# 1 Introduction to Media Coding and Processing

**• What is a digital image? (Lecture 1)**

Simple Image Formation Model: Illumination source + scene + imaging system + Projection of scene onto image plane = Digitized Image

Imaging System = Array of Sensors

Image Sampling and Quantization: Converting Sensed Data into Digital Form

* Sampling = Digitizing Coordinate value
  + Size
  + Shape
  + Pixel Location
* Quantization = Digitizing Amplitude value
  + Brightness
  + Contrast
  + Colour
  + Intensity

Digital Image = 2D Grid defined as function *f(x,y)*

* *x,y* = spatial coordinates of a pixel
* *f =*  amplitude of any pairs *x,y* of coordinates (Intensity/Grey Level)

**• Different types of digital images (Lecture 1)**

Image Modalities:

2D Images:

* Binary = 1s and 0s
* Grey = 0-255
* Colour = RGB Colour Space

2.5D Images:

- Stores Distance from Camera

3D Images:

* Stores 3D Data

**• What is digital image processing? (Lecture 2)**

Image/video Processing = Subclass of signal processing concerned with images

Image Processing Areas:

* Image Compression/Coding = Decrease redundancy of image data in order to store/transmit data efficiently (lossy/lossless)
  + Art of representing images with least information consistent which achieving acceptable image quality

Compression Ratio = Uncompressed Size / Compressed Size

Lossy vs Lossless:

* Lossless = Perfect Reconstruction from Data
  + Decoded data exact copies of original
  + Low compression ratio
  + Used for texts, programs, code, spreadsheet
* Lossy = Certain amount of data discards – cannot recreate image
  + Decoded data are approximates of original
  + High compression ratio
  + Used for image, video, audio
  + Focus on user perception

Why Compress? = Enables technology for image storage and distribution:

* Less storage
* Faster distribution

Criteria for a good coding scheme:

* Compression Ration
* Compression/Decompression Complexity (Speed)
* Implementation Easiness
* Progressive Easiness
* Scalability

Image Coding and Decoding System:

Image => [Mapper -> Quantizer -> Symbol Coder] = Compressed Image

Compressed Image => [Symbol Decoder -> Inverse Mapper] => Image

A picture containing text, clipart

Description automatically generatedImage enhancement = Improve human perception or provide better input for automated image processing

Computer Vision = Enables computer to see and understand images same way as humans and provide appropriate output:

* Know what is there by looking
* Understand where things are in the world
* Know their 3d properties
* Know actions taking place

# 2 Image Transformation - Intensity Transformation

Fundamentals of digital images:

* Made from pixels arranged in columns and rows
* Binary Image = Matrix of 1s and 0s
* Greyscale Image = Matrix of 0 to 255
* Colour: Three Components: R, G and B. Each Matrix 0 to 255

Bit Depth = Number of bits used to represent a pixel in rgb space

Colour Models:

* RGB Model
* Y Cb Cr: Y is luminance, Cb and Cr are colour
* HSV: Hue Saturation Value

Spatial Domain = Image Plane itself

* g(x,y) = T[f(x,y)]
* f and g are 2D matrices of size W x H
* T is a matrix of size w x h, which is much smaller than W x H
* Value of each pixel of g is result of applying operator T to that location in f
* T(f) = g

Frequency Domain = Consider image as a 2D Signal and apply Fourier or DCT transform

* Processing Image in transformed domain
* Apply inverse transform to return image to spatial domain
* DCT/DFT transforms spatial matrix to new space where components have different meaning
* Image processing takes place in frequency domain
* Inverse is called iDCT

Basic Image Transformations:

* Image Negatives = Reverse Image Intensity Levels
  + g(x,y) = L – 1 – f(x,y)
* Log Transformation = Expands dark pixels whilst compressing brighter pixels
  + Increases dynamic range (max and min intensity values) of low intensities
  + When dynamic range is greater than displaying device, lower intensity values are suppressed
  + Log Transformation enhances lower intensity values to show more details in image
  + g(x,y) = c log( 1 + f(x,y) )
* Power-law (Gamma) transformation
  + g(x,y) = cf(x,y)Y
  + c and Y are positive constants
  + A family of different transformation curves obtained by varying Y, for example log transform
* Piecewise Linear Transformation: Have two points and contrast stretch based upon them
  + Used for contrast stretching
* Bit plane slicing: Slice each plane from most to least significant (Plane 8 to Plane 1)

# 3 Image Histogram and its Applications

Histogram = Graphical way to represent data

* Height of each bar is number of data points whose value fall into range of bar (bin)
* Define over all possible intensity values
* For each intensity level, value is equal to number of pixels with that intensity

