Heuristics for ray tracing using space subdivision

Arne Sieverling

12. Februar 2010

1 Einführung

Im Rahmen des Seminars "Computergrafik" an der Georg-August Universität Göttingen im Wintersemester 09/10, betreut von Prof. Dr. Winfried Kurth und Reinhard Hemmerling, beschäftigte ich mich mit einem "klassischen" Paper zur hierarchischen Raumaufteilung beim Ray Tracing aus dem Jahre 1990. In dem gehaltenen Vortrag und dieser Ausarbeitung werde ich die Grundzüge der Arbeit erläutern. Ich zog als weitere Quelle eine Dissertation von V. Havran aus dem Jahre 2000 hinzu, die mit aktuelleren Methoden die Kernpunkte des vorgestellten Verfahrens validiert.

2 Grundlagen

2.1 Ray Tracing

Ray Tracing ist eine Technik zur Erstellung von Bildern aus einer dreidimensionalen Szenenbeschreibung. Hierbei werden Strahlen von der Bildebene aus in die Szene ausgesendet und verfolgt. Treffen sie auf ein Objekt, so wird zum Einen die Lichtintensität des Objektes selber im Strahl erfasst, zum Anderen werden die gebrochenen und reflektierten Strahlen rekursiv weiterverfolgt. Hauptaufwand bei dieser Technik ist die Berechnung der Schnittpunkte der Strahlen mit den Objekten in der Szene. Ziel ist es die Anzahl dieser Tests zu minimieren, indem man weit vom Strahl entfernte Objekte nicht berücksichtigt.

Eine Methode zur Vereinfachung der Schnitttests ist die Einführung von *Bounding Volumes*. Diese sind simple Volumina wie z.B. Kugeln oder Quader, die um die Objekte gelegt werden. Schneidet der Strahl das Volumen

nicht, so muss der aufwendige Schnitttest mit dem in der Box enthaltenen Objekt nicht durchgeführt werden.

2.2 Hierarchische Raumaufteilung

Verbesserung der Geschwindigkeit des Ray Tracings erreicht man durch eine hierarchische Aufteilung der Szene. Unterschieden wird zwischen *Object Subdivision* und *Space Subdivision*.

Object Subdivision beschreibt die Gruppierung der im Raum enthaltenen Objekte. Eine Möglichkeit ist die *Bounding Volume Hierachy*. Hierbei werden Bounding Volumes um die Objekte gelegt. Um mehrere dieser Volumen dann wieder ein weiteres Bounding Volume und so weiter, bis das letzte Bounding Volume die gesamte Szene umfasst. Gespeichert wird diese Hierarchie dann in einer Baumstruktur. Die Erstellung guter *Bounding Volume Hierarchies* ist kein triviales Problem, an dieser Stelle wird auf dieses Verfahren nicht weiter eingegangen.

Space Subdivision beschreibt die Aufteilung des Raumes in kleinere Teilräume. Auch diese Aufteilung lässt sich hierarchisch durchführen. So wird der gesamte Raum z.B. in mehrere Teilräume aufgeteilt, diese Teilräume dann wieder, bis man kleine Räume hat, die nur noch wenige Objekte umfassen. Auch hier kann zur Speicherung dann wieder eine Baumstruktur benutzt werden.

Eine einfache Möglichkeit besteht in der Benutzung eines *Octrees*. Hier wird der Raum in jedem Schritt in 8 kleinere disjunkte Teilvolumen aufgeteilt. Die genauen Koordinaten, an denen in x,y und z-Richtung geschnitten wird sind variabel.

