Image Processing Techniques and Smart Image Manipulation

Maneesh Agrawala

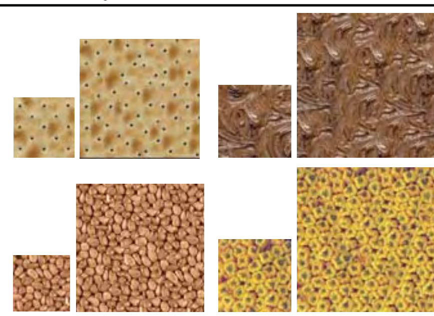
Topics

Texture Synthesis
High Dynamic Range Imaging
Bilateral Filter
Gradient-Domain Techniques
Matting
Graph-Cut Optimization
Least-Squares Optimization
Color
…

Topics

Texture Synthesis
High Dynamic Range Imaging
Bilateral Filter
Gradient-Domain Techniques
Matting
Graph-Cut Optimization
Least-Squares Optimization
Color
…

Texture Synthesis



Slides from:
Alexei Efros, CMU, Fall 2008
Ganesh Ramanarayanan Cornell

Weather Forecasting for Dummies™

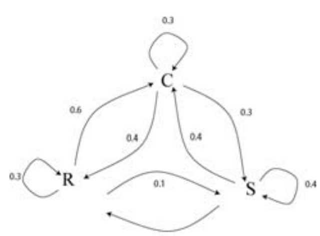
Let's predict weather:

- Given today's weather only, we want to know tomorrow's
- Suppose weather can only be {Sunny, Cloudy, Raining}

The "Weather Channel" algorithm:

- Over a long period of time, record:
  - How often S followed by R
  - How often S followed by S
  - Etc.
- Compute percentages for each state:
  - P(R|S), P(S|S), etc.
- Predict the state with highest probability!
- It's a Markov Chain

Markov Chain



$$\begin{pmatrix} 0.3 & 0.6 & 0.1 \\ 0.4 & 0.3 & 0.3 \\ 0.2 & 0.4 & 0.4 \end{pmatrix}$$

What if we know today and yestarday's weather?

Text Synthesis

[Shannon,'48] proposed a way to generate English-looking text using N-grams:

- Assume a generalized Markov model
- Use a large text to compute prob. distributions of each letter given N-1 previous letters
- Starting from a seed repeatedly sample this Markov chain to generate new letters
- Also works for whole words

