Texture Synthesis

15-463: Rendering and Image Processing
Alexei Efros

Weather Forecasting for Dummies™

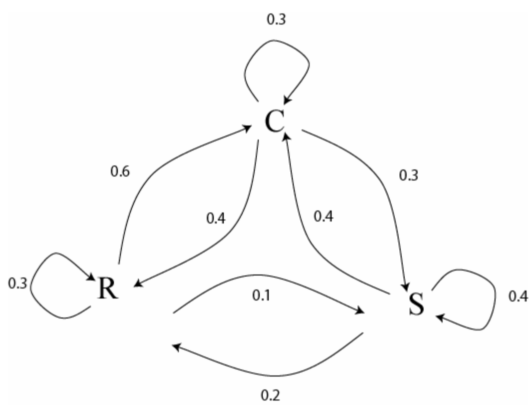
Let's predict weather:

- 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 yesterday'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

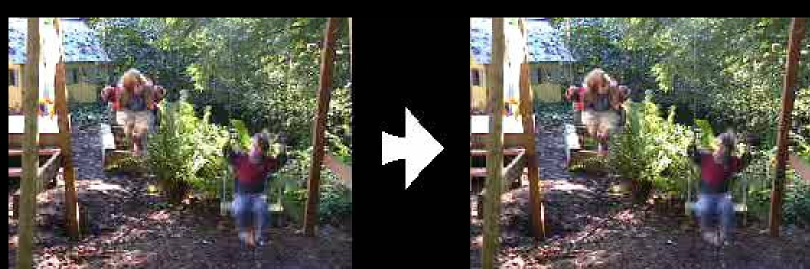
Microsoft Research, Georgia Tech

Still photos

Video clips

Video textures

Problem statement



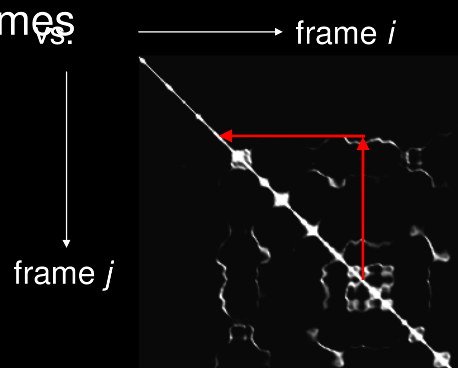
video clip video texture

Our approach

- How do we find good transitions?

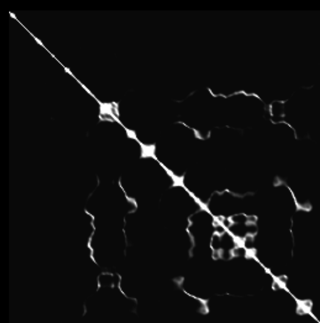
Finding good transitions

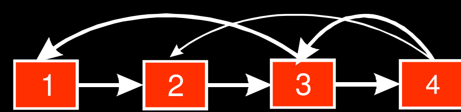
- Compute L_2 distance $D_{i,j}$ between all frames



Similar frames make good transitions

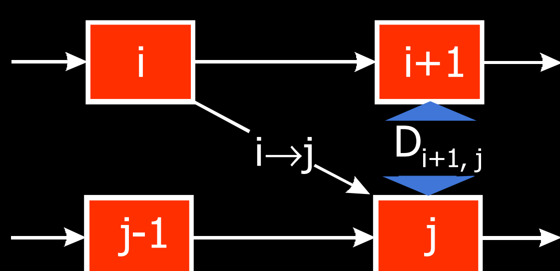
Markov chain representation





Similar frames make good transitions

Transition costs

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

Transition probabilities

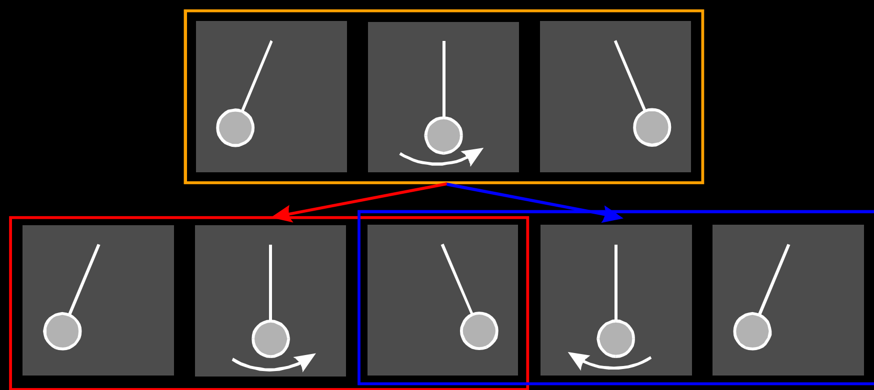
- Probability for transition $P_{i \to j}$ inversely related to cost:
  - $P_{i \to j} \sim \exp(-C_{i \to j} / \sigma^2)$

high σ

low σ

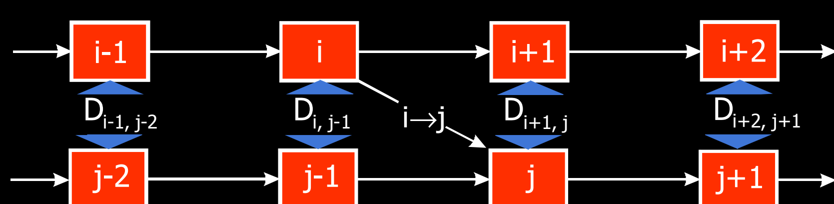
Preserving dynamics

Preserving dynamics



Preserving dynamics

- Cost for transition $i \to j$
  - $C_{i \to j} = \sum_{k=-N}^{N-1} w_k D_{i+k+1,\, j+k}$

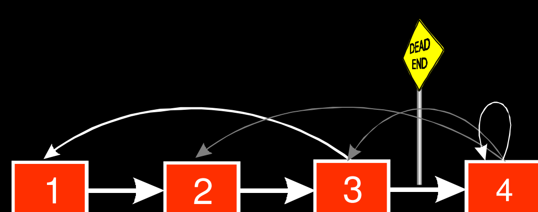


Preserving dynamics – effect

- Cost for transition $i \to j$
  - $C_{i \to j} = \sum_{k=-N}^{N-1} w_k D_{i+k+1,\, j+k}$

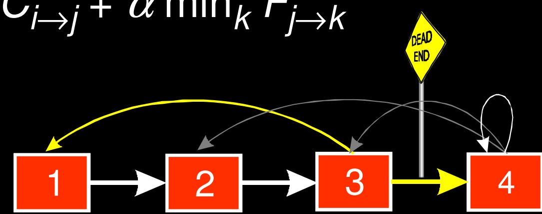
Dead ends

- No good transition at the end of sequence



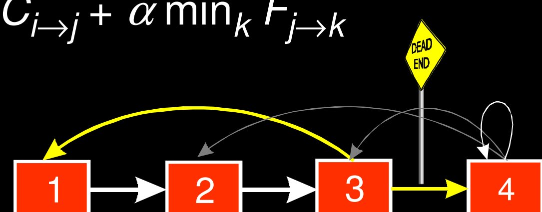
Future cost

- Propagate future transition costs backward
- Iteratively compute new cost
    - $F_{i\to j} = C_{i\to j} + \alpha \min_k F_{j\to k}$



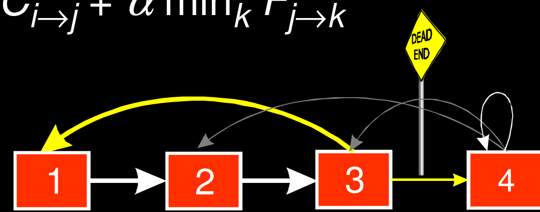
Future cost

- Propagate future transition costs backward
- Iteratively compute new cost
    - $F_{i\to j} = C_{i\to j} + \alpha \min_k F_{j\to k}$



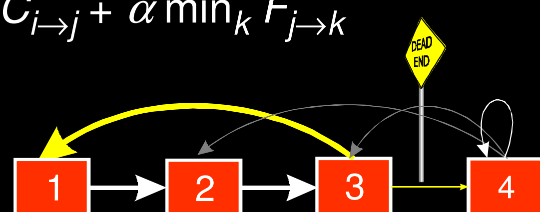
Future cost

- Propagate future transition costs backward
- Iteratively compute new cost
    - $F_{i \to j} = C_{i \to j} + \alpha \min_k F_{j \to k}$



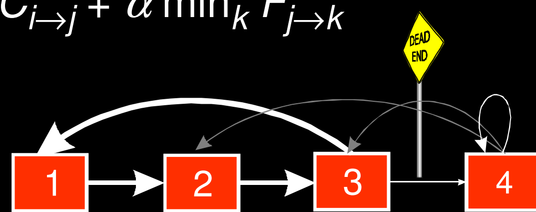
Future cost

- Propagate future transition costs backward
- Iteratively compute new cost
    - $F_{i \to j} = C_{i \to j} + \alpha \min_k F_{j \to k}$



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

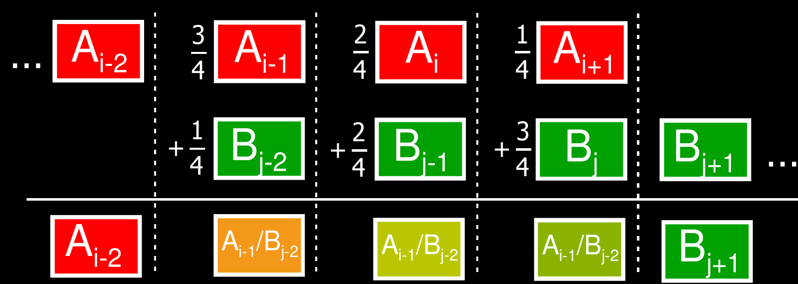
- Alternative to random transitions
- Precompute set of loops up front

Visual discontinuities

- Problem: Visible "Jumps"

Crossfading

- Solution: Crossfade from one sequence to the other.



