[SoSe2016PunktZuNetzDistanz](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz)

**Inhalt**

* [SoSe2016PunktZuNetzDistanz](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#SoSe2016PunktZuNetzDistanz)
  + [Projektbearbeiter](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Projektbearbeiter)
  + [Zusammenfassung / Abstract](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Zusammenfassung_47_Abstract)
  + [Keywords](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Keywords)
  + [Arbeitsplan](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Arbeitsplan)
  + [Einführung](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Einf_252hrung)
    - [AABB Axis Aligned Bounding Box](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#AABB_Axis_Aligned_Bounding_Box)
    - [OBB Oriented Bounding Box](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#OBB_Oriented_Bounding_Box)
    - [CGAL AABB-Bäume](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#CGAL_AABB_45B_228ume)
    - [Rekonstruktion von AABB-Teilbäumen](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Rekonstruktion_von_AABB_45Teilb_228umen)
    - [OBB Bäume](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#OBB_B_228ume)
    - [Vergleich OBB und AABB Bäume](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Vergleich_OBB_und_AABB_B_228ume)
  + [Aufgabenanalyse](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Aufgabenanalyse)
  + [Lösungsansatz](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#L_246sungsansatz)
  + [Design](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Design)
  + [Implementierung](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Implementierung)
  + [Resultate](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Resultate)
  + [Diskussion](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Diskussion)
  + [Fazit und Ausblick](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Fazit_und_Ausblick)
  + [Arbeitsbuch](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Arbeitsbuch)
  + [Referenzen](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Referenzen)
  + [Ressourcen](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz#Ressourcen)

Projektbearbeiter

| [**Name**](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz?sortcol=0;table=1;up=0#sorted_table) | [**Mail Hochschule**](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz?sortcol=1;table=1;up=0#sorted_table) | [**Mail extern**](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/SoSe2016PunktZuNetzDistanz?sortcol=2;table=1;up=0#sorted_table) |
| --- | --- | --- |
| Gert-Jan Rozing | [roge1013@hs-karlsruhe.de](mailto:roge1013@hs-karlsruhe.de) | [gert.rozing@myestro.de](mailto:gert.rozing@myestro.de) |

Zusammenfassung / Abstract

Die Bestimmung der Entfernung eines beliebigen Punktes zu einem Polygonnetz ist eine typische Aufgabe der Informatik. Diese zum Beispiel angewandt in der Vermessungstechnick oder der Computerspielebranche. Hierzu gibt es schon eine bestehende Lösung, welche eine realtiv simple "brute-force" Methode verwendet. Die Aufgabe ist es, einen Algorithmus zu finden, welcher eine bessere Komplexität hat als der "Brute-Force" Algorithmus.

Keywords

Geometrische Algorithmen, Baumstrucktur, AABB, CGAL, OBB, PQP

Arbeitsplan

* 06.04 Literaturrecherche, Lösungsansatz, Klärung der letzten ungenauigkeiten, Bestimmung der einzelnen Teile.
* 20.04 Implementation Teil A Untersuchung des gefunden CGAL AABB Algorithmus.
* 04.05 Implementation Teil B (Dieser wird noch näher definiert)
* 01.06 Implementation Teil C (Dieser wird noch näher definiert)
* 25.06 Vergleich der Ergebnisse und Vortrag.
* Endspurt

Einführung

<Kurze Betrachtung des aktuellen technischen/wissenschaftlichen Standes, Aufzeigen der Motivation für die vorliegende Arbeit.>

AABB Axis Aligned Bounding Box

Eine AABB ist ein Hüllkörper, welches ein beliebiges Polygon vollständig umschlingt. Der Hüllkörper hat immer eine quadratische, bzw. kubische Form dessen Kanten parallel mit dem Achsen des Koordinatensystems ist. Wird das Polygon im Hüllkörper verschoben, so kann der Hüllkörper entsprechend mit verschoben werden. Wird jedoch das Polygon gedreht, so muss der Hüllkörper neu berechnet werden.

OBB Oriented Bounding Box

Eine OBB ist wie eine AABB ein Hüllkörper, mit dem Unterschied, dass seine Kanten nicht mehr parallel zu den Koordinatenachsen sind. Wird das Polygon gedreht, so muss eine OBB nicht neu berechnet werden und kann einfach mitgedreht werden.

