Texture Synthesis

Adam Finkelstein
for Tom Funkhouser
Princeton University
COS526, Fall 2014

Slides from Efros, Freeman, Lazebnik, Wei

Texture

- Texture has spatially repeating patterns
- Lacks full range of complexity of photos
- But good starting point for study of image-based techniques


radishes


rocks


yogurt

Texture Synthesis

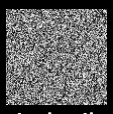
- Goal:
  Create new samples from given example texture
- Many applications:
  virtual environments, fill holes, textured surfaces


exemplar

Challenge

Need to model the whole spectrum:
from repeated to stochastic texture


repeated


stochastic

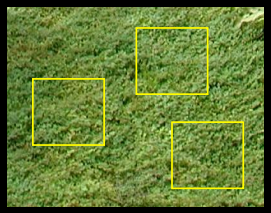

both?

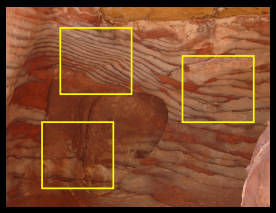
Some History

- Stochastic textures
  - [Heeger & Bergen,'95]
  - [DeBonet,'97]
  - [Portilla & Simoncelli,'98]
- Structured textures
  - [Liu, '04]
- Both
  - [Efros & Leung,'99]
  - [Efros & Freeman,'01]
  - [Kwatra, `05]

Statistical modeling of texture

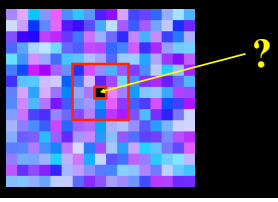
- Assume stochastic model of texture
  Markov random field (MRF)
- *Stationarity:*
  stochastic model is same, regardless of position


stationary texture


non-stationary texture

Statistical modeling of texture

- Assume stochastic model of texture
  Markov random field (MRF)
- *Stationarity:*
  stochastic model is same, regardless of position
- *Markov property*:
  p(pixel | rest of image) = p(pixel | neighborhood)



Motivation from Language

Shannon (1948) proposed a way to generate English-looking text using *N-grams:*

- Assume a Markov model
- Large corpus gives probability distribution for each letter, given N–1 previous letters
- Starting from a seed, repeatedly sample conditional probabilities to generate new letters
- Can also use whole words instead of letters

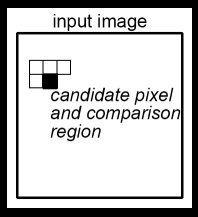
Efros

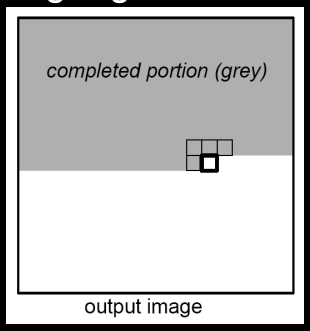
Mark V. Shaney (Bell Labs)

- Results (using alt.singles corpus):
  - *"As I've commented before, really relating to someone involves standing next to impossible."*
  - *"One morning I shot an elephant in my arms and kissed him."*
  - *"I spent an interesting evening recently with a grain of salt."*
- Notice how well local structure is preserved!
  - Now let's try this in 2D...

Efros

Efros & Leung Algorithm

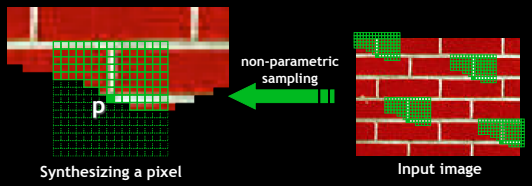




Initially proposed by Garber (1981),
but dismissed as too computationally expensive!

