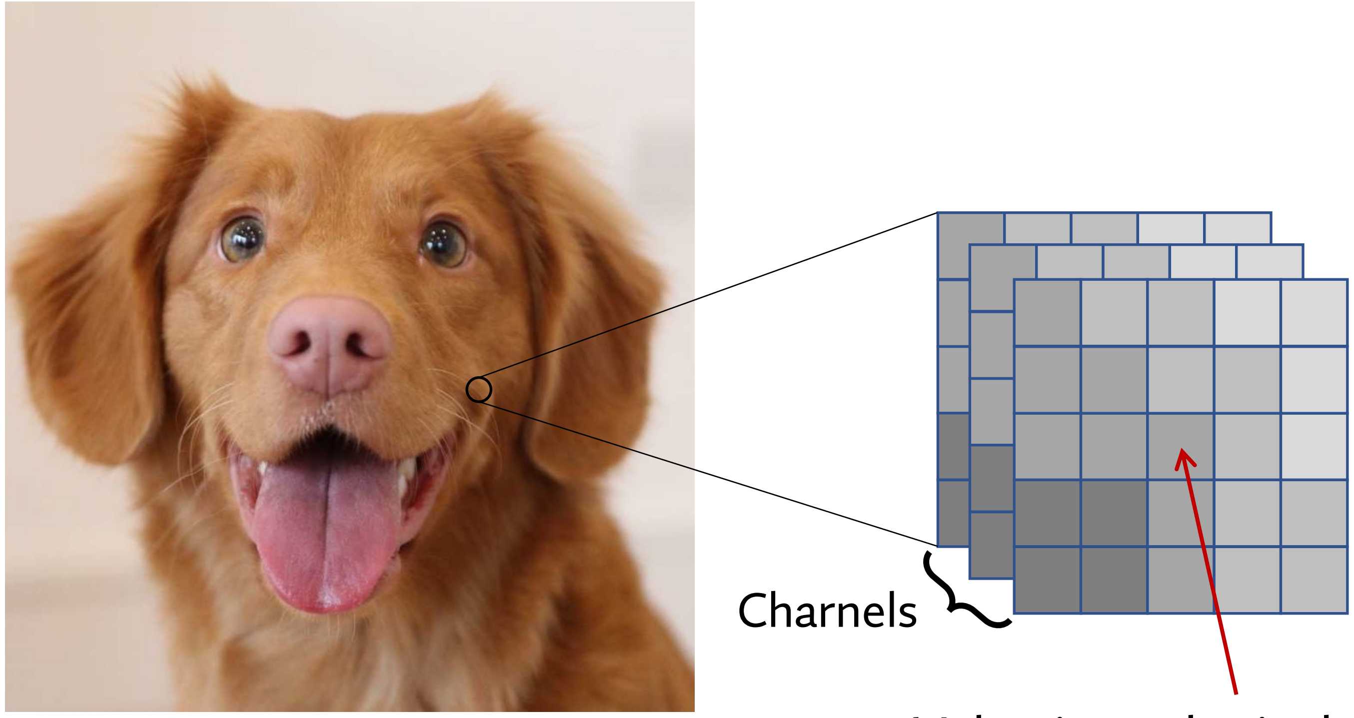
Feature Engineering

Papangkorn Inkeaw, Ph.D.

Feature Extraction

Chapter 5 (Part I) – Feature Extraction for Image Data

Image – Basic Knowledge



Value in each pixel varies between [0,255]

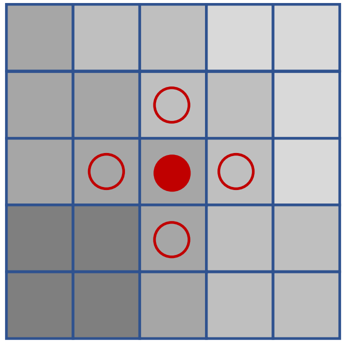
Color image contains 3 charnels.

Gray image contains 1 charnel.

Binary image contains 1 charnel which value of each pixel is 0 or 1.

