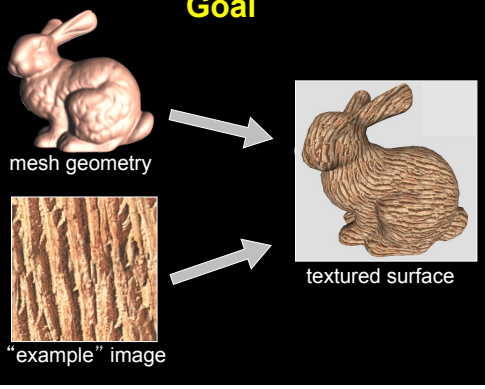
Lapped Textures

Emil Praun *Princeton University*
Adam Finkelstein *Princeton University*
Hugues Hoppe *Microsoft Research*

[SIGGRAPH 2000]

Goal



Goal

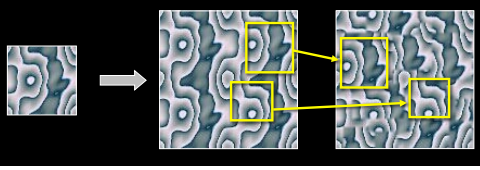
- Little user effort
- No apparent seams
- No obvious periodicity
- Low distortion
- Local texture control
- Anisotropy

Previous 2D Texture Synthesis

- Histogram equalization [Heeger ’96]
- Laplacian block shuffling [de Bonet ’97]
- Pixel template matching [Efros ’99] [Wei ’00]

Previous 2D Texture Synthesis

- Histogram equalization [Heeger ’96]
- Laplacian block shuffling [de Bonet ’97]
- Pixel template matching [Efros ’99] [Wei ’00]
- Random pasting of image blocks [Xu ’00]



Previous 3D Texturing

Volumetric textures:

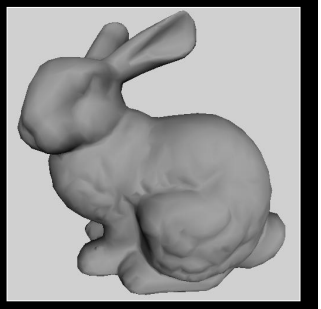
- Noise functions [Perlin ’85, Worley ’96]
- Solid textures by example [Ghazanfarpour ’96]

Synthesizing texture on a surface:

- Reaction-diffusion [Turk ’91, Witkin ’91]
- Cellular textures [Fleischer ’95]
- Covering surface with triangular tiles [Neyret ’99]

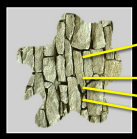
Approach

texture patch

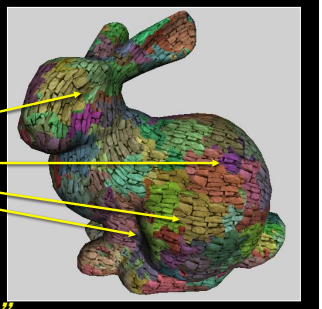


surface

Key Idea: Patch Pasting



texture patch



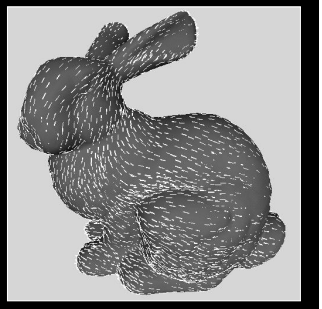
"lapped textures"