Morphing

- Interpolation task:

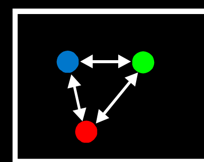
$$\frac{2}{5}\,A + \frac{2}{5}\,B + \frac{1}{5}\,C$$

Morphing

- Interpolation task:

  $\frac{2}{5}$ A $+ \frac{2}{5}$ B $+ \frac{1}{5}$ C

- Compute correspondence between pixels of all frames

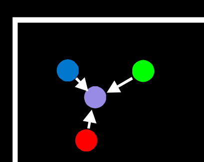


Morphing

- Interpolation task:

  $\frac{2}{5}$ A $+ \frac{2}{5}$ B $+ \frac{1}{5}$ C

- Compute correspondence between pixels of all frames
- Interpolate pixel position and color in morphed frame
- based on [Shum 2000]



Results – crossfading/morphing

Results – crossfading/morphing

Jump Cut Crossfade Morph

Crossfading

Frequent jump & crossfading

Video portrait

- Useful for web pages

Region-based analysis

- Divide video up into regions

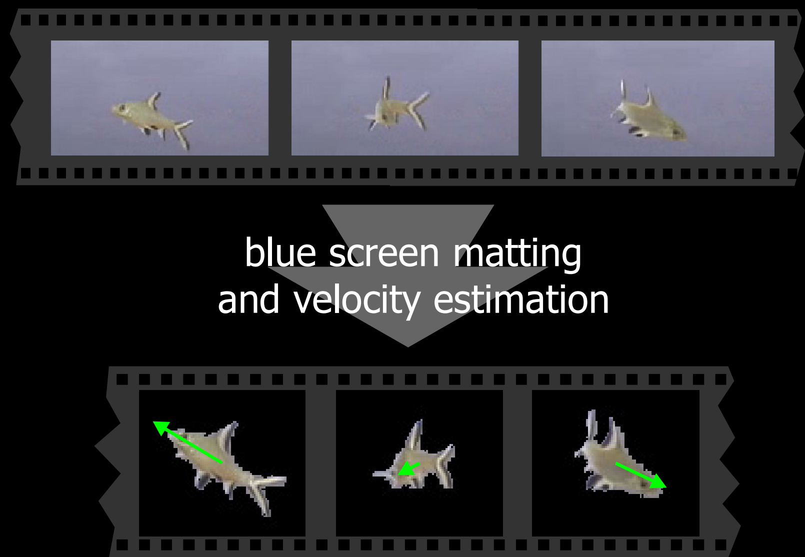
- Generate a video texture for each region

Automatic region analysis

Video-based animation

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

Video sprite extraction



Video sprite control

- Augmented transition cost:

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

Interactive fish

Lord of the Flies

Summary

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

Texture

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

radishes

rocks

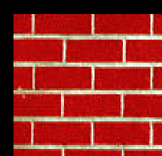
yogurt

Texture Synthesis

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

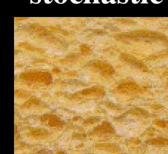
The Challenge

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



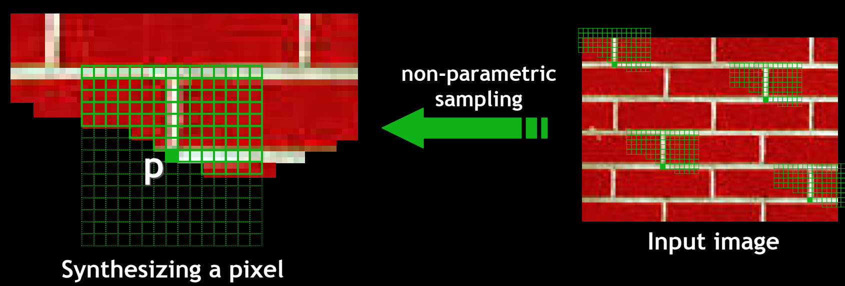
repeated

stochastic



Both?

