Two Patch-based Algorithms for By-example Texture Synthesis

Bruno Galerne

bruno.galerne@parisdescartes.fr

MAP5, Université Paris Descartes

Master MVA

Cours “Méthodes stochastiques pour l’analyse d’images”

Lundi 27 février 2017

Slide credits:

Alyosha Efros, Bill Freeman (SIGGRAPH presentation)

Rob Fergus (NYU course)

Textures

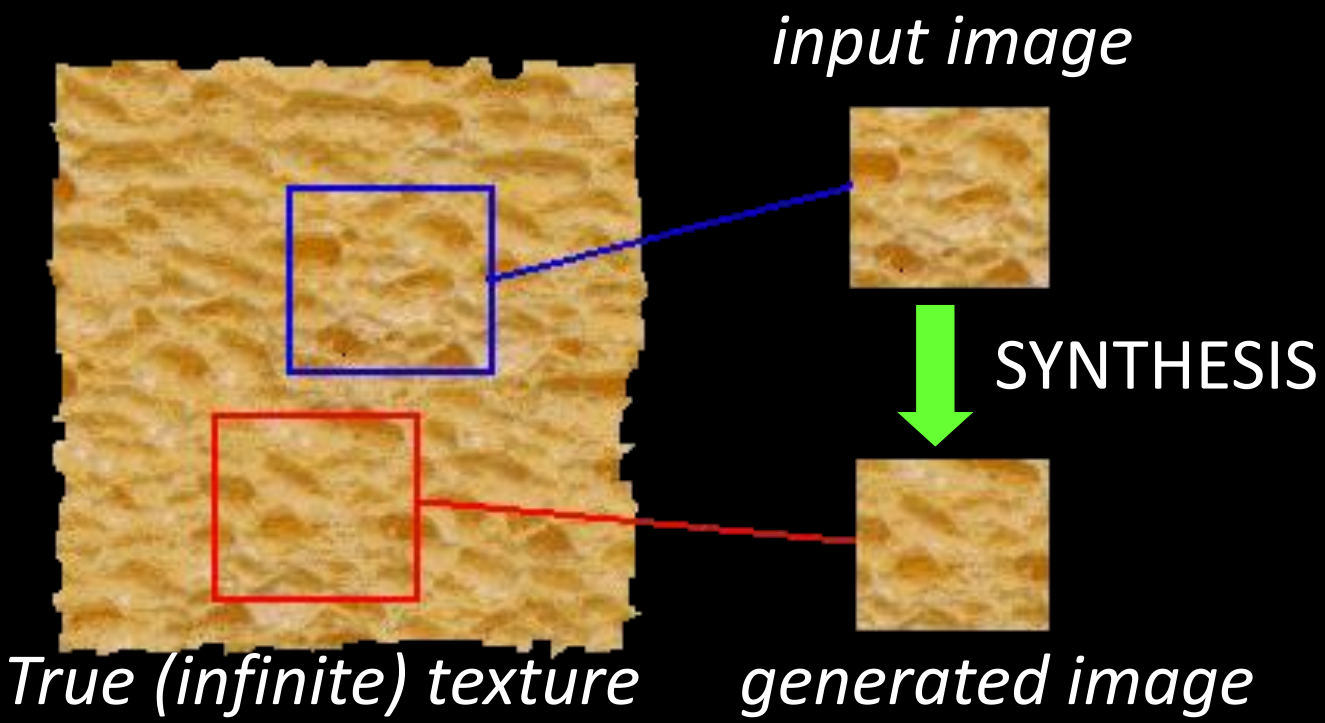
- Texture depicts spatially repeating patterns
- Many natural phenomena are textures

radishes

rocks

yogurt

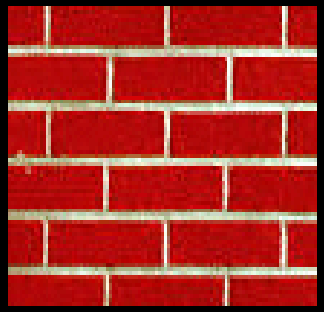
The Goal of Texture Synthesis

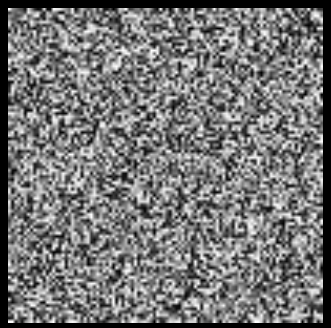


- Given a finite sample of some texture, the goal is to synthesize other samples from that same texture

Challenge

- Need to model the whole spectrum: from repeated to stochastic texture
- Micro-textures and macro-textures





Outline

- The “Alexei Efros course”
  - Texture Synthesis by Non-parametric Sampling [Efros & Leung, ICCV’99]
  - Image Quilting for Texture Synthesis & Transfer [Efros & Freeman, SIGGRAPH 2001]
- TP
  - Test Efros-Leung with IPOL demo
  - Implement Image Quilting in Matlab! (with some help)

Improvements…

- Of course many more contributions and improvements since 1999
  - Multi-scale approach [Wei & Levoy, 2000]
  - … with global optimization instead of sequential synthesis [Kwatra et al. 2003, Kwatra et al. 2005]
  - … with parallel evaluation [Lefebvre & Hoppe, 2005]

Importance of Efros & Leung ‘99

- First paper with patch-based algorithm
- Concept of redundancy of patches
- Strong influence for many applications
  - Inpainting
  - Image editing (Image analogies, PatchMatch)
  - Denoising (Non-Local Means algorithm)

Texture Synthesis by Non-parametric Sampling

[Efros & Leung, ICCV'99]

IPOL demo [Aguerrebere, Gousseau, Tartavel, 2013]

Shannon

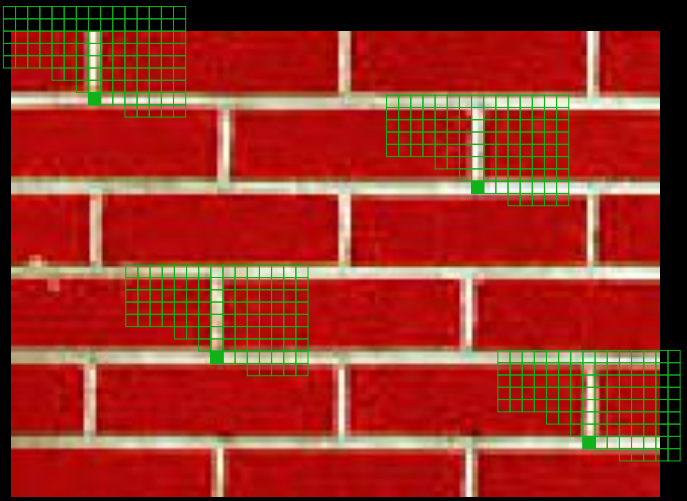
Shannon's approach for text synthesis

- Markov model
- Draw an N-gram with marginal distribution
- Add characters sequentially using the conditional distribution of N-gram given the first N-1 characters
- Using 4-grams:

THE GENERATED JOB PROVIDUAL BETTER
TRAND THE DISPLAYED COVE ABOVERY

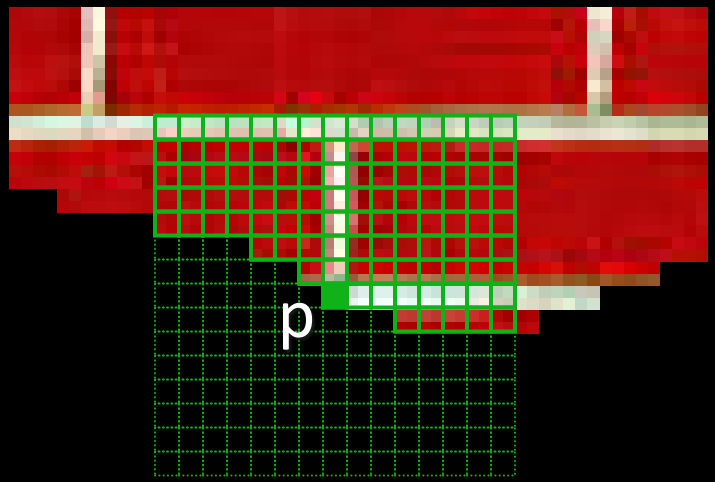
Shannon for Texture Synthesis?

- Differences between textures and texts for applying Shannon's approach
  - No natural order in a 2D pixel grid
  - Pixel similarity: two close pixel values can exchanged without altering the texture while it is not possible for udys (= text +/- 1)
  - Compute the conditional probability is not feasible
- Still Markov model is OK!



