Hierarchical Domain Partitioning for Image Compression based on Normal Offsets

W. Van Aerschot, M. Jansen, E. Vanraes , A. Bultheel

K.U. Leuven

Sixth International Conference on
Curves and Surfaces
Avignon - FRANCE
June 29 - July 5, 2006

Images to compress



- Dominated by smooth contours
- Smoothly colored areas

Smooth functions

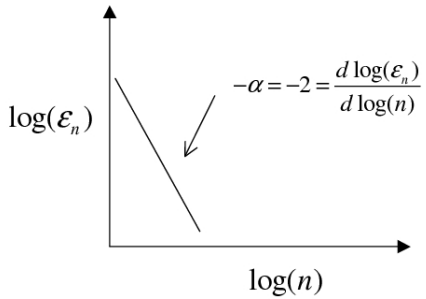
Consider functions $c(x) : [0,1] \rightarrow [0,1]$ such that:

- $c(x) \in \mathcal{C}^2$, and $\ddot{c}(x) \leq M$



These functions have an n terms approximation rate of $\alpha = 2$ using splines, wavelets , i.e. $\epsilon_n = n^{-\alpha}$

- rapid decay of error as n increases $\rightarrow$ good for compression: small n for good approximation

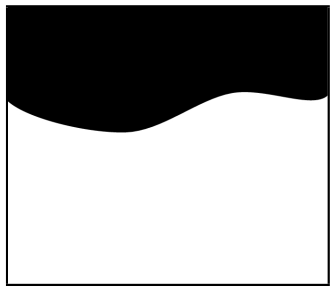


Horizon functions

Horizon Image $H(x, y)$

Definition $H(x, y) : [0, 1]^2 \rightarrow \{0, h\}$:

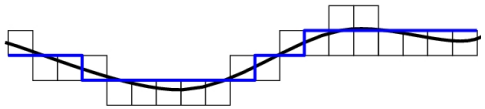
$$\begin{cases} h & \text{if } y > c(x) \\ 0 & \text{if } y \leq c(x) \end{cases}$$



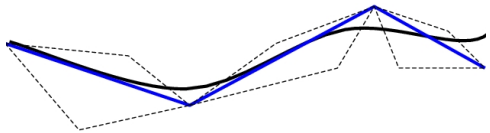
- $H(x, y)$ has the same complexity as $c(x)$, so optimally we want the same approximation rate.
- However for piecewise constant functions 2D tensor product wavelets cannot exploit the regularity of $c(x)$ and their approximation rate staggers at $\alpha = \frac{1}{2}$.

Support of base functions

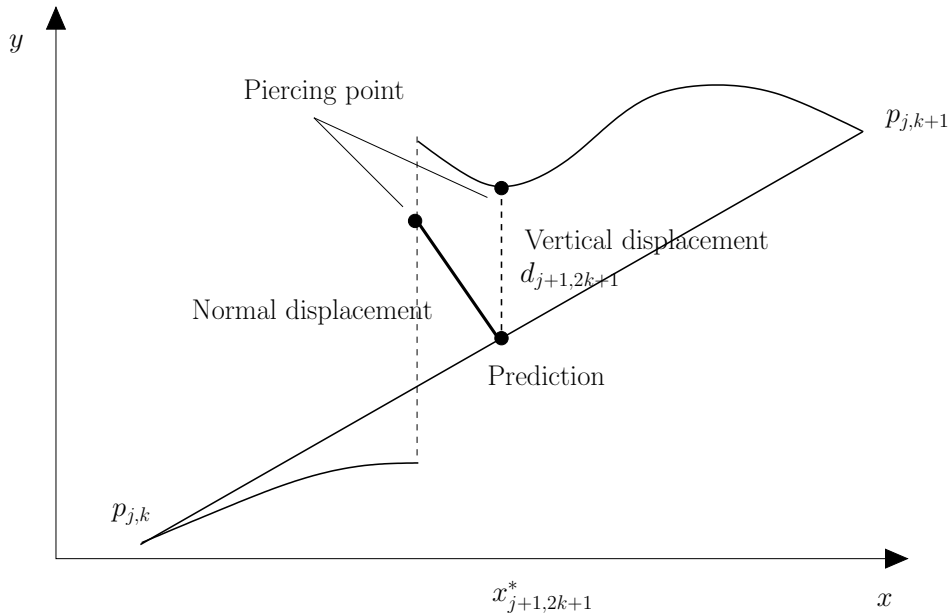
Tensor product wavelets:

- As the building blocks get sharper, their support shrinks like 2^{-2j}. Piecewise constant approximation of the contour: 
- Gibbs effects around the contours due to the vanishing moments.

Adaptive domain triangulation method:

- **construct** an adaptive domain partitioning which tesselates the domain such that the areas are stretched against $c(x)$ to form a polyline approximation of the contour: 

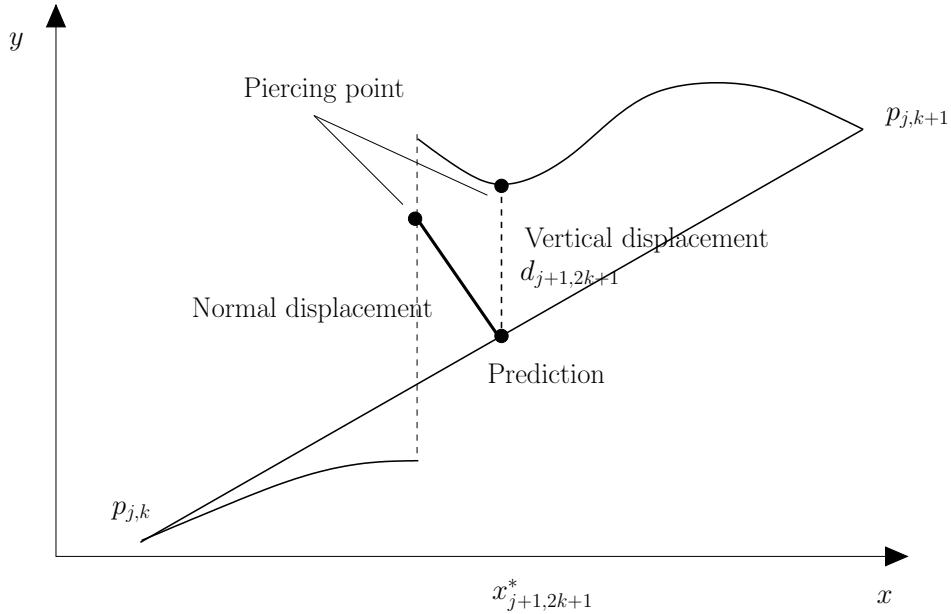
Concept



Wavelets: iterated filtering from fine to coarse

Normal offsets: iterated coarse to fine refinement $\longrightarrow$ no information about fine scales $\longrightarrow$ update step

Properties



- **Attracted towards discontinuïties** $\Rightarrow$ **adaptive partitioning** of the domain (small support at high frequencies).
- Normal offset gives information **both about the what** (function value) **and the where** (location wrt. domain).
  $\Rightarrow$ no *index* needed to point out the location

Extension towards 2D

- **Geometry**: new points are found by application of 1D algorithm on each edge.
- **Connectivity**: pattern based connectivity steered by an error norm. This **increases the degree of freedom** compared to subdivision connectivity.
- $\Rightarrow$ **Hierachical triangulation**

Extension towards 2D

- **Geometry**: new points are found by application of 1D algorithm on each edge.
- **Connectivity**: pattern based connectivity steered by an error norm. This **increases the degree of freedom** compared to subdivision connectivity.
- $\Rightarrow$ **Hierachical triangulation**

