CS534: Introduction to Computer Vision
Local Image Features

Ahmed Elgammal
Dept. of Computer Science
Rutgers University

Outlines

- Descriptors for local features and image patches
- Histogram of Gradient Orientations
- Detecting corners in images.
- LoG interest point detectors
- SIFT Features

- Source: Ch 5, Forsyth and Ponce "Computer Vision, a modern approach" – 2nd ed

Gradient-based edge detection (recall):

- Compute image derivatives (with smoothing) by convolution

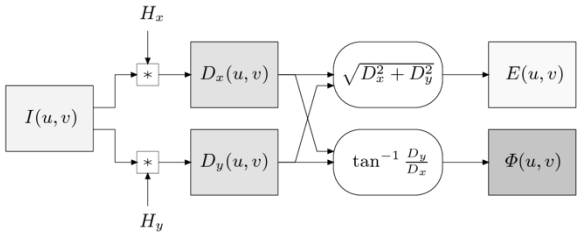
$$D_x(u,v) = H_x * I \quad \text{and} \quad D_y(u,v) = H_y * I$$

- Compute edge strength - gradient magnitude

$$E(u,v) = \sqrt{\big(D_x(u,v)\big)^2 + \big(D_y(u,v)\big)^2}$$

- Compute edge orientation - gradient direction

$$\Phi(u,v) = \tan^{-1}\Big(\frac{D_y(u,v)}{D_x(u,v)}\Big) = \mathrm{ArcTan}\big(D_x(u,v), D_y(u,v)\big)$$



Edge Maps Depend on Shading

- If the image is brighter (resp. darker)
  - because the camera gain is higher (resp. lower)
  - because there is more (resp. less) light
  - because the pixel values got multiplied by a constant
- Then the gradient magnitude is bigger (resp. smaller)
- So scaling image brightness changes the edge map
  - because some magnitudes will go above (resp. below) the test threshold
- Edge maps differ for brighter/darker copies of a picture

Orientations

- Gradient magnitude is affected by illumination changes, but gradient direction isn't

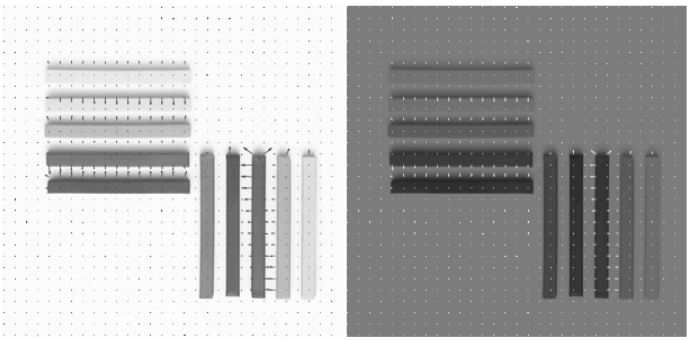


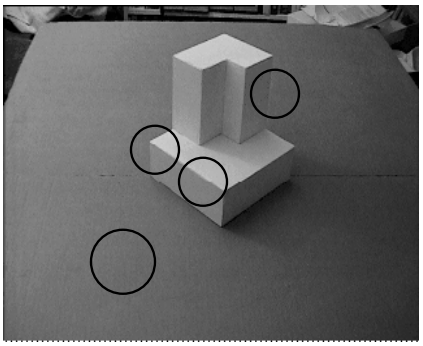
FIGURE 5.7: The magnitude of the image gradient changes when one increases or decreases the intensity. The orientation of the image gradient does not change; we have plotted every 10th orientation arrow, to make the figure easier to read. Note how the directions of the gradient arrows are fixed, whereas the size changes. *Philip Gatward © Dorling Kindersley, used with permission.*

