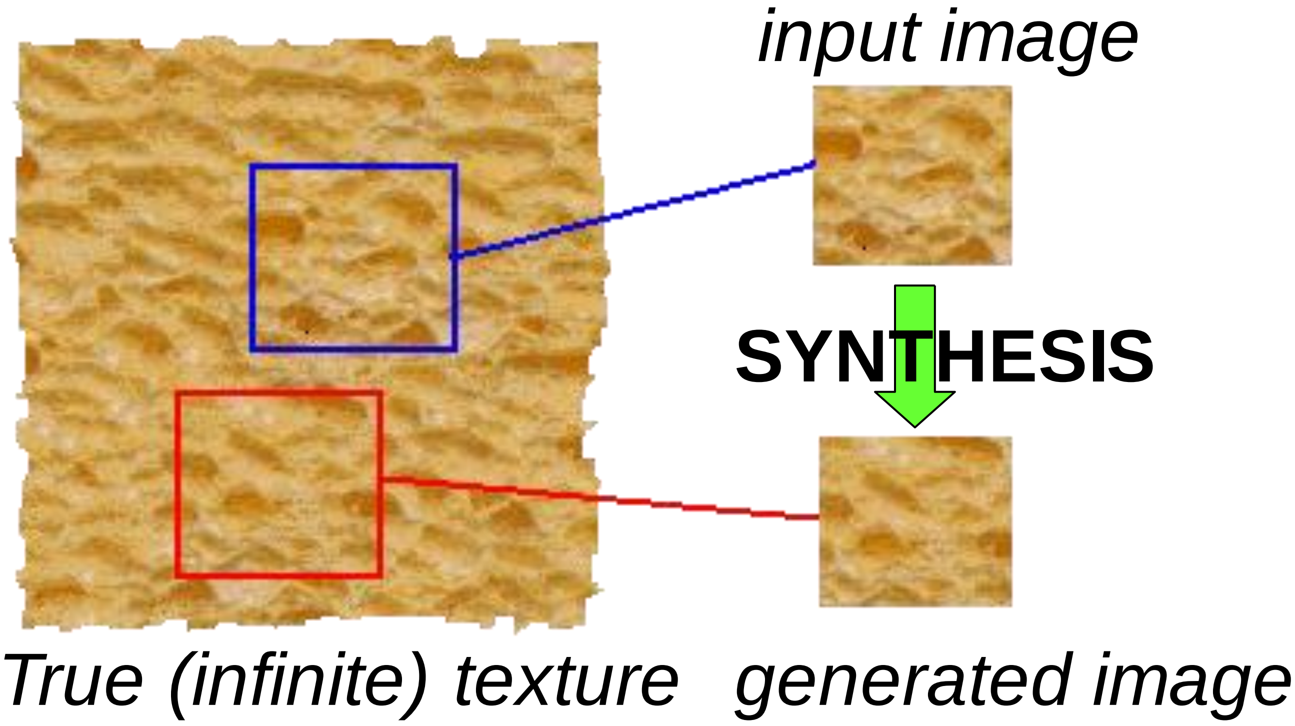
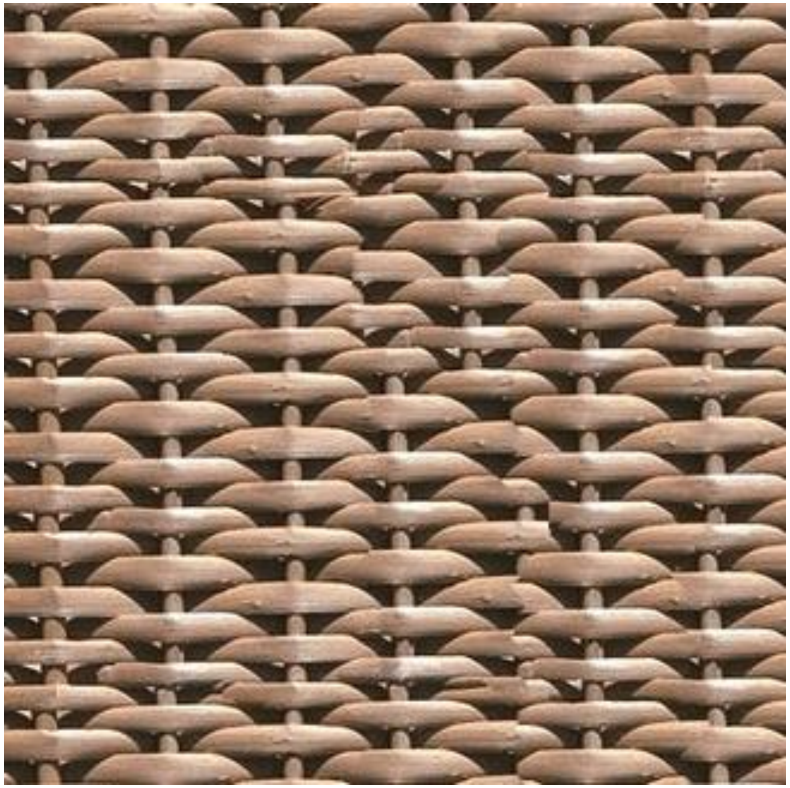
texture synthesis

Given an input sample texture synthesize a texture that is sufficiently different from the given sample texture, yet appears perceptually to be generated by the same underlying stochastic process.

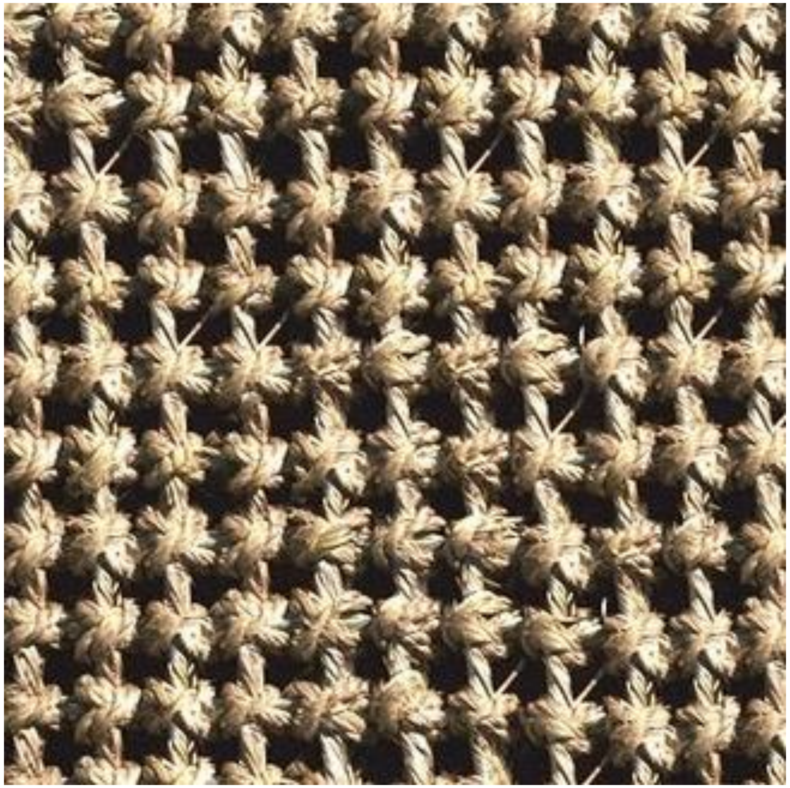


Slides taken from A. Efros & L. Svetlana

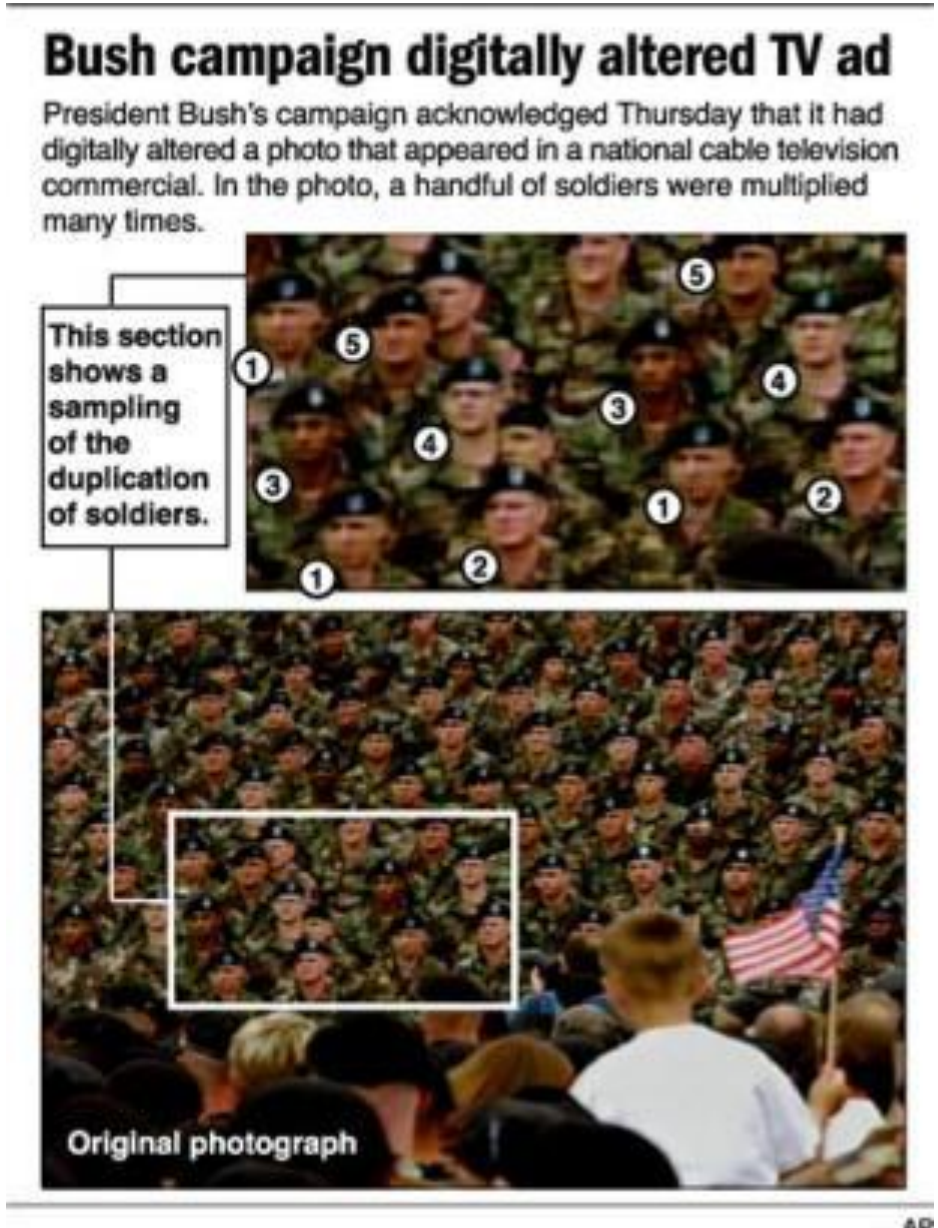







Political Texture Synthesis!

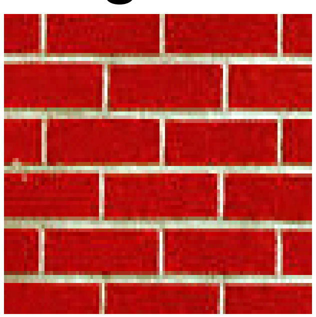


Classification of texture

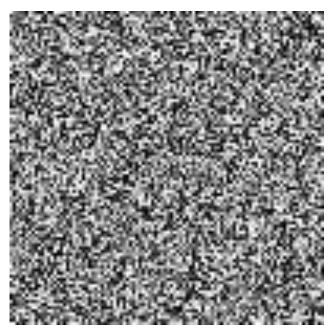
Traditionally textures has been classified as:

- regular : repeated textons
- stochastic without explicit textons

regular



stochastic



both?

Some previous approaches

- multi-scale filter response histogram matching [Heeger and Bergen,'95]
- sampling from conditional distribution over multiple scales [DeBonet,'97]
- filter histograms with Gibbs sampling [Zhu et al,'98]
- matching 1st and 2nd order properties of wavelet coefficients [Simoncelli and Portilla,'98]

These methods focus on both `synthesis' and `analysis'

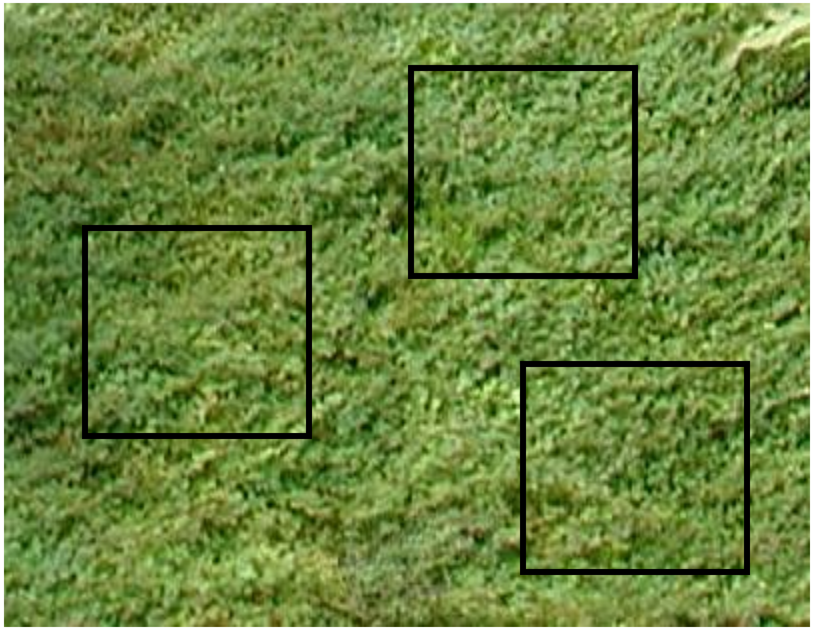
Methods for purely synthesis

- goals:
  - preserve local structure
  - model wide range of real textures

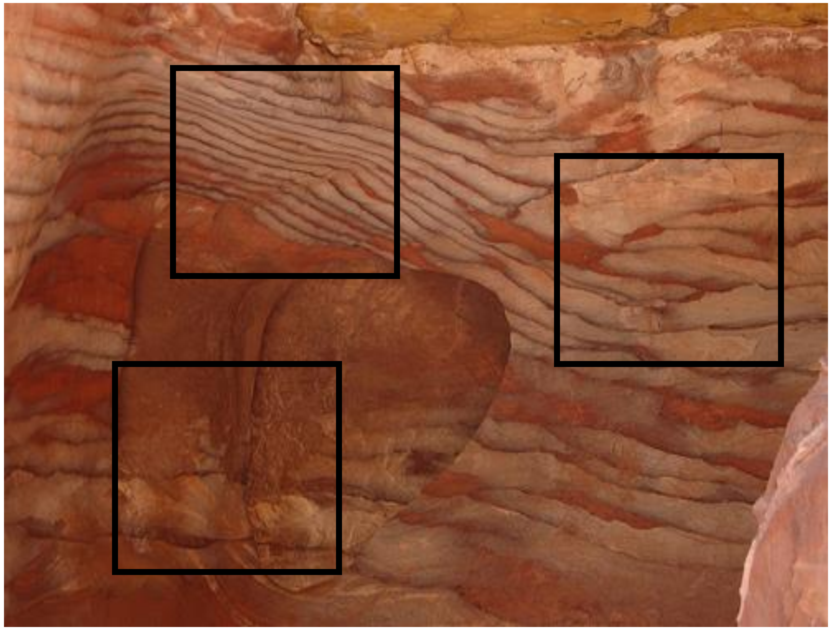
- method:
  - inspired by N-gram language model of Shannon, texture is modelled as Markov Random Field (MRF)
  - texture is “grown” one pixel at a time
  - conditional pdf of a pixel given its neighbors **synthesized thus far** is estimated by searching the the sample image for similar neighborhoods

Statistical modeling of texture

- Assume stochastic model of texture (*Markov Random Field*)
- *Stationarity*: the stochastic model is the same regardless of position



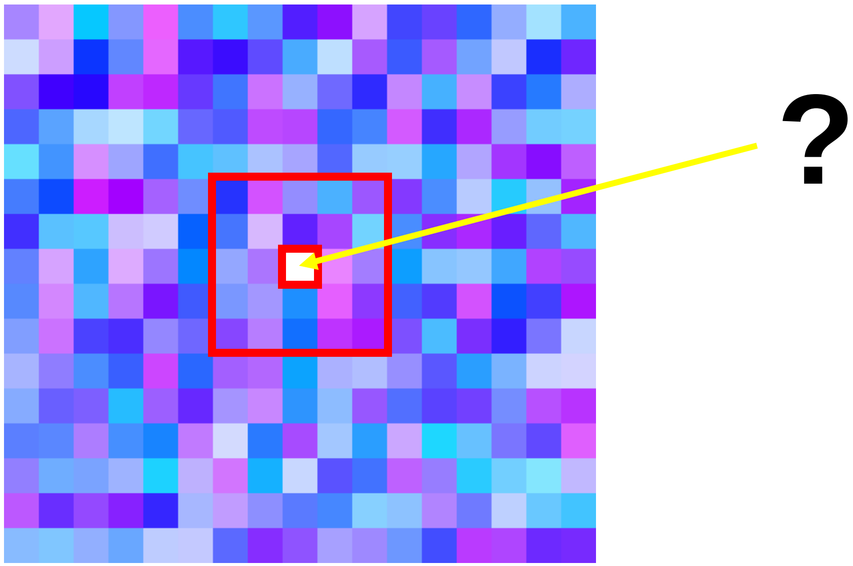
stationary texture



non-stationary texture

Statistical modeling of texture

- Assume stochastic model of texture (*Markov Random Field*)
- *Stationarity*: the stochastic model is the same regardless of position
- *Markov property*:
  p(pixel | rest of image) = p(pixel | neighborhood)



N-gram model of the English language

Shannon: Model language as a generalized Markov chain, where a set of n letters (words) completely determine the pdf of the next letter (word).

