

RHEINISCHE FRIEDRICH-WILHELMS-UNIVERSITÄT BONN
INSTITUT FÜR INFORMATIK I



Das Voronoi Spiel

DIPLOMARBEIT

in der Studienrichtung

Informatik

Betreuung:

Dr. Elmar Langetepe
Prof. Dr. Rolf Klein

Eingereicht von:

Csilla Bükki

Bonn, September 2005

Danksagung

An dieser Stelle möchte ich all jenen danken, die durch ihre fachliche und persönliche Unterstützung zum Schaffen dieser Diplomarbeit beigetragen haben.

Besonders bin ich dankbar meinem Mann, Dr. Christian Szegedy, der mir dieses Informatik Studium durch seine Unterstützung ermöglicht hat.

Ich bedanke mich weiters bei Herrn Prof.Dr.Rolf Klein und Herrn Dr. Elmar Langetepe für die Betreuung meiner Diplomarbeit und den vielen wissenschaftlichen Ratschläge, welche zur Verbesserung der Diplomarbeit beigetragen haben.

Erklärung

Ich habe die Diplomarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht.

Bonn, September 2005

Csilla Bükki

Inhaltsverzeichnis

1	Einleitung	3
2	Das Voronoi Spiel	9
2.1	Einführung mit einigen Definitionen	9
2.2	Anwendung des Voronoi-Diagramms	13
3	Ergebnisse im Kreis	19
3.1	1-dimensionaler Fall am Kreisrand	19
3.2	2-dimensionaler Fall im Kreis	28
4	Ergebnisse auf der Kugeloberfläche	37
4.1	Sphärische Geometrie	37
4.2	Stabile und instabile Polyeder	42
5	Einige Charakteristiken	51
5.1	Beschreibungsfunktion	51
5.2	Konvergenz des Verfahrens	60
6	Implementierungen	67
6.1	Die zwei Realisierungen des Spieles	68
6.2	Ermittlung der Voronoi-Knoten und Voronoi-Nachbarschaft	74
6.3	Implementierung einiger Körper	78
7	Benutzung der Applets	87
7.1	Java 2D Applet	87
7.2	Java 3D Applet	89
	Literaturverzeichnis	93

Kapitel 1

Einleitung

Kurze Überblick

In dieser Diplomarbeit wird das sogenannte **Voronoi Spiel** betrachtet und analysiert, das man anschaulich folgendermaßen beschreiben kann:

Das Voronoi Spiel im Kreis: Es gibt n Personen (durchnummeriert von 1 bis n) zufälligerweise in einem kreisförmigen Raum. Am Anfang kommt der erste Spieler mit Spielnummer 1 an die Reihe, die anderen dürfen sich nicht bewegen. Der erste Spieler soll so lange im Raum spazieren, bis er, am weitesten entfernt von den anderen Spielern, im Raum ankommt, das heißt, dass der minimale Abstand zu allen anderen Spielern maximiert wird, danach bleibt er stehen und der zweite Spieler kommt an die Reihe. Er muss auch so weit wie möglich von den anderen Spielern weggehen. Danach kommt der Dritte, danach der Vierte und so weiter. Nach dem letzten Spieler mit Spielnummer n kommt wieder der erste Spieler an die Reihe und das Spiel wird beliebig lange weitergespielt. Wir möchten gern wissen, wo sich die Spieler nach langer Zeit befinden, falls sich jeder schon beliebig vielmal bewegen durfte? Bewegen sie sich immer im Raum oder bleiben sie stehen?

Das Voronoi Spiel auf der Kugeloberfläche: Jetzt sind die n Personen auf einem kugelförmigen Planeten. Sie können nur auf der Oberfläche des Planeten spazieren. Die Spielregeln sind gleich wie im Fall des Kreises: am Anfang haben die Spieler zufällige Positionen, in jedem Schritt kann sich nur ein Spieler (nach Spielnummer) bewegen und er muss die weiteste Stelle auf dem Planeten von den anderen Mitbewohnern finden. Die Frage lautet auch hier: wo sind die Spieler nach beliebig vielen Iterationsschritten auf dem Planeten? Bleiben sie stehen oder müssen sie immer weiterspazieren?

Für beide Fälle stellt sich die Frage, ob dieses Verfahren überhaupt gegen eine stabile Konfiguration konvergieren kann. Die Charakterisierung der stabilen Konfiguration ist auch spannend und im allgemeinen Fall nicht trivial.

Bei der mathematischen Betrachtung des Spieles sind die Spieler durch Punkte innerhalb des Kreises bzw. auf der Kugeloberfläche zu repräsentieren.

Im zweiten Kapitel wird eine noch genauere Beschreibung des Voronoi Spieles gegeben. Es werden hier mehrere, schon bekannte Definitionen (u.a. der Euklidische Abstand, das Voronoi-Diagramm, die Delaunay-Zerlegung), und einige neue Definitionen eingeführt, die in mehreren Kapiteln benötigt werden. Viele Definitionen werden sowohl auf dem Kreis als auch auf der Kugeloberfläche bezogen betrachtet. Es wird noch diskutiert, zum welchen Zweck man das Voronoi-Diagramm und die Delaunay-Zerlegung benutzen kann.

Eine interessante Frage beschäftigt uns zuerst: wo kann der neue Punkt im Kreis bzw. auf der Kugeloberfläche liegen, das heisst wie findet man die Position für den aktuell beweglichen Punkt? Welche Punkte sollte man in jedem Iterationsschritt betrachten? Sollte man alle Punkte im Spielraum überprüfen oder reicht es, nur solche Punkte zu betrachten, die bestimmten Eigenschaften haben? Diese Frage lässt sich bejahen. Es gibt solche Eigenschaften sowohl im Kreis als auch auf der Kugeloberfläche. Folgendes stellte sich heraus: um die neue Position des beweglichen Punktes zu finden, reicht es nur die Voronoi-Knoten des Voronoi-Diagramms der festen Punkte und im Kreis noch die Punkte am Kreisrand zu betrachten.

Ein weiteres interessantes Problem, das bei der Visualisierung des Spieles auftritt, ist die rechnerisch effiziente Berechnung des Polyeders, eigentlich die Bestimmung der konvexen Hülle. Wie kann man die Polyeder-Kanten, das heißt die Punktpaare, die bei der Visualisierung verbunden werden sollen, finden, wenn man nur die Eckpunkte auf der Kugeloberfläche kennt?

Eine Möglichkeit, das Polyeder zu bestimmen, besteht darin, dass man mehrere Halbraumtests mit Berechnung von Determinanten durchführt. Dieser Weg wurde nicht in dieser Diplomarbeit untersucht bzw. implementiert. Zum Glück können wir das schon zur Verfügung stehende Voronoi-Diagramm zu diesem Zweck auch gebrauchen:

Zwei Punkte auf der Kugeloberfläche sind genau dann Voronoi-benachbart, das heißt sie sind mit einer Delaunay-Kante verbunden, wenn sie im Polyeder mit einer Kante verbunden sind. Diese schöne Eigenschaft ist für die Kugel spezifisch und lässt sich nicht auf andere (konvexe) Flächen verallgemeinern.

Im dritten Kapitel wird zuerst der eindimensionale Fall untersucht, wobei die Punkte nur am Kreisrand liegen dürfen. Das Voronoi Spiel in diesem Fall besitzt die folgende Eigenschaft: Der Ablauf des Spieles kann in zwei Phasen gegliedert werden. Wir wissen, dass jeder Punkt zwei Voronoi-Nachbarn am Kreisrand hat.

In der ersten Phase können die Nachbarn der Punkte ausgetauscht werden. Nach endlich vielen Iterationsschritten gibt es in der Nachbarschaftsbeziehung aber keine Änderung mehr, und das Verfahren konvergiert. Dieser stabile Zustand ist dadurch charakterisiert, dass die Winkel zwischen den benachbarten Punkten bezüglich des Zentrums $\frac{2\pi}{n}$ sind, also die Punkte voneinander äquidistant liegen.

Dieses Resultat brauchen wir für den Fall, wenn die Punkte im Kreis liegen, das heißt die Punkte dürfen sowohl im Inneren des Kreises als auch am Kreisrand liegen. Die Frage lautet hier: bekommen wir dasselbe Ergebnis wie am Kreisrand? Die Situation ist viel komplexer und die stabilen Konfigurationen können nicht so leicht charakterisiert werden. Wenn die Anzahl der Punkte zwischen zwei und fünf liegt, müssen sich die Punkte in einer stabilen Position am Kreisrand äquidistant befinden. Falls ein Punkt im Zentrum und fünf Punkte am Kreisrand voneinander äquidistant liegen, ist dieser Zustand stabil, aber beim beliebigen Start erreicht das Spiel nicht unbedingt diesen Zustand, also kann man auch andere stabile Konfigurationen finden.

Für größere Punktmengen enthalten die stabilen Konfigurationen immer Punkte ausserhalb des Randes. Bis zu neun Punkten kann man sogar zeigen, dass einer dieser Punkte der Mittelpunkt sein muss, wobei die restlichen Punkte regelmäßig am Rand verteilt sein müssen.

Ein berühmter und spannender Bereich der diskreten Geometrie sind die Kugelpackungen. Es wird behauptet, dass von allen unendlichen Kreispackungen die hexagonal angeordnete Kreispackungen die größte Packungsdichte besitzt. Das heißt sie nutzt die zur Verfügung stehende Fläche am besten aus. Für drei und sieben Kreise liefert sicher die hexagonale Anordnung die maximale Packungsdichte, aber für beliebigen endlichen Kreispackungen ist es noch fraglich. Kann das Voronoi Spiel dabei helfen, mehr von dieser Theorie zu verstehen? Für drei und sieben Punkte mit umschriebenen Kreisen, (die dasselbe Radius haben und sich nicht überlappen dürfen und deren Mittelpunkte im Einheitskreis liegen), liefert das Voronoi Spiel hexagonale Kreispackungen mit der größten Packungsdichte. Im Allgemeinen liegen die Punkte im stabilen Zustand quasi regelmäßig verteilt im Kreis. Lässt sich die Delaunay-Zerlegung der Punkte nach dem Spielablauf darstellen, scheinen die Punkte, nach unseren Rechnerexperimenten, in einem hexagonal ähnlichen Netz zu liegen. Der Mittelwert der Packungsdichte des Voronoi Spieles beträgt etwas mehr als 0,7. Dieser Wert liegt nahe bei dem Mittelwert der endlichen hexagonalen Kreispackungen.

Im vierten Kapitel wird das Voronoi Spiel auf der Kugeloberfläche untersucht. Hier werden nicht nur die Punkte dargestellt, sondern auch das Polyeder, das von den Punkten eindeutig bestimmt ist. An dieser Stelle interessieren wir uns besonders für reguläre und halbrekuläre Eigenschaften des Polyeders. Am Anfang dieses Kapitels findet man eine Zusammenstellung von regulären (Platonische Körper) und halbrekulären (Archimedische Körper, Prismen und Antiprismen) Polyedern. Es werden noch einige Regeln eingeführt, die in der sphärischen Trigonometrie

gelten.

Mehrere Fragen ergeben sich daraus: Gibt es stabile reguläre bzw. halbrekuläre Polyeder? Werden bei beliebigem Start, das heißt die Punkte werden zufällig auf die Kugeloberfläche gesetzt, diese ein stabile Polyeder erreichen? Falls für n Punkte ein (halb)reguläres Polyeder gibt, ist diese die einzige stabile Konfiguration?

Wir haben experimentell Folgendes festgestellt: es gibt mehrere stabile reguläre bzw. halbrekuläre Polyeder, aber einige von ihnen sind instabil. In den meisten Fällen werden die gefundenen stabilen Körper beim beliebigen Start nicht erreicht. Aber die stabilen Polyeder, wie das Tetraeder (mit vier Ecken), das Oktaeder (mit sechs Ecken), das Antiprisma (mit acht Ecken) und das Ikosaeder (mit zwölf Ecken) scheinen beim zufälligen Start sehr gut angenähert zu sein.

Die Antwort ist für die letzte Frage eindeutig nein: sogar für vier Punkte gibt es eine stabile Konfiguration, die nicht ein reguläres Tetraeder ist. Dieses Gegenbeispiel, ein Quadrat, ist aber nicht generisch und nach der kleinsten Perturbation konvergiert das Spiel gegen ein reguläres Tetraeder. Eine ähnliche Aussage scheint für andere (halb)reguläre Polyeder zu gelten: andere, stabile, aber nicht generische Konfigurationen sind zum Beispiel das Prisma (mit sechs Ecken), der Würfel (mit acht Ecken) und das Kuboktaeder (mit zwölf Ecken).

Im Allgemeinen liegen die Punkte im stabilen Zustand quasi regelmäßig auf der Kugeloberfläche verteilt. Sie scheinen wie im Fall des Kreises in einem sphärischen hexagonal ähnlichen Netz zu liegen. Der durchschnittliche Eckengrad ist zwischen fünf und sechs, wenn die Anzahl der Punkte mindestens zwölf beträgt, und der Mittelwert der sphärischen Packungsdichte vom Voronoi Spiel beträgt 0,76.

Im fünften Kapitel suchen wir nach Funktionen der Konfigurationen, die während des Voronoi Spieles monoton wachsend oder fallend und noch dazu eventuell differenzierbar sind. Sie charakterisieren das Verfahren (Voronoi Spiel), und die Extrempunkte solcher Funktionen sind gerade die stabilen Konfigurationen. Aus diesem Grund nennen wir sie **Beschreibungsfunktionen**.

Zuerst geben wir eine schöne nützliche Beschreibungsfunktion für den eindimensionalen Fall an, die während des Spieles monoton fällt. Der Gradient der Beschreibungsfunktion kann dabei helfen, stabile Konfigurationen zu finden. Der Gradient ist genau dann 0, falls die Punkte voneinander äquidistant am Kreisrand liegen. Es bleibt noch die Frage offen, ob es solche (differenzierbare) Beschreibungsfunktionen auch für die zweidimensionalen Fälle (Kreis, Kugeloberfläche) gibt.

Für beide Fälle liefert der minimale Abstand zwischen den Punkten eine geeignete Beschreibungsfunktion. Der minimale Abstand lässt sich durch das folgende Verfahren berechnen: Es gibt endlich viele (n) Punkte innerhalb des Kreises oder auf der Kugeloberfläche. Man soll alle Euklidische Abstände zwischen zwei beliebigen Punkte berechnen und von diesen Werten das Minimum nehmen. In

beiden Fällen hat die Größe des minimalen Abstandes die Komplexität $O(\frac{1}{\sqrt{n}})$, vermutlich $\Theta(\frac{1}{\sqrt{n}})$.

Wir gehen noch auf das Konvergenzverhalten des Verfahrens sowohl theoretisch als auch numerisch ein. Wir haben in unserem Testprogramm die Anzahl der Iterationsschritte gemessen, die notwendig ist, um eine stabile Konfiguration mit einer vorgegebener Genauigkeit anzunähern. Unsere Experimente ergaben, dass die Anzahl der notwendigen Schritte zwischen $O(n \log n)$ und $O(n^2)$ liegen. Diese Beobachtung weist auf die Konvergenz des Verfahrens hin.

Im sechsten Kapitel wird dargestellt, wie der Algorithmus auf zwei verschiedene Weisen implementiert werden kann. In der ersten Implementierung werden in jedem Iterationsschritt in einem fest gewählten Gitter alle Punkte betrachtet, bei der zweiten nur die Voronoi-Knoten des Voronoi-Diagramms der festen Punkte sowie die Randpunkte im Kreis. Weiter wird auf die Implementierung der Ermittlung der Voronoi-Knoten und Voronoi-Nachbarschaft eingegangen und die angewandten Generierungsmethoden für reguläre und halbrekuläre Polyeder auf der Kugeloberfläche beschrieben.

Im siebten Kapitel stellen wir zwei neulich entwickelte Java Applets (für den Kreis und die Kugeloberfläche) vor, mit denen weitere praktische Experimente auf einer einfachen, benutzerfreundlichen Weise durchgeführt werden können.

Kapitel 2

Das Voronoi Spiel

2.1 Einführung mit einigen Definitionen

Einige Definitionen, auf dem d -dimensionalen Raum \mathbb{R}^d verallgemeinert, stammen aus dem Buch: R. Klein: Algorithmische Geometrie.

Definition 2.1.1 [2] **Der Euklidische Abstand** zwischen zwei Punkten $p = (p_1, \dots, p_d) \in \mathbb{R}^d$ und $q = (q_1, \dots, q_d) \in \mathbb{R}^d$ ist durch:

$$|pq| = \sqrt{(p_1 - q_1)^2 + \dots + (p_d - q_d)^2}$$

definiert

Auf der Kugeloberfläche $S_2 = \{x \in \mathbb{R}^3; |x| = 1\}$ kann man einen **sphärischen Abstand** $|\cdot|_k$ definieren, der die Länge des kürzesten Weges zwischen zwei Punkten angibt, eigentlich auch im Euklidischen Maß angegeben. Wir betrachten die Einheitskugel, und sei M der Mittelpunkt der Kugel, liege $p, q \in S_2$ und bezeichne α den Winkel $\angle pMq$. Jeder Großkreisbogen ist die kürzeste Verbindung zweier Punkte p und q auf der Kugeloberfläche. Deswegen gilt $|pq| = 2 \sin \frac{\alpha}{2}$. Daraus folgt:

$$|pq|_k = \alpha = 2 \arcsin \frac{|pq|}{2}$$

Definition 2.1.2 [2] **Der Bisektor** von zwei Punkten $p, q \in \mathbb{R}^d$ ist durch:

$$B(p, q) = \{x \in \mathbb{R}^d; |px| = |qx|\}$$

definiert, also alle Punkte des \mathbb{R}^d , die zu p und q denselben Abstand haben. Der Bisektor zerlegt den Raum \mathbb{R}^d in zwei offene Hyperebenen:

$$D(p, q) = \{x \in \mathbb{R}^d; |px| < |qx|\} \quad \text{und} \quad D(q, p) = \{x \in \mathbb{R}^d; |px| > |qx|\}$$

Auf der Kugeloberfläche S_2 kann man einen **sphärischen Bisektor** $B_k(\cdot)$ definieren, sogar auf zwei verschiedene Weisen, liefern sie aber denselben sphärischen Bisektor. Wir können uns wieder auf die Einheitskugel beschränken. Im ersten Fall kann man den obigen Bisektor, (eine Ebene in \mathbb{R}^3), mit der Kugeloberfläche schneiden. Hier liegen die Bisektorpunkte in S_2 , aber es werden alle Punkte in \mathbb{R}^3 betrachtet:

$$B_k(p, q) = B(p, q) \cap \{x \in \mathbb{R}^3; x \in S_2\}$$

Im zweiten Fall kann man den sphärischen Bisektor mit dem sphärischen Abstand $|\cdot|_k$ definieren. Hier liegen sowohl die Bisektorpunkte als auch die untersuchten Punkte in S_2 .

$$B_k(p, q) = \{x \in S_2; |px|_k = |qx|_k\}$$

Der so entstandene sphärische Bisektor ist ein Großkreis der Kugel, der die Kugel halbiert, das heißt die Fläche von $D_k(\cdot)$ ist eine Halbkugel:

$$D_k(p, q) = \{x \in S_2; |px|_k < |qx|_k\} \quad \text{und} \quad D_k(q, p) = \{x \in S_2; |px|_k > |qx|_k\}$$

Definition 2.1.3 [2] Sei eine Punktmenge $S = \{p_1, \dots, p_k\}$ in \mathbb{R}^d gegeben.

Die Voronoi-Region von einem Punkt p bezüglich S ist durch:

$$VR(p, S) = \bigcap_{q \in S \setminus \{p\}} D(p, q)$$

definiert

Auf der Kugeloberfläche definieren wir **die sphärische Voronoi-Region** als Schnitt von mehreren Halbkugeln ($D_k(\cdot)$). Es ist ein sphärisches Vieleck, es kann sogar ein Zweieck sein, (siehe Abbildung 2.2).

Definition 2.1.4 [2] Die Voronoi-Region $VR(p, S)$ von p besteht aus allen Punkten des angegebenen Raumes (Fläche), denen p näher ist als irgendein anderer Punkt aus S . Entfernt man alle Voronoi-Regionen bezüglich S , bleiben diejenige Punkte übrig, die keinen eindeutigen, sondern zwei oder mehr nächste Nachbarn in S besitzen. Diese Punktmenge wird **das Voronoi-Diagramm** $V(S)$ von S genannt, (siehe Abbildung 2.1).

Diese zum Beispiel ist in der Ebene ein ebener Graph mit Voronoi-Kanten und Voronoi-Knoten (die Eckpunkte der Voronoi-Kanten). Auf der Kugeloberfläche besteht **das sphärische Voronoi-Diagramm**, dieser sphärischer Graph aus Großkreisbögen, (die Voronoi-Kanten), und aus den Eckpunkten der Großkreisbögen, (die Voronoi-Knoten).

Definition 2.1.5 [2] Sei nun eine Punktmenge $S = \{p_1, \dots, p_k\}$ in der Ebene gegeben und sei $V(S)$ ihr Voronoi-Diagramm. Zwei Punkte $p, q \in S$ werden durch das Liniensegment pq miteinander verbunden, falls ihre Voronoi-Regionen

$VR(p, S)$ und $VR(q, S)$ an einer gemeinsame Voronoi-Kante in $V(S)$ angrenzen. Alle sämtliche solche Liniensegmente bezüglich S bilden die **Delaunay-Zerlegung** $DT(S)$ der Punktmenge S . Das Liniensegment pq heißt eine *Delaunay-Kante*, (siehe Abbildung 2.1).

Auf der Kugeloberfläche definiert man **die sphärische Delaunay-Zerlegung** ganz analog aus dem sphärischen Voronoi-Diagramm. Die sphärischen Delaunay-Kanten sind Großkreisbogen.

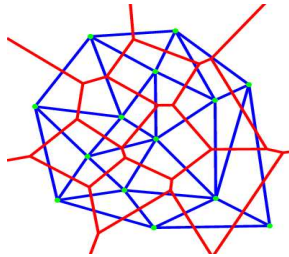


Abbildung 2.1: Voronoi-Diagramm und Delaunay-Zerlegung in der Ebene [19]

Jetzt wird **der iterative Algorithmus - das Voronoi Spiel** allgemein vorgestellt:

Seien n Punkte $P = \{p_0, p_2, \dots, p_{n-1}\}$ in einem kompakten, zusammenhängenden Gebiet $\Omega \in \mathbb{R}^d$ gegeben. Sei ein Zähler c gegeben, der am Anfang den Wert 0 hat. In jedem Iterationsschritt muss der Punkt $p_{(c \bmod n)}$ so neu gewählt werden, dass er am weitesten von allen anderen Punkten in Ω liegt, das heißt der Euklidische Abstand zu seinem nächsten Nachbarn maximiert, und nachher der Zähler $c = c + 1$ inkrementiert werden soll. Wenn der neue Abstand nicht größer als der alte Abstand gewählt werden kann, darf sich der Punkt nicht bewegen. In einem Iterationsschritt kann sich also ein Punkt bewegen, und die restlichen $n-1$ Punkte halten wir fest.

Was passiert mit den Punkten in Ω , falls $\lim c \rightarrow \infty$ ist?

In dieser Diplomarbeit werden die folgenden zwei Spezialfälle untersucht:

1. Der Abschluss eines Kreises in \mathbb{R}^2 , also liegen die Punkte innerhalb oder am Rand des Kreises. o.B.d.A betrachten wir den Einheitskreis.
2. Der Rand einer Kugel in \mathbb{R}^3 , also liegen die Punkte auf der Kugeloberfläche. O.B.d.A betrachten wir die Einheitskugel.

Auf der Kugeloberfläche kann man den Euklidischen Abstand $|\cdot|$ oder den sphärischen Abstand $|\cdot|_k$ nehmen. Sie liefern dasselbe Ergebnis. Es liegt daran, dass die Funktion $f : \alpha \mapsto 2 \sin \frac{\alpha}{2}$ im Bereich $\alpha \in [0, \pi]$ streng monoton wachsend ist.

Die Bisektoren sind Großkreise und die Voronoi Regionen sind sphärische Vielecke, begrenzt durch Großkreisbögen.

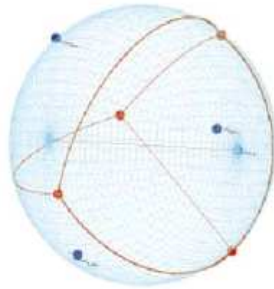


Abbildung 2.2: Voronoi-Region auf der Kugeloberfläche [7]

Spezialfall von oben: Am Anfang kann das Gebiet Ω leer sein, und in der ersten Runde (n Iterationsschritte) werden die Punkte positioniert. Zum Beispiel der erste Punkt p_0 im Fall 1. (das Gebiet Ω ist der Kreis) kann in der ersten Iteration auf den Rand gelegt werden. Die Position des zweiten Punktes hängt nur von der Position des ersten Punktes ab. Die Position des dritten Punktes hängt von der Position der ersten und zweiten Punkte ab und so weiter. Ab der zweiten Runde verhält sich der Algorithmus wie oben, das heißt die neue Position eines Punktes hängt von allen anderen Punkten ab.

Definition 2.1.6 *Zwei Punkte aus $P = \{p_0, \dots, p_{n-1}\}$ seien **(Voronoi-)Nachbarn**, oder **(Voronoi-)benachbart**, falls sie eine gemeinsame Voronoi-Kante haben, (also ein gemeinsamer Voronoi-Knoten reicht nicht aus). Mit anderen Worten, sind zwei Punkte in der (sphärischen) Delaunay-Zerlegung verbunden, sind sie Nachbarn.*

Definition 2.1.7 (Voronoi-)Nachbarschaftsbeziehung: *Es gibt für alle Punkte an, mit welchen anderen Punkten sie benachbart sind. Diese Beziehung kann in einer quadratischen Matrix $N \in \mathbb{N}^{n \times n}$ dargestellt werden. Die Einträge der (Nachbarschafts-)Matrix sind $n_{ik} = 1$, falls die Punkte p_i und p_k benachbart sind, sonst $n_{ik} = 0$ und $n_{ii} = 0 \quad \forall i \in \{1, \dots, n\}$.*

Definition 2.1.8 Ein Iterationsschritt ist ein **Sprung**, falls sich die Nachbarschaftsbeziehung ändert.

Definition 2.1.9 Die Punktmenge ist durch $P = \{p_0, \dots, p_{n-1}\}$ gegeben. Die Konfiguration der Punkte sind **stabil** oder die Punkte sind **im stabilen Zustand**, wenn sie sich während n nacheinanderfolgenden Iterationsschritten, das heißt während einer Runde, nicht bewegen.

Definition 2.1.10 Der **Minimaler Abstand** einer Punktmenge $P = \{p_0, \dots, p_{n-1}\}$ ist der kleinste Abstand zwischen zwei beliebigen Punkten, das heißt

$$D(P) = \min_{i,k=0,\dots,n-1,i \neq k} |p_i p_k|$$

2.2 Anwendung des Voronoi-Diagramms

Definition 2.2.1 Sei eine Punktmenge $P = \{p_0, \dots, p_{n-1}\}$ im Gebiet Ω gegeben. In jedem Iterationsschritt kann sich ein Punkt bewegen und die restlichen $n - 1$ Punkte halten wir fest. Wir müssen die neue Position des beweglichen Punktes, das heißt den **neuen Punkt**, im Gebiet Ω finden.

Im Kreis

Betrachte das Gebiet, den Einheitskreis mit seinem Inneren:

$$\Omega = \{(p_x, p_y)^t \mid p_x^2 + p_y^2 \leq 1, p_x \in \mathbb{R}, p_y \in \mathbb{R}\}.$$

Wo kann der neue Punkt liegen? Das heißt wie finden wir die neue Position des beweglichen Punktes? Die ganze Problematik erinnert uns an das Störquellen Problem:

Störquellen Problem [2] Störquellen Problem mit dem größten leeren Kreis: Sei ein Gebiet A als ein konvexes Polygon mit m Ecken und seien n Punkte $S = \{p_0, \dots, p_{n-1}\}$ in der Ebene als Störquellen gegeben. Die Aufgabe ist einen maximalen leeren Kreis zu finden, dessen Mittelpunkt in A liegt. Dazu erwähne ich ein Lemma aus dem Buch R.Klein: Algorithmische Geometrie.

Lemma 2.2.2 [2] Sei $C(x)$ der größte Kreis mit in Polygon A liegenden Mittelpunkt x , der keinen Punkt aus S in seinem Inneren enthält. Dann ist x ein Voronoi-Knoten von $V(S)$, ein Schnittpunkt einer Voronoi-Kante mit dem Rand von A oder ein Eckpunkt von A .

Im Grunde genommen brauchen wir eine ähnliche Behauptung:

Satz 2.2.3 Sei p der aktuelle bewegliche Punkt, und wir halten die restlichen $n - 1$ Punkte fest. Für die Wahl des neuen Punktes gibt es folgende Kandidaten:

1. Der neue Punkt ist ein Voronoi-Knoten des Voronoi-Diagramms der festen Punkte.
2. Der neue Punkt liegt am Kreisrand.
3. Die Position von p ändert sich nicht.

Beweis vom Satz 2.2.3 Nach Lemma 2.2.2 sind die Störquellen die Punkte im Kreis und das Polygon ist ein reguläres Polygon mit unendlich vielen Ecken, also ein Kreis. Der erste Punkt ist trivial, für die zweite muss man Folgendes überlegen: wenn n beliebig groß wird, dann konvergiert das reguläre Polygon gegen einen Kreis und das Polygon besteht nur aus Eckpunkten, welche den Kreisrand bilden. Der dritte Punkt folgt aus den Regeln des Voronoi Spieles.