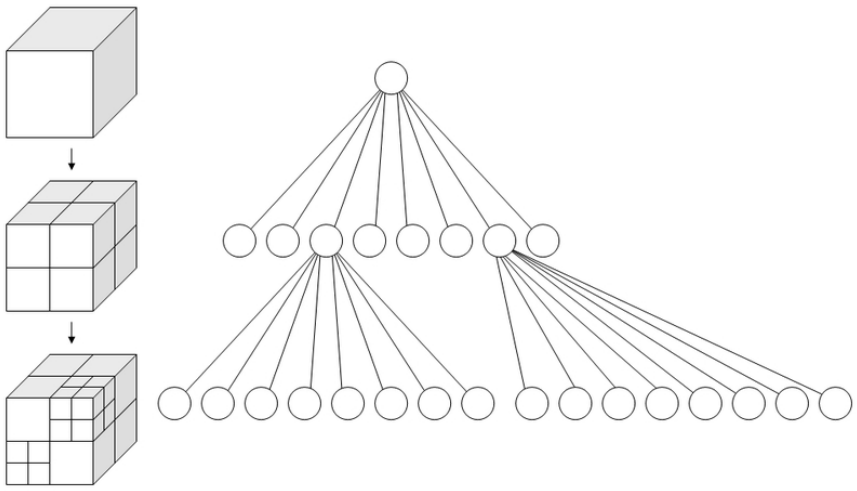

Abbildung 1: Wikipedia-User "Nü": Schemazeichnung eines de:Octrees, einer Datenstruktur der Informatik http://commons.wikimedia.org/wiki/File:Octree2.png, 7.4.2006

Eine weitere Möglichkeit der Raumaufteilung ist der *kd-Tree* (k-dimensional tree). Dabei wird der Raum in jedem Schritt in eine der drei Raumrichtungen auf beliebiger Koordinate aufgeteilt. Diese Raumrichtung wird in dem Baum in jeder Ebene mit gespeichert.

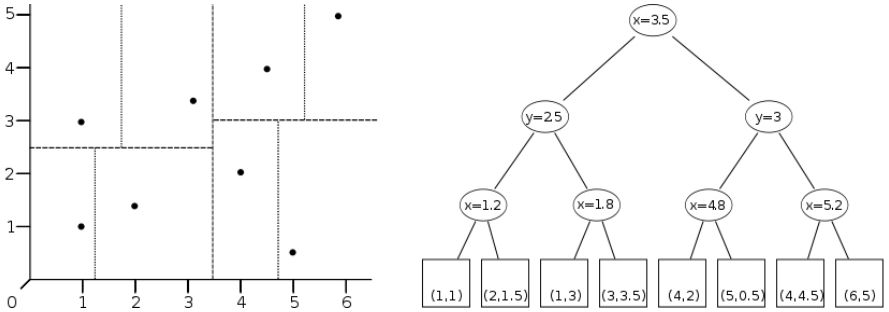

Abbildung 2: Wikipedia user "3bit": "2-d-Tree Example, Computer Science", http://commons.wikimedia.org/wiki/File:2dbaum.svg, 2.9.2006

Der kd-Tree ist eine Verallgemeinerung des Octrees, denn jeder Octree kann durch einen äquivalenten kd-Tree beschrieben werden, indem man nur zyklische Schnitte in x,y und dann z-Richtung zulässt. Deswegen wird im weiteren nur der kd-Tree genauer betrachtet.

Ausschlaggebend für die Performance einer Raumaufteilung ist die Strategie, mit der die Schnitte in die Raumrichtungen durchgeführt werden. Eine

einfache Möglichkeit besteht darin, den Raum immer in der Mitte zu teilen (räumlicher Median), was zu einer gleichmäßigen Raumaufteilung führt. Bei fester Baumtiefe erreicht man so eine äquidistantes Gitter auf dem Volumen. Eine zweite Möglichkeit besteht darin, den Objekt Median zu nutzen. hierbei wird so geschnitten, dass sich links und rechts vom Schnitt gleich viele Objekte befinden.

Die Reihenfolge der Achsen in der geteilt wird ist ebenfalls relevant. Zyklische Teilung in x,y, und z-Richtung führt zu einem Octree, alternativ kann man immer die längste Achse zuerst teilen, um den Raum in würfelähnliche Volumina aufzuteilen.