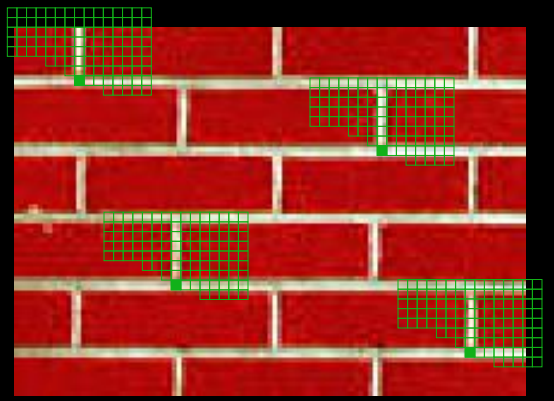
General Idea

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



Synthesizing a pixel

non-parametric sampling



Input image

Algorithmic Details

- Start from a 3x3 seed from the input
- The new pixel **p** to fill is randomly picked among all the ones that have the larger number of neighborhood

-> synthesis by layer

Neighborhood Comparison

- Given a partial patch *N*(p) in the ouput **B** at pixel p, we compute the distance to all partial patches p' in input image **A**

$$d(\mathcal{N}(p),\mathcal{N}(p')) = \frac{1}{\sum_{i\in\mathcal{N}(p)} G_\sigma(i)} \sum_{i\in\mathcal{N}(p)} (\mathbf{A}(p'+i) - \mathbf{B}(p+i))^2 G_\sigma(i)$$

- Squared difference with Gaussian weights since the center of the patch has more importance

Similar Neighborhood

- The set of “similar neighborhoods” are defined as all patches *N*(p’) *such that*

$$d(\mathcal{N}(p), \mathcal{N}(p')) \leq (1+\varepsilon) \min_{p'} d(\mathcal{N}(p), \mathcal{N}(p'))$$

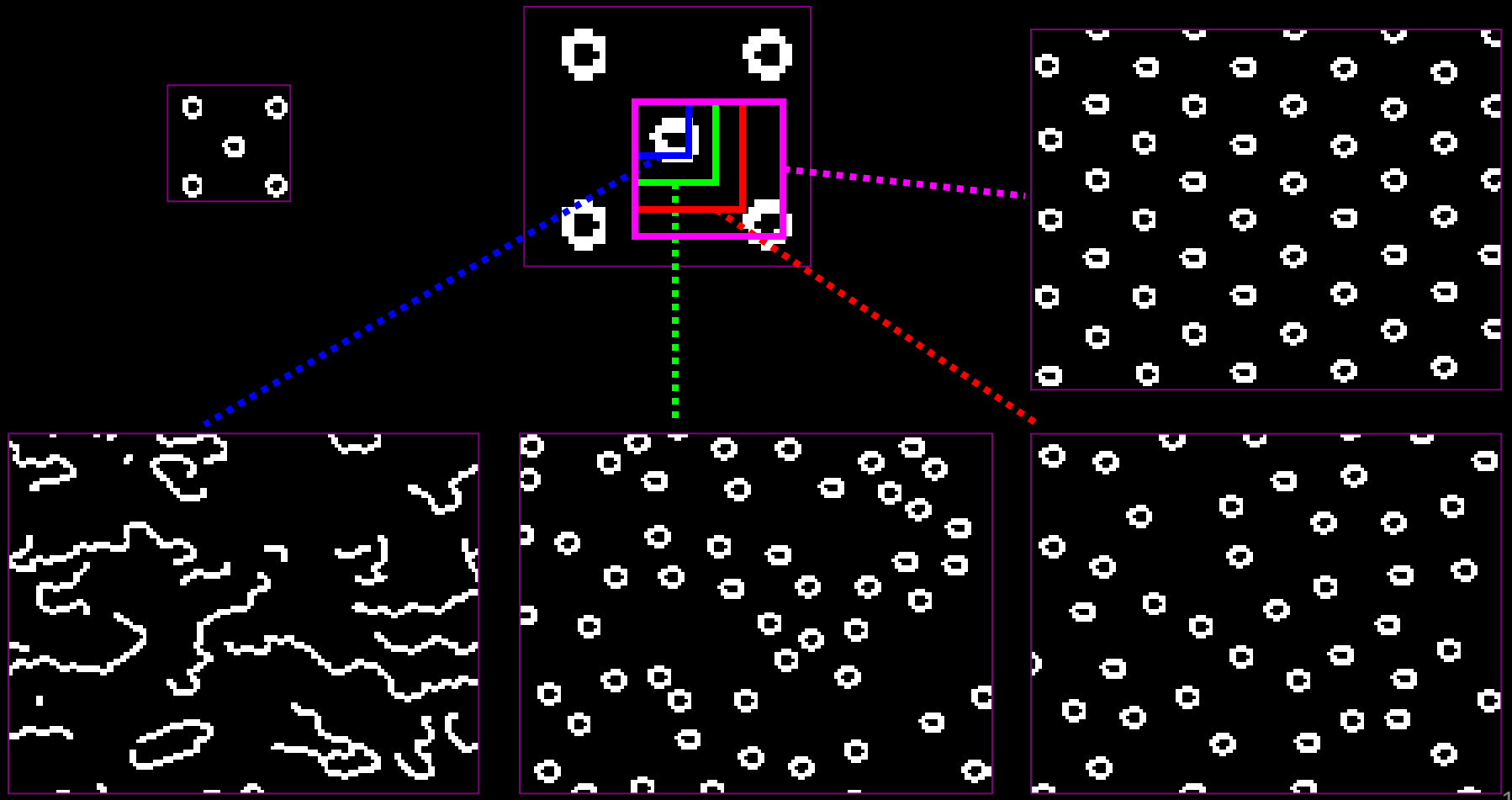
- $\varepsilon > 0$ is a global parameter

Pseudo-code of the Efros-Leung Algorithm

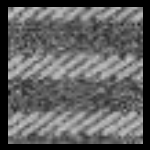
- **Input arguments:** Input image **A**, output size
- **Parameters :** neighborhood size, ε

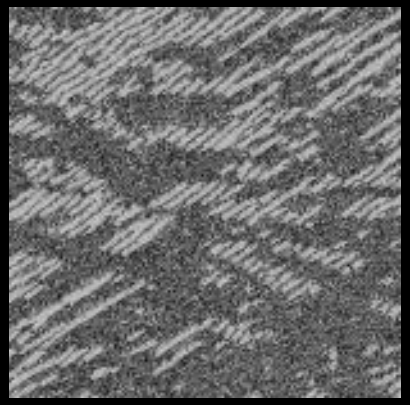
1. Initialize with 3x3 random seed of **A**
2. While output **B** not filled
   1. Pick a pixel p in **B** with maximal neighbor pixels
   2. Compute the distance of *N*(p) to all patches of input **A**
   3. Pick randomly one the similar neighborhood p'
   4. Set **B**(p) = **A**(p') (= Fill p with central value)

