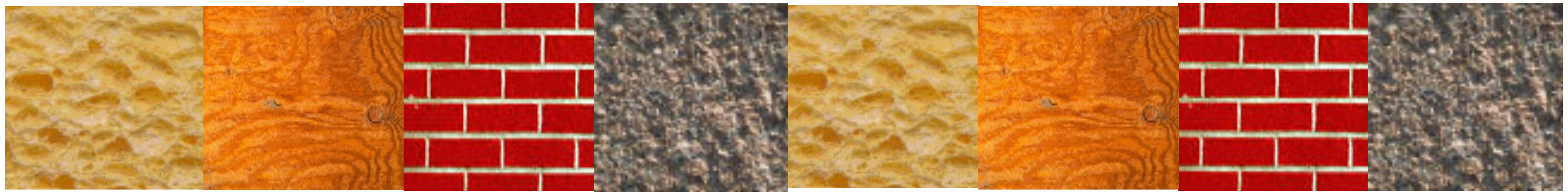
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 14: Texture

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

Menu for Today (**October 5, 2018**)

Topics:

— Texture Synthesis
— Texture Analysis

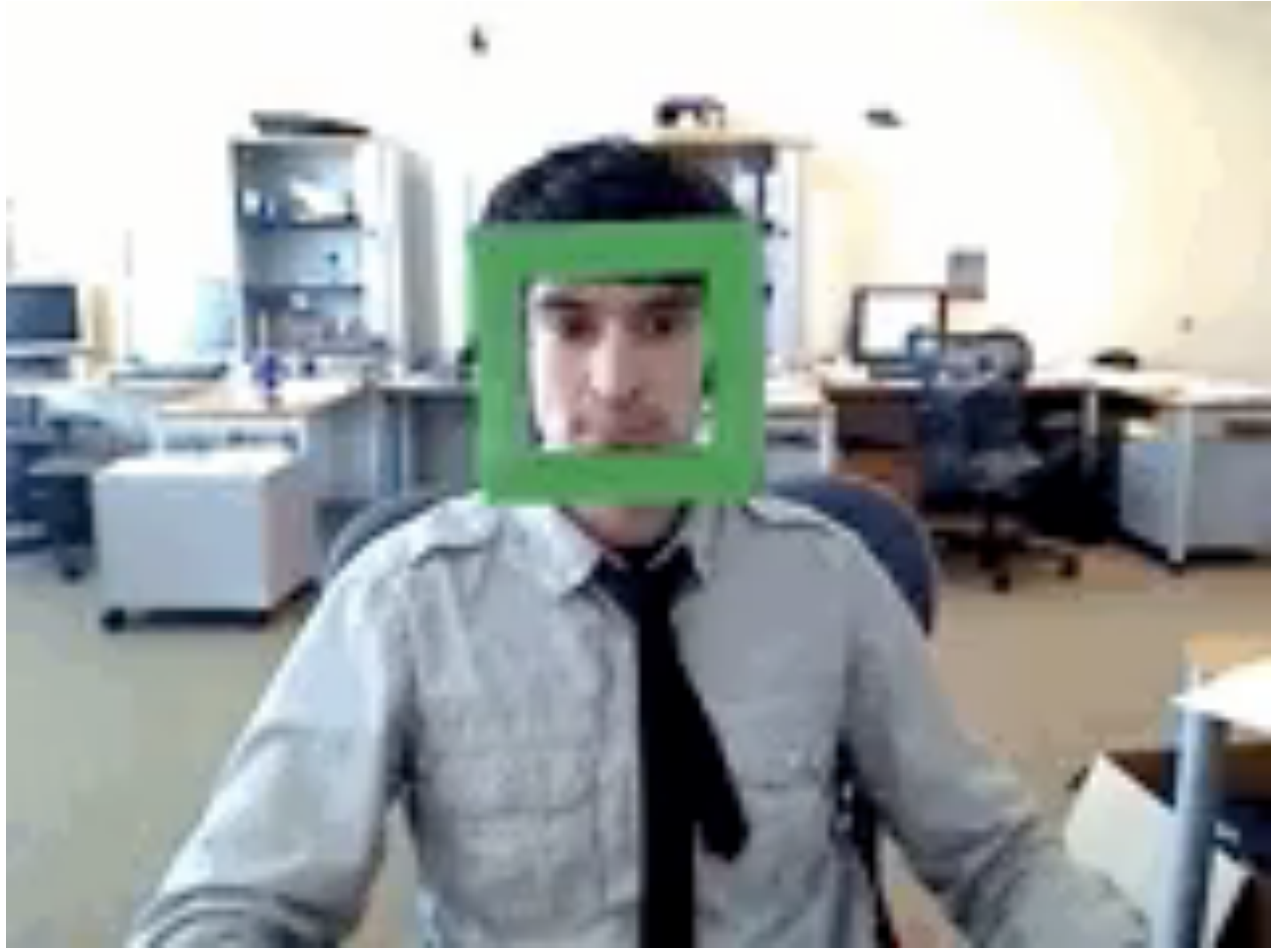
Redings:

— **Today's** Lecture: Forsyth & Ponce (2nd ed.) 6.1, 6.3
— **Next** Lecture: Forsyth & Ponce (2nd ed.) 3.1-3.3

Reminders:

— **Assignment 2**: Face Detection in a Scaled Representation is **October 10th**

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

Lecture 13: Re-cap

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**

We begin with texture synthesis to set up **Assignment 3**

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

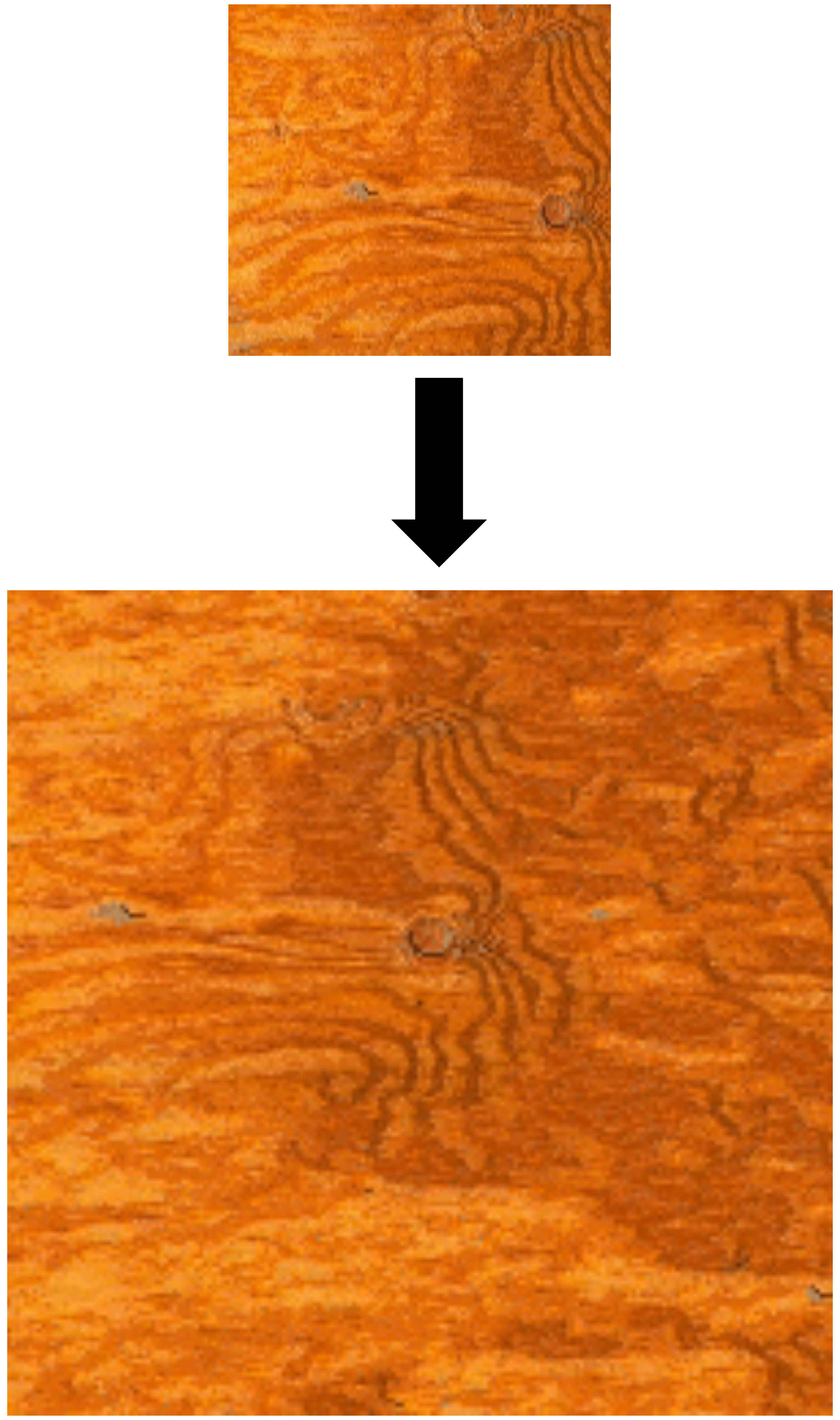
— Success depends on choosing a correct “distance”

Texture Synthesis by Non-parametric Sampling

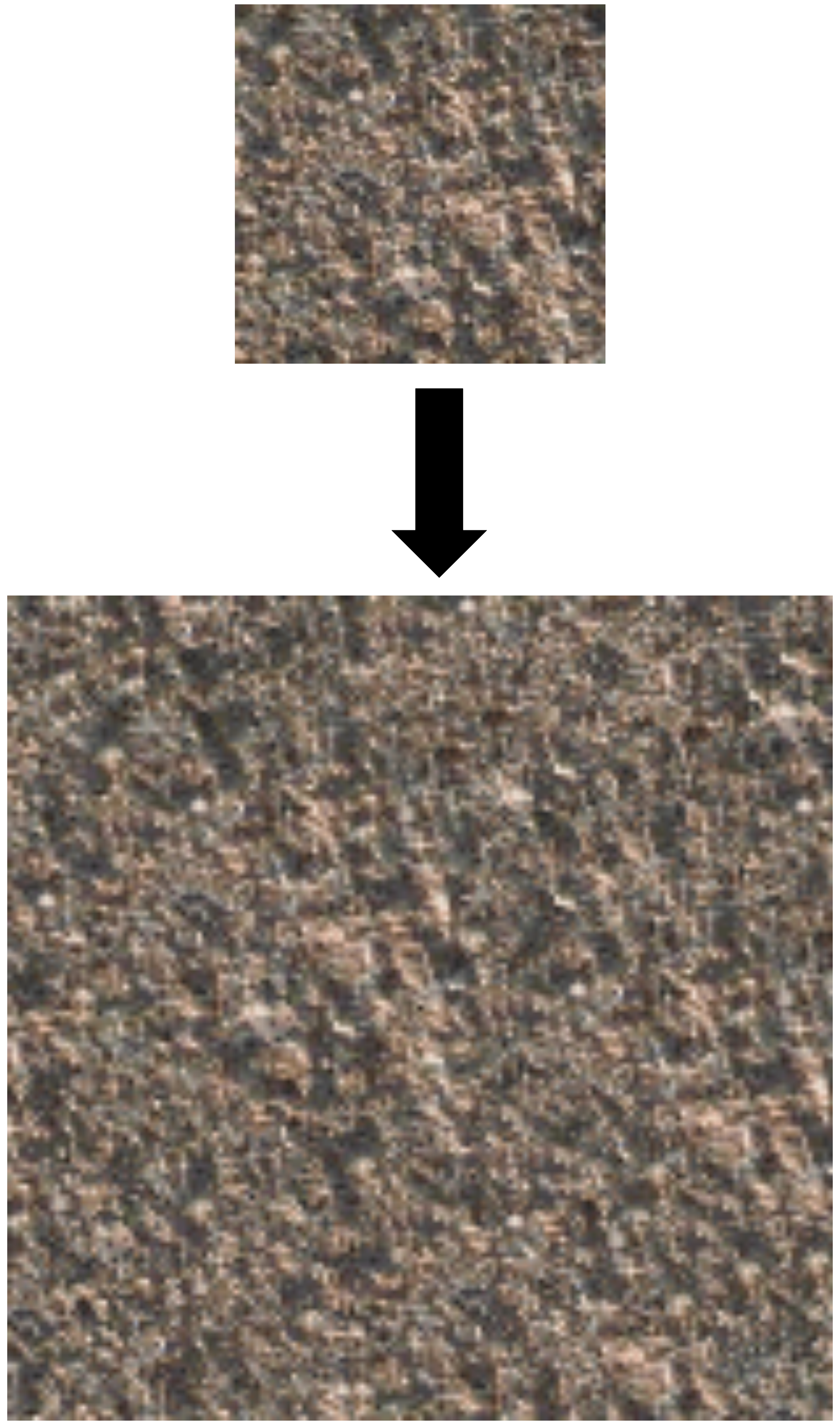
Alexei Efros and Thomas Leung
UC Berkeley

