Interaktive Visualisierung von globalen Höhendaten unter Verwendung verbesserter Bounding Shapes

Jutta Adelsberger

Institut für Numerische Simulation
Universität Bonn
Wegelerstr. 6, D–53115 Bonn
adelsberger@ins.uni-bonn.de

1 Einleitung

Das Ziel dieser Arbeit ist es, die Erdkugel interaktiv auf dem Rechner zu visualisieren und dabei bereits bekannte ebene Methoden der Landschaftsvisualisierung auf das Höhenmodell der Erde zu übertragen. Um trotz der riesigen Datenmengen eine Visualisierung in Echtzeit zu ermöglichen, muß gefordert werden, daß der Rechenaufwand im Verhältnis zur Anzahl der gezeichneten Dreiecke steht und daß diese den sichtbaren Bereich der Landschaft so gut wie möglich approximieren. Dabei soll ein adaptives zusammenhängendes Dreiecksnetz ohne hängende Knoten entstehen.

Um dieses Ziel zu erreichen, werden üblicherweise folgende Adaptivitätskriterien gestellt:

1. **Geometrische Adaptivität**: Je rauher das Gelände ist, d.h. je größer die lokalen Höhenunterschiede sind, desto stärker sollte das Dreiecksnetz verfeinert werden.

2. **Blickpunktabhängigkeit**: Je näher das Gelände sich am Blickpunkt des Beobachters befindet, desto feiner sollte es aufgelöst werden.

3. **Sichtbarkeitstest**: Befindet sich das Gelände außerhalb des Sichtbereichs des Beobachters, braucht es gar nicht verfeinert zu werden.

Als Grundlage benutzen wir dazu den SOAR-Algorithmus von Lindstrom und Pascucci für ebene Datensätze [6]. In ihrem Ansatz werden die Dreiecke beginnend bei der gröbsten Auflösung so lange rekursiv unterteilt (longest edge bisection), bis ein bestimmtes Abbruchkriterium erfüllt ist. Zur Berechnung dieses Kriteriums werden Bounding Shapes für die Dreiecke verwendet. Dies sind einfache geometrische Objekte wie beispielsweise in [6] Kugeln, die die einzelnen Dreiecke einschließen und für die sich das Abbruchkriterium, das z.B. aus einer Abstandsfunktion besteht, schneller berechnen läßt. So wird dann top-down, also beginnend bei der gröbsten Auflösung, ein adaptives Dreiecksnetz erzeugt, das in einem einzigen sogenannten Triangle-Strip abgespeichert wird, was ein effizientes Zeichnen ermöglicht.

Bei der Übertragung dieses Algorithmus von der Ebene auf die Erdkugel stellen sich die Fragen,

- wie ein reguläres Gitter auf der Erdkugel zu wählen ist,

- ob dieses Gitter in einem oder mehreren Triangle-Strips verwaltet wird und

- wie man die Bounding Shapes der Dreiecke an die Kugelgeometrie anpassen sollte.

2 Vorangegangene Arbeiten

Eine große Anzahl von Arbeiten beschäftigt sich mit der Visualisierung von Höhendaten. Für uns sind dabei besonders diejenigen interessant, die mit regulären strukturierten Gittern arbeiten. Die Algorithmen auf solchen Gittern teilen sich dabei im wesentlichen in solche mit sogenannten Bintrees [1, 2, 3, 5, 6] und Quadtrees [4, 9, 11], zu denen Pajarola [10] einen Überblick gibt, auf. Es sei außerdem das aktuelle Buch von Luebke [7] erwähnt, das die verschiedenen Konzepte und Algorithmen zur Level-of-Detail-Visualisierung vorstellt und miteinander vergleicht.

In dem nun folgenden kurzen Rückblick gehen wir lediglich auf reguläre Gitter ein und fokussieren dabei auf die unterschiedlichen Techniken für blickpunktabhängige Verfeinerung.

Einer der ersten Algorithmen, mit dem die Landschaft blickpunktabhängig adaptiv trianguliert wurde, war der von Lindstrom et al. [4]. Der Algorithmus arbeitet bottom-up, d.h. es werden beginnend bei der feinsten Auflösung so lange Paare von Dreiecken zusammengefaßt, bis eine vorgegebene Fehlerschranke erreicht wird. Hängende Knoten werden dadurch vermieden, daß die Abhängigkeiten der Knoten explizit gespeichert werden.

Der ROAM-Algorithmus von Duchaineau et al. [1] verwendet hingegen einen top-down Ansatz und ist inkrementell in dem Sinne, daß von einer gegebenen Triangulierung Dreiecke hinzugenommen bzw. entfernt werden, wenn sich der Betrachter bewegt. Außerdem wird für die Fehlermetrik eine Hierarchie von Bounding Volumen erzeugt.

Die Arbeit von Pajarola [9] war dann eine der ersten, die einen monotonen geometrischen Fehlerindikator einführte, ebenso wie diejenige von Ohlberger und Rumpf [8]. Letztere beinhaltet einen top-down Algorithmus mit einer fast geschachtelten Abstandsmetrik. Hängende Knoten werden dabei durch die Wahl einer bestimmten Abstandsfunktion kompensiert.

Der SOAR-Algorithmus von Lindstrom und Pascucci [5, 6] stellt im Prinzip eine Zusammenfassung aller vorangehenden Arbeiten über Landschaftsvisualisierung dar. Insbesondere führen Lindstrom und Pascucci geschachtelte Bounding Spheres ein, was eine dynamische Fehlerschranke und allgemeinere Fehlermetriken erlaubt und gleichzeitig hängende Knoten vermeidet. Unsere vorliegende Arbeit basiert auf diesem SOAR-Algorithmus und erweitert ihn um die Darstellung der gesamten Erde anstelle von ebenen Landschaftsausschnitten. Außerdem unterscheidet sich unser Ansatz in der Konstruktion und Anwendung der Fehler- und Abstandsmetriken wie bei Gerstner [3].

In der aktuellen Arbeit von Platings und Day [11] wird die Erde

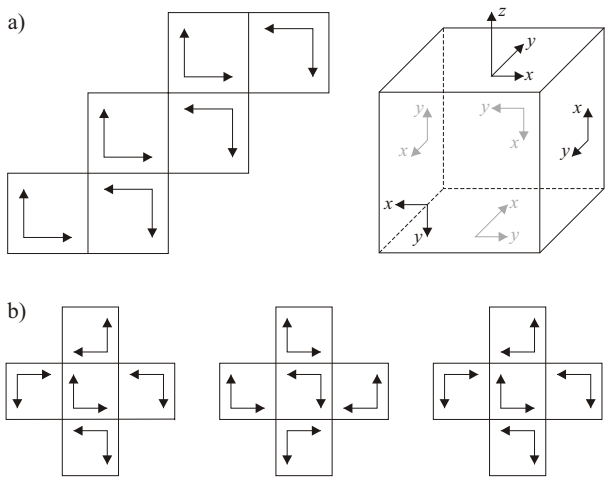

Abb. 1: Numerierung und Orientierung der Würfelseiten. Seite 1 sei dabei die oberste des Würfels. Durch eine solche Anordnung erhalten wir eine Orientierungsinvarianz bezüglich der Nachbarseiten, wie in b) zu sehen ist.

wie bei uns durch einen Polyeder approximiert, um auf bereits bekannte ebene Methoden zurückgreifen zu können. Der eigentliche Fokus dieser Arbeit liegt jedoch auf der Speicherung und Komprimierung der Höhendaten.

