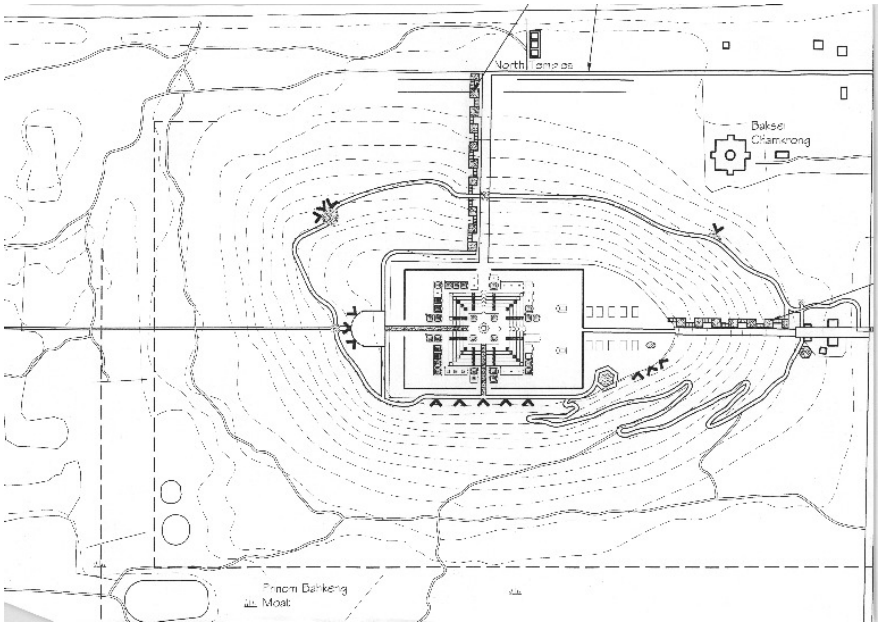
Terraindigitalisierung

Inhaltsverzeichnis

Praktikumsziel

- Erstellung einer Heightmap ausgehend von einer topographischen Karte

- Passendes Ausgabeformat für Jens Rannachers „TerrainViewer“



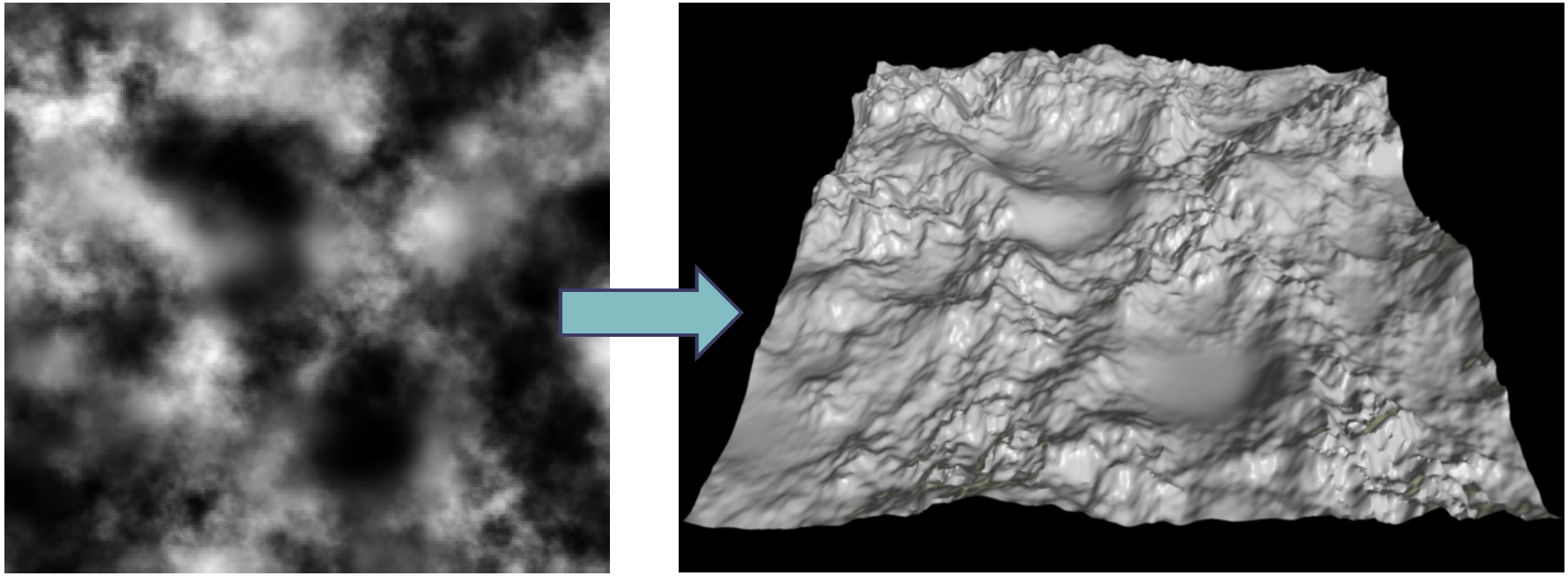
Theoretische Grundlagen

- Höhenfelder (Heightmaps)
- Triangulierung
- Punktlokalistation / lineare Höheninterpolation
- Delaunay-Triangulierung

Heightmaps (Höhenfelder)

- 2-dimensionales skalares Feld
- beschreibt die Geländeoberfläche eines geographischen Ortes
- jedes Feldelement speichert einen Höhenwert
- Visualisierung als Graustufenbild
- Grundlage für 3D-Visualisierung des Geländes

Beispiel für eine Heightmap



Triangulierung

- topologische Grundbegriffe:

  konvexe Menge, konvexe Hülle conv()
- 2D-Punktwolke: endliche Teilmenge $P \subseteq \mathbb{R}^2$
- Unter einer Triangulierung T(P) einer 2D-Punktwolke $P \subseteq \mathbb{R}^2$ versteht man eine Zerlegung von conv(P) in Dreiecke derart, dass die Eckpunkte der Dreiecke genau die Punkte aus P sind.

