Professor Brendan Morris, SEB 3216, brendan.morris@unlv.edu

ECG782: Multidimensional Digital Signal Processing

Spring 2014
TTh 14:30-15:45 CBC C313

Lecture 08
Image Pre-Processing II
13/02/13

http://www.ee.unlv.edu/~b1morris/ecg782/

Outline

- Smoothing
- Edges
  - Canny Edge Detector
- Frequency Domain Processing
- Interest Point Detection
- Maximally Stable Regions

Image Pre-Processing

- Low level operations
  - Lowest-level of abstraction
  - Image-to-image transformations
- Does not increase image information content
  - Actually decreases entropy
  - However, it can suppress irrelevant info
    - Not needed for analysis task
- Improve image by suppressing unwanted distortions and enhancing important image features
  - Note: geometric transforms also considered
  - Utilizes information redundancy
    - Large number of similar pixels for statistical characterization

Pixel Brightness Correction

- Modify pixel brightness with regard to position
- Systematic imaging degradation can be suppressed
  - E.g. CCD sensitivity on borders
- Multiplicative error model
  - $f(i,j) = e(i,j)g(i,j)$
  - f – degraded image
  - g – reference (“good”) image
  - e – multiplicative noise, error coefficient
- Recovery of good image
  - $g(i,j) = \frac{f(i,j)}{e(i,j)}$
  - Estimate error by imaging a known constant value

Gray-Scale Transformation

- Change pixel brightness without regard for position in image
  - E.g. histogram equalization
- Define a mapping between one gray level to another
  - Represented as a lookup table
  - Generalizes to multi-spectral images
    - Color conversion ta
- Typically used for hum observation
  - Contrast is needed

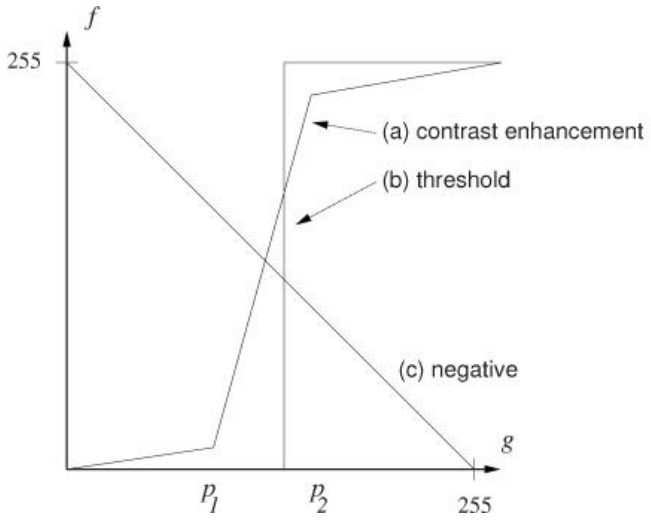
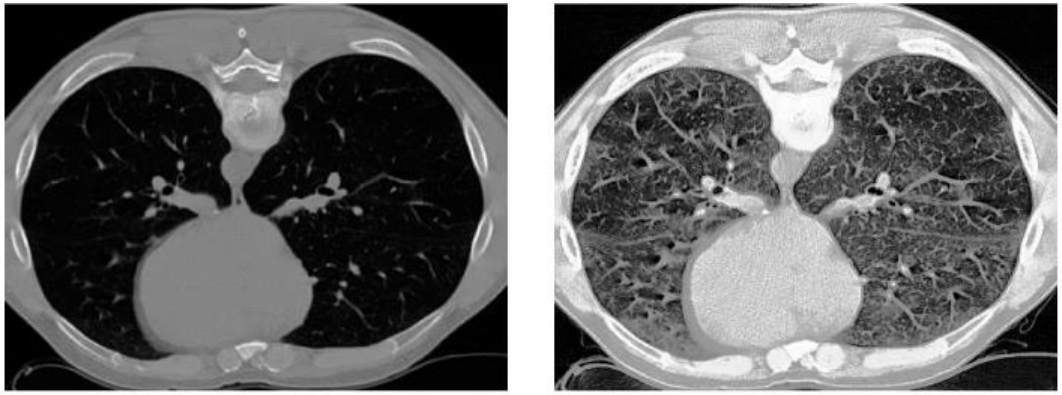


Figure 5.1: Some gray-scale transformations. © *Cengage Learning 2015.*



(a) (b)

Figure 5.3: Histogram equalization. (a) Original image. (b) Equalized image. © *Cengage Learning 2015.*

Local Pre-Processing

- Smoothing
  - Suppress noise and other small fluctuations
  - Equivalent to suppression of high frequency content
  - Blurs sharp edges
    - May lose information content
- (Sharpening) Gradient operators
  - Based on local derivatives of image
  - Suppress low frequency content
    - Accentuate edges
  - Increases noise level
- Linear transformations
  - Output value is a linear combination of local neighborhood values
  - $f(i,j) = \sum\sum_{(m,n)\in O} h(i-m, j-n)g(m,n)$
  - Discrete convolution(correlation) definition
    - Use rectangular neighborhoods with odd dimensions
- Non-linear transforms
  - Non-linear relationship between neighborhood
  - More computationally expensive
  - No strict frequency representation

Smoothing

- Want to edge-preserving smoothing
  - Remove noise but leave edges
- Averaging filter
  - Noise should be smaller in size than smallest object of interest
  - Significant edge blurring
  - $h = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$
- Gaussian approximation
  - Put more weight in center
  - $h = \frac{1}{10}\begin{bmatrix}1 & 1 & 1\\ 1 & 2 & 1\\ 1 & 1 & 1\end{bmatrix}$
- Separable filters
  - Used to significantly speed up convolution neighborhood operation
  - Kernel can be factorized into the product of two 1D vectors
    - Separate convolution summations
- 2D binomial kernel (Gaussian approximation)
- $h(x,y) = 4^{-(N-1)}\binom{N-1}{x}\binom{N-1}{y}$
  - $N = 3$
  - $h(x,y) = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$
  - $h(x,y) = \left(\frac{1}{4}\right)^2\begin{bmatrix}1\\ 2\\ 1\end{bmatrix}\begin{bmatrix}1 & 2 & 1\end{bmatrix}$
  - Elements from Pascal's triangle

Averaging with Limited Data Validity

- Avoid blurring by averaging only pixels that meet a criterion function
  - Try to avoid including pixels from separate features
    - E.g. two sides of edge
- Define an invalid data interval $[min, max]$
- $h(i,j) = \begin{cases} 1 & for\ g(m+i, n+j) \notin [min, max] \\ 0 & else \end{cases}$
  - Convolution mask defined for each neighborhood
    - Only changes invalid data
    - Uses only valid data for averaging
- Define a brightness interval around central pixel
- Use gradient strength to only average those pixels with low gradients

(a)

(b)

Figure 5.10: Averaging with limited data validity. (a) Original corrupted image. (b) Result of corruption removal.

Rotating Mask Averaging

- Non-linear smoothing technique
  - Also sharpens image
- Idea is to determine a good neighborhood for averaging
- Calculate average over different mask rotations

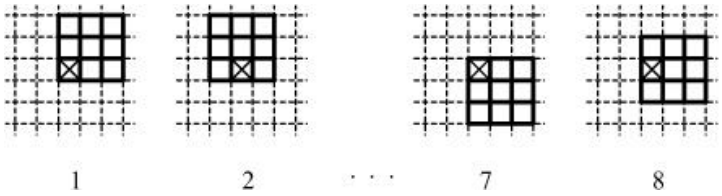


Figure 5.11: Eight possible rotated 3×3 masks. © *Cengage Learning 2015.*

- Homogeneity of region measured by a brightness dispersion

$$\sigma^2 = \frac{1}{n} \sum_{(i,j) \in R} \left(g(i,j) - \frac{1}{n} \sum_{(i,j) \in R} g(i,j) \right)^2 . \qquad (5.31)$$

Smoothing with Rotating Mask

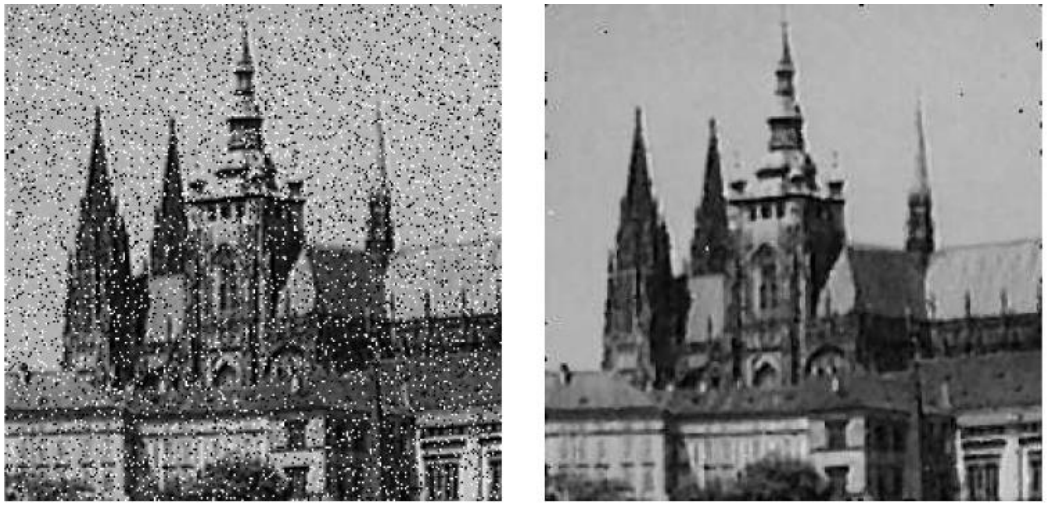
Algorithm 5.2: Smoothing using a rotating mask

1. Consider each image pixel (i, j).
2. Calculate dispersion for all possible mask rotations about pixel (i, j) according to equation (5.31).
3. Choose the mask with minimum dispersion.
4. Assign to the pixel $f(i, j)$ in the output image f the average brightness in the chosen mask.

