HOCHSCHULE **TRIER**
Trier University of Applied Sciences
Informatik - Computer Science

Alpha zu Mesh Tool

Jeremias Boos

Hausarbeit zur Vorlesung Tool- und Pluginprogrammierung

Betreuer: Prof. Dr.-Ing. Christof Rezk-Salama

Trier, 29. August 2015

Inhaltsverzeichnis

1

Motivation

Wenn Sprites mit großen transparenten Flächen in Spielen verwendet werden, wird viel Rechenzeit für die Berechnung und Verwerfung von absolut transparenten Pixeln aufgewendet. In solchen Fällen kann es besser sein, anstelle eines einfachen Quadrates einen angepassten Mesh zur Textur zu verwenden. Außerdem kann der Mesh als Kollisionsgeometrie benutzt werden, wenn per Pixelkollision zu aufwendig ist.

2

Ergebnis

Im Kern besteht das Programm aus 3 Algorithmen.

2.1 Randdetektion

Das Bild wird Zeilenweise durchlaufen, bis ein Pixel gefunden wurde, der über dem Schwellwert für das Alpha liegt. Sobald ein entsprechender Pixel gefunden wurde, wird im Uhrzeigersinn nach einem Pixel gesucht, der Transparent ist und dessen rechter Nachbar im Alpha über dem Schwellwert liegt. Sobald der Start erreicht wird, ist der Pfad vervollständigt und weitere Bilder werden gesucht.

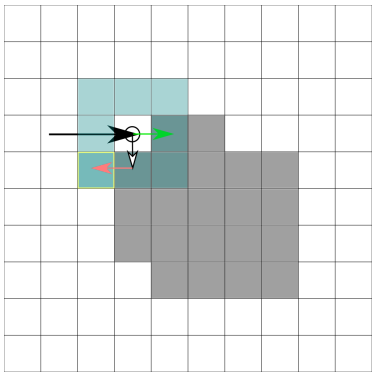

Abb. 2.1: Rand Anfang
Schwarzer Pfeil: Pixel Scan
Grüner Pfeil: Pixel über Schwellwert
Schwarz-weißer Pfeil: Potenzieller Pfad
Roter Pfeil: Rechts vom Potenziellen Pixel
Grüne Felder: Punkte eines potenziellen Pfads
Grünes Feld mit gelber Umrandung: Nachbar Scan

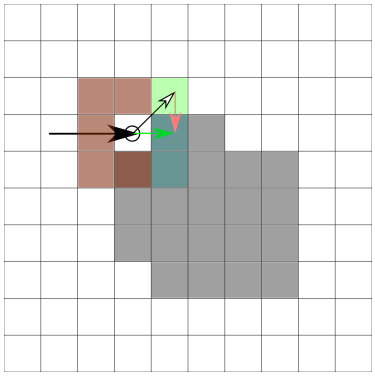

Abb. 2.2: Gefundener nächster Punkt
Roter Felder: Ausgeschlossene Kandidaten
Hellgrünes Feld: Gefundener Rand

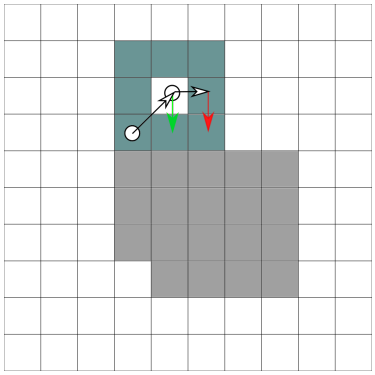

Abb. 2.3: nächster Punkt

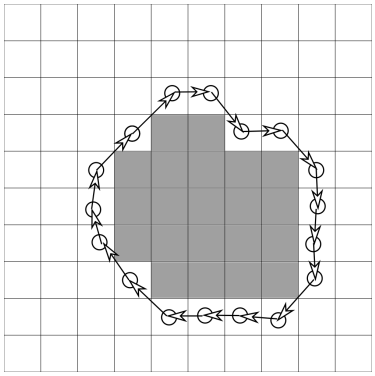

Abb. 2.4: Rand vollständig

2.2 Ramer-Douglas-Peucker-Algorithmus

Der Pfad besteht nun aus allen Pixeln, die den Rand darstellen. Da dies auch einige Geraden beinhaltet, lässt sich der Rand gut vereinfachen, um später in der Triangulierung Dreiecke zu sparen.
Zur Vereinfachung wird der "'Ramer-Douglas-Peucker-Algorithmus"'verwendet. Dieser lässt sich leicht implementieren und arbeitet zuverlässig.

