Raytracing: Einfache Schnitttests

P. Hofmann, 22. August 2010
http://www.uninformativ.de

Einleitung

Tests, ob ein Strahl ein Objekt schneidet, sind die grundlegenden Operationen beim Raytracing. Sie entscheiden, was letztendlich auf dem Bildschirm zu sehen ist. Drei dieser Tests sollen hier nun vorgestellt werden:

- „Strahl vs. Kugel“
- „Strahl vs. Ebene“
- „Strahl vs. Dreieck“

Die Listings zeigen dabei gekürzten Programmcode aus dem NetTracer, einem in Java geschriebenen, netzwerkfähigen Raytracer:

http://www.uninformativ.de/projects/?q=nettracer

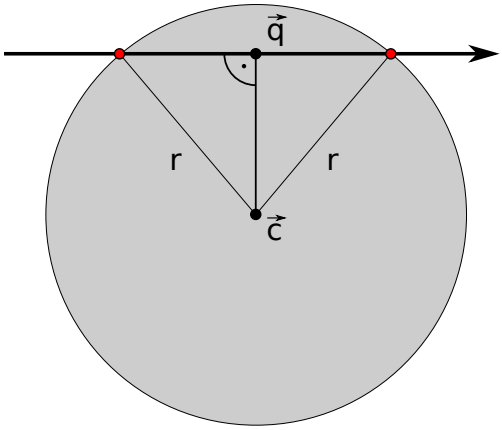
Im gesamten Dokument sei ein Strahl immer parametrisiert als:

$$\vec{q} = \vec{p} + \alpha \cdot \vec{u}$$

$\vec{p}$ ist der Ursprungspunkt des Strahls und $\vec{u}$ seine Richtung, dabei sei $\vec{u}$ immer normiert. In diese Richtung bewegt man sich α Schritte weit. Gewöhnlich sind also $\vec{p}$ und $\vec{u}$ fest, α wird meistens positiv sein, da man sich nach vorne bewegt, um die möglichen Punkte $\vec{q}$ auf dem Strahl zu erreichen.

1 Schnitttest mit einer Kugel

Eine Kugel sei gegeben über ihren Mittelpunkt $\vec{c}$ und den Radius r. Vorausgesetzt, der Strahl schneidet die Kugel, dann liegt folgende Situation vor:



Gesucht ist nun zuerst der Punkt $\vec{q}$: Dieser liegt auf dem Strahl und hat den geringsten Abstand zum Kugelmittelpunkt. Hat man $\vec{q}$ gefunden, dann kann man sich über den Satz des Pythagoras die beiden Schnittpunkte errechnen. Von diesen wird später gegebenenfalls ein geeigneter ausgewählt.

Der Punkt $\vec{q}$ muss nun die Bedingung erfüllen, dass er auf dem Strahl der nächste Punkt zum Kugelmittelpunkt ist. Das heißt, die Verbindungsstrecke von $\vec{c}$ nach $\vec{q}$ muss senkrecht zum Strahl sein, wie auch schon in der Skizze eingezeichnet. Außerdem kann $\vec{q}$ über den Strahl dargestellt werden als $\vec{q} = \vec{p} + \alpha \cdot \vec{u}$, wobei α gesucht ist. Formal ist dann:

$$(\vec{q} - \vec{c}) \cdot \vec{u} = 0$$

$$\Rightarrow \quad \left(\underbrace{\vec{p} + \alpha \cdot \vec{u}}_{\vec{q}} - \vec{c} \right) \cdot \vec{u} = 0$$

Nutzt man nun, dass $\vec{u} \cdot \vec{u} = 1$, da $\vec{u}$ normiert ist, dann lässt sich diese Gleichung in die folgende Form bringen und der Parameter α bestimmen:

$$\alpha = -(\vec{p} - \vec{c}) \cdot \vec{u}$$

Mittels dieses α gelangt man zum Punkt $\vec{q}$. Nun lässt sich bereits testen, ob der Strahl die Kugel überhaupt schneiden kann. Dazu wird einfach die Strecke von $\vec{c}$ nach $\vec{q}$ gemessen. Ist diese größer als der Kugelradius, gibt es keinen Schnitt. Der folgende Satz 1 fasst diesen ersten Schnitttest zusammen.

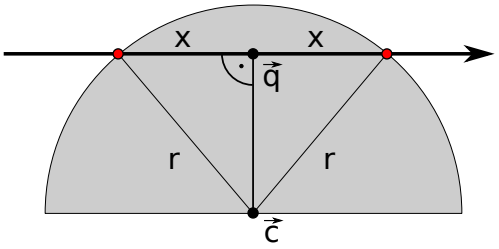
Satz 1 (*Bedingung für Schnitt mit Kugel*)

Ein Strahl $\vec{q} = \vec{p} + \alpha \cdot \vec{u}$ schneidet eine Kugel mit Mittelpunkt $\vec{c}$ und Radius r genau dann, wenn $|\vec{q} - \vec{c}| \leq r$, wobei $\alpha = -(\vec{p} - \vec{c}) \cdot \vec{u}$.

