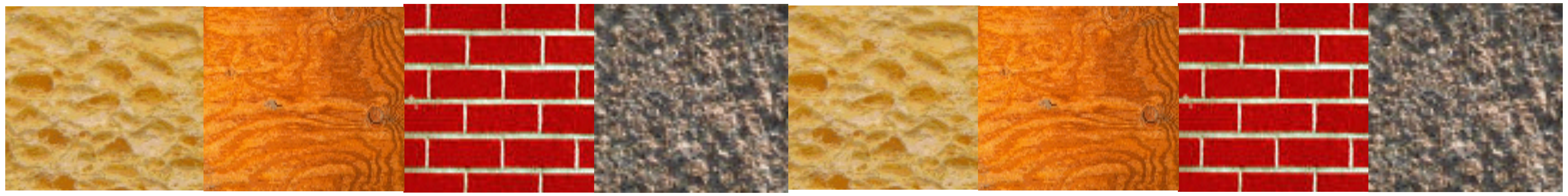
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 11: Texture (cont.)

(unless otherwise stated slides are taken or adopted from **Bob Woodham, Jim Little** and **Fred Tung**)

Menu for Today (**February 11, 2020**)

Topics:

— Texture Analysis

— Texture Synthesis

Readings:

— **Today's** Lecture: Forsyth & Ponce (2nd ed.) 3.1-3.3

— **Next** Lecture:

Reminders:

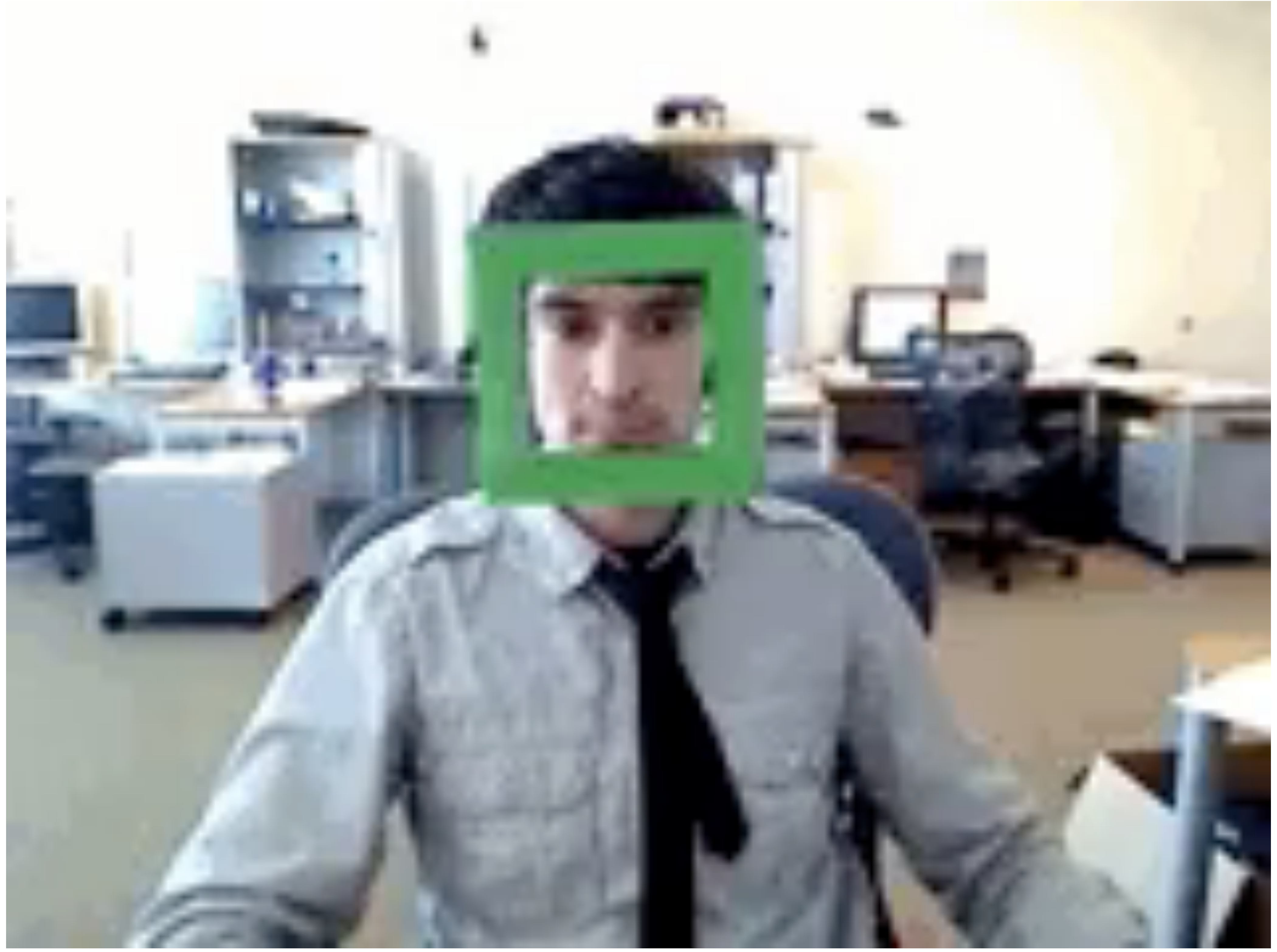
— **Assignment 2**: Face Detection in a Scaled Representation is **due today**

— **Quiz** grades (Canvas) and **quizzes** (Piazza) have been posted

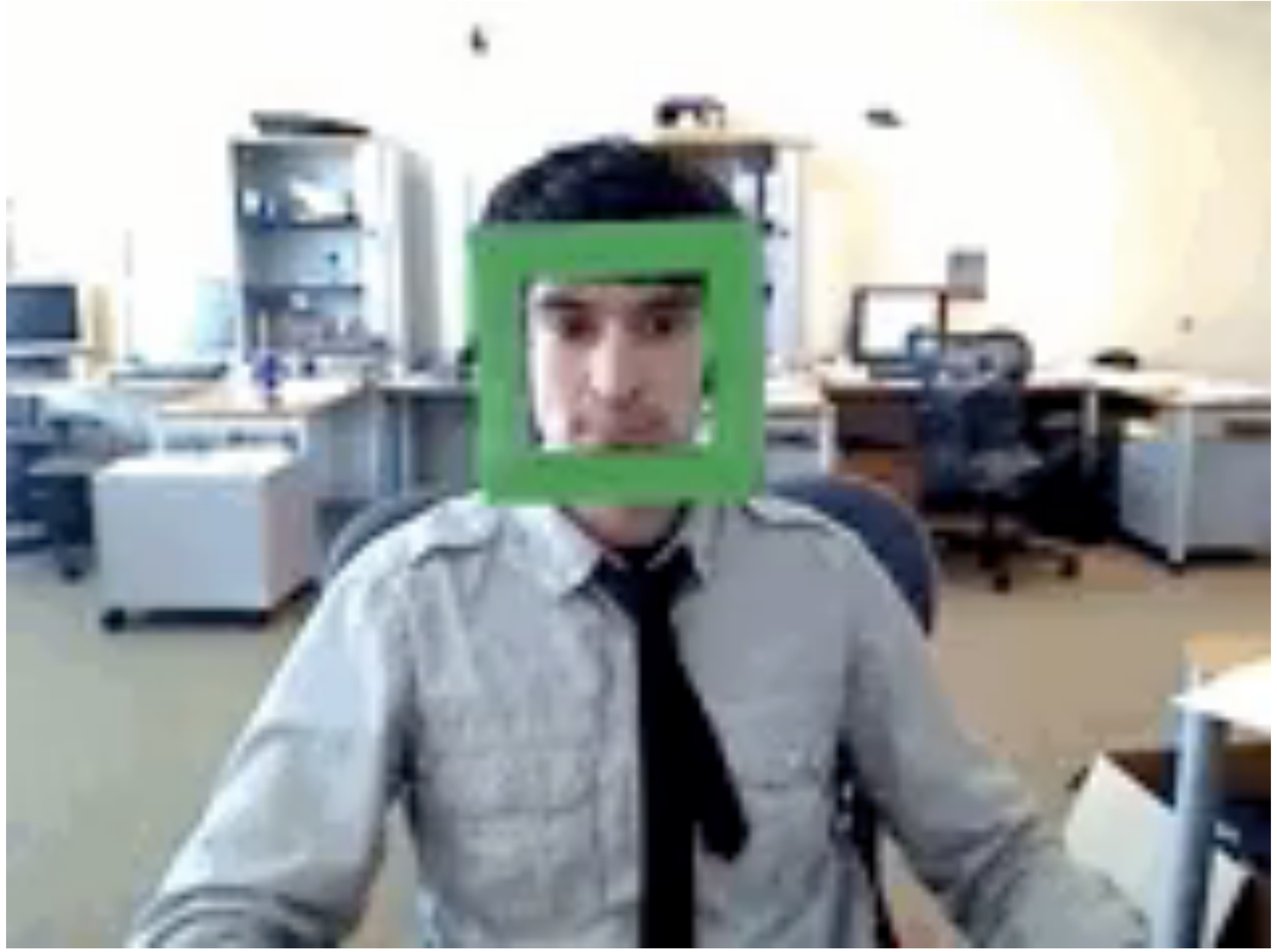
— **Assignment 3** will be posted by the **end of today**

— **Assignment 1** will be graded this week

Today's "**fun**" Example: Face Detection



Today's "**fun**" Example: Face Detection



Today's "**fun**" Example: Face Detection

Today's "**fun**" Example: Face Detection

Today's "**fun**" Example: Face Detection

https://www.youtube.com/watch?v=gWjBleSfZBk

Today's "**fun**" Example: Face Detection

https://www.youtube.com/watch?v=gWjBleSfZBk

Lecture 10: Re-cap of Texture

(Functional) **Definition**:

Texture is detail in an image that is at a scale too small to be resolved into its constituent elements and at a scale large enough to be apparent in the spatial distribution of image measurements

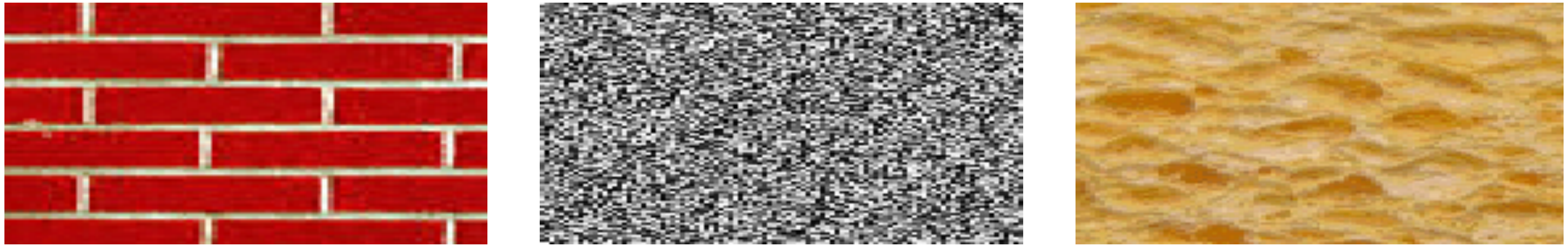


Figure Credit: Alexei Efros and Thomas Leung

Lecture 10: Re-cap of Texture

Texture can be a strong cue to **object identity** if the object has distinctive material properties

Texture can be a strong cue to an **object's shape** based on the deformation of the texture from point to point.

— Estimating surface orientation or shape from texture is known as "**shape from texture**"

Lecture 10: Re-cap of Texture

We will look at two main questions:

1. How do we represent texture?
   → Texture **analysis**

2. How do we generate new examples of a texture?
   → Texture **synthesis**

Lecture 10: Texture Representation

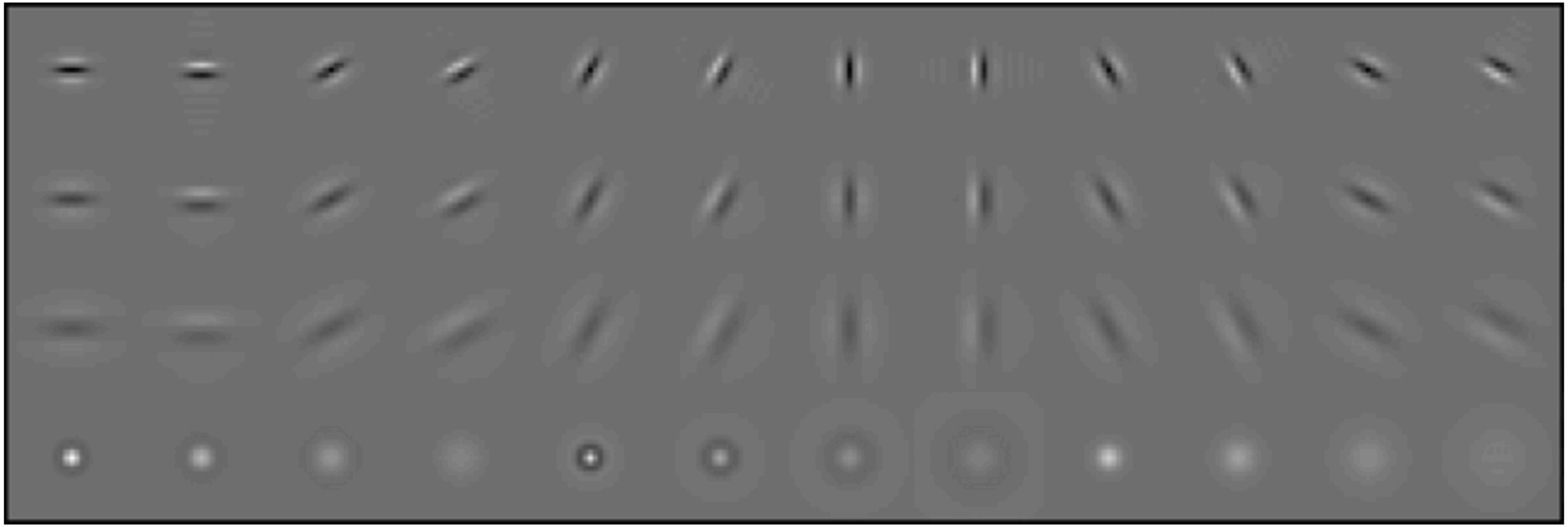
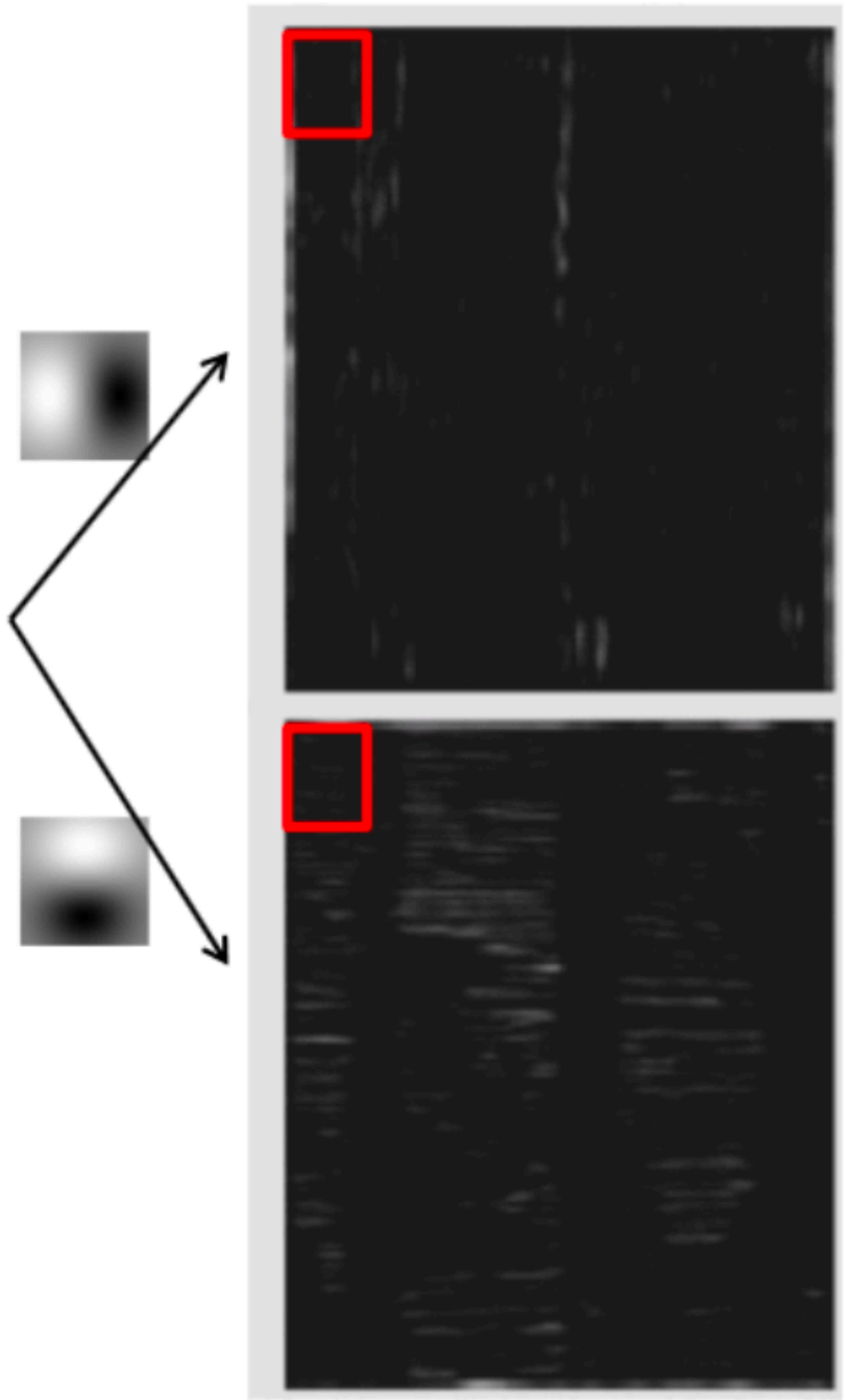


