Histogram of Gradient (HOG)
Shape Context
Template Matching

Computer Vision (CS0029)

Outline

- **Histogram of Gradients**
- Shape Context
- Template Matching

Object Recognition

- Want a descriptor that can be employed to reliably recognize objects

- Observation: local object appearance and shape often well-characterized by distribution of local orientation
  - Do not need precise knowledge (spatial position) of corresponding gradient or edge positions

Histogram of Gradients (HOG)

- Davide image into small cells
- Accumulate histogram of orientations of pixels within cells (from gradients)
- Concatenate histograms for final object representation

- Can improve accuracy by normalizing histogram for getter invariance to illumination
- Parameters vary depending on application

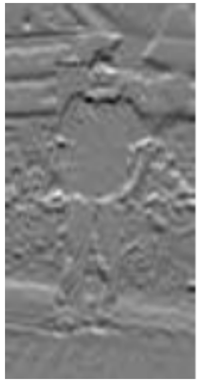
Gradient Computation

- Use simple 1-D mask [-1 0 1]
- Smoothing not used
- Compute gradients for each color channel separately
  - Use the color with largest magnitude as pixel's gradient

Input image

X gradient

Y gradient

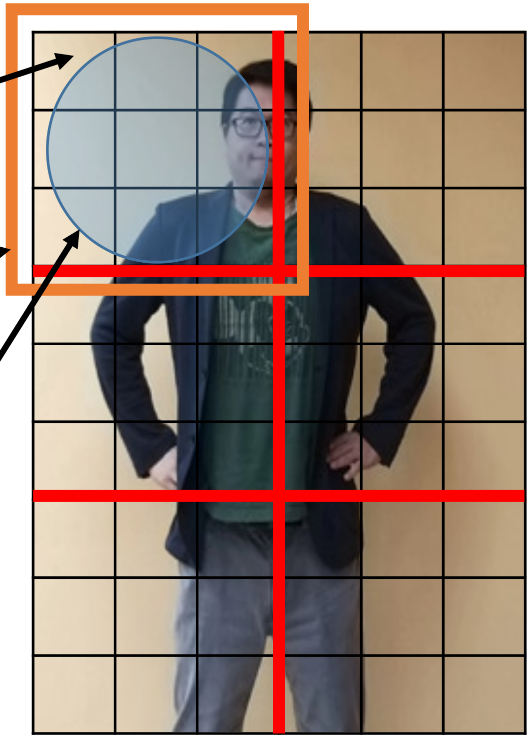


Orientation Binning

- Each pixel contributes weighted vote for “edge orientation” histogram (1-D)
  - Edge orientation is orthogonal to gradient direction rotations of gradient angle by 90 degrees
  - Weighted vote based on gradient magnitude
  - Orientation bins evenly spaced between 0 and 180 degrees
- Votes accumulated over local spatial cells
- Group cells into larger spatial blocks

Rectangular Blocks

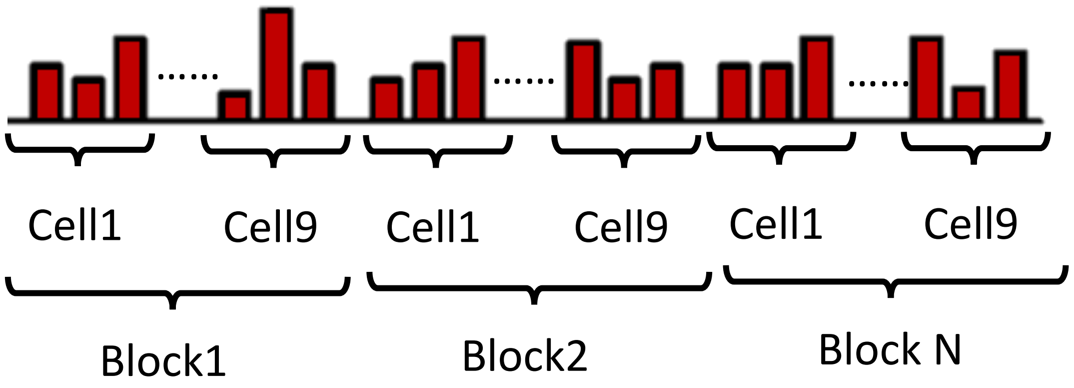
- Blocks generally square
- 3parameters
  - Number pixels per cell
  - Number cells per block
  - Number bins per gradient histogram
- Dalal:
  - Cell = 6x6 pixels
  - Block= 3x3 cells
  - 9 bin histogram (0-180 degrees)
- Reduce weight for pixels near border of block in cell histogram by applying Gaussian spatial windows over block