Normalised Image Histogram = Histogram function divided by total number of pixels of image

* p(rk) = h(rk)/n = nk/n
* Gives measure of how likely a pixel is to have a certain intensity

Table

Description automatically generatedSum of normalised histogram function over range of all intensities is 1

Applications:

* Detecting image acquisition issues:
  + Brightness
  + Over/under exposure
  + Contrast
* Point operators can be used to alter histogram:
  + Addition
  + Multiplication
  + Exp and Log
  + Histogram Equalization
  + Histogram Matching

We cannot reconstruct image from histogram

Histogram Equalization = Improve contrast in images by spreading out most frequent values

* Apply a point operation that changes histogram of modified image into uniform distribution

Histogram Equalization Algorithm:

1. Create image histogram
2. Create normalised image histogram
3. Calculate cumulative distribution
4. Table

   Description automatically generatedApply transformation = New intensity = floor (maximum intensity x cumulative)

Tries to make intensity distribution uniform

Not always good when irregular initial distribution

Histogram Matching/Specification = Find transformation that matches distribution of original image

A picture containing shape

Description automatically generated

1. Find histogram of original image and specified image
2. Calculate inverse of specified
   1. Pick one by one the values from “Transformation (original)” column
   2. find it in the “Transformation (specified)” column and note down the value in the “Intensity” column
   3. If value doesn’t exist in specified, find nearest one
   4. If multiple values exist, pick lowest one

Table

Description automatically generated

Original -> Specified -> Intensity = Final Map i.e. **0 -> 1 -> 1** or **3 -> 1 -> 0**

Colour Histograms = Represent colour distribution in image (Left to right == 0% to 100%)

Chart, histogram

Description automatically generated

Approaches to histogram equalisation on colour images:

1. Split Channels and Equalise each Channel separately
   * Inappropriate as it would be disturbing intensity values
2. Convert colour space of RGB image into another colour space that separates intensity values from colour components. Then perform histogram equalisation of intensity plane. Then convert back to RGB.
   * Some colour spaces are:
     + HSV
     + YUV
     + YCbCr (preferred)

# 4 Spatial filtering

(look at part two)

Spatial filter consists of a neighbourhood and a mask/filter/kernel

* neighbourhood is rectangular and smaller than image

Types of special filters:

* smoothing spatial filters = used for blurring and noise reduction
  + linear filters (mean filters).
    - Averaging filter.
    - Waited averaging filter (Gaussian filter).
  + Order statistics (non-linear) filters.
    - Minimum filter
    - maximum filter
    - median filter
* sharpening spatial filter (derivative filter) = removal of blurring and highlighting edges
  + unsharp masking
  + first order derivative
  + second order derivative

**Smoothening**

linear filters

A picture containing text, electronics

Description automatically generatedaveraging filter = finds average from filter and sets as output. AKA convolution.

Convolution: each output pixel is the weighted sum of neighbouring input pixels.

1. Convolve each column of input image with vertical projection of kernel to create intermediate image
2. A screenshot of a computer

   Description automatically generated with low confidenceConvolve each row of intermediate image with horizontal projection of kernel

Averaging Filter:

* Reduce details of image by changing **filter size**

Weighted averaging filter:

* Pixels multiplied by different coefficients. Centre multiplied by higher value than average. Gives more importance to some pixels
* Control reduction by changing **filter size** and **filter weights** so more flexible than averaging filter
* Gaussian Filter is an example of this

Gaussian Filter: Bell shaped distribution used for a weighted averaging filter

* We use 2d Gaussian function as we need a dimension for x and a dimension for y

A picture containing text, clock, watch, gauge

Description automatically generated

Mean = (x,y) and σ = Weight

Weight affects the width of the distribution

Diagram

Description automatically generated with low confidence

Higher kernel size = Blurrier image

Non linear filters:

Minimum Filter = value of centre replaced by smallest value in window

Maximum Filter = value of centre replaced by largest value in window

Median filter = first neighbouring pixels are sorted and original values of pixel replaced by median of list

**Sharpening**

I­original - Iblur = Idetail

I­original + Idetail = Isharp

Isharp = 2I­original - Iblur

# 5 Edge Detection

Edge:

* significant local change in image intensity
* Boundary between two regions with distinct intensity level properties

Goal of edge detection:

* Identify sudden changes in image
* Convert 2d image into set of points where image intensity changes rapidly

Why do we care about edges?:

