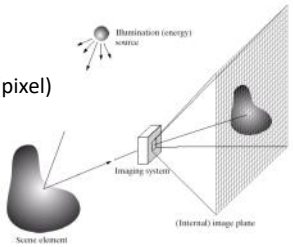
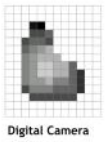
# Digital Image Analysis

Friday, February 2, 2024 8:48 PM

## Basis



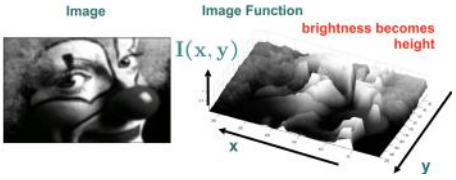

- Digital Images consists of 2D array of values representing brightness of the image of a small region (pixel)
  - Grey level value - values of each pixel
    - Amount of light reflected towards that observer at that position
    - Proportion of the X-rays that are absorbed by the tissue
    - … Other Applications

---

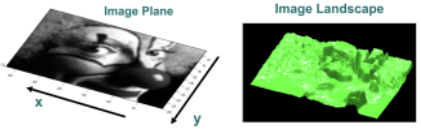
## Image Representation

- Various ways of representing an image depending on the task in hand

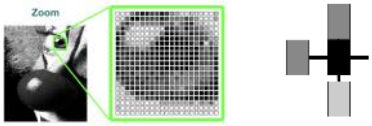
- **Image Function**
  - A (grayscale) image can be thought of as a function $I: \boldsymbol{R^2} \rightarrow \boldsymbol{R}$
  - Where $I(x, y)$ gives the **intensity** at position $(x, y)$
    - The brightness / intensity became the height
  - A **digital** Image is a discrete (**sampled** and **quantized**) version of this function
  - The intensity (0-255) and location (x & y) are the **range** and **domain** of the image function



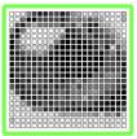

- **Image Landscape**
  - Similar to image function but instead of chopping into discrete values it is more smooth
  - "Throwing" a silk texture on top of the image function
  - Allow use of continuous math and functions

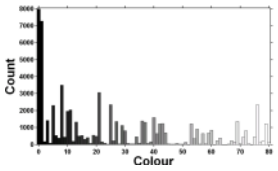



- **Arrays of Pixels/Matrix/Grid of Pixels**
  - Most common
  - Captures values and spatial relationship
    - The neighbouring pixels possibly represents the same object
  - Discrete
  - Grayscale matrices usually use 1 byte to represent intensity
    - 0=black, 255=white




- **Image Histograms**
  - Sorting pixels by grayscale value / colour and stacking them up
  - Captures values but lost spatial relationships
  - Could be use when the content of the image is less important / darker regions are bg
  - Can help improving image / analyzing / spreading
  - Discrete

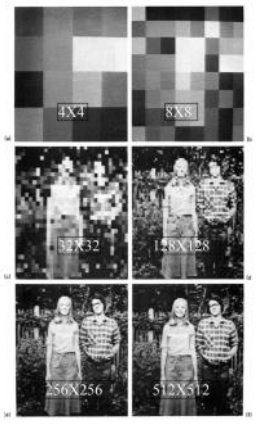





---

## Spatial Resolution vs Gray-level Resolution

Spatial




Gray Level

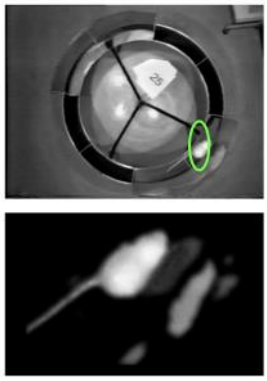

- Spatial resolution depends on
  - Level of detail that can be perceived
    - Scale of structures represented - more pixels = smaller scale
  - Determined by numbers of pixels - more pixels = finer details
    - Digital cameras - 1k x 1k to 2.5k x 2k
    - Medical images - 4k x 4k

- Gray-level resolution depends on
  - Differences in brightness
  - Word length at each pixel
    - Larger word length = finer small changes in brightness can be captured
    - CCD 8-bit (256)
    - Medical images 12-bit (4k)

---

## Gray Image Processing - Image Arithmetic

- As with any function, operators can be applied to an image
- **Image transformation**
  - 
    
  - Convolution (linear filtering) is a special kind of operator

- **Image Addition**
  - Taking two images and adding them
  - Taking average over images in sequence
  - Reduce noise

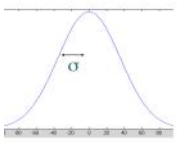
- **Image Subtraction**
  - Taking difference
    - Either take absolute values or shift and scale back to [0:255] for -ve values
  - Detect changes in Static background
  - Object, shadows & reflections in real-world scenes

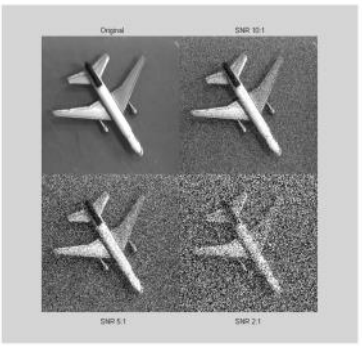
---

## Noise

- Images contain noise even if the world freezes and nothing happens or changes
  - Identical images do not have identical values due to electronics
  - In small scale image (more pixels) - are they fine details or noise?
    - Compare identical images , if the values varies = noise, else = details

$$SNR = \frac{\max signal}{\sigma_{noise}}$$

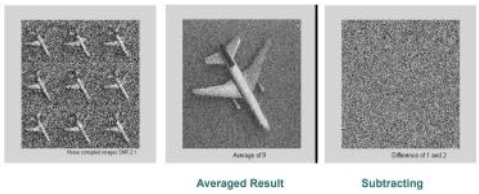


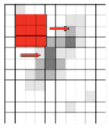
- Noises are modelled as **stochastic** (random) **small fluctuations**

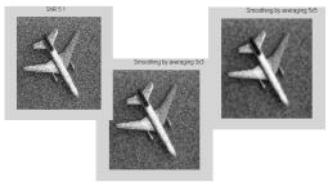
- Signal to Noise Ratio
  - Noise is assumed or expected to be normally distributed
    - Mean=0
    - Amount of noise characterized by standard deviation of the distribution
  - Compares level of signal values to level of noise
    - Higher ratio = better specification (as there is more signal(information) than noise)

- Reducing Noise
  - There are several ways to remove noise
  - **By temporal averaging**
    - Averaging N images - $\sigma_N = \frac{\sigma}{\sqrt{N}}$




  - **By spatial averaging (local averaging / smoothing)**
    - Replace center value with an average in (3x3) neighbourhood
      - $I(p) = \frac{1}{N}\sum_{i}^{n} I(p_i)\ where\ n \in (neighbour(p)_N)$



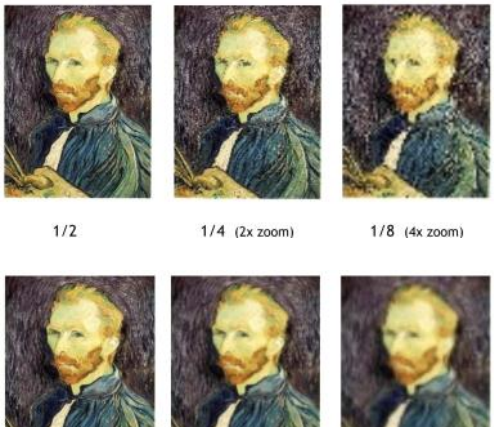


    - As neighbouring pixels are likely to have similar values
    - Some fine details / small scale structures could be loss (blurring) because
      - Assumptions of neighbouring pixels with similar values not being always true (for example in edges)

---

## Scale

- Size of structures of interest
- Small scale structure may be "noise"




- Reducing size of images in situation where details are less important
  - E.g. for algorithms use or storage purposes
  - Sub-sampling can be used
    - Throw away every other row and column to create a 1/2 size image
    - But applying subsampling could make the image look "scruffy"

- Solution - filter first, then sub-sample:
  - Local averaging takes away small scale structures which could be noises
    - But also taking away real small scale structures
  - Now less pixels are needed for the same information

- Solution - filter first, then sub-sample:
  - Local averaging takes away small scale structures which could be noises
    - But also taking away real small scale structures
  - Now less pixels are needed for the same information
  - But of course lower resolution
    - (but details are less interested in these situations)

Gaussian 1/2 G 1/4 G 1/8

- Other motivations to remove smaller scale structure
  - Images can be described by fewer pixels hence more compact
  - Detection of larger scale objects is more reliable as there would be less confusing details
  - Detection can proceed from the highest to lowest scale by adding in detail at each stage - faster and more robust
    - Multi-scale processing
      - Facial detection - blurring the face first allow faster detection
      - Then finer details can be added back

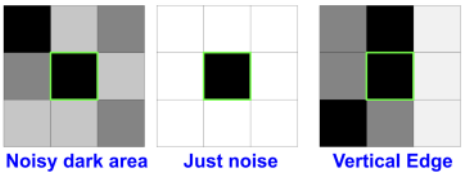
# Neighbourhood Processing

Friday, February 2, 2024 8:52 PM

## Basis

- Neighbourhood processing - considering a single pixel value in context of neighbours
- Neighbourhood is 3x3 in this example

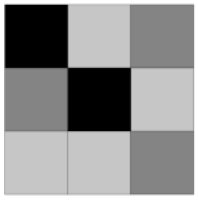
- Two main methods
  - Rank Filtering / Non-Linear (e.g. maximum, minimum, median)
  - Convolution / Linear (result can be represented as linear combination of source)

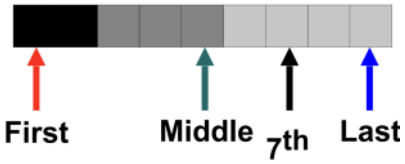


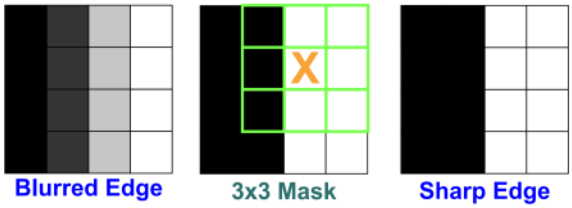

---

## Rank Filtering

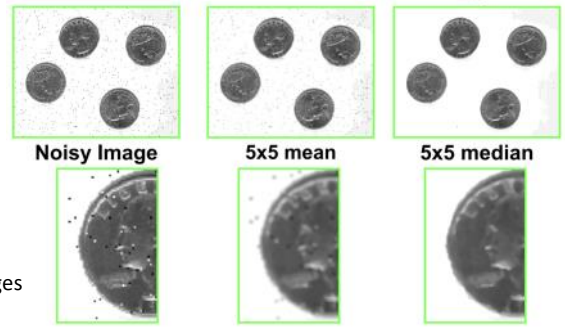
- Take values under filter windows (e.g. 3x3) and sort them
- Output value of the pixel can be:
  - First - minimum / erosion
    - Expands dark spots in this case
  - Last - maximum / dilation
    - Expands light spots in this case
  - Middle value - median filtering
    - Robust to salt and pepper noise
    - Good at removing noise whilst preserving edges
  - Other - e.g. $7^{th}$ / $9^{th}$






- Example - Mean Filtering vs Median Filtering
  - 

  - As the second example showed mean filtering has lighter speckles and softer edges

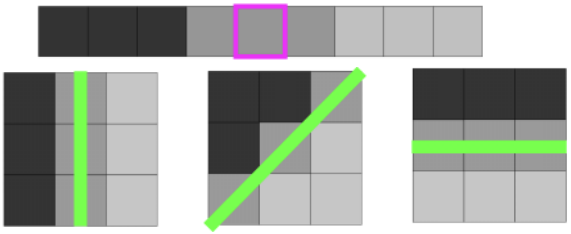



---

## Convolution

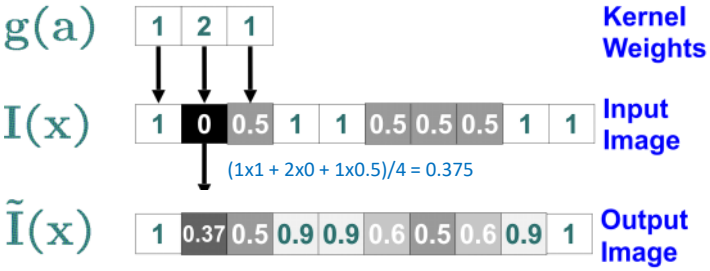
- Transform an input (gray-level image) into some other form suitable for finer analyzes
- Often used as a pre-processing step

- Motivation
  - In rank filtering the spatial relationship is lost as they are sorted by values
  - All three neighbourhoods result in the same set ignoring their pattern



- Convolution adds **weights** to the neighbouring pixels (**weighted mean**) to capture these patterns
- Terms - convolution filter / convolution window / convolution mask / kernel are the same

- 1D example



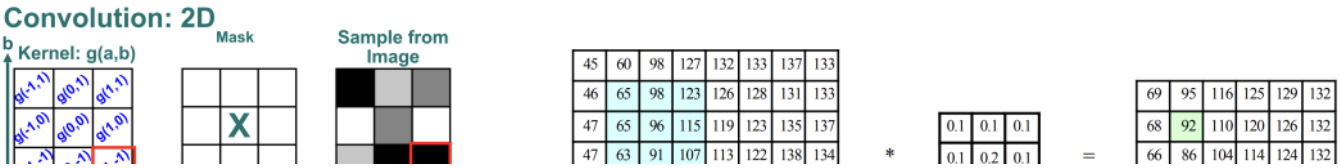

$$\tilde{I}(x) = \frac{\sum_a g(a) I(x+a)}{\sum_b g(b)}$$

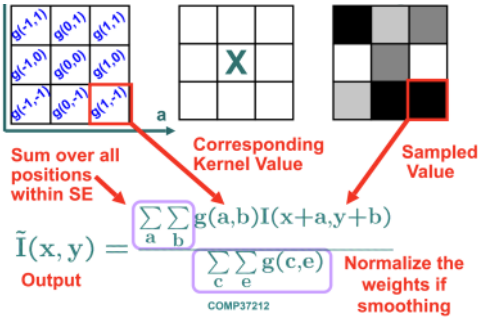
$$\tilde{I} = g * I \qquad \tilde{I} = I * g$$

Useful shorthand, asterisk notation

Where the g(b) is the smoothing

- In 2D



- 


| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 46 | 65 | 98 | 123 | 126 | 128 | 131 | 133 |
| 47 | 65 | 96 | 115 | 119 | 123 | 135 | 137 |
| 47 | 63 | 91 | 107 | 113 | 122 | 138 | 134 |
| 50 | 59 | 80 | 97 | 110 | 123 | 133 | 134 |
| 49 | 53 | 68 | 83 | 97 | 113 | 128 | 133 |
| 50 | 50 | 58 | 70 | 84 | 102 | 116 | 126 |
| 50 | 50 | 52 | 58 | 69 | 86 | 101 | 120 |

*

| | | |
|---|---|---|
| 0.1 | 0.1 | 0.1 |
| 0.1 | 0.2 | 0.1 |
| 0.1 | 0.1 | 0.1 |

=

| | | | | | |
|---|---|---|---|---|---|
| 69 | 95 | 116 | 125 | 129 | 132 |
| 68 | 92 | 110 | 120 | 126 | 132 |
| 66 | 86 | 104 | 114 | 124 | 132 |
| 62 | 78 | 94 | 108 | 120 | 129 |
| 57 | 69 | 83 | 98 | 112 | 124 |
| 53 | 60 | 71 | 85 | 100 | 114 |

- Note that the resulting image is smaller than the original one as values cannot be generated for the pixels at the edges
  - They don't have enough neighbours aka **Padding (border effects)**
- Solutions
  - Zero - expand and set all pixels outside the source image to zero
  - Constant (border colour) - expand and set all pixels outside the source image to a specified border value
  - Mirror - reflect pixels across the image edge (d|b where the | is the edges)

---

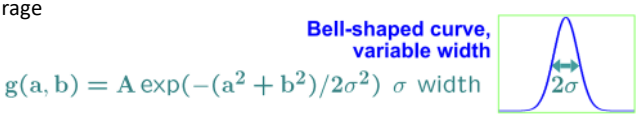
## Smoothing Kernels

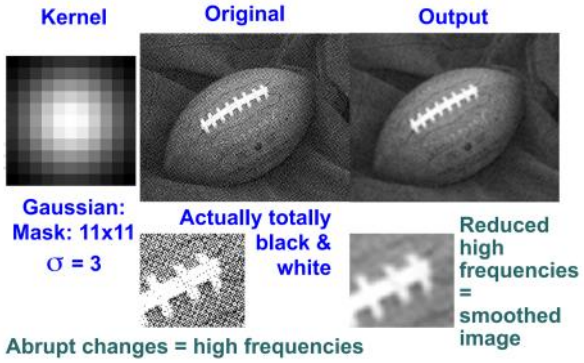
- Smoothing - to remove noise or produce less pixelated images by applying a kernel / filter
- Any single-bump (aka unimodal) kernel will smooth or blur the image
- **Simple weighted mean**

| | | |
|---|---|---|
| 0.5 | 1 | 0.5 |
| 1 | 2 | 1 |
| 0.5 | 1 | 0.5 |

- **Mean Filtering (sometimes referred as Mean Smoothing)**
  - Note that it still smooths / blurs the image (as the average still affects each value)
  - 


- **Gaussian Smoothing**
  - Using Gaussian distribution as the weighted average
  - $\sigma$ indicates how pointy the distribution is
    - Small $\sigma$ = less weight to neighbours
    - Broader $\sigma$ = more weight to neighbours

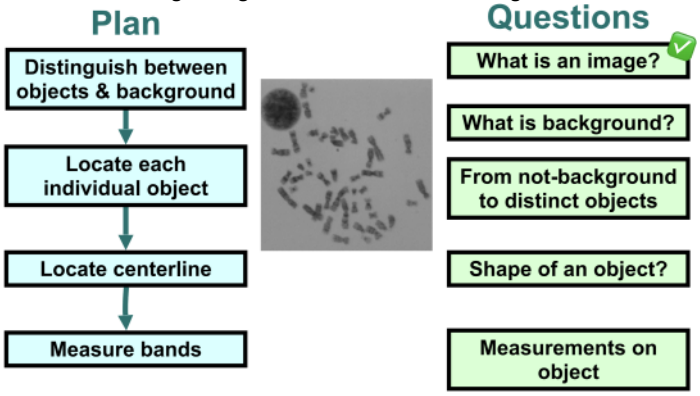



  - Example
  - 

# Thresholding

Friday, February 2, 2024 8:53 PM

## Motivation Problem

- Sample Problem - distinguishing chromosomes from the image
  - 

  - What is an image is answered in the previous lectures
  - Here we tackle the question what is background and from not-background to distinct objects

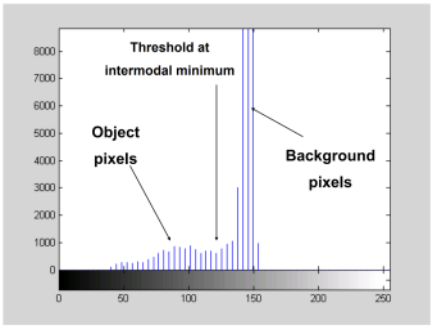
---

## Segmentation - Thresholding

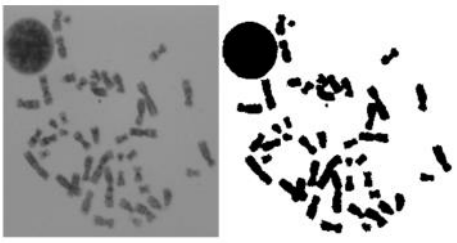
- Partitioning images into parts that are interested and parts that are not
  - In this case the chromosomes and the background
- Task here - label each pixel as either object / background
  - Converts a grayscale image into a binary image
  - Thresholding is one of the simplest segmentation method for **high-contrast images** only
- (Global) Thresholding
  - Set threshold value $T$ such that $b(x,y) = g(x,y) < T$ or $b(x,y) = T_1 < g(x,y) < T_2$
  - Where $b(x,y)$ is a **binary image**

---

## Selecting the Threshold Value




- Thresholds can be selected by many methods
  - Manually - by experience
  - Automated - Otsu
  - Iterative
  - **Histogram analysis**
  - ...
- Histogram analysis - bunch the gray-level values of the image into a histogram
- In this case the histogram is bimodal (two modes / peaks)
  - Intermodal minimum separates the two types of pixels (the two peaks)
- Histogram is the derivative of the area as a function of threshold
  - $H(D) = \frac{dA(D)}{dD}$ (Histogram of grey-levels, D; Area thresholded at grey-level D)
  - Threshold placed where rate of change of segmented area is the **smallest**
    - As the smallest change of rate means that it is stable
    - Thresholding with 100 is more or less same as 101 - probably getting the separation right




- Global thresholding is very sensitive to the threshold value and not as useful for images with **shaded background**
  - As it assumes uniform background

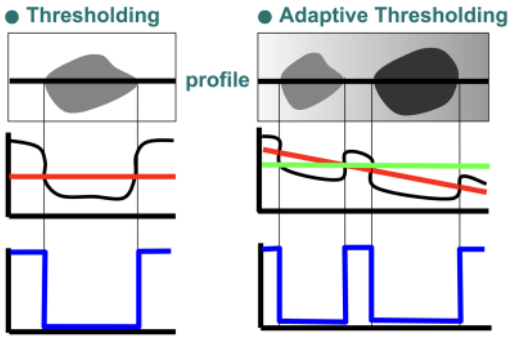
---

## Adaptive (Local) Thresholding

● Thresholding ● Adaptive Thresholding

# Adaptive (Local) Thresholding

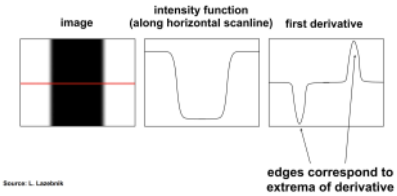
- Applicable for simple shaded background images
  - Instead of using a global threshold value (green line) use a local one (red)
  - able provided that reasonable estimate of background shading can be obtained



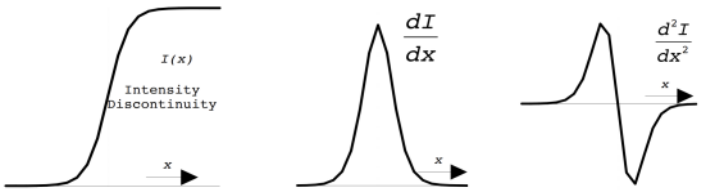
# Edges

Friday, February 2, 2024 8:50 PM

## Basis

- **Edges** - pixels or line regions where the gray level changes rapidly
  - Boundaries between physical distinct regions
  - Discontinuities in intensity values (gray level values)
  - Rapid change in the image intensity function
    - Locating the edges = differentiating the image function
    - 

    - Here the left edge is negative as it changes from white (255) to black (0)

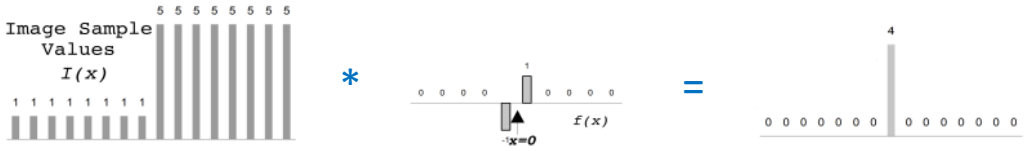


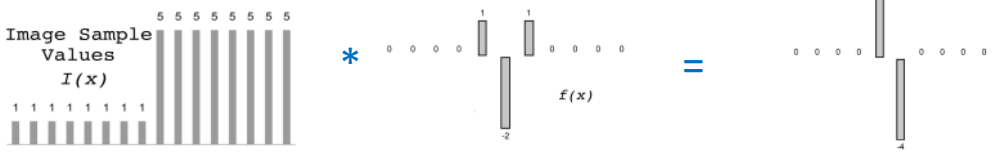

- Images are often conveniently and compactly represented by edges
- Edges occurs in physical property of images changes
  - Boundaries, shadows, different colors/textures, etc.
- Physical boundaries often (but not necessarily) mean anything useful in terms of image processing


---

## Image Filtering - Convolution for Approximating Derivatives in 1D

- Consider the following image and its 1D scanline image function $I(x)$ assuming no noise:


- **First derivative** (aka local gradient) represents the presence of edge / rapid change
  - Apply the kernel $[-1, 0, 1]$
  - 

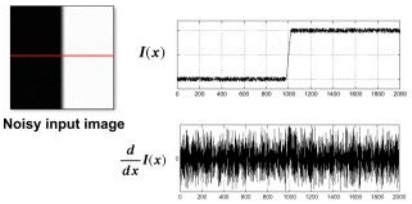

- **Second derivative** (sometimes called Laplacian) represents the direction of the change (left/right side of the edge)
  - Note that the edges is where the second derivative crosses 0
  - Apply the kernel $[1, -2, 1]$
  - 

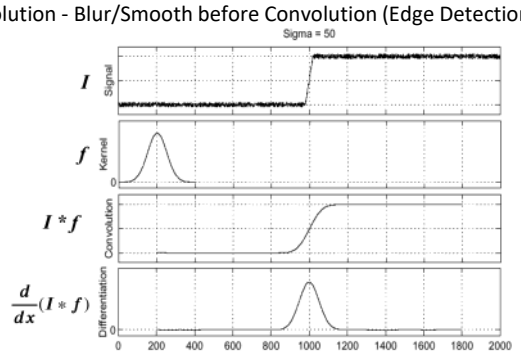
  - Positive to negative - possibly from white to black (hence the left side of the edge)
- **Note that values in differentiation kernels should add up to 0**

---

## Effects of Noise

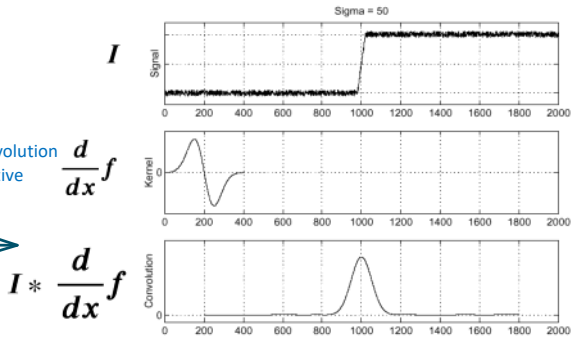
- Noisy images or images with small scale structures gives noisy derivatives

- 


- Solution - Blur/Smooth before Convolution (Edge Detection)
- 


But since differentiation is convolution and convolution is associative

$$\frac{d}{dx}(I * f) = I * \frac{d}{dx} f$$




**To find edges, look for peaks in $\frac{d}{dx}(I * f)$**

  - Note how the kernel $f$ is a smoothing kernel as it is unimodal

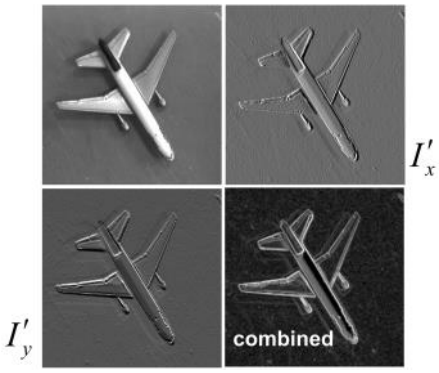
# 2D Edge Detection Convolution

Monday, February 5, 2024 1:50 AM

## Edge Detectors (2D)

- These edge detectors usually have 2 filters that generate transformed gray level images - **the Derivate Images**
  - $I'_x$ - detects (vertical) edges in x-direction while smoothing (integrating) the y-direction
  - $I'_y$ - detects (horizontal) edges in y-direction while smoothing (integrating) the x-direction
- **Edge Magnitude** - indicates how strong (strength) the edges are
  - $|I'| = \sqrt{I'_x{}^2 + I'_y{}^2}$
- **Gradient Direction** (perpendicular to the Edge orientation)
  - $\phi = \tan^{-1}\frac{I'_y}{I'_x}$
- 

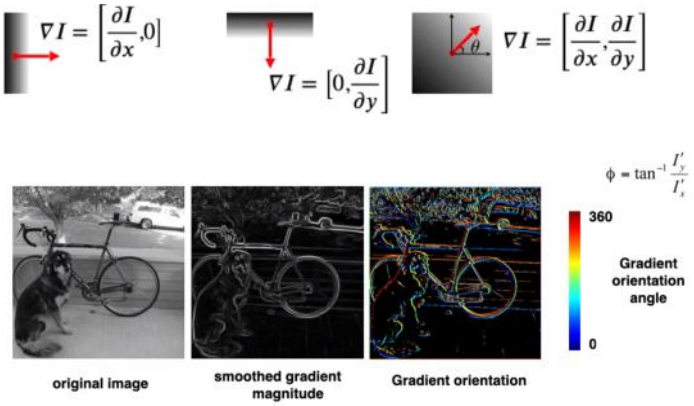
| Edge Detector | **Prewitt** | **Sobel** |
|---|---|---|
| x-differentiating kernel | Edge detecting x-direction (Vertical edges)<br>-1 0 1<br>-1 0 1<br>-1 0 1<br>Smoothing y-direction | Edge detecting x-direction (Vertical edges)<br>-1 0 1<br>-2 0 2<br>-1 0 1<br>Smoothing y-direction |
| y-differentiating kernel | Smoothing x-direction<br>-1 -1 -1<br>0 0 0<br>1 1 1<br>Edge detecting y-direction (Horizontal edges) | Smoothing x-direction<br>-1 -2 -1<br>0 0 0<br>1 2 1<br>Edge detecting y-direction (Horizontal edges) |
| smoothing | Uniform [1,1,1] | Weighted [1,2,1] |
| derivatives | 1st([-1,0,1]) | 1st([-1,0,1]) |




---

## Image Gradient

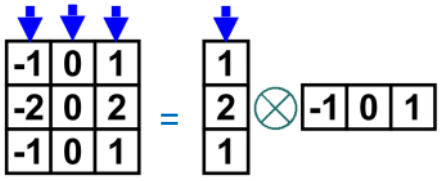
- Gradient points in the **direction** of most rapid increase in intensity
  - $\nabla I = \left[\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y}\right]$
- Edge Strength / Edge Magnitude is given by the **Gradient Strength**
  - $||\nabla I|| = \sqrt{\left(\frac{\delta I}{\delta x}\right)^2 + \left(\frac{\delta I}{\delta y}\right)^2} = \sqrt{I'_x{}^2 + I'_y{}^2} = |I'|$
- **Gradient Direction** is given by
  - $\theta = \tan^{-1}\left(\frac{\delta I}{\delta y} \div \frac{\delta I}{\delta x}\right) = \tan^{-1}\left(I'_y \div I'_x\right) = \phi$
- It is relative instead of absolute - good !




---

## Decomposable Kernels

- **Decomposable Kernels** - kernels that can be broken down into smaller ones (usually in 1D)
  - In terms of a cascade of 1D convolutions
- Prewitt, Sobel and Gaussian are all decomposable - saves computations
- $(a \otimes b) * I \equiv a * (b * I)$
  - $(a \otimes b) * I$ is 1 convolution with $n \times n$ ($n^2$ in total)
  - a $*$ (b $*$ I) is 2 convolutions with $1 \times n$ then $n \times 1$ ($2n$ in total)
- Example - Gaussian (decomposable, symmetric in both 1D and 2D)
  - $g(a,b) = A \cdot \exp\left(-\left(a^2 + b^2\right)/2\sigma^2\right)$



$$= A \cdot \exp\left(\left(-a^2/2\sigma^2\right) + \left(-b^2/2\sigma^2\right)\right)$$
$$= A \cdot \exp\left(-a^2/2\sigma^2\right) \times \exp\left(-b^2/2\sigma^2\right)$$

- Where $\exp\left(-a^2/2\sigma^2\right)$ indicates the pattern between columns
- $\exp\left(-b^2/2\sigma^2\right)$ indicates pattern within a column

# Gaussian

Monday, February 5, 2024 1:50 AM

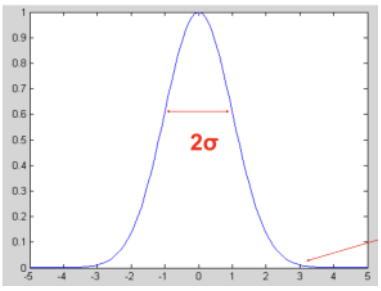
## Selecting Edge Scale

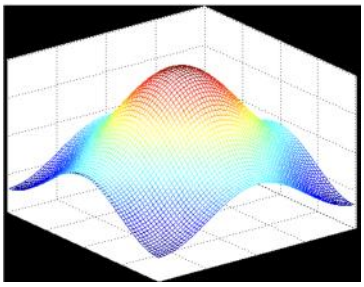
- Fine-scale edges maybe just noises
- Detection of edges at larger scales is more reliable because there is less confusing detail
- Location of edges at coarse scale can direct the search for finer-scale edges
  - Faster and more robust
- In Gaussian this is done by selecting sigma and size of the kernel

---

## Properties of Gaussian

- Aka Normal Distribution
- Symmetric
- Depends on both
  - Sigma - determines spread
    - smaller sigma = pointier peak
  - Size of kernel - determines number of contributed pixels
    - larger kernel = more pixels contributed to the pixel
- Sometimes need to normalize integer mask by
  - Divide by the sum of values on all pixels inside the kernel
- Anti Aliasing - less prune to noises
- Separable (Gaussian kernel is decomposable)
  - $e^{-\frac{x^2+y^2}{2\sigma^2}} = e^{-\frac{x^2}{2\sigma^2}} \times e^{-\frac{y^2}{2\sigma^2}}$
  - Note 1D Gaussian is just the x component of the equation



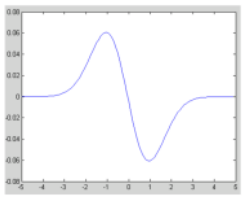



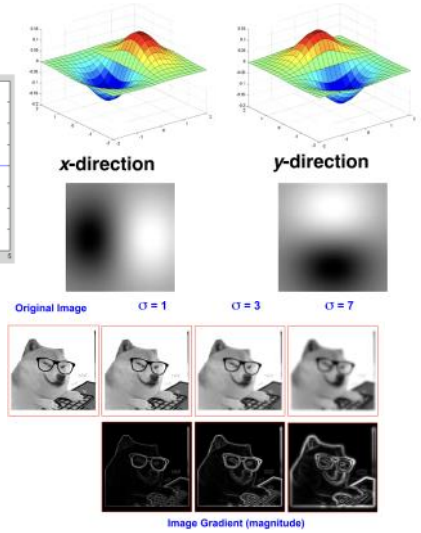
- Differentiable - in edge detection, derivative approximation kernels (e.g. Sobel) no longer necessary

---

## Analytical 1st Derivatives (First part of Canny Operator)

- Canny uses result of Gaussian Smoothing + Edge detection using 1st derivate
  - $G'(x) = -\frac{\sqrt{x^2+y^2}}{\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$
  - Where $\sigma$ determines the **scale**
- Just like the other edge detection kernels it has two differentiation kernels
  - X and Y direction
- In addition Canny also
  - Thinning (suppress non-maxima of derivate)
    - Precise edge locations (sub-pixel precision)
    - Finding local maxima across edges (not along edges) -getting rid of fat edges
  - Track using hysteresis thresholds
    - Select edges that are interested

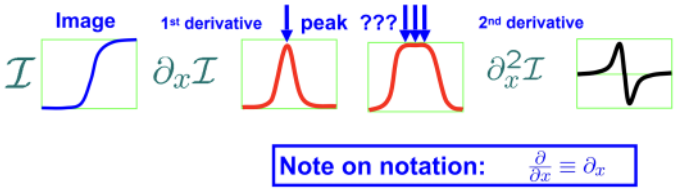





---

## 2nd Derivatives (2D Laplacian)

- Sometimes instead of the 1st derivatives , the 2nd is used to find edges
  - As there could be fat edges in 1st
  - Edges are at the 0 crossing of the 2nd derivatives - more precise
- 2D Laplacian Kernels
  - Negative center and positive surround (or reverse)
  - All values add up to 0
  - Two variations




| 2 | -1 | 2 |
|---|---|---|

| 1 | 1 | 1 |
|---|---|---|

- ~~Negative center and positive surround (or reverse)~~
- All values add up to 0
- Two variations

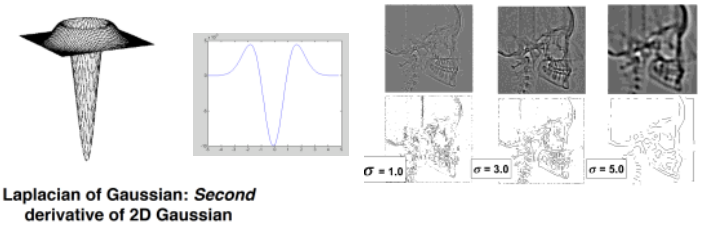
| | | |
|---|---|---|
| 2 | -1 | 2 |
| -1 | -4 | -1 |
| 2 | -1 | 2 |

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | -8 | 1 |
| 1 | 1 | 1 |

- Isotropic - give the same response to edges irrespective of their orientation (unlike in 1st)
  - One filter (instead of x and y direction)
  - No edge direction
  - Indirect edge magnitude (unlike in 1st)

---

## Analytical 2nd Derivatives (Marr-Hildreth)




Laplacian of Gaussian: ***Second*** derivative of 2D Gaussian

- Aka Mexican Hat
- Uses results of Gaussian Smoothing + Laplacian
  - $G''(x) = \frac{1}{\sigma^2}\left(\frac{x^2+y^2}{\sigma^2} - 1\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$
  - Where $\sigma$ again is the scale selection

# Generalized Hough

Monday, February 12, 2024 9:38 PM

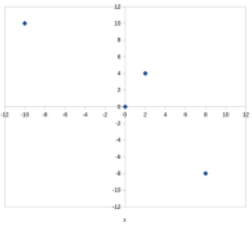
## Basis

- Hough - Feature extraction technique to identify objects within images even if distorted
- Transfer image space to (Hough) **parameter space** which is then quantized into **Accumulator space**
- Uses a voting mechanism to find peeks, which represents the values of the parameters of the line, circles or ellipses