Extension towards 2D

Image Compression using Normal Offsets

Van Aerschot Ward

Problem

Concept

Discontinuities

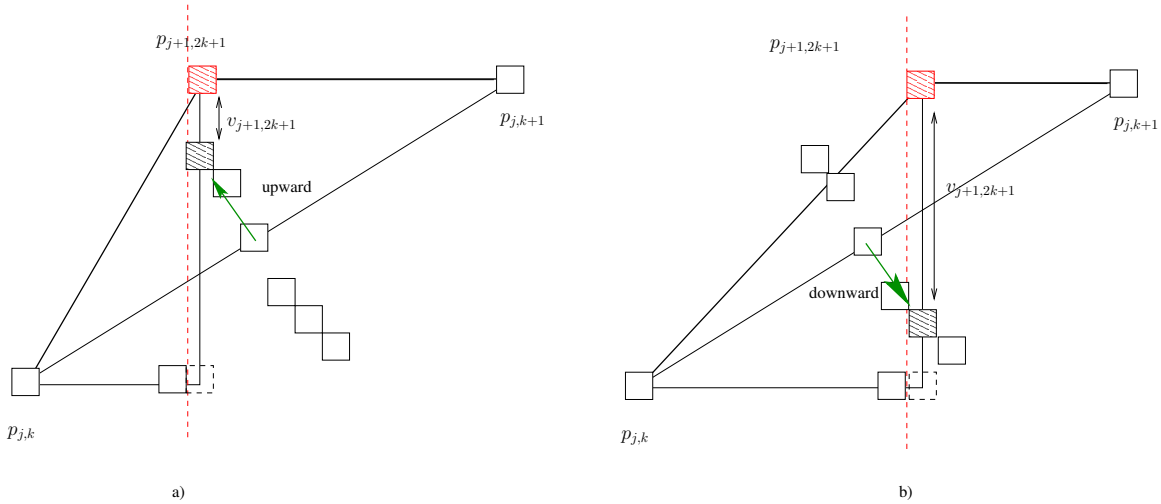
BaseMesh

- **Geometry**: new points are found by application of 1D algorithm on each edge.
- **Connectivity**: pattern based connectivity steered by an error norm. This **increases the degree of freedom** compared to subdivision connectivity.
- $\Rightarrow$ **Hierachical triangulation**

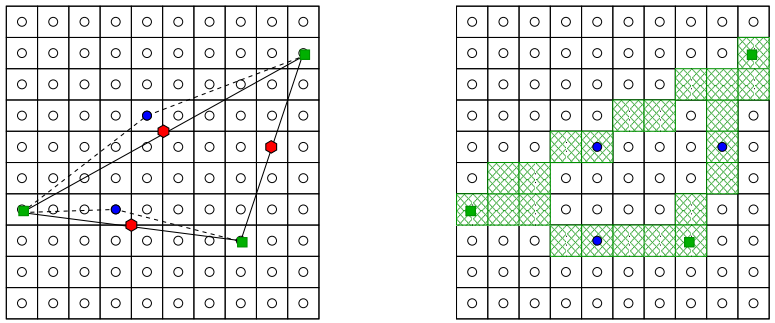
Discrete setting

Introduction of:

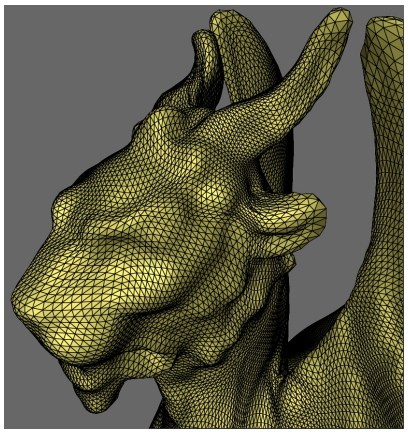
a) Vertical offsets:

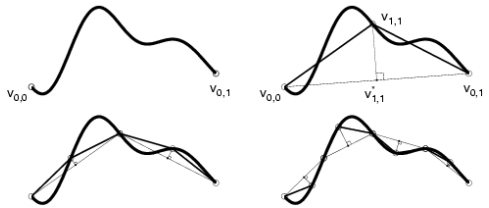


b) Line discretization:



Difference with previous research





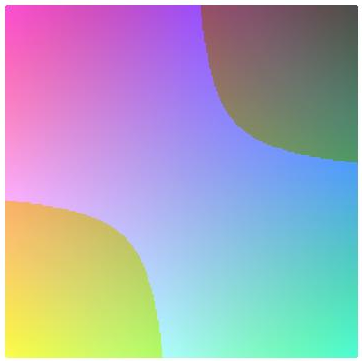
Remeshing

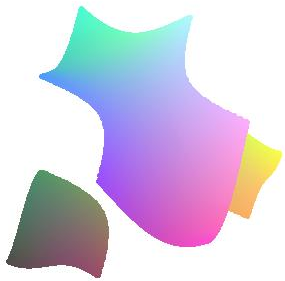
Approximation

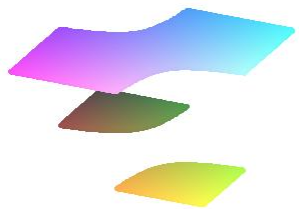
Smooth setting

⇳ versus

Piecewise smooth



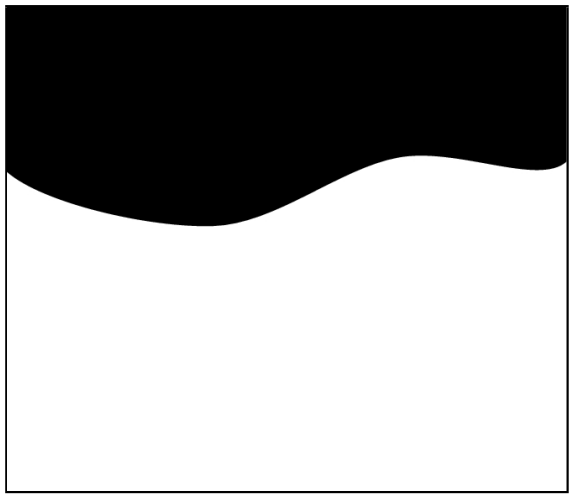


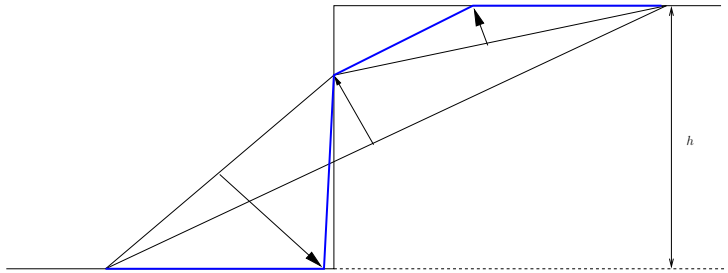


Normal offsets and discontinuities

1D When starting from a coarse grid of the two end points of a step function, or more general a step function adding a smooth function The normal offset procedure works fine.

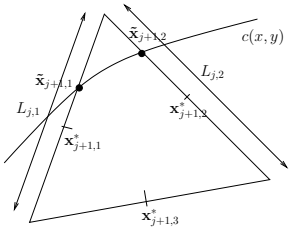
2D the same holds for 2D functions having only contour.

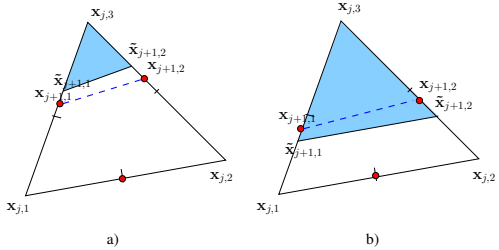


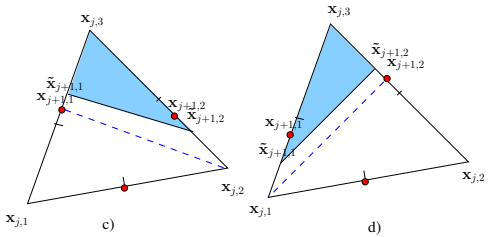


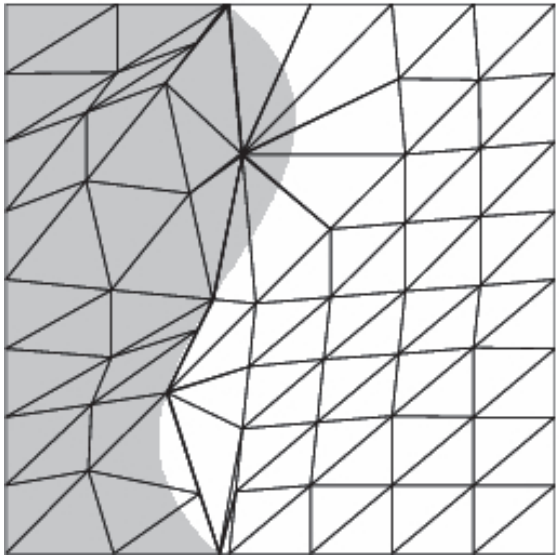
Problems arise when there are **multiple** contours lying close to each other.

