Image Enhancement

- **Enhancement**: to process an image so that the result is more suitable than the original image for a specific application.
- **Enhancement approaches**:
  1. Spatial domain
  2. Frequency domain

Spatial Domain: Background

- \( g(x,y) = T[f(x,y)] \)
  - \( f(x,y) \): input image, \( g(x,y) \): processed image
  - \( T \): an operator

Spatial Domain: Point Processing

- \( s = T(r) \)
  - \( r \): gray-level at \((x,y)\) in original image \(f(x,y)\)
  - \( s \): gray-level at \((x,y)\) in processed image \(g(x,y)\)
  - \( T \) is called gray-level transformation or mapping

Spatial Domain: Point Processing

- **Contrast Stretching**: to get an image with higher contrast than the original image
  - The gray levels below \( m \) are darkened and the levels above \( m \) are brightened.
Spatial Domain: Point Processing

**Contrast Stretching**

Original | Enhanced

Exercise

- A 4x4 image is given as follow.
  1) The image is transformed using the point transform shown. Find the pixel values of the output image.

<table>
<thead>
<tr>
<th>17</th>
<th>64</th>
<th>128</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>63</td>
<td>132</td>
<td>133</td>
</tr>
<tr>
<td>11</td>
<td>60</td>
<td>142</td>
<td>140</td>
</tr>
<tr>
<td>11</td>
<td>60</td>
<td>142</td>
<td>138</td>
</tr>
</tbody>
</table>

| 0   | 20  | 130 | r   |

Gray-level Transforms

- Limiting case: produces a binary image (two level) from the input image

\[ s = \begin{cases} T(r) & \text{if } r \geq m \\ 255 - T(r) & \text{if } r < m \end{cases} \]

Thresholding

Original | Enhanced

Image Negative

- Suited for enhancing white detail embedded in dark regions.
- Has applications in medical imaging.
Log Transformation

\[ s = c \log(1 + r) \]

- Log transformation: maps a narrow range of low gray-level input image into a wider range of output levels.
- Expand the values of dark pixels in an image while compressing the higher-level values.

Power-law Transformation

\[ S = CR^{\gamma} \]

- If \( \gamma < 1 \): maps a narrow range of dark input values into a wider range of output values.
- If \( \gamma > 1 \): opposite of the above effect.
- The process used to correct this power-law response phenomena is called gamma correction.
Power-law Transformation

\[
s(s) = \begin{cases} 
\alpha r, & 0 \leq r \leq r_1 \\
\beta (r - r_1) + s_1, & r_1 \leq r \leq r_2 \\
\gamma (r - r_2) + s_2, & r_2 \leq r \leq L - 1 
\end{cases}
\]

Contrast Stretching

Gray-level Slicing

- Highlights a specific range of gray-levels in an image

2 basic methods:
1. Display a high value for all gray levels in the range of interest and a low value for all other
2. Brighten the desired range of gray levels but preserve the gray level tonalities

Piecewise-Linear Transform

Bit-plane Slicing

One 8-bit pixel value

Bit plane 7
(most significant)

Bit plane 0
(least significant)

Original Image
Bit-plane Slicing

• Higher order bit planes of an image carry a significant amount of visually relevant details.

• Lower order planes contribute more to fine (often imperceptible) details.

Exercise

• A 4x4 image is given as follow.
  1) The image is transformed using the point transform shown. Find the pixel values of the output image.
  2) What is the 7-th bit plane of this image

| 17 | 64 | 128 | 128 |
| 15 | 63 | 132 | 133 |
| 11 | 60 | 142 | 140 |
| 11 | 60 | 142 | 138 |

Bit-plane Slicing

Histogram Processing

• Histogram is a discrete function formed by counting the number of pixels that have a certain gray level in the image.