HOG Descriptor

- Local variations in illumination and foreground/background contrast cause gradient strengths to vary
  - Normalize each block descriptor separately

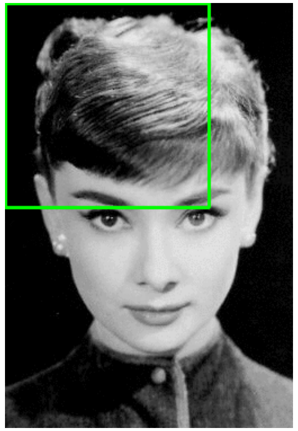
- HOG descriptor is concatenation of normalized block descriptors



- Blocks could overlap
  - Each cell could contribute to multiple blocks descriptors

Example of Object Detection using HOG

- Step1: to collect P positive images
- Step2: to collect N negative images (N>>P)
- Step3: calculate HOG from each positive and negative images
  - Feature vector of each images: its HOG
- Step4: label each feature vectors and use Adaboost, SVM, … to train a classifier
- Step5: Given a test image, calculate HOG of sub-windows
  - Each HOG is sent to the classifier to get the prediction



Histogram of Optical Flow (HOF)

- HOG focuses on static appearance information
- What about local motion information
- Histogram of Optical Flow (HOF)
  - Analogous to HOG
  - Weighted quantization of flow vectors
  - Some approaches have separate bin for no motion

Outline

- Histogram of Gradients
- **Shape Context**
- Template Matching

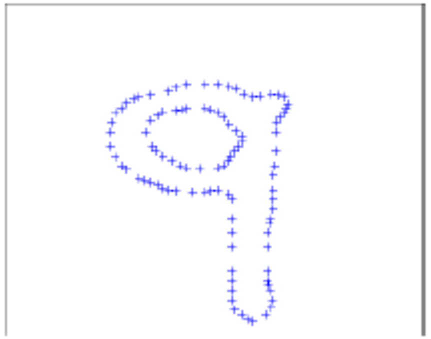
Which objects have similar shapes?

Shape Context

- Shape context is a descriptor to coarsely describe the distribution of points along shape boundary with respect to a given point
  - Inherently invariant to translation
  - Can make invariant to scale and rotation
  - Empirically shown to be invariant to small nonlinear transformations, occlusions and presence of outliers
- Corresponding points on similar shapes will have similar shape context
- Can be incorporated in shape similarity measurement
  - For object recognition

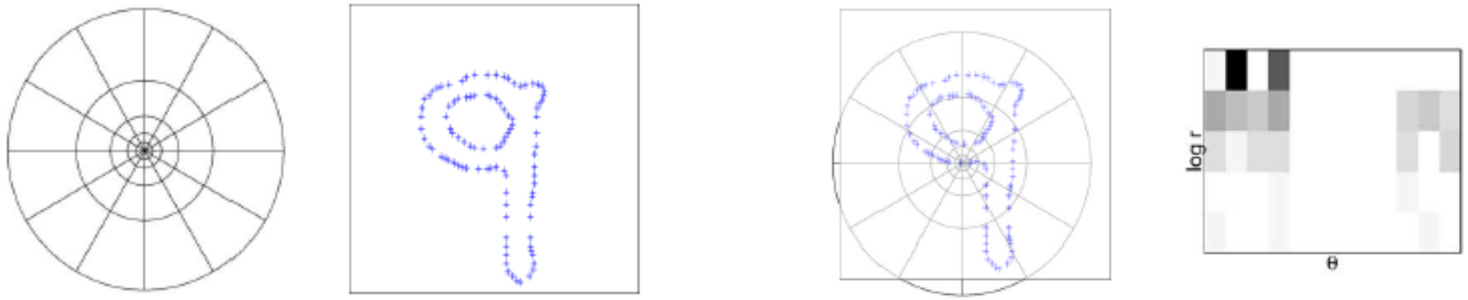
Feature points

- Represent shapes as discrete set of points on internal and external contours $P = \{p_1, p_2, \ldots, p_n\}$
  - Randomly sample points, ensuring a minimum distance between points
  - Do not have to be key points (corners)

