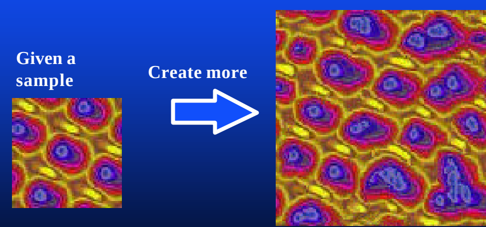
# Texture Synthesis on Surfaces

**Greg Turk**
GVU Center and College of Computing
Georgia Institute of Technology

## Texture Synthesis




## Previous Work (Texture Analysis/Synthesis)

Popat & Picard 93 - Multiscale neighborhoods
Heeger & Bergen 95 - Histogram matching
De Bonet 97 - Multiscale region selection
Efros & Leung 99 - Neighborhood sampling
Wei & Levoy 00 - Fast multiscale

## Synthesis on Surfaces

## Previous Work (Texturing Surfaces)

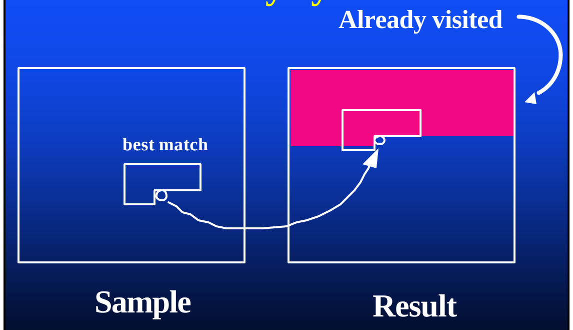
Perlin 85 - Solid texture & noise
Turk 91 - Reaction-diffusion
Fleischer et al. 95 - Cellular textures
Neyet & Cani 99 - Pattern-based texture
Praun, Finkelstein, Hoppe 00 - Lapped texture
Wei & Levoy 01 - Surface synthesis
Ying, Hertzmann, Zorin 01 - Surface synthesis

## Wei-Levoy Synthesis

Given: Original texture (*sample*)
Create: New image with similar texture (*result*)

Algorithm:
- Fill *result* with random pixels from *sample*
- Visit pixels in *result* image in raster-scan order
- Find best neighborhood match in *sample*
- Copy best match pixel from *sample* to *result*

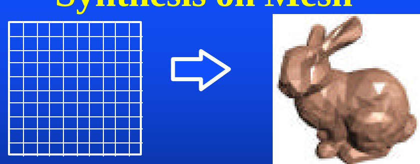
## Wei-Levoy Synthesis




## Speed/Quality Tradeoff

- Best results use big neighborhoods
- Speed proportional to neighborhood size
- Can mimic large neighborhoods using multi-scale methods (image pyramids)

## Synthesis on Mesh



- Colors at *vertices*, not pixels
- Define meaning of neighborhood
- Create equivalent of raster-scan order
- Multi-scale version (like image pyramid)

## Overview of Method

Given: Polygon mesh & sample image

Create: Textured mesh

Algorithm:

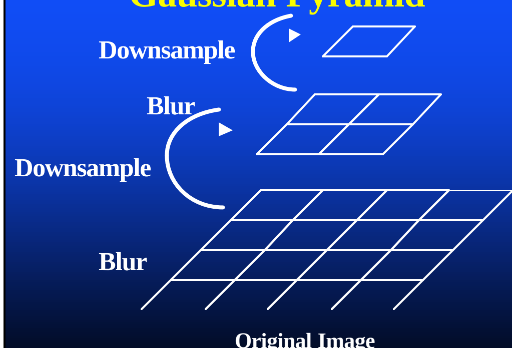
Create mesh hierarchy

Define vector field on mesh

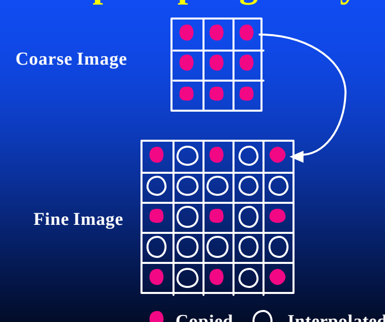
Calculate sweep distances

Perform multiscale synthesis

## Gaussian Pyramid




## Upsampling on Pyramid



## Image Pyramid Operations

- Low-Pass Filter (Blur)
- Downsample
- Interpolate
- Upsample

Want these operations on mesh hierarchy!

