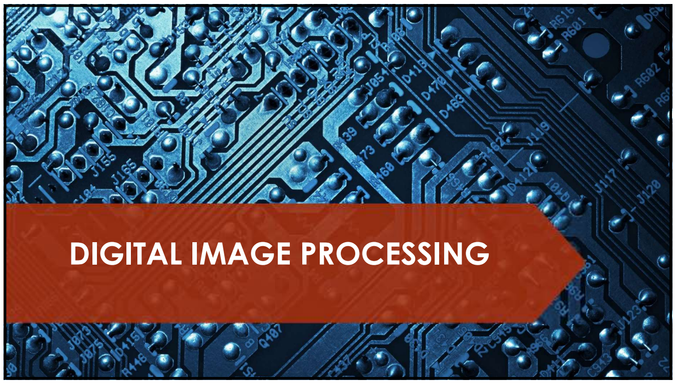
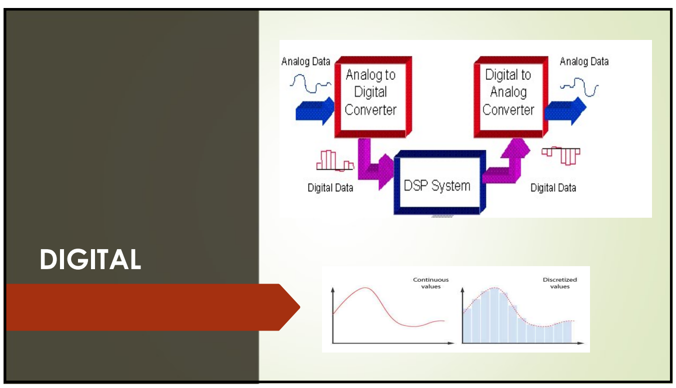
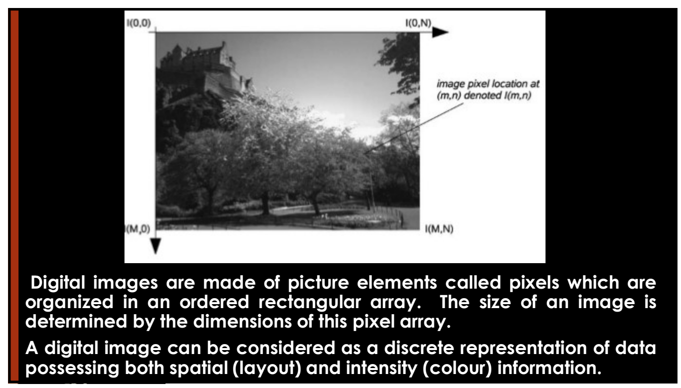


1

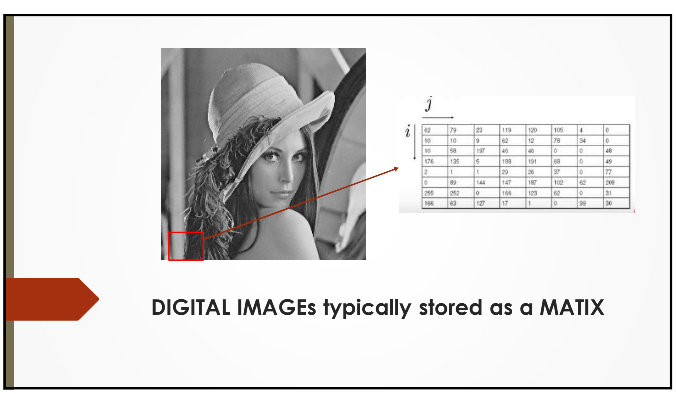
2



3

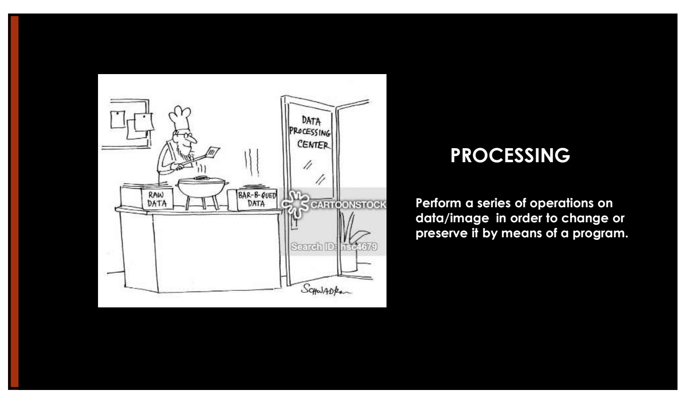


4



62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

5



6

Why Digital Image Processing?

Improvement of Pictorial Information for Human Interpretation

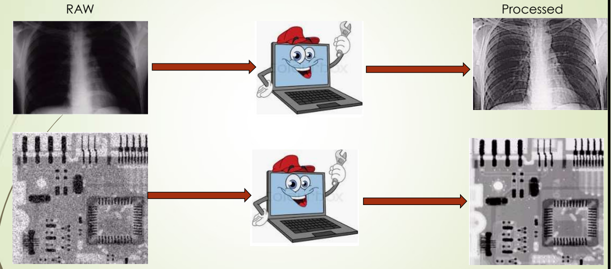
Processing of Image Data for Storage, Transmission and Representation for Autonomous Machine Perception

7

Improvement of Pictorial Information for Human Interpretation

RAW

Processed

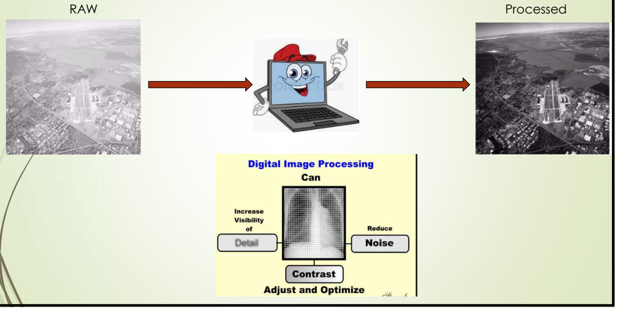


8

Improvement of Pictorial Information for Human Interpretation

RAW

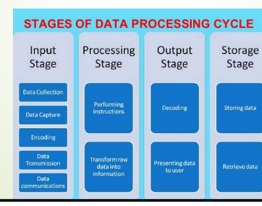
Processed



9

Processing of Image Data for Storage, Transmission and Representation for Autonomous Machine Perception. Cont….

Focuses on procedures for extracting from an image information in a form suitable for computer processing



10

Applications of Digital Image Processing

- **Image sharpening and restoration**: Zooming, blurring , sharpening , gray scale to colors conversion, detecting edges and vice versa , Image recognition.
- **Medical field**: MRI,CT,PET
- **Remote sensing**: Infrastructure damages caused by an earthquake.
- **Machine/Robot vision**: Make robot able to see things , identify them , identify the hurdles etc.
- **Pattern recognition**: Face recognition, Hand writing analysis
- **Others**

11

What are color spaces?

A way to represent the color channels present in the image that gives the image that particular hue.

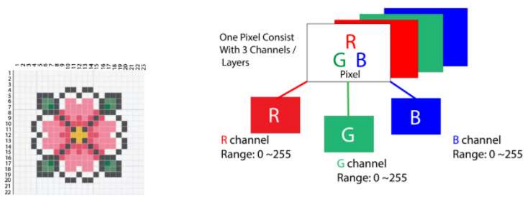
- Color spaces are different types of color modes, used in image processing and signals and system for various purposes. Some of the **common color space**s are:
- RGB (Red, Green, Blue),
- CMYK (Cyan, Magenta, Yellow, Black),
- HSV (Hue, Saturation, Value), etc.

