Texture Synthesis: Review

- What is stationarity?
- What is the Markov assumption?
- How does the Efros & Leung texture synthesis algorithm work?
- What is the most important user-selected parameter in E&L and how does it affect performance?
- What are some shortcomings of E&L?
- What are some ways of accelerating texture synthesis?
  - Gaussian pyramids
  - Tree-structured vector quantization (Wei & Levoy)
  - “Coherence search” heuristic (Ashikhmin)
  - Image quilting (Efros & Freeman)
Today

- Some extensions of texture synthesis
- Image analogies
- Super-resolution

Slide credits: A. Efros, L.-Y. Wei, W. Freeman, A. Hertzman et al.
Extensions: Texture Synthesis on Surfaces

Extensions: Progressively Variant Textures

Extensions: Solid Texture Synthesis

Extensions:
Inverse Texture Synthesis

- L.-Y. Wei, J. Han, K. Zhou, H. Bao, B. Guo, and H. Shum, “Inverse Texture

\[ \text{original} \rightarrow \text{inverse texture synthesis result} \]

\[ \text{stationary} \]
\[ \text{globally varying} \]
Extensions:
Video Textures

Extensions:
Stochastic Motion Textures

Last Time: Texture Transfer

- Take the texture from one object and “paint” it onto another object
parmesan + rice =

parmesan + rice =

parmesan + rice =
Source correspondence image

Source texture

Target correspondence image
Source correspondence image

Source texture

Target correspondence image

Target image
Image Analogies

A

A'

B

B'
Image Analogies

Unfiltered source

Filtered source

Unfiltered target

Filtered target
The Approach

Unfiltered source

Filtered source

Unfiltered target

Filtered target
The Approach

Unfiltered source

Filtered source

Unfiltered target

Filtered target
The Approach

**function** `CREATEIMAGEANALOGY(A, A', B):
Compute Gaussian pyramids for A, A', and B
Compute features for A, A', and B
Initialize the search structures (e.g., for ANN)
for each level \( \ell \), from coarsest to finest, do:
  for each pixel \( q \in B'_\ell \), in scan-line order, do:
    \[ p \leftarrow \text{BESTMATCH}(A, A', B, B', s, \ell, q) \]
    \[ B'_\ell(q) \leftarrow A'_\ell(p) \]
    \[ s_\ell(q) \leftarrow p \]
return \( B'_L \)

**function** `BESTMATCH(A, A', B, B', s, \ell, q):
\[ p_{\text{app}} \leftarrow \text{BESTAPPROXIMATEMATCH}(A, A', B, B', \ell, q) \]
\[ p_{\text{coh}} \leftarrow \text{BESTCOHERENCEMATCH}(A, A', B, B', s, \ell, q) \]
\[ d_{\text{app}} \leftarrow \| F_\ell(p_{\text{app}}) - F_\ell(q) \|^2 \]
\[ d_{\text{coh}} \leftarrow \| F_\ell(p_{\text{coh}}) - F_\ell(q) \|^2 \]
if \( d_{\text{coh}} \leq d_{\text{app}} (1 + 2^{\ell-L} \kappa) \) then
  return \( p_{\text{coh}} \)
else
  return \( p_{\text{app}} \)
Implementation Details

- Use approximate nearest neighbor search and Ashikhmin’s *coherence search* heuristic
- Use *feature vectors* instead of pixel values
  - Feature vector can consist of RGB values plus additional “channels” such as luminance, outputs of derivative filters
- *Luminance remapping* to align color histograms of source and target images
Blur Filter

Unfiltered source ($A$)  Filtered source ($A'$)

Unfiltered target ($B$)  Filtered target ($B'$)
Edge Filter

Unfiltered source ($A$)  Filtered source ($A'$)

Unfiltered target ($B$)  Filtered target ($B'$)
Colorization

- Unfiltered source (A)
- Filtered source (A')
- Unfiltered target (B)
- Filtered target (B')
Texture Synthesis

• Source images (A, B) are blank/constant
Texture Synthesis

Input

Wei-Levoy

Input

Wei-Levoy

Ashikhmin

Ours

Ashikhmin

Ours
Texture Transfer

- A and A’ is the same (or A is a blurred version of A’)
- Optional: Tunable weight to control the tradeoff between matching (A, B) and (A’, B’)

Unfiltered source  Filtered source

Unfiltered target  Filtered target
Artistic Filters

A

B

A’

B’
Artistic Filters
Artistic Filters

A

A'

B

B'
Artistic Filters
Artistic Filters
Artistic Filters
Artistic Filters
Artistic Filters
More Artistic Filters

A

A’

B

B’
• Project idea: inverse “texture by numbers”
Dealing with progressively variant textures
Super-resolution
Super-resolution (result!)

B

B'
Approaches to perceptual sharpening

• Traditional: Boost existing high frequencies

![Graph showing amplitude vs. spatial frequency with an upward arrow indicating increased amplitude for a higher spatial frequency]
Unsharp masking

Original

(Note that filter sums to 1)
Unsharp masking

Original

Sharpening filter
- Accentuates differences with local average

Source: D. Lowe
Unsharp masking

before

after

Source: D. Lowe
Approaches to perceptual sharpening

- Traditional: Boost existing high frequencies

- Example-based: Estimate high frequencies not present in image, although implicitly defined
Example-Based Super-Resolution

W. Freeman, T. Jones, and E. Pasztor

Pixel replication

Cubic spline

Training-based super-resolution

Slide credits: W. Freeman
Representation

Zoomed low-freq.  
Full freq. original
(to minimize the complexity of the relationships we have to learn, we remove the lowest frequencies from the input image, and normalize the local contrast level).
Gather ~100,000 patches

Training data samples (magnified)
Nearest neighbor estimate

Input low freqs.

True high freqs.

Estimated high freqs.

Training data samples (magnified)
Why nearest neighbor fails

Input patch

Closest image patches from database

Corresponding high-resolution patches from database

Solution: enforce spatial coherence constraints between hi-res patches
Example

Source image patches

Band-pass filtered and contrast normalized

True high-resolution pixels

High-resolution pixels chosen by super-resolution

Band-pass filtered and contrast normalized best-match patches from training data

Best-match patches from training data
Now we examine the effect of the prior assumptions made about images on the high resolution reconstruction. First, cubic spline interpolation.
Next, train the algorithm on a world of random noise images.
The algorithm learns that, in such a world, we add random noise when zoom to a higher resolution.
Next, train on a world of vertically oriented rectangles.
The algorithm hallucinates those vertical rectangles that it was trained on.
Original

Example-based

Training images

True
Original

Example-based

Training images

True
Now train on a generic collection of images.
The algorithm makes a reasonable guess at the high resolution image, based on its training images.
Generic training images
Shortcoming: Amplifying noise and compression artifacts

Original image

Cubic interpolation, upsampling by a factor of 4

Super-resolution
Project Idea: Hallucinating Faces

Project Idea: Hallucinating Faces