3 Idee des Verfahrens

Ist das Prinzip zur Erzeugung eines Dreiecksnetzes für ein ebenes Höhenmodell bzw. einen Landschaftsausschnitt bekannt, so läßt sich dies auf die gesamte Erde übertragen, indem die Erdkugel durch einen Polyeder wie etwa einen Würfel approximiert wird und die einzelnen Seiten wie ebene Landschaftsdatensätze behandelt werden. So erzeugen wir ein Gitter, das im Anschluß nur noch auf die Erdkugel projiziert werden muß, siehe dazu die Abbildung 2. Auf diese Art erübrigt sich zum einen die Behandlung von Singularitäten, die bei der Verwendung von Polarkoordinaten entstehen würden, und zum anderen können wir dadurch einen Teil der Vorberechnungen vom Ebenenfall übernehmen (siehe Kapitel 4). Im folgenden arbeiten wir also seitenorientiert auf dem regulären Gitter des Würfels bzw. auf dessen Projektion auf die Kugel.

Im Kapitel 5 beschreiben wir dann, wie wir mit einem top-down Durchlauf auf jeder Seite ein adaptives Dreiecksnetz mit der bekannten Bisektionsstrategie aufbauen und zeichnen. Die Entscheidung, ob ein Dreieck verfeinert werden muß, hängt dabei im wesentlichen von Sichtbarkeits- und Abstandstests ab, für die es nötig ist, Bounding Shapes um die einzelnen Knoten zu definieren. Dadurch wird nämlich zum einen die Abstandsberechnung vereinfacht, und zum anderen garantieren geschachtelte Bounding Shapes (siehe Saturierungseigenschaft im Kapitel 5.2), daß keine hängenden Knoten entstehen.

Wir diskutieren im Kapitel 6 zwei verschiedene Formen solcher Bounding Shapes, und zwar die einfach zu handhabenden Kugeln und die an die Erdkugelgeometrie besser angepaßten Kegelstümpfe. Für beide Formen behandeln wir dann im folgenden Kapitel 7 Metriken, die uns die gewünschten Adaptivitätskriterien liefern.

Im Kapitel 8 sehen wir uns dann konkrete Ergebnisse an und geben Schlußbemerkungen im Kapitel 9.

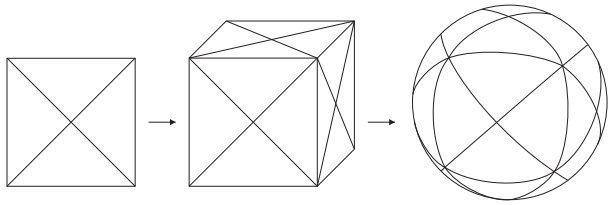

Abb. 2: Konzept der Übertragung des ebenen Dreiecksnetzes auf den Würfel und dann mittels Projektion auf die Kugel.

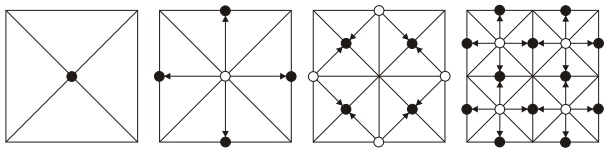

Abb. 3: Bisektionsstrategie. Die jeweiligen Verfeinerungsknoten sind schwarz markiert und die Väter weiß. Die Pfeile verdeutlichen die Vater-Kind-Beziehungen.

4 Das Gitter auf der Kugel

Für das Gitter auf der Erdkugel fordern wir

1. eine einfache Übertragung von dem bekannten ebenen Fall,
2. eine möglichst gleichmäßige Verteilung der Gitterpunkte, um eine gute Approximation zu gewährleisten, und
3. insbesondere keine Singularitäten an den Polen.

Den einfachsten Ansatz bietet da die Approximation der Kugel durch einen Würfel mit uniformem Gitter, was direkt Punkt 1 und 3 erfüllt. Projiziert man dieses Gitter auf die Kugeloberfläche, so sind die Punkte zwar nicht mehr äquidistant verteilt, doch bei genügend kleiner Maschenweite läßt sich dennoch von einer guten Approximation, wie in Punkt 2 gefordert, sprechen.

Bei der Übertragung des ebenen Falles auf die sechs Seiten des Würfels ist der Bezug zu den jeweiligen Nachbarseiten wichtig. Es werden nämlich in einem Vorverarbeitungsschritt für jeden Knoten charakteristische Parameter berechnet (Kapitel 6), für die man Zugriff auf Nachbardreiecke und -knoten haben muß, die auch auf einer anderen Würfelseite liegen können.

Wir numerieren und orientieren die Seiten des Würfels daher nun wie in der Abbildung 1. Dadurch ist eine Orientierungsinvarianz bezüglich der Nachbarseiten gewährleistet, wodurch keine Fallunterscheidung der Seiten nötig ist. So können wir nun in der Vorverarbeitung problemlos auf die Nachbarn eines jeden Knotens zugreifen, die wir später auch als Kind- und Vaterknoten bezeichnen werden, vergleiche das Kapitel 5.1.

Die Projektion der Würfelpunkte auf die Kugel geschieht dann über die folgende Formel. Sei dazu m die halbe Seitenlänge des Würfels, $R = \sqrt{3}m$ der Radius der Kugel, $\boldsymbol{w} = (x, y, m)^T$ ein Punkt auf dem Würfel mit Höhenwert z und $\boldsymbol{v}$ der projizierte Punkt. Der Ursprung liege dabei im Mittelpunkt des Würfels und der Kugel. Dann gilt

$$\boldsymbol{v} = (R + z)\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}.$$

Im Anschluß daran müssen noch die Seitenkoordinaten richtig orientiert werden:

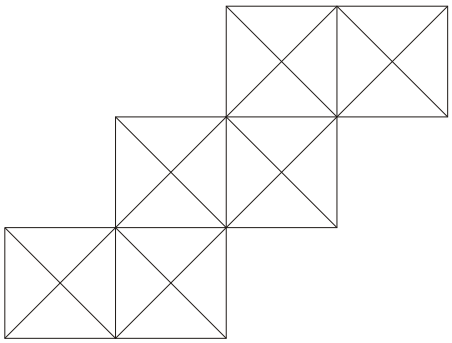

Abb. 4: Durchlaufreihenfolge der Dreiecke, um einen einzigen Triangle-Strip zu erzeugen. Jedes Dreieck wird dabei im Uhrzeigersinn durchlaufen.

