Automatisches Auflösen von Selbstschneidungen in zweidimensionalen Polygonen für ein CSG System

(Diplom-)Hausarbeit

an der

Verwaltungs- und Wirtschafts-Akademie in Leer e.V.

von

Heiko Alexander Weber

Leer, Groninger Str.

Betreuung:

Prof. Dr. Thorsten Spitta

2017

>

Inhalt

[Abstrakt 3](#_Toc492486195)

[1 Einleitung und Zieldefinition 3](#_Toc492486196)

[2 Betriebswirtschaftlicher Aspekt 4](#_Toc492486197)

[3 Definition der Problemstellung 5](#_Toc492486198)

[3.1 Was ist ein Punkt? 5](#_Toc492486199)

[3.2 Was ist eine Kante? 5](#_Toc492486200)

[3.3 Was ist ein Polygon? 5](#_Toc492486201)

[3.4 Was ist eine Selbstschneidung? 5](#_Toc492486202)

[4 Datenstruktur 6](#_Toc492486203)

[4.1 Kriterien 6](#_Toc492486204)

[4.1.1 Komplexität bei Modifikation der Struktur 6](#_Toc492486205)

[4.1.2 Speicherbedarf 6](#_Toc492486206)

[4.2 Doubly connected edge list (DCEL) 6](#_Toc492486207)

[5 Operationen an einem n-gon 6](#_Toc492486208)

[5.1 Flächenberechnung 6](#_Toc492486209)

[6 Beschreibung des Algorithmus 6](#_Toc492486210)

[6.1 Prä-prozessor 7](#_Toc492486211)

[6.1.1 Validität 7](#_Toc492486212)

[6.1.2 Selbstschneidung 7](#_Toc492486213)

[6.2 Prozessor 7](#_Toc492486214)

[6.3 Post-prozessor 7](#_Toc492486215)

[7 Bewertung des Algorithmus 7](#_Toc492486216)

[7.1 Ermittlung des Deltas von P0 und PE 7](#_Toc492486217)

[8 Fazit 7](#_Toc492486218)

[Literatur 8](#_Toc492486219)

# Abstrakt

Diese Arbeit findet im Rahmen des Studiengangs 16 and der VWA-Leer unter Betreuung von Prof. Dr. Spitta (Uni-Bielefeld) als (Diplom-)Hausarbeit statt. Hierbei handelt es sich um eine Vorstudie zu dem „Automatisches Auflösen von Selbstschneidungen in zweidimensionalen Polygonen für ein *CSG*[[1]](#footnote-1) System“.

# Einleitung und Zieldefinition

Ein bekanntes Problem bei *CSG*1 Systemen ist die Verwendung eines sich selbst schneidenden planaren Polygonen (im *R2*Raum) für weitere Berechnungen oder kinematische Operationen (wie Extrusionen). Diesem Problem ist bereits eine Menge an Literatur gewidmen. In diesem Kontext stellen die genannten Polygone die originären Daten dar.

Diese Arbeit hat die Entwicklung eines Algorithmus bei dem die Integrität dieser originären Daten, unter dem Aspekt der Selbstschneidung, bei deren Erfassung sichergestellt bzw. wieder hergestellt wird zum Ziel.

# Betriebswirtschaftlicher Aspekt

Die ORGADATA AG hat in der Vergangenheit bereits Vorkommnisse des in „*Einleitung und Zieldefinition*“ genannten Problems verzeichnet. Eine Lösung dieses Problems würde an der Stelle der Verwendung der originären Daten, zum Erzeugen abgeleiteter Daten (Gitternetzgeometrien im *R3* Raum) einen zu hohen Aufwand erfordern. Zusätzlich wäre dies nur eine Korrektur eines Symptoms und somit eine Verlagerung bzw. Verschleierung des Problems. Da dieses System, wie die meisten anderen, nach dem *GiGo*[[2]](#footnote-2) Prinzip arbeitet, führt eine Injizierung falscher Eingabeparameter zu einem falschen Ergebnis, im schlimmsten Fall sogar zu einem nicht definierten Ergebnis.

Momentan wird dieses Problem von der Abteilung, welche die entsprechenden originären Daten erfasst, gelöst in dem die Zeichnungen manuell auf Integrität überprüft und ggf. korrigiert werden. Dies ist aufgrund der massiven Datenmenge (mehrere zig tausend Zeichnungen) und der Tatsache, dass diese Überprüfung und Korrektur nur im bereits aufgetretenen Fehlerfall erfolgt, nicht sinnvoll. Ein weiterer Aspekt hierbei sind die Kosten, die das manuelle überprüfen bzw. korrigieren verursacht. Es sei folgende Beispielrechnung gegeben:

Ein Mitarbeiter *e*, verantwortlich für die Erfassung originärer Daten, überprüft und korrigiert eine Menge an Zeichnungen *d* pro Monat, bei welchen durchschnittlich ein Zeitaufwand *h* pro Einheit notwendig ist. Genannter Mitarbeiter verursacht der ORGADATA AG monatlich Kosten *k*, bei einer Gesamtarbeitszeit (*H*) pro Monat.

