# Geometric Displacement on Plane and Sphere

Elodie Fourquet, William Cowan, Stephen Mann

University of Waterloo

May 30, 2008

# Motivation: Texture

Considering a surface, texture-based algorithms allow

- separating material attributes from underlying geometry,
- minimizing memory usage,
- simplifing acquisition, and
- optimizing performance ($2D$ map + pixel independent).

# Motivation: Bump Mapping

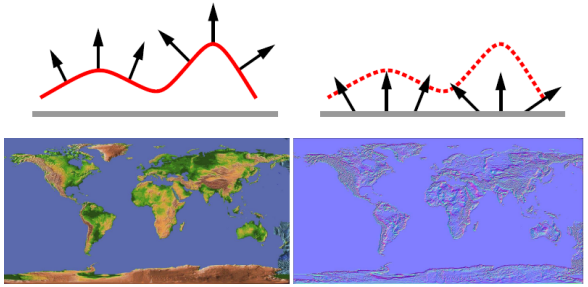
Given texture colour map and normal map, bump mapping is

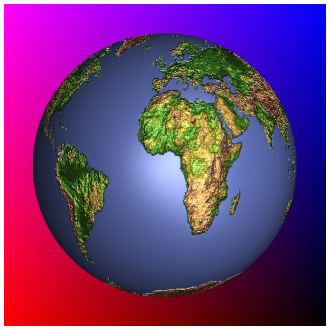
- a lighting approximation,
- $2D$ map,
- very effective from far away.

Bump mapping fails to produce,

- self-shadowing and self-occlusion, and
- correct silhouettes.

Displacement mapping can produce those effects.

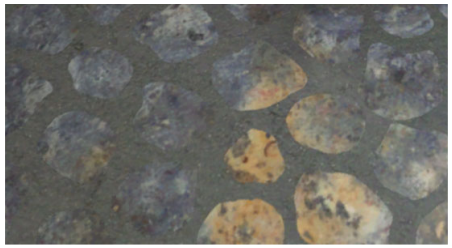


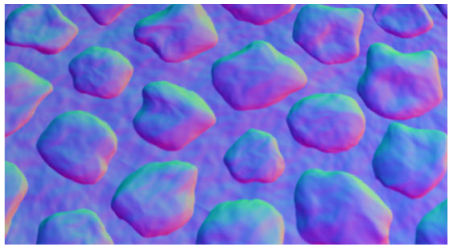


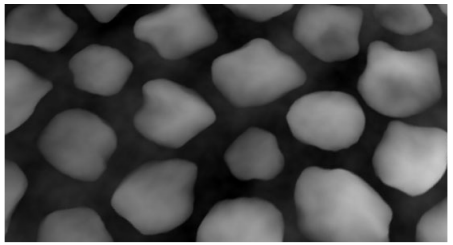
# Displacement Mapping Algorithm Goals

Considering a simple underlying surface, we want to

- render the height convincingly from $2D$ height map values,
- understand the height map intersection geometry,
- use $1D$ and $2D$ computations as much as possible, and
- identify possible parallelism.





