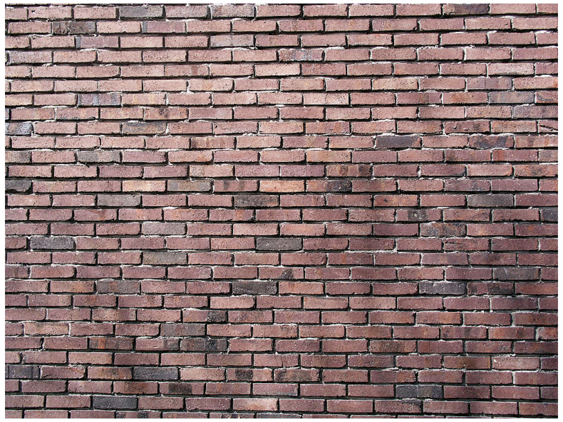
# Texture

**Prof. Dr. Elli Angelopoulou**

**Pattern Recognition Lab (Computer Science 5)**

**University of Erlangen-Nuremberg**

## What is Texture?

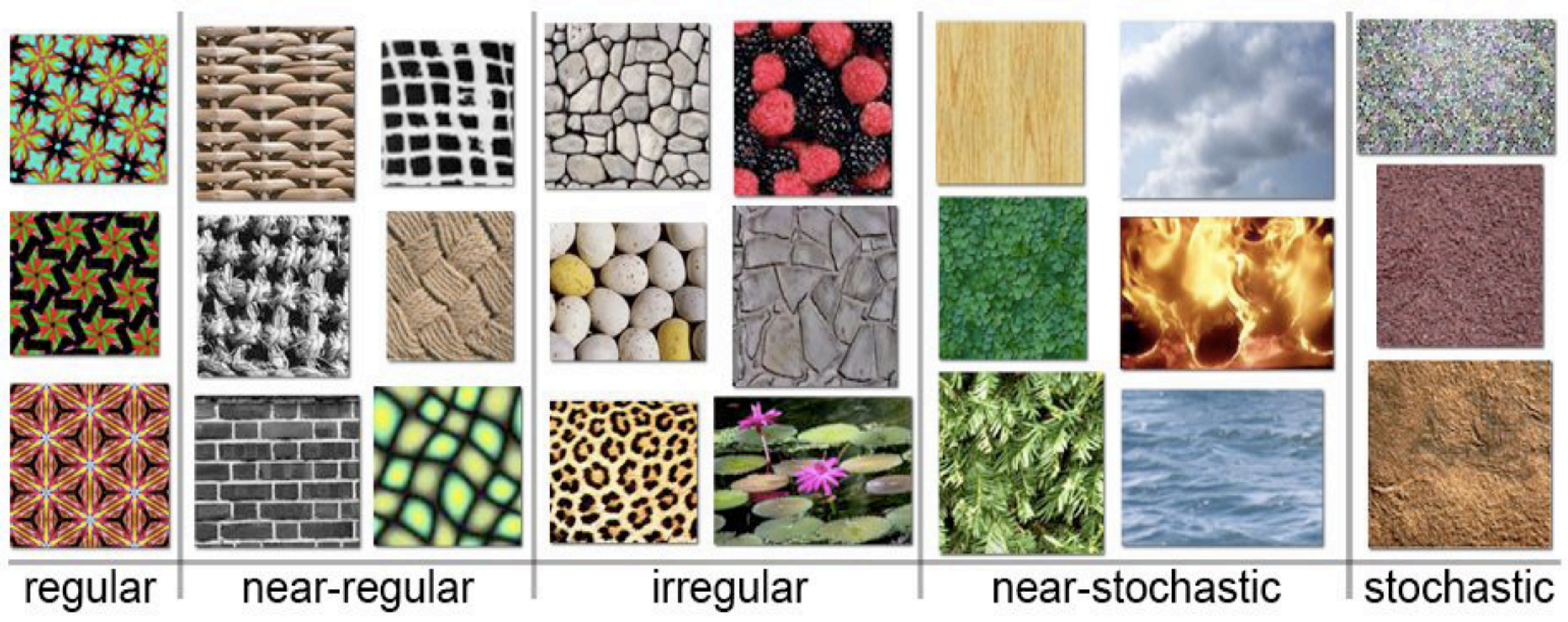


## Texture

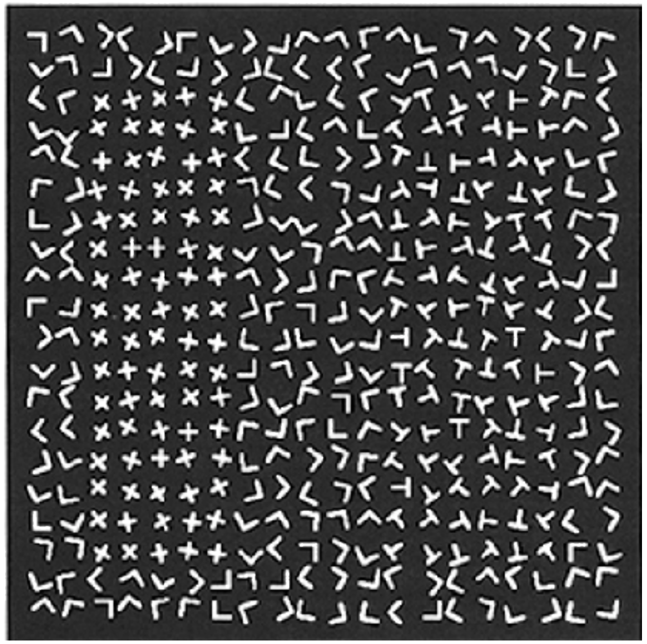
- Texture: a repeatable pattern of small elements
  - Stripes
  - Brick wall
- No precise definition. It is appearance-related (scale-related)
  - Single leaf vs. foliage
  - Single stripe vs. the stripes on a zebra
- Texture can be formed by:
  - The presence of a large number of small objects
    - Pebbles
    - Coffee beans
    - Grass
  - Orderly patterns that look like large numbers of small elements
    - Spots on cats
    - Grains on wooden surfaces
    - Pores on an orange peel

## Different Types (Scales) of Texture

- Texture itself can vary from highly regular to purely stochastic.



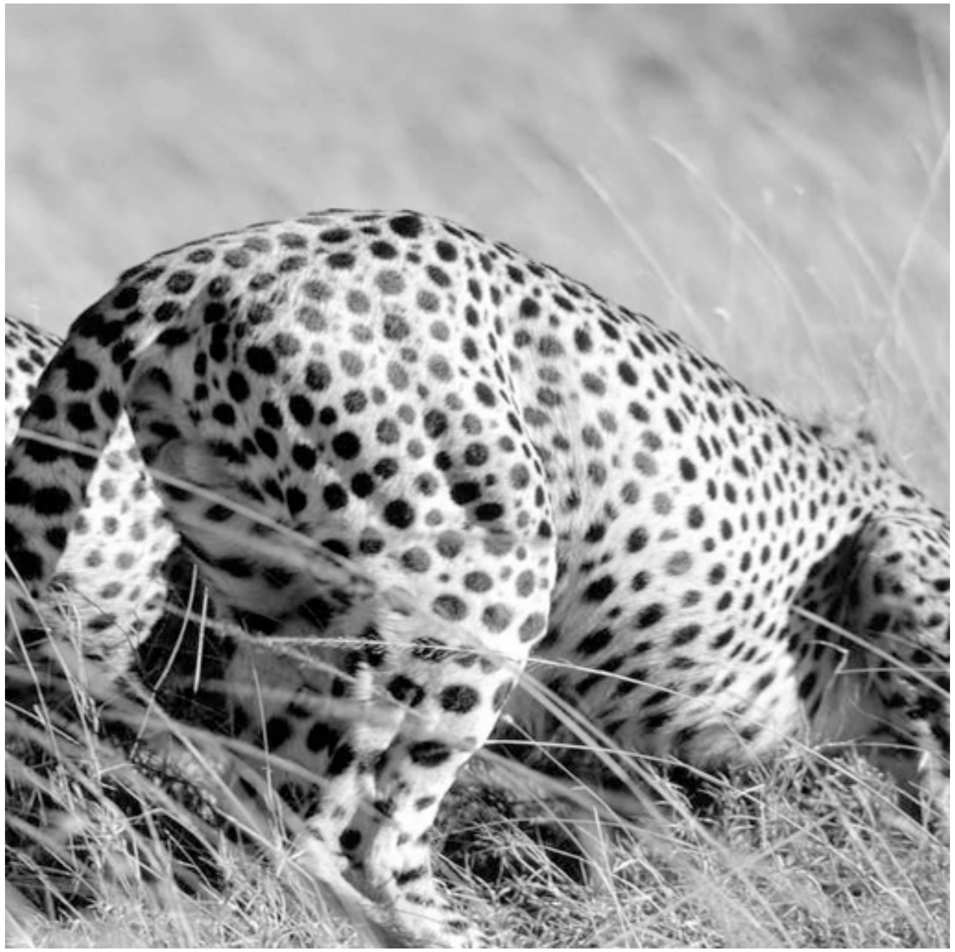
## Texton

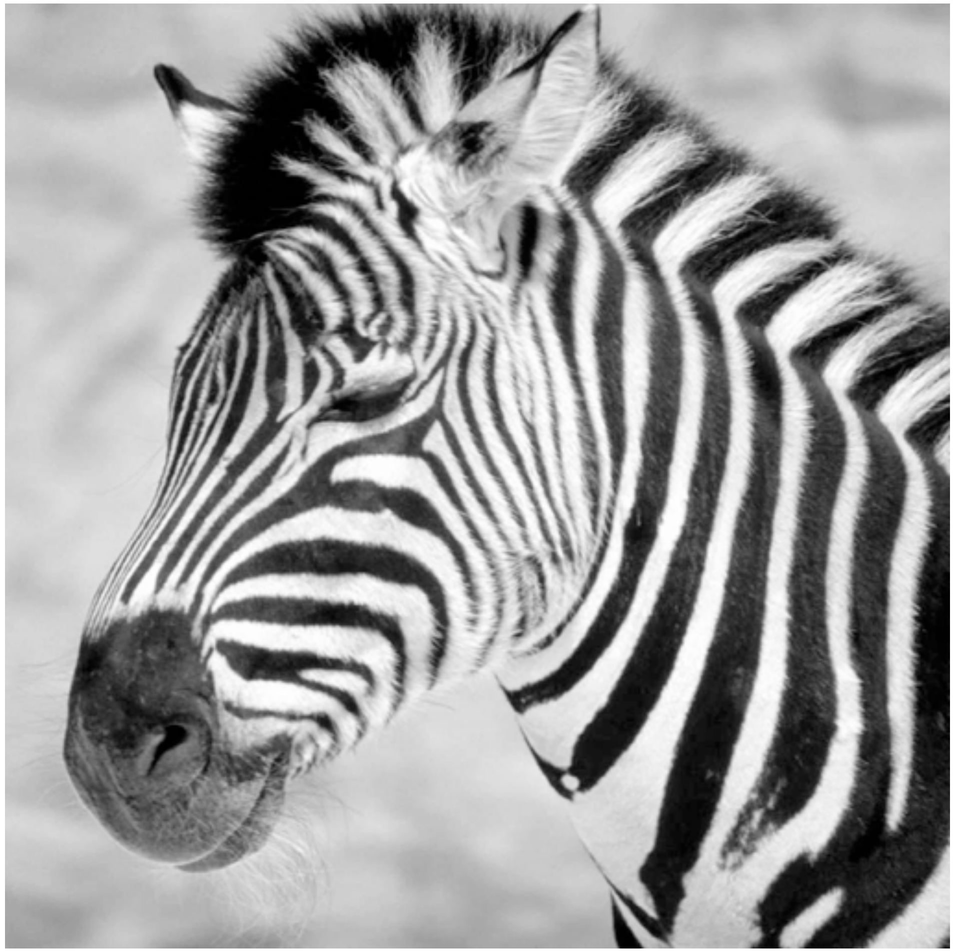


A typical synthetic image used in experiments on pre-attentive texture discrimination. Image courtesy of J. R. Bergen and M. S. Landy. *Computational Modeling of Visual Texture Segregation,* pages 253-271. MIT Press, 1991.

