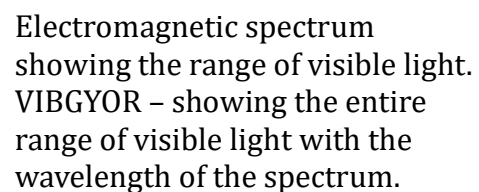
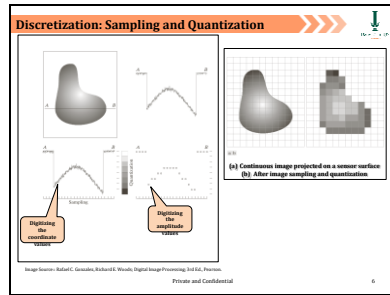


Slide 4



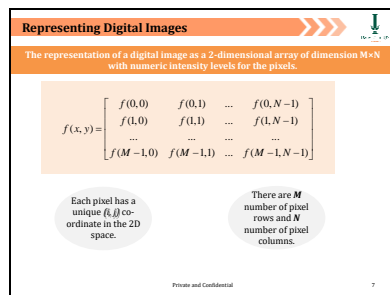
**Image formation:** Light falls on a source; light is reflected from the surface of the object; the reflected ray is captured by a device or falls on the eyes of a viewer making the object visible.

### Slide 6



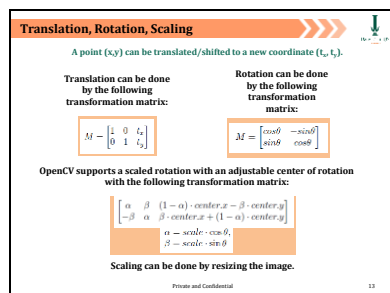
Sampling rate decides how far one sample point is spaced from the adjacent point. While the quantization levels decide how much information can be encoded in every pixel.

### Slide 7



2D matrix representing the spatial extent and coordinates of the pixels. The values of each cell in the matrix represents the intensity value of the corresponding pixel.

### Slide 13



This slide shows the various transformation matrices for different operations – translation, rotations and scaling.

### Slide 14

**Computing Image Statistics**

- The **sample mean** ( $m_A$ ) of an image  $A$  ( $N \times M$ ):
 
$$m_A = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} A(i,j)}{NM}$$
- The **sample variance** ( $\sigma_A^2$ ) of  $A$ :
 
$$\sigma_A^2 = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (A(i,j) - m_A)^2}{NM}$$
- The **sample standard deviation**,  $\sigma_A = \sqrt{\sigma_A^2}$ .

Mean and standard deviation are very common statistic calculated on an image.

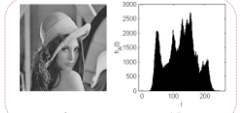
They are of huge importance and being used in most of the statistical analysis of the image such as equalization, matching, texture analysis with moments, segmentation with binarization etc.

Private and Confidential 14

Computing basic statistics from a 2D matrix (image)

### Slide 15

**Image Histogram**



Let  $S$  be a set and define  $\#S$  to be the cardinality of this set, i.e.,  $\#S$  is the number of elements of  $S$ .

The histogram  $h_A(i)$  ( $i=0, \dots, 255$ ) of the image  $A$  is defined as:

$$h_A(i) = \# \{ (i,j) \mid i(i,j) = i, i=0, \dots, N-1, j=0, \dots, M-1 \}$$

Note that:  $\sum_{i=0}^{255} h_A(i) = \text{Number of pixel in } A$

Private and Confidential 15

Histogram is the frequency distribution of intensity levels in an image.


### Slide 16

**Spatial and Intensity Resolution**

**Spatial resolution**


A measure of the smallest discernible detail in an image

- Stated with *line pairs per unit distance*, dots (*pixels*) per unit distance, dots per inch (*dpi*)
- Can be expressed in terms of the number of pixels in the image



**Intensity or Gray Level resolution**

- The smallest discernible change in intensity level
- Stated with 8 bits, 12 bits, 16 bits, etc.
- Can be expressed in terms of number of gray levels in the image



Private and Confidential 16

Spatial resolution represents the smallest discernible detail in an image. Represented by dpi. While intensity resolution represents the number of bits per pixel.

Slide 18

**Spatial Resolution**

- Re-sampling (by replication) to the original size



**Effect of Up-sampling from a low resolution image**  
 (a) 1024x1024 8-bit image (b) 512x512 resampled into 1024x1024 by row and column duplication (c) through (f) 256x256 → 128x128 → 64x64 → 32x32


Image Source: Rafael C. Gonzales, Richard E. Woods, Digital Image Processing, 3rd Ed., Pearson.

Private and Confidential

Self-explanatory. See the amount of degradation in subsequent images.