WE NEED TO EAT CAKE

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"*

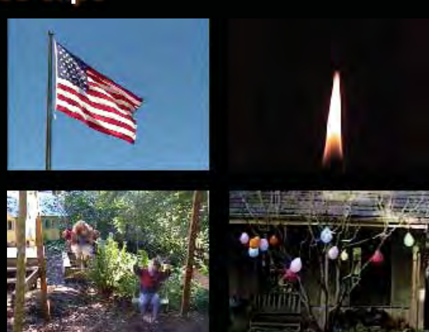
Video Textures

Arno Schödl
Richard Szeliski
David Salesin
Irfan Essa

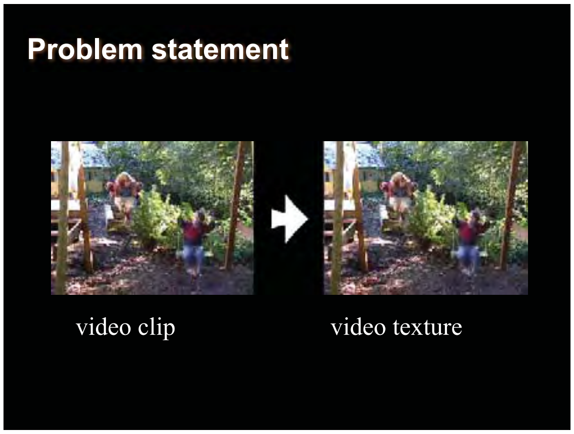
Microsoft Research, Georgia Tech

Still photos

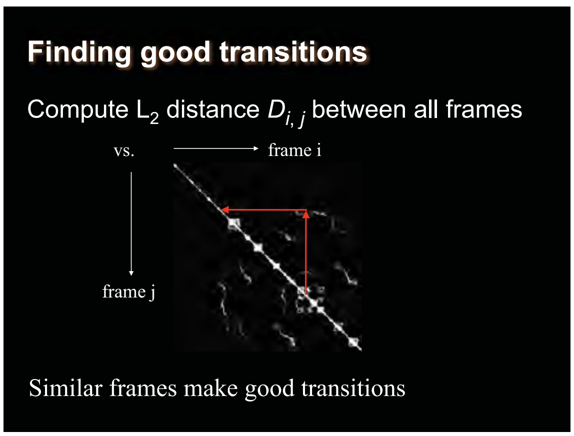
Video clips

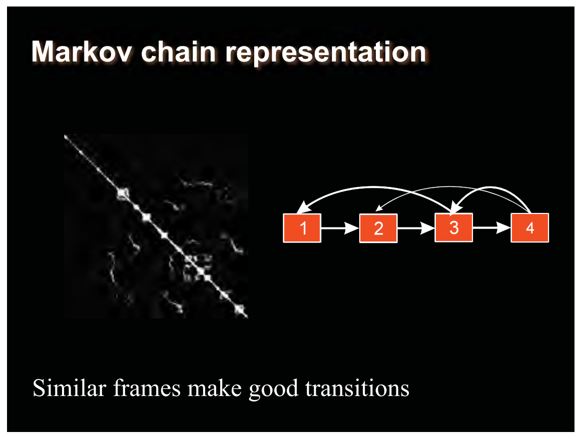


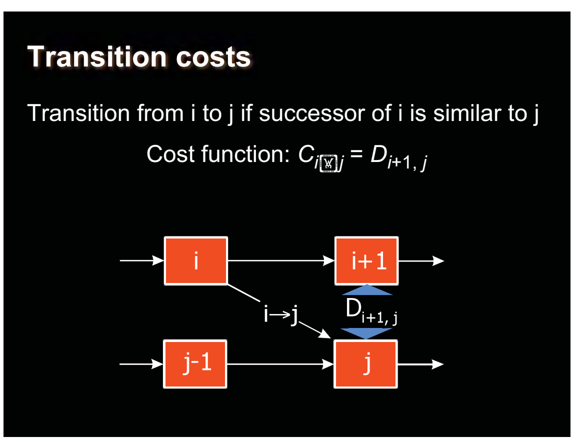
Video textures

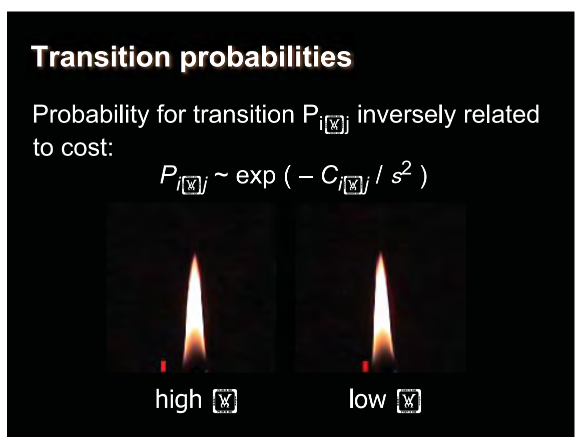

Problem statement
video clip
video texture


Our approach
How do we find good transitions?


Finding good transitions
Compute L_2 distance $D_{i,j}$ between all frames
vs.
frame i
frame j
Similar frames make good transitions

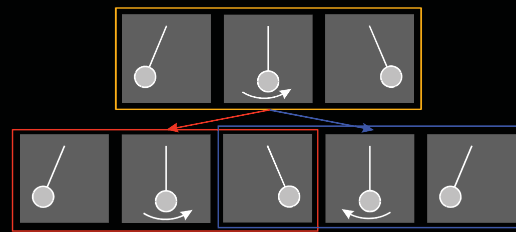

Markov chain representation
1
2
3
4
Similar frames make good transitions


Transition costs
Transition from i to j if successor of i is similar to j
Cost function: $C_{i \to j} = D_{i+1,j}$
i
i+1
$i \to j$
$D_{i+1,j}$
j-1
j


Transition probabilities
Probability for transition $P_{i \to j}$ inversely related to cost:
$P_{i \to j} \sim \exp(-C_{i \to j} / s^2)$
high
low

Preserving dynamics

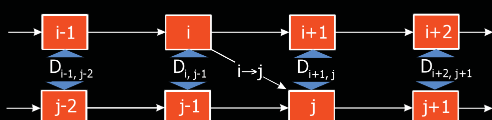
Preserving dynamics



Preserving dynamics

Cost for transition $i \rightarrow j$

$$C_{i \rightarrow j} = \sum_{k=-N}^{N-1} w_k D_{i+k+1,\, j+k}$$



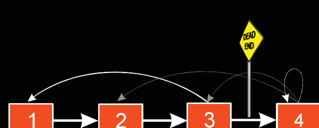
Preserving dynamics – effect

Cost for transition $i \rightarrow j$

$$C_{i \rightarrow j} = \sum_{k=-N}^{N-1} w_k D_{i+k+1,\, j+k}$$

Dead ends

No good transition at the end of sequence



Future cost

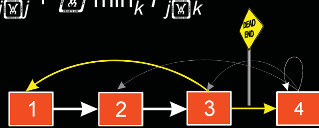
- Propagate future transition costs backward
- Iteratively compute new cost

$$F_{i \rightarrow j} = C_{i \rightarrow j} + \alpha \min_k F_{j \rightarrow k}$$

Future cost

- Propagate future transition costs backward
- Iteratively compute new cost

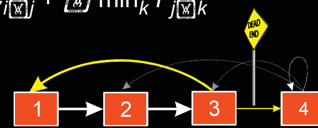
$$F_{i \to j} = C_{i \to j} + \alpha \min_k F_{j \to k}$$



Future cost

- Propagate future transition costs backward
- Iteratively compute new cost

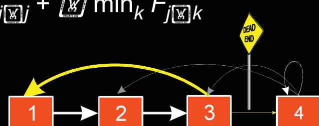
$$F_{i \to j} = C_{i \to j} + \alpha \min_k F_{j \to k}$$