- Only use "best" mask for pixel replacement
  - Looking for "stable" average
- Iterative solution convergence depends on mask size and shape
  - Smaller mask has smaller changes and more iterations
  - Large mask suppresses noise faster and has more sharpening
    - Small detail is lost

Median Filtering

- Non-linear smoothing method that reduces blurring of edges
  - Median not affected by noise spikes like mean
    - Removes impulse noise very well
- Iterative application is possible since blurring is not a problem
- More computationally expensive than linear filtering
  - Must sort values of all pixels in a neighborhood



(a) (b)

Figure 5.12: Median filtering. (a) Image corrupted with impulse noise (14% of image area covered with bright and dark dots). (b) Result of 3×3 median filtering. *© Cengage Learning 2015.*

Efficient Median Filtering

- Don't sort neighborhood for each pixel
  - Each pixel advance removes a column of pixels and adds a new column
  - $mn - 2m$ pixels are unchanged in pixel advance and do not need re-sorting
- Retain a histogram of pixel neighborhood values
  - Update histogram counts with removed and added values
  - Keep track of median value and adjust based on new values coming in
- See Algorithm 5.3

Median Filtering Upgrades

- Thin lines and sharp corners are destroyed
  - Look like noise in the neighborhood
  - Use preserving kernel

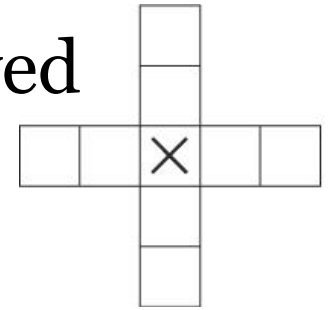


Figure 5.13: Horizontal/vertical line preserving neighborhood for median filtering. © *Cengage Learning 2015.*

- Rank filtering
  - Generalization of median filtering to use other statistics on ordered neighborhood values
    - E.g. max, min
- Order statistics
  - Neighborhood values ordered into sequence
  - Output value is a linear combination of sequence

Edge Detection

- Locate changes in image intensity function
  - Edges are abrupt changes
- Very important pre-processing step for many computer vision techniques
  - Object detection, lane tracking, geometry
- Edges are important neurological and psychophysical processes
  - Part of human image perception loop
  - Information reduction but not understanding
- Edgels – edge element with strong magnitude
  - Pixels with large gradient magnitude

Informative Edges

- Edges arise from various physical phenomena during image formation
  - Trick is to determine which edges are most important

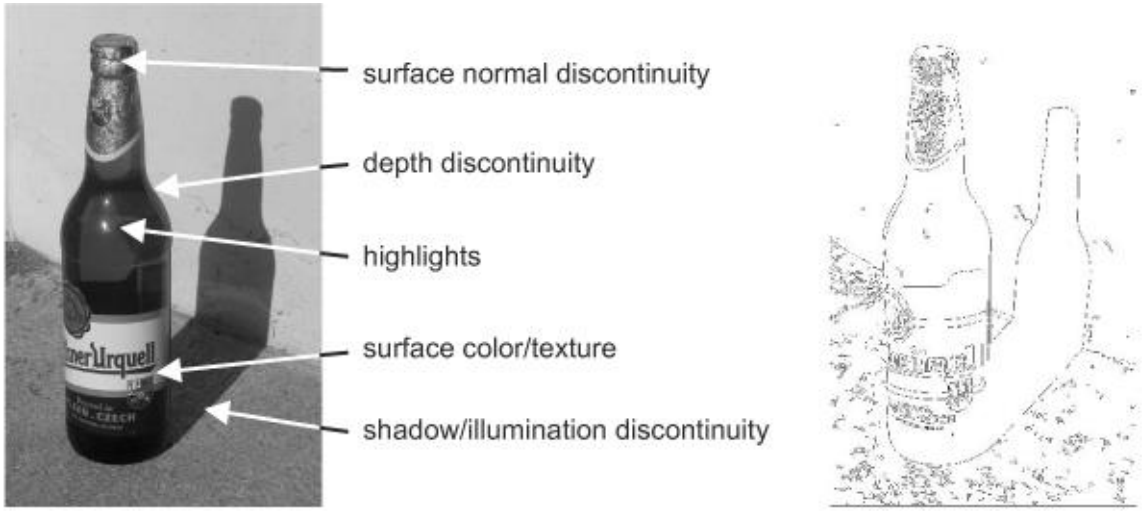


Figure 5.15: Origin of edges, i.e., physical phenomena in image formation process which lead to edges in images. At right, a Canny edge detection (see Section 5.3.5). © *Cengage Learning 2015.*

Edge Definition

- Edge defined at each pixel by gradient vector
  - Gives direction of maximal change
    - Points from black (0) to white (255)
- Describe edge by magnitude and direction
  - Edge direction is 90 degrees from gradient direction

$$|\operatorname{grad} g(x,y)| = \sqrt{\left(\frac{\partial g}{\partial x}\right)^2 + \left(\frac{\partial g}{\partial y}\right)^2}, \qquad \psi = \arg\left(\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}\right),$$

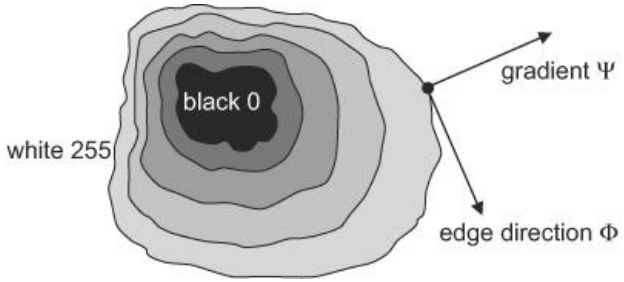


Figure 5.16: Gradient direction and edge direction. © *Cengage Learning 2015.*

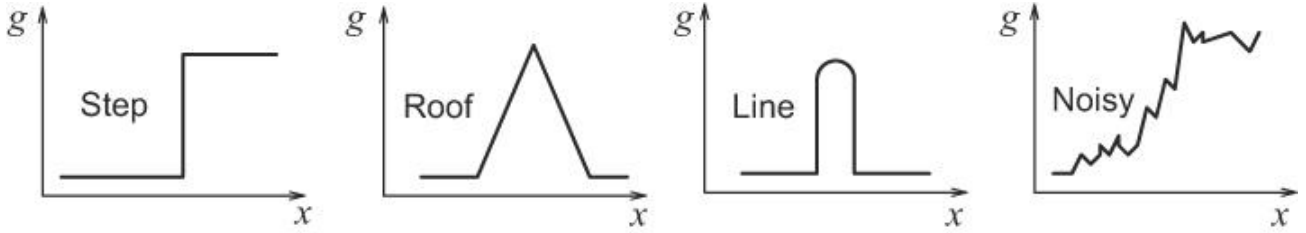


Figure 5.17: Typical edge profiles. © *Cengage Learning 2015.*

Image Sharpening

- Basic idea is to add a edge emphasized image back to the original image
  - The Laplacian operator is used to provide rotation invariance and focus only on edge strength

$$\nabla^2 g(x,y) = \frac{\partial^2 g(x,y)}{\partial x^2} + \frac{\partial^2 g(x,y)}{\partial y^2} . \qquad (5.35)$$

(a) (a) (b)

Figure 5.18: Laplace gradient operator. (a) Laplace edge image using the 8-connectivity mask. (b) Sharpening using the Laplace operator (equation 5.36, $C = 0.7$). Compare the sharpening effect with the original image in Figure 5.9a.