12

Color spaces : RGB Images

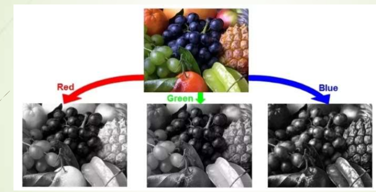
Each pixel coordinate (x, y) contains 3 values ranging for intensities of 0 to 255 (8-bit)
- Red - Green - Blue

Mixing different intensities of each color gives us the full color spectrum.



13

Color spaces : RGB Images

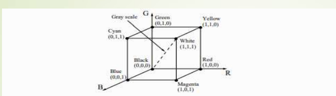


The **RGB color model is additive**: red, green, and blue light are **added together in varying proportions** to produce an extensive range of colors.

RGB image is simply a composite of three independent grayscale images that correspond to the intensity of red, green, and blue light.

14

The CMY Color Model



- any color is written as a sum of the primary colors R(ed), G(reen) and B(lue):

$$Color = rR + gG + bB, \quad r, g, b \in [0,1] \qquad (1)$$

- any color is written as a sum of the primary colors C(yan), M(agenta) and Y(ellow):

$$Color = cC + mM + yY, \qquad (2)$$

- subtractive model

The primary colors can be added to produce the secondary colors:
magenta = red + blue
cyan = green + blue
yellow = red + green

$$\begin{pmatrix}C\\M\\Y\end{pmatrix} = \begin{pmatrix}1\\1\\1\end{pmatrix} - \begin{pmatrix}R\\G\\B\end{pmatrix}$$

Primary colors: red, green and blue

15

The HSV Color Model

Hue is the color portion of the model, expressed as a number from 0 to 360 degrees:

Saturation describes the amount of gray in a particular color, from 0 to 100 percent. Reducing this component toward zero introduces more gray and produces a faded effect. Sometimes, saturation appears as a range from 0 to 1, where 0 is gray, and 1 is a primary color

Value works in conjunction with saturation and describes the brightness or intensity of the color, from 0 to 100 percent, where 0 is completely black, and 100 is the brightest and reveals the most color.

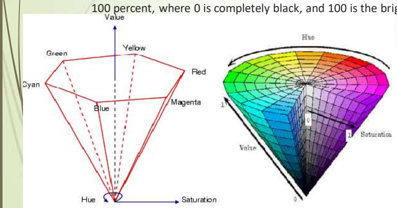
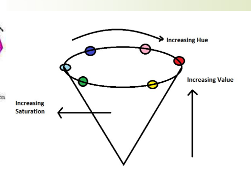


Figure 1. (a) HSV coordinates system (b) HSV color model



16

Important terms

- **Resolution.**
- Pixel Depth
- Dynamic Range

17

Resolution and Bit depth

Resolution is the ability of the imaging system to produce the smallest discernible details.

Spatial Resolution depends on two parameters: **Number of Pixels** and **bit depth** (number of bits necessary for adequate intensity resolution)

BIT DEPTH is determined by **the number of bits used to define each pixel.** The greater the bit depth, the greater the number of tones (grayscale or color) that can be represented.

18

Measuring Spatial Resolution

A ***bitonal image:*** It is represented by pixels consisting of 1 bit each, which can represent two tones (typically black and white), using the values 0 for black and 1 for white or vice versa. To represent two values one bit is sufficient

A ***grayscale image*** is composed of pixels represented by multiple bits of information, typically ranging from 2 to 8 bits or more. (bit depth of 2 and 8 bit image is 2 and 8 respectively)

Binary calculations for the number of tones represented by common bit depths:

1 bit (2^1) = 2 tones
2 bits (2^2) = 4 tones
3 bits (2^3) = 8 tones
4 bits (2^4) = 16 tones
8 bits (2^8) = 256 tones
16 bits (2^{16}) = 65,536 tones
24 bits (2^{24}) = 16.7 million tones

19

Measuring Spatial Resolution

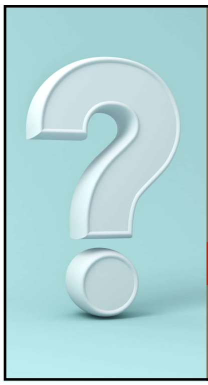
Total number of bits necessary to represent the image is :
Number of Rows X Number of Columns X Bit depth

As spatial resolution refers to clarity, so for different devices, different measure has been made to measure it.

For example:

- **Dots per inch:** DPI is usually used in monitors.
- **Lines per inch:** LPI is usually used in laser printers
- **Pixels per inch:** PPI is measure for different devices such as tablets , Mobile phones etc..

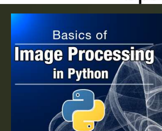
20



QUIZ 1

https://quizizz.com/admin/quiz/5f2a296b6e2cf5001b6a8b9f/image-processing

21



PYTHON

Using Python, perform following task on a given image:

I. Reading and Displaying the Image
II. Know the dimension of an image
III. Know the pixel intensities inside a ROI and at specific locations
IV. Cropping the Image and changing the size of an image

22

Introduction to Image Enhancement

23

Image Enhancement

Need of Image Enhancement: Provides better contrast and a more detailed image

Spatial Domain: Manipulation with pixels

Frequency Domain: Modifying the frequency transform of an image

24

Image Enhancement

The transformation function has been given below:

- **s = T (r)**

Or

- **g(x,y)=T(f(x,y))**

- where **r; f(x,y)**: the pixels of the input image and **s; g(x,y)**: the pixels of the output image.

- **T** is a transformation function that maps each value of **r** to each value of **s**.

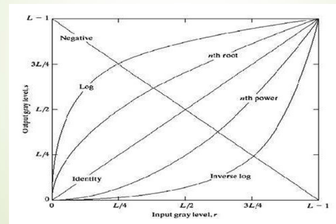
- **Image enhancement can be done through gray level transformations**

25

Gray Level Transformations

There are three basic gray level transformation.

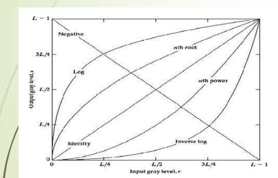
- **Linear**: The Identity and Negative curves
- **Logarithmic**: The Log and Inverse-Log curves
- **Power – law**: *n*th root and *n*th power transformations



26

Linear Gray Level Transformations

Also called the Identity and Negative curves



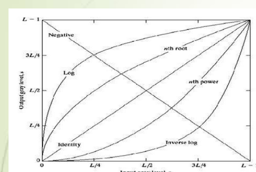
The negative of an image with grey levels in the range [0, L-1] is obtained by the negative transformation shown in figure, which is given by the expression,
s = L - 1 – r **or** *g(x,y) = L - 1 – f(x,y)*

For Binary image: changes 0 to 1 and 1 to 0

27

Logarithmic Gray level Transformations

The Log and Inverse-Log curves



The log transformation curve shown in fig, is given by the expression, ***s = c log(1 + r)***

where **c** is a constant and it is assumed that $r \geq 0$.

Maps

- a narrow range of low-level grey scale intensities into a wider range of output values.
- the wide range of high-level grey scale intensities into a narrow range of high-level output values.

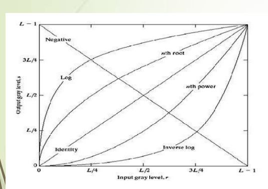
Expand values of dark pixels (increases the dynamic range of dark regions)
and
compress values of bright pixels (reduces the dynamic range of lighter regions)
.

The opposite is true for inverse-log transform.

