HOCHSCHULE OSNABRÜCK

UNIVERSITY OF APPLIED SCIENCES

Institut für Duale Studiengänge

Ausarbeitung im Studiengang Wirtschaftsinformatik

Anwendung zur Durchführung eines Punkt-in-Polygon-Tests in C

Eingereicht von:	Jonas Jansen geb.: 08.03.1998 in Lingen (Ems) Höfener Straße 9, 49716 Meppen
Matrikelnummer:	750295
Studiengruppe:	16 DWF 1
Betreuer:	Dr. Alexander Kling
Modul:	Technische Programmierung

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Aufgabenstellung

Es soll ein Programm in der Programmiersprache C geschrieben werden, das folgende Aufgabe löst:

Gegeben sei ein fiktives Verteilungsnetz, welches aus mindestens 3 Verteilungszentren, bzw. Standorten besteht, die durch Wege miteinander verbunden sind. Es ist zu prüfen ob sich ein fiktiver Kunde innerhalb dieses Netzes befindet. Sowohl der Standort des Kunden als auch die Standorte der Netzknoten werden dabei durch Punkte im zweidimensionalen Raum dargestellt, die durch X- und Y-Koordinaten definiert sind.

Mathematisch betrachtet ergibt sich aus dem Netz damit ein zweidimensionales Polygon, wobei die Aufgabe vorgibt, dass keine Selbstüberschneidungen vorliegen dürfen. Demnach soll überprüft werden ob ein Punkt in einem beliebigen (validen) Polygon liegt.

Ziel dieser Ausarbeitung ist es dabei, die grundlegende Struktur und Vorgehensweise des Programms zu erläutern, um eine zusätzliche Dokumentation neben der Dokumentation im Quellcode zu bieten.

2 Projektstruktur

Im Folgenden soll der Aufbau des Quellcodes erläutert werden.

Die zu Grunde liegende Struktur und alle zugehörigen Funktionen, die das Verteilungsnetz abbilden sollen, sind in „net.h“ und „net.c“ definiert. Die wichtigsten Elemente dieser Struktur sind zwei verkettete Listen („pointList“ und „lineList“), wobei die eine Liste alle Punkte („point“), bzw. Knoten des Netzes enthält, während die andere Liste die Linien („line“) enthält, welche die Seiten des Polygons definieren. Die Linien sind wiederum ebenfalls durch zwei Punkte und die daraus errechneten Variablen der Geradengleichung definiert. Um verkettete Listen bilden zu können, sind die Inhalte der Listen in die jeweiligen Listenelemente integriert, also „pointItem“ und „lineItem“.

Die Datei „test.c“ enthält eine Beispielanwendung, die die beschriebenen Strukturen nutzt, um die vorliegende Aufgabenstellung exemplarisch zu lösen. Dabei wird ein Polygon festgelegt welches in dieser Ausarbeitung als Beispiel verwendet wird und in Abbildung 1 dargestellt wird.