Building Orientation Representations

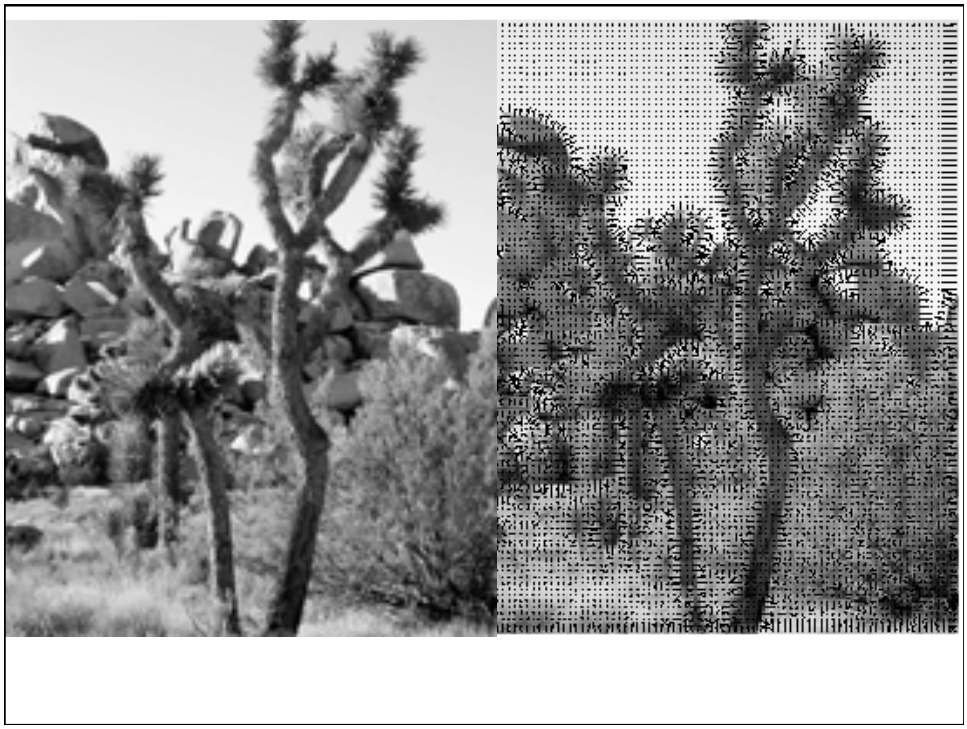
- We would like to represent a pattern in an image patch
  - to detect things in images
  - to match points in one image to corresponding points in another image
- Necessary properties
  - we have to know which patch to describe
  - think of this as knowing the center and size of an image window
- Desirable features
  - representation doesn't change much if the center is slightly wrong
  - representation doesn't change much if the size is slightly wrong
  - representation is distinctive
  - representation doesn't change much if the patch gets brighter/ darker
  - large gradients are more important than small gradients

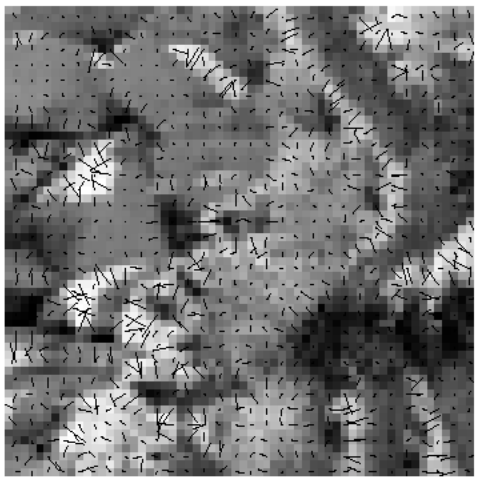
Corners and Interest Points

- We are looking for image locations that can be accurately localized
- An interest point is a point which has a well-defined position and can be robustly detected
- Aperture problem
- Critical to many problems:
  - Tracking
  - Image registration
  - Mosaicing and stitching
  - Object Recognition
  - …



- Corners are features that can be accurately localized
- What is a corner:
  - Intersection of two edges
- Edge detectors often fail at corners
- A corner is characterized by:
  - Large gradient
  - two dominant and different edge directions in a local neighborhood of the corner





- Characterizing gradient in an image window: local structure matrix – also called structure tensor

$$H = \sum_{window} (\nabla I)(\nabla I)^T$$

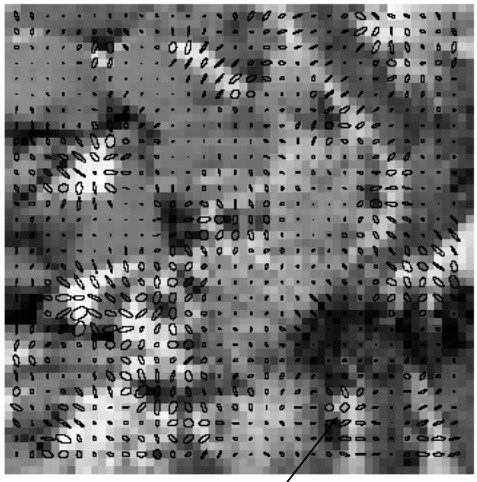
$$H = \sum \begin{bmatrix} (\frac{\partial G_\sigma}{\partial x} * I)(\frac{\partial G_\sigma}{\partial x} * I) & (\frac{\partial G_\sigma}{\partial x} * I)(\frac{\partial G_\sigma}{\partial y} * I) \\ (\frac{\partial G_\sigma}{\partial x} * I)(\frac{\partial G_\sigma}{\partial y} * I) & (\frac{\partial G_\sigma}{\partial y} * I)(\frac{\partial G_\sigma}{\partial y} * I) \end{bmatrix}$$