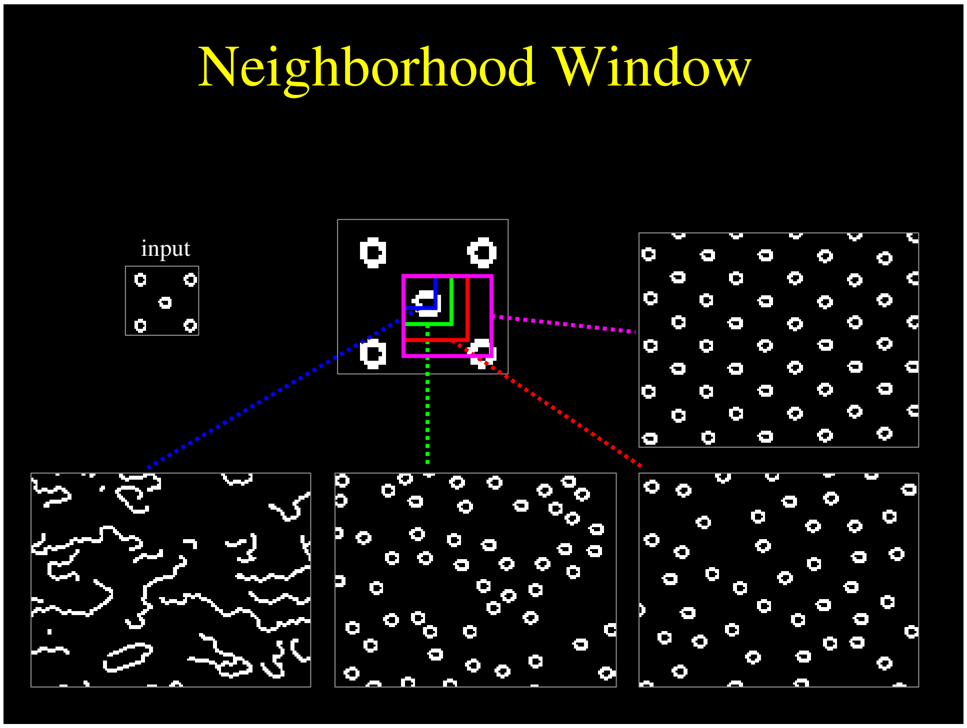
Efros & Leung Algorithm

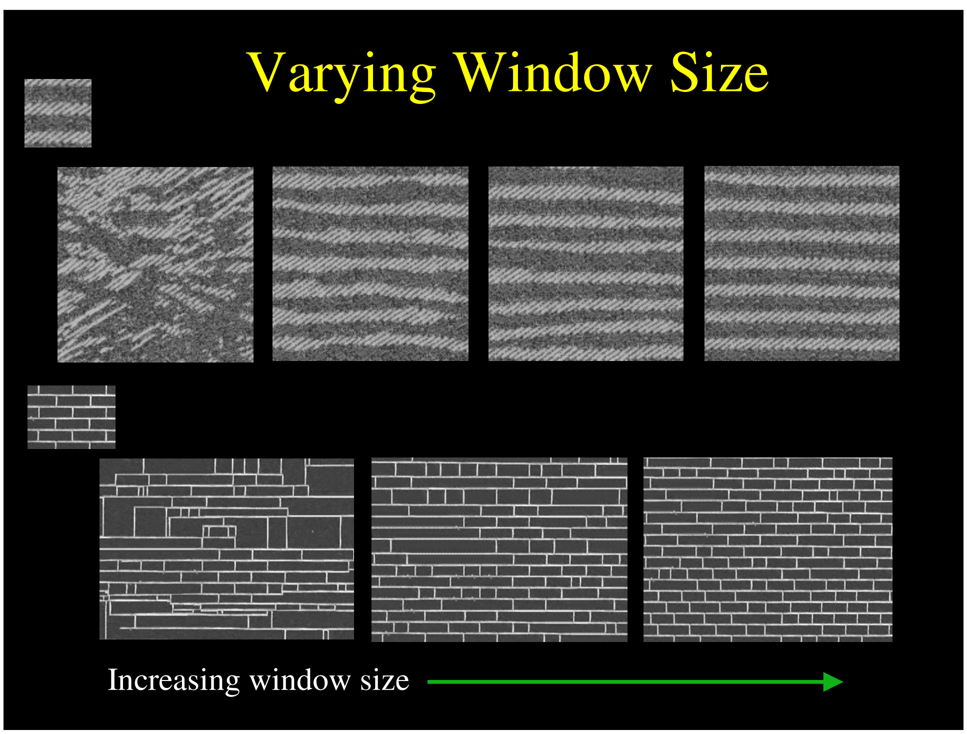


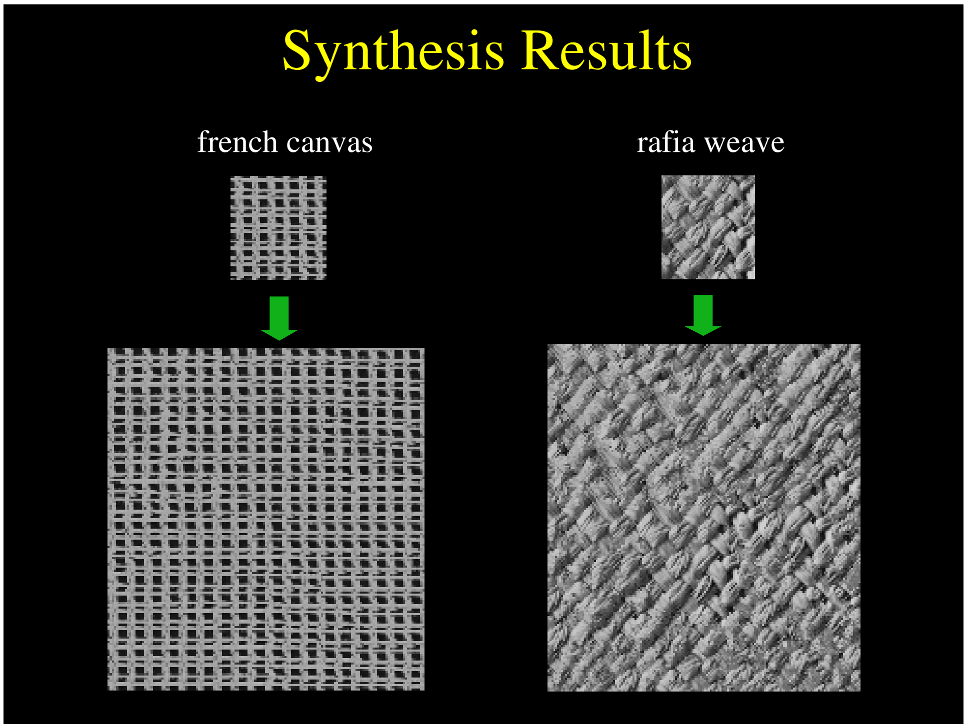
- Assuming Markov property, compute P(**p**|N(**p**))
  - Building explicit probability tables infeasible
  - Instead, we *search the input image* for all similar neighborhoods — that's our pdf for **p**
  - To sample from this pdf, just pick one match at random

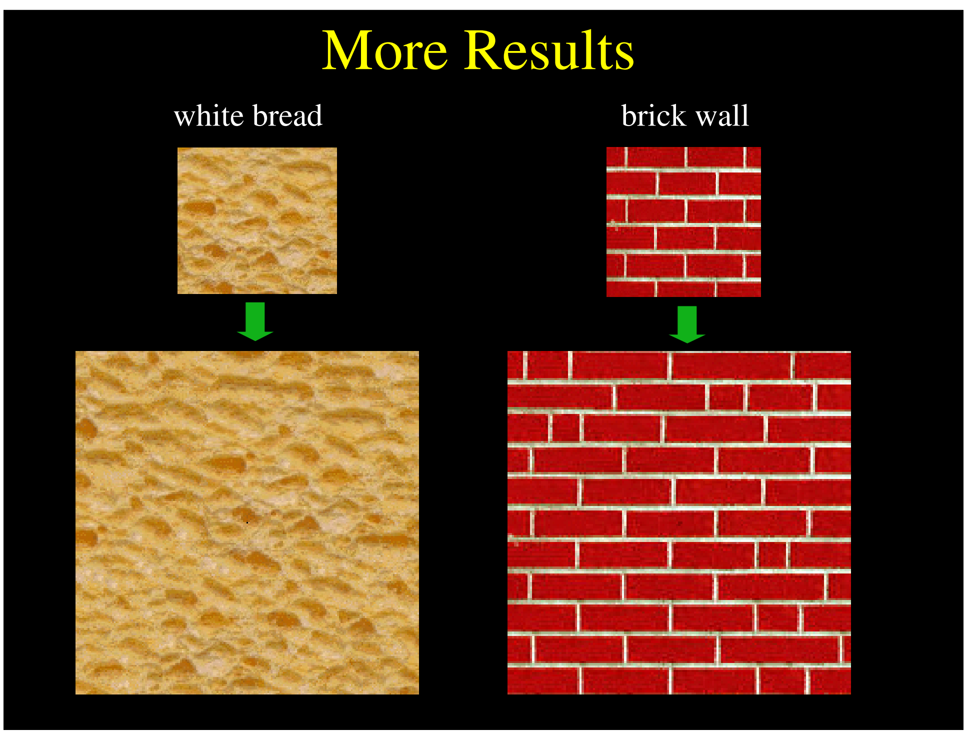
Some Details

- 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
input

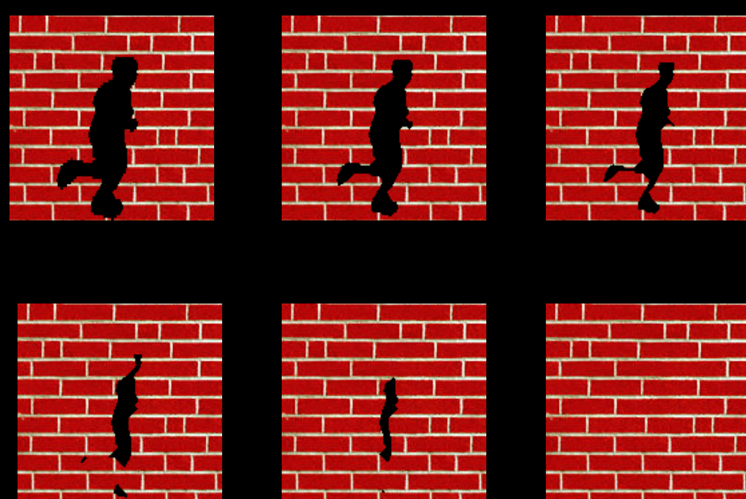

Varying Window Size
Increasing window size


Synthesis Results
french canvas
rafia weave

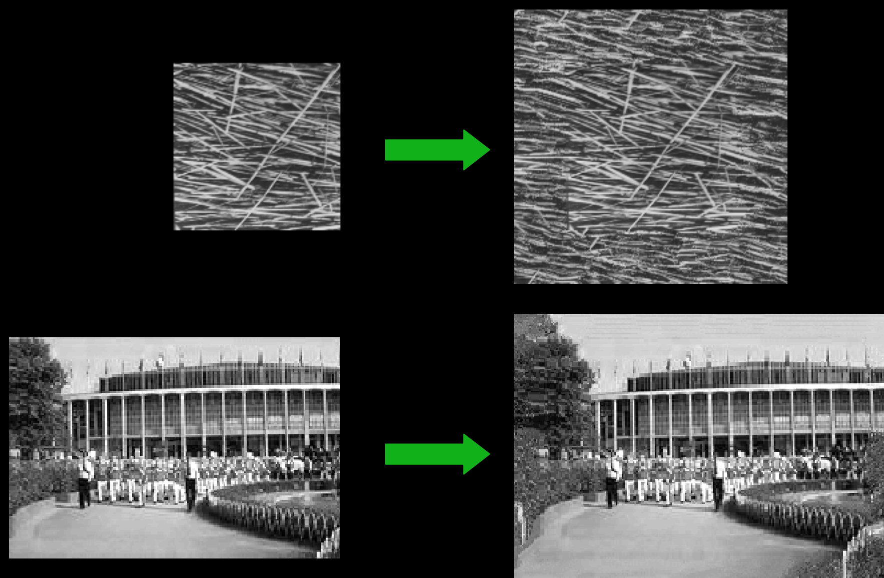

More Results
white bread
brick wall

Homage to Shannon

Hole Filling



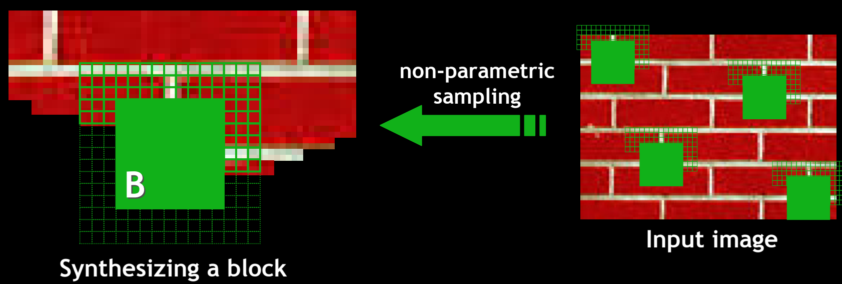
Extrapolation



Summary

- The Efros & Leung algorithm
  - Very simple
  - Surprisingly good results
  - Synthesis is easier than analysis!
  - …but very slow