Sei nun ein gültiges $\vec{q}$ gegeben, sodass der Strahl die Kugel schneidet. Bisher ist aber noch unklar, *wo* dieser Schnitt stattfindet – dafür ist noch keine Bedingung gegeben, der Schnittpunkt kann also auch in negativer Richtung „vor“ dem Strahlursprung liegen, was ungewünscht ist. Diese Situation muss beim Raytracing als

„kein Schnitt“ gewertet werden, obwohl der Strahl als Gerade aufgefasst die Kugel tatsächlich schneidet. Ferner ist bei zwei Schnittpunkten noch unbekannt, welcher gewählt werden soll.

Um über den Pythagoras zu den beiden Schnittpunkten zu kommen, sei noch die skalare Größe x eingeführt:



x ist noch unbekannt. Es ist der Abstand von $\vec{q}$ zu einem der beiden Schnittpunkte und lässt sich über eines der beiden Dreiecke bestimmen, also:

$$r^2 = x^2 + |\vec{q} - \vec{c}|^2$$

$$\Rightarrow \quad x = \sqrt{r^2 - |\vec{q} - \vec{c}|^2}$$

Geht man nun davon aus, dass der Strahl außerhalb der Kugel startet und diese in positiver Strahlrichtung schneidet, dann ist klar, dass $\vec{h} = \vec{p} + (\alpha - x) \cdot \vec{u}$ der gewünschte Schnittpunkt ist. Man geht also zuerst zu $\vec{q}$ und dann wieder um x zurück zum Schnittpunkt.

Es kann aber vorkommen, dass ein Strahl *innerhalb* der Kugel startet. Das ist vorallem dann der Fall, wenn man Transmission-Rays durch lichtdurchlässige Materialien verfolgt. Hier ist dann der Austrittspunkt aus der Kugel gesucht, das heißt, man geht erst zu $\vec{q}$, aber dann in positiver Strahlrichtung um x weiter. Der Schnittpunkt ist dann gegeben als $\vec{h} = \vec{p} + (\alpha + x) \cdot \vec{u}$.

Diese beiden Fälle gilt es als mögliche Schnittszenarien zu unterscheiden. Der dritte mögliche Fall, wenn nämlich der Strahl „hinter“ der Kugel startet, also bereits wieder außerhalb, führt dazu, dass es nur in negativer Strahlrichtung Schnittpunkte gibt. Diese interessieren dann nicht weiter und die Situation wird als „kein Schnitt“ gewertet.

Satz 2 stellt nun formale Bedingungen an α und x, die zur Wahl des richtigen Schnittpunktes genutzt werden.

Satz 2 (*Auswahl des Kugelschnittpunktes*)

Gegeben sei α zum Punkt $\vec{q}$, der dem Kugelmittelpunkt am nächsten ist, außerdem der Variationsparameter x. $\vec{q}$ erfülle die Bedingung von Satz 1 auf Seite 2. Dann ist der gesuchte Schnittpunkt $\vec{h}$:

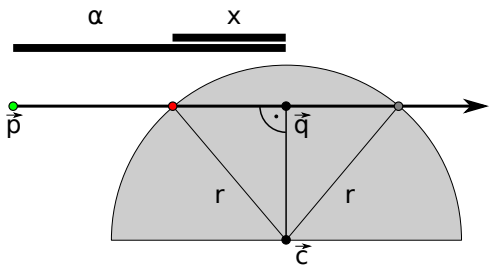
a) *Gilt $\alpha \geq x$, dann ist $\vec{h} = \vec{p} + (\alpha - x) \cdot \vec{u}$.*

b) *Ist das nicht der Fall, aber $\alpha + x > 0$, dann ist $\vec{h} = \vec{p} + (\alpha + x) \cdot \vec{u}$.*

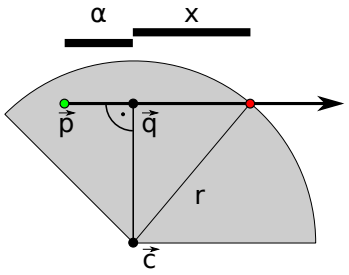
c) *Trifft keine dieser Bedingungen zu, dann existiert kein Schnittpunkt in positiver Strahlrichtung.*