* More compact than pixels (Less data to be processed)
* Semantic and shape information can be encoded in edges (Object detection and recognition)
* Recover geometry and viewpoint
* Image segmentation (separates objects from background)
* Motion analysis (reliable object tracking)

Origin of edges:

* Geometric events:
  + Discontinuity in surface orientation
  + Discontinuity in depth
  + Discontinuity in surface colour/texture
* Non geometric events:
  + Discontinuity in illumination (shadows)

A picture containing text, indoor

Description automatically generated

Types of edges:

* Step edge = image intensity sharply changes from one value to another
  + Occur in images generated by computers
  + Clean and ideal edges
  + Occur over distance of **one** pixel

Graphical user interface, application

Description automatically generated

* Ramp edge = step edge when intensity changes over finite distance rather than immediate
  + Occurs in images where edges are blurred and noisy
  + A picture containing diagram

    Description automatically generatedEdge point is now any point contained in ramp
* Roof edge = intensity changes from one value to another over a finite distance and then returns to starting value over a finite distance
  + Mirrored ramp ?
  + Generated by intersection of services
  + A picture containing shape

    Description automatically generatedUsed in line drawings and satellite images with thin features (roads)

Steps of edge detection algorithm:

1. Calculate rate of intensity change at each pixel in image
   1. By using magnitude of gradient at each pixel (derivatives)
2. Make decision based on threshold whether pixel intensity changes abruptly
   1. By defining a threshold value

Derivative = Rate of change of a function in respect to a variable

* Represents rate of change at a point on the function
* Usually represented by df/dx

Example of Derivative in Mathematics:

* Function f(x) has two points: (x, f(x)) and (x + ∆x, f(x + ∆x))
* Slope of secant line = average rate of change function f(x) over interval [x, x + ∆x]

Diagram

Description automatically generated

If we make ∆x smaller to the extent that ∆x -> 0, then the second point (x + ∆x, f(x + ∆x)) overlaps the original point (x, f(x)) and the secant line becomes the tangent line (Diagonal becomes Straight)

Slope of tangent line = derivative of function

df/dx = f’(x) = ->

Chart

Description automatically generated with medium confidence

For function f(x), derivative f’(x) represents instantaneous rate of change of f at x

i.e. rate at which f changes the instant x

Text, letter

Description automatically generated

First derivative used to measure instantaneous rate of change of function f(x) at each point x

Minimum distance between pixels is 1, therefore minimum value of ∆x is 1

A picture containing diagram

Description automatically generated

Diagram

Description automatically generated

Central Difference is Forward + Backward

Text, letter

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

1D Image Derivative:

Diagram, table

Description automatically generated

Use convolutions to find df/dx

2D Image Derivative:

Use f(x,y) rather than f(x) and use partial derivatives for

Partial derivative = Using a derivative for a function with multiple variables, with respect to only one variable whilst other variables from the function are constant

Graphical user interface

Description automatically generated

Derivative in X direction = Right Pixel – Left Pixel

Derivative in Y direction = Down Pixel – Up Pixel

Implemented as such:

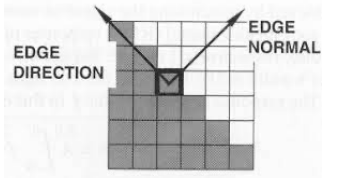
Calendar

Description automatically generated with medium confidence

Image Gradient = ∇f = [ δf/δx , δf/δy ]

* Points to a direction:
* Right = [ δf/δx , 0]
* Down = [ 0, δf/δy ]
* 45° = [ δf/δx , δf/δy ]
* Edge Strength given by Gradient Magnitude = ||∇f ||
  + = ||∇f ||
* Gradient Direction given by θ = tan-1( δf/δx  / δf/δy )

Edge is Perpendicular to the Gradient



Noise corrupts profiles of edges, so we smooth input before edge detection:

* Average filter -> Prewitt Edge Detector
* Gaussian Filter -> Sobel Edge Detector

Prewitt Edge Detector:

1. Original image convolved with average filter like [1, 1, 1]to get smooth image

Diagram

Description automatically generated

1. Diagram

   Description automatically generated with medium confidenceSmooth image convolved with derivative filter like [-1, 0, 1] to get edges

Graphical user interface

Description automatically generated with medium confidence

Sobel Edge Detector:

1. Original image convolved with Gaussian Filter such as [1,2,1] to get a new smooth image
2. Smooth image convolved with derivative filter such as central difference [-1,0,1] to get edges

Graphical user interface

Description automatically generated

Roberts Edge Detector:

A picture containing graphical user interface

Description automatically generated

Thresholding = Pixels with gradient magnitudes greater than or equal to a threshold value are selected as edges