Figure Credit: Leung and Malik, 2001

Lecture 10: Texture Representation

original image



derivative filter responses, squared

	mean d/dx value	mean d/dy value
Win. #1	4	10

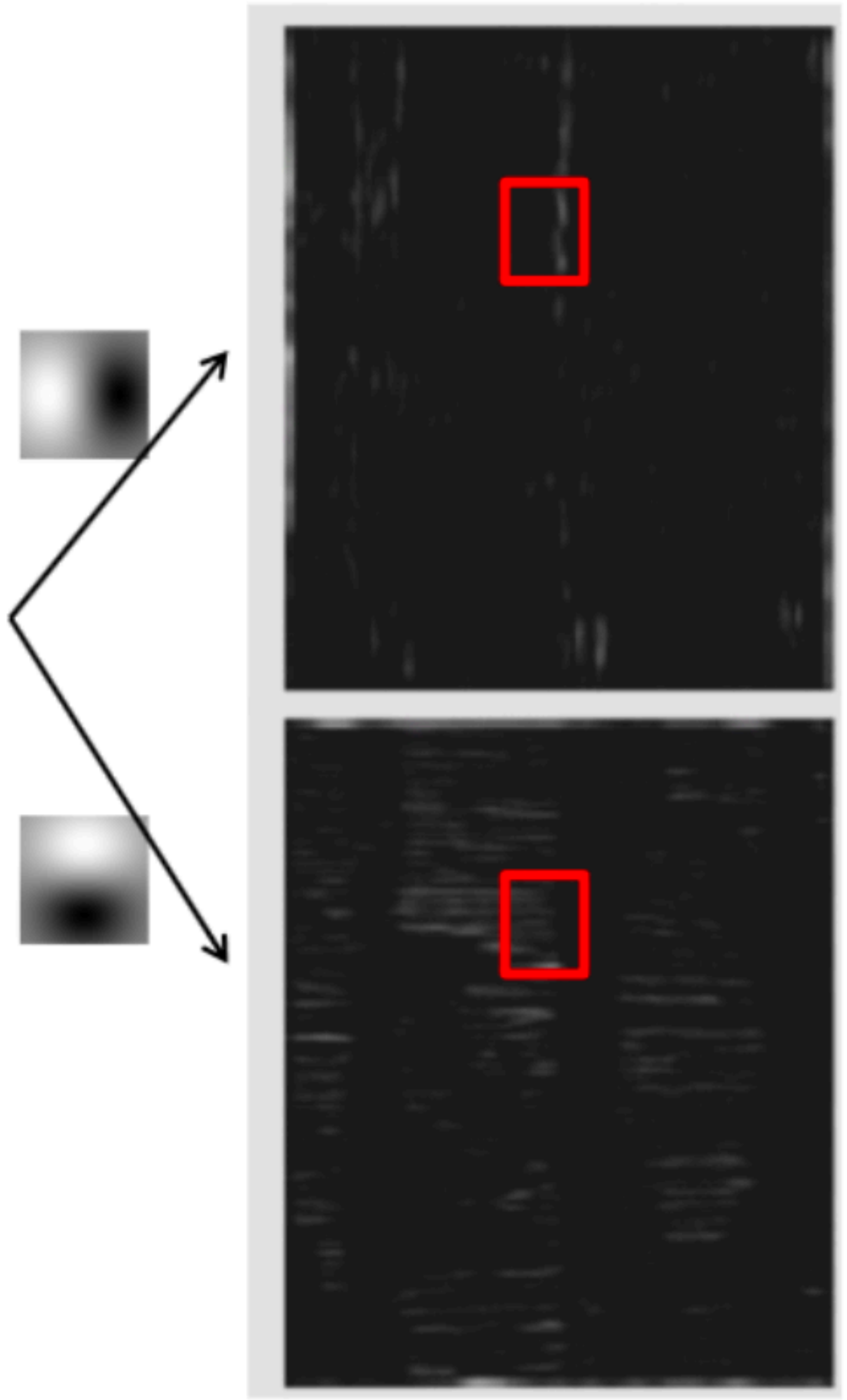
⋮

statistics to summarize patterns in small windows

Slide Credit: Trevor Darrell

Lecture 10: Texture Representation

original image



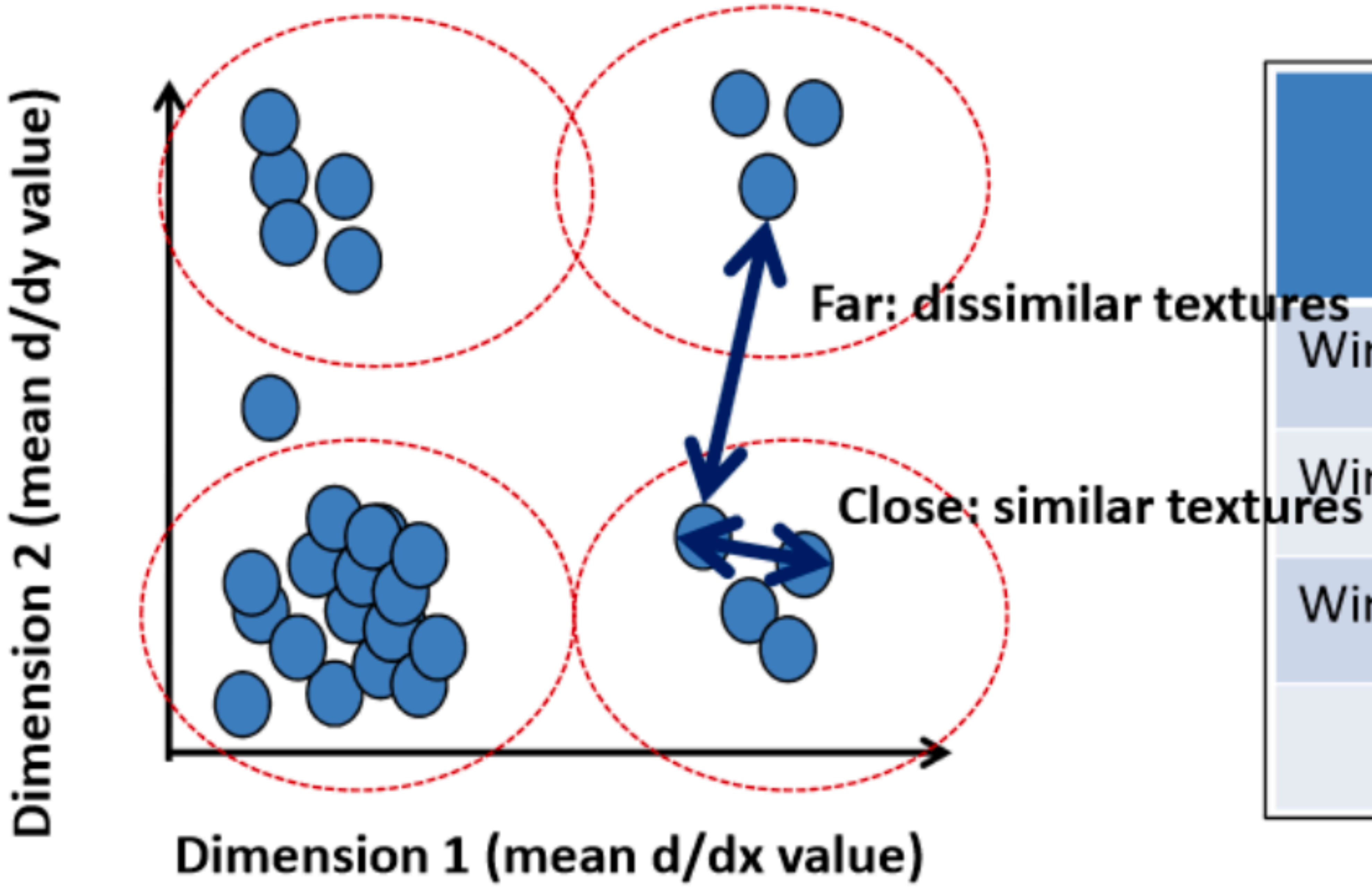
derivative filter responses, squared

	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

⋮

statistics to summarize patterns in small windows

Slide Credit: Trevor Darrell



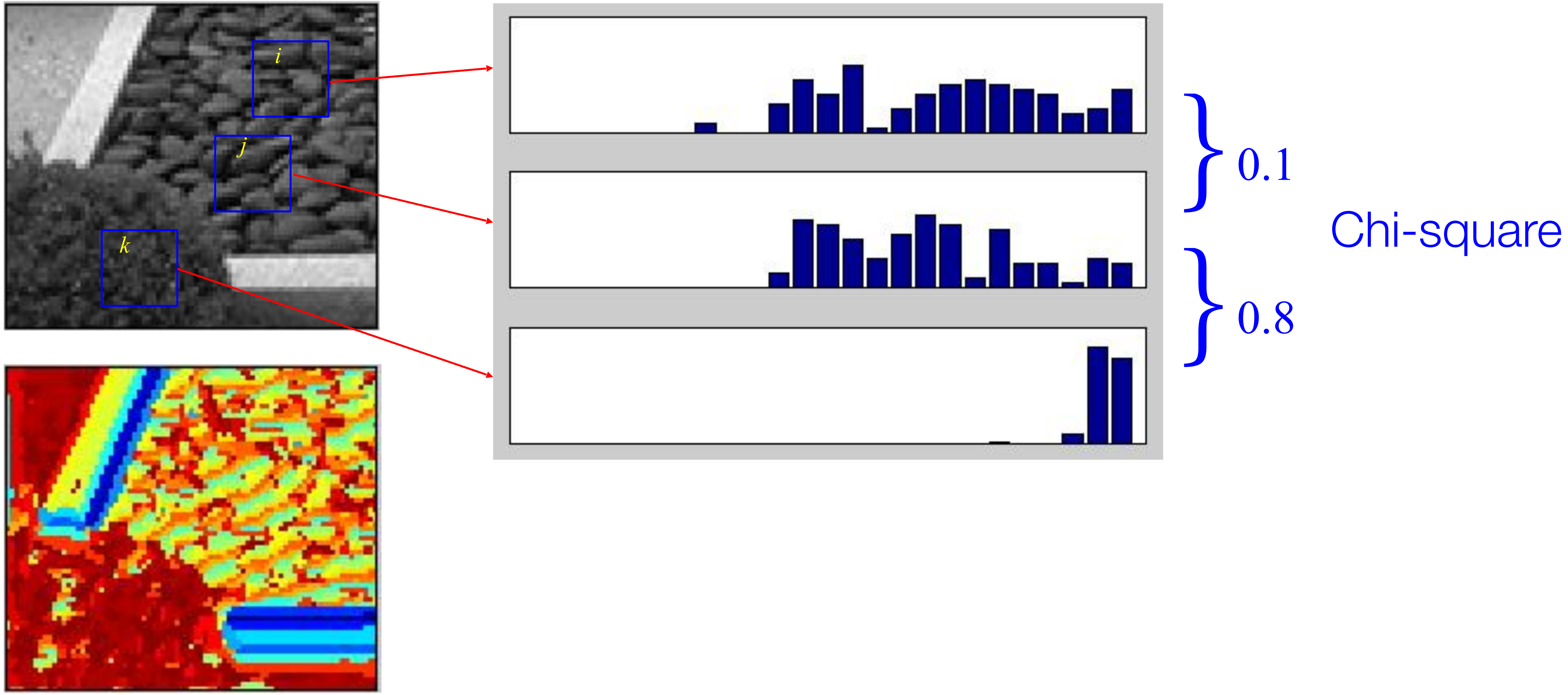
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

⋮

statistics to summarize patterns in small windows

Slide Credit: Trevor Darrell

Lecture 10: Texture Representation



Bag-of-Words Representation

Take a large **corpus of text**:

Bag-of-Words Representation

Take a large **corpus of text**:

— Represent every **letter** by a 26 dimensional (unit) vector

$$a = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 0 \end{bmatrix} \qquad b = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 0 \end{bmatrix}$$

Bag-of-Words Representation

Take a large **corpus of text**:

— Represent every **letter** by a 26 dimensional (unit) vector

— Represent each **word** by an average of letter representations in it

$$ab = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 0 \end{bmatrix}$$

Bag-of-Words Representation

Take a large **corpus of text**:

— Represent every **letter** by a 26 dimensional (unit) vector

— Represent each **word** by an average of letter representations in it

— Cluster the words, to get a "**dictionary**". Words that have very similar representations would get clustered together (e.g., smile and smiled)

Bag-of-Words Representation

Take a large **corpus of text**:

— Represent every **letter** by a 26 dimensional (unit) vector

— Represent each **word** by an average of letter representations in it

— Cluster the words, to get a "**dictionary**". Words that have very similar representations would get clustered together (e.g., smile and smiled)

— Now represent every document by **histogram** of "dictionary" atoms by associating every word to an atom that is closest in terms of distance in 26D

Bag-of-Words Representation

Take a large **corpus of text**:

— Represent every **letter** by a 26 dimensional (unit) vector

— Represent each **word** by an average of letter representations in it

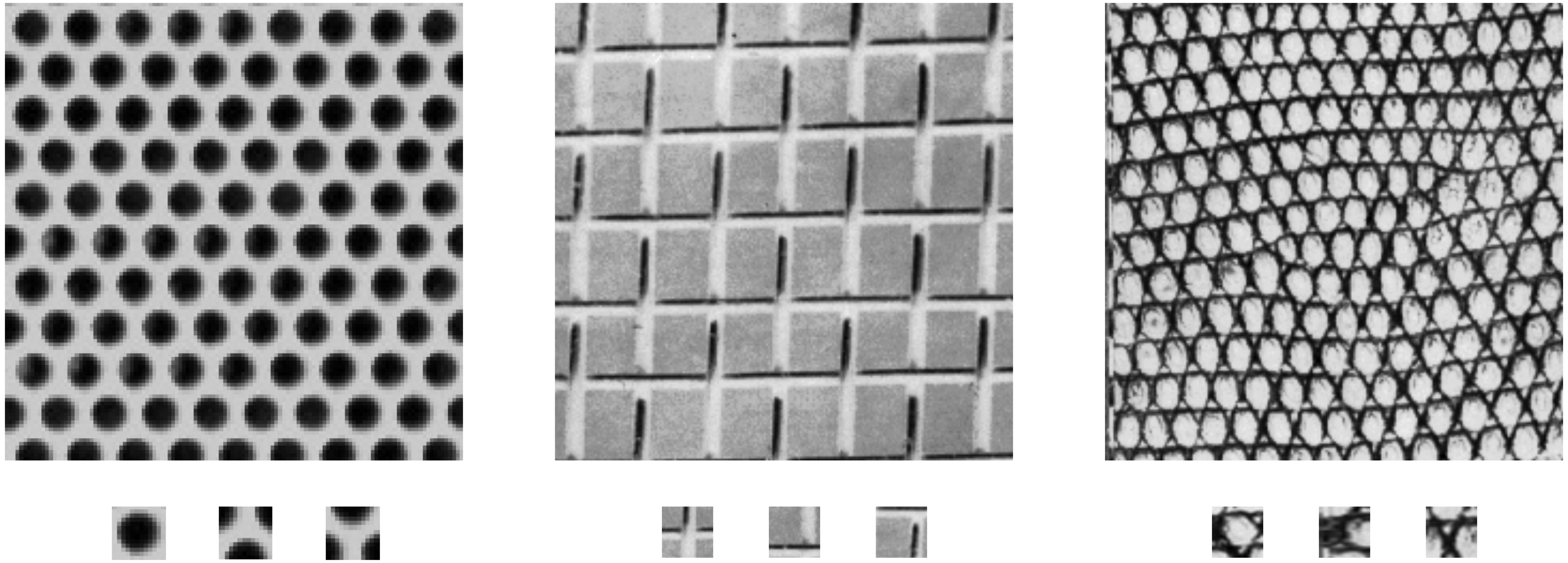
— Cluster the words, to get a "**dictionary**". Words that have very similar representations would get clustered together (e.g., smile and smiled)

— Now represent every document by **histogram** of "dictionary" atoms by associating every word to an atom that is closest in terms of distance in 26D

corpus of text = collection of images
letter = pixel location
word = patch with pixel in the center
dictionary = textons

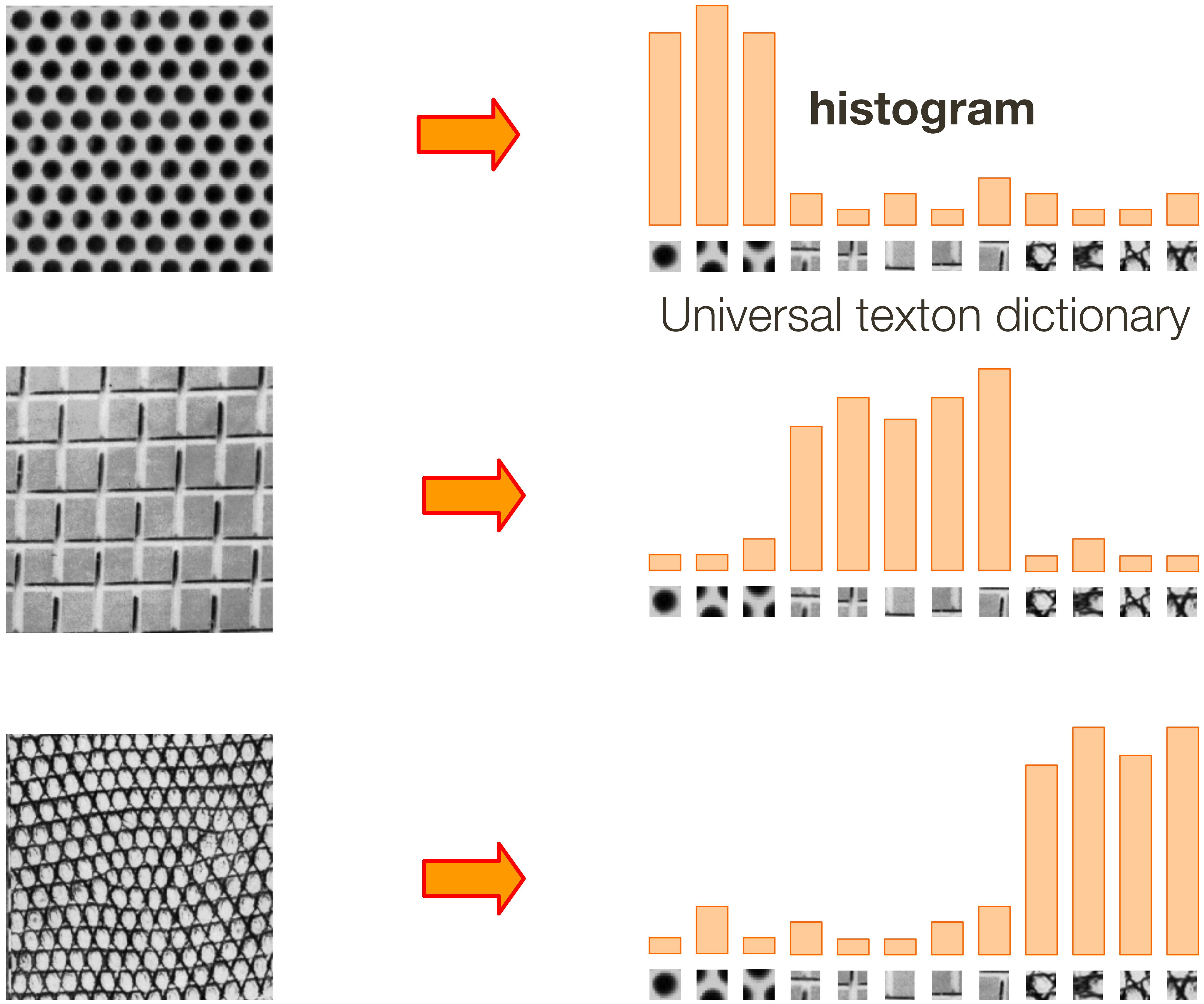
Texture representation and recognition

- Texture is characterized by the repetition of basic elements or **textons**
- For stochastic textures, it is the **identity of the textons**, not their spatial arrangement, that matters