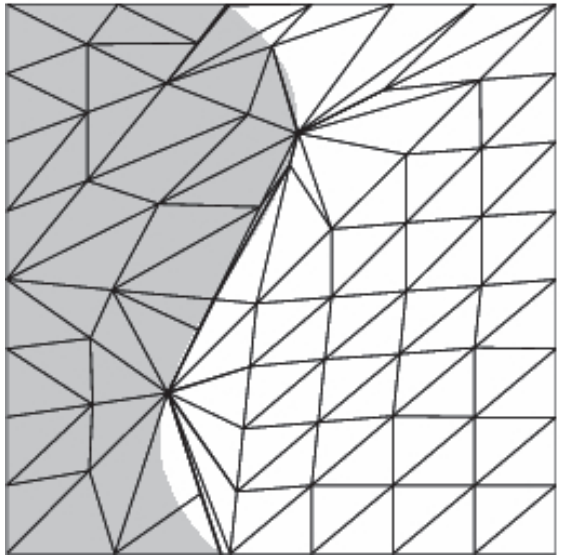
Adaptive triangulation





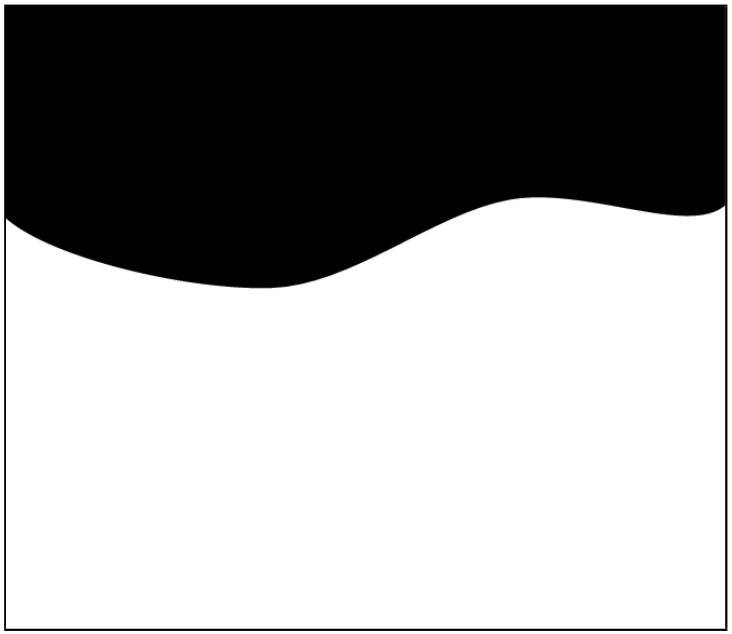






Example

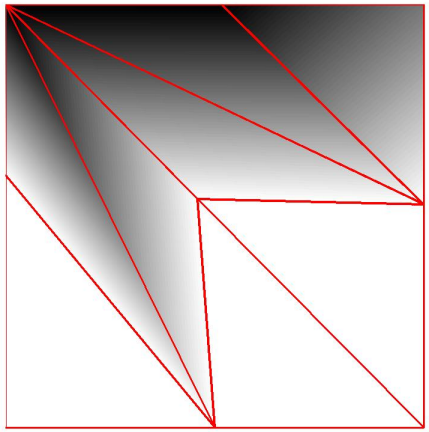
- Original image



Example

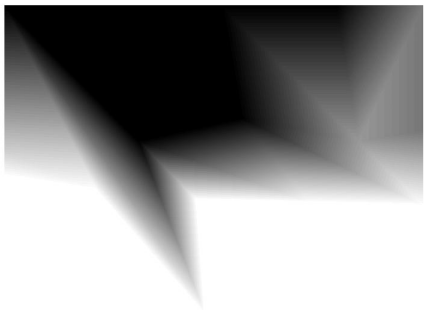
- Iteration level: 1

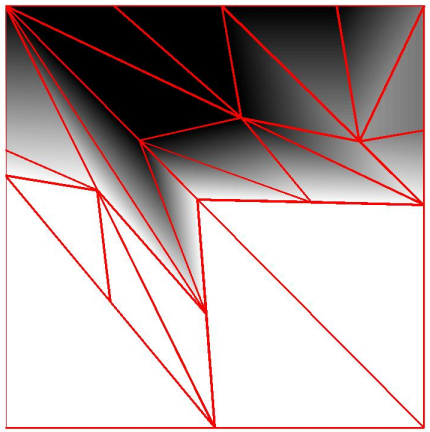




Example

- Iteration level: 2

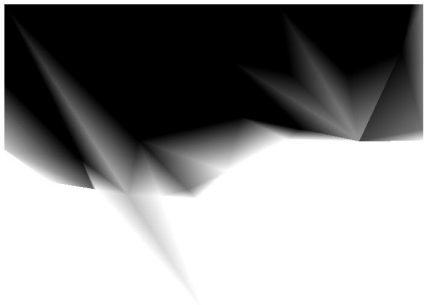


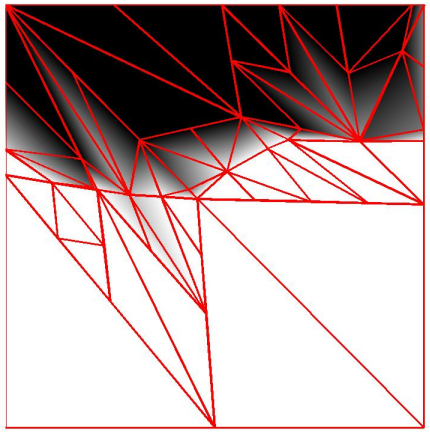


Thin triangles missing the contour!