Auf der Kugeloberfläche

Betrachte das Gebiet, die Oberfläche der Einheitskugel:

$$\Omega = \{(p_x, p_y, p_z) \mid p_x^2 + p_y^2 + p_z^2 = 1, p_x \in \mathbb{R}, p_y \in \mathbb{R}, p_z \in \mathbb{R}\}.$$

Wir haben dieselbe Frage wie im Fall des Kreises: Wie finden wir die neue Position des beweglichen Punktes? Diese Frage zu beantworten brauchen wir eine ähnliche Behauptung wie beim Kreis.

Satz 2.2.4 Sei p der aktuelle bewegliche Punkt, und wir halten die restlichen $n - 1$ Punkte fest. Der neue Punkt ist ein Voronoi-Knoten des sphärischen Voronoi-Diagramms der restlichen $n - 1$ festen Punkte oder die Position von p ändert sich

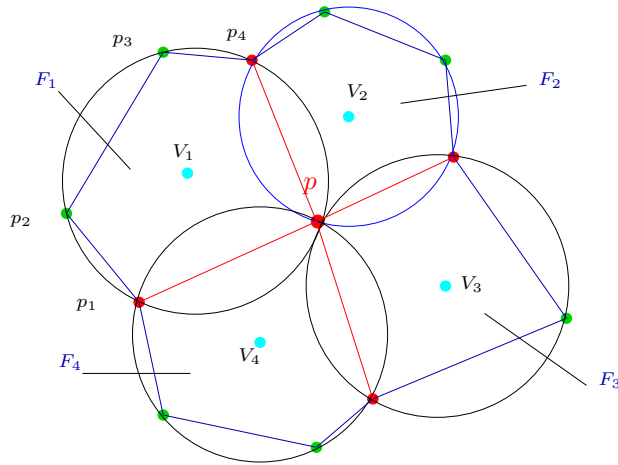


Abbildung 2.3: Jede Polyeder-Kante ist eine Delaunay-Kante

nicht, falls der neue Abstand nicht größer als der alte Abstand gewählt werden kann.

Wir nehmen an, dass der neue Abstand größer als der alte ist. Man kann in diesem Fall die Behauptung so formulieren: Sei eine endliche Punktmenge P und ihr sphärisches Voronoi-Diagramm $P(V)$ auf der Kugeloberfläche gegeben. Sei $C(p)$ der größte sphärische Kreis mit dem Mittelpunkt p , der keinen Punkt aus P in seinem Inneren enthält. Dann ist p ein Voronoi-Knoten des sphärischen Voronoi Diagramms.

Beweis vom Satz 2.2.4 Sei p auf der Kugeloberfläche beliebig und $C(p)$ ein sphärischer Kreis um p , der kein $p \in P$ enthält.

Zuerst wird $C(p)$ bei festem Mittelpunkt solange vergrößert, bis der sphärische Kreisrand einen Punkt aus P erreicht. Falls jetzt drei Punkte aus P auf dem Rand von $C(p)$ liegen, dann ist p ein Voronoi-Knoten. Falls nur zwei Punkte p_1 und p_2 auf dem Rand liegen, dann liegt p auf einer sphärischen Voronoi-Kante von $V(P)$, bzw. auf dem Bisektor $B_k(p_1, p_2)$. Nun können wir p entlang $B_k(p_1, p_2)$ verschieben, so dass p vom Schnittpunkt des Bisektors und des Großkreisbogen $\widehat{p_1 p_2}$ immer weiter entfernt ist. Während dieser Verschiebung berührt weiter $C(p)$ die Punkte p_1 und p_2 . Wir schieben p so lange, bis der sphärische Kreis $C(p)$ einen dritten Punkt aus P berührt, das heißt p ein Voronoi-Knoten von $V(P)$ wird. Damit haben wir den Satz bewiesen. \square

Wie kann man das Polyeder bestimmen? Mit anderen Worten, wie lässt sich die konvexe Hülle der Punkte effizient berechnen, falls man die Eckpunkte und ihr Voronoi-Diagramm kennt?

Satz 2.2.5 Das Delaunay-Polyeder: *Zwei Punkte auf der Kugeloberfläche sind genau dann Voronoi-benachbart (es gibt gemeinsame Voronoi-Kante), falls sie mit einer Kante im Polyeder verbunden sind.*

Bevor wir mit dem Beweis beginnen, möchten wir die Unterschiede zwischen Polyeder im Kugel und sphärischen Polyeder auf der Kugeloberfläche und zwischen Delaunay-Zerlegung und sphärischen Delaunay-Zerlegung der Punkte betrachten.

Ein Polyeder (siehe auch in Kapitel 4) ist ein zusammenhängendes Gebiet $P \in \mathbb{R}^3$, dessen Oberfläche aus ebenen Vielecken besteht. Polyeder haben Ecken, Kanten und Facetten (Seitenflächen). Auf der Kugeloberfläche kann man das sphärische Polyeder zeichnen, indem man alle geradlinigen Polyeder-Kanten mit den Eckpunkten p und q folgendermaßen ersetzt: Die sphärische Polyeder-Kante von p und q liegt jetzt auf der Kugeloberfläche und sie ist der kürzeste Weg zwischen p und q , das heißt sie ist ein Großkreisbogen.

Das Voronoi-Diagramm der Punkte ist im Raum \mathbb{R}^3 haben folgende Voronoi-Kanten: sie sind konvexe ebene Vielecke oder Vielecke ohne eine Seite, das heißt seine Fläche ist unendlich groß, wir nennen es offenes Vieleck. Schneidet man im Fall der Kugeloberfläche alle offenen Dreiecke (es gibt nur solche) mit der Kugeloberfläche, erhält man das sphärische Voronoi-Diagramm.

Nach der Konstruktion gilt: Zwei Punkte sind genau dann Voronoi-Nachbarn bezüglich des Voronoi-Diagramms in \mathbb{R}^3 , wenn sie bezüglich des sphärischen Voronoi-Diagramms Voronoi-Nachbarn sind. Aus diesem Grund folgt, dass die Delaunay-Zerlegung in \mathbb{R}^3 entspricht der sphärischen Delaunay-Zerlegung und zwischen zwei benachbarten Punkten p und q läuft die Delaunay-Kante geradlinig, und die sphärische Delaunay-Kante auf der Kugeloberfläche entlang dem Großkreisbogen zwischen p und q .

Beweis vom Satz 2.2.5 Betrachten wir einige Punkte $\tilde{P} := (p_{k_1}, \dots, p_{k_m}) \subset P$ zuerst, die in einer Ebene in \mathbb{R}^3 auf der Kugeloberfläche liegen, und ihr sphärisches Voronoi-Diagramm. Die sphärischen Bisektoren treffen sich in einem Punkt, da die Punkte auf einem (sphärischen) Kreis liegen. Daraus folgt, dass eine (sphärische) Polyeder-Kante eine (sphärische) Delaunay-Kante ist und umgekehrt, aber Delaunay-Kante darf keine (sphärische) Diagonale sein. (*) Jetzt liegen alle Punkte allgemein auf der Kugeloberfläche.

- jede Polyeder-Kante ist eine Delaunay-Kante.

Betrachte beliebig einen Eckpunkt p vom Polyeder P . Betrachte dazu beliebig eine Facette, zum Beispiel F_1 , die diesen Punkt als Ecke besitzt. Die

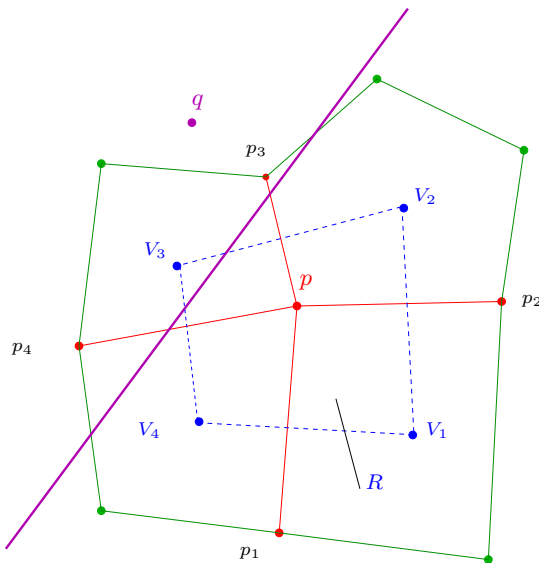


Abbildung 2.4: Jede Delaunay-Kante ist eine Polyeder-Kante

Punkte dieser Facette, liegen auf einem (sphärischen) Kreis. Wegen (*) sind die zwei Polyeder-Kanten pp_1 und pp_4 der Facette F_1 Delaunay-Kanten und die Diagonale der Facette, ausgehend vom Eckpunkt p , keine Delaunay-Kanten. Da der Eckpunkt und die Facette beliebig waren, haben wir alle Polyeder-Kanten betrachtet, (siehe Abbildung 2.3).

- jede Delaunay-Kante ist eine Polyeder-Kante.

Wegen (*) reicht zu zeigen, dass eine Delaunay-Kante nicht innerhalb vom Polyeder läuft (eine Delaunay-Kante darf keine Diagonale einer Facette sein).

Betrachte wieder beliebig einen Eckpunkt p und das sphärische Polyeder. Sei R das sphärische Gebiet, begrenzt von den sphärischen Bisektoren des Punktes p und eines Polyeder-Nachbarn (p_1, p_2, p_3, p_4) von p . Die Eckpunkte vom Gebiet R sind Voronoi-Knoten im sphärischen Voronoi-Diagramm. Diese Voronoi-Knoten (V_1, V_2, V_3, V_4) liegen in den sphärischen Facetten mit Eckpunkt p , (siehe Abbildung 2.4).

Sei jetzt angenommen, dass eine Delaunay-Kante mit Eckpunkten p und q innerhalb vom Polyeder läuft. Der sphärische Bisektor von p und q schneidet das Gebiet R , deswegen wird ein Voronoi-Knoten, der in der Facette mit Eckpunkt p liegt, abgeschnitten, das heißt die entsprechende sphärische Delaunay-Kante schneidet mindestens zwei sphärischen Voronoi-Kanten. Es führt zum Widerspruch. \square

Der neue Punkt kann trivialerweise überall im Kreis bzw. auf der Kugeloberfläche liegen. Diese Tatsache wird in der ersten Implementierung ausgenutzt, (siehe Kapitel 6), hier wird kein Voronoi-Knoten-(Randpunkt-)Test durchgeführt. Die Sätze 2.2.3 und 2.2.4 erlauben uns aber, dass man nur die Voronoi-Knoten des Voronoi-Diagramms der festen Punkte und im Kreis noch die Kreisrandpunkte überprüfen soll. In diesem Fall wird in der Implementierung Voronoi-Knoten-(Randpunkt-)Test benutzt.

Kapitel 3

Ergebnisse im Kreis

3.1 1-dimensionaler Fall am Kreisrand

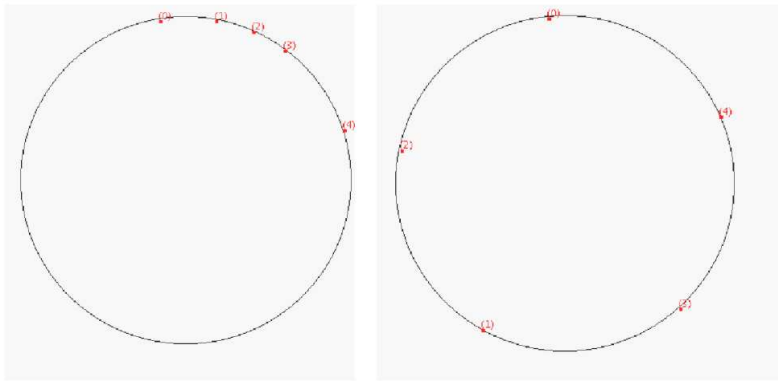


Abbildung 3.1: Punkte am Kreisrand vor dem Spiel und nach dem Spiel

Betrachte das Gebiet, den Rand des Einheitskreises:

$$\Omega = \{(x, y) \mid x^2 + y^2 = 1, x \in \mathbb{R}, y \in \mathbb{R}\}.$$

Alle Punkte liegen am Kreisrand. Jeder Punkt hat 2 Voronoi-Nachbarn. Der einzige Voronoi-Knoten ist das Zentrum des Kreises. Bezeichne Z das Zentrum des Kreises. Hierzu bemerken wir, dass das Maß für den Abstand zwischen zwei Punkten p und q durch den Winkel pZq auch zu definieren ist. Es liegt daran, dass die Funktion $f : \mathbb{R} \rightarrow \mathbb{R}, \alpha \mapsto 2 \sin \frac{\alpha}{2}$ im Bereich $\alpha \in [0, \pi]$ streng monoton wachsend ist. Nun stellt sich die Frage: Was passiert mit den Punkten während des Spieles?

Satz 3.1.1 *Der Ablauf des Spieles kann in zwei Phasen gegliedert werden, dass heißt es gibt ein $k \in \mathbb{N}$, so dass gilt : falls Zähler $c < k$ ist, dann kann ein Sprung vorkommen, also die Nachbarschaftbeziehung kann sich ändern, sonst falls $c \geq k$ ist, dann ist kein Sprung mehr möglich, also die Nachbarschaftbeziehung bleibt unverändert.*

Beweis vom Satz 3.1.1 Über Lemma 3.1.2 und Lemma 3.1.4.

Lemma 3.1.2 *Betrachte alle Winkel zwischen den benachbarten Punkten bezüglich des Zentrums, davon sei der größte Winkel mit α_g bezeichnet. Falls $\alpha_g < \frac{2\pi}{n-1}$ ist, dann kann es kein Sprung mehr vorkommen und α_g wird kleiner und damit gilt weiter : $\alpha_g < \frac{2\pi}{n-1}$, (siehe Abbildung 3.2 z.B. $\alpha_g = \alpha_4$).*

Bemerkung 3.1.3 *Es gilt immer $\frac{2\pi}{n} \leq \alpha_g$*

Beweis vom Lemma 3.1.2 Seien die Winkel $(\alpha_1, \alpha_2, \dots, \alpha_n)$ zwischen den benachbarten Punkten mit folgender Eigenschaft gegeben: $\alpha_i \leq \alpha_n = \alpha_g \quad \forall i \in 1, \dots, n-1$ (*), also bezeichnen wir den größten Winkel mit α_g . Seien o.B.d.A die Winkel α_1 und α_2 benachbart und sie werden beim nächsten Iterationsschritt betrachtet.

Zuerst müssen wir zeigen, dass der größte Winkel bei jedem Iterationsschritt kleiner oder gleich ist:

1. mit Sprung : Hier muss vor dem Iterationsschritt gelten, dass $\alpha_1 + \alpha_2 \leq \alpha_g$ ist. Daraus folgt, dass nach dem Iterationsschritt drei neuen Winkel, $\alpha_s := \alpha_1 + \alpha_2 \leq \alpha_g$ und zweimal $\frac{\alpha_g}{2}$, entstehen, und α_1, α_2 und α_g verschwinden, oder bei Gleichheit kann es keine Veränderung geben. (In der Java Implementierung wird keine Veränderung vorgenommen, falls $\alpha_1 + \alpha_2 = \alpha_g$ ist.)
2. ohne Sprung : Sei o.B.d.A $\alpha_1 \leq \alpha_2$. Hier muss vor dem Iterationsschritt gelten, dass $\alpha_1 + \alpha_2 > \alpha_g$ ist. Daraus folgt, dass nach dem Iterationsschritt zwei neuen und gleichen Winkel $\frac{\alpha_1 + \alpha_2}{2}$ entstehen, und α_1 und α_2 verschwinden. Es gilt $\frac{\alpha_1 + \alpha_2}{2} \leq \alpha_2 \leq \alpha_g$.

Jetzt sei $\alpha_s := \alpha_1 + \alpha_2 \leq \alpha_g$ und damit gilt $\alpha_s + \alpha_g \leq 2\alpha_g$. Aus diesem folgt, dass

$$\alpha_3 + \alpha_4 + \dots + \alpha_{n-1} \geq 2\pi - 2\alpha_g \quad (**)$$

Wegen der Voraussetzung (*) gilt

$$\alpha_3 + \alpha_4 + \dots + \alpha_{n-1} \leq (n-3)\alpha_g \quad (***)$$

Aus (**) und (***) folgt, dass bei jedem Iterationsschritt mit Sprung gelten muss:

$$\frac{2\pi}{n-1} \leq \alpha_g$$

Lemma 3.1.4 *Mit n Punkten ist die maximale Anzahl der Sprünge s endlich und $s \in O(\log n)$.*

Beweis vom Lemma 3.1.4 Der größte Winkel α_g wird bei jedem Iterationsschritt kleiner oder bleibt unverändert. Bei einem Sprung wird dieser Winkel halbiert. Aus Lemma 3.1.2 folgt, dass man die maximale Anzahl s der Sprünge bestimmen soll, wobei $\frac{2\pi}{n} \leq \alpha_g < \frac{2\pi}{n-1}$ ist, und mit $s-1$ Sprünge $\alpha_g \geq \frac{2\pi}{n-1}$ gelten soll. Es reicht den Grenzfall zu betrachten:

1. $\alpha_g = 2\pi$
2. Es kommen nur Sprünge vor, solange $\alpha_g \geq \frac{2\pi}{n-1}$ gilt.

$$\frac{2\pi}{n} \leq \frac{1}{2^s} \alpha_g < \frac{2\pi}{n-1} \Leftrightarrow \frac{2\pi}{n} \leq \frac{1}{2^s} 2\pi < \frac{2\pi}{n-1} \Leftrightarrow \frac{1}{n} \leq \frac{1}{2^s} < \frac{1}{n-1}$$

Durch Umrechnung erhalten wir:

$$\log(n-1) < s \leq \log n \quad \wedge \quad s = \lceil \log(n-1) \rceil$$

und dies bedeutet, dass $s \in O(\log n)$ ist.

Satz 3.1.5 *Falls es keinen Sprung mehr gibt, mit anderen Worten, wenn $c \geq k$ ist, sind die Punkte im stabilen Zustand oder konvergiert das Verfahren gegen solchen Zustand, welche dadurch charakterisiert ist, dass die Winkel zwischen den benachbarten Punkten bezüglich des Zentrums $\frac{2\pi}{n}$ sind, (siehe Abbildung 3.1).*

Beweis vom Satz 3.1.5 Falls kein Sprung mehr möglich ist, ist der Winkel $\alpha \in \{0, \dots, \pi\}$ zwischen den benachbarten Punkten. Betrachte einen beliebigen Punkt p am Rand. Die Nachbarn von p liegen näher zu p als die andere Punkte, da der Winkel bezüglich des Zentrums zwischen zwei benachbarten Punkten kleiner ist als der zwischen nicht benachbarten Punkten.

Daraus folgt, dass man in jedem Iterationsschritt für die neue Position von p den Winkel zwischen den Nachbarn von p halbieren soll. Sei jetzt o.B.d.A. die Reihenfolge der Punkte im Gegen-Uhrzeiger-Sinn gleich wie am Anfang des Spieles. Sonst kann man eine Permutationmatrix anwenden.

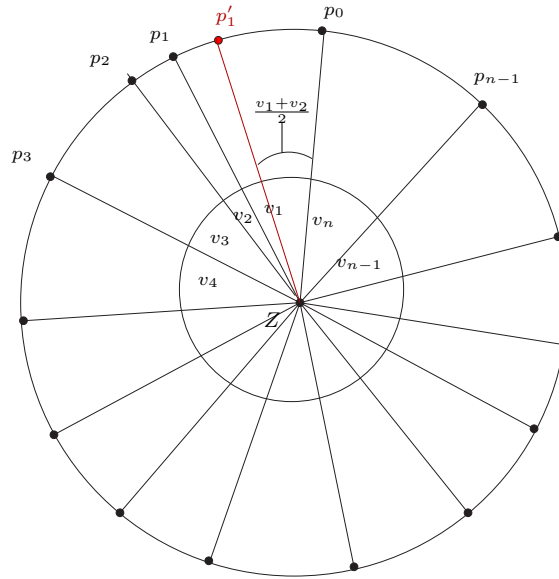


Abbildung 3.2: Winkelhalbierung

Wir definieren einen Vektor $v \in \mathbb{R}^n$, wobei v_i der Winkel zwischen den benachbarten Punkten p_{i-1} und p_i bezüglich des Zentrums Z für $i \in \{1, \dots, n-1\}$ und v_n der Winkel zwischen den benachbarten Punkten p_{n-1} und p_0 bezüglich des Zentrums angibt.

Ein Iterationsschritt besteht jetzt aus zwei Schritten:

1. Winkelhalbierung (siehe Abbildung 3.2)
2. Umnummerierung der Punkte, um jedesmal den ersten und zweiten Eintrag des Vektors v neu zu berechnen.

Die beiden Schritten sind lineare Abbildungen, also mit geeignete Matrix darstellbar. Die Winkelhalbierung sei durch die Matrix $A \in \mathbb{R}^{n \times n}$ und die Umnummerierung sei durch die Matrix $B \in \mathbb{R}^{n \times n}$ beschrieben.

$$A := \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & \dots & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \dots & \vdots \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix}$$

$$B := \begin{pmatrix} 0 & \dots & \dots & 0 & 1 \\ 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{pmatrix}$$

Die beiden Schritten zusammen sind auch eine lineare Abbildung, die durch die Matrix $M \in \mathbb{R}^{n \times n}$ beschrieben ist:

$$M = BA = \begin{pmatrix} 0 & \dots & \dots & \dots & 0 & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 & \dots & \dots & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \dots & \dots & \vdots \\ 0 & 0 & 1 & 0 & \dots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & 1 & 0 \end{pmatrix}$$

Also die nächste Winkelbeziehung v' nach einem Iterationsschritt ist $v' = Mv$. Nach Satz 3.1.6 (siehe unten) gilt, dass die Winkelbeziehung v' nach unendlich vielen Iterationsschritten

$$v' = \lim_{k \rightarrow \infty} M^k v = \begin{pmatrix} \frac{1}{n} & \dots & \frac{1}{n} \\ \vdots & \ddots & \vdots \\ \frac{1}{n} & \dots & \frac{1}{n} \end{pmatrix} v = \begin{pmatrix} \frac{2\pi}{n} \\ \vdots \\ \frac{2\pi}{n} \end{pmatrix}$$

ist, da $\sum_{i=1}^n v_i = 2\pi$.

Falls $v \neq \begin{pmatrix} \frac{2\pi}{n} \\ \vdots \\ \frac{2\pi}{n} \end{pmatrix}$, dann konvergiert v' gegen $\begin{pmatrix} \frac{2\pi}{n} \\ \vdots \\ \frac{2\pi}{n} \end{pmatrix}$, aber erreicht es nie.

Satz 3.1.6 Sei die Matrix $M \in \mathbb{R}^{n \times n}$ so definiert:

$$M := \begin{pmatrix} 0 & \dots & \dots & \dots & 0 & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 & \dots & \dots & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \dots & \dots & \vdots \\ 0 & 0 & 1 & 0 & \dots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & 1 & 0 \end{pmatrix}$$

dann gilt

$$\lim_{k \rightarrow \infty} M^k = \begin{pmatrix} \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \ddots & \vdots \\ \frac{1}{n} & \cdots & \frac{1}{n} \end{pmatrix}$$

Beweis vom Satz 3.1.6 Betrachte die Struktur der Matrix M . Es ist abzulesen, dass die zweite und dritte Zeile in die ersten und in die von vierten bis n -ten Zeilen weiterpermutiert werden, also reicht es die Einträge der zweiten und dritten Zeile zu betrachten. Da die erste und zweite Spalte in die von dritten bis n -ten Spalten weiterpermutiert werden, reicht es die Einträge der ersten und zweiten Spalte zu betrachten. Aus diesen zwei Beobachtungen folgt, dass nur die Einträge $m_{21} = m_{22} = m_{31} = m_{32}$, die sogar immer gleich sind, betrachtet werden sollen.

Betrachte o.B.d.A. den Eintrag m_{21} . Sei die k -te Iteration mit $m_{21}^{(k)}$ bezeichnet. Dann gilt $m_{21}^{(0)} = \frac{1}{2}$ und

$$m_{21}^{(k)} = \frac{1}{2}m_{21}^{(k-1)} + \frac{1}{2}m_{11}^{(k-1)} = \frac{1}{2}m_{21}^{(k-1)} + 0$$

für $0 < k \leq n - 2$ und

$$m_{21}^{(k)} = \frac{1}{2}m_{21}^{(k-1)} + \frac{1}{2}m_{11}^{(k-1)} = \frac{1}{2}m_{21}^{(k-1)} + \frac{1}{2}m_{21}^{(k-n+1)}$$

für $k \geq n - 1$, da $m_{11}^{(k)} = m_{31}^{(k-(n-2))} = m_{21}^{(k-n+2)}$.

Hier erkennen wir eine Rekursion, und ergibt sich eine Frage:

$$\lim_{k \rightarrow \infty} m_{ij}^k = \lim_{k \rightarrow \infty} m_{21}^k = ? \quad \forall i, j \in \{1, \dots, n\}$$

Die Frage führt zum Lemma 3.1.7. Der Beweis beruht auf der Vorkenntnis der linearen Algebra. [5]

Lemma 3.1.7 Sei $n \in \mathbb{N}$ eine feste Zahl. Definiere die halb-Fibonacci Folge $\{F_k\}_{k \in \mathbb{N}}$ folgendermaßen:

$$F_i = 0 \quad i \in \{0, \dots, n - 3\} \quad , \quad F_{n-2} = \frac{1}{2}$$

$$F_k = \frac{F_{k-1} + F_{k-n+1}}{2}, \quad k > n - 2$$

Dann gilt

$$\lim_{k \rightarrow \infty} F_k = \frac{1}{n}$$

Bemerkung 3.1.8 Beachte, dass $F_0 = m_{11}^{(0)}, \dots, F_{n-3} = m_{11}^{(n-3)}, F_{n-2} = m_{21}^{(0)}, \dots, F_k = m_{21}^{(k-n+2)}, \dots$

Beweis vom Lemma 3.1.7 Speichere die ersten $n-1$ Folgenmitglieder in einem

Vektor $\mathbb{R}^m \ni u_0 = \begin{pmatrix} F_{n-2} \\ \vdots \\ F_1 \\ F_0 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$. Dann kann die k -te Folgenmitglieder

F_k durch lineare Abbildung berechnet werden:

$$\begin{pmatrix} F_{k+n-2} \\ \vdots \\ F_{k+1} \\ F_k \end{pmatrix} = u_k = F^k u_0 \quad \wedge \quad F := \begin{pmatrix} \frac{1}{2} & 0 & \dots & 0 & \frac{1}{2} \\ 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & 0 & \dots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{pmatrix} \in \mathbb{R}^{m \times m}$$

Diagonalisiere die Halb-Fibonacci-Matrix F mit der Transformationmatrix S , also $F = SD_F S^{-1}$. In der Hauptdiagonale von D_F stehen die Eigenwerte, sonst sind alle Einträge Null und in den Spalten von S stehen die Eigenvektoren. Das charakteristische Polynom der Matrix F ist gegeben durch:

$$P_\lambda(F) = \left(\frac{1}{2} - \lambda\right)\lambda^{m-1}(-1)^{m-1} + \frac{1}{2}(-1)^{m-1}$$

Die Nullstellen des Polynoms sind die Eigenwerte der Matrix F .

$$(\lambda - 1)\left(\lambda^{m-1} + \frac{1}{2}\right) = 0 \quad \Leftrightarrow \quad D_F = \begin{pmatrix} \lambda_1 & 0 & \vdots & 0 \\ 0 & \lambda_2 & 0 & \dots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \lambda_m \end{pmatrix}$$

mit (o.B.d.A.) $\lambda_1 = 1$.

