Advances in Computer Science Research (ACSR), volume 76
7th International Conference on Education, Management, Information and Mechanical Engineering (EMIM 2017)

Correlation and Convolution Image Filtering Application Analysis

Xihu Zhi[1, a] and Shengli Jiang[1]
[1]College of Information Technology, Luoyang Normal University, Henan Luoyang, 471934, China
[a]zhixihuedu@163.com

Keywords: Correlation; Convolution; Image filter; Boundary points expansion; Filter function redundancy

Abstract. Correlation and convolution are two mathematical operations commonly used in image filtering. This article aims to provide real case analysis to clear the confusions associated with what correlation and convolution essentially are, how to process the boundary points in image processing using correlation and convolution, and how different processing methods affect the result. The redundancy among the common functions in MATLAB image processing using correlation and convolution operations will also be demonstrated here.

Introduction

Correlation and convolution operations are commonly used in mathematical image processing. How correlation and convolution operations work, how the boundary points are processed in image processing using correlation and convolution, why the image size increase or decrease, why there are image artifacts and how to eliminate them, three functions are available in MATLAB, i.e. conv2(), filter2() and imfilter(), to achieve image filtering via correlation and convolution operations on a given image [1]. The redundancy relation can be established by comparing the three function, this article provides analysis on this using actual scenarios.

What Are Correlation and Convolution Operations

Correlation

For scattering computation, correlation operation is essentially multiplication and addition [2].

When the computation involves one-dimensional or linear array A = $\{\ldots, a_{i-1}, a_i, a_{i+1}, \ldots\}$, and given B = $\{b_1, b_2, b_3\}$ as the computation template for array A, the correlation operation for A and B is defined as, align the center element of array B with the first element of array A, multiply the corresponding elements in B and A, and sum up the values to get the first element of the resultant array. Move B one step backward successively and repeat the operations, until all the elements in A have been computed, as illustrated in Figure 1.

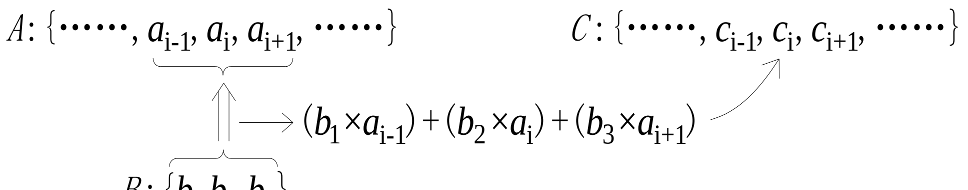
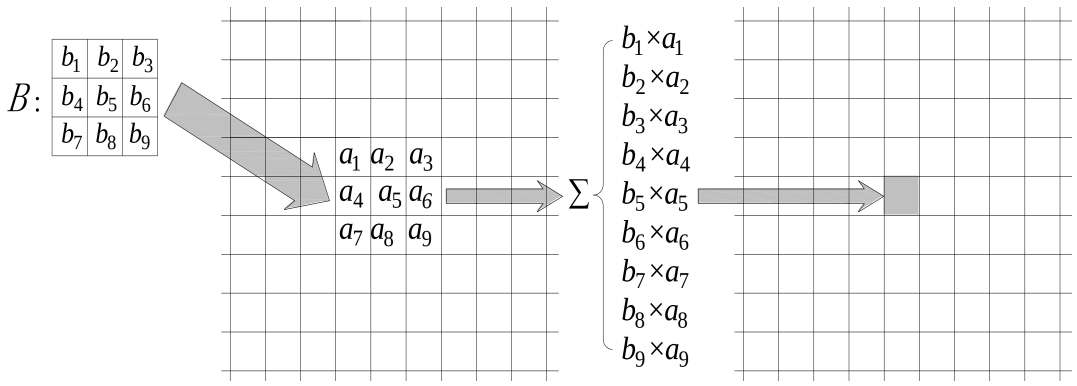

$A: \{\cdots\cdots, a_{i-1}, a_i, a_{i+1}, \cdots\cdots\}$
$C: \{\cdots\cdots, c_{i-1}, c_i, c_{i+1}, \cdots\cdots\}$
$(b_1 \times a_{i-1}) + (b_2 \times a_i) + (b_3 \times a_{i+1})$
$B: \{b_1, b_2, b_3\}$