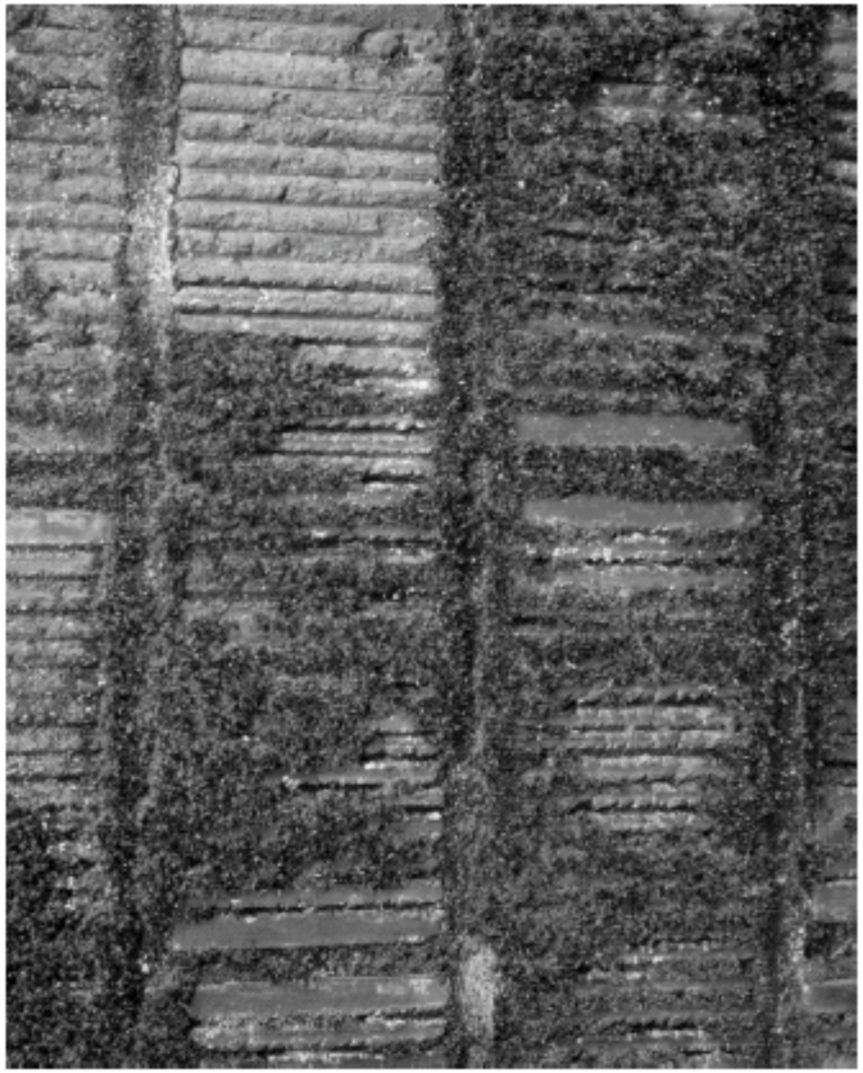
- Loosely defined, **textons** are the small elements that keep repeating themselves in a regular pattern.
- They can be thought of as the atomic units of texture.
- Work in psychology (B. Julesz 1962) showed that for humans texture perception is one of the early steps towards identifying objects and understanding a scene.

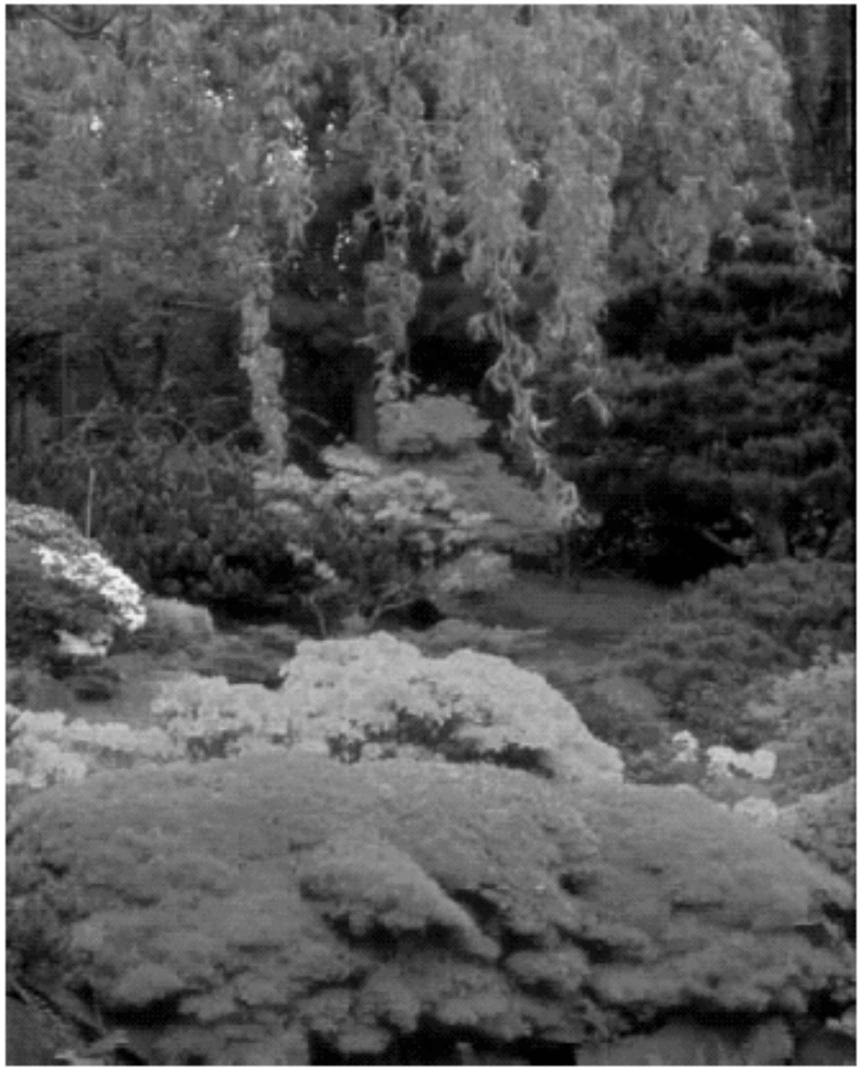
## Examples of Texture





## Examples of Texture





## Texture Topics

- Image Segmentation

  Since texture is an important clue in images, the goal of image segmentation is to identify continuous regions of uniform texture.

- Texture Synthesis

  Often a topic of Computer Graphics.

  It is one of the subjects that overlap Computer Graphics and Computer Vision.

- Shape from Texture

  The key idea behind shape from texture is to exploit texture deformations to infer 3D shape.

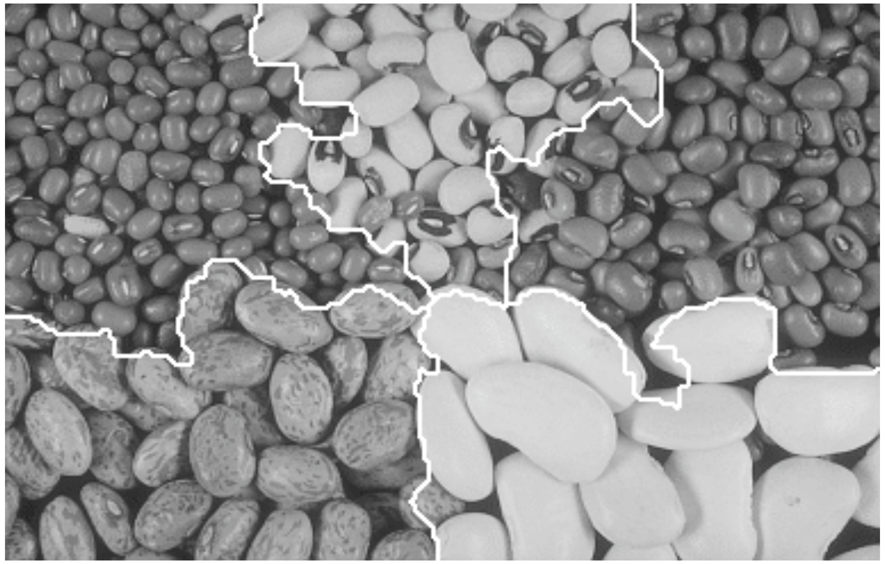
## Image Segmentation

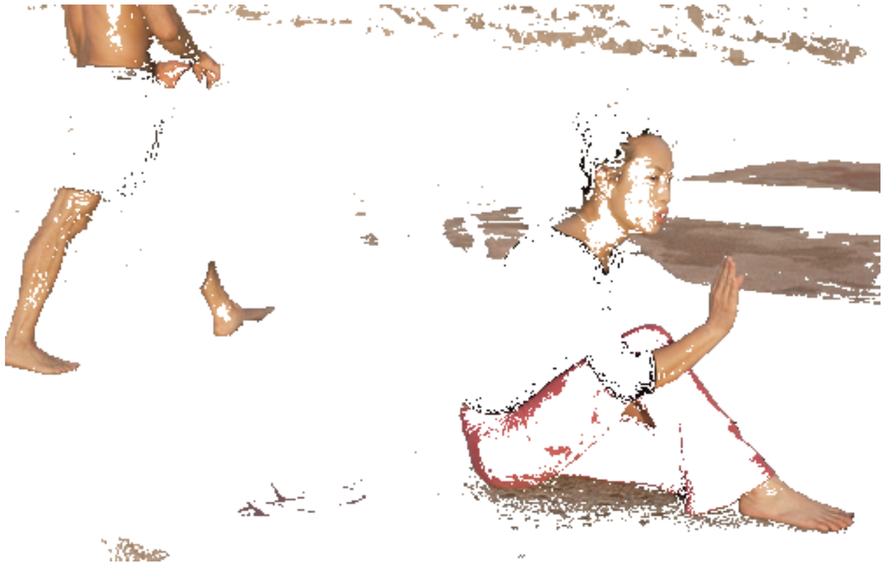
- The goal of image segmentation is, given an image I, to break it into regions so that:
  - Each region has a unique characteristic.
  - Regions are non-overlapping.

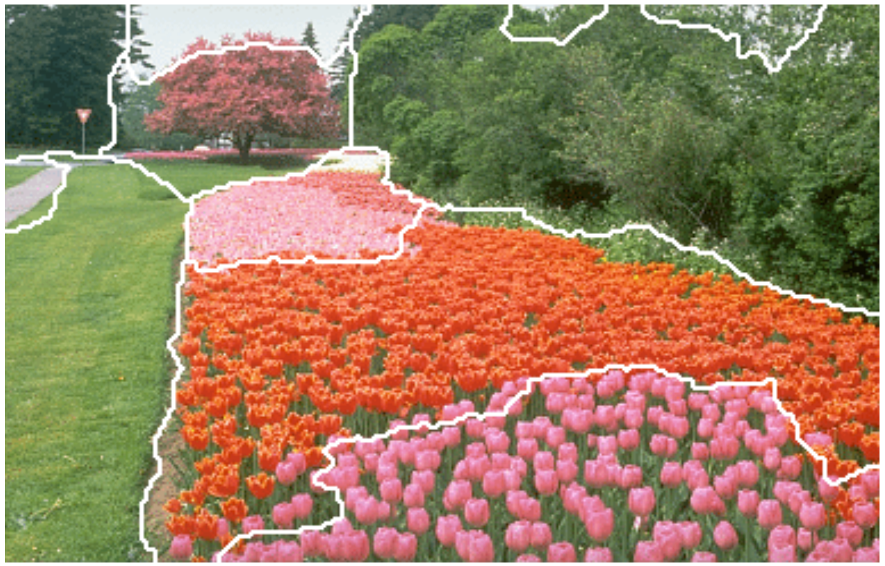
$$I = \bigcup_{\forall i} R_i \quad \text{and} \quad R_i \cap R_j = \varnothing, \quad \text{for} \quad i \neq j$$

- Each region can then be separately processed.
  - Process only a region of interest (ROI)
  - Apply different algorithms to different regions, based on the peculiarities of the region.
- The unique characteristic used in segmentation can be:
  - Color
  - Lightness (bright or dim pixels)
  - Texture
  - …

## Image Segmentation Examples

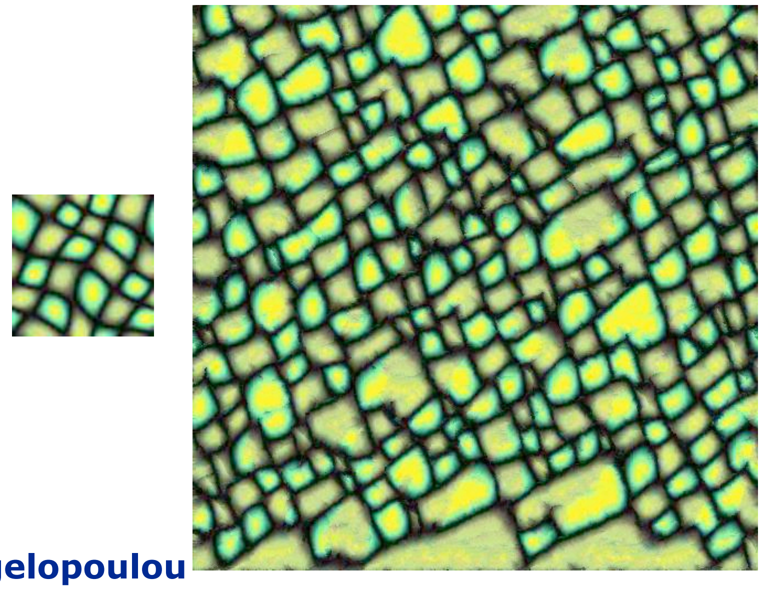
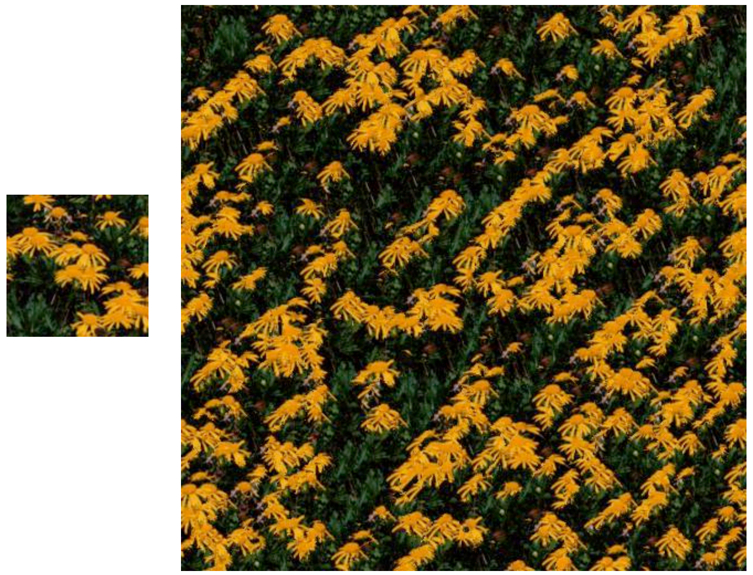