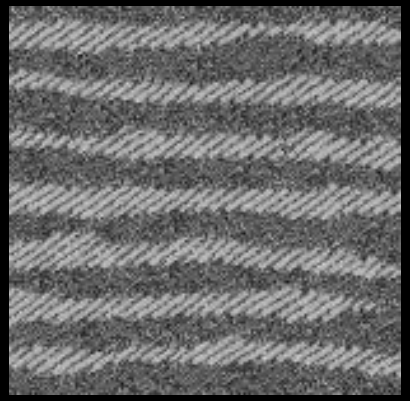
Neighborhood Size

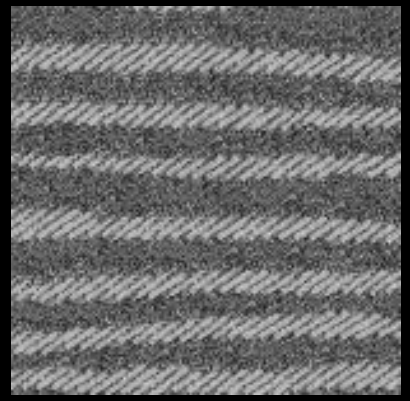


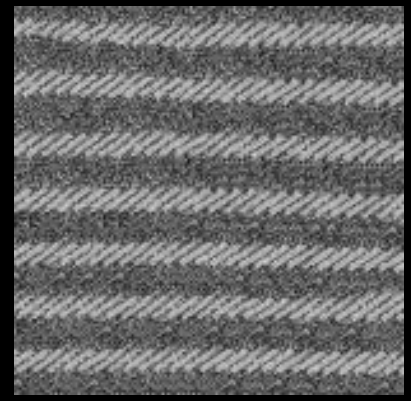
Varying Neighborhood Size

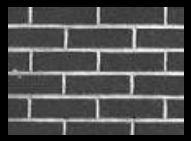


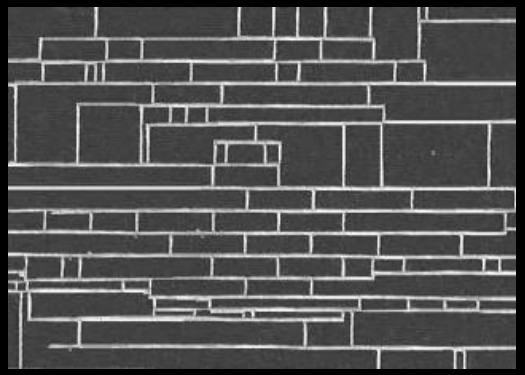


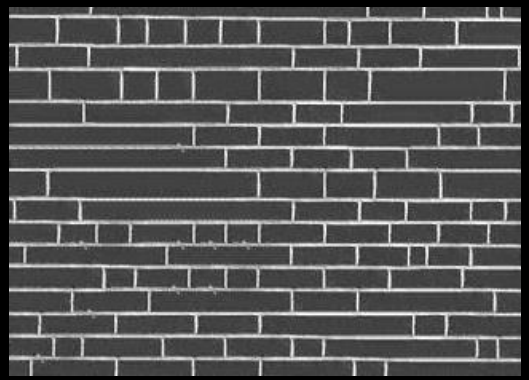


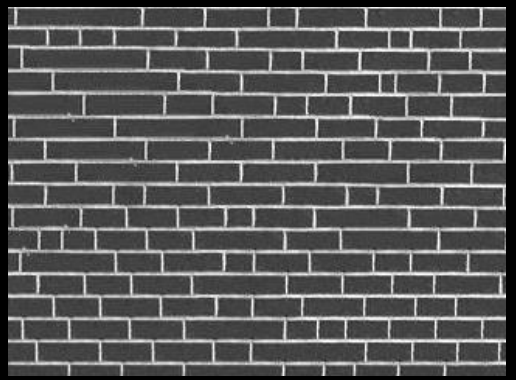




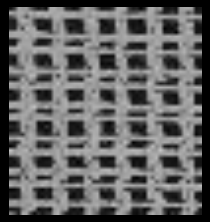


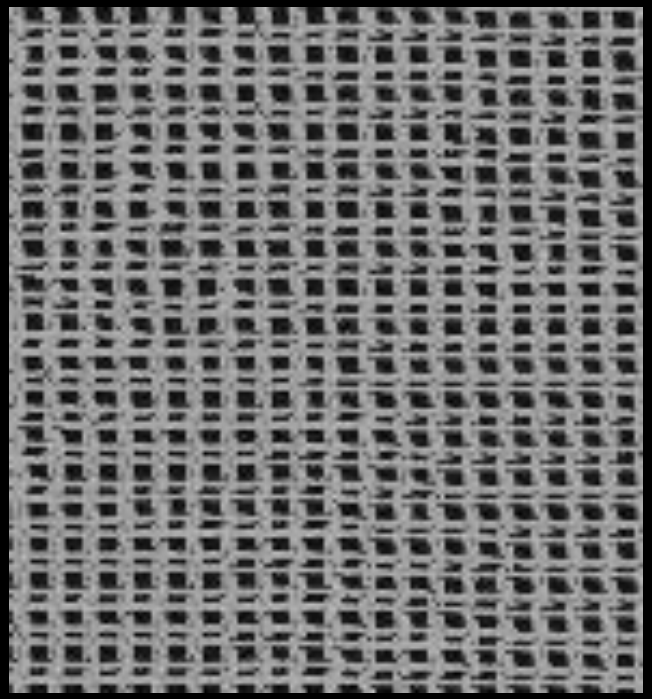


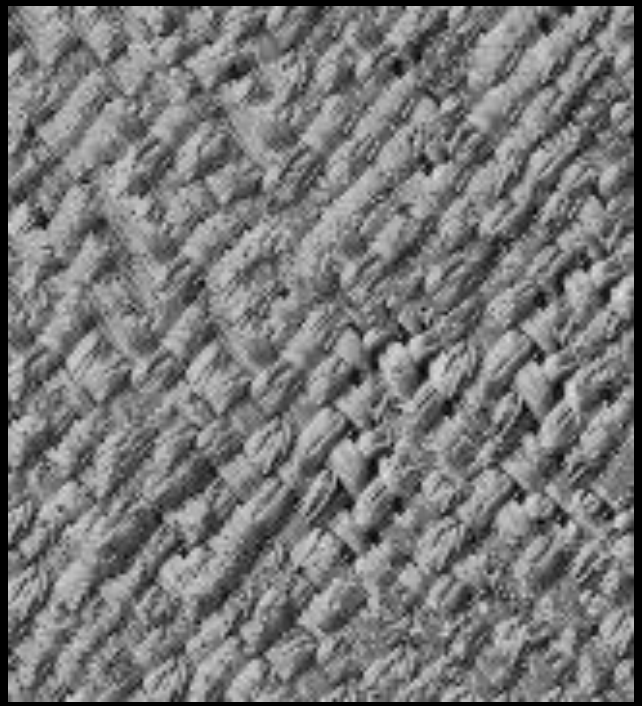




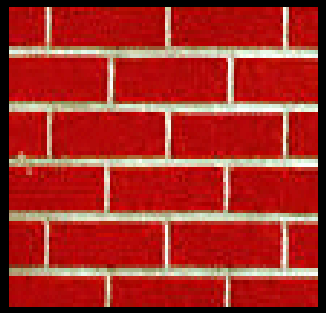
Synthesis Results

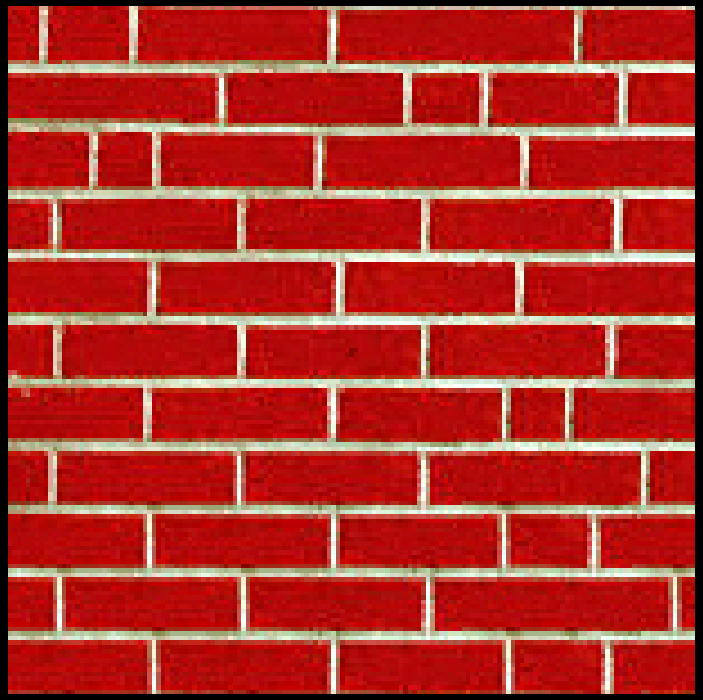




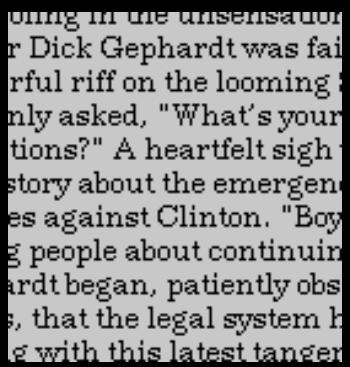


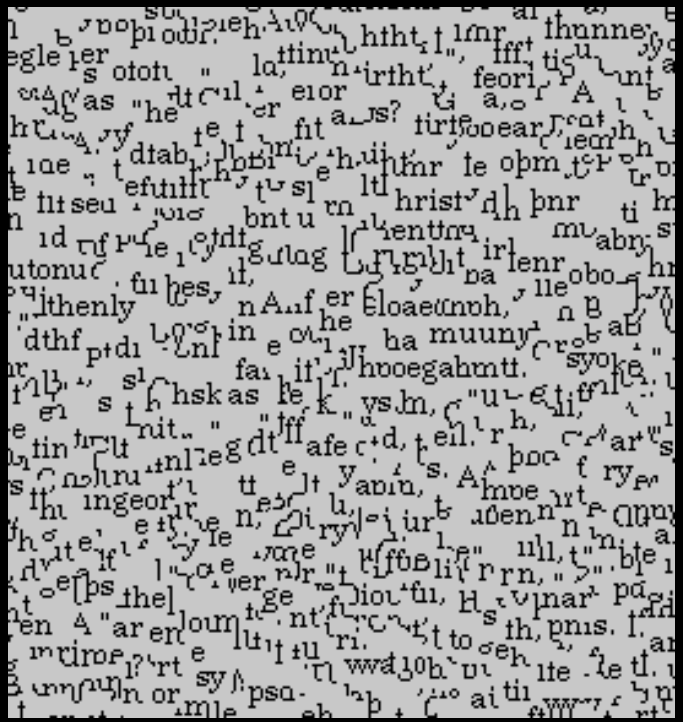
Synthesis Results

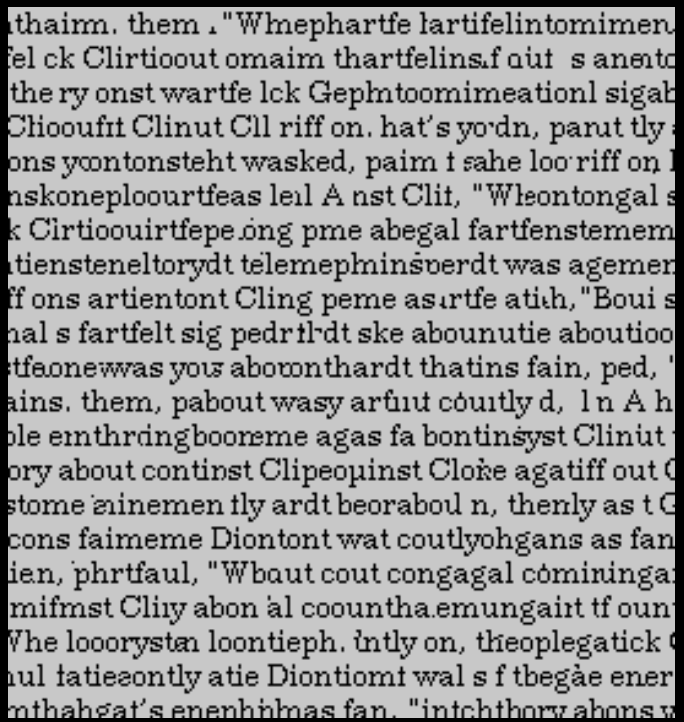




Homage to Shannon







Not only for texture synthesis

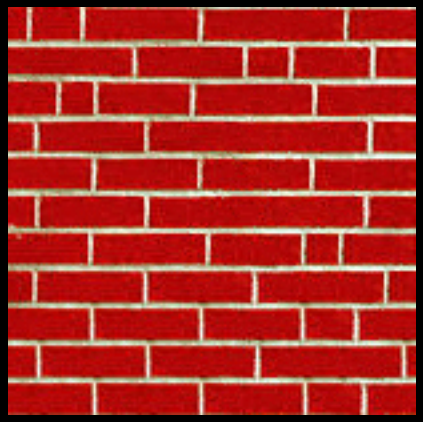
Other applications

- Texture inpainting (or hole filing)
