

Texture Synthesis and Hole-Filling



Computational Photography
Derek Hoiem, University of Illinois

Next section: The digital canvas



Cutting and pasting objects,
filling holes, and blending

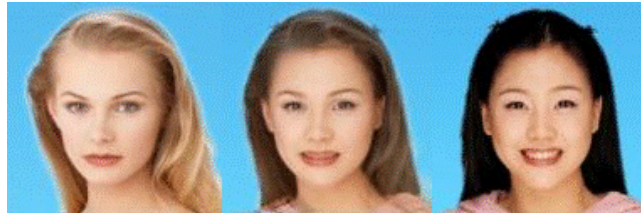


Image warping and object
morphing

Today's Class

- Texture synthesis and hole-filling



Texture

- Texture depicts spatially repeating patterns
- Textures appear naturally and frequently



radishes



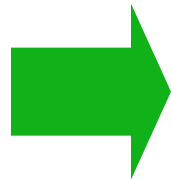
rocks



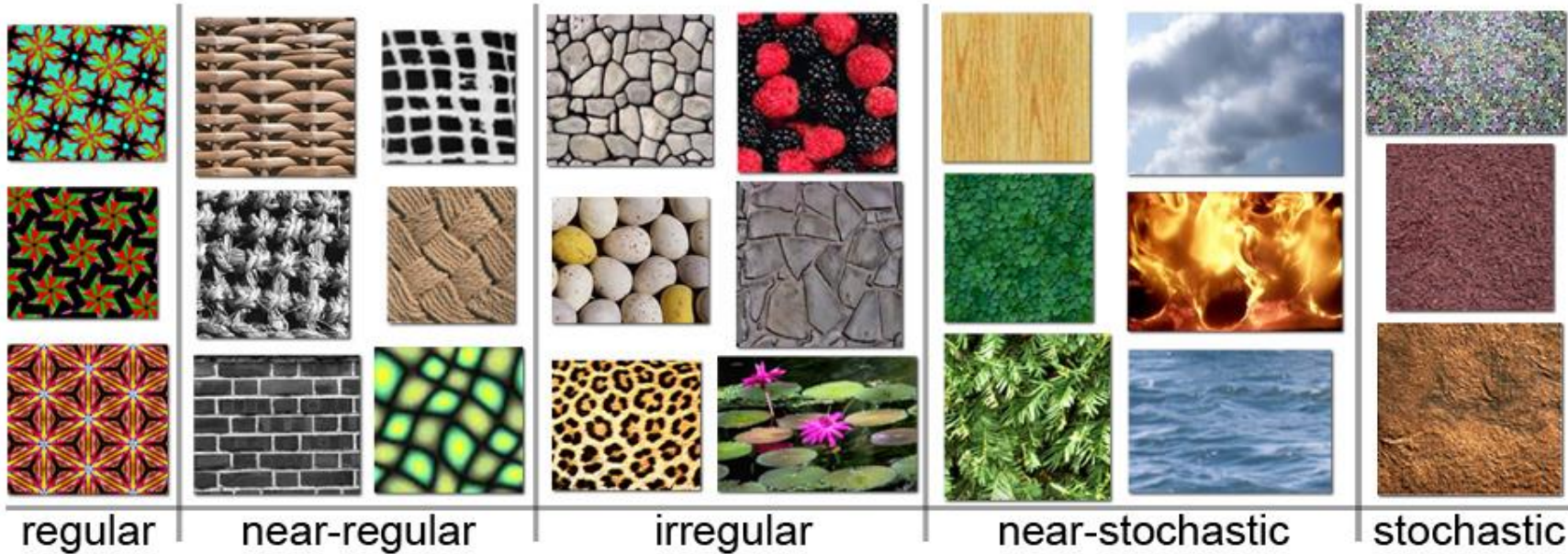
yogurt

Texture Synthesis

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



The Challenge

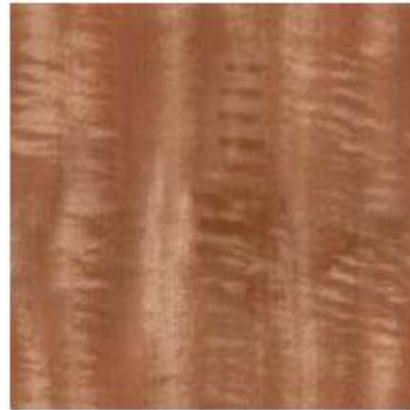


Need to model the whole spectrum: from repeated to stochastic texture

One idea: Build Probability Distributions

Basic idea

1. Compute statistics of input texture (e.g., histogram of edge filter responses)
2. Generate a new texture that keeps those same statistics



- D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *SIGGRAPH '95*.
- E. P. Simoncelli and J. Portilla. Texture characterization via joint statistics of wavelet coefficient magnitudes. In *ICIP 1998*.

One idea: Build Probability Distributions

But it (usually) doesn't work

- Probability distributions are hard to model well

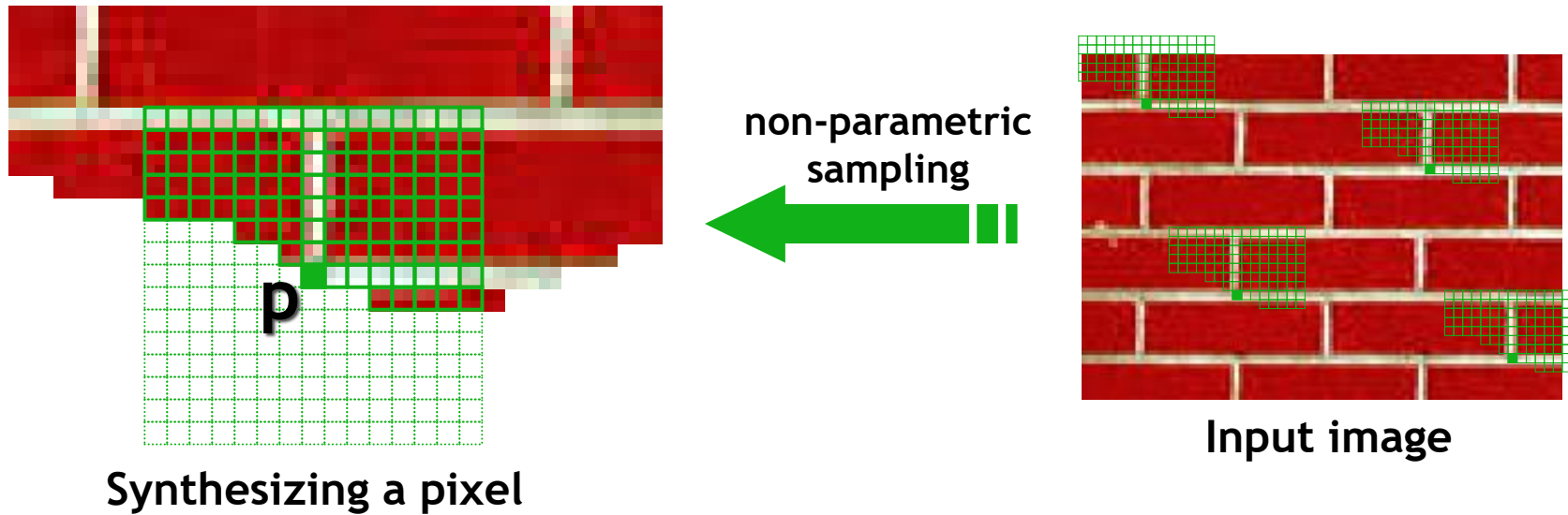
Input



Synthesized



Another idea: Sample from the image



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

Idea from Shannon (Information Theory)

- Generate English-sounding sentences by modeling the probability of each word given the previous words (n-grams)
- Large “n” will give more structured sentences

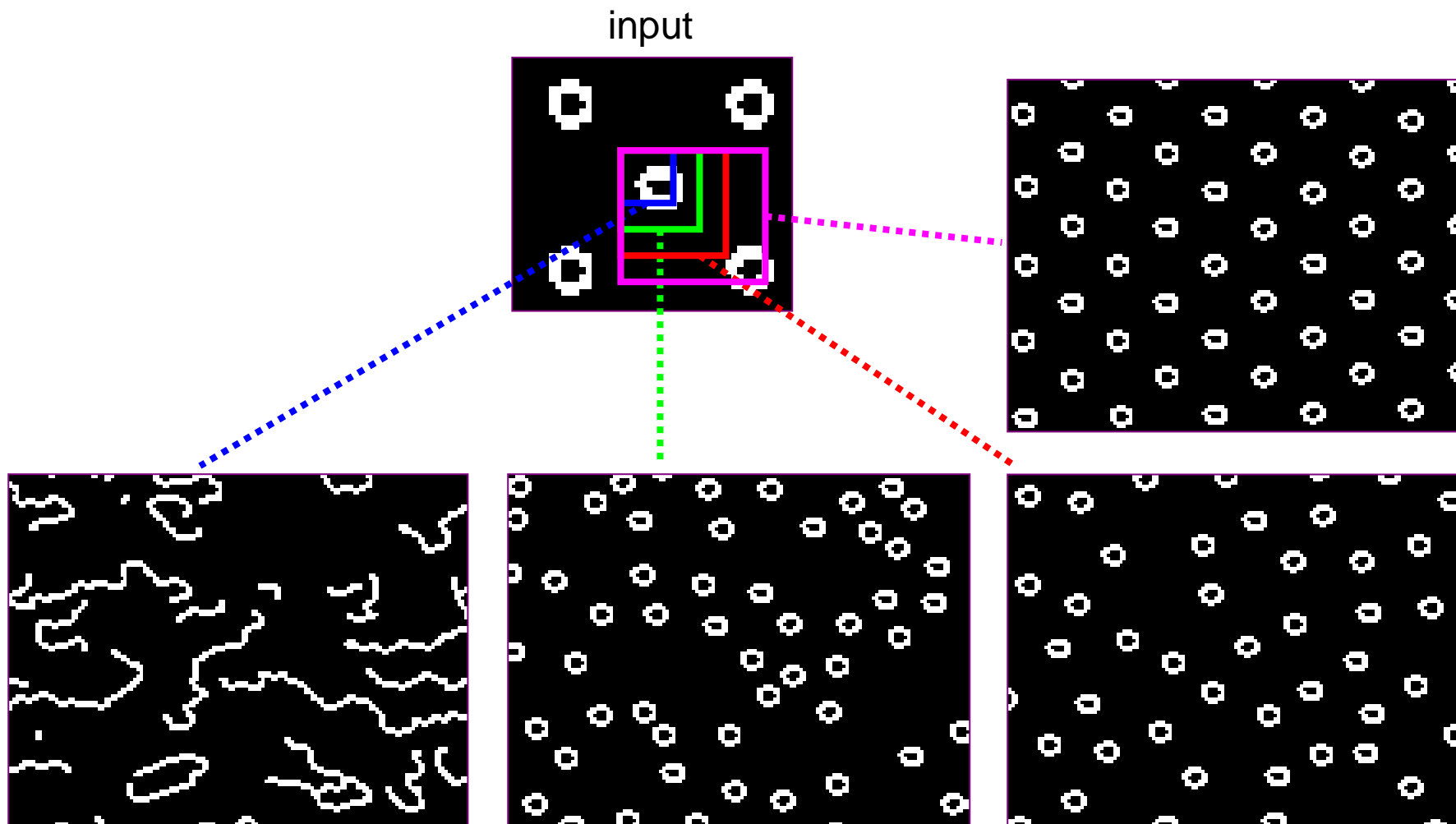
“I spent an interesting evening recently
with a grain of salt.”

(example from fake single.net user [Mark V Shaney](#))