Um sich diese Bedingungen richtig klar zu machen, beginnt man am besten mit dem ersten Fall: Der Strahl startet links außerhalb der Kugel. Jetzt ist α größer als x, der Weg zu $\vec{q}$ also länger als der Weg zurück

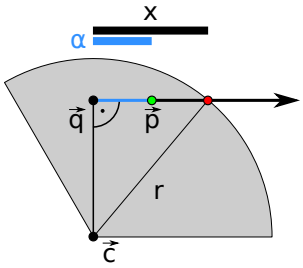
zum Schnittpunkt. α ist zu diesem Zeitpunkt auch positiv, es trifft also zu, dass $\alpha \geq x$. Nähert sich der Anfangspunkt $\vec{p}$ des Strahls immer weiter der Kugel an, dann wird α immer kleiner, bis es direkt an der Kugeloberfläche genau so groß wie x ist:



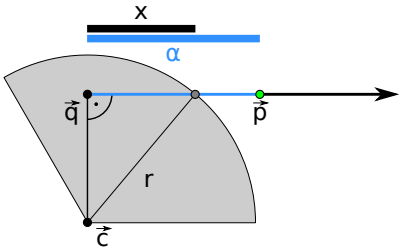
Nachdem $\vec{p}$ das Kugelinnere betreten hat, ist α kleiner als x geworden. Addiert man beide, ist die Summe aber immernoch größer als Null – das muss so sein, beide sind positiv. Es gilt nun, dass $\alpha + x > 0$ ist. Ab jetzt wird also der rechte Schnittpunkt ausgewählt:



Das ändert sich auch nicht, wenn α vorübergehend negativ wird. Dann muss man erst ein Stück „zurück“ laufen, bis man $\vec{q}$ erreicht hat, um dann über die Strecke x zum Schnittpunkt zu gelangen:



Erst, wenn $\vec{p}$ die Kugel wieder verlassen hat, ist α betragsmäßig großer als x und negativ. Dadurch ist deren Summe nicht mehr größer als Null, es gibt also keinen Schnittpunkt in positiver Strahlrichtung:



Das folgende Listing 1 ist ein leicht angepasster Auszug aus dem NetTracer-Code und zeigt, wie der Schnitttest mit einer Kugel programmatisch umgesetzt werden könnte.

Listing 1: Eine Kugel testet sich auf Schnittpunkte mit einem Strahl.

```
1  public class Sphere3D
2  {
3    private Vec3 origin;
4    private double radius;
5
6    public Vec3 intersectionTest(Ray r)
7    {
8      // Wo ist der Punkt auf dem Strahl, der am nächsten an mir
9      // liegt?
10     double alpha = -r.direction.dot(r.origin.minus(this.origin));
11     Vec3 q = r.evaluate(alpha);
12
13     // Abstand zum Kugelmittelpunkt?
14     q.subtract(this.origin);
15     double distToCenter2 = q.lengthSquared();
16
17     if (distToCenter2 > this.radius2)
18       return null;
19
20     // Über Pythagoras zu den beiden Schnittpunkten.
21     double x = Math.sqrt(this.radius2 - distToCenter2);
22
23     // Welcher von beiden liegt näher am Ray-Ursprung und in
24     // positiver Strahlrichtung?
25     double dist = 0.0;
26     if (alpha >= x)
27     {
28       dist = alpha - x;
29     }
30     else if (alpha + x > 0)
31     {
32       dist = alpha + x;
33     }
34     else
35     {
36       return null;
37     }
38
39     // Das ist unser finaler Schnittpunkt:
40     q = r.evaluate(dist);
41     return q;
42   }
43 }
```

2 Schnitttest mit einer Ebene

Eine Ebene sei gegeben als die Menge aller Punkte $\vec{x}$, für die gilt:

$$(\vec{x} - \vec{a}) \cdot \vec{n} = 0$$

Dabei ist $\vec{a}$ ein beliebiger Punkt der Ebene, den man als Aufpunkt auffassen kann. $\vec{n}$ ist der Normalenvektor der Ebene. Anschaulich kann man sich dieses Konstrukt so vorstellen: Man startet am Aufpunkt $\vec{a}$. Wählt man nun einen beliebigen Punkt $\vec{x}$ in der Ebene, dann ist $\vec{x} - \vec{a}$ der Verbindungsvektor von $\vec{a}$ zu diesem Punkt. Dieser Verbindungsvektor steht immer senkrecht auf $\vec{n}$, das Skalarprodukt muss also 0 sein.

