Announcements

- HW 1 is due today
- HW 2 will be available later today or tomorrow
- See links on web page for reading on binary image processing (e-reserves)
- Midterm May 5

---

**Binary System Summary**

1. Acquire images and binarize (thresholding, color labels, etc.).
2. Possibly clean up image using morphological operators.
3. Determine regions (blobs) using connected component exploration
4. Compute position, area, and orientation of each blob using moments
5. Compute features that are rotation, scale, and translation invariant using Moments (e.g., Eigenvalues of normalized moments).

---

**Four & Eight Connectedness**

- **Four Connected**
- **Eight Connected**

---

**Recursive Labeling**

Connected Component Exploration

---

**Properties extracted from binary image**

- A tree showing containment of regions
- Properties of a region:
  1. Genus – number of holes
  2. Centroid
  3. Area
  4. Perimeter
  5. Moments (e.g., measure of elongation)
  6. Number of “extrema” (indentations, bulges)
  7. Skeleton
Moments

(related to moments of inertia)

$S = \{(x, y) | f(x, y) = 1\}$

Given a pair of non-negative integers $(j, k)$ the discrete $(j, k)^{th}$ moment of $S$ is:

$$M_{jk}(S) = \sum_{(x, y) \in S} x^j y^k$$

The order of the $M_{jk}$ moment is $j + k$.

$M_{jk} = \sum_{i=1}^{n} \sum_{j=1}^{m} B(x, y) x^i y^j$

- Fast way to implement computation over an by $m$ image or window
- One object

Central Moments

$S = \{(x, y) | f(x, y) = 1\}$

$\bar{x} = \frac{M_{01}(S)}{M_{00}(S)} \quad \bar{y} = \frac{M_{10}(S)}{M_{00}(S)}$

Given a pair of non-negative integers $(j, k)$ the central $(j, k)^{th}$ moment of $S$ is given by:

$$\mu_{jk}(S) = \sum_{(x, y) \in S} (x - \bar{x})^j (y - \bar{y})^k$$

Or the central moments can be computed from precomputed regular moments

$$\mu_{jk} = \sum_{i=1}^{n} \sum_{j=1}^{m} B(x, y) (-x)^i (-y)^j M_{ij}$$

Normalized Moments

$S = \{(x, y) | f(x, y) = 1\}$

$$\mu_{jk}(S) = \sum_{(x, y) \in S} (x - \bar{x})^j (y - \bar{y})^k$$

$\sigma_x = \sqrt{\frac{\mu_{20}(S)}{M_{00}(S)}} \quad \sigma_y = \sqrt{\frac{\mu_{02}(S)}{M_{00}(S)}}$

Given a pair of non-negative integers $(j, k)$ the normalized $(j, k)^{th}$ moment of $S$ is given by:

$$m_{jk}(S) = \sum_{(x, y) \in S} \frac{(x - \bar{x})^j (y - \bar{y})^k}{\sigma_x \sigma_y}$$

Normalized moments are scale and translation invariant.

Moments

- Regular Moments $M_{jk}$
- Central Moments $\mu_{jk}$: Translation invariant
- Normalized Moments $m_{jk}$: Translation and scale Invariant
- Eigenvalues of Second Moment Matrix: translation, scale, and rotation invariant.
- Hu Moments: Higher than second order, translation, rotation and scale invariant

What is image filtering?

- Modify the pixels in an image based on some function of a local neighborhood of the pixels.

(From Bill Freeman)
Smoothing by Averaging

Kernel:

General process:
- Form new image whose pixels are a weighted sum of original pixel values, using the same set of weights at each point.

Properties:
- Output is a linear function of the input
- Output is a shift-invariant function of the input (i.e. shift the input image two pixels to the left, the output is shifted two pixels to the left)

Example: smoothing by averaging
- Form the average of pixels in a neighbourhood

Example: smoothing with a Gaussian
- Form a weighted average of pixels in a neighbourhood

Example: finding a derivative
- Form a weighted average of pixels in a neighbourhood

Convolution

Image (I) * Kernel (K)

Note: Typically Kernel is relatively small in vision applications.

Convolution: R = K*I

Kernel size is m+1 by m+1

\[ R(i, j) = \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} K(h, k) I(i-h, j-k) \]
Convolution: \( R = K \ast I \)

Kernel size is \( m+1 \) by \( m+1 \)

\[
R(i, j) = \sum_{h=-m/2}^{m/2} \sum_{k=-m/2}^{m/2} K(h, k) I(i-h, j-k)
\]
Convolution: \( R = K \ast I \)

Kernel size is \( m+1 \) by \( m+1 \)

\[ R(i, j) = \sum_{h=1}^{m} \sum_{k=1}^{m} K(h, k) I(i-h, j-k) \]

Impulse Response

Linear filtering (warm-up slide)

Kernal (1-D)

Kernal (2-D)

Linear filtering (warm-up slide)

Filtered (no change)
**Shift**

- Original image
- Pixel offset
- Shifted image

**Linear Filtering**

- Original image
- Pixel offset
- Filtered image

**Blurring**

- Original image
- Pixel offset
- Blurred image

Blurred (filter applied in both dimensions).

