ICS 683
Advanced Computer Vision

Department of Information and Computer Sciences
University of Hawai`i at Mānoa

Kyungim Baek

Lecture 11

- Photometric Stereo
- Texture
  - Definition
  - Quantitative measures and representations

- A significant part of the materials in this lecture notes are from the similar courses offered by George C. Stockman, David Forsyth, and Francesc Moreno-Noguer. I would like to thank the instructors for the slides and content used in this lecture.

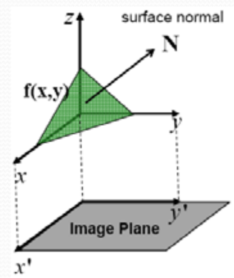
Announcements

- Homework assignment #3
  - Due: **Tuesday, October 15**
- Project proposal with comments is uploaded in your Drop Box folder in Laulima
  - Grading has been posted also
- Project presentations
  - Intermediate presentation: Nov. 5 & Nov. 7
  - Final presentation: Dec. 5, Dec. 10, and Dec. 12

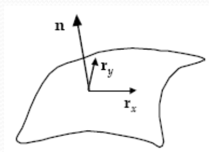
Previously… Surface Orientation

- **Surface:** $s = (x, y, f(x, y))$
- **Tangent vectors:** $\dfrac{\partial s(x,y)}{\partial x} = \left(1, 0, f_x\right)$

  $\dfrac{\partial s(x,y)}{\partial y} = \left(0, 1, f_y\right)$



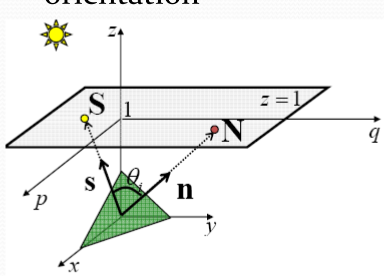
- **Surface normal**: $\mathbf{N}(x,y) = \dfrac{\partial s}{\partial x} \times \dfrac{\partial s}{\partial y}$



$$= \left(-f_x, -f_y, 1\right)$$
$$= (p, q, 1)$$

Previously… Gradient Space

- $z = 1$ plane is called the Gradient Space (pq plane)
  - Its components p and q are the surface slopes in the x- and y- direction
  - Every point on it corresponds to a particular surface orientation



unit normal vector:

$$\mathbf{n} = \frac{\mathbf{N}}{|\mathbf{N}|} = \frac{(p, q, 1)}{\sqrt{p^2 + q^2 + 1}}$$

unit source vector:

$$\mathbf{s} = \frac{\mathbf{S}}{|\mathbf{S}|} = \frac{(p_s, q_s, 1)}{\sqrt{p_s^2 + q_s^2 + 1}}$$

Previously… Reflectance Map

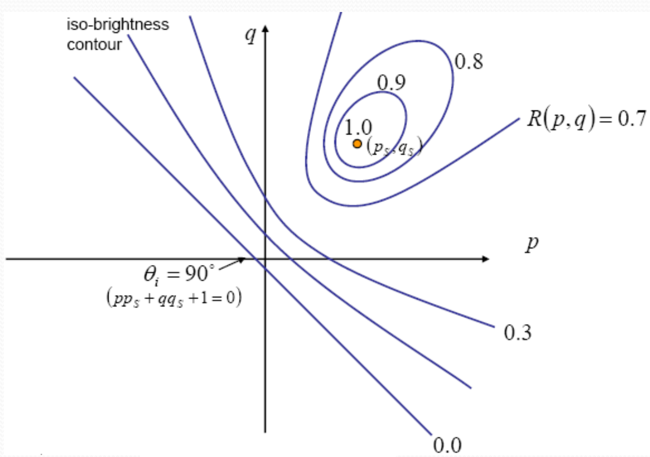
- Relates image intensity $I(x, y)$ to surface orientation (p, q) for GIVEN source direction and surface reflectance
- Lambertian case: $I = \rho_d L_i \cos\theta_i = \rho_d L_i (\mathbf{n} \cdot \mathbf{s})$

Let $\rho_d L_i = 1$ then $\boxed{I = \cos\theta_i = \mathbf{n} \cdot \mathbf{s}}$