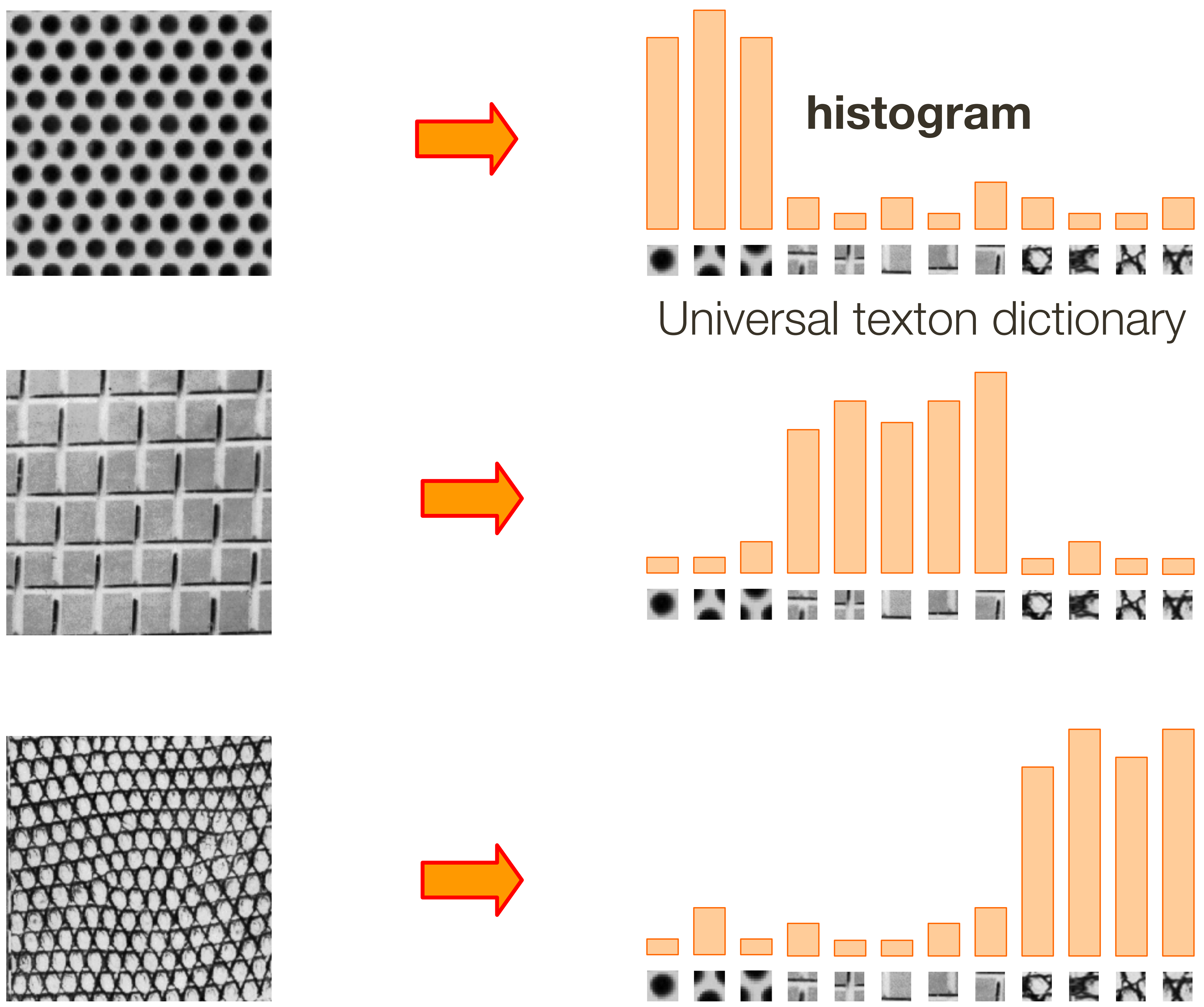


Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Texture representation and recognition



Texture representation and recognition



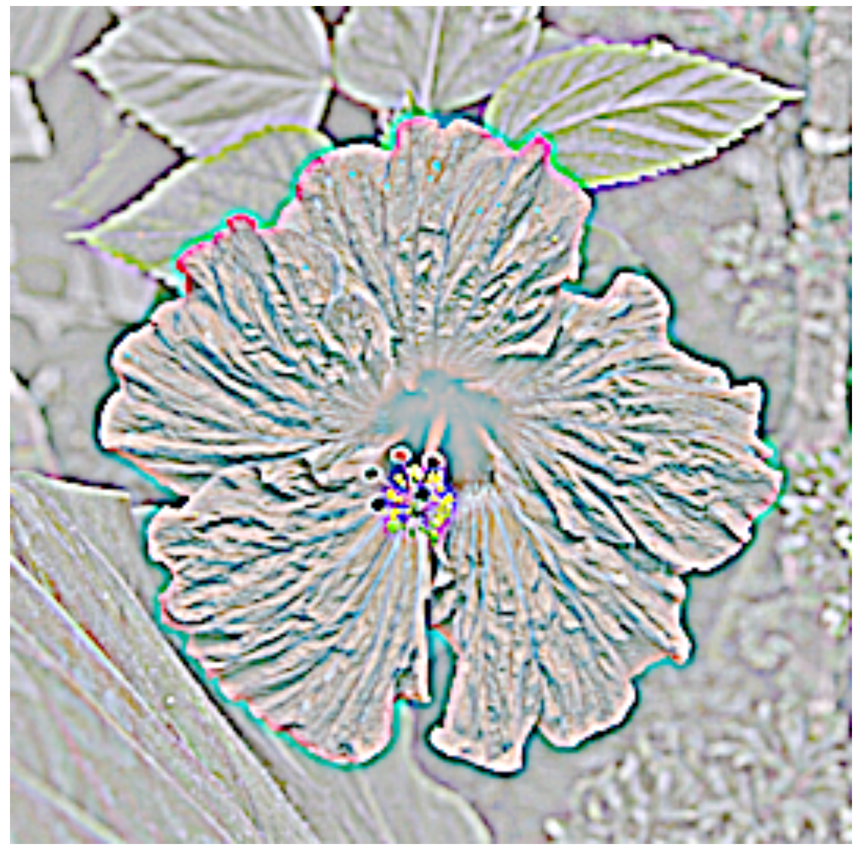
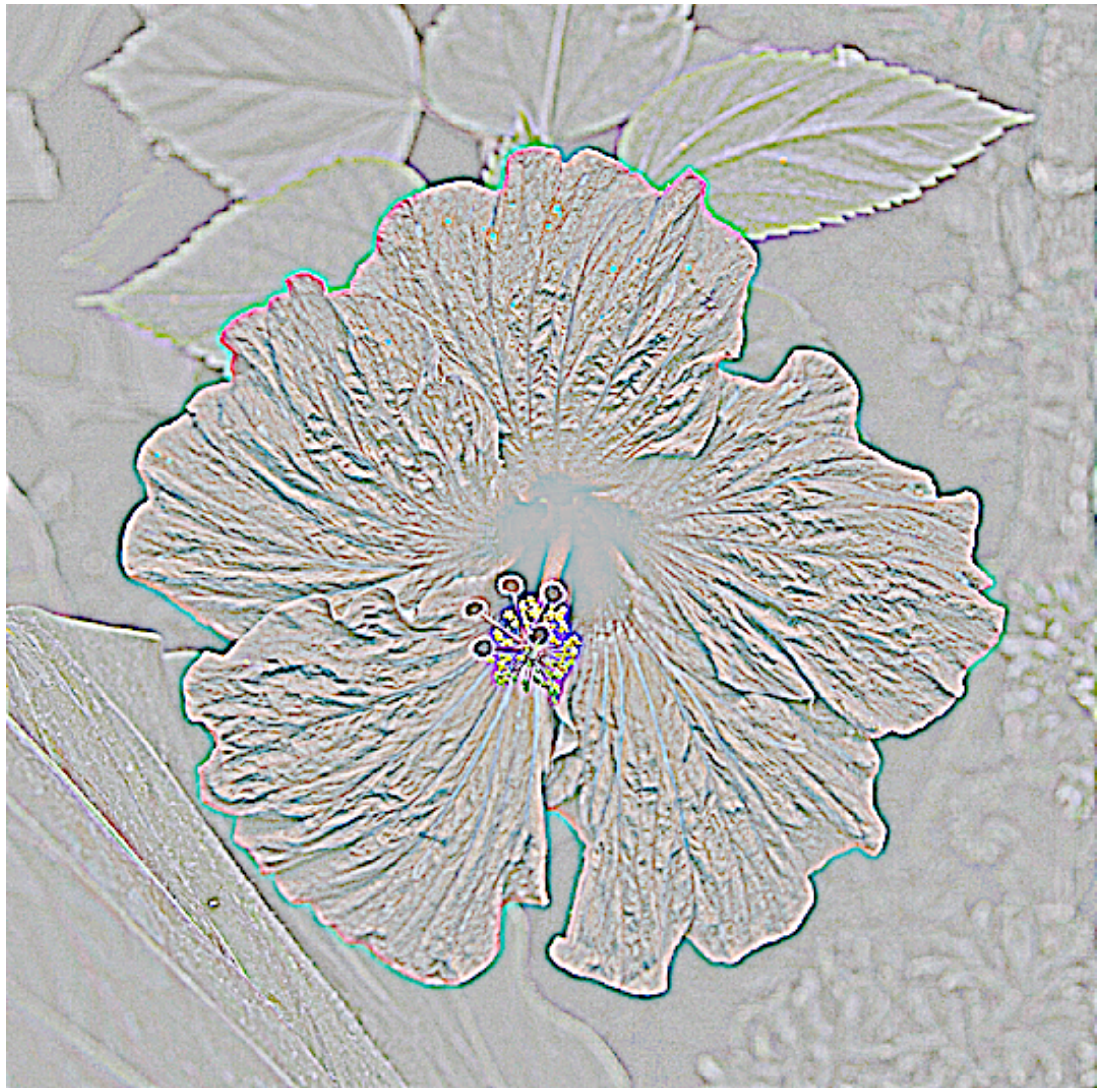
Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Lecture 10: Re-cap — Laplacian vs. Gaussian Pyramids




Shown in opposite order for space

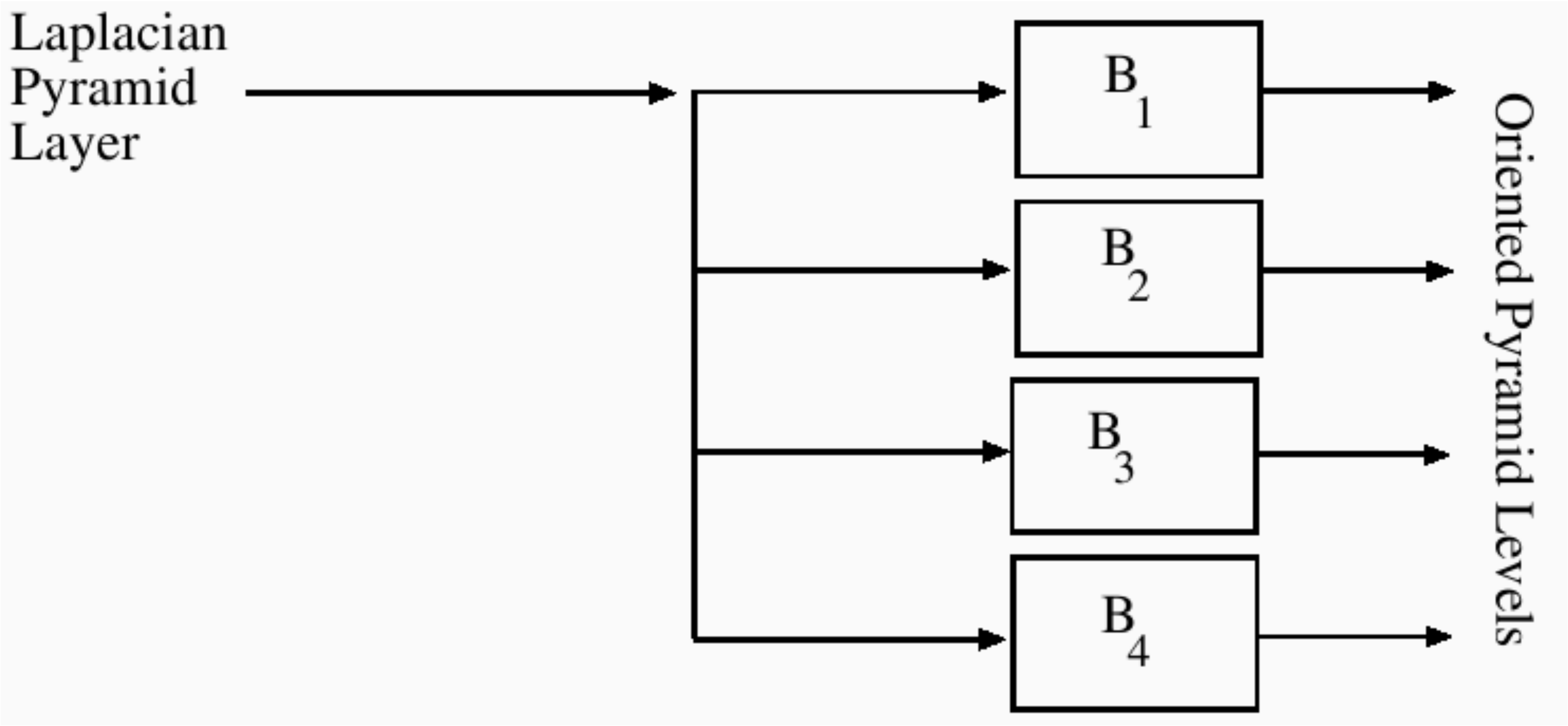


Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

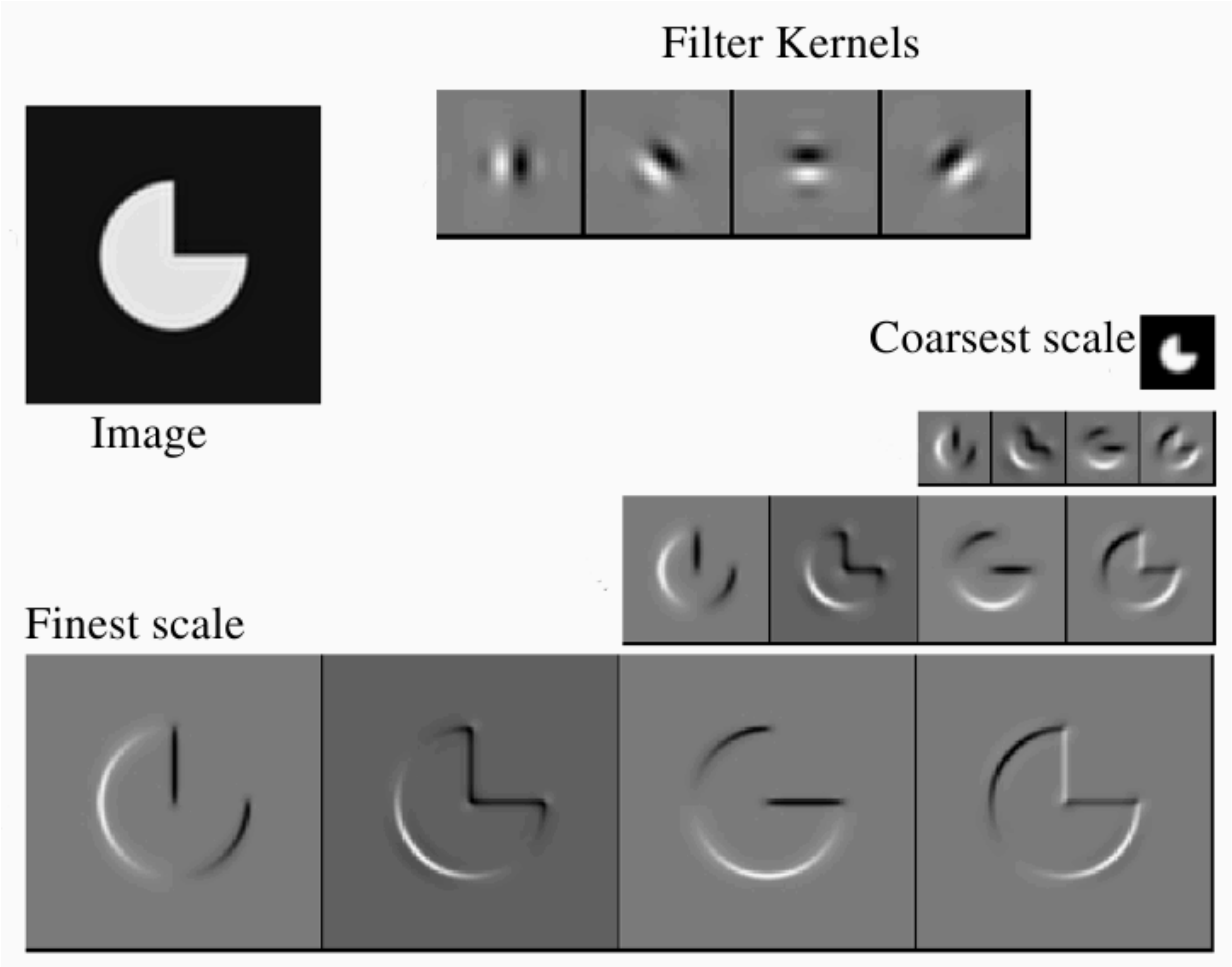
Lecture 10: Re-cap — Oriented Pyramids

Oriental Filters



Forsyth & Ponce (1st ed.) Figure 9.14

Lecture 10: Re-cap — Oriented Pyramids



Forsyth & Ponce (1st ed.) Figure 9.13

Texture

We will look at two main questions:

1. How do we represent texture?
   → Texture **analysis**

2. How do we generate new examples of a texture?
   → Texture **synthesis**

Texture **Synthesis**

Why might we want to synthesize texture?

1. To fill holes in images (**inpainting**)

— Art directors might want to remove telephone wires. Restorers might want to remove scratches or marks.

— We need to find something to put in place of the pixels that were removed

— We synthesize regions of texture that fit in and look convincing

Texture **Synthesis**

Why might we want to synthesize texture?

1. To fill holes in images (**inpainting**)

— Art directors might want to remove telephone wires. Restorers might want to remove scratches or marks.

