Texture

Objects do not appear as uniform color/intensity regions due to variable surface geometry, lighting, albedo etc.

Sometimes the "variation is uniform" - Texture

• Variation
• Repetitiveness
Texture presence - Pros & Cons

(-) Texture generates edges inside objects
(+) Texture helps to identify objects
(+ ) Texture helps to segments objects (can overcome significant intensity change, shadows)
(+ ) Texture helps to find 3D surface shape.

Human vision can detect texture differences, pre-attentively

Texture tasks

Texture Analysis
• Take two patches
• Are they the same “stuff”? 
• NOT “Is this a wall picture?”

“Same” or “different”

Texture Synthesis
• Take a patch
• Synthesize a different patch of the same texture
Analysis

How to decide that two patches are the same texture?

• Find a descriptor for each texture patch
• Compare descriptors

But which descriptors?
How to quantify the type of variation?

Some not so effective ideas:
• The gray levels vector
• “busyness” – How many edges?
• Histogram of gradient directions

Co-occurrence matrix

• Co-occurrence matrix = probability distributions for intensity pairs
• Information on some aspects of the spatial configurations

• Problem: How to choose the spatial vector d.
Filter based approaches

- Filter the image with several filters emphasizing different local properties of the variation
  - Laws filters: intensity, edge, LoG, ripple
  - Gabor filters: products of sin/cos function with Gaussians, similar to Gaussian derivatives.
- Possibly normalize by intensity to get invariance.
- Average over neighborhoods to get a vector of descriptors.

Laws filters:

Another “filter bank”

- Gabor filters: products of sin/cos function with Gaussians, similar to Gaussian derivatives.
Gabor filter based segmentation

- Gabor filter descriptor
- K-means clustering

- Broadatz collection

Histogram of textons

Also called ”Bag of Words”

- **Preprocessing:** Learn a set of “textons” (types)
  - For all images of interest, characterize every local neighborhood by some vector, e.g. responses of different (Gabor) filters.
  - Run K-means clustering. Every cluster is a textone.

- **Usage:** Characterize a patch of texture
  - Characterize every local neighborhood by a vector
  - Assign it to a textone = the nearest cluster.
  - Calculate an histogram of textons

For stochastic textures, it is the identity of the textons, not their spatial arrangement, that matters
An idealized example

Texture characterization type histogram
Local Binary Pattern (LBP)

• Specifies a 8 bit descriptor to every point

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>101</td>
<td>103</td>
</tr>
<tr>
<td>40</td>
<td>50</td>
<td>80</td>
</tr>
<tr>
<td>50</td>
<td>60</td>
<td>90</td>
</tr>
</tbody>
</table>

• Simple and surprisingly effective.
• Sometimes better in coarser scale
• Descriptors with two transitions are common.

• The texture descriptor is a histogram over point descriptors - histogram over "types" of points

Synthesis

The problem is of great theoretical interest:

What is the distribution of images?

• What is a good model for randomizing a function that looks like an image
• Definitely not a randomly drawn i.i.d set of grey levels.

• Here: Example-based non-parametric models
Non parametric Texture Synthesis

Efros & Leung: Non Parametric Sampling

Assuming Markov property, compute $P(p|\mathcal{N}(p))$
- Building explicit probability tables is infeasible
- Instead, we search the input image for all similar neighborhoods — that's our pdf for $p$
- To sample from this pdf, just pick one match at random

Synthesis Results

french canvas

rafia weave
Varying Window Size

Increasing window size