

Image Analysis

(SE3IA11 + SEMIP12)

Lectured by

Dr Hong Wei (HW) and Prof. James Ferryman (JMF)

h.wei@reading.ac.uk; j.m.ferryman@reading.ac.uk

Room 145 (HW) and Room 123 (JMF)
the SSE Building

Autumn-term 2015

SE31A11+SEMIP12 Image Analysis

1

Reference books

- Rafael Gonzalez & Richard Woods, *Digital Image Processing*, 2nd edition, Prentice Hall, 2002
 - S G Hoggar, *Mathematics of Digital Images – Creation, Compression, Restoration, Recognition*, Cambridge Univ. Press, 2006
 - Anil K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, 1989
 - Milan Sonka, Vaclav Hlavac, and Roger Boyle, *Image Processing, Analysis, and Machine Vision*, Thomson Learning, 2007

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

2

Recommended Journals

- *IEEE Transaction on Image Processing*
 - *Image and Vision Computing*
 - *IEEE Transaction on Pattern Analysis and Machine Intelligence (PAMI)*
 - Many conference proceedings, e.g.
 - International Conference of Image Processing (ICIP)
 - International Conference of Pattern Recognition (ICPR)
 - International Conference of Computer Vision (ICCV)
 - Computer Vision and Pattern Recognition (CVPR)
 - British Machine Vision Conference (BMVC)
 - More

Autumn term 2015

SE31A11+SEMIR12 Image Analysis

3

Arrangements for the module

- This is a 10 credit module running through the autumn term. Two lectures a week.
 - 14:00-16:00 Fridays for 10 weeks (HARB164)
 - Week 3 is used for a tutorial in a lab (SSE G21)
 - Week 11 is for a demonstration in a lab (SSE G21)
- Two pieces of coursework associated with lab practicals (unsupervised)
 - Assignment 1: image enhancement (15% assessment)
 - Assignment 2: image compression (15% assessment)
- One 2-hour written examination in May/June 2016. It counts 70% of the final assessment.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

4

Contents covered in this module

- Fundamentals of digital images
- Image enhancement
- Colour image processing
- Image compression
- Mathematical morphology in image processing
- Image segmentation

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

5

Detailed scheduling

Week	Topic	Lecturer
Week 1 (2 nd October)	Fundamentals of Image Analysis	HW
Week 2 (9 th October)	Image Enhancement (Spatial)	JMF
Week 3 (16 th October)	Tutorial (MatLab/OpenCV)	HW
Week 4 (23 rd October)	Image Enhancement (Frequency)	JMF
Week 5 (30 th October)	Colour Image Processing	HW
Week 7 (13 th November)	Image Compression (IC)	JMF
Week 8 (20 th November)	IC2 / Symbolic Feature Extraction	JMF
Week 9 (27 th November)	Morphological Image Processing	HW
Week 10 (4 th December)	Image Segmentation	HW
Week 11 (11 th December)	Compression coursework assessment	HW&JMF

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

6

Digital image analysis

- Questions to be answered in this module
 - What is digital image?
 - How is a digital image formed mathematically?
 - Are there any applications where image analysis is required?
 - How to make the applications feasible?
 - Various techniques and processes

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

7

Applications of image analysis (1)

- Medical and biomedical images



(a) Gastric body from endoscope



(b) Hip joints from X-ray



(c) Cells from microscope

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

8

Applications of image analysis (2)

- Further medical images



(a) MRI of human spine



(b) Ultrasound image of a baby

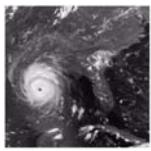
Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

9

Applications of image analysis (3)

- Remotely sensed data in climate and environment research – observation



(a) Multispectral image of hurricane (from GEO Satellite)



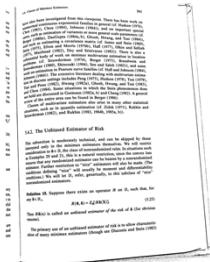
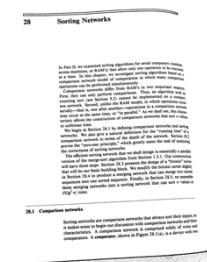
(b) SAR image: surface of Venus (from Magellan probe)



(c) LIDAR data (from airborne laser scanning)

Applications of image analysis (4)

- Image-based document analysis



Applications of image analysis (5)

- Security (CCTV): visual surveillance
- Webcam application: looking after elders through mobile phones
- London congestion control: automatic number plate recognition (ANPR)
- Biometrics: face images, finger prints, iris images,...
- Information retrieve in the Internet: image based information search (Google Goggles).
- The [Google Book Search](#) project unveiled in late 2004 aims to make all the world's books discoverable online.
- And more

Typical processes in image analysis
(from perception to cognition)

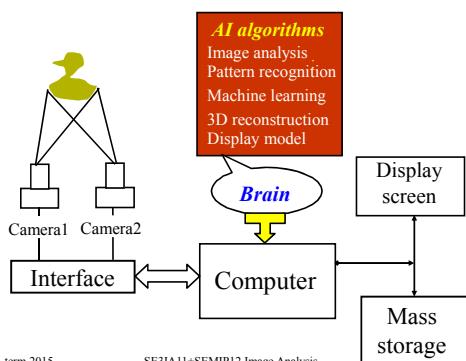
- Image acquisition
- Image enhancement and restoration
- Image compression
- Morphological processing
- Image segmentation and description
- Recognition: understanding and interpretation (intelligent?)

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

13

A computer imaging system



Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

14

Image acquisition: lighting
(electromagnetic radiation – EMR)

- “Light” in imaging covers a part of the electromagnetic spectrum beyond the visible spectrum (wavelength 380-760 nm).
- Light has two forms of energy: electricity and magnetism which are transmitted together as EMR in the speed of $2.998 \times 10^8 \text{ m/s}$.
- The electromagnetic spectrum can be expressed in terms of wavelength, frequency (or energy), and speed as

$$\lambda = \frac{c}{\nu} \quad \text{and} \quad \text{energy } E = h\nu \quad \text{with } h \text{ a constant.}$$

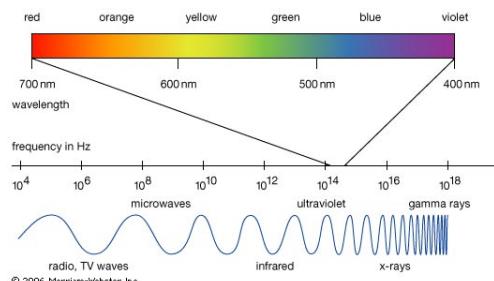
where λ is wavelength, ν frequency, and c speed.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

15

The electromagnetic spectrum



Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

16

Image acquisition: sensors (1)

- Imaging is a process in which the reflected energy from illuminated scenes is received (or collected) by sensing elements or materials, called sensors.
- Four components: light source, scene, **sensors**, and **lens – optical systems**.
- Sensors must be responsive to light sources, *e.g.* X-ray, infrared, visible lights, etc. by using different sensing materials (*e.g.* different CCD or CMOS).
- Sensing: transform incoming illumination energy (reflected from scene) into a voltage by sensors, from which the output voltage is digitised to generate a value of a pixel in a digital image.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

17

Image acquisition: sensors (2)

- Three principal sensor arrangements
 - *Single sensor*: sensor moving, *e.g.* raster scanning
 - *Line sensor*: sensor strip in linear motion
 - *Sensor arrays*: popularly used in normal digital cameras and cam-recorders.
- Outputs of all the above different sensors could be arranged to a 2D array reflecting illumination energy of a scene.

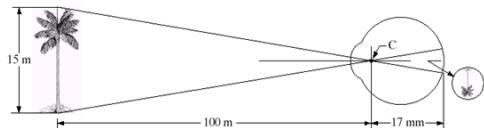
Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

18

Image acquisition: optical systems

- An optical system (lens) plays the role to refract an observed scene onto an imaging plane (i.e. a focal plane in the pin-hole model).



C is the optical centre of the lens.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

19

PSF and MTF in an optical system

- PSF: the *point spread function* indicates an optical system response to an impulse $\delta(x, y)$.
 - All physical optical systems blur (spread) a point of light to some degree. After an optical system, $\delta(x, y)$ becomes to $h(x, \alpha, y, \beta) = H[\delta(x - \alpha, y - \beta)]$, mathematically called convolution, where α and β are parameters related with the optical system.
- MTF: the *modulation transfer function* is defined as

$$MTF(f_s) = \frac{I_{\max} - I_{\min}}{I_{\max} + I_{\min}}$$

where I_{\max} and I_{\min} are maximum and minimum grey values of pixels in an image, respectively. MTF is a function of spatial frequency f_s . It is affected by the arrangement of sensor arrays.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

20

Representation of digital images

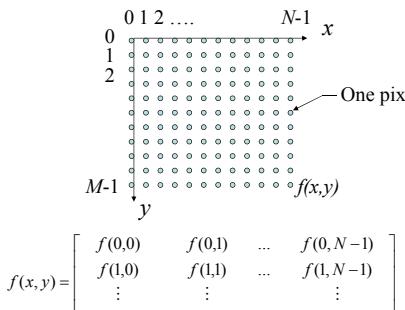
- A digital image is normally represented by a two-dimensional (2D) array, e.g. $f(x, y)$, where $x=1, 2, \dots, M$ and $y=1, 2, \dots, N$, representing the size of the image.
- Each element in the array is a real number, called a **pixel** and represented by a finite number of bits.
- A matrix denotes a digital image in the formal mathematic term.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

21

Digital image: matrix expression



Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

22

Image sampling and quantisation (1)

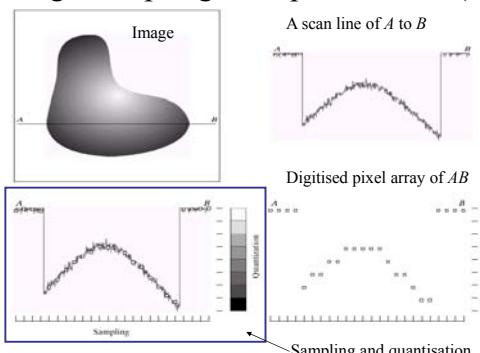
- Image sampling deals with issues of how many pixels are used to represent a scene.
 - Digitising the coordinate values: the Nyquist sampling theorem is applied.
 - The Nyquist states that the minimum sampling frequency is twice as high as the signal frequency.
- Image quantisation regards issues of how many bits used for one pixel.
 - Digitising the amplitude values of illumination energy: the visual quality should be satisfied.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

23

Image sampling and quantisation (2)



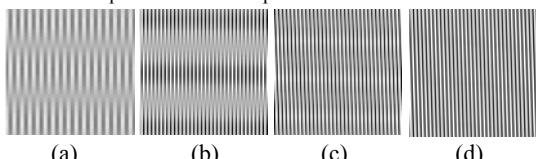
Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

24

Example of image sampling (1)

- Sampling black and white bars with 100 lines/inch.
 - 100 dpi (0.5 Nyquist-Shannon sampling rate)
 - 200 dpi (with Nyquist-Shannon sampling rate)
 - 300 dpi (1.5 Nyquist-Shannon sampling rate)
 - 600 dpi (3 Nyquist-Shannon sampling rate)
 - where dpi stands for “dots per inch”.



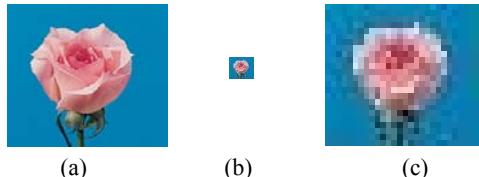
Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

25

Example of image sampling (2)

- Pink rose in colours: details are lost in (c).
 - 599×812×3 uint8
 - 82×111×3 uint8
 - The image in (b) re-sampled into 599×812.

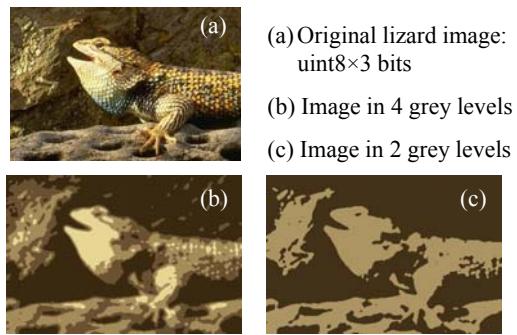


Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

26

Example of image quantisation



Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

27

Image resizing

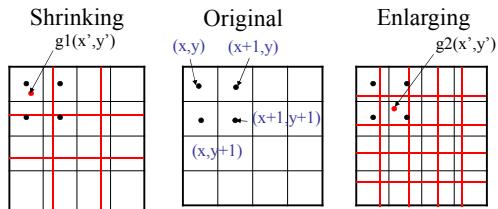
- Shrinking digital images: operation of undersampling, e.g. $512 \times 512 \Rightarrow 256 \times 256$
- Enlarge digital images: operation of oversampling, e.g. $256 \times 256 \Rightarrow 512 \times 512$
- Two main steps in the operations
 - Creation of new pixel locations
 - Assignment of grey levels to those new locations
- Techniques used in the image resizing operation
 - Nearest neighbour interpolation (NNI)
 - Bilinear interpolation (BLI)

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

28

Image resizing: interpolation (NNI)



- For NNI: $g1(x',y') = g(x,y)$; $g2(x',y') = g(x+1,y+1)$
- How about $g1$ and $g2$ based on the BLI algorithm?

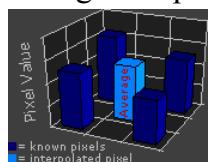
<http://www.cambridgeincolour.com/tutorials/image-interpolation.htm>

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

29

Image resizing: interpolation (BLI)



- Bilinear interpolation considers the closest 2×2 neighbourhood of known pixel values surrounding the unknown pixel.
- It takes a weighted average of these 4 pixels to arrive at its final interpolated value.
- This algorithm results in much smoother looking images than NNI.

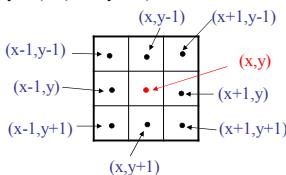
Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

30

Relationships between pixels (1)

- Neighbours of a pixel $p(x,y)$
 - 4-neighbours $N_4(p)$: $(x-1,y)$, $(x+1,y)$, $(x,y-1)$, $(x,y+1)$
 - Four diagonal neighbours $N_D(p)$: $(x-1,y-1)$, $(x-1,y+1)$, $(x+1,y-1)$, $(x+1,y+1)$



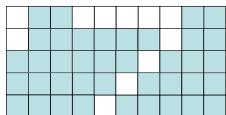
Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

31

Relationships between pixels (2)

- Regions and boundaries: binary images have two sets $\{1\}$ and $\{0\}$
 - **Connectivity**: two pixels are neighboured, but they are connected only if they both belong to set $\{1\}$ or set $\{0\}$.
 - **Connected components**: for any pixel p in set S , the set of pixels are connected to p in S is called a connected component in S .



$S\{1\}$: blue
 $S\{0\}$: white

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

32

Relationships between pixels (3)

- Distance between pixels $p(x_1,y_1)$ and $q(x_2,y_2)$
 - The Euclidean distance
 - The D_4 distance: $D_4 = |x_1 - x_2| + |y_1 - y_2|$
 - The D_8 distance: $D_8 = \max(|x_1 - x_2|, |y_1 - y_2|)$
- Image operations on a pixel basis
 - Each pixel represents an element in a matrix.
 - Image operations follow operations among matrices.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

33

Image quality evaluation

- Two types of criteria for image evaluation: subjective and quantitative.
 - Quantitative evaluation of image $f(x,y)$:
 - A reference image $f'(x,y)$ is needed.
 - The average mean square error is used for the evaluation as
- $$\sigma^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} |f(x,y) - f'(x,y)|^2$$
- Image fidelity criteria are useful for
 - Measuring image quality
 - Rating the performance of a processing techniques.

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

34

Questions for further thinking

- How to express a digital image mathematically ?
- Issues of digital image visualisation (sampling and quantisation)
- Based on the concept of image resizing, work out how to deal with image rotation (matrix rotation) and shape distortion.
- Image quality evaluation: try to think about more efficient measures for the purpose. Do we need benchmarks for the purpose? If so, how to establish the benchmarks (with assumptions)?

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

35

End of the first two lectures

- Summary of what you have learned in the two lectures.
- Revision on matrix operations

Autumn-term 2015

SE3IA11+SEMIP12 Image Analysis

36

SE3IA11/SEMIP12 Image Analysis



Image Enhancement in the Spatial Domain

Lecturer:

Prof. James Ferryman, Computational Vision Group

Email: j.m.ferryman@reading.ac.uk

Introduction

- The purpose of image enhancement is to emphasise some selected (desirable) image features and suppress others (undesirable features)
- Precursor to information extraction
- The desirability of image features is application/task dependent
 - E.g. for human observers: to improve the interpretability or perception of detail
 - E.g. for automated image processing: to provide “enhanced” input
- Hence, there exists no formal definition of image enhancement
- Process is usually interactive and iterative

Introduction

- Image enhancement approaches fall into 2 broad categories
 - Spatial domain methods
 - Refers to the image plane itself, and operate directly on the image pixels
 - Frequency domain methods
 - Operate on the Fourier transform of an image
- In this lecture, we will concentrate on the spatial domain, with frequency domain methods covered in the next lecture

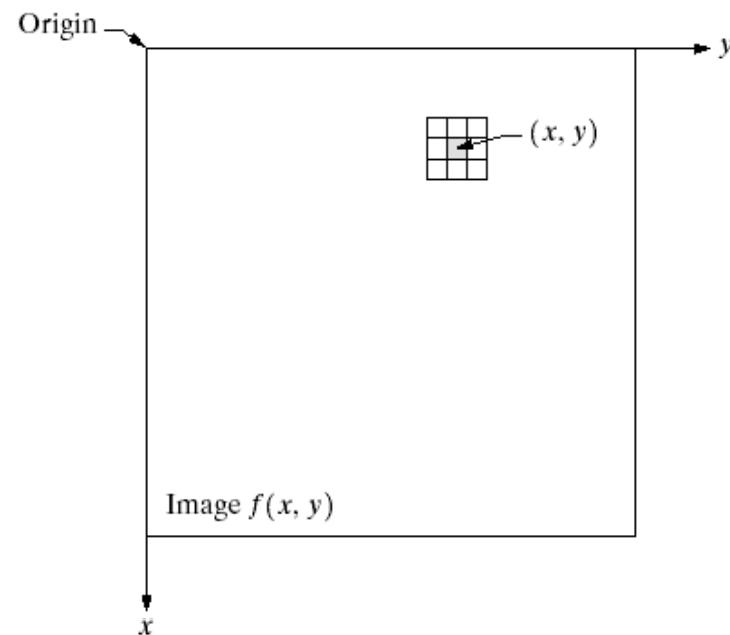
Preliminaries

- The term *spatial domain* refers to an aggregate of pixels composing an image
- Spatial domain methods operate directly on these pixels
 - Denoted by $g(x, y) = T[f(x, y)]$
where $f(x, y)$ is the input image, $g(x, y)$ is the processed image, and T is an operator defined some neighbourhood of (x, y)

T can also operate over a set of input images (e.g. pixel by pixel sum of K images for noise reduction)

Preliminaries

- The main approach to defining a neighbourhood about a pixel (x,y) is to use a square (or rectangular) subimage centered at (x,y)



Preliminaries

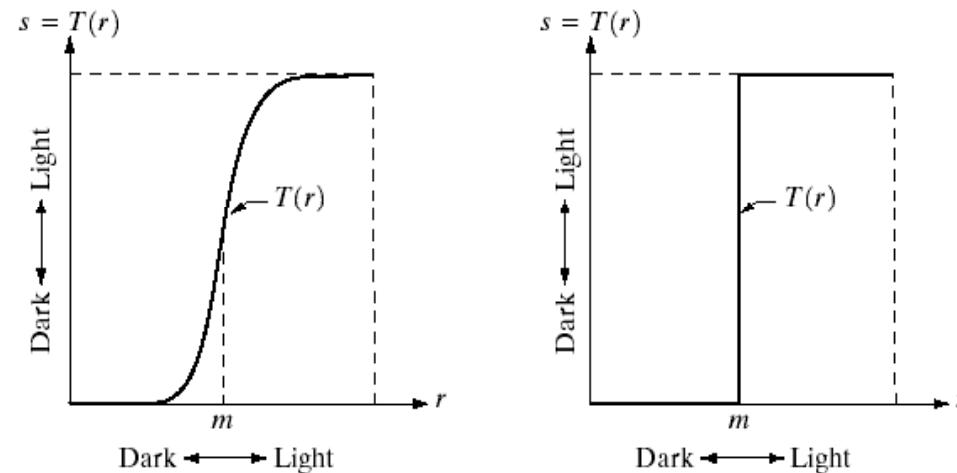
- The centre of the subimage is moved from pixel to pixel starting, e.g. at top left hand corner
- Operator T is applied at each location (x,y) to obtain the output, g , at that location
- Process uses only the pixels in the area of the image spanned by the neighbourhood
- Square and rectangular arrays are easiest to implement

Introduction to Grey Level Transforms

- Simplest form of T is when neighbourhood is 1x1 (i.e. single pixel): T becomes a grey level transformation function of the form

$$s = T(r)$$

- where r and s are respectively the grey level of $f(x,y)$ and $g(x,y)$ at any point (x,y)



Grey Level Transformations

- Among the simplest of all enhancement techniques
- Assume $s = T(r)$, where r and s are value of pixels before and after processing
- Values of transformation function $T(r)$ are typically stored in 1D array, and mappings from $r \rightarrow s$ are implemented via table lookups
 - For an 8-bit image, lookup table contains 256 entries
- 3 basic types of functions used frequently for enhancement are:
 - Linear (negative and identity transformations)
 - Logarithmic (log and inverse-log transformations)
 - Power-Law (n th power and n th root transformations)

Identity function is simplest: output intensities identical to input

Grey Level Transformations

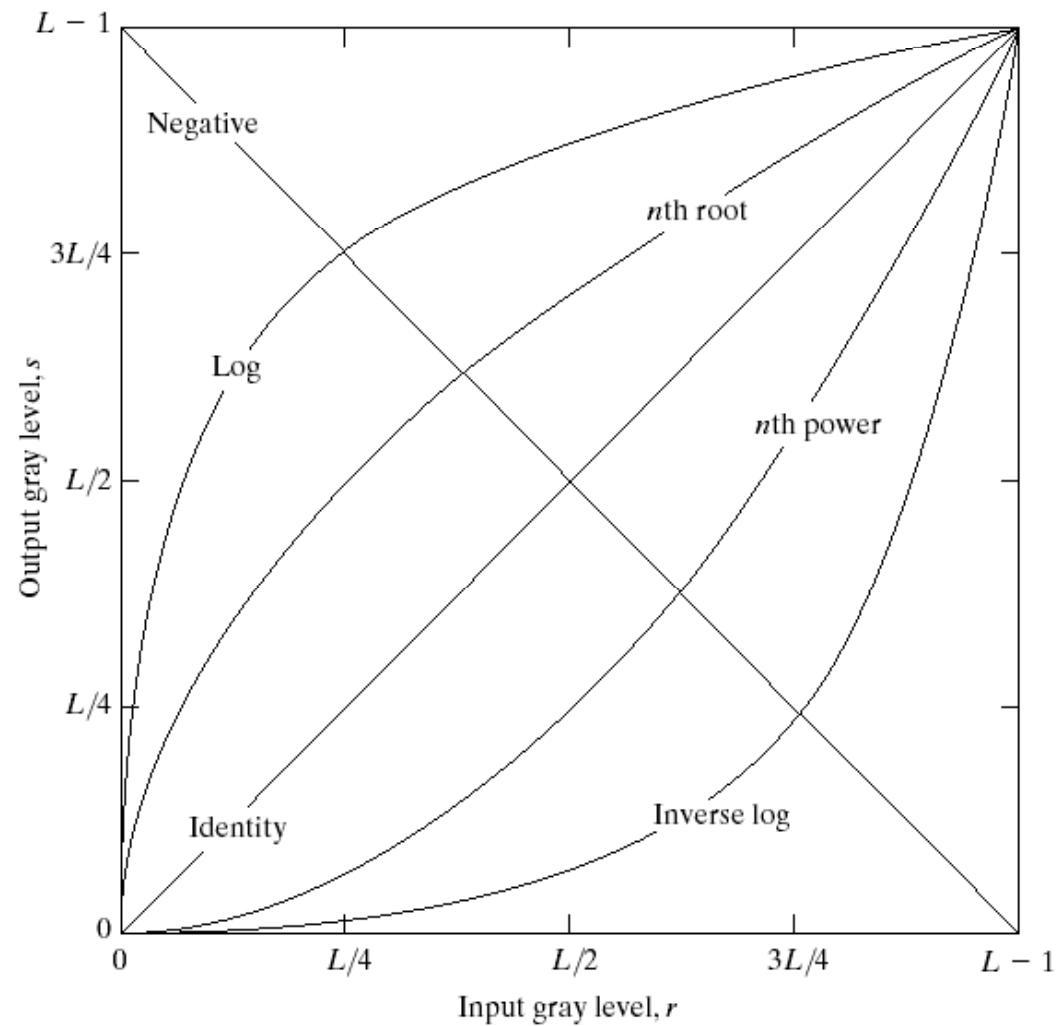
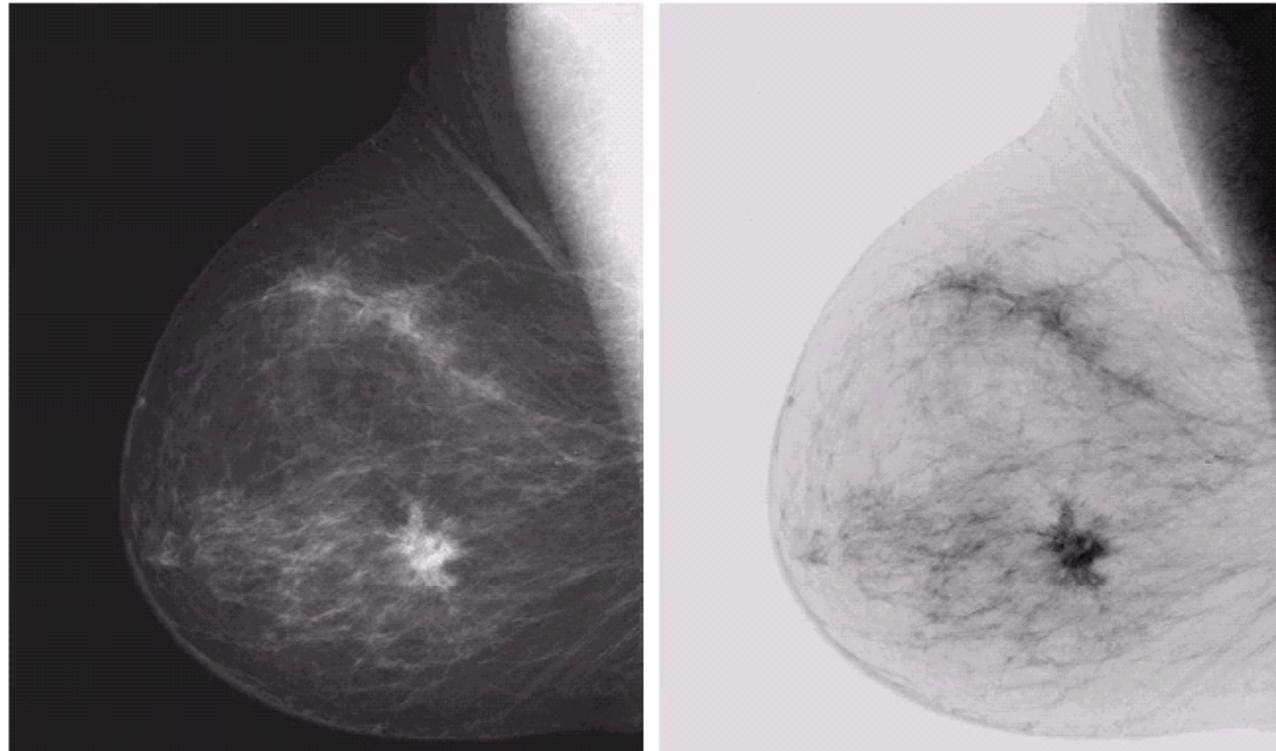


Image Negatives

- The negative of an image with grey levels in the range $[0, L-1]$ is obtained by using negative transformation (see previous slide) and given by the expression $s = L - 1 - r$
- Reversing the intensity levels produces equivalent of photographic negative
- Processing is especially suited for enhancing white or grey level detail embedded in dark image regions

Image Negatives

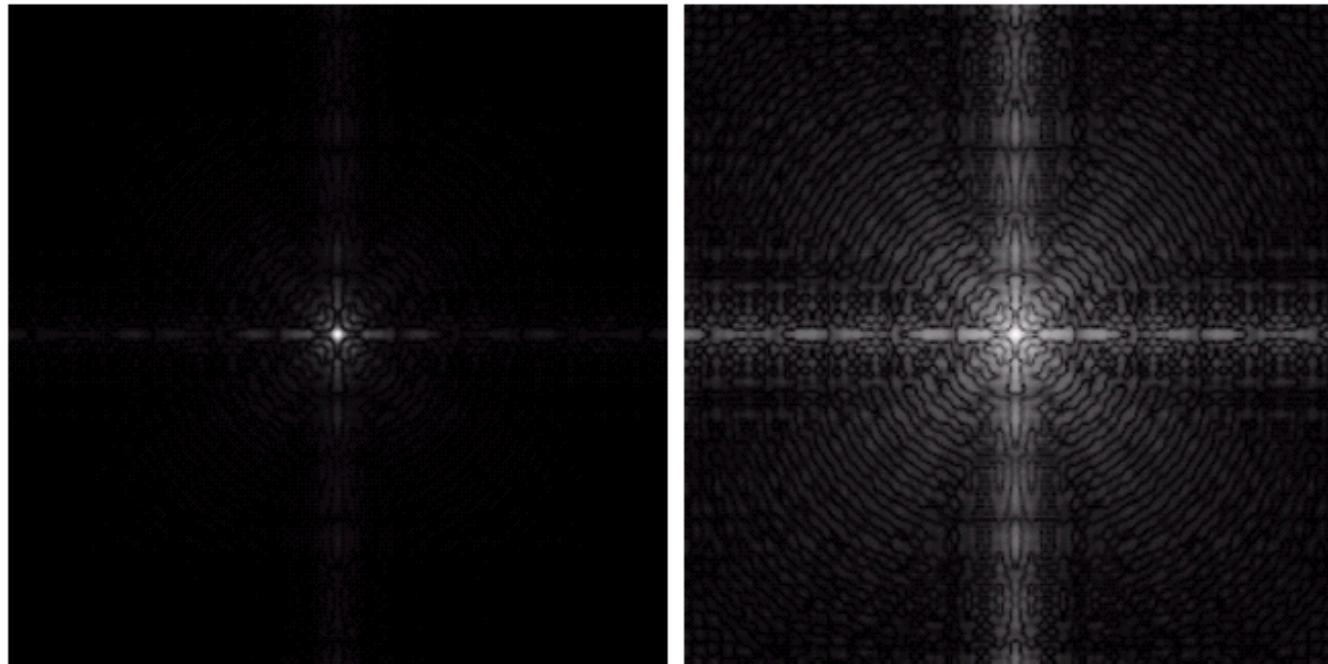


- Original image is digital mammogram showing small legion
- Note that while visual content is same in both images, it is easier to analyse the breast tissue in the negative image (in this case)

Log Transformations

- The general form of log transformation is
$$S = c \log (1 + r)$$
where c is a constant, and it is assumed that $r \geq 0$
- The transformation maps a narrow range of low greylevel values in input image into a wider range of output levels; opposite true of higher values of input levels
- Transformation is used to expand values of dark pixels, while compressing higher-level values
 - Compresses dynamic range of images with large variations in pixel values
- Inverse log transformation performs the opposite

Log Transformations

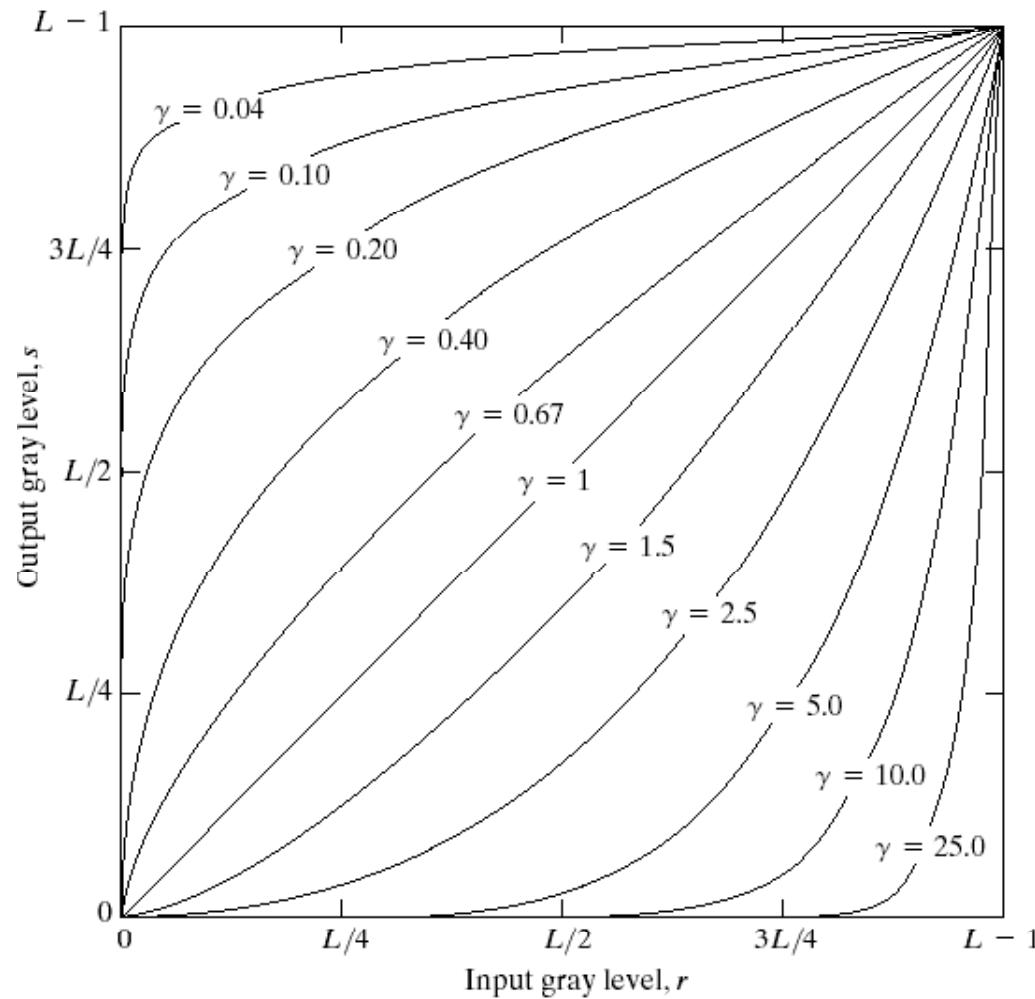


- Figure (left) shows a Fourier spectrum (next lecture) with values in range 0 to 1.5×10^6 : brightest pixels dominate
- Post linear scaling ($c = 1$) (right image) visible detail is enhanced
- Most Fourier spectra seen in image processing are scaled in this way

Power-Law Transformations

- Power-law transformations have basic form $s = cr^y$ where c and y are positive constants
- As for log transformation, power-law curves with fractional values y map a narrow range of dark input values into a wider range of output values
- Unlike log function, a *family* of transformation curves can be produced, by varying y
- Many devices used for capture, printing, display of images respond according to power law
 - Exponent in power-law equation is referred to as *gamma*
 - Process used for correction is called gamma correction

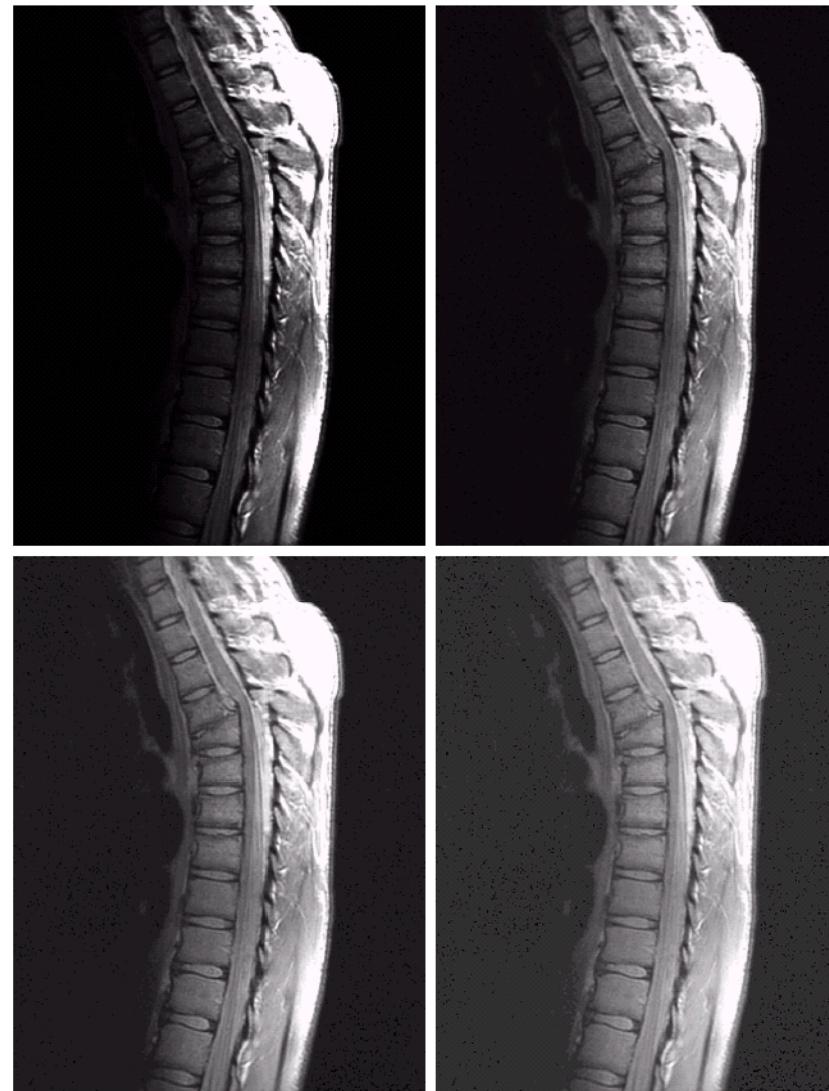
Power-Law Transformations



Power-Law Transformations

- Power-law functions are useful for general-purpose contrast manipulation
- Following figure is a magnetic resonance (MR) image of an upper thoracic human spine with fracture dislocation
 - Fracture is visible near vertical centre of spine
 - As image is dark, expansion of grey levels is desirable
 - Accomplished with power-law transformation
 - $c = 1$ fixed; y varies from 0.6, 0.4, 0.3

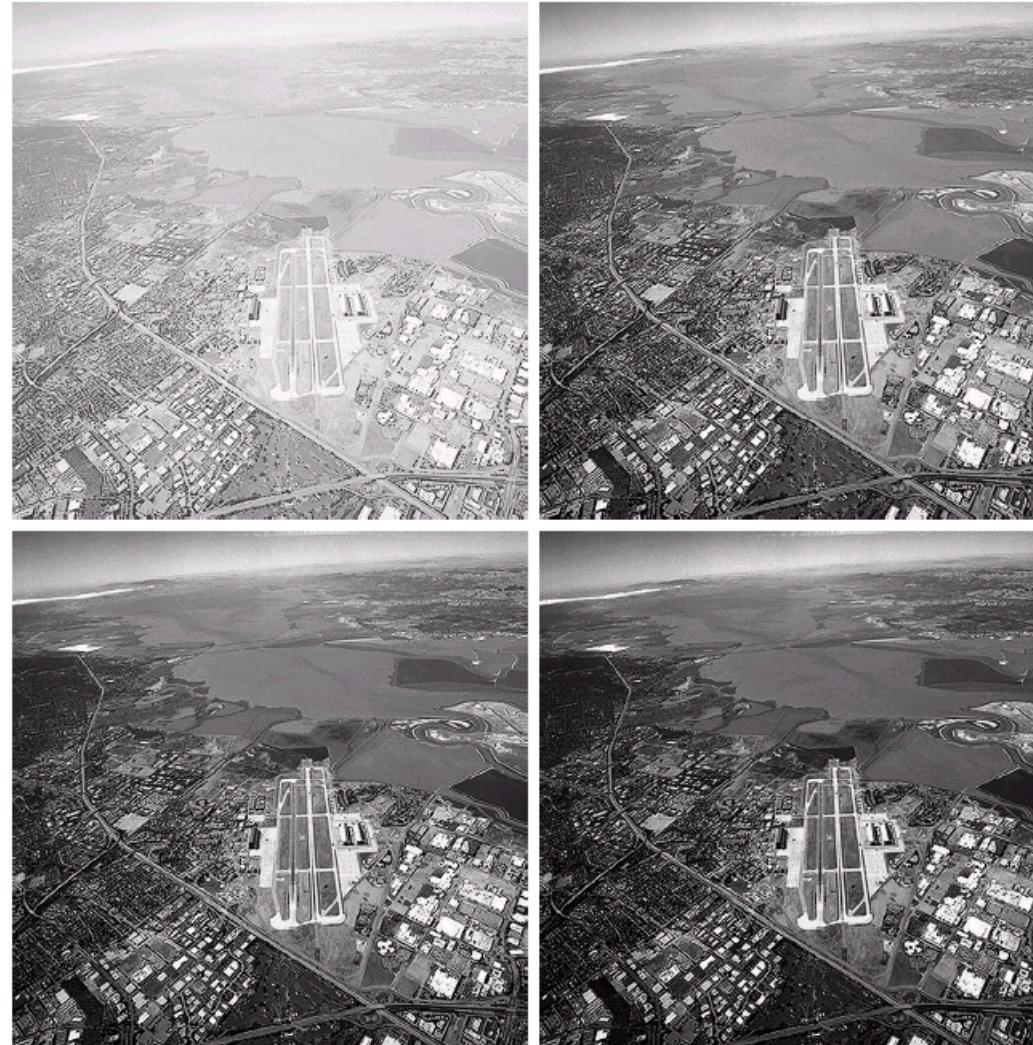
Power-Law Transformations - Example



Power-Law Transformations - Example

- The next figure illustrates the opposite problem to the previous example
- The image has a washed-out appearance
 - Indicates that a compression of grey levels is desirable
 - Can be achieved using power-law transform with $\gamma > 1$
- Images show processing with $y = 3.0, 4.0, 5.0$

Power-Law Transformations - Example



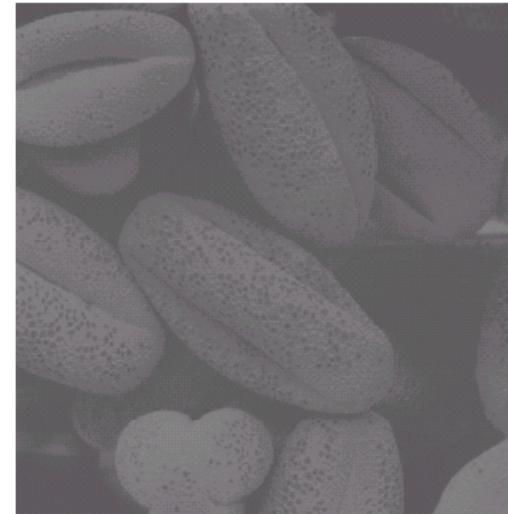
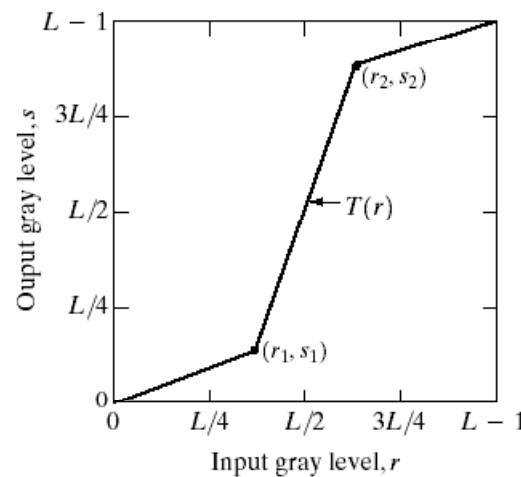
Piecewise-Linear Transformations

- Complementary approach to previous methods
- Main advantage is that form of piecewise functions can be arbitrarily complex
- Main disadvantage is that their specification requires more user input

Contrast Stretching

- One of simplest piecewise linear functions
- Idea is to increase dynamic range of image grey levels
- Main application is low contrast images
 - Due to poor illumination, lack of dynamic range in sensor, wrong settings during image capture, ...
- Example shows typical transformation used
- Locations of points (r_1, s_1) and (r_2, s_2) control shape of transformation

Contrast Stretching - Example



$(r_1, s_1) = (r_{\min}, 0)$ &
 $(r_2, s_2) = (r_{\max}, L-1)$,
 where r_{\min} & r_{\max} =
 min/max image
 greylevels

$r_1 = r_2 = m$
 (mean image
 greylevel)

Contrast Stretching

- If $r_1 = s_1$ and $r_2 = s_2$, the transform is a linear function, producing no change
- If $r_1 = r_2, s_1 = 0$, and $s_2 = L - 1$, transform is a thresholding function
- Intermediate values of (r_1, s_1) and (r_2, s_2) produces varying degrees of spread in grey levels of output image, i.e. change in contrast
- In general, $r_1 \leq r_2$ and $s_1 \leq s_2$ is assumed so function is single valued and monotonically increasing
 - Preserves order of greylevels and prevents artifacts in output image