Example

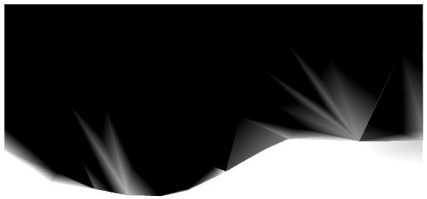
- Iteration level: 3

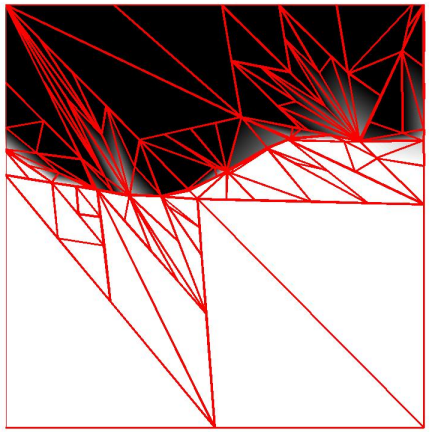




Example

- Iteration level: 4

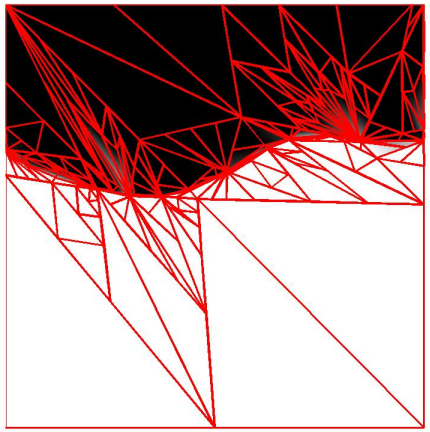




Example

- Iteration level: 5

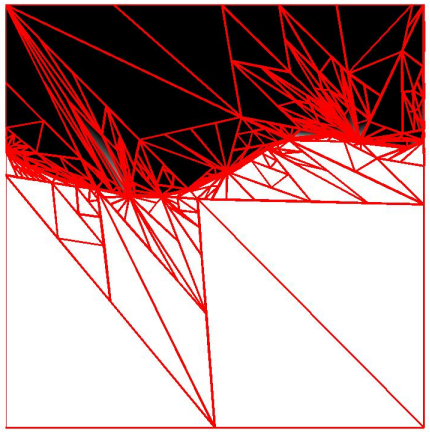




Example

- Iteration level: 6

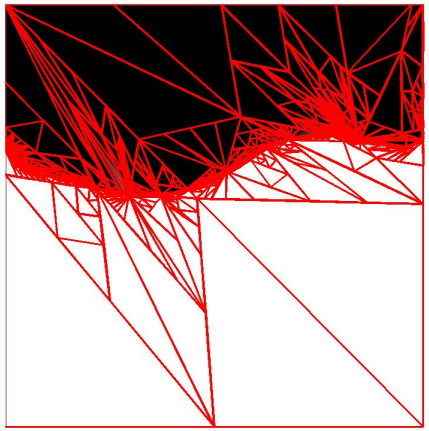




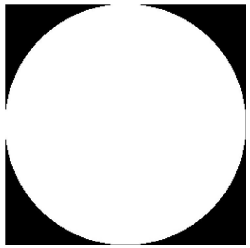
Example

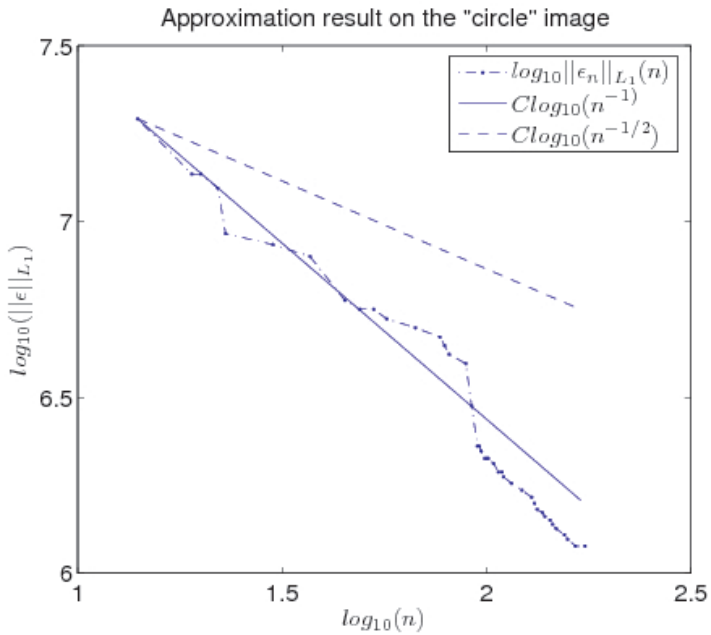
- Iteration level: 7



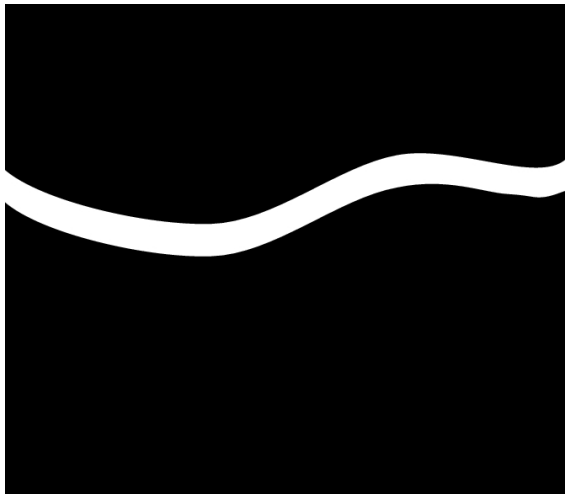


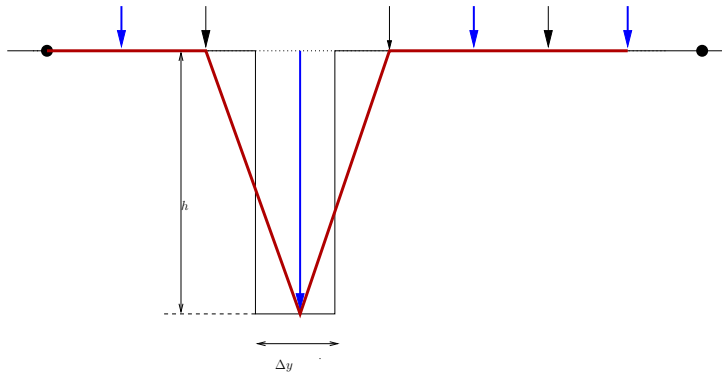
Approximation rate





Multiple contours



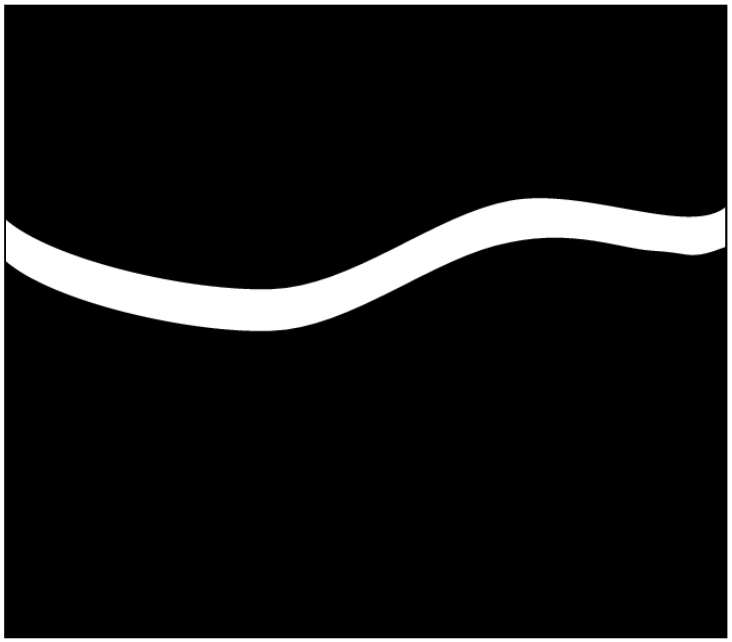


Consider $c(x)$ and $c(x) + \Delta y$ with $\Delta y << 1$ (image dimensions).

- Starting from an initial grid consisting out of the 4 image corners the points the approximation will **convergate slowly** towards the real image.
- Once a point is found the convergence will speed up only **locally**.