Slide 21

**Memory Requirements for Storing**



If an image is  $N \times N$  pixels

- Each pixel has up to 24 gray/intensity levels
- We can reduce spatial or gray level intensity to reduce the size of an image
- Total number of bits stored =  $N \times N \times k$  (with no compression)


Image Source: Rafael C. Gonzales, Richard E. Woods, Digital Image Processing, 3rd Ed., Pearson.

Private and Confidential

Image size depends on number of bits assigned per pixel.

Slide 23

**OpenCV**



**OpenCV** is an Open Source Computer Vision and image processing library Originally created by Intel and maintained by Willow Garage and Itseez.

*It was officially launched in 1999 aiming at real-time computer vision.*

- Written natively in C++ - it is a cross-platform library and has support for C++, Python, Java and MATLAB.
- The library has more than 2500 optimized algorithms
- Licensed under BSD - makes it easy for businesses to utilize and modify code.
- Supports Windows, Linux, Android and Mac OS.

Private and Confidential

Introduction to OpenCV

### Slide 25

**Spatial Filtering**

A spatial filter consists of (a) a neighborhood, and (b) a predefined operation

Linear spatial filtering of an image of size  $M \times N$  with a filter of size  $m \times n$  is given by the expression

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

Private and Confidential

Defining spatial domain filtering mathematically.

The definition of the filter  $w(s, t)$  will vary for the actual operation (e.g. smoothing, median filter etc.).

### Slide 26

**Spatial Convolution**

The convolution of a filter  $w(x, y)$  of size  $m \times n$  with an image of  $f(x, y)$ , denotes as  $w(x, y) \star f(x, y)$

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

Where  $m = 2a+1$  and  $n = 2b+1$

Private and Confidential

Defining convolution: taking dot product of corresponding elements and summing up all the values.

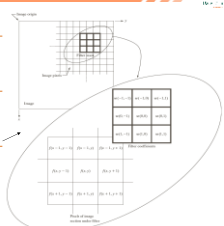
Dot product is the element-wise product of the terms and summing up all such product terms into a single value.

Optionally the values are normalized by dividing the sum of all the values in the filter.

### Slide 27

**Convolution**

- i) The mask/filter/kernel acts like a sliding window.
- ii) The mask is placed/aligned on top of an image.
- iii) Value of the center pixel in the output image is decided by taking the sum of element-wise product of the image pixel intensity and the value of the mask at that location.



Private and Confidential

Visually showing the process of convolution with a sliding window (the window represents the convolution filter).

Slide 28

**Smoothing**

Smoothing filters are used for blurring and for noise reduction.

Blurring is used in removal of small details and bridging of small gaps in lines or curves.

The general implementation for filtering an  $M \times N$  image with a weighted averaging filter of size  $m \times n$  is given

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

where  $m = 2a + 1$ ,  $n = 2b + 1$ .

Private and Confidential

Image smoothing operations with a smoothing filter (e.g. averaging filter).

Slide 29

**Smoothing by Averaging Filters**

Two  $3 \times 3$  smoothing (averaging) filter masks.

The constant multipliers is the sum of the values of all the coefficients in the filter as required to compute average.

We can change the size of the filter and increase the effect of smoothing (blurring). e.g. instead of using a  $3 \times 3$  mask, we can use a  $5 \times 5$  or  $9 \times 9$  or  $13 \times 13$  and so on.

Private and Confidential

Image smoothing operations with a smoothing filter (e.g. averaging filter).

Slide 30

**Smoothing by Averaging Filters**

See the effect of larger smoothing masks.

Larger masks  $\rightarrow$  more smoothing (blurring) effect as more neighboring pixels participate in the averaging process.

(a) Original image (b) is smoothed with a  $3 \times 3$  filter (b). (c) to (f) with increasing mask dimensions  $5 \times 5$ ,  $9 \times 9$ ,  $15 \times 15$ ,  $35 \times 35$ .

Private and Confidential

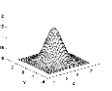
Results of smoothing (averaging filter).

### Slide 32

#### Gaussian Filtering

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

(a) 2D Gaussian Function



(b) Distribution with mean (0,0) and standard deviation (sigma)

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
1	4	7	4	1
4	16	26	16	4

(c) 2D Gaussian filter mask with sigma=1

The Gaussian filter is a type of image smoothing (blurring) filter that uses the Gaussian function for calculating the transformations to apply to each pixel in an image.

Sigma controls the amount of blurring.

Private and Confidential 11

Showing the equation of a Gaussian mask; its plot in 2D; and a representative 5x5 Gaussian filter.