Image – Basic Knowledge

Pixel Relationship


4-neighbors

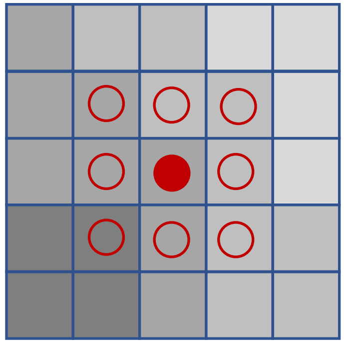
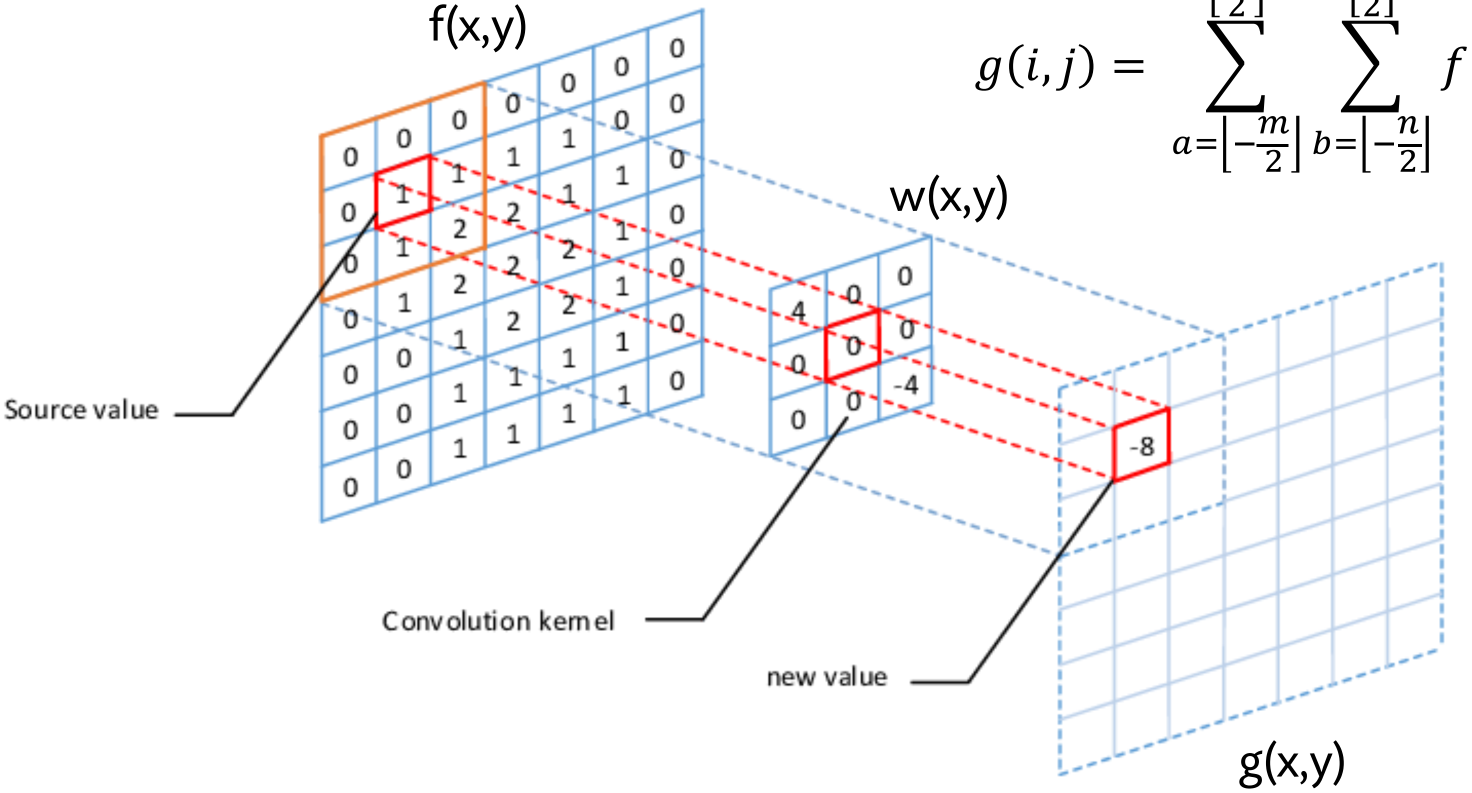