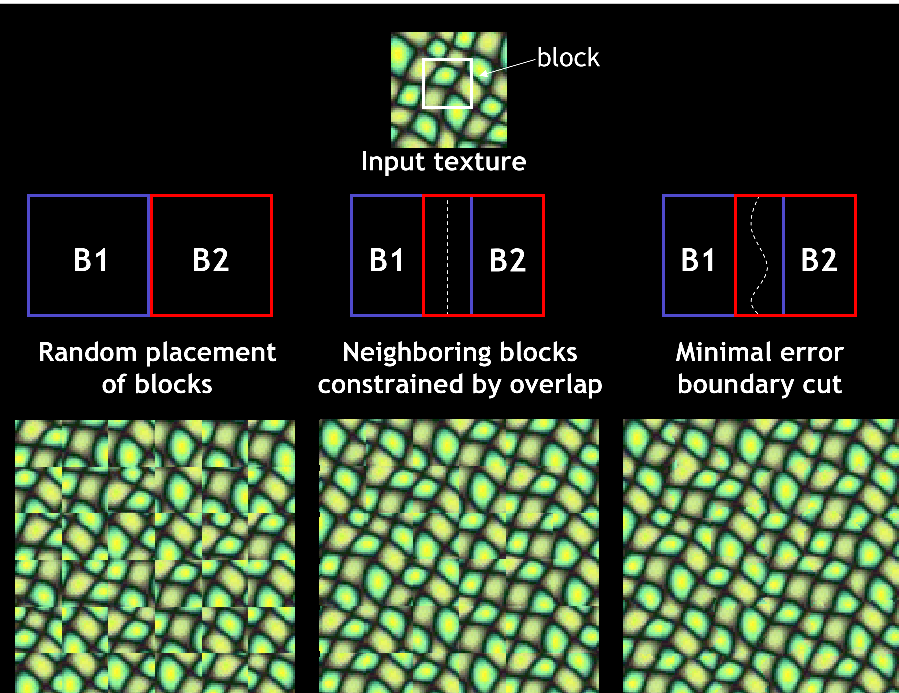
Image Quilting [Efros & Freeman]



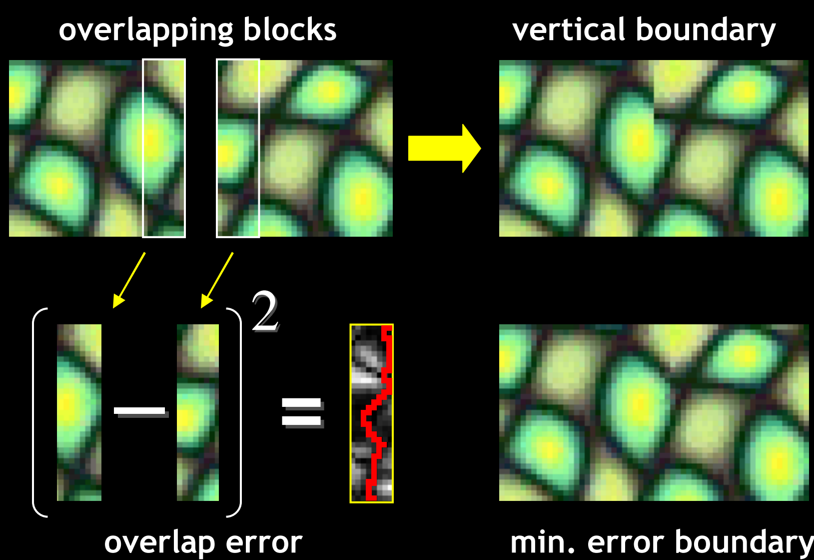
- Observation: neighbor pixels are highly correlated

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!