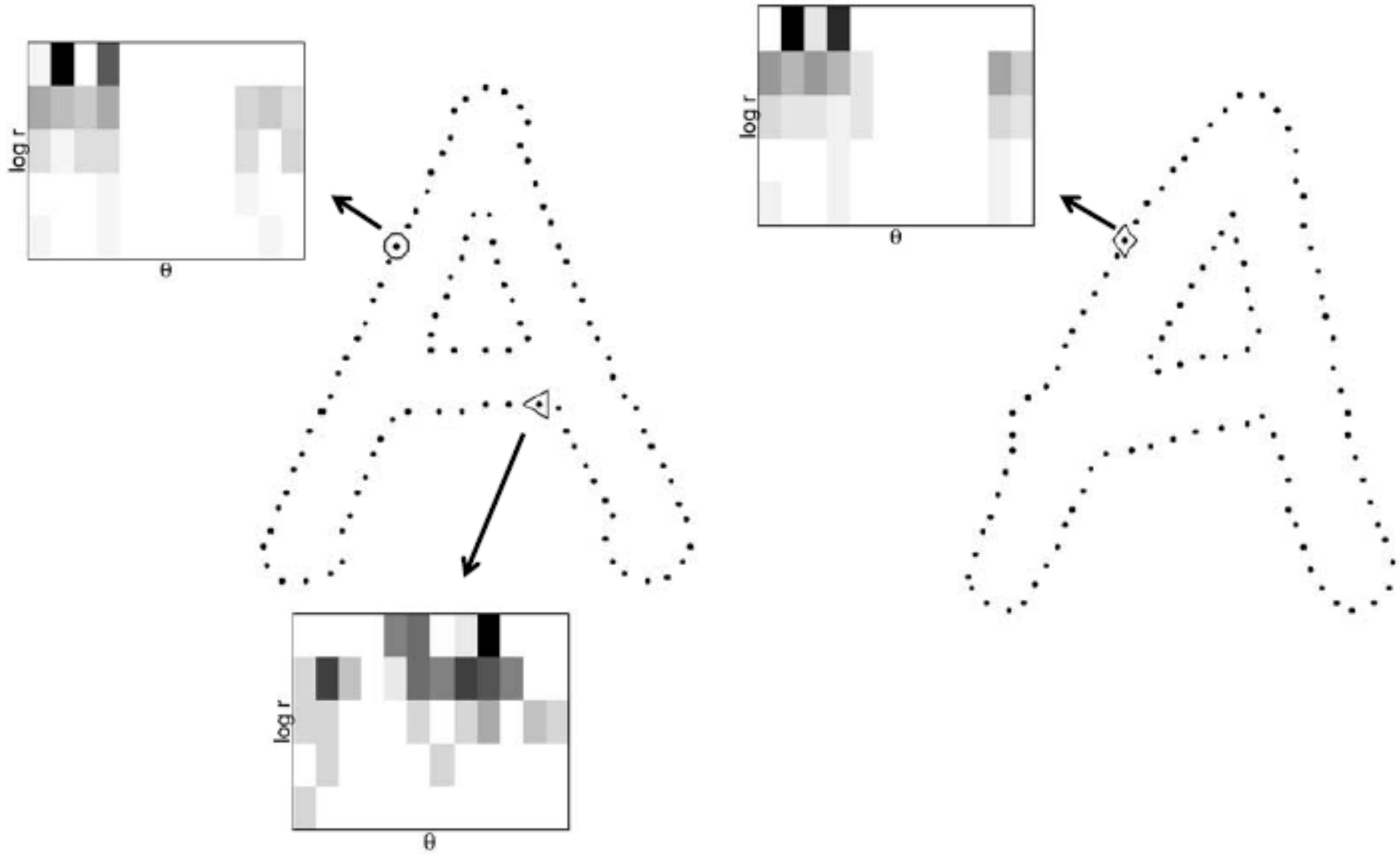


Shape Context

- For each point p_i compute coarse histogram h_i of the relative coordinates for the remaining $n-1$ points
    - $h_{i(k)} = \{q \neq p_i : (q - p_i) \in bin(k)\}$
- Distribution of relative positions is robust, compact, and highly discriminative
- Use bins in log-polar space
    - 12 angular bins, 5 radial bins



Shape Context



Invariance and Robustness

- Representation is translation invariant
- Scale invariance achieved by normalizing all radial distance by mean distance of the point pairs
- Rotational invariance can be achieved by aligning shape contexts to tangent/gradient vectors for each point
  - Not always desirable (e.g., differentiating 6 from 9)
- Empirically shown to be robust to small nonlinear transformations, occlusions, and presence of outliers when transformation model is incorporated

Point Correspondence

- Compute cost of matching point p_i on first shape to point q_j on second shape using χ^2 statistic between normalized histogram
  - $C(p_i, q_i) = \frac{1}{2}\sum_{k=1}^{n} \frac{\left(h_i(k) - h_j(k)\right)^2}{h_i(k) + h_j(k)}$
- Minimize total matching cost having one-to-one mapping constraint
  - $H(\pi) = \sum_i C\left(p_i, q_\pi(i)\right)$
  - Can add dummy nodes for outliers and when unequal number of points in shapes
  - Solve using Hungarian algorithm

Hungarian Algorithm (Allocation Problem)

	P1	**P2**	**P3**	**P4**
P1	80	40	50	46
P2	40	70	20	25
P3	30	10	20	30
P4	35	20	25	30

- Step1:
  - subtract each row's minimum value from each row
  - subtract each column's minimum value from each column