8-neighbors

Image – Basic Knowledge

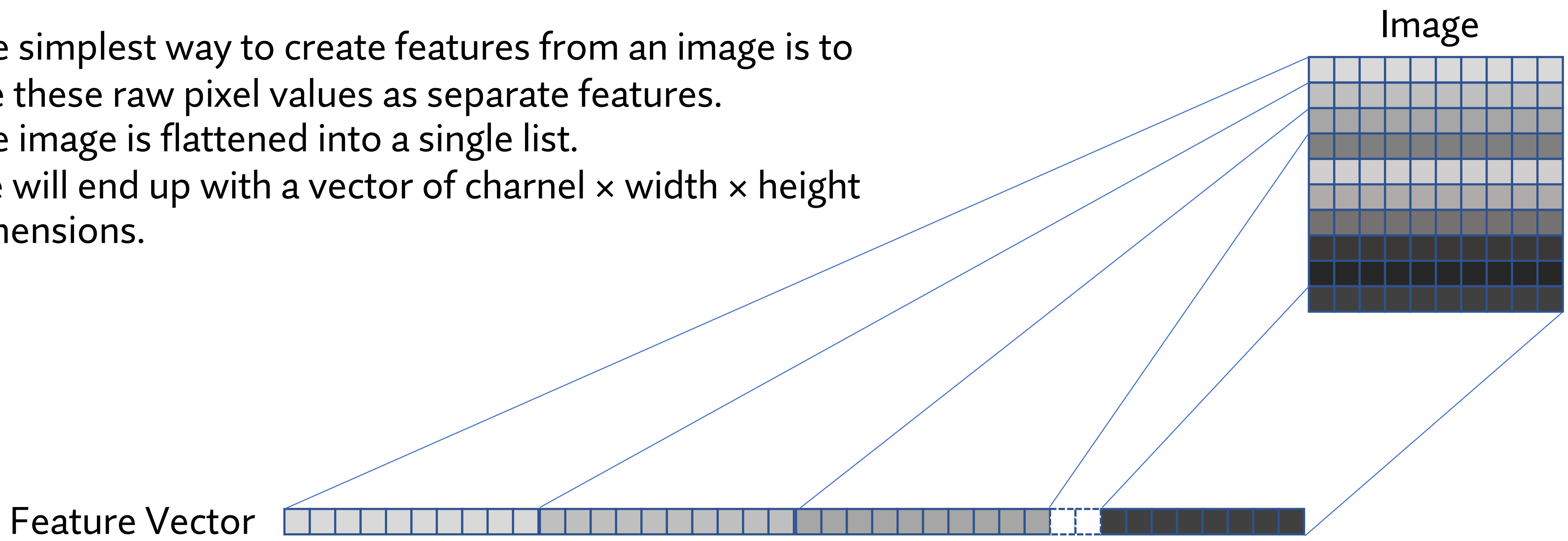
Convolution Operation

$$g(i,j) = \sum_{a=\lfloor -\frac{m}{2} \rfloor}^{\lfloor \frac{m}{2} \rfloor} \sum_{b=\lfloor -\frac{n}{2} \rfloor}^{\lfloor \frac{n}{2} \rfloor} f(i-a, j-b)\, w\left(\frac{m}{2}+a, \frac{n}{2}+b\right)$$