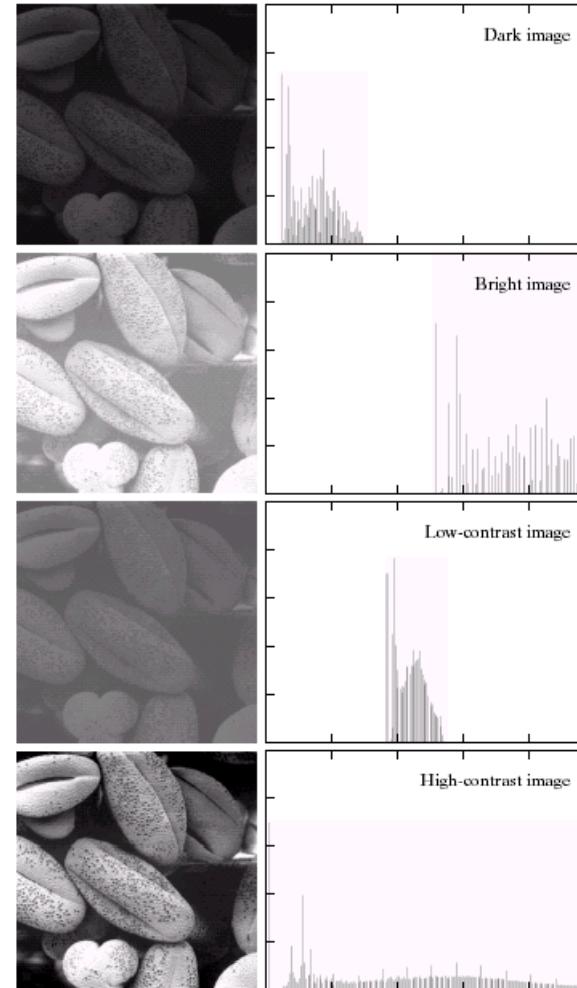
Greylevel & Bitplane Slicing

- Greylevel Slicing
 - Highlighting specific range of greylevels in image is often desired
 - One approach is to display high value for all greylevels in range of interest, and low value for all others
 - Another approach brightens desired range of grey levels but preserves background and tones in image
- Bitplane slicing
 - Instead of emphasising ranges, contribution made by specific bits to total image appearance, may be desired
 - Separating image into bit planes is useful for analysis of relative importance of each bit to an image, providing information on adequacy of number of bits used to quantize each pixel

Histogram Processing

- The histogram of an image with grey levels in range $[0, L-1]$ is a discrete function $h(r_k) = n_k$, where r_k is the k_{th} grey level and n_k is the number of image pixels having greylevel r_k
- It is usual practice to normalise a histogram by dividing each of the values by total number, N , of image pixels
 - Normalised histogram given by $p(r_k) = \frac{n_k}{N}$ for $k = 0, 1, \dots, L-1$
- $p(r_k)$ provides an estimate of probability of occurrence of greylevel r_k

Histogram Processing - Example



Histogram Equalisation

- Powerful enhancement method that seeks to optimise image contrast
 - Flattens (equalises) image histogram – redistribute greylevels uniformly
 - Non-linear
- Especially useful in images with large regions of similar appearance (e.g. light background, dark foreground)
- Can be used to expose hidden detail

(Discrete) Histogram Equalisation

- Uses the cumulative distribution function (CDF) as a lookup table
- CDF essentially answers: “what percentage of samples in image are equal to or less than value k ?
- The normalized CDF is defined as: $C_k = \sum_{i=0}^k p(r_i)$ where $p(r_k) = \frac{n_k}{N}$
- CDF is monotonically increasing
 - Derivative (slope) is steep where source image has much information, flat conversely
 - CDF of perfect equalised image is straight line, with slope = 1
- The equalised histogram can then be computed as:

$$s_k = (L-1) \sum_{i=0}^k p_r(r_i) \quad 0 \leq s_k \leq 1$$

The values of s_k will have to be scaled up by 255 and rounded to the nearest integer so that the output values of this transformation will lie in the range $[0, 255]$. Thus, the discretisation and rounding of s_k to the nearest integer will mean that the transformed image will not have a perfectly uniform histogram

Histogram Equalisation - Example

52	55	61	66	70	61	64	73
63	59	55	90	109	85	69	72
62	59	68	113	144	104	66	73
63	58	71	122	154	106	70	69
67	61	68	104	126	88	68	70
79	65	60	70	77	68	58	75
85	71	64	59	55	61	65	83
87	79	69	68	65	76	78	94



Value	Count								
52	1	64	2	72	1	85	2	113	1
55	3	65	3	73	2	87	1	122	1
58	2	66	2	75	1	88	1	126	1
59	3	67	1	76	1	90	1	144	1
60	1	68	5	77	1	94	1	154	1
61	4	69	3	78	1	104	2		
62	1	70	4	79	2	106	1		
63	2	71	2	83	1	109	1		



Value	cdf								
52	1	64	19	72	40	85	51	113	60
55	4	65	22	73	42	87	52	122	61
58	6	66	24	75	43	88	53	126	62
59	9	67	25	76	44	90	54	144	63
60	10	68	30	77	45	94	55	154	64
61	14	69	33	78	46	104	57		
62	15	70	37	79	48	106	58		
63	17	71	39	83	49	109	59		

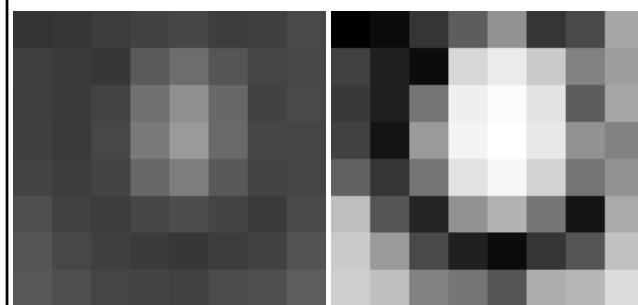
$$s_k = \text{round} \left[(L-1) \times \frac{cdf(k) - cdf_{\min}}{N - cdf_{\min}} \right]$$

$$s_{78} = \text{round} \left[255 \times \frac{cdf(k) - 1}{63} \right]$$

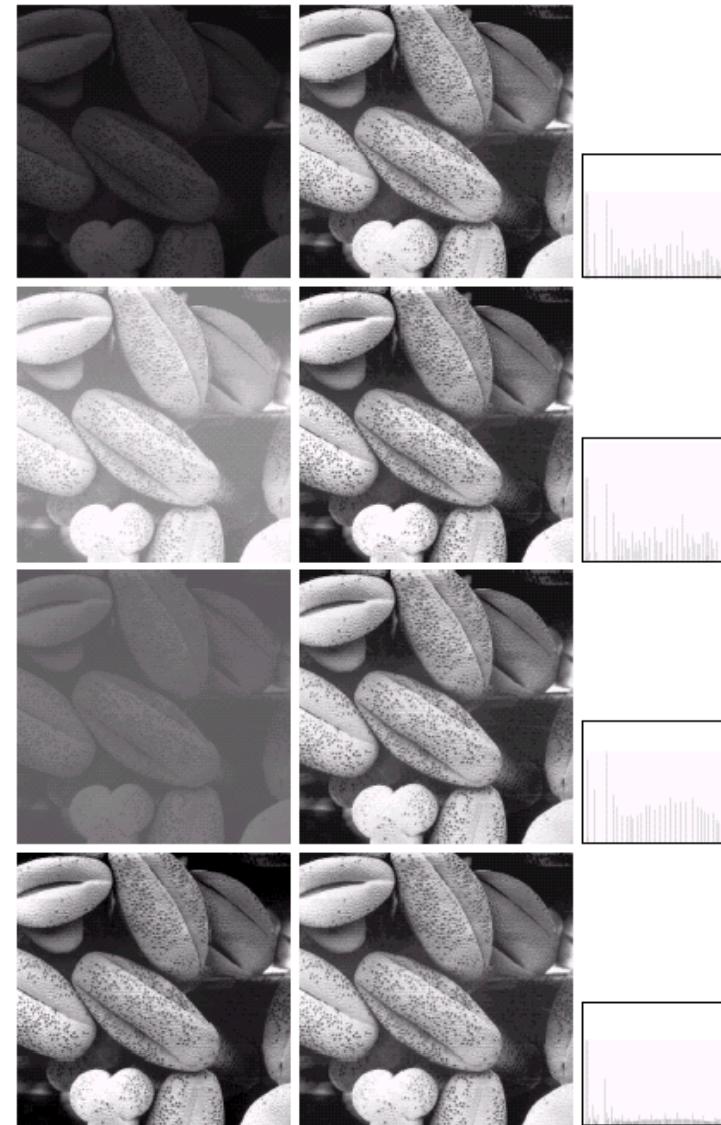
$$s_{78} = \text{round}(0.714286 \times 255) = 182$$



0	12	53	93	146	53	73	166
65	32	12	215	235	202	130	158
57	32	117	239	251	227	93	166
65	20	154	243	255	231	146	130
97	53	117	227	247	210	117	146
190	85	36	146	178	117	20	170
202	154	73	32	12	53	85	194
206	190	130	117	85	174	182	219



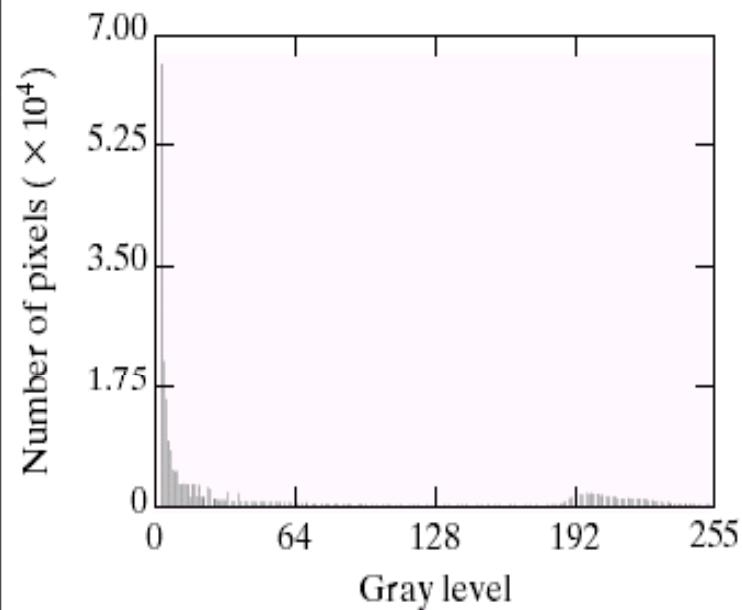
Histogram Equalisation



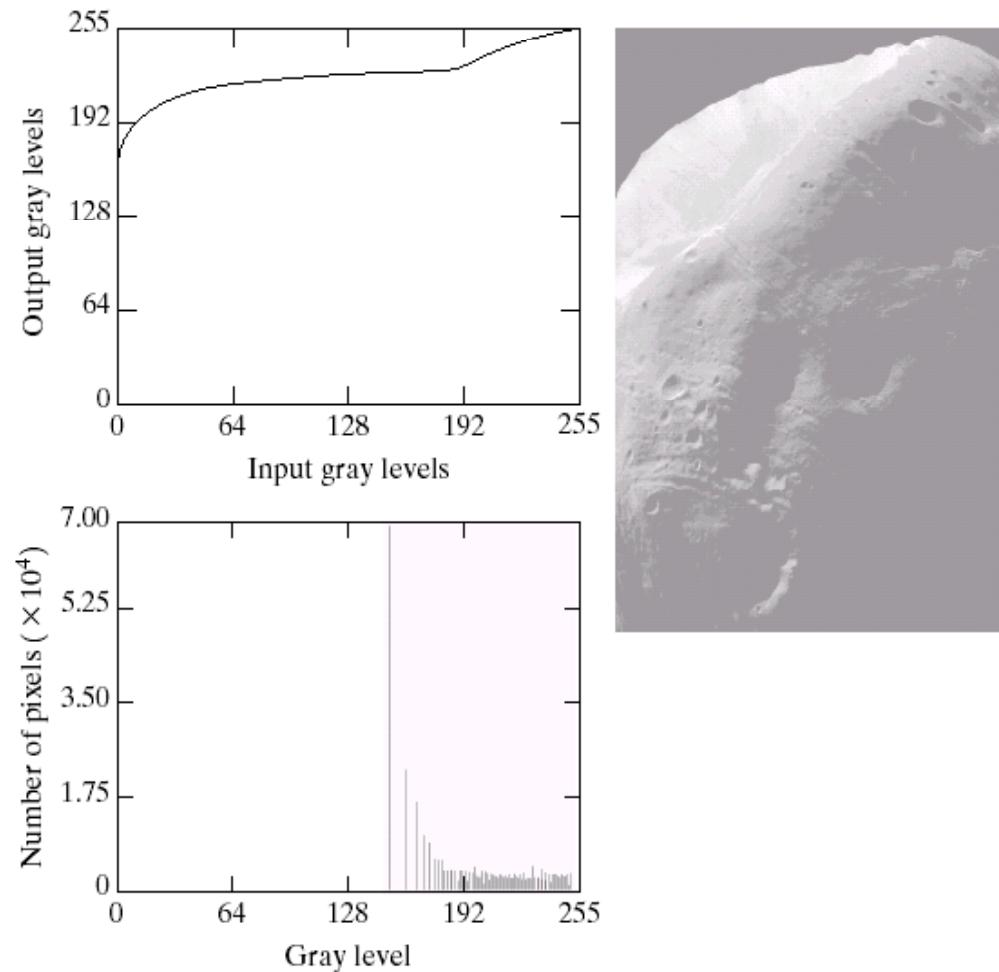
Equalisation: Summary

- Determines a transformation function that seeks to produce an output image with uniform histogram
- May be performed automatically
- Results are predictable
- Method is simple to implement
- However....
there exist applications in which enhancement based on a *uniform* histogram is not optimal; it is better to specify the *shape* of the histogram that the processed image should possess
- Method used to undertake this is called *histogram matching* (specification)

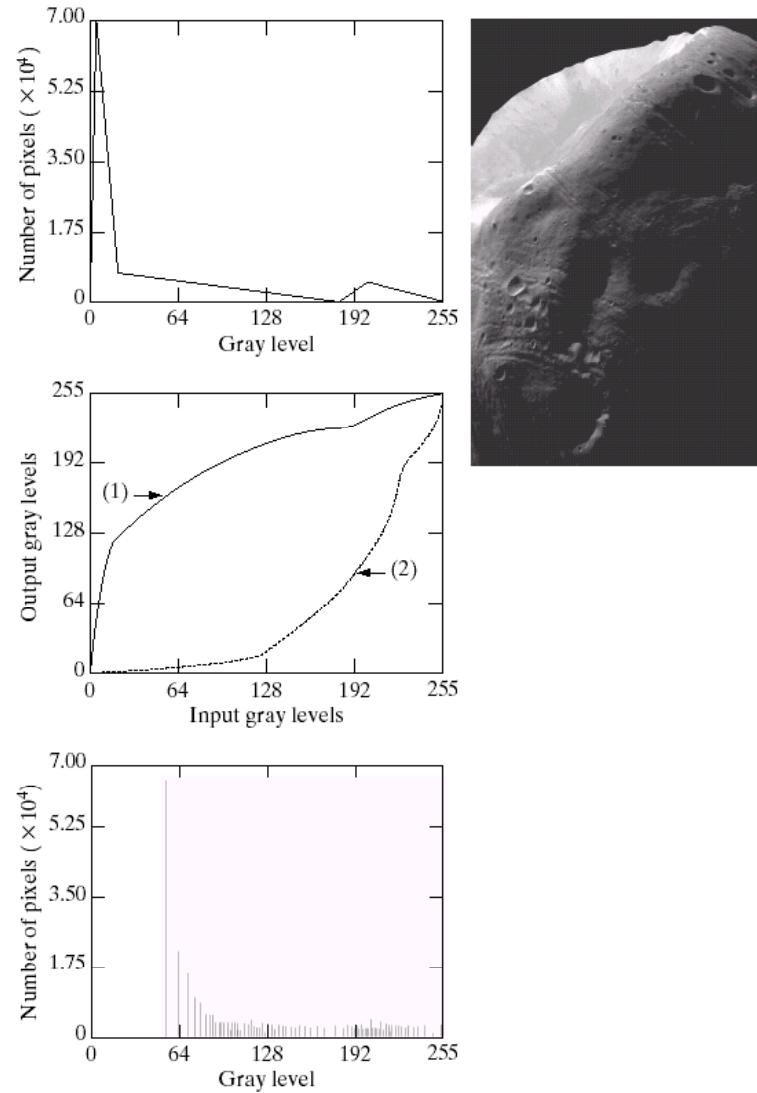
Histogram Matching



Histogram Matching



Histogram Matching



Histogram Matching

- For most part, a trial-and-error process
- Guidelines may be available for certain types of data
- There may exist cases in which an “average” histogram should look like and use this as specified histogram
- In general, however, there are no rules for specifying histograms
- One must resort to case-by-case analysis for enhancement

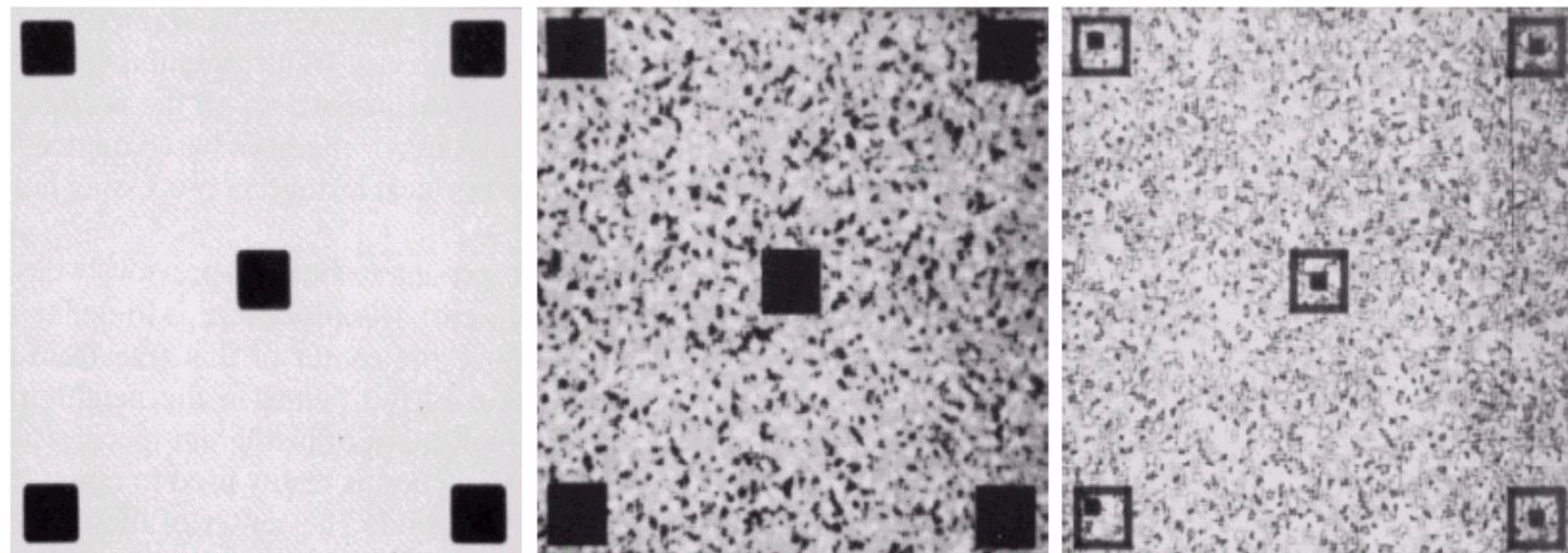
Local Enhancement

- Previous histogram methods are global
 - Pixels are modified by transformation function based on greylevel content of entire image
- Sometimes necessary to enhance detail over small image areas
- Solution is to devise transformation functions based on the greylevel distribution in the neighborhood of every pixel

Local Enhancement

- The procedure is as follows:
 - Define a square (or rectangular) neighborhood and move the centre of this area from pixel to pixel
 - At each location, the histogram of the points in the neighborhood is computed and either a histogram equalisation or histogram specification transformation function is obtained
 - This function is finally used to map the greylevel of the pixel centered in the neighborhood
 - The centre is then moved to an adjacent pixel location and the procedure repeated

Local Enhancement: Example



Arithmetic/Logic Operations

- Performed on a pixel-by-pixel basis between 2 or more images (except NOT on single image)
- Operations include:
 - Subtraction, Addition, Multiplication, Division
 - NOT
 - Used for negative transformation
 - AND, OR
 - Used for masking (selecting subimage/ROI within an image)

Image Subtraction

- The difference between two images $f(x,y)$ and $h(x,y)$, expressed as $g(x,y) = f(x,y) - h(x,y)$ is obtained by computing the difference between all pairs of corresponding pixels from f and h
- Provides enhancement of *differences* between images

Image Subtraction: Radiography

- Major application of image subtraction in medical imaging, specifically *mask mode radiography*
- In this case, $h(x,y)$, the mask, is an X-ray image of region of patient's body captured by intensified camera located opposite X-ray source
- Procedure involves injecting *contrast medium* into patient's bloodstream
- Series of images of same anatomical region as $h(x,y)$ captured
- Mask is subtracted from series of incoming images
- Areas different between $f(x,y)$ and $h(x,y)$ provide detail
- Overall effect is video showing how medium propagates through arteries in area under observation

Image Subtraction: Radiography

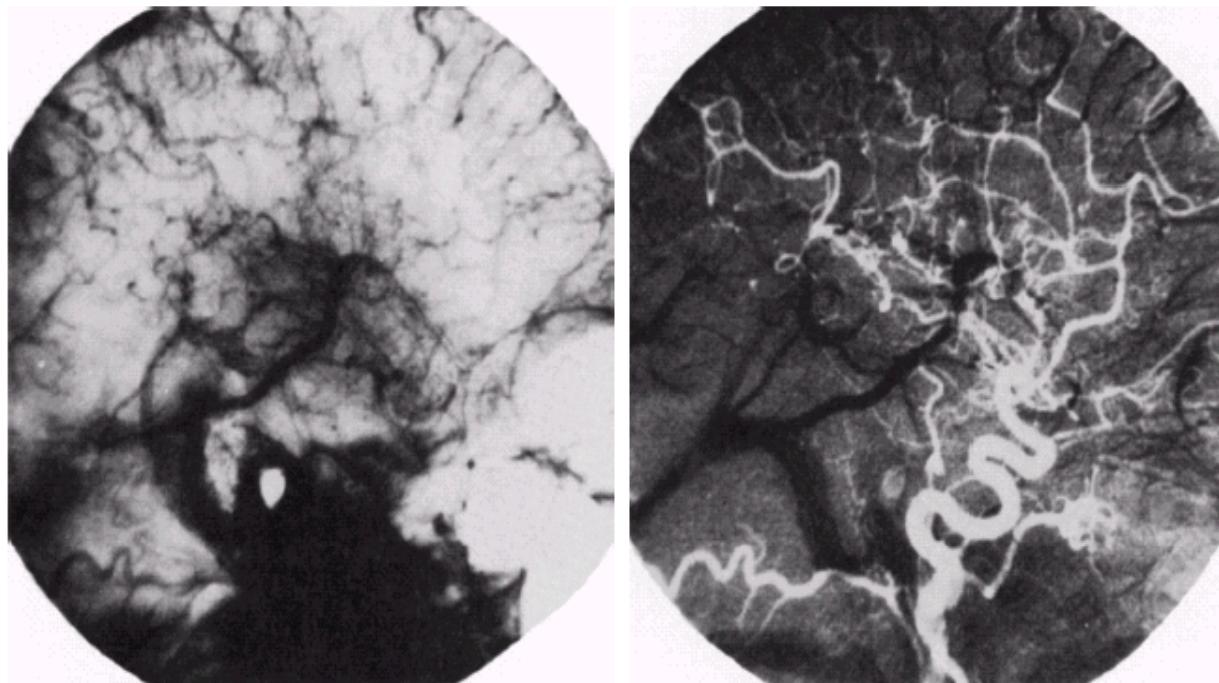


Image Averaging

- Consider a noisy image $g(x,y)$ formed by the addition of noise $n(x,y)$ to an original image $f(x,y)$

$$g(x, y) = f(x, y) + n(x, y)$$

where the assumption is the noise is uncorrelated and has zero average value

- Objective is to reduce noise content by adding a set of noisy images $\{g_i(x,y)\}$
- Averaging M different noisy images

$$\bar{g}(x, y) = \frac{1}{M} \sum_{i=1}^M g_i(x, y)$$

Image Averaging

- As M increases, the variability of the pixel values at each location decreases
 - This means that $g(x,y)$ approaches $f(x,y)$ as the number of noisy images used in the averaging process increases
- Registering (alignment) of the images is necessary to avoid blurring in the output image

Image Averaging

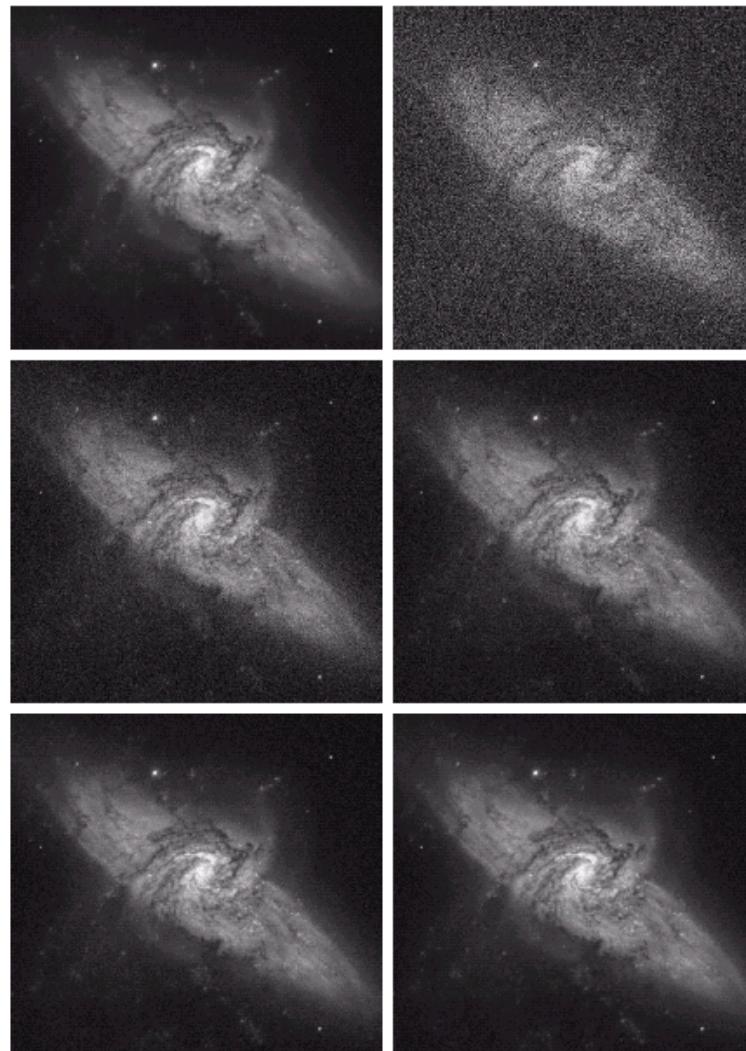
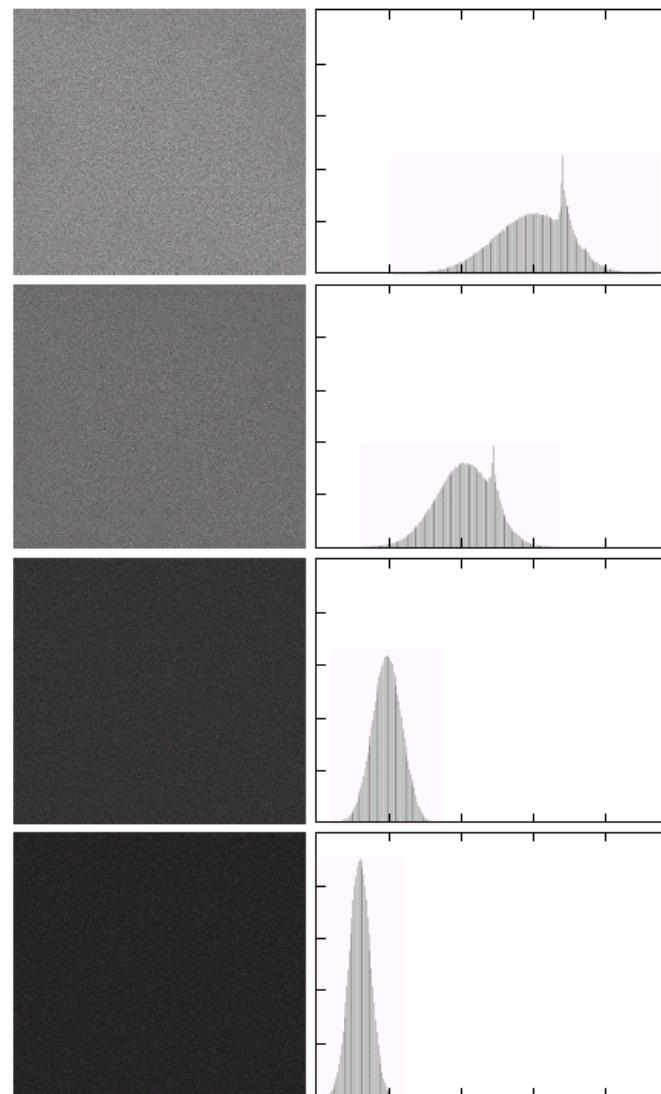


Image Averaging



Introduction to Spatial Filtering

- Use of spatial masks (kernels) for image processing (spatial filters)
- Linear and nonlinear filters
- Low-pass filters eliminate or attenuate high frequency components in the frequency domain (sharp image details), and result in image blurring
- High-pass filters attenuate or eliminate low-frequency components (resulting in sharpening edges and other sharp details)
- Band-pass filters remove selected frequency regions between low and high frequencies (for image restoration, not enhancement)

Introduction to Spatial Filtering

- In general, linear filtering of an image f of size $M \times N$ with a filter mask of size $m \times n$ is given by the expression

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

where $a=(m-1)/2$ and $b=(n-1)/2$; $m \times n$ (odd numbers)

for $x=0, 1, \dots, M-1$ and $y=0, 1, \dots, N-1$

- The process is also called *convolution* (used primarily in the frequency domain)

Spatial Filtering

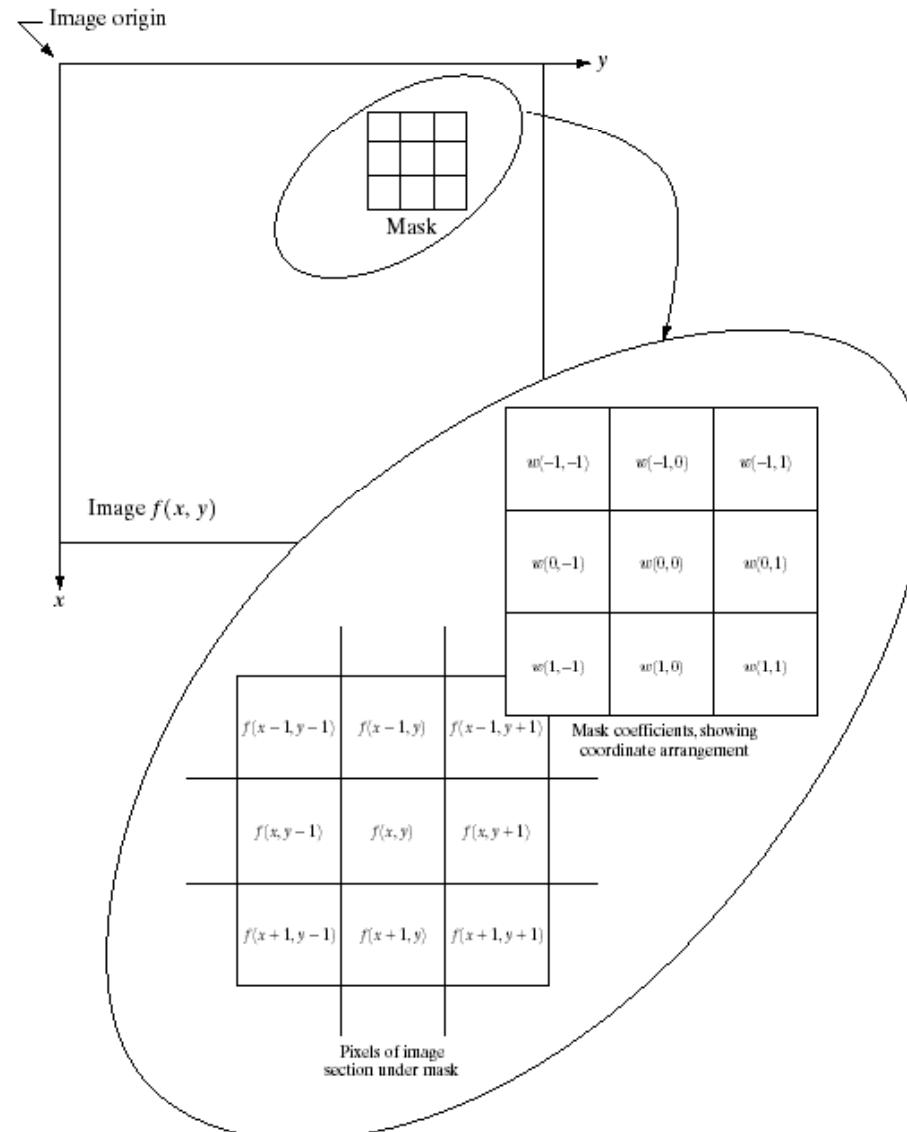
- The basic approach is to sum products between the mask coefficients and the intensities of the pixels under the mask at a specific location in the image:

$$R = w_1z_1 + w_2z_2 + \dots + w_9z_9$$

for a 3x3 image

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Spatial Filtering



Spatial Filter: Image Smoothing

- Aim of image smoothing is to diminish effects of camera noise, spurious pixel values, missing pixel values, etc.
- There exist many different techniques for image smoothing
- We will consider *neighbourhood averaging* and *edge-preserving* smoothing

Smoothing: Local Neighbourhood

- Each point in the smoothed image $F_s(x,y)$ is obtained from average pixel value in neighbourhood (x,y) in input image
- For example, for a 3x3 neighbour hood around each pixel, the following mask would be used:

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

- Each pixel value is multiplied by 1/9, summed, and the result placed in output image
- The mask is successively moved across the image until every pixel has been covered
- In other words, the image has been *convolved* with smoothing mask

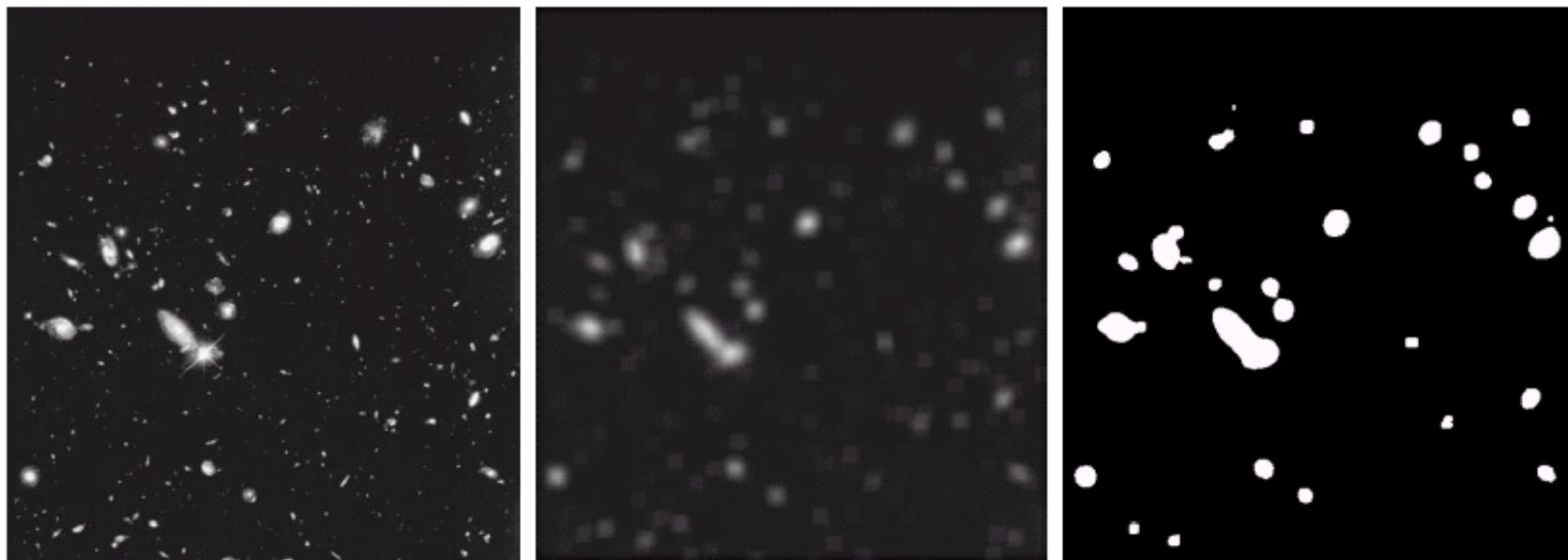
Smoothing: Local Neighbourhood

- However, one usually expects value of pixel to be more closely related to values of pixels close to it than those further away
 - Most image points are spatially coherent with their neighbours
 - Therefore usual to weight pixels near centre of mask more strongly

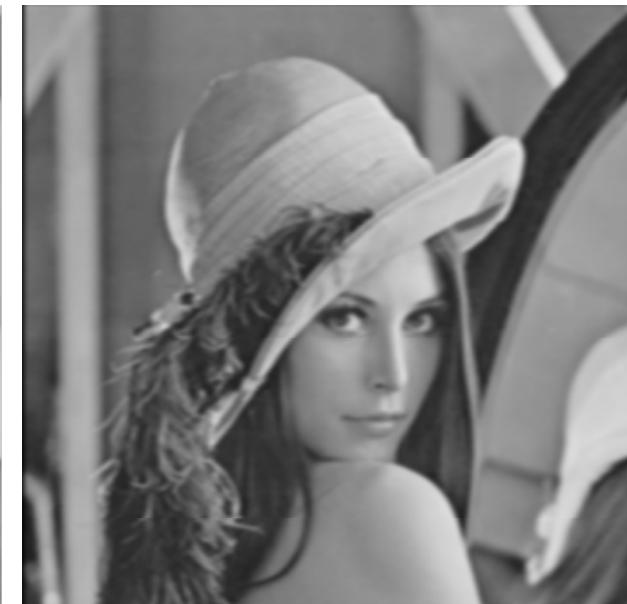
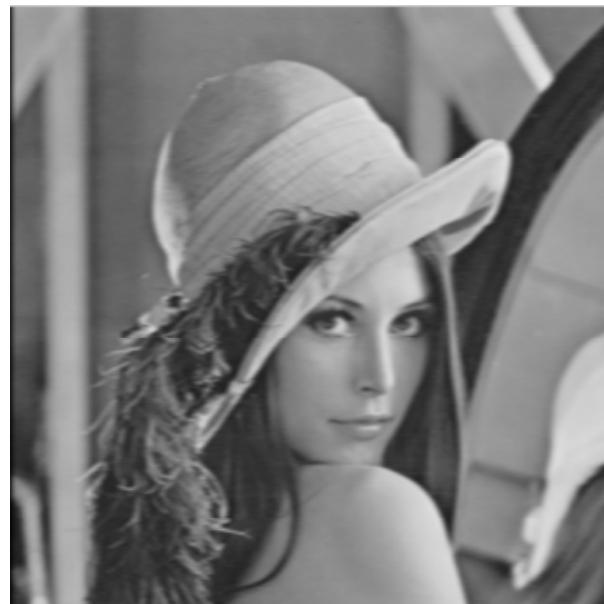
$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

- Common weighting functions include
 - Rectangular, triangular, Gaussian
- In practice, Gaussian smoothing is most commonly used
 - Frequency components are modified in smooth manner

Averaging Mask: Example



Averaging Mask: Example



Smoothing: Edge-Preserving

- Neighbourhood averaging or Gaussian smoothing tends to blur edges, due to attenuation of high frequencies in image
- An alternative approach is to use *median filtering*
- The greylevel of each pixel is replaced by the median of the greylevels in the neighborhood of that pixel (instead of by the average as before)
- Pixels with outlying values are forced to become their neighbours, and edges are preserved
- Non-linear filter
- Used primarily for noise reduction (eliminates isolated spikes)

Spatial Filter: Image Sharpening

- Main aim is to highlight fine image detail, or to enhance detail that has been blurred (e.g. due to motion, noise)
- Require to enhance high-frequency components
 - Implies filter shape with high positive component at centre
- Masks that achieve image sharpening include use of two 2x2 masks as follows:

1	0
0	-1

0	1
-1	0

- Computes sum of the squares of the differences between diagonally adjacent pixels
- Known as the Roberts Operator
- Each pixel in output image computed as
 - $\text{tmp1} = \text{absolute_value}(\text{input_image}(x,y) - \text{input_image}(x+1,y+1))$
 - $\text{tmp2} = \text{absolute_value}(\text{input_image}(x+1,y) - \text{input_image}(x,y+1))$
 - $\text{output_image}(x,y) = \text{tmp1} + \text{tmp2}$
- Note that all mask coefficients sum to 0, indicating a response of 0 in constant areas, as expected of a derivative operator

Spatial Filter: Image Sharpening

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

- Prewitt Operator

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

- Sobel Operator

Edge Filter: General Sobel

- The Sobel algorithm operates on the full array and evaluates

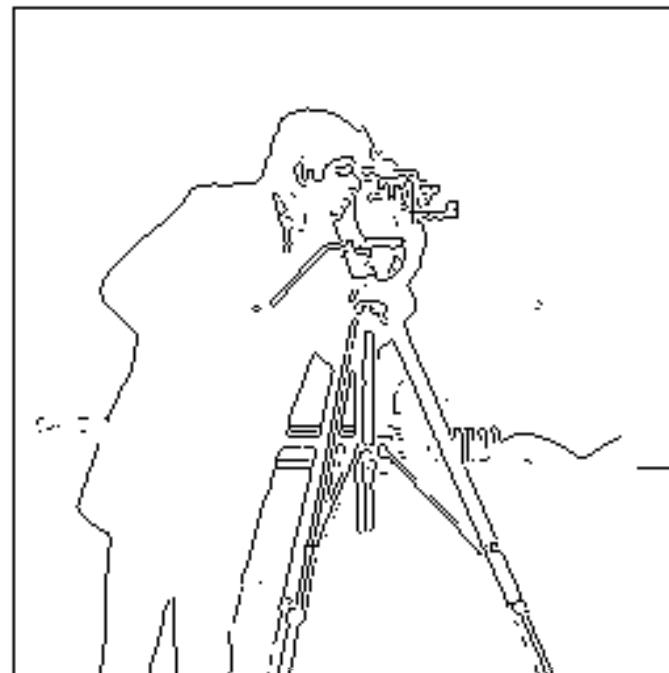
$$S(e) = \frac{1}{8} [| (a + 2b + c) - (g + 2h + i) |] + | (a + 2d + g) - (c + 2f + i) |]$$

a	b	c
d	e	f
g	h	i

Basic 3x3 mask

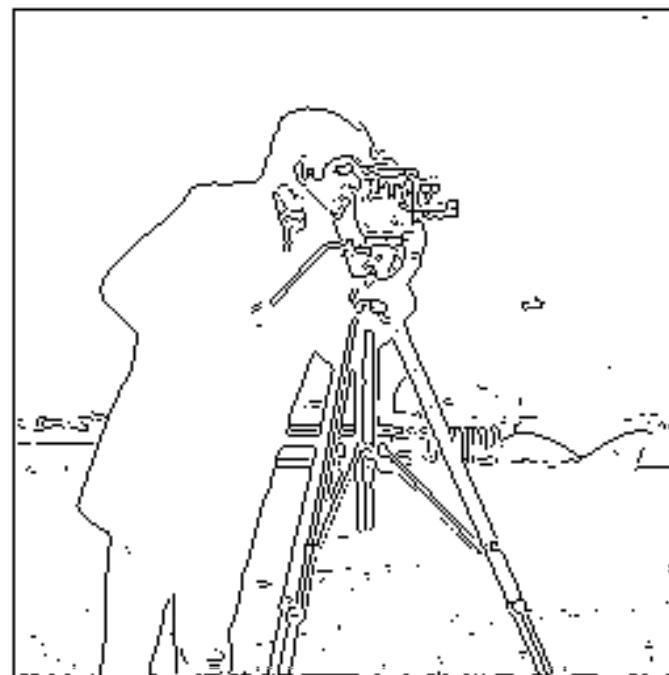
for each pixel. This output is a measure of the edge components passing through the kernel and is independent of both the polarity of the edge and, to a large extent, its orientation

Edge Filter: General Sobel



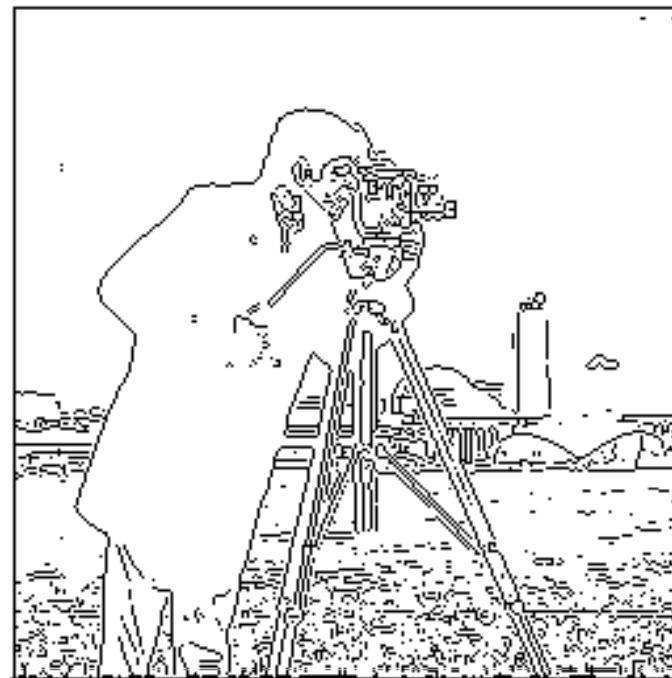
Threshold 1

Edge Filter: General Sobel



Threshold 2

Edge Filter: General Sobel



Threshold 3

Edge Filter: Laplacian



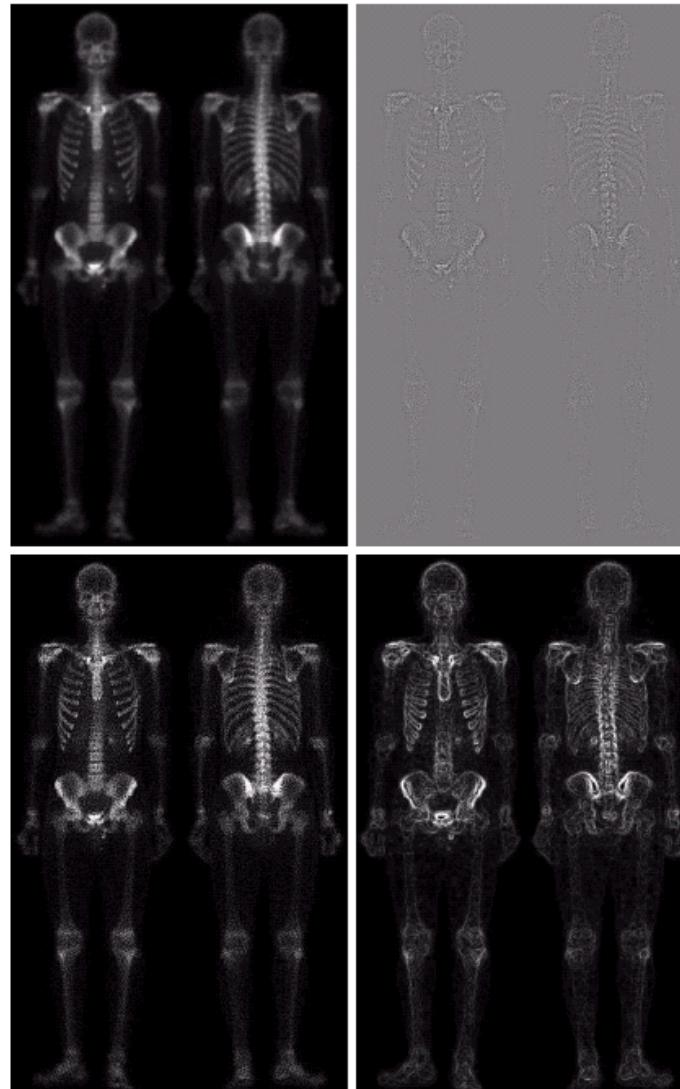
Edge Filter: Laplacian

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



Combining Enhancement Methods

- Attention has been paid to individual approaches
- Frequently, given enhancement requires application of several complementary techniques to achieve acceptable result

Summary

- Overview provided of common spatial enhancement techniques

SE3IA11/SEMIP12

Image Analysis



Image Enhancement in the Frequency Domain

Lecturer:

Prof. James Ferryman, Computational Vision Group

Email: j.m.ferryman@reading.ac.uk

Frequency Domain Enhancement

- The purpose of image enhancement is to emphasise some selected (desirable) image features and suppress others (undesirable features)
- In the last lecture, we examined enhancement in the spatial domain
- Several enhancement operations can be conveniently carried out in the frequency domain (where each image is represented by its Fourier transform)
- In this lecture we examine this topic, commencing with an introduction to image transforms and Fourier methods

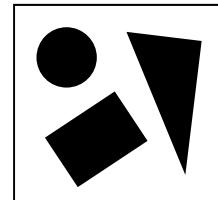
Image Transforms

- A digital image $f(x,y)$ is normally represented as a 2D array of intensity values, with x & y being the horizontal and vertical indices – spatial domain representation
- Many image analysis tasks are made easier in other alternative representations

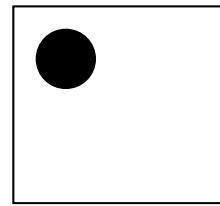
Image Decomposition

- How would you describe the content of the following hypothetical image?