Seite	Koordinaten bzgl. Seite 1
1	$(\ x,\ \ y,\ \ z)$
2	$(\ z, -y,\ \ x)$
3	$(-y,\ \ z, -x)$
4	$(-x, -z, -y)$
5	$(-z, -x,\ \ y)$
6	$(\ y,\ \ x, -z)$

5 Triangulierung

Kommen wir nun zu der Erzeugung eines adaptives Dreiecksnetzes. Dazu ist es nötig, eine rekursive Verfeinerungsregel zu definieren (Kapitel 5.1) und sich Gedanken um die sogenannte Saturierungseigenschaft zu machen (Kapitel 5.2).

5.1 Bisektionsstrategie

Wir wählen zur Verfeinerung die wohlbekannte Bisektionsstrategie, bei der in der Mitte der längsten Seite eines Dreiecks ein neuer sogenannter Verfeinerungsknoten eingefügt und mit der gegenüberliegenden Ecke verbunden wird (Abbildung 3). Ausgehend von zwei rechtwinkligen Dreiecken auf jeder Würfelseite erhält man so mit einem top-down Durchlauf ein reguläres Gitter.

Man verdeutliche sich außerdem anhand der Abbildung 3 die Vater-Kind-Struktur der Knoten eines solchen Gitters. Ein Verfeinerungsknoten wird auf der längsten Seite eines Dreiecks eingefügt und gehört daher stets zu zwei benachbarten Dreiecken. Ränder gibt es wegen der Würfelstruktur (Abbildung 2) nicht. So lassen sich also jedem Verfeinerungsknoten, der im folgenden auch Kindknoten genannt wird, zwei Dreiecke zuordnen und somit auch zwei Vaterknoten, die im Netz enthalten sein müssen, ehe der Kindknoten eingefügt werden kann. Die Abbildung 3 verdeutlicht diesen Zusammenhang. Bei der Erzeugung eines adaptiven Gitters muß nun darauf geachtet werden, daß auch tatsächlich zu jedem Knoten seine beiden Väter im Netz enthalten sind, da es sonst zu hängenden Knoten kommen kann, wie im folgenden Kapitel ausgeführt wird.

5.2 Saturierungseigenschaft

Hängende Knoten treten dann auf, wenn zwei Dreiecke mit demselben Verfeinerungsknoten, d.h. mit derselben Hypotenuse, nicht gleichzeitig verfeinert werden. Dies führt nämlich zu Brüchen in der Landschaft, weil die Triangulierung an dieser Stelle nicht mehr kontinuierlich ist, siehe die Abbildung 5a). Um dies zu vermeiden, muß man fordern, daß zu jedem Knoten seine Vaterknoten im Netz enthalten sind. Zur Laufzeit wollen wir es jedoch vermeiden,

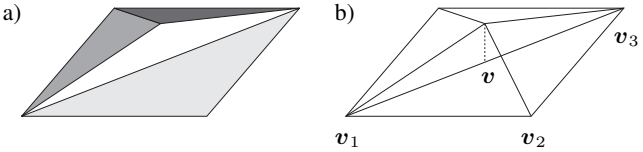

Abb. 5: a) Problem der hängenden Knoten. b) 1-level look ahead Fehler. Die lokale Höhenänderung durch Einfügen eines Verfeinerungsknotens $\boldsymbol{v}$ ist gestrichelt eingezeichnet.

auf Vater-Kind-Beziehungen zurückzugreifen, so daß wir diese Forderung geeignet umformulieren müssen.

Betrachten wir dazu zunächst einmal den Kernalgorithmus zur Konstruktion unseres adaptiven Dreiecksnetzes:

```
visit(Triangle T, Level l) {
  if (metric(T)<=eps or l=lmax) {
    render(T)
  }
  else {
    visit(Child1(T), l+1)
    visit(Child2(T), l+1)
  }
}
```

Eine auf jedem Dreieck T definierte Metrik ist dabei das lokale Abbruchkriterium, das diesen Algorithmus unabhängig von Nachbardreiecken macht. Die Frage ist nun, welche Bedingungen wir an die Metrik $\mu(T)$ stellen müssen, damit hängende Knoten ausgeschlossen sind.

Ein erster Ansatz ist, die Metrik nicht auf dem Dreieck T, sondern auf seinem Verfeinerungsknoten $\boldsymbol{v}(T)$ zu definieren

$$\mu(T) = \mu(\boldsymbol{v}(T)).$$

Dadurch stellen wir sicher, daß zwei Dreiecke mit demselben Verfeinerungsknoten dieselbe Metrik erhalten und deshalb auch gleichzeitig verfeinert werden. Dies schließt jedoch noch nicht alle hängenden Knoten aus, da noch der Fall eintreten kann, daß nur einer der beiden Vaterknoten im Netz enthalten ist. Wir müssen daher noch die sogenannte Saturierungseigenschaft

$$\mu(T) \geq \max\{\mu(\text{Child}_1(T)), \mu(\text{Child}_2(T))\}$$

fordern. Mit diesem Monotoniekriterium sind nun alle Väter eines zu zeichnenden Knotens im Netz enthalten.

5.3 Zeichnen

Mit obigem Algorithmus erzeugen wir also vom gröbsten Level ausgehend top-down ein adaptives Dreiecksnetz. Das Zeichnen geschieht dann im ebenen Fall meist effizient durch die Erzeugung eines einzigen sogenannten Triangle-Strips. Auch unsere seitenorientierte Erdkugel läßt sich so durchlaufen, daß ein einziger zusammenhängender Triangle-Strip entsteht, wie in der Abbildung 4 gezeigt wird.

Ein Problem könnte sich nur durch die Anbindung einer Textur ergeben, da die Größe von 2D-Texturen etwa bei OpenGL beschränkt ist. Da ist es dann einfacher, z.B. sechs getrennte Triangle-Strips, also für jede Seite einen, zu erzeugen und die passenden kleineren Texturen einzubinden.

Ehe wir uns im Kapitel 7 die konkrete Konstruktion der Metrik ansehen, die uns ein solches Dreiecksnetz liefert, benötigen wir noch das Konzept der Bounding Shapes.

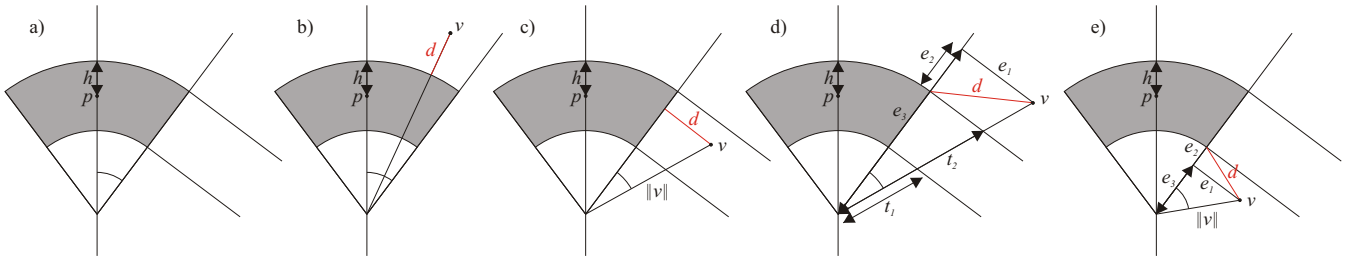

Abb. 6: Fallunterscheidung bei der Abstandsberechnung des Blickpunkts zu den Kegelstümpfen. a) zeigt die 6 verschiedenen Bereiche und b) - e) die Konstruktion der jeweiligen Abstände.