Categories of Gradient Operators

- Differences for derivative approximation
  - Rotational invariance can be computed in a single convolution
    - E.g. Laplacian concerned with magnitude only
  - First derivatives use multiple masks
    - Estimate orientation based on each mask response
- Zero crossing operators
  - Operate on the second derivative
    - E.g. Canny edge detector
- Parametric edge models
  - Define edge model and perform matching

- Very important operation for a variety of vision tasks

Edge Convolution Kernels

- A number of popular kernels have been designed
- Roberts operator
  - Small and fast
  - Sensitive to noise

$$h_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad h_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix},$$

- Laplace operator
  - Approximation of Laplacian
  - Rotation invariant
  - 4 or 8-connected neighbors

$$h = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

- Prewitt, Kirsch, Sobel
  - Family of kernels to approximate first derivatives at different orientations
  - Sobel used often for horizontal and vertical edges (`gradient.m`)
    - Magnitude - $|h_1| + |h_3|$
    - Direction - $\arctan(h_1/h_3)$

$$h_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \quad h_2 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}, \quad h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad \ldots$$

Oriented Edge Image

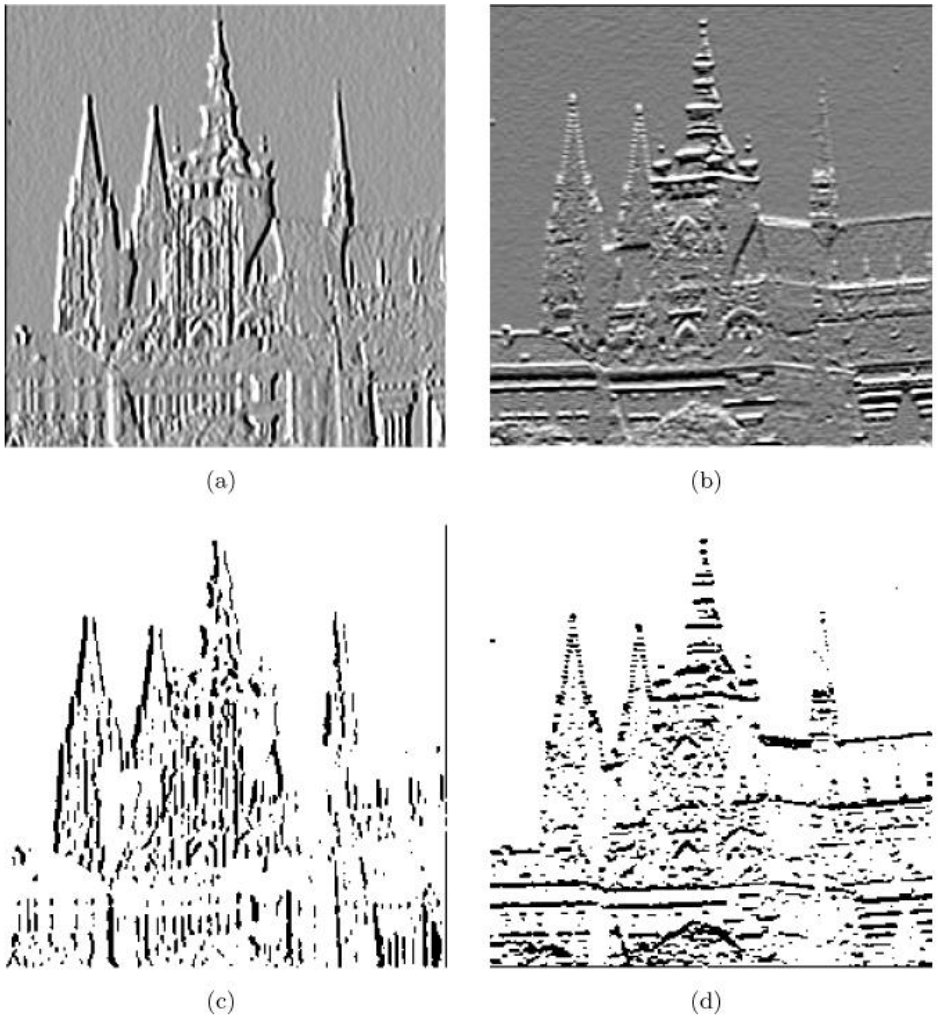


Figure 5.19: First-derivative edge detection using Prewitt operators. (a) North direction (the brighter the pixel value, the stronger the edge). (b) East direction. (c) Strong edges from (a). (d) Strong edges from (b).

Zero-Crossings of Second Derivative

- Edge can be localized with second derivative
  - Easier to find zero-crossing than maxima value

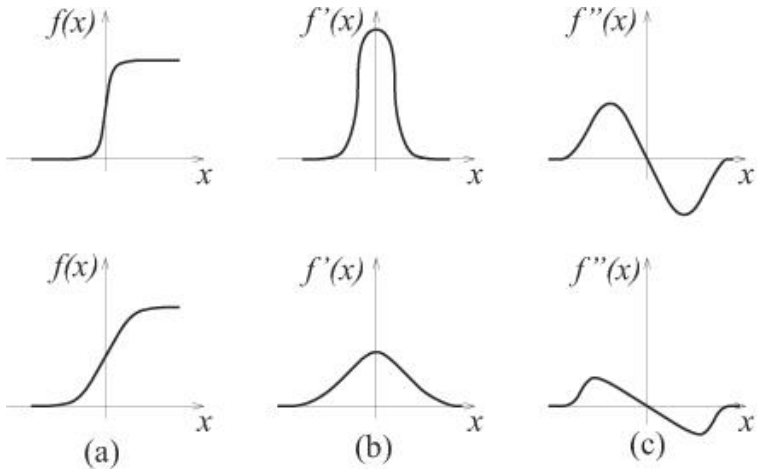


Figure 5.20: 1D edge profile of the zero-crossing. © *Cengage Learning 2015.*

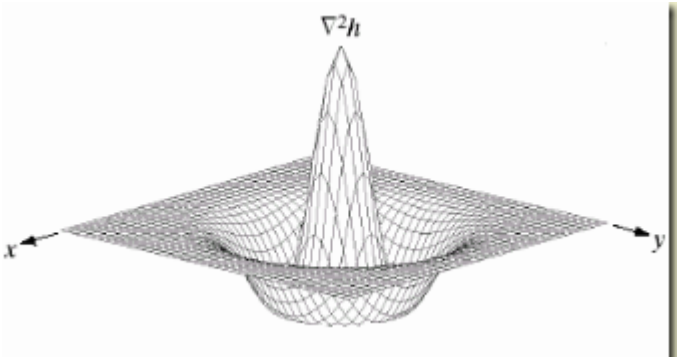
- Marr-Hildreth edge detector
  - Smooth image first to reduce noise before computing second derivative
  - Must determine how much to smooth
    - Bandlimit frequencies of change
    - Limit the spatial neighborhood

Gaussian Smoothed Edges

- Use Gaussian to smooth image $G(x,y) = e^{-(x^2+y^2)/2\sigma^2}$,
  - Standard deviation controls neighborhood size
- Compute second derivative of image after smoothing
  - Use Laplace operator ∇^2 $\nabla^2[G(x,y,\sigma) * f(x,y)]$.
    - Laplacian of Gaussian (LoG)
- Notice this requires convolution of image twice
  - Use linearity to simplify with derivative of Gaussian filter $\nabla^2 G$ - LoG operator (Mexican hat)

$$[\nabla^2 G(x,y,\sigma)] * f(x,y) .$$

$$h(x,y) = c\left(\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right) e^{-(x^2+y^2)/2\sigma^2} ,$$



Zero-Crossing Issues

- Generally will smooth image before derivative
- Gaussian σ can better control neighborhood
  - Larger σ more globally significant edges
  - Operations at many σ levels can give scale space response
- $\nabla^2 G$ can be effectively approximated
  - Difference of Gaussians (DoG)
  - Use difference between Gaussians of different σ values
- Zeros do not exist in LoG (DoG) image
  - Threshold generally gives disconnected edges
  - Need to search for transitions between polarities
    - Edge is between pixels
  - Improve performance by accounting for first derivative response