Ebenso wichtig für beide Bäume ist es festzulegen, bei welcher Tiefe der Raum nicht mehr weiter unterteilt wird. Eine Möglichkeit ist es, bei einer festen Tiefe abzubrechen, was zu einem balancierten Baum führt. Eine andere Möglichkeit besteht darin so lange zu teilen, bis jedes Objekt genau einem Blatt des Baumes entspricht.

3 Oberflächenheuristik

Wir wollen alle diese Strategien miteinander Vergleichen. Dafür leiten wir eine Metrik her, die die Kosten eines kd-Trees angibt. Hiermit können wir die genannten Strategien vergleichen und bessere Strategien entwickeln.

3.1 Oberflächenmetrik

Um das Problem der Schnittpunkte eines Strahles mit Objekten in einem Volumen zu vereinfachen machen wir ein paar mehr oder weniger realistische Annahmen.

- Die Strahlen sind in Ursprung und Richtung gleichverteilt.
- Die Strahlquellen sind weit entfernt von Objekten.
- Die Strahlen treffen kein Objekt im Modell.

Alle diese Annahmen sind natürlich bei allen realistischen Ray Tracing Anwendungen verletzt, führen aber zu einem Modell, mit dem sich gut weiter rechnen lässt.
In diesem Modell lässt sich leicht folgende Relation zeigen:

$$\text{Anzahl der Strahlen, die Knoten } K \text{ schneiden } \sim SA(K),$$

wobei $SA(K)$ die Oberfläche des Baumknotens K beschreibt.

Daraus folgt direkt:

$$P(\text{Strahl schneidet Knoten } K) = \frac{SA(K)}{SA(R)},$$

R ist der Wurzelknoten des Baumes.

Weiterhin kann man zeigen:

- Anzahl getroffene Knoten: $n_K = \sum_{k=1}^{N_k} \frac{SA(k)}{SA(R)}$
- Anzahl getroffene Blätter: $n_B = \sum_{b=1}^{N_b} \frac{SA(b)}{SA(R)}$
- Anzahl Strahl-Objekt-Test: $n_t = \sum_{b=1}^{N_b} \frac{SA(b)N_{obj}(b)}{SA(R)}$

N_k ist die Anzahl der Knoten im Baum, N_b die Anzahl der Blätter und $N_{obj}(b)$ die Anzahl der Objekte in Blatt b.

Nun können wir die Kosten berechnen, die die Traversierung des Baumes für einen Strahl verursacht.

$$C_{ges} = c_K n_K + c_B n_B + c_t n_t$$

c_K sind Kosten für die Traversierung eines Knotens, c_B die Kosten für die Traversierung eines Blattes und c_t die Kosten für Strahl-Objekt-Test.
Anhand eines sehr simplen Ray Tracing Problems wurde diese Metrik validiert.