Im Beweis muss man zeigen, dass $\lim_{k \rightarrow \infty} F_k = \frac{1}{n}$ ist, also genügt es zu zeigen, dass

$$\lim_{k \rightarrow \infty} (SD_F S^{-1})^k = \lim_{k \rightarrow \infty} SD_F^k S^{-1} = \begin{pmatrix} \frac{2}{n} & \frac{1}{n} & \dots & \frac{1}{n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{2}{n} & \frac{1}{n} & \dots & \frac{1}{n} \end{pmatrix} \in \mathbb{R}^{m \times m} \quad (*)$$

daraus folgt nämlich

$$\lim_{k \rightarrow \infty} \begin{pmatrix} F_{k+n-2} \\ \vdots \\ F_{k+1} \\ F_k \end{pmatrix} = \lim_{k \rightarrow \infty} u_k = \lim_{k \rightarrow \infty} F^k u_0 = \lim_{k \rightarrow \infty} SD_F^k S^{-1} \begin{pmatrix} \frac{1}{2} \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{n} \\ \vdots \\ \frac{1}{n} \end{pmatrix} \in \mathbb{R}^m.$$

Zu (*) muss man Folgendes zeigen:

- a,

$$\lim_{k \rightarrow \infty} D_F^k = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{pmatrix}$$

- b,

$$S = \begin{pmatrix} 1 & * & \dots & * \\ 1 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ 1 & * & \dots & * \end{pmatrix} \quad S^{-1} = \begin{pmatrix} \frac{2}{n} & \frac{1}{n} & \dots & \frac{1}{n} \\ * & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \dots & * \end{pmatrix}$$

zu a,

$$\lim_{k \rightarrow \infty} D_F^k = \lim_{k \rightarrow \infty} \begin{pmatrix} \lambda_1^k & 0 & \dots & \dots & 0 \\ 0 & \lambda_2^k & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & \lambda_m^k \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & 0 & 0 \end{pmatrix}$$

$\lim_{k \rightarrow \infty} \lambda_1^k = 1$, da $\lambda_1 = 1$. Da $\forall i \in \{2, \dots, m\}$ gilt $\lambda_i^{m-1} = -\frac{1}{2}$, (siehe $P_\lambda(F)$), folgt

$$\lim_{k \rightarrow \infty} \lambda_i^k = \lim_{l \rightarrow \infty} (\lambda_i^{m-1})^l = \lim_{l \rightarrow \infty} \left(-\frac{1}{2}\right)^l = 0$$

zu b, Die Eigenvektoren stehen in den Spalten von S . In der ersten Spalte steht der Eigenvektor ev_1 mit $EW = 1$. Den Eigenraum $Eig(F, 1)$ zu bestimmen, muss das lineares Gleichungssystem (LSG) gelöst werden:

$$\begin{pmatrix} -\frac{1}{2} & 0 & \dots & 0 & \frac{1}{2} \\ 1 & -1 & 0 & \dots & 0 \\ 0 & 1 & -1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 1 & -1 \end{pmatrix} ev_1 = \begin{pmatrix} 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{pmatrix}$$

Aus dem LGS folgt, dass $Eig(F, 1) = c \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ mit $c \in \mathbb{R}$, deswegen sei der

Eigenvektor $ev_1 = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ und damit $S = \begin{pmatrix} 1 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ 1 & * & \dots & * \end{pmatrix}$

Da $F^t = (SD_F S^{-1})^t = (S^{-1})^t D_F^t S^t = (S^{-1})^t D_F S^t$, reicht es zu zeigen :

$$(S^{-1})^t = \begin{pmatrix} \frac{2}{n} & * & \dots & * \\ \frac{1}{n} & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & * & \dots & * \end{pmatrix}$$

Die Eigenvektoren von F^t stehen in den Spalten von $(S^{-1})^t$. In der ersten Spalte steht der EV ev_1 mit $EW = 1$. Den Eigenraum $Eig(F^t, 1)$ zu bestimmen, muss das lineares Gleichungssystem gelöst werden:

$$\begin{pmatrix} -\frac{1}{2} & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & 0 \\ 0 & \vdots & \ddots & \ddots & 1 \\ \frac{1}{2} & 0 & \dots & 0 & -1 \end{pmatrix} ev_1 = \begin{pmatrix} 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{pmatrix}$$

Aus dem LGS folgt, dass $Eig(F^t, 1) = c \begin{pmatrix} 1 \\ \frac{1}{2} \\ \vdots \\ \frac{1}{2} \end{pmatrix}$ mit $c \in \mathbb{R}$

Da $S^{-1}S = Id$, folgt, dass $ev_1^t * ev_1 = c \begin{pmatrix} 1 \\ \frac{1}{2} \\ \vdots \\ \frac{1}{2} \end{pmatrix}^t \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} = 1$, also $c = \frac{2}{n}$ und

$$ev_1^t = \left(\frac{2}{n} \quad \frac{1}{n} \quad \dots \quad \frac{1}{n} \right) \quad \wedge \quad S^{-1} = \begin{pmatrix} \frac{2}{n} & \frac{1}{n} & \dots & \frac{1}{n} \\ * & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \dots & * \end{pmatrix}$$

Damit wurde gezeigt, dass

$$\lim_{k \rightarrow \infty} SD_F^k S^{-1} = \begin{pmatrix} 1 & * & \dots & * \\ 1 & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ 1 & * & \dots & * \end{pmatrix} \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 0 & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \dots & \dots & 0 \end{pmatrix} \begin{pmatrix} \frac{2}{n} & \frac{1}{n} & \dots & \frac{1}{n} \\ * & * & \dots & * \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \dots & * \end{pmatrix} =$$

$$= \begin{pmatrix} \frac{2}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{2}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \end{pmatrix}$$

also $\lim_{k \rightarrow \infty} F_k = \frac{1}{n}$.

Damit ist Lemma 3.1.7 bewiesen.

3.2 2-dimensionaler Fall im Kreis

Betrachte das Gebiet, den Einheitskreis mit seinem Inneren:

$$\Omega = \{(p_x, p_y) \mid p_x^2 + p_y^2 \leq 1, p_x \in \mathbb{R}, p_y \in \mathbb{R}\}$$

Was passiert mit den Punkten im Voronoi Spiel? Verhalten sich die Punkte wie am Kreisrand? Gibt es stabile Konfigurationen der Punkte, die sich während des Spieles nicht ändern?

Satz 3.2.1 Falls $n = 3$ ist, wandern zuerst die Punkte auf dem Rand (in endlich vielen Iterationsschritten), danach verhalten sich die Punkte wie im eindimensionalen Fall, das heißt die Winkel zwischen den benachbarten Punkten bezüglich des Zentrums sind genau $\frac{2\pi}{3} = 120^\circ$ oder konvergieren gegen diesen Wert.

Bemerkung 3.2.2 Falls am Anfang die Punkte voneinander äquidistant am Rand liegen, ist diese Konfiguration stabil.

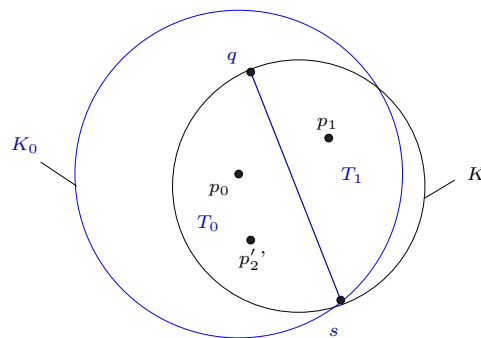


Abbildung 3.3: $n = 3$

Beweis vom Satz 3.2.1 Es reicht hier zu zeigen, dass die Punkte nach endlich vielen Iterationsschritten auf dem Rand wandern. Sei o.B.d.A. p_2 der aktuelle bewegliche Punkt und liege p_0 und p_1 beliebig im Kreis. Bezeichne q und s die Schnittpunkte des Bisektors $B(p_0, p_1)$ mit dem Kreisrand ∂K (siehe Abbildung 3.3) und gelte $|p_i q| < |p_i s|$ $i \in \{0, 1\}$. Der Bisektor $B(p_0, p_1)$ teilt den Kreis K in zwei Teile T_0, T_1 auf.

Jetzt nehmen wir an, dass sich der neue Punkt p'_2 im Inneren des Kreises befindet, ($p'_2 \notin \partial K$) und o.B.d.A liege der Punkt $p'_2 \in T_0$.

Wir kreisen um den Punkt p_0 mit dem Radius $r = |p_0s|$ herum, so entsteht der Kreis K_0 . Nach unserer Konstruktion gilt, dass $p'_2 \in K_0 \wedge p'_2 \notin \partial K_0$ ist, und somit gilt, dass

$$\min_{i=1,2} |p_i p'_2| < \min_{i=1,2} |p_i s|$$

da $|p_0 p'_2| < |p_0 s| = |p_1 s|$ ist. Es führt zum Widerspruch. \square

Satz 3.2.3 Falls $n = 4$ ist, wandern zuerst die Punkte auf dem Rand (in endlich vielen Iterationsschritten), danach verhalten sich die Punkte wie im eindimensionalen Fall, das heißt die Winkel zwischen den benachbarten Punkten bezüglich des Zentrums sind genau $\frac{2\pi}{4} = 90^\circ$ oder konvergieren gegen diesem Wert.

Bemerkung 3.2.4 Falls am Anfang die Punkte voneinander äquidistant am Rand liegen, ist diese Konfiguration stabil.

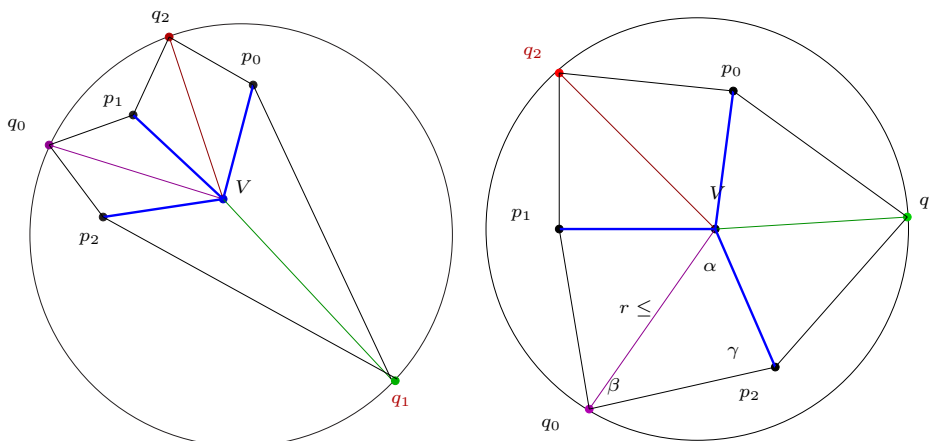


Abbildung 3.4: Konkaves und konvexes Deltoid

Beweis vom Satz 3.2.3 Wie beim Satz 3.2.1 reicht es zu zeigen, dass die Punkte nach endlich vielen Iterationsschritten auf dem Rand wandern. Sei o.B.d.A p_3 der aktuelle bewegliche Punkte, und liegen die Punkte p_0, p_1 und p_2 im Kreis beliebig und sei das Voronoi-Diagramm $V(p_0, p_1, p_2)$ vorgegeben. Dann können die folgende Punkte als Kandidat für den Punkt p_3 ins Frage kommen: Der gemeinsame Voronoi-Knoten V von p_0, p_1 und p_2 oder die Schnittpunkte (q_0, q_1, q_2) der Voronoi-Kanten mit dem Kreisrand oder irgendwelche Punkt am Kreisrand, (siehe Abbildung 3.4). Wir möchten zeigen, dass für p_3 einer Randpunkt bessere Wahl als der Voronoi-Knoten ist. Deswegen reicht es zu zeigen, dass einer von den Schnittpunkten q_0, q_1, q_2 bessere Wahl als der Voronoi-Knoten V ist. Die Punkte V, p_0, q_2, p_1 bilden ein Deltoid, ebenso die Punkte V, p_1, q_0, p_2 und V, p_2, q_1, p_0 . Man muss zwei Fälle unterscheiden:

- Es gibt ein konkaves Deltoid, das heißt ein Winkel im Deltoid ist größer als 180° , (siehe Abbildung 3.4 links).

In diesem Fall leistet das Gewünschte der Randpunkt des konkaven Deltoids: Wegen der Konkavität folgt, dass $|p_0V| < |p_0q_1|$ und $|p_2V| < |p_2q_1|$.

Es bleibt die Frage, ob $|p_1V| < |p_1q_1|$ ist. Der Punkt p_1 liegt in der Fläche, begrenzt von den durch die Punkten V, p_0 und durch die Punkte V, p_2 durchgehenden Geraden und vom Kreisrand und diese Fläche und das konkave Deltoid sind distjunkt. Der Winkel ist $\angle q_1Vp_0 > 90^\circ$, deswegen ist auch $\angle q_1Vp_1 > 90^\circ$ und damit gilt $|p_1V| < |p_1q_1|$.

- Es gibt nur konvexe Deltoide, (siehe Abbildung 3.4 rechts). Auch reicht zu zeigen, dass einer von den Schnittpunkten q_0, q_1, q_2 bessere Wahl als der Voronoi-Knoten ist.

Sei zuerst der Voronoi-Knoten V der Mittelpunkt des Kreises und wir führen einen indirekten Beweis durch. Also nehmen wir an, dass der Voronoi-Knoten bessere Wahl als alle andere Schnittpunkte ist, das heißt $|p_iq_j| < |p_i, V|$ mit $i \neq j$ (*). Bezeichne r den Radius des Kreises.

Es gilt Folgendes (o.B.d.A betrachten wir das Dreieck $\triangle(V, p_2, q_0)$): $|p_2V| \leq |q_0V| = r$. Aus diesem und (*) folgt, dass $\alpha < \beta \leq \gamma$ ist. Es gilt in einem Dreieck, dass $\alpha + \beta + \gamma = 2\pi$ ist, also der Winkel α muss kleiner als 60° sein. Nach Annahme gilt diese für alle Winkel um den Voronoi-Knoten V herum, das heißt $\sum_{i,j,i \neq j} \angle p_iVq_j < 6 \cdot 60^\circ = 360^\circ$. Es ist ein Widerspruch dazu, dass ein Vollwinkel 360° sein muss.

Der Beweis ist noch unvollständig, falls der Voronoi-Knoten V nicht der Mittelpunkt ist.

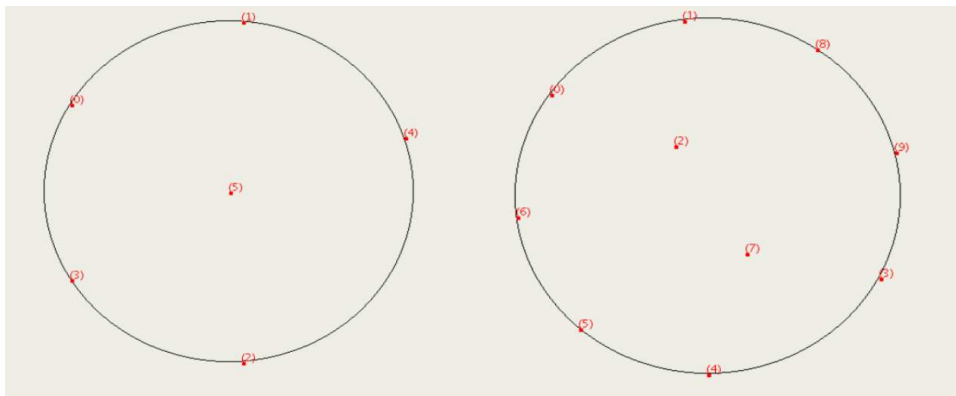


Abbildung 3.5: a, $n = 6$ b, $n = 10$

Beobachtung: Falls $n = 5$ ist, wandern die Punkte zuerst auf dem Rand (in endlich vielen Iterationsschritten), danach verhalten sich die Punkte wie im eindimensionalen Fall, das heißt die Winkel zwischen den benachbarten Punkten sind genau $\frac{2\pi}{5} = 72^\circ$ oder konvergieren gegen diesen Wert.

Beobachtung: Falls $n = 6$ ist, wandern $n - 1$ Punkte auf dem Rand und ein Punkt p zum Zentrum (in endlich vielen Iterationsschritten), danach bleibt der Punkt p im Zentrum, und die restlichen Punkte verhalten sich so, dass zwischen zwei benachbarten Punkten der Abstand größer als der Radius des Kreises wird, (siehe Abbildung 3.5 links). In diesem Fall gibt es mehrere stabile Konfigurationen, aber bei jeder liegt ein Punkt im Zentrum. (Falls ein Punkt im Zentrum und fünf Punkte am Kreisrand äquidistant liegen, ist diese Konfiguration stabil.)

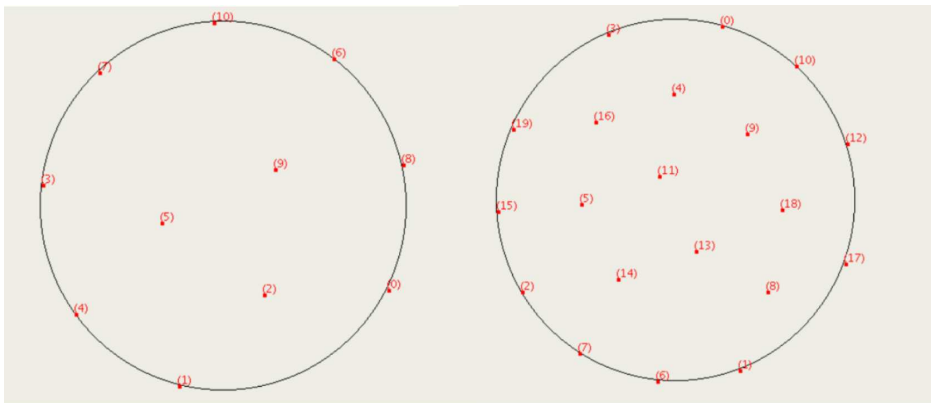


Abbildung 3.6: a, $n = 11$ b, $n = 20$

Beobachtung: Falls $n \in \{7, \dots, 9\}$ ist, wandern $n - 1$ Punkte auf dem Rand und ein Punkt p zum Zentrum (in endlich vielen Iterationsschritten), danach bleibt der Punkt p im Zentrum, und die restlichen Punkte verhalten sich wie im eindimensionalen Fall, das heißt die Winkel zwischen den benachbarten Punkten bezüglich des Zentrums sind genau $\frac{2\pi}{n-1}$ oder konvergieren gegen diesen Wert.

Beobachtung: Falls $n = 10$ ist, liegen zwei Punkte weit vom Zentrum entfernt im Inneren des Kreises und die restlichen Punkte am Rand im stabilen Zustand, (siehe Abbildung 3.5 rechts).

Beobachtung: Falls $n > 10$ ist, liegen mehr als zwei Punkte im Inneren des Kreises und die restlichen Punkte am Rand, (siehe Abbildung 3.6).

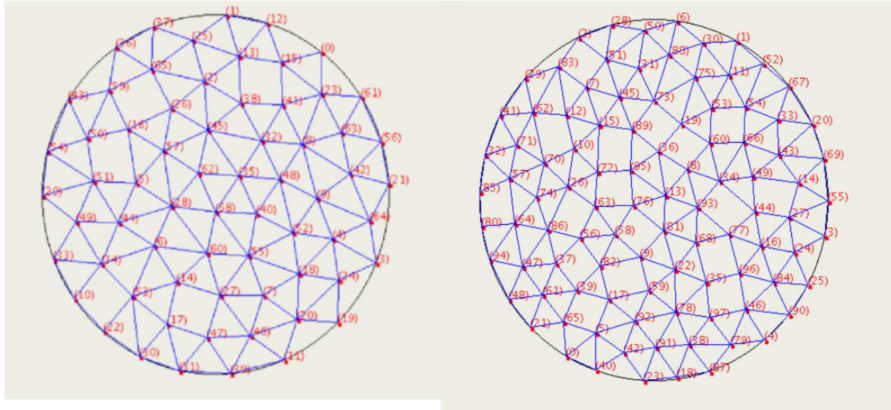


Abbildung 3.7: hexagonal ähnlich Netze

Beobachtung: Wenn die Anzahl der Punkte groß ist, liegen die Punkte in einem hexagonal ähnlichen Netz. Im stabilen Zustand ist der durchschnittliche Ecken-grad der Delaunay-Zerlegung (selbst ein Punkt im Spiel) zwischen 5 und 6, (siehe Abbildung 3.7). Insgesamt scheinen die Punkte bei einer stabilen Konfiguration regelmäßig im Kreis verteilt zu sein.

Es gibt einen bekannten Bereich in der Geometrie: **Kugelpackungen**, deren Aussagen unseren numerischen Ergebnissen ähneln. Deswegen werden wir einen kleinen Teil dieser Theorie kennenlernen, und mit unseren Beobachtungen vergleichen. Die folgenden Definitionen, Feststellungen und Vermutungen dieser schönen Theorie stammen aus dem Buch: Max Leppmeier: Kugelpackungen von Kepler bis heute.

unendliche Packungen und ihre Packungsdichte:

Definition 3.2.5 [4] Seien a_1, a_2, \dots, a_m Vektoren in \mathbb{R}^m . Für jedes $i = 1, \dots, m$ gelte:

$$a_j \neq \sum_{i=1, i \neq j}^m \lambda_i a_i \quad (\lambda_i \in \mathbb{R})$$

Dann heißt

$$G = \left\{ \sum_{i=1}^m m_i a_i \mid m_i \in \mathbb{Z} \right\}$$

ein m -dimensionales Gitter und $\{a_1, a_2, \dots, a_m\}$ eine Basis.

Definition 3.2.6 [4] Sei G ein m -dimensionales Gitter und $\{a_1, a_2, \dots, a_m\}$ eine Basis. Dann heißt

$$\mathcal{F}(G) = \left\{ \sum_{i=1}^m \lambda_i a_i \mid \lambda_i \in [0; 1] \right\}$$

Fundamentalparallelotop (in der Ebene auch Fundamentalparallelogramm).

Definition 3.2.7 [4] Gegeben seien eine m -dimensionale offene Einheitskugel B^m und ein m -dimensionales Gitter G . Für alle Gittervektoren $g \in G$ sei $B_g^m = B^m + g$ die um g verschobene Kopie von B^m . Falls je zwei verschiedene B_g^m und B_h^m ($g, h \in G$) disjunkt sind, das heißt $B_g^m \cap B_h^m = \emptyset$ für $g \neq h$, heißt das Tupel $GP(B^m, G)$ eine (m -dimensionale) **unendliche Gitterpackung**.

Definition 3.2.8 [4] Sei $GP(B^m, G)$ eine m -dimensionale unendliche Kugeligitterpackung und sei $\mathcal{F}(G)$ das G zugrundeliegende Fundamentalparallelotop. Dann heißt

$$\delta(B^m, G) = \frac{\text{Vol}(B^m)}{\text{Vol}(\mathcal{F}(G))} = \frac{\text{genutztes Volumen}}{\text{benutztes Volumen}}$$

die (**unendliche**) **Gitterpackungsdichte** der Kugeligitterpackung $GP(B^m, G)$.

Wir sehen sofort, dass die Packungsdichte Werte zwischen 0 und 1 annimmt.

Satz 3.2.9 (dichteste Kreisgitterpackung, Lagrange 1773): [4] Unter allen Kreisgitterpackungen besitzt allein diejenige mit hexagonalem Gitter die maximale Packungsdichte

$$\delta_{max} = \frac{\pi}{2\sqrt{3}}$$

Ein hexagonales Gitter sieht folgendermaßen aus: Die Voronoi-Regionen des Voronoi-Diagramms der Gitterpunkte sind regelmäßige Sechsecke und die Delaunay-Zerlegung besteht aus regelmäßigen Dreiecken.

Der Satz 3.2.9 enthält damit zwei Aussagen: Zum einen besagt er, dass die Packungsdichte ein Maximum besitzt. Zum anderen sagt der Satz, dass dieses Maximum von nur einer Klasse von zweidimensionalen Gittern angenommen wird, nämlich dem hexagonalen Gitter.

endliche Packungen und ihre Packungsdichte:

Definition 3.2.10 [4] Sei B^m eine offene Einheitskugel im \mathbb{R}^m . Sei $C_n = \{c_1, c_2, \dots, c_n\}$ eine Menge von n Ortsvektoren im \mathbb{R}^m . Für jeden Ortsvektor $c_i \in C_n$ ($1 \leq i \leq n$) sei $B_i = B^m + c_i$ die um c_i verschobene Kopie von B^m . Falls je zwei verschiedene B_i und B_j ($1 \leq i < j \leq n$) disjunkt sind, heißt die Menge

$$P(B^m, C_n) = \{B_i \mid 1 \leq i \leq n\}$$

eine **endliche Kugelpackung**.

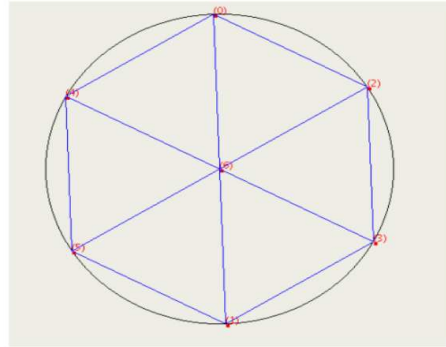
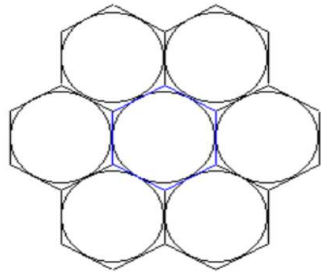


Abbildung 3.8: a, Dichteste Kreispackung in der Ebene [20], b, Das Resultat des Spieles mit 6 Punkten

Definition 3.2.11 [4] Sei $P(B^m, C_n)$ eine endliche Kugelpackung. Dann heißt

$$d(B^m, C_n) = \frac{n \cdot \text{Vol}(B^m)}{\text{Vol}(\text{conv}(B^m + C_n))}$$

die zugehörige **endliche Kugelpackungsdichte**

Dabei ist die konvexe Hülle einer Menge M , $\text{conv}(M)$, die kleinste konvexe Menge, die M als Teilmenge enthält. Es ist leicht zu sehen, dass für die Packungsdichte einer endlichen Packung stets gilt:

$$0 < d(B^m, C_n) \leq 1.$$

Definition 3.2.12 [4] Sei $P(B^m, C_n)$ eine endliche Kugelpackung. Falls

$$\dim(\text{conv}(C_n)) > 1,$$

dann heißt $P(B^m, C_n)$ eine **Clusterpackung**.

Bemerkung 3.2.13 [4] Es gibt endliche hexagonale Kreispackungen $P(B^2, C_n^{\text{hex}})$, deren Packungsdichte im Limes $n \rightarrow \infty$ gegen die Packungsdichte der hexagonalen Kreisgitterpackung $GP(B^2, G^{\text{hex}})$ konvergiert. Für die hexagonale und allgemein für beliebige Clusterpackungen gilt:

$$d(B^2, C_n) < \frac{\pi}{2\sqrt{3}}.$$

Bemerkung 3.2.14 [4] Bei einer endlichen quadratischen Kreispackungen ist die Packungsdichte ungefähr 0,6768 und bei einer endlichen hexagonalen Kreispackungen beträgt sie ungefähr 0,7075.

Ein weitaus kompliziertes Problem ist es, bei gegebener Anzahl der Kreise unter allen möglichen Clusterpackungen diejenige Packung mit maximaler Packungsdichte herauszufinden. Was schon bekannt geworden ist, stellen wir in zwei Beispielen dar:

Unter allen hexagonalen Kreispackungen von 7 Kreisen $P(B^2, C_7^h)$ besitzt nur die sechseckige Konfiguration $P(B^2, C_7^{hex})$ maximale Packungsdichte, (siehe Abbildung 3.8 links).

Unter allen Kreispackungen von 3 Kreisen $P(B^2, C_3)$ besitzt die hexagonale Konfiguration $P(B^2, C_3^{hex})$ maximale Packungsdichte.

Vermutung 3.2.15 [4] *Für beliebiges n ist die Lösung noch unbekannt. Aber man hat eine intuitive Ahnung davon, wie diese ideale Kreispackung aussehen soll: Sie ist hexagonal und die Form der konvexen Hülle ist ebenfalls so regulär wie möglich.*

Es wurde bis jetzt festgestellt, dass das Voronoi Spiel mit 3 und 7 Punkten hexagonale Kreispackungen liefert, (siehe Abbildung 3.8 rechts), wobei der Radius dem minimalen Abstand gleicht, aber mit beliebig vielen Punkten ist es noch fragwürdig.

Der minimale Abstand wurde bei stabilen Konfigurationen der Punkte gemessen. Die Kreispackungsdichte, mit der Hälfte des minimalen Abstandes als Radius berechnet, ergibt einen durchschnittlichen Wert von 0,7, (siehe Kapitel 5). Vergleichen wir diese Aussagen und die letztere Vermutung 3.2.15 mit unserem experimentellen Resultat, führt es uns zur folgenden Frage:

Approximiert im Allgemeinen das Voronoi Spiel eine endliche, eventuell hexagonale Kreispackung mit maximaler Packungsdichte? Mit anderen Worten konvergiert die Packungsdichte einer stabilen Konfiguration gegen eine endliche Kreispackung mit maximaler Packungsdichte? Leider kann diese Frage noch nicht beantwortet werden.

Kapitel 4

Ergebnisse auf der Kugeloberfläche

4.1 Sphärische Geometrie

Obwohl wir beim Aufbau dieser sphärischen Geometrie auf uns wohlbekanntere Geometrie der Ebene und des Raumes (die Euklidische Geometrie) zurückgreifen können, wird sich beim tieferem Eindringen herausstellen, dass die sphärische Geometrie als völlig unabhängige, eigenständige Theorie aufgefasst werden kann, die mit der ebenen Geometrie wichtige Gemeinsamkeiten und Ähnlichkeiten aber auch eine Reihe von Unterschieden aufweist. Zuerst werden wir in diesem Abschnitt Grundformeln vom beliebigen sphärischen Dreieck, das heißt Beziehungen zwischen deren Seiten und Winkel vorstellen. Der zweite Teil dieses Abschnittes beschäftigt sich mit bestimmten Polyedern, deren Seitenflächen Regelmäßigkeiten aufweisen. Dieses Thema kann auch zur sphärischen Geometrie gezählt werden, da die Eckpunkte eines Polyeders auf einer Kugeloberfläche liegen.

Sphärische Trigonometrie

Definition 4.1.1 [1] *Alle Kreise der Kugel, deren Mittelpunkt mit dem der Kugel identisch sind, heißen **Großkreise**.*

Definition 4.1.2 [16] *Unter einem **sphärischen Dreieck** verstehen wir ein von drei Großkreisen berandeten Kugeldreieck. Sphärische Dreiecke werden auch als **Eulerische Dreiecke** bezeichnet.*

Ein sphärisches Dreieck wird wie ein Dreieck in der ebenen Geometrie bezeichnet. Trotzdem gibt es aber einen wichtigen Unterschied zur ebenen Geometrie. Eine

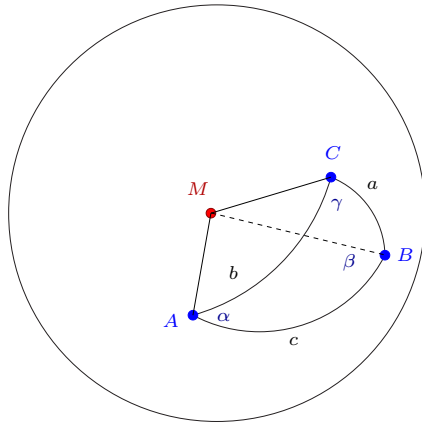


Abbildung 4.1: Das sphärische Dreieck

sphärische Dreieckseite kann auch als Winkel betrachtet werden, (siehe Abbildung 4.1). Zum Beispiel kann die Seite a als Winkel $\angle BMC$ aufgefasst werden, wobei M der Mittelpunkt der Kugel ist. Auf der Einheitskugel ist das Bogenmaß dieses Winkels die Länge der Seite a . Ein weiterer Unterschied besteht darin, dass die drei Winkel eines sphärischen Dreieckes nicht mehr voneinander abhängig sind.