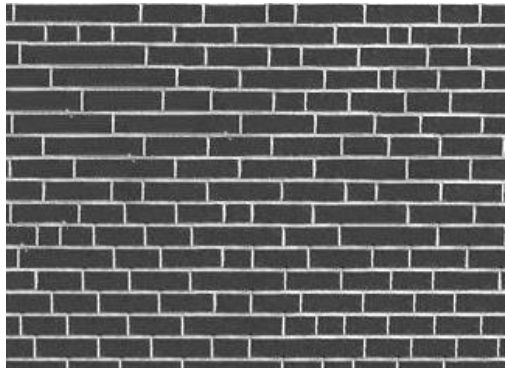
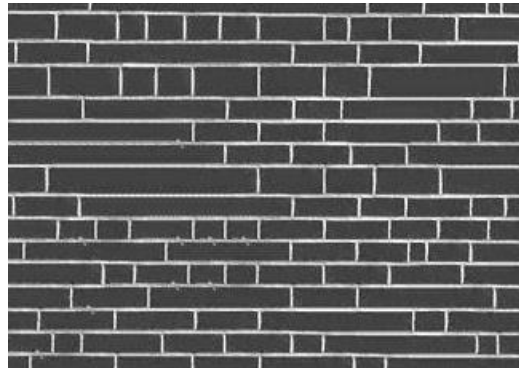
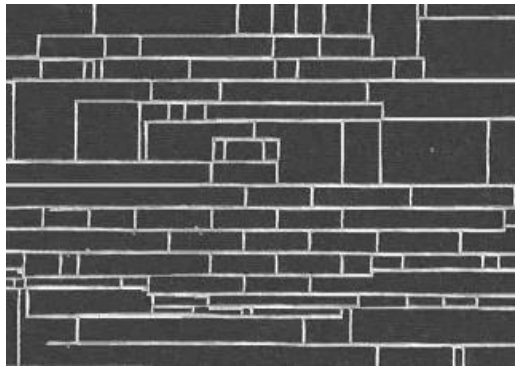
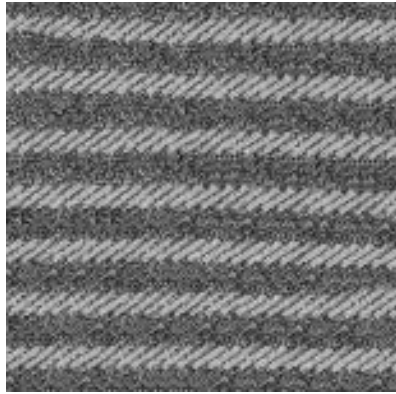
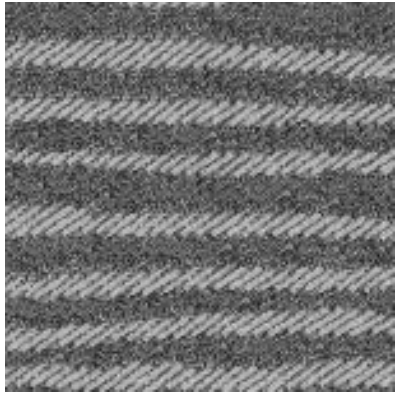
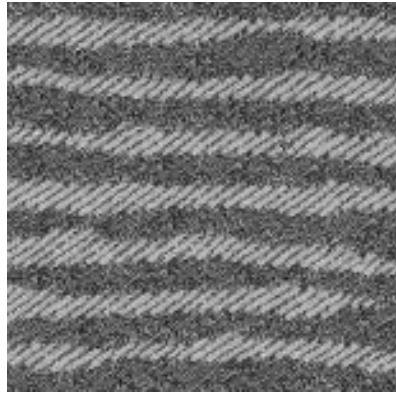
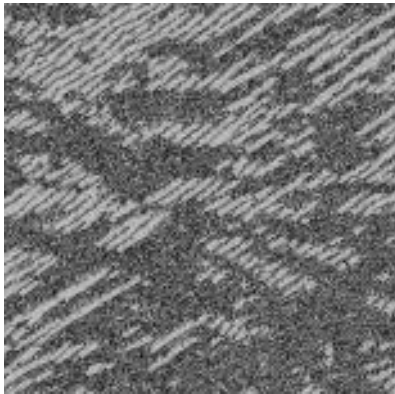
Details

- How to match patches?
 - Gaussian-weighted SSD (more emphasis on nearby pixels)
- What order to fill in new pixels?
 - “Onion skin” order: pixels with most neighbors are synthesized first
 - To synthesize from scratch, start with a randomly selected small patch from the source texture
- How big should the patches be?

Size of Neighborhood Window



Varying Window Size



Increasing window size



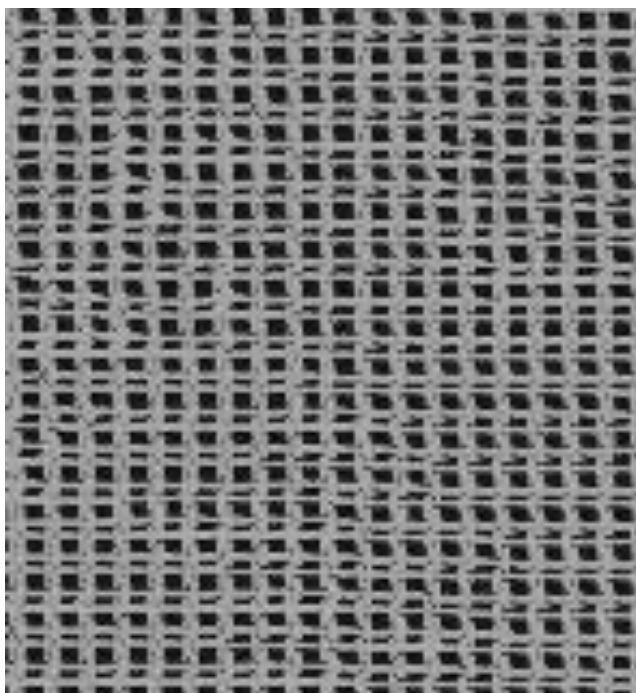
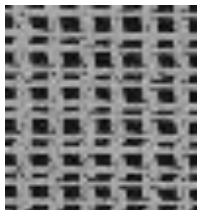
Texture synthesis algorithm

While image not filled

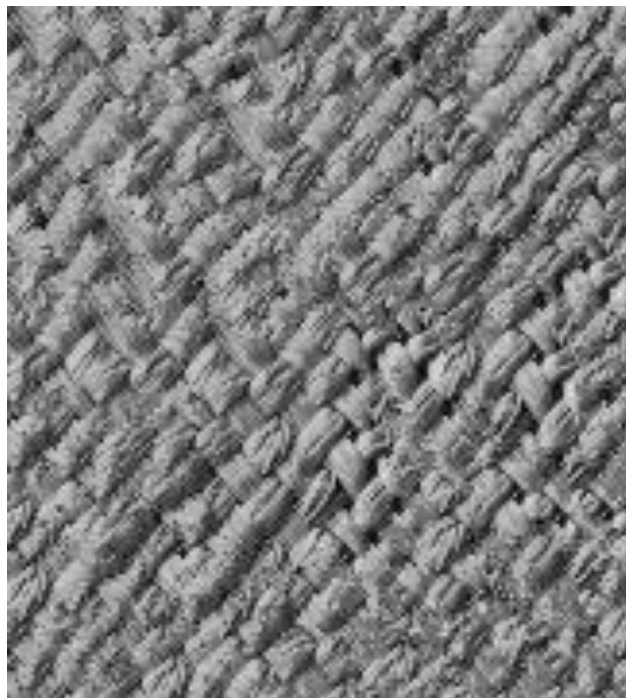
1. Get unfilled pixels with filled neighbors, sorted by number of filled neighbors
2. For each pixel, get top N matches based on visible neighbors
 - Patch Distance: Gaussian-weighted SSD
3. Randomly select one of the matches and copy pixel from it

Synthesis Results

french canvas



rafia weave

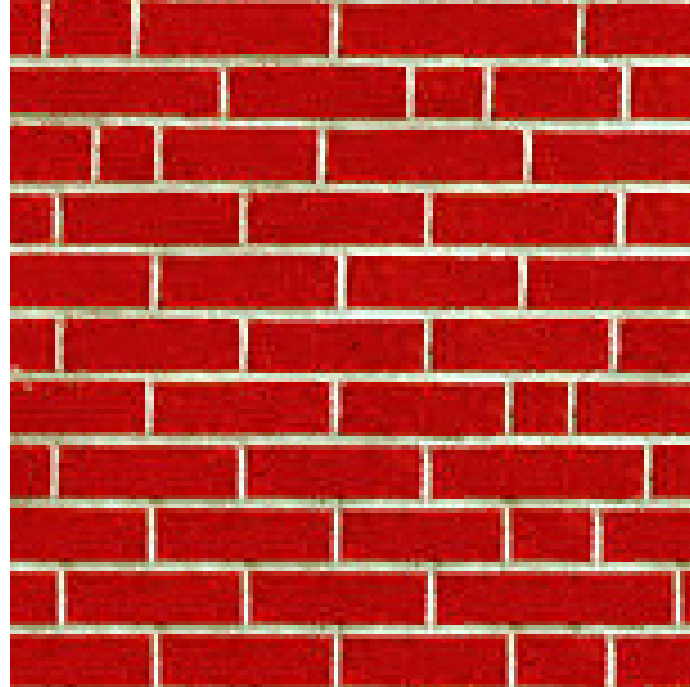
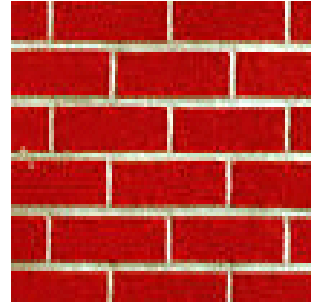


More Results

white bread



brick wall



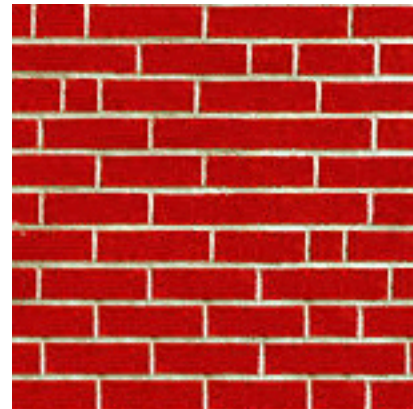
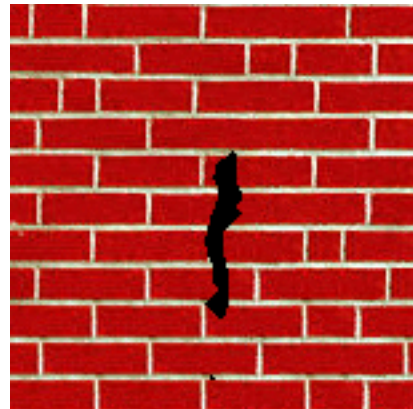
Homage to Shannon

coming in the unsensational
r Dick Gephardt was fair
rful riff on the looming
nly asked, "What's your
tions?" A heartfelt sigh
story about the emergen
es against Clinton. "Boy
g people about continuin
ardt began, patiently obs
s, that the legal system k
g with this latest tanger

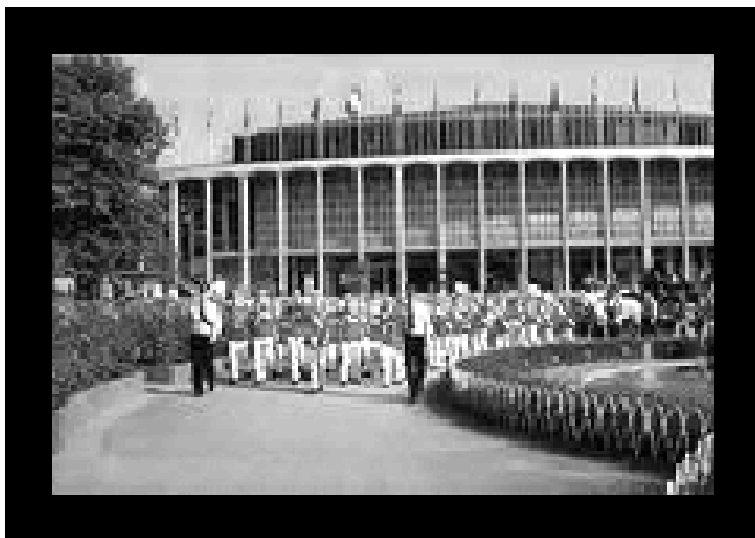
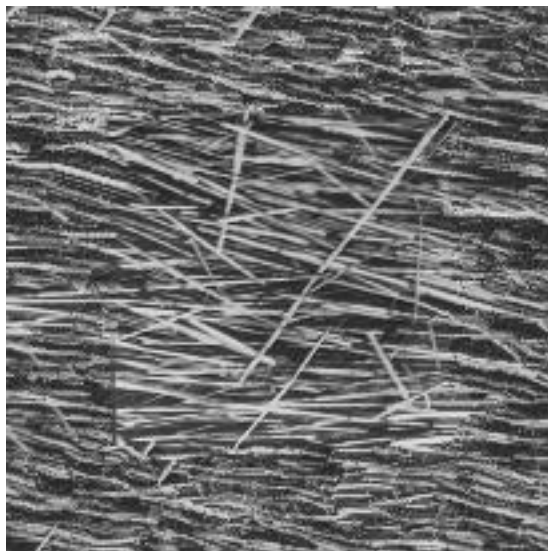
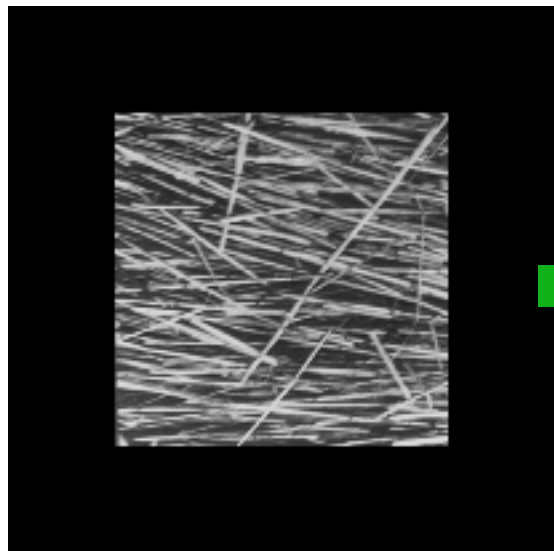


ithaim. them . "Whnephartfe lartifelintomimen
lel ck Clirtioout omaim thartfelins.f out s anetc
the ry onst wartfe lck Gephtoomimeationl sigak
Chiooufit Clinut Cll riff on. hat's y'odn, parut tly:
ons yoonstehst waked, paim t sahe loo riff on l
nskoneploourtfeas leil A nst Clit, "Wleontongal s
k Cirtioouirtfepe.ong pme abegal fartfenstemem
tiensteneltorydt telemephminsverdt was agemer
ff ons artientont Cling peme as artfe atich, "Boui s
nal s fartfelt sig pedrthdt ske abounutie aboutioo
stfaonevwas yow abounthardt thatins fain, ped, '
ains. them, pabout wasy arfiut coutly d, l n A h
le emthrdngbooreme agas fa bontinsyst Clinut
ory about continst Clipeopinst Cloke agatiff out C
stome zinemen tly ardt beorabou n, thenly as t C
cons faimeme Diontont wat coutlyohgans as fan
ien, phrtfaul, "Wbaut cout congagal comininga:
mifmst Cliiy abon al coountha.emungait tfoun
The loocrysta loontieph. intly on, theoplegatick C
ul fatiecontly atie Diontiomt wal s f tbegea ener
nthahgat's enenhhbas fan. "intchthorz abons v