### Slide 33

#### Results of Smoothing with Gaussian Filter






Gaussian Filter

Private and Confidential 12

Result of smoothing with a Gaussian filter.

### Slide 34

#### Median Filtering

1	4	7	4	1
4	6	8	9	4
7	26	17	11	7
1	14	6	5	1
4	16	26	16	4

Consider this 5x5 image. Place a 3x3 median filter on top of it.

3x3 Median filter

The value for the center pixel of the output is decided as follows:

Sort the pixel values falling within the spatial extent of the 3x3 filter (highlighted region)

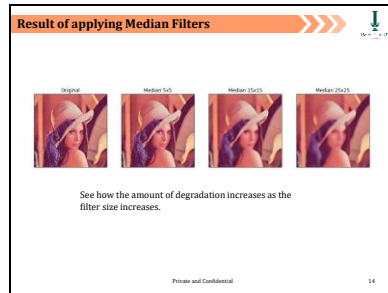
Sorted order:      5   6   6   8   9   11   14   17   26

Take the median value: the middle one in the list which is 9.

Private and Confidential 13

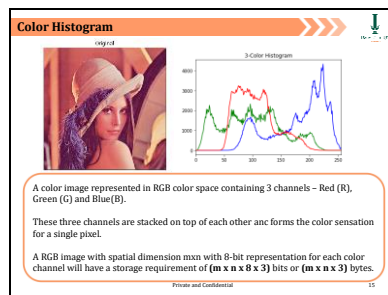
Defining a median filter.

Slide 35



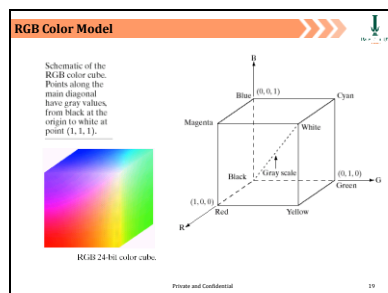
Results of applying a median filter.

Slide 36



Plotting the color histogram – plot of red, green, blue channels of an RGB image.

Slide 40



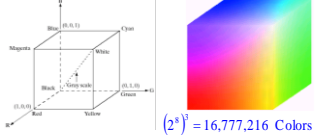
Color cube defining a 3D space of all possible color that is a combination of the Red, Green and Blue color components. Here R, G and B represent three axis in a 3 dimensional coordinate system. We can consider a unit cube and all the colored pixel is a data point in this 3D space. Each pixel is having a specific value for each of the R, G and B component.



### Slide 41

**Pixel Depth**

- Pixel depth: the number of bits used to represent each pixel in RGB space
- Full-color image: 24-bit RGB color image
  - (R, G, B) = (8 bits, 8 bits, 8 bits)



$(2^8)^3 = 16,777,216$  Colors

The diagonal in the color cube represents the different shades of gray.


Private and Confidential

Color cube defining a 3D space of all possible color that is a combination of the Red, Green and Blue color components. Number of possible color to be represented depends on how many bits are assigned to each pixel for representing each of the basic R, G and B components.

### Slide 42

**CMY model (+Black = CMYK)**

- CMY: secondary colors of light, or primary colors of pigments
- Used to generate hardcopy output


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


Private and Confidential

CMY a subtractive color model.

### Slide 43

**Color Space Conversion RGB to HSV/YCbCr**



There are many color space representation possible for a digital image. One such color model is HSV.

**HSV – (Hue, Saturation, Value)**

- Is often used by artists because it is often more natural to think about a color in terms of hue and saturation than in terms of additive or subtractive color components.

**YCbCr color space**

- Is another widely used color model for digital video. In this format, luminance information is stored as a single component (Y), and chrominance information is stored as two color-difference components (Cb and Cr). Cb represents the difference between the blue component and a reference value. Cr represents the difference between the red component and a reference value.

Private and Confidential

Choosing an RGB image and the same image converted to HSV color-space.

Slide 44

**RGB to HSV Conversion**

General Strategies for Processing Color Images - Compute luminance (weighted average of RGB), process intensity matrix

- Define the following values  
 $C_{high} = \max(R, G, B)$ ,  $C_{low} = \min(R, G, B)$ , and  
 $C_{mag} = C_{high} - C_{low}$
- Find **saturation** of RGB color components ( $C_{max} = 255$ )  
 $S_{HSV} = \begin{cases} \frac{C_{mag}}{C_{high}} & \text{for } C_{high} > 0 \\ 0 & \text{otherwise} \end{cases}$
- And luminance value  
 $V_{HSV} = \frac{C_{high}}{C_{max}}$

Private and Confidential

How to convert RGB image to HSV color space?