- Why not linear regression (finding line of best fit)?
  - Hough allows finding multiple lines, not just "a line of best fit" among all data points

---

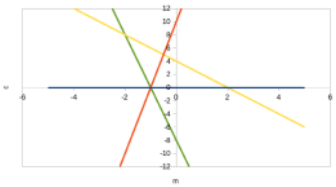
## Hough Lines

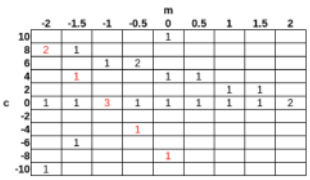


points (0,0), (-10,10), (2,4), (8,-8)

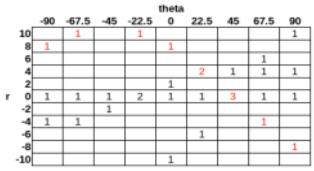
- In image space lines are just segments of pixels
- Computer determines whether a group of pixels represent a line by the Hough Line Transform

- Assume there is a set of points and are to find whose that are on the same line
  - Set a threshold value of pixels (votes) $v$ - how many points are required to represent a line (e.g. 3)
  - **Cartesian**
    - Rearrange the equation of straight lines to make $x$ the gradient and $y$ the $y$-intercept
      - $y = mx + c \Rightarrow c = -xm + y$
    - Substitute the points to the **parameter space equation** to find their possible values
      - $(0,0) \Rightarrow c = -(0)m + 0 = 0$
      - $(-10,10) \Rightarrow c = -(-10)m + 10 = 10m + 10$
      - $(2,4) \Rightarrow c = -(2)m + 4 = -2m + 4$
      - $(8,-8) \Rightarrow c = -(8)m + 8 = -8m + 8$
    - Now draw these lines in parameter space and find the point where $v$ lines intercepts
      - These represent all m and c values possible at the fixed points given
      - at $v = 3 \Rightarrow\ c = 0, m = -1$





    - Or Programmatically, create a 2D array where columns = m and row = c
    - Issue with Cartesian
      - If the line is vertical, then $m = \infty$ and cannot be represented by an array
  - **Polar**
    - Same procedure but in polar coordinates with equations of line
      - $r = x\cos\theta + y\sin\theta$
    - Columns = $\theta$ and rows = $r$



- In OpenCV, polar coordinate is used, requiring parameters $r, \theta, v$

---

## Hough Circles

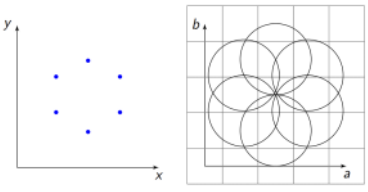
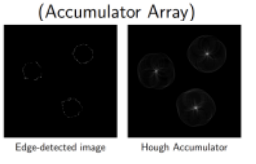


Image space | Parameter space (Accumulator Array)

- Circles with Fixed Radius
  - Given some points, find a circle that goes through all of these points
    - For any image point $(x, y)$, there can be infinitely many circles that go through it
    - $(x-a)^2 + (y-b)^2 = r^2 \xrightarrow{Parameter\ Space} (a-x)^2 + (b-y)^2 = r^2$

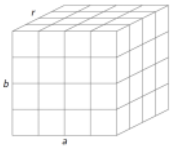
      $C = (a,b) \xrightarrow{Parameter\ Space} C = (x,y)$
  - For each point in the image, a circle is drawn into the accumulator array like Hough Lines
    - Using $(x, y)$ (now the parameters) as the center
    - The center of the circle in the image space could be at any point of the circle in the parameter space
      - The intersection of all the circles in the parameter space is the center of the circle in image space

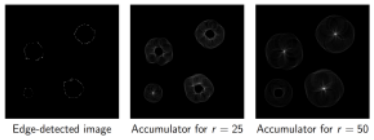


Edge-detected image | Hough Accumulator

Bright locations show the centres of the circles in the image

- Circles with different radii
  - Requires a 3D array instead of 2D for parameters $a, b$ annd $r$



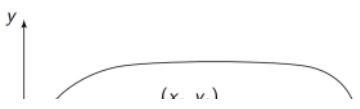


Edge-detected image | Accumulator for $r = 25$ | Accumulator for $r = 50$

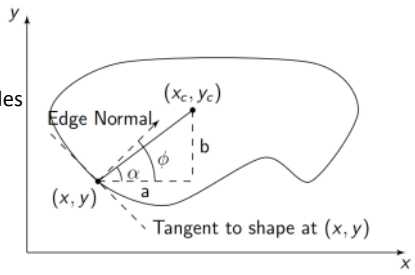
One bright spot can be seen in the $r = 25$ plane and three can be seen in the $r = 50$ plane

---

## Generalized Hough Transformation



# Generalized Hough Transformation




- Generalized Hough Transform uses image **gradient** instead of just edge magnitude in lines and circles
  - With Hough Lines and Circles - mathematical formulas of the shapes are needed
  - With General Hough - description of the shape is needed
    - (which is performed by the algorithm given a shape image)

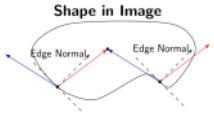
- Finding Shapes with fixed orientation and size given a **template** (example shape):
  1. **Modelling the template shape**
     - Choose a sensible point $(x_c, y_c)$ within the shape as a reference point
     - For any point on the contour (outline of the shape)
       - $x_c = x + a; \quad y_c = y + b \quad \Rightarrow x_c = x + r\cos\alpha\,; \quad y_c = y + r\sin\alpha$
       - $a = r\cos\alpha\,; \quad b = r\sin\alpha$
     - Calculate the normal (gradient) to the image at $(x, y)$
     - If there is a normal where $\phi = \alpha$ , the point $(x, y)$ is a candidate for this template point

$$\begin{array}{ll} \phi_1\text{:} & (r_{1,1}, \alpha_{1,1}),\ (r_{1,2}, \alpha_{1,2}),\ \ldots \\ \phi_2\text{:} & (r_{2,1}, \alpha_{2,1}),\ (r_{2,2}, \alpha_{2,2}),\ \ldots \\ \vdots & \\ \phi_n\text{:} & (r_{n,1}, \alpha_{n,1}),\ (r_{n,2}, \alpha_{n,2}),\ \ldots \end{array}$$

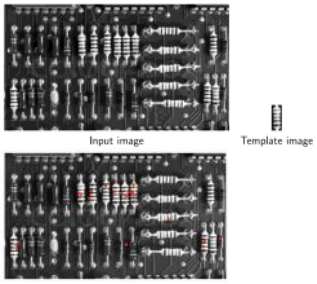
     - R - table consists of a range of edge angles $\phi$(at some resolution, 1° e.g.)
     - For every contour pixel in the shape, $\phi$ is found and $(r, \alpha)$ is calculated for that $\phi$
     - A potential shape center $(x_c, y_c)$ can be calculated using these $(r, \alpha)$ pairs

**Example R-table**

| | |
|---|---|
| 0°: | $(r_{0,1}, \alpha_{0,1}), \ldots$ |
| 1°: | $(r_{1,1}, \alpha_{1,1}), \ldots$ |
| ⋮ | |
| 45°: | $(20, 40°), (18, 140°), \ldots$ |
| ⋮ | |
| 359°: | $(r_{359,1}, \alpha_{359,1}), \ldots$ |




  2. **Shape detection**
     - An accumulator array is created (of some chosen resolution)
     - For each edge pixel $(x, y)$ in the target image, the edge gradient $\phi$ is calculated
     - Each $(r, \alpha)$ pair for this $\phi$ is used to calculate a potential shape center using
       - $x_c = x + r\cos\alpha\,; \quad y_c = y + r\sin\alpha$
     - The item in the accumulator for this $(x_c, y_c)$ is incremented / voted
       - Highest value pair = most likely center
       - Multiple instances of the shape (template) can be detected in an image
       - Separate R-table for each template to be found in an image



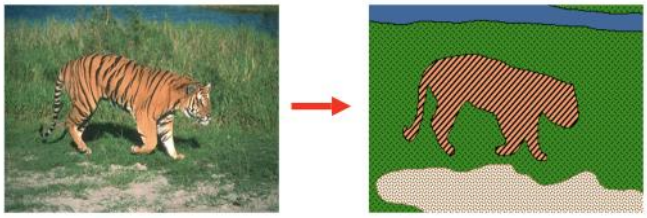

- General Case - find shapes at any scale $s$ and any orientation $\theta$
  - A 4D accumulator array is needed - $A(x_c, y_c, s, \theta)$
  - $x_c = x - s(a\cos\theta - b\sin\theta); \quad y_c = y - s(a\sin\theta + b\cos\theta)$

# Grouping

Monday, February 12, 2024 9:38 PM

## Basis

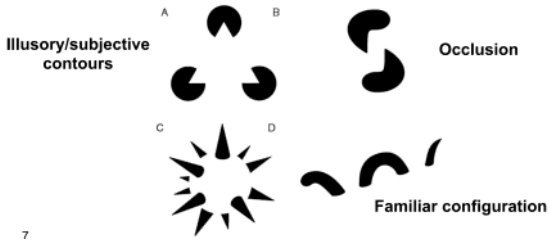
- Goal - gathering features that "belong together"
  - But there could be more than one correct answer



- Obtain an intermediate representation that compactly describes key parts
- Usually (but not necessarily) an intermediate step
  - Determining image regions
  - Grouping video frames into shorts
  - Figure-ground
  - Image Segmentation
  - …Snapchat sticker

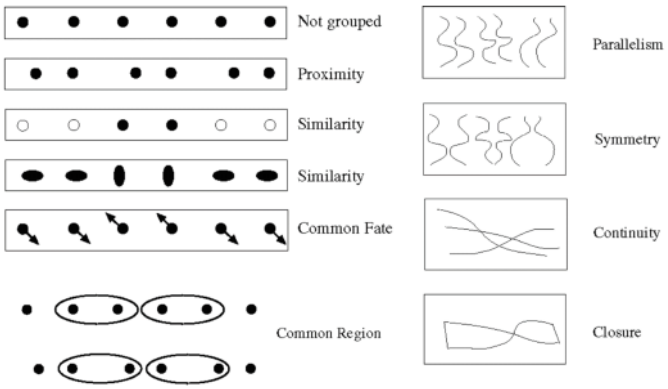
---

## The Gestalt School




- Grouping is key to visual perception
- Elements in a collection can have properties that result from relationships

- Gestalt Factors




- Whole or Group
  - "Whole is greater than sum of its part"
  - Relationships among parts can yield new properties/features

- Psychologists identified series of factors that predispose set of elements to be grouped (by the human version system)

---

## Grouping in Computer Vision

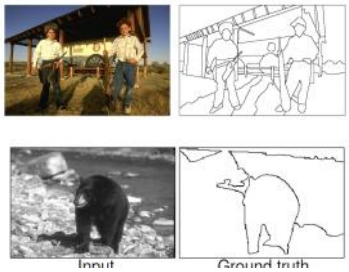
- Some of the factors are mapped to algorithms
  - Similarities, Proximities, Common Fate (direction of gradients)

- Segmentation methods
  - Top down: pixels belong together because they are from the same object
  - Bottom up: pixels belong together because they look "similar"

- Success is hard to measure as it depends on the application

---

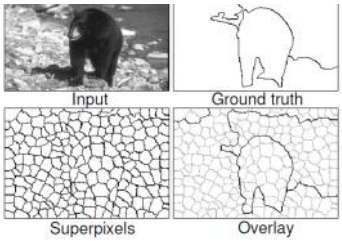
## "Good" Segmentation ?




- First idea - compare to human segmentation or to "ground truth"
  - E.g. Berkeley Segmentation dataset
  - However there is no object definition - different answers even between human

- Second idea - Superpixels
  - Ignore the notion of "correctness"

- Second idea - Superpixels
  - Ignore the notion of "correctness"
  - Content with an "over-segmentation" - each region is very likely to be uniform
  - Group together similar-looking pixels for efficiency of further processing

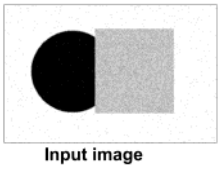


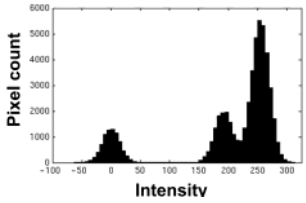
# Clustering

Tuesday, February 20, 2024 2:29 PM

## Basis

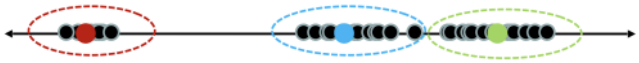
- Allows quantitative metrics to measure "correctness"
- Treat pixels as data in machine learning
  - Less interested (or ignore) pixel layout and neighborhood (spatial cues)



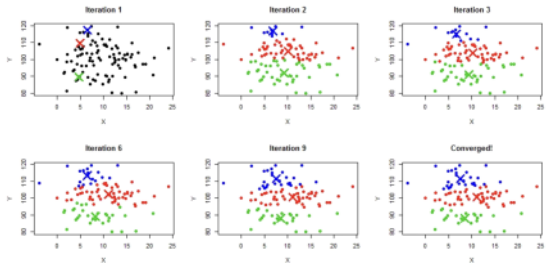

- Intensity values can be used for segmentation
  - However practically there is often **noise** (random variations) in the image
  - Clustering is needed to determine the intensities that define the groups
    - **Unsupervised learning** - detecting patterns with unlabeled data

- Goal - choose three "**centers**" as representative intensities and label every pixel according to the nearest center
  - **Minimize Square-Sum-Difference (SSD)** between all points and the nearest cluster center
  - "**Chicken and Egg**" problem:
    - Need the *cluster centers* to allocate groups
    - Need to know *group membership* to get cluster centers



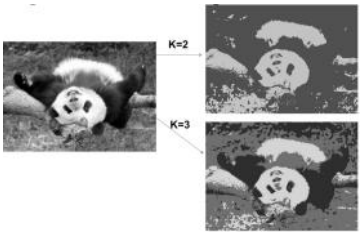
---

## K-means




- Break the "Chicken and Egg" problem by **randomly** assigns cluster centers
  1. Randomly initialize the cluster centers, $c_1, c_2, \dots c_k$ for clusters $\{1,2, \dots k\}$
  2. Given the cluster centers, determine points in each cluster
     i. For each point $p$, find the closest $c_i$ and allocate $p$ to cluster $i$
  3. Given points in each cluster, calculate the new $c_i$
     i. Set $c_i$ to be the mean of points in cluster $i$
  4. If $c_i$ is changed, repeat step (2)

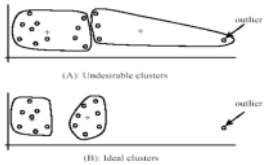
- Properties
  - Always converge to some solution (but not necessarily optimal)
  - Can be a Local minimum - not always find global minimum of objective function
    - $\sum_{i}^{k} \sum_{p \in cluster_i}^{len(cluster_i)} \|p - c_i\|^2$
  - Dependent on initial random allocation - different local minimum every time
  - Each pixel is treated equally / contributes equally



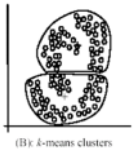

- K-means clustering based on intensity / colour is essentially **vector quantization** of the image attributes
  - Clusters do **not** have to be <u>spatially coherent</u>
    - But can be enforced more by using $(r, g, b, x, y)$

- Pros:
  - Simple and quick to compute
  - Converges to local minimum of within-cluster squared error

- Cons:
  - Need to select value of k (through parameter tuning etc.)
  - Sensitive to initial centers
  - Sensitive to outliers
  - Detects spherical clusters only
  - Every pixel can only be allocated to exactly 1 cluster - too "harsh"



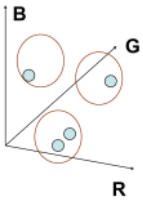


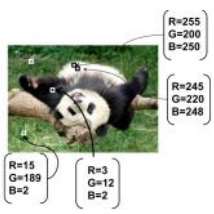



---

## Feature Space

- In the previous example, intensity value (1D - gray level) is the **feature space**
  - The decision of feature space allows different ways of clustering pixels
  - Other feature space can be colour value (3D - RGB)



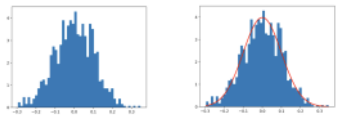
# Probabilistic Clustering

Tuesday, February 20, 2024 2:29 PM

## Basis

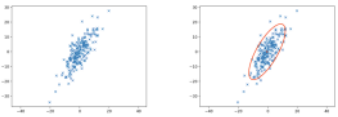
- Allows clusters that are overlapping and non-circular shape
- Include notion of uncertainty

- Gaussian Mixture Models
  - Clusters modeled as Gaussians not just their means
  - EM algorithm - assign data to cluster with some probability
  - Gives **probability model** of x ("generative")
    - Assume data are generated by sampling a continuous function
      - (**generative model / parametric model** Instead of points)
    - The generative model is defined by a vector of parameters $\boldsymbol{\theta}$



$\mathcal{N}(x; \mu, \sigma)$

Gaussian (normal) distribution gives probability model of $x$ ("**generative**")



$\mathcal{N}(x; \mu, \Sigma)$

---

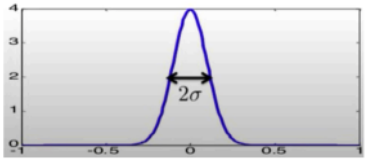
## Univariate vs Multivariate Normal (Gaussian) Distribution

- **Univariate (1D)**
  - Describes a single continuous variable (e.g. intensity)
  - $\mathcal{N}(x; \mu, \sigma) = Pr(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp[-0.5(x-\mu)^2/\sigma^2]$
    - Where it takes 2 parameters
      - $\mu$ is the mean
      - $\sigma^2 > 0$ is the variance
  - Default mean = 0 and $3\sigma$ covers all data

**Maximum Likelihood estimates**

$$\hat{\mu} = \frac{1}{N}\sum_i x^{(i)}$$

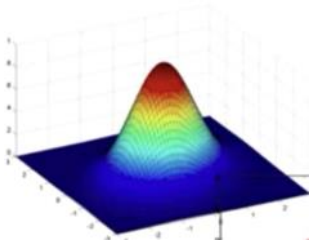
$$\hat{\sigma}^2 = \frac{1}{N}\sum_i (x^{(i)} - \hat{\mu})^2$$



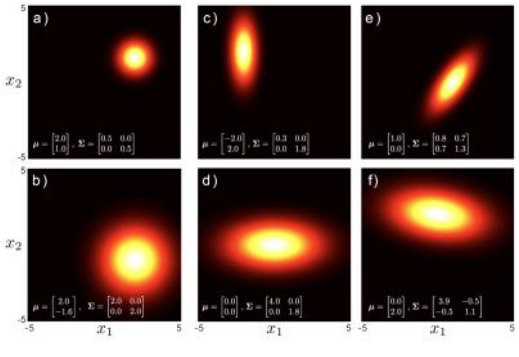

- **Multivariate**
  - Describes multiple (d) continuous variables (e.g. RGB space)
  - $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}}} |\Sigma|^{-\frac{1}{2}} \exp\{-0.5(\boldsymbol{x}-\boldsymbol{\mu})\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})^T\}$
    - Where it takes 2 parameters
      - $\boldsymbol{\mu}$ is a length-d row vector containing mean position
      - $\Sigma$ is a d × d symmetric "positive definite" convariance matrix
        - $|\Sigma|$ is the matrix determinate

$$\hat{\mu} = \frac{1}{m}\sum_j \boldsymbol{x}^{(j)}$$

$$\hat{\Sigma} = \frac{1}{m}\sum_j \left(\boldsymbol{x}^j - \hat{\mu}\right)^T \left(\boldsymbol{x}^j - \hat{\mu}\right)$$



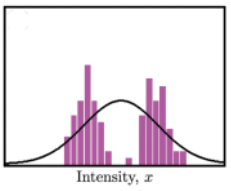
- Types of covariance
  - Spherical - $\Sigma_{\text{spher}} = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$
    - Positive multiple of the identity matrix
  - Diagonal - $\Sigma_{\text{diag}} = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$
    - Where $\sigma_1$ and $\sigma_2$ are different values
    - Direction of the ellipses is parallel to the $x, y$ axis
      - Meaning that the variance can be measured
  - Full - $\Sigma_{\text{full}} = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 \end{bmatrix}$
    - Tends to be symmetric : $\sigma_{12}^2 = \sigma_{21}^2$
  - When covariance is spherical / diagonal - variables are independent

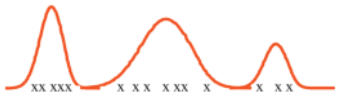



---

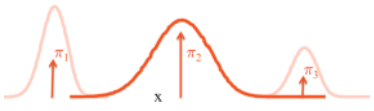
## Gaussian Mixture Model

- Single Gaussian Model cannot fit complex data
  - Ideally each cluster is modelled by one Gaussian Model

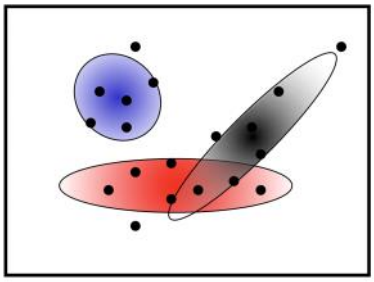


- Gaussian Mixture Models
  - Start with parameters describe each cluster
    - $\mu_c,\ \sigma_c,\ \pi_c$ (size or weight - how many samples)
    - Probability Distribution $p(x) = \sum_c \pi_c \mathcal{N}(x; \mu_c, \sigma_c)$
    - Select a mixture component (a Gaussian dist) with probability $\pi$
      - $p(z = c) = \pi_c$
      - More weight = higher chance to be selected
    - Sample from that component's Gaussian
      - $p(x|z = c) = \mathcal{N}(x; \mu_c, \sigma_c)$



        - More weight = higher chance to be selected
    - Sample from that component's Gaussian
        - $p(x|z=c) = \mathcal{N}(x; \mu_c, \sigma_c)$



- Mixture of Gaussians (MoG) is a generative model
    - K Gaussian blobs in total
    - Each blob $b$ is modelled by means $\boldsymbol{\mu_b}$ covariance matrices $V_b$ in dimension $d$ (feature space)
        - $b$ is defined by $P(x|\mu_b, V_b) = \frac{1}{\sqrt{(2\pi)^d |V_b|}} \exp\left(-\frac{1}{2}(x-\mu_b)^T V_b^{-1}(x-\mu_b)\right)$
    - Blob b is selected with probability $\alpha_b$ (or mixture weights)
        - The likelihood of observing x is a weighted mixture of Gaussians
        - $P(x|\theta) = \sum_{b=1}^{K} \alpha_b P(x|\theta_b)\,, \qquad \theta = \{\mu_1, \dots \mu_K, \qquad V_1, \dots, V_K\}$

---

## Expectation Maximization (EM) Algorithm

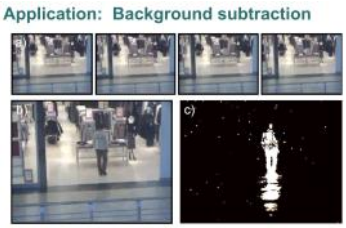
- Goal - Find blob parameters $\boldsymbol{\theta}$ that maximize the likelihood function $P(data|\theta) = \prod_x P(x|\theta)$

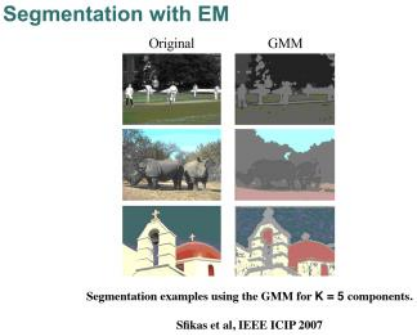
- Approach:
    1. E-step: Given current guess of blobs, compute ownership (responsibility) of each point
        - Compute the probability that point $x$ is in blob $b$ given current guess of $\boldsymbol{\theta}$
            - $P(b|x, \mu_b, V_b) = \frac{\alpha_b P(x|\mu_b, V_b)}{\sum_{i=1}^{K} \alpha_i P(x|\mu_i, V_i)}$
    2. M-step: Given ownership probabilities, update blobs to maximize likelihood function
        - Compute the probability that blob $b$ is selected
            - $\alpha_b^{new} = \frac{1}{N} \sum_{i=1}^{N} P(b|x_i, \mu_b, V_b)$
        - Update new mean of blob $b$ with $N$ data points
            - $\mu_b^{new} = \frac{\sum_{i=1}^{N} x_i P(b|x_i, \mu_b, V_b)}{\sum_{i=1}^{N} P(b|x_i, \mu_b, V_b)}$
        - Update the covariance of blob $b$
            - $V_b^{new} = \frac{\sum_{i=1}^{N} (x_i - \mu_b^{new})(x_i - \mu_b^{new})^T P(b|x_i, \mu_b, V_b)}{\sum_{i=1}^{N} P(b|x_i, \mu_b, V_b)}$
    3. Repeat until convergence

---

## Applications of MoG (GMM) and EM

- Some Common applications
    - Any clustering problem
    - Any model estimation problem
    - Missing data problems
    - Finding outliers
    - Segmentation problems




- Pros:
    - Probabilistic interpretation
    - Soft assignments between data points and clusters - allow overlaps
    - Generative model - allow prediction of novel data points
    - Relatively compact storage - only need to store the distribution (which is defined by the parameters) instead of all data points

- Cons:
    - Local minima - might not be optimal
    - Initialization effects final result - often a good idea to start with some k-means iterations
    - Need to know number of components (value of K)
    - Need to choose a generative model (Gaussian might not be optimal for all cases)
    - Numerical problems are often nuisance

# Model-Free Clustering

Tuesday, February 20, 2024 2:31 PM

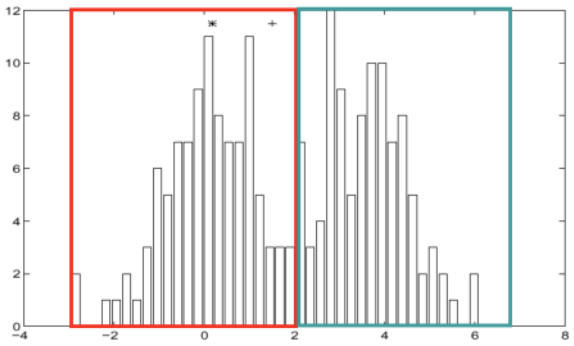
## Basis

- K-means and MoG model feature vectors from an unknown probability density function and try to find clusters or modes
- Parametric models made model assumptions about :
  - The shape (Gaussian)
  - Number of clusters (K)
  - The density function is a superposition of a small number of similar distributions
  - Covariance and shape of the data can be estimated

- Non-parametric model does not have to made these assumptions
  - Smooth the distribution and find the peaks without actually needing to completely compute the function explicitly

---

## Mean-Shift Algorithm

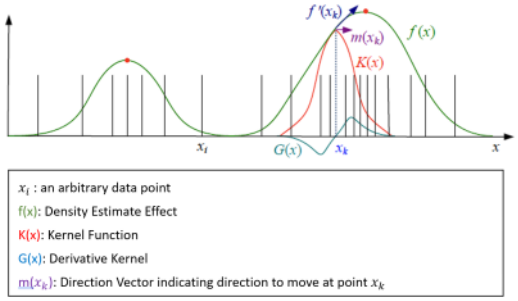


- An advanced and versatile technique for clustering-based segmentation

- Mode = local maximum of the density of a given distribution
  - Easy to spot but hard to compute
    - The point where gradient = 0 and value=max

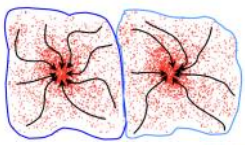
- Mean-Shift Algorithm - Iterative Mode Search
  1. Initialize random seed and window (or kernel) W
     - The random seed is the "estimated" center at W sampled within W
  2. Calculate center of gravity (the "mean") of W:
     - $\sum_{x \in W} xH(x)$
  3. Shift the search window to the "mean"
  4. Repeat Step 2 until convergence

- General Idea:
  - Start with a guess at a local maximum point $x_k$ where
    - $x_k$ is one of the random input data from the sample set
  - Estimate the density estimate or function effects $f(x)$
    - By convolving a kernel function $K(x)$ around point $x_k$
  - Compute the gradient $f'(x)$ by convolving with a derivative kernel $G(x)$
    - Which results in a mean-shift vector $m(x_k)$ that tells the direction of the shift
  - Update the guess by moving $x_k$ in the direction given
  - Repeat until convergence

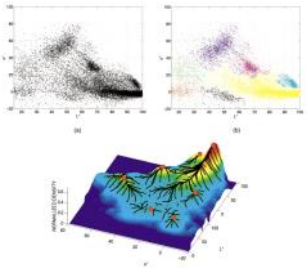



---

## Mean-Shift Clustering



- In Mean-Shift Clustering:
  - Cluster - all data points in the attraction basin of a mode
  - Attraction basin - the region for which all trajectories lead to the same mode



D. Comaniciu and P. Meer, 2002

1. Find features (colour, gradients, texture, etc)
2. Initialize windows at individual pixel locations
3. Perform mean-shift for each window until convergence
   - This can be done concurrently
4. Merge windows that end up near the same "peak" or mode

---

## Properties

- Pros:
  - General and application-independent
  - Model-free - no assumptions about any prior shape on data clusters (spherical, elliptical, etc)
  - Just one single parameter (window size h)
  - Finds varied number of modes
  - Potential global optimal depending on window size
  - Robust to outlier

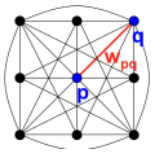
- Cons:
  - Output depends heavily on window size
  - Window size (bandwidth) selection is not trivial
  - Computationally (relatively) expensive (~2s/image)
  - Does not scale very well with dimension of feature space (as m(x) points to all d directions)

# Graph-Theoretic Segmentation

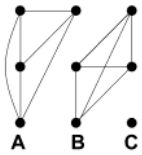
Thursday, February 22, 2024 6:21 AM

## Basis

- View Segmentation as a problem of cutting graphs into pieces
- Images are represented as **Fully-Connected Graph**
  - Node(vertex) for every pixel
  - Link (edge) between every pair of pixels $(p, q)$
  - Affinity weight (cost) $w_{pq}$ for each edge
    - $w_{pq}$ measures similarity
    - Similarity is **inversely proportional** to difference



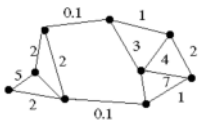

- Break Graph into Segments
  - Delete links that cross between segments
  - Easiest to break links that have low similarities (low weight)
    - Similar pixels should be in the same segment
    - Dissimilar pixels should be in different segment

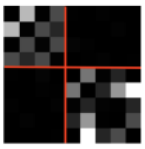


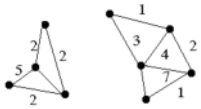

---

## Affinity Matrices

- The $(i, j)$ element in the matrix represents weight of the edge from vertex $i$ to vertex $j$
  - $p \times p$ matrixes
- For undirected graphs, symmetric matrices are used
  - Place half the weight in each $(i, j)$ and $(j, i)$ element
- Larger values = higher intensity



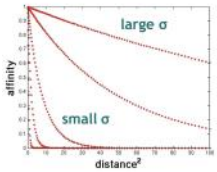



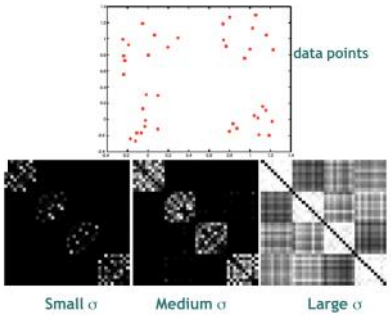


Here, the cut is nicely defined by the block-diagonal structure of the affinity matrix.

- Matrices can be shuffled by associating vertices with rows and columns in a different order

- Affinity Measures include
  - Distance - $\text{aff}(x, y) = \exp\left\{-\frac{1}{2\sigma_d^2}\|x - y\|^2\right\}$
  - Intensity - $\text{aff}(x, y) = \exp\left\{-\frac{1}{2\sigma_d^2}\|I(x) - I(y)\|^2\right\}$
  - Colour - $\text{aff}(x, y) = \exp\left\{-\frac{1}{2\sigma_d^2}\, dist\left(c(x), c(y)\right)^2\right\}$
  - Where $\sigma_d$ is a parameter (scale)
    - Large when distant points need to be grouped
    - Small if nearby points need to be grouped

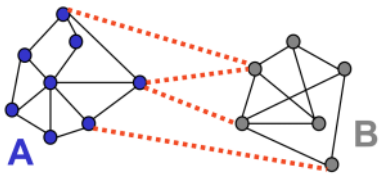


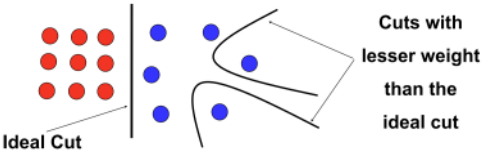

---

## Graph Cutting

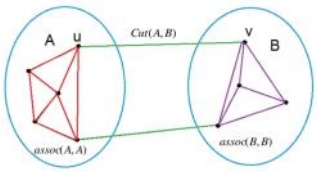
- **Graph Cut** - Set of edges whose removal makes a graph disconnected hence a segmentation
- Cost of a cut = sum of weights of cut edges
  - $cut(A, B) = \sum_{p \in A, q \in B} w_{p,q}$




- **Minimal Cut**
  - Cuts with minimum cost
  - Efficient algorithms exist for doing this
  - But minimal cut is not optimal
    - Weight of cut proportional to number of edges in the cut
    - Minimum cut tends to cut off very small isolated components




- **Normalized Cut (NCut)**
  - Normalize the size of segments and use minimal cut
  - Normalized cut cost - $Ncut(A, B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)}$
    - Where $assoc(A, A) = \sum_{p \in A, q \in A} w_{p,q}$ is the association (sum of all weights) within a cluster
    - $assoc(A, V) = assoc(A, A) + cut(A, B)$ is the sum of all the weights associated with nodes in A
    - Intuition - big segments will have a larger $assoc(A, V)$ thus decreasing $Ncut(A, B)$
  - Finding the globally optimal cut is NP-complete
    - But a relaxed version can be solved using a generalized eigenvalue problem




$$Ncut(A,B) = \frac{green}{red + green} + \frac{green}{purple + green}$$

- Pros:
  - Generic framework - flexible to choice of function that computes weights between nodes
  - Reliable as there is no estimation made
  - Does not require any model of the data distribution
- Cons:
  - Time and memory complexity can be high (slowest among all other methods)
    - Dense highly connected graphs = many affinity computations
    - Solving eigenvalue problem

---

## GrabCut


- Uses user input to choose an region of interest (ROI)
- Immediately assumes pixels outside the box as background
- Iteratively improve assumptions of background and foreground

- Iterate between two steps
  1. Segmentation using graph cuts
     - Requires having foreground model (GMM)
  2. Foreground-background modelling using unsupervised clustering
     - Requires having segmentation

- An optimization algorithm that finds a global optimal solution
- Binary segmentation (either background or foreground)

---

## Improving Efficiency of Segmentation

- Problem - Images contain many pixel
  - Even with efficient graph cuts

- Efficiency trick - Superpixels
  - Group together similar-looking pixels for efficiency of further processing
  - Cheap, local over-segmentation

- Several different approaches possible but important to ensure that superpixels
  - Do not cross boundaries
  - Have similar size
  - Have regular shapes
  - Algorithm has low complexity

---

## Evaluate Segmentation

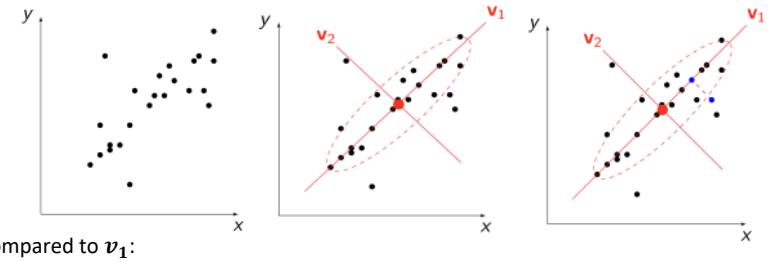

- $F = \frac{2PR}{P+R}$ where
  - $P$ = Precision
    - The percentage of the marked boundary points that are real ones
    - i.e. $\frac{TP}{TP+FP}$
  - $R$ = Recall
    - The percentage of the real boundary points that were marked
    - i.e. $\frac{TP}{TP+FN}$

# Principal Component Analysis (PCA)

Friday, February 23, 2024 2:38 PM

## Dimensionality Reduction (2D)

- Consider a set of data (24 samples) we can draw
  - Line of best fit $\boldsymbol{v_1}$ (largest variation in values)
  - Smallest variation in values $\boldsymbol{v_2}$
    - Note that $\boldsymbol{v_1} \perp \boldsymbol{v_2}$
  - Mean point
  - Region (ellipse) that contains most data




- If the variation in the direction of $\boldsymbol{v_2}$ is quite small compared to $\boldsymbol{v_1}$:
  - All data points can be (approximated) projected to $\boldsymbol{v_1}$ - only need to save $\boldsymbol{v_1}$ instead of all points
  - Each point can be represented by a single value (distance along $\boldsymbol{v_1}$) instead of two
  - Note that the narrower the ellipse is, the more accurate the approximation is

- Data points with four variables can be visualized in 3D or 2D

---

## PCA Algorithm

1. Assemble the data into a matrix (size = #samples ($S$) × #variables ($N$))
   - In the above example - 24 points × 2 variables

| x | y |
|---|---|
| 1 | 1 |
| 3 | 0 |
| -1 | -1 |

This data consists of three data points of two variables

So, covariance matrix will be size 2 × 2

2. Compute the Covariance Matrix $C$
   - $\text{cov}(x, y) = \mathbb{E}[\text{xy}] - \mathbb{E}[\text{x}]\mathbb{E}[\text{y}]$
     - Where $\mathbb{E}[]$ is the expected value
     - $\mathbb{E}[\text{x}, \text{y}] = \frac{1}{S}\sum_{j=1}^{N} x_j y_j$
   - In openCV : `calcCovarMatrix(samples, output, mean)`
   - In numpy: `np.cov(sample)`

| | x | y |
|---|---|---|
| x | $\text{cov}(x,x) = \text{var}(x)$ | $\text{cov}(x,y)$ |
| y | $\text{cov}(y,x)$ | $\text{cov}(y,y) = \text{var}(y)$ |