Representing Windows

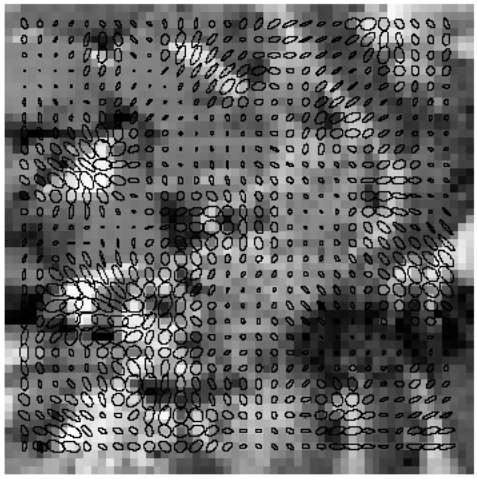
- The eigenvalues of H captures the local structure
- Types
  - constant
    - small eigenvalues
  - Edge
    - one medium, one small
  - Flow
    - one large, one small
  - corner
    - two large eigenvalues

$$H = \sum_{window} (\nabla I)(\nabla I)^T$$

$$H = \sum \begin{bmatrix} (\frac{\partial G_\sigma}{\partial x} * I)(\frac{\partial G_\sigma}{\partial x} * I) & (\frac{\partial G_\sigma}{\partial x} * I)(\frac{\partial G_\sigma}{\partial y} * I) \\ (\frac{\partial G_\sigma}{\partial x} * I)(\frac{\partial G_\sigma}{\partial y} * I) & (\frac{\partial G_\sigma}{\partial y} * I)(\frac{\partial G_\sigma}{\partial y} * I) \end{bmatrix}$$



Plotting : $(x,y)^T H^{-1} (x,y)=\varepsilon$
3x3 windows



5x5 windows

Harris Corners

$$H = \begin{pmatrix} a & c \\ c & b \end{pmatrix}$$

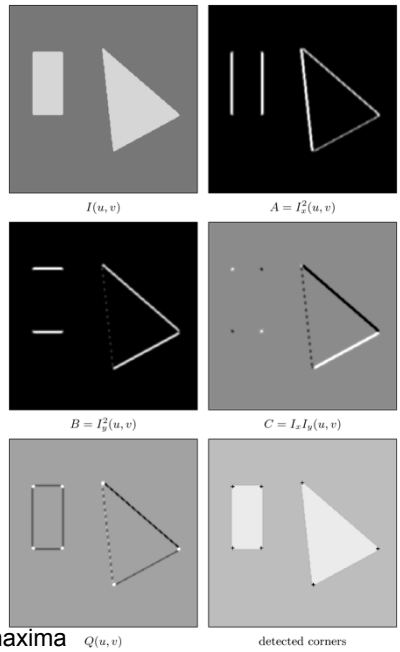
$$\lambda_{1,2} = \tfrac{1}{2}(a + b \pm \sqrt{a^2 - 2ab + b^2 + 4c^2})$$

$$\lambda_{1,2} = \frac{trace(H)}{2} \pm \sqrt{\left(\frac{trace(H)}{2}\right)^2 - \det(H)}$$

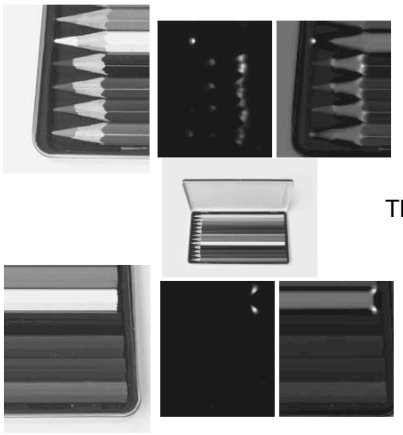
- Corner response function: find local maxima of

$$Q(u,v) = \det(H) - k\left(\frac{trace(H)}{2}\right)^2$$

  - k: steering parameter <1



We need to find a local maxima (non maxima suppression)