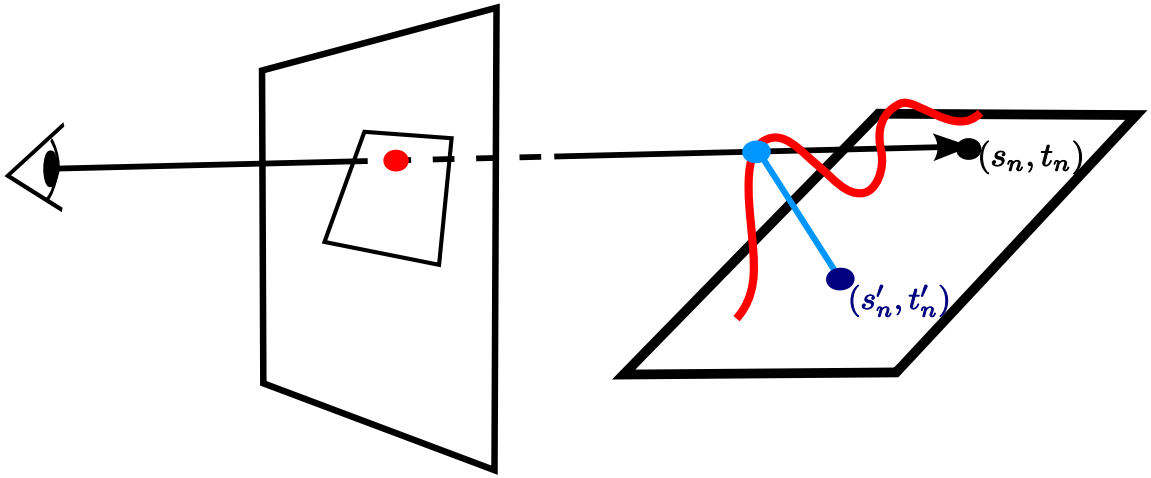


# Outline

- Geometry
- Stepping Algorithm
- Results
- Conclusions

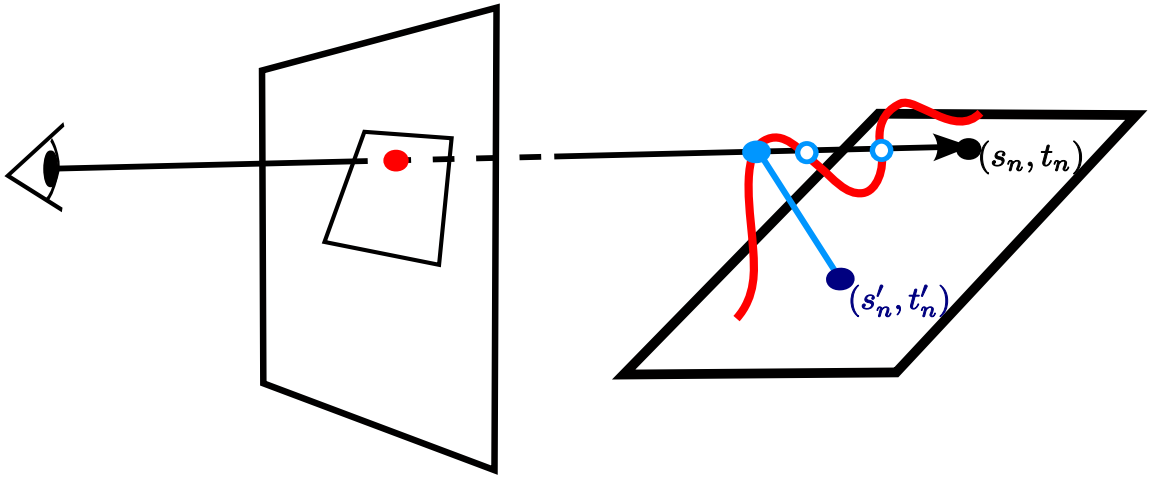
# Displacement on Plane: Geometry

Given an eye ray, finding the offset solution with height map.



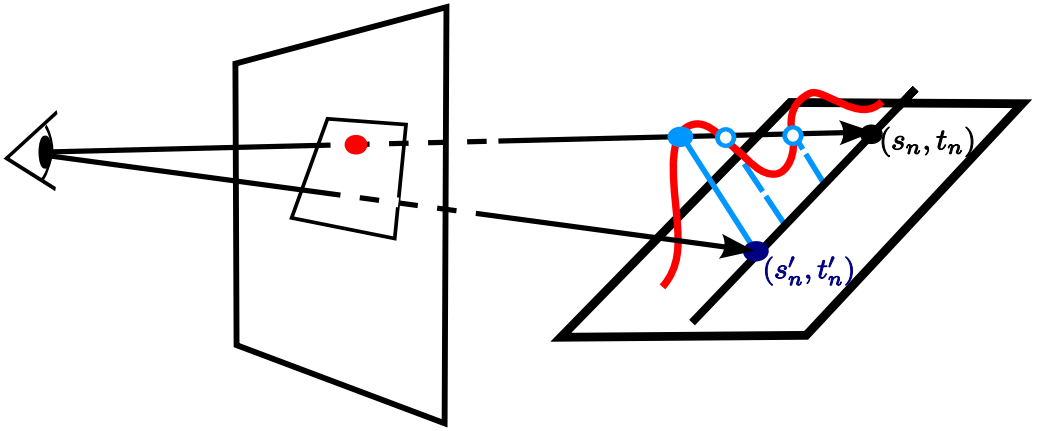
# Displacement on Plane: Geometry

There may be multiple intersections, find the closest one.



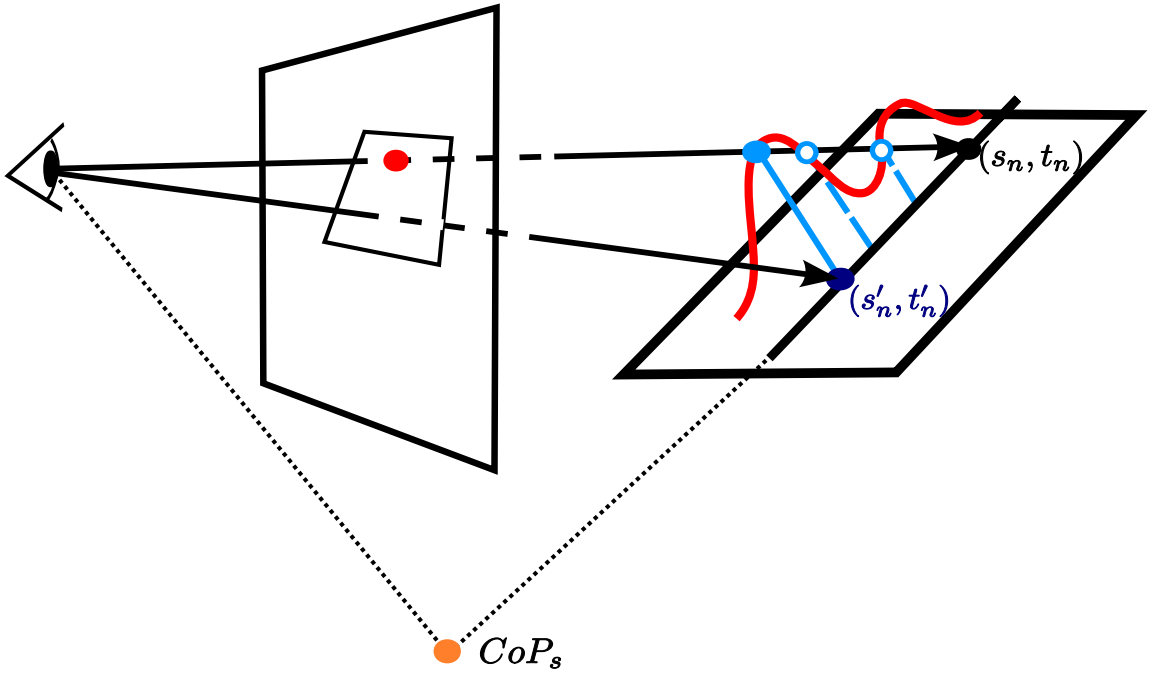
# Displacement on Plane: Geometry

The multiple intersections are offset along a common line. Thus, they lie on the same plane.