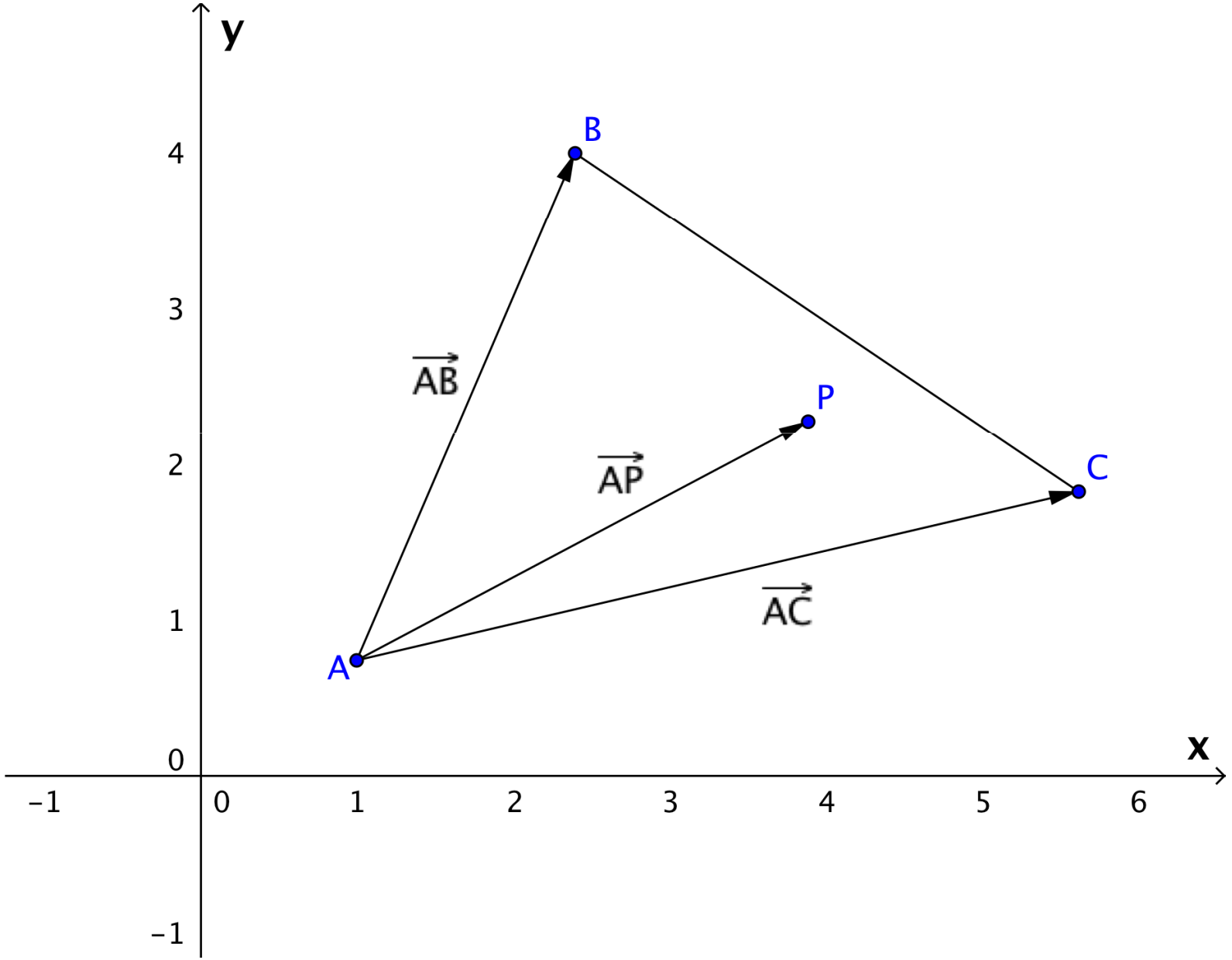
Punktlokalisation

- Gegeben: Dreieck Δ (ABC) und $P \in \mathbb{R}^2$
- Dann gilt:

$$P \in \triangle(ABC) \Leftrightarrow \overrightarrow{AP} = r\overrightarrow{AB} + s\overrightarrow{AC} \text{ mit } r, s \geq 0, r + s \leq 1$$

- Lineare Höheninterpolation innerhalb des Dreieckes:

$$h(P) = h(\overrightarrow{OP}) = h(\overrightarrow{OA}) + r \cdot h(\overrightarrow{AB}) + s \cdot h(\overrightarrow{AC})$$


y
x
B
P
C
A
$\overrightarrow{AB}$
$\overrightarrow{AP}$
$\overrightarrow{AC}$
4
3
2
1
0
−1
−1
0
1
2
3
4
5
6

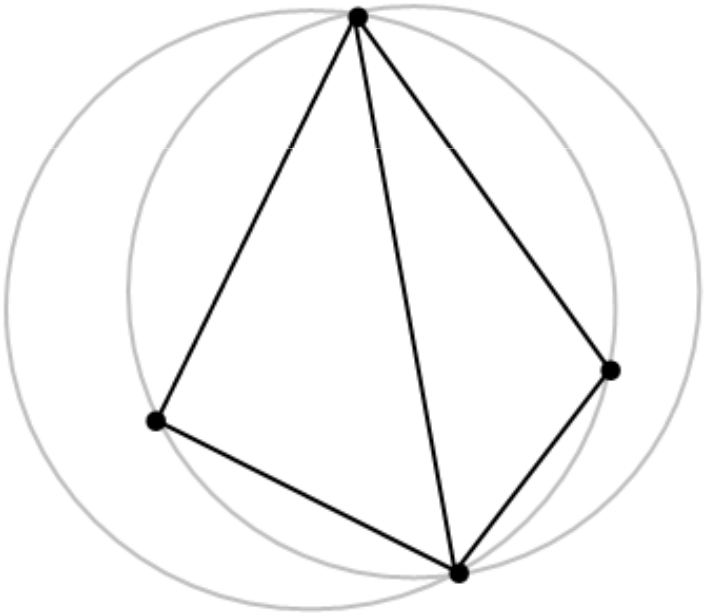
Delaunay – Triangulierung

- Umkreisbedingung: Ein Dreieck Δ aus T(P) erfüllt die Umkreisbedingung, wenn gilt:

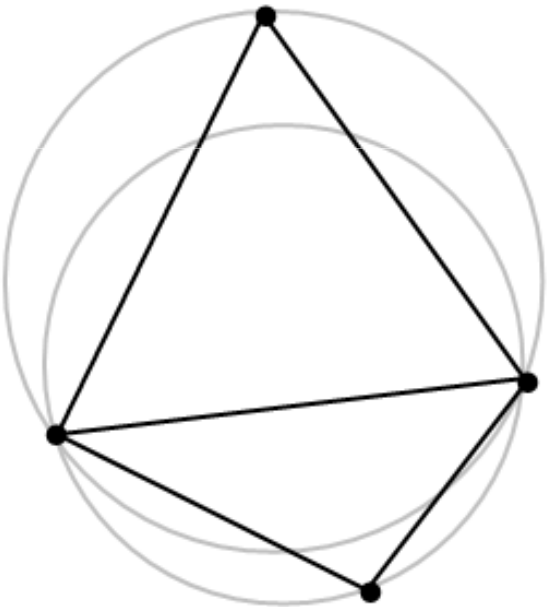
$$int(Umkreis(\triangle)) \cap P = \emptyset$$

- Eine Triangulierung T(P) heißt Delaunay-Triangulierung DT(P), wenn jedes Dreieck in DT(P) die Umkreisbedingung erfüllt.