## Mesh Hierarchy

Mesh hierarchy:

- Meshes of same object, varying vertex count
- Some vertices on both coarse and fine meshes

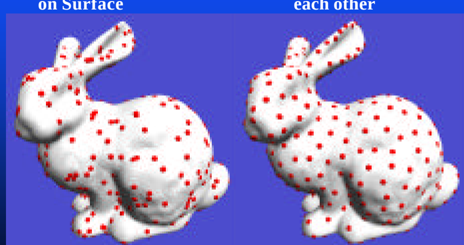
Typical vertex counts: 4,000 16,000 64,000 256,000

Many ways to create mesh hierarchies

One way: *Repulsion* to create irregular hierarchy

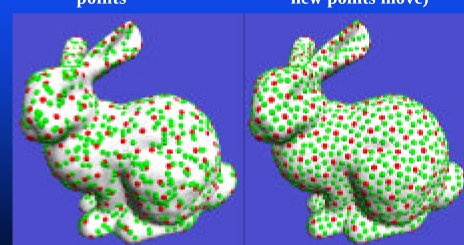
## Creating Mesh Hierarchy

Random points on Surface | Points repel each other



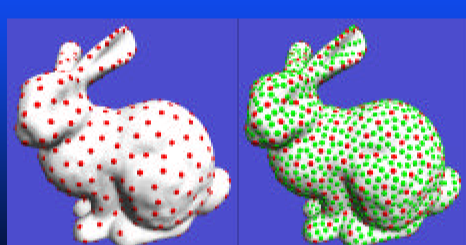
## Adding More Points

More random points | Repel again (only new points move)



## Mesh Hierarchy

Coarse mesh | Fine mesh

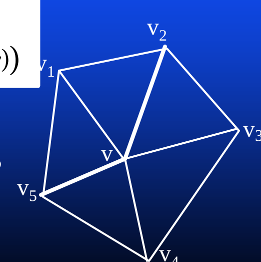


Delaunay triangulation gives mesh edges (not shown)

## Blur Colors on Mesh

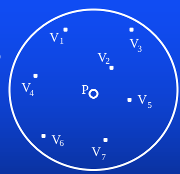
$$C_{new}(v) = C_{old}(v) + \sum_{i=1}^{n} w_i \left( C(v_i) - C_{old}(v) \right)$$

Repeat to blur colors



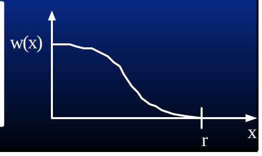
## Interpolation on Mesh

Color at point P?




Weighted average of nearby vertices:

$$C(p) = \frac{\sum_{i=1}^{k} w(|p - v_i|) C(v_i)}{\sum_{i=1}^{k} w(|p - v_i|)}$$




## Downsampling & Upsampling

Downsampling (fine to coarse):

- Blur colors
- Throw out fine mesh vertices

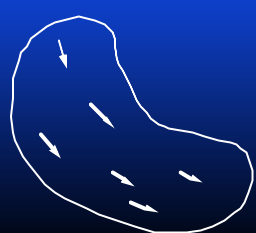
Upsampling (coarse to fine):

- Start with colors on coarse mesh
- Interpolate colors at fine vertices from coarse mesh

## Raster-Scan Order on Mesh

Create vector field on mesh

- One vector per vertex
- Vectors lie in tangent plane of surface



## Vector Field Creation

Method:

- All vertices start with zero-length vectors
- User places arrows on surface
- Freeze non-zero vectors and blur (like colors)
- Vectors spread to entire surface

Problem: Slow

Solution: Multi-scale approach

## Vector Field Creation

Multi-Scale Method:

- Downsample vectors to coarsest mesh
- Blur vectors on coarse mesh
- Upsample vectors to finer levels

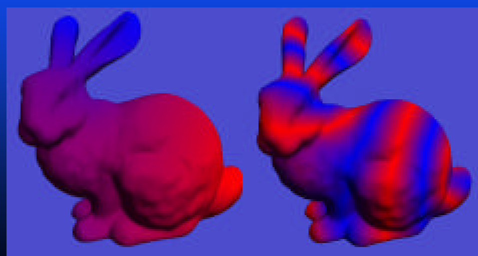
Like pull/push sparse interpolation from Lumigraph paper (Gortler et al. 96)

## Vector Field

## Sweep Distance

Measure distance from vector field start

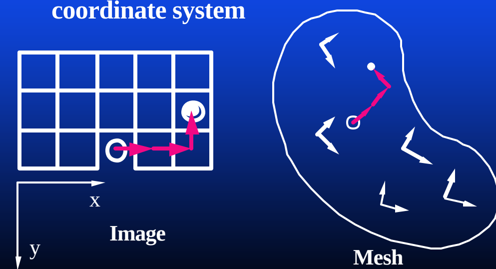
Sweep distances gives order to visit vertices