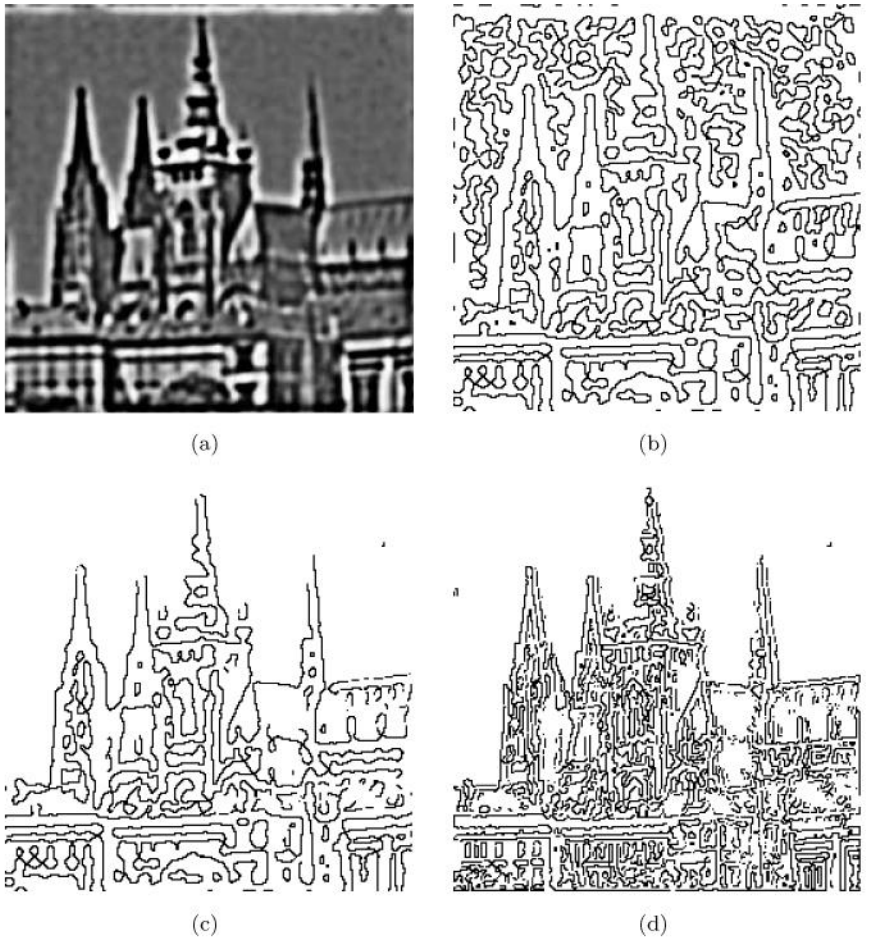


Figure 5.21: Zero-crossings of the second derivative, see Figure 5.9a for the original image. (a) DoG image ($\sigma_1 = 0.10, \sigma_2 = 0.09$), dark pixels correspond to negative values, bright pixels to positive. (b) Zero-crossings of the DoG image. (c) DoG zero-crossing edges after removing edges lacking first-derivative support. (d) LoG zero-crossing edges ($\sigma = 0.20$) after removing edges lacking first-derivative support—note different scale of edges due to different Gaussian smoothing parameters.

Canny Edge Detection

- Optimal edge detection algorithm
  - Returns long thin (1 pixel wide) connected edges
- Non-maximal edge suppression technique to return a single pixel for an edge
  - Examine pixels along gradient direction
  - Only retain pixel if it is larger than neighbors
- Hysteresis threshold to remove spurious responses and maintain long connected edges
  - High threshold used to find definite edges
  - Low threshold to track edges

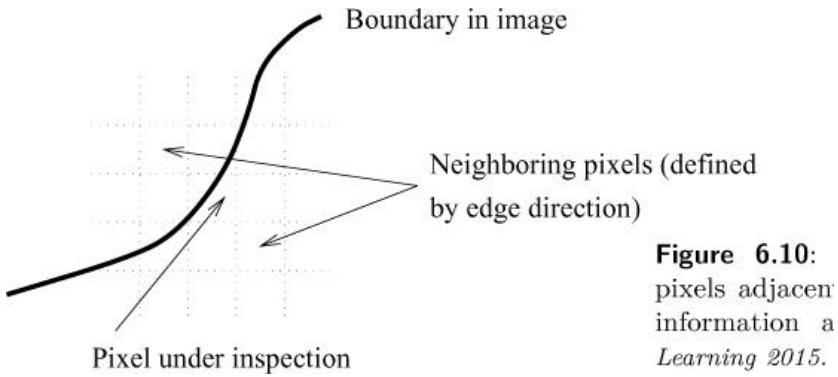
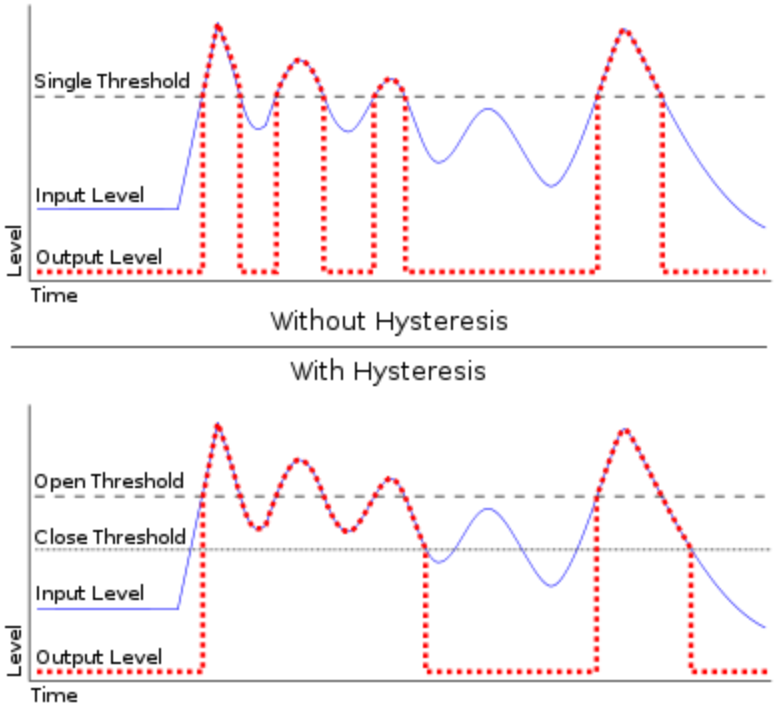
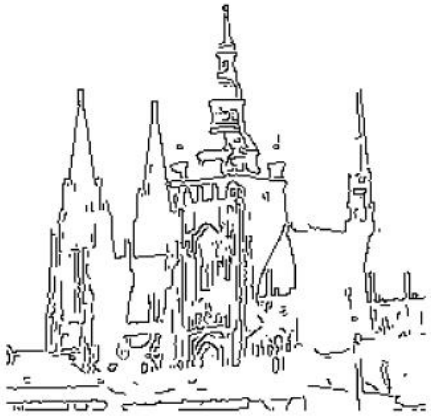


Figure 6.10: pixels adjacen information a *Learning 2015.*



Canny Edge Examples

(a)



(b)

(a)

(b)

Figure 6.11: (a) Non-maximal suppression of the data in Figure 6.9b. (b) Hysteresis applied to (a); high threshold 70, low threshold 10. © *Cengage Learning 2015*

(a)

(b)

Figure 5.23: Canny edge detection at two different scales. © *Cengage Learning 2015.*

Multispectral Edges

- Pixel (i, j) has n-dimensional vector representation
- Trivial edge detection
  - Operate on each spectral band separately
  - Combine all bands to form single edge image
- Multiband (Roberts-like) edge operator
  - $2 \times 2 \times n$ - neighborhood

$$\frac{\sum_{r=1}^{n} [d(i,j)]\,[d(i+1,j+1)]}{\sqrt{\sum_{r=1}^{n} [d(i,j)]^2 \sum_{r=1}^{n} [d(i+1,j+1)]^2}} \quad \frac{\sum_{r=1}^{n} [d(i+1,j)]\,[d(i,j+1)]}{\sqrt{\sum_{r=1}^{n} [d(i+1,j)]^2 \sum_{r=1}^{n} [d(i,j+1)]^2}},$$

where $d(k,l) = f_r(k,l) - \overline{f}(k,l)$.

(5.60)

Frequency Domain Pre-Processing

- Use Fourier transform for spatial frequency filtering
  - Convolution becomes multiplication in frequency domain
  - $G = F .* H$
    - Image $g(x, y)$ obtained by inverse Fourier transform
- Basic filters (rotationally symmetric)
  - Lowpass – smoothing
  - Highpass – edge detection/enhancement
  - Bandpass – enhancement (structured noise)

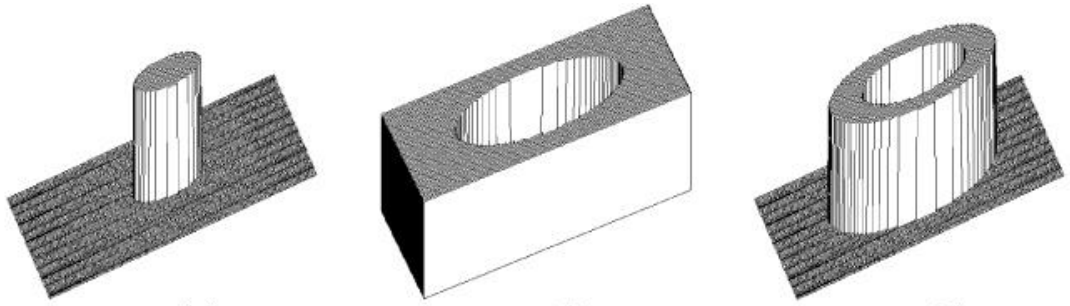


Figure 5.24: Frequency filters displayed in 3D. (a) Low-pass filter. (b) High-pass filter. (c) Band-pass filter. © *Cengage Learning 2015.*

