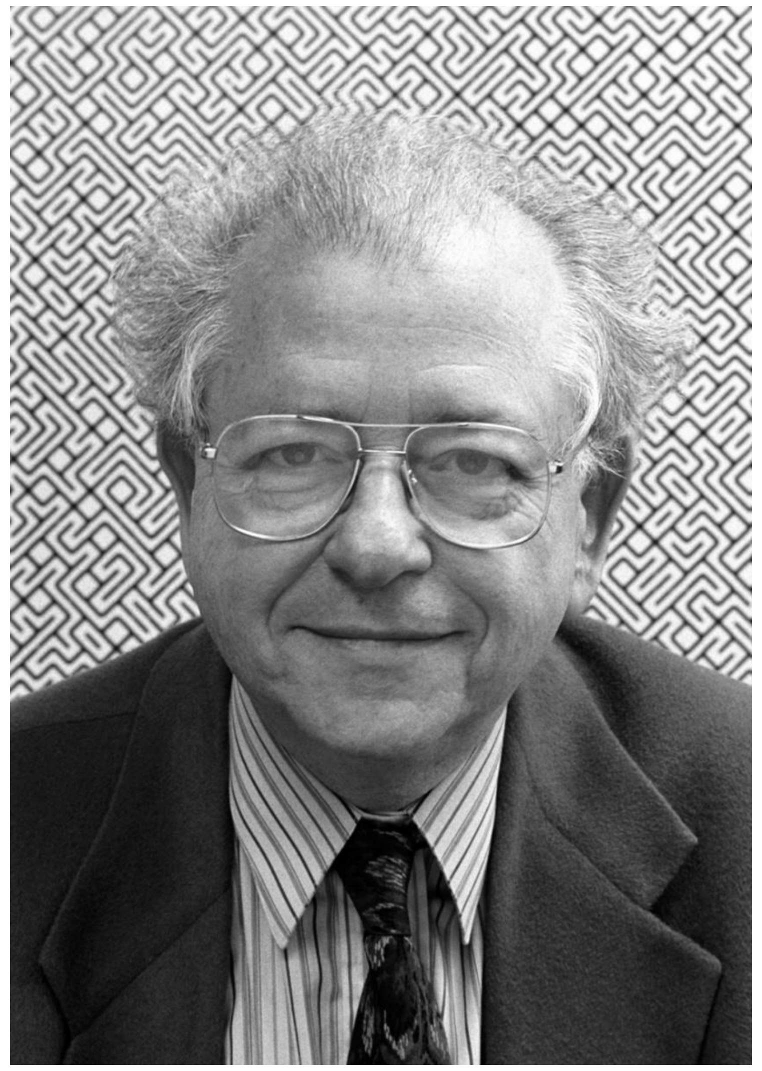
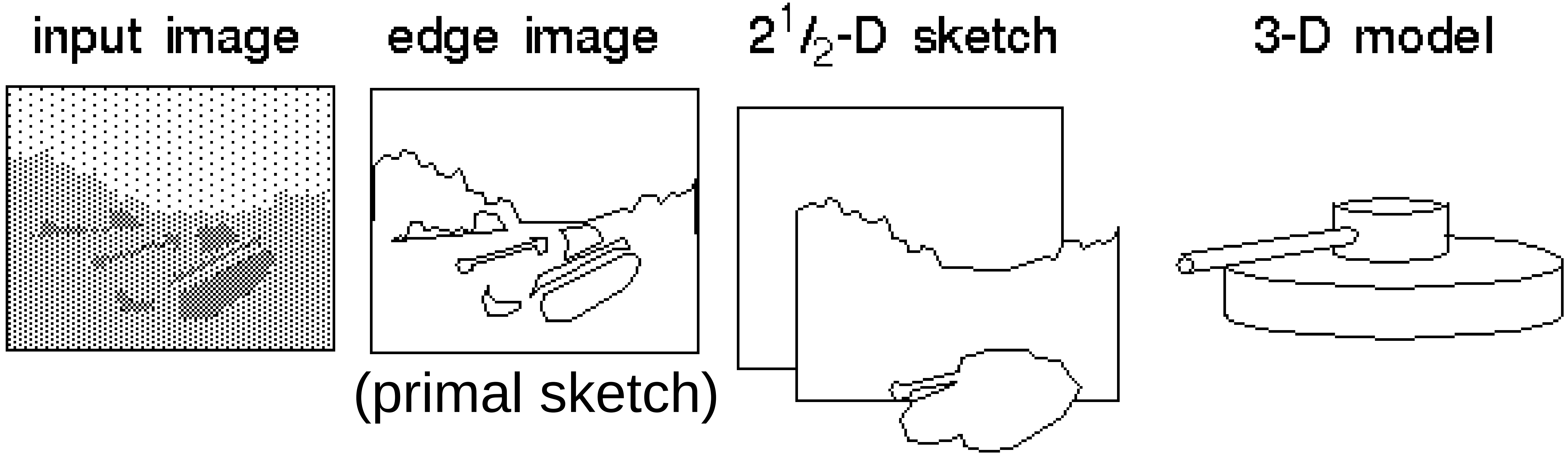
Statistical Images Representation (Everything is Texture!)



Bela Julesz

Alexei Efros, CS280, Spring 2024

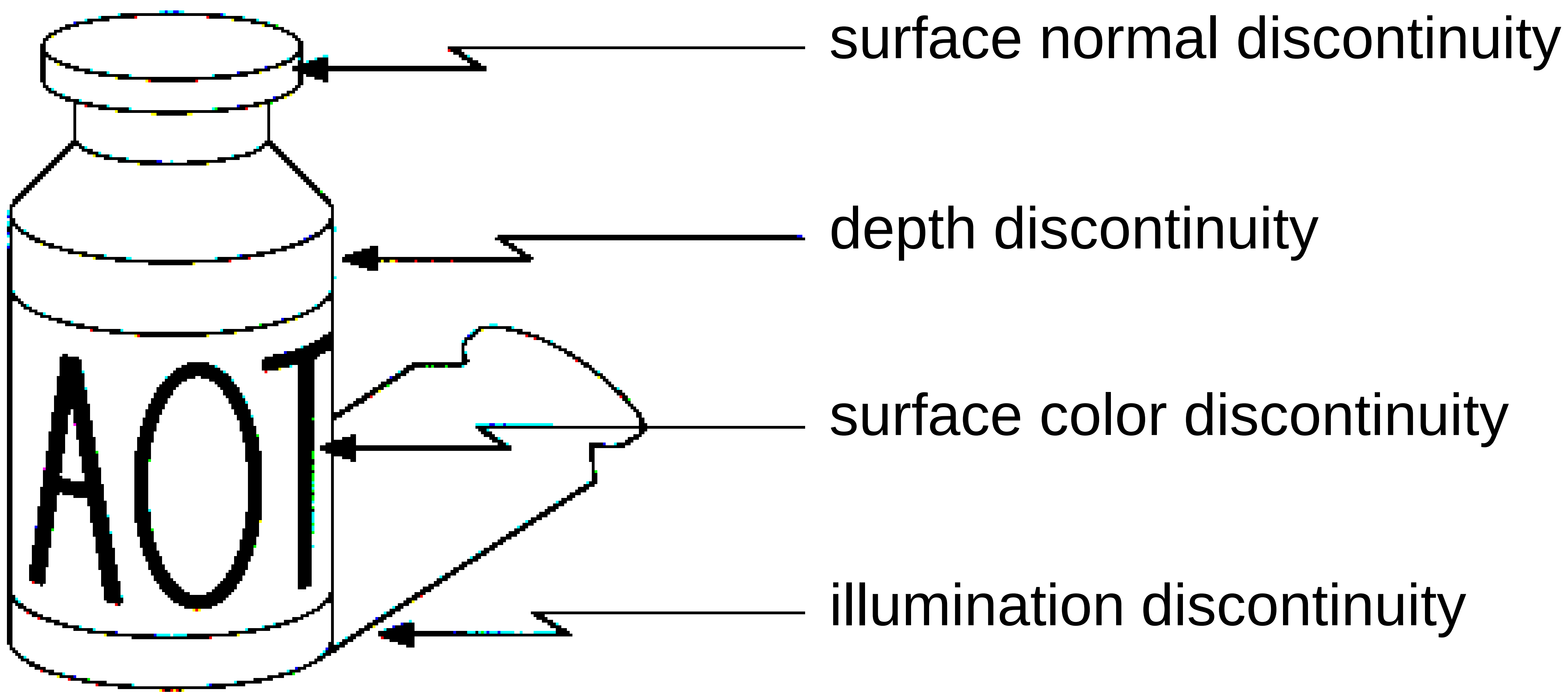
Things used to be so easy…



David Marr’s View of Vision (1980s)

Origin of edges

Edges are caused by a variety of factors:

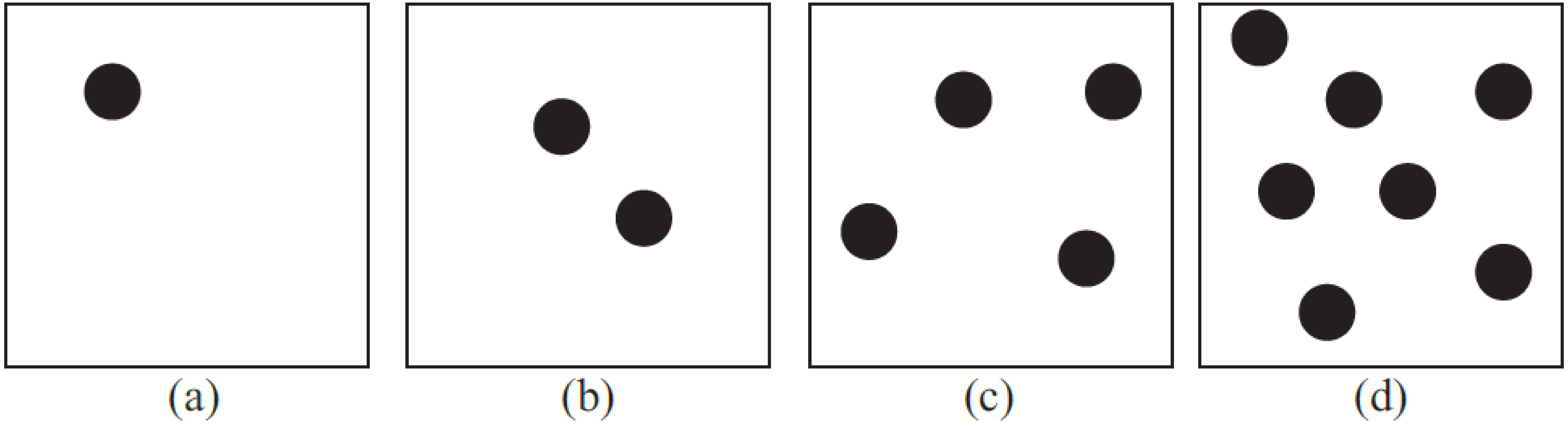


Source: Steve Seitz

Need a more statistical representation

Subitizing

Figure 28.2: When looking at these images, we can count the number of circles at a glance if there are less than five circles. When an image has more than five items, we have to count them one by one.



Crowding

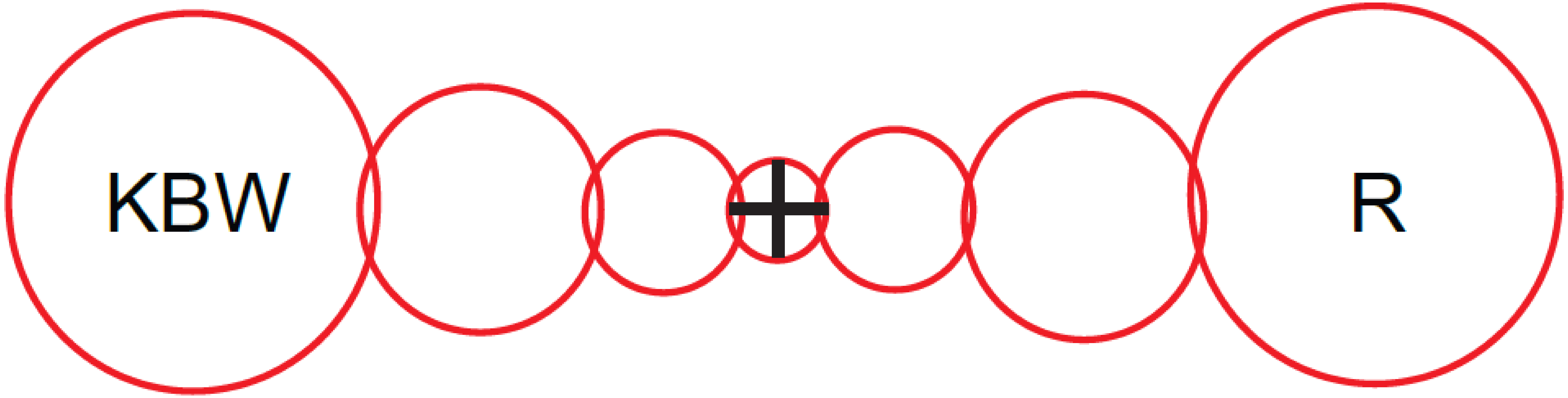


Figure 28.3: Crowding. If you look at the central cross, the letter R on the right can be recognized, however the letter B on the left is hard to read.

Texture as statistical image representation

- Narrowly: texture depicts spatially repeating patterns


radishes

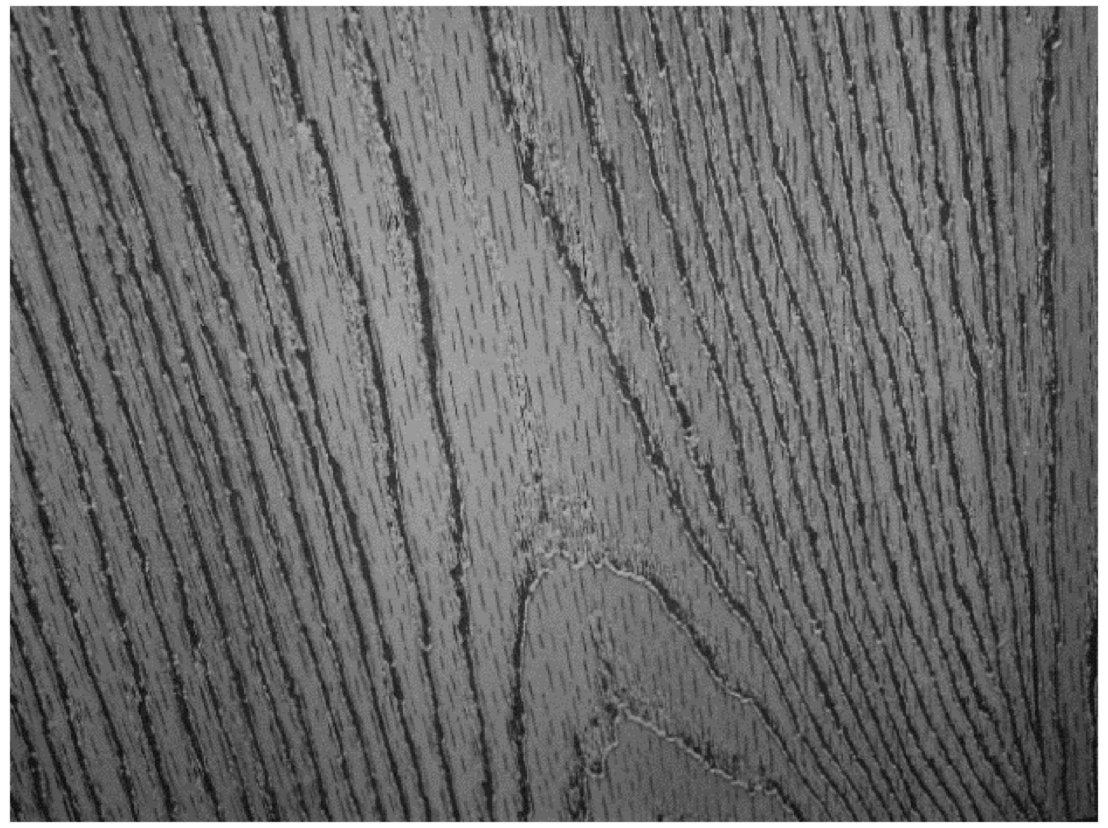

rocks

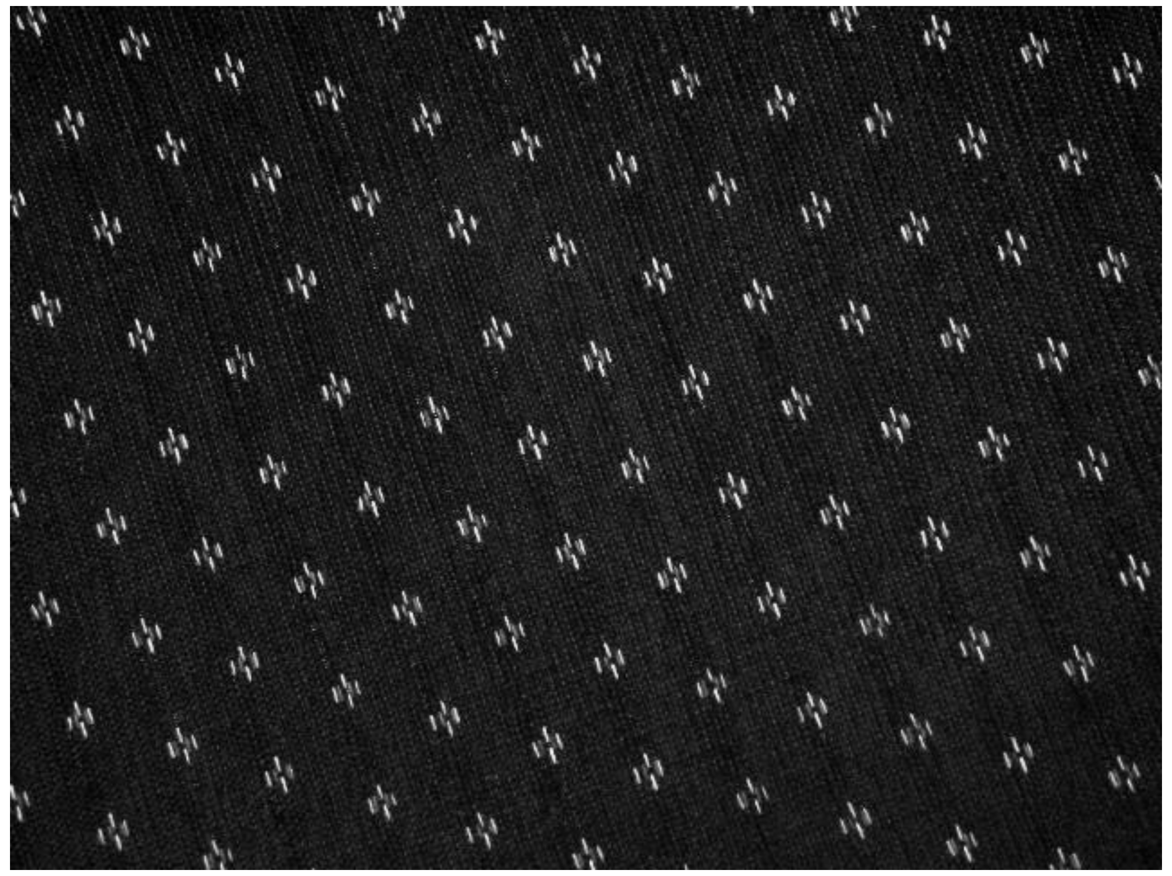

yogurt

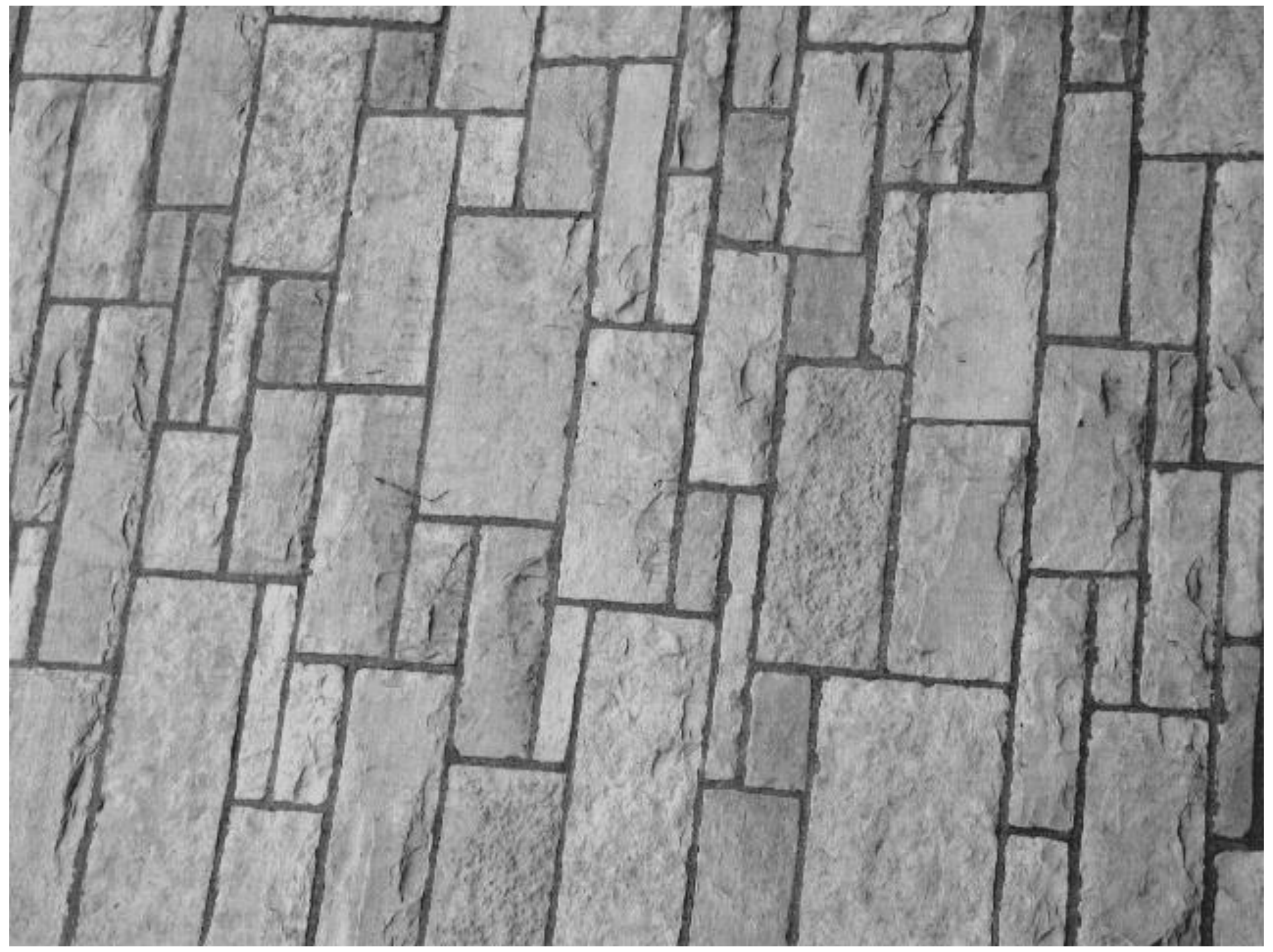
Texture as “stuff”

Source: Forsyth

Texture and Material







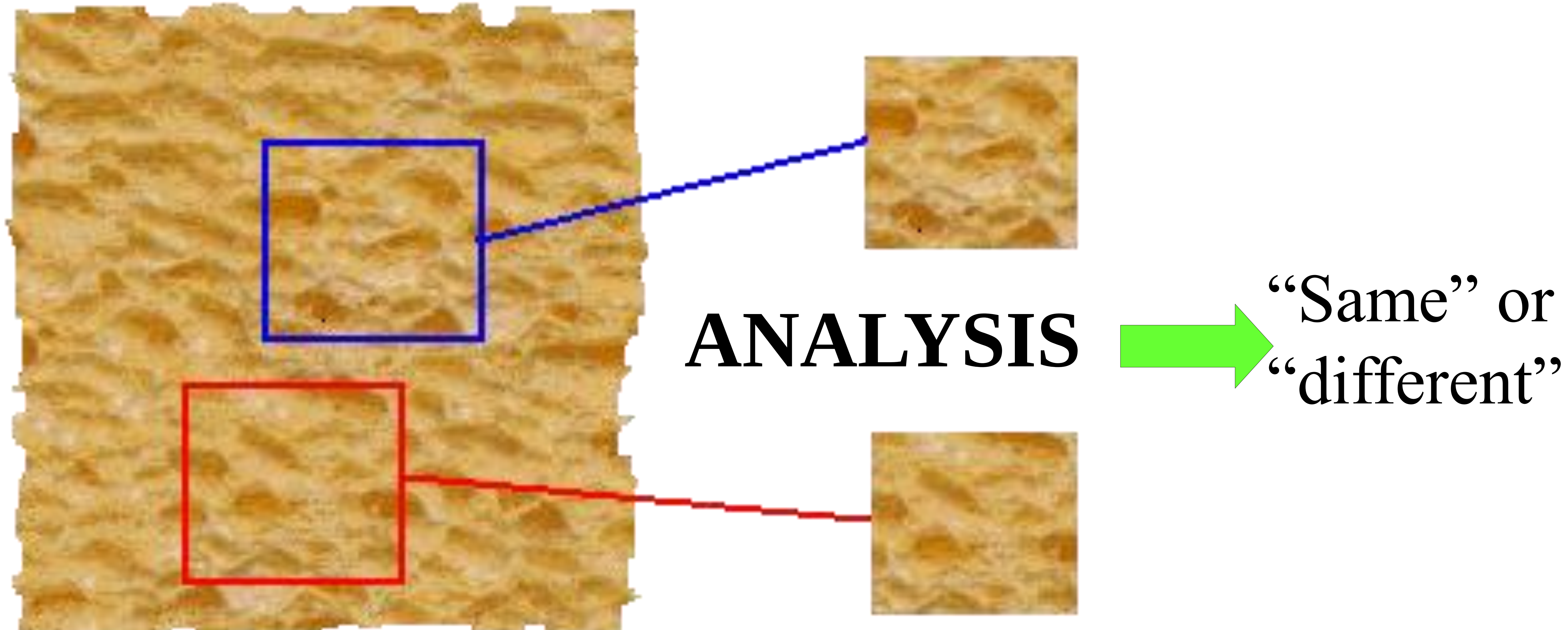
http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Statistical, not exact, similarity





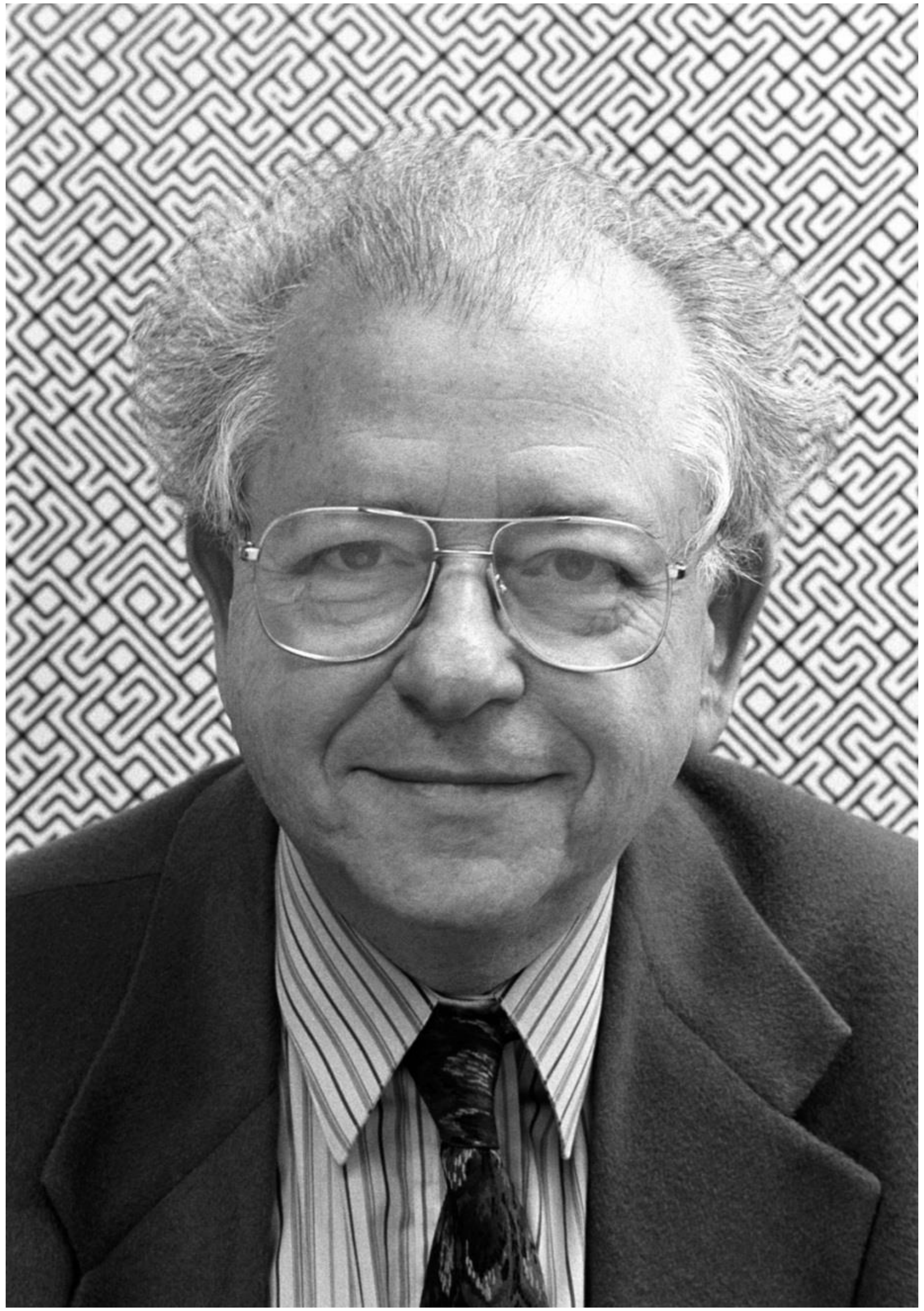
Texture Analysis



True (infinite) texture

Compare textures and decide if they’re made of the same “stuff”.

Béla Julesz, father of texture

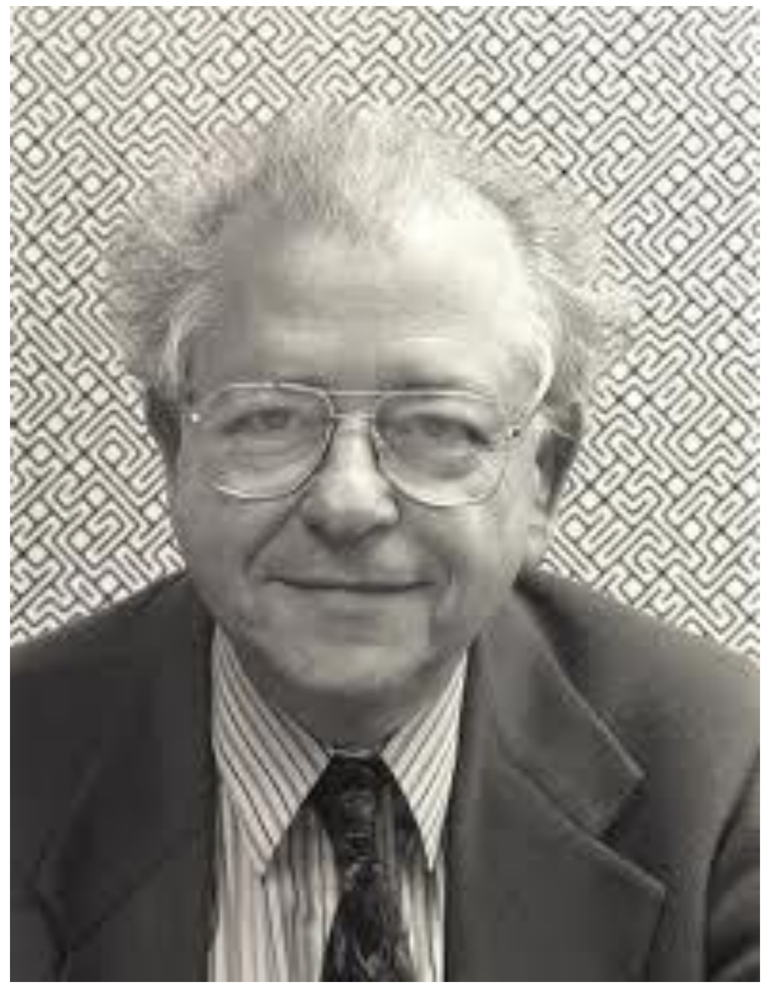


REVIEW ARTICLES

Textons, the elements of texture perception, and their interactions

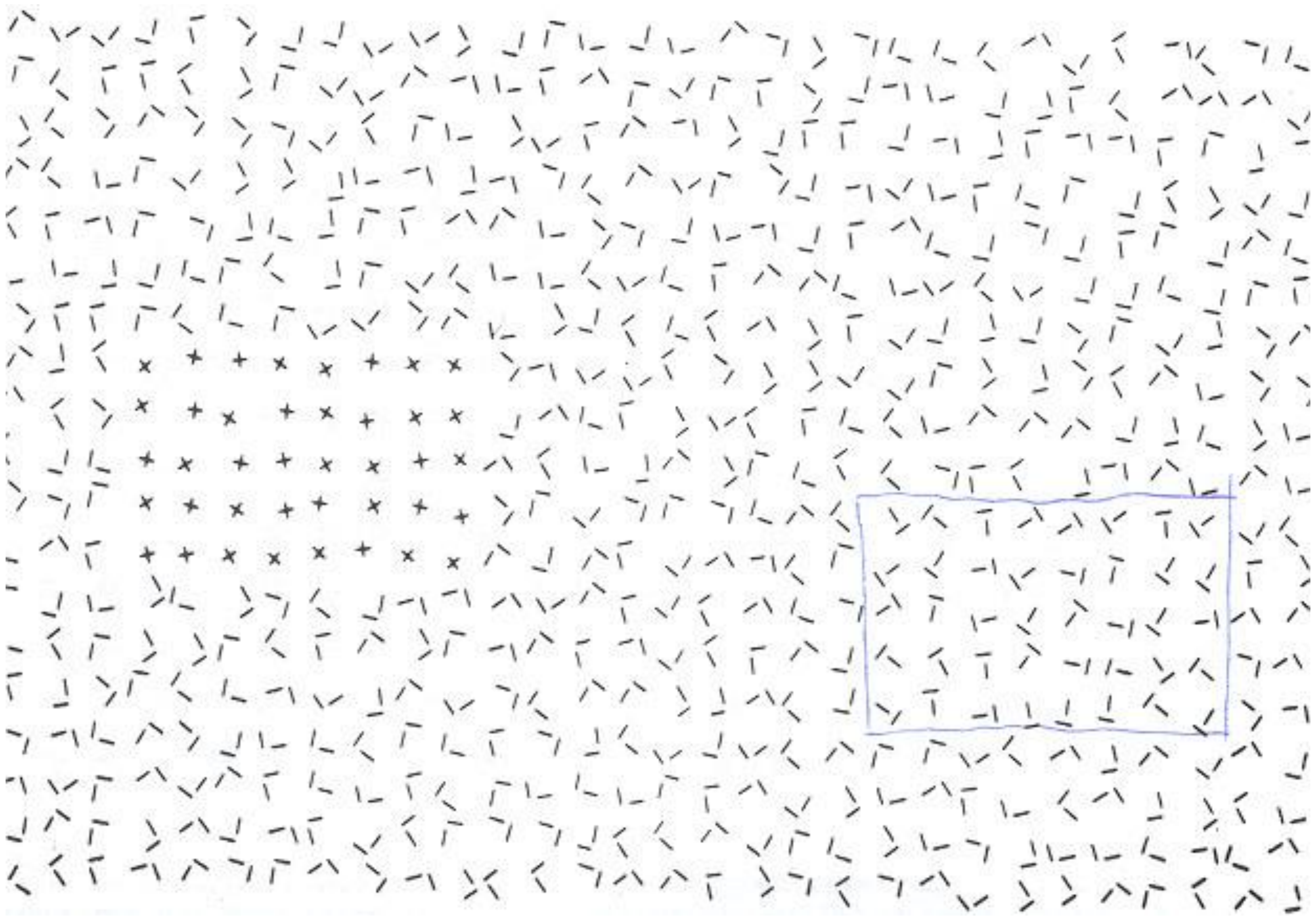
Bela Julesz
Bell Laboratories, Murray Hill, New Jersey 07974, USA

Research with texture pairs having identical second-order statistics has revealed that the pre-attentive texture discrimination system cannot globally process third- and higher-order statistics, and that discrimination is the result of a few local conspicuous features, called textons. It seems that only the first-order statistics of these textons have perceptual significance, and the relative phase between textons cannot be perceived without detailed scrutiny by focal attention.



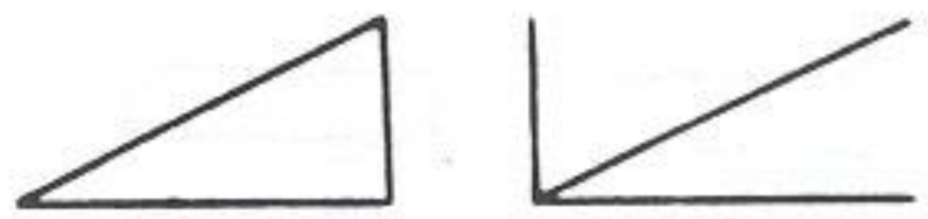
Bela Julesz, "Textons, the Elements of Texture Perception, and their Interactions". Nature 290: 91-97. March, 1981.

Texton Discrimination (Julesz)

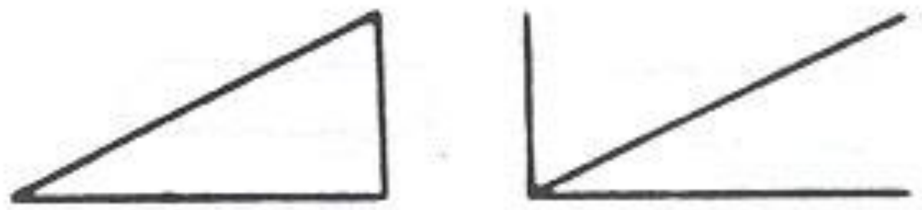


Human vision is sensitive to the difference of some types of elements and appears to be “numb” on other types of differences.

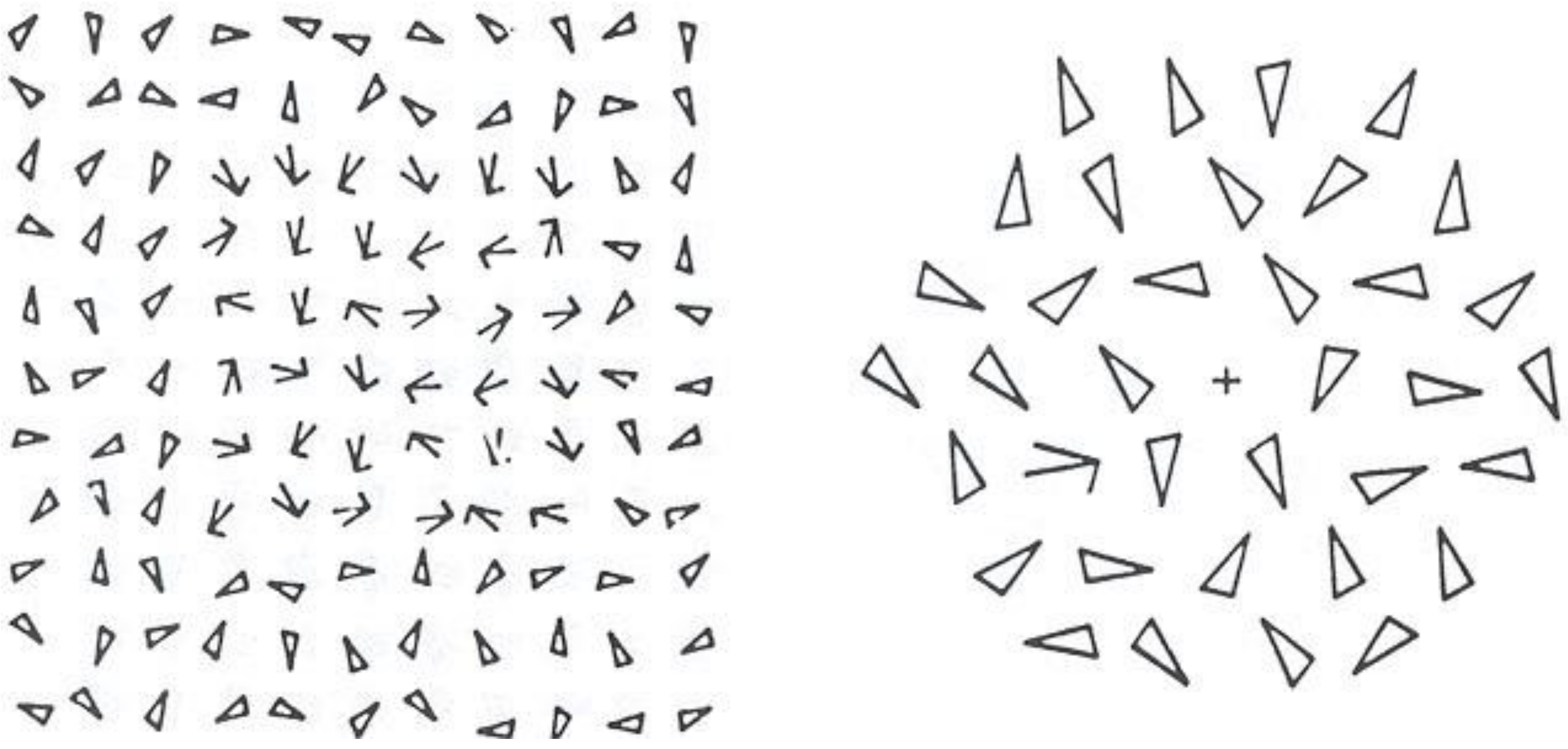
Search Experiment I



Search Experiment I

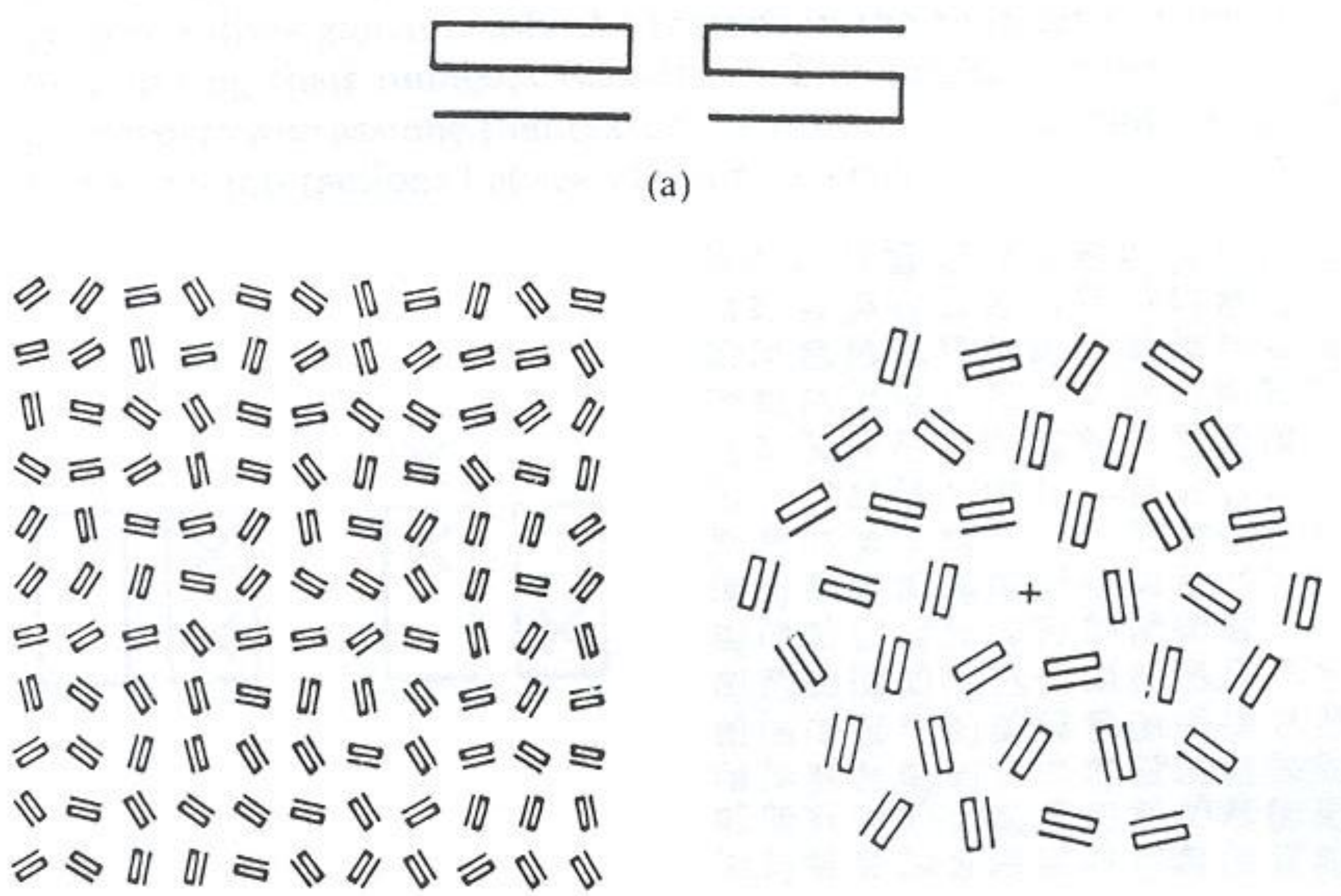


(a)



The subject is told to detect a target element in a number of background elements.
In this example, the detection time is independent of the number of background elements.

Search Experiment II



In this example, the detection time is proportional to the number of background elements,
And thus suggests that the subject is doing element-by-element scrutiny.

Pre-attentive vs. Attentive Vision

Julesz then conjectured the following axiom:

Human vision operates in two distinct modes:

1. Preattentive vision

   parallel, instantaneous *(~100--200ms)*, without scrutiny,
   independent of the number of patterns, covering a large visual field.

2. Attentive vision

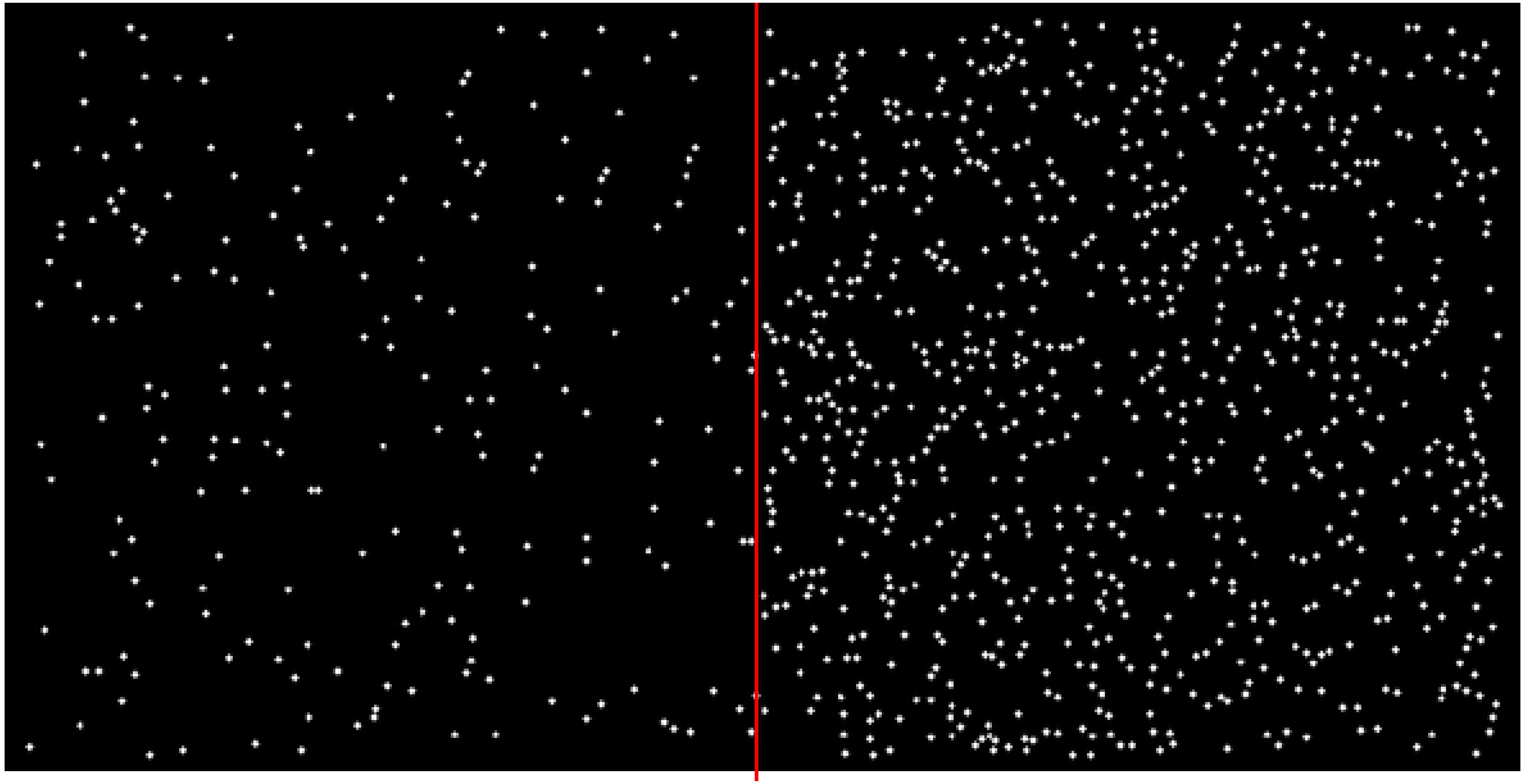
   serial search by focal attention in 50ms steps limited to small aperture.

See also System 1 vs. System 2

Julesz Conjecture

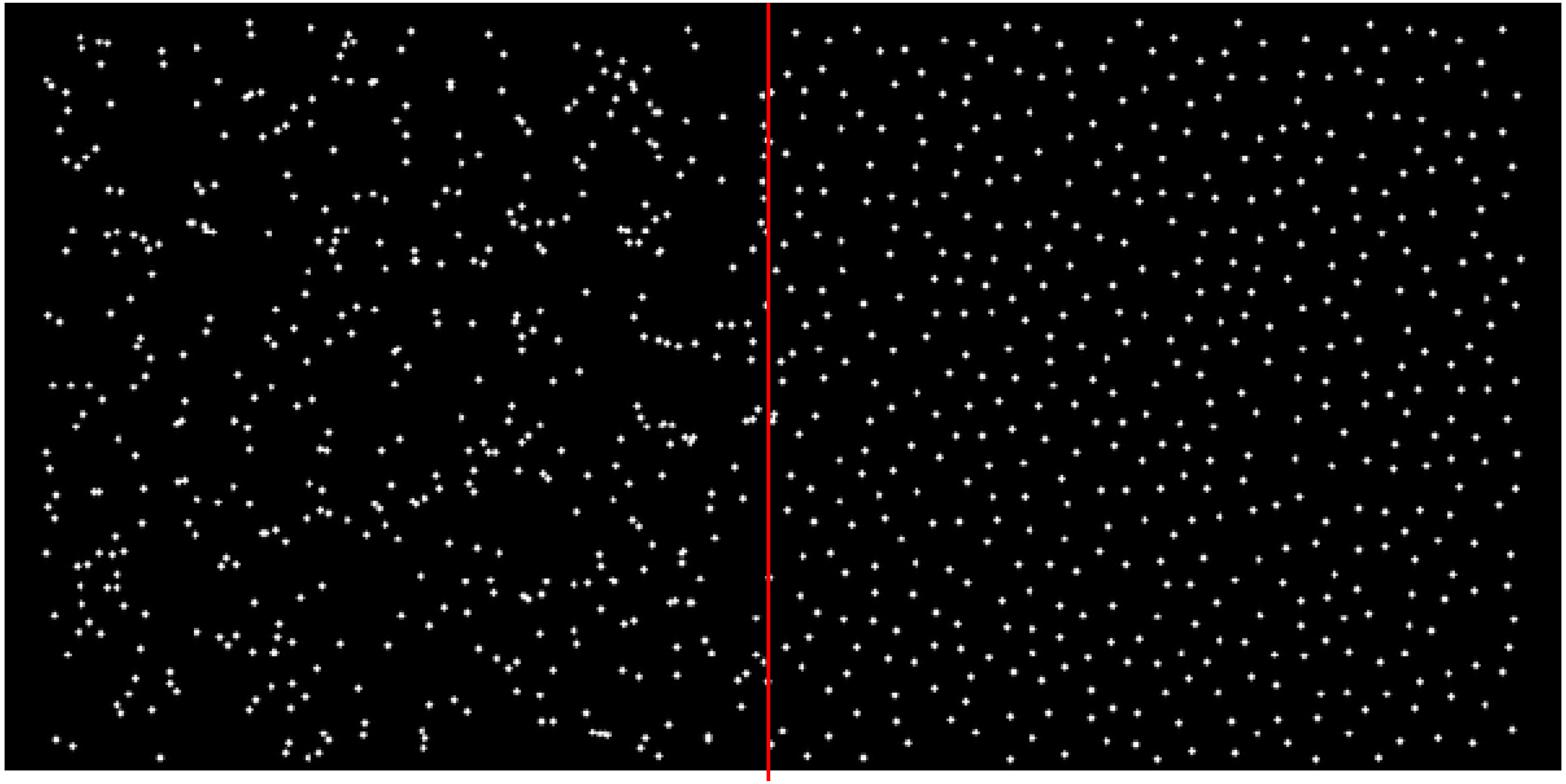
Textures cannot be spontaneously discriminated if they have the ***same first-order and second-order statistics*** *of texture features (textons) and differ only in their third-order or higher-order statistics.*

1st Order Statistics



5% white 20% white

2^{nd} Order Statistics



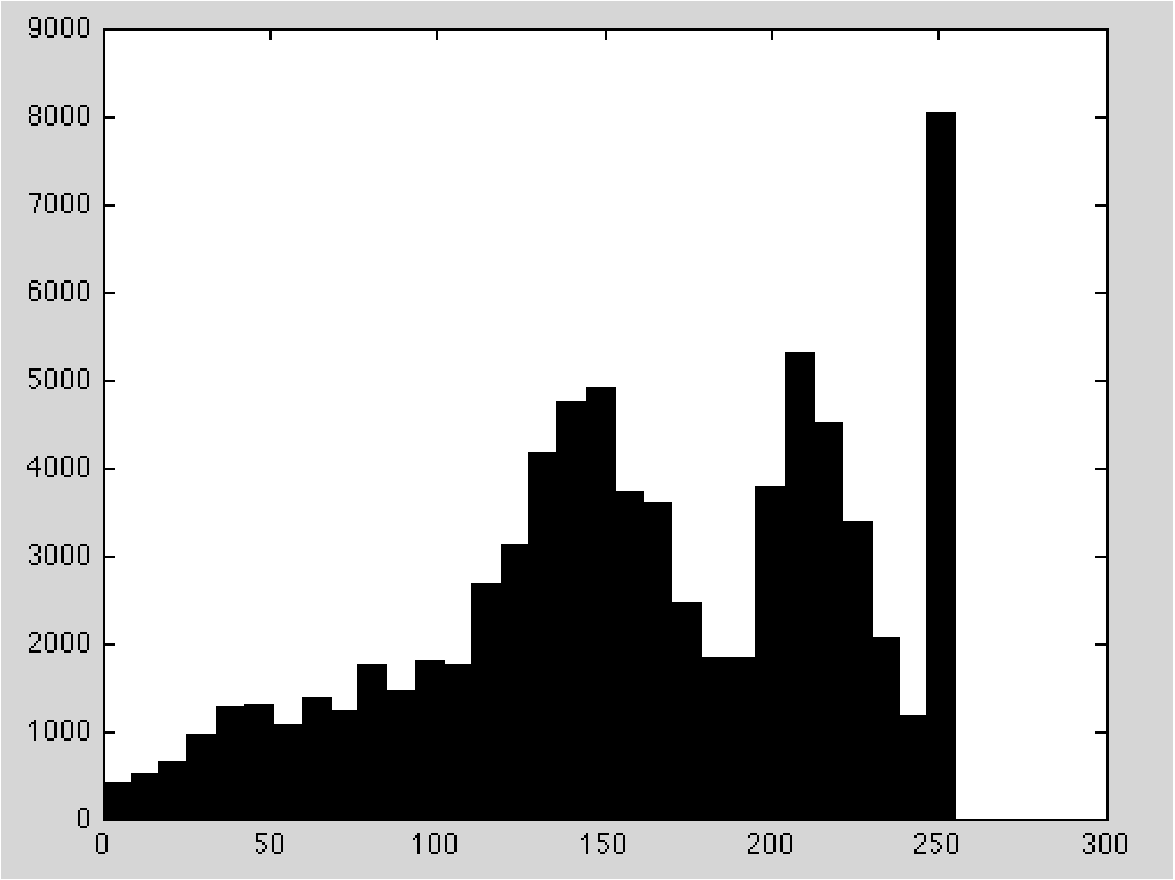
10% white

Two questions of texture modeling

- What are the texture features (textons)?
  - Pixels
  - Pixel patches
  - Outputs of V1-like filters
  - Clusters of patches / filter outputs
  - CNN features
  - Etc.

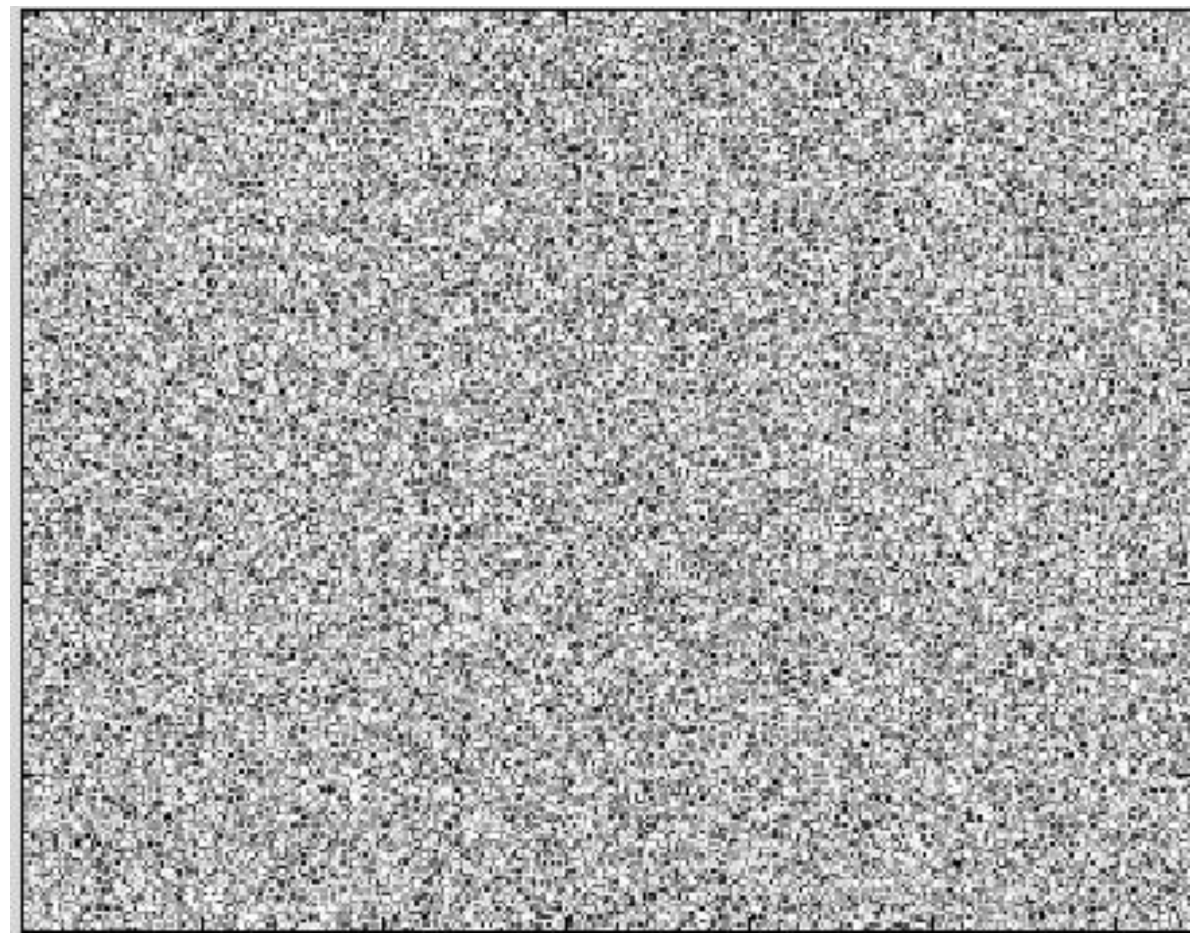
- How do we aggregate statistics
  - Various types of histograms
  - Implicit or explicit

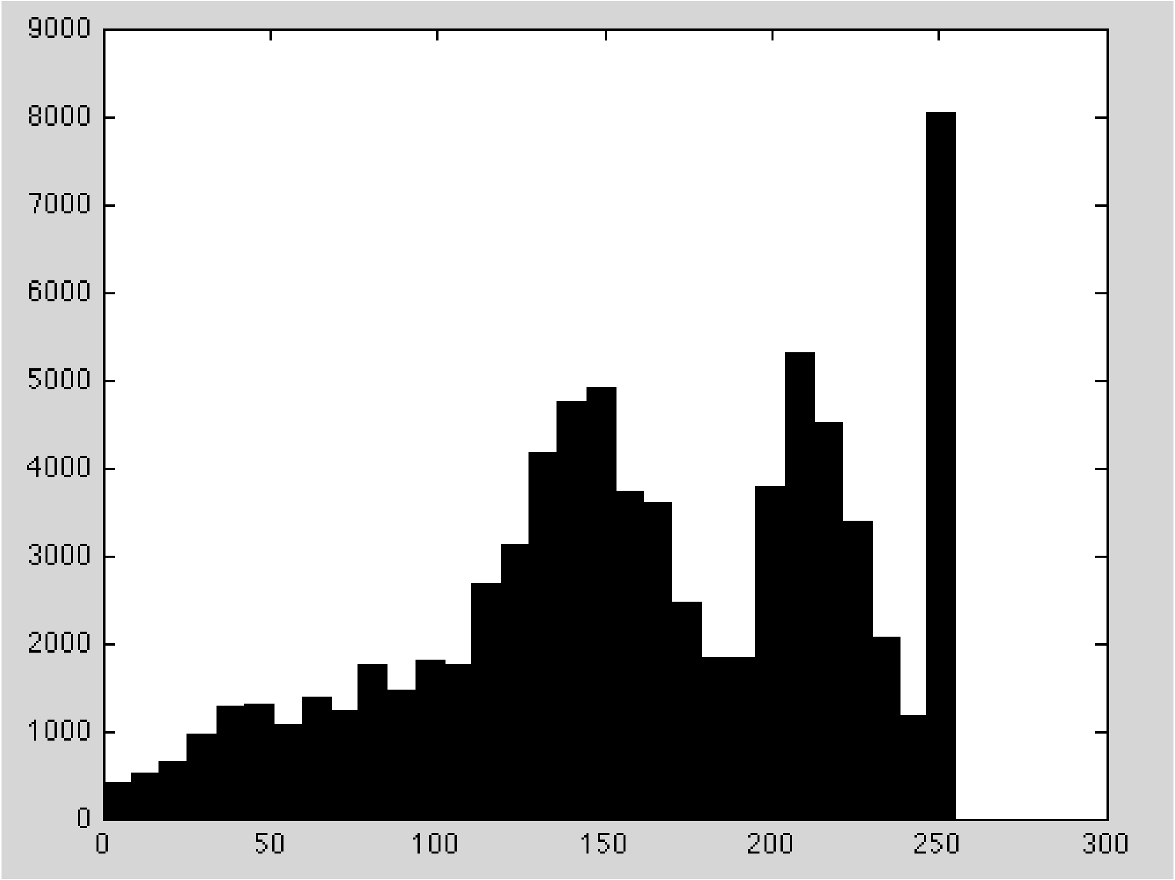
Pixel Histograms

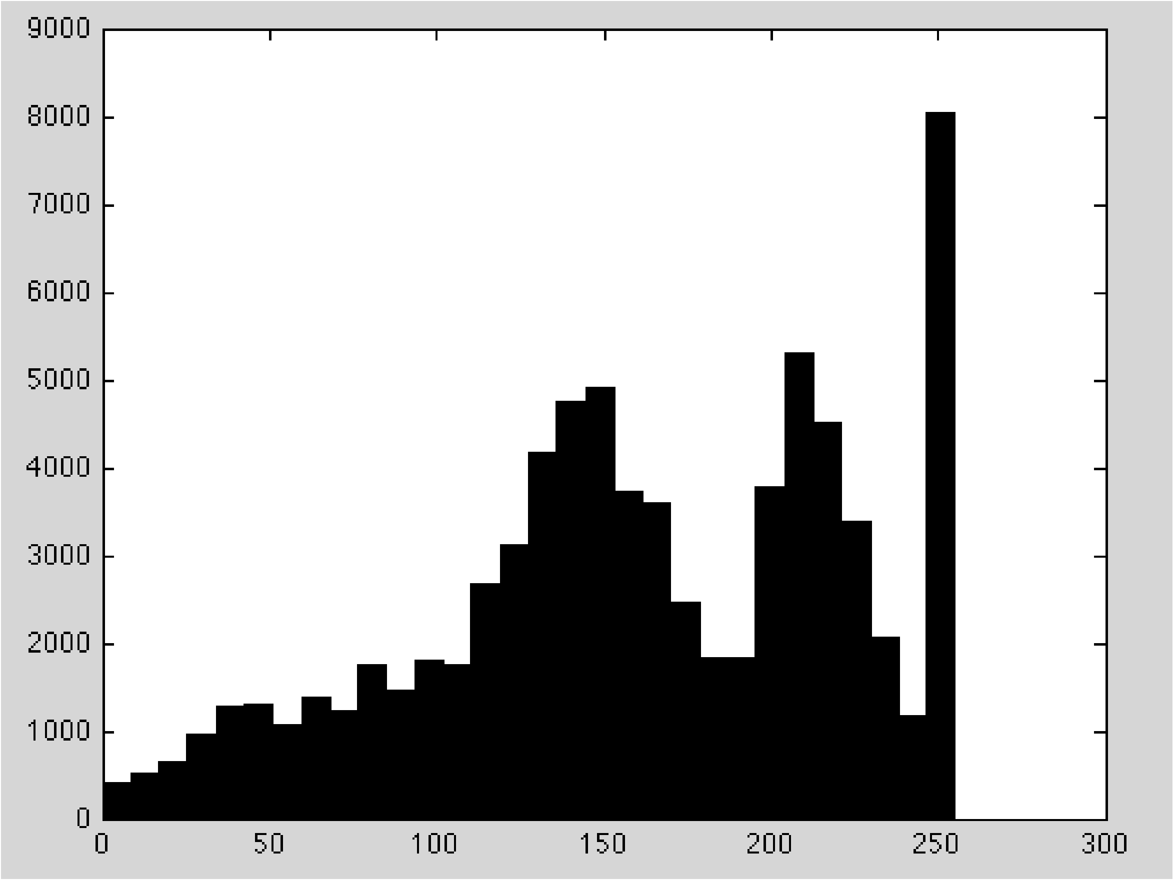


Slide by Erik Learned-Miller

Pixel Histograms

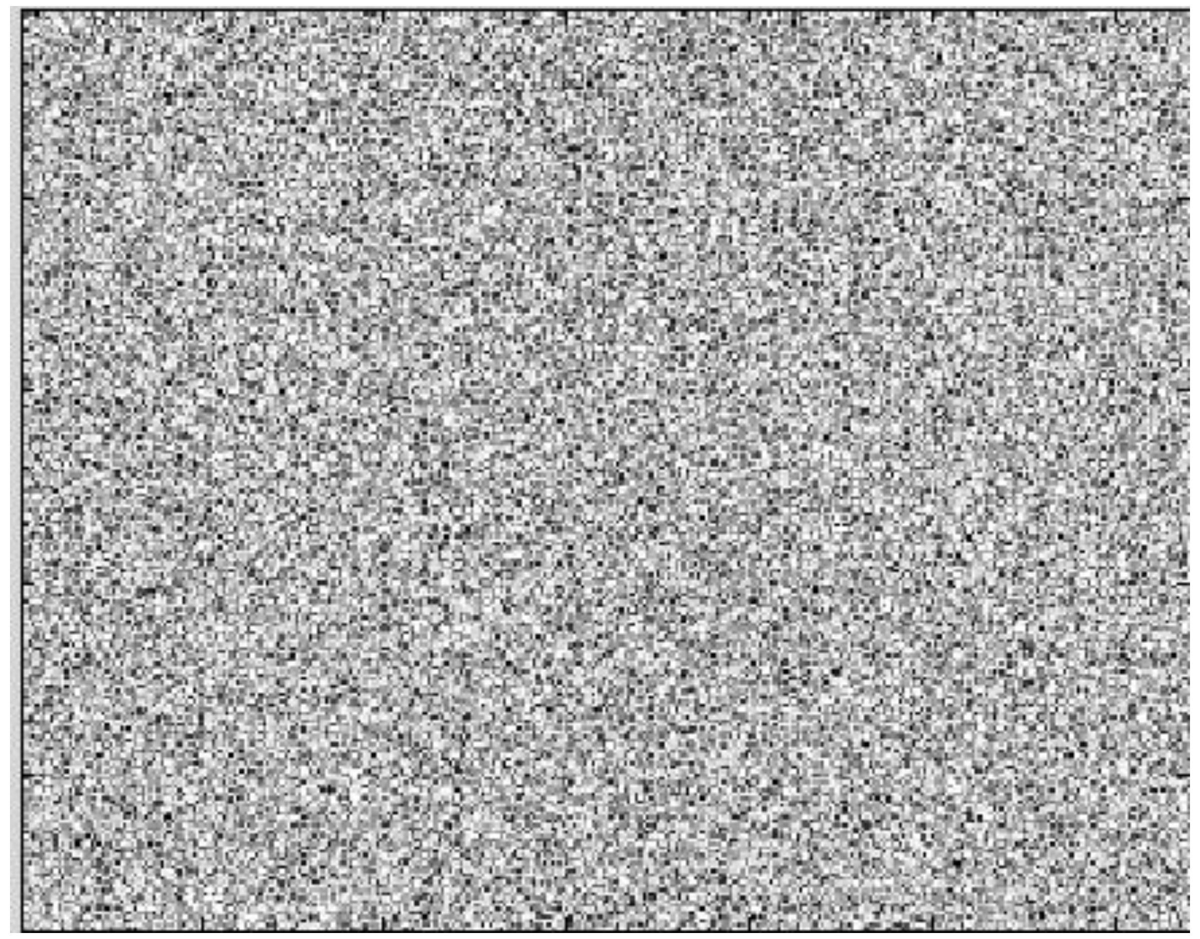






Slide by Erik Learned-Miller

Gray value histogram comparisons

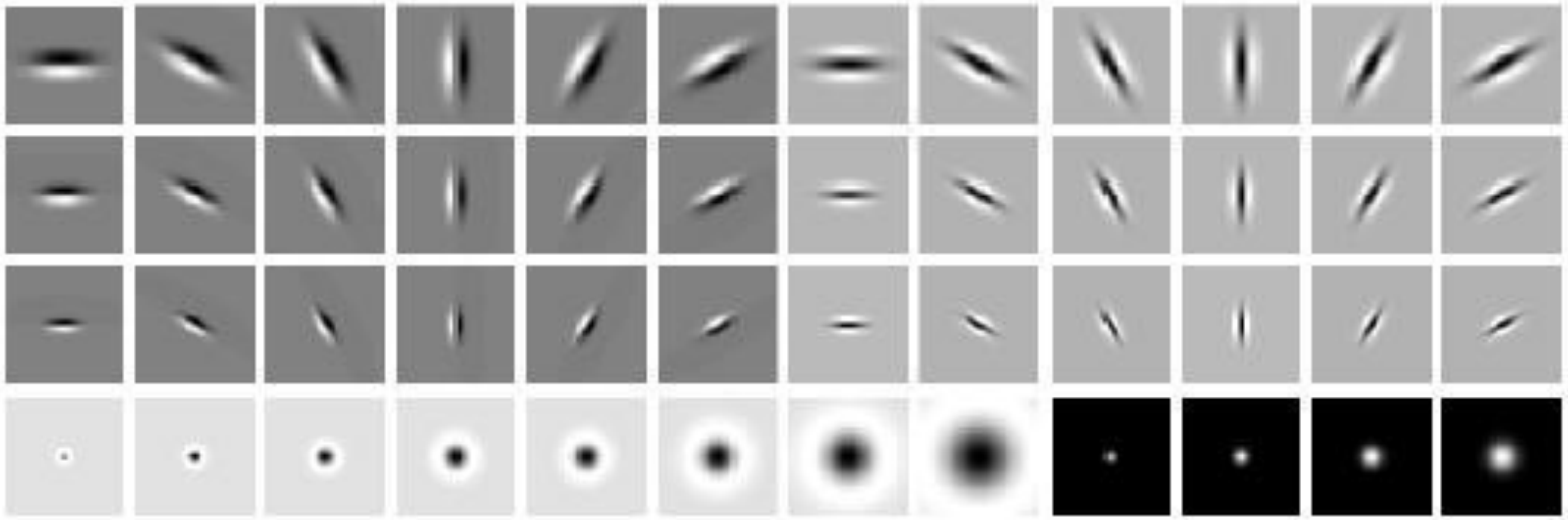


They’re equal

Slide by Erik Learned-Miller

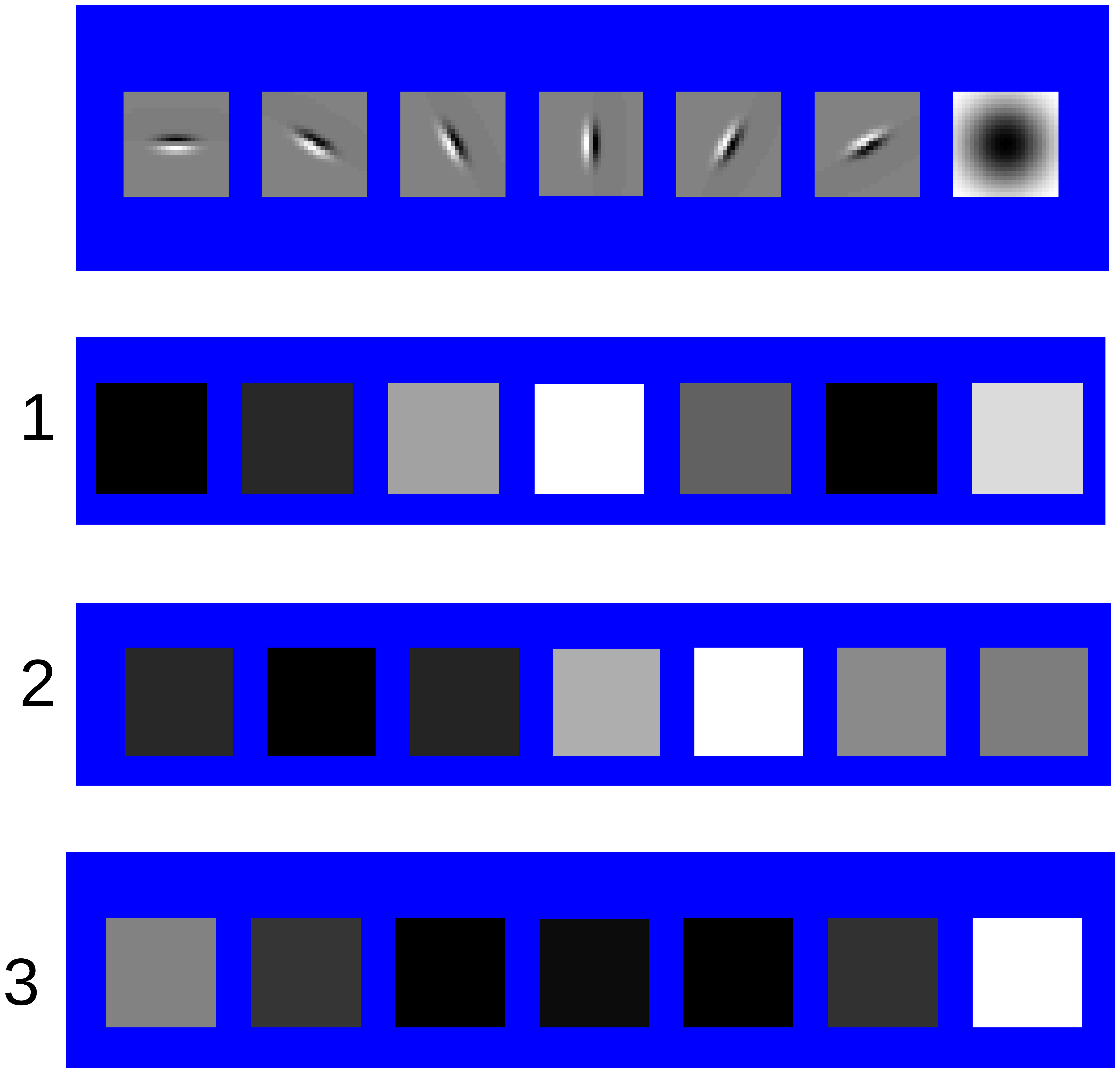
Going up from pixels: V1 filter-banks

LM Filter Bank

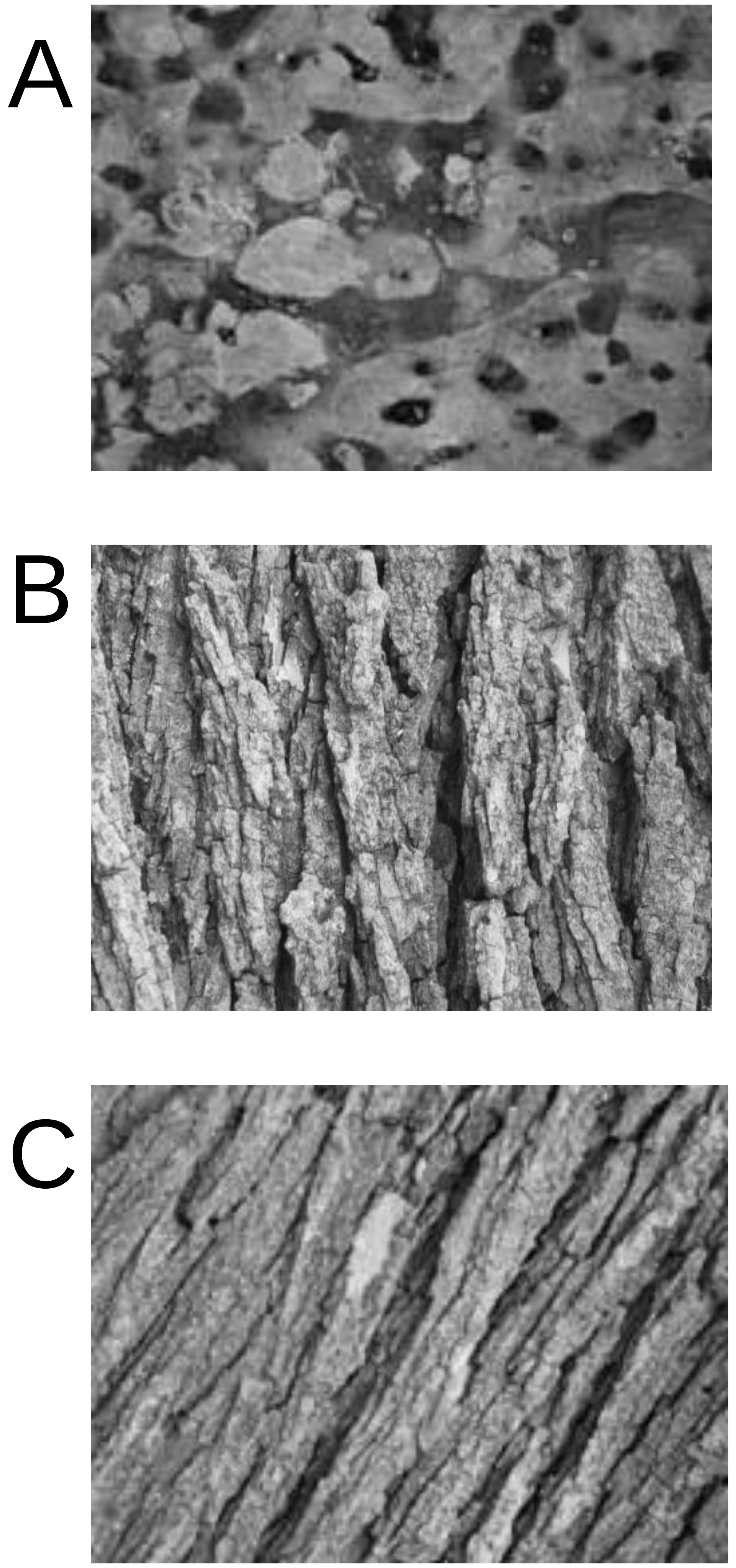


Can you match the texture to the response?