The detector is affected by contrast

FIGURE 5.10: The response of the Harris corner detector visualized for two detail regions of an image of a box of colored pencils (**center**). **Top left**, a detail from the pencil points; **top center**, the response of the Harris corner detector, where more positive values are lighter. The **top right** shows these overlaid on the original image. To overlay this map, we added the images, so that areas where the overlap is notably dark come from places where the Harris statistic is negative (which means that one eigenvalue of $\mathcal{H}$ is large, the other small). Note that the detector is affected by contrast, so that, for example, the point of the mid-gray pencil at the top of this figure generates a very strong corner response, but the points of the darker pencils do not, because they have little contrast with the tray. For the darker pencils, the strong, contrasty corners occur where the lead of the pencil meets the wood. The **bottom** sequence shows corners for a detail of pencil ends. Notice that responses are quite local, and there are a relatively small number of very strong corners. *Steve Gorton © Dorling Kindersley, used with permission.*

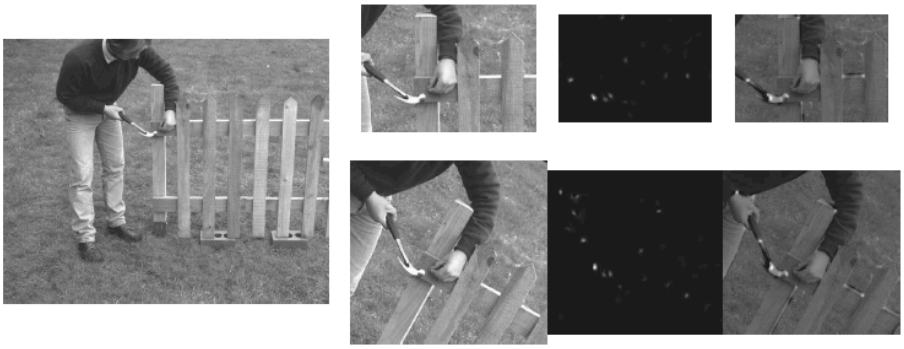
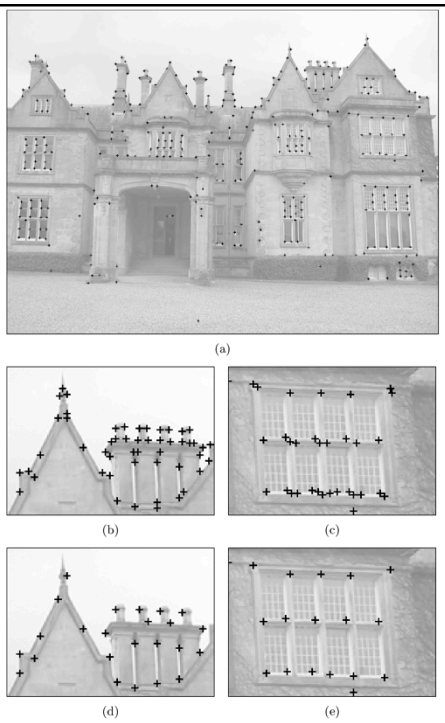
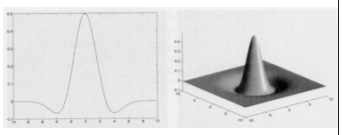


FIGURE 5.11: The response of the Harris corner detector is unaffected by rotation and translation. The **top row** shows the response of the detector on a detail of the image on the **far left**. The **bottom row** shows the response of the detector on a corresponding detail from a rotated version of the image. For each row, we show the detail window (**left**); the response of the Harris corner detector, where more positive values are lighter (**center**); and the responses overlaid on the image (**right**). Notice that responses are quite local, and there are a relatively small number of very strong corners. To overlay this map, we added the images, so that areas where the overlap is notably dark come from places where the Harris statistic is negative (which means that one eigenvalue of $\mathcal{H}$ is large, the other small). The arm and hammer in the top row match those in the bottom row; notice how well the maps of Harris corner detector responses match, too. © *Dorling Kindersley, used with permission.*



- From corners to neighborhoods
- We need to estimate the scale of the neighborhood around the corner and it's orientation
- Scale? Scale should get larger proportionally when the image is larger…