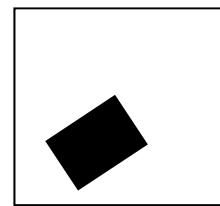
$f(x,y)$:



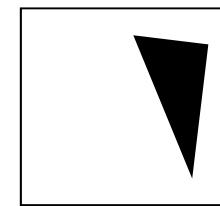
- If we have



$f_0(x,y)$:



$f_1(x,y)$:



$f_2(x,y)$:

then $f(x,y) = f_0(x,y) + f_1(x,y) + f_2(x,y)$, where $f_i(x,y)$, $i=0,1,2$, are used as reference or basis images (also called basis functions)

Hence the original image $f(x,y)$ is represented as a sum (linear combination) of three basis images

Image Transforms

- The above idea can be generalised to an arbitrary image $f(x,y)$ and a pre-defined set of N basis images/functions $\{f_i(x,y); i=0,1,\dots,N-1\}$

$$f(x,y) = \sum_{i=0}^{N-1} w_i f_i(x,y)$$

where w_i are the weighting coefficients which uniquely describe $f(x,y)$ for a given set of basis images

Introduction to Fourier Analysis

- Important properties of processes are often made clearer by transforming the data to *represent the information* in another form
- As with many areas of IT, image analysis makes extensive use of the Fourier Transform (FT)
- The FT uses spatial frequencies (i.e. sinusoidal intensity distributions in space) to describe the image and the operations on it
- The importance of Fourier Analysis is due to the fact that separate sine-wave components of a signal can be analysed individually, to determine the performance of a linear system under any signal

Introduction to Fourier Analysis

- Knowledge of the response of a (linear) imaging system to the set of all possible sinusoids allows the output to be determined for *any* image
- The sinusoids “don’t interact” under linear transformations, they form an *orthonormal basis set for additive signals*
- This is fairly familiar in acoustics – where we refer to sounds being high or low in frequency, and distinguish the smooth tones of a flute from the strident tones of a violin
- Both instruments may play the same *fundamental note*, but differ in their *harmonic components*
- Likewise, we expect a good quality amplifier to transmit pure tones without adding unwanted *distortion* components, thus allowing both the violin and the flute to remain identifiable
- The same sine-wave analysis can be applied to 2D images

Convolution

- Assume $F(u,v)$, $P(u,v)$ and $Q(u,v)$ are the FTs of images $f(x,y)$, $p(x,y)$ and $q(x,y)$
- If $F(u,v) = P(u,v)Q(u,v)$, what is $f(x,y)$ in terms of $p(x,y)$ and $q(x,y)$?
- Someone has proved for you that it's given by:

$$f(x, y) = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} p(k, l)q(x - k, y - l)$$

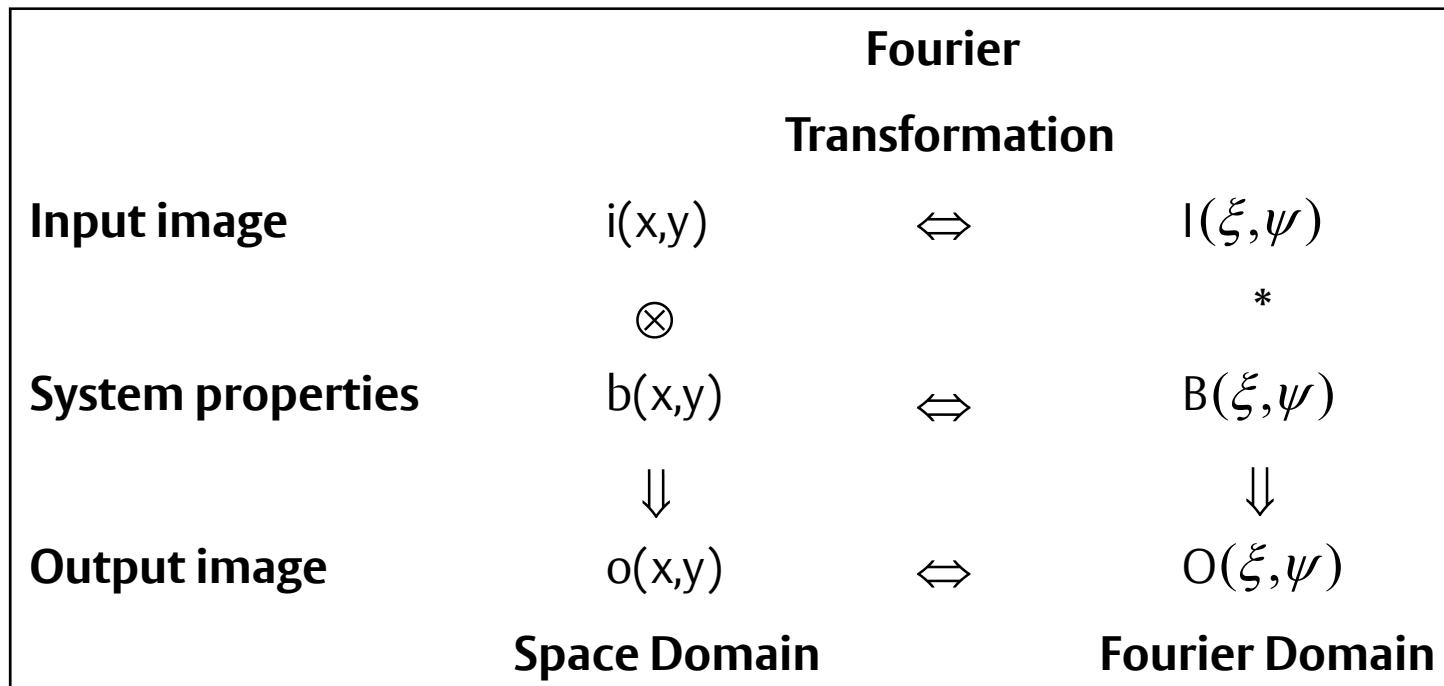
- Calculation of the above forms appears frequently in signal and image analysis, so it's written more compactly as

$$f(x, y) = p(x, y) * q(x, y)$$

- The above calculation is called *convolution*. Convolution is commutative, i.e., $p * q = q * p$. When computing $p * q$ or $q * p$, p & q are assumed to be periodic functions to cope with negative indices, i.e., $p(k, l) = p(N+k, N+l)$, etc.

Illustrating the Convolution Theorem

- Consider an input image $i(x,y)$, which is blurred by a weighting function $b(x,y)$, to form an output $o(x,y)$.
- We have $o = i \otimes b$ (space domain) or $O = I * B$ (frequency domain)
- (x,y) are space domain parameters, (ξ, ψ) are frequencies.



Illustrating the Convolution Theorem

- Notes:
 - The Blur function, $b(x,y)$, is often called the Point Spread Function (PSF) – it measures how an infinitesimal point (also called a delta function) is *spread out* by the system
 - Its spectrum, $B(\xi,\psi)$, is often called the Modulation Transfer Function (MTF) – it measures how the modulation of each component is *transferred* (i.e. amplified or attenuated) by the system
 - Either the PSF or MTF provide a *complete description* of a linear shift-invariant system

Convolution

- In summary, we have
multiplication (convolution) in freq domain

↔

convolution (multiplication) in spatial domain

This is called the *convolution theorem*. It often allows complicated processing (e.g. filtering) in the spatial domain to be performed very efficiently in the frequency domain

Fourier Properties

- A summary of some important properties of Fourier Analysis:
 - Any image can be represented as the sum of a set of sine and cosine distributions – this is more neatly expressed as a complex exponential, which conveniently combines the cos and sin terms

Thus:
$$F(\xi, \psi) = \frac{1}{\sqrt{2\pi}} \sum f(x, y) e^{-j(\xi x + \psi y)}$$

Any function is identically equal to the sum of its harmonics

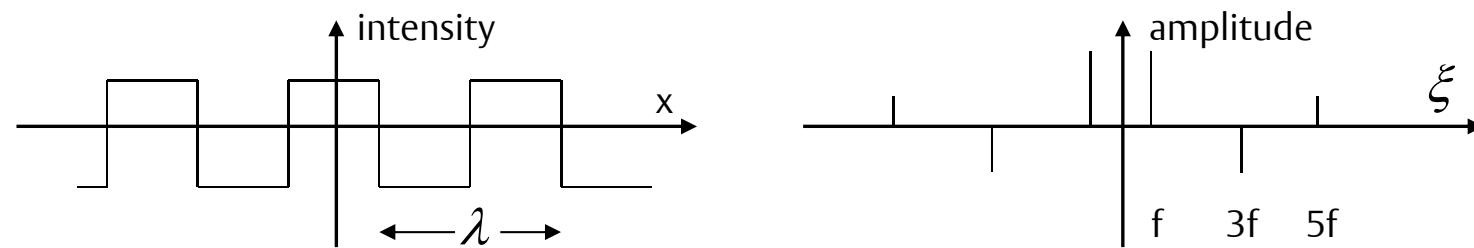
- Reciprocal analysis & synthesis: if $F(\xi, \psi)$ is the transform of $f(x, y)$, then $f(x, y)$ is also the transform of $F(\xi, \psi)$; they are said to be Fourier mates (or pairs).

Fourier Properties

- A summary of some important properties of Fourier Analysis:
 - The Convolution Theorem: $h = f \otimes g$ (each a function of (x,y)) can be expressed very easily in the Fourier Domain.
The Fourier Transform of h is the *product* of the Fourier Transforms of the separate functions: $H = F^*G$ (each a function of (ξ, ψ)).
Hence: under any convolution, the harmonics only ever change their amplitudes, and do not change frequencies (i.e. there is no distortion).
 - Reciprocal size relationship between the two domains: large objects in space, are small in frequency space & vice-versa.
 - This is due to the fact that frequencies represent cycles *per* space metric
 - **(For Mathematicians only!)** Complex exponentials are the *eigenvectors* of the class of linear transformations, they always transform into scalar products of themselves. The *eigenvalues* (the scalars, which form the system *spectrum*) completely characterise the transformation

Spectral Analysis

- Return to considering convolution operators
- Both the *input image* (the signal) and the convolution *weighting function* (the operator system response) can be represented as frequency distributions, i.e. their Fourier Transforms
- A very simple example: the square wave (in 1D):



- Any periodic distribution, of period λ , transforms into a discrete series of Fourier components: the *fundamental* ($f = 1/\lambda$), and all integral multiples, $2f, 3f, 4f, \dots$
- In the case of a *square wave*, the values of all even harmonics is 0 and only the odd harmonics are present – ($\xi = f, 3f, 5f, 7f, \dots$). Their amplitudes decay as the inverse of frequency: $4a/\pi, 4a/3\pi, 4a/5\pi, \dots$ where a is the amplitude of the square wave, i.e. $(\max - \min)/2$

Spectral Analysis

- “Blurring” a square wave, by means of a linear, shift-invariant operator, merely changes the amplitudes of the components
 - It does not introduce any new components

[Amplitudes are multiplied by the amplitude of the weighting function at the frequency]

- We can also go the other way: an image can be synthesised simply by adding up the values of a set of components at each point. This is expressed mathematically as the inverse Fourier Transform:

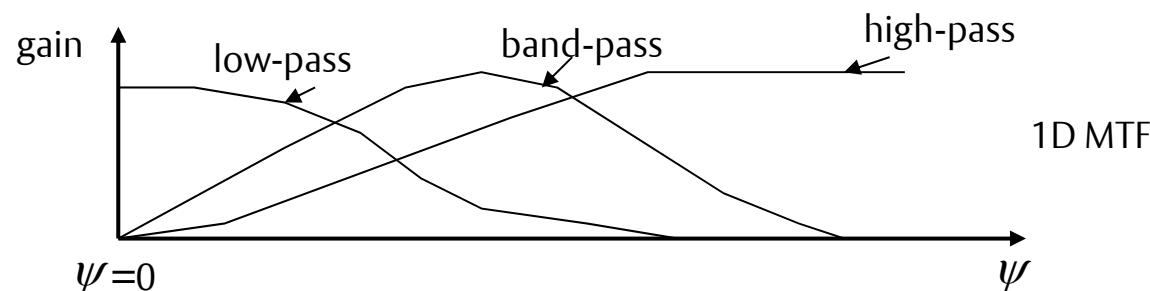
$$f(x, y) = \frac{1}{\sqrt{2\pi}} \sum F(\xi, \psi) e^{j(\xi x + \psi y)}$$

for all (ξ, ψ)

- Note the near symmetry of the forward and inverse transforms – only the Fourier pairs and the sign of the complex exponential changes

Fourier Jargon

- Fourier methods provide an *alternative description* of the information content of a signal, and jargon from radio theory has been adopted
- Filtering processes are often said to be low-, high-, or band- pass according to the gain at different frequencies
- Band & High-pass filters are said to be AC-coupled, and have zero response (i.e. the value at $\psi=0$ is 0)
- Low-pass filters transmit the steady signal (the mean over time or space), and are said to be DC-coupled



- The finest detail in an image is carried by its highest frequencies. Low-pass filtering therefore corresponds to smoothing, which suppresses the fine detail, but has little effect on the low frequencies

Discrete Fourier Transform (DFT)

- DFT is the sampled Fourier Transform
 - When dealing with digital images, we are never given a continuous function
 - Must work with finite number of discrete samples
 - Samples are pixels that compose an image
 - Analysis of images requires the DFT
 - DFT does not contain all frequencies forming an image; instead contains a set of samples sufficient enough to fully describe the spatial domain image
 - Number of frequencies corresponds to number of image pixels
 - i.e. image in spatial and Fourier domain are of same size

Discrete Fourier Transform (DFT)

- The general forms of image transforms

Forward:
$$F(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y)g(x, y, u, v)$$

Inverse:
$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v)h(x, y, u, v)$$

In DFT both transformation kernels (g & h) are complex functions, and the results are in general also complex.

If the kernels are real functions of the following form

$$g(x, y, u, v) = h(x, y, u, v) = \frac{2c(u)c(v)}{N} \cos\left(\frac{2x+1}{2N}u\pi\right) \cos\left(\frac{2y+1}{2N}v\pi\right)$$

where $c(0) = 1/\sqrt{2}$ and $c(w) = 1$ for $w = 1, 2, \dots, N-1$, then the transform is called the discrete cosine transform or DCT (because kernels are cosines).

Note that the cosines are also separable in x & u and y & v , so a 2D DCT can be obtained from two 1D DCTs as in DFT

The DCT kernels may be written in terms of DFT kernels. This allows fast DCT via FFT.

Discrete Fourier Transform (DFT)

- For a square image of size $N \times N$, let the basis set be defined by the following N^2 complex sinusoidal functions:

$$h(x, y, u, v) = \frac{1}{N} \exp \left[\frac{j2\pi}{N} (ux + vy) \right]; u, v = 0, \dots, N-1$$

then we can represent an image $f(x, y)$ as a weighted sum of these sinusoidal functions:

$$\begin{aligned} f(x, y) = & F(0,0)h(x, y, 0, 0) + F(0,1)h(x, y, 0, 1) \\ & + F(0,2)h(x, y, 0, 2) + \dots + \\ & F(N-1, N-1)h(x, y, N-1, N-1) \end{aligned}$$

or more compactly

Discrete Fourier Transform (DFT)

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) h(x, y, u, v)$$

or

$$f(x, y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) \exp\left[\frac{j2\pi}{N}(ux + vy)\right] \quad (1)$$

Interestingly, the weighting coefficients $F(u, v)$ are calculated from the original image in a similar equation:

$$F(u, v) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} f(x, y) \exp\left[\frac{-j2\pi}{N}(ux + vy)\right] \quad (2)$$

Discrete Fourier Transform (DFT)

Let

$$g(x, y, u, v) = \frac{1}{N} \exp\left[-\frac{j2\pi}{N}(ux + vy)\right]$$

then (2) may be equivalently written as

$$F(u, v) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} f(x, y) g(x, y, u, v)$$

Equations (1) and (2) are said to define a Fourier transform (DFT) pair, where (2) is the forward transformation (i.e. to map image to Fourier domain), and $g(x, y, u, v)$ the forward transformation kernel, and (1) the inverse transformation (i.e. to map from Fourier to spatial domain) and $h(x, y, u, v)$ the inverse transformation kernel. Since u & v are related to the frequencies of the sinusoidal functions, they are called frequency indices

- Note $F(u, v)$ calculated in (2) is in general a complex value, i.e., it has a real part $F_R(u, v)$ and an imaginary part $F_I(u, v)$:

$$F(u, v) = F_R(u, v) + jF_I(u, v)$$

Discrete Fourier Transform (DCT)

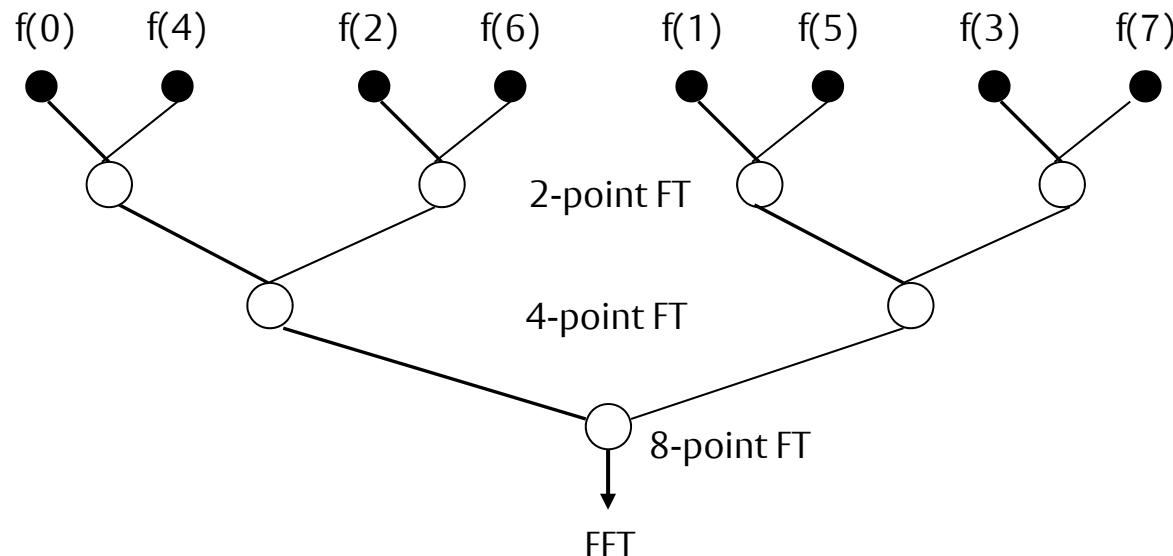
- FT produces a complex number valued output image
 - can be displayed as two images
 - Either *real* and *imaginary* part or *magnitude* and *phase*
- In image analysis, often only magnitude of FT is displayed
 - contains most of information of geometric structure of spatial domain image
- However, if one needs to re-transform the Fourier image into the correct spatial domain after some processing in the frequency domain, must ensure *both magnitude and phase of Fourier image are preserved*

Computational Methods

- The Fast Fourier Transform
- Direct calculation of the 1D FT of a N-point function $f(x)$ requires multiplications in the order of N^2 , a slow process for large N s
- Efficient method discovered in 1942, but not applied until later due to lack of computing power

Fast Fourier Transform (FFT)

- It was found that a N -point FT can be obtained from two $N/2$ -point DFTs each of which can in turn be obtained from two $N/4$ -point DFTs and so on – the *successive doubling* method (see below)
- Coined the Fast Fourier Transform (FFT) (Cooley & Tukey, 1965)
- FFT reduces number of complex multiplications from N^2 to order of $N \log_2 N$



Fast Fourier Transform (FFT)

- The divide and conquer technique used allows for significant speedup over DFT for 2D images

Image Size	DFT Multiplications	DFT Time	FFT Multiplications	FFT Time
256 x 256	4.3 E9	71 min	1,048,576	1.0 sec
512 x 512	6.8 E10	19 hr	4,718,592	4.8 sec
1024 x 1024	1.1 E12	12 days	20,971,520	21.0 sec
2048 x 2048	1.8 E13	203 days	92,274,688	92.2 sec

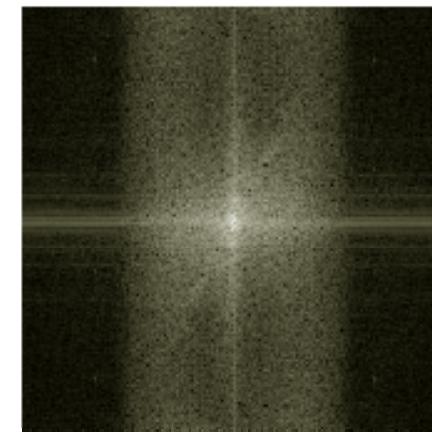
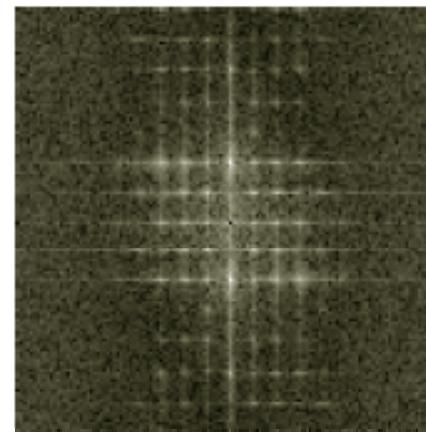
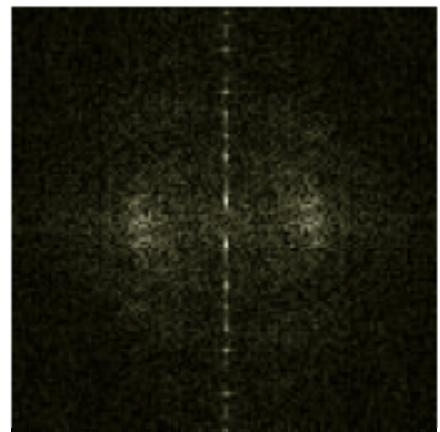
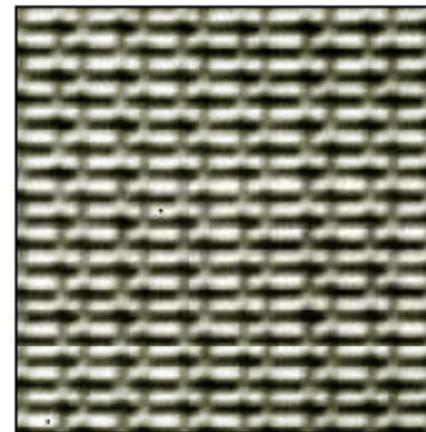
Savings when using FFT on 2D image data

Fast Fourier Transform (FFT)

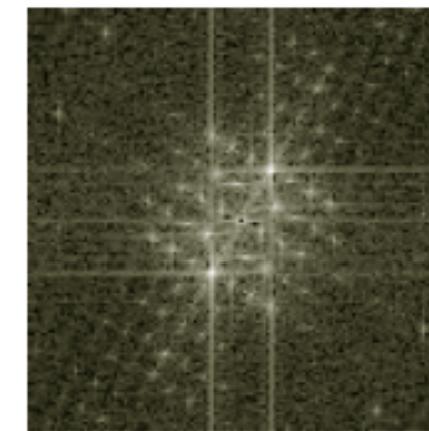
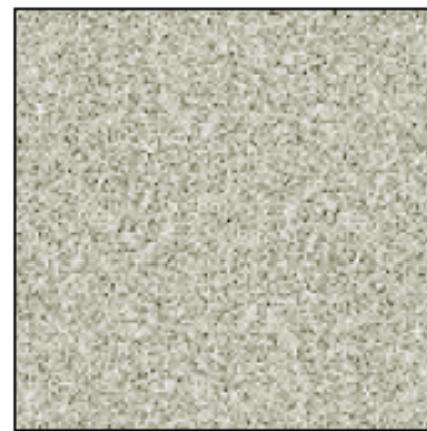
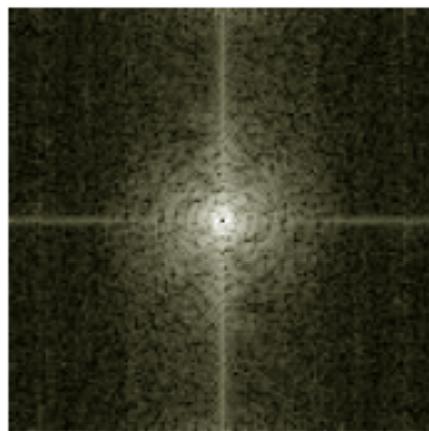
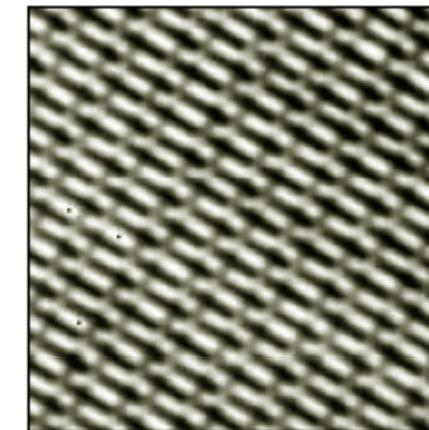
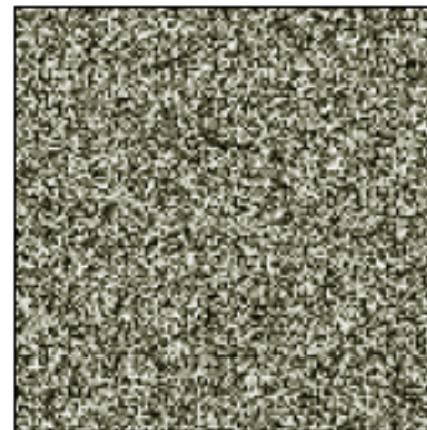
- For an efficient implementation of the FFT in C, consult the book *Numerical Recipes in C*, available on the Web at www.nr.com
 - <http://www.nrbook.com/a/bookcpdf.php>
 - *Chapter 13: Fourier and Spectral Applications*

Examples of Fourier Spectra

This is called subbed unsubscribing available or to find commands volunteers and staff to them manage for example, volume



Examples of Fourier Spectra



Significance of Spectral Peaks

- Peaks in the frequency domain indicate global periodic structures in the spatial domains
- Direction (angle with u-axis) of spectral peaks is direction (angle with x-axis) of corresponding global structure +/- 90⁰

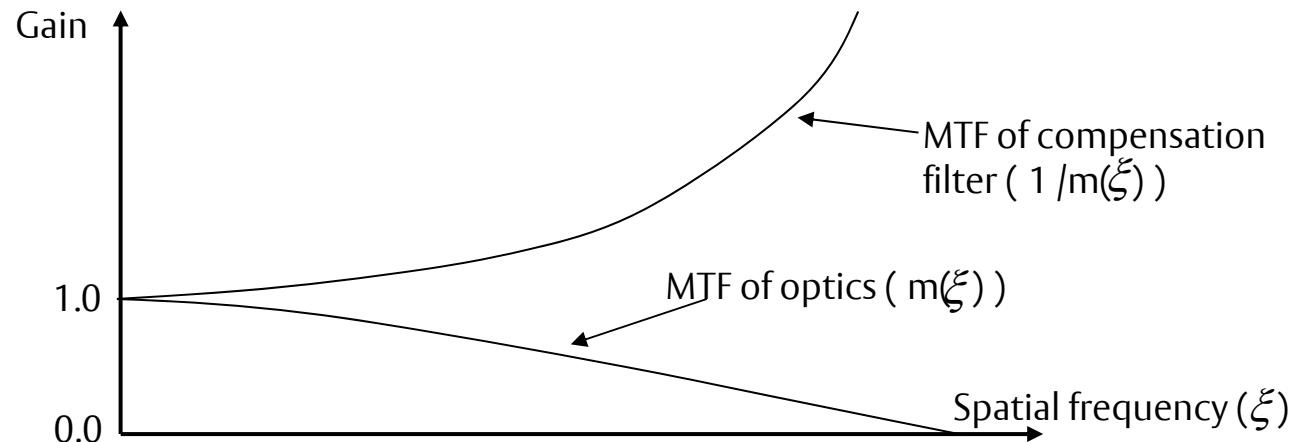
Fourier Transform – Guidelines for Use

- FT is used to access the geometric characteristics of spatial domain image
- Since image in FD is decomposed into its sinusoidal components, easy to examine or process certain image frequencies, influencing geometric structure in spatial domain
- In most implementations, Fourier image is shifted such that DC-value (i.e. image mean) $F(0,0)$ is displayed in centre of image
 - i.e. further away from centre an image point is, the higher its corresponding frequency
- Phase information is crucial to reconstruct image in spatial domain

Application: Image Reconstruction

- One of the simplest applications of Fourier Methods in image analysis is to correct the image for blur due to optical defects
- A typical lens, or any form of electronic sensor, imposes a degree of smear on an image, due to poor optics or the finite size of the sensor element
- The MTF of a system, $m(\xi)$ can be measured experimentally – simply present the system with a sine-wave input, and measure the amplitude (and phase) of the output
- The MTF of an imperfect system commonly takes the form of a low-pass filter which attenuates the high frequencies progressively, until the cut-off frequency, where $m(\xi) = 0$, and the information is lost

Application: Image Reconstruction



- But it is perfectly feasible to define a compensation filter whose gain at each frequency is exactly $1/m(\xi)$
- Consider what happens if we convolve the (blurred) measured image with the new filter
- Convolution is multiplication in the Fourier domain, so each component of the input is first attenuated by the blur, but then amplified back to its original level
- This happens at each frequency, so the result (in theory) is a perfectly reconstructed image, with all the blur removed

Application: Repetitive Signal Removal

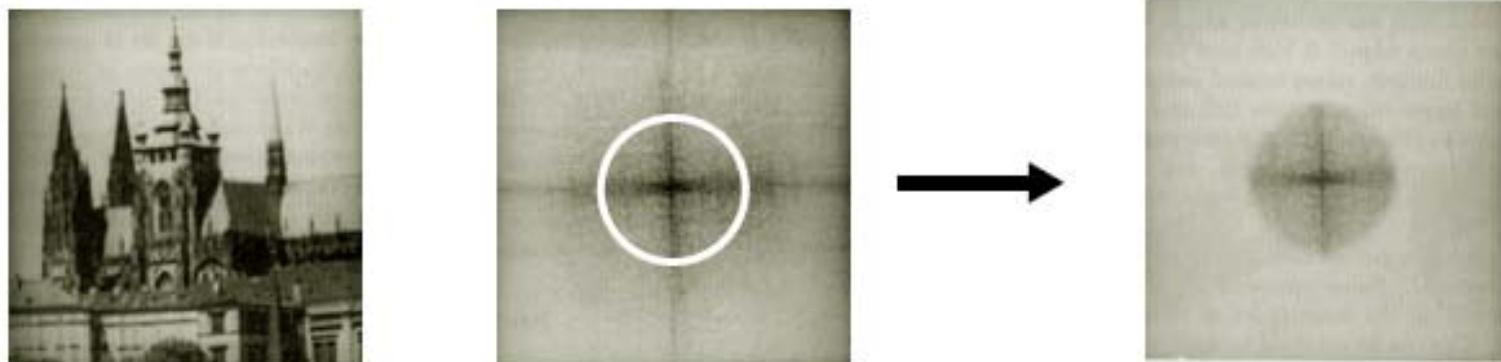
- Another example of Fourier methods in image analysis is to remove unwanted repetitive signals – e.g. eliminating video lines by blocking the (vertical) frequencies which they introduce
- Note the similarity to “notch filters”, which remove hum, or the modulation frequency in audio samples
- These methods have often been implemented optically, since the processing can be done at the speed of light!
- Their digital equivalents are also widely used, since although slower, they are far more flexible

Application: Filtering

- Filtering is one operation whereby some image components are removed (i.e. filtered out) so as to enhance others

Lowpass Filtering

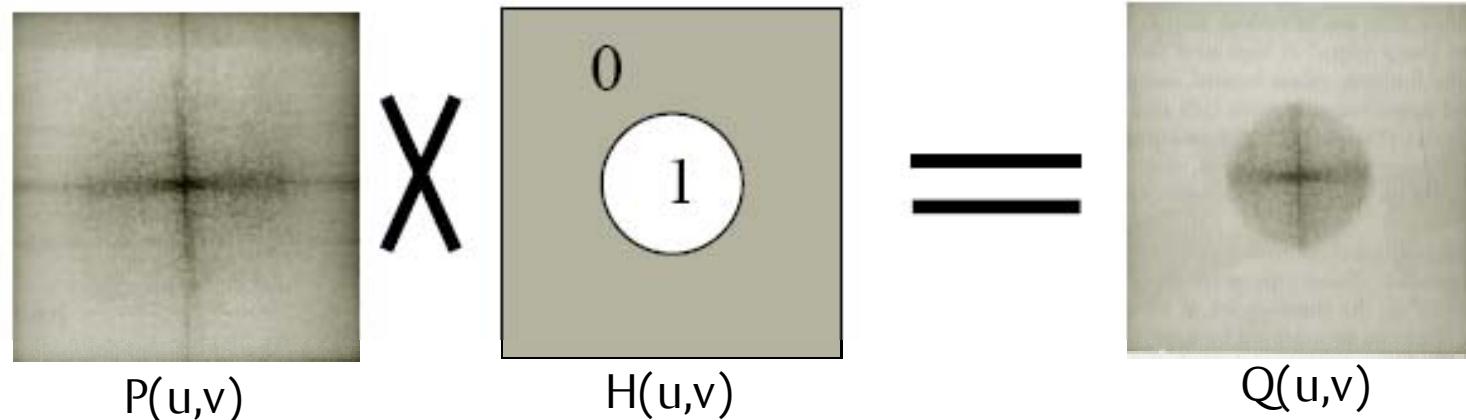
Let's say we wish to remove HF components in order to enhance the LF ones. We can do this by setting the value of the Fourier spectrum at HF locations to zero and leave the LF values unchanged. The inverse FT of the modified frequency domain (FD) representation gives the desired image



Lowpass Filtering

- The above zero-setting operation is equivalent to multiplying the original FD representation by a function $H(u,v)$ defined as

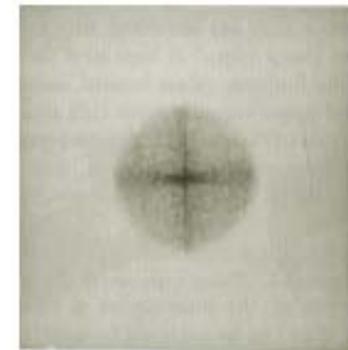
$$H(u,v) = \begin{cases} 1; & \text{for LF locations (e.g. within a circle)} \\ 0; & \text{for HF locations (e.g. outside the circle)} \end{cases}$$



$$Q(u,v) = P(u,v)H(u,v)$$

Lowpass Filtering

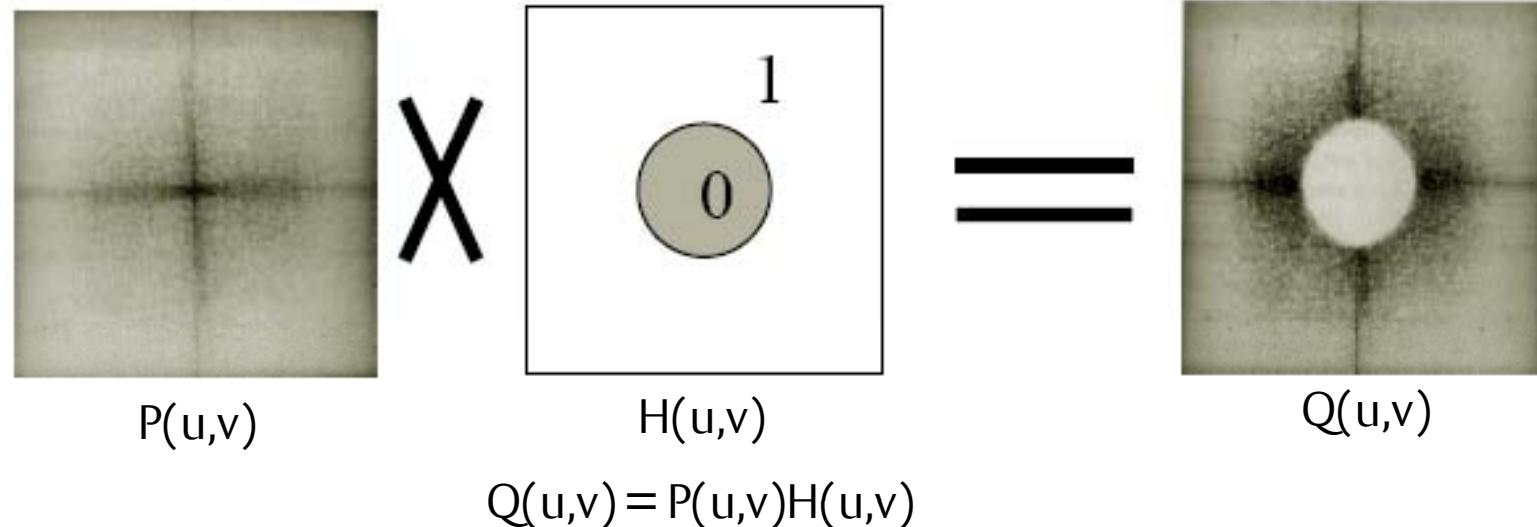
- Since $H(u,v)$ serves as a filter which only lets LF components pass, it's called a low-pass filter (LPF). Its shape doesn't have to be circular. Further examples of lowpass filtering are given below:



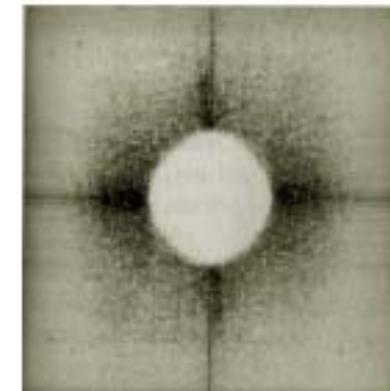
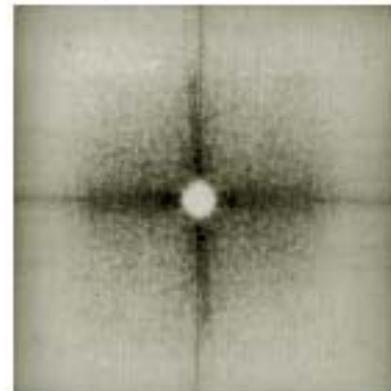
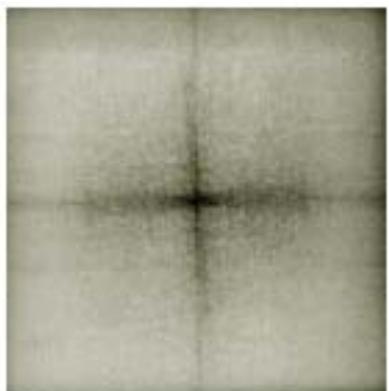
Highpass Filtering

- This is performed similarly. In this case, the filter function $H(u,v)$ keeps HF values unchanged and erase all LF ones,

e.g.:
$$H(u,v) = \begin{cases} 0; & \text{If } (u,v) \text{ is within a circle} \\ 1; & \text{otherwise} \end{cases}$$



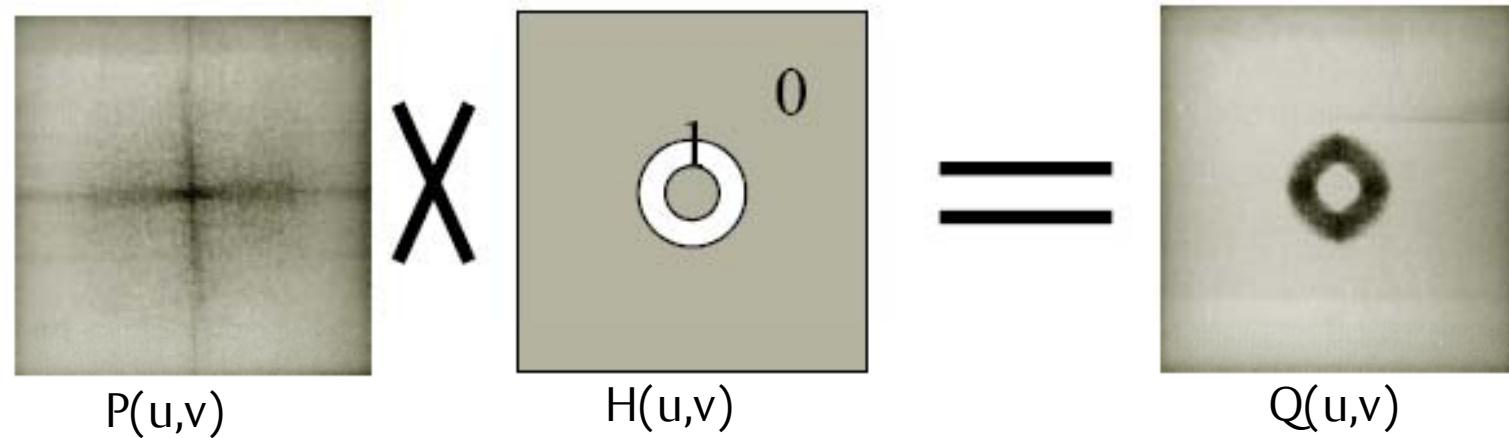
Examples of Highpass Filtering



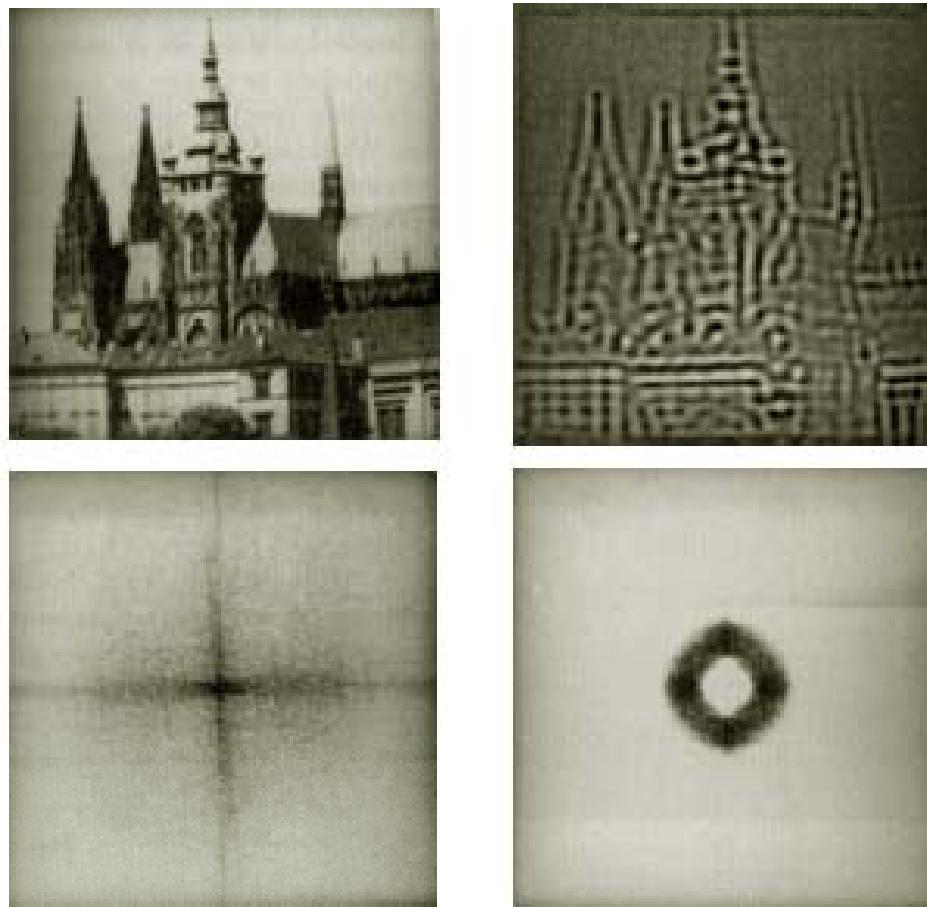
Bandpass Filtering

- Only keeps values corresponding to frequencies in a given interval, so a bandpass filter (BPF) may be defined as

$$H(u, v) = \begin{cases} 1; & \text{If } (u, v) \text{ is in overlaps of two circles} \\ 0; & \text{otherwise} \end{cases}$$



Example of Bandpass Filtering



Application: Noise Removal

- Intensity values of pixels do not always reflect the true information about the underlying scene (i.e. the object photographed) since these values may also have been altered undesirably (e.g. due to dirt on the scanner, camera, etc.) If so, the image is said to be *noisy* or have been contaminated by noise
- If the measured intensity value is a sum of the true value and a small term due to noise, the image contains *additive noise*
- If the intensity of some of the pixels is predominantly due to noise and is significantly different from those of neighbouring pixels, the image contains *impulse noise* (or salt-and-pepper noise)
- If the noise term forms some sort of structure, the image contains *structured noise* (or coherent noise)

Examples of Noisy Images



Additive

Impulse

Structured

Lowpass Filtering

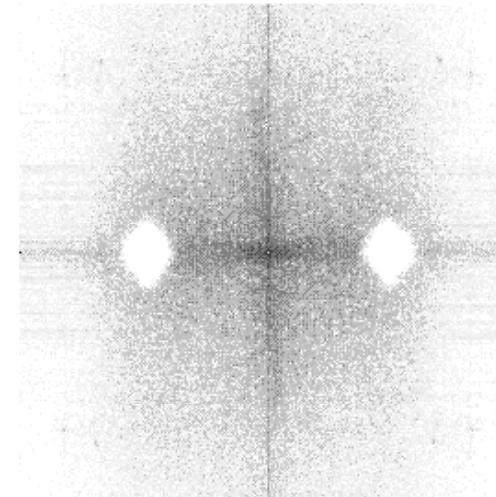
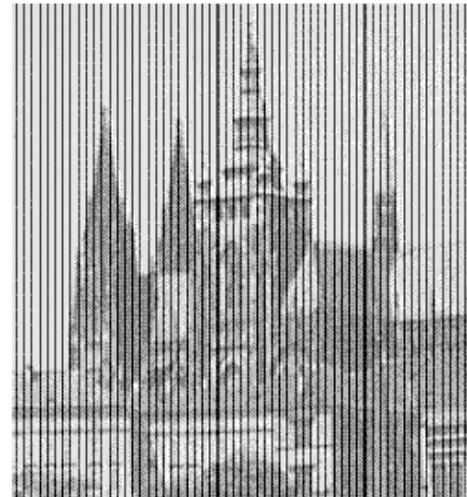
- Natural images usually contain relatively low frequency components and significant HF components are likely due to noise
- Hence a large part of noise may be removed by means of lowpass filtering
- This can be done by using a LPF in the FD, or alternatively by local averaging (see last lecture.)