Efros

Efros & Leung Algorithm



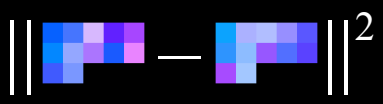
Assume Markov property, sample from P(**p**|N(**p**))

- Building explicit probability tables infeasible
- Instead, we search the input image for all sufficiently similar neighborhoods and pick one match at random

Efros

Finding matches

- Sum of squared differences (SSD)



Efros

Finding matches

- Sum of squared differences (SSD)
  - *Gaussian-weighted* to make sure closer neighbors are in better agreement

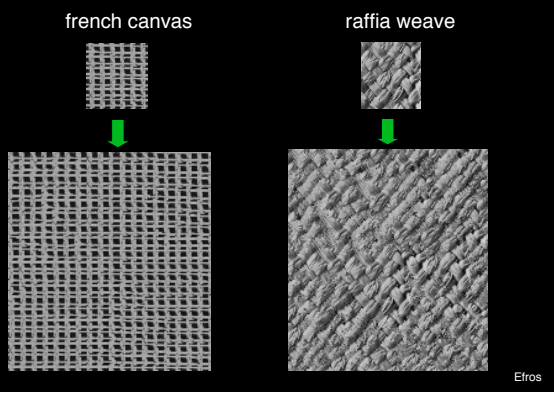
Efros

Implementation Details

- Initialization
  - Start with few rows of white noise
    - grow in scanline order
  - Start with a "seed" in middle
    - grow outward in layers
- Sampling
  - Random sampling from set of candidates vs. picking the best candidate