## Neighbors on Mesh

Use vector field to step over surface

Each vector and 90° rotation define local coordinate system

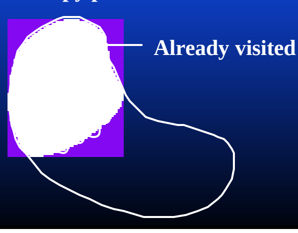



## Synthesis on Mesh

Copy colors randomly from image to vertices

Using sweep order, visit each vertex:

- Find best match neighborhood between vertex and sample image
- Copy pixel color from best match to vertex




## Multi-Scale Synthesis

Multi -Scale Algorithm:

Synthesize texture on coarse level

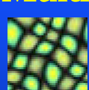
Repeat at each mesh level

- Extrapolate colors to finer mesh
- Refine pattern with another sweep

Typically use 3 or 4 mesh levels

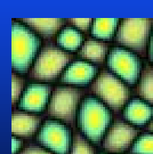
## Multi-level Synthesis

Sample:

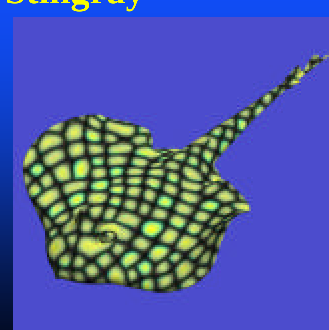


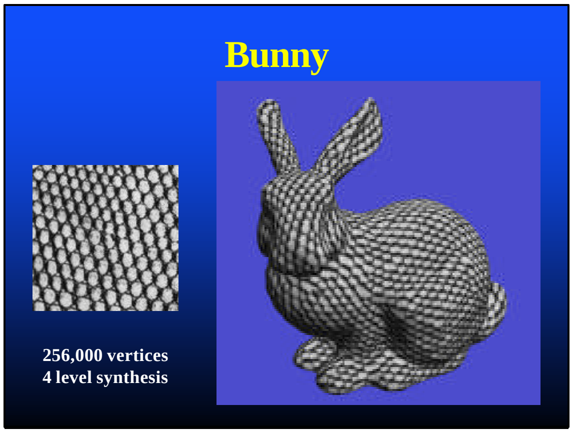
| Coarse | Medium | Fine |
|---|---|---|
| 1,000 vertices | 4,000 vertices | 16,000 vertices |