Die folgenden Grundformeln gelten für die Einheitskugel.[17][1]

Satz 4.1.3 Seiten-Cosinus-Satz *In einem beliebigen Eulerischen Dreieck mit den Seiten a , b , und c sowie jeweils gegenüber liegenden Innenwinkeln α , β und γ gelten die Beziehungen.*

$$\cos(c) = \cos(a) \cos(b) + \sin(a) \sin(b) \cos(\gamma)$$

$$\cos(a) = \cos(b) \cos(c) + \sin(b) \sin(c) \cos(\alpha)$$

$$\cos(b) = \cos(c) \cos(a) + \sin(c) \sin(a) \cos(\beta)$$

Satz 4.1.4 Winkel-Cosinus-Satz *In einem beliebigen Eulerischen Dreieck mit den Seiten a , b und c sowie jeweils gegenüber liegenden Innenwinkeln α , β und γ gelten die Beziehungen.*

$$\cos(\gamma) = -\cos(\alpha) \cos(\beta) + \sin(\alpha) \sin(\beta) \cos(c)$$

$$\cos(\alpha) = -\cos(\beta) \cos(\gamma) + \sin(\beta) \sin(\gamma) \cos(a)$$

$$\cos(\beta) = -\cos(\gamma) \cos(\alpha) + \sin(\gamma) \sin(\alpha) \cos(b)$$

Aus dem Winkel-Cosinus-Satz folgt der Sinus-Satz.

Satz 4.1.5 Sinus-Satz *In einem beliebigen Eulerischen Dreieck mit den Seiten a , b , und c sowie jeweils gegenüber liegenden Innenwinkeln α , β und γ gelten die Beziehungen.*

$$\frac{\sin(a)}{\sin(\alpha)} = \frac{\sin(b)}{\sin(\beta)} = \frac{\sin(c)}{\sin(\gamma)}$$

In der sphärischen Trigonometrie gelten weitere Formeln, welche keine Entsprechung in der ebenen Trigonometrie haben. Sie lassen sich aus den vorherigen Formeln auf rechnerischem Wege herleiten.

Satz 4.1.6 Cotangens-Satz *In einem beliebigen Eulerischen Dreieck mit den Seiten a , b , und c sowie jeweils gegenüber liegenden Innenwinkeln α , β und γ gelten die Beziehungen.*

$$\cot(b) \sin(c) = \cos(c) \cos(\alpha) + \sin(\alpha) \cot(\beta)$$

$$\cot(c) \sin(a) = \cos(a) \cos(\beta) + \sin(\beta) \cot(\gamma)$$

$$\cot(a) \sin(b) = \cos(b) \cos(\gamma) + \sin(\gamma) \cot(\alpha)$$

$$\cot(b) \sin(a) = \cos(a) \cos(\gamma) + \sin(\gamma) \cot(\beta)$$

$$\cot(c) \sin(b) = \cos(b) \cos(\alpha) + \sin(\alpha) \cot(\gamma)$$

$$\cot(a) \sin(c) = \cos(c) \cos(\beta) + \sin(\beta) \cot(\alpha)$$

Reguläre und halbrekuläre Polyeder

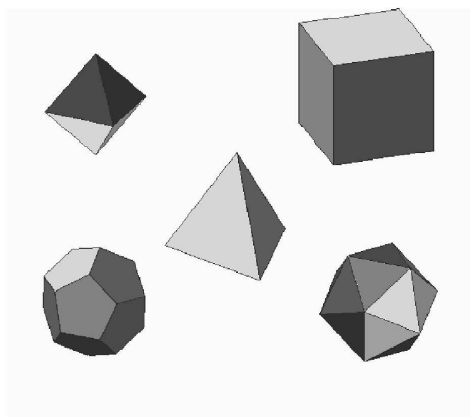


Abbildung 4.2: Die fünf Platonischen Polyeder [14]

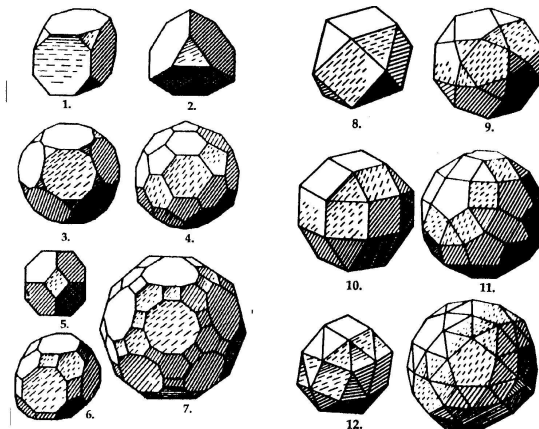


Abbildung 4.3: Die dreizehn Archimedischen Polyeder [9]

Definition 4.1.7 [8] *Im folgenden soll unter einem **Polyeder** ein zusammenhängendes Gebiet im Raum verstanden werden, dessen Oberfläche sich aus ebenen Vielecken (Polygonen) zusammensetzt ist. Polyeder besitzen Ecken, Kanten und Seitenflächen (Facetten).*

Während jede Kante genau zwei Ecken verbindet und genauso zwei Seiten voneinander trennt, können im Ecken beliebig viele Kanten zusammenlaufen und Seiten durch beliebig viele Kanten begrenzt sein.

Definition 4.1.8 [8] *Ein Polyeder heißt **konvex**, wenn er Durchschnitt endlich vieler Halbräume ist.*

Definition 4.1.9 [8] *Ein konvexes Polyeder heißt **regulär (regelmäßig)**, wenn einerseits seine Seitenfläche kongruente reguläre Polygone sind, und andererseits an jeder Ecke gleich viele derartige Vielecke aneinanderstossen. Sie werden als **Platonische Körper** bezeichnet, (siehe Abbildung 4.2).*

Um die fünf Platonische Körper zu finden, gehen wir systematisch vor. Für reguläre Polyeder sind Ecken und Seitengrad Invarianten. Wir sortieren nach ihnen.[8]

- **Seitengrad 3, von Dreiecken begrenzte Polyeder.** An einer Ecke müssen mindestens drei Seiten zusammenstoßen. Ebenso könnten es vier oder fünf sein. Sechs Dreiecke würden in der Ebene liegen, weil $6 * 60^\circ = 360^\circ$ einen Vollwinkel bilden. Es gibt zu gegebenen Ecken und Seitengrad höchstens ein Polyeder.

Seiten und Eckengrad drei führt auf das reguläre Tetraeder. Mit Eckengrad vier entsteht das Oktaeder und mit Eckengrad fünf entsteht das Ikosaeder.

	EG	Schema	Ecken	Kanten	Flächen	Name
1	3	3,3,3	4	6	4	Tetraeder
2	3	4,4,4	8	12	6	Hexaeder
3	3	5,5,5	20	30	12	Dodekaeder
4	4	3,3,3,3	6	12	8	Oktaeder
5	5	3,3,3,3,3	12	30	20	Ikosaeder

Tabelle 4.1: Die fünf Platonischen Polyeder

- **Seitengrad 4, von Quadraten begrenzte Polyeder.** Da vier Quadrate in einer Ebene liegen, muss der Eckengrad drei betragen und das Polyeder somit ein Würfel (Hexaeder) sein.
- **Seitengrad 5, von Fünfecken begrenzte Polyeder.** Der Zentralwinkel eines Fünfecks beträgt 72° , also einer seiner Innenwinkel 108° . Da $4 \cdot 108^\circ > 360^\circ$ gilt, muss der Eckengrad also drei betragen und das Polyeder somit ein Dodekaeder sein.

Die folgende Tabelle 4.1 zeigt die fünf Platonischen Körper [16],[8]. Unter Schema versteht man, welche regelmäßigen Vielecke in einer Ecke zusammenstoßen. Aus regulären Polygonen lassen sich weitere Polyeder zusammensetzen, wir erwähnen zwei Klassen.

Definition 4.1.10 [8] Eine Klasse sind die **Prismen**, die entstehen, indem man zwei n -Ecke durch n Quadrate verbindet. (Ein Spezialfall ist der Würfel mit $n = 4$.)

Definition 4.1.11 [8] Die andere Klasse sind die **Antiprismen**, bei der die beiden Seiten n -Ecken versetzt angeordnet sind und durch zwei Reihen von regulären Dreiecken verbunden werden. (Ein Spezialfall ist der Oktaeder mit $n = 3$.)

Definition 4.1.12 [8] Ein konvexes Polyeder heißt **halbregulär**, falls seine Seitenflächen reguläre Polygone sind und es je zwei Ecken eine Drehung des Polyeders auf sich gibt, die die Ecken ineinander überführt.

Bemerkung 4.1.13 [8] Damit sind reguläre Polyeder, Prismen und Antiprismen halbregulär. Alle weiteren halbregulären Polyeder werden als **Archimedische Körper** bezeichnet, (siehe Abbildung 4.3).

Es gibt genau dreizehn Archimedische Körper, die in der Tabelle 4.2. zu sehen sind [8],[9]. Um sie zu finden, kann man wieder, wie vorher bei den Platonischen

	EG	Schema	Ecken	Kanten	Flächen	Name
1	3	3,8,8	24	36	14	abgestumpftes Hexaeder
2	3	3,6,6	12	18	8	abgestumpftes Tetraeder
3	3	3,10,10	60	90	32	abgestumpftes Dodekader
4	3	5,6,6	60	90	32	abgestumpftes Ikosaeder
5	3	4,6,6	24	36	14	abgestumpftes Oktaeder
6	3	4,6,8	48	72	26	großes Rhombenkuboktaeder
7	3	4,6,10	120	180	62	großes Rhombenikosidodekaeder
8	4	3,4,3,4	12	24	14	Kuboktaeder
9	4	3,5,3,5	30	60	32	Ikosidodekaeder
10	4	3,4,4,4	24	48	26	kleines Rhombenkuboktaeder
11	4	3,4,5,4	60	120	62	kleines Rhombenikosidodekaeder
12	5	3,3,3,3,4	24	60	38	Cubus simus
13	5	3,3,3,3,5	60	150	92	Dodekaedron simum

Tabelle 4.2: Die dreizehn Archimedischen Polyeder [8]

Körpern, vorgehen und diese nach ihrem Eckenrad sortieren. So entsteht aber nicht alle halbregulären Polyeder. Ein anderer Zugang besteht darin, durch Abschneiden von Ecken neuen Körper zu konstruieren. Auf dieser Weise können elf der dreizehn Archimedischen Körper konstruiert werden. Ausnahme sind die beiden Polyeder mit Eckenrad fünf.

Bemerkung 4.1.14 [8] *Wie die Platonischen und Archimedischen Polyeder besitzen die Prismen und Antiprismen auch eine Umkugel und zu je zwei Ecken gibt es eine Drehung des Polyeders, die die Ecken ineinander überführt.*

Eine schöne große Sammlung von verschiedenen interessanten Polyedern ist bei der freien Online-Enzyklopedie (Wikipedia) zu finden [15].

4.2 Stabile und instabile Polyeder

In diesem Abschnitt werden sowohl die theoretischen als auch die numerischen Ergebnisse des Voronoi Spieles dargestellt.

Betrachte das Gebiet, die Oberfläche der Einheitskugel:

$$\Omega = \{(p_x, p_y, p_z)^t \mid p_x^2 + p_y^2 + p_z^2 = 1, p_x \in \mathbb{R}, p_y \in \mathbb{R}, p_z \in \mathbb{R}\}$$

Was passiert mit den Punkten im Voronoi Spiel ? Kann man explizit stabile Polyeder angeben? Werden diese stabile Konfigurationen beim beliebigen Start des Spieles erreicht? Wenn für ein n die Antwort "Ja" ist, gibt es aber andere stabile Konfigurationen?

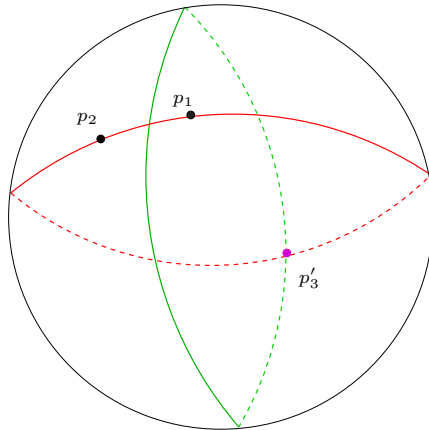


Abbildung 4.4: Die Punkte liegen auf einem Großkreis

Satz 4.2.1 *Mit $n = 3$ liegen die Punkte im stabilen Zustand äquidistant auf einem Kreis (am Kreisrand) mit Radius $r = 1$, das heißt auf einem Großkreis. Bei allen Anfangskonfigurationen der Punkte konvergiert das Verfahren gegen ein regelmäßigen Dreieck, deren Punkte auf einem Kreis mit Radius $r = 1$, also auf einem Großkreis der Kugel liegen.*

Beweis vom Satz 4.2.1 Seien die Punkte p_1, p_2, p_3 beliebig auf der Kugeloberfläche, und o.B.d.A sei p_3 der aktuelle bewegliche Punkt. Betrachte den Großkreis, der durch die Punkte p_1 und p_2 durchgeht. Der neue p'_3 Punkt liegt auf dem Bisektor von p_1 und p_2 . Der weiteste Punkt auf dem Bisektor liegt auf dem Großkreis von p_1 und p_2 , also liegen die Punkte p_1, p_2 und p'_3 auf demselben Großkreis, (siehe Abbildung 4.4), damit kann man den Beweis auf den eindimensionalen Fall zurückführen, (siehe Abbildung 4.5). \square

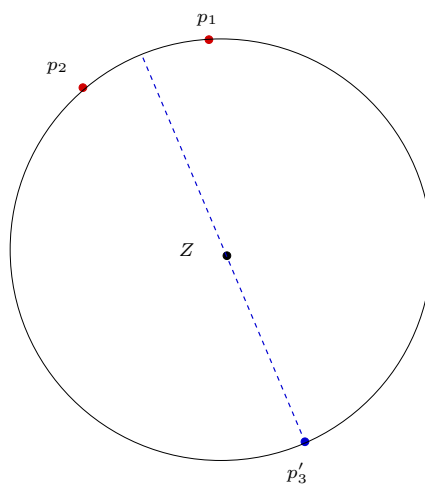


Abbildung 4.5: Zurückführung auf den eindimensionalen Fall

Beobachtung: Mit $n = 4$ sind das reguläre Tetraeder (Antiprisma) und das regelmäßige Quadrat (Prisma) (mit Seitenlänge $\sqrt{2}$ in der Einheitskugel) stabil. Verschiebt man die Punkte von Quadrat in einer winzigen Umgebung, dann konvergiert das Polyeder gegen ein regelmäßiges Quadrat, sonst in allen anderen Fällen gegen ein reguläres Tetraeder.

Überlegung von der Beobachtung 4.2 (Konvergenz gegen das Tetraeder) Mit dem Ansatz: seien die Eckpunkte von Tetraeder durchnummeriert p_1, p_2, p_3, p_4 , (siehe Abbildung 4.6). Bezeichne α_{123} den Winkel $\angle p_2 p_1 p_3$ auf der Kugeloberfläche, (also die erste Nummer zeigt immer, wo die Spitze des Winkels ist.)

Die folgende Beziehungen gelten:

$$\sum \alpha_{1ij} = 2\pi \quad , \quad \sum \alpha_{2ij} = 2\pi \quad , \quad \sum \alpha_{3ij} = 2\pi \quad , \quad \sum \alpha_{4ij} = 2\pi \quad (*)$$

Seien die Winkel im Tetraeder auf der Kugeloberfläche :

$$\alpha_1 = \alpha_{234} \quad \alpha_2 = \alpha_{324} \quad \alpha_3 = \alpha_{423} \quad \alpha_4 = \alpha_{314} \quad \alpha_5 = \alpha_{134} \quad \alpha_6 = \alpha_{413}$$

$$\alpha_7 = \alpha_{412} \quad \alpha_8 = \alpha_{124} \quad \alpha_9 = \alpha_{214} \quad \alpha_{10} = \alpha_{213} \quad \alpha_{11} = \alpha_{123} \quad \alpha_{12} = \alpha_{312}$$

Nun o.B.d.A seien die Punkte p_2, p_3, p_4 fest und wir möchten die neue Position für den Punkt p_1 bestimmen. Die Abbildung $F : \mathbb{R}^4 \mapsto \mathbb{R}^4$ beschreibe die Veränderung der Winkel nach einem Iterationsschritt. α_1, α_2 und α_3 bleiben unverändert. Aus (*) folgt: $\alpha_4, \alpha_6 \mapsto x, \alpha_{10}, \alpha_{12} \mapsto y, \alpha_7, \alpha_9 \mapsto z$, wobei

$$x = \pi + \frac{1}{2}(\alpha_1 - \alpha_2 - \alpha_3), \quad y = \pi + \frac{1}{2}(\alpha_2 - \alpha_1 - \alpha_3), \quad z = \pi + \frac{1}{2}(\alpha_3 - \alpha_1 - \alpha_2)$$

Um die neuen Winkel α'_5, α'_8 und $\alpha_{11'}$ zu bekommen, brauchen wir einige Formeln aus der sphärischen Trigonometrie. Es wird nur der Rechenweg für den Winkel α'_5 gezeigt. Aus dem Winkel-Cosinus-Satz für das sphärische Dreieck $\triangle p_1 p_3 p_4$ folgt:

$$\cos(\alpha'_5) = -\cos(x) \cos(x) + \sin(x) \sin(x) \cos(a_{34})$$

Es gibt aber noch eine Unbekannte, die Seite a_{34} . Diese Seite zu bestimmen, brauchen wir einen Cotangens-Satz für das sphärische Dreieck $\triangle p_2 p_3 p_4$.

$$\cot(a_{34}) \sin(a_{24}) = \cos(a_{24}) \cos(\alpha_3) + \sin(\alpha_3) \cot(\alpha_1)$$

Mit dem Sinus Satz ist $\sin(a_{24}) = \frac{\sin(a_{34})}{\sin(\alpha_1)} \sin(\alpha_2)$ und damit kann man den Cotangens-Satz so schreiben:

$$\frac{\cos(a_{34})}{\sin(\alpha_1)} \sin(\alpha_2) = \sqrt{\left(1 - \left(\frac{\sin(a_{34})}{\sin(\alpha_1)} \sin(\alpha_2)\right)^2\right) \cos(\alpha_3) + \sin(\alpha_3) \cot(\alpha_1)}$$

Nach Umrechnung bekommen wir mit $x = \cos(a_{34})$ ein zweigradiges Polynom: $ax^2 + bx + c = 0$, wobei (man kann merken, die Koeffizienten sind symmetrisch)

$$a = \sin^2(\alpha_2) \sin^2(\alpha_3), \quad b = -2 \sin(\alpha_2) \sin(\alpha_3) \cos(\alpha_1), \quad c = \cos^2(\alpha_1) - \cos^2(\alpha_2) \cos^2(\alpha_3)$$

Aus diesem zweigradigen Polynom kann $\cos(a_{34})$ berechnet, und in den Winkel-Cosinus-Satz eingesetzt werden. Also endlich bekommen wir den Wert für den Winkel α'_5 . Analogerweise kann man die Winkel α'_8 und α'_{11} berechnen.

Also insgesamt sieht die Funktion F mit Permutation, (um mehrere Iterationsschritte nacheinander ausgeführt werden zu können) so aus:

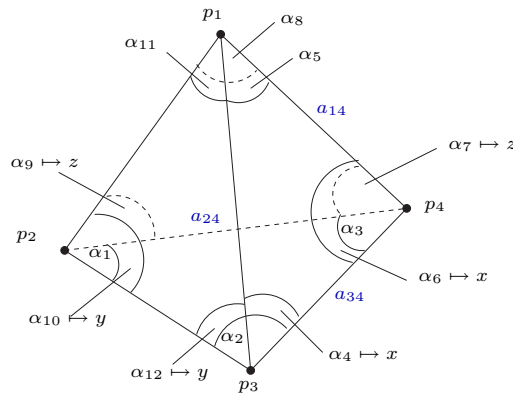


Abbildung 4.6: Das Polyeder mit vier Ecken in allgemeiner Lage

$$F \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ \alpha_9 \\ \alpha_{10} \\ \alpha_{11} \\ \alpha_{12} \end{pmatrix} = \begin{pmatrix} x \\ \alpha'_5 \\ x \\ z \\ \alpha'_8 \\ z \\ y \\ \alpha'_{11} \\ y \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix}$$

Es bleibt zu zeigen, dass

$$\lim_{k \rightarrow \infty} F^{(k)} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ \alpha_9 \\ \alpha_{10} \\ \alpha_{11} \\ \alpha_{12} \end{pmatrix} = \begin{pmatrix} \pi \\ \pi \\ \pi \\ \pi \\ \pi \\ \pi \\ \pi \\ \pi \\ \pi \\ \pi \\ \pi \\ \pi \end{pmatrix}$$

wobei $F^{(k)}$ bedeutet, dass k -mal nacheinander Iterationsschritte ausgeführt werden.

Dieses Konvergenzverhalten wurde mit Java Programm getestet, und man bekommt dieses Ergebnis, (Es wurde kein Gegenbeispiel gefunden).

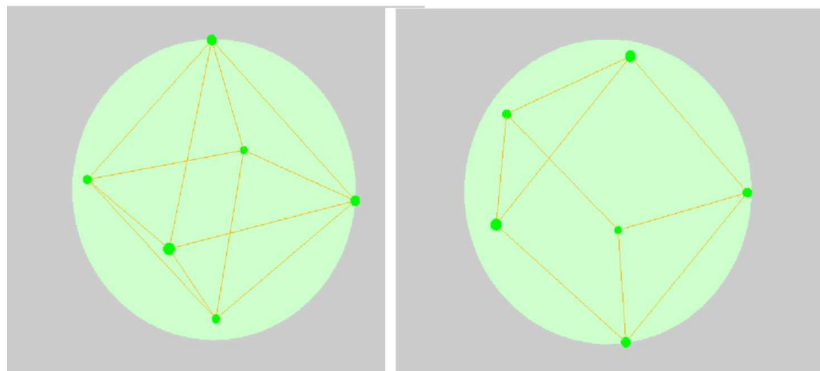


Abbildung 4.7: a, das Oktaeder (Antiprisma) b, das Prisma

Beobachtung: Mit $n = 5$ ist die Pyramide stabil. Ihre Seitenpolygone sind vier regelmäßigen Dreiecke und ein regelmäßiges Quadrat. Alle Kantenlängen der Pyramide sind $\sqrt{2}$ in der Einheitskugel. Bei einer beliebigen Anfangskonfiguration der Punkte konvergiert das Polyeder oft gegen eine Pyramide. Der minimale Abstand der Pyramide ist $\sqrt{2}$ und bei allen anderen numerisch stabilen Körpern ist der minimale Abstand kleiner als $\sqrt{2}$.

Beobachtung: Mit $n = 6$ sind das Oktaeder (Antiprisma) und das Prisma stabil, (siehe Abbildung 4.7). Bei einer beliebigen Anfangskonfiguration der Punkte

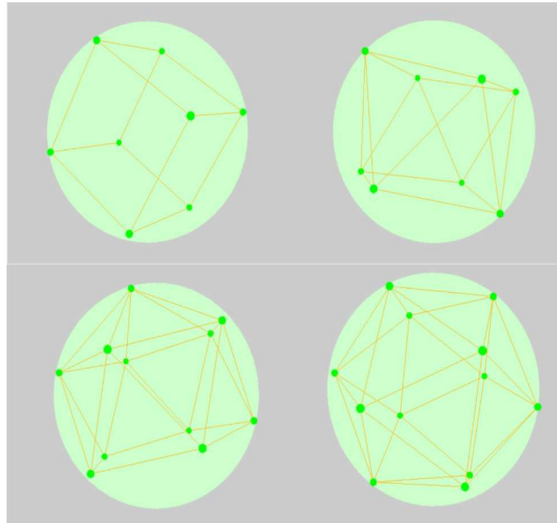


Abbildung 4.8: a, der Würfel (Prisma) b, das Antiprisma c, die Krone d, das Ikosaeder

konvergiert das Polyeder oft gegen ein Oktaeder oder teilweise auch gegen ein Prisma. Der minimale Abstand des Prismas (und damit alle Kantenlängen) ist 1,3093 und der des Oktaeders ist $\sqrt{2} = 1,4142$. Bei allen anderen numerisch stabilen Körpern ist der minimale Abstand kleiner als $\sqrt{2}$, folglich hat das Oktaeder den größten minimalen Abstand.

Beobachtung: Mit $n = 8$ sind der Würfel (Prisma) und das Antiprisma stabil, (siehe Abbildung 4.8 oben). Verschiebt man die Punkte des Würfels in einer kleinen Umgebung, dann konvergiert das Polyeder gegen einen Würfel, sonst scheinen sie in allen anderen Fällen gegen ein Antiprisma zu konvergieren. Der minimale Abstand des Würfels ist 1,1547 und der des Antiprismas ist 1,2155. Bei allen anderen numerisch stabilen Polyedern ist der minimale Abstand kleiner als 1,2155, folglich hat das Antiprisma den größten minimalen Abstand.

Beobachtung: Mit $n = 11$ ist die Krone (eine Ecke des Ikosaeders fehlt) stabil, (siehe Abbildung 4.8 unten links). Bei einer beliebigen Anfangskonfiguration der Punkte konvergiert das Polyeder gegen eine **Krone**. Der minimale Abstand der Krone ist 1,05146 und bei allen anderen numerisch stabilen Polyedern ist der minimale Abstand kleiner als dieser Wert.

Beobachtung: Mit $n = 12$ sind das Ikosaeder (siehe Abbildung 4.8 unten rechts) und das Kuboktaeder (siehe Abbildung 4.9 oben links) stabil, das abgestumpfte Tetraeder (siehe Abbildung 4.9 oben rechts) ist instabil. Das abgestumpfte Te-

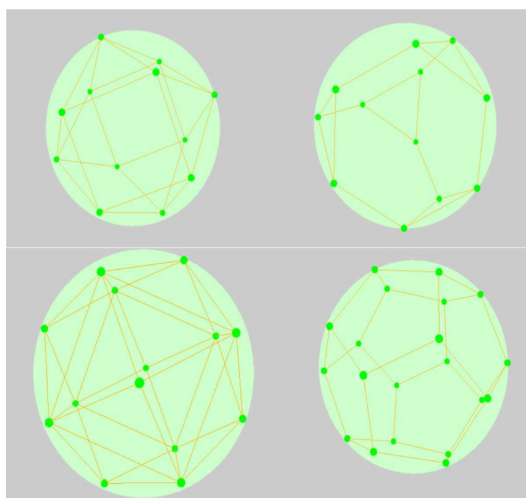


Abbildung 4.9: a, das Kuboktaeder b, das abgestumpfte Tetraeder c, der Pyramidenwürfel d, das Dodekaeder

traeder und ein beliebiges Polyeder scheinen gegen ein Ikosaeder zu konvergieren, es sei denn, das Polyeder unterscheidet sich kaum vom Kuboktaeder. Der minimale Abstand des Ikosaeders ist 1,0514, der des Kuboktaeder ist 1,0, der des abgestumpften Tetraeders vor dem Spiel ist 0,8528 und nach dem Spiel (es ist natürlich ein anderes Polyeder) im stabilen Zustand ist 1,0425. Bei allen anderen numerisch stabilen Körpern ist der minimale Abstand kleiner als des Ikosaeders.

Beobachtung: Mit $n = 14$ ist der Pyramidenwürfel (Tetrakisheptaeder) stabil, (siehe Abbildung 4.9 unten links). Der minimale Abstand des Pyramidenwürfels ist 0,9194 und bei allen anderen numerisch stabilen Polyedern ist er kleiner als der des Pyramidenwürfels. Es bleibt die Frage: Konvergieren die Polyeder bei beliebigen Anfangskonfiguration der Punkte gegen einen Pyramidenwürfel?

Beobachtung: Mit $n = 20$ ist das Dodekaeder stabil, (siehe Abbildung 4.9 unten rechts). Der minimale Abstand des Dodekaeders beträgt 0,713644. Bei einer beliebigen Anfangskonfiguration der Punkte scheinen die Polyeder nicht gegen ein Dodekaeder zu konvergieren, sondern gegen ein Polyeder mit größerem minimalem Abstand als der des Dodekaeders. Entfernt man einen Punkt des Dodekaeders, so bekommt man auch ein stabiles Polyeder. In allgemeiner Lage der Punkte werden die Polyeder aber nicht gegen dieses Polyeder konvergieren, sie konvergieren gegen einen Körper mit größerem minimalem Abstand als der des geschnittenen Dodekaeders.

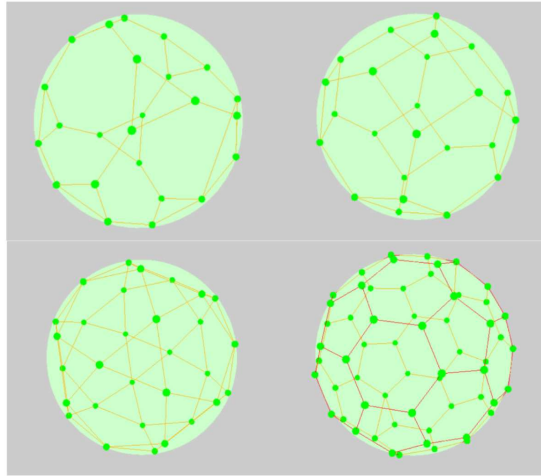


Abbildung 4.10: a, der abgestumpfte Würfel b, das abgestumpfte Oktaeder c, das Ikosidodekaeder d, das Fußball

Beobachtung: Mit $n = 24$ sind das Rhombenkuboktaeder und der abgeschrägte Würfel (cubus simus) stabil, der abgestumpfte Würfel (Cubus truncus) und das abgestumpfte Oktaeder (Octaeder truncum) instabil, (siehe Abbildung 4.10 oben). Der minimale Abstand des Rhombenkuboktaeders beträgt 0,7148, der des abgeschrägten Würfels beträgt 0,7442, der des abgestumpften Würfels vor dem Spiel ist 0,5622 und nach dem Spiel (es ist ein anderes Polyeder) ist 0,6879 und der des abgestumpften Oktaeders vor dem Spiel ist 0,6325 und nach dem Spiel ist 0,6929. Es bleibt die Frage: Konvergieren die Polyeder bei irgendwelchem Start und beim abgestumpften Würfel bzw. Oktaeder gegen einen abgeschrägten Würfel oder ein Rhombenkuboktaeder?