Figure 1. One-Dimensional Correlation Operation

For two-dimensional correlation operation, assume there is a gray scale input image represented by a two-dimensional array A, and two-dimensional array B is the correlation computation template for A. The correlation operation for A and B is defined as, align the center element of array B with the first-row first-column A element, multiply the corresponding elements in B and A, and sum up the values to get the first element of the resultant array C. Move B one step backward successively and repeat the operations, until all the elements in A have been computed, as illustrated in Figure 2.



a) Template Array B b) Input Image Array A c) Resultant Array C

Figure 2. Two-Dimensional Correlation Operation

Convolution

The similarity between convolution and correlation, is that both operations align the center element of the template array B with every element in the input array A, and successively carry out multiplication and addition [3]. The difference is, for convolution the array B will be preprocessed. For one-dimensional or linear convolution, the template array B is reversed before multiplication and addition. For two-dimensional convolution, the template array B is rotated 180 ° clockwise about its center element (or flipped horizontally and then vertically), before multiplication and addition.

Boundary Points Processing during Image Correlation and Convolution Operation

When processing images using convolution and correlation, the data overlay of template data on input array data is divided into complete overlay scenario and partial overlay scenario. The partially overlaid data are normally the boundary points of the input array data (i.e. the array representing the input image) [4]. Note that how these boundary points are processed affects the visual effect of the resultant image boundaries, or the size of the resultant image, and consequently affects the subsequent processing of the resultant image.

The following strategies are commonly adopted for boundary points processing.

Neglect the Boundary Points

There are two ways to neglect the boundary points:

1) The values of the partially overlaid points are unchanged. This processing method will produce image artifacts as a result of the difference between the output processed pixels and the unprocessed pixels.

2) The values of the partially overlaid points are substituted with assigned constant. If assigned constant is not zero, the resultant image may contain artifacts at the boundary points. If the assigned constant is zero, the resultant image becomes smaller than the original image, and this affects the subsequent processing.

Expand the Boundary Points

This method expands the data points at the boundary of the input image to allow partially overlaid data to become completely overlaid data. The boundary points expansion can be realized as follows.

1) Zero-value expansion [5]. All values outside the image are assumed to be zero. This method causes the boundary of the processed image to darken. The darkened width is directly proportional to the convolution template size.

2) Equivalent value expansion. Values outside the image are assumed to be equivalent to the image boundary values. This method reduces the impact of image artifacts.

3) Mirror image expansion. Values outside the image are assumed to be the same as the row or column values near the boundary of the mirror image.

4) Periodical expansion. The image is assumed to be a two-dimensional periodical function, the pixel values appear periodically in horizontal and vertical directions.

Methods to Eliminate Boundary Artifact in Resultant Image

Boundary image artifact in the resultant image is due to improper boundary value processing. Depending on the situation, the methods detailed in clause 2.1 and 2.2 can be used to process the boundary values; or the expansion method with appropriate expansion value can be adopted, and tested to confirm.

Common Correlation and Convolution Functions in MATLAB

Function Filter2()

Function format:

C=filter2(B,A,'shape')

where:

B: Correlation template array with size mb*nb

A: Input image array with size ma*na

C: Resultant array after correlation

'shape" can take the value 'full', 'same' and 'valid', which are defined as follows:

'full': size of the resultant array C depends on the expansion factor, it is with the size (ma+mb-1)*(na+nb-1). In this case array C is bigger than array A in size.

'same': size of the resultant array C is independent of the expansion factor, it is with the size ma*na. In this case array C is the same as array A in size.

'valid': resultant array C partially excludes the results involved 0 in the computation, it is with the size (ma-mb+1)*(na-nb+1). In this case array C is smaller than array A in size.

For given arrays A and B, the function defines the correlation operation to process the specific boundary points to get resultant array C.

Function Conv2()

Function format:

C=conv2(A,B,'shape')

where:

A: Input image array

B: Convolution template array

C: Resultant array after convolution

Values and definitions of 'shape' are defined in clause 3.1

For given arrays A and B, the function defines the convolution operation to process the specific boundary points to get resultant array C.

Function Imfilter()

Function format:

C=imfilter(A,B,mode,boundary_options,size_options)

where:

A: Input image array

B: Convolution template array

C: Resultant array after correlation or convolution

The value and definition of parameters 'mode', 'boundary_options' and 'size_options' are defined in table 1.