$\text{cov}(x, x) = \text{var}(x)$
and $\text{cov}(x, y) = \text{cov}(y, x)$
So, **Cov. Matrix is symmetrical**

Plugging in the values:
$\text{cov}(x,x) = 8/3$
$\text{cov}(x,y) = 2/3$
$\text{cov}(y,x) = \text{cov}(x,y) = 2/3$
$\text{cov}(y,y) = 2/3$

gives the covariance matrix:
$\mathbf{C} = \begin{bmatrix} 8/3 & 2/3 \\ 2/3 & 2/3 \end{bmatrix}$

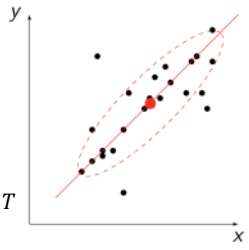
3. Find Eigenvalues $\lambda_i$ and Eigenvectors $\boldsymbol{v_i}$ of $C$ where $i \in \{1, .., N\}$
   - Eigenvalues $\lambda$ and Eigenvectors $\boldsymbol{v}$ satisfy
     - $C\boldsymbol{v} = \lambda\boldsymbol{v}$
     - To find eigenvalues, solve
       - $|C - \lambda \boldsymbol{I}| = 0$
       - Where $|\square|$ is the determinant $(ad - bc)$
       - I is the identity matrix
     - To find eigenvectors, use the equation $C\boldsymbol{v} = \lambda\boldsymbol{v}$
     - In python:
     - `From numpy import linalg as la`
       `vals, vecs = la.eig(C)`
   - Each $\lambda_i$ gives the variance in the direction $\boldsymbol{v_i}$
   - Total variance $T = \sum_{i=1}^{N} \lambda_i$

Using the data from the 24 exam scores data, above:

The covariance matrix, $\mathbf{C} = \begin{bmatrix} 0.70 & 0.50 \\ 0.50 & 0.68 \end{bmatrix}$

The eigenvalues of **C** are 1.19 and 0.19

These values show the large variance in the direction of $\mathbf{v}_1$ (the line of best fit) and small variance in the direction of $\mathbf{v}_2$




- The eigenvectors are $\begin{bmatrix} 0.71 & -0.701 \\ 0.70 & 0.71 \end{bmatrix}$
- So, $\mathbf{v}_1 = [0.71, 0.70]$
- This is $y = \frac{0.70}{0.71}x$ which is roughly the line $y = x$
- and $\mathbf{v}_2$ is perpendicular to that

4. Choose the $K$ largest eigenvalues (to get rid of) to account for $p\%$ (to keep) of $T$
   - Reduce number of dimensions to keep $p\%$ of the variance

From the variances that were found above, the total variance,
$T = 1.19 + 0.19 = 1.38$

   - The proportion of variance in the data for $\mathbf{v}_1$ is $p = 1.19/1.38 = 0.86$

So if we projected the data points onto $\mathbf{v}_1$,
we would still retain 86% of the variation in the data

# Active Shape Models

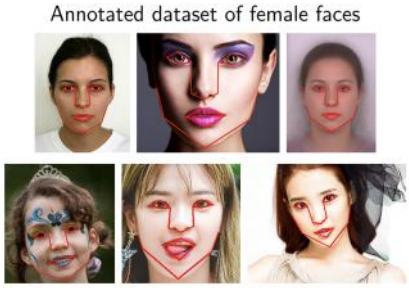
Monday, February 26, 2024 4:45 AM

## Basis

- **Model-based vision** - Analyze images using predefined models in an iterative process
- Active Shape Models - one method of MBV
  - Allows for **non-rigid** shape matching
    - Allow shapes to scale, rotate, translated, etc.
    - Allow shapes to change so that parts of the shape can move independently
  - Able to generate different (valid) shapes based on the shapes supplied in the training dataset
  - Allows shape to be found in an unseen (new) image

---

## Dataset




- Data samples need to be annotated with the landmark first
- For example, there are 6 samples (in reality there would be more)
- For each sample
  - 20 datapoints (landmarks)
    - Each data points there is (x, y) feature => 40 variables
    - (x, y) = the coordinates of the points
  - More data points = better results
    - But increase the time for manual annotation

---

## PCA on Dataset

1. Assemble data in a matrix, $\mathbf{D}$ ($S \times N$ *or* $N \times S$ )
   - $N = 40$ rows (20 for x, 20 for y)
   - $S = 6$ columns (per image)

| | $x_1$ | ... | $x_{20}$ | $y_1$ | ... | $y_{20}$ |
|---|---|---|---|---|---|---|
| $s_1$ | | | | | | |
| ... | | | | | | |
| $s_6$ | | | | | | |

2. Compute the Covariance matrix $\mathbf{C}$
   - $N = 40 \Rightarrow$ Size of $\mathbf{C} = 40 \times 40$

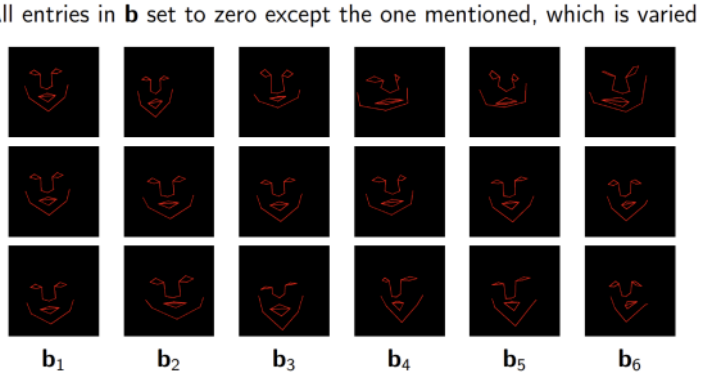
| | $x_1$ | ... | $y_{20}$ |
|---|---|---|---|
| $x_1$ | $\text{cov}(x_1, x_1)$ | | $\text{cov}(x_1, y_{20})$ |
| ... | | | |
| $y_{20}$ | $\text{cov}(y_{20}, x_1)$ | | $\text{cov}(y_{20}, y_{20})$ |

3. Find the Eigenvalues and Eigenvectors of $\mathbf{C}$ by solving $|\mathbf{C} - \lambda\mathbf{I}|$
   - $40$ Eigenvalues and $40 \times 40$ Eigenvector matrix $\mathbf{V}$
4. Choose the $K$ largest Eigenvalues
   - For ASM this is achieved with a **Shape Parameter** Vector $\mathbf{b}$ with size $N \times 1$
   - Instead of removing them, set their value to 0

---

## Generative Shape Model

- A Mean Shape $\bar{\mathbf{x}}$ can be obtained using $\mathbf{D}$
  - Represents the average position of feature points in the dataset
- Generative Shape Models $\mathbf{x}$ can then be generated from the model
  - $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Vb}$
  - Where by changing values in $\mathbf{b}$ generates different shapes
  - Note how if all entries in $\mathbf{b}$ then the model is just the mean shape



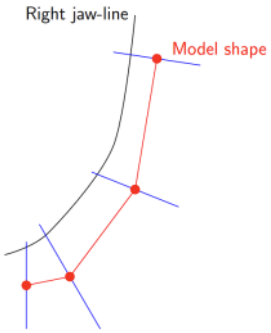

- Each of the elements in $\mathbf{b}$ varies different **modes of variation**:
  - $\mathbf{b_1}$ changes the vertical position
  - $\mathbf{b_2}$ varies the face between skinny and fat
  - $\mathbf{b_4}$ varies the face between looking left / right
- PCA automatically finds these different modes of variation
- Entries of $\mathbf{b}$ can be altered together to get more variation in the face
  - If values are too extreme, shape is no longer valid
- Notes on notation
  - $v_{i,j} = v_j^i =$ $j$th component in the $i$th data point

## Shape Fitting

- Apply an iterative localized search in the image using the model generated from the annotated dataset to an new image:

1. Place the shape into the image
    - Total set of parameters - $(s, \theta, \mathbf{r}, \mathbf{b})$
    - Allows scaling, rotation and translation with the shape parameter
    - Needs to ensure values of b still retains a valid shape

2. Search the neighbourhood of current feature points for better location
    - Calculate normals to the model curve at each point $x$
    - Search along the normal for the strongest edge
        - Note that in reality strongest edge != real edge
        - This gives a set of suggested points $x'$ (as they might not describe a valid shape)
        - Find $(s, \theta, \mathbf{r}, \mathbf{b})$ that rigidtransform $x$ to $x''$ that best fit $x'$




3. Fit the model to the new suggested shape
    - Use the model backwards $\mathbf{b} = \mathbf{V}^{-1}(\mathbf{x}'' - \bar{\mathbf{x}})$
    - New $\mathbf{b}$ that generates a new valid shape $\mathbf{x}$

4. Repeat until convergence

## Application

- Medical field
    - Positioning an artificial hip
    - Finding the shape of the hip so surgeon knows size/shape needed

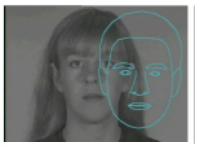
- Expression Changing
    - Splitting faces into triangles
    - Texture within triangles is warped to fit the model
    - Imagine face printed on a rubber sheet and stretched about

# Local Invariant Features

Tuesday, February 27, 2024 6:56 AM

## Motivation

- Global Representation - utilizing learned statistical shape and models and apply some transformation to create a fit to an image
  - Captures all features of an object using a single model
  - Limitations - vulnerable to:
    - Occlusions
    - Intra-category variations

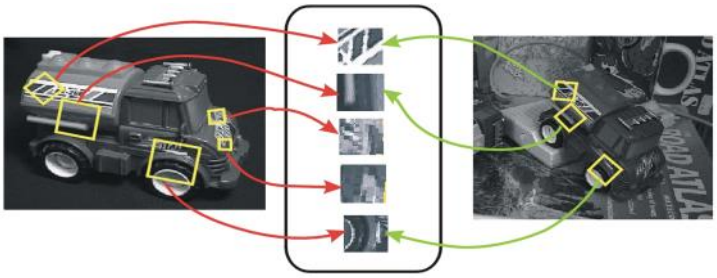


- Local Representation - representing different features of an object independently
  - Captures individual features of an object using different models
  - Describe and match only local features
  - Increase Robustness to occlusion and intra-category variations


- Example - Object Recognition
  - Model Based Object Recognition
    - Using a pre-defined holistic model to find objects in image
    - Global
  - Image Based Object Recognition
    - Using local features and patterns based on image information to identify objects
    - Local

---

## Local Invariant Descriptors



- Aka Image Features, Patches
- Features
  - Local, meaningful, detectable parts of the image
  - Location of a sudden change
- Features are useful
  - High content information
  - Invariant to change of view point and illumination
  - Reduces computational burden - only needs to care about these features instead of the whole image / object

---

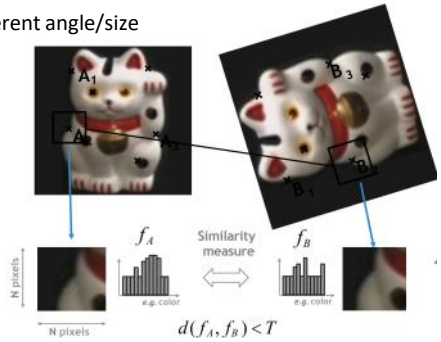
## Application

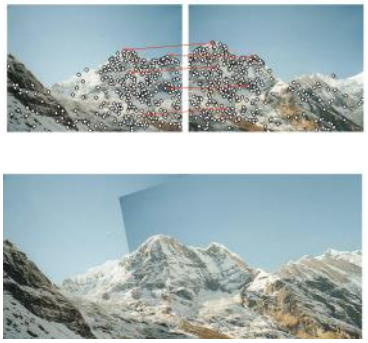
- Visual Simultaneous Localization And Mapping (SLAM)
  - 3D reconstruction
  - Motion tracking
- Image Matching / Image Retrieval
  - Image Alignment (Homography, fundamental matrix)
  - Indexing and database retrieval
- NASA Mars Rover feature matches
  - Robot navigation

---

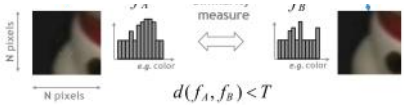
## Application - Image Stitching

- Stitch a bigger image based on different images from different angle/size
- Procedure
  - Detect feature points in both image
  - Find corresponding pairs
  - Align the images
- General Approach
  1. Find a set of distinctive keypoints (interest points)
     - Candidate locations / features
     - High information / unique signature
  2. Define a region around each keypoint
  3. Extract and normalize the region content



        - Candidate locations / features
        - High information / unique signature
    2. Define a region around each keypoint
    3. Extract and normalize the region content
    4. Compute a local descriptor(signature) from the normalized region
    5. Match local descriptors




- Common Requirements
    - Detect the same point <u>independently</u> in both image
        - This implies the need of a **repeatable detector**


    - For each point correctly recognize the corresponding one
        - Descriptor needs to be **reliable** and **distinctive**

---

## Requirements

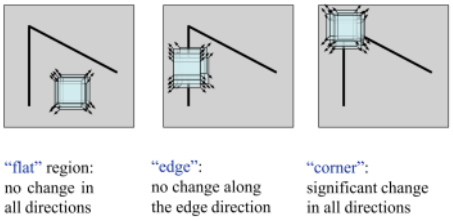
- Region Extraction needs to be **repeatable** and
    - **Invariant** to translation, rotation, scale changes
    - **Robust** or **Covariant** to out-of-plane (affine) transformation
    - **Robust** to lighting variations, noise, blur, quantization
- **Locality** - Features are local therefore robust to occlusion and clutter
- **Quantity** - a sufficient number of regions is needed to cover the object
- **Distinctiveness** - The region should contain 'interesting' structure
- **Efficiency** - Close to real-time performance

# Local Interest Point Detection (Keypoint Localization)

Tuesday, February 27, 2024 6:53 AM

## Harris Corner Detector - Basis

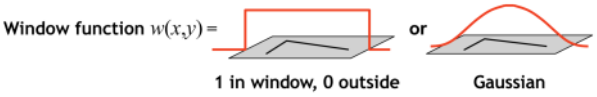
- A detector that detects the same point independently in different images is needed
- Interest Points need to provide information that allows that independency
  - Edges - only localize in one direction
  - Corners - provide repeatable points for matching
    - In the region around a corner, image gradient has two or more dominant directions
- Detector that detects Corners as Distinctive Interest Points
  - Shifting a window in any direction should give a large change in intensity




---

## Harris Corner Detection - Formulation

- A change in intensity for the shift $[u, v]$ is given by:
  - $E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$
  - Where:
    - $w(x,y)$ is a window function centered at $(x,y)$
    - $I(x,y)$ is the intensity function of point $(x,y)$




- Approximation for **small** shift:
  - $E(u,v) \cong [u,v]\, M \begin{bmatrix} u \\ v \end{bmatrix}$
  - Where $M$ is a 2x2 **2nd moment** matrix computed from image derivatives (see Edge Detection):
    - $M = \sum_{x,y \in \mathrm{w}} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$
    - $I_x$ and $I_y$ are the gradients with respect to $x$ and $y$ respectively
  - This sum over image region (window) that are checking for corner


---

## Multivariate Gaussian Model & Singular Value Decomposition (SVD)

- Assume the data is a Multivariate Gaussian Model
  - $\mathcal{N}\left(\underline{x}; \underline{\mu}, \Sigma\right) = \frac{1}{(2\pi)^{\frac{d}{2}}} |\Sigma|^{-\frac{1}{2}} \exp[-\frac{1}{2}\left(\underline{x} - \underline{\mu}\right)^T \Sigma^{-1} (\underline{x} - \underline{\mu})]$
  - Where $\Sigma$ is a symmetric "positive definite" covariance martix
    - In the case of Harris, the 2x2 $M$ matrix

**Maximum Likelihood estimates**

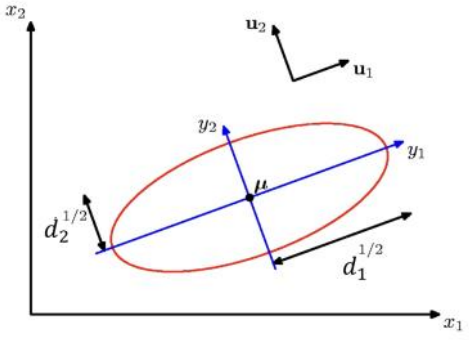
$$\hat{\mu} = \frac{1}{m} \sum_i x^{(i)}$$

1st moment of the data

$$\hat{\Sigma} = \frac{1}{m} \sum_i (x^{(i)} - \hat{\mu})^T (x^{(i)} - \hat{\mu})$$

Mean centered 2nd moment of the data

- Major directions of variance can be analyzed using the 2nd moment matrix
  - In cases of spherical and diagonal matrices, variance can be easily measured (see Segmentation)
  - For Full, a new coordinate system parallel to the ellipses is needed
- SVD - any $N \times N$ matrix can be written as a product of 3 matrices
  - where $U$ and $V$ are unitary matrices - the columns are orthogonal vectors and have unit length vectors
  - $D$ is a diagonal matrix where $d_1, d_2 \geq 0$
- In this case:
  - $$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{matrix} \boldsymbol{u_1} & \boldsymbol{u_2} \\ \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix} \end{matrix} \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix}^T = U \cdot D \cdot V^T$$
  - Where the vectors $\boldsymbol{u_1}$ and $\boldsymbol{u_2}$ are arranged in order of importance
    - describing the variance of the columns of Matrix $A$
  - Since the covariance matrix $M$ is a 2x2 squared matrix
    - $U = V$ and values of $D$ is guaranteed to be real and positive
      - $A = U \cdot D \cdot U^T$
    - $tr(A) = a_{11} + a_{22}$
    - $\det(A) = a_{11}a_{22} - a_{12}a_{21}$




- In order to know the direction of the variance of the data (for a full covariance matrix)
  - SVD (aka Eigenvalue decomposition) of the covariance matrix $M$ tells direction

$$\begin{matrix} & \boldsymbol{u_1} & \boldsymbol{u_2} \\ \text{eigenvectors} & \begin{bmatrix} v_{11} \\ v_{21} \end{bmatrix} & \begin{bmatrix} v_{12} \\ v_{22} \end{bmatrix} \end{matrix}$$

eigenvalues $d_1, d_2 \geq 0$

- $\det(A) = a_{11}a_{22} - a_{12}a_{21}$

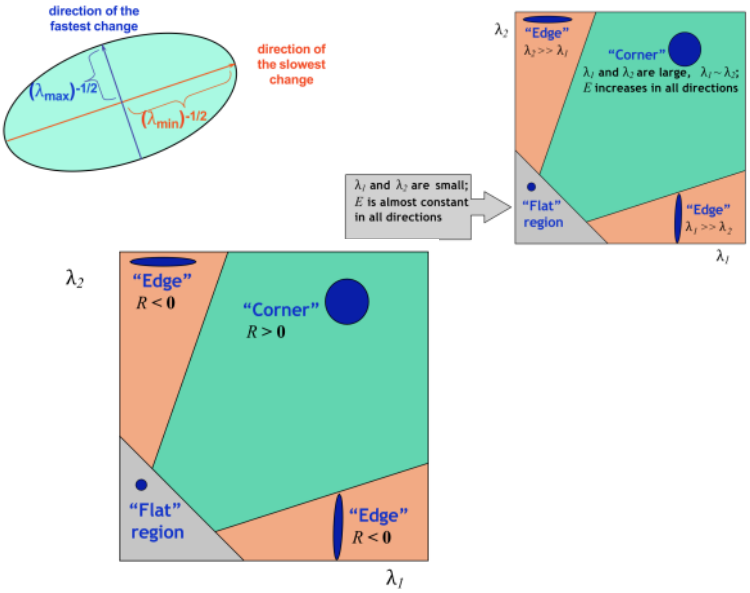
- In order to know the direction of the variance of the data (for a full covariance matrix)
  - SVD (aka Eigenvalue decomposition) of the covariance matrix $M$ tells direction
  - Scale of the numbers $d_1$ and $d_2$ gives direction of the spread

eigenvectors $\begin{bmatrix} v_{11} \\ v_{21} \end{bmatrix} \begin{bmatrix} v_{12} \\ v_{22} \end{bmatrix}$

eigenvalues $d_1, d_2 \geq 0$

---

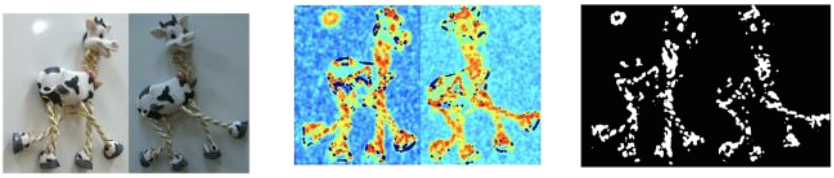
## Harris Corner Detection - Mathematics




- Intensity change in shifting window: $E(u,v) \cong [u,v]M\begin{bmatrix} u \\ v \end{bmatrix}$
- Let $\lambda_1$ , $\lambda_2$ be the eigenvalues $(d_2, d_1)$ of $M$
  - Note how the negative in index changes
- The eigenvalues can be used for classification
  - When $\lambda_1$ , $\lambda_2$ are large and $\lambda_1 \approx \lambda_2$ => corner
  - But what is "large" => **Corner Response** $R$
- $R = \det M \ - k(trace\ M)^2$
  $= \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$ as $M$ is 2x2 squared matrix
- Where $\alpha$ is a constant (0.04 - 0.06)
- Note that R:
  - Depends only on eigenvalues of $M$
  - Is Large for a corner
  - Is negative with large magnitude for an edge
  - $|R|$ is small for a flat region

---

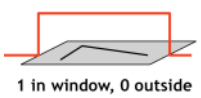
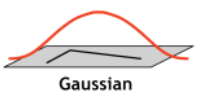
## Harris Corner Detection - Workflow

- 
- Original Images => Compute Corner Responses R => Threshold to preserve only larger R
  - Note that when threshold is carried out, pixels close to the point x,y instead of the exact location is found => local maxima needed
- 
- => Takes local maxima only => Results

---

## Harris Corner Detection - Fast Approximation

- Window Functions can be:

| | |
|---|---|
| • Option 1 - uniform window<br>○ Sum over square window<br>○ $M = \sum_{x,y \in w} \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}$<br>○ Problem - not rotation invariant<br>  | • Option 2 - Smooth with Gaussian<br>○ Gaussian already performs weighted sum<br>○ $M = g(\sigma) * \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}$<br>○ Result is rotation invariant<br>  |

- By using Gaussian, $R$ can be computed without the actual eigenvalues and can be done with convolution instead of summation

- The Second Moment Matrix can be computed by
  - $M(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$
  - Where $\sigma_I$ defines the window size and $\sigma_D$ is the value used to blur the image

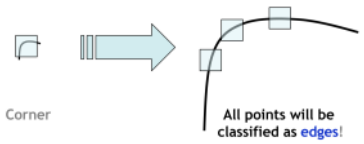
$$R = \det[M(\sigma_I, \sigma_D)] - \alpha\left[trace\left(M(\sigma_I, \sigma_D)\right)\right]$$
$$= g(I_x^2)g\left(I_y^2\right) - \left[g\left(I_x I_y\right)\right]^2 - \alpha\left[g(I_x^2) + g\left(I_y^2\right)\right]^2$$

1. Image derivatives (**blur first**)
2. Square of derivatives
3. Gaussian filter $g(\sigma_I)$
4. Cornerness function - both eigenvalues are strong
5. Non-maxima suppression

---

## Harris Corner Detection - Properties

- Precise Localization
- High repeatability
- Rotation Invariant to image rotation
  - The ellipse rotates but the shape (eigenvalues) remains the same




- **Not** Scale invariant to image scale
  - If the corner is enlarged, different parts of it will be classified as edges
  - Given an interest point, how much surrounding regions do we need to grab?
  - Same window size for both image, what if we know the scale of the image so different window size can be used? (See LoG)

# Scale Invariant Region Selection

Tuesday, February 27, 2024 6:57 AM

## Scale Selection - Naïve Approach: Exhaustive Search


- **Scale** - the size of the region around an interest point
- Multi-scale procedure
  - Given an interest point, compare the descriptors while varying the patch size
  - 

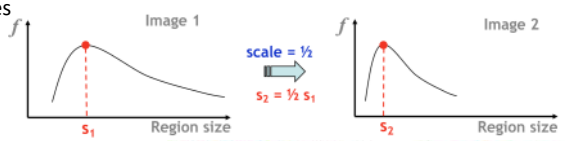
  - Until the exact match is found

- Computationally inefficient (almost impossible)
- Inefficient but possible for matching
- Prohibitive for retrieval in large database
- Prohibitive for recognition

---

## Automated Scale Selection

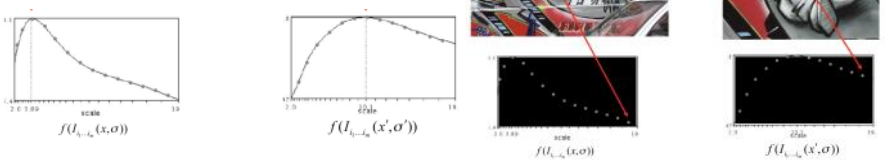
- Design a function on the region that is "scale invariant"
  - Give the same value for corresponding regions, even if they are at different scales
- For a point in one image, consider it as a function of region size (patch width)




- Common Approach:
  - Take a local maximum of the function
  - Region size for which the maximum is achieved should be invariant to images scale

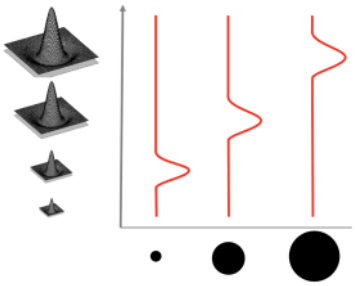
- This scale invariant region size is found in each image **independently**

- Example:
  - Start off with same window size
  - Calculate corner response $R$ for each image
  - Increment the scale (scale signature) and find $R$
  - The maxima indicates the scale for each image
  - Normalize: Rescale to fixed size



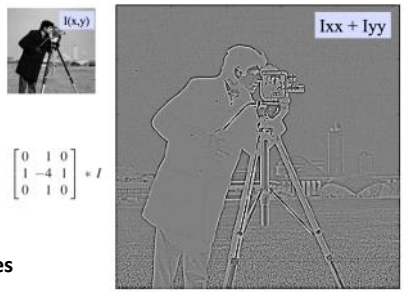

---

## The "function" - Laplacian



- Automated scale selection requires a scale invariant function $f$
- Laplacian of Gaussian (aka "blob" detector) has such property

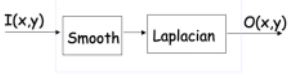
- Edge Position occurs at the zero-crossing in the 2nd derivative (See Edge Detection)
  - Which is more precise than 1st derivatives (beneficial for finding local interest points)
- $\nabla^2 I(x,y) = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2}$
  $= I_{xx} + I_{yy}$




- Laplacian is a linear filter (convolution)

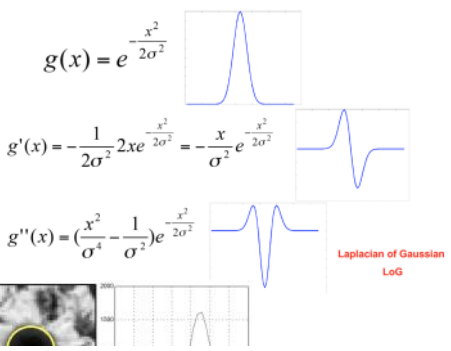
$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} * I$$

- Notes about Laplacian
  - $\nabla^2 I(x,y)$ is **scalar**
    - It can be found using a single mask
    - However orientation information is lost
  - $\nabla^2 I(x,y)$ is the sum of **second-order derivatives**
    - But taking derivatives increases noise
    - Very noise sensitive
  - It is always combined with a smoothing operation
    - Usually Gaussian as it allows LoG

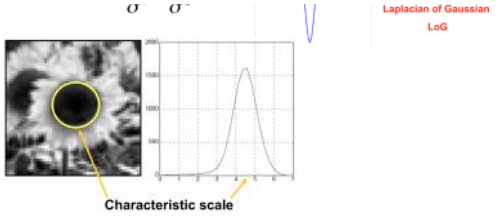



- If $K_{\nabla^2}$ is Laplacian Kernel and $G_\sigma$ is a Gaussian Kernel (again See Edge Detection)
  - $\left(K_{\nabla^2} * \left(G_\sigma * I\right)\right) = \left(K_{\nabla^2} * G_\sigma\right) * I = \left(\nabla^2 G_\sigma\right) * I$

- LoG locates blobs and gets its maximum response when the size of the blob is fit perfectly



- $\left(K_{\nabla^2} * \left(G_\sigma * I\right)\right) = \left(K_{\nabla^2} * G_\sigma\right) * I = \left(\nabla^2 G_\sigma\right) * I$




- LoG locates blobs and gets its maximum response when the size of the blob is fit perfectly
  - Scale of the blob in radius of pixels is determined by the $\sigma$ parameter of the LoG
  - Scale is proportional to $\sigma$

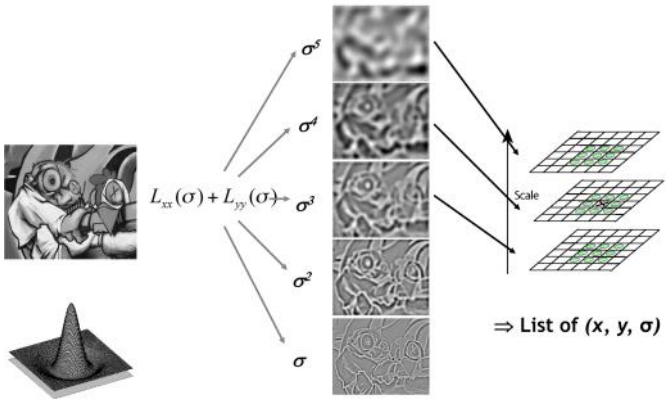
- **Characteristic Scale** - the scale that produces extreme values (peak) of LoG response

---

## Laplacian-of-Gaussian (LoG) Detector

- Interest points - Local maxima in scale space of LoG
  - Blobs instead of corners
  - The $x, y$ points do need to correspond to particular features

- How to find them
  1. Convolve the image with LoG at different scales $\sigma$
  2. Find the local extreme for each $\sigma$
  3. Look along scale space
     i. Compare with 8 neighbours + 9 neighbours below and 9 above
  4. If it is maxima among all 26 neighbours, add to the list
     i. List of $(x, y, \sigma)$




- Workflow
  - 
  - Apply LoG with different $\sigma$ (detecting different features) => Combine all the local extrema
  - Note that so far we have only found local interest features (corners/blobs) but not yet encoded the region to form a signatur e for matching

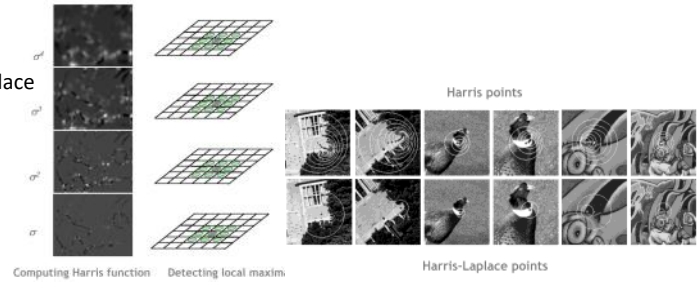
---

## Harris-Laplace

- Allow Harris Corner Detector to be scale invariant by combining with Laplace

1. Initialization - Multiscale Harris Corner Detection
   i. For different scales, compute the corner response
   ii. Find local maxima for each image

2. Scale Selection based on LoG
   i. Only choose the points that are **also** maxima in LoG




- Combining these detectors give more varieties of features (LoG gives blobs and Harris gives corners)

---

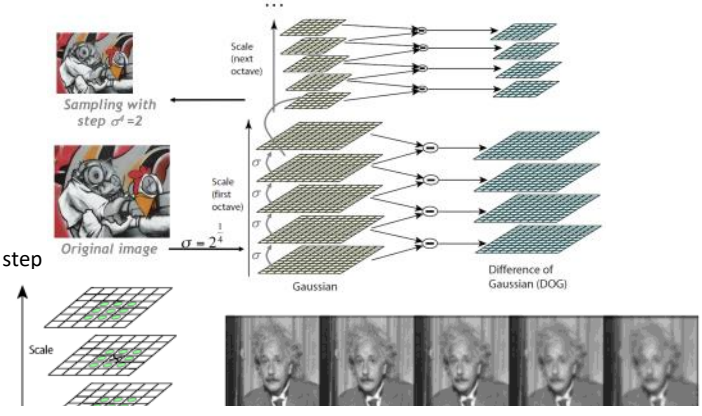
## Difference-of-Gaussian (DoG) Detector

- A fast approximate of LoG using a difference of Gaussian (DoG) at different scales
  - $DoG = G\left(x, y, k\sigma\right) - G(x, y, \sigma)$

- Used for feature detection (e.g. Lowe's SIFT pipeline)

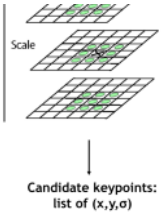
- Advantages
  - No need to compute 2nd derivatives
  - Gaussians are computed anyway, e.g. in a Gaussian pyramid

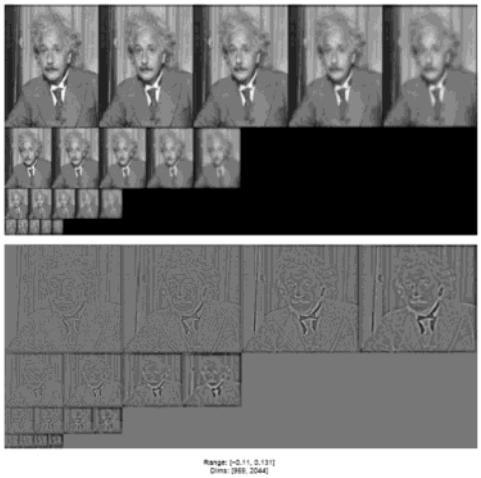
- Computation in Gaussian scale pyramid
  - Convolve the original image with Gaussian
  - Convolve the result with Gaussian again 4 times (octaves)
    - If $\sigma$ is chosen well, this computation is efficient and allows $\sigma$ to step
  - Note that increasing $\sigma$ = increase scale = reduce resolution
  - Subsampling




- The similar steps as LoG:
  1. Detect maxima of DoG in scale space

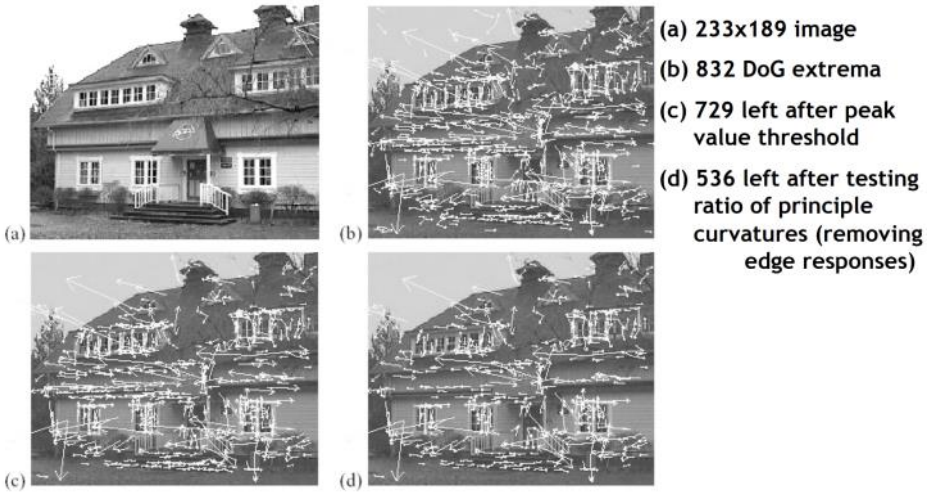
- Subsampling



- The similar steps as LoG:
  1. Detect maxima of DoG in scale space
  2. Reject points with low contrast (threshold)
  3. Eliminate edge responses

- Example:




---

## Scale Invariant Detection Summary

- Given: Two images of the same scene with a large scale difference between them
- Goal: Find the same interest points independently in each image

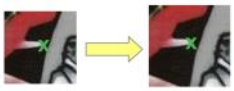
- Solution: Search for maxima of suitable functions in scale and in space (over the image)
- Two strategies - LoG and DoG

- These can be used either on their own, or in combination with single scale keypoint detectors (e.g. Harris)

# Local Descriptors

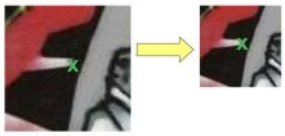
Tuesday, February 27, 2024 6:57 AM

## Basis



- Aim: Matching identified features / points between images

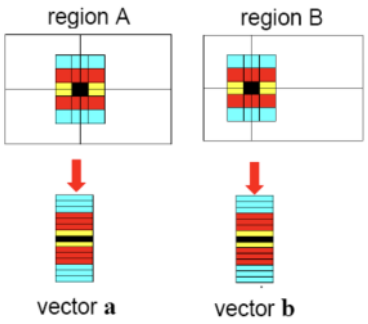
- Need to achieve Feature Invariance before matching
  1. Make sure the detector is invariant to translation, rotation and scale
     - Know how to find interest points (locations & corresponding characteristic scales)
     - Know how to remove the effects of difference in scale once one of these interest points is detected
       - By normalizing the scale either way:
         - Scaling one image to fit another
         - Scale both images to a fixed scale



  2. Design an invariant feature *descriptor*
     - A **descriptor** captures information in a region around the detected interest point ('signature' of the region's content)