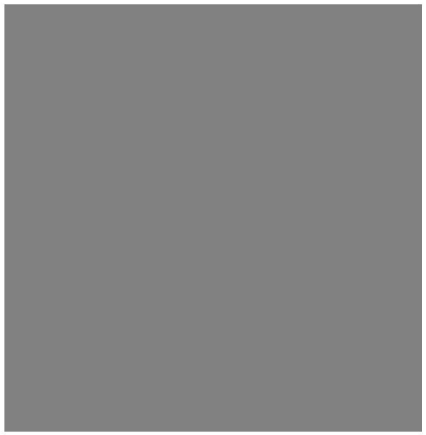
Example

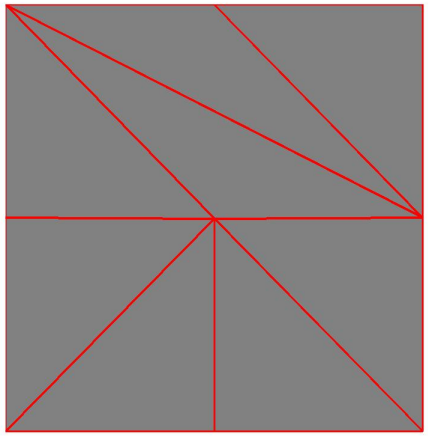
- Original image



Example

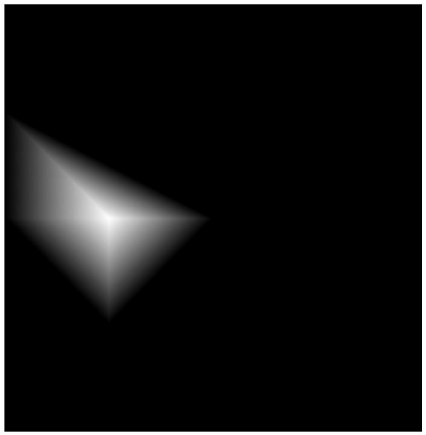
- Iteration level: 1

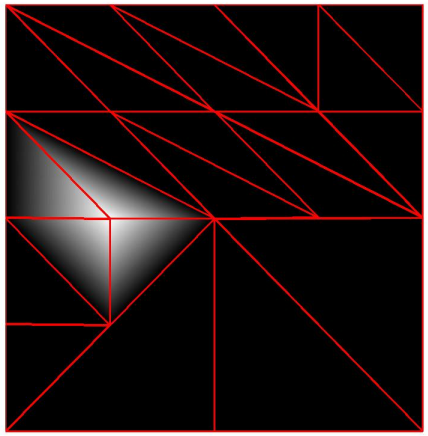




Example

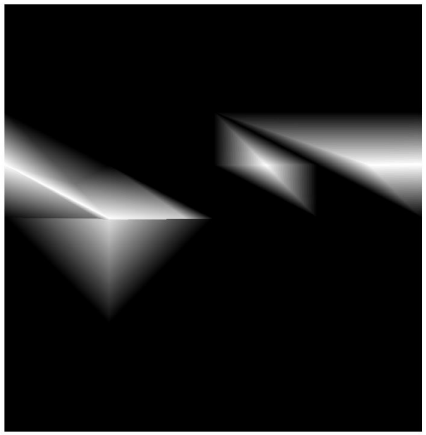
- Iteration level: 2

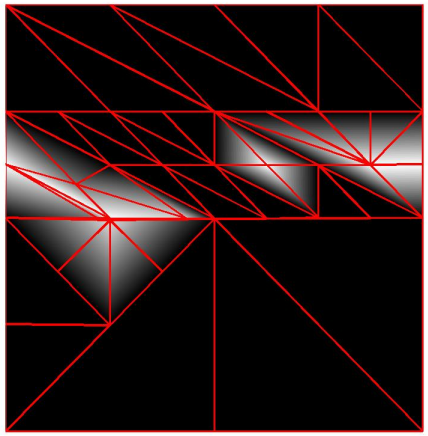




Example

- Iteration level: 3

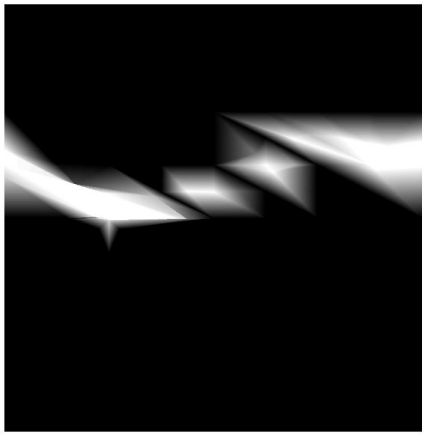




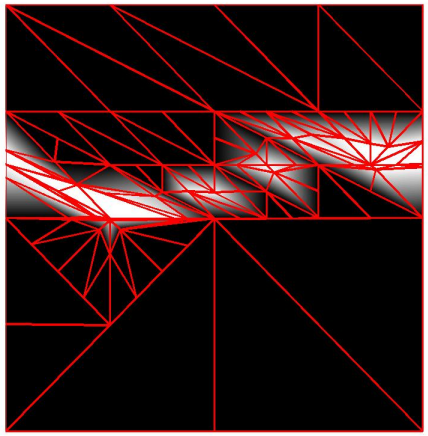
Contour broken into pieces

Example

- Iteration level: 4



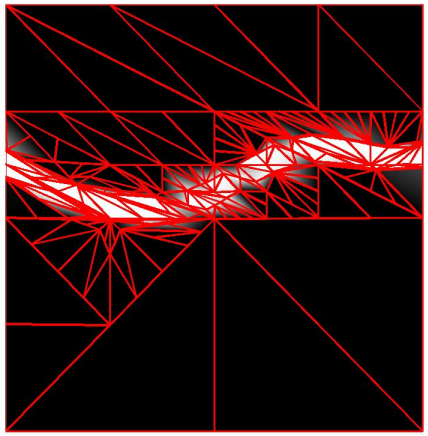
Local convergence



Example

- Iteration level: 5

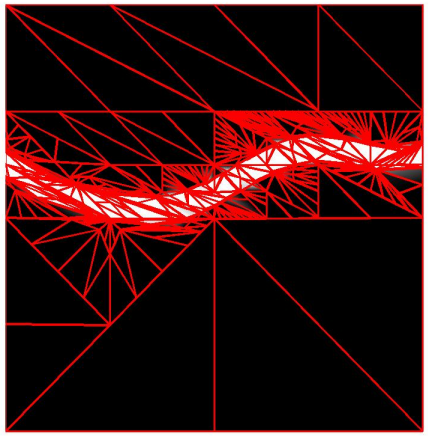




Example

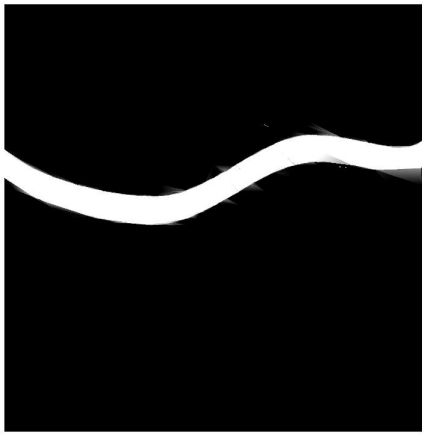
- Iteration level: 6

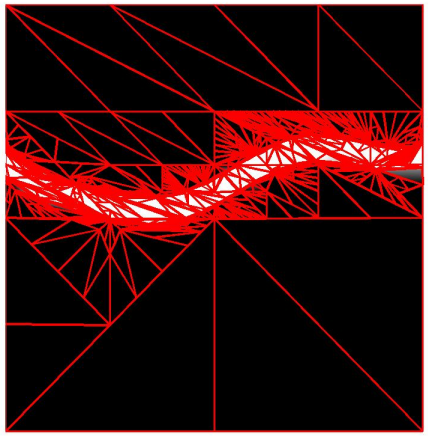




Example

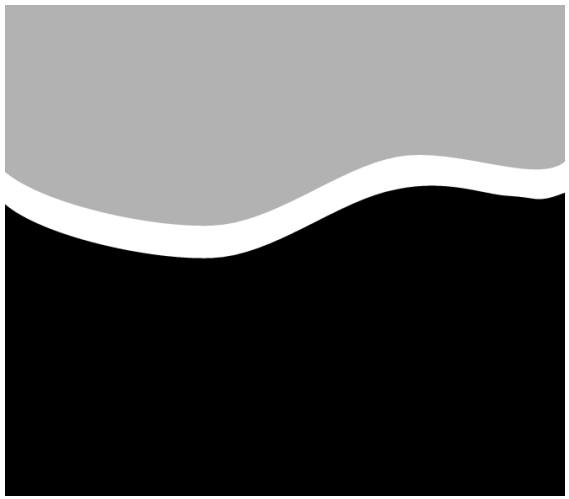
- Iteration level: 7

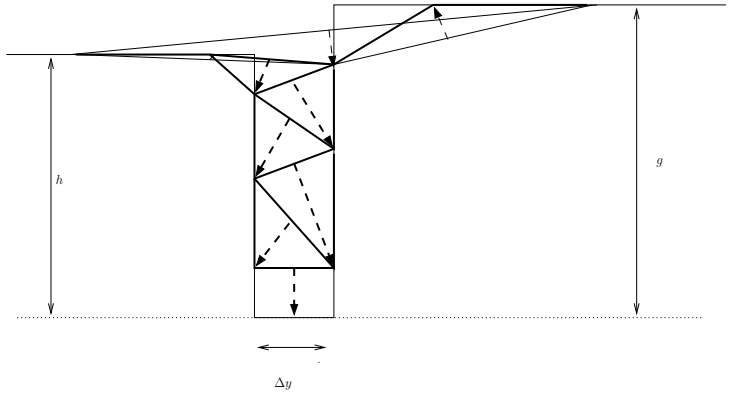




Multiple contours

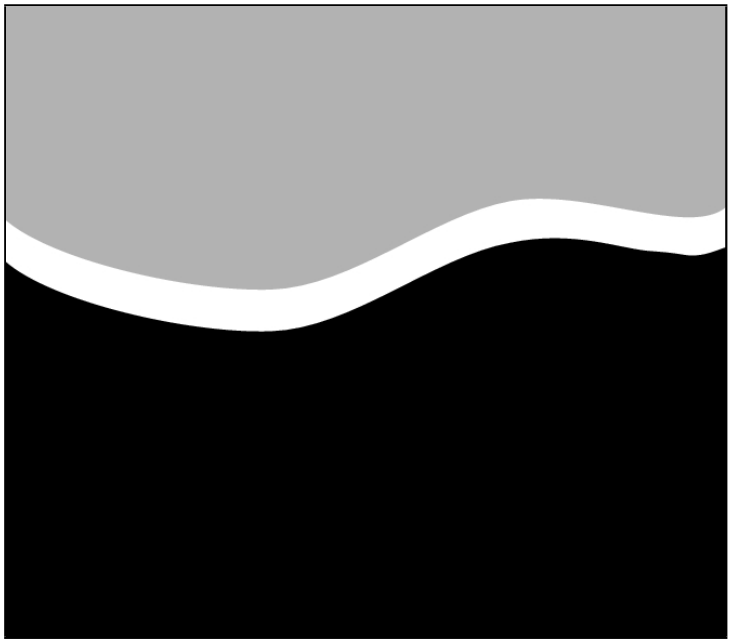
The problem becomes even worse when the zero area separates region having height h en g with $h >> 0$, $g > h$.