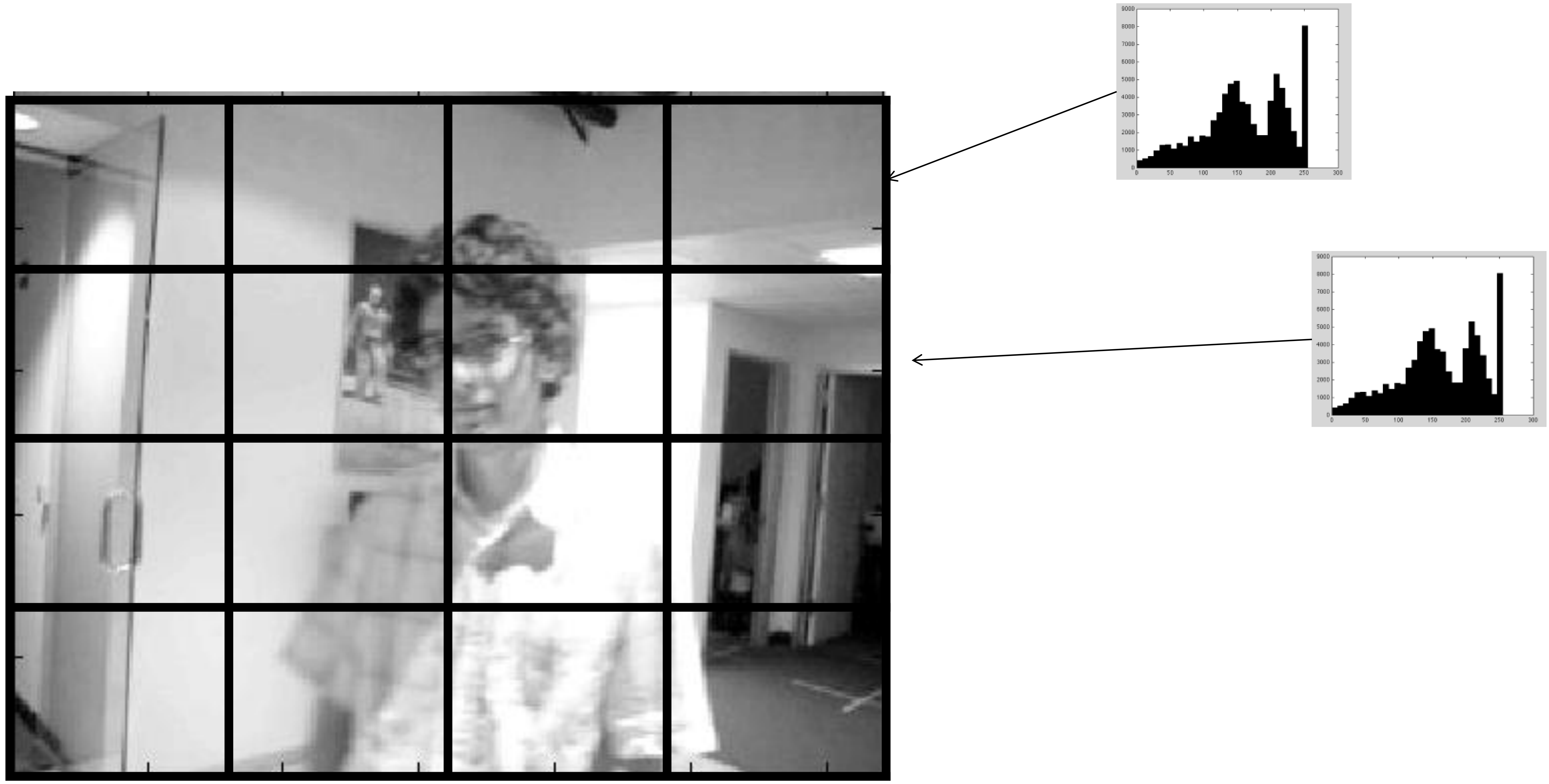
Filters



Mean abs responses



Adding spatial structure

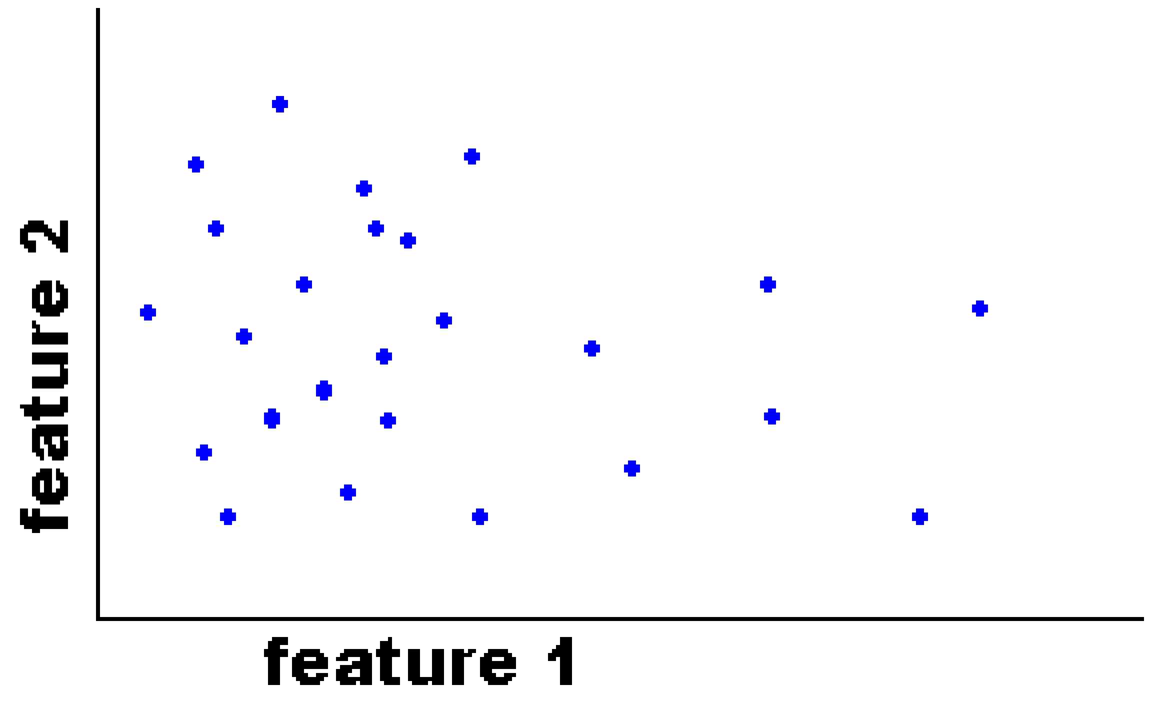


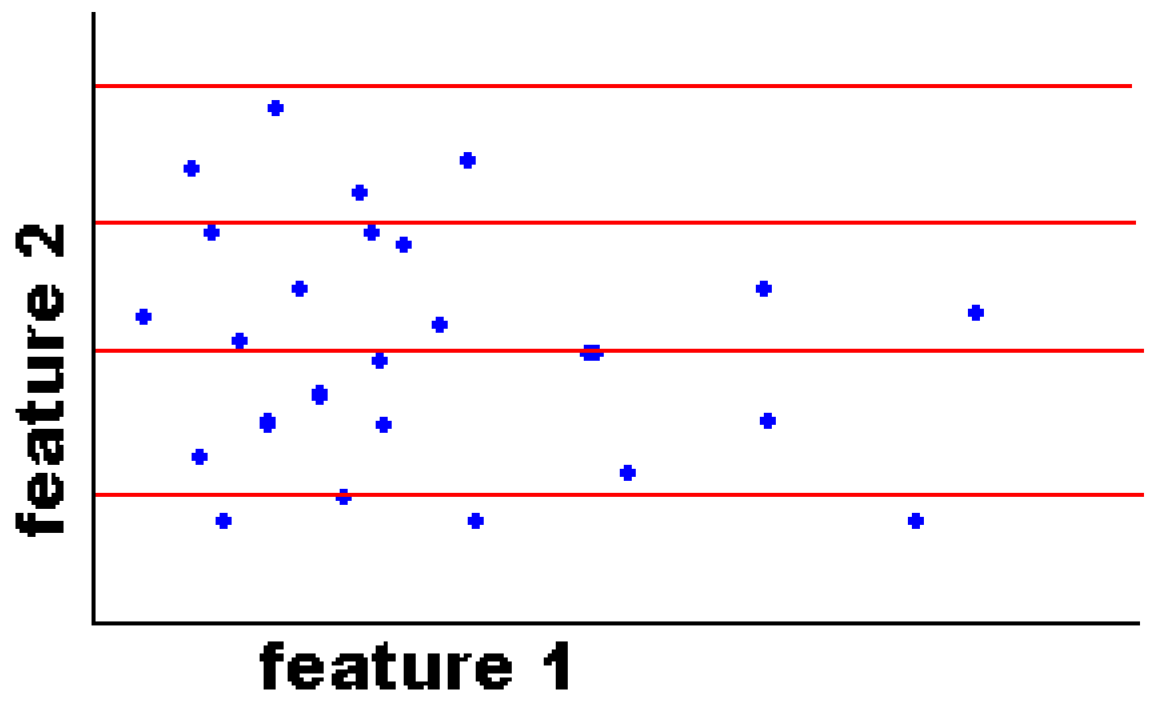
A separate histogram for each region.

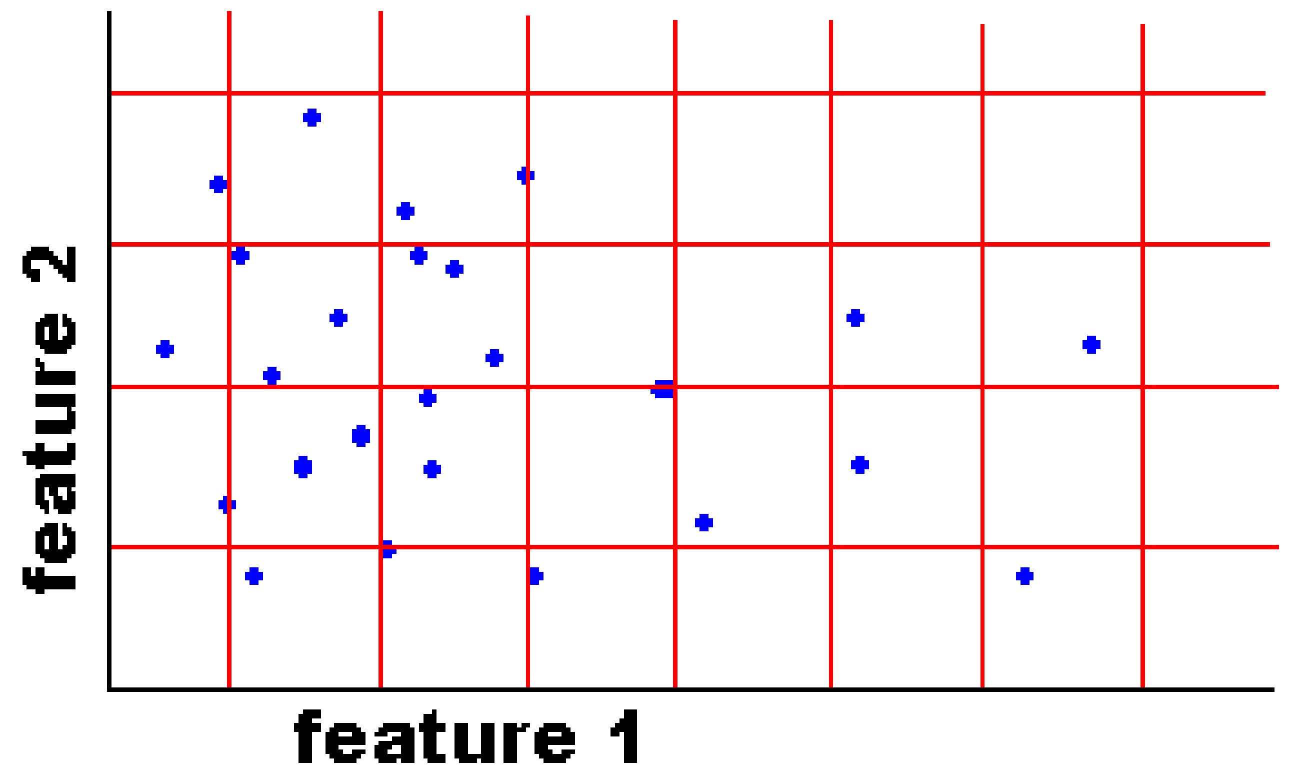
Slide by Erik Learned-Miller

Image Representations: Histograms

Images from Dave Kauchak

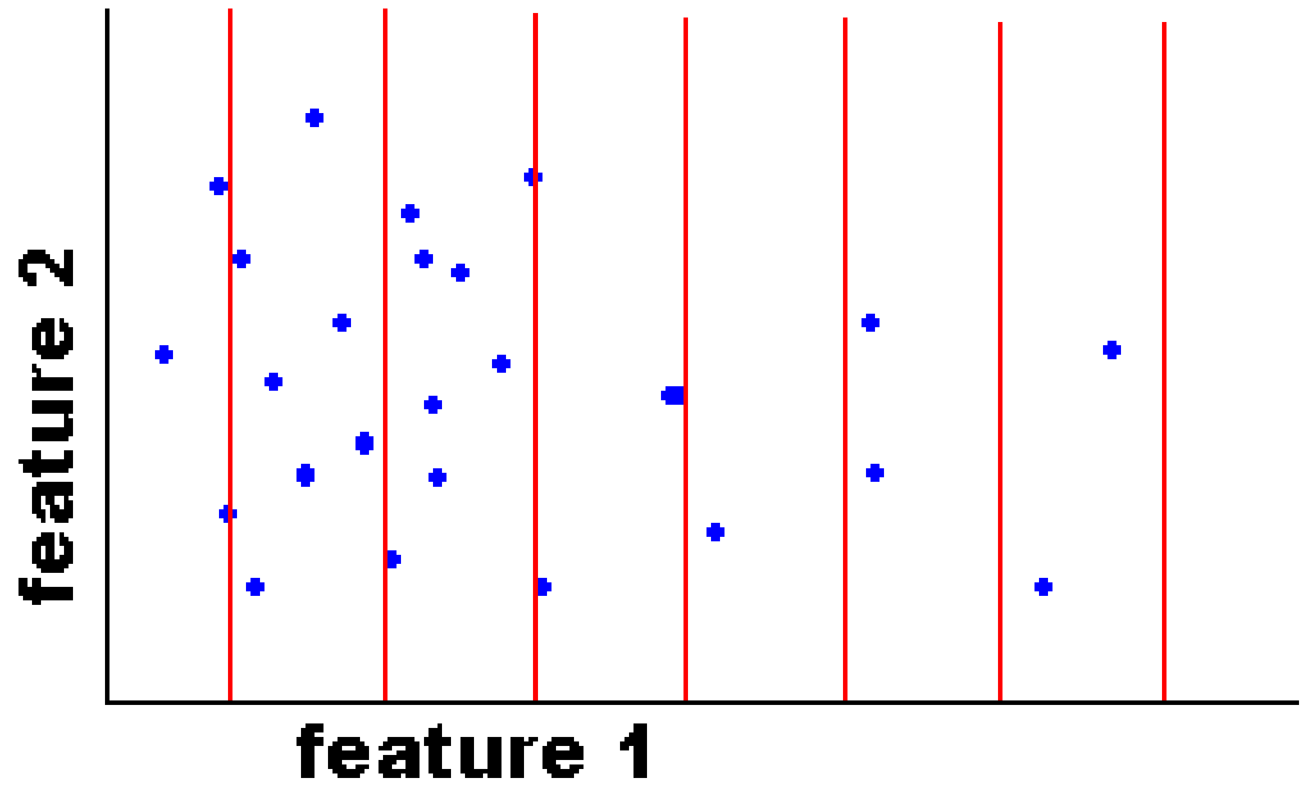






Joint histogram

- Requires lots of data
- Loss of resolution to avoid empty bins

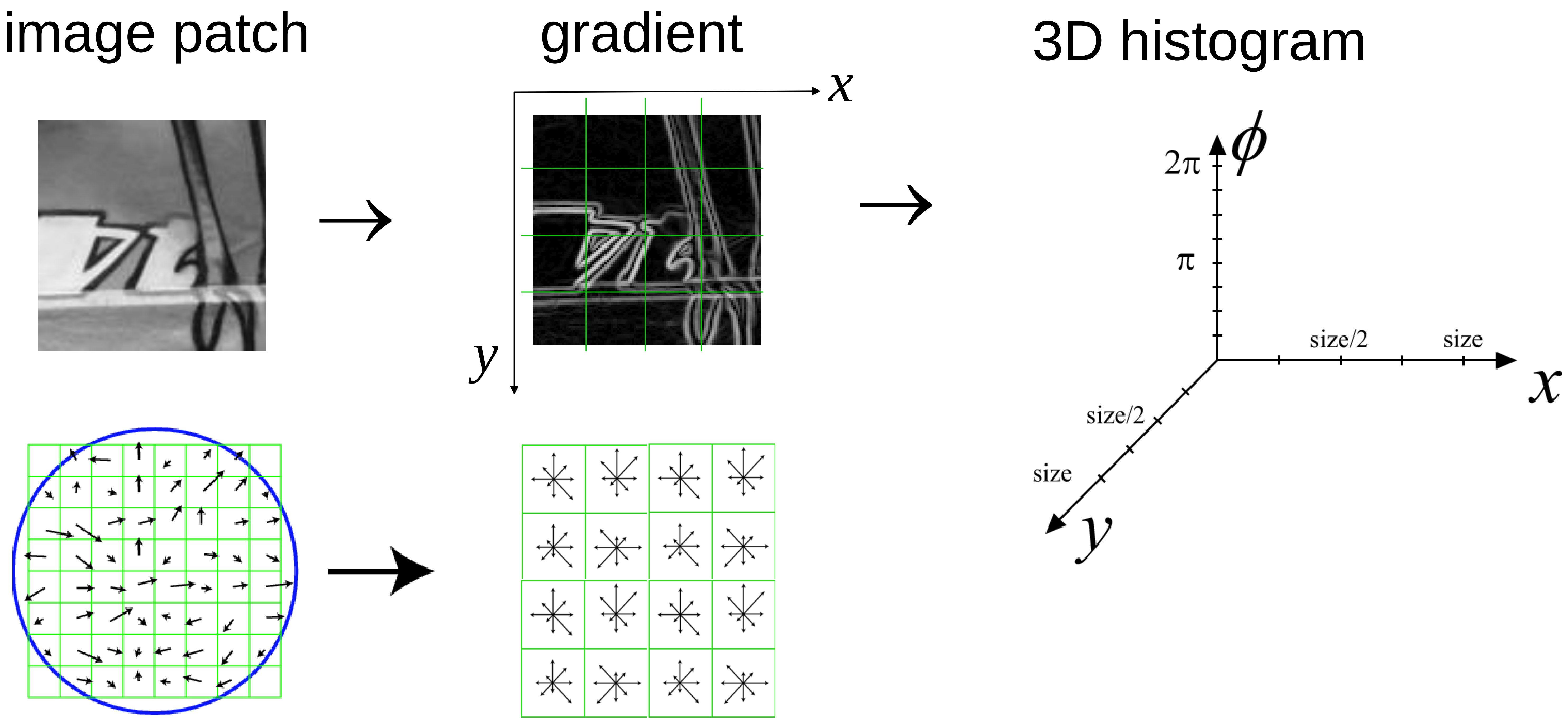


Marginal histogram

- Requires independent features
- More data/bin than joint histogram

Ex: SIFT descriptor [Lowe'99]

distribution of the gradient over an image patch

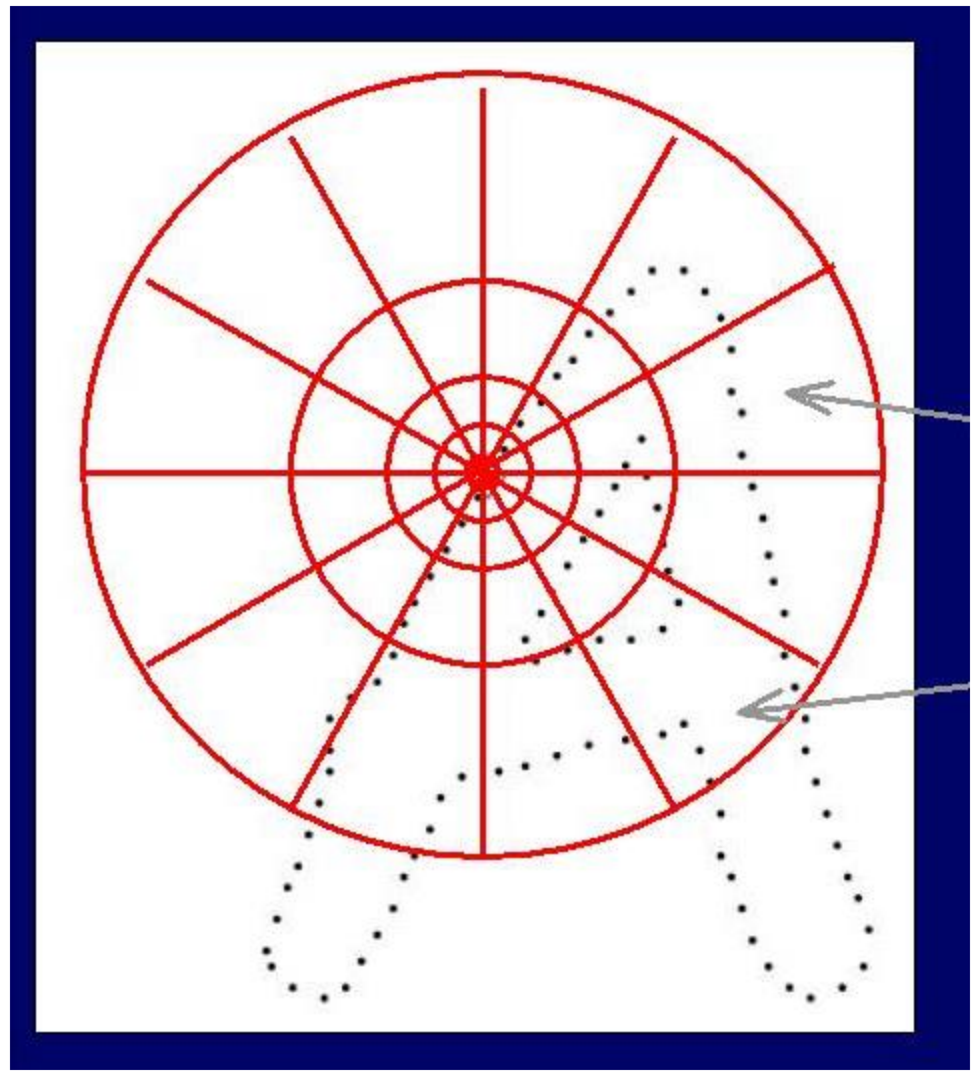


4x4 location grid and 8 orientations (128 dimensions)

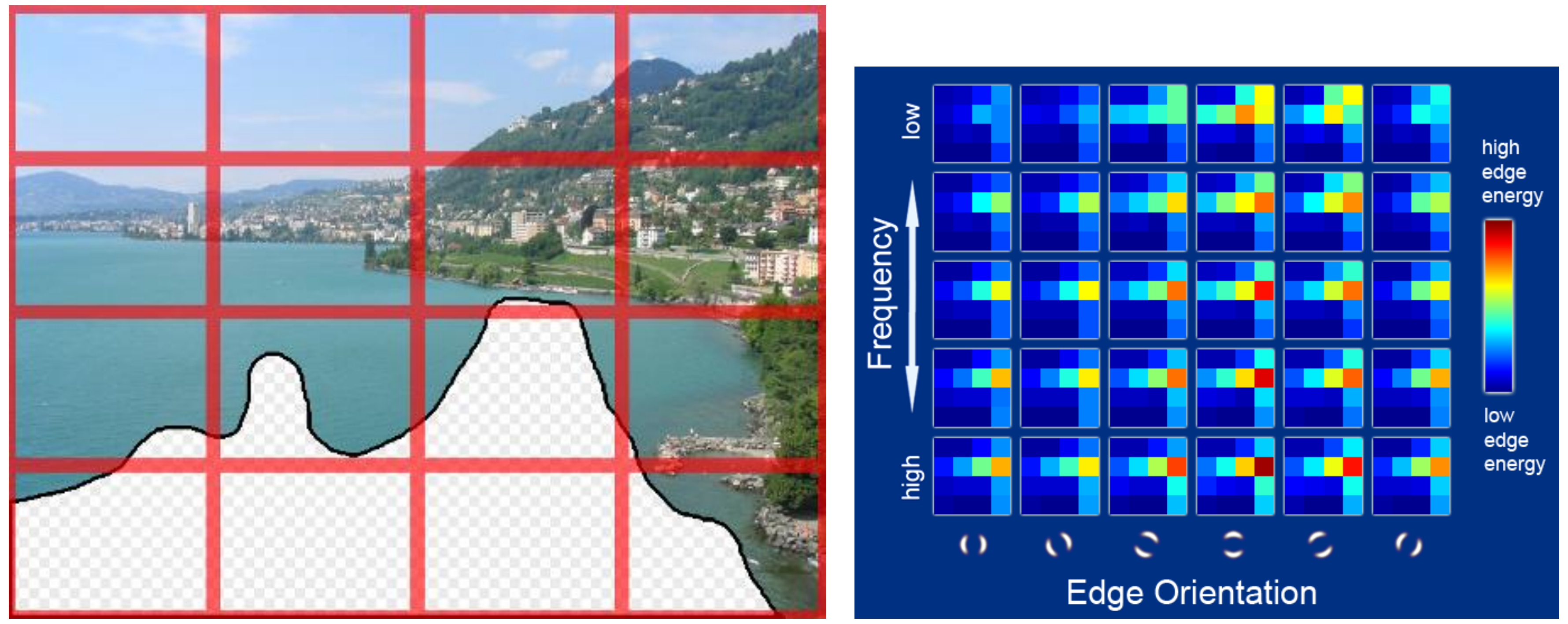
very good performance in image matching [Mikolaczyk and Schmid'03]

Gradient Histograms pop-up everywhere

HOG descriptor



Generalized Shape Context



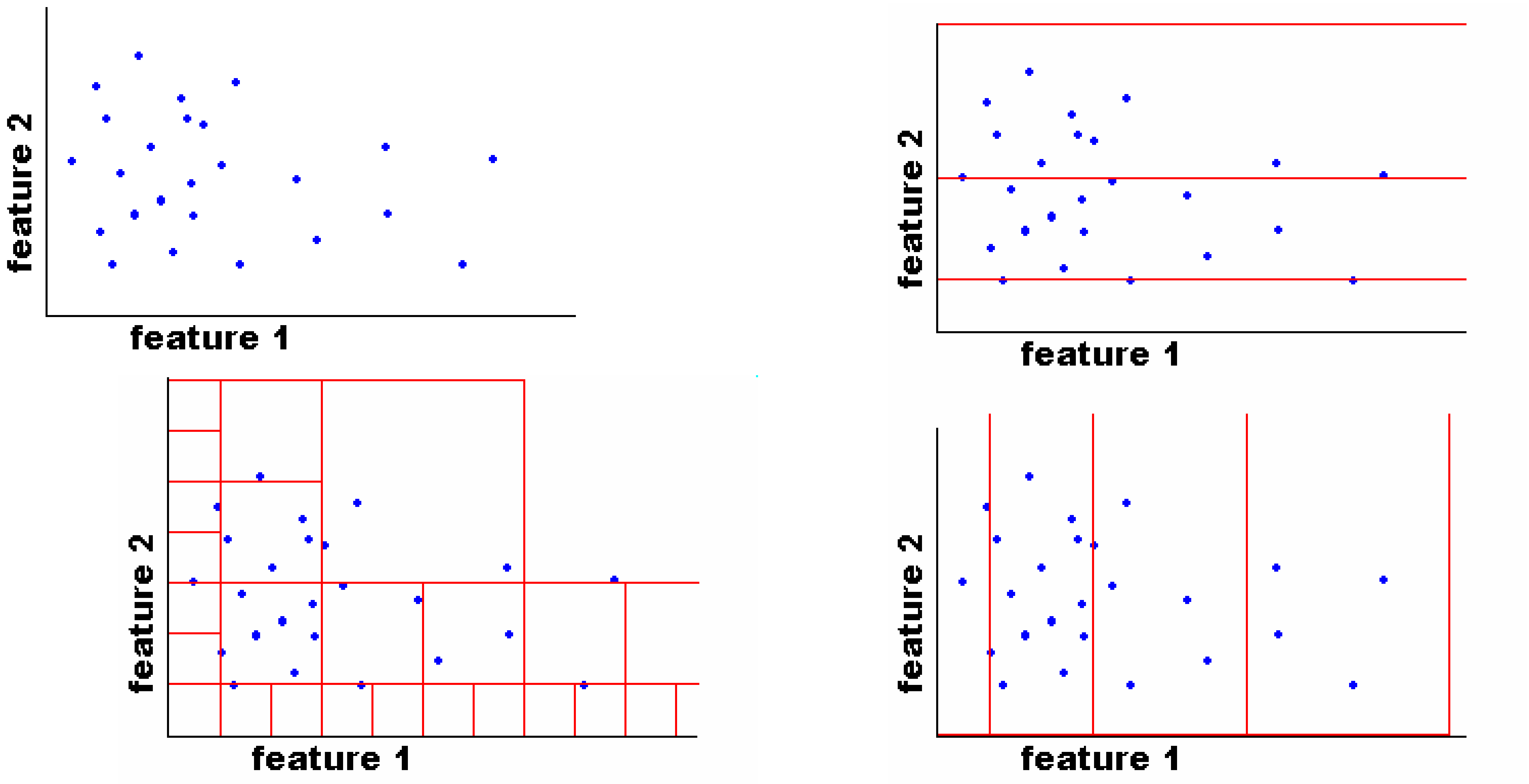
Gist Descriptor

Freeman and Roth IAFGR 1995
Lowe ICCV1999
Oliva & Torralba, 2001
Belongie et al, 2001
Dalal &Triggs CVPR05

Binning achieves invariance to small patch offsets

Adaptive Representations

Images from Dave Kauchak

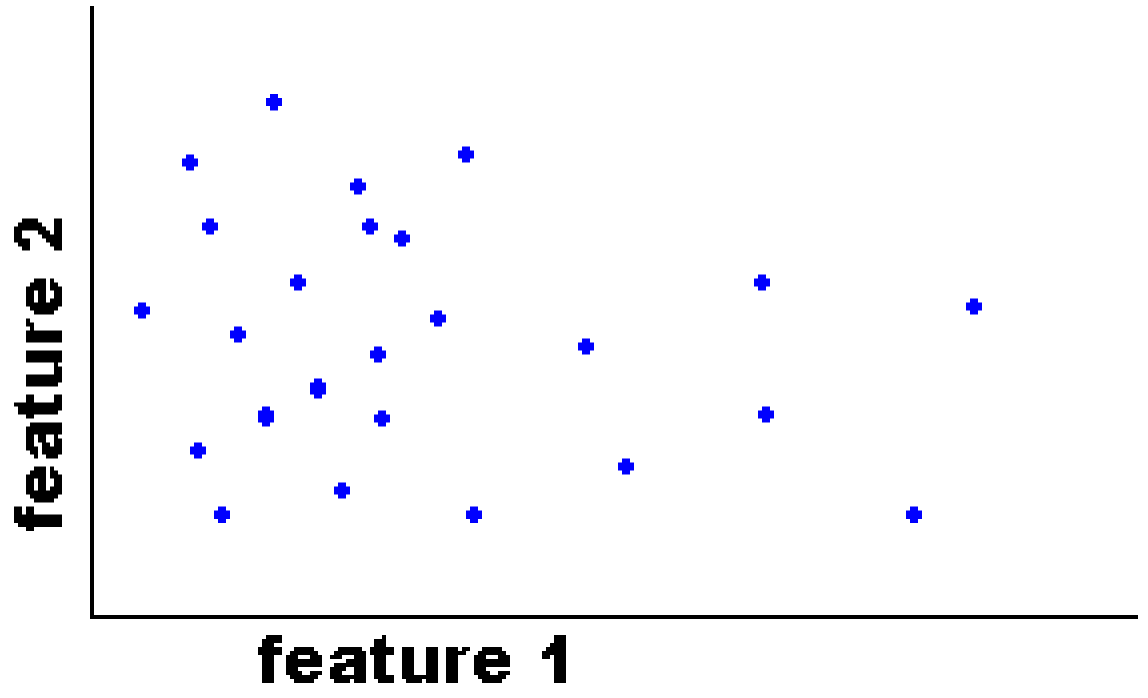


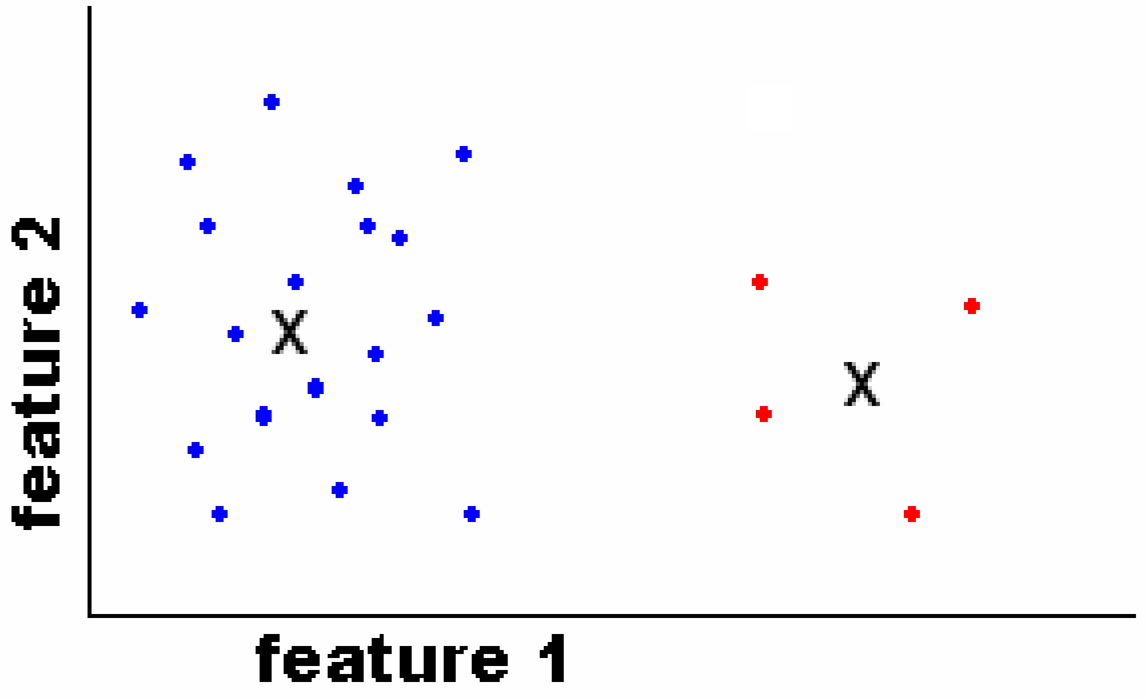
Adaptive binning

- Better data/bin distribution, fewer empty bins
- Can adapt available resolution to relative feature importance

Clustering: very adaptive representations

Images from Dave Kauchak



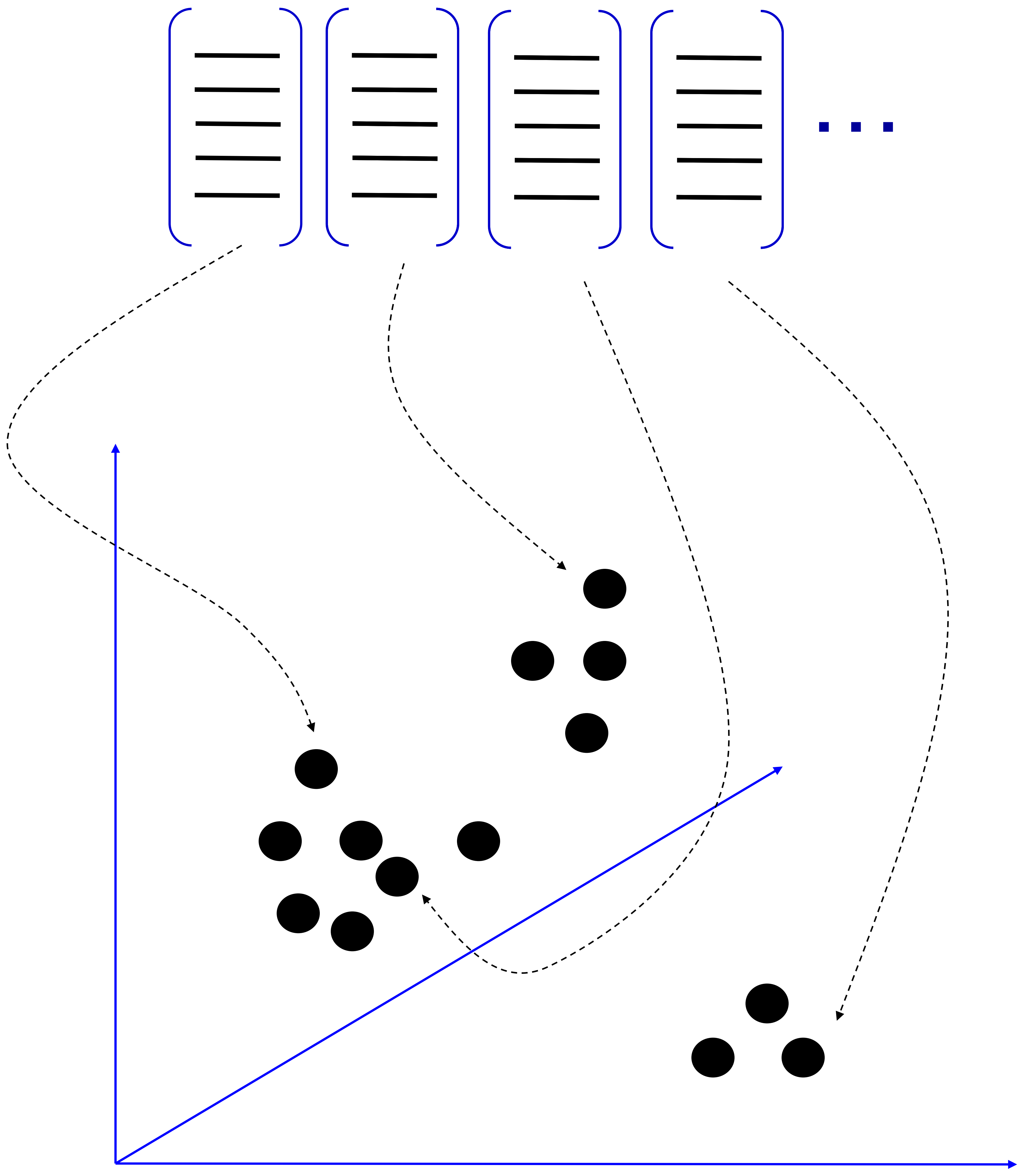


Clusters / Signatures

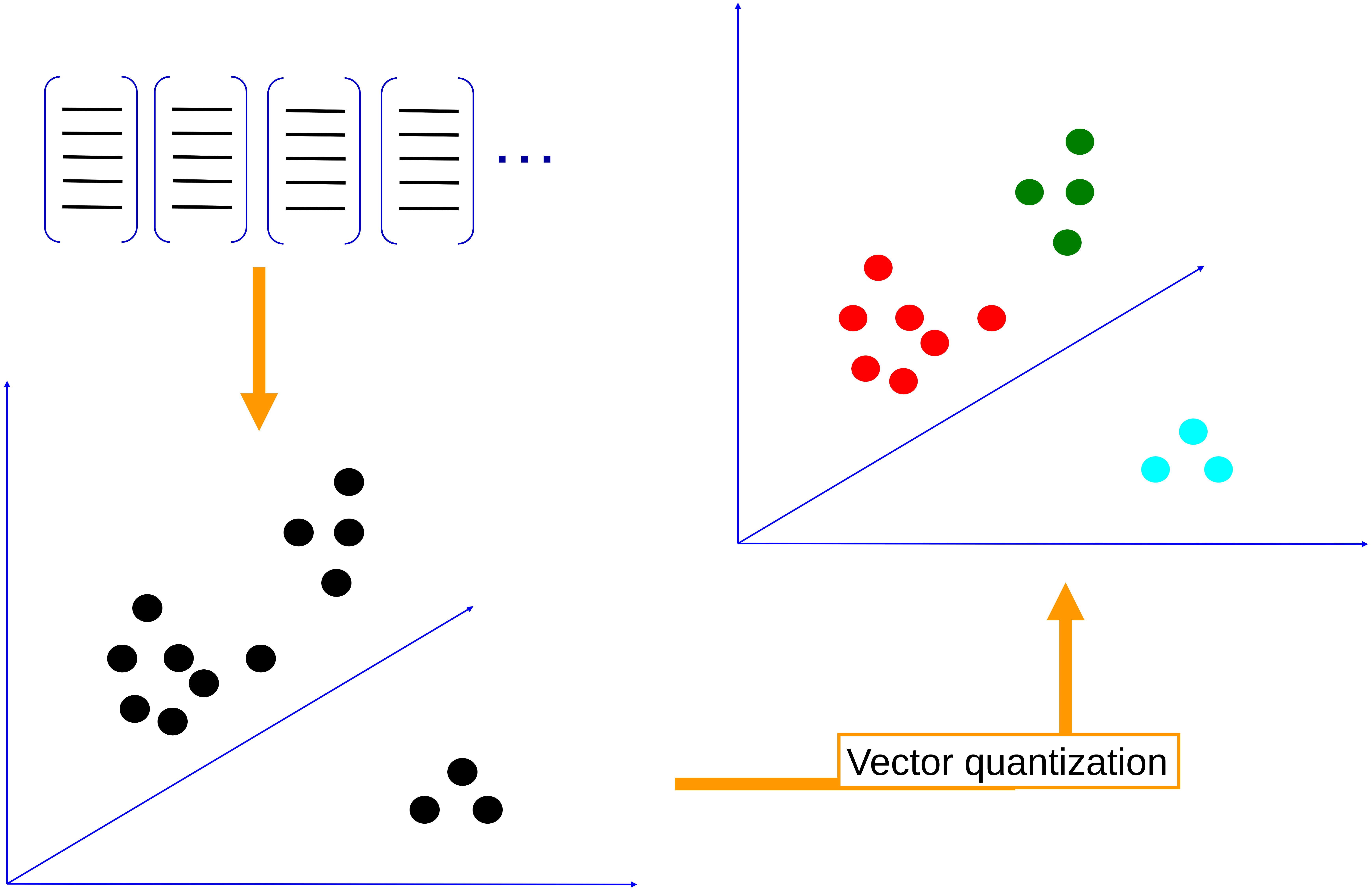
- “super-adaptive” binning
- Does not require discretization along any fixed axis

Patch Features

dictionary formation

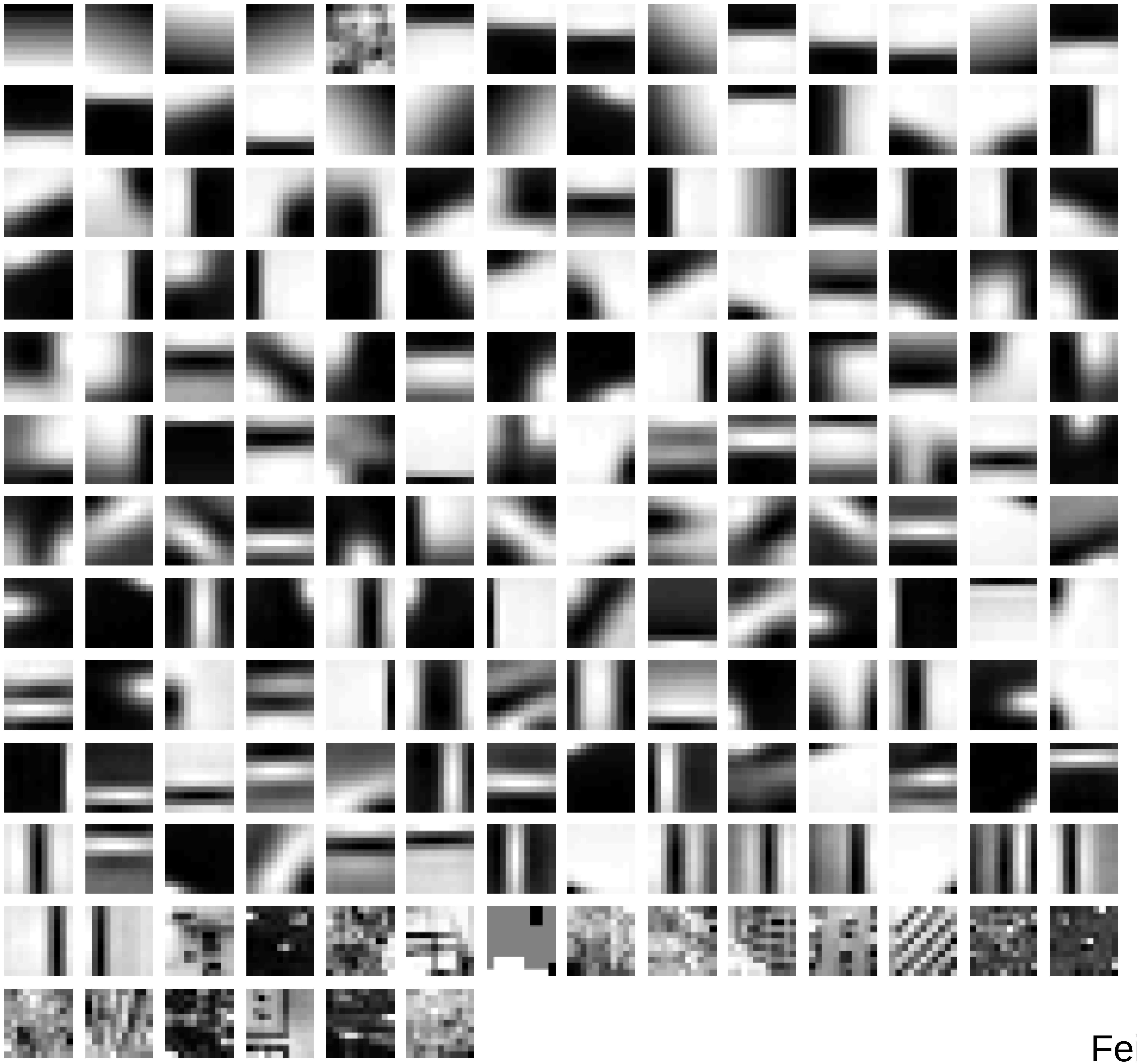


Clustering (usually k-means)



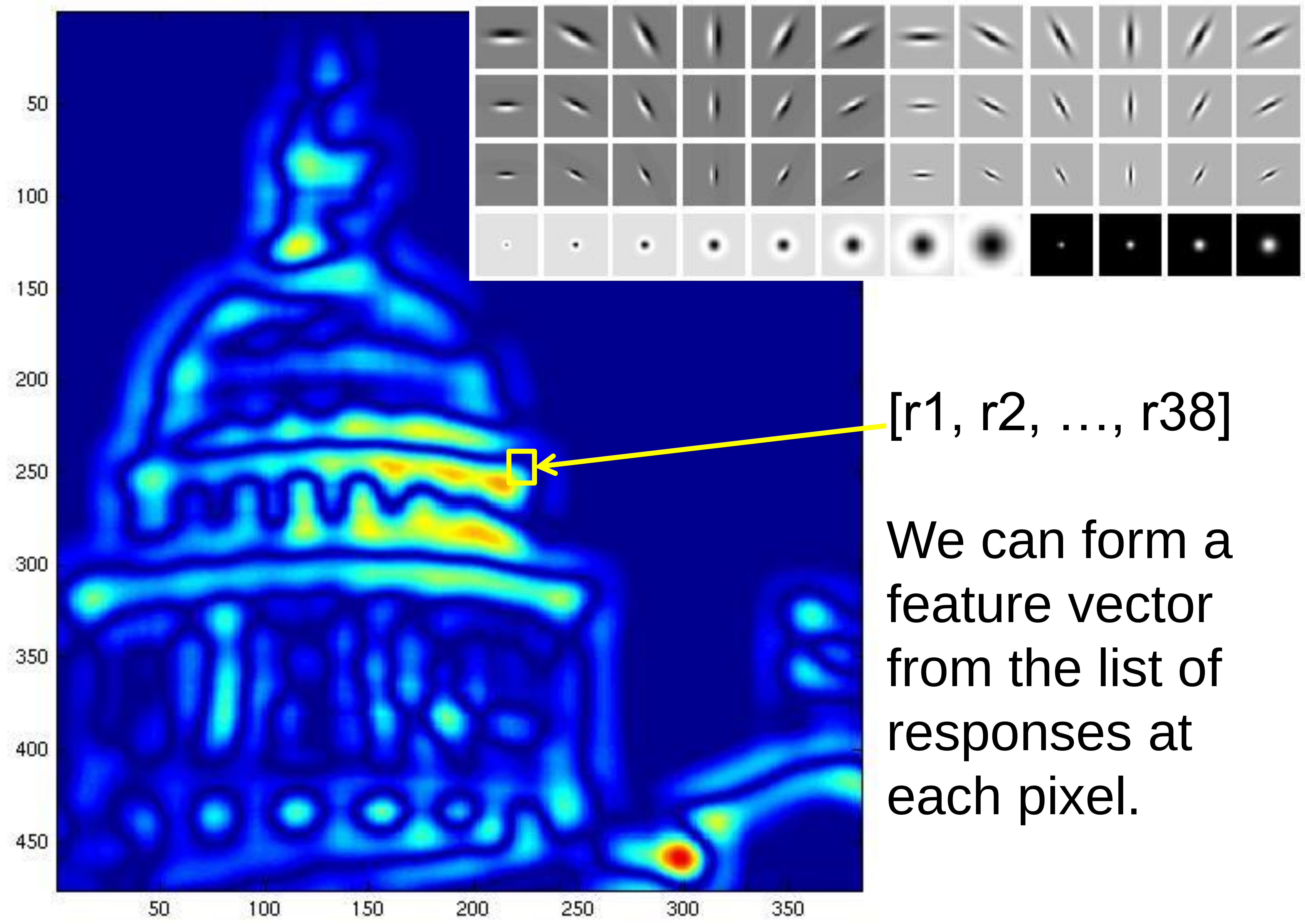
Slide credit: Josef Sivic

Clustered Image Patches



Fei-Fei et al. 2005

Filter Response Vectors

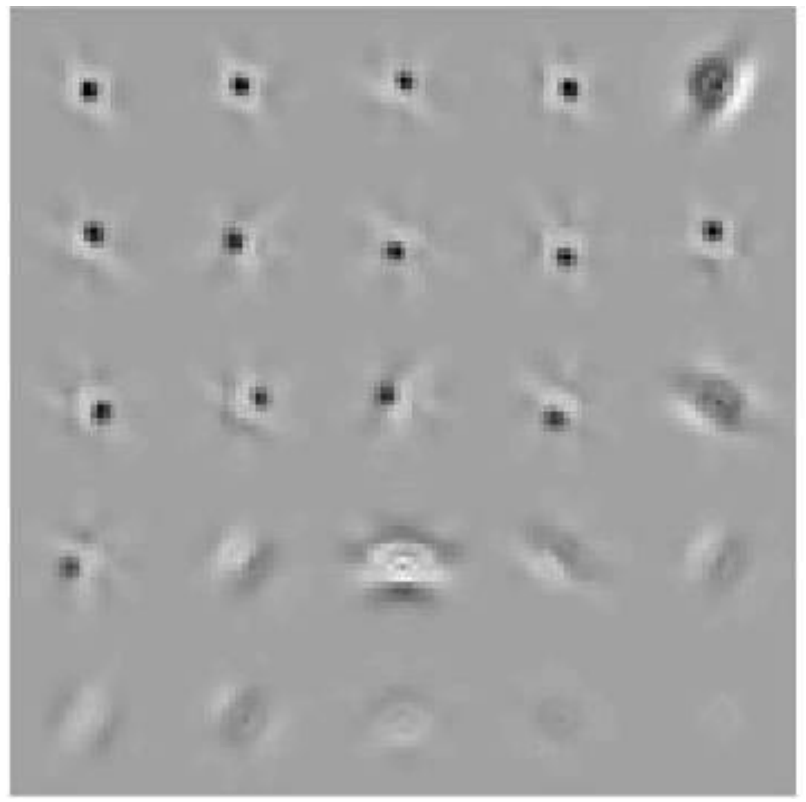


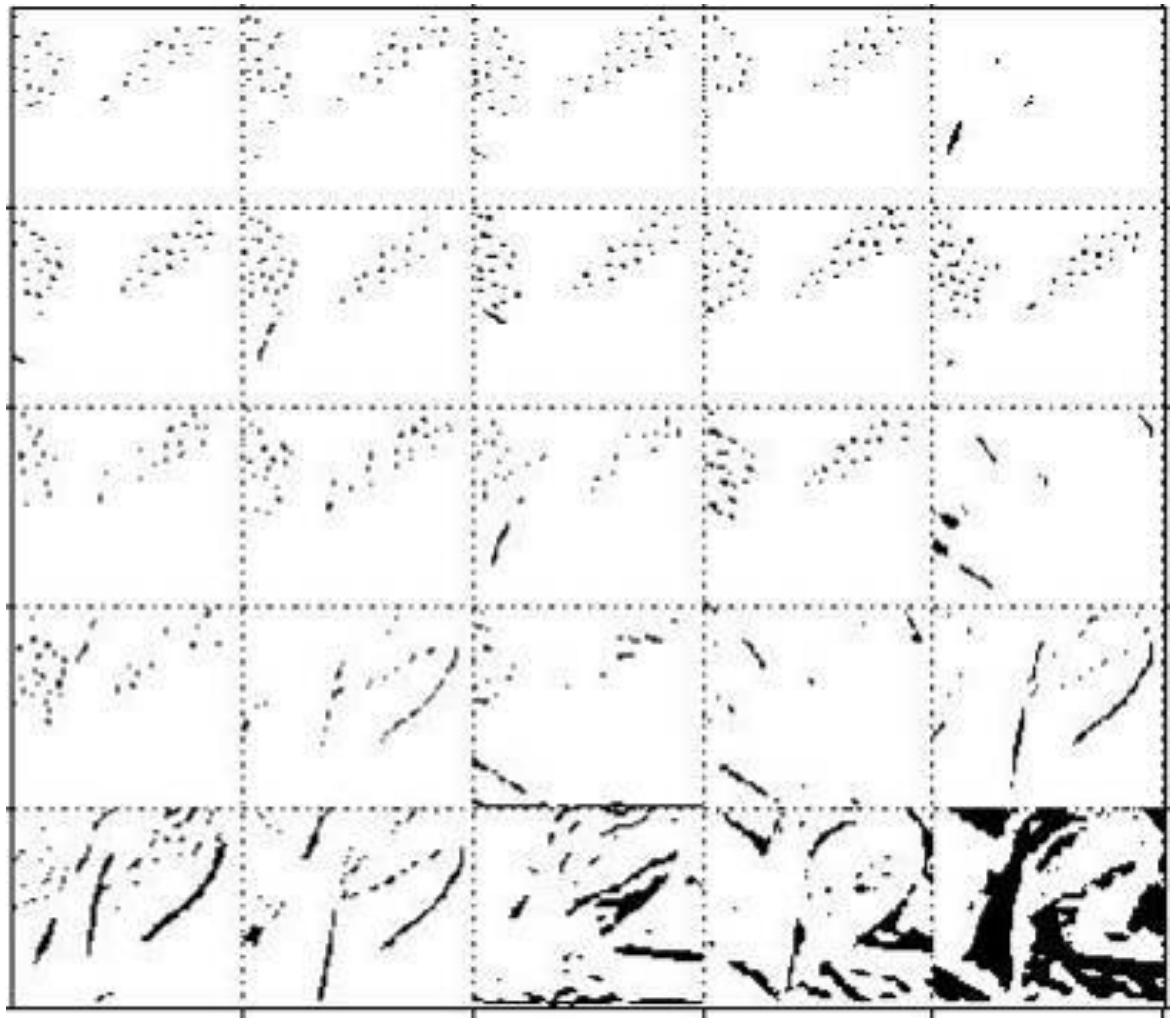
[r1, r2, …, r38]

We can form a feature vector from the list of responses at each pixel.

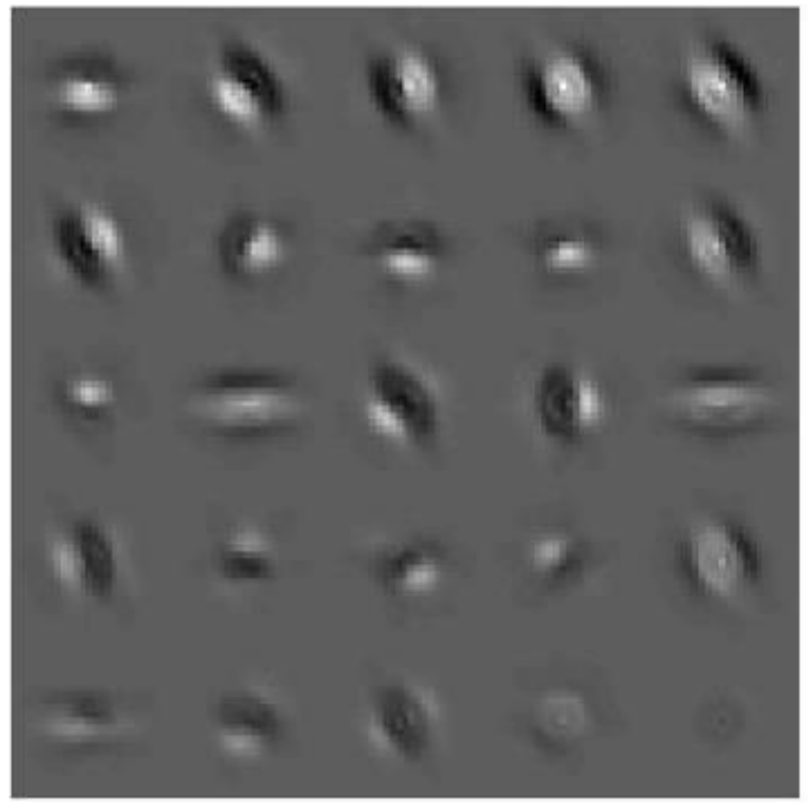
Textons (Malik et al, IJCV 2001)

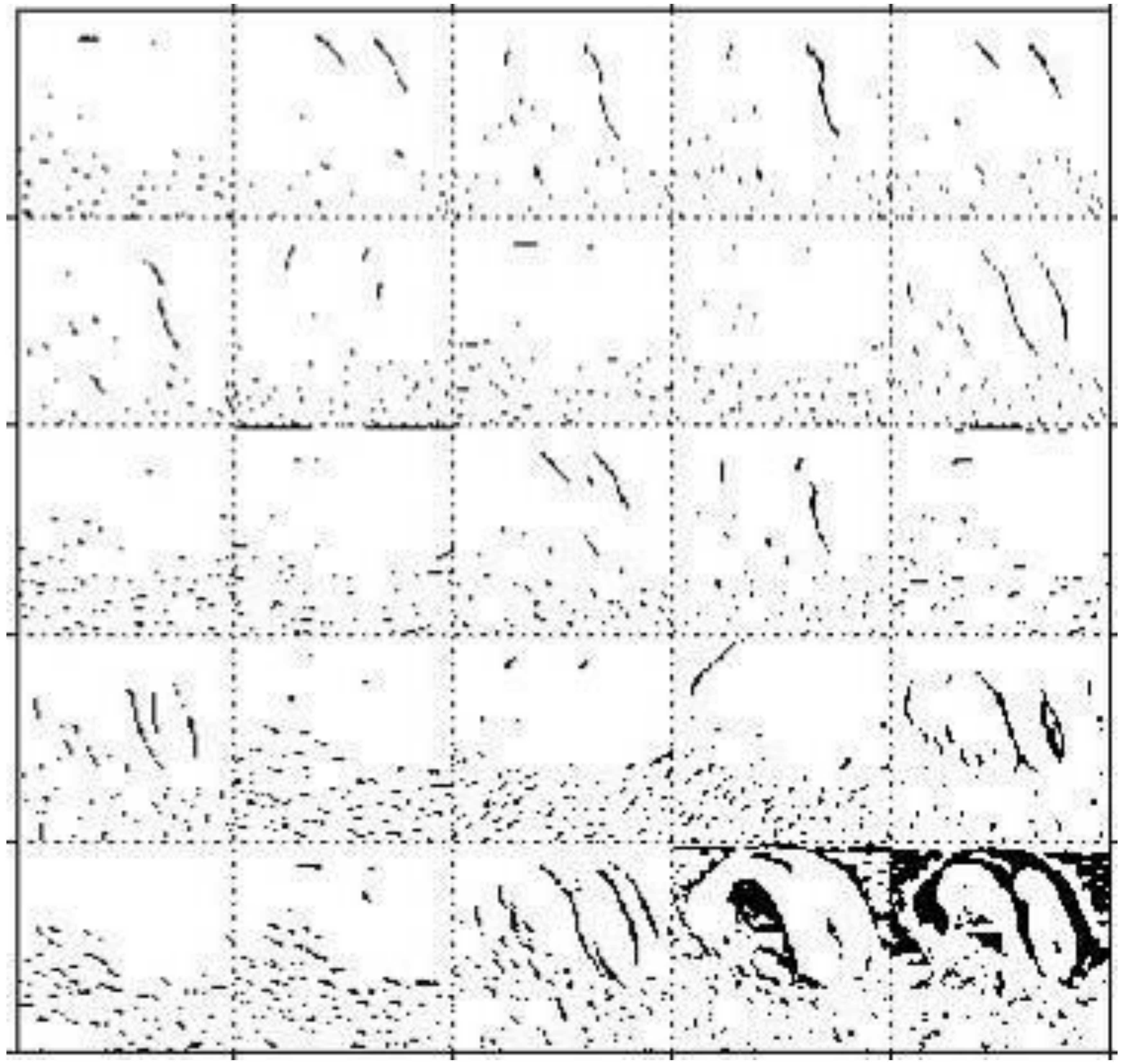
Cluster vectors of filter responses



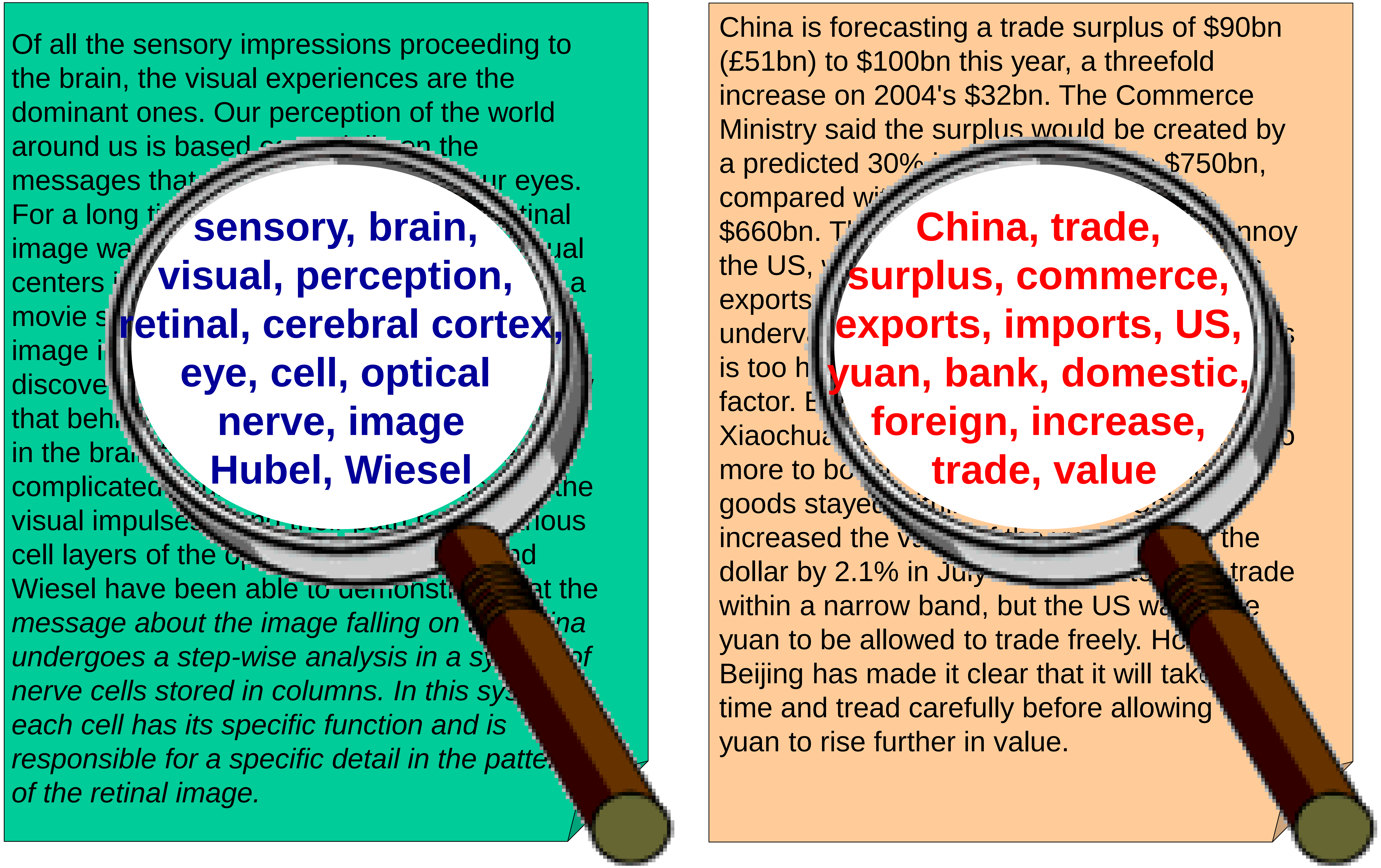


Textons (cont.)





“bag of words”: cluster histograms



Object → **Bag of ‘words’**

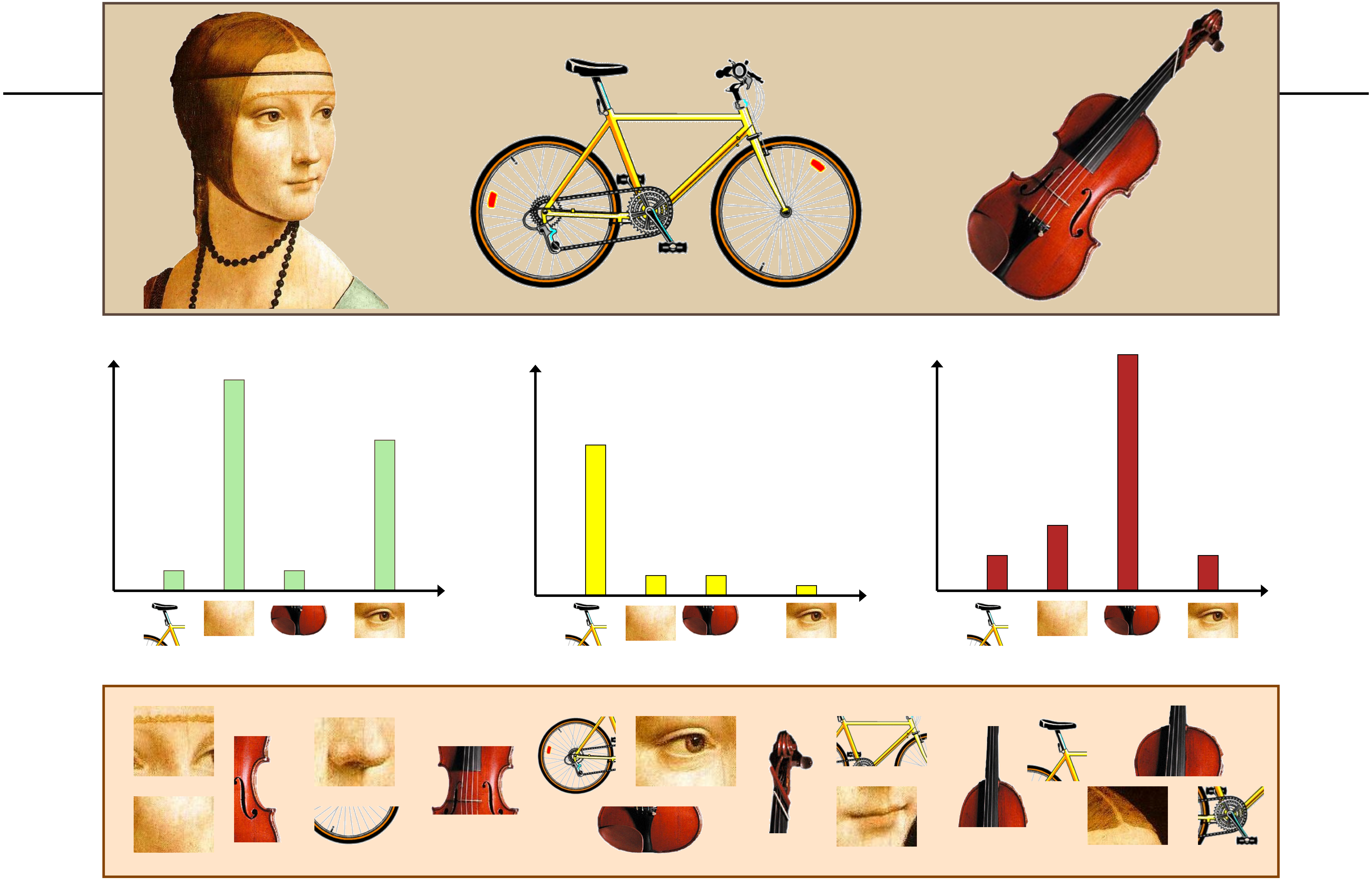
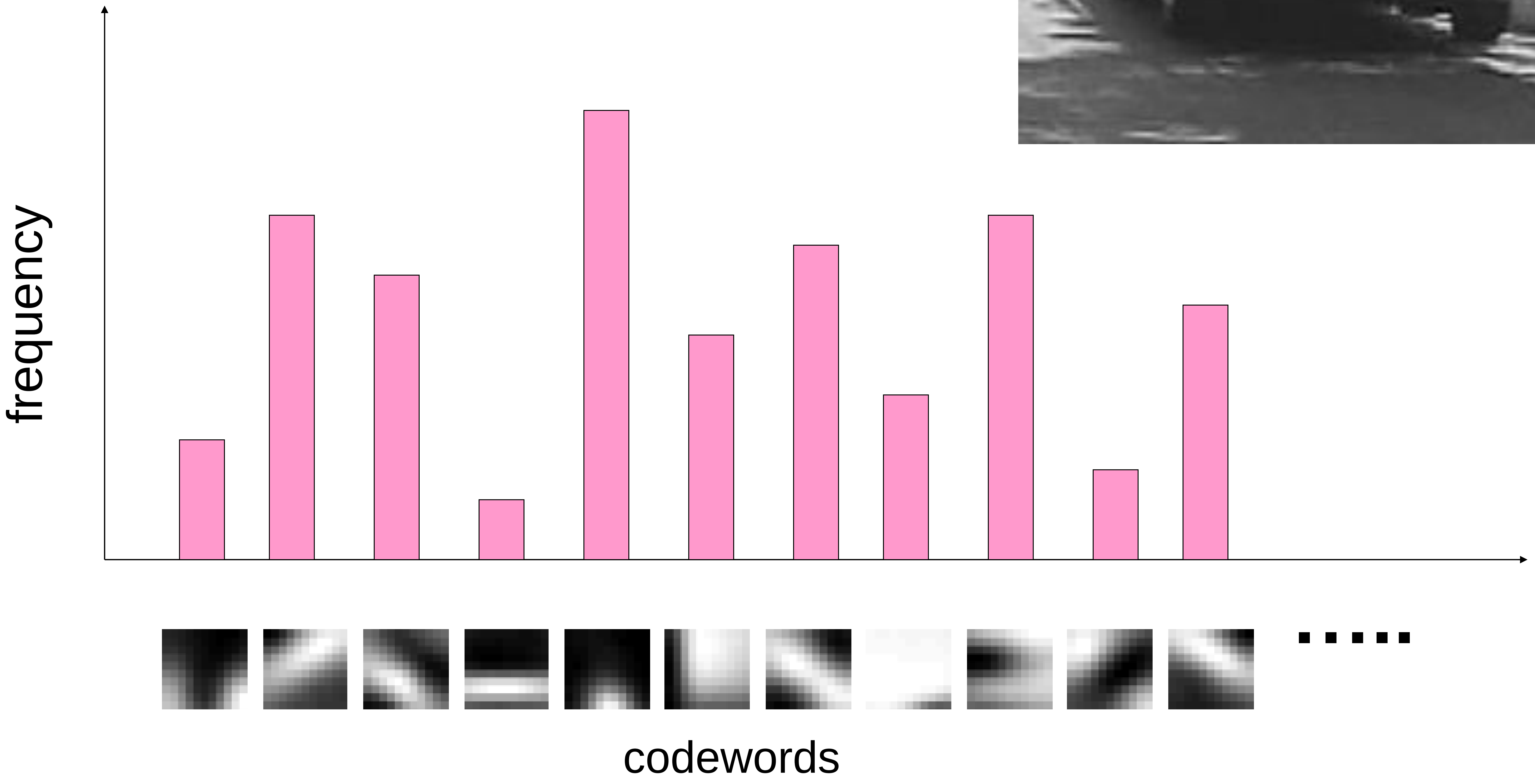


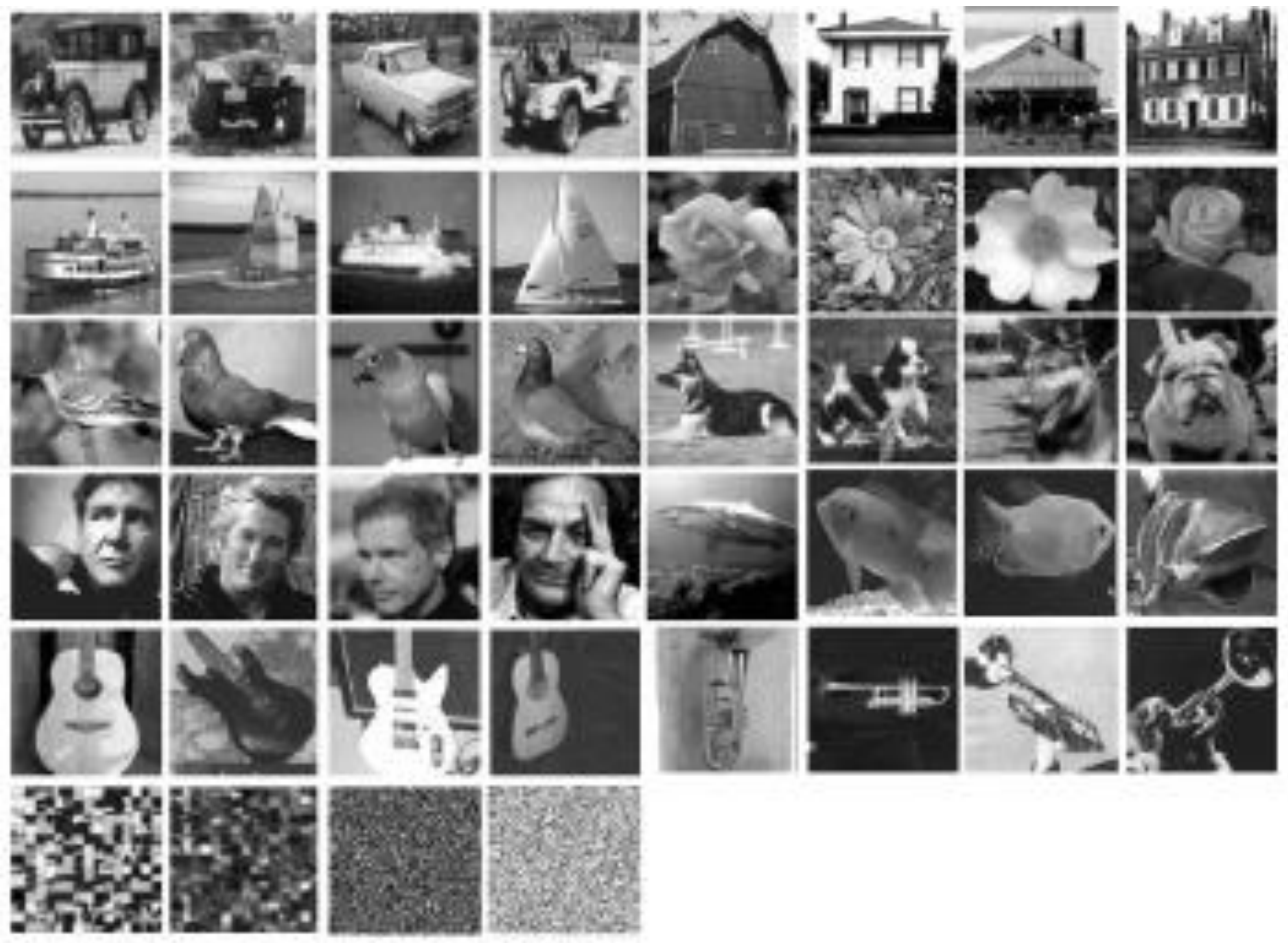
Image representation



Scene Classification (Renninger & Malik)

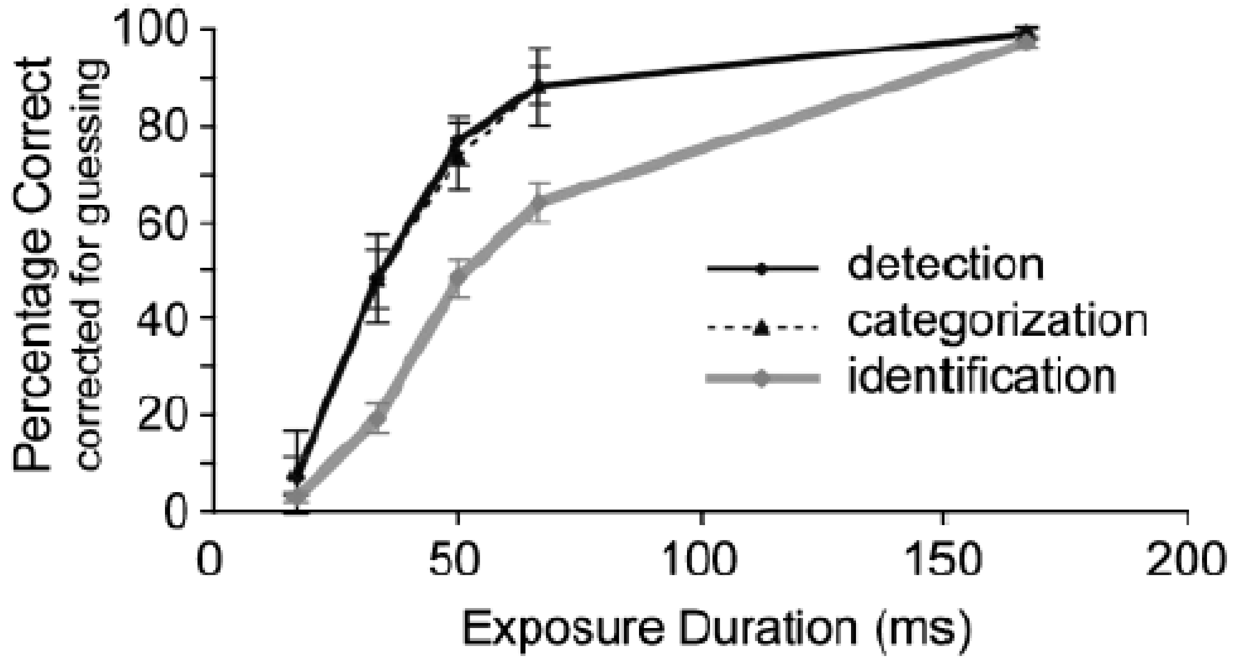
Image classification can be pre-attentive!