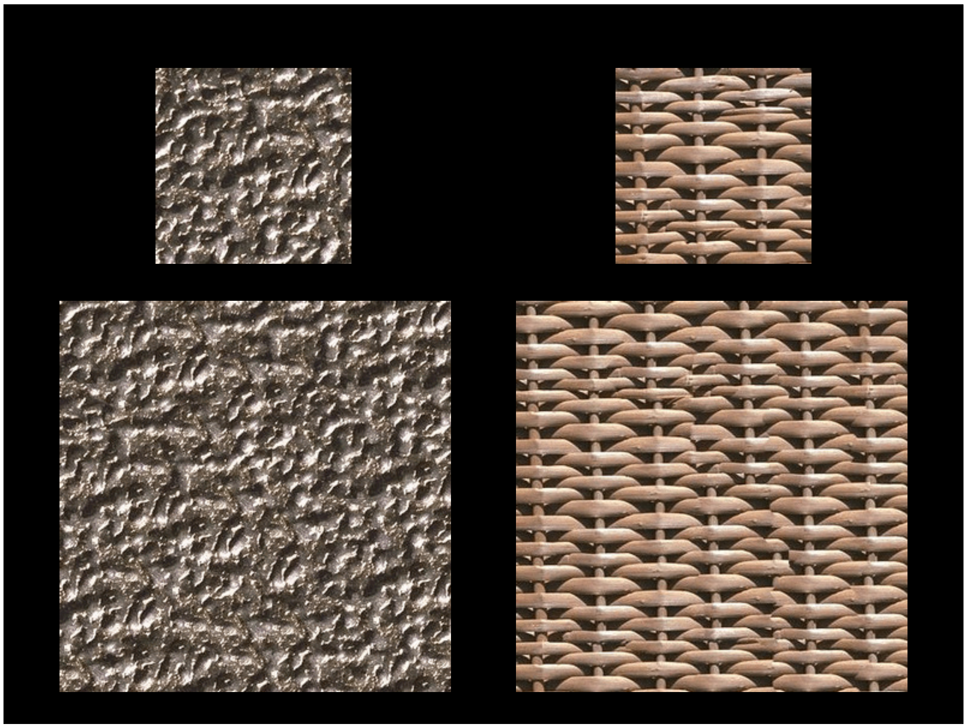


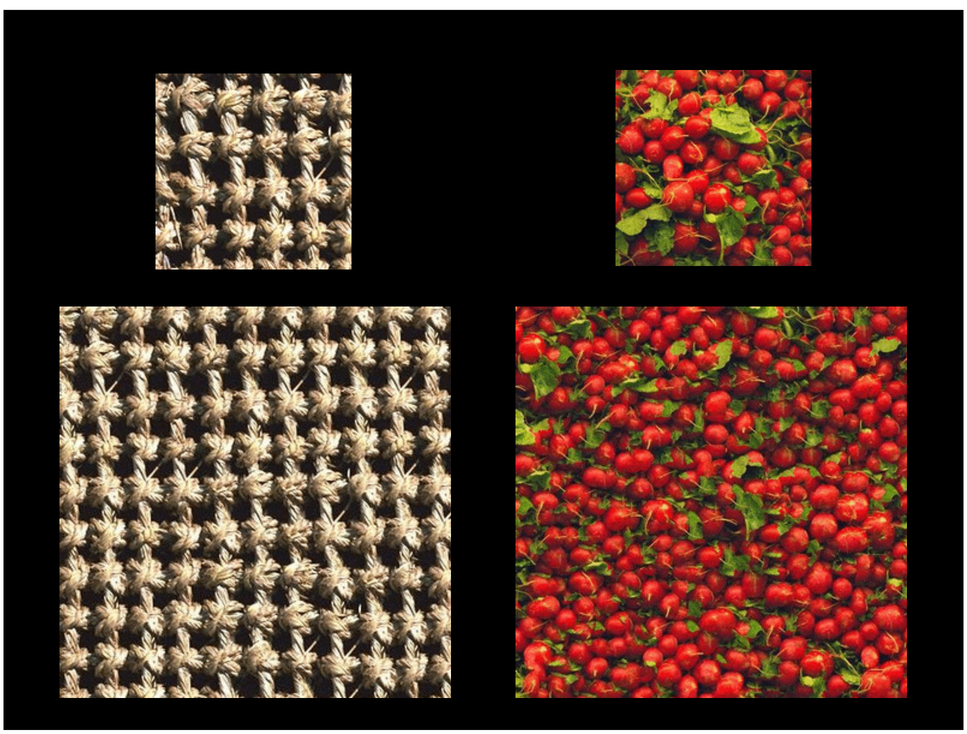
Minimal error boundary

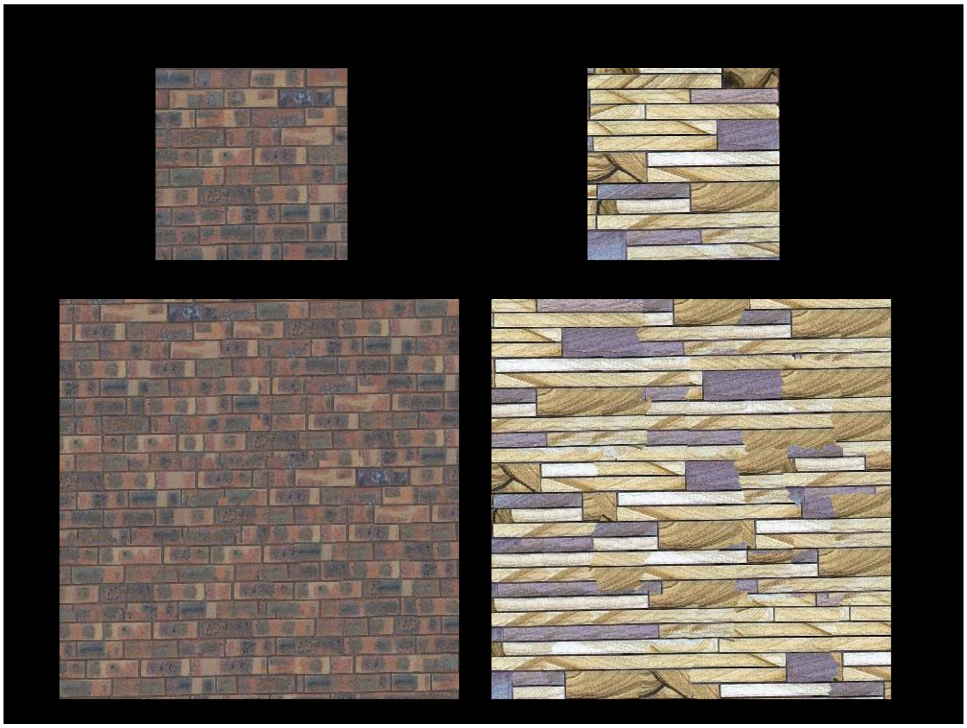


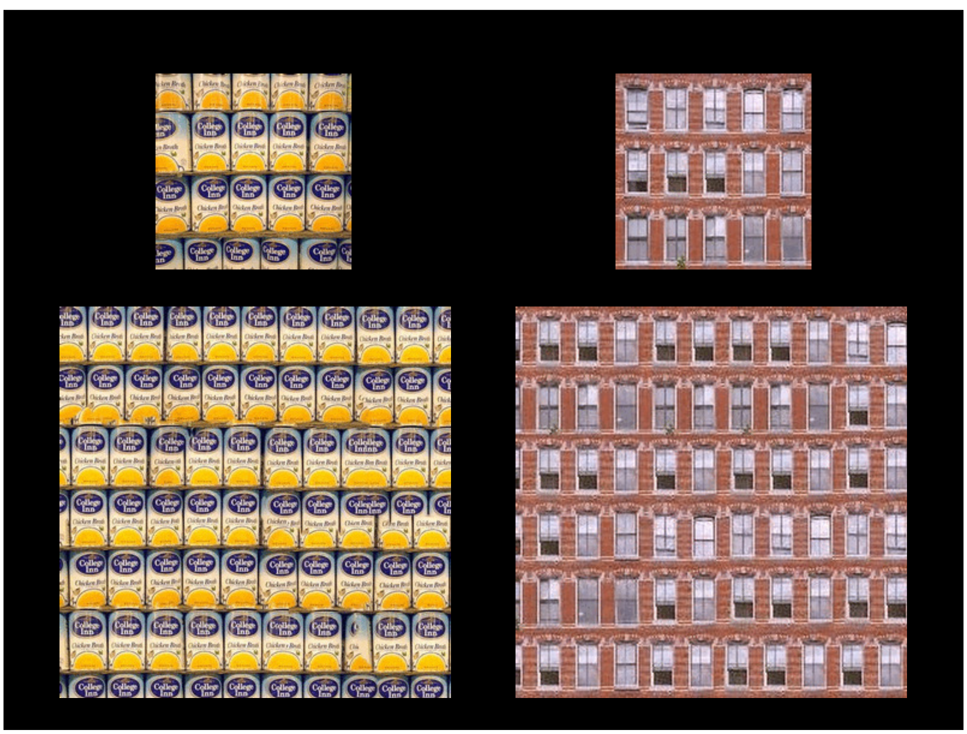
Our Philosophy

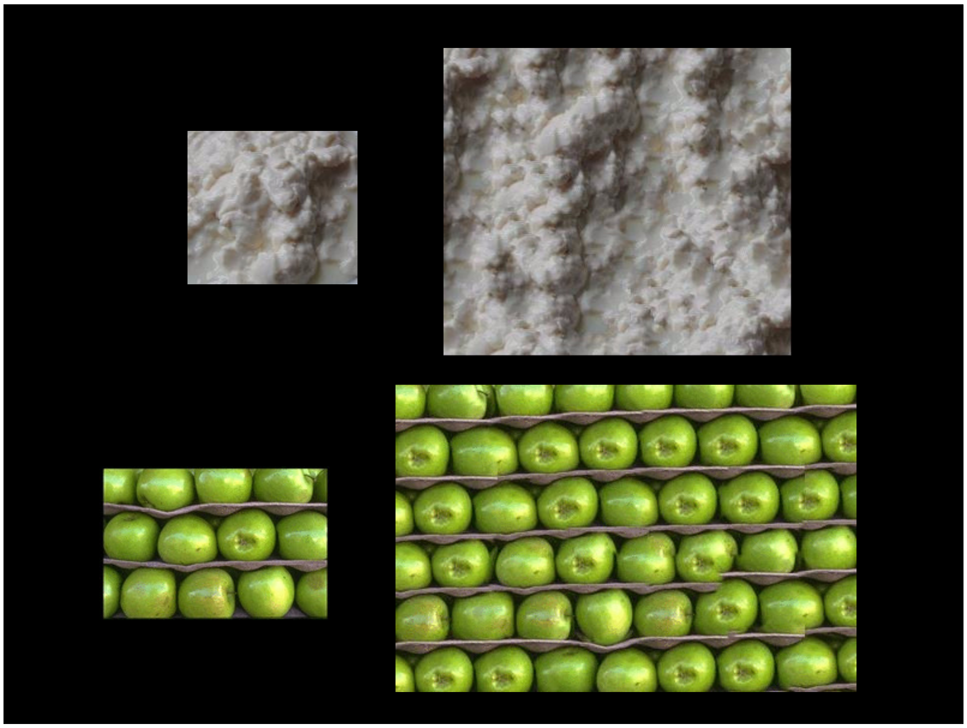
- The “Corrupt Professor’s Algorithm”:
  - Plagiarize as much of the source image as you can
  - Then try to cover up the evidence
- Rationale:
  - Texture blocks are by definition correct samples of texture so problem only connecting them together