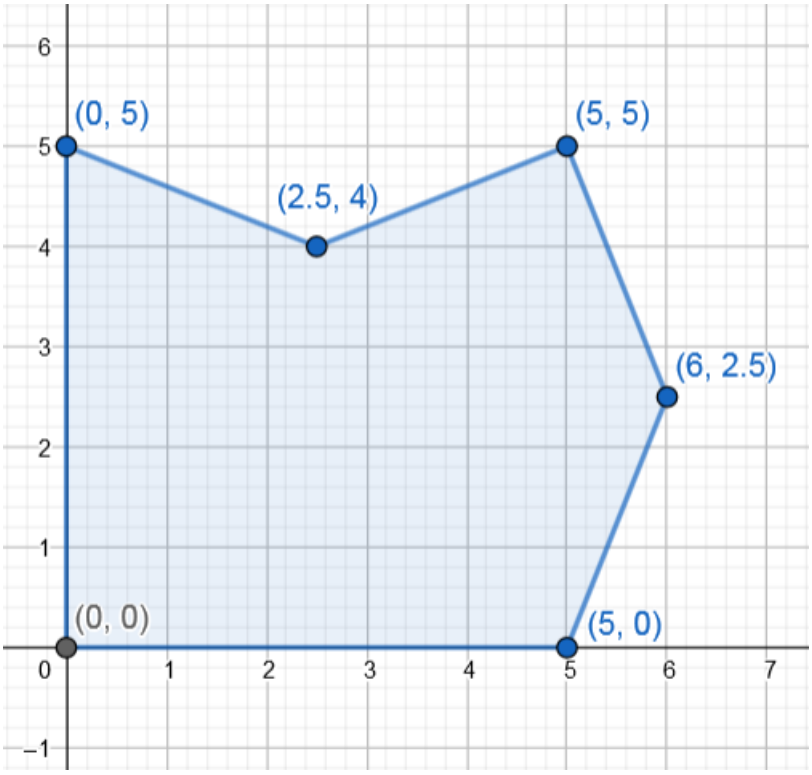

Abbildung 1: Beispielpolygon

Auf Basis der beschriebenen Struktur soll im Folgenden der Ablauf Programmablauf zur Lösung der Aufgabe näher erläutert werden.

3 Erläuterung des Programmablaufs

Die Lösung des Problems lässt sich in folgende Schritte unterteilen, die in diesem Kapitel näher erläutert werden:

1. Erstellung des Netzes
2. Prüfung des Netzes
3. Punkt-in-Polygon-Test

3.1 Erstellung des Netzes

Die Erstellung des Netzes geschieht primär über die Definition von Punkten, die über die Funktion *appendPoint* an die Liste der Netzknoten angehängt werden. Hierbei ist ebenfalls die Reihenfolge relevant, in der die Punkte hinzugefügt werden, da diese auch die spätere Anordnung der Seiten des Polygons festlegen.

Des Weiteren werden durch diese Funktion beim Hinzufügen neuer Punkte zudem einige weitere Attribute des Netzes aktualisiert: Zum einen ist dies die Gesamtknotenanzahl *numOfNodes* und zum anderen sind es die Werte *maxX, maxY, minX* und *minY,* die die äußersten Koordinaten des Netzes widerspiegeln.

Nachdem beliebig viele Punkte hinzugefügt wurden, können über die Funktion *generateLines* die Linien erzeugt werden, die die Seiten des Polygons darstellen. In dieser Funktion werden die Linien aus jeweils zwei aufeinander folgenden Punkten und einer daraus errechneten Geradengleichung erzeugt und an die Liste der Linien angehängt. Nachdem die letzte Linie aus dem letzten und dem ersten Punkt des Netzes erstellt wurde, wird zudem die verkettete Liste der Linien über die Funktion *closeLineList* zu einer Ringliste geschlossen. Danach ist die Erstellung des Netzes abgeschlossen.

3.2 Prüfung des Netzes

Die Prüfung des Netzes erfolgt über Funktion *isValidNet*. In dieser Funktion wird zunächst sichergestellt, ob das Netz die Mindestanzahl von 3 Knoten enthält, danach wird geprüft ob das Netz Selbstüberschneidungen enthält, letzteres soll im Folgenden näher erläutert werden.

Um mögliche Selbstüberschneidungen zu ermitteln wird die Funktion *haveIntersection* verwendet, wobei jede Linie mit jeder anderen Linien verglichen wird, die nicht bereits über einen gemeinsamen Knoten verbunden ist. In dieser Funktion wird geprüft ob zwei Linien einen Schnittpunkt haben, indem die Position der Punkte der ersten Linie mit der Gerade durch die Punkte der anderen Linie verglichen wird und umgekehrt.

