Arithmatic and Logic Operations

Topics to be covered

Arithmatic Operations

- Addition
- Subtraction
- Multiplication
- Division

Logical Operations

- Logical and
- Logical or
- Logical xor
- Logical not

Arithmatic Operations

Arithmetic operations are performed on a pixel-by-pixel basis. It means that the operation is independently applied to each pixel in the image.

Given a 2D image (I) and another 2D array of the same size (S), the resulting array (R) is obtained by

$$R = I \odot S$$

$$R = I \odot S$$

where $\odot$ is any of the following arithmatic operations.

- Addition
- Subtraction
- Multiplication
- Division

Addition

Adding an scalar to an image causes an increase in the intensity of each pixel of an image if the value of the scalar is greater than zero.

Original Image

$$Scalar = \begin{bmatrix} 100 & 100 & \dots & 100 \\ 100 & 100 & \dots & 100 \\ \vdots & \vdots & \ddots & \vdots \\ 100 & 100 & \ddots & 100 \end{bmatrix}$$

Image After Addition

Subtraction

Subtracting an scalar from an image causes the decrease in the intensity of each pixel of an image if the value of the scalar is greater than zero.

Subtraction

Original Image

$$Scalar = \begin{bmatrix} 100 & 100 & \dots & 100 \\ 100 & 100 & \dots & 100 \\ \vdots & \vdots & \ddots & \vdots \\ 100 & 100 & \ddots & 100 \end{bmatrix}$$

Image After Subtraction

Multiplication and Division

Multiplication and division of an image by an scalar factor (greater than zero) is performed for adjusting the brightness of the image.

As a result of multiplicative scaling, brighter image is obtained.

Dividing an image with the scalar factor (greater than zero) results in decrease in brightness and loss of details from the image.

Original Image

$* \quad \text{Scaling Factor} = 2$

Image After Multiplication

Original Image

$\div$ Scaling Factor $= 20$

Image After Division

Logical Operations

Logic operations are performed in a bit-wise manner on the binary contents of each pixel value. Most of the logical operations require two or more input binary contents.

A bitwise operation operates on two-bit patterns of equal lengths by positionally matching their individual bits

We will cover the following most commonly used logical operations on the images.

- Logical and
- Logical or
- Logical xor
- Logical not

Logical and

Logical and of each bit pair results in a 1 if both the first and second bits are 1. If only one bit is a 1, the result is 0.

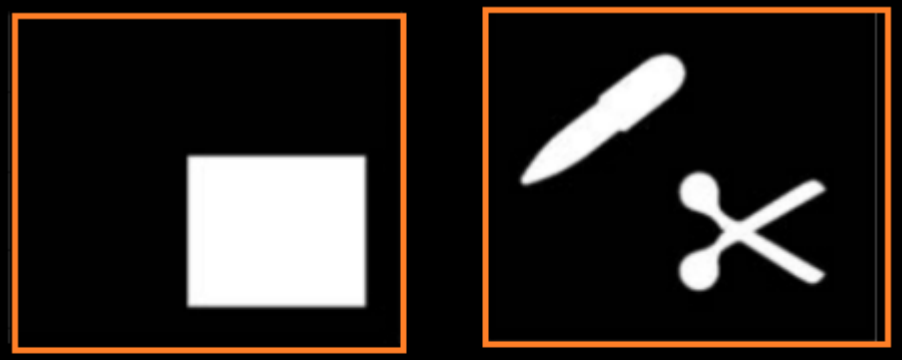
Logical and is used for masking or to extract region of interest (ROI).

Logical and

X_1	X_2	y
0	0	0
0	1	0
1	0	0
1	1	1

Truth Table

For standard convention, 1 represents white pixels and 0 represents black pixels.



X_1 X_2

$X_1 and X_2$

Logical or

Logical or of each bit pair results in a 0 if both bits are 0. If only one bit is a 1, the result is 1.

Logical or is also used for masking or to extract region of interest (ROI).

Logical or

X_1	X_2	y
0	0	0
0	1	1
1	0	1
1	1	1

Truth Table

X_1 X_2

$X_1 or X_2$

Logical xor

Logical xor of each bit pair results in a 0 if both bits are same and 1 if both bits are different.

The Logical xor operation is basically equivalent to calculating the absolute difference between two images.

Logical xor

X_1	X_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Truth Table

X_1

X_2

$X_1 \, xor \, X_2$

Logical not

The Logical not operation is equivalent to find the complement of an image.

X	$not\ X$
0	1
1	0

Truth Table

Logical not

X_2

$not\ X_2$