# 6 Advanced Edge Detectors

Canny Edge Detector = Introduced to further enhance edge detection

1. Blur greyscale image (Gaussian)
2. Calculate gradient magnitude and direction (Sobel or Prewitt)
3. Non-maxima suppression

* Also known as edge thinning as it preserves sharpest gradients only
* Image magnitude produces thick edges but final image should have thin edges
* Gradient magnitude of current pixel is compared with magnitude of neighbouring pixels that point in same direction
* Direction rounded to 4 angles: 0, 45, 90, 135

1. Double thresholding

* Algorithm classifies magnitude in three categories: Strong, weak and non-edge pixels
  + Pixels with gradient magnitudes less than low threshold are discarded (set to 0)
  + Pixels with gradient magnitudes higher than high threshold are chosen as strong edges and appear in final result
  + Pixels with gradient magnitudes between both thresholds considered weak edges

1. Edge tracking by hysteresis

* Weak edges either taken to strong or non-edge group
* Weak edges could be meaningful or just caused by noise
* True edges connected to strong edges, whilst noise edges not connected
  + For each weak edge, compare magnitude of pixel being processed with its eight surrounding pixels
  + If at least one surrounding pixel belongs to strong edges group, value of pixel preserved
  + Otherwise discarded

Forward difference used to calculate second derivative

Graphical user interface, text

Description automatically generated with medium confidence

Laplacian Edge Detector: Uses only one filter

* Laplacian L(x,y) of an Image with Pixel Intensity Values f(x,y):

Text, letter

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

* In a single pass, Laplacian Detection performs second order derivatives and hence are sensitive to noise
* To avoid sensitivity, Gaussian smoothing is performed on an image
* First convolve Gaussian Filter with Laplacian Filter then convolve the hybrid filter with image for result
  + Advantage = LoG (Laplacian of Gaussian) kernel can be precalculated in advance so only one convolution needs to be performed on run time on image
  + LoG calculates second spatial derivative of an image:
    - Areas with constant intensity, LoG will be 0
    - At sharp edges between two regions of uniform but different intensities, LoG response will be:
      * Zero at a long distance from the edge
      * Positive just to one side of the edge
      * Negative just to the other side of the edge
      * Zero at some point in between, on the edge itself
* In Image Sharpening, LoG filtered image can be added to original image to make it much sharper and increase contrast

# 7 Image Matching

Image Matching = Establishing Correspondence between a pair of images

* Applications:
  + Face recognition
  + Image retrieval
  + Image stitching (panorama)
  + Object tracking
  + 3D Reconstruction
* Challenges:
  + Illumination Variation
  + Scale Variation
  + Rotation and Perspective Variation
* Holistic = Entire image used to calculate similarity
  + Shows how image overlaps
  + Distance Function used to measure how different two images are
    - Result is 0 for same image and infinite or very high for non-matching images
    - Distance Function takes threshold
    - Assume images are greyscale

1. Create vectors of size HxW for images
2. Resize Images if Images are not greyscale
3. Calculate SSD (Sum of Squared Differences)

Text, schematic

Description automatically generated

* + Does not always work when:
    - Images are shifted
      * Mitigate issue by dividing image into partitions and calculating the histogram of intensity or colour
    - Illumination is changed
  + Feature Descriptor = Representation of an image or image patch that simplifies image by extracting useful information only from image
  + Histogram of Oriented Gradients (HOG) = Feature Descriptor for detecting objects in computer vision and image processing

1. Calculate magnitude and direction of gradient at each pixel in image
2. Divide image into 8x8 cells
3. Calculate Histogram of Gradients in each cell
   * + For each cell a 9 point histogram is calculated, each bin with an angle range of 20 degrees
       - 0, 20, 40, 60, 80, 100, 120, 140, 160
     + Histogram of Gradient Orientation created by quantization
4. Block Normalisation
   * + 4 Cells merged into Blocks in an overlapping manner in strides of one cell
     + 16x16 block has 4 histograms which can be concatenated to form 36x1 vector
     + Normalise histogram