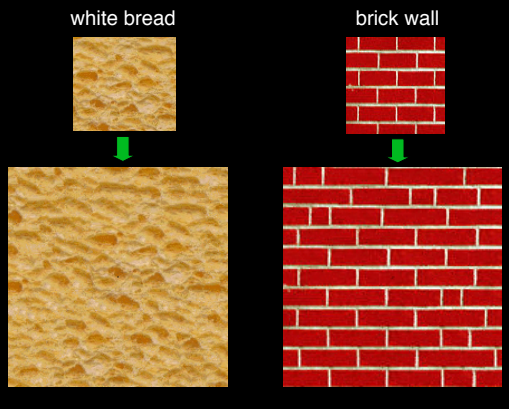
Efros

Synthesis Results



Efros

More Results

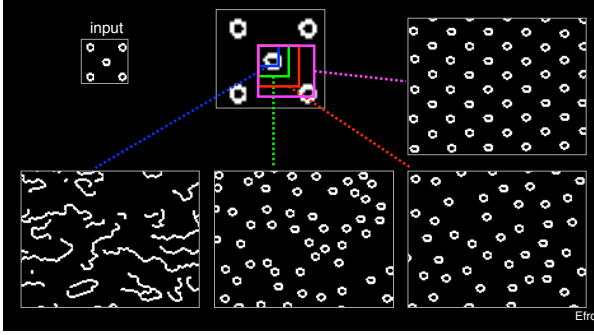


Efros

Homage to Shannon

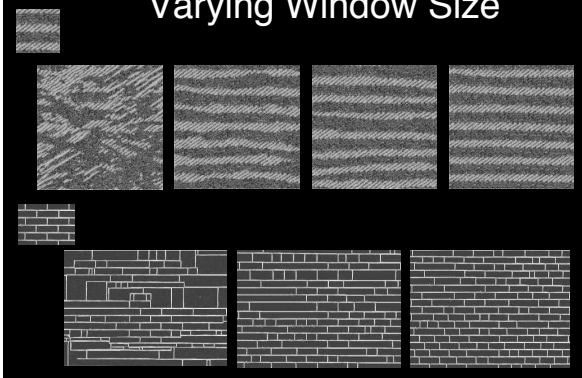
Efros

Varying Window Size



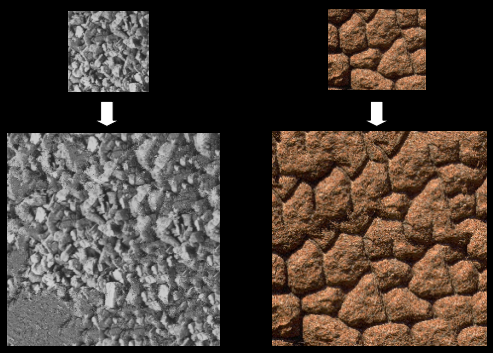
Efros

Varying Window Size



Efros

Failure Cases



Growing garbage **Verbatim copying**

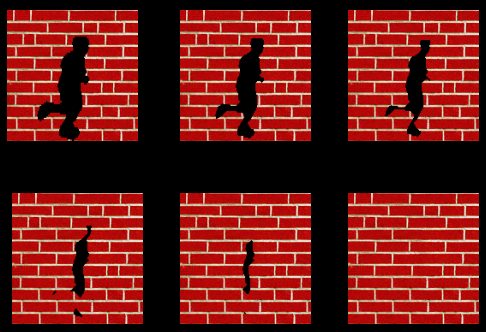
Efros

Example Applications

- Hole filling and extrapolation
  - Fill pixels in “onion skin” order
    - Within each “layer”, pixels with most neighbors are synthesized first
    - Normalize error by the number of known pixels
    - If no close match can be found, the pixel is not synthesized until the end

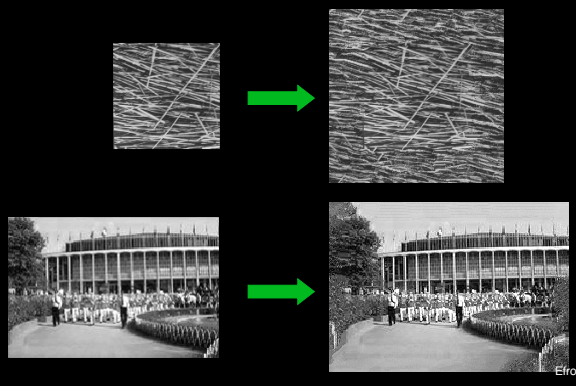
Efros

Hole Filling



Efros

Extrapolation



Efros

Summary

- The Efros & Leung algorithm
  - Very simple
  - Surprisingly good results
- Problems?

Accelerating texture synthesis

- Indexed similarity search
- Coherence
- Multiresolution
- Patches

Indexed Similarity Search

Perform fast approximate nearest neighbor search using spatial data structure

- *tree-structured vector quantization (TSVQ)*
- *kd-tree (optionally with PCA)*

Indexed Similarity Search

- Improves efficiency, but can degrade quality


original

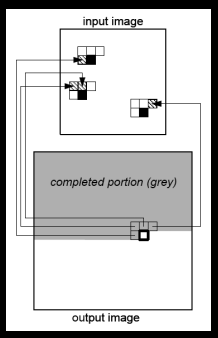

full search


TSVQ

Wei00

Coherence

Original position of synthesized pixels in the neighborhood implies "short list" of candidates for current pixel



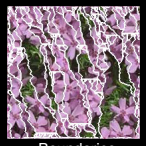
Ashikhmin01

Coherence


Original sample


Wei & Levoy

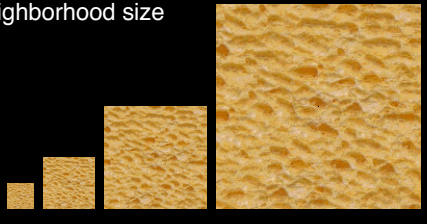

Ashikhmin


Boundaries

Ashikhmin01

Multiresolution

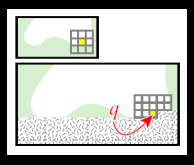
- For textures with large-scale structures, use a *Gaussian pyramid* to reduce required neighborhood size



Wei00

Multiresolution

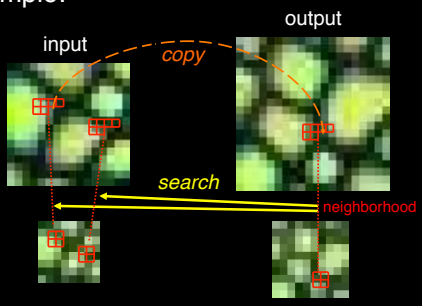
- For textures with large-scale structures, use a *Gaussian pyramid* to reduce required neighborhood size
  1. Synthesize at low-resolution
  2. Repeat for higher-res levels: "neighborhood" consists of generated pixels at this level and all neighboring pixels at lower level



Wei00

Multiresolution

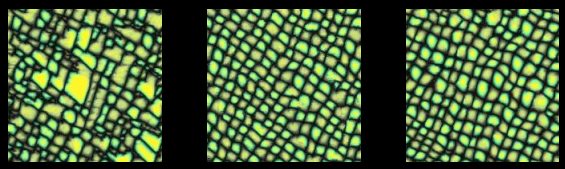
Example:



