Real-Time Rendering of Water Surfaces with Cartography-Oriented Design

9th International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging

Amir Semmo ¹  Jan Eric Kyprianidis ²  Matthias Trapp ¹  Jürgen Döllner ¹

¹Hasso-Plattner-Institut, Germany
²TU Berlin, Germany
Overview

1) Motivation
2) Design Principles
3) System Overview
4) Method
5) Results
6) Questions
MOTIVATION
State-of-the-Art Water Rendering

- computer-generated illustrations of water surfaces mainly based on photorealistic rendering
- but have neglected the challenges water surfaces exhibit for map design
- how to ease orientation, navigation, and analysis tasks within 3D geovirtual environments?
- challenges: emphasize land-water interface, consider figure-ground relations, express motion

Cartographers developed design principles that address these challenges
State-of-the-Art Water Rendering

- computer-generated illustrations of water surfaces mainly based on photorealistic rendering
- but have neglected the challenges water surfaces exhibit for map design
- how to ease orientation, navigation, and analysis tasks within 3D geovirtual environments?
- challenges: emphasize land-water interface, consider figure-ground relations, express motion

Cartographers developed design principles that address these challenges
Cartographic Design

The visualization of geospatial features using a map-like representation that considers its purpose and the target audience.
Waterlining
Design Principles – Waterlining

- became popular in the first half of the 20th century for lithographed maps
- fine solid lines are drawn parallel to shorelines with gradually increasing spaces
- waterlining provides dynamism and communicates distance information
Design Principles – Water Stippling

- small dots aligned to shorelines with non-linear distance
- contrary to traditional artwork, stipple have a varying density and irregularly overlap
- varying density to depict flow velocity or at occluded areas to enhance depth cues
Design Principles – Hatching & Vignetting

- Individual strokes placed with high density near shorelines complemented by loose lines
- Drawn excessively wavy with increasing irregularity towards middle stream to express motion
- Non-feature-aligned cross-hatches for land-water-distinction, coastal vignettes

www.hpi3d.de Semmo et al.
Design Principles – Thematic Visualization: Annotation / Symbolization

- names depicted with italic (slanted) letters, following principal curvature directions
- irregular placement of signatures with area-wide coverage to communicate water features
- placement of glyphs (e.g., arrows) along streamlines to symbolize flow direction of rivers
SYSTEM OVERVIEW

Local Orientation

Euclidean Distance

Medial Axis

Feature-aligned Distance

Quantitative Surface Analysis

Contour-Hatching

Water Stippling

Cross-Hatching

Sharp scan with large surrounding area
System Overview

Model Input & Masking

3D Models

Surface masking

Model Input & Masking

Texture Features

Hatches

Strokes

Glyphs

Cartography-Oriented Shading

Water Stippling

Waterlining

Contour-Hatching

Cross-Hatching

texture mapping

Labels

Output

www.hpi3d.de Semmo et al.
System Overview

Quantitative Surface Analysis

- Water Surfaces
- Euclidean Distance
- Local Orientation
- Medial Axis
- Feature-aligned Distance

3D Models

- Iterative processing
- Texture mapping

Output

- Cartography-Oriented Shading
- Texture Features
- Labels
System Overview

Cartography-Oriented Shading & Texture Features
System Overview

Process Iteration & Texture Mapping
Method – Data Input & Euclidean Distance Transform

- input: 2D or 3D water surfaces defined as polygons or triangular irregular networks.
Method – Data Input & Euclidean Distance Transform

- input: 2D or 3D water surfaces defined as polygons or triangular irregular networks
- using orthographic projections, the models’ shape are captured in 2D binary masks

Input: Shoreline Distance
Shoreline Direction

www.hpi3d.de
Semmo et al.
Method – Data Input & Euclidean Distance Transform

- input: 2D or 3D water surfaces defined as polygons or triangular irregular networks
- using orthographic projections, the models’ shape are captured in 2D binary masks
- distance transform to obtain minimum Euclidean distance of each pixel to a shoreline
Method – Data Input & Euclidean Distance Transform

▶ input: 2D or 3D water surfaces defined as polygons or triangular irregular networks
▶ using orthographic projections, the models’ shape are captured in 2D binary masks
▶ distance transform to obtain minimum Euclidean distance of each pixel to a shoreline
▶ “Parallel Banding Algorithm” [Cao et al., I3D 2010] obtains distance map in real-time [CUDA]
Method – Data Input & Euclidean Distance Transform