Chart

Description automatically generated with low confidence

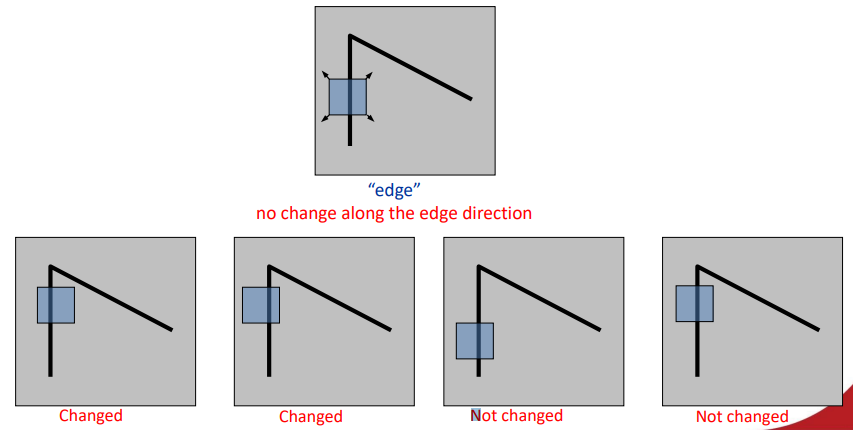
1. Form HOG Feature Vector
   * + Concatenate all 36x1 vectors into one giant vector
   * For colour images:
     + Convert to grey
     + Gradient computed in each colour band and gradient with greatest magnitude at each pixel used to computer HOG Feature

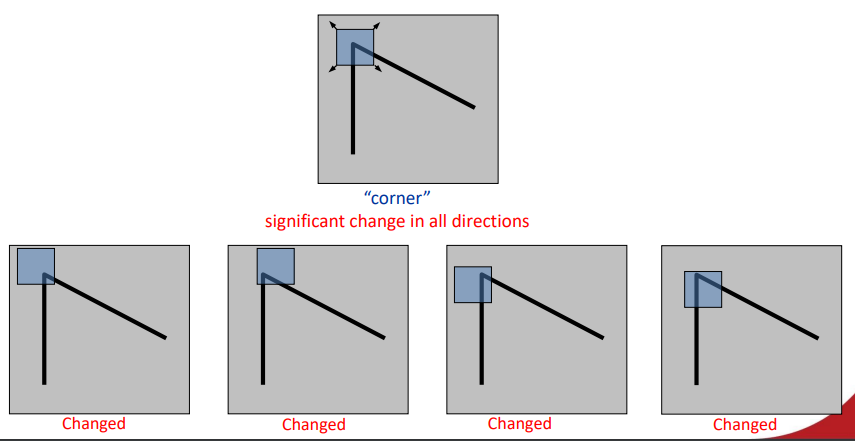
* Local Image Matching:
  + Advantages:
    - Can deal with occlusions
    - Can deal with clutter
      * (a lot of objects in the image and thus it is difficult for an observer to focus their mind on any particular object)
    - More invariant to image transformations
    - More robust to noise
    - Sparse representation of the image
  + Good Local Feature Criteria:
    - Accurate and Repeatable Localisation of Feature Points
    - Invariance to:
      * Translation
      * Rotation
      * Scale
      * Viewpoint
    - Robustness to:
      * Noise
      * Lighting Conditions
      * Compression
      * Blur
    - Distinctiveness of Descriptor

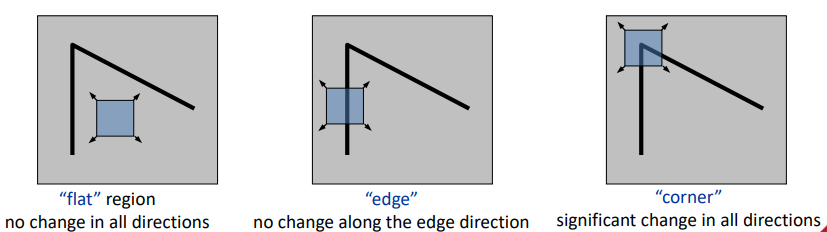
* + Motivation = Patch Matching
    - Elements to be matched are image patches of fixed size

1. Interest Point Detection = Identify interest points (e.g. Harris corner and SIFT)
2. Local Feature Description = Extract feature description surrounding interest point (e.g. HOG and SIFT)
3. Feature Matching = Determine correspondence between descriptors in two views

* + Harris Corner Detector:
    - Intuition:
      * Flat Regions are not distinctive
      * Edge regions are more distinctive than flat regions
      * Corners are even more distinctive than edges
    - Finds points different from neighbours
    - Finds corners by finding points that are different from their neighbourhood by shifting window in all possible direction
    - (u,v) = Offset in x and y
    - Compare each pixel before and after by summing up SSD







* + - Math:

Text

Description automatically generated with medium confidence

* + - Nearly constant patches: E(u,v) = 0
    - Very distinct pathes: E(u,v) will be larger
    - We want patches where E(u,v) is large
    - It is slow to compute for each pixel and each offset (u,v)

# 8

# 9

# 10