Slide 45

**RGB to HSV Conversion**

- Normalize each component using  
 $R' = \frac{C_{high} - R}{C_{mag}}$   $G' = \frac{C_{high} - G}{C_{mag}}$   $B' = \frac{C_{high} - B}{C_{mag}}$
- Calculate preliminary hue value  $H'$  as  
 $H' = \begin{cases} B' - G' & \text{if } R = C_{high} \\ R' - B' + 4 & \text{if } G = C_{high} \\ G' - R' + 4 & \text{if } B = C_{high} \end{cases}$
- Finally, obtain final hue value by normalizing to interval [0,1]  
 $H_{HSV} = \frac{1}{6} \begin{cases} (H' + 6) & \text{for } H' < 0 \\ H' & \text{otherwise} \end{cases}$

Private and Confidential

How to convert RGB image to HSV color space?

Slide 46

**HSV**



Original RGB Image



HSV values in grayscale

Private and Confidential

Example

## How to convert HSV image to RGB color space?

## How to convert HSV image to RGB color space?

## How to convert HSV image to RGB color space?

## How to convert HSV image to RGB color space?

Tow other color models – YUV and YCbCr.

Tow other color models – YUV and YCbCr.

Slide 50

**YUV**

- Basis for color encoding in analog TV in north america (NTSC) and Europe (PAL)
- Y components computed from RGB components as
 
$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$
- UV components computed as:
 
$$U = 0.492 \cdot (B - Y) \quad \text{and} \quad V = 0.877 \cdot (R - Y)$$

Private and Confidential

Computing YUV components from RGB.

Slide 51

**YUV and RGB**

- Entire transformation from RGB to YUV
 
$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$
- Invert matrix above to transform from YUV back to RGB
 
$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.000 & 0.000 & 1.140 \\ 1.000 & -0.395 & -0.581 \\ 1.000 & 2.032 & 0.000 \end{pmatrix} \begin{pmatrix} Y \\ U \\ V \end{pmatrix}$$

Private and Confidential

Transformation matrix for conversion between YUV and RGB.

Slide 53

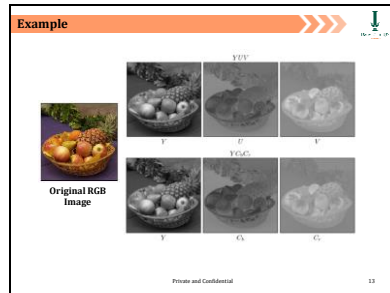
**YCbCr and RGB**

- ITU recommendation BT.601 specifies values:
 
$$w_R = 0.299, \quad w_G = 0.114, \quad w_B = 1 - w_R - w_G = 0.587$$
- Thus the transformation
 
$$\begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$
- And the inverse transformation becomes
 
$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.000 & 0.000 & 1.403 \\ 1.000 & -0.344 & -0.714 \\ 1.000 & 1.773 & 0.000 \end{pmatrix} \begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix}$$

Private and Confidential

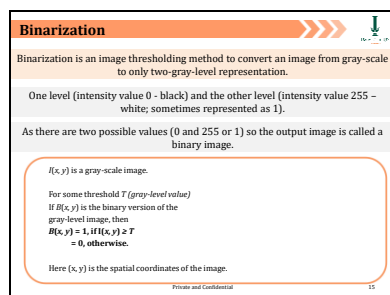
The formula for computing Y is sometimes used to convert a RGB image to grayscale just by considering the Y component.

Slide 54



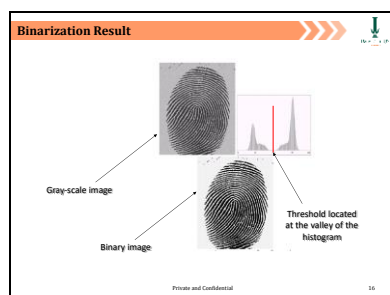
Example: Converting a RGB image to YUV and YCbCr.

Slide 56



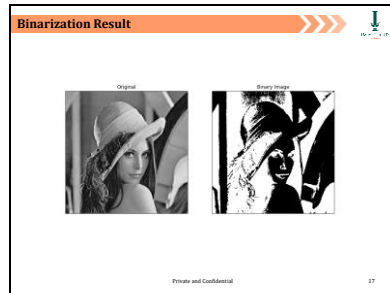
Algorithm for Binarization.

Slide 57



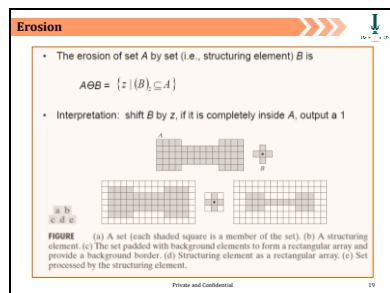
Example of image binarization.