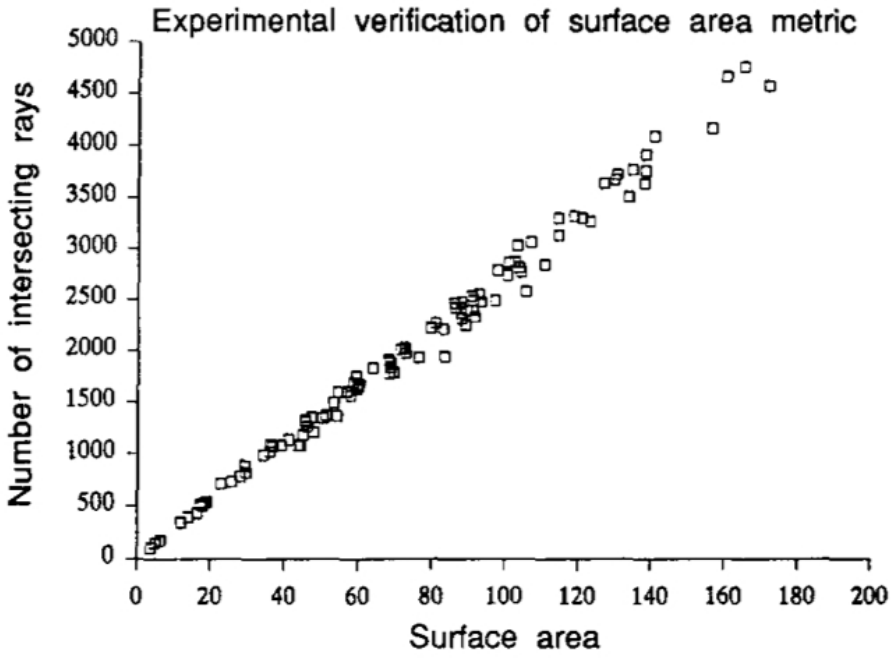

Abbildung 3: Mac Donald,J.D. , Booth, K.S.: "Heuristics for ray tracing using space subdivision (1990), The Visual Computer 6: S. 158"

Man sieht den linearen Zusammenhang zwischen der Anzahl der Strahlen, die einen Knoten schneiden und der Oberfläche des Knotens. Die Testszene bestand aber nur aus sehr einfachen geometrischen Objekten, die von allen Seiten beleuchtet wurden, also keinesfalls ein modernes Ray Tracing Beispiel. Bei komplexeren Szenen mit nicht verteilten Lichtquellen würde die Metrik ein deutlich schlechteres Ergebnis erzielen.

3.2 Optimale Raumaufteilung

Mittels der hergeleiteten Kostenfunktion können nun die Kosten berechnet werden, die eine bestimmte Raumaufteilungstechnik verursacht. Wir führen die Variable b ein, die angibt, bei welcher Koordinate der Raum aufgeteilt wird. $b = 0$ ist dabei der kleinstmögliche Wert, $b = 1$ der größtmögliche. Die Anzahl der getroffenen inneren Knoten und Blätter des Baumes in der Kostenfunktion bleibt unverändert. Der Term $c_t n_t$ wird dann abhängig vom Schnitt an b zu:

$$f(b) = LSA(b) \cdot L(b) + RSA(b) \cdot (n - L(b))$$

$LSA(b)$ ist die Oberfläche des Knotens links von b, $RSA(b)$ die Oberfläche

der rechten Hälfte. $L(b)$ ist die Anzahl aller Objekte links von b und n die Anzahl aller Objekte im Volumen.

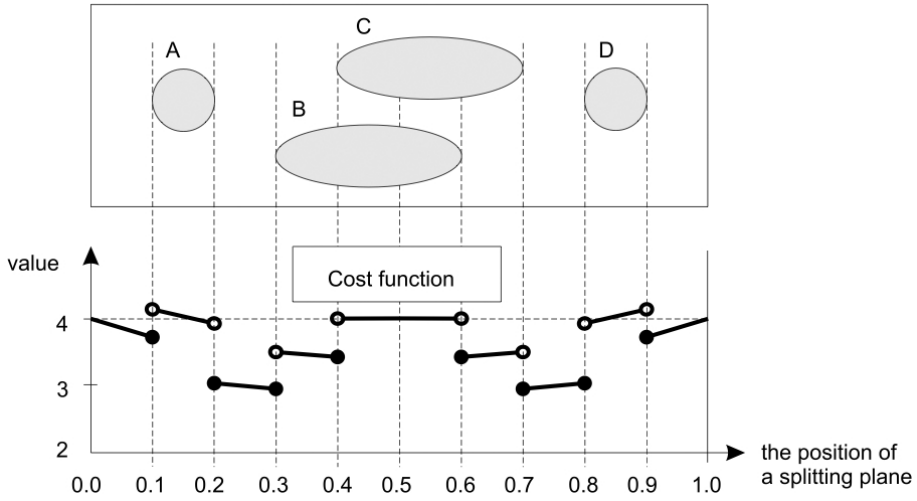

Abbildung 4: Havran, V.: "Heuristic Ray Shooting Algorithms"(2000) S.57

Nun untersuchen wir die vorgestellten Techniken. Für den räumlichen Median bei $b = 0.5$ gilt:

$$f(b_{rm}) = LSA(0.5) \cdot L(0.5) + RSA(0.5) \cdot (n - L(0.5)) = n \cdot LSA(0.5)$$

Denn $LSA(0.5) = RSA(0.5)$.