Lowpass Filtering

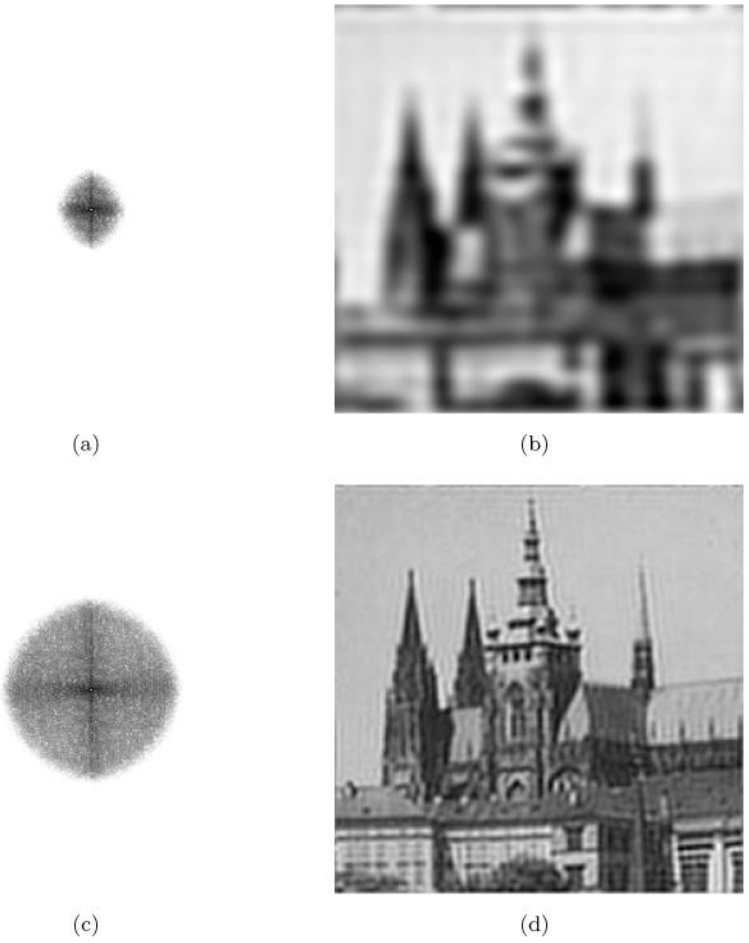


Figure 5.25: Low-pass frequency-domain filtering—for the original image and its spectrum see Figure 3.7. (a) Spectrum of a low-pass filtered image, all higher frequencies filtered out. (b) Image resulting from the inverse Fourier transform applied to spectrum (a). (c) Spectrum of a low-pass filtered image, only very high frequencies filtered out. (d) Inverse Fourier transform applied to spectrum (c).

Highpass Filtering

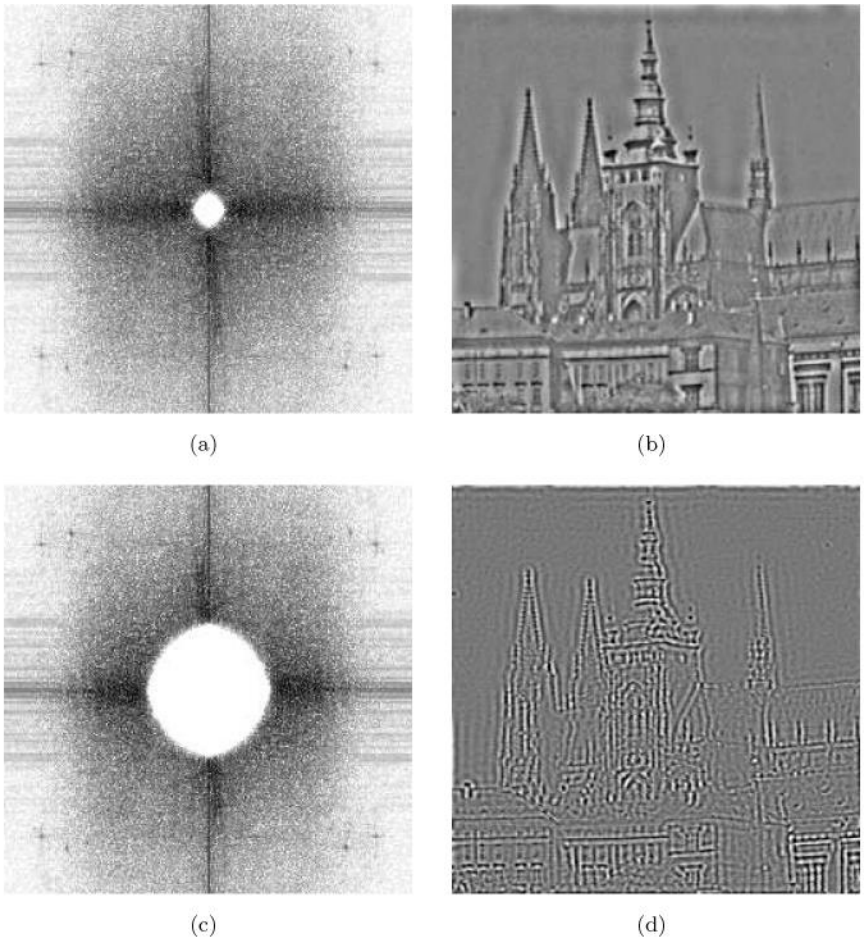
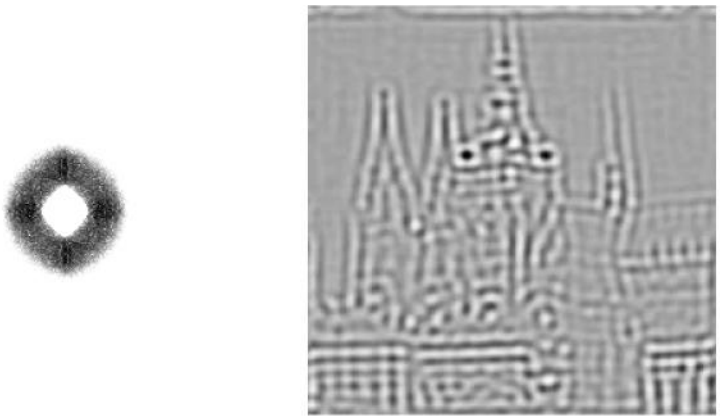


Figure 5.26: High-pass frequency domain filtering. (a) Spectrum of a high-pass filtered image, only very low frequencies filtered out. (b) Image resulting from the inverse Fourier transform applied to spectrum (a). (c) Spectrum of a high-pass filtered image, all lower frequencies filtered out. (d) Inverse Fourier transform applied to spectrum (c).

Bandpass Filtering

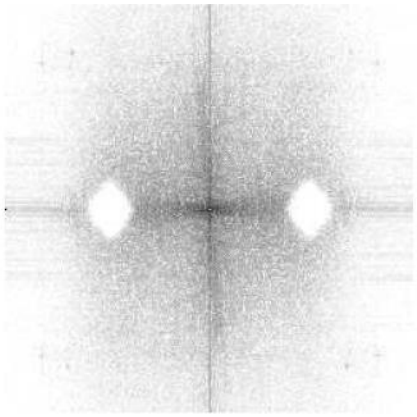


(a) (b)

Figure 5.27: Band-pass frequency domain filtering. (a) Spectrum of a band-pass-filtered image, low and high frequencies filtered out. (b) Image resulting from the inverse Fourier transform applied to spectrum (a).

Bandpass Noise Removal

(a)



(b)

(c)

Figure 5.28: Periodic noise removal. (a) Noisy image. (b) Image spectrum used for image reconstruction—note that the areas of frequencies corresponding with periodic vertical lines are filtered out. (c) Filtered image.

Fourier Image Processing

- Take Fourier transform of input image $f(x, y)$
- Take Fourier transform of kernel $h(x, y)$
  - Need to pad images
- Multiply $G = F .* H$
  - Element-wise multiplication
- Inverse Fourier transform for output image $g(x, y)$
  - Crop borders

- Filtering is generally more intuitive in frequency domain
- Frequency domain filtering is better with larger kernels
  - More effieicent

- Padding
- Remember the output of a convolution is a longer sequence
  - $g(x) = f(x) * h(x)$
    - $f(x)$ is length A
    - $h(x)$ is length B
    - $g(x)$ is length $A + B - 1$
- Need to pad to ensure the output of the convolution fits
  - Use a power of 2 for FFT
  - Padding makes each have the same implicit period
  - Crop off un-needed areas

Detection of Corners (Interest Points)

- Useful for fundamental vision techniques
  - Image matching or registration
- Correspondence problem needs to find all pairs of matching pixels
  - Typically a complex problem
  - Can be made easier only considering a subset of points
- Interest points are these important image regions that satisfy some local property
  - Corners are a way to get to interest points

Feature Detection and Matching

- Essential component of modern computer vision
  - E.g. alignment for image stitching, correspondences for 3D model construction, object detection, stereo, etc.

- Need to establish some features that can be detected and matched

Determining Features to Match

- What can help establish correspondences between images?