28

Power Law Transformations /Gamma correction

*n*th power and *n*th root curves

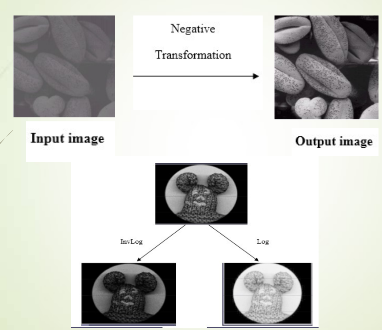


The gamma transformation curve shown in fig, is given by the expression, $s = c\,r^{\gamma}$

For various values of γ different levels of enhancements can be obtained

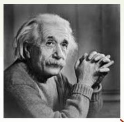
Different display monitors display images at different intensities and clarity. That means, every monitor has built-in gamma correction in it with certain gamma ranges

29



30

Input image



Negative transformation

Log transformation

Gamma = 10

Output image

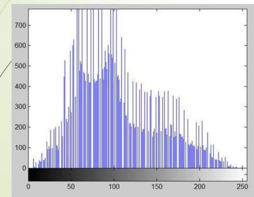
Power–Law transformations

31

Histogram

Histogram of an image, shows frequency of pixels intensity values.

Histograms plots show many times (frequency) each intensity value in image occurs



x axis: the range of pixel values; gray level intensities

y axis: frequency of these intensities or count of these intensities.

8 bpp image has 256 levels of gray.

Graph shows that most of the bars that have high frequency lies in the first half portion which is the darker portion. That means that the image we have got is darker.

32

Histogram

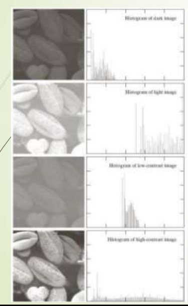
Use of Histogram of an image:

1. For the analysis of the image.
2. For adjusting contrast of an image/ enhancement.
3. To equalize an image.
4. Use in Segmentation: thresholding.

33

Histograms

Four basic image type and their corresponding histograms:



Dark image

Light image

Low contrast image

High contrast image

34

Histogram Equalization

Histogram Equalization used to improve contrast in images. It accomplishes this by effectively spreading out the most frequent intensity values, i.e. stretching out the intensity range of the **image**.

Histogram equalization is a technique for adjusting image intensities to enhance contrast.

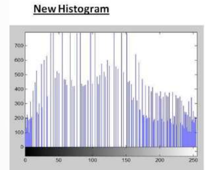
It is not necessary that contrast will always be increase in this. There may be some cases were histogram equalization can be worse. In that cases the contrast is decreased.

35

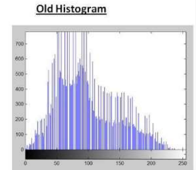
New Image

Old image

New Histogram



Old Histogram



Histogram Equalization

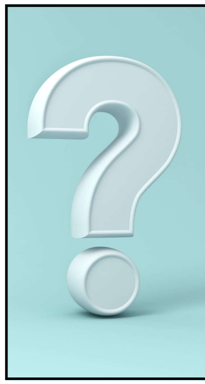
36

PYTHON

Using Python, perform following task on an image:

1. Histogram Equalization
2. Gray level Transformation

37



QUIZ 2

https://quizizz.com/admin/quiz/5f3614dd2f75f7001c09ce1a/image-processing-part

38

Arithmetic operations

- Arithmetic operations between images are array operations means that arithmetic operations are carried out between corresponding pixel pairs.

- **Addition**
- **Subtraction**
- **Division**
- **Averaging**
- **Multiplication**

39

Addition

Useful for combining information between two images

$$Q(i,j) = P_1(i,j) + P_2(i,j)$$

Or if it is simply desired to add a constant value C to a single image then:

$$Q(i,j) = P_1(i,j) + C$$

a b c

40

Subtraction

Useful for "change" detection., background elimination, Brightness reduction

The difference between two images $f(x, y)$ and $h(x, y)$, expressed as

$$g(x, y) = f(x, y) - h(x, y).$$

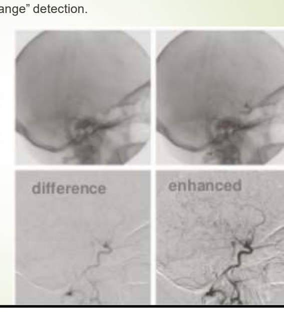
$$O(r,c) = |I_1(r,c) - I_2(r,c)|$$

41

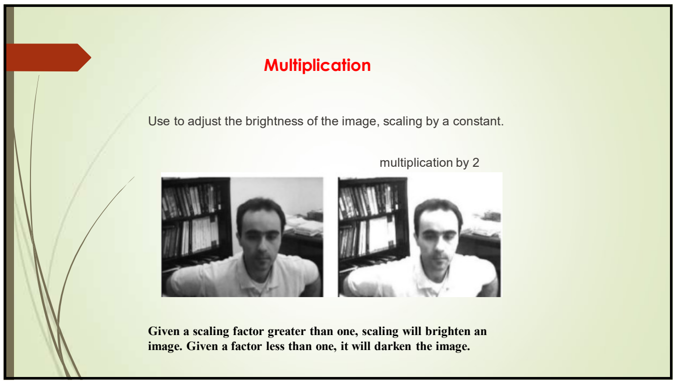
Subtraction

Useful for "change" detection.

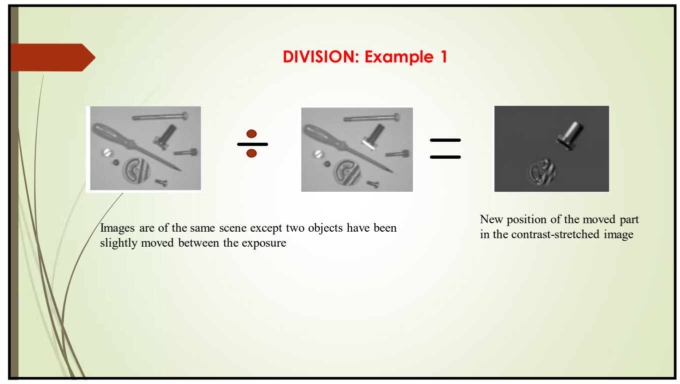
Medical application: Iodine medium injected into the bloodstream



