Discontinuous Displacement Mapping for Volume Graphics

Carlos D. Correa, Deborah Silver

Rutgers, The State University of New Jersey

Min Chen

University of Wales, Swansea, UK

Introduction

- One key issue in graphics is the rendering of cuts and deformations

Previous Approaches

- Physically based, e.g., finite elements, mass-spring models, meshless methods
  - Nealen et al. 2005
  - Volumes → typically via a proxy mesh
- Non-physically based
  - Surface → free-form , procedural, displacement-based
  - Volume → free-form (Westermann, 2000)
    - → Ray deflectors (Kurzion & Yagel, 1997)
    - → Spatial TFs (Chen et al, 2003)
    - → Volume Browsing (McGuffin, 2003)

Our Approach

$$M\ddot{u} + D\dot{u} + Ku = f_{ext}$$

Physics-Based Deformation

Illustration/Image ← Computer Generated Image ← Physics (Time integration)

Illustration/Image → Physics (Simplification)

Illustrative Deformation

Illustration/Image → Computer Generated Image

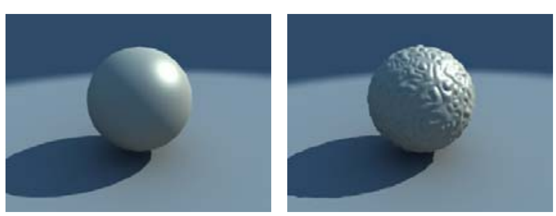
Concept, abstraction
Gross specification

Combination
refinement

Goal

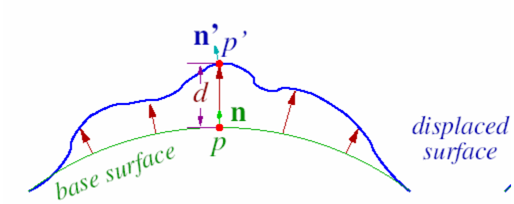
- To start with an illustration of what we want and *put it* in the object
- Similar to displacement maps

Traditional Displacement Mapping



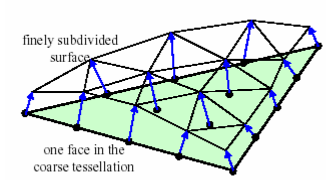
V-ray rendering system http://www.spot3d.com/vray

- Commonly used to add details to a base surface
- Applied along normal
- Usually continuous



Traditional Displacement Mapping (cont.)

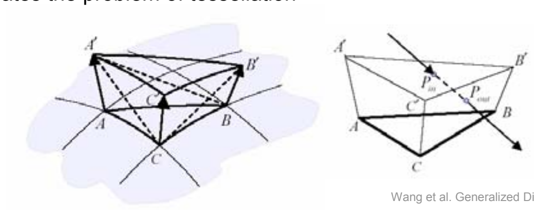
- Surface subdivision



Lee et al. Displaced Subdivision Surfaces

  – Limitations of surface representation can be solved using a tetrahedral mesh: further complicates the problem of tessellation

- Ray tracing



Wang et al. Generalized Displacement Maps

Cuts

- Difficult to model cuts and large deformations
- Surface Subdivision → Needs re-meshing
- Ray Tracing → Handle intersection with new surface, can't model large unorthogonal deformations
- Ray Deflectors → Difficult to model surface of cut as rays get deflected
- For ray tracing/warping, is easier to render cuts and deformations with inverse mapping

Solution

- 3D Displacements: model large deformations and cuts (no re-meshing)
- Inverse Mapping: High resolution rendering of cuts
- Directly on volumes: need an “inside” to properly model cuts

Discontinuous Displacement Mapping

- Given a common reference coordinate frame Λ
- A position function P and a displacement D

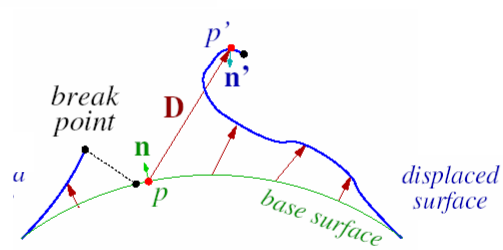
$$P'(\lambda) = P(\lambda) + \overrightarrow{D}^{\triangleright}(\lambda) \qquad \lambda \in \Lambda$$