---

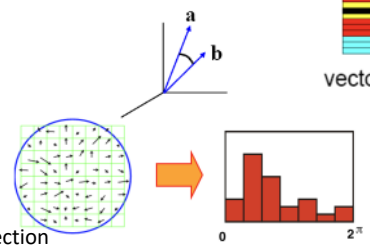
## Simplest Local Feature Descriptor: Square Window of Pixels




- Aim: Given a region with its content, encode the information uniquely for matching

- Simplest solution - list of intensities within a patch
  - Take the pixels in a square window and encode their intensities in order in a vector
  - Then compute the distance between the vectors that regions
  - Invariant to translation
  - Can be invariant to scale (after normalization)

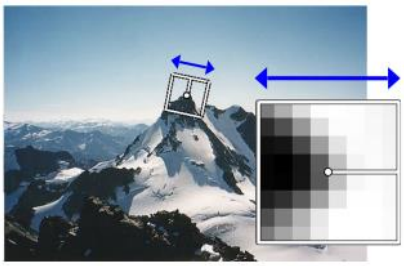
- Issues - Not invariant to rotation / illumination / 3D viewpoint change




- Solution - histograms of gradient directions within the patch
  - For every pixel within the patch, compute its gradient
  - Then look about orientation by building the histogram of gradient direction
  - Note: Magnitude is "ignored" as it is affected by illumination, in which the descriptor should be insensitive to

---

## Rotation Invariant Descriptors

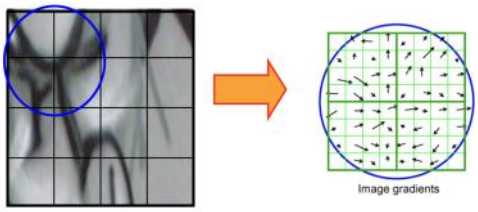


- Find local orientation
  - Dominant direction of gradient for the image patch
    - Gradient > Intensity as it allows orientation to be varied

- Rotate each patch according to this angle
  - This puts the patches into a canonical orientation

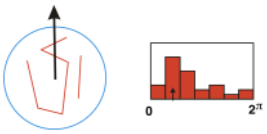
---

## Scale Invariant Features Transformation (SIFT)

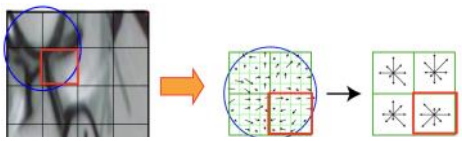



1. SIFT feature location and scale (size) using DoG
   - Note: DoG aligns towards edges (as it is an approximate of Laplacian)
     - Which could introduce ambiguity as edges are not unique enough
   - Solution is to check and reject features along edges after feature location and size selection
     - 2nd Moment Matrix based on 1st derivative can be used to generate the corner response to tell if a feature is an edge

2. Scale Normalization

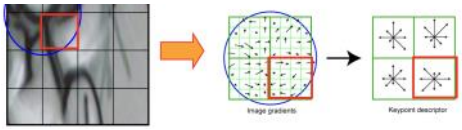



3. Orientation Normalization
   a. Gradient orientation over a 16x16 pixel region (see image) around the interest point
   b. Compute histogram of image gradient orientations for all pixels within 16x16 region
      - Gradient orientation over the region is quantized into **8** bins spread over the range 0-360°
        - Compute orientation histogram
        - Select dominant orientation
        - Normalize - rotate(points upwards) and re-compute to fixed orientation

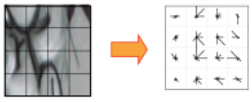


4. Divide patch (16x16 window) into 4x4 sub-patches - 16 cells per sub-patch

    - Select dominant orientation
    - Normalize - rotate(points upwards) and re-compute to fixed orientation



4. Divide patch (16x16 window) into 4x4 sub-patches - 16 cells per sub-patch
    - Note: Blue circle indicates 1/4 of the window for the sake of illustration
5. Compute histogram of gradient orientations (8 reference angles / direction) for all pixels inside each sub-patch
    - Note: these sub-patch gradient is normalized (scale & orientation) as the patch is normalized
    - Resulting descriptor - 4x4 (pixels) x 8 (directions) = 128 dimensions



- Note there is a specific order in each these vectors are concatenate
    - Unique identifier for each feature hence comparable to feature in another image
- One image yields:
    - $\boldsymbol{n}$ 128-dimensional descriptors - each one is a histogram of gradient orientations within a patch
        - $[\boldsymbol{n} \times 128]$ matrix
    - $\boldsymbol{n}$ scale parameters specifying the size of each patch
        - $[\boldsymbol{n} \times 1]$ vector
    - $\boldsymbol{n}$ orientation parameters specifying the angle of each patch
        - $[\boldsymbol{n} \times 1]$ vector
    - $\boldsymbol{n}$ 2D points giving positions of the patches
        - $[\boldsymbol{n} \times 2]$ matrix

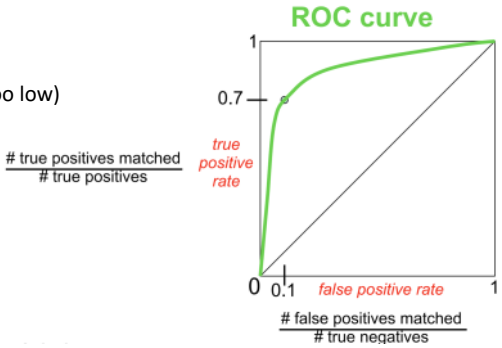
- Extraordinary robust matching technique
    - Can handle changes in viewpoint up to $\approx 60°$ out-of-plane rotation
    - Can handle significant changes in illumination (sometimes even day vs. night)
    - Fast & efficient - can run in real time
    - Lots of code available

---

## Feature Matching


- Given a feature in $I_1$, how to find the best match in $I_2$?
    1. Define **distance function** (usually Euclidean distance) that compares two descriptors
        - Simple approach - Sum of Square Difference (SSD) between features $SSD(f_1, f_2)$
            - Allow good scores to very ambiguous (bad) matches
        - Better approach - ratio distance $SSD(f_1, f_2) / SSD(f_1, f_{2'})$
            - $f_2$ is best SSD match to $f_1$ in $I_2$
            - $f_{2'}$ is second best SSD match to $f_1$ in $I_2$
            - Gives large values ($\sim 1$) for ambiguous matches - too many good matches

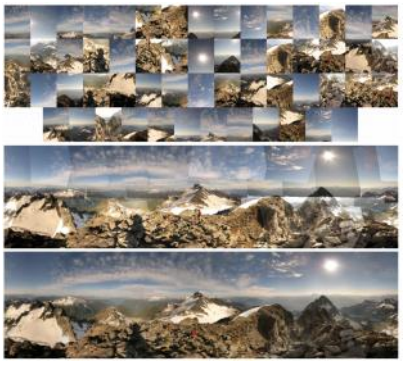

    2. Test all the features in $I_2$ and find the one with the min distance
- Eliminating bad matches
    - Thresholding - throw out features with distance > threshold
        - Ideally maximizing true positives
        - Minimizing false positives (threshold too high) and false negatives (threshold too low)
    - Evaluating results - Receiver Operator Characteristics Curve (ROC)
        - Generated by counting correct and incorrect matches for different threshold
            - True Positive Rate = $\frac{TP}{TP+FN}$
            - False Positive Rate = $\frac{FP}{FP+TN}$
        - Ideally maximize area under ROC curve (AUC)
        - Useful for comparing different feature matching methods




---

## Application (after having Local Features)

- Wide-Baseline Stereo
- Automatic Mosaicing
- Panorama Stitching
- Recognition of Objects and scenes
- SIFT application
    - Sony Aibo



- SIFT application
  - Sony Aibo

Sivic and Zisserman, 2003

---

## Summary: Advantages of Local Features

- Critical to find distinctive and repeatable local regions for multi-view matching
- Complexity reduction via selection of distinctive points
- Describe images, objects, parts without requiring segmentation
  - Robust to clutter and occlusion
- Robustness - similar descriptors in spite of moderate view changes, noise, blur, etc.

# Essential Reading

Saturday, March 9, 2024 10:14 AM

notes_SIFT descriptor...

(c) 2004 F. Estrada & A. Jepson & D. Fleet

## SIFT features

Scale Invariant Feature Transform (SIFT) is an approach for detecting and extracting local feature descriptors that are reasonably invariant to changes in illumination, image noise, rotation, scaling, and small changes in viewpoint.

Detection stages for SIFT features:

- Scale-space extrema detection
- Keypoint localization
- Orientation assignment
- Generation of keypoint descriptors.

In the following pages we'll examine these stages in detail.



(c) 2004 F. Estrada & A. Jepson & D. Fleet

## Scale-space extrema detection

Interest points for SIFT features correspond to local extrema of difference-of-Gaussian filters at different scales.

Given a Gaussian-blurred image

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

where

$$G(x, y, \sigma) = 1/(2\pi\sigma^2) \exp^{-(x^2+y^2)/\sigma^2}$$

is a variable scale Gaussian, the result of convolving an image with a difference-of-Gaussian filter

$$G(x, y, k\sigma) - G(x, y, \sigma)$$

is given by

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \qquad (1)$$

Which is just the difference of the Gaussian-blurred images at scales $\sigma$ and $k\sigma$.

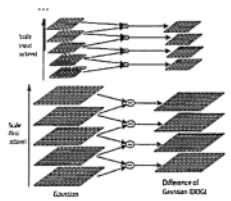


Figure 1: Diagram showing the blurred images at different scales, and the computation of the difference-of-Gaussian images (from Lowe, 2004, see ref. at the beginning of the tutorial)

The first step toward the detection of interest points is the convolution of the image with Gaussian filters at different scales, and the generation of difference-of-Gaussian images from the difference of adjacent blurred images.





## Scale-space extrema detection

The convolved images are grouped by octave (an octave corresponds to doubling the value of $\sigma$), and the value of $k$ is selected so that we obtain a fixed number of blurred images per octave. This also ensures that we obtain the same number of difference-of-Gaussian images per octave.

Note: The difference-of-Gaussian filter provides an approximation to the scale-normalized Laplacian of Gaussian $\sigma^2\nabla^2 G$. The difference-of-Gaussian filter is in effect a tunable bandpass filter.

## Scale-space extrema detection

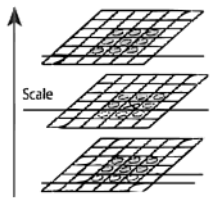



Figure 2: Local extrema detection, the pixel marked $\times$ is compared against its 26 neighbors in a $3 \times 3 \times 3$ neighborhood that spans adjacent DoG images (from Lowe, 2004)

Interest points (called keypoints in the SIFT framework) are identified as local maxima or minima of the DoG images across scales. Each pixel in the DoG images is compared to its 8 neighbors at the same scale, plus the 9 corresponding neighbors at neighboring scales. If the pixel is a local maximum or minimum, it is selected as a candidate keypoint.





## Scale-space extrema detection

For each candidate keypoint:

- Interpolation of nearby data is used to accurately determine its position.

- Keypoints with low contrast are removed

- Responses along edges are eliminated

- The keypoint is assigned an orientation

To determine the keypoint orientation, a gradient orientation histogram is computed in the neighborhood of the keypoint (using the Gaussian image at the closest scale to the keypoint's scale). The contribution of each neighboring pixel is weighted by the gradient magnitude and a Gaussian window with a $\sigma$ that is 1.5 times the scale of the keypoint.

Peaks in the histogram correspond to dominant orientations. A separate keypoint is created for the direction corresponding to the histogram maximum,

and any other direction within 80% of the maximum value.

All the properties of the keypoint are measured relative to the keypoint orientation, this provides invariance to rotation.





## SIFT feature representation

Once a keypoint orientation has been selected, the feature descriptor is computed as a set of orientation histograms on $4 \times 4$ pixel neighborhoods. The orientation histograms are relative to the keypoint orientation, the orientation data comes from the Gaussian image closest in scale to the keypoint's scale.

Just like before, the contribution of each pixel is weighted by the gradient magnitude, and by a Gaussian with $\sigma$ 1.5 times the scale of the keypoint.

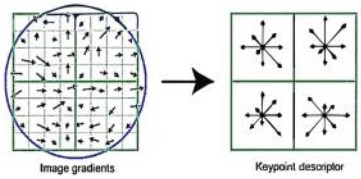



Figure 3: SIFT feature descriptor (from Lowe, 2004)

Histograms contain 8 bins each, and each descriptor contains an array of 4 histograms around the keypoint. This leads to a SIFT feature vector with $4 \times 4 \times 8 = 128$ elements. This vector is normalized to enhance invariance to changes in illumination.





## SIFT feature matching

- Find nearest neighbor in a database of SIFT features from training images.

- For robustness, use ratio of nearest neighbor to ratio of second nearest neighbor.

- Neighbor with minimum Euclidean distance → expensive search.

- Use an approximate, fast method to find nearest neighbor with high probability.

(c) 2004 F. Estrada & A. Jepson & D. Fleet

## Recognition using SIFT features

- Compute SIFT features on the input image

- Match these features to the SIFT feature database

- Each keypoint specifies 4 parameters: 2D location, scale, and orientation.

- To increase recognition robustness: Hough transform to identify clusters of matches that vote for the same object pose.

- Each keypoint votes for the set of object poses that are consistent with the keypoint's location, scale, and orientation.

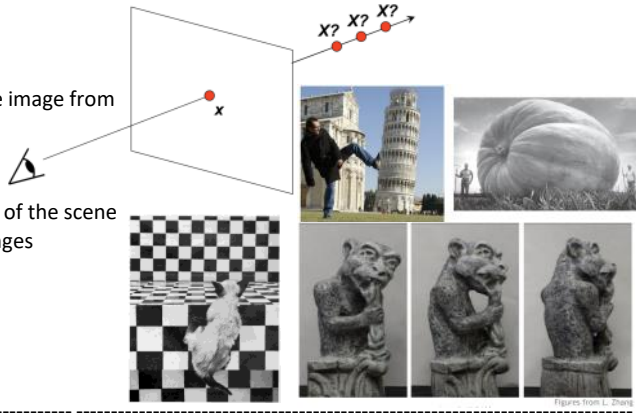
- Locations in the Hough accumulator that accumulate at least 3 votes are selected as candidate object/pose matches.

- A verification step matches the training image for the hypothesized object/pose to the image using a least-squares fit to the hypothesized location, scale, and orientation of the object.

# Stereo Analysis

Tuesday, February 27, 2024 6:30 AM

## Basis

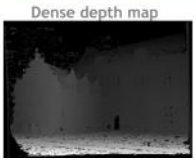


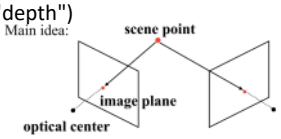

- Image - a point on the projection (through a vector) of the 3D world
  - One image is inherently **ambiguous** - which point along the trajectory is the image from
  - As structure and depth are inherently ambiguous from single views
- **Multi-view Geometry** is needed for recovery of structure due to this ambiguity
  - Analyzing how different perspectives in a scene is related to refer structure of the scene
  - How the projection of 3D points onto 2D images changes as viewpoint changes
- Goal: Recovery of 3D structure
  - Human uses visual cues - Shading, Texture, Focus, Perspective, Motion
  - But these cues are less helpful for computer

---

## Stereo Vision


- Generic problem formulation:
  - Given several images of the same object / scene
  - Compute a representation of its 3D shape
- Narrower Formulation:
  - Given a calibrated binocular stereo pair
    - Binocular stereo pair - Two images taken from slightly different perspectives (similar to left & right eyes)
    - Calibrated - where camera parameters are known
  - Fuse it to produce a depth image (similar to how human eyes perceive sense of "depth")

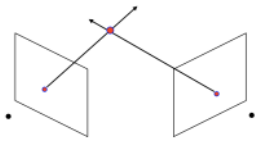


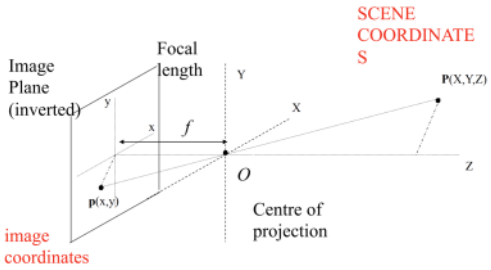




- Stereo - infer 3D shape of scene from two(multiple) images from different viewpoints
  - Note: viewpoint = camera = optical center
  - The scene point will have a different projection for each viewpoint
  - These projection can be extended **from** the images until the projection intercepts - indicating the position of the scene point

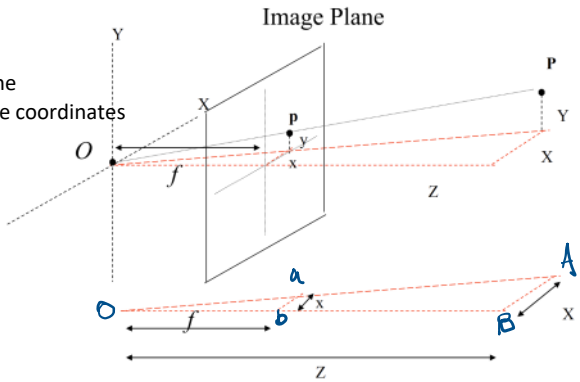
---

## Depth with Stereo





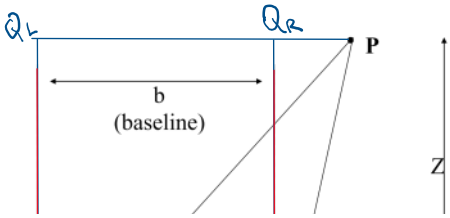

- Basic Principle - **Triangulation**
  - Gives reconstruction as intersection of 2 rays
  - Requires (Assumption):
    - Camera pose (calibration)
      - Cameras are placed in world coordinate
      - Where they are placed in relation to the other is known
    - Point correspondence
      - The two points correspond to the same point




- **<u>Camera Model</u>**
  - Pinhole camera - the center of projection
  - **Focus length** - the distance between center of projection and the image plane
  - Perspective transform - from scene coordinates (world coordinates) to image coordinates
    - $Z$ - the distance from the camera point to the scene point
    - We want to find $Z$ as it represents the "depth"
    - The triangle $Oab$ and $OAB$ are similar, therefore:
      - $\frac{x}{X} = \frac{f}{Z} \Rightarrow x = \frac{f}{Z}X$
      - $\frac{y}{Y} = \frac{f}{Z} \Rightarrow y = \frac{f}{Z}Y$

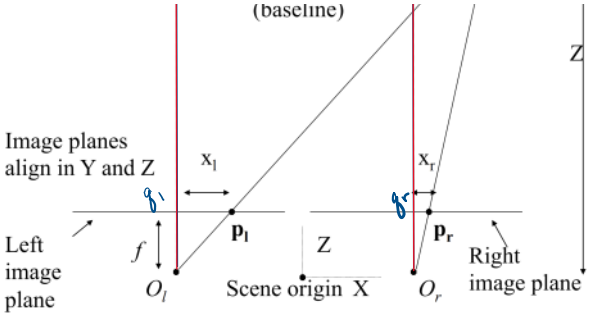
---

## A simple Stereo System

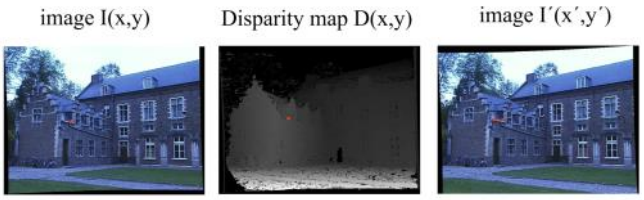



- **Baseline** - distance between the L and R camera
- For Left and Right image planes
  - L and R cameras are parallel, placed on the same line (same y axis)
  - L and R image planes are perpendicular to their camera's optical plane

- **Baseline** - distance between the L and R camera
- For Left and Right image planes
  - L and R cameras are parallel, placed on the same line (same y axis)
  - L and R image planes are perpendicular to their camera's optical plane
  - Scene (world) origin $X$ is at the midpoint of the baseline



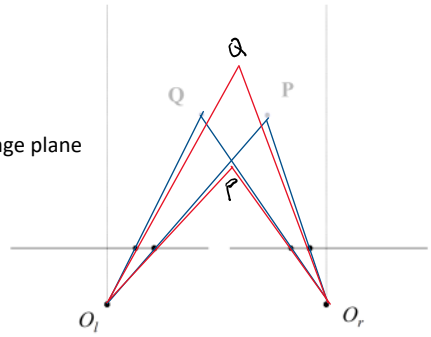

- The triangles $\Delta O_L Q_L P \sim \Delta\, O_L q_l p_l$ and $\Delta O_R Q_R P \sim \Delta O_r q_r p_r$
  - $\frac{x_l}{f} = \frac{X+\frac{b}{2}}{Z}$ and $\frac{x_r}{f} = \frac{X-\frac{b}{2}}{Z}$
  - $\frac{x_l - x_r}{f} = \frac{b}{Z} \Rightarrow Z = \frac{bf}{x_l - x_r}$
- **Disparity** - displacement between conjugate(corresponding) points in L&R images
  - $x_l - x_r$




- To get the sense of Depth ($Z$)
  - $Z = \dfrac{bf}{x_l - x_r}$
  - In theory baseline $b$ and focus length $f$ are known (often chosen)
  - we only need to estimate disparity
- Conclusion: If we can find the **corresponding points** in two images, we can estimate **relative depth** $(x',y')=(x+D(x,y), y)$

---

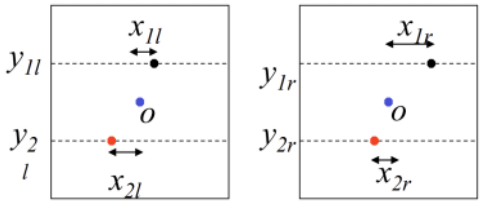
## Finding Correspondence




- <u>The Correspondence Problem</u>
  - It is difficult to reconstruct 3D world points from image points
    - We don't know how to make the correct correspondences between points in the image plane
    - Which is the correct sets of P and Q ?
- Difficult to find correspondences at every point
  - Sparse set of points
- Assume most scene points are visible in both views
- Assume corresponding points are similar
- Find correspondences at "interest" points
  - Need recognizable image structure

---

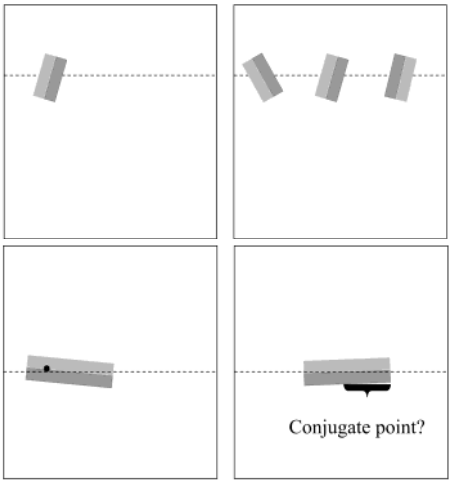
## The Epipolar Constraint




- Camera is placed at same $y$, therefore displacement only take place in $x$
  - We only need to look at points on the same $y$ value for each point
- Match for a give $(x_l, y_l)$ lies on the **Epipolar line** $y_r = y_l$
  - At least for a simple system
  - 1D search problem
- But there are still potentially large search space for each candidate point

---

## Correspondence Matching by Edges



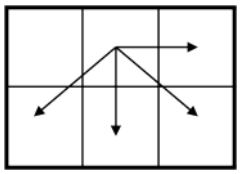

- Edges are convenient places to match
  - Corresponding to significant structure
  - Small number of points to match
  - Edge polarity and direction (orientation) provides cues for matching
  - Canny allows sub-pixel localization so points can be located accurately
  - Multi-scale location (coarse to fine search)
- However
  - Not all significant structures are on edges
  - Images gradients at corresponding points may not be equally high
    - Shadows, occlusions, illumination differences
  - Horizontal edges are difficult to match
    - Match points are poorly localized along Epipolar lines

---

## Correspondence by Correlation Matching

- Similar points do not necessarily lie on salient edges
- Corresponding points should look similar
  - Not identical - different viewpoints
- Cross-correlation search to find candidate matches
  - 1D search along epipolar lines

- Match only locally distinct Interest Points
  - Corners (Harris)
  - LoG / DoG
  - **Moravec Operator**
    - Non-linear filter
    - Calculate over some neighbourhood (e.g. 5x5)
      - $\sum\left(I_{i,j} - I_{(i+1,j)}\right)^2 \quad \sum\left(I_{i,j} - I_{(i-1,j+1)}\right)^2$
        $\sum\left(I_{i,j} - I_{(i+1,j+1)}\right)^2 \quad \sum\left(I_{i,j} - I_{(i,j+1)}\right)^2$
    - Output value is *minimum* of these values
    - Suppress non-maxima of the filter output
      - Isolate local maxima to get distinct points
    - Find points where intensity is varying quickly
      - Taking minimum eliminates edges as candidates



Pixel differences

---

## Components of Stereo Analysis

- **Correspondences**
  - Conjugate pairs of points
  - Potentially hard - lots of pairs

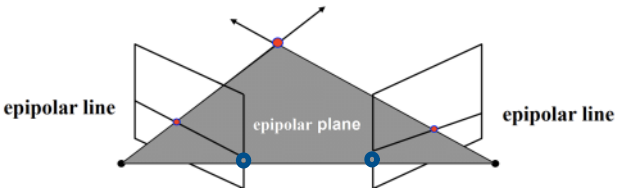
- **Reconstruction**
  - Calculate scene (world) coordinates according to image coordinates
  - Easy - once Calibration is done

- **Calibration**
  - Calculate parameters of cameras (b, f , etc.)

# Stereo Geometry

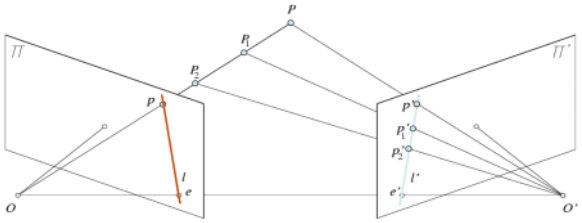
Saturday, March 9, 2024 10:10 AM

## General Case With Calibrated Cameras




- Most cases L&R cameras are not parallel on the same $y$ value
- But we can still use **Stereo Correspondence Constraints** to eliminates points
  - Use Epipolar plane as a constraint

- Terms Definitions:
  - **Baseline** - The line joining the cameras' centers
  - **Epipole** - The point of intersection between baseline and image planes
  - **Epipolar plane** - The plane containing baseline and world point
  - **Epipolar line** - The intersection of epipolar plane with the image plane

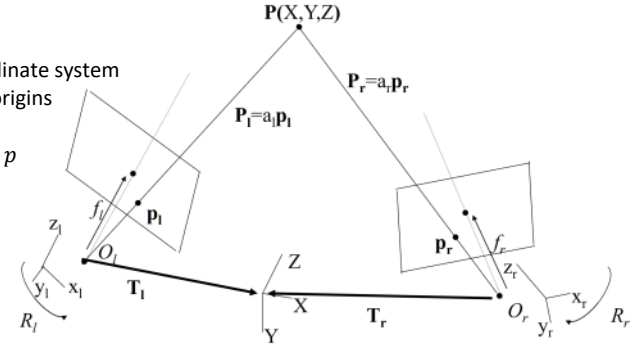


- Concepts:
  - All epipolar lines intersect at baseline
  - An epipolar plane intersects the left and right image planes in epipolar lines
  - Therefore for each point $p$ that lies on the epipolar line $l$
    - The corresponding point $p'$ must lies on the epipolar line $l'$
    - Line $l'$ is the intersection of the epipolar plane (defined by baseline and 3D point $P$) and the image plane
    - Baseline (line between cameras' centers) is chosen and known (as they are calibrated cameras)

- Conclusion: the general case is still a linear search problem

---

## Reconstruction



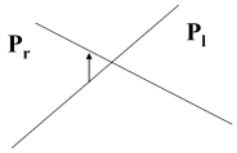

- Coordinate systems are related by
  - Rotation matrices $R_l$ and $R_r$ giving the **orientations** of each camera coordinate system
  - Translation vectors $\mathbf{T_l}$ and $\mathbf{T_r}$ between the camera origins and the scene origins

- The vectors $\boldsymbol{P_l}$ and $\boldsymbol{P_r}$ will be some factors ($a_l$ and $a_r$) along the line from $O$ to $p$
  - $\boldsymbol{P_l} = a_l \boldsymbol{p}_l, \qquad \boldsymbol{P_r} = a_r \boldsymbol{p}_r$

- The vectors $\boldsymbol{p}_l$ and $\boldsymbol{p}_r$ are $(x_l, y_l, f_l)$ and $(x_r, y_r, f_r)$ respectively

- Let $P_l'$ and $P_r'$ be vectors $P_l$ and $P_r$ in the world coordinate system
  - $\boldsymbol{P_l'} = \mathbf{T_l} + R_l \boldsymbol{P}_l \Rightarrow \boldsymbol{P_l'} = \mathbf{T_l} + R_l a_l \boldsymbol{p}_l$
  - $\boldsymbol{P_r'} = \mathbf{T_r} + R_r \boldsymbol{P}_r \Rightarrow \boldsymbol{P_r'} = \mathbf{T_r} + R_r a_r \boldsymbol{p}_r$

- Therefore $\boldsymbol{P_l'}$ and $\boldsymbol{P_r'}$ intersect at $P$ in world coordinate where $\mathbf{T_l} + R_l a_l \boldsymbol{p}_l = \mathbf{T_r} + R_r a_r \boldsymbol{p}_r$




- However measurement inaccuracies cause projected vectors to be unconcise
  - We need to find the mid-point of the vector between the closest points on each
  - This can also be found by solving a set of linear equations using the Essential Matrix

---

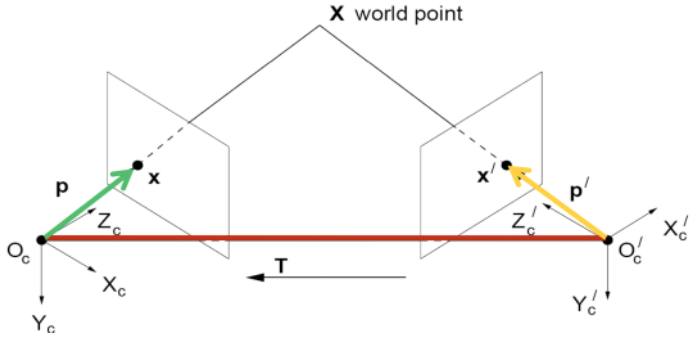
## Essential Matrix

- A matrix that relates the corresponding points from one image to another given the rotation and translation
- Essential Matrix is known when cameras are calibrated (otherwise fundamental matrix is used instead)

- Let $(X_C, Y_C, Z_C)$ and $(X_C', Y_C', Z_C')$ be two camera reference frames (camera coordinate systems)
- Since they are related by some 3x3 rotation matrix $\boldsymbol{R}$ and some translation vector $\mathbf{T}$
  - Therefore, for vectors $\boldsymbol{X}'$ ($\boldsymbol{X_C'}$) and $\boldsymbol{X}$ ($\boldsymbol{X_C}$) :
  - $\boldsymbol{X}' = R\boldsymbol{X} + \mathbf{T}$

- Recall: **Cross Product** $\times$ and **Dot Product** $\cdot$
  - $\vec{\boldsymbol{a}} \times \vec{\boldsymbol{b}} = \vec{\boldsymbol{c}}$ where $\vec{\boldsymbol{b}} \perp \vec{\boldsymbol{c}} \perp \vec{\boldsymbol{a}}$ and $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{c}} = \vec{\boldsymbol{b}} \cdot \vec{\boldsymbol{c}} = 0$

- Therefore to find the **Normal** to the epipolar plane defines by $\mathbf{T}$ and $\boldsymbol{X}'$
  - $\mathbf{T} \times \boldsymbol{X}' = \mathbf{T} \times (R\boldsymbol{X} + \mathbf{T})$
    $\mathbf{T} \times \boldsymbol{X}' = \mathbf{T} \times R\boldsymbol{X} + \mathbf{T} \times \mathbf{T}$
- Since the cross product between a vector and itself is 0:
  - $\mathbf{T} \times \boldsymbol{X}' = \mathbf{T} \times R\boldsymbol{X}$




- The dot product between $\boldsymbol{X}'$ and $\mathbf{T} \times \boldsymbol{X}'$ should be 0 as they are perpendicular

- $\boldsymbol{X}' \cdot (\mathbf{T} \times \boldsymbol{X}') = 0$ hence
  $\boldsymbol{X}' \cdot (\mathbf{T} \times R\boldsymbol{X}) = 0$

- Recall: Matrix form of cross product
  - $\vec{\boldsymbol{a}} \times \vec{\boldsymbol{b}} = [\boldsymbol{a}_x]\vec{\boldsymbol{b}}$

$$\vec{a} \times \vec{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \qquad [a_x] = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

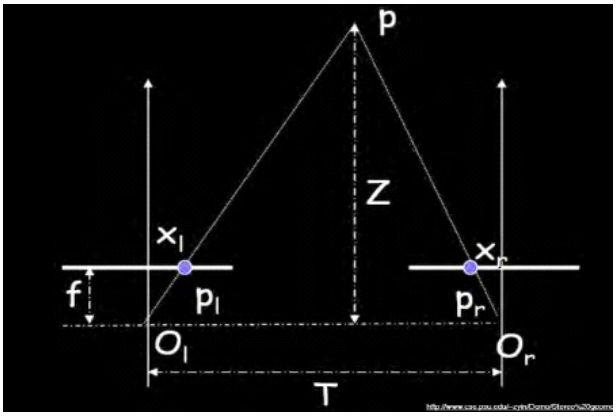
- Therefore $\boldsymbol{X}' \cdot (\mathbf{T} \times R\boldsymbol{X}) = 0$
  $\boldsymbol{X}' \cdot ([\mathbf{T_x}]R\boldsymbol{X}) = 0$

- $\boldsymbol{E} = [\mathbf{T_x}]R$
- $\boldsymbol{X}'^{\mathbf{T}}\boldsymbol{E}\boldsymbol{X} = 0$

---

## Essential Matrix in Parallel Cameras with same focus length

- Since cameras are align horizontally:
  - $R = \mathbf{I}$
  - $\mathbf{T} = [-d, 0, 0]^T$




- Therefore:
  - $\boldsymbol{E} = [\mathbf{T}_x]R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & d \\ 0 & -d & 0 \end{bmatrix}$

- For points $p\ (x, y, f)$ and $p'(x', y', f)$ in L & R coordinate systems:
  - $p'^T \boldsymbol{E} p = 0$
  - $[x', y', f\ ] \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & d \\ 0 & -d & 0 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ f \end{bmatrix} = 0$

$$[x', y', f\ ] \times \begin{bmatrix} 0 \\ df \\ -dy \end{bmatrix} = 0$$

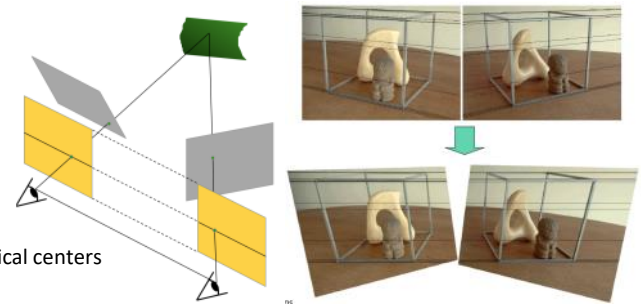
$$y'df - ydf = 0$$

$$y' = y$$

- Proven: For parallel cameras, image of any point must lie on same horizontal line in each image plane

---

## Rectification



- Since we know $R_l$ and $R_r$ (as they are defined by the calibrated cameras)
- We can **rectify** the image planes such that they are parallel
  - Epipoles are at infinity
  - Epipolar lines are parallel to horizontal axis

- Reproject image planes onto a common plane parallel to the line between optical centers
- Pixel motion is horizontal after this transformation
- 2 homographies (3x3 transforms), one for each input image reprojection

# Stereopsis & 3D Reconstruction

Monday, April 22, 2024 6:06 PM

## Basis

- For stereo reconstruction, we need to
  - Calibrate cameras (See Below)
  - Rectify images
  - **Compute disparity**
    - Finding correspondence - other than Epipolar Constraints, there are also "soft" constraints:
      - Similarity - the same interest point should be similar in both image
      - Uniqueness - a point in one image should only match to exactly one point in the other image
      - Ordering - points on the same surface (opaque object) will be in the same order in both views
  - Estimate depth from disparity

---

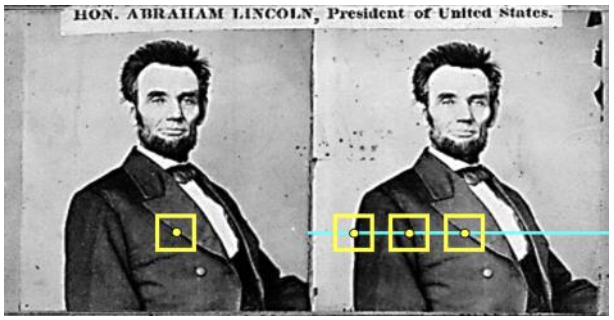
## Similarity

- When the two cameras have small changes and points are often similar

- **Dense Correspondence Search**
  - For each pixel in the first image
    - Find corresponding epipolar line in the other image
    - Examine all pixels on the epipolar line and pick the best match (e.g. SSD, correlation)
    - Triangulate the matches to get depth information
  - Easiest when epipolar lines are scanlines
    - Rectify image first
  - Window size needs to be:
    - Large enough to have sufficient intensity variation
    - Small enough to contain only pixels with about the same disparity

  ✓ Simple process
  ✓ More depth estimates which can be useful for surface reconstruction

  ☹ Breaks down textureless (flat) regions as well
  ☹ Raw pixel distances can be brittle
  ☹ Not robust for very different viewpoints