- Texture/Image extrapolation

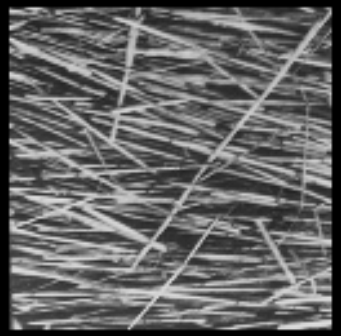
Application: Texture Inpainting

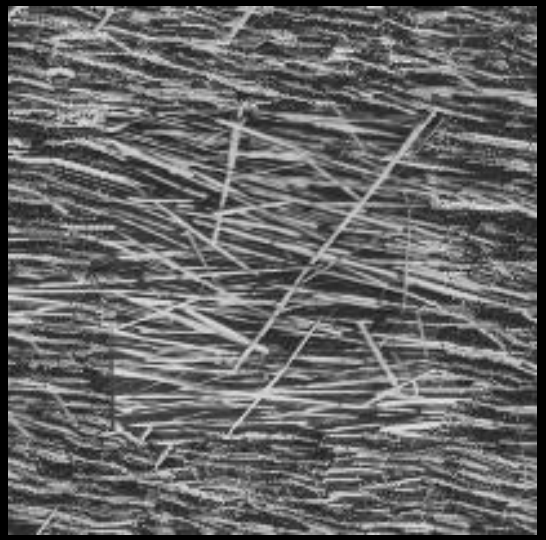




Application: Image Extrapolation

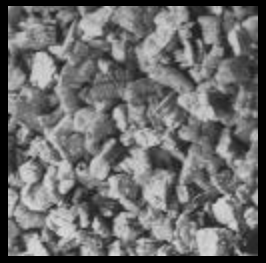


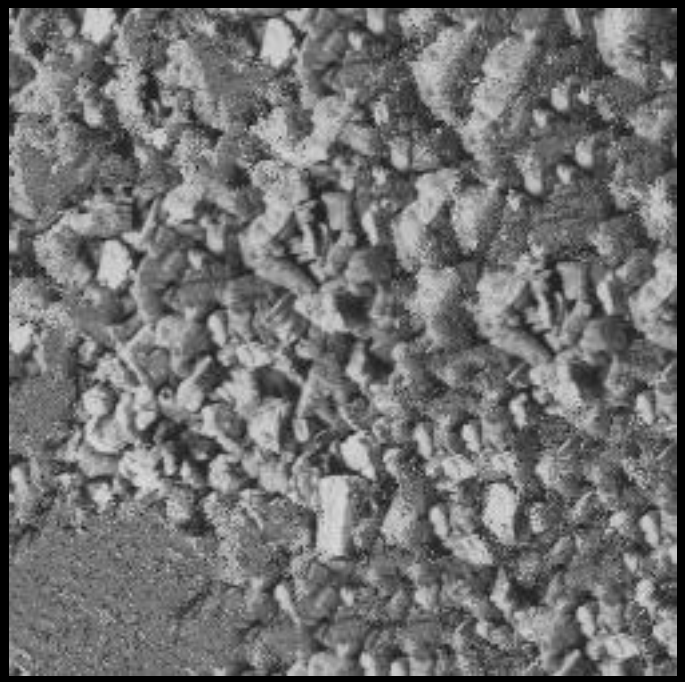


Back to Texture Synthesis

- Surprisingly good results for structured textures!
- **Failure cases**
  1. ***Verbatim copy***
  2. ***Growing garbage***

Failure Cases





Growing garbage

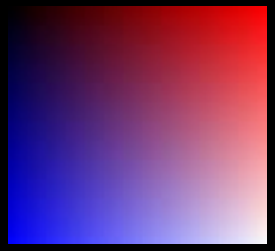
Verbatim copying

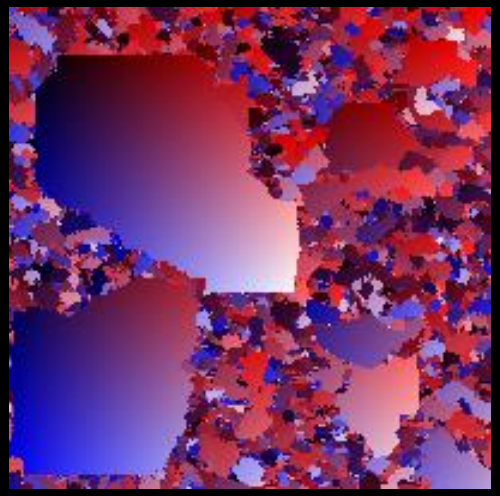
Pros & Cons

- Advantages
  - Conceptually simple
  - Satisfying for a wide range of real-world textures
  - Naturally does hole-filling
- Disadvantages
  - Slow
  - Parameters are hard to set: thin space (if any) between **verbatim copy** and **growing garbage**
  - No ensured quality: trial and error