Folgende Kosten fallen pro Mitarbeiter monatlich zur Bearbeitung dieser Sache an:

Unter der Annahme *beispielhaft angelehnter Werte*[[3]](#footnote-3) würden somit die vorhergehende Formel folgendermaßen materialisiert:

Es wurden hierbei verursachte Gesamtkosten pro Monat in Höhe von 2000€, 160 Stunden Arbeit pro Monat, 20 Zeichnungen pro Monat sowie ein Zeitaufwand von 0.5 Stunden pro Zeichnung veranschlagt.

Es sei gesagt, dass die *kaum schätzbaren Kosten, welche aus den auftretenden Fehlerfällen resultieren*[[4]](#footnote-4) in dieser Rechnung noch nicht berücksichtigt sind. Diese sind, grob geschätzt, weitaus höher. Deweiteren sind dies die Kosten, welche pro Monat, pro Mitarbeiter gelten. Die tatsächlichen Kosten sind die errechneten multipliziert mit der Anzahl an Mitarbeitern welche in dieser Domäne arbeiten.

Dieses Projekt wurde, unter dem Aspekt der wirtschaftlichkeit, mit der Intention gewählt, die genannten Kosten zu minimieren bzw. zu eliminieren.

# Definition der Problemstellung

Der Titel dieser Arbeit beschreibt bereits im groben die Problemstellung, definiert allerdings noch nicht alle Aspekte dieser. In der Fachliteratur gibt es für einige verwendete Begriffe verschiedene Definitionen, welche im Laufe dieser Arbeit zu Missverständnissen oder Fehlern führen könnten. Daher wird an dieser Stelle sowohl die Problemstellung, als auch die Randparameter des entwickelten Algorithmus exakt definiert.

## Was ist ein Punkt?

Ein Punkt ist ein durch zwei *single-precision-floats*[[5]](#footnote-5), hier und , beschriebener Ortsvektor in einem zweidimensionalen lokalen *euklidischen[[6]](#footnote-6)* System.

## Was ist eine Kante?

Eine Kante ist eine gerichtete Strecke zwischen zwei Punkten. Sie ist immer definiert durch , also durch ein Liniensegment vom ersten zum zweiten Vertex, wodurch auch implizit die Richtung der Kante gegeben ist.

## Was ist ein Polygon?

Ein Polygon, oder oft „*n-gon*“ ist ein geordnete Menge an Punkten (im folgenden „Vertex“ bzw. „Vertizes“ genannt) in Kombination zu einer Menge an Kanten (im folgenden „Edge“ bzw. „Edges“ genannt) , wobei , , …, . Eine Ergänzung zu dieser Definition ist, dass ein Vertex immer Bestandteil von genau zwei Edges ist. Da die Definition einer Kante (3.2) keine Bögen erlaubt, ist für ein Polygon welches eine Fläche hat, immer gegeben [sinngem. nach GrnBm\_PolyDef].

Diese Restriktion beschreibt in Gänze alle möglichen validen Polygone, welche durch den Algorithmus bearbeitet werden können. Es sei hervorgehoben, dass diese Definition ebenfalls keine *inneren Polygone*[[7]](#footnote-7) erlaubt.

## Was ist eine Selbstschneidung?

Eine Selbstschneidung ist gegeben wenn eine beliebige Edge , eine andere beliebige Edge schneidet. Eine Schneidung zweier Strecken ist gegeben, wenn die Mächigkeit der Menge der Punkte, welche auf beiden geraden liegen, .

Sind zwei Liniensegmente parallel zueinander, ist ein singulärer Schnittpunkt bereits ausgeschlossen. Seien die Segmente definiert durch jeweilige Richtungsvektoren , sowie . Mit folgender Gleichung lässt sich die Parallelität zweier Vektoren ( ) feststellen:

In dieser Arbeit werden alle Arten von Selbstschneidungen berücksichtigt, da auch alle in der Realität vorkommen. Der Schwerpunkt liegt jedoch auf dem Auflösen von Schneidungen mit einem singulären Schnittpunkt.

# Datenstruktur

Dieses Kapitel beschreibt die im Algorithmus verwendeten Datenstrukturen in welcher die Daten repräsentiert werden. Die folgenden Kapitel beschreiben Operationen auf einer eben solchen Datenstruktur.