- input: 2D or 3D water surfaces defined as polygons or triangular irregular networks
- using orthographic projections, the models’ shape are captured in 2D binary masks
- distance transform to obtain minimum Euclidean distance of each pixel to a shoreline
- “Parallel Banding Algorithm” [Cao et al., I3D 2010] obtains distance map in real-time [CUDA]
Method – Local Orientation Estimation & Medial Axes Computation

- smoothed structure tensor with eigenanalysis to obtain stable estimates of local orientation
  [Brox et al., 2006]
- used to derive medial axes for aligning design elements (e.g., labels) along middle stream
Method – Local Orientation Estimation & Medial Axes Computation

- smoothed structure tensor with eigenanalysis to obtain stable estimates of local orientation
  
  [Brox et al., 2006]

- used to derive medial axes for aligning design elements (e.g., labels) along middle stream

- compare and threshold the unsigned gradient orientation for each point
Method – Local Orientation Estimation & Medial Axes Computation

- smoothed structure tensor with eigenanalysis to obtain stable estimates of local orientation [Brox et al., 2006]
- used to derive medial axes for aligning design elements (e.g., labels) along middle stream
- compare and threshold the unsigned gradient orientation for each point

Thresholding Shoreline Directions
Method – Local Orientation Estimation & Medial Axes Computation

- smoothed structure tensor with eigenanalysis to obtain stable estimates of local orientation
  [Brox et al., 2006]
- used to derive medial axes for aligning design elements (e.g., labels) along middle stream
- compare and threshold the unsigned gradient orientation for each point
Method – Local Orientation Estimation & Medial Axes Computation

- smoothed structure tensor with eigenanalysis to obtain stable estimates of local orientation [Brox et al., 2006]

- used to derive medial axes for aligning design elements (e.g., labels) along middle stream

- compare and threshold the unsigned gradient orientation for each point
Method – Local Orientation Estimation & Medial Axes Computation

- smoothed structure tensor with eigenanalysis to obtain stable estimates of local orientation
  [Brox et al., 2006]
- used to derive medial axes for aligning design elements (e.g., labels) along middle stream
- compare and threshold the unsigned gradient orientation for each point
Method – Local Orientation Estimation & Medial Axes Computation

- smoothed structure tensor with eigenanalysis to obtain stable estimates of local orientation
  [Brox et al., 2006]
- used to derive medial axes for aligning design elements (e.g., labels) along middle stream
- compare and threshold the unsigned gradient orientation for each point
Method – Feature-aligned Distance Transform

Contour-Hatching

- requires design elements (e.g., strokes) to be exactly aligned with the shorelines
- requires fine control over their placement per rendering pass (water movements!)
Method – Feature-aligned Distance Transform

Contour-Hatching

- requires design elements (e.g., strokes) to be exactly aligned with the shorelines
- requires fine control over their placement per rendering pass (water movements!)

Our Approach

- parameterize level-set curves of distance map to obtain distances along its tangential field
- level sets correspond to integral part of shoreline distances for non-normalized distance map
- similar to vector propagation, distances are iteratively propagated in parallel [CUDA]
Method – Feature-aligned Distance Transform

Contour-Hatching

- requires design elements (e.g., strokes) to be exactly aligned with the shorelines
- requires fine control over their placement per rendering pass (water movements!)

Our Approach

- parameterize level-set curves of distance map to obtain distances along its tangential field
- level sets correspond to integral part of shoreline distances for non-normalized distance map
- similar to vector propagation, distances are iteratively propagated in parallel [CUDA]
Method – Feature-aligned Distance Transform

Contour-Hatching

- requires design elements (e.g., strokes) to be exactly aligned with the shorelines
- requires fine control over their placement per rendering pass (water movements!)

Our Approach

- parameterize level-set curves of distance map to obtain distances along its tangential field
- level sets correspond to integral part of shoreline distances for non-normalized distance map
- similar to vector propagation, distances are iteratively propagated in parallel [CUDA]

www.hpi3d.de

Semmo et al.
Method – Feature-aligned Distance Transform

Contour-Hatching

- requires design elements (e.g., strokes) to be exactly aligned with the shorelines
- requires fine control over their placement per rendering pass (water movements!)

Our Approach

- parameterize level-set curves of distance map to obtain distances along its tangential field
- level sets correspond to integral part of shoreline distances for non-normalized distance map
- similar to vector propagation, distances are iteratively propagated in parallel [CUDA]
Method – Feature-aligned Distance Transform

Contour-Hatching

- requires design elements (e.g., strokes) to be exactly aligned with the shorelines
- requires fine control over their placement per rendering pass (water movements!)

