Distortion-Free Displacement Mapping

Tobias Zirr $^1$ and Tobias Ritschel $^2$

$^1$ Karlsruhe Institute of Technology
$^2$ University College London
Displacement Mapping causes stretching

► Displacement mapping on a plane
Displacement Mapping causes stretching

Displacement mapping on a plane
Can we have proper texture unwrapping?

Per-texel texture coordinates?
Previous Work

“Indirection Mapping for Quasi-Conformal Relief Texturing”
[McGuire and Whitson, 2008]

- Spring relaxation on mesh vertices
  (Iterative, Hierarchical)
- Resampling of texture coordinates
Optimization by physical models can give good results.

Previous work [McGuire et al., 2008]

Also in graph embedding:

[Brandes, 2003]
Optimization by physical analogies can sometimes give good results.

Previous work [McGuire et al., 2008] also in graph embedding:

- Unknown step sizes
- Oscillations?
- Annealing?!
- Good convergence speed?!

- Indirectly defined energy
- Guarantees / objectives?!

[Brandes, 2005]
Energy minimization is more flexible and controllable

- Directly formulate your problem
  - E.g., metric on per-pixel texture coordinates
  - No resampling required (vs. previous work)
Energy minimization is more flexible and controllable

Directly formulate your problem

E.g., metric on per-pixel texture coordinates

No resampling required (vs. previous work)

Flexible constraints

Boundaries

Artist Intervention

Tradeoffs like angle vs. area

Avoid invalid results like collapsed triangles
Conformal mappings
- Preserve angles
- Guaranteed existence for “disks”
- May require area stretching

Least-Squares Conformal Mapping (LCSM) in Blender

[Hormann et al., 2008]
Angle preservation
Conformal vs. Isometric Mappings

- Conformal mappings
  - Preserve angles
  - Guaranteed existence for “disks”
  - May require area stretching

- Isometric mappings
  - Also preserve area
  - No general existence guarantee
  - Approach as closely as possible

Least-Squares Conformal Mapping (LCSM) in Blender

L2 Stretch Minimizing Parameterization [Sander et al., 2001]
Controllable Area Preservation (via energy exponent $\theta$)

$\theta = 0.3, \theta = 1.0, \theta = 3.0$
Texture Space

3D Space

Texture Mapping

Undistortion Energy [Degener et al., 2003]

[Hormann et al., 2008]
Undistortion Energy [Degener et al., 2003]

Texture Space

3D Space

Texture Mapping

[1] Hormann et al., 2008
Undistortion Energy [Degener et al., 2003]

Texture Space

Texture Mapping

3D Space

- $\sigma_1$ longest, $\sigma_2$ shortest stretch
- Ratio: angle preservation
- Product: change in area

[Hormann et al., 2008]
Texture Space

$\Omega$

$\mathbf{u}$

Texture Mapping

3D Space

$p$

$\sigma_1 U_1$

$\sigma_2 U_2$

Degener et al.: 

$$\left(\frac{\sigma_1}{\sigma_2} + \frac{\sigma_2}{\sigma_1}\right) \cdot \left(\sigma_1 \sigma_2 + \frac{1}{\sigma_1 \sigma_2}\right)^\theta$$

$\sigma_1$ longest, $\sigma_2$ shortest stretch

$\Rightarrow$ Ratio: angle preservation

$\Rightarrow$ Product: change in area

[Hormann et al., 2008]
Massively parallel gradient descent (GD) per texel
Massively parallel gradient descent (GD) per texel

- Find local optimum along gradient line for each texel

Gradient = direction of highest local change

Energy

Ideal Step Size
Massively parallel gradient descent (GD) per texel

- Find local optimum along gradient line for each texel

Gradient = direction of highest local change

Energy

Ternary line search of local optimum
Massively parallel gradient descent (GD) per texel

Find local optimum along gradient line for each texel

Gradient = direction of highest local change

Ternary line search of local optimum
Massively parallel gradient descent (GD) per texel

- Find local optimum along gradient line for each texel

Gradient = direction of highest local change

Ternary line search of local optimum
Parallel local optimum for constant neighborhood (otherwise oscillations!)

- One thread per correction texel quad
- Split across four iterations

\[4 \ast i + 0, 4 \ast i + 1, 4 \ast i + 2, 4 \ast i + 3\]
GLSL implementation can run interactively 😊

[MOVIE]
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
GLSL implementation can run interactively 😊
We don’t store per-texel coordinates, but per-texel offsets!

- Small magnitude (compact quantization)
- Works across texture tiling borders (preserving pixel derivatives)
Store relative per-texel coordinates!

Linear texcoord + per-texel offset

Albedo A

Displacement D

Per-texel texcoord offsets

Warped A
Simplest solution works OK:

Mip Maps for both Correction Offsets & Color Textures
Optimization can lead to unwanted distortion

Perturbed straight lines
Optimization can lead to unwanted distortion
Correlations between texture and geometry
Optimization can lead to unwanted distortion

- Correlations between texture and geometry
- Perturbed straight lines

We allow to mark parts of textures fixed by an additional fixation energy term
Example implementation: [https://github.com/tszirr/ic.js/tree/master/unwarp](https://github.com/tszirr/ic.js/tree/master/unwarp)
Questions?

Contact

Tobias Zirr

tobias.zirr@alphanew.net
Twitter: @alphanew
http://www.alphanew.net

Tobias Ritschel

http://www.homepages.ucl.ac.uk/~ucactri

Project sources:
http://www.alphanew.net
Default evaluation of displacement: on texel boundaries

Default evaluation of correction offsets: on texel centers
Match undistortion with displacement grid
- Use half texel offset
  - Apply during correction optimization
  - Undo during correction evaluation

Fixes resampling artifacts