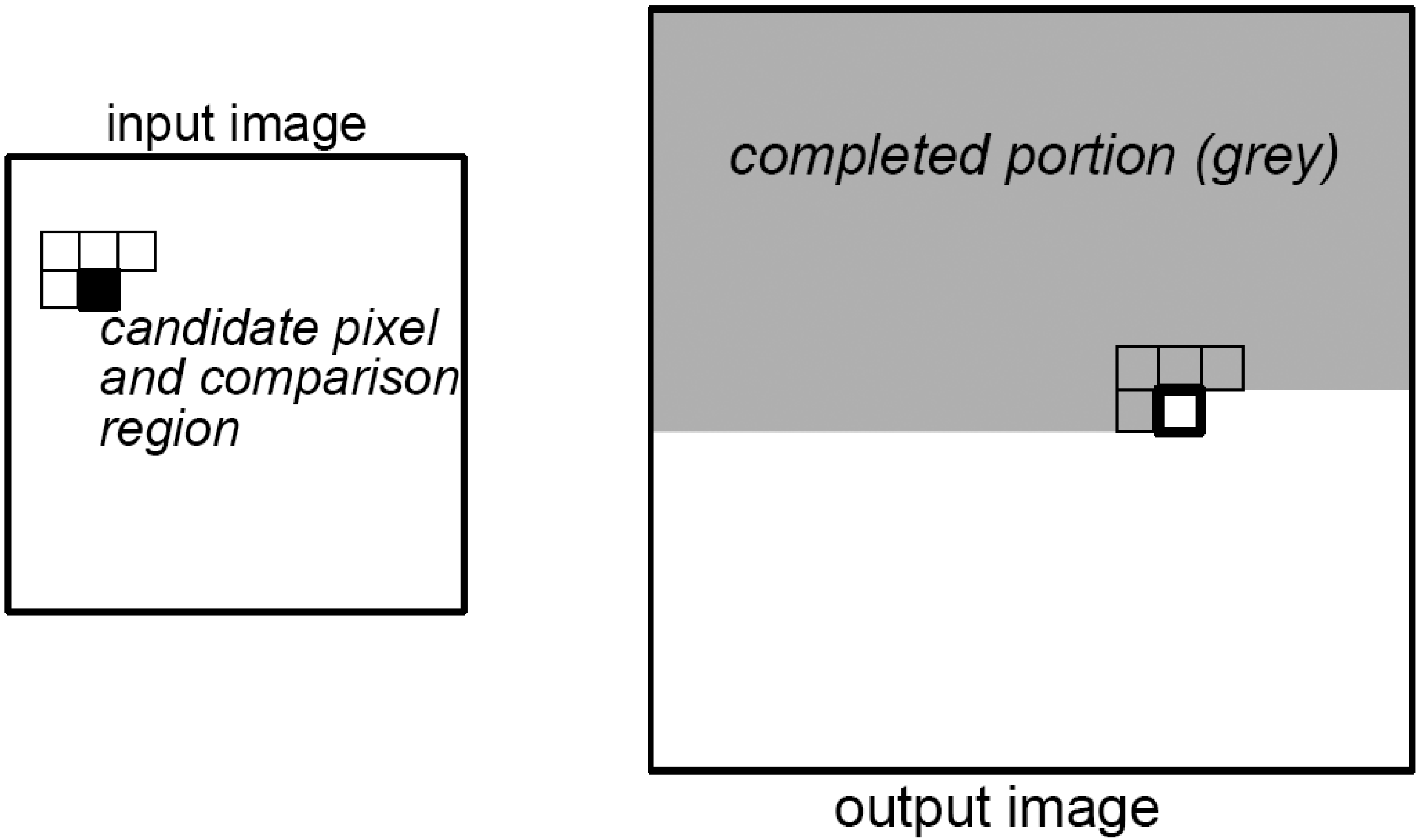
Results (using alt.singles corpus) [*Mark V. Shaney*]:

"One morning I shot an elephant in my arms and kissed him."

"I spent an interesting evening recently with a grain of salt"

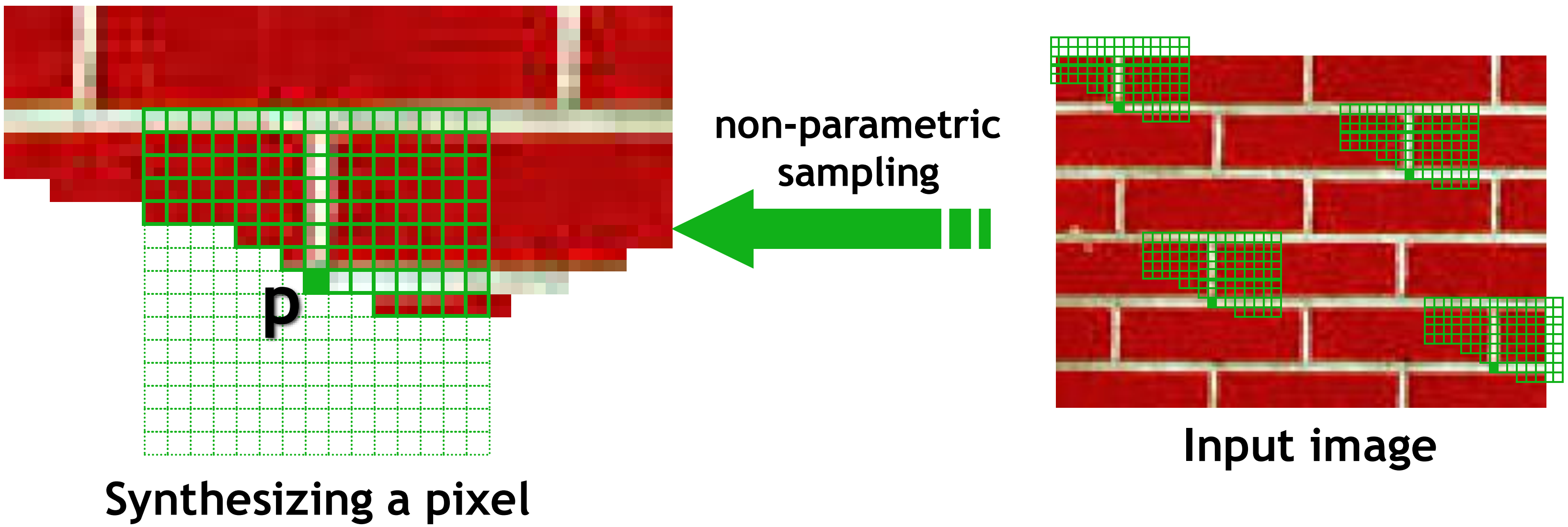
Assuming Markov property, texture can be modeled as a MRF

Efros & Leung Algorithm



Idea initially proposed in 1981 (Garber ’81), but dismissed as too computationally expensive!

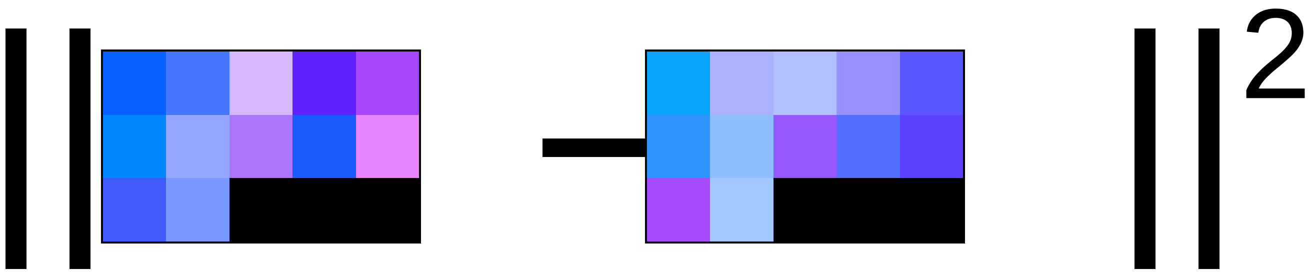
Efros & Leung Algorithm



- 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

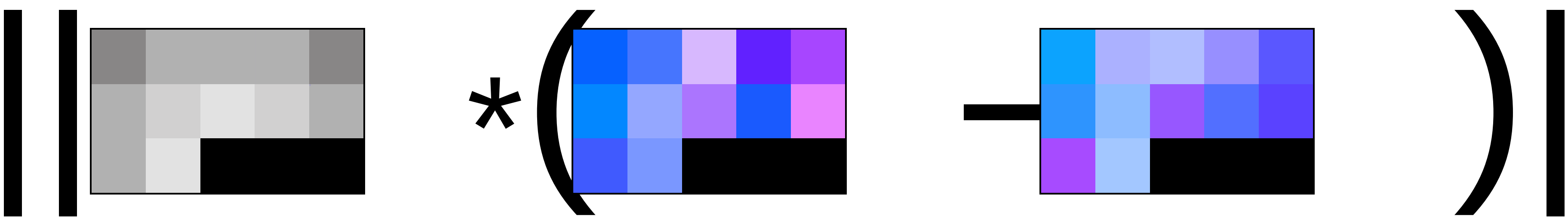
Finding matches

- Sum of squared differences (SSD)



Finding matches

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



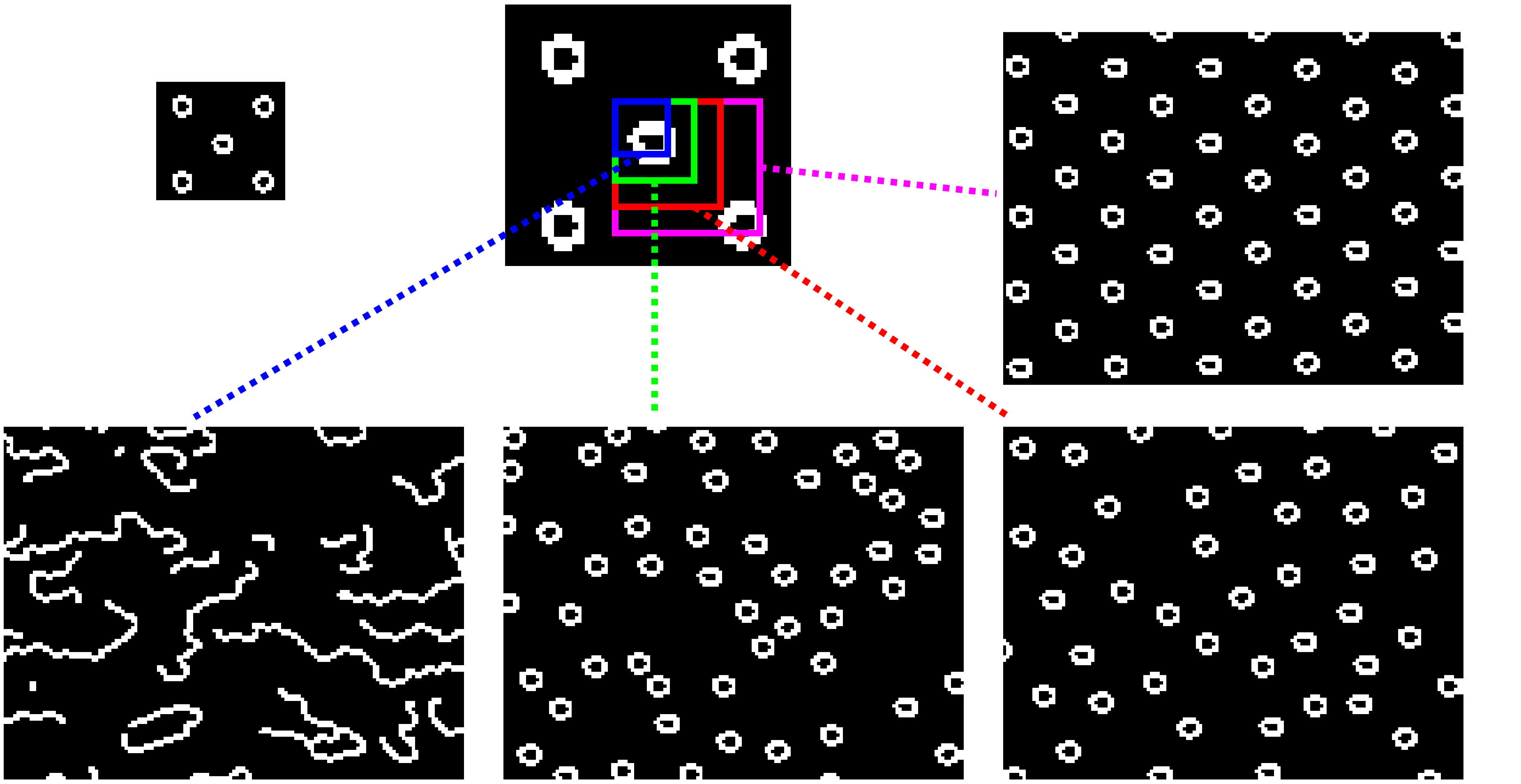
Details

- Random sampling from the set of candidates vs. picking the best candidate
- Initialization
  - Start with a few rows of white noise and grow in scanline order
  - Start with a “seed” in the middle and grow outward in layers
- Hole filling: growing is 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