Slide 58



Results of binarization.

Slide 60



Explaining different binary morphological operations.

Slide 72

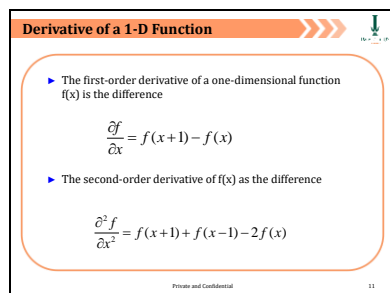


Image derivate approximated with difference of intensity values of consecutive pixels.

### Slide 73

**Image Gradient**

For function  $f(x, y)$ , the gradient of  $f$  at coordinates  $(x, y)$  is defined as

$$\nabla f = \text{grad}(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The magnitude of vector  $\nabla f$ , denoted as  $M(x, y)$

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

Magnitude of Gradient Image

Private and Confidential

Defining image gradient vector and its magnitude.

$g_x$  is the derivative of the image w.r.t  $x$

$g_y$  is the derivative of the image w.r.t  $y$

$\text{grad}(f)$  is the vector containing the  $g_x$  and  $g_y$  as component.

$M(x, y)$  is the magnitude of the vector. Standard Euclidean norm formula.

### Slide 74

**Image Gradient Vector**

The magnitude of vector  $\nabla f$ , denoted as  $M(x, y)$

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

$$M(x, y) \approx |g_x| + |g_y|$$

Reduce computation time by taking sum of absolute values rather than computing square and square root.

In a 2D image, take the difference of intensity of adjacent pixels as shown.

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$z_6$
$z_7$	$z_8$	$z_9$

$$M(x, y) \approx |z_8 - z_5| + |z_6 - z_5|$$

Private and Confidential

Approximating image gradient with difference of adjacent pixel-intensities in a 3x3 image region. Refer to the previous slide for a definition of gradient.

### Slide 75

**Sobel's Mask for Edge Detection**

$z_1$	$z_2$	$z_3$
$z_4$	$z_5$	$z_6$
$z_7$	$z_8$	$z_9$

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

Sobel's approximation, coefficients turn into the weights in the mask

Horizontal and Vertical Sobel masks detecting horizontal and vertical edges in the images

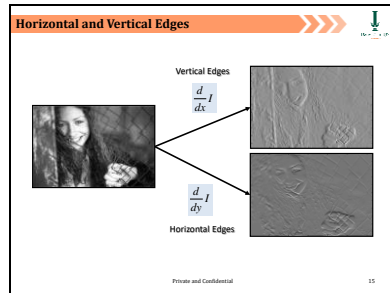
Sobel Operators

$$M(x, y) \approx (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) + (z_5 + 2z_6 + z_9) - (z_3 + 2z_4 + z_7)$$

Private and Confidential

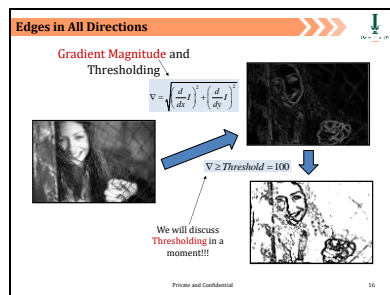
Approximating image gradient with difference of adjacent pixel-intensities in a 3x3 image region.

Slide 76



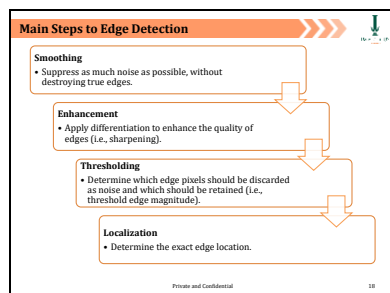
Results of image gradients – derivative w.r.t x and y.

Slide 77



Results of image gradients – derivative w.r.t x and y.  
Taking the magnitude of the gradient vector.

Slide 79



Algorithm for edge detection.



### Slide 80

**Sobel Operator**

- The Sobel Operator is a discrete differentiation operator. It computes an approximation of the gradient of an image intensity function.
- The Sobel Operator combines **Gaussian smoothing** and differentiation.
- Assuming that the image to be operated is  $I$ :

- We calculate two **derivatives**:
  - Horizontal Changes for Detecting Vertical Edges:** Vertical derivative approximation of an image  $I$  is denoted as  $G_x$ . This is computed by convolving  $I$  with a kernel with odd size. For example for a kernel size of 3,  $G_x$  would be computed as:
 
$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ +1 & 0 & +1 \end{bmatrix} * I$$

Private and Confidential

Defining the Sobel's operator for edge detection.