Reflectance Map

$$I = \mathbf{n} \cdot \mathbf{s} = \frac{(pp_s + qq_s + 1)}{\sqrt{p^2 + q^2 + 1}\sqrt{p_s^2 + q_s^2 + 1}} = \boxed{R(p, q)}$$

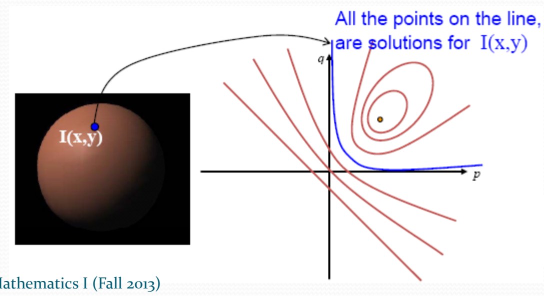
Previously… Reflectance Map



Note: $R(p, q)$ is maximum when $(p, q) = (p_s, q_s)$

Shape from a Single Image?

- Given a single image of an object with known surface reflectance taken under a known light source, can we recover the shape of the object?
- Given $R(p, q)$ ((p_s, q_s) and surface reflectance) can we determine (p, q) uniquely for each image point?



NO!

Photometric Stereo: Concept

- Solution: Take more images with varying direction of illumination while holding the viewing direction constant (i.e. no change in imaging geometry)
  - Pixel (x, y) in the images corresponds to the same object point

    $\Rightarrow$ corresponds to the same gradient (p, q)

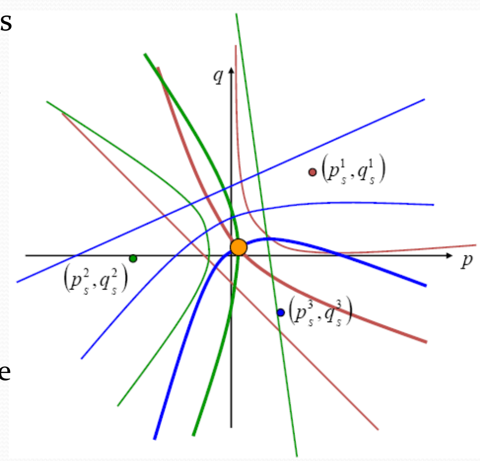
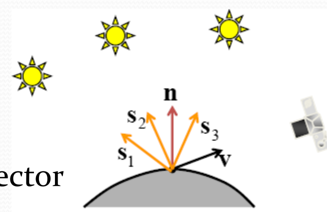


Image Model

- For each point source, assume that we know the source vector ($\mathbf{s}_j$), the surface reflectance, and the scaling constant k of the linear camera
- Image intensity (Lambertian case): assume that $kL_j = 1$

$$I_j(x, y) = k\rho(x, y)L_j \cos\theta_j$$
$$= \rho(x, y)(\mathbf{n}(x, y) \cdot \mathbf{s}_j)$$

  - $\rho(x, y)$: albedo
  - $\mathbf{n}(x, y)$: unknown surface normal vector



With Three Light Sources

$$I_1(x, y) = \rho(x, y)(\mathbf{n}(x, y) \cdot \mathbf{s}_1)$$
$$I_2(x, y) = \rho(x, y)(\mathbf{n}(x, y) \cdot \mathbf{s}_2)$$
$$I_3(x, y) = \rho(x, y)(\mathbf{n}(x, y) \cdot \mathbf{s}_3)$$

- In matrix form:

surface description

$$\begin{bmatrix} I_1(x, y) \\ I_2(x, y) \\ I_3(x, y) \end{bmatrix} = \rho(x, y)\begin{bmatrix} \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \mathbf{s}_3^T \end{bmatrix}\mathbf{n}(x, y) = \begin{bmatrix} \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \mathbf{s}_3^T \end{bmatrix}\mathbf{g}(x, y)$$

$\mathbf{g}(x, y)$
$= \rho(x, y)\mathbf{n}(x, y)$

Illumination (& camera) property

Solving the Equations

$$\underbrace{\begin{bmatrix} I_1(x,y) \\ I_2(x,y) \\ I_3(x,y) \end{bmatrix}}_{3\times 1} = \underbrace{\begin{bmatrix} \mathbf{s}_1^T \\ \mathbf{s}_2^T \\ \mathbf{s}_3^T \end{bmatrix}}_{3\times 3} \underbrace{\mathbf{g}(x,y)}_{3\times 1} \Rightarrow \mathbf{I} = \mathbf{S}\mathbf{g}(x,y)$$