On a task of judging animal vs no animal, humans can make mostly correct saccades in 150 ms (Kirchner & Thorpe, 2006)

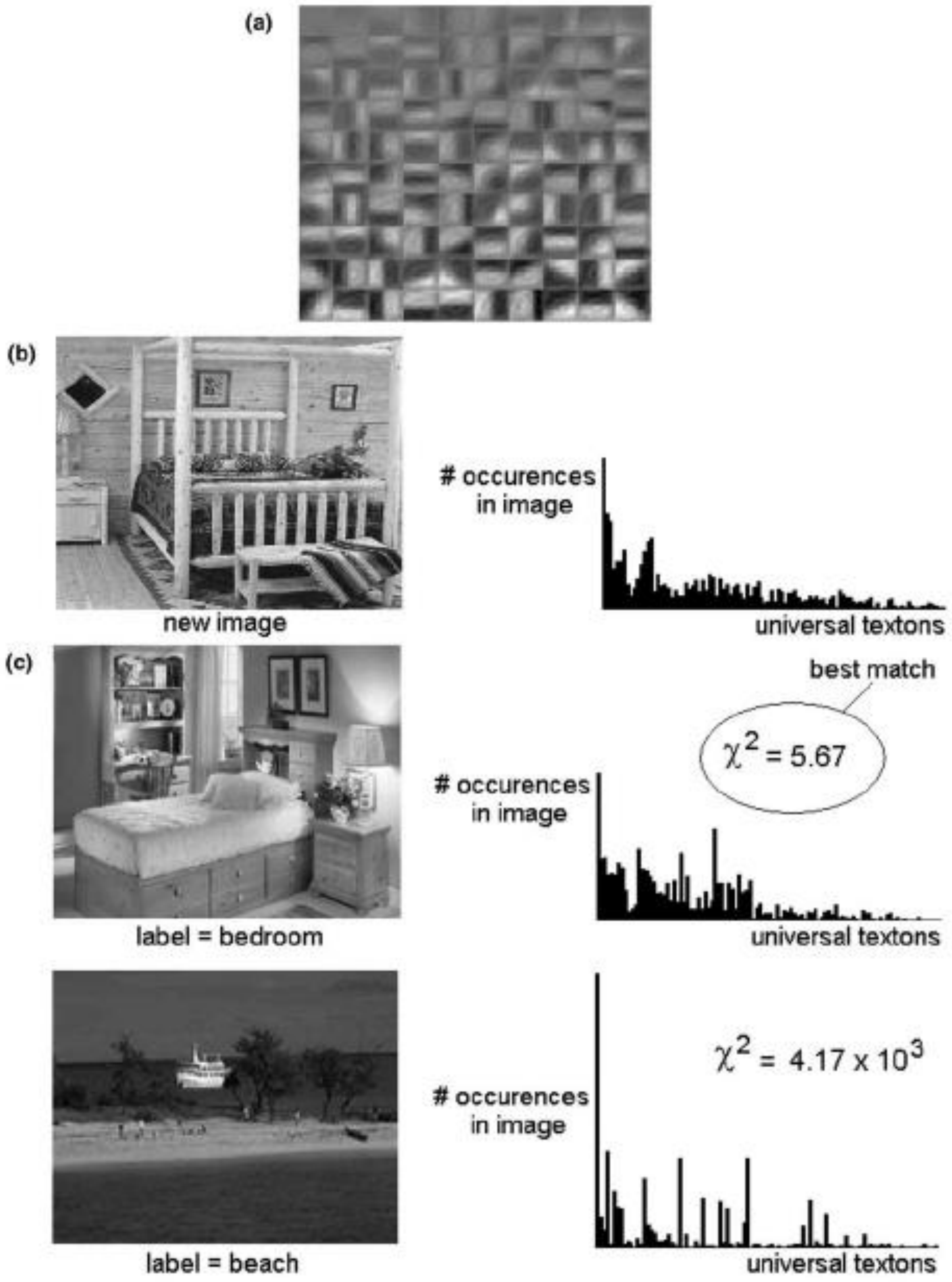


- Comparable to synaptic delay in the retina, LGN, V1, V2, V4, IT pathway.
- Doesn't rule out feed back but shows feed forward only is very powerful

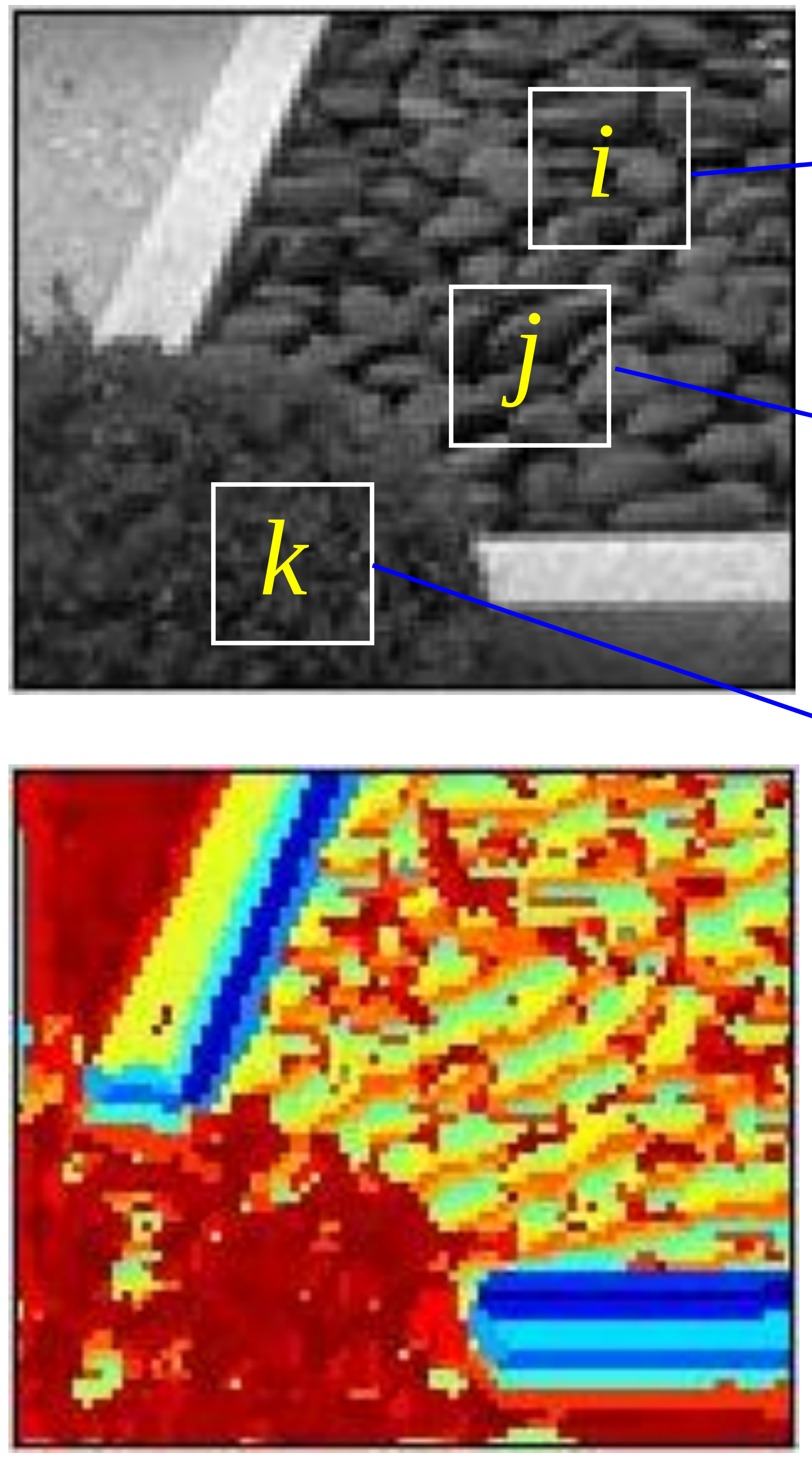
Detection and categorization are practically simultaneous (Grill-Spector & Kanwisher, 2005)

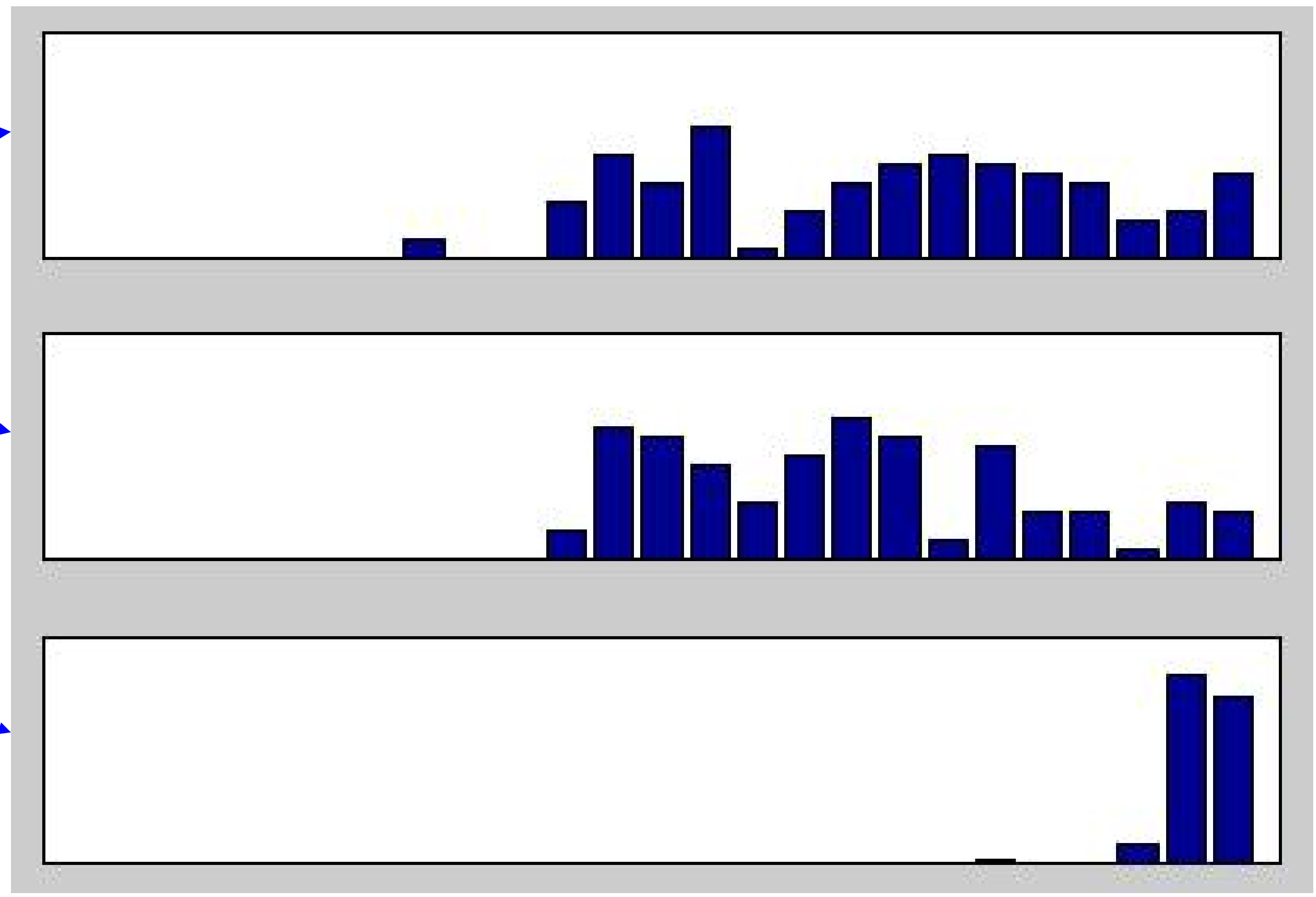


Texton Histogram Matching



Texton histograms work on patches too

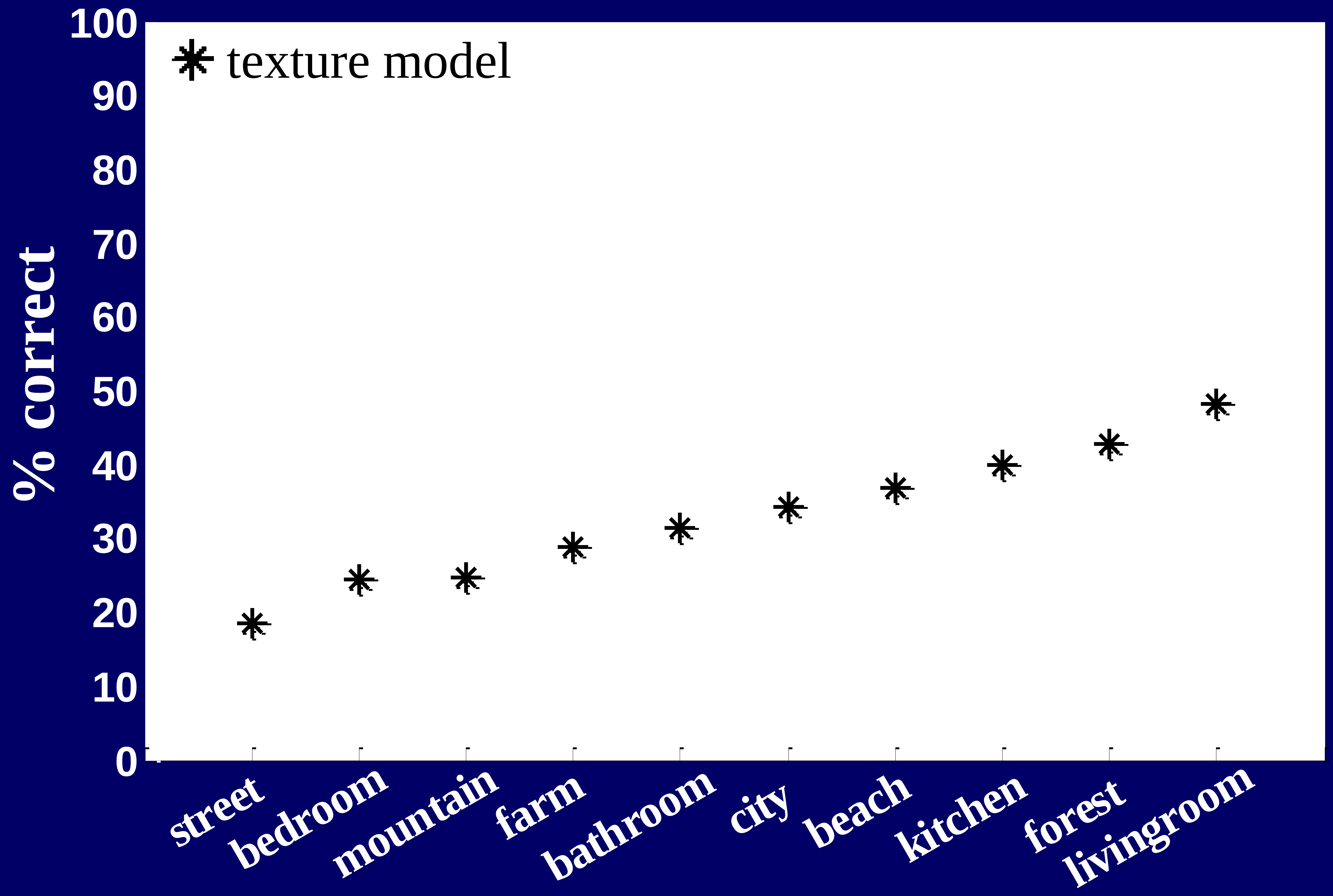




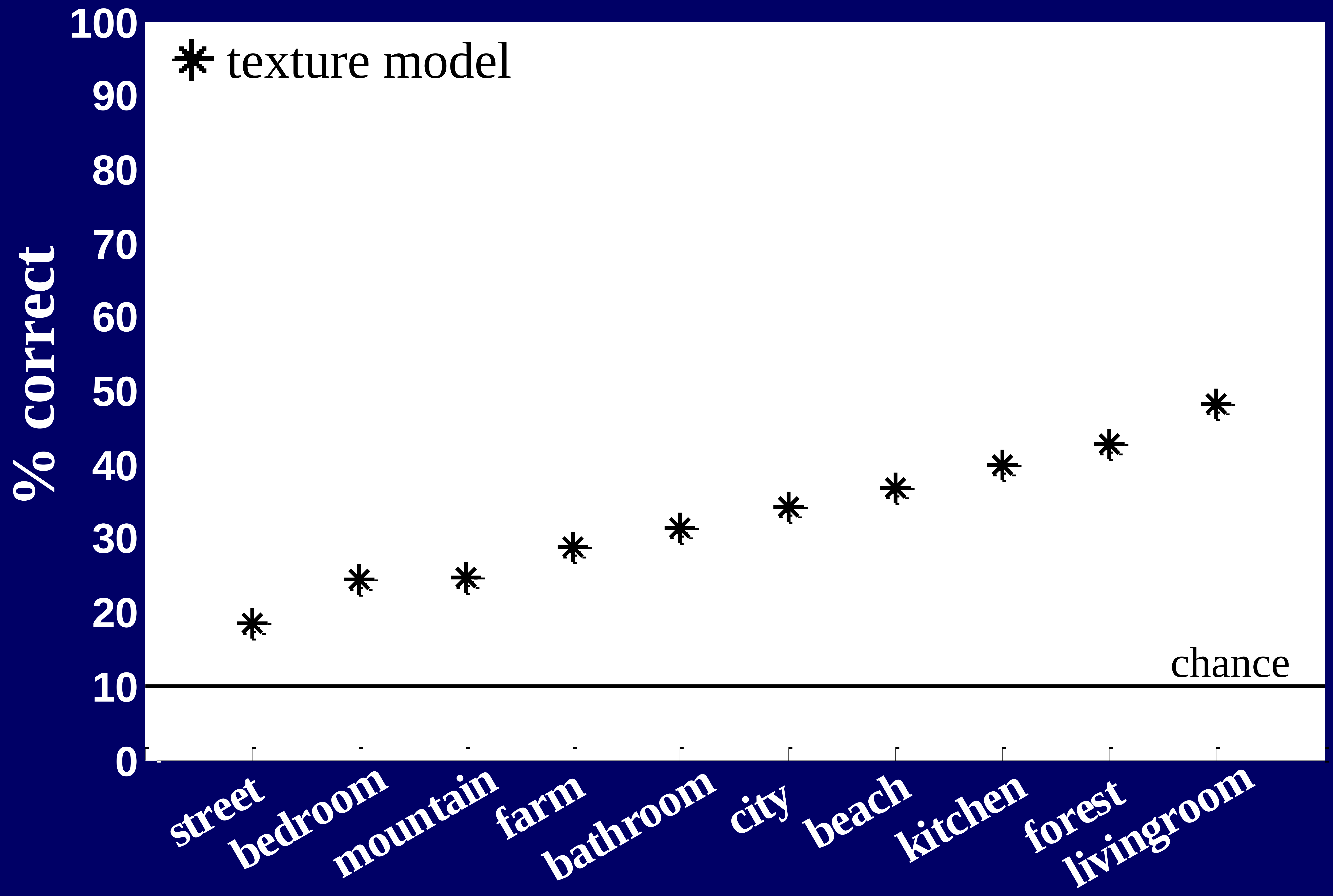
$$\chi^2(h_i, h_j) = \frac{1}{2}\sum_{m=1}^{K}\frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$

(Malik et al, 99)

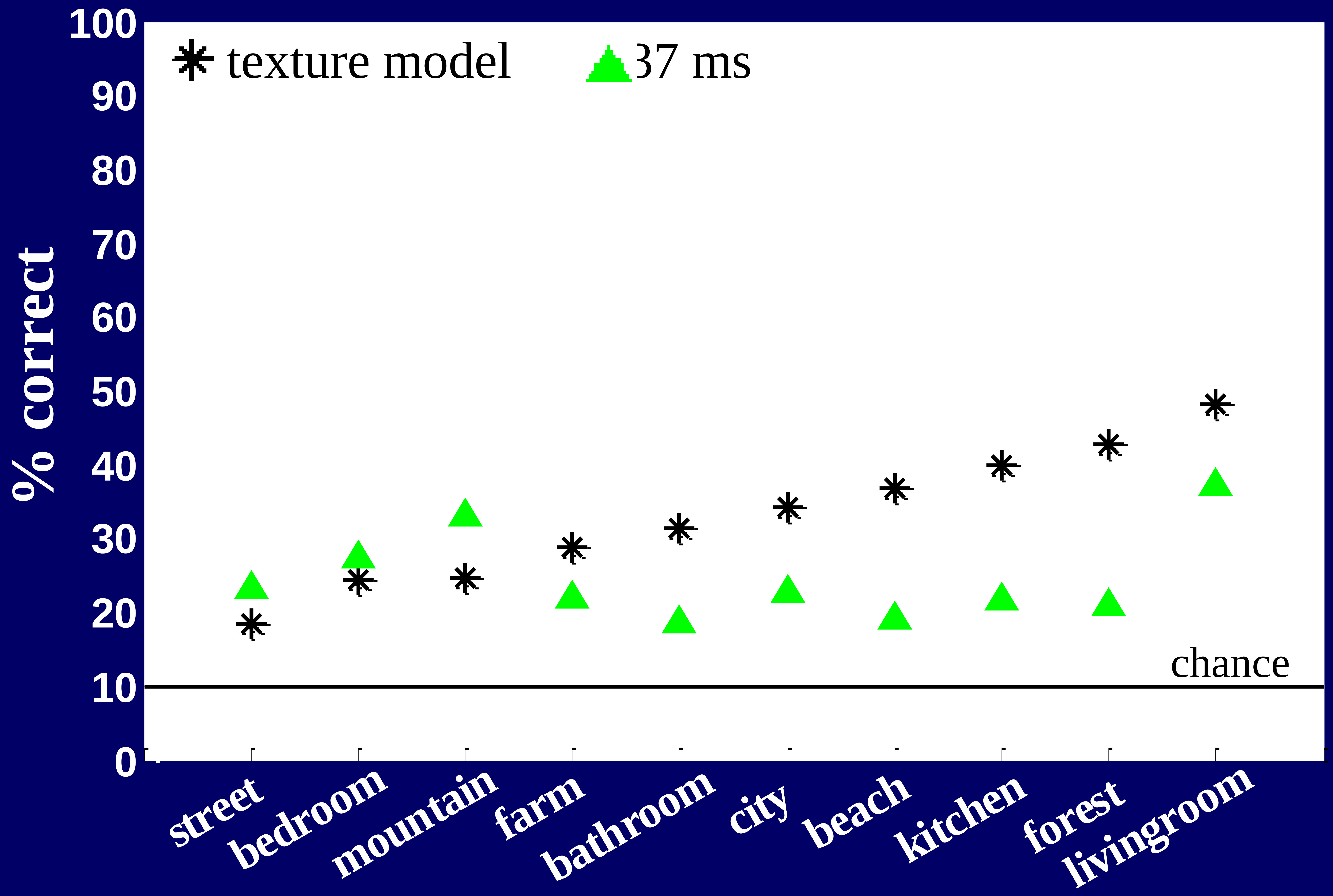
Discrimination of Basic Categories



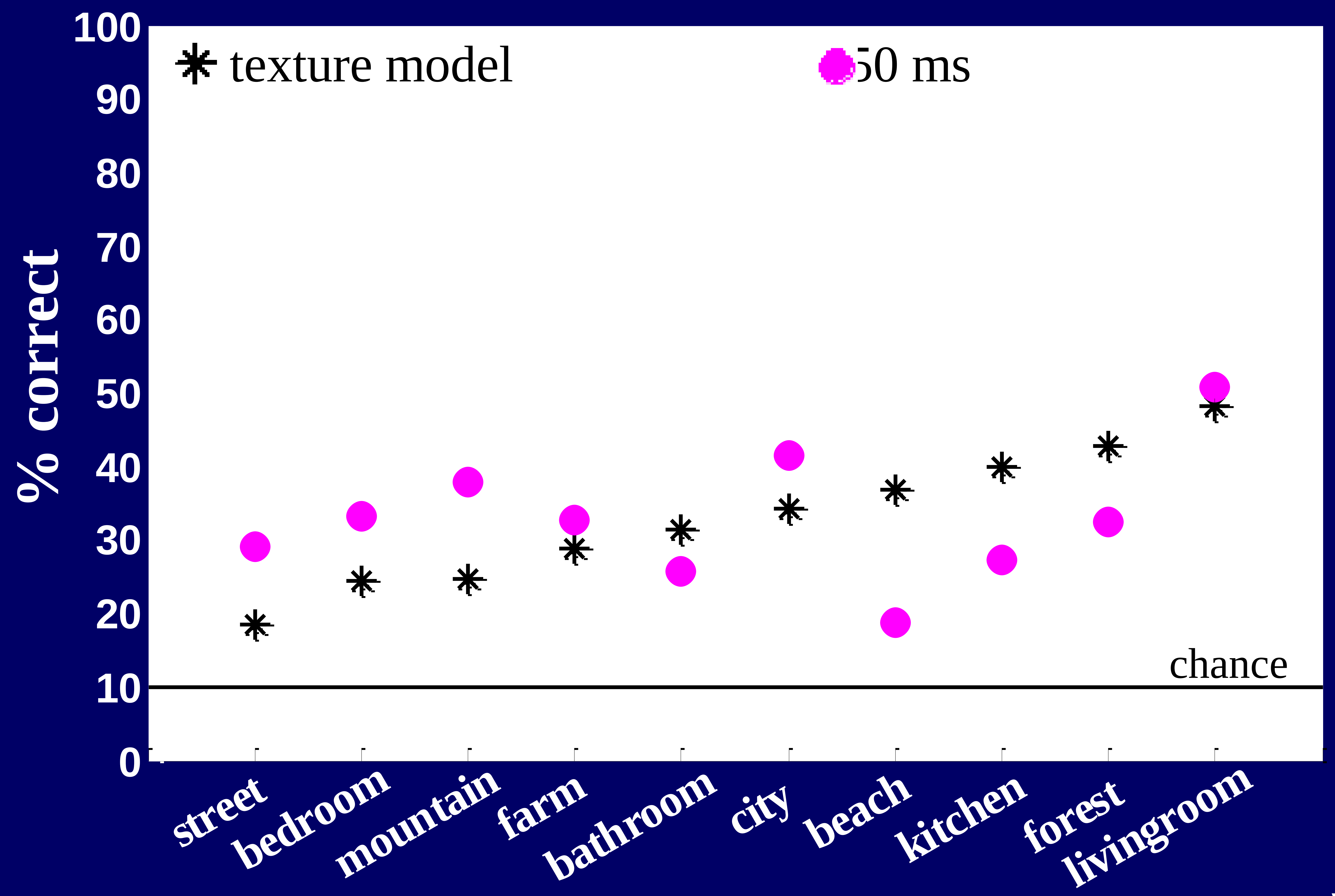
Discrimination of Basic Categories



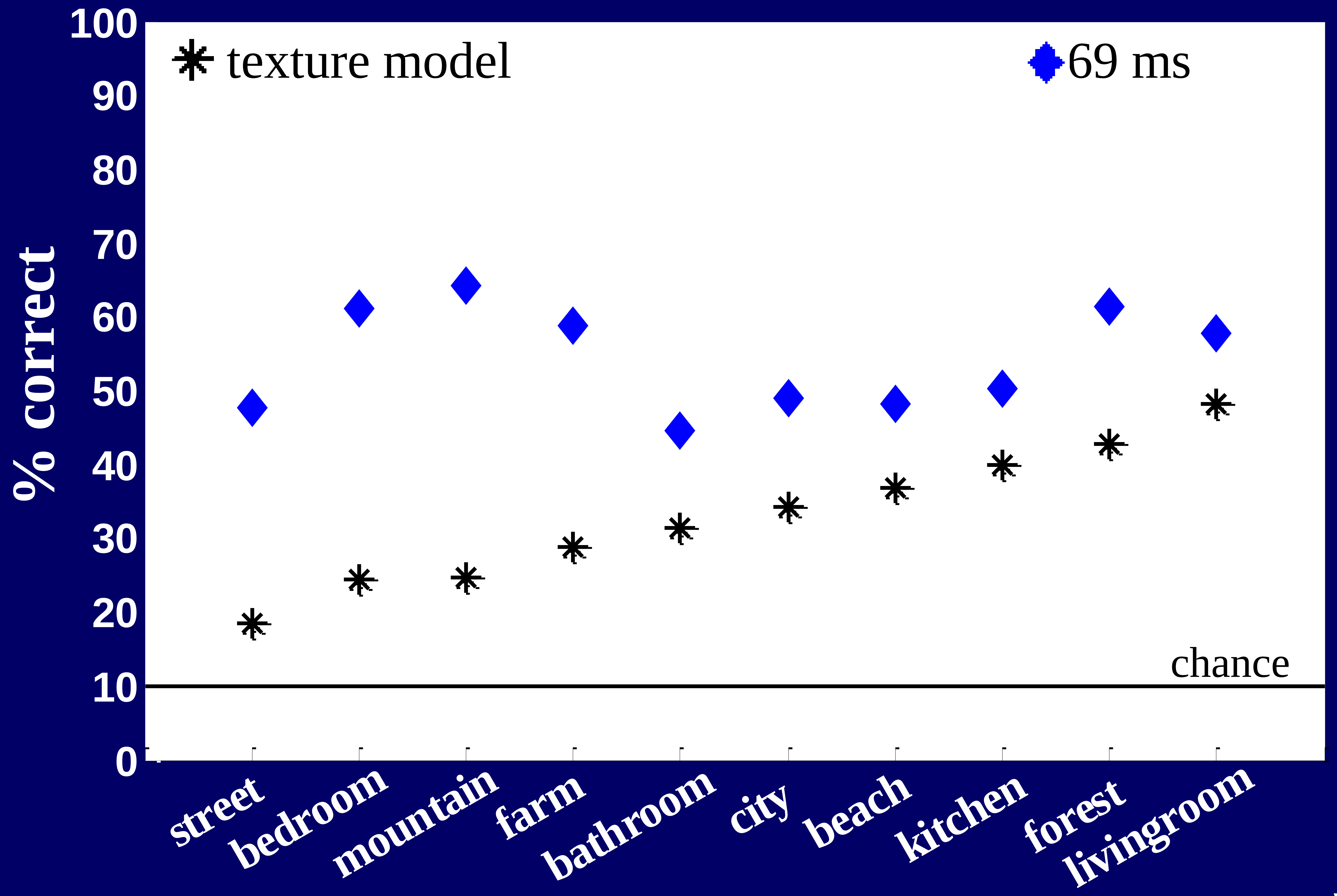
Discrimination of Basic Categories



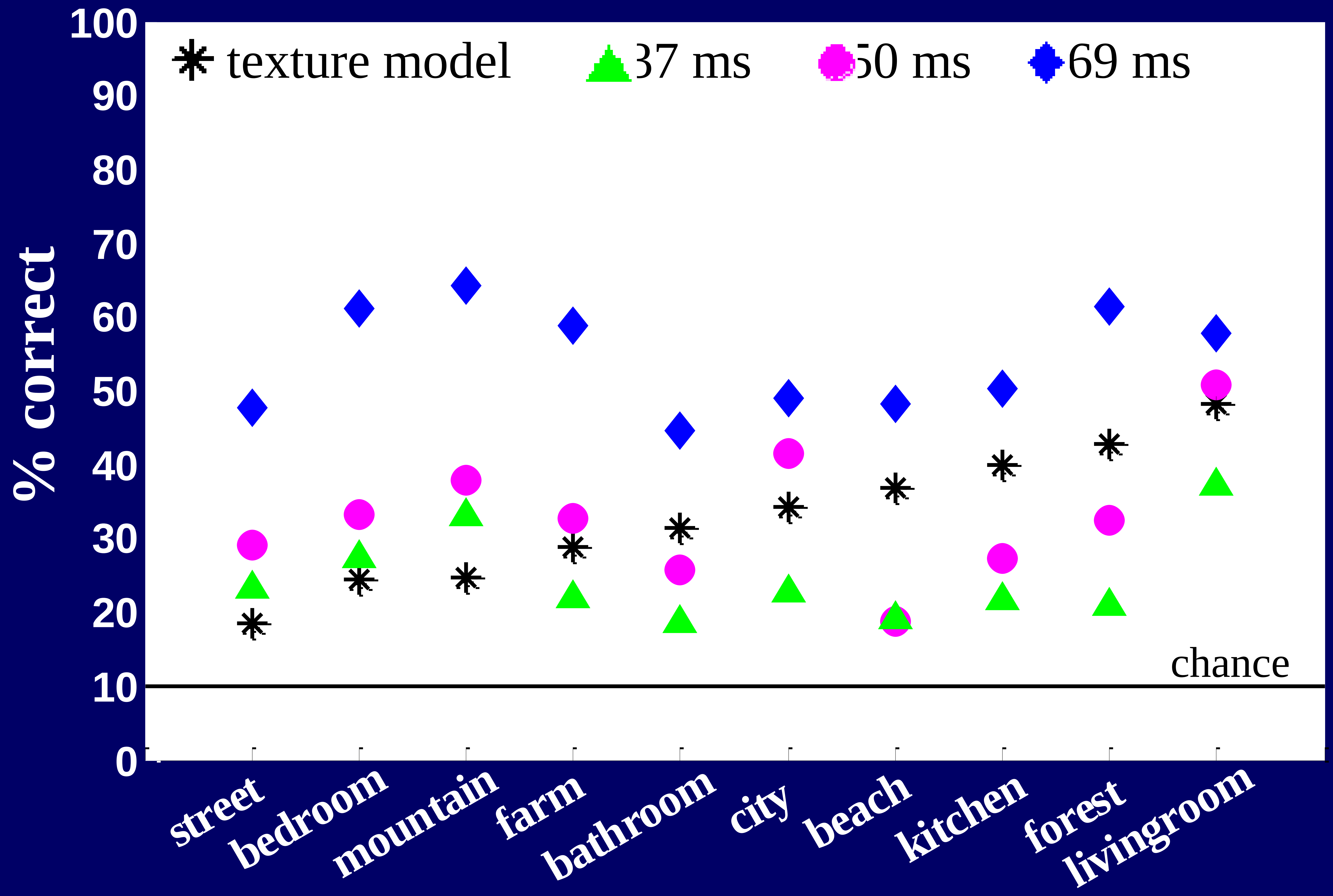
Discrimination of Basic Categories



Discrimination of Basic Categories

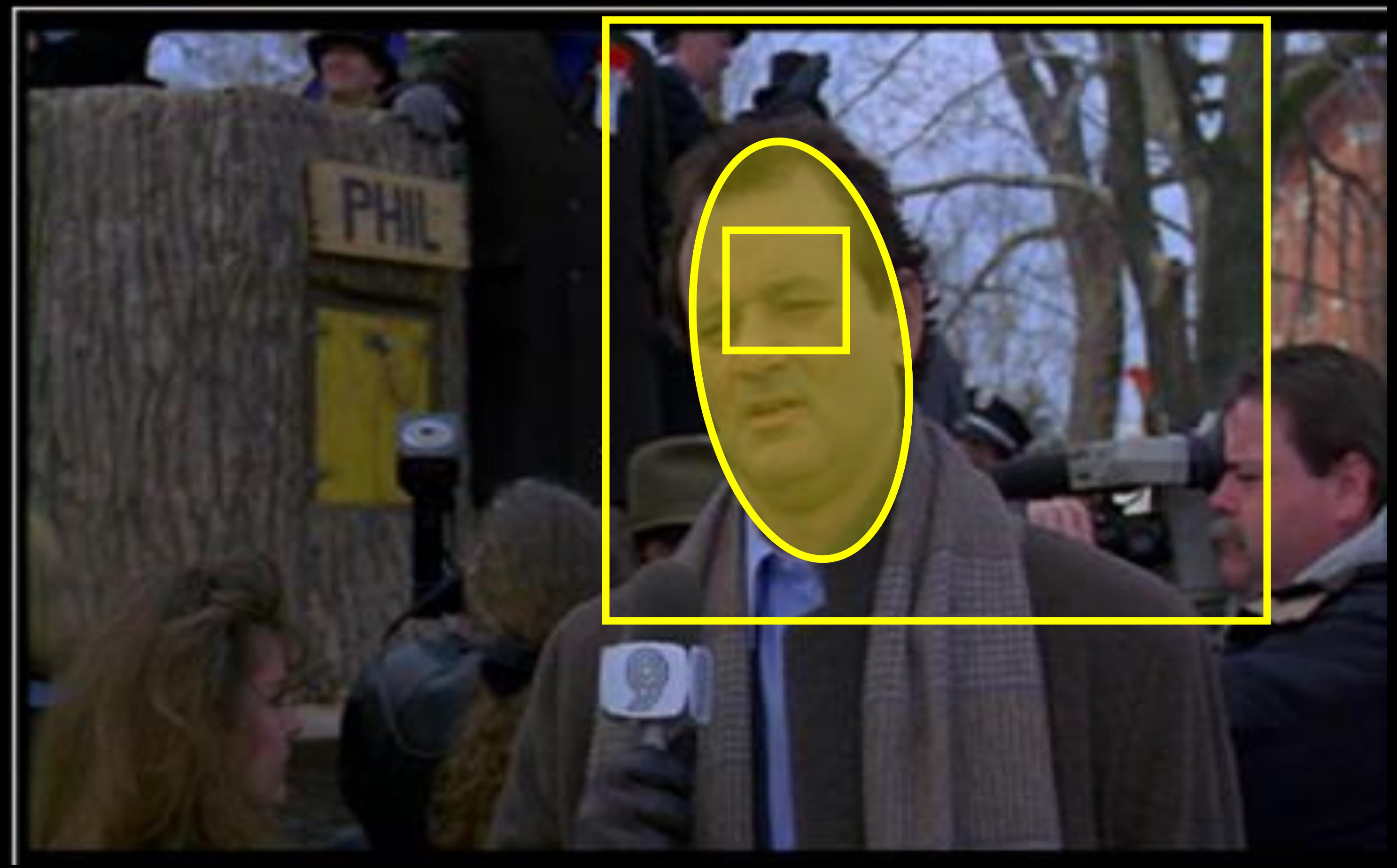


Discrimination of Basic Categories



Object Recognition using Texture

Spatial Support

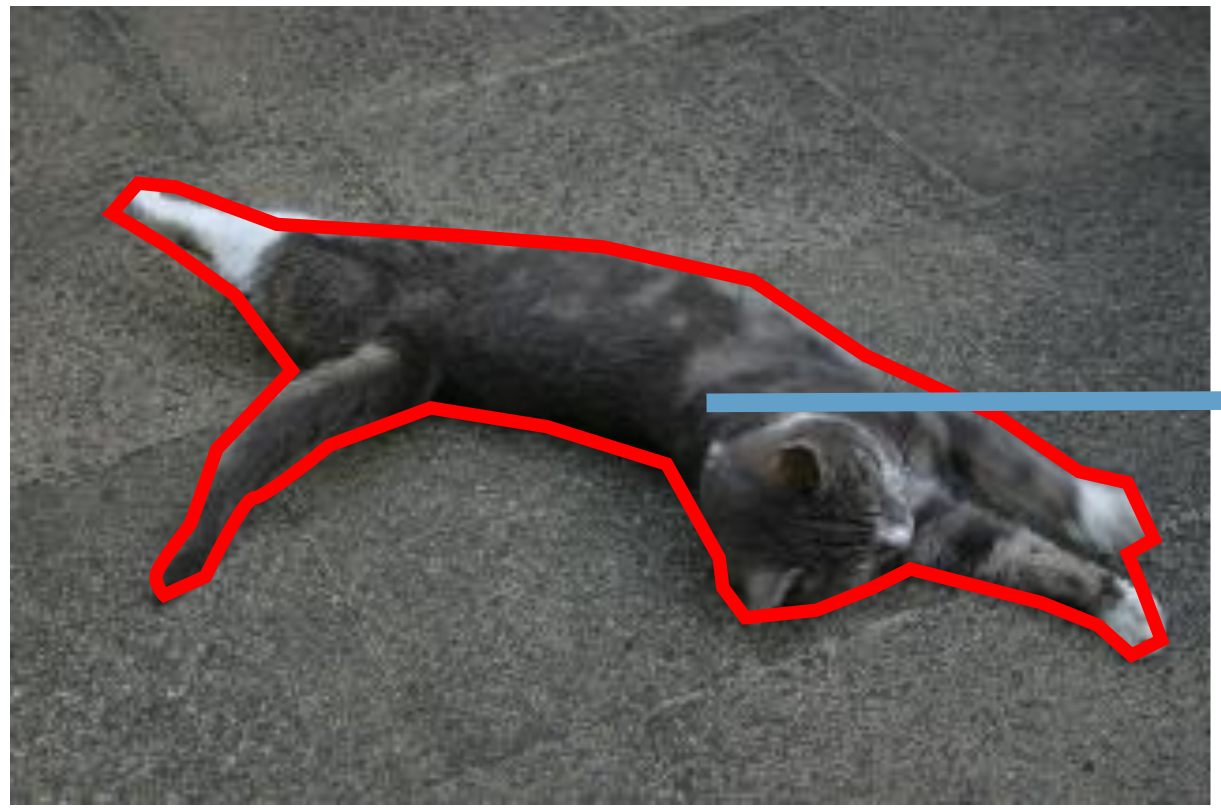


model

- Spatial Support
  - Which pixels to include?
- Similarity Metric
  - Which statistics to compute?

<u>Conjecture:</u> second will get easier, if we make progress on the first

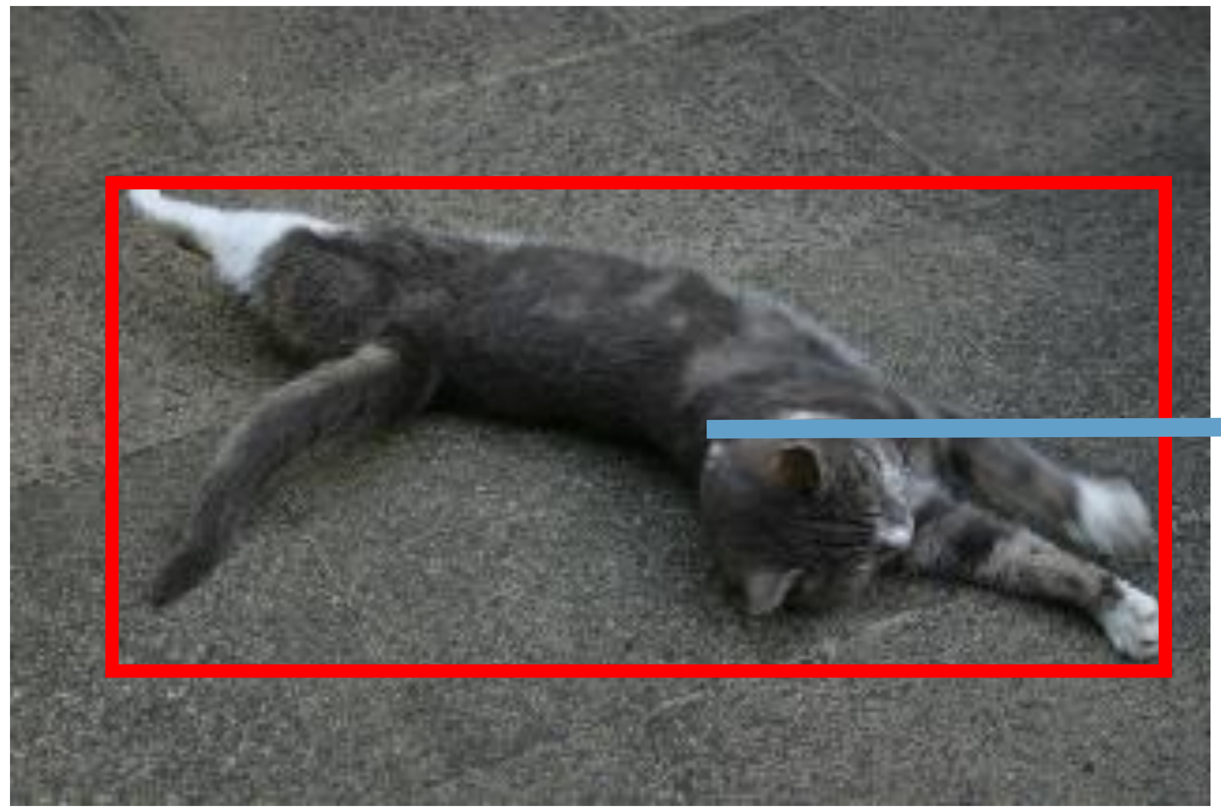
Does Spatial Support Matter?



Classify

Ground-Truth Segment

vs.

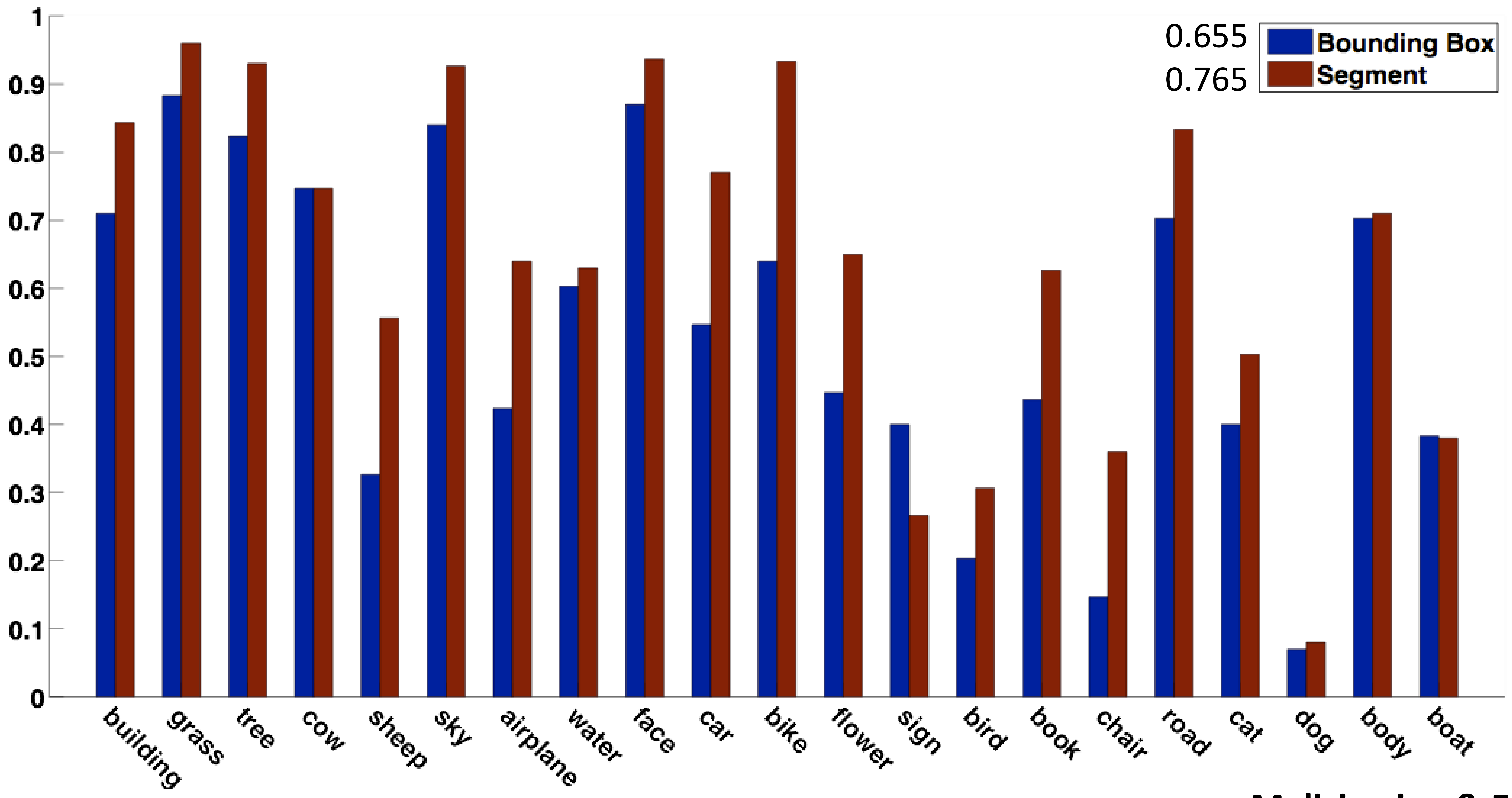


Classify

Bounding Box

Malisiewicz & Efros, BMVC 2007

Does Spatial Support Matter?



Malisiewicz & Efros, BMVC 2007

Texture-motivated Parametric Generative Image Models!

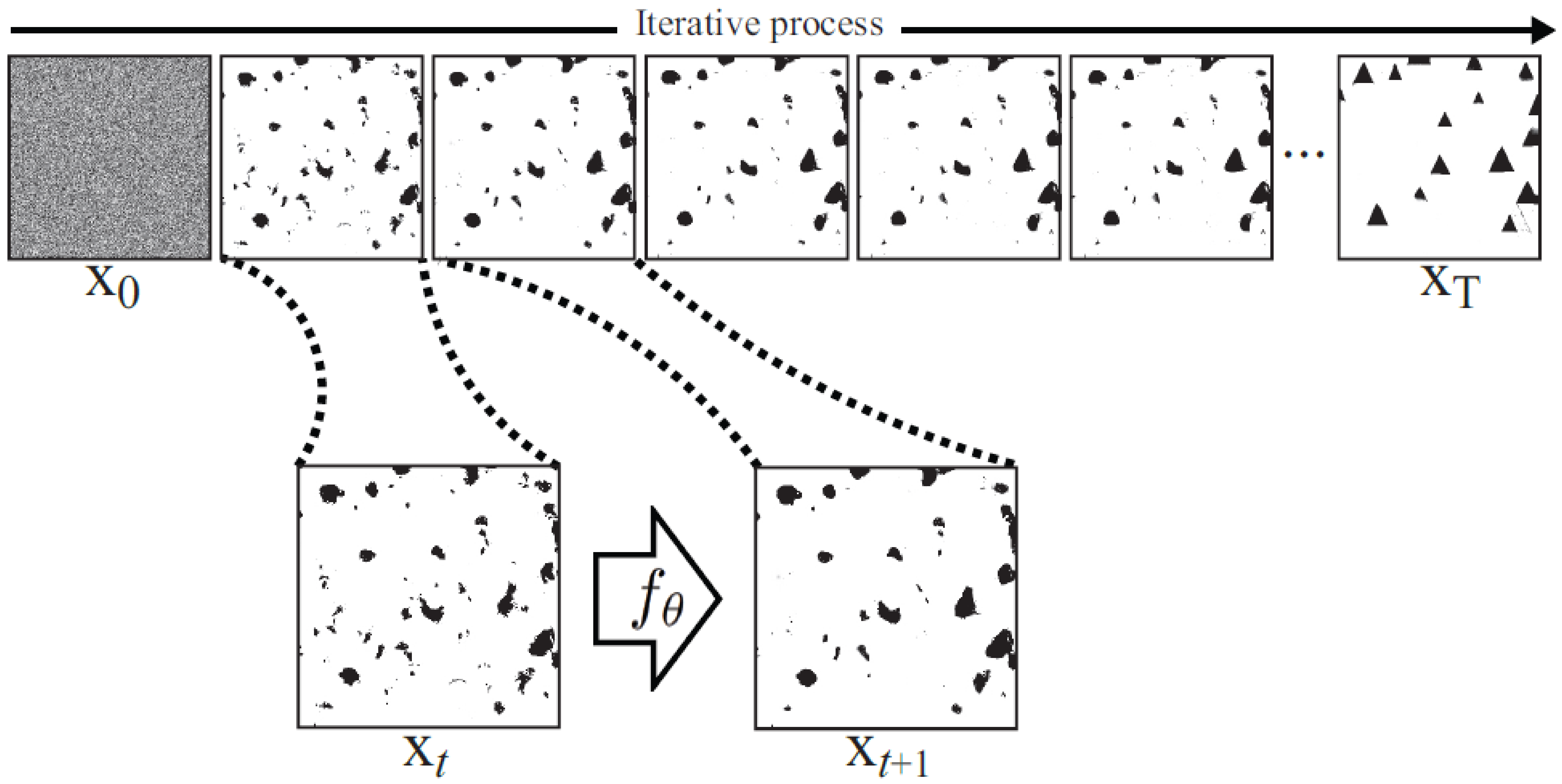


Figure 28.6: The steps of the Heeger-Bergen texture synthesis algorithm. The process starts with white noise input image. Each step takes as input the previous output, and it is modified by a function f_θ, where θ are the parameters describing a texture. At each step the output image x_t gets closer to the appearance of the reference texture (figure 28.5). The result of 500 iterations is shown in the right.

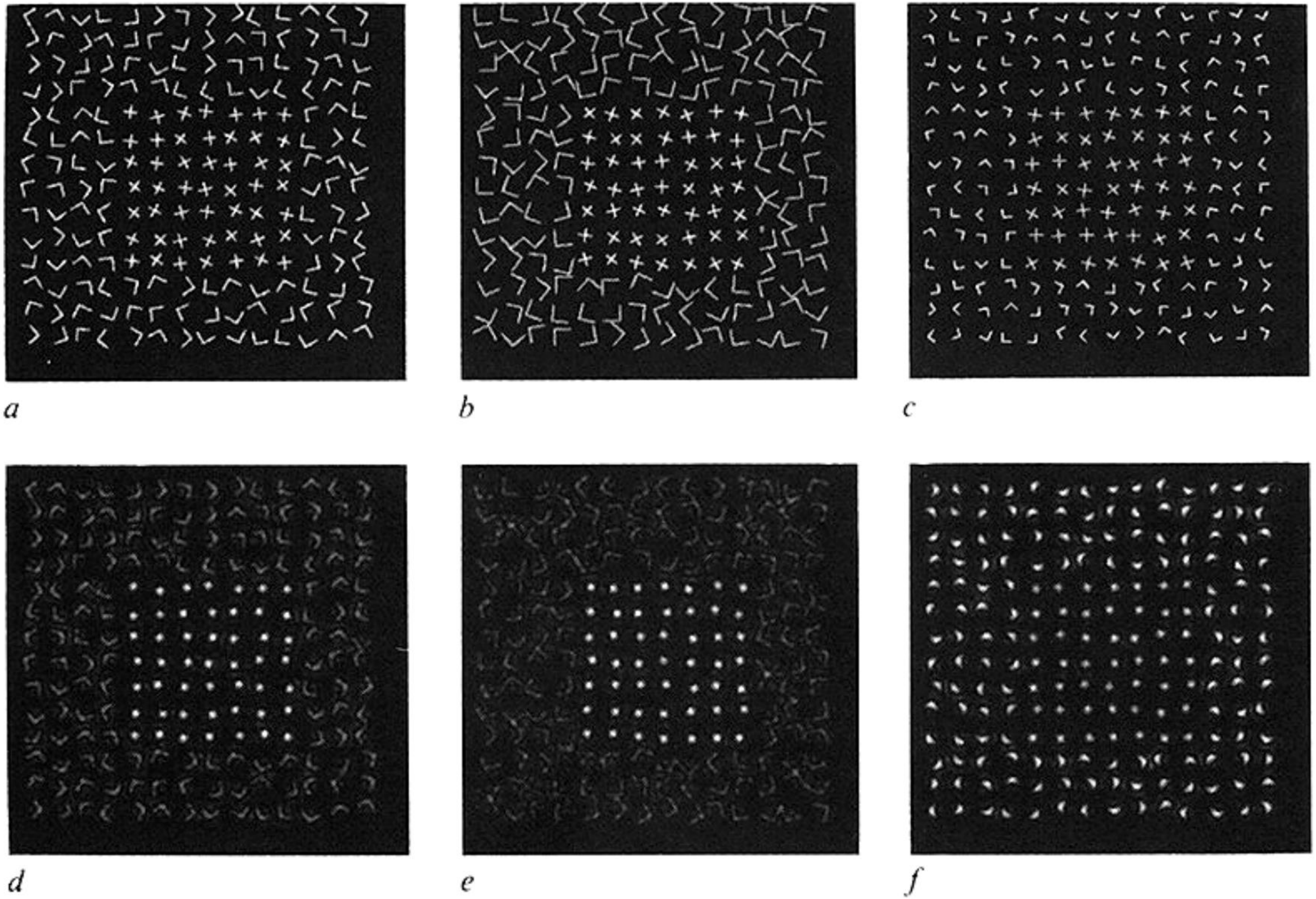


Fig. 1 *Top row*, Textures consisting of Xs within a texture composed of Ls. The micropatterns are placed at random orientations on a randomly perturbed lattice. *a*, The bars of the Xs have the same length as the bars of the Ls. *b*, The bars of the Ls have been lengthened by 25%, and the intensity 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*.

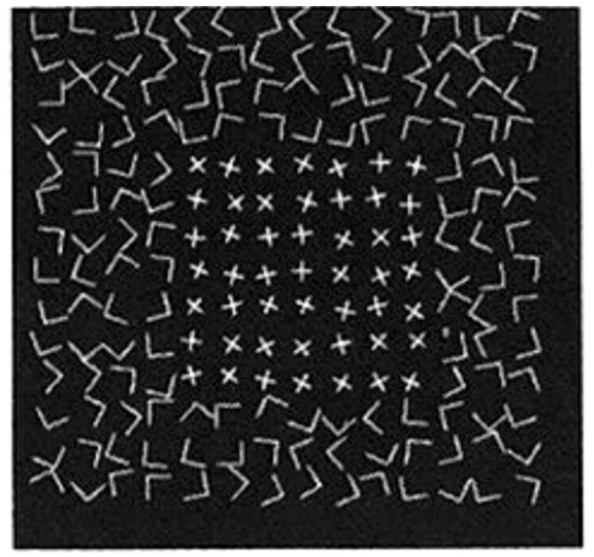
Early vision and texture perception

James R. Bergen* & Edward H. Adelson**

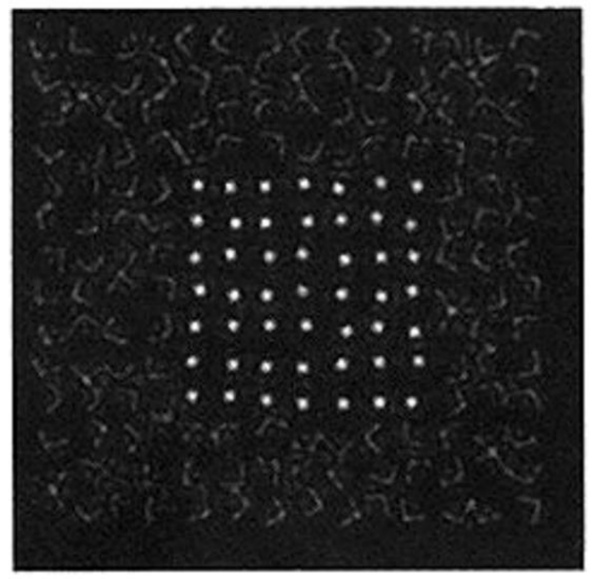
* SRI David Sarnoff Research Center, Princeton, New Jersey 08540, USA
** Media Lab and Department of Brain and Cognitive Science, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA

Reprinted from Nature, Vol. 333. No. 6171. pp. 363-364, 26 May 1988

Jim Bergen's conjecture



b



e

"If matching the mean amplitude of a bandpass spatial filter's response goes a little way towards mimicking human texture perception, then maybe matching the histogram of the responses of filters (the marginal statistics) will do an even better job of capturing human texture perception."

Dave Heeger's response

"You mean if I took a steerable pyramid of some random noise, and forced the histograms of each subband level to match those of some target texture, that the modified noise image would then look like that texture??? No way! I'll prove it to you; here, let me try it out…   hmm, gee, that worked pretty well…"

Pyramid-Based Texture Analysis/Synthesis

David J. Heeger*
Stanford University

James R. Bergen†
SRI David Sarnoff Research Center

Abstract

This paper describes a method for synthesizing images that match the texture appearance of a given digitized sample. This synthesis is completely automatic and requires only the "target" texture as input. It allows generation of as much texture as desired so that any object can be covered. It can be used to produce solid textures for creating textured 3-d objects without the distortions inherent in texture mapping. It can also be used to synthesize texture mixtures, images that look a bit like each of several digitized samples. The approach is based on a model of human texture perception, and has potential to be a practically useful tool for graphics applications.

1 Introduction

Computer renderings of objects with surface texture are more interesting and realistic than those without texture. Texture mapping [15] is a technique for adding the appearance of surface detail by wrapping or projecting a digitized texture image onto a surface. Digitized textures can be obtained from a variety of sources, e.g., cropped from a photoCD image, but the resulting texture chip may not have the desired size or shape. To cover a large object you may need to repeat the texture; this can lead to unacceptable artifacts either in the form of visible seams, visible repetition, or both.

Texture mapping suffers from an additional fundamental problem: often there is no natural map from the (planar) texture image to the geometry/topology of the surface, so the texture may be distorted unnaturally when mapped. There are some partial solutions to this distortion problem [15] but there is no universal solution for mapping an image onto an arbitrarily shaped surface.

An alternative to texture mapping is to create (paint) textures by hand directly onto the 3-d surface model [14], but this process is both very labor intensive and requires considerable artistic skill.

Another alternative is to use computer-synthesized textures so that as much texture can be generated as needed. Furthermore, some of the synthesis techniques produce textures that tile seamlessly.

Using synthetic textures, the distortion problem has been solved in two different ways. First, some techniques work by synthesizing texture directly on the object surface (e.g., [31]). The second solution is to use *solid textures* [19, 23, 24]. A solid texture is a 3-d array of color values. A point on the surface of an object is colored by the value of the solid texture at the corresponding 3-d point. Solid texturing can be a very natural solution to the distortion problem: there is no distortion because there is no mapping. However, existing techniques for synthesizing solid textures can be quite cumbersome. One must learn how to tweak the parameters or procedures of the texture synthesizer to get a desired effect.

This paper presents a technique for synthesizing an image (or solid texture) that matches the appearance of a given texture sample. The key advantage of this technique is that it works entirely from the example texture, requiring no additional information or adjustment. The technique starts with a digitized image and analyzes it to compute a number of texture parameter values. Those parameter values are then used to synthesize a new image (of any size) that looks (in its color and texture properties) like the original. The analysis phase is inherently two-dimensional since the input digitized images are 2-d. The synthesis phase, however, may be either two- or three-dimensional. For the 3-d case, the output is a solid texture such that planar slices through the solid look like the original scanned image. In either case, the (2-d or 3-d) texture is synthesized so that it tiles seamlessly.

2 Texture Models

Textures have often been classified into two categories, deterministic textures and stochastic textures. A deterministic texture is characterized by a set of primitives and a placement rule (e.g., a tile floor). A stochastic texture, on the other hand, does not have easily identifiable primitives (e.g., granite, bark, sand). Many real-world textures have some mixture of these two characteristics (e.g. woven fabric, woodgrain, plowed fields).

Much of the previous work on texture analysis and synthesis can be classified according to what type of texture model was used. Some of the successful texture models include reaction-diffusion [31, 34], frequency domain [17], fractal [9, 18], and statistical/random field [1, 6, 8, 10, 12, 13, 21, 26] models. Some (e.g., [10]) have used hybrid models that include a deterministic (or periodic) component and a stochastic component. In spite of all this work, scanned images and hand-drawn textures are still the principle source of texture maps in computer graphics.

This paper focuses on the synthesis of stochastic textures. Our approach is motivated by research on human texture perception. Current theories of texture discrimination are based on the fact that two textures are often difficult to discriminate when they produce a similar distribution of responses in a bank of (orientation and spatial-frequency selective) linear filters [2, 3, 7, 16, 20, 32]. The method described here, therefore, synthesizes textures by matching distributions (or histograms) of filter outputs. This approach depends on the principle (not entirely correct as we shall see) that all of the spatial information characterizing a texture image can be captured in the first order statistics of an appropriately chosen set of linear filter outputs. Nevertheless, this model (though incomplete) captures an interesting set of texture properties.

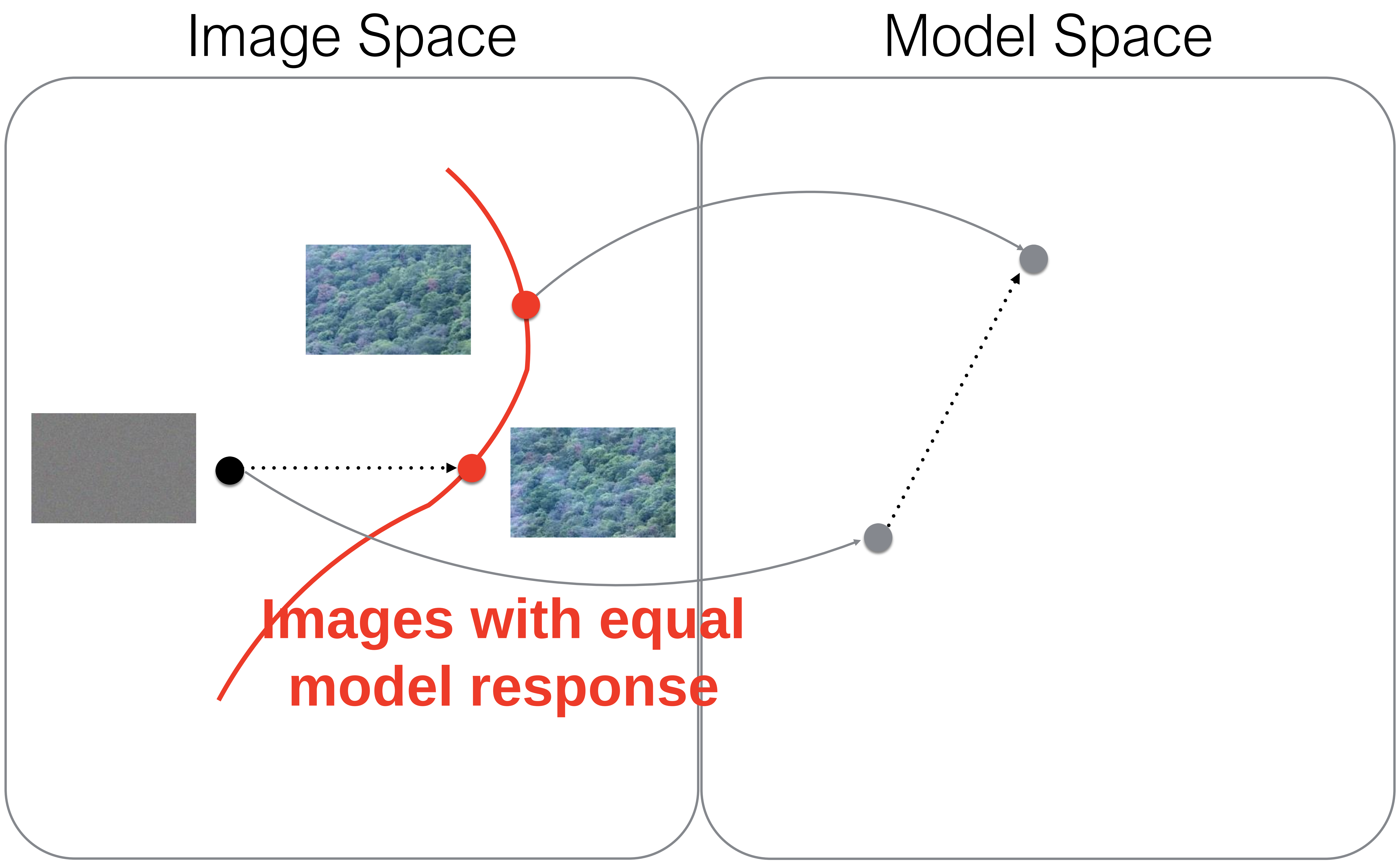
*Department of Psychology, Stanford University, Stanford, CA 94305. heeger@white.stanford.edu http://white.stanford.edu
†SRI David Sarnoff Research Center, Princeton, NJ 08544. jrb@sarnoff.com

Figure 5: (Top Row) Original digitized sample textures: red granite, berry bush, figured maple, yellow coral. (Bottom Rows) Synthetic solid textured teapots.

https://www.cns.nyu.edu/heegerlab/content/publications/Heeger-siggraph95.pdf

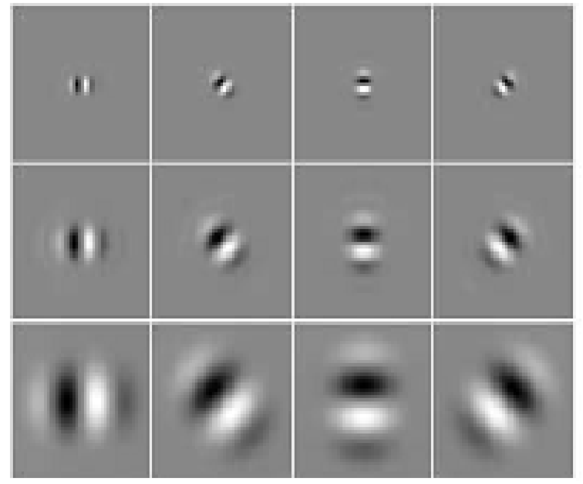
SIGGRAPH 1995

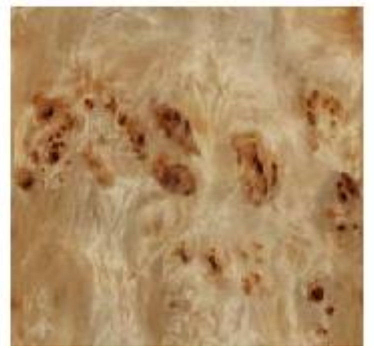
Parametric Texture Synthesis



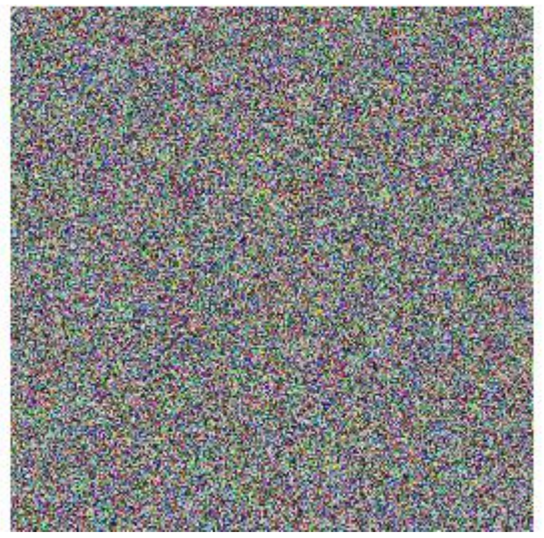
Portilla & Simoncelli (2000)

Step 1: Convolve with filterbank

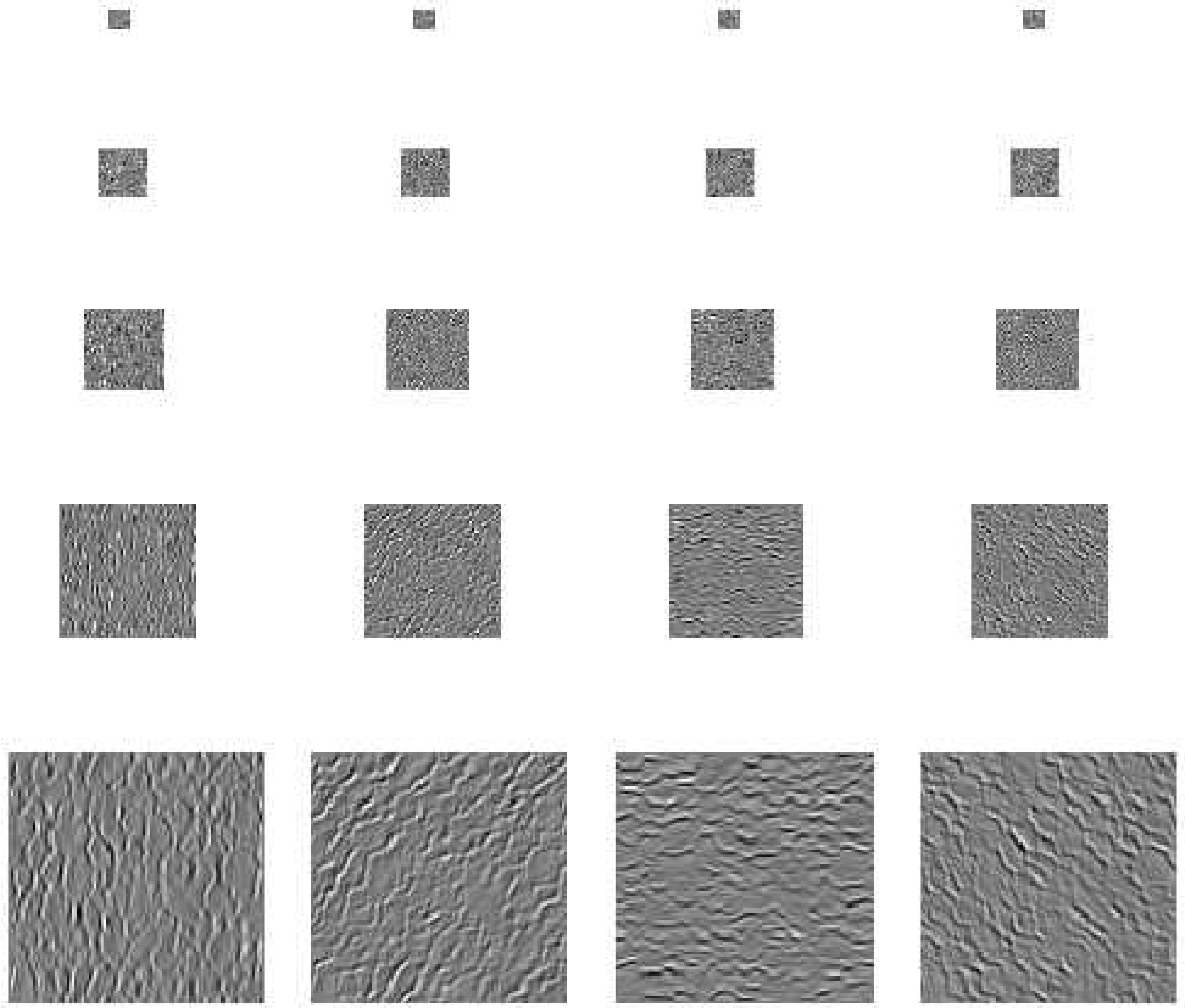


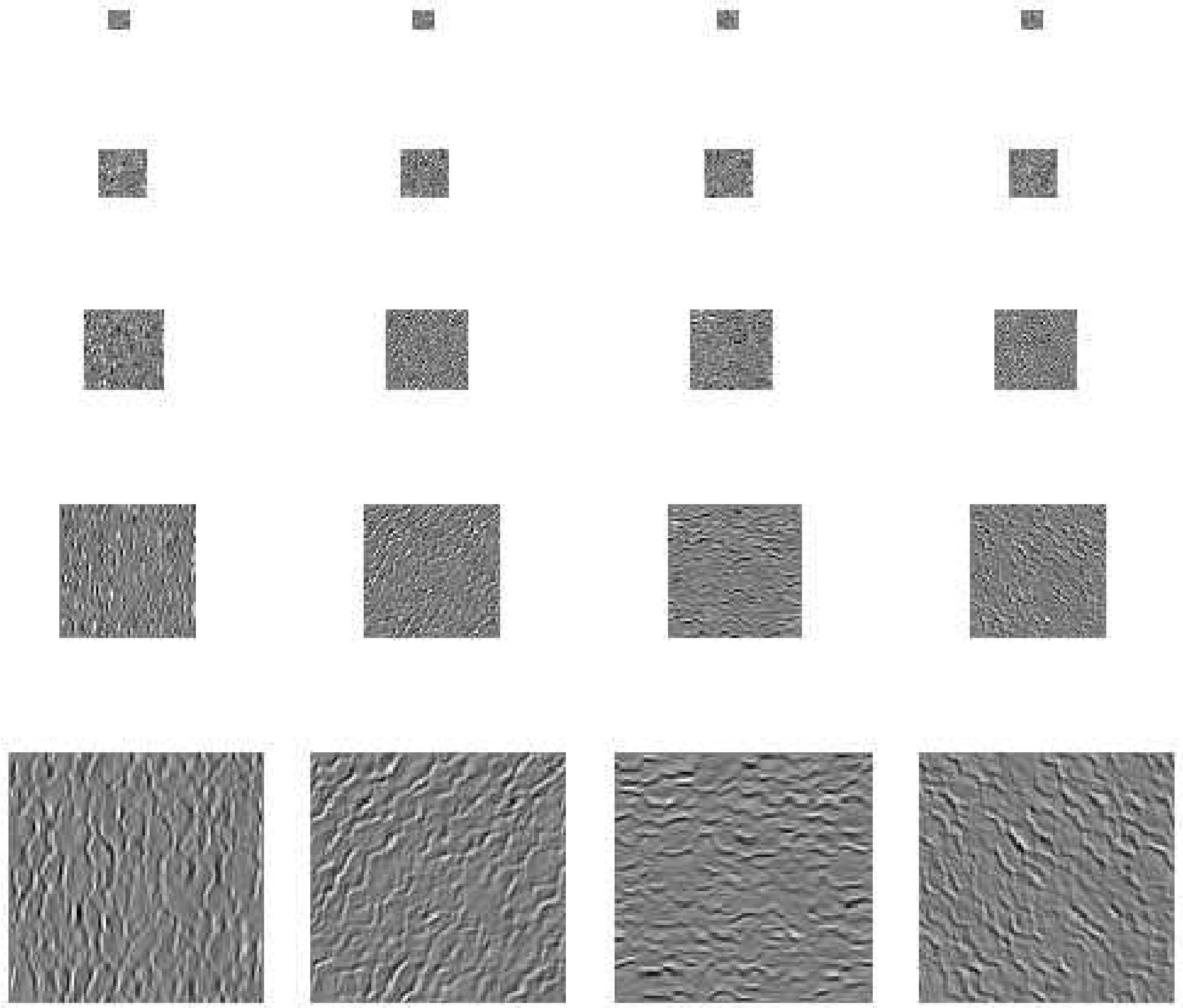


input

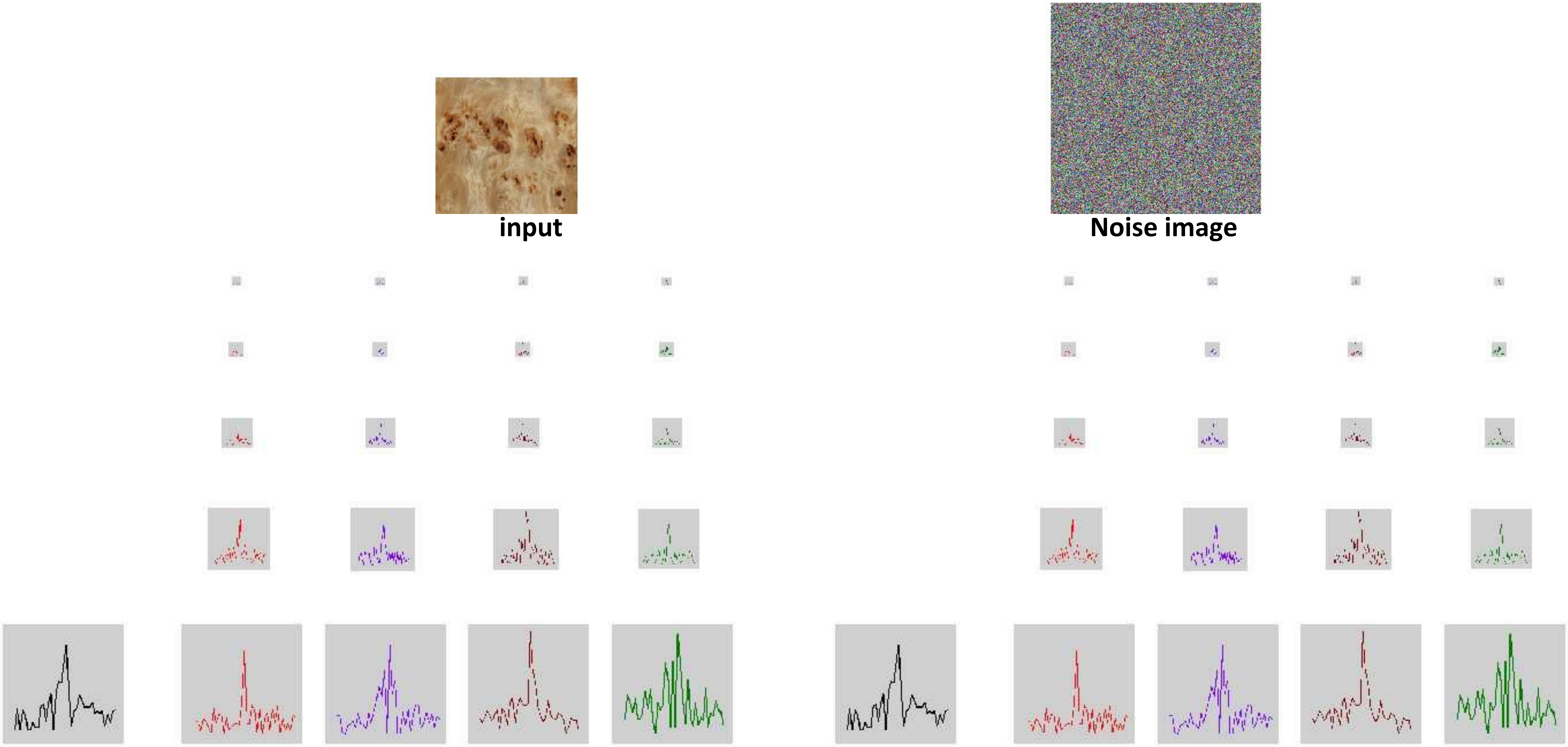


Noise image

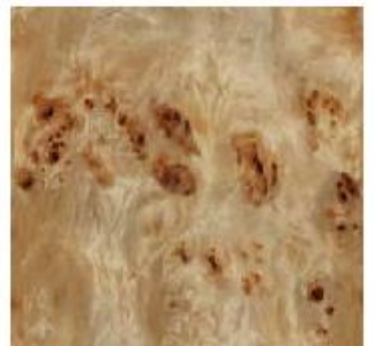




Step 2: match per--channel histograms



Step 3: collapse pyramid and repeat!



input

Noise image

Overview of the algorithm

```
Match-texture(noise,texture)
  Match-Histogram (noise,texture)
  analysis-pyr = Make-Pyramid (texture)
  Loop for several iterations do
     synthesis-pyr = Make-Pyramid (noise)
     Loop for a-band in subbands of analysis-pyr
          for s-band in subbands of synthesis-pyr
          do
          Match-Histogram (s-band,a-band)
     noise = Collapse-Pyramid (synthesis-pyr)
     Match-Histogram (noise,texture)
```

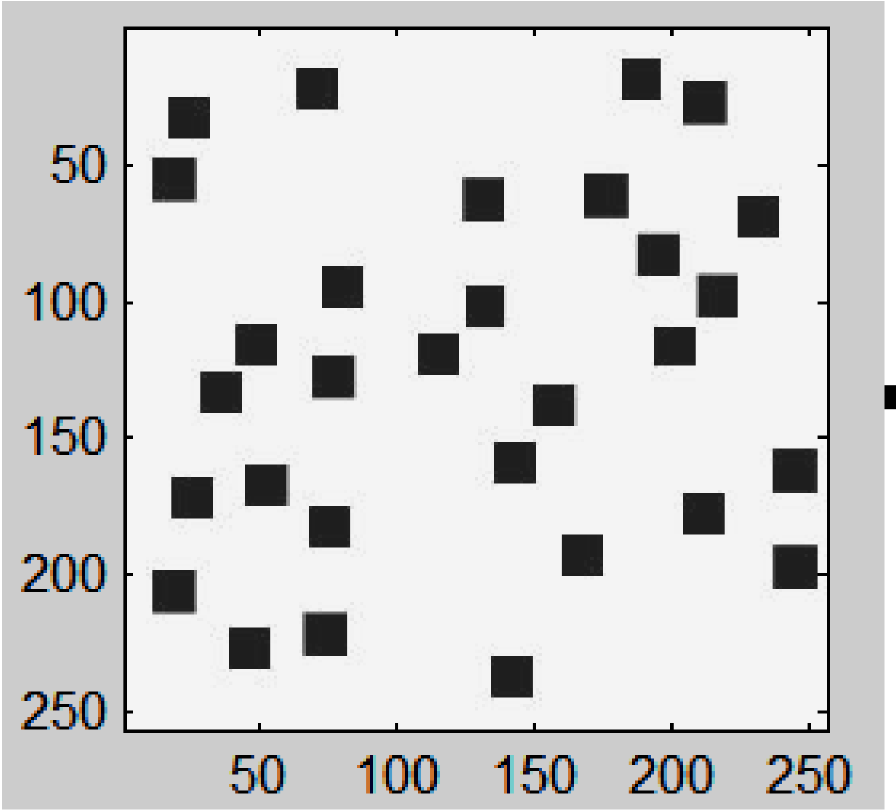
Two main tools:

1- steerable pyramid

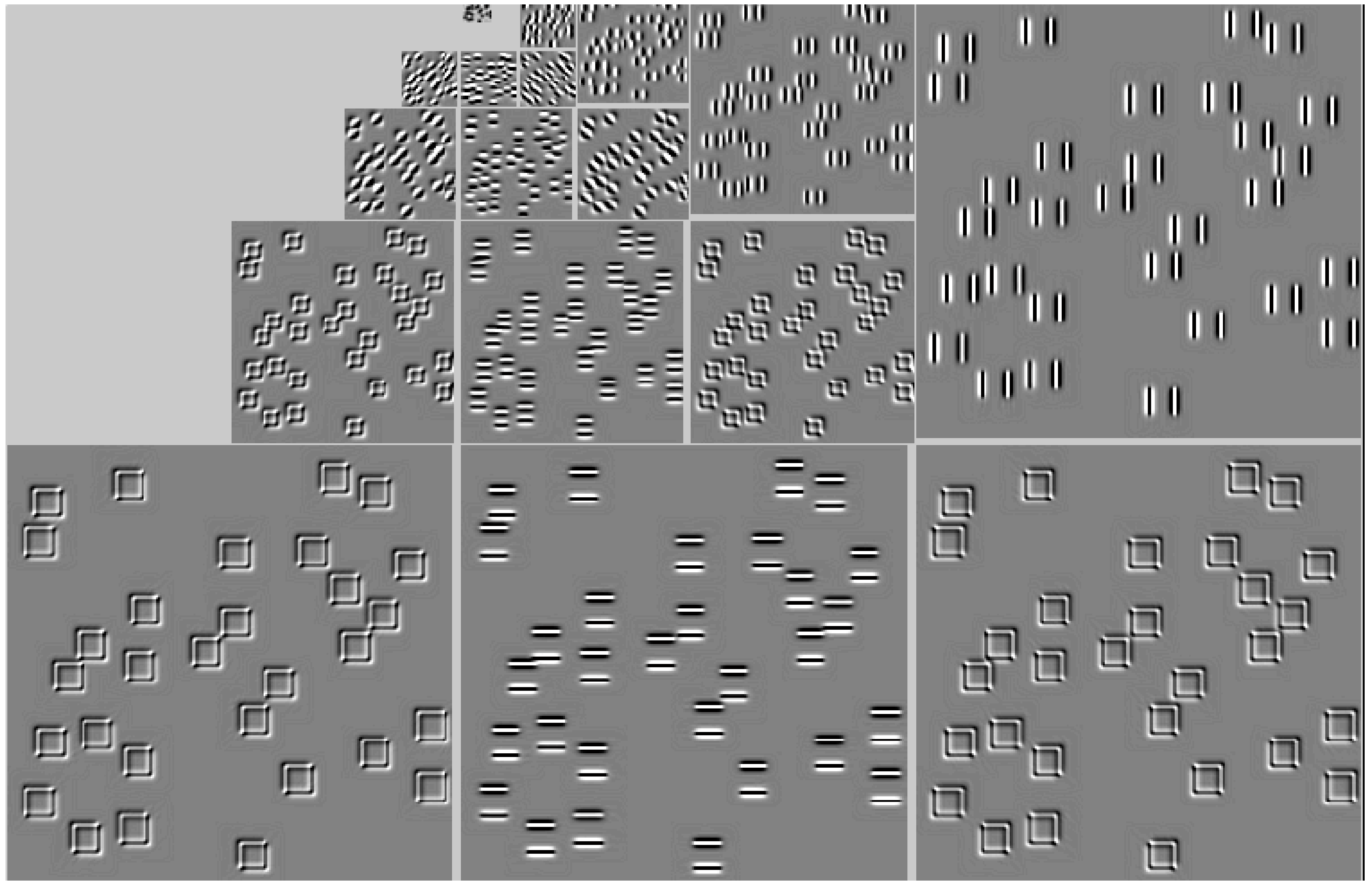
2- matching histograms

1-The steerable pyramid

Input texture



Steerable pyr



Steerable Pyramid

Overview of the algorithm

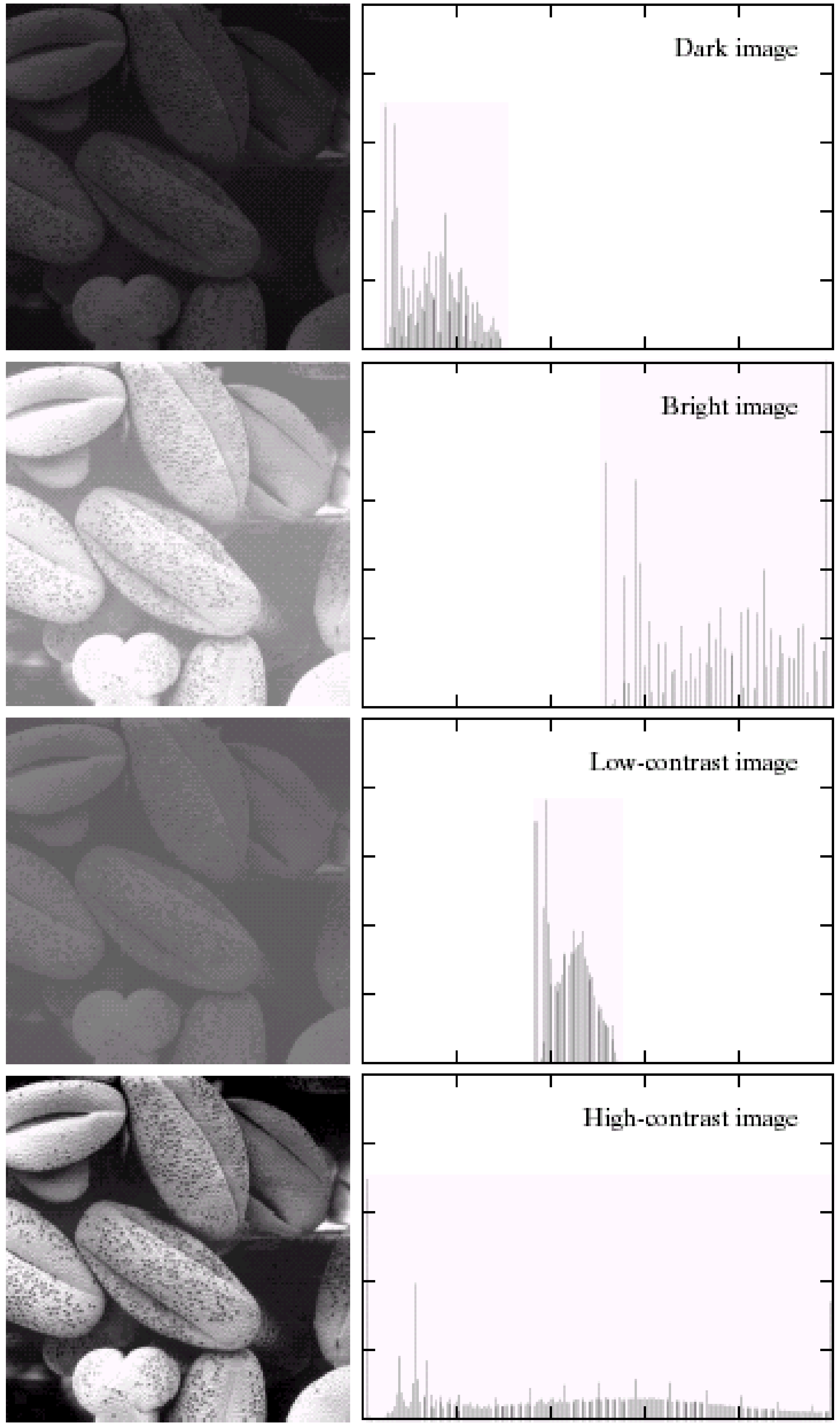
```
Match-texture(noise,texture)
  Match-Histogram (noise,texture)
  analysis-pyr = Make-Pyramid (texture)
  Loop for several iterations do
     synthesis-pyr = Make-Pyramid (noise)
     Loop for a-band in subbands of analysis-pyr
          for s-band in subbands of synthesis-pyr
          do
          Match-Histogram (s-band,a-band)
     noise = Collapse-Pyramid (synthesis-pyr)
     Match-Histogram (noise,texture)
```