Da $\vec{n}$ und $\vec{a}$ für eine Ebene fest sind, lässt sich obige Gleichung noch vereinfachen:

$$(\vec{x} - \vec{a}) \cdot \vec{n} = 0$$

$$\iff \quad \vec{x} \cdot \vec{n} = d \ , \quad \text{dabei sei } d = \vec{n} \cdot \vec{a}$$

Man muss also nur das Skalarprodukt eines möglichen Punktes der Ebene mit dem Normalenvektor derselben betrachten und kann alleine damit feststellen, ob der Punkt in der Ebene liegt. Nachdem man seine Ebene erst einmal definiert hat, spielt der Aufpunkt $\vec{a}$ keine praktische Rolle mehr, es reichen $\vec{n}$ und d, um die Ebene zu charakterisieren.

Stellt man nun $\vec{x}$ über alle möglichen Punkte des verfolgten Strahls dar, lässt sich wieder ein α als Strahlparameter bestimmen, welches angibt, ob und wo der Strahl die Ebene schneidet:

$$d = \underbrace{(\vec{p} + \alpha \cdot \vec{u})}_{\vec{x}} \cdot \vec{n}$$

$$\iff \quad \alpha = \frac{d - \vec{p} \cdot \vec{n}}{\vec{u} \cdot \vec{n}}$$

Offensichtlich muss für einen gültigen Schnittpunkt in positiver Strahlrichtung $\alpha > 0$ sein. Ist $\vec{u} \cdot \vec{n} = 0$, dann handelt es sich um einen der beiden Sonderfälle „Strahl liegt in der Ebene“ oder „Strahl ist parallel zur Ebene“. Beide werden in der Praxis als „kein Schnitt“ gewertet

Liegt der Strahl in der Ebene, gäbe es eigentlich unendlich viele Schnittpunkte. Es ist dennoch angebracht, dies als „kein Schnitt“ zu werten: Damit ein Strahl in einer Ebene liegt, muss man direkt parallel zur Ebene blicken. Würde man hier Schnittpunkte auswerten, sähe man die Ebene und sie bekäme eine Dicke. Ebenen sind aber unendlich dünn, also wäre dies keine Ebene mehr sondern ein Quader. Daher ergibt es Sinn, die Ebene in diesem Sonderfall zu ignorieren.

Der folgende Satz 3 formalisiert die Schnittbedingung.

Satz 3 (*Bedingung für Schnitt zwischen Strahl und Ebene*)

Ein Strahl $\vec{q} = \vec{p} + \alpha \cdot \vec{u}$ schneidet eine Ebene, gegeben durch $\vec{n}$ und d, genau dann in positiver Strahlrichtung im Punkt $\vec{q}$, wenn gilt:

$$\alpha = \frac{d - \vec{p} \cdot \vec{n}}{\vec{u} \cdot \vec{n}} > 0 \quad \textit{für } \vec{u} \cdot \vec{n} \neq 0$$

Auf ein Codebeispiel wird an dieser Stelle verzichtet, da dieses Thema im nächsten Abschnitt noch einmal aufgegriffen wird. Dort gibt es dann ein explizites Beispiel.

3 Schnitttest mit einem Dreieck

Ein Dreieck wird gewöhnlich über drei Punkte $\vec{A}$, $\vec{B}$ und $\vec{C}$ beschrieben. Diese Punkte spannen eine Ebene auf. Damit ein Strahl das Dreieck schneiden kann, muss er auch zwangsläufig die Ebene schneiden, in welcher das Dreieck liegt.

Um das herauszufinden, kann man die Erkenntnisse des Schnitttests mit einer Ebene nutzen, also Satz 3 von Seite 6. Dazu müssen natürlich vorher die Ebenenparameter bestimmt werden – siehe hierfür Satz 4. Die Bezeichner, die dort eingeführt werden, werden auch im Folgenden benutzt.

Satz 4 (*Ebenenparameter aus drei Punkten*)

Gegeben seien drei Punkte $\vec{A}$, $\vec{B}$ und $\vec{C}$. Ermittle die Parameter der von diesen drei Punkten aufgespannten Ebene wie folgt:

- *Wähle $\vec{A}$ als Aufpunkt der Ebene.*
- *Die beiden Kantenvektoren seien $\vec{b} = \vec{B} - \vec{A}$ und $\vec{c} = \vec{C} - \vec{A}$. Über deren Kreuzprodukt ist der Normalenvektor der Ebene bestimmt: $\vec{n} = \vec{b} \times \vec{c}$, danach normiere $\vec{n}$ auf die Länge 1.*
- *Folglich ist $d = \vec{n} \cdot \vec{A}$.*

Die Ebene ist also bekannt und darüber auch ein eventueller Schnittpunkt. Wie schon erwähnt, gilt Satz 5:

Satz 5 (*Strahl muss Ebene eines Dreiecks schneiden*)

Schneidet ein Strahl die Ebene, die von den drei Punkten eines Dreiecks aufgespannt wird, nicht oder nicht in positiver Strahlrichtung, so schneidet er auch das Dreieck nicht.

