

Uebung 5 Dokumentation

Jonas Schwammberger

5. Semester (HS 2013)

Inhaltsverzeichnis

1	Einleitung	1
2	Algorithmen	1
2.1	Algorithmus für die Konvexe Hülle	1
2.1.1	Modifikation	1
2.2	Algorithmus für die Berechnung des minimalen Rechtecks	1
2.2.1	Implementierung	1

1 Einleitung

2 Algorithmus für die konvexe Hülle

Für die Berechnung der konvexen Hülle standen zwei Algorithmen zur Verfügung. Der Algorithmus von Graham und von Jarvis March. Welcher Algorithmus ($O(n \log n)$ versus $O(n^2)$) besser ist, hängt vom Eingabeverhalten des Benutzers ab. Da ich zum durchschnittlichen Verhalten keine Daten habe wurde Graham ausgewählt. Der Algorithmus von Graham wurde aus Interesse von mir selber implementiert und modifiziert.

2.1 Modifikation

Die Modifikation beschränkt sich darauf, dass nach dem Abbrechen der Hauptschleife nochmals eine `isConvex()` Prüfung mit Punkt P_0 , P_N und P_{N-1} . Im Falle das P_N auf der Strecke zwischen P_0 und P_{N-1} ist P_N nicht in der minimalen konvexen Hülle. Dies prüft der Algorithmus von Graham nicht, deshalb wurde diese Modifikation vorgenommen.

3 Algorithmus für die Berechnung des minimalen Rechtecks

Gegeben sei die konvexe Hülle einer Punktmenge, von der das minimale Rechteck berechnet werden soll.

1. Erstelle ein achsenparalleles Rechteck um die konvexe Hülle. Jede Seite berührt die konvexe Hülle in einem Punkt.
2. Drehe alle Seiten des Rechtecks um den Berührungspunkt.
3. Wenn sich eine Seite mit einem weiteren Punkt berührt, ist das eine mögliche Lösung.
4. Fahre mit Schritt 2 weiter, jedoch soll sich die Seite nun um den neuen Berührungspunkt drehen.

Das wird durchgeführt, bis der totale Winkel, mit dem das Rechteck gedreht wurde, mindestens 90° ist.

Der Algorithmus braucht mindestens vier Punkte, deshalb muss beim Input von drei Punkten eine Sonderbehandlung durchgeführt werden.

3.1 Sonderbehandlung bei drei Punkten

Bei drei Punkten muss man nur das minimale Rechteck eines Dreiecks finden. Das minimale Rechteck liegt immer mit einer Seite an der längsten Seite des Dreiecks.

3.2 Implementation

3.2.1 ModifiedGraham

3.2.2 SmallestRectangle

3.2.3 Line

3.2.4 Vector

3.3 Fazit

Die totale Laufzeit des Algorithmus setzt sich aus zwei Laufzeiten zusammen

1. Laufzeit für das Finden der konvexen Hülle.

2. Laufzeit für das finden des minimalen Rechtecks.

Die Laufzeit vom Graham-Algorithmus ist bekannt, $O(n \log(n))$. Die Laufzeit für das minimale Rechteck setzt sich wie folgt zusammen:

- Nächsten Berührungspunkt finden. Der nächst mögliche Berührungspunkt einer Geraden ist der Index (in der konvexen Hülle) des momentanen Berührungspunktes $+1$ (-1 wenn man im Gegenuhrzeigersinn dreht). Das ergibt eine Laufzeit von $O(1)$, falls man die 4 Indizes speichert.
- Die vier Winkel für die nächsten Berührungspunkte berechnen, den Kleinsten auswählen und die Geraden drehen. Es bleiben immer vier Geraden deshalb $O(1)$
- Flächenberechnung braucht ebenfalls keine Schleife, deshalb $O(1)$

Das wird solange ausgeführt, bis der totale Winkel 90° ist. Das ist genau dann der Fall, wenn jeder Punkt in der konvexen Hülle genau ein Mal ein Berührungspunkt einer Geraden war. Das ergibt eine Laufzeit von $O(m)$ (m sei die Anzahl der Punkte der konvexen Hülle).

Der grösste Nachteil dieser Implementierung ist, dass für die Drehung Floating Point Datentypen verwendet werden müssen. Bei der Konvertierung von Floating Point zu Integer kommen oft Fehler von ± 1 vor, welches auf der Graphik sofort sichtbar ist.

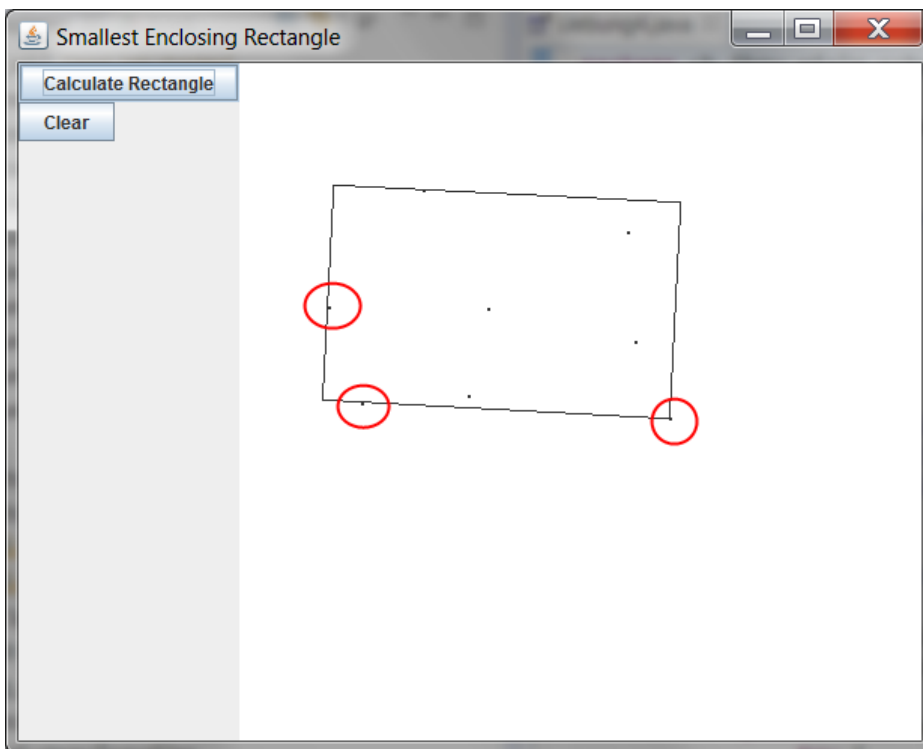


Abbildung 1: Fehler durch Floating Point Rundung

Ein vielleicht genauerer Ansatz wäre, eine Gerade an eine Seite der konvexen Hülle zu legen und daraus das minimale Dreieck berechnen. Wenn man das für die Hälfte aller Seiten durchführt, erhält man ebenfalls das minimale Rechteck. Jedoch muss man alleine für die Parallele den Punkt mit dem grössten Abstand zur Geraden finden, welches einen Aufwand von $O(m)$ pro Seite bedeutet (m sei die Anzahl der Punkte der konvexen Hülle). Wenn man den Algorithmus für ca $m/2$ Punkte ausführt, ergibt das eine Laufzeit von $O(m^2)$, was ebenfalls nicht erwünschenswert ist.