Hole Filling



Extrapolation



In-painting natural scenes



Key idea: Filling order matters

In-painting Result



Image with Hole



Raster-Scan Order



Onion-Peel
(Concentric Layers)



Gradient-Sensitive
Order

Filling order

Fill a pixel that:

1. Is surrounded by other known pixels
2. Is a continuation of a strong gradient or edge



Comparison



Original



With Hole



Onion-Ring Fill



Criminisi

Comparison



a



b



Concentric Layers

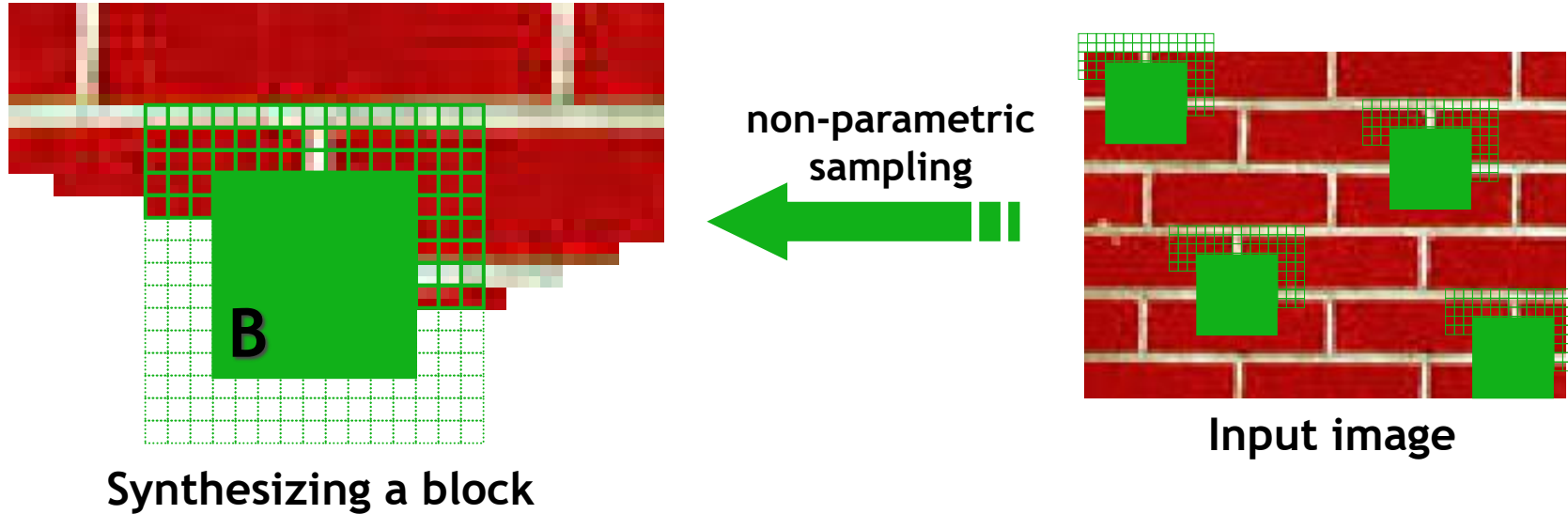


Gradient Sensitive

Summary (so far)

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

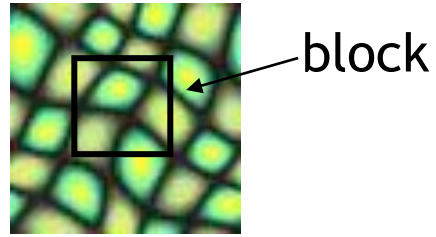
Image Quilting [Efros & Freeman 2001]



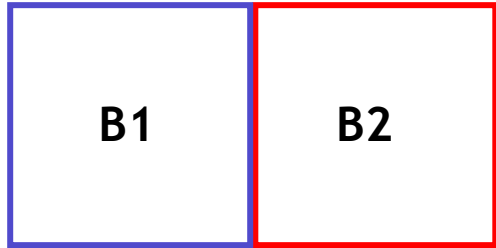
- Observation: neighbor pixels are highly correlated

Idea: unit of synthesis = block

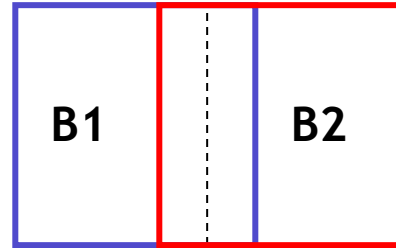
- Exactly the same but now we want $P(\mathbf{B} | \mathbf{N}(\mathbf{B}))$
- Much faster: synthesize all pixels in a block at once



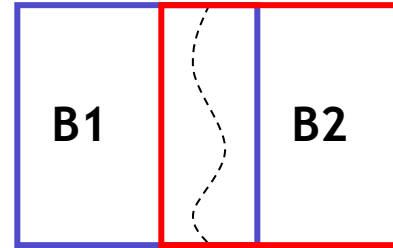
Input texture



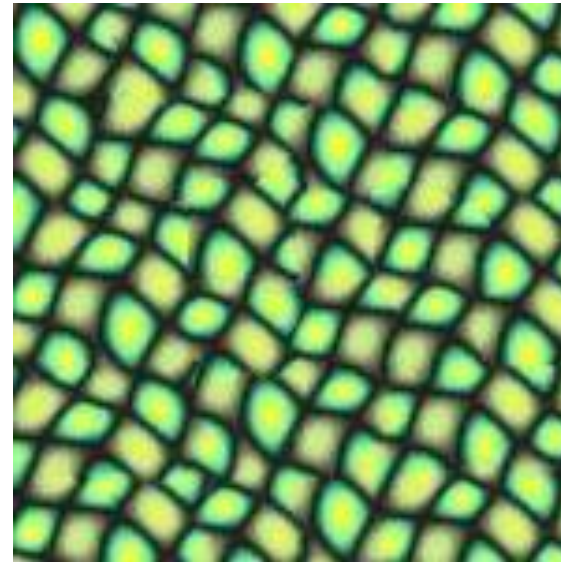
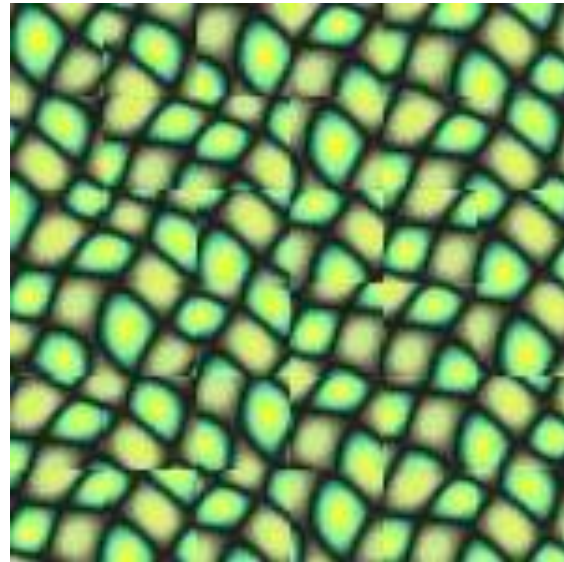
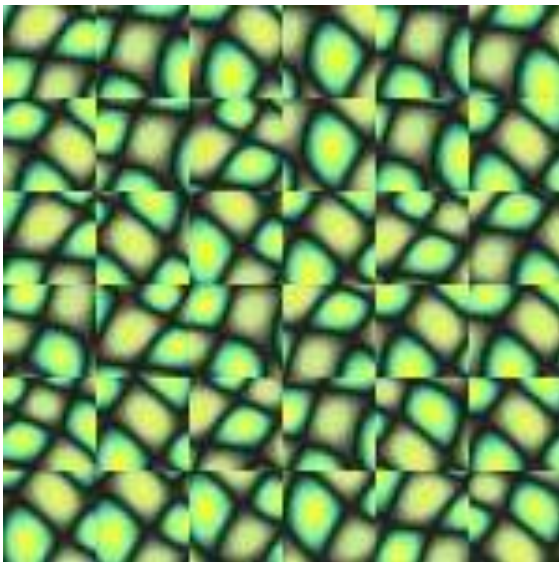
Random placement of blocks



Neighboring blocks constrained by overlap

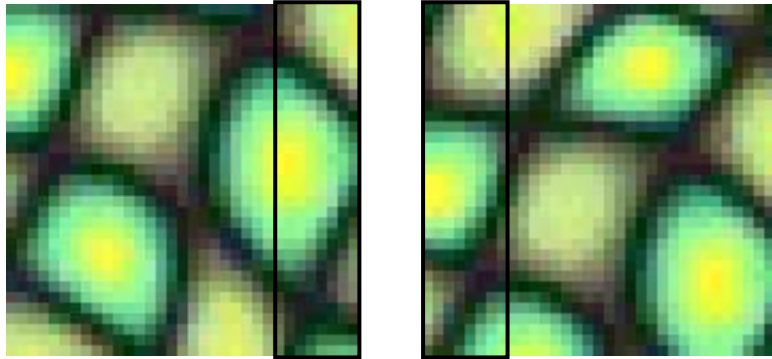


Minimal error boundary cut

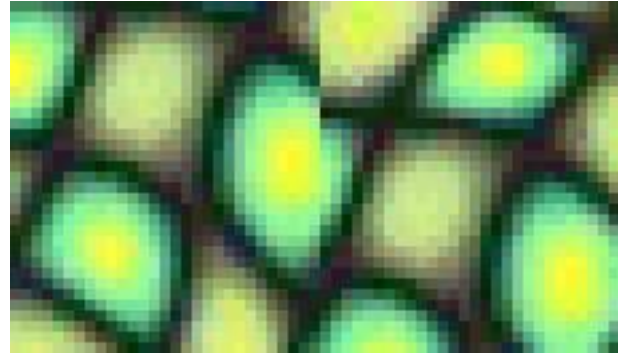


Minimal error boundary

overlapping blocks



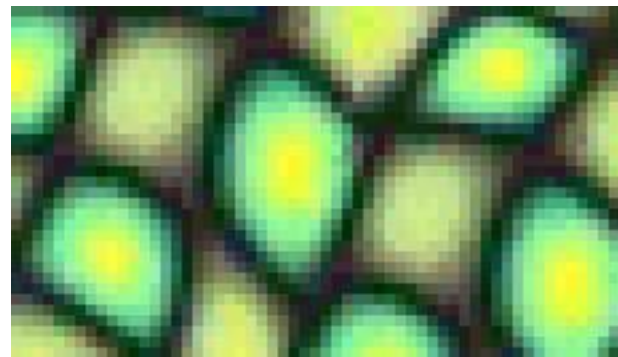
vertical boundary



$$\left[\begin{array}{c} \text{block 1} \\ \text{block 2} \end{array} \right] - \left[\text{block 1} \right] = \text{error map}$$

The diagram shows two overlapping blocks of the cell image. A minus sign is placed between them, followed by a large number 2, indicating a squared difference. This is followed by an equals sign and a vertical error map where the boundary is highlighted in red.