Beispiel für eine Delaunay-Triangulierung



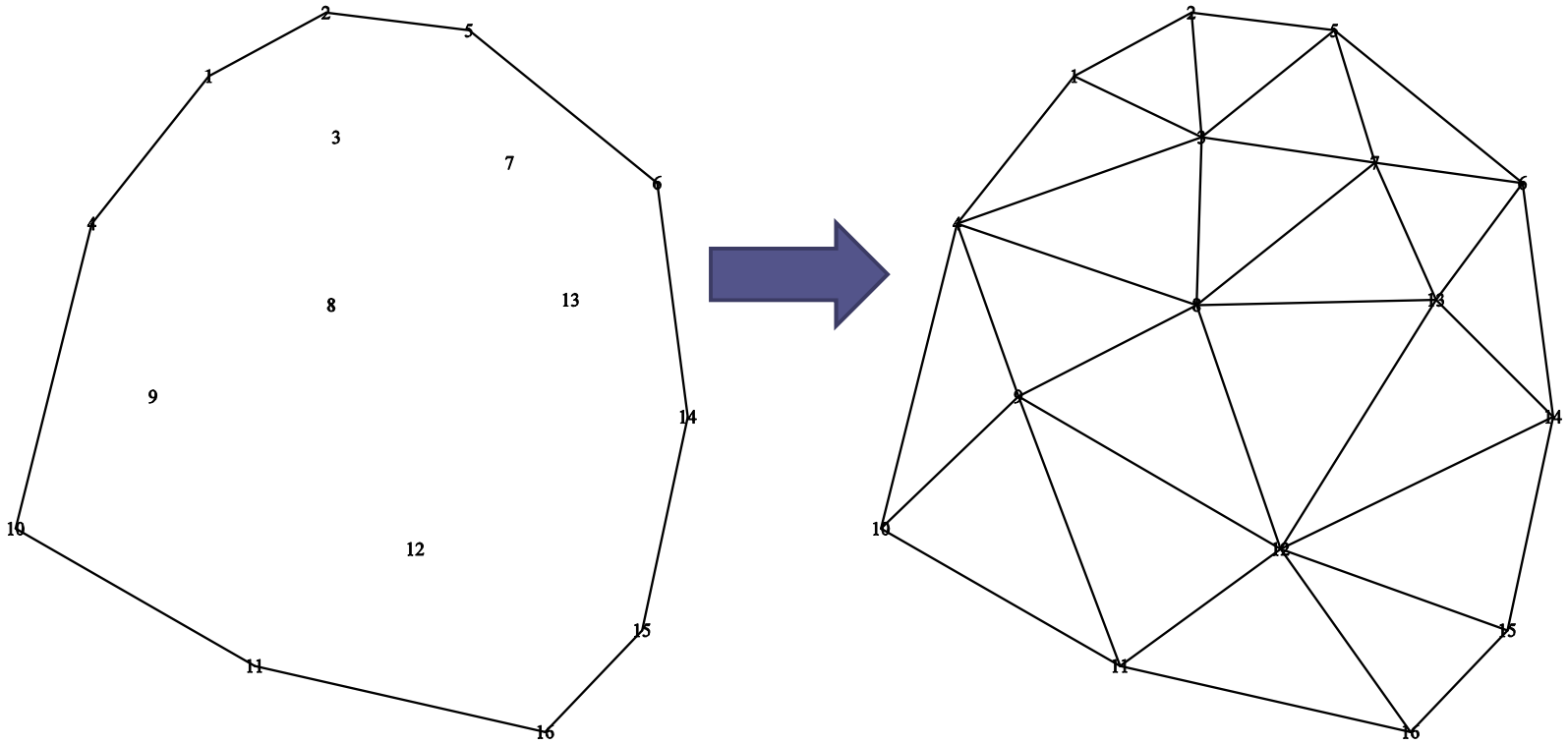
edge
flipping



Triangulierung, welche die Umkreisbedingung nicht erfüllt

Delaunay-Triangulierung

Delaunay – Triangulierung mehrere Punkte



Konvexe Hülle und Delaunay-Triangulierung

Eigenschaften der Delaunay - Triangulierung

- Maximierung des minimalen Winkels
- Minimierung des durchschnittlichen Abstands aller Punkte einer Dreiecksfläche vom Nächsten ihrer Eckpunkte
- Algorithmus in O(n log(n)) möglich

Die Software

Software Schritt 1: Erstellen der Input-Datei

- Ausgangspunkt ist eine topographische Karte
- Manuelles Erstellen eines TIFF-Bildes mit Höhenstützpunkten mit GIMP
- Definition der (relativen) Höhe durch Grauwerte (8 bit)
- Abspeichern im TIFF-Format

Software Schritt 2: Einlesen und Korrekturen

- Einlesen des Bildes in ein C-Programm
- Dazu Unterstützung durch die OpenSource-Bibliothek LibTIFF
- Transformieren in 16-bit-Graustufen-Bild
- Pixelkorrektur
- Setzen neuer Punkte am Rand

