BIM472 Image Processing

Processing in Spatial Domain
Part 2. Spatial Filtering

Outline

• Spatial Filtering Intro
• Spatial Correlation and Convolution
• Smoothing Spatial Filters
  – Linear Smoothing Filters
  – Nonlinear Smoothing Filters
• Sharpening Filters
  – Laplacian Operator
  – Unsharp Masking and Highboost Filtering
  – Using First-Order Derivatives for Nonlinear Image Sharpening - The Gradient
• Matlab Example
A spatial filter consists of
(a) a neighborhood
(b) a predefined operation

Linear spatial filtering of an image of size (M x N) with a filter of size (m x n) is given by the expression

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

\[ m = 2a + 1, \quad n = 2b + 1 \]
Spatial Correlation and Convolution

• Spatial filtering is actually a correlation or convolution process.

• Correlation is the process of moving a filter mask over the image and computing the sum of products at each location.

• The mechanics of convolution are the same, except that the filter is first rotated by 180 degree.

• Correlation and convolution are functions of displacements.

Spatial Correlation and Convolution

The correlation of a filter $w(x, y)$ of size $m \times n$ with an image $f(x, y)$, denoted as $w(x, y) \star f(x, y)$

$$w(x, y) \star f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)$$
The convolution of a filter $w(x, y)$ of size $m \times n$ with an image $f(x, y)$, denoted as $w(x, y) \star f(x, y)$

$$w(x, y) \star f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x-s, y-t)$$

---

**FIGURE 3.30**
Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.
Smoothing Spatial Filters

- Smoothing filters are used for blurring and for noise reduction.
- Blurring is used in removal of small details and bridging of small gaps in lines or curves.
- Smoothing spatial filters include linear filters and nonlinear filters.

Linear Smoothing Filters

The general implementation for filtering an $M \times N$ image with a weighted averaging filter of size $m \times n$ is given:

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

where $m = 2a + 1$, $n = 2b + 1$. 
Linear Smoothing Filters

Two Smoothing Averaging Filter Masks

\[
\begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\times \frac{1}{9}
\quad
\begin{array}{ccc}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{array}
\times \frac{1}{16}
\]

**FIGURE 3.32** Two 3 × 3 smoothing (averaging) filter masks. The constant multiplier in front of each mask is equal to the sum of the values of its coefficients, as is required to compute an average.

Linear Smoothing Filters

**FIGURE 3.33** (a) Original image, of size 500 × 500 pixels. (b-f) Results of smoothing with square averaging filter masks of sizes m = 3, 5, 9, 13, and 15, respectively. The black squares at the top are of sizes 3, 5, 8, 15, 25, 35, 45, and 55 pixels, respectively, their borders are 25 pixels apart. The letter at the bottom range is size from 10 to 20 pixels, in increments of 2 pixels; the large letter at the top is 60 pixels. The vertical bars are 7 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circle is 25 pixels, and their borders are 15 pixels apart; their intensity levels range from 0% to 100%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.
Linear Smoothing Filters

Example: Gross Representation of Objects

Nonlinear Smoothing Filters

- Nonlinear (order-statistic)
- Based on ordering (ranking) the pixels contained in the filter mask
- Replacing the value of the center pixel with the value determined by the ranking result (median, max, min, etc.)
- Median filter is the best known filter for this category. This filter is particularly effective in the presence of impulse noise (salt-and-pepper noise).
Nonlinear Smoothing Filters

Example:

• Suppose that a 3x3 neighborhood has values

\[
\begin{array}{ccc}
10 & 25 & 20 \\
20 & 100 & 20 \\
20 & 20 & 15 \\
\end{array}
\]

• Median of these values is 20 which is the middle value. 
  \((10, 15, 20, 20, 20, 20, 20, 25, 100)\)

• Principal function of median filters is to force points with distinct intensity levels to be more like their neighbors.

Nonlinear Smoothing Filters

Example: Use of Median Filtering for Noise Reduction

FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3 x 3 averaging mask. (c) Noise reduction with a 3 x 3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)
Sharpening Filters

- The principal objective of sharpening is to highlight transitions in intensity.
- Uses of image sharpening include applications ranging from electronic printing and medical imaging to industrial inspection and autonomous guidance in military systems.
- We know that blurring can be obtained by integration.
- So, we can state that sharpening can be obtained by differentiation.