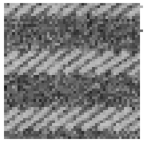
Growing texture on pixel at the time

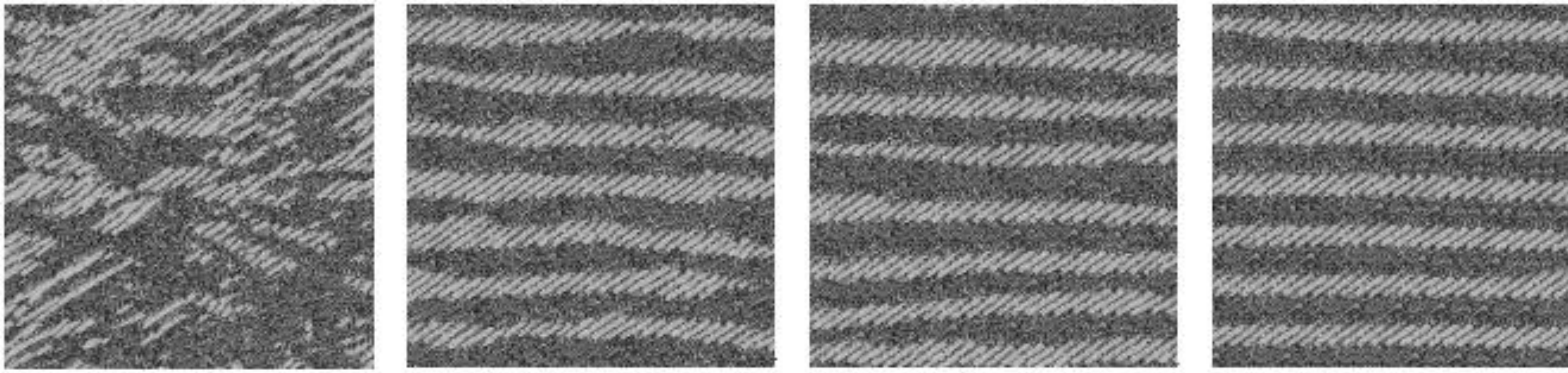
- User defined window size indicates the randomness of the texture
- To grow from from scratch a 3x3 random seed from the sample is used
- Unless no close match is found pixels with most neighbors are synthesized first
- Importance of Gaussian-weighted similarity measure

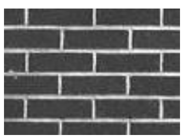
Neighborhood window size / Randomness parameter