Lowpass Filtering: Noise Removal



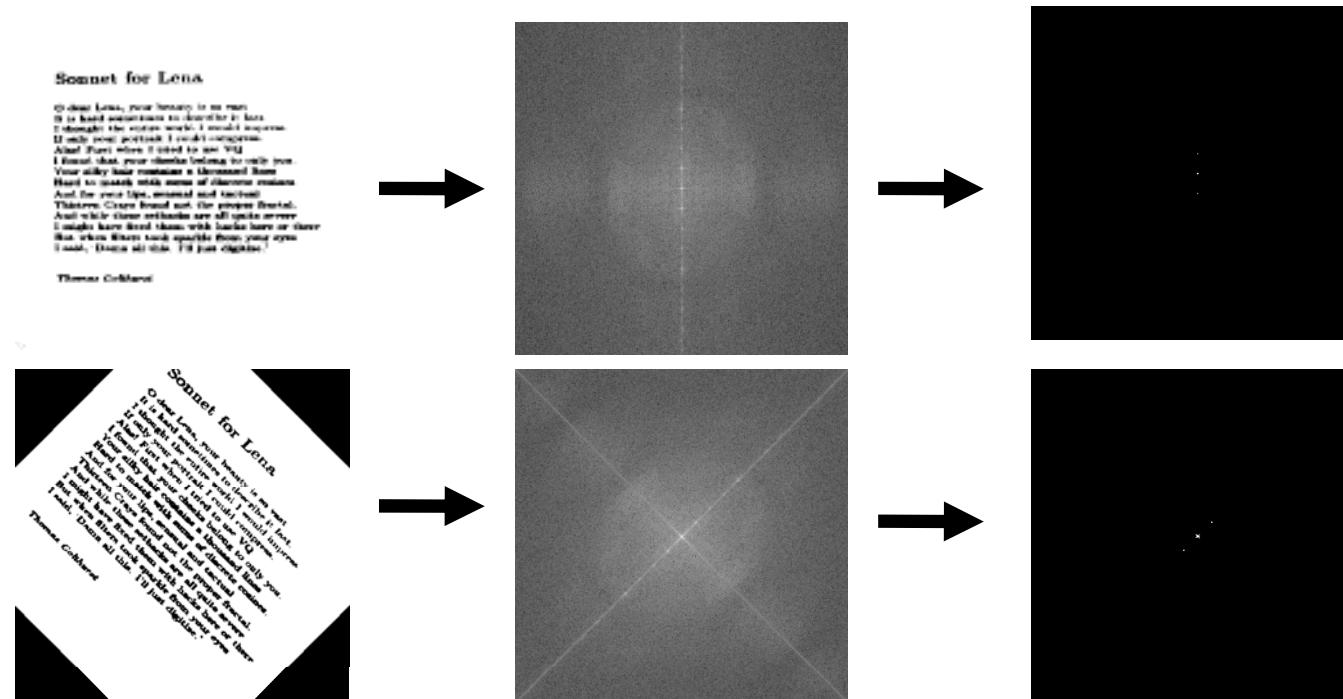
- A problem with noise removal via LPF is the loss of sharpness as edges are blurred
- Can be solved via e.g. Median Filtering (see last lecture)

Example of Periodic Noise Removal



Application: Text Orientation

- Text recognition using image analysis techniques is simplified if we assume that text lines are in a predefined direction
- FT can be used to locate initial orientation of text
- A rotation can then be applied to correct error



Other Transforms

- The FT is not the only transform used in image analysis
- Other transforms include:
 - Hilbert, Hartley, Hough, Hotelling, Hadamard, Haar, Walsh, Wavelet, Karhunen-Loève
 - As well as the DCT, DST
- However, the FT, due to its wide application base, is one of the most popular
- The next slide summarises the Wavelet Transform
- We then examine some applications of the FT to image enhancement

Wavelet Transform

- The DFT *is good* for analysing global image features such as periodicity of certain image structures
- However, DFT is *not useful* in describing more local features (e.g. frequency components in small image region)
- Can be overcome by applying DFT to small image regions at various locations (called *windowed Fourier Transform*)
- The solution appears to be obvious: use a set of sizes (scales) and a set of locations (to enable you to see both the forest and trees, so to speak.)
- The above is the basic idea behind the Wavelet Transform, where the basis functions are the scaled (dilated) and translated versions of a “mother” function or wavelet, e.g.:
- Mother wavelet: $g(x)$
Other wavelets: $g_{a,b}(x) = \frac{1}{\sqrt{a}} g\left(\frac{x-b}{a}\right)$

Most research has focussed on the choice of the Mother wavelet

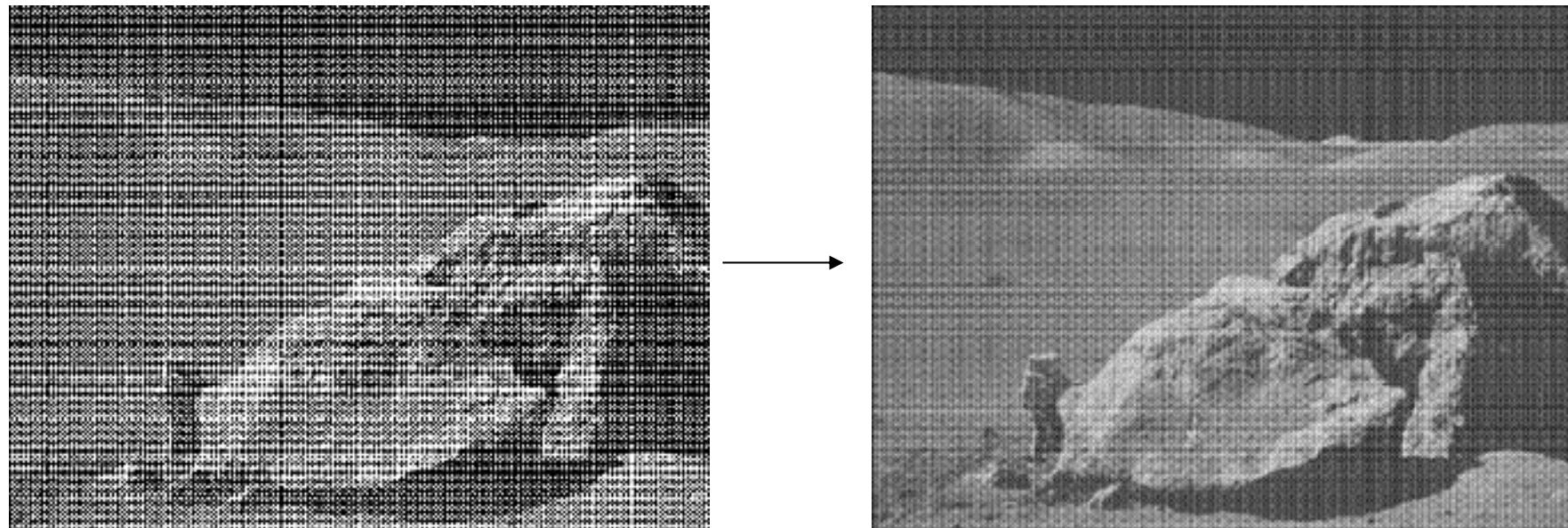
Case Study: Customised Digital Filters

- The following provides a case of study of customised digital filtering applied to noisy image data

Why Image Enhancement?

- Noise/ Distortion reduction: preprocessing step before image segmentation
- Easiest way: blind smoothing using a spatial LP filter
- BUT: still regular unwanted patterns

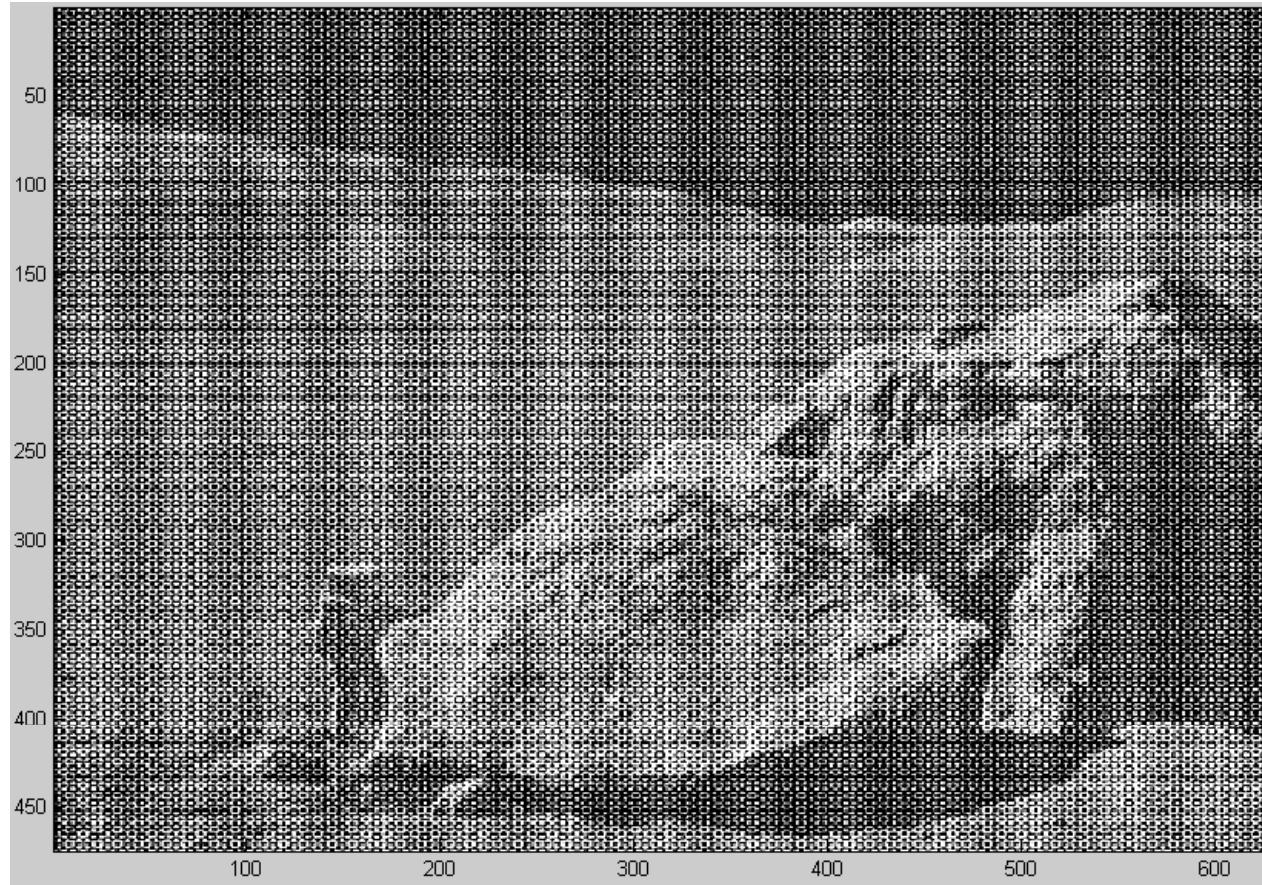
$$mask = \frac{1}{9} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



The Algorithm

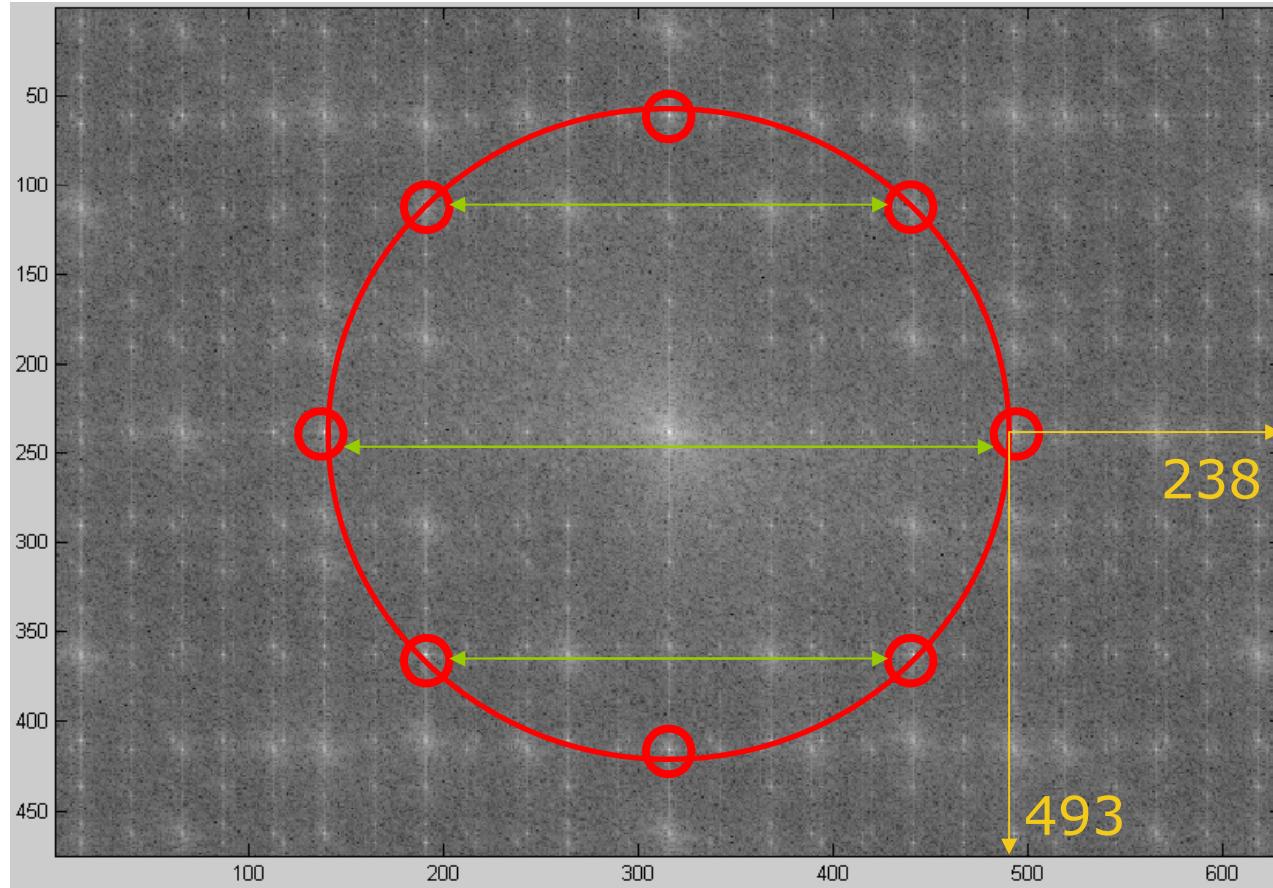
1. Perform a 2D-FFT and centre the spectrum of the corrupted image
2. Estimate disturbing frequency by finding the maximum energy
3. Design a band reject filter according to the estimated frequency
4. Filter corrupted image with band reject filter
5. Perform a 2D-IFFT to obtain the enhanced image

The Original Distorted Image $f(x,y)$ from the Moon's Surface



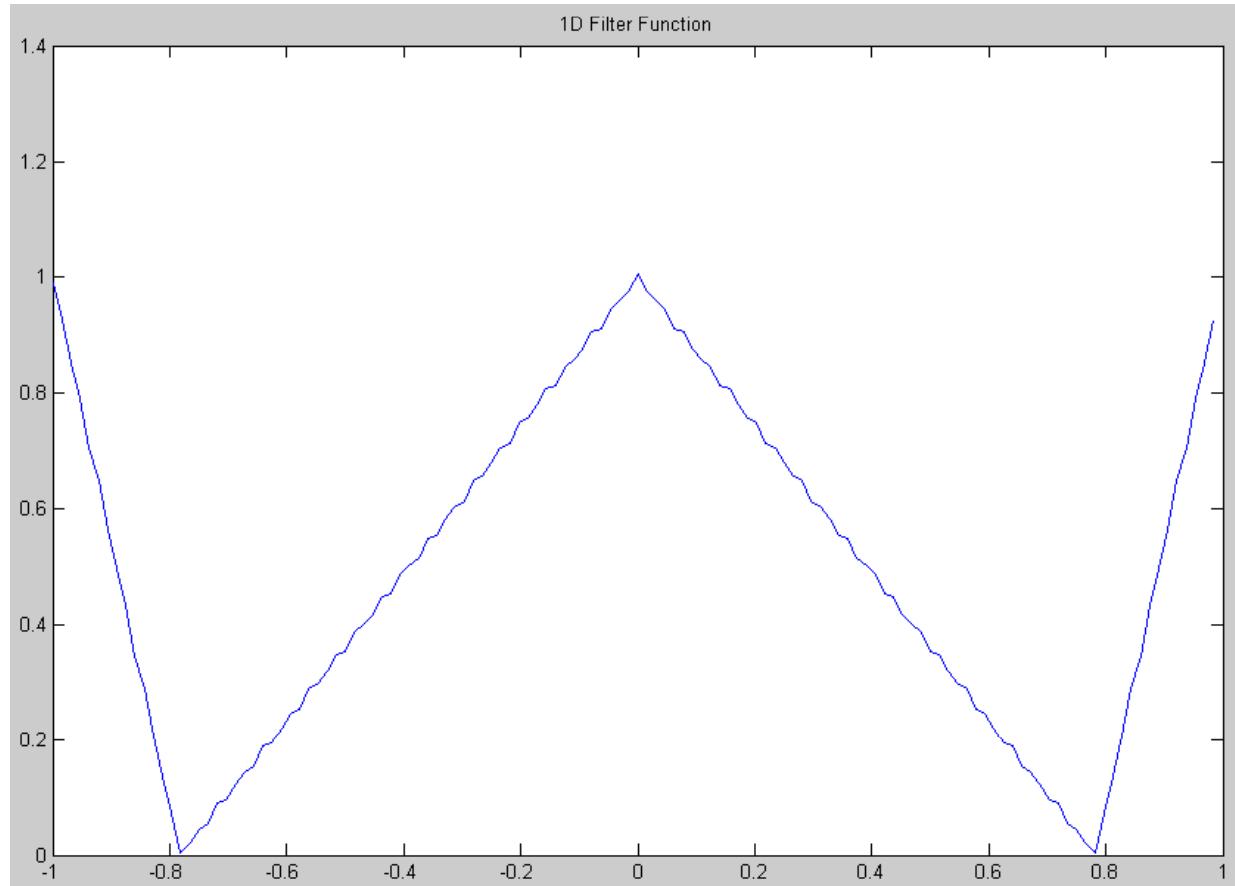
- Periodic Noise
- Frequent signal interference during image acquisition and / or transmission
- Human eye is tolerant enough, whereas edge detection algorithms will fail

The Original Image's Spectrum $F(x,y)$



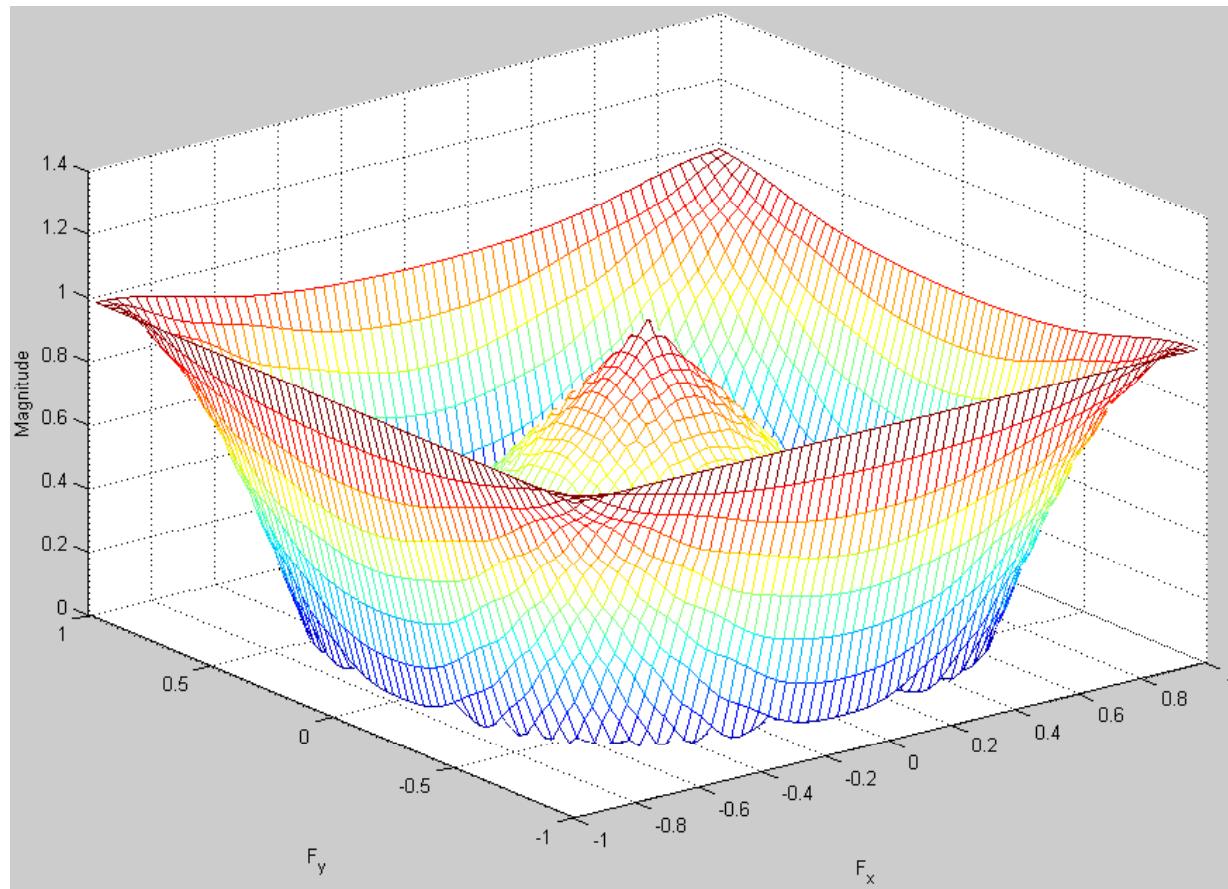
- Most energy at most frequent grey values
- Pairwise pulses mirrored at the Nyquist frequency
- Normed frequency to be attenuated:
 - $f_0 = 493 / 630 \approx 0.7825$

The Magnitude Response of the Filter (2D View)



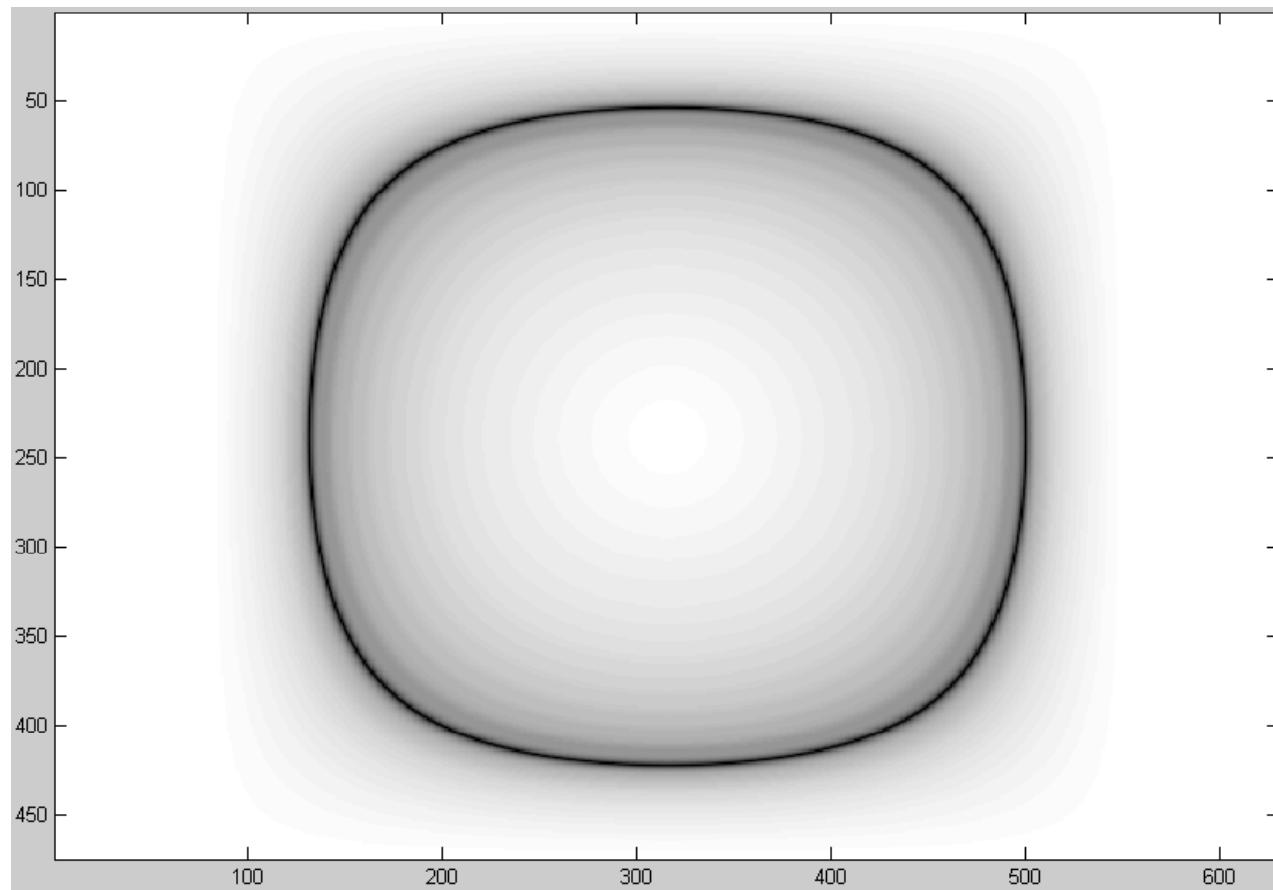
- Band reject filter
(here: notch filter)
- Attenuates image at the
normalised frequency
 $f_0 = 0.7825$

The Magnitude Response of the Filter (3D View)



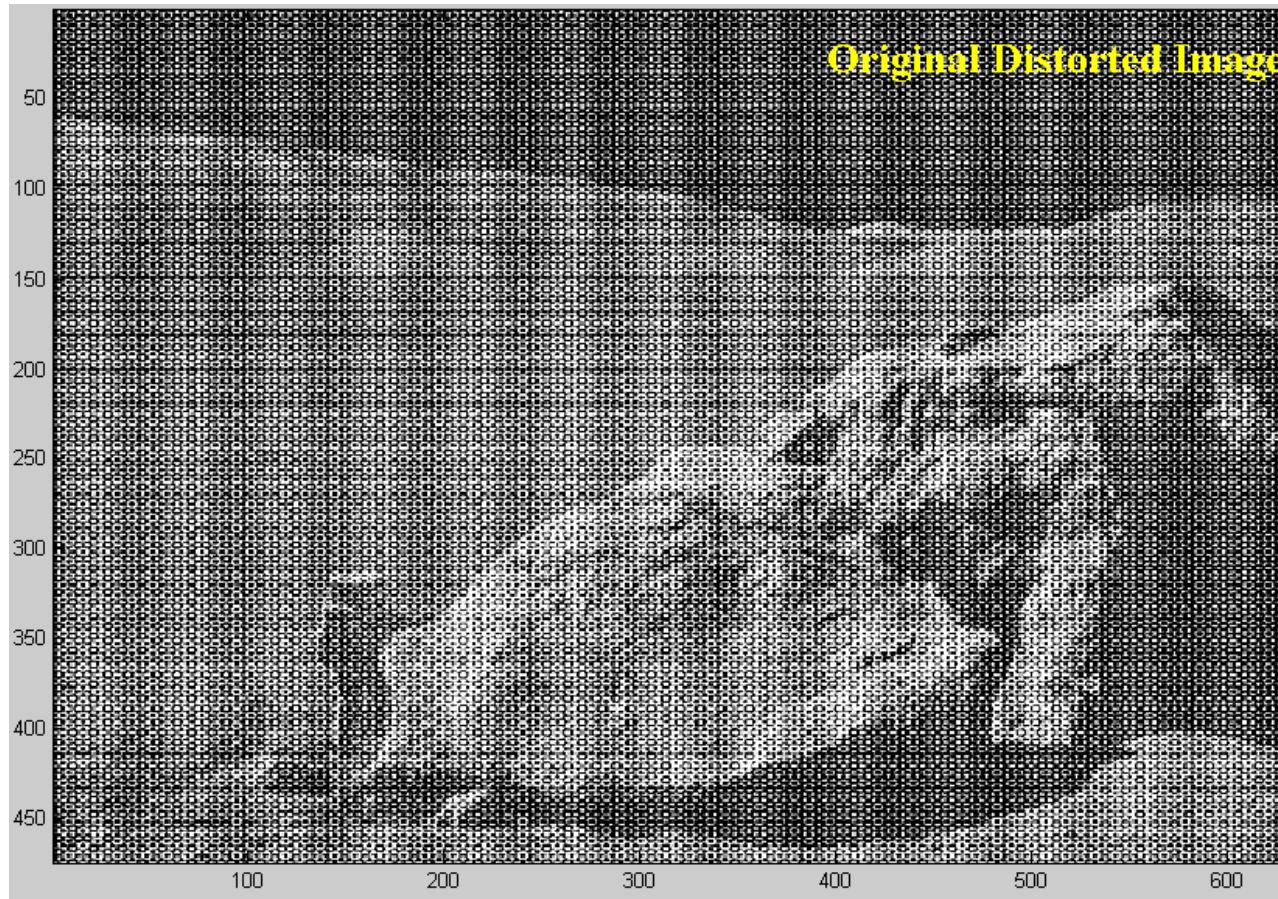
- Band reject filter
(here: notch filter)
- Attenuates image at the
normalised frequency
 $f_0 = 0.7825$

Transfer Function $H(x,y)$ (= the Filter Function's Spectrum)



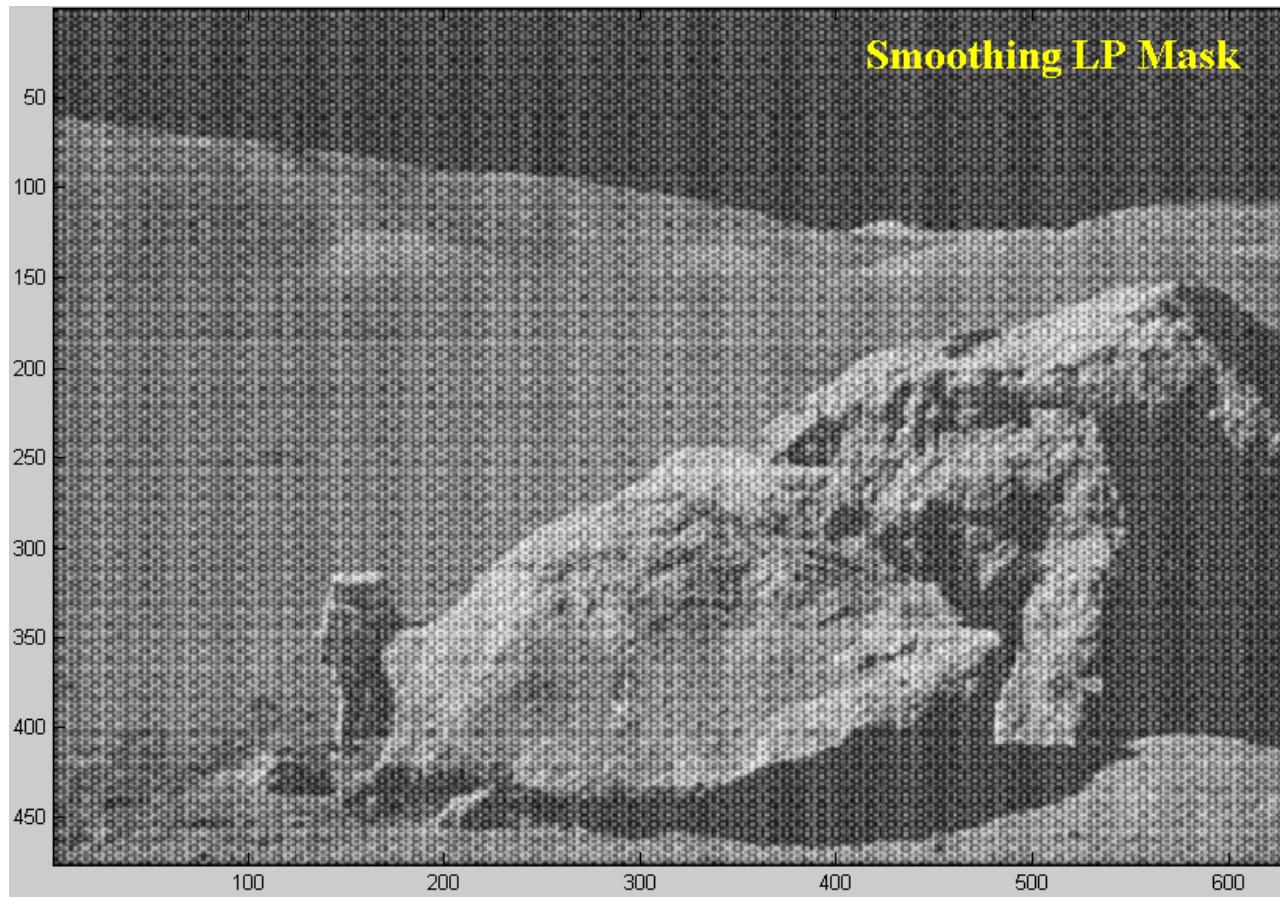
- Based on the convolution theorem
- $g(x,y) = f(x,y) * h(x,y)$ 
- $G(x,y) = F(x,y) \cdot H(x,y)$
- Simply multiply transfer function with the image's spectrum

I) Result: Original Image $f(x,y)$



- Attenuation of the periodic disturbing signal by using a simple band reject filter

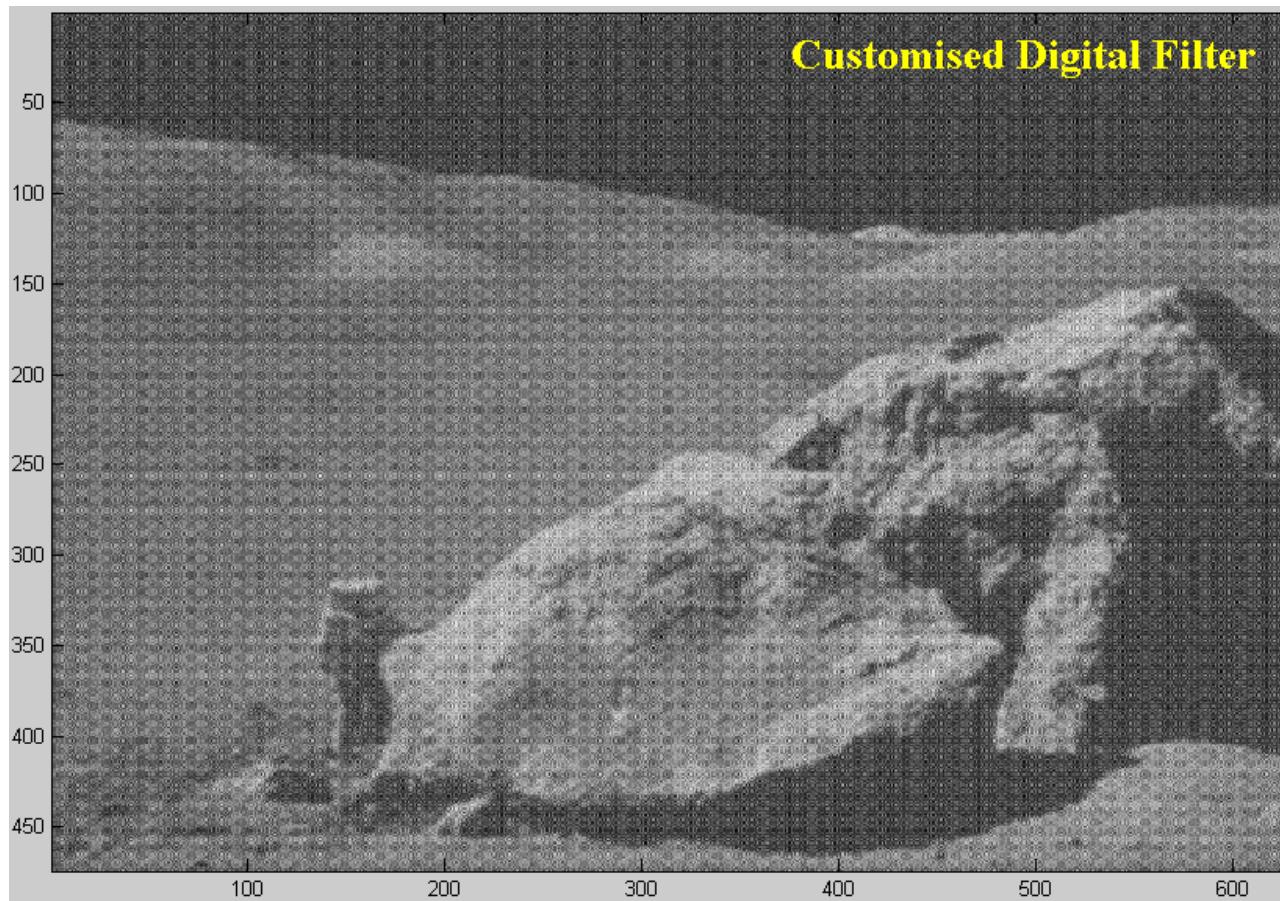
II) Result: Spatial Filtered Image $g(x,y)$ using the Mask



$$mask = \frac{1}{9} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

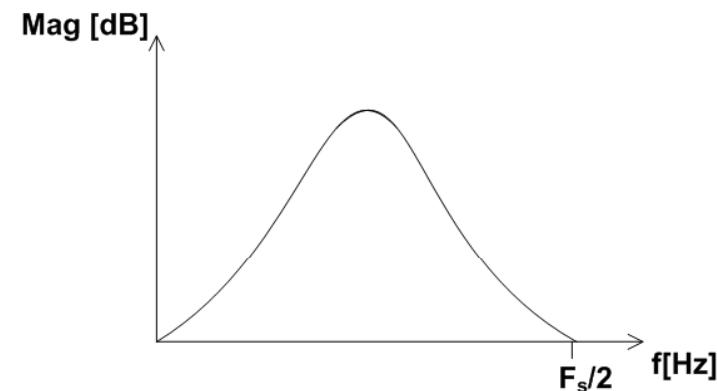
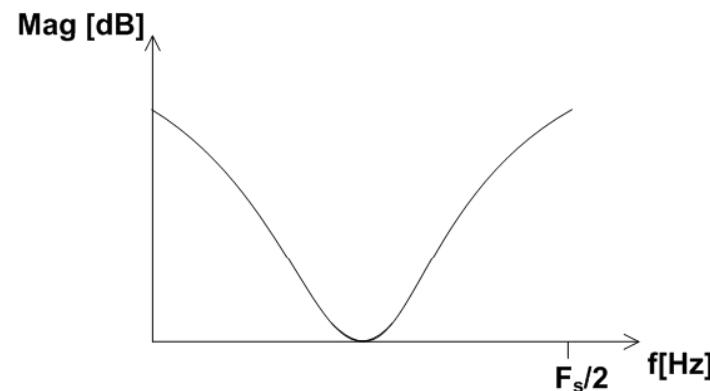
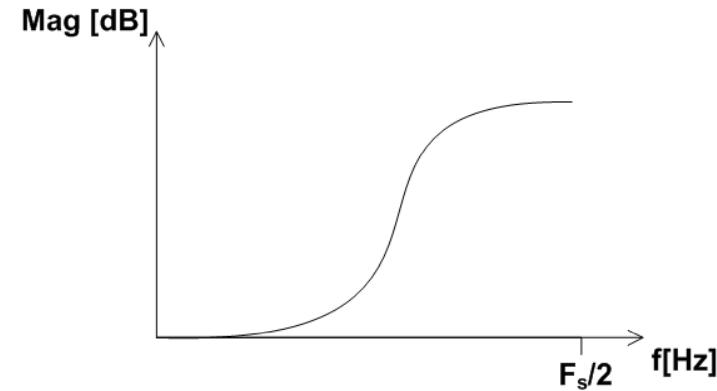
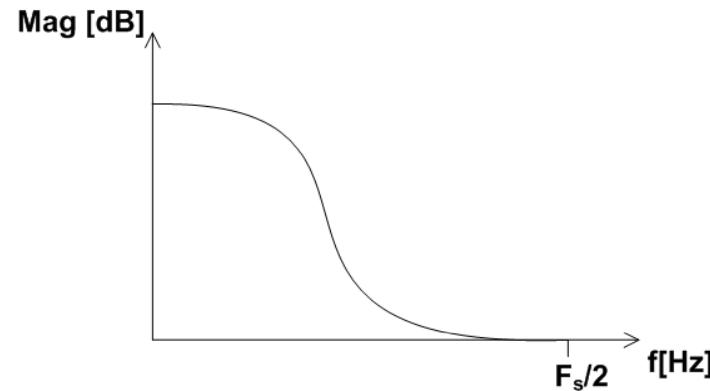
- Attenuation of the periodic disturbing signal by using a simple band reject filter

III) Result: Filtered Image $g(x,y)$ using the Digital Filter

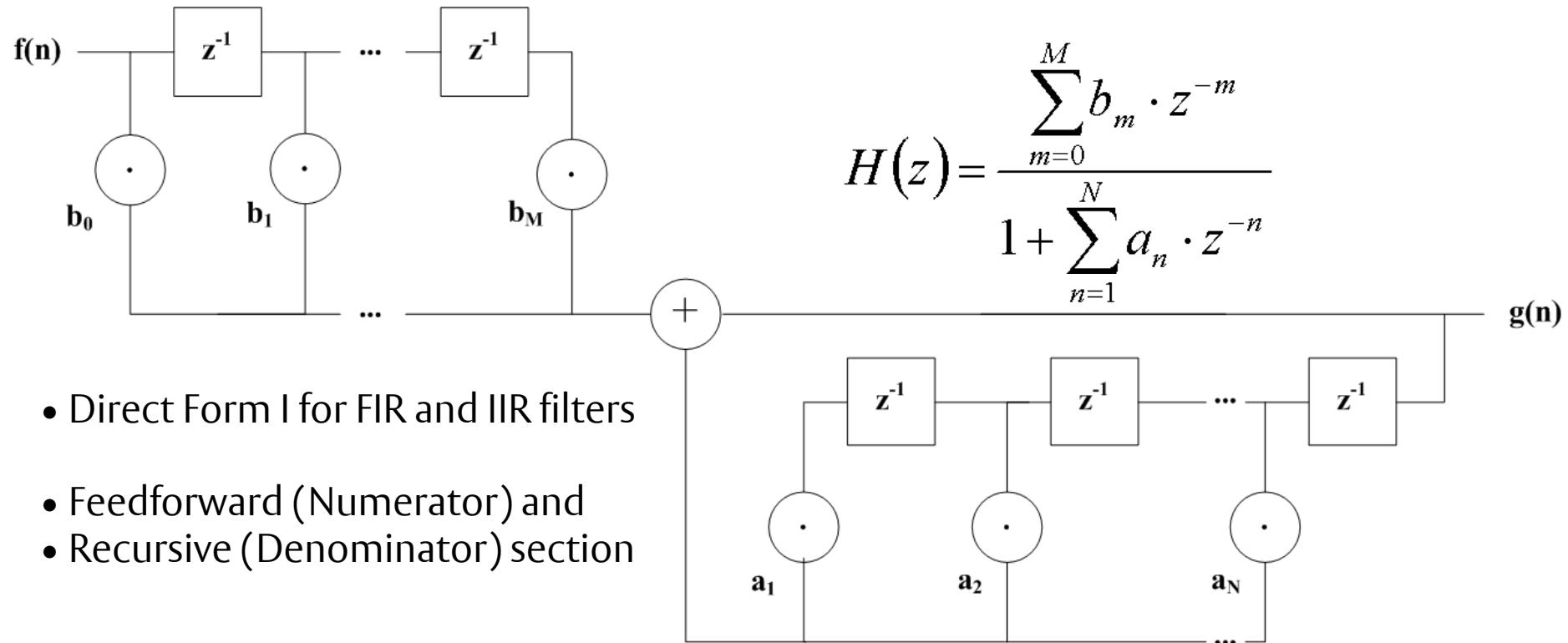


- Attenuation of the periodic disturbing signal by using a simple band reject filter