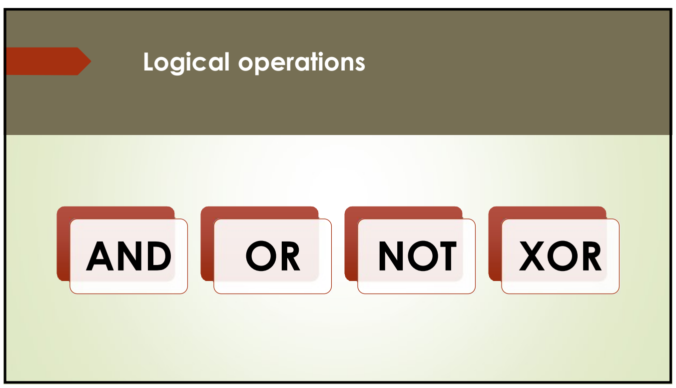
42



43

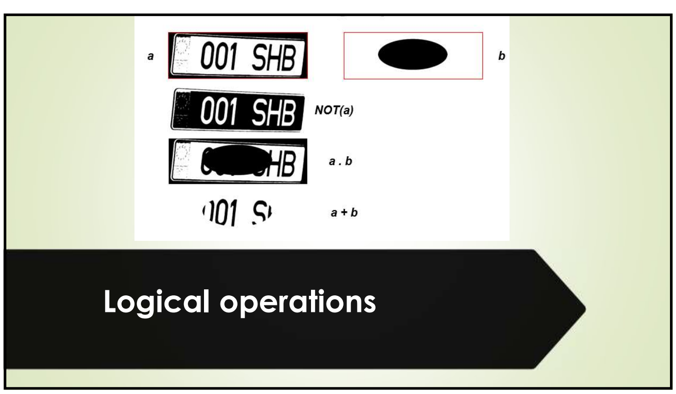


44

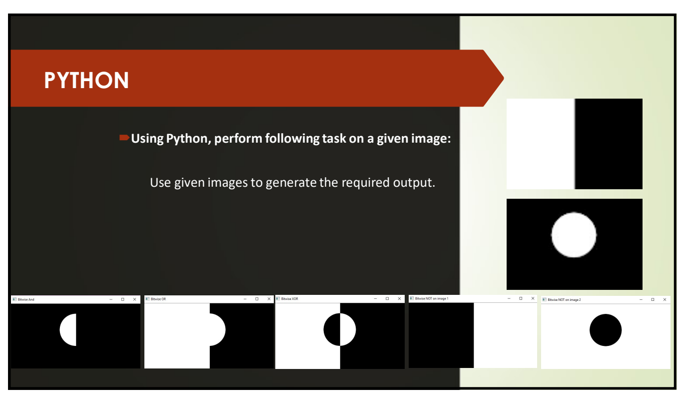


45

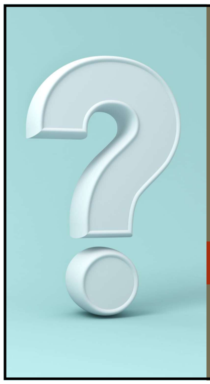
46



47



48



QUIZ 3

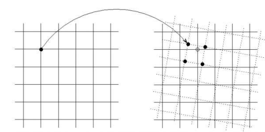
https://quizizz.com/admin/quiz/5f45556f8841b100 1b725bd5/image-processing

49

Geometrical operations

In a spatial transformation each point (x, y) of image A is mapped to a point (u, v) in a new coordinate system.

$$u = f_1(x, y)$$
$$v = f_2(x, y)$$



50

Geometrical operations

A spatial transformation of an image is a geometric transformation of the image coordinate system. It is often necessary to perform a spatial transformation to:

- Align images that were taken at different times or with different sensors
- Correct images for lens distortion
- Correct effects of camera orientation

51

Geometrical operations

- Scale - change image content size
- Rotate - change image content orientation
- Reflect - flip over image contents
- Translate - change image content position
- Affine Transformation - general image content linear geometric transformation

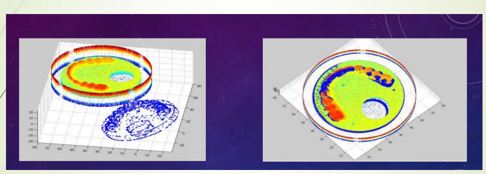
An affine transformation is any transformation that preserves collinearity (i.e., all points lying on a line initially still lie on a line after transformation) and ratios of distances

An affine transformation is a composition of rotations, translations, dilations, and shears.

52

Geometrical operations

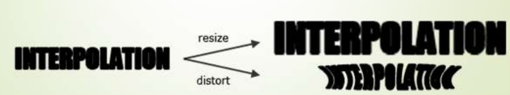
Example: Dental Scan Image processing



53

DIGITAL IMAGE INTERPOLATION

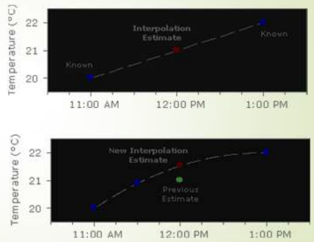
- To resize or remap (distort) an image from one pixel grid to another.
  - Image resizing: increase or decrease the total number of pixels,
  - Image remapping: correcting for lens distortion, changing perspective, and rotating an image.
- Image will always lose some quality each time interpolation is performed.



54

DIGITAL IMAGE INTERPOLATION

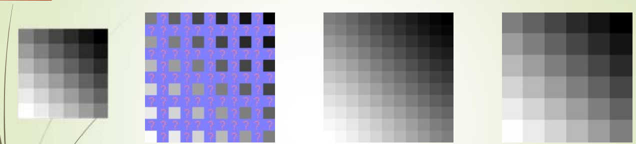
- Interpolation works using known data to estimate values at unknown points.
- For example:
  - To know the temperature at noon, but only measured it at 11AM and 1PM, you could estimate its value by performing a linear interpolation.
  - If we have an additional measurement at 11:30AM, we could see that the bulk of the temperature rise occurred before noon, and could use this additional data point to perform a quadratic interpolation



The more temperature measurements you have which are close to noon, the more sophisticated (and hopefully more accurate) your interpolation algorithm can be

55

DIGITAL IMAGE INTERPOLATION



Tries to achieve a best approximation of a pixel's color and intensity based on the values at surrounding pixels

45° Rotation

90° Rotation

Detail is lost in just the first rotation, although the image continues to deteriorate with successive rotations

56

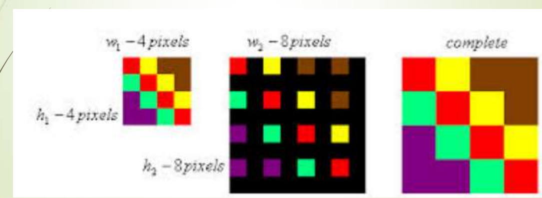
TYPES OF INTERPOLATION

- Nearest neighbour Interpolation
- Bilinear Interpolation
- Bicubic interpolation

57

Nearest neighbour Interpolation

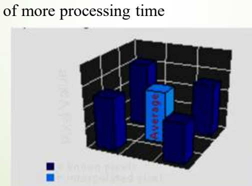
- Most basic method
- Requires the least processing time
- Only considers one pixel: the closest one to the interpolated point
- Has the effect of simply making each pixel bigger



58

Bilinear Interpolation

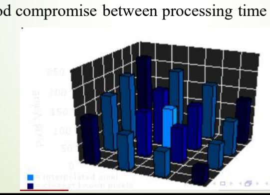
- Considers the closest 2x2 neighborhood of known pixel values surrounding the unknown pixels
- Takes a weighted average of these 4 pixels to arrive at the final interpolated values
- Results in smoother looking images than nearest neighborhood
- Needs of more processing time



59

Bicubic Interpolation

- One step beyond bilinear by considering the closest 4x4 neighborhood of known pixels, for a total of 16 pixels
- Since these are at various distances from the unknown pixel, closer pixels are given a higher weighting in the calculation
- Produces sharper images than the previous two methods.
- Good compromise between processing time and output quality



60

PYTHON

Using Python:

Perform geometric operation such as resizing, translation, rotation, affine transformation

61

SPATIAL FILTERING

62

FILTERING

- SPATIAL filtering deals with operation on pixels then on frequencies
- **Filters in image processing can be characterized into the following three categories:**
  - **Convolution- based filters:** Use spatial masks (kernels, templates etc.)
  - **2. Order Statistics :** Do Not use Convolution techniques. Simply arrange the pixels that are under the mask in a desired order. Then rank is chosen as the resultant e.g. Median filter.
  - **3. Hybrid Filters:** Uses the concept of both the ranking and convolution. Eg Unsharp masking

63

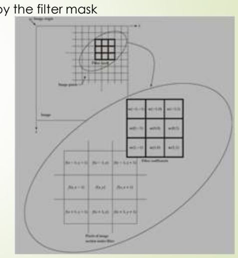
SPATIAL FILTERING

Convolution Method: Moving the filter mask from point to point in an image.