surface

PROCESS

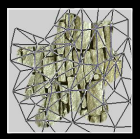
Algorithm

texture patch

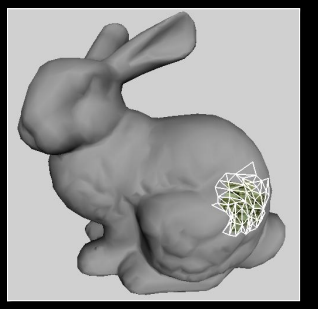


surface

Algorithm

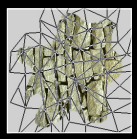


texture patch

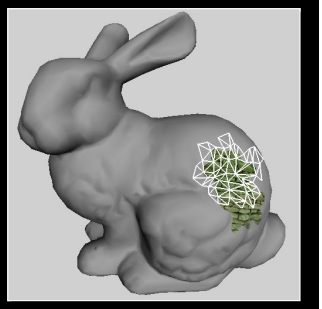


surface

Algorithm



texture patch



surface

Algorithm

texture patch

surface

Issues

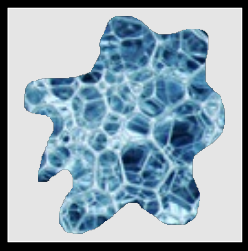
1. Texture patch creation
2. Specifying direction field
3. Surface patch growth
4. Patch parametrization
5. Face coverage estimation
6. Texture storage and rendering

Issues

1. **Texture patch creation**
2. Specifying direction field
3. Surface patch growth
4. Patch parametrization
5. Face coverage estimation
6. Texture storage and rendering

Texture Patch Creation

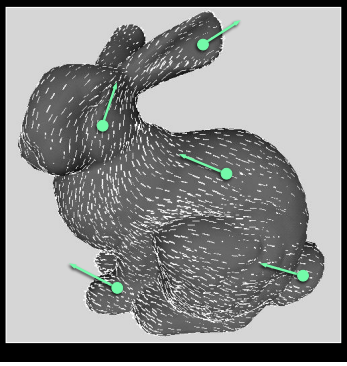
Less Structure → Splotch



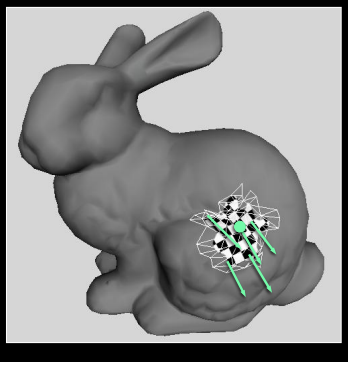
Issues

1. Texture patch creation
2. **Specifying direction field**
3. Surface patch growth
4. Patch parametrization
5. Face coverage estimation
6. Texture storage and rendering

Direction Field: User-specified



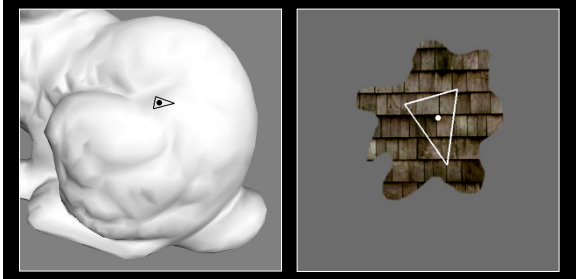
Direction Field: Local to Patch



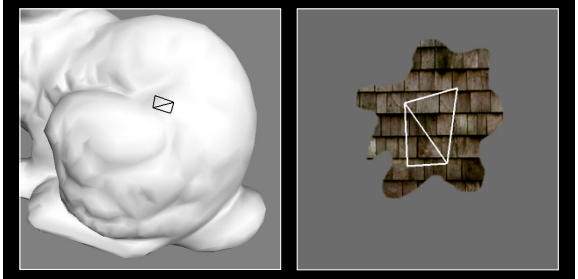
Issues

1. Texture patch creation
2. Specifying direction field
3. Surface patch growth
4. Patch parametrization
5. Face coverage estimation
6. Texture storage and rendering