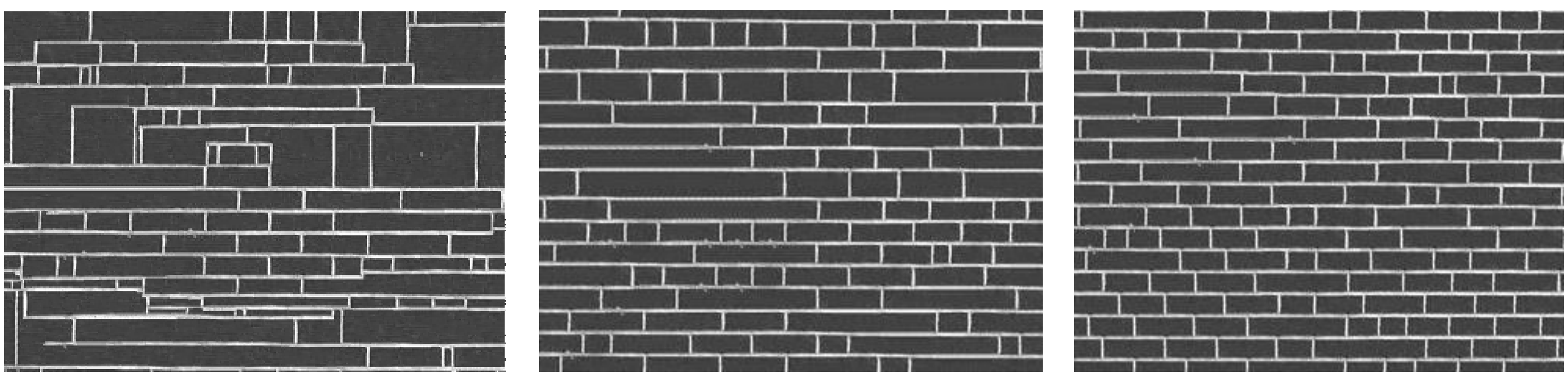


More Synthesis Results





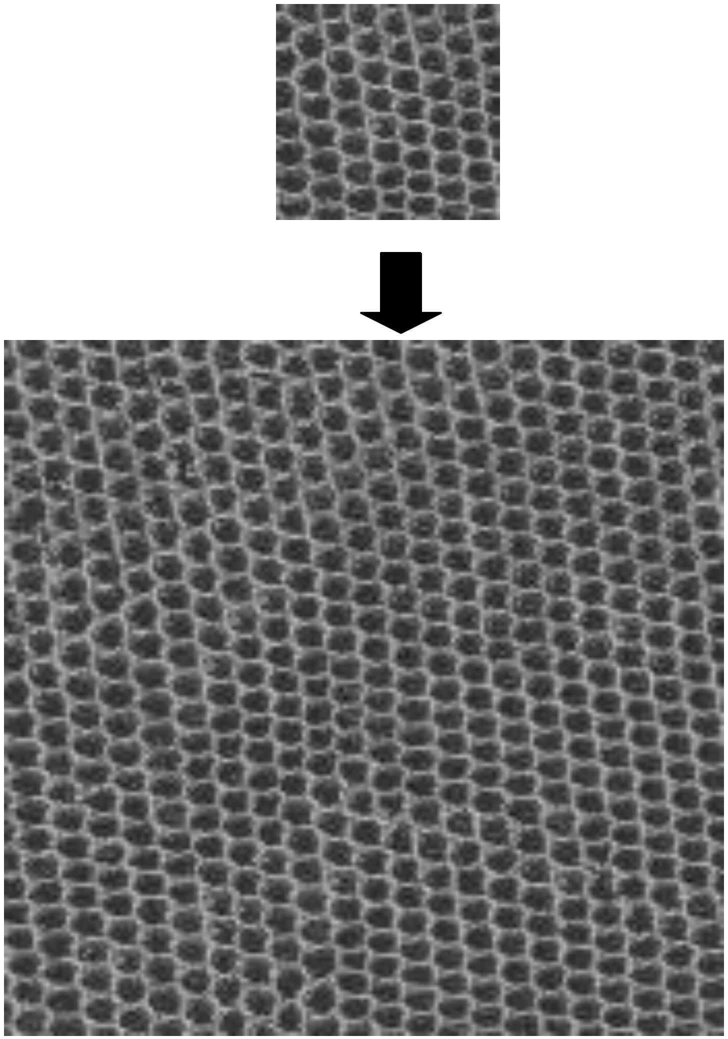




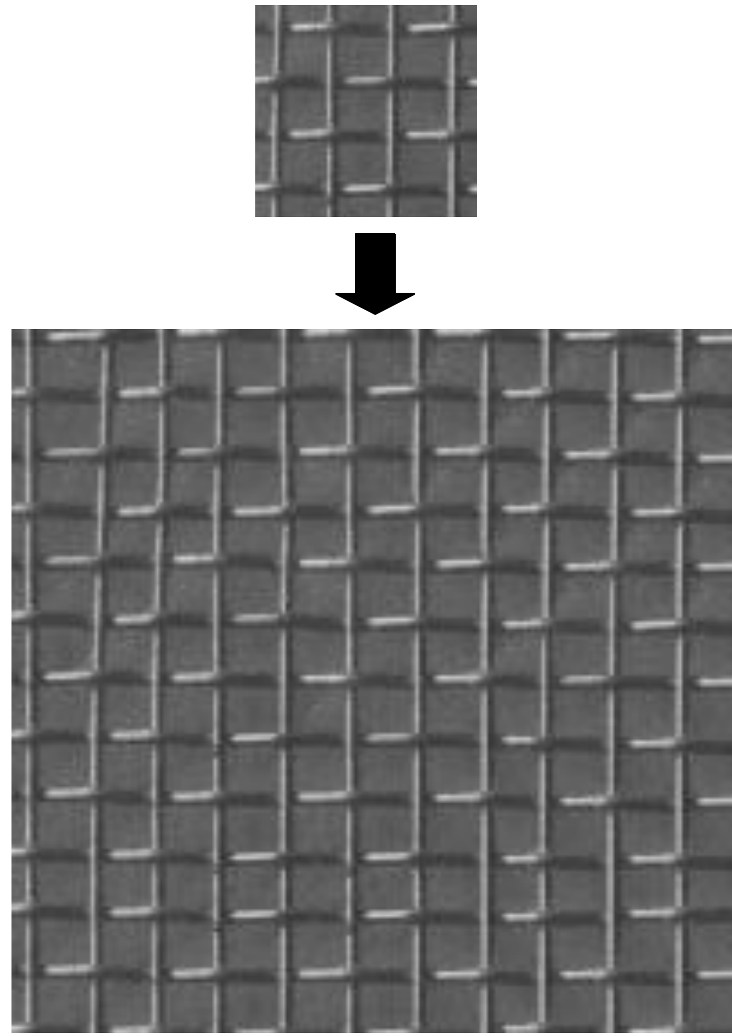
Increasing window size

Results

reptile skin

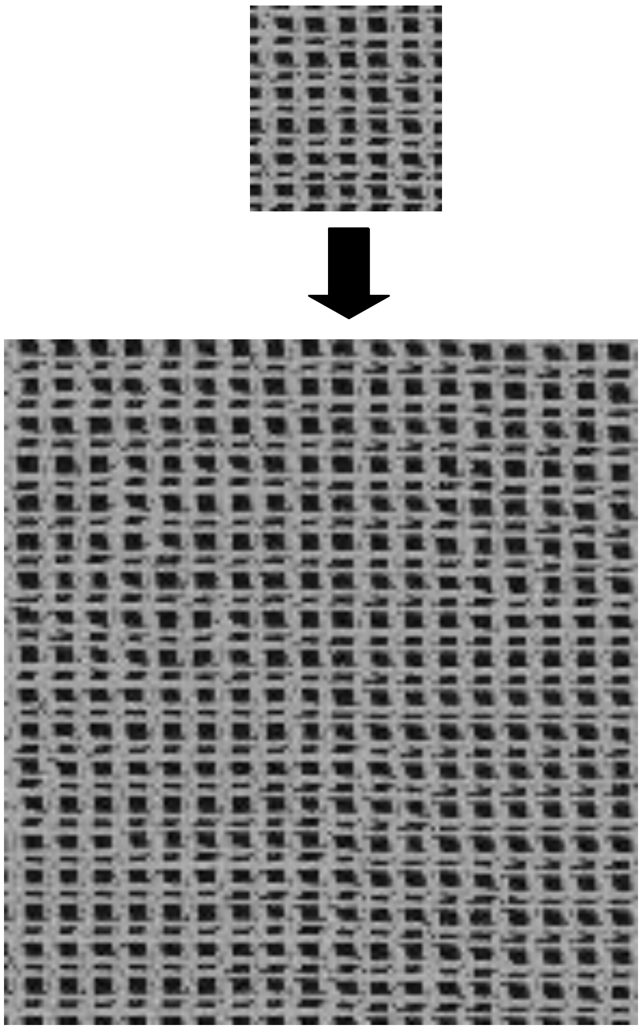


aluminium wire

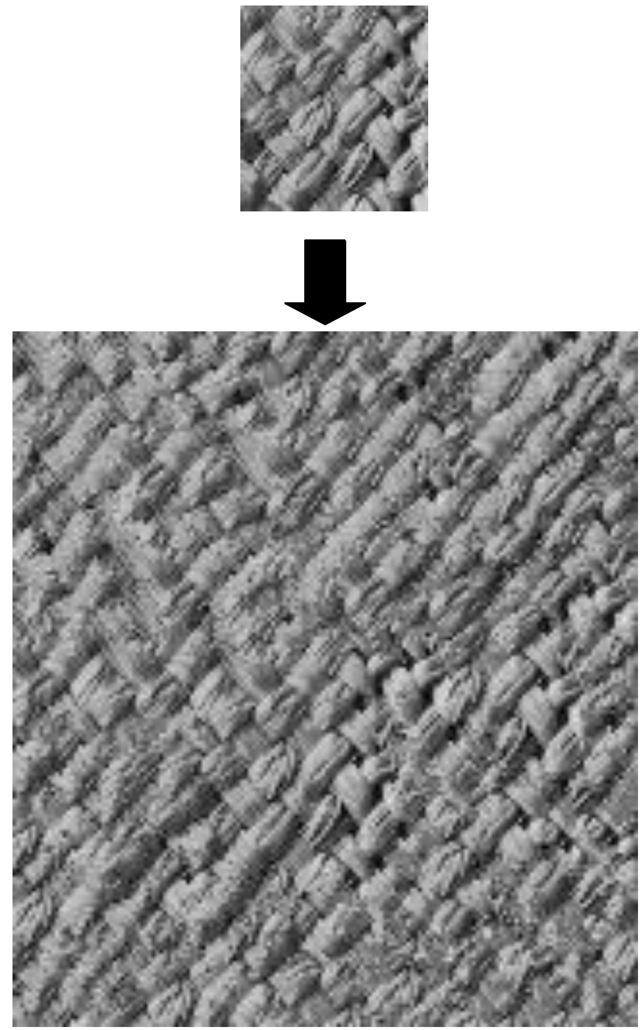


More results

French canvas

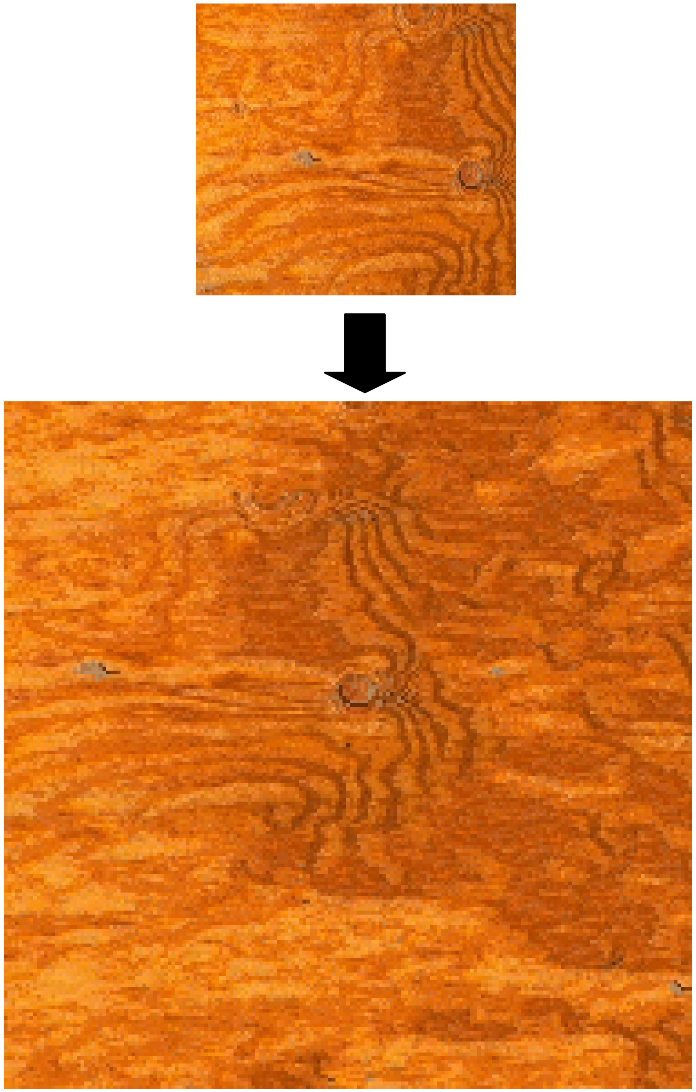


rafia weave

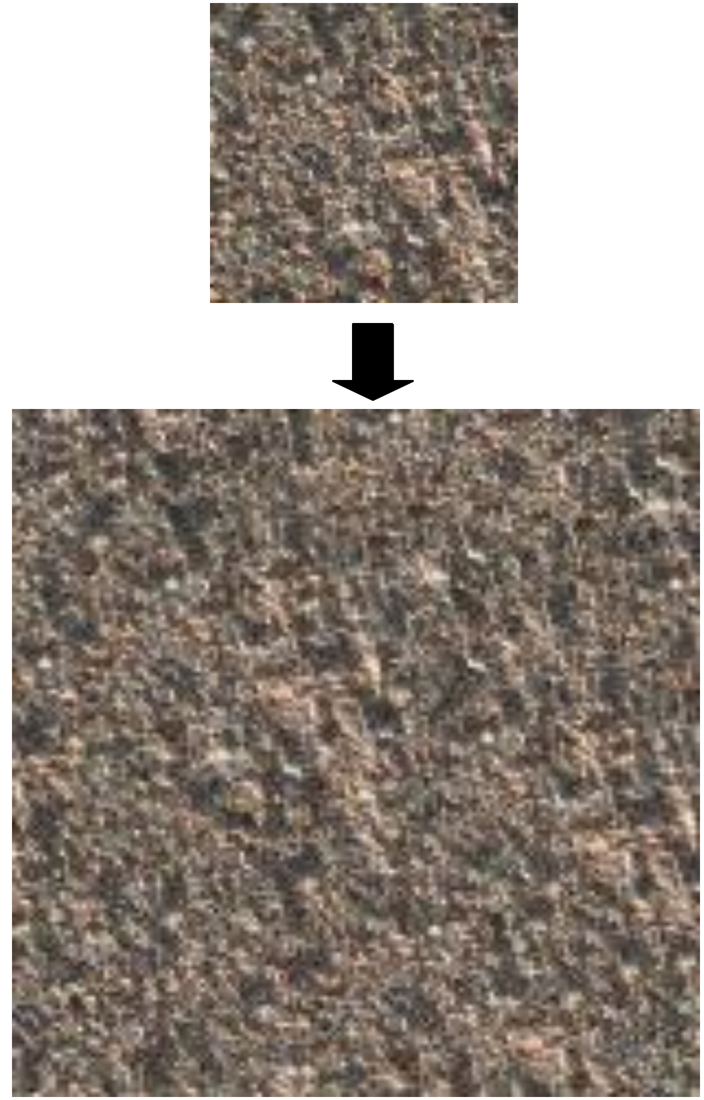


More results

wood



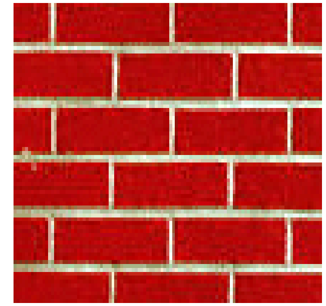
granite

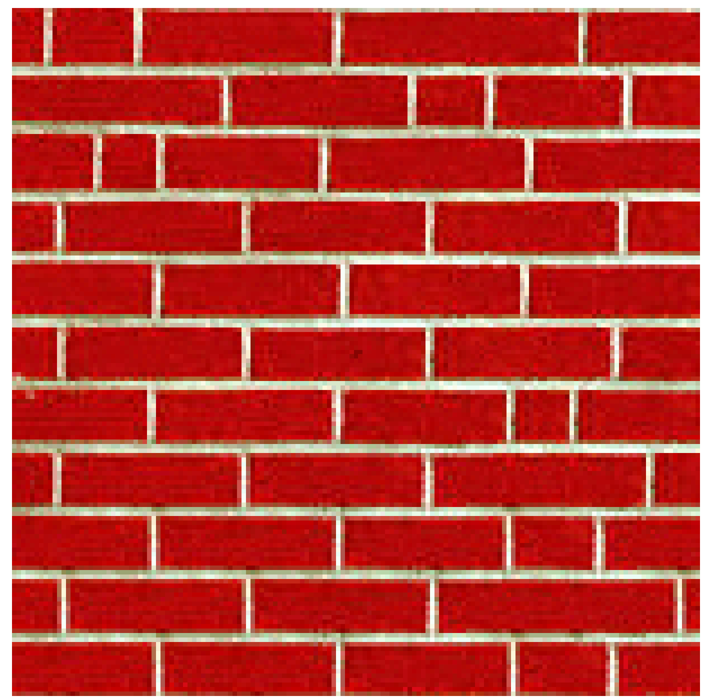


More results

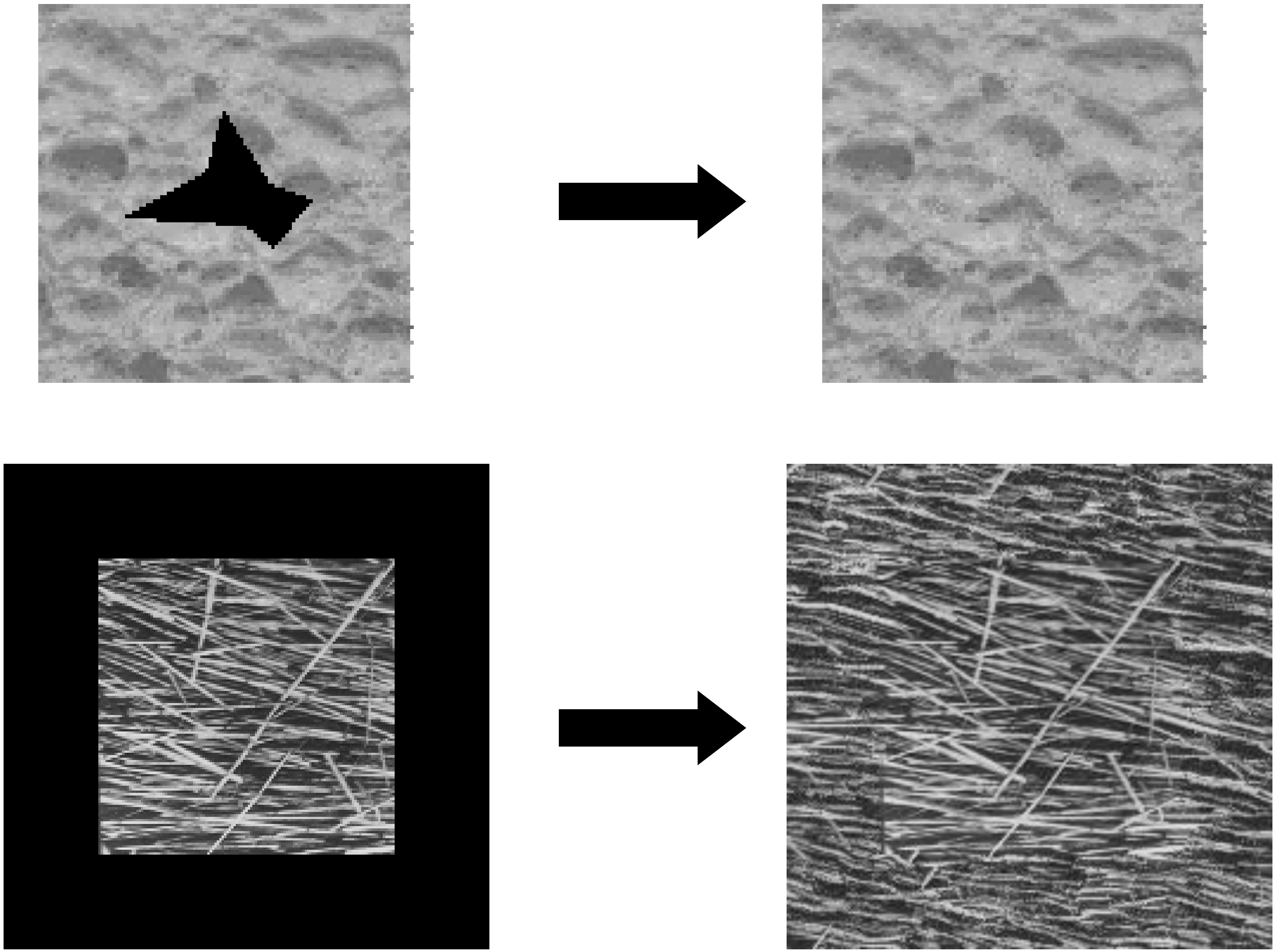
white bread

brick wall



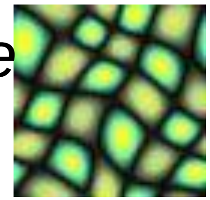


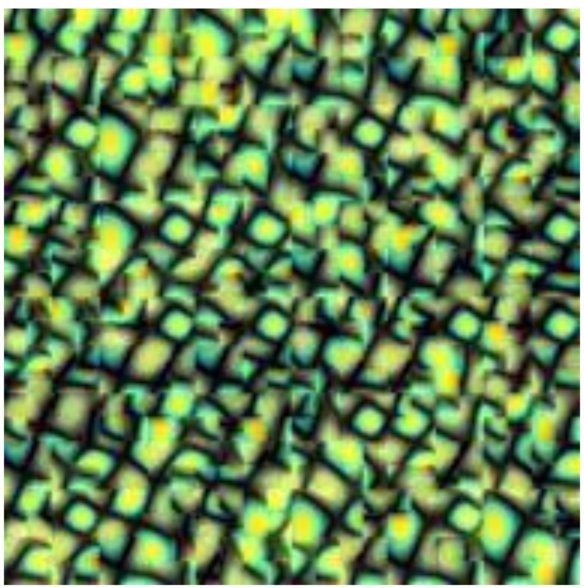
Constrained synthesis



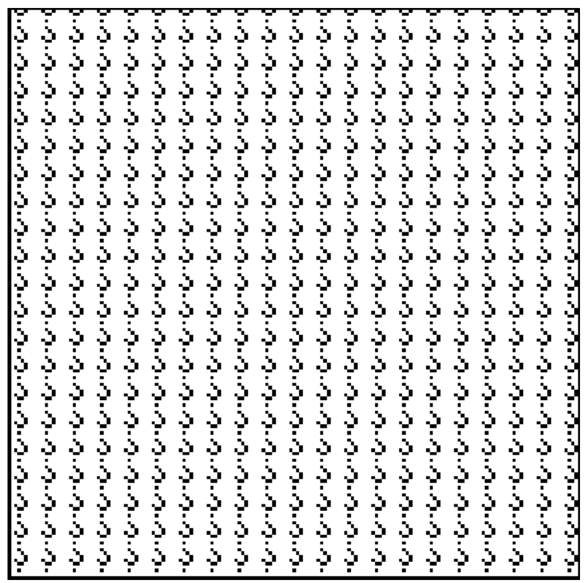
Visual comparison

Synthetic tilable texture

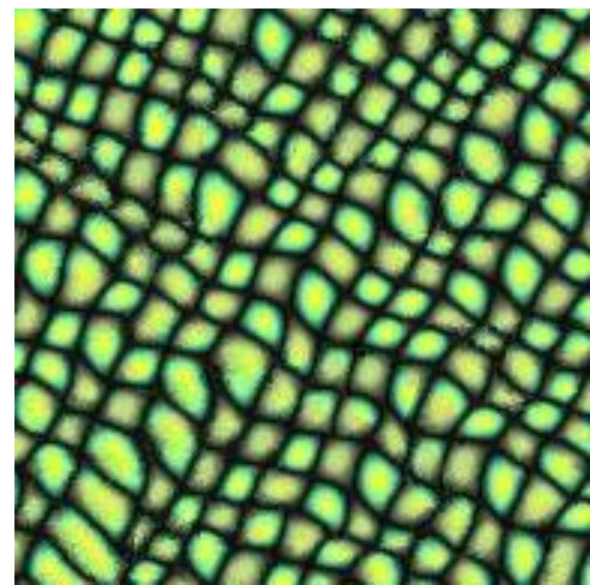




[DeBonet, ‘97]