6 Bounding Shapes

Wie bereits motiviert, werden in die Metrik μ Sichtbarkeits- und Abstandstests einfließen, die möglichst einfach berechenbar sein sollten. Daher ist es üblich, ein Dreieck des Netzes durch eine einfachere Struktur, wie z.B. eine umschreibende Kugel, in den Berechnungen zu ersetzen. Diese sogenannten Bounding Shapes $B(T)$ werden dann so definiert, daß dasjenige des Vaters die Bounding Shapes der Kinder enthält

$$B(T) \supseteq \cup\{B(\text{Child}_1(T)), B(\text{Child}_2(T))\}$$

mit $T \subseteq B(T)$. Dieses Kriterium spiegelt die Saturierungseigenschaft wider.

Im folgenden werden wir zwei verschiedene Formen von Bounding Shapes, nämlich Kugeln und Kegelstümpfe mit konzentrisch gewölbter Boden- und Deckelfläche, betrachten. Sie lassen sich bereits in einem Vorverarbeitungsschritt für jeden Knoten berechnen, so daß wir zur Laufzeit folgende Parameter für jeden Knoten $\boldsymbol{v}$ zur Verfügung haben:

- $r(\boldsymbol{v})$: Radius der Bounding Sphere
- $\alpha(\boldsymbol{v})$: Öffnungswinkel des Kegelstumpfes
- $h(\boldsymbol{v})$: Höhe des Kegelstumpfes
- $e(\boldsymbol{v})$: geometrischer Fehler (siehe Kapitel 7.1).

h ist dabei lediglich von den Höhenwerten (z-Koordinaten) der Vater- und Kindknoten abhängig und läßt sich somit noch vor der Projektion des Gitters auf die Erdkugel berechnen. Sei dazu $\boldsymbol{v} = (v_x, v_y, v_z)^T$ der zu betrachtende Knoten auf dem Würfel und $\boldsymbol{c}^1(\boldsymbol{v})$, $\boldsymbol{c}^2(\boldsymbol{v})$ seine beiden Kinder. Dann ist

$$h(\boldsymbol{v}) = \max_{i=1,2}\{|v_z - c_z^i(\boldsymbol{v})| + h(\boldsymbol{c}^i(\boldsymbol{v}))\}.$$

Für die Berechnung von r und α müssen die Knoten zuerst auf die Erdkugel transformiert werden. Sei $\boldsymbol{v}$ also nun der Knoten nach der Projektion. Dann können wir r analog über

$$r(\boldsymbol{v}) = \max_{i=1,2}\{\|\boldsymbol{v} - \boldsymbol{c}^i(\boldsymbol{v})\| + r(\boldsymbol{c}^i(\boldsymbol{v}))\}$$

berechnen, wobei mit $\|\cdot\|$ die euklidische Norm gemeint ist, und ebenso

$$\begin{aligned}\alpha(\boldsymbol{v}) &= \max_{i=1,2}\{\sphericalangle(\boldsymbol{v}, \boldsymbol{c}^i(\boldsymbol{v})) + \alpha(\boldsymbol{c}^i(\boldsymbol{v}))\} \\ &= \max_{i=1,2}\left\{\arccos\left(\frac{\langle \boldsymbol{v}, \boldsymbol{c}^i(\boldsymbol{v})\rangle}{\|\boldsymbol{v}\| \cdot \|\boldsymbol{c}^i(\boldsymbol{v})\|}\right) + \alpha(\boldsymbol{c}^i(\boldsymbol{v}))\right\}\end{aligned}$$

mit dem Standard-Skalarprodukt $\langle\cdot,\cdot\rangle$.

h, r und α werden so levelweise bottom-up, d.h. beginnend bei dem feinsten Level, ausgerechnet. Wegen Rundungsfehlern bei der Berechnung kann es dazu kommen, daß das Monotoniekriterium trotzdem nicht erfüllt ist, so daß wir noch levelweise eine kleine Korrektur aufaddieren müssen, die im Verhältnis zu den Parametern von der Größenordnung 10^{-6} ist. Auch sollte man darauf achten, daß die Parameter an den doppelt vorhandenen Randknoten bei zwei angrenzenden Seiten jeweils identisch sind.

7 Verfeinerungsmetriken

Wir wollen uns nun der Konstruktion unserer Metrik zuwenden, die ein Kriterium dafür sein soll, ob ein Dreieck verfeinert wird oder nicht. Zuvor haben wir uns ja schon überlegt, daß dazu geometrische Adaptivität, Blickpunktabhängigkeit und Sichtbarkeit berücksichtigt werden sollten. Die Idee ist nun, drei unabhängige Metriken zu konstruieren, die diese drei Aspekte widerspiegeln und die dann im Anschluß geeignet kombiniert werden.

7.1 Metrik der Geometrie

Die erste Metrik soll ein Maß dafür sein, wie rauh das Gelände ist. Wird ein neuer Knoten in das Dreiecksnetz eingefügt, ändert sich die Höhe dort lokal, und zwar wird an dieser Stelle nun statt des linearen Interpolationswertes die exakte Höhe des Knotens verwendet. Dieser Höhenunterschied, auch 1-level look ahead Fehler genannt, dient uns als Maß für den geometrischen Fehler, siehe dazu die Abbildung 5b).

Da nur die Höhenwerte der Knoten in die Berechnung eingehen, läßt sich dies wieder vor der Projektion des Würfelgitters auf die Kugel durchführen. Sei dazu $\boldsymbol{v} = (v_x, v_y, v_z)^T$ der Verfeinerungsknoten des Dreiecks $T = \Delta(\boldsymbol{v}^1, \boldsymbol{v}^2, \boldsymbol{v}^3)$ auf dem Würfel, wobei $\boldsymbol{v}^1$ und $\boldsymbol{v}^3$ die Eckpunkte des Dreiecks sind, die die Hypotenuse einschließen. Die geometrische Metrik definieren wir dann über

$$\tilde{\mu}_{geo}(T) = \left| v_z - \frac{1}{2}(v_z^1 + v_z^3) \right|.$$

Diese Metrik erfüllt im allgemeinen nicht die Saturierungseigenschaft, was sich jedoch wieder in einem bottom-up Durchlauf über die Formel

$$\mu_{geo}(T) = \max\{\tilde{\mu}_{geo}(T), \mu_{geo}(\text{Child}_1(T)), \mu_{geo}(\text{Child}_2(T))\}$$

korrigieren läßt. Dabei gilt $\mu_{geo}(T) = \tilde{\mu}_{geo}(T)$ auf dem feinsten Level. Diese Metrik läßt sich also vorab berechnen, so daß wir sie als festen Parameter $e(\boldsymbol{v})$ für alle Knoten abspeichern können.