Different Types of Features

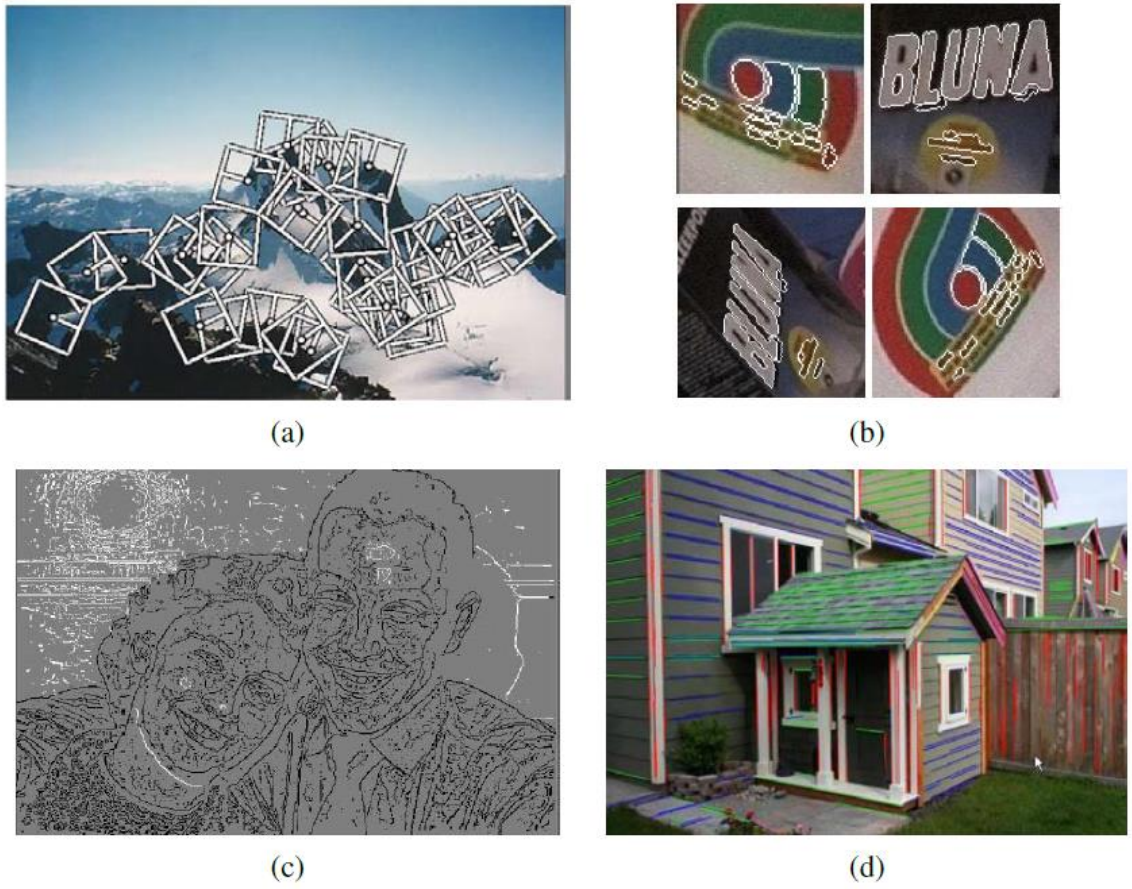


Figure 4.1 A variety of feature detectors and descriptors can be used to analyze, describe and match images: (a) point-like interest operators (Brown, Szeliski, and Winder 2005) © 2005 IEEE; (b) region-like interest operators (Matas, Chum, Urban *et al.* 2004) © 2004 Elsevier; (c) edges (Elder and Goldberg 2001) © 2001 IEEE; (d) straight lines (Sinha, Steedly, Szeliski *et al.* 2008) © 2008 ACM.

Different Types of Features

- Points and patches
- Edges
- Lines

- Which features are best?
  - Depends on the application
  - Want features that are robust
    - Descriptive and consistent (can readily detect)

Points and Patches

- Maybe most generally useful feature for matching
  - E.g. Camera pose estimation, dense stereo, image stitching, video stabilization, tracking
  - Object detection/recognition

- Key advantages:
  - Matching is possible even in the presence of clutter (occlusion)
  - and large scale and orientation changes

Point Correspondence Techniques

- Detection and tracking
  - Initialize by detecting features in a single image
  - Track features through localized search
  - Best for images from similar viewpoint or video
- Detection and matching
  - Detect features in all images
  - Match features across images based on local appearance
  - Best for large motion or appearance change

Keypoint Pipeline

- Feature detection (extraction)
  - Search for image locations that are likely to be matched in other images
- Feature description
  - Regions around a keypoint are represented as a compact and stable descriptor
- Feature matching
  - Descriptors are compared between images efficiently
- Feature tracking
  - Search for descriptors in small neighborhood
  - Alternative to matching stage best suited for video

Feature Detectors

- Must determine image locations that can be reliably located in another image

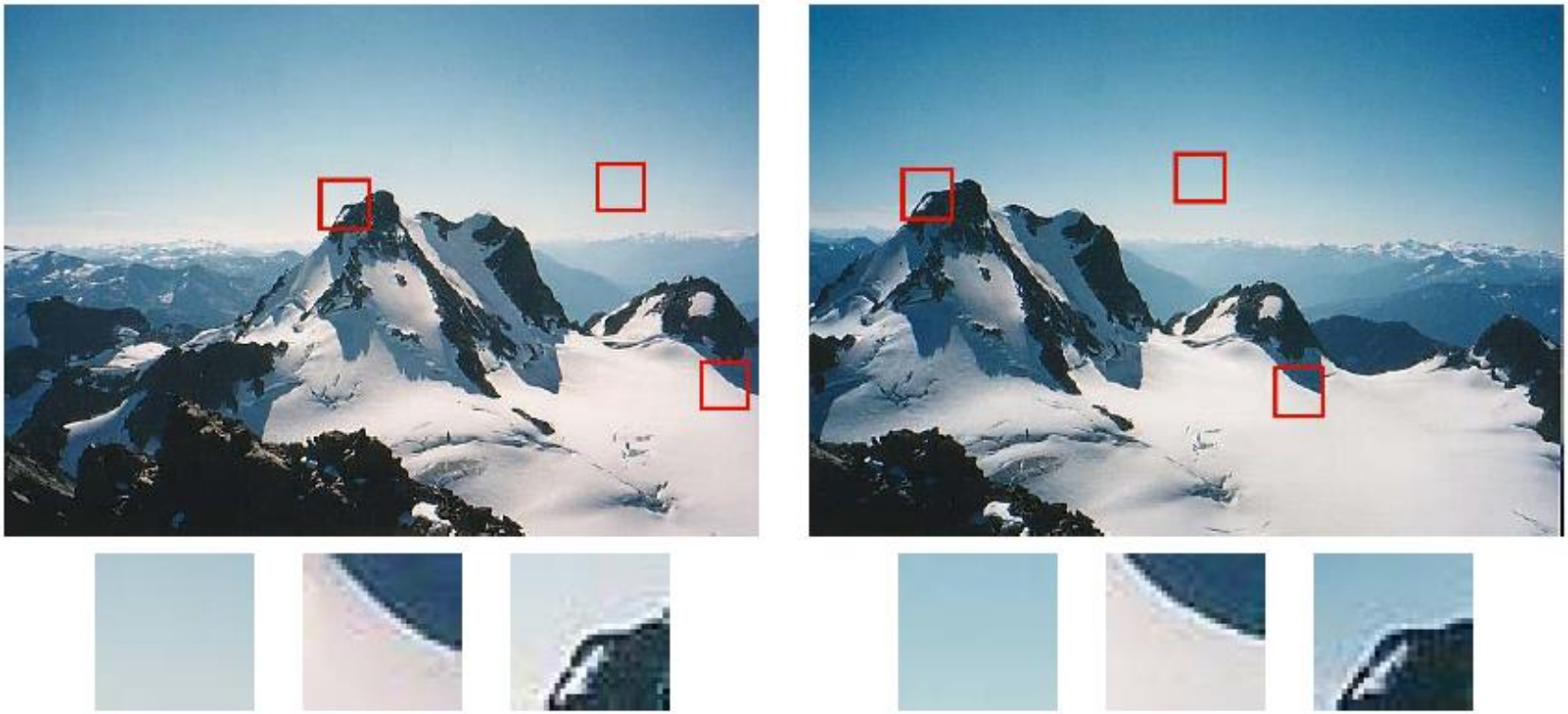


Figure 4.3 Image pairs with extracted patches below. Notice how some patches can be localized or matched with higher accuracy than others.