- Solving for $\mathbf{g}(x, \mathrm{y})$:

$$\mathbf{g}(x,y) = \mathbf{S}^{-1}\mathbf{I}(x,y)$$

Recovering Normal and Albedo

- $\mathbf{g}(x, y) = \rho(x, y)\mathbf{n}(x, y)$
  $\Rightarrow$ albedo (surface reflectance) is the magnitude of $\mathbf{g}$

$$\rho(x,y) = |\mathbf{g}(x,y)| \longleftarrow \text{Since } \mathbf{n} \text{ is a unit vector}$$

- This yield a check: if the magnitude of $\mathbf{g}(x, \mathrm{y})$ is greater than 1, there is a problem
- Recovering surface normal

$$\mathbf{n}(x,y) = \frac{\mathbf{g}(x,y)}{|\mathbf{g}(x,y)|} = \frac{\mathbf{g}(x,y)}{\rho(x,y)}$$

More than Three Light Sources

$$\begin{bmatrix} I_1(x,y) \\ \vdots \\ I_N(x,y) \end{bmatrix} = \begin{bmatrix} \mathbf{s}_1^T \\ \vdots \\ \mathbf{s}_N^T \end{bmatrix} \mathbf{g}(x,y) \Rightarrow \mathbf{I}_{N\times 1} = \mathbf{S}_{N\times 3}\mathbf{g}_{3\times 1}$$

- Least squares solution:

pseudo-inverse

$$\mathbf{S}^T\mathbf{I} = \mathbf{S}^T\mathbf{S}\mathbf{g}$$

$$\mathbf{g} = \boxed{(\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T}\mathbf{I}$$

- Measure albedo and recover surface normal as before

Trick for Dealing with Shadows

- Shadowed pixels are outliers
- Weigh each equation by the pixel brightness (zeros out the contributions from shadowed pixels)

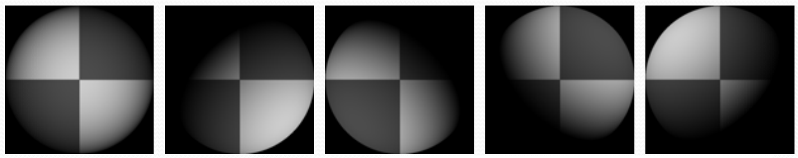
$$I_i(x,y)\big(I_i(x,y)\big) = I_i(x,y)\big(\rho(x,y)(\mathbf{n}(x,y)\cdot\mathbf{s}_i)\big)$$

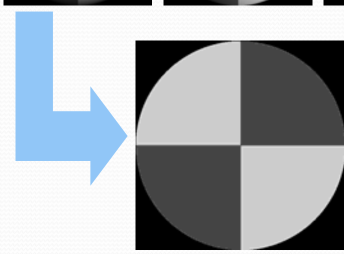
- Gives weighted least-squares matrix equation:

$$\begin{bmatrix} I_1^2 \\ \vdots \\ I_N^2 \end{bmatrix} = \begin{bmatrix} I_1\mathbf{s}_1^T \\ \vdots \\ I_N\mathbf{s}_N^T \end{bmatrix} \rho\mathbf{n} = \begin{bmatrix} I_1\mathbf{s}_1^T \\ \vdots \\ I_N\mathbf{s}_N^T \end{bmatrix} \mathbf{g}$$

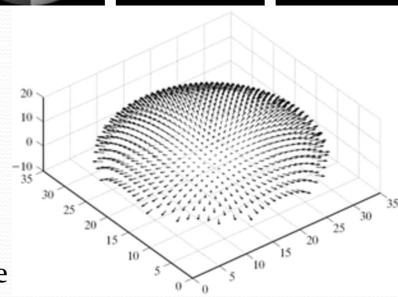
- Solve for $\boldsymbol{g}$ as before

Example





Measured reflectance



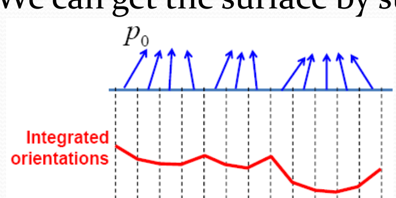
Recovered normal field

Depth from Normals (I)

- Photometric stereo "just" recovers the surface normals
- We need to compute the depth!