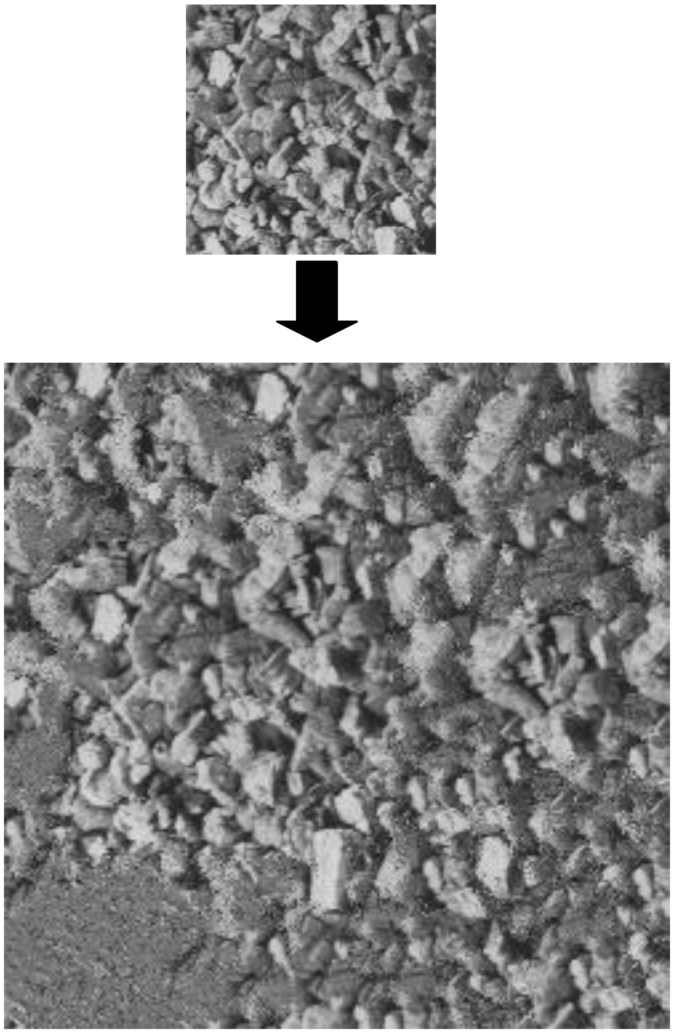


Simple tiling

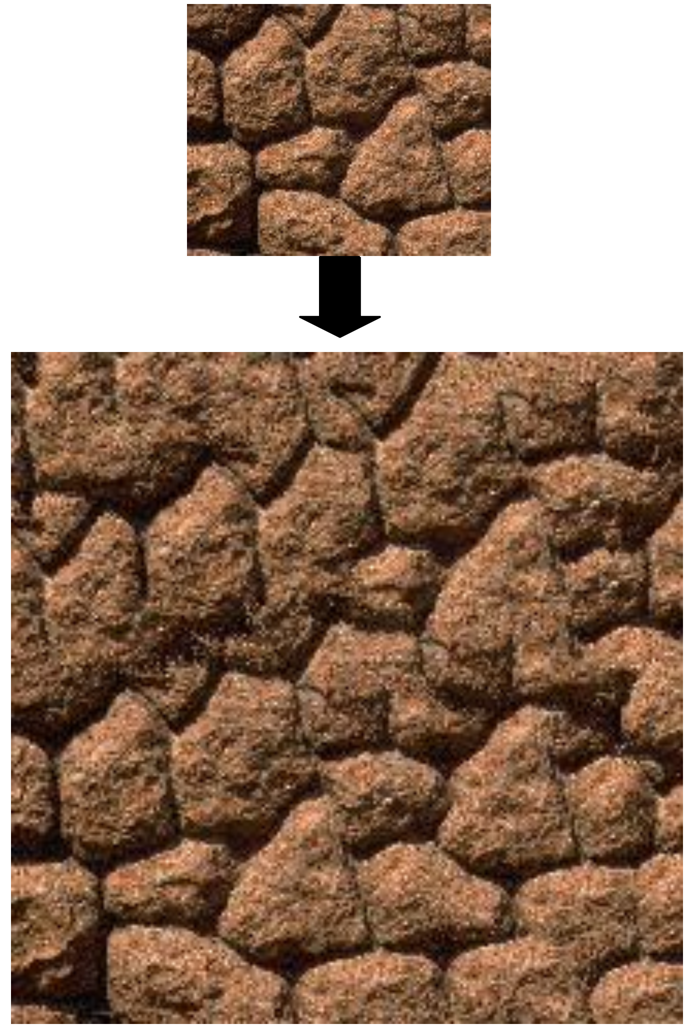


Our approach

Failure cases

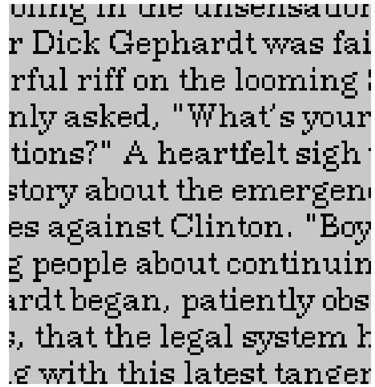


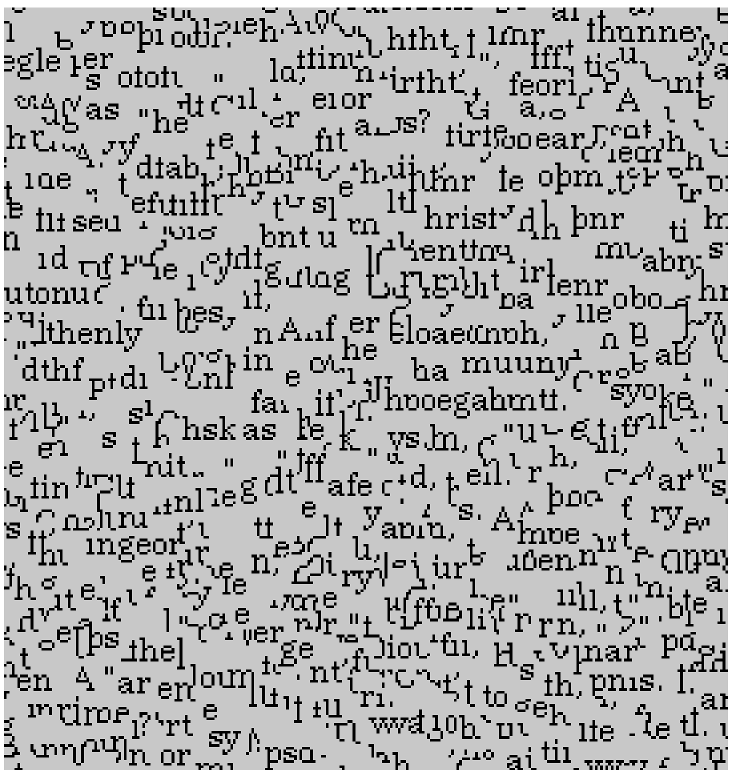
Growing garbage

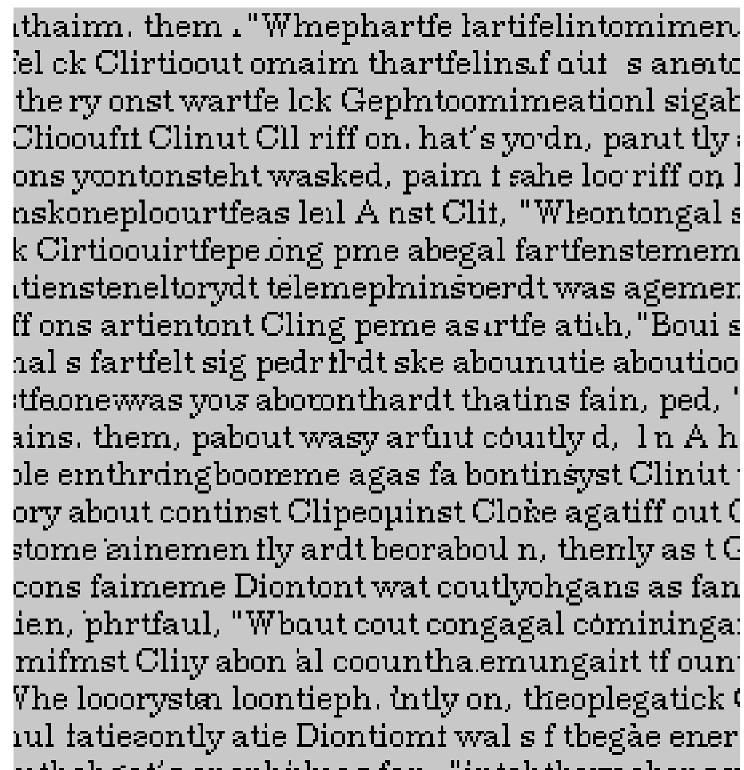


Verbatim copying

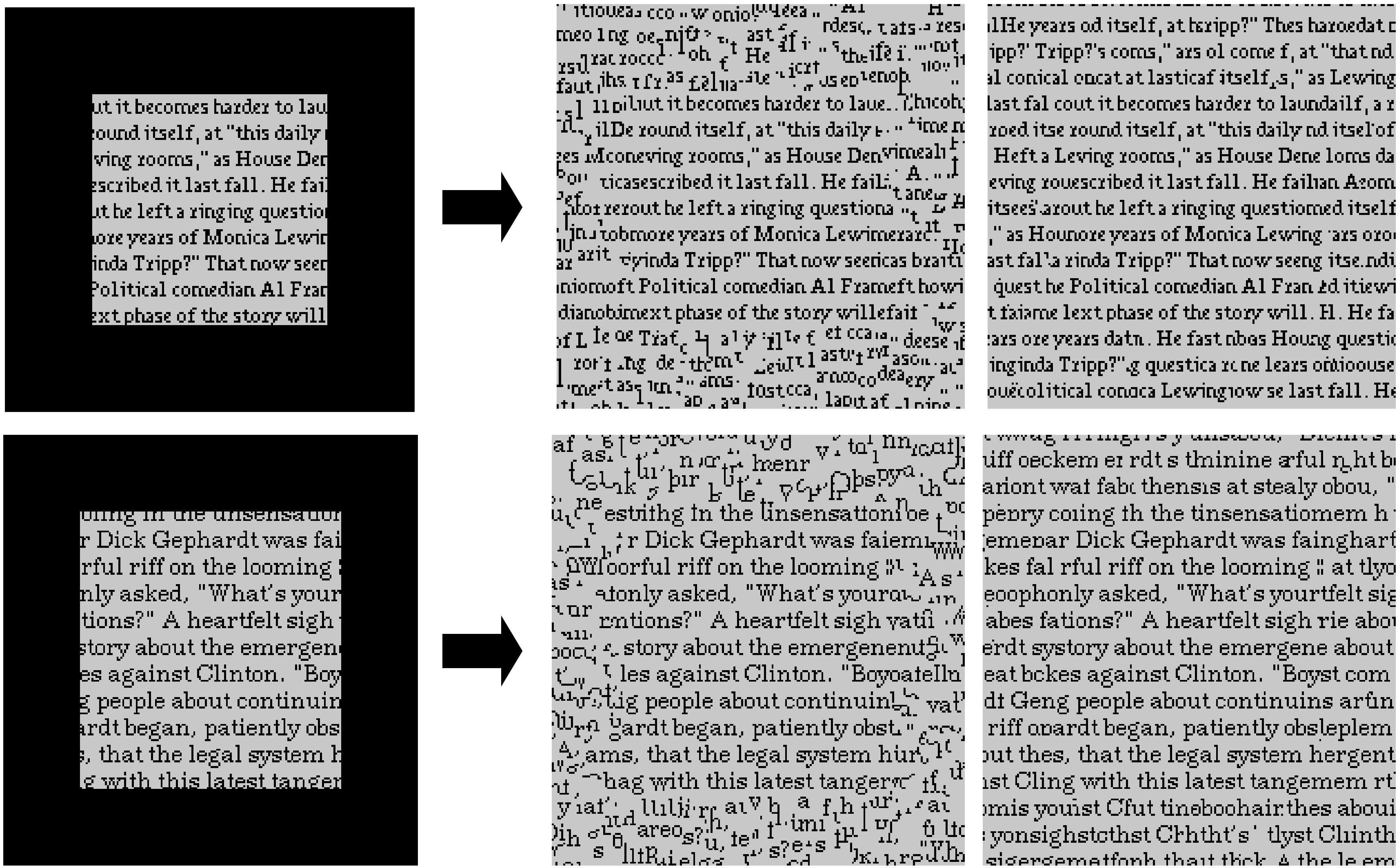
Homage to Shannon





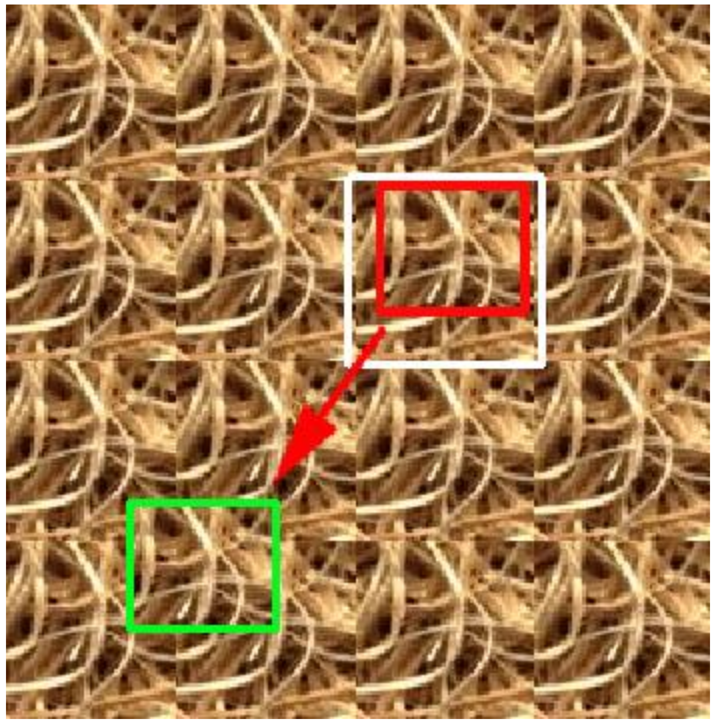


Constrained text synthesis



Chaos Mosaic [Xu, Guo & Shum, ‘00]

input



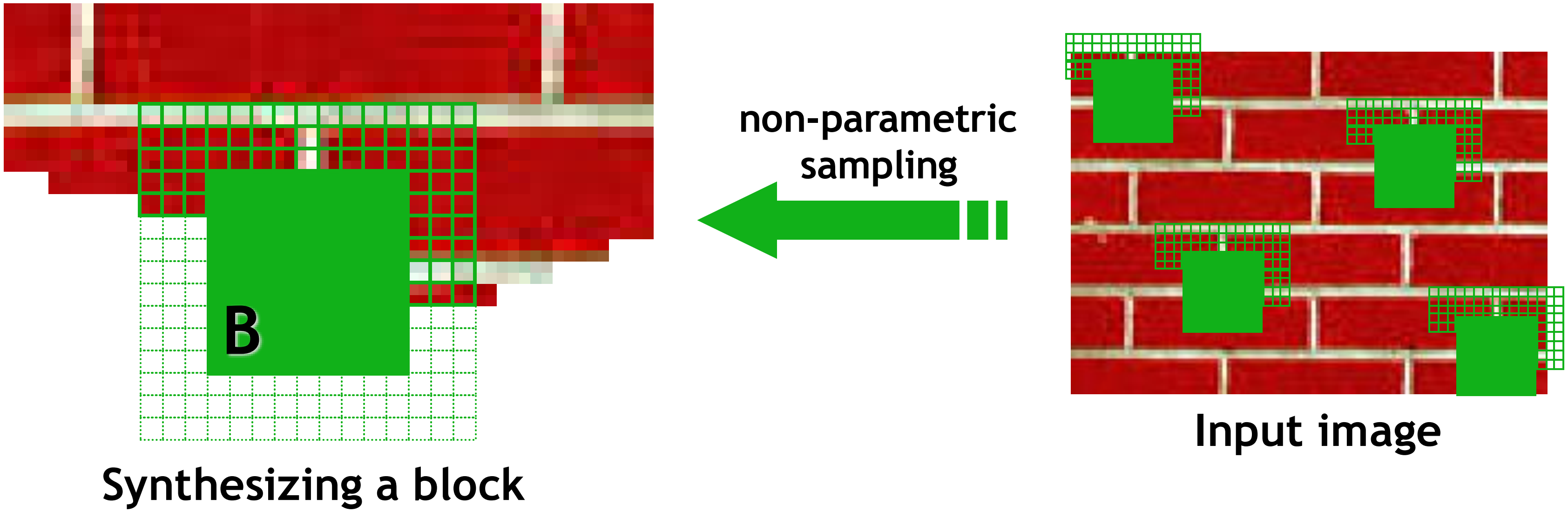
idea

result

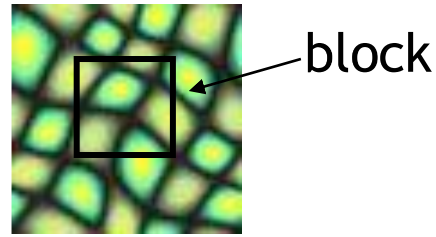
- Process: 1) tile input image; 2) pick random blocks and place them in random locations 3) Smooth edges

Used in Lapped Textures [Praun et.al,’ 00]

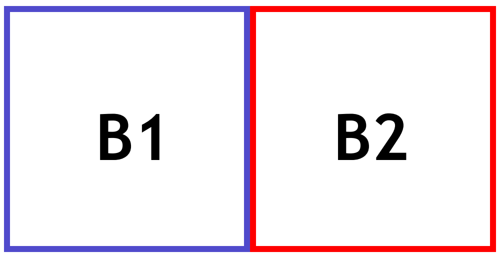
Image Quilting



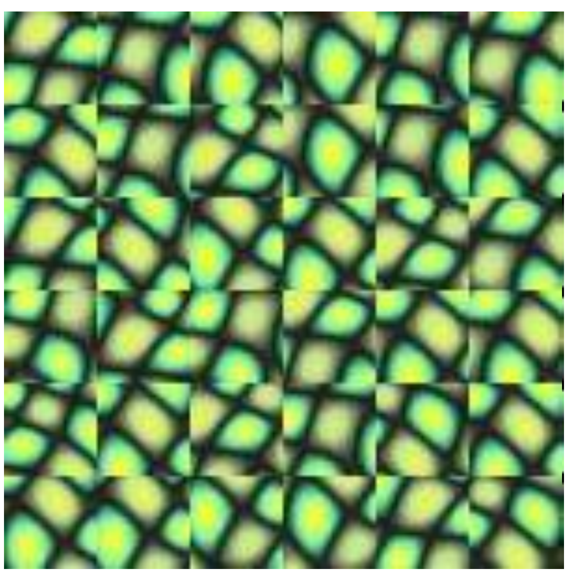
- **Idea**: let's combine random block placement of Chaos Mosaic with spatial constraints of Efros & Leung
- Unit of synthesis is a block
- Exactly the same but now we want P(**B**|N(**B**))
- Much faster: synthesize all pixels in a block at once