— We need to find something to put in place of the pixels that were removed

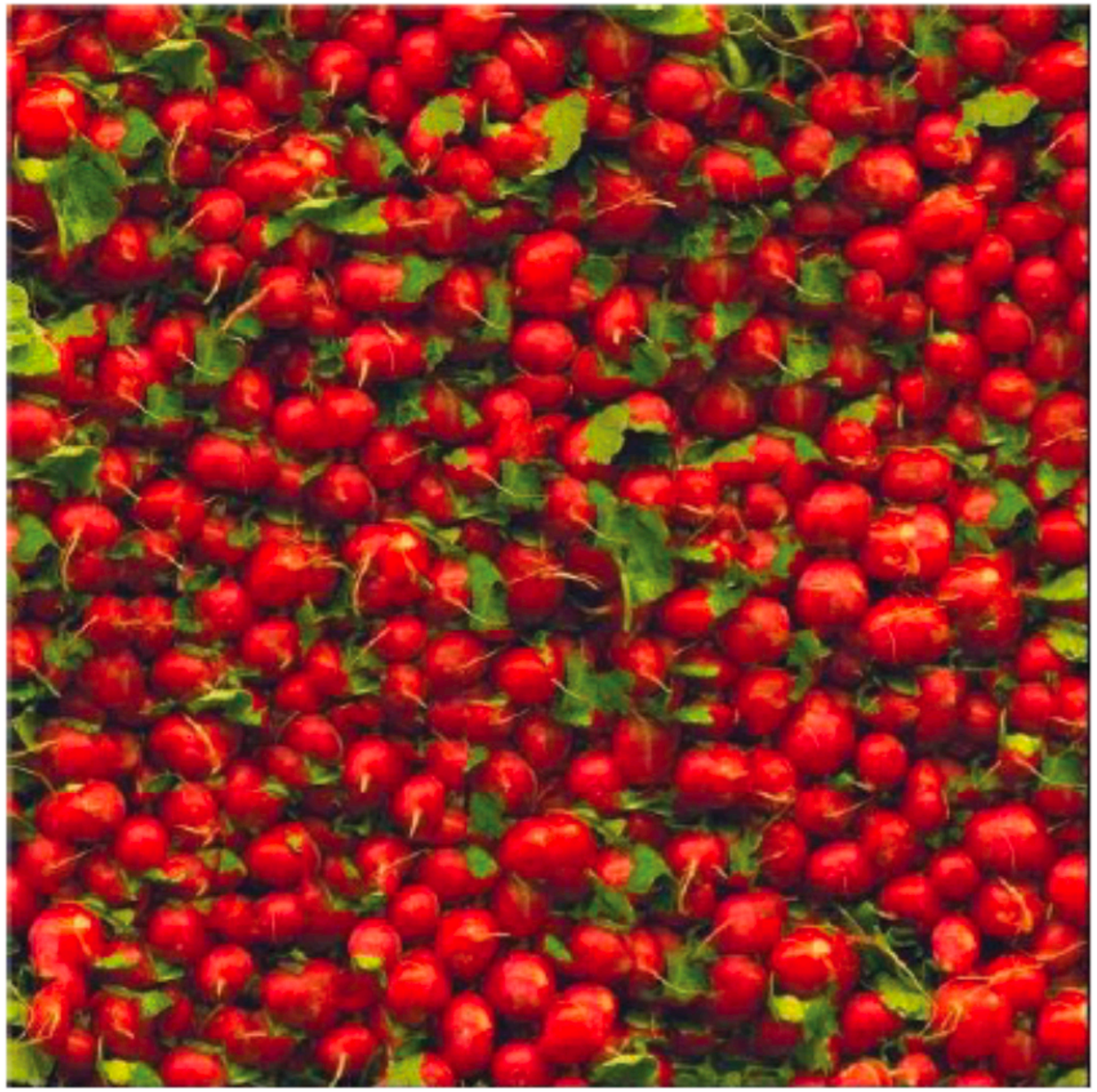
— We synthesize regions of texture that fit in and look convincing

2. To produce large quantities of texture for computer graphics

— Good textures make object models look more realistic

Texture **Synthesis**

radishes



lots more radishes

Szeliski, Fig. 10.49

Texture **Synthesis**

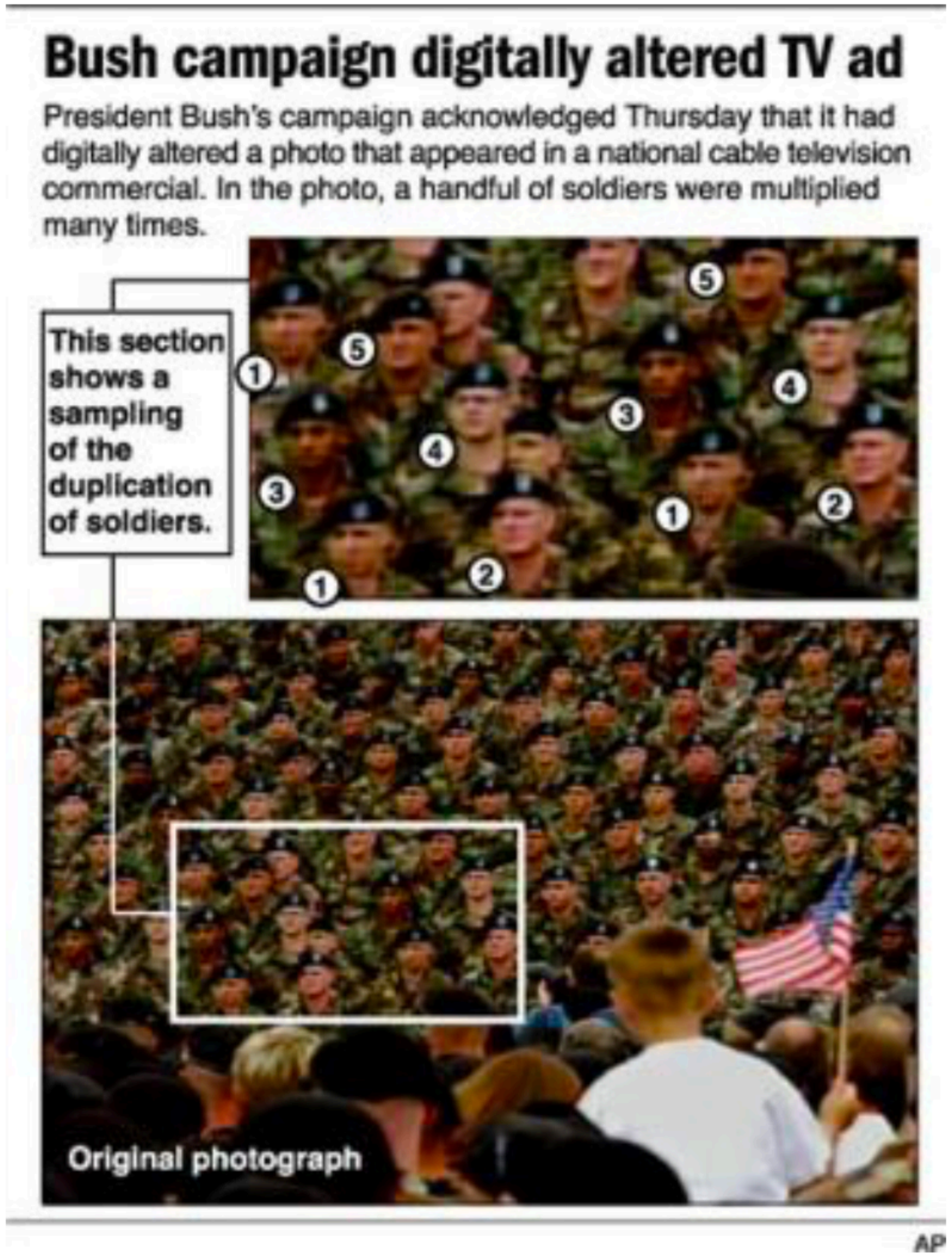


Photo Credit: Associated Pres

Texture **Synthesis**

Cover of “The Economist,” June 19, 2010

Photo Credit (right): Reuters/Larry Downing

Assignment 3 Preview: Texture Synthesis

Task: Make donkey vanish

Assignment 3 Preview: Texture Synthesis

Task: Make donkey vanish

Method: Fill-in regions using texture from the white box

Assignment 3 Preview: Texture Synthesis

Task: Make donkey vanish

Method: Fill-in regions using texture from the white box

Texture Synthesis

Objective: Generate new examples of a texture We take a “data-driven" approach

Idea: Use an image of the texture as the source of a probability model

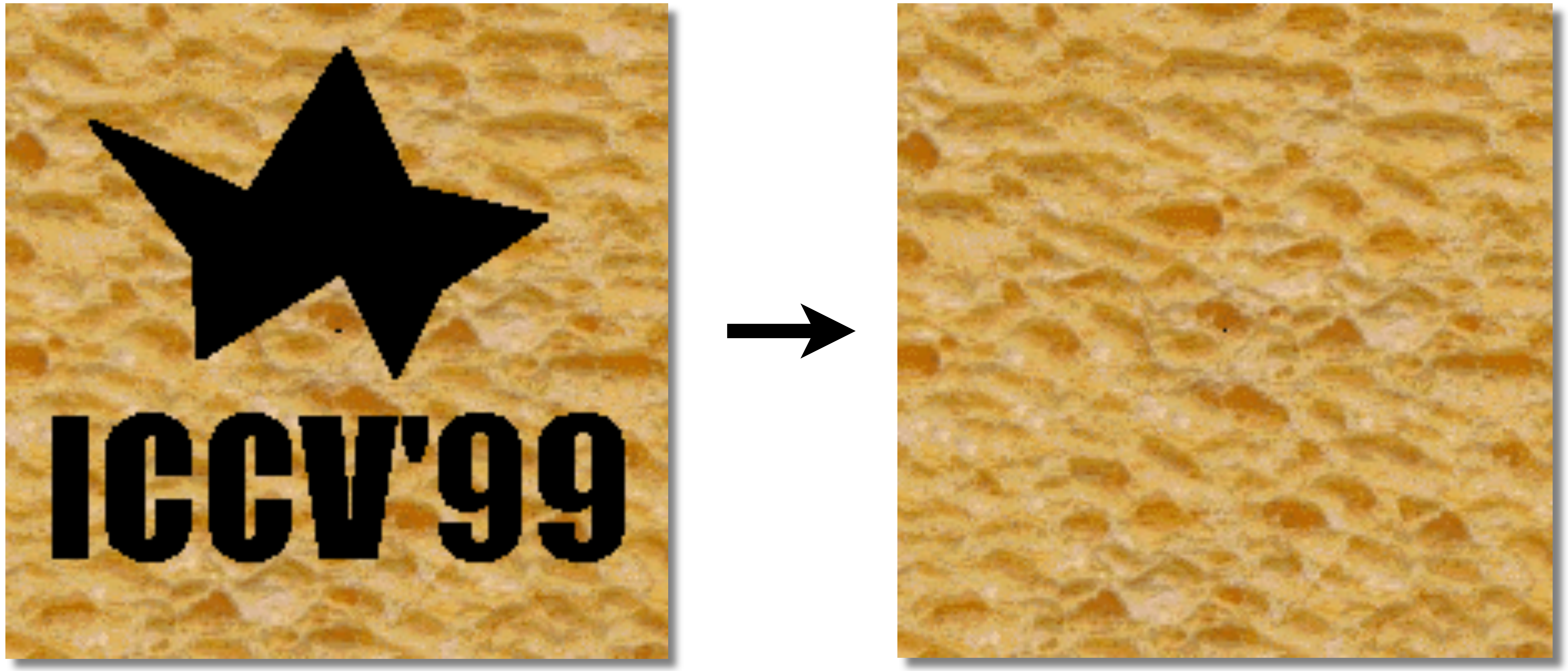
— Draw samples directly from the actual texture

— Can account for more types of structure

— Very simple to implement

— Success depends on choosing a correct “distance”

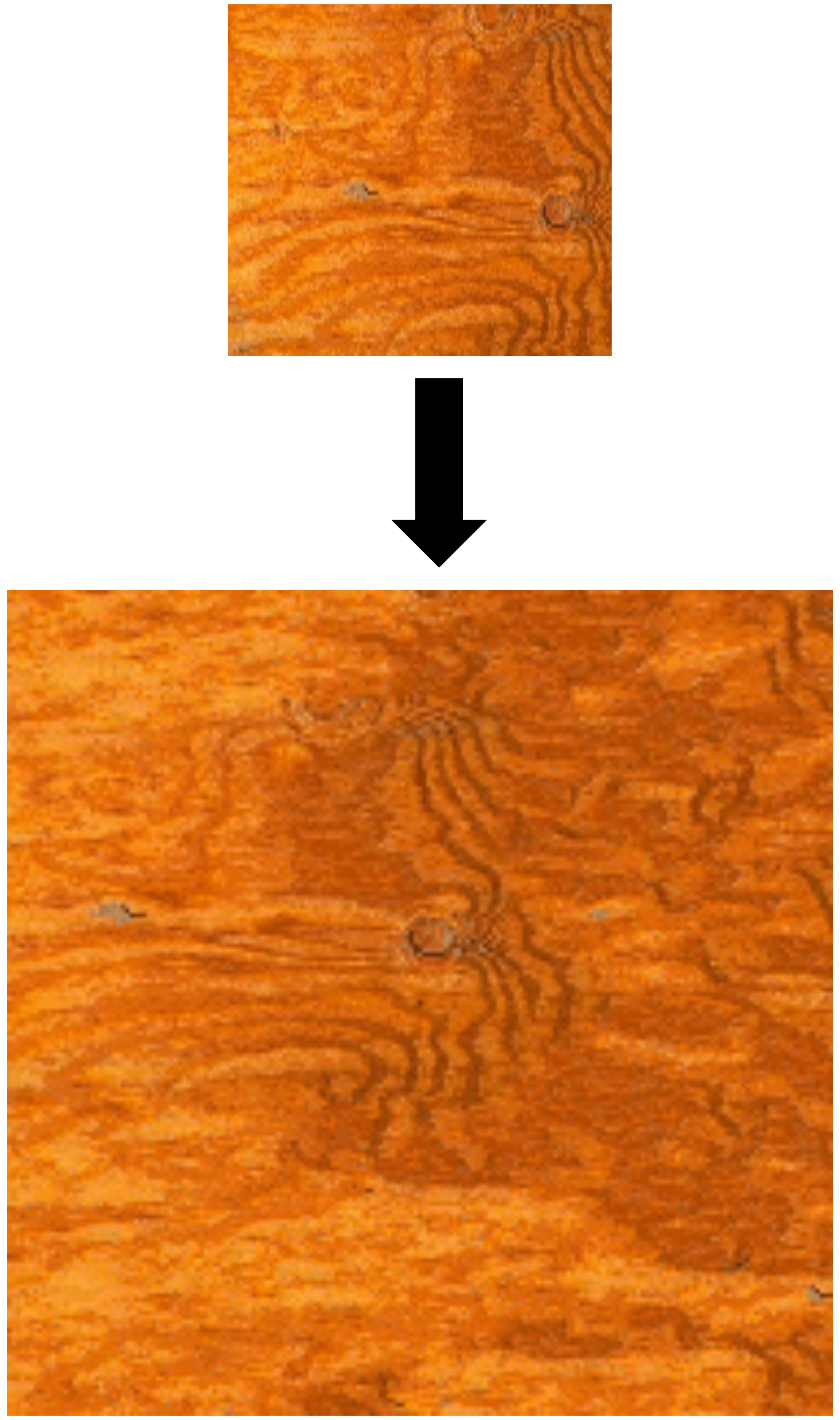
Texture Synthesis by Non-parametric Sampling



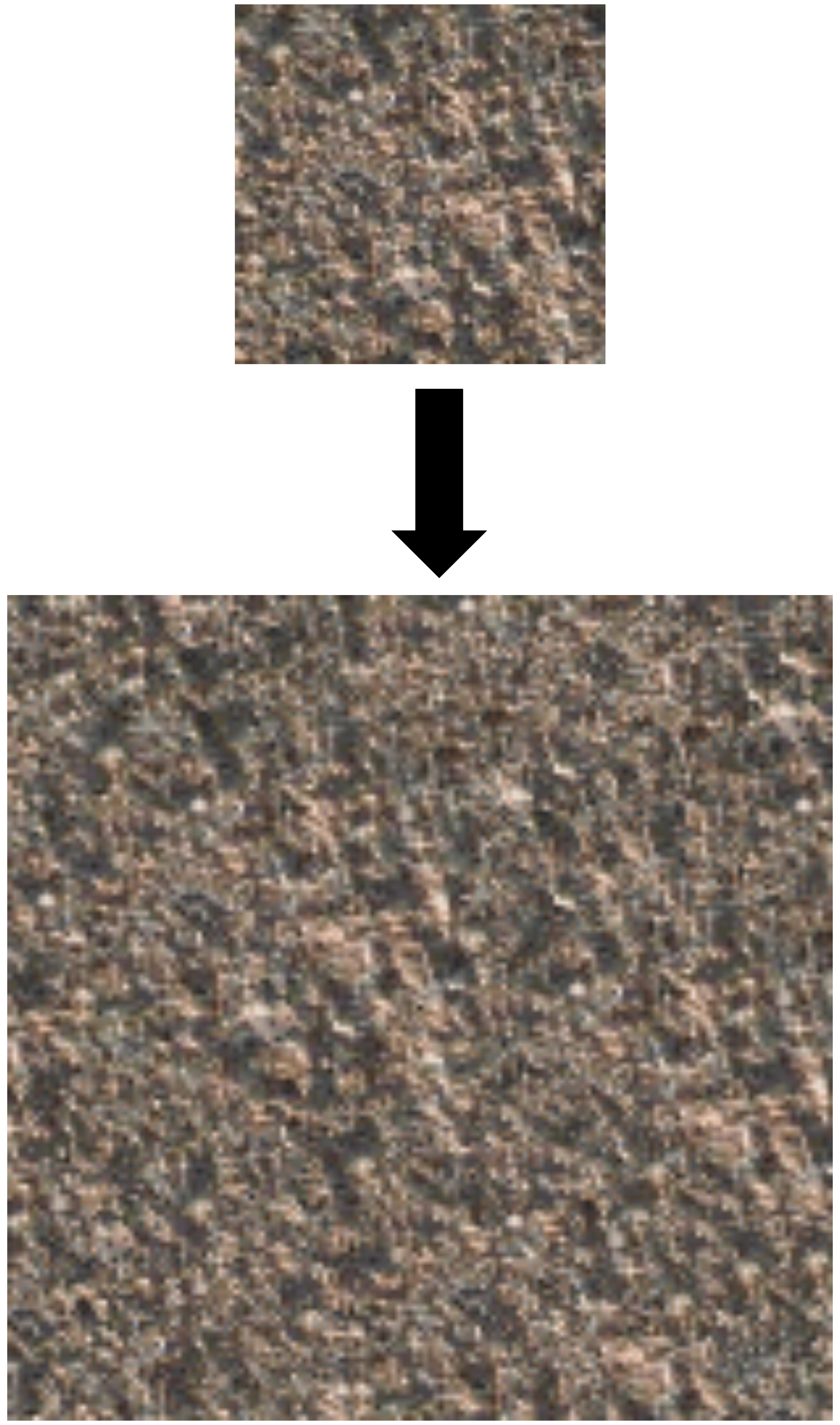
Alexei Efros and Thomas Leung
UC Berkeley

Slide Credit: http://graphics.cs.cmu.edu/people/efros/research/NPS/efros-iccv99.ppt