For given arrays A and B, the function defines the correlation or convolution operation to process the specific boundary points to get resultant array C.

Table 1 Value and definition of Mode, Boundary Options and Size Options

Parameter	Value	Definition
mode	'corr'	Filter using correlation operation, the value is default
	'conv'	Filter using convolution
boundary_options	X	Expand the boundary of input array A with X, X has default value 0
	'replicate'	Expand the input array A by duplicating the outer boundary value
	'symmetric'	Expand the input array A by mirror imaging the boundary value
	'circular'	Expand the input array A by treating it as one cycle of a two-dimensional periodic function
size_options	'full'	Output image and expanded input image are of the same size
	'same'	Output image and input image are of the same size, the value is default

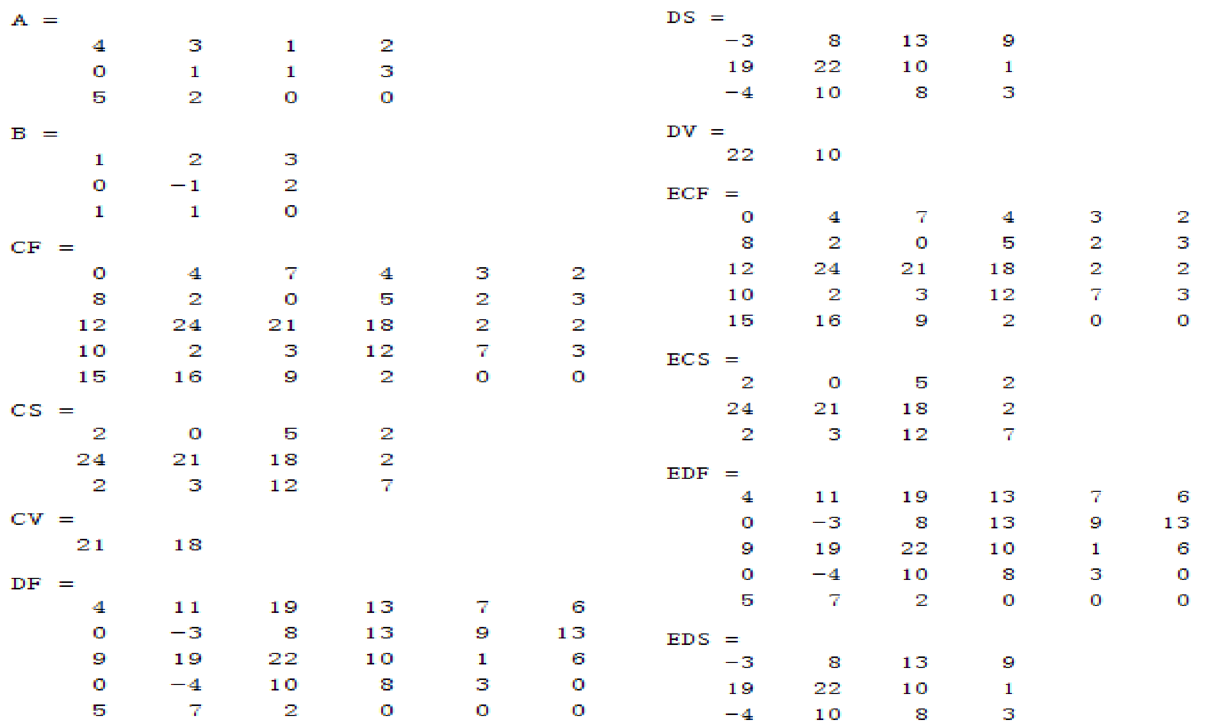
Redundancy Relations between Functions filter2(), conv2() and Function imfilter()

Below is a simple example to illustrate the relation among the three functions.