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

Efros and Leung

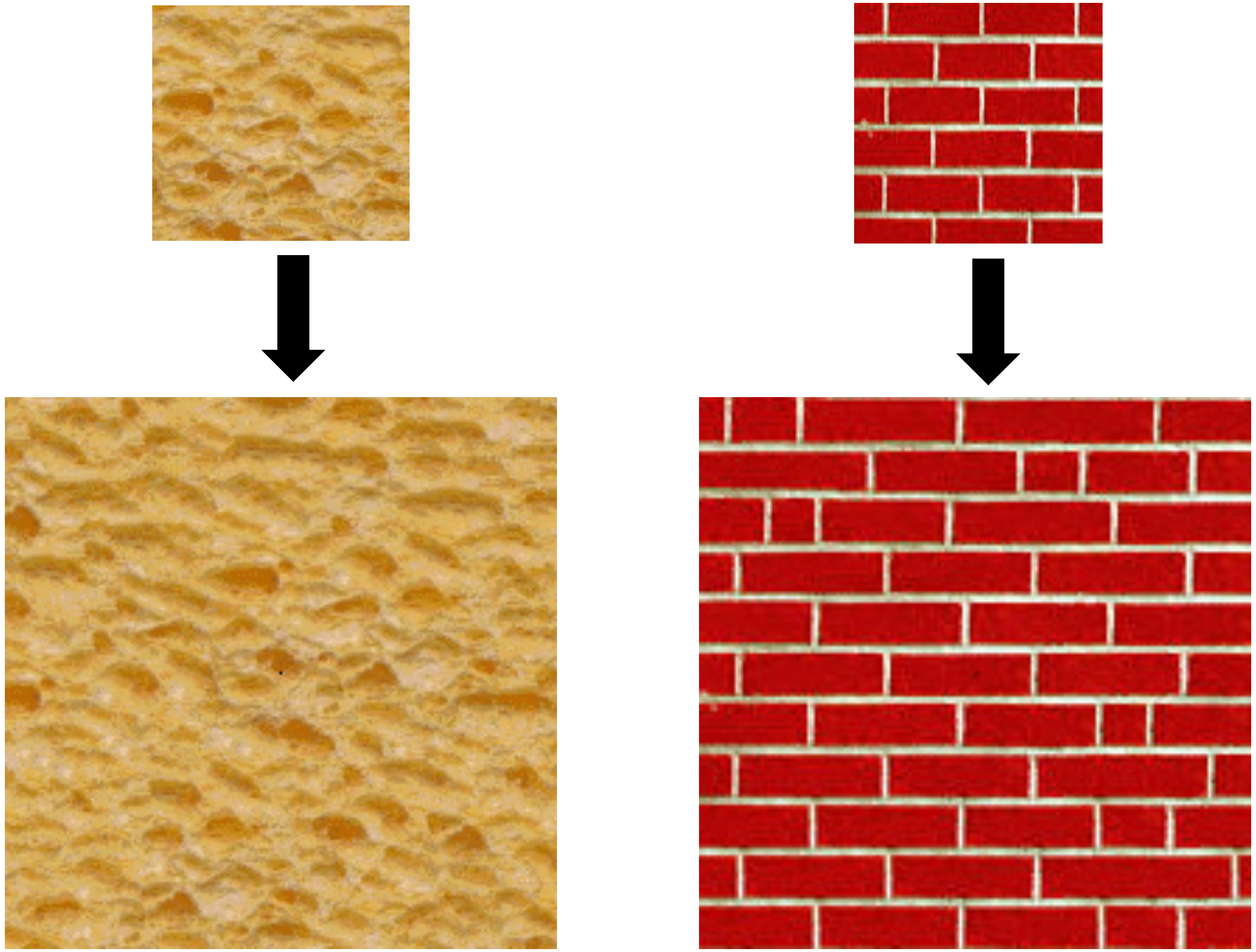


wood



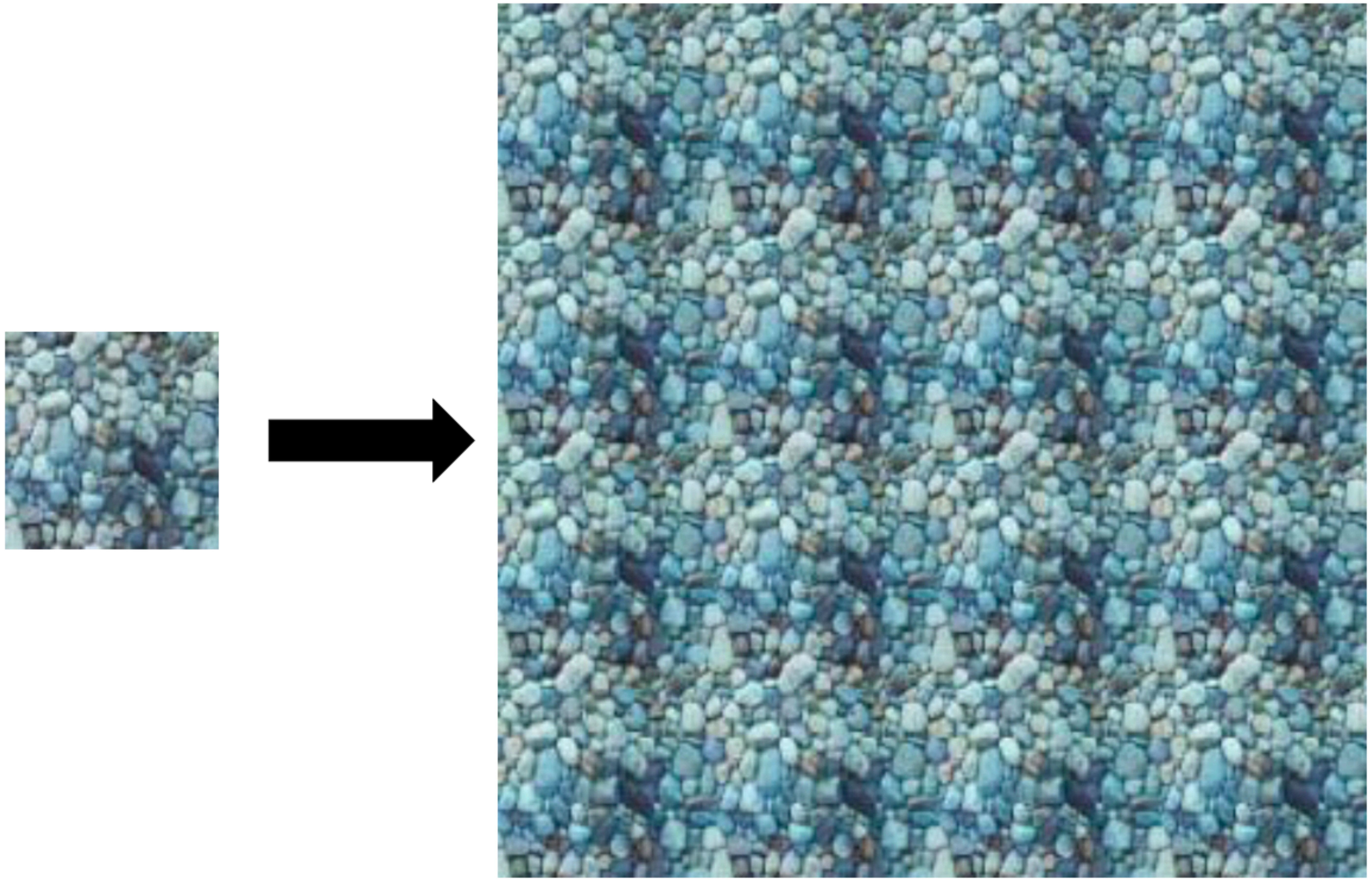
granite

Efros and Leung

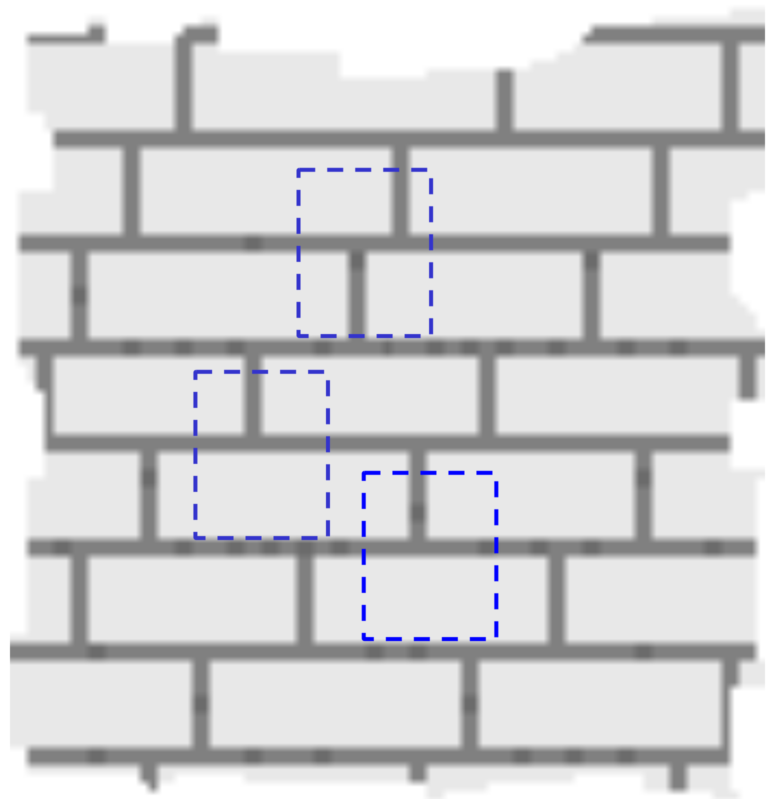


white bread   brick wall

Like **Copying**, But not Just Repetition

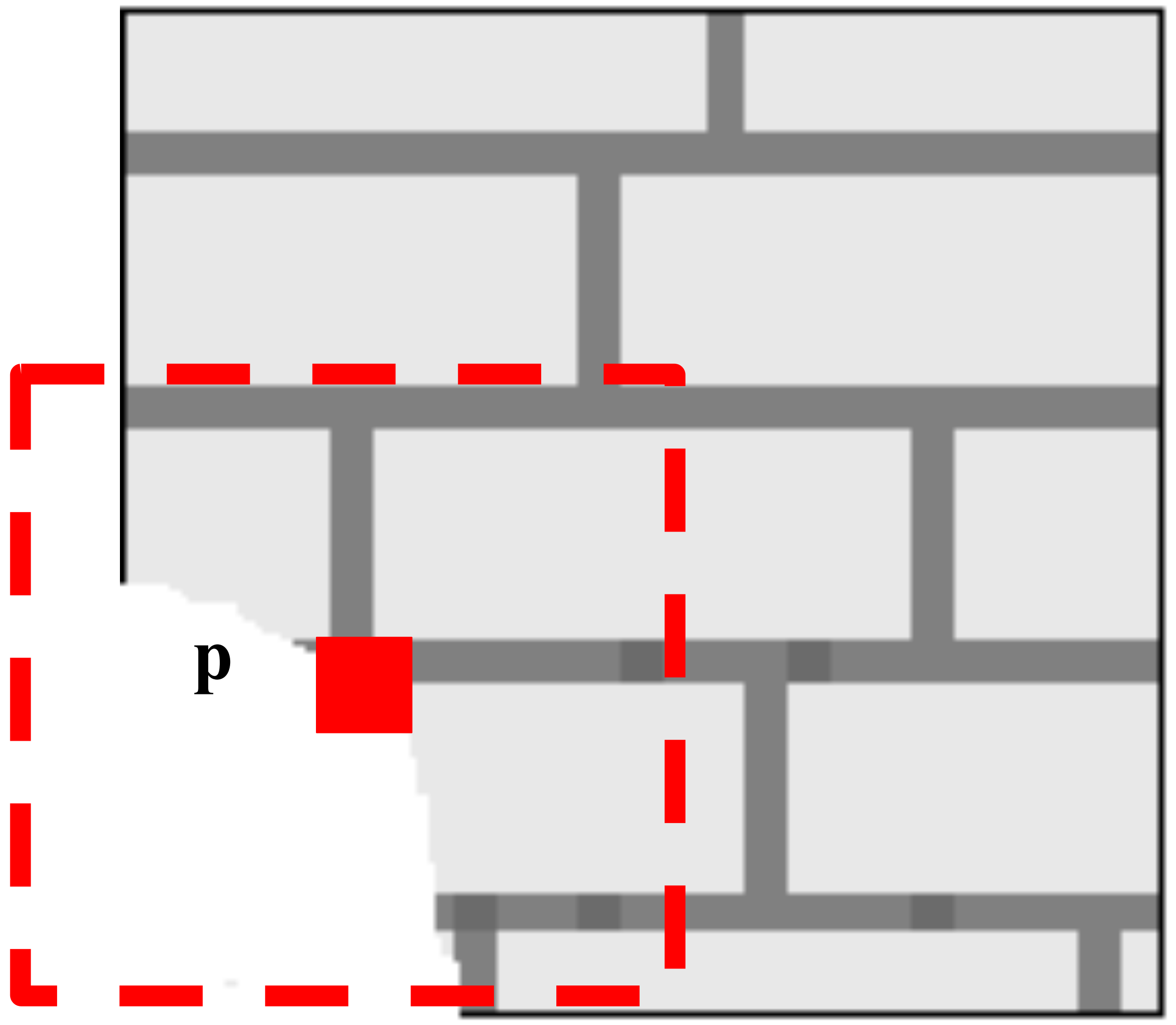


Efros and Leung: Synthesizing One Pixel



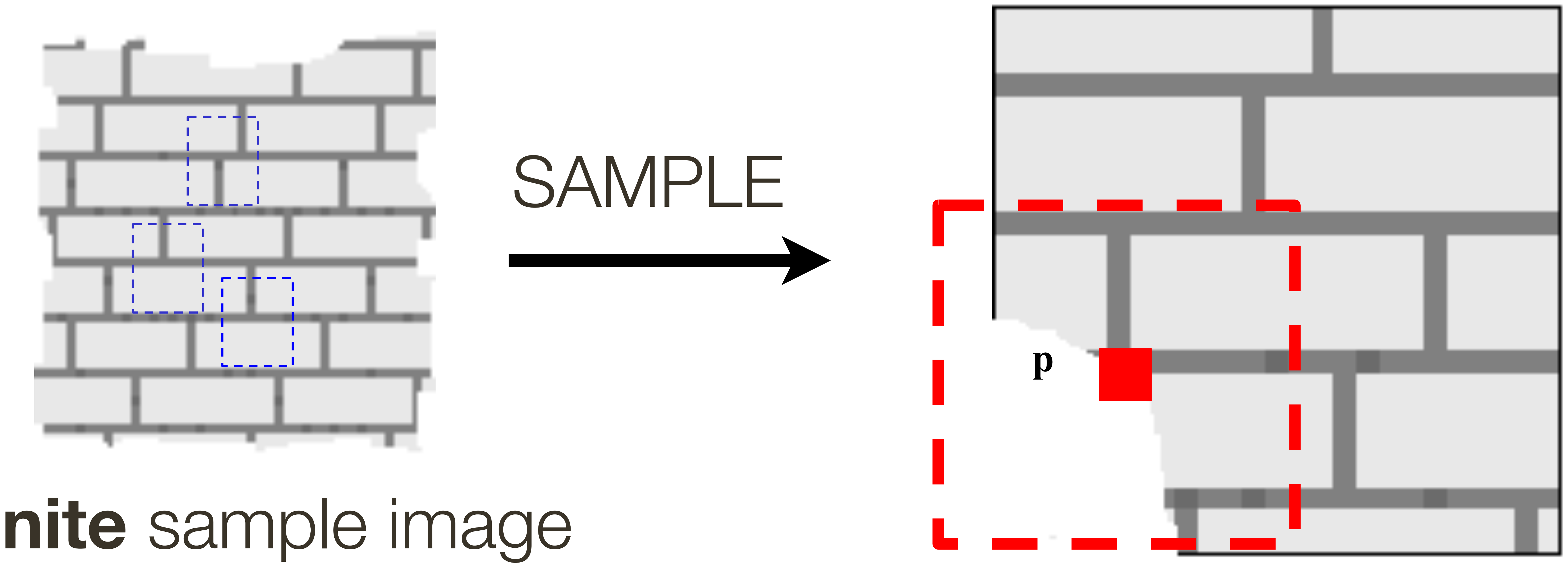
Infinite sample image

SAMPLE



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

Efros and Leung: Synthesizing One Pixel

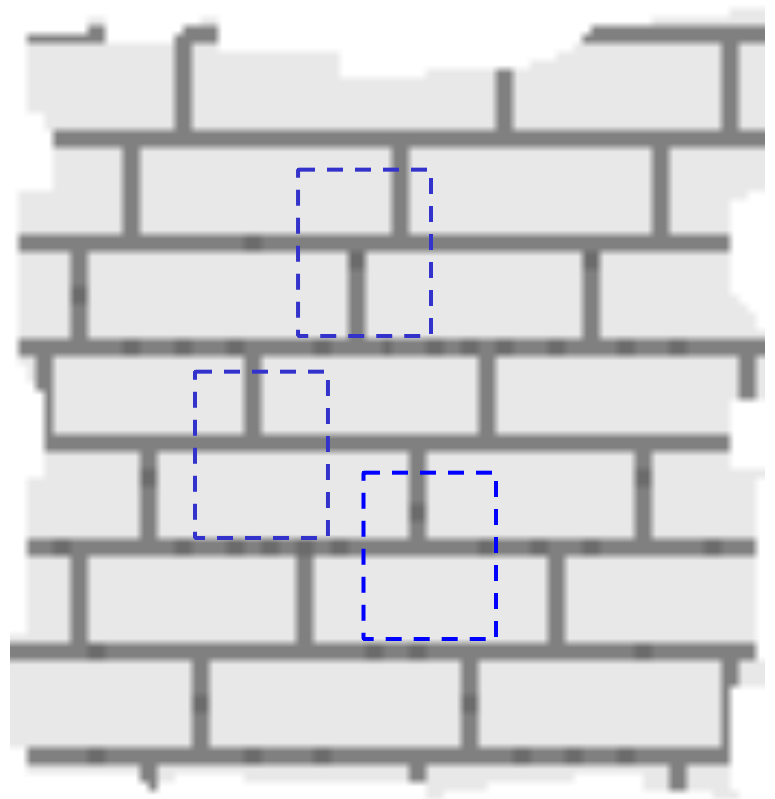


Infinite sample image

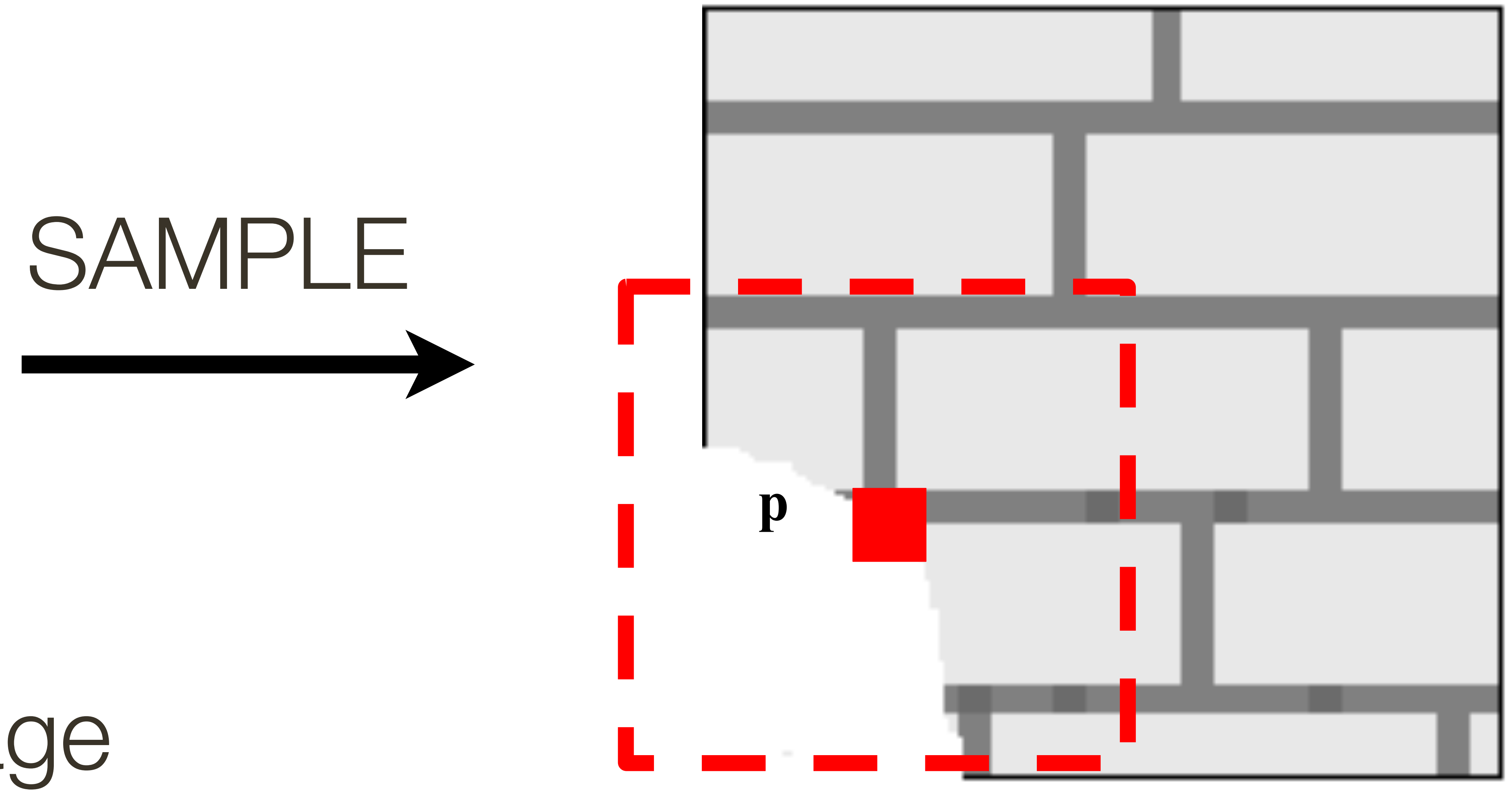
— What is **conditional** probability distribution of *p*, given the neighbourhood window?