Future cost

- Propagate future transition costs backward
- Iteratively compute new cost

$$F_{i \to j} = C_{i \to j} + \alpha \min_k F_{j \to k}$$



Future cost

- Propagate future transition costs backward
- Iteratively compute new cost

$$F_{i \to j} = C_{i \to j} + \alpha \min_k F_{j \to k}$$

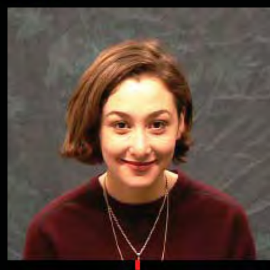
- Q-learning

Future cost – effect

Finding good loops

- Alternative to random transitions
- Precompute set of loops up front

Video portrait



Useful for web pages

Region-based analysis

- Divide video up into regions

- Generate a video texture for each region

Automatic region analysis

User-controlled video textures

slow variable fast

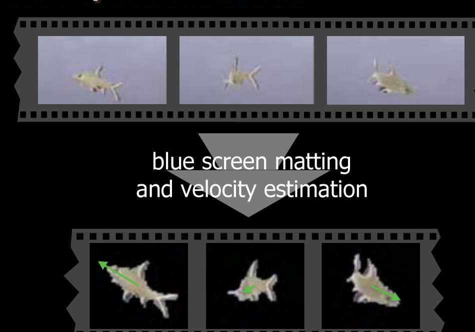
User selects target frame range

Video-based animation

- Like sprites computer games
- Extract sprites from real video
- Interactively control desired motion

©1985 Nintendo of America Inc.

Video sprite extraction



Video sprite control

- Augmented transition cost:

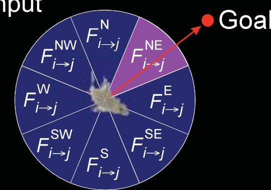
$$C^{\text{Animation}}_{i\to j} = \underbrace{\alpha\, C_{i\to j}}_{\text{Similarity term}} + \underbrace{\beta\ \text{angle}}_{\text{Control term}}$$

vector to mouse pointer

velocity vector

Video sprite control

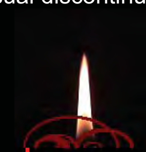
- Need future cost computation
- Precompute future costs for a few angles.
- Switch between precomputed angles according to user input
- [GIT-GVU-00-11]



Interactive fish

Summary

- Video clips video textures
  - define Markov process
  - preserve dynamics
  - avoid dead-ends
  - disguise visual discontinuities

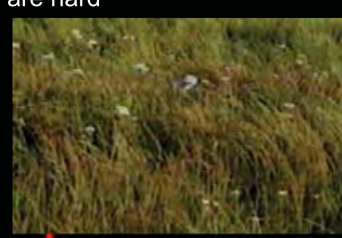


Discussion

- Some things are relatively easy

Discussion

- Some are hard



“Amateur” by Lasse Gjertsen

http://www.youtube.com/watch?v=JzqumbhfxRo

Michel Gondry train video

http://youtube.com/watch?v=qUEs1BwVXGA

Texture

- Texture depicts spatially repeating patterns
- Many natural phenomena are textures

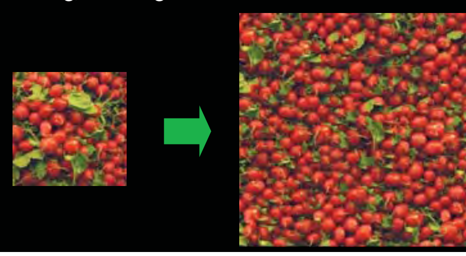

radishes


rocks


yogurt

Texture Synthesis

- Goal of Texture Synthesis: create new samples of a given texture
- Many applications: virtual environments, hole-filling, texturing surfaces

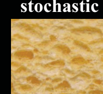


The Challenge

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


repeated

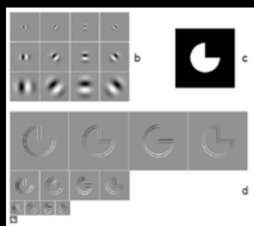

stochastic


Both?

Heeger Bergen 1995

- Seminal paper that introduced texture synthesis to the graphics community
- Algorithm:
  - Initialize *J* to noise
  - Create multiresolution pyramids for *I* and *J*
  - Match the histograms of *J*’s pyramid levels with *I*’s pyramid levels
  - Loop until convergence
  - Can be generalized to 3D

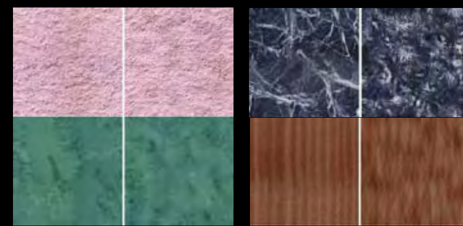
Heeger Bergen 1995 - Algorithm



- Image pyramids
  - Gaussian
  - Laplacian
- Steerable pyramids [SimoncelliFreeman95]
  - b): multiple scales of oriented filters
  - c): a sample image
  - d): results of filters in b) applied to c)