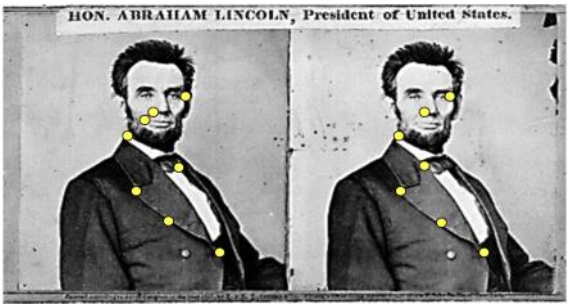




- **Sparse Correspondence Search**
  - Restrict search to sparse set of detected features
  - Feature descriptors and an associated feature distance instead of pixel values
  - Still uses the epipolar constraint to narrow the search further

  ✓ Efficiency
  ✓ Allow more reliable feature matches
    - Less sensitive to illumination than raw pixels

  ☹ Need to define and find "good" feature points
  ☹ Sparse information - less data points (compare to raw pixels)
    - Difficult to give lots of depth estimates

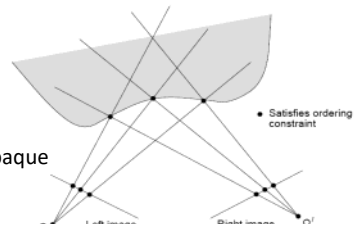


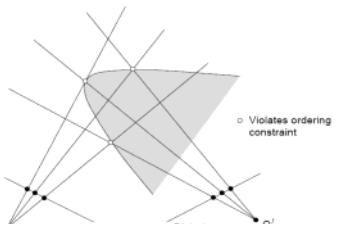

- Similarity constraints are less robust for
  - Untextured (flat) surfaces (e.g. background)
  - Occlusions

---

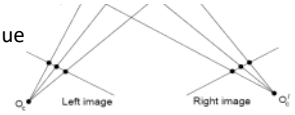
## Uniqueness & Ordering

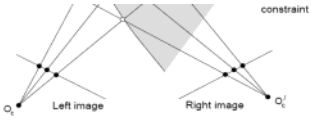
- Order - the order the matches were found in both viewpoints
  - Ordering should be the same for both images
    - Given that they are on the same surface and object is opaque
  - The ordering constraint is violated when object is transparent



    - Ordering should be the same for both images
        - Given that they are on the same surface and object is opaque
    - The ordering constraint is violated when object is transparent

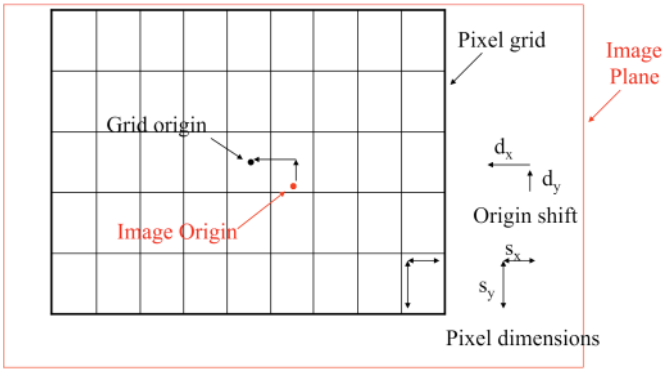


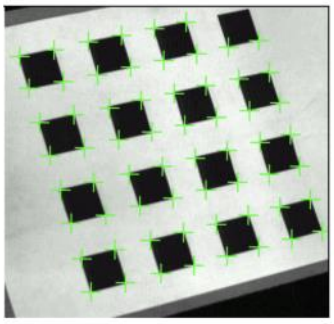

- Unique - there should only be one exact match for each point
- Possible source of errors
    - Low-contrast / textureless image regions
    - Occlusions
    - Camera calibration errors
    - Violations of brightness constancy (e.g. specular reflection)
    - Large motions

---

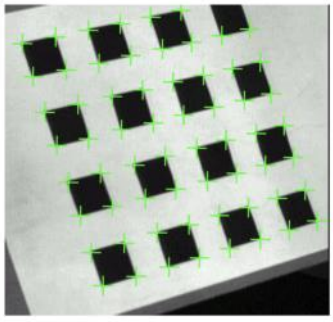
## Stereo Calibration




- Camera Parameters:
    - **Extrinsic parameters**
        - Rotation matrix $R$ (3x3)
        - Translation vector $\mathbf{T}$
    - **Intrinsic parameters** - related to camera sensors
        - Related pixel coordinates to image coordinates
        - Pixel size $(s_x, s_y)$ : pixel does not need to be squares
        - Origin offset $(d_x, d_y)$ : pixel origin may not be on optic axis
        - Focus length $f$
        - Not totally independent (Need $d_x, d_y, f, s_x/s_y$)
- We need to know camera parameters
    - $R, \mathbf{T}$ and $f$ to calculate the triangulation
    - $d_x, d_y, f, s_x/s_y$ to calculate image coordinates from pixel coordinates
- We can calculate these parameters if we know the scene coordinates of sufficient image points
- Calibration using target image
    - Accurately measured feature positions
    - Reliable location on images
    - E.g. Corners of squares can be located by edge finding
- Calibration algorithms compromises between
    - Accuracy of parameter estimation
    - Robustness of parameter estimation
    - Complexity of calculation
        - Least squares…non-linear optimizations
    - Engineering requirements of targets
        - Points on plane
        - Points throughout 3D volume



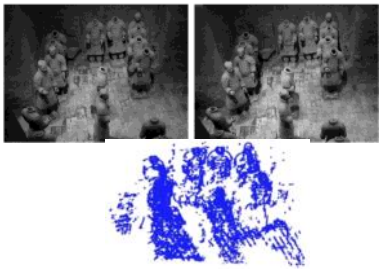
Left Image



Right Image

---

## Uncalibrated Stereo



- Calibration is necessary to determine absolute 3D positions
- But ***relative*** 3D positions (up to a scale factor) can be computed without calibration
- If at least 8 correspondences in the scene are known (avoiding "degenerate" configurations)
    - Sufficient camera parameters can be estimates
    - Similar to human stereopsis

---

## Summary

- Stereopsis can be provided accurate 3D measurements of positions
- Need to establish correspondences
    - Epipolar lines
    - Matching criteria - edges, corners, interest points

    - Constraints

- Calibration
    - Extrinsic parameters: $R, \mathbf{T}$
    - Intrinsic parameters: $d_x, d_y, f, s_x/s_y$

- Reconstruction by triangulation

- Depth application:
    - Segmentation
    - View interpolation
    - Virtual Viewpoints videos

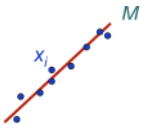
# Image Transformation

Tuesday, February 27, 2024 6:30 AM

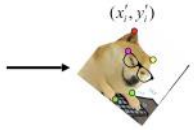
## Image Stitching Basis

Feature based Image alignment

- Stitching multiple images together to resemble bigger/more features in the images
  - Usually panorama images
  - NB: Wide cameras causes corners of each image to be darker
- Normalize images - illuminations, outside plane rotations, scales, etc.
- Find corresponding points and match - SIFT
- Feature-based alignments
  - Given a set of corresponding points
  - Find a model that fits the matches
    - Residual errors - distance




Find model $M$ that minimises $\sum_i \text{residual}(x_i, M)$

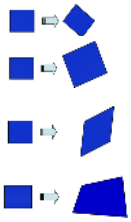

Find transformation $T$ that minimizes $\sum_i \text{residual}(T(x_i), x_i')$

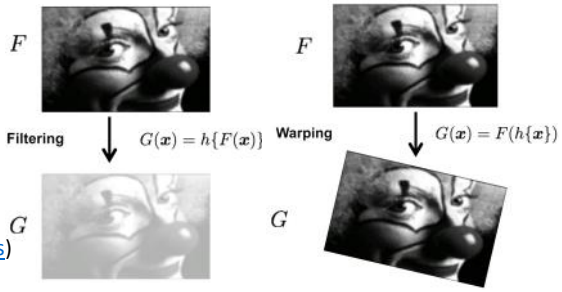
- Alignments - fitting a model to a transformation between pairs of features (matches) in 2 images

---

## Image Transformations

- 2D transformation models
  - Euclidean (translation + rotation)
    - On plane rotations
  - Similarity (translation + rotation + scaling)
  - Affine (translation + rotation + scaling + shear)
    - Shear = stretching
  - **Projective (Homography)**
    - Includes everything as well as out plane rotations



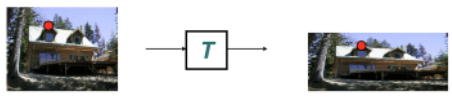



- Image transformation
  - Filtering - change pixel values (intensities / RGB values) (range of image functions)
  - Warping - change pixel locations (domain of image functions)

---

## Global Warping / Transformation

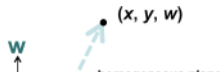

- $G(x) = F(h\{x\})$
- (Linear) Transformation $T$ is a coordinate-changing machine
  - $p' = T(p)$ or $\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix}$
- $T$ is **Global**, meaning that it:
  - Is the same for any point $p$ (i.e. apply to all pixels)
  - Can be described by just a few parameters
- All 2D linear transformations are combinations of
  - Scaling, Rotation, Shear, Mirror
  - $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} i & j \\ k & l \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$



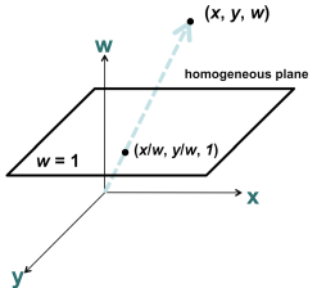

- Examples:
  - Scaling : $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$
  - Rotation (about origin): $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$
- Properties of Linear Transformation
  - Origin maps to origin
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratio are preserved - even in shearing (aspect ratio is preserved)
  - Closed under composition - several transformation matrices can be multiplied into 1 single transformation which has the same effect
- Note that translation is **NOT** a linear transformation on 2D coordinates
  - It is impossible to represent the translation in 2x2 matrix form for 2x1 coordinates
  - However it could be a linear transformation in other dimensions

---

## Affine Transformations



## Affine Transformations




- **Homogenous image coordinates** - Add one more coordinate
  - $(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
  - Represent 2D point with a 3D vector
    - 3D vectors are only defined up to scale
    - $\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow \left(\frac{x}{w}, \frac{y}{w}\right)$
- Translation can be represented as a Affine Linear Transformation in Homogenous
- Affine Transformations - 3x3 transformations with the last row $[0 \quad 0 \quad 1]$
- Basic Affine Transformations:

| Translation | Scaling | 2D in-plane rotation | Shear |
|---|---|---|---|
| $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ | $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ | $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ | $\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ |

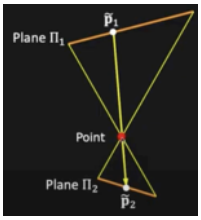
- Affine Transformations are combinations of
  - Linear Transformations (scaling, shearing, rotation, mirror) and
  - Translation

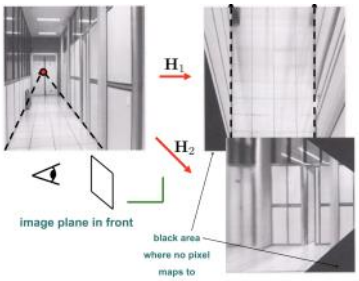
$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- Properties:
  - Same as 2D linear transformation but
  - **Origin does not necessarily map to origin**

---

## Homographies




- Aka Plane Projective Transformations / Planar perspective Maps
  - Aka "Mess with the $[0 \quad 0 \quad 1]$ row"
- The transformation matrix $H$ projects one plane to another plane through a **point**
  - $H = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$
- Combinations of **Affine transformations** and **Projective wraps**
- Properties:
  - **Origin does not necessarily map to origin**
  - Lines map to lines
  - **Parallel lines do not necessarily remain parallel**
  - **Ratios are not preserved**
  - Closed under composition

---

## Solving Homographies

- Homographies can be written in a normalized form for easier calculations:
  - $\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ where the vector $[h_{00}, h_{01}, \dots, h_{22}]$ has the length of 1
- Therefore for each pixel $i$, the $(x_i', y_i')$ can be calculated:
  - $x_i' = \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}}$
  - $y_i' = \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}$

$$\begin{aligned} x_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned} \longrightarrow \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1'x_1 & -x_1'y_1 & -x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y_1'x_1 & -y_1'y_1 & -y_1' \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_n'x_n & -x_n'y_n & -x_n' \\ 0 & 0 & 0 & x_n & y_n & 1 & -y_n'x_n & -y_n'y_n & -y_n' \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{A}_{2n \times 9} \quad \mathbf{h}_{9} \quad \mathbf{0}_{2n}$$

- The normalized form allows the formula to constraint from 9 unknowns to 8 unknowns (4 corresponding points)
  - Since we have pair points ($x$ and $x'$ are known)
  - The problem is constrained to a **least squares problem**

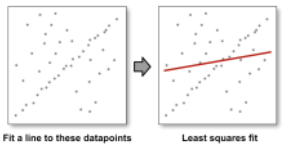
- Minimize $\|\mathbf{Ah} - 0\|^2$

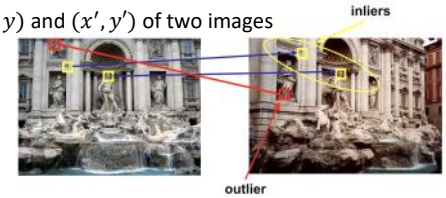
# Panoramas Stitching

Thursday, April 25, 2024 1:37 AM

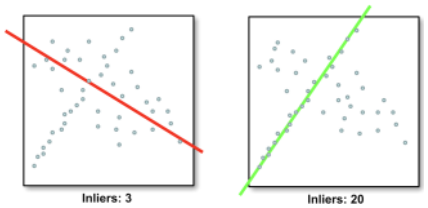
## Robust Feature Alignment (Dealing with outliers)

- Generally, it is extremely difficult to find correct correspondences between the pixels $(x, y)$ and $(x', y')$ of two images
- Image alignments is about finding the transformation $T$
  - Such that minimizes $\sum_i residual(T(x_i), x_i')$
  - Problem with linear regression is robustness
    - It is sensitive to outliers




- Solution Idea:
  - Given a hypothesized line
  - Count the number of points that "agree" with the line
    - "Agree" = within a small distance of the line
    - i.e. the **inliers** to that line
  - For all possible lines, select one with the largest number of inliers




---

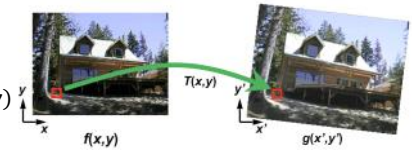
## Random Sample Consensus (RANSAC)

- If the set of Local Feature matches contains a very high percentage of outliers
- RANSAC idea:
  - All the inliers will agree with each other
  - The (hopefully small) number of outliers will (hopefully) disagree with each other
    - RANSAC only has guarantees if there are < 50% outliers (i.e. #inliers > #outliers)
- RANSAC loop:
  1. Randomly select $\boldsymbol{s}$ sample matches
     - $\boldsymbol{s}$ = minimum sample size that allows a transformation model to be fitted
     - For homographies, at least 4 points are needed
  2. Compute transformation $H$ (See homographies transformation) from sample group
  3. Apply $H$ and find **inliers** to this transformation
  4. If # inliers is sufficiently large
     i. re-compute least-squares estimate of transformation on all of the inliers to get new $H$
  5. Repeat $N$ times
  6. Keep the transformation with the **largest number of inliers**
- Number of rounds ($N$) related to percentage of expected outliers and prefered probability of success to guarantee
  - For 20% outliers, and we want to find correct answer with at least 99% probability

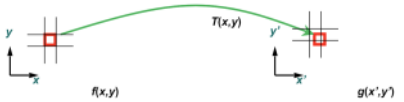
---

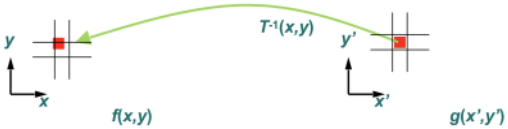
## Image Warping

- Given a coordinate transformation $(x', y') = T(x, y)$ and a source image $f(x, y)$
  - Compute a transformed image $g(x', y') = f(T(x, y))$



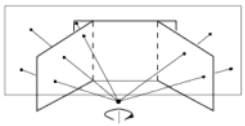

- Issues of directly applying the Homography
  - Pixels could mapped "between" multiple pixels in the other image
  - Solution: Splatting - Add "contribution" to pixels and normalize later
    - E.g. contribute intensities into 70%, 20%, 6%, 4% to the landed pixels
  - Can **still** results in holes (pixels map to undefined area / no mapping)
  - Solution: Take each of the location from the target image
    - Inverse transform the Homography and apply to source image
    - Resample **colour value** from interpolated (prefiltered) source image




---

## Panoramas



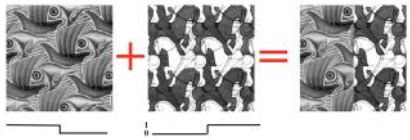
- As long as images are projected in the same plane
  - It is fine if locations are not exactly the same
- Given two (or more images):
  1. Detect Features

- It is fine if locations are not exactly the same

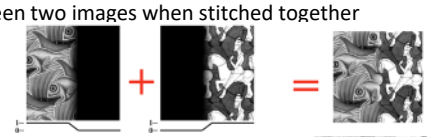
- Given two (or more images):
  1. Detect Features
  2. Match Features
  3. Compute homography using RANSAC
  4. Combine Images together (Image Blending)

---

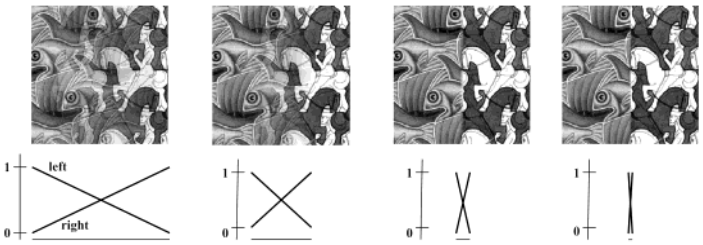
## Image Blending



- To reduce (intensity) difference between two images when stitched together
- Simple solution: Alpha Blending
  - $I_{blend} = \alpha I_{left} + (1 - \alpha) I_{right}$



- Effect of window size:
  - If the window size is too large = ghosting effect
    - The other image is somehow visible in the background
  - If window size is too small = seams
    - Looks as if there is no blending




- "Optimal" Window Size = smooth but not ghosted
  - Avoid Seams: window size = size of largest prominent feature
  - Avoid Ghosting: window size <= 2*size of smallest prominent feature

# Drone Self Localization

Monday, April 29, 2024 10:53 AM

## Basis

- Application of SIFT - Visual Self-localization of a drone
- Most drones have a GPS (Global Positioning System) sensor for them to detect their location
  - Or GNSS (Global Navigation Satellite System)
  - Gives longitude and latitude of the drone's location
- GPS have issues under some circumstances:
  - Poor or non-existent satellite visibility - e.g. in a valley or 'urban canyon'
  - Multipath reflections - signals bounce off buildings causing GPS sensor to miscalculate positions
  - GPS spoofing - satellite signals being received and re-transmitted after a delay
  - GPS signal blocking - a signal is transmitted at a higher power than the received satellite signals to prevent them from being received

---

## Visual Localization

- Attach cameras to drones
  - If the drone is level and camera is pointed directly downwards
    - Center of image shows the location of the drone

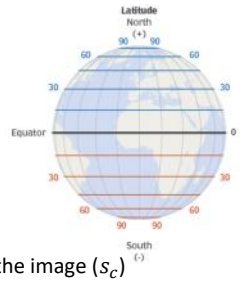
1. Information about the rough location of the drone is needed as a starting point
   - E.g. Launch position and information from other sensors (e.g. GPS a short time ago)
2. The image from the drone can then be compared with images from other databases or maps (e.g. Google Map)
   - E.g. using ORB feature detector

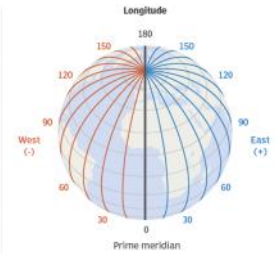
3. A homography (transformation matrix) $\mathbf{H}$ is then produced
   - $\mathbf{H}$ maps pixels from the drone image to the google image
4. Apply $\mathbf{H}$ to the drone image to get the location of the drone on the Google image
   - The red dot is the center of the drone image (location of the drone)

- If the registration (matching) was done accurately
  - The center of the drone and its corresponding center in the Google image should line up

---

## Location Calculations




- The latitude and longitude bounds of the Google image are:
  - Top: 48.920433°
  - Left: - 122.351681°
  - Bottom: 48.917700°
  - Right: - 122.342380°
- The drone image center mapped to 0.531 across the image ($s_r$) and 0.447 down the image ($s_c$)
  - Latitude = $I_T + s_r(I_B - I_T) = 48.920433 + 0.531(48.917700 - 48.920433) = 48.918982$
  - Longitude = $I_L + s_c(I_R - I_L) = -122.351681 + 0.447(-122.342380 - (-122.351681)) = -122.347523$
  - Therefore the calculated location is $(48.918982, -122.347523)$
- The actual location is $(48.918953, -121.652469)$
  - The calculation was 50.8 km away from the actual location
- GPS error on ground (lease signal) is 1~2m
  - Errors with tagging

---

## Summary

- This methods still has a lot of problems
  - Feature detector parameters need to be tuned for different images
  - Images can not always be matched
  - If the aerial image covers a larger area, more opportunities to find the drone image features in the wrong place

    - Many other difficulties such as weather conditions, not being perfectly downward-facing, features changing in images, etc.

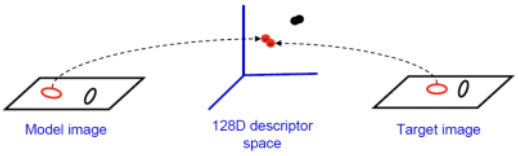
- So we still need GPS

# Indexing Local Features

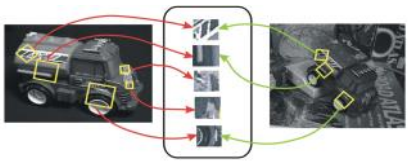
Monday, April 29, 2024 10:52 AM

## Basis

- In Local Features, each patch / region has a descriptor
  - which is a point in some high dimensional feature space (e.g. 128-D in SIFT)
  - In additional to size, orientation, and the 2D location of each patch




- Close points in feature space indicates similar local content
- True for both 3D reconstruction and retrieving images of similar objects

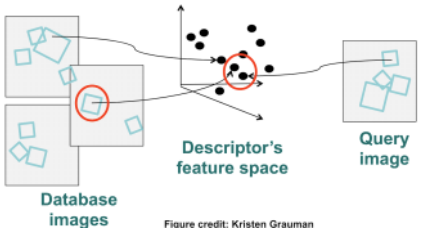


- Given hundreds to millions of images, where each image has thousands of features
  - How to efficiently find images that are relevant to a new image

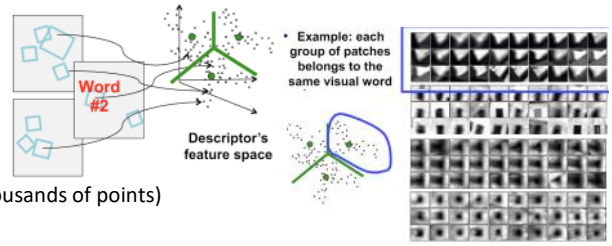
---

## Visual Words

- Inverted File Index
  - For text documents, an index is often used to find all **pages** on which a **word** occurs
    - We want to find all **images** where a **feature** occurs
    - In order to use this idea features need to be mapped to "**visual words**"



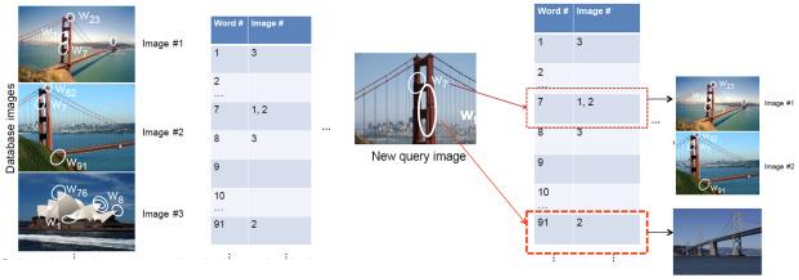

- Visual Words Main Idea:
  - Extract some local features from a number of images to a feature space
    - Each point in the feature space is a local descriptor (e.g. SIFT)
  - Close points in feature space means similar local content
    - Similar high-dimensional descriptors can be mapped to tokens/words
    - By quantizing / clustering the feature space
    - Quantize via clustering - let cluster centers be the prototype "words"
    - Determine which word to assign to each new image region
      - By finding the closest cluster center (instead of searching over thousands of points)




---

## Inverted File for Images of Visual Words

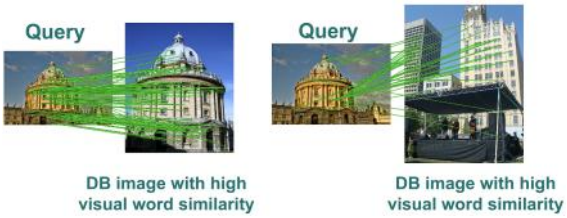
- For each "word" identified, store the occurrence of the word
- Database images are loaded into the index
  - Mapping words to image numbers
- New query image is mapped to indices that share a word
- Method:
  1. Extract words in query
  2. Inverted file index to find relevant frames / images
  3. Compare word counts - lots of common visual words indicate relevance
  4. Spatial Verification




---

## Spatial Verification

- Two images with many common visual words does not mean correct matches
- Spatial verification strategy - Generalized Hough Transform
  - Let each matched feature vote on location, scale, orientation of the model object
  - Verify parameters with enough vote




---

## Visual Vocabulary Formulation

- Things to consider:
  - Sampling strategies - where to extract features
    - Sparse interest points - for finding specific, textured objects

        - Multiple interest operators - offer more image coverage
        - Dense uniformly - offers better coverage for object categorization
        - Randomly

- Clustering / Quantization Algorithm
    - K-means, agglomerative clustering, mean-shift, etc.

- Unsupervised vs. supervised
- What corpus provides features (universal vocabulary)
- Vocabulary size, number of words

# Object Categorization

Friday, May 3, 2024 5:44 AM

## Basis

- Aim: Recognize ANY specific objects (cars, cats, etc.)
- Task Description: Given a small number of training images of a category
  - Recognize **a-priori unknown** instances of that category
    - Unsure whether or not present
  - Assign the correct category label

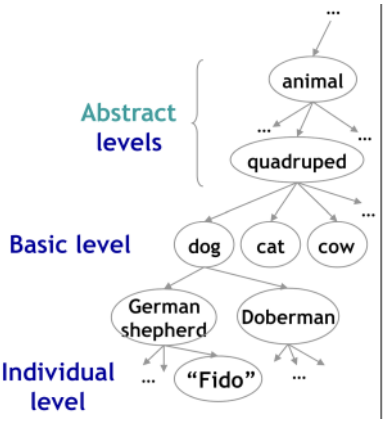
---

## Visual Object Categories

- **Basic-level Categories** in humans are
  - Defined predominately visually
  - Human (usually) start with basic-level categorization before identification
    - Basic-level categorization is easier and faster for human
    - Most promising starting point for visual classification
  - There are about 10k to 30k object categories
  - But objects from same categories can look different visually
- **Functional Categories**
  - E.g. chairs = "something you can sit on"
- **Ad-hoc Categories**
  - E.g. "Something you can find in an office environment"








---

## Challenges

- Robustness
  - Changes in illumination
  - Object pose
  - Clutter
  - Occlusions
  - Intra-class appearances
  - Viewpoint
- Learning with Minimal Supervision
  - The level of supervisions
    - Less: Unlabeled, multiple objects
    - Mid: Classes labeled, some clutter
    - More: Cropped to object, parts and classes labeled


Illumination

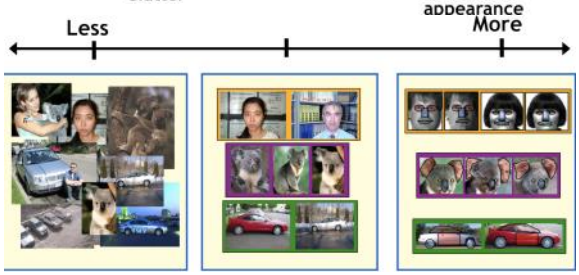

Object pose


Clutter


Occlusions


Intra-class appearance


Viewpoint




---

## Category Representation Motivation

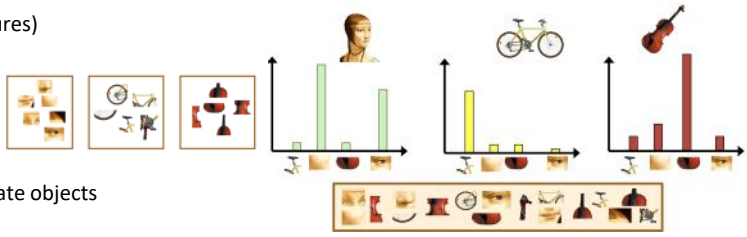
- If a local image region is a visual word, how to summarize an image (the document)?
- How to build a representation that is suitable for representing an entire category
  - Robust to intra-category variation, deformation, articulation, etc.
    - It needs to be general to include all instances of a category
  - But still discriminative

# Bag-of-Words Representation

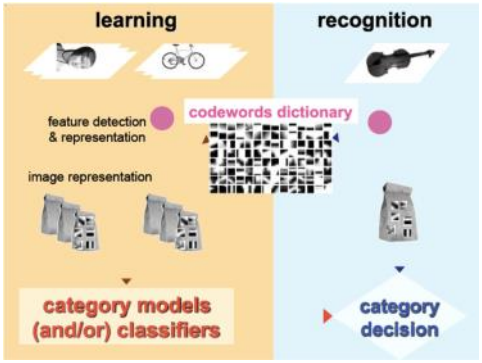
Friday, May 3, 2024 6:10 AM

## Basis

- An object can be represented by a bag of visual words (local features)
- Bag of Words (BoW) Definitions
  - Looser: Independent features
  - Stricter: Independent features & Histogram Representation
    - There would be common words between objects
    - Shape of the histogram is important to help differentiate objects



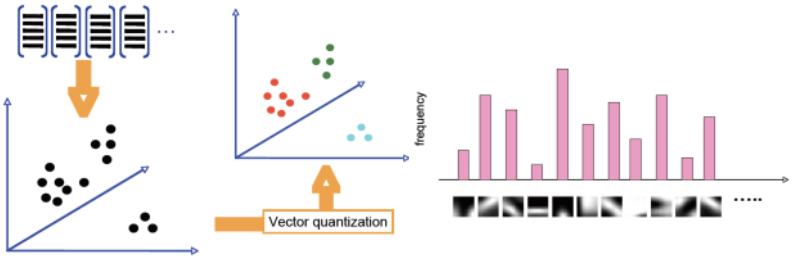
- Allows summarization of entire images based on the distribution (histogram) of word occurrences
  - From images that contain thousands of local features -> fixed vector (#features, frequencies)
  - Allows comparison between images (vectors)
- BoW Model Overview Steps:
  1. Feature Detection & Representation
     - Detect local features and represent using descriptors (SIFT)
  2. Create Codewords Dictionary
     - Cluster feature space and form visual words
  3. Image Representation
     - Put visual words into "bags"
     - For each word in the bag, build histogram
  4. Build Category Models
     - Image Classification
  5. Repeat 1 to 3 to query images and use 4 to make category decision




---

## Learning Phase

1. Feature Detection & Representation
   - Capture regular grid and represent using histograms / Capture interest points and represent using features
   - (Normalize) and compute a collection of the descriptors
2. Code-words Dictionary Formulation
   - Each descriptor represents a point in the feature space
   - Quantize the feature space to form visual words
   - Create a dictionary of the visual words
3. Image Representation
   - Each image can be represented as vectors
     - Categories and frequencies

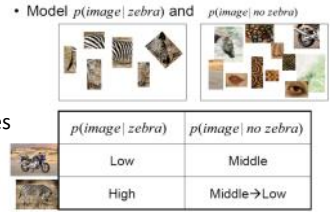


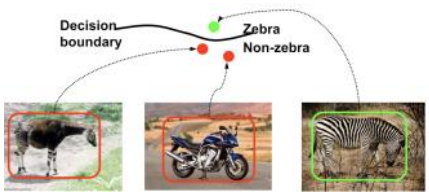

- BoW representations makes it possible to describe the unordered point set with a single vector
  - Vectors are in fixed dimension across the image examples
  - Allow comparison
    - Frames can be ranked by normalized scalar product between their (possibly weighted) occurrence counts
- Provides easy way to use distribution of feature types with various learning algorithms requiring vector inputs
- Works pretty well for image-level classification

---

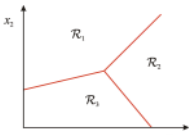
## Category Models / Classifiers

- Task: Given the bag-of-features representations of images from different classes
  - Learn a model for distinguishing them
- Generative Methods
  - Given samples of images belonging to different classes
  - Learn the likelihood of a image belonging to different classes
  - E.g. Naïve Bayes
- Discriminative Methods
  - Learn a decision rule (classifier) to assign BoW representations of images to different classes
  - E.g. KKN



## Discriminative - KNN



- Assign input vector to one of two or more classes
  - Vectors (#visual words, histogram)
- Any decision rule divides input space into decision regions separated by decision boundaries

- Nearest Neighbours (NN)
  - Assign label of nearest training data point to each test data point
  - Sensitive to noise

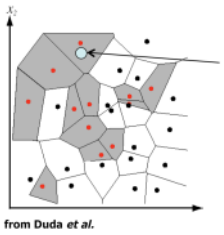
- K-Nearest Neighbours (KNN)
  - For a new point, find the k closest points from the training data
  - Use the labels of the k points to "vote" for category

  ✓ Simple to implement
  ✓ Flexible to feature / distance choices
  ✓ Naturally handles multi-class cases
  ✓ Can do well in practice with enough representative data

  ☹ Large search problem to find nearest neighbours
  ☹ Storage of data
  ☹ Must know a meaningful distance function

- Other discriminative classifiers
  - Boosting
  - Support vector machines (SVMs)

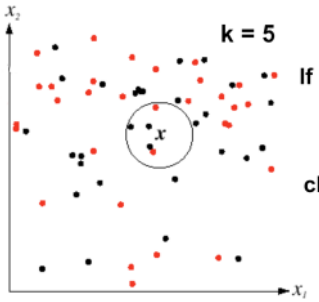
Black = negative
Red = positive



Novel test example

Closest to a positive example from the training set, so classify it as positive.

from Duda *et al.*




If query lands here, the 5 NN consist of 3 negatives and 2 positives, so we classify it as negative.

---

## Generative - Naïve Bayes

- Assume that each feature is conditionally independent given the class
  - $p(\mathbf{x}|c) = p(x_1, \dots, x_N|c) = \prod_{i=1}^{N} p(x_i|c)$
- The map decision would be the one with the maximum likelihood
  - Or the maximum of prior probability of the object classes multiplied by the likelihood of the $i^{th}$ visual word given the class
    - $c^* = \text{argmax}_c \left( p(c) * \prod_{i=1}^{N} p(x_i|c) \right)$
    - Likelihood - Estimated by the empirical frequencies of visual words in images from a given class
    - Prior (dist. probability among all class) - Can be uniform or based on knowledge (e.g. prob of a polar bear on a motor way is low)

- Classify image using histograms of occurrences on visual words
- If only present/absent of a word is taken into account:
  - $x_i \in \{0,1\}$


---

## Spatial Information

- A BoW is an **order-less** representation
  - Throwing out spatial relationships between features
  - As long as distinctive and sufficient features are present, does not care how disformed the object is
    - E.g. BoW is likely to classify Picasso's art into human faces just like real faces

- Middle ground:
  - Visual "phrases" - frequently co-occurring words
  - Semi-local features - describe configuration, neighborhood for "votes"
  - Let position be part of each features - but loss generality
  - Count bags of words only within sub-grids of an image - preserved the global spatial information
  - After matching, verify spatial consistency - e.g. check if neighbours are similar too

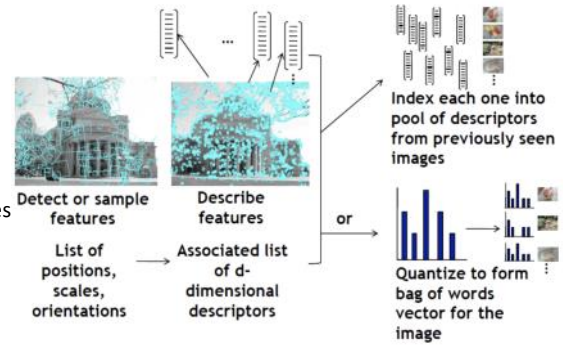
---

## Summary

- Indexing features

# Summary

- Indexing features
  1. Detect or sample features
     - Get list of positions, scales, orientation
  2. Describe features
     - Get associated list of d-dimensional descriptors
  3. Either
     i. Index each one into pool of descriptors from previously seen images
     ii. Quantize to form BoW vector for the image




- Bag of Words
  - Quantize feature space to make discrete sets of visual words
  - Summarize images by distribution of words
  - Indexing individual words
  - Inverted index - pre-compute index to enable faster search at query time

  ✓ Flexible to geometry / deformations / viewpoints
  ✓ Compact summary of image content
  ✓ Provides vector representation for sets
  ✓ Empirically good recognition results in practice

  ☹ Basic model ignores geometry - must verify afterwards or encode via features
  ☹ Background and foreground mixed when bags cover whole image
  ☹ Interest points or sampling - no guarantee to capture object-level parts
  ☹ Optimal vocabulary formation remains unclear

- Local Invariant Features
  - Distinctive matches possible in spite of significant view change
    - Useful not only to provide matches for multi-view geometry but also to find objects and scenes
  - To find correspondences among detected features
  - Measure distance between descriptors and look for most similar patches

# Essential Reading

Friday, May 3, 2024 5:43 AM

csurka-eccv-04

---

## Summary

- **Aim** - Generalized Visual Categorization using "Bags of Keypoints"

- **Method**:
  1. Detection and Description of Image Patches
     - i. Uses Harris Affine Detection - scale invariant detection insufficient to deal with affine transformation
     - ii. Uses SIFT as Descriptors
       - Stabler with noise than higher Gaussian derivatives
       - Simple Euclidean distance can be used with the SIFT covariance matrices
       - 128 components for each features - potentially richer and discriminative representation

  2. Vector Quantization
     - i. Uses K-means to quantize vectors into a vocabulary
       - K is selected through experiment instead automatic process
         - Since there is no prior knowledge of density and compactness of clusters
         - Not interested in "correct" clusters, but accurate categorization
     - ii. Assign descriptors to clusters

  3. Bag of Keypoints Construction
     - i. Create histogram counting occurrences of each "words"

  4. Classification
     - i. Uses Naïve Bayes
       - Assumes feature independence and classifies based on the maximum **posterior** (not just likelihood and prior)probability
       - Uses Laplace Smoothing to avoid probabilities of 0

     - ii. Uses Support Vector Machine
       - Find a hyperplane that best separates different classes in the feature space
         - Optimal hyperplane maximizes margin
           - Margin - distance between hyperplane and nearest data point from each class (support vectors)
       - Uses kernel to transform data to higher space for linear separation and C as penalty for misclassification
         - Both need to be determined through experiments

       - Uses One-Against All (OvA) Approach to reduce multi-class classification to binary classification
         - Train a SVM for each class

- **Experiment and Results**
  - 7 categories: cars, faces, buildings, books, phones, trees, bikes
  - SVM > Naïve Bayes particularly in images with significant degree of background clutter and class variability
  - Both works well
    - Achieves good categorization accuracy
    - Is robust against background clutter
    - Does not require geometric information about the object's positions

---

# Visual Categorization with Bags of Keypoints

Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, Cédric Bray

Xerox Research Centre Europe
6, chemin de Maupertuis
38240 Meylan, France
{gcsurka,cdance}@xrce.xerox.com

**Abstract.** We present a novel method for generic visual categorization: the problem of identifying the object content of natural images while generalizing across variations inherent to the object class. This *bag of keypoints* method is based on vector quantization of affine invariant descriptors of image patches. We propose and compare two alternative implementations using different classifiers: Naïve Bayes and SVM. The main advantages of the method are that it is simple, computationally efficient and intrinsically invariant. We present results for simultaneously classifying seven semantic visual categories. These results clearly demonstrate that the method is robust to background clutter and produces good categorization accuracy even without exploiting geometric information.