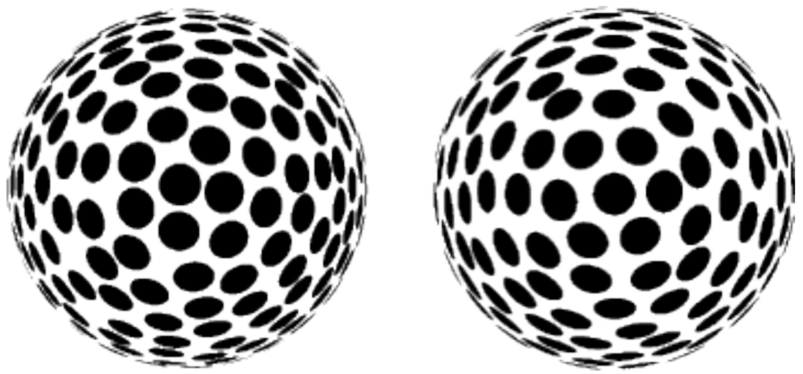
## Texture Synthesis

- The goal of texture synthesis is, given an image of a texture $I_T$, to construct large regions of that texture.
- E.g.: Use a patch of grass to create a picture of a lawn.
- Simple repetition will not work. There should be some variation in the pattern.
- Idea: Use statistics in generating such a variation.

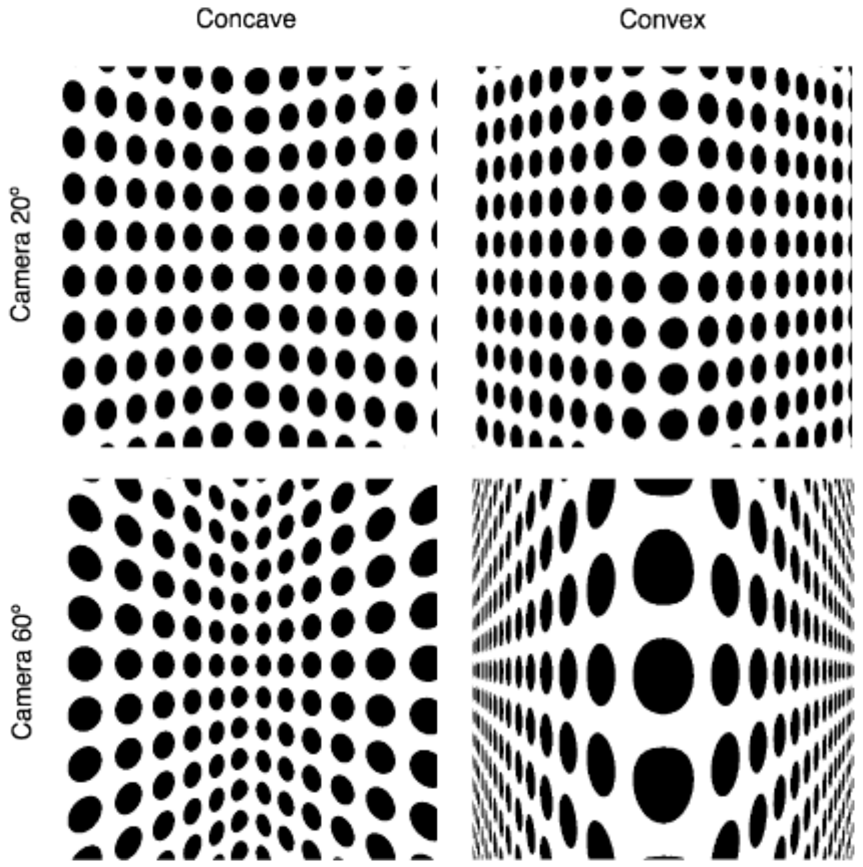



## Shape from Texture

- When the texture pattern is known, we can use its distortion to infer shape.
- We can only compute the surface normals.



The sphere on the left is projected on the image plane using perspective projection. The one on the right using orthographic projection.



Texture images courtesy of J.T. Todd, L. Thaler, T.M.H. Dijkstra, J.J. Koenderink, and A.M.L. Kappers.

## Texture Representation

- From the image perspective, a texture is an organized pattern of subelements (textons).
- Problem: There is no canonical set of textons. Imagine trying to read without a precise definition of an alphabet.
- So how do can we find such subelements?
- Think of popular subelements and design filters which will produce a high response if a subelement is present.

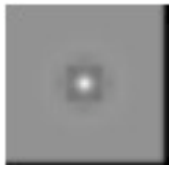
## Basic Texton Shapes

- Human vision suggests that spots and bars at different scales and orientation are valid filters for finding subelements.
- Spots
  - They are typically symmetric Gaussians
  - They can identify small regions that differ from their neighbours
- Bars
  - They are typically oriented Gaussians
  - They can identify oriented structure.
- Use Spots and Bars filters at multiple scales

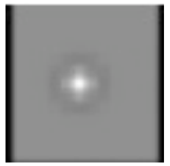
## Spot Filter

- Goal: Design a filter that when it is superimposed with a spot it will give a high response.
- Symmetric Gaussians are circular. The variance σ controls the width and intensity of the spot.
- A common spot filter is a weighted sum of symmetric Gaussians:

$$S = w_1 G(\sigma_1) + w_2 G(\sigma_2) + w_3 G(\sigma_3) \quad \text{where} \quad G(\sigma) = e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

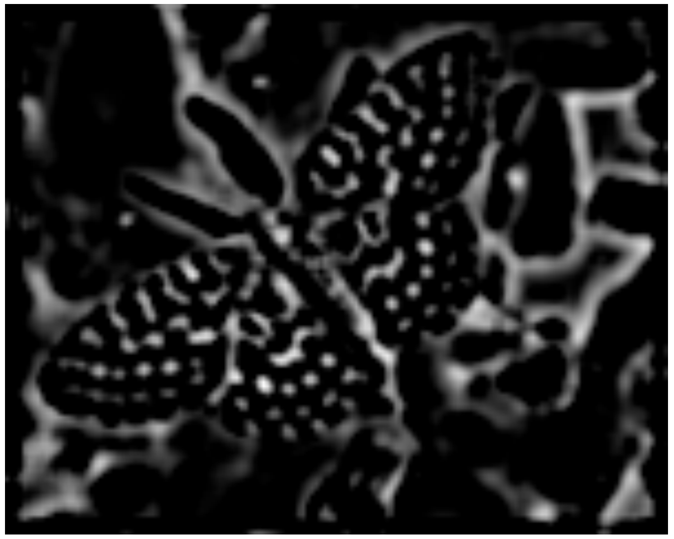


$$w_1 = w_3 = 1 \quad w_2 = -2$$
$$\sigma_1 = 0.62 \quad \sigma_2 = 1 \quad \sigma_3 = 1.6$$



$$w_1 = 1 \quad w_2 = -1$$
$$\sigma_1 = 0.71 \quad \sigma_2 = 1.14$$

## Sample Response to a Spot Filter



Positive responses

## Bar Filter

- When we stretch the Gaussian in one direction it becomes a ridge, a bar.
- A typical bar filter is a weighted sum of 3 stretched Gaussians with offset centers.
- A typical bar filter:

$$S = w_1 G(x, y_1, \sigma_{x_1}, \sigma_{y_1}) + w_2 G(x, y_2, \sigma_{x_2}, \sigma_{y_2}) + w_3 G(x, y_3, \sigma_{x_3}, \sigma_{y_3})$$

where

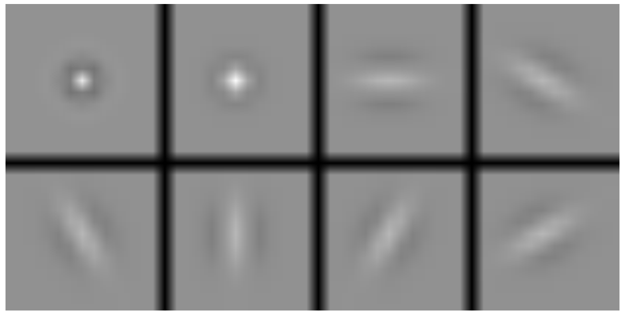
$$G(x, y_c, \sigma_x, \sigma_y) = e^{-\left(\frac{x^2}{2\sigma_x^2} + \frac{(y-y_c)^2}{2\sigma_y^2}\right)}$$

- An example horizontal bar filter uses:

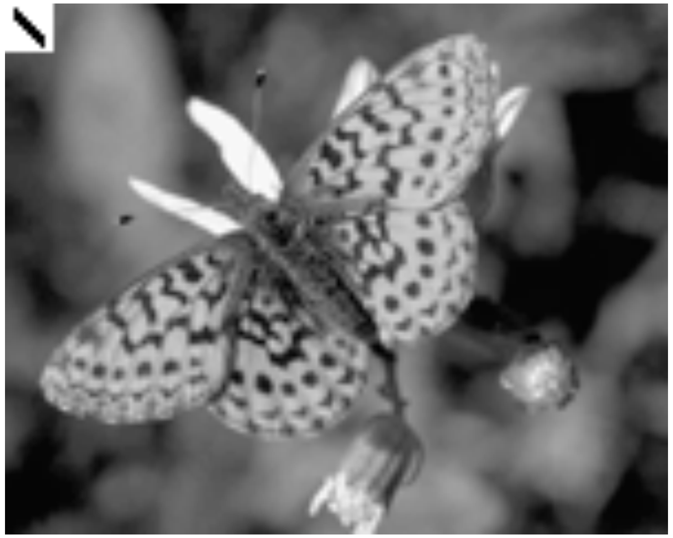
$$w_1 = w_3 = -1 \quad w_2 = 2 \qquad \sigma_{x_1} = \sigma_{x_2} = \sigma_{x_3} = 2$$
$$y_1 = 1 \quad y_2 = 0 \quad y_3 = -1 \qquad \sigma_{y_1} = \sigma_{y_2} = \sigma_{y_3} = 1$$

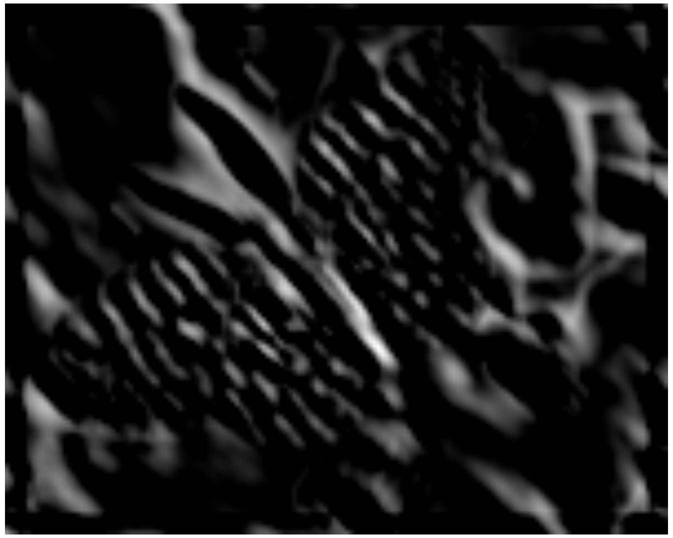
## Dot and Bar Filters (continued)

- We want to detect bars in all orientations, not just horizontal.
- Use a suite of bar filters for different orientations.
- Studies have shown that we need at least 6 orientations.
- We usually apply many dot and bar filters, and thus talk about a filter bank.