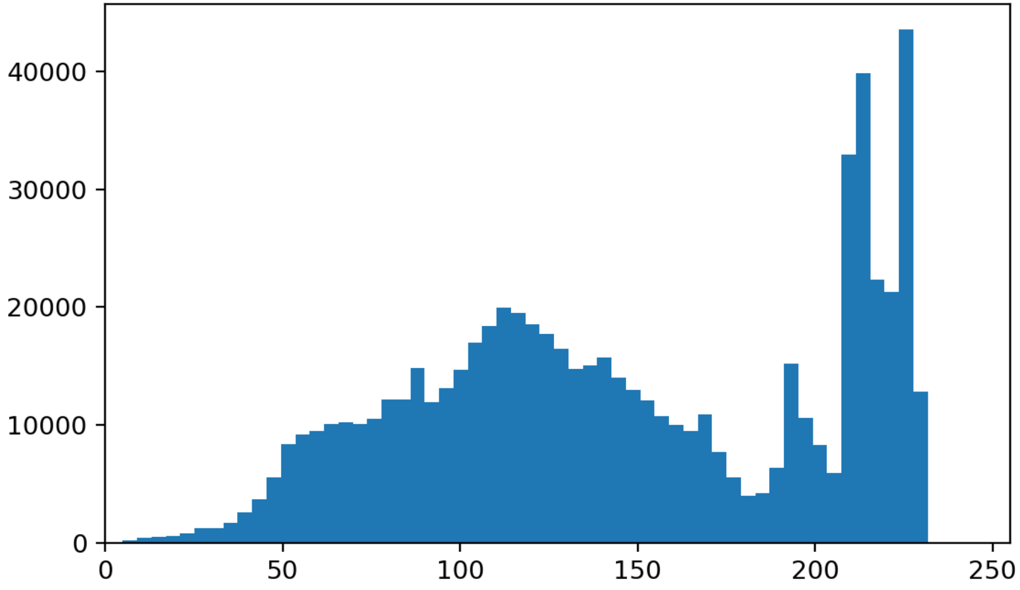
Raw Image as Feature

- The simplest way to create features from an image is to use these raw pixel values as separate features.
- The image is flattened into a single list.
- We will end up with a vector of charnel × width × height dimensions.



Color Histrogram

1. Range of the input data is split into **a number of bins**.
2. The number of data points falling in each bin are counted.
3. Normalize
4. Construct a feature vector
   1. Use the histogram as a feature
   2. Use the mean, SD, skewness of the histogram as feature



Zoning and ROI Approach

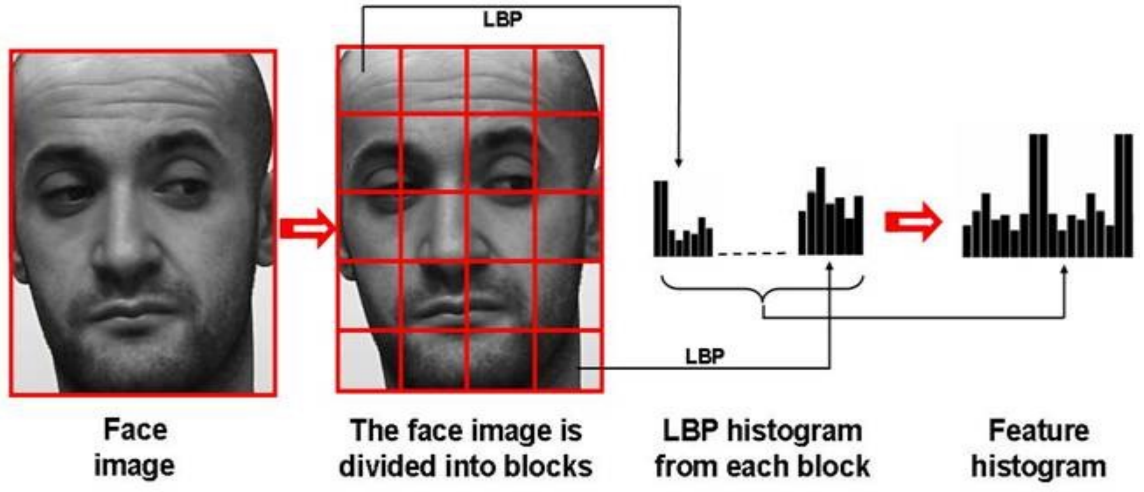
Zoning Approach

- To preseve spatial information on a feature vector.
- Divide image into blocks and then extract feature vector from each block.
- Cancatinate them together to form a final feature vector.

ROI Approach

- To ignore information from the region that we don’t want.
- Mask the region we interested.
- Extract feaures from the region of interest.

Local Binary Patterns



Source: http://www.scholarpedia.org/article/File:LBP-face.jpg

1. Convert the image to grayscale
2. For each pixel in the grayscale image
    1. select a neighborhood of size r surrounding the center pixel.
    2. calculate LBP value for this center pixel
3. Divide the image into blocks
4. For each block
    1. Compute a histogram over the output LBP array
5. Concatenate the LBP histograms obtained from each block together.