Der Algorithmus sucht rekursiv den am weitesten entfernten Punkt zwischen Anfang und Ende. Wenn der Punkt von der Linie weiter als der Schwellwert entfernt ist, wird eine Spaltung der Linie durchgeführt. Wenn das Maximum innerhalb des Schwellwertes liegt, werden Anfangs- und Endpunkt direkt miteinander verbunden. [Wik]

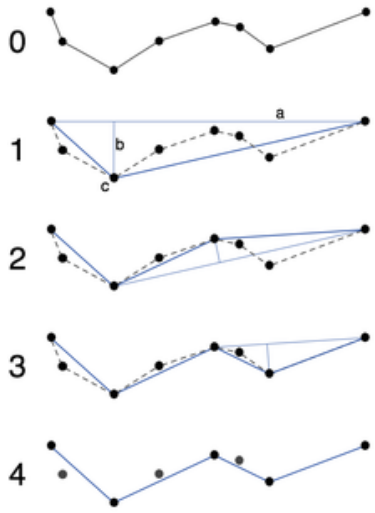

Abb. 2.5: Linienglättung nach dem Douglas-Peucker-Algorithmus

2.3 Ear clipping

Ear clipping ist ein sehr einfach umzusetzender Triangulierungsalgorithmus von Polygonen.
Die Idee ist, jedes Ohr vom Polygon abzuschneiden, bis das ganze Polygon in einzelne Dreiecke zerlegt ist. [Ebe] [del]

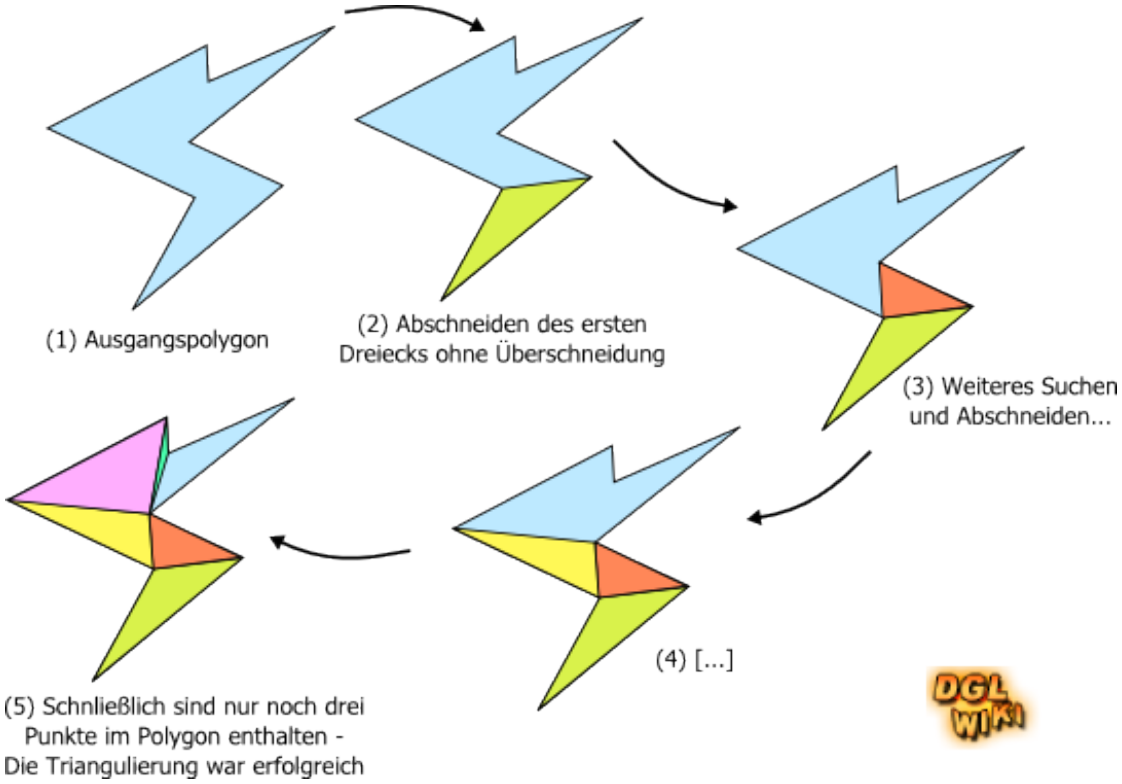

Abb. 2.6: Ablauf des Ear Clipping Verfahrens

3

Probleme und Ausblick

Der umgesetzte Trianjulierungsalgorithmus liegt in $O(n^2)$. Er kann durch den Monton-Polygon-Triangulierungalgorithmus ausgetauscht werden, wodurch die Triangulierung nur noch in $O(n * log(n))$ liegt.
Die Triangulierung könnte auch noch erweitert werden, um Löcher im Sprite zu beachten.
Eine weitere Verbesserungen wäre die Implementierung des Gauß-Algorithmus, um einen gröberen Rahmen um den Sprite zu Ziehen.
Abschließend könnte man noch eine Option bieten, die Polygone in konvexe Polygone zu zerlegen, um die generierten Meshes leichter in einer 2D Physik-Engine zu nutzten.

Literaturverzeichnis

del. *Ear Clipping Triangulierung.*
wiki.delphigl.com/index.php/Ear_Clipping_Triangulierung.

Ebe. Eberly, David: *Triangulation by Ear Clipping.*
www.geometrictools.com/Documentation/TriangulationByEarClipping.pdf.

Wik. *Wikipedia - Ramer-Douglas-Peucker algorithm.*
en.wikipedia.org/wiki/Ramer-Douglas-Peucker_algorithm.