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

Efros and Leung: Synthesizing One Pixel



Infinite sample image

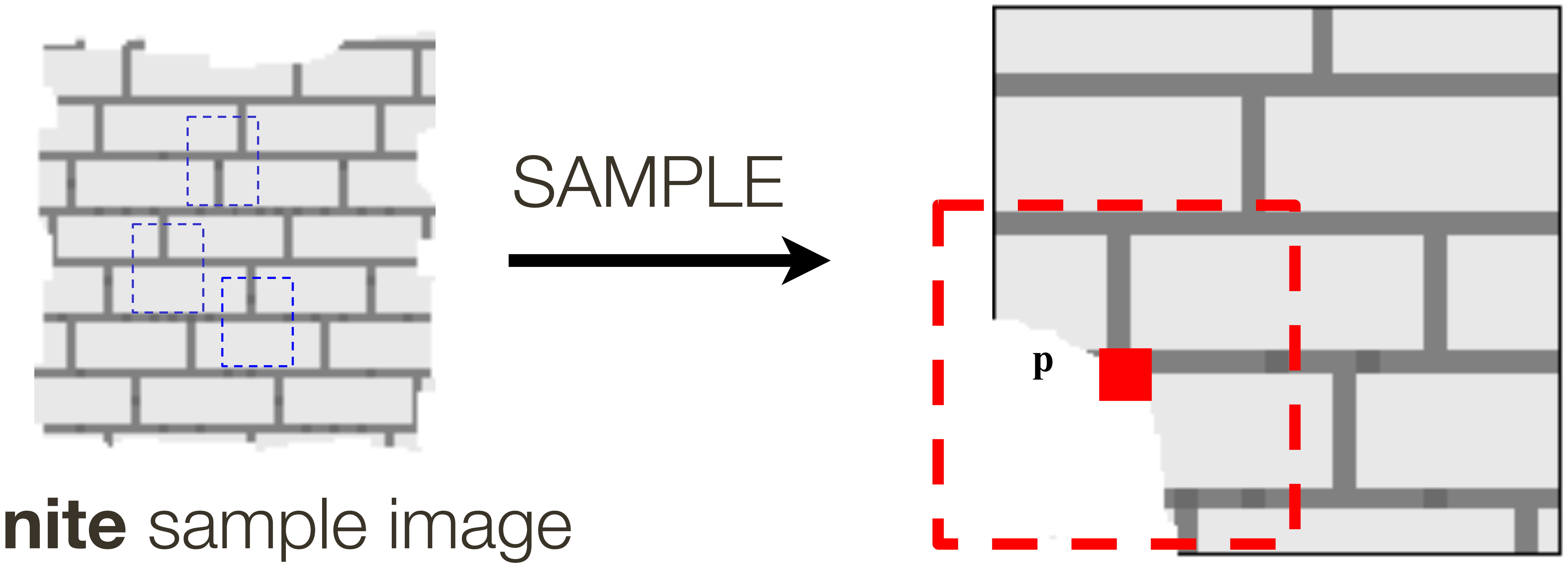


— What is **conditional** probability distribution of *p*, given the neighbourhood window?

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

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

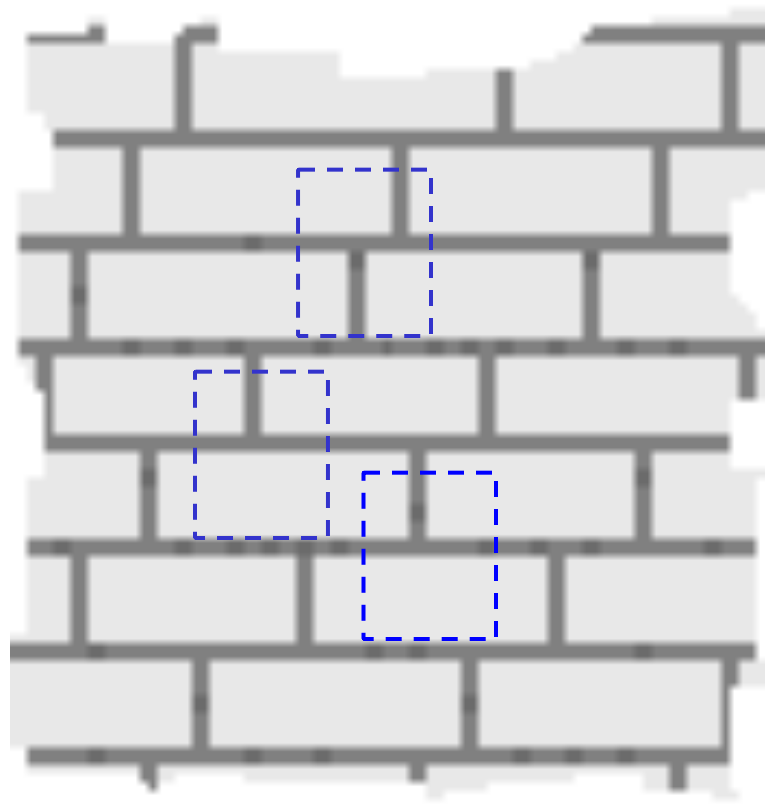
Efros and Leung: Synthesizing One Pixel



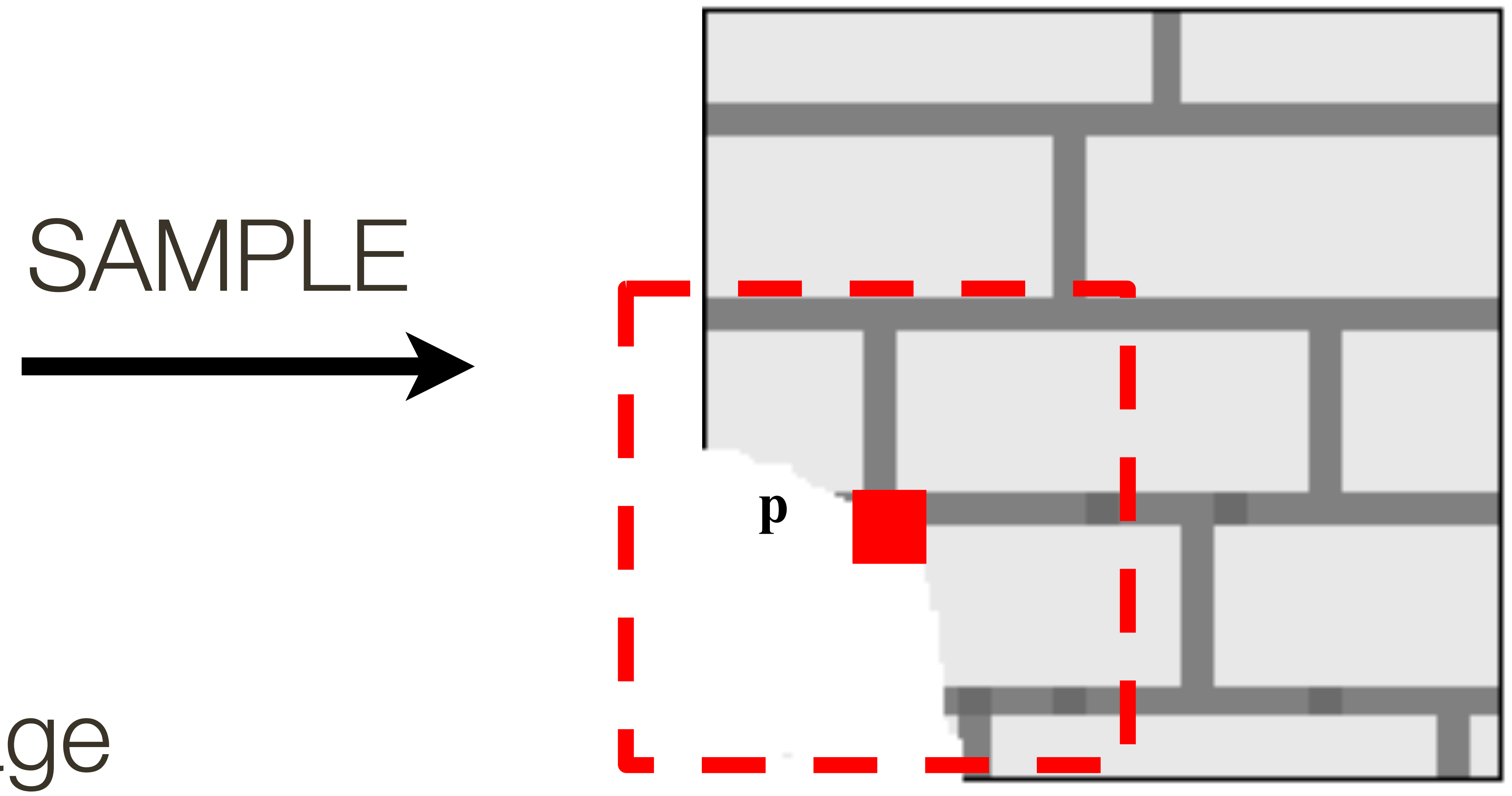
Infinite sample image

— 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

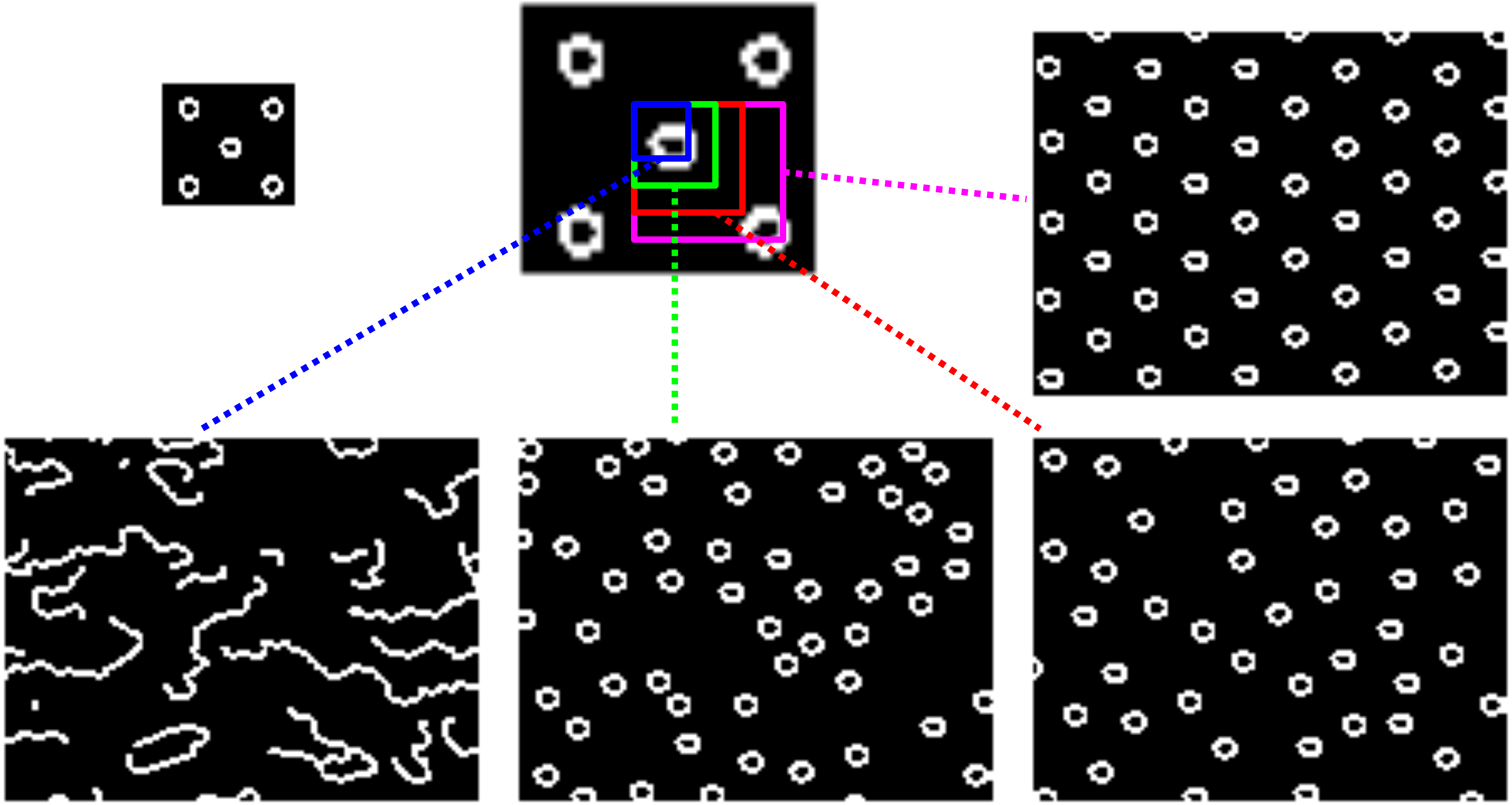
— In the case of hole-filling, start from the edges of the hole

For an interactive demo, see

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

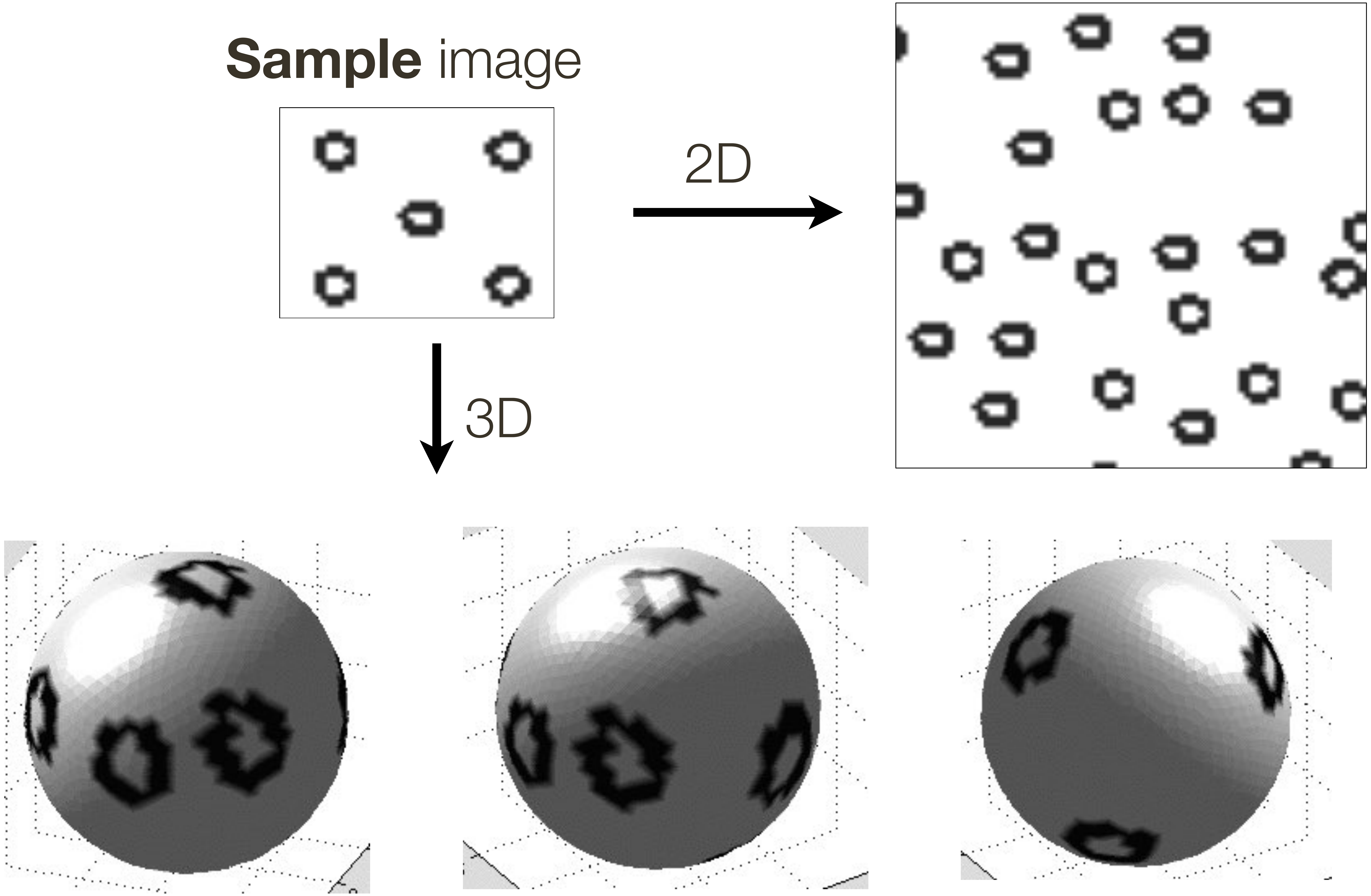
(written by Julieta Martinez, a previous CPSC 425 TA)