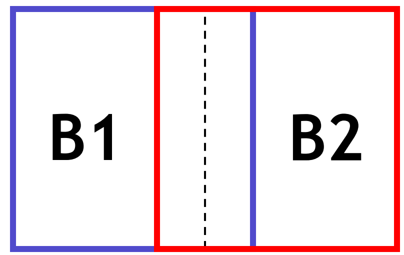


Input texture

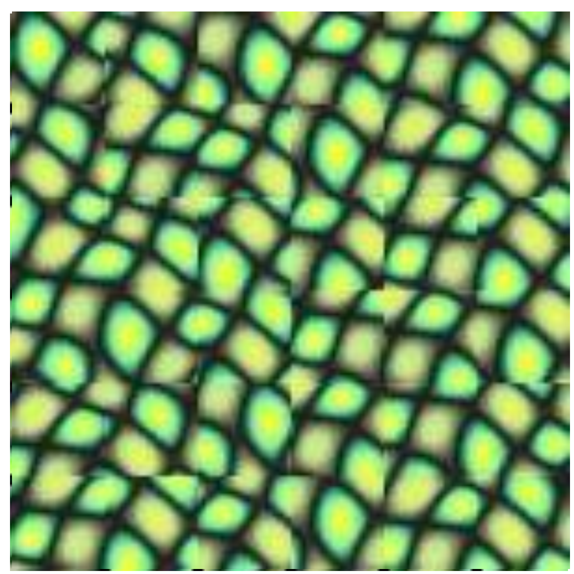


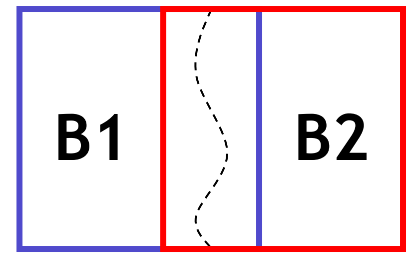
Random placement of blocks



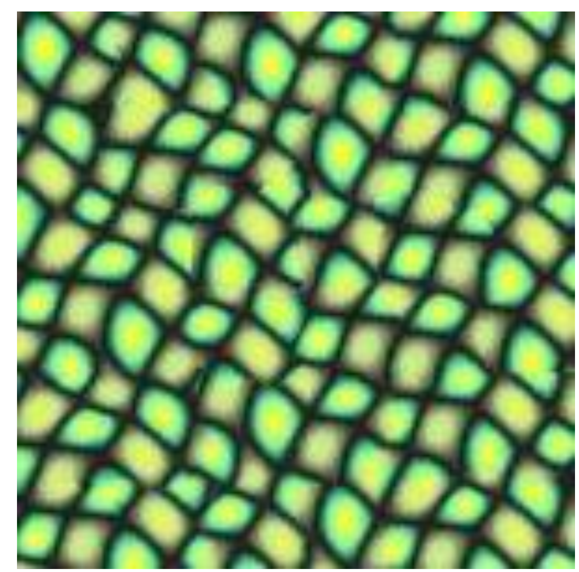


Neighboring blocks constrained by overlap

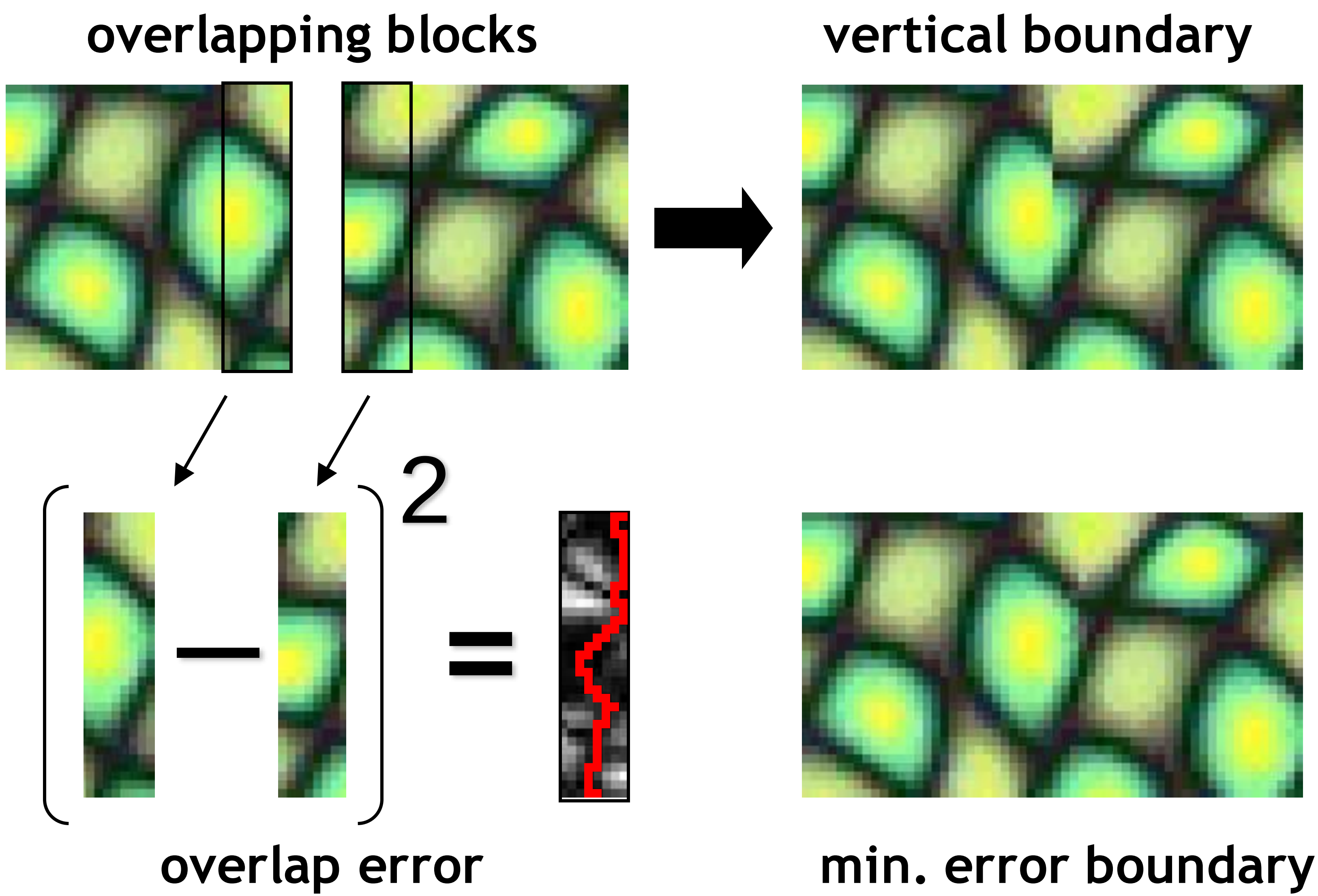




Minimal error boundary cut



Minimal error boundary

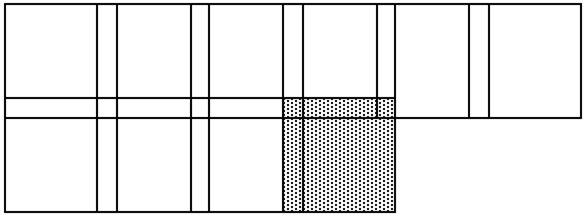


The Philosophy

- 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

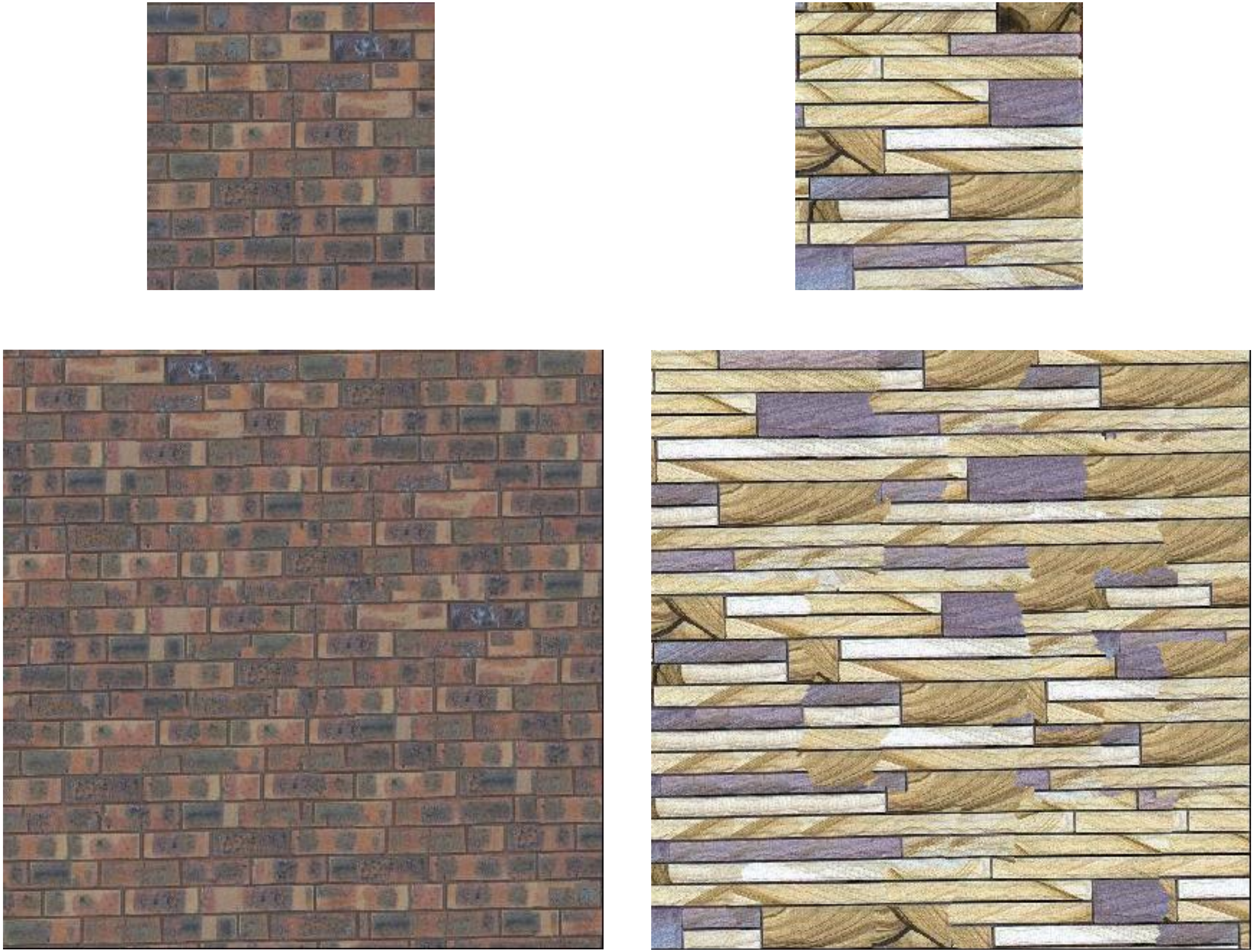
Algorithm

- 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)

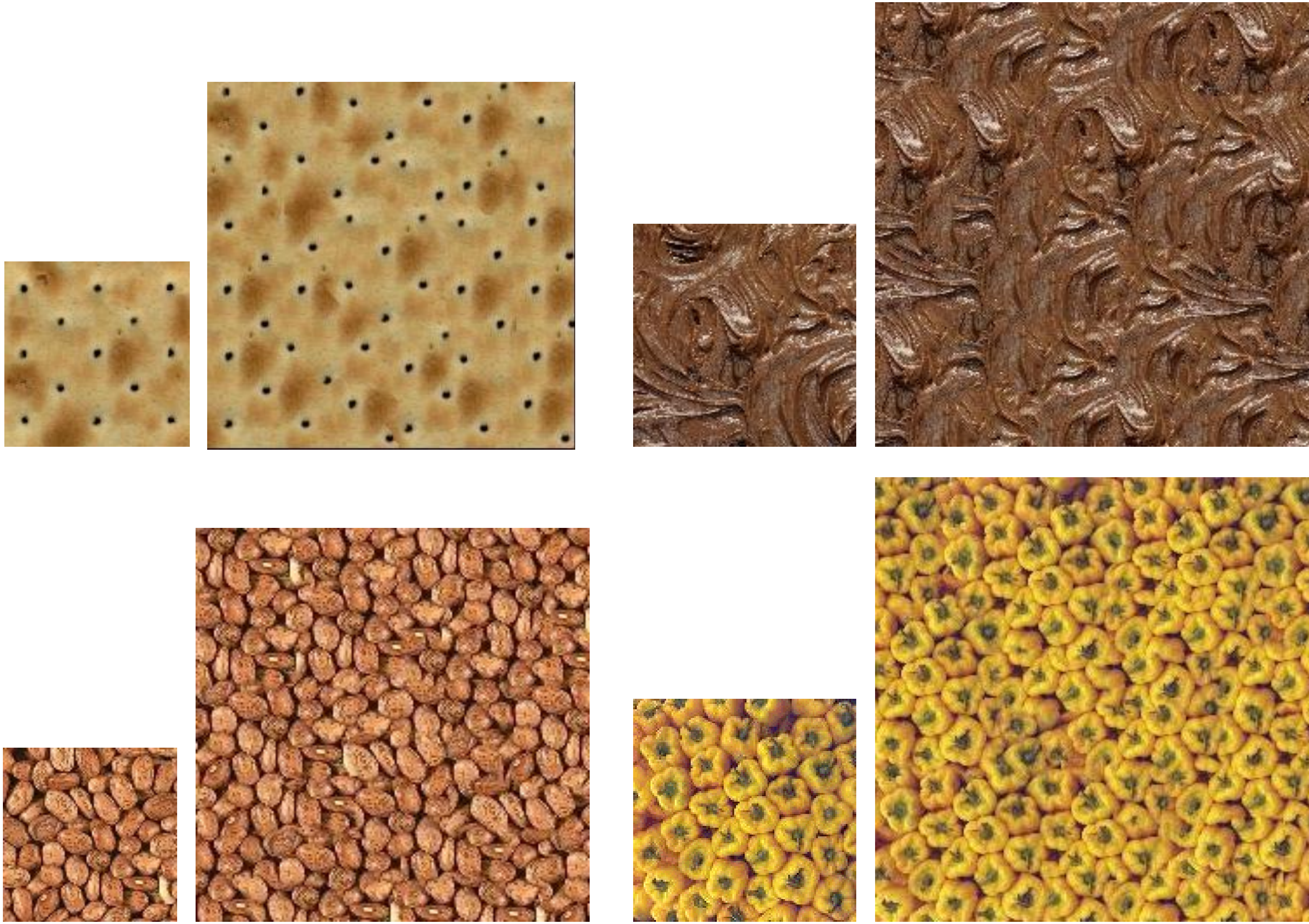
- Paste new block into resulting texture
  - use dynamic programming to compute minimal error boundary cut