Verbatim Copy

- To get a better understanding, one can plot the pixel coordinate map



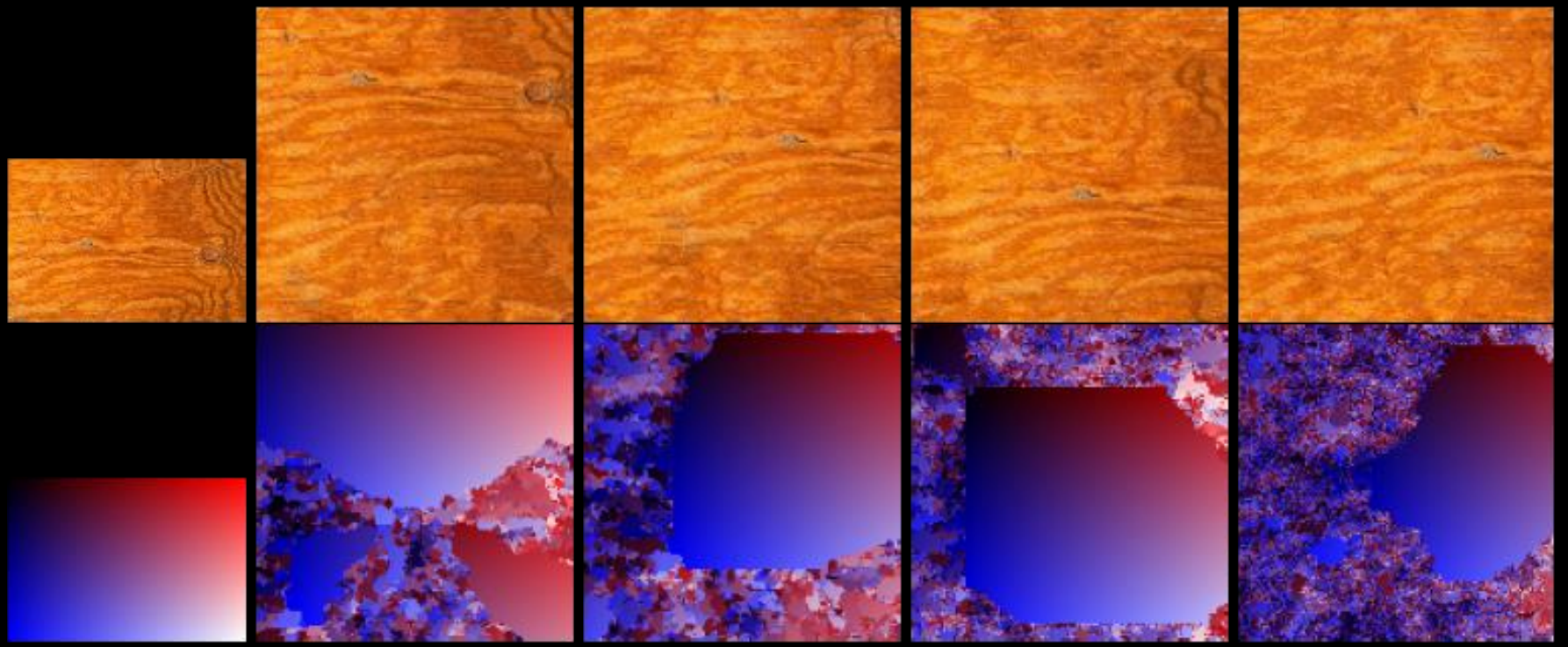
- Although pixels are generated one pixel at a time, whole pieces of input are reproduced

Verbatim Copy

- The problem comes from the fact that when dmin = 0 reproducing the input is generally the only possibility

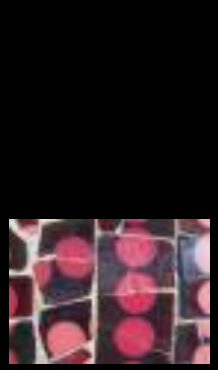
Innovation Capacity

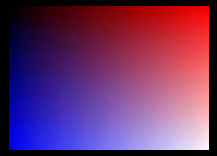
- Innovation capacity increases with ε

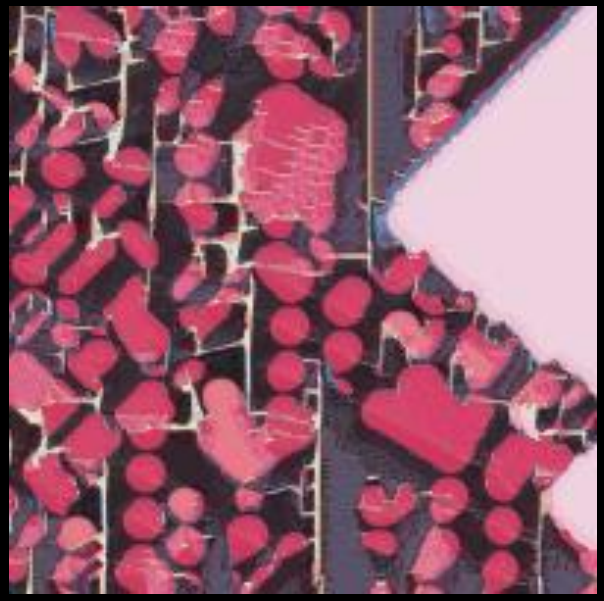


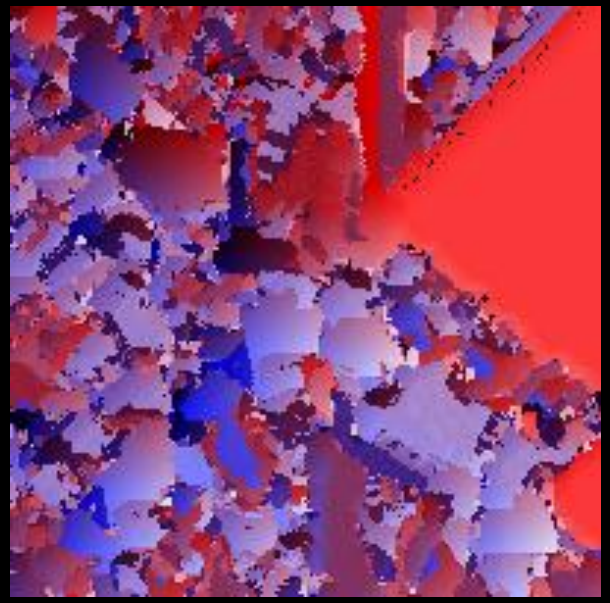
Growing Garbage

- Happens when the algorithm is stuck in some local loop



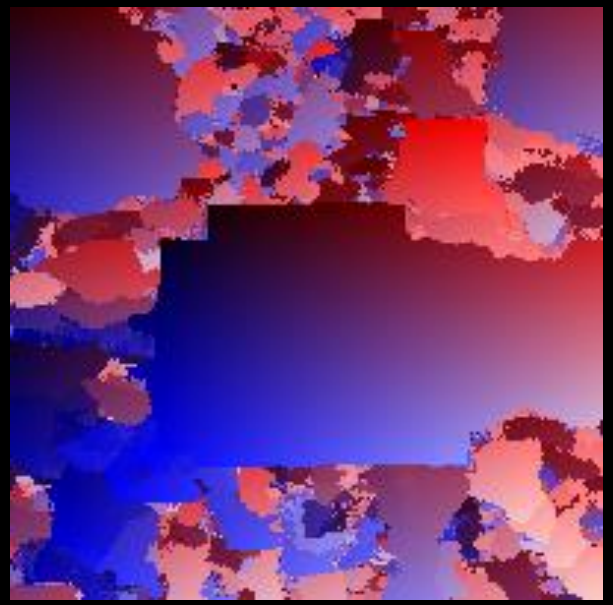






Avoid Failure

- For some textures it is hard to find a balance between verbatim copy and growing garbage



Acceleration

- The computational cost comes from the comparison to all patches of the input textures at each step
- The neighborhoods have different shapes so it is hard to use some data structure to accelerate the computation of the distance
- Squared sum computation
  - Partial sum of distance: discard as soon as larger than threshold
  - Works even better with PCA coordinates (more variance for first coordinates) (of half-patches)

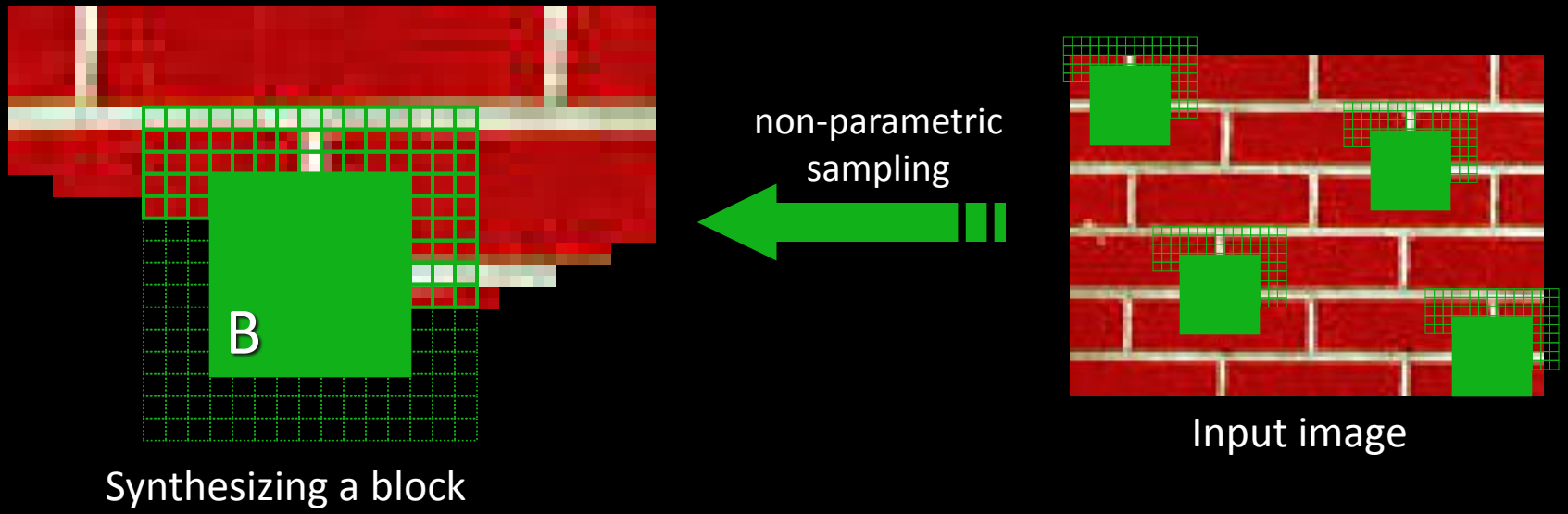
Questions?