• In an image with gray levels in \([0, L-1]\), the histogram is given by \(p(r_k) = n_k/n\) where:
  – \(r_k\) is the \(k\) th gray level, \(k=0, 1, 2, \ldots, L-1\)
  – \(n_k\) number of pixels in the image with gray level \(r_k\)
  – \(n\) total number of pixels in the image

• Loosely speaking, \(p(r_k)\) gives an estimate of the probability of occurrence of gray level \(r_k\).

Histogram Processing

• Problem: an image with gray levels between 0 and 7 is given below. Find the histogram of the image

| 1 | 6 | 2 | 2 |
| 1 | 3 | 3 | 3 |
| 4 | 6 | 4 | 0 |
| 1 | 6 | 4 | 7 |

| 0: 1/16 | 4: 3/16 |
| 1: 3/16 | 5: 0/16 |
| 2: 2/16 | 6: 3/16 |
| 3: 3/16 | 7: 1/16 |
Histogram Equalization

- Goal: find a transform $s=T(r)$ such that the transformed image has a flat (equalized) histogram.
- A) $T(r)$ is signle-valued and monotonically increasing in interval $[0,1]$;
- B) $0 \leq T(r) \leq 1$ for $0 \leq r \leq 1$.

$$p_k(r_k) = \frac{n_k}{n} \quad k = 0,1,2,\ldots,L-1$$

$$s_i = T(r_i) = \sum_{j=0}^{k} p_j(r_j) = \sum_{j=0}^{k} \frac{n_j}{n}$$

Local Histogram Processing

- Transformation should be based on gray-level distribution in the neighborhood of every pixel.
- Local histogram processing:
  - At each location the histogram of the points in the neighborhood is computed and a histogram equalization or histogram specification transformation function is obtained
  - The gray level of the pixel centered in the neighborhood is mapped
  - The center of the neighborhood is moved the next pixel and the procedure repeated
Local Enhancement

• Mean of gray levels in an image: a measure of darkness, brightness of the image.
  \[ m = \sum_{i=0}^{L-1} r_i p(r_i) \]
  \[ \sigma^2 = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i) \]

• Variance of gray levels in an image: a measure of average contrast.

Local Enhancement

• Local mean and variance
  \[ m_{xy} = \sum_{(x',y') \in S} r_{xy} p(r_{xy}) \]
  \[ \sigma_{xy}^2 = \sum_{(x',y') \in S} (r_{xy} - m_{xy})^2 p(r_{xy}) \]

• Local Enhancement
  \[ g(x,y) = \begin{cases} 
    E \cdot f(x,y) & m_{xy} \leq k_1 M_G, k_1 D_G \leq \sigma_{xy} \leq k_2 D_G \\
    f(x,y) & \text{otherwise}
  \end{cases} \]

Local Enhancement

• Local Enhancement Using Arithmetic/Logic Operations
  - Arithmetic/Logic operations are performed on the pixels of two or more images.
  - Arithmetic: p and q are the pixel values at location (x,y) in first and second images respectively
    - Addition: p+q
    - Subtraction: p-q
    - Multiplication: p.q
    - Division: p/q

Enhancement Using Arithmetic/Logic Operations
Logic Operations

- When dealing with logic operations on gray-level images, pixel values are processed as strings of binary numbers.
- AND, OR, COMPLEMENT (NOT)

\[
\begin{align*}
\text{AND} & : 0 \land 0 = 0; 0 \land 1 = 0; 1 \land 0 = 0; 1 \land 1 = 1 \\
\text{OR} & : 0 \lor 0 = 0; 0 \lor 1 = 1; 1 \lor 0 = 1; 1 \lor 1 = 1 \\
\text{NOT} & : \neg 0 = 1; \neg 1 = 0
\end{align*}
\]

Image Subtraction

\[ g(x,y) = f(x,y) - h(x,y) \]

- Example: imaging blood vessels and arteries in a body. Blood stream is injected with a dye and X-ray images are taken before and after the injection
  - \( f(x,y) \): image after injecting a dye
  - \( h(x,y) \): image before injecting the dye
- The difference of the 2 images yields a clear display of the blood flow paths.

