Today

Early vision on a single image

- **Textures**

Texture

The main difference between different textured surfaces: change in pattern elements and repetitions

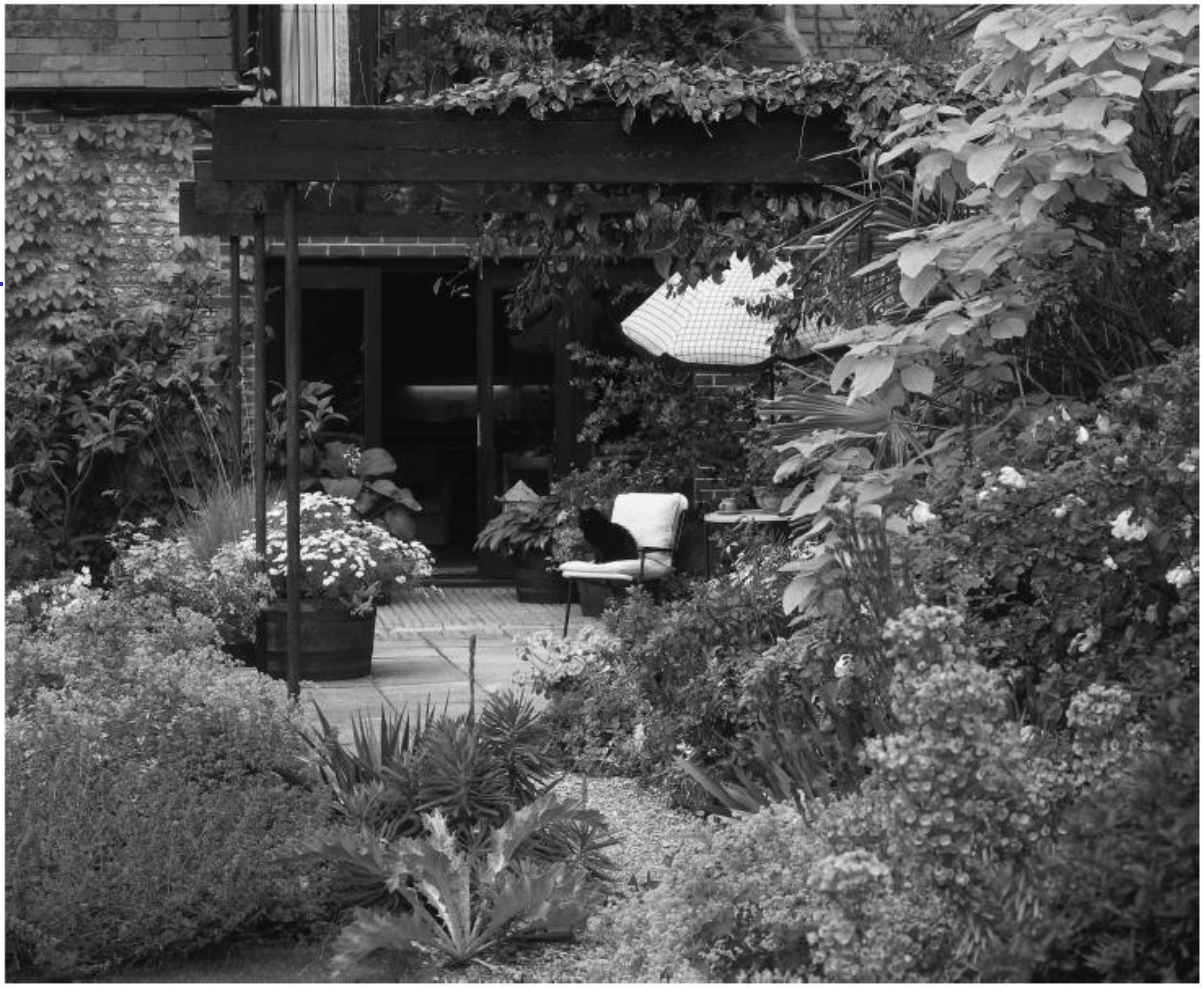


FIGURE 6.1: Although texture is difficult to define, it has some important and valuable properties. In this image, there are many repeated elements (some leaves form repeated "spots"; others, and branches, form "bars" at various scales; and so on). Our perception of the material is quite intimately related to the texture (what would the surface feel like if you ran your fingers over it? what is soggy? what is prickly? what is smooth?). Notice how much information you are getting about the type of plants, their shape, the shape of free space, and so on, from the textures. *Geoff Brightling © Dorling Kindersley, used with permission.*

Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"

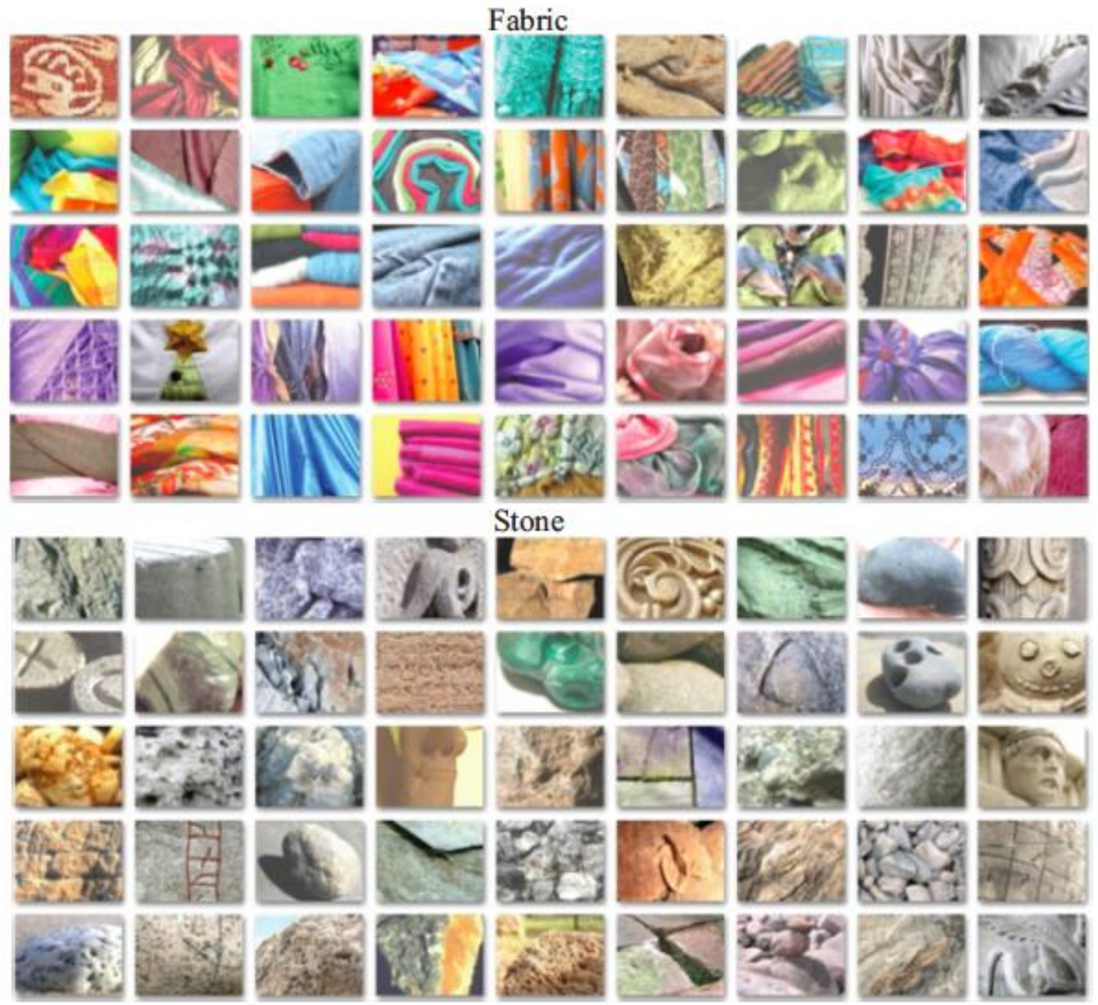


FIGURE 6.2: Typically, different materials display different image textures. These are example images from a collection of 1,000 material images, described in by Sharan *et al.* (2009); there are 100 images in each of the ten categories, including the two categories shown here (fabric and stone). Notice how (a) the textures vary widely, even within a material category; and (b) different materials seem to display quite different textures. *This figure shows elements of a database collected by C. Liu, L. Sharan, E. Adelson, and R. Rosenholtz, and published at* `http://people.csail.mit.edu/lavanya/research_sharan.html`. *Figure by kind permission of the collectors.*

Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"

Texture

Patterns of structure from

- changes in surface albedo (eg printed cloth)
- changes in surface shape (eg bark)
- many small surface patches (eg leaves on a bush)

Hard to define; but texture tells us

- what a surface is like
- (sometimes) object identity
- (sometimes) surface shape

Texture

Core problems:

- Texture segmentation
- Texture based recognition
  - Objects, materials, textures
- Texture synthesis
  - Create synthetic images, fill in holes, image editing using computer graphics
  - Give some insight into quality of representation
- Shape from texture

Key issue: representing texture

Representing Textures

Textures are made up of

- stylized subelements
- Spatially repeated in meaningful ways

Representation:

  - find the subelements
  - represent their statistics

Representing Textures

But what are the subelements, and how do we find them?

- find subelements by applying filters, looking at the magnitude of the response
- Extreme case – template matching by normalized cross correlation $\frac{1}{MN}\sum_{x,y}\frac{(f(x,y)-\bar{f})(T(x,y)-\bar{T})}{\sigma_f\sigma_T}$

Representing Textures

What filters?

- experience suggests **spots** and **oriented bars** at a variety of different scales
- Each filter corresponds to one pattern element

What statistics?

- within reason, the more the merrier.
- At least, mean and standard deviation
- better, various conditional histograms.