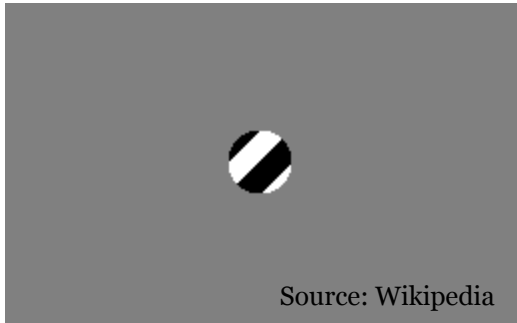
Comparison of Image Patches

- Textureless patches
  - Nearly impossible to localize and match
    - Sky region “matches” to all other sky areas
- Edge patches
  - Large contrast change (gradient)
  - Suffer from aperture problem
    - Only possible to align patches along the direction normal the edge direction
- Corner patches
  - Contrast change in at least two different orientations
  - Easiest to localize

Aperture Problem I

- Only consider a small window of an image
  - Local view does not give global structure
  - Causes ambiguity

- Best visualized with motion (optical flow later)
  - Imagine seeing the world through a straw hole
  - Aperture Problem - MIT – Demo
  - Also known as the barber pole effect



Source: Wikipedia

Aperture Problem II

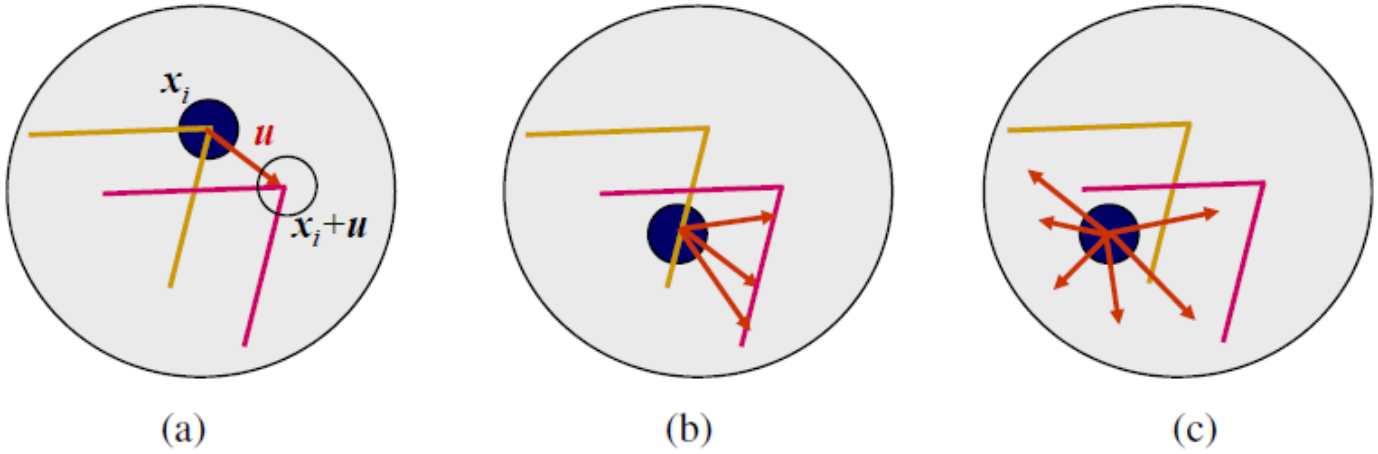


Figure 4.4 Aperture problems for different image patches: (a) stable ("corner-like") flow; (b) classic aperture problem (barber-pole illusion); (c) textureless region. The two images I_0 (yellow) and I_1 (red) are overlaid. The red vector u indicates the displacement between the patch centers and the $w(x_i)$ weighting function (patch window) is shown as a dark circle.

- Corners have strong matches
- Edges can have many potential matches
  - Constrained upon a line
- Textureless regions provide no useful information

WSSD Matching Criterion

- Weighted summed squared difference
  - $E_{WSSD}(\boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)\,[I_1(\boldsymbol{x}_i - \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$
    - I_1, I_0 - two image patches to compare
    - $\boldsymbol{u} = (u, v)$ – displacement vector
    - $w(\boldsymbol{x})$ - spatial weighting function
- Normally we do not know the image locations to perform the match
  - Calculate the autocorrelation in small displacements of a single image
    - Gives a measure of stability of patch
  - $E_{AC}(\Delta\boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)\,[I_0(\boldsymbol{x}_i - \Delta\boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$

Image Patch Autocorrelation

$$E_{AC}(\Delta \boldsymbol{u}) = \sum_i w(\boldsymbol{x}_i)\,[I_0(\boldsymbol{x}_i - \Delta \boldsymbol{u}) - I_0(\boldsymbol{x}_i)]^2$$
$$= \sum_i w(\boldsymbol{x}_i)\,[\nabla I_0(\boldsymbol{x}_i) \cdot \Delta \boldsymbol{u}]^2$$
$$= \Delta \boldsymbol{u}^T A \Delta \boldsymbol{u}$$

- $\nabla I_0(\boldsymbol{x}_i)$ - image gradient
  - We have seen how to compute this
- A – autocorrelation matrix

$$A = w * \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}$$

  - Compute gradient images and convolve with weight function
  - Also known as second moment matrix
  - (Harris matrix)

- Example autocorrelation

Image Autocorrelation II

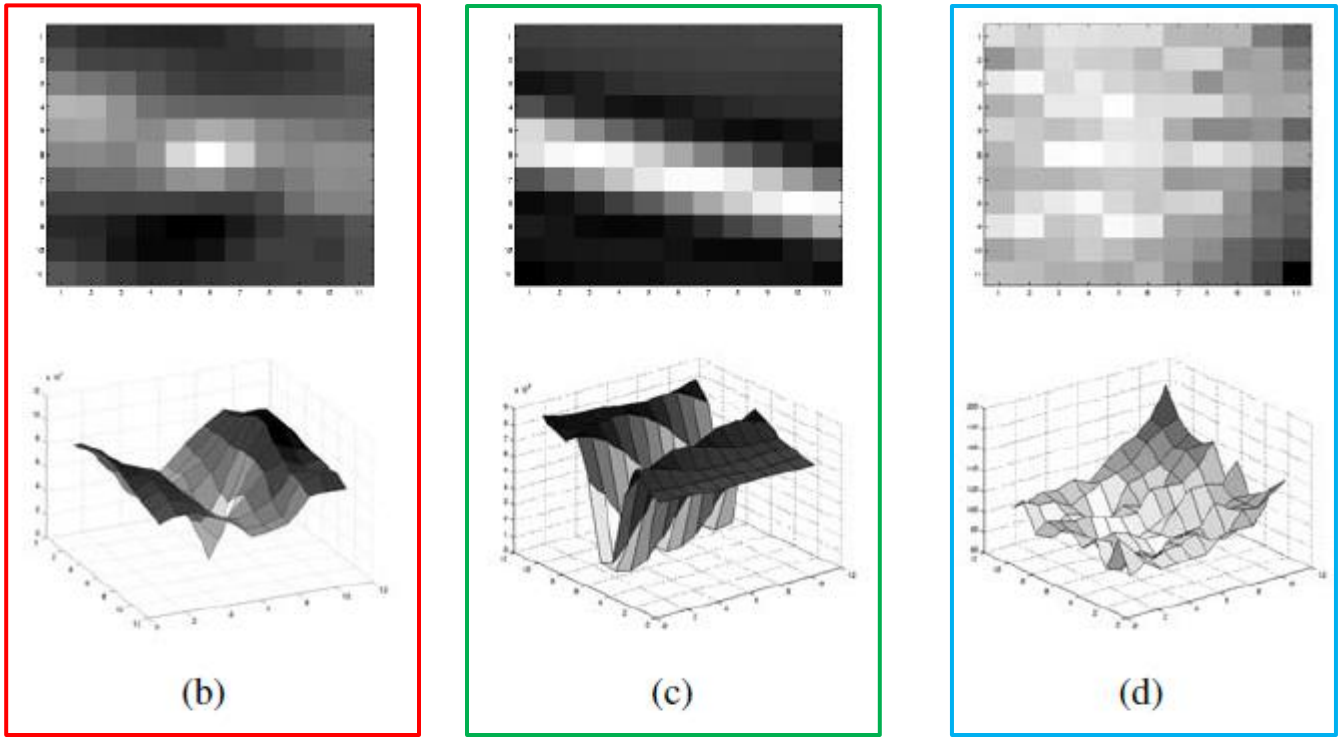


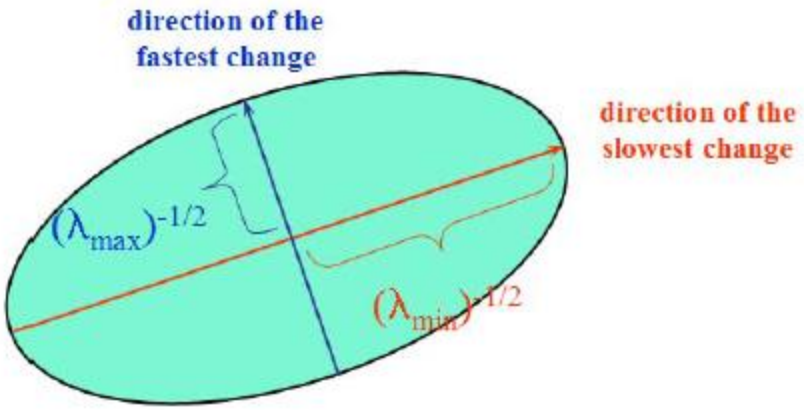
Figure 4.5 Three auto-correlation surfaces $E_{\mathrm{AC}}(\Delta u)$ shown as both grayscale images and surface plots: (a) The original image is marked with three red crosses to denote where the auto-correlation surfaces were computed; (b) this patch is from the flower bed (good unique minimum); (c) this patch is from the roof edge (one-dimensional aperture problem); and (d) this patch is from the cloud (no good peak). Each grid point in figures b–d is one value of Δu.