Damit ist also eine erste Abbruchbedingung für den Schnitttest gegeben. Im Folgenden sei diese Bedingung erfüllt und der Schnittpunkt $\vec{q}$ mit der Ebene gegeben. Es stellt sich nun die Frage, ob $\vec{q}$ innerhalb des Dreiecks liegt.

Da Dreiecke für gewöhnlich texturiert dargestellt werden sollen und oft auch eine Normale über das Dreieck hinweg interpoliert werden muss, bietet es sich an, den Punkt $\vec{q}$ in baryzentrischen Koordinaten bezüglich des Dreieckspunktes $\vec{A}$ anzugeben. Damit fällt ein späteres Texturieren leicht.

Die baryzentrischen Koordinaten gleichen in gewissem Sinne der Parameterdarstellung einer Ebene. Man startet am Punkt $\vec{A}$ und bewegt sich dann ein Stück in Richtung der Kante $\vec{b}$, wie sie in Satz 4 eingeführt wurde. Danach bewegt man sich ein Stück in Richtung der Kante $\vec{c}$. Auf diese Weise kann man genau wie bei der Parameterdarstellung einer Ebene jeden Punkt des Dreiecks erreichen. Formal sieht das so aus:

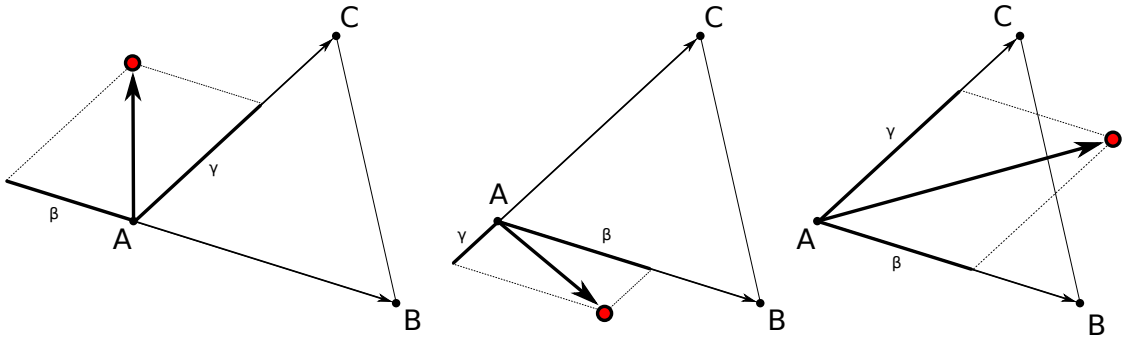
$$\vec{q} = \vec{A} + \beta \cdot \underbrace{(\vec{B} - \vec{A})}_{\vec{b}} + \gamma \cdot \underbrace{(\vec{C} - \vec{A})}_{\vec{c}}$$

β und γ geben also an, wie weit man sich in Richtung der jeweiligen Kanten bewegt. Wählt man sie jedoch „ungünstig“, dann kann man das Dreieck auch verlassen. Ist einer der beiden Parameter Null, dann bewegt man sich nur entlang einer Kante des Dreiecks. Man darf also schon einmal nicht in negativer Richtung laufen, so würde man das Dreieck sofort verlassen. Auch in positiver Richtung darf man nicht zu weit gehen, denn nach einer vollen Kantenlänge hat man das Dreieck auch wieder verlassen. Bewegt man sich zusätzlich auch entlang der zweiten Kante, dann sieht man, dass β und γ zusammen eine weitere Bedingung stellen, denn ihre Summe darf nicht größer als Eins sein. Dies kann man leicht sehen, wenn man sich gedanklich

in den Punkt $\vec{A}$ begibt und den Maximalfall $\beta + \gamma = 1$ annimmt. Dann lässt sich der Punkt $\vec{q}\,'$ nämlich schreiben als:

$$\begin{aligned}\vec{q}\,' &= \beta \cdot \vec{b} + \gamma \cdot \vec{c} \\ &= \beta \cdot \vec{b} + (1-\beta) \cdot \vec{c}\,, \quad \text{da } \beta + \gamma = 1\end{aligned}$$

Dies ist also eine lineare Interpolation zwischen $\vec{b}$ und $\vec{c}$, welche genau entlang der dritten Kante des Dreiecks verläuft. Die drei ungültigen Fälle sind also:



Formal ergeben sich an die beiden Parameter β und γ folgende Bedingungen:

Satz 6 (*Bedingung für Punkt innerhalb des Dreiecks*)