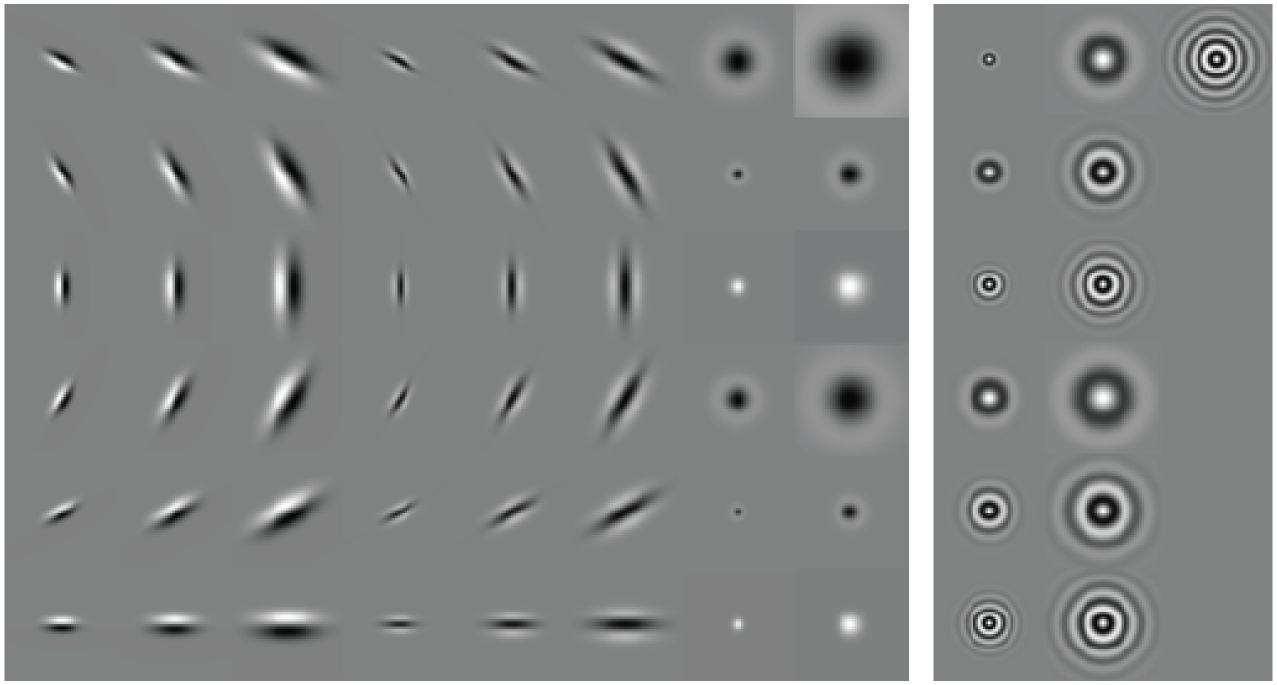
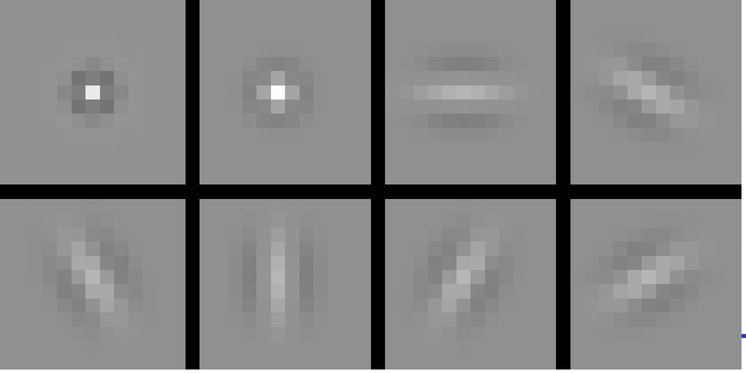
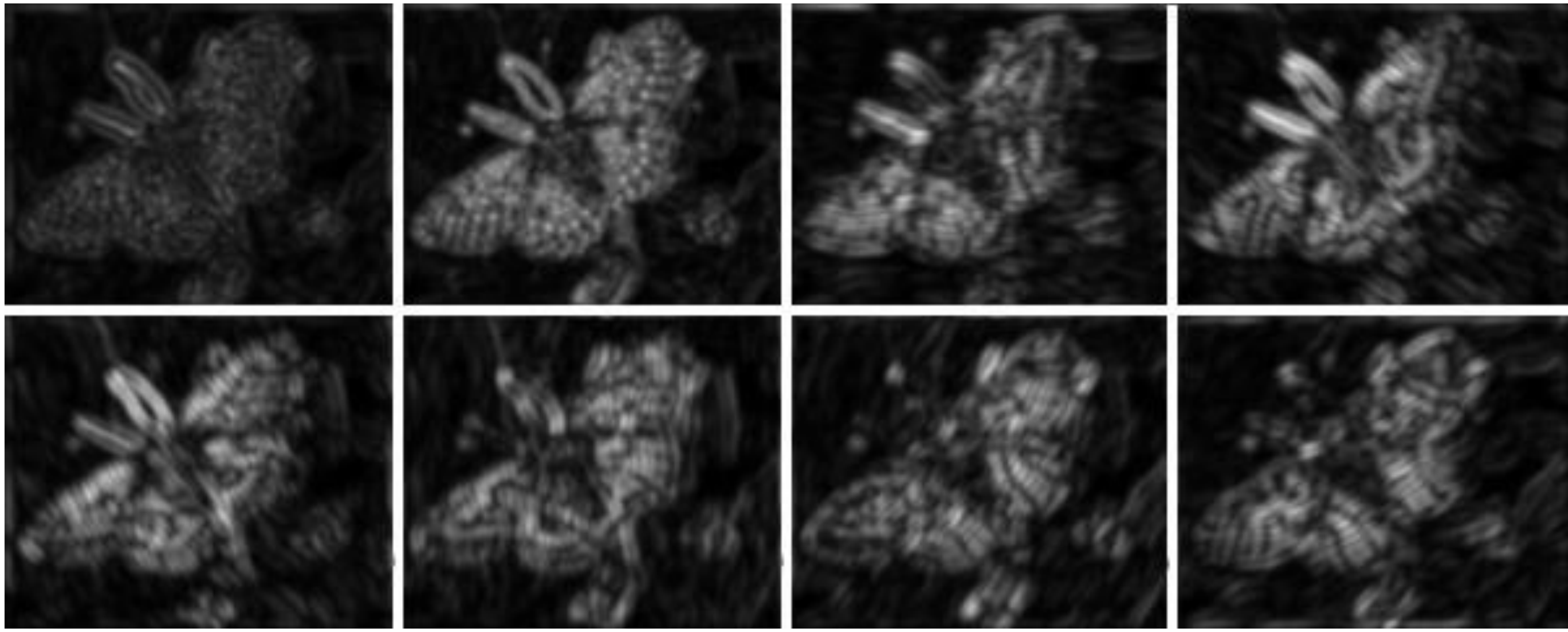


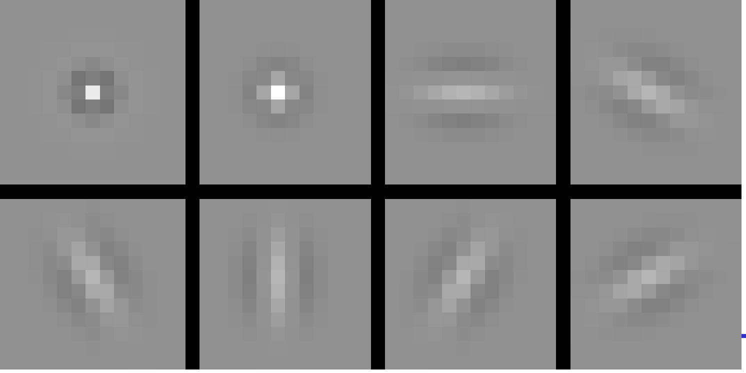
FIGURE 6.4: **Left** shows a set of 48 oriented filters used for expanding images into a series of responses for texture representation. Each filter is shown on its own scale, with zero represented by a mid-gray level, lighter values being positive, and darker values being negative. The left three columns represent edges at three scales and six orientations; the center three columns represent stripes; and the right two represent two classes of spots (with and without contrast at the boundary) at different scales. This is the set of filters used by Leung and Malik (2001). **Right** shows a set of orientation-independent filters, used by Schmid (2001), using the same representation (there are only 13 filters in this set, so there are five empty slots in the image). The orientation-independence property means that these filters look like complicated spots.

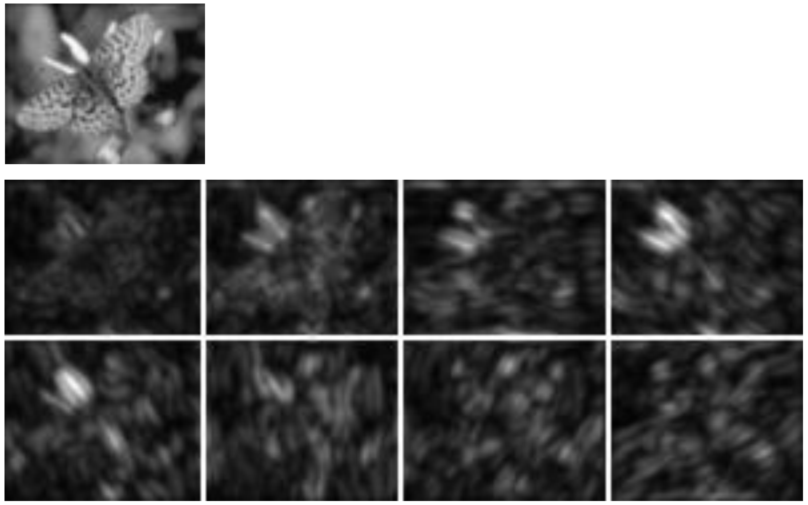
Forsyth and Ponce, “Computer Vision – A Modern Approach 2e”