## 1. Introduction

The proliferation of digital imaging sensors in mobile phones and consumer-level cameras is producing a growing number of large digital image collections. To manage such collections it is useful to have access to high-level information about objects contained in the image. Given an appropriate categorization of image contents, one may efficiently search, recommend, react to or reason with new image instances.

We are thus confronted with the problem of *generic visual categorization*. We should like to identify processes that are sufficiently generic to cope with many object types simultaneously and which are readily extended to new object types. At the same time, these processes should handle the variations in view, imaging, lighting and occlusion, typical of the real world, as well as the intra-class variations typical of semantic classes of everyday objects.

The task-dependent and evolving nature of visual categories motivates an example-based machine learning approach. This paper presents a *bag of keypoints* approach to visual categorization. A bag of keypoints corresponds to a histogram of the number of occurrences of particular image patterns in a given image. The main advantages of the

method are its simplicity, its computational efficiency and its invariance to affine transformations, as well as occlusion, lighting and intra-class variations.

It is important to understand the distinction of visual categorization from three related problems:

*Recognition:* This concerns the identification of particular object instances. For instance, recognition would distinguish between images of two structurally distinct cups, while categorization would place them in the same class.

*Content Based Image Retrieval:* This refers to the process of retrieving images on the basis of low-level image features, given a query image or manually constructed description of these low-level features. Such descriptions frequently have little relation to the semantic content of the image.

*Detection:* This refers to deciding whether or not a member of *one visual category* is present in a given image. Most previous work on detection has centered on machine learning approaches to detecting faces, cars or pedestrians [1]-[6] While it would be possible to perform generic categorization by applying a detector for each class of interest to a given image, this approach becomes inefficient given a large number of classes. In contrast to the technique proposed in this paper, most existing detection techniques require precise manual alignment of the training images and the segregation of these images into different views [5], neither of which is necessary in our approach.

Our *bag of keypoints* approach can be motivated by an analogy to learning methods using the *bag-of-words* representation for text categorization [7]-[10]. The idea of adapting text categorization approaches to visual categorization is not new. Zhu et al [11] investigated the vector quantization of small square image windows, which they called *keyblocks*. They showed that these features produced more "semantics-oriented" results than color and texture based approaches, when combined with analogues of the well-known vector-, histogram-, and n-gram-models of text retrieval. In contrast to our approach, their keyblocks do not possess any invariance properties.

The idea of clustering invariant descriptors of image patches has previously been applied to the problem of texture classification [12]-[14]. Clearly the problem of texture classification is rather different from that of generic categorization. Therefore it is natural that these approaches differ from ours. While our method uses clustering to obtain quite high-dimensional feature vectors for a classifier, these texture recognizers use clustering to obtain relatively low-dimensional histograms and evaluate the similarity of these histograms to previously seen probability densities. In [12]-[13] filter responses are clustered and the recognition is done using the closest model measured by a $\chi^2$ test. Lazebnik *et al* [14] cluster affine invariant interest points in each image individually and summarize the distribution of the descriptors in form of a signature composed of representative cluster members and weights proportional to cluster sizes. Signatures of different images are compared using the Earth Mover's Distance [15].

Recently Fergus *et al* [16] proposed a visual categorization method based on invariant descriptors of image patches. Their method exploits a probabilistic model that combines likelihoods for appearance, relative scale and position, as well as a model of

the statistics of their patch detector. This elegant approach has a number of limitations. Firstly the method is not efficient: even when models are restricted to 6 image patches and training images only contain up to 30 patches, days of CPU time are required to learn several categories. Secondly, views of objects used for training must be segregated, for instance into cars (rear) and cars (side). This is unsurprising given the use of an explicit 2D model of relative positions.

In section 2 we explain the categorization algorithms and the choice of their components. In section 3 we present results from applying of the algorithms to the dataset of Fergus *et al* and to a more challenging seven class dataset. We demonstrate that our approach is robust to the presence of background clutter and produces state-of-the-art recognition performance.

## 2. The method

The main steps of our method are:

- Detection and description of image patches
- Assigning patch descriptors to a set of predetermined clusters (a *vocabulary*) with a vector quantization algorithm
- Constructing a *bag of keypoints*, which counts the number of patches assigned to each cluster
- Applying a multi-class classifier, treating the bag of keypoints as the feature vector, and thus determine which category or categories to assign to the image.

Ideally these steps are designed to maximize classification accuracy while minimizing computational effort. Thus, the descriptors extracted in the first step should be invariant to variations that are irrelevant to the categorization task (image transformations, lighting variations and occlusions) but rich enough to carry enough information to be discriminative at the category level. The vocabulary used in the second step should be large enough to distinguish relevant changes in image parts, but not so large as to distinguish irrelevant variations such as noise.

We refer to the quantized feature vectors (cluster centres) as “keypoints” by analogy with “keywords” in text categorization. However, in our case “words” do not necessarily have a repeatable meaning such as “eyes”, or “car wheels”, nor is there an obvious best choice of vocabulary. Rather, our goal is to use a vocabulary that allows good categorization performance on a given training dataset. Therefore the steps involved in training the system allow consideration of multiple possible vocabularies:

- Detection and description of image patches for a set of labeled training images
- Constructing a set of vocabularies: each is a set of cluster centres, with respect to which descriptors are vector quantized.
- Extracting bags of keypoints for these vocabularies
- Training multi-class classifiers using the bags of keypoints as feature vectors

- Selecting the vocabulary and classifier giving the best overall classification accuracy.

We now discuss the choices made for each step in more detail.

### 2.1 Feature extraction

In computer vision, local descriptors (i.e. features computed over limited spatial support) have proved well-adapted to matching and recognition tasks, as they are robust to partial visibility and clutter. Such tasks require descriptors that are repeatable. Here, we mean repeatable in the sense that if there is a transformation between two instances of an object, corresponding points are detected and (ideally) identical descriptor values are obtained around each. This has motivated the development of several scale and affine invariant point detectors, as well as descriptors that are resistant to geometric and illumination variations [17]-[21].

It was shown in [21] that if we have affine transformation between two images a scale invariant point detector is not sufficient to have the stability of the point's location. Therefore we preferred to work with the *Harris affine detector* described in [21]. However, the reader should be aware that the benefits of this choice are not clear cut: firstly because most real world objects have three-dimensional structures whose variations are not well captured by affine transformations; secondly since attempts to increase the invariance of a feature typically result in a loss of discriminative information.

Harris affine points are detected by an iterative process. Firstly, positions and scales of interest points are determined as local maxima (in position) of a scale-adapted Harris function, and as local extrema in scale of the Laplacian operator. Then an elliptical (i.e. affine) neighborhood is determined. This has a size given by the selected scale and a shape given by the eigenvalues of the image's second moment matrix. The selection of position/scale and the elliptical neighborhood estimation are then iterated and the point is kept only if the process converges within a fixed number of iterations.

The affine region is then mapped to a circular region, so normalizing it for affine transformations. Scale Invariant Feature Transform (SIFT) descriptors [18] are computed on that region. SIFT descriptors are multi-image representations of an image neighborhood. They are Gaussian derivatives computed at 8 orientation planes over a 4x4 grid of spatial locations, giving a 128-dimension vector. Fig. 1 shows an example of the maps of gradient magnitude corresponding to the 8 orientations.

**Fig. 1.** (*From left to right)* A Harris affine region; the normalized region; and the 8 maps of gradient magnitude constituting the SIFT descriptor.

We prefer SIFT descriptors to alternatives such as steered Gaussian derivatives or differential invariants of the local jet for the following reasons:

1. They are simple linear Gaussian derivatives. Hence we expect them to be more stable to typical image perturbations such as noise than higher Gaussian derivatives or differential invariants.
2. The use of a simple Euclidean metric in the feature space seems justified. In the case of differential invariants obtained by a combination of the components of the local jet, the use of a Mahalanobis distance is more appropriate. For instance, one would expect a second derivative feature to have a higher variance than a first derivative. Selecting an appropriate Mahalanobis distance a priori seems challenging. It would not be appropriate to use the covariance matrix of SIFT descriptors over the entire dataset, since this is predominantly influenced by inter-class variations (or more precisely, by variations between keypoints that we do not wish to ignore). Measuring a Mahalanobis distance would probably requiring manual specification of multiple homologous matching points between different images of objects of the same category, seriously working against our objective of producing a simple and automated categorization system.
3. There are far more components to these feature vectors (128 rather than 12 to 16). Hence we have a richer and potentially more discriminative representation.

Recently Mikolajczyk *et al* [22] have compared several descriptors for matching and found that SIFT descriptors perform best.

### 2.3 Visual vocabulary construction

In our method, the vocabulary is a way of constructing a feature vector for classification that relates "new" descriptors in query images to descriptors previously seen in training. One extreme of this approach would be to compare each query descriptor to all training descriptors: this seems impractical given the huge number of training descriptors involved (over 600 000 for our data set). Another extreme would be to try to identify a small number of large "clusters" that are good at discriminating a given class: for instance [16] operates with 6 parts per category. In practice we find that the best tradeoffs of accuracy and computational efficiency are obtained for intermediate sizes of clustering.

Most clustering or vector quantization algorithms are based on iterative square-error partitioning or on hierarchical techniques. Square-error partitioning algorithms attempt to obtain the partition which minimizes the within-cluster scatter or maximizes the between-cluster scatter. Hierarchical techniques organize data in a nested sequence of groups which can be displayed in the form of a dendrogram or a tree. They need

some heuristics to form clusters and hence are less frequently used than square-error partitioning techniques in pattern recognition.

We chose to use the simplest square-error partitioning method: *k*-means [23]. This algorithm proceeds by iterated assignments of points to their closest cluster centers and recomputation of the cluster centers. Two difficulties are that the k-means algorithm converges only to local optima of the squared distortion, and that it does not determine the parameter *k*. There exist methods allowing to automatically estimating the number of clusters. For example, Pelleg et al [24] use cluster splitting to do it, where the splitting decision is done by computing the Bayesian Information Criterion. However, in the present case we do not really know anything about the density or the compactness of our clusters. Moreover, we are not even interested in a "correct clustering" in the sense of feature distributions, but rather in accurate categorization. We therefore run *k*-means several times with different number of desired representative vectors (*k*) and different sets of initial cluster centers. We select the final clustering giving the lowest empirical risk in categorization [25].

### 2.3 Categorization

Once descriptors have been assigned to clusters to form feature vectors, we reduce the problem of generic visual categorization to that of multi-class supervised learning, with as many classes as defined visual categories. The categorizer performs two separate steps in order to predict the classes of unlabeled images: training and testing. During training, labeled data is sent to the classifier and used to adapt a statistical decision procedure for distinguishing categories. Among many available classifiers, we compared the Naïve Bayes classifier for its simplicity and its speed, and the Support Vector Machine since is it often known to produce state-of-the-art results in high-dimensional problems.

#### 2.3.2 Categorization by Naïve Bayes

Naïve Bayes [26] is a simple classifier used often in text categorization. It can be viewed as the maximum *a posteriori* probability classifier for a generative model in which: 1) a document category is selected according to class prior probabilities; 2) each word in the document is chosen *independently* from a multinomial distribution over words specific to that class. While independence is a *naïve* assumption, the accuracy of the Naïve Bayes classification is typically high [27].

Now, considering visual categorization, assume we have a set of labeled images $I=\{\mathbf{I}_i\}$ and a vocabulary $V=\{\mathbf{v}_i\}$ of representative keypoints (i.e. cluster centers). Each descriptor extracted from an image is labeled with the keypoint to which it lies closest in feature space. We count the number N(t,i) of times keypoint $\mathbf{v}_i$ occurs in image $\mathbf{I}_i$. To categorize a new image, we apply Bayes's rule and take the largest *a posteriori* score as the prediction:

$$P(C_j \mid I_i) \propto P(C_j)P(I_i \mid C_j) = P(C_j)\prod_{t=1}^{|V|} P(v_t \mid C_j)^{N(t,i)}. \quad (1)$$

It is evident in this formula that Naïve Bayes requires estimates of the class-conditional probabilities of keypoint $\mathbf{V_t}$ given category $\mathbf{C_j}$. In order to avoid probabilities of zero, these estimates are computed with Laplace smoothing:

$$P(v_t \mid C_j) = \frac{1 + \sum_{\{I_i \in C_j\}} N(t, i)}{|V| + \sum_{s=1}^{|V|} \sum_{\{I_i \in C_j\}} N(s, i)}. \quad (2)$$

### 2.3.2 Categorization by SVM

The SVM classifier finds a hyperplane which separates two-class data with maximal *margin* [25]. The margin is defined as the distance of the closest training point to the separating hyperplane. For given observations **X**, and corresponding labels **Y** which takes values $\pm 1$, one finds a classification function:

$$f(x) = \mathbf{sign}(w^T x + b) \quad (3)$$

where **w, b** represents the parameters of the hyperplane.

Data sets are not always linearly separable. The SVM takes two approaches to this problem. Firstly it introduces an error weighting constant ***C*** which penalizes misclassification of samples in proportion to their distance from the classification boundary. Secondly a mapping $\Phi$ is made from the original data space of **X** to another feature space. This second feature space may have a high or even infinite dimension. One of the advantages of the SVM is that it can be formulated entirely in terms of scalar products in the second feature space, by introducing the *kernel*

$$K(u, v) = \Phi(u) \cdot \Phi(v) \quad (4)$$

Both the kernel **K** and penalty ***C*** are problem dependent and need to be determined by the user.

In the kernel formulation, the decision function can be expressed as

$$f(x) = \mathbf{sign}(\sum_i y_i \alpha_i K(x, x_i) + b). \quad (5)$$

where $\mathbf{x_i}$ are the training features from data space ***X*** and $\mathbf{y_i}$ is the label of $\mathbf{x_i}$. Here the parameters $\alpha_i$ are typically zero for most *i*. Equivalently, the sum can be taken only over a select few of the $\mathbf{x_i}$. These feature vectors are known as *support vectors.* It can be shown that the support vectors are those feature vectors lying nearest

to the separating hyperplane. In our case, the input features $\mathbf{x}_i$ are the binned histograms formed by the number of occurrences of each keypoint $\mathbf{v}_i$ from the vocabulary V in the image $\mathbf{I}_i$.

In order to apply the SVM to multi-class problems we take the one-against-all approach. Given an *m*-class problem, we train *m* SVM's, each distinguishes images from some category *i* from images from all the other *m-1* categories *j* not equal to *i*. Given a query image, we assign it to the class with the largest SVM output.

## 3 Experiments

We present results from three experiments. In the first we explore the impact of the number of clusters on classifier accuracy and evaluate the performance of the Naïve Bayes classifier. We then explore the performance of the SVM on the same problem. The first two experiments were conducted on an in-house seven class dataset. In the last experiment we describe results on the four class dataset employed in [16].

Our in-house database contains 1776 images in seven classes[1]: faces, buildings, trees, cars, phones, bikes and books. Fig. 2 shows some examples from this dataset.

**Fig. 2.** Example images from our in-house database

It is a challenging dataset, not only because of the large number of classes, but also because it contains images with highly variable poses and significant amounts of

[1] We hope to make this dataset publicly available soon. It contains 792 faces, 150 buildings, 150 trees, 201 cars, 216 phones, 125 bikes and 142 books.

background clutter, sometimes presence of objects from multiple classes although a large proportion of each image's area is occupied by the target category. The images have resolutions between 0.3 and 2.1 megapixels and were acquired with a diverse set of cameras. The images are color but only the luminance component is used in our method. These were gathered by XRCE and Graz University, excepting faces which were downloaded from the Web.

We used three performance measures to evaluate our multi-class classifiers.

– The confusion matrix:

$$M_{ij} = \frac{\left|\left\{\mathbf{I_k} \in \mathbf{C_j} : \mathbf{h}(\mathbf{I_k}) = \mathbf{i}\right\}\right|}{\left|\mathbf{C_j}\right|} \tag{6}$$

where $\mathbf{i, j} \in \{\mathbf{1},...,\mathbf{N_c}\}$, $\mathbf{C_j}$ is the set of test images from category $\mathbf{j}$ and $\mathbf{h}(\mathbf{I_k})$ is the category which obtained the highest classifier output for image $\mathbf{I_k}$.

– The overall error rate:

$$R = \mathbf{1} - \frac{\sum_{\mathbf{j=1}}^{\mathbf{N_c}} \left|\mathbf{C_j}\right| M_{jj}}{\sum_{\mathbf{j=1}}^{\mathbf{N_c}} \left|\mathbf{C_j}\right|} \tag{7}$$

– The mean ranks. These are the mean position of the correct labels when labels output by the multi-class classifier are sorted by the classifier score.

Each performance metric was evaluated with 10-fold cross validation.

### 3.2 Naïve Bayes Results

In Fig. 3 we present the overall error rates using Naïve Bayes as a function of the number of clusters $k$. Each point in Fig. 3 is the best of 10 random trials of $k$-means. The standard-error on the maximum is in the range [1, 3]%.[2] The error rate only improves slightly as we move from $k$ = 1000 to $k$ = 2500. We therefore assert that $k$ = 1000 present a good trade-off between accuracy and speed[3].

[2] That is, if the experiment of taking the maximum of 10 is repeated multiple times with different random clusterings, the standard deviations of the result for the different classes are in the range [1,3]%.

[3] It takes about one minute to get predicted labels on the whole database using the Naive Bayes classifier with $k$ = 1000 on a 2 GHz processor.

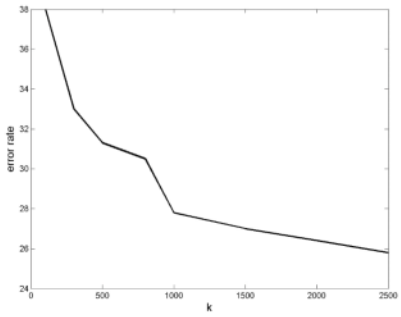



**Fig. 3.** The lowest overall error rate (percentage) found for different choices of *k*

Table 1 shows the performance as a function of category obtained with this *k*.

**Table 1.** Confusion matrix and the mean rank for the best vocabulary (*k=1000).*

| True classes → | ***faces*** | ***buildings*** | ***trees*** | ***cars*** | ***phones*** | ***bikes*** | ***books*** |
|---|---|---|---|---|---|---|---|
| ***faces*** | **76** | 4 | 2 | 3 | 4 | 4 | 13 |
| ***buildings*** | 2 | **44** | 5 | 0 | 5 | 1 | 3 |
| ***trees*** | 3 | 2 | **80** | 0 | 0 | 5 | 0 |
| ***cars*** | 4 | 1 | 0 | **75** | 3 | 1 | 4 |
| ***phones*** | 9 | 15 | 1 | 16 | **70** | 14 | 11 |
| ***bikes*** | 2 | 15 | 12 | 0 | 8 | **73** | 0 |
| ***books*** | 4 | 19 | 0 | 6 | 7 | 2 | **69** |
| ***Mean ranks*** | 1.49 | 1.88 | 1.33 | 1.33 | 1.63 | 1.57 | 1.57 |

In the clustering process, there is a risk of bias since images from different categories contain different numbers of interest points. We therefore used random samples of the training data in the clustering step, each sample containing 5000 interest points randomly chosen from each class (there are around 640 000 interest points extracted from 1776 training images).

Fig. 4 illustrates example clusters obtained for the selected “best” vocabulary.

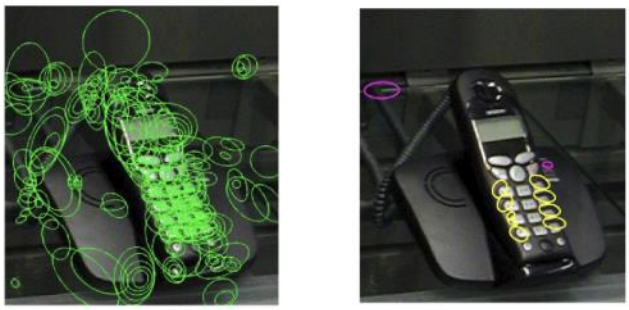


**Fig. 4.** *Left* all patches detected for this image. *Right* patches from two selected clusters occurring in this image (yellow and magenta ellipses).

The algorithm handles easily multiple objects in the same images (see Fig. 5), occlusion, partial view, any orientation (Fig. 6).

**Fig. 5.** Images correctly classified containing multiple objects of the same category.

**Fig. 6.** Profile face, partial view of a car, roof of a house correctly classified as face, cars, building.

In Fig. 7 we show examples where images were well classified even if most of the detected interest points were on the "background".

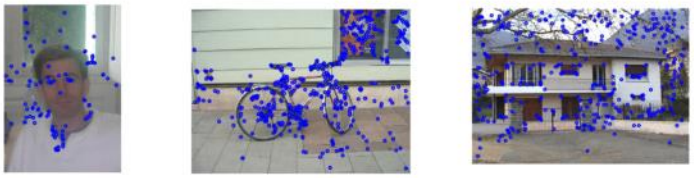


**Fig. 7.** Images where background clutter is in a higher percentage than interest points on the object.

Fig. 8 shows some images from the database where objects from other categories were also present in the image and the first 3 ranked categories for each of them. These images were not considered as multiple labels but labeled by the main object in the image.


**Fig. 8.** Images where multiple objects were present and the first three ranked labels.

Finally, Fig. 9 shows some false alarm with the first label and the true label with its rank.

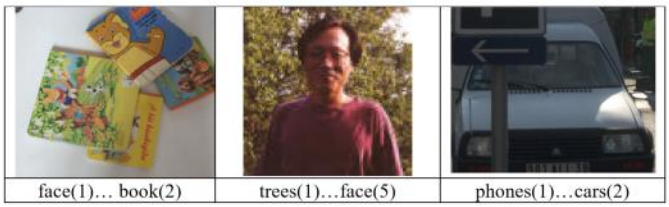



**Fig. 9.** Examples of incorrectly ranked images. The correct label's rank is also shown.

### 3.2 SVM Results

Results from applying the SVM are given in Table 2.

**Table 2.** Confusion matrix and mean rank for SVM (*k=1000, linear kernel).*

| True classes → | *faces* | *buildings* | *trees* | *cars* | *phones* | *bikes* | *books* |
|---|---|---|---|---|---|---|---|
| ***faces*** | **98** | 14 | 10 | 10 | 34 | 0 | 13 |
| ***buildings*** | 1 | **63** | 3 | 0 | 3 | 1 | 6 |
| ***trees*** | 1 | 10 | **81** | 1 | 0 | 6 | 0 |
| ***cars*** | 0 | 1 | 1 | **85** | 5 | 0 | 5 |
| ***phones*** | 0 | 5 | 4 | 3 | **55** | 2 | 3 |
| ***bikes*** | 0 | 4 | 1 | 0 | 1 | **91** | 0 |
| ***books*** | 0 | 3 | 0 | 1 | 2 | 0 | **73** |
| ***Mean ranks*** | 1.04 | 1.77 | 1.28 | 1.30 | 1.83 | 1.09 | 1.39 |

As expected the SVM outperformed Naïve Bayes, reducing the overall error rate from 28 to 15%. We also obtained better mean ranks except in the case of cars. The 2% error rate for faces seems to be comparable with state-of-the-art approaches, and is perhaps surprising that our method exploits virtually no geometric information. However a direct comparison is not possible since our method must also cope with background images from the other classes. This low error rate comes at a price of increased confusion of other categories with faces because of the larger number of faces in the training set.

In training this SVM we used the same best vocabulary with $k=1000$ as for Naïve Bayes. We compared linear, quadratic and cubic SVM's and found that linear method gave the best performance (except in the case of cars where a quadratic SVM gave better results). The parameter $C$ was determined for each SVM and values of around $C$=0.005 typically gave the best results.

### 3.3 Results on database [16]

We also tested our approach with the freely available[4], database used in [16]; i.e. the following five object classes and the set of background images: faces (450 images), airplanes (side) (1074 images), cars (rear) (651 images), cars (side) (720 images), and motorbikes (side) (826 images).

The results in Table 3 shows, that the images in this database are much easier to classify than our database. These results cannot be directly compared with the results stated in [16]. This is because their correct classification rate is a result of classifying images of a given class against background images (a two-class problem), which is different from our case where we conduct a full multi-class comparison using ranks (we hope to publish direct comparison soon). However, the diagonal elements of Table 3 suggest that our approach will give at least as good results as theirs in two-class

[4] Downloaded from http://www.robots.ox.ac.uk/~vgg/data except for cars (side) that comes from http://l2rcs.cs.uiuc.edu/~cogcomp/index_research.html.

classification (generally the two-class classification is more robust than the multi-class one). While the confusion matrices given in Table 2 of [16] are not directly comparable with our results, they suggest a rather high confusion rate in the multi-class case.

**Table 3.** .Confusion matrix and mean rank for SVM (*k=1000, linear kernel).*

| *True classes →* | *faces (frontal)* | *airplanes (side)* | *cars (rear)* | *cars (side)* | *motorbikes (side)* |
|---|---|---|---|---|---|
| *faces(frontal)* | **94** | 0.4 | 0.7 | 0 | 1.4 |
| *airplanes (side)* | 1.5 | **96.3** | 0.2 | 0.1 | 2.7 |
| *cars (rear)* | 1.9 | 0.5 | **97.7** | 0 | 0.9 |
| *cars(side)* | 1.7 | 1.9 | 0.5 | **99.6** | 2.3 |
| *motorbikes (side)* | 0.9 | 0.9 | 0.9 | 0.3 | **92.7** |
| *Mean ranks* | 1.07 | 1.04 | 1.03 | 1.01 | 1.09 |

## 4 Conclusion

We have presented a simple but novel approach to generic visual categorization using feature vectors constructed from clustered descriptors of image patches. This approach has been evaluated on a seven category database, demonstrating the method is robust to background clutter and produces good categorization accuracy even without exploiting geometric information. Our results with SVM are clearly superior to those obtained with the simple Naïve Bayes classifier. To our best knowledge this is the largest number of visual categories that has ever been subjected to simultaneous experiment.

Much future work remains. As we extend to more visual categories, the discriminative power of the appearance of unordered image patches will not suffice and we will need to extend the categorizer to incorporate geometric information. We will also need to extend our method for cases where the object of interest occupies a small fraction of the field of view, and to investigate many promising alternatives for each step of the basic method (point detection, clustering and classification).

**Acknowledgments**

This work was supported by the European Project IST-2001-34405 LAVA (Learning for Adaptable Visual Assistants, http://www.l-a-v-a.org). We are grateful to DARTY for their permission to acquire images in their shops, to INRIA for the use of their multi-scale affine interest point detector and to TU Graz for the bikes image database.

## References


[1] E. Osuna, R. Freund, F and Girosi. Training support vector machines: An application to face detection, CVPR (Computer Vision and Pattern Recognition), 1997.

[2] C. Papageorgiou, T. Evgeniou and T. Poggio. A trainable pedestrian detection system, IEEE Conference on Intelligent Vehicles, 1998.

[3] H. Schneiderman and T. Kanade, "A Statistical method for 3D object detection applied to faces and cars", CVPR, 2000.

[4] P. Viola and M. Jones, Rapid object detection using a boosted cascade of simple features, CVPR, 2001

[5] S.Z. Li, L. Zhu, Z.Q. Zhang, A. Blake, H.J. Zhang and H. Shum, Statistical learning of multi-view face detection, ECCV (European Conference on Computer Vision), 2002.

[6] R. Ronfard, C. Schmid, and B. Triggs, Learning to parse pictures of people, ECCV, 2002.

[7] T. Joachims. Text categorization with support vector machines: Learning with many relevant features, ECML, 1998.

[8] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. ICML, 2000.

[9] H. Lodhi, J. Shawe-Taylor, N. Christianini and C. Watkins, Text classification using string kernels. NIPS (In Advances in Neural Information Processing Systems), Vol 13, 2001.

[10] N. Cristianini, J.Shawe-Taylor and H. Lodhi, Latent Semantic Kernels, *Journal of Intelligent Information Systems*, **18** (2), 127-152, 2002.

[11] L. Zhu, A. Rao and A. Zhang, Theory of Keyblock-based image retrieval, *ACM Transactions on Information Systems,* **20**, (2), 224-257, 2002.

[12] Th. Leung and J. Malik, Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons, ICCV 1999.

[13] M. Varma and A. Zisserman, Classifying materials from images: to cluster or not to cluster?, ECCV 2002.

[14] S. Lazebnik, C. Schmid and J. Ponce, Sparse texture representation using affine-invariant neighborhoods, CVPR 2003.

[15] Y. Rubner and C. Tomasi, Texture-based Image Retrieval Without Segmentation, ICCV 1999.

[16] R. Fergus, P.Perona, and A. Zisserman, Object Class Recognition by Unsupervised Scale-Invariant Learning, CVPR 2003.

[17] T. Lindenberg, Scale-space theory in computer vision, Kluwer Academic Publishers, 1994.

[18] D. G. Lowe, Object Recognition from local scale–invariant features, ICCV (International Conference on Computer Vision), 1999.

[19] J. Matas, J. Burianek, and J. Kittler. Object recognition using the invariant pixel-set signature, BMVC (British Machine Vision Conference), 2000.

[20] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo, ICCV, 2001.

[21] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector, ECCV, 2002.

[22] K. Mikolajczyk and C. Schmid, A performance evaluation of local descriptors, CVPR, 2003.

[23] O. Duda, P.E. Hart, D.G. Stork, Pattern classification, John Wiley & Sons, 2000.

[24] D. Pelleg and A. Moore. X-Means: Extending K-means with Efficient Estimation of the Number of Clusters, International Conference on Machine Learning, 2000.

[25] V. Vapnik. Statistical Learning Theory. Wiley, 1998

[26] D. D. Lewis, Naïve Bayes at forty: The independence assumption in information retrieval, ECML, 1998.

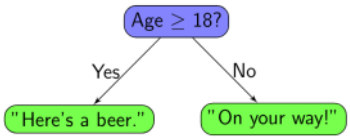
[27] P. Domingos and M. Pazzani, On the optimality of simple Bayesian classifier under zero-one loss, *Machine Learning,* **29**, 1997.

# Decision Trees & Random Forest

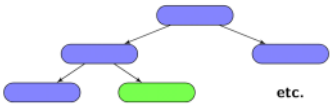
Monday, May 6, 2024 7:01 AM

## Decision Trees

- Decision Trees consist of
  - Nodes - the condition
  - Leaves - results
  - Branches - decisions




| Chest pain | Good circu-lation | Blocked arteries | Heart dis-ease |
|---|---|---|---|
| No | No | No | No |
| Yes | Yes | Yes | Yes |
| Yes | Yes | No | No |
| Yes | No | No | Yes |
| ... | ... | ... | ... |

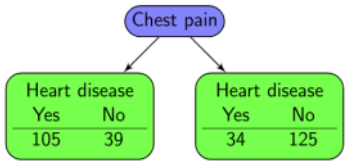


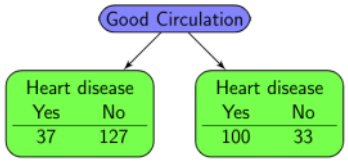

- **Stump** - A tree with a single node and two leaves
- **Root node** - variable that best predicts the result alone
- Given a set of training data, how to decide which condition to be the root node?
  - One measure is **Gini Impurity**

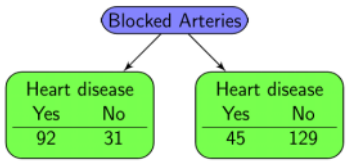
---

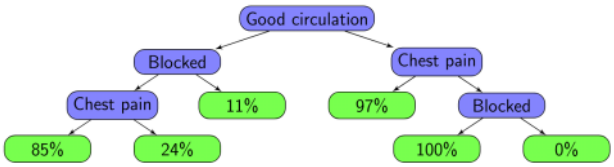
## Gini Impurity

- Gini = $1 - p_y^2 - p_n^2$
  - $p_y$ is the probability under the "yes" branch
  - $p_n$ is the probability under the "no" branch
- Chest pain (Yes):
  - Gini = $1 - \left(\frac{105}{105+39}\right)^2 - \left(\frac{39}{105+39}\right)^2 = 0.395$
- Chest pain (No):
  - Gini = $1 - \left(\frac{34}{125+34}\right)^2 - \left(\frac{125}{125+34}\right)^2 = 0.336$
- Chest pain (weighted average):
  - Samples in Chest Pain (Yes) = $105 + 39 = 144$
  - Samples in Chest Pain (No) = $34 + 125 = 159$
  - Avg = $144(0.395) + 159(0.336) = 0.364$
- Choose the question with the lowest Gini Impurity represents the separation
  - Given that the Weighted Average of the symptoms are:
  - | Chest Pain | Good Circulation | Blocked Arteries |
    |---|---|---|
    | 0.364 | 0.360 | 0.381 |
  - Good Circulation should be the root
- Then Gini Impurity for the other variables are recalculated
  - Based on the number of samples that go down each branch
  - i.e. If patient has good circulation, the probability of Blocked (yes),...
- Note that this tree does not perfectly classify the training data




---

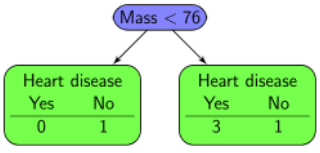
## Numerical Nodes

- For questions with numerical values

1. Sort the data into ascending order
2. Then calculate the average value between each sample
3. Calculate the Gini Impurity at each of these averages
   - Mass < 76 (Yes):
     - Gini = $1 - \left(\frac{0}{0+1}\right)^2 - \left(\frac{1}{0+1}\right)^2 = 0$
   - Mass < 76 (No):
     - Gini = $1 - \left(\frac{3}{3+1}\right)^2 - \left(\frac{1}{3+1}\right)^2 = 0.375$
   - Weighted Average = $0 \cdot \frac{(0+1)}{(0+1)+(3+1)} + 0.375 \cdot \frac{3+1}{(0+1)+(3+1)} = 0 \cdot \frac{1}{5} + 0.375 \cdot \frac{4}{5} = 0.3$
- Given the Gini Impurity of each mass, Mass < 93 would be the root node

| Patient Mass (kg) | Heart disease |
|---|---|
| 70 | No |
| 82 | Yes |
| 86 | No |
| 100 | Yes |
| 102 | Yes |

| Patient Mass (kg) | Heart disease |
|---|---|
| 70 | No |
| **−76−** | |
| 82 | Yes |
| **−84−** | |
| 86 | No |
| **−93−** | |
| 100 | Yes |
| **−101−** | |
| 102 | Yes |




| Gini Impurity | |
|---|---|
| Mass < 76 | 0.3 |
| Mass < 84 | 0.47 |
| Mass < 93 | 0.27 |
| Mass < 101 | 0.4 |

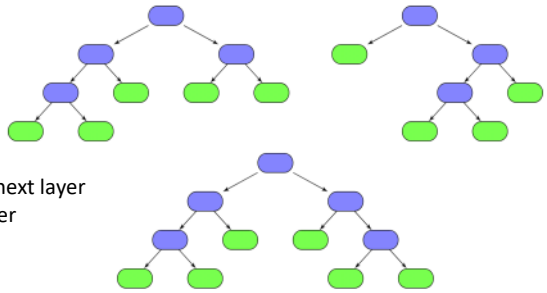
---

## Random Forest

| Chest pain | Good circu-lation | Blocked arteries | Mass (kg) | Heart dis-ease |
|---|---|---|---|---|
| No | No | No | 57 | No |
| Yes | Yes | Yes | 82 | Yes |
| Yes | Yes | No | 95 | No |
| Yes | No | No | 76 | Yes |
| ... | ... | ... | ... | ... |

- Decision trees do not always perfectly classify the training data
  - Therefore it is not likely to be perfect in classifying new samples
  - In general not very accurate

- **Random Forest** - a bootstrapped dataset
  - A collection of decision trees created by random sampling
  - Increase accuracy over a single decision tree



1. Create a bootstrapped dataset the same size as the original
   - Copy samples **randomly** into the new dataset
   - **Duplicates are allowed**
2. After selecting the root, a **random** set of the remaining variables are chosen as the next layer
   - In this case 2 of the 3 remaining variables are chosen randomly as the next layer
3. Repeat the process many times to build many decision trees