Heeger Bergen 1995 - Results

Successes Failures



I *J* *I* *J*

Heeger Bergen 1995 - Results

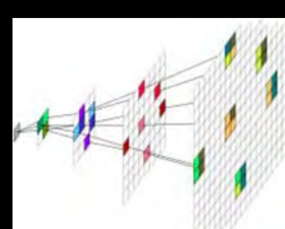
Heeger Bergen 1995 - Verdict

- Texture model:
  - Histograms of responses to various filters
- Avoiding copying:
  - Inherent in algorithm
- No user intervention required
- Captures stochastic textures well
- Does not capture structure
  - Lack of inter-scale constraints

De Bonet 1997

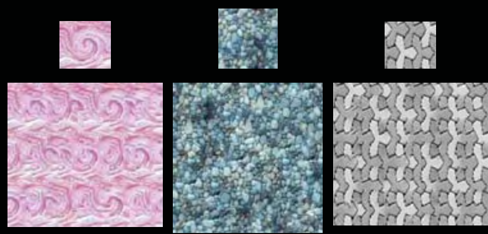
- Propagate constraints downwards by matching statistics all the way up the pyramid
- ***Feature vector:*** multiscale collection of filter responses for a given pixel
- Algorithm:
  - Initialize *J* to empty image
  - Create multiresolution pyramids for *I* and *J*
  - For each pixel in level of *J*, randomly choose pixel from corresponding level of *I* that has <u>similar</u> feature vector

De Bonet 1997 - Algorithm



- 6 feature vectors shown
- Notice how they share parent information

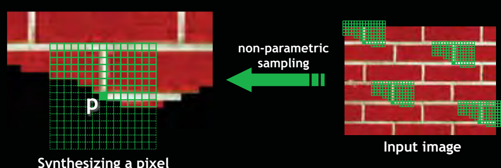
De Bonet 1997 - Results



De Bonet 1997 - Verdict

- Texture model:
  - Feature vector containing multiscale responses to various filters
- Avoiding copying:
  - Random choice of pixels with 'close' feature vectors, but copying still frequent on small scale
- Individual per-filter thresholds are cumbersome
- Feature vectors used in later synthesis work

Efros & Leung 1999 - Algorithm

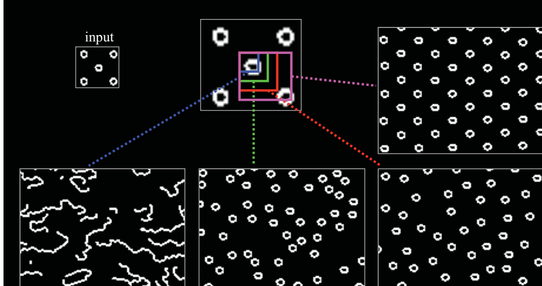


- Assuming Markov property, compute P(**p**|N(**p**))
  - Building explicit probability tables infeasible
  - Instead, we *search the input image* for all similar neighborhoods — that's our pdf for **p**
  - To sample from this pdf, just pick one match at random

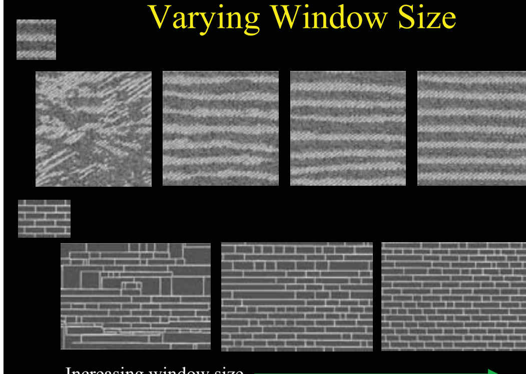
Some Details

- Growing is in "onion skin" order
  - Within each "layer", pixels with most neighbors are synthesized first
  - If no close match can be found, the pixel is not synthesized until the end
- Using *Gaussian-weighted* SSD is very important
  - to make sure the new pixel agrees with its closest neighbors
  - Approximates reduction to a smaller neighborhood window if data is too sparse