Dies soll auch in Abbildung 2 verdeutlicht werden. Hier sieht man, dass die Punkte F und E ober- und unterhalb der Gerade durch A und B liegen, während gleichzeitig die Punkte A und B ober- und unterhalb der Gerade durch F und E liegen, damit liegt ein Schnittpunkt vor. Für das Verhältnis der Linien AB und CD gilt hingegen nur eine dieser Voraussetzungen, damit liegt hier kein Schnittpunkt vor

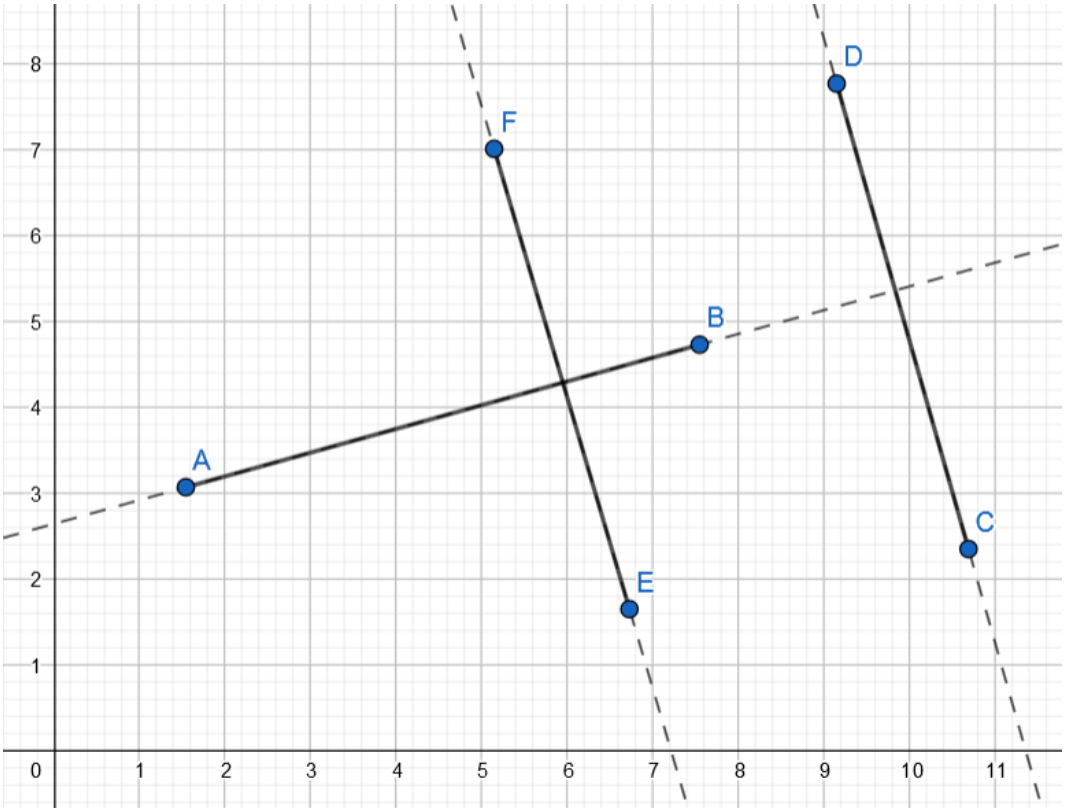

Abbildung 2: Ermittlung von Schnittstellen

Insofern keine Selbstüberschneidungen festgestellt werden, ist das Netz valide und kann im nächsten Schritt für einen Punkt-in-Polygon-Test verwendet werden.

3.3 Punkt-in-Polygon-Test

Im letzten Schritt wird durch die Funktion *isWithinNet* ermittelt, ob ein beliebiger Punkt, welcher den Standort des Kunden widerspiegelt innerhalb eines Polygons liegt.

Dabei wird der Punkt zunächst mit den äußersten Koordinaten des Netzes verglichen, um Punkte, die weit außerhalb liegen bereits frühzeitig auszuschließen (Vgl. Abb. 3 Punkt G). Falls ein Punkt hierdurch nicht bereits ausgeschlossen wird, folgt

der nächste Schritt, in dem eine Linie zwischen diesem Punkt und einem Punkt, der außerhalb der äußersten Koordinaten liegt erzeugt wird. Im Normalfall liegt der Punkt innerhalb des Polygons, wenn diese Linie eine ungerade Anzahl von Schnittpunkten mit den Seiten des Polygons hat. Ist es eine gerade Anzahl, so liegt der Punkt außerhalb des Polygons (Vgl. Abb. 3 Punkt H).