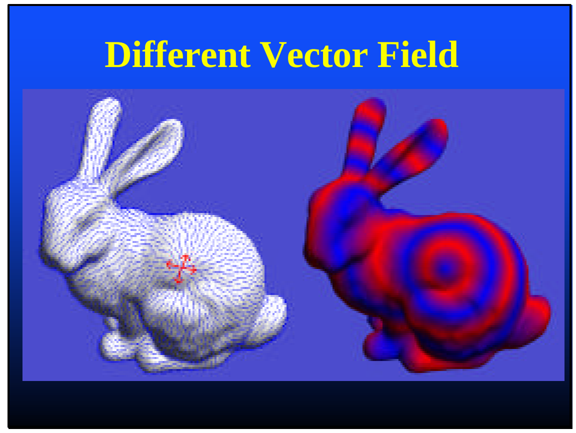
## Stingray

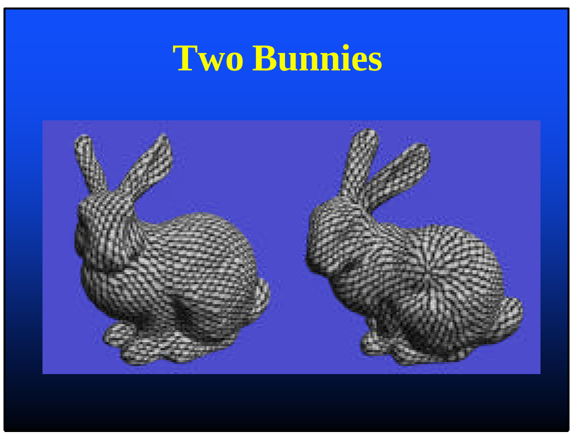


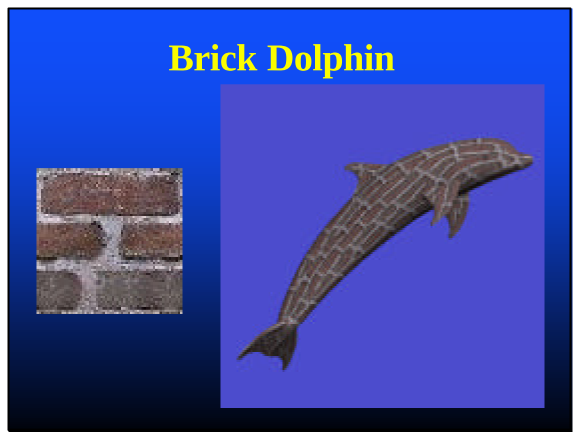
64,000 vertices
3 level synthesis

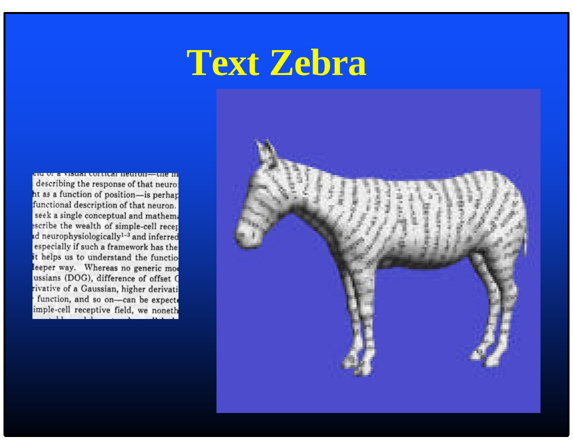



Bunny
256,000 vertices
4 level synthesis


Different Vector Field

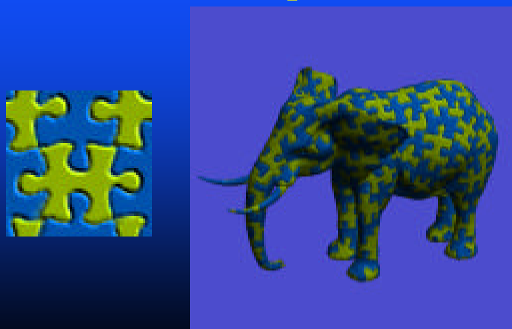

Two Bunnies


Brick Dolphin

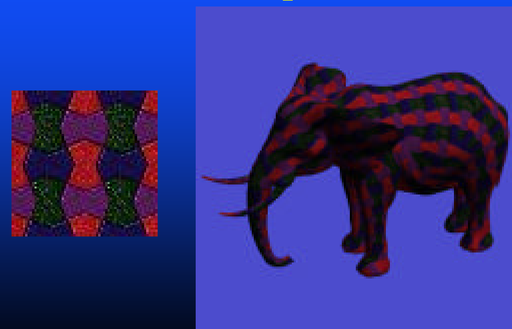

Text Zebra


Seahorse

## Puzzle Elephant



## Tiled Elephant



## Interlocking Tori

## Future Work

Invent new 2D synthesis methods

Adapt other 2D synthesis methods to meshes (such as image analogies and image quilting)

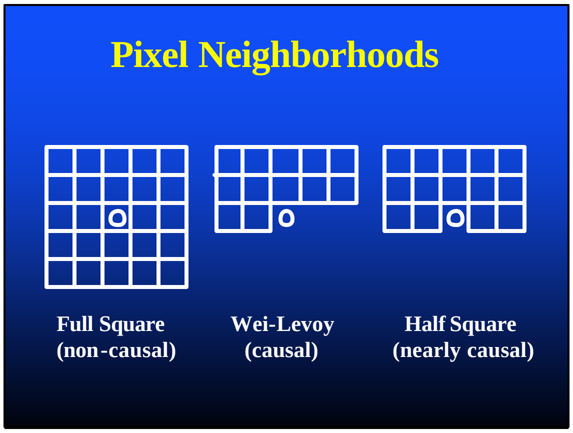
Perform texture analysis on meshes

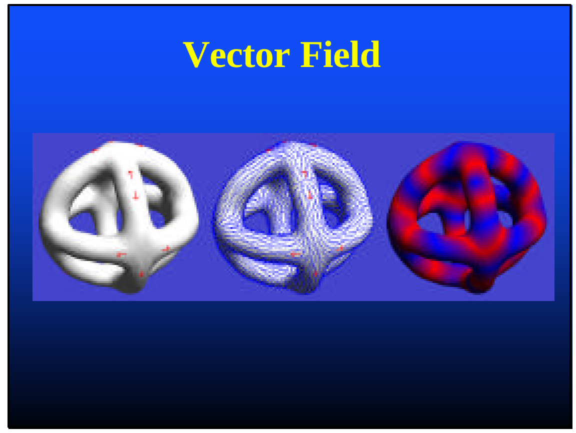
## Acknowledgements


Thanks to my students and colleagues at Georgia Tech

Research funded by NSF CAREER grant CCR-9703265.


## Questions?


Pixel Neighborhoods
Full Square
(non-causal)
Wei-Levoy
(causal)
Half Square
(nearly causal)


Vector Field