- Step2:
  - Cover the zero elements with minimum number of lines. If this minimum number is same as the size of the matrix, then go to step 4
- Step3:
  - Let *m* be the minimum uncovered element. The matrix is augmented by reducing all uncovered elements by *m* and increasing all elements covered by two lines by *m.* Return to step 2
- Step3:
  - There is a maximal match using only zeros.

Point Correspondence

Key Takeaways

- Shape context: match the similar shape objects
- Sample points on object edges. Check the samples points of two candidates are similar or not
  - Use shape context: 2D histogram defined in log-polar space
  - Calculate shape context on each points (an objects have multiple shape contexts)
- Match points between two candidates (rough idea: find the point with the most similar shape context)
  - Use Hungarian Algorithm to solve it as an allocation problem

Outline

- Histogram of Gradients
- Shape Context
- **Template Matching**

Introduction

- Want to find areas of a search image that are similar to given template image T

Template T

Search image

Best Matching Patch in Search Image

General Approaches

- Template-based:
  - Utilize raw template (pixels) and find best matching patches in search image
    - Sum-of-absolute differences (SAD)
    - Sum-of-squared differences (SSD)
    - Normalized cross-correlation (NCC)

Sum-of-Absolute Differences (SAD)

- Compute absolute differences of pixels intensities of template T and image patch P extracted from search image (note that P is same size as template T）
  - $SAD(P, T) = \sum_{R,G,B} \sum_{x,y} |P(x, y) - T(x, y)|$