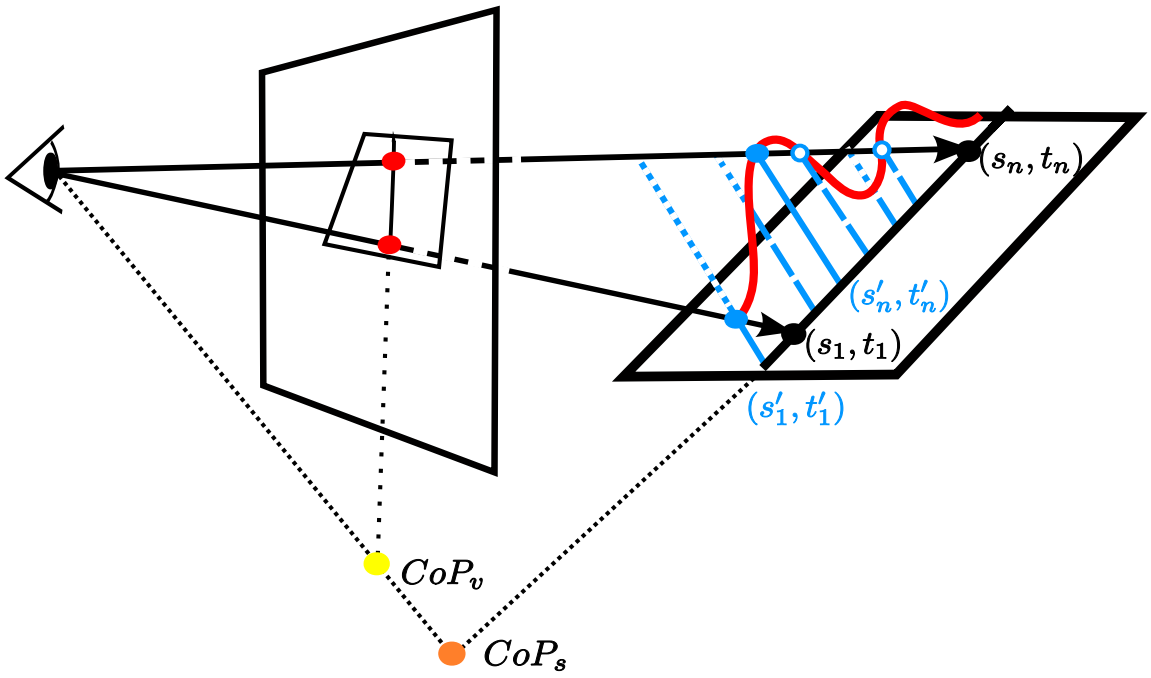
# Displacement on Plane: Geometry

$CoP_s$: closest point to eye on underlying surface.