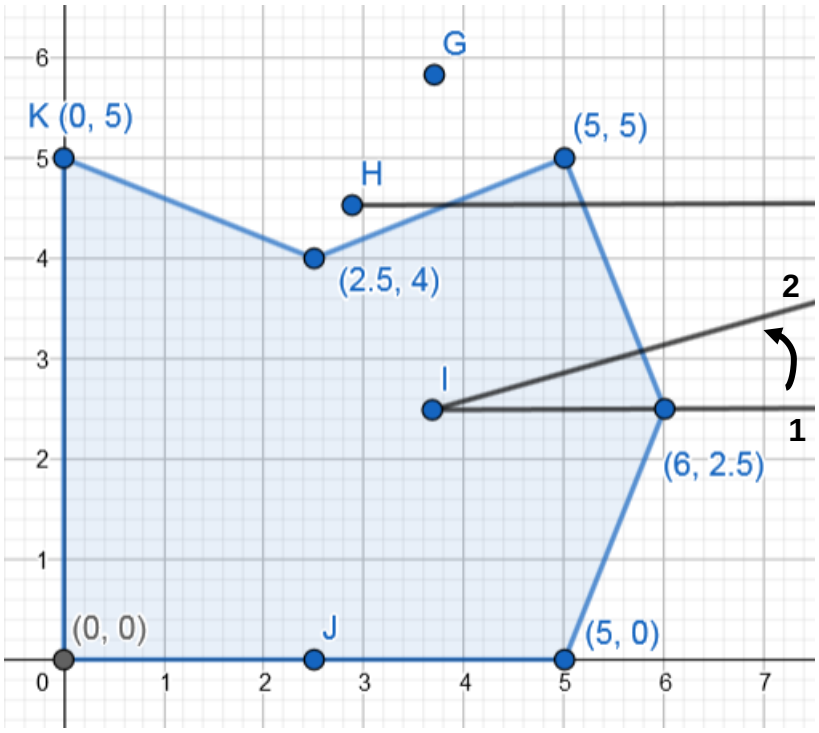

Abbildung 3: Punkt-in-Polygon-Test

Hierbei müssen jedoch gewisse Sonderfälle beachtet werden: Der erste Fall ist, dass der Punkt die gleichen Koordinaten besitzt wie einer der Netzknoten, da hier zwei Schnittpunkte festgestellt werden. Dies wird zu Beginn durch einen Vergleich aller Netzknoten mit dem Punkt abgefangen (Vgl. Abb. 3 Punkt K).

Der nächste Fall ist, dass der Punkt auf einer Linie des Netzes liegt. Auch hier wird ein Schnittpunkt festgestellt der je nach berechneter Linie das Ergebnis verfälschen kann. Aus diesem Grund wird der Punkt als innerhalb des Polygons angesehen, wenn der betrachtete Punkt und der Schnittpunkt der Linie die gleichen Koordinaten besitzen (Vgl. Abb. 3 Punkt J).

Der letzte Fall ist, dass die berechnete Linie einen der Netzknoten schneidet, wobei auch hier ein doppelter Schnittpunkt festgestellt wird, der je nach Position des

Punktes das Ergebnis verfälschen kann.[1] Falls dieser Fall eintritt wird eine neue Linie gebildet, die zu einem anderen Punkt außerhalb des Polygons führt. Dieser Schritt wird so oft wiederholt bis eine Linie ohne Schnittpunkt mit einem Netzknoten ermittelt wurde (Vgl. Abb. 3 Punkt I).

Durch Abfangen der Sonderfälle kann somit eindeutig festgestellt werden, ob der Punkt innerhalb des Polygons liegt oder nicht.

[1] Vgl. https://stackoverflow.com/questions/14130742/ray-through-vertex-special-case-when-detecting-point-in-polygon