- Compute SAD for all unique patch locations within the search image

- Keep patch with minimum SAD or patches with SAD less than given threshold

SAD Example

Template Image T

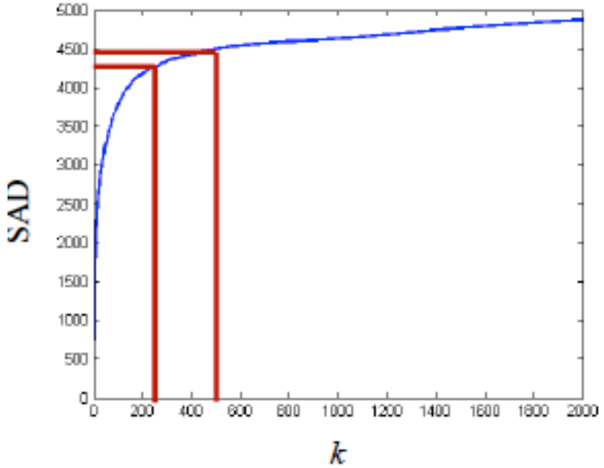
Search Image

Negative SAD
Origin is in center of patch

SAD Example

- k^{th} best matching patch

k=1 k=10 k=50 k=100 k=150 k=200 k=250 k=500 k=1000



Sum-of Squared Differences

- Similar to SAD, but replace absolute differences with squared difference
  - SSD(P,T)= $\sum_{R,G,B} \sum_{x,y} (P(x,y) - T(x,y))^2$

- Compute SSD for all unique patches within the search image

- Keep patch with minimum SDD

SSD Example

Template Image T

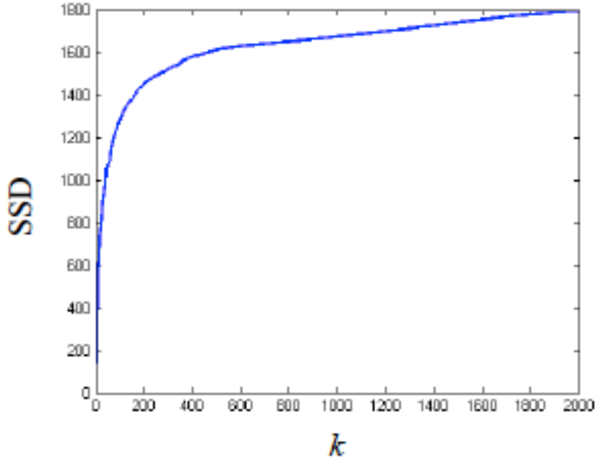
Search Image

Negative SSD
Origin is in center of patch

SSD Example

- k^{th} best matching patch

k=1 k=10 k=50 k=100 k=150 k=200 k=250 k=500 k=1000



Illumination Changes

- SAD and SSD can work well if the template and search images have the same brightness
  - Problem: images can have varying illumination condition

k^{th} best matching patch

k=1 k=10 k=50 k=100

k=150 k=200 k=250 k=500 k=1000

Template Image T

Search Image

Normalized Cross-Correlation (NCC)

- Normalize image to remove variations from illumination condition

Mean of pixel values in patch
(each color computed independently)

$$NCC(P,T) = \sum_{R,G,B} \frac{1}{n-1} \sum_{x,y} \frac{(P(x,y)-\bar{P})(T(x,y)-\bar{T})}{\sigma_p \sigma_T}$$

Standard deviation of pixel values in patch
(each color computed independently)

- Larger values of NCC is better!

NCC Example

Template Image T

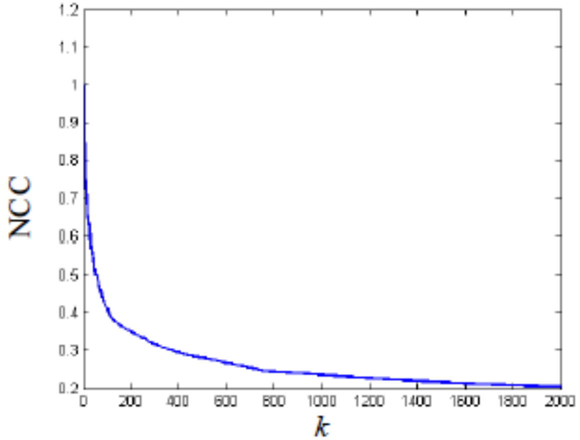
Search Image

NCC
Origin is in center of patch

NCC Example

k^{th} best matching patch

k=1 k=10 k=50 k=100 k=150 k=200 k=250 k=500 k=1000



Key Takeaways

- SAD, SSD and NCC to calculate the similarity/difference between two image patches
- SAD: not robust
- SSD: Not good at illumination change
- NCC: the most robust among these three metric