## Sample Response to a Bar Filter

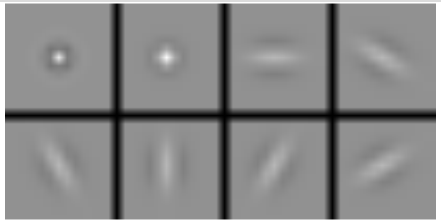




Positive responses

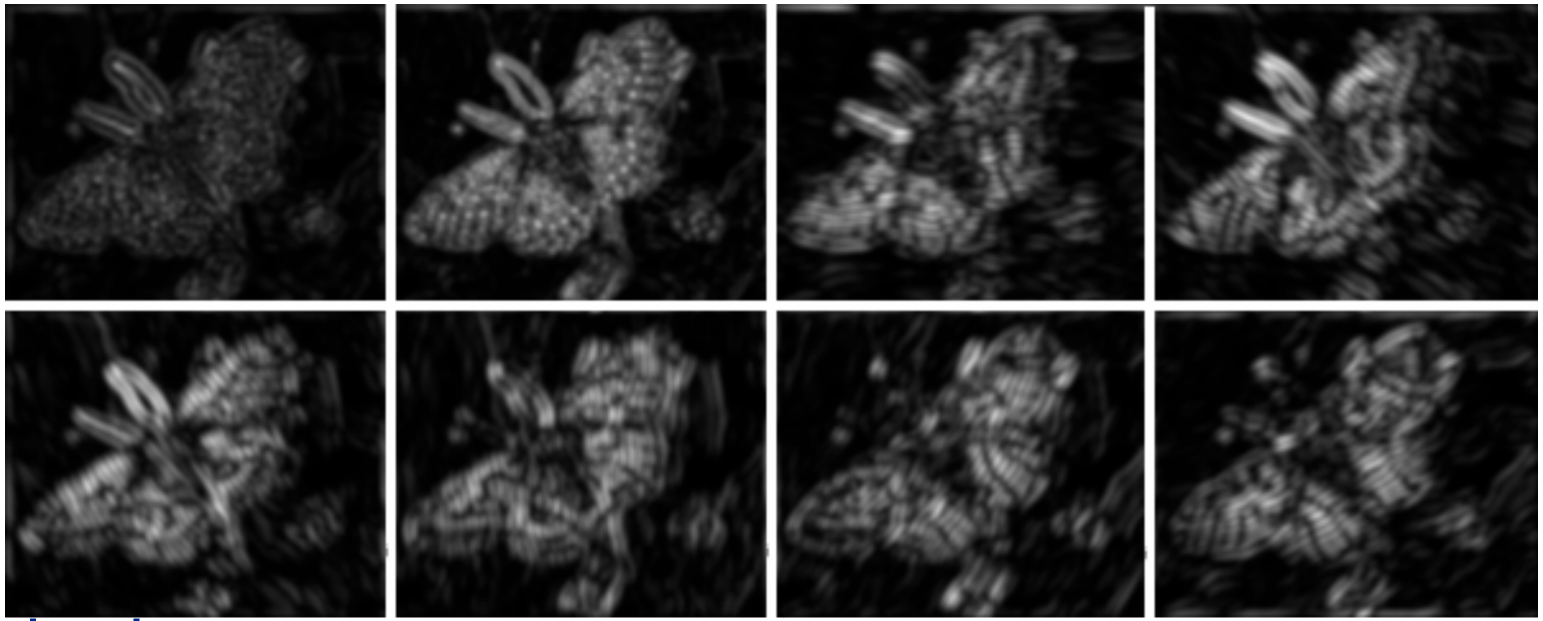
## Example Response of Filter Banks

- Sample texture filters
  - 2 dot filters
  - 6 bar filters



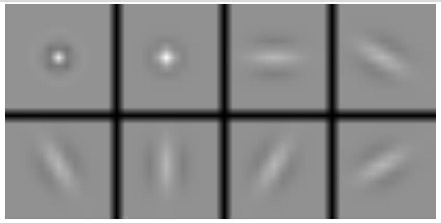
- Original image

- Squared response of each texture filter.



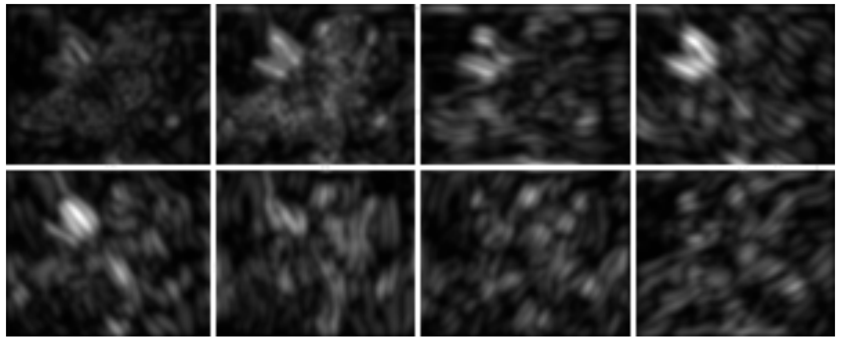
## Filter Banks at Different Scales

- Same sample texture filters
  - 2 dot filters
  - 6 bar filters



- Original image at half size

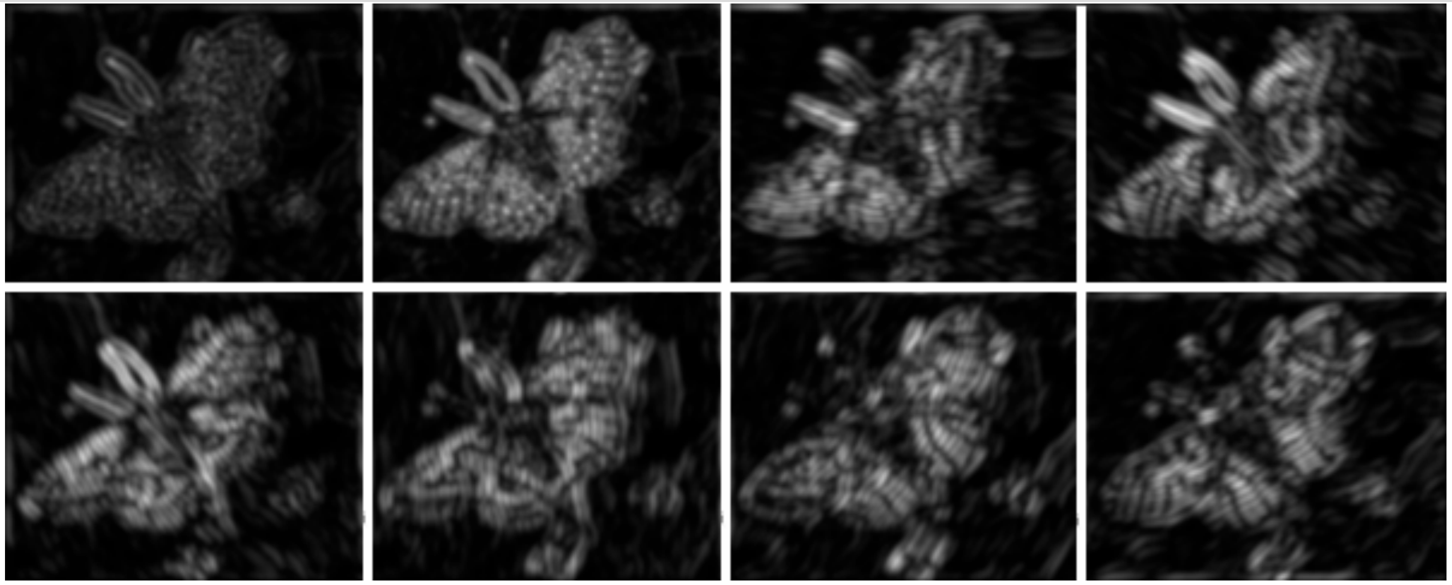
- Squared response of each texture filter.



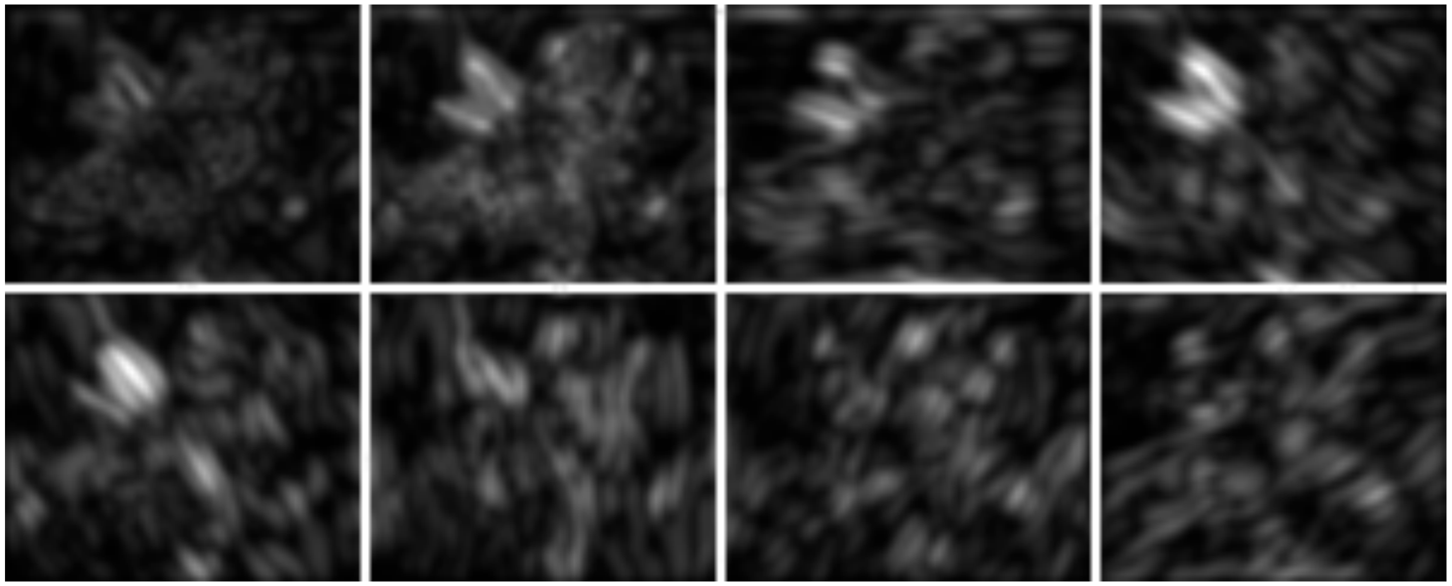
- Filtering was performed at coarser scale, since the filter size remained fixed but the image was half the size of the original.

## Texture Filtering at Different Scales

- Finer Scale



- Enlarged coarser scale



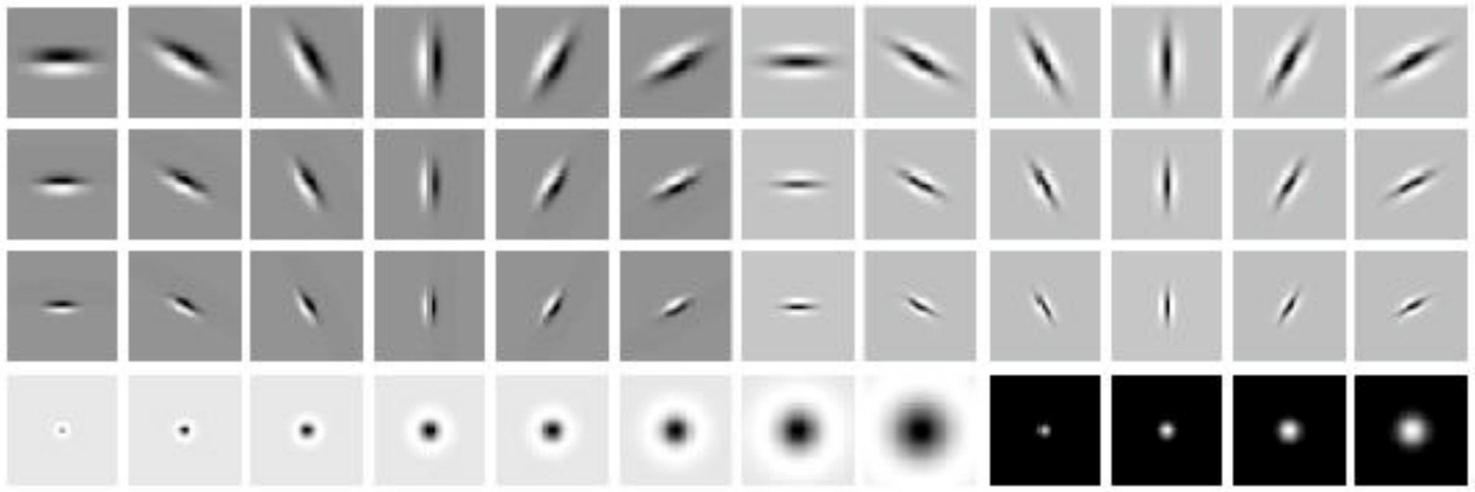
## Which Texture Filters?

- Extensive experimentation has shown that:
- There should be a collection of spot and bar filters covering
  - Different sizes of spots
  - Different sizes of bars
  - Different orientations of bars
- The effectiveness of texture detection algorithms depends on using a variety of filters.
- Studies on trying to find an optimal filter bank showed that such an optimal search does not have a significant impact on the effectiveness of textons.