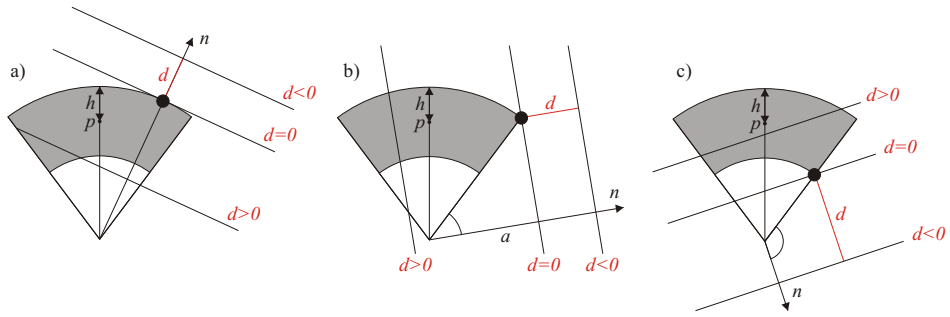

Abb. 7: Fallunterscheidung bei der Abstandsberechnung der Clipping-Ebenen zu den Kegelstümpfen. Der Abstand bei a) - c) wird immer zum schwarz markierten Punkt berechnet.

7.2 Metrik des Blickpunkts

Die Idee für die zweite Metrik besteht darin, daß Landschaft, die nahe am Blickpunkt des Beobachters liegt, stärker aufgelöst sein sollte als solche, die weiter weg ist. Das bedeutet, daß wir den Abstand eines Dreiecks zum Augpunkt berechnen müssen. Bei dieser Abstandsberechnung ersetzen wir das jeweilige Dreieck T durch seinen Bounding Shape $B(T)$, was zum einen die Berechnung vereinfacht und zum anderen direkt die inverse Saturierungseigenschaft

$$d(\boldsymbol{v}, B(T)) \leq d(\boldsymbol{v}, B(\text{Child}_i(T)))$$

erfüllt. Dabei sei d der Abstand, $\boldsymbol{v}$ der Augpunkt und $i = 1, 2$.

Die saturierte Metrik des Blickpunkts definieren wir dann über

$$\mu_{dist}(T) = \frac{1}{d(\boldsymbol{v}, B(T))^2}.$$

Der Abstand geht dabei quadratisch ein, weil die zweidimensionale Projektion eines Objektes quadratisch mit dem Abstand zum Beobachter kleiner wird.

Betrachten wir nun kurz die tatsächliche Berechnung des Abstands anhand unserer beiden Test-Bounding Shapes. Im folgenden sei immer $\boldsymbol{p}$ der Verfeinerungsknoten des aktuellen Dreiecks und somit auch der Mittelpunkt des jeweiligen Bounding Shapes, $\boldsymbol{v}$ der Augpunkt des Beobachters und $\|\cdot\|$ die euklidische Norm. Für die Kugeln ergibt sich die einfache Formel

$$d(\boldsymbol{v}, B_{sphere}(T)) = \|\boldsymbol{p} - \boldsymbol{v}\| - r(\boldsymbol{p}).$$

Für die Kegelstümpfe ist eine Fallunterscheidung nötig, wie in der Abbildung 6a) zu sehen ist. Da ein Kegelstumpf ein Rotationskörper ist, genügt es dabei, seinen Querschnitt zu betrachten.

Sei $\beta = \sphericalangle(\boldsymbol{v}, \boldsymbol{p})$, $\gamma = \beta - \alpha$ und $d = d(\boldsymbol{v}, B_{cone}(T))$. Die Fälle 1, 2 und 3 erhält man dann direkt über die Abfrage $\beta \leq \alpha$ und

- $\|\boldsymbol{v}\| > \|\boldsymbol{p}\| + h$: Fall 1, Abstand zum Deckel:

$$d = \|\boldsymbol{v}\| - (\|\boldsymbol{p}\| + h),$$

- $\|\boldsymbol{v}\| < \|\boldsymbol{p}\| - h$: Fall 3, Abstand zum Boden:

$$d = (\|\boldsymbol{p}\| - h) - \|\boldsymbol{v}\|,$$

- sonst: Fall 2, Augpunkt im Kegelstumpf:

$$d = 0.$$

Gilt $\beta > \alpha$, so lassen sich die Fälle 4, 5 und 6 wie folgt unterscheiden. Dazu seien $t_1 = \frac{\|\boldsymbol{p}\| - h}{\cos\gamma}$ und $t_2 = \frac{\|\boldsymbol{p}\| + h}{\cos\gamma}$, siehe Abbildung 6d).

- $\gamma < 90°$ und $t_1 \leq \|\boldsymbol{v}\| \leq t_2$: Fall 5, Abstand zur Mantelfläche:

$$d = \|\boldsymbol{v}\| \sin\gamma,$$

- $\gamma < 90°$ und $\|\boldsymbol{v}\| > t_2$: Fall 4, Abstand zum Rand des Deckels:

$$d^2 = \|\boldsymbol{v}\|^2 - 2\|\boldsymbol{v}\| \cos\gamma(\|\boldsymbol{p}\| + h) + (\|\boldsymbol{p}\| + h)^2,$$

- sonst: Fall 6, Abstand zum Rand des Bodens:

$$d^2 = \|\boldsymbol{v}\|^2 - 2\|\boldsymbol{v}\| \cos\gamma(\|\boldsymbol{p}\| - h) + (\|\boldsymbol{p}\| - h)^2.$$

Vergleiche zu den Formeln jeweils die Abbildungen 6b) - e).

7.3 Metrik der Sichtbarkeit

Die dritte Metrik ist nun ein Maß für die Sichtbarkeit. Landschaft, die nicht im Sichtbarkeitsbereich des Betrachters liegt, muß prinzipiell auch gar nicht gezeichnet werden. Beschrieben wird dieser Bereich standardmäßig durch sechs sogenannte Clipping-Ebenen, die einen Pyramidenstumpf bilden. Die Idee ist nun, den Abstand jeder Clipping-Ebene zum jeweiligen Bounding Shape auszurechnen und anhand des Vorzeichens zu entscheiden, auf welcher Seite der Ebene es liegt. Die Clipping-Metrik ist dann über

$$\mu_{clip}(T) = \begin{cases} 1 & \text{falls } d(\text{Ebene}_i, B(T)) \leq 0 \quad \forall\, i \\ 0 & \text{sonst} \end{cases}$$

definiert. Durch die Schachtelungseigenschaft der Bounding Shapes ist sie bereits saturiert.