Original-Bild

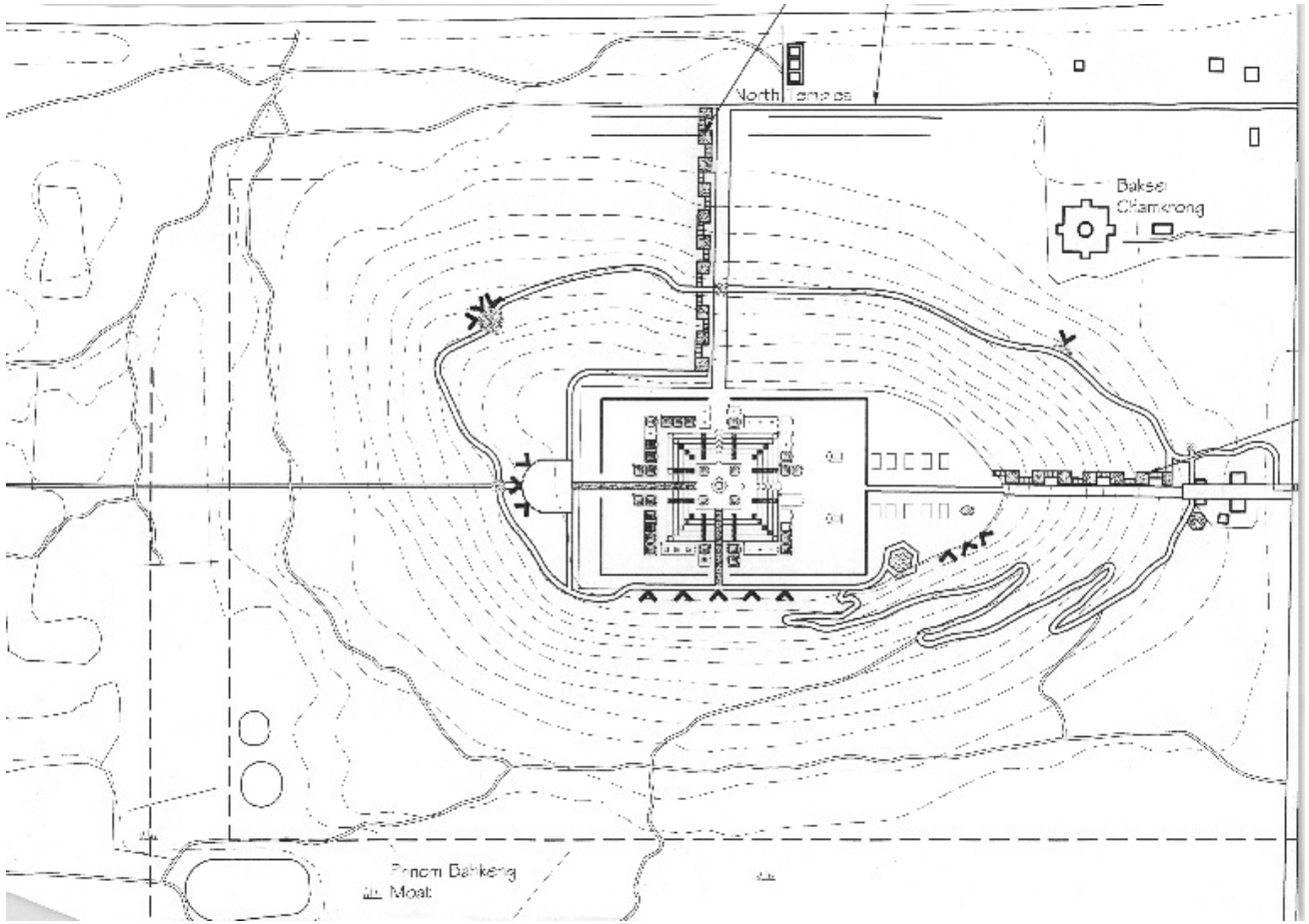


Bild nach Schritt 1 + 2

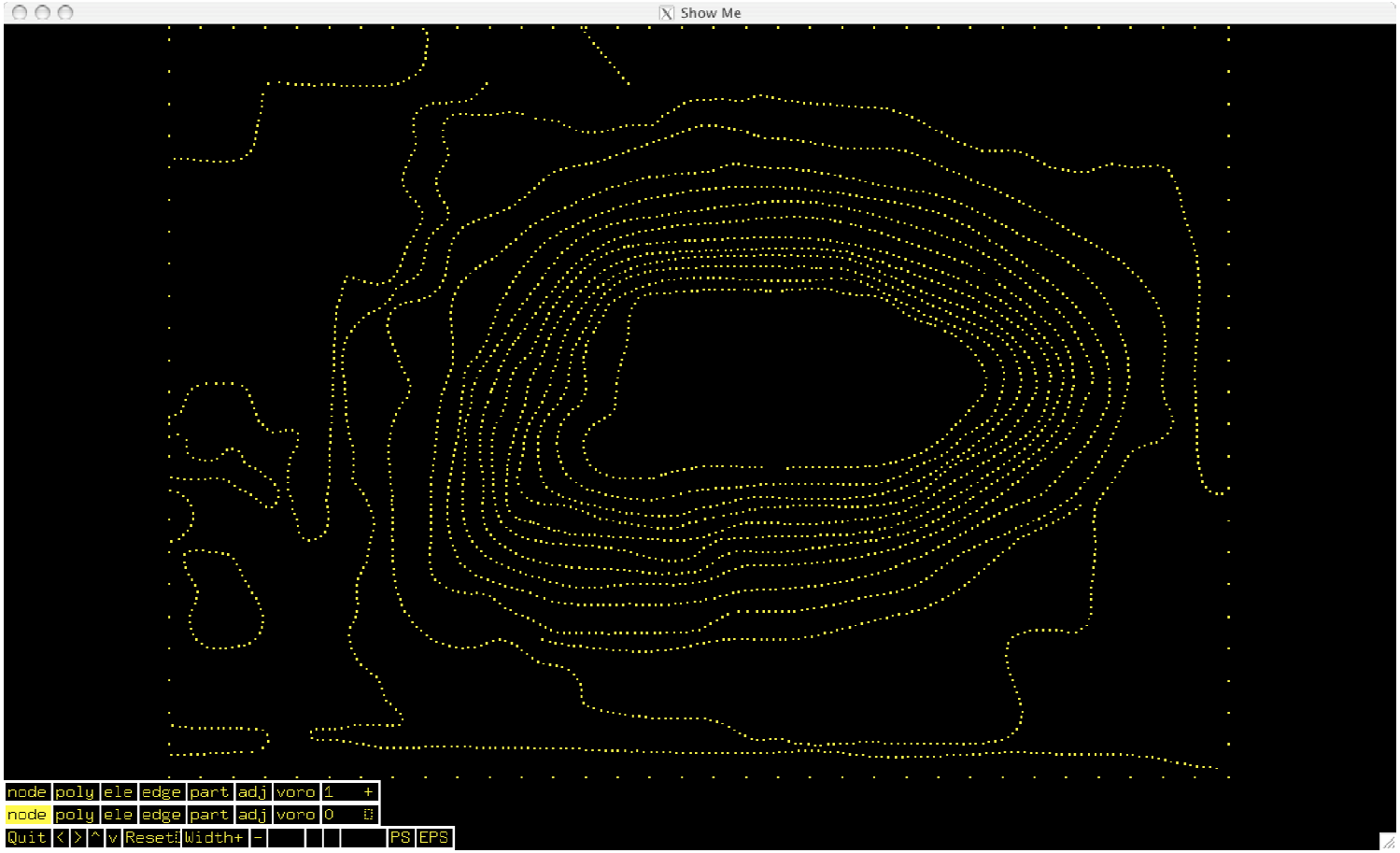
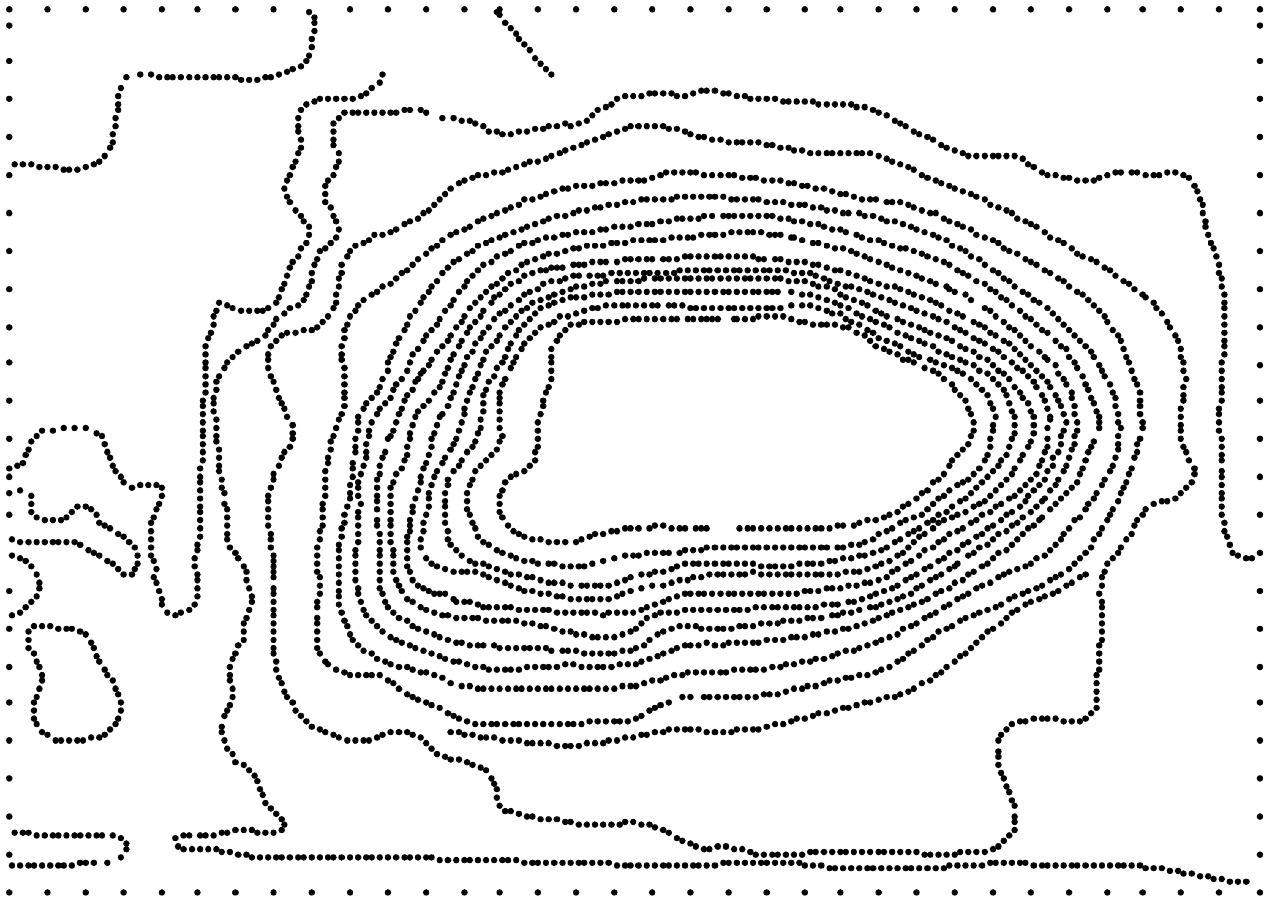


Bild nach Schritt 1 + 2



Software Schritt 3: Delaunay - Triangulierung

- Liste der im Bild markierten Punkte wird in eine Textdatei exportiert
- Programmaufruf „Triangle“ liefert dann die Delaunay – Triangulierung
- Abspeicherung dieser Triangulierung in Datenarray