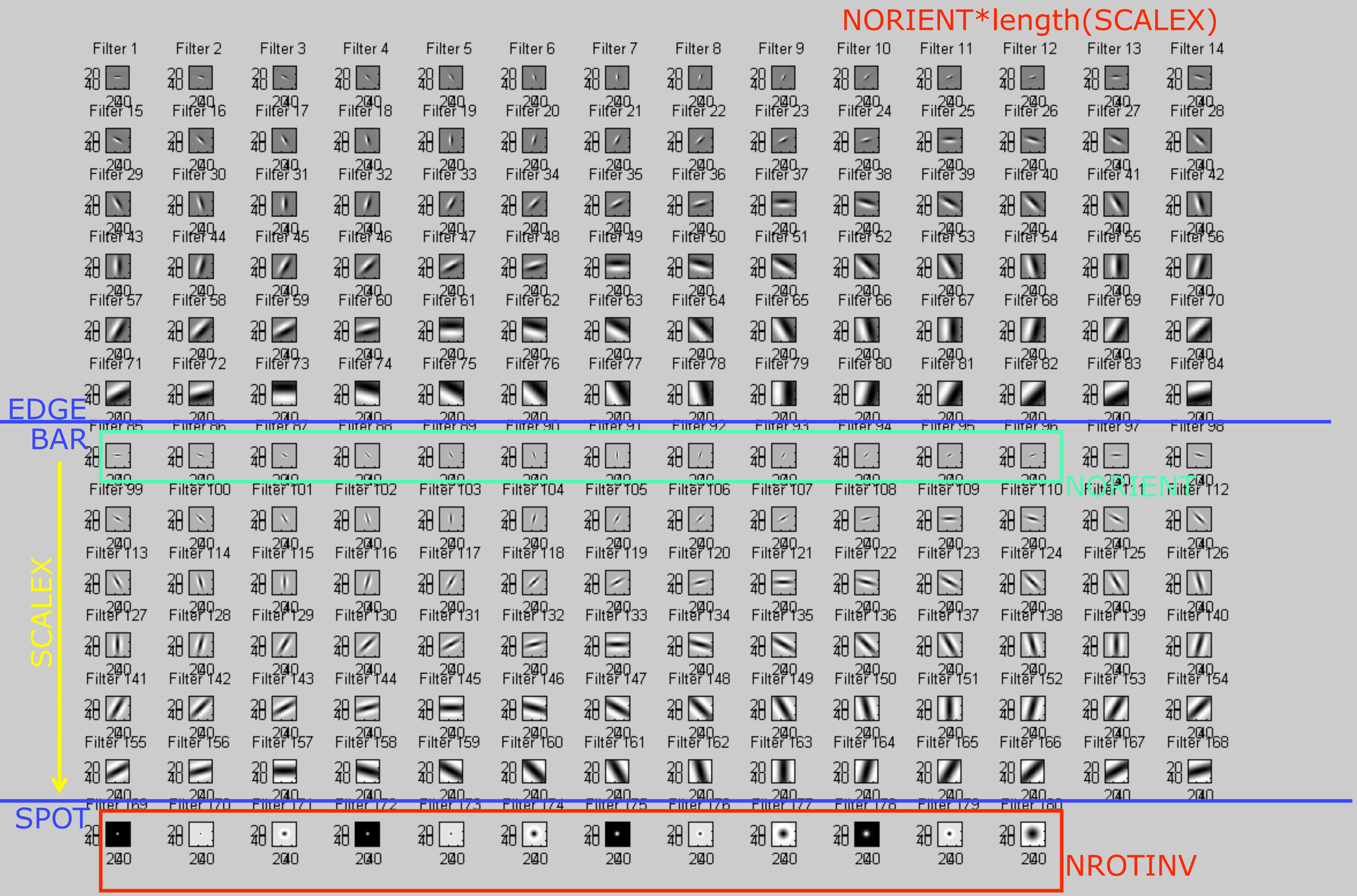
## More on Filter Choices

- There are different types of oriented filters:
  - Weighted sums of Gaussians ✓
  - Wavelets
  - Gabor filters (Gaussian smoothed oriented sine waves)
- There is no obvious advantage to any particular type of oriented filters.
- Typical recommendations is 4-11 scales and 6-18 orientations.
- If one uses many filters:
  - one obtains a detailed representation
  - at the expense of too many convolutions

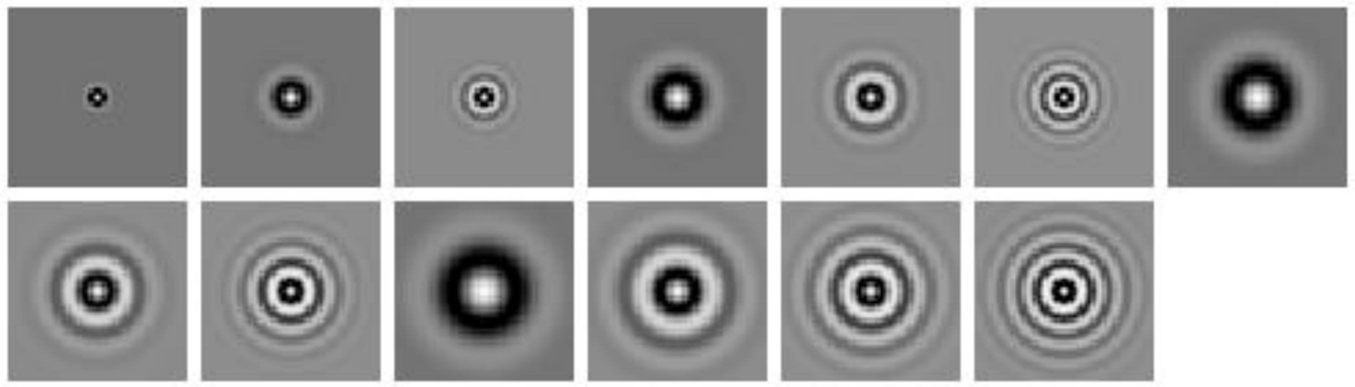
## Leung-Malik Filter Bank



The LM filter bank has a mix of edge, bar and spot filters at multiple scales and orientations. It has a total of 48 filters - 2 Gaussian derivative filters at 6 orientations and 3 scales, 8 Laplacian of Gaussian filters and 4 Gaussian filters.


NORIENT*length(SCALEX)
EDGE
BAR
NORIENT
SCALEX
SPOT
NROTINV

## Schmid Filter Bank

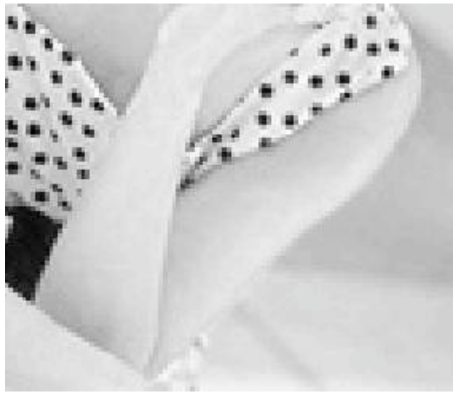


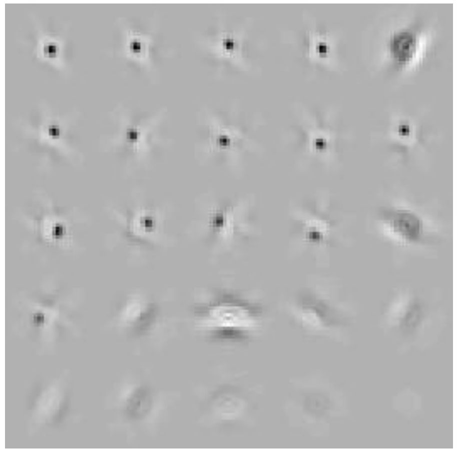
The Schmid filter bank consists of 13 rotationally invariant sinusoidal ("Gabor-like") filters.

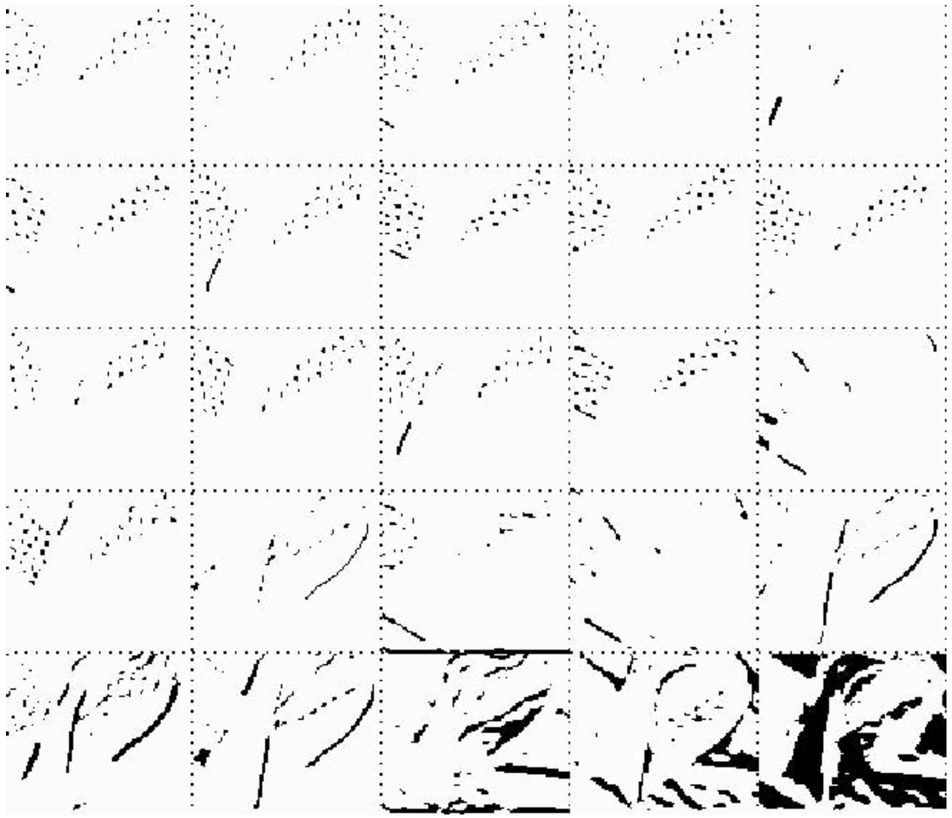
## Use of Oriented Filters

- A high response to a particular oriented filter indicates the presence of a particular subelement at a specific location.
- Is there a pattern in the detected subelements so as to conclude the presence of a texture?
- Investigate a larger region (a larger window 10x10, 100x100) in an image.
- Collect a number of filter bank responses for different locations in the large window.
- Statistically analyze the set of filter responses (mean, variance, etc.) in order to identify a texture.