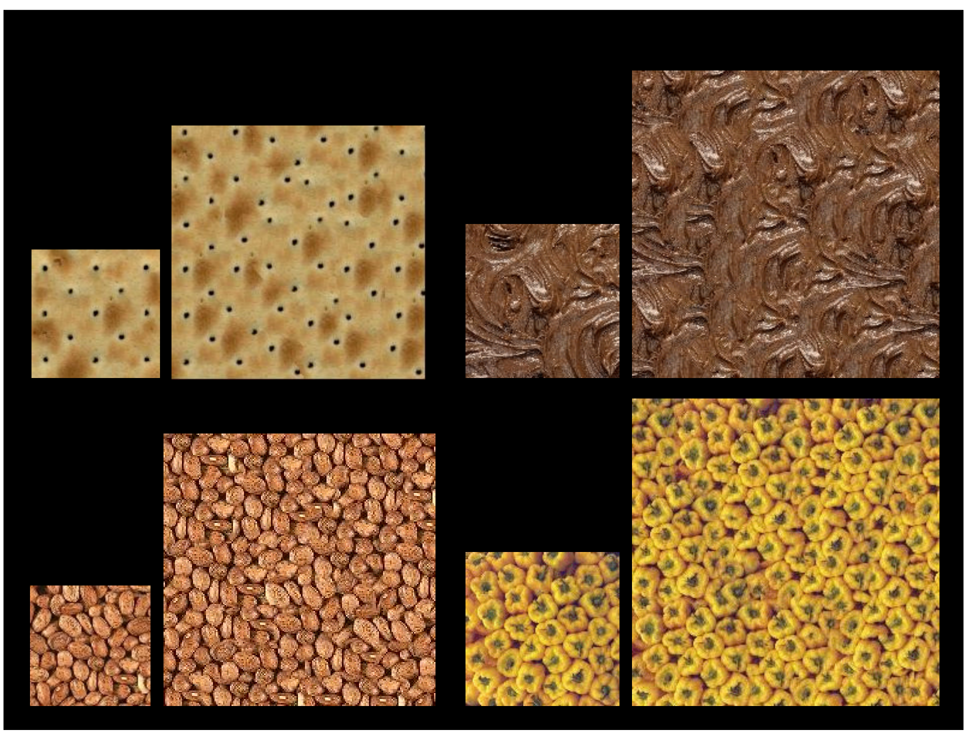




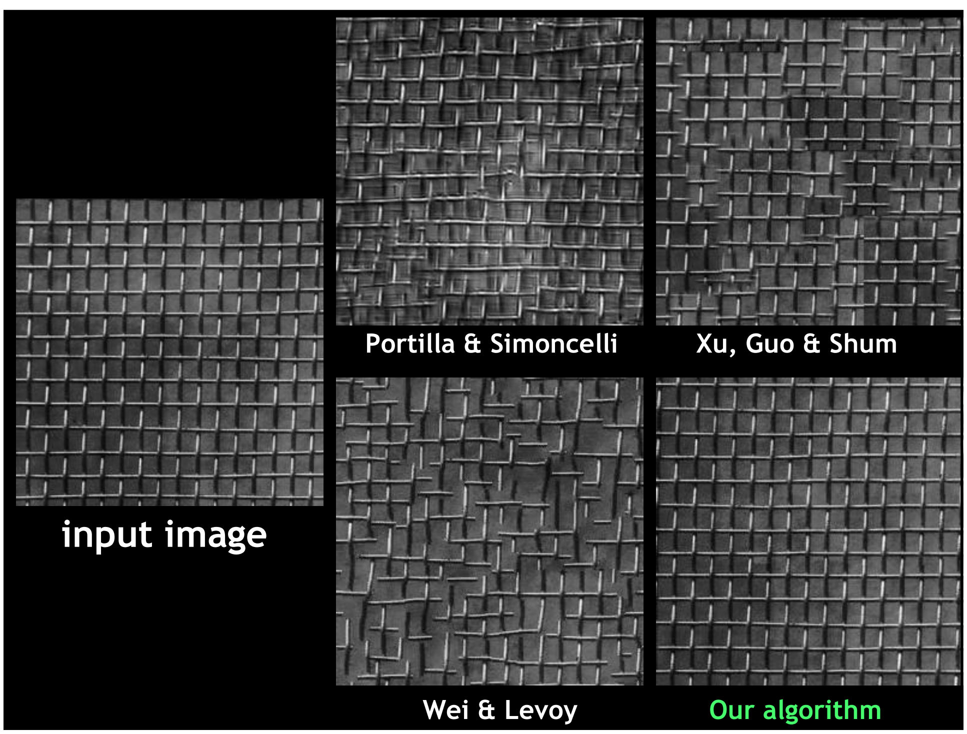


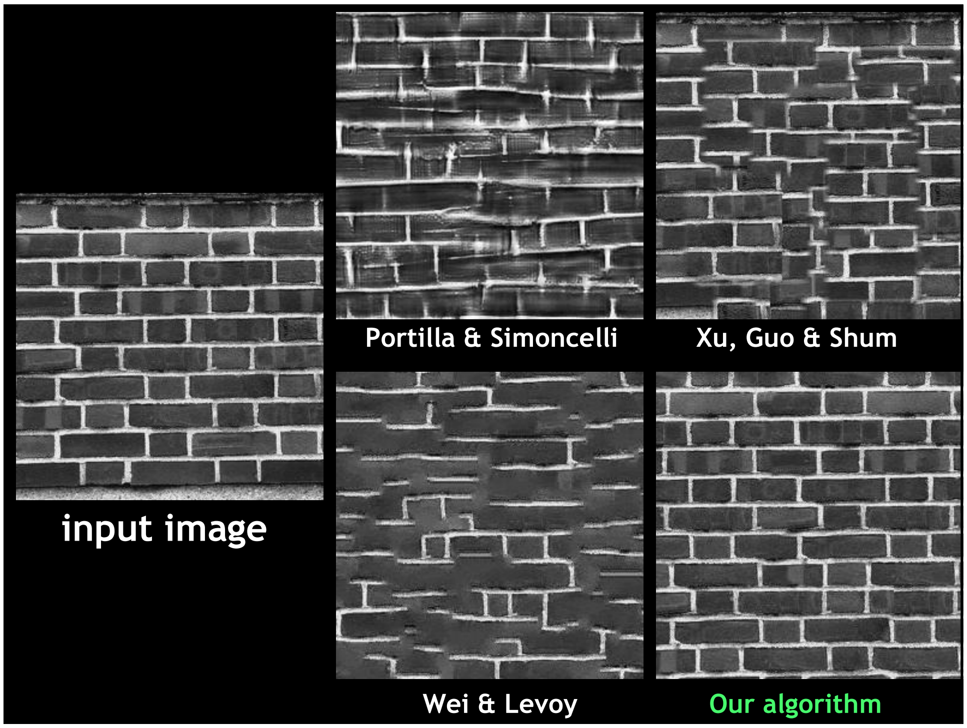

College Inn

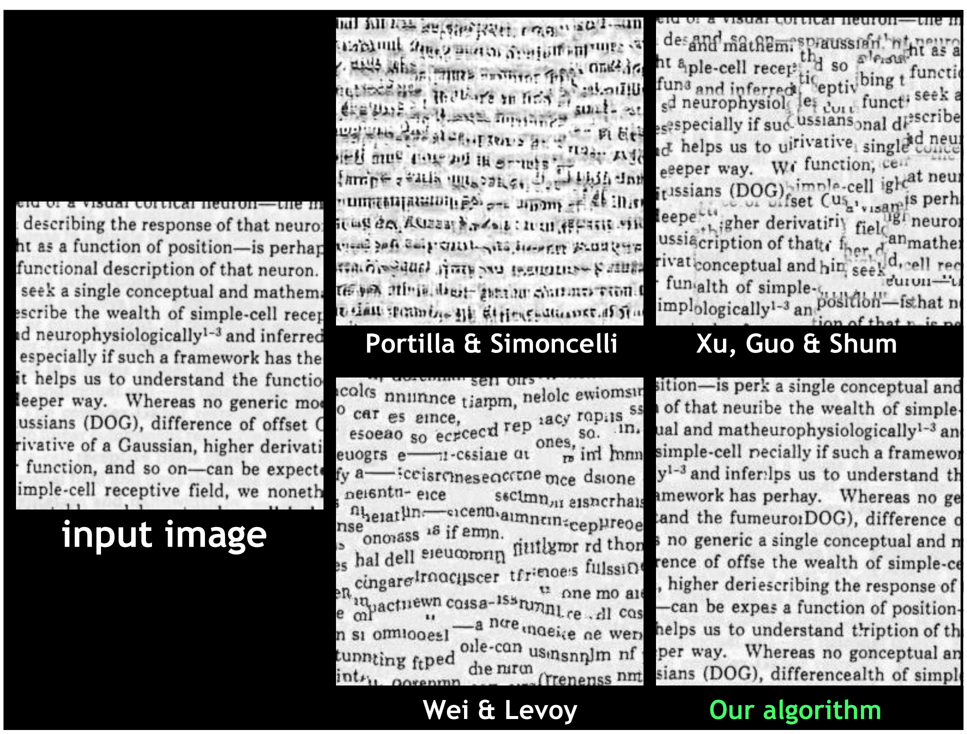




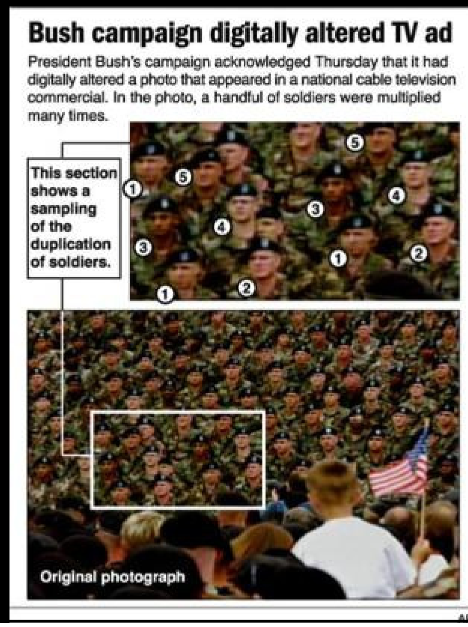

Failures
(Chernobyl
Harvest)


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


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


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

Political Texture Synthesis!