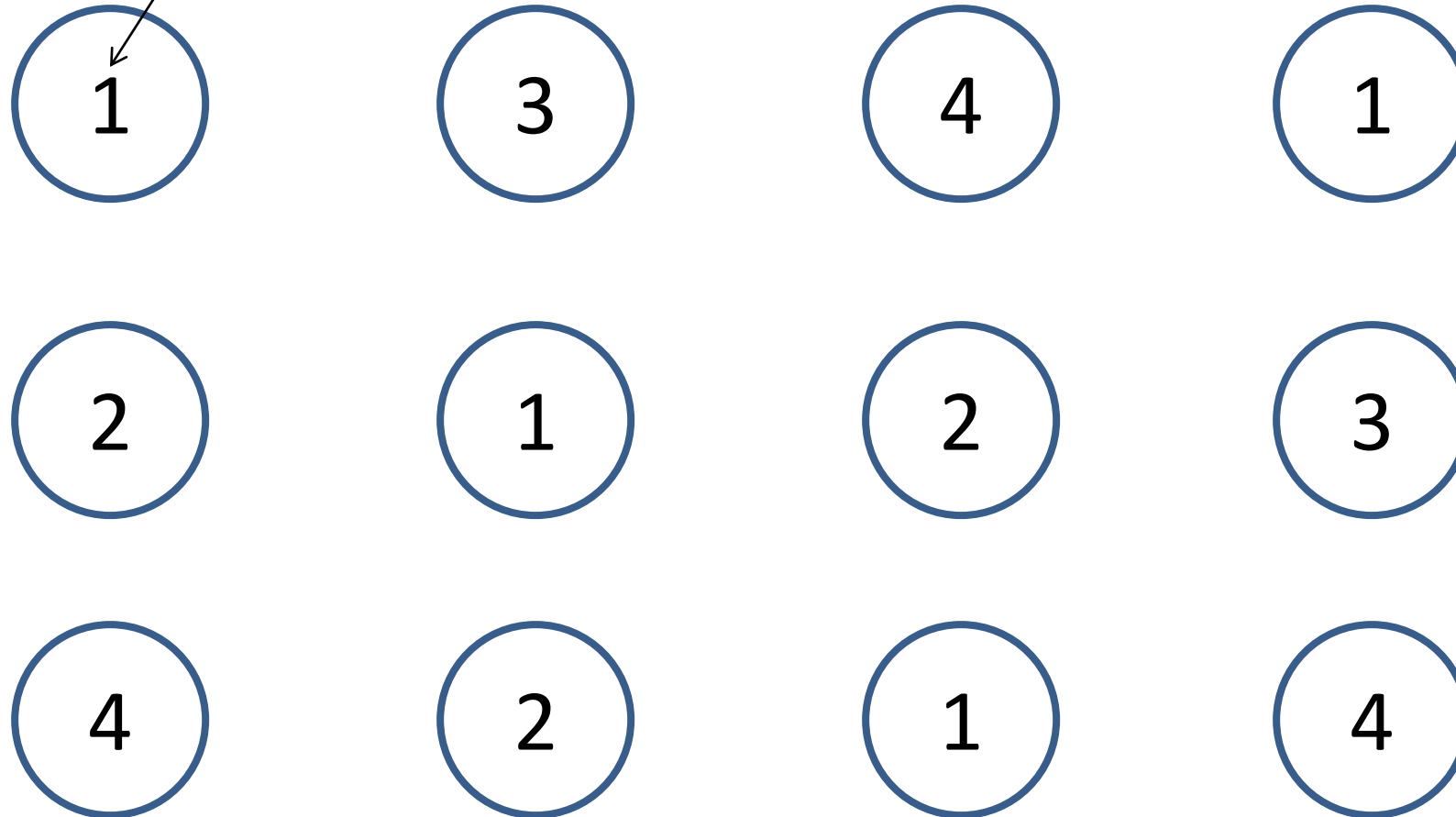
overlap error



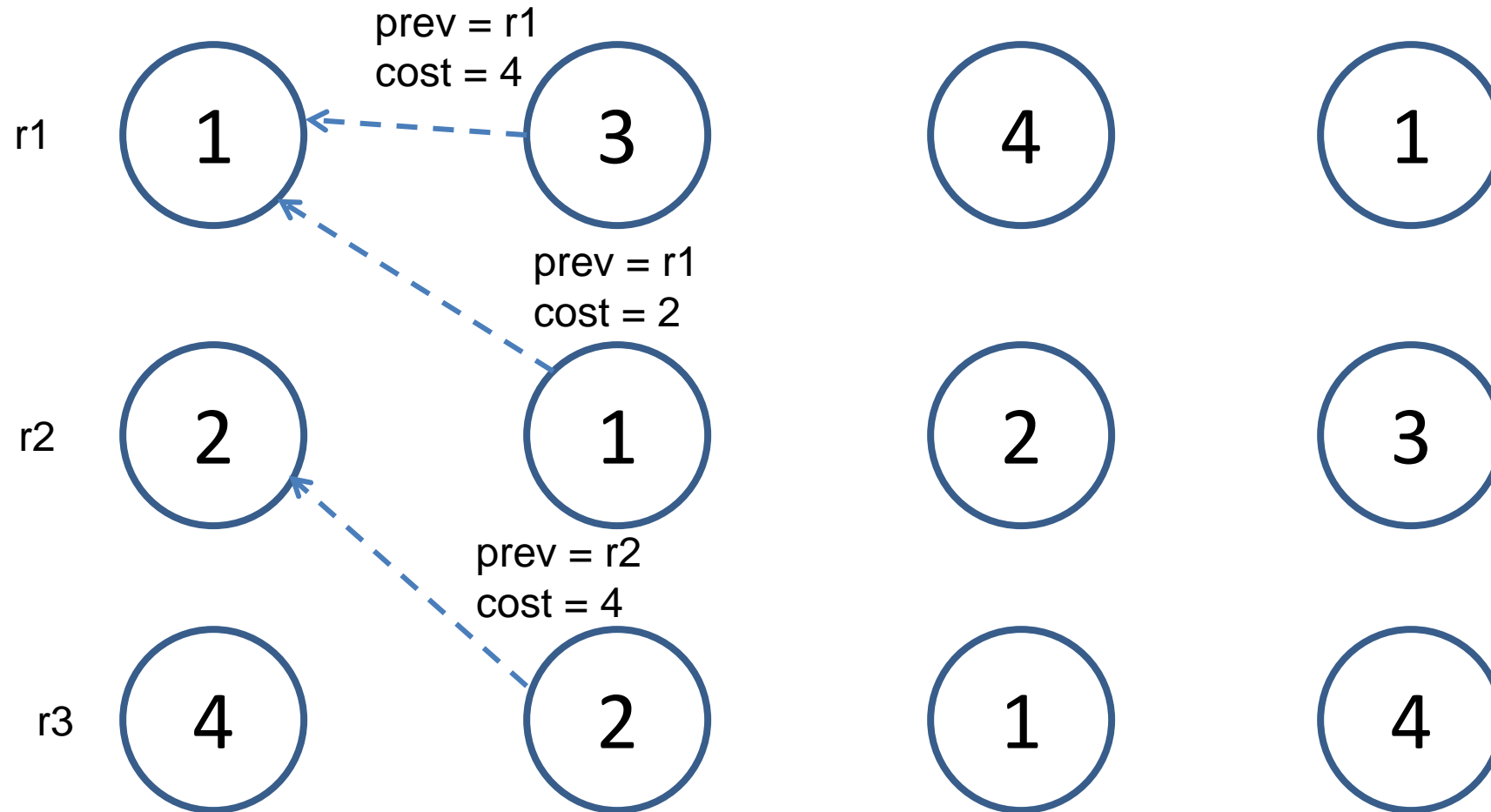
min. error boundary

Solving for Minimum Cut Path

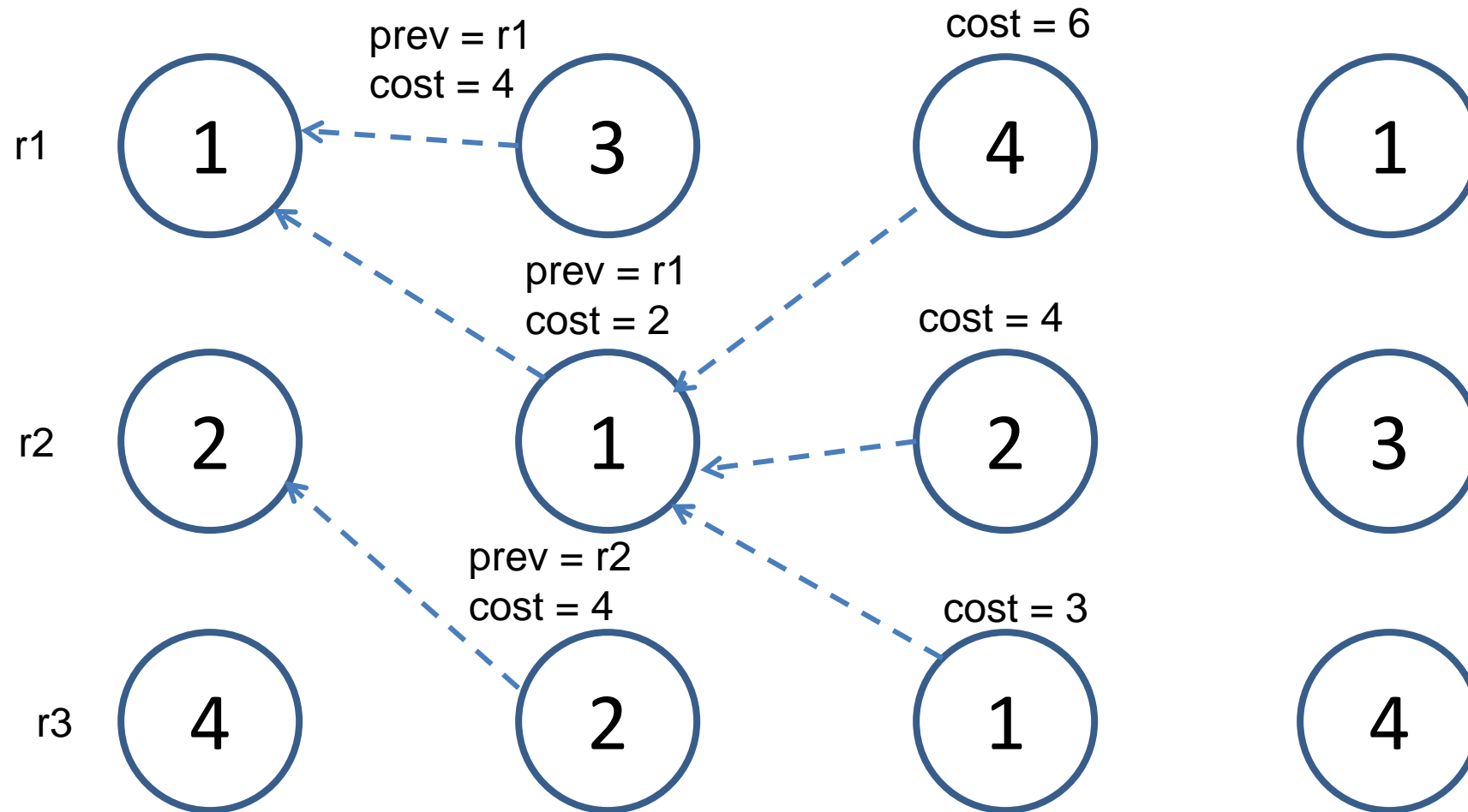
Cost of a cut through this pixel



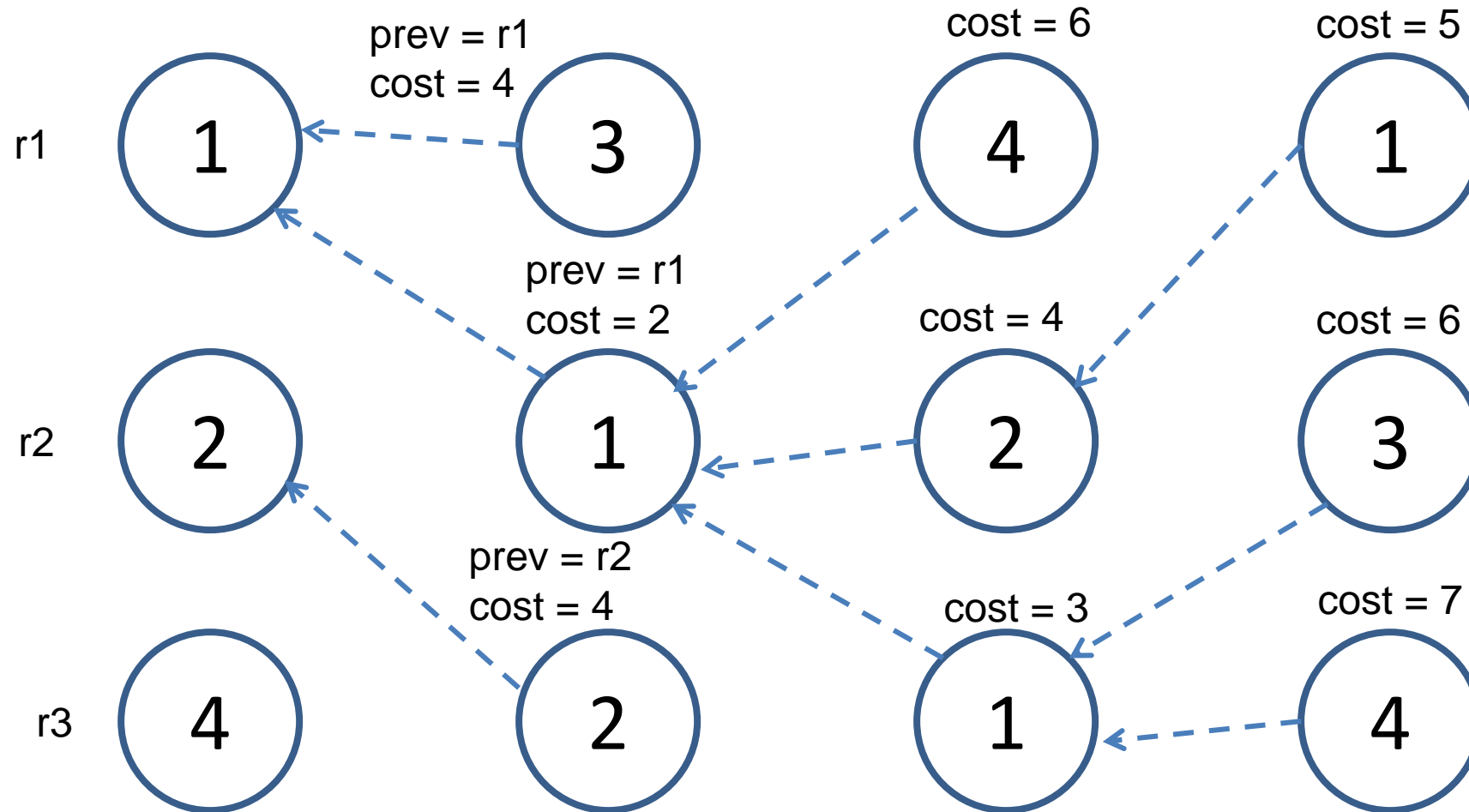