Sharpening Filters

- Foundation
- Laplacian Operator
- Unsharp Masking and Highboost Filtering
- Using First-Order Derivatives for Nonlinear Image Sharpening — The Gradient
Sharpening Filters

Foundation

- The first-order derivative of a one-dimensional function \( f(x) \) is the difference

\[
\frac{\partial f}{\partial x} = f(x + 1) - f(x)
\]

- The second-order derivative of \( f(x) \) is the difference

\[
\frac{\partial^2 f}{\partial x^2} = f(x + 1) + f(x - 1) - 2f(x)
\]

FIGURE 3.36 Illustration of the first and second derivatives of a 1-D digital function representing a section of a horizontal intensity profile from an image. In (a) and (c) data points are joined by dashed lines as a visualization aid.
Sharpening Filters

Laplace Operator

- The second-order isotropic (rotation invariant) derivative operator is the Laplacian for a function (image) \( f(x,y) \)

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

\[ \frac{\partial^2 f}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y) \]

\[ \frac{\partial^2 f}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y) \]

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

Sharpening Filters

Figure 3.37

(a) Filter mask used to implement Eq. (3.6-6).
(b) Mask used to implement an extension of this equation that includes the diagonal terms.
(c) and (d) Two other implementations of the Laplacian found frequently in practice.
Sharpening Filters

Laplace Operator

Image sharpening in the way of using the Laplacian:

\[ g(x, y) = f(x, y) + c \left[ \nabla^2 f(x, y) \right] \]

where,

- \( f(x, y) \) is input image,
- \( g(x, y) \) is sharpened images,
- \( c = -1 \) if \( \nabla^2 f(x, y) \) corresponding to Fig. 3.37(a) or (b)
- and \( c = 1 \) if either of the other two filters is used.

**Figure 3.38**
(a) Blurred image of the North Pole of the moon.
(b) Laplacian without scaling.
(c) Laplacian with scaling.
(d) Image sharpened using the mask in Fig. 3.37(a).
(e) Result of using the mask in Fig. 3.37(b).
(Original image courtesy of NASA.)
Sharpening Filters

Unsharp Masking and Highboost Filtering

- Unsharp masking sharpen images by subtracting an unsharp (smoothed) version of an image from the original image (e.g., printing and publishing industry)
- Steps
  1. Blur the original image
  2. Subtract the blurred image from the original (the resulting difference is called the masks).
  3. Add the mask to the original image

Let \( \overline{f}(x, y) \) denote the blurred image, unsharp masking is

\[
g_{\text{mask}}(x, y) = f(x, y) - \overline{f}(x, y)
\]

Then add a weighted portion of the mask back to the original

\[
g(x, y) = f(x, y) + k \cdot g_{\text{mask}}(x, y) \quad k \geq 0
\]

- When \( k=1 \), we have unsharp masking
- When \( k>1 \), the process is referred to as high boost filtering
Unsharp Masking: Illustration

- Figure 3.39(a) can be interpreted as a horizontal scan line through a vertical edge that transitions from a dark to a light region in an image.

Unsharp Masking and Highboost Filtering: Example
Sharpening Filters

Image Sharpening based on First-Order Derivatives

For function $f(x, y)$, the gradient of $f$ at coordinates $(x, y)$ is defined as

$$\nabla f = \text{grad}(f) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

The magnitude of vector $\nabla f$, denoted as $M(x, y)$

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$
Sharpening Filters

Image Sharpening based on First-Order Derivatives

Roberts Cross-gradient Operators

\[
M(x, y) \approx |z_9 - z_5| + |z_8 - z_6|
\]

Sobel Operators

\[
M(x, y) \approx |(z_7 + 2z_6 + z_9) - (z_1 + 2z_2 + z_3)| + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|
\]
Sharpening Filters

Example

Combining Spatial Enhancement Methods

Goal:

Enhance the image by sharpening it and by bringing out more of the skeletal detail

FIGURE 3.42
(a) Optical image of contact lens (note defects on the perimeter at 4 and 5 o’clock).
(b) Sobel gradient.
(Original image courtesy of Pete Sites, Perceptics Corporation.)