Efros and Leung

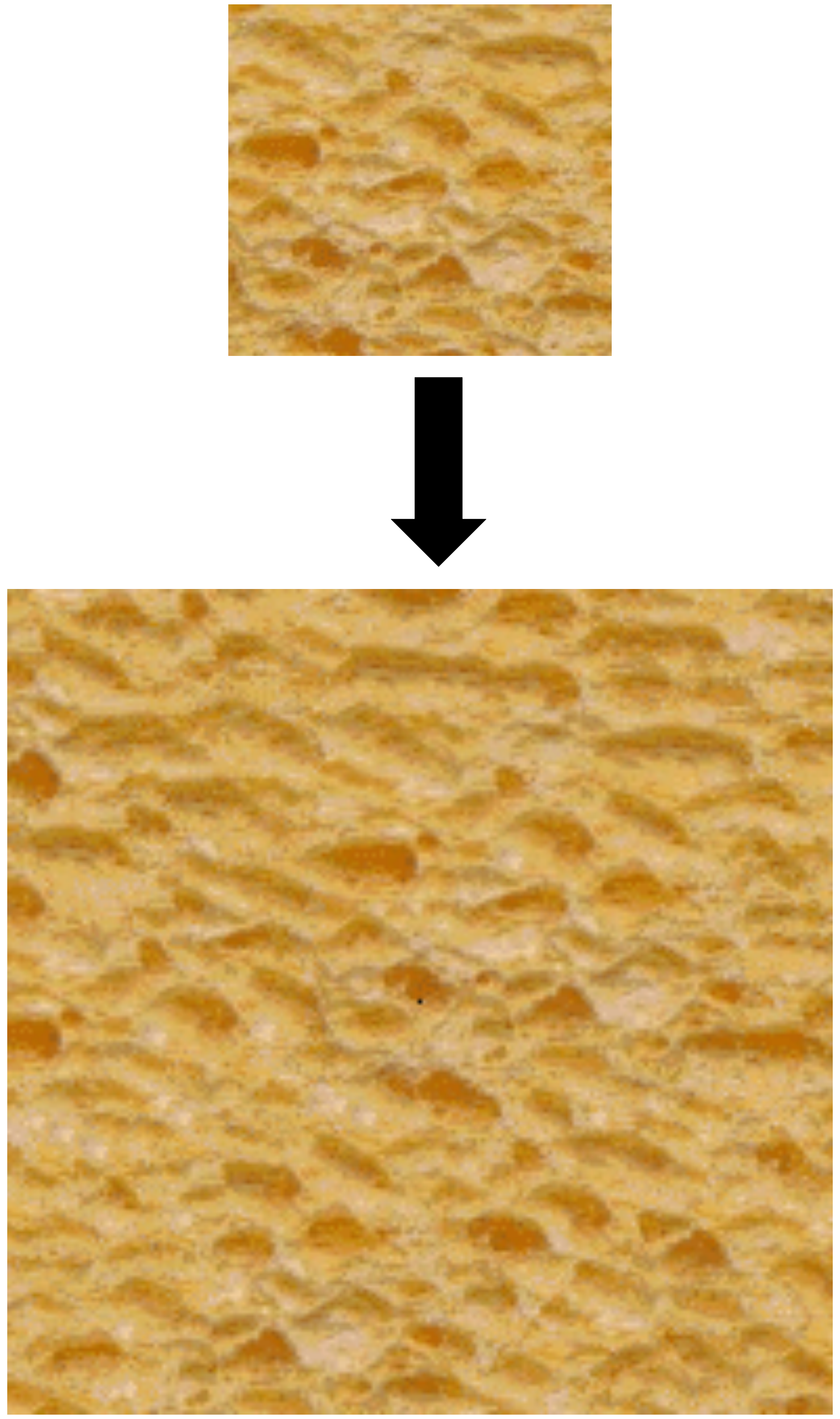


wood

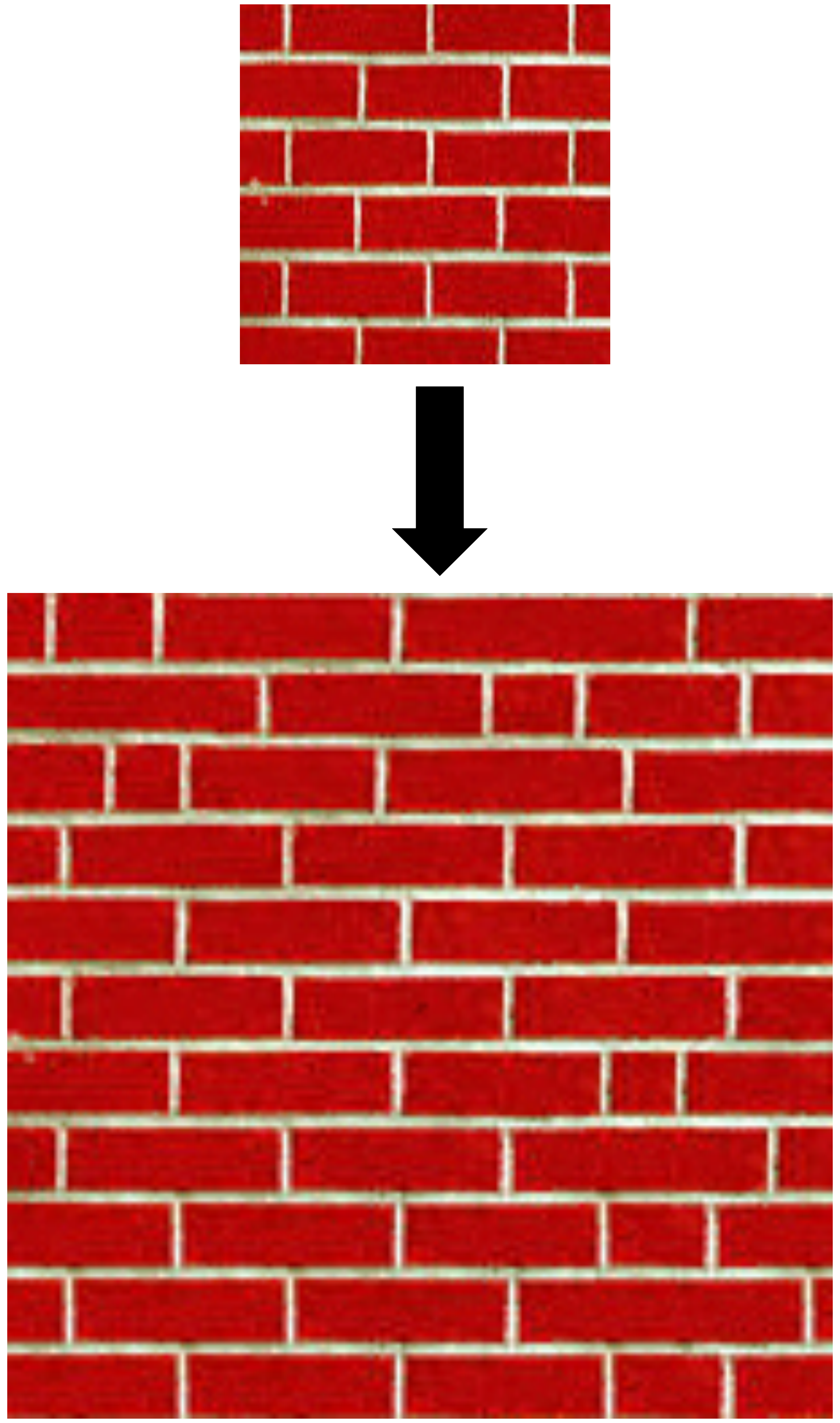


granite

Efros and Leung

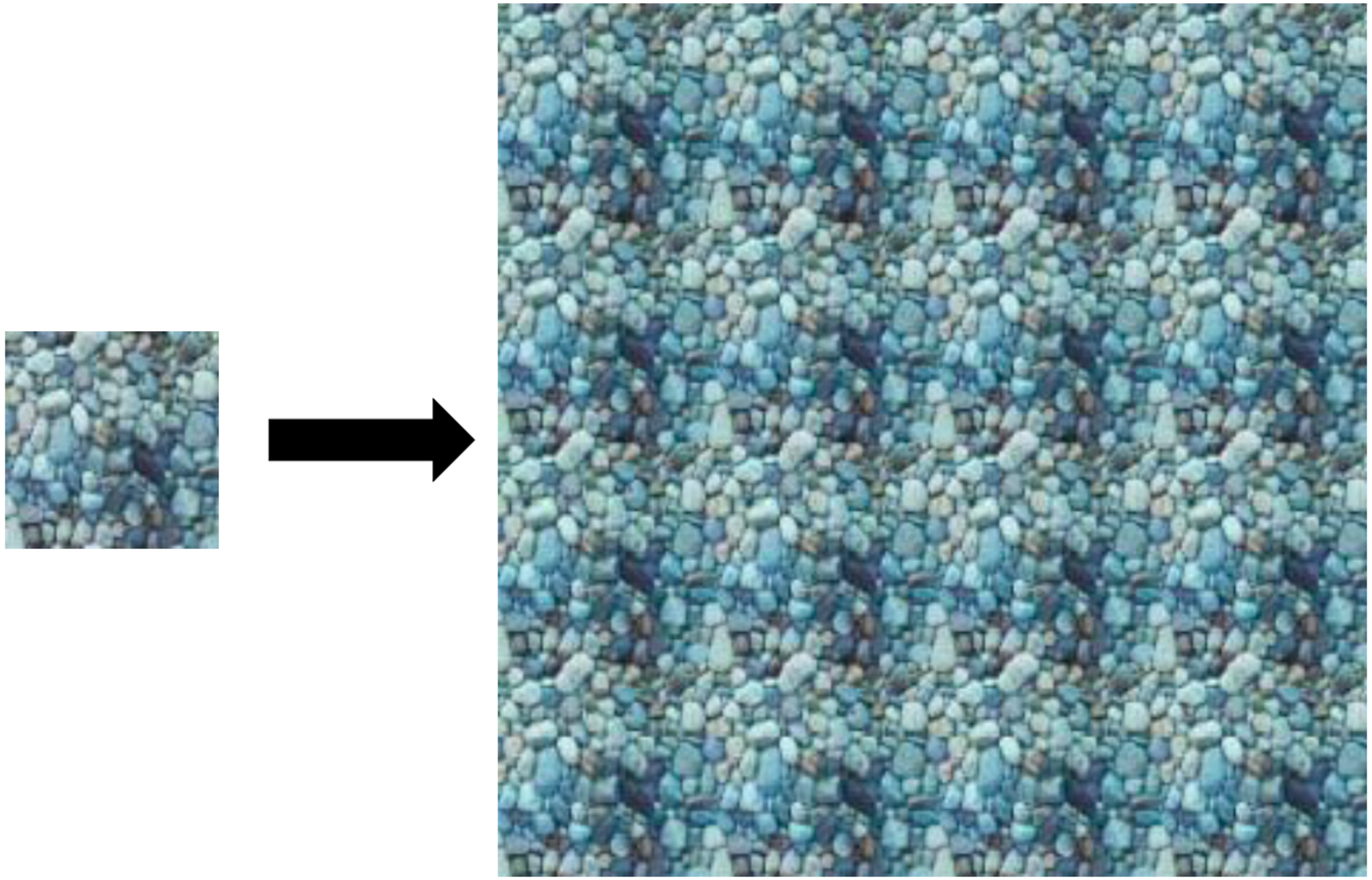


white bread

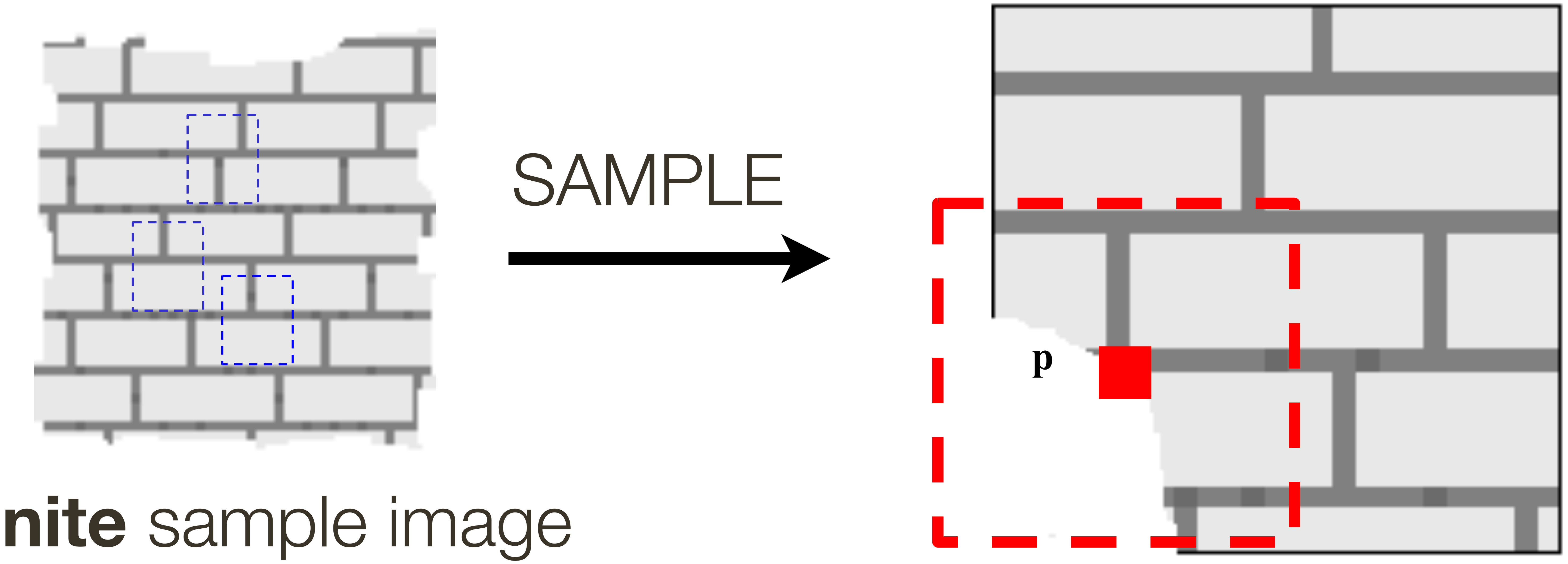


brick wall

Like **Copying**, But not Just Repetition



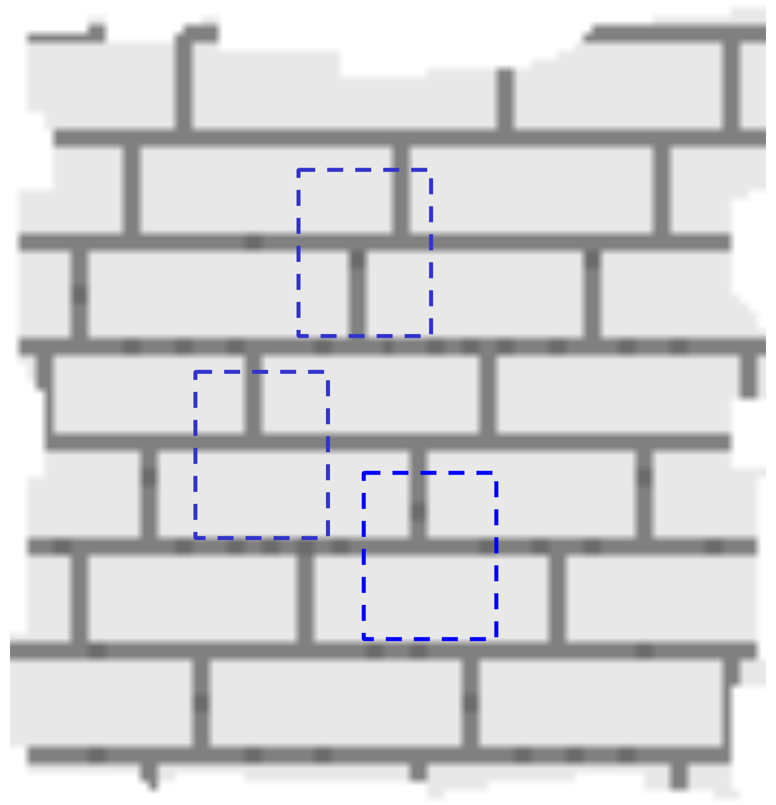
Efros and Leung: Synthesizing One Pixel



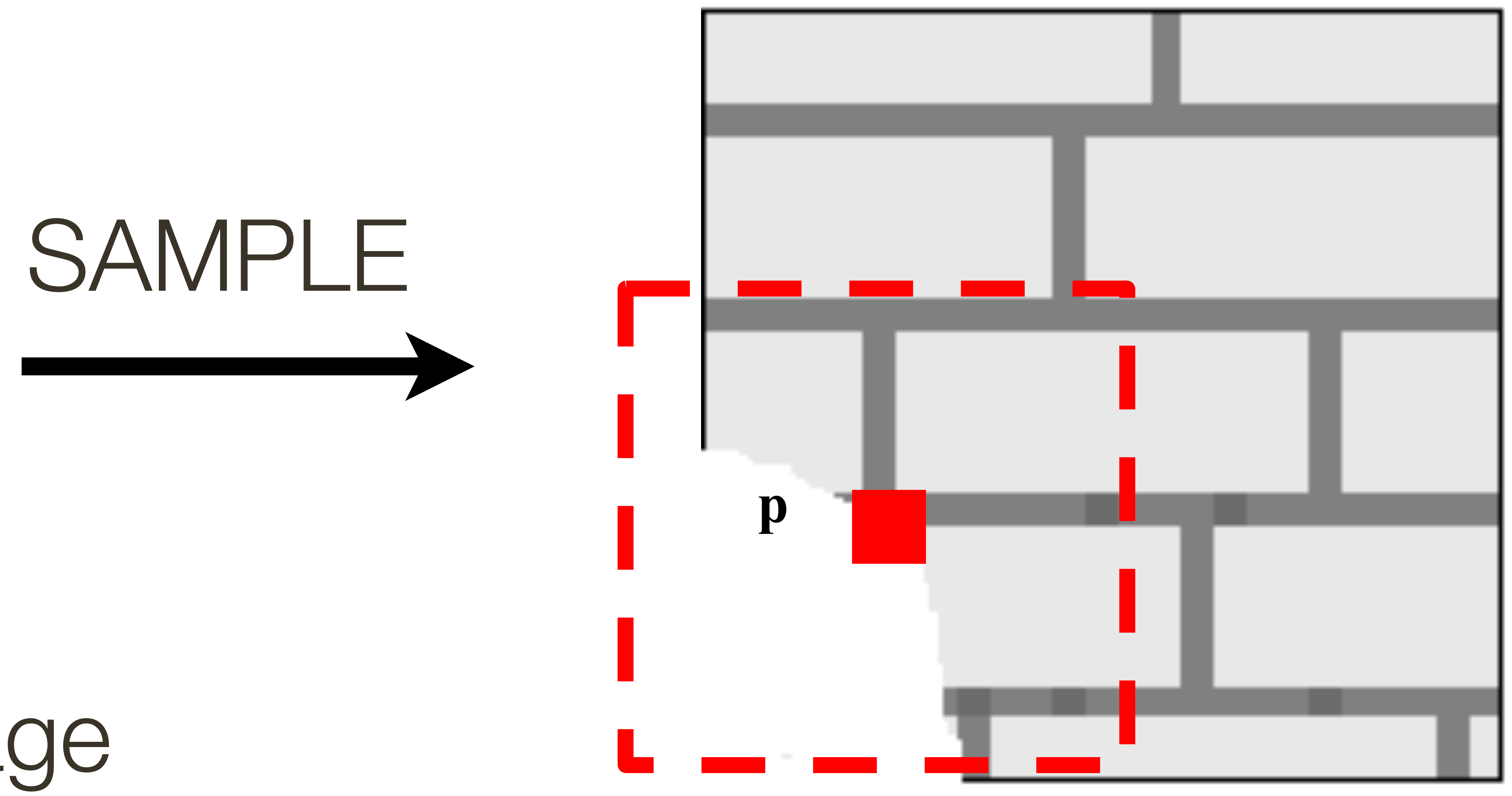
Infinite sample image

— What is **conditional** probability distribution of *p*, given the neighbourhood window?