## Polka Dots Example

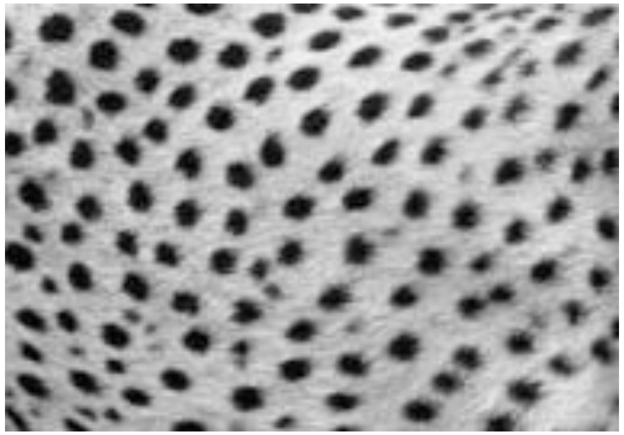


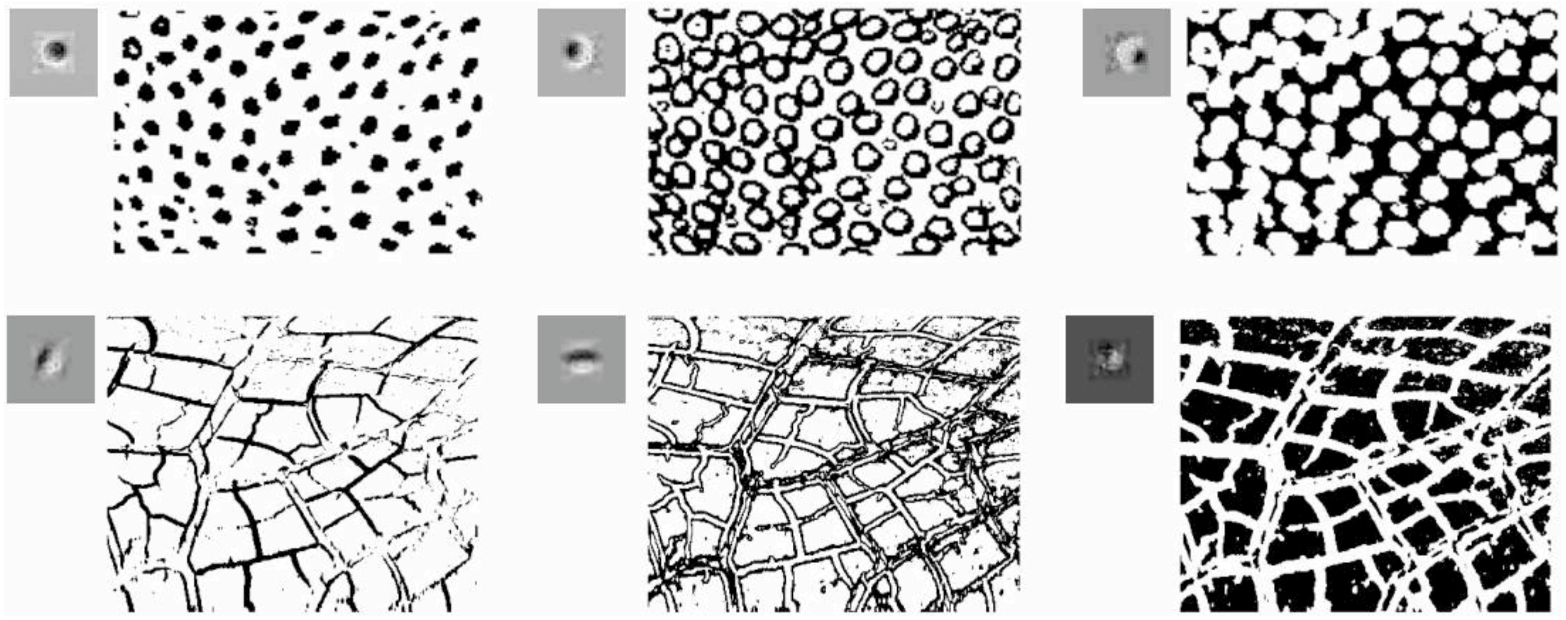




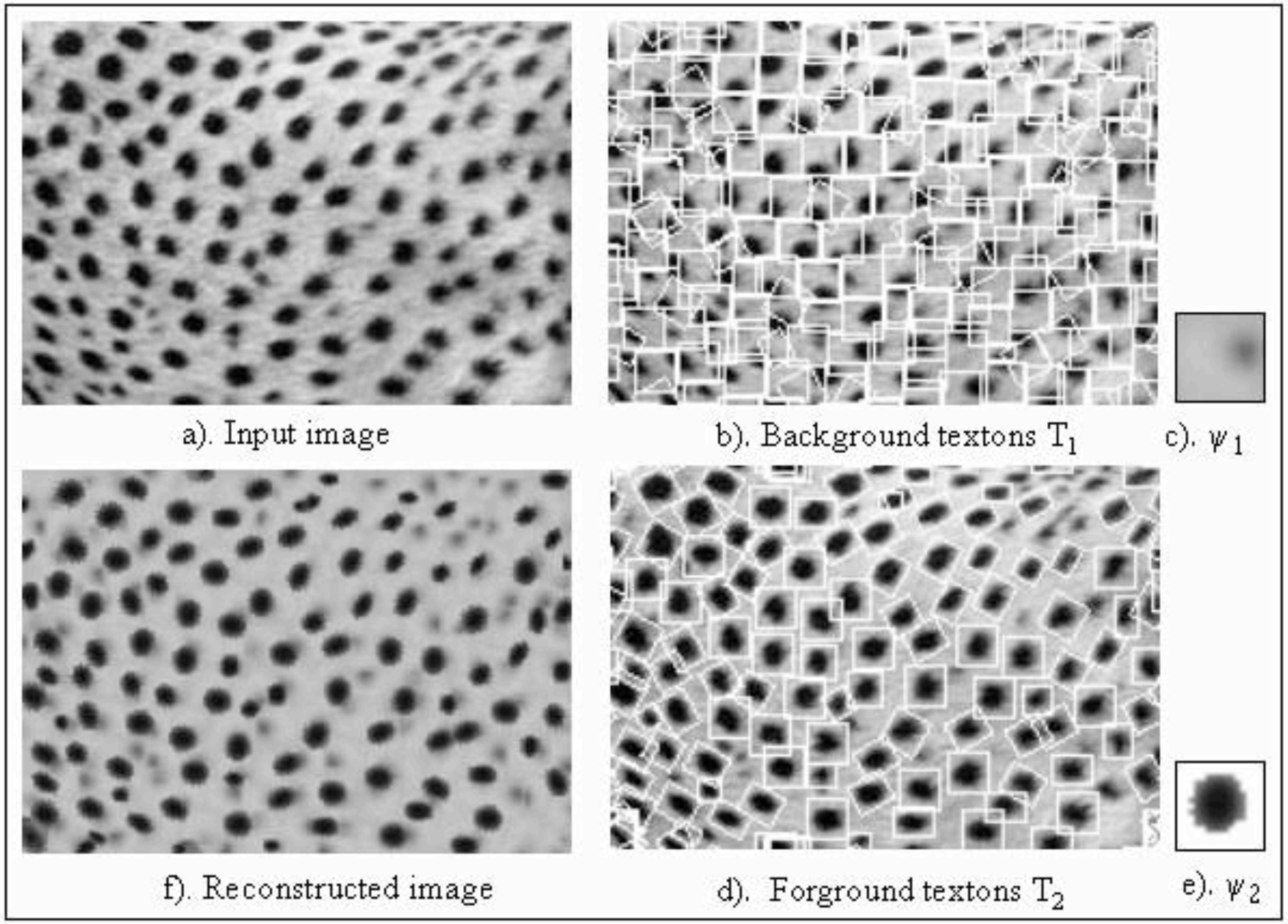
- Polka dots detected using only the textons with the most prominent responses. The LM filter bank was used.

# Selection of Prominent Textons





## Selection of Prominent Textons



## Computation of Texture Statistics

- Let $c$ be the number of texton filters used for texture detection, and $I$ be the input image.
- First compute the textons across the entire image.

```
For each texture filter $F_j$, where $j$ = 1..$c$
    $R_j = F_j * I$
    $R_j = R_j^2$  (optional)
```

- Optional step: Reduce the dimensionality of ***R***, by choosing the most prominent or representative textons. Let $n$ be the new, lower dimension of ***R***.

- Then compute a texture vector for each pixel as follows.

```
For each pixel location (x,y)
    Initialize texture vector $\mathbf{t}$(x,y).
    For each k x k (e.g. k=11, 25, 51, …) texture window centered at (x,y)
        For each texton filter response $R_j$
            $m_{jk}$ = mean($R_{jk}$) % mean of filter j over a (k x k) window centered at x,y
            $s_{jk}$ = std.dev.($R_{jk}$) % std. of filter j over a (k x k) window centered at x,y
            $\mathbf{t}$(x,y) = append_vector ($\mathbf{t}$(x,y), $m_{jk}$, $s_{jk}$)
```

## Texture Detection

- Input: A sample texture image $T_1$ and an image $I$.
- Goal: Return all locations in $I$ where $T_1$ is present in $I$.

1. Apply your favorite texture filter bank in the texture image $T_1$.

   For each texture filter $\underline{F_j}$, where $j = 1..c$

   $RT_j = F_j * T_1$

   $RT_j = RT_j^2$ (optional)

2. Optional step: Reduce the dimensionality of ***RT***, by choosing the most prominent or representative textons. Let $n$ be the new, lower dimension of ***RT***.

3. Then using the previously described function compute a texture vector $\boldsymbol{t}_1$ for the central pixel $(x_c,y_c)$ of the texture image $T_1$.

## Texture Detection (continued)

4. Apply the same filter bank in the image I.

   For each texture filter $F_j$, where $j = 1..c$

   $R_j = F_j * I$

   $R_j = R_j^2$ (optional; be consistent, if the squared response was used for the sample texture, it should be used for the image too.)

5. If the dimensionality of $\boldsymbol{RT}$, was reduced for $T_1$, use the same reduced basis for $\boldsymbol{R}$ as well.

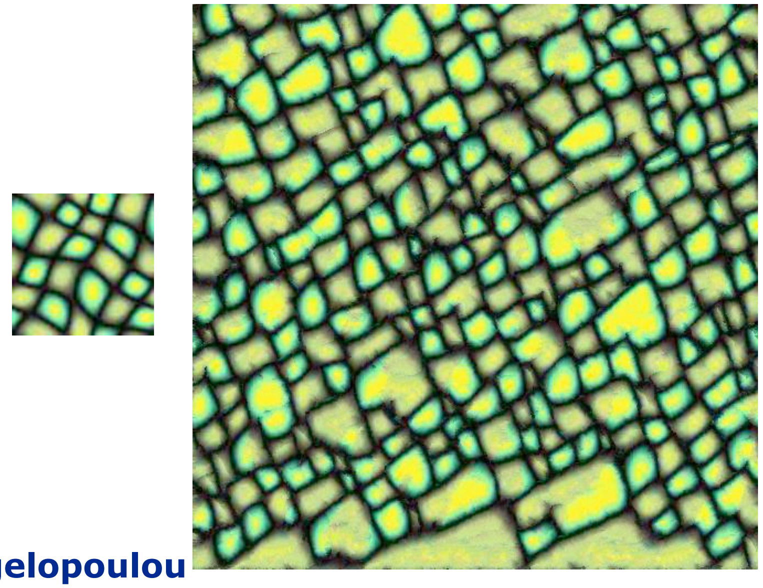
6. Using the previously described function compute a texture vector $\boldsymbol{t}_I$(x,y) for *each* pixel location (x,y) in the image $I$.

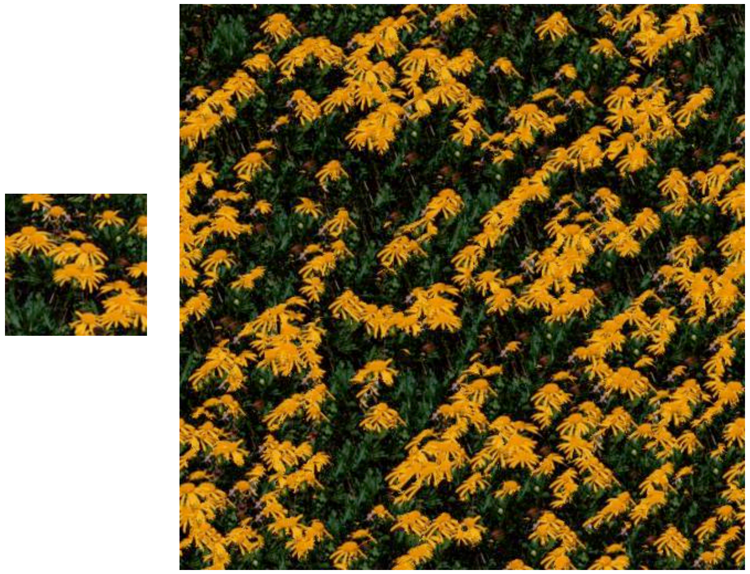
7. Search image $I$ for a vector $\boldsymbol{t}_I$(x,y) that matches the sample texture vector $\boldsymbol{t}_1$.

```
For each pixel location (x,y)
    If d(t_1, t_I(x,y)) < threshold  % d() is a distance metric
        TextureList = Add2List((x,y))
```

## Texture Synthesis

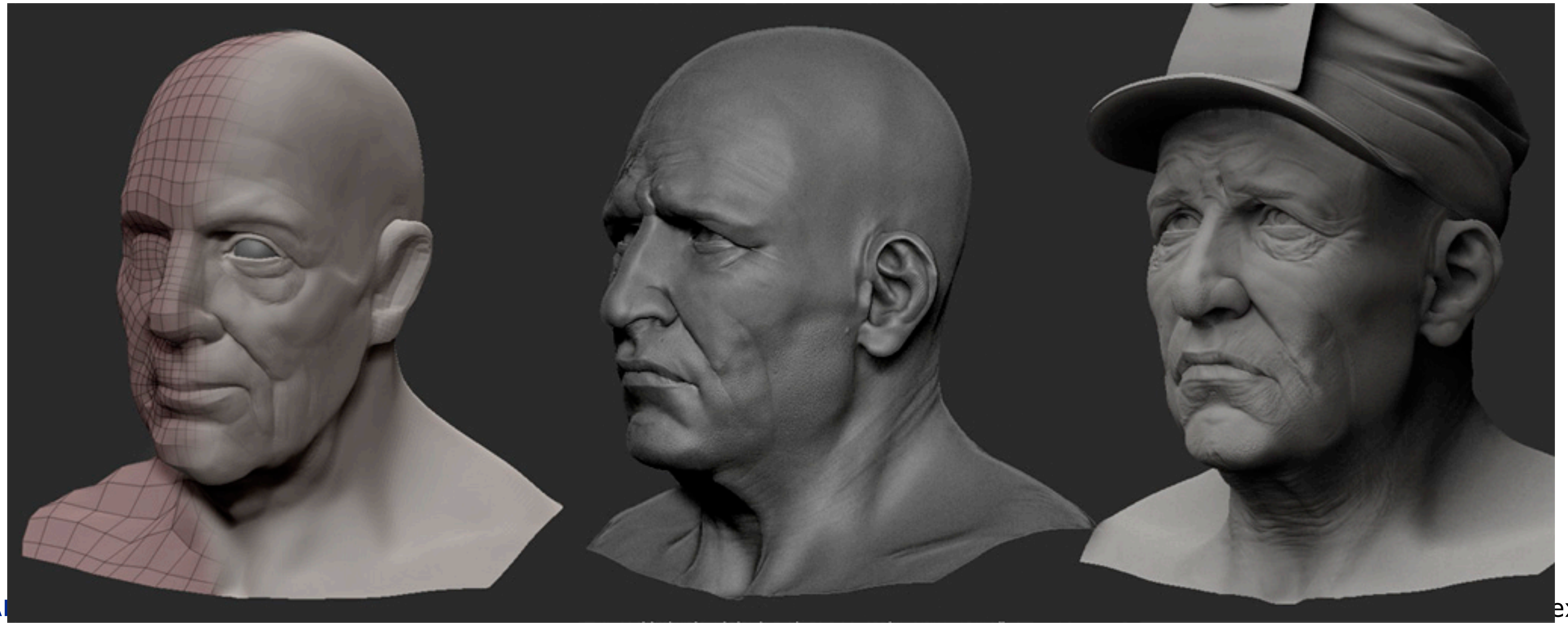
- The goal of texture synthesis is, given an image of a texture $I_T$, to construct large regions of that texture.
- E.g.: Use a patch of grass to create a picture of a lawn.
- Simple repetition will not work. There should be some variation in the pattern.
- Idea: Use statistics in generating such a variation.