Local Binary Patterns

How to calculate LBP values

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

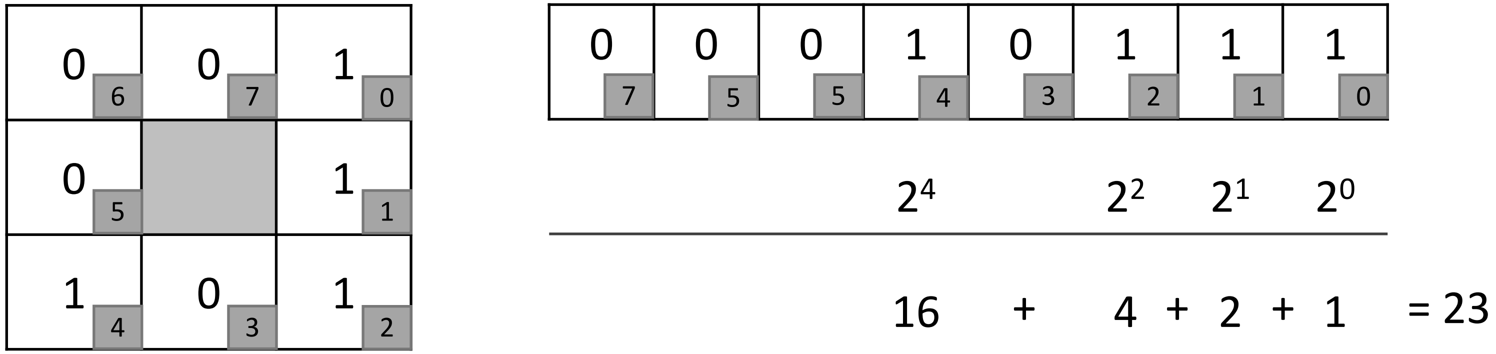
0	0	1
0		1
1	0	1

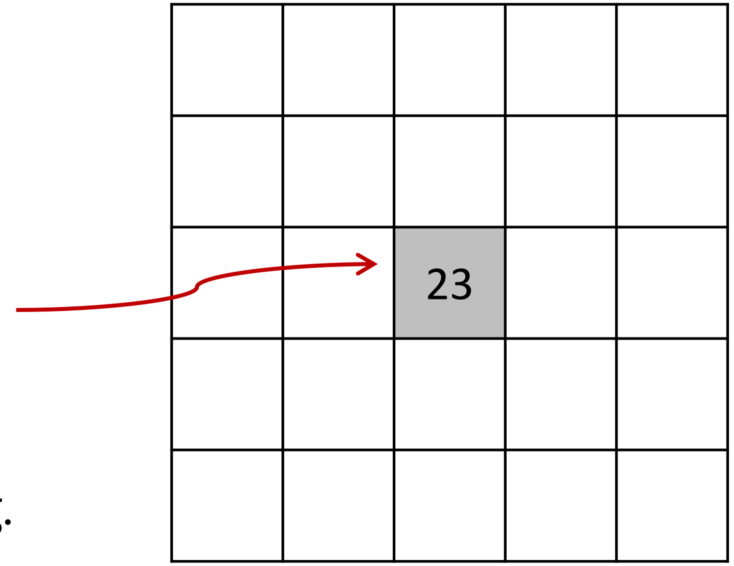
Considering each pixel as the center, assign a value to each neighbor pixel:

- If the intensity of the center pixel is greater than or equal to its neighbor, then we set the value to *1*;
- Otherwise, we set it to *0*.