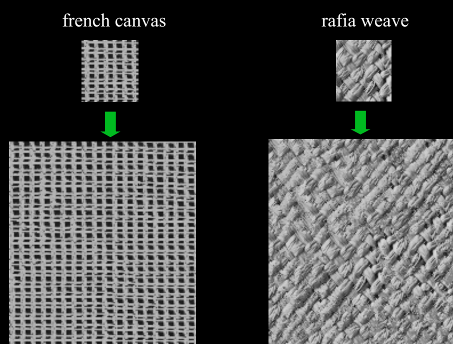
Neighborhood Window



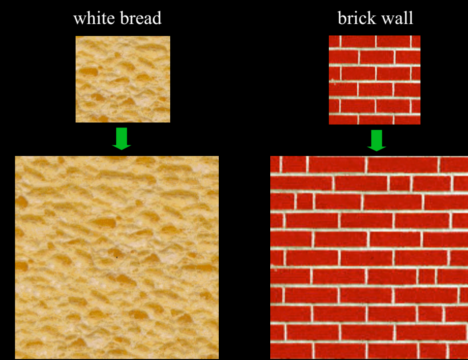
Varying Window Size



Synthesis Results

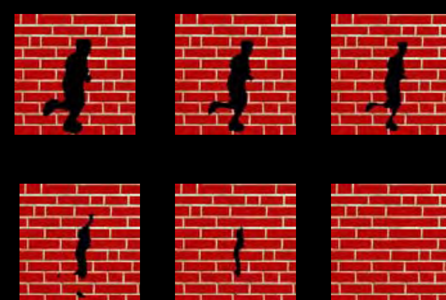


More Results

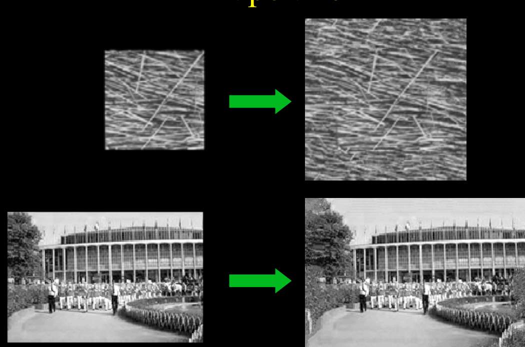


Homage to Shannon

Hole Filling



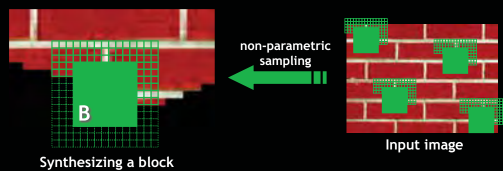
Extrapolation



Efros Leung 1999 – Verdict

- Texture model:
  - MRF
- Avoiding copying:
  - MRF
- Neighborhood size = largest feature size
- Markov model is surprisingly good
  - "I spent an interesting evening recently with a grain of salt."
- Search is very slow with large neighborhoods

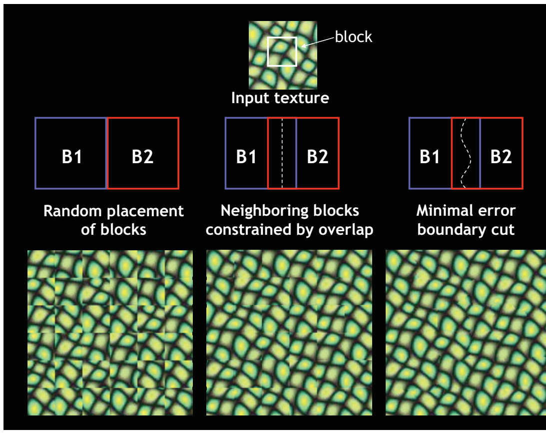
Image Quilting [Efros & Freeman]



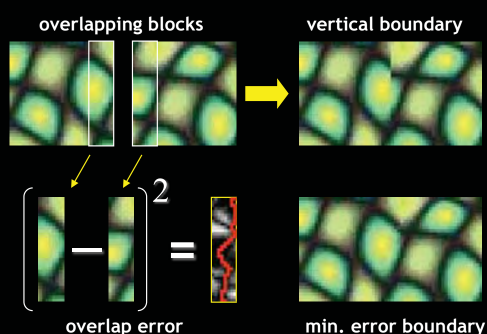
- Observation: neighbor pixels are highly correlated

Idea: unit of synthesis = block

- Exactly the same but now we want P(B|N(B))
- Much faster: synthesize all pixels in a block at once
- Not the same as multi-scale!