### Slide 81

**Sobel Operator**

- Vertical Changes for Detecting Horizontal Edges:** Horizontal derivative approximation of an image  $I$  is denoted as  $G_y$ . This is computed by convolving  $I$  with a kernel with odd size. For example for a kernel size of 3,  $G_y$  would be computed as:
 
$$G_y = \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ +1 & -2 & +1 \end{bmatrix} * I$$
- At each point of the image we calculate an approximation of the **gradient** in that point by combining both results above:
 
$$G = \sqrt{G_x^2 + G_y^2}$$

- Sometimes the following **approximation** is used to reduce computational requirements:
 
$$G = |G_x| + |G_y|$$

Private and Confidential

Defining the Sobel's operator for edge detection.  
Gradient may be approximated by taking the sum of absolute values of the gradients computed in the X and Y directions in order to reduce computational complexity.

### Slide 82

**Results of Sobel Edge Detection**

Original



Gray Scale



Sobel Edge Map

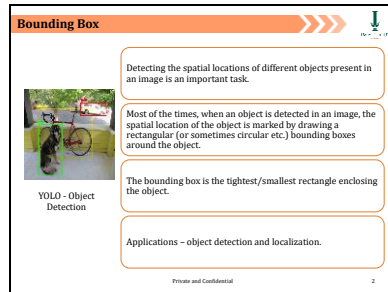


Private and Confidential

Results of edge detection.

Slide 83

### Bounding Box



Detecting the spatial locations of different objects present in an image is an important task.

Most of the times, when an object is detected in an image, the spatial location of the object is marked by drawing a rectangular (or sometimes circular etc.) bounding boxes around the object.

The bounding box is the tightest/smallest rectangle enclosing the object.

Applications – object detection and localization.

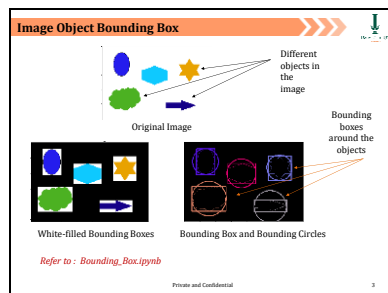
YOLO - Object Detection

Private and Confidential

Object detection from images – demarcation of spatial extents of individual object with a suitable bounding box.

Slide 84

### Image Object Bounding Box



Different objects in the image

Original Image

Bounding boxes around the objects

White-filled Bounding Boxes

Bounding Box and Bounding Circles

Refer to : Bounding\_Box.ipynb

Private and Confidential

Detecting and drawing bounding boxes around each object in the image.

Slide 88

### Template Matching by Correlation

#### Normalized Correlation Coefficient

- Normalized correlation coefficient:

$$\gamma(x,y) = \frac{\sum_s \sum_t [w(s,t) - \bar{w}] [f(x+s,y+t) - \bar{f}_w]}{\left\{ \sum_s \sum_t [w(s,t) - \bar{w}]^2 \sum_s \sum_t [f(x+s,y+t) - \bar{f}_w]^2 \right\}^{\frac{1}{2}}}$$

- $\gamma(x,y)$  takes values in  $[-1,1]$ .
- The maximum occurs when the two regions are identical.

Private and Confidential

Computing normalized correlation coefficient for image matching.  
 $w(\cdot)$  is the sliding window – representing an image patch with which matching is done – template image.

$f()$  is the original base image. We are looking for a match in the base image.

$f(x+s, y+t)$  is the part of the image under the window  $w(s,t)$  at a given time.

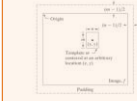
$w\_bar$  is the average of the intensities of the pixels inside a given window – template image.

$f_{xy\_bar}$  is the average of the intensities of the pixels of the base image falling inside a given window.

Slide 89

**Template Matching by Correlation**

**Matching by Correlation**

$$\gamma(x, y) = \frac{\sum_{s,t} [w(s, t) - \bar{w}] [f(x+s, y+t) - \bar{f}_{xy}]}{\left( \sum_{s,t} [w(s, t) - \bar{w}]^2 \sum_{s,t} [f(x+s, y+t) - \bar{f}_{xy}]^2 \right)^{\frac{1}{2}}}$$


- It is robust to changes in the amplitudes.
- Normalization with respect to scale and rotation is a challenging task.

Private and Confidential 7

Computing normalized correlation coefficient for image matching.

$w(\cdot)$  is the sliding window – representing an image patch with which matching is done – template image.

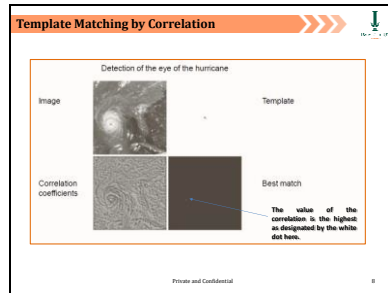
$f()$  is the original base image. We are looking for a match in the base image.