Randomness Parameter



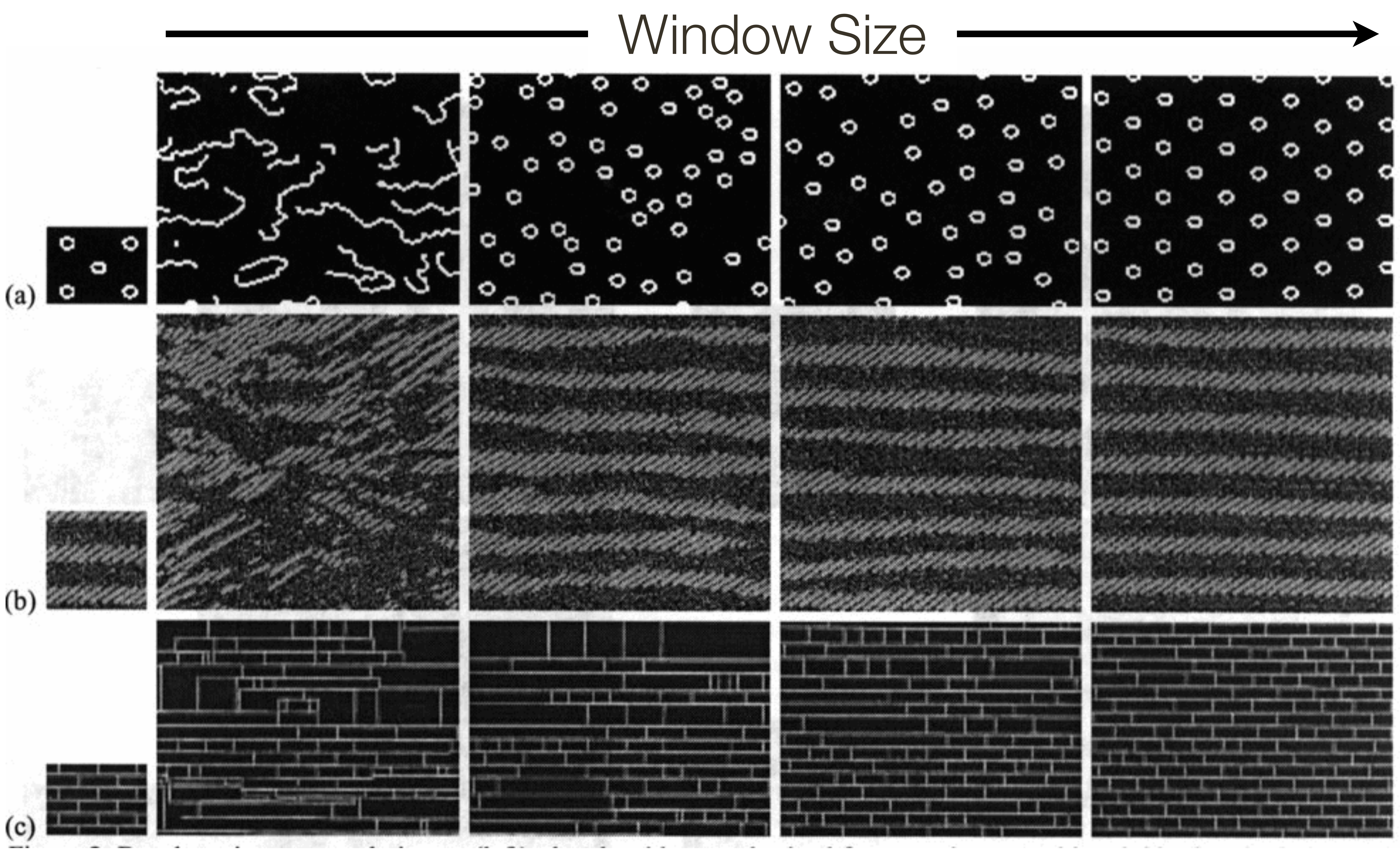
Slide Credit: http://graphics.cs.cmu.edu/people/efros/research/NPS/efros-iccv99.ppt

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

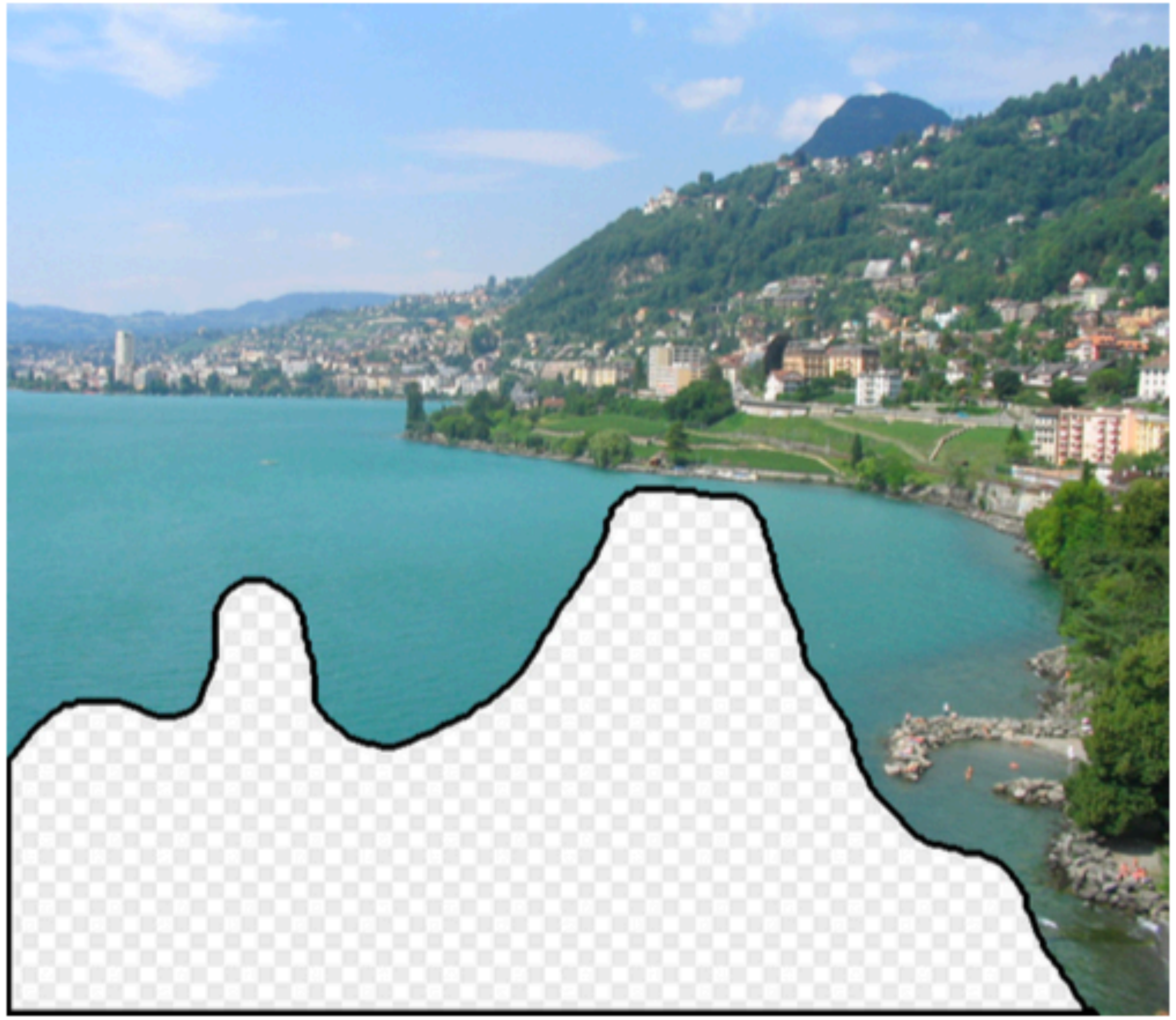
“**Big** Data” Meets Inpainting

“**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

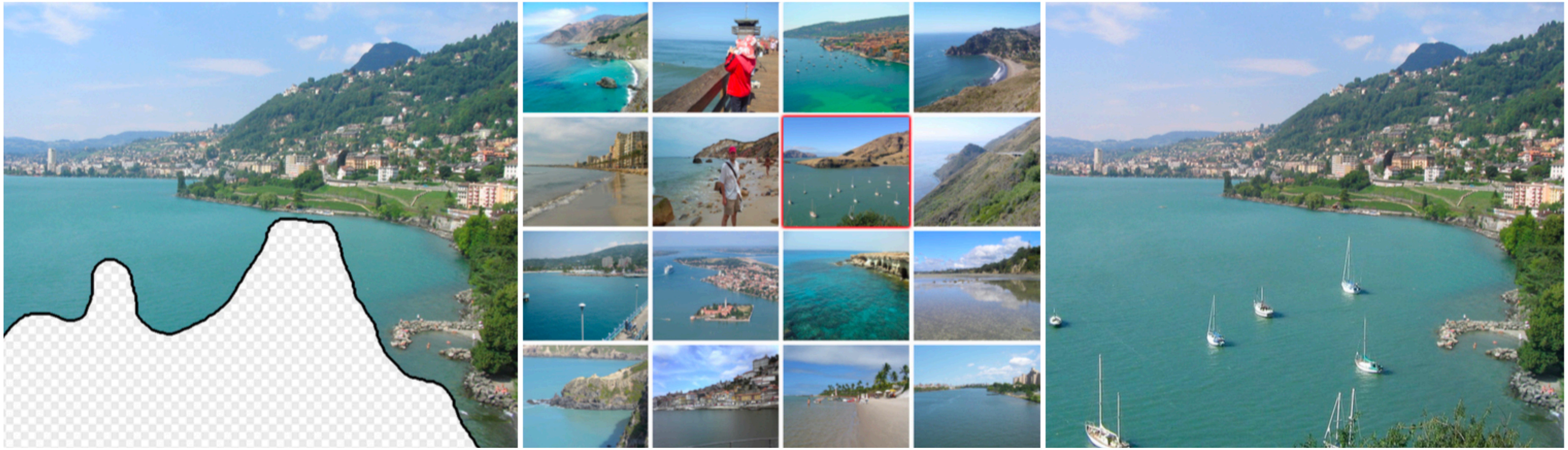
Original Image



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"

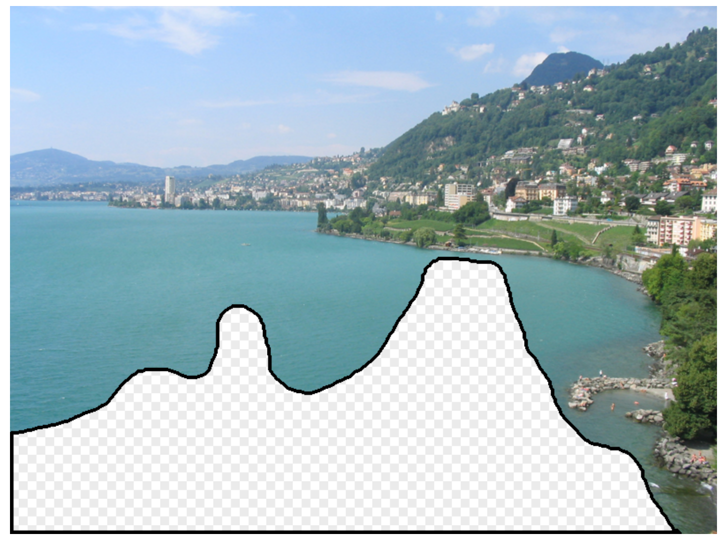
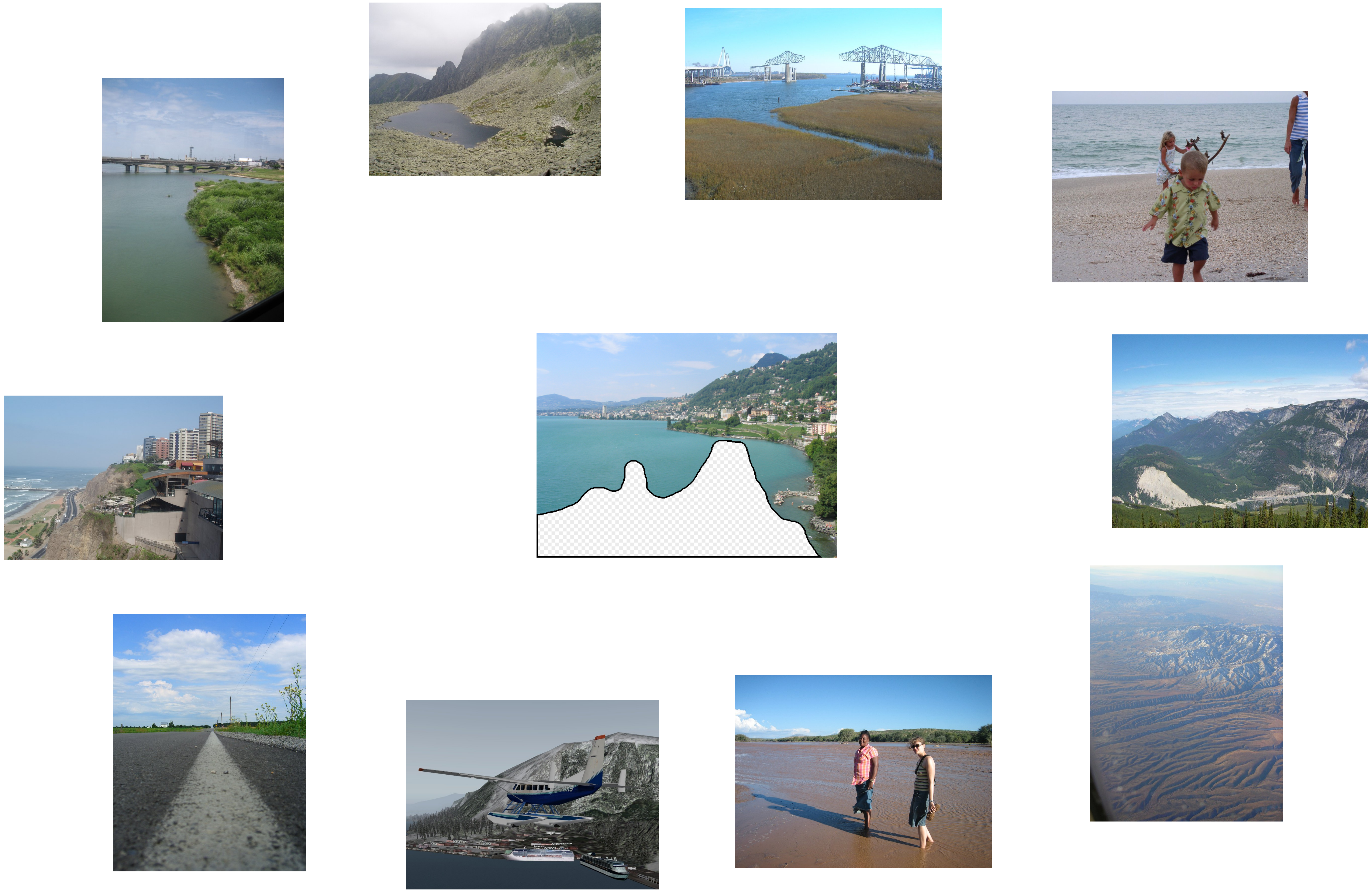