Our Approach

- parameterize level-set curves of distance map to obtain distances along its tangential field
- level sets correspond to integral part of shoreline distances for non-normalized distance map
- similar to vector propagation, distances are iteratively propagated in parallel [CUDA]
Method – Feature-aligned Distance Transform

Contour-Hatching

- requires design elements (e.g., strokes) to be exactly aligned with the shorelines
- requires fine control over their placement per rendering pass (water movements!)

Our Approach

- parameterize level-set curves of distance map to obtain distances along its tangential field
- level sets correspond to integral part of shoreline distances for non-normalized distance map
- similar to vector propagation, distances are iteratively propagated in parallel [CUDA]
Method – Feature-aligned Distance Transform

- **Texture Coordinates**
  
  are directly derived by combining shoreline and feature-aligned distances

- **Texture-based Shading**
  
  that is capable of feature-aligned hatching at real-time frame rates

- **Artistic Control**
  
  with dynamic parameterization at run-time to express water movements
Method – Feature-aligned Distance Transform

- contrary to energy minimization, our approach does not provide continuity across level sets
- individual texture features are aligned with level-set curves → no such constraint is required
- piecewise-linear approximation of shoreline / feature-aligned distances by bilinear sampling
- choose distance map resolution that balances rendering quality and performance

Bilinear Sampling  Discrete Input + Distance Lines  Continuous Output
Method – Waterlining

- waterlines correspond to shaded areas with equal shoreline distance
- non-linear step function to compute target distance values with non-equidistant interspaces
- waterlines are padded by fade-in and fade-out intervals for antialiasing and parameterized by the view distance for a continuous level-of-abstraction

Shoreline Distance + Non-Linear Step → Waterlines
Method – Waterlining

- Waterlines correspond to shaded areas with equal shoreline distance
- Non-linear step function to compute target distance values with non-equidistant interspaces
- Waterlines are padded by fade-in and fade-out intervals for antialiasing and parameterized by the view distance for a continuous level-of-abstraction

Shoreline Distance + Non-Linear Step → Waterlines

\[ s=40, \ e=0.6, \ h=0 \]
Method – Water Stippling

Texture Bombing
- ... from Glanville [2004] enhanced to place water stipples with feature-aligned distribution and irregular density

Stipples aligned with shorelines
- ... by stipple selection, stipple displacement, and stipple filtering

Texture-based parameterization
- ... using tonal art maps to locally vary stipple density and tone, layered / iterative approach to regularize / vary stipple density [GLSL]
Method – Water Stippling

Texture Bombing
- ... from Glanville [2004] enhanced to place water stipples with feature-aligned distribution and irregular density

Stipples aligned with shorelines
- ... by stipple selection, stipple displacement, and stipple filtering

Texture-based parameterization
- ... using tonal art maps to locally vary stipple density and tone, layered / iterative approach to regularize / vary stipple density [GLSL]
Method – Water Stippling

Texture Bombing
- ... from Glanville [2004] enhanced to place water stipples with feature-aligned distribution and irregular density

Stipples aligned with shorelines
- ... by stipple selection, stipple displacement, and stipple filtering

Texture-based parameterization
- ... using tonal art maps to locally vary stipple density and tone, layered / iterative approach to regularize / vary stipple density [GLSL]
Method – Water Stippling

Texture Bombing

- ... from Glanville [2004] enhanced to place water stipples with feature-aligned distribution and irregular density

Stipples aligned with shorelines

- ... by stipple selection, stipple displacement, and stipple filtering

Texture-based parameterization

- ... using tonal art maps to locally vary stipple density and tone, layered / iterative approach to regularize / vary stipple density [GLSL]

Selection  Displacement
Method – Water Stippling

**Texture Bombing**
- ... from Glanville [2004] enhanced to place water stipples with feature-aligned distribution and irregular density

**Stipples aligned with shorelines**
- ... by stipple selection, stipple displacement, and stipple filtering

**Texture-based parameterization**
- ... using tonal art maps to locally vary stipple density and tone, layered / iterative approach to regularize / vary stipple density [GLSL]
Method – Water Stippling

Texture Bombing

- ... from Glanville [2004] enhanced to place water stipples with feature-aligned distribution and irregular density

Stipples aligned with shorelines

- ... by stipple selection, stipple displacement, and stipple filtering

Texture-based parameterization

- ... using tonal art maps to locally vary stipple density and tone, layered / iterative approach to regularize / vary stipple density [GLSL]
Method – Contour-Hatching

- parameterize shoreline / feature-aligned distances to irregularly place stroke maps [GLSL]
- control over stroke length, thickness, spacing, randomness according to shoreline distance
- enables time-varying parameterization for water movements and frame-to-frame coherence

\[
\phi(d) + t
\]