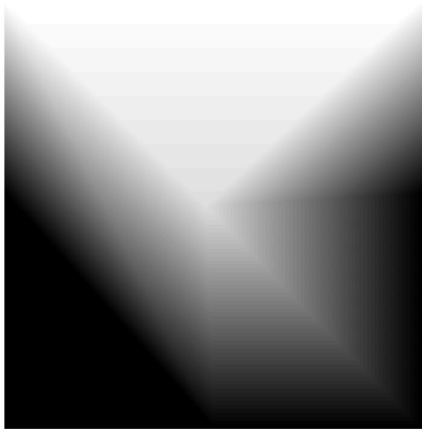
Example

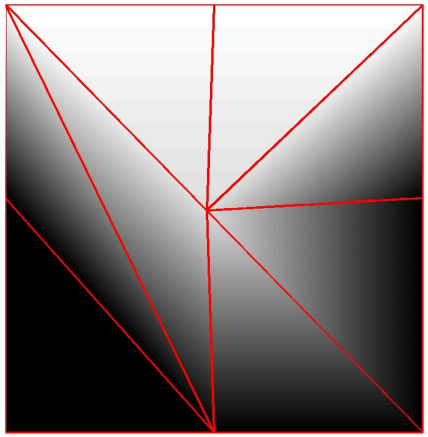
- Original image



Example

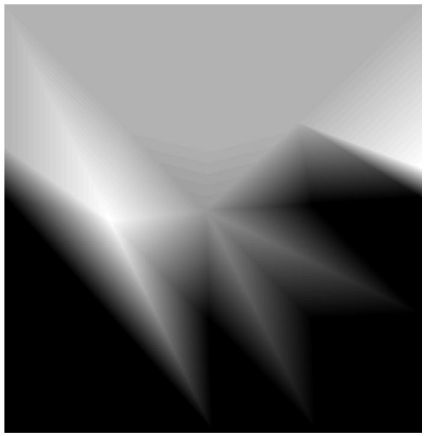
- Iteration level: 1

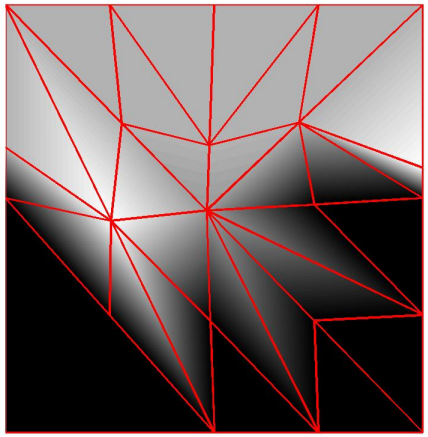




Example

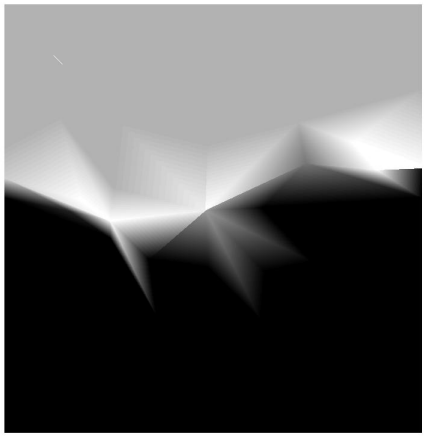
- Iteration level: 2

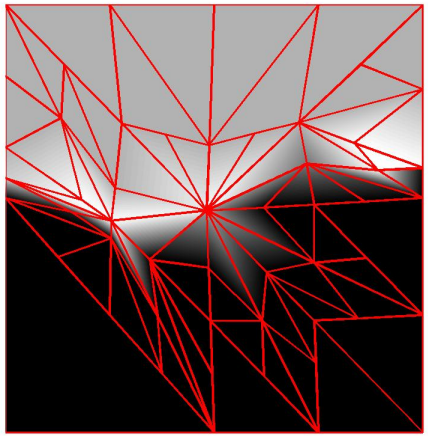




Example

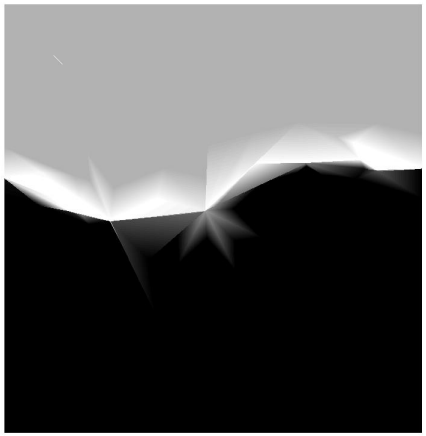
- Iteration level: 3

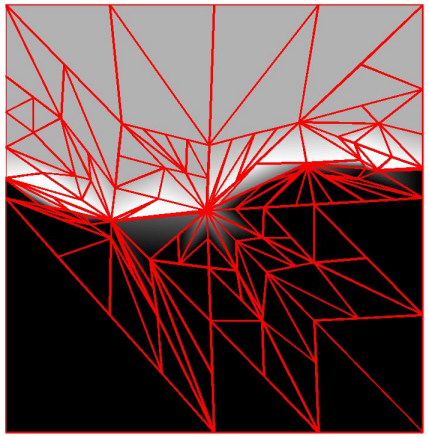




Example

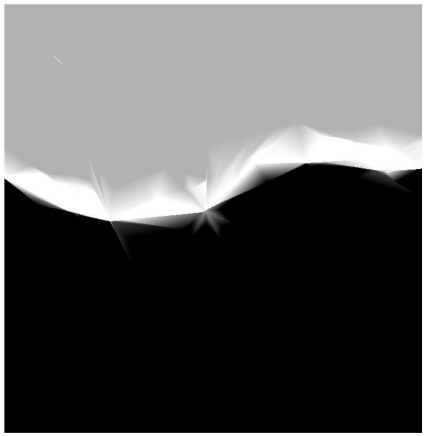
- Iteration level: 4

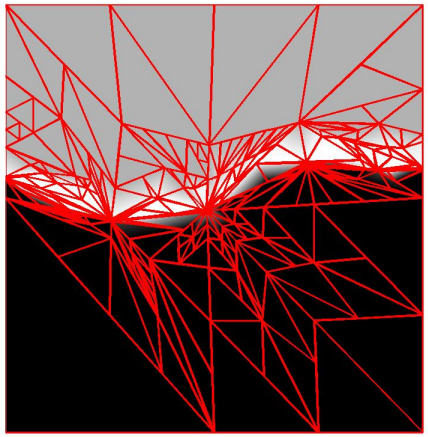




Example

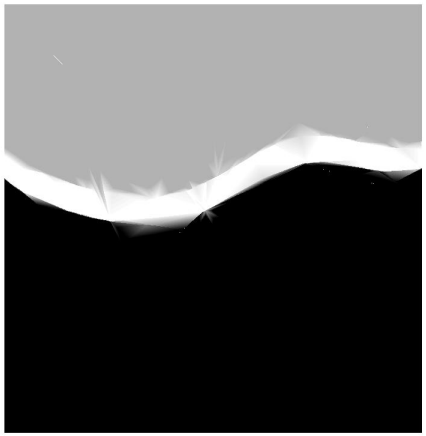
- Iteration level: 5

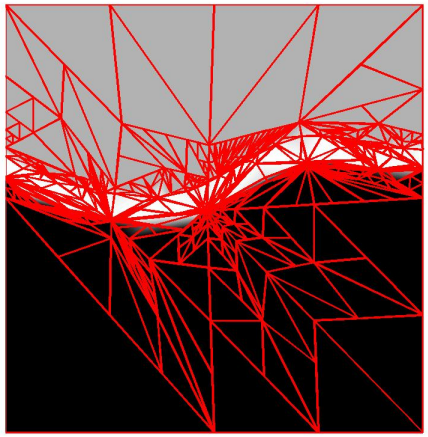




Example

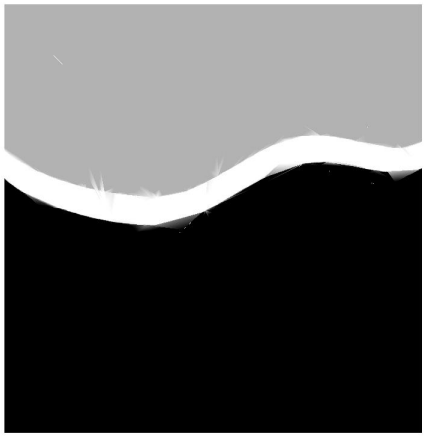
- Iteration level: 6

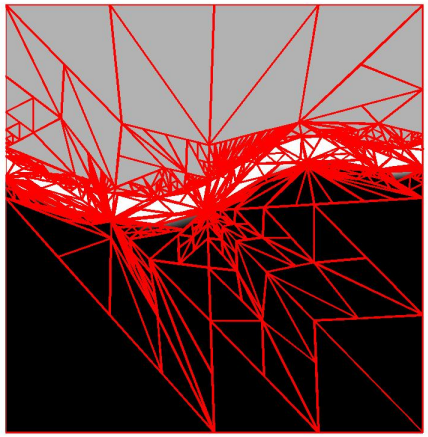




Example

- Iteration level: 7





Solution

Start from a base mesh comprising the narrow geometrical features of the image. $\Rightarrow$ increase in degrees of freedom.

Solution

Image Compression using Normal Offsets

Van Aerschot Ward

Problem

Concept

Discontinuities

BaseMesh

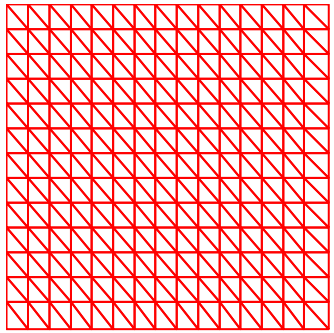
How to find such a mesh?

- Start from a dense equidistant mesh having **K** points