Image Averaging

\[ K \text{ noisy observation images} \]

\[ g(x,y) = f(x,y) + \eta(x,y) \]

Averaging

\[ \bar{g}(x,y) = \frac{1}{K} \sum_{k=1}^{K} g(x,y) \]

We have

\[ E\{g(x,y)\} = f(x,y) \quad \sigma^2_{g(x,y)} = \frac{1}{M} \sigma^2_{\eta(x,y)} \]

Spatial Filtering

- Linear filters
  - Average filtering
  - Weighted averaging
- Non-linear filters
  - Order-statistics filters
  - Median filters
  - High-boost filters
  - Derivative filters
  - Smoothing filters
  - Sharpening filters
Spatial Filtering

\[ w(s, t) : (2a + 1) \times (2b + 1) \text{ mask} \]

Filtering by using mask \( w(s, t) \): convolution

\[ g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t) \]

**Example 1**

![Example 1](image1)

**Example 2**

![Example 2](image2)

**Median Filtering**

- Median filtering is particularly effective in the presence of impulse noise (salt and pepper noise).
- Unlike average filtering, median filtering does not blur too much image details.
- **Example:** Consider the example of filtering the sequence below using a 3-pt median filter:
  
  \[ 16 \ 14 \ 15 \ 12 \ 2 \ 13 \ 15 \ 52 \ 51 \ 50 \ 49 \]
  
  - The output of the median filter is:
    
    \[ 15 \ 14 \ 14 \ 12 \ 12 \ 13 \ 15 \ 51 \ 51 \ 50 \ 50 \]
Median Filtering

- Advantages:
  - Removes impulsive noise
  - Preserves edges

- Disadvantages:
  - Poor performance when # of noise pixels in the window is greater than 1/2 # in the window
  - Poor performance with Gaussian noise

Sharpening Filters

- Objective: highlight fine detail in an image or to enhance detail that has been blurred.
- First and second order derivatives are commonly used for sharpening:
  \[
  \frac{df}{dx} = f(x+1) - f(x)
  \]
  \[
  \frac{d^2f}{dx^2} = f(x+1) + f(x-1) - 2f(x)
  \]

Sharpening Filters

1. First-order derivatives generally produce thicker edges in an image.
2. Second order derivatives have stronger responses to fine details such as thin lines and isolated points.
3. Second order derivatives produce a double response at step changes in gray level.
4. For image sharpening, second order derivative has more applications because of the ability to enhance fine details.
Laplacian

- The filter is expected to be isotropic: response of the filter is independent of the direction of discontinuities in an image.
- Simplest 2-D isotropic second order derivative is the Laplacian:

\[
\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}
\]

\[
\nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)
\]

Laplacian Enhancement

- Image background is removed by Laplacian filtering.
- Background can be recovered simply by adding original image to Laplacian output:

\[
g(x,y) = \begin{cases} 
   f(x,y) - \nabla^2 f(x,y) & \text{Laplacian mask center is negative} \\
   f(x,y) + \nabla^2 f(x,y) & \text{Laplacian mask center is positive}
\end{cases}
\]
Unsharp masking & High-boost filtering

- **Unsharp masking:** \( f'(x, y) = f(x, y) - \tilde{f}(x, y) \)
  
  \( \tilde{f}(x, y) \) : blurred version of original image

- **High-boost filtering:**
  
  \( f_{ab}(x, y) = Af(x, y) - f(x, y) \)
  
  \( f_{ab}(x, y) = (A-1)f(x, y) + f(x, y) - \tilde{f}(x, y) \)
  
  \( f_{ab}(x, y) = (A-1)f(x, y) + f(x, y) \)

Exercises

- A 4x4 image is given as follows.
  
  1) Suppose that we want to process this image by replacing each pixel by the difference between the pixels to the top and bottom. Give a 3x1 mask that performs this.
  
  2) Apply the mask to the second row of the image.
  
  3) Design a mask that can detect vertical edges, and process the following image with this mask.