---

## Using the Random Forest

| Chest pain | Good circula-tion | Blocked arteries | Mass (kg) | Heart disease |
|---|---|---|---|---|
| Yes | No | No | 78 | ? |

| Heart disease | |
|---|---|
| Yes | No |
| 6 | 17 |

- Given a new patient, predict the probability of they having heart disease

- The data of the patient is processed by every decision tree
- A tally of the results is kept and the highest vote gives the result for the patient
  - Each tree has an equal weighting in the vote for the final decision

- In this case, out of 23 decision trees, 6 said yes and 17 said no

# AdaBoost

Monday, May 6, 2024 10:34 AM

## Basis

- **AdaBoost** - Adaptive Boosting
  - Introduced by Yoav Freund and Robert E. Schapire in 1995
  - No neural network involved

- AdaBoost - a discriminative classifier that uses weak learners which are usually stumps
  - Some stumps get more of a say than others - instead of random forest votes
  - Each stump is created taking into account the mistakes of previous stumps

- **Forest of Stumps** - a forest of decision stumps

---

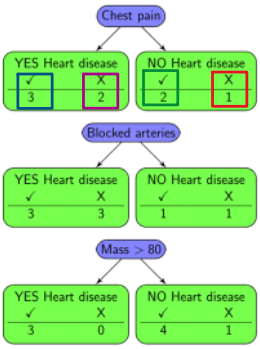
## First Stump in the Forest

1. Each sample in the training data needs to be given a **weight**
   - Shows how important the sample is to be correctly classified
   - Weights must add up to 1
   - At initial stage, all samples are equally important

2. Use the sample procedure to calculate the Gini Impurity
   - This time check the correctness of the stumps
     - E.g. for the left leaf of Chest Pain (Yes): CP(Yes) HD(Yes) = 3, CP(Yes) HD(No)=2
     - For right leaf of Chest Pain (No): CP(No)HD(No)=2, CP(No)HD(Yes)=1

3. Select the stump with the lowest Gini Impurity as the first stump
   - In this example, Mass > 80 is the first stump

| Chest pain | Blocked arteries | Mass (kg) | Heart disease | Weight |
|---|---|---|---|---|
| Yes | Yes | 93 | Yes | 1/8 |
| No | Yes | 82 | Yes | 1/8 |
| Yes | No | 95 | Yes | 1/8 |
| Yes | Yes | 76 | Yes | 1/8 |
| No | Yes | 71 | No | 1/8 |
| No | Yes | 57 | No | 1/8 |
| Yes | No | 76 | No | 1/8 |
| Yes | Yes | 78 | No | 1/8 |




| | |
|---|---|
| Chest pain | 0.47 |
| Blocked arteries | 0.5 |
| Mass > 80 | 0.2 |

---

## Adjusting the Weight

- The **amount of say** of a stump depends on how well it classified the samples
- **Total Error** of a stump - the sum of the weights for all the samples it got wrong
  - E.g. For Chest Pain, total error = $2 * \frac{1}{8} + 1 * \frac{1}{8} = \frac{3}{8}$
  - For Mass > 80, total error = $\frac{1}{8} * 0 + \frac{1}{8} * 1 = \frac{1}{8}$


- **Amount of Say** $= \frac{1}{2}\ln\left(\frac{1-Total\ Error}{Total\ Error}\right)$
  - For Mass > 80, amount of say $= \frac{1}{2}\ln\left(\frac{1-\frac{1}{8}}{\frac{1}{8}}\right) = \frac{1}{2}\ln 7 = 0.97$

- For the Current stump with the **lowest Gini**,
  - The **incorrectly** classified samples need to **increase their weights**
    - New Weight = old weight $\times\ e^{amount\ of\ say}$
  - The **correctly** classified samples can **decrease** their weights
    - New Weight = old weight $\times\ e^{-\ amount\ of\ say}$

- The new weights are then normalized to add up to 1

| Chest pain | Blocked arteries | Mass (kg) | Heart disease | Weight | New weight | Normalised |
|---|---|---|---|---|---|---|
| Yes | Yes | 93 | Yes | 1/8 | 0.05 | 0.07 |
| No | Yes | 82 | Yes | 1/8 | 0.05 | 0.07 |
| Yes | No | 95 | Yes | 1/8 | 0.05 | 0.07 |
| Yes | Yes | 76 | Yes | 1/8 | 0.33 | 0.49 |
| No | Yes | 71 | No | 1/8 | 0.05 | 0.07 |
| No | Yes | 57 | No | 1/8 | 0.05 | 0.07 |
| Yes | No | 76 | No | 1/8 | 0.05 | 0.07 |
| Yes | Yes | 78 | No | 1/8 | 0.05 | 0.07 |

---

## Next Stump

- The **weighted Gini impurities** are calculated to create the next stump
  - Like before, but the weights will now be taken into account to give more emphasis to the incorrectly classified samples

- Each stump has an influence on the following stump getting them to concentrate on fixing the mistakes it made

---

## Classifying Unseen Example

- After creating a forest of stump
  - Each stump has their own amount of say

- For a new patient:
  - Stump 1,2,3,4 decided that they has heart disease
  - Stump 5,6 decided that they does not have heart disease

- Final decision is the one with the largest total say
  - In this case, the person has heart disease as 2.7 > 1.23

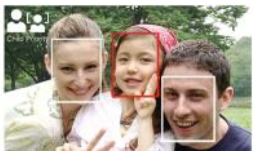
| Stump | Amount of say |
|---|---|
| 1 | 0.97 |
| 2 | 0.32 |
| 3 | 0.78 |
| 4 | 0.63 |
| **Total** | **2.7** |

| Stump | Amount of say |
|---|---|
| 5 | 0.41 |
| 6 | 0.82 |
| **Total** | **1.23** |

# Viola-Jones Algorithm

Monday, May 6, 2024 7:01 AM

## Basis

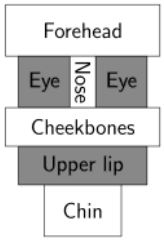


- Applications:
  - Face Detection
    - Focus, exposure and red-eye removal
    - Tracking
    - Adult/Child detection
  - Cat & Dog Detection
    - Automatically take image when face-on
    - Even pet recognition
  - Local a face for shape-fitting

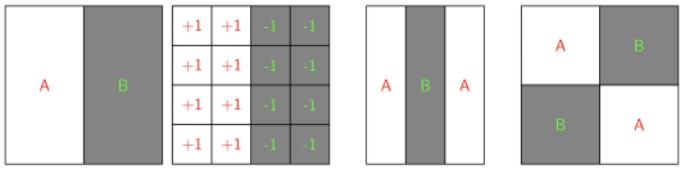
- **Viola Jones** Fast Face Detection
  - Requirements
    - Speed of computation (15 frames per second)
    - ~ 90-95% detection rate
    - $\sim 10^{-5}$ false positive rate
  - Approach
    - Limit to frontal upright faces
    - Efficient-to-compute features
    - Efficient image representation
    - AdaBoost for efficient choice of features
    - Cascade of classifiers

---

## Rectangular Features




- Clues for detecting human faces
  - Find basic features and their relationship
  - Simple pattern in the blurred contrast-adjusted image
  - Dark and light patches in horizontal and vertical patterns
- Faces can be detected by looking for dark & light patches in certain arrangements
  - Each rectangle potentially represents a feature
  - A difference in the sum of features (rectangles) can be calculated
    - $F = \sum_{r \in A} I(r) - \sum_{r \in B} I(r)$

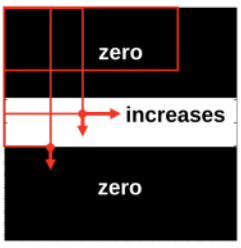



- Similar concept to convolution but convolution would be too slow
  - Faces can be different sizes - multiple kernel in different sizes would be needed on the same image

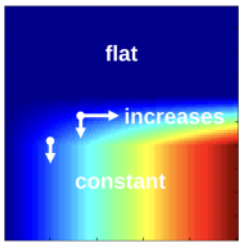
---

## Image Representation - Integral Image


- Viola Jones represent images using the integral image
  - $\mathfrak{T}(x,y) = \sum_{a \le x} \sum_{b \le y} I(a,b)$
- The pixel location of the integral image is the sum of all intensities inside the window
- The integral image can be calculated in 1 pass
- The values won't fit into an array of unsigned bytes (0-255)

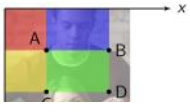



$I(x,y)$

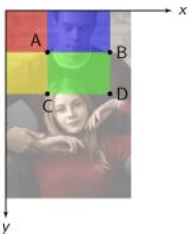


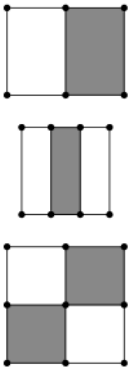

$\mathcal{I}(x,y)$

- The integral image retains all information in the original
  - In the flat area, intensity is 0 so it remains flat (blue)
  - When it reaches the white area, intensity increases in both x and y direction
  - When it reaches the black area again, intensity is constant in the y direction but keep increasing in the x direction (as it is still white)
- Calculating sum of intensities of a $\mathfrak{T}$ image
  - The sum of intensities in the rectangle ABCD can be calculated:
    - Sum in the red rectangle is $\mathfrak{T}(A)$



- Calculating sum of intensities of a $\mathfrak{I}$ image
  - The sum of intensities in the rectangle ABCD can be calculated:
    - Sum in the red rectangle is $\mathfrak{I}(A)$
    - Sum in the red and blue rectangle is $\mathfrak{I}(B)$
    - Sum in the red and yellow rectangle is $\mathfrak{I}(C)$
  - Therefore Sum in the green rectangle is
    - $\mathfrak{I}(D) - \mathfrak{I}(B) - \mathfrak{I}(C) + \mathfrak{I}(A)$
  - This is **calculated by 4 array references (lookups)** irrespective of the rectangle size



- Cheat sheet:
  - Two-rectangle: 6 array references
  - Three-rectangle: 8 array references
  - Four-rectangle: 9 array references
- Calculating the intensity sums in rectangular features is very efficient using integral images

---

## Normalizing Intensities in Windows

- Rectangular Features on Faces
  - Viola Jones processes faces of 24x24 pixels roughly aligned (looking to camera)
  - 2-, 3- and 4-rectangle features = 180k possibilities
  - Intensities in each face must be normalized
    - Mean = 0 and standard deviation = 1

- Intensities in a window can be normalized with
  - $x' = \frac{x - \bar{x}}{\sigma}$
  - Where $\bar{x}$ is the **mean intensity** and $\sigma$ is the **standard deviation**
- Mean $\bar{x}$ and standard deviation $\sigma$ can also be calculated using the integral image:
  - $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \Rightarrow \bar{I} = \frac{1}{N}\sum_{r \in W} I(r)$
    - The mean of the image pixel $\bar{I}$ is just the sum of all pixels inside the window divided by N
  - $\sigma^2 = \bar{x}^2 - \frac{1}{N}\sum_{i=1}^{N} x_i^2 \Rightarrow \sigma^2 = \bar{I}^2 - \frac{1}{N}\sum_{r \in W} \left(I(r)\right)^2$
    - The standard deviation can be calculated from the integral image of **the image squared**
- Normalization can be done using 2 integral images (usual one & one from a squared image)

---

## AdaBoost Classifier

- 180k possible rectangular features in a 24x24 face window takes a long time to process
  - AdaBoost Classifier is used to find **important rectangular features** only
  - First **single-stage weak** classifier

- Training data: 5k 24x24 faces and 10k 24x24 non-faces
- Results: 200 features, 95% detection rate, 1 in 14,084 false-positive rate
  - 0.7 sec to scan an 384 by 288 pixel image - still too slow

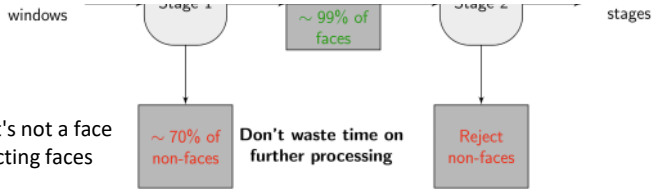
---

## Cascade Classifier

- Whole image search
  - Need to consider a range of possible scales
  - Need to consider all possible patch positions
  - Most patches are not faces
- Viola Jones uses Cascade Classifier to quickly get rid of non-faces (reduce false positives)
  - At each stage small number of faces might be rejected (false negatives)
  - 10-stage cascade classifier is used
    - False positive rate = $0.3^{10} \approx 6 \times 10^{-6}$
    - Detection rate = $0.99^{10} \approx 0.9 = 90\%$
  - Which satisfy their initial requirements


- The number of features in the first 5 stages is 1, 10, 25, 25, 50

    - Detection rate = $0.99^{10} \approx 0.9 = 90\%$
  - Which satisfy their initial requirements




- The number of features in the first 5 stages is 1, 10, 25, 25, 50
  - E.g. in the first stage, if the area does not have the particular feature, it's not a face
- Remaining stages have increasingly more features - more careful about rejecting faces
- Because many patches are rejected earlier on, detection speed increases

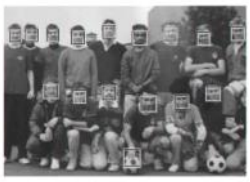
---

## Result Summary

- 38 stages of varying complexity, 6060 features in all
- Trained in a day using parallel processing
- Searches 384x288 pixel image in 0.067 seconds
- Applies features at multiple scales and locations
- 15x faster than previous approaches
- Can be generalized to other objects (people, cats, cars, dogs, etc.)


Detects multiple faces in an image


Fails for faces in profile (as expected)


Occasional mistakes (see the football)

# Welcome to Lok Gave up :D

Thursday, September 21, 2023 4:22 AM

## Title1

- Text

---

## Title2

- Text

---

## Title3

- Text

---

## Title4

- Text

---

## Title5

- Text

# I gave up :D

Friday, May 3, 2024 5:44 AM

## Q1 a)

- 

**Figure 1:** Examples of normal (**Left**) and diseased (**Right**) lettuce leaves. The diseased plant is infected with the lettuce big-vein virus.

As can be seen in **Figure 1**, lettuces infected with the lettuce big-vein virus show a different appearance to normal leaves. A student is trying to construct a computer vision system to classify lettuce plants as either normal or diseased. Describe what computer vision/image processing steps could be used to achieve this. Justify your choice of algorithm(s) and how your algorithm(s) might achieve the desired results. You should mention any difficulties that might arise, and how the values of any parameters used might be determined.

- As it is obvious that diseased leaves contains much more small edges compare to the normal one, edge detection can be used to detect the number of edges and the size or strength of the edges of the leaves. Then a threshold value can be used to differentiate the leaves that are diseased or normal based on these edges.

  Simple edge detection such as Canny can be used. First convert the image into greyscale image. Then, blur the image using a Gaussian filter to get rid of noise and small scale structures, then using the first derivate kernel to detect edges in the image. Canny then uses non-maxima suppression to find the precise edge location. Since Canny allows us to set threshold to control the length and orientation of the edges, a threshold can be used to differentiate diseased leaves where there are a lot of small edges.

  However the thresholds will need to be tuned through some experiments and observations. Furthermore, although Canny already applied Gaussian filter, it could still be sensitive to noises and this method works on the assumption that the leave images were taken in a similar background and setting, with no occlusions or objects that could affect the detection.

---

## Q1 b)

- **22. Essay: Q1.b: Is the Harris corner detector a linea...** Points: 4

  | Question | Is the Harris corner detector a linear filter? Argue why or why not. Assume that the Harris corner detector is applied to an unsmoothed image. What type of image would trigger the detector at places that clearly don't contain a corner? |
  |---|---|

- Harris Corner is not a linear filter. Although the convolutions are linear, it includes non-linear process such as image multiplications, calculating determinant and traces of the matrices. Images with lots of salt and pepper noise or images that were pixilated would trigger the detector to treat non-corner as corners. This is because these noises also include small sudden change of intensity in multiple directions, which shares the same property as corners.

---

## Q1 c)

- **23. Essay: Q1.c: Explain the role of the following par...** Points: 6

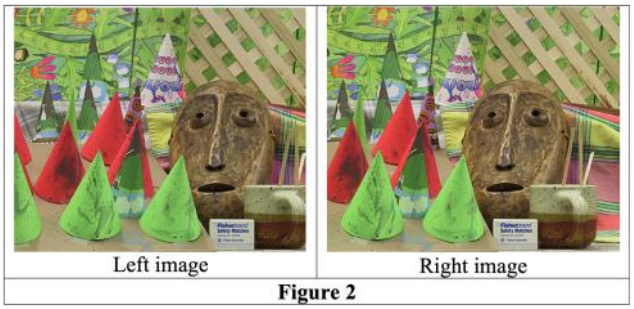
  | Question | Explain the role of the following parameters in the SIFT algorithm:<br>i. Contrast threshold<br>ii. Curvature threshold<br>iii. Dimensionality of feature vector |
  |---|---|

i. Contrast threshold is in charge of getting rid of low-contrast key points identified in the local maxima of the DoG features. They have to be larger than the threshold to be considered as valid keypoints or blobs
ii. Curvature threshold is used to get rid of edges and only preserve corners or local features as key points
iii. SIFT has 128 dimensions per feature, with 4x4 pixels, and 8 directions bins within each pixel.

---

## Q2 a)

**Figure 2** shows a pair of stereo images that have been captured using a pair of calibrated cameras.



Left image | Right image

**Figure 2**

Propose and describe an algorithm to segment one of the two images in **Figure 2** (i.e. the left image) into semantically coherent regions. In your answer you need to consider and justify your choice of feature space and segmentation algorithm.

- Text

---

## Q2 b)

**25. Essay: Q2.b: Figure 2 shows a pair of stereo image...** Points: 10

Question

**Figure 2** shows a pair of stereo images that have been captured using a pair of calibrated cameras.

Left image | Right image

**Figure 2**

Explain how you could use the pair of images in **Figure 2** to calculate the distances from the camera of the surface features that appear in the scene.

In your answer you need to consider all steps in the process, from images to depth values.

- Text

---

## Q2 c)

**26. Essay: Q2.c: Can you use the depth information fro...** Points: 5

Question

Can you use the depth information from Question **Q2.b** to improve your segmentation results in Question **Q2.a**? Explain and justify your answer.

- Text

# 2022

Thursday, September 21, 2023 4:22 AM

## Q2

Figure 1

- In a recent trip, a student captured the above images (Figure 1) and wants to write a computer vision application that can create a seamless panorama from these images.
  a. Describe the assumptions under which this is possible. **[1 mark]**
  b. Describe what computer vision/image processing steps could be used to achieve this and list the order of these steps. **[4 marks]**
  c. Justify your choice of algorithm(s), describe the algorithm(s) briefly, and explain and how the algorithm(s) might achieve the desired results. **[10 marks]**
  d. You should mention any difficulties that might arise, and how the values of any parameters used might be determined. **[5 marks]**

a. The assumption would be that these images were taken using the same panorama camera and the images were roughly projected ont o the same plane.

b. The processing steps would be
   1. Find corresponding points from each image to be stitched using local features. At least 4 pairs of matches are needed.
   2. Calculate the Homographies matrix which describe the transformation between two images according to these matching points.
   3. Use RANSAC to handle outliers and minimizes the residual errors of detected matches
   4. Use image blending to stitch the images together

c. DoG can be used to find interest points with stepping sigma to make these feature scale invariant, then SIFT can be used as d escriptors for these features, which are more robust to variations in orientation, illumination, and scale. Using a Lowie ratio test on the SSD t o improve the performance of matching. These gives as some corresponding points from the images.

   Homographies is calculated using these corresponding points. By adding an extra dimension to the xy space such that translati on, rotation, scaling and sheering are all linear transformation and the homographies describe the linear transformation from one image to another. By normalizing the H matrix, we restraint the problem into a least square problem which is more computationally efficient and only require 4 pai rs matching points.

   After calculating the H matrix, we can use RANSAC to get rid of outliers by applying H to the image finding inliers to the im age, if more inliers are found, the H can be updated using these inliers for more accuracy. This process is repeated N times until some pre -determinant percentage of accuracy is met.

   Lastly, alpha blending can be used to blend images together where the intensity, or alpha of the left image changes from 1 to 0 and the right from 0 to 1 using some window size, blending the left side of image 1 and right side of image 2 together as one single image.

d. In SIFT, we need to determine scale sigma for the DoG possibly through experiment or trials and errors. The ratio test thresh old possibly needs to be tuned according to performance. RANSAC needs the requirement of inliers are more than outliers for it to work properly, hence the matching points need to have good performance. Lastly, the window size of alpha blending needs to be chosen so that it is not too large to ge nerate ghosting effect and not too small for seams, this can be chosen from observations, or following the principles of windows size = size of larg est prominent feature and <= 2* size of smallest prominent features.

---

## Q3 a)

a) Describe in detail the Hough Lines transform (you may use Cartesian or polar coordinates). **[4 marks]**

- Given a set of points in the coordinate space, the aim of Hough Lines Transform is to find lines that potentially pass throug h these points.

A parameter is set as a threshold to indicate the minimum number of votes or points are required to be considered as a line.
Then we rearrange the line formulas from the coordinate space to the parameter space. For example, in Polar Coordinates:
  - $r = x\cos\theta + y\sin\theta$
- Then substitutes the points into the parametric equations to find the possible r and theta values
- These lines are then plotted into the accumulator array (r = columns, theta=rows)
- The cell inside the array + 1 if the lines passes through it
- The cell with the highest votes indicate the value of r and theta for the line that passes through the points

---

## Q3 b)

b) Describe in detail the Generalised Hough transform. In your answer you should include details about the model creation and finding a shape within an image.

**[6marks]**

- **Model Creation**:
  - Given a template, or example shape, a point $(x_c, y_c)$ inside the shape region is selected as the reference point.
    Then for any sample point $(x, y)$ on the shape contour, calculate the edge normal, or gradient $\phi$ and the angle $\alpha$ which is the angle between the vector $r$ from the sample point to the reference point and store $r$ and $\alpha$ for that $\phi$ inside a R-table, which stores all the candidates template points and a range of $\phi$ .

- **Shape Detection**:
  - For arbitrary size and orientation, a 4D accumulator array created to store the coordinates, the size and the orientation of the shape. For each point in the target image, $\phi$ is calculated, and for each pair of $(r, \alpha)$ for that $\phi$ in the R-table, is used to calculate a potential object center of the shape. The 4D accumlator stores the votes of each pair, and the highest votes indicates a match of template shape in the targ et image.

---

## Q3 c)

c) A group of students enjoy creating memes. Using the left and the centre image as input, how can they *automatically* create the image on the right when the *only* other user inputs are:
- the red oval on the left image in Figure 2, indicating the area/object of interest
- the green dot on the centre image in Figure 2, indicating the desired location for the object

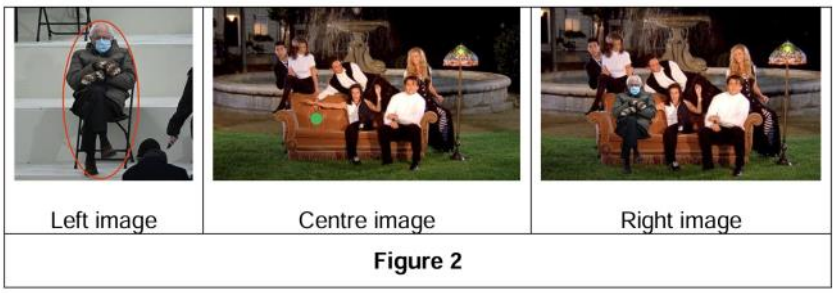



**Figure 2**

i. State the computer vision algorithms that can be used to achieve this.
**[2 marks]**

ii. Explain why and how the algorithm(s) might achieve the desired results.
**[4 marks]**

iii. How are they going to handle the difference in scale? What information can they extract from the input images to help them with this?
**[4 marks]**

i. GrabCut, face detection and affine transformations.

ii. GrabCut treats anything out of the ROI, which is the red oval immediately as background
and use unsupervised learning to improve the foreground / background modelling of GMM to improve the segmentation assumption.
Face detection can then be used to detect faces and their patch size.
Affine transformation can be used to scale Bernie into the correct size and potentially orientation if needed to the target i mage

iii. The faces in both images can be detected using some feature detectors, it could be via some face detection data set, or using Viola Jones.
After detecting the face we can capture the information of the size of the faces and calculate a responsible ratio of the fac e sizes.
This ratio can be used to scale the Bernie image so that it has the same size as the other faces in the target image.

# AI Generated MCQs by Liangji

Wednesday, May 15, 2024 2:46 PM

Computer vision is concerned with modeling and replicating human vision using computer software and hardware. 1 point

- (●) True
- ( ) False
- ( ) Can't be true or false
- ( ) May be both true and false

Clear selection

What is the representation of image in the context of computer vision? 1 point

- ( ) A. Image function
- ( ) B. Landscape
- ( ) C. Array of pixels
- ( ) D. Image histogram
- (●) E. All above

Clear selection

What is a characteristic of a stochastic process in the context of image noise? 1 point

- A. Identical images have identical values
- B. Identical images do not have identical values
- C. It increases the signal to noise ratio
- D. It reduces the need for spatial averaging

Clear selection

What is the advantage of describing images with fewer pixels? 1 point

- A. Detection of larger-scale objects is less reliable
- B. Detection of larger-scale objects is more reliable
- C. It increases the confusing detail in the image
- D. It reduces the speed and robustness of detection

Clear selection

What is the purpose of Image Arithmetic (Addition)? 1 point

- A. To increase the noise in the image
- B. To detect changes in a static background
- C. To take an average over images in a sequence to reduce noise
- D. To increase the scale of the image

Clear selection

What is the purpose of Image Arithmetic (Subtraction)? 1 point

- A. To increase the noise in the image
- B. To take an average over images in a sequence to reduce noise
- C. To detect changes in a static background
- D. To increase the scale of the image

Clear selection

What is the purpose of Rank Filtering in image processing? 1 point

- ○ A. To increase the noise in the image
- ○ B. To detect changes in a static background
- ○ C. To reduce spatial resolution
- ◉ D. To sort the pixel values in the neighborhood of each pixel and replace the pixel value with the value determined by the ranking result

Clear selection

Which of the following is NOT a type of Rank Filtering? 1 point

- ○ A. Maximum/dilation
- ○ B. Minimum/erosion
- ○ C. Median
- ◉ D. Mean

Clear selection

What is the difference between Mean and Median in the context of smoothing? 1 point

- (●) A. Mean considers all values equally, while Median is less sensitive to outliers
- ( ) B. Mean is less sensitive to outliers, while Median considers all values equally
- ( ) C. Mean and Median are the same in the context of smoothing
- ( ) D. Mean is used for smoothing, while Median is not

Clear selection

What is the primary goal of Segmentation in image processing? 1 point

- ( ) A. To convert a grayscale image to a binary label image
- (●) B. To separate background pixels from object pixels
- ( ) C. To allow for better thresholding
- ( ) D. To stochastically choose the correct objects in the image

Clear selection

How is the threshold typically selected in Thresholding? 1 point

- ( ) A. It is selected randomly
- (●) B. It is selected based on the histogram, where the rate of change of the segmented area is smallest
- ( ) C. It is selected based on the mean intensity of the image
- ( ) D. It is selected based on the median intensity of the image
- ( ) E. It is selected based on the histogram, where the rate of change of the segmented area is largest

Clear selection

What is a key advantage of adaptive thresholding over simple thresholding in image processing? 1 point

- A. Adaptive thresholding uses a single global threshold for the entire image, leading to more consistent results.
- B. Adaptive thresholding uses a local or region-based threshold, allowing it to handle varying lighting conditions across different areas of the image.
- C. Adaptive thresholding applies a threshold that is predetermined and constant for all pixels, providing faster processing.
- D. Adaptive thresholding requires no parameter tuning and is fully automated, unlike simple thresholding.

Clear selection

In mean filter, the value of the current pixel is replaced with 1 point

- mean value of pixels of the image
- mean value of neighbours
- mean value of the neighbours and current pixel
- None of the above

Clear selection

The aliasing effect on an image can be reduced using which of the following methods? 1 point

- a) By reducing the high-frequency components of image by clarifying the image
- b) By increasing the high-frequency components of image by clarifying the image
- c) By increasing the high-frequency components of image by blurring the image
- d) By reducing the high-frequency components of image by blurring the image

Clear selection

Which side of the greyscale is the components of the histogram concentrated in a dark image? 1 point

- a) Medium
- b) Low
- c) Evenly distributed
- d) High

Clear selection

# AI Generated MCQs by Liangji

Wednesday, May 15, 2024 2:51 PM

What causes edges in image processing? 1 point

- ○ A. Depth discontinuity
- ○ B. Surface colour discontinuity
- ○ C. Illumination discontinuity
- ◉ D. All of the above

Clear selection

What is an edge in the context of image processing? 1 point

- ◉ A. A place of rapid change in the image intensity function
- ○ B. A place where the image is blurred
- ○ C. A place where the image is distorted
- ○ D. A place where the image is pixelated

Clear selection

Which of the following is true about convolution in image processing? 1 point

- A. Convolution is not commutative
- B. Smoothing is not a type of convolution
- C. Convolution is associative
- D. None of the above

Clear selection

Which of the following best describes the purpose of image gradients in image processing? 1 point

- A. To measure the rate and direction of change in pixel values within an image
- B. To apply convolution with a smoothing kernel to reduce noise
- C. To convert an image from the spatial domain to the frequency domain
- D. To reconstruct a 3D scene from 2D images

Clear selection

What is a decomposable kernel in the context of image processing? 1 point

- A. A kernel that can be broken into simpler components for ease of understanding
- B. A kernel that separates images into distinct regions
- C. A kernel that can be separated into smaller kernel components, allowing for computational efficiency
- D. A kernel used to extract specific features, like edges or textures, from an image

Clear selection

In Gaussian smoothing, what is the role of the standard deviation (σ)? 1 point

- A. It defines the central value around which pixel values are centered
- B. It specifies the number of times the kernel is applied to the image
- C. It controls the extent of smoothing, with larger values resulting in more blur
- D. It determines the number of iterations for smoothing

Clear selection

What is the primary use of the Sobel operator in image processing? 1 point

- ○ A. To smooth images with Gaussian blurring
- ◉ B. To detect edges by calculating image gradients in horizontal and vertical directions
- ○ C. To separate color channels for further processing
- ○ D. To perform decompositions in the frequency domain

Clear selection

Why might edge detection at a coarser scale be more reliable compared to finer scales in image processing? 1 point

- ○ A. Edge detection at coarser scales generally has better performance with high-resolution images
- ○ B. Coarser scales provide more detailed information
- ○ C. Coarser scales require less computational resources
- ◉ D. There is less confusing detail at coarser scales

Clear selection

What is a unique property of the Laplacian of Gaussian (LoG) filter in image processing? 1 point

- ○ A. It detects only horizontal edges in an image
- ◉ B. It combines Gaussian smoothing and Laplacian edge detection in a single operation
- ○ C. It calculates image gradients in both vertical and horizontal directions simultaneously
- ○ D. It provides additional color information to enhance edges

Clear selection

How does using a weighted average in smoothing operations improve image processing results? 1 point

- ○ A. It provides enhanced contrast in high-frequency regions
- ○ B. It decreases the noise reduction capacity of the filter
- ○ C. It allows for quicker smoothing operations
- ◉ D. It reduces the influence of outliers by assigning more weight to central pixels

Clear selection

Why is it important to normalize a Gaussian kernel in image processing? 1 point

- A. To allow for a smoother application of the kernel across the image
- B. To increase the computational efficiency of convolution
- C. To ensure that the output pixel values are within a consistent range
- D. To reduce edge distortions during processing

Clear selection

What makes the Gaussian kernel particularly useful for anti-aliasing in image processing? 1 point

- A. Its ability to smooth high-frequency noise before subsampling, reducing aliasing effects
- B. Its capacity to enhance the clarity of edges during downsampling
- C. Its characteristic of preserving high-frequency components while filtering out low-frequency noise
- D. Its capability to detect edges in both horizontal and vertical directions

Clear selection

Why is it useful to locate edges at a coarse scale before searching for them at a finer scale? 1 point

- A. It eliminates the need to perform edge detection at finer scales entirely
- B. It provides a general guide for focusing edge detection efforts, improving efficiency and robustness
- C. It produces stronger edges with greater contrast
- D. It allows for edge detection without requiring additional normalization

Clear selection

A Sobel filter can be written as the image below. 1 point
Which of the following statements are true.

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

- (a) Separating the filter in the above manner, reduces the number of computations.
- (b) It is similar to applying a gaussian filter followed by a derivative.
- (c) Separation leads to spurious edge artifacts.
- (d) Separation approximates the second derivative of gaussian.
- a and b
- b and c

Clear selection

The Canny edge detector is a linear filter because it uses the Gaussian filter to blur the image and then uses the linear filter to compute the gradient. 1 point

- True
- False

Clear selection

What is the fundamental equation used in the Hough Transform for line detection? 1 point

- y = mx + b
- ax^2 + bx + c = 0
- ρ = xcos(θ) + ysin(θ)
- x^2 + y^2 = r^2

Clear selection

Why is the Hough Transform for line detection commonly implemented in polar coordinates rather than Cartesian coordinates? 1 point

- (●) A) It allows representation of lines with vertical slopes without division by zero
- ( ) B) It reduces computational complexity
- ( ) C) It directly corresponds to the visual representation of images
- ( ) D) It simplifies calculations involving curved lines

Clear selection

In the context of the Hough Transform for line detection, what do the parameters $\rho$ and $\theta$ represent? 1 point

- (●) A) ρ is the distance from the origin to the line, and θ is the angle of the normal to the line.
- ( ) B) ρ is the length of the line segment, and θ is its slope.
- ( ) C) ρ represents the radial distance from the center, and θ is the angle in radians
- ( ) D) ρ is the y-intercept, and θ is the angle relative to the x-axis

Clear selection

What role does the threshold parameter play in the Hough Transform for line detection? 1 point

- ( ) A) It determines the sensitivity of the algorithm to edge gradients
- ( ) B) It sets the maximum distance between parallel lines
- ( ) C) It adjusts the angle step size for the polar grid
- (●) D) It determines the minimum number of points required to consider a potential line

Clear selection

What is the role of the accumulator array in the Hough Transform for circle detection? 1 point

- A) It accumulates the distances from edge points to their corresponding circle centers
- B) It holds the pixel values for each detected circle in the image
- C) It counts the number of votes for each possible set of circle parameters (center coordinates and radius)
- D) It stores the results of edge detection prior to applying the Hough Transform

Clear selection

How can the Hough Transform be used to detect circles of different radii in an image? 1 point

- A) By iterating over a range of possible radii and counting votes in the accumulator array for each radius
- B) By fixing the radius and varying the center coordinates in a 2D accumulator array
- C) By adjusting the brightness threshold for edge detection to detect circles of different radii
- D) By performing multiple iterations with different edge detection parameters

Clear selection

In the generalized Hough transform, how is the shape template used to detect shapes in an image at any scale and orientation? 1 point

- A) By creating a 3D histogram of gradient magnitudes to detect edges and contours
- B) By mapping edge orientations to an R-table and using a 4D accumulator array to find shape matches in the input image
- C) By extracting the pixel intensity values and matching them with a predefined shape pattern
- D) By applying Fourier Transform to capture shape patterns and then comparing them with the template

Clear selection

What is the role of the R-table in the generalized Hough transform for shape detection? 1 point

- ○ A) It counts the number of edge points that match the given template.
- ◉ B) It stores the relationship between edge orientations and the shape's reference point, allowing the accumulator array to detect the shape in various scales and orientations.
- ○ C) It represents a histogram of gradient magnitudes, indicating the strength of edges in the image.
- ○ D) It stores contour points derived from the grey levels, used to find the most probable shape in the image.

Clear selection

Why is a 4D accumulator array needed in the generalized Hough transform to find a template shape at any scale and orientation? 1 point

- ○ A) It holds the pixel intensity values at each contour point for improved precision.
- ○ B) It stores the results of multiple iterations to improve accuracy.
- ◉ C) It accounts for variations in the x and y positions, orientation, and scale of the shape, allowing flexible shape detection.
- ○ D) It ensures computational efficiency by reducing the complexity of the problem.