- perform 1 step of the normal offset algorithm
- Select **N** points having the largest vertical + normal offset
- Add simplices of the convex hull.
- perform a Delaunay triangulation
- Remove points to avoid thin triangles untill the minimal angle $\geq \theta$ or the number of vertices equals **L**
- If # vertices $\neq L$ then remove those lying close to each other on a straight line.
- Add midpoints to all remaining edges
- Remove points to avoid thin triangles untill the minimal angle $\geq \alpha$

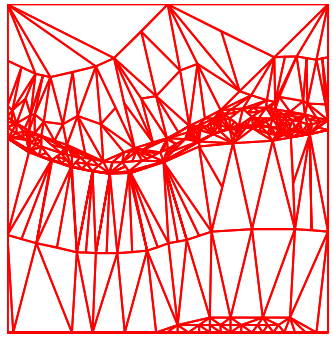
Solution

How to find such a mesh?

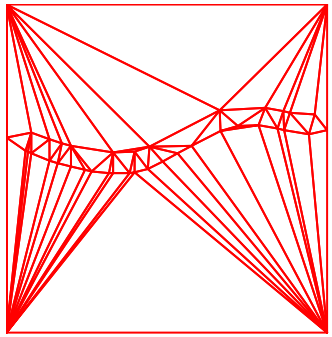
- Start from a dense equidistant mesh having **K** points
- perform 1 step of the normal offset algorithm
- Select **N** points having the largest vertical + normal offset
- Add simplices of the convex hull.
- perform a Delaunay triangulation
- Remove points to avoid thin triangles untill the minimal angle $\geq \theta$ or the number of vertices equals **L**
- If # vertices $\neq L$ then remove those lying close to each other on a straight line.
- Add midpoints to all remaining edges
- Remove points to avoid thin triangles untill the minimal angle $\geq \alpha$