Stroke Placement

Texture Coordinates

Rendering Result
Method – Contour-Hatching

- parameterize shoreline / feature-aligned distances to irregularly place stroke maps \([\text{GLSL}]\)
- control over stroke length, thickness, spacing, randomness according to shoreline distance
- enables time-varying parameterization for water movements and frame-to-frame coherence

Stroke Placement

Texture Coordinates

Rendering Result
Method – Contour-Hatching

- parameterize shoreline / feature-aligned distances to irregularly place stroke maps [GLSL]
- control over stroke length, thickness, spacing, randomness according to shoreline distance
- enables time-varying parameterization for water movements and frame-to-frame coherence

![Diagram showing stroke placement, texture coordinates, and rendering result.](image-url)
Method – Contour-Hatching

- parameterize shoreline / feature-aligned distances to irregularly place stroke maps [GLSL]
- control over stroke length, thickness, spacing, randomness according to shoreline distance
- enables time-varying parameterization for water movements and frame-to-frame coherence

![Diagram showing stroke placement, texture coordinates, and rendering result.](image)
Method – Cross-Hatching & Labeling

- Cross-hatching at shorelines by a tonal art map with varying stroke sizes / densities [Praun et al., 2001]
- Our approach does not affect shading of landmass, i.e., terrain can be stylized separately

- Downsample medial axes and use tangent information of structure tensor for arc-length parameterization of piecewise cubic Bezier curves
- OpenGL extension `NV_path_rendering` to render / transform instance-based text in a single pass
Method – View-Dependent Level-of-Abstraction

Level-of-Abstraction 0

Cross-Hatching
Contour-Hatching
Water Stippling
Waterlining

Level-of-Abstraction 1

Waterlining
Water Stippling
Contour-Hatching
Cross-Hatching
Results – 3D Mapping

3D mapping within the environment of Mount St. Helens / waterlining applied to a globe
Results – Flooding Simulation

- Waterlining expresses uncertainty, conveys motion, and enhances the depiction of land cover.
- Flooding simulation: assess distances to nearest safety zones for evacuation planning.
- A plane is used as clipping mask and temporally shifted upwards to represent the change of the mean sea level, uses orthographic projection to obtain flooded areas.

Flooding simulation for the city of Boston
Contour-hatching and water stippling to express uncertainty
Conclusions

▶ **design principles** from cartography can be used to improve **figure-ground perception**, which requires further **validation**

▶ shoreline and feature-aligned **distance maps** are suitable for **texture-based** waterlining, water stippling, contour-hatching

▶ **context-aware parameterization** of shading effects remains an important issue for **view-dependent level-of-abstraction**
Questions and Contact Information

Thank You For Your Attention!

▶ Amir Semmo
  amir.semmo@hpi.uni-potsdam.de

▶ Computer Graphics Systems Group
  Prof. Dr. Jürgen Döllner
  hpi3d.de  youtube.com/hpicgs  @hpi3d
Performance Analysis

Performance evaluation (in ms): distance / feature-aligned distance transform ($D/T$), orientation and medial axis computation.

<table>
<thead>
<tr>
<th>Image Resolution</th>
<th>$D$</th>
<th>$T$</th>
<th>Orient.</th>
<th>M. Axes</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>$128 \times 128$</td>
<td>1.6</td>
<td>26.6</td>
<td>0.1</td>
<td>0.7</td>
<td>29.0</td>
</tr>
<tr>
<td>$256 \times 256$</td>
<td>2.2</td>
<td>96.3</td>
<td>0.2</td>
<td>0.8</td>
<td>99.5</td>
</tr>
<tr>
<td>$512 \times 512$</td>
<td>4.5</td>
<td>371.6</td>
<td>0.6</td>
<td>0.9</td>
<td>377.6</td>
</tr>
</tbody>
</table>

Performance evaluation for rendering techniques using different screen resolutions (in frames-per-second).

<table>
<thead>
<tr>
<th>Screen Resolution</th>
<th>waterlining</th>
<th>stippling</th>
<th>contour-hatching</th>
</tr>
</thead>
<tbody>
<tr>
<td>$800 \times 600$</td>
<td>534</td>
<td>162</td>
<td>159</td>
</tr>
<tr>
<td>$1280 \times 720$</td>
<td>523</td>
<td>88</td>
<td>84</td>
</tr>
<tr>
<td>$1600 \times 900$</td>
<td>521</td>
<td>59</td>
<td>55</td>
</tr>
<tr>
<td>$1920 \times 1080$</td>
<td>514</td>
<td>42</td>
<td>41</td>
</tr>
</tbody>
</table>