Dazu betrachten wir noch den durchschnittlichen Eckengrad des Polyeders: Er ist 4 beim Rhombenkuboktaeder und 5 beim abgeschrägten Würfel. Vor dem Spiel beim abgestumpften Würfel und abgestumpften Oktaeder ist er 3 und nach dem Spiel beträgt er etwa mehr als 5. Bei einer beliebigen Anfangskonfiguration ist der durchschnittliche Eckengrad etwa mehr als 5, aber der minimale Abstand ist gegen 0,7. Anscheinend werden weder den abgeschrägten Würfel noch das Rhombenkuboktaeder erreicht.

Beobachtung: $n = 30$ ist das Ikosidodekaeder stabil, (siehe Abbildung 4.10 unten links). Der minimale Abstand ist 0,6177. Startet man beliebig die Punkte, dann konvergiert das Polyeder nicht gegen ein Ikosidodekaeder. Der minimale

Abstand kann kleiner und auch größer sein als der des Ikosidodekaeders. Der durchschnittliche Eckengrad des Ikosidodekaeders beträgt 4, und beim Randomstart ist er mehr als 5.

Beobachtung: Mit $n = 60$ ist das abgestumpfte Ikosaeder (Ikosaeder truncum, Fußball) stabil, (siehe Abbildung 4.10 unten rechts). Der minimale Abstand ist 0,4120. Bei einem beliebigen Start konvergiert das Polyeder nicht gegen ein abgestumpften Ikosaeder. Der minimale Abstand ist durchschnittlich 0,433, also größer als 0,4120.

Beobachtung: Die Prismen und Antiprismen sind stabil, falls die Anzahl der Punkte nicht größer als 10 bei den Prismen und nicht größer als 8 bei den Antiprismen ist.

Beobachtung: Im stabilen Zustand ist der durchschnittliche Eckengrad des Polyeders zwischen 5 und 6, falls die Anzahl der Punkte mindestens 12 ist. Bei stabiler Konfiguration scheinen die Punkte auf der Kugeloberfläche wie im Fall des Kreises ungefähr hexagonal, gleichmässig verteilt zu sein, und die Seitenpolygone des Polyeders sind meistens Dreiecke und Vierecke. Die sphärische Packungsdichte besitzt den durchschnittlichen Wert 0,76 und die Messwerte liegen zwischen 0,66 und 0,88, (siehe Kapitel 5).

Kapitel 5

Einige Charakteristiken

5.1 Beschreibungsfunktion

Definition 5.1.1 Eine Beschreibungsfunktion $F : \mathbb{R}^n \rightarrow \mathbb{R}$ ist eine monoton wachsende (oder fallende) Funktion während des Verfahrens, die eventuell differenzierbar ist.

Betrachte das Gebiet, den Rand des Einheitskreises:

$$\Omega = \{(p_x, p_y) | p_x^2 + p_y^2 = 1, p_x \in \mathbb{R}, p_y \in \mathbb{R}\}.$$

Definition 5.1.2 Sei die Beschreibungsfunktion $E(v) : \mathbb{R}^n \rightarrow \mathbb{R}$ definiert durch:

$$E(v) := \frac{1}{2} \sum_{i,j} (v_i - v_j)^2 = \sum_{i,j} \omega_{i < j}^2 \quad i, j \in \{1, \dots, n\}$$

wobei $v \in \mathbb{R}^n$ die Winkelbeziehung zwischen den benachbarten Punkten angibt (Siehe im Beweis vom Satz 3.1.5), und $\omega_{ij} := |v_i v_j| = |v_j v_i|$.

Satz 5.1.3 In jedem Iterationsschritt wird $E(v)$ kleiner, das heißt $\Delta E(v) = \sum_{ij} \Delta \omega_{ij}^2 < 0$ ist, falls die neue Position des aktuellen beweglichen Punktes nicht die alte Position dieses Punktes ist, sonst $\Delta E(v) = 0$ ist.

Bemerkung 5.1.4 Sprungschritte sind nur erlaubt, falls der neue Abstand zum nächsten Nachbarn größer als der alte Abstand zum nächsten Nachbarn, sonst bleibt $E(v)$ unverändert, (Also in diesem Fall : neue Position = alte Position).

Beweis vom Satz 5.1.3 Seien o.B.d.a die Winkel v_1 und v_2 bei dem nächsten Iterationsschritt betrachtet und sei v_n der größte Winkel.

1. mit Sprung:

Nach dem Iterationsschritt haben wir: $v_1 = v_1 + v_2$ und $v_2 = v_n = \frac{v_n}{2}$,
daraus folgt:

$$\Delta E(v)' = \sum_{i < j} \Delta \omega_{ij}^2 = 0 \quad \text{mit } i \in \{3, \dots, n-2\} \quad j \in \{4, \dots, n-1\}$$

Für den restlichen Omegas rechnen wir ausführlich:

$$\begin{aligned} -(\Delta E(v) - \Delta E(v)') &= \sum_{i=2}^n (v_1 - v_i)^2 + \sum_{i=3}^n (v_2 - v_i)^2 + \sum_{i=3}^{n-1} (v_i - v_n)^2 - \\ &\quad - 2(v_1 + v_2 - \frac{v_n}{2})^2 - \sum_{i=3}^{n-1} (v_1 + v_2 - v_i)^2 - 2 \sum_{i=3}^{n-1} (\frac{v_n}{2} - v_i)^2 = \\ &\quad v_n^2(n-1) - 2v_1v_2 - v_n^2 \frac{1}{2}(n-2) - 2v_1v_2(n-1) = \frac{1}{2}nv_n^2 - 2nv_1v_2 \geq 0 \end{aligned}$$

falls $v_n = v_1 + v_2 \geq 2\sqrt{v_1v_2}$ und

$$-(\Delta E(v) - \Delta e(v)') = \frac{1}{2}nv_n^2 - 2nv_1v_2 > 0$$

falls $v_n > v_1 + v_2 \geq 2\sqrt{v_1v_2}$.

Also haben wir gesehen, dass $E(v) \leq 0$ ist, falls $v_1 + v_2 = v_n$ ist, und dass
 $E(v) < 0$ ist, falls $v_1 + v_2 < v_n$ ist.

2. ohne Sprung:

Nach dem Iterationsschritt gilt: $v_1 = v_2 = \frac{v_1+v_2}{2}$, daraus folgt:

$$\Delta \omega_{12}^2 = 0 \quad \Delta \omega_{ij}^2 = 0 \quad \forall i \in \{3, \dots, n-1\}$$

Betrachte $\Delta \omega_{1k}^2 + \Delta \omega_{2k}^2$ mit $v_1 + v_2 > v_3$, wobei $k \in \{3, \dots, n\}$.

Ausführlicher:

$$\begin{aligned} -(\Delta \omega_{1k}^2 + \Delta \omega_{2k}^2) &= (v_1 - v_k)^2 + (v_2 - v_k)^2 - 2(\frac{v_1 + v_2}{2} - v_k)^2 = \\ &= v_1^2 + v_2^2 + 2v_k^2 - 2(v_1v_k + v_2v_k) - 2(\frac{v_1 + v_2}{2} - v_k)^2 = \dots = \\ &\quad \frac{1}{2}v_1^2 + \frac{1}{2}v_2^2 - v_1v_2 \geq 0 \end{aligned}$$

□

Also haben wir gesehen, dass $\Delta \omega_{1k}^2 + \Delta \omega_{2k}^2 \leq 0$ ist. Falls $v_1 \neq v_2$ ist, gilt
insbesondere $\Delta \omega_{1k}^2 + \Delta \omega_{2k}^2 < 0$. Da k beliebig zwischen 3 und n liegt, folgt,
dass $\Delta E(v) = \sum_{i < j} \Delta \omega_{ij}^2 < 0$ ist, falls $v_1 \neq v_2$ ist, und $E(v) = 0$ sonst.

Bemerkung 5.1.5 *Der Gradient von $E(v)$*

$$\nabla E(v) = \begin{pmatrix} \frac{\partial E(v)}{\partial v_1} \\ \vdots \\ \frac{\partial E(v)}{\partial v_n} \end{pmatrix} = n \begin{pmatrix} v_1 - \frac{2\pi}{n} \\ \vdots \\ v_n - \frac{2\pi}{n} \end{pmatrix}$$

ist genau dann Null, falls $v = \begin{pmatrix} \frac{2\pi}{n} \\ \vdots \\ \frac{2\pi}{n} \end{pmatrix}$ ist, das heißt die einzige (analytische) stabile Konfiguration ist dadurch charakterisiert, dass die Punkte äquidistant am Kreisrand liegen.

Betrachten wir alle Winkel $v = \{v_0, \dots, v_s\}$ zwischen den Voronoi-benachbarten Punkten auf der Kugeloberfläche bezüglich des Mittelpunktes der Kugel. Wir definieren die Beschreibungsfunktion also

$$E(v) = \frac{1}{2} \sum_{i=0}^s v^2$$

Wird $E(v)$ in jedem Iterationsschritt kleiner? Leider ist die Antwort noch unbekannt, also fragen wir:

Gibt es Beschreibungsfunktion (monoton wachsend oder fallend) für die Fälle:

$$\Omega = \{(p_x, p_y)^t \mid p_x^2 + p_y^2 \leq 1, p_x \in \mathbb{R}, p_y \in \mathbb{R}\}$$

der Einheitskreis mit seinem Inneren oder

$$\Omega = \{(p_x, p_y, p_z)^t \mid p_x^2 + p_y^2 + p_z^2 = 1, p_x \in \mathbb{R}, p_y \in \mathbb{R}, p_z \in \mathbb{R}\}$$

die Oberfläche der Einheitskugel?

Satz 5.1.6 *Ein Ansatz ist für die Beschreibungsfunktion $D : \Omega^n \rightarrow \mathbb{R}$, die während des Verfahrens monoton wachsend ist, die Berechnung des minimalen Abstands der Punkte [10].*

$$D(P) = \min_{i,k=0,\dots,n-1, i \neq k} |p_i p_k|,$$

wobei $P = \{p_0, \dots, p_{n-1}\}$ die Punktmenge im Spiel ist.

Beweis vom Satz 5.1.6 Bezeichne $Q_i \in \mathbb{R}$ den Abstand des Punktes p_i zu seinem nächsten Nachbarn für alle $i \in \{0, \dots, n-1\}$. Dann gilt $D(P) = \min_{i=0,\dots,n-1} Q_i$. Sei $Q_m := D(P)$. Es gibt zwei Möglichkeiten bei einem Iterationsschritt.

n	$D(P)$ OV	$D(P)$ MV	$\frac{1}{\sqrt{n}}$	$\frac{2\sqrt{2}}{\sqrt{n}}$	Packungsdichte
2	2,0	*	0,707	2,0	0,5
3	$\sqrt{3}$	*	0,577	1,633	0,64619
4	$\sqrt{2}$	1,411	0,5	$\sqrt{2}$	0,6862
5	1,172	1,1747	0,447	1,265	0,6826
6	1,0	0,998	0,408	1,154	0,6666
7	1,0	1,0	0,377	1,069	0,777
8	0,86	0,846	0,35	1,0	0,7233
9	0,764	0,738	0,33	0,942	0,6876
10	0,7047	0,7	0,31	0,89	0,6788
11	0,67	0,673	0,3	0,85	0,6926
12	0,64	0,62	0,288	0,816	0,7052
13	0,61	0,5929	0,277	0,784	0,71
14	0,58	0,587	0,267	0,756	0,7075
15	0,548	0,5415	0,258	0,73	0,6938

Tabelle 5.1: Messungen des minimalen Abstandes und Berechnung der Packungsdichte im Kreis

Die erste Möglichkeit ist, dass der nächste Punkt im Spiel p_m ist. Nach dem Iterationsschritt ist $\Delta Q_m \geq 0$ und damit gilt $\Delta D(P) \geq 0$.

Die zweite Möglichkeit ist, dass der nächste Punkt im Spiel p_a ist, und $p_a \neq p_m$. Vor und nach dem Iterationsschritt ist $Q_m \leq Q_a$ und damit gilt $\Delta D(P) \geq 0$. \square

In den Tabellen 5.1., 5.2 sind die Messwerte des minimalen Abstandes $D(P)$ im Fall des Einheitskreises zu sehen. In der ersten Spalte steht die Anzahl der Punkte, und der Kreis ist der Einheitskreis. Die Werte des Spieles sind ohne Voronoi-Knoten (Kreisrand) Test in der zweiten Spalte, mit Voronoi-Knoten Test in der dritten Spalte zu sehen. In der vierten und fünften Spalte stehen die Werte der beobachteten unteren bzw. oberen Schranken, (siehe Abbildung 5.1). In der letzten Spalte ist die Packungsdichte zu sehen.

In den Tabellen 5.3, 5.4 sind die Messwerte des minimalen Abstandes $D(P)$ im Fall der Oberfläche der Einheitskugel zu sehen. In den ersten fünf Spalten stehen dieselben Quantitäten wie im Fall des Kreises. In der sechsten Spalte findet man eine bessere obere Schranke für die Messungen (siehe Abbildung 5.2), und in der siebten Spalte findet man die sphärische Packungsdichte.

n	$D(P)$ OV	$D(P)$ MV	$\frac{1}{\sqrt{n}}$	$\frac{2\sqrt{2}}{\sqrt{n}}$	Packungsdichte
16	0,524	0,525	0,25	0,707	0,6896
17	0,52	0,523	0,24	0,686	0,7238
18	0,509	0,51	0,235	0,66	0,74
19	0,48	0,5076	0,23	0,64	0,7117
20	0,45	0,4743	0,223	0,632	0,6747
26	0,39	0,387	0,196	0,554	0,6923
32	0,34	0,347	0,176	0,5	0,6994
40	0,307	0,3125	0,158	0,447	0,73
50	0,265	0,271	0,141	0,4	0,712
60	0,245	0,246	0,129	0,365	0,7197
70	0,22	0,225	0,119	0,338	0,7158
80	0,204	0,2059	0,112	0,316	0,7
90	0,194	0,1955	0,105	0,298	0,7136
100	0,18	0,183	0,1	0,28	0,7027

Tabelle 5.2: Messungen des minimalen Abstandes und Berechnung der Packungsdichte im Kreis

n	$D(P)$ OV	$D(P)$ MV	$\frac{1}{\sqrt[3]{n}}$	$\frac{2\sqrt[3]{3}}{\sqrt[3]{n}}$	$\frac{2\sqrt{3}}{\sqrt{n}}$	Dichte
2	2,0	**	0,79	2,28	2,45	0,999
3	$\sqrt{3}$	**	0,69	2,0	2,0	0,75
4	1,629	1,63	0,63	1,817	1,732	0,841
5	$\sqrt{2}$	1,4127	0,58	1,68	1,55	0,7322
6	$\sqrt{2}$	1,408	0,55	1,587	$\sqrt{2}$	0,8786
7	1,244	1,244	0,52	1,507	1,309	0,7594
8	1,1469	1,182	0,5	1,44	1,224	0,7415
9	1,137	1,14	0,48	1,38	1,154	0,8025
10	1,07	1,0721	0,46	1,33	1,095	0,779
11	1,038	1,035	0,44	1,29	1,044	0,7987
12	1,0	1,034	0,43	1,25	1,0	0,864
13	0,93	0,935	0,425	1,22	0,96	0,754
14	0,911	0,9	0,415	1,196	0,925	0,7683
15	0,857	0,858	0,405	1,169	0,894	0,7252

Tabelle 5.3: Messungen des minimalen Abstandes und Berechnung der sphärischen Packungsdichte auf der Kugeloberfläche

n	$\mathbf{D(P)}$ OV	$\mathbf{D(P)}$ MV	$\frac{1}{\sqrt[3]{n}}$	$\frac{2\sqrt[3]{3}}{\sqrt[3]{n}}$	$\frac{2\sqrt{3}}{\sqrt{n}}$	Dichte
16	0,858	0,856	0,396	1,144	0,866	0,7735
17	0,82	0,826	0,38	1,12	0,84	0,7587
18	0,793	0,809	0,38	1,1	0,8164	0,7691
19	0,789	0,778	0,37	1,08	0,794	0,77
20	0,764	0,747	0,36	1,06	0,774	0,7583
26	0,673	0,6762	0,337	0,973	0,679	0,7581
32	0,614	0,586	0,31	0,9	0,6123	0,7726
40	0,537	0,536	0,29	0,84	0,547	0,7344
50	0,478	0,438	0,27	0,783	0,489	0,7245
60	0,433	0,412	0,25	0,73	0,447	0,115
80	0,372	0,3696	0,23	0,66	0,387	0,698
100	0,325	0,3223	0,215	0,621	0,34	0,6645

Tabelle 5.4: Messungen des minimalen Abstandes und Berechnung der sphärischen Packungsdichte auf der Kugeloberfläche

Beobachtung: Im stabilen Zustand ist der minimale Abstand im Kreis zwischen zwei beliebigen Punkten $\frac{1}{\sqrt{n}} \leq D(P) \leq \frac{2\sqrt{2}}{\sqrt{n}}$, und die Werte des minimalen Abstandes liegen nahe bei $\frac{2}{\sqrt{n}}$.

Satz 5.1.7 *Im stabilen Zustand ist im Kreis der minimale Abstand zwischen zwei beliebigen Punkten ist $O(\frac{1}{\sqrt{n}})$, (Vermutung $D(P) \in \Theta(\frac{1}{\sqrt{n}})$).*

Beweis vom Satz 5.1.8 Bezeichne d der minimale Abstand ($d = D(P)$) und sei n beliebig. Wir konstruieren einen Hilfskreis, dessen Radius doppelt so lang ist wie der Einheitskreis, (siehe Abbildung 5.3). Im stabilen Zustand kreisen wir mit $r = \frac{d}{2}$ um alle Punkte herum. Da der minimale Abstand r kleiner als 2 ist, liegen alle Kreise mit Radius r im Hilfskreis. Der Flächeninhalt des Hilfskreises ist durch die Formel $A_h = 4\pi$ gegeben. Es gilt Folgendes für die Summe A_s vom Flächeninhalt der n Kreise mit Radius r .

$$A_s = n \cdot r^2 \cdot \pi = n \cdot \left(\frac{d}{2}\right)^2 \cdot \pi \leq A_h = 4\pi \quad \rightarrow \quad \frac{n \cdot d^2}{16} \leq 1$$

Daraus folgt, dass $d \leq \frac{4}{\sqrt{n}}$ ist. □

Beobachtung: Im stabilen Zustand ist auf der Kugeloberfläche der minimale Abstand zwischen zwei beliebigen Punkten $\frac{1}{\sqrt{n}} \leq D(P) \leq \left(\frac{2\sqrt[3]{3}}{\sqrt[3]{n}}\right)$, wenn die Anzahl der Punkte nicht zu groß ist, (Es ist mit Programm relativ schnell messbar, siehe Tabelle 5.3, 5.4). Die Werte des minimalen Abstandes liegen nahe bei $D(P) \leq \frac{2\sqrt{3}}{\sqrt{n}}$.

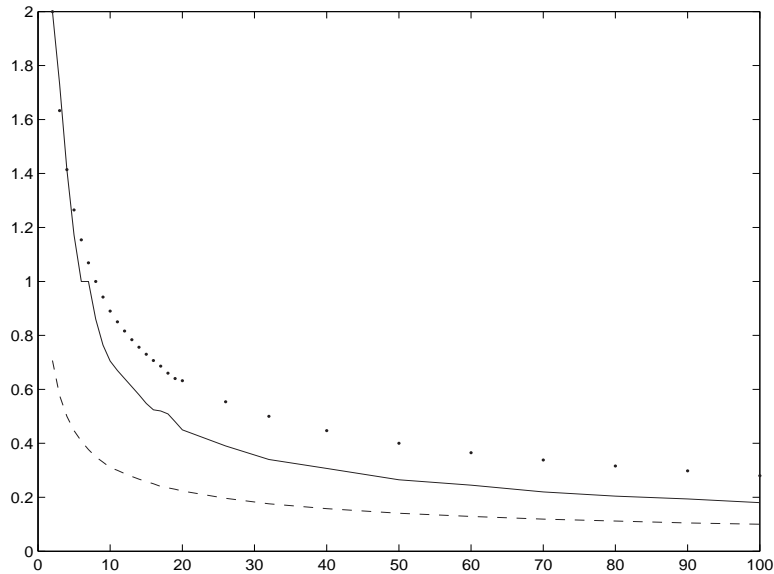


Abbildung 5.1: Der minimale Abstand im Kreis '-', untere Schranke '-', obere Schranke '.'

Satz 5.1.8 *Im stabilen Zustand ist der minimale Abstand auf der Kugeloberfläche zwischen zwei beliebigen Punkten ist $O(\frac{1}{\sqrt{n}})$, (Vermutung $D(P) \in \Theta(\frac{1}{\sqrt{n}})$).*

Beweis vom Satz 5.1.8 Bezeichne d wieder den minimalen Abstand ($d = D(P)$), und sei n beliebig. Der Oberflächeninhalt der Einheitskugel ist durch die Formel $A_e = 4R^2\pi = 4\pi$ gegeben. Im stabilen Zustand kreisen wir auf der Kugeloberfläche mit Radius $r = \frac{d}{2}$ um die Punkte herum, das heißt r ist gleich dem Euklidischen Abstand zwischen dem Mittelpunkt und einem Randpunkt, und nicht dem Abstand auf der Kugeloberfläche, (siehe Abbildung 5.4). Die sphärischen Kreise überlappen sich nicht. Jetzt nehmen wir an, dass der Flächeninhalt eines solchen sphärischen Kreises so groß ist, als ob er in der Ebene liegen würde. (In der Wirklichkeit ist es größer). Dann gilt für die Summe vom Oberflächeninhalt der n sphärischen Kreise mit Radius r .

$$A_s = n \cdot r^2 \cdot \pi = n \cdot \left(\frac{d}{2}\right)^2 \cdot \pi \leq A_e = 4\pi \quad \rightarrow \quad \frac{n \cdot d^2}{16} \leq 1$$

Daraus folgt, dass $d \leq \frac{4}{\sqrt{n}}$ ist. Da der Flächeninhalt eines sphärischen Kreises in der Wirklichkeit größer ist, kann der minimale Abstand nicht größer sein als d . □

Im zweiten Kapitel haben wir gesehen, dass das Voronoi Spiel theoretisch mit und ohne Voronoi-Knoten-(Randpunkt-)Test dasselbe Ergebnis liefert. Deswegen ist es nicht überraschend, dass sich die Messwerte des minimalen Abstandes kaum

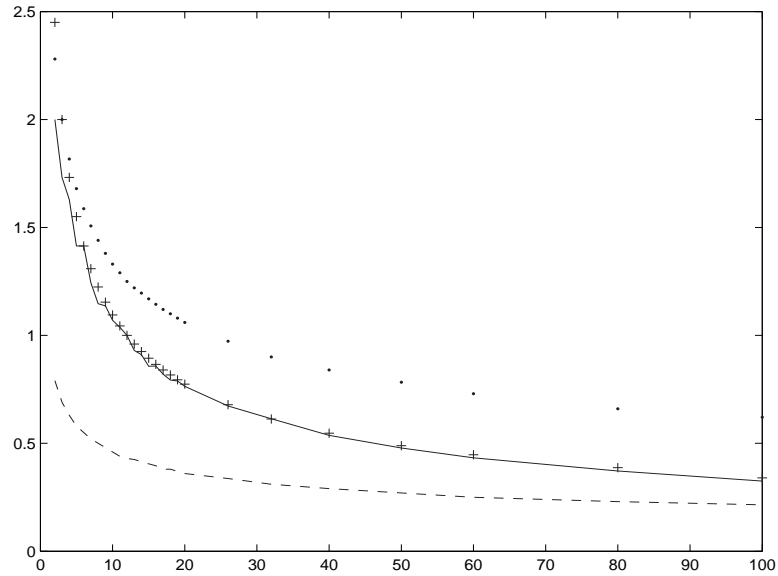


Abbildung 5.2: Der minimale Abstand auf der Kugel '-', untere Schranke '-', obere Schranke '+', bessere (kleinere) obere Schranke '+'

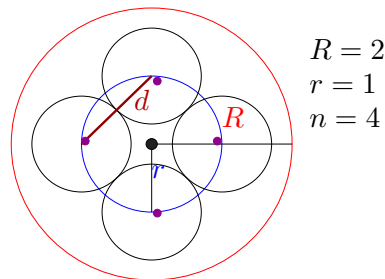


Abbildung 5.3:

in den beiden Implementierungen unterscheiden, (siehe noch in Kapitel 6).

Da die konvexe Hülle von n Kreisen mit demselben Radius $r = \frac{d}{2}$ (die Hälfte des minimalen Abstandes) schwer zu berechnen ist, nehmen wir für das benutzte Volumen (siehe Kapitel 3) den Kreis mit Radius $1+r$, (mit demselben Mittelpunkt des Einheitskreises, wo sich die Punkte befinden) Damit bekommen wir eine etwas kleinere Kreispackungsdichte, was man beim Vergleich mit anderen Resultaten beachten soll. Also die Packungsdichte wird so berechnet:

$$d(B^2, C_n) = \frac{n \cdot r^2 \pi}{(1+r)^2 \pi}$$

Die Werte der Packungsdichte sind gegen 0.7. Dieses Resultat haben wir schon in Kapitel 3 mit der Theorie der Kugelpackungen verglichen.

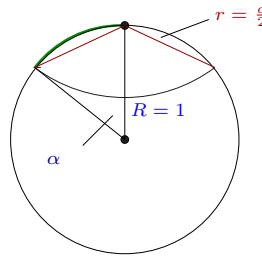


Abbildung 5.4:

Auf der Kugeloberfläche können wir auch die Packungsdichte definieren. Hier betrachten wir die ganze Kugeloberfläche als das benutzte Volumen und das genutzte Volumen sind n sphärischen Kreise mit (sphärischem) Radius $\alpha = \arcsin \frac{d}{2}$, (siehe Abbildung 5.4). Wir definieren die sphärische Packungsdichte folgendermaßen:

$$d_s(B_s^2, C_n) = \frac{n \cdot \text{Vol}(\text{sph. Kreis})}{4\pi} = \frac{n \cdot 2\pi(1 - \cos \alpha)}{4\pi}$$

Woher kommt diese Formel für den Flächeninhalt des sphärischen Kreises?

Flächen sind als zweidimensionale Punktmenge durch Parameterdarstellungen $P = (\lambda, t)$ definiert [6]. Für die Parameterdarstellung allgemeiner Rotationsflächen benötigt man eine Darstellung der Profilkurve. Hier wird davon ausgegangen, dass eine in der (x, z) -Ebene verlaufende Profilkurve um die z -Achse rotiert. Es ergibt sich also:

$$P(\lambda, t) = \begin{pmatrix} \cos \lambda & -\sin \lambda & 0 \\ \sin \lambda & \cos \lambda & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \rho(t) \\ 0 \\ h(t) \end{pmatrix} = \begin{pmatrix} \rho(t) \cos \lambda \\ \rho(t) \sin \lambda \\ h(t) \end{pmatrix}$$

wobei $0 \leq \lambda \leq 2\pi$, $t_0 \leq t \leq t_1$.

Für den sphärischen Kreis mit Radius α und Mittelpunkt $(0, 0, 1)$ ist die Profilkurve durch den in der (x, z) -Ebene verlaufende Kreissektor mit Winkel α gegeben:

$$\begin{pmatrix} \rho(t) \\ 0 \\ h(t) \end{pmatrix} = \begin{pmatrix} \sin t \\ 0 \\ \cos t \end{pmatrix} \quad 0 \leq t \leq \alpha$$

und damit die Parameterdarstellung für den sphärischen Kreis:

$$P(\lambda, t) = \begin{pmatrix} \sin t \cos \lambda \\ \sin t \sin \lambda \\ \cos t \end{pmatrix}$$

wobei $0 \leq \lambda \leq 2\pi$, $0 \leq t \leq \alpha$, $\alpha = \arcsin \frac{d}{2}$.

Der Inhalt einer durch $P(\lambda, t)$ parametrisierten gekrümmten Fläche ist [6]:

$$\int_{t_0}^{t_1} \int_0^{2\pi} \left| \frac{\partial P}{\partial \lambda} \times \frac{\partial P}{\partial t} \right| d\lambda dt = 2\pi \int_{t_0}^{t_1} \rho(t) \sqrt{((h'(t))^2 + (\rho'(t))^2)} d\lambda dt$$

Wir rechnen also die Fläche des sphärischen Kreises:

$$2\pi \int_0^\alpha \sin(t) \sqrt{\sin^2 t + \cos^2 t} dt = 2\pi \int_0^\alpha \sin(t) dt = 2\pi(1 - \cos \alpha)$$

Der durchschnittliche Wert der numerischen Ergebnisse (nur bis 100 Punkte gemessen) der Packungsdichte beträgt 0,76. Nimmt man anstatt des minimalen Abstandes die Werte $\frac{2\sqrt{3}}{\sqrt{n}}$, die nahe bei den Werten des minimalen Abstandes liegen, dann konvergiert die Packungsdichte gegen 0,75. Es fällt aber auf, dass die Packungsdichte für größere n etwas abnimmt, was an der Ungenauigkeit der Berechnung auf dem Rechner liegen kann. Deshalb bleibt die Frage offen, wie groß kann theoretisch die Packungsdichte sein, wenn mit beliebig vielen Punkte gespielt wird?