Local Binary Patterns

How to calculate LBP values





LBP array

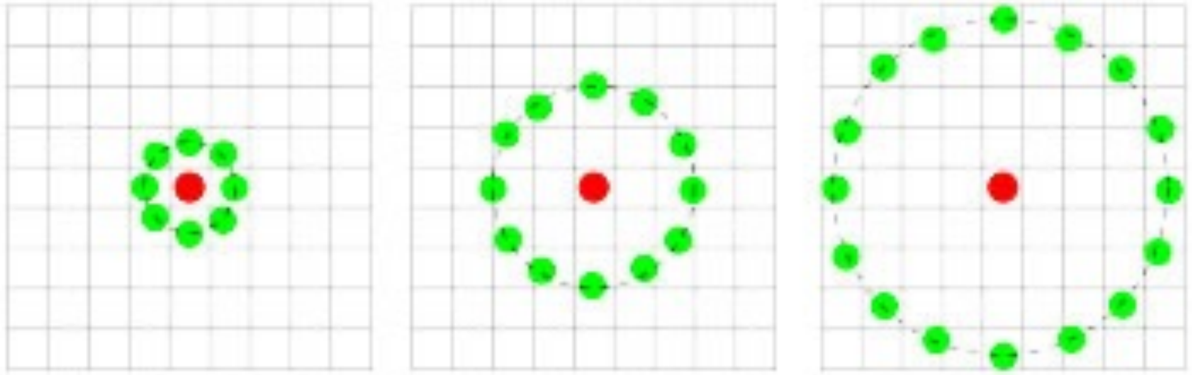
- Start at the top-right point and work our way ***clockwise*** accumulating the binary string as we go along.
- Convert this binary string to decimal.
- Store in an output array with the same width and height as the original image.

Local Binary Patterns

Neighborhood Sizes

To account for variable neighborhood sizes, two parameters were introduced:

- p: the number of points in a circularly symmetric neighborhood to consider.
- r: the radius of the circle, which allows us to account for different scales.



p=8, r=1 p=12, r=2 p=16, r=4

Local Binary Patterns

The Concept of LBP Uniformity

- A LBP is considered to be uniform if it has at most two 0-1 or 1-0 transitions.
  - 00001000 : 2 transitions -> uniform pattern
  - 10000000 : 1 transitions -> uniform pattern
  - 01010010 : 6 transitions -> non-uniform pattern
- Uniform LBP patterns add an extra level of rotation and grayscale invariance.

Moments

- A weighted average of image pixel intensities.
- The simplest kind of moment we can define is given as:

$$M = \sum_x \sum_y I(x,y)$$

- More complex moments

$$M_{pq} = \sum_x \sum_y x^p y^q I(x,y)$$

where p and q are integers.

- These moments often referred to as **raw moments.**
- Raw image moments are a projection of image $I(x,y)$ onto the basis $x^p y^q$
- These moments are capturing some notion of shape.

Moments

Central Moments

- Central moments are very similar to the raw image moments.
- Except that we subtract off the centroid from the x and y in the moment formula.

$$\mu_{pq} = \sum_{x} \sum_{y} (x - \bar{x})^p (y - \bar{y})^q I(x, y)$$

$$\bar{x} = \frac{M_{10}}{M_{00}}, \bar{y} = \frac{M_{01}}{M_{00}}$$

- Above moments are translation invariant.
- To make the moment invariant to **scale**, we need **normalized central moments** as shown below.

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\frac{(p+q)}{2}+1}}$$

Moments

Hu Moments

- A set of 7 numbers calculated using central moments.
- The 7 moments are calculated using the following formulas:
  1. $h_o = \eta_{20} + \eta_{02}$
  2. $h_1 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$
  3. $h_2 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$
  4. $h_3 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$
  5. $h_4 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$
  6. $h_5 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})]$
  7. $h_6 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$
- The first 6 moments have been proved to be invariant to translation, scale, and rotation.
- The 7th moment's sign changes for image reflection.