Bild nach Schritt 3

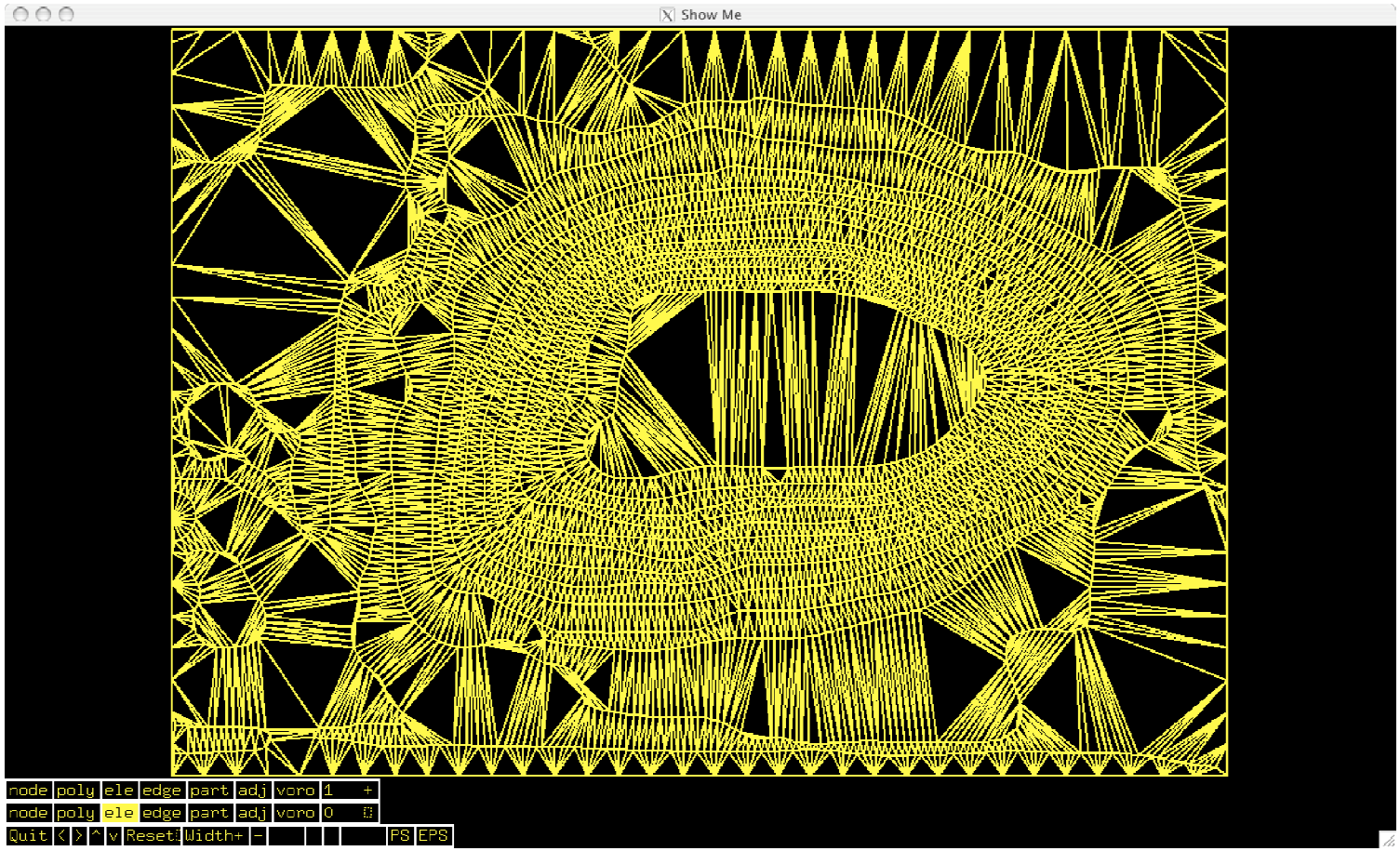
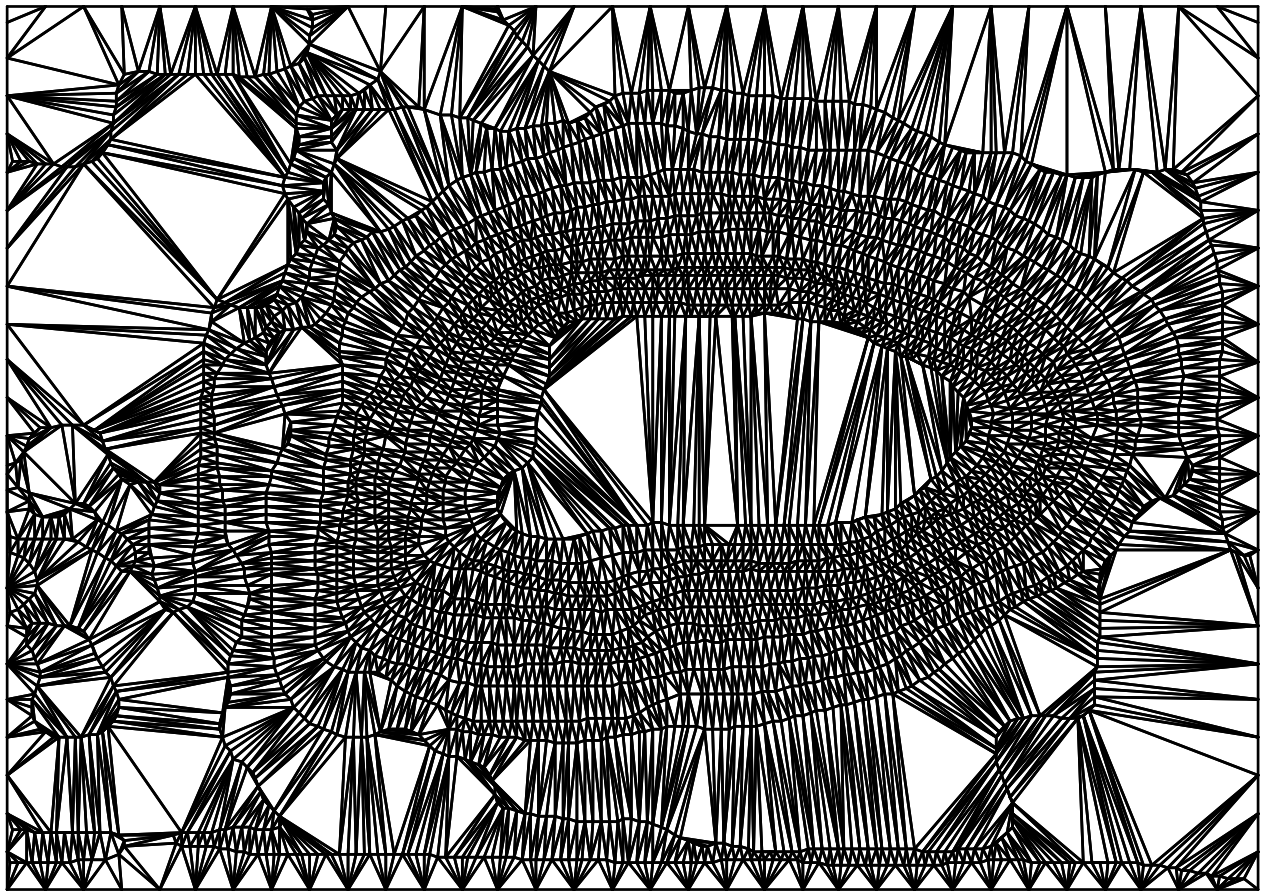