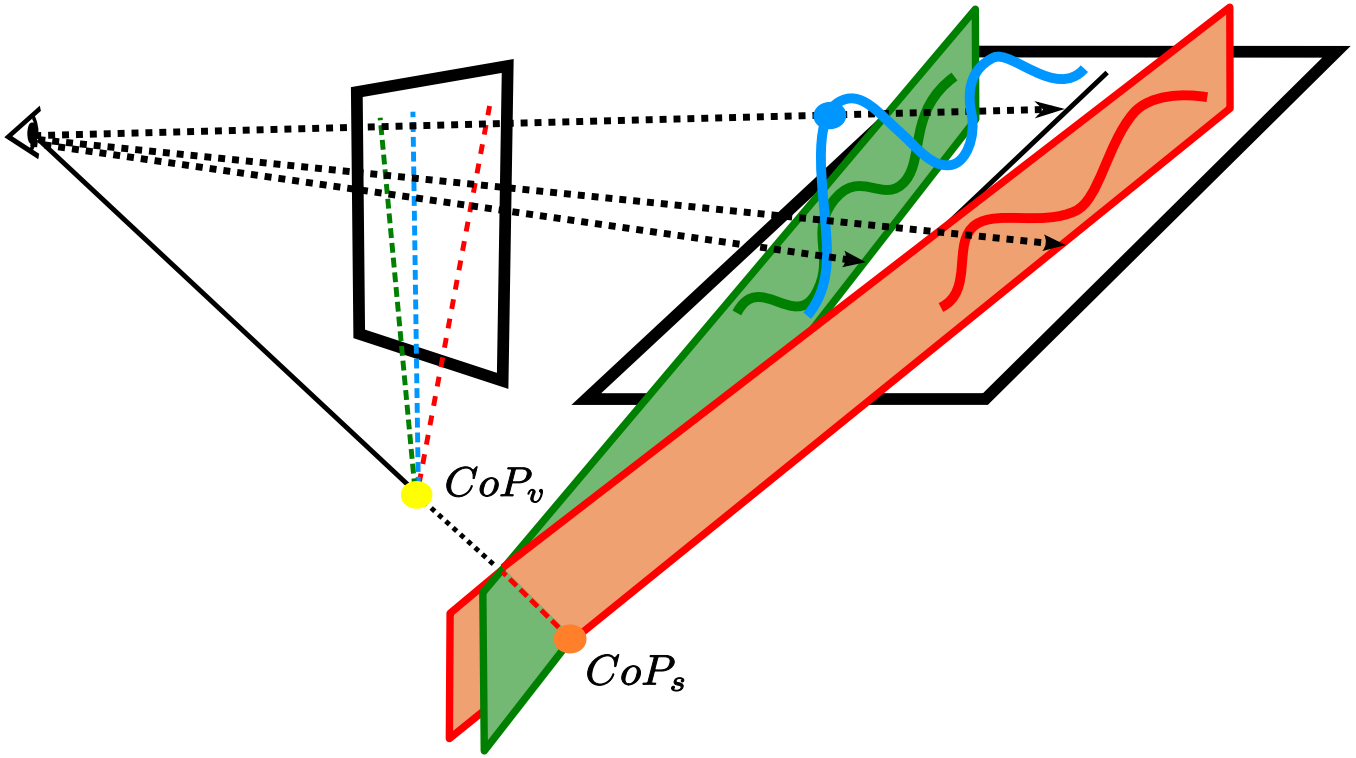
# Displacement on Plane: Geometry

Ordered $2D$ problem: spatial coherence can be exploited.



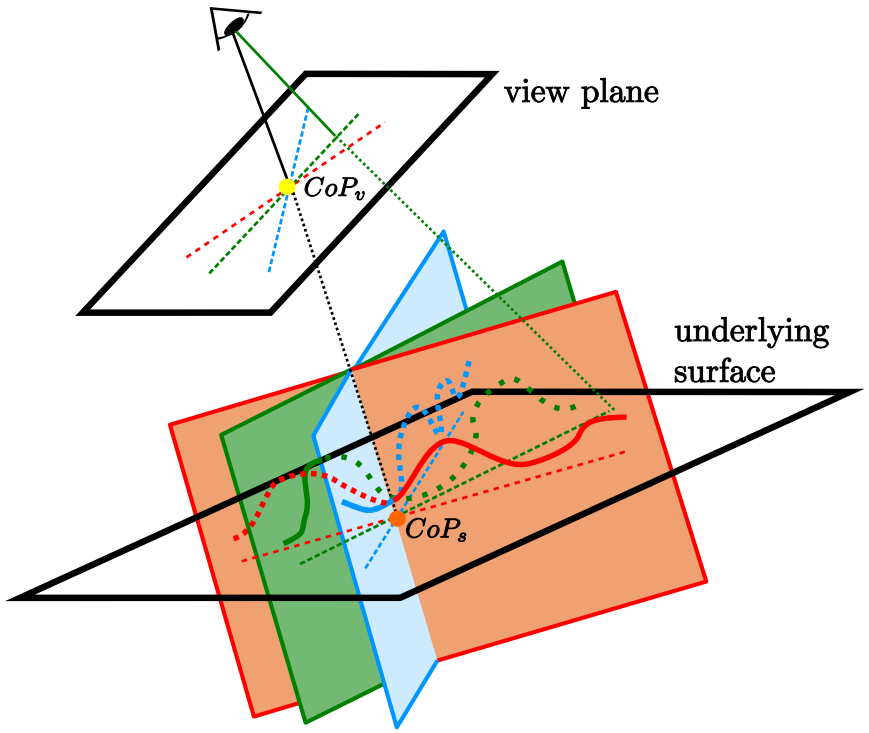
# Displacement on Plane: Visibility Lines

All underlying lines share the $CoP_s$.



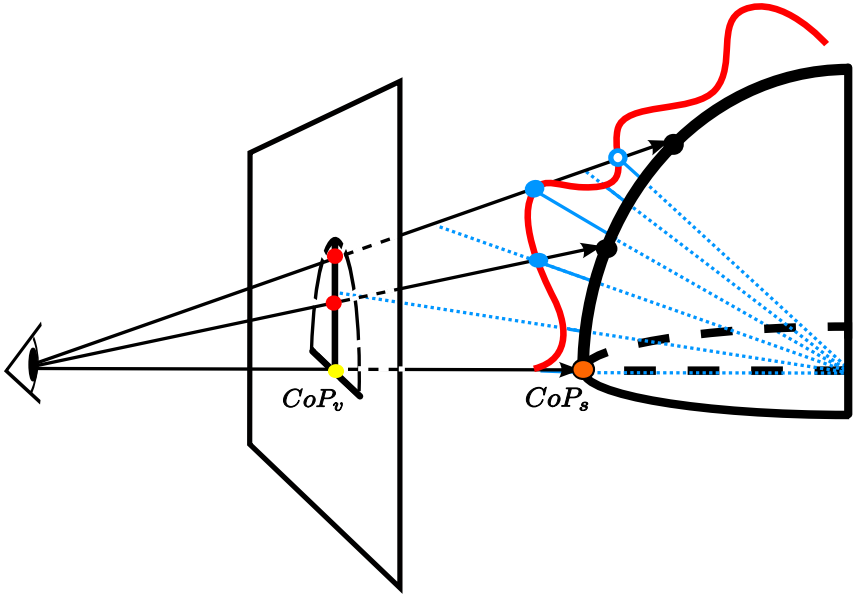
# Displacement on Plane: Visibility Lines

Pencil of lines on both underlying surface and view plane.