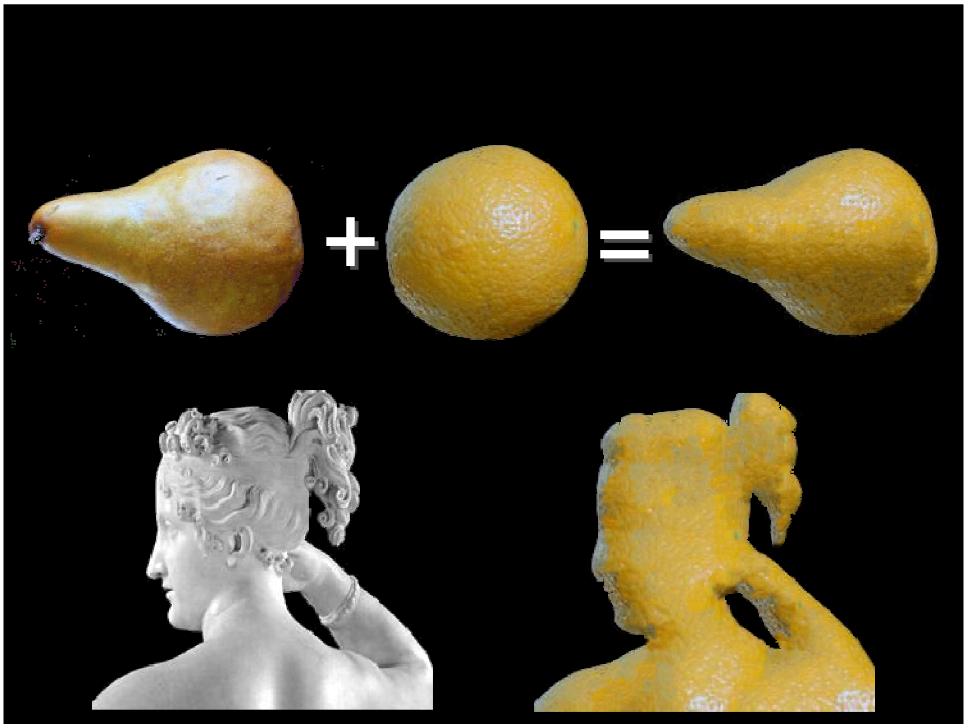


Application: Texture Transfer

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

Same as texture synthesis, except an additional constraint:

1. Consistency of texture
2. Similarity to the image being "explained"



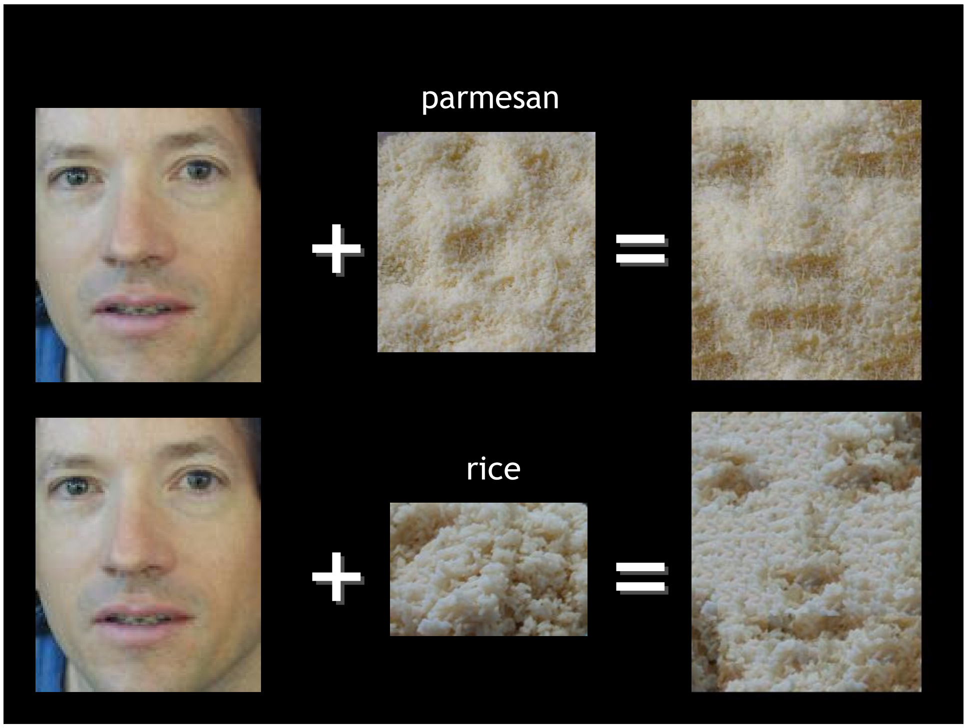


Image Analogies

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

- Solution:

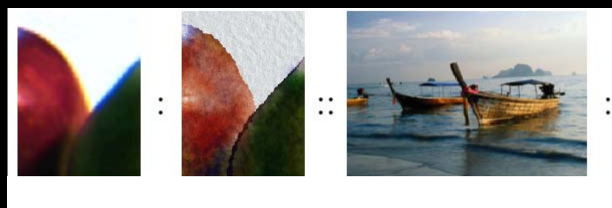
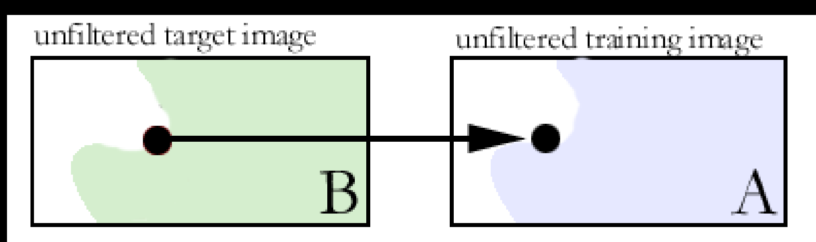


Image Analogies Implementation



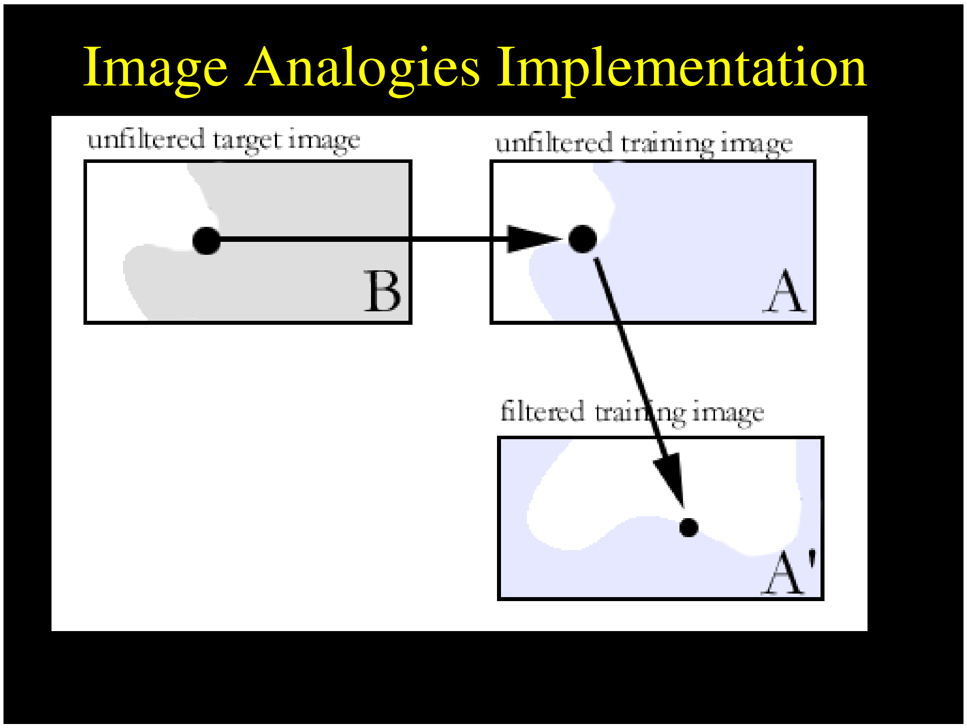


Image Analogies Implementation
unfiltered target image
unfiltered training image
B
A
filtered training image
A'