Simple Method: Shape by Integration (1D)

- Partial derivatives give the change in surface height with a small step
- We can get the surface by summing the changes

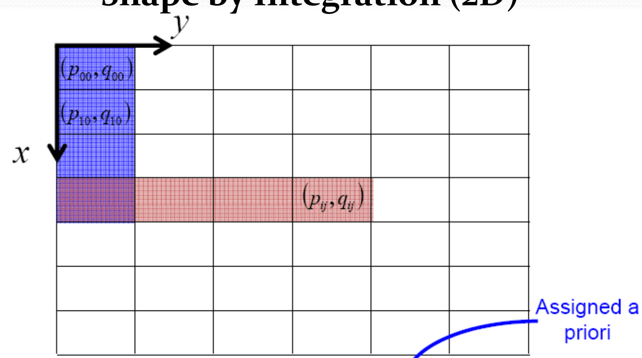


$$p_i = \left.\frac{\partial f}{\partial x}\right|_{x=x_i}$$

$$f(x_i) = \int_0^{x_i} \frac{\partial f}{\partial x} dx = f(x_{i-1}) + p_i$$

Depth from Normals (II)

Shape by Integration (2D)

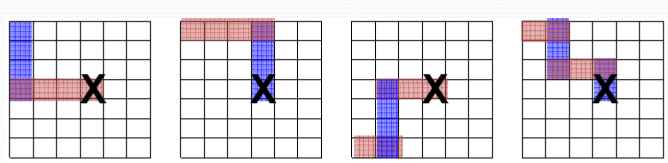


$$f_{ij} = \int_0^{x_i} \frac{\partial f}{\partial x} dx + \int_0^{y_j} \frac{\partial f}{\partial y} dy = f_{00} + \sum_0^i p_{i0} + \sum_0^j q_{ij}$$

Depth from Normals (III)

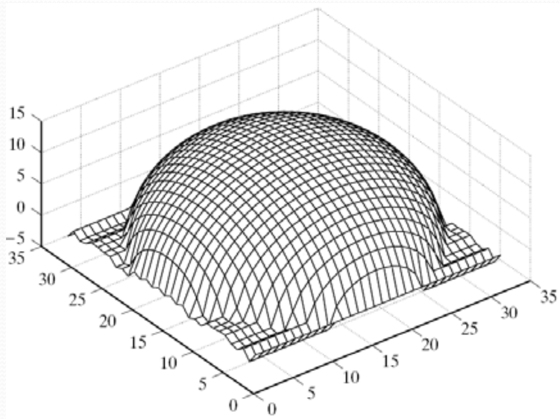
Shape by Integration (2D)

- Limitations:
  - Integration is good only if object shape is continuous
  - It is sensitive to noise
    - Solution: Average the results over many different paths
    - High cost and still inaccurate