Ein in baryzentrischen Koordinaten dargestellter Punkt $\vec{q} = \vec{A} + \beta \cdot (\vec{B} - \vec{A}) + \gamma \cdot (\vec{C} - \vec{A})$ liegt genau dann im Dreieck der Punkte $\vec{A}$, $\vec{B}$ und $\vec{C}$, wenn gilt:

- $\beta \geq 0$
- $\gamma \geq 0$
- $\alpha = 1 - \beta - \gamma \geq 0$

Wenn man die baryzentrischen Koordinaten eines Punktes bezüglich eines Dreieckspunktes kennt, kann man die Frage nach einem Schnitt mit dem Strahl also beantworten: Suche einen Schnittpunkt mit der Ebene des Dreiecks und stelle diesen Schnittpunkt in baryzentrischen Koordinaten dar. Genügen diese Satz 6, dann schneidet der Strahl das Dreieck.

Bleibt die Frage zu beantworten, wie man die baryzentrischen Koordinaten findet. Da Raytracing ein Thema ist, über das sich schon viele Menschen intensiv Gedanken gemacht haben, gibt es hierfür die verschiedensten Methoden. Im Folgenden soll eine davon vorgestellt werden, die einfach zu verstehen ist, ohne Betrachtung von Sonderfällen auskommt und sich dennoch gut optimieren lässt.

Betrachte die Ausgangsgleichung:

$$\vec{q} = \vec{A} + \beta \cdot \underbrace{(\vec{B} - \vec{A})}_{\vec{b}} + \gamma \cdot \underbrace{(\vec{C} - \vec{A})}_{\vec{c}}$$

Durch die Vektoren handelt es sich hier eigentlich um drei Gleichungen und die zwei Unbekannten β und γ. Nimmt man folgende Umformung vor, indem man $\vec{A}$ von beiden Seiten subtrahiert, dann beschreiben

beide Seiten der Gleichung einen Vektor vom Punkt $\vec{A}$ zum Punkt $\vec{q}$ – dieser wird zur Vereinfachung $\vec{a}$ genannt:

$$\underbrace{\vec{q} - \vec{A}}_{\vec{a}} = \beta \cdot \vec{b} + \gamma \cdot \vec{c}$$

Dieses $\vec{a}$ projiziert man nun durch Bilden des Skalarproduktes einmal auf die Kante $\vec{b}$ und einmal auf die Kante $\vec{c}$, wodurch nur noch zwei Gleichungen und zwei Unbekannte übrig bleiben:

$$\vec{a} \cdot \vec{b} = \beta \cdot \vec{b} \cdot \vec{b} + \gamma \cdot \vec{c} \cdot \vec{b} \tag{I}$$

$$\vec{a} \cdot \vec{c} = \beta \cdot \vec{b} \cdot \vec{c} + \gamma \cdot \vec{c} \cdot \vec{c} \tag{II}$$

Löse (I) nach β auf:

$$\beta = \frac{\vec{a} \cdot \vec{b} - \gamma \cdot \vec{c} \cdot \vec{b}}{\vec{b} \cdot \vec{b}} \tag{III}$$

Da keine degenerierten Dreiecke gerendert werden, ist immer $\vec{b} \neq \vec{0}$. Setze nun (III) in (II) ein, dann ergibt sich nach Vereinfachung:

$$\gamma = \frac{\vec{a} \cdot \vec{c} \;\cdot\; \vec{b} \cdot \vec{b} \;-\; \vec{a} \cdot \vec{b} \;\cdot\; \vec{b} \cdot \vec{c}}{\vec{c} \cdot \vec{c} \;\cdot\; \vec{b} \cdot \vec{b} \;-\; \vec{c} \cdot \vec{b} \;\cdot\; \vec{b} \cdot \vec{c}} \tag{IV}$$

Und damit nach Einsetzen von (IV) in (III) und dessen Vereinfachung:

$$\beta = \frac{\vec{a} \cdot \vec{b} \;\cdot\; \vec{c} \cdot \vec{c} \;-\; \vec{a} \cdot \vec{c} \;\cdot\; \vec{c} \cdot \vec{b}}{\vec{c} \cdot \vec{c} \;\cdot\; \vec{b} \cdot \vec{b} \;-\; \vec{c} \cdot \vec{b} \;\cdot\; \vec{b} \cdot \vec{c}} \tag{V}$$

Angemerkt sei, dass auch diese Nenner nicht Null werden können: $\vec{c}$ und $\vec{b}$ haben beide eine Länge ungleich Null, außerdem sind sie nicht identisch.

Mit den Gleichungen (IV) und (V) sind also die baryzentrischen Koordinaten bekannt und Satz 6 auf Seite 8 kann für den Schnitttest angewandt werden.