Sei jede Clipping-Ebene gegeben durch ihren Normalenvektor $\boldsymbol{n}$ und den Abstand a zum Ursprung. Um nun den Abstand zu den Kugeln als Bounding Shapes auszurechnen, muß lediglich der Mittelpunkt in die Hessesche Normalenform der Ebene eingesetzt und mit dem Radius r verglichen werden:

$$d(\text{Ebene}, B_{sphere}(T)) = \langle \boldsymbol{n}, \boldsymbol{p} \rangle + a.$$

Gilt dann $d > r$, so liegt die Kugel komplett außerhalb des Sichtbereichs und bei $d < -r$ komplett innerhalb.

Bei den Kegelstümpfen sind wieder Fallunterscheidungen für die Abstandsberechnung nötig. Sei diesmal $\beta = \sphericalangle(\boldsymbol{n}, \boldsymbol{p})$, $\gamma = \beta - \alpha$ und $d = d(\text{Ebene}, B_{cone}(T))$. Zeigt der Normalenvektor zur Ebene hin, so ist der Abstand a der Ebene zum Ursprung kleiner Null. Es existieren drei Fälle, wie in der Abbildung 7 gezeigt wird:

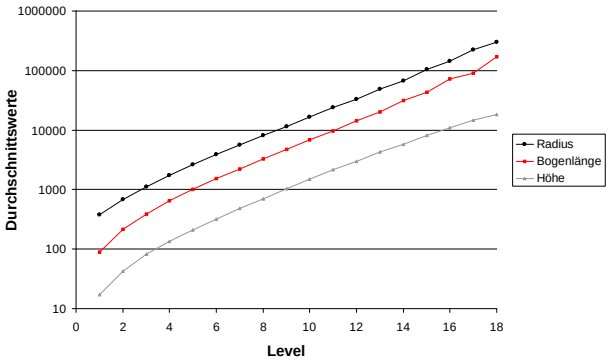

Abb. 8: Durchschnittswerte der Radien der Kugeln bzw. der Höhen und Bogenlängen der Kegelstümpfe pro Level in logarithmischer Darstellung. 0 ist dabei das feinste Level und 18 das gröbste.

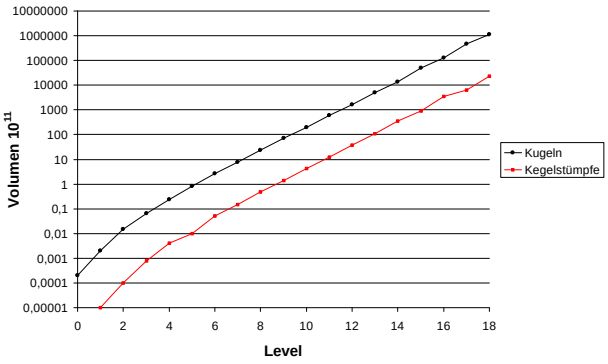

Abb. 9: Volumenvergleich der Kugeln und Kegelstümpfe pro Level, wobei 0 das feinste und 18 das gröbste Level ist. Beachte auch hier die logarithmische Skalierung der 2. Achse.

- $\gamma \leq 0$: Fall 1, Abstand zum Deckel:

$$d = a + (\|\boldsymbol{p}\| + h),$$

- $0 < \gamma \leq 90°$: Fall 2, Abstand zum Rand des Deckels:

$$d = a + \cos\gamma(\|\boldsymbol{p}\| + h),$$

- $\gamma > 90°$: Fall 3, Abstand zum Rand des Bodens:

$$d = a + \cos\gamma(\|\boldsymbol{p}\| - h).$$

Invertiert man dann die Ebene, indem man den Normalenvektor und den Abstand zum Ursprung negiert, und berechnet den Abstand zum Kegelstumpf erneut, so läßt sich durch einen Vergleich der Vorzeichen dieser beiden Abstände entscheiden, auf welcher Seite der Kegelstumpf bezüglich der Ebene liegt. Gilt $d_1 > 0$ und $d_2 < 0$, so liegt er komplett außerhalb des Sichtbereiches, bei $d_1 < 0$ und $d_2 > 0$ komplett innerhalb, und ansonsten schneidet er die Ebene.

7.4 Kombination der Metriken

Nun wollen wir die drei Metriken geeignet kombinieren. Wir haben sie so konstruiert, daß sie alle einzeln die Saturierungseigenschaft erfüllen. Da die Clipping-Metrik außerdem lediglich eine Indikatorfunktion ist, bietet es sich an, das Produkt der einzelnen Metriken

$$\mu(T) = \mu_{geo}(T) \cdot \mu_{dist}(T) \cdot \mu_{clip}(T)$$

zu bilden. Das Ergebnis ist wieder saturiert. Auf diese Weise lassen sich die einzelnen Metriken dynamisch an- und abschalten, oder es lassen sich weitere hinzunehmen.

8 Ergebnisse

Als Test haben wir das Höhenmodell des GTOPO30-Datensatzes [12] verwendet und dazu die in Polarkoordinaten gegebenen Knoten mit ihren Höhenwerten analog zu Kapitel 4 auf ein reguläres Würfelgitter transformiert. Dabei wurde der Datensatz auf $1/5$ verkleinert, so daß auf jeder Seite genau 1025^2 Knoten liegen. Abzüglich doppelter Ränder bedeutet dies eine Überdeckung der Erde mit 6.291.458 Knoten, was einer mittleren horizontalen Maschenweite von 9 km entspricht. Als Textur verwenden wir eine höhenabhängige Farbtabelle.

Wir wollen uns nun konkret die Ergebnisse für die Kugeln und Kegelstümpfe ansehen und vergleichen. Zunächst einmal ist der Speicherplatz zu erwähnen, der für jeden Knoten zur Verfügung gestellt werden muß. Abgesehen von den 3D-Koordinaten und dem geometrischen Fehler e benötigen wir für die Kugeln den Radius r und für die Kegelstümpfe die Höhe h und den Öffnungswinkel α.

Für unseren Beispieldatensatz nimmt die Berechnung dieser Parameter eine Vorverarbeitungszeit von 15 Sekunden in Anspruch, und es wird dabei eine Datei in der Größe von 200 MB erzeugt. Getestet wurde alles unter Linux auf einem 3,2 GHz Pentium 4 Prozessor mit einer NVIDIA Quadro FX 1300 Graphikkarte.

Ein Vergleich der durchschnittlichen Größe von r, h und der mittleren Bogenlänge der Kegelstümpfe ist in der Abbildung 8 dargestellt. Die mittlere Bogenlänge $b = \|\boldsymbol{p}\|\alpha$ ist dabei ein Maß für die horizontale Ausdehnung der Kegelstümpfe. Man stellt fest, daß sie im Schnitt fünfmal so groß ist wie die Höhe, d.h. daß die Kegelstümpfe sehr flach sind. Dies läßt erwarten, daß besonders in der Vertikalen Einsparungen gegenüber den Kugeln als Bounding Shapes gemacht werden können, was wir später noch sehen werden.