Estimating scale



- Assume we have detected a corner
- How big is the neighborhood?
- Use Laplacian of Gaussian filter
  - Kernel looks like fuzzy dark blob on pale light foreground
  - Scale (sigma) of Gaussian gives size of dark, light blob
- Strategy
  - Apply Laplacian of Gaussian at different scales at corner
    - response is a function of scale
  - Choose the scale that gives the largest response
    - the scale at which the neighborhood looks "most like" a fuzzy blob

Estimating scale

- Laplacian of a function $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$
- Gaussian $G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{\left(\frac{-x^2-y^2}{2\sigma^2}\right)}$
- So Laplacian of Gaussian

$$\nabla^2 G_\sigma(x,y) = \left(\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right) G_\sigma(x,y)$$

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

- Convolve with image

$$\nabla^2_\sigma \mathcal{I}(x,y) = \left(\nabla^2 G_\sigma(x,y)\right) ** \mathcal{I}(x,y)$$

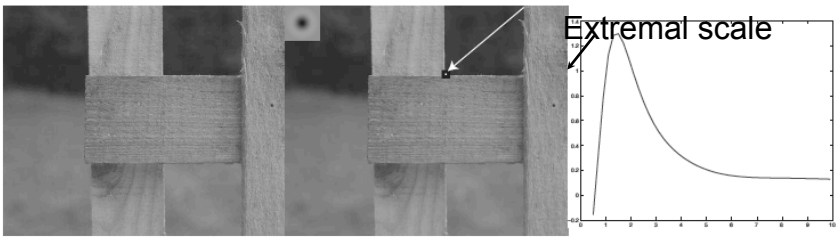


FIGURE 5.12: The scale of a neighborhood around a corner can be estimated by finding a local extremum, in *scale* of the response at that point to a smoothed Laplacian of Gaussian kernel. On the **left**, a detail of a piece of fencing. In the center, a corner identified by an arrow (which points to the corner, given by a white spot surrounded by a black ring). *Overlaid* on this image is a Laplacian of Gaussian kernel, in the **top right** corner; dark values are negative, mid gray is zero, and light values are positive. Notice that, using the reasoning of Section 4.5, this filter will give a strong positive response for a dark blob on a light background, and a strong negative response for a light blob on a dark background, so by searching for the strongest response at this point as a function of scale, we are looking for the size of the best-fitting blob. On the **right**, the response of a Laplacian of Gaussian *at the location of the corner*, as a function of the smoothing parameter (which is plotted in pixels). There is one extremal scale, at approximately 2 pixels. This means that there is one scale at which the image neighborhood looks most like a blob (some corners have more than one scale). © *Dorling Kindersley, used with permission.*

Estimating orientation of neighborhood

- We need to achieve rotation-invariant descriptor of image patches
- A natural reference orientation is the most common orientation of the patch
- Within neighborhood, estimate image orientations
- Form a histogram of orientations
  - Weighting by distance to center, or unweighted
- Choose the orientation with the maximum count
  - if there are two or more
    - make copies of the neighborhood
    - each has one of the peak count orientations

Assume a fixed scale parameter k
Apply a corner detector to the image $\mathcal{I}$
Initialize a list of patches
For each corner detected
  Write (x_c, y_c) for the location of the corner
  Compute the radius r for the patch at (x_c, y_c) as
    $r(x_c, y_c) = \underset{\sigma}{\text{argmax}} \ \nabla^2_\sigma \mathcal{I}(x_c, y_c)$
    by computing $\nabla^2_\sigma \mathcal{I}(x_c, y_c)$ for a variety of values of σ, interpolating these values, and maximizing
  Compute an orientation histogram $H(\theta)$ for gradient orientations within a radius kr of (x_c, y_c).
  Compute the orientation of the patch θ_p as
    $\theta_p = \underset{\theta}{\text{argmax}} \ H(\theta)$. If there is more than one theta that maximizes this histogram, make one copy of the patch for each.
  Attach (x_c, y_c, r, θ_p) to the list of patches for each copy

Algorithm 5.2: Obtaining Location, Radius and Orientation of Pattern Elements Using a Corner Detector.

