Announcements

- See links on web page for reading on binary image processing (e-reserves)
- Reading on filtering is in text
- Next homework will be posted later today – I think…

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 orientation invariant using Moments (e.g., Eigenvalues of normalized moments).

P-Tile Method

- If the size of the object is approx. known, pick $T$ s.t. the area under the histogram corresponds to the size of the object:

Recursive Labeling

Connected Component Exploration

Four & Eight Connectedness

Four Connected
Eight Connected
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)

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

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

- Fast way to implement computation over \(n\) by \(m\) image or window
- One object

Area: Moment \(M_{00}\)

\[
S = \{(x, y)|f(x, y) = 1\}
M_{00}(S) = \sum_{(x,y) \in S} x^0 y^0 = \sum_{(x,y) \in S} 1 = \#(S)
\]

Example:

Area of \(S\)!

Computing the centroid with Moments

\[
M_{10}(S) = \sum_{(x,y) \in S} x = \frac{\sum_{(x,y) \in S} x}{\#(S)} = \overline{x}
M_{01}(S) = \sum_{(x,y) \in S} y = \frac{\sum_{(x,y) \in S} y}{\#(S)} = \overline{y}
\]

Center of gravity (Centroid) of \(S\)!

Shape recognition by Moments

Recognition could be done by comparing moments

However, moments \(M_{jk}\) are not invariant under:
- Translation
- Scaling
- Rotation
- Skewing

Central Moments

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

\[
\mu_{jk}(S) = \sum_{(x,y) \in S} (x - \overline{x})^j (y - \overline{y})^k
\]
Central Moments
\[ S = \{(x, y)| f(x, y) = 1\} \]
\[ \mu_{jk}(S) = \sum_{(x,y)\in S} (x - \bar{x})^j(y - \bar{y})^k \]
Translation by \( T = (a,b) \):
\[ S_T = \{(x', y')| x' = x + a, y' = y + b, (x, y) \in S\} \]
\[ \bar{x}' = \frac{M_{20}(S_T)}{M_{02}(S_T)} \bar{x} + a \quad \bar{y}' = \frac{M_{02}(S_T)}{M_{02}(S_T)} \bar{y} + b \]
\[ m_{jk}(S_T) = m_{jk}(S) \]
Translation INVARIANT!

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:
\[ m_{jk}(S) = \sum_{(x,y)\in S} \left( \frac{x - \bar{x}}{\sigma_x} \right)^j \left( \frac{y - \bar{y}}{\sigma_y} \right)^k \]

Region orientation from Second Moment Matrix
1. Compute second centralized moment matrix
\[ \begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix} \]
2. Compute Eigenvectors of Moment Matrix to obtain orientation
3. Eigenvalues are independent of orientation, translation!

Binarization using Color
- Object’s in robocup are distinguished by color.
- How do you binarize the image so that pixels where ball is located are labeled with 1, and other locations are 0?
- Let \( C_b = (r,g,b)^T \) be the color of the ball.
- Let \( c(u,v) \) be the color of pixel \((u,v)\)
- Simple method
\[ b(u,v) = \begin{cases} 1 & \text{if } ||c(u,v) - c_b||^2 \leq \varepsilon \\ 0 & \text{otherwise} \end{cases} \]
- Better alternative (why?)
  - Convert \( c(u,v) \) to HSV space \( H(u,v), S(u,v), V(u,v) \)
  - Convert \( c_b \) to HSV
  - Check that HS distance is less than threshold \( \varepsilon \) and brightness is greater than a threshold \( V > \tau \)
Blob Tracking

Main tracking notions

• State: usually a finite number of parameters (a vector) that characterizes the “state” (e.g., location, size, moments, pose) of thing being tracked. (e.g., Φ)
• Dynamics: How does the state change over time? How is that changed constrained? (e.g., dΦ/dt)
• Trajectory: Φ(t)
• Prediction: Given the state at time t-1, what is an estimate of the state at time t?
• Data Association: Given predicted state, and measurement of multiple blobs in image at time t, which blob is being tracked?

Other ideas

• Binarization of color images
• Blob Tracking
  – Binary regions
  – State (e.g., x,y,orientation, scale, etc.)
  – Prediction
  – Data Association

Linear Filters

• 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)

Convolution

• 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

What is image filtering?

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

(From Bill Freeman)

Note: Typically Kernel is relatively small in vision applications.
Kernel size is \( m+1 \) by \( m+1 \) for \( m=2 \):

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

Kernel size is $m+1$ by $m+1$

$R(i,j) = \sum_{m=-2}^{m+2} \sum_{k=-2}^{m+2} K(h,k)I(i-h, j-k)$

Impulse Response
Linear filtering (warm-up slide)

Original

Kernel

Coefficient

Pixel offset

Original

Filtered (no change)

Linear filtering

Original

Shifted

Linear filtering

Original

Blurred (filter applied in both dimensions)
Blur examples

- impulse
- coefficient: 2.4
- Pixel offset
- original
- filtered

Linear filtering (warm-up slide)

- 2.0
- 1.0
- ?
- original
- Filtered (no change)

Linear filtering

- coefficient: 0.13
- original
- Blurred (filter applied in both dimensions)
Properties of convolution

Let $f, g, h$ be images and $*$ denote convolution

$$f * g(x, y) = \iint f(x-u,y-v)g(u,v) \, du \, dv$$

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

$$\frac{\partial}{\partial x}(f * g) = \frac{\partial f}{\partial x} * g = f * \frac{\partial g}{\partial x}$$

Smoothing by Averaging

Kernel

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.

Noise

- Issues
  - this model allows noise values that could be greater than maximum camera output or less than zero
  - for small standard deviations, this isn’t too much of a problem - it’s a fairly good model
  - independence may not be justified (e.g. damage to lens)
  - may not be stationary (e.g. thermal gradients in the ccd)
**Gaussian Noise:**
sigma=1

**Gaussian Noise:**
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.

(Camps)

\[
\begin{bmatrix}
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
\end{bmatrix}
\]

**An Isotropic Gaussian**
- The picture shows a smoothing kernel proportional to
  \[
  \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)
  \]
  (which is a reasonable model of a circularly symmetric fuzzy blob)

**Smoothing by Averaging**

**Smoothing with a Gaussian**

Kernel: \[
\begin{bmatrix}
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
\end{bmatrix}
\]

**The effects of smoothing**
Each row shows smoothing with gaussians of different width; each column shows different realizations of an image of gaussian noise.
Efficient Implementation

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 $g_1 * g_2$ has variance $\sigma_1^2 + \sigma_2^2$