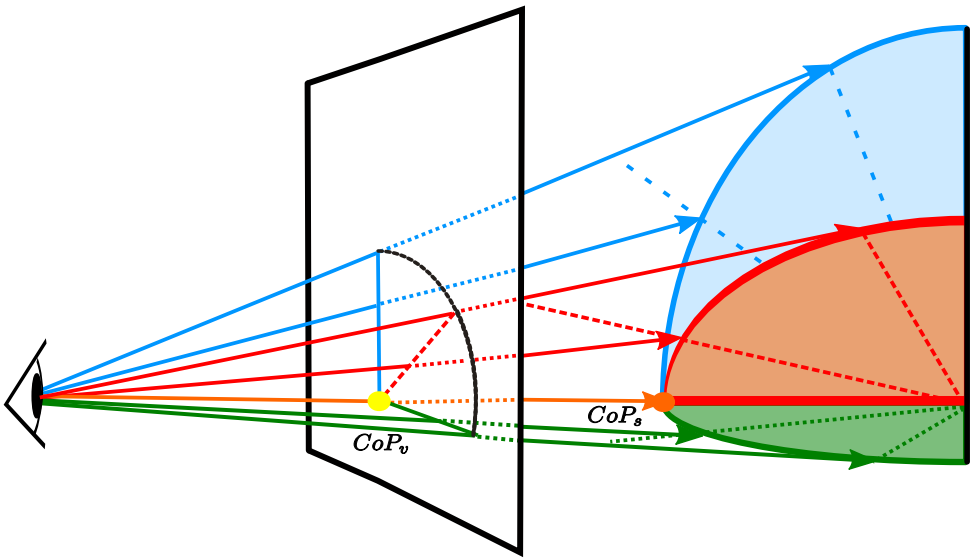
# Displacement on Sphere: Geometry

Eye ray multiple intersections are on great circle.



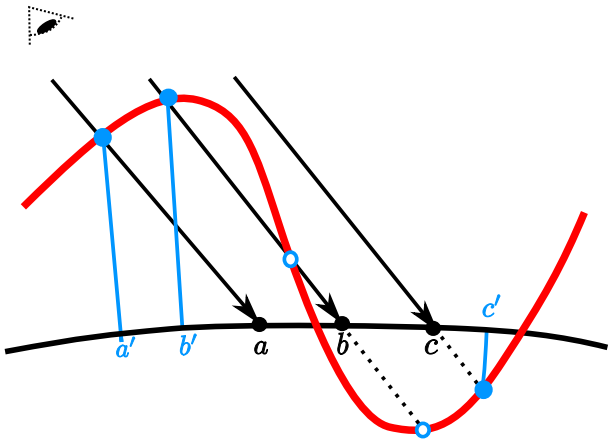
# Displacement on Sphere: Visibility Curves/Lines

Great circles project to a pencil of lines on view plane.

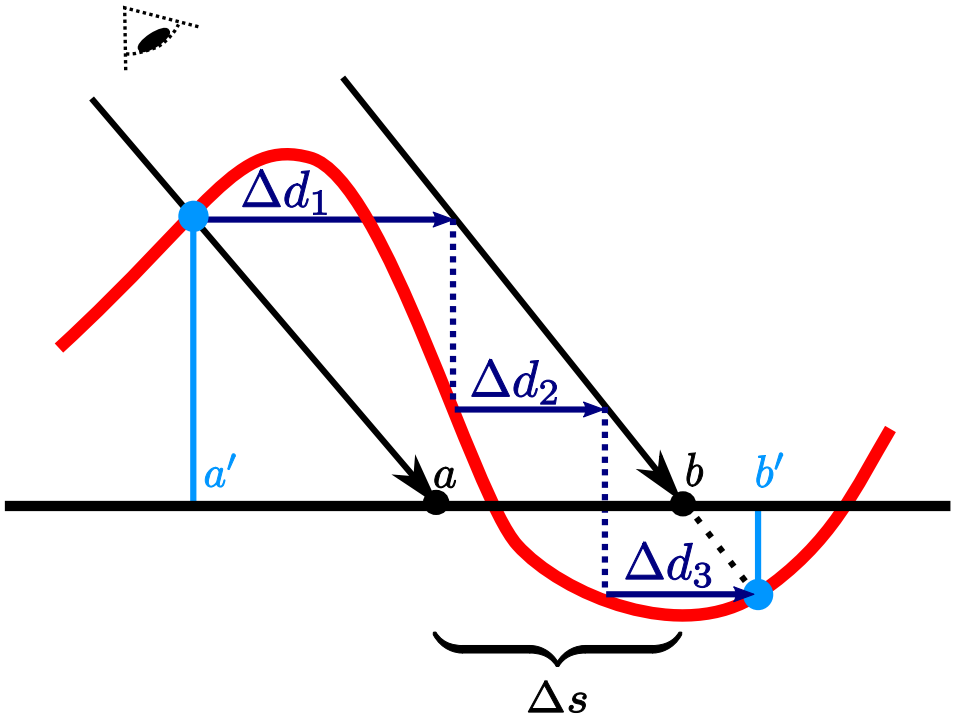


# Stepping Algorithm: Overview

- Separation by cutting planes.
- Stepping along each visibility line/curve from *CoP*.
- Spatial coherence:
  - from known solution, it is easy to find next one.
  - for multiple intersections, the first one is desired.