Für den Objekt Median gilt hingegen:

$$f(b_{om}) = LSA(b) \cdot n/2 + RSA(b) \cdot n/2 = n \cdot LSA(0.5)$$

Da $LSA(b) + RSA(b)$ konstant. $\Rightarrow LSA(b) + RSA(b) = 2 \cdot LSA(0.5)$

Etwas unerwartet ergibt sich also, dass die simple Technik der Aufteilung des Raumes in der Hälfte den gleichen Erfolg liefert wie die Suche nach einem Objekt Median.

Interessant ist nun die Frage, ob es eine Technik mit besserer Kostenfunktion gibt. Zur Suche des Minimums können wir $f(b)$ nach b differenzieren:

$$f'(b) = LSA'(b) \cdot L(b) + LSA(b) \cdot L'(b) + n \cdot RSA'(b) - RSA'(b) \cdot L(b) - RSA(b) \cdot L'(b)$$

Wir setzen nun $-LSA'(b) = RSA'(b)$, da $LSA(b) + RSA(b)$ konstant ist.

$$\Rightarrow f'(b) = (2 \cdot L(b) - n) \cdot LSA'(b) + (LSA(b) - RSA(b)) \cdot L'(b)$$

$L'(b)$ ist nichtnegativ.

Wir betrachten nun zuerst den Fall, in dem der Objektmedian links von dem räumlichen Median liegt, also $b_{om} < b_{rm}$ ist und $LSA(b) < RSA(b)$ für b zwischen diesen Medianen.

$$\begin{aligned} f'(b_{om}) &= (2 \cdot \frac{n}{2} - n) \cdot LSA'(b_{om}) + (LSA(b_{om}) - RSA(b_{om})) \cdot L'(b_{om}) \\ &= (LSA(b_{om}) - RSA(b_{om})) \cdot L'(b) < 0 \end{aligned}$$

An der Stelle des räumlichen Medians gilt für die Ableitung:

$$\begin{aligned} f'(b_{rm}) &= (2 \cdot L(b_{rm} - n) \cdot LSA'(b_{rm}) + (LSA(b_{rm}) - RSA(b_{rm})) \cdot L'(b_{rm}) \\ &= (2 \cdot L(b_{rm}) - n) \cdot LSA'(b_{rm}) > 0 \end{aligned}$$

Für den Fall, dass der Objektmedian rechts vom räumlichen Median liegt lässt sich analog $f'(b_{om}) > 0$ und $f'(b_{rm}) < 0$ zeigen. Daraus folgt, dass das Minimum von f genau zwischen den beiden Medianen angenommen wird, sofern sich dort keine Singularitäten befinden.

Mit diesem Wissen können wir nun neue Techniken zur Bestimmung eines guten Wertes für b herleiten:

Eine einfache Möglichkeit besteht in der Teilung bei $(b_{rm} + b_{om})/2$. Eine weitere ist das Sampling von Werten zwischen den Medianen, was zu einem weiteren Verfahren führt:

- Berechne Objektmedian b_{om} in x,y und z-Richtung
- Sample Werte b_i mit $n/2 \leq b_i \leq b_{om}$
- Berechne $f(b_i)$
- Teile bei $\min f(b_i)$

Hierbei wird auch gleich eine optimale neue Raumrichtung gefunden, in der geschnitten wird.

Auch für die Maximaltiefe des Baumes lässt sich eine gute Methode finden. Man teilt einfach den Raum auf, bis $f(b) \leq k$ Für alle b, mit Konstante k, also kein genügend großer Gewinn mehr durch eine Teilung zu erwarten ist.