5.2 Konvergenz des Verfahrens

In diesem Abschnitt geht es um die Frage, ob das Verfahren konvergiert. Dazu werden wir im ersten Teil einige Vermutungen kennenlernen, und im zweiten Teil betrachten wir numerische Ergebnisse.

Theorie

Die Frage, ob das Verfahren nach beliebig vielen Iterationsschritten konvergiert, wurde auf zwei verschiedenen Wegen angenähert. Der erste Weg ist ähnlich demjenigen, den wir im Kapitel 3 gesehen haben. Die folgenden Vermutungen gelten sowohl für den Kreis als auch für die Kugeloberfläche.

Vermutung 5.2.1 *Es gibt zwei Phasen im Spiel, das heißt es existiert ein $k \in \mathbb{N}$, so dass gilt: falls der Zähler $c < k$ ist, kann noch Sprung vorkommen, also kann sich die Nachbarschaftsbeziehung ändern, sonst falls $c \geq k$ ist, dann ist kein Sprung mehr möglich, also die Nachbarschaftsbeziehung bleibt unverändert.*

Vermutung 5.2.2 *Die Anzahl der Sprünge sind beschränkt.*

Ist diese Anzahl $s \in O(\log n)$? Welche Parameter sollte man benutzen, um diese Vermutungen bestätigen zu können? Falls man auf der Kugeloberfläche alle Winkel zwischen zwei Voronoi-benachbarten Punkten bezüglich des Mittelpunktes der Kugel betrachtet, ist es ein guter Ansatz?

Auf Grund der vorherigen Vermutungen kann man noch eine Vermutung formulieren:

Vermutung 5.2.3 *Wenn sich die Nachbarschaftsbeziehung nach endlich vielen Iterationsschritten nicht mehr ändert, konvergiert das Verfahren.*

Bemerkung 5.2.4 *In diesem Fall hängt in jedem Iterationsschritt die Position des beweglichen Punktes nur von seinen Voronoi-Nachbarn ab, die anderen Punkte haben keinen Einfluß auf diesen Punkt.*

Der zweite Weg hat mehr analytische Aspekte als der erste. Die Definitionen und einige Behauptungen stammen aus dem Buch: Königsberger Analysis 1.

Definition 5.2.5 [3] *Eine Folge $\{a_m\}_{m \in \mathbb{N}} \in \mathbb{R}^n$ heißt **konvergent**, wenn es eine Zahl $a \in \mathbb{R}^n$ mit folgender Eigenschaft gibt: Zu jedem $\epsilon > 0 \exists N \in \mathbb{N}$ so, dass*

$$|a_m - a| < \epsilon \quad \forall m > N.$$

Die Zahl a heißt **Grenzwert** und man schreibt

$$\lim_{m \rightarrow \infty} a_m = a$$

Definition 5.2.6 [3] $h \in \mathbb{R}^n$ heißt **Häufungswert** der Folge $\{a_m\}_{m \in \mathbb{N}}$, wenn zu jedem $\epsilon > 0$ gilt:

$$|h - a_m| < \epsilon$$

Definition 5.2.7 [3] *Ist $\{a_m\}_{m \in \mathbb{N}}$ eine Folge und $\{m_k\}$ eine streng monoton wachsende Folge von Indizes, so heißt die durch $k \rightarrow a_{m_k}$, $k \in \mathbb{N}$ definierte Folge $\{a_{m_k}\}_{k \in \mathbb{N}}$ **Teilfolge** von $\{a_m\}$.*

Lemma 5.2.8 [3] $h \in \mathbb{R}^n$ ist Häufungswert einer Folge $\{a_m\}_{m \in \mathbb{N}}$ genau dann, wenn h der Grenzwert einer Teilfolge $\{a_{m_k}\}_{k \in \mathbb{N}}$ ist.

Satz von Bolzano-Weierstraß [3] *Jede beschränkte Folge besitzt eine konvergente Teilfolge.*

Bezeichne $D(P)^{(k)}$ den minimalen Abstand nach dem k -ten Iterationsschritt. Sei die Folge $\{a_k\}_{k \in \mathbb{N}}$ durch $a_k = D(P)^{(k)}$ definiert. Diese Folge ist beschränkt (im Einheitskreis und in der Einheitskugel ist $D(P)^{(k)} \leq 2$), also besitzt eine konvergente Teilfolge und damit einen Häufungswert $h \in \mathbb{R}$. Da die Folge $\{a_k\}$ monoton wachsend ist, hat sie nur einen Häufungswert, also ist dieser Häufungswert der Grenzwert der Folge $\{a_k\}$, das heißt die Folge konvergiert. Bezeichne $d(P)$ den Grenzwert der Folge:

$$d(P) = \lim_{k \rightarrow \infty} D(P)^{(k)} \quad (1)$$

Nun bezeichne $p_i^{(k)}$, $i \in \{0, \dots, n-1\}$ den i -ten Punkt im Kreis bzw. auf der Kugeloberfläche nach dem k -ten Iterationsschritt. (Hier werden nur solche Iterationsschritte mitgezählt, bei denen sich der Punkt p_i bewegt.) Also am Anfang vor dem Spiel haben wir die Punkte $p_i^{(0)}$. Seien die Folgen $\{b_k^i\}_{k \in \mathbb{N}}$ durch

$$b_k^i = p_i^{(k)} \quad i \in \{0, \dots, n-1\}$$

definiert.

Die Folgen $\{b_k^i\}$ sind beschränkt, da die Gebiete: Abschluß vom Kreis und die Kugeloberfläche beschränkt sind. Also folgt aus dem Lemma und Satz von Bolzano-Weierstraß, dass jede Folge $\{b_k^i\}$ mindestens einen Häufungswert h_i besitzt. Um etwas mehr sagen zu können, benutzen wir noch eine Behauptung vom Königsberger' Buch:

Satz Bolzano-Weierstraß-Charakterisierung:[3] Eine Teilmenge $K \subset \mathbb{R}^n$ ist genau dann kompakt, wenn jede Folge in K eine Teilfolge besitzt, die gegen einen Punkt in K konvergiert.

Aus diesem Satz folgt, dass die Häufungswerte der Folgen $\{b_k^i\}$ in Ω , also im Kreis bzw. auf der Kugeloberfläche liegen. Wir haben also insgesamt festgestellt, dass alle Folgen $\{b_k^i\}_{k \in \mathbb{N}}$, $i \in \{0, \dots, n-1\}$, wobei $b_k^i = p_i^{(k)}$, im Gebiet Ω mindestens einen Häufungswert haben. (2)

Konvergiert jede Folge $\{b_k^i\}$, das heißt existiert das Limes:

$$q_i = \lim_{k \rightarrow \infty} b_k^i = \lim_{k \rightarrow \infty} p_i^{(k)} = ? \quad i \in \{0, \dots, n-1\}$$

Jetzt modifizieren wir das Spiel: In jedem Iterationsschritt bewegt sich ein Punkt so, dass der minimaler Abstand $D(P)$ monoton wächst. Das Voronoi Spiel ist dann Spezialfall von diesem.

Gelte die Voraussetzungen (1) und (2) für eine endliche Punktmenge. Garantiert diese zwei Tatsache, dass die Folgen $\{b_k^i\}$ konvergieren? (Falls eine Folge mehrere Häufungswerte besitzt, sind sie gleich?)

Die Antwort bekommen wir mit dem folgenden Satz:

Satz 5.2.9 Sei eine Punktmenge $P = \{p_0, \dots, p_{n-1}\}$ in einem kompakten (abgeschlossenen und beschränkten) Gebiet Ω gegeben. Die Punkte bewegen sich nach den Regeln des modifizierten Spieles.

1. Es gelte: Jede Folge $\{b_k^i\}_{k \in \mathbb{N}}$, wobei $b_k^i = p_i^{(k)}$ und $i \in \{0, \dots, n-1\}$, besitzt einen Häufungswert h_i in Ω .
2. Es gelte: Der minimaler Abstand $D(P)$ ist monoton wachsend während des Verfahrens und es gilt: Die Folge $\{a_k\}_{k \in \mathbb{N}}$, wobei $a_k = D(P)^{(k)}$, hat den Grenzwert $d(p)$.

Dann müssen die Folgen $\{b_k^i\}$ nicht unbedingt konvergieren.

Beweis vom Satz 5.2.9 Wir geben ein Gegenbeispiel zur Konvergenz an. Es gebe zwei Punkte p_0 und p_1 im Spiel. Sei das Gebiet Ω der Einheitskreis mit seinem Inneren mit dem Zentrum $(0,0)$. Die Punkte bewegen sich folgendermaßen:

$$p_0^{(k)} = \left(\frac{2^k - 1}{2^k}, 0\right), \quad \text{und} \quad p_1^{(k)} = \left(0, (-1)^k \cdot \frac{2^k - 1}{2^k}\right)$$

Nach der Konstruktion gilt, dass $D(P)$ monoton wachsend ist. Die Folge $\{b_k^0\}$ hat einen Häufungswert $h_0 = (1, 0)$, aber die Folge $\{b_k^1\}$ hat zwei Häufungswerte $h_{11} = (0, 1)$ und $h_{12} = (0, -1)$. \square

Bemerkung 5.2.10 Wären die Punkte im Spiel so definiert:

$$p_0^{(k)} = \left(\frac{2^k - 1}{2^k}, 0\right), \quad \text{und} \quad p_1^{(k)} = \left(0, \frac{2^k - 1}{2^k}\right)$$

so hätte die Folge $\{b_k^0\}$ einen Grenzwert $q_0 = (1, 0)$ und die Folge $\{b_k^1\}$ einen Grenzwert $q_1 = (0, 1)$.

Wir haben also gesehen, dass die Existenz der Häufungswerte der Folge in Gebiet Ω und das monotone Wachstum sowie die Existenz des Grenzwertes des minimalen Abstandes nicht ausreichend für die Konvergenz ist. Also es scheint wichtig zu sein, dass man die neue Position eines Punktes immer mit dem größtmöglichen Abstand zum nächsten Nachbarn auswählt. Es führt zu unserer letzten Vermutung:

Vermutung 5.2.11 Wenn man noch bei dem vorherigen Satz verlangt, dass in jedem Iterationsschritt der Abstand des beweglichen Punktes zu seinem nächsten Nachbarn maximal sein soll, das heißt die Punkte bewegen sich nach den Regeln des Voronoi Spieles, dann konvergieren die Folgen $\{b_k^i\}$ für alle $i \in \{0, \dots, n-1\}$, und der Grenzwerte q_i existieren:

$$\lim_{k \rightarrow \infty} b_k^i = \lim_{k \rightarrow \infty} p_i^{(k)} = q_i \quad i \in \{0, \dots, n-1\}$$

mit anderen Worten, falls eine Folge mehrere Häufungswerte besitzt, sind die Häufungswerte gleich.

n	OV	MV	n	OV	MV
2	3	*	17	155	182
3	10	*	18	168	205
4	18	13	19	190	228
5	24	17	20	193	254
6	27	17	30	353	302
7	56	53	40	555	561
8	64	39	50	836	876
9	86	51	60	1186	1388
10	111	70	70	1530	1119
11	165	71	80	1668	1375
12	129	68	90	1307	1124
13	117	151	100	2195	1599
14	113	102			
15	120	100			
16	127	260			

Tabelle 5.5: Messungen im Kreis

Numerische Ergebnisse

Die Vermutung 5.2.11 basiert auf folgenden Messungen. Es wurde in mehreren Spielen gemessen, wieviele Iterationsschritte nötig sind, um die stabile Konfiguration zu erreichen.

Bezeichne H_i den Fehler zwischen zwei nacheinanderfolgenden Positionen des i -ten Punktes: $H_i = |p_i^{(\text{letzte})} p_i^{(\text{vorletzte})}|$, das heißt dieser Fehler gibt an, wie weit sich der neue und der alte i -ten Punkt voneinander befinden. In jedem Iterationsschritt wird die Summe der Fehlerquadrate

$$H(P) = \sum_{i=0}^{n-1} H_i^2$$

gemessen. Wenn $H(p) = 0$ während einer Runde ist, das heißt während n nacheinanderfolgenden Iterationsschritten, ist der stabile Zustand der Punkte erreicht, also bewegen sich die Punkte nicht mehr.

In der Tabelle 5.5 sind die Ergebnisse im Fall des Kreises zu sehen: In der ersten und vierten Spalte steht die Anzahl der Punkte. Die Zahlen in der zweiten

n	OV	MV	n	OV	MV
2	3	*	17	152	116
3	13	*	18	114	177
4	21	17	19	153	134
5	18	18	20	138	146
6	29	29	30	226	332
7	29	30	40	318	495
8	49	34	50	319	477
9	62	46	60	651	454
10	83	60	70	650	775
11	103	94	80	632	611
12	84	73	100	1623	846
13	119	103			
14	103	77			
15	107	125			
16	138	145			

Tabelle 5.6: Messungen auf der Kugeloberfläche

und fünften bzw. dritten und sechsten Spalte geben an, wieviele Iterationsschritte nötig sind, um den stabilen Zustand zu erreichen. In der Tabelle 5.6 sind die Ergebnisse im Fall der Kugeloberfläche zu sehen: In den Spalten stehen die gleiche Quantitäten als im Fall des Kreises. **OV** bedeutet, dass das Voronoi Spiel ohne Voronoi-Knoten-(Randpunkt-)Test durchgeführt wurde, und **MV** bedeutet, dass das Voronoi Spiel mit Voronoi-Knoten-(Randpunkt-)Test abgespielt wurde.

Beobachtung: Um den stabilen Zustand der Punkte zu erreichen, wird öfters weniger Iterationsschritte gebraucht, falls das Voronoi Spiel mit Voronoi-Knoten-(Randpunkt-)Test durchgeführt wird.

Bezeichne wieder n die Anzahl der Punkte. Sowohl im Kreis als auch auf der Kugeloberfläche scheinen die benötigten Iterationsschritte zwischen $n \log n$ und n^2 zu liegen. Es weist auf die Konvergenz des Voronoi Spieles hin.

Kapitel 6

Implementierungen

In diesem Kapitel wird es vorgestellt, wie der im ersten Kapitel eingeführte Algorithmus im Kreis und auf der Kugeloberfläche implementiert wurde. In beiden Fällen wurde der Algorithmus auf zwei verschiedene Weisen realisiert.

Im ersten Fall betrachten wir folgende Punkte, um die neue Position des beweglichen Punktes zu bestimmen: Man betrachtet alle Punkte in einem Gitter, das den Kreis abdeckt oder auf der Kugeloberfläche liegt. Die Verfeinerung des Gitters kann unterschiedlich eingestellt werden, das heißt wie groß der Abstand zwischen zwei benachbarten Punkten im Gitter ist.

Im zweiten Fall betrachten wir folgende Punkte, um die neue Position des beweglichen Punktes zu bestimmen: man berücksichtigt nur die Voronoi-Knoten des Voronoi-Diagramms der restlichen $n - 1$ festen Punkte. Im Fall des Kreises werden in jedem Iterationsschritt auch die Punkte am Kreisrand auch betrachtet, (siehe Kapitel 2).

Die Methoden, wie das Voronoi Spiel realisiert wurde, werden in der Gliederung ausführlicher beschrieben und danach wird das Ergebnis nach genügend vielen Iterationsschritten in beiden Methoden verglichen (Gitter oder mit Voronoi-Knoten).

Es wird noch dargestellt, wie die Voronoi-Knoten gefunden werden können, und wie die Voronoi-Nachbarschaft der Punkte ermittelt werden kann. Die Voronoi-Nachbarschaft kann nach jedem Iterationsschritt angegeben werden. Im Kreis erhält man damit die bekannte Delaunay-Zerlegung. Auf der Kugeloberfläche werden aber nicht direkt die sphärische Delaunay-Zerlegung implementiert, sondern das Delaunay-Polyeder wird gezeichnet, eigentlich das Polyeder, das die Punkte auf der Kugeloberfläche eindeutig bestimmen, (siehe Kapitel 2).

Es werden noch einige Generierungsmethoden beschrieben, wie die regulären, halbrekulären Polyeder und Prismen und Antiprismen im Applet der Kugel konstruiert wurden.

6.1 Die zwei Realisierungen des Spieles

Implementierung mit Gitterpunkten im Kreis: ohne Voronoi-Knoten-Randpunkt-Test

Es reicht, wenn wir nur einen Iterationsschritt betrachten, wobei ein Punkt beweglich ist und die restlichen $n - 1$ Punkte p_1, \dots, p_n nicht, sie werden also festgehalten. Sei o.B.d.A p_0 dieser bewegliche Punkt, und p_0 muss so weit wie möglich von den anderen Punkten im Kreis gesetzt werden. (Also der Abstand zum nächsten Nachbarn muss maximal sein.) Der Kreis hat im Applet den Mittelpunkt $(250, 250)$. Dazu sei erwähnt, dass die Koordinaten der Punkte Integer-Werte haben. Die x -Achse läuft von links nach rechts und die y -Achse läuft von oben nach unten, das heißt das Origo befindet sich in der oberen-linken Ecke des Appletfensters. Der Kreis hat einen 200 langen Radius, z.B der linkeste Randpunkt des Kreises hat die Koordinaten $(50, 250)$, der unterste Randpunkt hat die Koordinaten $(250, 450)$. In jedem Iterationsschritten betrachten wir das Gitter mit obere-linke Ecke $(50, 50)$, obere-rechte Ecke $(450, 50)$, untere-linke Ecke $(50, 450)$ und untere-rechte Ecke $(450, 450)$. Der Abstand zwischen zwei benachbarten Gitterpunkten ist 1, also werden alle möglichen Integer Punkte im Gitter betrachtet.

Nun wird der Prozess in einem Iterationsschritt für den Punkt p_0 beschrieben. In einer ineinander geschachtelten For-Schleife, die äußere für die x -Achse, die innere für die y -Achse, werden alle Punkte im obigen Gitter betrachtet. Natürlich kommen nur solche Punkte des Gitters in Frage, die im Kreis liegen, es wird mit einer if-Bedingung überprüft. Bevor die Doppelschleife durchgeführt wird, wird der Abstand von p_0 zum nächsten Nachbarn berechnet und als alter Abstand gespeichert. Die alten Koordinaten von p_0 werden selbst auch gespeichert. In der Doppelschleife werden für jeden Gitterpunkt, der im Kreis liegt, der Abstand zum nächsten Nachbarn berechnet, und mit dem alten Abstand verglichen. Falls dieser Abstand größer als der alte Abstand ist, wird dieser Abstand als alter Abstand gespeichert und dieser Gitterpunkt vorübergehend als neuer Kandidat für p_0 gespeichert, und der nächste Gitterpunkt in der Doppelschleife betrachtet. Falls dieser Abstand kleiner oder gleich groß als der alte Abstand ist, passiert nichts und es wird der nächste Gitterpunkt betrachtet. Nachdem alle Gitterpunkte in der Doppelschleife geprüft wurden, speichert man für p_0 den zuletzt gespeicherten neuen Kandidat (mit dem größten Abstand zum nächsten Nachbarn). Falls es keine bessere Lösung in einem Iterationsschritt gibt, werden die alten Koordinaten von p_0 beibehalten. Im nächsten Iterationsschritt werden für den Punkt p_1 wieder alle Punkte im Gitter überprüft.

Implementierung mit Gitterpunkten auf der Kugeloberfläche: ohne Voronoi-Knoten-Test

Da die Implementierung des Algorithmus auf der Kugeloberfläche ähnlich ist wie im Kreis, werden wir die Unterschiede betrachten. Zwischen zwei Punkten auf der Kugeloberfläche wird der Euklidische Abstand $|\cdot|$ genommen, das heißt im Programm benutzt, und nicht der sphärische Abstand $|\cdot|_k$. Die Kugel hat in der Implementierung einen 0.6f langen Radius und die Punkte p_0, \dots, p_{n-1} werden als kleine Kugeln realisiert [12],[13]. Alle Koordinaten und Abstände werden in **Float** gerechnet bzw. angegeben. Ähnlich wie im Kreis werden bei der Implementierung des Algorithmus ein Gitter berechnet und dessen Punkte in jedem Iterationsschritt geprüft. Das Gitter (Netz) wird auf der Kugeloberfläche mit Polarkoordinaten angegeben. Mit $\alpha \in [0, \pi]$ und $\beta \in [0, 2\pi]$ und mit Radius r kann man die x, y, z -Koordinaten eines Punktes auf der Kugeloberfläche bestimmen:

$$x = r \cdot \sin \alpha \cdot \cos \beta, \quad y = r \cdot \sin \alpha \cdot \sin \beta, \quad z = r \cdot \cos \alpha$$

Jetzt wird im Pseudocode zu dieser Implementierung dargestellt, was in einem Iterationsschritt passiert. Die Implementierungen Im Fall des Kreises und der Kugeloberfläche sind hier zusammen. Steht der Pseudotext zwischen $k :: \dots :: k$, bezieht er sich nur auf den Kreis, steht er zwischen $s :: \dots :: s$, bezieht er sich nur auf die Kugeloberfläche, sonst auf beide. Wir haben n Punkte, und ii gibt den Index des Punktes an, der sich bewegt. Die Koordinaten der Punkte p_x, p_y, p_z sind während des Spieles in den Arrays $x[n], y[n], z[n]$ gespeichert. Vor der Berechnung der neuen Position des Punktes p_{ii} , speichern wir die alte Koordinaten von p_{ii} in $point_x_old, point_y_old$ und $point_z_old$. Der Zähler d gibt an, wieviele bessere Kandidaten als die alte Position gefunden werden. Mit der Methode eut_dist_nn wird der Euklidische Abstand eines Punktes zu seinem nächsten Nachbarn berechnet. Für den (alten) Punkt p_{ii} wird dieser Abstand in $dist_old$ gespeichert.

Implementierung mit Gitter, ohne Voronoi-Knoten-(Randpunkt-)Test:

```
d = 0; point_x_old = x[ii]; point_y_old = y[ii];
s :: point_z_old = z[ii]; :: s dist_old = eut_dist_nn(ii, x, y, s :: z :: s)
```

```
//doppelte For-Schleife für die Gitterpunkte mit Polarkoordinaten
s:: for ( int  $\alpha = 0$ ;  $\alpha < 101$ ;  $\alpha ++$  ){
    for ( int  $\beta = 0$ ;  $\beta < 200$ ;  $\beta ++$  ){
        x[ii] = r sin  $\frac{\pi\alpha}{100}$  cos  $\frac{\pi\beta}{100}$ ;
```

$$y[ii] = r \sin \frac{\pi\alpha}{100} \sin \frac{\pi\beta}{100};$$

$$z[ii] = r \cos \frac{\pi\alpha}{100}; \quad ::s$$

```

k :: for (int j = 50; j < 451; j ++){
  x[ii] = j;
  for (int k = 50; k < 451; k ++){
    y[ii] = k;

//ob der Punkt im Kreis liegt
    a = (x[ii] - 250)2 + (y[ii] - 250)2
    if ((a < 40000)|| (abs(a - 40000) < 200)) ::k

    dist_new = euc_dist_nn(ii, x, y, s :: z :: s);

//der neue Abstand zum nächsten Nachbarn wird mit dem alten Abstand
vergleichen
    if (dist_new > dist_old ) {
      d++;
      dist_old = dist_new;
      point_x_new = x[ii];
      point_y_new = y[ii];
      s :: point_z_new = z[ii]; :: s
    }
  }
}

//es gibt bessere Wahl für den neuen Punkt
if (d ≠ 0) {
  x[ii] = point_x_new;
  y[ii] = point_y_new;
  s :: z[ii] = point_z_new :: s;
}
else {
  x[ii] = point_x_old;
  y[ii] = point_y_old;
  s :: z[ii] = point_z_old; :: s
}

```

Implementierung mit Voronoi-Knoten und Randpunkte im Kreis: mit Voronoi-Knoten-Randpunkt-Test

Wir betrachten hier auch, was in einem Iterationsschritt passiert. Sei p_0 der bewegliche Punkt, und wir halten die restlichen Punkte p_1, \dots, p_n fest. Am Anfang wird der Abstand von p_0 zu seinem nächsten Nachbarn berechnet, und als alter Abstand gespeichert. Die alten Koordinaten von p_0 werden selbst auch gespeichert. Es ist eine naive Implementierung, das heißt es werden mit relativ hohem Aufwand nur die Voronoi-Knoten des Voronoi-Diagramms von $n - 1$ festen Punkten $P_r = \{p_1, \dots, p_n\}$ berechnet.

Hier wird wieder das Gitter wie in der vorherigen Implementierung benutzt. Für jeden Gitterpunkt wird ein Test durchgeführt, ob dieser Punkt ein Voronoi-Knoten von P_r ist. Die gefundenen Voronoi-Knoten werden in einem Array gespeichert. Nach Kapitel 5 wissen wir schon, dass für p_0 nicht nur die Voronoi-Knoten von P_r , sondern auch die Randpunkte des Kreises als neue Kandidaten in Frage kommen. Also werden anschließend die Randpunkte mit Polarkoordinaten berechnet und im gleichem Array gespeichert, wo die Voronoi-Knoten schon vorher gespeichert wurden.

In einer For-Schleife werden alle Einträge vom Array betrachtet. In jedem Rechenschritt der Schleife wird also der Abstand eines Voronoi-Knotens oder eines Randpunktes zum nächsten Nachbarn bestimmt und mit dem alten Abstand verglichen. Wir führen also denselben Vergleich der Abstände wie bei der gittermäßigen Implementierung durch. Am Ende speichert man für p_0 den letzten gespeicherten neuen Kandidat (mit dem größten Abstand zum nächsten Nachbarn). Falls es keine bessere Lösung gibt, werden die alten Koordinaten für p_0 beibehalten.

Implementierung mit Voronoi-Knoten auf der Kugeloberfläche: mit Voronoi-Knoten-Test

Auf der Kugeloberfläche reicht es auch, nur die Voronoi-Knoten in $V(P_r)$ zu betrachten ($P_r = \{p_1, \dots, p_{n-1}\}$ und p_0 ist der bewegliche Punkt). Hier wird wieder das mit Polarkoordinaten bestimmte Gitter (Netz) benutzt. Für jeden Gitterpunkt wird ein Test durchgeführt, ob dieser Punkt ein Voronoi-Knoten ist. Der Rest der Implementierung erfolgt wie im Kreis.

Jetzt wird im Pseudocode zu dieser Implementierung beschrieben, was in einem Iterationsschritt passiert. Die Implementierung im Fall des Kreises und der Kugeloberfläche sind hier wieder zusammen, wie bei dem vorherigen Pseudocode. Die Markierungen $k :: \dots :: k$ und $s :: \dots :: s$ haben dieselbe Bedeutung wie oben. Wir haben n Punkte, und ii gibt den Index des Punktes an, der sich bewegt. Die Koordinaten der Punkte p_x, p_y, p_z sind während des Spieles in den Arrays $x[n], y[n], z[n]$ gespeichert. Vor der Berechnung der neuen Position des Punktes p_{ii} , speichern wir die alten Koordinaten von p_{ii} in $point_x_old, point_y_old$

und $point_z_old$, und alle Punkte in einem Array `points[]` (Point2f/Point3f). Der Zähler d gibt an, wieviele bessere Kandidaten als die alte Position gefunden werden. Mit der Methode `euc_dist_nn` wird der Euklidische Abstand eines Punktes zum nächsten Nachbarn berechnet. Für den (alten) Punkt p_{ii} wird dieser Abstand in `dist_old` gespeichert. Mit der Methode `Vor_compute` werden hier die Voronoi-Knoten des Voronoi-Diagramms der festen Punkten berechnet und in einem Array (Point2f/Point3f) gespeichert. Mit der Methode `get_count_VKn` wird die Anzahl der gefundenen Voronoi-Knoten berechnet.

