5. Feature Extraction from Images
Aim of this Chapter: Learn the Basic Feature Extraction Methods for Images

Main features:
• Color
• Texture
• Edges
Wie funktioniert ein Mustererkennungssystem

Test Data $x_i$

Feature Extraction

Classifier

Model

Training Data

Training Algorithm

$\omega_1$

$\omega_2$

$\omega_n$

$\ldots$
5.1. Histograms and Color Features
Color Histogram

Calculate percentage of color present in image

Deficiency: loss of regional information
Measurements at Pixels

• An image I is a set of pixels
• At each pixel:
  – Measure some m-dimensional property

Example:
  Each pixel of an RGB image is a 3-dimensional vector

Formally:

\[ f_I : \mathbb{R}^2 \subset \mathbb{R}^m \rightarrow M \subset \mathbb{R}^m \]
Create a finite Partition of M

Create finite partition of M:

\[ M = \bigcup_{k=1}^{K} B_k \]

\( B_k \) are subsets of M

\( B_k \) are called bins and \( k \) is the label of the bin
Example:

Let $M$ be the grey levels of an image $M = [0 : 255]$

<table>
<thead>
<tr>
<th>Label of bin</th>
<th>Gray levels $B_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-31</td>
</tr>
<tr>
<td>2</td>
<td>32-63</td>
</tr>
<tr>
<td>3</td>
<td>64-95</td>
</tr>
<tr>
<td>4</td>
<td>96-127</td>
</tr>
<tr>
<td>5</td>
<td>128-159</td>
</tr>
<tr>
<td>6</td>
<td>160-191</td>
</tr>
<tr>
<td>7</td>
<td>192-213</td>
</tr>
<tr>
<td>8</td>
<td>214-255</td>
</tr>
</tbody>
</table>
From Bins to Histograms

Indicator function:

\[ b_k(x) = \begin{cases} 
1 & \text{if } f_I(x) \in B_k \\
0 & \text{otherwise} 
\end{cases} \]

x is element of the image
A histogram is a vector

\[ \vec{H} = (h_1, \ldots, h_K) \]

with

\[ h_k = \frac{\int_{x \in R} b_k(f_I(x)) \, dx}{\int_{x \in R} 1 \, dx} \]

In words:

for each bin:

count which fraction of pixels falls into that bin.
Histogram Distances

Motivation:
measures the similarity of
- images
- speech
- music

Issue:
how to capture perceptual similarity
Histogram Distances

$L_1$ distance (Manhattan distance)

$$d_1(H, L) = \sum_{k=1}^{K} |h_k - l_k|$$

$L_2$ distance (Euclidean distance)

$$d_2(H, L) = \sqrt{\sum_{k=1}^{K} (h_k - l_k)^2}$$

$L_\infty$ distance (maximum distance)

$$d_\infty(H, L) = \max_k (|h_k - l_k|)$$
Exercise
Example for potential problem with histogram distance
Example for potential problem with histogram distance

(a) \[ \frac{1}{3} \]

(b) \[ \frac{1}{3} \]

(c) \[ \frac{1}{3} \]
Distances of the three checkerboard images

<table>
<thead>
<tr>
<th>Distance type</th>
<th>d(a,b)</th>
<th>d(a,c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L₁</td>
<td>2</td>
<td>~0.67</td>
</tr>
<tr>
<td>L₂</td>
<td>~0.82</td>
<td>~0.47</td>
</tr>
<tr>
<td>L∞</td>
<td>~0.33</td>
<td>~0.33</td>
</tr>
</tbody>
</table>

None of the distances captures perceptual similarity
Realistic example for problem with distances
Realistic example for problem with distances

Creating distance measures that capture the human notion of similarity is difficult.
Potential problem with histogram distance

• There are alternative distance measures
• Details beyond the scope of this lecture
• If you seem to have such a problem: look into the literature
Issue: loss of regional information

Partition the image
One histogram per region
5.3. Texture Features
What’s in the image?
What is texture?

- Texture has no precise definition.
- Texture is a tactile or visual characteristic of a surface.
- Texture primitives (or texture elements, texels) are building blocks of a texture.
- Texel: A small geometric pattern that is repeated frequently on some surface resulting in a texture.
Use of Texture Analysis

- Segment an image into regions with the same texture, i.e. as a complement to gray level or color
- Recognize or classify objects based on their texture
- Find edges in an image, i.e. where the texture changes
- "shape from texture"
- Object detection, compression, synthesis
Difficulties of Texture Analysis

• Which scale to use?
Texture Analysis

- Generic research area of machine vision
- Topic of research for over three decades
- Aim: to find a unique way of representing the underlying characteristics of textures and represent them in some simpler but unique form, so then they can be used to accurately and robustly classify and segment objects.
Types of Texture

• **Strong Texture**
  – spatial interactions between primitives are somewhat regular
  – frequency of occurrence of primitive pairs in some spatial relationship used for description

• **Weak Texture**
  – small spatial interactions between primitives
  – frequencies of primitive types appearing in some neighborhood used for description

• **Two basic texture description approaches:**
  – syntactic
  – statistical
Syntactic texture description

• Not used as widely as statistical approach

• Analogy between texture spatial relationships and structure of a formal language.