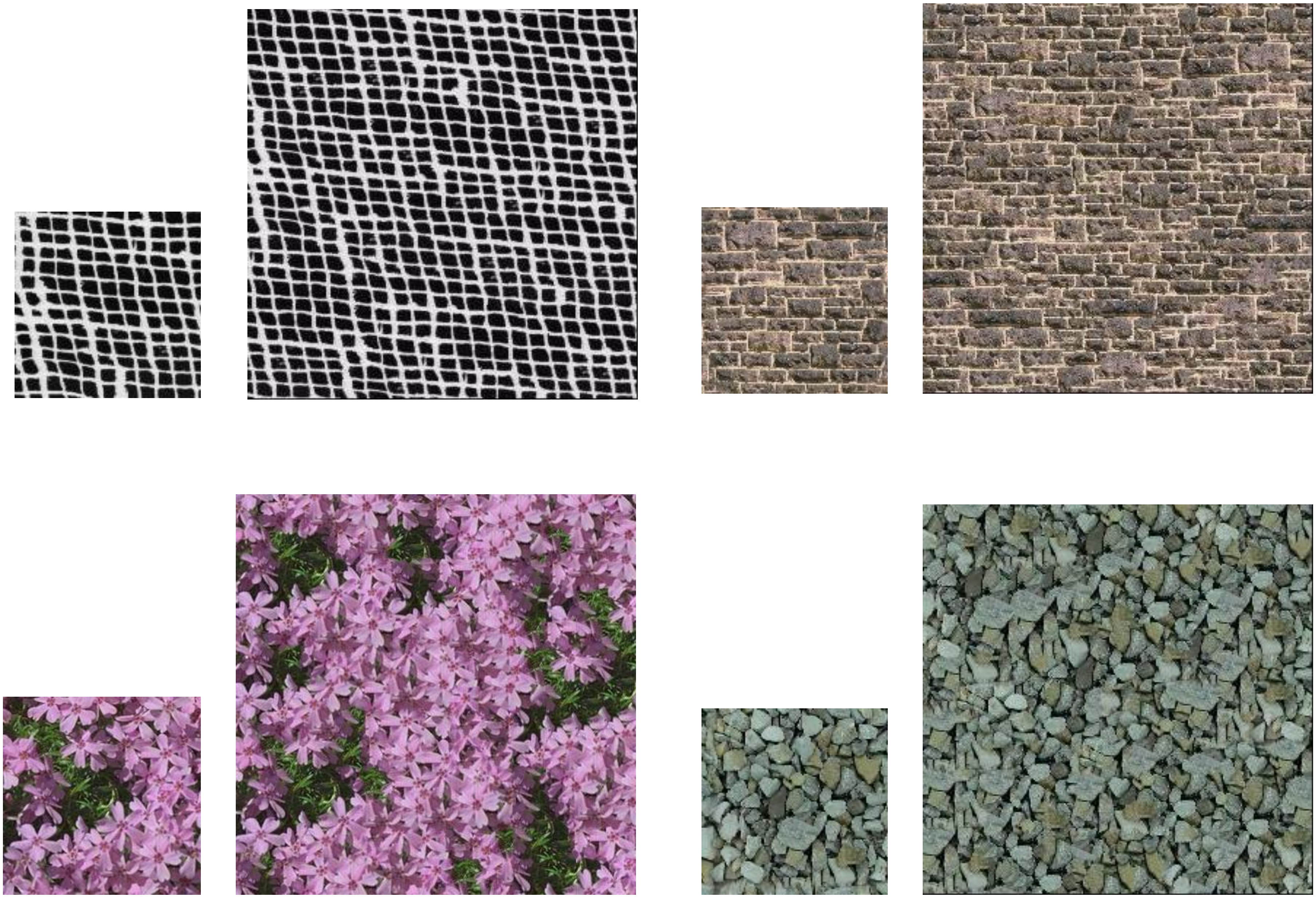



College Inn

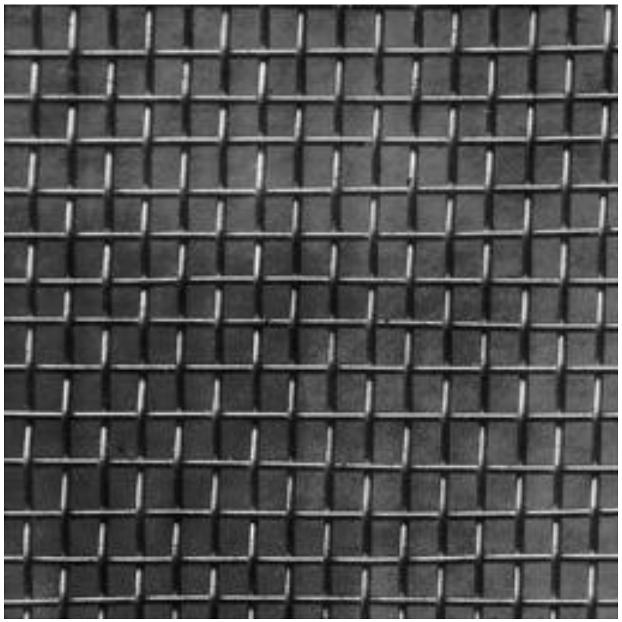

College Inn



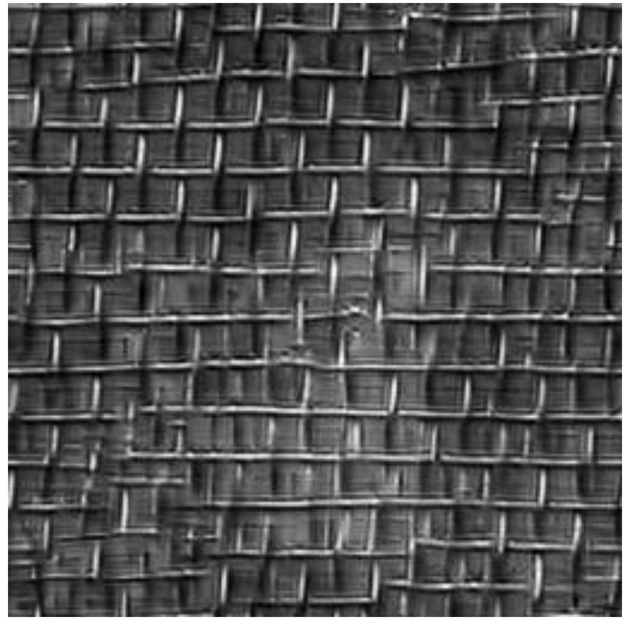





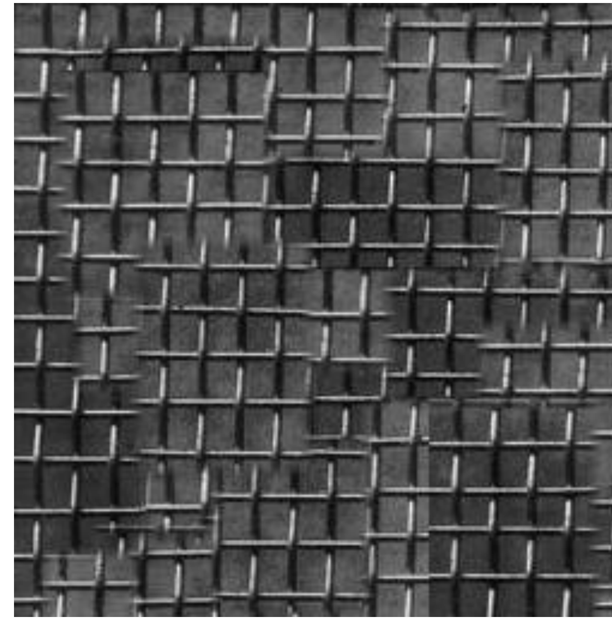
Comparison



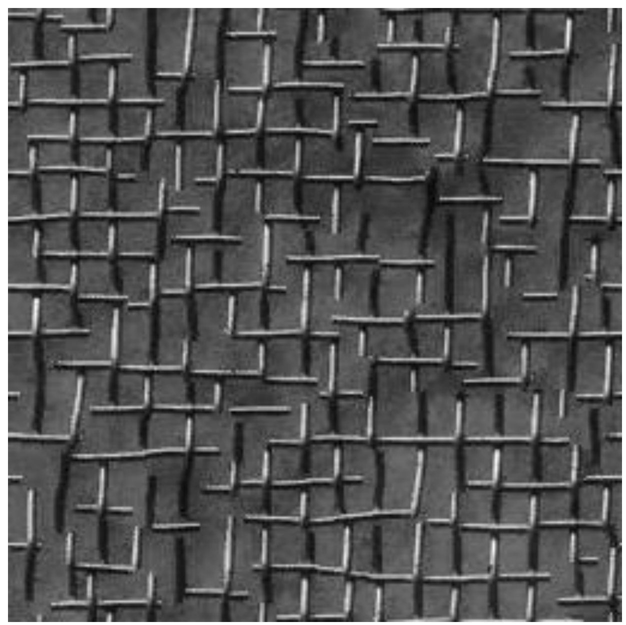
input image



Portilla & Simoncelli



Xu, Guo & Shum



Wei & Levoy

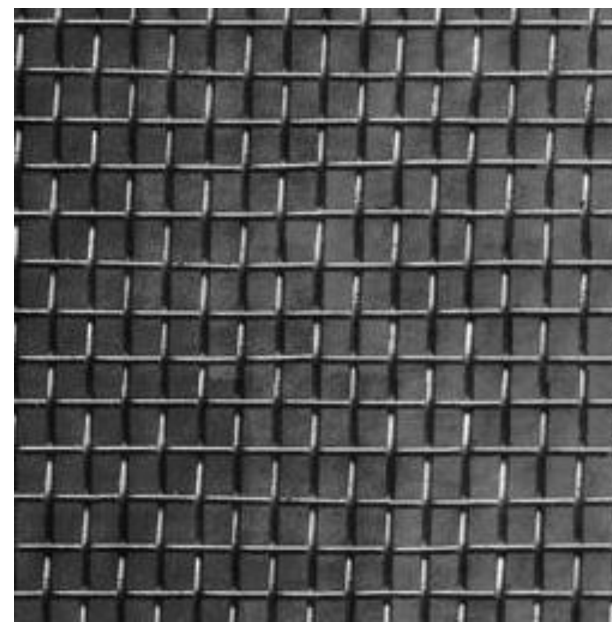
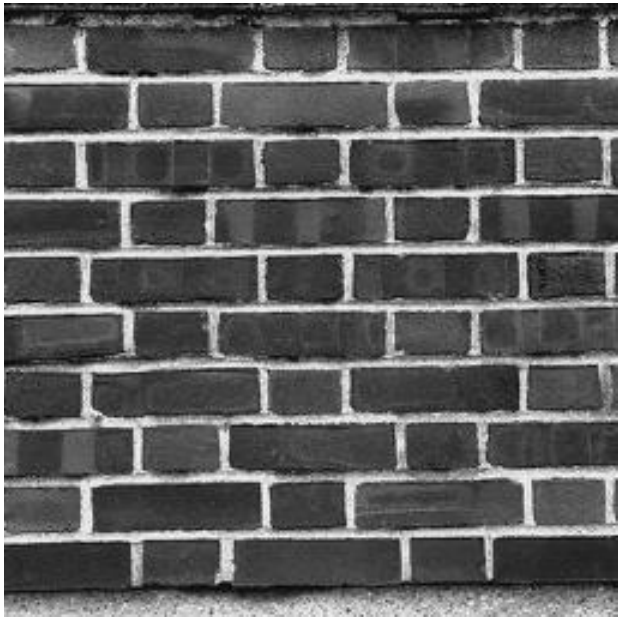


Image Quilting

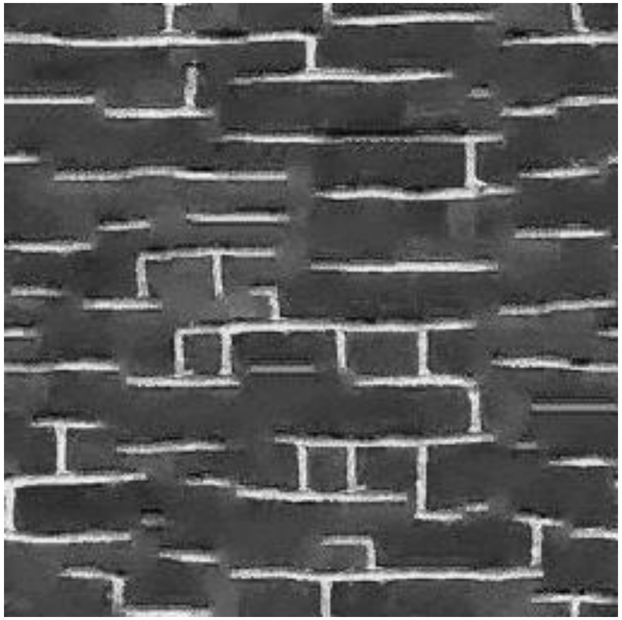
Comparison



input image

Portilla & Simoncelli

Xu, Guo & Shum



Wei & Levoy

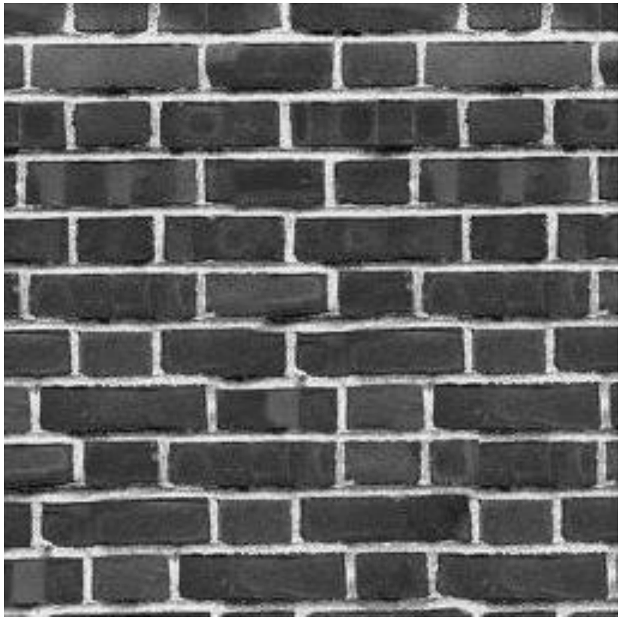
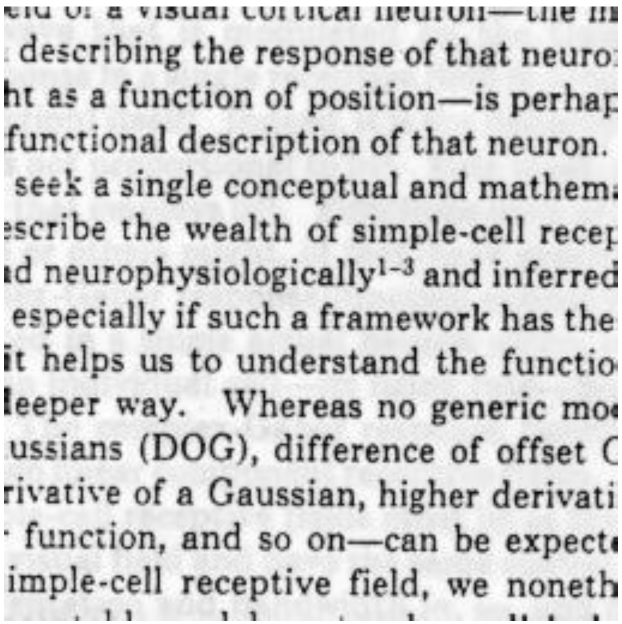
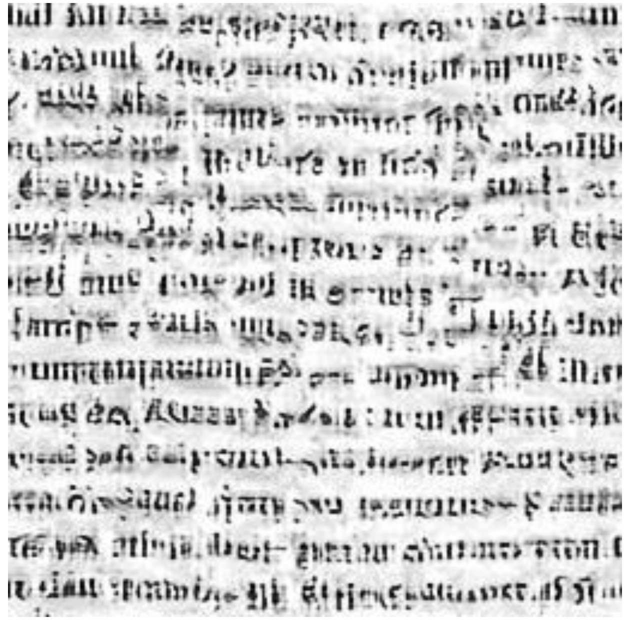