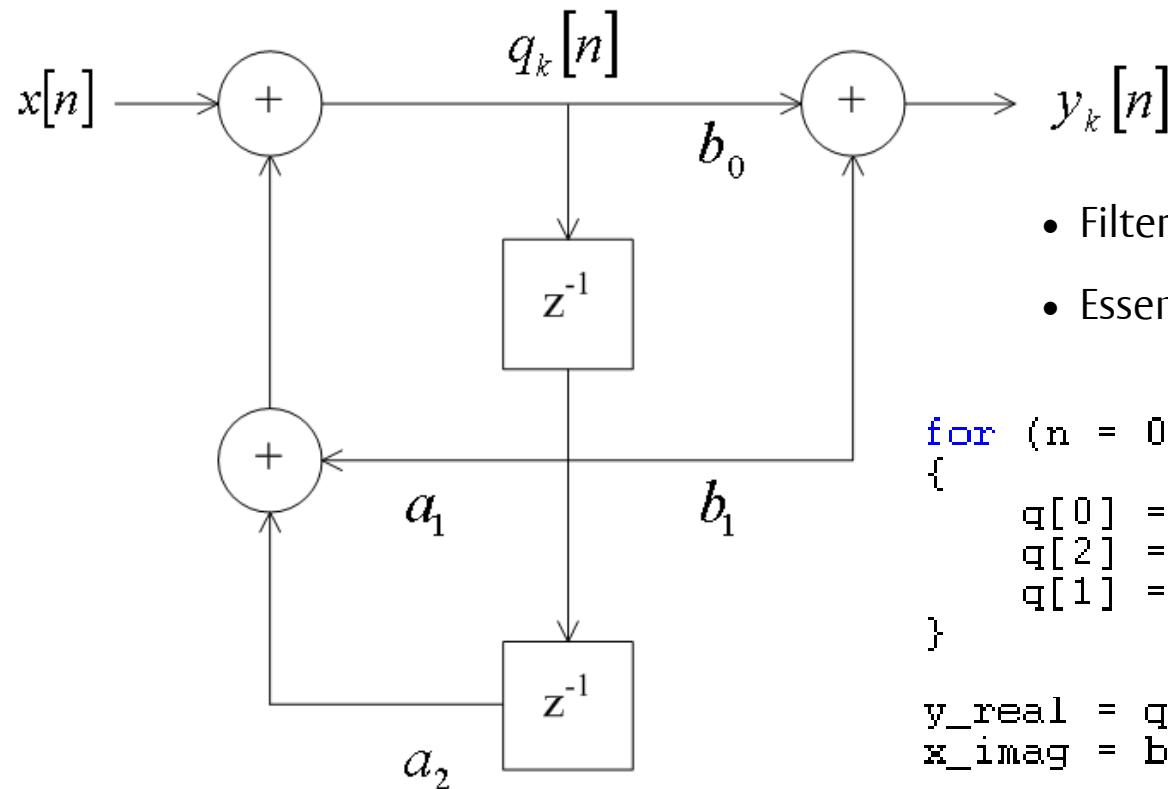
Filter Theory and Design – Basic Filter Types



Filter Theory and Design – Digital Filter Structure



Filter Theory and Design – Digital Filters in Practice



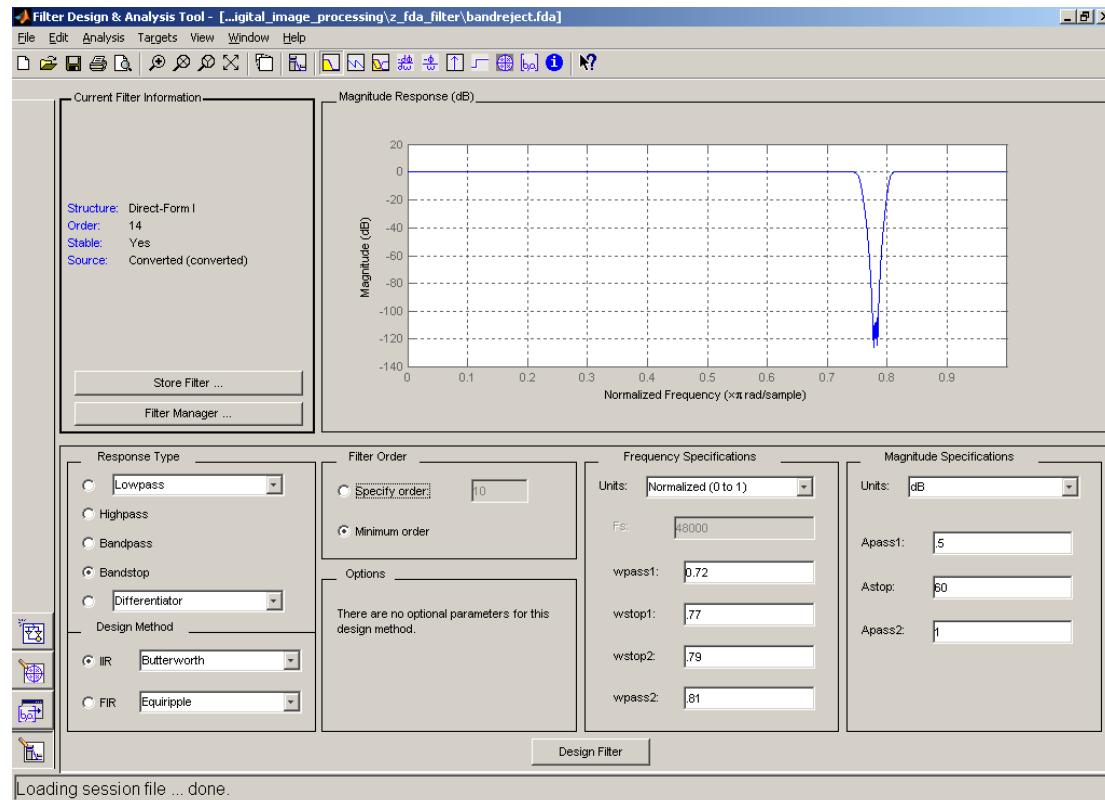
- Filter structure easy to implement
- Essential: Filter Coefficients

```

for (n = 0; n < length; n++)
{
    q[0] = a1 * q[1] - q[2] + x[n];
    q[2] = q[1];
    q[1] = q[0];
}

y_real = q[0] - b1_real * q[1];
x_imag = b1_imag * q[1];
  
```

Filter Theory and Design – The Filter Coefficients



- Pole-Zero-Diagram
- Tools from the WWW
- MatLab: `>> fdatool`
- Normed frequency to be attenuated: $f_0 \approx 0.78$

Case Study Conclusions

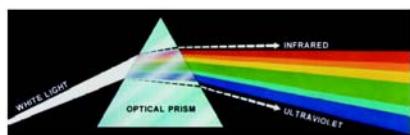
- Image enhancement is a necessary pre-processing step before segmentation
 - There are simple spatial masks for a rough enhancement
 - The use of customised digital filters is a more accurate alternative
- There are two steps in the design and implementation of Digital Filters
 - Fix the structure of the (FIR or IIR) filter in code
 - Compute the filter coefficient with a Design & Analysis tool (e.g. MatLab)
- However
 - Frequency domain techniques alone are insufficient
 - Take spatial domain and Wavelets into account

Overall Summary

- Overview provided of common frequency enhancement techniques
 - Introduction to Fourier Analysis
 - Application to Enhancement
 - Case Study

SE3IA11 Image Analysis

Colour Image Processing



Chapter 6 in the Gonzalez & Woods book

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

1

Contents in this topic

- Fundamentals of the physical nature of colour
- Colour models
- Pseudocolour images
- Colour transformations
- Colour segmentation
- Smoothing and sharpening in colour images
- Brief introduction of graphics formats

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

2

Colour fundamentals

- “Colour” here refers to chromatic light (visible light) in the electromagnetic spectrum from wavelength about 400 to 700 nm.
- Three basic quantities for chromatic light
 - Radiance (measured in watts): the total amount of energy from the light source.
 - Luminance (measured in lumens): a measure of the amount of energy that an observer perceives.
 - Brightness: a subjective descriptor for intensity (grey level).
- Three primary colours
 - Blue (B): 435.8 nm; Green (G): 546.1 nm; Red (R): 700 nm.

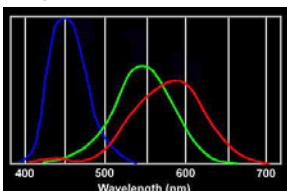
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

3

Three primary colours

- Human eyes do not perceive individual three primary colours separately, but integrate three types of colour receptor (cones) over parts of the spectrum.
- It is possible to characterise a psycho-visual colour by mixing **Red**, **Green**, and **Blue**.



Absorption of light in the human eye as a function of wavelength.
 Blue: 445
 Green: 535
 Red: 575

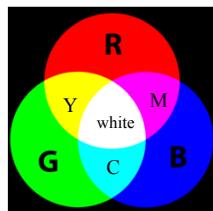
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

4

Combination of primary colours

- The secondary colours of light can be generated from R, G, and B.
- They are
 - Magenta (M): R+B
 - Cyan (C): G+B
 - Yellow (Y): R+G
- Additive R, G and B produce white light.



What would it be with additive M, C and Y?

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

5

Characteristics of colours

- Three characteristics of colour light
 - Brightness: intensity not related with colour
 - Hue: the dominant colour perceived by an observer
 - Saturation: the relative purity or the amount of white light mixed with a hue
- Hue and saturation taken together are called *chrominance* (or *chromaticity*).
- A colour can be characterised by *brightness* and *chrominance*.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

6

Tri-chromatic coefficients

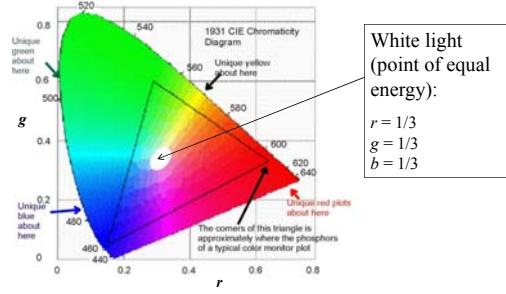
- R, G, and B represent the intensity of red, green and blue to form any particular colours, respectively.
 - Tri-chromatic coefficients, r , g , and b are defined as
- $$r = \frac{R}{R+G+B} ; g = \frac{G}{R+G+B} ; \text{ and } b = \frac{B}{R+G+B}$$
- It is noted that $r + g + b = 1$
 - For any value of r and g , the corresponding value of $b = 1 - (r + g)$.
 - R, G, B may be written as X, Y, Z, called tristimulus.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

7

CIE chromaticity diagram



- CIE: Commission Internationale de l'Eclairage (International Commission on Illumination)

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

8

Colour models – general

- Definition: a colour model is a specification of a coordinate system and a subspace within that system where each colour is represented by a single point.
- Examples of colour model usage
 - RGB model for colour monitors and colour video cameras
 - CMY model for colour printing
 - HSI model for applications where brightness and chrominance are processed separately.
 - YUV and YIQ models for colour TV systems
 - CIEL*a*b* model to approximate human vision

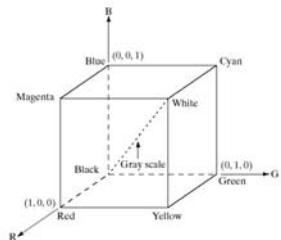
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

9

Colour models – RGB

- The RGB model is represented by a Cartesian coordinate system, mathematically normalised to $[0,1]$.
- The origin is Black.
- The diagonal, starting with Black and ending at White, represents grey scale.
- Different colours are defined by vectors extending from the origin.

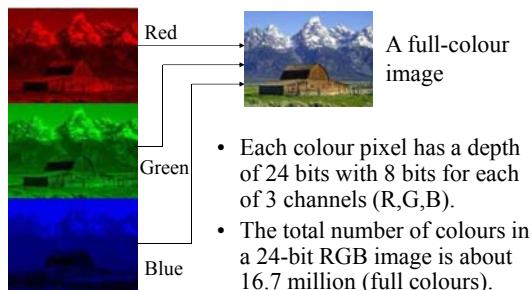


Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

10

Generating images from the 3 channels



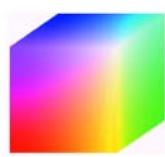
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

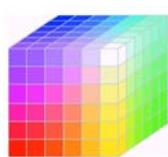
11

Safe colours v.s. full colours

- The 216 safe colours are used for web-page design in the RGB model.
- Each channel can only have values of
 - 0, 51, 102, 153, 204, 255 in decimal or
 - 00, 33, 66, 99, CC, FF in Hexdecimal.



24-bit RGB full colour cube



RGB safe colour cube

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

12

CMY or CMYK colour model

- The CMY model is a subtractive colour model, which is related to the secondary colours of light, and often used in colour printing.
 - The CMYK, called “four-colour printing”, has the fourth colour “black” added.
 - The CMY model can be converted from RGB.

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad \begin{array}{l} \text{Taking red out of white} \\ \text{Taking green out of white} \\ \text{Taking blue out of white} \end{array}$$

Values of R, G, and B are normalised to the range of [0,1].

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

13

The HSI colour model

- HSI stands for Hue, Saturation, and Intensity.
 - The HSI model is suited for describing colours.
 - Hue: a colour attribute describing a pure colour.
 - Saturation: a measure of the degree to which a pure colour is diluted by white light.
 - Intensity: a measure corresponding to the subjective descriptor of brightness. It is an achromatic notion.
 - Compared to RGB and CMY models, the HSI model is normally used for developing image processing algorithms based on colour descriptions.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

14

The RGB and HSI colour models

- The intensity axis in the RGB cube is the connected line of the white and black points.
 - Intensity* of a colour point: the intersection of the intensity axis with a plane perpendicular to it and containing the colour point.
 - Saturation*: a colour function of distance from the intensity axis.
 - Hue*: a plane determined by three points – black, white, and a colour point in the cube.

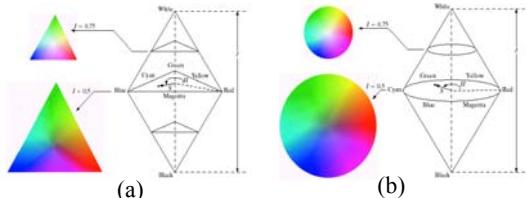
Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

15

The HSI space

- The HSI space is represented by a vertical intensity axis and colour points that lie on planes perpendicular to this axis.
 - In (a), triangular colour planes are used.
 - In (b), circular colour planes are used.



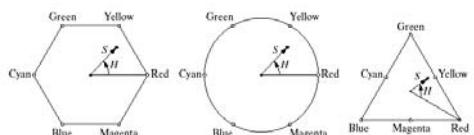
Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

16

Colour planes in the HSI space

- Values of Hue and Saturation of each colour point can be determined in colour planes.
 - Hue is defined as an angle from the red axis.
 - Saturation is the length of the vector from the origin to the colour point.



Autumn term 2015

SE31A11/SEMIR12 Image Analysis

17

Converting colours from RGB to HSI

Hue: H ; Saturation: S ; and Intensity: I

$$H = \begin{cases} \theta & \text{if } B \leq G \\ 360 - \theta & \text{if } B > G \end{cases} \quad \begin{matrix} R, G, B \text{ have} \\ \text{been normalised} \\ \text{to } [0,1]. \end{matrix}$$

$$\theta = \cos^{-1} \left\{ \frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2 + (R-B)(G-B)]^{1/2}} \right\}$$

$$S = 1 - \frac{3}{(R+G+B)}[\min(R, G, B)], \text{ and } I = \frac{(R+G+B)}{3}$$

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

18

Converting colours from HSI to RGB (1)

There are three sectors based on the value of H .

1. RG sector ($0^0 \leq H < 120^0$)

$$R = I[1 + \frac{S \cos H}{\cos(60^\circ - H)}]$$

$$B = I(1 - S)$$

$$G = 3I - (R + B)$$

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

19

Converting colours from HSI to RGB (2)

2. *GB sector* ($120^\circ \leq H < 240^\circ$), $H = H - 120^\circ$

$$R = I(1 - S)$$

$$G = I[1 + \frac{S \cos H}{\cos(60^\circ - H)}]$$

$$B = 3I - (R + G)$$

3. *BR sector* ($240^\circ \leq H < 360^\circ$), $H = H - 240^\circ$

$$G = I(1 - S)$$

$$B = I[1 + \frac{S \cos H}{\cos(60^0 - H)}]$$

$$R = 3I - (G + B)$$

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

20

The YUV colour model

- The YUV colour model is similar to HSI , in which Y represent the brightness and U, V for chrominance.
 - The model is used in PAL, NTSC, and SECAM composite colour video standards.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$Y \in [0, 1], \quad U \in [-0.436, 0.436], \quad V \in [-0.615, 0.615]$$

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

21

The YIQ colour model

- Similar to the YUV model, the YIQ colour model is only used in the NTSC colour TV system.
 - Y is for brightness and I, Q represent chrominance.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.595716 & -0.274453 & -0.321263 \\ 0.211456 & -0.522591 & 0.311135 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$R, G, B, Y \in [0, 1], \quad I \in [-0.5957, 0.5957], \quad Q \in [-0.5226, 0.5226]$$

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

22

The CIEL*a*b* colour model (1)

- The CIEL*a*b* colour model is a device independent model. This means that the colours are defined independent of their nature of creation or the device they are displayed on.
 - Its gamut covers the entire visible spectrum and can represent any monitor, printer, and input device.
 - Similar to the HSI model, the L*a*b* model decouples intensity (L*) and colour (a* - red minus green, and b* - green minus blue).

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

23

The CIEL*a*b* colour model (2)

- Let R, G, B be gray-level of a pixel; and R_w, G_w , and B_w reference white tristimulus values (CIE standard D65 illumination $r=0.3127$, and $g=0.3290$ in the CIE chromaticity diagram).
 - The $L^*, a^*,$ and b^* can be calculated as

$$\begin{aligned}
 L^* &= 116 \cdot h\left(\frac{G}{Gw}\right) - 16 & \text{where } h(q) = \begin{cases} \sqrt[3]{q} & q > 0.008856 \\ 7.787q + 16/116 & q \leq 0.008856 \end{cases} \\
 a^* &= 500\left[h\left(\frac{R}{Rw}\right) - h\left(\frac{G}{Gw}\right)\right] \\
 b^* &= 200\left[h\left(\frac{G}{Gw}\right) - h\left(\frac{B}{Bw}\right)\right]
 \end{aligned}$$

R, G, B are normalised to [0,1].

Autumn term 2015

SE31A11/SEMIR12 Image Analysis

34

Pseudocolour image processing

- What is pseudocolour image?
 - Colours of an image are added to each pixel based on a specified criterion rather than from a true colour observation of a scene.
- The operation involved is to assign colours to grey value pixels.
- Applications of pseudocolour image processing mainly contribute to visualisation.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

25

Assigning colours to grey images: intensity slicing (1)

- The information in a 2D grey image pixel consists of intensity (or grey level) and its x, y coordinates in the image.
- The grey image can be manipulated in a 3D space with intensity I as the other coordinate.
- Slicing the 3D space with planes parallel to the xoy plane in various intensity levels.
- Intensity between two adjacent levels is assigned a given colour.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

26

Assigning colours to grey images: intensity slicing (2)

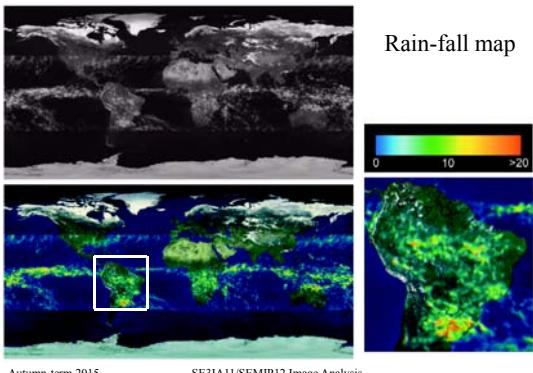
- In summary (in mathematical terms)
 - Let $[0, L-1]$ represent the grey scale; and
 - l_0 represent black ($f(x,y)=0$) and l_{L-1} represent white ($f(x,y)=L-1$).
 - P ($0 < P < L-1$) planes parallel to xoy at levels l_1, l_2, \dots, l_p , and divide the grey scale into $P+1$ intervals V_1, V_2, \dots, V_{P+1}
 - k colours c_k are assigned to the grey image, we have
$$f(x, y) = c_k \quad \text{if } f(x, y) \in V_k$$
- It is straightforward!

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

27

An example of intensity slicing



Colour transformation

- Transformation of colour images can be modelled as $g(x, y) = T[f(x, y)]$
 - where $f(x, y)$ is an input colour image, $g(x, y)$ the transformed output, and T is the transformation operator.
- Using r_i and s_i for variables denoting the colour components of $f(x, y)$ and $g(x, y)$ at point (x, y) , respectively, it can be expressed as $s_i = T[r_1, r_2, \dots, r_n] \quad i = 1, 2, \dots, n$
- Value n depends on what colour models are selected (process channels separately).
 - RGB: $n=3$, CMYK: $n=4$

Autumn-term 2015 SE3IA11/SEMIP12 Image Analysis 29

Example of colour transformation (1)

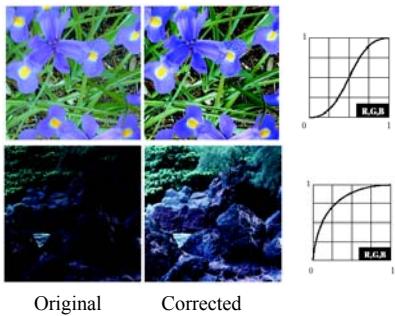
- Linear transformation in the RGB space.
 - The transformation is done in $s_i = kr_i \quad i = 1, 2, 3$ where $k = 0.7$
 - r_1, r_2 , and r_3 represent R, G and B, respectively.
 - s_1, s_2 , and s_3 correspond to the transformed output.



Autumn-term 2015 SE3IA11/SEMIP12 Image Analysis 30

Examples of colour transformation (2)

- Non-linear transformation in the RGB space



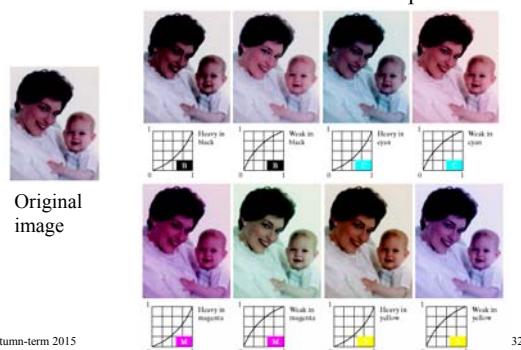
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

31

Example of colour transformation (3)

- Non-linear transformation in the CMYK space



Autumn-term 2015

32

The colour vector space

- A colour pixel $c(x,y)$ can be represented by a vector as

$$\bar{c}(x,y) = \begin{bmatrix} R(x,y) \\ G(x,y) \\ B(x,y) \end{bmatrix}$$

- For an image of size $M \times N$, there are $M \times N$ such vectors.
- Results of processing R, G, and B separately are not always equal to those from processing the equivalent colour vectors.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

33

Colour segmentation – 1

- Segmentation in the HSI colour space
 - Separation of chromatic information without intensity influence.
 - Processes on individual planes which are perpendicular to the intensity axis.
 - Application: human skin detection
 - Information of “hue” can be used for skin colour detection. This has been used in face and hand detection in human machine interaction.
 - More applications...

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

34

Colour segmentation – 2

- Segmentation in the RGB vector space
 - Supervised segmentation which needs a sample RGB vector as a reference.
 - Compare each colour vector of the image to the reference vector, and the distance between two vectors decides the segmentation result.
 - A specified threshold is needed for the distance to judge “true” or “false”.
 - Applications: separate objects purely based on RGB information.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

35

Mahalanobis Distance

- **Mahalanobis distance** is used to identify similarity between an unknown sample set to a known one. Here the sample set is represented by a feature vector.
 - It is based on correlations between two samples.
 - For two feature vectors $v = (v_1, v_2, \dots, v_N)^T$ and $u = (u_1, u_2, \dots, u_N)^T$, the Mahalanobis distance between them is defined as

$$D_M(v, u) = \lceil (v - u)^T C^{-1} (v - u) \rceil^{1/2}$$

where C is the covariance matrix of v and u .

Autumn term 2015

SE31A11/SEMIR12 Image Analysis

36

Colour segmentation – forest inspection

- Try to design algorithms for the following purposes (the L*a*b* model may be used).

Identify burnt area in the forest



Identify smoke/forest fire in the image



Identify bare soil/dead forest



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

37

Smoothing and sharpening

- You have learned detailed techniques and algorithms for image smoothing and sharpening in the topic of image enhancement.
- For colour images, smoothing and sharpening are operated in the colour vector space, *i.e.*

$$\text{smoothing} \quad \bar{c}(x, y) = \begin{bmatrix} \frac{1}{K} \sum_{(x,y) \in S_{x,y}} R(x, y) \\ \frac{1}{K} \sum_{(x,y) \in S_{x,y}} G(x, y) \\ \frac{1}{K} \sum_{(x,y) \in S_{x,y}} B(x, y) \end{bmatrix} \quad \text{and sharpening} \quad \nabla^2[C(x, y)] = \begin{bmatrix} \nabla^2 R(x, y) \\ \nabla^2 G(x, y) \\ \nabla^2 B(x, y) \end{bmatrix}$$

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

38

Operations of colour image: noise removal and edge detection

- For noise removal, if different noise models are applied to three channels (R, G, B), noise removal can be operated to each of them separately. If there is only one channel is corrupted by noise, when transform to the HSI space, the three channels (H, S, I) are all affected.
- For edge detection based on gradient operation, the colour vector space should be considered. It is not working to separate channel operations.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

39

Image (graphics) formats: GIF

GIF87a, GIF89a:

- Graphics Interchange Format (GIF) devised by the UNISYS Corp. and Compuserve, initially for transmitting graphical images over phone lines via modems.
 - Uses the Lempel-Ziv Welch algorithm for compression.
 - Supports only 8-bit (256) colour images.
 - GIF89a supports simple animation.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

40

Image (graphics) formats: JPEG

- A standard for photographic image compression created by the Joint Photographic Experts Group (JPEG).
 - Takes advantage of limitations in the human vision system to achieve high rates of compression.
 - Lossy compression which allows user to set the desired level of quality/compression.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

41

Image (graphics) formats: TIFF

- Tagged Image File Format (TIFF), stores many different types of images (e.g., monochrome, grayscale, 8-bit & 24-bit RGB, etc.).
 - Developed by the Aldus Corp. in the 1980's and later supported by Microsoft .
 - TIFF is a lossless format (when not utilising the new JPEG tag which allows for JPEG compression).
 - It does not provide any major advantages over JPEG and is not as user-controllable.
 - It appears to be declining in popularity.

Autumn-term 2015

SF31A11/SFM1P12 Image Analysis

42

Image (graphics) formats: PDF and BMP

Postscript/ PDF:

- A typesetting language which includes text as well as vector/structured graphics and bit-mapped images.
 - Does not provide compression, files are often large.

Windows (BMP):

- A system standard graphics file format for Microsoft Windows.
 - It is capable of storing 24-bit bitmap images.
 - Used in PC Paintbrush and other programs.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

43

Image (graphics) formats: PNG

- The Portable Network Graphics (PNG) format was designed to replace the older and simpler GIF format and, to some extent, the much more complex TIFF format.
 - Advantages over GIF:
 - Alpha channels (variable transparency)
 - Gamma correction (cross-platform control of image brightness)
 - Better Compression (5-25% better)
 - Features:
 - Supports three main image types: *truecolour*, *greyscale* and *palette-based* ('8-bit'). JPEG only supports the first two; GIF only the third.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

44

Image (graphics) formats: CGM and SVG

- CGM stands for Computer Graphics Metafile, which is an open international standard file format for 2D vector and raster graphics, as well as text, defined by ISO/IEC 8632.
 - It is independent from software and hardware.
 - SVG stands for Scalable Vector Graphics, which is an XML-based vector image format for 2D graphics.
 - Developed by the World Wide Web Consortium
 - Supported by all major modern web browsers

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

45

Questions for further thinking

- How to use different colour models in practice? Give examples.
 - What is the relationship of colours in various colour spaces? (converting between colour models)
 - How to assign colours to a grey-level image?
 - How to use colour transformation techniques to enhance images?
 - Think about algorithms in image segmentation based on colour information.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

46

End of the two lectures

- Summary what you have learned in the two lectures.
 - Try to use different colour models to represent human skin colours.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

47

SE3IA11/SEMIP12 Image Analysis

Image Compression - The Fundamentals

Lecturer:

Prof. James Ferryman Computational Vision Group

Email: j.m.ferryman@reading.ac.uk

Image Compression

- **The problem:**

Consider the storage and transmission of a 60min full-motion colour movie (images are 8-bit and of size 512x512):

Storage: $512 \times 512 \times 3 \times 25 \times 3600 \sim 71\text{GB!}$

or 120 CDROMs or 16 DVDs ...

Transmission: downloading at $10\text{M/s} \sim$ about 2 hours!

On top of the above is the cost associated with audio data!

- **The solution:**

Compress the images by throwing out redundant and insignificant data

Another Example

- A Landsat D satellite broadcasts 85×10^6 bits / s
- A typical image from one pass contains 6100×6100 pixels in 7 spectral bands ~ 260 Mb image data
- Japanese Advanced Earth Observing Satellite (ADEOS) has spatial resolution of 8 metres for the polychromatic band and 16 metres for the multispectral bands has a transmitted data rate of 120 Mbps

Satellite Imagery

- How would you compress the following?

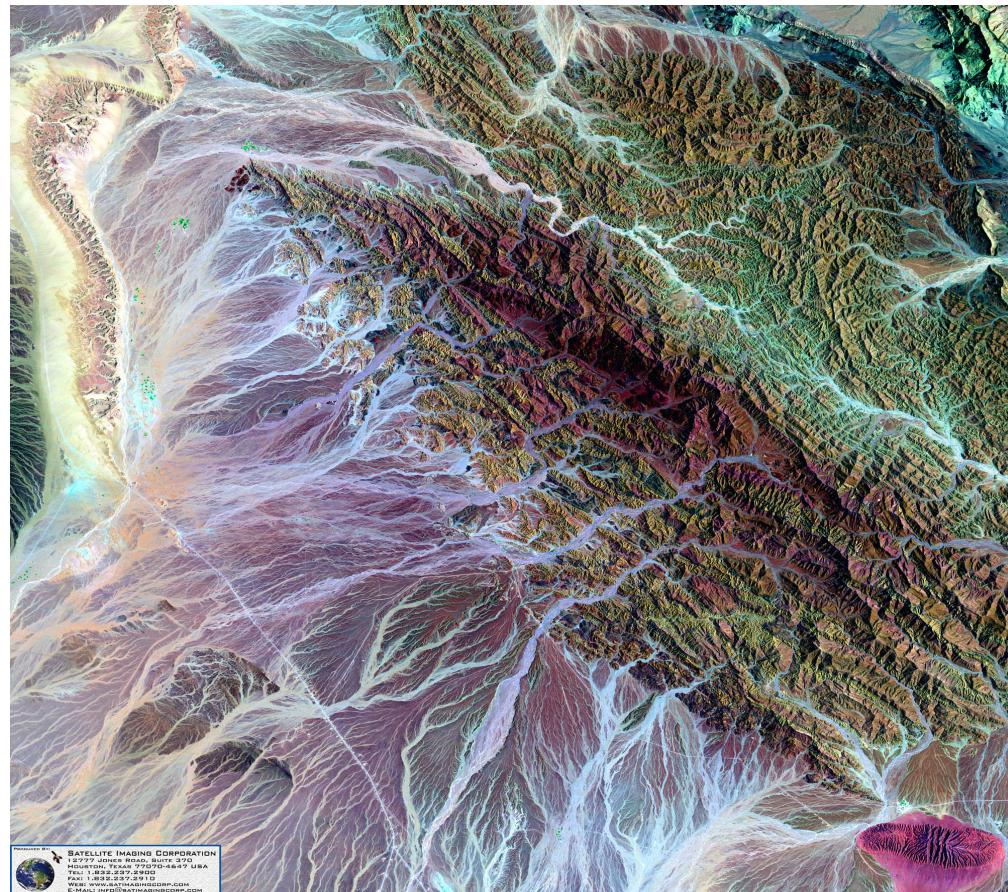


Image Compression

- Compressing an image is significantly different from compressing raw binary data
 - Naturally, you may use a general purpose compression program but the result is less than optimal
- Images contain certain statistical properties which can be exploited by encoders specifically designed for them
- Additionally, some of the finer image detail can be sacrificed for the sake of saving more bandwidth/ storage space

Data Redundancy

- Three types of redundancies associated with the digital coding of intensity values, intensity correlations among neighbouring pixels, and the psychovisually insignificant image data

Coding redundancy

In a digital 8-bit image, all 256 intensity values are represented using 8 bits, i.e. the digital codes for all values are of the same length (8), So for value 0, you have 00000000; value 1 you have 00000001; etc.

The above is based on an important assumption that all intensity values are equally probable in the image

This is a VERY unrealistic assumption (look at the intensity histogram of any natural image you come across to see whether it's really flat).

Coding Redundancy

- A better code design method is to let the length of the codes be linked to the probability of the intensity values
- For example, shorter codes for more probable values and longer codes for less frequent ones
- Consider the following:

0	2	2
1	2	3

Intensity values

01	10	10
01	10	11

Fixed-length codes
(no. of bits: 12)

10	0	0
10	0	110

Variable-length codes
(no. of bits: 10)

- Note this does not involve any change of image content. It merely changes the representation (the codes) of the intensity values in the computer

Spatial Redundancy

- Images often include large areas of similar intensity values
- Intensities of surfaces of real-world objects usually change fairly slowly
- The intensity of a pixel can be predicted rather accurately based on intensity values of neighbouring pixels – spatial redundancy



Psychovisual Redundancy

- The eye is a strange optical device. It does not respond to all visual information with equal sensitivity
- Some visual information is simply unnoticeable to the naked human eyes
- If the ultimate information receiver is the human eyes (such as watching a movie), then there is no point to process such *psychovisually redundant* information
- Psychovisual redundancy is fundamentally different from coding and spatial redundancy as its removal always incurs the loss of information (though psychovisually insignificant)

Psychovisual Redundancy



Image
represented in
256 greylevels



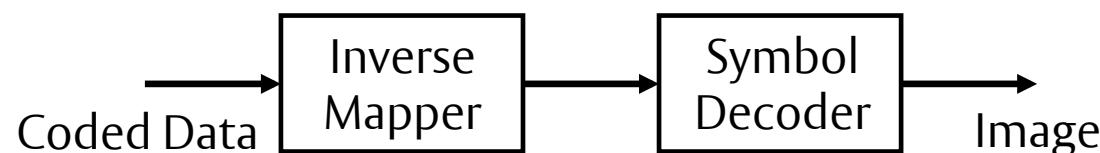
Image
quantized to 16
greylevels

General Encoder/Decoder Models

- Compression techniques attempt to remove one or more of the above redundancies according to the following general model:



Encoder Model



Decoder Model

General Encoder/Decoder Models

- The *mapper* maps the image from the spatial domain representation to a representation which makes spatial redundancy more accessible. The mapping (e.g. DCT) is usually invertible so no information is lost
- The *quantizer* represents mapper outputs with a reduced accuracy, especially those outputs which correspond to psychovisual redundancy. This is an often irreversible operation so it must not be included if error-free compression is required
- The *symbol encoder* assigns codes to the quantized output values. It removes coding redundancy by allocating short codes to more frequent quantized values and long ones to less frequent values
- The *decoder model* is the opposite of the encoder model but has no de-quantizer (since quantization is irreversible)

Lossless vs. Lossy

- *Lossless compression* – incurs no information loss
 - Involves compressing data, which when decompressed, will be an exact replica of the original data
 - Appropriate for data that has been expensive to capture (e.g. triband satellite images)
 - Need to be exactly reproduced when decompressed
- *Lossy compression* – some information is lost and cannot be recovered
 - *Approximation of the original image is often sufficient for most purposes*
 - *Error between original and compressed image must be tolerable*

Compression

- The usual steps involved in compressing an image are:
 1. Specify the *rate* (bits available) and *distortion* (tolerable error) parameters for the target image
 2. Dividing the image data into various classes, based on their importance
 3. Dividing the available bit budget among the classes, such that the distortion is a minimum
 4. Quantize each class separately using the bit allocation information derived in Step 3
 5. Encode each class separately using an entropy coder and write to file

Decompression

- Reconstructing an image from compressed data is usually a faster process than compression
- The steps involved are:
 1. Read in the quantized data from the file, using an entropy decoder (reverse of Step 5 on previous slide)
 2. Dequantize the data (reverse of Step 4 on previous slide)
 3. Rebuild the image (reverse of Step 2 on previous slide)

Codec Performance Measures

- Three important factors to consider when choosing a compression technique
 1. Fidelity measures the quality of the compressed-then-decompressed image. Objective measures include the *root-mean-square error* E_{rms} between original image $f(x,y)$ and decompressed image $\hat{f}(x,y)$

$$E_{rms} = \sqrt{\frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} [f(x,y) - \hat{f}(x,y)]^2}$$

and the *mean-square-to-noise ratio* SNR_{ms}

$$SNR_{ms} = \frac{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)^2}{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} [f(x,y) - \hat{f}(x,y)]^2}$$

Codec Performance Measures

- Mean Square Error (MSE) represents cumulative squared error between compressed and original image
 - Lower value represents less error
- Signal Noise Ratio (SNR)
 - High value represents less error

Codec Performance Measures

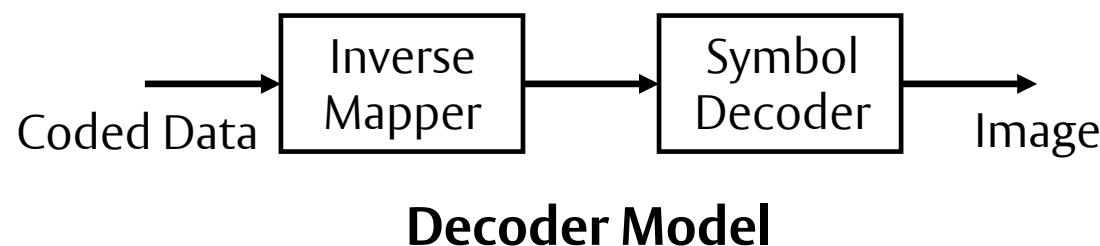
- Subjective measures include absolute ratings (excellent, fine, passable, marginal, inferior, unusable), and relative ratings (much worse, worse, slightly worse, the same, slightly better, better, much better)
2. Compression ratio
CR = number of bits of original image divided by that of compressed image
3. Computational cost
Computation required to compress and decompress an image

Encoder/Decoder Model

- Compression techniques attempt to remove one or more of the above redundancies according to the following general model:



Encoder Model

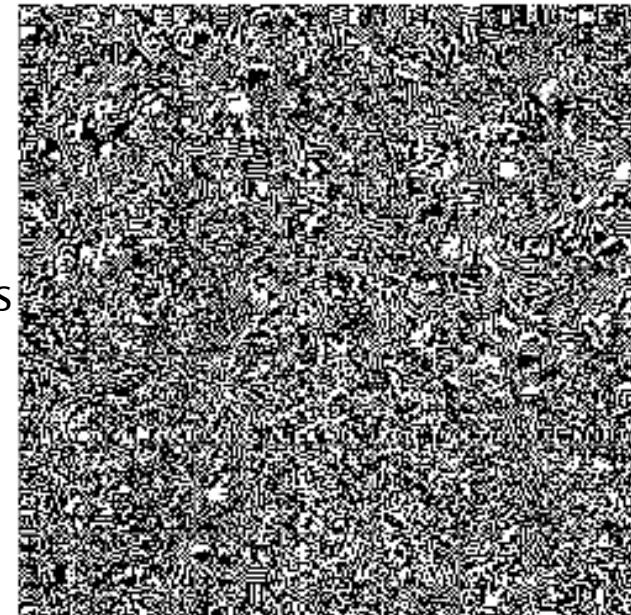


Decoder Model

Mapper



Each 8x8 block
processed with
DCT to obtain
DCT coefficients

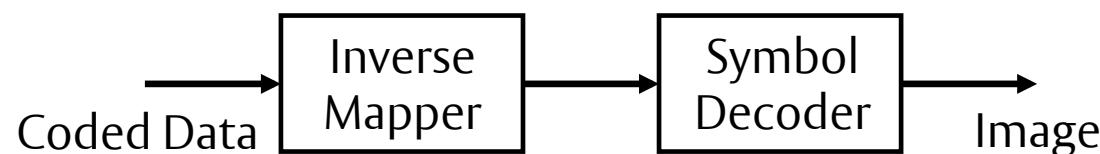


Encoder/Decoder Model

- Compression techniques attempt to remove one or more of the above redundancies according to the following general model:



Encoder Model



Decoder Model

Quantization

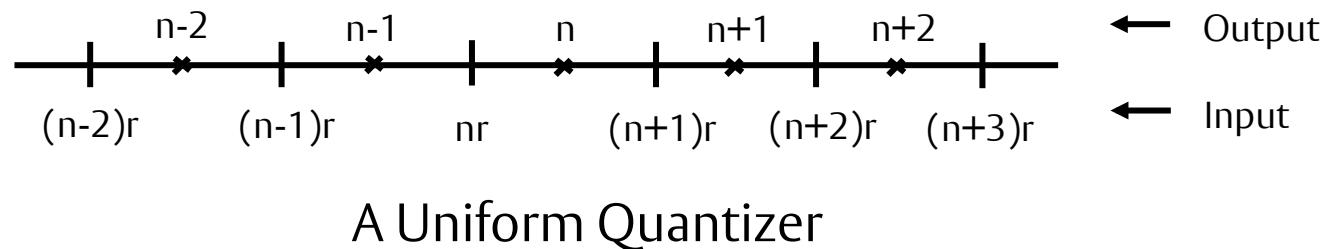
- Refers to process of approximating continuous set of image data values with finite (preferably small) set of values
- Input to quantizer is original data, output is always one among a finite number of levels
- Quantizer is a function whose set of output values are discrete, and usually finite
- Essentially a process of approximation
- A good quantizer is one which represents original signal with minimum loss of information

Quantization

- There are two types of quantization
 - Scalar Quantization
 - Vector Quantization
- In Scalar Quantization, each input symbol is treated separately in producing the output
- In Vector Quantization, the input symbols are grouped together into “vectors”, and processed to give output

Scalar Quantization

- A quantizer can be specified by its input partitions and output levels (also called *reproduction points*)
- If the input range is divided into levels of equal spacing, then quantizer is termed *Uniform Quantizer*, otherwise a *Non-Uniform Quantizer*
- A Uniform Quantizer can be easily specified by its lower bound and the step size, and is also easier to implement

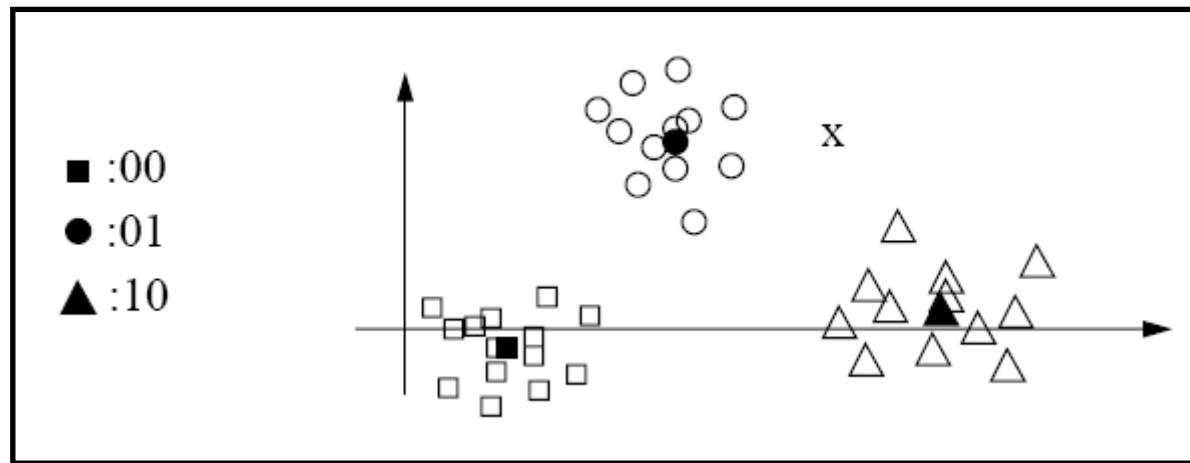


- If the input falls between $n*r$ and $(n+1)*r$, the quantizer outputs the symbol n

Vector Quantization

- Divide image into small blocks (e.g. $n \times n$) and regard the n^2 intensity values as forming a n^2 -dimension vector (so each block is a point in the n^2 -dimension space)
- Describe each cluster of such vectors/points by a representative prototype vector
 - e.g. a vector located at the centre of the clusterand represent the prototype vector using a pre-defined codeword
- Use of prototype increases optimality of vector quantizer, but at expense of increased computational complexity

Vector Quantization



- Compression is achieved by representing each vector using the codeword of the nearest prototype vector
- Decompression is performed by look-up of a codeword table

Dequantization

- Receives output levels of a quantizer and converts them into normal data
 - Translates each level into a *reproduction point* in the actual range of data
- The optimum quantizer (encoder) and optimum dequantizer (decoder) must satisfy the following conditions:
 - Given the output levels or partitions of the encoder, the best decoder is one that puts the reproduction points x' on the centres of mass of the partitions. Known as the *centroid condition*
 - Given the reproduction points of the decoder, the best encoder is one that puts the partition boundaries exactly in the middle of the reproduction points, i.e. each x is translated to nearest reproduction point. Known as *nearest neighbour condition*
- The quantization error ($x - x'$) is used a measure of the optimality of the quantizer and dequantizer

Bit Allocation

- First step in compressing image is segregation of data into different classes
- Depending on importance of data it contains, each class is allocated a portion of total bit budget, such that compressed image has minimum possible distortion
- This procedure is called *Bit Allocation*
- The Rate-Distortion theory (RDT) – Shannon - is often used for solving problem of allocating bits to set of classes
- Theory aims at reducing distortion for given target bitrate, by optimally allocating bits to various classes of data

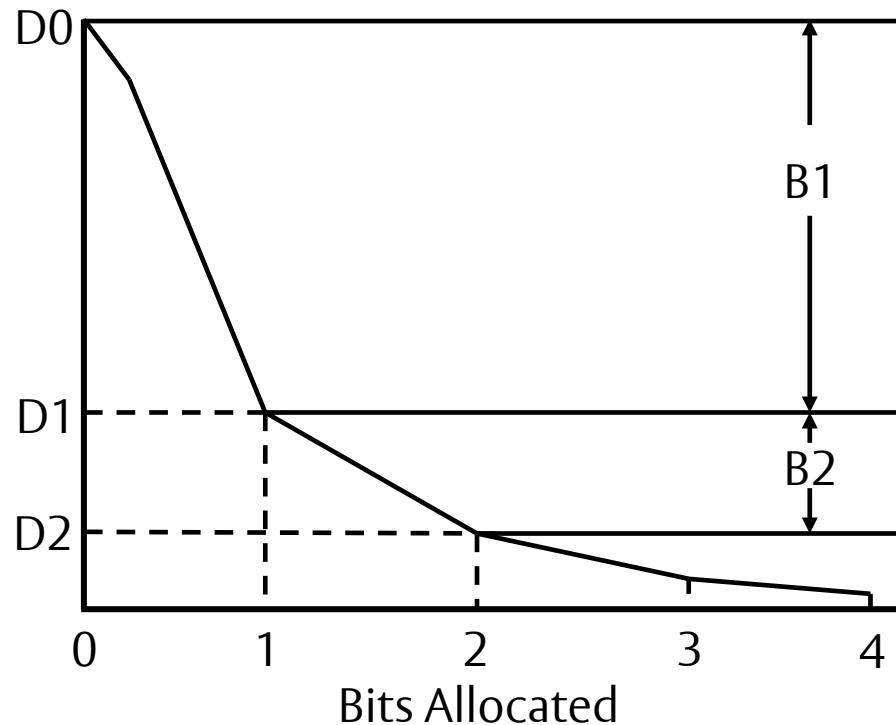
Optimal Bit Allocation using RDT

1. Initially, all classes are allocated a maximum number of bits
2. For each class, one bit is reduced from its quota of allocated bits, and the distortion due to the reduction of that 1 bit is calculated
3. Of all classes, the class with minimum distortion for a reduction of 1 bit is noted, and 1 bit is reduced from its quota of bits
4. The total distortion for all classes D is calculated
5. The total rate for all classes is calculated as $R = p(i) * B(i)$, where p is the probability and B is the bit allocation for each class
6. Compare the target rate and distortion specifications with the values obtained above. If not optimal, go to Step 2.