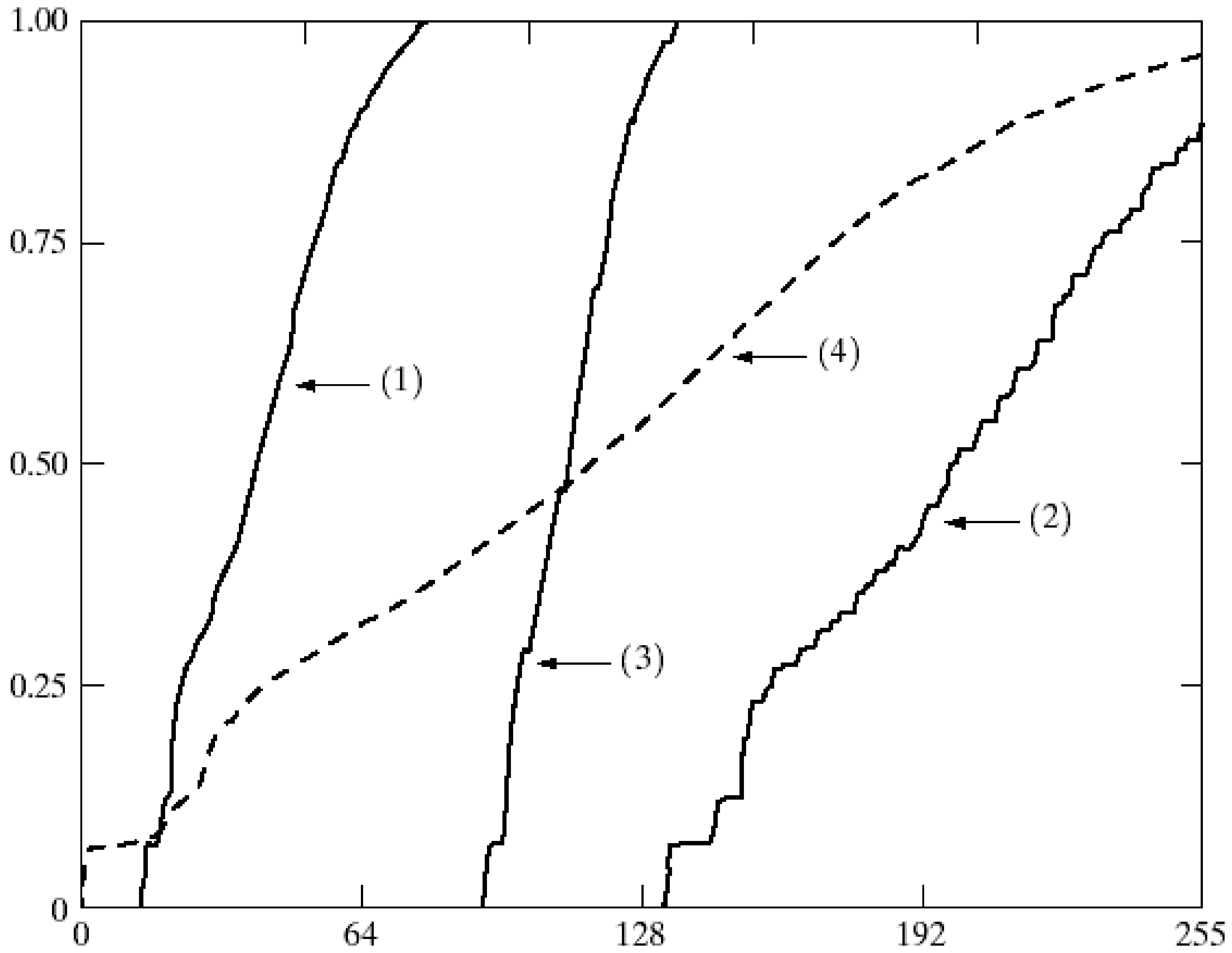
Two main tools:

1- steerable pyramid

2- matching histograms

Pixel Histograms





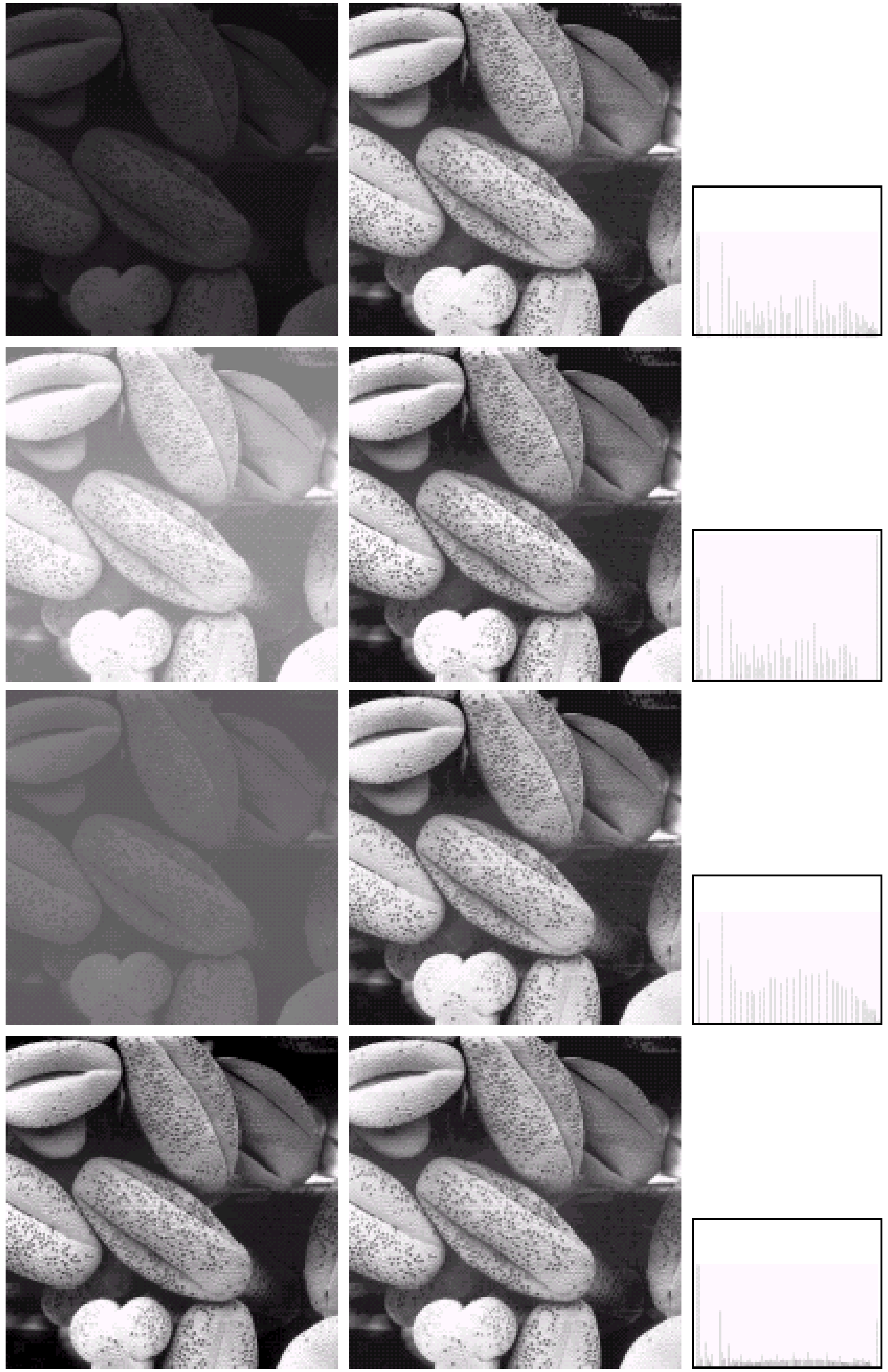
Cumulative Histograms

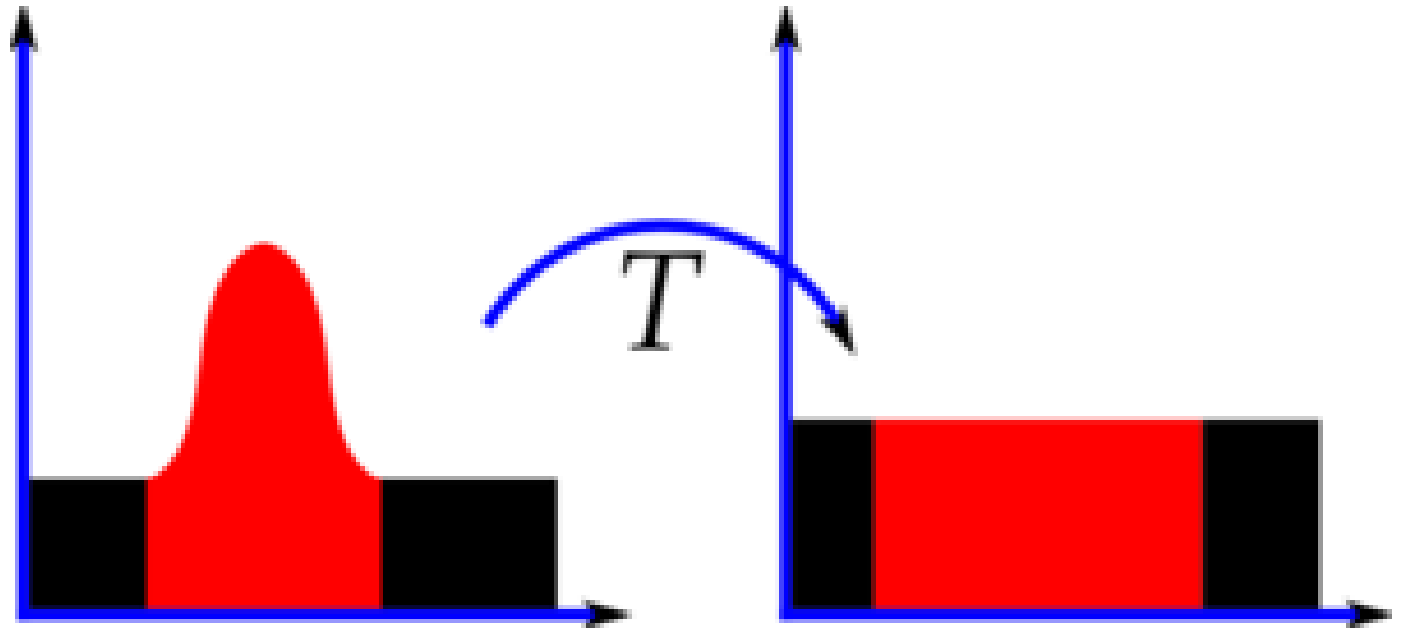
$$T(f(x,y))$$

a b

FIGURE 3.15 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Histogram Equalization





credit @ Zefram

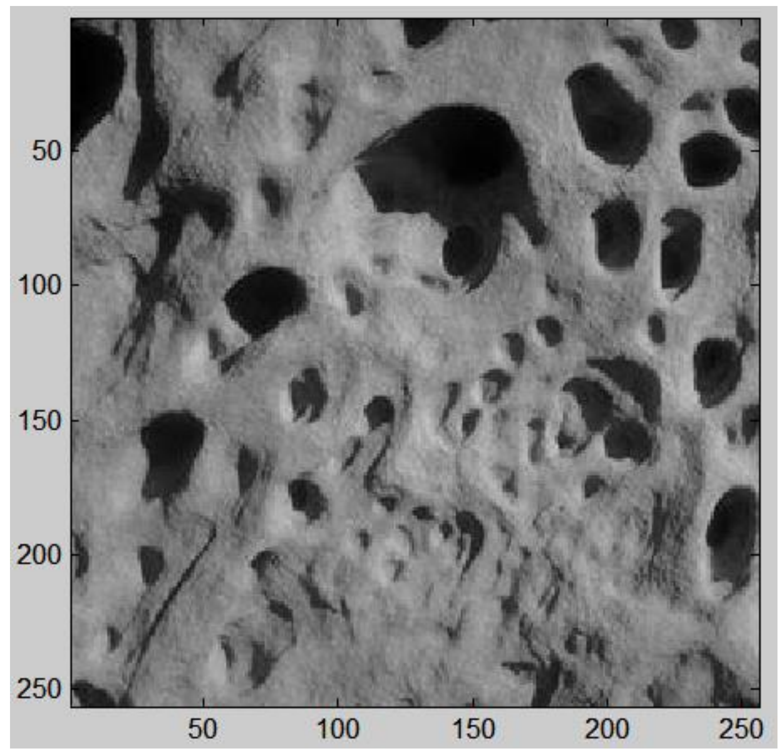
a b c

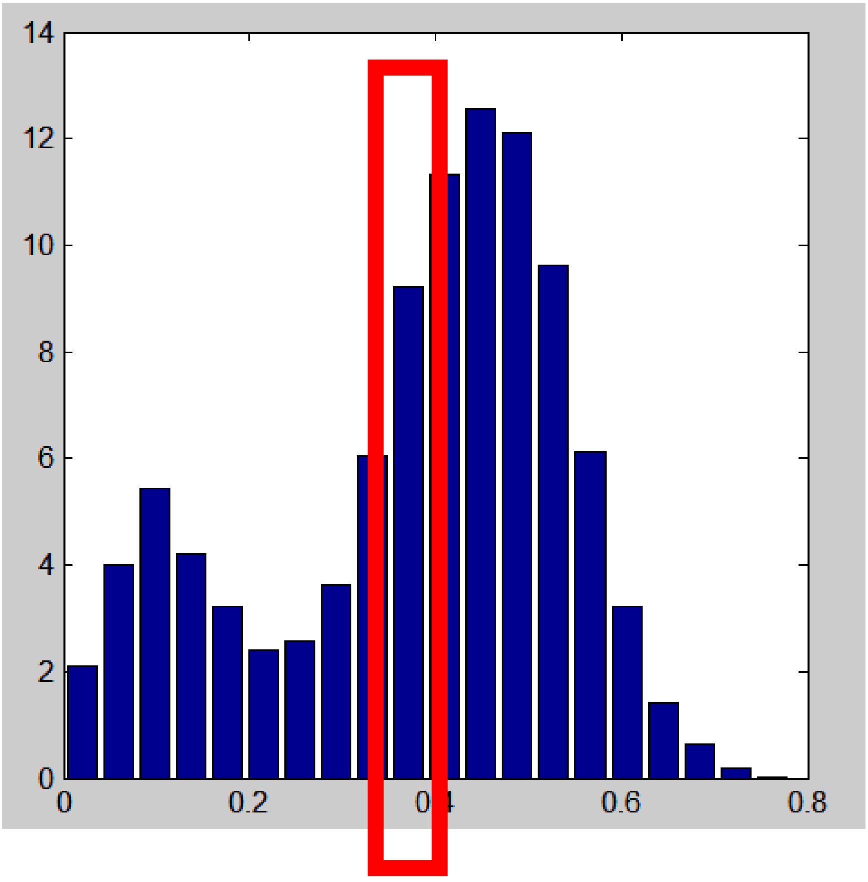
FIGURE 3.17 (a) Images from Fig. 3.15. (b) Results of histogram equalization. (c) Corresponding histograms.

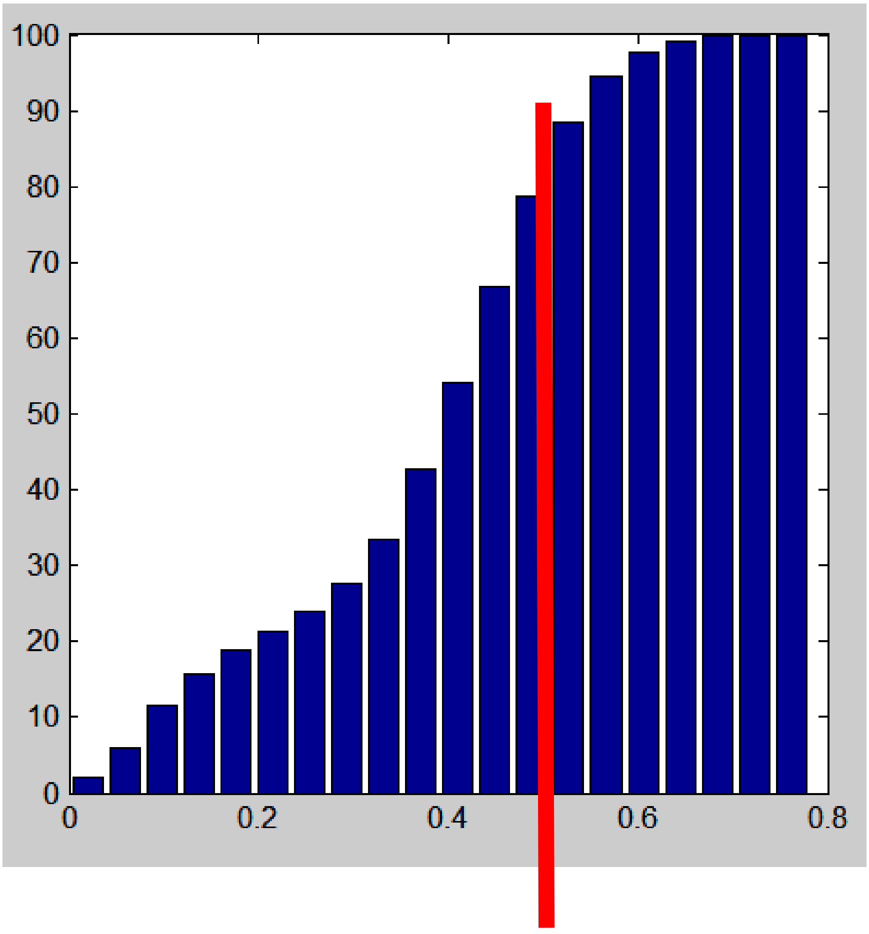
2-Matching histograms

Histograms

Cumulative Histograms

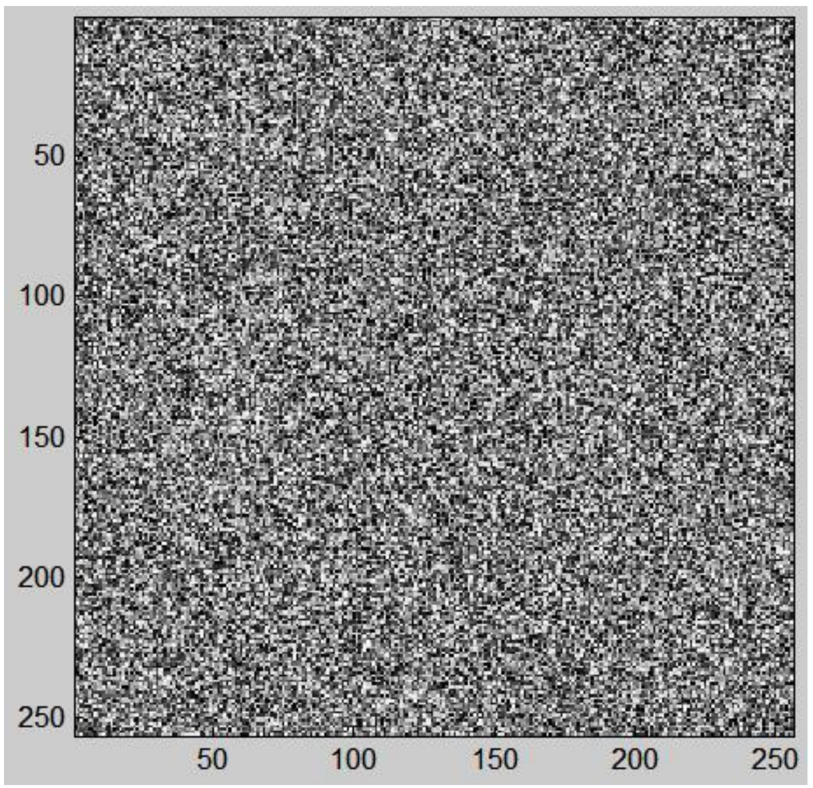


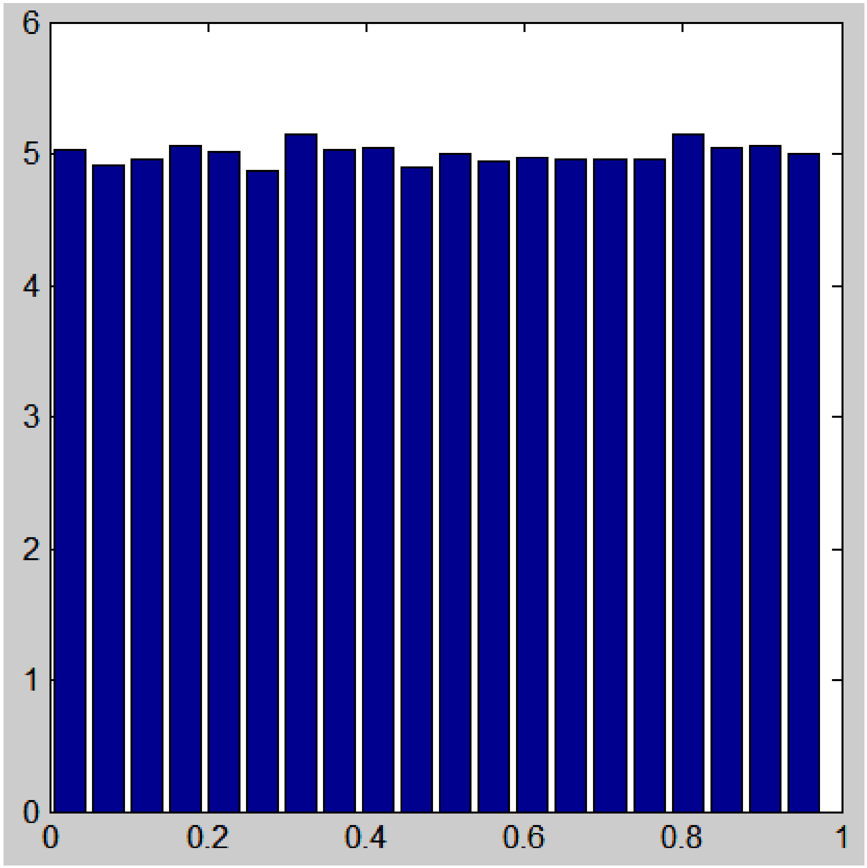


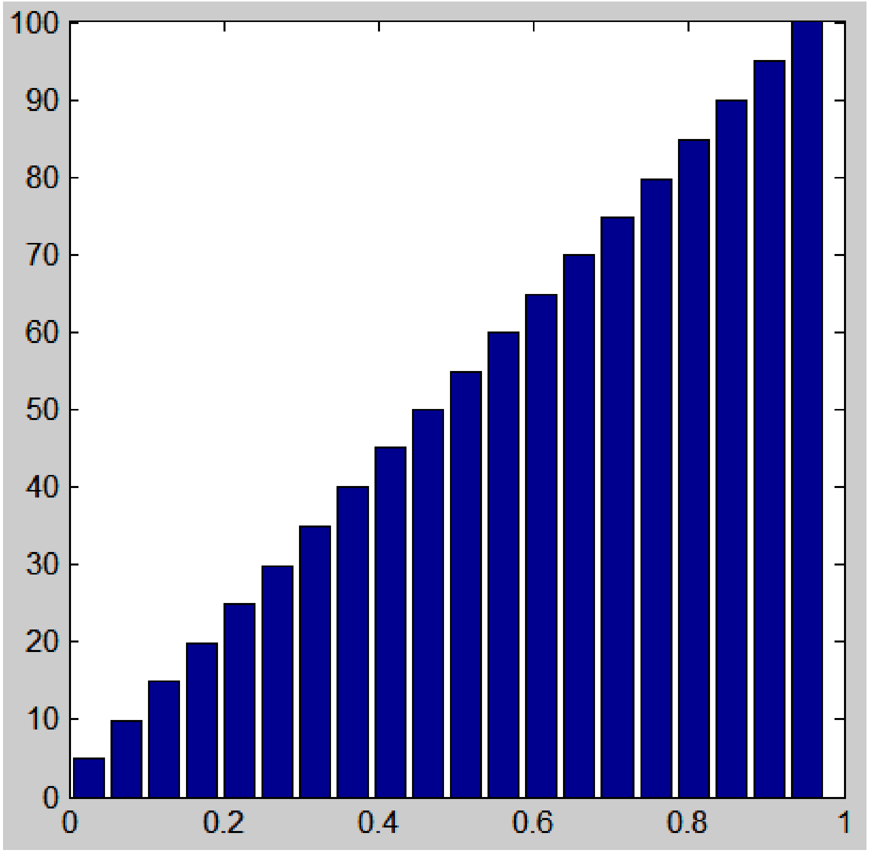


9% of pixels have an intensity value within the range[0.37, 0.41]

75% of pixels have an intensity value smaller than 0.5



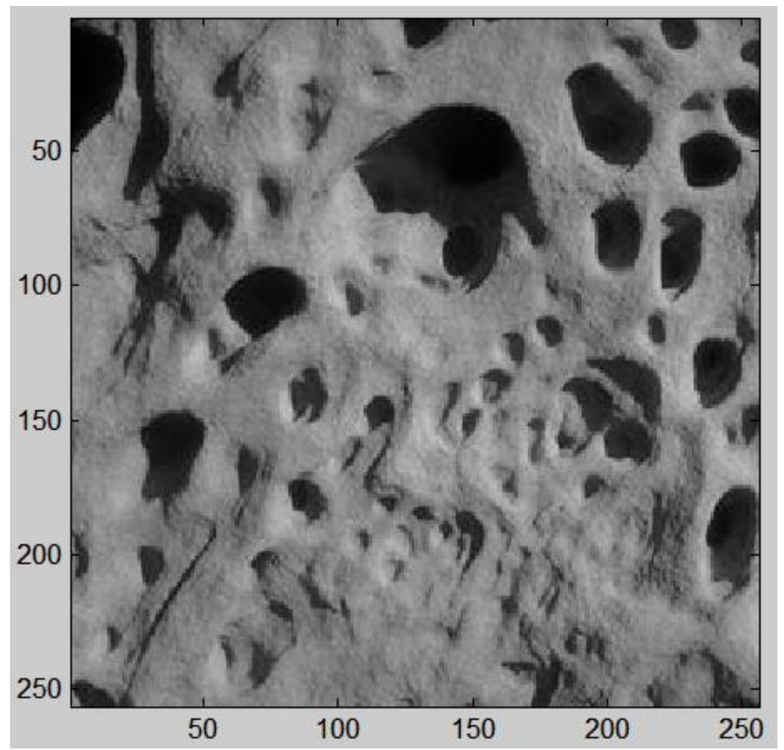


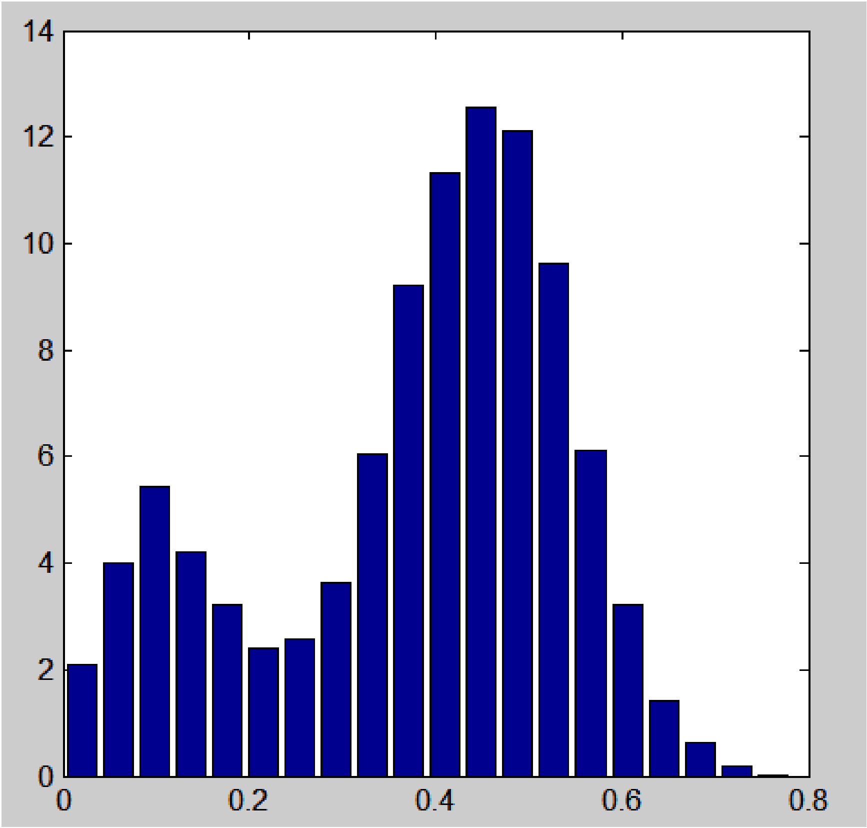


5% of pixels have an intensity value within the range[0.37, 0.41]

2-Matching histograms

Z(x,y)





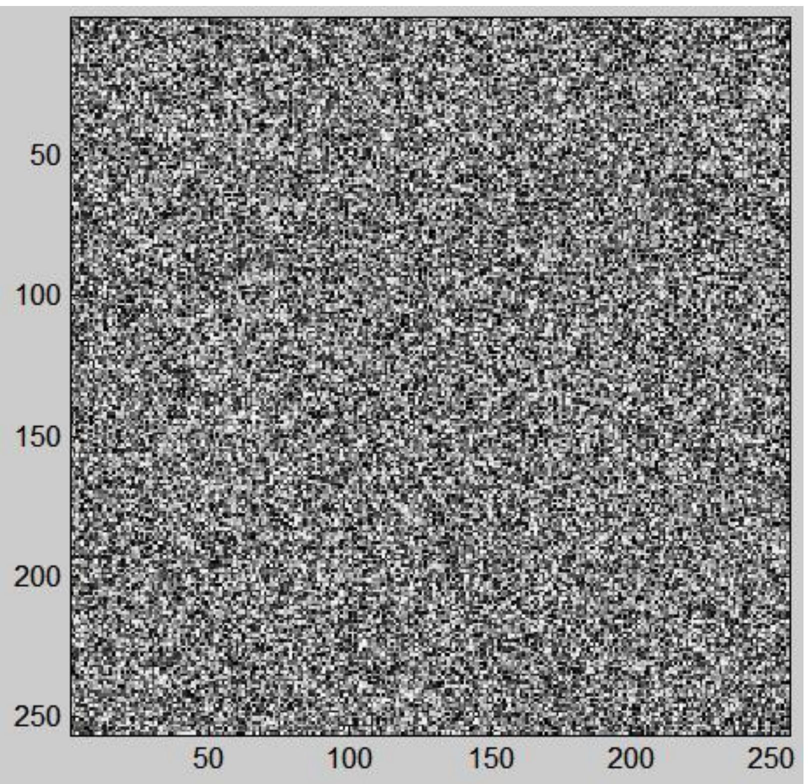
We look for a transformation of the image Y

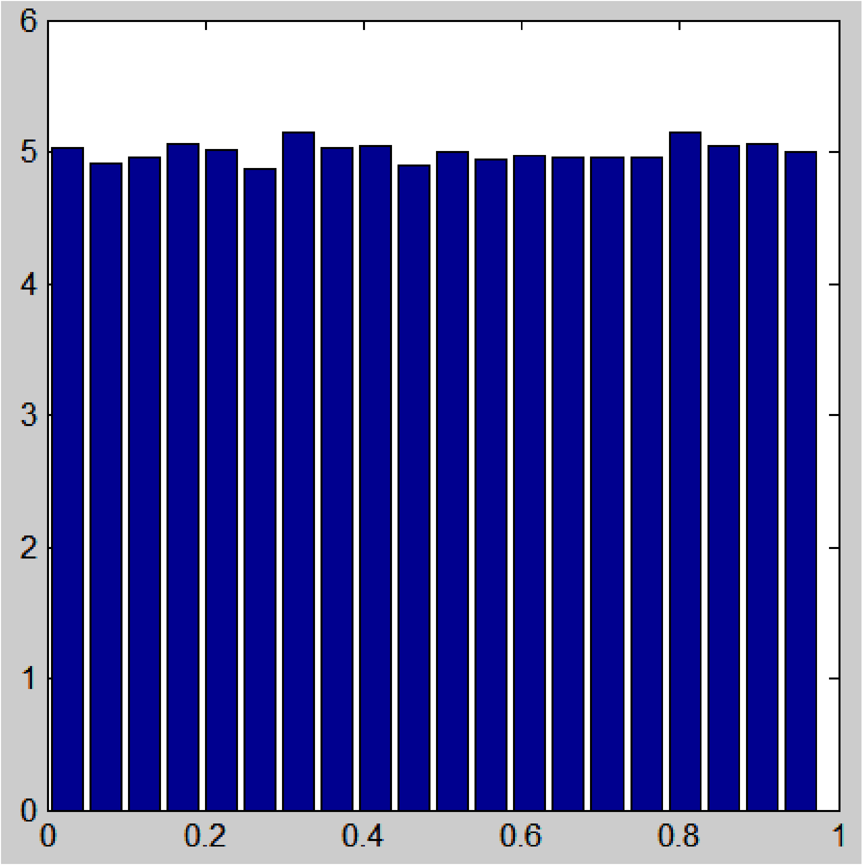
Y’ = f (Y)

Such that
Hist(Y) = Hist(f(Z))

?

Y(x,y)

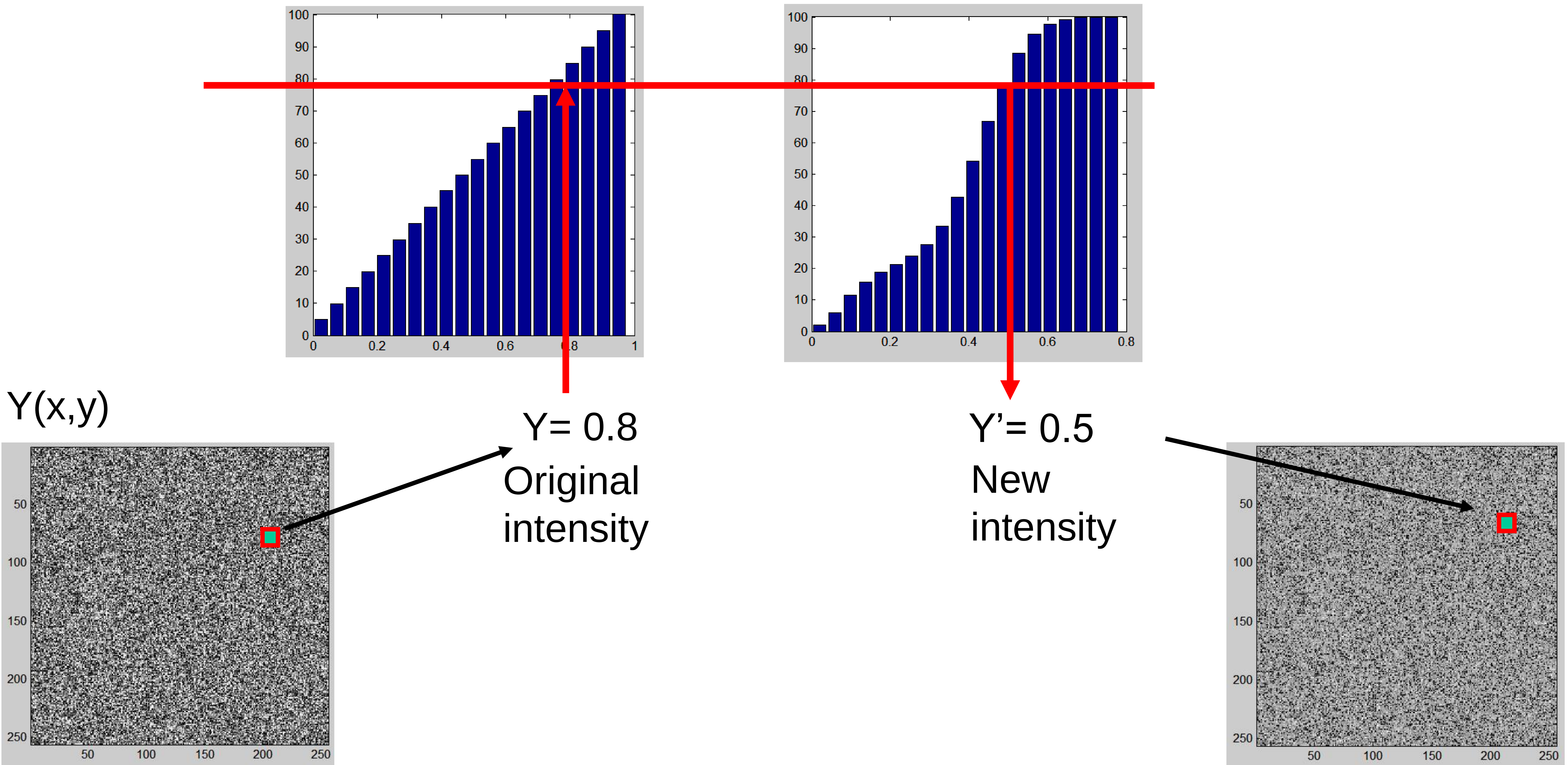




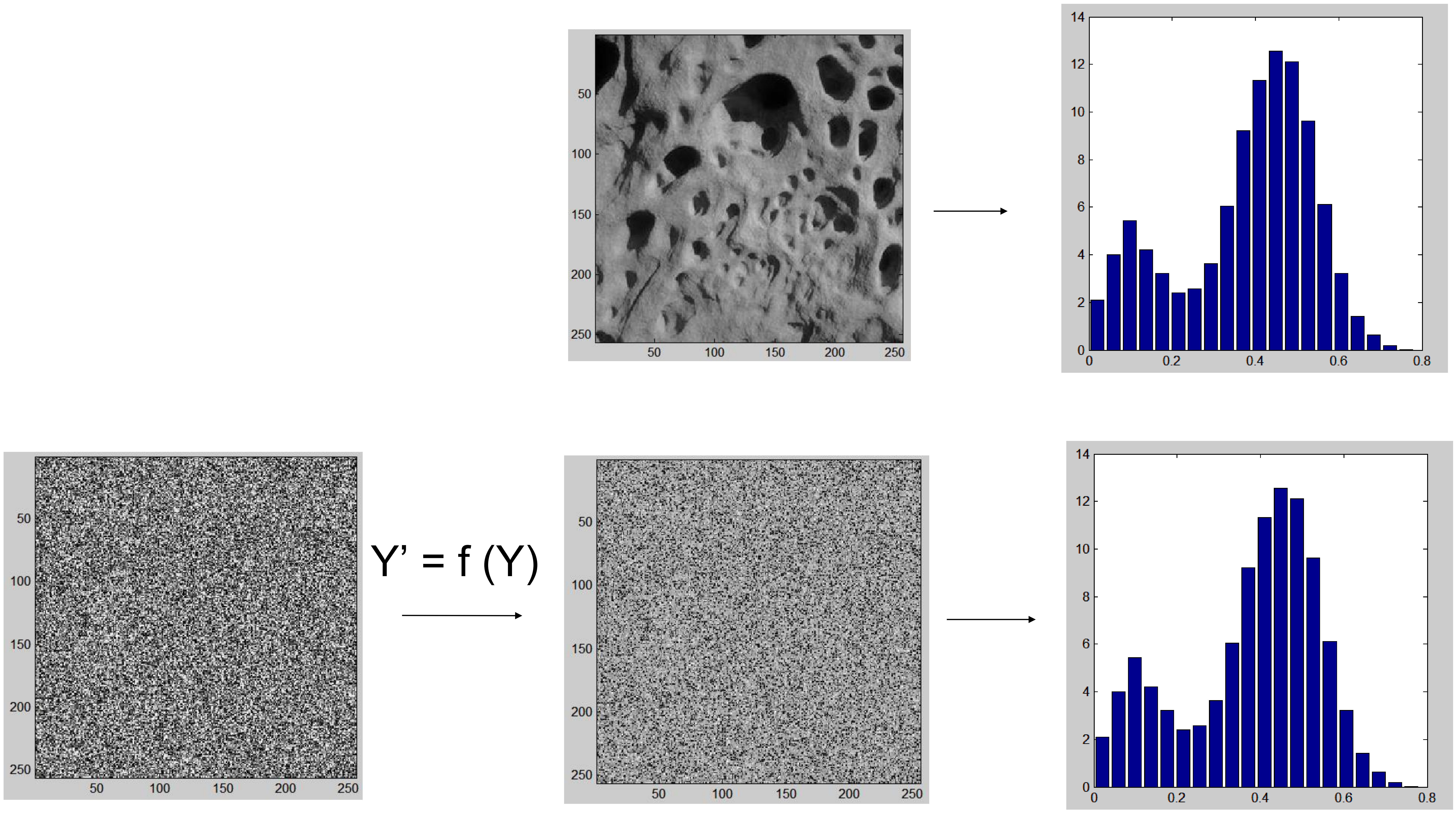
2-Matching cumulative histograms

The function f is just a look up table: it says, change all the pixels of value Y into a value f(Y).

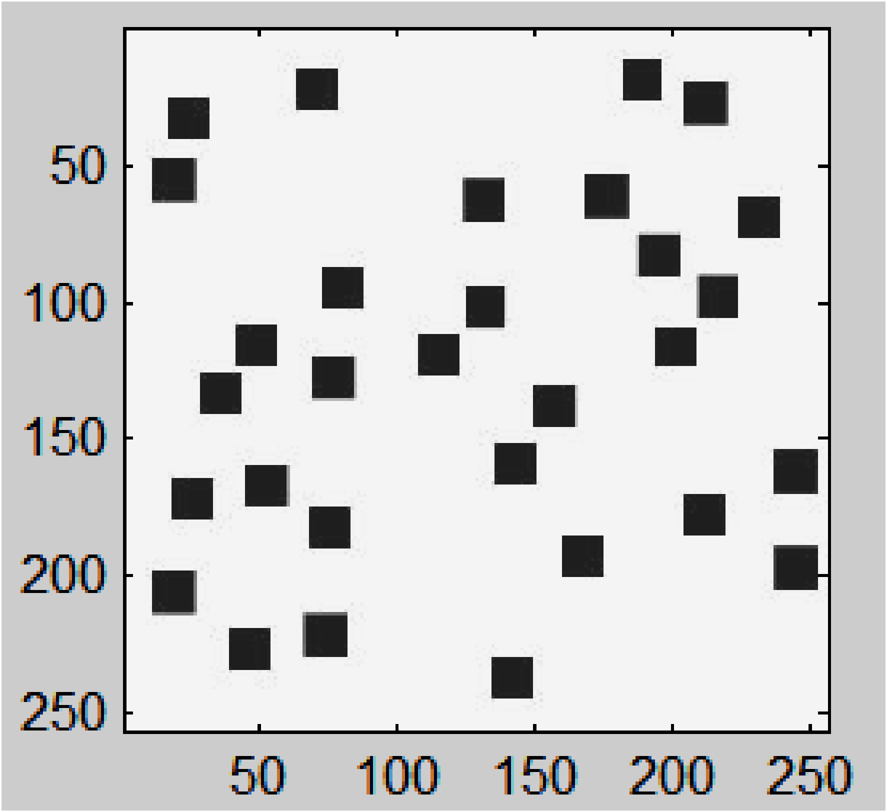
$$Y' = f\,(Y)$$



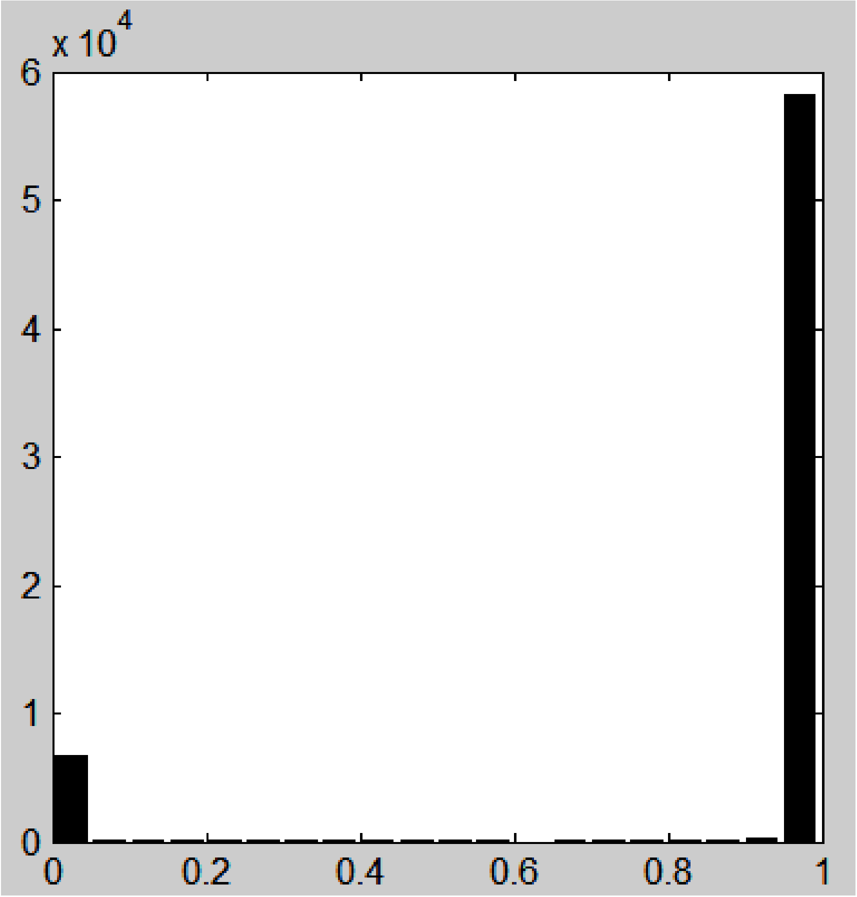
2-Matching histograms



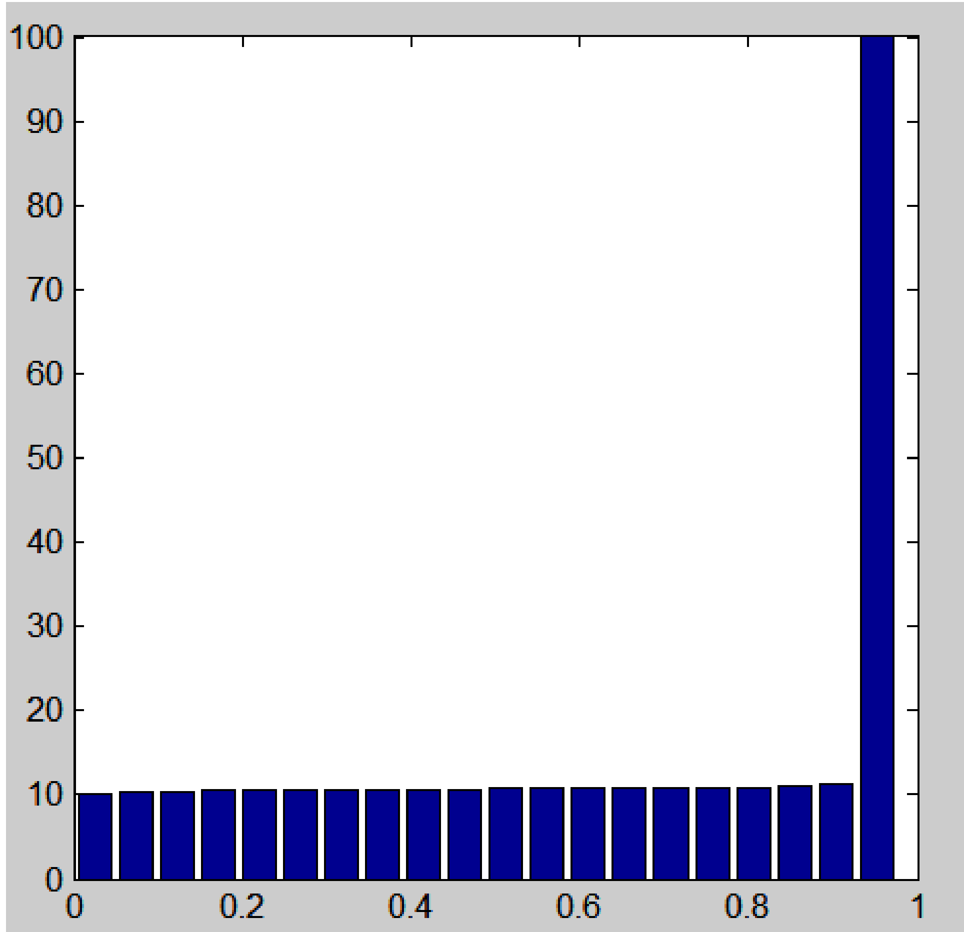
Matching histograms example



Histograms



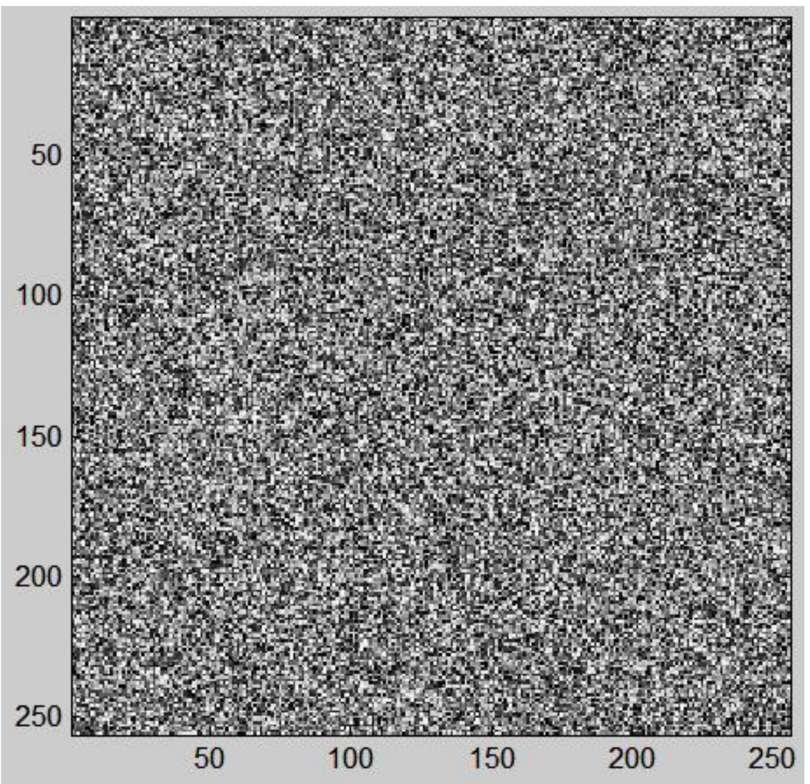
Cumulative Histograms

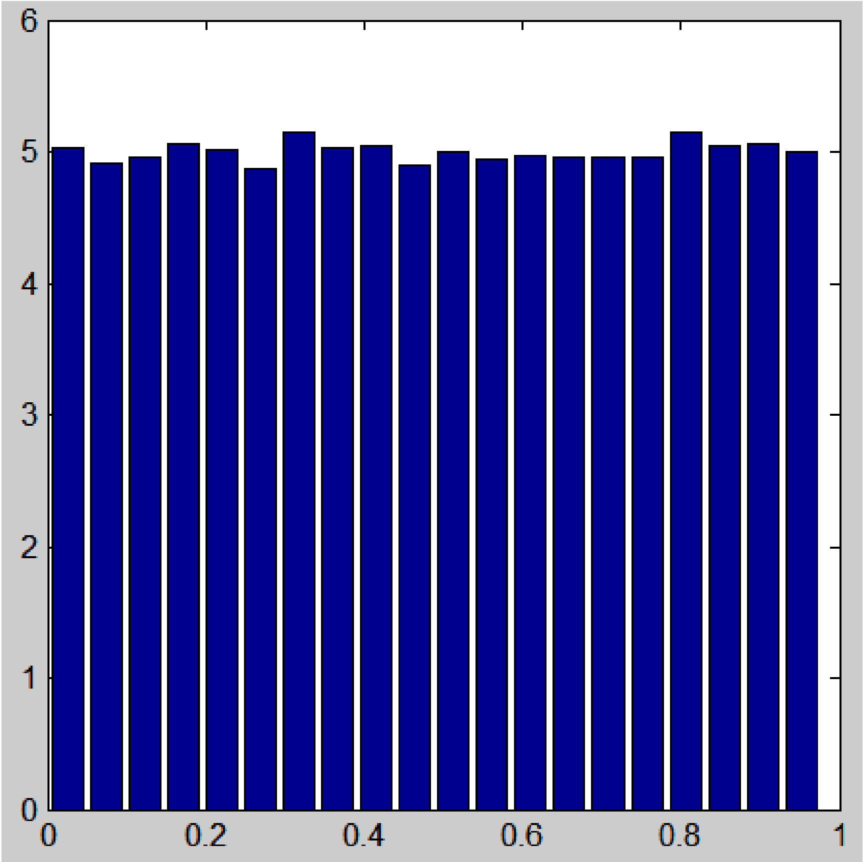


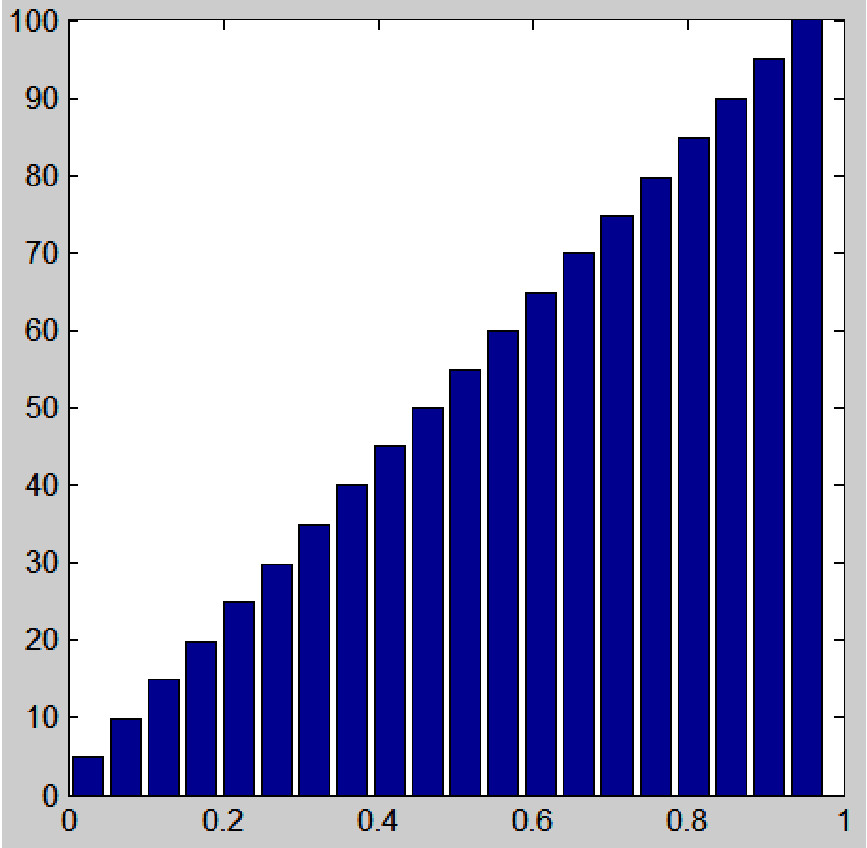
?

?

10% of pixels are black
and 90% are white



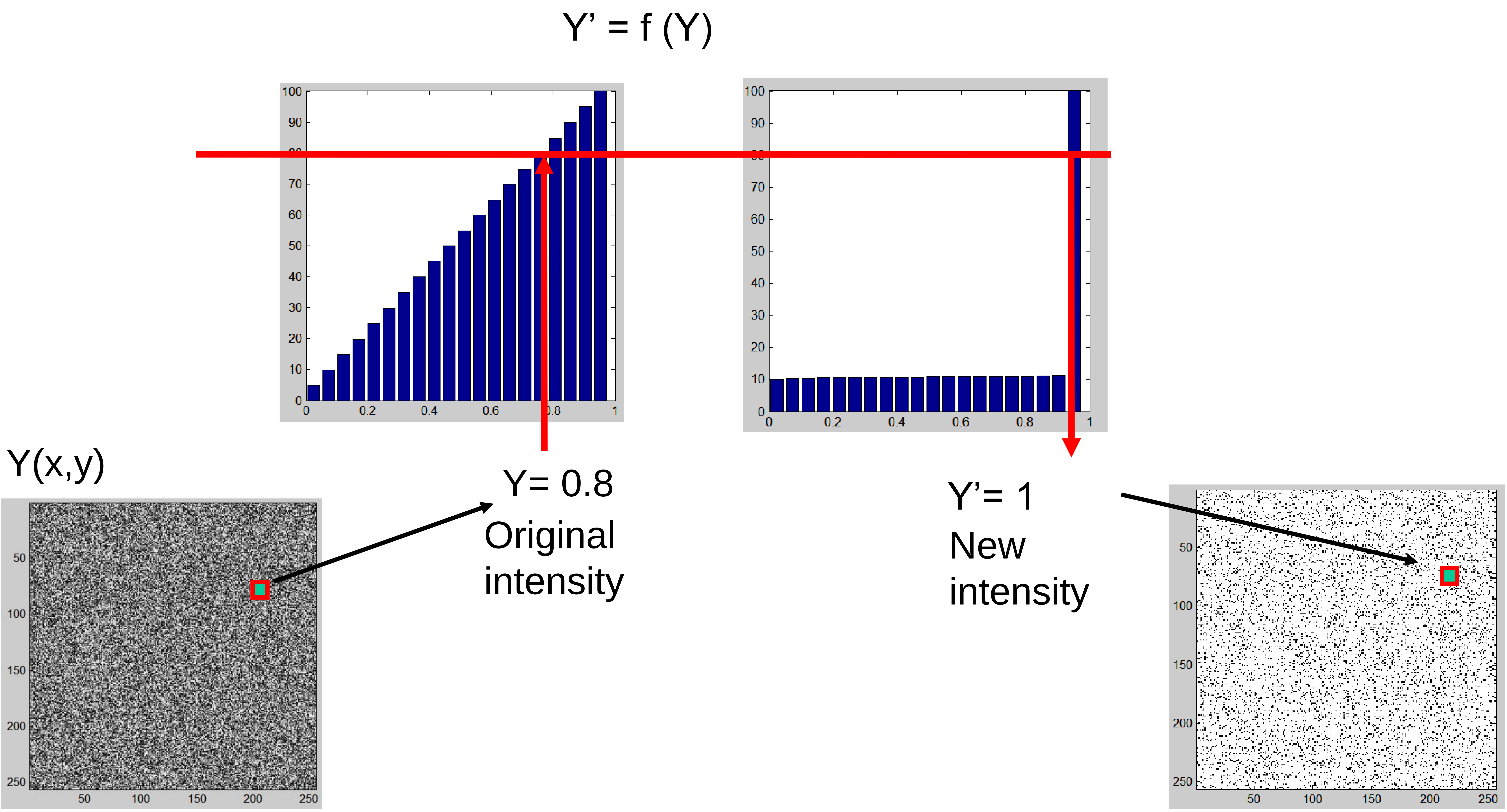




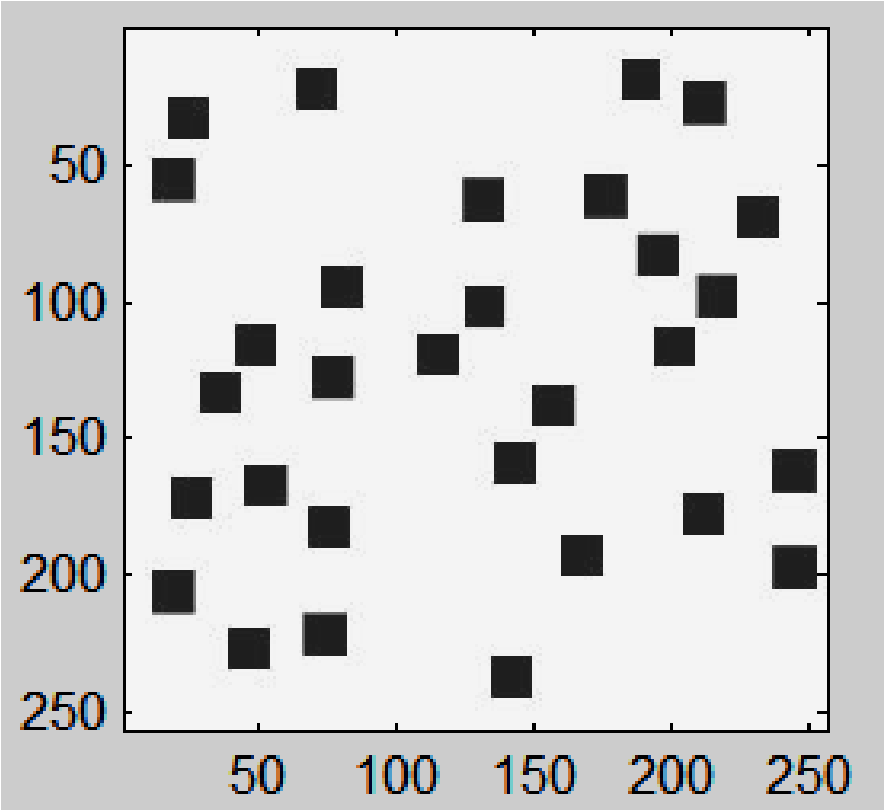
5% of pixels have an intensity value
within the range[0.37, 0.41]

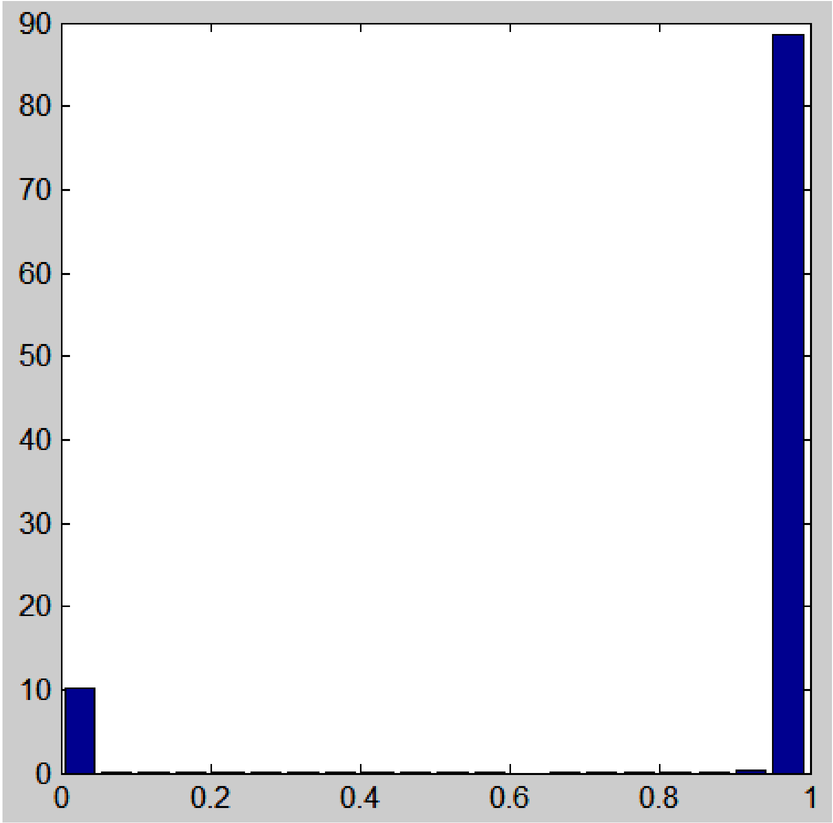
Matching histograms example

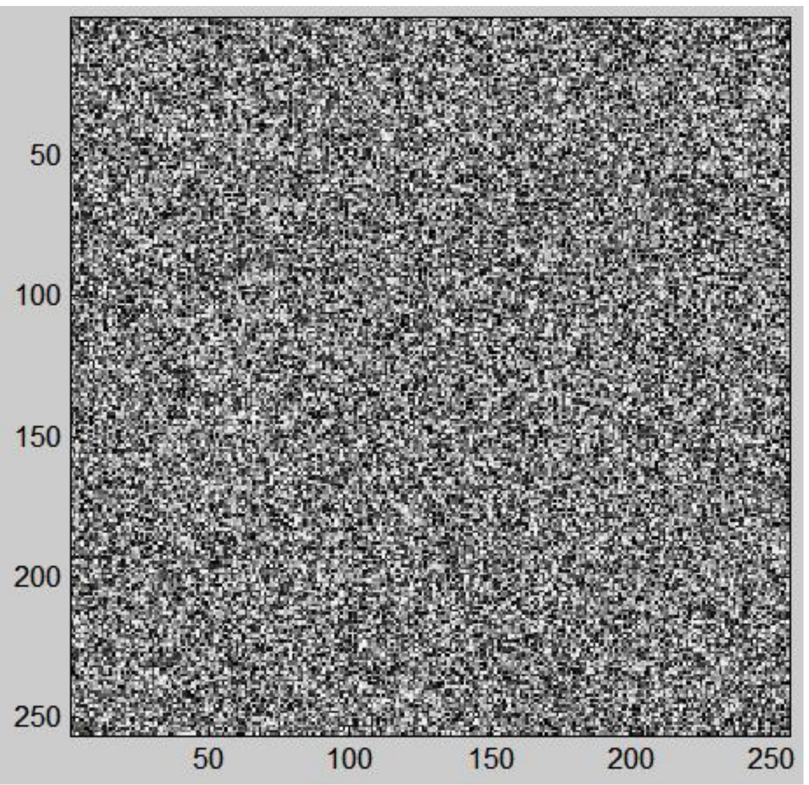
The function f is just a look up table: it says, change all the pixels of value Y into a value f(Y).



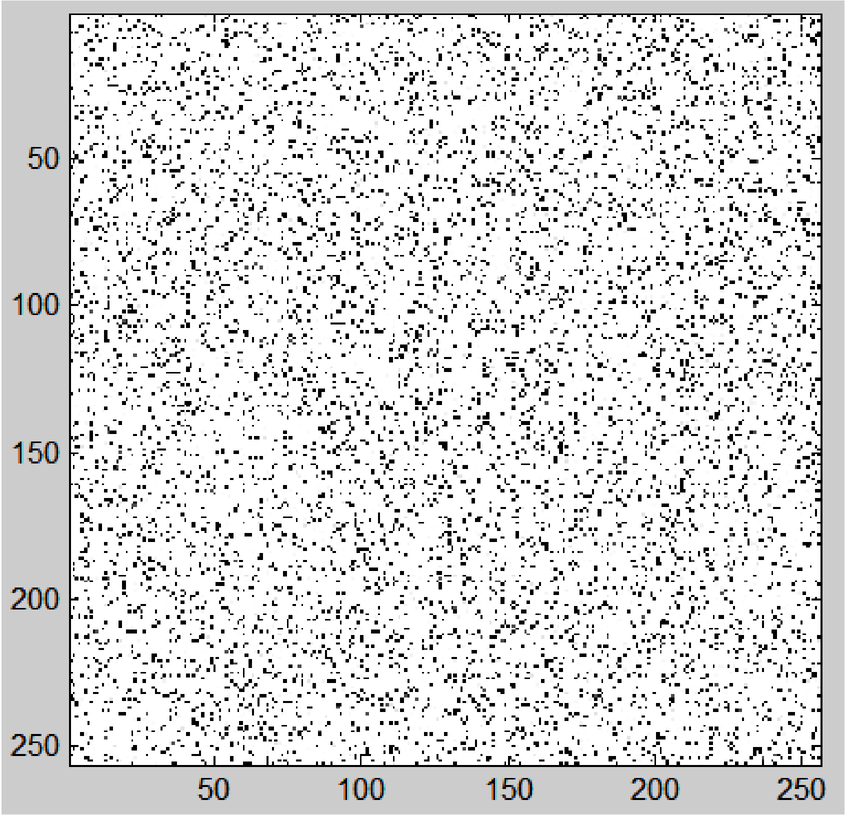
Matching histograms example

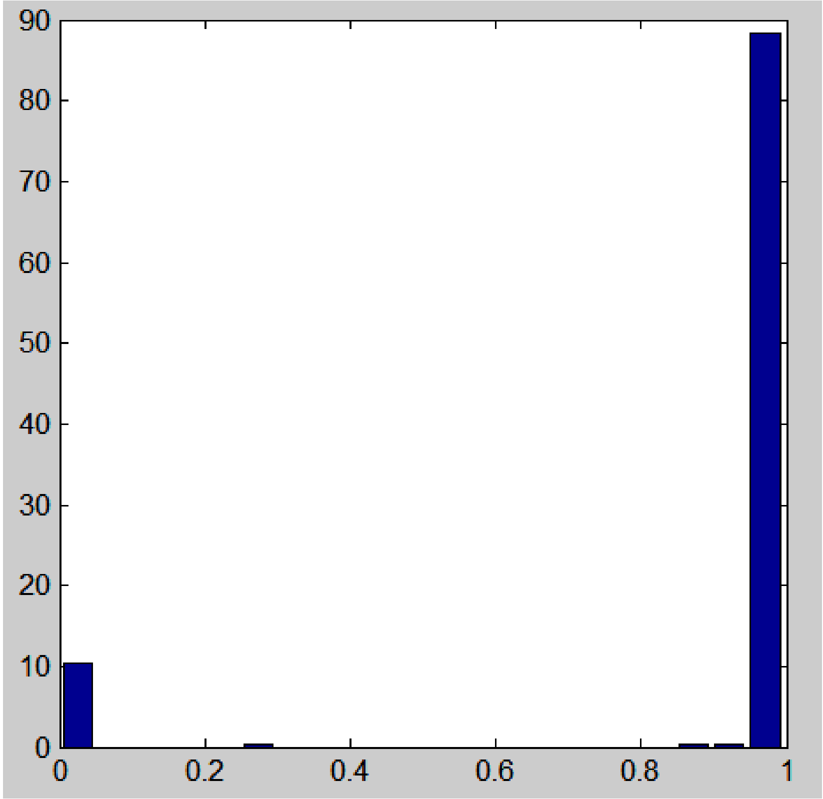






Y’ = f (Y)



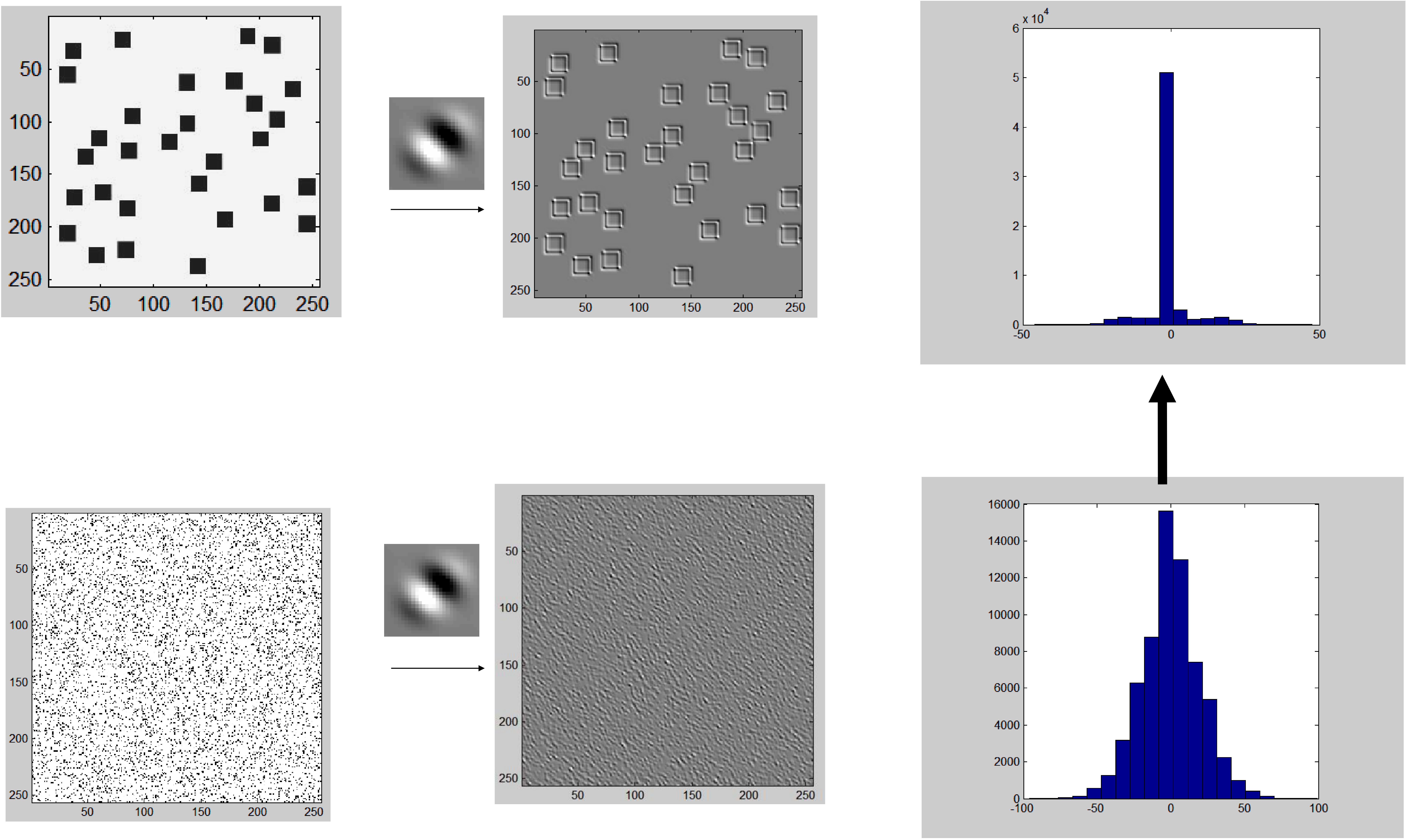


In this example, f is a step function.

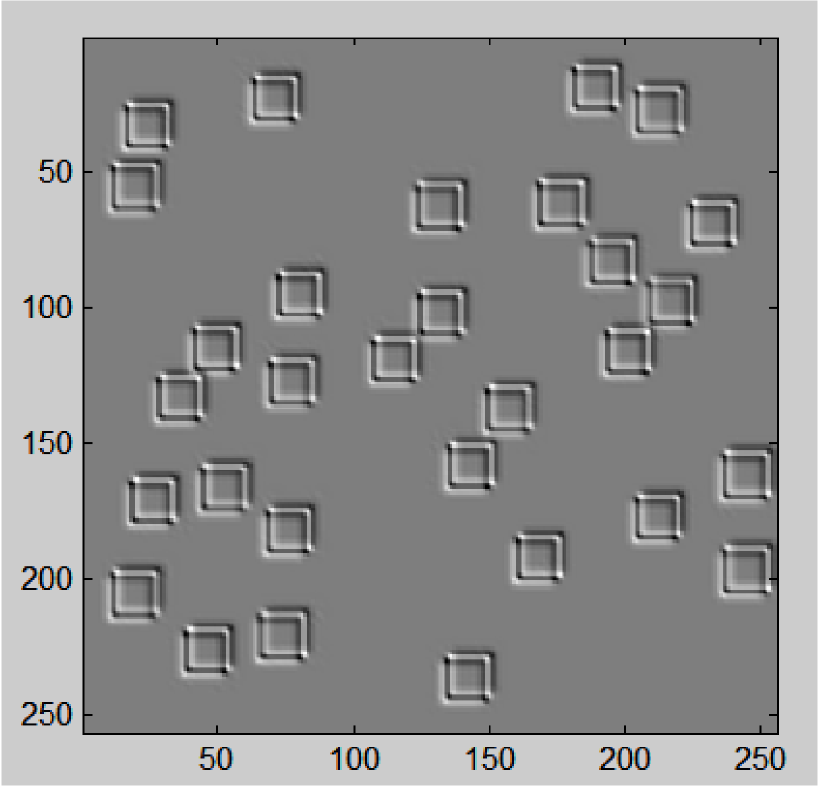
Color Transfer [Reinhard, et al, 2001]

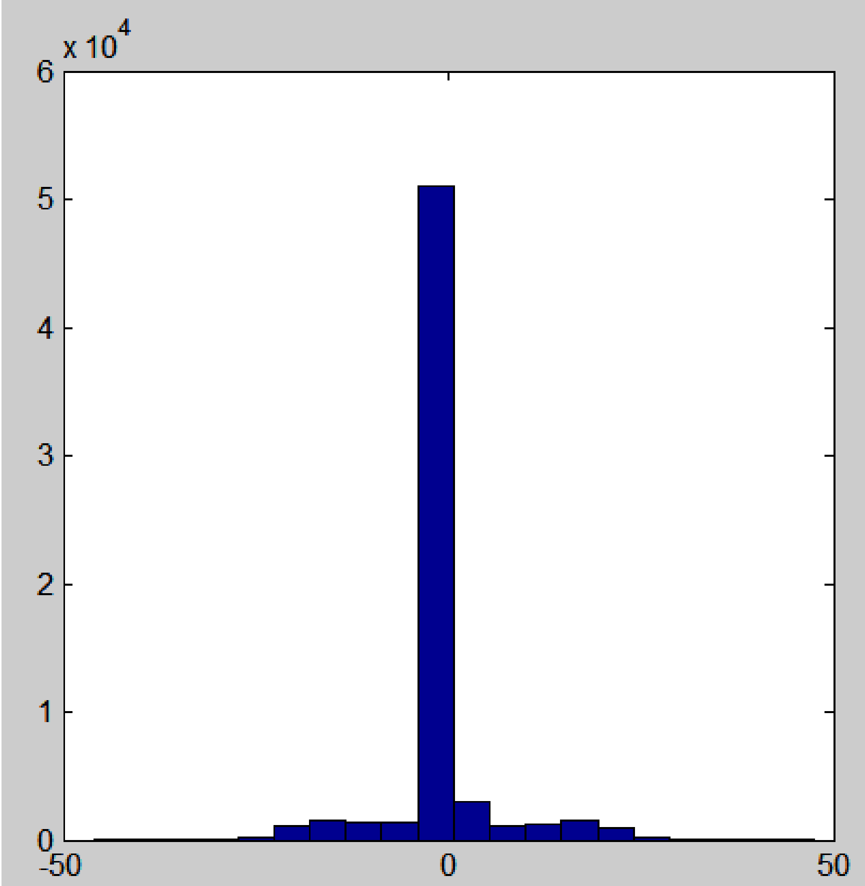
Erik Reinhard, Michael Ashikhmin, Bruce Gooch, Peter Shirley, **Color Transfer** between Images. *IEEE Computer Graphics and Applications*, 21(5), pp. 34–41. September 2001.