Recovered Surface Shape

- Surface recovered by integration



Photometric Stereo: Example

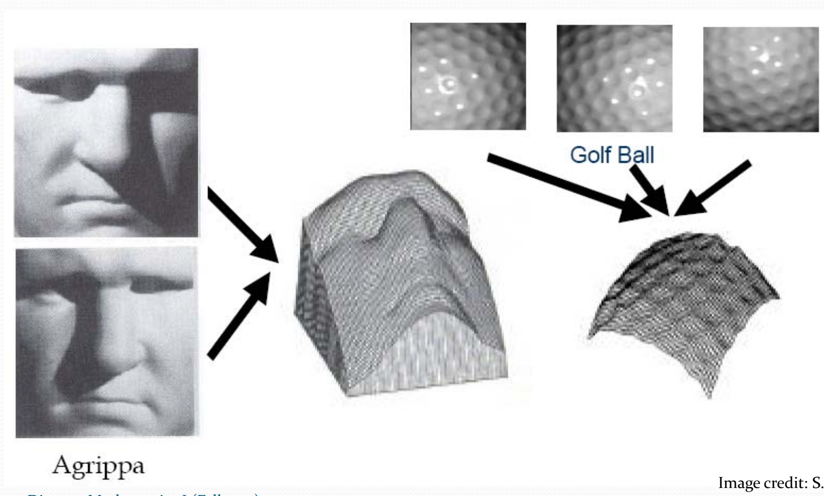


Image credit: S. Park

Photometric Stereo: Example

Input (1 of 12) | Normals | Shaded rendering | Textured rendering

Image credit: S. Seitz

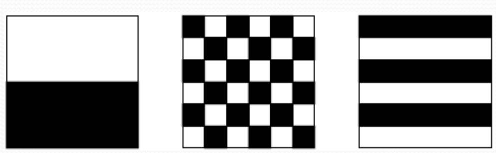
Photometric Stereo: Limitations

- Simplistic reflectance and lighting model
  - Doesn't work for shiny things, semi-translucent things
  - No inter-reflections
- Camera and lights have to be distant
- Calibration requirements
  - Measure light source directions, intensities
  - Camera response function
- Integration is tricky
- Newer work addresses some of these issues

Texture

Introduction (I)

- Texture
  - is another feature that can help to segment images into regions of interest and to classify those regions
  - gives us information about the spatial arrangement of the colors or intensities in an image

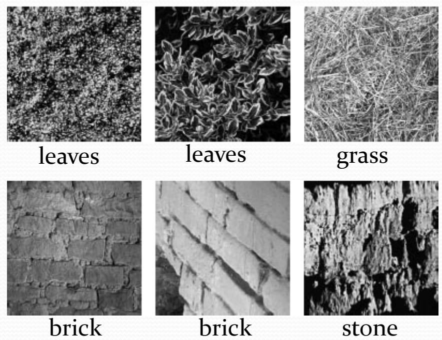


block pattern checkerboard striped pattern

Three different textures with the same distribution of black and white

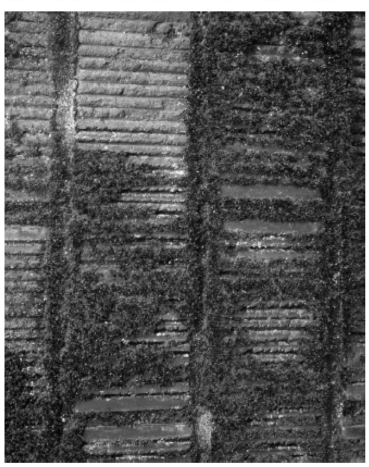
Introduction (II)

- Images of most natural objects exhibit visual texture which provides important visual cue



Nature textures (from the MIT Media Lab VisTex database)

Typical Textured Images

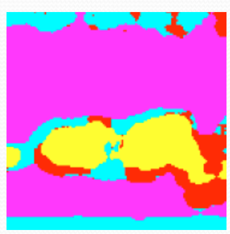


Standard Problems

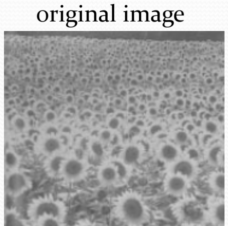
- Texture segmentation
  - Breaking an image into components within which the texture is constant
  - Issues: texture representation and how to determine the segment boundaries
- Texture synthesis
  - Construct large regions of texture from small example images
- Shape from texture
  - Recover surface orientation or surface shape from image texture