Optimal Bit Allocation using RDT

- In the approach, one bit at a time is reduced until optimality is achieved either in distortion or target rate, or both
- An alternate approach is to initially start with zero bits allocated for all classes, and to find the class which is most *benefitted* by obtaining an additional bit
- The *benefit* of a class is defined as the decrease in distortion for that class

Optimal Bit Allocation using RDT



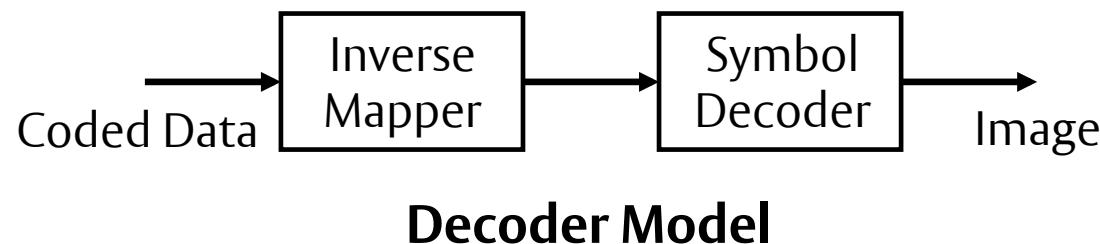
- A benefit of a bit is a decreasing function of the number of bits allocated previously to the same class

Encoder/Decoder Model

- Compression techniques attempt to remove one or more of the above redundancies according to the following general model:



Encoder Model



Decoder Model

Symbol (Entropy) Coding

- After the data has been quantized into a finite set of values, it can be encoded using an Entropy Coder to provide additional compression
- Entropy refers to the amount of information present in the data
- If an image has G greylevels, and the probability of greylevel k is $P(k)$, then Entropy H_e , not considering correlation of greylevels, is defined as

$$H_e = - \sum_{k=0}^{G-1} P(k) \log_2(P(k))$$

- A good estimate of entropy is usually not available
- Image entropy can however be estimated from a greylevel histogram

Entropy

- Let $h(k)$ be the frequency of greylevel k in an image f , with $0 \leq k \leq 2^b - 1$, and let the image size be $M \times N$
- The probability of occurrence of greylevel k can be estimated as

$$\tilde{P}(k) = \frac{h(k)}{MN}$$

and the entropy can be estimated as

$$H_e = - \sum_{k=0}^{2^b-1} \tilde{P}(k) \log_2(\tilde{P}(k))$$

Entropy Coding

- An entropy encoder encodes the given set of symbols with the minimum number of bits required to represent them
- Two of the most popular entropy coding schemes are:
 - Huffman Coding
 - Arithmetic Coding

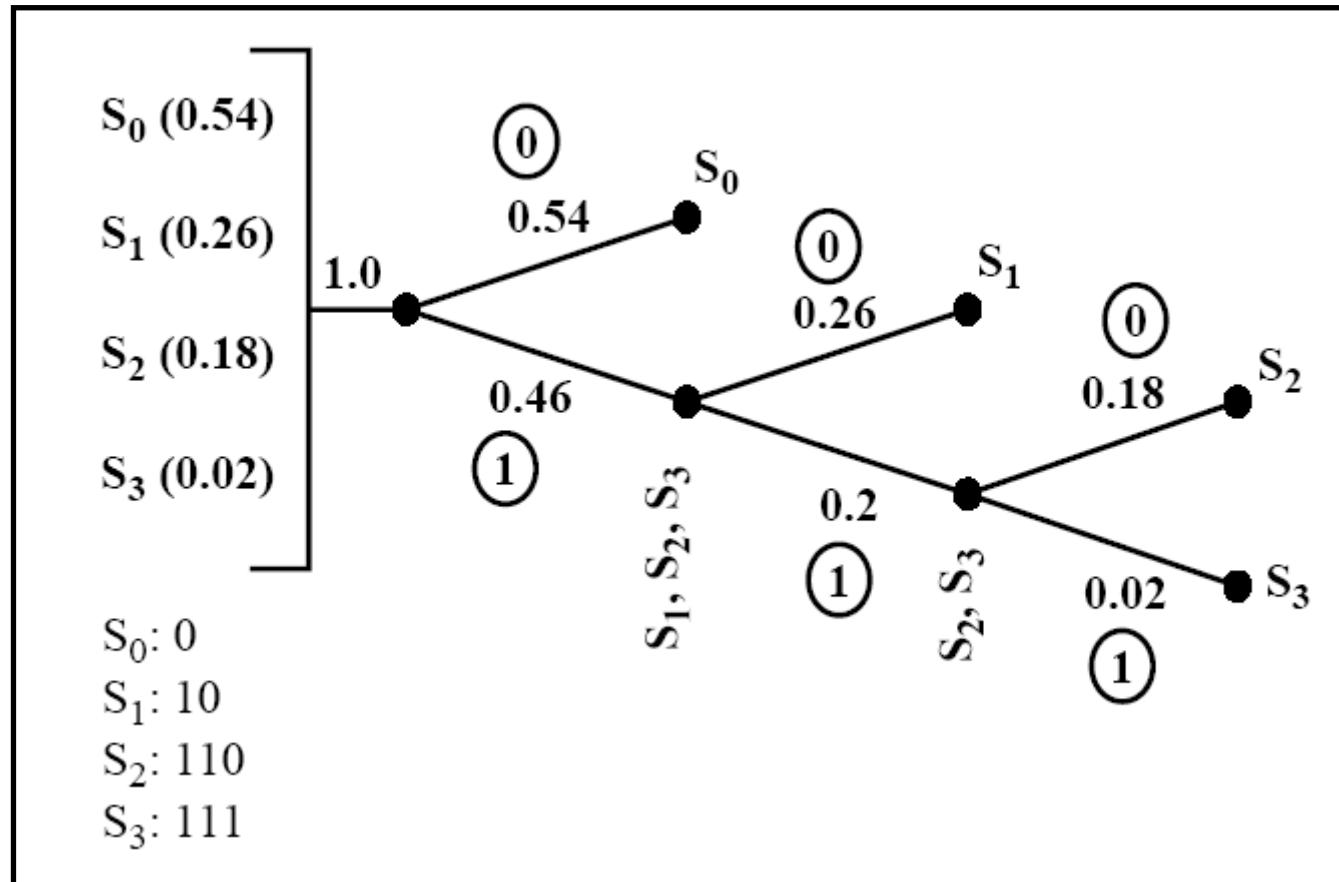
Huffman Coding

- A widely used technique for lossless compression (also called entropy coding); simply by removing coding redundancy (i.e. shorter codes for more frequent symbols, long ones for less frequent symbols)

Generating Huffman Codes

- Recursively divide symbols into two groups, one containing only the most frequent symbol and the other the remaining
- At each level of the tree, assign 0 to the more frequent group and 1 to the other
- Huffman codes are obtained by traversing the tree from root to symbols by recording 0s and 1s encountered

Huffman Coding



Huffman Coding

- Huffman codes are uniquely decodable by means of lookup tables.
- A string of Huffman codes can only have one valid interpretation
- In the example on the previous slide, the string 110100111111 can only be decoded as the symbol sequence $S_2S_1S_0S_3S_3$.

Truncated Huffman Coding

- If the number of symbols is large, generating the Huffman codes can be a time-consuming process and codes for some symbols can be very long
- A solution to this is to represent only the most frequent symbols using Huffman codes and the others using the usual fixed-length codes prefixed by a suitable Huffman code – Truncated Huffman Coding

Problems with Huffman Coding

- Huffman codes have to be an integral number of bits long
 - For example, if the probability of symbol is $1/3$, the optimum number of bits to code that symbol is around 1.6
 - The Huffman coding scheme has to assign either 1 or 2 bits
 - Either choice leads to a longer compressed image than is theoretically possible
- In non-adaptive data compression, a single pass over data is required to collect statistics; data is then encoded using statistics; decoder needs a copy of the statistics, which generates an overhead
- In adaptive data compression, both encoder and decoder start with statistical model in same state; models updated after each symbol processed
- Problem with combining adaptive modelling with Huffman coding is that rebuilding Huffman tree is a very expensive process

Arithmetic Coding

- Respectable candidate to replace Huffman coding
- Completely bypasses idea of replacing input symbol with specific code
- Takes a stream of input symbols and replaces it with a single floating point output number
- The longer (and more complex) the data, the more bits are needed in the output number
- Coding best achieved with 16 bit or 32 bit integer math
- No floating point math is required; incremental transmission scheme is used

Further Representation/ Coding Methods

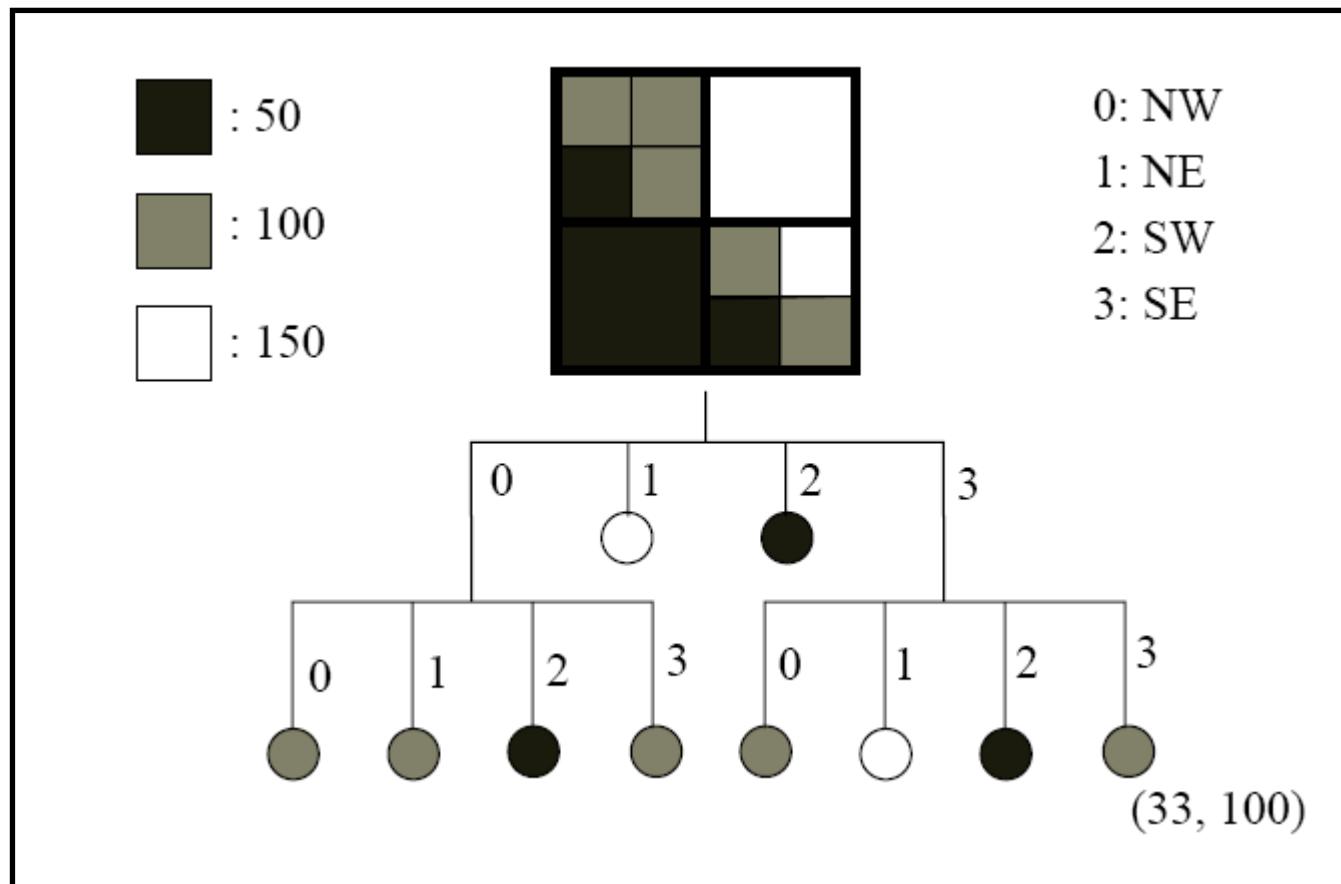
Run Length Coding

- Widely used technique for lossless data compression
- In greylevel images, a run is a sequence of consecutive pixels (say in a row) having constant intensity
- In binary images, a run is a string of consecutive 1s or 0s
- Each run is described by a pair of numbers showing the length and intensity of the run
- Example: 33332223333222333300111
RL codes: (4,3)(3,2)(4,3)(3,2)(4,3)(2,0)(3,1) or in symbols
 $S_0S_1S_0S_1S_0S_2S_3$
- Coding redundancy in RL codes can be removed by Huffman coding:
Fixed-length codes: 00 01 00 01 00 10 11
Huffman codes: 1 01 1 01 1 001 000

Quad Tree Representation

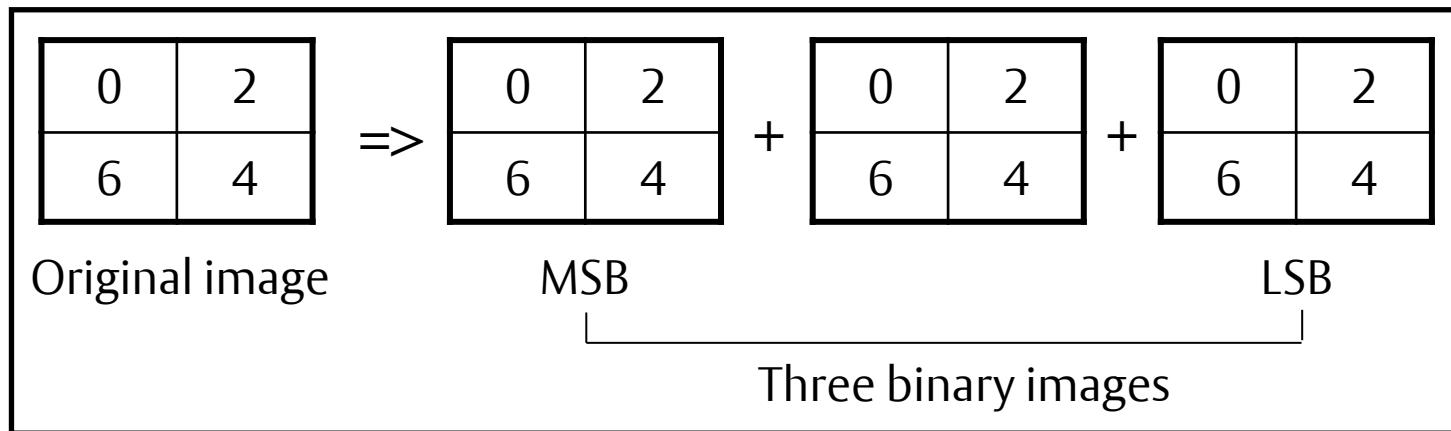
- A popular data structure for digital images obtained by recursively splitting image (quadrant) into 4 quadrants. This generates a tree (called quad-tree) where each node has 4 children nodes
- The division of a quadrant is terminated if all pixels in the quadrant have identical intensities
- Compression is achieved by representing each leaf node (i.e. a node with no children) using a pair (leaf code, intensity)

Quad Tree Representation



Bit Plane Coding

- An N-bit greyscale image may be regarded as a set of N binary images each of which corresponds to one bit (i.e. the MSB)



- Each binary image can be coded using RLC or quad-tree.

Predictive Coding

- The intensity of a pixel is predicted using some formula based on intensities of neighbouring pixels

0	A		
B	X		

$$X' = (A+B)/2$$

$$D_X = X - X'$$

- Difference (D_X) between predicted (X') and actual (X) intensity value is then coded using RLC and/or Huffman coding
- Many prediction methods exist that may be used to determine the predicted value X' . The above shows a simple example.

Lossy Predictive Coding

- Differences between predicted and actual intensities are quantized to pre-defined values
- Quantized differences are then coded using Huffman Coding
- The compression is lossy since quantization is involved

Lossy Quad Trees

- Identical to lossy quad tree representations but stop dividing a quadrant if the pixels in the quadrant have *similar* intensities
 - i.e. the intensity variance is less than a threshold
- The average intensity of the quadrant is then assigned to all pixels in the quadrant
- We may also terminate the quadrant division if the intensity surface of the quadrant can be well approximated by a pre-defined surface
 - e.g. a sloped plane; mathematically $Z=aX+bY+c$

Block Truncation Coding

- Divide image into small blocks and for each block compute its average intensity m_1
- For each pixel in the block, if its intensity is less than m_1 , set its intensity to A; otherwise to B
- The two greylevels A and B are chosen in such a way that the first and second moments of the intensity data in the block are preserved

Block Truncation Coding

a ₁ a ₂ a ₃ a ₄	B B A A	1 1 0 0
a ₅ a ₆ a ₇ a ₈	A B B B	0 1 1 1
a ₉ a ₁₀ a ₁₁ a ₁₂	A A A B	0 0 0 1
a ₁₃ a ₁₄ a ₁₅ a ₁₆	A A A B	0 0 0 1
Original image	Bilevel image	bit-plane
$m_1 = \frac{1}{N} \sum_{i=1}^N a_i$		$A = m_1 - \sqrt{\frac{q(m_2 - m_1^2)}{N-q}}$
$m_2 = \frac{1}{N} \sum_{i=1}^N a_i^2$		$B = m_1 + \sqrt{\frac{(N-q)(m_2 - m_1^2)}{q}}$
m ₁ : first moment; m ₂ : second moment; q: no. of B pixels N: no. of pixels in block		

- Each block is then described by the one-bit plane (which may be run-length coded) together with the 2 greylevels

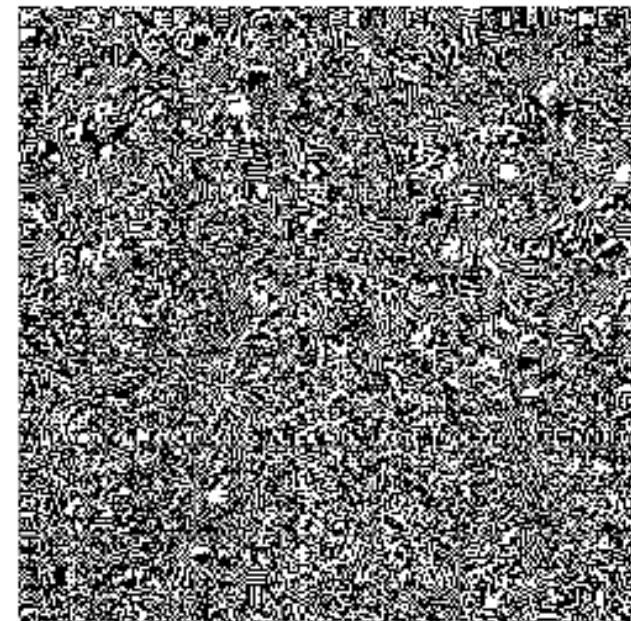
Review and DCT Example

- Most of the image transforms we discussed earlier are capable of packing information into a much smaller number of items – *transform coefficients*
- Transform coding achieves compression by only keeping the small number of significant information-carrying components
- Among all image transforms DCT is by far the most widely used transform in image compression (see last lecture)
- In practice it's implemented by
 - Dividing image into small blocks (e.g. 8x8)
 - Perform DCT on each block
 - Quantize DCT coefficients
 - Code quantized coefficients using RLC and/or Huffman coding
- DCT decompression is performed by taking the opposite of the above operations in the reverse order (no dequantization, of course)

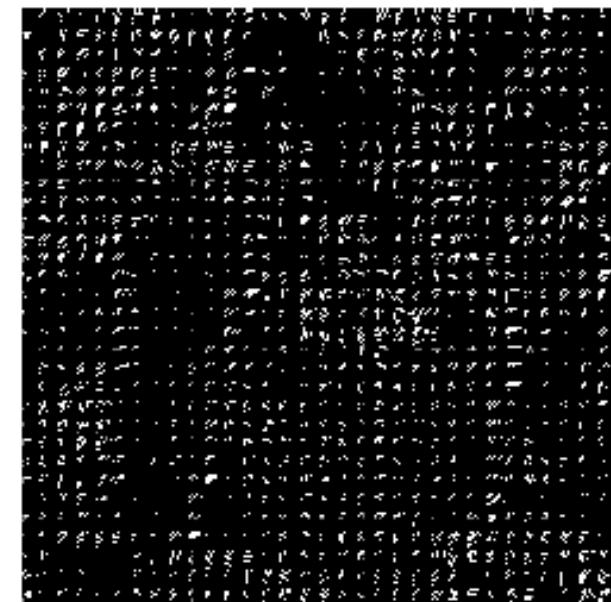
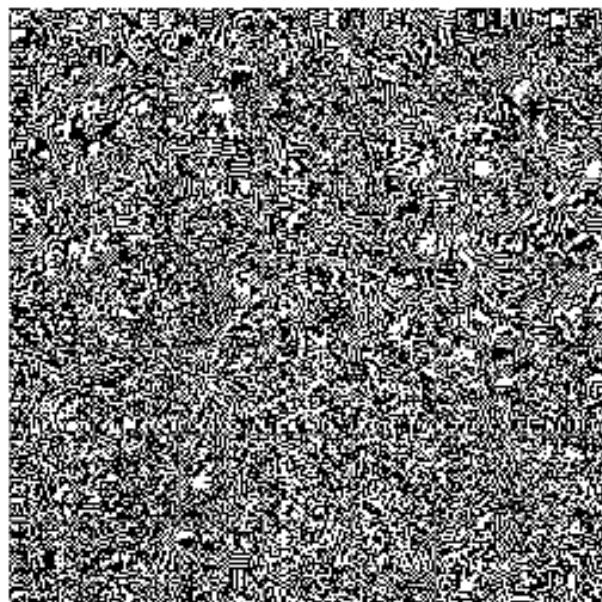
Pepper Example



Each 8x8 block
processed with
DCT



Pepper Example



Each element in each block of the image is quantized using a quantization matrix of quality level 50. Many of the elements become zeroed out, and the image takes up much less space to store

Pepper Example

The image can now be decompressed using the inverse DCT. At quality level 50 there is almost no visible loss in this image, but there is high compression. At lower quality levels, the quality reduces significantly, but the compression increases moderately



Original Image



Quality 50:
84% Zeros

Pepper Example



Quality 20:
91% Zeros

Quality 10:
94% Zeros

Further Example



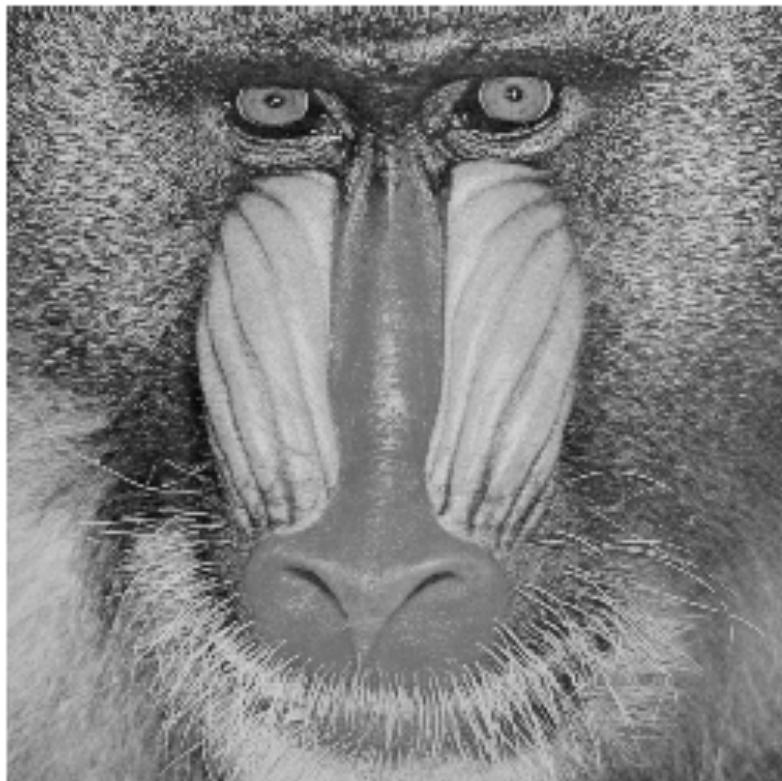
Original Image



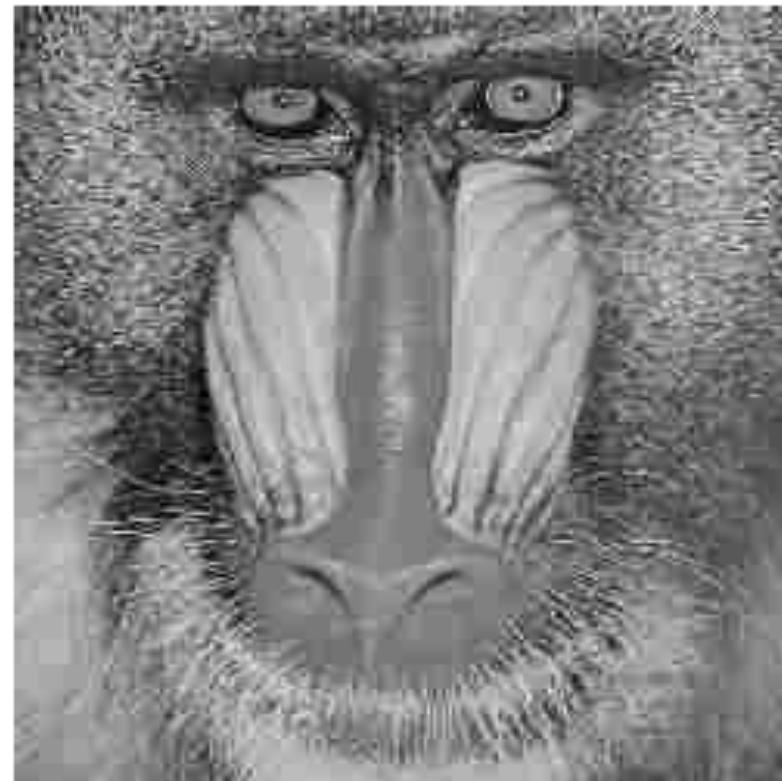
Quality 15:
90% Zeros

High contrast images, or images with a large proportion of high frequencies do not compress as well as smooth, low frequency images

Further Example



Original Image



Quality 15:
88% Zeros

Colour Image Compression

- The algorithms described can be easily extended to colour images
 - By processing each of the colour planes separately
 - Transforming image from RGB representation to other convenient representations, e.g. YUV, in which processing becomes easier

Other Coding Methods

- Arithmetic Coding
- Hybrid Coding
- Sub-band / Wavelet Coding
- Contour and Texture Coding
- Fractals
- Model-Based Coding
- Document/Binary Image Coding
- Image Sequence Coding
- Colour and Multispectral Coding
- JPEG, JBIG, MPEG
- ...

SE3IA11/SEMIP12

Image Analysis

Image Compression – Revisited

Lecturer:

Prof. James Ferryman

Computational Vision Group

Email: j.m.ferryman@reading.ac.uk

Introduction

- In the first lecture on image compression we examined the general process and methods utilised for image data reduction
- This lecture continues the theme to:
 - Compare image compression methods
 - Examine image sequence coding (video compression)

Comparison of Compression Methods

- *Transform-based* methods
 - better preserve subjective image quality
 - less sensitive to statistical image property changes
- *Prediction* methods
 - can achieve larger compression ratios in a much less expensive way
 - tend to be much faster than transform-based or vector quantization compression schemes
 - easily realised in hardware

Comparison of Compression Methods

- If compressed images are transmitted, an important property is *insensitivity* to transmission channel noise
- Transform-based techniques are significantly less sensitive to channel noise
 - if transform coefficient is corrupted during transmission, resulting image distortion is homogeneously spread through the image and is not too detrimental

Comparison of Compression Methods

- Erroneous transmission of a difference value in prediction compression causes not only an error in a particular pixel, it influences values in the neighbourhood
 - the predictor involved has a considerable visual effect in a reconstructed image
- Pyramid based schemes have a natural compression ability and show potential for further improvement of compression ratios
 - suitable for dynamic image compression and for progressive and smart transmission approaches

Introduction to Video Compression

- Motivation:
 - Raw video contains an immense amount of data
 - Communication and storage capabilities are limited
- Example HDTV video signal:
 - 720×1280 pixels/frame, progressive scanning at 60 frames/s

$$\left(\frac{720 \times 1280 \text{ pixels}}{\text{frame}} \right) \left(\frac{60 \text{ frames}}{\text{sec}} \right) \left(\frac{3 \text{ colours}}{\text{pixel}} \right) \left(\frac{8 \text{ bits}}{\text{colour}} \right) = 1.3Gb/s$$

- 20 Mb/s HDTV channel bandwidth
- Requires compression by factor of ~ 70

Introduction to Video Compression

- Video is a sequence of frames (images) that are *related*
- Related along the temporal dimension
 - therefore, temporal redundancy exists
- Main addition over image compression studied previously
 - temporal redundancy
 - video coder must exploit the *temporal redundancy*

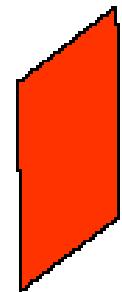
Temporal Processing

- Usually there exists a high frame rate
 - Significant temporal redundancy
- Possible representations along the temporal dimension include:
 - Transform methods
 - Good for constant velocity uniform global motion
 - Inefficient for non-uniform motion
 - real-world motion
 - Requires a large number of frame stores
 - leads to delay (+ memory cost)
 - Predictive methods
 - Good performance using only 2 frame stores
 - However, simple frame differencing is insufficient

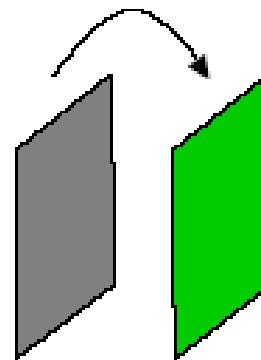
Video Compression

- Main advantage over image compression:
 - Exploit the temporal redundancy
- *Predict current frame* based on previously coded frames
- 3 types of coded frames:
 - I-frame: Intra-coded frame, coded independently of all other frames
 - P-frame: Predictively coded frame, coded based on previously coded frame
 - B-frame: Bi-directionally predicted frame, coded based on both previous and future coded frames

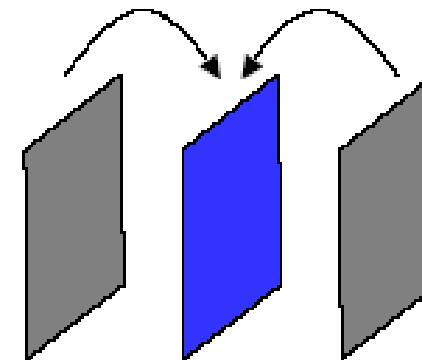
Coded Frames



I frame



P-frame



B-frame

Temporal Processing

Motion-Compensated Prediction

- Simple frame differencing *fails* when there is motion
- Motion must be taken into account
 - Motion-compensated (MC) prediction
- MC prediction generally provides significant improvements
- However:
 - how can we estimate motion?
 - how can we form MC-prediction?

Temporal Processing: Motion Estimation

- The ideal situation is to:
 - Partition the video into moving objects
 - Describe the object motion

In general, this is a very difficult task
- Practical approach: *block-matching motion estimation*
 - Partition each frame into blocks
 - Describe the motion of each block
 - Requires no object identification

In general, good robust performance

Block Matching Motion Estimation

- Assumptions:
 - Translational motion within block:
$$f(n_1, n_2, k_{\text{cur}}) = f(n_1 - m_{v1}, n_2 - m_{v2}, k_{\text{ref}})$$
 - All pixels within each block have the same motion
- ME Algorithm:
 - Divide current frame into non-overlapping $N_1 \times N_2$ blocks
 - For each block, find the best matching block in reference frame
- MC-Prediction Algorithm:
 - Use best matching blocks of reference frame as prediction of blocks in current frame

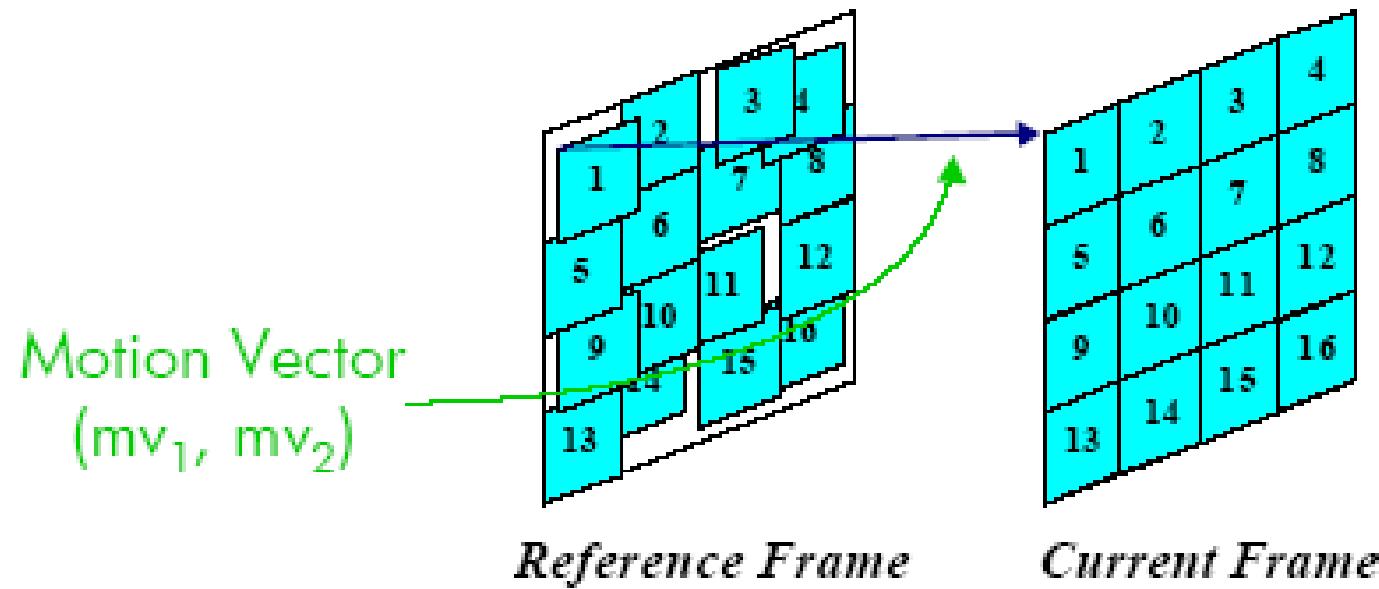
Block Matching Motion Estimation

- For each block in the current frame we need to search for best matching block in the reference frame
- A number of metrics exist for the “best match”, e.g.:

$$MSE = \sum_{(n_1, n_2) \in Block} \sum [f(n_1, n_2, k_{cur}) - f(n_1 - mv_1, n_2 - mv_2, k_{ref})]^2$$

- Candidate blocks include all blocks in e.g. $(\pm 32, \pm 32)$ pixel area
- Strategies for searching candidate blocks for best match include:
 - Full search: examine all candidate blocks
 - Partial (fast) search: examine a carefully selected subset
- Estimation of motion for best matching block: “motion vector”

Block Matching Motion Estimation



Motion Vectors and Motion Vector Field

- Motion vector
 - Expresses the relative horizontal and vertical offsets (mv_1 , mv_2), or motion, of a given block from one frame to another
 - Each block has its own motion vector
- Motion vector field
 - Collection of motion vectors for all the blocks in a frame

Block Matching: Summary

- The main issues to consider are:
 - Block size
 - Search range
 - Motion vector accuracy
- Motion typically estimated only from raw intensities
- Advantages:
 - Good, robust performance for compression
 - Resulting motion vector field is easy to represent (one MV per block) and useful for compression
 - Simple, periodic structure, easy hardware implementations
- Disadvantages:
 - Assumes translational motion model
 - Breaks down for more complex motion
 - Often produces blocking artifacts

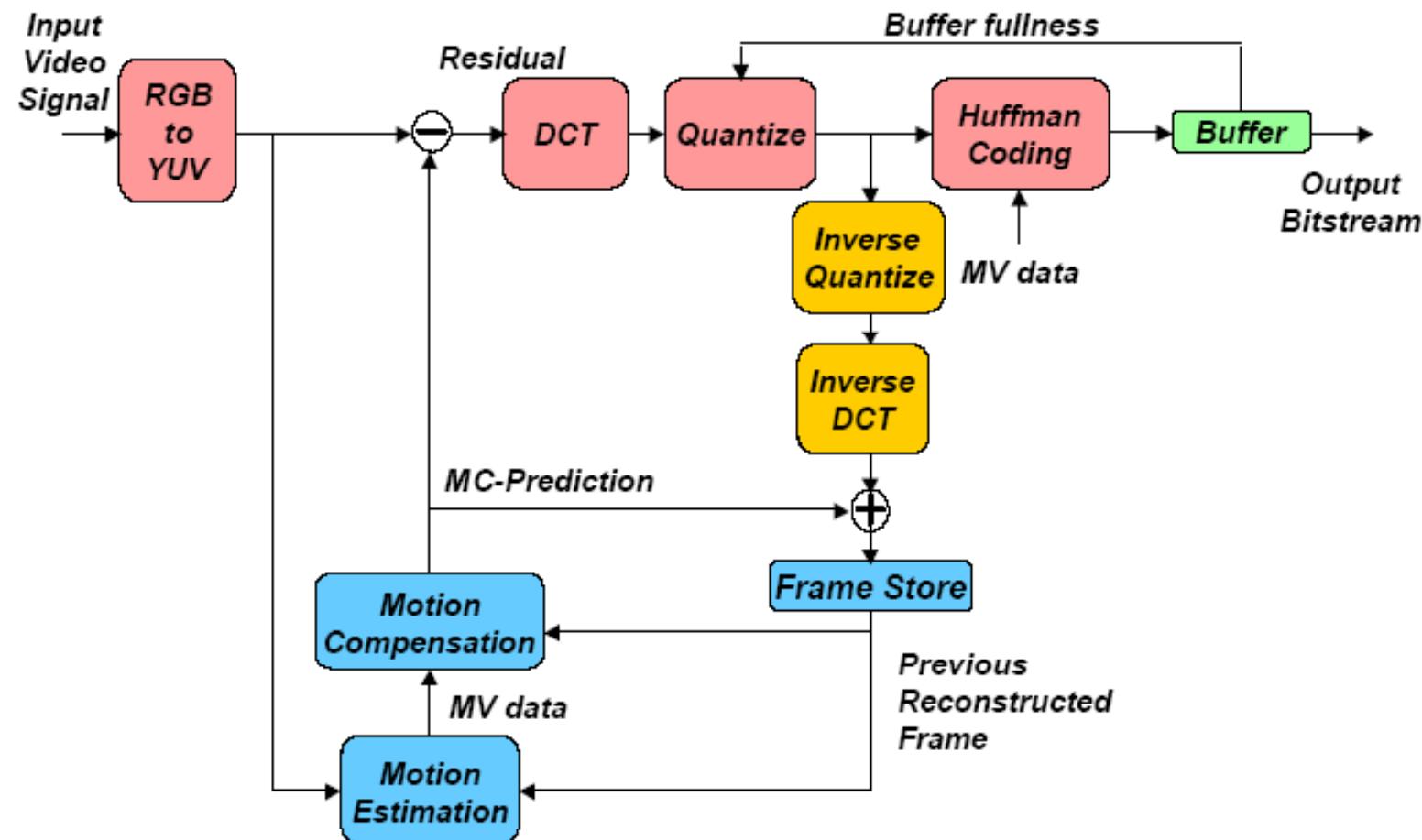
Summary of Temporal Processing

- Use MC-prediction (P and B frames) to reduce temporal redundancy
- MC-prediction usually performs well
- MC-prediction provides:
 - Motion vectors
 - MC-prediction error or residual
 - Code error with conventional image coder
- Occasionally MC-prediction may perform badly
 - E.g.: complex motion, new imagery (occlusions)
 - Approach:
 - Identify blocks where prediction fails
 - Code block *without* prediction

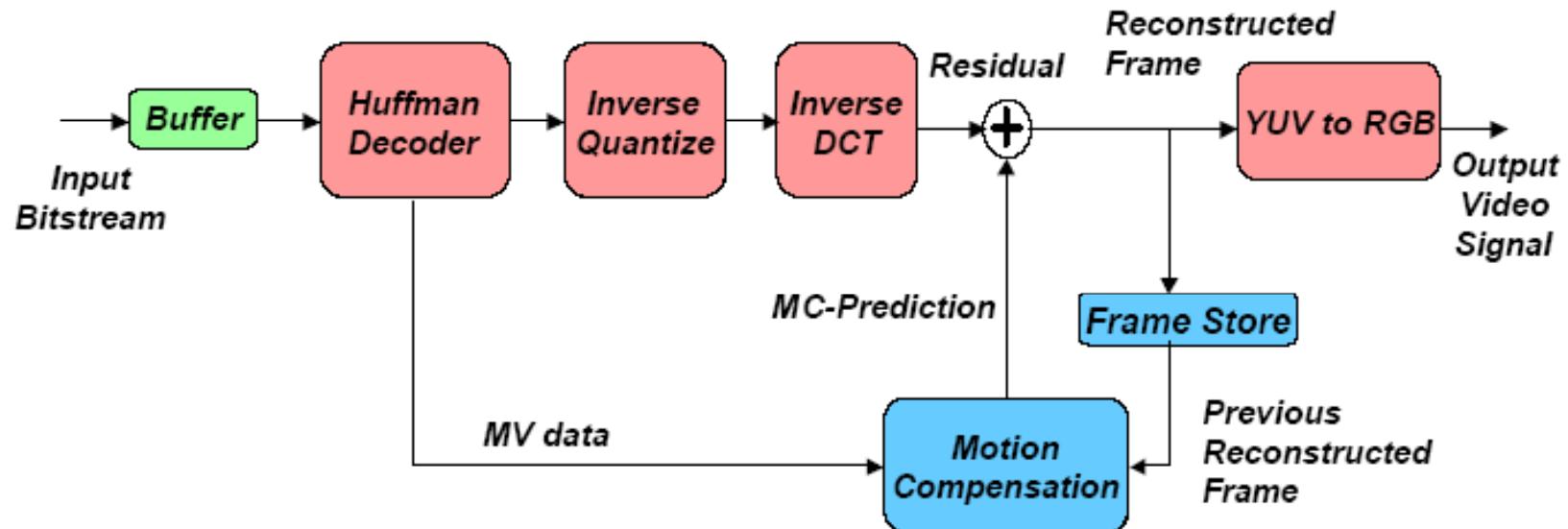
Video Compression Architecture

- Exploits the redundancies that we have learnt:
 - Temporal: MC-prediction (P and B frames)
 - Spatial: Block DCT
 - Colour: Colour space conversion
- Scalar quantization of DCT coefficients
- Zigzag scanning, run length & Huffman coding of the non-zero quantized DCT coefficients

Example Video Encoder



Example Video Decoder



Acknowledgements

- HP – Image and Video Coding

Summary

- We have examined some additional aspects of image compression
- The next lecture today covers symbolic feature extraction

SE3IA11/SEMIP12

Image Analysis



Symbolic Feature Extraction

Lecturer:
Prof. James Ferryman
Computational Vision Group
Email: j.m.ferryman@reading.ac.uk

Introduction

- Many vision applications require symbolic image features (such as lines, circles, etc.) rather than the raw intensity values (iconic image data)
- This lecture asks - how can we convert an iconic description to a symbolic one?
- The symbolic description is required before we can “reason” about the image – visual intelligence – to be covered next term

Describing the Image

- Grouping pixels according to *similarity*, or finding a boundary where there is an abrupt *difference*, are obviously related!
- In both cases, we are collecting together previously independent points in the image to make a “whole” which can be studied in its own right

Describing the Image

- There are many reasons for trying to do this:
 - Practical – It reduces the memory required to describe the image
 - e.g. [line [0 10] [100 110]]
uses far less computer memory than does (say):
[[0 10 [1 11] [100 110]]
Likewise: [circle a b r] (etc, etc.)