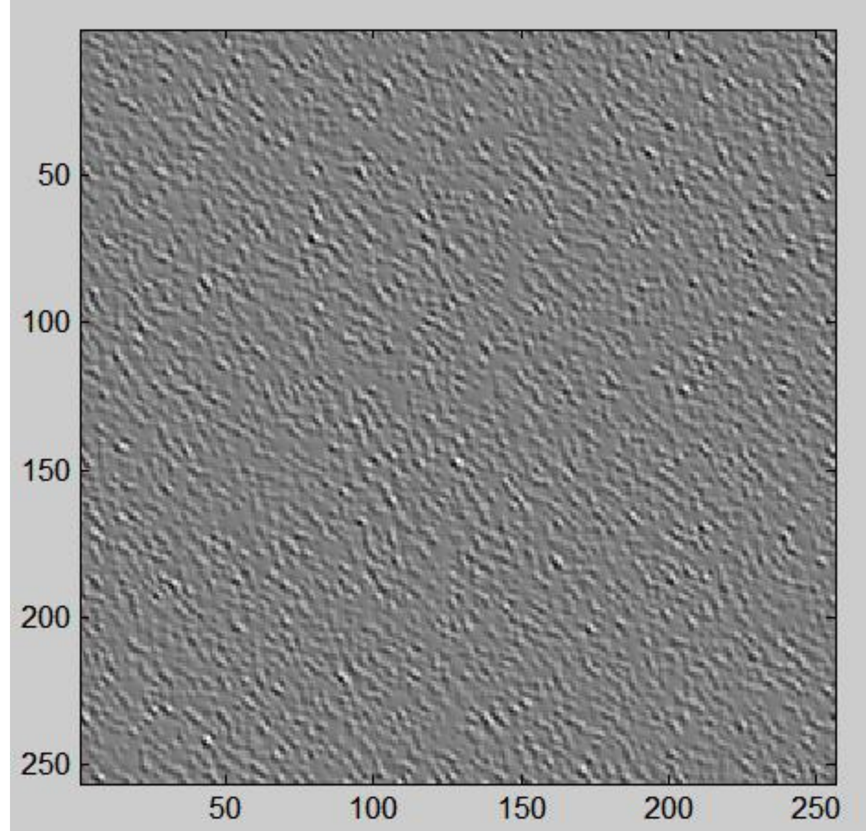
Matching histograms of a subband



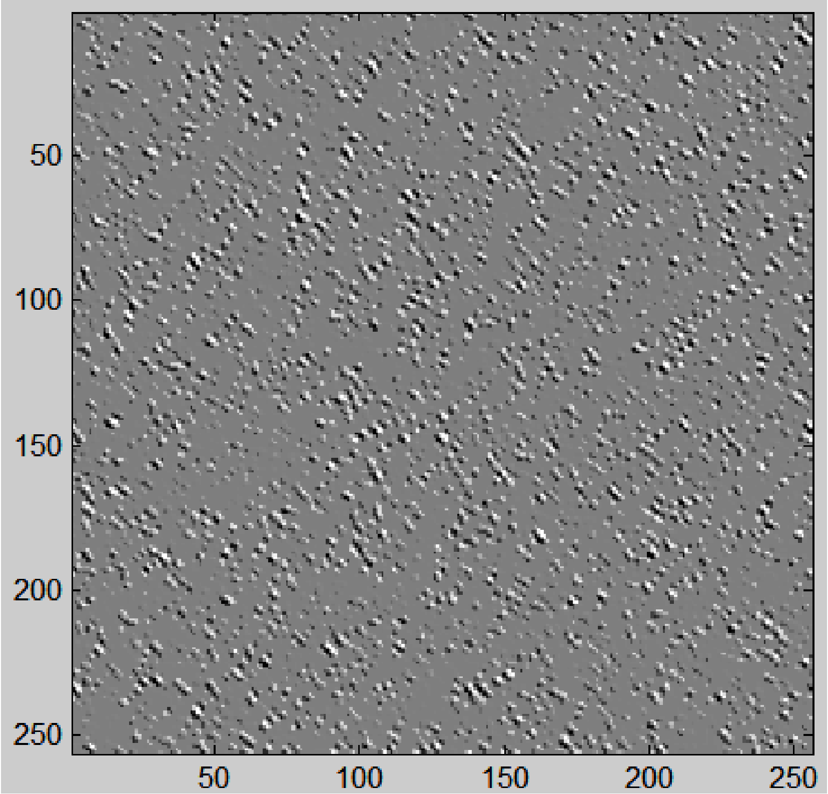
Matching histograms of a subband

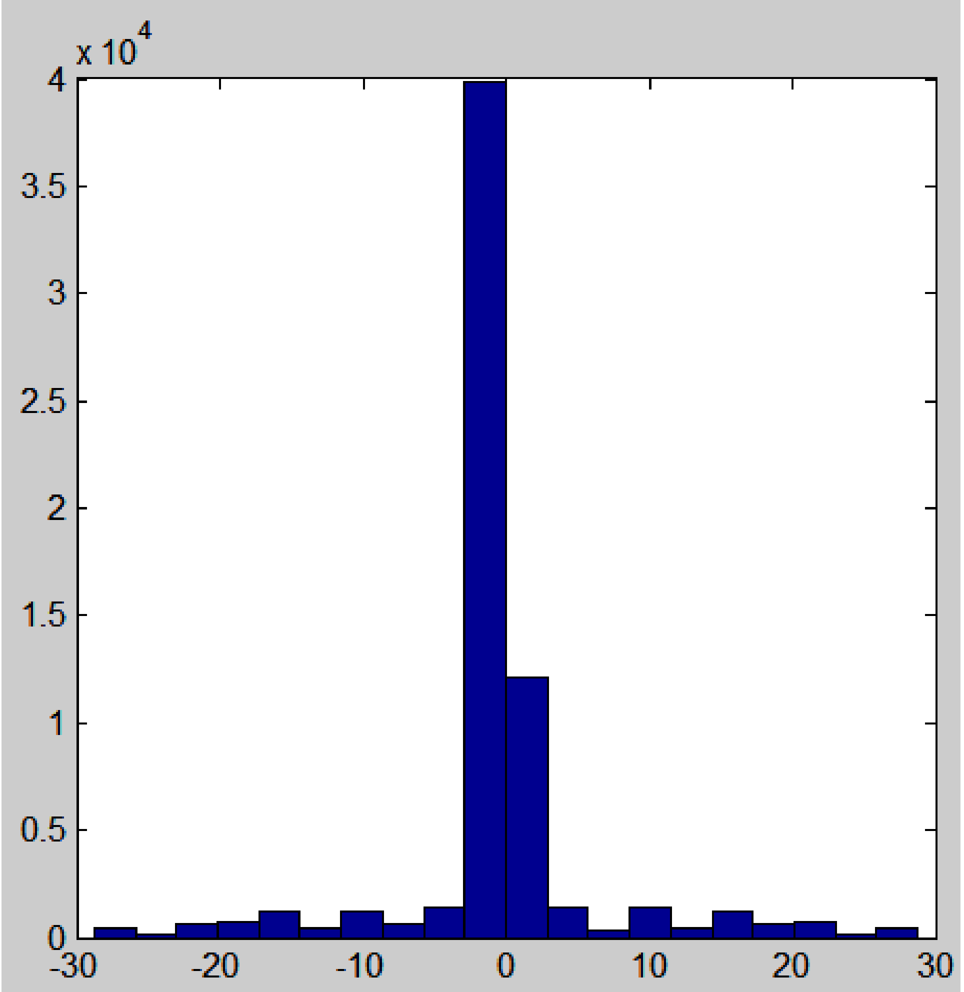






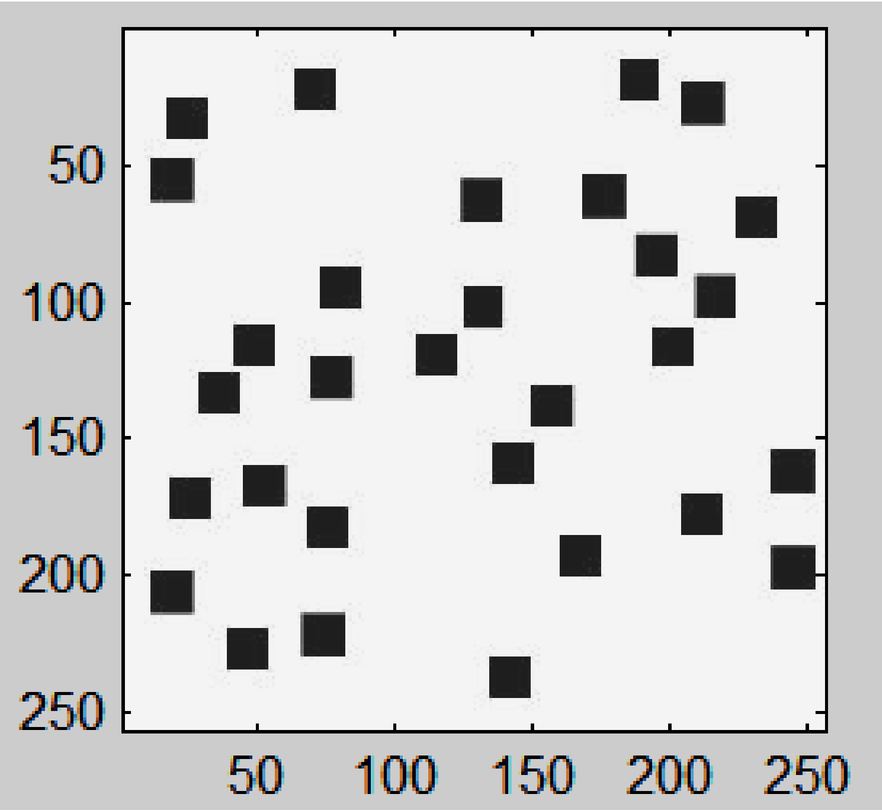
Y’ = f (Y)



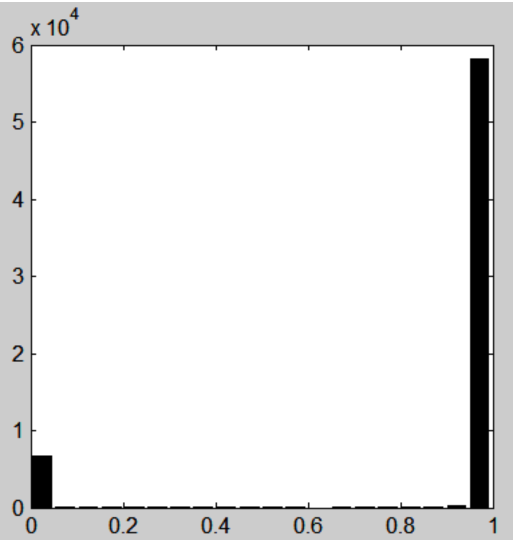


Texture analysis

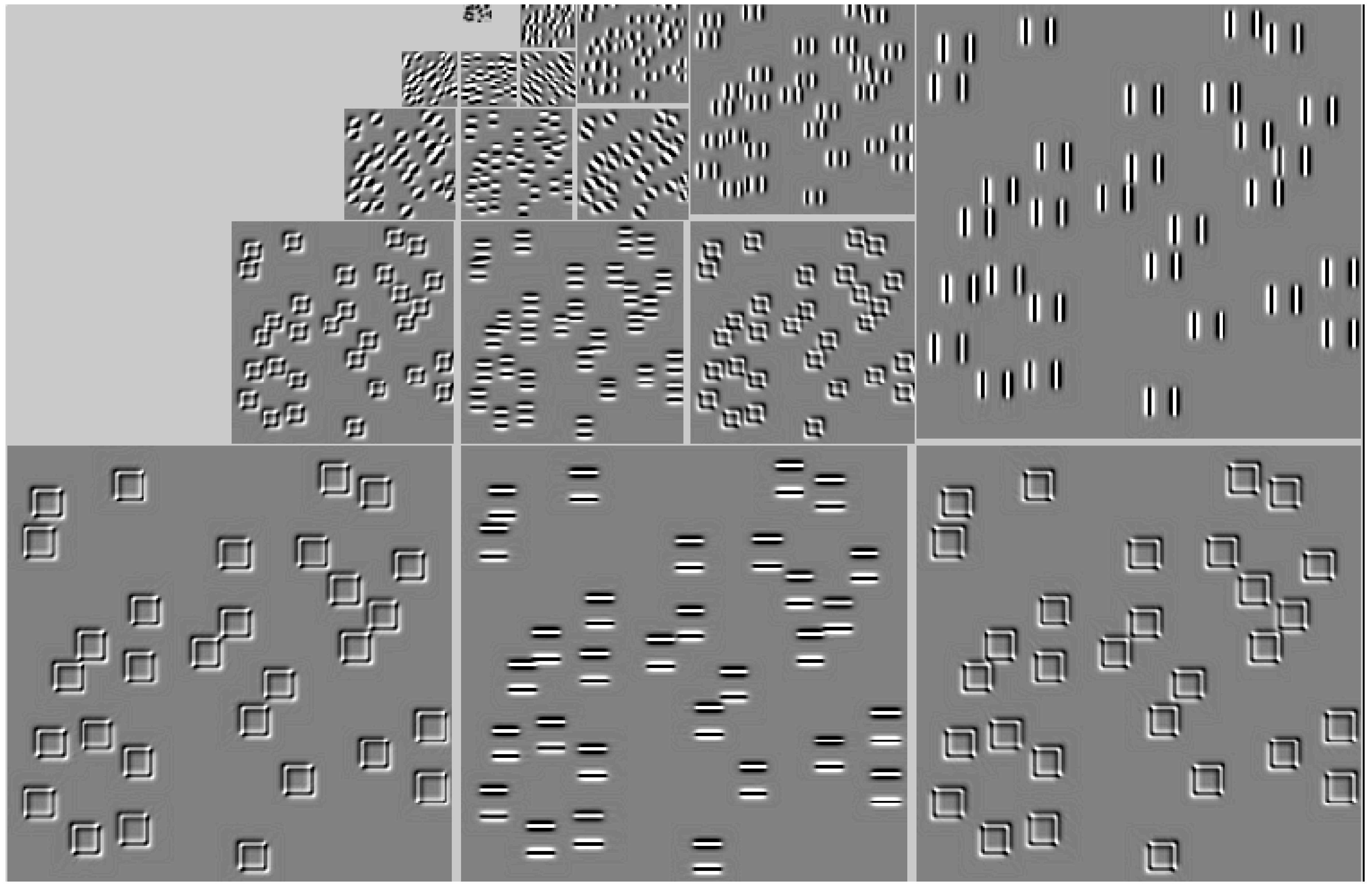
Input texture



(histogram)

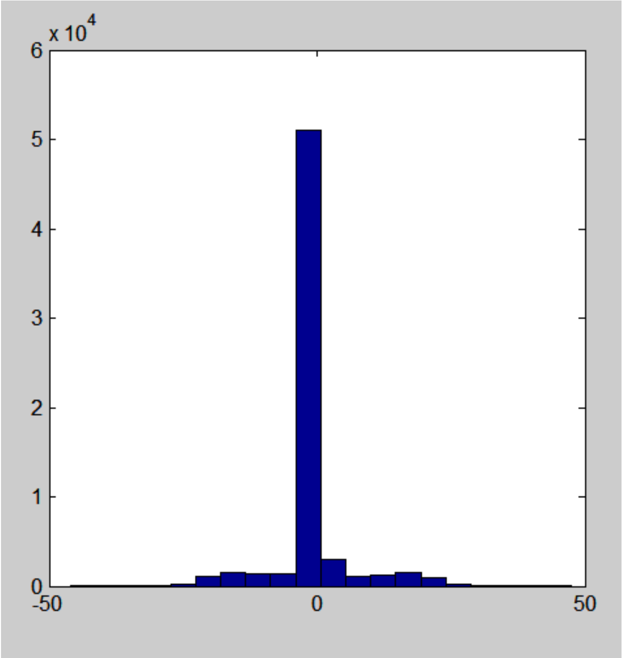


Wavelet decomposition (steerable pyr)



(Steerable pyr; Simoncelli & Freeman, ’95)

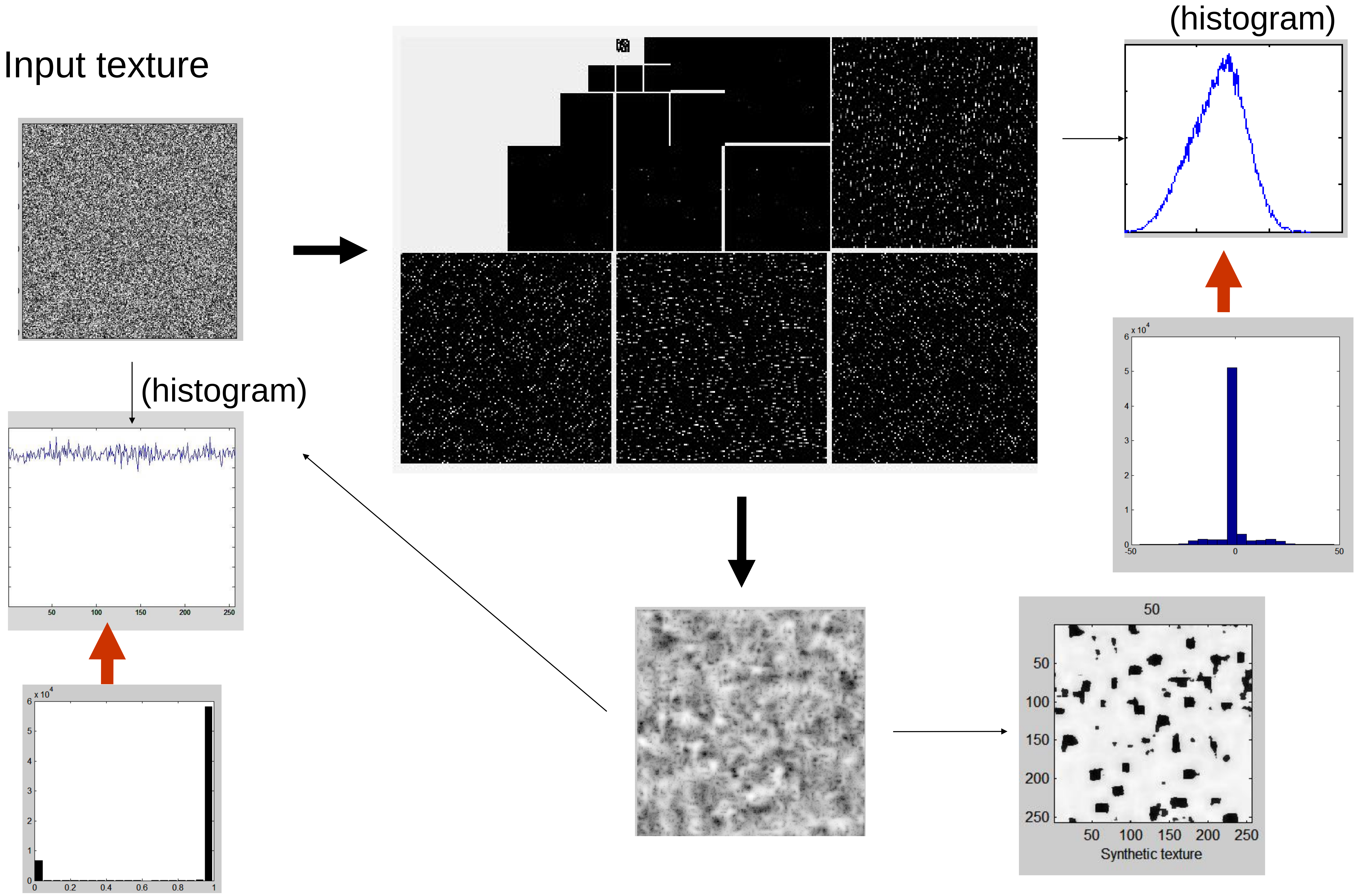
(histogram)



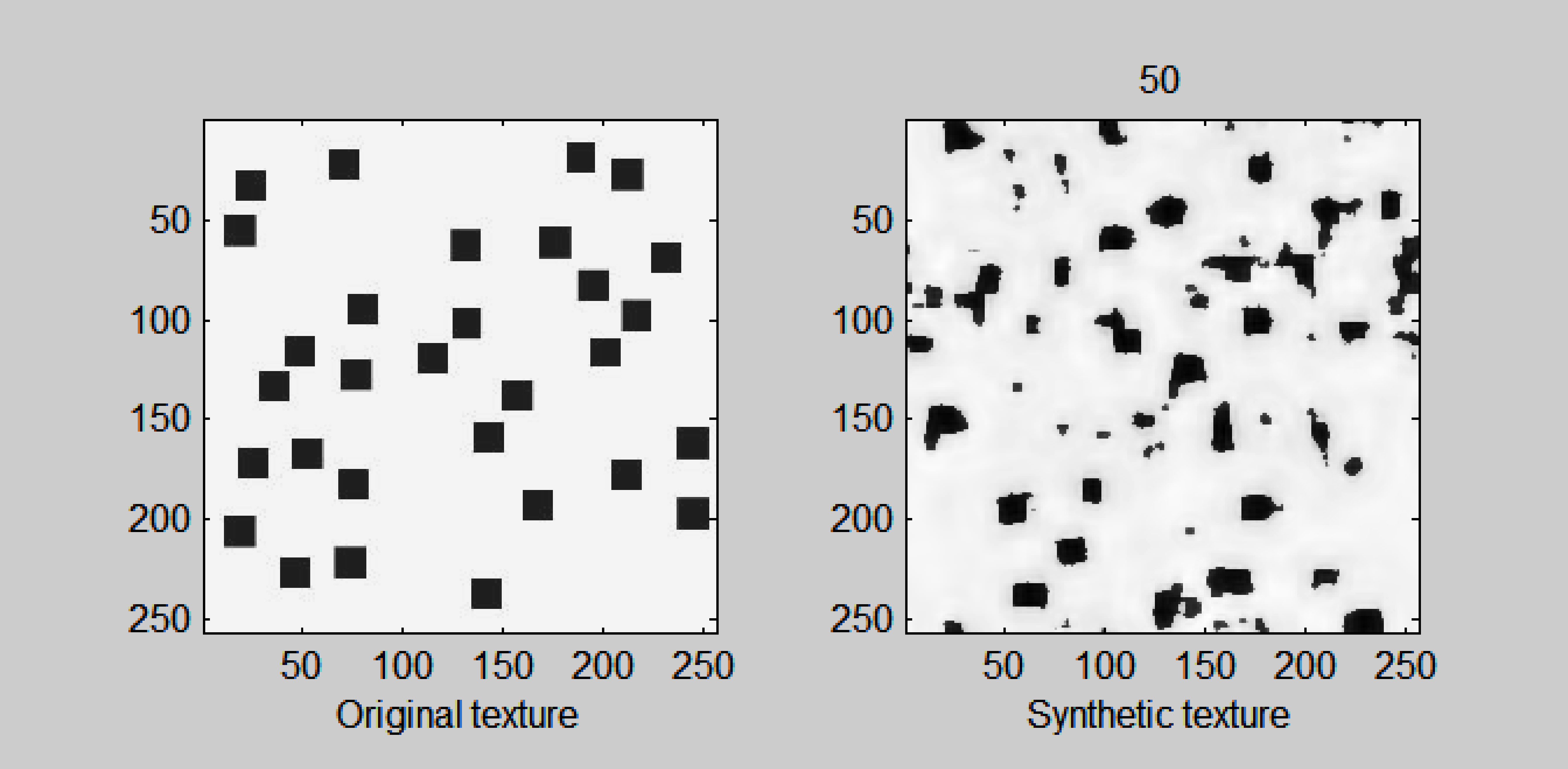
The texture is represented as a collection of marginal histograms.

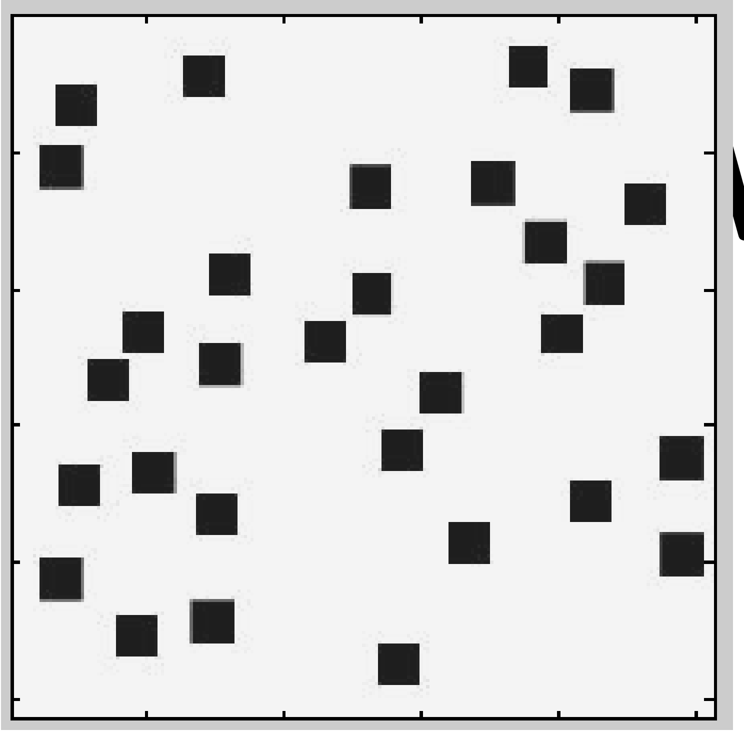
Texture synthesis

Heeger and Bergen, 1995



Why does it work? (sort of)



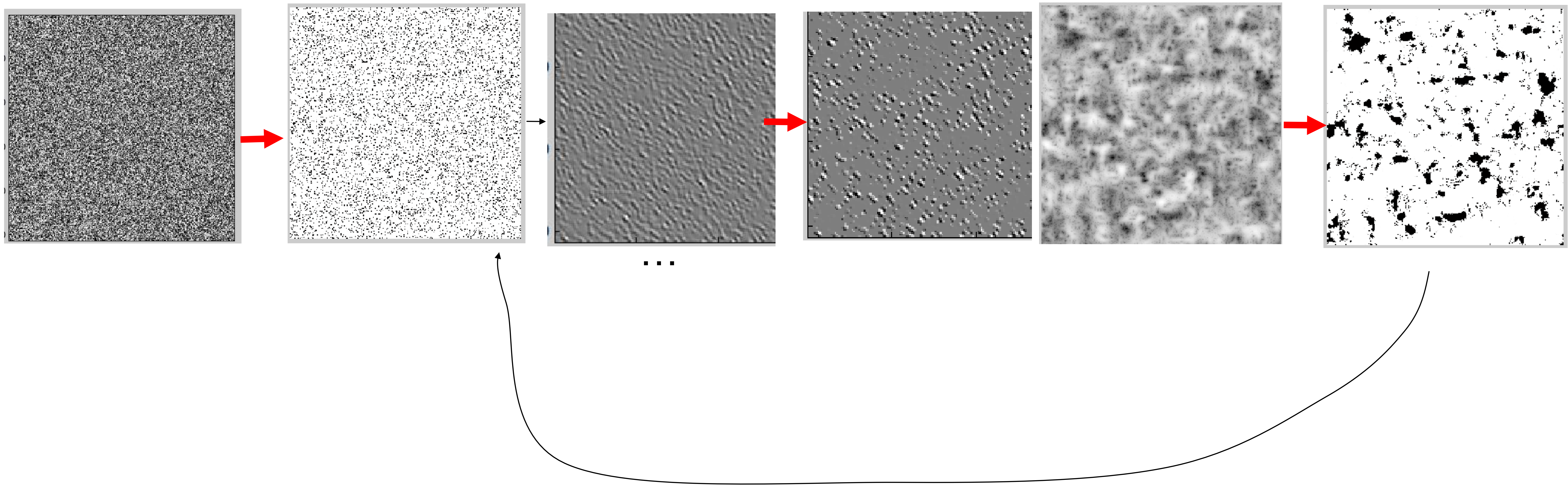


Why does it work? (sort of)

The black and white blocks appear by thresholding (f) a blobby image

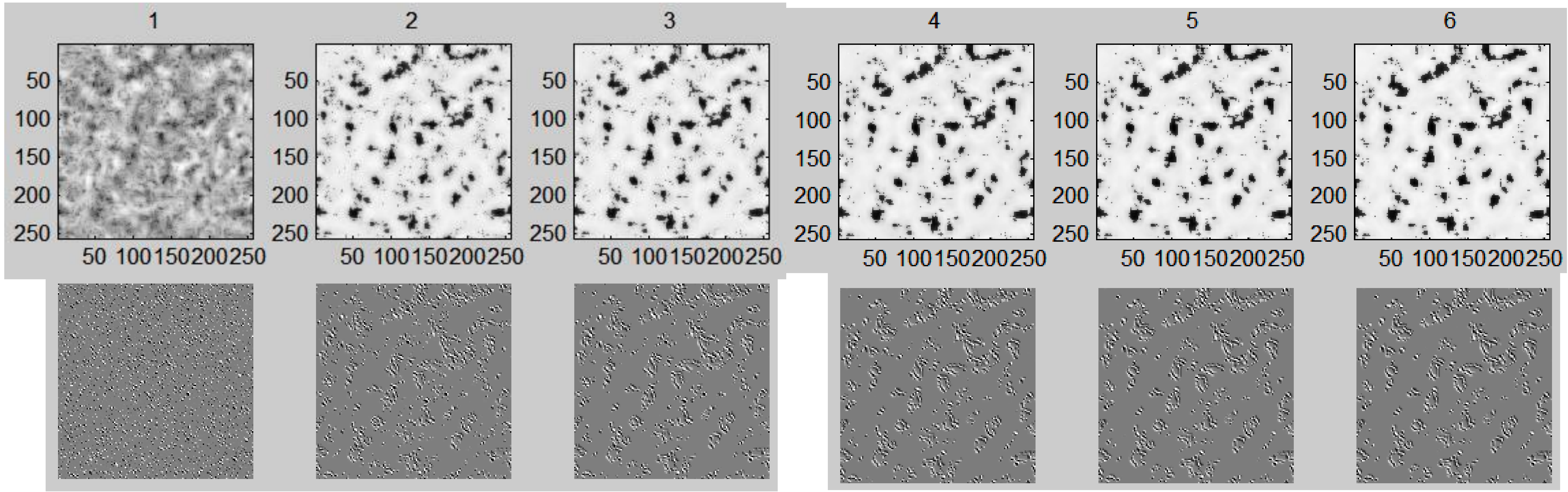
Iteration 0

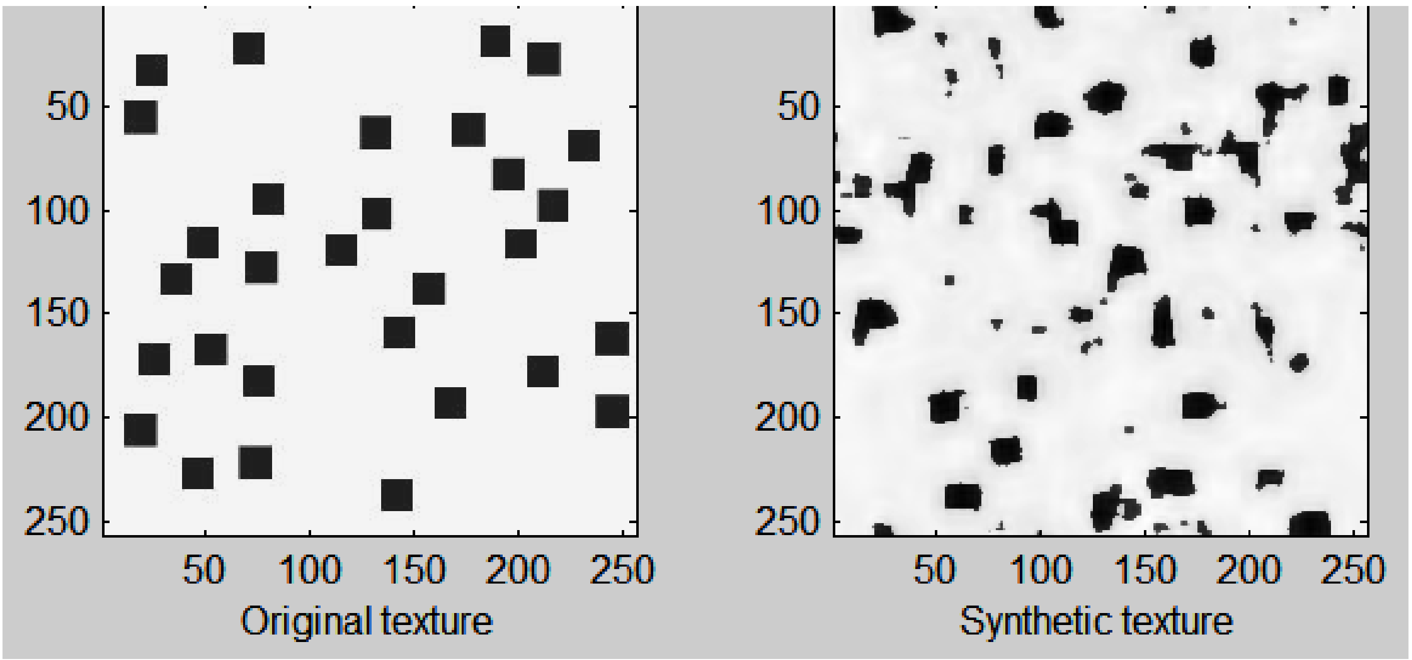
Filter bank



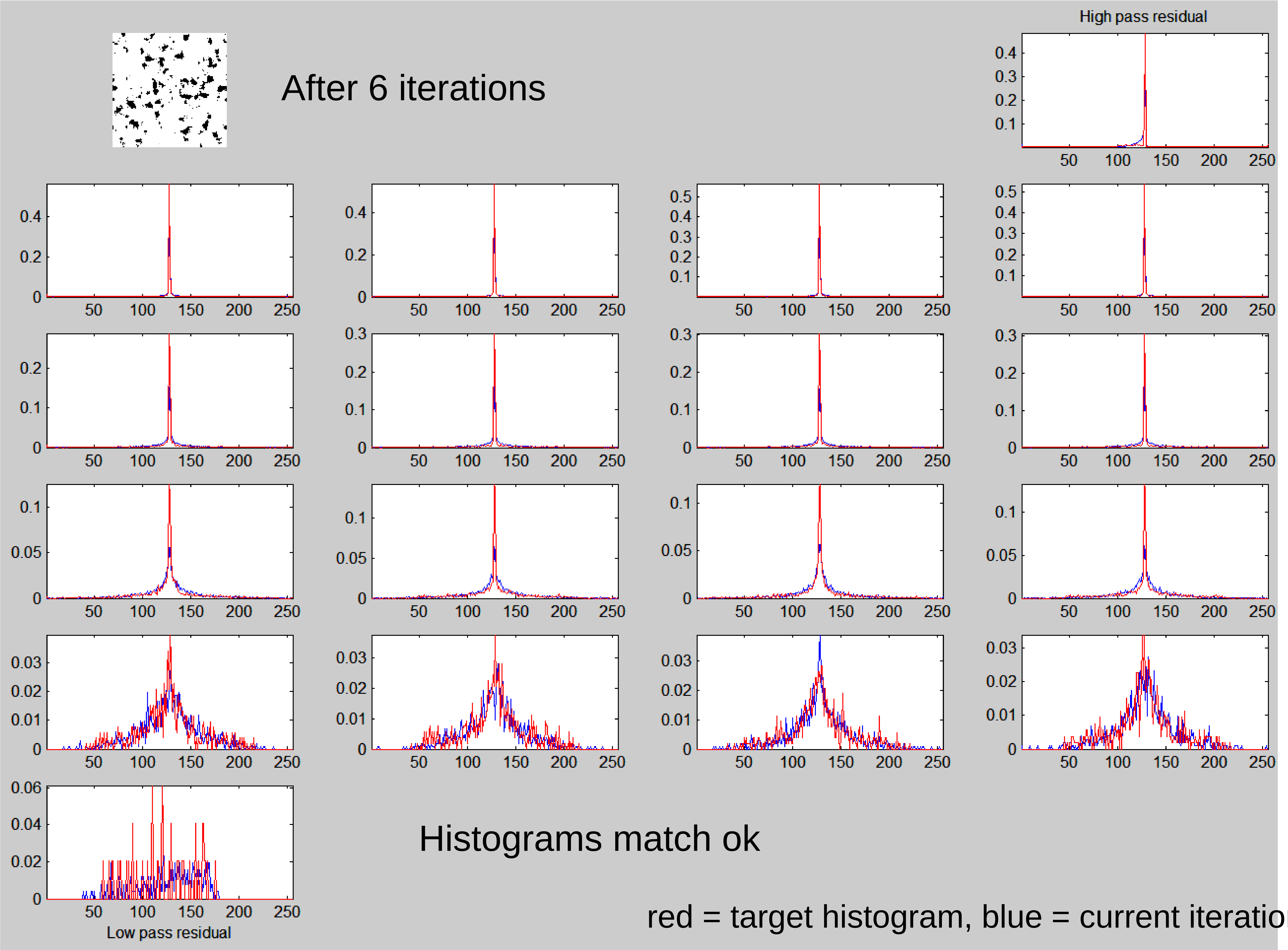
Why does it work? (sort of)

The black and white blocks appear by thresholding (f) a blobby image





Why does it work? (sort of)



red = target histogram, blue = current iteration

Examples from the paper

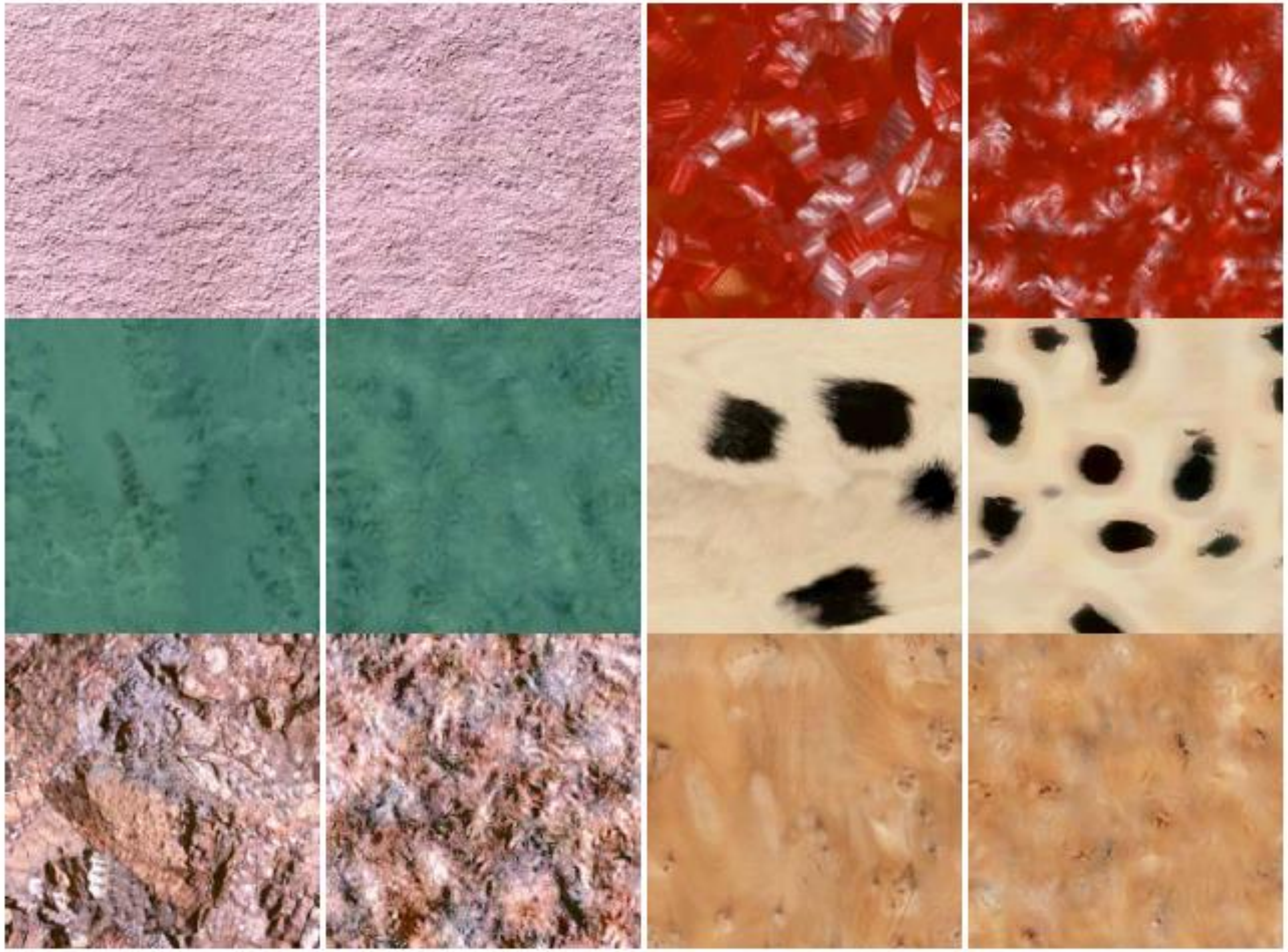
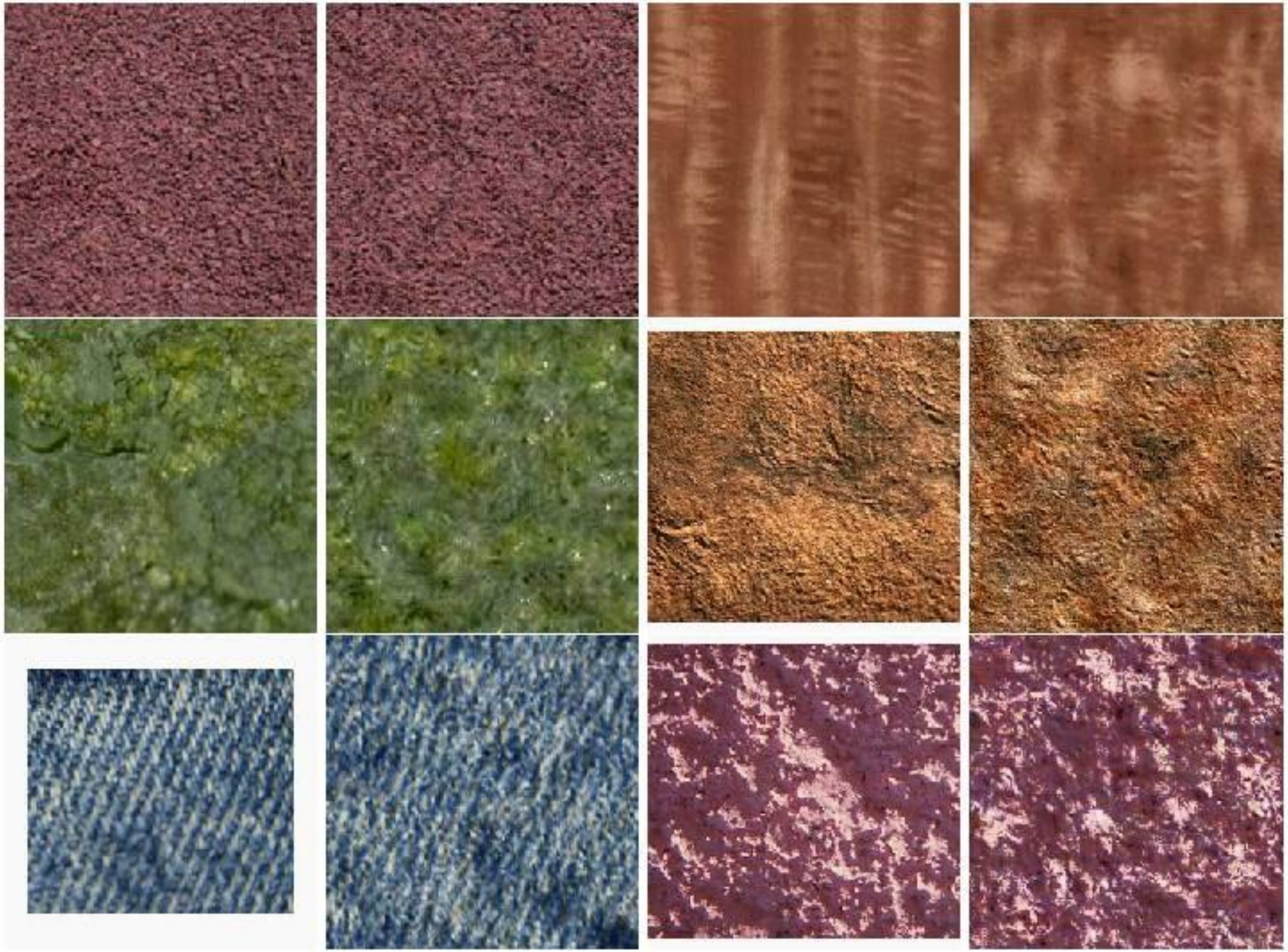


Figure 3: In each pair left image is original and right image is synthetic: stucco, iridescent ribbon, green marble, panda fur, slag stone, figured yew wood.

Heeger and Bergen, 1995



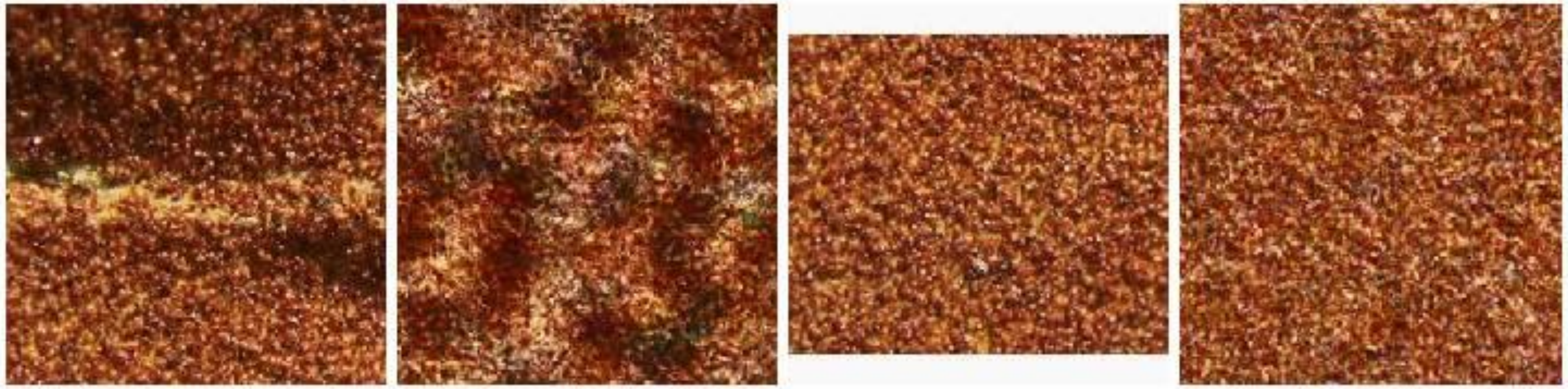


Figure 7: (Left pair) Inhomogeneous input texture produces blotchy synthetic texture. (Right pair) Homogenous input.

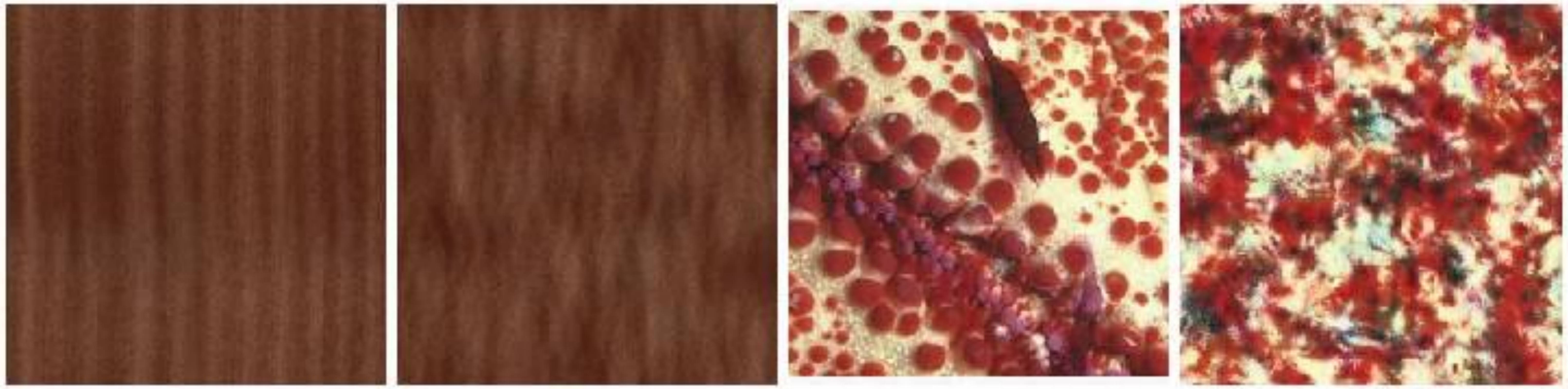


Figure 8: Examples of failures: wood grain and red coral.

Figure 9: More failures: hay and marble.

Simoncelli & Portilla ’98+

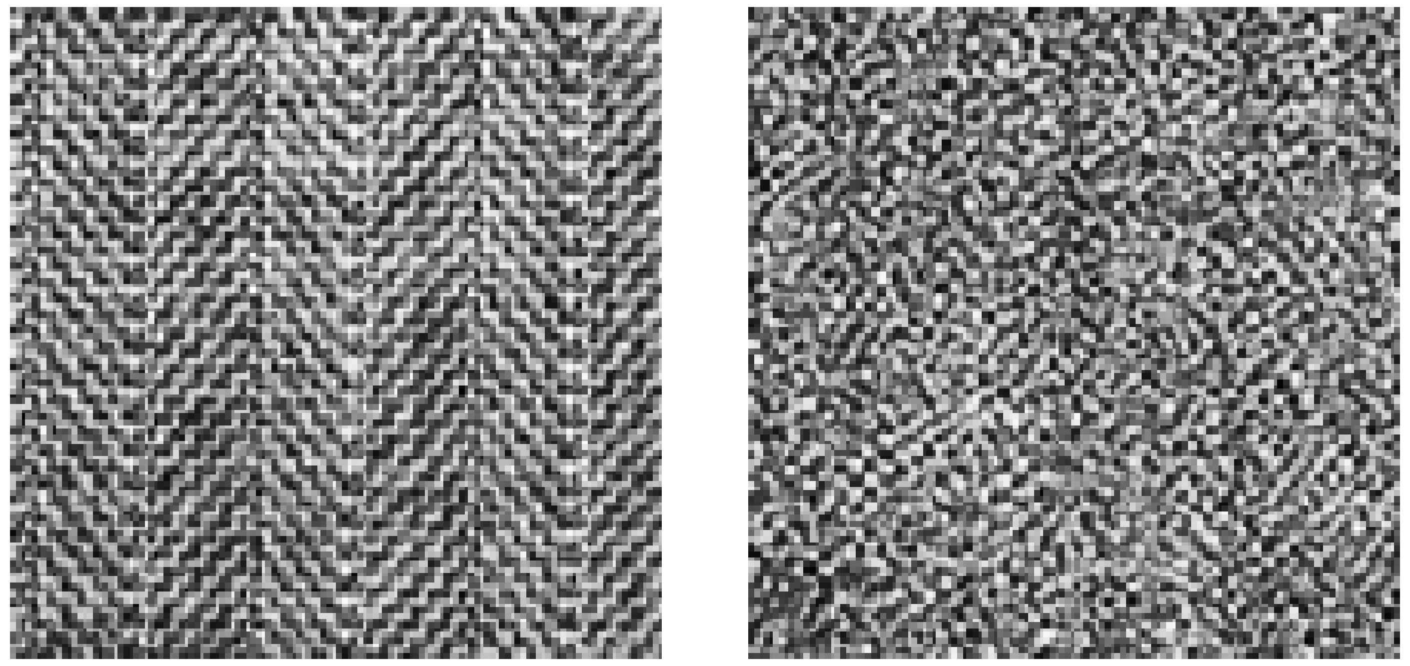
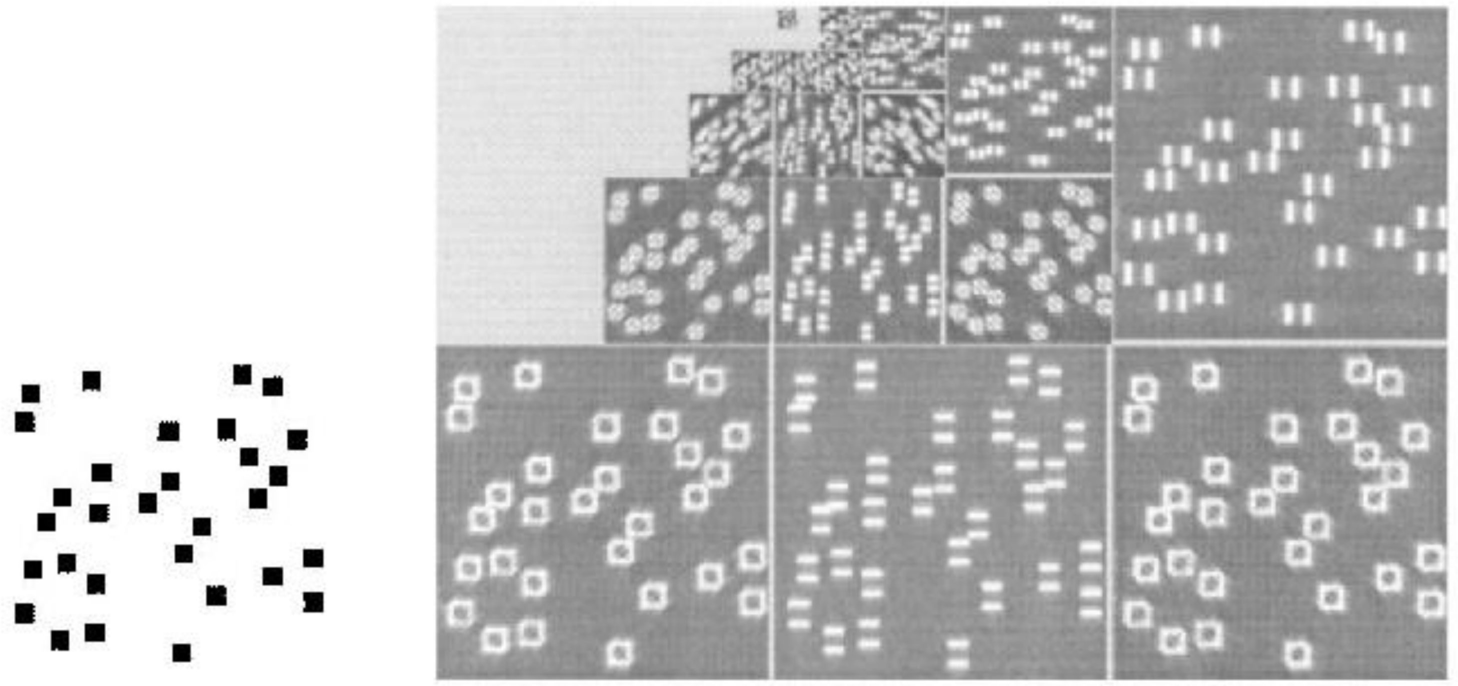


Figure 1. Textures with matching marginal statistics.

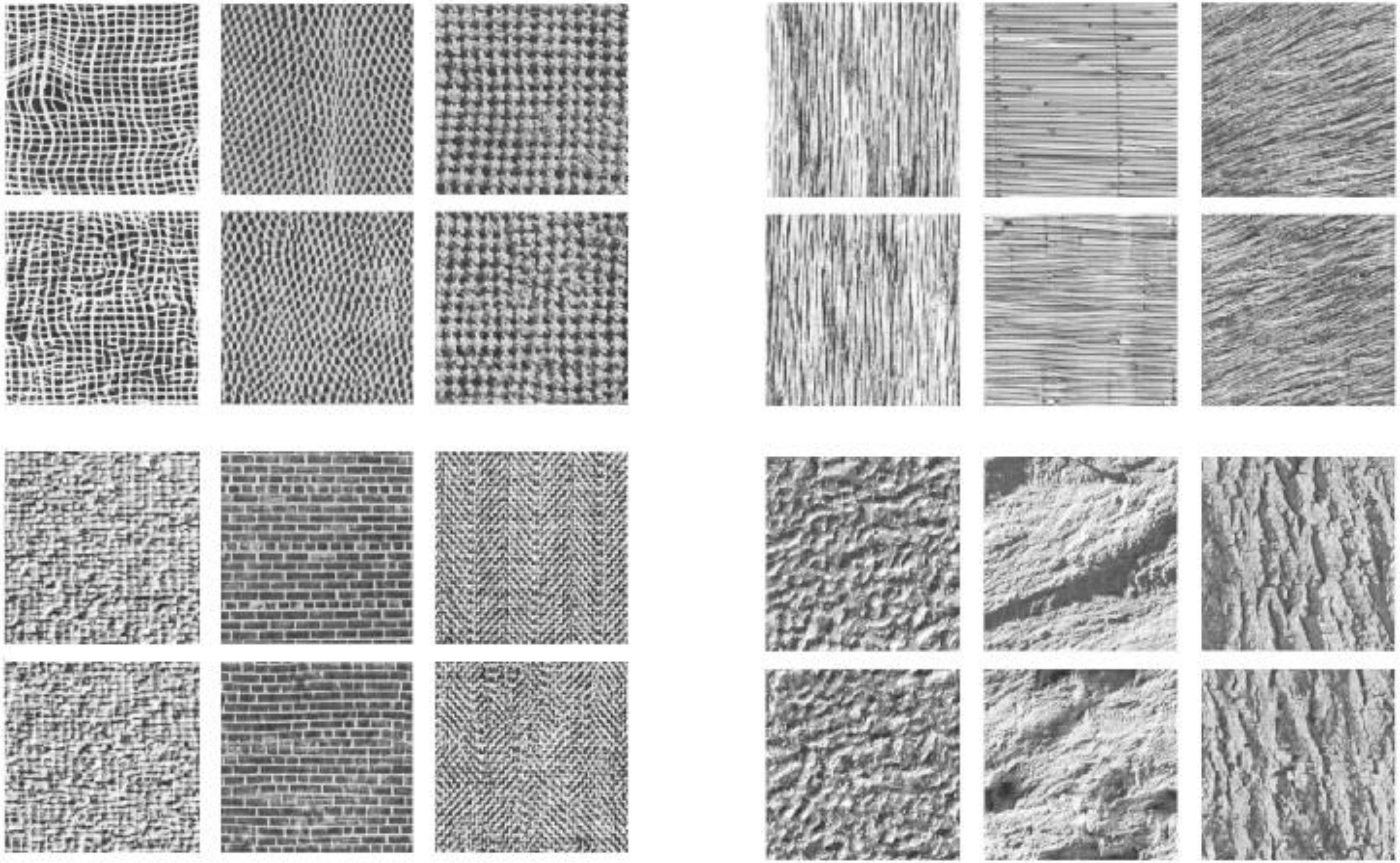
- Marginal statistics are not enough
- Neighboring filter responses are highly correlated
  - an edge at low-res will cause an edge at high-res
- Let’s match 2nd order statistics too!

- J Portilla and E P Simoncelli. *A Parametric Texture Model based on Joint Statistics of Complex Wavelet Coefficients.* Int'l Journal of Computer Vision. 40(1):49-71, October, 2000.

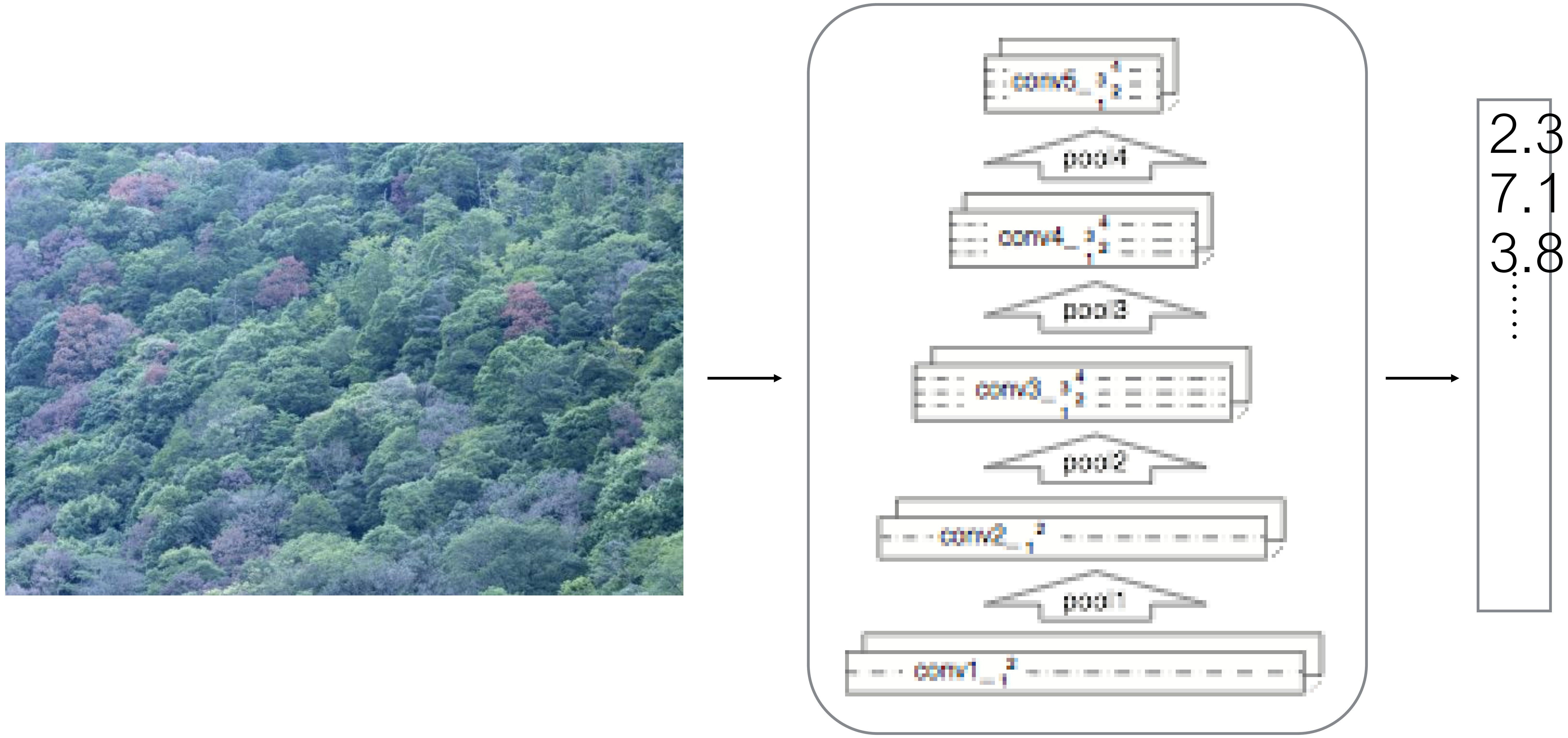
Simoncelli & Portilla ’98+



- Match joint histograms of pairs of filter responses at adjacent spatial locations, orientations, and scales.
- Optimize using repeated projections onto statistical constraint sufraces



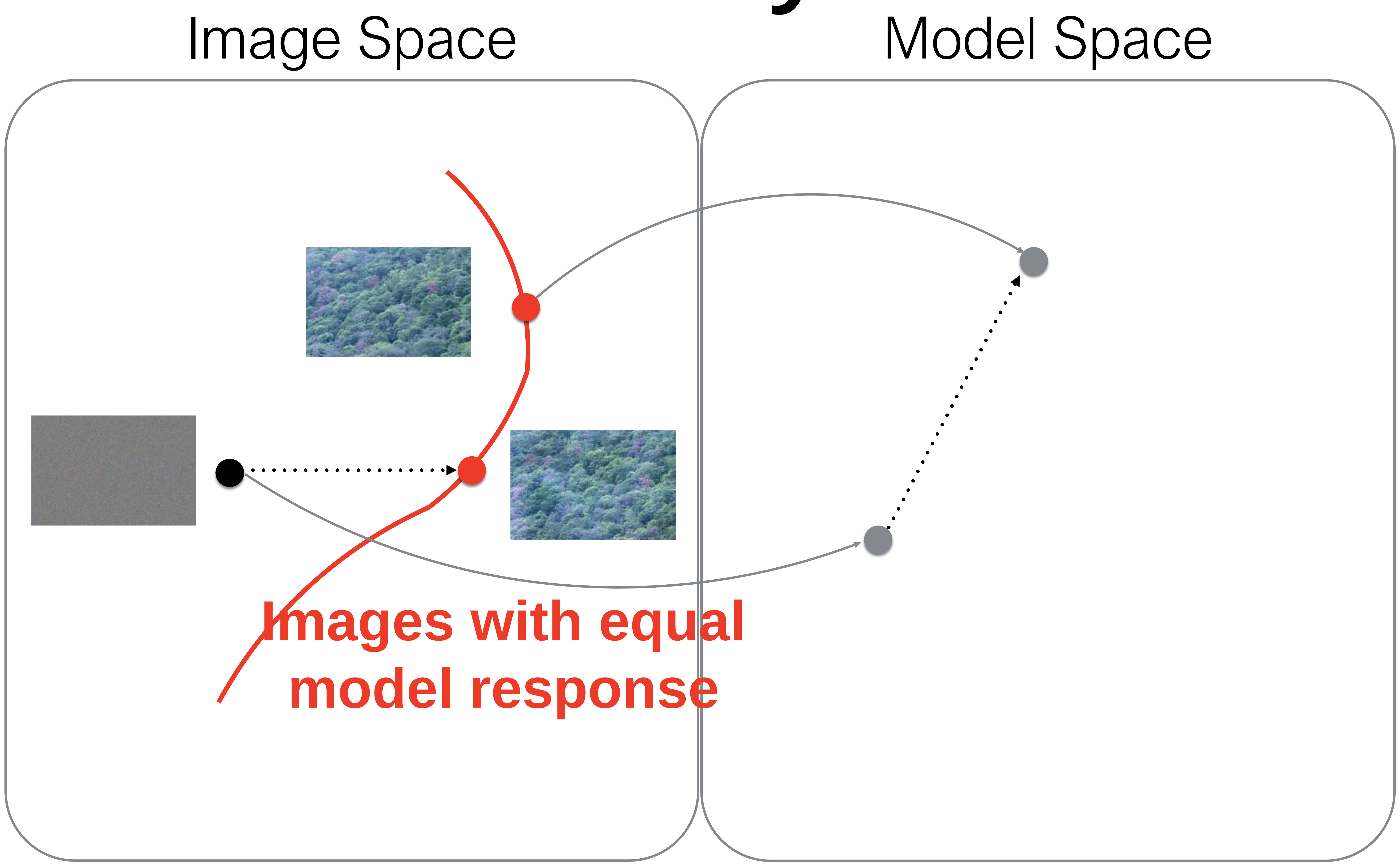
Convolutional Neural Network Texture Model



Convolutional Neural Network

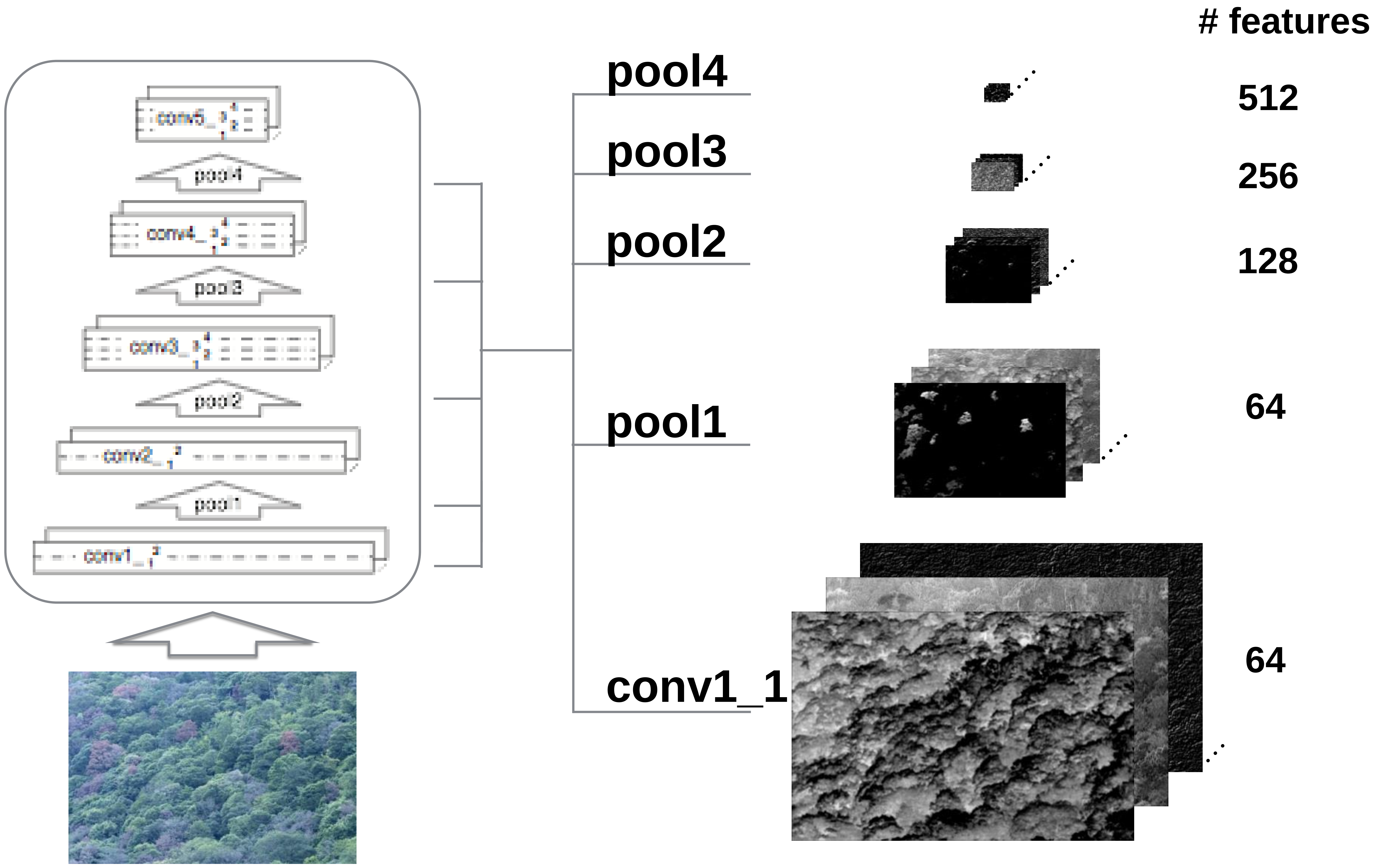
Gatys et al. (NIPS 2015)

Texture Synthesis

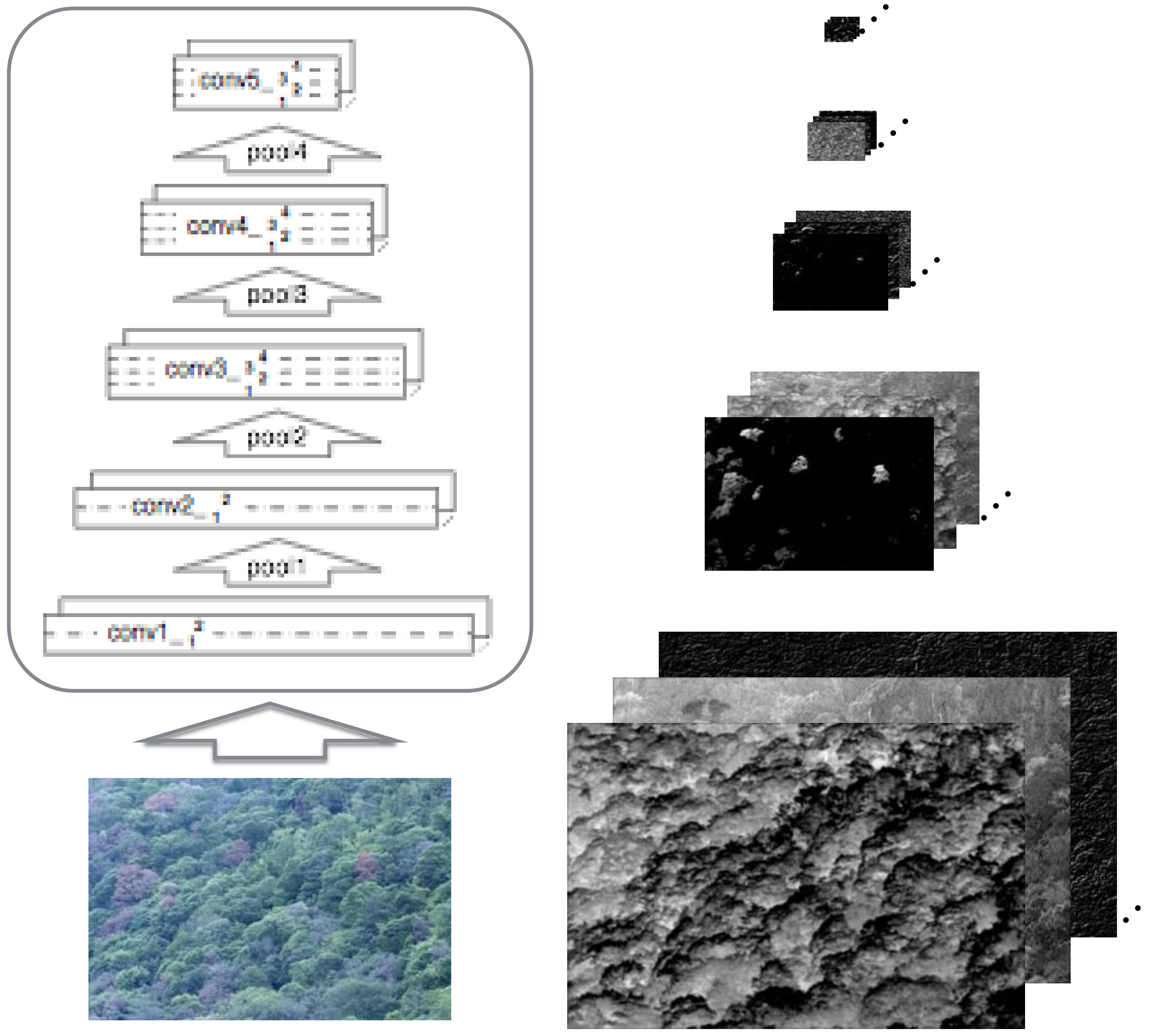


Portilla & Simoncelli (2000)

CNN - Multiscale Filter Bank



CNN - Texture Features



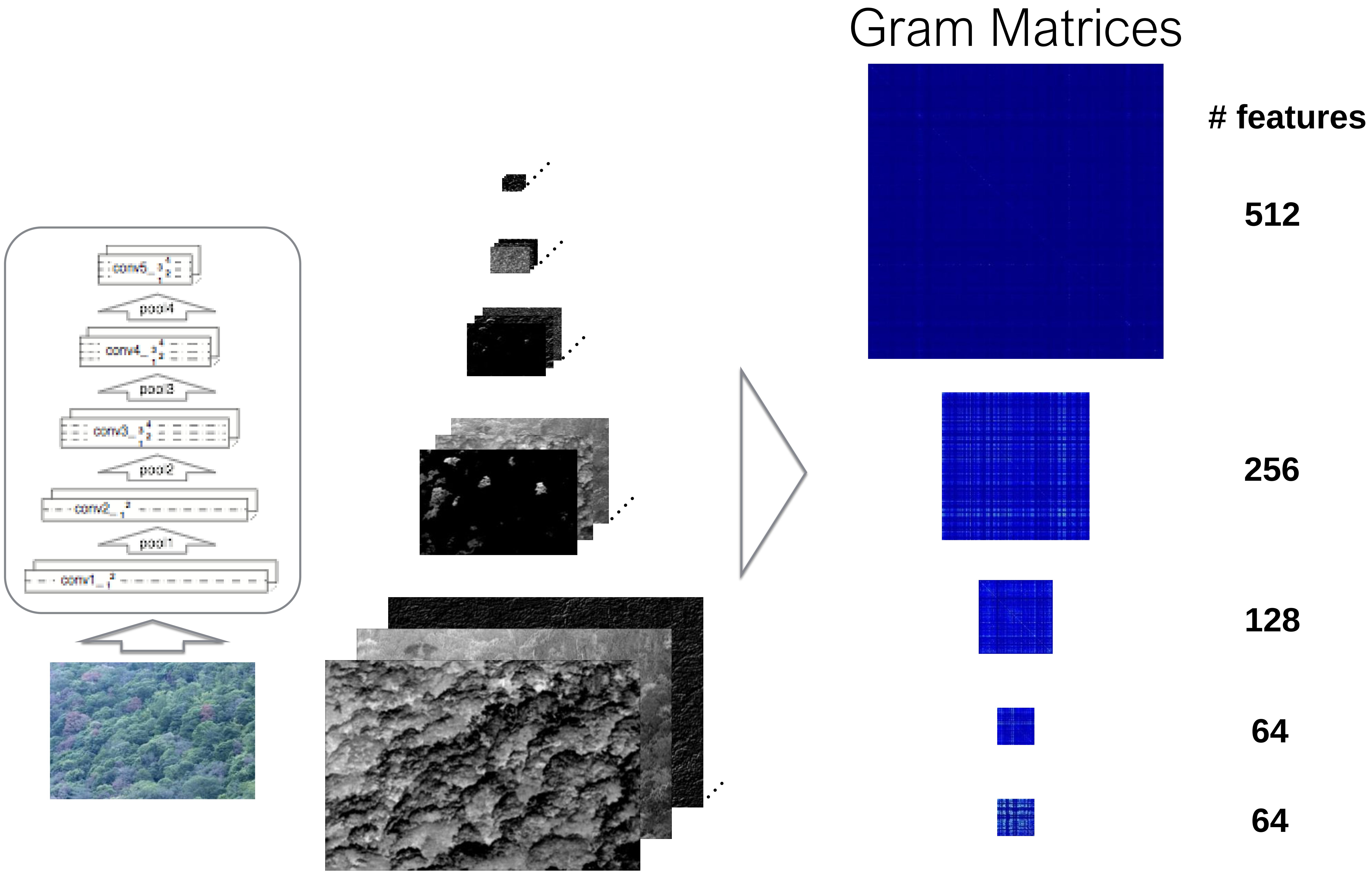
$$F = [\bar{f}_1, \bar{f}_2, \bar{f}_3, ..., \bar{f}_N]^T$$

$$G = FF^T$$

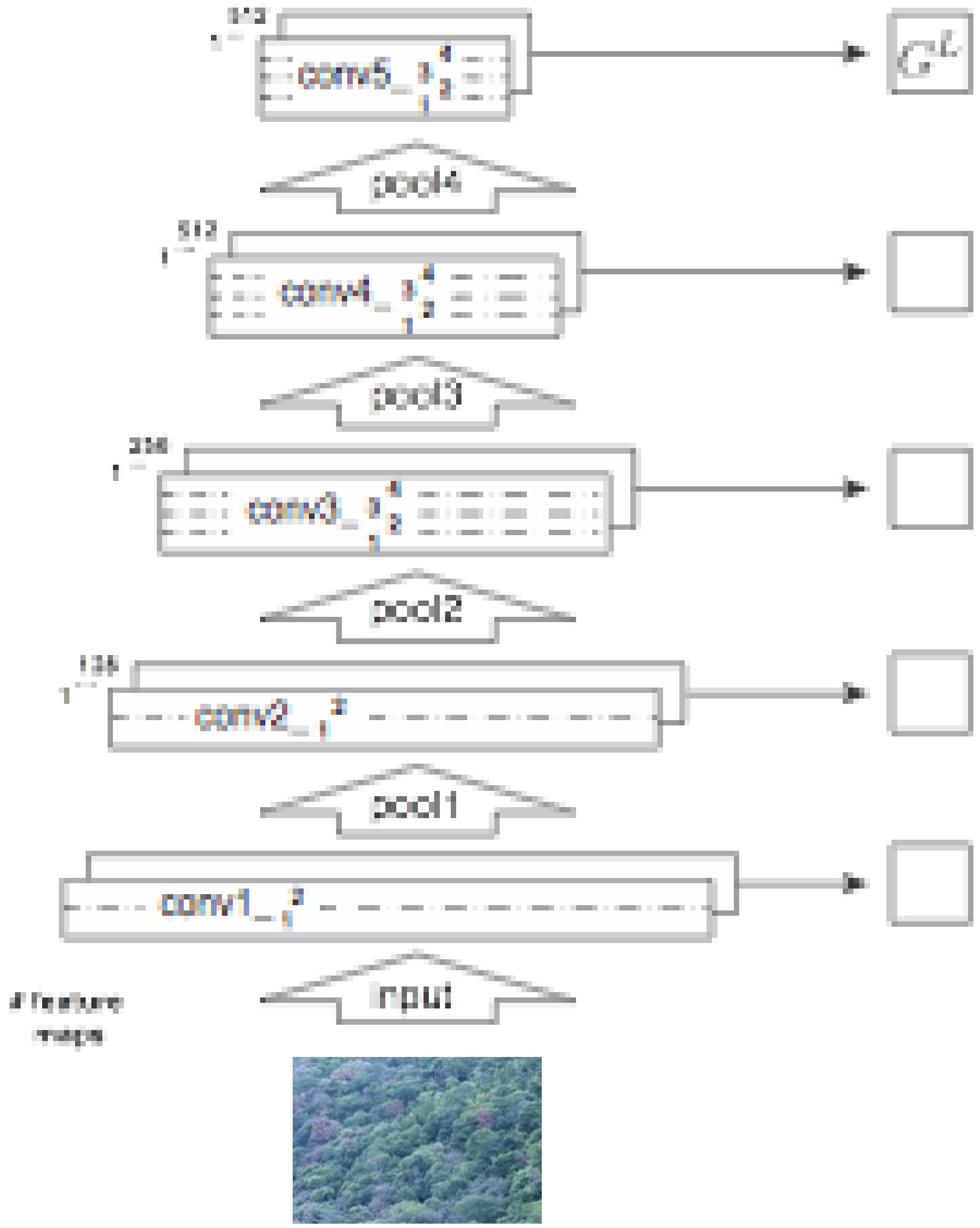
$$= \begin{pmatrix} \langle \bar{f}_1, \bar{f}_1 \rangle & \cdots & \langle \bar{f}_1, \bar{f}_N \rangle \\ \langle \bar{f}_2, \bar{f}_1 \rangle & & \vdots \\ \vdots & \ddots & \vdots \\ \langle \bar{f}_N, \bar{f}_1 \rangle & \cdots & \langle \bar{f}_N, \bar{f}_N \rangle \end{pmatrix}$$