FIGURE 3.43
(a) Image of whole body bone scan.
(b) Laplacian of (a).
(c) Sharpened image obtained by adding (a) and (b).
(d) Sobel gradient of (a).
Sharpening Filters

Example:
Combining Spatial Enhancement Methods
Goal:
Enhance the image by sharpening it and by bringing out more of the skeletal detail

MATLAB Example

- Averaging filter

\[
\begin{array}{ccc}
  a & b & c \\
  d & e & f \\
  g & h & i \\
\end{array}
\rightarrow \frac{1}{9}(a + b + c + d + e + f + g + h + i)
\]

To apply this to an image, consider the 5 × 5 “image” obtained by:

```matlab
>> x=uint8(10*magic(5))
```

```
x =
 170 240  10  80 150
230  80  70 140 160
 40 130 200 220
100 120  190 210  30
110 180 250  20  90
```
Edges of the image

- What happens at the edge of the image, where the mask partly falls outside the image?

Solutions

- Ignore the edges:
  - Apply mask excluding the edges.
  - Resulting image is smaller than original one.

- Pad with zeros:
  - All values outside the image are zero.
  - Resulting image is of the same size with the original one.
  - Causes unwanted artifacts around the image.
The `filter2` function does the job of linear filtering for us; its use is

```
filter2(filter, image, shape)
```

`filter2(filter, image, ‘same’)` is the default; it produces a matrix of equal size to the original image matrix. It uses zero padding:

```
>> a = ones(3,3)/9
    a =
    0.1111  0.1111  0.1111
    0.1111  0.1111  0.1111
    0.1111  0.1111  0.1111

>> filter2(a,x,’same’)  
```

```
ans =
    76.6667  85.5556  65.5556  67.7778  58.8889
    87.7778 111.1111 108.8889 128.8889 105.5556
    66.6667 110.0000 130.0000 150.0000 106.6667
    67.7778 131.1111 151.1111 148.8889  85.5556
    56.6667 105.5556 107.7778  87.7778  38.8889
```

`filter2(filter, image, ‘valid’)` applies the mask only to “inside” pixels. We can see that the result will always be smaller than the original:

```
>> filter2(a,x,’valid’)
```

```
ans =
    111.1111 108.8889 128.8889
    110.0000 130.0000 150.0000
    131.1111 151.1111 148.8889
```
MATLAB Example

We can create our filters by hand, or by using the \texttt{fspecial} function; this has many options which makes for easy creation of many different filters. We shall use the \texttt{average} option, which produces averaging filters of given size; thus

\texttt{fspecial('average',\{5,7\})}

will return an averaging filter of size $5 \times 7$; more simply

\texttt{fspecial('average',\{11\})}

will return an averaging filter of size $11 \times 11$. If we leave out the final number or vector, the $3 \times 3$ averaging filter is returned.

For example, suppose we apply the $3 \times 3$ averaging filter to an image as follows:

\begin{verbatim}
>> c=imread('cameraman.tif');
>> f1=fspecial('average');
>> cfl=filter2(f1,c);
\end{verbatim}

MATLAB Example

Some other useful functions:

- \texttt{imnoise} - Add noise to image.
- \texttt{medfilt2} - 2-D median filtering.
- \texttt{ordfilt2} - 2-D order-statistic filtering.
Summary

• Spatial Filtering Intro
• Spatial Correlation and Convolution
• Smoothing Spatial Filters
  – Linear Smoothing Filters
  – Nonlinear Smoothing Filters
• Sharpening Filters
  – Laplacian Operator
  – Unsharp Masking and Highboost Filtering
  – Using First-Order Derivatives for Nonlinear Image Sharpening - The Gradient
• Matlab Example

References

• “Lecture Notes”, Frank (Qingzhong) Liu, University of New Mexico Tech.
• “Lecture Notes”, Alasdair McAndrew, Victoria University of Technology