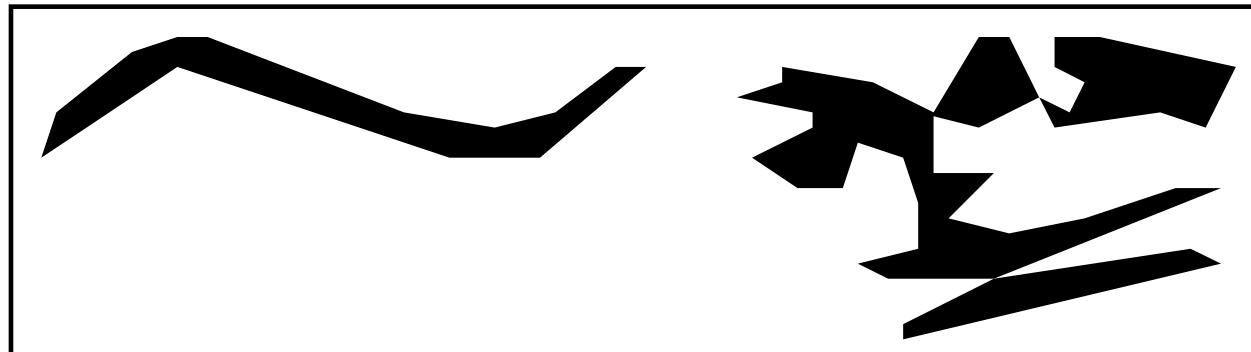
This is possible because the *higher level descriptions* use symbolic concepts which capture some of the redundancy of the *lower level descriptions*

Describing the Image

- There are many reasons for trying to do this:
 - Theoretical – High level descriptions allow new knowledge to be mobilised, which cannot be applied at lower levels
- *To use high-level concepts, we must identify and label the image fragments as symbolic entities, and make explicit their properties*

Describing Regions

- A region is a connected set of pixels – how can it be described?

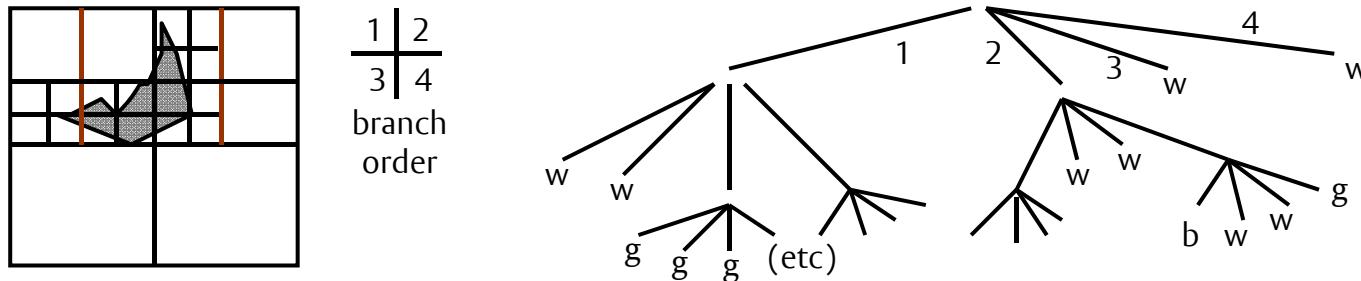


The abstract problem of analysing and describing continuous 2D patterns is very deep!

- Our task is somewhat simpler, because in discrete distributions there can be no infinite whorls, or other singularities (the subject of much discussion in mathematical analysis)

Describing Regions

- Unstructured representations:
 - Simple literal maps – e.g. 2D boolean matrix
Fastest, dumbest, highly inefficient on memory
 - Recursive subdivisions, e.g. quadtrees



Rule: recursively explode any grey (g) square, until all squares are white (w) or black (b) (up to some minimum element size)

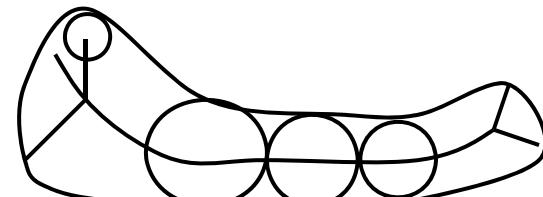
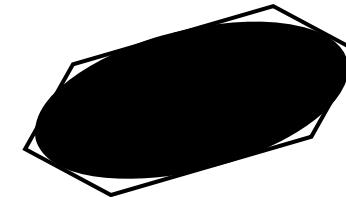
- Efficient on memory (especially for compact regions)
 - Very fast for many operations (esp. union & intersection)
- Extensively used in *constructive* applications (e.g. graphics)

However: non-topological (i.e. adjacency is not preserved), so some “simple” operations are difficult (e.g. nearest points in two regions)

Describing Regions

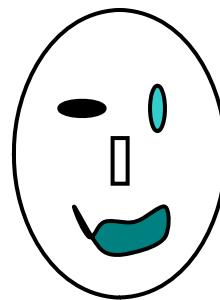
- Structured representations:

- Polyline approximations of the boundary: $[x_0 y_0] \dots [x_n y_n]$
- Ribbons: An axis, and a generator function. E.g. The median axis: the loci of centres of circles which touch the boundary at two places
- Splines: A set of control points with simple analytic curves (e.g. cubic) interpolated locally
- Fourier descriptions (of a 2D image or a 1D curve)



Describing Regions

- Note how the different representations *make explicit* different properties of the regions (which may or, equally, may not be useful)
- However, some patterns have special significance – even if bizarrely distorted!



We also need to capture symmetries, compactness, and relationships between the parts

Classifying Regions

- However, none of these representations *classify* the segmented region, i.e. associate it with a higher order class, which can be reasoned about
- Some classes have a convenient *analytic expression*, straight lines, circles, conics, polynomial curves, etc. etc. From these can be made triangles, squares, simple curved shapes
- *Classification* depends on making a comparison between properties of the data and stored definitions of classes
- Once the class membership is accepted, then the battery of concepts and methods associated with the high-level class become available to subsequent analysis

Classifying Regions

- Measuring properties of regions:
- A wide range of methods are available:
 - Statistical
 - Moments – 1st order (= mean), 2nd order (= inertia), ...
 - Topological
 - Number of connected regions
 - Number of holes (c.f. the Euler number)
 - Geometrical
 - Length of boundaries, types of concavity
 - Aspect ratio, compactness ...

2D regions turn out to be very difficult to describe adequately - more attention has been directed towards describing their boundary lines, in order to discover consistencies, symmetries, etc.

The Canny Operator

- Various operators (Roberts, Sobel, Prewitt) were described in previous lectures
 - for locating pixels where there is significant discontinuity in intensity (edge detection)
- Such pixels are often called *edgels* (for edge elements)
- A far more popular operator for edge detection is due to John Canny (hence the name) who reported his result in 1984 as part of his PhD work

The Canny Operator

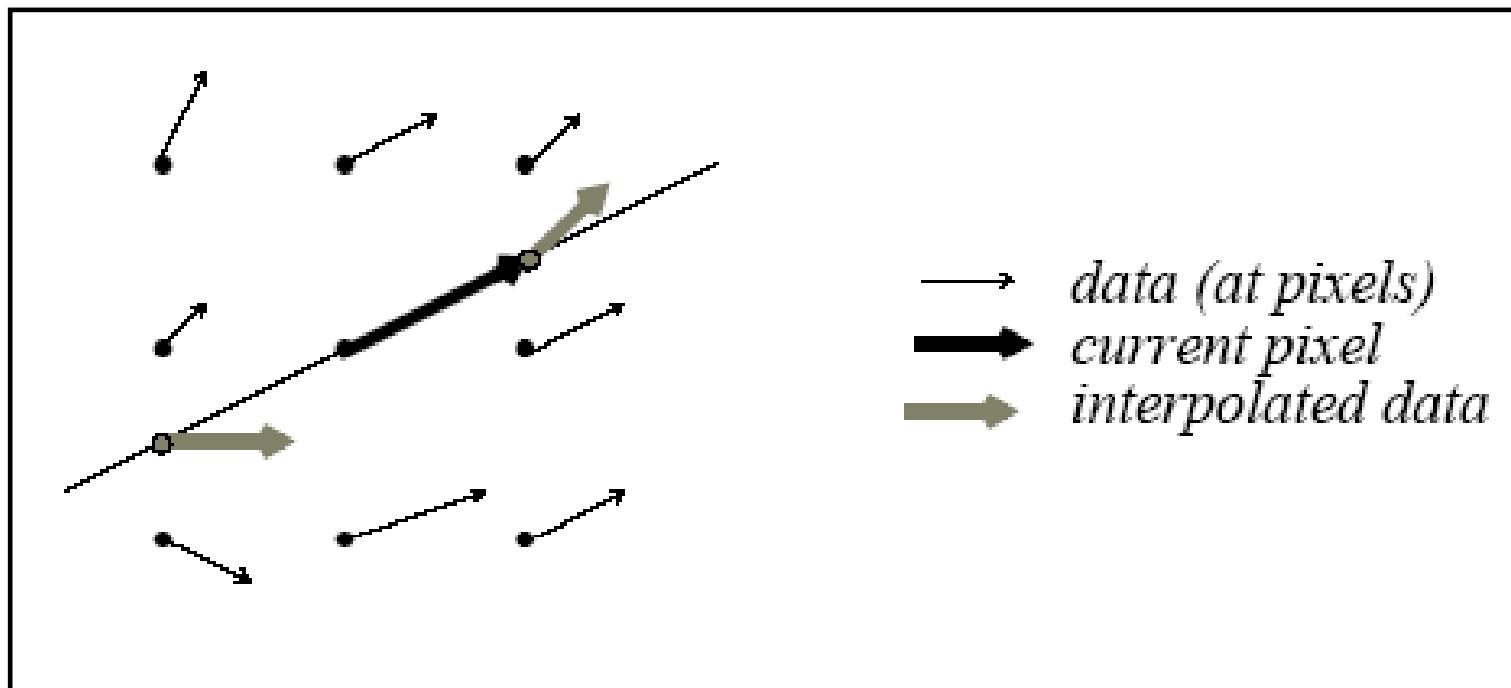
- The Canny Operator has become the “standard” method in advanced vision systems
- It overcomes some of the disadvantages of simpler operators but takes far more computing effort!
- The Canny Operator attempts to find the points in the image where the derivative of the greylevel has a local maximum
 - i.e. the points of inflection of the greylevel surface

Major Steps in Canny Operator

- (1) Estimate the gradient vector at each pixel by taking the simple partial derivatives defined by the weights $(-1 \ 0 \ 1)$. This provides the gradient direction and magnitude

- (2) At each pixel, examine the magnitudes of the gradient at adjacent points along the direction of the gradient at the current pixel. Since the gradient direction might not point directly towards a pixel, this requires the values to be interpolated between two nearest pixels

Interpolation of Image Data



Major Steps in Canny Operator

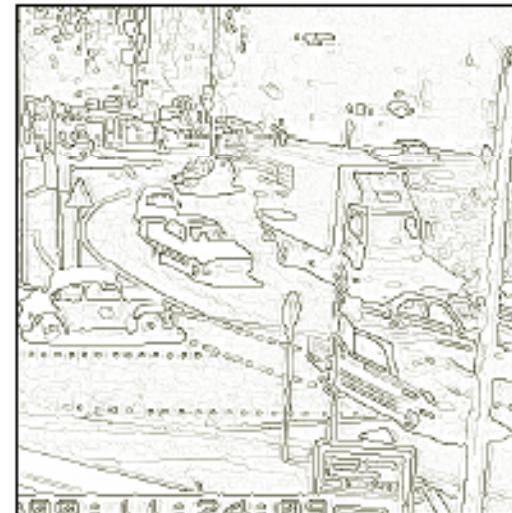
- (3) *Non-maximum suppression*: only record those pixels whose magnitudes are *greater* than their interpolated neighbours
 - also require them to exceed some minimal threshold to avoid pure “noise”

This picks out the peaks along *lines of steepest ascent* without suppressing points because they are smaller than their neighbours *along the contours*

Example of Canny Edge Detection



Intensity image



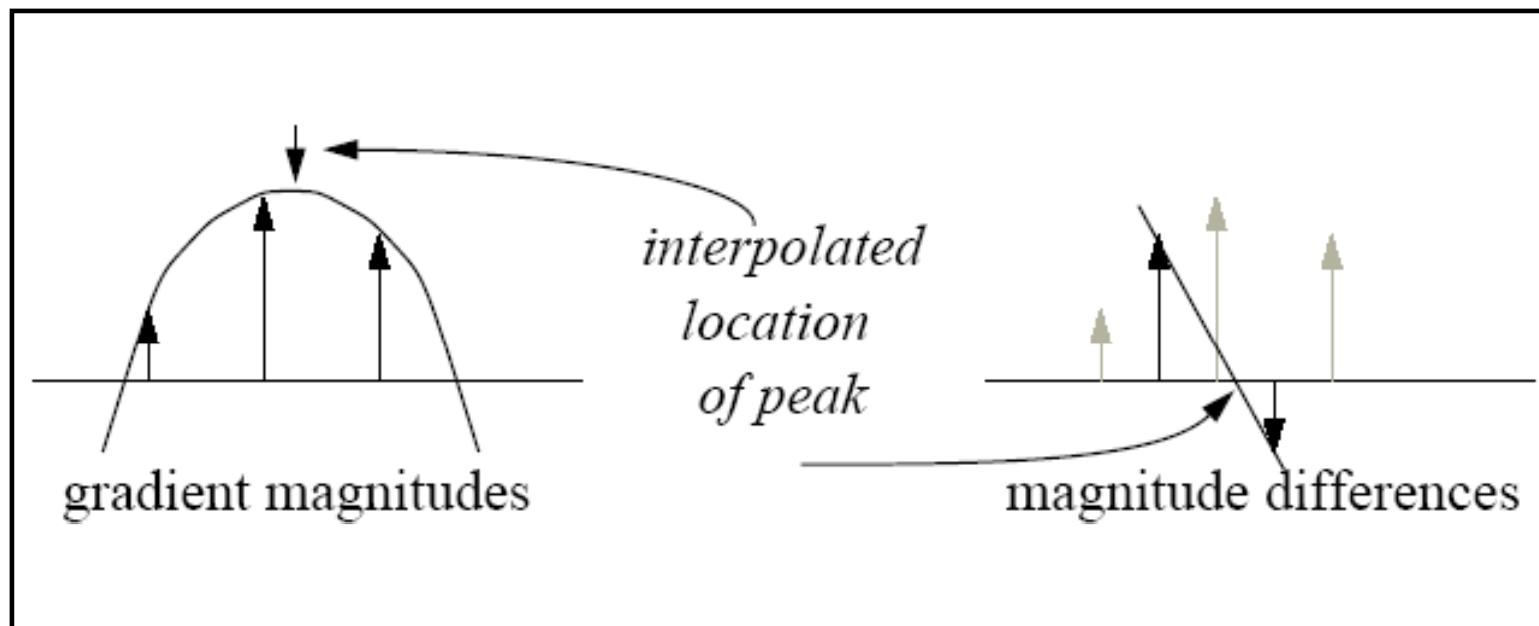
Grad mag. of Canny edgels

Major Steps in Canny Operator

- (4) The position of the edge can now be obtained to sub-pixel accuracy, by looking at the local derivative of the gradient
 - i.e. the differences between the gradient of the current pixel and its two interpolated neighbours, and finding the zero point between them
 - necessarily, one neighbour is +ve and the other is -ve

This is equivalent to fitting a parabola (a quadratic) to the gradient magnitude data and finding its peak

Edge Localisation



Major Steps in Canny Operator

- (5) We are then left with a sparse map of “edgels”, which (with luck) form long streams of profiles, running along the hillsides in the grey-surface, at the points where the hill is steepest
- Each edgel is tagged by its exact (sub-pixel) position, and the magnitude and direction of the gradient vector

Major Steps in Canny Operator

- (6) Except in very unusual image conditions, each edgel has only two neighbouring edgelets (or 1 if it is at an end)
- A connect process is then run, to tie together the profiles into sequences (or “strings”)
- This process uses *hysteresis* to help pay more attention to edgelets which form large structures, without being distracted by small weak strings
 - i.e. find an edgel which exceeds a fairly high threshold, then propagate along its neighbours provided they exceed a lower threshold

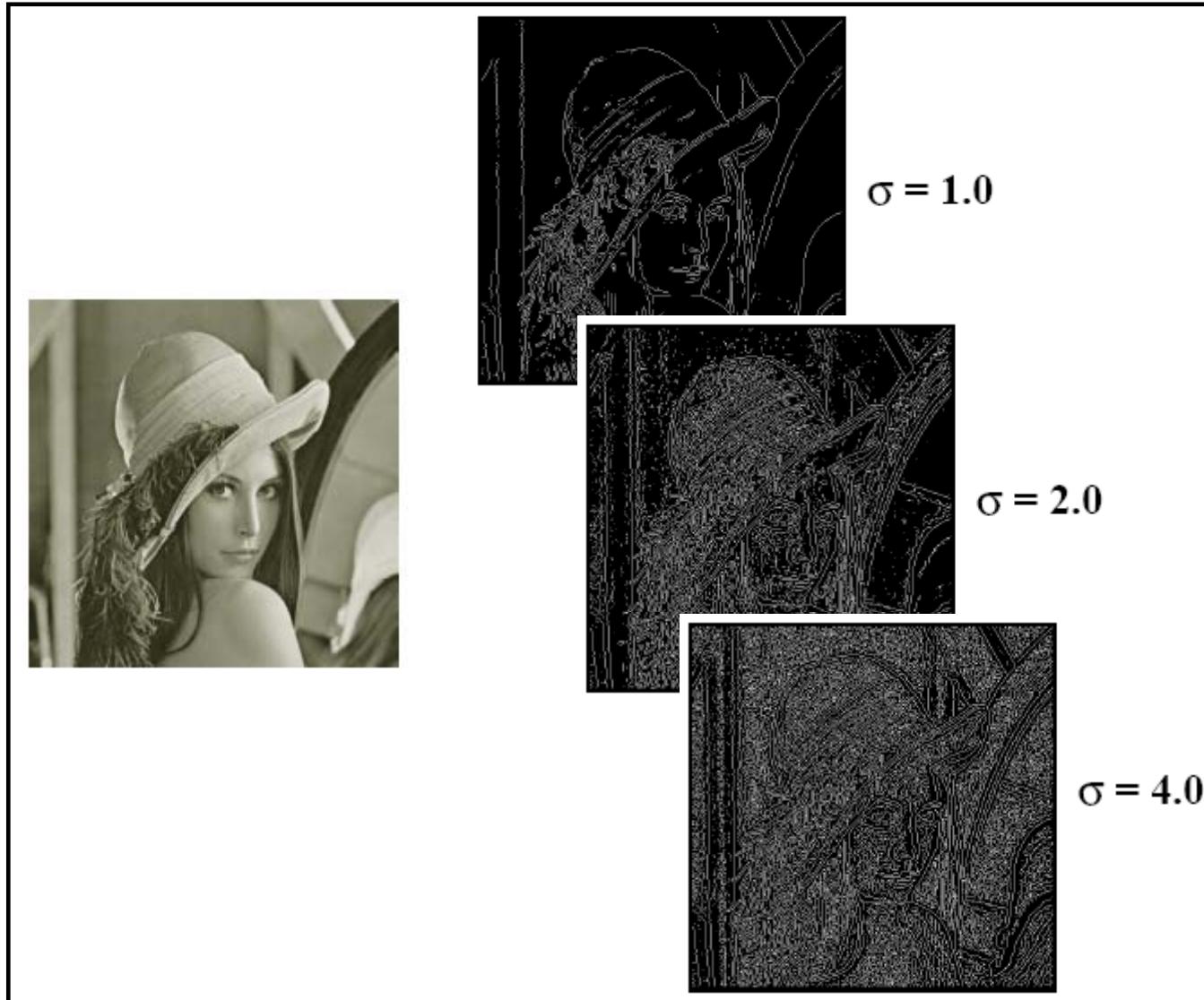
Multiple Scales

- Like all differentiation processes, the Canny operator is intrinsically sensitive to noise
- To maintain this under control, the image is first of all smoothed by means of a Gaussian blur function
 - whose width is controlled by parameter σ
- As we have seen in previous lectures, smoothing is essentially a process of integration, which is the “opposite” of differentiation
- By trading off the two we can select particular scales of edges
 - e.g. if we look at a bush, we might select the overall outline (low frequency, large scale), or pick out individual leaves (high frequency, fine scale)
- By varying the width of the Gaussian, it is possible to bias the Canny operator to pick out a particular *scale of detail*

Multiple Scales

- Thus it is possible to run the whole process several times at different scales to identify fine edges and coarse edges separately (see example next slide), but (of course) it makes the process far more computationally expensive
- Q: in practice how do you determine which scale is most relevant to any particular task?
- A: use of multiple scales

Canny Edgels at Multiple Scales



Describing Lines

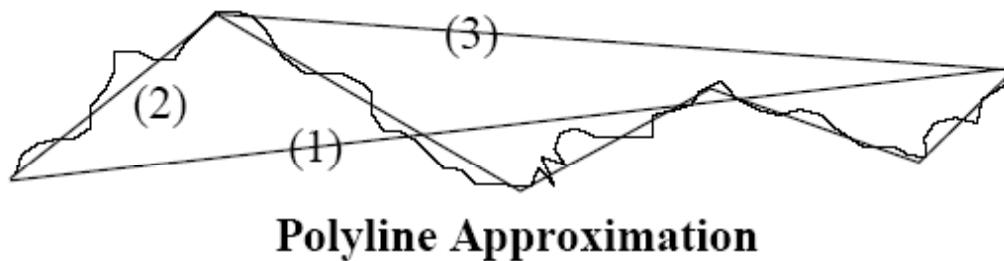
- As we have seen, early image processing (i.e. the initial processing of the image data) often yields a sequence of edgels, which form a connected line
- All that is made explicit is the geometrical positions of the edgels or strings and their ordinal positions
- There is a wide variety of methods for deriving descriptions of a line in terms of higher order entities (straight lines, arcs of circles, ...)

Line Extraction

- The next phase after the Canny operator is typically to try to build descriptions of the “strings” of Canny edgels in terms of high-level symbolic entities
 - for example, straight lines, and circular arcs
- A common approach is use a polyline approximation
 - i.e. each edgel string is described by a set of straight line segments
- The derivation of a good polyline description is surprisingly difficult, mainly since the approximation can be made at many different scales of detail
 - it is very hard to specify an appropriate scale for any particular purpose

Line Extraction

- One common approach is to specify a “noise” criterion (c) and use a *split-and-merge* technique:



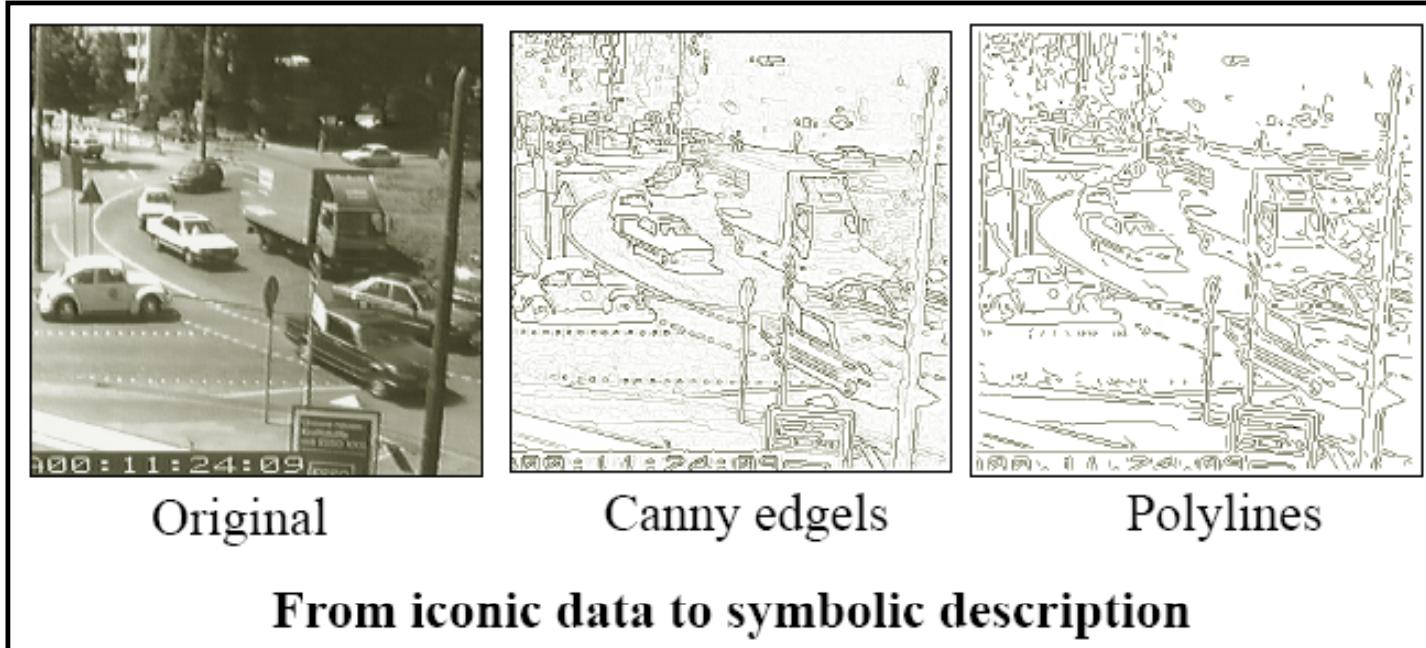
- *Fit a straight line between the two ends (1)*
- *If the mean squared error of the points from the line $>c$ then*
- *Split the input data at the point of maximum deviation, making sub-lines (2) & (3)*
- *Iterate for the two sub-lines (2) and (3) etc.*

Line Extraction

- Other error terms are possible which may be computationally cheaper
 - E.g.:
 - using the maximum deviation as the criterion of acceptability of a line
 - using the curvature maximum as the splitting point

Line Extraction

- The following is an example of polyline approximation:



Line Extraction

- Notice how poor the results are (it is much harder for a human to interpret the polyline image, than the original image – think WHY it's so!)
- However, the big achievement is that we have reduced the *mass of low-quality data* in the image (any individual pixel of which could have come from any possible object), to a far *smaller set of higher-quality data* in a symbolic description
 - which we hope will help discriminate between objects (the subject of exploration next term)
- In this case, we only have a few hundred entities, stored in a totally symbolic database consisting of theoretical lines and arcs, etc., which will be used to reason about the identity of the objects
 - we will also be exploring this topic next term too

Intrinsic Representation of a Curve

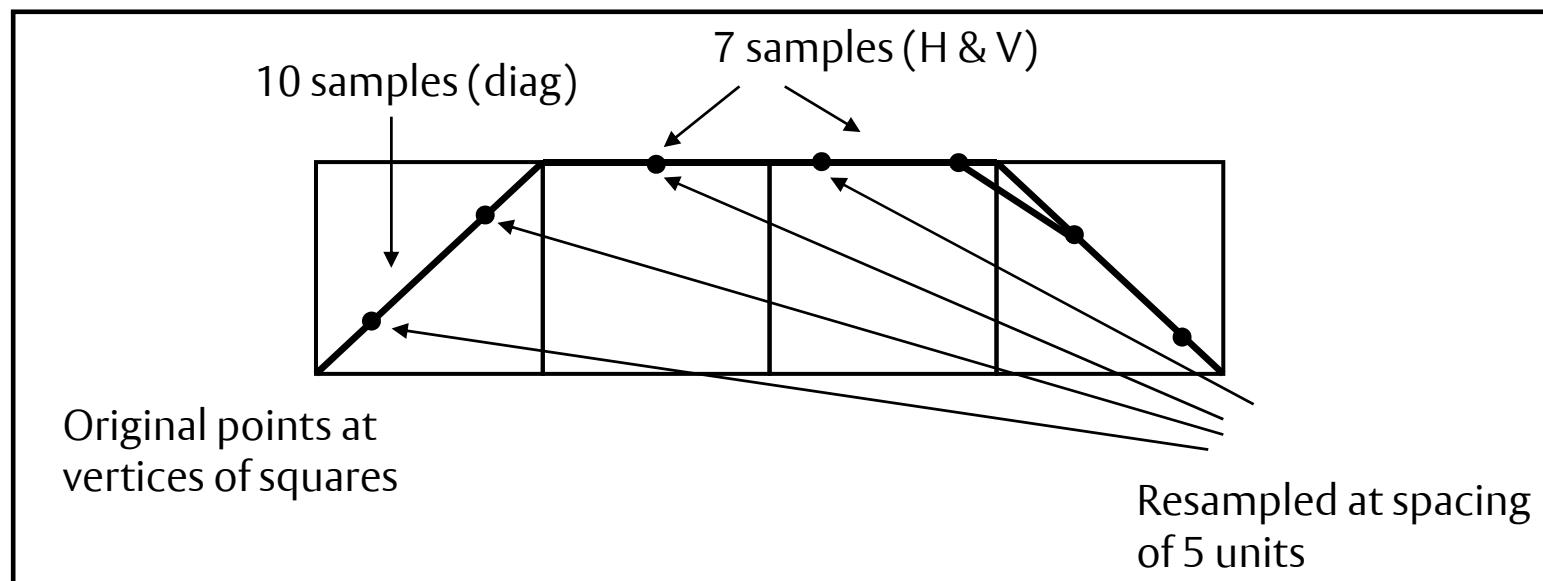
- A cartesian representation of a curve (e.g. $\{(x_i, y_i) : (i=1 \dots n)\}$) is usually inconvenient: it fails to make explicit important concepts like distance along the curve (e.g. the length of the curve)
- The use of an *intrinsic parameter* is common: i.e. the curve is parameterised by s , the distance along the curve from an origin
- The curve is now represented as $\{ (x(s), y(s)) : a \leq s \leq b \}$
- Note: we still need the curve to be sampled at discrete points, which we usually want to be equally spaced
- The new samples *will not correspond to the original (x, y) points*

Intrinsic Representation of a Curve

- One way of overcoming this difficulty is to *resample* the line at more nearly equal intervals
 - this is relatively straightforward, for a connected curve (with square resampling) since we know that successive points must be at angles of multiples of $\pi/4$.
- The distances between adjacent points are therefore 1 or $\sqrt{2}$, which are (approximately) equivalent to the ratio 7:10

Intrinsic Representation of a Curve

- So to produce a regular resampling: subsample the line fragments between pixel positions by 7 (H or V) or 10 (diagonals), then *resample* at a constant spacing (e.g. 5) to give the s parameter

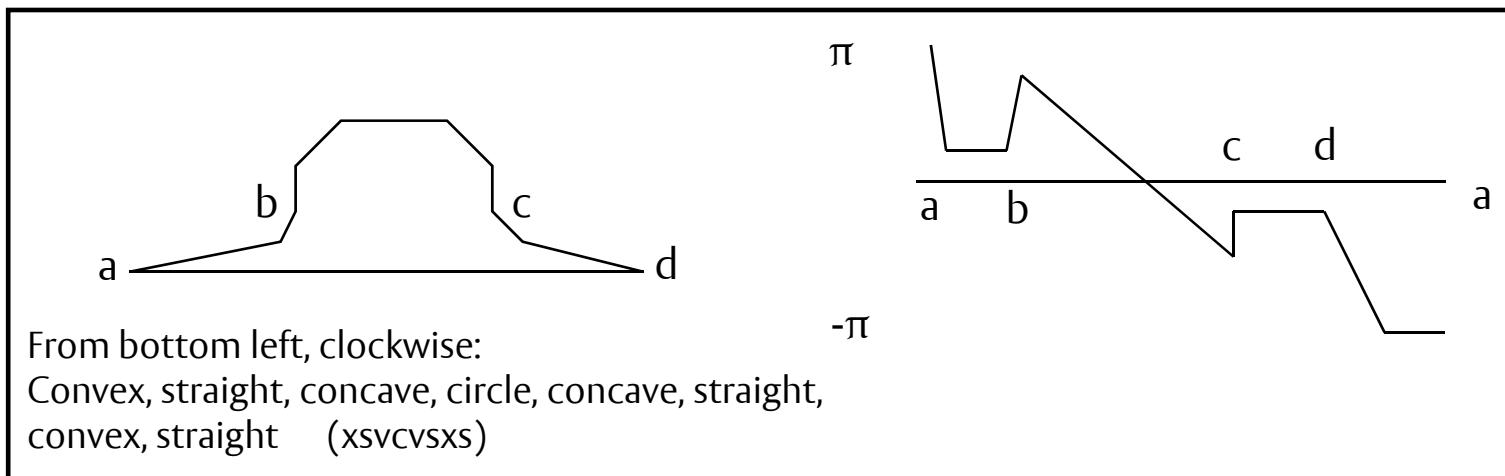


Curvature

- The use of the intrinsic equation with regular sampling, allows concepts such as *curvature* to be introduced:
 - the derivative of the $x(s)$ and $y(s)$, w.r.t. s define the tangent to the curve at s : $\psi(s) = \arctan(\frac{dy}{ds} / \frac{dx}{ds})$
[tends to: $\arctan(\frac{dy}{dx})$ for small ds]
- This representation is sometimes called the ψ - s function
- Note that the $d\psi/ds$ gives a measure of the *curvature*
- This is easy to approximate in the discrete sample, and provides an alternative means of segmenting a general curve into higher order components

Curvature

- E.g.: if the curvature is (locally):
 - 0, then the curve is a straight line
 - constant (non-zero), then the curve is circular
 - an extreme value, then there is a corner

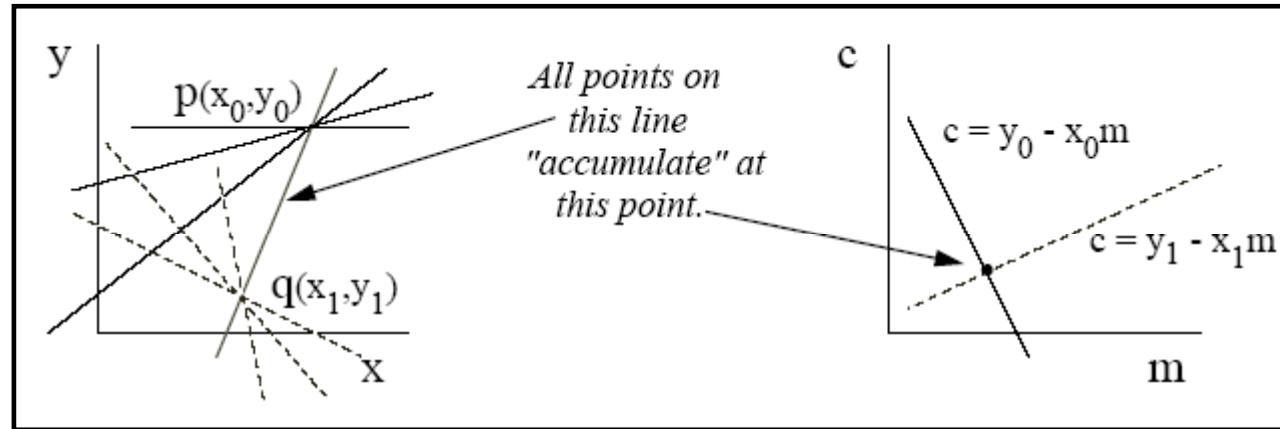


- A high level description of the curve has now been derived, as parts and their relationships along the curve (sometimes called a syntactic description)
- Allowing for cyclic permutations, it is independent of the orientation and position of the curve, but it is very dependent on the definitions of the components, and cannot distinguish distortions of size

Hough Transform

- An alternative and powerful way to detect a known symbolic entity or pattern (e.g. a straight line, circle, etc.) works by searching the *parameter space* of the pattern
 - the idea was first proposed by P. Hough which is not called the Hough Transform
- The simplest (and most widely used) example looks for straight lines in an image
 - suppose we have applied an “edge-finder” to derive an edge map
 - how can we find good, elongated lines?

Hough Transform



- Consider an edgel at p at (x_0, y_0) – it could arise from any line whose parameters m and c satisfy $y_0 = m_{x_0} + c$.
- However, this equation can be taken as a line in (m, c) space – which defines all the (m, c) pairs of all possible lines in (x, y) constrained to pass through p
- This is true for any edgel
 - e.g. that at $q(x_1, y_1)$, defining the line $c = y_1 - x_1 m$.

Hough Transform

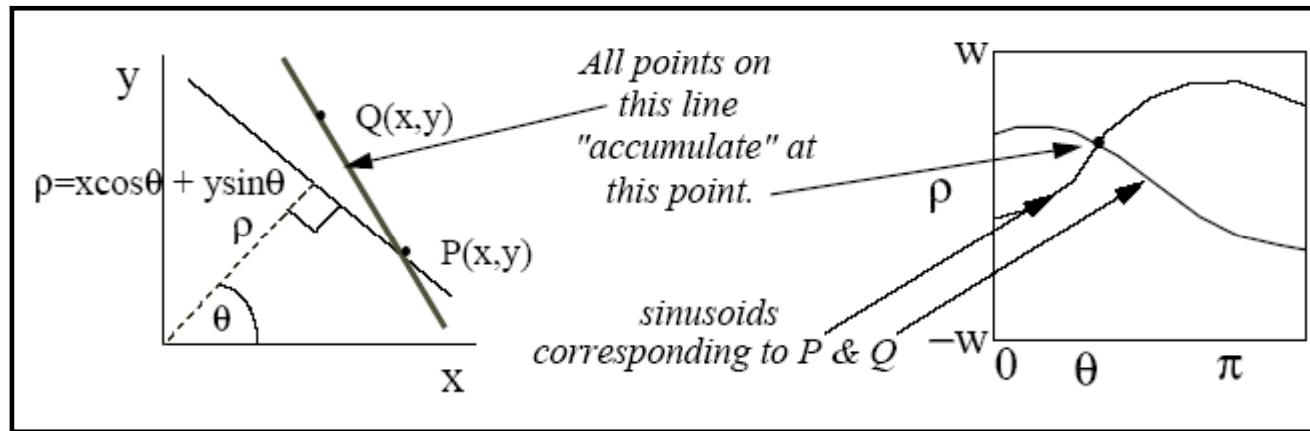
- Note the “dualism”: the *line* which is common to p & q , maps into the *point* which is common to the two lines in (m,c) space
- So, to discover the straight lines in an image, we draw all the lines in the “parameter” space (m,c) corresponding to all the edgels discovered in the image, and look for multiple intersections
- This is done by representing (m,c) as a discretely-sampled “accumulator array” (initially zeroed), which is incremented by one everywhere a line passes through it – then look for peaks in the “votes”

Alternative Parameterisation

- In practice (m, c) space is very inconvenient
 - both are unbounded, i.e. to cope with all cases, both m & c must take values between $\pm\infty$
- This makes it very awkward to represent the (m, c) space in a finite array
- An alternative parameterisation of the straight line is commonly used

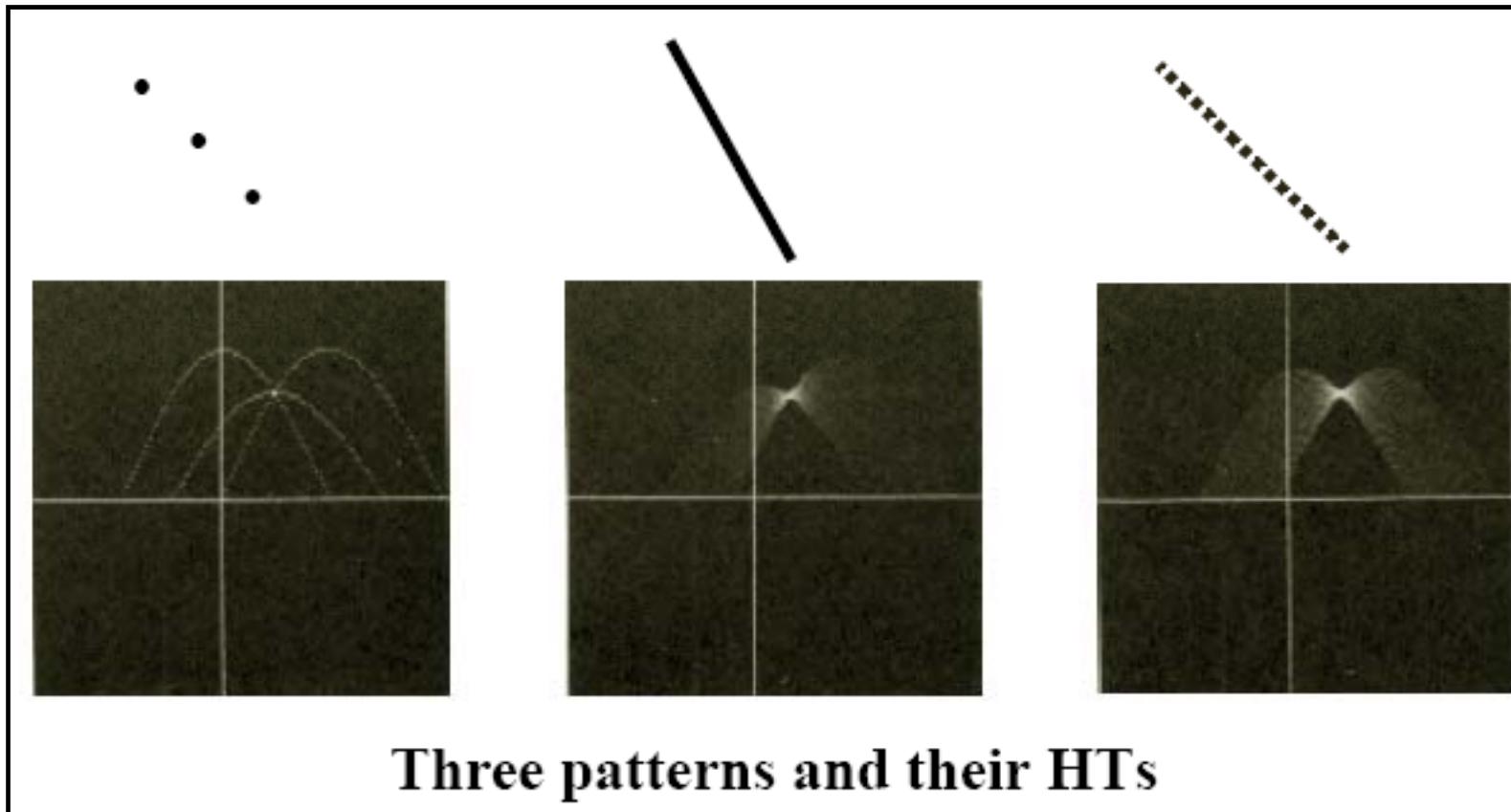
$$\rho = x \cos \theta + y \sin \theta$$

Alternative Parameterisation



- The set of all lines passing through a single edgel in (x,y) space now creates a sinusoid in (p,θ) space; however we now need only be concerned with values of (p,θ) in the bounded region:
 $0 \leq \theta \leq \pi$ (we only need $\frac{1}{2}$ a sinusoid, since (p, θ) is $(-p, -\theta)$)
 $-w \leq p \leq w$ (where w is the diagonal diameter of the image)
- This is far easier to store as an array (though the accumulation process is somewhat harder – drawing a sine wave vs. a line)

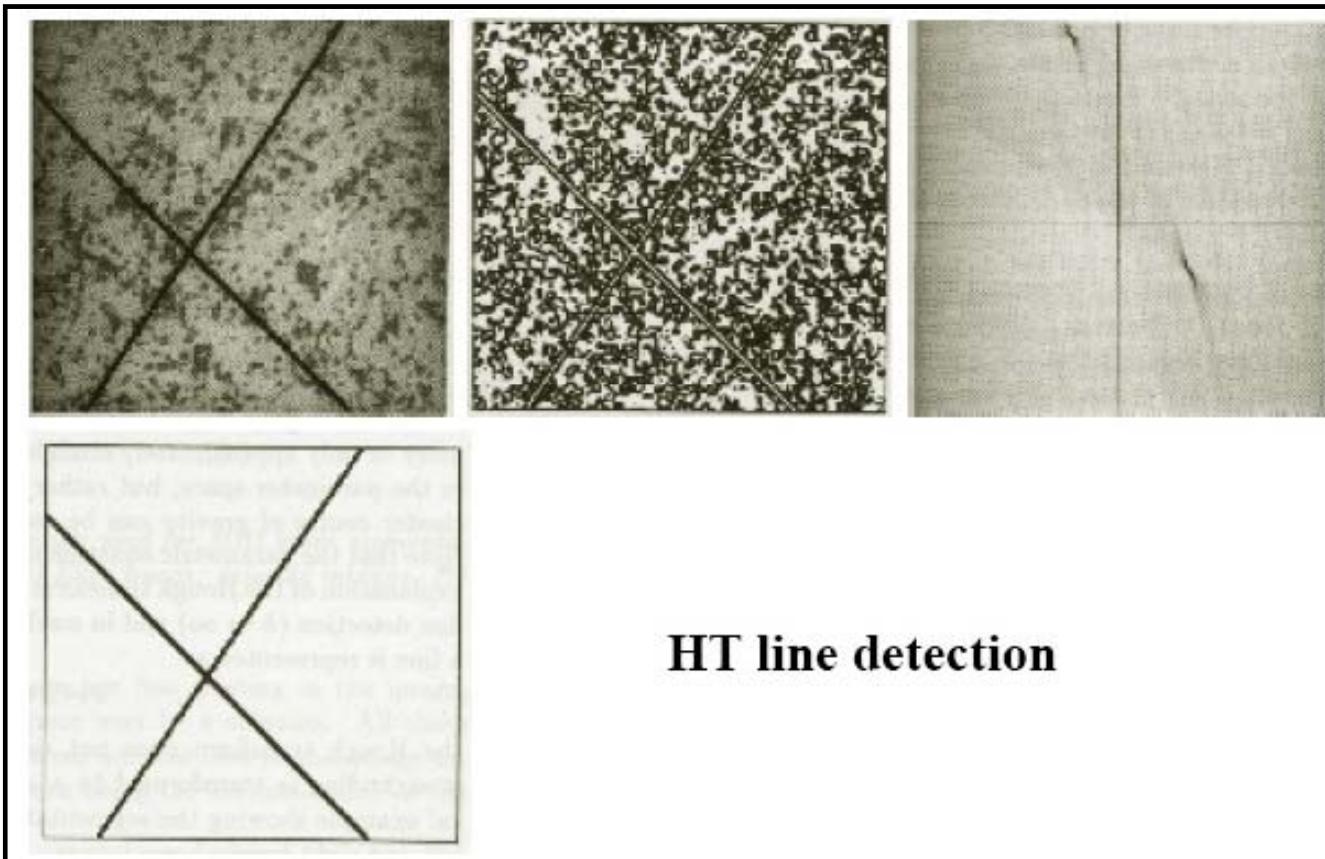
Example of Hough Transforms



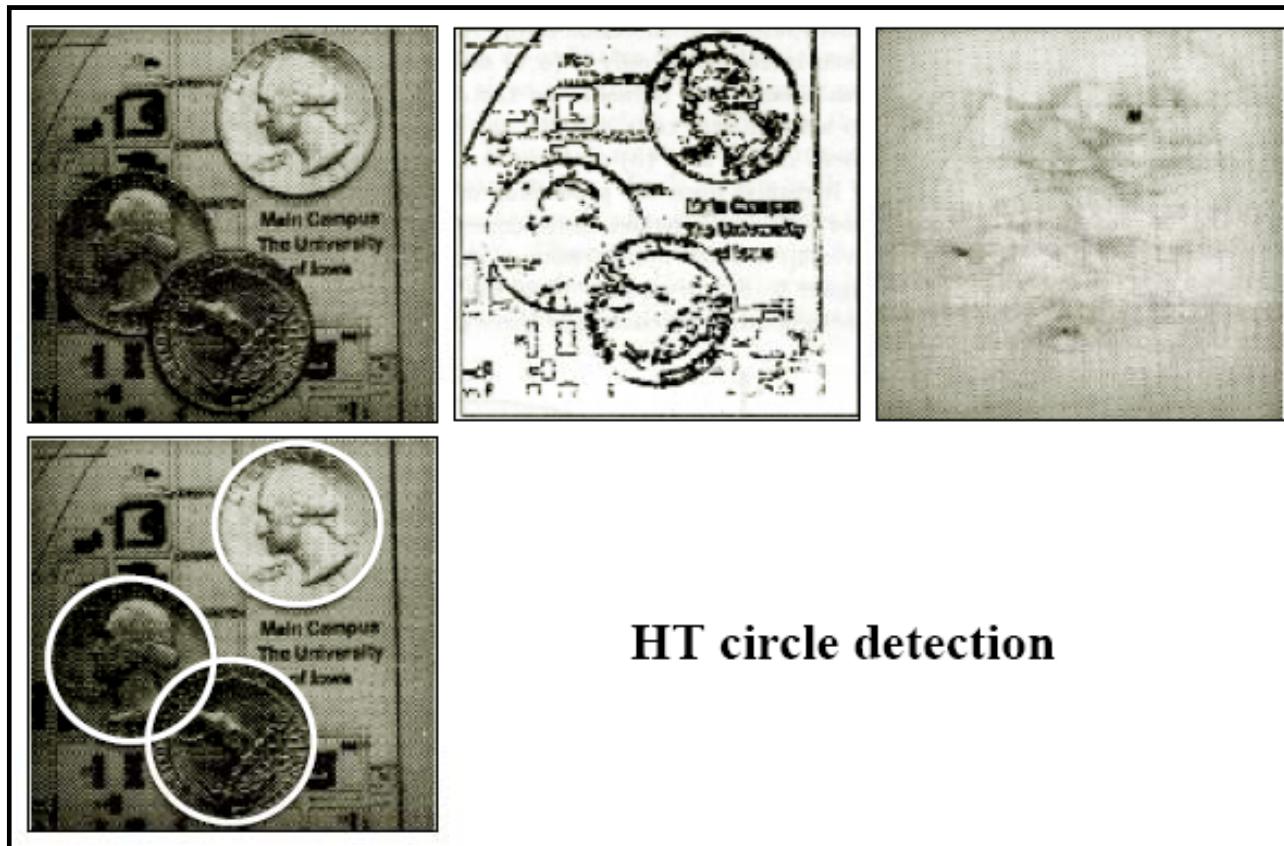
Generalised Hough Transform

- The Hough Transform can easily be generalised to any pattern which can be specified by an analytical equation $f(x,a)=0$, where x is an image point and a is the parameter vector
- The general procedure is as follows:
 - Initialise accumulator array $A(a)$ to zero
 - For each edgel x , compute all a such that $f(x,a)=0$ and increase $A(a)$ by 1
 - Local maxima in A correspond to instances of the pattern in the image
- The Hough transform may be further extended to arbitrary shapes specified by a sequence of boundary points

Examples of Hough Transform



Examples of Hough Transform



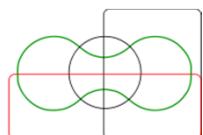
Summary

- Overview provided of symbolic feature extraction
- The lecture next week covers morphological image processing

SE3IA11/SEMIP12

Image Analysis

Morphological Image Processing



Chapter 9 in the Gonzalez & Woods book

Autumn-term 2015

SF31A11/SEM1P12 Image Analysis

1

Contents in this topic

- Basic concepts from set theory
 - Logic operations in binary images
 - Morphological operations
 - Dilation and erosion
 - Opening and closing
 - Hit-or-miss transform
 - Basic morphological algorithms
 - Some applications
 - Extension to grey-level images

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

2

General Introduction to Morphology

- Morphology is an important branch in biology to deal with objects, shapes and structures of animals and plants.
 - Mathematical morphology is used to denote similar methodology adopted in image analysis.
 - The language of mathematical morphology is “set theory”.
 - In the following discussion, we shall concentrate on binary images where object pixels are binary 1, and background pixels are binary 0.