CGAL AABB-Bäume

Für die schnelle Berechnung der Distanz eines punktes zu einem Modell gibt es in der externen Bibliothek [CGAL](http://www.cgal.org/) (The Computational Geometry Algorithms Library) eine existierende Lösung. Diese basiert auf dem Ansatz die bestehenden Geometrischen Informationen in einem sogenannten AABB Baum anzuordnen. Hierbei wird das Modell in eine Menge von Primitiven zerlegt (wie zum Beispiel Dreiecke) und in einer Baumstrucktur angeordnet. Dieser Vorgang hat eine Komplexität von O(nlog(n)), wobei N die Anzahl der Primitiven ist.

Es können beliebige neue Primitive zu dem Baum hinzugefügt werden, jedoch muss der gesamte Baum anschließend neu gebaut werden. Es ist somit nicht möglich ausschließlich Teilbäume zu bauen.

Für die Abstandsbestimmung werden intern in der statischen AABB Baumstrucktur KD-Bäume verwendet.

Rekonstruktion von AABB-Teilbäumen

In dem Paper (insert Referenz here) wurde Beschrieben, wie es möglich ist bei einem geänderten Modell nur Teile des AABB-Baums zu rekonstruieren. Eine Rekonstruktion ist immer deutlich schneller als die komplette neu generierung eines AABB-Baums, jedoch verliert der Baum durch die Rekonstruktion an Genauigkeit. Die Laufzeit einer Rekonstruktion ist die selbe wie die der Neugenerierung eines Baums O(n log(n)), wobei hier n die Anzahl geänderter AABBs sind.

Das Problem was entsteht ist, dass bei der Neugenerirung eines AABB-Baums die Überlappung einzelner AABBs möglichst gering gehalten wird. Wird jedoch ein Teilbaum neu generiert, so ist es möglich, dass dieser eine nicht mehr optimale Überlappung besitzt. Dies beeinflusst nicht die Anzahl an BBs innerhalb eines Baums. Eine ungeschickte Überlappung erhöht jedoch die Anzahl an abfragen die nötig sind um eine Abstandberechnung durchzuführen. Es ist somit möglich, dass im schlimmsten Fall eine Berechnung eine Laufzeit von O(n²) bebötigt.

OBB Bäume

Bei der Verwendung von OBB Bäumen, stellt sich ein neues Problem welches beachtet werden muss. Da OBB beliebig im Raum liegen können, können diese beliebig um ein Primitiv liegen. Es zeigt sich jedoch, dass für einen OBB Baum, die OBBs möglichst eng an den Primitiven liegen müssen. Die Berechnung eines solchen OBBs kann mit einer Komplexität von O(n³) berechnet werden. Dies ist für große Modelle nicht optimal, es gibt jedoch dokumentierte Algorithmen (füge referenz ein) welche dies unter bestimmten Umständen schneller können.

Genau diese generierung von OBBs ist auch der Grund, warum die generierung von OBB Bäumen länger brauchen. Da jedoch die OBBs näher an dem eigentlichen Primitiv liegen, hat der OBB Baum weniger Tiefe als ein AABB Baum.

Vergleich OBB und AABB Bäume

Aus den in (Insert Referenz hier) beschriebenen Tests hat sich gezeigt, dass OBB Bäume schnellere Abstandsberechnungen ermöglichen, jedoch mehr Zeit benötigen um erstellt zu werden, mehr Speicherplatz brauchen und nicht teilweise neu genereirt werden können. Im Vergleich dazu sind AABB Bäume schneller erstellt, verbrauchen weniger Speicherplatz, können teilweise neu generiert werden, haben dafür jedoch eine langsamere Abstandsberechnung. Da diese Ergebnisse aus den frühen 90ern sind gilt es diese zu bestätigen.

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Torus | X-Wing | Teapot |
| Build an OBB Tree | 0,35s | 0,46s | 0,27s |
| Build an AABB Tree | 0,11s | 0,18s | 0,08s |
| Refit an AABB Tree | 15ms | 18ms | 11ms |
| Test a pair of OBB Trees | 0,5ms | 2,3ms | 0,6ms |
| Test a pair of ABB Trees | 1,3ms | 3,3ms | 0,8ms |

Für den Vergleich der beiden gefunden Implementationen wurde ein kleines C Program geschrieben welches die folgenden Eigenschaften hat. Es werden zwei sich ähnliche Modelle eingelesen. Diese Modelle befinden sich im Stanford Triangle Format, auch als PLY bekannt. Das erste Modell dient als tatsächliches Modell an dem später ein OBB oder AABB Baum gebaut wird. Das zweite Modell dient lediglich als eine Punktwolke für die berechnung. Beide Modelle werden eingelesen und liegen im Speicher als zwei Listen vor, eine für die Ecken und eine für Flächen.