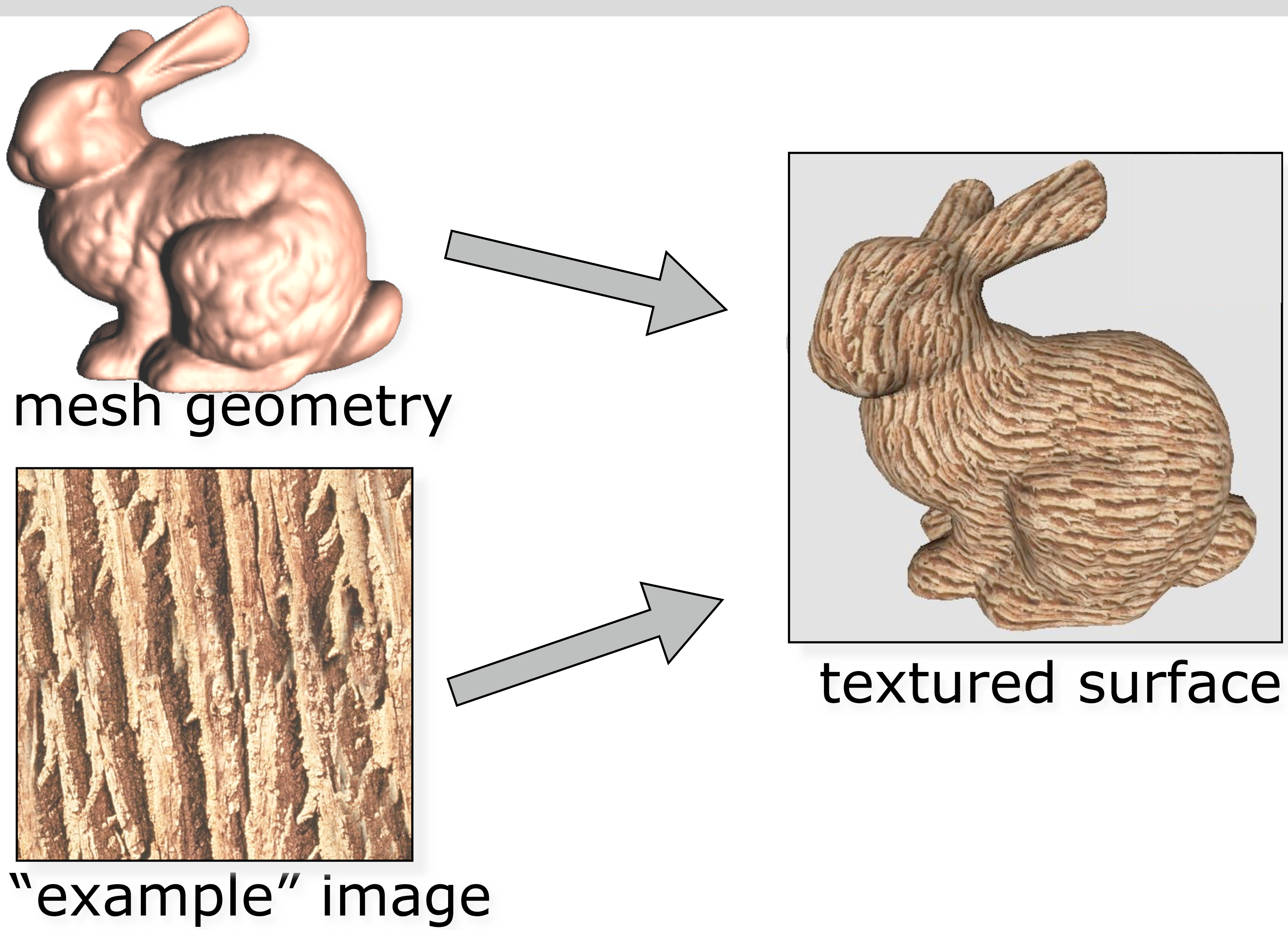


## Texture Synthesis Motivation

- Synthetic images look more realistic if they are textured.
- A common technique for applying a texture on a surface in a synthetic image is through texture mapping.
- Texture mapping is the term used for adding a separately defined texture, or color to a computer-generated 3D model. (Similar to applying a wall-paper to a wall)



## Goal of Texture Mapping



## Texture Mapping

- A texture is stored in an array T(s,t), where s and t are the texture coordinates. $0 \leq s,t \leq 1$
- Each individual pixel (element) in the texture image (texture array) is called a texel.
- Texture map: A mapping that associates a unique point of T with each point on a surface of an object O(x,y,z). It is a one-to-many map. $M : T(s,t) \rightarrow O(x,y,z)$
- We need a set of functions that describe which location (s,t) is mapped to which x-coord., y-coord. and z-coord. of the object O.

$$x = x_f(s,t)$$
$$y = y_f(s,t)$$
$$z = z_f(s,t)$$

## Linear Texture Map

- A surface of an object O(x,y,z) is typically represented in parametric form:

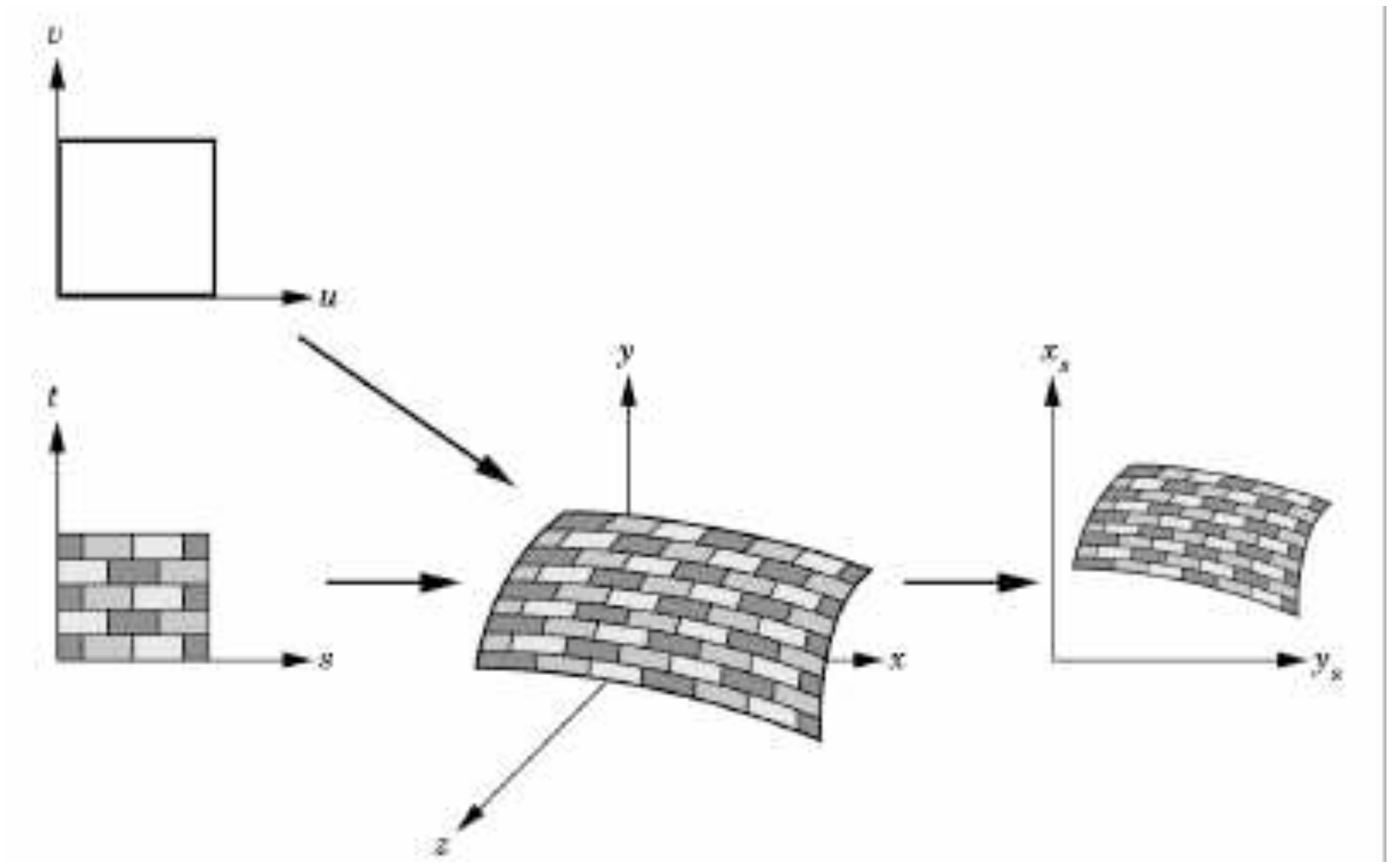
$$O(u,v) = \begin{matrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{matrix}$$

- Given a parameterized surface, we can map a texture point T(s,t) to a particular point p(u,v) by a linear map:

$$u = as + bt + c$$
$$v = ds + et + f$$

- If $ae \neq bd$, then the mapping is invertible.
- Such a linear map will produce incorrect results for curved surfaces.

## Texture Mapping Diagram



## Texture Mapping for Curved Surfaces

- For correct mapping to curved surfaces, one can follow a 2-step approach:

1. Map from T to a simple curved object like a sphere or a cylinder.

- For a cylinder:
$$x = r\cos(2\pi u)$$
$$y = r\sin(2\pi u)$$
$$z = v/h$$

- For a sphere:
$$x = r\cos(2\pi u)$$
$$y = r\sin(2\pi u)\cos(2\pi v)$$
$$z = r\sin(2\pi u)\sin(2\pi v)$$

- Since $0 \le u,v \le 1$, we set $s = u \quad t = v$

2. Map from the intermediate simple curved object to the final one.

## 2nd Step of Texture Mapping

- There are different methods to map from the intermediate to the final object:

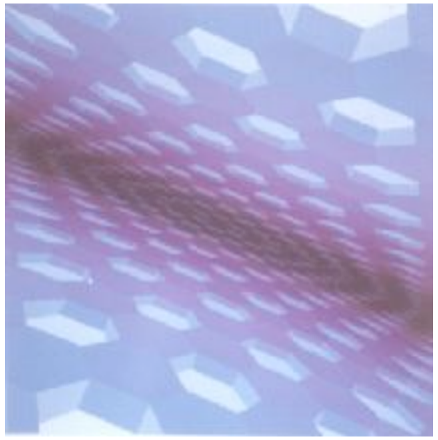
1. Take the surface normal **n**(x,y,z) of the intermediate object. See where it pierces the final object. Texture that intersection point using the texture of the corresponding intermediate point (x,y,z).
2. Take the surface normal **n'**(x,y,z) of the final object. See where it best matches the surface normal of the intermediate object. Texture point (x,y,z) using the texture of the matching intermediate point.
3. Take a line from a point (x,y,z) on the surface of the final object to the center of the intermediate object. See where this line pierces the intermediate object. Texture point (x,y,z) using the texture of the pierced intermediate point.

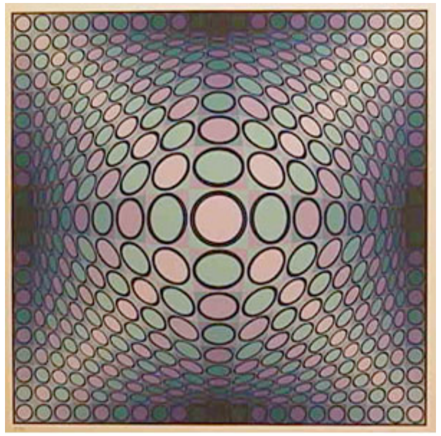
## Texture Pertrubation

- Consider the texture image T(s,t) as a sample of a large probability distribution.
- Obtain other samples of the same distribution by using the original sample as a starting point.
- For example, consider the starting point the mean of the distribution and create the other samples as offsets from the mean.
- If we have prior knowledge on the distribution of texture samples (Gaussian, or number of modes, or std. deviation, etc.) we can get more realistic texture variations.