Wei00

Multiresolution

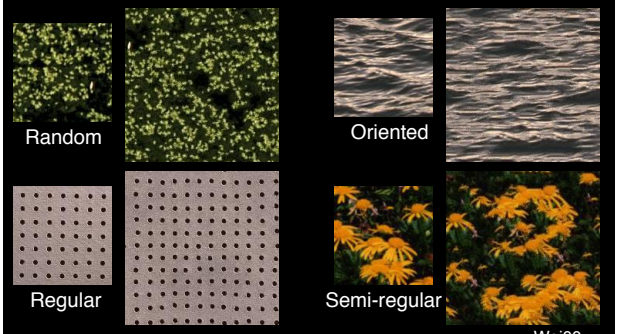
Results



1 level 5×5 | 1 level 11×11 | 3 levels 5×5

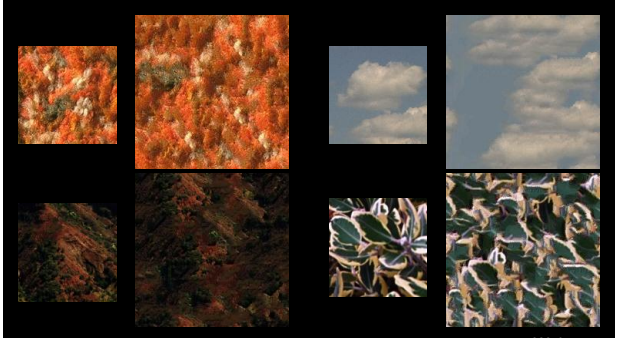
Wei00

Multiresolution



Wei00

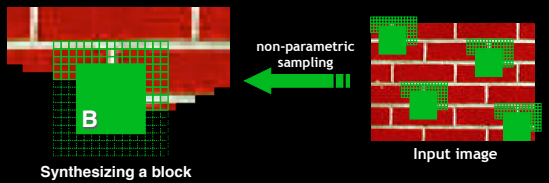
Multiresolution



Wei00

Patch-Based Synthesis

Copy patches of pixels rather than pixels



Observation: neighbor pixels are highly correlated

- Exactly the same as Efros & Leung but P(**B**|N(**B**))
- Much faster: synthesize all pixels in a block at once

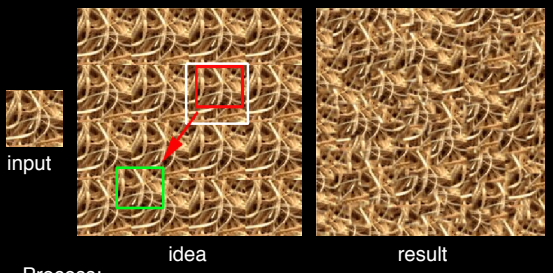
Efros01

Patch-Based Synthesis

- General approach:
  - Copy large blocks from input image
  - Then hide seams
- Rationale:
  - Texture blocks are by definition “correct”
  - So only problem is connecting them together

Efros01

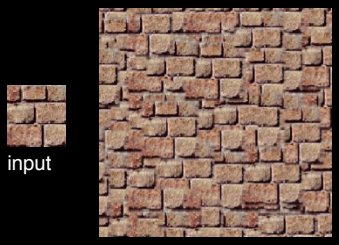
Chaos Mosaic



Process:
1) tile input image
2) place random blocks in random locations
3) smooth edges

Xu00

Chaos Mosaic

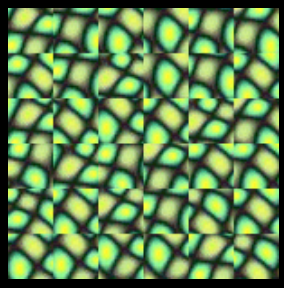


Does not work for structured textures, of course.