$$\langle \bar{f}_i, \bar{f}_j \rangle = \sum_k F_{ik} F_{jk}$$

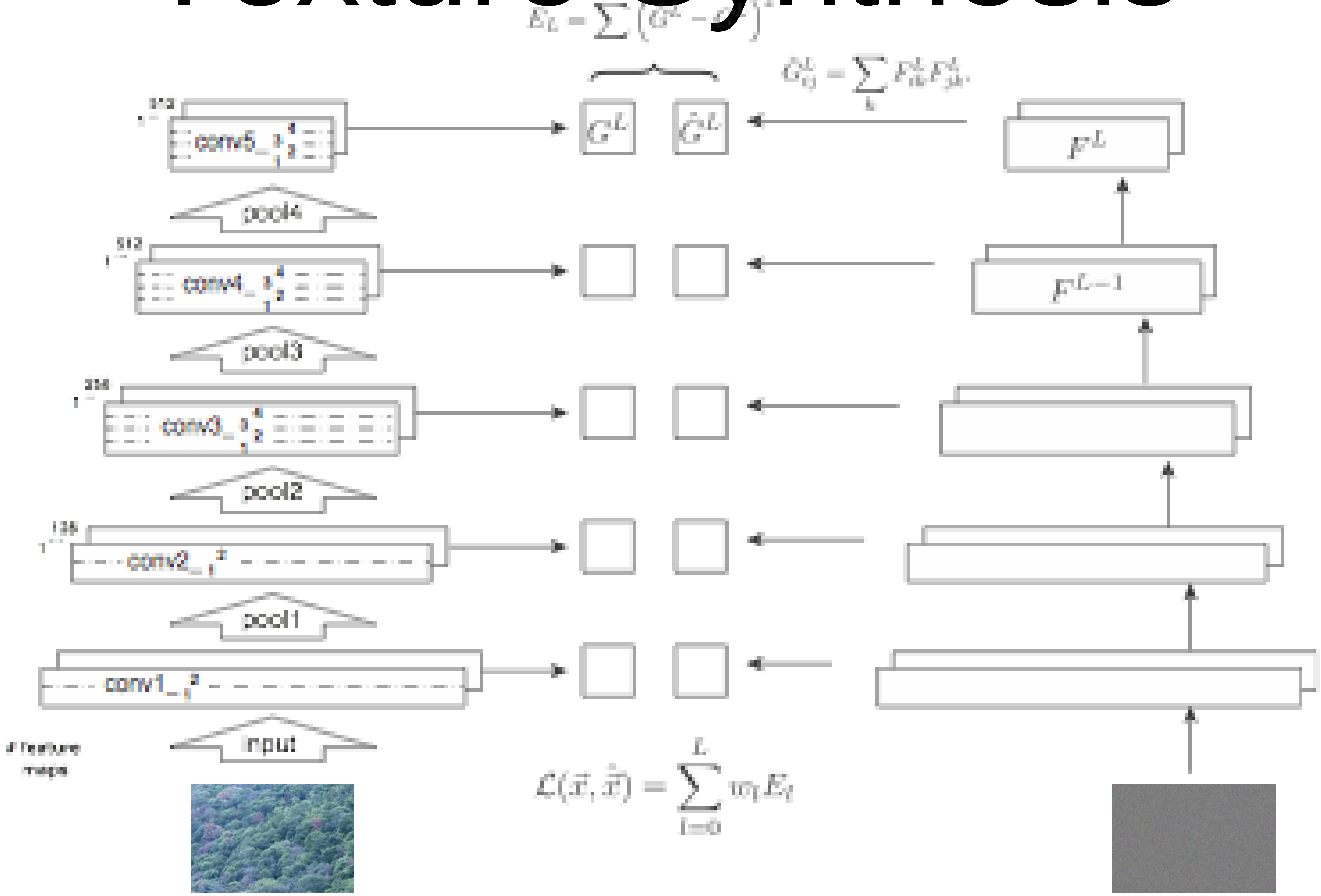
CNN-Texture Features



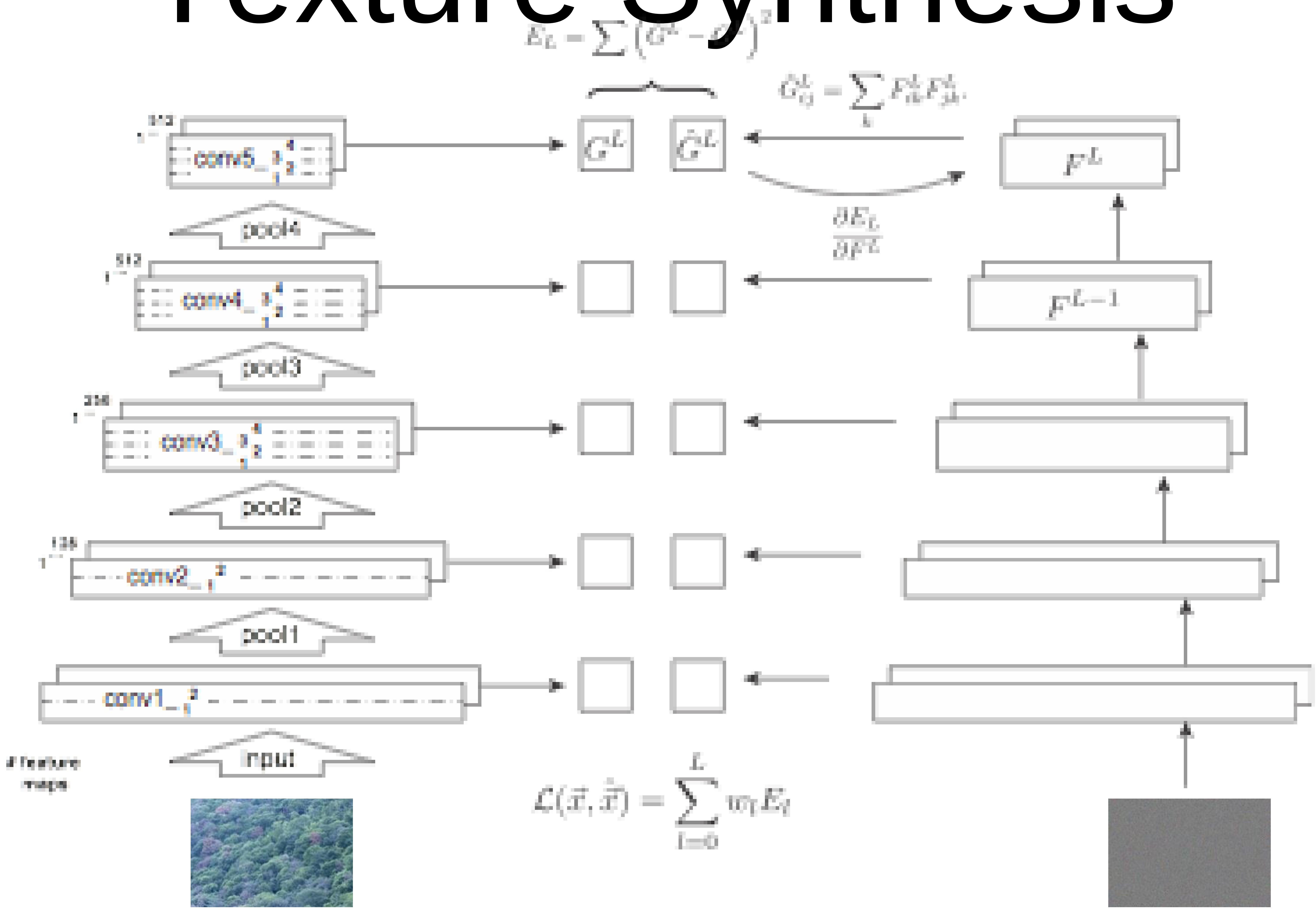
Texture Synthesis



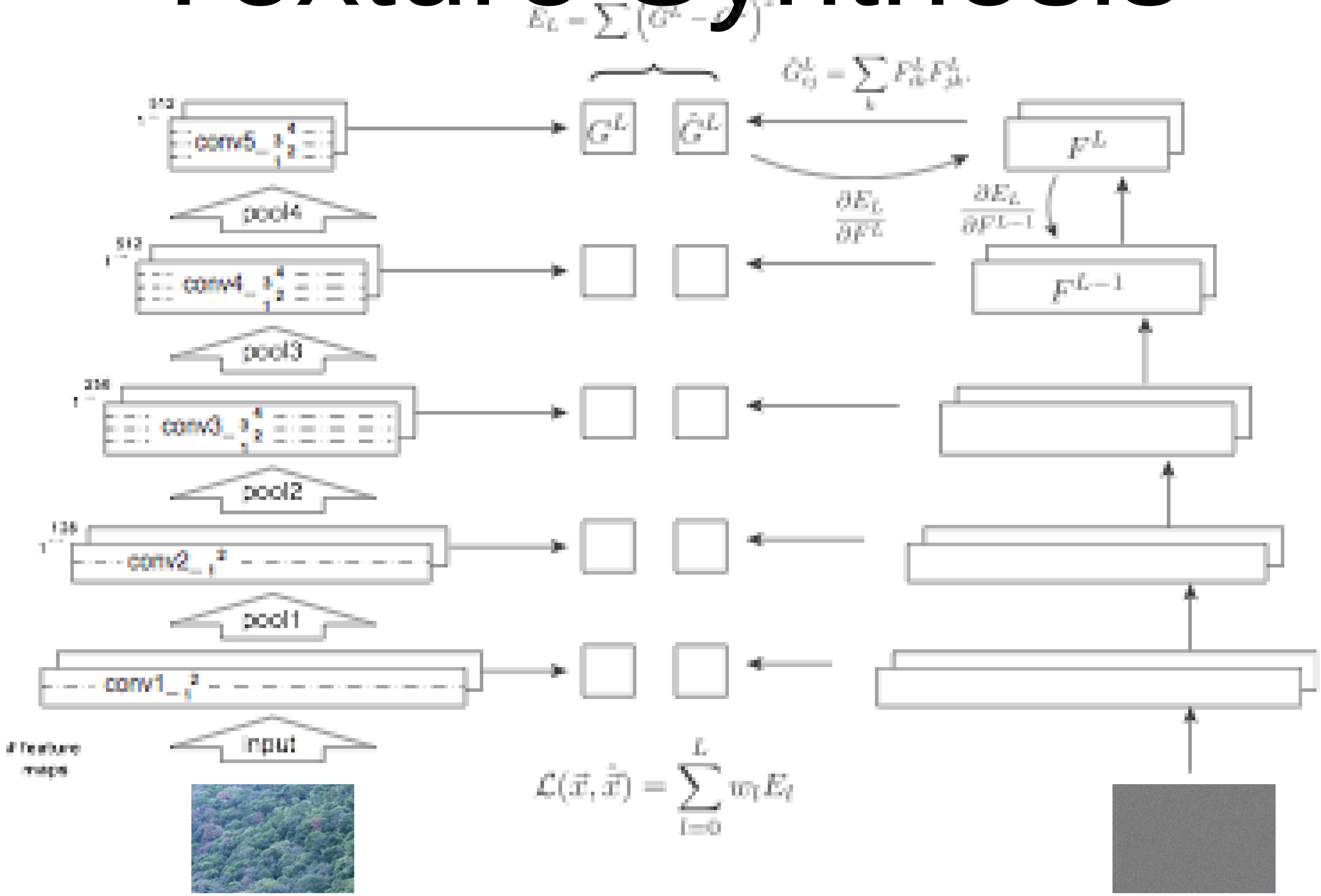
Texture Synthesis



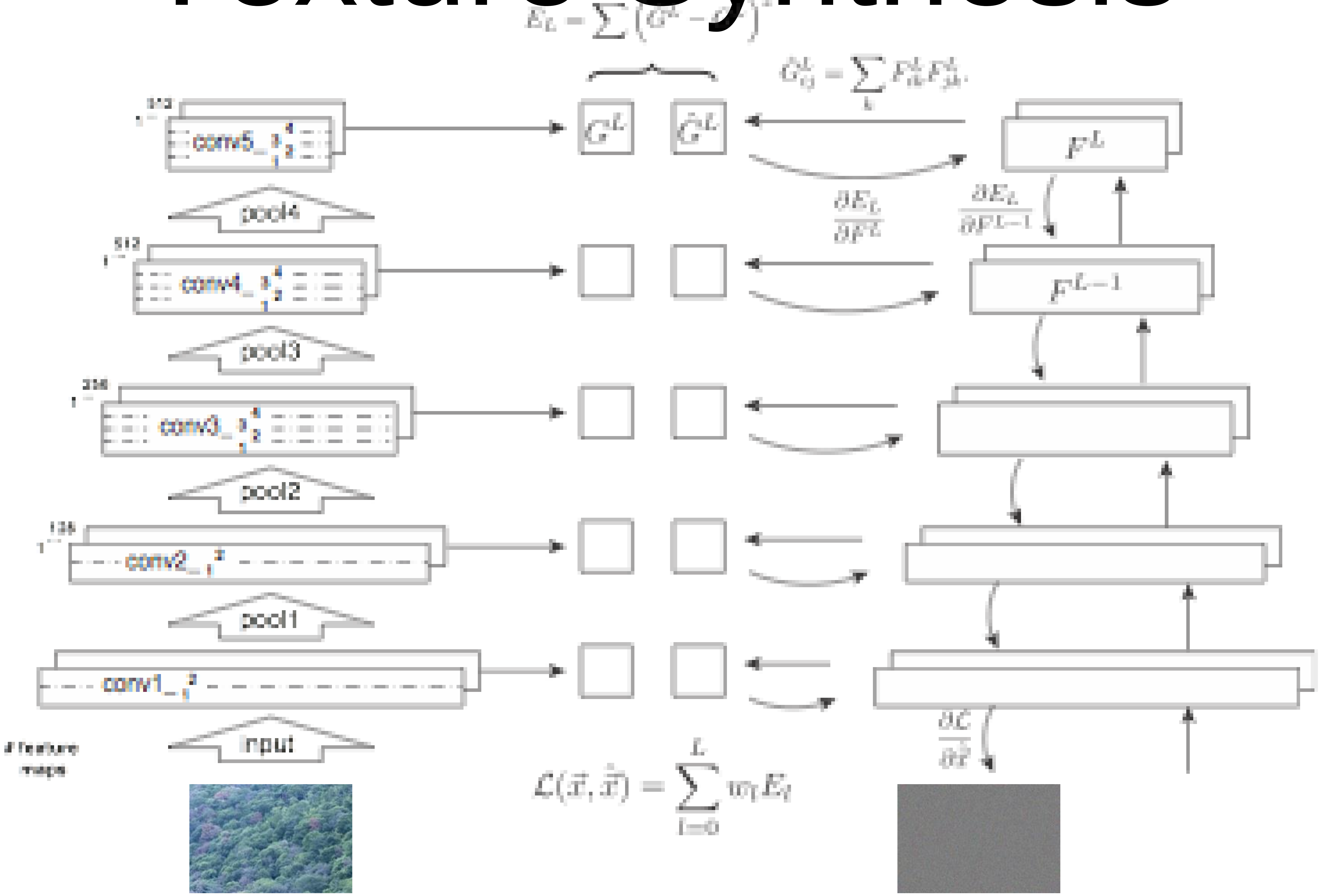
Texture Synthesis



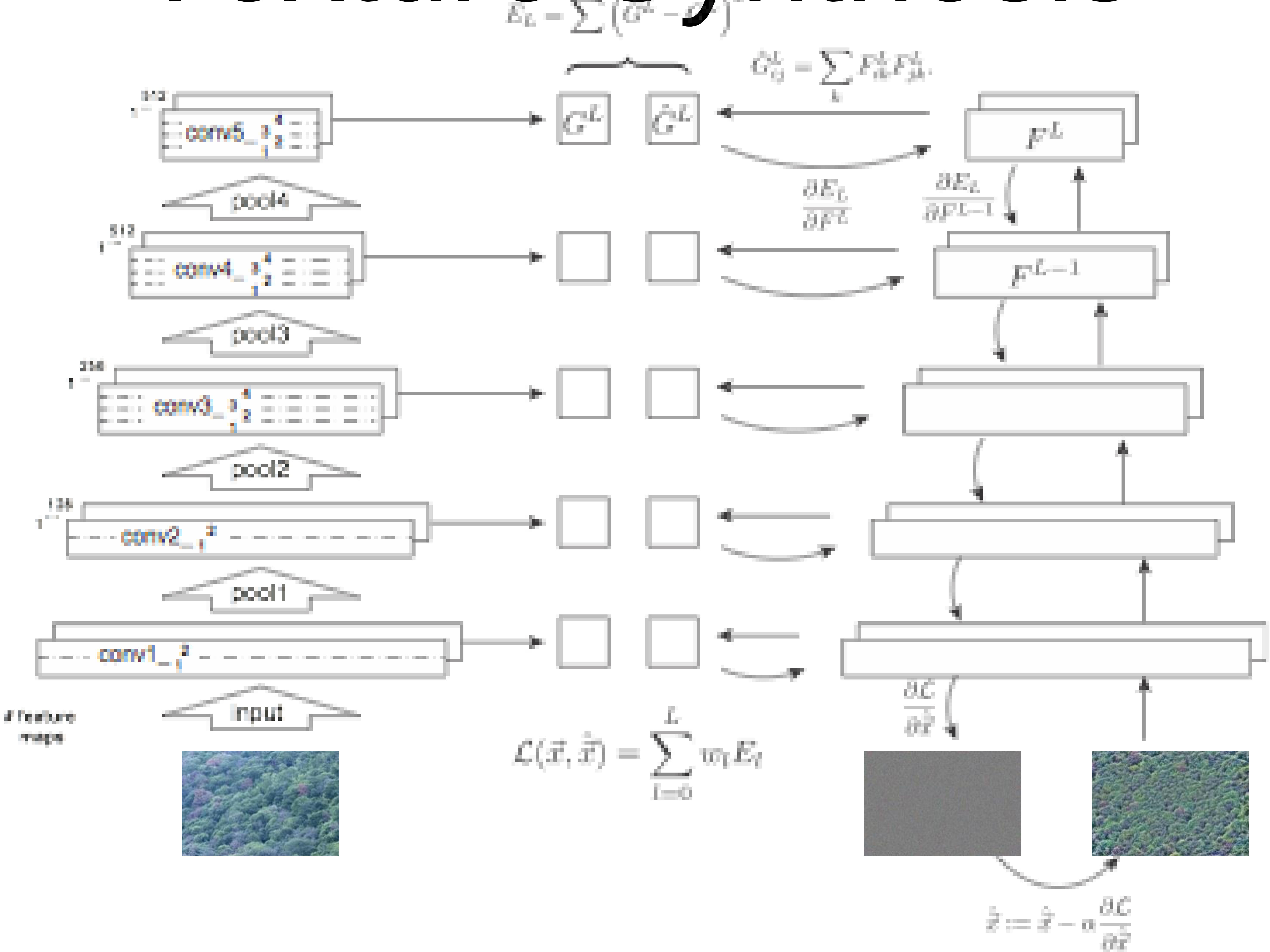
Texture Synthesis



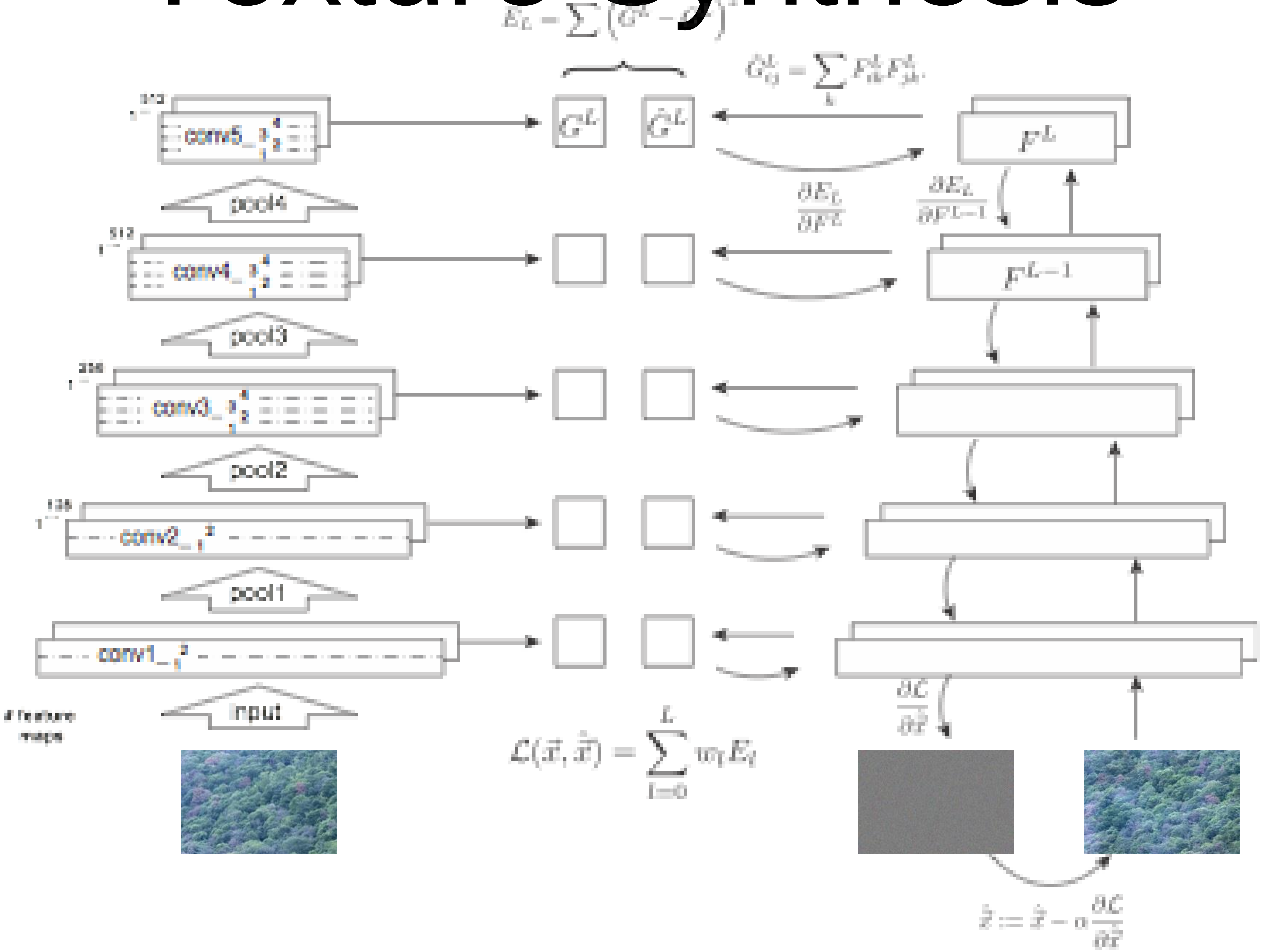
Texture Synthesis



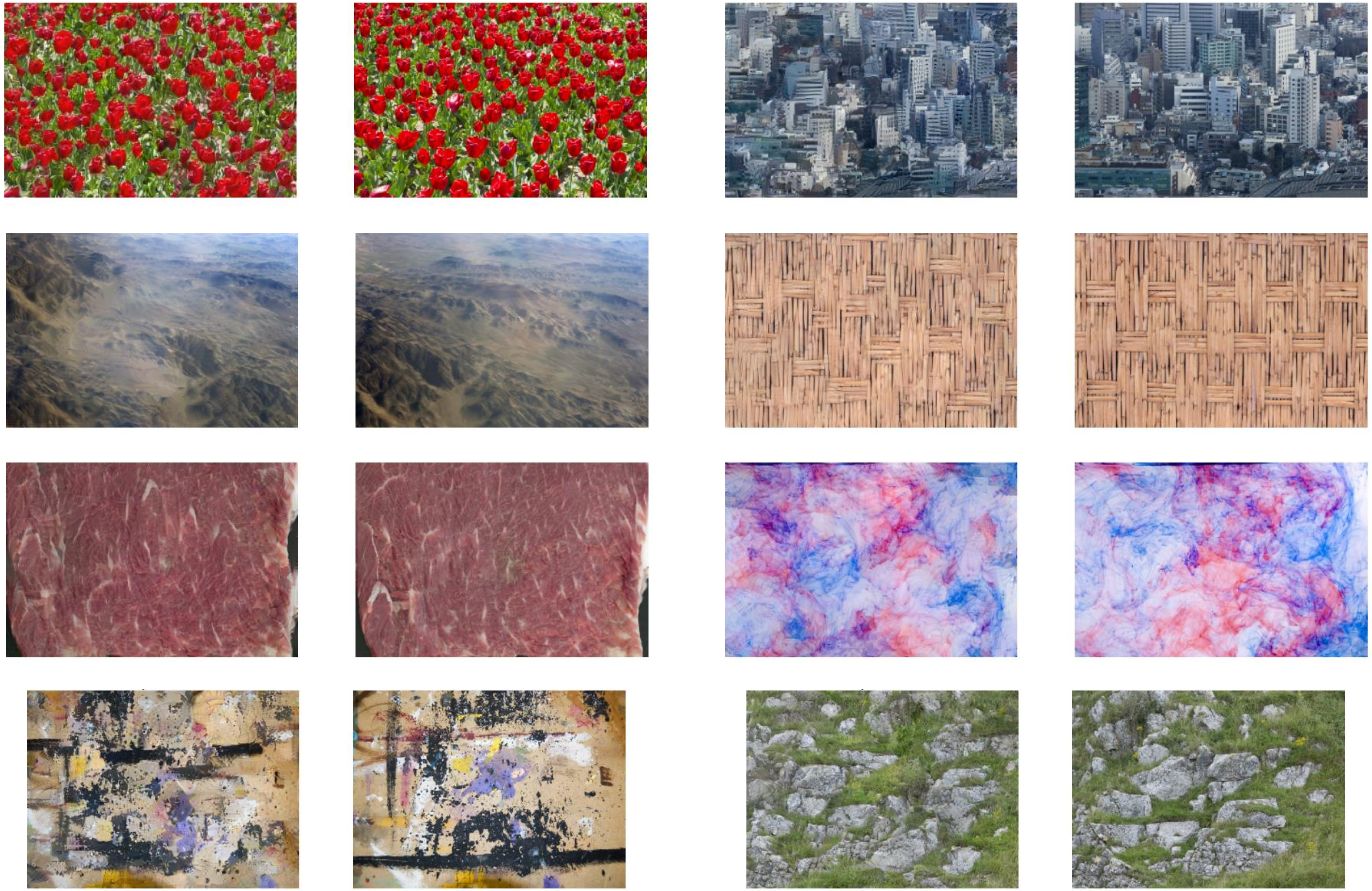
Texture Synthesis



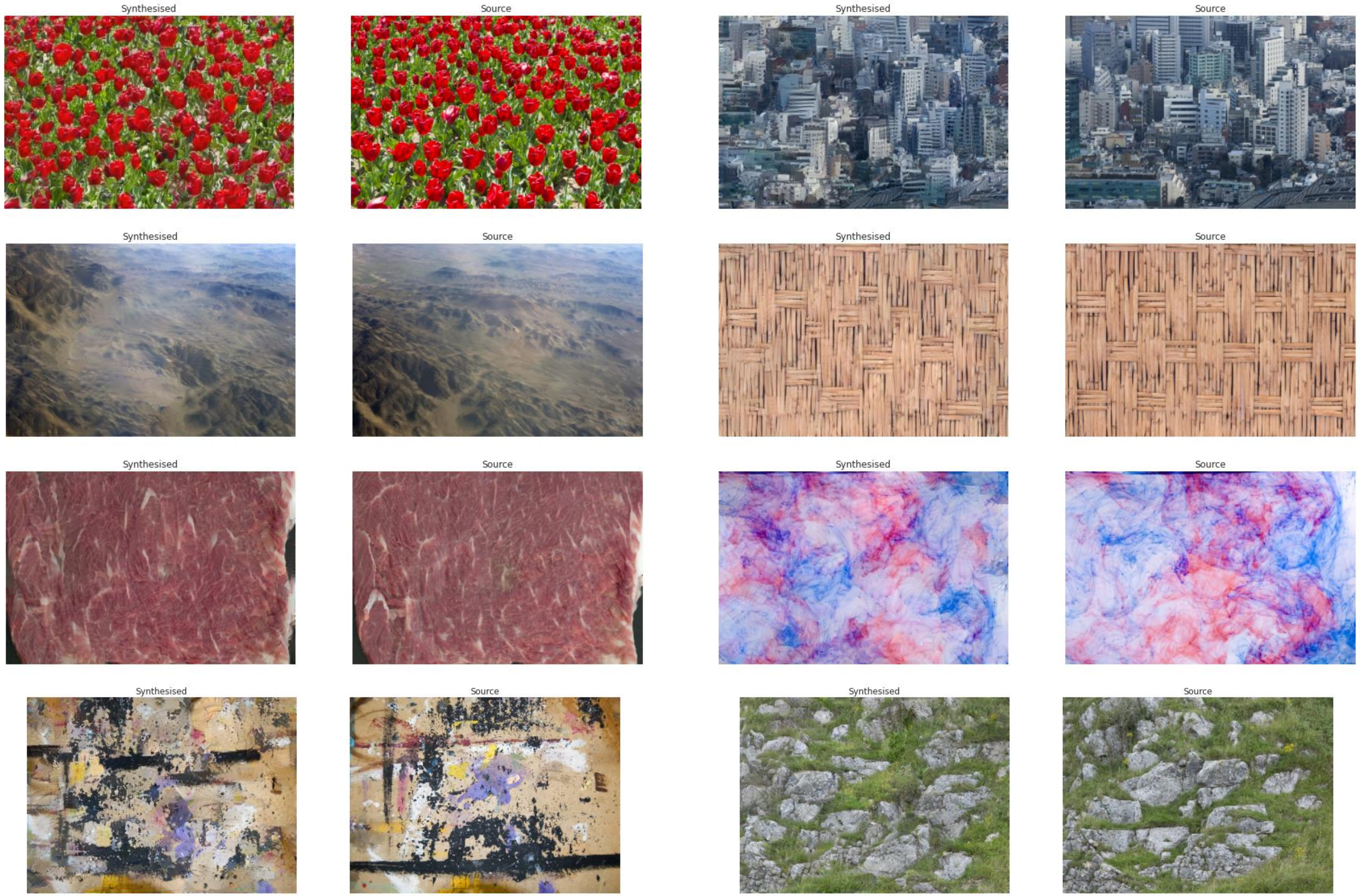
Texture Synthesis



Test Julesz' Conjecture



Test Julesz' Conjecture



ImageNet Recognition is just Texture Recognition

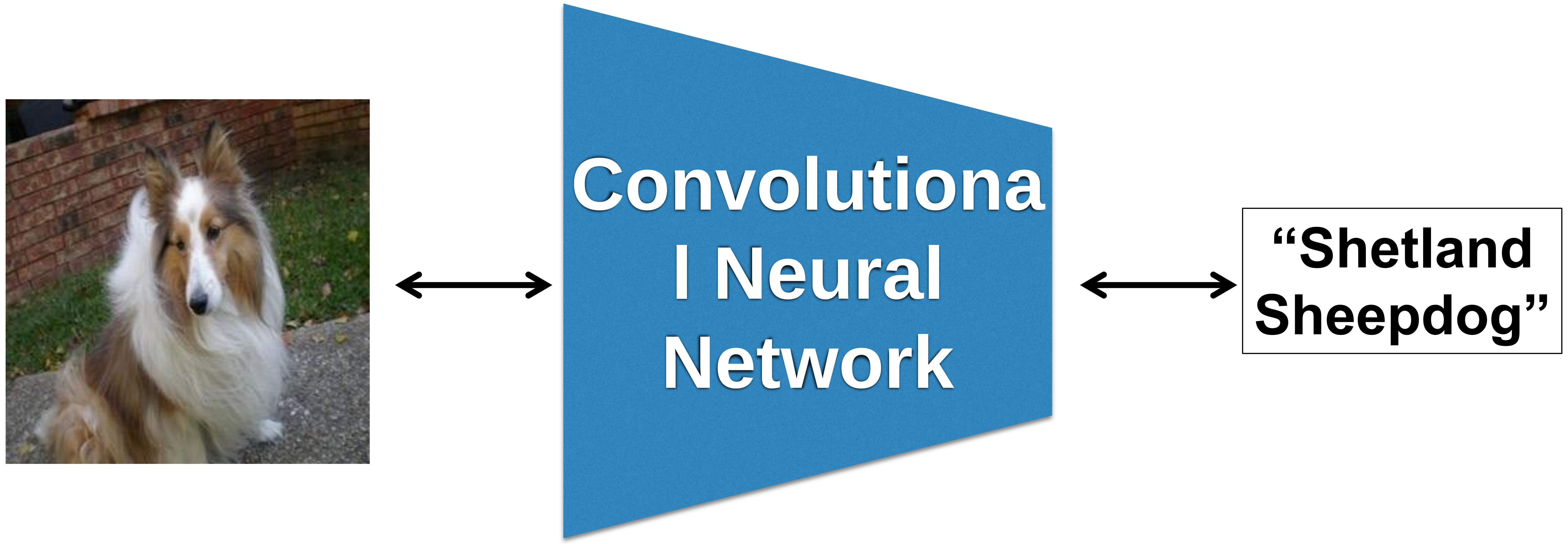
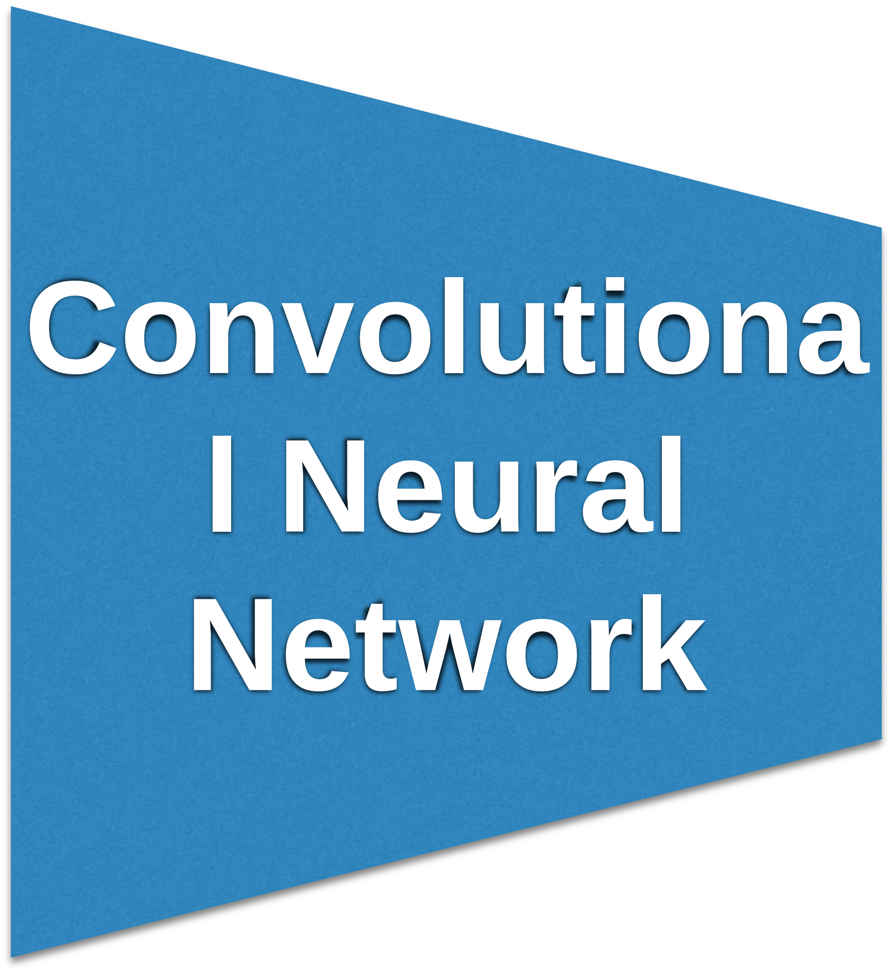


image *X*

label *Y*

ImageNet Recognition is just Texture Recognition



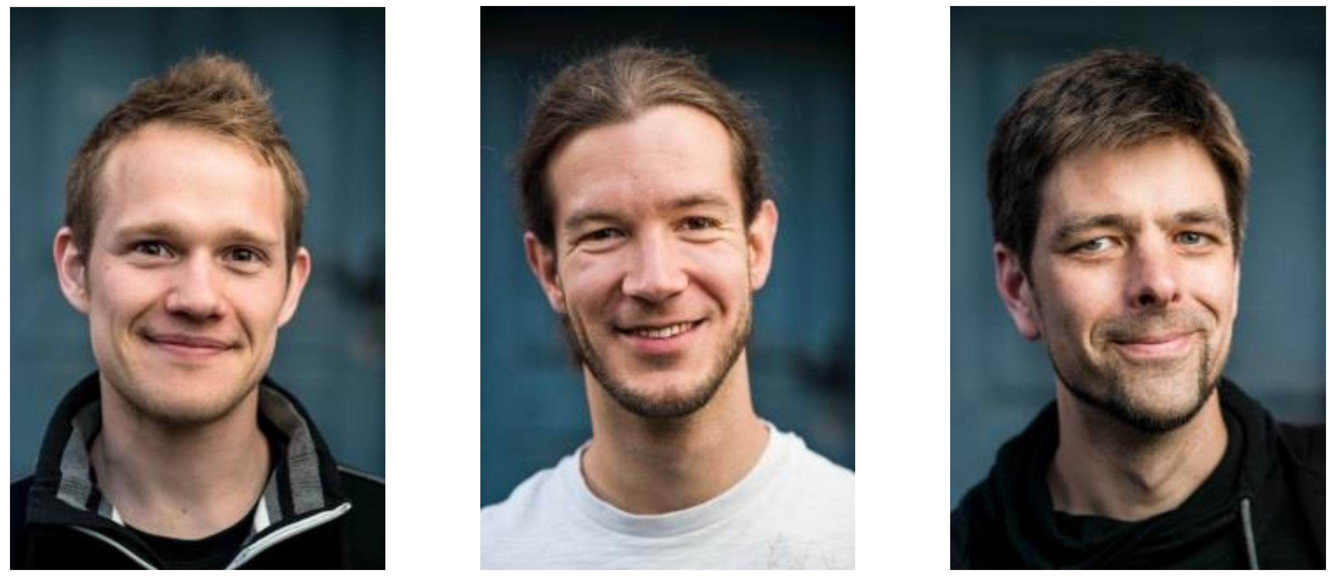
“Shetland Sheepdog”

image X

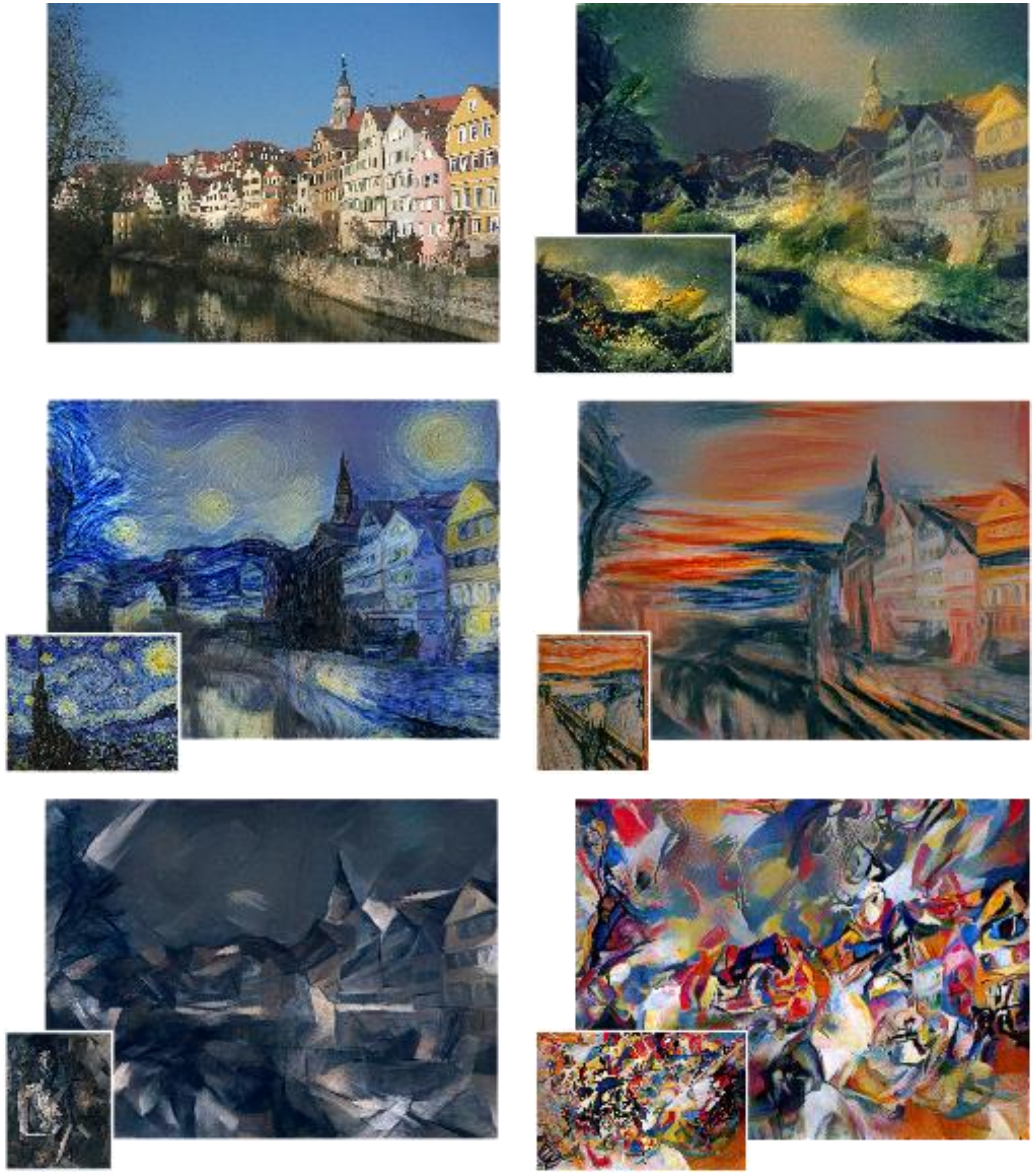
label Y

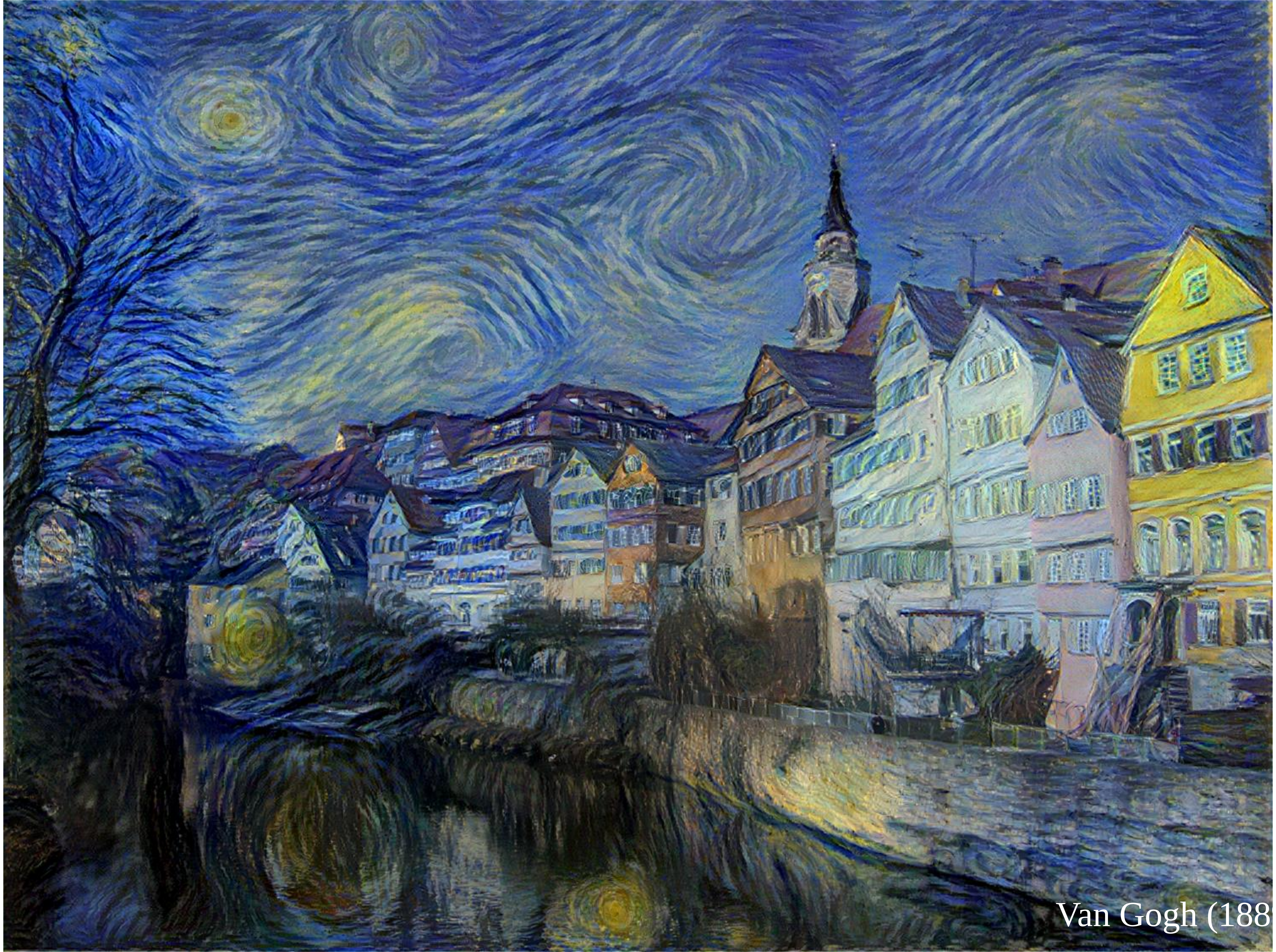
Gatys et al, 2017

A Neural Algorithm of Artistic Style



Gatys, Ecker, Bethge (arXiv 2015)





Van Gogh (188

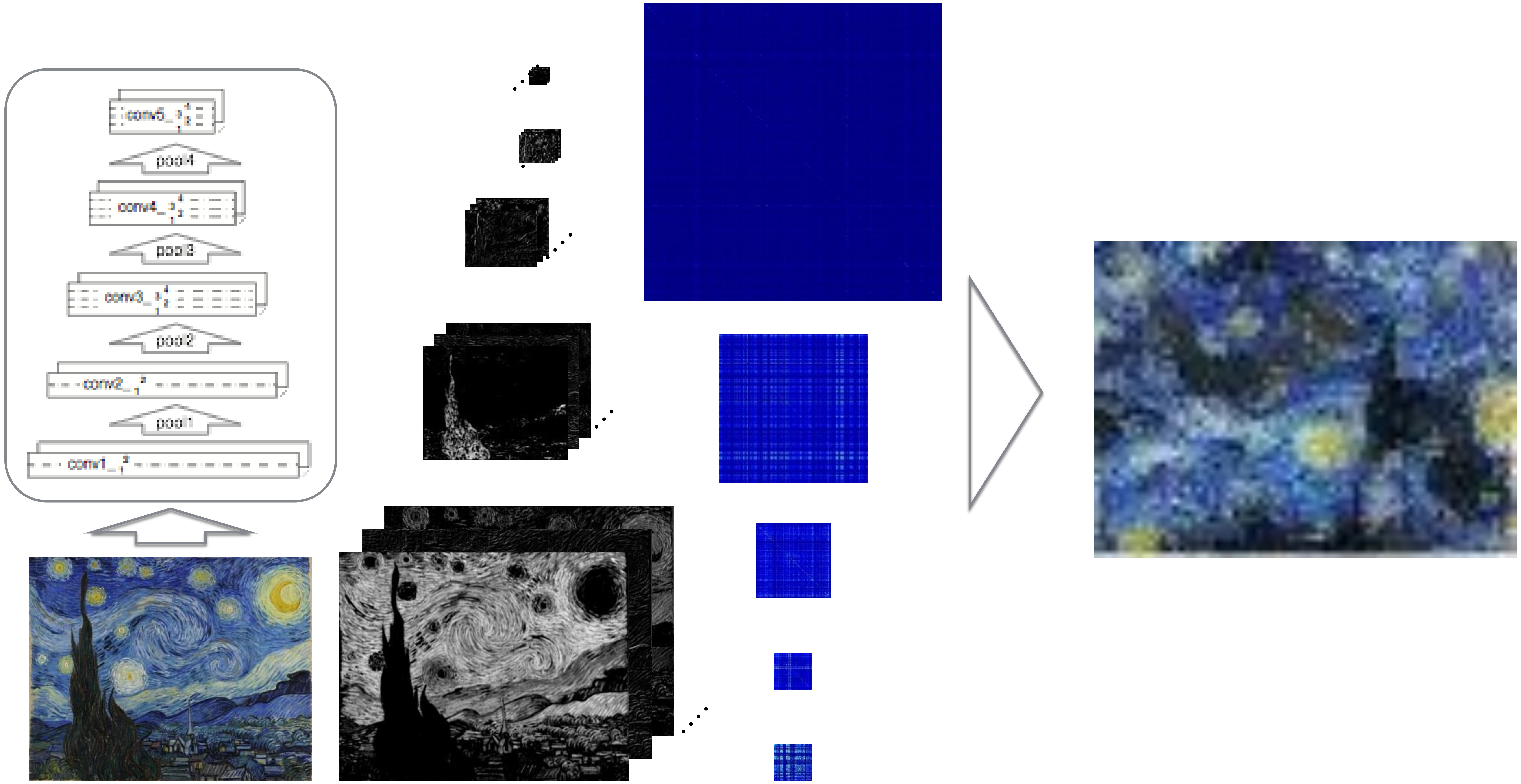

Picasso (1910)


Munch (1893)


Turner (1805)

Kandinsky (191

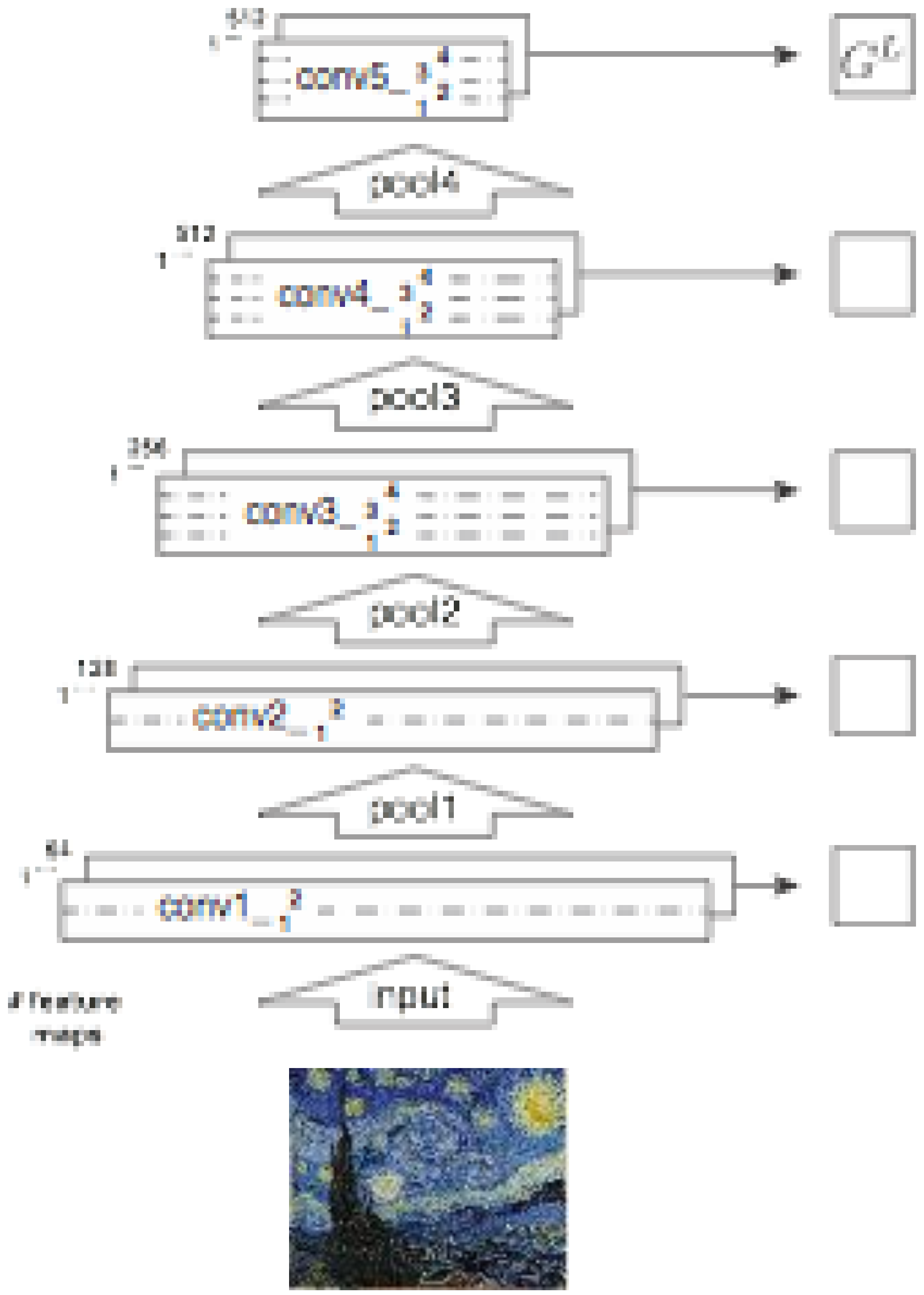
CNN - Texture Synthesis



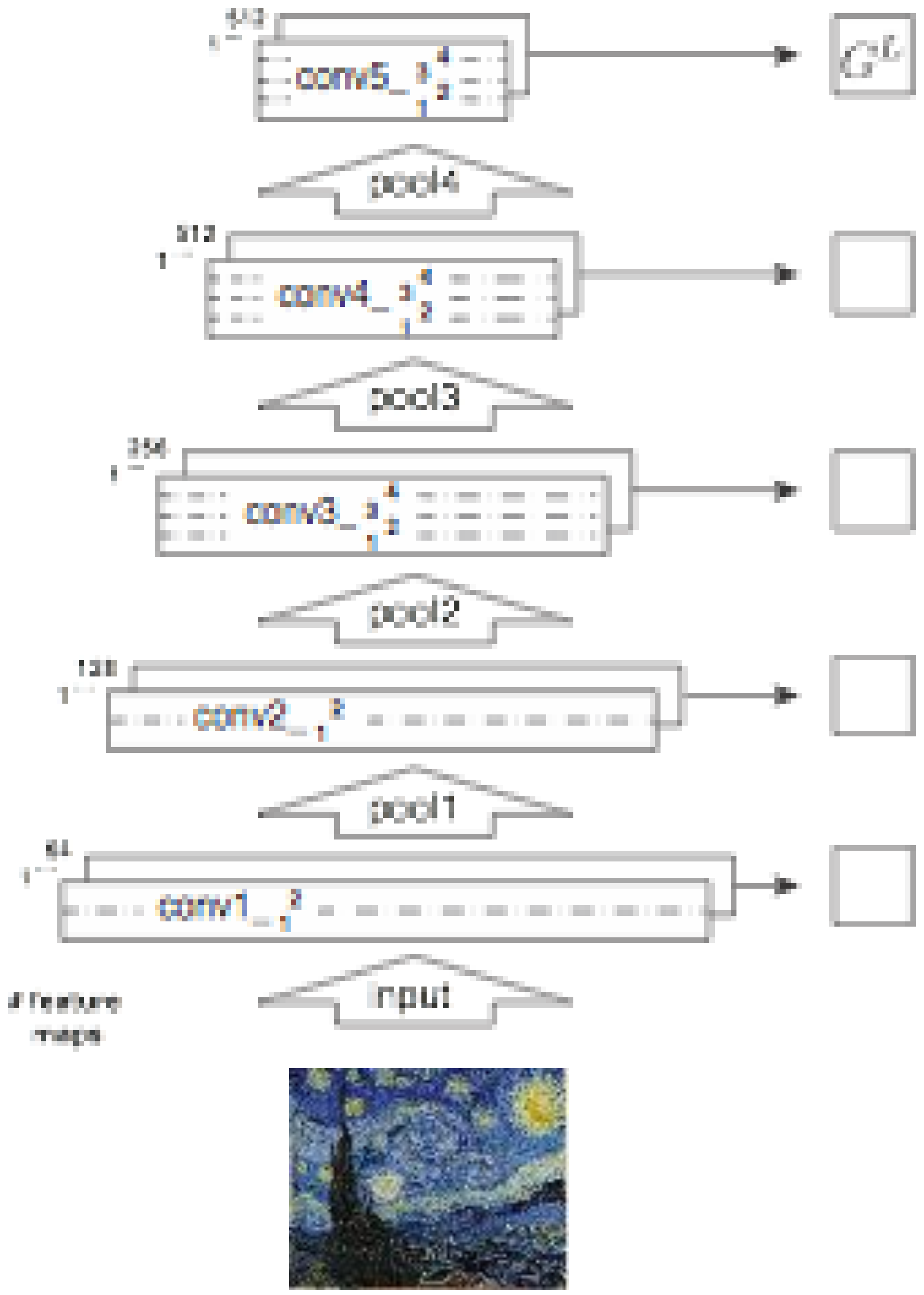
Gatys et al. (NIPS 2015)