# Stepping Algorithm: Advance Step

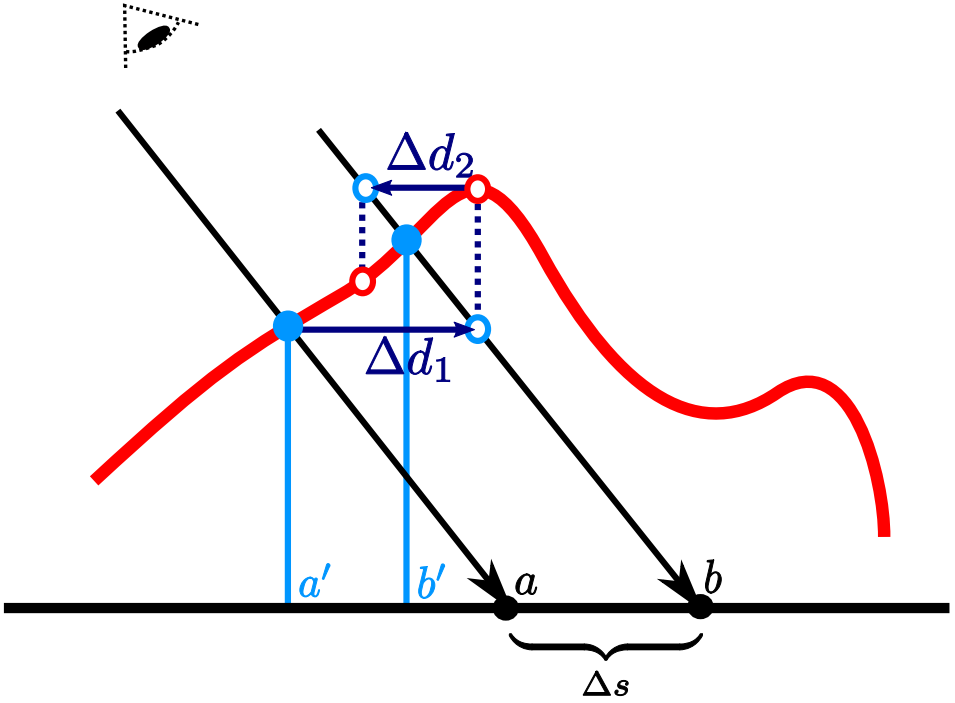



Sampling distance : $\Delta s$

Convergence threshold : $\epsilon$

Stopping criterion : $\Delta d \leq \epsilon \Delta s$

# Stepping Algorithm: Reverse Step

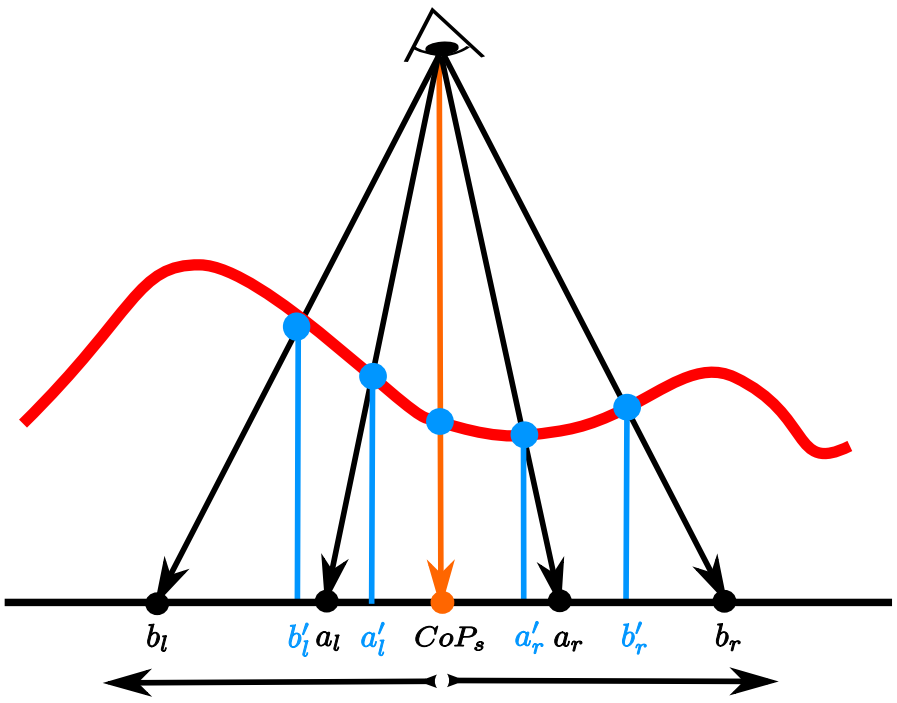



Reverse direction identified.
Solution found using binary search on $\Delta d$.