Efros and Leung: Synthesizing One Pixel



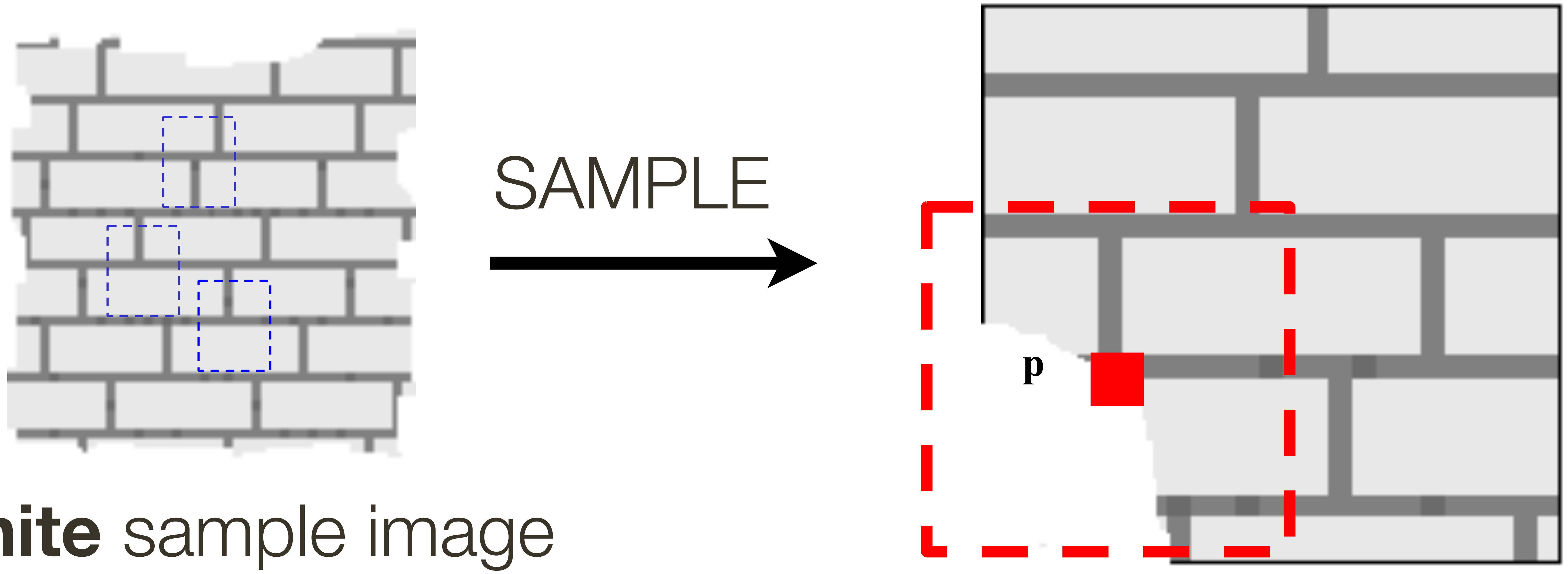
Infinite sample image



— What is **conditional** probability distribution of *p*, given the neighbourhood window?

— Directly search the input image for all such neighbourhoods to produce a **histogram** for *p*

Efros and Leung: Synthesizing One Pixel



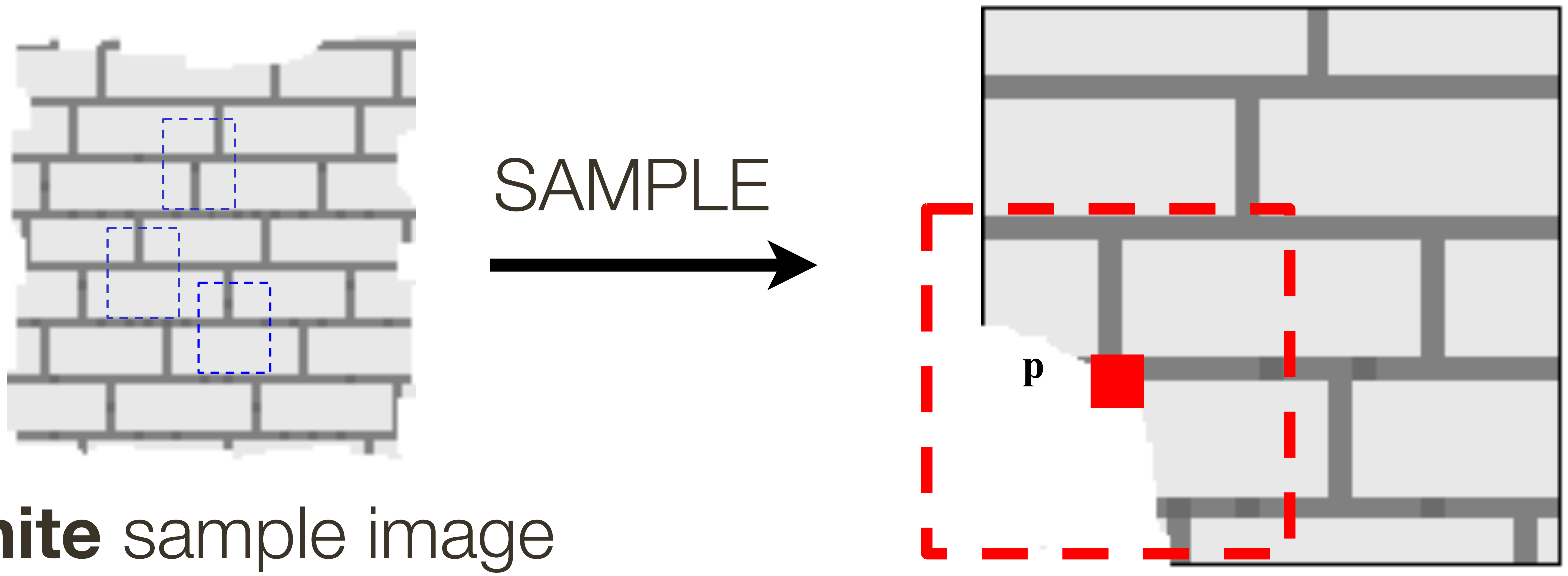
Infinite sample image

— What is **conditional** probability distribution of *p*, given the neighbourhood window?

— Directly search the input image for all such neighbourhoods to produce a **histogram** for *p*

— To **synthesize** *p*, pick one match at random

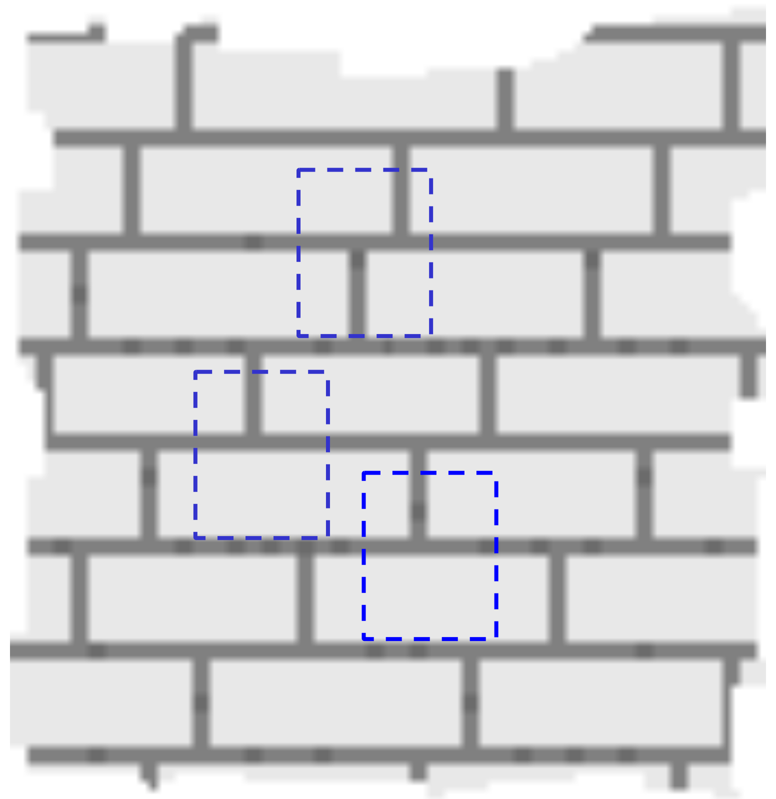
Efros and Leung: Synthesizing One Pixel



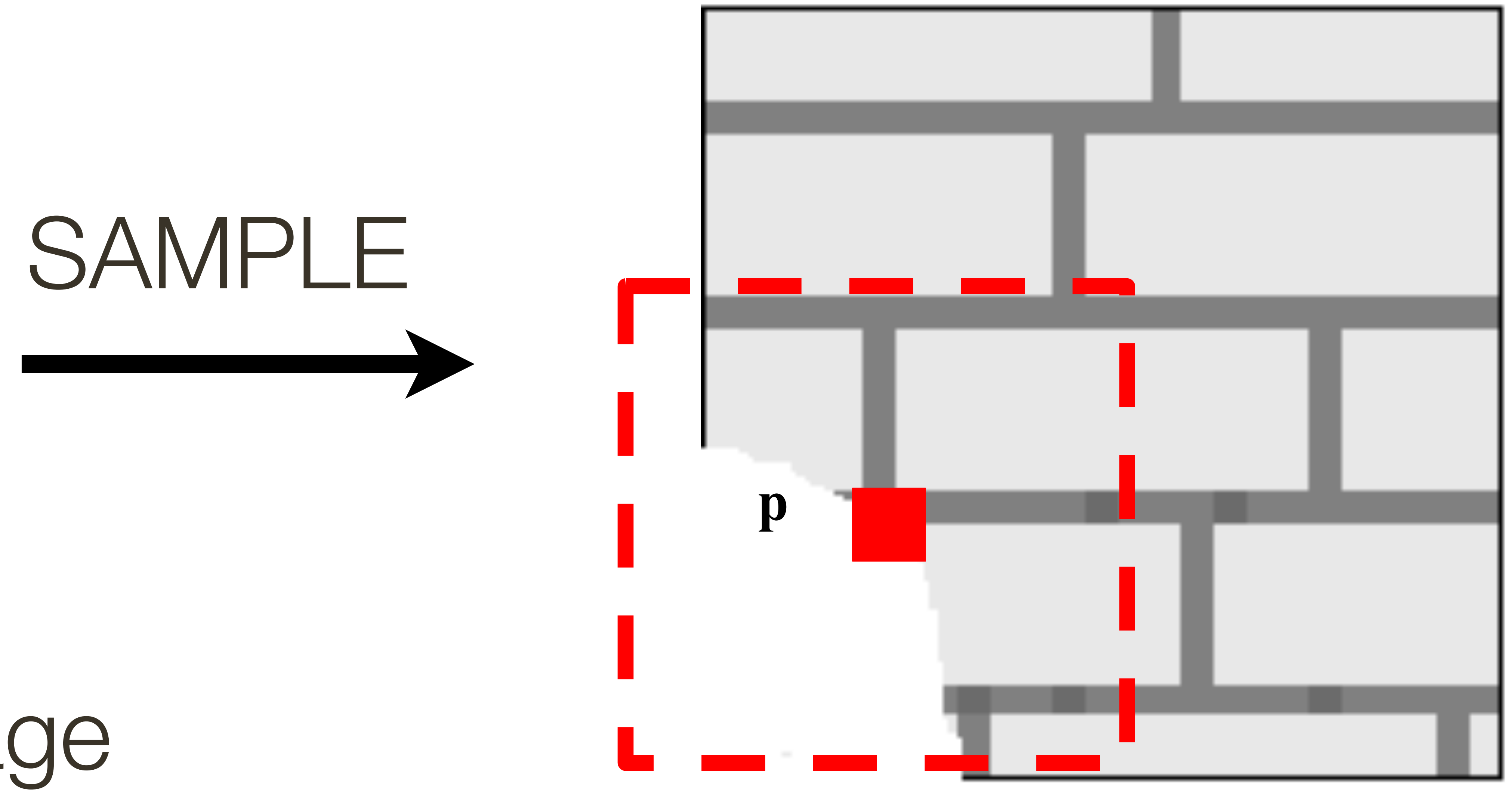
Infinite sample image

— Since the sample image is finite, an exact neighbourhood match might not be present

Efros and Leung: Synthesizing One Pixel



Infinite sample image



— Since the sample image is finite, an exact neighbourhood match might not be present

— Find the **best match** using SSD error, weighted by Gaussian to emphasize local structure, and take all samples within some distance from that match

Efros and Leung: Synthesizing Many Pixels

For multiple pixels, "grow" the texture in layers

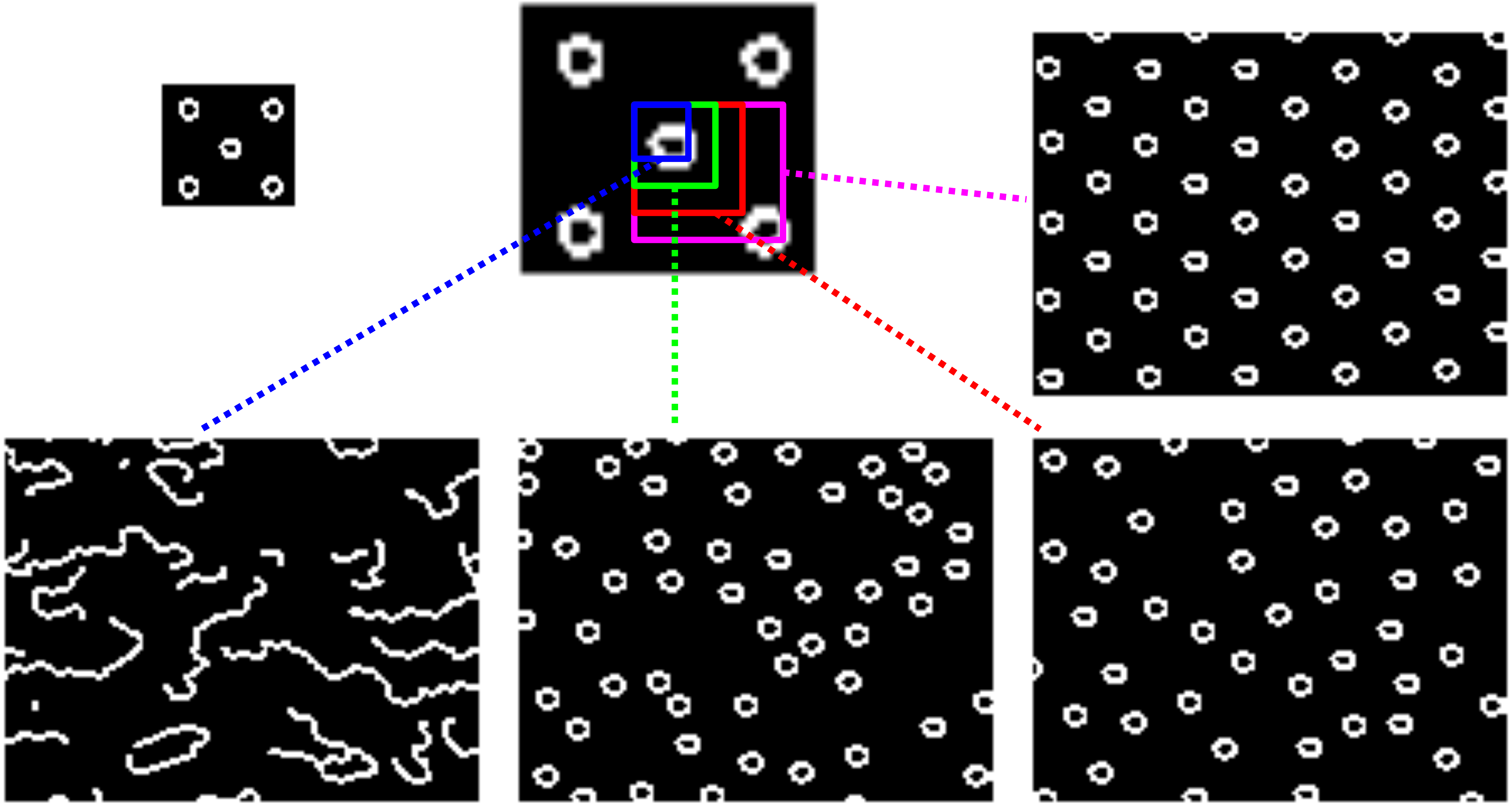
— In the case of hole-filling, start from the edges of the hole