Figure Credit: Hays and Efros 2007

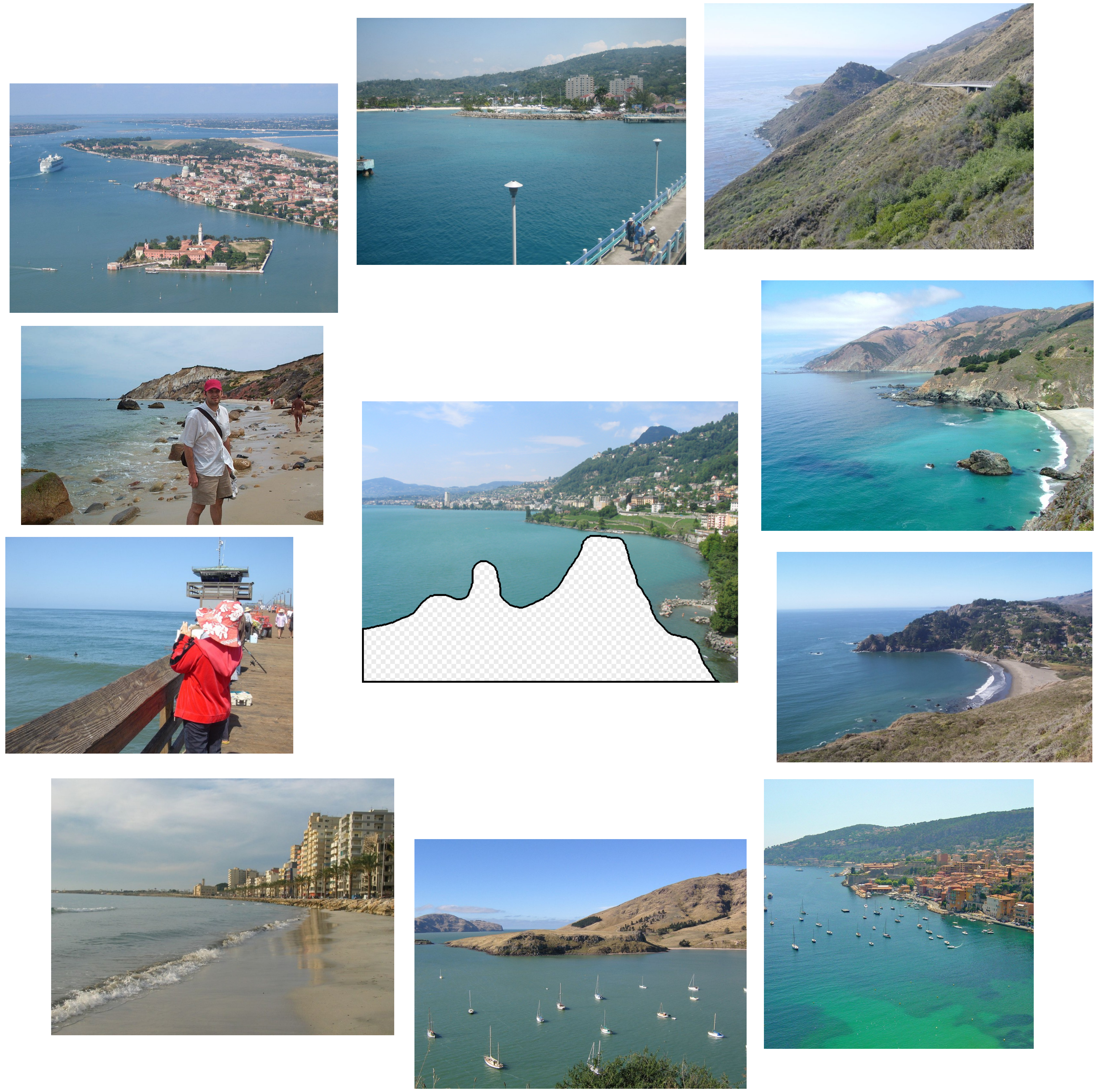
Effectiveness of “Big Data”



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

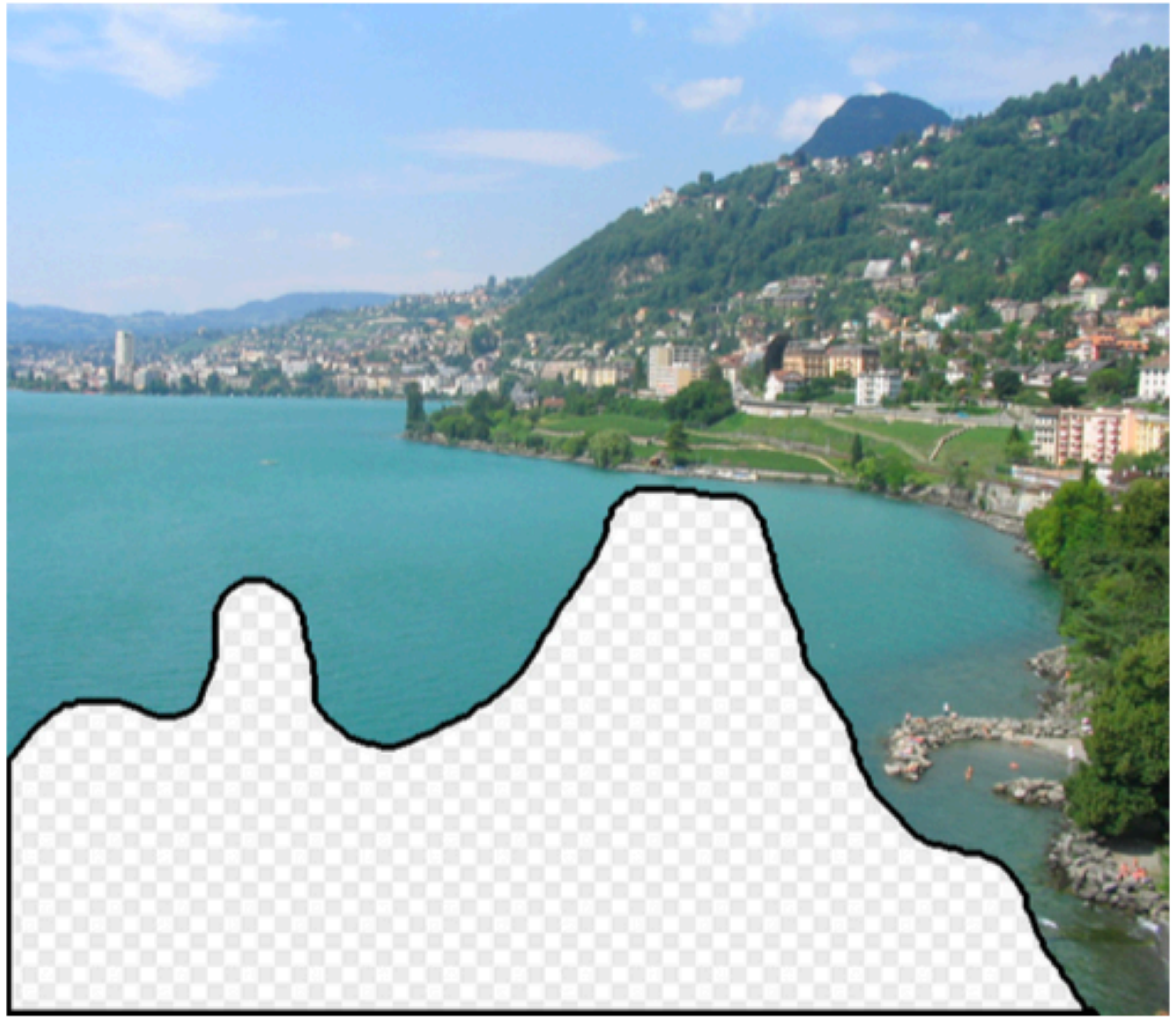
2. Find patches in the short list that match the context surrounding the image region we want to fill

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

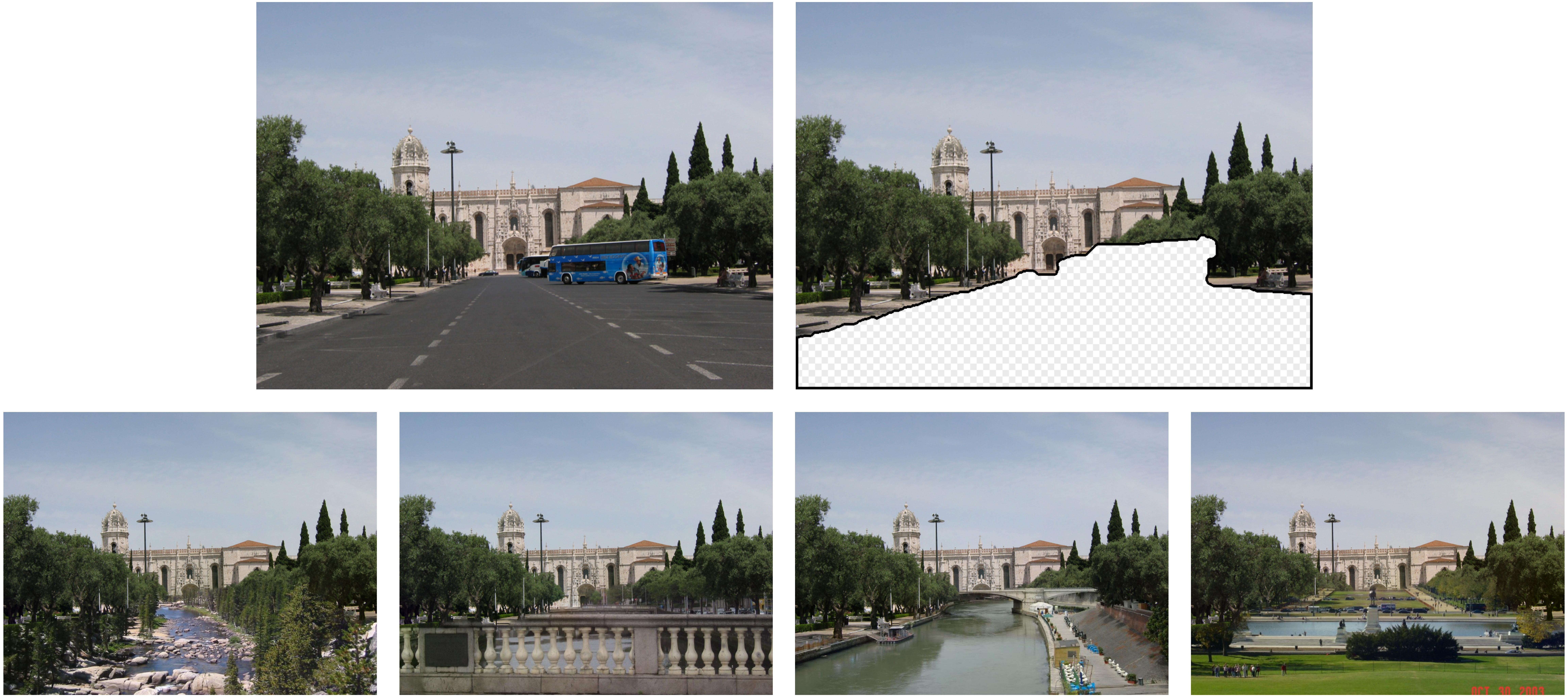
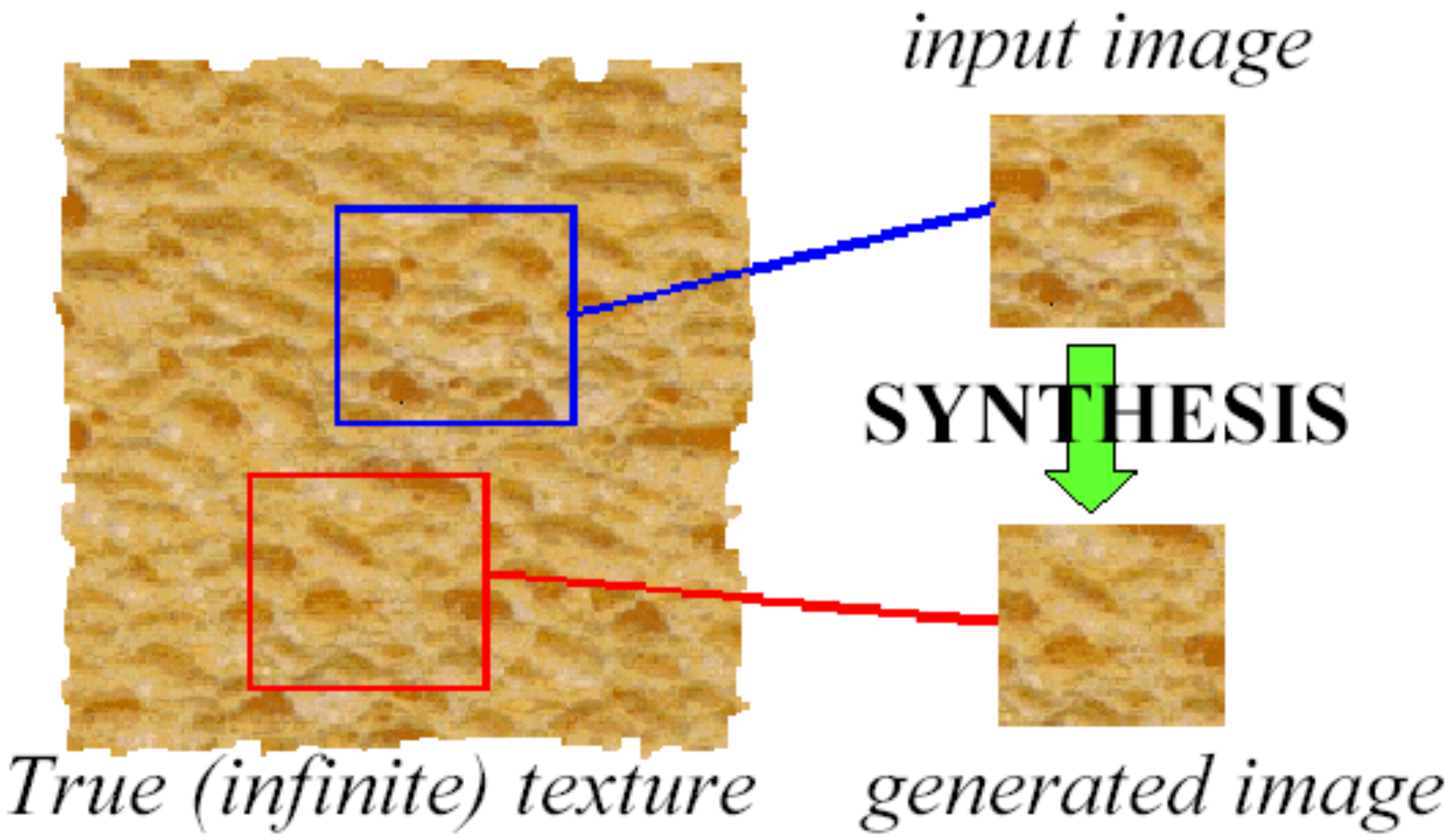


Figure Credit: Hays and Efros 2007

Goal of Texture **Synthesis**

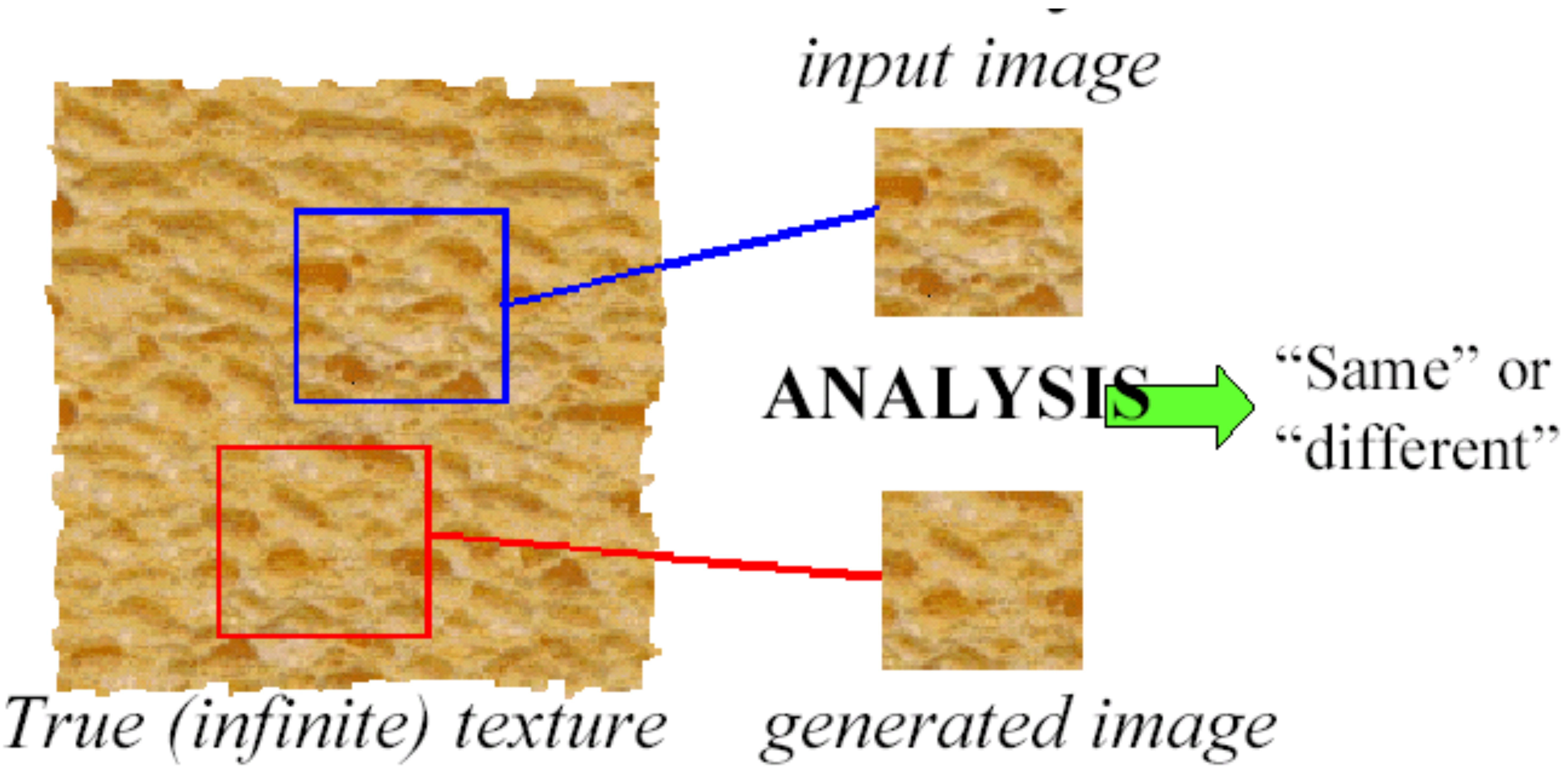


Given a finite sample of some texture, the goal is to synthesize other samples from that same texture

— The sample needs to be "large enough“

Credit: Bill Freeman

Goal of Texture **Analysis**



Compare textures and decide if they're mae of the same "stuff"

Credit: Bill Freeman

Definition of Texture (Re-Cap)

Recall that texture is easy to recognize but hard to define

— A functional definition was presented last class

We need representations that differ in ways that are easy to observe when two textures are significantly different.

