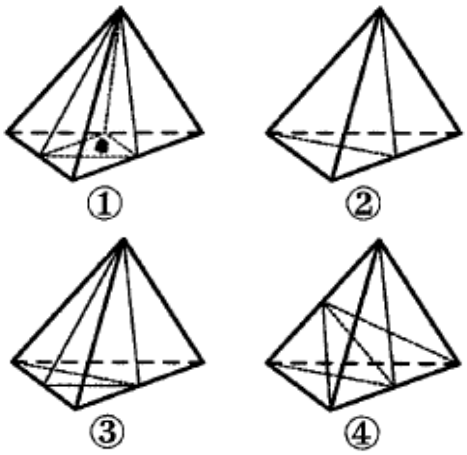
$$T_1 = \mathbf{x} + cQT_2 := \{\mathbf{x} + cQ\mathbf{x}' \mid \mathbf{x}' \in T_2\}.$$

D.h. T_1 lässt sich durch Rotationen, Translationen und Skalierungen in T_2 überführen und umgekehrt.

Offensichtlich stellt diese Kongruenz eine Äquivalenzrelation dar. Somit lassen sich alle Tetraeder in Äquivalenzklassen einteilen. Insbesondere lässt sich jedes durch RRA erzeugt Element einer von höchstens drei Äquivalenzklassen zuordnen.

Theorem 6 (Stabilität und Konsistenz)**.** *Für jeden (initialen) Tetraeder T liefern rekursive Ausführungen von RRA konsistente und stabile Triangulierungen von T.*

Wir wollen uns auf vier veschiedene Typen von irregulären Verfeinerungen eines Tetraeder beschränken. Dieses Vorgehen wollen wir analog zum 2D Fall „green closure "(GC) nennen.



Wir verwenden:

(1) Typ 1, falls ein benachbarter Simplex regulär verfeinert wurde, aber keine andere Kante unterteilt wurde,

(2) Typ 2, falls eine gemeinsame Kante refinement edge ist,

(3) Typ 3 und 4, falls zwei Kanten an einer gemeinsamen Fläche oder zwei gegenüberliegende Kanten unterteilt wurden.

Somit bleibt noch der Fall, dass drei oder mehr Kanten unterteilt wurden, die nicht zu einer gemeinsamen Seitenfläche gehören. Dies führt uns zu einer weiteren Bedingung, die wir uns für den globalen Algorithmus merken wollen.

(C6) Falls drei oder mehr Kanten von T unterteilt wurden, die nicht zu einer gemeinsamen Seitenfläche gehören, wird T regulär verfeinert.

Die zentralen Objekte in unserem Algorithmus sind Elemente (einzelne Tetraeder), Knoten (Eck- und Kantenmittelpunkte), sowie Kanten. Diese sind in folgender Weise auf einem Gitter G_k vom *Level* $k = 0, ..., k_{max}$ verteilt:

(1) G_0 enthält alle Elemente, Knoten und Kanten der Triangulierung $\mathcal{T}_0$.

(2) G_k, $k > 0$, enthält alle Elemente, Knoten und Kanten von $\mathcal{T}_k$, welche nicht schon in $\mathcal{T}_{k-1}$ in gleicher Weise enthalten sind, einschließlich der Kanten und Ecken. Zu den Objekten in G_k sagen wir, dass sie von *Level* k sind.

(3) (C5) impliziert, dass Elemente von Level k entsprechende Elemente von Level $k+1$ enthalten.

Definition 7. Für jeden Level-k-Knoten N gibt es mindestens ein verfeinertes Level-k-Element. Es gibt einen nachfolgenden Knoten N' mit gleichen Koordinaten aber vom Level $k+1$, dieser heißt analog zu Triangulierungen *Sohn* von N. Zwei Knoten von Level k heißen *Nachbarn*, falls sie Eckpunkte einer gemeinsamen Level-k-Kante sind. Wir sagen, dass zwei Elemente von Level k *Nachbarn* sind, falls sie eine gemeinsame Seitenfläche haben. Ein nicht verfeinertes Element aus G_k, $k \geq 0$, nennen wir *Blatt-Element* oder einfach *Blatt*.

Nun wenden wir uns dem Algorithmus zu. Im folgenden GRA abgekürzt.
Input: Eine Folge von Gittern $G_0, ..., G_{k_{max}}$, welche wie oben beschrieben zu einer Triangulierung mit Bedingungen (C1),..., C(5) korrespondiert. Jedes Blatt-Element ist entweder „marked for regular refinement “, „marked for no refinement“oder „marked for coarsening “. Die Markierungen erfolgen gemäß eine Fehlerabschätzung. Die Markierungen aller anderen Elemente werden auf „marked according to refinement “gesetzt.
Aktion: Es wird wie folgt vorgegangen:

- Falls Blatt T „marked for regular refinement“ist, führe RRA(T) aus.
- Falls ein irreguläres Element T „marked for regular refinement“ist, ersetze es durch eine reguläres T'.
- Falls ein irreguläres Element T nicht „marked for regular refinement“ist, führe keine Aktion auf T aus.
- Regulär verfeinertes Elemente T ist „marked for no refinement“, falls alle Söhne „marked for coarsening“sind.
- Führe GC unter Bedingung (C4) bis (C6) aus.

Output: eine Folge $G'_0, ..., G'_{k'_{max}}$ mit $k'_{max} \geq 0$, $G'_0 = G_0$ und $k'_{max} \in \{k_{max}, k_{max} \pm 1\}$.

```
1   GlobalRefinement(G_0, ..., G_{k_max})
2   {
3     for k := k_max down to 0 do
4     {
5       EvaluateMarks(G_k); //compute and changes marks
6       CloseGrid(G_k); // checks neighbors of regular T for irregular ref marks
7     }
8     for k := 0 to k_max if (G_k ≠ ∅) then do
9     {
10      if(k>0) then CloseGrid(G_k);
11      UnrefineGrid(G_k); // delete not used elements of the next level
12      RefineGrid(G_k);
13    }
14    if(G_{k_max} = ∅) then k_max := k_max − 1;
15    else if(G_{k_max+1} ≠ ∅) then k_max := k_max + 1;
16  }
```

Theorem 8 (Laufzeitkomplexität)**.** *Der Arbeitsaufwand von GRA ist proportional zu der Anzahl n^*_{out} von Blättern in der erzeugten Folge $G'_0, ..., G'_{k'_{max}}$.*

Theorem 9 (Konsistenz und Stabilität)**.** *Sei $\mathcal{T}_0$ eine konsistente initiale Triangulierung eines Polyeder-Gebietes Ω. Dann liefert rekursive Anwendung von GRA eine Folge von Triangulierungen, die den Bedingungen C1 bis C5 genügt.*

Quellen: Bey: Tetrahedral Grid Refinement, Deuflhard u. Weiser: Numerik 3, Braess: Finite Elemente.