Texture Segmentation

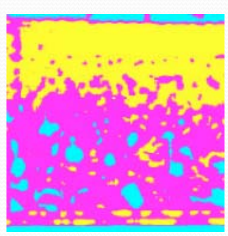
original image



segmentation into 4 clusters

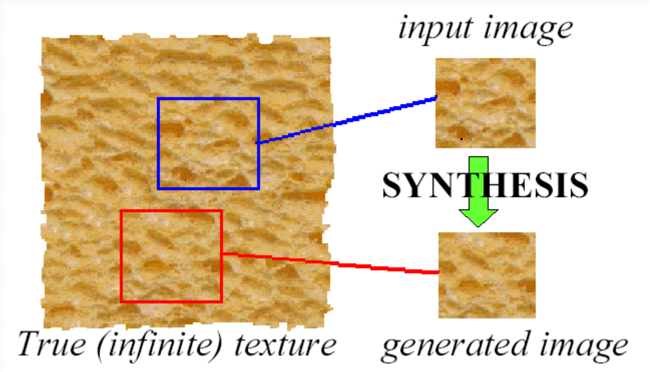


original image



segmentation into 3 clusters

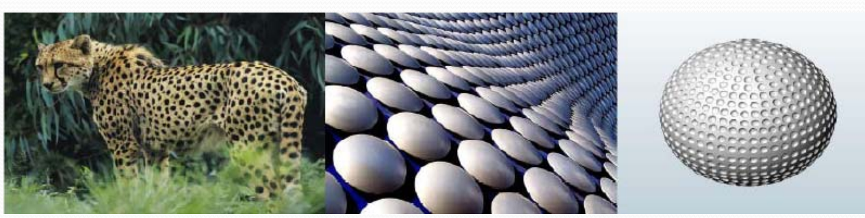
Texture Synthesis



- Given a finite example, generate texture sample (that is large enough, satisfies constraints,…)

Slide credit: A. Efros

Shape from Texture



- Guess the shape of a surface from the deformation of its texture element

What is Texture?

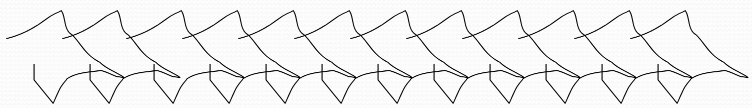
- **Structural approach**
  - Texture is a set of primitive *texels* (or *textons*) in some regular or repeated relationship
  - Work well for man-made, regular patterns
- **Statistical approach**
  - Texture is a quantitative measure of the arrangement of intensities in a region
  - More general and easier to compute
  - Used more often in practice

Aspects of Texture

- Size or granularity (sand versus pebbles versus boulders)
- Directionality (stripes versus sand)
- Random or regular (sawdust versus woodgrain; stucko versus bricks)
- Concept of texture elements (texel) and spatial arrangement of texels

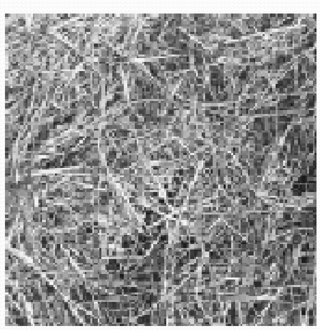
Problem with Structural Approach

How do you decide what is a texel?

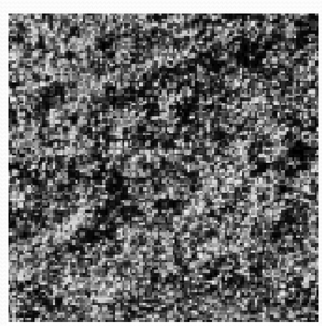


Ideas?

Natural Textures from VisTex



grass



leaves

What/Where are the texels?

The Case for Statistical Texture

- Segmenting out texels is difficult or impossible in real images
- Numeric quantities or statistics that describe a texture can be computed from the gray tones (or colors) alone
- This approach is less intuitive, but is computationally efficient
- It can be used for both classification and segmentation

Some Simple Statistical Texture Measures

- **Edge Density and Direction**
  - Use an edge detector as the first step in texture analysis
  - The number of edge pixels in a fixed-size region tells us how busy that region is
  - The directions of the edges (usually available as a byproduct of the edge-detection process) also help characterize the texture pattern

Two Edge-based Texture Measures

1. Edgeness per unit area

$$F_{edgeness} = |\{\, p \mid \text{gradient\_magnitude}(p) \geq \text{threshold}\}| \,/\, N$$

where N is the number of pixels in the unit area

2. Edge magnitude and direction histograms

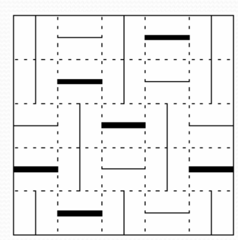
$$F_{magdir} = (H_{mag}(\mathrm{R}), H_{dir}(\mathrm{R}))$$

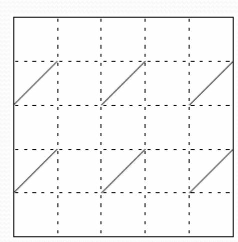
where $H_{mag}(\mathrm{R})$ and $H_{dir}(\mathrm{R})$ are the normalized histograms of gradient magnitudes and gradient directions in a region R, respectively.