Finding image neighborhoods (Without Corner detection)

- Fuzzy blob strategy
  - Find locations and scales where
    - image looks more like fuzzy blobs than their immediate neighbors
      - in location
      - in scale
  - Then estimate orientation
- Find local maxima of

$$\nabla^2_\sigma \mathcal{I}(x, y) = \left(\nabla^2 G_\sigma(x, y)\right) * * \mathcal{I}(x, y)$$

  - as a function of location and scale

Assume a fixed scale parameter k
Find all locations and scales which are local extrema of
  $\nabla^2_\sigma \mathcal{I}(x, y)$ in location (x, y) and scale σ forming a list of triples (x_c, y_c, r)
For each such triple
  Compute an orientation histogram $H(\theta)$ for gradient orientations within
    a radius kr of (x_c, y_c).
  Compute the orientation of the patch θ_p as
    $\theta_p = \underset{\theta}{\text{argmax}}\ H(\theta)$. If there is more than one θ that
    maximizes this histogram, make one copy of the patch for each.
  Attach (x_c, y_c, r, θ_p) to the list of patches for each copy

Algorithm 5.3: Obtaining Location, Radius, and Orientation of Pattern Elements Using the Laplacian of Gaussian.

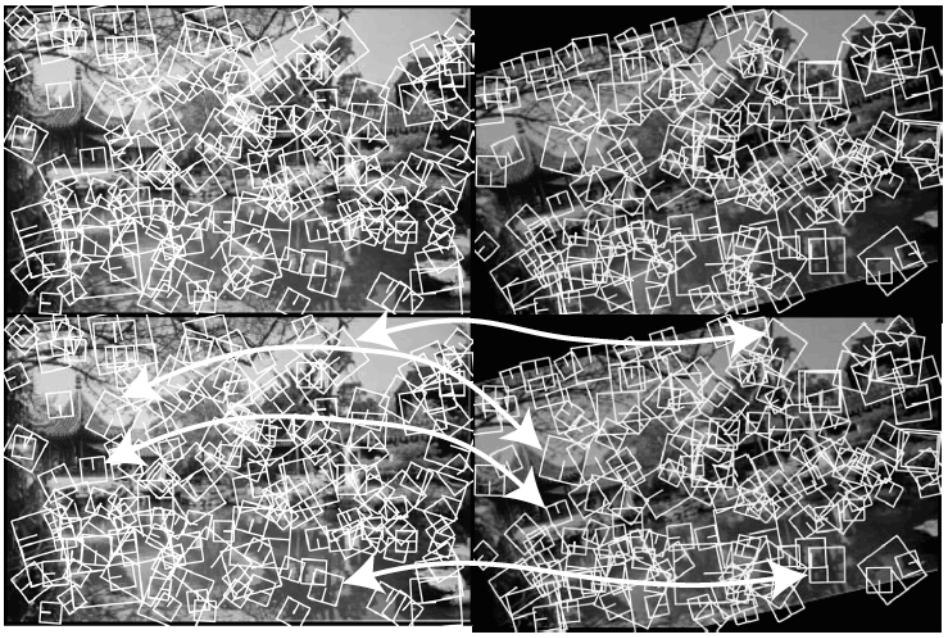


FIGURE 5.13: This figure shows local patches recovered using a method similar to that described in the text (the details of the corner detector were different). These patches

- Corners vs LoG for interest points:
  - Corner detectors respond to corner structure
  - LoG looks for circular blobs at a particular scale
  - Corner detector estimate the center very accurately, but the scale estimate is poor
  - Corners are most useful for matching with no large scale change
  - LoG is less accurate in estimating the center, but the scale estimate is better

Histograms of Oriented Gradients

For the moment, assume window is known

- Necessary properties
  - we have to know which patch to describe
  - think of this as knowing the center and size of an image window

- Desirable features
  - representation doesn't change much if the center is slightly wrong
  - representation doesn't change much if the size is slightly wrong
  - representation is distinctive
  - representation doesn't change much if the patch gets brighter/darker
  - large gradients are more important than small gradients

Use histograms

Use orientations

Break window into boxes, describe each separately

Weight orientation histogram entries

Histograms of Oriented Gradients

- Strategy:
  - break patch up into blocks
  - construct histogram representing gradient orientations in that block
    - which won't change much if the patch moves slightly
    - entries weighted by magnitude
- Variants
  - histogram of angles
  - histogram of gradient vectors, length normalized by block averages

Describing Neighborhoods

SIFT features

- SIFT=Scale Invariant Feature Transform
- Very strong record of effectiveness in matching applications
- Local histogram of gradient
  - use orientations to suppress intensity change effects
  - use histograms so neighborhood need not be exactly localized
  - weight large gradients higher than small gradients
- Weighting processes are different
- SIFT features behave very well using nearest neighbors matching
  - i.e. the nearest neighbor to a query patch is usually a matching patch