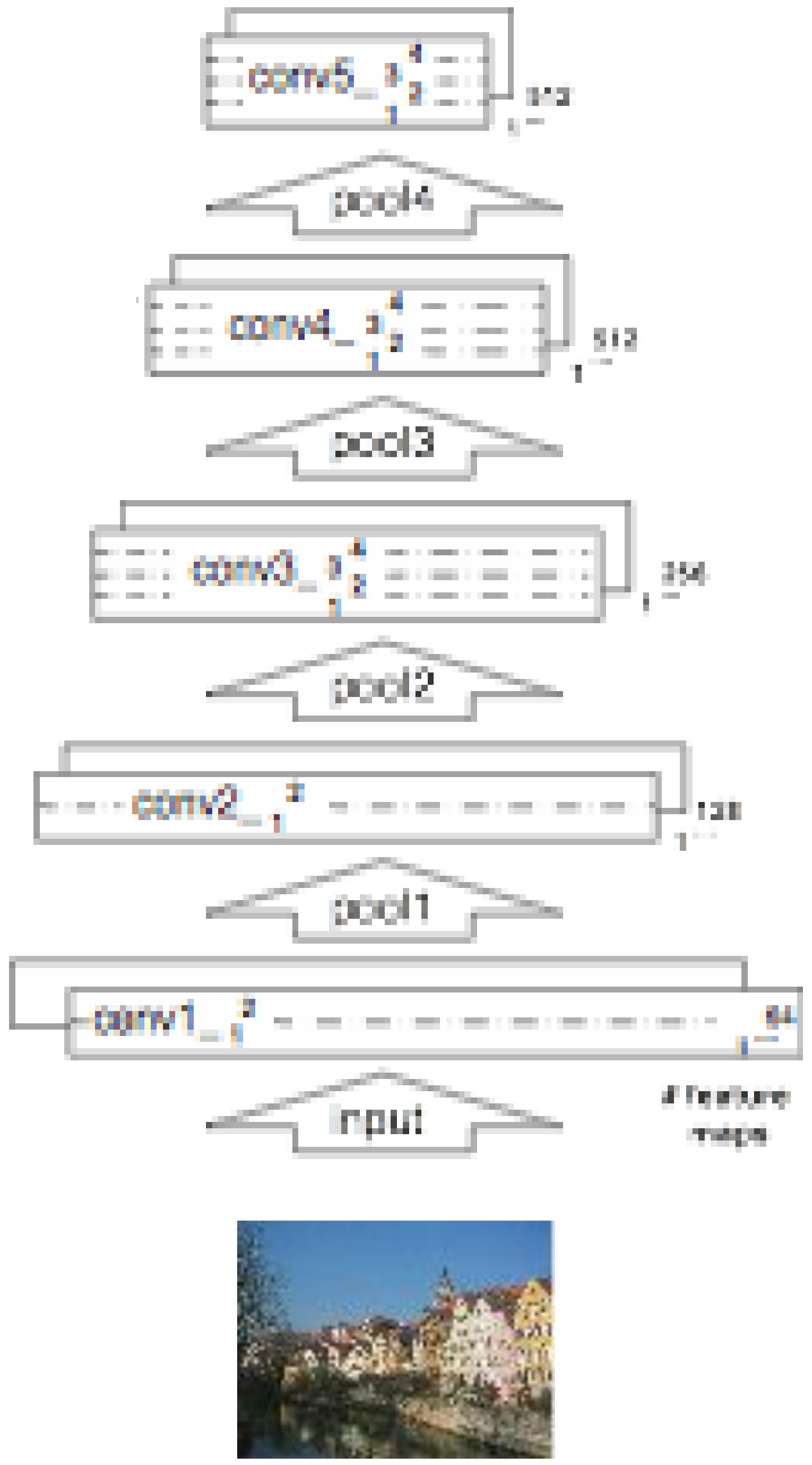
Artistic Style Transfer

Artistic Style Transfer

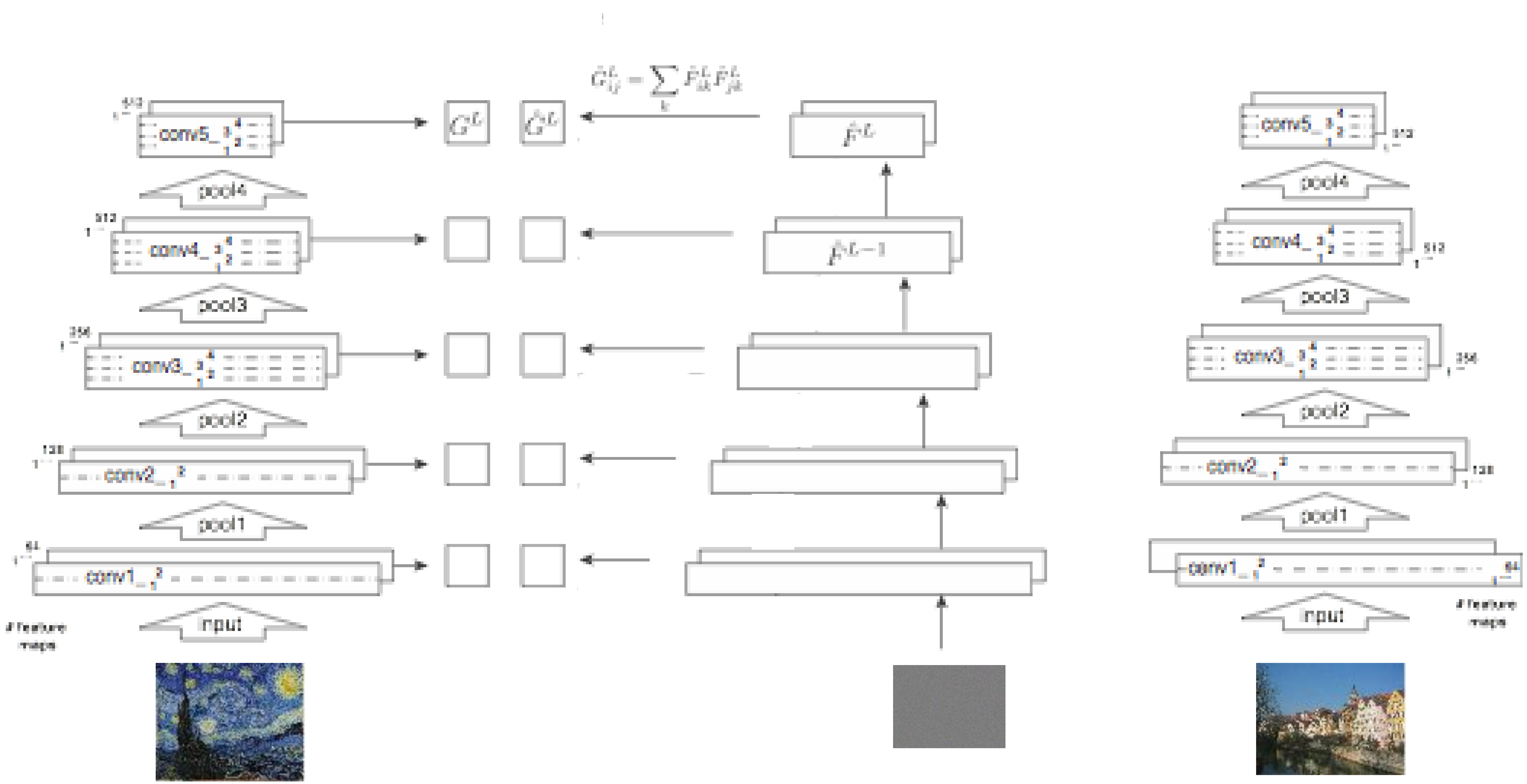


Artistic Style Transfer

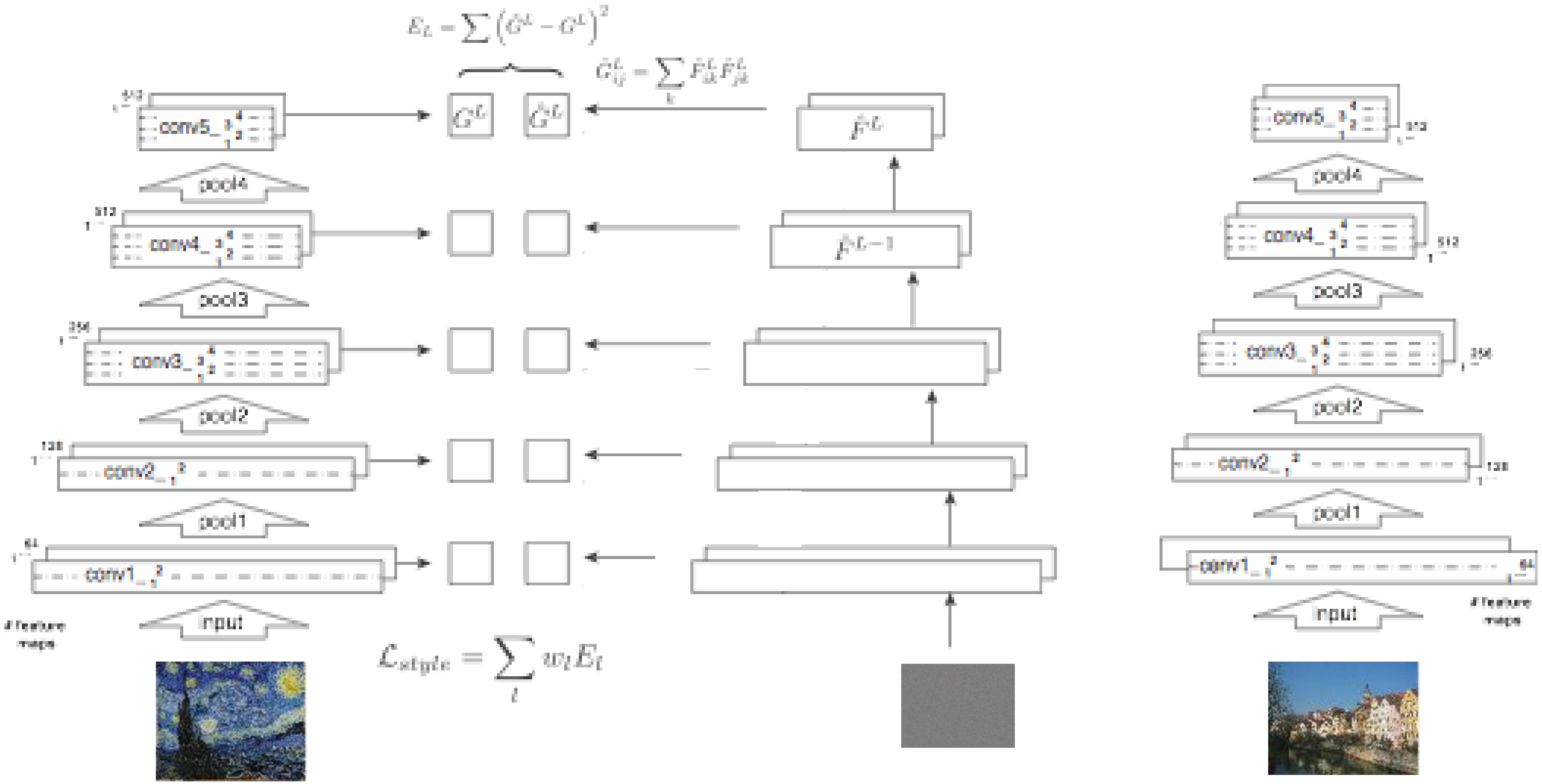




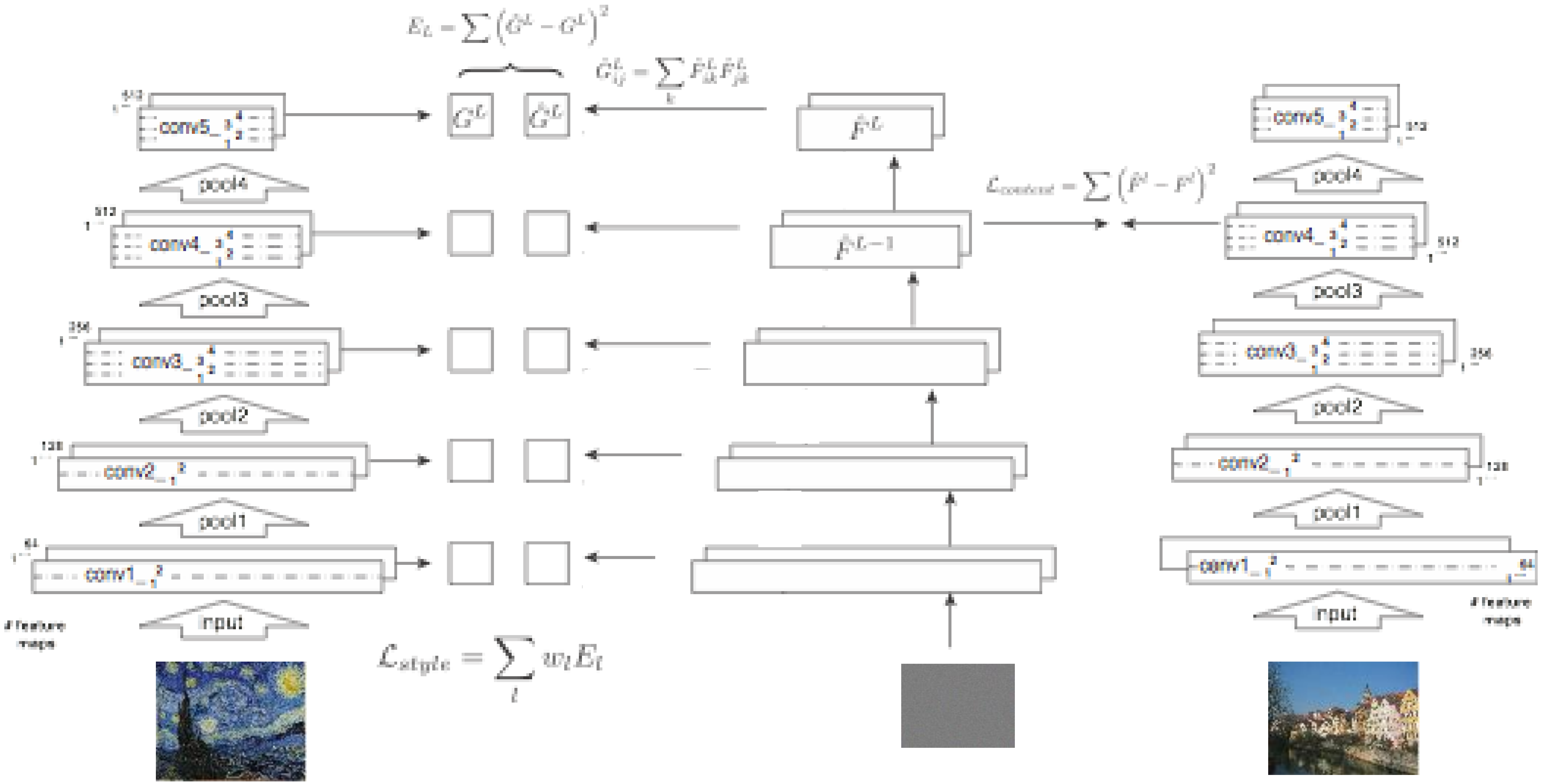
Artistic Style Transfer



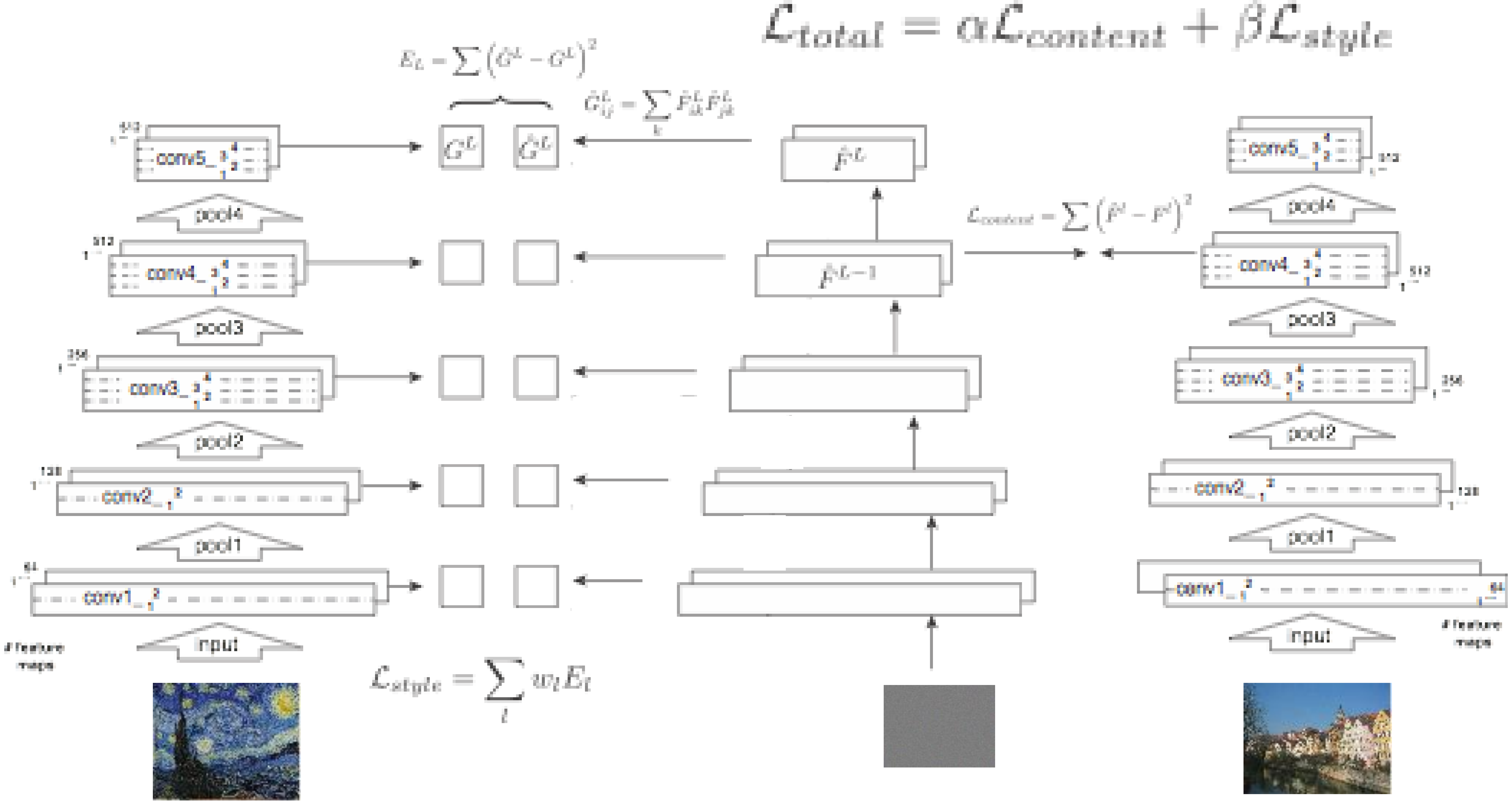
Artistic Style Transfer



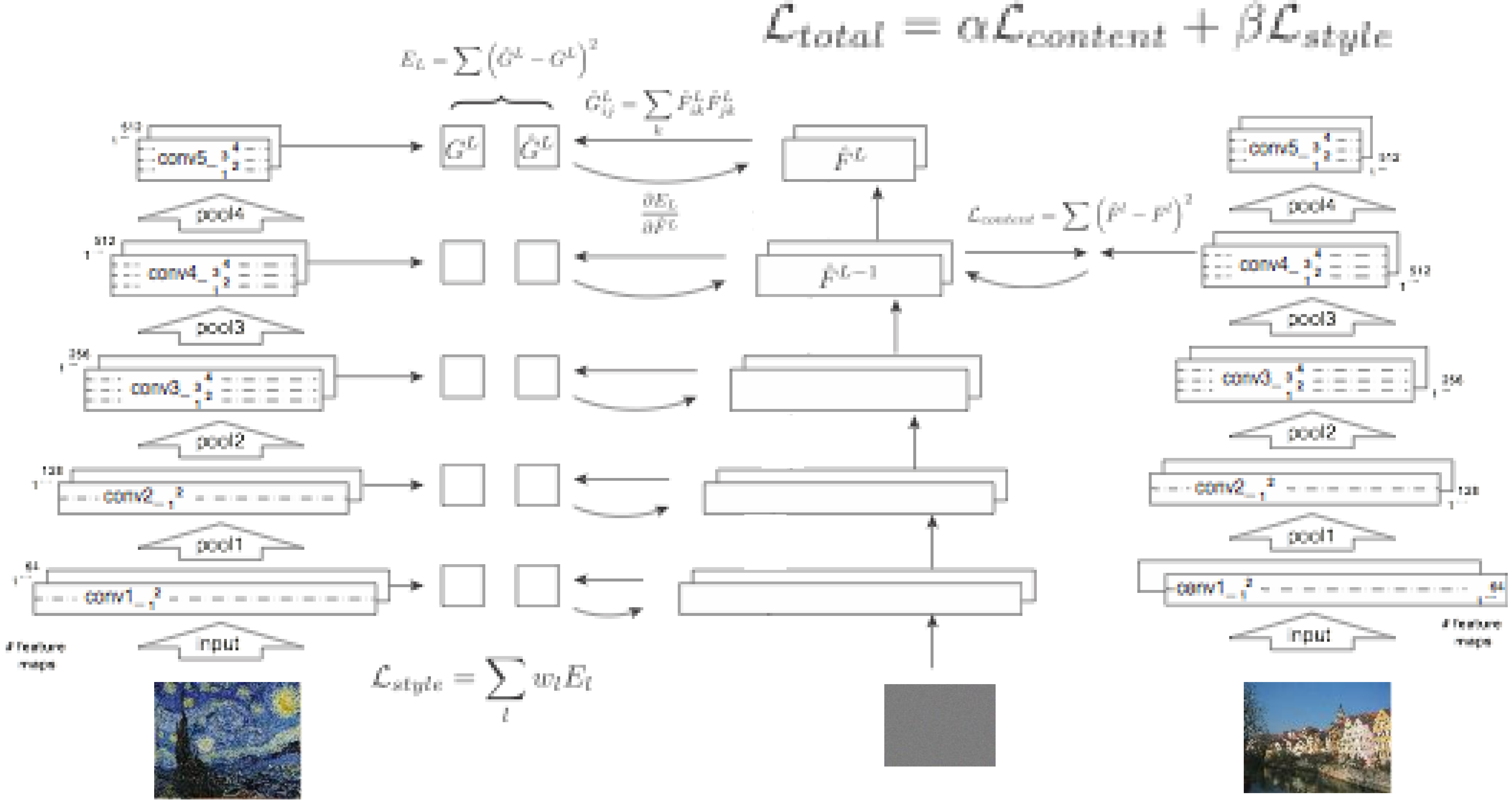
Artistic Style Transfer



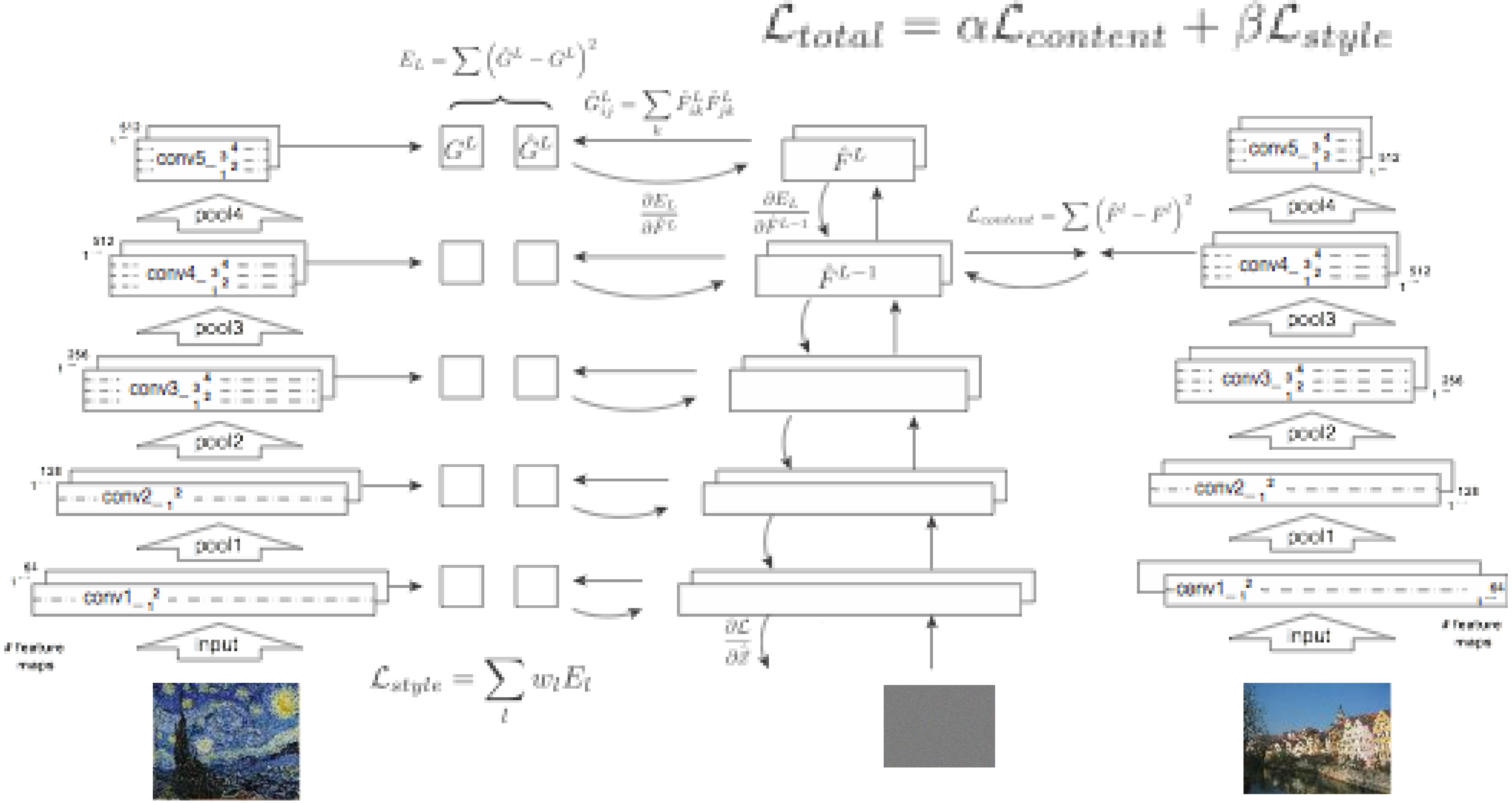
Artistic Style Transfer



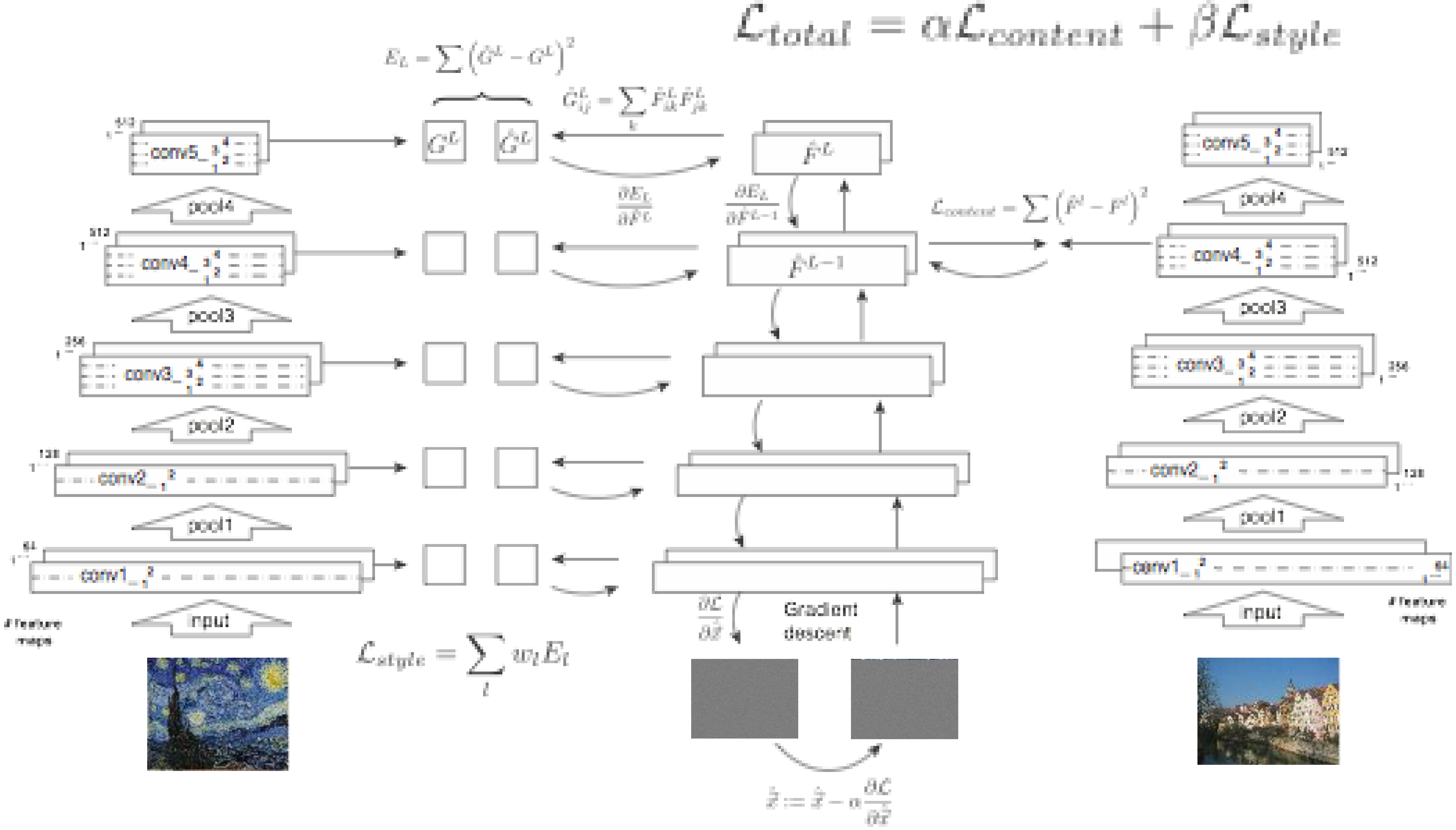
Artistic Style Transfer



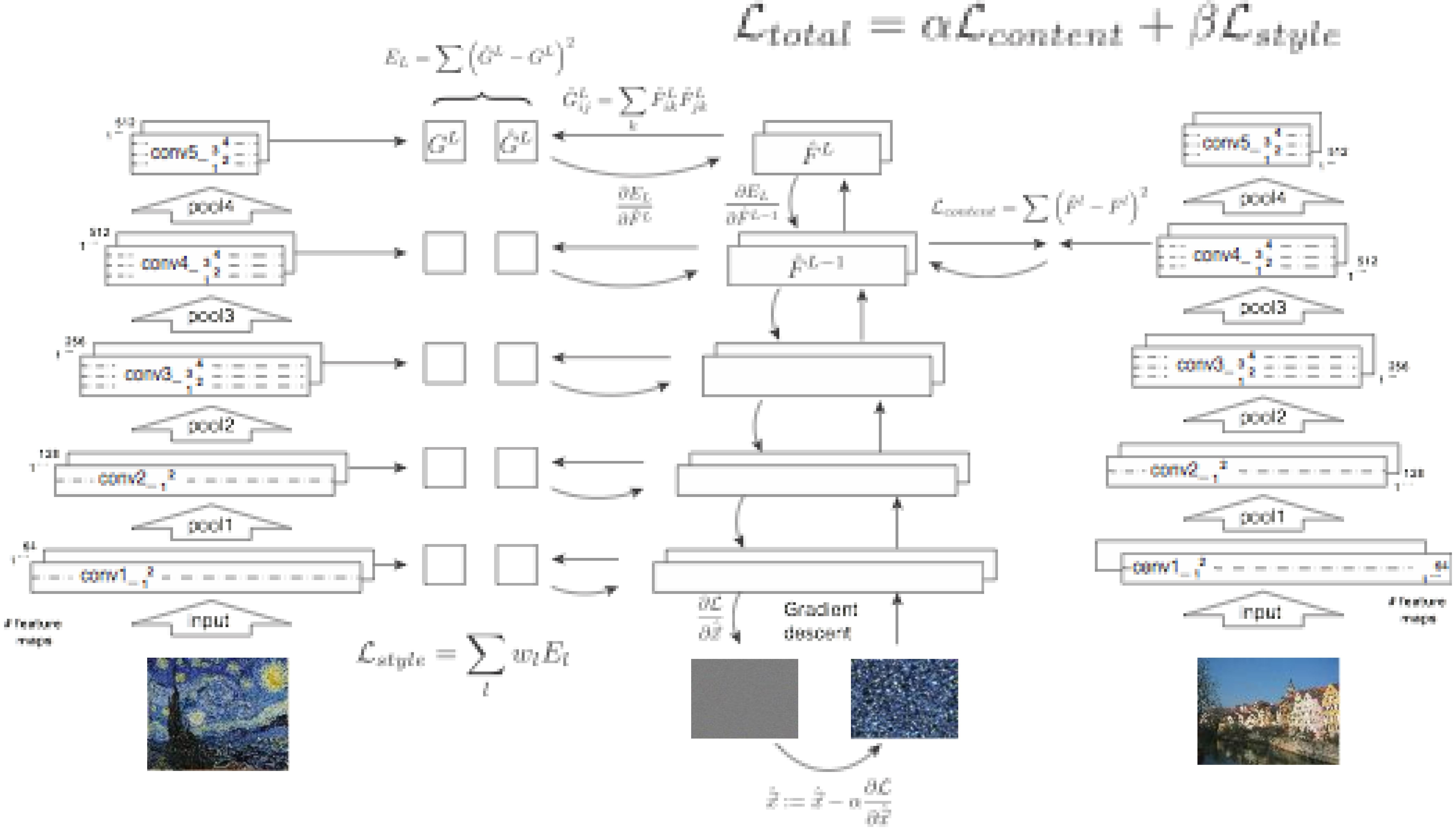
Artistic Style Transfer



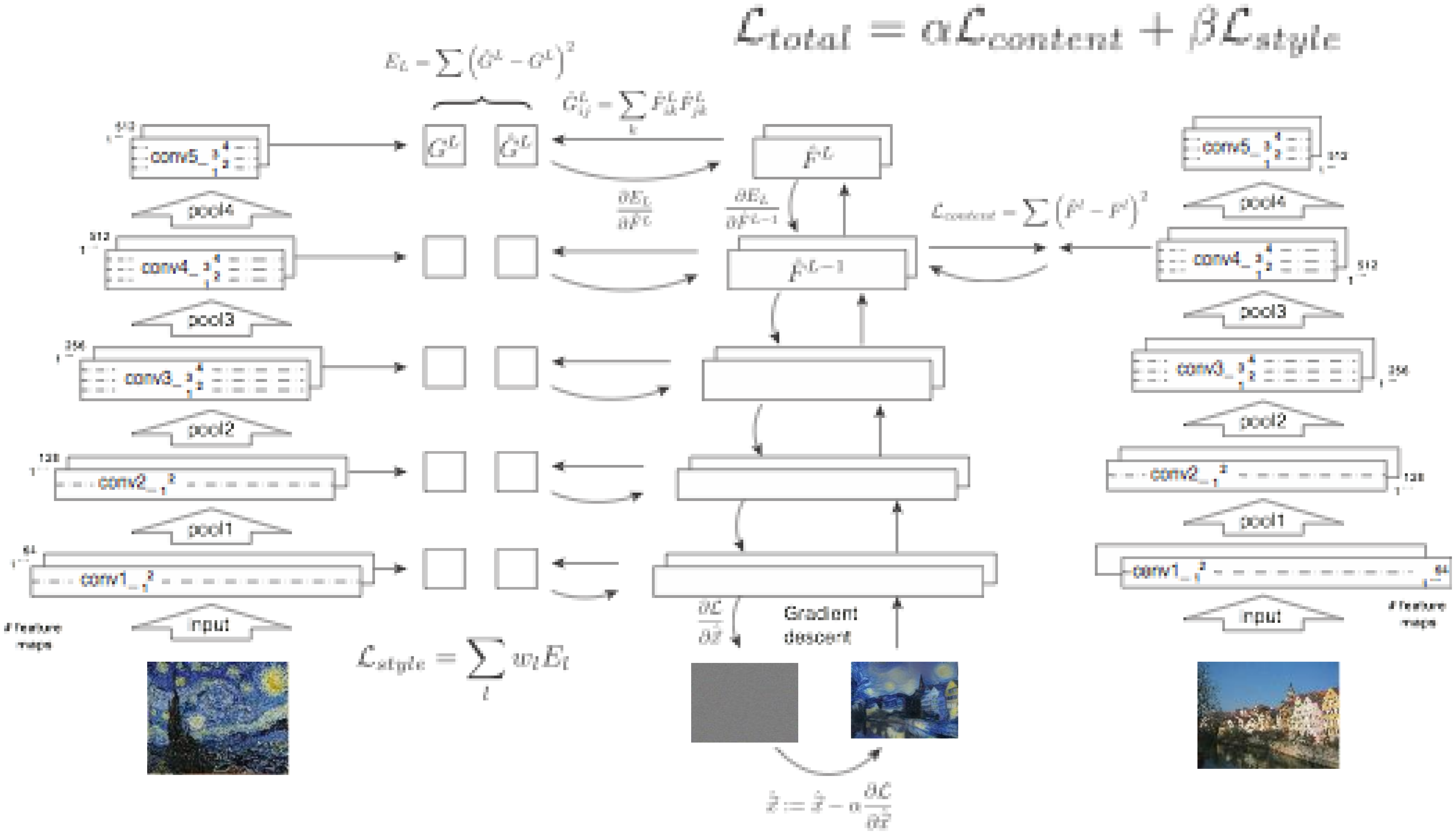
Artistic Style Transfer



Artistic Style Transfer



Artistic Style Transfer



Relative Weighting of Content and Style

1e-4

1e-3

1e-2

1e-1

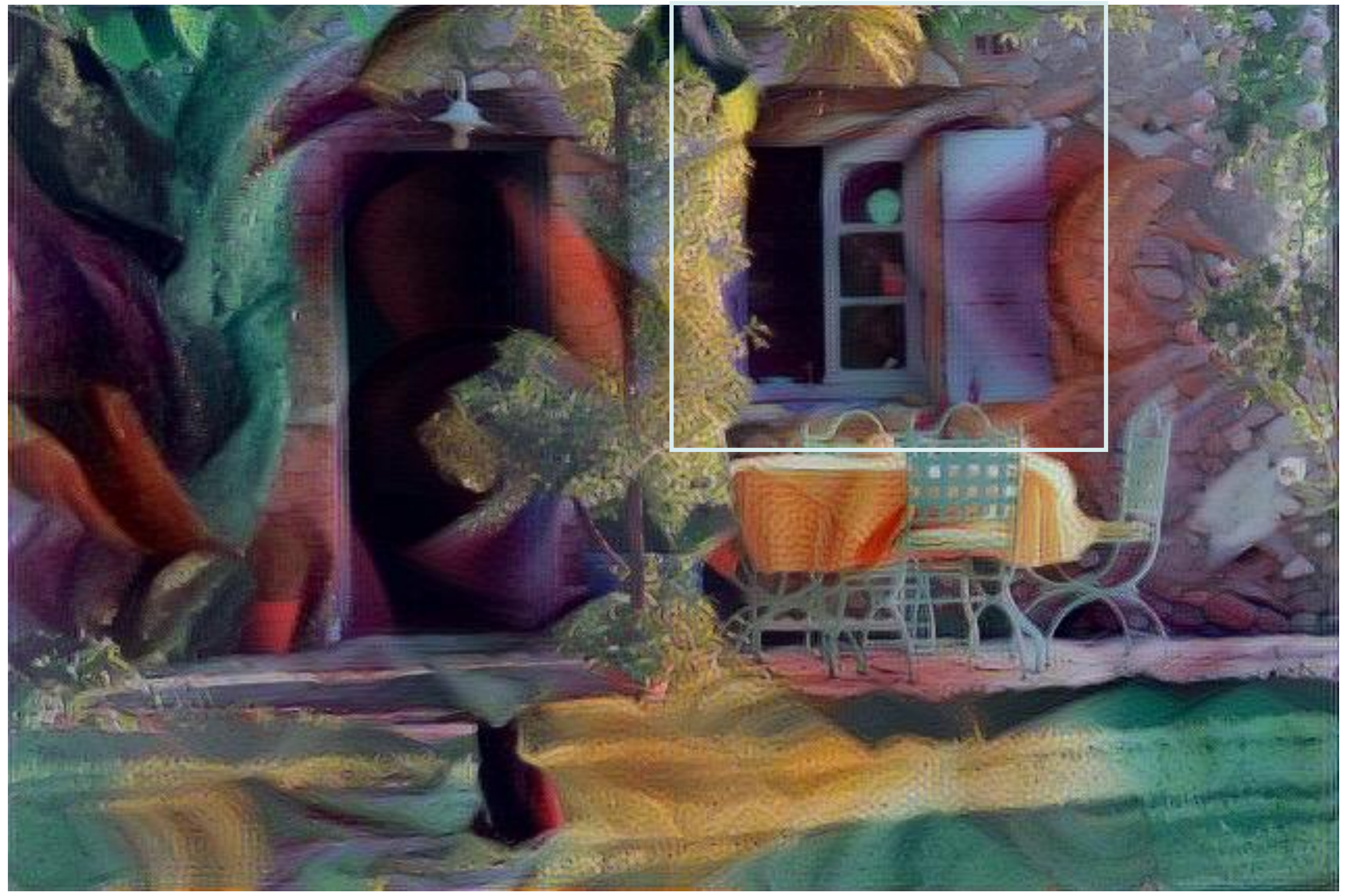
Different Reconstruction Layers

Conv2_2

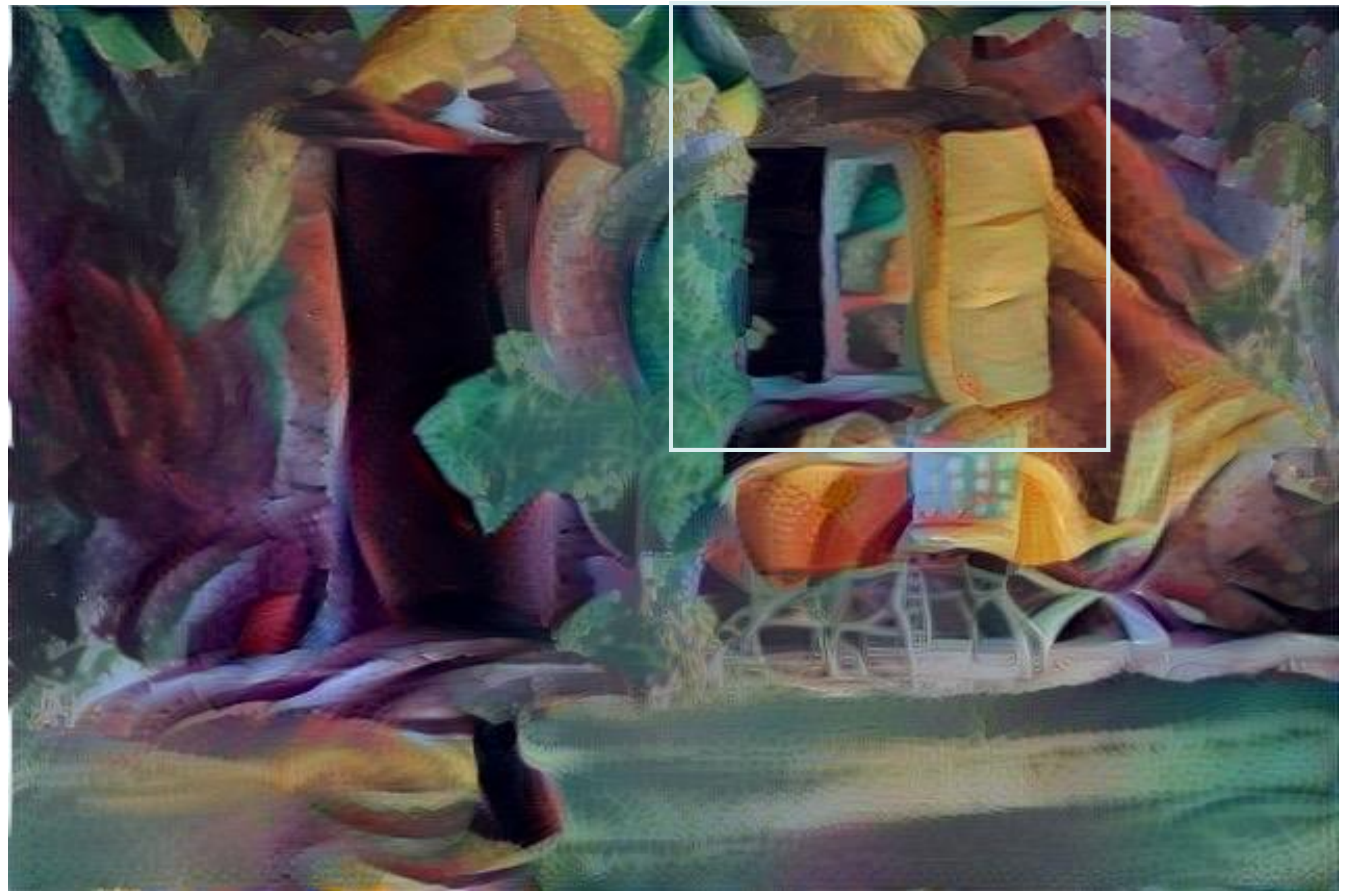
Conv4_2

Different Reconstruction Layers

Conv2_2



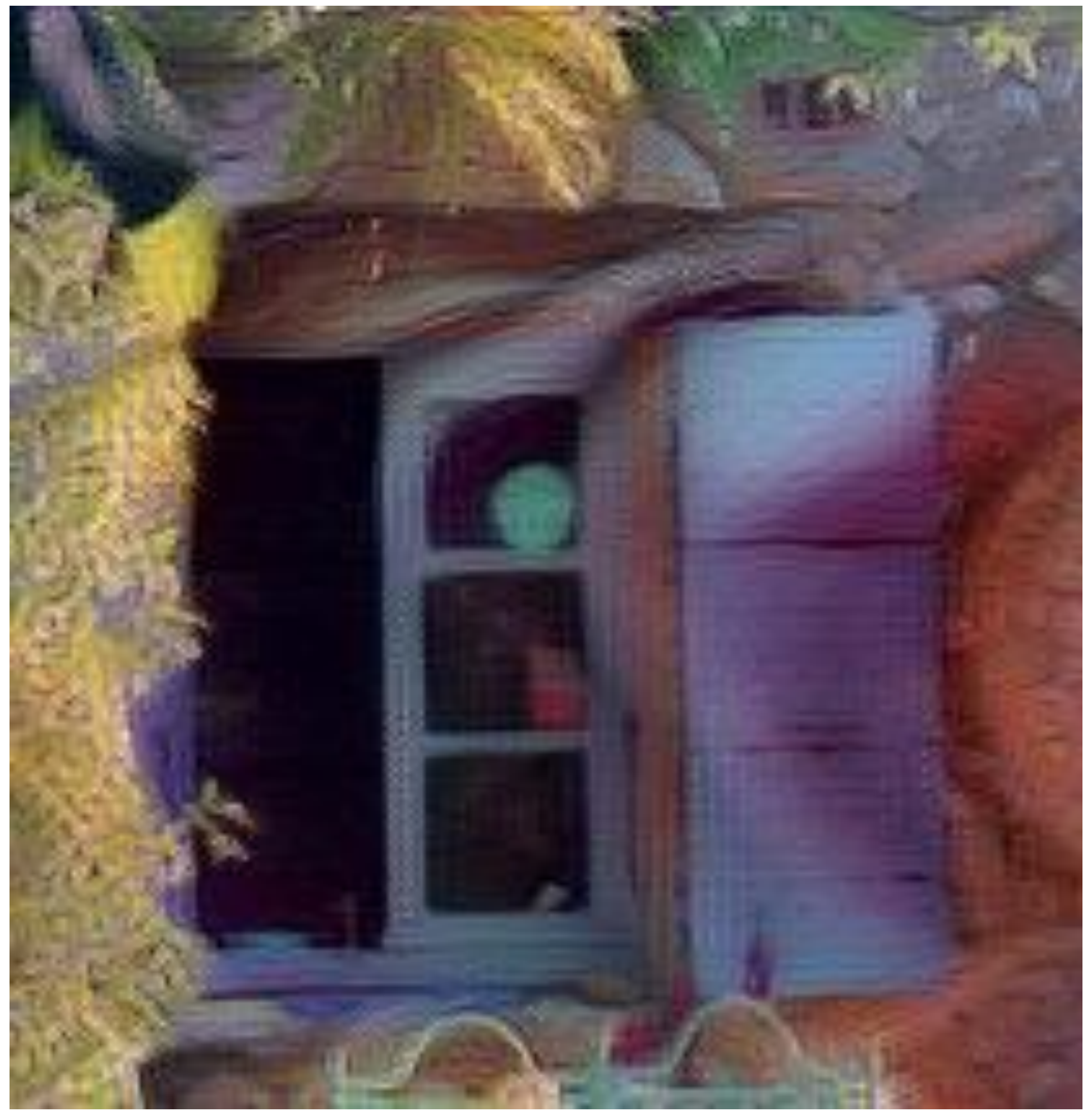
Conv4_2



Different Reconstruction Layers

Original

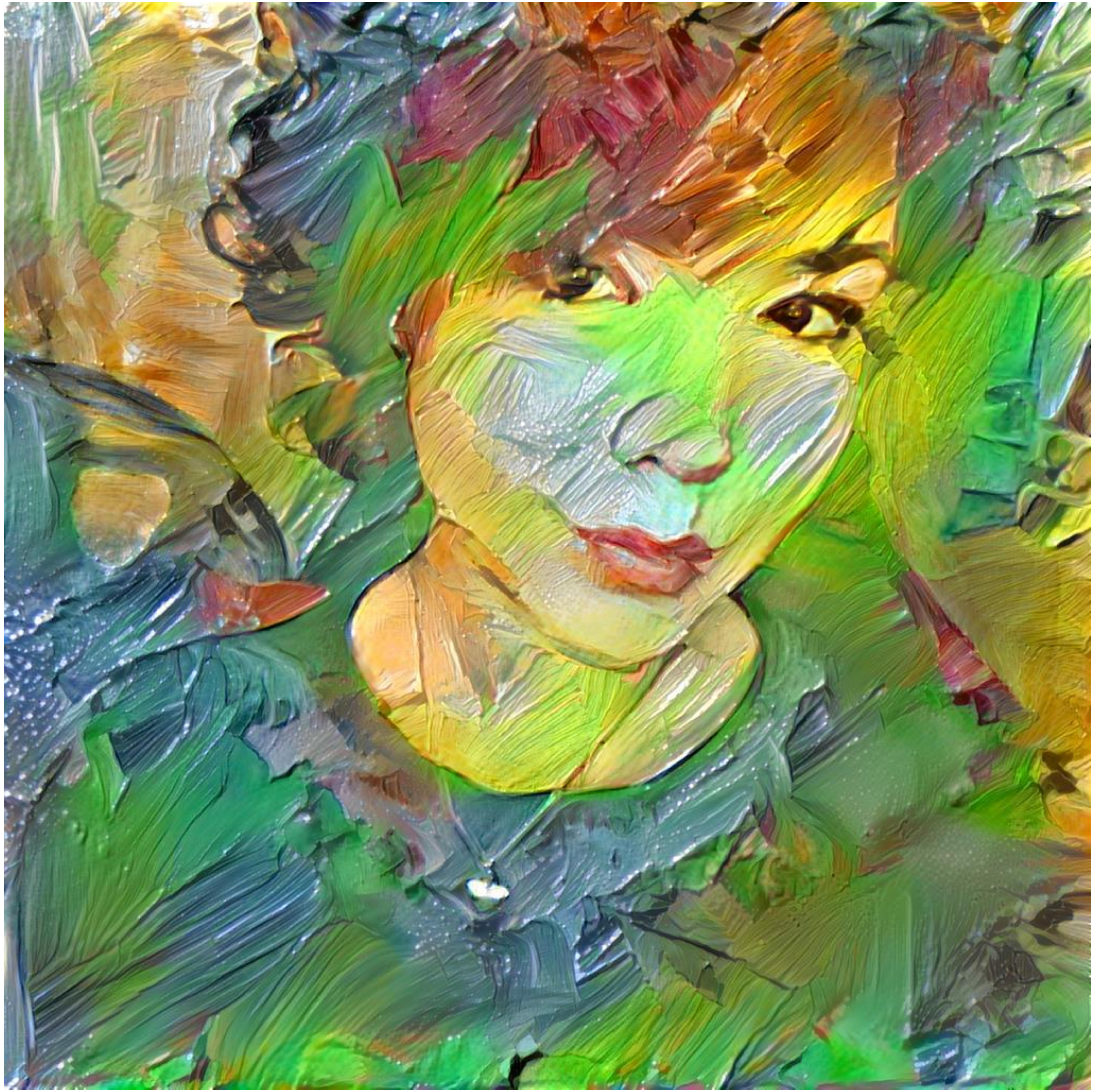
Conv2_2



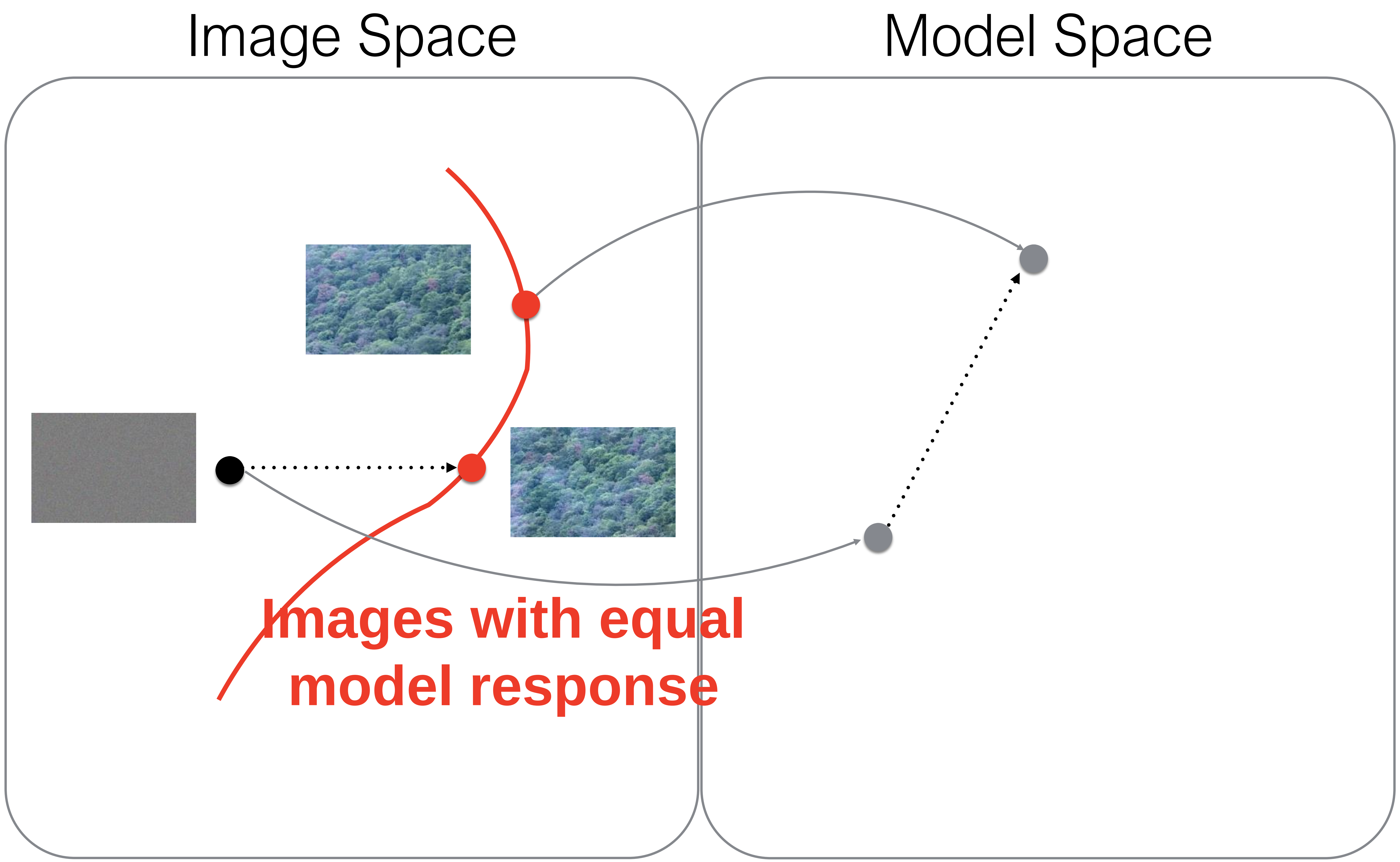
Conv4_2

General Style Transfer

General Style Transfer



Projecting onto the "Image Manifold"



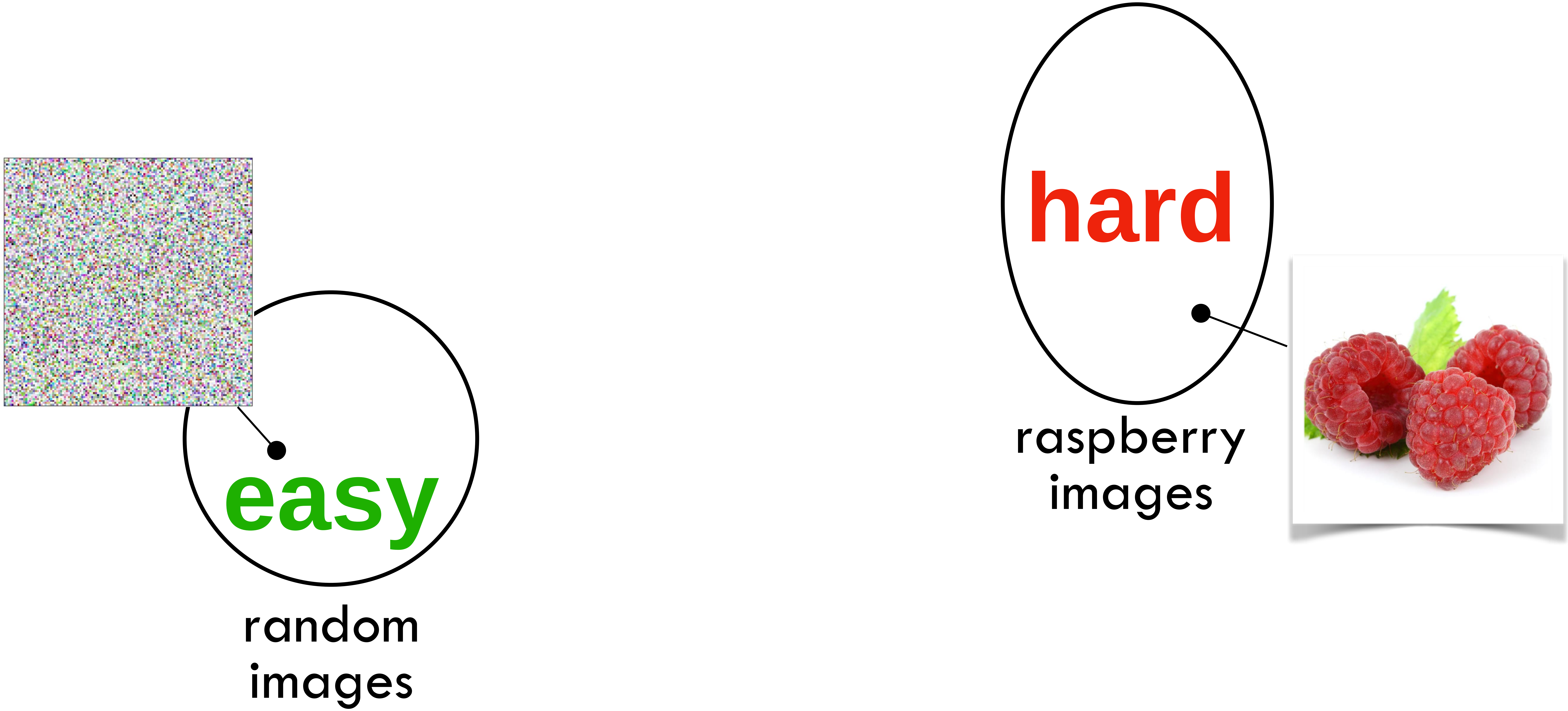
Portilla & Simoncelli (2000)

Imagen

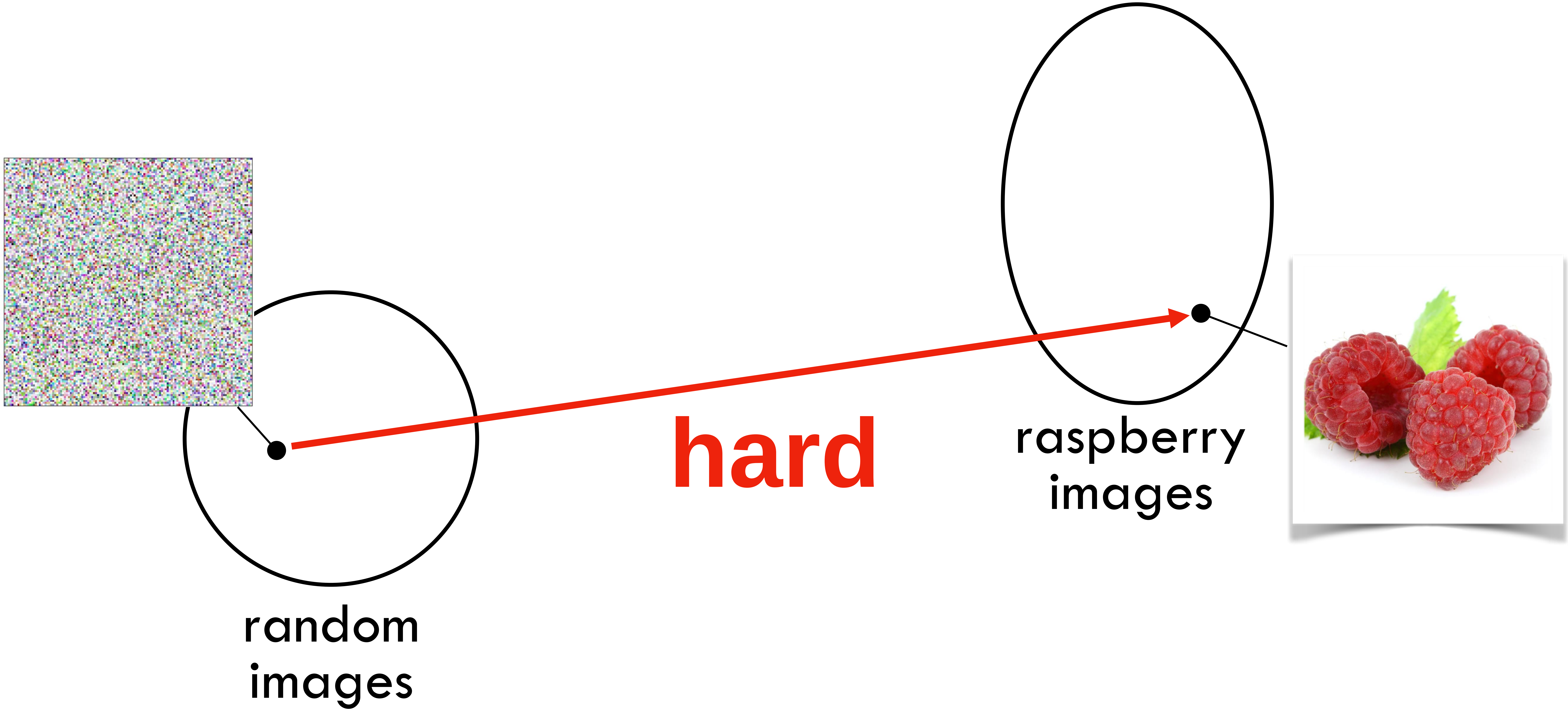
DALL-E 2

diffusion

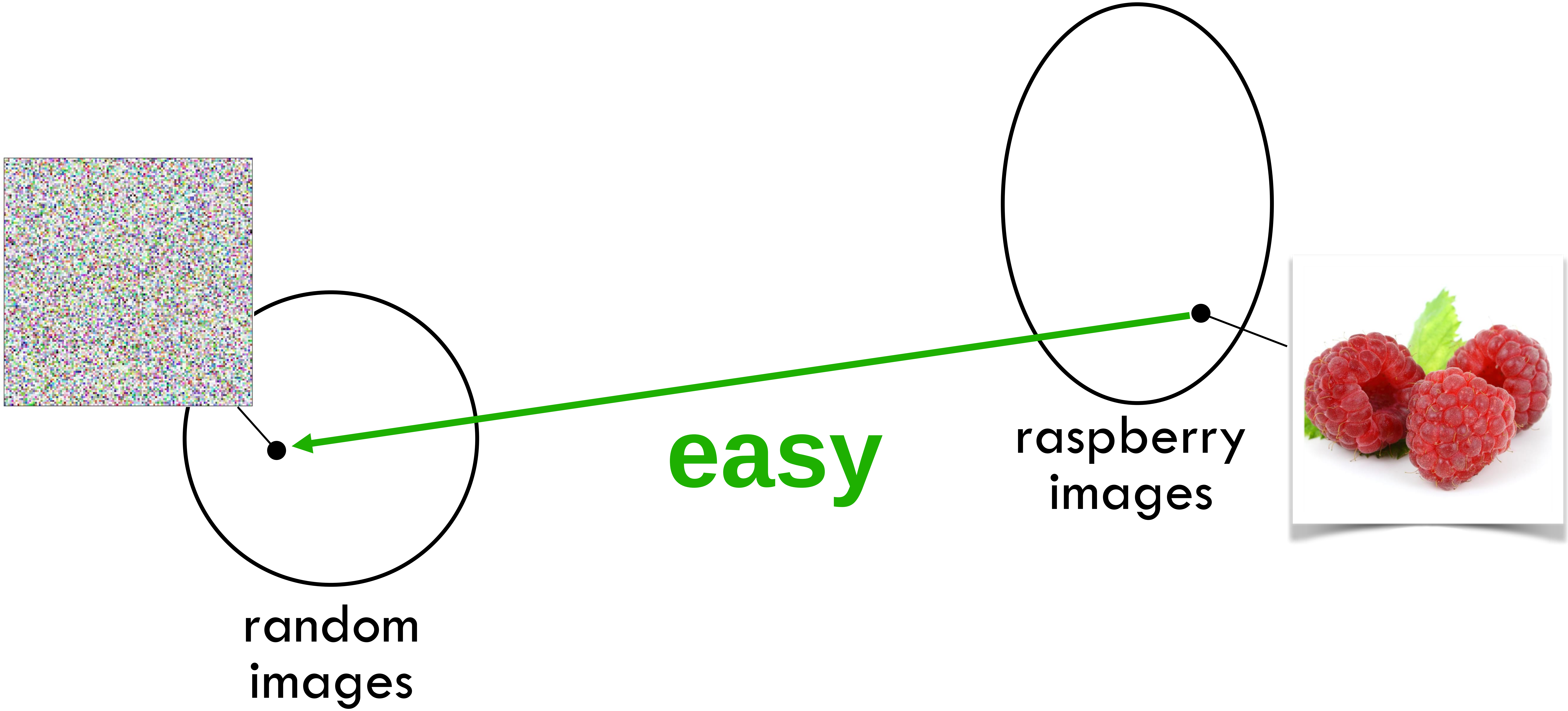
By Steve Seitz



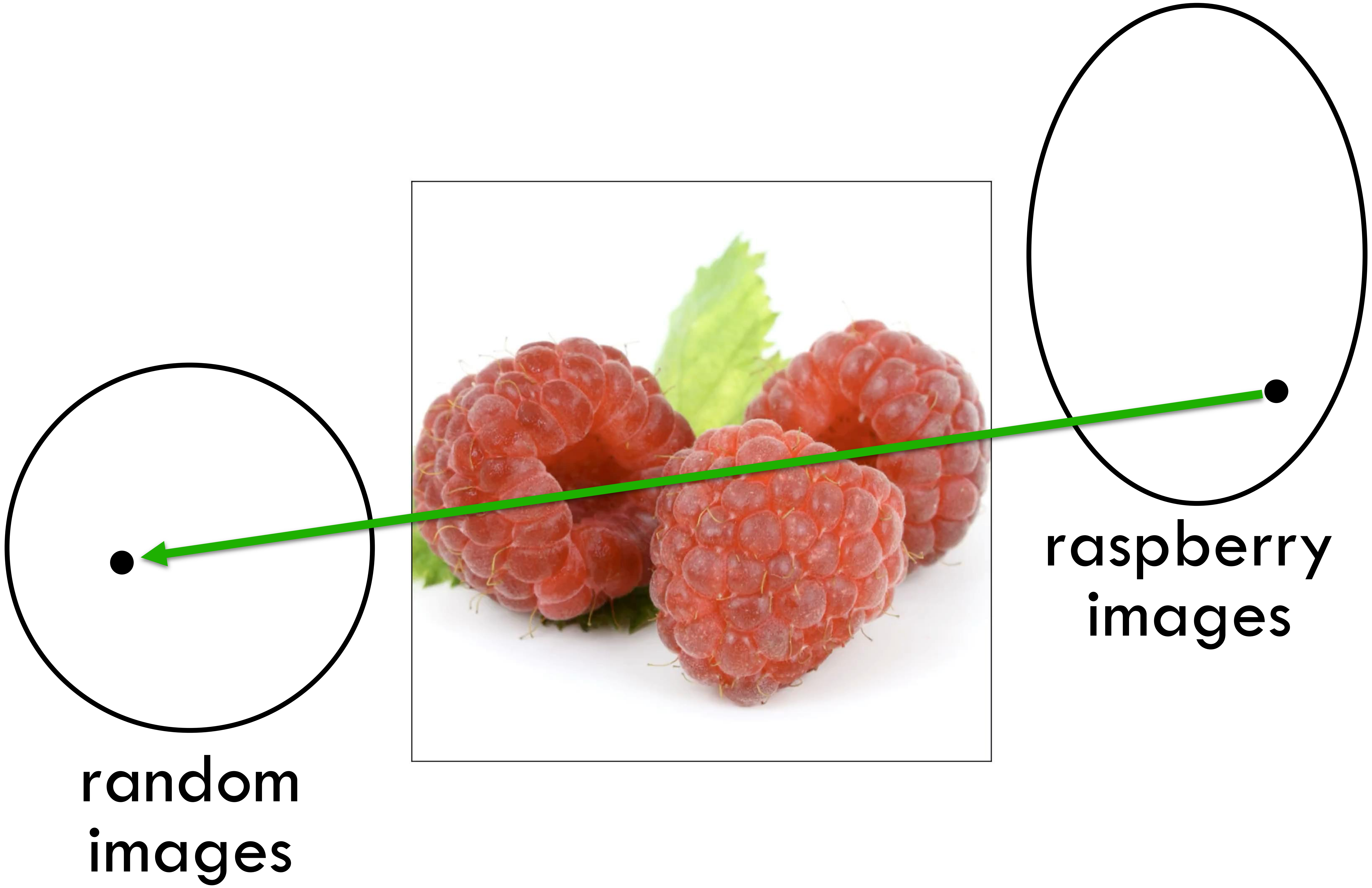
slide from Steve Seitz’s [video]



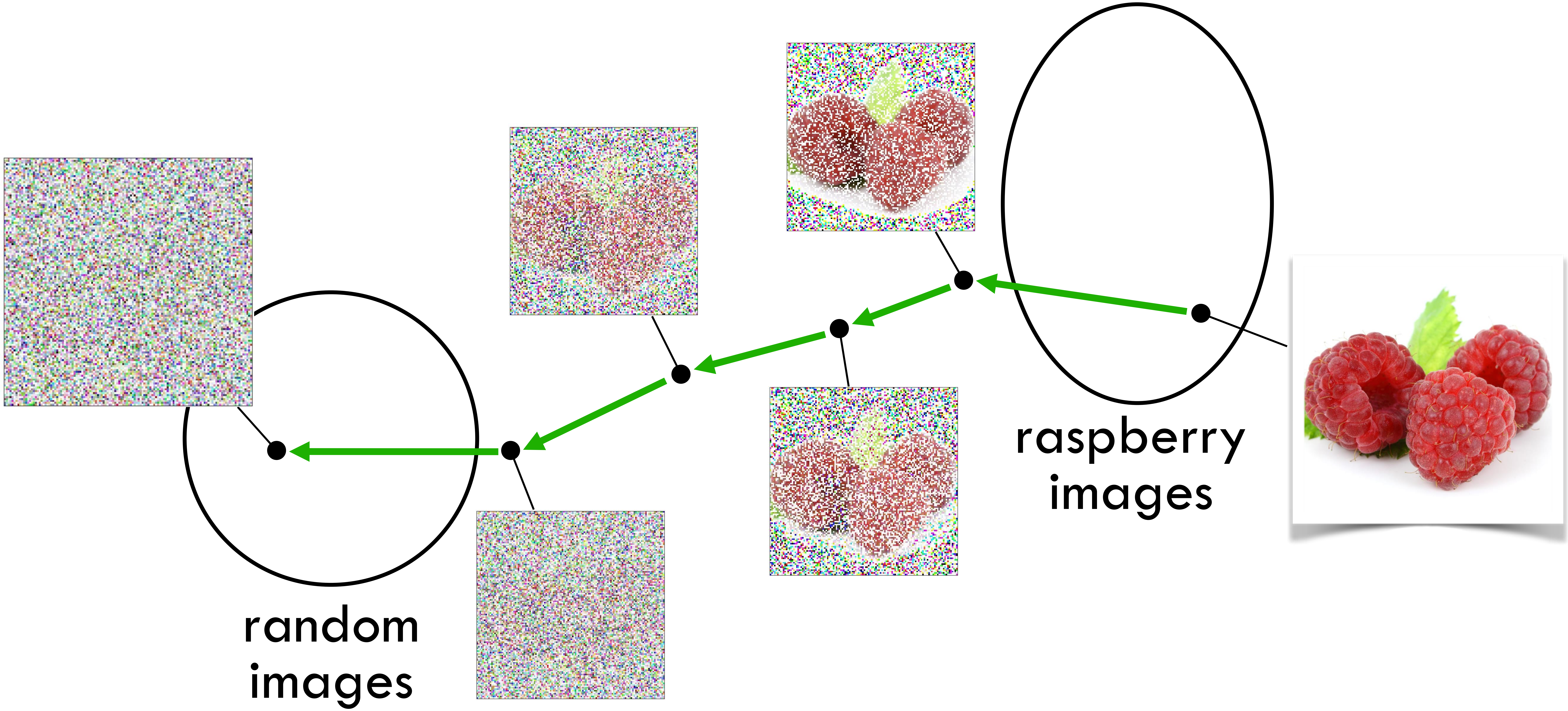
slide from Steve Seitz's video



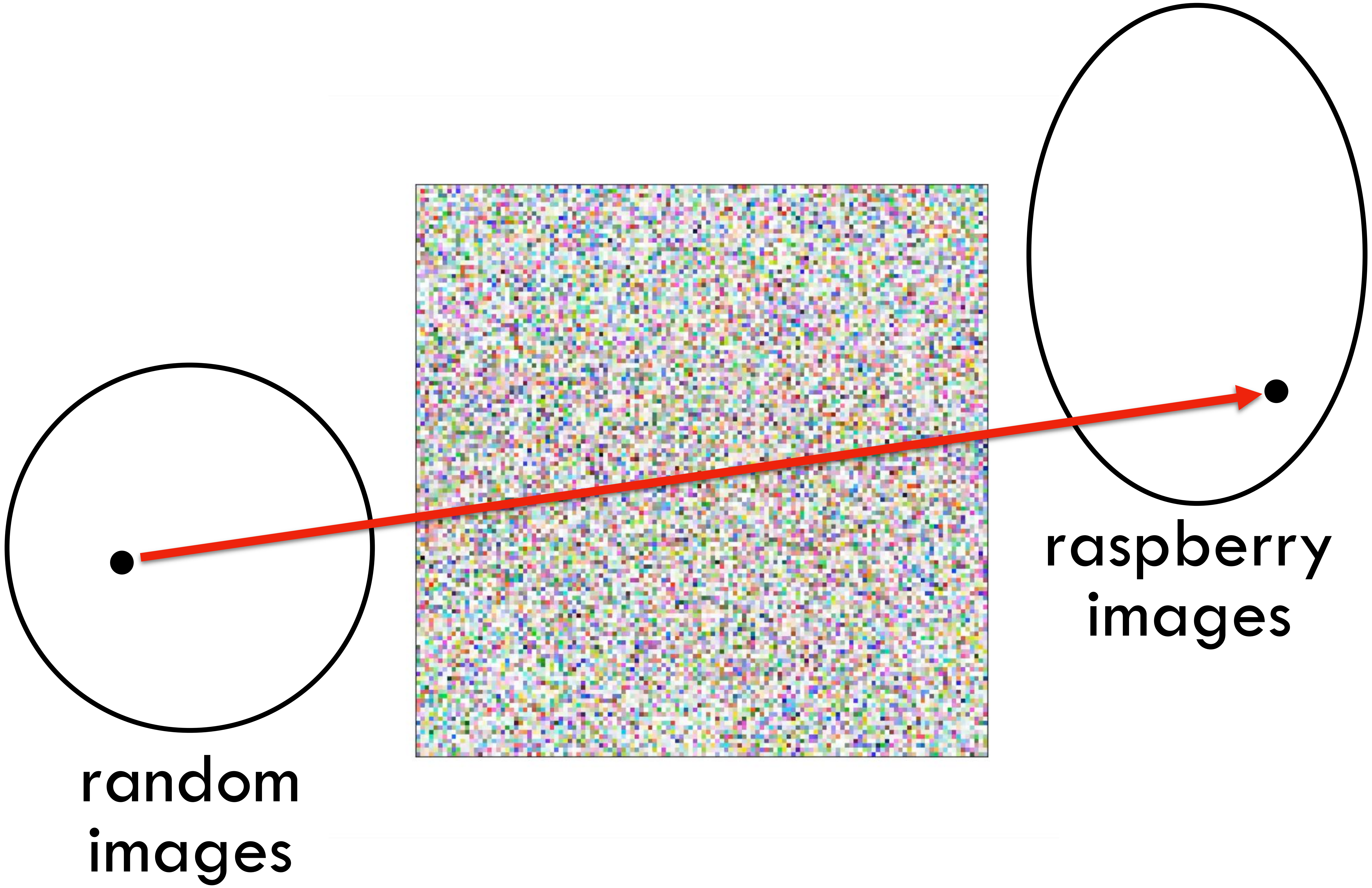
slide from Steve Seitz’s [video]



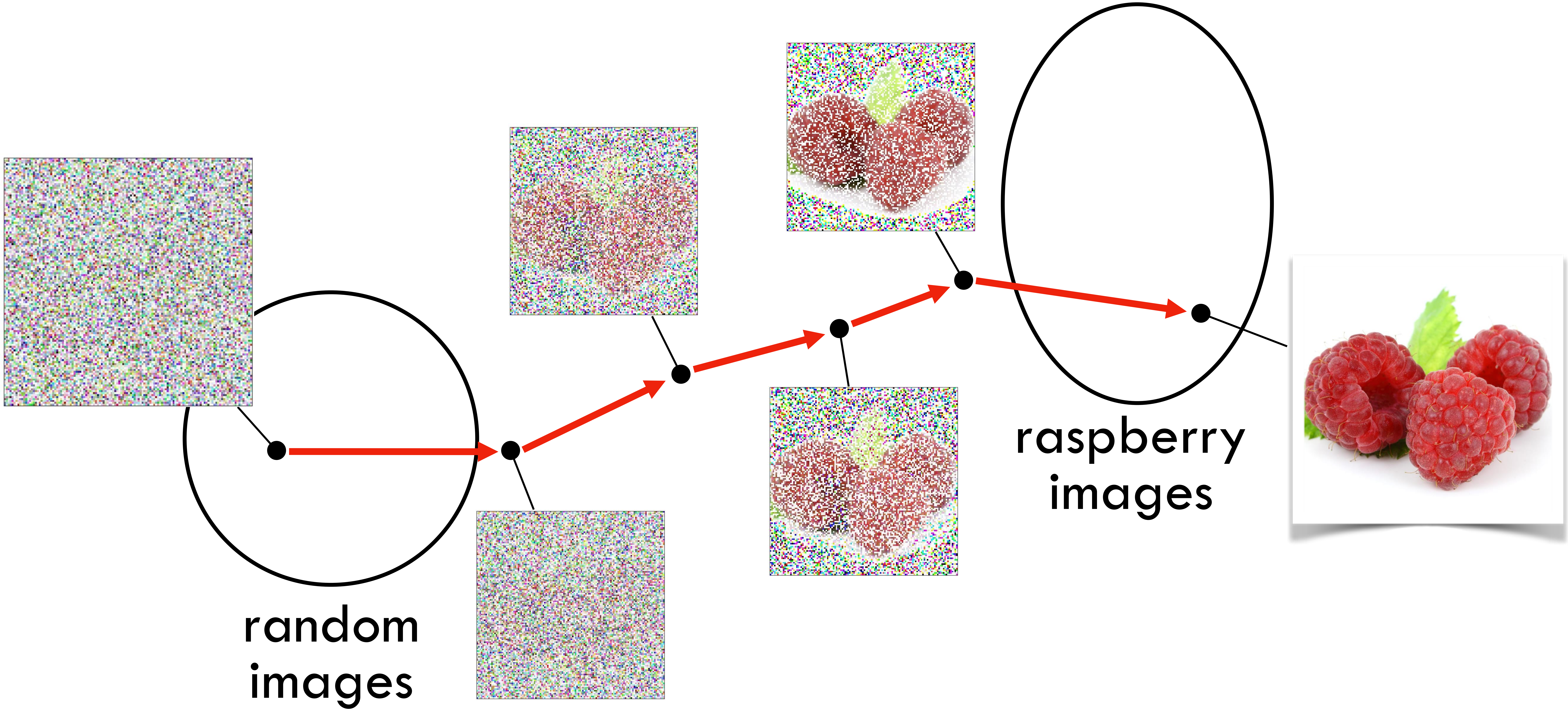
slide from Steve Seitz's [video]



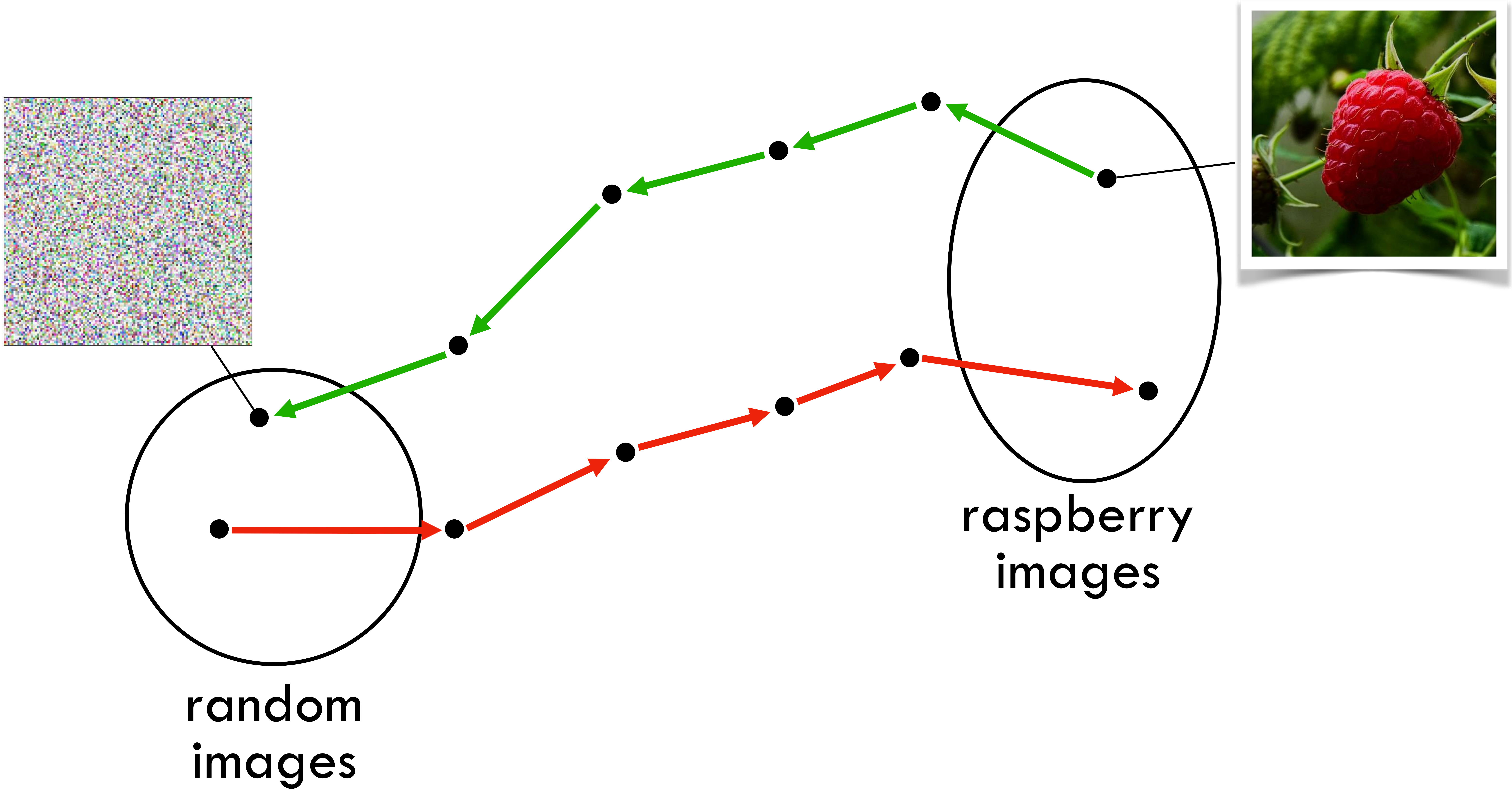
slide from Steve Seitz's [video]



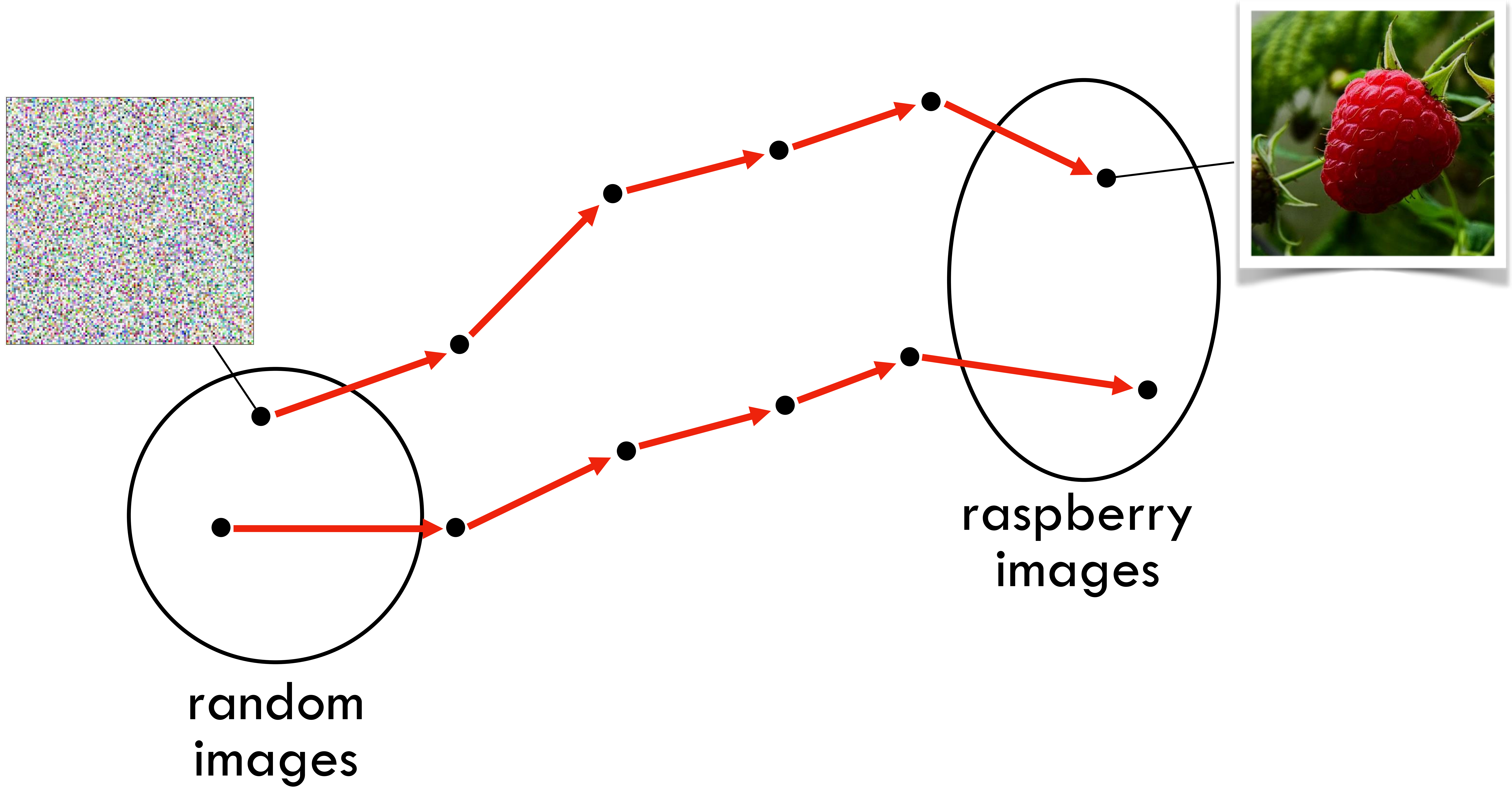
slide from Steve Seitz’s [video]



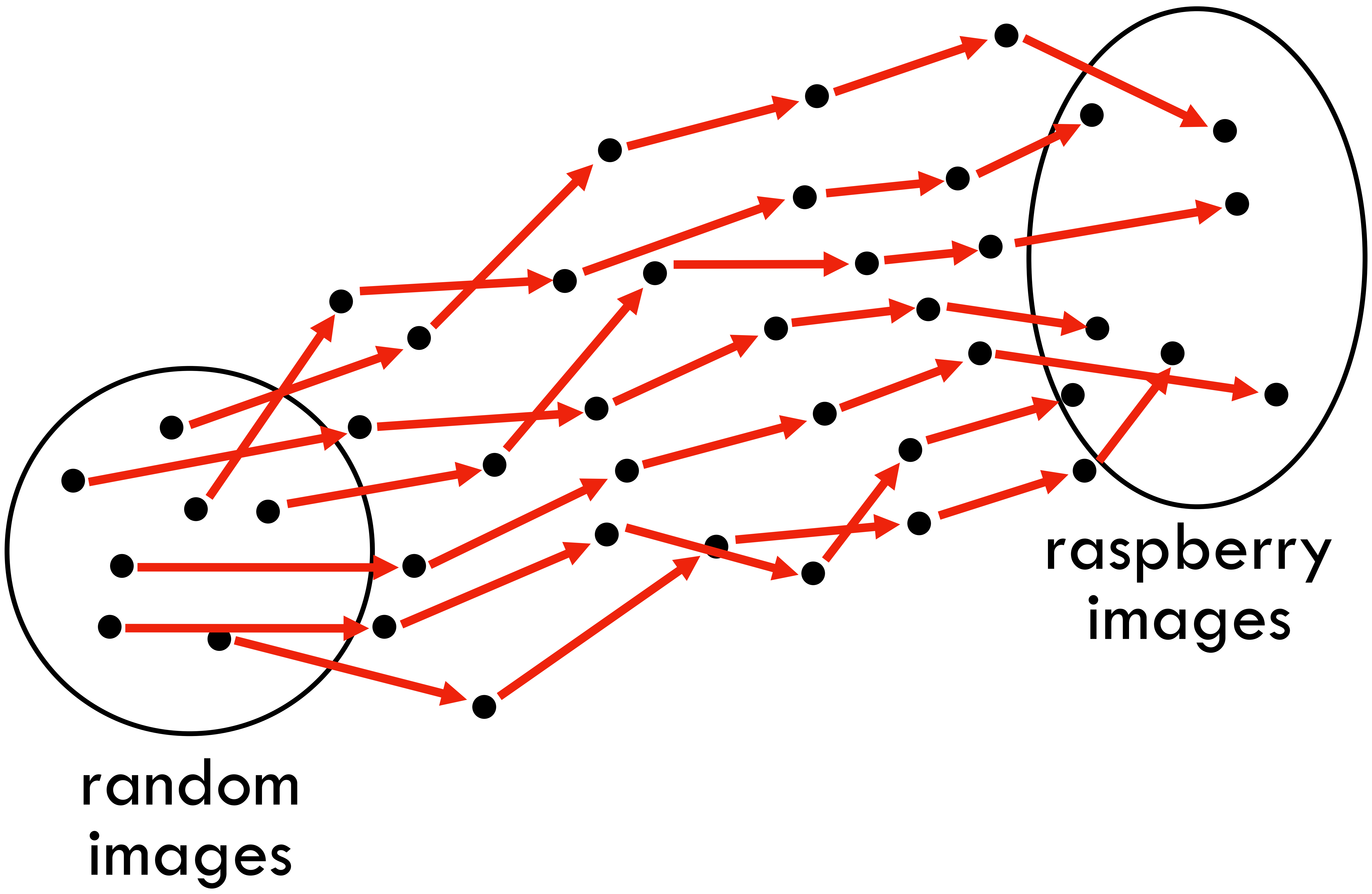
slide from Steve Seitz's [video]



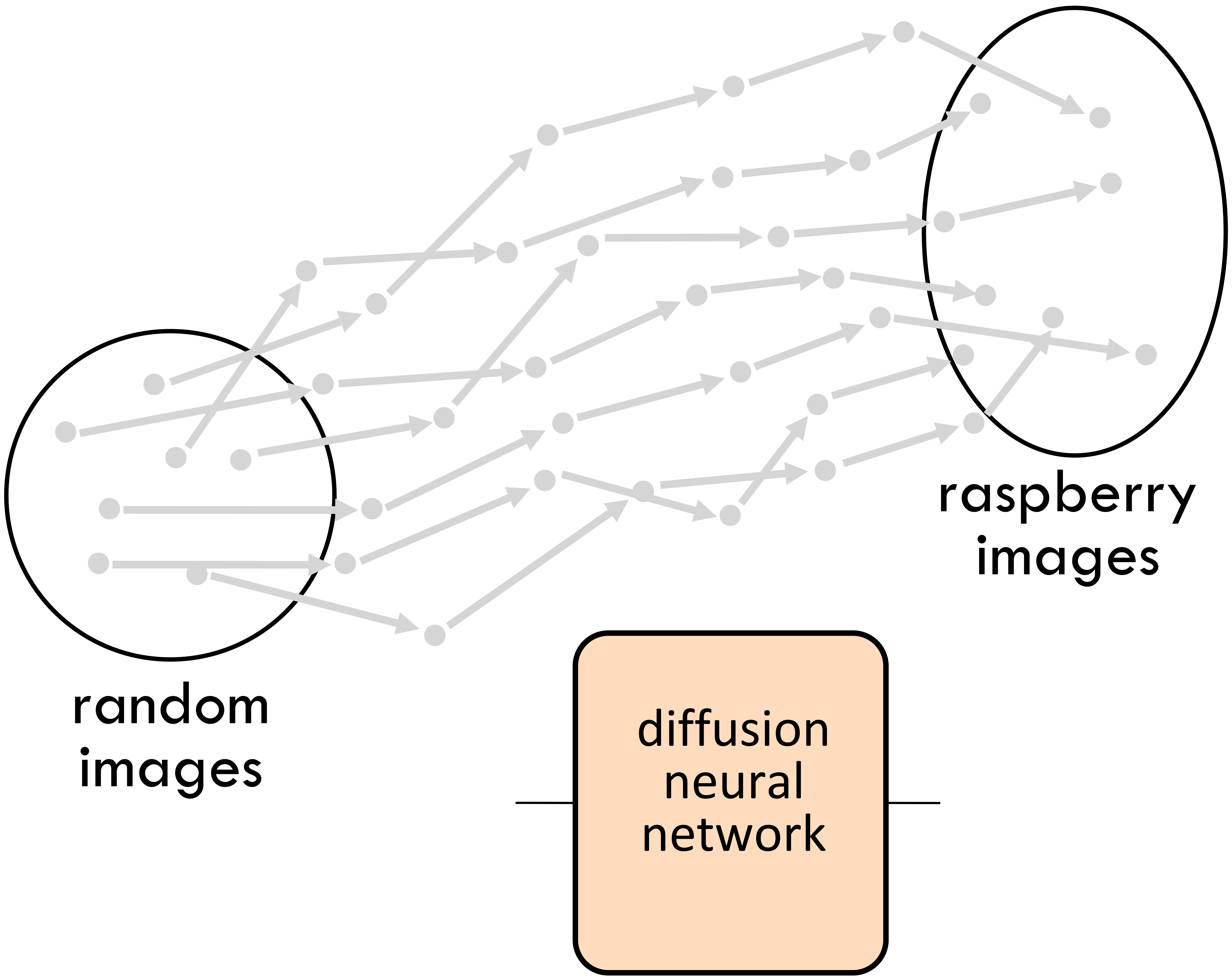
slide from Steve Seitz's video



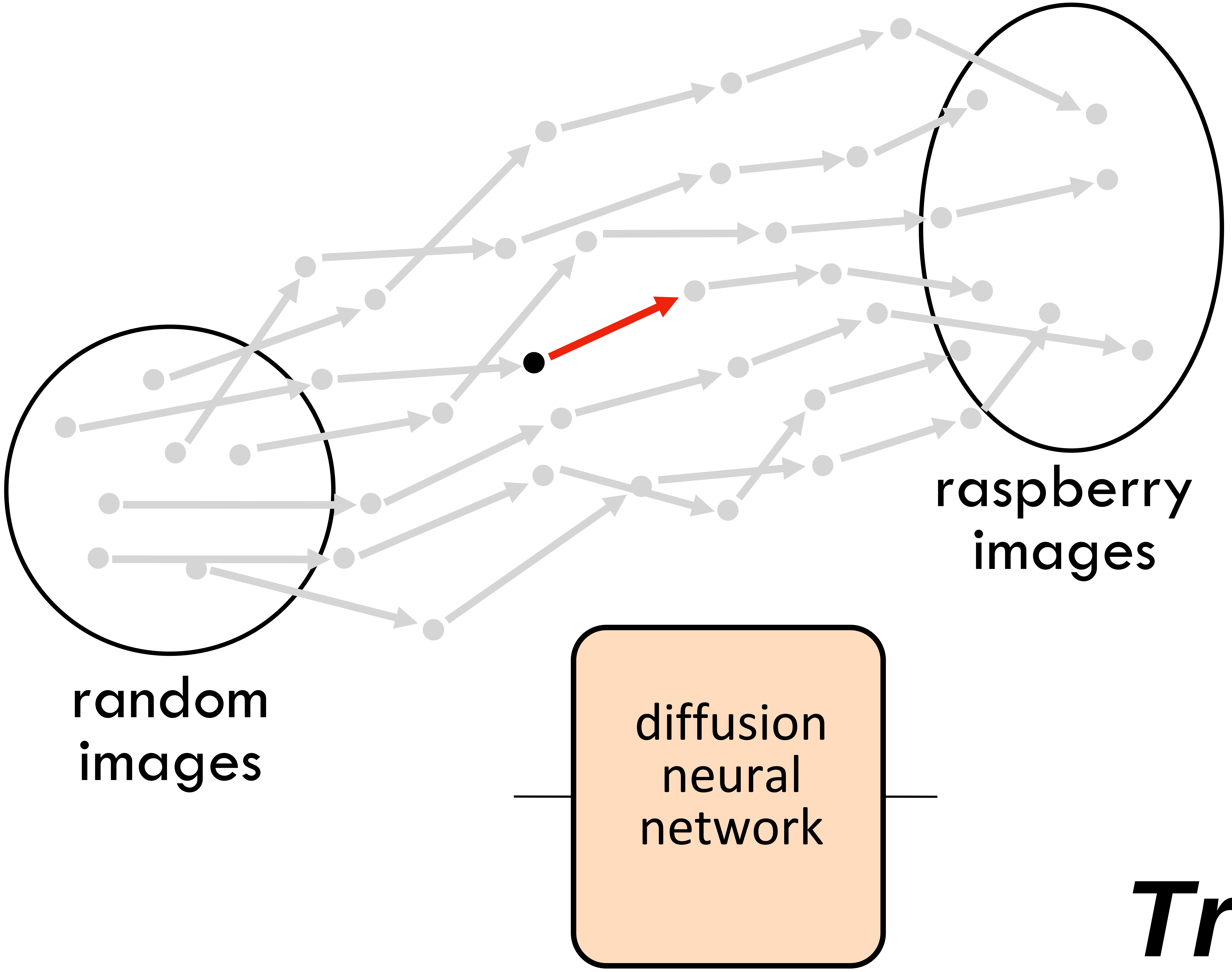
slide from Steve Seitz's [video]



slide from Steve Seitz's video

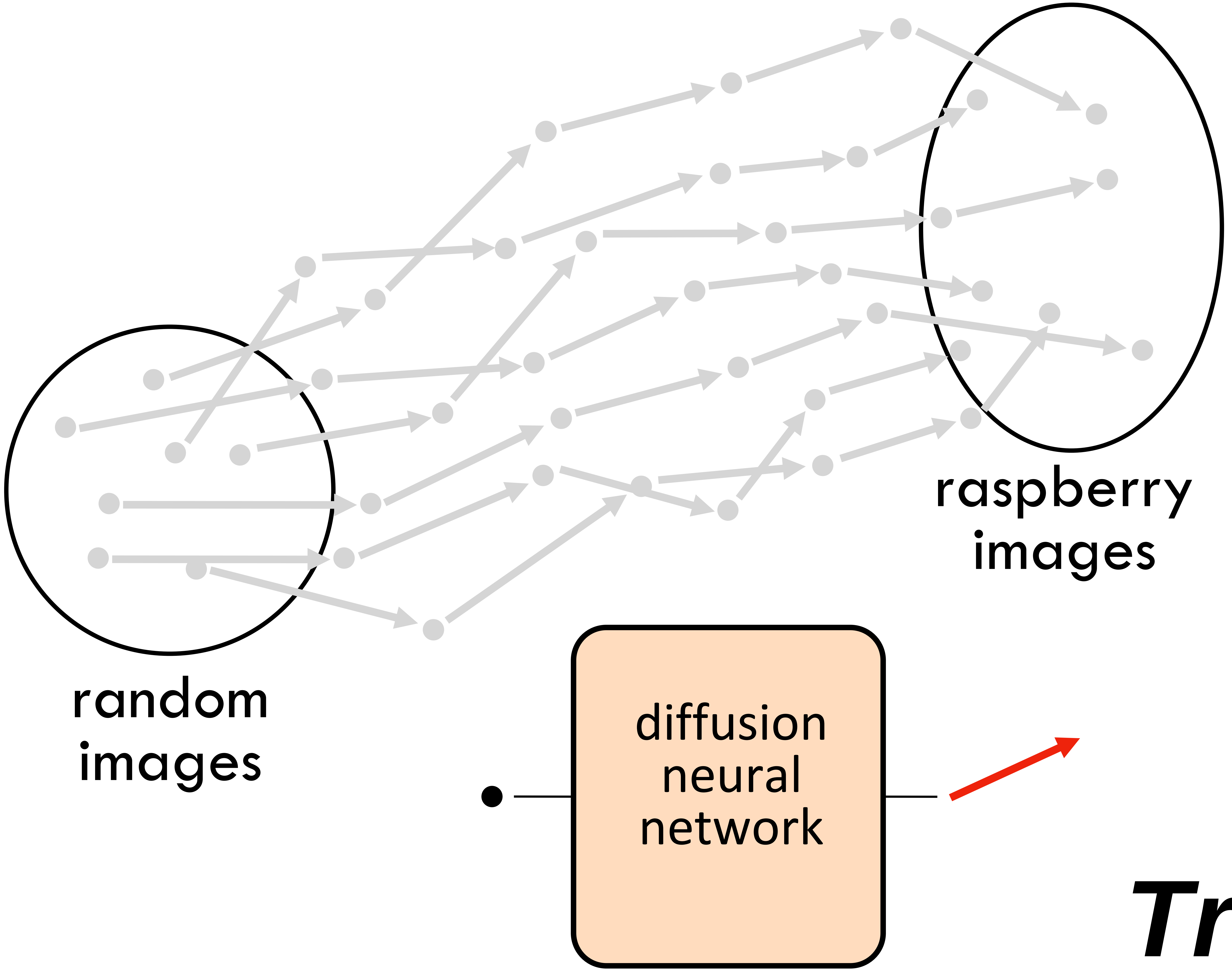


slide from Steve Seitz's video



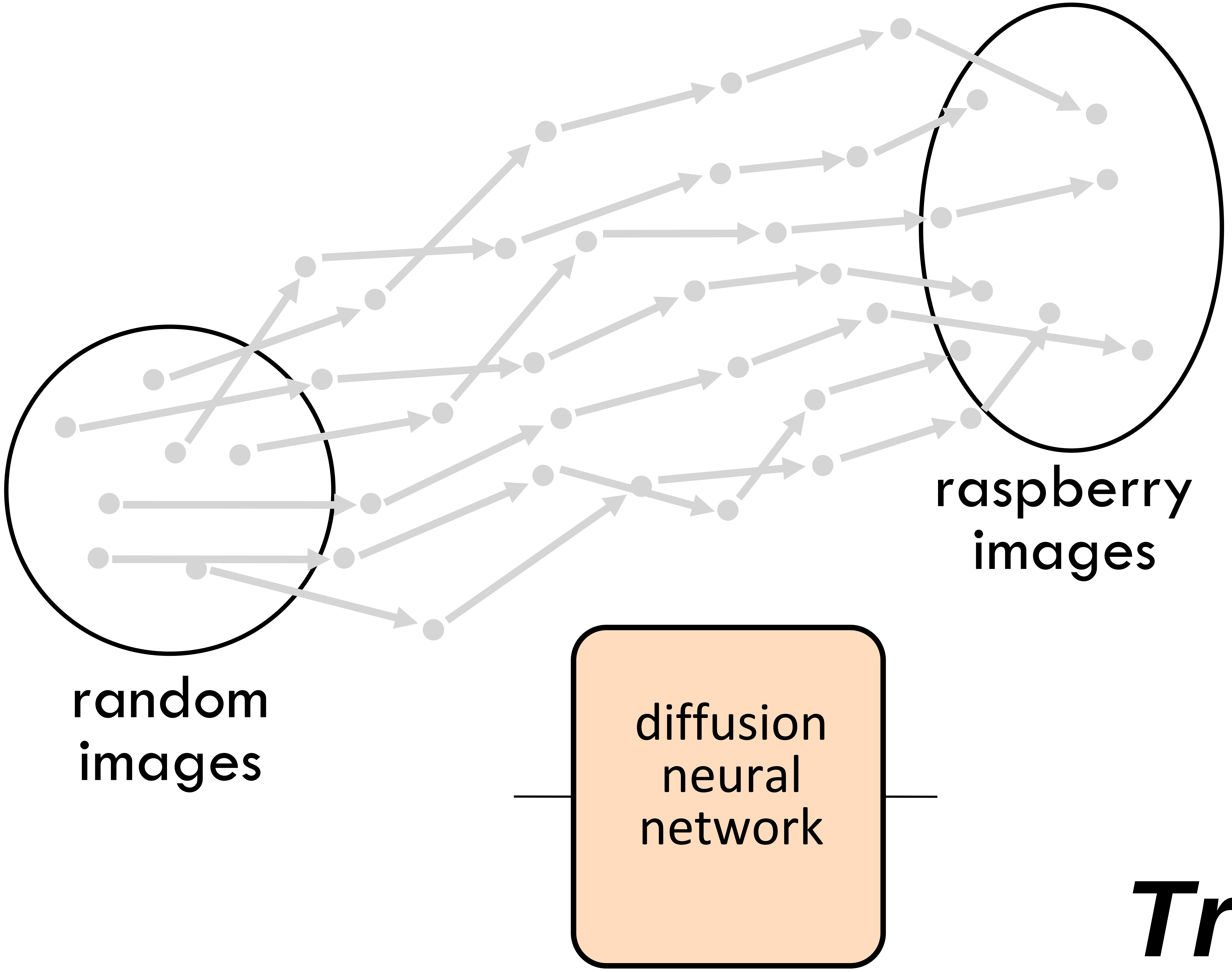
Training

slide from Steve Seitz's video



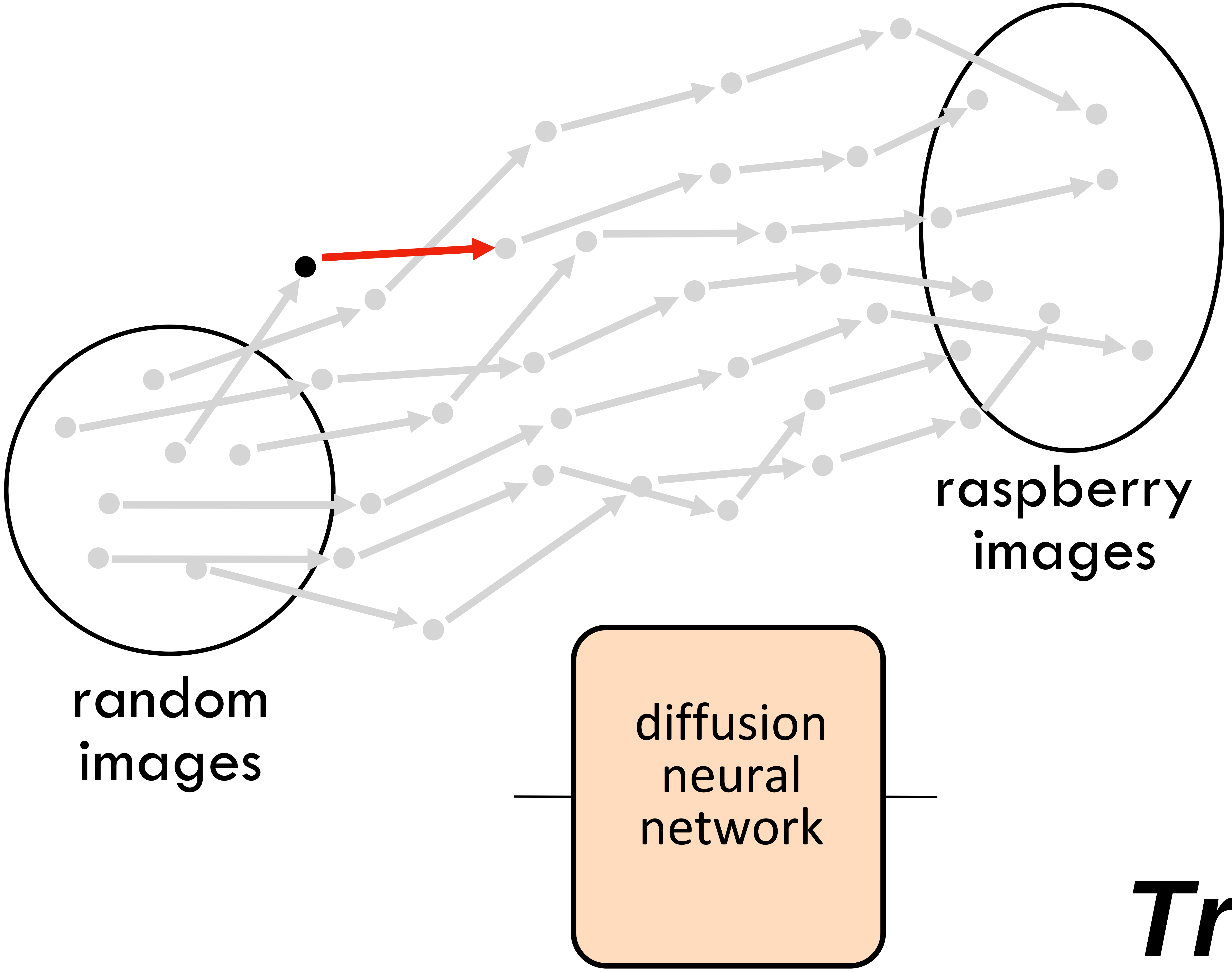
Training

slide from Steve Seitz's video



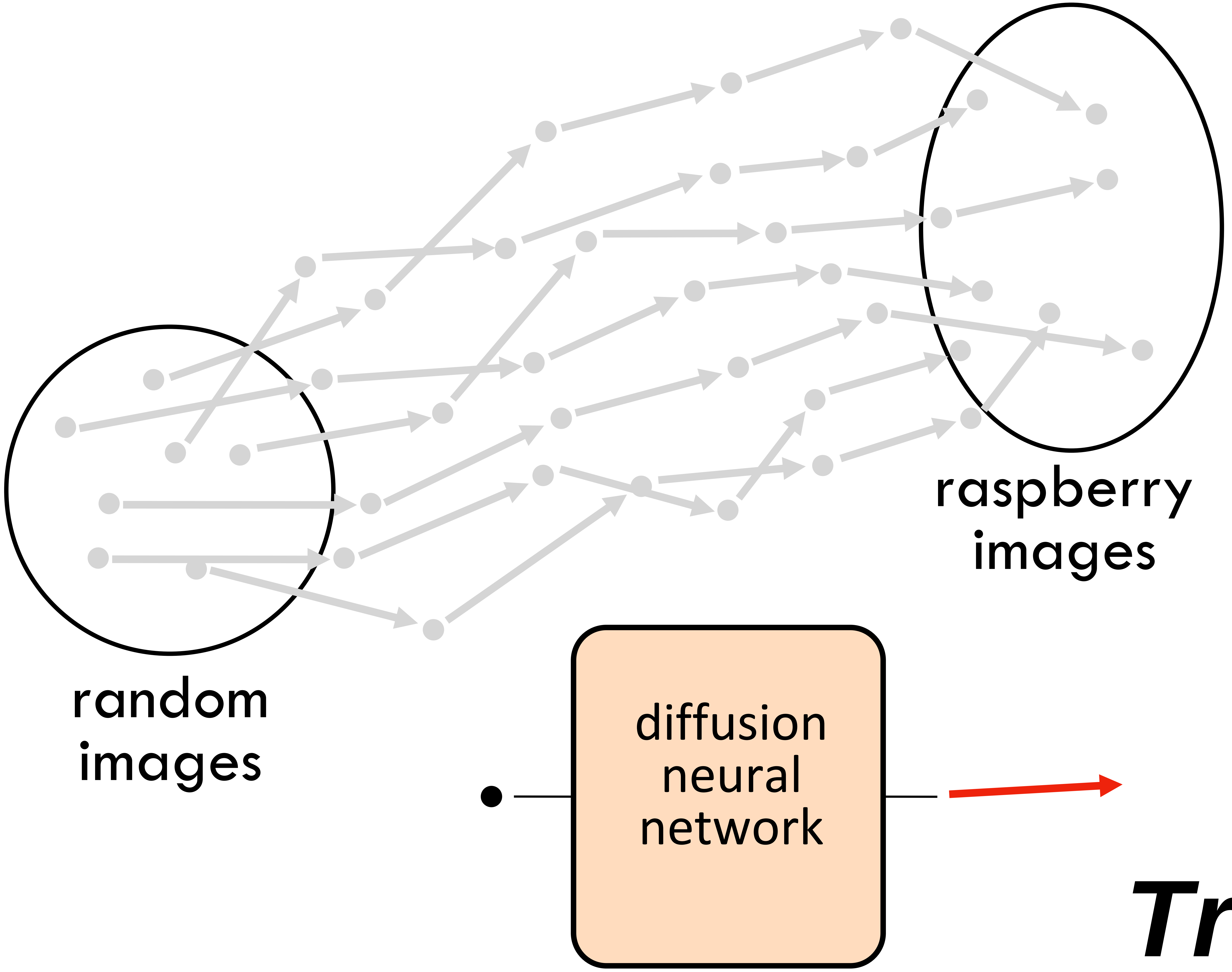
Training

slide from Steve Seitz's [video]



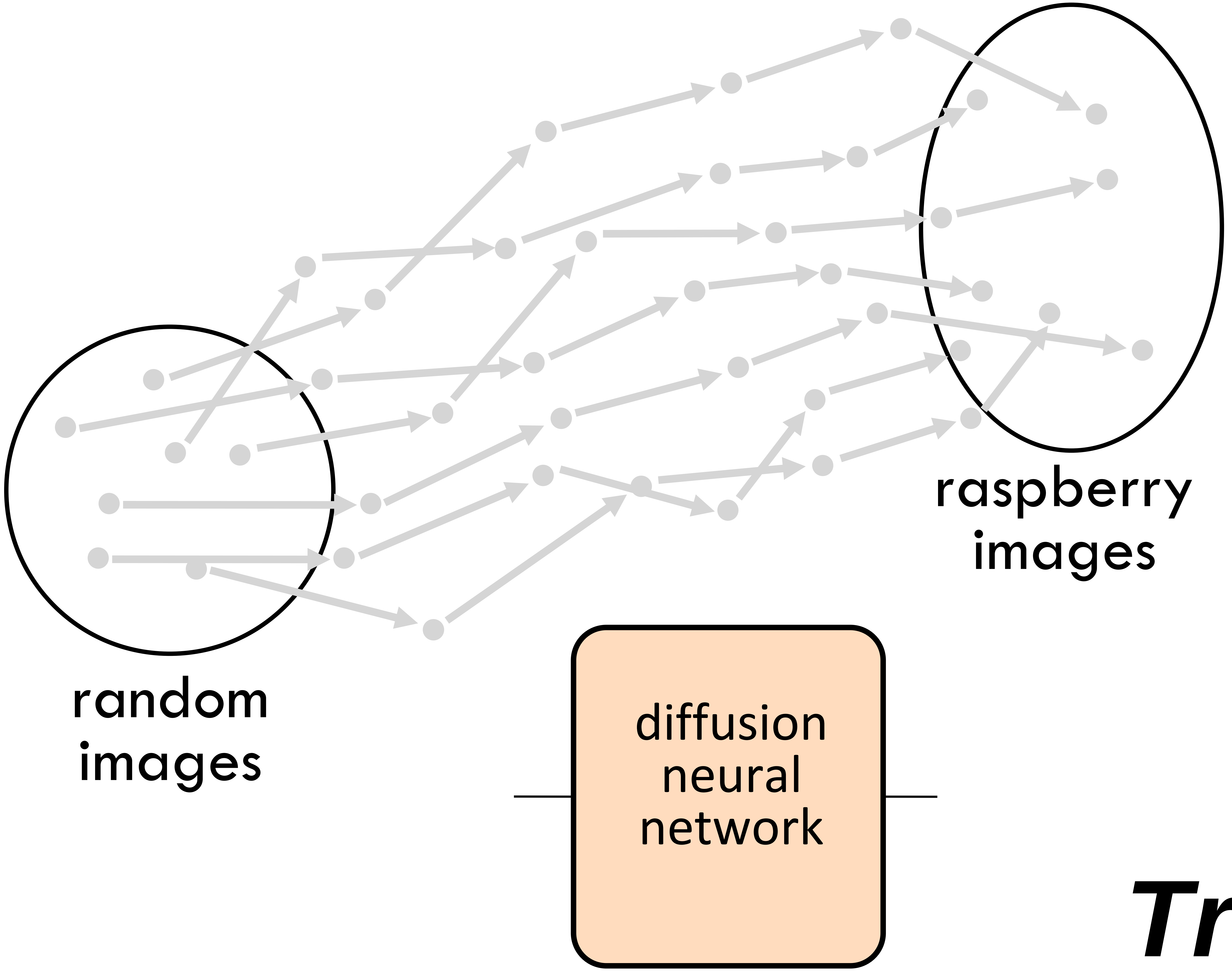
Training

slide from Steve Seitz's [video]



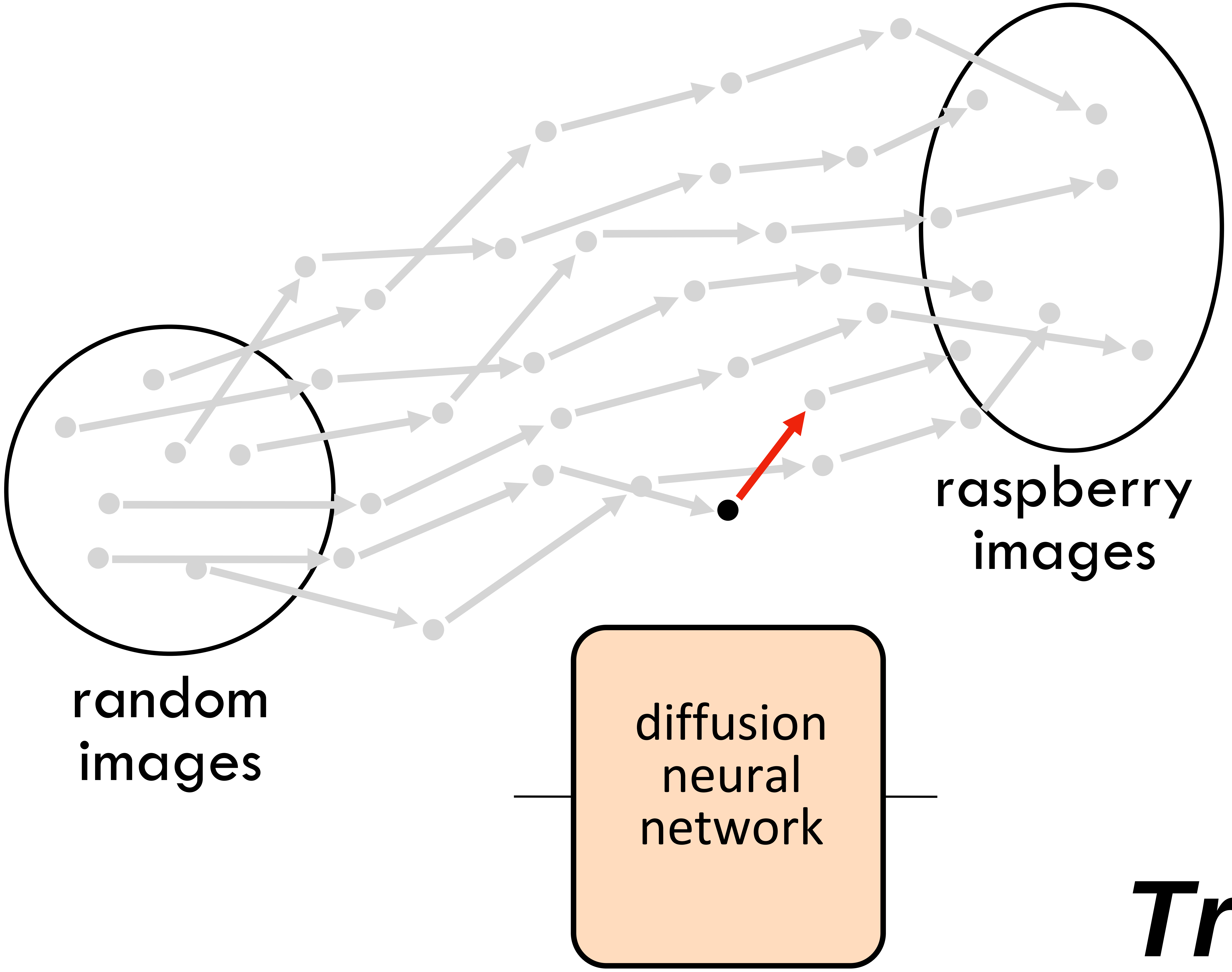
Training

slide from Steve Seitz's video



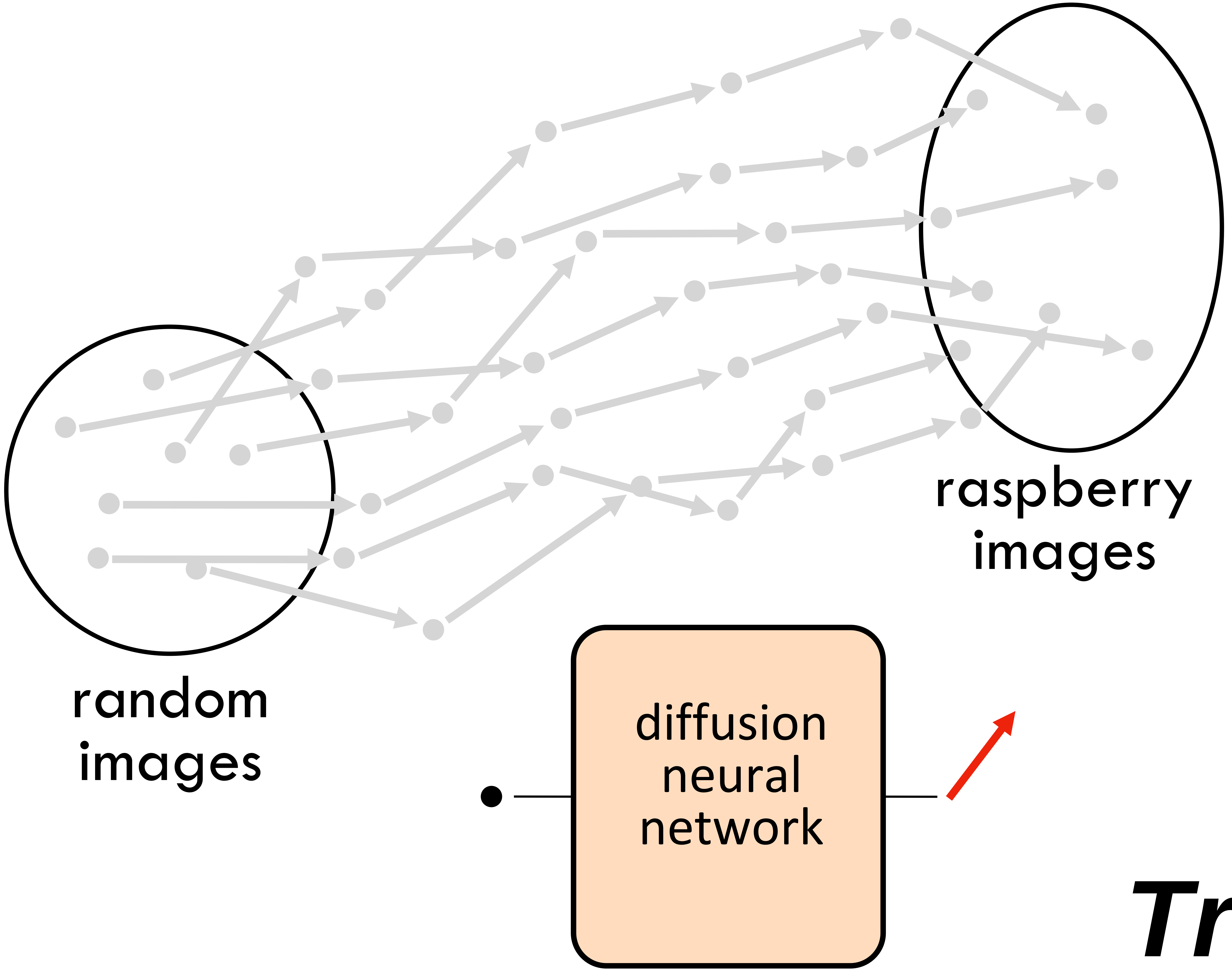
Training

slide from Steve Seitz's [video]