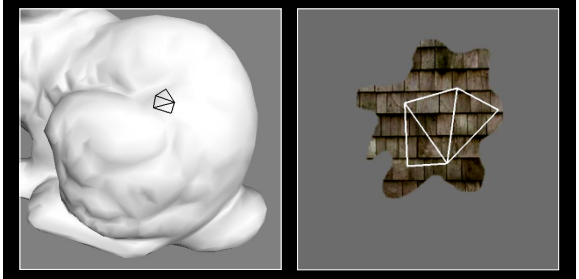
Patch Growth



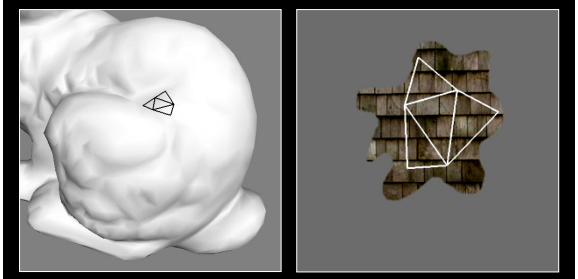
Patch Growth



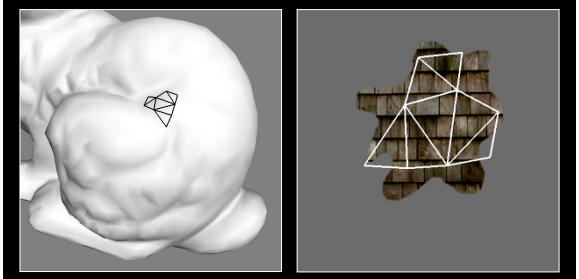
Patch Growth



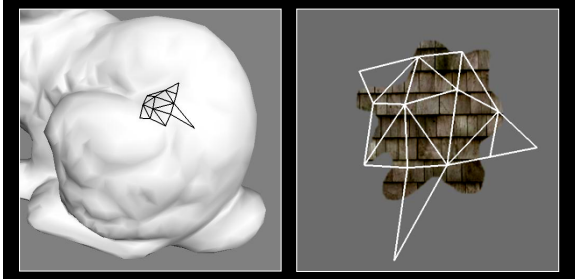
Patch Growth



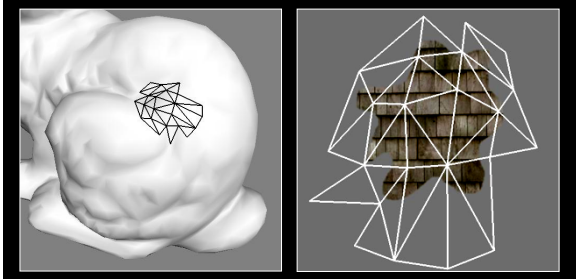
Patch Growth



Patch Growth



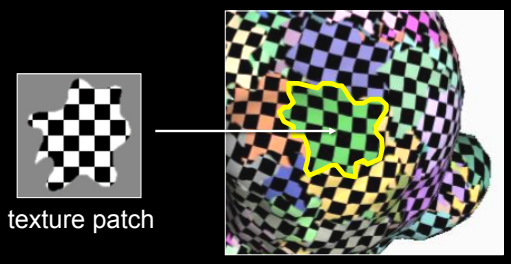
Patch Growth



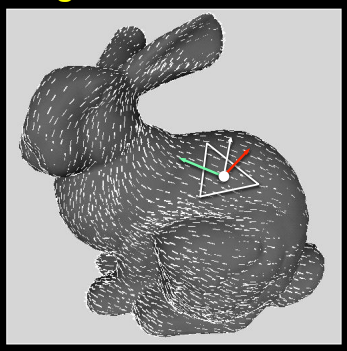
Issues

1. Texture patch creation
2. Specifying direction field
3. Surface patch growth
4. Patch parametrization
5. Face coverage estimation
6. Texture storage and rendering

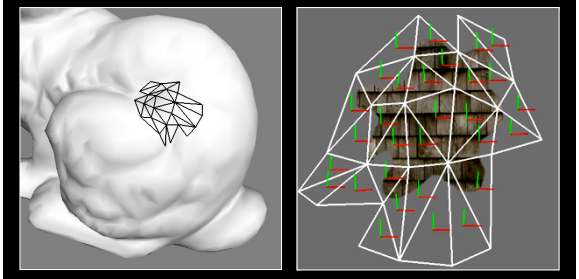
Align Patch to Direction Field



Tangential Vector Field

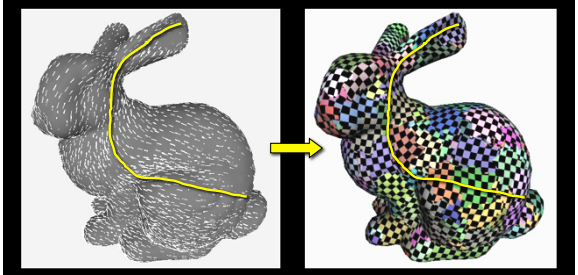


Optimizing the Parametrization



Least squares best match to unit axes
Sparse linear system. No explicit fairness functional