Solving for Minimum Cut Path



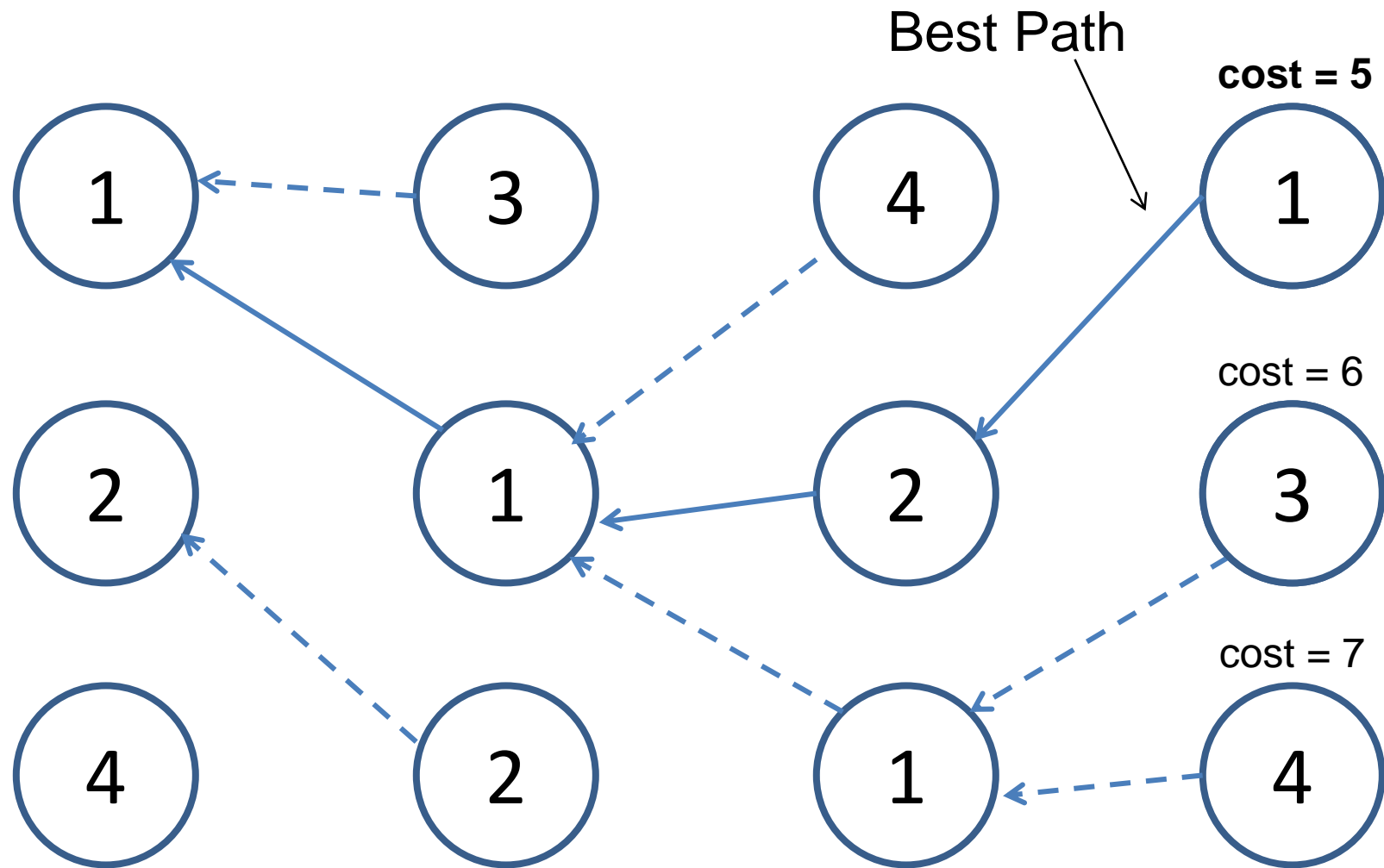
Solving for Minimum Cut Path



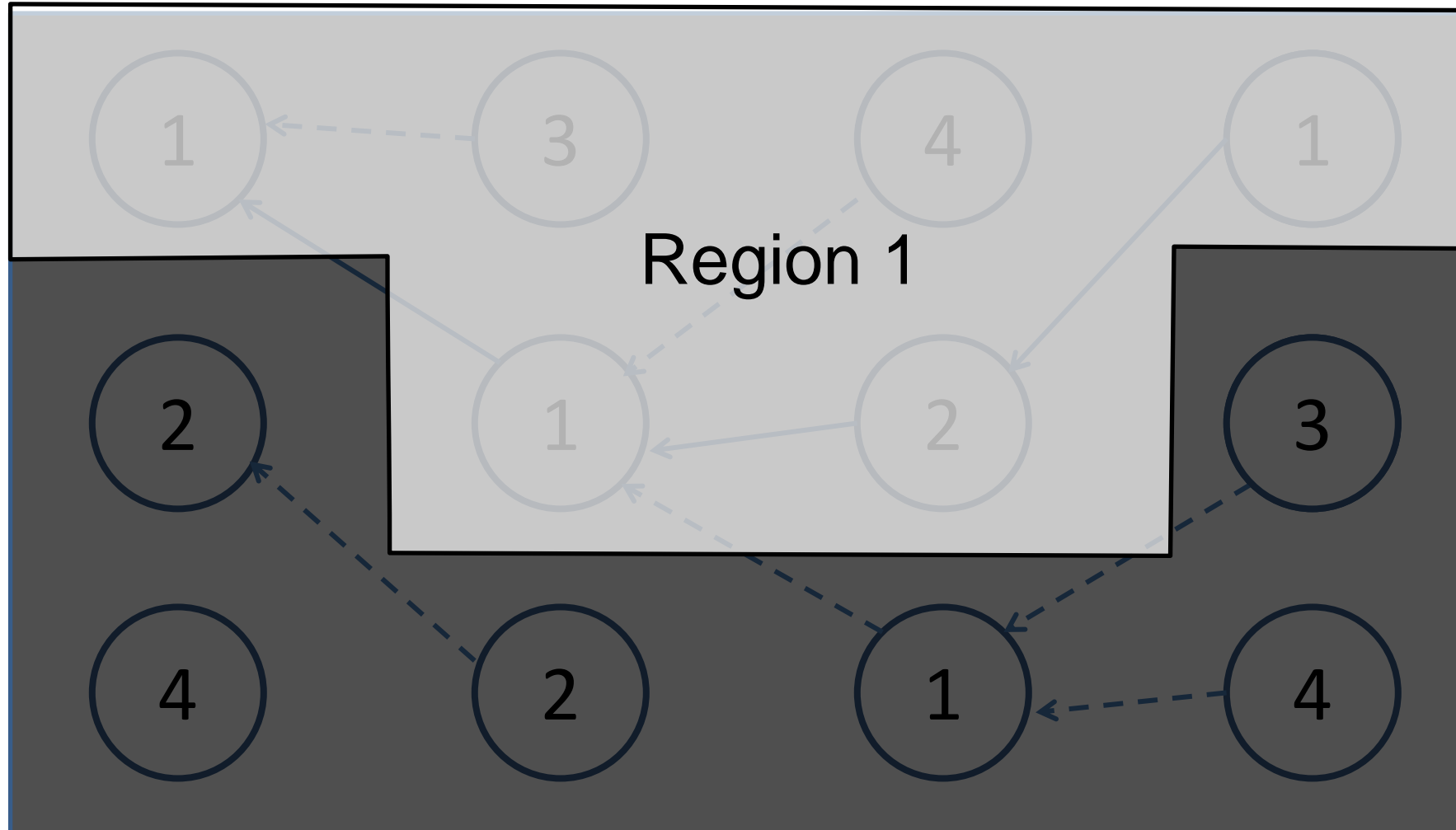
Solving for Minimum Cut Path



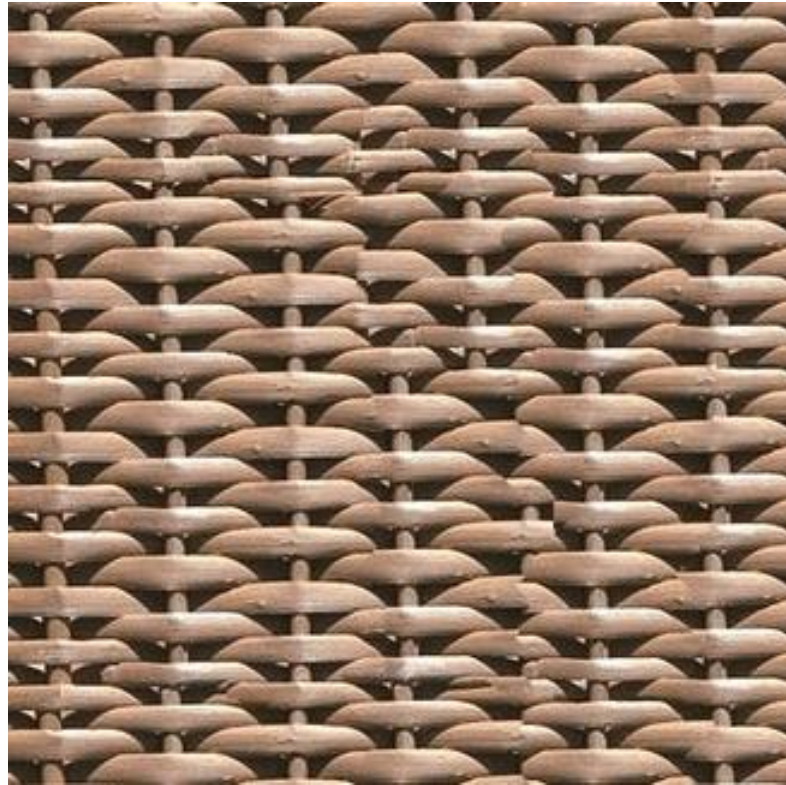
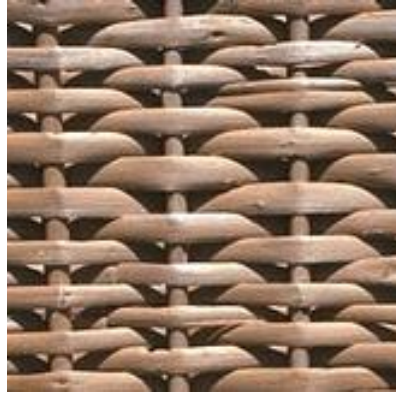
Solving for Minimum Cut Path