Image Quilting

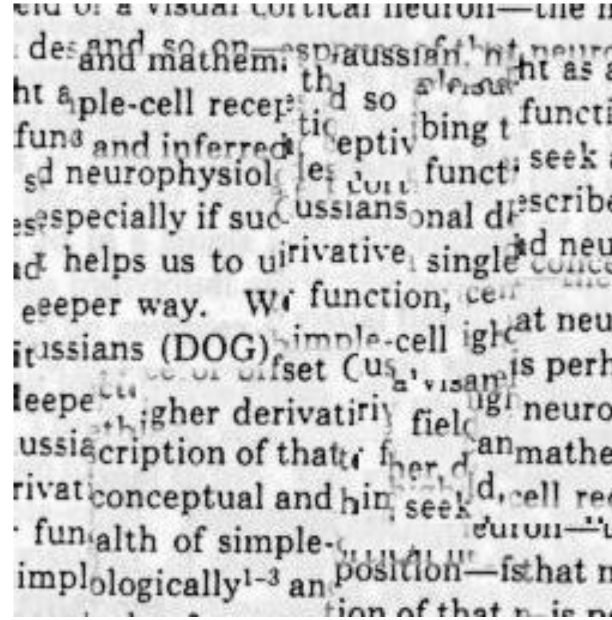
Homage to Shannon



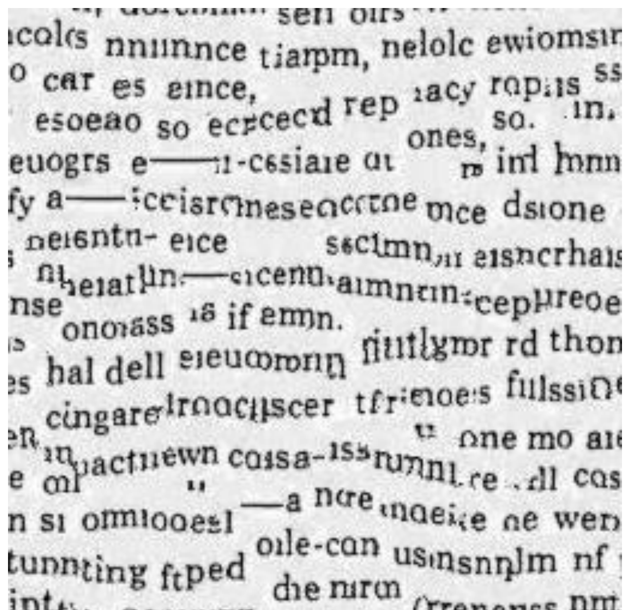
input image



Portilla & Simoncelli



Xu, Guo & Shum



Wei & Levoy

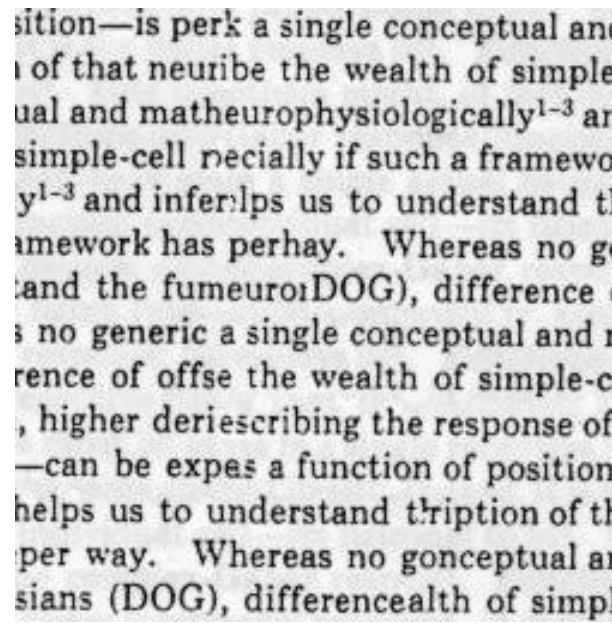


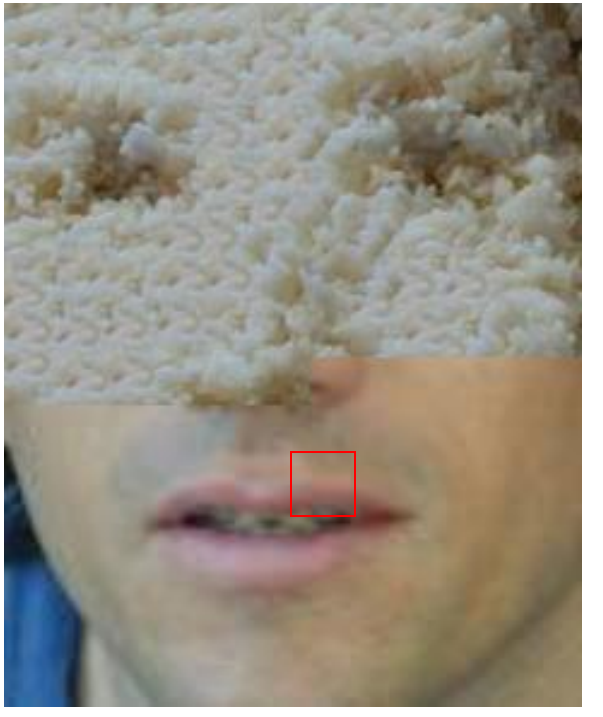
Image Quilting

Failures
(Chernobyl Harvest)

Texture transfer

Take the texture from one object and "paint" it onto another object

- This requires separating texture and shape
- That's HARD, but we can cheat
- Assume we can capture shape by boundary and rough shading

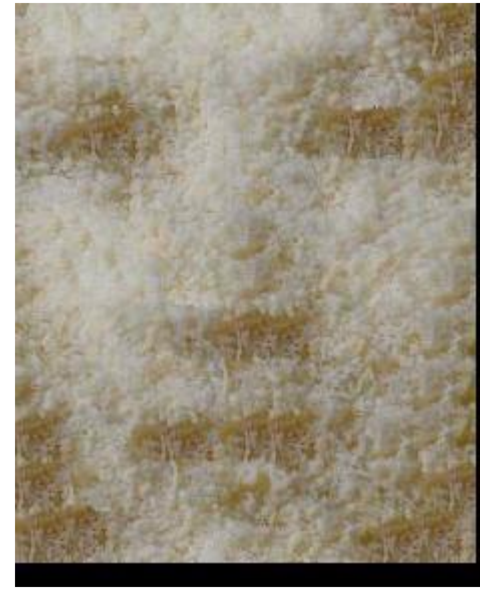


Idea: just add another constraint when sampling: similarity to underlying image at that spot

Correspondence can be based on: image intensity, blured image intensity, local image orientation angles, etc…

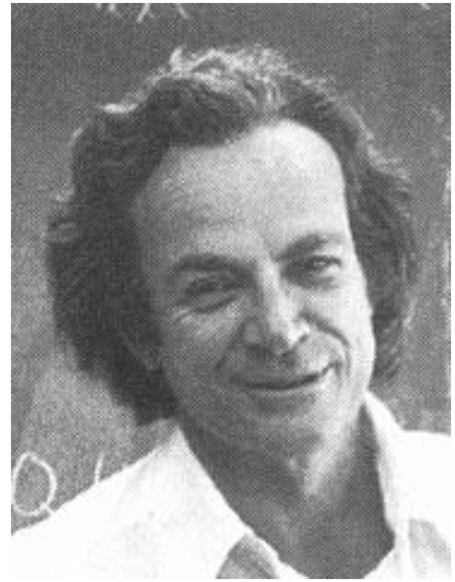
There is a **tradeoff** between the legitimacy of synthesized texture and the correctness of the correspondence mapping.

parmesan



rice

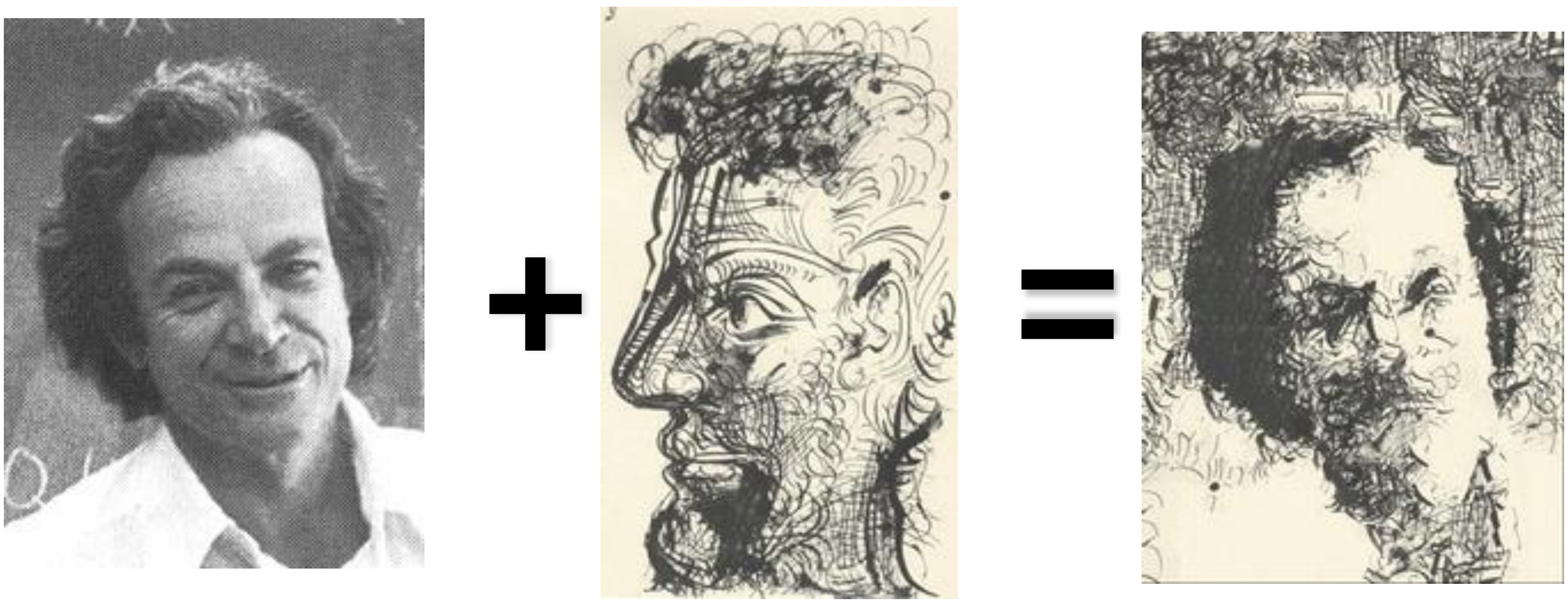
Source texture



Target image

Source correspondence image

Target correspondence image



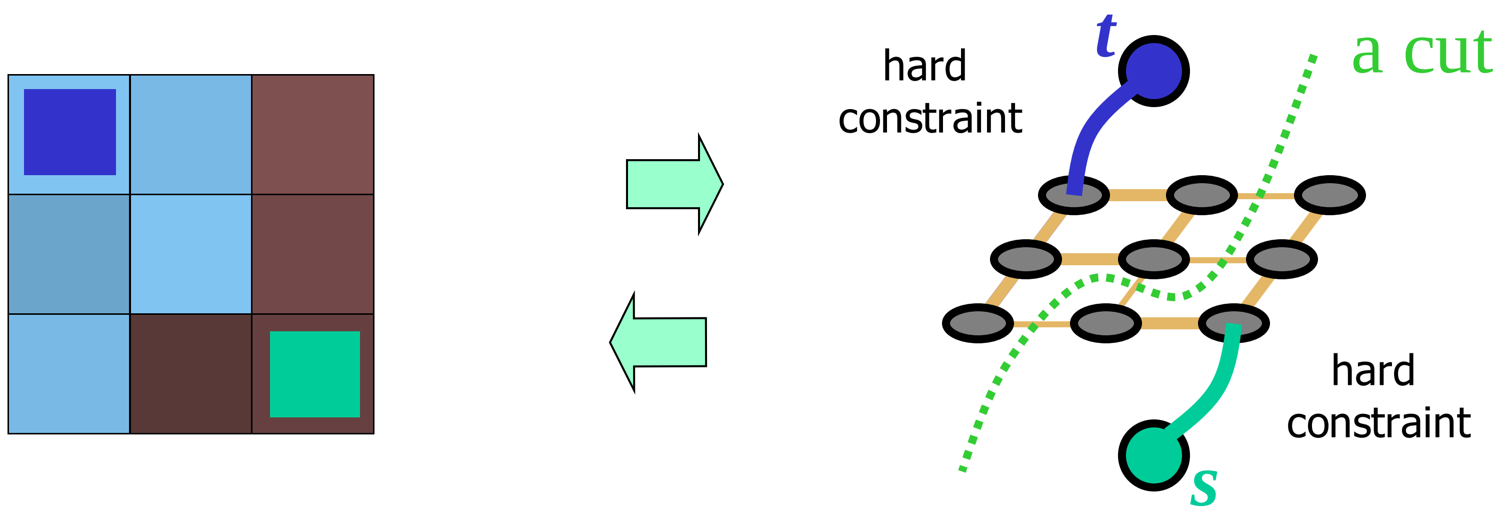
Applications of texture synthesis and transfer

- Occlusion fill-in
  - for 3D reconstruction
- region-based image and video compression
  - a small sample of textured region is stored
- Texturing non-developable objects
  - growing texture directly on surface
- Motion synthesis
- Synthesizing and transferring music and environmental sounds?
- Rendering object in a different style without explicit 3D information

Kwatra et al, 2003

Actually, for this example, DP will work just as well…

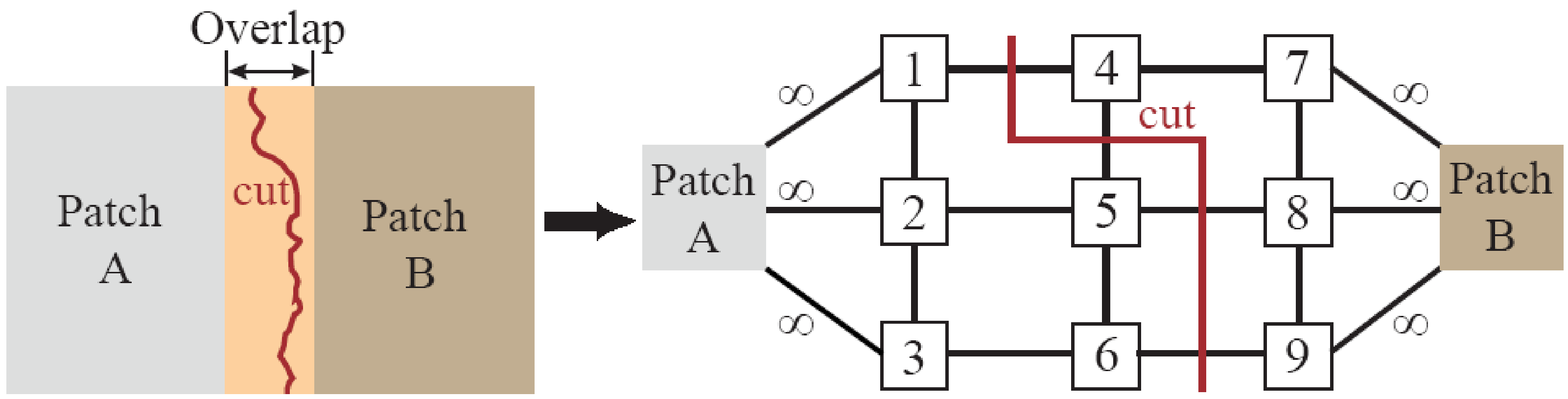
Graph cuts
(simple example à la Boykov&Jolly, ICCV' 01)



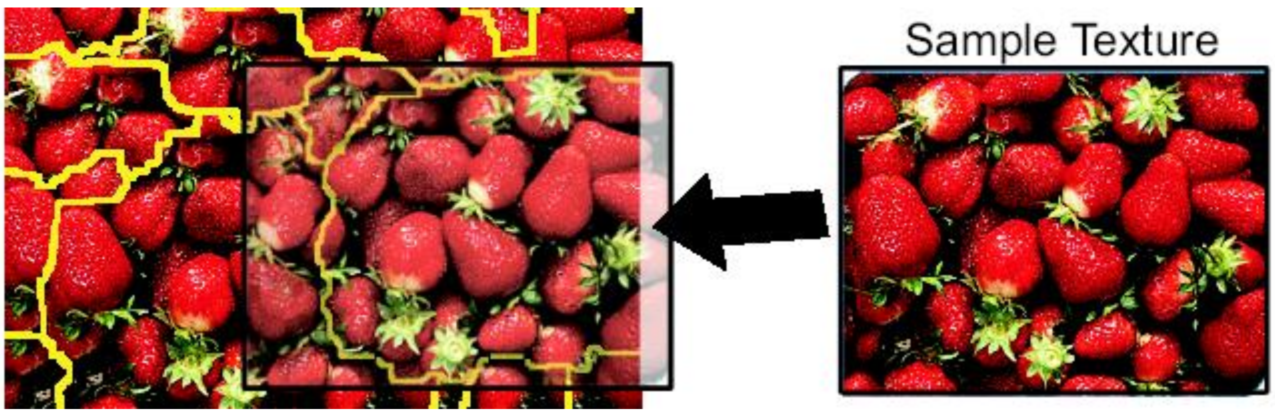
Minimum cost cut can be computed in polynomial time

(max-flow/min-cut algorithms)

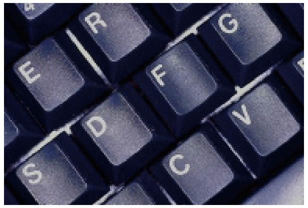
Kwatra et. al. 2003 - Algorithm



(assume cut region is 3x3 for simplicity)



Kwatra et. al. 2003 - Results



Lazy Snapping (Li el al., 2004)


(a) Girl (4/2/12) (b) Ballet (4/7/14) (c) Boy (6/2/13)
(c) Grandpa (4/2/11) (d) Twins (4/4/12)

Interactive segmentation using graphcuts