Satz 7 (*Berechnung der baryzentrischen Koordinaten*)

Gegeben ein Dreieck mit den Punkten $\vec{A}$, $\vec{B}$ und $\vec{C}$ und Kantenvektoren $\vec{b}$ und $\vec{c}$ bezüglich des Punktes $\vec{A}$. Für einen Punkt $\vec{q}$ in der Ebene des Dreiecks berechnen sich die baryzentrischen Koordinaten bezüglich $\vec{A}$ wie folgt:

$$\beta = \frac{\vec{a} \cdot \vec{b} \;\cdot\; \vec{c} \cdot \vec{c} \;-\; \vec{a} \cdot \vec{c} \;\cdot\; \vec{c} \cdot \vec{b}}{\vec{c} \cdot \vec{c} \;\cdot\; \vec{b} \cdot \vec{b} \;-\; \vec{c} \cdot \vec{b} \;\cdot\; \vec{b} \cdot \vec{c}}$$

$$\gamma = \frac{\vec{a} \cdot \vec{c} \;\cdot\; \vec{b} \cdot \vec{b} \;-\; \vec{a} \cdot \vec{b} \;\cdot\; \vec{b} \cdot \vec{c}}{\vec{c} \cdot \vec{c} \;\cdot\; \vec{b} \cdot \vec{b} \;-\; \vec{c} \cdot \vec{b} \;\cdot\; \vec{b} \cdot \vec{c}}$$

Wobei wie gehabt $\vec{a} = \vec{q} - \vec{A}$.

3.1 Optimierung durch Vorberechnung

Es fällt zunächst einmal auf, dass die Nenner von (IV) und (V) identisch sind. Außerdem sind $\vec{b}$ und $\vec{c}$ alleine vom Dreieck abhängig und nicht vom zu testenden Punkt. In $\vec{a}$ verbirgt sich hingegen noch $\vec{q}$. Wir rückstubstituieren an dieser Stelle aber nicht, sondern betrachten noch einmal Gleichung (V), führen den Bezeichner D als Verkürzung für den Nenner ein und formen etwas um:

$$\beta = \frac{\vec{a}\cdot\vec{b}\ \cdot\ \vec{c}\cdot\vec{c}\ \ -\ \ \vec{a}\cdot\vec{c}\ \cdot\ \vec{c}\cdot\vec{b}}{\underbrace{\vec{c}\cdot\vec{c}\ \cdot\ \vec{b}\cdot\vec{b}\ \ -\ \ \vec{c}\cdot\vec{b}\ \cdot\ \vec{b}\cdot\vec{c}}_{D}}$$

$$= \left(\vec{a}\cdot\vec{b}\ \cdot\ \vec{c}\cdot\vec{c}\ \ -\ \ \vec{a}\cdot\vec{c}\ \cdot\ \vec{c}\cdot\vec{b}\right)\cdot\frac{1}{D}$$

$$= \vec{a}\cdot\left(\vec{b}\cdot\frac{\vec{c}\cdot\vec{c}}{D}\right) - \vec{a}\cdot\left(\vec{c}\cdot\frac{\vec{c}\cdot\vec{b}}{D}\right)$$

$$= \vec{a}\cdot\underbrace{\left(\vec{b}\cdot\underbrace{\frac{\vec{c}\cdot\vec{c}}{D}}_{\in\mathbb{R}}\ -\ \vec{c}\cdot\underbrace{\frac{\vec{c}\cdot\vec{b}}{D}}_{\in\mathbb{R}}\right)}_{\vec{u}_\beta\in\mathbb{R}^3}$$

$\vec{c}\cdot\vec{c}$ ist ein Skalar, ebenso D. Das heißt, $\vec{b}$ wird zuerst skaliert, bevor das Skalarprodukt mit $\vec{a}$ gebildet wird. Analog wird $\vec{c}$ skaliert. Im letzten Schritt wird $\vec{a}$ ausgeklammert. Die gesamte Klammer namens $\vec{u}_\beta$ kann nun einmalig vorberechnet werden. Für statische Dreiecke muss dies nur einmal zu Beginn des Rendervorgangs geschehen, bei Animationen kann dieser Vorgang an den Anfang eines jeden Einzelbildes verlagert werden.

Auf die gleiche Weise lässt sich γ vereinfachen, sodass sich ergibt:

$$\gamma = \vec{a}\cdot\underbrace{\left(\vec{c}\cdot\underbrace{\frac{\vec{b}\cdot\vec{b}}{D}}_{\in\mathbb{R}}\ -\ \vec{b}\cdot\underbrace{\frac{\vec{b}\cdot\vec{c}}{D}}_{\in\mathbb{R}}\right)}_{\vec{u}_\gamma\in\mathbb{R}^3}$$

Im Moment sind also für die Berechnung von β und γ folgende Schritte nötig:

- Berechnen von $\vec{a} = \vec{q} - \vec{A}$, das sind drei elementare Subtraktionen.
- Berechnen von $\beta = \vec{a}\cdot\vec{u}_\beta$, also ein Skalarprodukt.
- Berechnen von $\gamma = \vec{a}\cdot\vec{u}_\gamma$, ein weiteres Skalarprodukt.