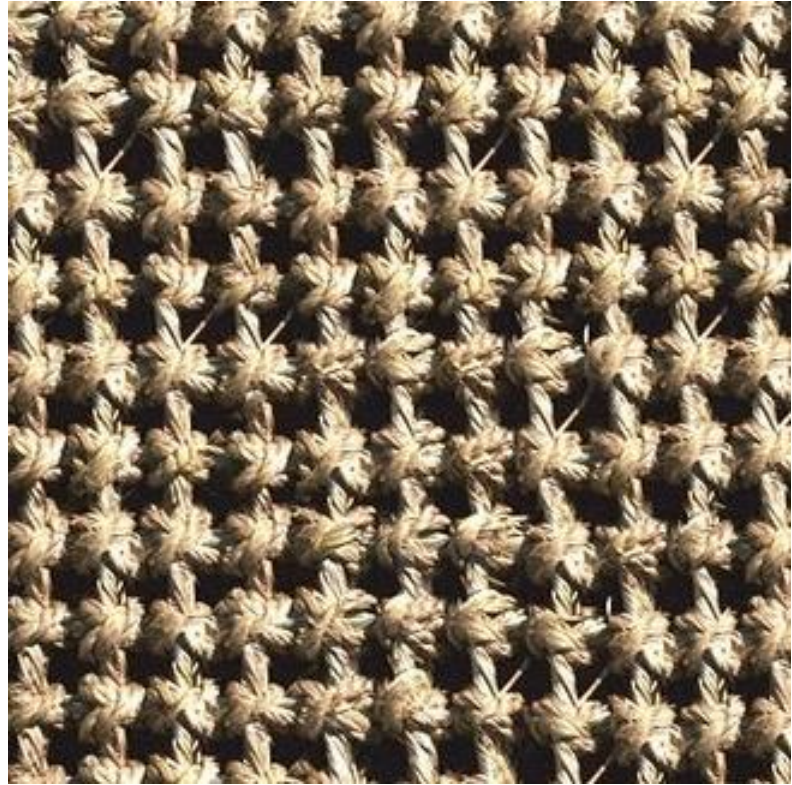


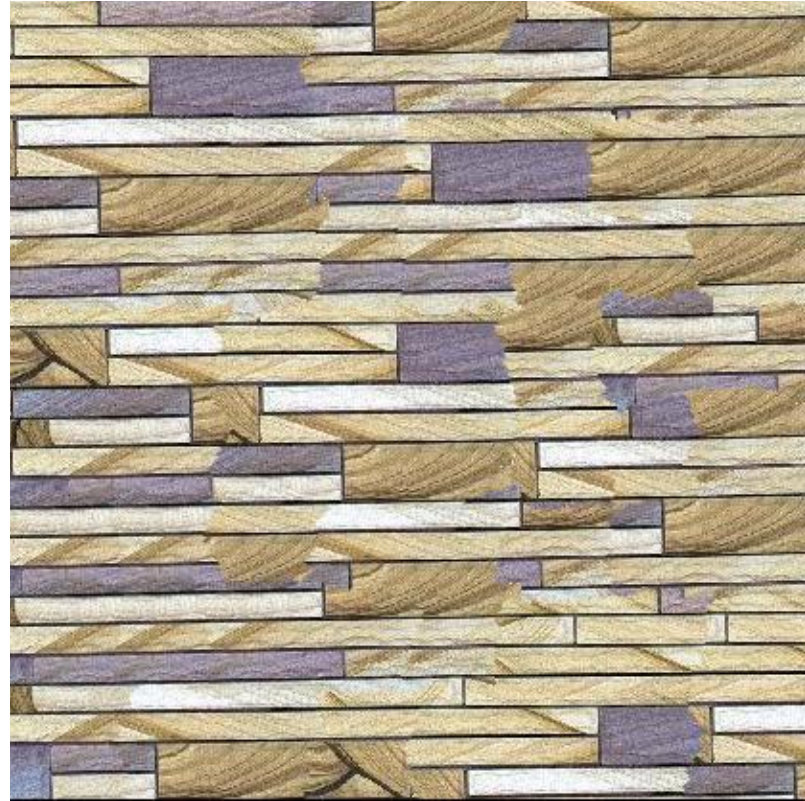
Solving for Minimum Cut Path



Mask Based on Best Path

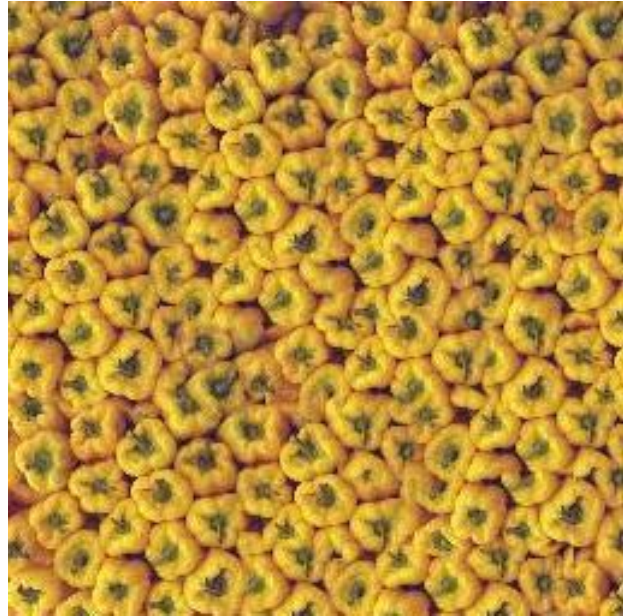
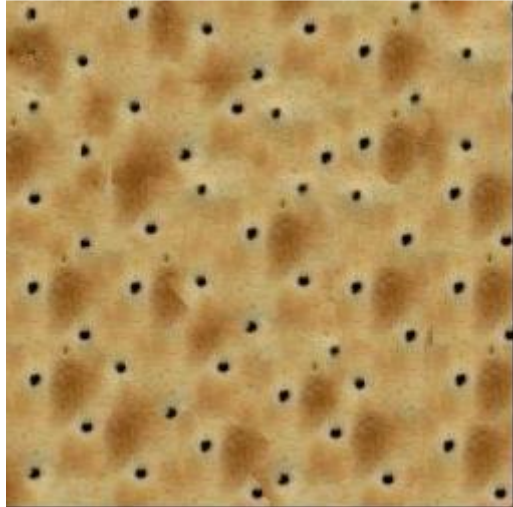
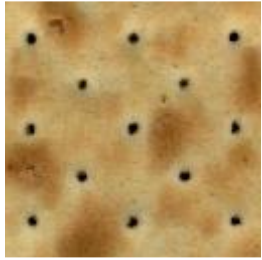


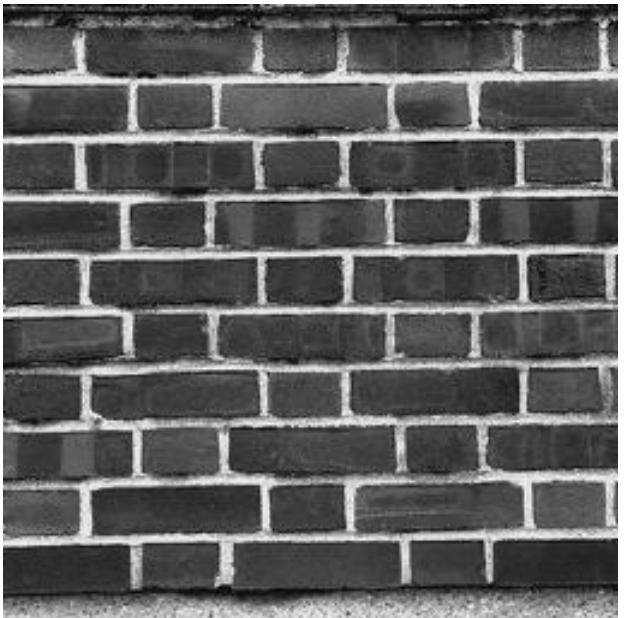








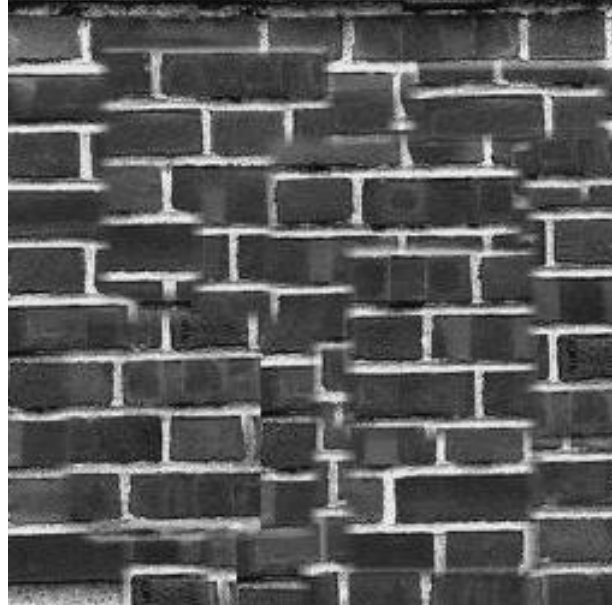




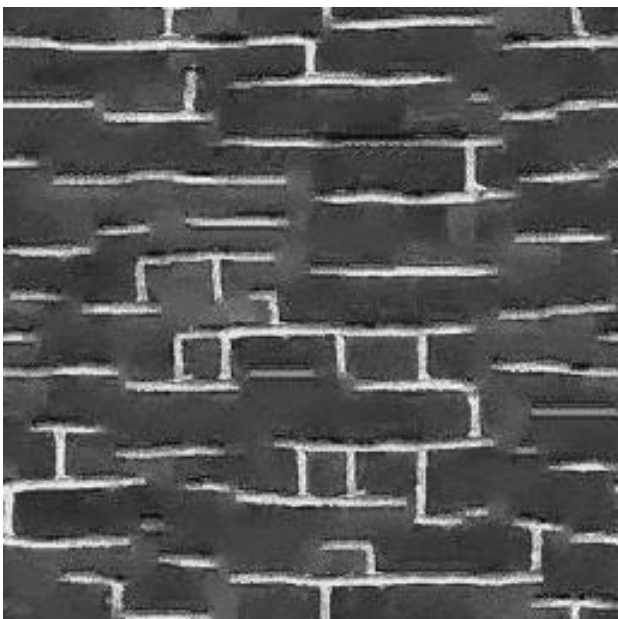
input image



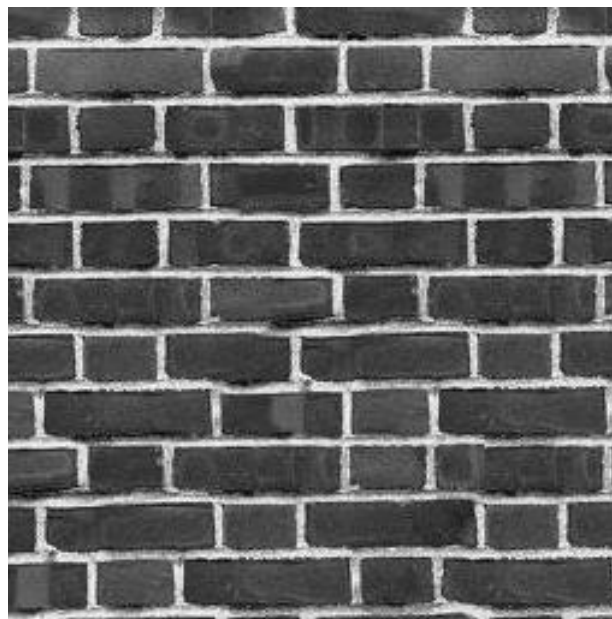
Portilla & Simoncelli



Xu, Guo & Shum



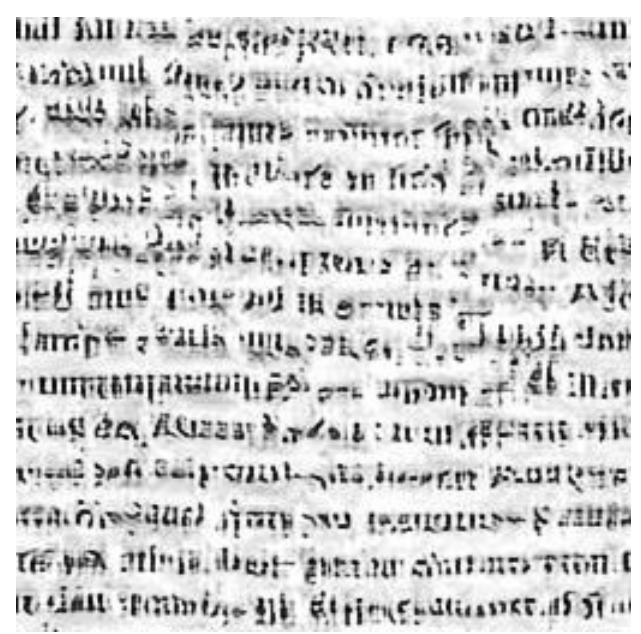
Wei & Levoy



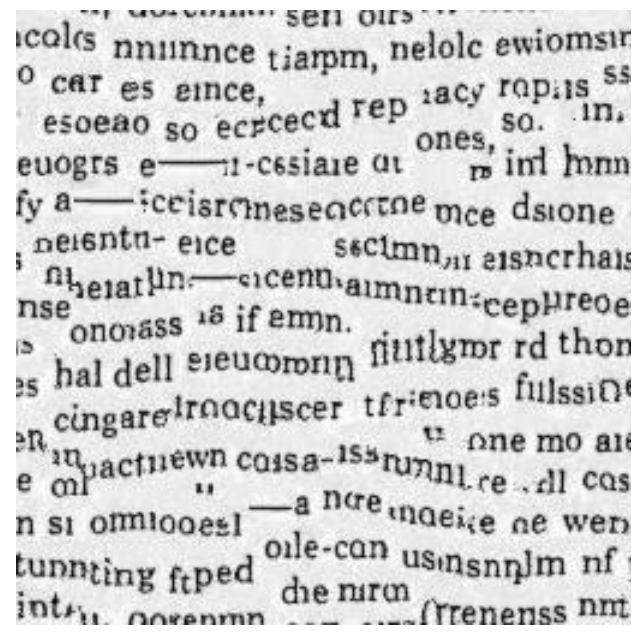
Quilting

end of a visual cortical neuron—the in-
 describing the response of that neuro-
 ht as a function of position—is perhap
 functional description of that neuron.
 seek a single conceptual and mathem.
 describe the wealth of simple-cell recep-
 and neurophysiologically¹⁻³ and inferred
 especially if such a framework has the
 it helps us to understand the functio
 keeper way. Whereas no generic mo-
 ussians (DOG), difference of offset C
 rivative of a Gaussian, higher derivati
 function, and so on—can be expecte
 imple-cell receptive field, we noneth