Anhand dieser Listen können für das Modell sowohl der AABB als auch der OBB Baum erstellt werden. Dieser Vorgang wird im Quellcode zeitlich gemessen und für eine spätere Verwendung gespeichert. Anschließend wird die Liste der Ecken von dem zweiten Modell dazu verwendet, um die Distanzen zum eigentlichen Modell zu berechnen. Auch diese Berechnung wird zeitlich gemessen. Hierbei gilt es zu beachten, dass bei der CGAL Implementation (AABB Bäume) sämtliche optimierungen eingeschalten wurden.

Die eigentliche Messung wird mit zwei Modellpaaren aus der MPI-Faust Datenbank durchgeführt. Diese bietet eine vielzahl von Polygonalen Modellen von Menschen in bestimmten Posen. Die Modelle liegen in zwei unterschiedlichen Detailstufen vor. Es wurden jeweils zwei sich ähnliche, jedoch unterschiedliche posen ausgesucht für die berechnung. Der Versuch mit der teilweise neuberechnung von AABB Bäumen wurde ausgelassen, da dieser nicht möglich ist unter OBB Bäumen.

|  |  |  |
| --- | --- | --- |
| Operation | Kleines Polygon | Großes Polygon |
| Build an OBB Tree | 59,749ms | 1,298s |
| Build an AABB Tree | 6,277ms | 0,329s |
| Test a pair of OBB Trees | 57,2302ms | 2,82s |
| Test a pair of AABB Trees | 35,0878ms | 5,903s |

Die Messungen wurden jeweils 5 mal wiederhohlt und anschließend der MIttelwert gebildet. Es gab keine nennenswerten Abweichungen. Aus der Tabelle wird deutlich, dass die Ergebnisse aus den Versuchen der frühen 90ern bestätigt werden können. Interessant ist, dass für detail arme Modelle, also Modelle mit einer geringen Polygon anzahl, die AABB Bäume wesentlich schneller sind als die OBB Bäume. Bei großen Modellen kehrt sich diese Beobachtung. Obwohl für große Modelle OBB Bäume eine höheren Zeitaufwand brauchen für die generierung, sind die Abstandsabfragen deutlich schneller als der AABB Bäume.

Aufgabenanalyse

Die Aufgabe ist es einen Algorithmus zu finden, welcher die Distanz eines Punktes zu einem Netz berechnet. Dieser Algorithmus muss von der Komplexität im O-Kalkül besser sein als der bestehende Brute-Force Algorithmus.

Die Punktewolken, sowie die Netze (Polygon Modelle) sind gegeben in Form von der [MPI-FAUST](http://faust.is.tue.mpg.de/)Datenbank. Diese Datenbank besteht sowohl aus Messdaten (Punktwolken) als auch detailierte Polygonmodelle der entsprechenden Messungen und eignen sich somit hervorragend für diese Arbeit. Hierbei sollte beachtet werden, dass die Polygon Modelle veränderbar sein werden.

Vorerst soll die Ausrichtung der Punktwolke nicht auf das Modell angepasst werden. Dies spart 6 Freiheitsgrade ein (Translation, Rotation). Jedoch sollte bei der bearbeitung diese zukünftige Bedingung nicht vergessen werden.

Lösungsansatz

Optimierung von CGAL AABB Bäumen, sodass Teilbäume neugebaut werden können.

Umwandlung des Modells in ein Voxelmodell. Je nach Auflösung des Voxelmodells wird entsprechend viel Genauigkeit verloren, jedoch können anschließend Abstandberechnungen beschleunigt werden. Teiländerungen an dem Polygon Modell können schnell in neue Voxel berechnet werden (Es müssen nur die geänderten Dreiecke und deren Nachbarn neu berechnet werden).

Design

<Wie sieht die Lösungsstruktur aus (mit Begründung). Die Designbeschreibung soll auch die Struktur für die folgende Implementierungsbeschreibung vorgeben.>

Implementierung

Die eigentlichen Algorithmen wurden nicht selbst Implementiert. Statt dessen wurden bekannte Algorithmen verwendet. Für die AABB Bäume wurde die [CGAL](http://doc.cgal.org/latest/AABB_tree/) Implementation gewählt, während für die OBB Bäume die [PQP](http://gamma.cs.unc.edu/SSV/" \t "_top)Implementation gewählt wurde. Für den eigentlichen Vergleich wurde ein kleines C Programm geschrieben, welches als Vergleichsfläche für die beiden Implementierungen verwendet wurde. Dieser Code befindet sich im Anhang im Projekt.zip.