# Stepping Algoritm: Centered First Solution

$CoP_s$ inside underlying surface:

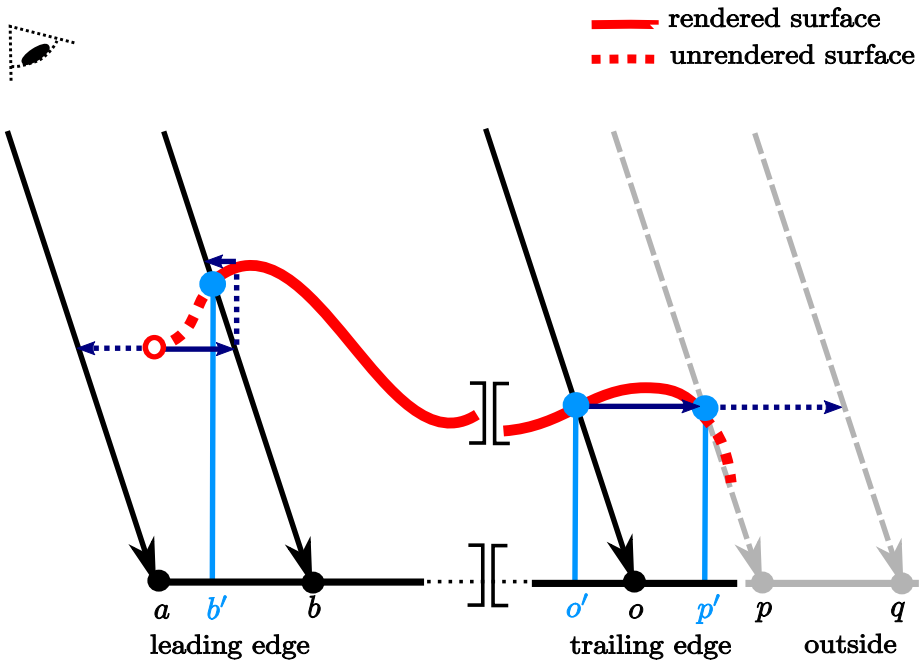
$CoP_s = 1^{st}$ solution for all curves.



# Stepping Algoritm: Boundary Solutions

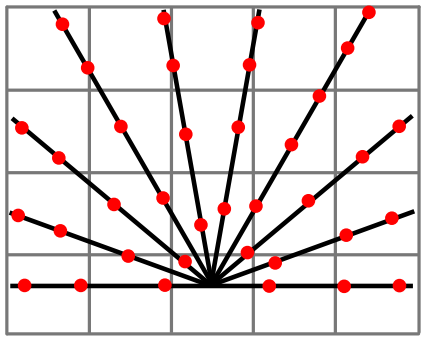
$CoP_s$ outside the underlying surface:

- at leading edge, some eye rays are not used,
- at trailing edge, extra eye rays may be needed.

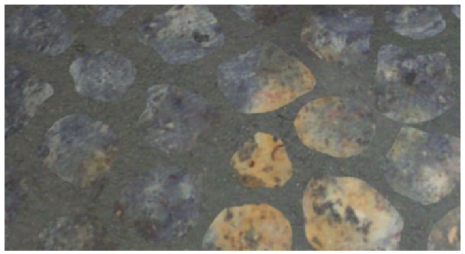


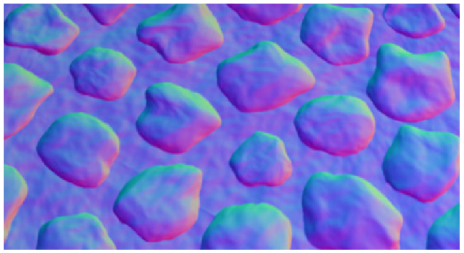
# Implementation

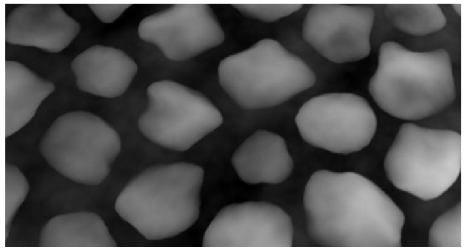
- Transformation between points on the view plane and on the underlying surface.
- Sampling algorithm on view plane visibility lines to get:
  - antialising using supersampling, and
  - uniform density of samples on view plane.