• Grammar representation - primitives are terminal symbols, relationships are represented as transformation rules.
Syntactic Approaches

- **Shape Grammars**
  - $G = \langle T, N, R, S \rangle$
  - $T$ : Primitive shapes
  - $R$ : Rules showing how elements can be composed
  - $S$ : Start symbol
  - $N$ : Non terminals
Example of Shape Grammar

\[ T = \{ \square \} \quad S = \{ \circ \} \quad N = \{ . \} \]

Need a few additional rules for completion.
Comments

• Syntactic approaches are often not practical
  • Difficult to model
  • Difficult to compute

• Natural textures are complex!!!
First Order Statistics

- Mean \[ \mu = \sum_{k=1}^{K} k p_k \]
- Variance \[ \sigma^2 = \sum_{k=1}^{K} (k - \mu)^2 p_k \]
- Skewness \[ \gamma_3 = \frac{1}{\sigma^3} \sum_{k=1}^{K} (k - \mu)^3 p_k \]
- Kurtosis \[ \gamma_4 = \frac{1}{\sigma^4} \sum_{k=1}^{K} (k - \mu)^4 p_k - 3 \]

(hardly a useful feature)

Why?

\[ p_k = \frac{h_k}{\sum_{k=1}^{K} h_k} \]
Example for first Order Statistics

**Granite texture**
Mean 157.08  
Variance 96.9573  
Skewness -0.73  
Kurtosis 3.25

**Brickwall texture**
Mean 79.13  
Variance 42.73  
Skewness 1.37  
Kurtosis 5.93
Autocorrelation Function

\[ \rho_{ff}(i, j) = \sum_{x} \sum_{y} f(x, y) f(x+i, y+j) \]

- What is the frequency of repetition of structures: coarse/fine texture
- How strongly are they correlated: Similarity of the texels
Autocorrelation: Example
Autocorrelation: Example
Gabor Filter

- Family of filters
  - Product of Gaussian with traveling waves
Gabor Filter

Base filter

\[ G_{00}(x, y) = \frac{1}{2\pi \sigma_x \sigma_y} \exp \left[ -\frac{1}{2} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) \right] \cos(2\pi \omega x) \]

Bank of Gabor Filters

\[ G_{mn}(x, y) = a^{-m} G_{00}(x', y') \]

with

\[
\begin{pmatrix} x' \\ y' \end{pmatrix} = a^{-m} \begin{pmatrix} \cos \theta_n & \sin \theta_n \\ -\sin \theta_n & \cos \theta_n \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}
\]

and \( \theta_n = n\pi / K \)
Position of Frequencies for a Set of Gabor Filters

Maple script
Gabor Filter

For each filter of the set, a filtered image is calculated

\[ w_{mn}(x, y) = \sum_{a,b} G_{mn}(x-a, y-b)I(a,b) \]

Features for image processing:
  Mean and variance of the filtered images
  \[ w_{mn}(x,y) \]
Feature Extraction
A look at gabor filter extraction for a 0 degree filter

From: http://tdil.mit.gov.in/rfp/OCR/OPTICAL%20CHARACTER%20RECOGNITION.ppt
Gabor filters, a 2D example

5.2. Edge Information
Characterizing Edges

- Images are *discrete* functions indicating the light intensity of a scene
- What happens at an edge?
Characterizing Edges (cont’d)

• Let’s look at one line for now
Detecting Edges

- Edges correspond to large changes in the image
- How do we detect such changes?
Gradient Definition

\[ \nabla I(x, y) = \frac{\partial I}{\partial x} \hat{x} + \frac{\partial I}{\partial y} \hat{y} \]

- The gradient is a vector with magnitude in the \( u \) and \( v \) directions equal to the respective partial derivatives.
- How do we compute the partial derivative of a discrete function?
Taylor Series…

\[ f(x+h) = f(x) + hf'(x) + \frac{1}{2} h^2 f''(x) + ... \]

or…

\[ f(x-h) = f(x) - hf'(x) + \frac{1}{2} h^2 f''(x) + ... \]

- Subtracting the second from the first we obtain

\[ f'(x) = \frac{f(x+h) - f(x-h)}{2h} + O(h^2) \]
Discrete Gradient Estimation

- Discrete functions: use first order approximation of the gradient
  \[ f'(x) \approx \frac{f(x + h) - f(x - h)}{2h} \]
  \( h \) corresponds to the step size