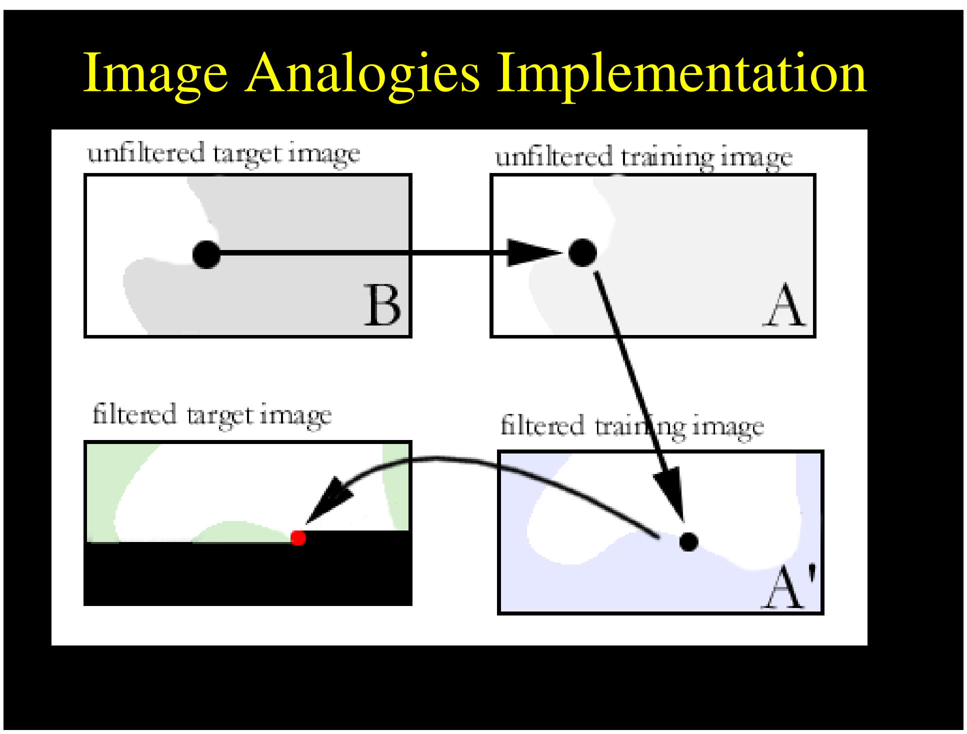
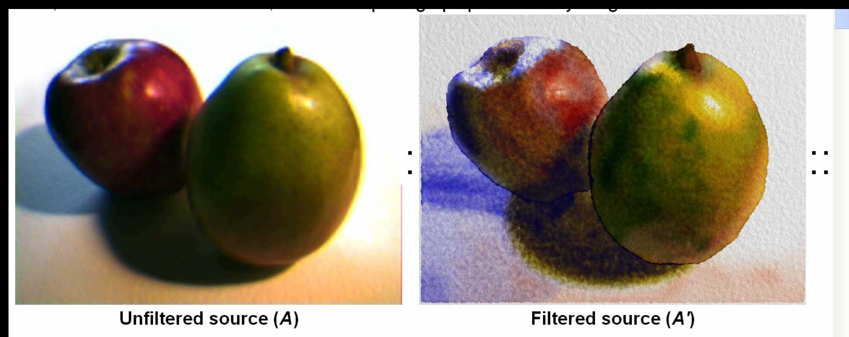
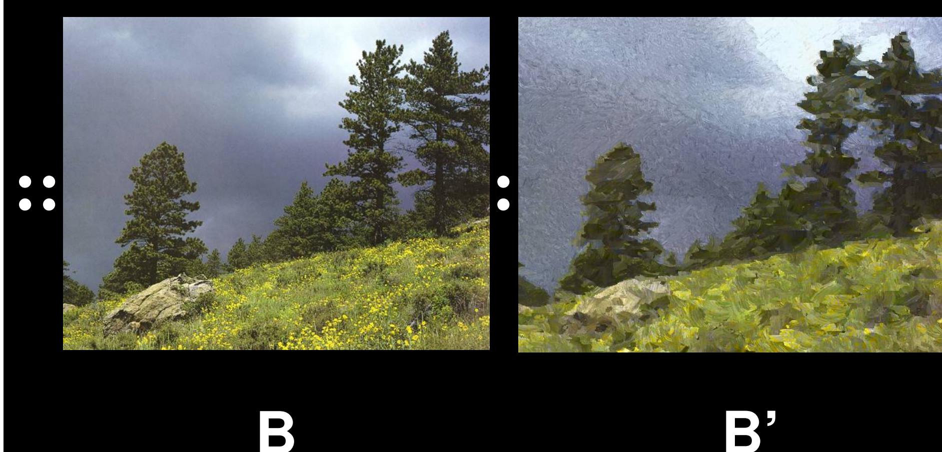
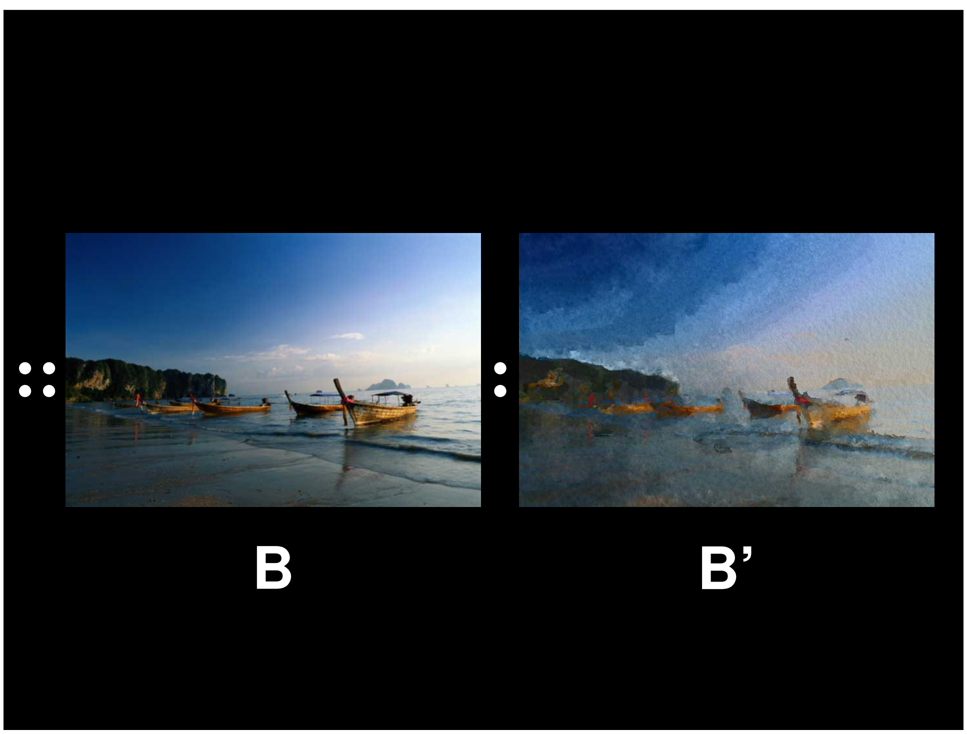


Image Analogies Implementation
unfiltered target image
unfiltered training image
B
A
filtered target image
filtered training image
A'

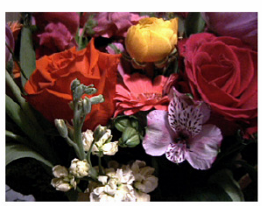
Training



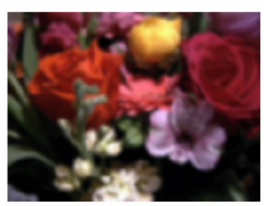



::
:
B
B’

Learn to Blur



Unfiltered source (A)

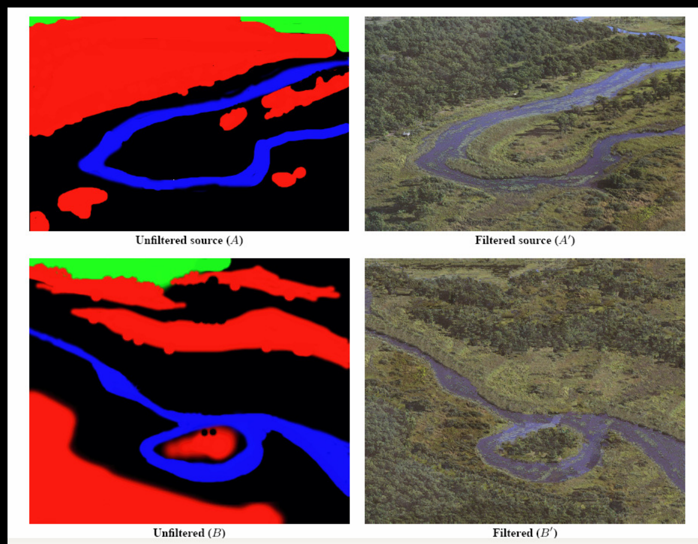


Filtered source (A')

Unfiltered target (B)

Filtered target (B')

Texture by Numbers



Colorization