For an interactive demo, see

https://una-dinosauria.github.io/efros-and-leung-js/

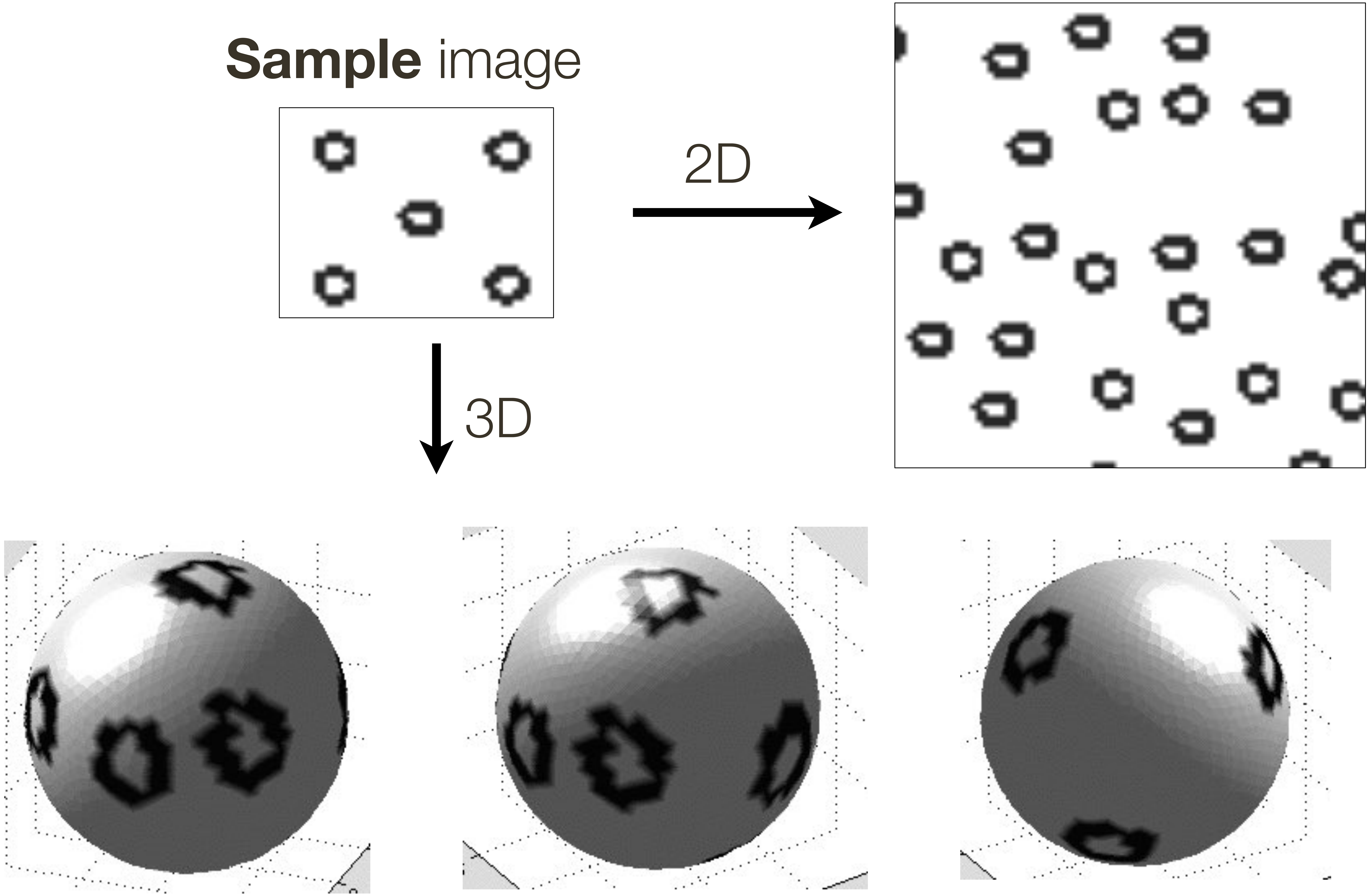
(written by Julieta Martinez, a previous CPSC 425 TA)

Randomness Parameter



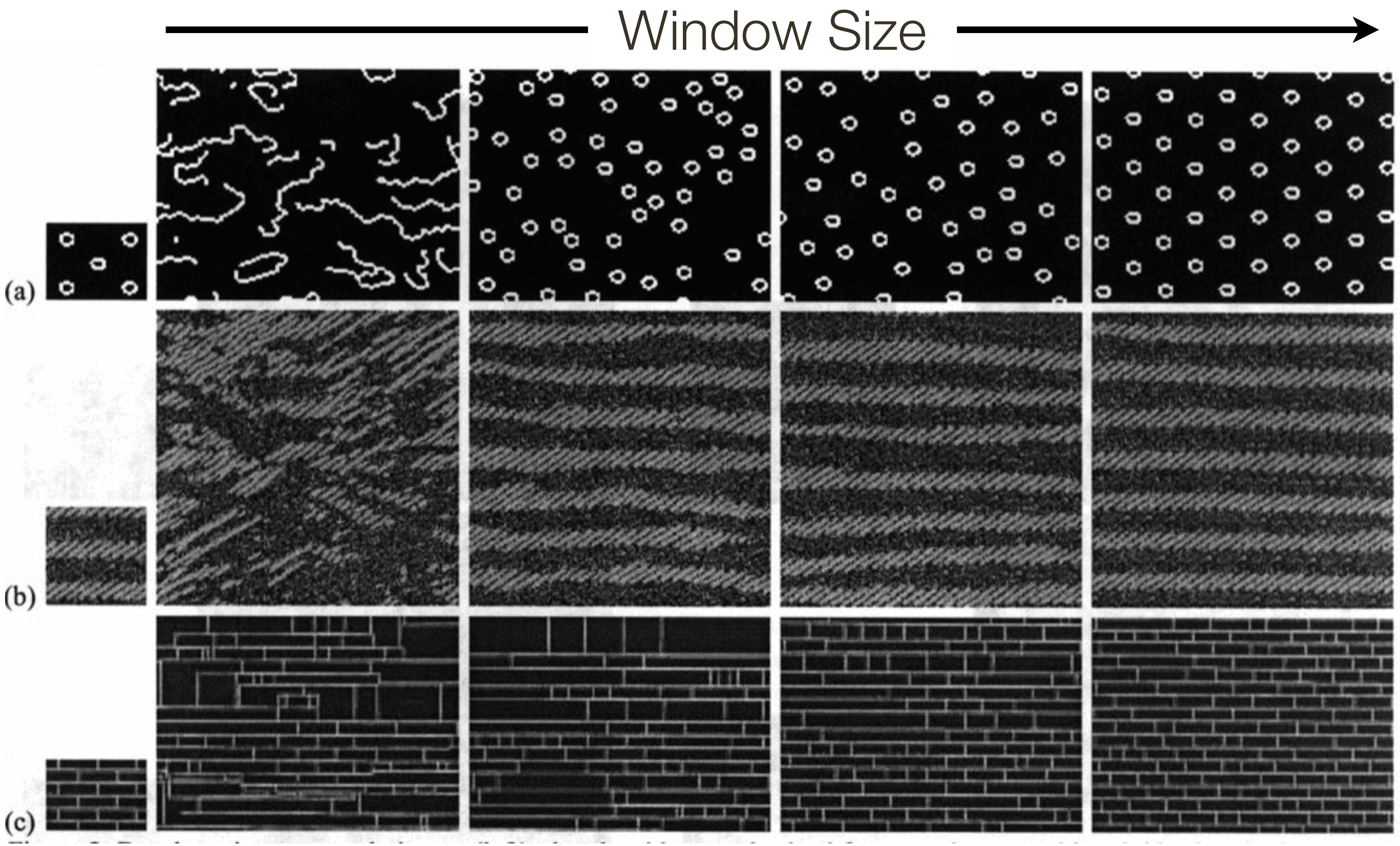
Slide Credit: http://graphics.cs.cmu.edu/people/efros/research/NPS/efros-iccv99.ppt

Texturing a Sphere



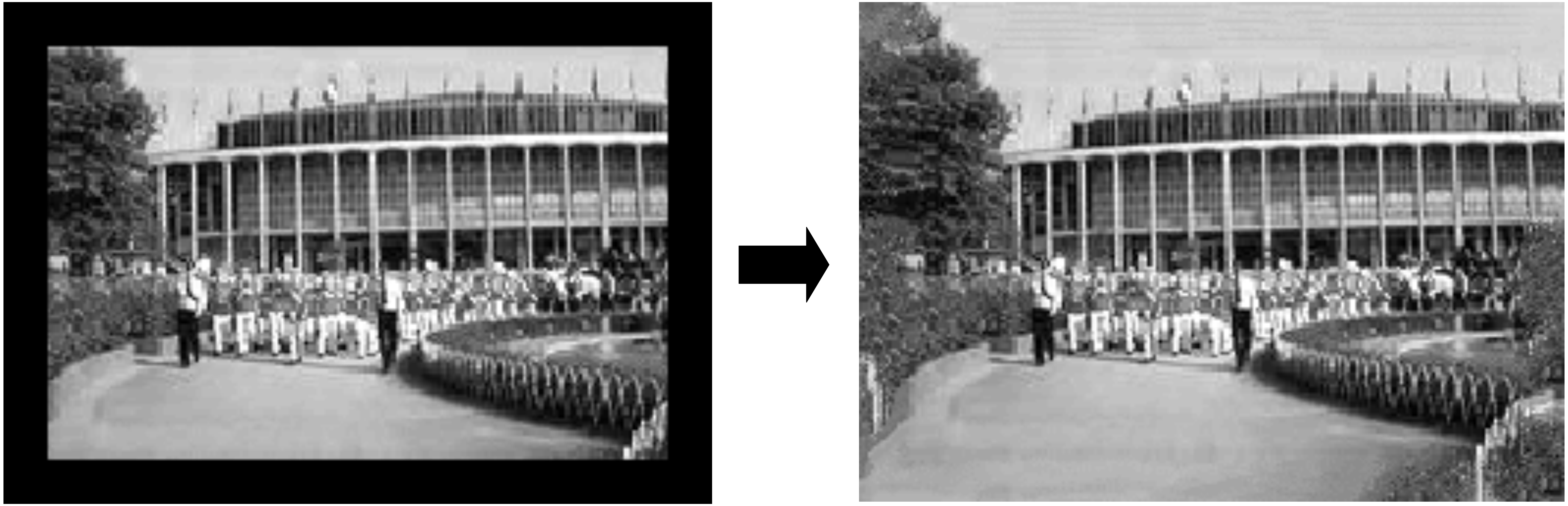
Slide Credit: http://graphics.cs.cmu.edu/people/efros/research/NPS/efros-iccv99.ppt

Efros and Leung: More Synthesis Results



Forsyth & Ponce (2nd ed.) Figure 6.12

Efros and Leung: Image Extrapolation



Slide Credit: http://graphics.cs.cmu.edu/people/efros/research/NPS/efros-iccv99.ppt

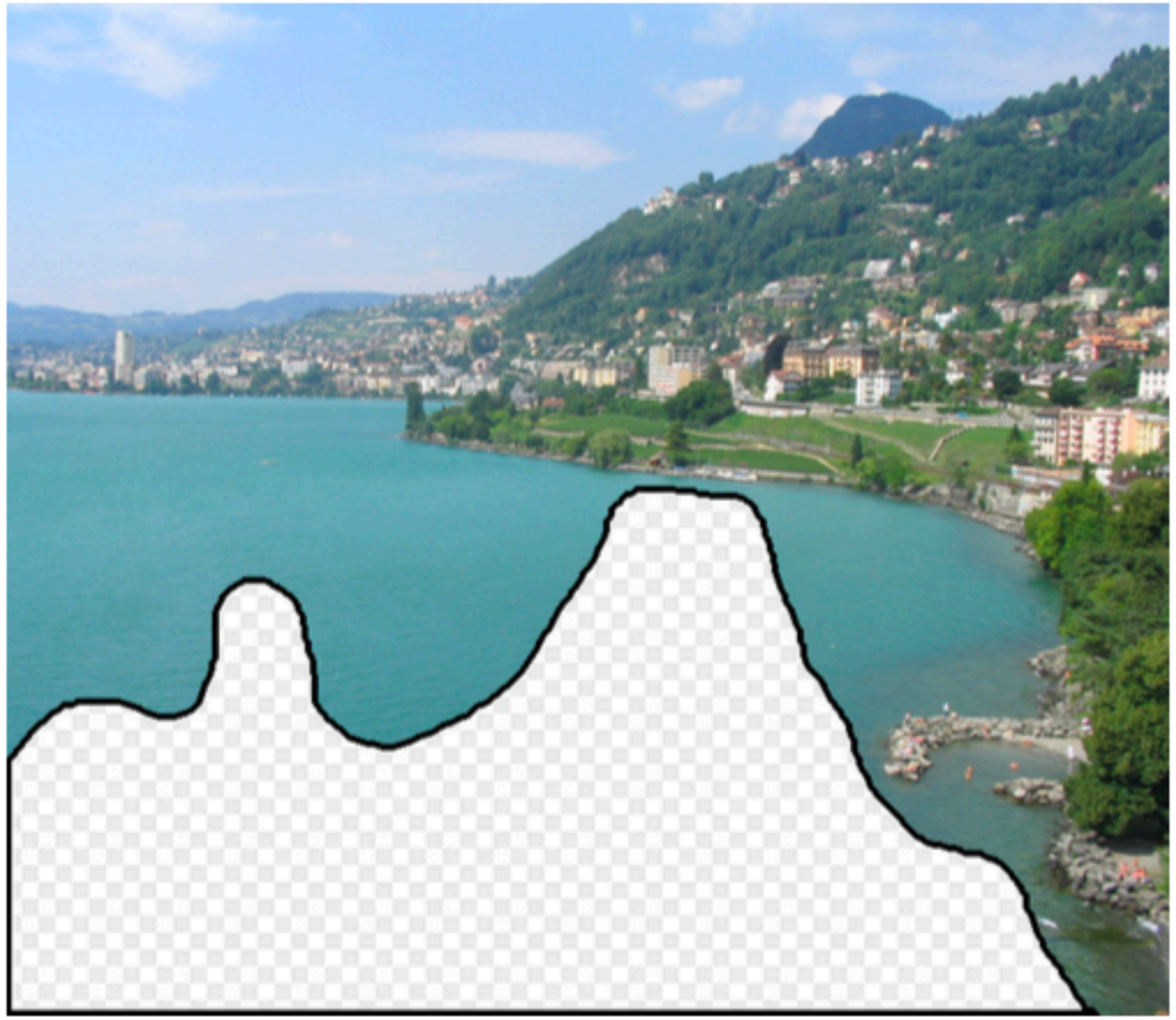
“**Big** Data” Meets Inpainting

“**Big** Data" enables surprisingly simple non-parametric, matching-based techniques to solve complex problems in computer graphics and vision.

Suppose instead of a single image, you had a massive database of a million images. What could you do?

“**Big** Data” Meets Inpainting

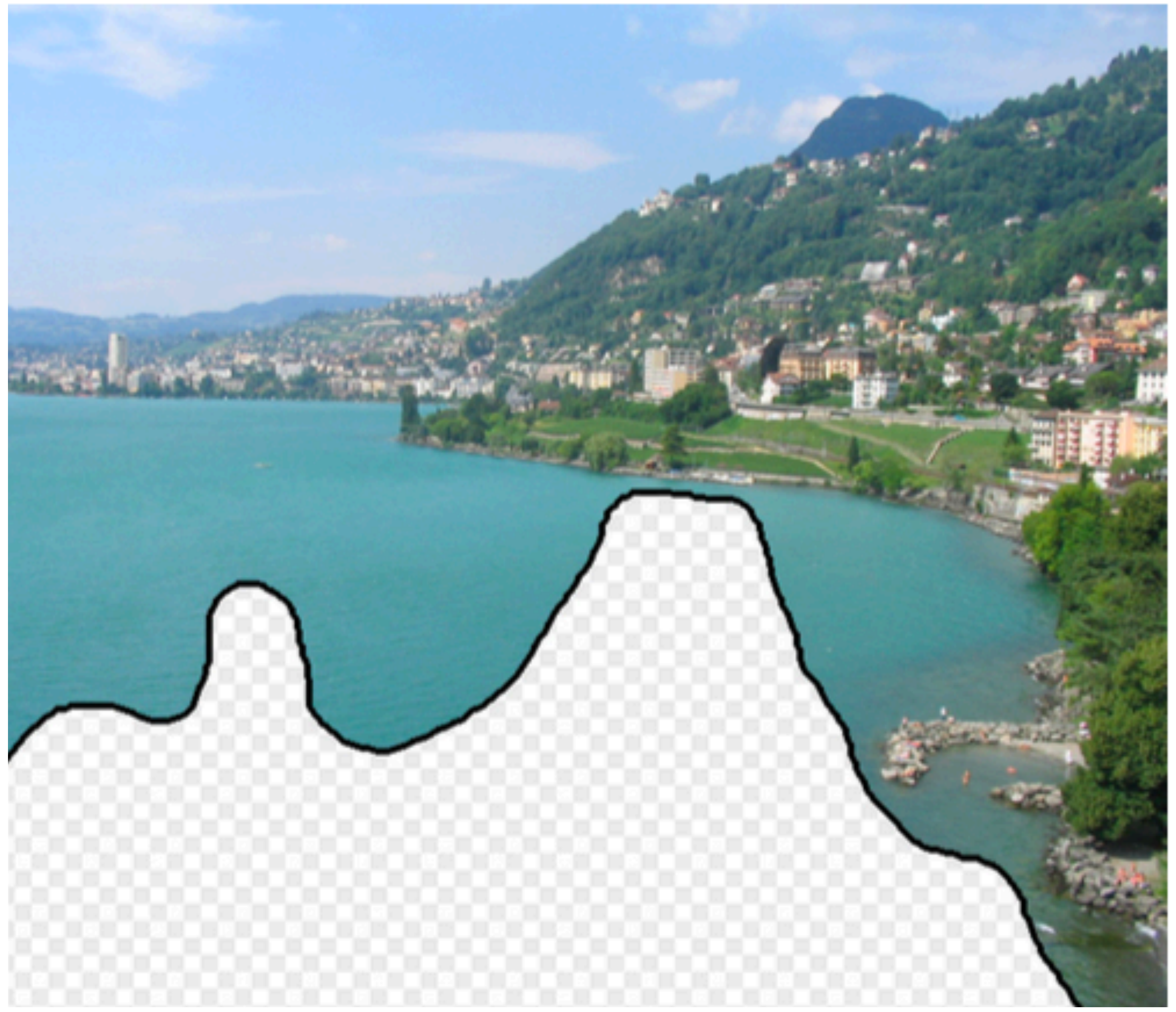
Original Image



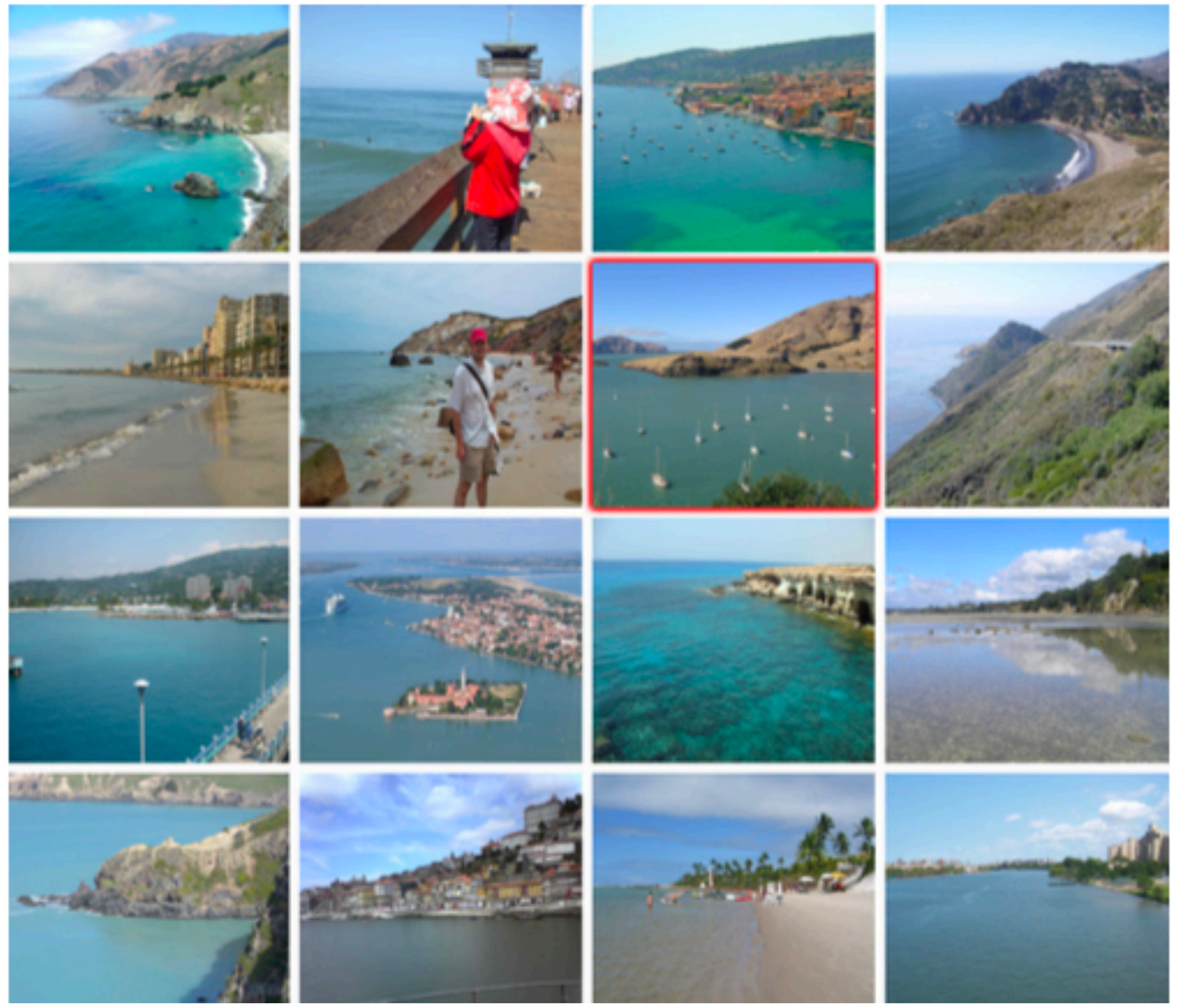
Input

Figure Credit: Hays and Efros 2007

“**Big** Data” Meets Inpainting



Input



Scene Matches

Output

Figure Credit: Hays and Efros 2007

Effectiveness of “Big Data”