Resultate

Aus den Messungen hat sich gezeigt, dass die Ergebnisse aus den 90ern noch aktuell sind. Für kleine, bzw. detail arme Polygone eignen sich AABB Bäume hervorragend für die Distanz berechnungen eines Punktes zu einem Modell. Bei größeren, bzw. detailierteren Polygonen sollten stattdessen OBB Bäume verwendet werden.

Problematisch war, dass die alte PQP Bibliothek nicht mithalten konnte mit der modernen CGAL Bibliothek. Die alte Bibliothek konnte genau 3 Funktionen, die nur mit Dreiecken und einer genauigkeit von double oder float arbeiteten konnte. Währenddessen war die moderne CGAL Bibliothek mit beliebigen Datentypen verwendbar, erweiterbar und bietete deutlich mehr Möglichkeiten.

Diskussion

<Wie sind die Ergebnisse zu interpretieren, welche Fragestellungen ergeben sich daraus, wie sehen die Ergebnisse im Vergleich zu anderen Publikationen aus.>

Fazit und Ausblick

Während der ersten Treffen wurden über diverse mögliche Ansätze diskutiert. Einige wurden aus dem guten Grund "das ist viel zu kompliziert" verworfen. Einer wurde nicht mehr weiter verfolgt, da dieser eventuell zu anspruchsvoll für eine Projektarbeit wäre. Hierbei ging es um die Berechnung des Abstandes über Voronoii zerlegungen. Während der bearbeitung der Projektarbeit bin ich zufällig über eine fertige Implementation, zusammen mit einem Paper, gestolpert. Diese vergleicht die Voronoii Lösung mit einer AABB Baum Lösung, hierbei Schnitt die Voronoii Lösung am besten ab. Es wäre sehr Interessant diese zu vergleichen mit der bestehenden OBB Implementation.

Auffällig war es auch, dass es keine moderne OBB Implementation existiert. Die ursprüngliche Implemenation unter RAPID, wurde nicht weiter entwickelt und erst Jahre später durch PQP vollständig implementiert. Es wäre eine bereicherung, wenn eine ordentliche OBB Baum Implementation in modernen Bibliotheken, wie zum Beispiel [OpenCv](http://iwi-i-lfm.hs-karlsruhe.de/foswiki/bin/view/IWI/OpenCv), oder auch CGAL hinzugefügt werden würde.

<Ähnlich wie das Abstract, eher zukunftsorientiert in Bezug auf weitergehende Fragestellungen. Beurteilung der Aufgabenstellung und der erreichten Ergebnisse. Wo und wie ließe sich das jetzt gewonnene Wissen weiterverwenden.>

Arbeitsbuch

06.04: An dem zweiten Treffen mit der Gruppe wurden die letzten unklarheiten in der Aufgabenstellung beseitigt und ein erster Lösungweg eingeschlagen. AABB Bäume aus der CGAL Bibliothek.  
08.04: Studieren der AABB Bäume von CGAL. Notizen dazu gemacht, jedoch noch nicht ins Wiki eingetragen.  
13.03 : Quellcode von meinem Betreuer erhalten und zum laufen gebracht. Erste Bilder betrachtet.  
18.04: Überlegung möglicher alternative Ansätze.  
19.04: Dokumentation der bisherigen Ergebnisse und Vorbereitung für das nächste Treffen.  
25:04 - 29.04: In dieser Woche ist mehr Zeit für anderen FH Kram und Arbeitskram draufgegangen. Die Woche war einfach zu schnell vorbei ;(.  
02.05: Lesen des zugekommenen Papers, sowie die in dem Paper angegebenen Quellen. (Sowie weiter interessante Quellen).  
03.04: Bearbeitung des gefunden Wissens und vorbereitung auf das nächste Treffen.  
10.05: Teil 2 vorbereitung auf das nächste Treffen, da dass letzte Treffen entfallen ist.

<Hier können sie ihre Labortätigkeit chronologisch protokollieren. Das Arbeitsbuch liefert dann die Inhalte für die logisch strukturierte Darstellung oben im Projektbericht.>

Referenzen

S. Gottschalk, M. C. Lin, and D. Manocha. 1996. OBBTree: a hierarchical structure for rapid interference detection. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*(SIGGRAPH '96). ACM, New York, NY, USA, 171-180. DOI=http://dx.doi.org/10.1145/237170.237244

*Efficient Collision Detection of Complex Deformable Models using AABB Trees* Gino van den Bergen Journal of Graphics Tools Vol. 2, Iss. 4, 1997

SUD, Avneesh, et al. Fast proximity computation among deformable models using discrete voronoi diagrams. *ACM Transactions on Graphics (TOG)*, 2006, 25. Jg., Nr. 3, S. 1144-1153.