Assume there are the input image array A and its correlation or convolution operation array B, we can use a program to carry out correlation operation with function filter2(), convolution operation with function conv2(), and correlation and convolution operations with function imfilter() [6]. The MATLAB codes are as follows:

```
close all;clear all;clc;  % close all image windows, clear all variables in
%the workspace, clear command lines
A=[4 3 1 2;0 1 1 3;5 2 0 0]  % array A for the computation
B=[1 2 3;0 -1 2;1 1 0]   % array B for the computation
CF=filter2(B,A,'full')   % CF is the result after correlation operation of A and B,
%it has the same size as A after A is expanded with 0
CS=filter2(B,A,'same')   % CS is the result after correlation operation of A and B,
%it has the same size as A before A is expanded with 0
CV=filter2(B,A,'valid')  % CV is the result after correlation operation of A and B,
%it has smaller size than A
DF=conv2(A,B,'full')     % DF is the result after convolution operation of A and B,
% it has the same size as A after A is expanded with 0
DS=conv2(A,B,'same')     % DS is the result after convolution operation of A and B, it has
%the same size as A before A is expanded with 0
DV=conv2(A,B,'valid')    % DV is the result after convolution operation of A and B,
%it has smaller size than A
ECF=imfilter(A,B,'corr',0,'full') % Carry out ‘full’ type of correlation operation on
%A and B, the result is the same as CF.
ECS=imfilter(A,B,'corr',0,'same')  % Carry out ‘same’ type of correlation operation
%on A and B, the result is the same as CS.
EDF=imfilter(A,B,'conv',0,'full')  % Carry out ‘full’ type of convolution operation
 %on A and B, the result is the same as DF.
EDS=imfilter(A,B,'conv',0,'same')  % Carry out ‘same’ type of convolution
%operation on A and B, the result is the same as DS.
```

Result of the program execution is as follows:



```
A =
     4     3     1     2
     0     1     1     3
     5     2     0     0

B =
     1     2     3
     0    -1     2
     1     1     0

CF =
     0     4     7     4     3     2
     8     2     0     5     2     3
    12    24    21    18     2     2
    10     2     3    12     7     3
    15    16     9     2     0     0

CS =
     2     0     5     2
    24    21    18     2
     2     3    12     7

CV =
    21    18

DF =
     4    11    19    13     7     6
     0    -3     8    13     9    13
     9    19    22    10     1     6
     0    -4    10     8     3     0
     5     7     2     0     0     0
```

```
DS =
    -3     8    13     9
    19    22    10     1
    -4    10     8     3

DV =
    22    10

ECF =
     0     4     7     4     3     2
     8     2     0     5     2     3
    12    24    21    18     2     2
    10     2     3    12     7     3
    15    16     9     2     0     0

ECS =
     2     0     5     2
    24    21    18     2
     2     3    12     7

EDF =
     4    11    19    13     7     6
     0    -3     8    13     9    13
     9    19    22    10     1     6
     0    -4    10     8     3     0
     5     7     2     0     0     0

EDS =
    -3     8    13     9
    19    22    10     1
    -4    10     8     3
```

It is clear that the results for correlation operation on arrays A and B using function imfilter() and correlation operation using function filter2() are the same; the results for convolution operation on arrays A and B using function imfilter() and convolution operation using function conv2() are also the same.

Summary

As discussed above, image filtering using correlation and convolution operations are essentially multiplication and addition. The difference between convolution and correlation, is that convolution requires horizontal and vertical flipping of the template before the computation. When processing the boundary points of an image, the resultant image visual effect and size are affected by the choice of neglecting or expanding the boundary points. To solve the problem of image artifact, different expansion values need to be tested. There is redundancy relation between MATLAB functions filter2(), conv2() and function imfilter(); function imfilter() can substitute functions filter2() and conv2().

References

[1] R.C. Gonzalez, R.E. Woods, and S.L. Eddins. Digital Image Processing Using MATLAB, Pearson Prentice Hall, 2004.

[2] R.C. Gonzalez and R.E. Woods, Digital Image Processing, 3rd edition, Prentice Hall, Upper Saddle River, NJ, 2008.

[3] A.A. Goshtasby. 2-D and 3_D Image Registration, Wiley, 2005.

[4] E.B. Goldstein. Sensation and Perception, 7TH edition, Thomson Wadsworth, Belmont, CA, 2007.

[5] T. Acharya and P.S. Tsai. JPGE2000 Standard for Image Compression: Concepts, Algorithms and VLSI Architectures, Wiley-Interscience, 2004.

[6] R. Lukac. Guest editorial: special issue on applied color image processing: editorials. International Journal of Image Systems and Technology, 17(3):103-104, 2007.