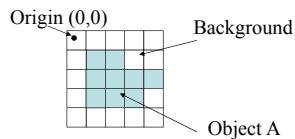
Autumn-term 2015

SF31A11/SFM1P12 Image Analysis

3

Object representation by a set

- Object A can be represented by set A with all pixel coordinates of A in a 2D integer space Z^2 .



$$A = \{(1,1), (2,1), (3,1), (1,2), (2,2), (3,2), (2,3), (3,3), (4,2)\}$$

Basics of Set Theory

- As stated before, set theory is the foundation of mathematical morphology.
- Let A be a set in a 2D integer space Z^2 .
 - If $a = (a_1, a_2)$ is an element of A , we have $a \in A$
 - If a is not an element of A , it says $a \notin A$
- The null or empty set \emptyset : no element in the set.
- A set is specified by the contents of two braces: $\{\cdot\}$
 - For the expression of $C = \{c | c = -d, \text{for } d \in D\}$, it means that set C is formed by elements c , which equal to elements d of set D multiplied by -1.

Definitions of sets (1)

- $A \subseteq B$: all elements in set A are also elements of set B , i.e. set A is a subset of B .
- The *union* of two sets A and B
 - $C = A \cup B$: Set C has all elements belonging to either A , B , or both.
- The *intersection* of two sets A and B
 - $D = A \cap B$: Set D is formed by elements belonging to both A and B .
- Two sets A and B are *disjoint* or *mutually exclusive*:
 $A \cap B = \emptyset$

Definitions of sets (2)

- **Translation** of a set A by $z = (z_1, z_2)$ is denoted by $(A)_z$ and defined as

$$(A)_z = \{c | c = a + z, \text{ for } a \in A\}$$
- **Reflection** of A is denoted by \hat{A} , and defined as

$$\hat{A} = \{c | c = -a, \text{ for } a \in A\}$$
- **Complement** of A is defined as

$$A^c = \{c | c \notin A\}$$
- The **difference** of two sets A and B is denoted by $A - B$, and defined as

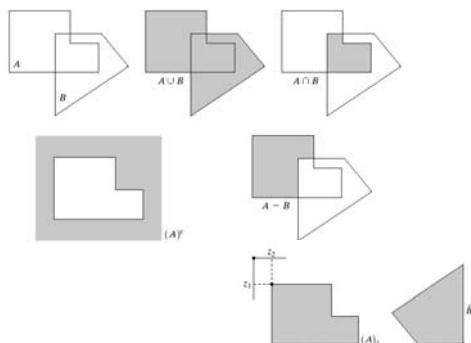
$$A - B = \{w | w \in A, w \notin B\} = A \cap B^c$$

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

7

Diagrams showing operations on sets



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

8

Binary images: logic operations

- Logic operations provide a powerful complement to implementation of binary image processing algorithms.
- There are three main logic operations used in image processing, *i.e.* AND, OR, and NOT.

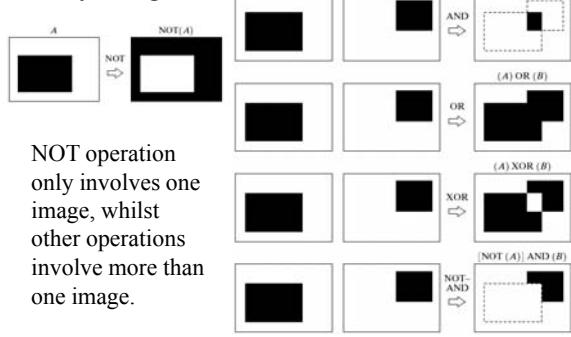
p	q	$p \text{ AND } q$ (also $p \cdot q$)	$p \text{ OR } q$ (also $p + q$)	$\text{NOT } (p)$ (also \bar{p})
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

9

Diagrams showing logic operations on binary images



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

10

Morphological operation: dilation

- Dilation is a very basic operation in mathematical morphology.
- Dilation of A by B is denoted by $A \oplus B$, and defined as $A \oplus B = \{c \mid c = a + b \text{ for } a \in A \text{ and } b \in B\}$
- Dilation is based on addition so that it is commutative, *i.e.* $A \oplus B = B \oplus A$.
- In practice, set A represents an image under processing, and set B is smaller and referred to as the *structuring element*.
- Dilation may be represented as a union of translations of the structuring element, *i.e.* $A \oplus B = \bigcup_{a \in A} (B)_a$

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

11

Structuring element

- The structuring element is a small binary image. Its pixel values are of 0 or 1.
 - The pattern of 1s and 0s specifies the shape of the structuring element.
 - The origin of the structuring element is usually one of its pixels, although this is not necessary.

1 1 1 1 1	0 0 1 0 0	0 0 1 0 0	■ ←→ Origin
1 1 1 1 1	0 1 1 1 0	0 0 1 0 0	
1 1 1 1 1	1 1 ■ 1 1	1 1 1 0 0	
1 1 1 1 1	0 1 1 1 0	0 0 1 0 0	
1 1 1 1 1	0 0 1 0 0	0 0 1 0 0	

■ ←→ Origin

Square 5x5 element

Diamond-shaped 5x5 element

Cross-shaped 5x5 element

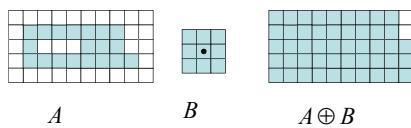
Square 3x3 element

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

12

An example of dilation



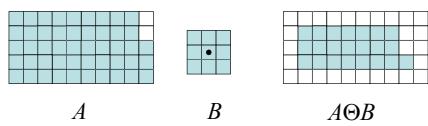
- Some important points are illustrated.
 - Dilation expands the original image.
 - Original image is contained in the dilated image, *i.e.* $A \subset A \oplus B$.
 - Dilation fill up small holes in objects.

Morphological operation: erosion

- Erosion is another very basic operation in mathematical morphology, opposite to dilation.
- Set A eroded by *structuring element* B is defined as $A \ominus B = \{c \mid \text{for each } b \in B \text{ there exists an } a \in A \text{ so that } c = a - b\}$
- Erosion is based on subtraction so that it is not commutative, *i.e.* $A \ominus B \neq B \ominus A$.
- Erosion may be described as an intersection of the negative translations of the image set.

$$A \ominus B = \bigcap_{b \in B} (A)_{-b}$$

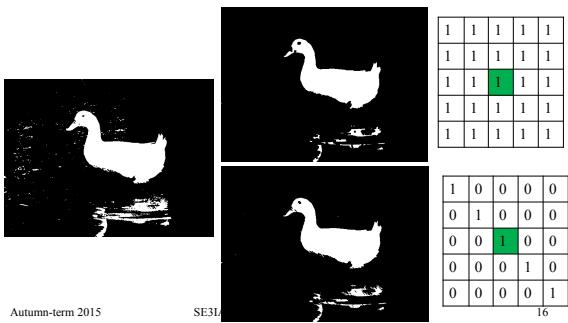
An example of erosion



- Erosion has effects on an image set opposite to dilation.
 - Erosion operation shrinks the original image.
 - The eroded image is contained in the original image.
 - Erosion removes or breaks narrow “bridges”.
 - It has been shown that $(A \oplus B) \ominus B \neq A$

An example of erosion (2)

- The shape of the structuring element matters.



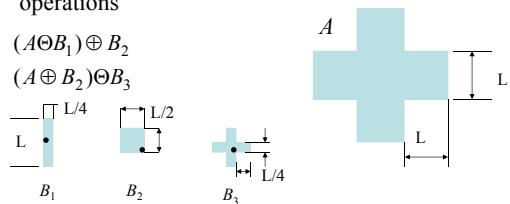
Autumn-term 2015

SE31A

16

Group discussions

- A is a set to be operated.
 - There are three different types of structuring elements as denoted by B_1 , B_2 , and B_3 .
 - Sketch the result of the following morphological operations



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

17

Morphological operation: opening

- Opening is a combined morphological operation, and defined as
$$A \circ B = (A \ominus B) \oplus B$$
 - The opening of A by B selects all those points of A each of which can be covered by some translation of the structuring element B while the translated structuring element is itself entirely contained in A .
 - In other words, the opening of A by B is obtained by taking the union of all translations of B that fit into A .
$$A \circ B = \bigcup_{B' \in T(B)} (B' \cap A) \subseteq A$$

$$A \circ B = \cup \{(B)_z \mid (B)_z \subset A\}$$

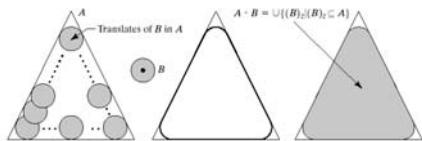
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

18

Geometric interpretation of opening

- Imagine the structuring element B as a rolling ball.
- The boundary of $A \circ B$ is given by the points on the boundary of B that are closest to the boundary of A as B is rolled around from the **inside** of A .



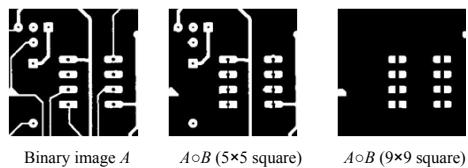
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

19

Example of opening operation (1)

- Opening can open up a gap between objects connected by a thin bridge of pixels.
- Regions that have survived the erosion are partially restored to their original size by the dilation.



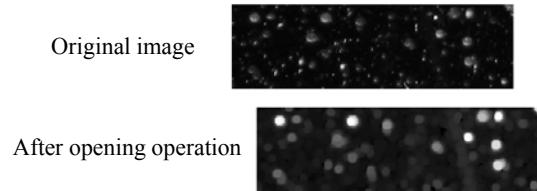
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

20

Example of opening operation (2)

- A disk-shaped structuring element with a radius of 5 pixels is applied to the original image.
- Snowflakes with a radius less than 5 pixels have been removed by the opening operation.



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

21

Morphological operation: closing

- Closing is another combined morphological operation, and defined as
$$A \bullet B = (A \oplus B) \ominus B$$
 - The closing of A by B includes all points satisfying the condition that each time one of these points is covered by a translation of the reflected structuring element \hat{B} , there must be at least one point in common between A and the translation of \hat{B} .

$$A \bullet B = \{x \mid x \in (\hat{B})_t \text{ implies } (\hat{B})_t \cap A \neq \emptyset\}$$

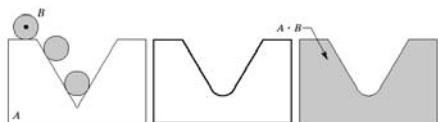
Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

22

Geometric interpretation of closing

- Imagine the structuring element B as a rolling ball.
 - The boundary of $A \bullet B$ is similarly obtained, except that we now roll B from the **outside** of A .
 - Closing smoothes the boundary, and eliminates small holes and fills gaps in the boundary



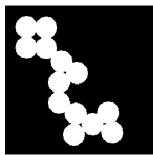
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

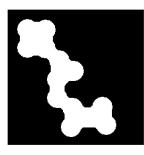
23

Example of closing operation

- A disk-shaped structuring element with a radius of 10 pixels is applied to the original image.
 - The gaps are closed.



Original image



After closing operation

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

24

Morphological operation: hit-or-miss transform

- The morphological hit-or-miss transform is used to extract various image features from an object. It is defined as

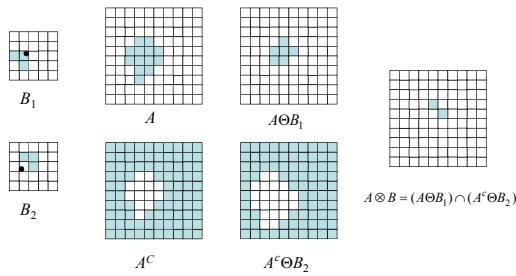
$$A \otimes B = (A \ominus B_1) \cap (A^c \ominus B_2)$$
- Where A is the set representing the object, and $B=(B_1, B_2)$ is called a composite structuring element, satisfying $B_1 \cap B_2 = \emptyset$.
- In order to extract features from object A , it is necessary to carefully design or choose the composite structuring element $B(B_1, B_2)$.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

25

Example of the hit-or-miss transform



The upper right-hand corner points of object A are found by hit-or-miss transform.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

26

Morphological algorithms

- Based on the basic morphological operations, e.g. erosion, dilation, opening, closing, and hit-or-miss, many morphological algorithms can be generated.
- Some morphological algorithms are listed here
 - Boundary extraction
 - Region filling
 - Extraction of connected components
 - Thinning and thickening
 - Convex hull
 - Skeletons: sketching frame
 - Pruning: removing parasitic components

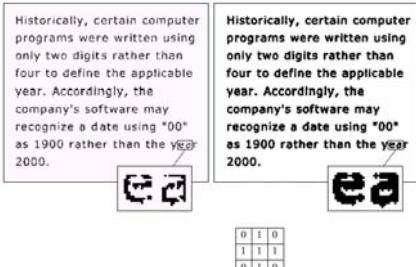
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

27

Selected applications (1)

- Enhancement of extracted characters



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

28

Selected applications (2)

- Enhancement of fingerprint



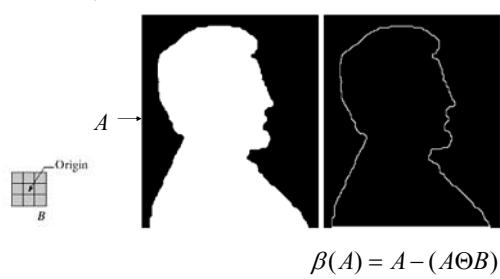
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

29

Selected applications (3)

- Boundary extraction



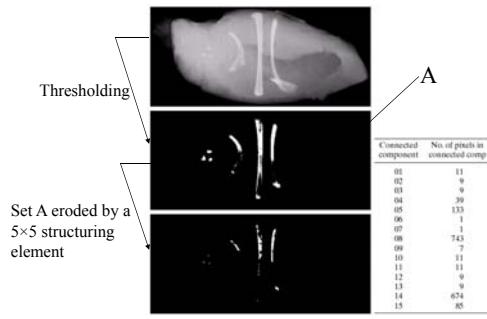
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

30

Selected applications (4)

- Extraction of connected components



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

31

Grey-scale morphology

- Grey-scale image $f(x,y)$, structuring element $b(x,y)$
- Dilation: enhance brightness

$$(f \oplus b)(s,t) = \max \{f(s-x,t-y) + b(x,y) \mid (s-x), (t-y) \in D_f; (x,y) \in D_b\}$$
- Erosion: enhance darkness

$$(f \ominus b)(s,t) = \min \{f(s+x,t+y) - b(x,y) \mid (s+x), (t+y) \in D_f; (x,y) \in D_b\}$$
- Opening: remove bright details

$$f \circ b = (f \ominus b) \oplus b$$
- Closing: remove dark details

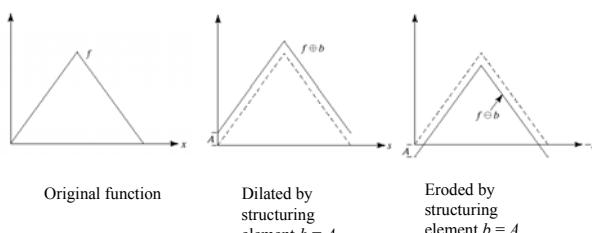
$$f \bullet b = (f \oplus b) \ominus b$$

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

32

Geometric interpretation of grey-scale dilation and erosion

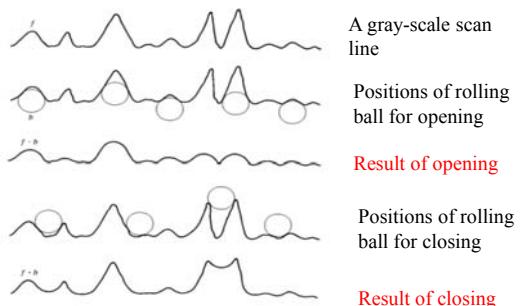


Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

33

Rolling ball analogy of opening and closing

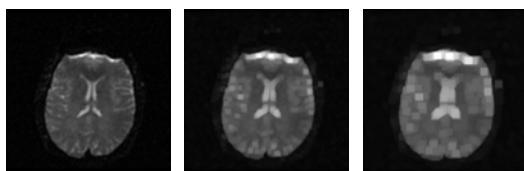


Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

34

Example of grey-scale dilation



Original function

Dilated by 3×3
unit structuring
element

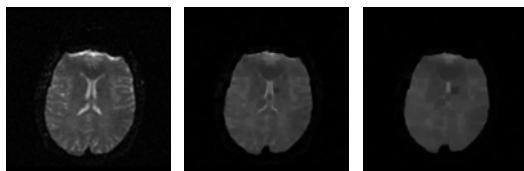
Dilated by 5×5
unit structuring
element

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

35

Example of grey-scale erosion



Original function

Eroded by 3×3
unit structuring
element

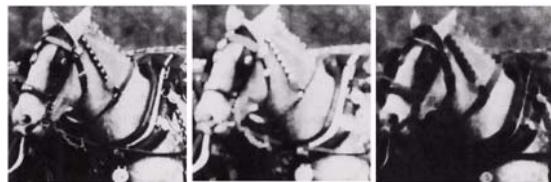
Eroded by 5×5
unit structuring
element

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

36

Example of grey-scale dilation and erosion



Original function

Dilated by 5x5
unit structuring
element (enhance
brightness)Eroded by 5x5
unit structuring
element (enhance
darkness)

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

37

Example of grey-scale opening and closing



Original function

Opening by 5x5
unit structuring
element (remove
bright details)Closing by 5x5
unit structuring
element (remove
dark details)

Autumn-term 2015

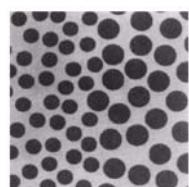
SE3IA11/SEMIP12 Image Analysis

38

Application of grey-scale morphology:
textural segmentation

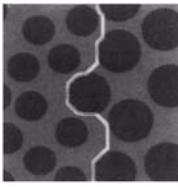
Objective: separate two texture regions

1. Closing the input image by a structuring element having size of the small blobs – light left;



Original image

2. Opening the output image by a structuring element with size larger than gaps between larger blobs – dark right;



Final segmentation

3. A simple thresholding to separate light left and dark right regions.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

39

End of the two lectures

- Summary what you have learned in the two lectures.
 - Think about to what applications mathematical morphology can be used (many – parallel to and associated with other image processing techniques).

Autumn-term 2015

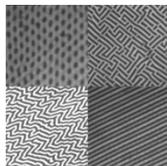
SE31A11/SEMIP12 Image Analysis

40

SE3IA11/SEMIP12 Image Analysis

Image Segmentation

1. Chapter 10 in the Gonzalez&Woods book
2. Chapter 2.1 *The handbook of pattern recognition and computer vision* (2nd edition – 1999)
3. Chapter 2.6 *The handbook of pattern recognition and computer vision* (3rd edition – 2005)
4. *Handbook of texture analysis* (2008)



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

1

Contents of this topic

- Basics of image segmentation (*Two properties: grey level discontinuity and texture similarity.*)
 - Thresholding
 - Region-based segmentation
 - Morphological watersheds
- Texture analysis in image segmentation
 - Grey-level co-occurrence matrix
 - Gabor transform and Gabor wavelets
 - Local binary patterns (LBP)
- The use of motion in segmentation (separation of background and foreground moving objects)
- Selected applications

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

2

Basics of image segmentation

- Segmentation is an essential preliminary step to convert input images into outputs attributes which are meaningful to computer.
- Operations of image segmentation involve grouping pixels with similar characteristics for feature extraction and then automated object recognition.
- Image segmentation algorithms are based on one of two basic properties of intensity values: discontinuity and similarity.
- We shall concentrate our discussions on detection of grey level similarity.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

3

Basic concept of thresholding

- For a grey level image $f(x,y)$, it is assumed that $f(x,y)$ is composed of light objects on a dark background.
- In such a way, object and background pixels can be grouped into two dominant modes in its histogram distribution.
- To extract objects, there exists a grey value T so that
$$f(x,y) = \begin{cases} 0 & \text{for all } f(x,y) \leq T & \text{Background pixels} \\ 1 & \text{for all } f(x,y) > T & \text{Object pixels} \end{cases}$$
- T is the threshold to separate objects and background pixels in the grey level image.

Global thresholding

- A single threshold T is used to partition the entire image histogram.
- Segmentation is done by scanning the image pixel by pixel and labelling each pixel as object or background, depending on whether the grey level of that pixel is greater or less than the value of T .
- An algorithm to obtain T is summarised as following.
 - Select an initial estimate for T ;
 - Segment the image using T . This produces two groups of pixels, $G_1 > T$ and $G_2 \leq T$;
 - Compute the average grey level values μ_1 and μ_2 for the pixels in regions G_1 and G_2 ;
 - Compute a new threshold value $T = \frac{1}{2}(\mu_1 + \mu_2)$
 - Repeat steps 2-4 until the difference in T and T^{n-1} is smaller than a predefined parameter T_0 , e.g. $T_0 = 0$.

Local to global: adaptive thresholding

- For images with uneven illumination, global thresholding may fail in object extraction.
- Localisation is a solution to deal with segmentation on images with uneven illumination.
- There are various approaches to localise information within an input image.
- Here we introduce a general approach by dividing the original image into sub-images.
- Two issues need to be addressed.
 - How to subdivide the image.
 - How to estimate the threshold for each resulting sub-image.

Niblack thresholding

- W. Niblack (1986) proposed a method to setup the threshold in the adaptive thresholding.
 - The method suggests to calculate a threshold surface by shifting a window across the image, and use local mean μ_i and standard deviation σ_i for each centre.
- $$T_i = \mu_i(x, y) + k\sigma_i(x, y)$$
- where k is a constant, which is highly tuneable to separate objects well.
- The size of neighbourhood is also highly tuneable so that, as it is chosen, it will be small enough to preserve local details, and large enough to suppress noises.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

7

Optimal global and adaptive thresholding: Otsu's algorithm

- The optimal way to determine the threshold from a bimodal histogram is due to Otsu's algorithm ("A Threshold Selection Method from Gray-Level Histograms", IEEE Transactions on System, Man, and Cybernetics. SMC-9(1), 1979)
- Let's use $z = f(x, y)$ to denote grey level values, which are viewed as random quantities, and their histogram is considered as an estimate of their Probability Density Function (PDF), $p(z)$.
- $p(z)$ is the mixture of two PDFs, one for the bright mode and the other for the dark mode in the histogram distribution.
- The basic idea of Otsu's algorithm is to reduce the within-class variance.

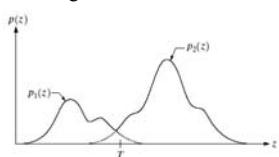
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

8

Otsu's algorithm (1)

- Assume that there are two PDFs $p_1(z)$ for object and $p_2(z)$ for background as shown in the figure below.
- The mixture PDF describing the overall grey-level variation in the image is $p(z) = P_1 p_1(z) + P_2 p_2(z)$
- where $P_1 + P_2 = 1$ denote the probability (a number) that a random pixel with value z is an object pixel P_1 or background pixel P_2 .
- The aim is to find T which minimises the average error for the object/background segmentation.



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

9

Otsu's algorithm (2)

- Let's write the segmentation error as following
$$E(T) = P_2 E_1(T) + P_1 E_2(T) = P_2 \int_{-\infty}^T p_2(z) dz + P_1 \int_T^{\infty} p_1(z) dz$$
- In order to find the threshold value for which this error is minimal, differentiation of $E(T)$ with respect to T is applied, and the result is set to zero. We have
$$P_1 p_1(T) = P_2 p_2(T)$$
- From the above equation, we can work out the value T to minimise the error E .
- When $P_1 = P_2 = 0.5$, T is the intersection point of $p_1(z)$ and $p_2(z)$.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

10

Multispectral thresholding

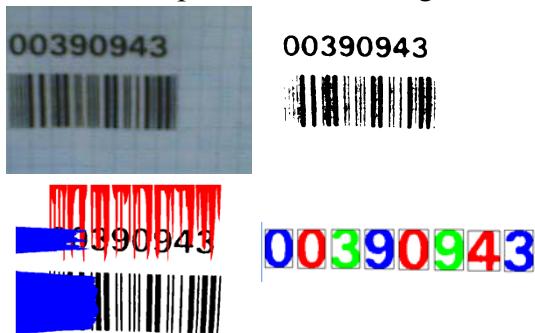
- When images are presented in the multispectral manner, the thresholds should be determined based on several variables.
- Colour imaging is a good example, in which each pixel has three values, *i.e.* RGB.
- For colour images, thresholding can also happen in other colour models, *e.g.* HSI.
- For example, it has been proved that "hue" in the HSI model is stable to human skin (paper: "Features of human skin in HSV colour space and new recognition parameter", *Optoelectronics Letters*, vol. 3, no. 4, 2007).
- We can use this characteristic to segment human skin in images based on supervised segmentation.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

11

Example of thresholding



Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

12

Region-based segmentation

- There are two categories, *i.e.* region growing and region splitting/merging.
 - Region growing
 - Select a set of “seed” points
 - Append to each seed those neighbour pixels that have properties similar to the seed.
 - Region splitting/merging
 - Subdivide an image into a set of arbitrary regions
 - Split or merge the regions in an attempt to satisfy pre-defined conditions
 - A quadtree algorithm can be used for the subdivision. (a quadtree is a tree in which nodes have exactly four descendants.)

Autumn-term 2015

SF31A11/SFM1P12 Image Analysis

13

Example of region growing



(a) Original image (b) Seeds (c) Growing regions

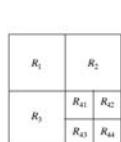
- From (a) to (b), seeds are selected as pixel grey values equal to 255.
 - From (b) to (c), two criteria have been chosen.
 - The absolute grey-level difference between any pixel and the seed should be less than 65
 - The pixel has to be 8-connected to at least one pixel in the region.

Autumn-term 2015

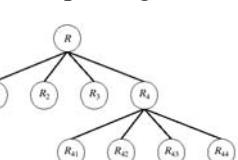
SE31A11/SEMIP12 Image Analysis

14

Example of quad-tree splitting technique



(a) Partitioned image



(b) Corresponding quadtree

- An image iteratively splits into smaller regions in the quadtree manner.
 - The termination of partition is set if a region has identical properties (e.g. same grey value.)

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

15

Segmentation by morphological watersheds

- The basic concept of watersheds is based on visualising an image in three dimensions: two spatial coordinates versus grey levels. A regional minimum is formed with pixels having low grey values (called catchment basin or watershed of that minimum).
 - The idea of this segmentation method is to find the watershed lines when flooding water comes through holes in each regional minimum at a uniform rate.
 - When rising water reaches a stage, a dam needs to be built to prevent the merging.
 - These dam boundaries correspond to the divide lines of the watersheds.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

16

Watershed segmentation algorithm (1)

- In an image $f(x,y)$, there are R sets M_1, M_2, \dots, M_R denoting the coordinates of the points in regional minima.
 - g_{\min} and g_{\max} are minimum and maximum grey values of $f(x,y)$, respectively.
 - $T[n]$ represents a set of coordinates (x,y) for which $f(x,y) < n$.
$$T[n] = \{(x, y) \mid f(x, y) < n\}$$
 - $C[n]$ denotes the union of all portions of the flooded catchment basins at n :
$$C[n] = \bigcup_{i=1}^R C_n(M_i)$$

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

17

Watershed segmentation algorithm (2)

- The algorithm is initialised with $C[g_{\min}+1] = T[g_{\min}+1]$.
 - The algorithm then proceeds recursively, assuming at step n that $C[n-1]$ has been constructed. Construction of $C[n]$ from $C[n-1]$ can be achieved based on the following three conditions of the result of $q \cap C[n-1]$ as
 - is empty. A new minimum is encountered. Connected component q is incorporated into $C[n-1]$ to form $C[n]$.
 - contains one connected component of $C[n-1]$. q lies within the catchment basin of some regional minimum. q is incorporated into $C[n-1]$ to form $C[n]$.
 - contains more than one connected component of $C[n-1]$. All, or part of a ridge separating two or more catchment basins is encountered. Further flooding would cause the water level in these catchment basins to merge.

where $q \in \mathcal{Q}[n]$ denoting the set of connected components in $T[n]$

 - At condition 3, a dam (or dams) is built to segment the image.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

18

Texture analysis in image segmentation

- Evidence has been provided by recent findings from psychophysics, neurophysiology and computer vision for a framework in which objects and scenes are represented as collections of viewpoint-specific features rather than 2D templates or 3D models.
 - Texture information in images can provide various features although no precise, general definition of texture exists in the computer vision literature.
 - Many approaches have been attempted, and some successful techniques are introduced in the following slides.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

19

Grey level co-occurrence matrix (GLCM)

- Co-occurrence matrix is given as second-order statistics to represent image texture features, e.g. positional relationships between pixels.
 - Assume that an image has G grey levels. For a pixel pair separated by a distance d in direction θ [(d, θ) is called a displacement vector], there exists a $G \times G$ matrix M in which the element at the i th row and the j th column represents the number of occurrences of all pixel pairs separated by (d, θ) and satisfying the condition that the first pixel has grey level i and the second pixel grey level j .
 - Such a matrix M is defined as a GLCM.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

20

An example of GLCM

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

4×4 image with
4 grey levels

2	2	1	0
0	2	0	0
0	0	3	2
0	0	0	1

GLCM with
 $d=(1, 0^\circ)$

1	0	0	0
1	1	0	0
3	1	0	0
0	0	2	0

GLCM with
 $d=(1, 135^\circ)$

- The example shows that GLCM may have very high dimension (for 8-bits grey level image, the size of GLCM is 256×256).
 - It is better to reduce the dimension for texture feature representation.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

21

Making the GLCM symmetrical

- In practice, texture calculations require a symmetric matrix. (<http://www.fp.ucalgary.ca/mhallbey/tutorial.htm>)
 - A symmetrical GLCM is formed as M by

$$M = GLCM + GLCM^T$$

- Example: GLCM with $d=(1, 0^\circ)$

$$\begin{array}{c}
 \begin{array}{|c|c|c|c|} \hline
 4 & 2 & 1 & 0 \\ \hline
 2 & 4 & 0 & 0 \\ \hline
 1 & 0 & 6 & 2 \\ \hline
 0 & 0 & 2 & 2 \\ \hline
 \end{array}
 & = & \begin{array}{|c|c|c|c|} \hline
 2 & 2 & 1 & 0 \\ \hline
 0 & 2 & 0 & 0 \\ \hline
 0 & 0 & 3 & 2 \\ \hline
 0 & 0 & 0 & 1 \\ \hline
 \end{array}
 & + & \begin{array}{|c|c|c|c|} \hline
 2 & 0 & 0 & 0 \\ \hline
 2 & 2 & 0 & 0 \\ \hline
 1 & 0 & 3 & 0 \\ \hline
 0 & 0 & 2 & 1 \\ \hline
 \end{array}
 \end{array}$$

Autumn term 2015

SE31A11/SEMIR12 Image Analysis

22

Normalised symmetrical GLCM

- Elements in the symmetrical GLCM are normally represented by their probability. It is called Normalised Symmetrical GLCM. We use N to represent it.

$$N(i, j) = \frac{M(i, j)}{\sum_{i \in G, j \in G} M(i, j)}$$

=>

.154	.077	.038	0
.077	.153	0	0
.038	0	.231	.077
0	0	.077	.077

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

23

Properties of N

- It is a square matrix.
 - It is symmetrical around the diagonal with each element value less than 1.
 - The diagonal elements represent pixel pairs with no grey level difference.
 - If there are high probabilities in these elements, the image does not show much contrast – most pixels are identical to their neighbours.
 - The farther away from the diagonal, the greater the difference between pixel grey levels

Autumn term 2015

SE31A11/SEMIR12 Image Analysis

24

Entropy and energy of GLCM

- Commonly entropy and energy of GLCM are used for texture feature representation.
 - Entropy of GLCM is defined as

$$-\sum_{i \in G} \sum_{j \in G} N_{(d, \theta)}(i, j) \log N_{(d, \theta)}(i, j)$$

- Energy of GLCM is defined as

$$\sum_{i \in G} \sum_{j \in G} N_{(d,\theta)}^2(i,j)$$

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

25

Gabor transform in image analysis

- The window Fourier transform (or short Fourier transform) is introduced to analyse local information in images in the spatial domain.
 - A one-dimensional window Fourier transform of signal $f(x)$ is defined as

$$F_w(u, \xi) = \int_{-\infty}^{+\infty} f(x) w(x - \xi) e^{-j2\pi ux} dx \quad (1)$$

- When the window function $w(x)$ is a Gaussian function, the transform becomes a Gabor transform.
 - Gabor function therefore can be expressed as a multiplication of a Gaussian function with a harmonic function.

$$g(x) = ae^{-\frac{(x-x_0)^2}{2\sigma^2}} \times e^{-j2\pi ux}$$

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

26

Gabor wavelets

- When a function $f(x)$ is scaled in time by b , it can be expressed as $f(bx)$. The function is contracted if $b>1$, and expanded when $b<1$. The wavelet transform can be written as

$$W_{f,b}(u, \xi) = \frac{1}{\sqrt{b}} \int f(x) h^*(\frac{x-\xi}{b}) dx \quad (2)$$

- The impulse response h^* of the filter bank is defined to be scaled versions of the same prototype function $h(x)$. We can write $h(x)$ as

$$h(x) = w(x)e^{-j2\pi ux} \quad (3)$$

- Substitute equation (3) to equation (2), and compare it with equation (1), we can see that Gabor transform obeys the wavelet transform. Therefore it is called Gabor wavelet function.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

27

2D Gabor wavelet function

- We use equation (4) to represent a 2D Gabor wavelet function

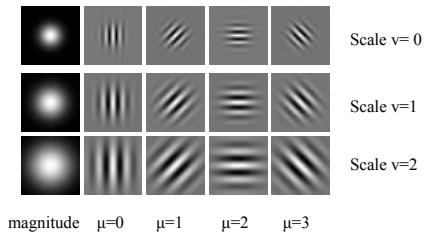
$$g_{u,v}(x,y) = \frac{\|k_{u,v}\|^2}{2\pi\sigma^2} e^{-\frac{\|k_{u,v}\|^2(x^2+y^2)}{2\sigma^2}} [e^{ik_{u,v}(x+y)} - e^{-\frac{\sigma^2}{2}}] \quad (4)$$
- where $k_{u,v}$ is the wave vector defined by both orientation u and scale v .
- The parameter σ determines the ratio of the Gaussian window width and wavelength.
- In Equation (4), the first term in the square bracket is the harmonic part in the Gabor kernel, and the second term contributes to the property of DC free.
- The 2D Gabor wavelet function can be used to transform an image to the Gabor space. It has proved that this process is closely related to processes in the primary visual cortex

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

28

Selected 2D Gabor wavelet kernels



- The figures show the magnitude and real part of Gabor kernels in 3 scales and 4 orientations as in equation (4).

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

29

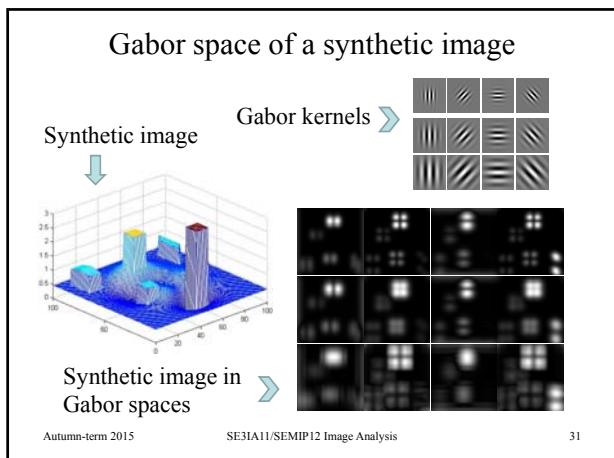
Gabor wavelets in image segmentation

- An input image is transformed into Gabor space by convolving a bank of Gabor kernels to the image at multiple scales and orientations.
- The texture feature for each pixel is computed as the response of the transformed value in the Gabor space (grouped pixels within a window may be used for the computation).
- A cluster analysis is performed in the Gabor feature space to label pixels with similar values of the transform responses (other measures can be used for a group of pixels, e.g. mean, standard deviation.)
- The labels segment the input image.

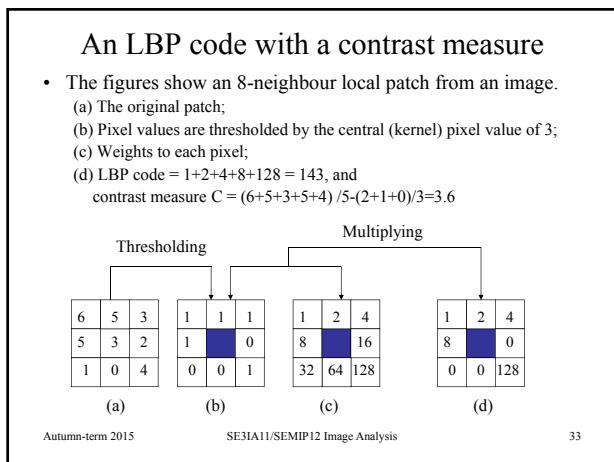
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

30



- LBP: local binary patterns
- LBP is regarded as a truly unifying approach for texture analysis in contrast to statistical and structural approaches.
 - LBP forms local patterns of an image, instead of trying to explain texture formation on a pixel level.
 - In this method, an LBP code is optionally combined with other measure, e.g. contrast to represent local information (pattern) of images.
 - A basic LBP code for a neighbourhood can be defined by values of neighbour pixels with weights assigned to them.
- Autumn-term 2015 SE3IA11/SEMIP12 Image Analysis 32



A general definition of LBP:
circularly symmetric neighbour sets

- In practice, an LBP operator is extended to a circularly symmetric neighbour set, which is rotation invariance.
 - Mathematically LBP can be defined as

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c) 2^p$$

- Where P is the number of neighbour pixels, R the radius of the set, g_c the kernel pixel value, g_p neighbour pixel values, and

$$s(g_p - g_c) = \begin{cases} 1 & \text{for } (g_p - g_c) \geq 0 \\ 0 & \text{for } (g_p - g_c) < 0 \end{cases}$$

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

34

Potential applications of LBP

- LBP can be used for image segmentation by identifying image features with similar LBP distributions (textural similarity).
 - LBP code with conjunction to other measures can be used as features to represent image properties. Based on these features, applications have been successfully used in the following areas.
 - Industrial visual inspection
 - Image retrieval: searching for specific texture patterns
 - Scene analysis: interpretation of natural scene with texture analysis
 - Human face detection and recognition, and more....

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

35

Other state-of-the-arts methods

- Image texture analysis has a long research history in machine vision and image processing. Many algorithms have been developed to explain visual texture, extract texture features, and interpret images based on texture analysis.
 - Here are listed other popular methods used for the purpose.
 - Markov Random Field (MRF): It is used to modelling image as it is able to capture the local contextual information in an image.
 - Fractals: it is developed based on the fact that many natural surfaces have a statistical quality of roughness and self-similarity at different scales.
 - Both of them are categorised as model based texture analysis methods, and popularly used in image segmentation and object extraction from images.

Autumn-term 2015

SE31A11/SEMIP12 Image Analysis

36

The use of motion in segmentation

- The basic approach for detecting changes between two image frames $f(x,y,t_i)$ and $f(x,y,t_j)$ is to compare them pixel by pixel to form a difference image.
 - A difference image may be defined as
- $$\delta_{ij}(x,y) = \begin{cases} 1 & \text{if } |f(x,y,t_i) - f(x,y,t_j)| > T \\ 0 & \text{otherwise} \end{cases}$$
- T is a pre-defined threshold. $\delta_{ij}(x,y)$ is a binary image, where 1s represent moving objects (foreground) and 0s background.
- Misclassification caused by noises and illumination uncertainty can be reduced by forming 4- or 8-connected regions of 1s in $\delta_{ij}(x,y)$ and ignoring smaller components.

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

37

Gaussian model in motion segmentation

- It is important to establish a reference image in motion detection. The Gaussian model is one of the successful models used in this purpose.
 - For a sequence of image frames $z = f(x,y)$, e.g. from a video sequence, the first k frames may be used to establish a Gaussian model as the reference image for each pixel. The PDF of a Gaussian model can be written as
- $$p(z_{ij}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z_{ij} - \mu_{ij})^2}{2\sigma^2}}$$
- where μ_{ij} is the mean, and σ the standard deviation of the Gaussian model at pixel position of $x = i, y = j$.
 - The new coming frames are tested against the reference image to form the difference image as

$$\delta_{ij}(x,y) = \begin{cases} 0 & \text{if } p(z_{ij}) > p_0 \text{ (a pre-defined threshold)} \\ 1 & \text{otherwise (motion objects)} \end{cases}$$

Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

38

Application: motion segmentation



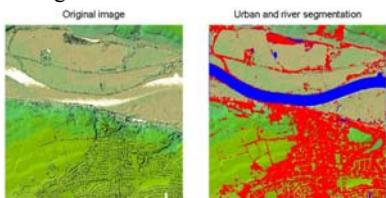
Autumn-term 2015

SE3IA11/SEMIP12 Image Analysis

39

Application: terrain feature segmentation

- Image is transformed to Gabor spaces;
- Thresholding is applied to separate the smooth field and building/tree areas.



Research results were published on *ICPR2006*,
Wei&Bartels, Vol. I, pp 667-670 .

Further thinking questions

- What are popular methods used in image segmentation? (summary)
- How can a thresholding technique be optimised in an effective image binarisation? (from global to local, from fixed to adaptive)
- What is the texture based image segmentation? Search for popular algorithms in this area.
- How could motion information be used in image segmentation?

End of the two lectures

- Image segmentation is a challenging research area, and it is also a primary stage for object classification.
- Summary what you have learned in the two lectures, and extend the knowledge to a wide scope.
- Compare and contrast different techniques used for image segmentation.