Image Quilting for Texture Synthesis & Transfer

[Efros & Freeman, SIGGRAPH 2001]

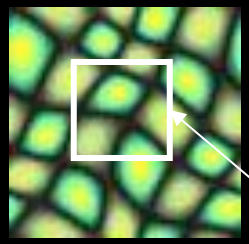
IPOL demo: [Raad, Galerne, 2017]

Efros & Leung ’99 extended

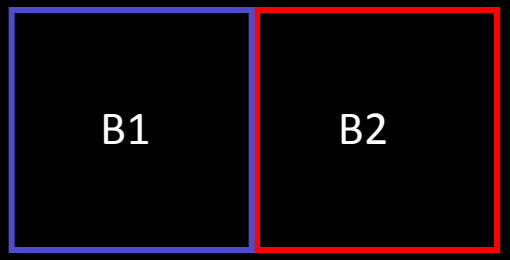


- Observation: neighbor pixels are highly correlated and copied one pixel at a time
- Idea: unit of synthesis = block
  - Exactly the same but now we want P(B|*N*(B))
  - Much faster: synthesize all pixels in a block at once

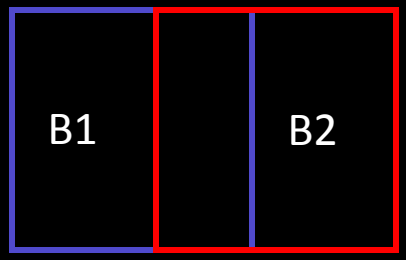
Quilting: Main Idea



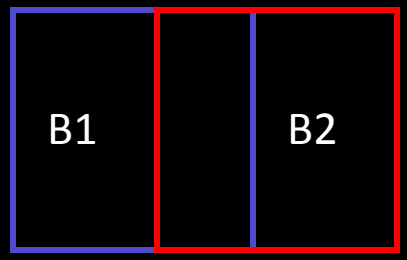
Input texture



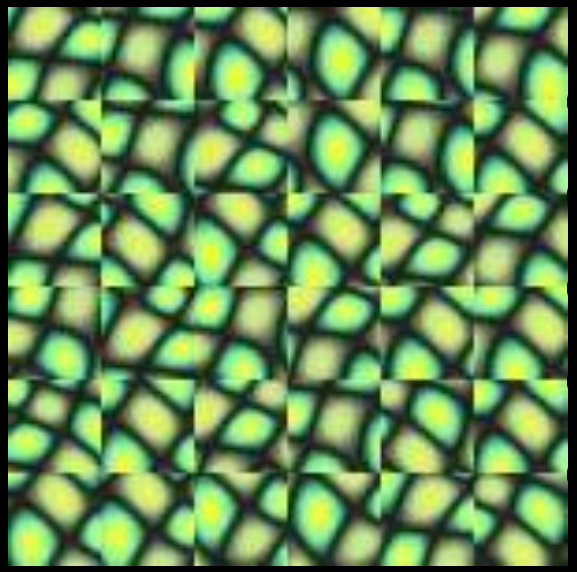
Random placement of blocks

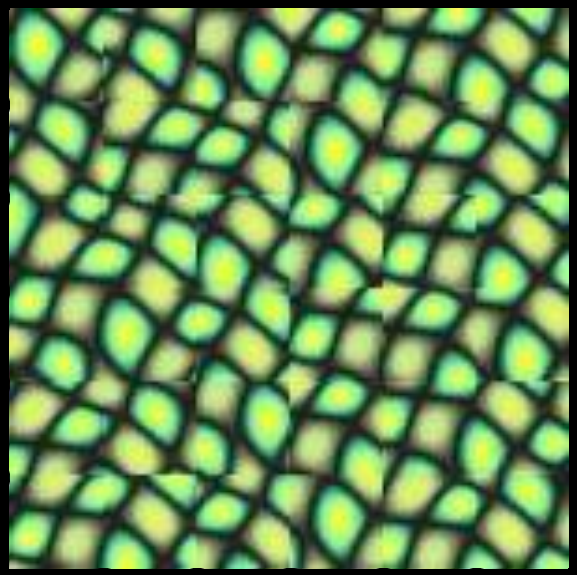


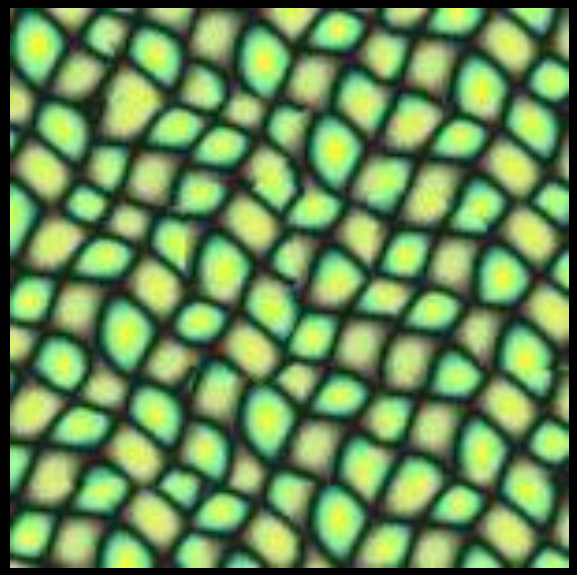
Neighboring blocks constrained by overlap