## Kriterien

Um eine optimale Datenstruktur wählen zu können, müssen zunächst die Daten welche abgebildet analysiert und die Low-Level-Operationen auf dieser definiert werden. Folgende Kriterien sind dabei besonders wichtig:

### Komplexität bei Modifikation der Struktur

Dieses Kriterium beschreibt die Effizienz der Datenstruktur, mit einer Modifikation der Elemente bzw. der Ordnung der Elemente umzugehen. Ein sehr wichtiger Aspekt hierbei ist die Effizienz bzgl. *Insertions and Removals[[8]](#footnote-8)*.

### Speicherbedarf

Ein weiteres wichtiges Kriterium ist der benötigte Speicherbedarf, welcher linear zur Komplexität der Geometrie bzw. zur Anzahl der Elemente in der Datenstruktur skalieren sollte.

## Doubly connected edge list (DCEL)

Die sogenannte „Double connected edge list“, oder auch „Halfedge data structure“, ist eine Datenstruktur welche in *Mesh Processing*[[9]](#footnote-9) Algorithmen sehr verbreitet ist.

# Operationen an einem n-gon

Dieses Kapitel beschreibt die im Algorithmus verwendeten Berechnungen bzw. Operationen, basierend auf einer *Halfedge Data Structure*.

## Flächenberechnung

Um die

# Beschreibung des Algorithmus

Der entwickelte Algorithmus teilt sich in drei Stufen.

1. *Präprozessor*
2. *Prozessor*
3. *Postprozessor*

Diese Stufen werden seriell ausgeführt und führen verschiedene Aufgaben durch. Diese werden im Folgenden beschrieben.

## Prä-Prozessor

Der Präprozessor bewertet den eingegebenen Datensatz und entscheidet, ob dieser für den Algorithmus geeignet bzw. ob dieser „fehlerhaft“ im Aspekt der Selbstschneidung ist. Anhand dieser Information kann der Datensatz für die Bearbeitung abgelehnt oder angenommen werden.

### Validität

Der erste Schritt ist die Überprüfung auf Validität der Daten selbst. Dies beinhaltet folgende Kriterien:

* Ist der Datensatz ein *n*-gon?
* Befinden sich alle Vertizes des Datensatzes auf einer Ebene (also im lokalen Raum)?
* Ist die Fläche ?

### Selbstschneidung

Anhand der Kriterien in *3.2* wird entschieden, ob der eingegebene Datensatz eine Selbstschneidende Kontour hat, oder nicht. Sollte dies nicht der Fall sein, lehnt der Algorithmus die Bearbeitung dieses Datensatzes ab.

## Prozessor

## Post-Prozessor

# Bewertung des Algorithmus

## Ermittlung des Deltas von P0 und PE

# Fazit

# Literatur

**[GrnBm\_PolyDef]** Grünbaum, Branko, Selfintersections of Polygons, WA 98195-4350, S.1.

**[ElNa09]** Elmasri, R.; Navathe, S.B.: Grundlagen von Datenbanksystemen – Ausgabe Grundstudium. 3.Aufl. Pearson, München – Boston et al. 2009 (Übers. US-Ausgabe 2000).

**[SpBi08]** Spitta, T.; Bick, M.: Informationswirtschaft – Eine Einführung. 2. Aufl., Springer, Berlin – Heidelberg – NewYork et al 2008.

1. Steht für „**C**onstructive **S**olid **G**eometry“ und beschreibt Systeme, mit denen feste Körper berechnet bzw. konstruiert werden. [↑](#footnote-ref-1)
2. steht für „Garbage in, Garbage out“ und bezeichnet eine Systemart bei der die Eingabe falscher Parameter zur Ausgabe eines falschen bzw. nicht definierten Ergebnis führt. [↑](#footnote-ref-2)
3. da die Offenlegung von Gehältern o.Ä. gegen die Geheimhaltungsvorschriften vertraulicher Daten verstöße. [↑](#footnote-ref-3)
4. Damit sind die durch genannte Fehlerfälle (Abstürze oder falsche Daten) auftretenden Kosten beim Kunden gemeint, welche indirekt der ORGADATA AG entstehen. Dazu gehört auch der Aufwand des Debuggens bzw. der Kategorisierung gemeldeter Fehler. [↑](#footnote-ref-4)
5. Gleitkommazahl mit einfacher Präzision (32 bit), IEEE 754 [↑](#footnote-ref-5)
6. Benannt nach dem griechischen [↑](#footnote-ref-6)
7. Stellen Löcher innerhalb der äußeren Kontour des Polygons dar und sind definiert durch ein eingebettetes Polygon. [↑](#footnote-ref-7)
8. Beschreibt das „Einsetzen“ eines Elementes an eine beliebige Position in der Struktur bzw. das entfernen eines Elementes aus dieser. [↑](#footnote-ref-8)
9. Beschreibt Algorithmen, welche Gitternetzgeometrien [↑](#footnote-ref-9)