Recall that textures can often be thought of as patterns composed of repeated instances of one (or more) identifiable elements, called **textons**

— e.g. bricks in a wall, spots on a cheetah

Texture **Segmentation**

Question: Is texture a property of a point or a property of a region?

Texture **Segmentation**

Question: Is texture a property of a point or a property of a region?

Answer: We need a region to have a texture.

Texture **Segmentation**

Question: Is texture a property of a point or a property of a region?

Answer: We need a region to have a texture.

There is a “chicken–and–egg” problem. Texture segmentation can be done by detecting boundaries between regions of the same (or similar) texture. Texture boundaries can be detected using standard edge detection techniques applied to the texture measures determined at each point

Texture **Segmentation**

Question: Is texture a property of a point or a property of a region?

Answer: We need a region to have a texture.

There is a “chicken–and–egg” problem. Texture segmentation can be done by detecting boundaries between regions of the same (or similar) texture. Texture boundaries can be detected using standard edge detection techniques applied to the texture measures determined at each point

We compromise! Typically one uses a local window to estimate texture properties and assigns those texture properties as point properties of the window’s center row and column

Texture **Representation**

Question: How many degrees of freedom are there to texture?

(**Mathematical**) Answer: Infinitely many

(**Perceptual Psychology**) Answer: There are perceptual constraints. But, there is no clear notion of a “texture channel” like, for example, there is for an RGB colour channel

Texture **Representation**

Observation: Textures are made up of generic sub-elements, repeated over a region with similar statistical properties

Idea: Find the sub-elements with filters, then represent each point in the image with a summary of the pattern of sub-elements in the local region

Texture **Representation**

Observation: Textures are made up of generic sub-elements, repeated over a region with similar statistical properties

Idea: Find the sub-elements with filters, then represent each point in the image with a summary of the pattern of sub-elements in the local region

Question: What filters should we use?

Answer: Human vision suggests spots and oriented edge filters at a variety of different orientations and scales

Texture **Representation**

Observation: Textures are made up of generic sub-elements, repeated over a region with similar statistical properties

Idea: Find the sub-elements with filters, then represent each point in the image with a summary of the pattern of sub-elements in the local region

Question: What filters should we use?

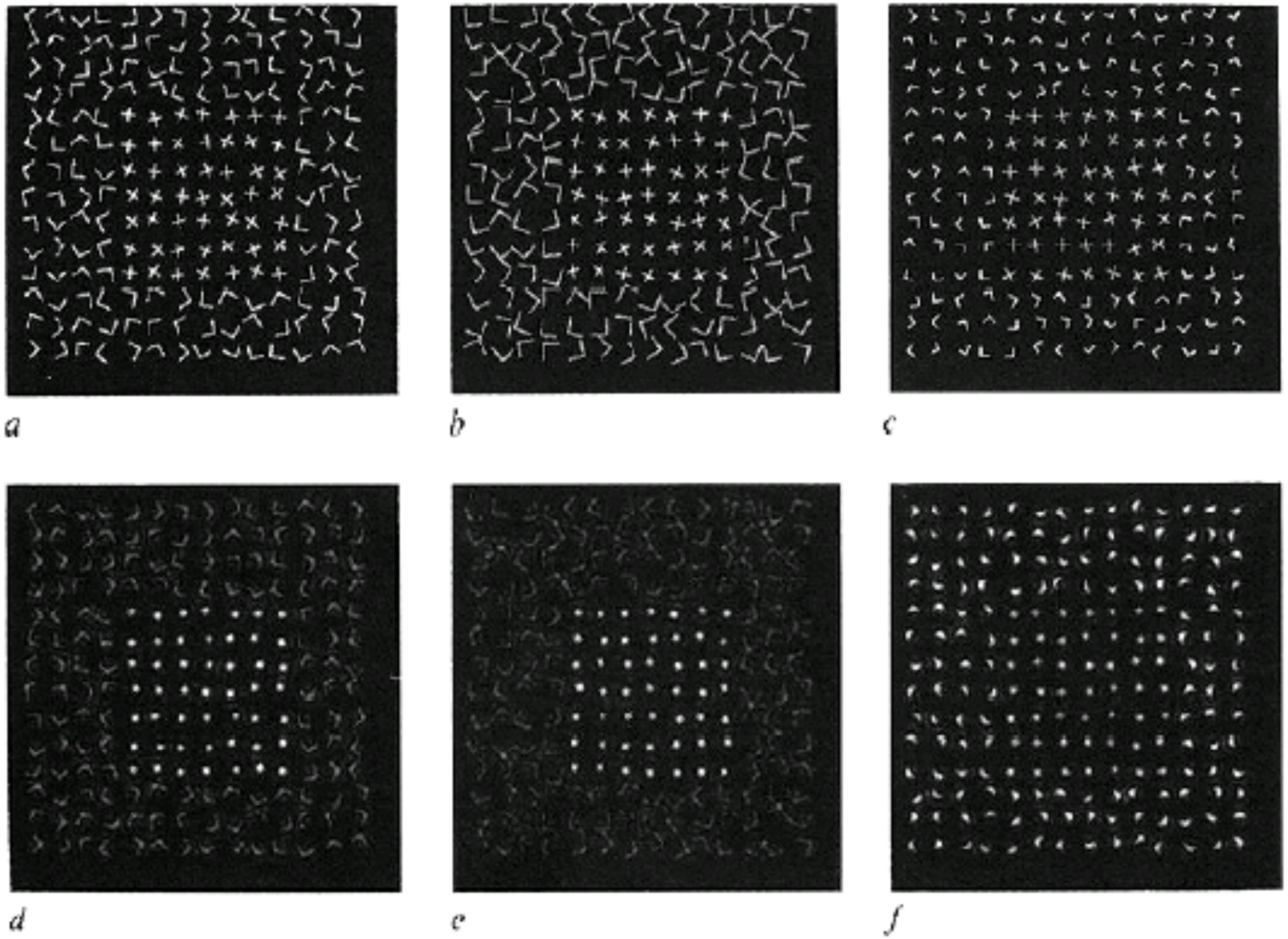
Answer: Human vision suggests spots and oriented edge filters at a variety of different orientations and scales

Question: How do we “summarize”?

Answer: Compute the mean or maximum of each filter response over the region
— Other statistics can also be useful

Human Texture Perception

Fig. 1 *Top row,* Textures consisting of Xs within a texture composed of Ls. The micropatterns are placed at random orientations on a randomly perturbed lattice. *a,* The bars of the Xs have the same length as the bars of the Ls. *b,* The bars of the Ls have been lengthened by 25%, and the intensity adjusted for the same mean luminance. Discriminabitity is enhanced. *c,* The bars of the Ls have been shortened by 25%, and the intensity adjusted for the same mean luminance. Discriminabitity is impaired. *Bottom row:* the responses of a size-tuned mechanism *d,* response to image *a*; *e,* response to image *b*; *f,* response to image *c*.



Credit: Bergen and Adelson, Nature, 1988

Texture **Representation**

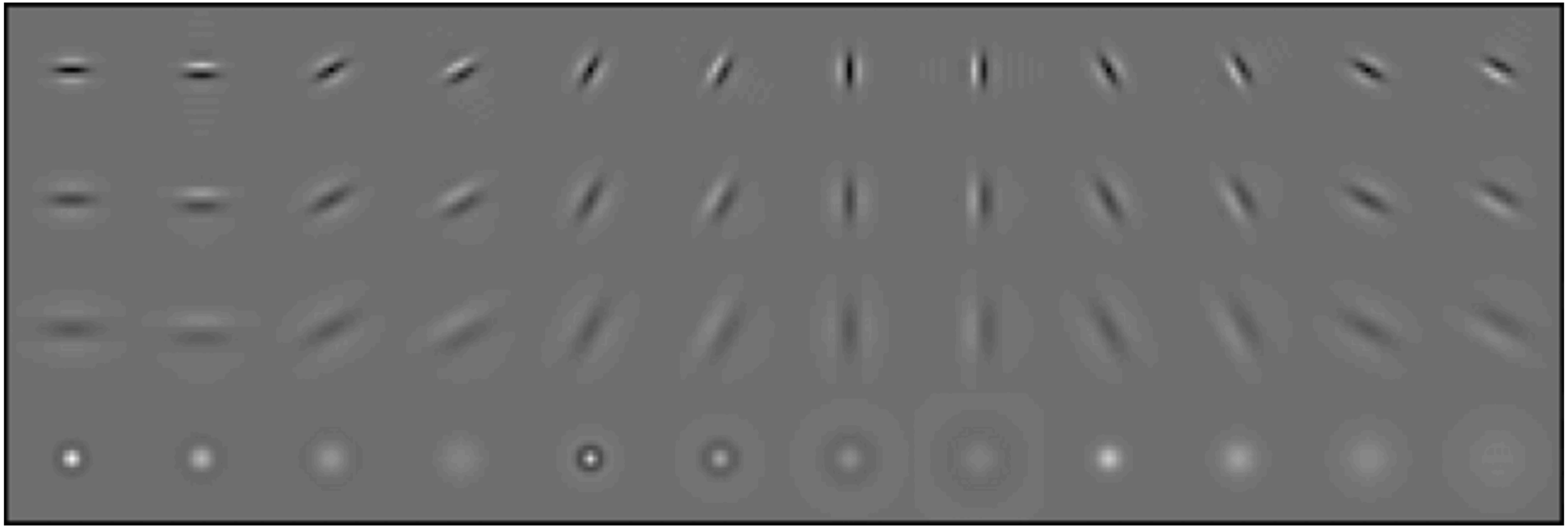
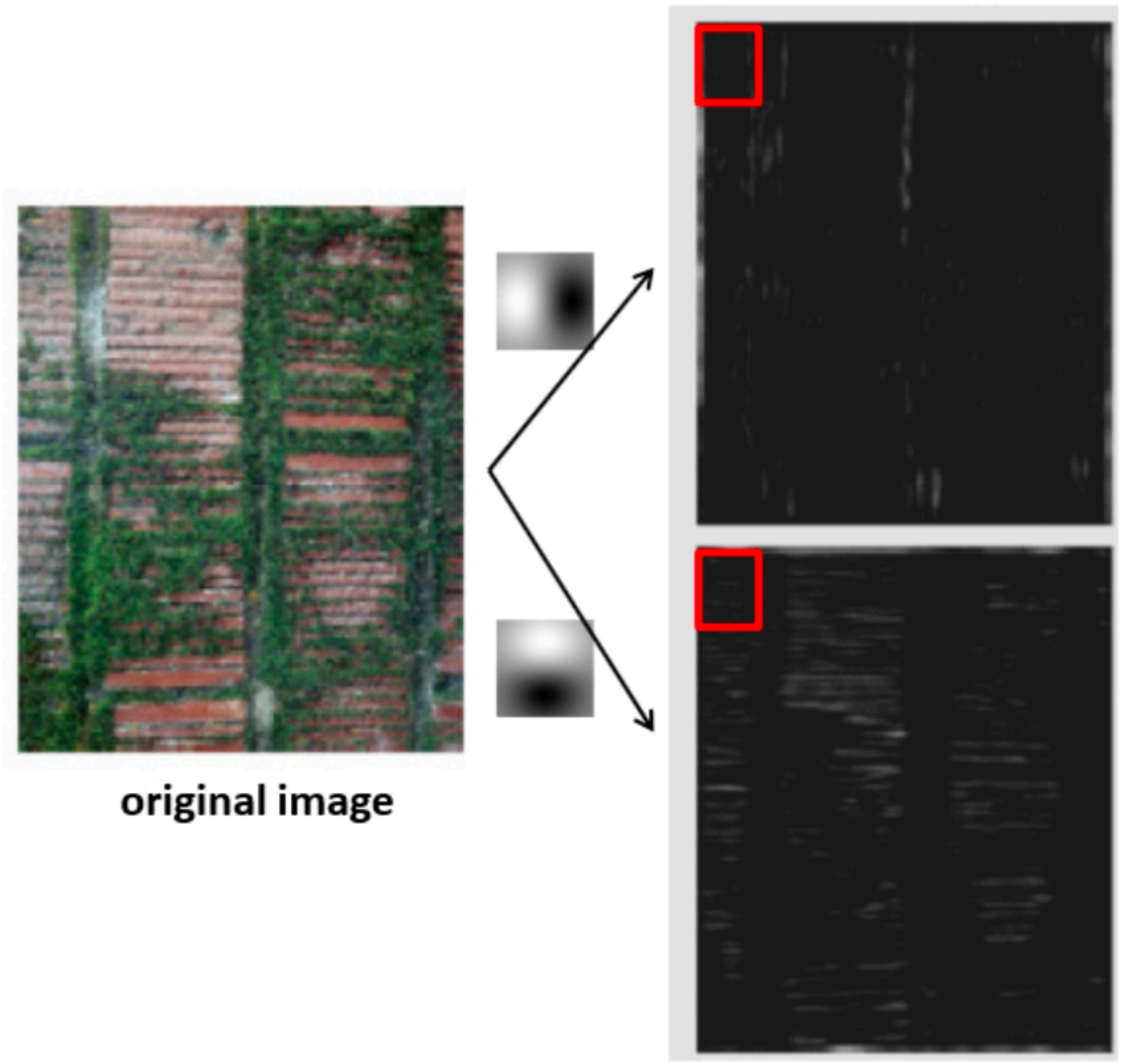


Figure Credit: Leung and Malik, 2001

Texture **Representation**



derivative filter responses, squared

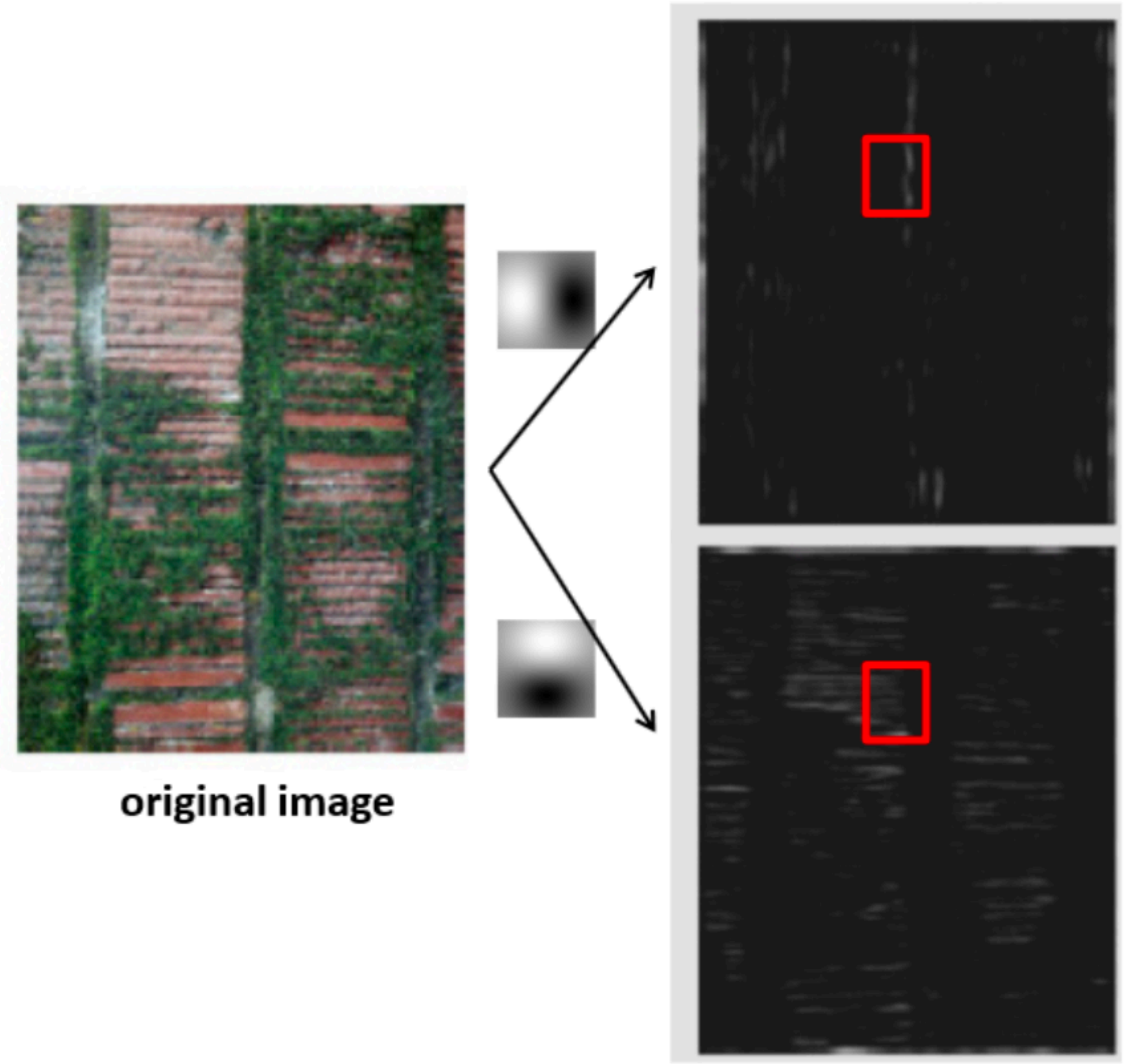
	mean d/dx value	mean d/dy value
Win. #1	4	10