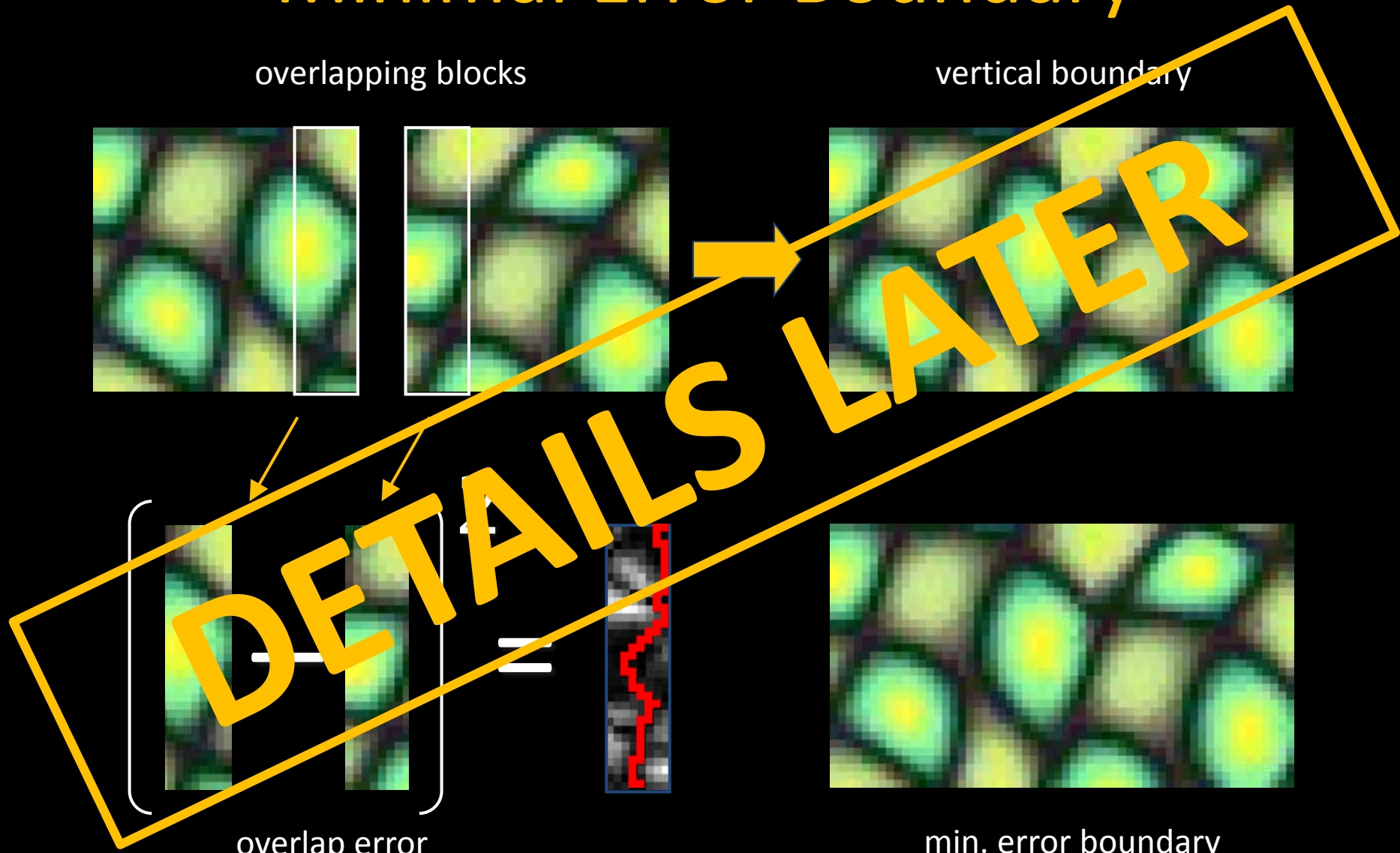
Minimal error boundary cut







Minimal Error Boundary

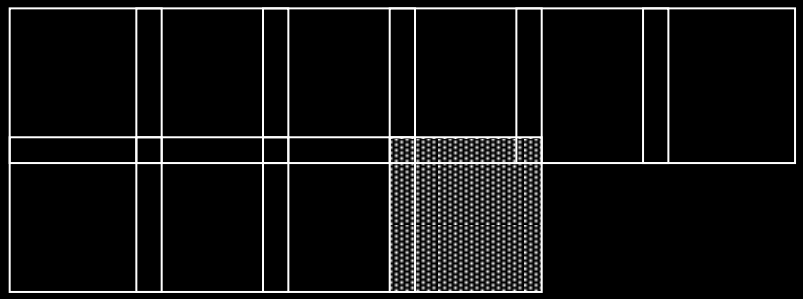


Philosophy

- Texture blocks are by definition correct samples of texture
- The problem is connecting them together

Algorithm

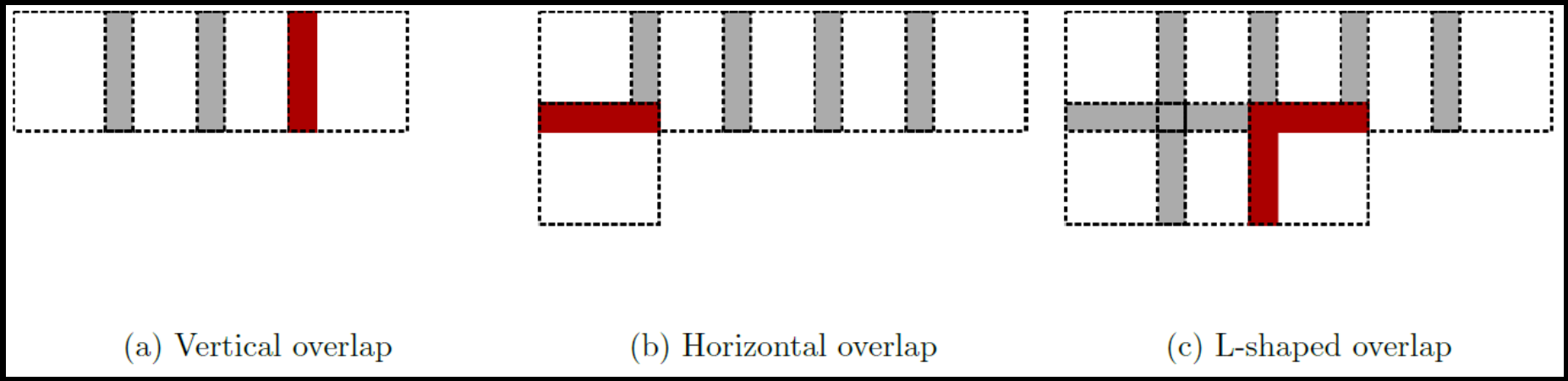
- Parameters: size of block, size of overlap (25%)
- Synthesize blocks in raster order



- 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

Three overlap cases

- There are 3 overlap cases
  - Vertical overlap (first row)
  - Horizontal overlap (first column)
  - L-shaped overlap (everywhere else)



(a) Vertical overlap (b) Horizontal overlap (c) L-shaped overlap

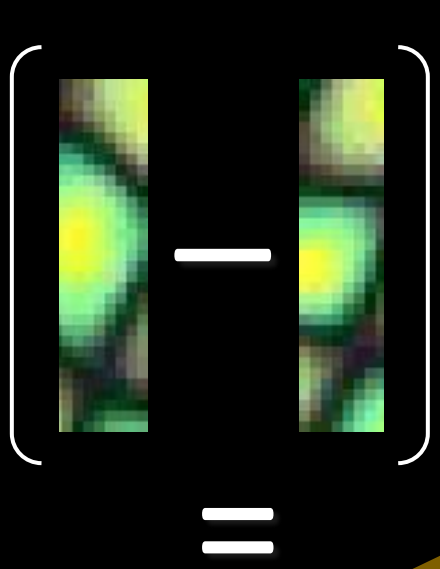
Overlap Constraint

- We need to search to all blocks that have a similar L-shape

$$d(P, P_A((x,y))) = \sum_{i,j=0}^{w-1} Q(i,j)(P(i,j) - A(x+i, x+j))^2$$

- Can be computed using Fast Fourier transform
- Then cost is only proportional to input size in O(MNlog(MN)), independent of block size
- Again, pick block at random with error tolerance ε

Boundary Error Computation



- For vertical boundary
- Search the pixel path that minimizes e = (A-B)²
- **Dynamic programing**
  - Starting from bottom compute minimal cumulative error E

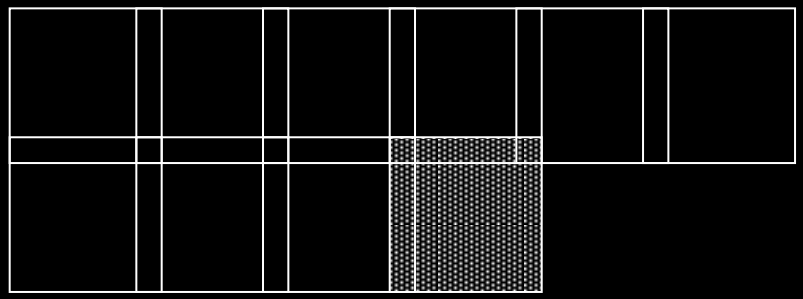
$$E_{i,j} = e_{i,j} + \min(E_{i+1,j-1}, E_{i+1,j}, E_{i+1,j-1})$$