How would you compare two histograms?

Example

- Two bins for gradient magnitude histograms: (dark edges, light edges)
- Three bins for gradient direction histograms: (horizontal, vertical, diagonal)
- Top image
  - $F_{edgeness} = 25/25 = 1.0$
  - $H_{mag} = (6/25, 19/25) = (0.24, 0.76)$
  - $H_{dir} = (12/25, 13/25, 0/25) = (0.48, 0.52, 0.0)$
- Bottom image
  - $F_{edgeness} = 6/25 = 0.24$
  - $H_{mag} = (0/25, 6/25) = (0.0, 0.24)$
  - $H_{dir} = (0/25, 0/25, 6/25) = (0.0, 0.0, 0.24)$





5×5 images

Local Binary Partition (LBP) Measure

- For each pixel p, create an 8-bit number $b_1b_2b_3b_4b_5b_6b_7b_8$, where $b_i = 0$ if neighbor i has value less than or equal to p's value and 1 otherwise

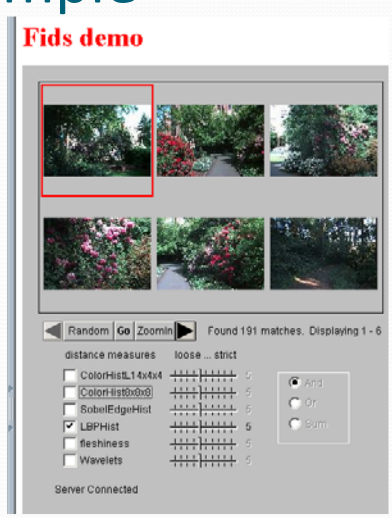
	1	2	3	
	100	101	103	
8	40	50	80	4
	50	60	90	5
	7	6		

$\longrightarrow$ 1 1 1 1 1 1 0 0

- Represent the texture in the image (or a region) by the histogram of these numbers
- Two images or regions are compared by computing the L1 distance between their histograms: $L_1(H_1, H_2) = \sum_{i=1}^{n} |H_1[i] - H_2[i]|$

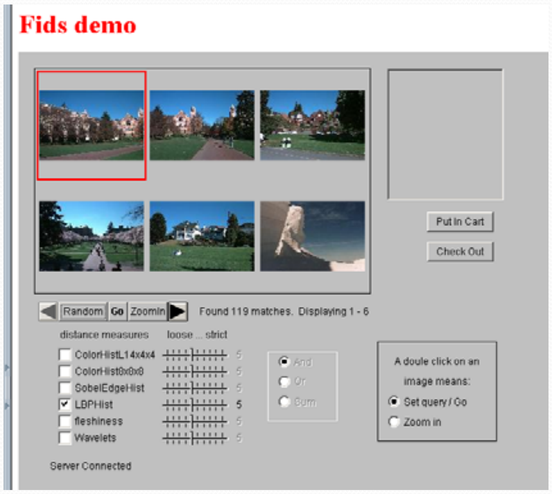
LBP Measure: Example

- Fids (Flexible Image Database System) is retrieving images similar to the query image using LBP texture as the texture measure and comparing their LBP histograms



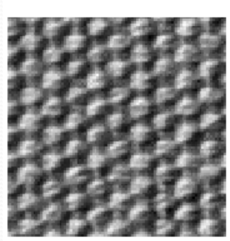
LBP Measure: Example

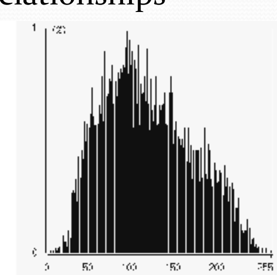
- Low-level measures don't always find semantically similar images



Histograms

- Principle
  - Intensity probability distribution
  - Captures global brightness information in a compact, but incomplete way
  - Doesn't capture spatial relationships
- Example

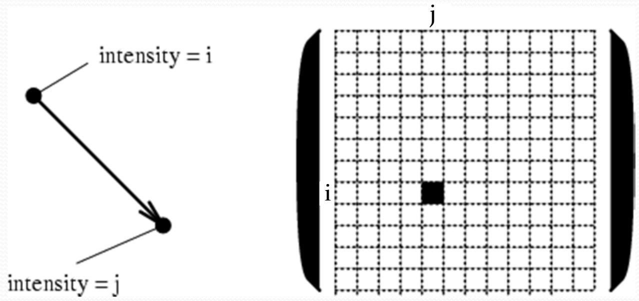




Slide adapted from M. Pollefeys

Co-occurrence Matrices (I)