Es läßt sich auch levelweise das Volumen der Kugeln und Kegelstümpfe vergleichen, was noch einen besseren Eindruck davon vermittelt, wieviel genauer das Gelände von den Kegelstümpfen eingeschlossen wird. Betrachte dazu die Abbildung 9. Im Schnitt ist das Volumen der Kugeln fünfzigmal größer als das der Kegelstümpfe.

Der Vorteil der Kugeln ist allerdings neben dem geringeren Speicherplatz eine schnellere Berechnung der Abstände, wie wir im Kapitel 7 gesehen haben. Die Berechnung des Abstandes zu einem Punkt bzw. zu einer Ebene liegt bei den Kegelstümpfen im Bereich von 10^{-7} Sekunden und bei den Kugeln im Bereich von 10^{-9} Sekunden. Bei beiden Versionen läuft das Programm in Echtzeit.

Zum Schluß wollen wir uns nun konkrete Screenshots des Programms ansehen und die Anzahl der gezeichneten Dreiecke vergleichen. Bei einem entfernten Blick auf die Erde, wie in der Abbildung 11 dargestellt ist, läßt sich nur ein geringer Unterschied zwischen den Kugeln und Kegelstümpfen als verwendete Bounding Shapes feststellen. Erst bei relativ grober Auflösung wird es deutlicher sichtbar, daß die Kugeln mehr Dreiecke benötigen.

In der Abbildung 12 betrachten wir nun einen Blick von den Alpen über Mitteleuropa in Richtung Norden. Der Sichtbereich wurde auf den rot umrandeten Teil eingeschränkt, um die Leistung der Clipping-Metrik beurteilen zu können. Betrachtet man die Anzahl

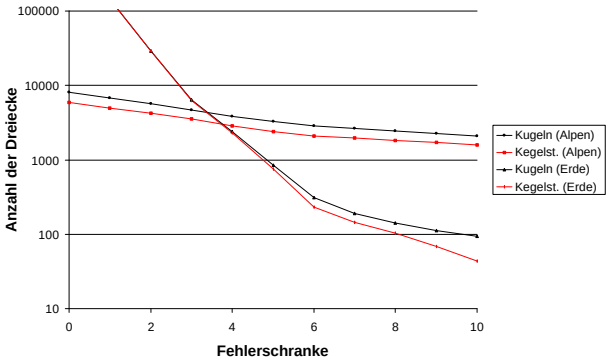

Abb. 10: Anzahl der gezeichneten Dreiecke in Abhängigkeit der vorgegebenen Fehlerschranke für die beiden Beispiele in der Abbildung 11 und 12. Auch hier ist die 2. Achse logarithmisch skaliert.

der gezeichneten Dreiecke in Abhängigkeit der vorgegebenen Fehlerschranke, so stellt man fest, daß die Anzahl der Dreiecke bei den Kugeln im Schnitt 1,4mal größer als bei den Kegelstümpfen ist, siehe dazu die Abbildung 10. Die Einsparungen werden dabei insbesondere außerhalb des Sichtbereichs gemacht und dort vor allem oberhalb, wie man in der Abbildung 12 deutlich sehen kann. Dies bestätigt unsere Beobachtung, daß die Kegelstümpfe sehr flach sind und daher das Gelände besonders in der Vertikalen enger einschließen.

9 Schlußbemerkungen

Wir haben gezeigt, daß es mit geringem Aufwand möglich ist, ein ebenes Visualisierungsverfahren, wie es z.B. aus [6] bekannt ist, auf die Erdkugel zu übertragen. Legt man ein reguläres Würfelgitter zugrunde, so läßt sich völlig analog triangulieren; lediglich eine Transformation der Knoten auf die Kugel ist zu Beginn des Programms nötig.

Zusätzlich muß man sich allerdings noch Gedanken um die Bounding Shapes machen, die auch nach Transformation der Knoten auf die Erdkugel geschachtelt sein sollten. Als einfachstes Beispiel haben wir Kugeln konstruiert, doch es lohnt sich auch, besser an die Erdkugelgeometrie angepaßte Bounding Shapes zu verwenden. Je genauer der Bounding Shape das Gelände nämlich einschließt, desto weniger Dreiecke müssen bei vorgegebenem Fehler gezeichnet werden.

Die Kegelstümpfe als Beispiel haben uns dann gezeigt, daß sie tatsächlich deutlich sichtbare Unterschiede liefern, insbesondere bezüglich der Clipping-Metrik, allerdings auf Kosten teurerer Abstandsberechnungen. Je optimaler und somit komplexer man die Bounding Shapes wählt, desto teurer sind auch die Abstandsberechnungen, so daß man ein möglichst ausgewogenes Mittel finden sollte. Im Fall der Kegelstümpfe halten sich die Kosten dieser Berechnungen noch im Rahmen, so daß auch mit ihnen eine Visualisierung in Echtzeit möglich ist.

Außer durch die Wahl der Bounding Shapes lassen sich noch Verbesserungen bzw. Beschleunigungen durch Nutzen der Graphik-Hardware oder durch veränderte Verfeinerungskriterien gewinnen. Dazu sei noch einmal darauf hingewiesen, daß wir bei unserer Konstruktion die Metriken als Module betrachten können, die durch simple Multiplikation verknüpft sind. Sie lassen sich also aufs Einfachste erweitern und zu einem anderen Verhalten abändern.

Literatur

[1] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, C. Aldrich, and M. B. Mineev-Weinstein. ROAMing terrain: Real-time optimally adapting meshes. In *Proceedings of IEEE Visualization '97*, pages 81–88, 1997.

[2] T. Gerstner. Multiresolution visualization and compression of global topographic data. *GeoInformatica*, 7(1):7–32, 2003. Shortened version in *Proceedings of Spatial Data Handling 2000*, pages 14–27, IGU/GISc, 2000.

[3] T. Gerstner. Top-down view-dependent terrain triangulation using the octagon metric. Technical report, Department for Applied Mathematics, University of Bonn, 2003.

[4] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, N. Faust, and G. A. Turner. Real-time, continuous level of detail rendering of height fields. In *Proceedings of SIGGRAPH '96*, pages 109–118, 1996.

[5] P. Lindstrom and V. Pascucci. Visualization of large terrains made easy. In *Proceedings of IEEE Visualization '01*, pages 363–370, 2001.

[6] P. Lindstrom and V. Pascucci. Terrain simplification simplified: A general framework for view-dependent out-of-core visualization. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):239–254, 2002.

[7] D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner. *Level of Detail for 3D Graphics*. Morgan Kaufman, 2002.

[8] M. Ohlberger and M. Rumpf. Adaptive projection methods in multiresolutional scientific visualization. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):74–94, 1999.

[9] R. Pajarola. Large scale terrain visualization using the restricted quadtree triangulation. In *Proceedings of IEEE Visualization '98*, pages 19–26, 1998.