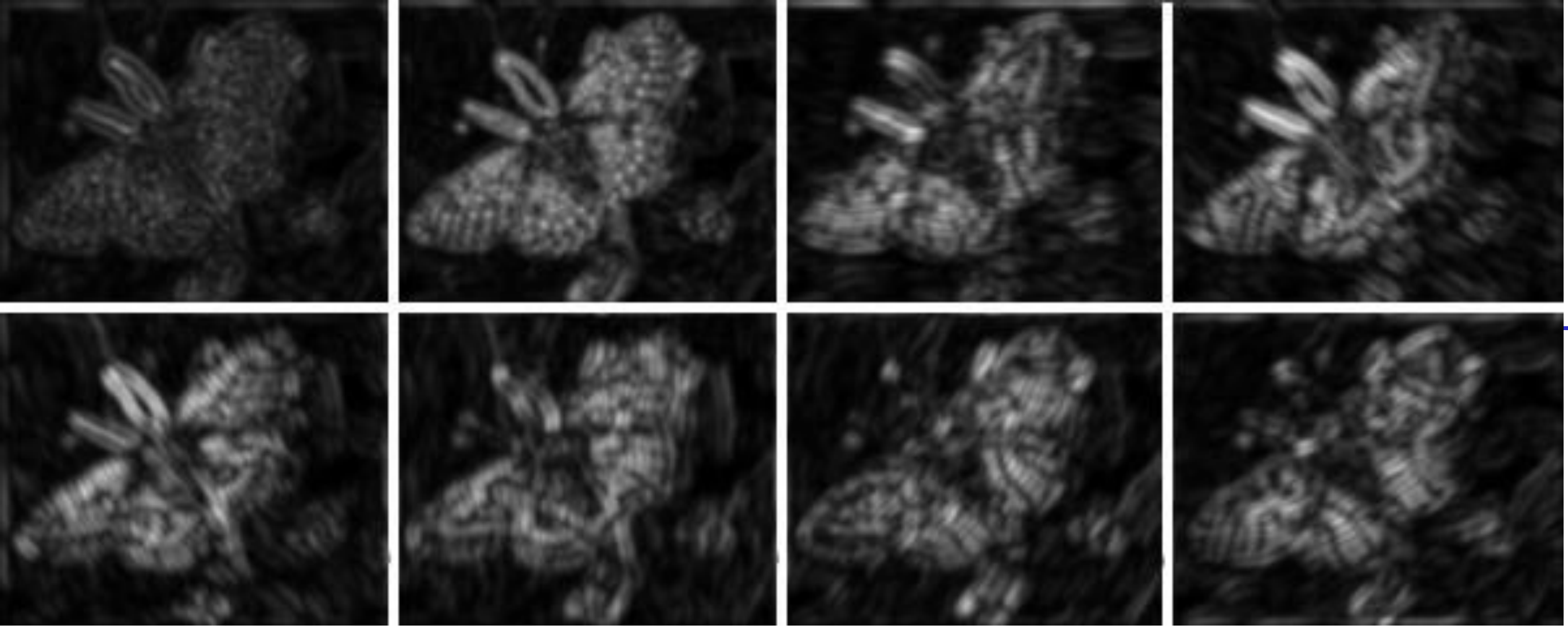
Squared response



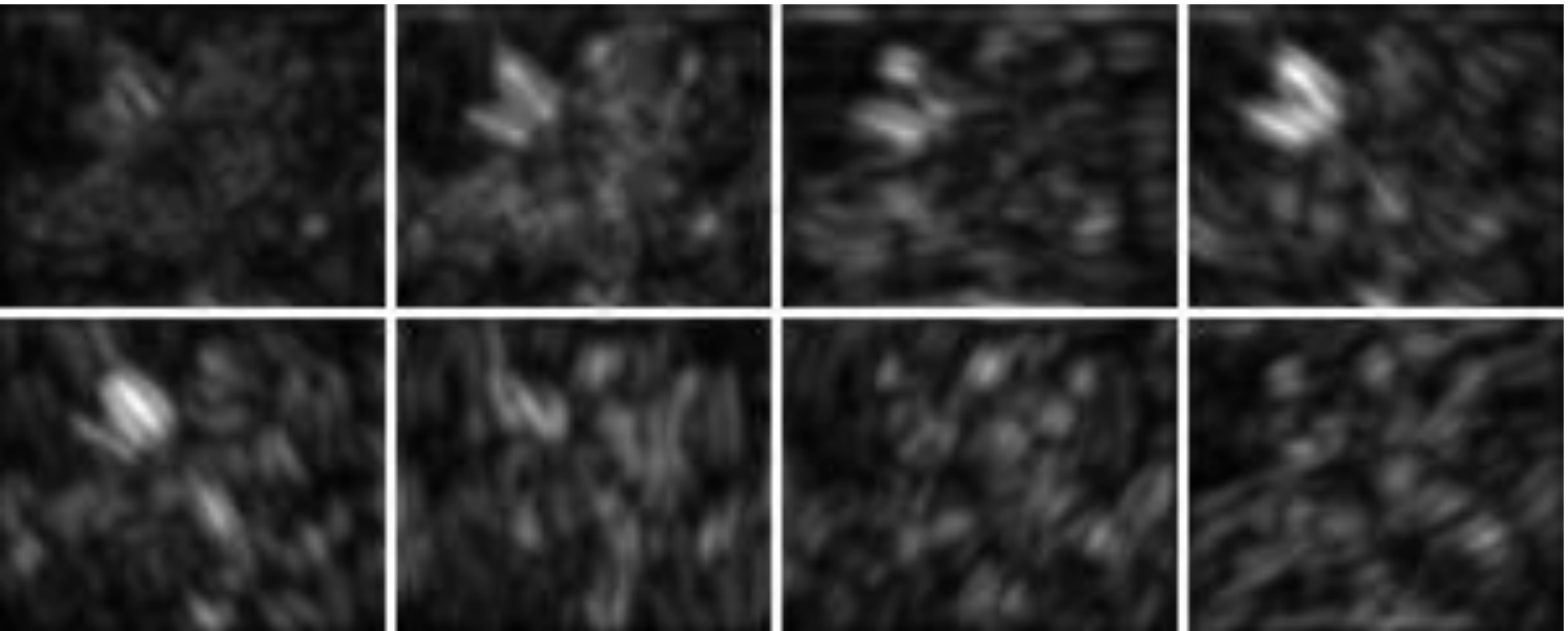




squared response



Fine



Coarse

Final Texture Representation

Form an oriented pyramid (or equivalent set of responses to filters at different scales and orientations).

Square the output

Take statistics of responses

- e.g. mean of each filter output (are there lots of spots)
- std of each filter output

Example based Texture Representations

How does one choose the filters?

Solution

- build a dictionary of subelements from pictures
- describe the image using this dictionary

Building a Dictionary

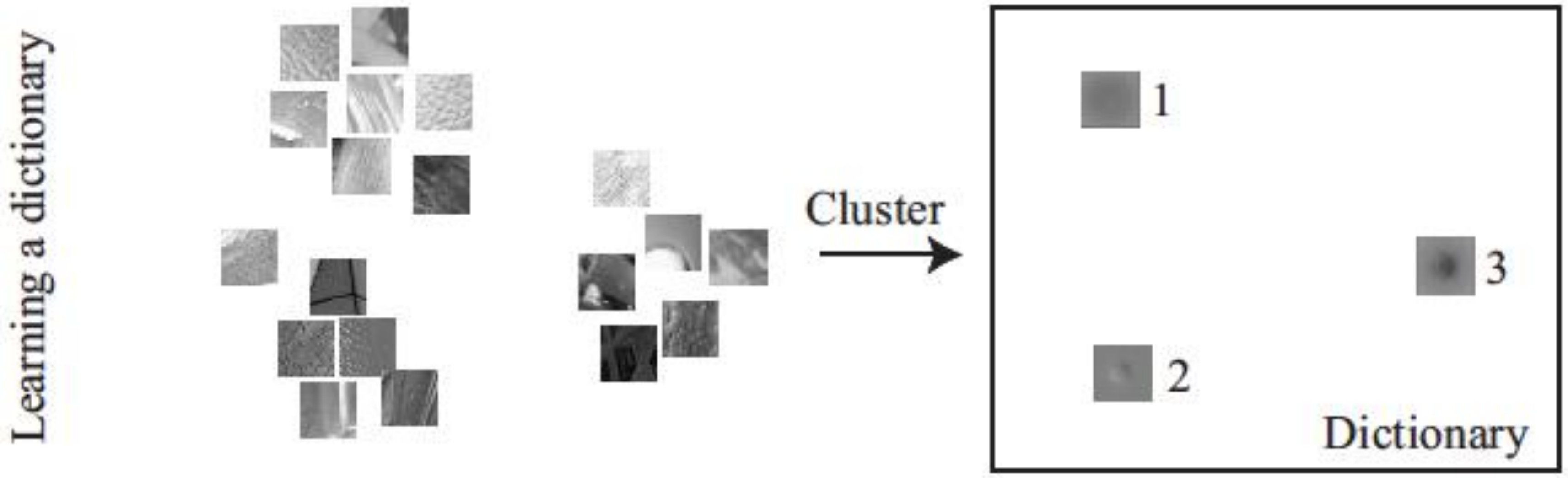


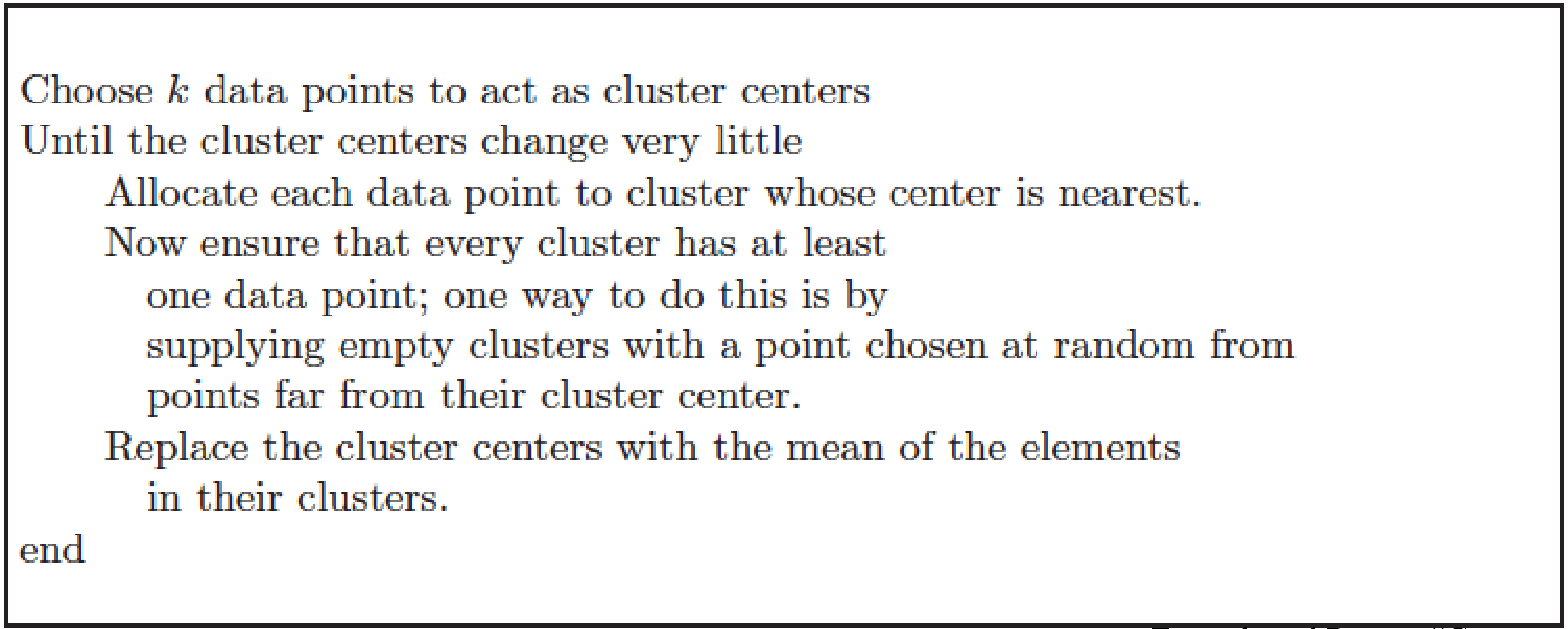
FIGURE 6.8: There are two steps to building a pooled texture representation for a texture in an image domain. First, one builds a dictionary representing the range of possible pattern elements, using a large number of texture patches. This is usually done in advance, using a training data set of some form.