- Images: \( h \) corresponds to the width of 1 pixel
  \[
  \frac{\partial I(x, y)}{\partial x} = \frac{I(x + 1, y) - I(x - 1, y)}{2}
  \]
  \[
  \frac{\partial I(x, y)}{\partial y} = \frac{I(x, y + 1) - I(x, y - 1)}{2}
  \]
Discrete Gradient as a Linear Filter

- Gradient can be written as a linear filter
- Drop factor 2 because it just scales the image

\[
\frac{\partial I}{\partial x} = I * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}
\]

\[
\frac{\partial I}{\partial y} = I * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}
\]
Taking the discrete derivative

\[
\begin{bmatrix}
-1 & 0 & 1
\end{bmatrix}
\]

abs()
Basic Edge Detection Step 1

INPUT IMAGE

$I(x, y)$

1) Edge Enhancement

Horizontal

$[-1 \ 0 \ 1]$  

Vertical

$[-1 \ 0 \ 1]^T$

$\frac{\partial I(x, y)}{\partial x}$

$\frac{\partial I(x, y)}{\partial y}$

Issue: sensitivity to noise?
Basic Edge Detection Steps 1-2

1) Noise Smoothing

\[ I(x, y) \]

INPUT IMAGE

\[
\begin{bmatrix}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{bmatrix}
\]

\( /16 \)

2) Edge Enhancement

Horizontal

\[
[-1 0 1]
\]

\[
\frac{\partial I(x, y)}{\partial x}
\]

Vertical

\[
[-1 0 1]^T
\]

\[
\frac{\partial I(x, y)}{\partial y}
\]
Discrete Gradient Estimation

- Gradient is a vector
- we have calculated the coefficients in the $x$ and $y$ directions at each point in the image
- After convolving, we get the magnitude of the gradient from at each point (pixel) from

$$G(x, y) = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$

- In practice, we often sum the absolute values of the components for computational efficiency
Basic Edge Detection (cont’d)

**INPUT IMAGE**

\[ I(x, y) \]

1) Noise Smoothing

\[
\begin{bmatrix}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1
\end{bmatrix} / 16
\]

2) Edge Enhancement

Horizontal \([-1 \ 0 \ 1]\)

Vertical \([-1 \ 0 \ 1]^T\)

\[ \frac{\partial I(x, y)}{\partial x} \]

\[ \frac{\partial I(x, y)}{\partial y} \]

\[ |\partial I(x, y)| = \left[ \frac{\partial I(x, y)^2}{\partial x} + \frac{\partial I(x, y)^2}{\partial y} \right]^{\frac{1}{2}} \]

“GRADIENT” IMAGE
Thresholding

- Remove lighting effects
- Convert to binary image using a threshold

Results from threshold values of 50 and 100
Basic Edge Detection Summary

\[ I(x, y) \]

**INPUT IMAGE**

1) Noise Smoothing

\[
\begin{bmatrix}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1
\end{bmatrix}
\]

\( /16 \)

2) Edge Enhancement

**Horizontal**

\([ -1 \ 0 \ 1 ] \)

**Vertical**

\([ -1 \ 0 \ 1 ]^T \)

\[ \frac{\partial I(x, y)}{\partial x} \]

\[ \frac{\partial I(x, y)}{\partial y} \]

3) Threshold

\[ | \frac{\partial I(x, y)}{\partial x} | = \left[ \left( \frac{\partial I(x, y)}{\partial x} \right)^2 + \left( \frac{\partial I(x, y)}{\partial y} \right)^2 \right]^{\frac{1}{2}} \]

**EDGE IMAGE**

**"GRADIENT" IMAGE**
The effects of Filtering Noise

Threshold 20

Threshold 50

Unsmoothed Edges

Gaussian Smoothing
Sobel Edge Detection

- Integrate smoothing and gradient calculation
- Sobel operators: widely spread scheme

\[ Sobel_V = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad Sobel_H = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \]

- Convolving generates horizontal and vertical gradient images
Other Edge Detectors

• *Prewitt*: similar to the Sobel, but different kernel

\[
P_V = \begin{bmatrix}
-1 & 0 & 1 \\
-1 & 0 & 1 \\
-1 & 0 & 1 \\
\end{bmatrix}, \quad P_H = \begin{bmatrix}
1 & 1 & 1 \\
0 & 0 & 0 \\
-1 & -1 & -1 \\
\end{bmatrix}
\]

• Canny edge detector
Other Edge Detectors

- **Roberts**: early edge detector kernel

\[
R_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad R_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}
\]

- Very sensitive to noise
- Very fast
Summary

Color features:
use histograms
issue: robust distances measures

Texture:
first order statistics
auto correlation function
Gabor filter
Summary

-Edges correspond to abrupt changes in image intensity

-Edges can be detected by
  - Smoothing out image noise
  - Estimating the gradient of the image at every point to generate a “gradient” image
  - Thresholding the gradient image