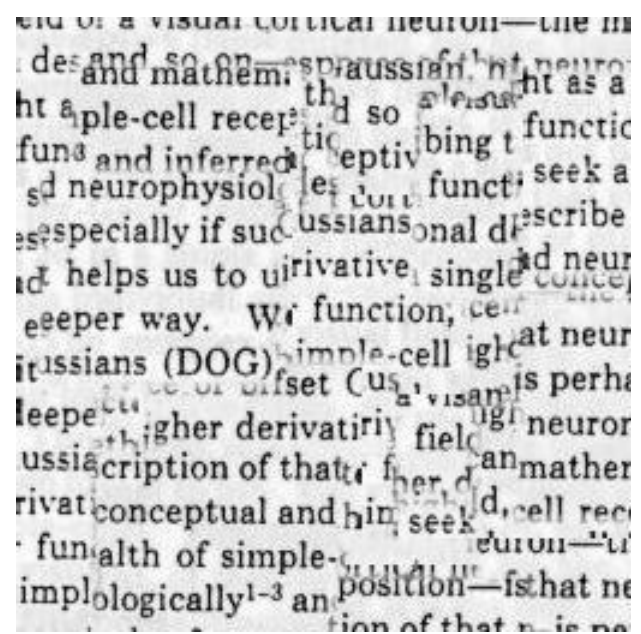
input image



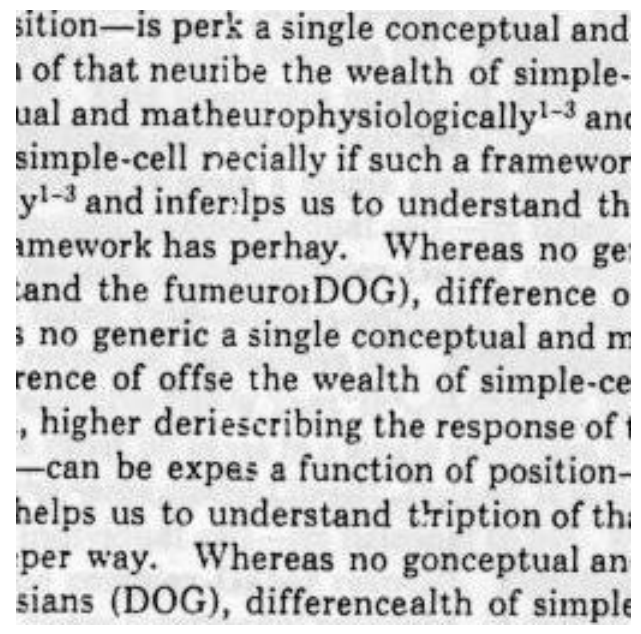
Portilla & Simoncelli



Wei & Levoy



Xu, Guo & Shum



Quilting

Political Texture Synthesis

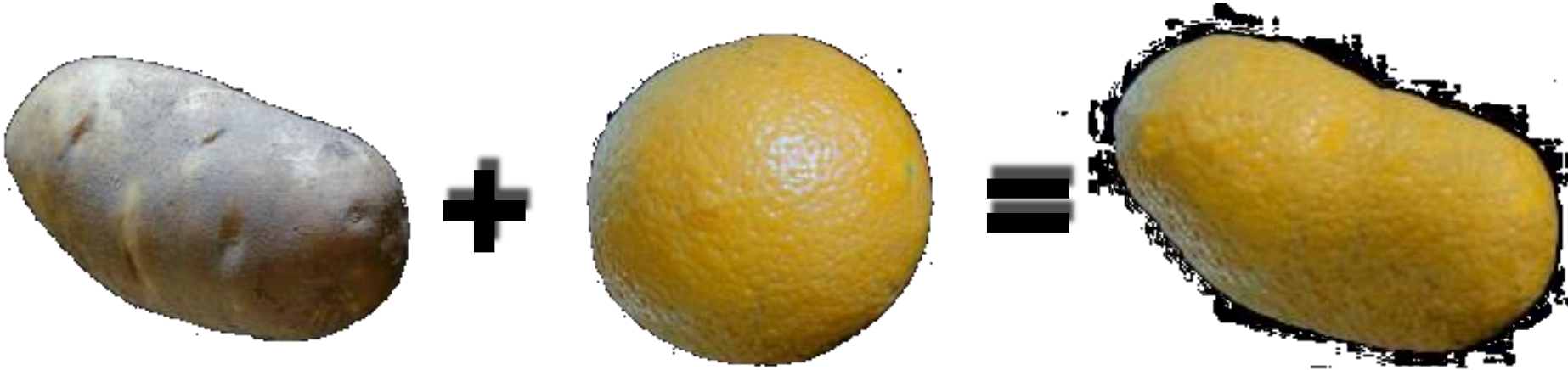
Bush campaign digitally altered TV ad

President Bush's campaign acknowledged Thursday that it had digitally altered a photo that appeared in a national cable television commercial. In the photo, a handful of soldiers were multiplied many times.



Texture Transfer

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



Texture Transfer



Constraint

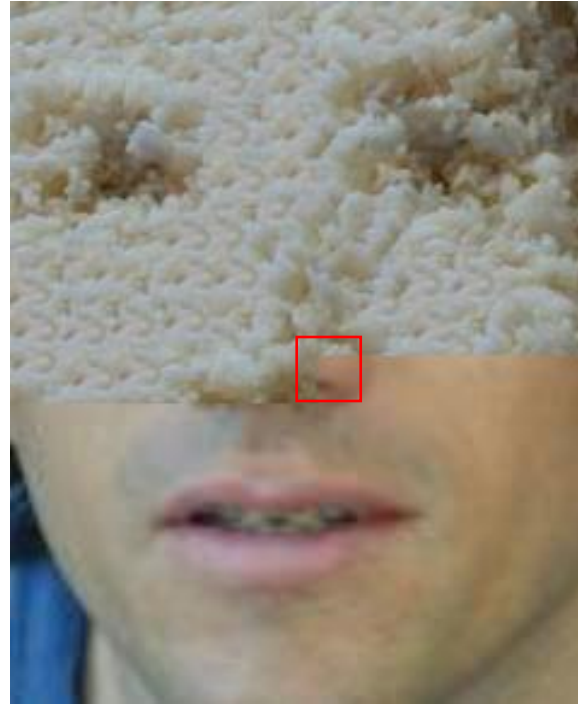


Texture sample



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. Patches from texture should correspond to patches from constraint in some way. Typical example: blur luminance, use SSD for distance

Correspondence maps

- Correspondence maps guide which patches from source are copied into texture
 - Cost to copy a patch is $\alpha * SSD_{overlap} + (1 - \alpha) * SSD_{transfer}$
 - $SSD_{overlap}$: sum sq dist of overlapping portion of patch with filled target image
 - $SSD_{transfer}$: sum sq dist of correspondence map in target and source
- Correspondence map typically is blurred grayscale version of original source and target
 - Want low frequency intensities to match but not details or color



source texture



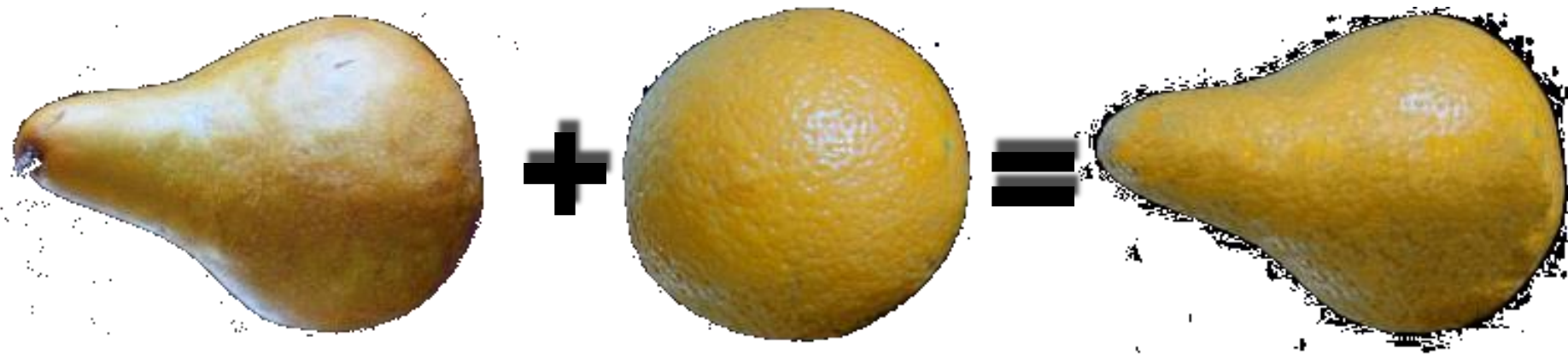
target image

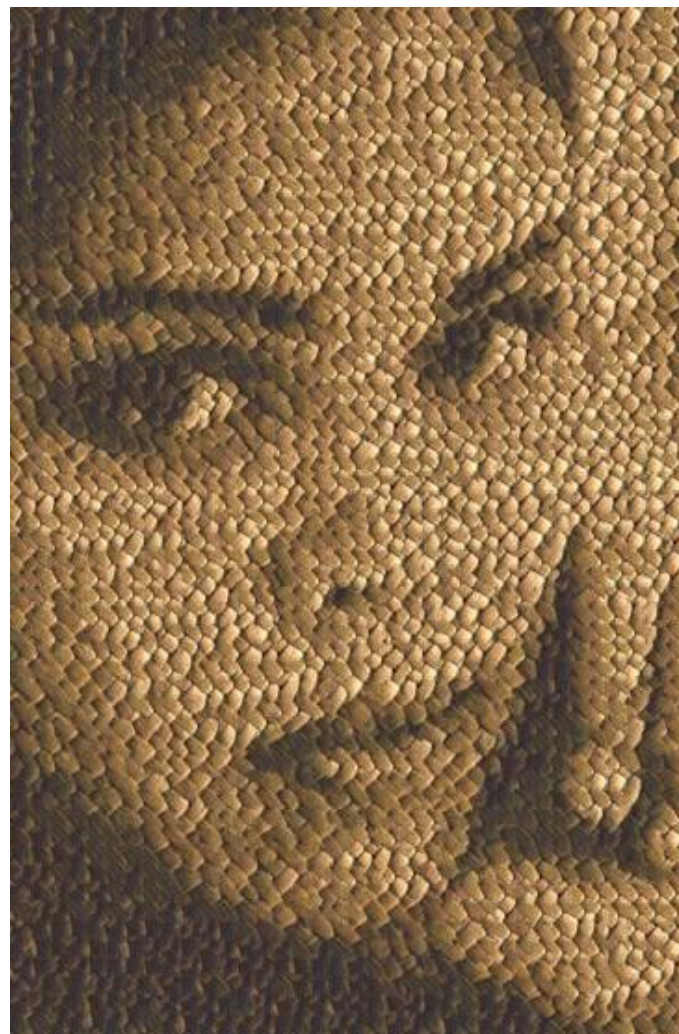


correspondence maps

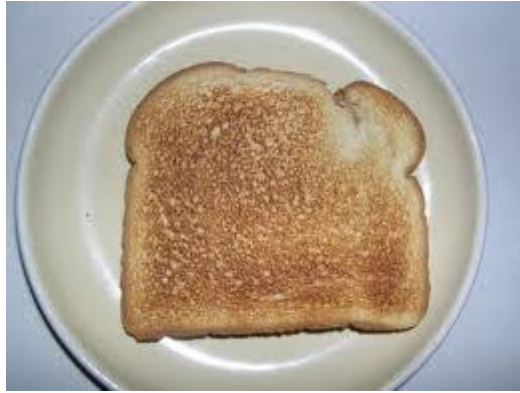


texture transfer result

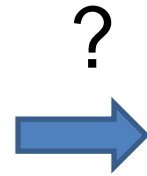




Making sacred toast



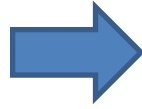
+



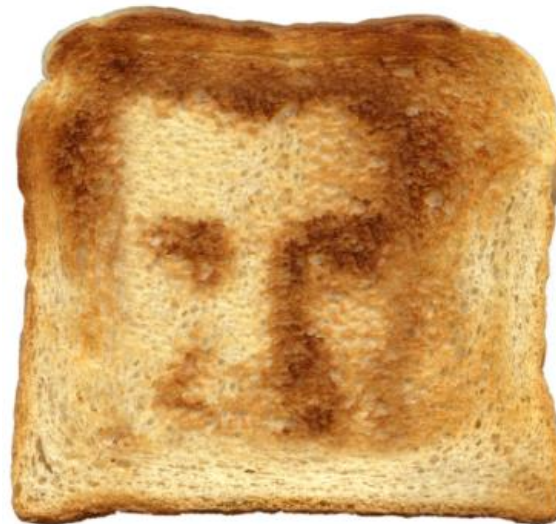
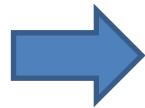
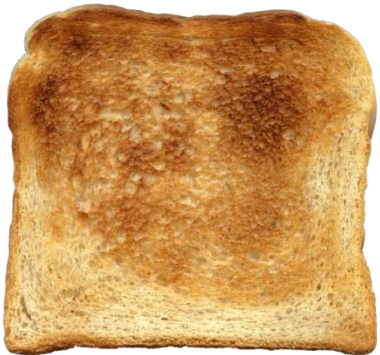
Project 2: texture synthesis and transfer

- https://courses.engr.illinois.edu/cs445/fa2022/projects/quilting/ComputationalPhotography_ProjectQuilting.html
- Note: this is significantly more challenging than the first project

ut it becomes harder to lau
ound itself, at "this daily
ving rooms," as House Der
cribed it last fall. He fai
at he left a ringing questi
ore years of Monica Lewit
inda Tripp?" That now see
Political comedian Al Fra
ext phase of the story will



und itself, at this it becomes harder itself, at this o
ing rooms," as Hound itself, at "thisrooms," as Hous
cribed it last fall ing rooms," as Hooded it last fall. H
he left a ringing quibed it last fall. left a ringing que
re years of Monica le left a ringing years of Monica I
da Tripp?" That noe years of Monic Tripp?" That now
olitical comedian ida Tripp?" That ntical comedian Al
ns," as Hoitself, at "this dre years of Monicaelf, at "
t last fal rooms," as Housda Tripp?" That norms," as
a ringing ed it last fall. He itical comedian At last fa
of Monicelf a ringing ques "this dairooms," as Hous
p?" That rears of Monica Las Houseibed it last fall. F
comes hardins daiborns," as fall. He left a ringing qu
tself, at "tHouse ed it last fall. He years of Monica l
orns," as fall. He fft a ringing questTripp?" That no
d it last faze years of Monica vica Les of Monicdiangir
ft a ringinda Tripp?" That nat now so?" That s of Mor
zs of Moolitical comediardian Al Fcomediapp?" Tha



Texture Synthesis and Transfer Recap



For each overlapping patch in the output image

1. Compute the cost to each patch in the sample
 - Texture synthesis: this cost is the SSD (sum of square difference) of pixel values in the overlapping portion of the existing output and sample
 - Texture transfer: cost is $\alpha * SSD_{overlap} + (1 - \alpha) * SSD_{transfer}$. The latter term enforces that the source and target correspondence patches should match.
2. Select one sample patch that has a small cost (e.g. randomly pick one of K candidates)
3. Find a cut through the left/top borders of the patch based on overlapping region with existing output
 - Use this cut to create a mask that specifies which pixels to copy from sample patch
4. Copy masked pixels from sample image to corresponding pixel locations in output image

PatchMatch

- Efros & Leung synthesis is very slow: for every pixel to be filled, match surrounding patch across the source image
- “Image Quilting” is faster by copying a patch at a time and blending it into the output
 - Typically neighboring pixels in source should stay together, so copy them patch by patch
- PatchMatch solves this in a more efficient and general way

PatchMatch

- Goal: Solve for labels that minimize some cost function



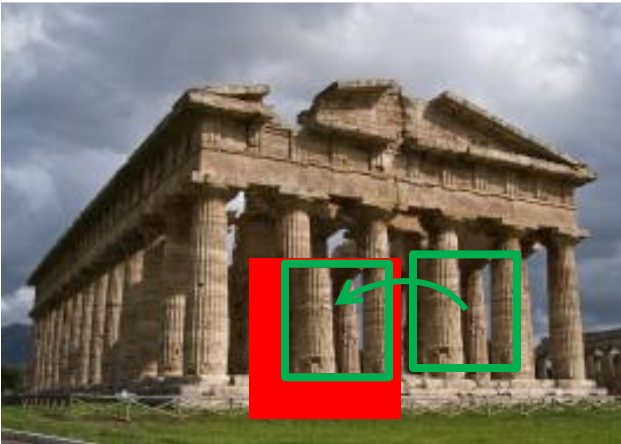
Hole Fill-in:

For each pixel (i,j) in the hole, solve for pixel coordinate (n,m) from source to minimize sum of SSD of patches between target and source

How to solve for this efficiently?