Xu00

Image Quilting [Efros & Freeman]

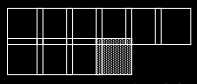
Regularly arranged patches



Efros01

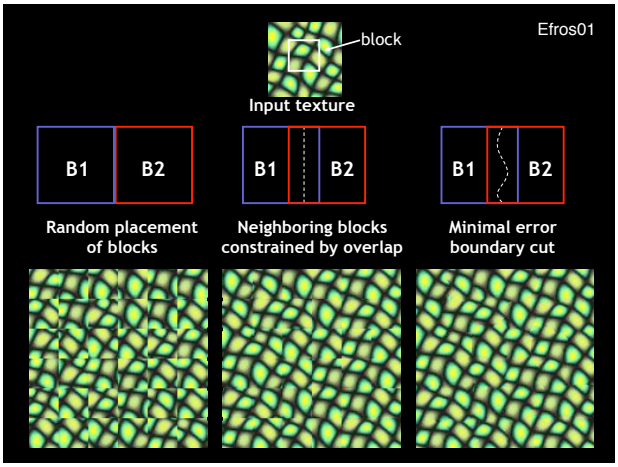
Algorithm

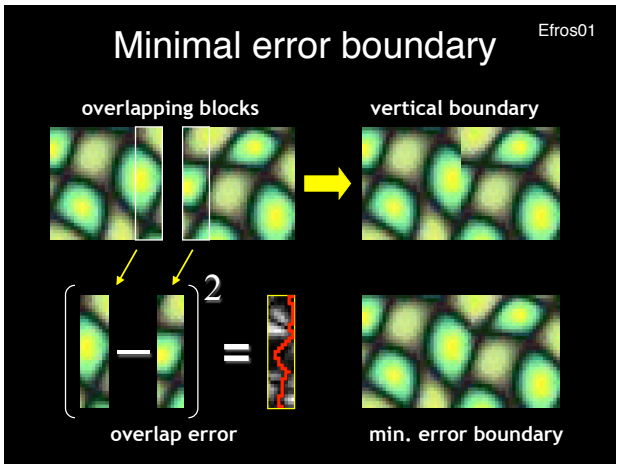
- Pick size of block and size of overlap
- Synthesize blocks in raster order

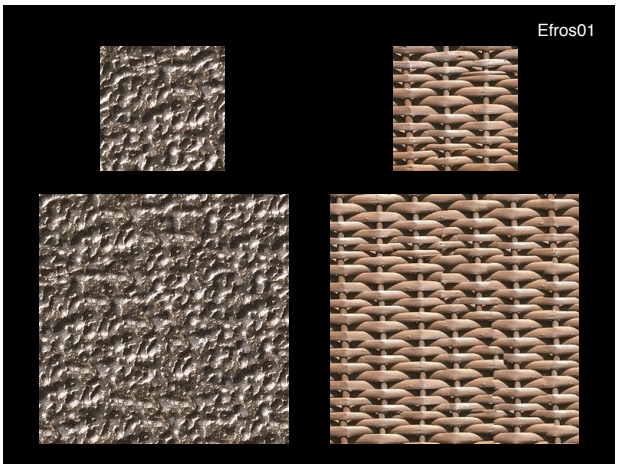


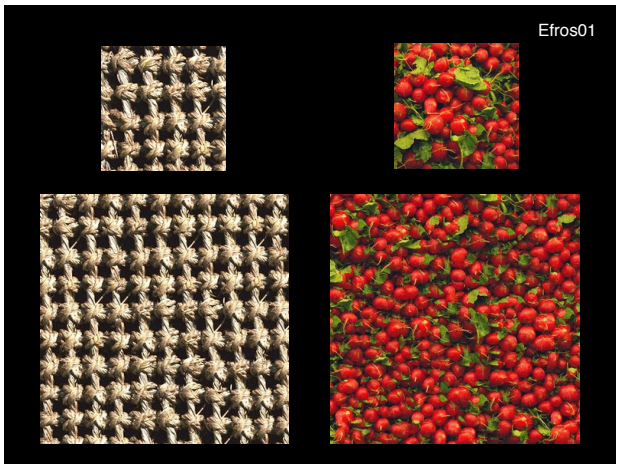
- Search input texture for block that satisfies overlap constraints (above and left)
- Paste new block into resulting texture
  - compute minimal-error boundary cut (dynamic programming)