- Since we use inverse mapping

$$P(\lambda) = P'(\lambda) + \overrightarrow{D}^{\triangleleft}(\lambda)$$

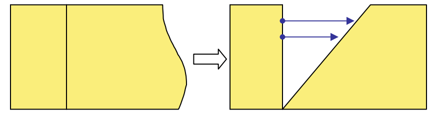
- We use $\Lambda = \mathbb{E}^3$

$$\boxed{\mathbf{p} = \mathbf{p}' + D(\mathbf{p}')}$$

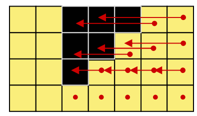


Displacement Setup

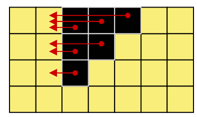
- Specify forward displacement



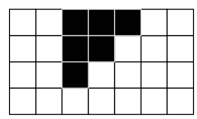
- Sample its inverse at discrete positions



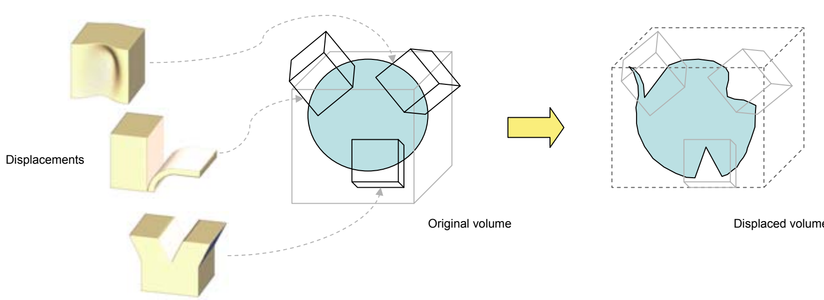
- What about empty space (due to cuts)?
  - Define displacement there too (to maintain C^0 continuity – correct tri-linear interpolation)



  - Define alpha map (represent cut geometry)

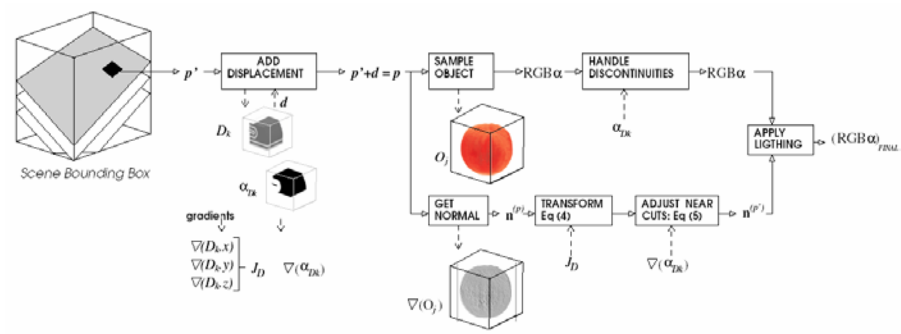


Displacements



- Define desired detail and “add” to scene
- Like displacement maps, we “add” the displacement to the volume. We then render this new volume

Rendering Pipeline



- Sample scene bounding box (resulting scene) then find opacity and color attributes of each point using inverse transformation

Problem: add lighting

- Displaced volume with no lighting

Using Original Normals

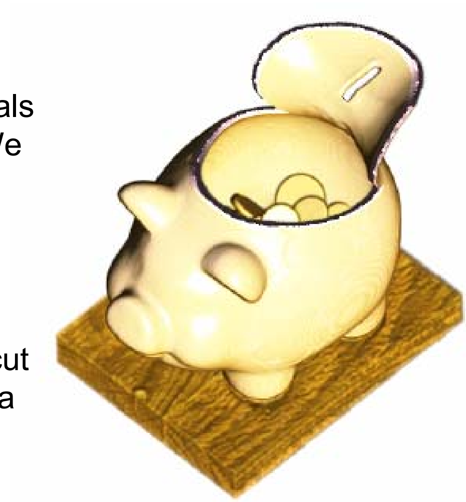
- Using the original normals result in incorrect lighting

Transformed Normals

- Barr [1984] derived transformation of normals for *forward mapping*. We derive for inverse displacements:

$$\vec{\mathbf{n}}^{(p')} = (\mathsf{I} + \mathsf{J}_D^{(p')})^\top \vec{\mathbf{n}}^{(p)}$$

- Normals on the rim of cut are still incorrect since a new surface has been created



Adjusted Normals near Cuts

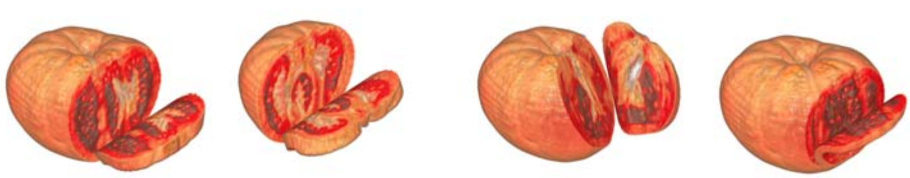
- Blend with normal of alpha map to handle normals at discontinuities

$$\vec{\mathbf{n}}^{(p')} = \omega(\mathrm{I}+\mathrm{J}_D)^\top \vec{\mathbf{n}}^{(p)} + (1-\omega)\nabla(\alpha_{D_k})^{(p')}$$