Moments

Moment Invariants

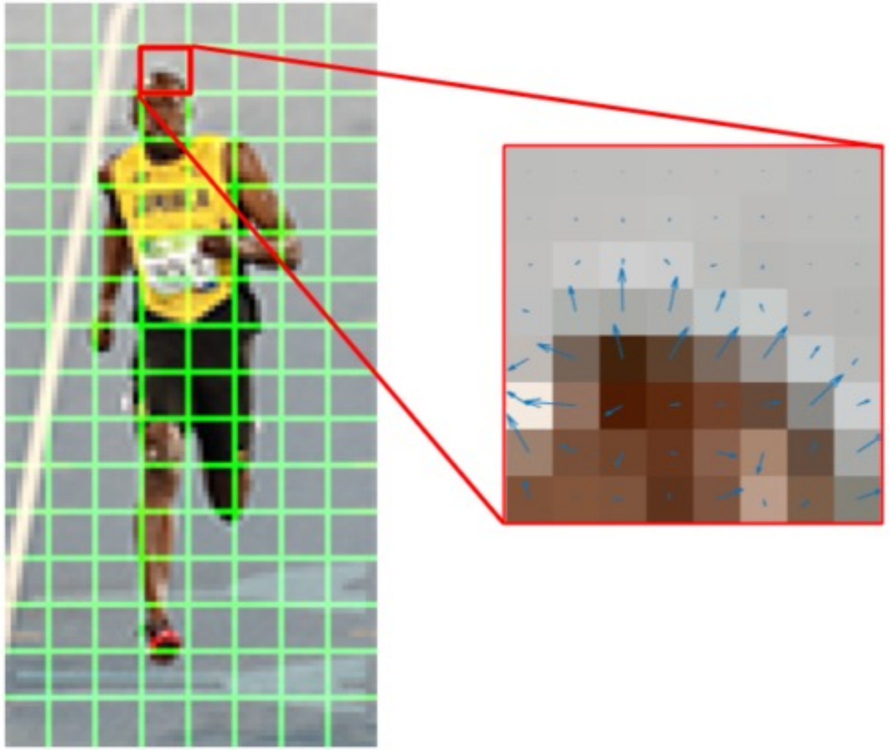
- Translation invariants

- Scale invariants

- Rotation invariants

Source: https://www.learnopencv.com/shape-matching-using-hu-moments-c-python/

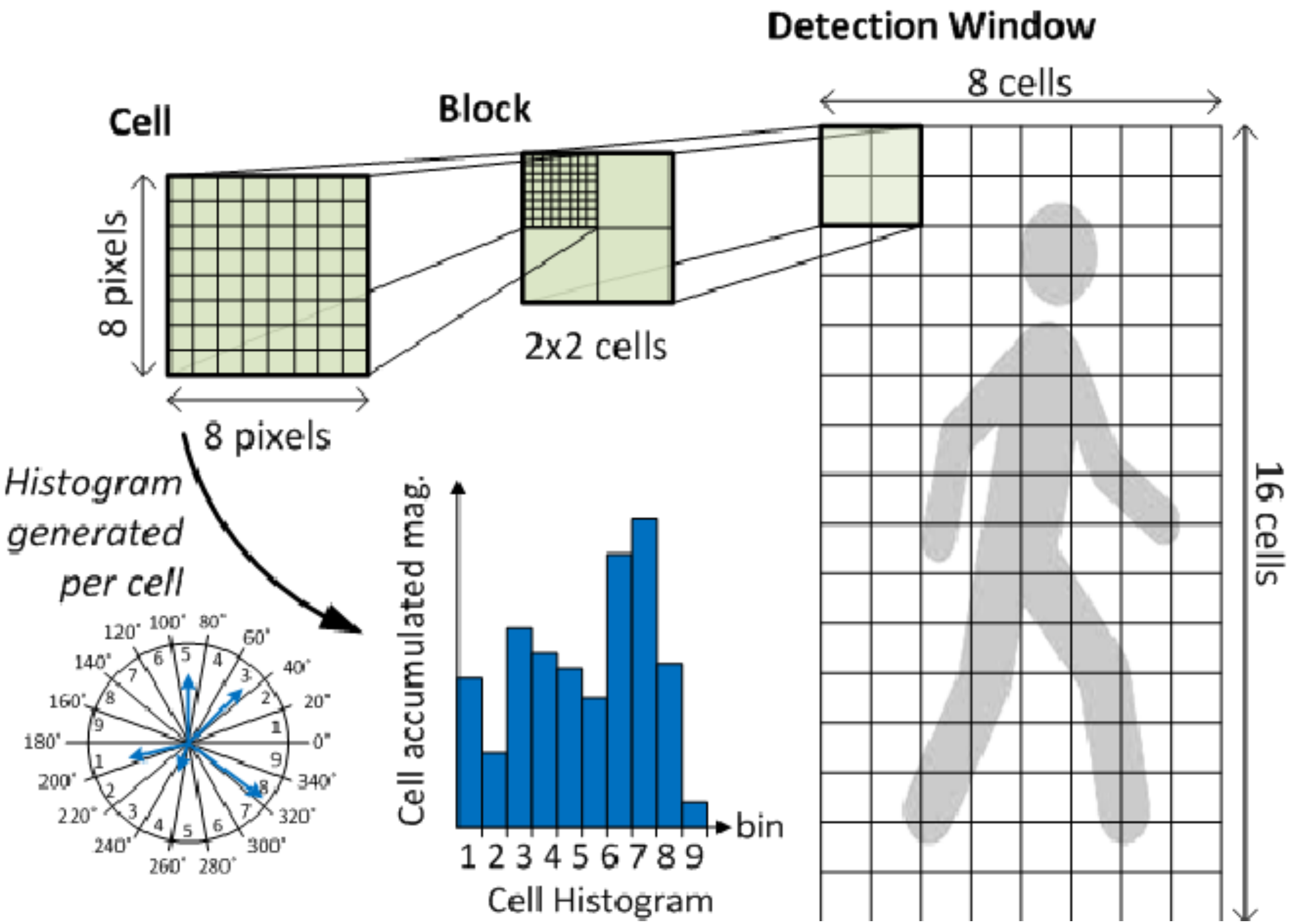
Histogram of Oriented Gradients (HOG)