- Probability distributions for intensity pairs
- Contains information on some aspects of the spatial configurations



Slide credit: M. Pollefeys

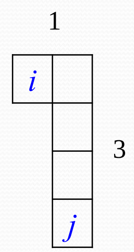
Co-occurrence Matrices (II)

- A co-occurrence matrix is a 2D array **C** in which
  - Both the rows and columns represent a set of possible image values
  - $\mathbf{C}_d(i, j)$ indicates how many times value i co-occurs with value j in a particular spatial relationship d
  - The spatial relationship is specified by a vector $d = (dr, dc)$
    - dr: displacement in rows (downward)
    - dc: displacement in columns (to the right)
  - $\mathbf{C}_d(i, j) = |\{(r, c) \mid \boldsymbol{I}(r, c) = i \text{ and } \boldsymbol{I}(r+dr, c+dc) = j\}|$

Co-occurrence Matrices (III)

1	1	0	0
1	1	0	0
0	0	2	2
0	0	2	2
0	0	2	2
0	0	2	2

gray-tone image



$d = (3, 1)$

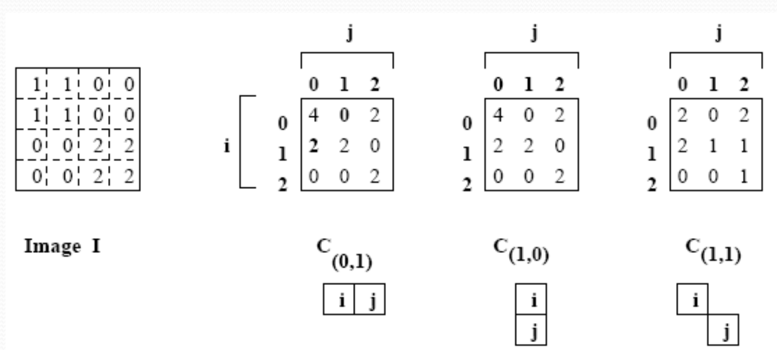
	0	1	2
0	1	0	3
1	2	0	2
2	0	0	1

co-occurrence matrix, $\mathbf{C}_d$

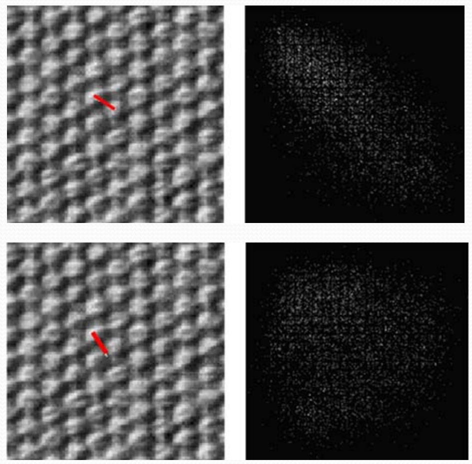
From $\mathbf{C}_d$ we can compute $\mathbf{N}_d$, the normalized co-occurrence matrix, where each value is divided by the sum of all the values.

Co-occurrence Matrices (IV)

- Illustration with a 4×4 image **I** and three different spatial configurations



Co-occurrence Matrices (V)



Slide credit: M. Pollefeys

Co-occurrence Features

- Standard features derivable from a normalized co-occurrence matrix, $\mathbf{N}_d$

$$Energy = \sum_i \sum_j N_d^2[i,j]$$

$$Entropy = -\sum_i \sum_j N_d[i,j] \log_2 \left(N_d[i,j] \right)$$

$$Contrast = \sum_i \sum_j (i-j)^2 N_d[i,j]$$

$$Homogeneity = \sum_i \sum_j \frac{N_d[i,j]}{1+|i-j|}$$

$$Correlation = \frac{\sum_i \sum_j (i-\mu_i)(j-\mu_j) N_d[i,j]}{\sigma_i \sigma_j}$$

μ_i, μ_j are the means and σ_i, σ_j are the standard deviations of the row and column sums