$f(x+s, y+t)$  is the part of the image under the window  $w(s, t)$  at a given time.

$w\_bar$  is the average of the intensities of the pixels inside a given window – template image.

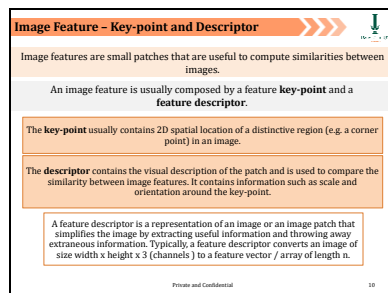
$f_{xy\_bar}$  is the average of the intensities of the pixels of the base image falling inside a given window.

Slide 90



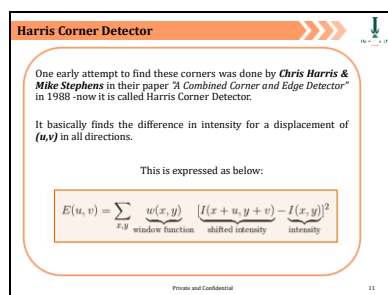
Example of matching by correlation.  
This is showing the result of the operation stated in the previous slide.

Slide 92



Defining image feature – key-point in an image and key-point descriptor.

Slide 93



Find the difference in intensity displacement in the image.

### Slide 94

**Harris Corner Detector**

Window function is either a rectangular window or Gaussian window which gives weights to pixels underneath.

We have to maximize this function  $E(u,v)$  for corner detection. That means, we have to maximize the second term.

Applying Taylor Expansion to above equation and using some mathematical steps we get the final equation as:

$$E(u,v) \approx [u \ v] M \begin{bmatrix} u \\ v \end{bmatrix}$$

and

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

Private and Confidential

Maximizing  $E(u,v)$  and computing gradients.

### Slide 95

**Harris Corner Detector**

Here,  $I_x$  and  $I_y$  are image derivatives in x and y directions respectively. It can be easily found out using **Sobel** mask.

After this, they created a score, basically an equation, which will determine if a window can contain a corner or not.

$$R = \det(M) - k(\text{trace}(M))^2$$

where

$$\det(M) = \lambda_1 \lambda_2$$

$$\text{trace}(M) = \lambda_1 + \lambda_2$$

$\lambda_1 + \lambda_2$  are the Eigen values of M

Private and Confidential

Determine the Eigen values.

Determinant is a single value that can be computed from a square non-singular matrix.

Trace of a square matrix is the summation of all the elements in the main diagonal.

These are standard linear algebra terms. Students are expected to have some background of linear algebra and calculus (10+2 maths.)

### Slide 96

**Harris Corner Detector**

So the values of these Eigen values decide whether a region is corner, edge or flat.

- When  $|R|$  is small, which happens when  $\lambda_1 + \lambda_2$  are small, the region is flat.
- When  $R < 0$ , which happens when  $\lambda_1 \gg \lambda_2 \gg 0$  or vice versa, the region is edge.
- When  $R$  is large, which happens when  $\lambda_1 + \lambda_2$  are large and  $\lambda_1 \sim \lambda_2$ , the region is a corner.

Result of Harris Corner Detection is a grayscale image with these scores. Thresholding with a suitable threshold gives the corners in the image.

Private and Confidential

Make decisions whether a given point is a corner or not.

### Slide 98

#### Histograms of Oriented Gradients (HoG)

HoG\* is a feature descriptor based on edge orientation histograms.

It is computed by considering the orientation of the gradient in localized region of an image.

It is easy to express the rough shape of the object with HoG.

On the contrary, rotation and scale changes are not supported (we may use SIFT for this).

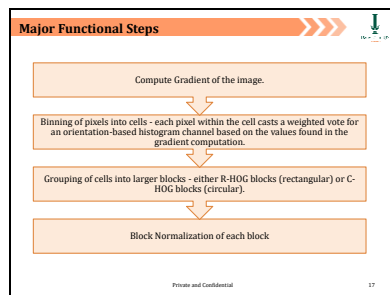
It is robust to variations in geometry and illumination changes.

\*Navneet Dalal and Bill Triggs; "Histogram of oriented gradients for human detection", CVPR 2005.

Private and Confidential

HoG: Original paper was published in 2005 and used for human detection. Major steps involved in HoG are stated in the subsequent slides.

### Slide 99



Major steps involved in HoG.

### Slide 100




#### Image Gradient

- Convolution with kernels
 

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

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

  - Magnitude =  $\sqrt{g_x^2 + g_y^2}$
  - Angle =  $\arctan(g_y/g_x)$
- Directional change in intensity

Private and Confidential

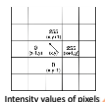
Computing image gradient.