## Shape from Texture

- The texture on a surface can be an important shape cue.
- The observed distortions are related to foreshortening. Foreshortening causes the texture elements and the gaps between them to "shrink" more in one direction than another.
- Texture Gradient (similar to intensity gradient used in edge detection) measures change in the texture pattern along some direction.
- Texture gradient often corresponds to change in:
  - Distance
  - Orientation





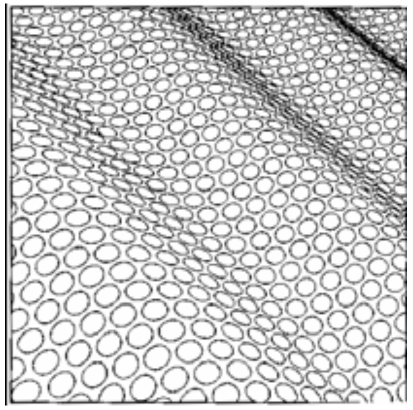
Texture images courtesy of M. Clerc, http://www.cmap.polytechnique.fr/~maureen/TextGrad.html

## Assumptions

1. Uniform texture: The same texture exists at different points on the surface.

- The uniformity assumption implies that the deformation of the texture is caused by the curvature of the surface.

2. The texture resides on the surface and has no thickness.
3. The texture lies on a single 3D surface.



Images courtesy of P. Fua, http://cvlab.epfl.ch/~fua/courses/vision/intro/notes/Texture.pdf

## Slant and Tilt Angles

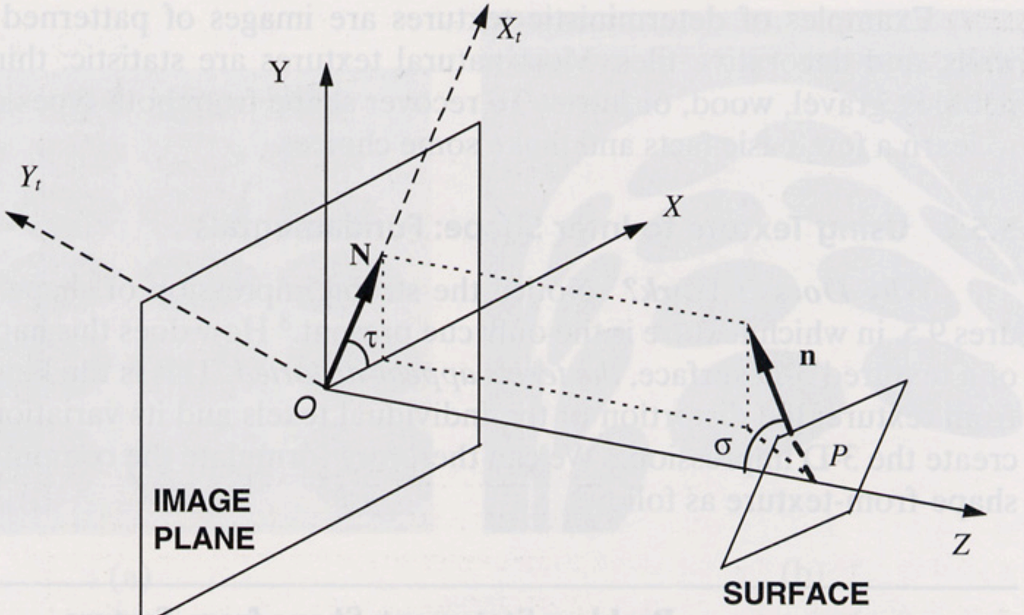
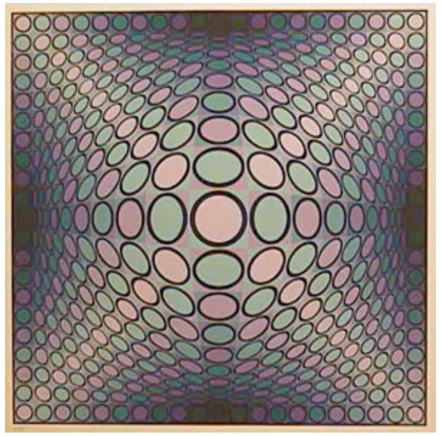


Figure 9.7 The angles tilt, $\tau$, and slant, $\sigma$, defining a unit normal **n**. Notice that this figure does *not* illustrate a projection, which is why the origin of the reference frames is on the image plane, and that these angles are *different* from those used in shape from shading (Figure 9.3).

- Slant angle, $\sigma$: angle between n and the optic axis.
- Tilt angle, $\tau$: angle between the projection of the normal on the image plane and the x- axis.

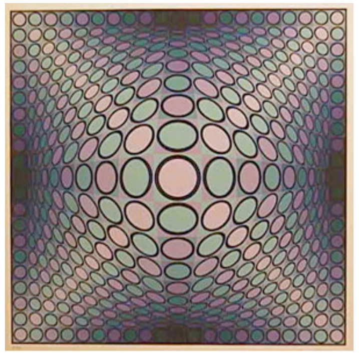
## Example

- Consider the case where the basic texture element is a disk of diameter *d*.
- Assume that there is a disk at the center of the image, where the optic axis pierces the surface. The optic axis intersects that central disk at its center.
- If there is no slant or tilt in the underlying surface, then the disk (circle) appears as a disk in the image.



# Slant, no tilt

- Consider the case where the surface normal **n**, at the center of the disk forms an angle $\sigma \neq 0$ with the optic axis. Assume that there is no tilt, $\tau = 0$ .

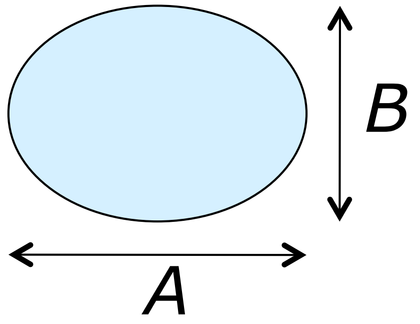


- The disk appears like an ellipse with a horizontal major axis.
- Major axis of the ellipse - no foreshortening

$$A = \left( f / z \right) d$$

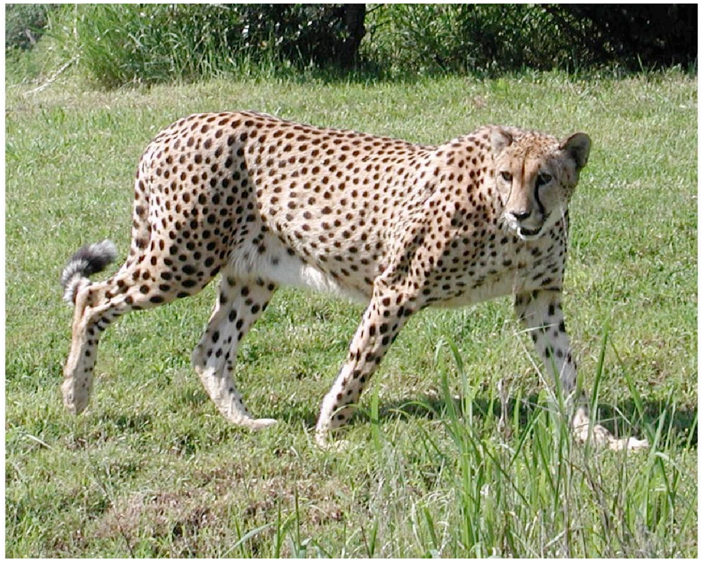
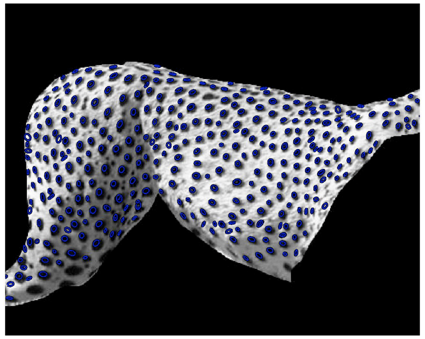
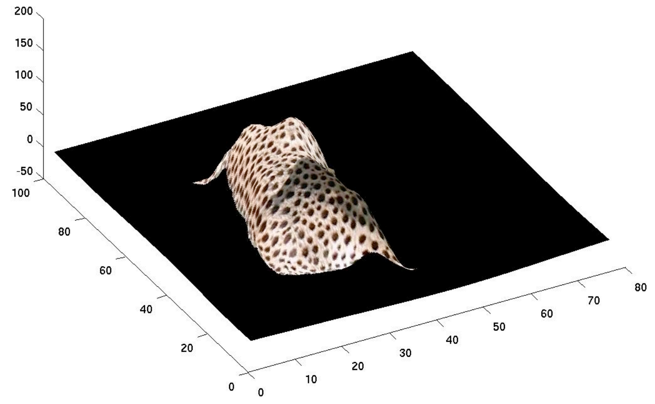
- Minor axis of the ellipse –foreshortening

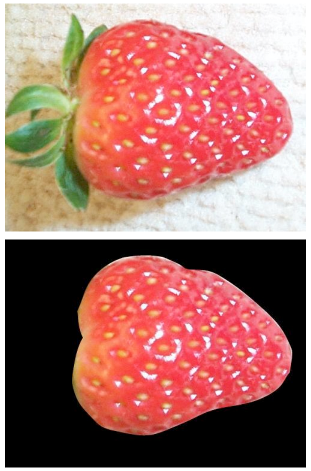
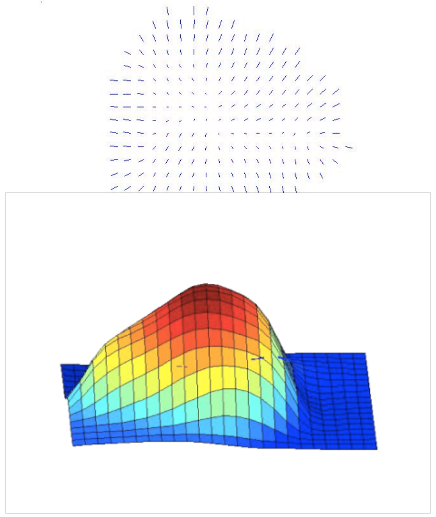
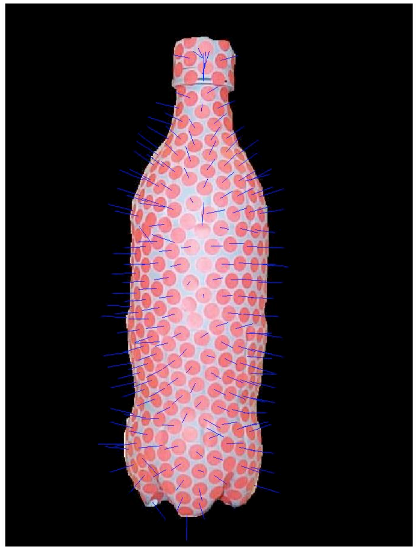
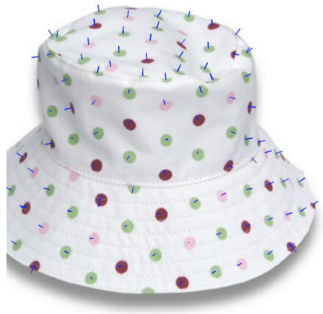
$$B = \left( f / z \right) d \cos \sigma$$



- Surface normal: $\mathbf{n} = A \times B$

## Sample Results of Shape from Texture

Images courtesy of A.M. Loh

## Summary

- Texture is a key property of objects which can be used in:
  - Segmentation
  - Shape recovery
  - Realistic Image Synthesis
- Challenges of texture analysis:
  - Non local
  - No precise definition
  - Subject to deformations
- Sometimes texture is our only (or most significant characteristic) but it can still be quite hard to use (exploit) in practice.

## Image Sources

1. The butterfly, zebra, cheetah and wall images are from the slides by D.A. Forsyth, University of California at Urbana-Champaign.
2. The brick wall image is from Wikipedia http://commons.wikimedia.org/wiki/File:Soderledskyrkan_brick_wall.jpg
3. The picture of the brick building is from Wikipedia http://commons.wikimedia.org/wiki/File:Cowleshall.jpg
4. The leather patch picture is courtesy of http://www.gowfb.com/images/Ferdinand-Furniture/Valencia-Sectional-Sofa-Set.jpg
5. The picture with the leather furniture is courtesy of the Wholesale Furniture Brokers Store http://www.promotionalblog.com.au/promotional-compendiums/promotional-leather-compendiums.html
6. The picture with the wooden grain is courtesy of http://mayang.com/textures/Wood/html/Flat%20Wood%20Textures/index.html
7. The wooden elephant is from Ethnic Indian Décor http://ethnicindianhome.blogspot.com/2010/02/mysore-elephants.html
8. The image with the range of texture types is courtesy of Y.Liu, W-C. Lin, J. Hays , „Near-regular Texture Analysis and Manipulation."
9. The image segmentation examples based on texture are courtesy of Wei Ma and B.S. Manjunath at the UCSB Vision Research Lab, http://vision.ece.ucsb.edu/segmentation/edgeflow/
10. The texture synthesis examples are courtesy of Matthew Fisher, http://www.its.caltech.edu/~matthewf/TextureSynthesis/index.html
11. The "Shape from Texture" images courtesy of J.T. Todd, L. Thaler, T.M.H. Dijkstra, J.J. Koenderink, and A.M.L. Kappers.
12. The texture mapping example is from http://forums.3dtotal.com/showthread.php?t=69468
13. The slide on the goal of texture mapping is courtesy of E. Praun, A. Finkelstein and H. Hoppe http://www.cs.princeton.edu/gfx/proj/lapped_tex/
14. Sketch of texture mapping courtesy of Shayan Sharkar, http://www.cs.cmu.edu/~fp/courses/02-graphics/pdf-2up/13-texture.pdf