Insgesamt lässt sich noch eine elementare Operation einsparen, indem man betrachtet:

$$\beta = \vec{a}\cdot\vec{u}_\beta = \left(\vec{q} - \vec{A}\right)\cdot\vec{u}_\beta$$

$$= \vec{q}\cdot\vec{u}_\beta + \underbrace{(-\vec{A}\cdot\vec{u}_\beta)}_{k_\beta}$$

k_β ist wieder nur von den Parametern des Dreiecks abhängig, nicht vom potentiellen Schnittpunkt. Analoges gilt für γ:

$$\begin{aligned}\gamma = \vec{a} \cdot \vec{u}_\gamma &= \left(\vec{q} - \vec{A}\right) \cdot \vec{u}_\gamma \\ &= \vec{q} \cdot \vec{u}_\gamma + \underbrace{(-\vec{A} \cdot \vec{u}_\gamma)}_{k_\gamma}\end{aligned}$$

Man kann also auch k_β und k_γ im Voraus berechnen und speichern. Dann sind pro Schnitttest nur noch folgende Operationen notwendig:

- Berechnen von $\beta = \vec{q} \cdot \vec{u}_\beta + k_\beta$, also ein Skalarprodukt und eine Addition.
- Berechnen von $\gamma = \vec{q} \cdot \vec{u}_\gamma + k_\gamma$, noch ein Skalarprodukt und eine Addition.

Detailliert zu sehen ist dies in Listing 2, das wieder einen gekürzten Auszug aus dem NetTracer-Code zeigt. Ausprogrammiert ist dort die Vorberechnung und der Test „Strahl vs. Dreieck“, der bekanntlich auch den Test „Strahl vs. Ebene“ enthält.

Listing 2: Ein Dreieck testet sich auf Schnittpunkte mit einem Strahl.

```
37  public class Triangle3D
38  {
39    private Vec3 n, uBeta, uGamma;
40    private double d, kBeta, kGamma;
41
42    /** Vorberechnen statischer Werte. */
43    public Triangle3D(Vec3[] vertices)
44    {
45      // Kantenvektoren.
46      Vec3 b = vertices[1].minus(vertices[0]);
47      Vec3 c = vertices[2].minus(vertices[0]);
48
49      // Ebenenparameter.
50      n = b.cross(c);
51      n.normalize();
52      d = n.dot(vertices[0]);
53
54      // Berechne die großen Klammern, sodass beim Schnitttest weniger
55      // Arbeit getan werden muss.
56      double bb = b.dot(b);
57      double bc = b.dot(c);
58      double cc = c.dot(c);
59
60      double D = 1.0 / (cc * bb - bc * bc);
61      double bbD = bb * D;
62      double bcD = bc * D;
63      double ccD = cc * D;
64
65      uBeta  = b.times(ccD).minus(c.times(bcD));
66      uGamma = c.times(bbD).minus(b.times(bcD));
67
68      kBeta  = -vertices[0].dot(uBeta);
69      kGamma = -vertices[0].dot(uGamma);
70    }
71
72    /** Führt einen Schnitttest mit einem Strahl durch. */
73    public Vec3 intersectionTest(Ray r)
74    {
75      // Schnitttest Ray -> Ebene
76      double rn = r.direction.dot(n);
77      if (Math.abs(rn) < 1e-15)
78        return null;
79
80      // Wie weit hat es der Ray von seinem Ursprung zum Schnittpunkt?
81      double alpha1 = (d - r.origin.dot(n)) / rn;
82      if (alpha1 <= 0.0)
83        return null;
84
85      // Schnittpunkt q mit Ebene gefunden, liegt er im Dreieck?
86      Vec3 q = r.evaluate(alpha1);
87      double beta = uBeta.dot(q) + kBeta;
88      if (beta < 0.0)
89        return null;
90
91      double gamma = uGamma.dot(q) + kGamma;
92      if (gamma < 0.0)
93        return null;
94
95      double alpha = 1 - beta - gamma;
96      if (alpha < 0.0)
97        return null;
98
99      // Es gibt also einen Schnittpunkt und dieser ist q. alpha, beta
100     // und gamma könnten für Texturierung oder ähnliches weiter-
101     // verwendet werden.
102     return q;
103   }
104 }
```