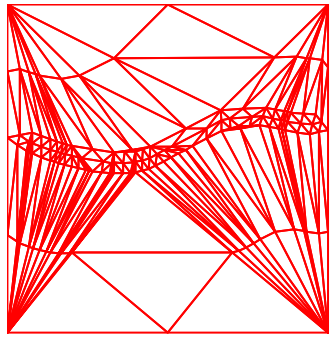
1. Equidistant grid



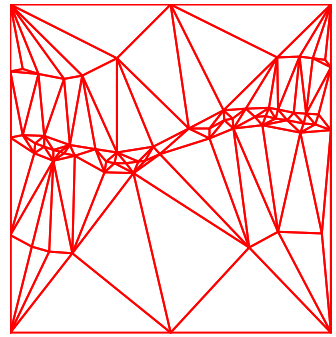
2. Selected piercing points



3. Removal of thin triangles



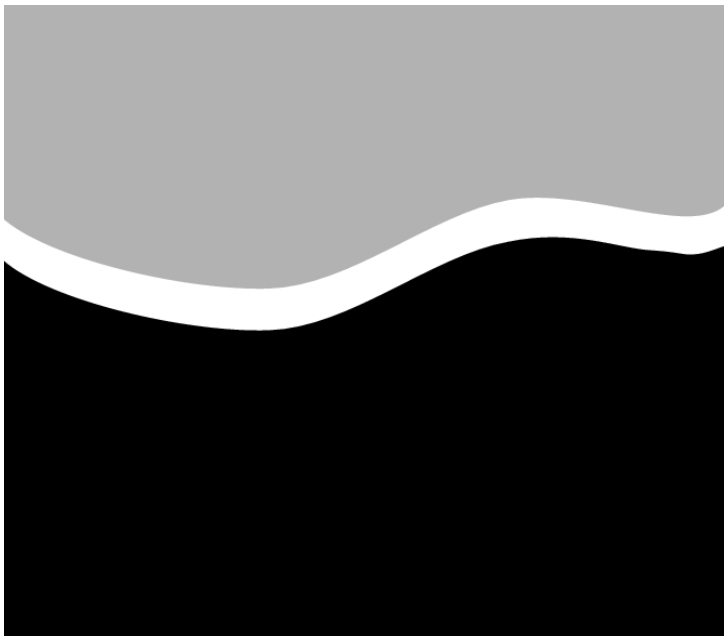
4. Inserting intermediate points



5. Removal of thin triangles

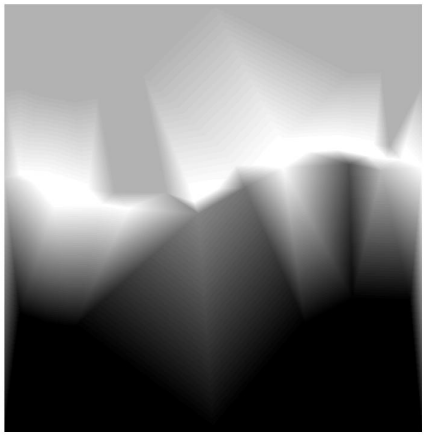
Comparison

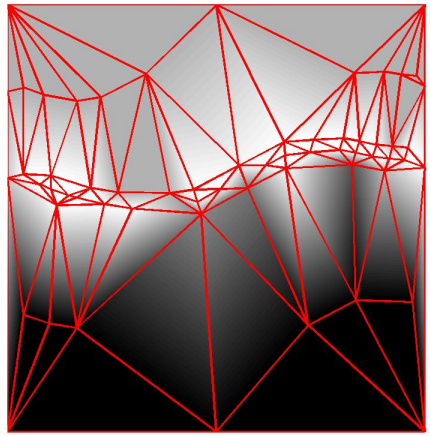
- Original image



Comparison

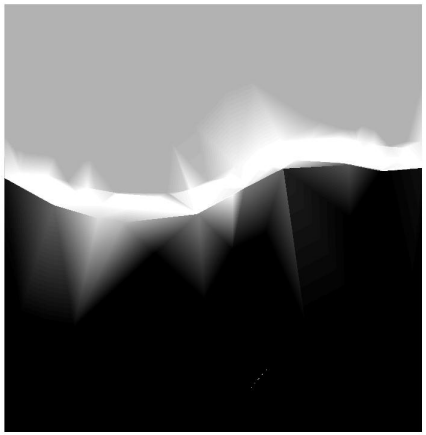
- Iteration level: 1

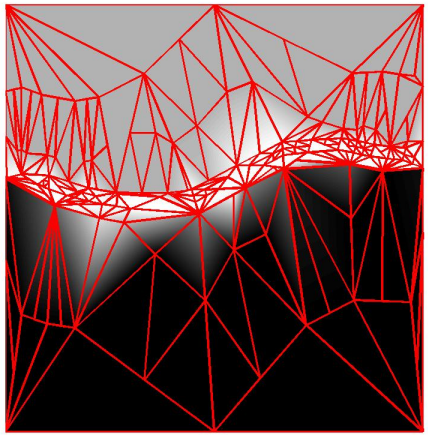




Comparison

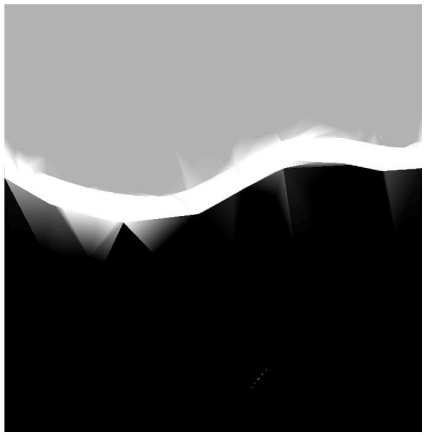
- Iteration level: 2

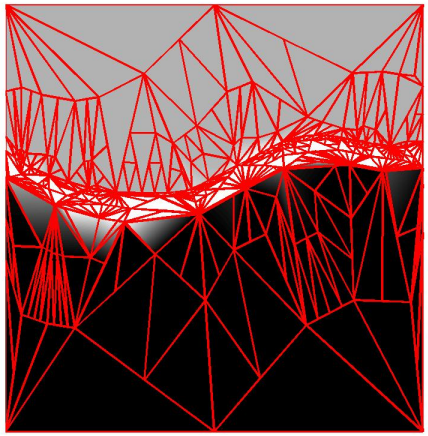




Comparison

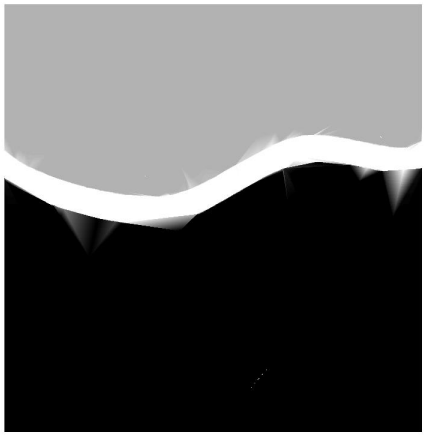
- Iteration level: 3

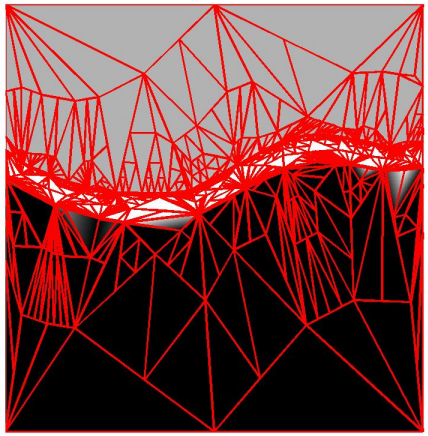




Comparison

- Iteration level: 4





Comparison

Image Compression using Normal Offsets

Van Aerschot Ward

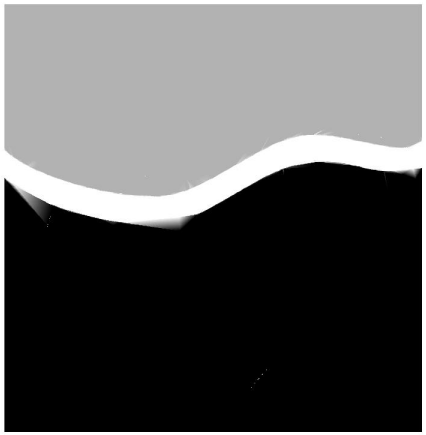
Problem

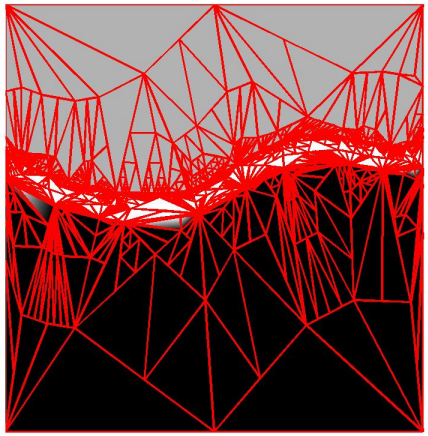
Concept

Discontinuities

BaseMesh

- Iteration level: 5





Conclusions

Image Compression using Normal Offsets

Van Aerschot Ward

Problem

Concept

Discontinuities

BaseMesh