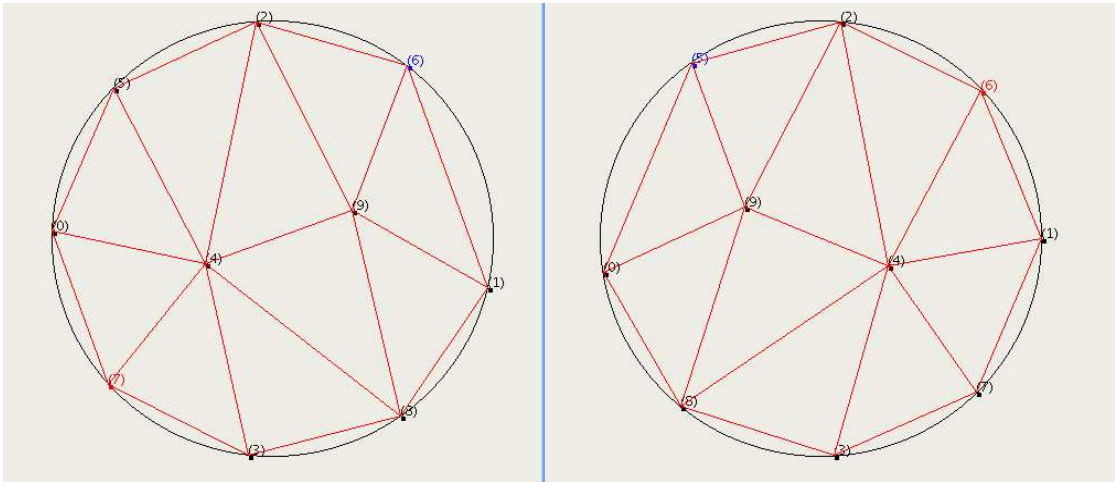


Abbildung 6.1: Dasselbe Ergebnis mit und ohne Voronoi

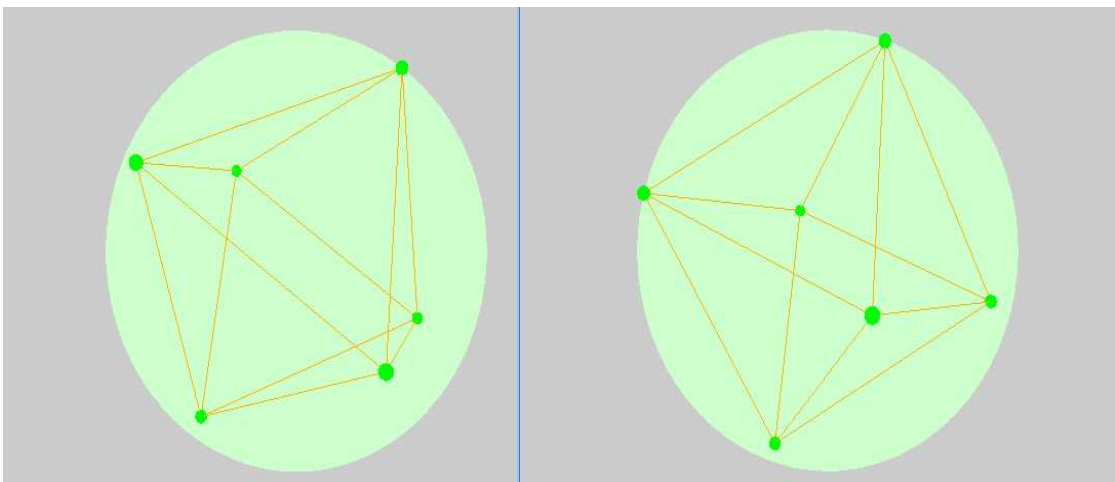


Abbildung 6.2: Dasselbe Ergebnis mit und ohne Voronoi

Implementierung mit Voronoi-Knoten-(Randpunkt-)Test:

```
s :: points[] ← new Point3f(x, y, z) :: s,
k :: points[] ← new Point2f(x, y) :: k,
d = 0; point_x_old = x[ii]; point_y_old = y[ii];
s :: point_z_old = z[ii]; :: s dist_old = euc_dist_nn(ii, x, y, s :: z :: s)

//die (n-1) festen Punkte sind in einem Array gespeichert
for (int k = 0; k < ii; k ++;) {
    points_vor[k] = points[k];
}
for (int k = ii; k < n - 1; k ++;) {
    points_vor[k] = points[k + 1];
}

//Berechnung der Voronoi-Knoten und ihre Anzahl
vor = new Voronoi3D(n - 1);
VKn = vor.Vor_compute(points_vor);
count_VKn = vor.get_count_VKn();

//im Kreis werden noch dazu die Randpunkte mit Polarkoordinaten berechnet
k :: start = count_VKn;
for (int b = start; b < start + 200; b ++;) {
    VKn.x[b] = 200 cos  $\frac{(b-start)\pi}{100}$  + 250;
    VKn.y[b] = -200 sin  $\frac{(b-start)\pi}{100}$  + 250;
} :: k

//Jetzt betrachten wir alle Voronoi-Knoten und Randpunkte
s:: for (int k = 0; k < count_VKn; k ++); ::s
k:: for (int k = 0; k < count_VKn + 200; k ++); ::k
x[ii] = VKn.x[k];
y[ii] = VKn.y[k];
s :: z[ii] = VKn.z[k]; :: s

dist_new = euc_dist_nn(ii, x, y, s :: z :: s);

//der neue Abstand zum nächsten Nachbarn wird mit dem alten Abstand
vergleichen
if (dist_new > dist_old) {
    d++;
    dist_old = dist_new;
    point_x_new = x[ii];
```

```

    point_y_new = y[ii];
    s :: point_z_new = z[ii]; :: s
  }
}

```

```

//Es gibt bessere Wahl für den neuen Punkt
if (d ≠ 0) {
  x[ii] = point_x_new;
  y[ii] = point_y_new;
  s :: z[ii] = point_z_new :: s;
}
else {
  x[ii] = point_x_old;
  y[ii] = point_y_old;
  s :: z[ii] = point_z_old; :: s
}

```

Liefere die beide Implementierungen im Kreis und auf der Kugeloberfläche genau dieselben Ergebnisse?

In den meisten Fällen bekommt man dieselben Ergebnisse, aber ab und zu sehen sie im Applet nur ähnlich aus, zum Beispiel eine Polyeder-Kante fehlt bei einer Version, was sich auf rechnerische Ungenauigkeit zurückführen lässt. Die gleichen Resultate bedeuten nicht unbedingt, dass die Koordinaten der Punkte identisch sind, sondern dass das Verhältnis der Punkte gleich ist, also mit Spiegelung oder Drehung der Punkte kann man die Resultate ineinander überführen, (siehe Abbildung 6.1, 6.2). In beiden Fällen liegen die gefundenen Messwerte des minimalen Abstandes nahe beieinander, (siehe die Tabelle 5.1-5.4).

6.2 Ermittlung der Voronoi-Knoten und Voronoi-Nachbarschaft

Als Nächstes geht es darum, wie man die Voronoi-Knoten des Voronoi-Diagramms einer Punktmenge $P_s = \{p_0, \dots, p_s\}$ im Kreis bzw. auf der Kugeloberfläche am einfachsten (aber mit leider hohem Aufwand) berechnen kann.

In einer Doppelschleife werden alle Gitterpunkte (Netz) für "Voronoi-Knoten"-Eigenschaft getestet. Betrachten wir jetzt einen Gitterpunkt p_g . Es werden alle s Abstände von p_g zu allen Punkten aus P_s berechnet und in einem Array $d[s]$ gespeichert.

Zur Erinnerung sei erwähnt, falls zwei Punkte aus P_s genauso weit von p_g liegen und die anderen Punkte aus P_s noch weiter von p_g liegen, liegt p_g auf einer

	d[0]	d[1]	d[2]	d[3]	d[4]	d[5]
d	3	2	3	2	4	6
k	1	1	1	2	2	2
m_0	0	1	1	1	1	1
m_1	-1	-1	-1	3	3	3
q	3	2	2	2	2	2

Tabelle 6.1: $k = 2$, der Punkt liegt auf einer Voronoi-Kante

	d[0]	d[1]	d[2]	d[3]	d[4]	d[5]
d	2	2	1	1	2	1
k	1	2	1	2	2	3
m_0	0	0	2	2	2	2
m_1	-1	1	-1	3	3	3
q	2	2	1	1	1	1

Tabelle 6.2: $k > 2$, der Punkt ist ein Voronoi-Knoten

Voronoi-Kante in $V(P_s)$. Falls drei Punkte aus P_s genauso weit von p_g liegen und die anderen Punkte noch weiter liegen, ist p_g ein Voronoi-Knoten in $V(P_s)$.

Unsere Aufgabe lautet also: Im Array d muss herausgefunden werden, ob der Punkt p_g auf einer Voronoi-Kante in $V(P_s)$ liegt oder ein Voronoi-Knoten in $V(P_s)$ ist oder keins davon. Dazu werden in einer For-Schleife alle Einträge von d nacheinander betrachtet. Vor dem Durchlauf der Schleife werden einige Hilfsvariablen eingeführt und initialisiert. Die erste Hilfsvariable ist am Anfang $k = 1$ und nach dem Durchlauf der Schleife gibt sie die Anzahl des Vorkommens des kleinsten Abstandes in Array d an. Die nächsten zwei Hilfsvariablen sind am Anfang $m_0 = 0$, $m_1 = -1$. Falls nach dem Durchlauf der Schleife $k = 2$ ist, haben m_0 und m_1 Bedeutung. Sie geben die Indizes von zwei Voronoi-banachbarten Punkten an. (Dieser Fall wird aber bei der Suche nach Voronoi-Knoten nicht benutzt, sondern bei der Ermittlung der Voronoi-Nachbarschaft, siehe unten bei der Delaunay-Zerlegung.) Wir brauchen noch eine Hilfsvariable, die am Anfang $q = d[0]$ ist. Nach dem Durchlauf der For-Schleife gibt sie den Wert von dem in d gespeicherten kleinsten Abstand an. Die Variable q kann nur gleich oder kleiner werden.

Während des Durchlaufs der For-Schleife werden sich die Hilfsvariablen folgendermaßen verändern. Falls der nächste zu betrachtende Eintrag $q[i]$ vom Array d gleich q ist, wird k inkrementiert. Falls noch dazu $m_1 = -1$ ist, bekommt m_1 den Wert i des Index des aktuellen Eintrags. Falls q , der bis jetzt gefundene kleinste Eintrag, größer als der nächste Eintrag $q[i]$ ist, das heißt wenn ein kleinerer Abstand gefunden wird, dann wird $q = q[i]$, $k = 1$, $m_0 = i$, $m_1 = -1$, also

werden die Hilfsvariablen wieder wie am Anfang initialisiert. Die Befunde von den vorherigen Einträgen sind dann irrelevant.

Es sind zwei Beispiele in Tabellen zu sehen. Falls $k = 2$ ist, liegt der Punkt p_g auf einer Voronoi-Kante in $V(P_s)$, (siehe Tabelle 6.1). Falls $k > 2$ ist, dann ist der Punkt p_g ein Voronoi-Knoten in $V(P_s)$, (siehe Tabelle 6.2).

Wie in Kapitel 2 schon gesehen, ist die Delaunay-Zerlegung der duale Graph des Voronoi-Diagramms. In der Delaunay-Zerlegung sind zwei Punkte von $P = \{p_0, \dots, p_{n-1}\}$ genau dann mit einer Kante verbunden, wenn sie in $V(P)$ eine gemeinsame Voronoi-Kante besitzen, das heißt wenn sie Voronoi-Nachbarn sind. Die Voronoi-Nachbarschaft kann man in einer $n \times n$ Matrix M speichern. Sind zwei Punkte p_0 und p_1 benachbart, speichert man die Information im Eintrag $M[0][1]$ (und im Eintrag $M[1][0]$, eigentlich redundant) und wird 1 zu diesem Eintrag zugewiesen.

Wie oben besprochen werden im jeden Rechenschritt der Doppelschleife die Abstände eines Gitterpunktes p_g zu allen Punkten aus $P = \{p_0, \dots, p_{n-1}\}$ berechnet, und in einem Array $d[n]$ gespeichert. Genauso wie oben dargestellt werden in einer For-Schleife die Hilfsvariablen berechnet. Falls für einen Gitterpunkt $k = 2$ ist, also dieser Gitterpunkt liegt auf einer Voronoi-Kante in $V(P)$, müssen die Hilfsvariablen m_0 und m_1 berücksichtigt werden. Diese Variablen zeigen, dass die Punkte p_{m_0} und p_{m_1} Voronoi-Nachbarn sind. Also wird der Eintrag $M[0][1] = 1$ gesetzt. (Am Anfang sind alle Einträge von der Matrix M mit Null initialisiert.)

Nach der Betrachtung aller Gitterpunkte sind in der Matrix M die Voronoi-Nachbarschaftsbeziehung gespeichert. Danach werden alle Punktpaare aus P im Kreis mit einer Linie (Delaunay-Kante) verbunden, deren Matrixeintrag den Wert 1 hat. Also mit diesem Verfahren kann man die Delaunay-Zerlegung von P zeichnen.

Es wird nun beschrieben, wie man **das Delaunay-Polyeder**, eigentlich das von n Punkten gebildete Polyeder auf der Kugeloberfläche gebildet, berechnen kann. Auf der Kugeloberfläche könnte man die sphärische Delaunay-Zerlegung zeichnen lassen, indem man die Voronoi-Nachbarschaftsbeziehung in einer Matrix M speichert (mit den Hilfsvariablen m_0 und m_1) und zwei benachbarte Punkte mit einem Großkreisbogen verbindet. Stattdessen werden zwei benachbarte Punkte nicht auf der Kugeloberfläche verbunden, sondern mit einer geraden Linie im Raum \mathbb{R}^3 . Die so entstandene Delaunay-Zerlegung ist ein Polyeder, (siehe Kapitel 2). Im Programm werden alle Punktpaare von $P = \{p_0, \dots, p_{n-1}\}$ mit einer Linie (Polyeder-Kante) verbunden, deren Matrixeintrag $M[m_0][m_1]$ den Wert 1 hat.

Wie man die Voronoi-Knoten findet und wie man die Voronoi-Nachbarschaft bestimmt

```
die Methode: ... s :: Point3f Vor_compute(Point3f[]points) ::s
die Methode: ... k :: Point2f Vor_compute(Point2f[]points) ::k
.....
// Die Gitterpunkte werden eingeteilt (Voronoi-Knoten, liegt auf einer Voronoi-
Kante, keins davon
s:: for (int  $\alpha = 0$ ;  $\alpha < 101$ ;  $\alpha ++$ ;) {
  for (int  $\beta = 0$ ;  $\beta < 200$ ;  $\beta ++$ ;) {
     $p.x = r \sin \frac{\pi\alpha}{100} \cos \frac{\pi\beta}{100}$ ;
     $p.y = r \sin \frac{\pi\alpha}{100} \sin \frac{\pi\beta}{100}$ ;
     $p.z = r \cos \frac{\pi\alpha}{100}$ ; :: s
  }
k:: for (int  $j = 50$ ;  $j < 451$ ;  $j ++$ ;) {
   $p.x = j$ ;
  for ( int  $k = 50$ ;  $k < 451$ ;  $k ++$ ;) {
     $p.y = k$ ;
     $a = (p.x - 250)^2 + (p.y - 250)^2$ ;
    if ( (  $a < 40000$ ) || ( $abs(a - 40000) < 40000$ ) ) { :: k
  }
}

// p ist der zu überprüfende Gitterpunkt, in points[] sind die Punkte gespeichert,
die das Voronoi-Diagramm  $V(P_s)$  bestimmen.
for (int  $l = 0$ ;  $l < s$ ;  $l ++$ ;) {
   $d[l] = p.distance(points[l])$ ;
}

// Ist dieser Gitterpunkt p ein Voronoi-Knoten, so wird er in einem Array
gespeichert.
if (Vor_node_test(d)) {
   $VKn[count_VKn].x = p.x$ ;
   $VKn[count_VKn].y = p.y$ ;
   $s :: VKn[count_VKn].z = p.z$ ;
}
}
}

//es wird ein Array mit allen gefundenen Voronoi-Knoten zurückgeliefert.
return VKn;
```

die Methode: ... boolean Vor_node_test(float[]d)

```
.....
//Initialisierung der Werte
```

```

q = d[0]; k = 1; m0 = 0; m1 = -1;

for (int i = 0; i < s; i ++; ) {

//der nächste Eintrag vom d ist gleich dem kleinsten vorherigen Wert
if ( abs(d[i] - q) ≤ ε ) {
    k ++;

//der Index des ersten bis jetzt gefundenen kleinsten Eintrags wurde in m0
und der Index des zweiten bis jetzt gefundenen kleinsten Eintrags wird jetzt in
m1 gespeichert.
    if (m1 == -1)
        m1 = i;
}

//der nächste Eintrag ist kleiner als der Vorherige
if ( ε < q - d[i] ) {
    k = 1;
    m0 = i; m1 = -1;
    q = d[i];
}
}

//der Gitterpunkt p liegt auf einer Voronoi-Kante von P_s
if (k == 2)
    matrix[m0][m - 1] = 1;

//Ist der Gitterpunkt p ein Voronoi-Knoten von P_s, wird true zurückgeliefert.
return (k > 2);

```

6.3 Implementierung einiger Körper

Die Koordinaten für zwei regulären und für ein halbreguläres Polyeder wurden von einer Webseite [18] genommen. Drei regulären Polyeder wurden als Prisma oder als Antiprisma implementiert. Einige Archimedische Körper wurden mit bestimmten Generierungsmethoden hergestellt[11]. Alle so oder so entstehenden Polyeder wurden noch so skaliert, dass sie auf der Kugeloberfläche liegen.

Die konvexen Polyeder P sind festgelegt durch das n -Tupel

$$P = [[p_{11}, p_{12}, p_{13}], \dots, [p_{n1}, p_{n2}, p_{n3}]]$$

ihren Eckpunktkoordinatentripel $p_k = [p_{k1}, p_{k2}, p_{k3}]$.

Die Seiten des Polyeders F_i mit den jeweiligen Eckpunktkoordinatentripeln $p_{i_1}, \dots, p_{i_{m_i}}$ werden durch die m_i -Tupel dieser Koordinatentripelindizes beschrieben als $F_i = [i_1, \dots, i_{m_i}]$, und die Oberfläche in der Form $Fl = [F_1, \dots, F_s]$.

Konstruktion der Prismen und Antiprismen

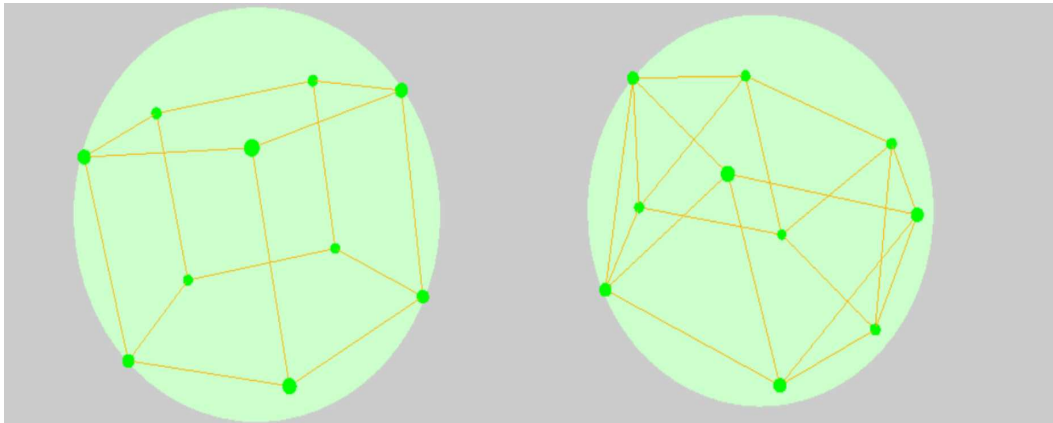


Abbildung 6.3: a, das Prisma mit 10 Punkten b, das Antiprisma mit 10 Punkten

Die Prismen (siehe Abbildung 6.3 links) sind spiegelsymmetrisch um die xy -Ebene. Die Eckpunkte des oberen $\frac{n}{2}$ regelmäßigen Vieleckes liegen auf einem Kreis mit Radius r der xy -Ebene mit Abstand $0 < z$ in der positiven Richtung entfernt. Sie sind durch:

$$\left\{ r \cdot \cos\left(4 \cdot k \cdot \frac{\pi}{n}\right), r \cdot \sin\left(4 \cdot k \cdot \frac{\pi}{n}\right), z \right\} \quad k \in \left\{0, \dots, \frac{n}{2} - 1\right\}$$

gegeben.

Die Eckpunkte des unteren $\frac{n}{2}$ regelmäßigen Vieleckes liegen auf einem Kreis mit Radius r der xy -Ebene mit Abstand z in der negativen Richtung entfernt. Sie sind durch:

$$\left\{ r \cdot \cos\left(4 \cdot k \cdot \frac{\pi}{n}\right), r \cdot \sin\left(4 \cdot k \cdot \frac{\pi}{n}\right), -z \right\} \quad k \in \left\{\frac{n}{2}, \dots, n - 1\right\}$$

gegeben, wobei R der Radius der Kugel bezeichne und

$$z = \sqrt{R^2 - r^2} \quad \text{und} \quad r = \frac{R}{\sqrt{1 + \sin^2\left(\frac{2\pi}{n}\right)}}$$

Die Antiprismen (siehe Abbildung 6.3 rechts) entstehen aus den Prismen, indem man das untere regelmäßige Vieleck noch mit dem Winkel $\frac{2\pi}{n}$ um die z -Achse dreht. Also sind die Antiprismen folgendermaßen zu beschreiben. Das obere Vieleck ist gedurch:

$$\left\{ r \cdot \cos\left(4 \cdot k \cdot \frac{\pi}{n}\right), r \cdot \sin\left(4 \cdot k \cdot \frac{\pi}{n}\right), z \right\} \quad k \in \left\{0, \dots, \frac{n}{2} - 1\right\}$$

gegeben, und das untere Vieleck ist durch:

$$\left\{ r \cdot \cos\left((2 \cdot k + 1) \cdot 2 \cdot \frac{\pi}{n}\right), r \cdot \sin\left((2 \cdot k + 1) \cdot 2 \cdot \frac{\pi}{n}\right), -z \right\} \quad k \in \left\{\frac{n}{2}, \dots, n - 1\right\}$$

gegeben, wobei R wieder der Radius der Kugel bezeichne und

$$z = \sqrt{R^2 - r^2} \quad \text{und} \quad r = \frac{2R}{\sqrt{3 \cdot \sin^2\left(\frac{2\pi}{n}\right) - \left(1 - \cos\left(\frac{2\pi}{n}\right)\right)^2 + 4}}$$

Das regelmäßige Quadrat und der Würfel wurden als Prismen, das Tetraeder und das Oktaeder wurden als Antiprismen implementiert.

Die Pyramide wurde auf folgender Weise implementiert:

$$p0_x = 0, \quad p0_y = 0, \quad p0_z = 1;$$

$$p1_x = 1, \quad p1_y = 0, \quad p1_z = 0;$$

$$p2_x = 0, \quad p2_y = 1, \quad p2_z = 0;$$

$$p3_x = -1, \quad p3_y = 0, \quad p3_z = 0;$$

$$p4_x = 0, \quad p4_y = -1, \quad p4_z = 0;$$

Die Eckpunktkoordinaten des Ikosaeders wurden folgendermaßen implementiert [18]: die halbe Seitenlänge des umschriebenen Würfels ist:

$$s = 0.6 / \sqrt{0.6 \cdot 0.6 + 0.37082 \cdot 0.37082}$$

Die Entfernung Ikosaeder-Ecke Würfel-Flächen-Mittelpunkt:

$$h = s \cdot \frac{1}{2} \cdot (\sqrt{5} - 1)$$

Die 12 Eckpunkte des Ikosaeders sind dann gegeben:

$$p0_x = -h, \quad p0_y = -s, \quad p0_z = 0; \quad p1_x = h, \quad p1_y = -s, \quad p1_z = 0;$$

$$\begin{aligned}
p2_x &= 0, & p2_y &= -h, & p2_z &= s; & p3_x &= 0, & p3_y &= -h, & p3_z &= -s; \\
p4_x &= s, & p4_y &= 0, & p4_z &= h; & p5_x &= s, & p5_y &= 0, & p5_z &= -h; \\
p6_x &= -s, & p6_y &= 0, & p6_z &= -h; & p7_x &= -s, & p7_y &= 0, & p7_z &= h; \\
p8_x &= 0, & p8_y &= h, & p8_z &= s; & p9_x &= 0, & p9_y &= h, & p9_z &= -s; \\
p10_x &= -h, & p10_y &= s, & p10_z &= 0; & p11_x &= h, & p11_y &= s, & p11_z &= 0;
\end{aligned}$$

Die Eckpunktkoordinaten des Dodekaeders wurden folgendermaßen implementiert [18]: die halbe Seitenlänge des eingeschriebenen Würfels ist: $s = 0.4$. Die Höhe einer Dodekaeder-Kante über dem Würfel:

$$s \cdot \frac{1}{2} \cdot (\sqrt{5} - 1)$$

Die 20 Eckpunkte des Dodekaeders sind dann gegeben:

$$\begin{aligned}
p0_x &= 0, & p0_y &= -h, & p0_z &= -(s + h); & p1_x &= 0, & p1_y &= h, & p1_z &= -(s + h); \\
p2_x &= s, & p2_y &= -s, & p2_z &= -s; & p3_x &= s, & p3_y &= s, & p3_z &= -s; \\
p4_x &= -s, & p4_y &= s, & p4_z &= -s; & p5_x &= -s, & p5_y &= -s, & p5_z &= -s; \\
p6_x &= s + h, & p6_y &= 0, & p6_z &= -h; & p7_x &= -(s + h), & p7_y &= 0, & p7_z &= -h; \\
p8_x &= h, & p8_y &= h + s, & p8_z &= 0; & p9_x &= -h, & p9_y &= h + s, & p9_z &= 0; \\
p10_x &= -h, & p10_y &= -(s + h), & p10_z &= 0; & p11_x &= h, & p11_y &= -(s + h), & p11_z &= 0; \\
p12_x &= s + h, & p12_y &= 0, & p12_z &= h; & p13_x &= -(s + h), & p13_y &= 0, & p13_z &= h; \\
p14_x &= s, & p14_y &= -s, & p14_z &= s; & p15_x &= s, & p15_y &= s, & p15_z &= s; \\
p16_x &= -s, & p16_y &= s, & p16_z &= s; & p17_x &= -s, & p17_y &= -s, & p17_z &= s; \\
p18_x &= 0, & p18_y &= -h, & p18_z &= s + h; & p19_x &= 0, & p19_y &= h, & p19_z &= s + h;
\end{aligned}$$

Die Eckpunktkoordinaten des abgestumpften Ikosaeders (Fußball) wurden so implementiert [18]: die halbe Seitenlänge des umschriebenen Würfels $s = 150$. Es wurden folgende Hilfsparameter benutzt:

$$\begin{aligned}
h &= s \cdot \frac{1}{2} \cdot (\sqrt{5} - 1), & h1 &= \frac{h}{3}, & h2 &= \frac{h \cdot 2}{3}, \\
s1 &= \frac{s}{3}, & s2 &= \frac{s \cdot 2}{3}, & k1 &= h + \frac{s - h}{3}, & k2 &= h + \frac{(s - h) \cdot 2}{3}
\end{aligned}$$

Die 60 Punkte des abgestumpften Ikosaeders sind dann gegeben:

$$p0_x = -s, \quad p0_y = 0, \quad p0_z = h1; \quad p1_x = -s, \quad p1_y = 0, \quad p1_z = -h1;$$

$$\begin{aligned}
& p2_x = -k2, \quad p2_y = s1, \quad p2_z = -h2; \quad p3_x = -k1, \quad p3_y = s2, \quad p3_z = -h1; \\
& p4_x = -k1, \quad p4_y = s2, \quad p4_z = h1; \quad p5_x = -k2, \quad p5_y = s1, \quad p5_z = h2; \\
& p6_x = -k2, \quad p6_y = -s1, \quad p6_z = -h2; \quad p7_x = -k1, \quad p7_y = -s2, \quad p7_z = -h1; \\
& p8_x = -k1, \quad p8_y = -s2, \quad p8_z = h1; \quad p9_x = -k2, \quad p9_y = -s1, \quad p9_z = h2; \\
& p10_x = -s2, \quad p10_y = -h1, \quad p10_z = k1; \quad p11_x = -s1, \quad p11_y = -h2, \quad p11_z = k2; \\
& p12_x = 0, \quad p12_y = -h1, \quad p12_z = s; \quad p13_x = 0, \quad p13_y = h1, \quad p13_z = s; \\
& p14_x = -s1, \quad p14_y = h2, \quad p14_z = k2; \quad p15_x = -s2, \quad p15_y = h1, \quad p15_z = k1; \\
& p16_x = -h1, \quad p16_y = k1, \quad p16_z = s2; \quad p17_x = -h2, \quad p17_y = k2, \quad p17_z = s1; \\
& p18_x = -h2, \quad p18_y = -k2, \quad p18_z = s1; \quad p19_x = -h1, \quad p19_y = -k1, \quad p19_z = s2; \\
& p20_x = s, \quad p20_y = 0, \quad p20_z = -h1; \quad p21_x = s, \quad p21_y = 0, \quad p21_z = h1; \\
& p22_x = k2, \quad p22_y = -s1, \quad p22_z = h2; \quad p23_x = k1, \quad p23_y = -s2, \quad p23_z = h1; \\
& p24_x = k1, \quad p24_y = -s2, \quad p24_z = -h1; \quad p25_x = k2, \quad p25_y = -s1, \quad p25_z = -h2; \\
& p26_x = k2, \quad p26_y = s1, \quad p26_z = h2; \quad p27_x = k1, \quad p27_y = s2, \quad p27_z = h1; \\
& p28_x = k1, \quad p28_y = s2, \quad p28_z = -h1; \quad p29_x = k2, \quad p29_y = s1, \quad p29_z = -h2; \\
& p30_x = s2, \quad p30_y = h1, \quad p30_z = -k1; \quad p31_x = s1, \quad p31_y = h2, \quad p31_z = -k2; \\
& p32_x = 0, \quad p32_y = h1, \quad p32_z = -s; \quad p33_x = 0, \quad p33_y = -h1, \quad p33_z = -s; \\
& p34_x = s1, \quad p34_y = -h2, \quad p34_z = -k2; \quad p35_x = s2, \quad p35_y = -h1, \quad p35_z = -k1; \\
& p36_x = h1, \quad p36_y = -k1, \quad p36_z = -s2; \quad p37_x = h2, \quad p37_y = -k2, \quad p37_z = -s1; \\
& p38_x = h2, \quad p38_y = k2, \quad p38_z = -s1; \quad p39_x = h1, \quad p39_y = k1, \quad p39_z = -s2; \\
& p40_x = -h1, \quad p40_y = s, \quad p40_z = 0; \quad p41_x = h1, \quad p41_y = s, \quad p41_z = 0; \\
& p42_x = h2, \quad p42_y = k2, \quad p42_z = s1; \quad p43_x = h1, \quad p43_y = k1, \quad p43_z = s2; \\
& p44_x = s1, \quad p44_y = h2, \quad p44_z = k2; \quad p45_x = s2, \quad p45_y = h1, \quad p45_z = k1; \\
& p46_x = s1, \quad p46_y = -h2, \quad p46_z = k2; \quad p47_x = s2, \quad p47_y = -h1, \quad p47_z = k1; \\
& p48_x = h1, \quad p48_y = -k1, \quad p48_z = s2; \quad p49_x = h2, \quad p49_y = -k2, \quad p49_z = s1; \\
& p50_x = h1, \quad p50_y = -s, \quad p50_z = 0; \quad p51_x = -h1, \quad p51_y = -s, \quad p51_z = 0; \\
& p52_x = -h2, \quad p52_y = -k2, \quad p52_z = -s1; \quad p53_x = -h1, \quad p53_y = -k1, \quad p53_z = -s2; \\
& p54_x = -s1, \quad p54_y = -h2, \quad p54_z = -k2; \quad p55_x = -s2, \quad p55_y = -h1, \quad p55_z = -k1; \\
& p56_x = -s1, \quad p56_y = h2, \quad p56_z = -k2; \quad p57_x = -s2, \quad p57_y = h1, \quad p57_z = -k1; \\
& p58_x = -h1, \quad p58_y = k1, \quad p58_z = -s2; \quad p59_x = -h2, \quad p59_y = k2, \quad p59_z = -s1;
\end{aligned}$$

Konstruktion durch Eckenschnitt [11]

Zur jeder Ecke p_i des Polyeders P soll das s -Tupel $K_i = [K_{i_1}, \dots, K_{i_s}]$ der von ihr ausgehenden und zur Ecke p_i^j führenden Kanten

$$K_{ij} = \{q \mid q = (1-t)p_i + tp_i^j, \quad 0 \leq t \leq 1\} \quad j \in \{1, \dots, s\}$$

betrachtet werden. Wählt man t mit $0 < t \leq \frac{1}{2}$ fest, so erhält man auf jeder Kante für $t < \frac{1}{2}$ zwei und für $t = \frac{1}{2}$ einen auf der Kante markierten Punkt, bei k Kanten also $n = 2k$ Punkte für $t < \frac{1}{2}$, und $n = k$ Punkte für $t = \frac{1}{2}$. Das System dieser Punkte definiert das Schnittpolyeder $P(t)$ zum Polyeder P .