Clear selection

____ can detect multiple instance of a model in a single pass. 1 point

- ◉ Hough Transform
- ○ Active shape model
- ○ K-means clustering
- ○ Affine Transform

Clear selection

# AI Generated MCQs by Liangji

Thursday, May 16, 2024 3:02 AM

Which Gestalt principle is associated with the perception of incomplete shapes or patterns as complete? 1 point

- A) Proximity
- B) Closure
- C) Continuity
- D) Similarity

Clear selection

Which Gestalt principle involves perceiving a pattern or shape as continuous, even if it is interrupted by gaps or other elements? 1 point

- A) Closure
- B) Proximity
- C) Continuity
- D) Similarity

Clear selection

What is a superpixel in the context of image segmentation? 1 point

- A) A pixel with a higher resolution than surrounding pixels
- B) A group of adjacent pixels that share similar visual characteristics, forming a cohesive region
- C) A pixel that has undergone extensive preprocessing for noise reduction
- D) A pixel located at the center of an object of interest in an image

Clear selection

What is a common advantage of using superpixels in image segmentation? 1 point

- [x] A) They reduce the computational complexity by grouping similar pixels into larger, cohesive regions
- [ ] B) They increase the image resolution by combining pixels
- [ ] C) They automatically highlight objects of interest
- [ ] D) They prevent overfitting in machine learning models

Clear selection

What is the primary goal of image segmentation? 1 point

- [ ] A) Deals with property in which images are subdivided successively into smaller and equal sized regions
- [ ] B) To enhance the sharpness of an image
- [ ] C) To reduce the noise in an image through filtering
- [x] D) To partition an image into meaningful regions or segments based on visual characteristics

Clear selection

Which of the following best describes a bottom-up approach to image segmentation? 1 point

- [ ] A) Using contextual information to determine which pixels should be grouped together
- [ ] B) Segmenting an image by first recognizing high-level objects and then determining which pixels belong to those objects
- [x] C) Grouping pixels based on low-level similarities like color and texture, creating segments without prior knowledge of objects or scenes
- [ ] D) Segmenting images by identifying important components and then refining them into smaller segments

Clear selection

What is the primary goal of the K-means clustering algorithm? 1 point

- (●) A) To find K cluster centers that minimize the sum of squared distances (SSD) between all points and their nearest cluster center
- ( ) B) To maximize the distance between cluster centers to create distinct clusters
- ( ) C) To create a hierarchical clustering structure with a specific depth
- ( ) D) To assign each data point to a cluster such that the distance to its center is maximized

Clear selection

What is the typical process for K-means clustering after initializing the cluster centers? 1 point

- (●) A) Alternating between assigning data points to clusters based on the nearest center and recalculating the centers by computing the mean per cluster
- ( ) B) Assigning data points to clusters and then stopping once a predetermined number of iterations is reached
- ( ) C) Assigning each data point to a random cluster, then determining the mean cluster center at the end
- ( ) D) Determining the closest cluster center for each data point and then stopping immediately

Clear selection

Which of the following statements about K-means clustering convergence is correct? 1 point

- ( ) A) It will always converge to the global minimum
- (●) B) It will always converge to some solution, but this solution may be a local minimum
- ( ) C) It converges only when the clusters are perfectly spherical
- ( ) D) It may not converge if the initial cluster centers are too far apart

Clear selection

Which of the following is an advantage of K-means clustering? 1 point

- ○ A) It finds the global minimum of the objective function
- ◉ B) It is simple, fast to compute, and converges to a local minimum of within-cluster squared error
- ○ C) It automatically detects the optimal number of clusters, K
- ○ D) It is robust to outliers and non-spherical clusters

Clear selection

Which of the following is a common issue with K-means clustering? 1 point

- ○ A) It finds non-spherical clusters more easily than other algorithms
- ○ B) It requires high computational resources to converge
- ◉ C) It is sensitive to the choice of initial cluster centers and can be affected by outliers
- ○ D) It finds the optimal number of clusters without additional tuning

Clear selection

What is meant by "feature space" in the context of image processing and clustering? 1 point

- ○ A) A special transformation applied to the pixel values to enhance clustering
- ○ B) A 2D plane where pixels are grouped based on their proximity
- ◉ C) A multidimensional space where each dimension represents a particular characteristic or feature of the data
- ○ D) A process where images are decomposed into their constituent frequencies

Clear selection

When clustering pixels, what can be achieved by adding spatial information (x, y) to the feature space along with intensity or color values? 1 point

- ○ A) It allows for creating clusters with varying shapes and sizes
- ◉ B) It helps to group pixels based on both similarity and spatial proximity, enforcing spatial coherence
- ○ C) It provides a way to separate clusters based on their distance from the image center
- ○ D) It helps to separate objects from their backgrounds more effectively

Clear selection

What is a Gaussian Mixture Model (GMM) in the context of probabilistic clustering? 1 point

- ○ A) A clustering technique that assigns each data point to the closest Gaussian distribution
- ◉ B) A model that represents a distribution as a mixture of multiple Gaussian distributions, allowing for complex cluster shapes
- ○ C) A clustering method that represents data points as fixed points within a Gaussian distribution
- ○ D) A model that uses a single Gaussian distribution to describe all data points

Clear selection

What does the covariance matrix Σ represent in a multivariate Gaussian distribution? 1 point

- ○ A) The mean position of the Gaussian distribution
- ◉ B) The shape and orientation of the Gaussian distribution, indicating how different variables are related
- ○ C) The spread or variance of the Gaussian distribution
- ○ D) The probability that a data point belongs to a specific Gaussian component

Clear selection

What are the main steps of the Expectation-Maximization (EM) algorithm for Gaussian Mixture Models? 1 point

- A) E-step: Compute the maximum likelihood for each data point; M-step: Reassign clusters
- B) E-step: Assign data points to the closest Gaussian; M-step: Recalculate Gaussian means
- C) E-step: Randomly initialize Gaussian parameters; M-step: Update probabilities for each data point
- (●) D) E-step: Compute probabilities for each data point's cluster assignment; M-step: Update the Gaussian parameters based on these probabilities

Clear selection

Which of the following is an advantage of EM algorithm with Gaussian Mixture Models (GMM) compared to K-means clustering? 1 point

- A) EM is guaranteed to find the optimal number of clusters
- B) EM requires fewer iterations to converge
- C) EM is less sensitive to initial parameters and local minima
- (●) D) EM provides soft assignments, allowing a data point to belong to multiple clusters with varying probabilities

Clear selection

What is the basic process for the mean-shift segmentation algorithm? 1 point

- A) Iteratively shift a search window to the medium of data points within the window, converging toward areas of high density
- B) Randomly initialize cluster centers and then assign data points to the nearest center
- C) Iteratively shift a search window to the mean of data points within the window, converging toward areas of low density
- (●) D) Iteratively shift a search window to the mean of data points within the window, converging toward areas of high density

Clear selection

What causes convergence in the mean-shift segmentation algorithm? 1 point

- A) The standard deviation of the data points within the search window falls below a threshold
- B) The algorithm finds a predefined number of clusters
- C) The number of iterations reaches a predefined limit
- D) The search window stops moving because it has reached the center of gravity of its contained points, indicating a high-density region or mode

Clear selection

In mean-shift clustering, what is a mode? 1 point

- A) A standard deviation measure of a given cluster
- B) A data point with the highest intensity value
- C) A cluster with a predefined number of elements
- D) An area of high data density to which data points tend to converge

Clear selection

What is an advantage of mean-shift clustering over other clustering techniques like K-means? 1 point

- A) It automatically selects the optimal number of clusters
- B) It converges faster due to fewer iterations
- C) It is model-free and does not assume any prior shape for the clusters
- D) It is not sensitive to the choice of initial cluster centers

Clear selection

What is a key challenge with mean-shift clustering? 1 point

- A) It is sensitive to outliers in the data
- B) It requires predefined initialization of cluster centers
- C) Selecting an appropriate window size (bandwidth) is non-trivial and can affect the results
- D) It does not work well with non-continuous data

Clear selection

Why is mean-shift clustering considered computationally expensive? 1 point

- A) It needs multiple iterations to optimize the cluster centers
- B) It has to handle a high-dimensional feature space without reducing dimensionality
- C) It needs complex operations like Fourier transforms
- D) It requires repeated calculations of the mean within a moving search window until convergence

Clear selection

In graph-based segmentation, what does an affinity weight represent? 1 point

- A) The number of connections between nodes in a fully connected graph
- B) The distance between two nodes in the image
- C) The similarity between two connected nodes (pixels), often based on color, intensity, or position
- D) The total cost of removing a set of edges from the graph

Clear selection

What is a minimum cut in the context of graph-based segmentation? 1 point

- A) A cut that minimizes the sum of the weights of the remaining edges
- B) A cut that divides a graph into two equal-sized segments
- C) A cut that minimizes the sum of the weights of edges that are removed to create disconnected segments
- D) A cut that isolates individual nodes with no edges

Clear selection

Why might a minimum cut not be the best approach for graph-based segmentation? 1 point

- A) It leads to a large number of segments, making interpretation difficult
- B) It requires solving complex eigenvalue problems
- C) It tends to cut off small, isolated components due to the tendency to minimize edge weights
- D) It requires predefined segment sizes

Clear selection

How does the normalized cut (NCut) improve upon the minimum cut for graph-based segmentation? 1 point

- A) By normalizing for the size of segments to prevent cutting off small components and ensure balanced segments
- B) By reducing the computational complexity of the cut
- C) By requiring fewer affinity computations due to sparsely connected graphs
- D) By automatically finding the optimal number of segments

Clear selection

What is an advantage of the normalized cut approach for graph-based segmentation? 1 point

- (●) A) It provides a generic framework and is flexible in choosing the function to compute weights between nodes
- ( ) B) It ensures that all segments are approximately equal in size
- ( ) C) It guarantees a globally optimal solution for graph segmentation
- ( ) D) It automatically selects the most appropriate cut with minimal user input

Clear selection

What is a disadvantage of the normalized cut approach for graph-based segmentation? 1 point

- ( ) A) It requires predefined segment sizes
- (●) B) The time and memory complexity can be high due to dense, highly connected graphs
- ( ) C) It does not work well with sparse data
- ( ) D) It is sensitive to the choice of initial cut

Clear selection

What is the primary concept behind the GrabCut algorithm for image segmentation? 1 point

- ( ) A) It employs a binary threshold to separate foreground from background
- ( ) B) It applies a series of filters to distinguish between different regions
- (●) C) It uses graph cuts to segment the image into foreground and background regions, based on a foreground model
- ( ) D) It relies on the Hough Transform to identify key segments in an image

Clear selection

In GrabCut, how are graph cuts used to segment an image? 1 point

- (●) A) The graph cut is used to separate foreground and background based on an optimal cut, aiming to find the best boundary between them
- ( ) B) The graph cut removes all edges that cross between the foreground and background
- ( ) C) The graph cut is applied to detect clusters within the image, identifying the main regions
- ( ) D) The graph cut is used to create a sparse representation of the image for faster processing

Clear selection

What is a common method to improve the efficiency of image segmentation? 1 point

- ( ) A) Increasing the image resolution to capture finer details
- (●) B) Grouping similar-looking pixels into superpixels for further processing
- ( ) C) Reducing the number of color channels to simplify processing
- ( ) D) Applying high-pass filters to enhance edges

Clear selection

Which of the following is NOT a typical characteristic of superpixels? 1 point

- ( ) A) They do not cross boundaries between distinct regions
- (●) B) They tend to have irregular shapes and sizes
- ( ) C) They are generated through local over-segmentation
- ( ) D) They can be used to reduce the complexity of further processing

Clear selection

How is precision defined in the context of evaluating segmentation? 1 point

- A) The percentage of actual boundary points that were marked
- B) The ratio of the number of true positives to the total number of boundaries
- C) The percentage of marked boundary points that are actual boundaries
- D) The ratio of the number of true positives to the number of false positives

Clear selection

If all of our data points are in R^2, which of the following clustering algorithms can handle clusters of arbitrary shape? 1 point

- (a) k-means.
- (b) KNN.
- (c) EM with a gaussian mixture model.
- (d) mean-shift.

Clear selection

# AI Generated MCQs by Liangji

Thursday, May 16, 2024 4:23 AM

What is the primary goal of Principal Component Analysis (PCA) in data analysis? 1 point

- (●) A) To reduce the dimensionality of a dataset while retaining the most important information
- ( ) B) To increase the dimensionality of a dataset for better visualization
- ( ) C) To find the line of best fit through a data distribution
- ( ) D) To calculate the expected values of different data points

Clear selection

In PCA, what does an eigenvalue indicate? 1 point

- ( ) A) The position of the mean point in the data distribution
- ( ) B) The strength of the relationship between two data points
- ( ) C) The average distance between data points
- (●) D) The amount of variance in a specific direction represented by an eigenvector

Clear selection

What is the role of the covariance matrix in PCA? 1 point

- ( ) A) It indicates the points of highest variation in the data
- ( ) B) It provides a measure of the average values for each variable
- (●) C) It measures how different variables in the data are related, indicating the strength of these relationships
- ( ) D) It describes the shape of the data distribution

Clear selection

How does PCA achieve dimensionality reduction? 1 point

- (●) A) By selecting a subset of eigenvectors that account for the majority of variance and projecting data onto these directions
- ( ) B) By discarding variables with lower correlation
- ( ) C) By increasing the number of variables and then combining similar ones
- ( ) D) By averaging the data points and removing outliers

Clear selection

What is the primary purpose of Active Shape Models (ASM)? 1 point

- ( ) A) To generate random shapes for data augmentation
- (●) B) To allow for non-rigid shape matching, enabling shapes to be scaled, rotated, and translated based on a training dataset
- ( ) C) To perform rigid transformations on a given shape
- ( ) D) To create 3D models from 2D shapes

Clear selection

In ASM, what are the parameters used to define the position and orientation of a shape within an image? 1 point

- (●) A) Scale (s), rotation angle (θ), translation (r), and shape parameter (b)
- ( ) B) Translation (t), scale (s), orientation (o), and brightness (b)
- ( ) C) Rigid transformation parameters only
- ( ) D) Scale (s), rotation angle (θ), and color (c)

Clear selection

How is the mean shape calculated in ASM? 1 point

- A) By applying PCA to derive the principal component and then projecting the shapes onto it
- B) By calculating the covariance matrix of the shape points
- C) By finding the shape that occurs most frequently in the dataset
- D) By averaging the positions of all the corresponding feature points in the dataset

Clear selection

What is the purpose of the neighborhood search in the ASM shape-fitting process? 1 point

- A) To identify the original shape in the training dataset
- B) To determine the ideal shape based on a non-rigid transformation
- C) To find better locations for the feature points by searching along the normals to the model curve for strong edges
- D) To adjust the color and texture of the shape

Clear selection

Why might it be necessary to repeat the shape-fitting process in ASM until convergence? 1 point

- A) Because the texture and color of the shape require fine-tuning
- B) Because the shape-fitting process involves trial and error
- C) Because the initial suggested points might not describe a valid shape, requiring adjustments to match a more accurate model
- D) Because ASM must iterate to maximize the size of the shapes

Clear selection

Which of the following is a common application of Active Shape Models? 1 point

- (●) A) Medical imaging, such as positioning an artificial hip
- ( ) B) Creating 3D animations for virtual characters
- ( ) C) Detecting moving objects in video streams
- ( ) D) Generating synthetic images for data augmentation

Clear selection

If you dont normalize your data to have zero mean, then the first principal component found via PCA is likely to be uninformative. 1 point

- (●) True
- ( ) False

Clear selection

# AI Generated MCQs by Liangji

Thursday, May 16, 2024 4:55 AM

What is a key advantage of using local features in image processing and computer vision? 1 point

- (●) A) Local features increase robustness to occlusions and intra-category variations by focusing on distinct regions of the image
- ( ) B) Local features provide a global representation of the entire image
- ( ) C) Local features eliminate the need for complex transformations and normalization
- ( ) D) Local features reduce the image resolution for simpler processing

Clear selection

What characteristics are desirable in local features used in image processing? 1 point

- ( ) A) Ability to represent the entire image in a single descriptor
- (●) B) High information content, invariant to changes in viewpoint and illumination, reducing computational burden
- ( ) C) High sensitivity to changes in illumination and viewpoint to detect subtle differences
- ( ) D) Specific to a single category of images

Clear selection

What is the general approach to using local features in image processing? 1 point

- ( ) A) Find large areas of similar color, compute color histograms, and apply transformations
- ( ) B) Extract all pixels, apply a Gaussian blur, and identify regions of interest
- (●) C) Find interest points, define regions around them, normalize, compute descriptors, and then match descriptors
- ( ) D) Compute gradients, find global maxima, and create feature maps

Clear selection

What is an important requirement for region extraction in local feature-based image processing? 1 point

- A) Invariant to changes in color and texture
- B) Specific to a predefined set of shapes or structures
- C) Repeatable, invariant to translation, rotation, and scale, robust to out-of-plane transformations and lighting variations
- D) Dependent on the spatial coherence of pixel values

Clear selection

Which of the following is NOT an essential property of local features in image processing? 1 point

- A) Locality, making them robust to occlusion and clutter
- B) The ability to represent global information about the image
- C) Sufficient quantity to cover the object
- D) Distinctiveness, with "interesting" structure

Clear selection

In the Harris Corner Detector, how is a corner detected in an image? 1 point

- A) By calculating the Laplacian of the image and identifying regions with high values
- B) By computing the eigenvalues of the gradient structure tensor and finding regions where both eigenvalues are small
- C) By computing the eigenvalues of the gradient structure tensor and finding regions where both eigenvalues are large
- D) By using edge detection algorithms and then finding intersections

Clear selection

What is the Singular Value Decomposition (SVD) of a matrix? 1 point

- A) A decomposition of a matrix into eigenvalues and eigenvectors
- B) A process that reduces the dimensionality of a matrix by removing low-rank components
- C) A decomposition of a matrix into two orthogonal matrices
- D) A factorization of a matrix into three matrices: U, D, V^T, where U and V are orthogonal matrices and D contains singular values on its diagonal.

Clear selection

Which components are used to construct the gradient structure tensor in the Harris Operator? 1 point

- A) The gradients in the x and y directions, calculated using partial derivatives
- B) The intensity values of the image
- C) The position and color information of each pixel
- D) The magnitude and angle of the gradients

Clear selection

What does the corner response function measure in the Harris Operator? 1 point

- A) The likelihood that a given point in the image is a corner based on the eigenvalues of the gradient structure tensor
- B) The edge strength at a given point in the image
- C) The change in intensity along an edge
- D) The number of corners within a specified region

Clear selection

What is the role of the tuning parameter in the Harris Operator's corner response function? 1 point

- (●) A) It is a constant used to balance the determinant and trace terms, affecting sensitivity to corners
- ( ) B) It determines the window size for calculating the gradient structure tensor
- ( ) C) It controls the level of smoothing applied to the image before corner detection
- ( ) D) It sets the threshold for identifying corners

Clear selection

What is the purpose of automatic scale selection in image processing? 1 point

- (●) A) To find a characteristic scale or region size that is invariant to changes in image scale
- ( ) B) To rescale an image to a fixed size for easier processing
- ( ) C) To normalize pixel values to consistent scale
- ( ) D) To detect edges and corners within an image

Clear selection

Which of the following methods is commonly used in automatic scale selection to detect characteristic scales? 1 point

- (●) A) Laplacian of Gaussian, which measures the second derivatives of an image
- ( ) B) Gaussian Blur, which smooths the image to reduce noise
- ( ) C) Sobel Operator, which calculates gradients in the image
- ( ) D) Simple linear iterative clustering, which segments the image for scale selection

Clear selection

What is a characteristic property of the Laplacian used in scale selection? 1 point

- (●) A) It is a scalar, meaning it does not contain orientation information
- ( ) B) It is a vector, containing both magnitude and direction
- ( ) C) It is noise-resistant, providing stable results in noisy images
- ( ) D) It represents the gradient of an image in two dimensions

Clear selection

What is a drawback of using the Laplacian in image processing? 1 point

- (●) A) It is sensitive to noise, which can lead to false positives and requires smoothing
- ( ) B) It requires high computational resources
- ( ) C) It is not suitable for multi-scale analysis
- ( ) D) It loses the second derivative information

Clear selection

Why is it important to compare local maxima across different scales in automatic scale selection? 1 point

- (●) A) To ensure that a local maximum at one scale remains a maximum at other scales, indicating a robust feature
- ( ) B) To check if the region size changes with different scales
- ( ) C) To confirm the stability of the scale normalization process
- ( ) D) To identify if the detected features are consistent with expected results

Clear selection

What is the Difference of Gaussians (DoG) in the context of image processing? 1 point

- A) A high-pass filter that detects edges in an image
- B) The subtraction of an image from its Gaussian-blurred version
- (●) C) The difference between two Gaussian-blurred versions of an image at different scales, often used to approximate the Laplacian of Gaussian (LoG)
- D) A combination of Gaussian and Laplacian filters to enhance image features

Clear selection

Why is the Difference of Gaussians (DoG) used as an approximation for the Laplacian of Gaussian (LoG)? 1 point

- A) It provides higher accuracy in detecting image features
- (●) B) It does not require explicit computation of second derivatives and can be more computationally efficient
- C) It generates less noise in the resulting image
- D) It does not require multiple image scales

Clear selection

How does a Gaussian pyramid play a role in the Difference of Gaussians (DoG)? 1 point

- (●) A) The Gaussian pyramid provides Gaussian-blurred versions of an image at multiple scales, allowing for fast computation of DoG at various levels
- B) It smooths the image, reducing noise before applying DoG
- C) It decomposes the image into its frequency components to create the DoG
- D) It is used to create a sequence of images with increasing levels of blurring

Clear selection

What is the Harris-Laplace detector used for in image processing? 1 point

- (●) A) It detects scale-invariant interest points by combining the Harris Corner Detector with Laplacian-based scale selection
- ( ) B) It detects edges by combining the Harris Corner Detector with the Laplacian of Gaussian
- ( ) C) It finds lines in an image at multiple scales
- ( ) D) It identifies regions of high intensity in an image

Clear selection

Why is it important for local feature descriptors to have orientation information? 1 point

- (●) A) It helps to achieve invariance to rotation, allowing feature matching even when objects are rotated
- ( ) B) It increases the scale invariance of the descriptor
- ( ) C) It allows descriptors to be used in different lighting conditions
- ( ) D) It helps to reduce noise in the feature descriptor

Clear selection

What is the SIFT descriptor, and what does it capture in an image? 1 point

- ( ) A) It is a gradient-based descriptor that emphasizes edge information
- ( ) B) It is a scale-based descriptor that captures variations in pixel intensity
- ( ) C) It is a rotation-based descriptor that focuses on the orientation of objects
- (●) D) It is an invariant feature descriptor that captures information in a region around a detected interest point, including orientation and scale

Clear selection

How is the orientation normalization process handled in the SIFT descriptor? 1 point

- (●) A) By quantizing the gradient orientations into bins and aligning the dominant gradient direction to a canonical direction, then normalizing other gradients with respect to this direction
- ( ) B) By computing the magnitude of the gradients and then normalizing the descriptor based on the strongest gradients
- ( ) C) By applying a Gaussian blur to the image to smooth the gradients
- ( ) D) By reorienting the entire image to point upwards.

Clear selection

What is the typical dimensionality of the SIFT descriptor for each feature point? 1 point

- ( ) A) 256 dimensions, derived from a 32 x 32 patch with 8 orientation bins each
- ( ) B) 64 dimensions, derived from a 16 x 16 patch with 4 orientation bins each
- (●) C) 128 dimensions, derived from a 16 x 16 patch divided into 4 x 4 sub-patches with 8 orientation bins each
- ( ) D) 32 dimensions, derived from a 16 x 16 patch with 2 orientation bins each

Clear selection

Why should interest points along edges be rejected in the SIFT process? 1 point

- ( ) A) Interest points along edges typically contain less information
- (●) B) Interest points along edges may cause ambiguity in orientation and localization, leading to unreliable descriptors
- ( ) C) Edge-based interest points can lead to increased computational complexity
- ( ) D) Edge-based interest points are not invariant to scale

Clear selection

What happens when the threshold for feature matching with SSD is increased? 1 point

- (●) A) More matches are considered good, potentially leading to more false positives
- ( ) B) Fewer matches are considered good, potentially leading to more false negatives
- ( ) C) It becomes easier to identify exact matches with high confidence
- ( ) D) The computational complexity of feature matching is reduced

Clear selection

What is the effect of decreasing the threshold for feature matching with SSD? 1 point

- ( ) A) More matches are considered good, leading to more false positives
- ( ) B) It results in higher computational costs
- ( ) C) It improves the matching accuracy without any drawbacks
- (●) D) Fewer matches are considered good, potentially leading to more false negatives

Clear selection

What does an ROC curve represent in the context of feature matching? 1 point

- (●) A) The relationship between true positive rate (TPR) and false positive rate (FPR) across different thresholds for feature matching
- ( ) B) The Receiver Operating Characteristic, showing the accuracy of feature matching with various algorithms
- ( ) C) The curve showing the relationship between precision and recall
- ( ) D) The curve that represents the change in distance between descriptors over time

Clear selection

What is the Area Under the Curve (AUC) in an ROC curve, and what does it indicate? 1 point

- A) It represents the area between the ROC curve and the baseline, with lower values indicating better performance
- B) It is a measure of the overall performance of the feature matching algorithm, with higher values indicating better performance
- C) It is the area covered by matched features in the image, with higher values indicating better matching
- D) It is the area within which all good matches are found, with larger areas indicating more accurate matching

Clear selection

In which scenario would ratio distance be more useful than SSD for feature matching? 1 point

- A) When computational efficiency is a primary concern
- B) When noise sensitivity needs to be minimized
- C) When you want to reduce the likelihood of false positives by ensuring a significant gap between the best match and the second-best match
- D) When you need to detect the exact scale and orientation of a feature

Clear selection

Why are local features robust to clutter and occlusion in object recognition? 1 point

- A) They rely on global image features that are unaffected by occlusion.
- B) They use advanced edge detection algorithms to ignore clutter.
- C) They do not require segmentation and can handle partial visibility.
- D) They are optimized for low-resolution images.

Clear selection

# AI Generated MCQs by Liangji

Thursday, May 16, 2024 5:31 PM

How does picture formation in the eye vary from image formation in a camera? 1 point

- ○ a) Fixed focal length
- ○ b) Varying distance between lens and imaging plane
- ○ c) No difference
- ◉ d) Variable focal length

Clear selection

We need ___ geometry because recovery of structure from one image is inherently ambiguous 1 point

- ○ single-view
- ○ line-view
- ◉ multi-view
- ○ geometrical correct view

Clear selection

___ axis is a line form the camera center perpendicular to the image plane. 1 point

- ○ Main Principal axis = optical axis which is perpendicular to the image plane
- ◉ Principal
- ○ Epipolar
- ○ Perpendicular

Clear selection

We can write the epipolar ___ in terms of unknown normalized coordinates 1 point

- (●) constraints
- ( ) base line
- ( ) pole
- ( ) plane

Clear selection

If we scale the entire scene by some factor k and, at the same time, scale the camera matrices by the factor of 1/k, the projections of the scene points in the image ____ 1 point

- (●) remain exactly the same
- ( ) change randomly
- ( ) convert to 3D
- ( ) enlarges by factor of k

Clear selection

In a simple stereo system, all ___ rays converge to one point on a plane located at the focal length f. 1 point

- (●) parallel
- ( ) perpendicular
- ( ) diverged
- ( ) intersect

Since the camera optical axis is parallel,
All rays from a point at infinity is considered parallel

Clear selection

What is the primary goal of stereo vision in computer vision? 1 point

- A) To extract 2D information from 3D scenes.
- B) To extract 3D information from multiple 2D views of a scene.
- C) To extract shading and texture details from 2D images.
- D) To extract focus and motion information from 2D images.

Clear selection

What is the first step in stereo reconstruction? 1 point

- A) Compute disparity
- B) Rectify images
- C) Calibrate cameras
- D) Estimate depth

Clear selection

What is the principle of triangulation in stereo vision? 1 point

- A) Finding the intersection of two rays to estimate depth.
- B) Aligning two cameras to compute disparity.
- C) Using perspective transform to estimate camera pose.
- D) Calibrating cameras to calculate internal parameters.

Clear selection

What is one of the main challenges in finding correspondences in stereo analysis? 1 point

- (●) A) Sparse set of points in the scene.
- ( ) B) Overlapping images.
- ( ) C) Image region for the matches are similar in appearance
- ( ) D) Too little occlusion

Clear selection

Which camera model is typically used in a simple stereo system? 1 point

- (●) A) Pinhole camera
- ( ) B) Orthographic camera
- ( ) C) Fish-eye lens camera
- ( ) D) Panoramic camera

Clear selection

What is the key benefit of the epipolar constraint in stereo vision? 1 point

- (●) A) It reduces the correspondence problem to a 1D search along conjugate epipolar lines.
- ( ) B) It limits the need for calibration in stereo camera systems.
- ( ) C) It allows for precise calibration of camera orientation.
- ( ) D) It provides a method to estimate depth.

Clear selection

What do epipolar lines represent in the context of stereo vision? 1 point

- A) The boundary lines of the stereo camera's field of view.
- B) Lines indicating the perspective transform in 3D reconstruction.
- (●) C) Lines where the corresponding points in two images must lie.
- D) The lines joining the camera centers in a stereo setup.

Clear selection

What is the essential matrix used for in stereo vision? 1 point

- (●) A) To relate corresponding image points between calibrated cameras.
- B) To determine the shading and texture within an image.
- C) To calculate the exact distance between two cameras.
- D) To extract 3D information from multiple 2D views of a scene.

Clear selection

What is a common issue in finding stereo correspondences due to the simple constraint? 1 point

- A) Excessive calibration requirements.
- (●) B) Low-contrast or textureless image regions.
- C) Difficulty in determining the epipoles.
- D) Excessive depth information.

Clear selection

What is the goal of rectification in stereo vision? 1 point

- ○ A) To adjust the camera orientation to align with the scene.
- ○ B) To transform images to obtain 3D information.
- ◉ C) To transform images so that the image planes are parallel.
- ○ D) To calculate the calibration parameters of the cameras.

Clear selection

What is one of the challenges with matching horizontal edges in stereo vision? 1 point

- ◉ A) They are poorly localized along the epipolar lines.
- ○ B) They often contain high gradient values
- ○ C) They do not correspond to significant structures.
- ○ D) They are not affected by lighting changes.

Clear selection

What is a characteristic of sparse correspondence search in stereo vision? 1 point

- ○ A) It focuses on finding depth information across the entire image.
- ◉ B) It make use of feature descriptors.
- ○ C) It requires extensive calibration between the cameras.
- ○ D) It uses epipolar constraint to extract pixel-by-pixel correspondence.

Clear selection

What is a potential advantage of dense correspondence search in stereo vision? 1 point

- (●) A) It provides more depth estimates, useful for surface reconstruction.
- ( ) B) It focuses on finding only key features within the image.
- ( ) C) It uses feature descriptors for accurate localization.
- ( ) D) It is less likely to fail in textureless regions.

Clear selection

What do extrinsic parameters in camera calibration represent? 1 point

- ( ) A) The internal characteristics of the camera, such as pixel size and focal length.
- ( ) B) The settings for camera exposure and white balance.
- (●) C) The rotation and translation that describe the camera's position and orientation.
- ( ) D) The camera's external casing and mounting structure.

Clear selection

Which of the following is NOT an intrinsic parameter in camera calibration? 1 point

- ( ) A) Focal length
- ( ) B) Pixel size
- (●) C) Rotation matrix
- ( ) D) Origin offset

Clear selection

What is typically used in camera calibration to derive accurate intrinsic and extrinsic parameters? 1 point

- (●) A) A calibration target with accurately measured feature positions.
- ( ) B) A collection of randomly chosen images.
- ( ) C) A single 3D point for reference.
- ( ) D) A calibration algorithm that relies on pixel brightness variations.

Clear selection

Which of the following best describes intrinsic parameters in camera calibration? 1 point

- (●) A) They relate pixel coordinates to image coordinates.
- ( ) B) They represent the rotation and translation of the camera.
- ( ) C) They describe the camera's field of view.
- ( ) D) They focus on the external structure of the camera.

Clear selection

Which set of parameters is required to calculate image coordinates from pixel coordinates in camera calibration? 1 point

- ( ) A) Rotation matrix, translation vector, and pixel size.
- ( ) B) Shutter speed, focal length, and exposure time.
- ( ) C) Field of view, aperture, and image resolution.
- (●) D) Focal length, origin offset, and pixel size ratio.

Clear selection

What can be determined if a camera is not calibrated but at least 8 correspondences in the scene are known? 1 point

- (●) A) Relative 3D positions (up to a scale factor).
- ( ) B) Absolute 3D positions with high precision.
- ( ) C) The exact intrinsic parameters of the camera.
- ( ) D) The exact extrinsic parameters of the camera.

Clear selection

What can be inferred from knowing the pixel size in camera calibration? 1 point

- (●) A) The physical dimensions of each pixel in the image sensor.
- ( ) B) The intrinsic parameters such as rotation matrix and translation vector.
- ( ) C) The color depth and resolution of the image.
- ( ) D) The len size and focal length

Clear selection

# AI Generated MCQs by Liangji

Thursday, May 16, 2024 7:02 PM

Which of the following is **not** a property of affine transformations? 1 point

- A) Parallel lines remain parallel.
- B) Ratios are preserved.
- C) Lines map to line.
- D) Origin always maps to origin.
- E) Closed under composition

Clear selection

What is a key difference between homographies and affine transformations? 1 point

- A) Homographies do not necessarily preserve parallel lines, while affine transformations do.
- B) Affine transformations are closed under composition, while homographies are not.
- C) Homographies are limited to scaling and rotation, while affine transformations include shear and translation.
- D) Affine transformations map lines to line, while homographies map curves to lines.

Clear selection

What is the main goal of the RANSAC (RAndom SAmple Consensus) algorithm in computer vision? 1 point

- A) To find the most efficient affine transformation for image stitching.
- B) To identify a model that best fits the majority of data points in the presence of outliers.
- C) To compute global warping transformations with high accuracy.
- D) To detect features in images and align them.

Clear selection

What is meant by global warping in the context of image transformations? 1 point

- (●) A) A transformation that applies to all points in an image with a few parameters.
- ( ) B) A transformation that applies to all points in an image with an extensive parameters.
- ( ) C) A transformation that focuses on rotating the entire image.
- ( ) D) A transformation that changes the pixel intensity based on neighboring pixels.

Clear selection

What is the purpose of computing a homography using RANSAC in the context of creating panoramas? 1 point

- (●) A) To align images by removing the effects of perspective distortion.
- ( ) B) To correct for color differences between images.
- ( ) C) To detect prominent features in the images.
- ( ) D) To adjust the exposure and contrast of the combined image.

Clear selection

In alpha blending, which of the following is true about the value of the parameter **α** 1 point

- ( ) A) It controls the level of sharpness in the combined image.
- ( ) B) It affects the contrast of the final blended image.
- (●) C) It determines the proportion of blending between two images.
- ( ) D) It indicates the number of RANSAC iterations needed.

Clear selection

Which of the following operations is NOT linear in a 2D coordinate system? 1 point

- (●) A) Translation
- ( ) B) Rotation
- ( ) C) Scaling
- ( ) D) Shear

Clear selection

What is the primary difference between forward warping and inverse warping? 1 point

- (●) A) Forward warping can result in holes, while inverse warping resamples from the source image.
- ( ) B) Inverse warping is more computationally efficient than forward warping.
- ( ) C) Forward warping based interpolation, while inverse warping does not.
- ( ) D) Inverse warping normalizes pixel values, while forward warping does not.

Clear selection

Which of the following is a common challenge in drone self-localization? 1 point

- ( ) A) Non-existent satellite visibility
- ( ) B) Multipath reflections
- ( ) C) GPS spoofing
- (●) D) All of the above

Clear selection

In visual localization, what is the role of homography? 1 point

- A) It detects the edges in the drone image.
- B) It enhances the contrast of the drone image for better recognition.
- (●) C) It maps pixels from the drone image to the Geo image.
- D) It calculates the drone's speed and trajectory.

Clear selection

What is a common issue when using visual feature matching for drone self-localization? 1 point

- A) It relies heavily on GPS data for accuracy.
- B) It does not work well with non-Google images.
- C) It always requires multiple iterations for accurate results.
- (●) D) It may be affected by weather conditions or changes in image features.

Clear selection

# AI Generated MCQs by Liangji

Thursday, May 16, 2024 10:09 PM

What is the purpose of the Generalized Hough Transform in object recognition? 1 point

- ○ A) To create a feature descriptor for object detection.
- ○ B) To categorize objects into different classes based on texture.
- ◉ C) To cast votes for the location, scale, and orientation of a model object.
- ○ D) To determine the shape and structure of an object in an image.

Clear selection

What is the primary characteristic of the Bag-of-Words model in object recognition? 1 point

- ○ A) It uses an ordered representation, containing spatial relationships between features.
- ◉ B) It uses an orderless representation, ignoring spatial relationships between features.
- ○ C) It contains multiple probability relationships between features.
- ○ D) Background and foreground are not mixed when bag covers whole image

Clear selection

What is a key advantage of K-Nearest Neighbour (K-NN) classification? 1 point

- ○ A) It requires less training data compared to other methods.
- ◉ B) It naturally handles multi-class cases and is flexible in terms of features and distance metrics.
- ○ C) It does not need extensive storage for data.
- ○ D) It is highly efficient in terms of computational requirements.

Clear selection

What assumption underlies the Naïve Bayes model in object recognition? 1 point

- (●) A) Each feature is conditionally independent given the class.
- ( ) B) The features are related to each other through a complex network.
- ( ) C) The classes are all equally likely in the training data.
- ( ) D) The prior probabilities are biased toward rare classes.

Clear selection

# AI Generated MCQs by Liangji

Thursday, May 16, 2024 10:18 PM

What is the primary function of stumps in the AdaBoost algorithm? 1 point

- (●) A) To act as weak classifiers in a forest of decision stumps.
- ( ) B) To create a bootstrapped dataset for building random forests.
- ( ) C) To represent small subsets of the training data.
- ( ) D) To compute visual features for face detection.

Clear selection

What is a key characteristic of Random Forests compared to a single decision tree? 1 point

- ( ) A) They rely solely on bootstrapped datasets without randomization.
- (●) B) They build multiple decision trees using random subsets of variables at each layer.
- ( ) C) They are more prone to overfitting than a single decision tree.
- ( ) D) They require fewer computational resources than a single decision tree.

Clear selection

What is a key advantage of the Viola Jones Cascade Classifier for face detection? 1 point

- ( ) A) It requires minimal training data for face detection.
- ( ) B) It uses a simple linear model to detect faces.
- (●) A) It can reject non-faces early on, increasing detection speed.
- ( ) D) It can detect non-frontal faces.

Clear selection

What is an integral image, and why is it useful in face detection? 1 point

- (●) A) It allows for efficient computation of features used in face detection.
- ( ) B) It provides a high-contrast version of the original image.
- ( ) C) It identifies key facial landmarks for detection.
- ( ) D) It simplifies the AdaBoost algorithm by reducing its complexity.

Clear selection

In the AdaBoost algorithm, how are sample weights adjusted after creating a stump? 1 point

- (●) A) Weights for incorrect samples are increased and weights for correct samples are decreased to affect later stumps.
- ( ) B) Weights for correct samples are increased and weights for incorrect samples are decreased to improve accuracy.
- ( ) C) Weights remain unchanged regardless of the stump's performance.
- ( ) D) Weights are randomly reassigned for greater diversity.

Clear selection

Given sufficiently many weak classifiers, AdaBoosting is guaranteed to get perfect accuracy on the training set no matter what the training data looks like. 1 point

- ( ) True
- (●) False

Clear selection

AdaBoosting always makes your algorithm generalize better. 1 point

- ○ True
- ◉ False

Clear selection