The response is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask

- Typically linear combinations of pixel values.
  - e.g., weight pixel values and add them together.
- Different results can be obtained using different weights.
  - e.g., smoothing, sharpening, edge detection).

mask



64

Input image * Weights → Output image

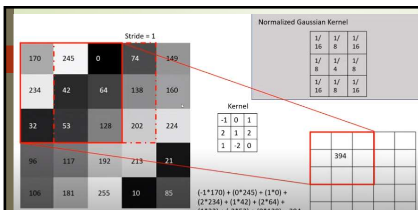
4	5	7	6	6
3	2	8	0	7
6	7	7	1	5
3	0	1	1	1
4	3	2	1	7

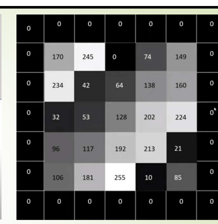
*

0	0	0
1	0	1
0	0	0

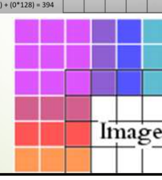
→

65



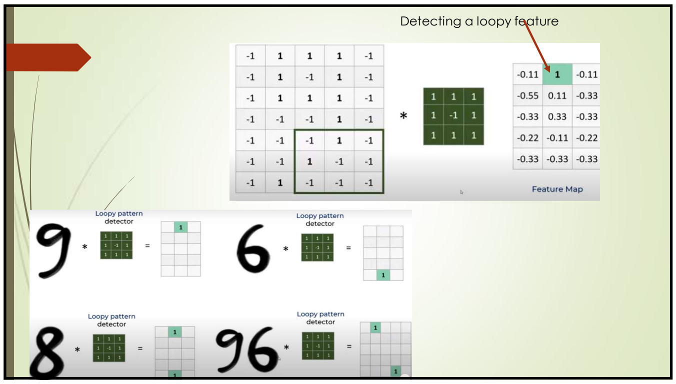


Padding

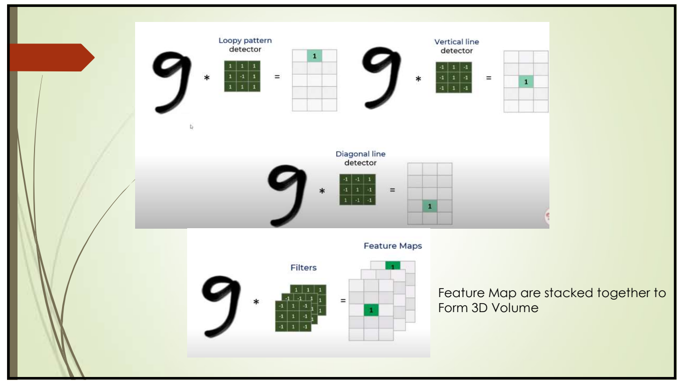


Replicating

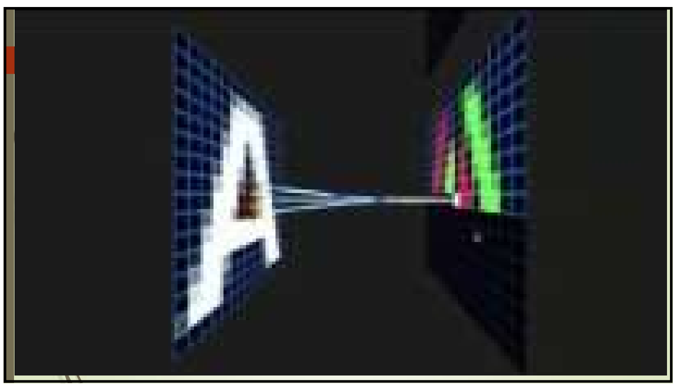
66



67



68



69

SMOOTHING SPATIAL FILTER

70

SMOOTHING SPATIAL FILTERING

One of the simplest spatial filtering operations we can perform is a smoothing operation

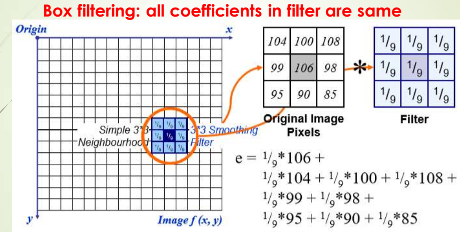
- Simply **average** all of the pixels in a neighbourhood around a central value
- Especially useful in removing noise from images
- Also useful for highlighting gross detail

$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$
$^1/_9$	$^1/_9$	$^1/_9$

71

SMOOTHING SPATIAL FILTERING:

Box filtering: all coefficients in filter are same

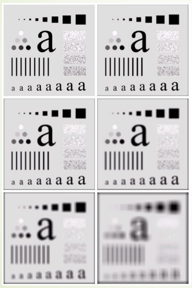


$e = {}^1/_9*106 + {}^1/_9*104 + {}^1/_9*100 + {}^1/_9*108 + {}^1/_9*99 + {}^1/_9*98 + {}^1/_9*95 + {}^1/_9*90 + {}^1/_9*85 = 98.3333$

The above is repeated for every pixel in the original image to generate the smoothed image.

72

SMOOTHING SPATIAL LINEAR FILTERING



- The image at the top left is an original image of size 500*500 pixels
- The subsequent images show the image after filtering with an averaging filter of increasing sizes
  - 3, 5, 9, 15 and 35
- Notice how detail begins to disappear

73

WEIGHTED SMOOTHING FILTERING

More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function

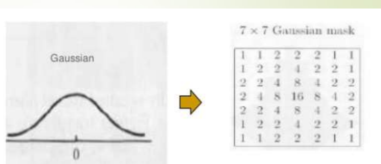
- Pixels closer to the central pixel are more important
- Often referred to as a *weighted averaging*

$^1/_{16}$	$^2/_{16}$	$^1/_{16}$
$^2/_{16}$	$^4/_{16}$	$^2/_{16}$
$^1/_{16}$	$^2/_{16}$	$^1/_{16}$

Weighing the centre point the highest and reducing the value of the coefficient as a function of increasing distance from origin is to reduce blurring in smoothing process.

74

WEIGHTED SMOOTHING FILTERING



75

SMOOTHING SPATIAL Non LINEAR FILTERING

	Ascending order / Descending order →	Median Value
Median Filter	1 ,7, 4 ,10, 9	8, 16 ,21, 7 ,9, 18, 9
	{1 4 7 9 10}	{21 18 16 12 9 8 7}

The **Median Filter** is a non-linear digital filtering technique, often used to remove noise from an image or signal.

76

SMOOTHING SPATIAL Non LINEAR FILTERING

8	9	10	11	12
12	8	9	10	11
13	12	8	9	10
14	13	12	8	9
15	14	13	12	8

0	0	0			
0	8	9	10	11	12
0	12	8	9	10	11
	13	12	8	9	10
	14	13	12	8	9
	15	14	13	12	8

Ascending order

0 0 0 0 0 8 8 9 12

Median value

0	0	0		
8	9	10	11	12
12	8	9	10	11
13	12	8	9	10
14	13	12	8	9
15	14	13	12	8

Descending order

12 10 9 9 8 8 0 0 0

Median Value

	0	0	0	
8	9	10	11	12
12	8	9	10	11
13	12	8	9	10
14	13	12	8	9
15	14	13	12	8

Ascending order

0 0 0 8 9 9 10 10 11

Median Value

0	8	9		

77

SMOOTHING SPATIAL Non LINEAR FILTERING

- The **Median Filter is 50th percentile of a ranked set of number**

The **Other Filter:**

- **Max filter is 100th percentile filter: Useful in finding brightest points in image**
- **Min filter is 0 percentile filter: Useful in finding dark points in image**

78

BILATERAL FILTERING

Tonal weighting makes that the bilateral filter is capable of preserving edges (large differences in tonal value) while smoothing in the more flat regions (small tonal differences).

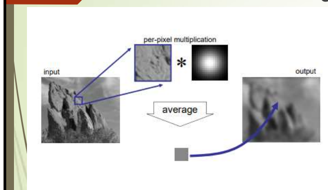
Based on :

- **Space parameter:** Weighting (based on a Gaussian distribution) intensity based on Euclidean distance of pixels,
- **Range parameter**: But also, on the radiometric differences (e.g., range differences, such as color intensity, depth distance, etc.). This preserves sharp edges.