3.3 Validierung

In der Dissertation von V. Havran wurden die genannten Strategien auf einem modernen Ray Tracing System mit 30 verschiedenen Testszenen verglichen

Strategie	avg	best	worst
3	0%	0%	0%
1	+1804%	+25%	+36604%
2	+354%	-7%	+1755%
4	-3%	-12%	+5%
5	+7%	-8%	+88%

- 3: Optimale Schnittebene in allen Richtungen berechnet mit Kostenfunktion (Referenzalgorithmus)
- 1: Räumlicher Median und zyklischer Achsendurchlauf (entspricht Octree)
- 2: Objekt Median und zyklischer Achsendurchlauf
- 4: Optimale Schnittebene, Suche zwischen den Medianen
- 5: Optimale Schnittebene und zyklischer Achsendurchlauf

Angegeben ist die durchschnittliche, beste und schlechteste relative Laufzeit des kompletten Ray Tracing Vorganges, inklusive Aufbau des kd-Trees im Vergleich zu dem Referenzalgorithmus 3.

Man sieht eine deutlich bessere Performance der Algorithmen, die auf der Heuristik basieren. Zwischen diesen Algorithmen 3,4 und 5 bestehen nur geringe Unterschiede, die vor allem abhängig von der charakteristik der Szene sind.

Ebenfalls interessant ist es, dass der Objekt Median deutlich besser ist als der räumliche Median, was den theoretischen Ergebnissen widerspricht. Hier merkt man deutlich die Grenzen der anfangs geforderten Annahmen.

4 Implementierung

In dem Paper werden noch ein paar Dinge über die Datenstruktur in der der Baum gehalten werden soll gesagt. Verfolgt man einen Strahl durch die Szene, so wandert man zuerst den Baum von der Wurzel bis hinunter zu

einem Blatt. Schlägt dort der Schnitttest fehl, so bewegt sich der Strahl weiter durch die Szene. In einer simplen Baumstruktur würde dies bedeuten, dass man von dem Blatt aus den Baum erst nach oben traversieren muss um dann wieder nach unten zu dem nächsten Blatt zu gelangen, welches den Nachbarn des zuerst besuchten Knotens darstellt. Man kann *Nachbarschafts Links* einführen, um die hochtraversierung des Baumes komplett zu verhindern. Dafür speichert man für jeden Blattknoten Verknüpfungen zu den 6 Nachbarblättern in der Datenstruktur ab. Man wandert den Baum dann einmal von der Wurzel bis zum Blatt hinab und verfolgt dann nur noch diese Nachbarschaftslinks, wenn sich der Strahl durch die Szene bewegt.
In der Praxis lassen sich mit diesem Verfahren bei komplexeren Szenen Geschwindigkeitsvorteile erzielen.

5 Diskussion

Zusammenfassend lässt sich sagen, dass die mit der Heuristik entwickelten Verfahren leistungsfähiger als einfachere Median-Verfahren sind. Auch wenn hier innerhalb der letzten 20 Jahre viel verbessert wurde liefert die Oberflächen Heuristik eine Strategie zur Erstellung von kd-Trees, die den Ray Tracing Prozess deutlich schneller macht.
In der Arbeit unbeantwortet bleibt die Frage, ob Object Subdivision Verfahren bessere Ergebnisse liefern können. Diese haben einen großen Vorteil bei nicht-statischen Szenen, denn eine kleine Änderung in der Szene erfordert nur eine kleine Änderung in der Objekt-Hierarchie. Bei Space Subdivision-Verfahren muss hingegen teilweise der ganze Baum neu berechnet werden.

Literatur

[1] Mac Donald,J.D. , Booth, K.S.: “Heuristics for ray tracing using space subdivision“ (1990) aus: The Visual Computer 6: S. 153-166

[2] Havran, V.: “Heuristic Ray Shooting Algorithms“ (2000),Dissertation an der technischen Universität Prag