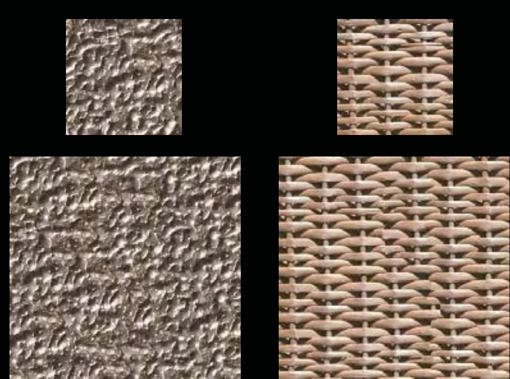


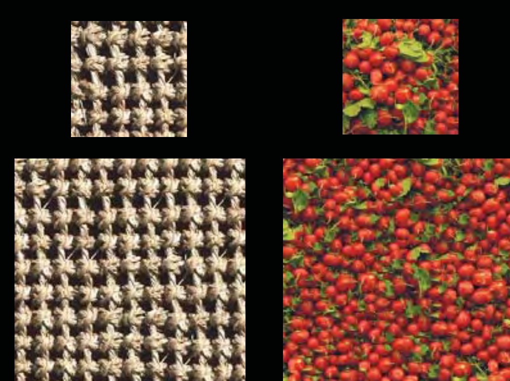
Minimal error boundary

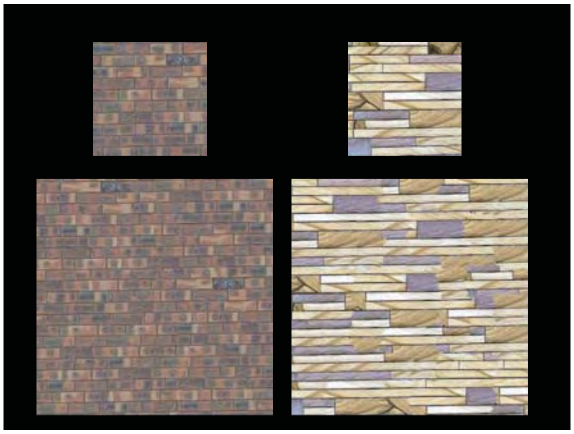


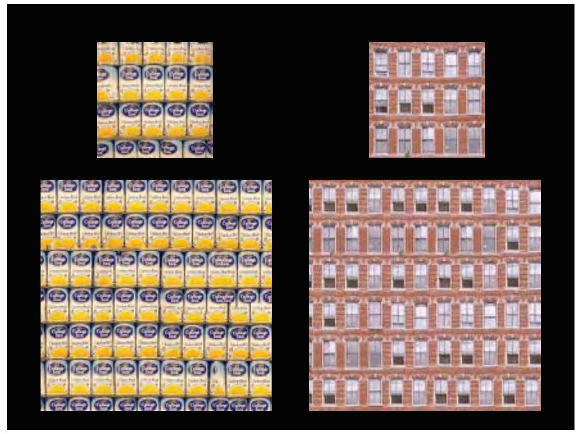
Our Philosophy

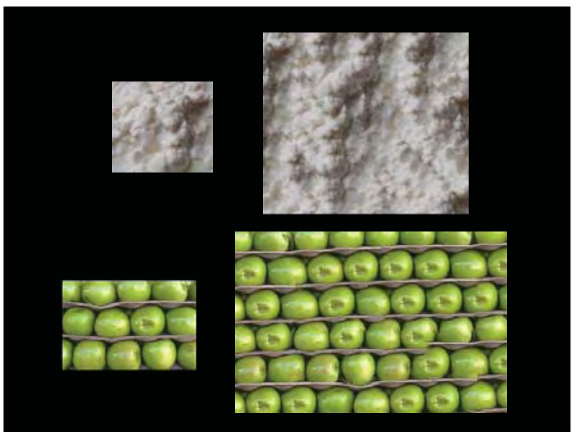
- The “Corrupt Professor’s Algorithm”:
  - Plagiarize as much of the source image as you can
  - Then try to cover up the evidence
- Rationale:
  - Texture blocks are by definition correct samples of texture so problem only connecting them together