Forsyth and Ponce, “Computer Vision – A Modern Approach 2e”

Clustering the Examples

K-means

- represent patches with
  - intensity vector
  - vector of filter responses over patch



```
Choose $k$ data points to act as cluster centers
Until the cluster centers change very little
    Allocate each data point to cluster whose center is nearest.
    Now ensure that every cluster has at least
      one data point; one way to do this is by
      supplying empty clusters with a point chosen at random from
      points far from their cluster center.
    Replace the cluster centers with the mean of the elements
      in their clusters.
end
```

Algorithm 6.3: Clustering by K-Means.

Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"

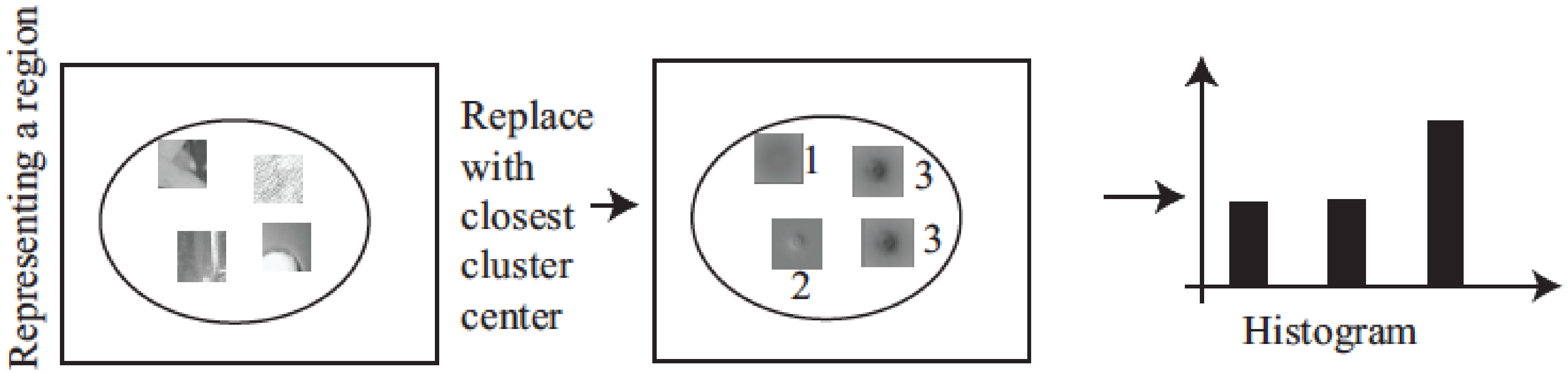
Representing a Region

Vector Quantization

- Represent a high-dimensional data item with a single number
- Find and use the index of the nearest cluster center in dictionary

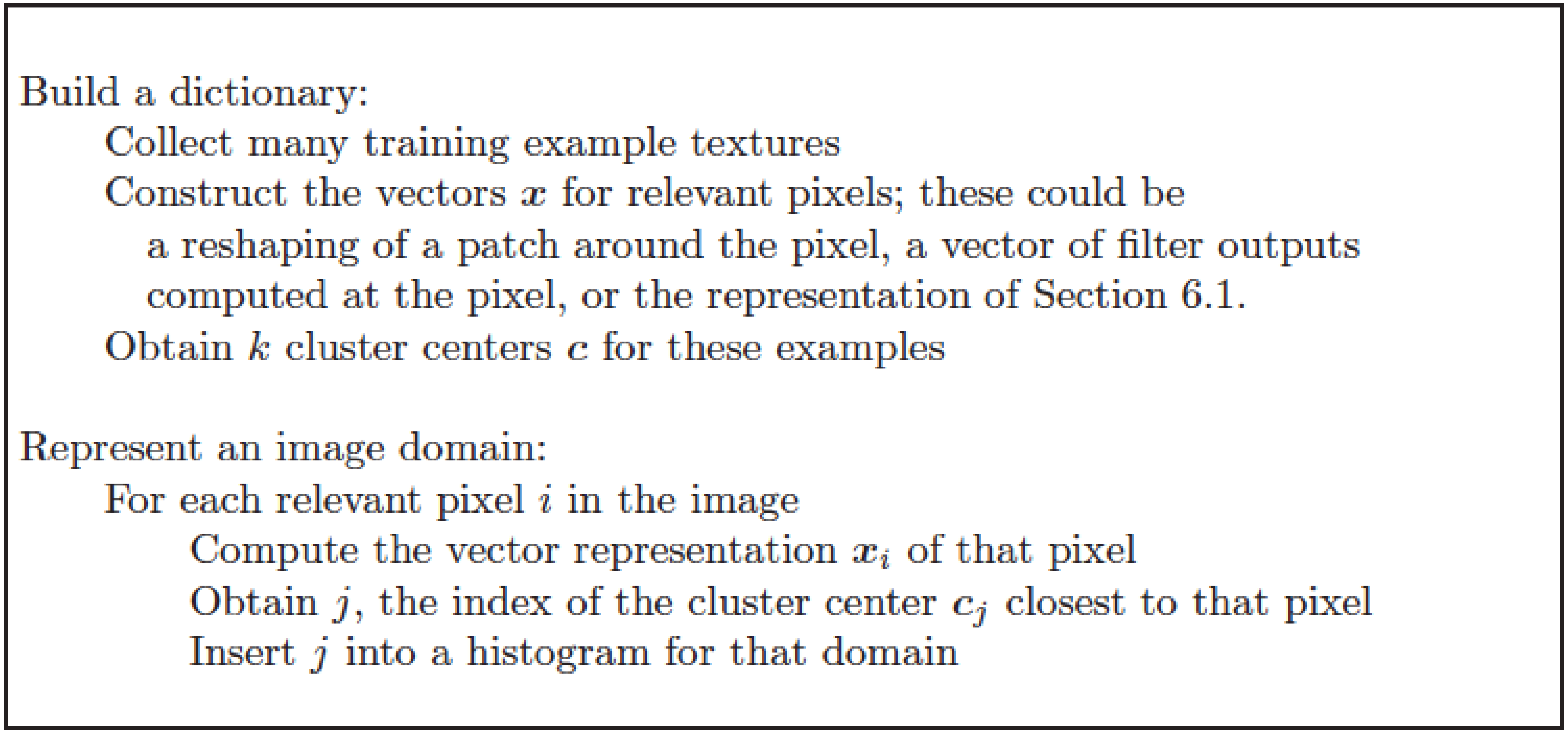
Summarize the pattern of patches

- Cut region into patches
- Vector quantize - vector quantized image patches often called visual words
- Build histogram of resulting numbers



Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"

Representing a Region

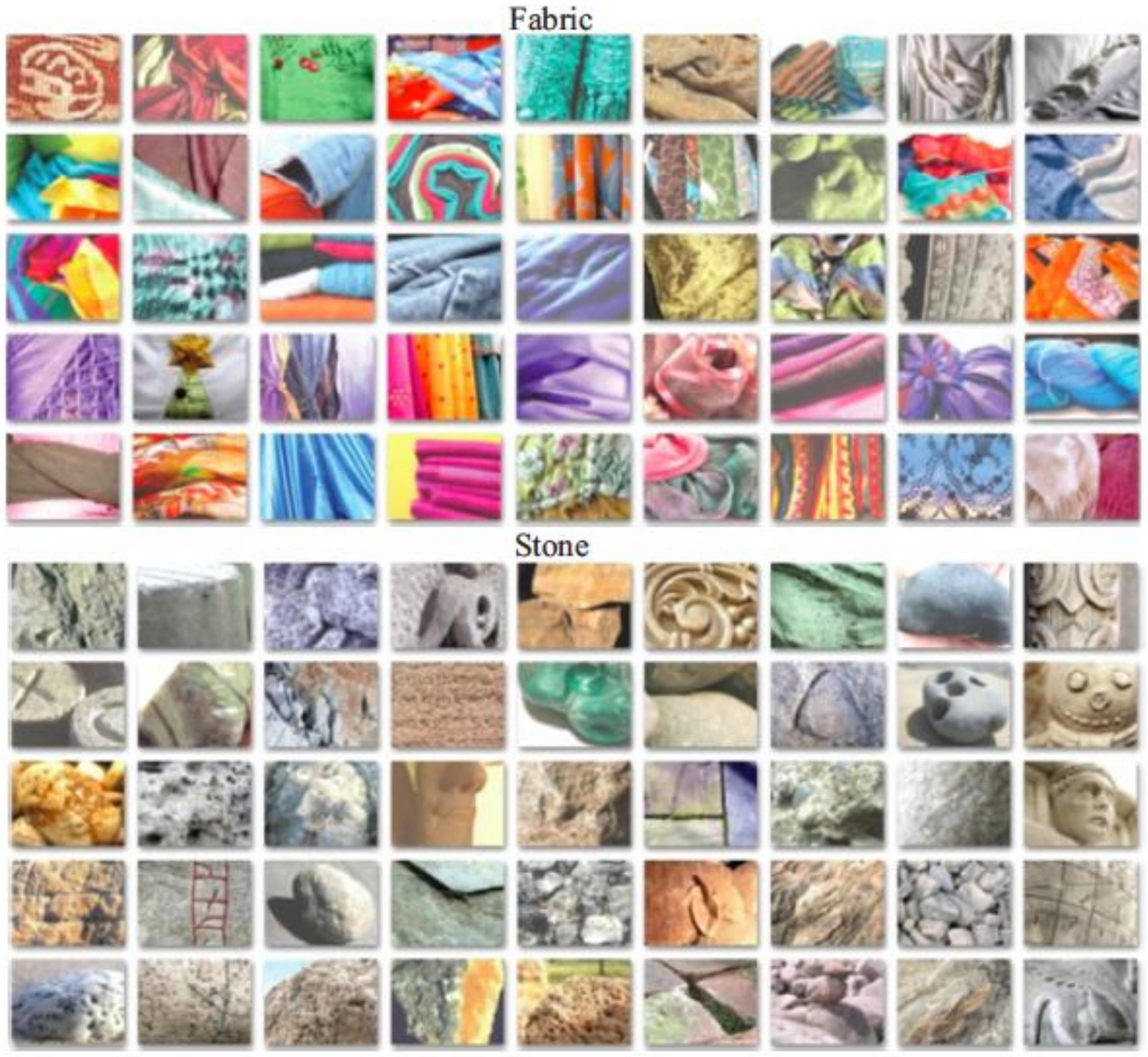


```
Build a dictionary:
    Collect many training example textures
    Construct the vectors $x$ for relevant pixels; these could be
      a reshaping of a patch around the pixel, a vector of filter outputs
      computed at the pixel, or the representation of Section 6.1.
    Obtain $k$ cluster centers $c$ for these examples

Represent an image domain:
    For each relevant pixel $i$ in the image
        Compute the vector representation $x_i$ of that pixel
        Obtain $j$, the index of the cluster center $c_j$ closest to that pixel
        Insert $j$ into a histogram for that domain
```

Algorithm 6.2: Texture Representation Using Vector Quantization.

Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"



Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"

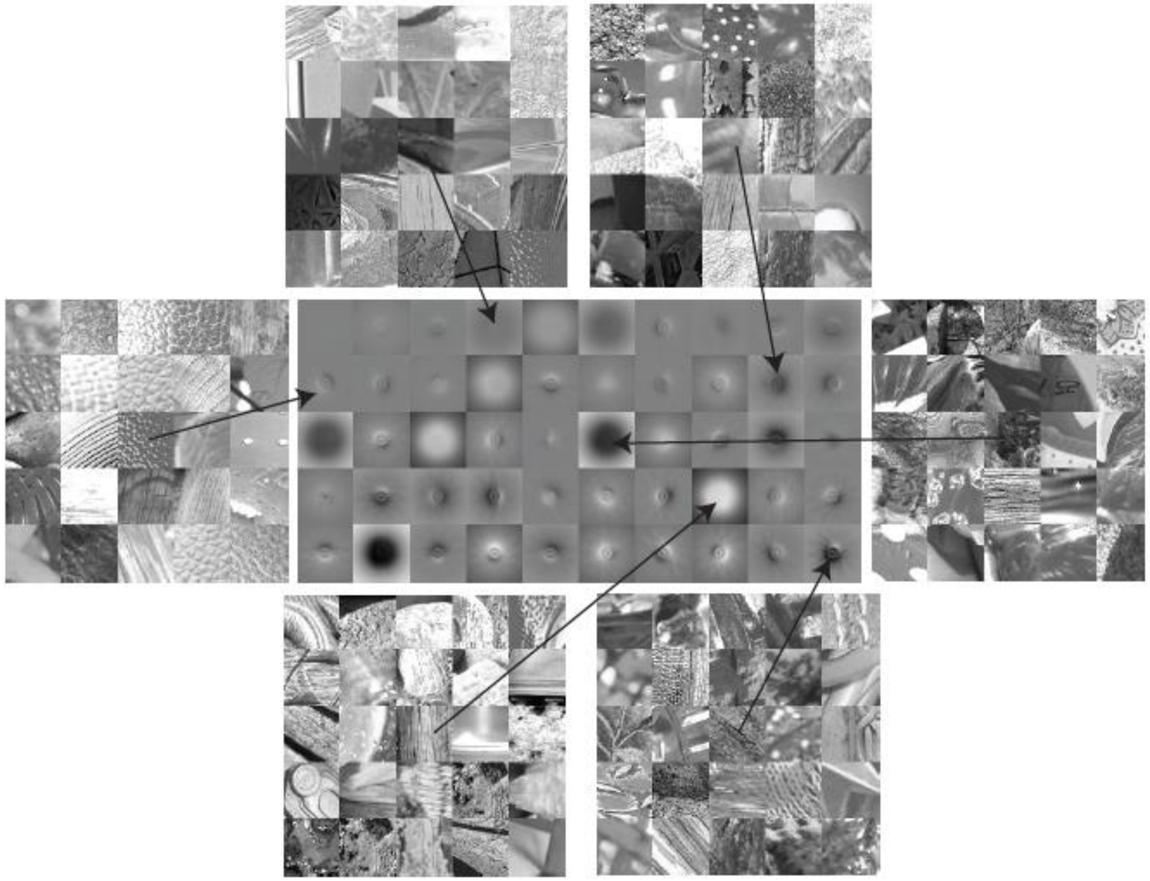


FIGURE 6.9: Pattern elements can be identified by vector quantizing vectors of filter outputs, using k-means. Here we show the top 50 pattern elements (or textons), obtained from all 1,000 images of the collection of material images described in Figure 6.2. These were filtered with the complete set of oriented filters from Figure 6.4. Each subimage here illustrates a cluster center. For each cluster center, we show the linear combination of filter kernels that would result in the set of filter responses represented by the cluster center. For some cluster centers, we show the 25 image patches in the training set whose filter representation is closest to the cluster center. *This figure shows elements of a database collected by C. Liu, L. Sharan, E. Adelson, and R. Rosenholtz, and published at* ***http://people.csail.mit.edu/lavanya/research_sharan.html****. Figure by kind permission of the collectors.*

Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"

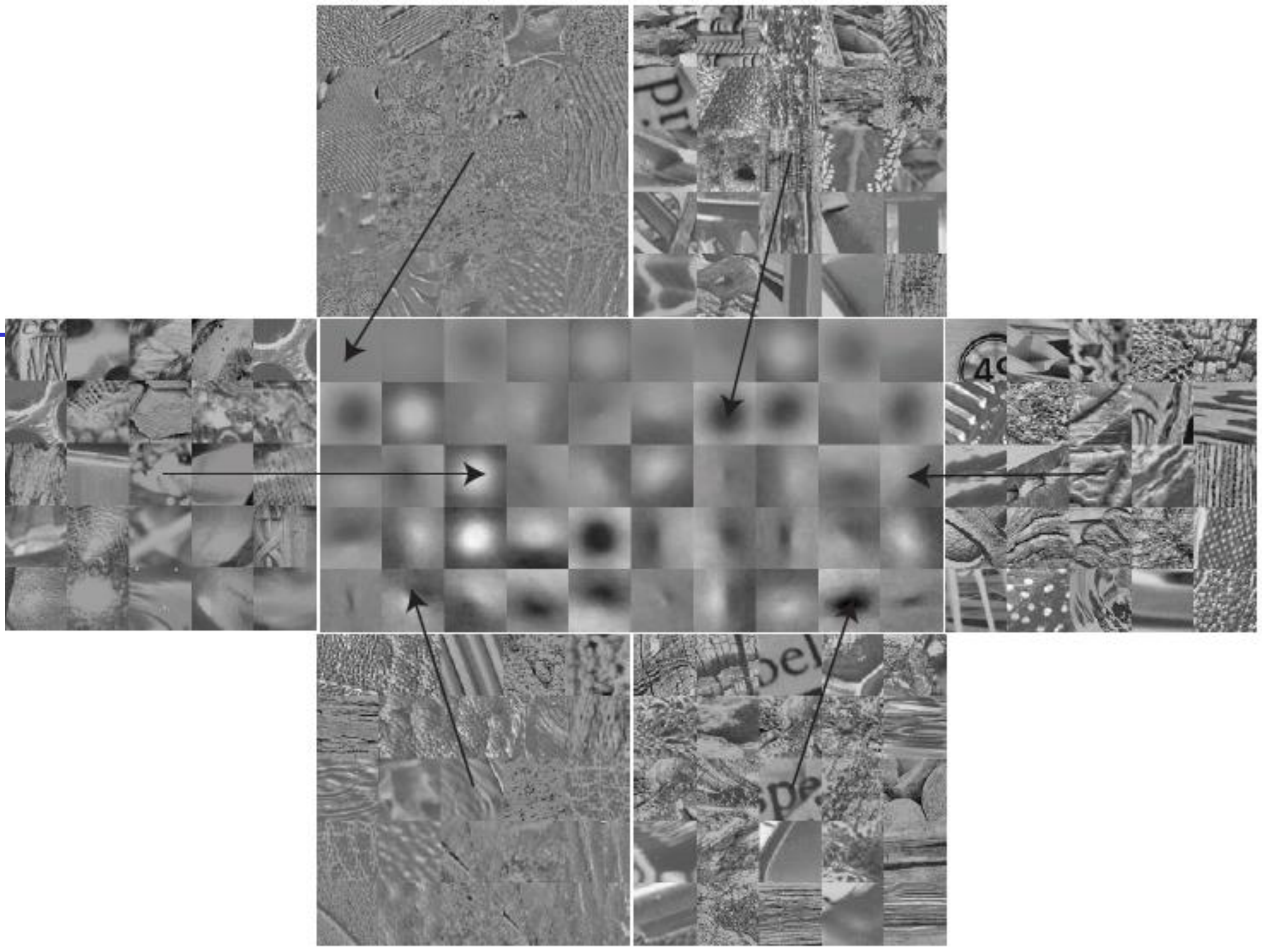


FIGURE 6.10: Pattern elements can also be identified by vector quantizing vectors obtained by reshaping an image window centered on each pixel. Here we show the top 50 pattern elements (or textons), obtained using this strategy from all 1,000 images of the collection of material images described in Figure 6.2. Each subimage here illustrates a cluster center. For some cluster centers, we show the closest 25 image patches. To measure distance, we first subtracted the average image intensity, and we weighted by a Gaussian to ensure that pixels close to the center of the patch were weighted higher than those far from the center. *This figure shows elements of a database collected by C. Liu, L. Sharan, E. Adelson, and R. Rosenholtz, and published at* `http://people.csail.mit.edu/lavanya/research_sharan.html`*. Figure by kind permission of the collectors.*

Texture Representations

A vector summarizing the trends in pattern elements

- either overall trend in filter responses
- or histogram of vector quantized patches

At a pixel

- compute representations for a patch centered on the pixel

For a region

- compute representations for the whole region

Texture Synthesis

Problem:

- Take a small example image of pure texture
- Use this to produce a large domain of “similar” texture

Why:

- Computer graphics demands lots of realistic texture
- Fill in holes in images created by removing objects

Texture Synthesis

Simple case -- Fill holes

- Synthesize a single pixel in a large image
- Approach:
  - Match the window around that pixel to other windows in the image
  - Choose a value from the matching windows
    - most likely, uniformly and at random