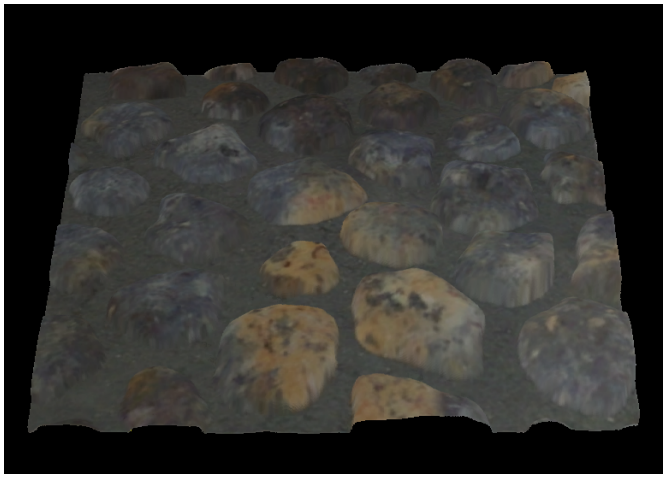


# Results: Plane

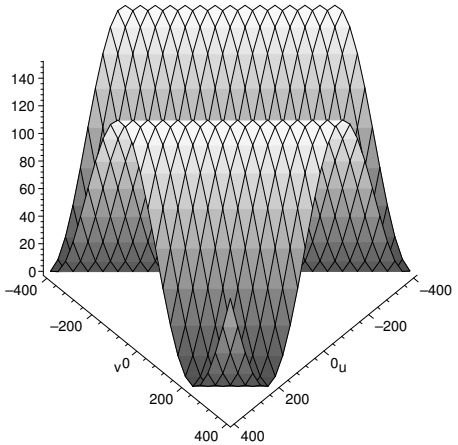


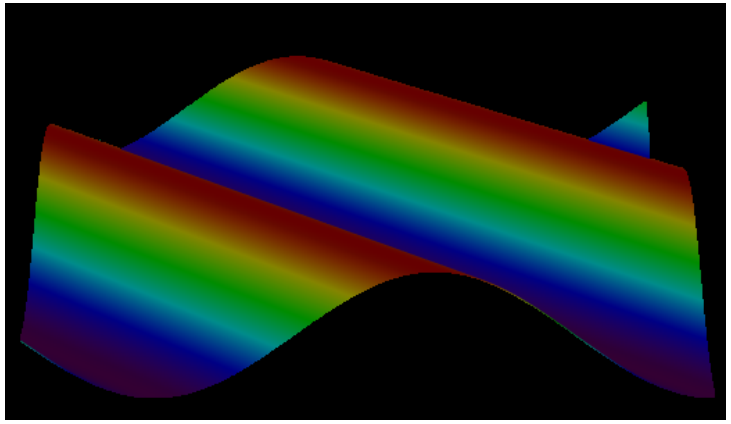






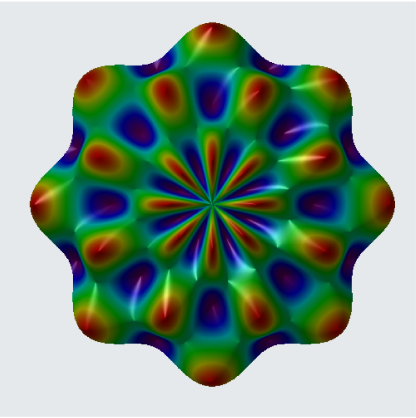
# Results: Plane

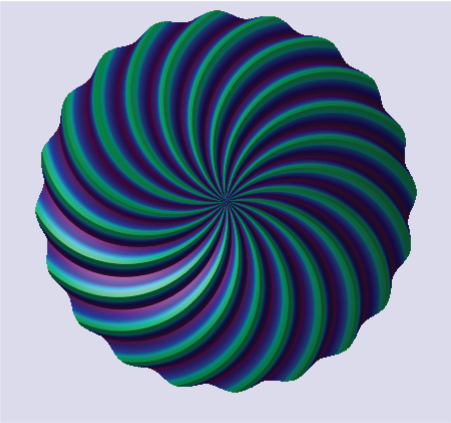


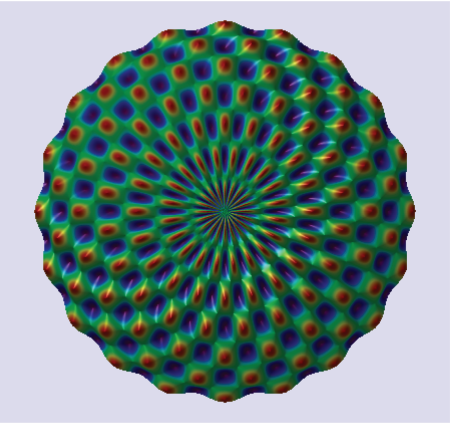



$$L\sin^2(eu + ev)$$

# Results: Sphere

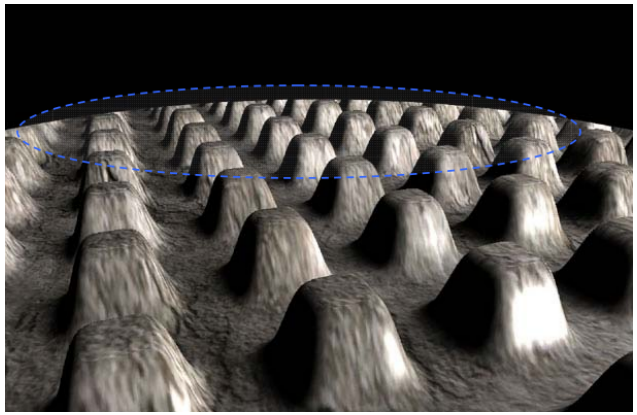






# Related Work

- 2D techniques
  Relief Texture: orthogonal displacements per texel, pre-warp the image.[Oliveira00]
- 3D techniques
  - View-Dependent Displacement Mapping: parallax effect by storing texel relationship from several viewing directions.
  - Ray-tracing: find intersection along eye ray with height. [Wang03][Donnelly05]

# Geometric Displacement Contributions

- Identify a geometric property of displacement mapping.
- Made an algorithm closer to texture mapping.
  - Stepping on a $2D$ map.
  - Well-defined parallelism.
- Provide better silhouettes.

# Future Work

- Parallelize the implementation.
- Generalize the underlying geometry.
- Better understand the relationships between $\epsilon$, viewing angle, and sampling.

# References


Donnelly05 W. Donnelly, *Per-Pixel Displacement Mapping with Distance Functions*, GPU Gems2, 2005.

Oliveira00 M. Oliveira et al, *Relief Texture Mapping*, Siggraph 2000.

Wang03 L. Wang et al, *View-Dependent Displacement Mapping*, Siggraph 2003.