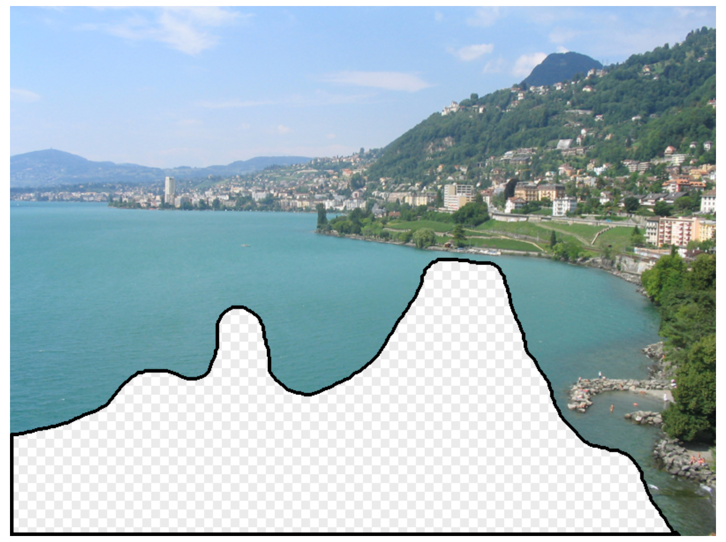
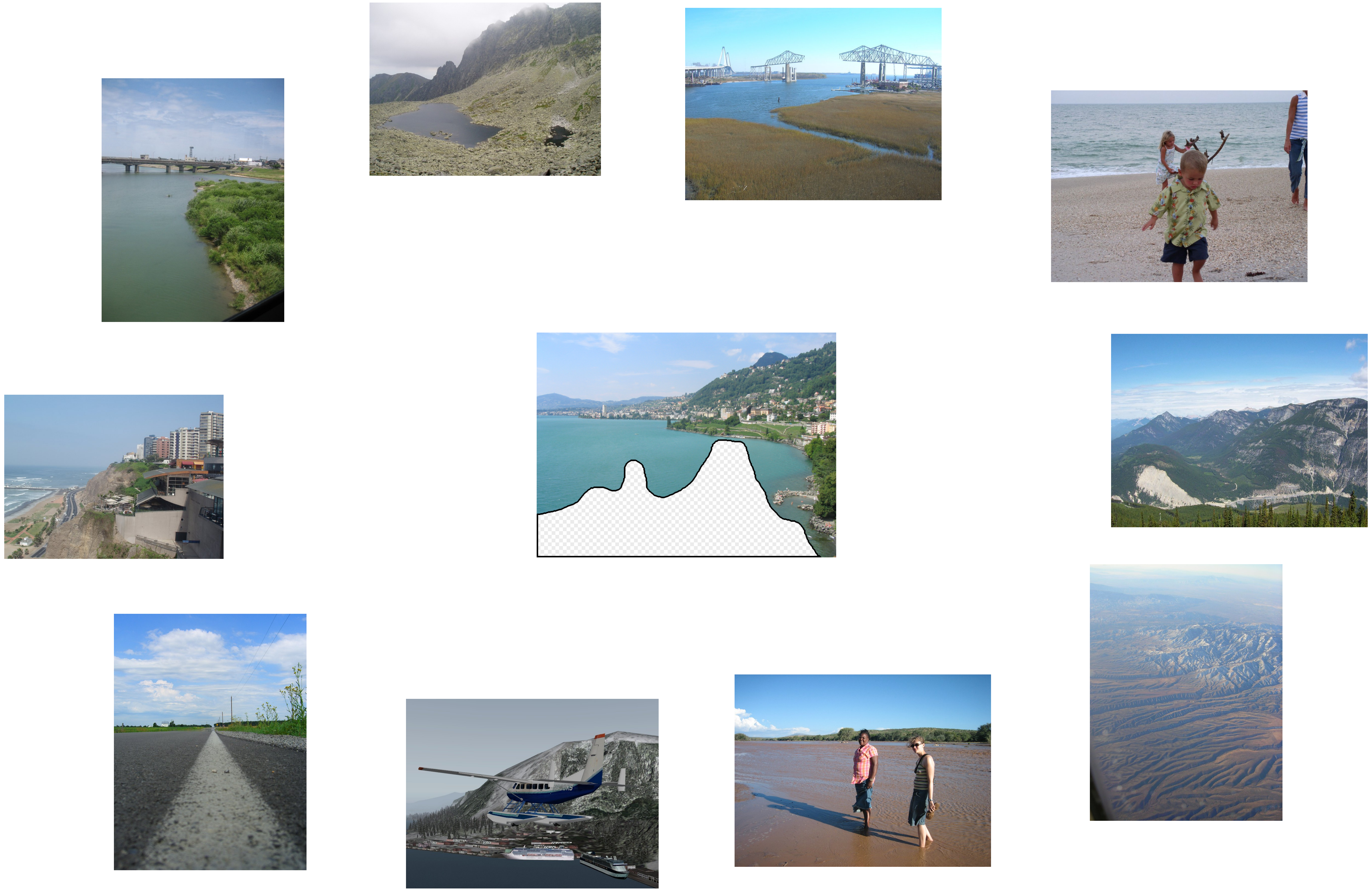


Figure Credit: Hays and Efros 2007

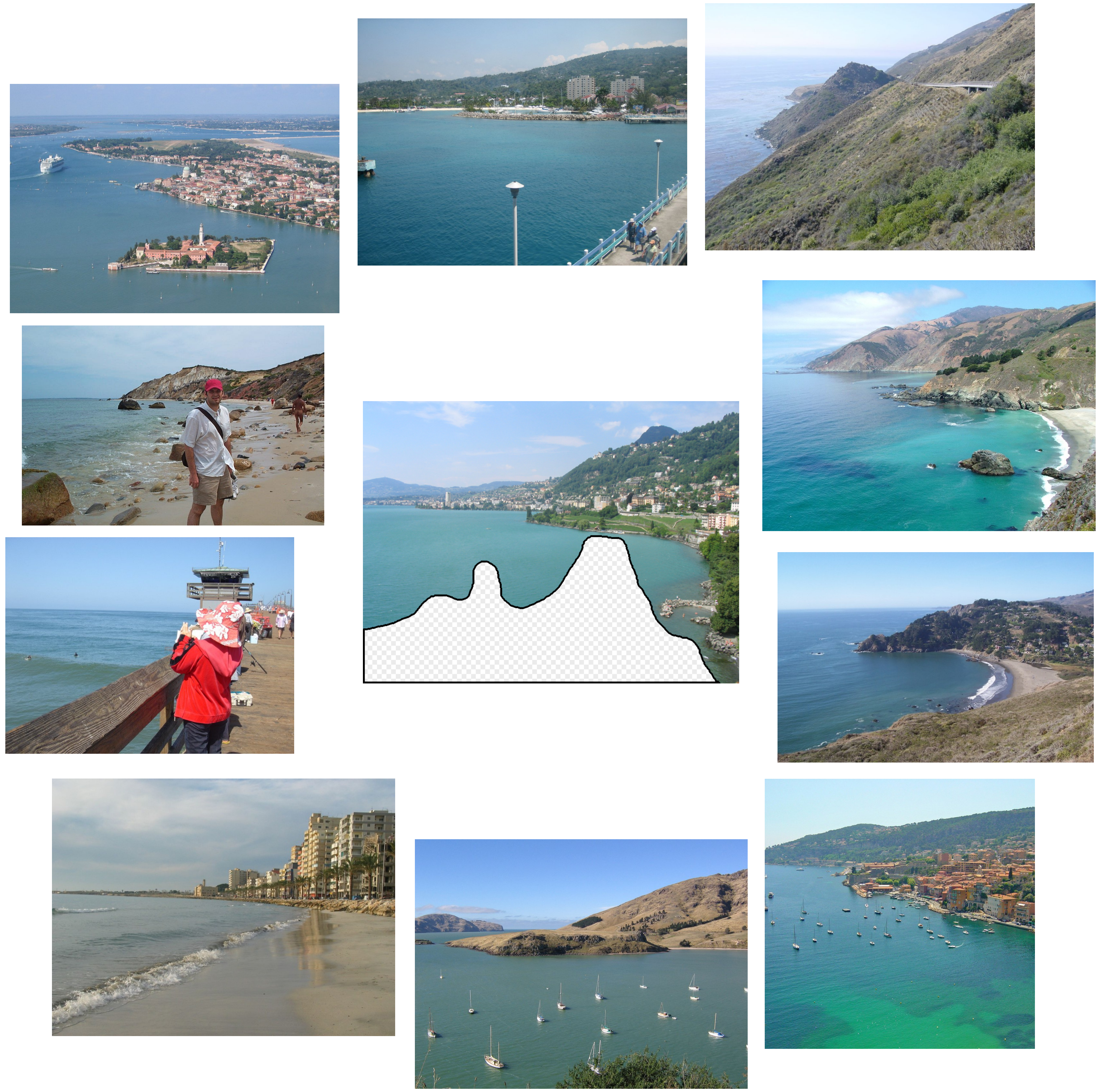
Effectiveness of “Big Data”



10 nearest neighbors from a collection of 20,000 images

Figure Credit: Hays and Efros 2007

Effectiveness of “Big Data”



10 nearest neighbors from a collection of 2 million images

Figure Credit: Hays and Efros 2007

“**Big** Data” Meets Inpainting

Figure Credit: Hays and Efros 2007

“**Big** Data” Meets Inpainting

Algorithm sketch (Hays and Efros 2007):

1. Create a short list of a few hundred “best matching" images based on global image statistics

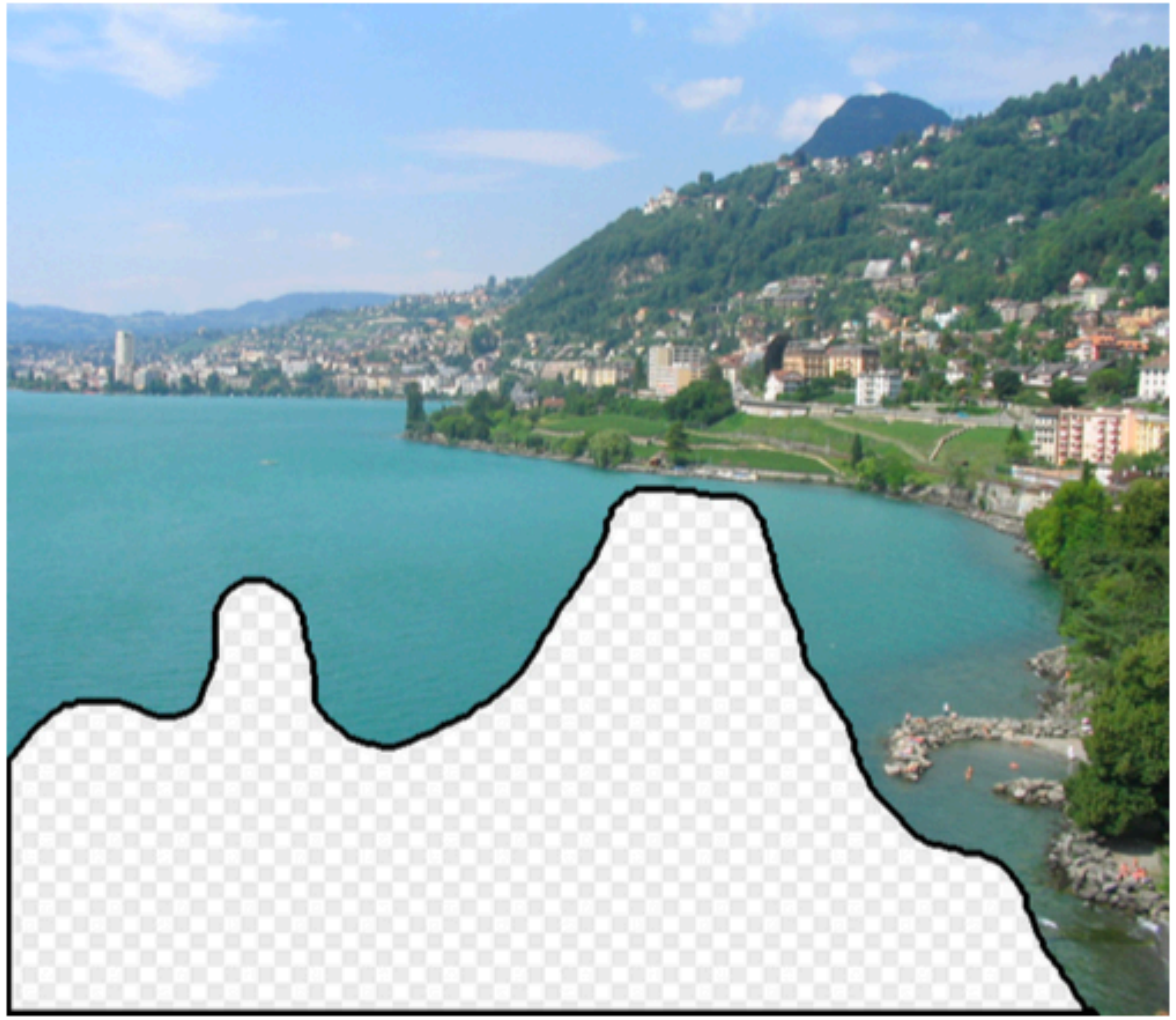
2. Find patches in the short list that match the context surrounding the image region we want to fill

3. Blend the match into the original image

Purely **data-driven**, requires no manual labeling of images

“**Big** Data” Meets Inpainting

Original Image



Input

Figure Credit: Hays and Efros 2007

“**Big** Data” Meets Inpainting

Figure Credit: Hays and Efros 2007

“**Big** Data” Meets Inpainting

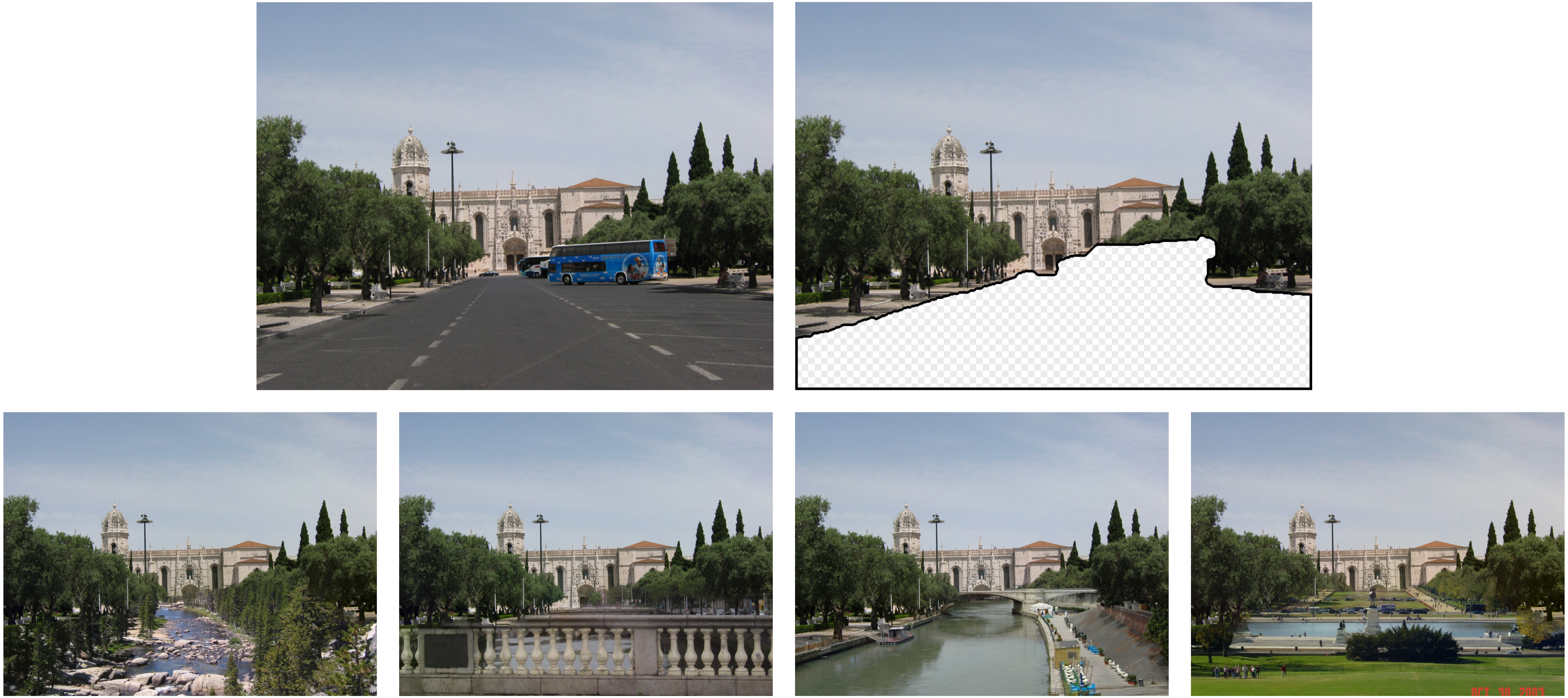


Figure Credit: Hays and Efros 2007

Summary

Texture representation is hard

— difficult to define, to analyze

— texture synthesis appears more tractable

Objective of texture **synthesis** is to generate new examples of a texture

— Efros and Leung: Draw samples directly from the texture to generate one pixel at a time. A “data-driven" approach.

Approaches to texture embed assumptions related to human perception