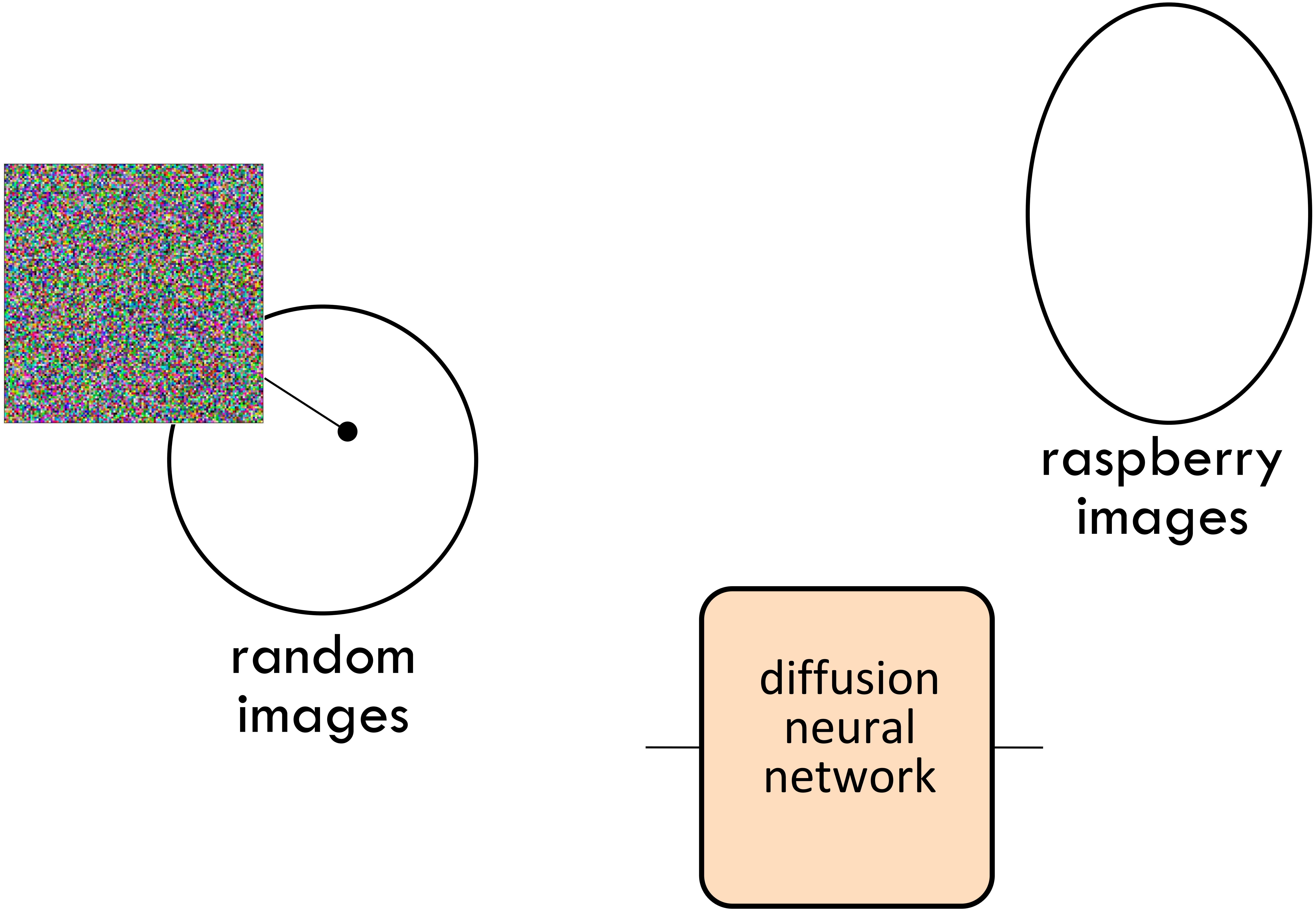
Training

slide from Steve Seitz's video

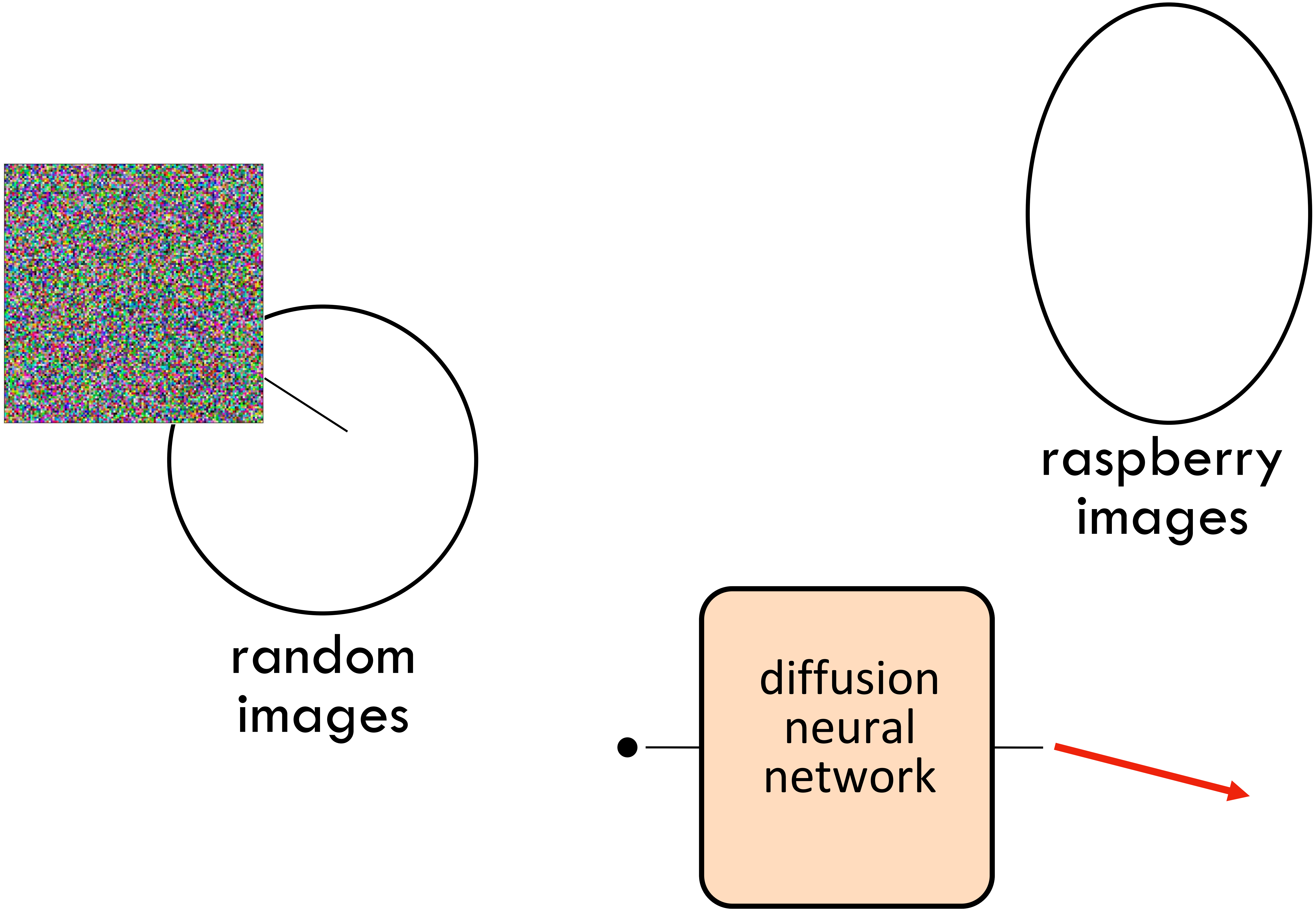


Training

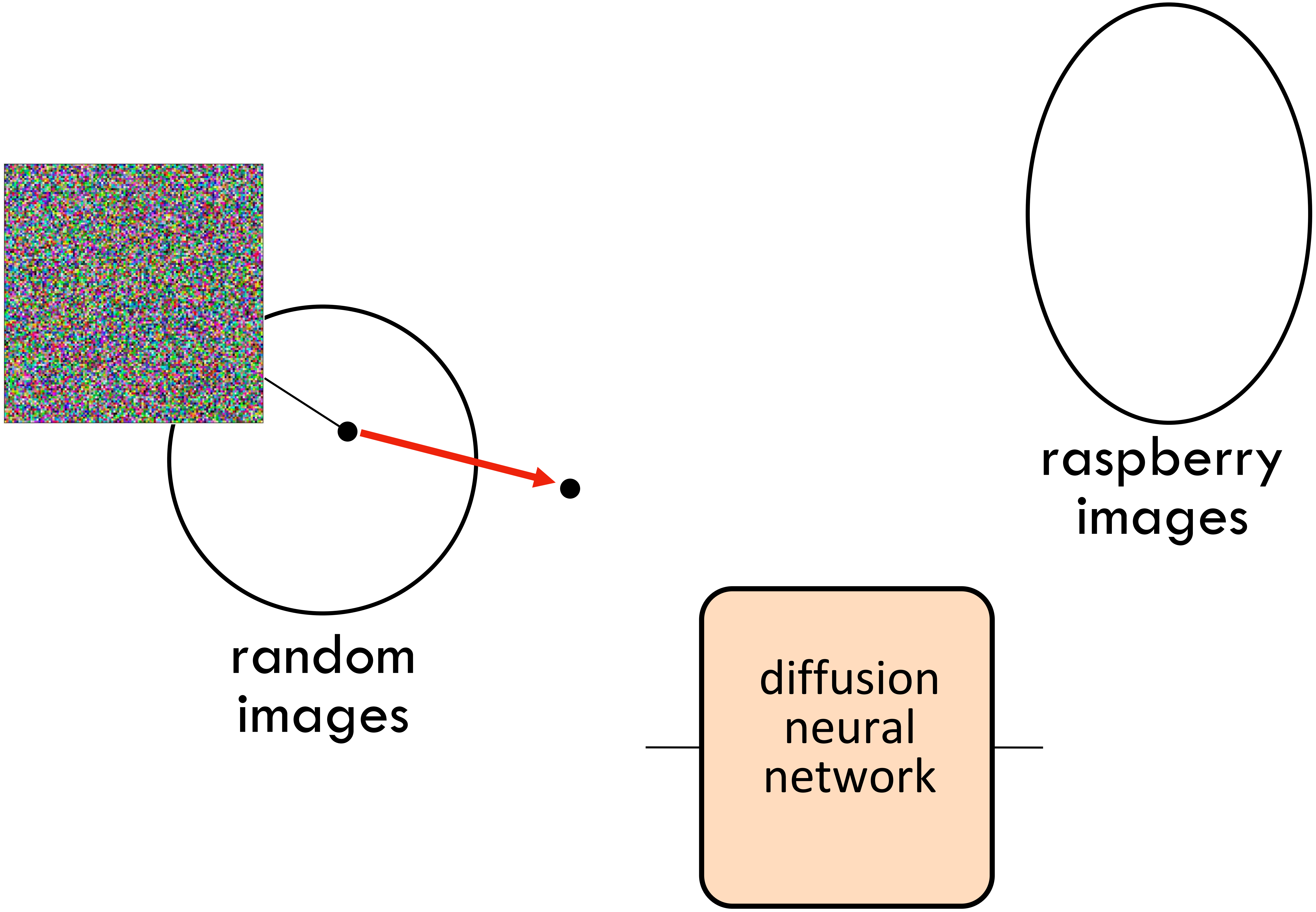
slide from Steve Seitz's video



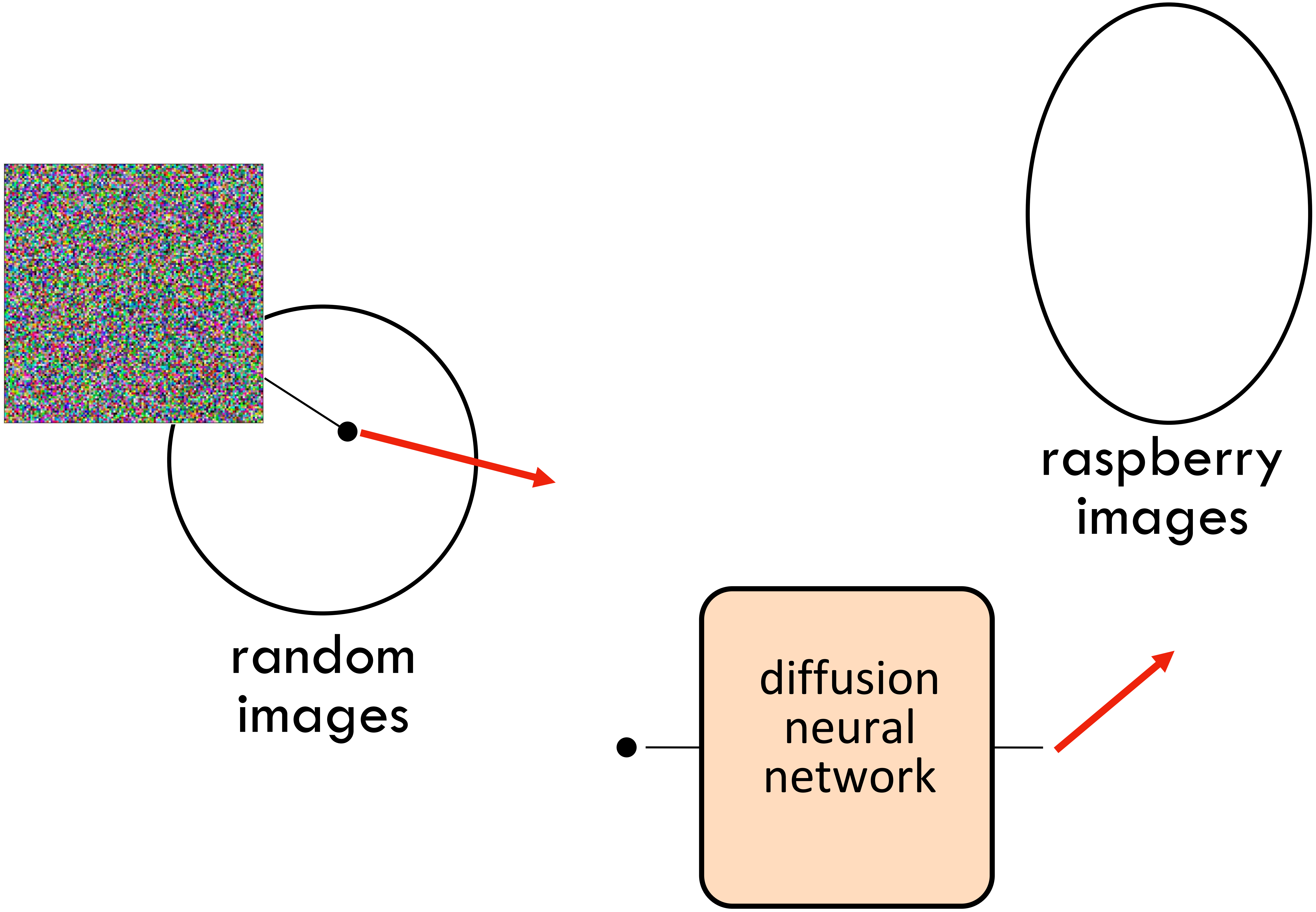
slide from Steve Seitz's [video]



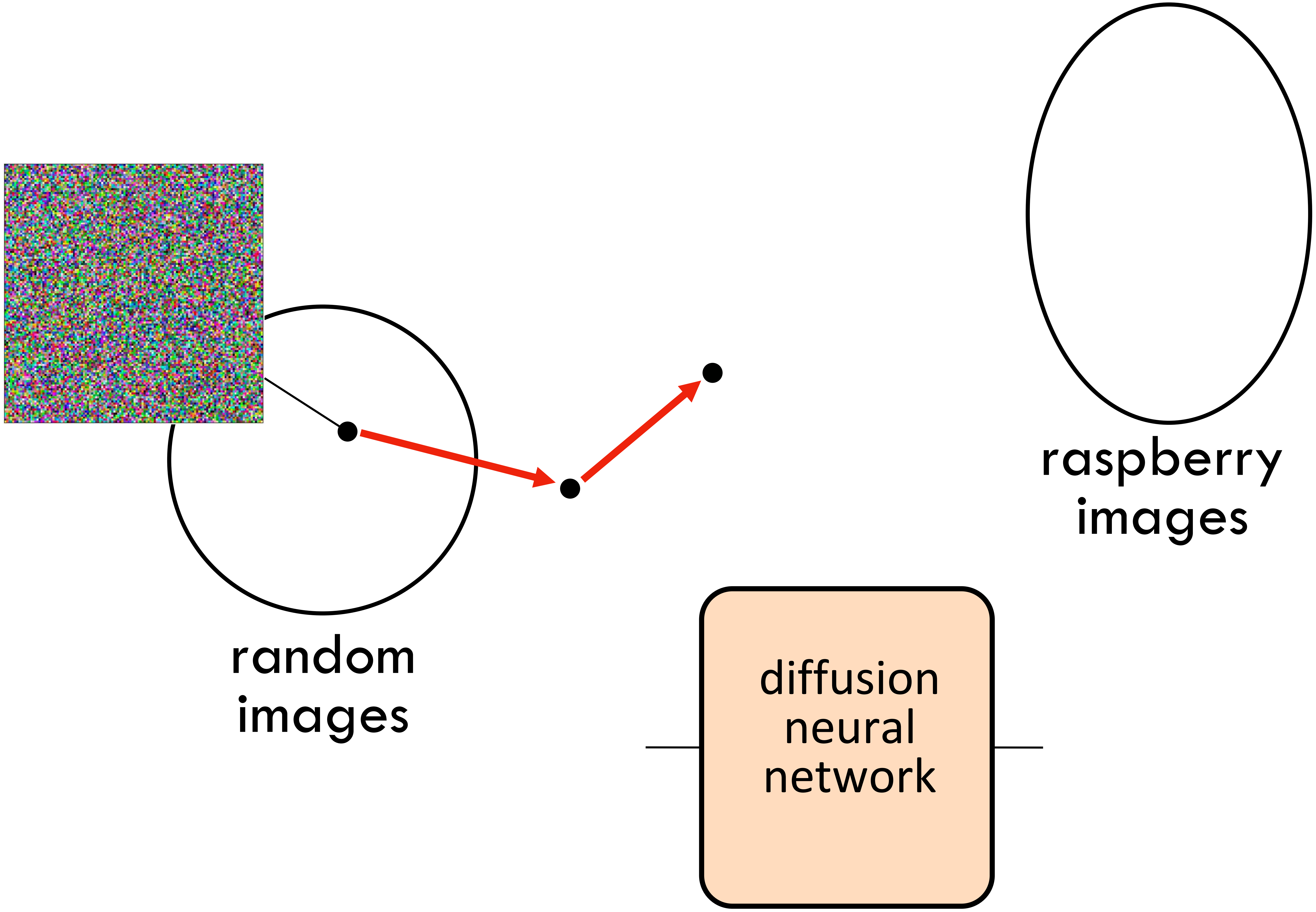
slide from Steve Seitz's video



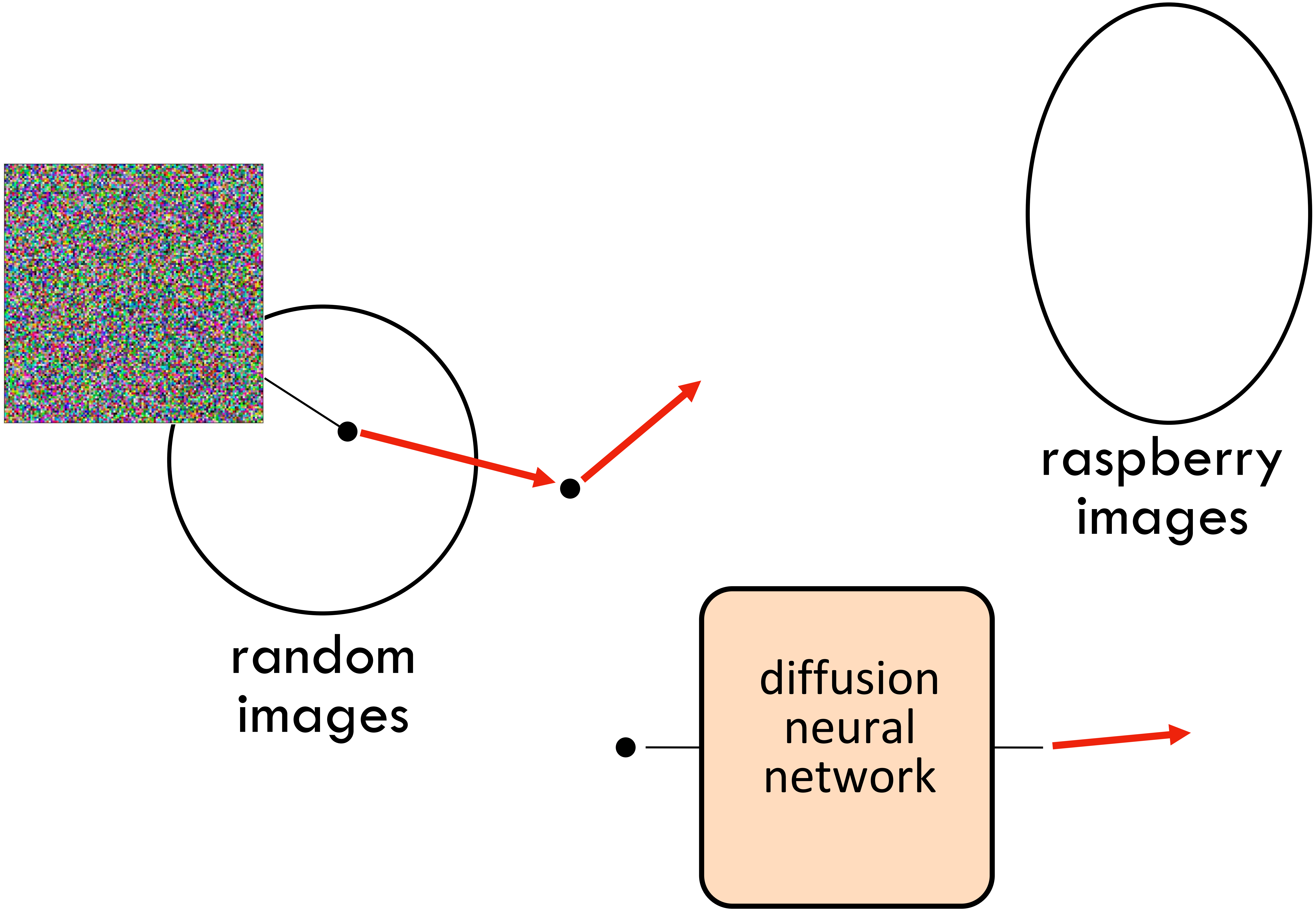
slide from Steve Seitz's video



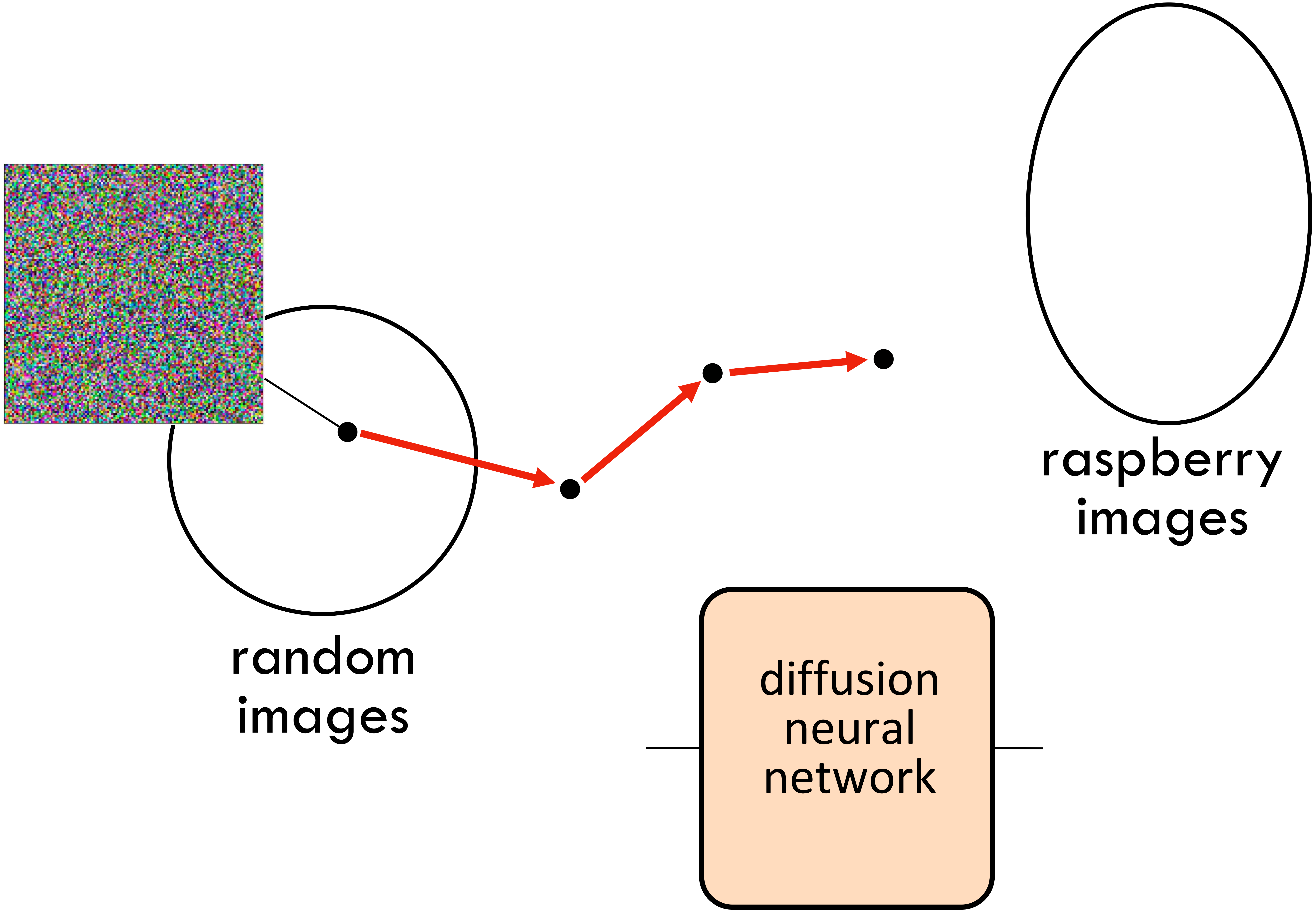
slide from Steve Seitz's [video]



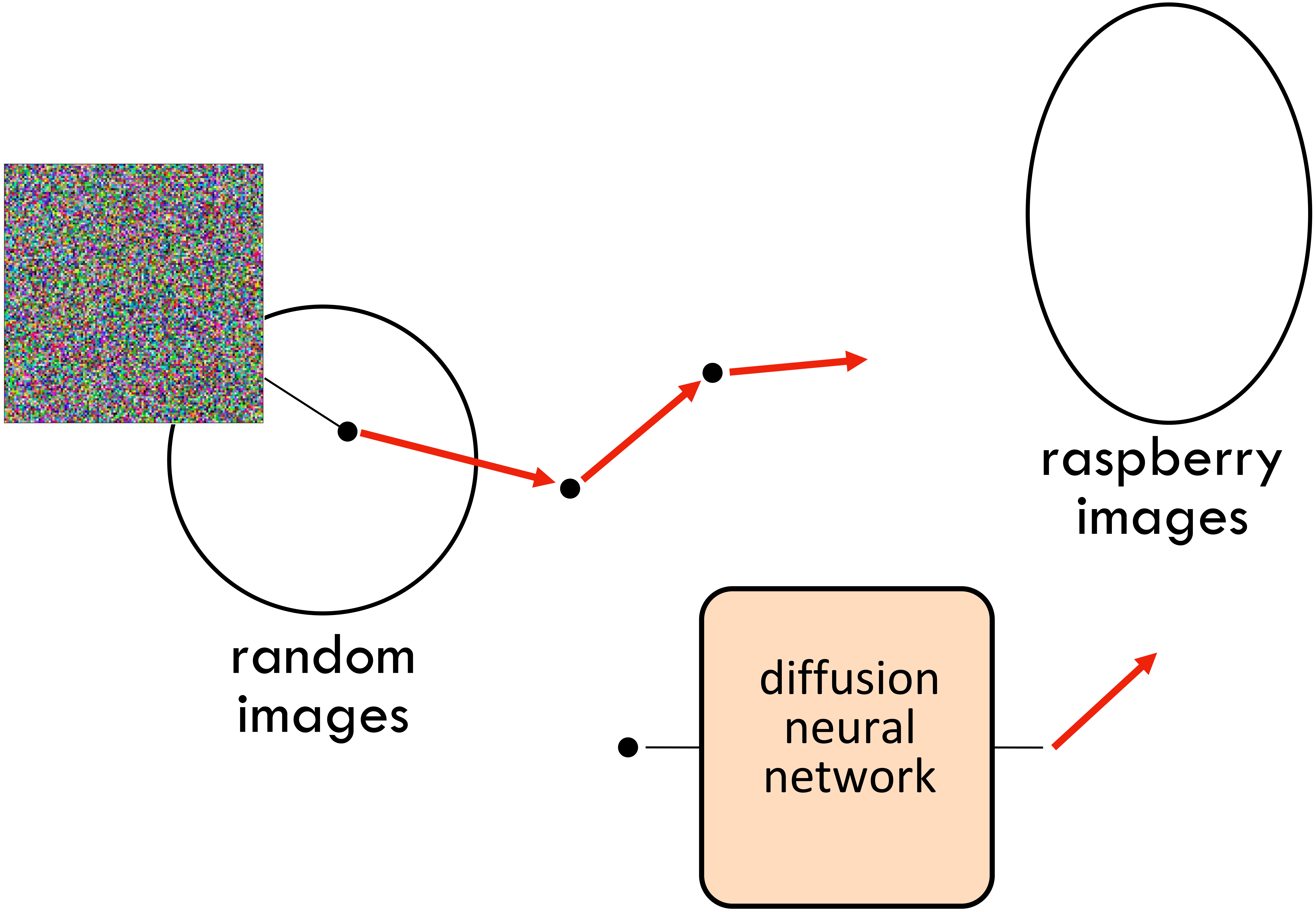
slide from Steve Seitz's video



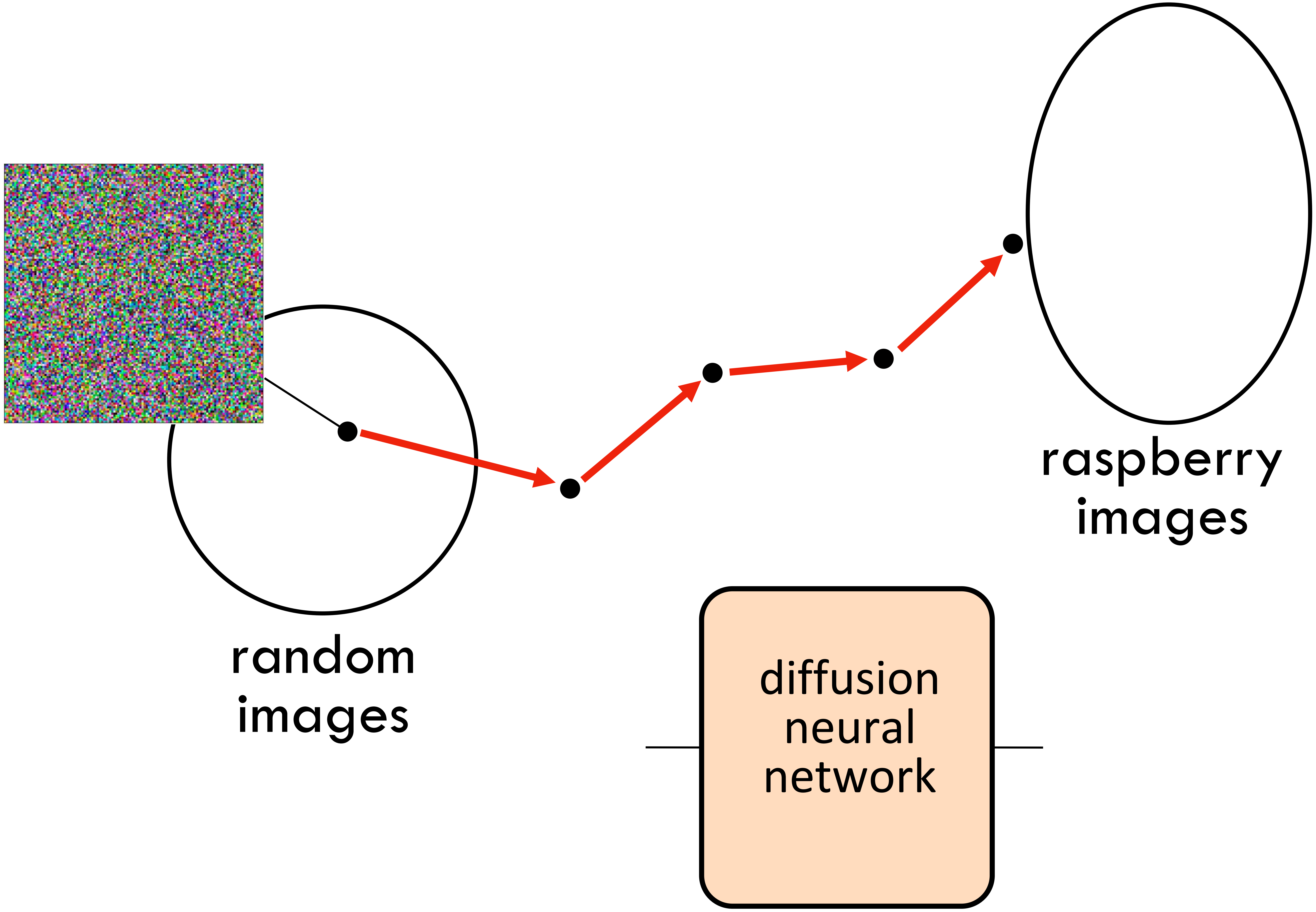
slide from Steve Seitz's video



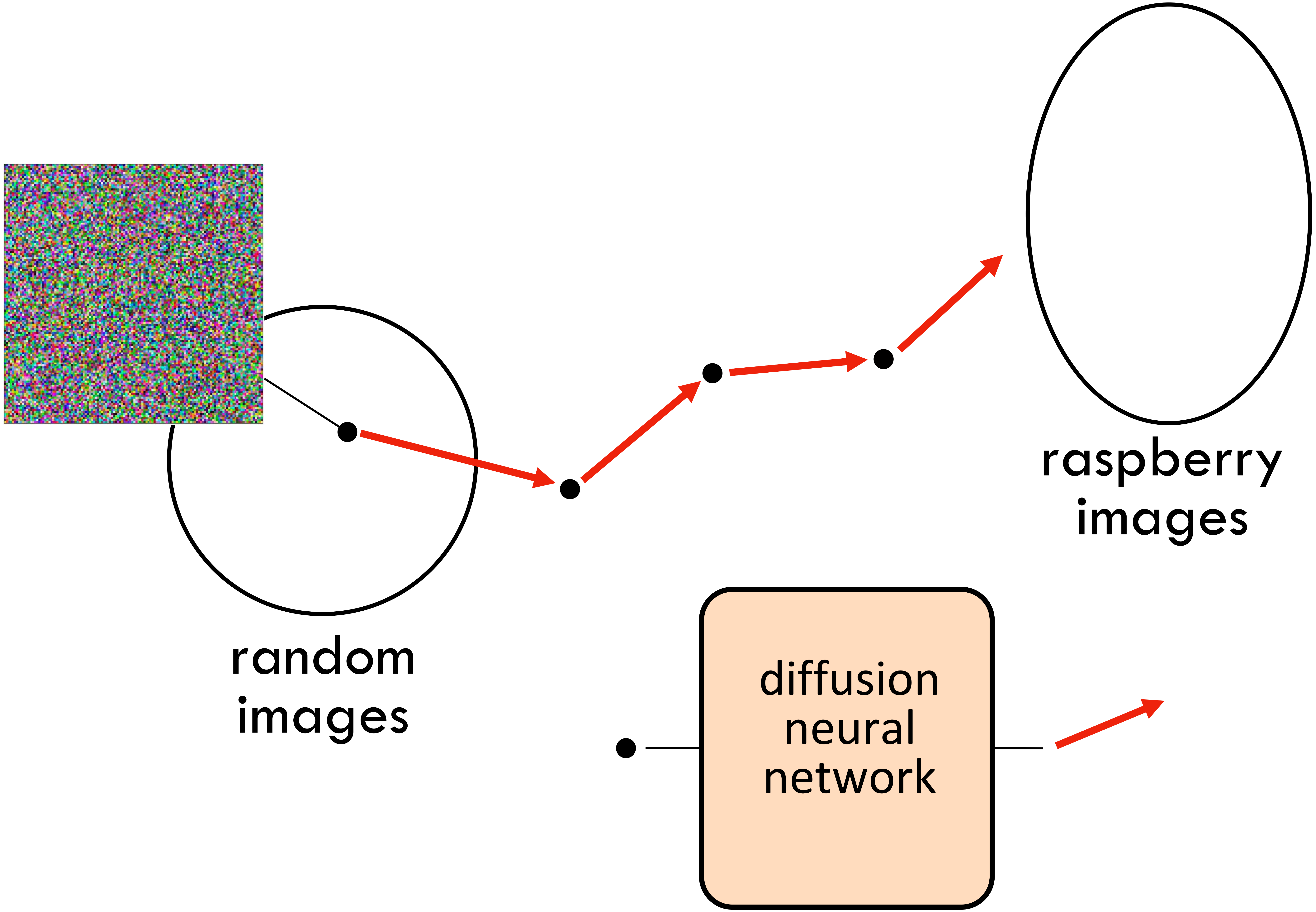
slide from Steve Seitz's video



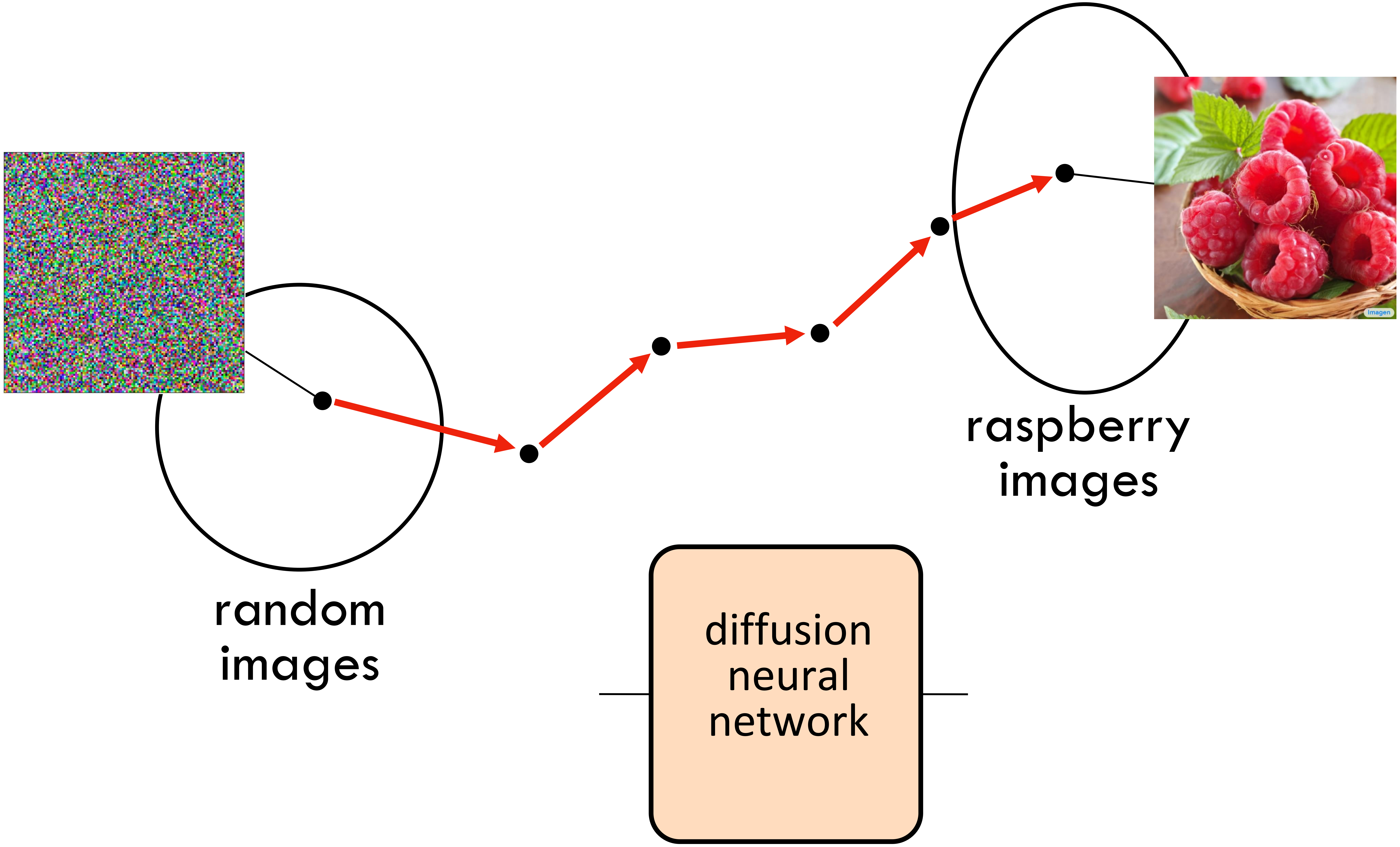
slide from Steve Seitz's video



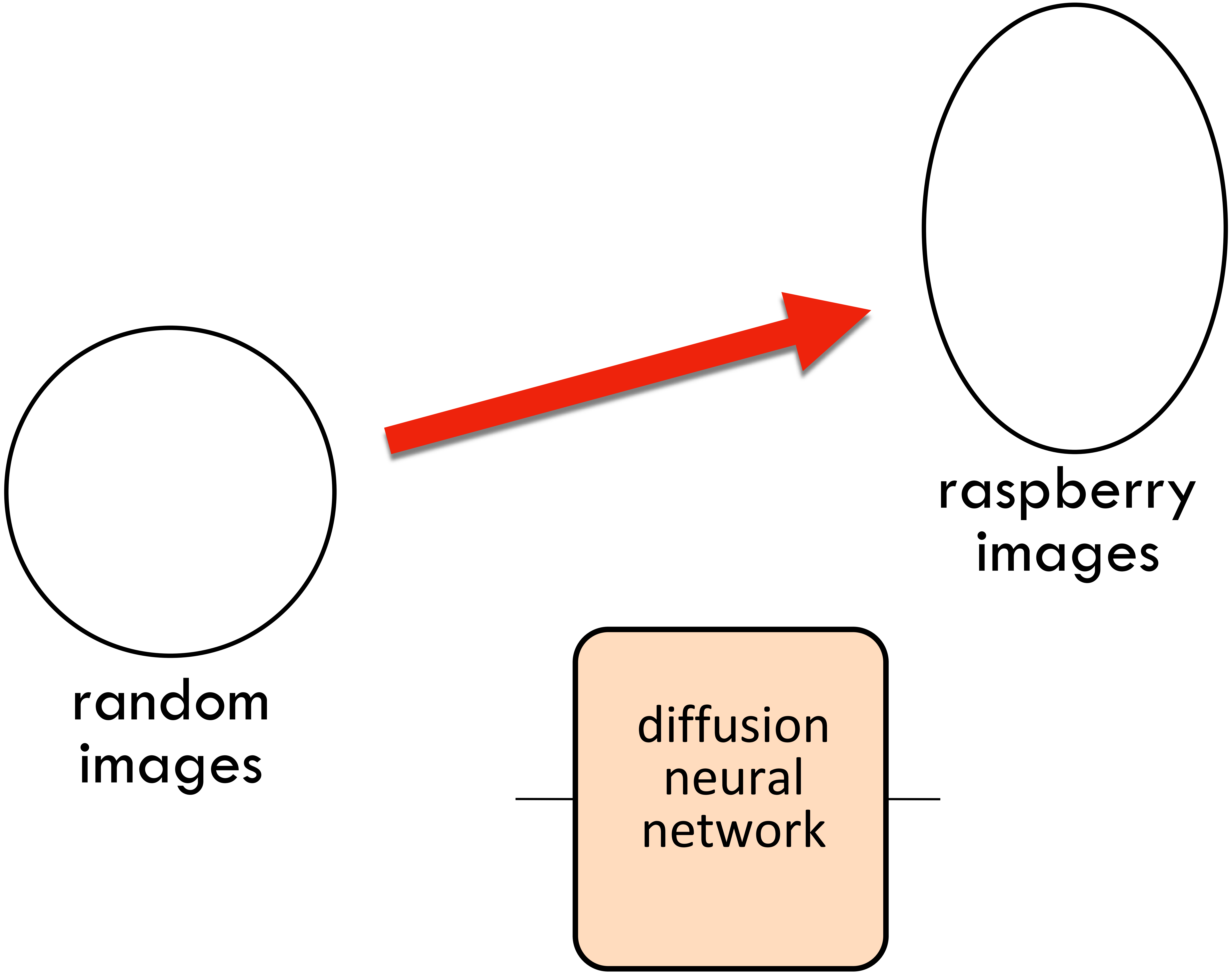
slide from Steve Seitz's [video]



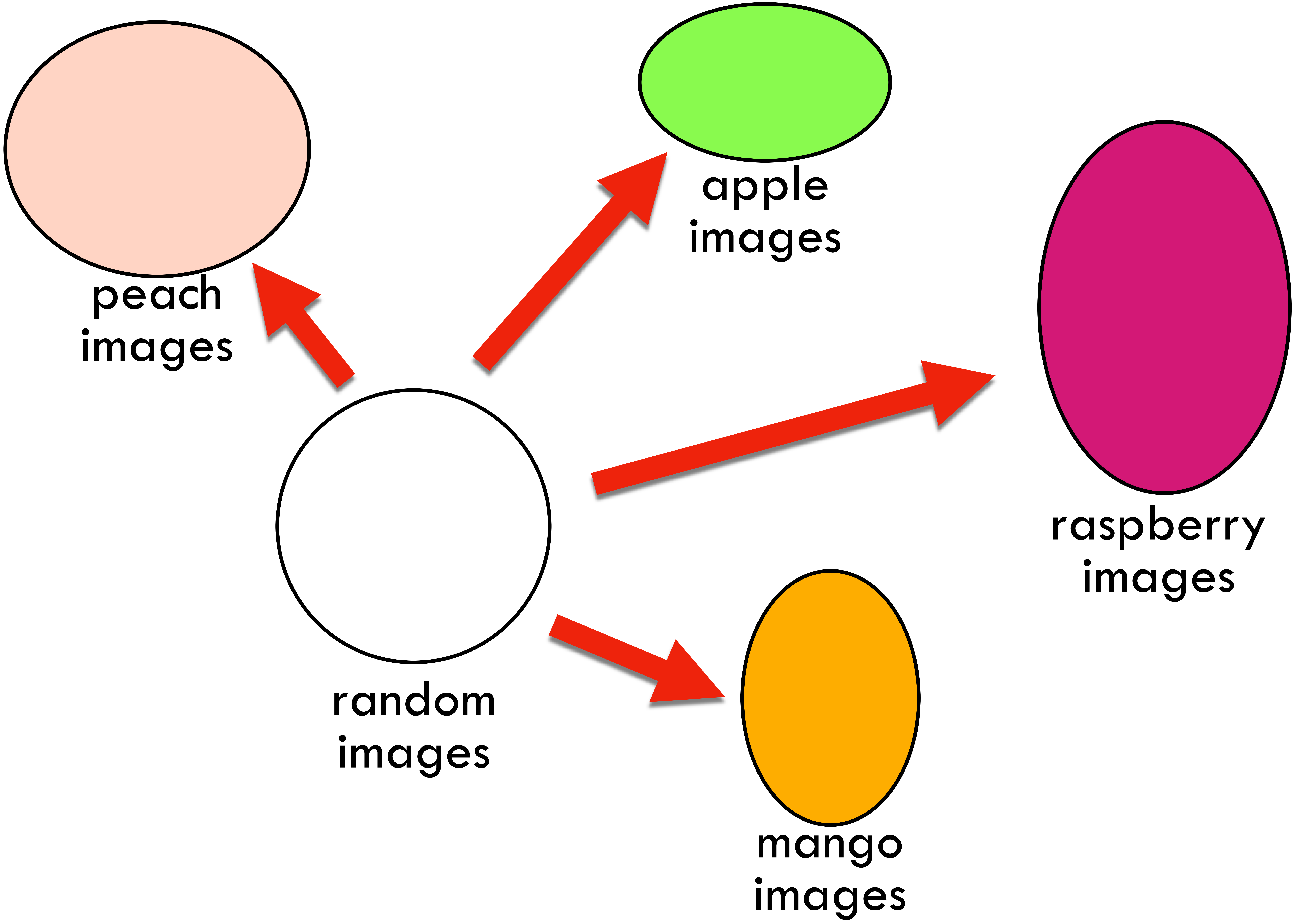
slide from Steve Seitz's [video]



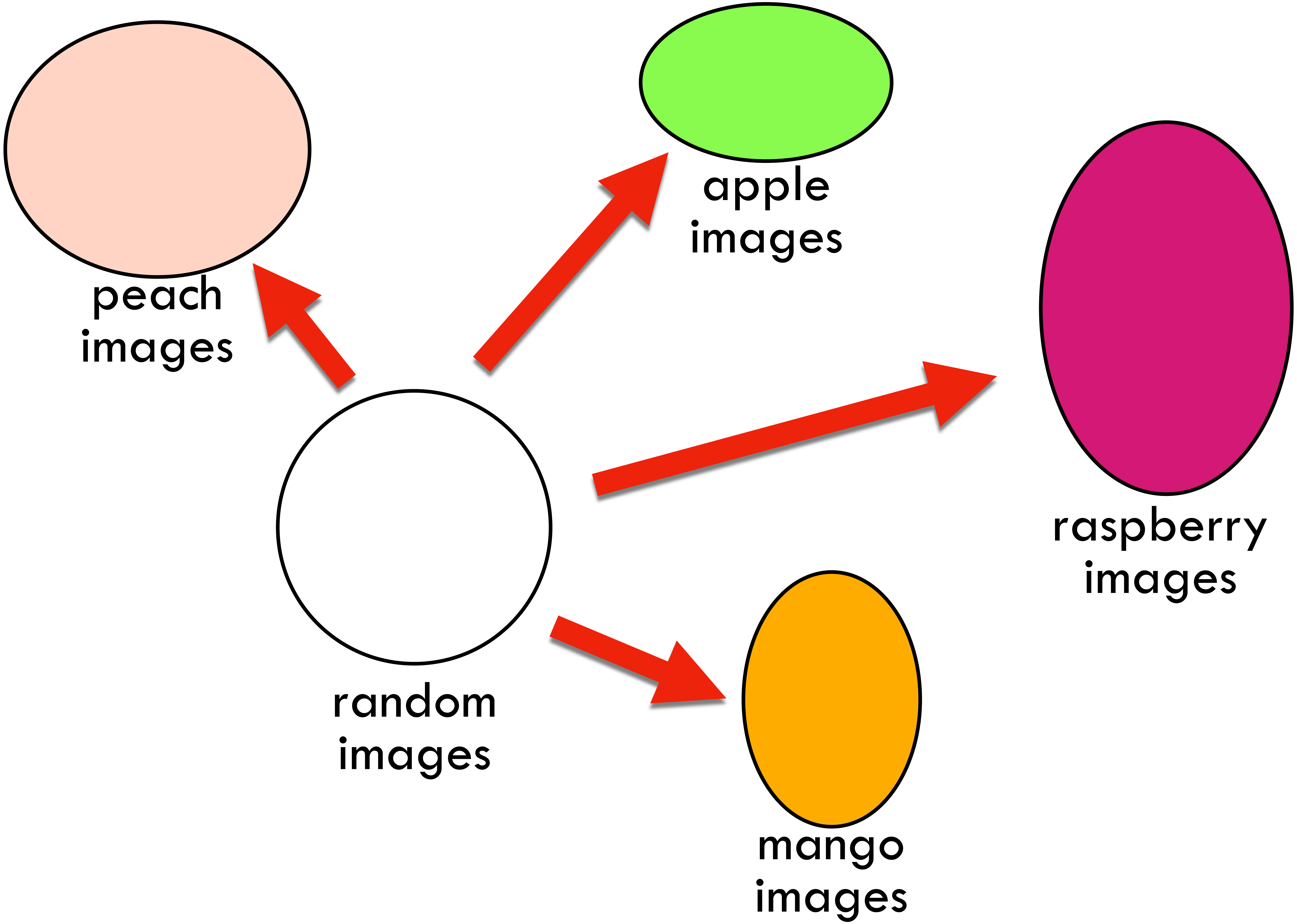
slide from Steve Seitz's video



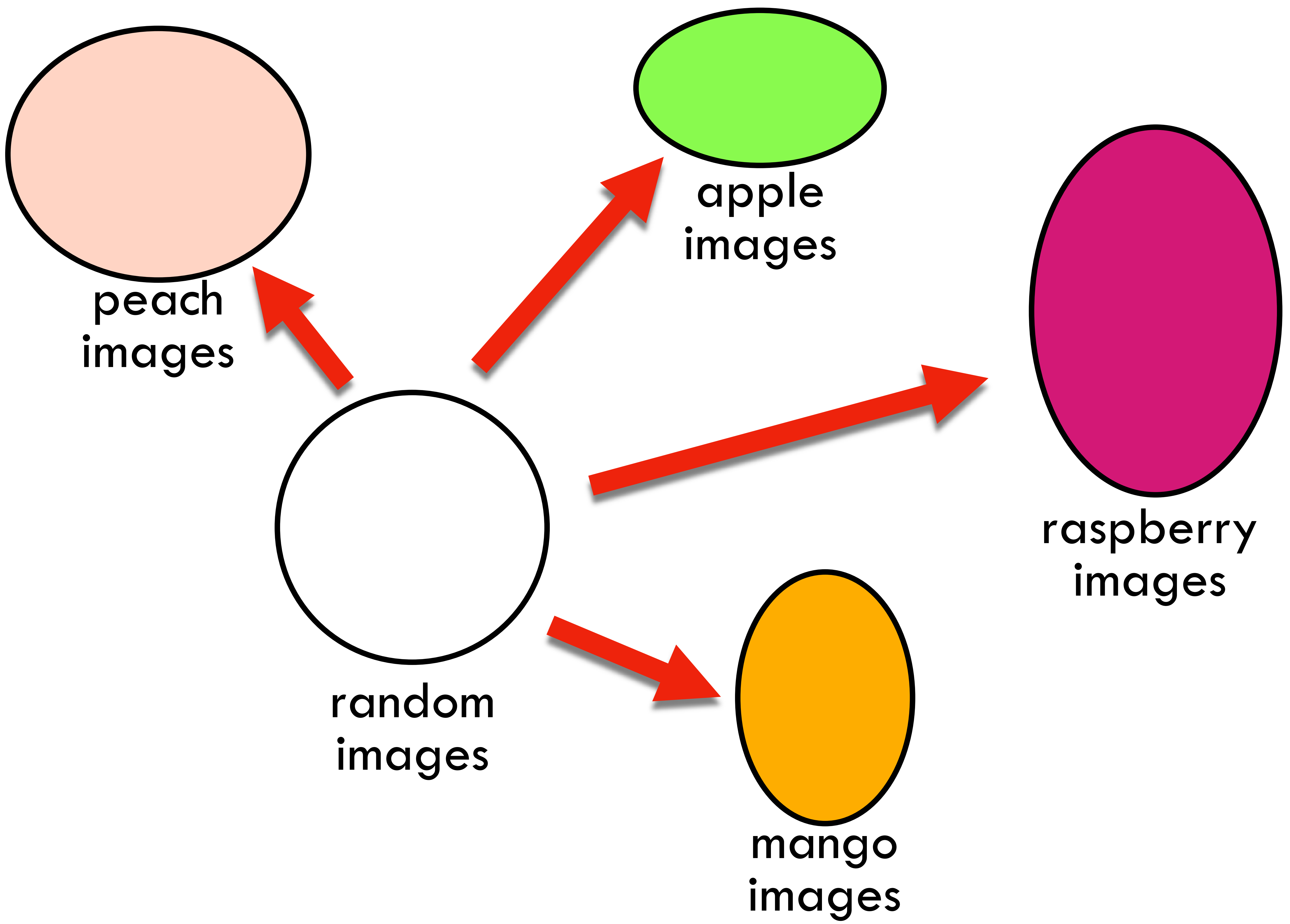
slide from Steve Seitz's [video]

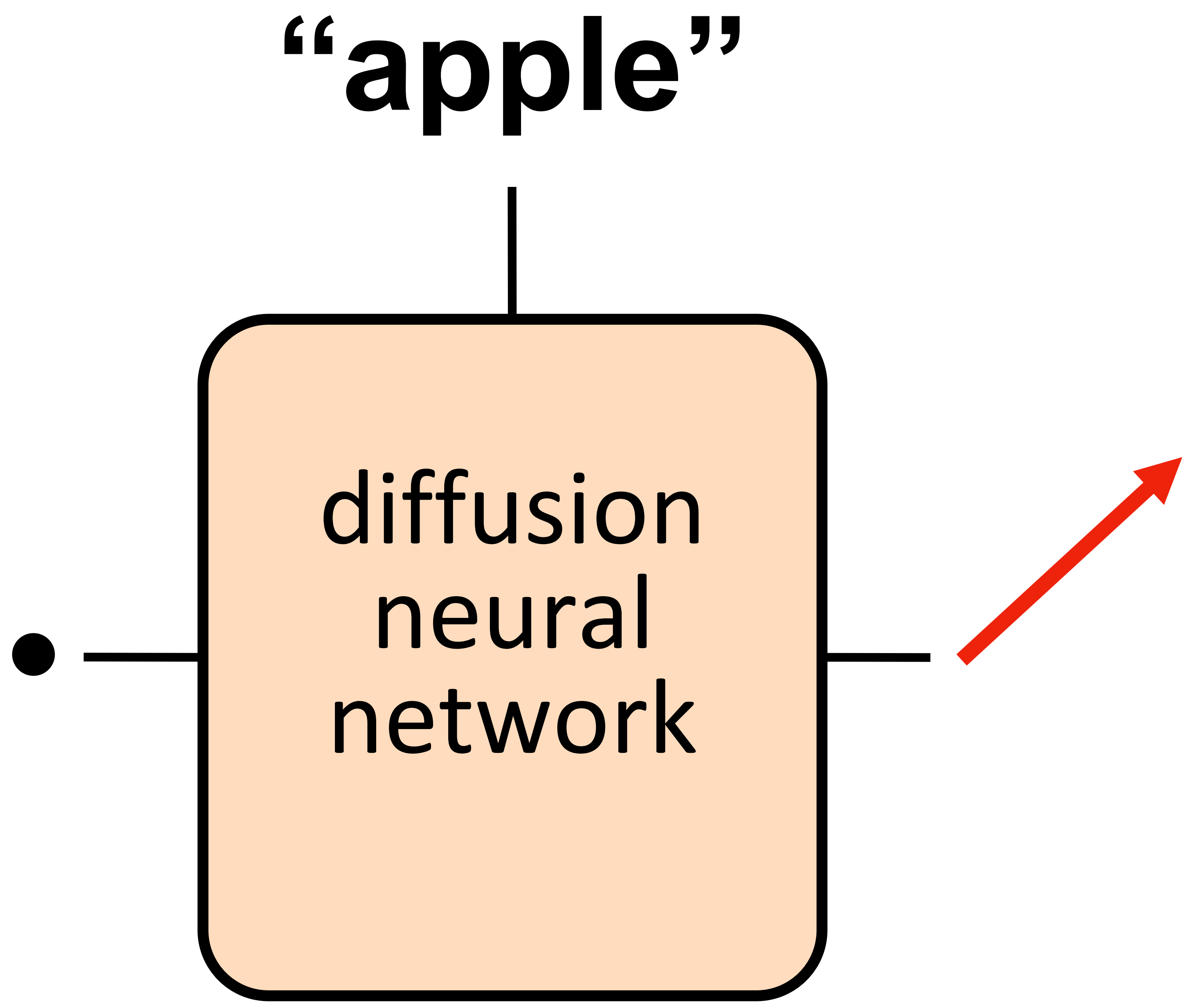


slide from Steve Seitz's video

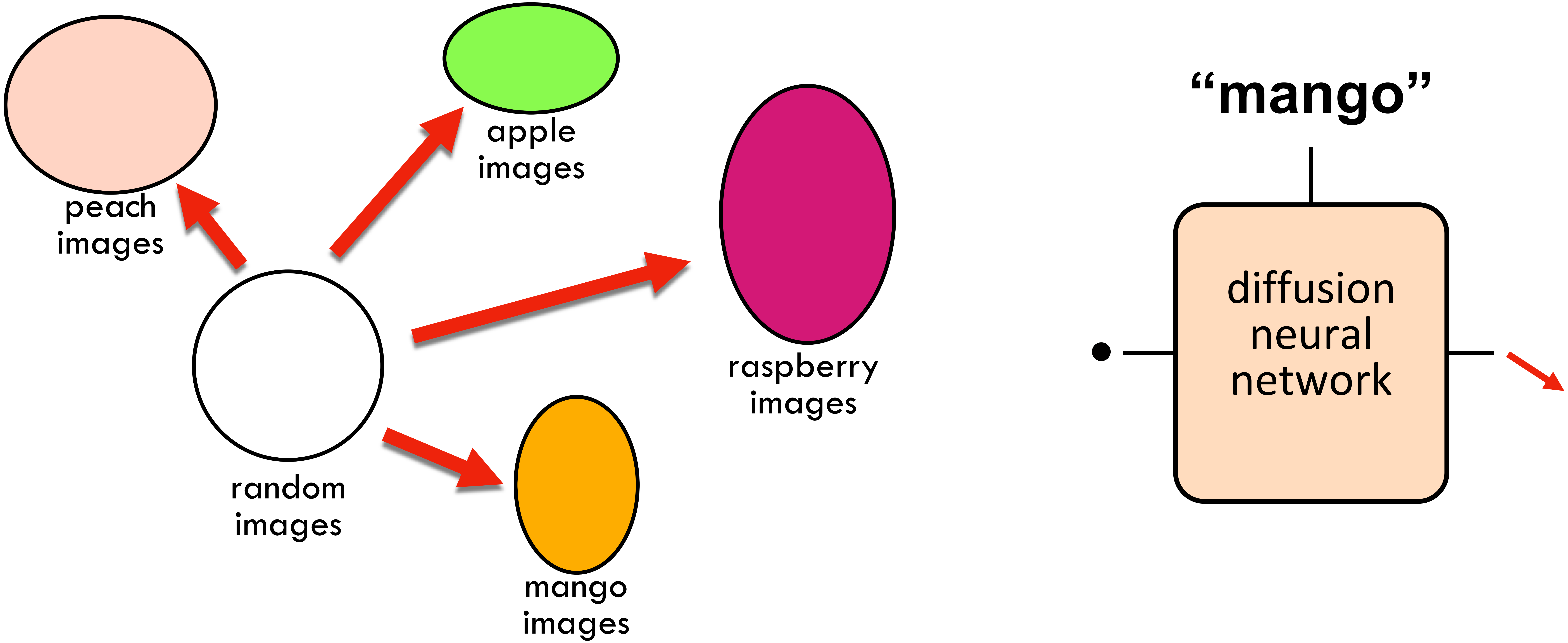


slide from Steve Seitz's [video]



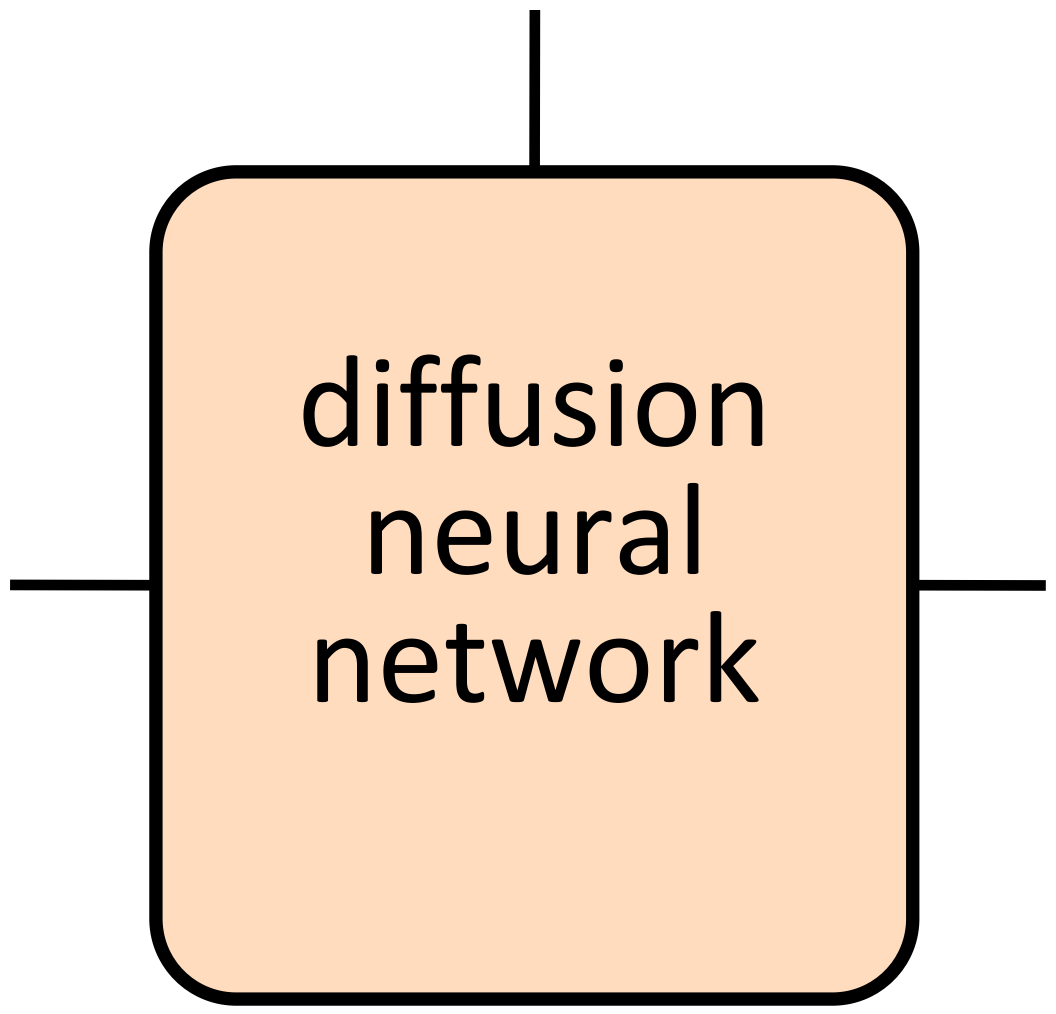


slide from Steve Seitz's [video]

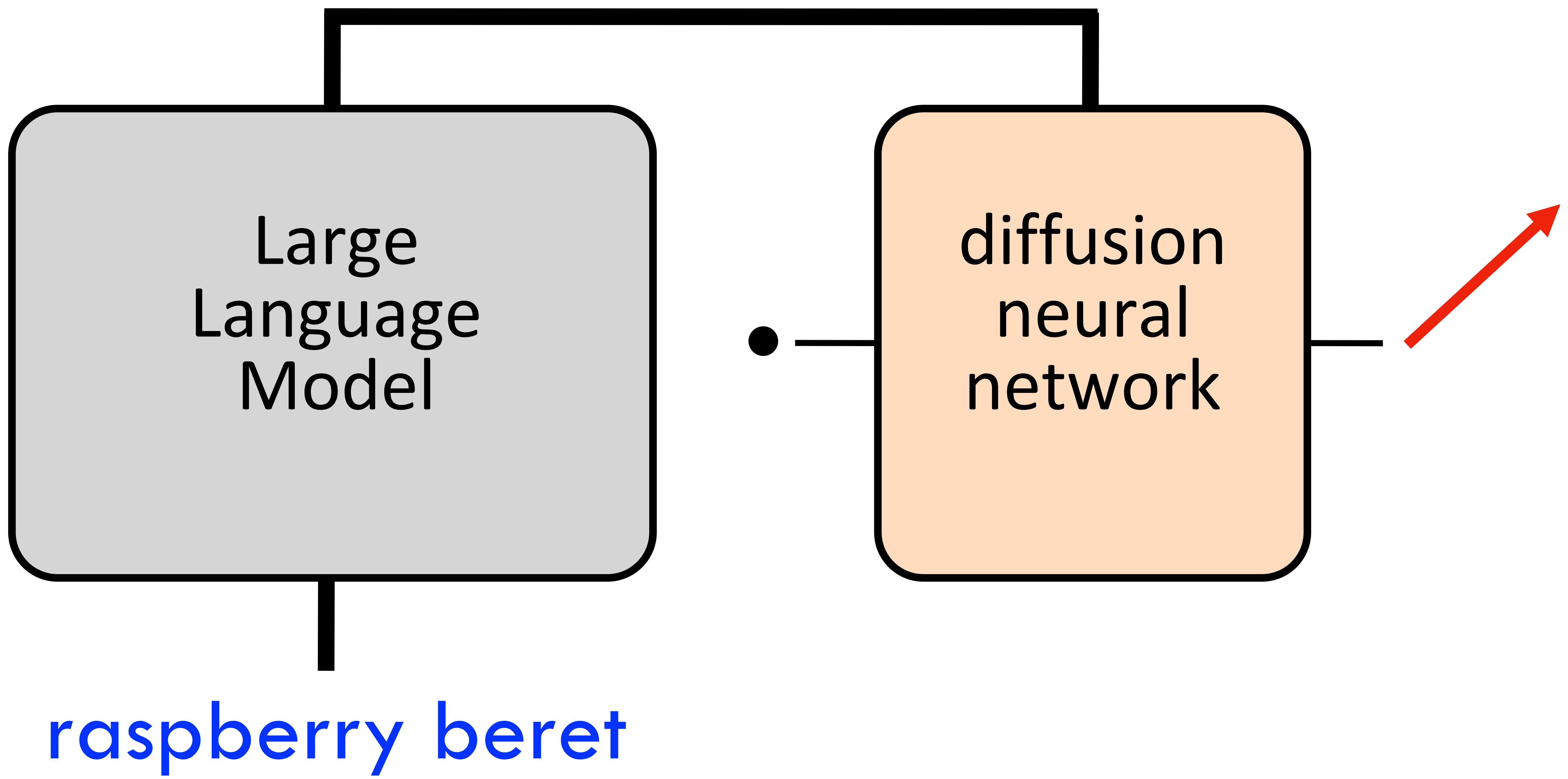


slide from Steve Seitz's [video]

raspberry beret



slide from Steve Seitz's [video]


Large
Language
Model
diffusion
neural
network
raspberry beret

raspberry beret

slide from Steve Seitz’s video

beret of raspberries

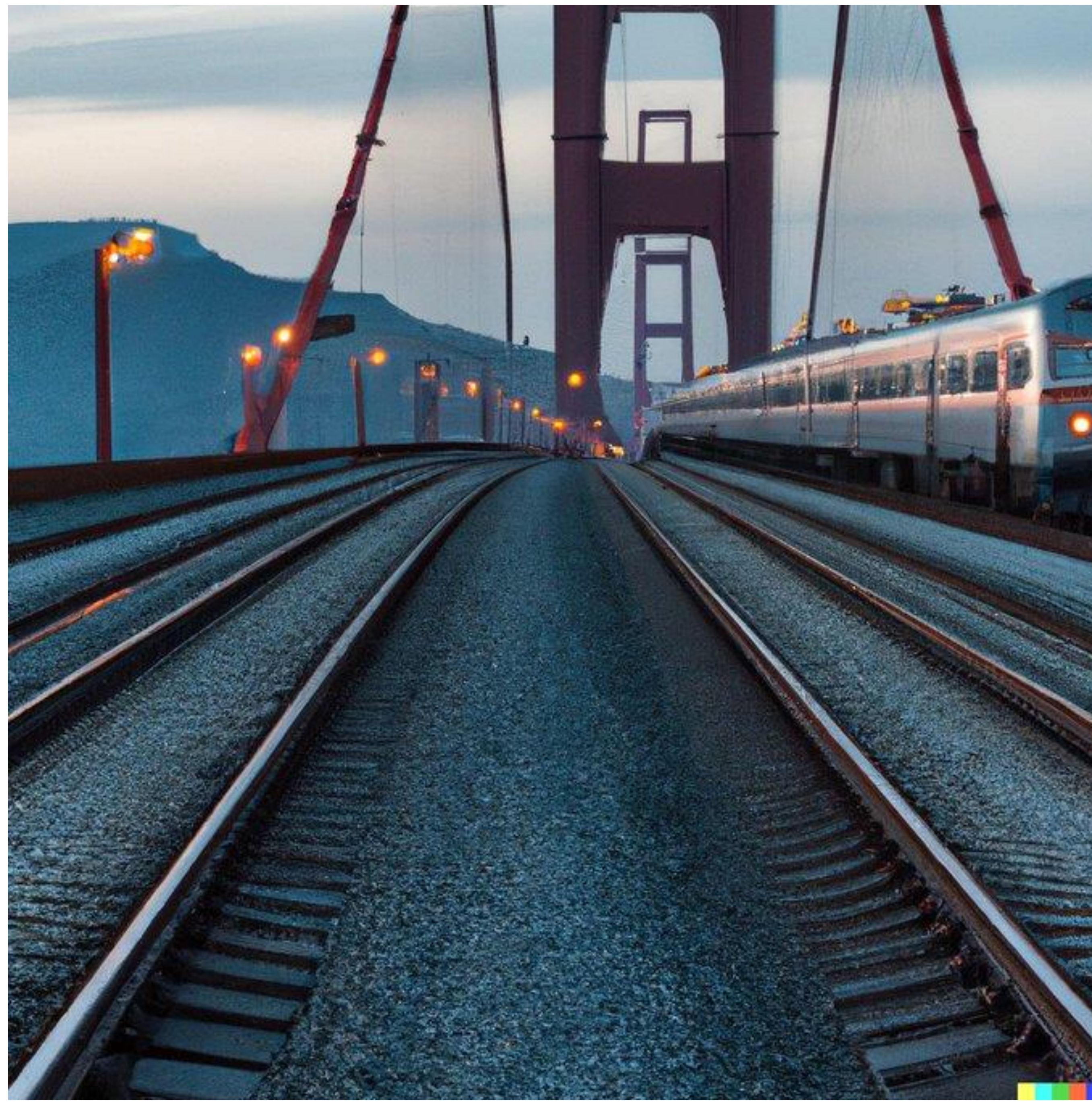
beret of raspberries

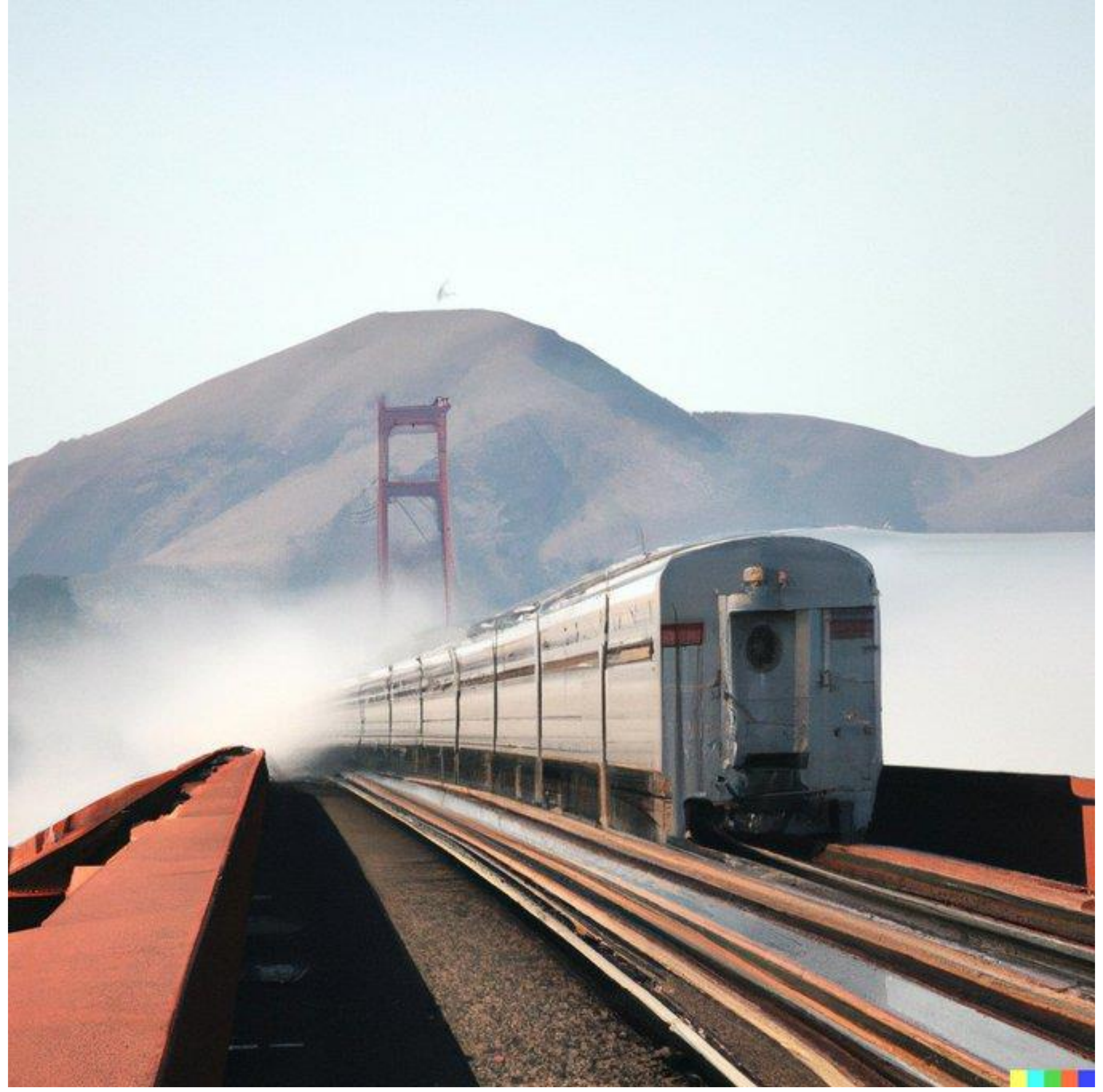
slide from Steve Seitz's video

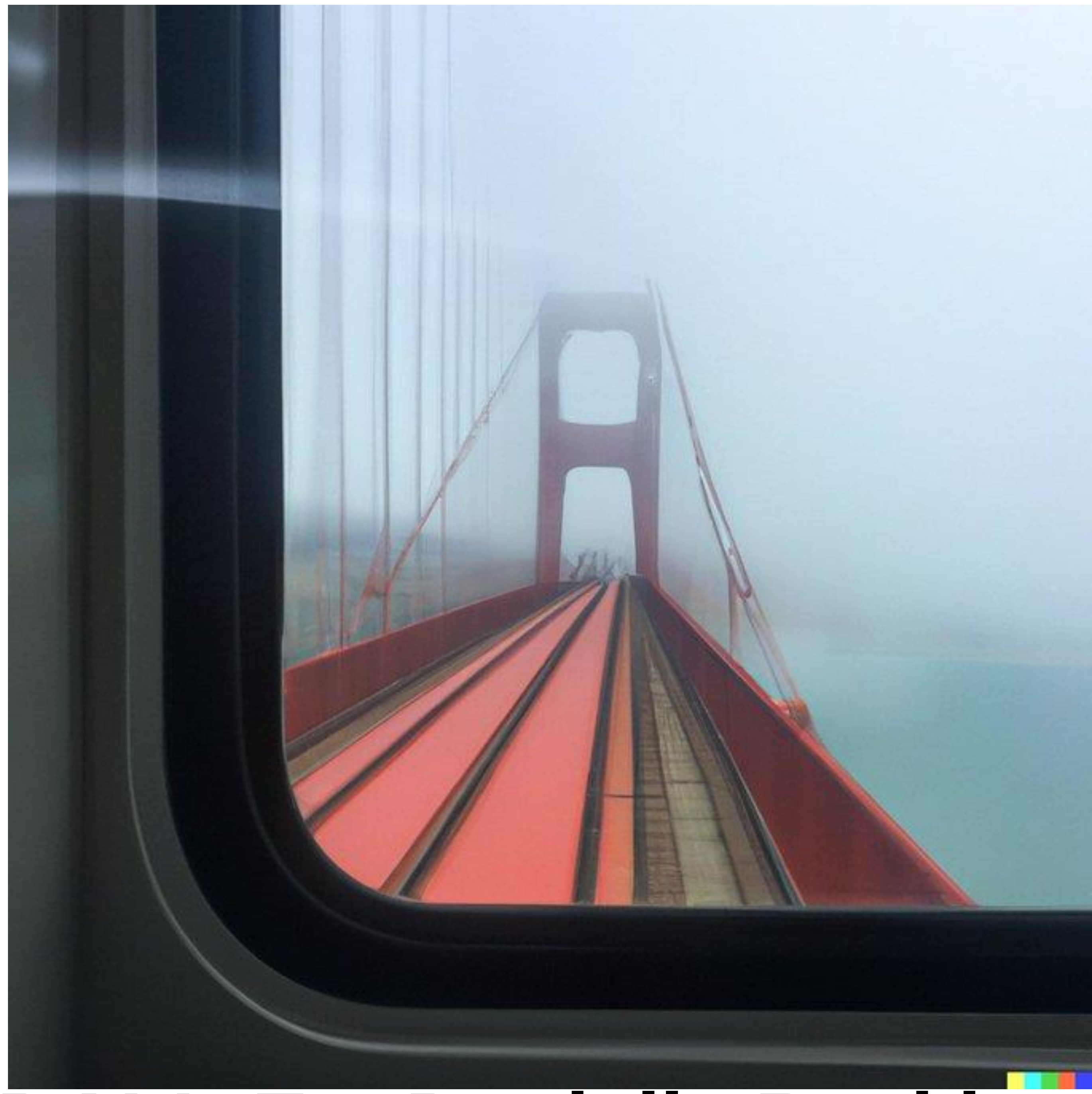
beret of raspberries

slide from Steve Seitz's video

Impressive compositionality:







DALL-E + Danielle Baskin

“Person holding a heavy box”

“Person holding a laptop”

“Person holding birthday cake”

“Person offering tea”

“Person holding a watermelon”

“Person holding a self-portrait”

“Person holding a painting of himself”