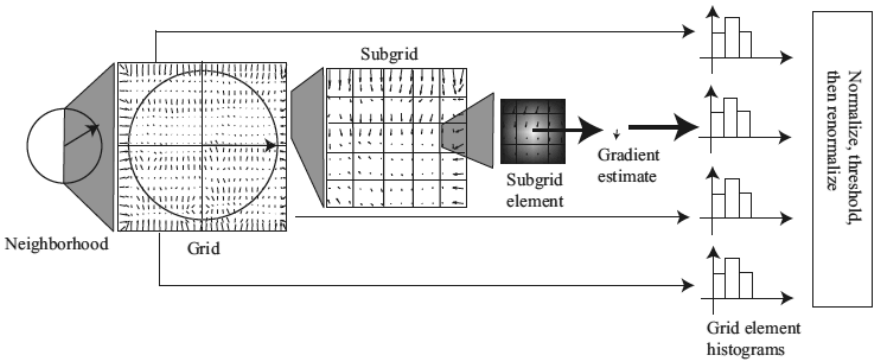


FIGURE 5.14: To construct a SIFT descriptor for a neighborhood, we place a grid over the rectified neighborhood. Each grid is divided into a subgrid, and a gradient estimate is computed at the center of each subgrid element. This gradient estimate is a weighted average of nearby gradients, with weights chosen so that gradients outside the subgrid cell contribute. The gradient estimates in each subgrid element are accumulated into an orientation histogram. Each gradient votes for its orientation, with a vote weighted by its magnitude and by its distance to the center of the neighborhood. The resulting orientation histograms are stacked to give a single feature vector. This is normalized to have unit norm; then terms in the normalized feature vector are thresholded, and the vector is normalized again.

Given a grid cell $\mathcal{G}$ for patch with center $\boldsymbol{c} = (x_c, y_c)$ and radius r

Create an orientation histogram
For each point $\boldsymbol{p}$ in an $m \times m$ subgrid spanning $\mathcal{G}$
  Compute a gradient estimate $\nabla\mathcal{I}\,|_{\boldsymbol{p}}$ estimate at $\boldsymbol{p}$
    as a weighted average of $\nabla\mathcal{I}$, using bilinear weights centered at $\boldsymbol{p}$.
  Add a vote with weight $\|\nabla\mathcal{I}\| \frac{1}{r\sqrt{2\pi}} \exp\left(-\frac{\|\boldsymbol{p}-\boldsymbol{c}\|^2}{r^2}\right)$
    to the orientation histogram cell for the orientation of $\nabla\mathcal{I}$.

Algorithm 5.5: Computing a Weighted q Element Histogram for a SIFT Feature.

HOG features

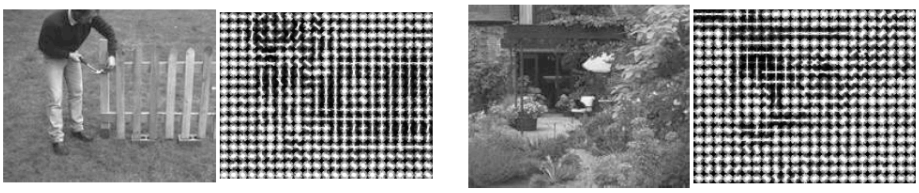


FIGURE 5.15: The HOG features for each the two images shown here have been visualized by a version of the rose diagram of Figures 5.7–5.9. Here each of the cells in which the histogram is taken is plotted with a little rose in it; the direction plotted is at right angles to the gradient, so you should visualize the overlaid line segments as edge directions. Notice that in the textured regions the edge directions are fairly uniformly distributed, but strong contours (the gardener, the fence on the **left**; the vertical edges of the french windows on the **right**) are very clear. This figure was plotted using the toolbox of Dollár and Rabaud. *Left: © Dorling Kindersley, used with permission. Right: Geoff Brightling © Dorling Kindersley, used with permission.*

HOG - Crucial Points

- Gradient orientations are not affected by intensity
- Orientations with larger magnitude are more important
- Describe an image window of known location, size
  - Histograms reduce the effect of poor estimate of location, size
  - Break window into subwindows
    - for each, compute an orientation histogram, weighting orientations by magnitude
  - Numerous variants available