Result of Optimization

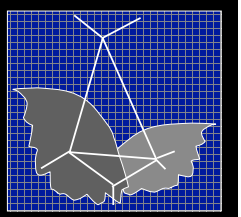


Issues

1. Texture patch creation
2. Specifying direction field
3. Surface patch growth
4. Patch parametrization
5. Face coverage estimation
6. Texture storage and rendering

Coverage estimation

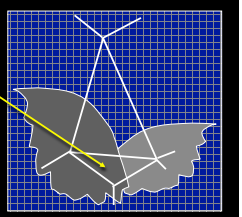
Render patch triangles
Flag covered triangles
Remember 1 pixel
per uncovered triangle



off-screen buffer

Coverage estimation

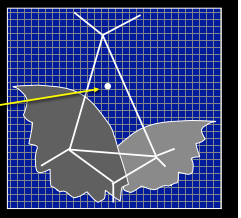
Render patch triangles
Flag covered triangles
Remember 1 pixel
per uncovered triangle



off-screen buffer

Coverage estimation

Render patch triangles
Flag covered triangles
Remember 1 pixel
per uncovered triangle



off-screen buffer

Issues

1. Texture patch creation
2. Specifying direction field
3. Surface patch growth
4. Patch parametrization
5. Face coverage estimation
6. Texture storage and rendering

Texture Storage and Rendering

Method 1: *Texture Atlas*

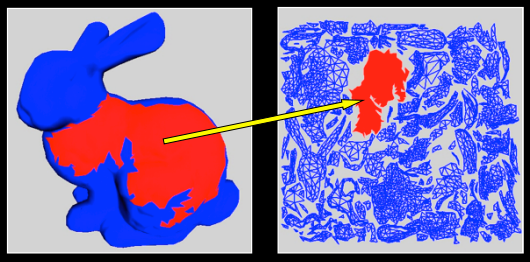
- Pre-composite into a global texture map.

-- OR --

Method 2: *Runtime pasting*

- Composite at run-time using hardware

Method 1: Texture Atlas



Patches of triangles with similar normals
2D packing problem for arbitrary polygons

Method 2: Runtime Pasting

Store vertex coordinates for each patch
Composite at run-time using hardware
May render triangles several times

Atlas vs. Runtime Pasting

Atlas

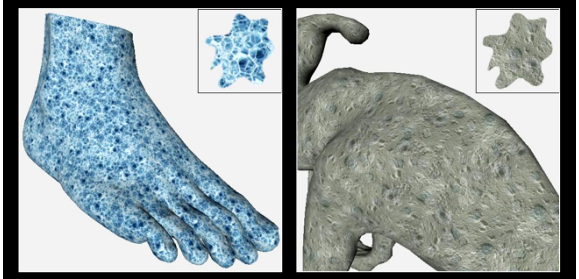
+ Faster rendering, more portable
+ Easy to paint unique details (eyes, nose on bunny)
– Sampling artifacts; user effort

Pasting

– Increases model complexity (~ ×1.6 –3)
+ Huge effective resolution
+ Reuse splotch parameterization for many textures