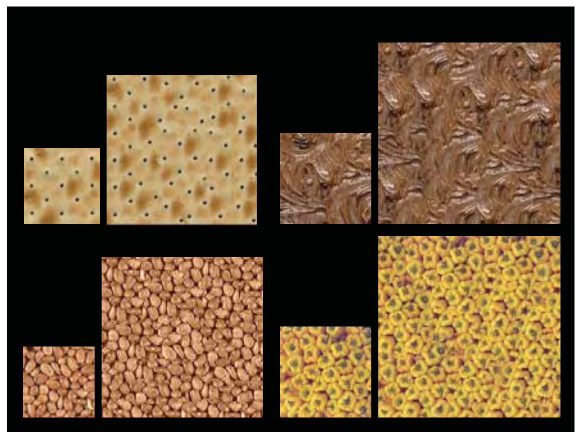


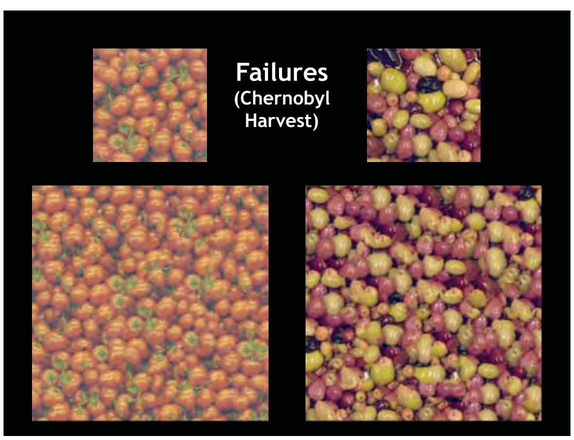


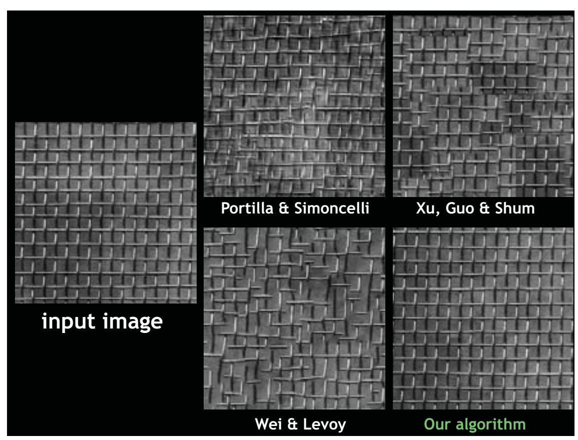


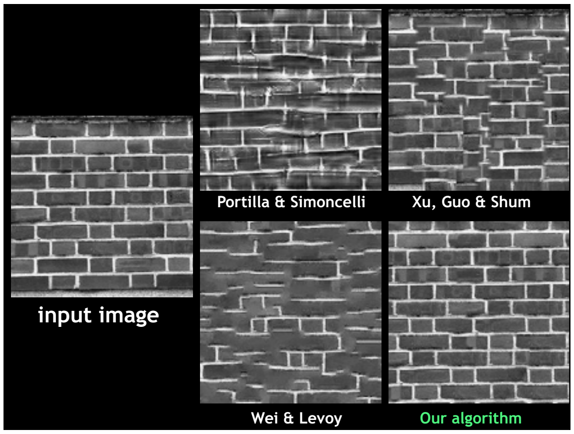


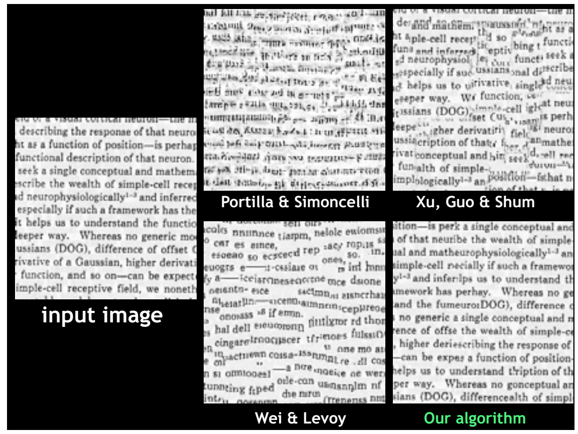





Failures
(Chernobyl
Harvest)


Portilla & Simoncelli
Xu, Guo & Shum
input image
Wei & Levoy
Our algorithm





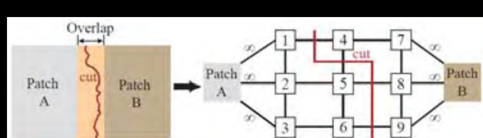
Efros Freeman 2001 - Verdict

- Texture model:
  - MRF
- Avoiding copying:
  - Randomized patch selection, but still noticeable
- Patch size is a hard parameter to understand
- Results are surprisingly good given algorithm
- Multiscale goes on a brief hiatus

Kwatra et. al. 2003

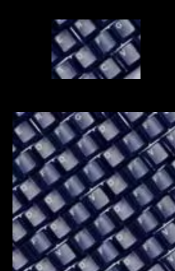
- Generalizes seam computation in overlap regions as a graph cut problem
  - Based on [Boykov et. al. 99] (with Ramin Zabih)
- Algorithm:
  - Initialize J to empty
  - Copy pieces of I to J using a variety of methods
  - Formulate graph in overlap region based on error (differences) and compute minimum cut
  - Copy sink-side pixels to J
  - Variety of strategies to further hide seams

Kwatra et. al. 2003 - Algorithm



(assume cut region is 3x3 for simplicity)

Kwatra et. al. 2003 - Results



Kwatra et. al. 2003 - Verdict

- Texture model:
  - MRF
- Avoiding copying:
  - Even with a multitude of patch selection methods, still noticeable when it happens repeatedly
- Paper presents a bag of synthesis tricks without much intuition for when to use what
- Graph cut formalization is useful and powerful

Fill Order

- In what order should we fill the pixels?

Fill Order

- In what order should we fill the pixels?
  - choose pixels that have more neighbors filled
  - choose pixels that are continuations of lines/curves/edges

Criminisi, Perez, and Toyama. "Object Removal by Exemplar-based Inpainting," Proc. CVPR, 2003.

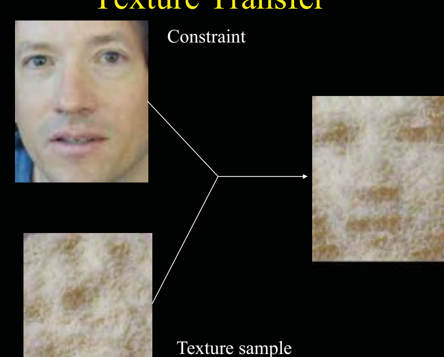
Exemplar-based Inpainting demo

http://research.microsoft.com/vision/cambridge/i3l/patchworks.htm

Application: Texture Transfer

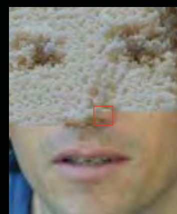
- Try to explain one object with bits and pieces of another object:

Texture Transfer



Texture Transfer

- Take the texture from one image and “paint” it onto another object



Same as texture synthesis, except an additional constraint:
1. Consistency of texture
2. Similarity to the image being “explained”