[10] R. Pajarola. Overview of quadtree-based terrain triangulation and visualization. Technical report, Department for Information & Computer Science, University of California Irvine, 2002.

[11] M. Platings and A. M. Day. Compression of large-scale terrain data for real-time visualization using a tiled quad tree. *Computer Graphics Forum*, 23(4):741–759, 2004.

[12] U.S. Geological Survey. GTOPO30. Available at http://edcdaac.usgs.gov/gtopo30/gtopo30.asp.

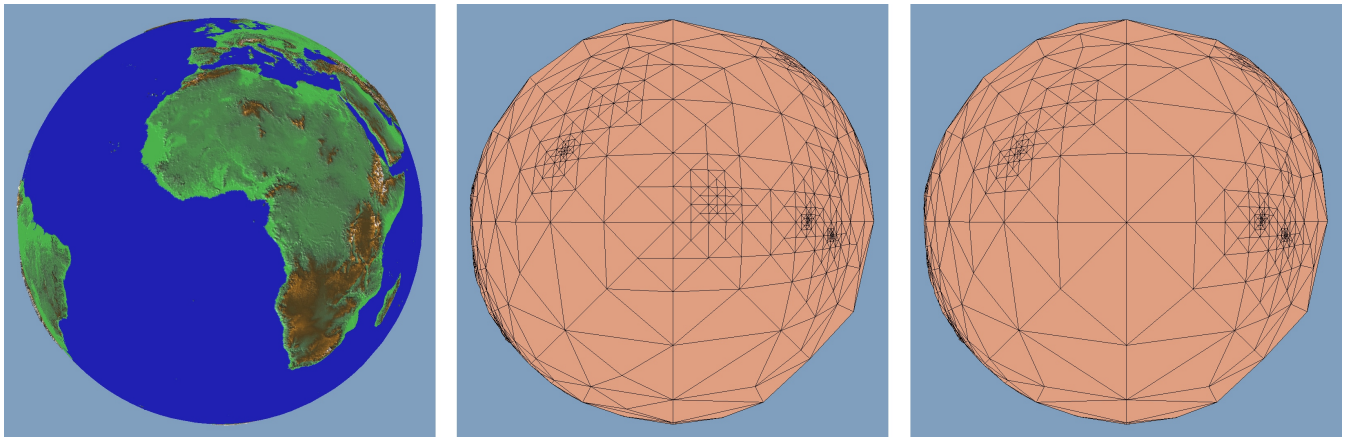

Abb. 11: Blick auf die Erde. Das Bild in der Mitte zeigt die Triangulierung mit Kugeln als Bounding Shapes und das Bild rechts mit Kegelstümpfen bei gleicher vorgegebener Fehlerschranke.

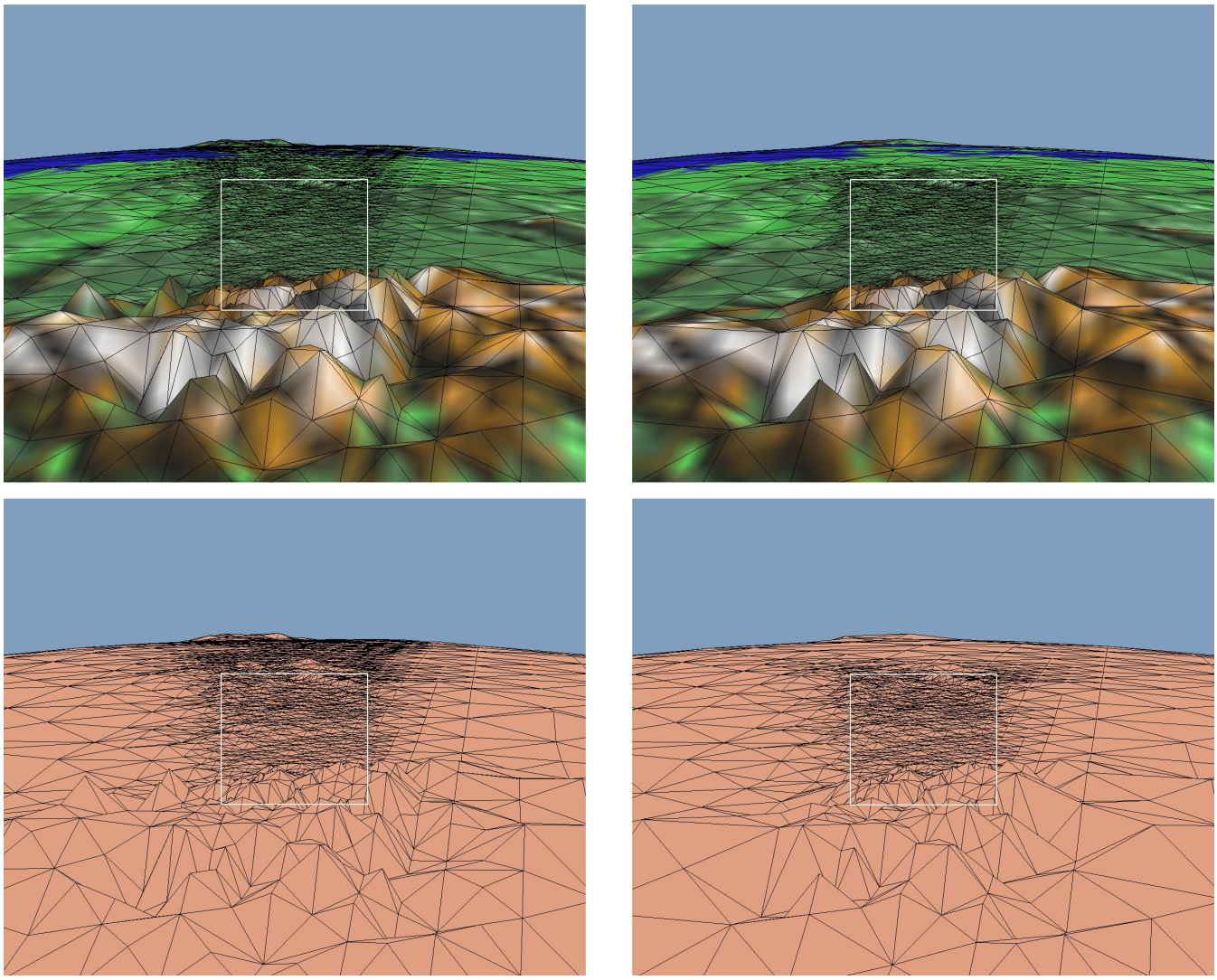

Abb. 12: Blick von den Alpen über Mitteleuropa in Richtung Norden. Zur Beurteilung der Clipping-Metrik ist der Sichtbereich auf den weiß umrandeten Teil eingeschränkt. Bei den Bildern links wurden Kugeln und rechts Kegelstümpfe als Bounding Shapes bei gleicher vorgegebener Fehlerschranke verwendet.