⋮

statistics to summarize patterns in small windows

Slide Credit: Trevor Darrell

Texture **Representation**



original image

derivative filter responses, squared

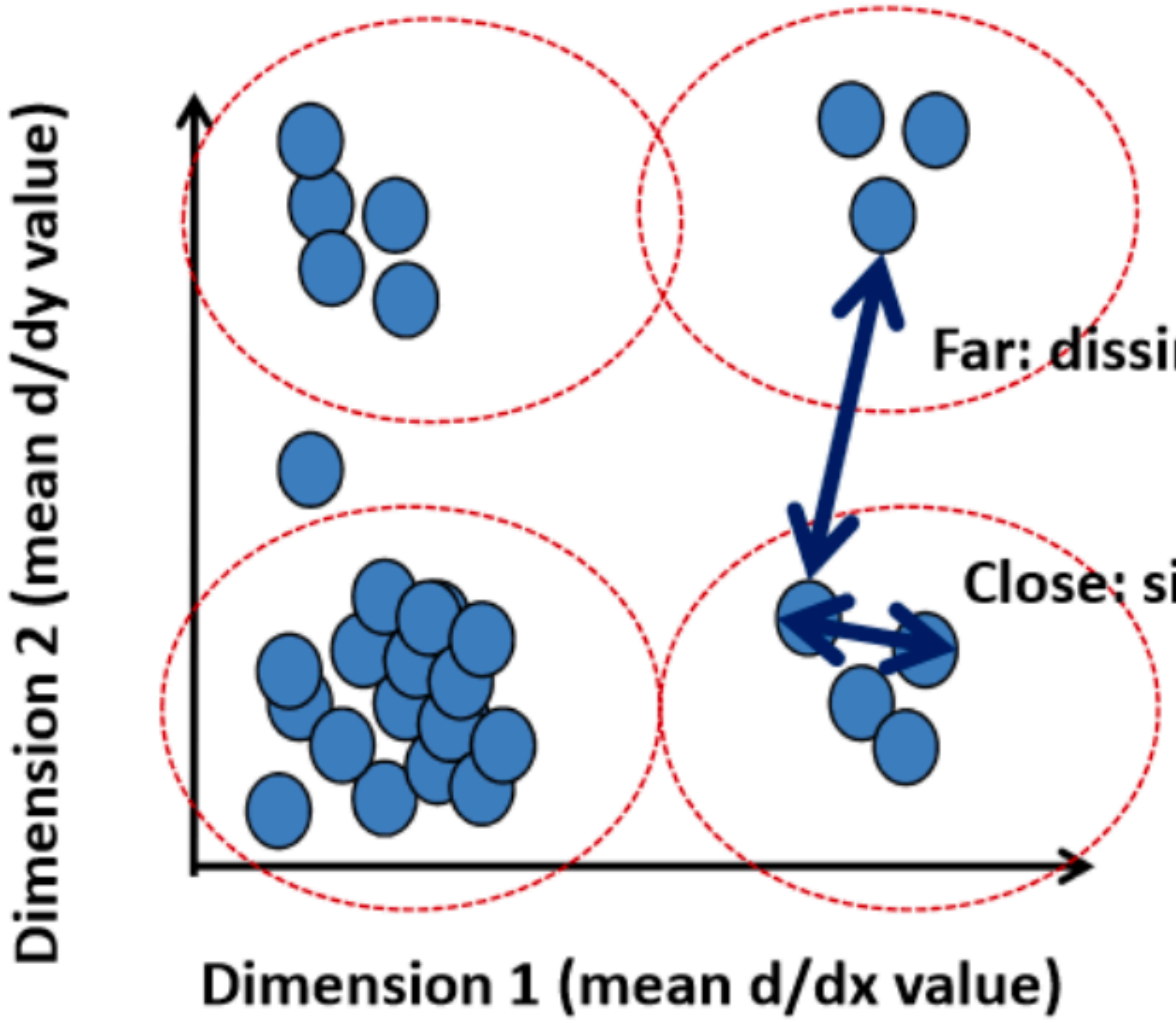
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2 ⋮	18	7
Win.#9	20	20

⋮

statistics to summarize patterns in small windows

Slide Credit: Trevor Darrell

Texture **Representation**



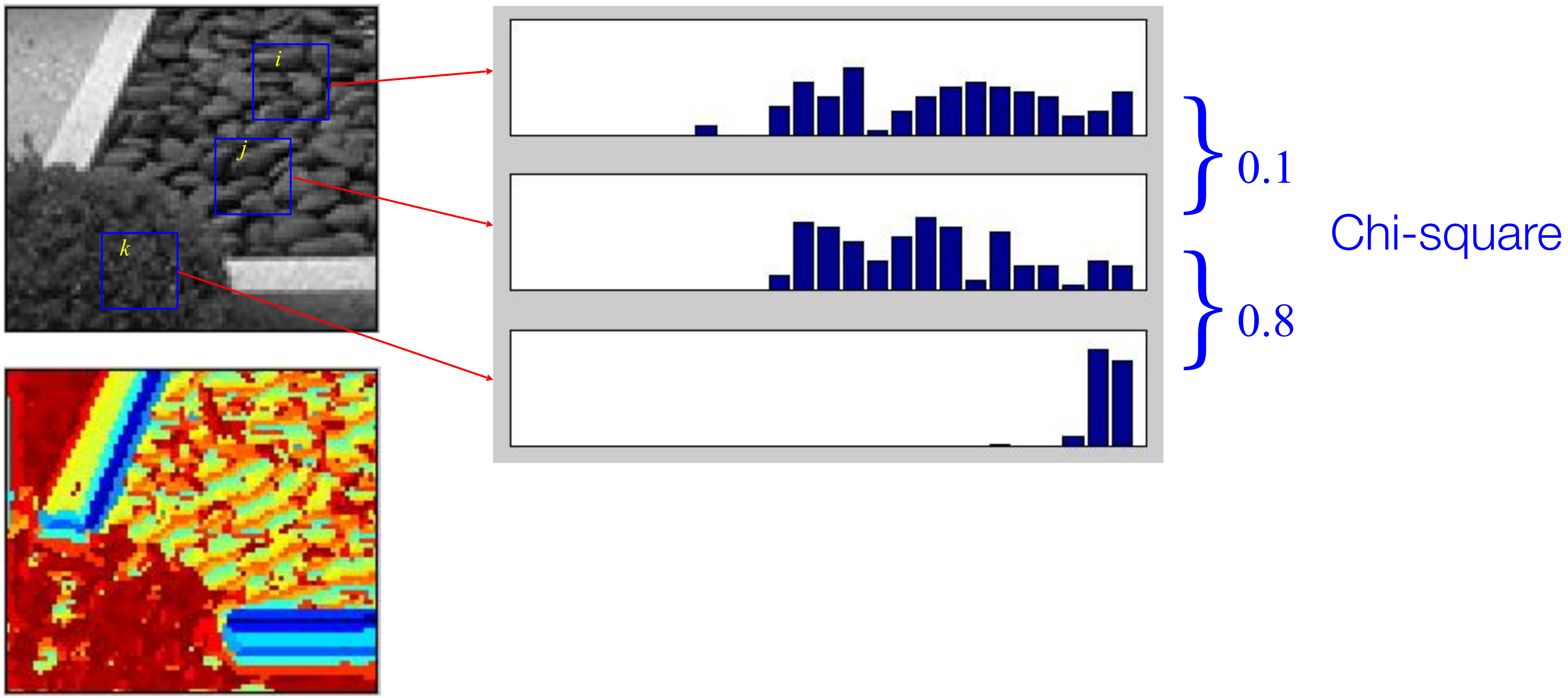
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

⋮

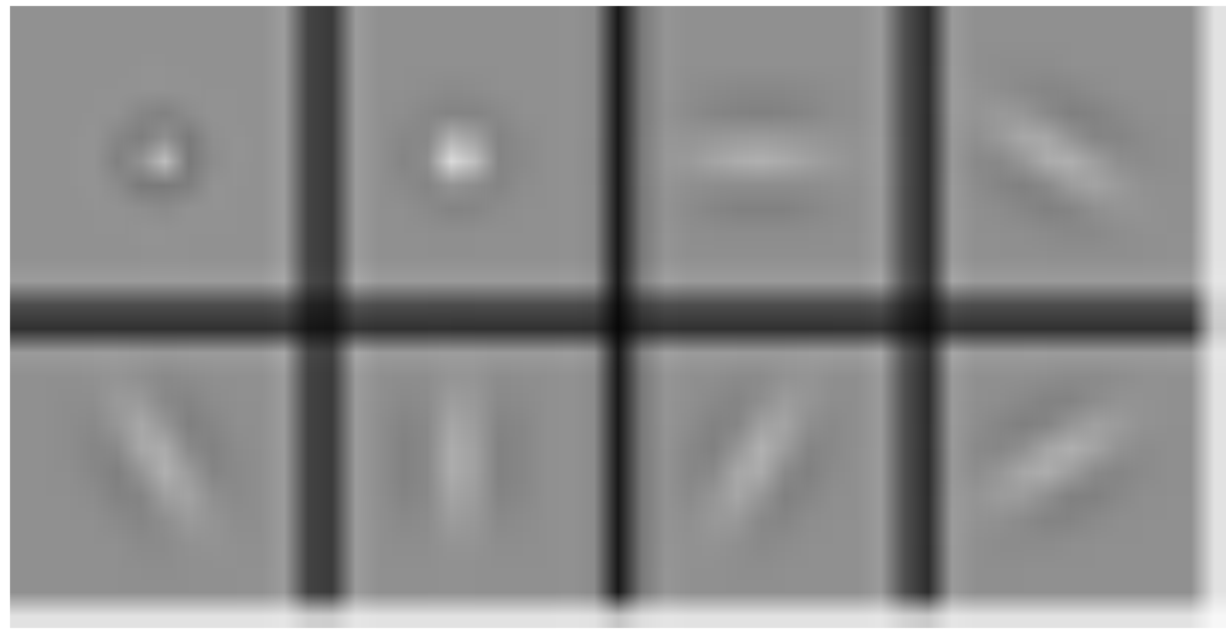
statistics to summarize patterns in small windows

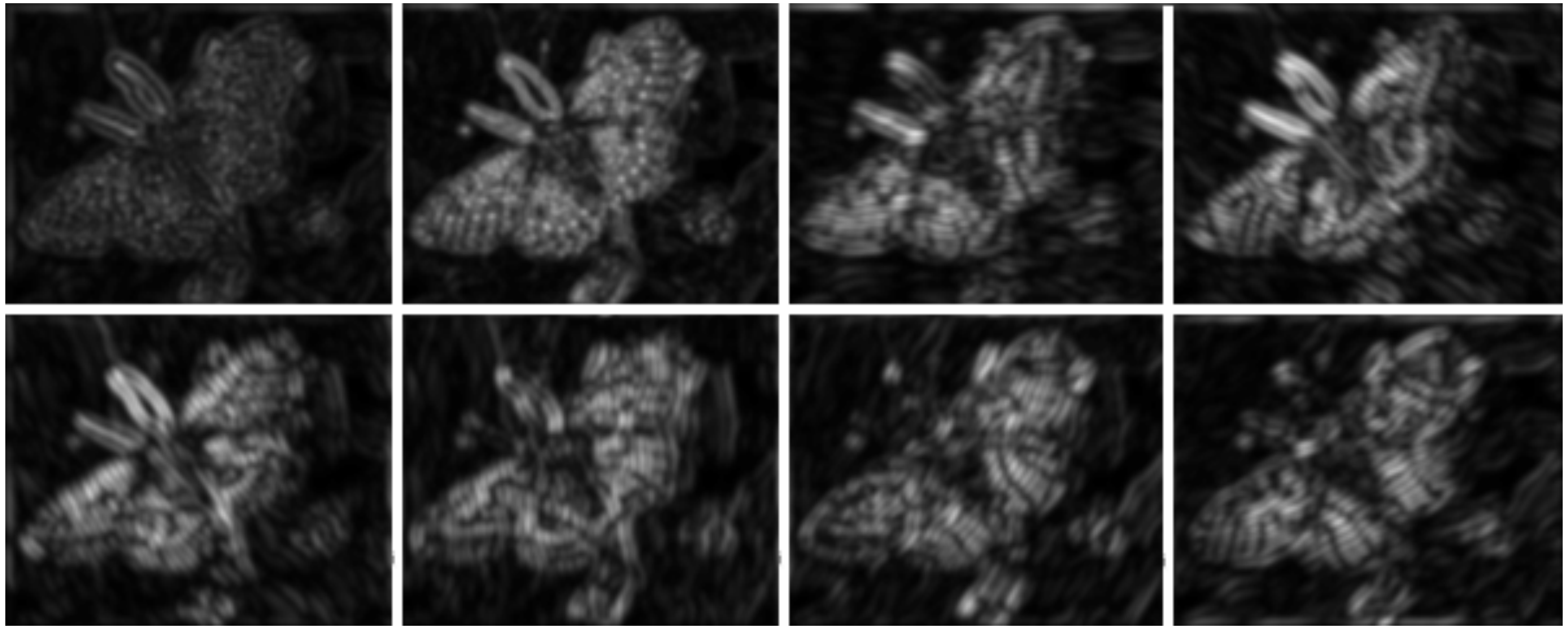
Slide Credit: Trevor Darrell

Texture **Representation**



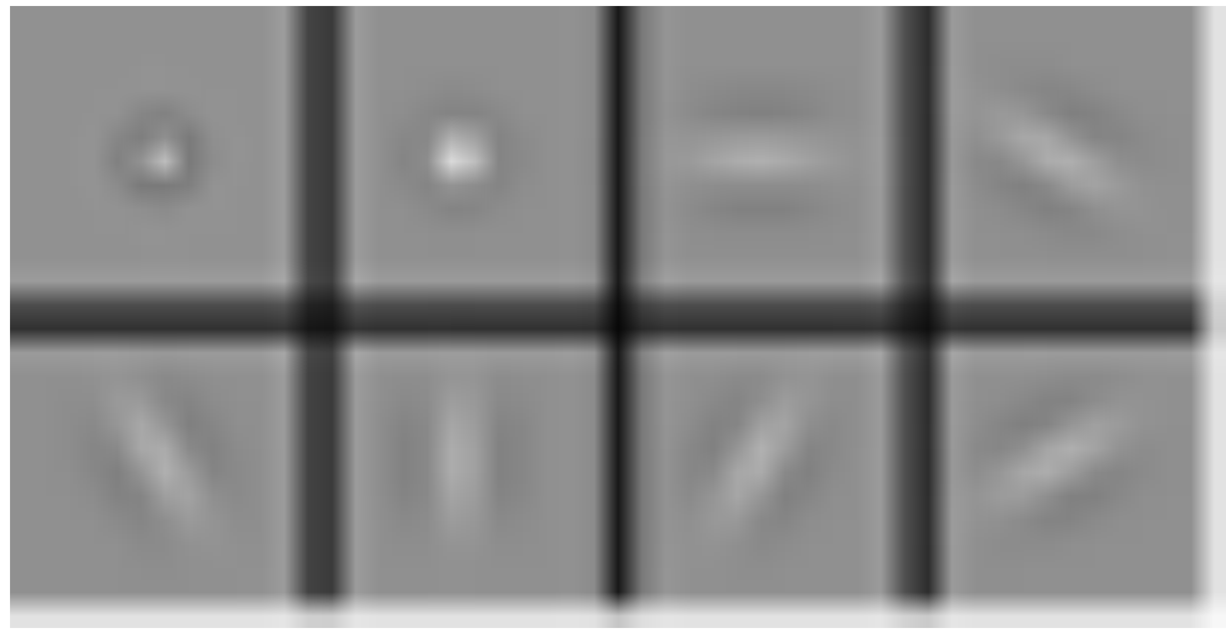
Spots and Bars (Fine Scale)