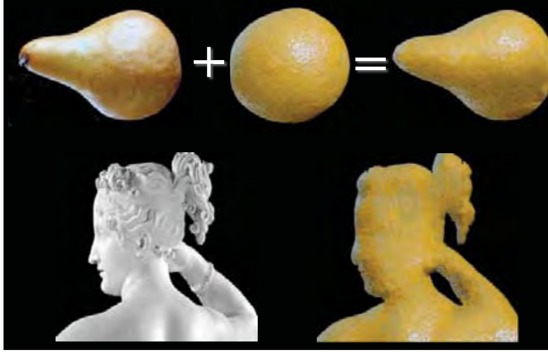


Image Analogies

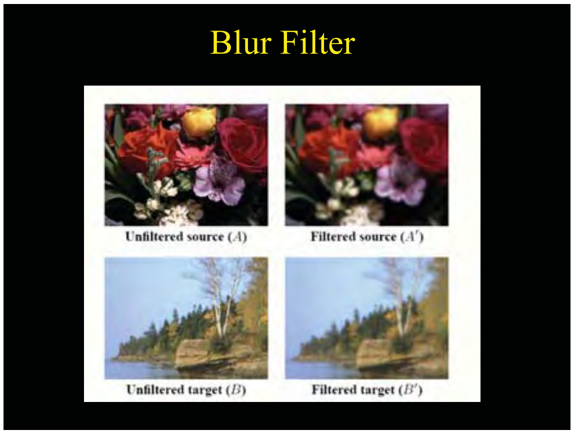
Aaron Hertzmann[1,2]
Chuck Jacobs[2]
Nuria Oliver[2]
Brian Curless[3]
David Salesin[2,3]

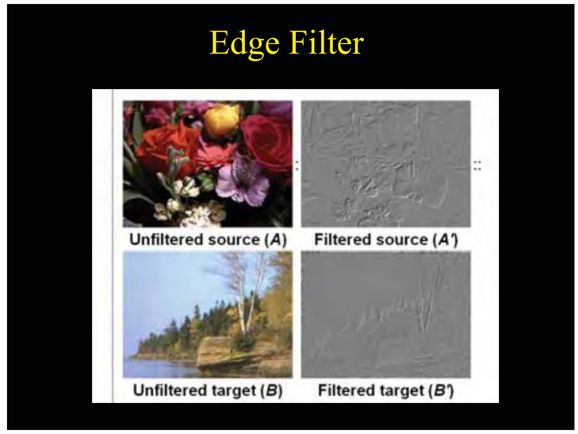
[1]New York University
[2]Microsoft Research
[3]University of Washington

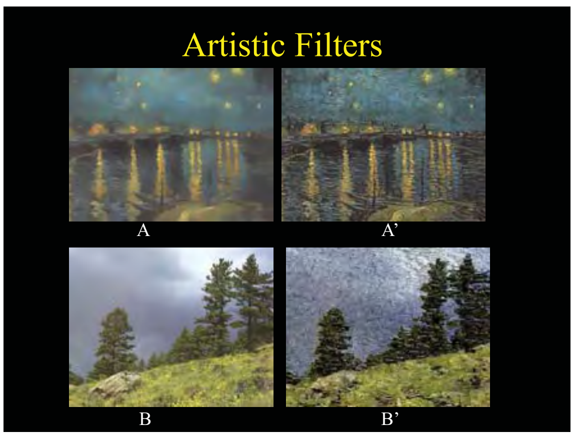
Image Analogies

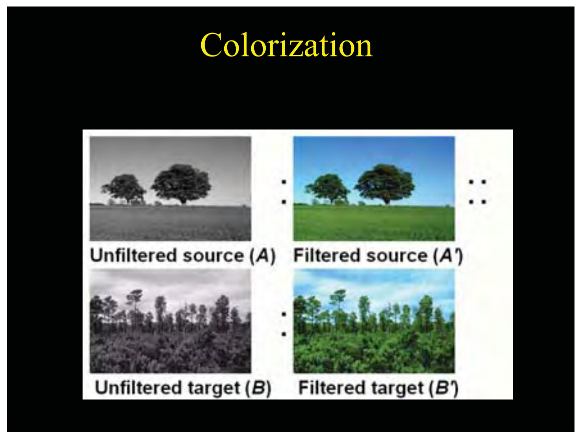
A A’

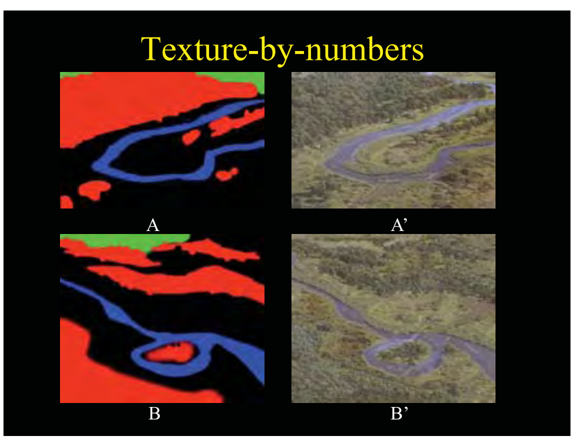
B B’


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′)


Artistic Filters
A
A’
B
B’


Colorization
Unfiltered source (A)
Filtered source (A′)
Unfiltered target (B)
Filtered target (B′)


Texture-by-numbers
A
A’
B
B’


Super-resolution
A
A’

Super-resolution (result!)

B

B’