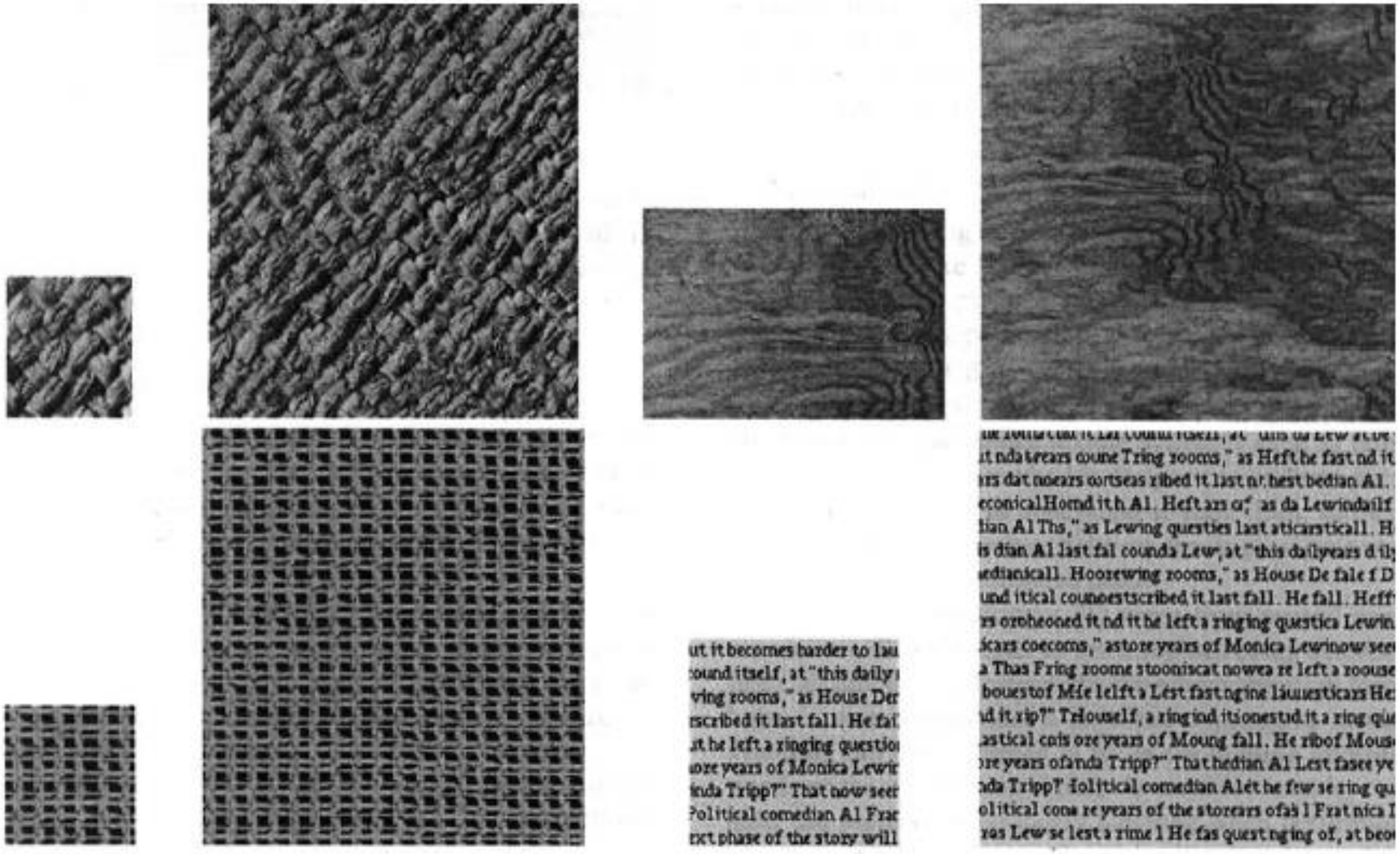
Expand to large images

- Start: take a piece of the example image
- Fill in pixels on the boundary
  - But these are missing more than the center
    - Discount missing pixels when matching
- Each time you fill in a pixel, you can use that to match

```
Choose a small square of pixels at random from the example image
Insert this square of values into the image to be synthesized
Until each location in the image to be synthesized has a value
    For each unsynthesized location on
        the boundary of the block of synthesized values
        Match the neighborhood of this location to the
            example image, ignoring unsynthesized
            locations in computing the matching score
        Choose a value for this location uniformly and at random
            from the set of values of the corresponding locations in the
            matching neighborhoods
    end
end
```

Algorithm 6.4: Non-parametric Texture Synthesis.

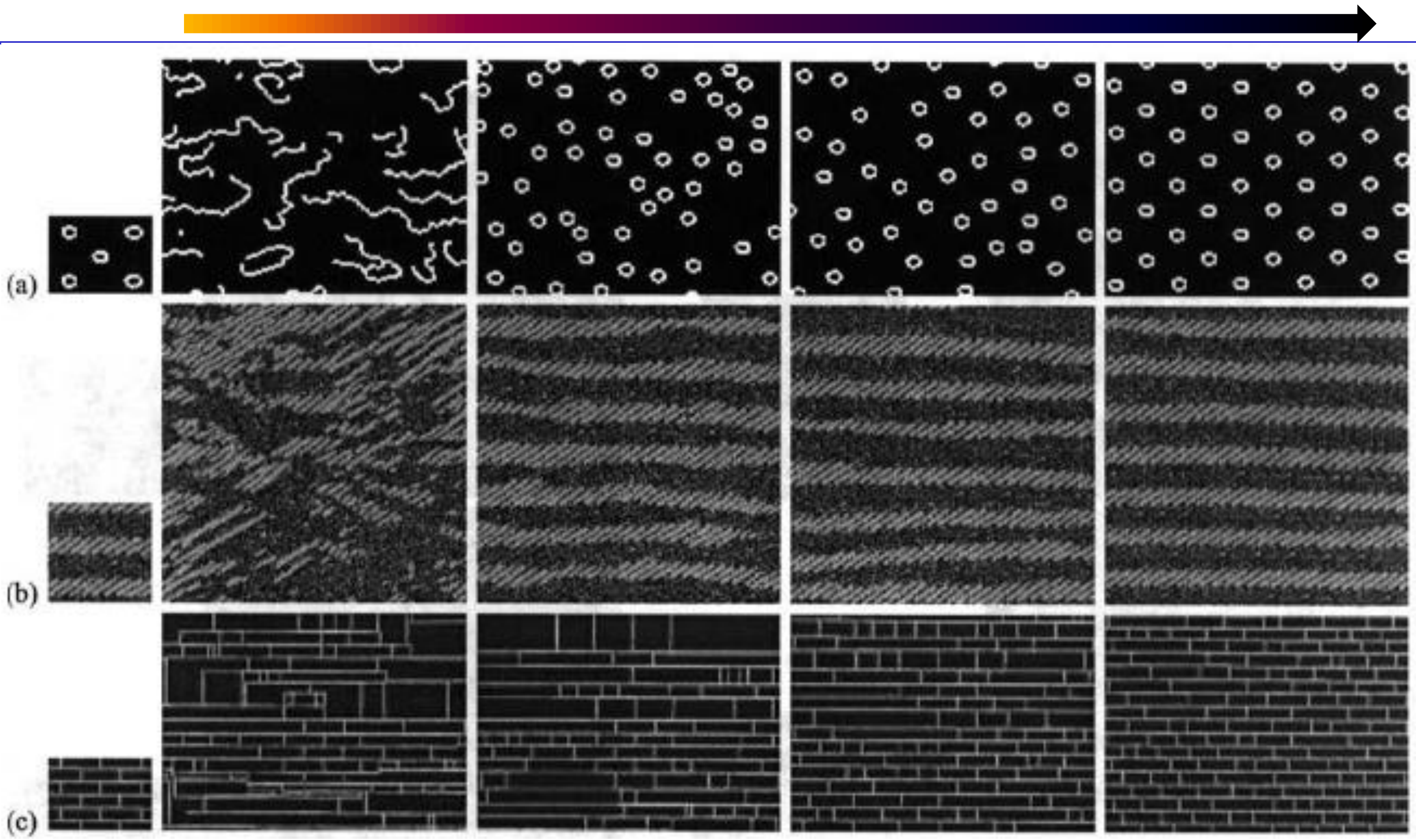
Forsyth and Ponce, "Computer Vision – A Modern Approach 2e"



Small blocks are examples, large are synthesized. Notice how (for example) synthesized text looks like actual text.