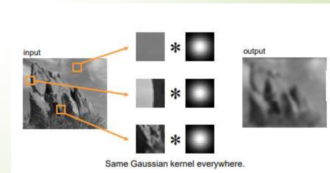
79

Gaussian Vs Bilateral Filtering

Gaussian



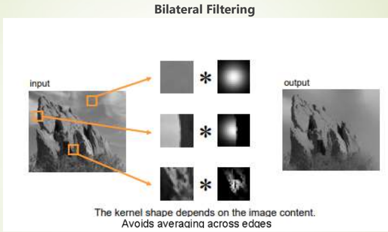
Space parameter: size of the window



80

Gaussian Vs Bilateral Filtering

Bilateral Filtering



Based on :

- **Space parameter:** Weighting (based on a Gaussian distribution) intensity based on on Euclidean distance of pixels,
- **Range parameter**: But also, on the radiometric differences (e.g., range differences, such as color intensity, depth distance, etc.). This preserves sharp edges.

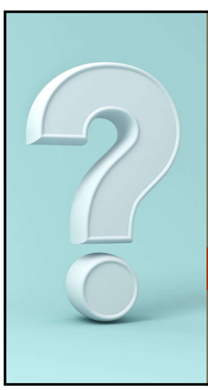
81

PYTHON

Using Python:

Perform Image smoothing using different smoothing filters.

82



QUIZ 4

https://quizizz.com/admin/quiz/5f60410777d3bf001b2e5417/image-processing

83

SHARPENING SPATIAL FILTER

84

SHARPENING SPATIAL FILTERING

Previously we have noticed that: **The image blurring is accomplished by pixel averaging in a neighborhood. Averaging is analogous to integration.**

Blurring vs. Sharpening

- Blurring/smooth is done in spatial domain by pixel averaging in a neighbors, it is a process of integration
- Sharpening is an inverse process, to find the difference by the neighborhood, done by **spatial differentiation.**

85

SHARPENING SPATIAL FILTERING

Derivative operator

- The strength of the response of a derivative operator is proportional to the degree of discontinuity of the image at the point at which the operator is applied.
- Image differentiation
  - enhances edges and other discontinuities (noise)
  - deemphasizes area with slowly varying gray-level values.

86

SHARPENING SPATIAL FILTERING

First and Second-order derivative of 2D

- when we consider an image function of two variables, f(x, y), at which time we will dealing with partial derivatives along the two spatial axes.

Gradient operator (linear operator)

$$\nabla \mathrm{f} = \frac{\partial f(x,y)}{\partial x \partial y} = \frac{\partial f(x,y)}{\partial x} + \frac{\partial f(x,y)}{\partial y}$$

Laplacian operator (non-linear)

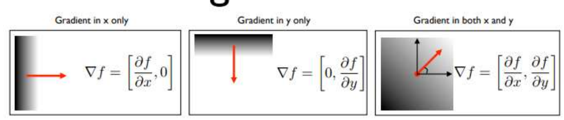
$$\nabla^2 f = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$$

Gradient refers to the difference between the pixels of an image.

If neighboring pixels have the same intensity, the difference is zero therefore **no edge** is present.

87

Image Gradient



Gradient direction

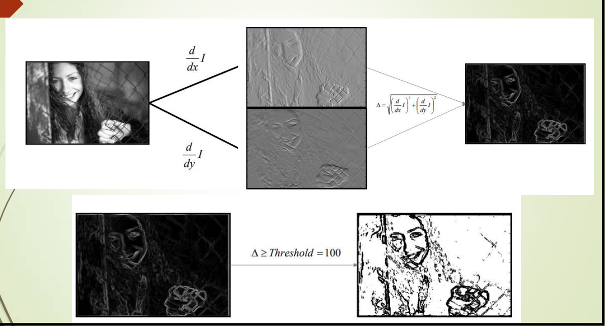
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$$

Gradient magnitude

$$||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

88

Edge Detection Using the Gradient



89

Gradient

How do you compute the image gradient?

Choose a derivative filter

$$S_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Run filter over image

$$\frac{\partial f}{\partial x} = S_x \otimes f \qquad \frac{\partial f}{\partial y} = S_y \otimes f$$

Image gradient

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

90

SHARPENING SPATIAL FILTERING: Laplacian Operator

- Laplacian Operator is
  - a derivative operator
  - highlights regions of rapid intensity change
  - used to find edges in an image.
- The major difference between Laplacian and other operators like Prewitt, Sobel, Robinson and Kirsch is that these all are first order derivative masks but Laplacian is a second order derivative mask.
- Derivatives are strongly affected by noise, thus image is first smoothened such as with Gaussian smoothing filter in order to reduce its sensitivity to noise.

91

SHARPENING SPATIAL FILTERING: Laplacian Operator

In Laplacian mask we have two classifications
- Positive Laplacian Operator and
- Negative Laplacian Operator.

Positive Laplacian Operator

In Positive Laplacian we have standard mask in which center element of the mask should be negative and corner elements of mask should be zero.

0	1	0
1	-4	1
0	1	0

Negative Laplacian Operator

In negative Laplacian operator we also have a standard mask, in which center element should be positive. All the elements in the corner should be zero and rest of all the elements in the mask should be -1.

0	-1	0
-1	4	-1
0	-1	0

92

SHARPENING SPATIAL FILTERING: Laplacian Operator

Effect of Laplacian Operator

- as it is a derivative operator,
  - it highlights gray-level discontinuities in an image
  - it deemphasizes regions with slowly varying gray levels
- tends to produce images that have
  - grayish edge lines and other discontinuities, all superimposed on a dark,
  - featureless background.

While applying Laplacian highlights fine detail, it de-emphasizes smooth regions (e.g., background features). It results in featureless background with grayish fine details.

93

SHARPENING SPATIAL FILTERING: Laplacian Operator

Correct the effect of featureless background

- easily by adding the original and Laplacian image.
- be careful with the Laplacian filter used

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is negative} \\ f(x,y) + \nabla^2 f(x,y) & \text{if the center coefficient of the Laplacian mask is positive} \end{cases}$$

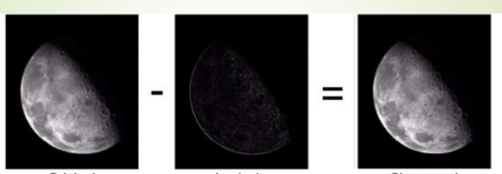
Important :

Subtract the resultant image from the original image to get the sharpened image if positive Laplacian operator is used

Add the resultant image onto original image to get the sharpened image if negative Laplacian operator is used.

94

SHARPENING SPATIAL FILTERING: Laplacian Operator



In the final sharpened image edges and fine detail are much more obvious

95

SHARPENING SPATIAL FILTERING: Laplacian Operator

ORIGINAL IMAGE

SHARPENED IMAGE

96

Unsharp Masking

$$f_s(x,y) = f(x,y) - \bar{f}(x,y)$$

sharpened image = original image – blurred image

- to subtract a blurred version of an image produces sharpening output image.

Where $f_s(x,y)$ denotes the sharpened image obtained by unsharp masking, an $\bar{f}(x,y)$ is a blurred version of $f(x,y)$

97

Unsharp Masking and Highboost filtering

How we can sharpen an image or perform edge enhancement using a smoothing

- First, blur the image: Smoothing an image suppress most of the high-frequency components.
- Then, subtract this smoothed image from the original image (the resulting difference is known as a mask). The output image will have most of the high-frequency components that are blocked by the smoothing filter.
- Adding this mask back to the original will enhance the high-frequency components.

As blurred or unsharp image is used to create a mask this technique is known as Unsharp Masking.