  - Starting point is minimal value on top
  - Retrace back the minimal path

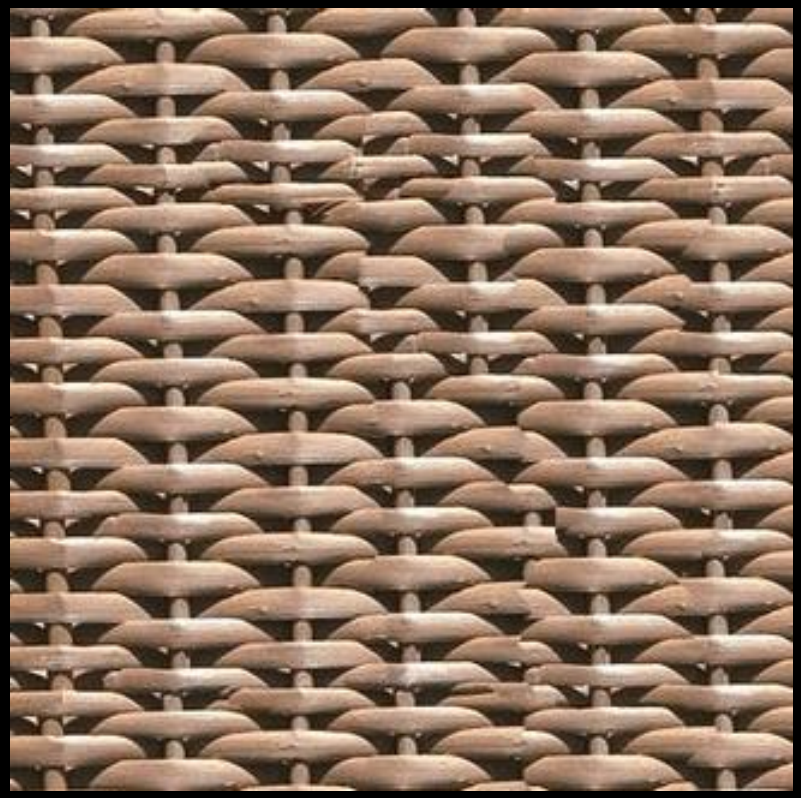
DETAILS…

Boundary Error Computation

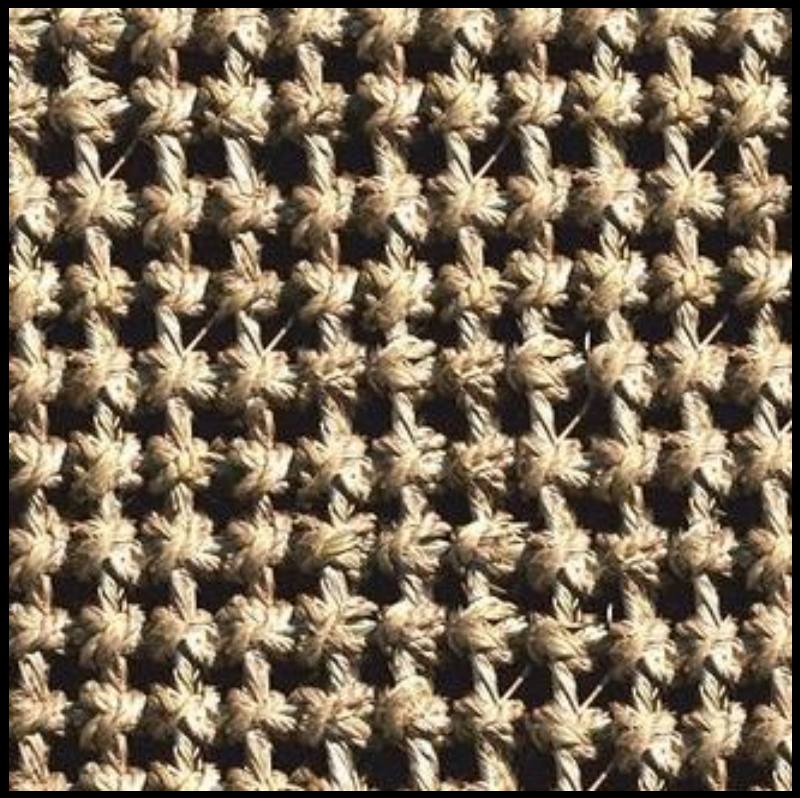
- What about L-shape overlap?



Results



Results

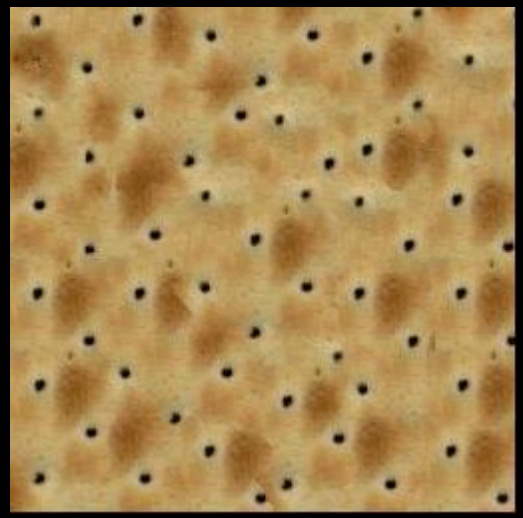


Results

Results

Results

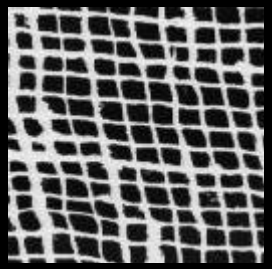
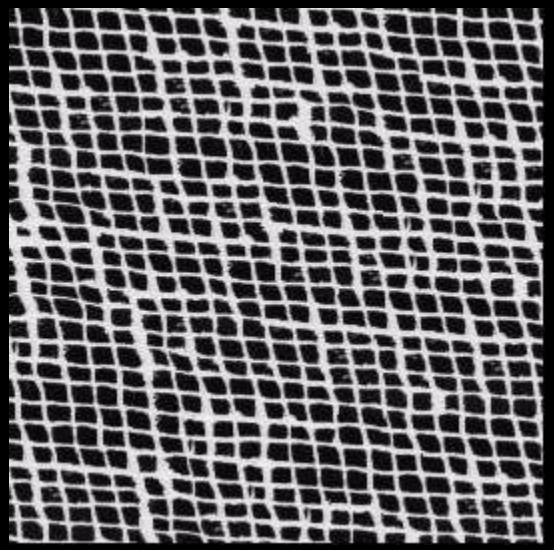
Results





Results





Failures

Texture Transfer

- Take the texture from one object and “paint” it onto another object
  - This requires separating texture and shape
  - Assume we can capture shape by boundary and rough shading
- Then, just add another constraint when sampling: similarity to underlying image at that spot

Texture Transfer

+

parmesan

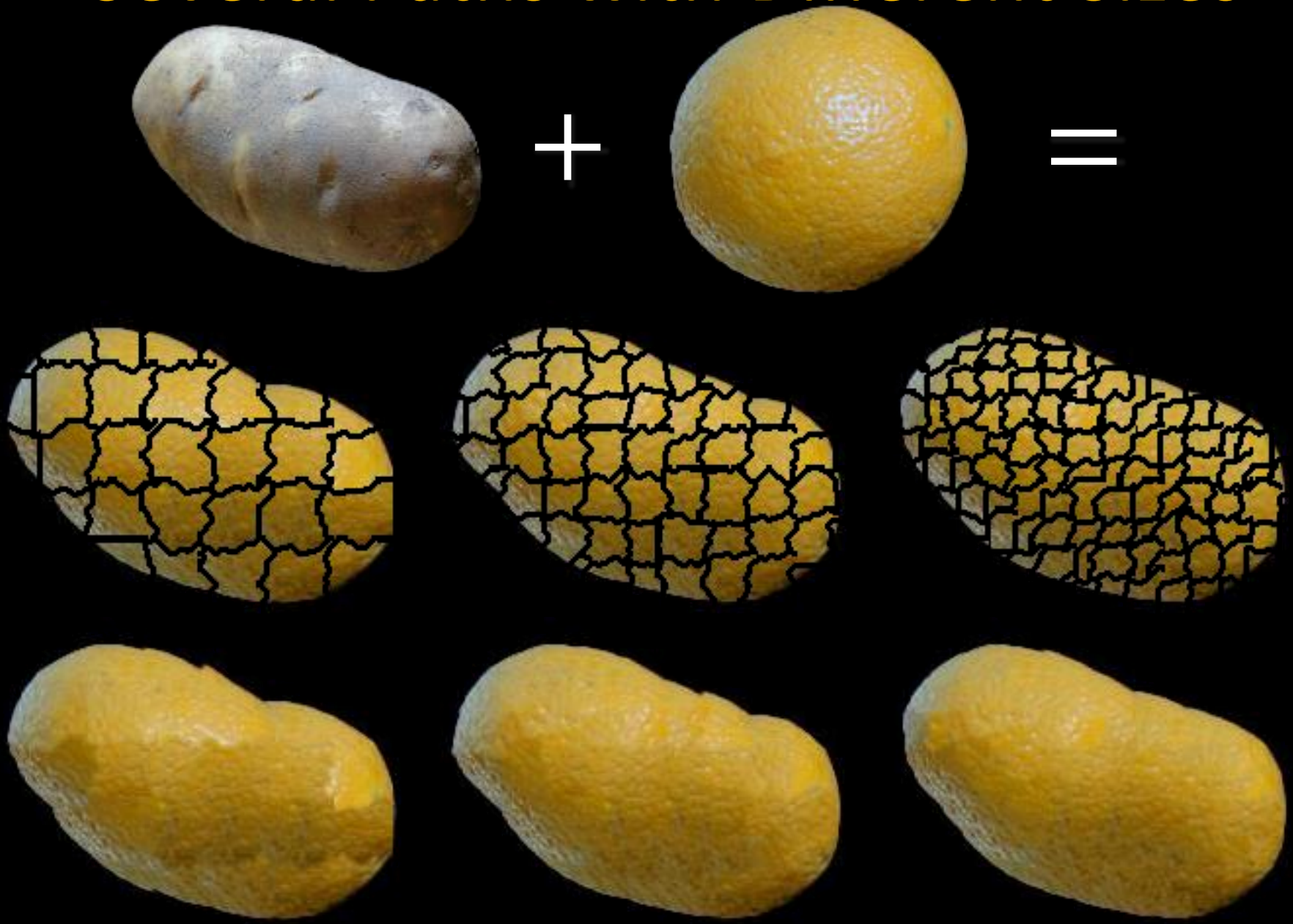
=

+

rice

=

Several Paths with Different Sizes



Pros & Cons

- Advantages
  - Fast and very simple
  - Good results
- Disadvantages
  - Size parameter hard to set
  - Scanning order: Quality tends to be better in top left corner for large images (TP)

Questions?