**Blur Examples**

- Impulse original
- Filtered
- Edge original
- Filtered

**Linear Filtering (Warm-up Slide)**

- Original image
- Filtered image
- Pixel offset

- 2.0
- 1.0
- ?
Linear filtering (no change)

(remember blurring)

Sharpening example

Sharpening
Smoothing by Averaging

Kernel: [image of a kernel]

Filters are templates

- Applying a filter at some point can be seen as taking a dot-product between the image and some vector.
- Filtering the image is a set of dot products.

Insight

- Filters look like the effects they are intended to find.
- Filters find effects they look like.

Properties of convolution

Let \( f, g, h \) be images and \( \ast \) denote convolution.

\[
(f \ast g)(x, y) = \int \int f(x-u, y-v)g(u, v) \, du \, dv
\]

- Commutative: \( f \ast g = g \ast f \)
- Associative: \( f \ast (g \ast h) = (f \ast g) \ast h \)
- Linear: for scalars \( a \) & \( b \) and images \( f, g, h \)
  \[ (af + bg) \ast h = a(f \ast h) + b(g \ast h) \]
- Differentiation rule

\[
\frac{\partial}{\partial x} (f \ast g) = \frac{\partial f}{\partial x} \ast g = f \ast \frac{\partial g}{\partial x}
\]

Filtering to reduce noise

- Noise is what we’re not interested in.
  - We’ll discuss simple, low-level noise today: Light fluctuations; Sensor noise; Quantization effects; Finite precision.
  - Not complex: shadows; extraneous objects.
- A pixel’s neighborhood contains information about its intensity.
- Averaging noise reduces its effect.

Additive noise

- \( I = S + N \). Noise doesn’t depend on signal.
- We’ll consider:

\[
I_i = s_i + n_i \text{ with } E(n_i) = 0
\]

\( s_i \) deterministic, \( n_i \) a random var.

\( n_i, n_j \) independent for \( i \neq j \)

\( n_i, n_j \) identically distributed

Gaussian Noise: \( \text{sigma}=1 \)

Gaussian Noise: \( \text{sigma}=16 \)
Average Filter

- Mask with positive entries, that sum 1.
- Replaces each pixel with an average of its neighborhood.
- If all weights are equal, it is called a BOX filter.

![Average Filter](image)

(Camps)

Smoothing by Averaging

Kernel: [image]

An Isotropic Gaussian

- The picture shows a smoothing kernel proportional to
  \[ \exp \left( \frac{x^2}{2\sigma^2} \right) \]

(which is a reasonable model of a circularly symmetric fuzzy blob)

![An Isotropic Gaussian](image)

Smoothing with a Gaussian

Kernel: [image]

Efficient Implementations

Both, the BOX filter and the Gaussian filter are separable:
- First convolve each row with a 1-D filter
- Then convolve each column with a 1-D filter.

For Gaussian kernels \( g_1(x) \) and \( g_2(x) \),
- If \( g_1 \) & \( g_2 \) respectively have variance \( \sigma_1^2 \) & \( \sigma_2^2 \)
- Then \( g_1 \ast g_2 \) has variance \( \sigma_1^2 + \sigma_2^2 \)
Other Types of Noise

- Impulsive noise
  - randomly pick a pixel and randomly set to a value
  - saturated version is called salt and pepper noise

- Quantization effects
  - Often called noise although it is not statistical

- Unanticipated image structures
  - Also often called noise although it is a real repeatable signal.

Some other useful filtering techniques

- Median filter
- Anisotropic diffusion

Median filters: Principle

Method:
1. rank-order neighborhood intensities in a window
2. take middle value

- non-linear filter
- no new grey levels emerge...

Median filters: Example for window size of 3

1,1,1,7,1,1,1,1

? . . . . . . .

Advantage of this type of filter is that it eliminates spikes (salt & pepper noise).

Median filters: example

filters have width 5:

<table>
<thead>
<tr>
<th></th>
<th>INPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MEDIAN</td>
</tr>
<tr>
<td></td>
<td>MEAN</td>
</tr>
</tbody>
</table>

Median filters: analysis

- median completely discards the spike, linear filter always responds to all aspects
- median filter preserves discontinuities, linear filter produces rounding-off effects

DON’T become all too optimistic
Median filter: images
3 x 3 median filter:

- sharpens edges, destroys edge cusps and protrusions

Median filters: Gauss revisited
Comparison with Gaussian:

- e.g. upper lip smoother, eye better preserved

Example of median
10 times 3 X 3 median

- patchy effect
- important details lost (e.g. ear-ring)