Bild nach Schritt 3

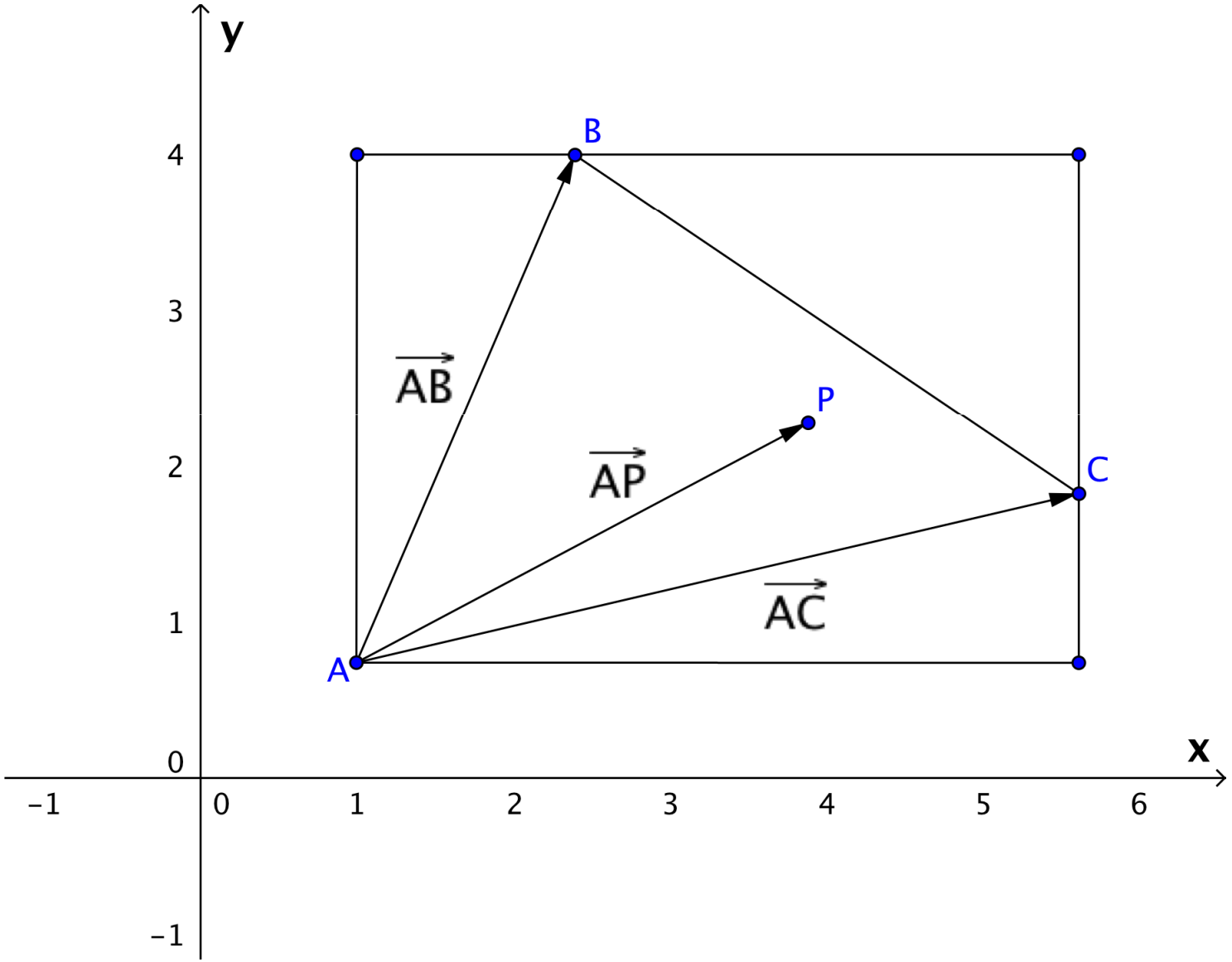


Software Schritt 4: Punktlokalisierung und Höheninterpolation

- Lokalisierung des umgebenden Dreiecks für jeden Pixel (Algorithmus gleich)
- Lineare Interpolation der Pixelhöhe aus den bekannten Höhen der Dreieckspunkte
- Summa summarum entsteht eine vollständige Heightmap

Erläuterungen zum Algorithmus „Punktlokalisierung“

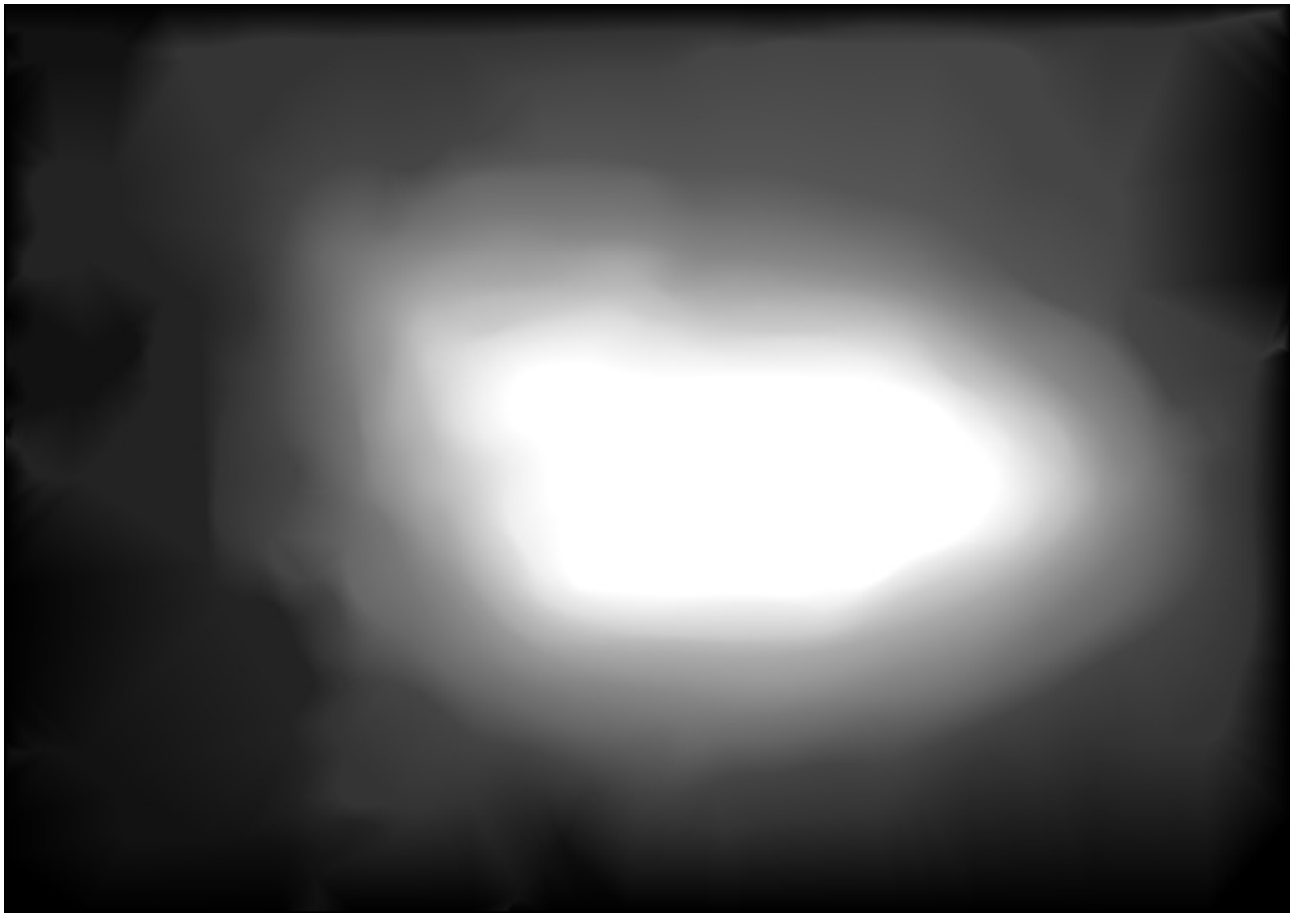
1. Überprüfen, ob Pixel im selben Dreieck liegt wie der Letzte (Trefferquote >95%)
2. Indexarray gibt Auskunft, welche Dreiecke überhaupt in Frage kommen (vorher einmalig erstellt)
3. Überprüfe in dieser Menge, ob Punkt in umgebendem Quadrat
4. Wenn im Quadrat, überprüfe ob im Dreieck


y
x
B
P
C
A
$\overrightarrow{AB}$
$\overrightarrow{AP}$
$\overrightarrow{AC}$
4
3
2
1
0
−1
−1
0
1
2
3
4
5
6

Software Schritt 5: Exportieren

- Ausgabe des Höhenfeldes als:
  - TIFF (16 bit Graustufen Bild)
  - ARC/INFO ASCII Grid mit Header
- Weitere Ausgabeformate denkbar, aber bisher nicht implementiert

