Programming Assignment #4

• Due two weeks from today
  – Any questions?
  – How is it going?
Where are we?

• We have two umbrella schemes for object recognition
  – Bag of Features, Constellations
• We bootstrap these with feature detections
  – Interest points, regions, etc.
• To implement these, we looked at
  – Clustering (K-Means, EM)
  – Classification (SVM, Backprop, Bayes nets, Decision trees, Nearest neighbors)
• These algorithms view samples as points in high-dimensional feature spaces.

Where do the features come from?
High-dimensional Feature Descriptors

• Goal: describe the properties of image features
  – “similar” features should be near each other in feature descriptor space
  – “dissimilar” features should not be
  – Insensitive to changes in
    • Viewpoint
    • Scale
    • Illumination
Terminology Confusion

• Feature : a distinctive local property of an image
  – Interest point
  – Region
  – Line, curve, etc.

• Descriptor : a high dimensional vector describing a feature
  – Vectorized image patch
  – SIFT descriptor, LBP, Haar, …

When we say feature space, we typically mean feature descriptor space
SIFT Interest Point Descriptor

- SIFT Interest Points are extrema of the DoG responses to an image pyramid
- SIFT descriptors are 128-dimensional vectors describing the image patch around a SIFT interest Point
  - SURF descriptors are very similar (but avoid the patent issues)
- In OpenCV, you can compute the SURF descriptor for any (x,y,s) image point
  - Even if its not a SIFT/SURF interest point
Review: Interest Points

- Properties
  - Location \((x, y)\)
  - Scale
    - Measured in octaves
    - SIFT: \(1/3\) octaves

http://computervisionblog.wordpress.com/tag/sift-feature-point/
SIFT Descriptor Step 1: Scale

- If scale $\neq 1$, image is down-scaled around the interest point
  - Every octave is a power of 2
  - Non-integer octaves require bilinear image interpolation (see beginning of course)
- SIFT descriptors are based on the 16x16 scaled image patch around the interest point
Step 2: Rotation

• Goal: compensate for in-plane rotation
• Calculate the intensity derivatives in \((x,y)\) of the 16x16 scaled image patch
  – Convolution with Sobel masks
  – Produces \((\partial I/\partial x, \partial I/\partial y)\) for every pixel
• Produce the structure tensor:

\[
\begin{pmatrix}
\left(\frac{\partial I}{\partial x}\right)^2 & \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\
\frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \left(\frac{\partial I}{\partial y}\right)^2
\end{pmatrix}
\]
Step 2: Rotation (cont.)

- The first eigenvector of the structure tensor is the dominant edge direction.
- Rotate the scaled image patch so that the x axis is aligned with the dominant edge direction.
Step 3: Localized Edge Orientation Histograms

- The 1\textsuperscript{st} 2 steps produce a scaled and rotated 16x16 image patch.
- Divide this patch with a 4x4 grid. Each cell contains 4x4 pixels.
Step 3: Localized Edge Orientation Histograms (cont.)

- For each grid cell, histogram the rotated (dx, dy) edges
  - Histogram buckets are edge orientations
    - 8 orientation buckets (45° each)
  - Weights voted by edge magnitude
    - $\sqrt{(dx)^2+(dy)^2}$
    - Smoothed by a $1/2\sigma$ Gaussian

- Feature vector is the concatenation of 16 8-bucket histograms (128 dimensions)
SIFT descriptors: why?

• Two points are similar if:
  – They have similar nearby edges (orientation & strength)
  – In similar positions, relative to the points

• Descriptors are insensitive to:
  – Scale (points are rescaled)
  – In-plane rotation (points are rotated to dominant edge direction)
  – Average illumination (based on edges; assuming no clipping or floor effects)
SIFT descriptors: why? (cont.)

- Sensitivity is minimized with regard to:
  - Small translations (± 1 pixel)
    - Histograms insensitive to movements within 4x4 grid cell
    - Edge weight smoothing minimizes boundary effects
  - Small affine distortions
    - Small viewpoint changes can be roughly approximated by small changes in in-plane rotation and translation
Feature Vector Length Intuition

• 128 dimensions is in feature vector “sweet spot”
  – Too few dimensions \(\Rightarrow\) not enough ways for samples to differ
  – Too many dimensions \(\Rightarrow\) distributions become uniform
  – Many of the best feature descriptors are in the range \([50, 500]\) in length
HoG: Histogram of Gradients

- HoG is a variation on SIFT descriptors
  - Operates on image patches (not necessarily around interest points)
  - No compensation for scale or rotation
  - Computes magnitude-weighted edge orientation histograms (like SIFT)
  - Allows for different number of cells, cell shapes, and orientation bin counts
- Biggest difference: descriptor blocks
HoG Descriptor Blocks

- HoG descriptors are applied to larger image patches, which may have internal changes in illumination
- HoG descriptors use more cells (to cover large patches)
- Cells grouped in descriptor blocks
  - Descriptor blocks overlap; cells are in more than one block
Descriptor block normalization

• A normalization constant is calculated for every descriptor block
• Based on edge magnitudes within the block
  – Most often based on sum of Euclidean lengths
  – Other normalization function get played with
• Edge magnitudes are normalizes prior to histogram voting
Texture: Localized Binary Patterns (LBP)

• A feature vector to describe the texture within an image (patch)
• Like SIFT & HoG, begin by dividing the image patch into localized cells
• Compute a histogram for each cell
• Concatenate the histograms of the cells into a longer vector
LBP (theory)

- The difference is that in LBP, a texture measure is histogrammed (not edges)
- For every pixel, do the following:
  - Evenly sample 8 points on a circle of radius r, centered at the pixel
    - Interpolate pixel values (bilinearly) as needed
  - For each sample, return ‘1’ if sample is brighter than center pixel, ‘0’ otherwise
  - Interpret string of 8 bits as a binary integer
    - 0 to 255
- Create a histogram of the 256 texture values
The value of the LBP code of a pixel \((x_c, y_c)\) is given by:

\[
LBPP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p
\]

\[
s(x) = \begin{cases} 
1, & \text{if } x \geq 0; \\
0, & \text{otherwise.}
\end{cases}
\]

1. Sample
2. Difference
3. Threshold
4. Multiply by powers of two and sum

1 \* 1 + 1 \* 2 + 1 \* 4 + 1 \* 8 + 0 \* 16 + 0 \* 32 + 0 \* 64 + 0 \* 128 = 15
LBP (practice)

• Problem: 256 bins is “too many”
  – Concatenation creates vectors outside of “sweet spot” (length > 500)
  – Most values in histogram are zero
• Solution: transitions in 8-bit vector are rare
  – Count how many 1’s are followed by 0’s (and vice-versa)
  – 90+% of pixels in practice have fewer than 2 transitions
    • So just histogram these!
• Pattern transition counts
  – Two patterns have 0 transitions (all 1’s, all 0’s)
  – 16 patterns have 1 transition
  – 32 patterns have 2 transitions
• 48-dimension LBP
  – Histogram all 1 & 2 transition patterns
• 51-dimension LBP
  – Add 0 transition patterns, and one bin for “other”
LBP Applied

Iconic Representation

- A lower-dimensional representation of a point (or small image patch)
- Result of multi-scale convolution with low-order wavelets
- “Steerable” – response at orientation $\theta_i$ predicts the response at $\theta_j$.
- Responses are independent of each other
Iconic Representation Masks

- Each row is a scale
- 10 masks per scale
- Masks are “steerable”