PatchMatch

- Goal: Solve for labels that minimize some cost function



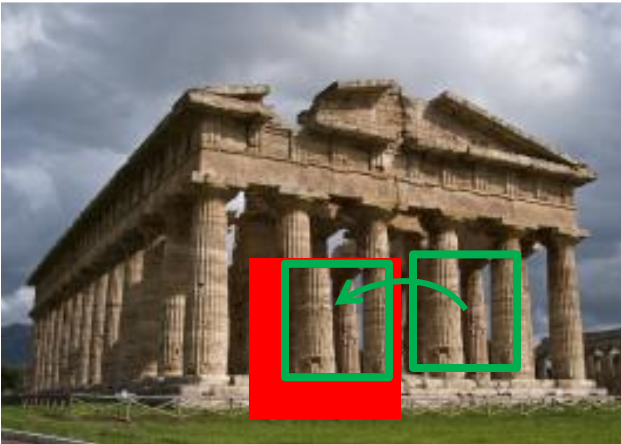
Hole Fill-in:

For each pixel (i,j) in the hole, solve for **pixel coordinate (n,m)**
offset $(a,b)=(n,m)-(i,j)$ from source to minimize sum of SSD of patches between target and source

When copying a patch, offset is piecewise constant

PatchMatch

- Goal: Solve for labels that minimize some cost function
- Key assumption: a pixel and its neighbor very likely have the same or similar labels



Hole Fill-in:

For each pixel (i,j) in the hole, solve for **offset** $(a,b)=(n,m)-(i,j)$ from source to minimize sum of SSD of patches between target and source

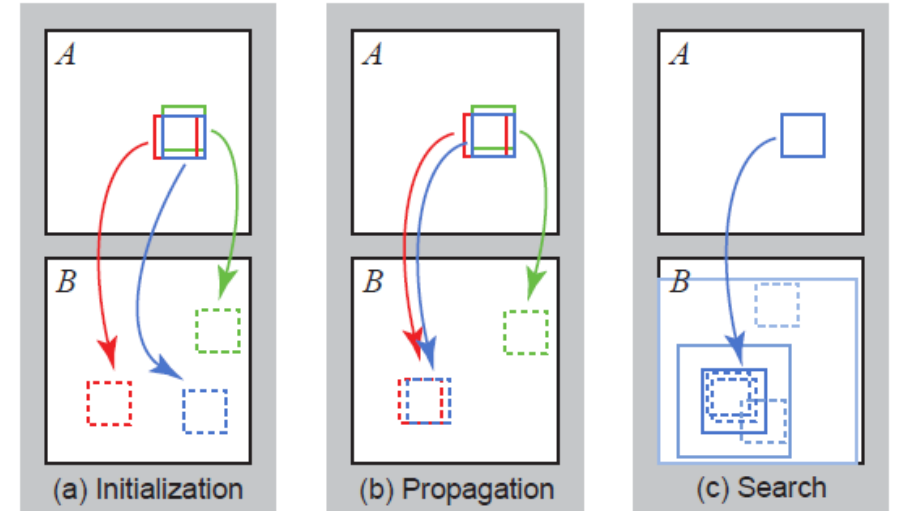
When copying a patch, offset is piecewise constant

PatchMatch: Optimization

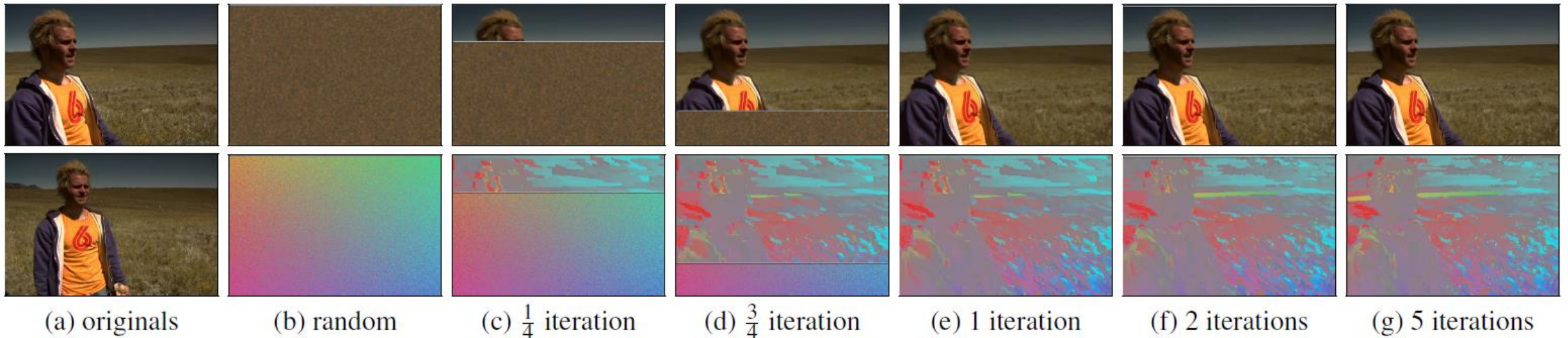
- Goal: Solve for labels that minimize some cost function
- Key assumption: a pixel and its neighbor very likely have the same or similar labels

PatchMatch Algorithm basics

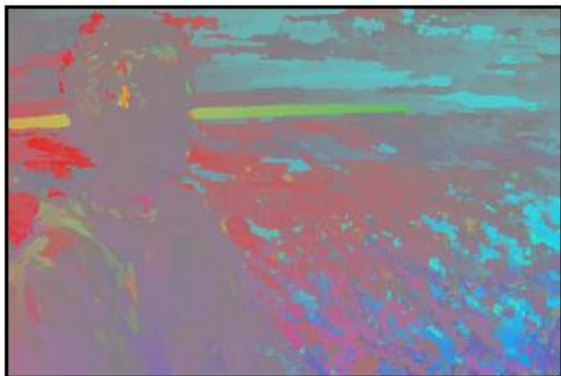
1. Randomly initialize matches
2. Scan across image (forward and backward)
 - a. Check if neighbor's offsets or random perturbations around current offset produce better scores
 - b. Keep best found so far
3. Repeat (2) several times



PatchMatch: Convergence



Example of convergence with retargeting (find offsets to map bottom image onto top)



Why so fast?

- Offset has constant or similar values in large regions
- Very good chance that at least one pixel gets lucky in random assignment
- Good assignments propagate quickly

PatchMatch: Image Completion

Guides constrain search



(a) original

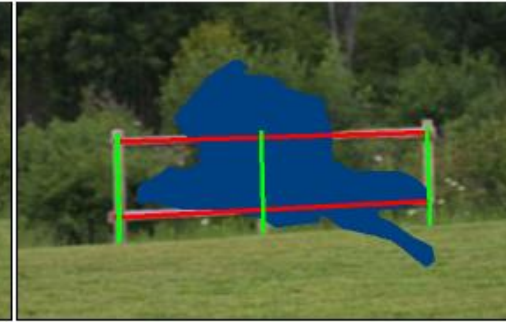
(b) hole+constraints



(c) hole filled



(a) input



(b) hole and guides



(c) completion result



(d) input



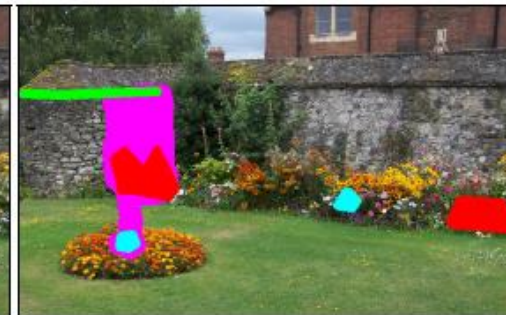
(e) hole



(f) completion (close up)



(g) same input



(h) hole and guides



(i) guided (close up)

PatchMatch: retargeting

- Produce output image of target size with optional constraints
- Bi-directional matching: each patch in source should match something in target and vice-versa



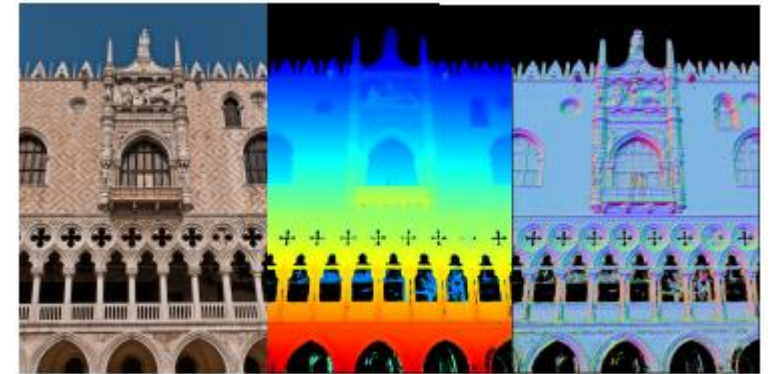
PatchMatch: other applications and extensions

Applications

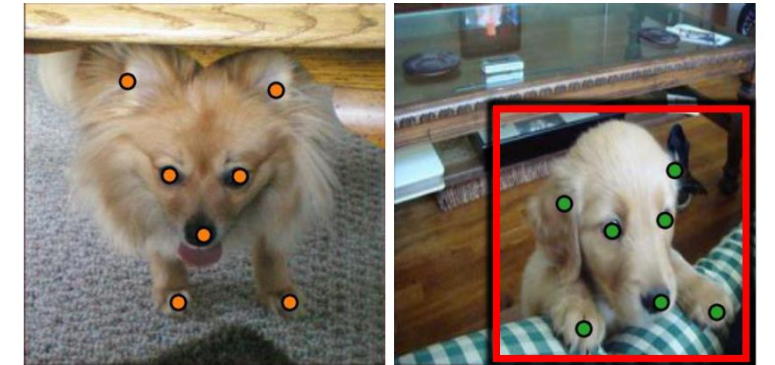
- Two-view stereo: labels are displacements
- Multiview stereo: labels are plane parameters
- Semantic correspondence: labels are offsets
- Denoising, symmetry detection, ...

Extensions

- Red/black propagation for efficient GPU implementation
- Varying propagation/scoring schemes

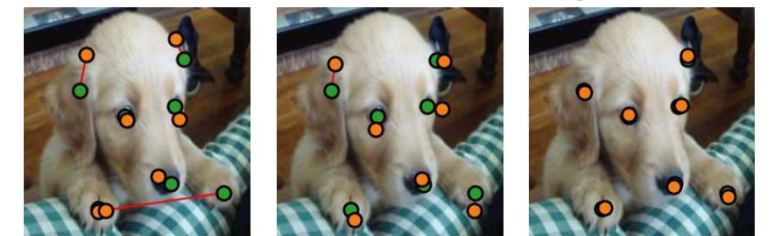


Schoenberger et al. ECCV 2016



Source

Target



NC-Net

ANC-Net

PMNC (Ours)

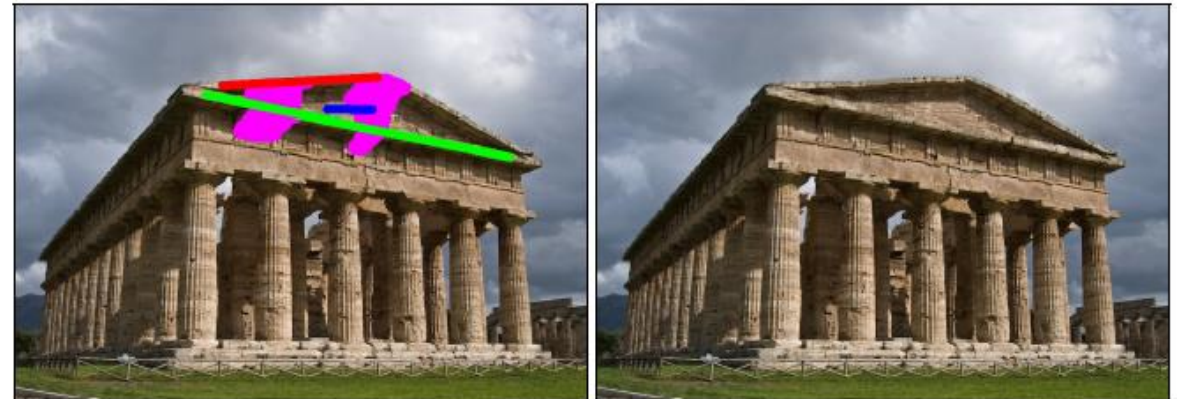
Lee et al. CVPR 2021

Things to remember

- Texture synthesis and hole-filling can be thought of as a form of probabilistic hallucination
- Simple, similarity-based matching is a powerful tool
 - Synthesis
 - Hole-filling
 - Retargeting
 - And much more...



- Key is how to define similarity and efficiently find neighbors
- PatchMatch provides flexible and highly efficient optimization



Next class

- Cutting and seam finding