98

Unsharp Masking and Highboost filtering

Thus, unsharp masking first produces a mask m(x,y) as

$$m(x,y) = f(x,y) - f_b(x,y)$$

f(x,y) is original image and fb(x,y) is blurred image

Then this mask is added back to the original image which results in enhancing the high-frequency components.

$$g(x,y) = f(x,y) + k * m(x,y)$$

where k specifies what portion of the mask to be added. When k= 1 this is known as Unsharp masking. For k>1 we call this as high-boost filtering because we are boosting the high-frequency components by giving more weight to the masked (edge) image.

$$g(x,y) = (k+1) * f(x,y) - k * f_b(x,y)$$

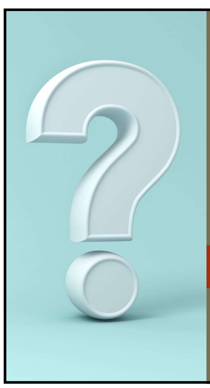
99

PYTHON

Using Python:

Perform Image sharpening using different filters.

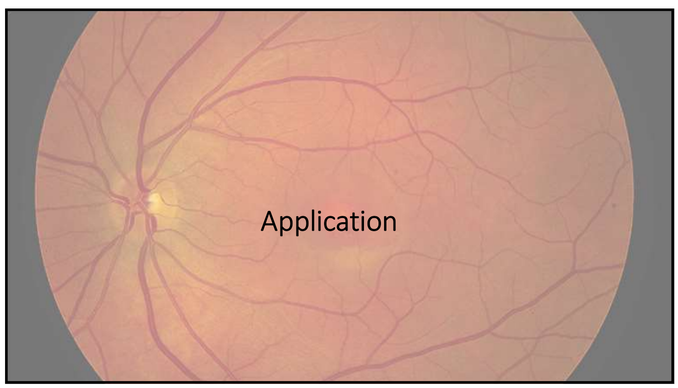
100



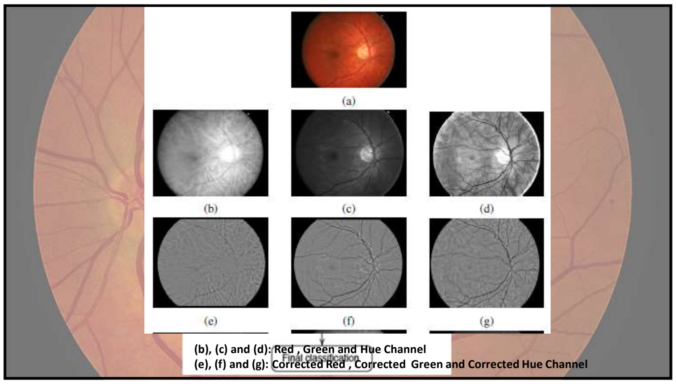
QUIZ 5

https://quizizz.com/admin/quiz/5f4d05443478a8001b8624f7/image-processing

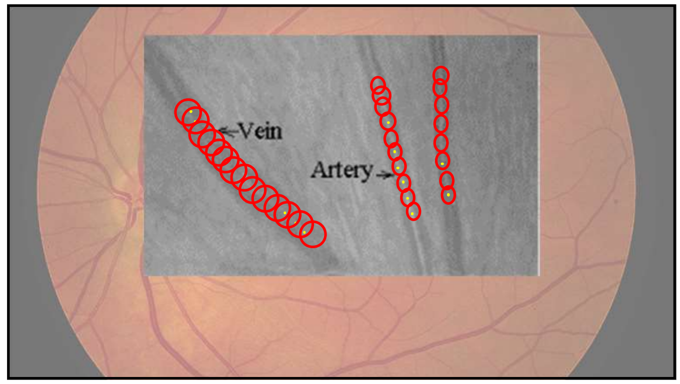
101



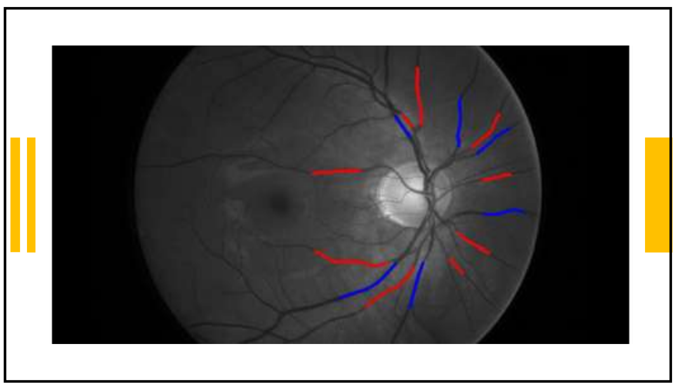
102



103



104



105

106

Frequency Domain FILTERING

- Based on Frequency content
- Fourier transform such as discrete cosine transform, Fourier transforms, Hartley transforms etc. are Frequency domain analysis methods.
- Example :
  - Edges and sharp transitions (example Noise) in gray values in an image contributes significantly to **high frequency content of its Fourier Transform.**
  - Regions of relatively uniform gray values in an image contribute to low-frequency content of its Fourier transform. In other words **Low frequency contents in the Fourier Transform** are responsible to the general appearance of the image over smooth area

107

Fourier transform (FT)

One Dimensional FT

The Fourier transform, $F(u)$, of a single variable, continuous function, $f(x)$, is defined by the equation

$$F(u) = \int_{-\infty}^{\infty} f(x)e^{-j2\pi ux}\,dx \qquad (4.2\text{-}1)$$

inverse Fourier transform

$$f(x) = \int_{-\infty}^{\infty} F(u)e^{j2\pi ux}\,du.$$

108

Fourier transform (FT)

FT/Fourier series using two variable

$$F(u, v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y)e^{-j2\pi(ux+vy)}\, dx\, dy \qquad (4.2\text{-}3)$$

and, similarly for the inverse transform,

$$f(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u, v)e^{j2\pi(ux+vy)}\, du\, dv. \qquad (4.2\text{-}4)$$

The Fourier transform simply states that the non periodic signals whose area under the curve is finite can also be represented into integrals of the sines and cosines after being multiplied by a certain weight.

109

Discreate Fourier transform (DFT)

DFT using one variable

The Fourier transform of a discrete function of one variable, $f(x)$, $x = 0, 1, 2, \ldots, M-1$, is given by the equation

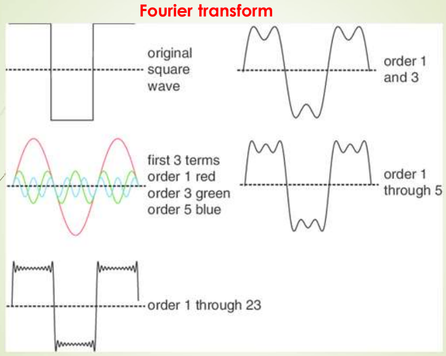
$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)e^{-j2\pi ux/M} \qquad \text{for } u = 0, 1, 2, \ldots, M-1. \qquad (4.2\text{-}5)$$

Similarly, given $F(u)$, we can obtain the original function back using the inverse DFT:

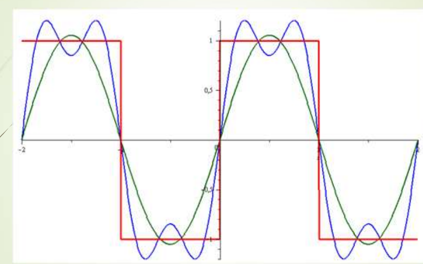
$$f(x) = \sum_{u=0}^{M-1} F(u)e^{j2\pi ux/M} \qquad \text{for } x = 0, 1, 2, \ldots, M-1. \qquad (4.2\text{-}6)$$

110

Fourier transform



111



112

Frequency Domain FILTERING

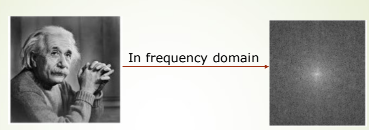
- **Image in frequency domain has two components :**
  - **Magnitude (Consists of the frequencies content of images) and**
  - **Phase (used to restore the image back to the spatial domain)**

The magnitude of the sinusoid directly relates with the contrast.