Source: https://www.learnopencv.com/histogram-of-oriented-gradients/

Histogram of Oriented Gradients (HOG)

1. Calculate the Gradient Images.
2. Calculate Histogram of Gradients
3. Block Normalization
4. Calculate the HOG feature vector



Source:
https://www.semanticscholar.org/paper/Histogram-of-oriented-gradients-front-end-An-FPGA-Kelly-Siddiqui/a3a7ff0f872615dcbd39deacdd477ff9b0cca298/figure/2

Histogram of Oriented Gradients (HOG)

Calculate the Gradient Images

- Apply a convolution operation to obtain the gradient images:

$$G_x = I * H_x, \qquad G_y = I * H_y$$

- Compute the final gradient magnitude

$$|G| = \sqrt{G_x^2 + G_y^2}$$

- Compute the orientation of the gradient

$$\theta = \arctan \frac{G_y}{G_x}$$

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

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

Histogram of Oriented Gradients (HOG)

Calculate Histogram of Gradients

1 Cell

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

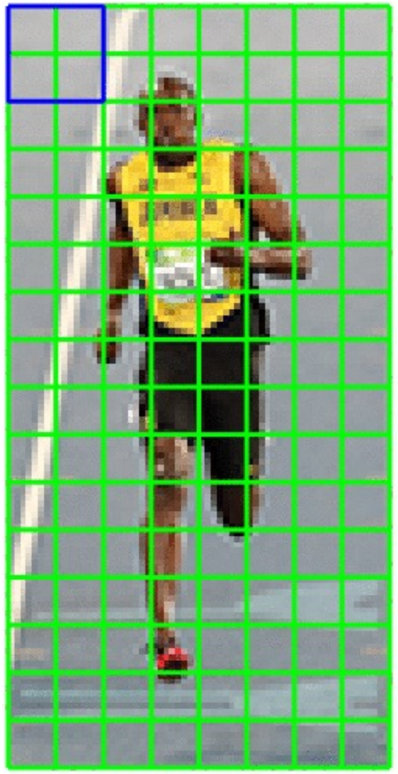
2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

Histogram of Gradients

Histogram of Oriented Gradients (HOG)

Block Normalization



For each of the cells in the current block

- Concatenate their corresponding gradient histograms
- Perform L1 or L2 normalization by dividing each element of the histogram by L1 or L2 norm.

L1 Norm: $\|x\|_1 = \sum_{i=1}^{n} |x_i|$

L2 Norm: $\|x\|_1 = \sqrt{\sum_{i=1}^{n} x_i^2}$

Histogram of Oriented Gradients (HOG)

Calculate the HOG feature vector

After all blocks are normalized

- we take the resulting histograms
- concatenate them
- treat them as our final feature vector.

References & Study Resources

- Guozhu Dong and Huan Liu. (2020). *Feature Engineering for Machine Learning and Data Analytics*. CRC Press.
- https://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/
- https://www.learnopencv.com/shape-matching-using-hu-moments-c-python/