Discussion I: Interactive Manipulation

- Displacements can be placed and rotated within the volume. This is done via linear transformations, represented as 4x4 matrices.

$$\mathbf{p} = \mathrm{M} \times \left(\mathrm{M}^{-1}\mathbf{p}' + D(\mathrm{M}^{-1}\mathbf{p}')\right) = \mathbf{p}' + \mathrm{M} \times D(\mathrm{M}^{-1}\mathbf{p}')$$

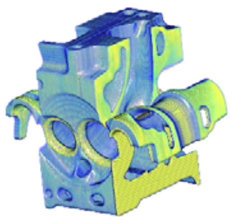


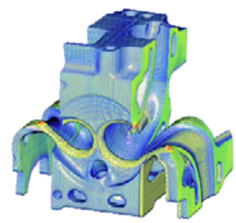
translation rotation scaling

Discussion II: Multiple Displacements

- Multiple Displacements can be placed, e.g., through addition of displacements

$$\boxed{\mathbf{p} = \mathbf{p}' + D_1(\mathbf{p}') + D_2(\mathbf{p}')}$$





Discussion III: Composition

- One of the advantages of Displacement Maps is the ability to operate algebraically, e.g., through composition.

$$\boxed{\begin{aligned} \mathbf{p}_1 &= \mathbf{p}' + D_1(\mathbf{p}') \\ \mathbf{p} &= \mathbf{p}_1 + D_2(\mathbf{p}_1) \end{aligned}}$$

- Composition is, in general, not commutative

D_1: ripples
D_2: peel

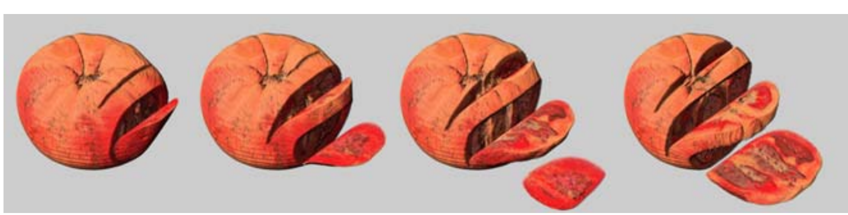
$\mathbf{p} = \mathbf{p}' + D2(\mathbf{p}' + D1(\mathbf{p}'))$

$\mathbf{p} = \mathbf{p}' + D1(\mathbf{p}' + D2(\mathbf{p}'))$

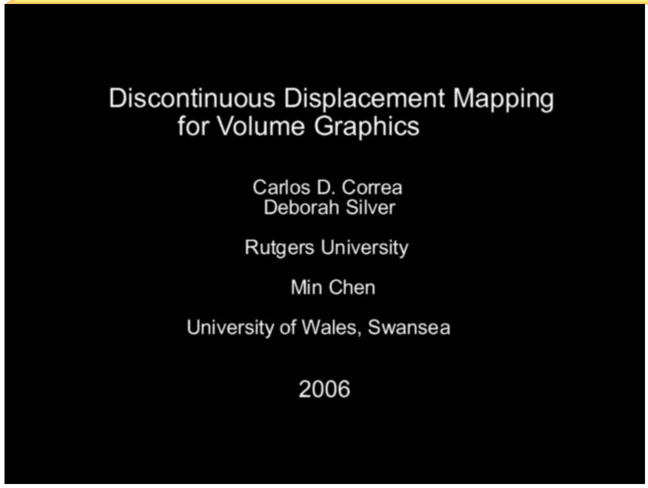
Volume Datasets

Teddy bear	256x256x224
Piggy Bank	190x190x134
Bunny	256x256x256
Tomato	256x256x162
Engine	256x256x128

Results



Results



Discussion IV: GPU

- GPU texture memory size
  - Current displacements occupy 320 KB - 6 MB. Still much smaller than the volume
  - Complex 3D displacements might require larger sizes

Semantics

- Displacement is specified with no regards of semantics of data (planar cuts)
- Need a mechanism to preserve/constrain to features in the dataset (*to appear, IEEE Visualization 2006*)

Conclusions

- Illustrative Deformation is a powerful technique to simulate effects in volume graphics, such as fracturing, slicing, deforming and cutting.
- Can be efficiently implemented via *Discontinuous Displacement Maps*
- We devised a collection of techniques for implementing this method in real-time in contemporary GPUs
- Applications: Special effects, VR, Illustration, Surgical Planning, Games.

Thanks!

More info

http://www.caip.rutgers.edu/~cdcorrea/displacement