Phase contains the color information.

113

Discreate Fourier transform (DFT)



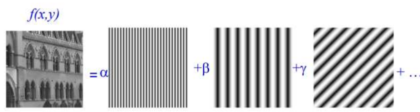
The relationship between the spatial domain and the frequency domain can be established by convolution theorem.

114

The spatial function $f(x, y)$

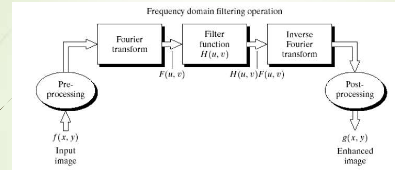
$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{j2\pi(ux+vy)}\,du\,dv$$

is decomposed into a weighted sum of 2D orthogonal basis functions in a similar manner to decomposing a vector onto a basis using scalar products.



115

Frequency Domain filtering: Basic operation



To filter an image in the frequency domain:
- Compute F(u,v) the DFT of the image
- Multiply F(u,v) by a filter function H(u,v)
- Compute the inverse DFT of the result

116

FILTERS

Filter function H(u, v) can:
- amplifies some frequencies and
- suppresses certain frequency components in an image.

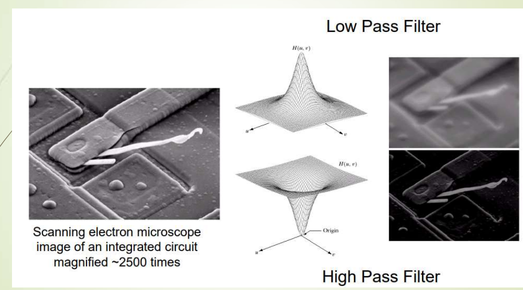
Application: blurring/smoothing, sharpening and edge detection in an image.

Based on the property of using the frequency domain the image filters are broadly classified into two categories:

1. Low-pass filters / Smoothing filters.
2. High-pass filters / Sharpening filters.

117

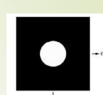
Low-pass filters / Smoothing filters VS High-pass filters / Sharpening filters.



Scanning electron microscope image of an integrated circuit magnified ~2500 times

118

Image Smoothing
uses Low-pass Frequency Domain Filters



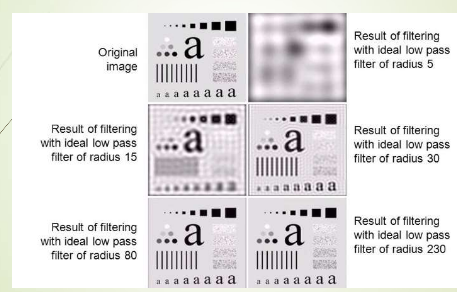
A low-pass filter **attenuates (suppresses) high frequencies** while **passing the low frequencies** which results in creating a blurred (smoothed) image.

In other words: It leaves the low frequencies of the Fourier transform relatively unchanged and ignores the high frequency noise components.

Three main low-pass filters are:
- Ideal low-pass filter (ILPF) (very sharp)
- Butterworth low-pass filter (tunable)
- Gaussian low-pass filter (very smooth)

119

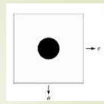
Ideal Low-pass filters



120

Image Sharpening

uses High-pass Frequency Domain Filters



High pass filtering process attenuates (suppress) low frequency components without disturbing high frequency information in the Fourier transform of the image.

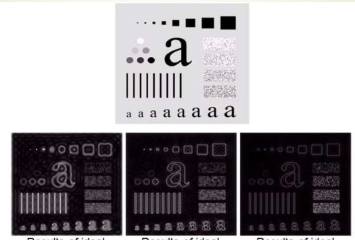
The high-pass filter H_{hp} is often represented by its relationship to the low-pass filter (H_{lp}) as: **H_{hp} (u, v) =1- H_{lp} (u, v)   (reverse of each other)**

Three main low-pass filters are:

- Ideal high-pass filter (ILPF)
- Butterworth high-pass filter
- Gaussian high-pass filter

121

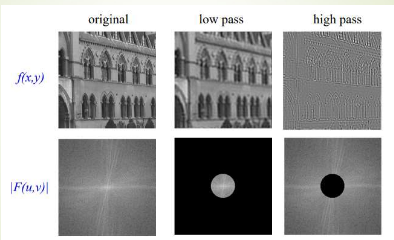
Ideal High-pass filters



Results of ideal high pass filtering with D_0 = 15 | Results of ideal high pass filtering with D_0 = 30 | Results of ideal high pass filtering with D_0 = 80

122

Action of filters on a real image



123

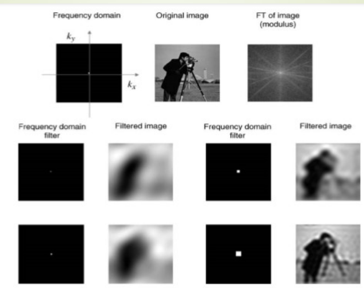


Figure 5.7 A basic illustration of frequency-domain filtering. The Fourier transform of the original image is multiplied by the filters indicated above (white indicates spatial frequency pairs which are preserved and black indicates total removal). An inverse Fourier transform then returns us to the spatial domain and the filtered image is displayed to the right. (The Matlab code for this figure can be found at http://www.fundipbook.com/materials/.)

124

The convolution theorem

$$\mathbf{F}\{f(x,y) ** h(x,y)\} = F(k_x, k_y)H(k_x, k_y) \quad (5.14)$$

The Fourier transform of the convolution of the two functions is equal to the product of the individual transforms.

$$\mathbf{F}\{f(x,y)h(x,y)\} = F(k_x, k_y) ** H(k_x, k_y) \quad (5.15)$$

The Fourier transform of the product of the two functions is equal to the convolution of their individual transforms.

125

Homomorphic filtering

- Removing multiplicative noise. $\tilde{I}(x,y) = I(x,y)\, n(x,y)$.
- Most commonly used for correcting non-uniform illumination in image
- According to illumination-reflectance model :
  Intensity at any pixel , I(x,y)(which is the amount of light reflected by a point on the object) is the product of the **illumination of the scene L(x,y)** and the **reflectance of the object**(s) R(x,y) in the scene, i.e.,

$$I(x,y) = L(x,y)\, R(x,y)$$

R arises from the properties of the scene and
L results from the lighting conditions at the time of image capture.

126

Homomorphic filtering

- To **compensate for the non-uniform illumination**: **remove the L** component and **keep only the R component**.
- If we consider illumination as the noise signal (which we want to remove), this model is similar to the multiplicative noise model shown earlier.
- Illumination typically varies slowly across the image and reflectance which can change quite abruptly at object edges.

This difference is the key to separating out the illumination component from the reflectance component.

127

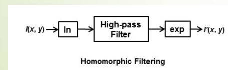
Homomorphic filtering

- In homomorphic filtering we first transform the multiplicative components to additive components by moving to the log domain:

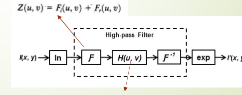
$$\ln(I(x,y)) = \ln(L(x,y)\,R(x,y))$$

$$\ln(I(x,y)) = \ln(L(x,y)) + \ln(R(x,y))$$

- Use a high-pass filter in the log domain to remove the low-frequency illumination component while preserving the high-frequency reflectance component.



$$Z(u,v) = F_i(u,v) + F_r(u,v)$$

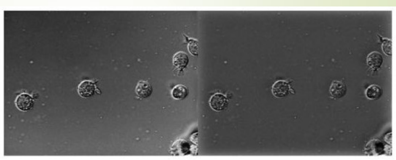


$$H(u,v)F_i(u,v) + H(u,v)F_r(u,v)$$

128

Homomorphic filtering

Before After



129

THANK YOU

130