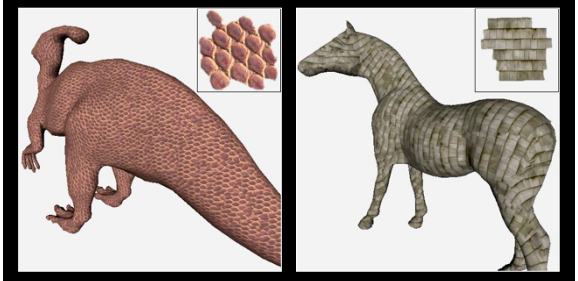
RESULTS

Results: Splotches

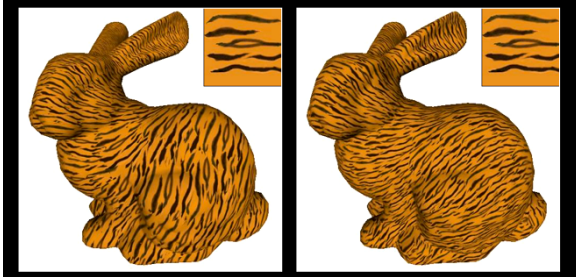


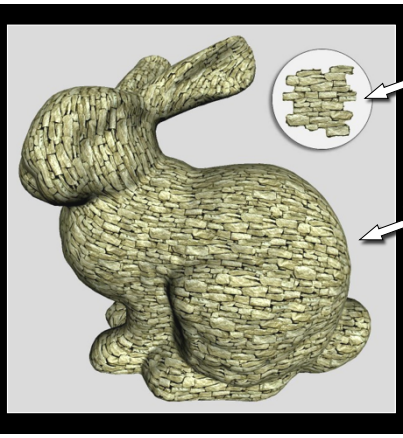
(completely automatic: no direction field)

Results: Anisotropic



Controlling Direction and Scale





Limitations

low-frequency components | boundary mismatches | direction field singularities

Video

Interactive Paint Demo

Timings

Texture patch creation: 1 min } Human effort
Specifying direction field: 15 min }

Surface patch growth
Patch parameterization } Preprocessing: 20sec – 6 min
Face coverage estimation }

Rendering: 25fps @ 1024^2

Pentium III 733MHz, GeForce graphics

Conclusions

Effective texture-by-example through:
- Overlapping texture patches
- Minimal edge blending

Aligning to direction field
→ fast optimization

Runtime pasting
→ high effective resolution

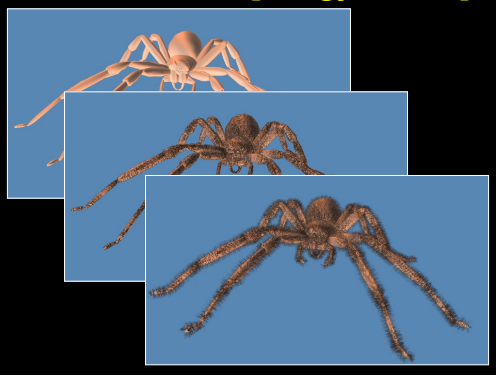
Future Work

Other texture types:
- Animated
- "Thick" (volumetric) textures → fur
- NPR rendering

Greater automation

Fine-tuning patch placement

Real-Time Fur [Lengyel 2001]



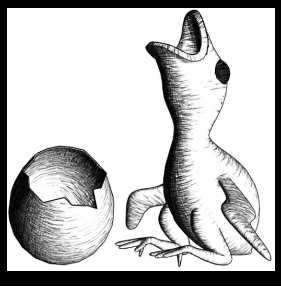
Real-Time Fur [Lengyel 2001]

Real-Time Hatching [Praun2001]

Stroke-based rendering of 3D models

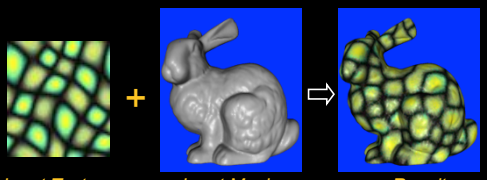
Strokes convey:

- tone
- material
- shape



Texture Synthesis over Arbitrary Manifold Surfaces [Wei2001]

Synthesize a surface texture by coloring mesh vertices



Input Texture *Input Mesh* *Result*

Texture synthesis on surfaces [Turk 2001]