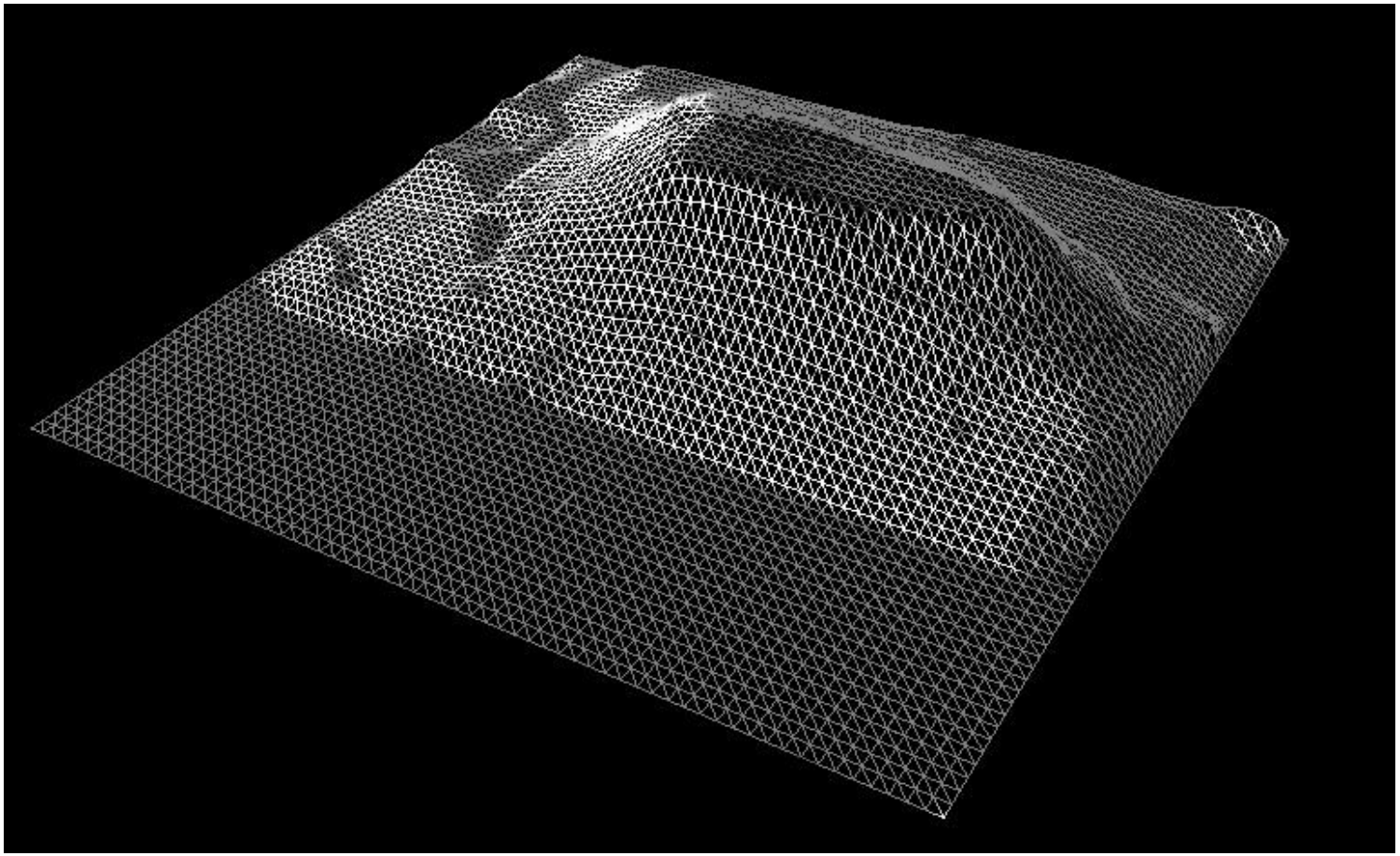
Ausgabeformat TIFF



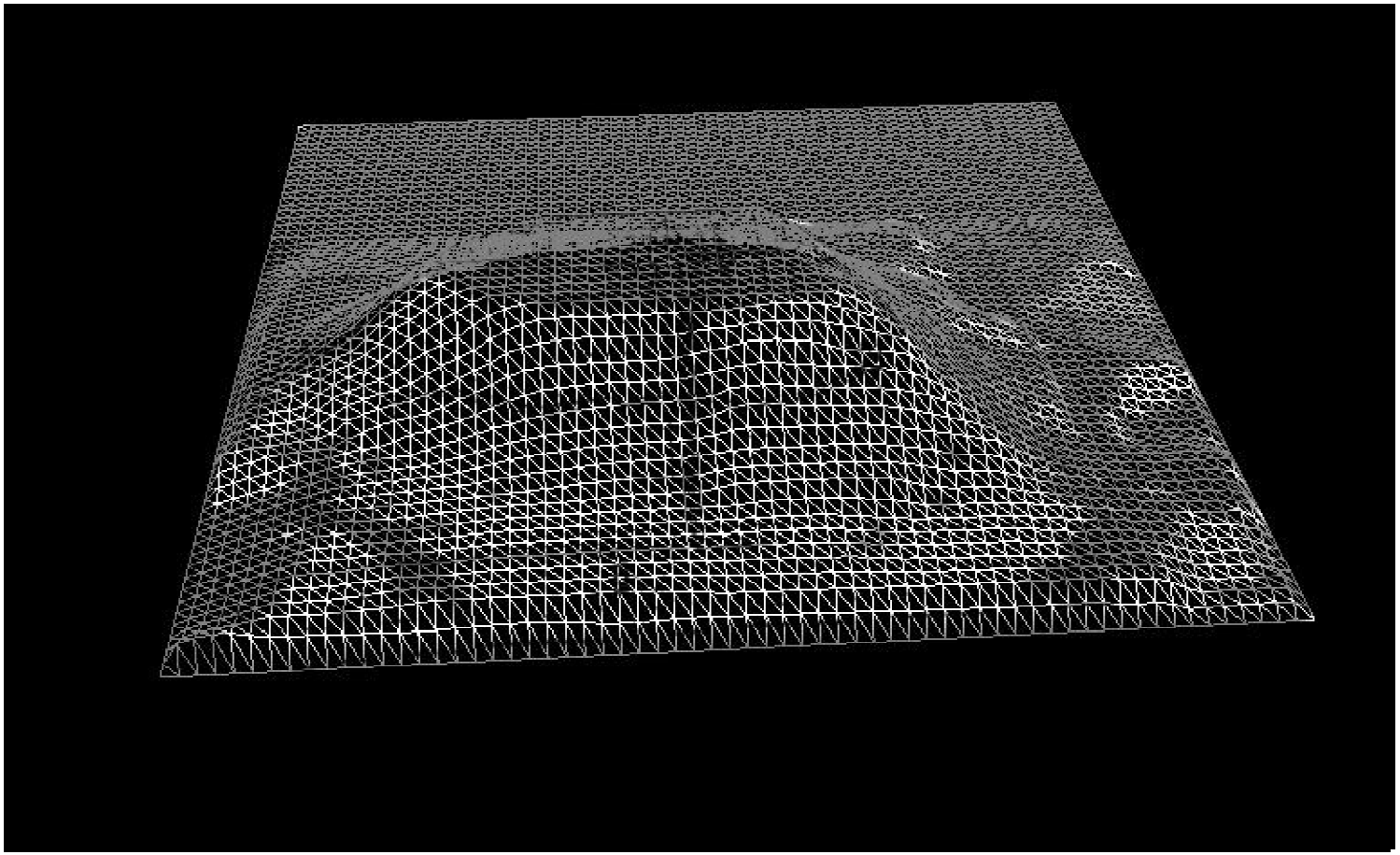
Ausgabeformat ARC/INFO ASCII

```
ncols 6614
nrows 4678
xllcorner 0
yllcorner 0
cellsize  0.000151
NODATA_value  65535
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
. 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
. 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
. 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 33  33  33  33  33  33  33  33  33
.   61  61  61  61  61  61  61  61  61
.   73  73  73  73  73  73  73  73  73
.   88  88  88  88  88  88  88  88  88
.   87  87  87  87  87  87  87  87  87
.   91  91  91  91  91  91  91  91  91
.   113 113 113 113 113 113 113 113 113
0 65  65  65  65  65  65  65  65  65
.   122 122 122 122 122 122 122 122 122
.   145 145 145 145 145 145 145 145 145
.   175 175 175 175 175 175 175 175 175
.   174 174 174 174 174 174 174 174 174
```