### Slide 101

#### Image Gradient

- The gradient is calculated at each pixel.
- The gradient is a vector with **magnitude m** and **orientation  $\theta$**  represented by the change in the luminance.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1} \left( \frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)} \right)$$


Intensity values of pixels

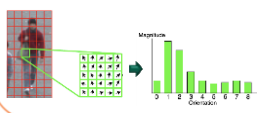
Private and Confidential

Computing image gradient – magnitude and orientation direction w.r.t a pixel located at (x,y).

### Slide 102

#### Binning

- To create a histogram of gradient orientations for each cell (e.g. 5x5 pixels) using the gradient magnitude and orientation.
- The orientations are evenly spaced over **Nine** separate bins
- Gradient magnitude added to bin



Bin	Range
Bin 0	0 - 39
Bin 1	40 - 79
Bin 2	80 - 119
Bin 3	120 - 159
Bin 4	160 - 199
Bin 5	200 - 239
Bin 6	240 - 279
Bin 7	280 - 319
Bin 8	320 - 359

Private and Confidential

Gradient orientation binning.

### Slide 103

#### Descriptor Blocks


Cells are grouped together into larger sized spatially connected blocks (e.g. 4 cells per block).

Blocks overlap with each other. That means, each cell contributes more than once to the final descriptor.

The HOG descriptor is the concatenated vector of the components of the normalized cell histograms from all of the block regions.

Blocks typically overlap, meaning that each cell contributes more than once to the final descriptor.

Two main block geometries exist - Rectangular R-HOG blocks and Circular C-HOG blocks.



Private and Confidential

Image is divided into larger size blocks. Each block is divided into cells. Breaking down the image into such different sized spatial-extents ensure better invariance to changes that may happen during capture of the image.

Slide 104

### Block Normalization

Gradient magnitudes vary over a wide range due to local variations in the illumination and foreground-background contrast.

Normalization is performed to make the process robust against such changes in illumination.

Let  $v$  be the non-normalized vector containing all histograms in a given block.

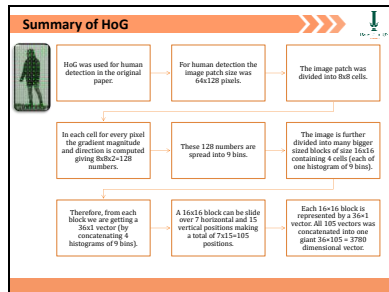
The normalization factor  $f$  can be computed as follows:

$$f = \frac{1}{\sqrt{\|v\|_2^2 + \epsilon^2}}$$

Private and Confidential

Block normalization reduces the effect of illumination changes in the surroundings. The process of normalization reduces the effect of lighting changes in the scene.

Slide 105




Summarizing the whole process for a human detection case.


Slide 106

### Applications of HoG


Human/  
Pedestrian  
detection  
from images




Object  
matching



Real-time  
Object  
Recognition



OCR



Etc..

Private and Confidential

Applications of HoG



Slide 108

**Stream Video Processing with OpenCV**

A video consists of many frames (still images) and also the audio properly in sync with the frames.

In computer vision using OpenCV, we can extract the frames from a video (only the image frames without audio).

The extracted images can further be processed like any other image using image processing and computer vision techniques.

We will see, how to capture images from a webcam using the OpenCV library.

Private and Confidential

What is a video stream?

Slide 109

**Stream Video Processing with OpenCV**

Import the library using the following statement:

```
import cv2
```

In OpenCV, in order to capture/create an Image or Video we use the **VideoCapture()** method which allows us to capture the video stream from our webcam.

The **VideoCapture()** method is accessed using the cv2 namespace as follows:

```
videoStreamObject = cv2.VideoCapture(0)
```

The above statement will create a VideoCapture object and will return it, hence we need to store the value in a variable (name is videoStreamObject).

Since a video is a stream of picture frames, using this VideoCapture object we can access the camera and retrieve each frame and display it on the screen.

```
cap, frame = videoStreamObject.read()
```

Private and Confidential

Step-by-step description of how to extract frame from a video stream captured by the webcam.

Slide 113

**Python Program**

```
import cv2

videoCaptureObject = cv2.VideoCapture(0)
while(True):
    ret, frame = videoCaptureObject.read()
    cv2.imshow('Capturing Video', frame)
    if(cv2.waitKey(1) & 0xFF == ord('q')):
        videoCaptureObject.release()
        cv2.destroyAllWindows()
```

Private and Confidential

The complete Python code for extracting frames from the video captured by a webcam.