Bezeichne T das Tetraeder, O das Oktaeder, C den Würfel (Cubus), I das Ikosaeder und D das Dodekaeder.

Für das gestumpftes Tetraeder (Tetraeder truncum) gilt dann $TT = T(\frac{1}{3})$. Da man das Tetraeder zum Beispiel wie folgt beschrieben kann:

$$P = [[0, 0, 3], [2\sqrt{2}, 0, -1], [-\sqrt{2}, \sqrt{6}, -1], [-\sqrt{6}, -\sqrt{2}, -1]]$$

bekommt man die Eckpunkte von TT mit $a = \sqrt{2}$ und $b = \sqrt{6}$:

$$P = [[2a, 0, 5], [-a, b, 5], [-a, -b, 5], [4a, 0, 1], [3a, -b, -3], [3a, b, -3],$$

$$[-2a, 2b, 1], [0, 2b, -3], [-3a, b, -3], [-2a, -2b, 1], [-3a, -b, -3], [0, -2b, -3]]$$

Es wurde bei der Implementierung des abgestumpften Tetraeders ausgenutzt.

Ausgehend vom Oktaeder erhält man das Kuboktaeder $CO = O(\frac{1}{2})$ und das abgestumpfte Oktaeder (Oktaeder truncum) $OT = O(\frac{1}{3})$.

Ausgehend vom Würfel erhält man den abgestumpften Würfel (Cubus truncus) $CT = C(\frac{1}{2+\sqrt{2}})$ und mit $CO = C(\frac{1}{2})$ nochmals das Kuboktaeder. Schneidet man das Kuboktaeder wiederum mit $t = \frac{1}{2}$, so entsteht ein ungefähres Rhombenkuboktaeder $RCO \approx CO(\frac{1}{2})$.

Mit dem Dodekaeder als Ausgangspolyeder findet man das Ikosidodekaeder $ID = D(\frac{1}{2})$ und das abgestumpfte Dodekaeder (Dodekaeder truncum) $DT = D(\frac{5-\sqrt{5}}{10})$.

Vom Ikosaeder ausgehend findet man das abgestumpfte Ikosaeder, das Fussball $IT = I(\frac{1}{3})$.

Konstruktion aus Parameterdarstellung [11]

Einige Polyeder lassen sich in einer Parameterform darstellen:

$$P = P(a, b, c) = [p_1(a, b, c), \dots, p_n(a, b, c)]$$

Eine Parameterdarstellung ist zum Beispiel durch: $P = P(a, b, c) =$

$$\begin{aligned} & [[a, c, b], [c, b, a], [a, -b, c], [c, a, -b], [-a, b, c], [-c, a, b], \\ & [b, c, -a], [-b, -a, c], [-c, -b, a], [c, -b, -a], [-a, -b, -c], [-b, c, a], \\ & [-a, -c, b], [-a, c, -b], [a, -c, -b], [-c, -a, -b], [-b, -c, -a], [-b, a, -c], \\ & [b, -a, -c], [-c, b, -a], [c, -a, b], [b, a, c], [a, b, -c], [b, -c, a]] \end{aligned}$$

gegeben.

Das abgestumpfte Oktaeder OT , das Kuboktaeder CO , der abgestumpfte Würfel CT , das Rhombenkuboktaeder RCO und der abgeschrägte Würfel (Cubus simus) CS kann man mit dieser Parameterdarstellung konstruieren, sie wurden auch so implementiert.

$$OT = P(0, 1, 2) \quad CO = P(0, 1, 1) \quad CT = P\left(\frac{1}{1 + \sqrt{2}}\right)$$

$$RCO = P(1, 1 + \sqrt{2}, 1) \quad CS = P(1, 1.83926753, 3.382975767)$$

Konstruktion durch Mittelbildung [11]

Mit der Beschreibung $P = [[p_{11}, p_{12}, p_{13}], \dots, [p_{n1}, p_{n2}, p_{n3}]]$ für die Ecken und $Fl = [F_1, \dots, F_s]$ mit $F_i = [i_1, \dots, i_{m_i}]$ für die Seitenpolygone des Polyeders ergibt sich die Eckendarstellung des dualen Mittelpolyeders $m(P)$ als

$$q = [[q_{11}, q_{12}, q_{13}], \dots, [q_{s1}, q_{s2}, q_{s3}]]$$

$$q_i = \frac{1}{m_i} \sum_{k=1}^{k=m_i} p_{i_k}$$

wobei die Summe über alle m_i Ecken der Fläche F_i zu nehmen ist und i von 1 bis s läuft.

Vom Kuboktaeder ausgehend findet man mit 14 Ecken den Pyramidenwürfel (Hexakistetraeder) $TH = m(CO)$. Wie vorher gesehen, kann man die Koordinaten des Kuboktaeders mit Parameterdarstellung angeben:

$$\begin{aligned}
p0_x = 1, \quad p0_y = 1, \quad p0_z = 0; \quad p1_x = 1, \quad p1_y = -1, \quad p1_z = 0; \\
p2_x = 1, \quad p2_y = 0, \quad p2_z = 1; \quad p3_x = 1, p3_y = 0, p3_z = -1; \\
p4_x = -1, \quad p4_y = 1, \quad p4_z = 0; \quad p5_x = 0, \quad p5_y = 1, \quad p5_z = 1; \\
p6_x = 0, \quad p6_y = 1, \quad p6_z = -1; \quad p7_x = -1, \quad p7_y = -1, \quad p7_z = 0; \\
p8_x = -1, \quad p8_y = 0, \quad p8_z = 1; \quad p9_x = -1, \quad p9_y = 0, \quad p9_z = -1; \\
p10_x = 0, \quad p10_y = -1, \quad p10_z = 1; \quad p11_x = 0, \quad p11_y = -1, \quad p11_z = -1;
\end{aligned}$$

Damit kann man die Seiten des Kuboktaeders erhalten:

$$\begin{aligned}
Fl = [[2, 4, 12], [1, 4, 7], [1, 3, 6], [2, 3, 11], [8, 10, 12], [5, 7, 10], [5, 6, 9], [8, 9, 11], \\
[5, 8, 9, 10], [1, 2, 3, 4], [4, 7, 10, 12], [1, 5, 6, 7], [3, 6, 9, 11], [2, 8, 11, 12]]
\end{aligned}$$

Erzeugung als Gruppeninvariante [11]

Man definiert mittels der erzeugenden Ausgangsmartizen mit Drehwinkel $\frac{2\pi}{5}$

$$A := \begin{pmatrix} \frac{1}{4}(\sqrt{5}-1) & -\frac{1}{4}\sqrt{2}\sqrt{5+\sqrt{5}} & 0 \\ \frac{1}{4}\sqrt{2}\sqrt{5+\sqrt{5}} & \frac{1}{4}(\sqrt{5}-1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$C := \begin{pmatrix} \frac{1}{2}\frac{3+\sqrt{5}}{5+\sqrt{5}} & -\frac{1}{2}\frac{\sqrt{2}}{\sqrt{5+\sqrt{5}}} & -\frac{2}{5}\frac{5-\sqrt{5}}{1-\sqrt{5}} \\ \frac{1}{2}\frac{\sqrt{2}\sqrt{5}}{\sqrt{5+\sqrt{5}}} & -\frac{1}{1-\sqrt{5}} & 0 \\ \frac{2}{5}\frac{\sqrt{5}}{1-\sqrt{5}} & \frac{\sqrt{2}}{\sqrt{5+\sqrt{5}}} & \frac{3+\sqrt{5}}{5+3\sqrt{5}} \end{pmatrix}$$

die Drehmatrizen $S = C^i A^j C^k A^l$ für $i, j, k, l \in \{0, \dots, 4\}$. Beschreibt man die Matrix S durch ihr Exponententupel $[i, j, k, l]$, so lässt sich die Gruppe in der Form schreiben:

$$\begin{aligned}
M = \{ & [[1, 1, 1, 1], [1, 1, 1, 2], [1, 1, 1, 3], [1, 1, 1, 4], [1, 1, 1, 0], [1, 1, 2, 1], \\
& [1, 1, 2, 2], [1, 1, 2, 3], [1, 1, 2, 4], [1, 1, 2, 0], [1, 1, 3, 1], [1, 1, 3, 2], \\
& [1, 1, 3, 3], [1, 1, 3, 4], [1, 1, 3, 0], [1, 1, 4, 1], [1, 1, 4, 2], [1, 1, 4, 3], \\
& [1, 1, 4, 4], [1, 1, 4, 0], [1, 1, 0, 1], [1, 1, 0, 2], [1, 1, 0, 3], [1, 1, 0, 4], \\
& [1, 1, 0, 0], [1, 2, 1, 1], [1, 2, 1, 2], [1, 2, 1, 3], [1, 2, 1, 4], [1, 2, 1, 0],
\end{aligned}$$

$[1, 2, 3, 1], [1, 2, 3, 2], [1, 2, 3, 3], [1, 2, 3, 4], [1, 2, 3, 0], [1, 2, 4, 1],$
 $[1, 2, 4, 2], [1, 2, 4, 3], [1, 2, 4, 4], [1, 2, 4, 0], [1, 3, 3, 1], [1, 3, 3, 2],$
 $[1, 3, 3, 3], [1, 3, 3, 4], [1, 3, 3, 0], [1, 3, 4, 1], [1, 3, 4, 2], [1, 3, 4, 3],$
 $[1, 3, 4, 4], [1, 3, 4, 0], [1, 4, 3, 1], [1, 4, 3, 2], [1, 4, 3, 3], [1, 4, 3, 4],$
 $[1, 4, 3, 0], [2, 3, 3, 1], [2, 3, 3, 2], [2, 3, 3, 3], [2, 3, 3, 4], [2, 3, 3, 0]]$.

Für jedes $u \in \mathbb{R}^3$ besitzt die Menge $M(u) = \{v \mid v = Su, S \in M\}$ höchstens 60 Tupel und ist invariant gegenüber der Drehgruppe M . $M(u)$ sei dabei wie bisher mit dem konvexen Polyeder $P = P(u) = \Upsilon(M(u))$ identifiziert. Man kann damit zum Beispiel das Ikosaeder $I = \Upsilon(M([0, 0, 1]))$, das Ikosidodekaeder $ID = \Upsilon(M([0, -1, 0]))$ berechnen, das letztere wurde auf dieser Weise implementiert.

Kapitel 7

Benutzung der Applets

Es werden hier zwei Gebrauchsanweisungen von zwei Applets, (ein für den Kreis, das andere für die Kugeloberfläche)gegeben. Um die Erklärung zu vereinfachen, wurden Schnappschüsse von ihnen fertiggestellt, (siehe Abbildung 7.1 und 7.2).

7.1 Java 2D Applet

In diesem Abschnitt geht es darum, wie man das Applet im Fall des Kreises benutzen sollte. Es wird eine ausführliche Gebrauchsanweisung gegeben.

Bevor man mit einem Spiel beginnt, kann man die Anzahl der Punkte einstellen. Die Knöpfe dafür haben die folgende Bedeutung: Die Knöpfe **1+** und **1-** inkrementieren bzw. dekrementieren die Anzahl der Punkte. Mit den Knöpfen **10+** und **10-** wird zehn addiert bzw. subtrahiert. Die aktuelle Anzahl wird in einem Textfeld angezeigt.

Jetzt kann man die Punkte im Kreis mit der Maus setzen, indem man zuerst auf den Knopf **Start** drückt. Hat man genauso viele Punkte gesetzt, wie im Textfeld angezeigt wird, kann man das Voronoi Spiel starten. Das Spiel kann mit einem der drei Knöpfe **Iter**, **Round**, **Stable** durchgeführt werden, sie unterscheiden sich darin, wieviele Iterationsschritte nacheinander ausgeführt werden. Mit dem Knopf **Iter** wird ein Iterationsschritt durchgeführt, das heißt bewegt sich nur ein Punkt. Mit dem Knopf **Round** werden nacheinander n Iterationsschritte ausgeführt, also bewegen sich alle n Punkte einmal. Mit dem Knopf **Stable** werden solange Iterationsschritte durchgeführt, bis es keine Bewegung in n nacheinanderfolgenden Iterationsschritten gibt, also führt es zum stabilen Zustand.

Defaultmäßig wird das Voronoi Spiel so ausgeführt, dass für die neue Position eines Punktes alle Gitterpunkte in einem Gitter geprüft werden. Im Appletfenster ist es unten auf dem zweiten Knopf mit der Beschriftung: **without Voronoi Nodes** zu sehen. Das Voronoi Spiel lässt sich auch so durchführen, indem man

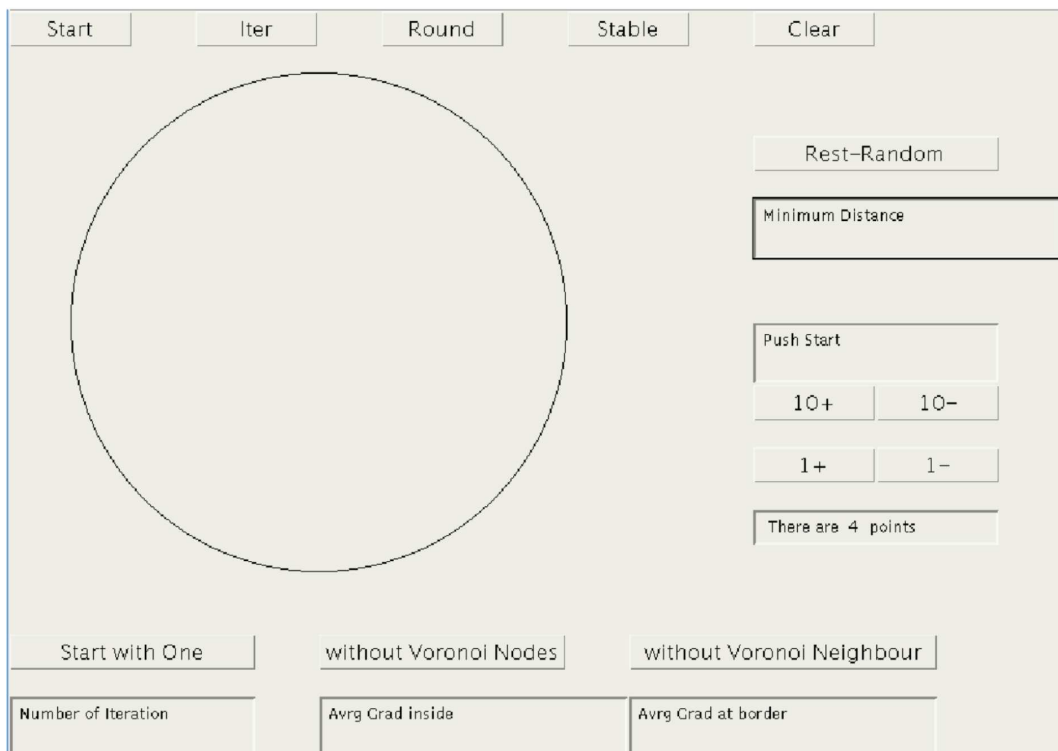


Abbildung 7.1: Java 2D Applet

bei einem Iterationsschritt nur Voronoi-Knoten bzw. Randpunkte betrachtet. Vor einem solchen neuen Spiel muss man den Knopf **Clear** drücken (nach oder vor mit diesem Knopfdruck kann man die Anzahl der Punkte umstellen). Danach muss man unten auf den zweiten Knopf drücken, und es erscheint der Text: **with Voronoi Nodes**. Jetzt kann man wieder auf den Knopf **Start** drücken und die Punkte setzen.

Möchte man nicht alle Punkte mit der Maus setzen, so gibt es die Möglichkeit mit dem Knopf **Rest-Random** die restliche fehlenden Punkten zufällig zu initialisieren.

Es gibt eine Variante des Spieles, wobei nur ein Punkt am Anfang irgendwo am Kreisrand mit der Maus gesetzt wird, und in der ersten Runde (n Iterationsschritte nacheinander) hängt die Position vom nächsten Punkt p_i von der Position von den Punkten p_0, \dots, p_{i-1} ab. Ab der zweiten Runde verhält sich das Spiel wie oben. Diese Variante kann mit dem Knopf **Start with One** gestartet werden.

Defaultmäßig wird die Delaunay-Zerlegung nicht angezeigt, was im Appletfenster

unten auf dem dritten Knopf mit der Beschriftung **without Voronoi Neighbour** zu sehen ist. Nach dem Spiel oder während des Spieles kann man die Delaunay-Zerlegung anschauen, indem man auf diesen Knopf drückt. Auf diesem Knopf erscheint der Text: **with Voronoi Neighbour**.

In den Textfeldern (unten und auf der rechten Seite im Appletfenster) kann man die folgenden Parameter beobachten: **Number of Iteration**: Anzahl der Iterationen, **Minimum Distance**: Der kleinste Abstand zwischen zwei beliebigen Punkten im Spielraum, **Avrg Grad inside**: Der durchschnittliche Eckengrad der Delaunay-Zerlegung im Inneren des Kreises und **Avrg Grad at border**: Der durchschnittliche Eckengrad der Delaunay-Zerlegung am Kreisrand.

Es müssen hier einige wichtige Regeln betont werden.

- Vor einem ganz neuen Spielstart muss der Knopf **Clear** gedrückt werden.
- Möchte man zwischen den Optionen **without Voronoi Nodes** oder **with Voronoi Nodes** wählen, muss man das auch vor dem Spielstart ändern. (Danach den Knopf **Start** oder **Start with One** drücken.)
- Möchte man das Spiel mit derselben Anfangskonfiguration starten, darf man keinen **Clear**, sondern noch einmal den Knopf **Start** drücken, (oder **Start with One**, falls ein solches Spiel vorher abgespielt wurde).

7.2 Java 3D Applet

In diesem Abschnitt geht es darum, wie man das Applet im Fall der Kugeloberfläche benutzen sollte. Es wird eine ausführliche Gebrauchsanweisung gegeben.

Bevor man mit einem Spiel beginnt, kann man die Anzahl der Punkte einstellen (genauso wie im Kreis). Alle nötigen Punkte werden zufällig auf die Kugeloberfläche gesetzt. Das Spiel kann mit einem der drei Knöpfe **Iter**, **Round**, **Stable** durchgeführt werden, sie haben dieselbe Bedeutung wie im Kreis.

Defaultmäßig wird das Spiel ohne Voronoi-Knoten Test durchgeführt. Im Appletfenster ist es oben in der dritten Reihe auf dem zweiten Knopf mit der Beschriftung **without Voronoi Nodes** zu sehen. Mit einem Druck auf diesen Knopf zu drücken, lässt sich das Spiel mit Voronoi-Knoten Test durchführen. Auf dem Knopf erscheint der Text: **with Voronoi Nodes**.

Mit dem Knopf **start again random** kann man ein ganz neues Spiel starten. Mit dem Knopf **start again the same** kann man das Spiel mit der vorherigen Anfangskonfiguration starten. Es gibt eine Variante des Spieles, wo alle Punkte am Anfang auf den Nordpol gesetzt werden, das heißt sie haben die Koordinaten

das reguläre Polyeder	Anzahl der Punkte	Knopf	Stabilität
Tetraeder	4	Antiprisma	stabil
Pyramide	5	Pyramide	stabil
Oktaeder	6	Antiprisma	stabil
Hexaeder (Würfel)	8	Prisma	stabil
Ikosaeder	12	Ikosaeder	stabil
Dodekaeder	20	Dodekaeder	stabil

Tabelle 7.1: Die 5 Platonischen Körper und die Pyramide

($0.0f, 0.0f, 0.6f$). Um mit dieser Version zu starten, muss man auf den Knopf **North Pole** drücken.

Defaultmäßig wird das (Delaunay-)Polyeder nicht angezeigt, was im Appletfenster oben in der dritten Reihe auf dem ersten Knopf mit der Beschriftung **without Polyeder** zu sehen ist. Nach dem Spiel oder während des Spieles kann man mit diesem Knopf das Polyeder ein- und ausschalten. Ist das Polyeder zu sehen, erscheint der Text auf dem Knopf **with Polyeder**.

Es sind einige reguläre und halbbreguläre Körper implementiert worden, unter anderem um ihre Stabilität zu testen. Wenn sich ein solches Polyeder nach der ersten Runde nicht bewegt, ist es stabil, (siehe Kapitel 4). Um mit einem von ihnen das Spiel zu starten, muss man im Appletfenster unten auf einen bestimmten Knopf drücken. In der Tabelle 7.1 sind die regulären Polyeder und die Pyramide zu finden. Beim Polyeder wie das Tetraeder, das Oktaeder und der Würfel muss die Anzahl vorher eingestellt werden, beim Ikosaeder, Dodekaeder und bei der Pyramide nicht, es wird mit dem Knopfdruck automatisch eingestellt.

Die folgenden halbbregulären Polyeder wurden implementiert: die Prismen und Antiprismen und einige Archimedische Körper, die in der Tabelle 7.2 zu sehen sind.

Bemerkung zu den Knöpfen **Prisma** und **Antiprisma**: Falls die Anzahl der Punkte ungerade ist, wird sie inkrementiert, und ein Prisma bzw. ein Antiprisma gezeichnet (ohne oder mit Polyeder Darstellung).

In den Textfeldern oben im Appletfenster kann man die folgenden Parameter beobachten: **Number of Iteration**: Anzahl der Iterationen, **Minimum Distance**: der Abstand der kleinsten Polyeder-Kante, **Maximum Distance**: der Abstand der größten Polyeder-Kante, und **Average Grad of Nodes**: Der durchschnittliche Eckengrad eines (Delaunay-)Polyeders.

das halbreguläre Polyeder	Anzahl der Punkte	Knopf	Stabilität
abgestumpftes Tetraeder	12	Tetraeder truncum	instabil
Kuboktaeder	12	Kuboktaeder	stabil
Pyramidenwürfel	14	Tetrakishexaeder	stabil
abgeschrägter Würfel	24	Cubus simus	stabil
Rhombenkuboktaeder	24	Rhombenkuboktaeder	stabil
abgestumpftes Würfel	24	Cubus truncus	instabil
abgestumpftes Oktaeder	24	Oktaeder truncum	instabil
Ikosidodekaeder	30	Ikosidodekaeder	stabil
Fussball - abgest. Ikosaeder	60	Ikosaeder truncum	stabil

Tabelle 7.2: Einige Archimedische Körper

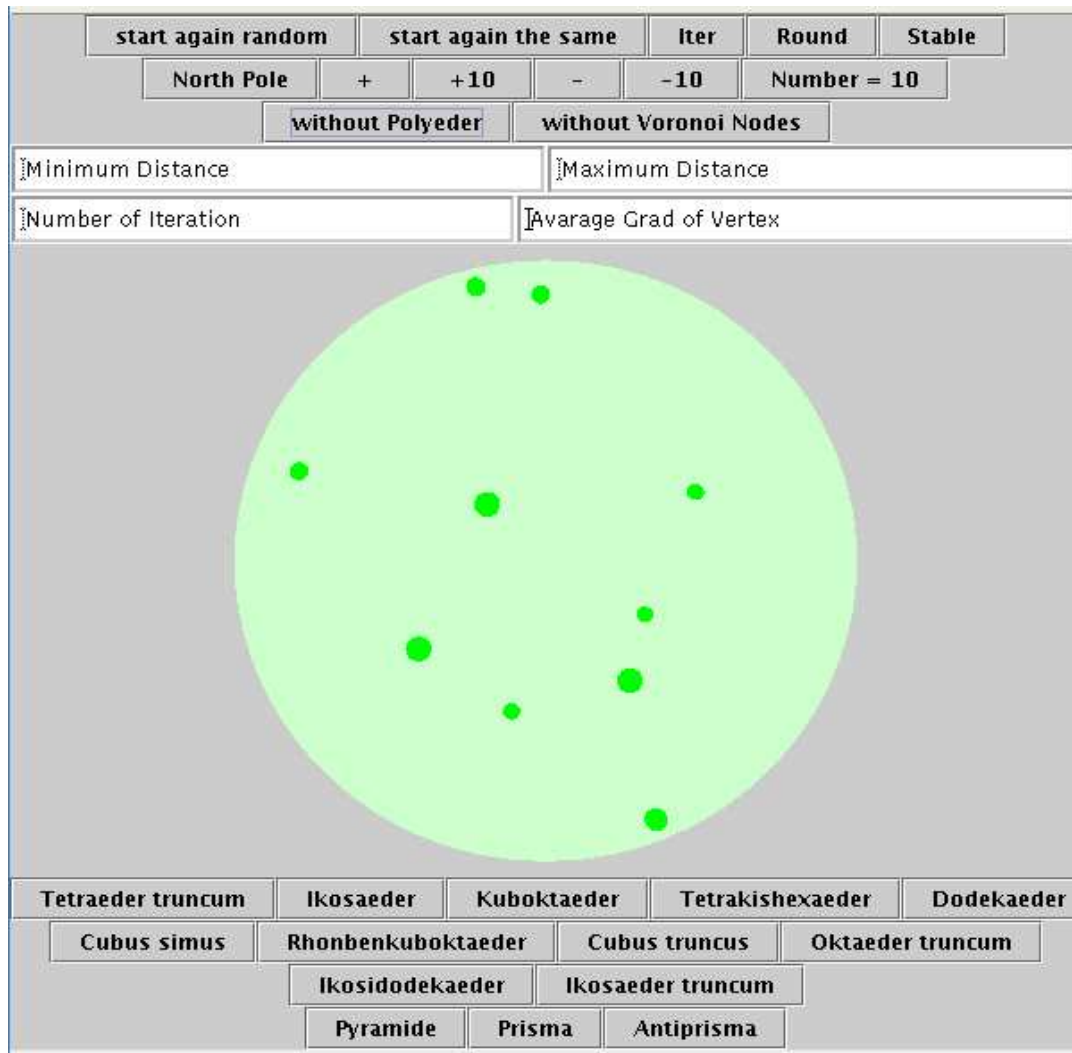


Abbildung 7.2: Java 3D Applet

Literaturverzeichnis

- [1] Andreas Filler, Euklidische und nicht Euklidische Geometrie, Wissenschaftsverlag, 1993
- [2] Rolf Klein, Algorithmische Geometrie, Bonn, Addison-Wesley-London, 1997
- [3] Konrad Königsberger, Analysis 1 Springer, 1999
- [4] Max Leppmeier, Kugelpackungen von Kepler bis heute, Braunschweig/Wiesbaden, 1997
- [5] Falko Lorenz, Lineare Algebra 1, Wissenschaftsverlag, 1992
- [6] Martin Nitschke Geometrie: Anwendungsbezogene Grundlagen und Beispiele, Fachbuchverlag Leipzig, 2005
- [7] <http://www.uni-koblenz.de/~cg/veranst/ws0304/voronoi.html>
- [8] <http://www.mathe.tu-freiberg.de/~wirth/Schueler/Koerper.pdf>
- [9] <http://www.mathe.tu-freiberg.de/~hebisch/cafe/archimedische.html>
- [10] <http://www.tfh-berlin.de/~baierl/inhalt/Hybride01/hybrid000201.pdf>
- [11] http://www.tfh-berlin.de/~baierl/inhalt/polyhed/p_konst000530.pdf
- [12] http://java3d.j3d.org/downloads/Java3D_schnell_tutorial.pdf
- [13] <http://java.sun.com/products/java-media/3D/collateral/index.html>
- [14] <http://www.ohg-sb.de/lehrer/bauer/bildpoly/Plato01.jpg>
- [15] <http://en.wikipedia.org/wiki/Polyhedron>

- [16] http://e-collection.ethbib.ethz.ch/ecol-pool/lehr/lehr_77_5.pdf
- [17] http://e-collection.ethbib.ethz.ch/ecol-pool/lehr/lehr_77_6.pdf
- [18] <http://www.jjam.de/Java/Applets/Applets.html>
- [19] <http://www-sop.inria.fr/geometrica/CGAL/Talks/Revmeet99/Revmeet99/PSfiles/voronoi.png>
- [20] <http://geometrie.diefenbach.at/Waben/Waben.11.gif>