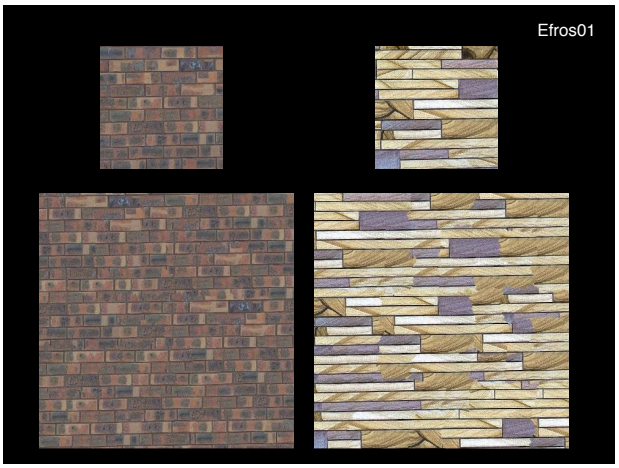
Efros01

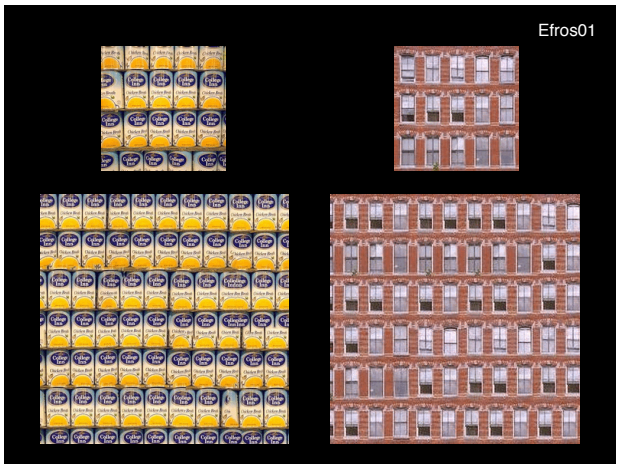


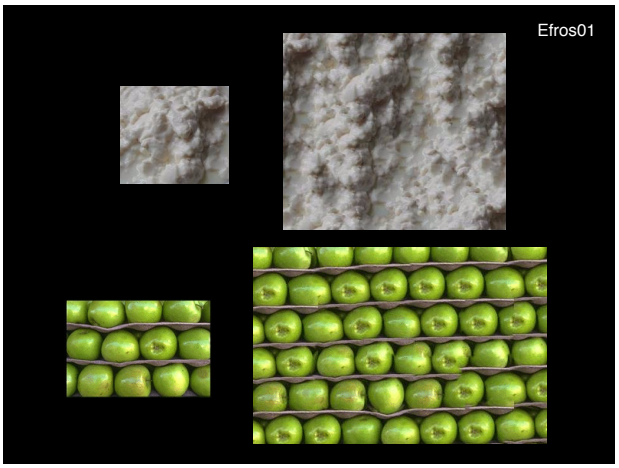

Minimal error boundary
Efros01
overlapping blocks
vertical boundary
2
overlap error
min. error boundary

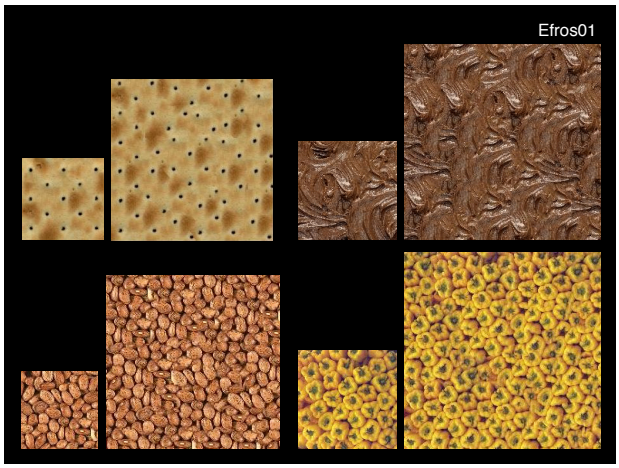

Efros01


Efros01


Efros01


Efros01


Efros01



Summary

- Texture synthesis
  - create new samples of a given texture
- Non-parametric methods
  - Copy samples from input based on neighborhood similarity
- Acceleration techniques
  - Multiresolution
  - Indexing
  - Coherence
  - Patches