Efros and Leung 1999

Neighborhood size



Efros and Leung 1999

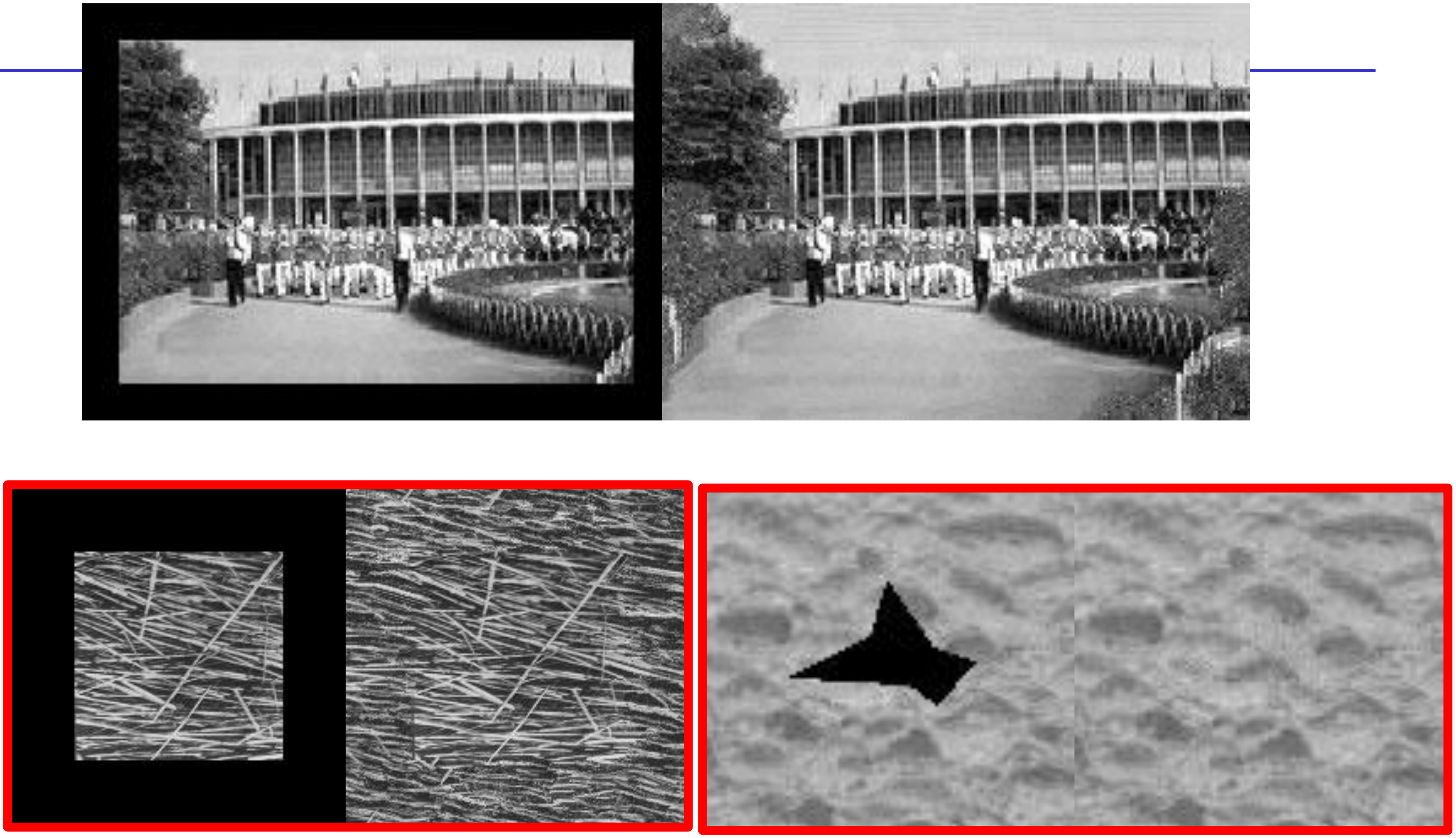
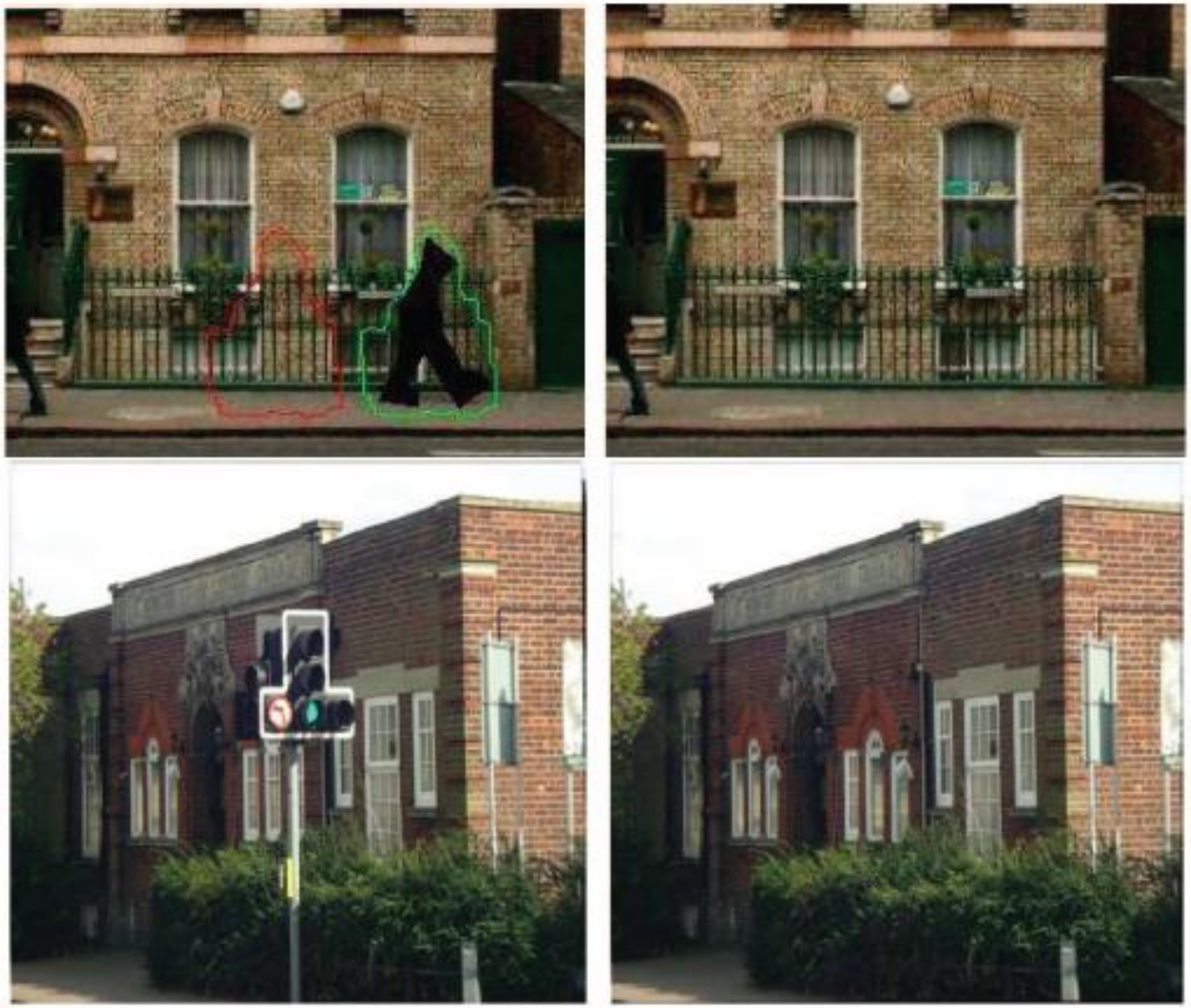


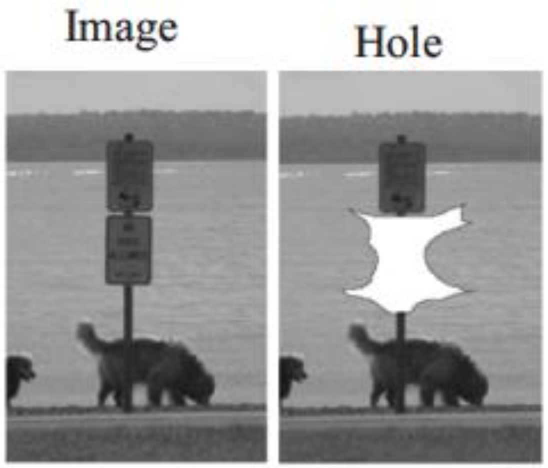
Figure from Texture Synthesis by Non-parametric Sampling, A. Efros and T.K. Leung, Proc. Int. Conf. Computer Vision, 1999 copyright 1999, IEEE

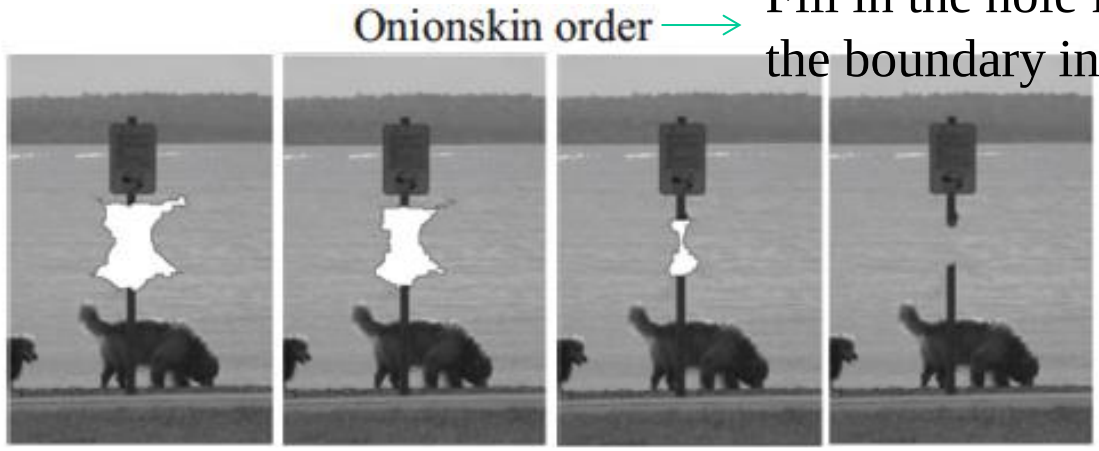


Fill in holes by looking for example patches in the image. If needed, rectify faces (lower images).