Image Autocorrelation III

- The matrix A provides a measure of uncertainty in location of the patch
- Do eigenvalue decomposition
  - Get eigenvalues and eigenvector directions

- Good features have both eigenvalues large
  - Indicates gradients in orthogonal directions (e.g. a corner)

- Uncertainty ellipse



- Many different methods to quantify uncertainty
  - Easiest: look for maxima in the smaller eigenvalue

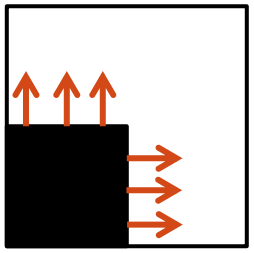
Basic Feature Detection Algorithm

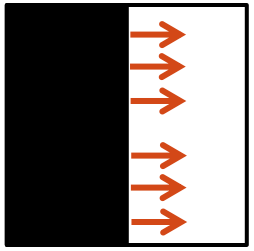
1. Compute the horizontal and vertical derivatives of the image I_x and I_y by convolving the original image with derivatives of Gaussians (Section 3.2.3).
2. Compute the three images corresponding to the outer products of these gradients. (The matrix $\boldsymbol{A}$ is symmetric, so only three entries are needed.)
3. Convolve each of these images with a larger Gaussian.
4. Compute a scalar interest measure using one of the formulas discussed above.
5. Find local maxima above a certain threshold and report them as detected feature point locations.

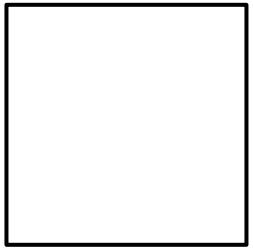
Algorithm 4.1 Outline of a basic feature detection algorithm.

Interest Point Detection

- The correlation matrix gives a measure of edges in a patch
- Corner
  - Gradient directions
    - $\begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}$
  - Correlation matrix
    - $A \propto \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$
- Edge
  - Gradient directions
    - $\begin{bmatrix}1\\0\end{bmatrix}$
  - Correlation matrix
    - $A \propto \begin{bmatrix}1 & 0\\0 & 0\end{bmatrix}$
- Constant
  - Gradient directions
    - $\begin{bmatrix}0\\0\end{bmatrix}$
  - Correlation matrix
    - $A \propto \begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$







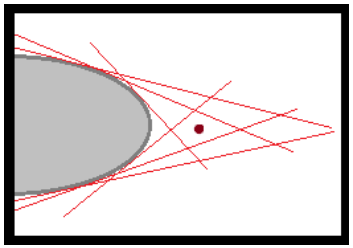
Harris Corners

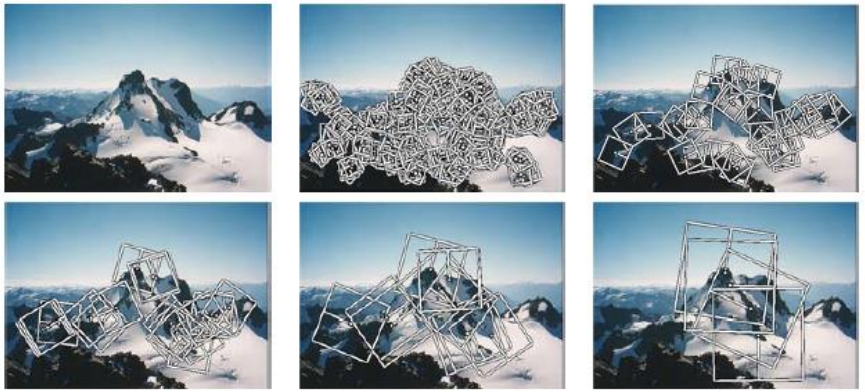
Improving Feature Detection

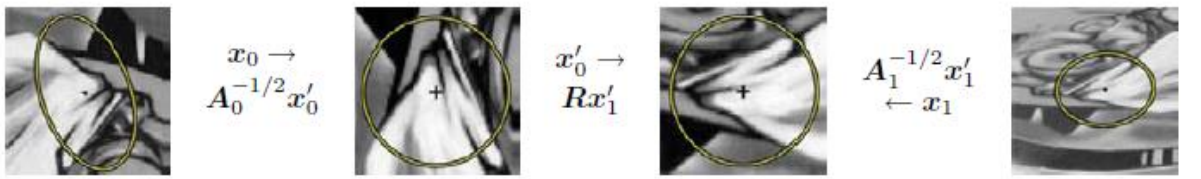
- Corners may produce more than one strong response (due to neighborhood)
  - Estimate corner with subpixel accuracy – use edge tangents
  - Non-maximal suppression – only select features that are far enough away
    - Create more uniform distribution – can be done through blocking as well
- Scale invariance
  - Use an image pyramid – useful for images of same scale
  - Compute Hessian of difference of Gaussian (DoG) image
  - Analyze scale space [SIFT – Lowe 2004]
- Rotational invariance
  - Need to estimate the orientation of the feature by examining gradient information
- Affine invariance
  - Closer to appearance change due to perspective distortion
  - Fit ellipse to autocorrelation matrix and use it as an affine coordinate frame
  - Maximally stable region (MSER) [Matas 2004] – regions that do not change much through thresholding

(a) Strongest 250

(c) ANMS 250, $r = 24$



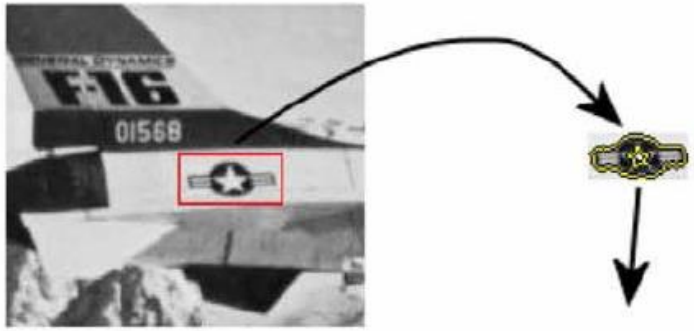


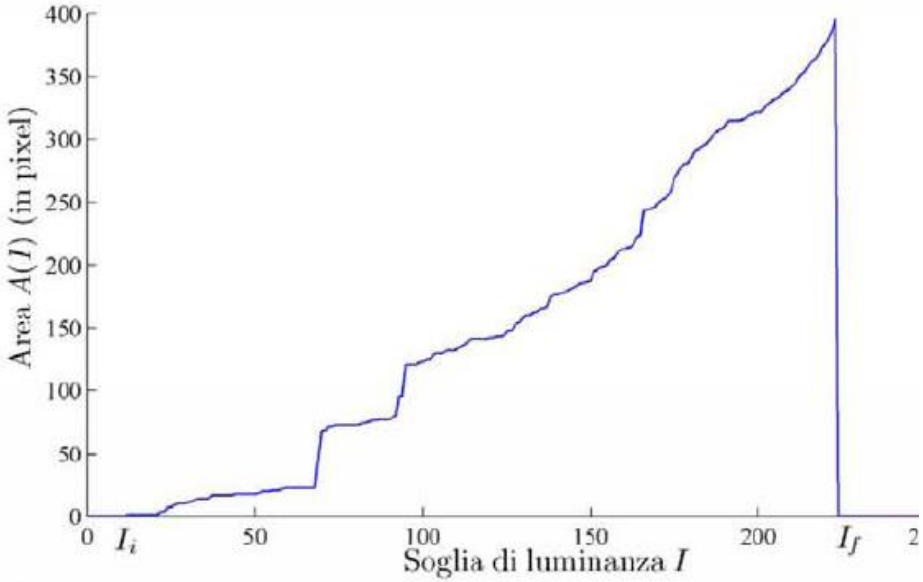


Maximally Stable Extremal Regions

- MSERs are image structures that can be recovered after translations, rotations, similarity (scale), and affine (shear) transforms
- Connected areas characterized by almost uniform intensity, surrounded by contrasting background
- Constructed based on a watershed-type segmentation
  - Threshold image a multiple different values
  - MSERs are regions with shape that does not change much over thresholds
- Each region is a connected component but no global or optimal threshold is selected

MSER





- Red borders from increasing intensity
- Green boarders from decreasing intensity

MSER Invariance

- Fit ellipse to area and normalize into circle