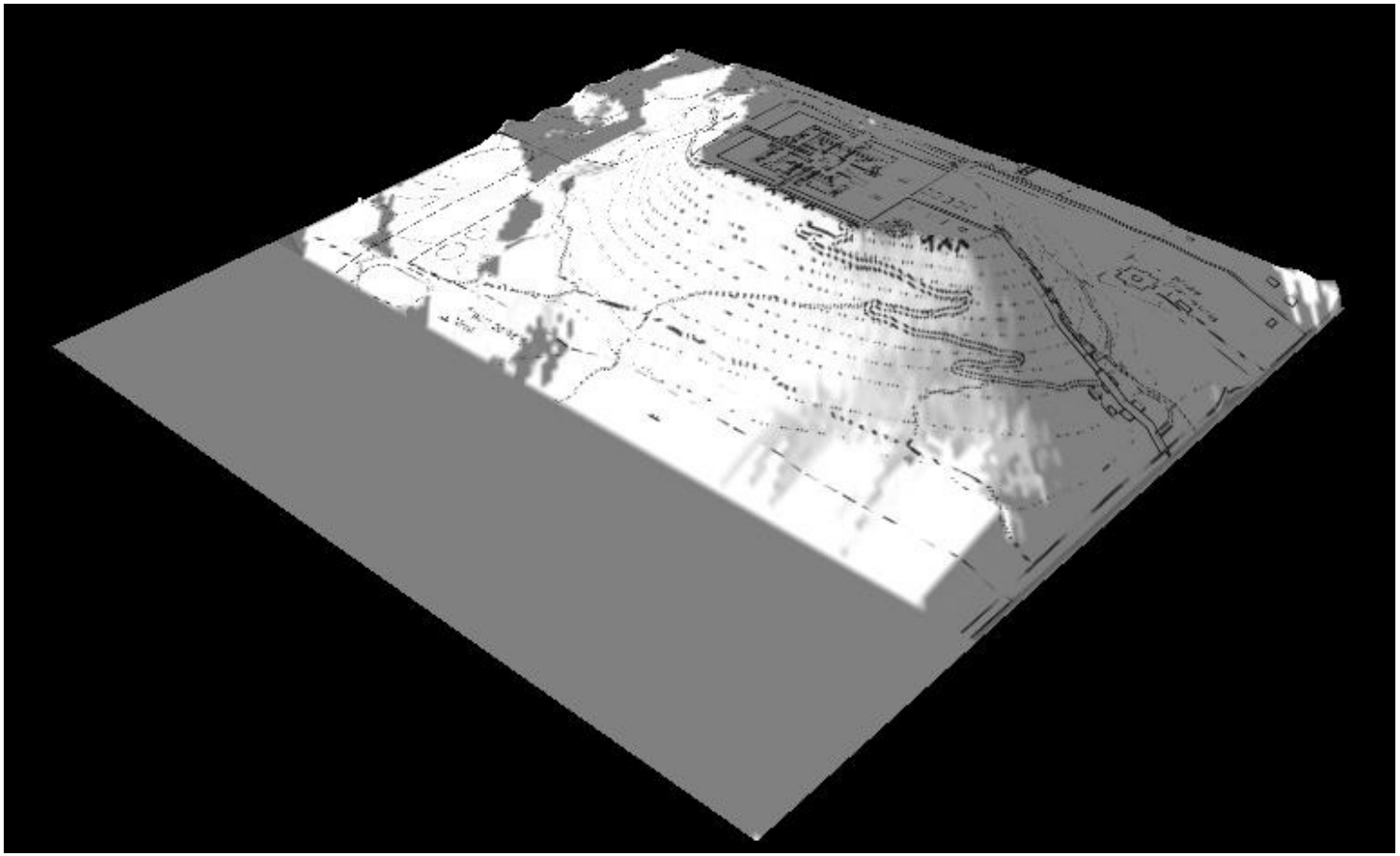
Visualisierung unserer Ergebnisse mit Jens Rannachers „TerrainViewer“



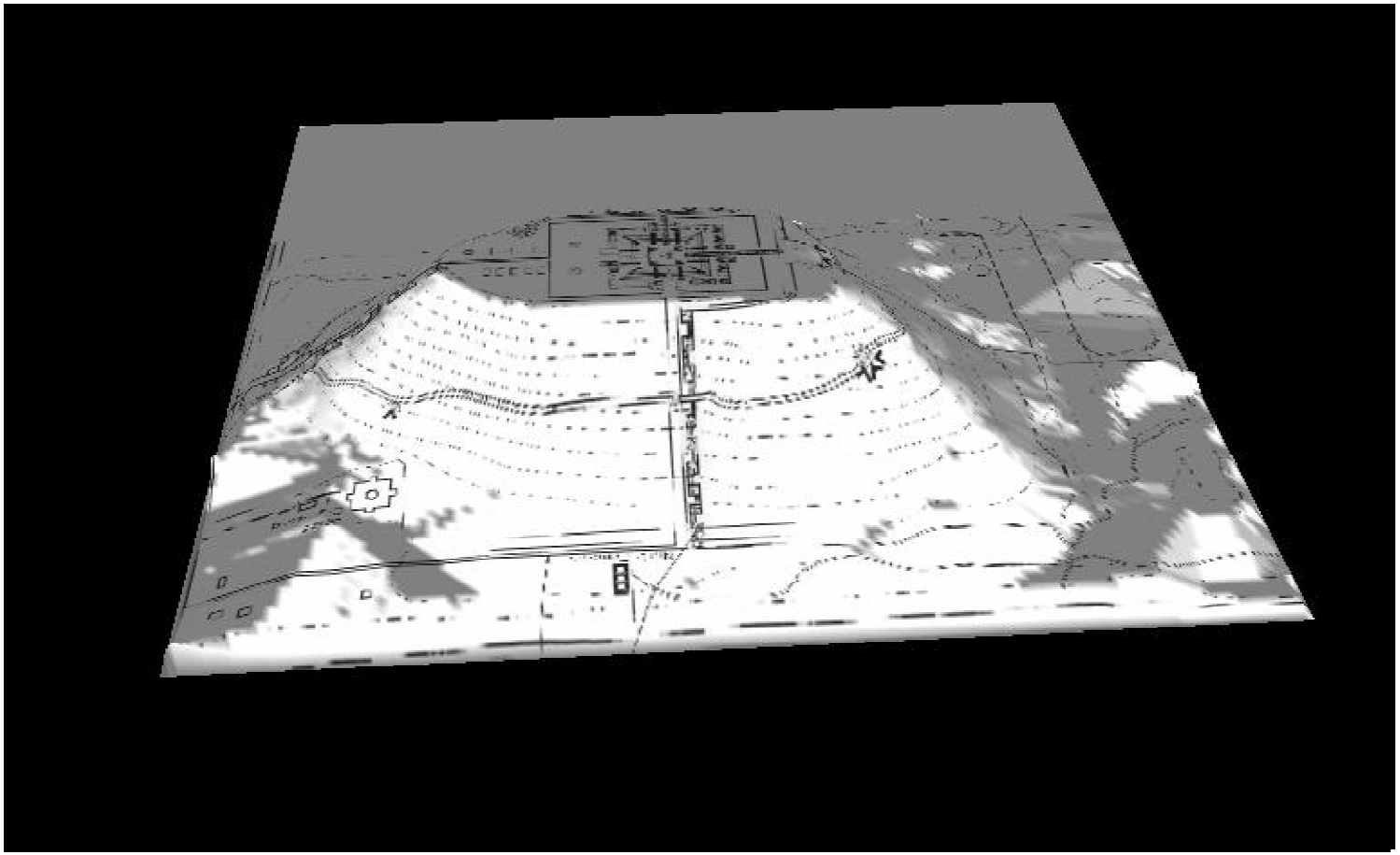
Visualisierung unserer Ergebnisse mit Jens Rannachers „TerrainViewer“



Visualisierung unserer Ergebnisse mit Jens Rannachers „TerrainViewer“



Visualisierung unserer Ergebnisse mit Jens Rannachers „TerrainViewer“



Ausblick

Ausblick

- Weitere Ausgabeformate denkbar
- Automatische Erkennung der topographischen Karte
- Algorithmus verschnellern

Ende

Kontakt:
Andreas Reifenberger (a.reifenberger@stud.uni-heidelberg.de)
Simon Rube (rube@stud.uni-heidelberg.de)

Homepage zu diesem Projekt:
http://pille.iwr.uni-heidelberg.de/~terrain03/