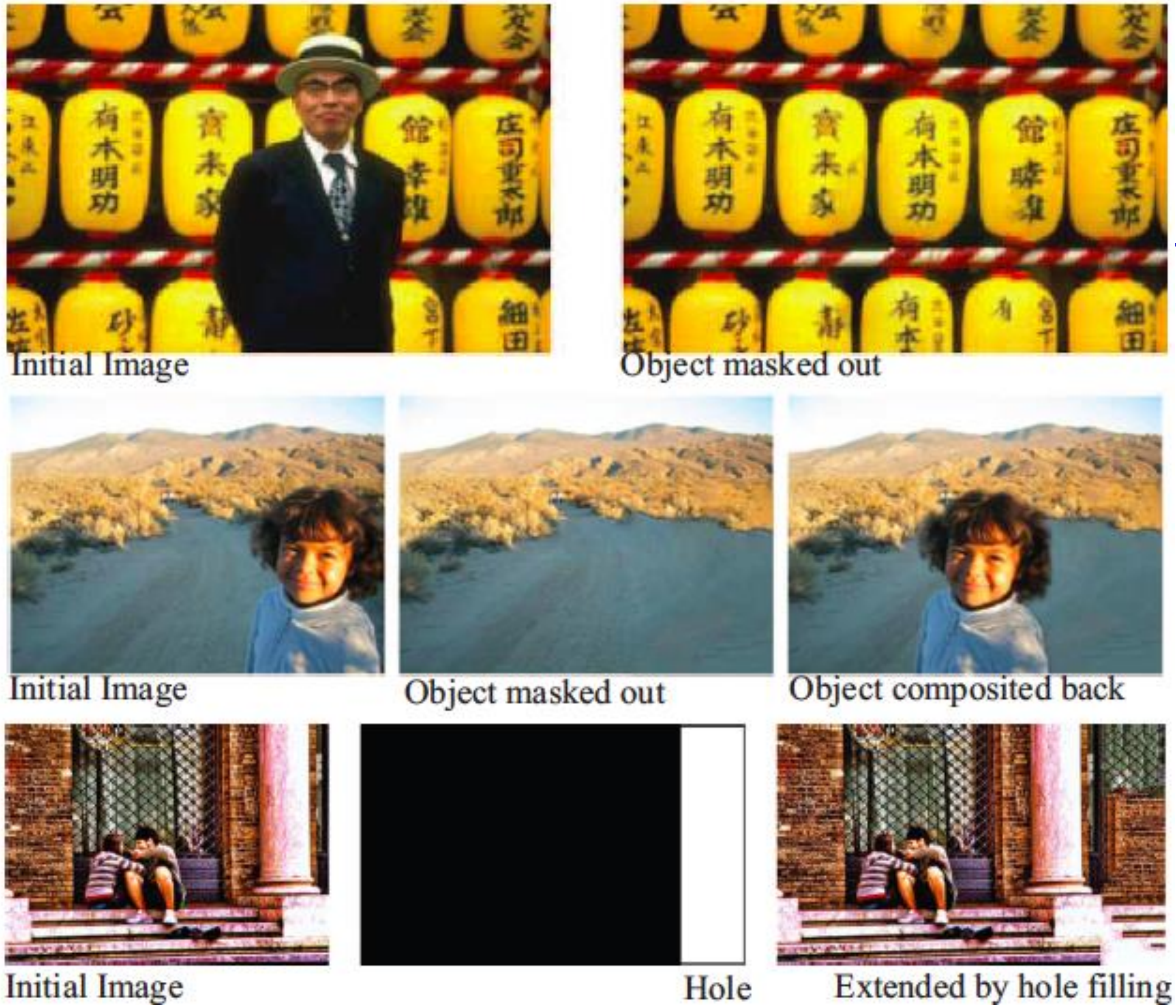
Wilczkowiak et al., BMVC 2005

Fill in the hole from the boundary inward





Criminisi et al., IEEE TIP 2004



State-of-the-art in image fill-in combines texture synthesis, coherence, and smoothing by Bugeau et al., IEEE TIP.

Shape from Texture

Texture is a powerful shape cue

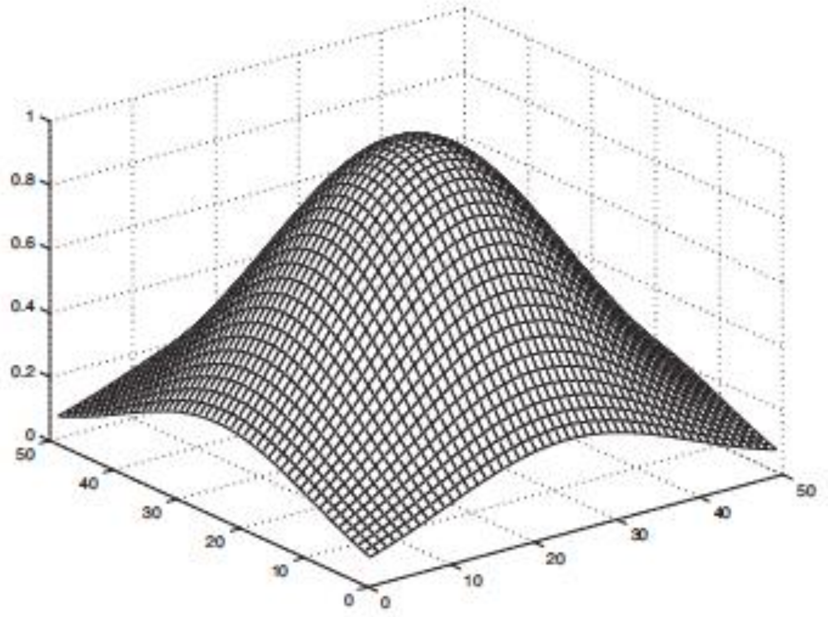
- most likely because small pattern elements deform in predictable ways

Recovering shape from texture

- Identify repeating pattern elements
- Determine frontal view
- From this, determine normal
- Integrate normals to get surface

Shape from texture offers information about lighting

- If pattern elements are repetitions, then
  - brighter (resp. darker) ones receive more (resp. less) light



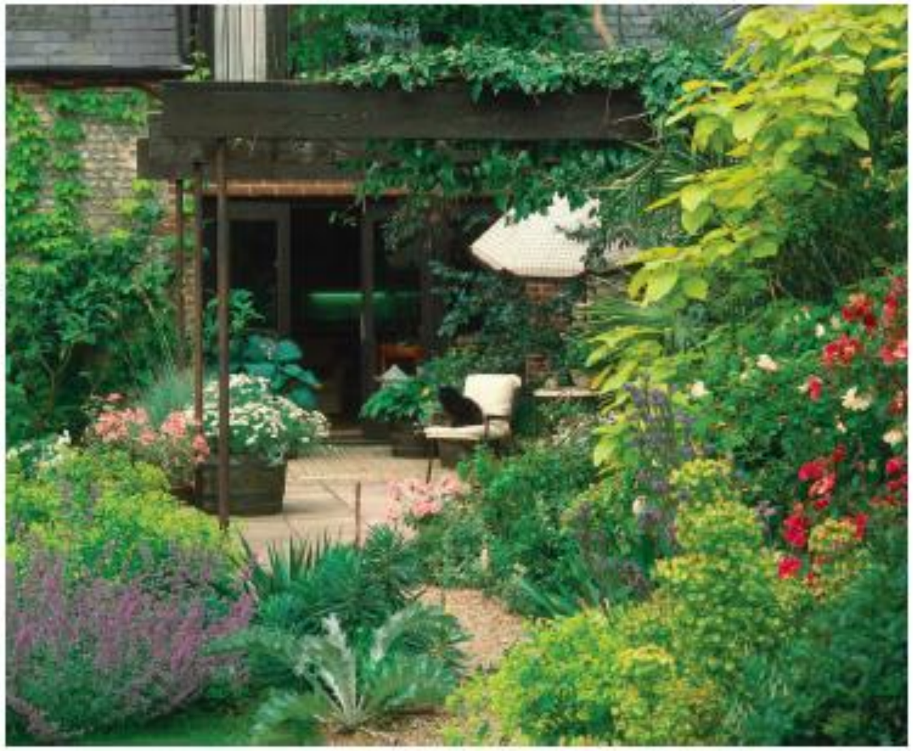


FIGURE 6.19: Humans obtain information about the shape of surfaces in space from the appearance of the texture on the surface. The figure on the **left** shows one common use for this effect; away from the contour regions, our only source of information about the surface depicted is the distortion of the texture on the surface. On the **right**, the texture gives a clear sense of the orientation of the ground plane, how the plants stand out from the path, and how far away the building at the back is. *Geoff Brightling © Dorling Kindersley, used with permission.*

Forsyth and Ponce, “Computer Vision – A Modern Approach 2e”

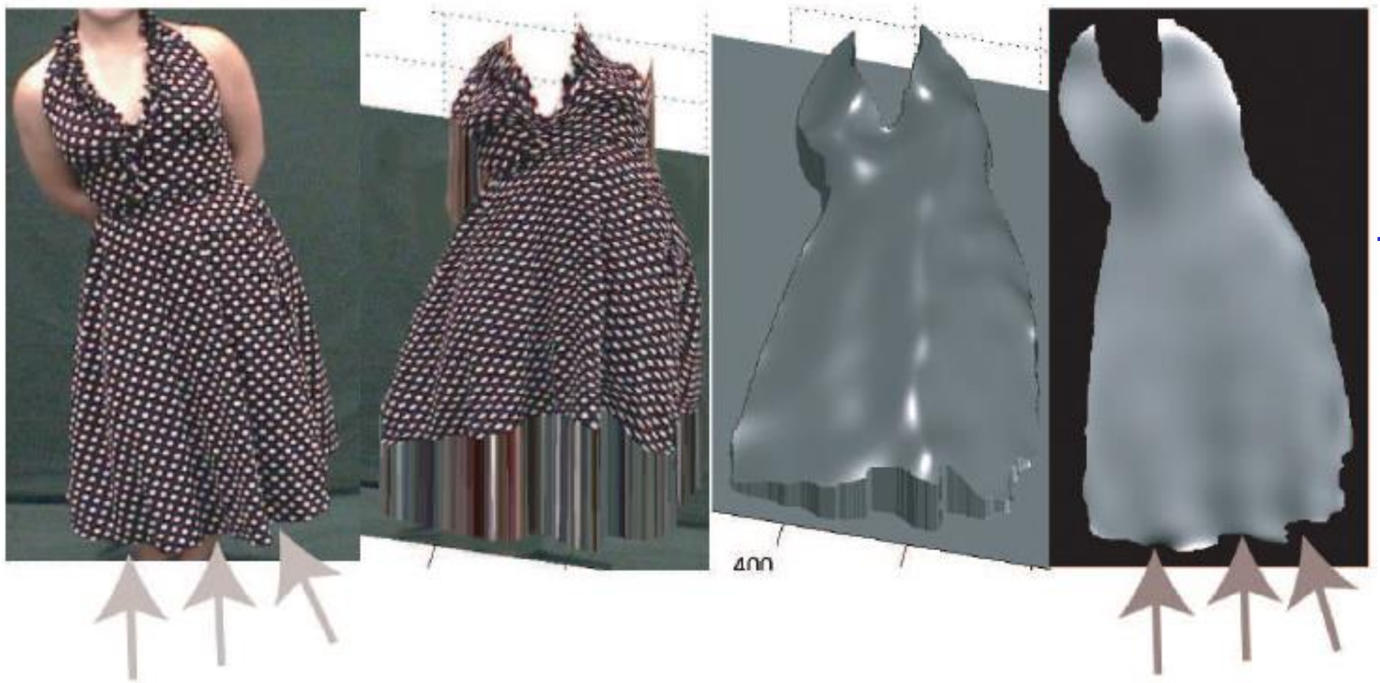


FIGURE 6.21: On the **left**, a textured surface, whose texture is a set of repeated elements, in this case, spots. **Center left**, a reconstruction of the surface, made using texture information alone. This reconstruction has been textured, which hides some of its imperfections. **Center right**, the same reconstruction, now rendered as a slightly glossy gray surface. Because texture elements are repeated, we can assume that if different elements have a significantly different brightness, this is because they experience different illumination. **Right** shows an estimate of the illumination on the surface obtained from this observation. Notice how folds in the dress (arrows) tend to be darker; this is because, for a surface element at the base of a fold, nearby cloth blocks a high percentage of the incident light. *This figure was originally published as Figure 4 of "Recovering Shape and Irradiance Maps from Rich Dense Texton Fields," by A. Lobay and D. Forsyth Proc. IEEE CVPR 2004 © IEEE, 2004.*

If pattern elements are repetitions, then brighter (resp. darker) ones receive more (resp. less) light, so we get an estimate of lighting.

Lobay and Forsyth ,
CVPR 2004

Reading Assignment

Chapter 6 (Texture) of Forsyth & Ponce