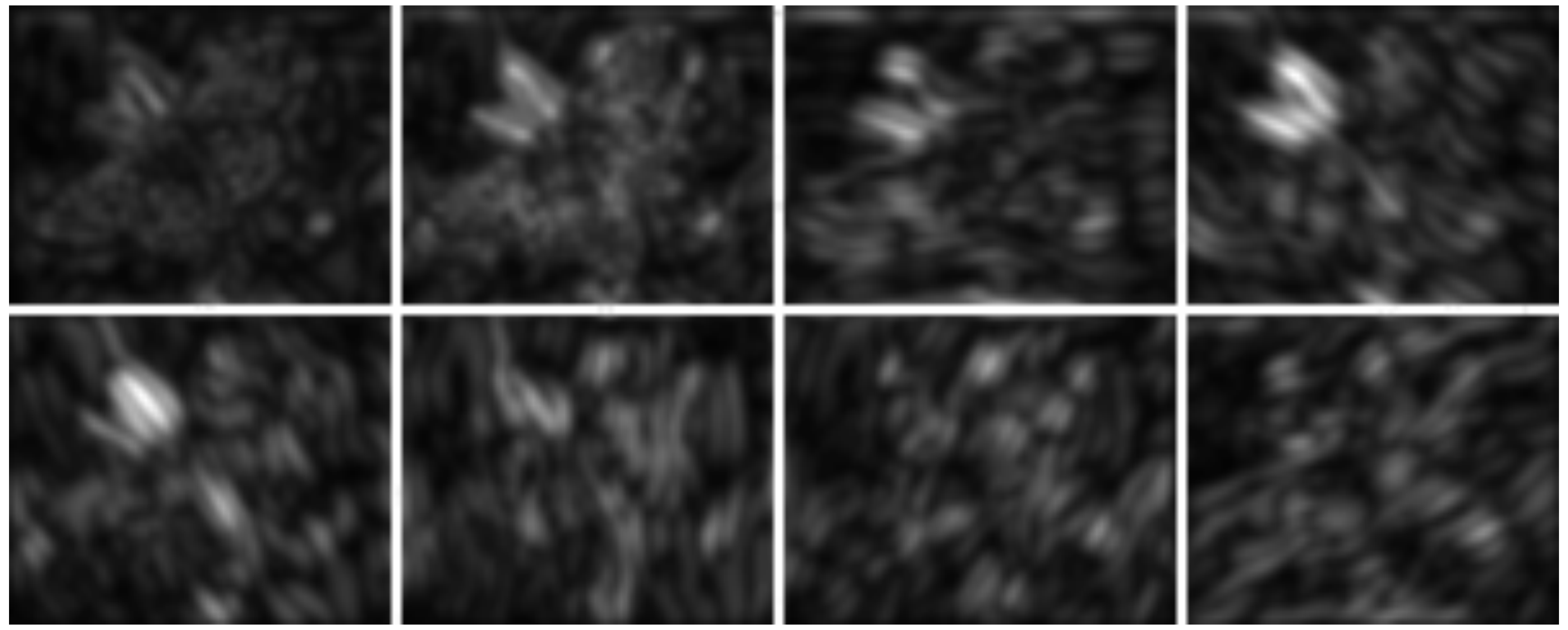




Forsyth & Ponce (1st ed.) Figures 9.3–9.4

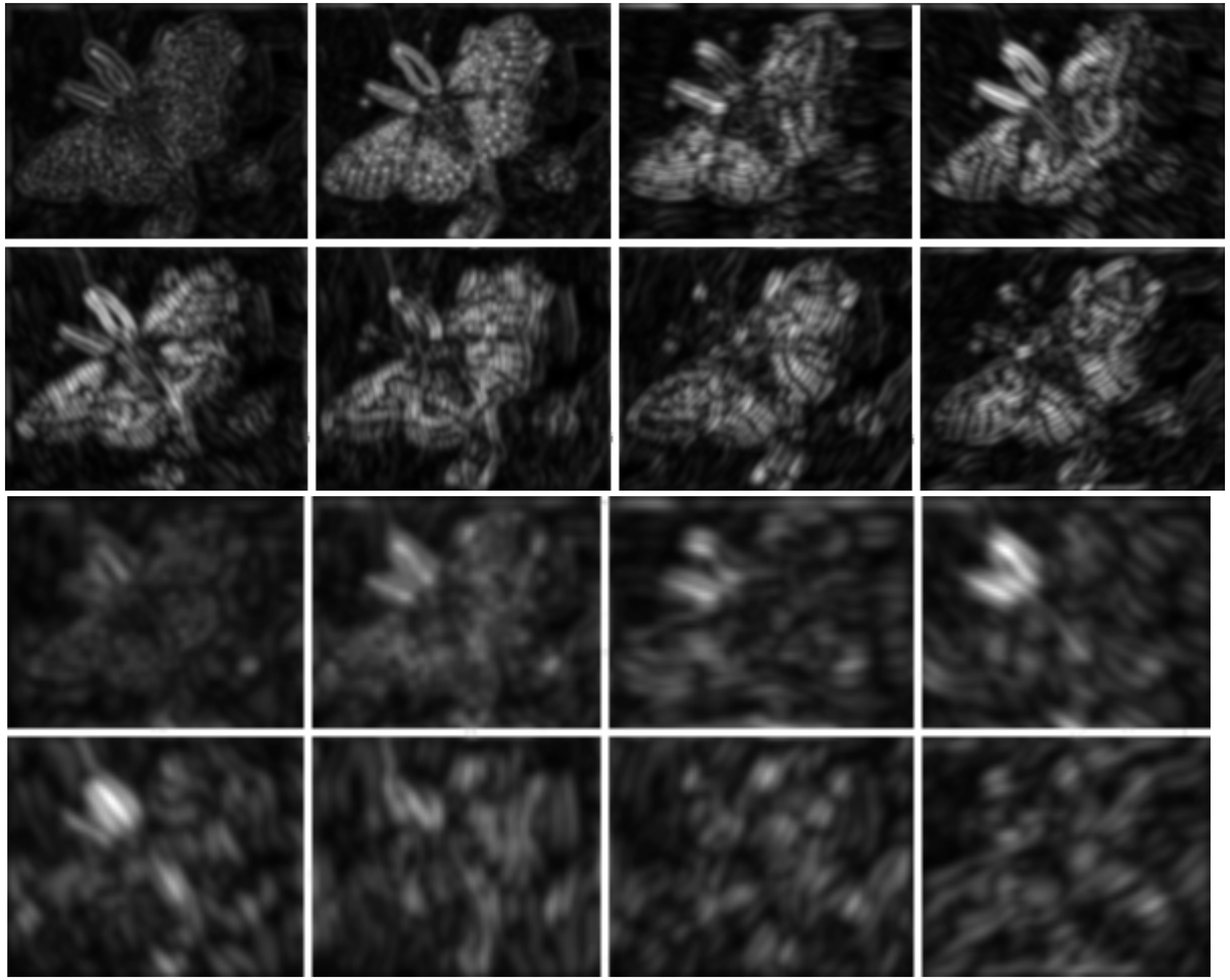
Spots and Bars (Coarse Scale)





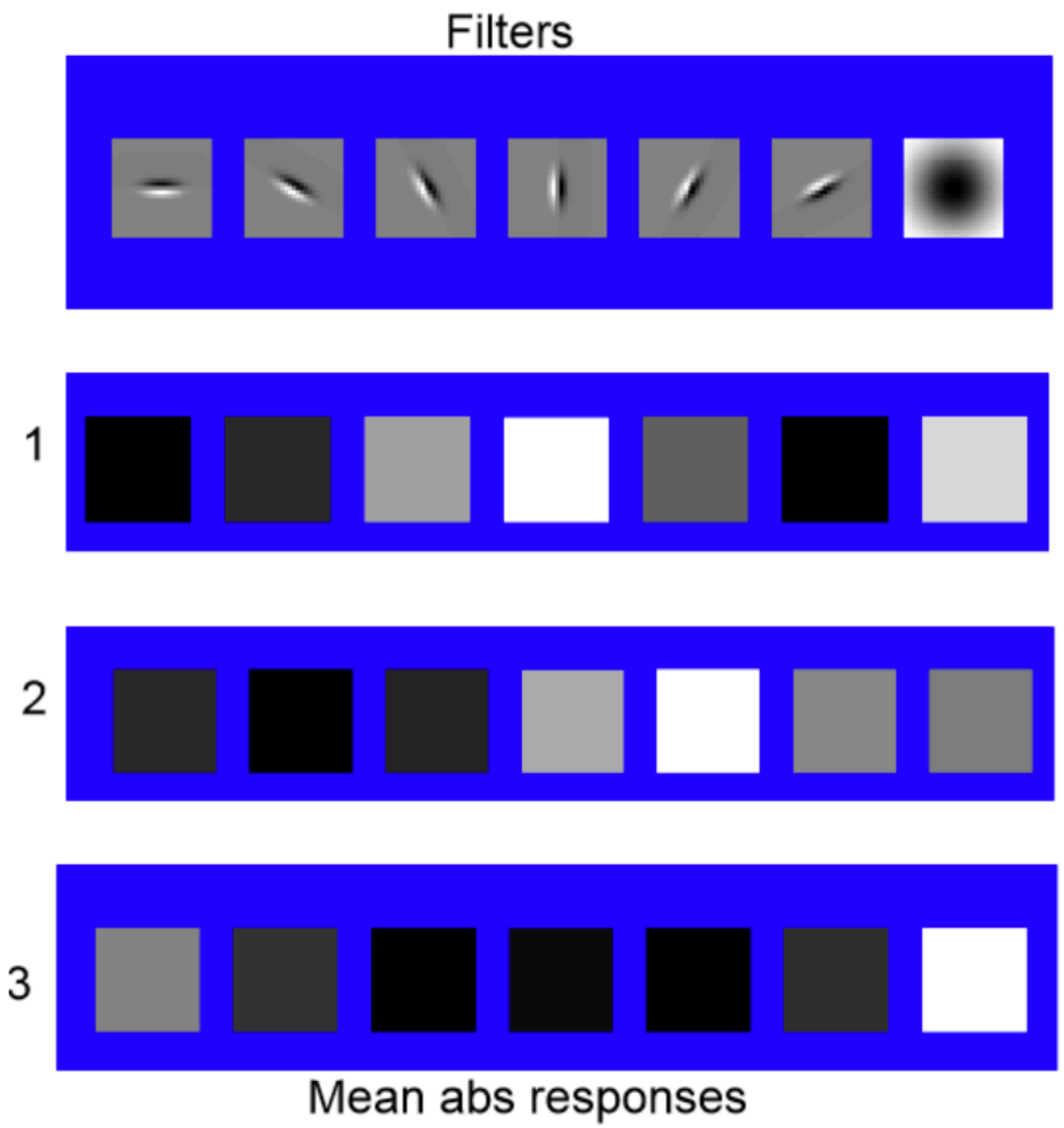
Forsyth & Ponce (1st ed.) Figures 9.3 and 9.5

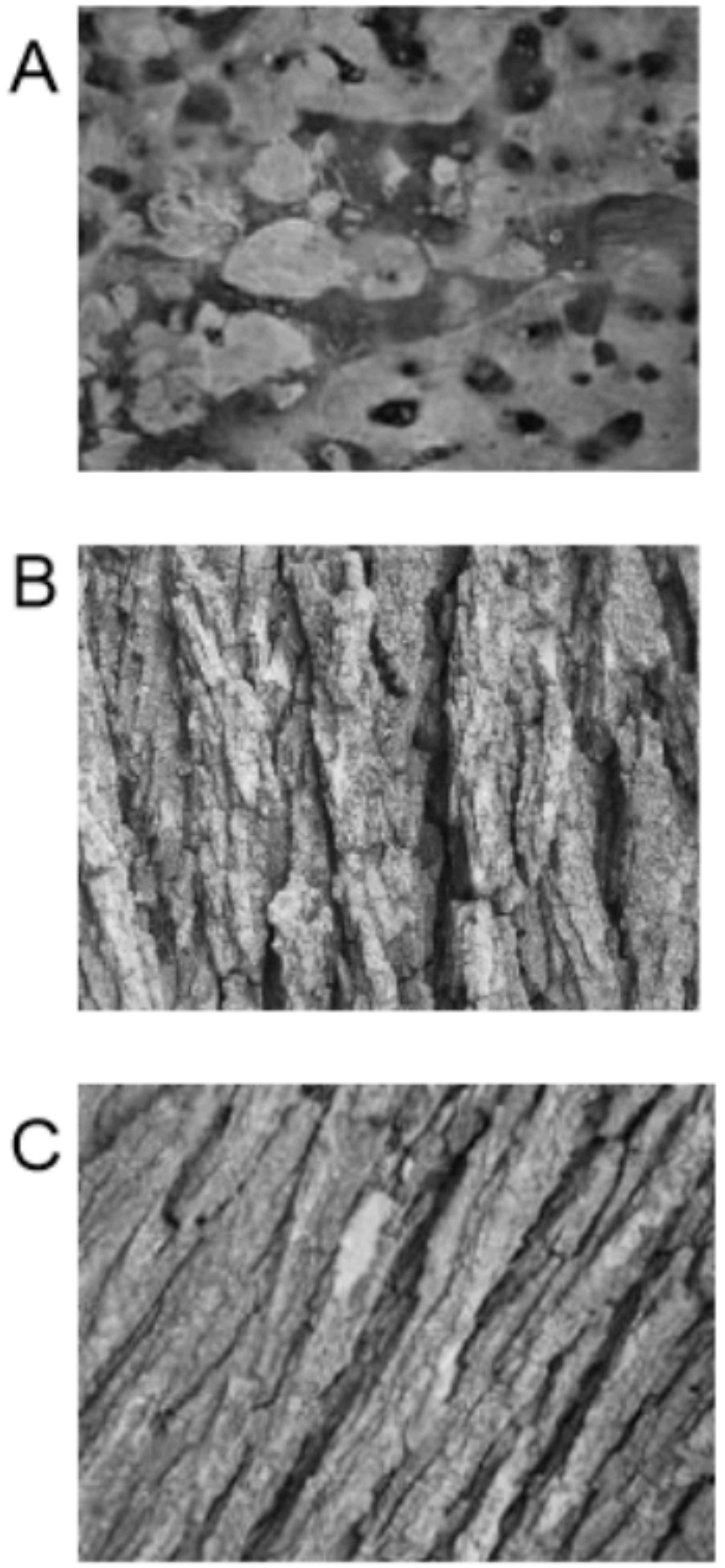
Comparison of Results



Forsyth & Ponce (1st ed.) Figures 9.4–9.5

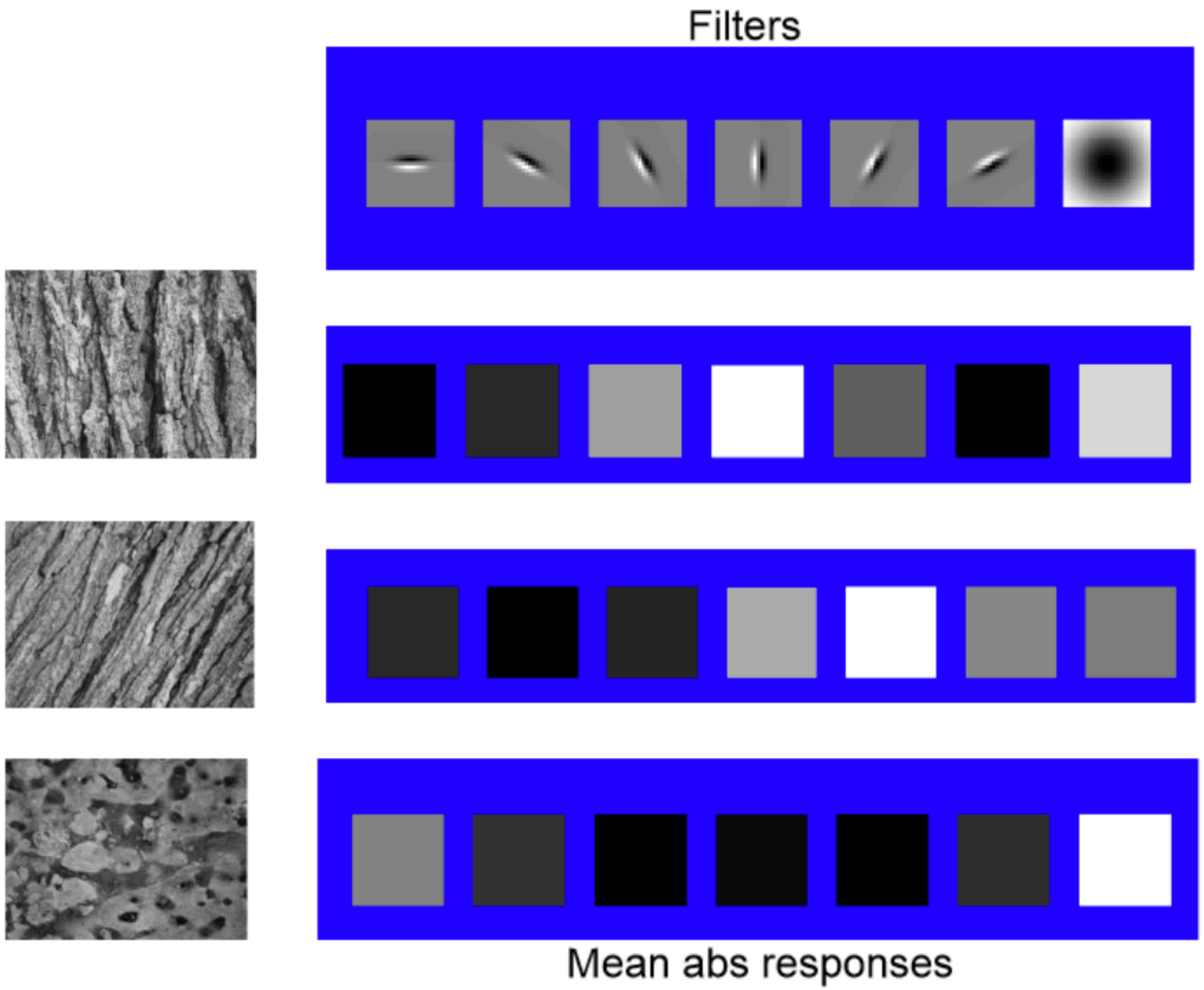
A Short **Exercise**: Match the texture to the response





Slide Credit: James Hays

A Short **Exercise**: Match the texture to the response



Slide Credit: James Hays

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