Computer Vision

Edge Detection

Augusto Silva

Department of Electronics, Telecommunications and Informatics
University of Aveiro

augusto.silva@ua.pt

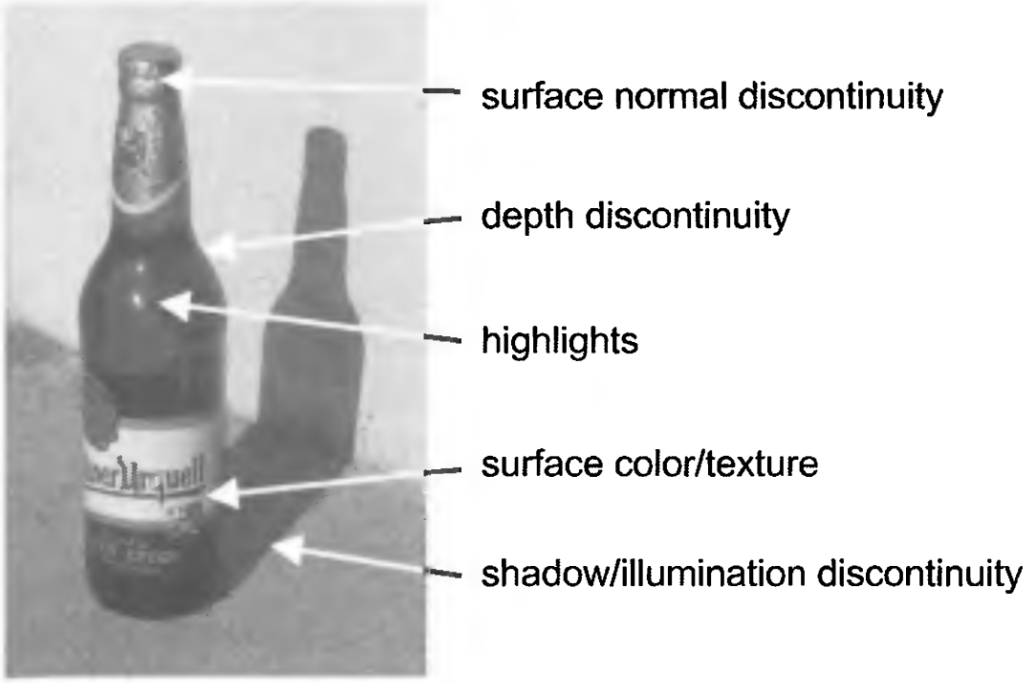
Outline

Edge detectors

- Very important local image pre-processing methods used to locate changes in the intensity function;
- Edges play a key role in human and machine perception
- Sometimes only edge elements with strong magnitude (edgels) suffice for image understanding.
- Significant data reduction for image representation
- A change of the image function can be described by a gradient that points in the direction of the largest growth of the image function.
- An edge is a property attached to an individual pixel and is calculated from the image function behavior in a neighborhood of that pixel.

Edge Sources

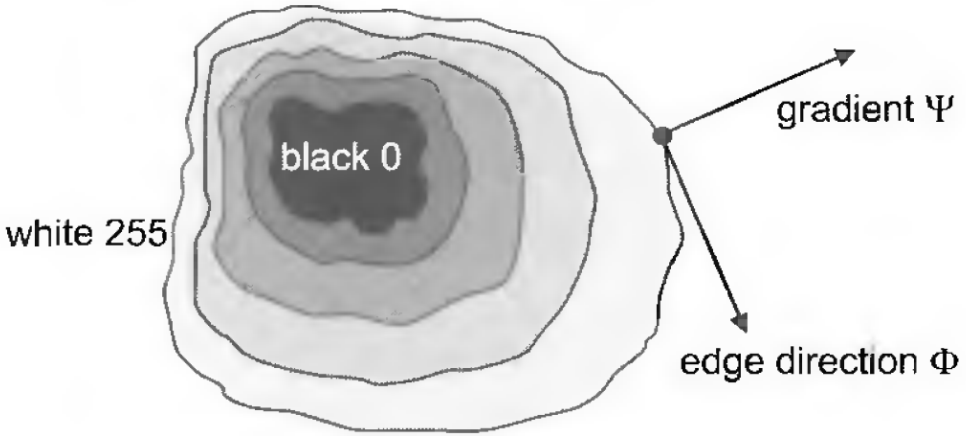
- Problem: How to retrieve the relevant edges?



Adapted from Image Processing , Analysis, and Machine Vision 3Ed, Sonka et al

Edge Concepts

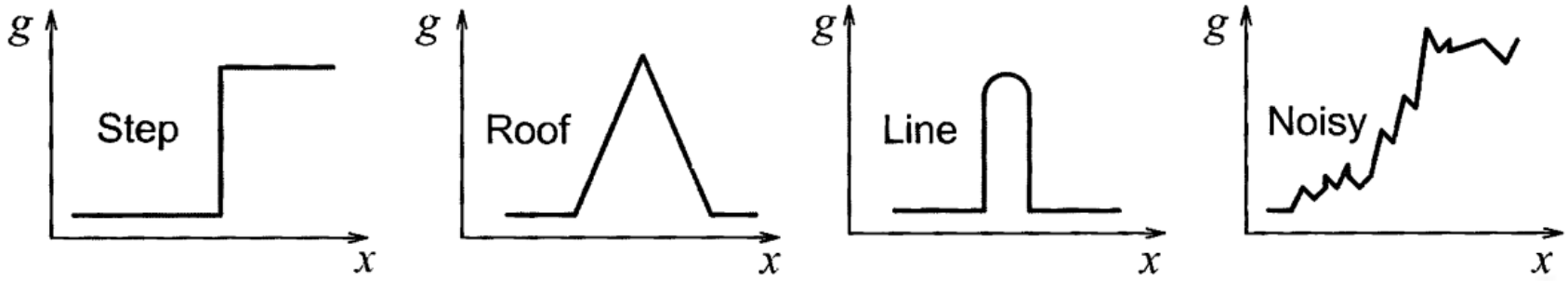
- An edge is a vector variable with two components, magnitude and direction
  - The edge magnitude is the magnitude of the gradient
  - The edge direction f is rotated with respect to the gradient direction by $90°$.
  - The gradient direction points to the maximum growth of the function.
  - Ideally boundaries are made of pixels with edge magnitues



Adapted from Image Processing , Analysis, and Machine Vision 3Ed, Sonka et al

Edge Profiles

- The edge profile is a plot of image values along the gradient direction (perpendicular to the edge direction).
- Typical profiles:



Adapted from Image Processing , Analysis, and Machine Vision 3Ed, Sonka et al

- Roof edges are typical for objects corresponding to thin lines in the image.
- Edge detectors are usually tuned for some type of edge profile.

Mathematical Tools

- The Gradient (continuous version) is a vector field requiring magnitude and angular descriptors

$$\nabla f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) = (g_x, g_y)$$

$$|\nabla f(x,y)| = \sqrt{g_x^2 + g_y^2},\ \phi = \arg(g_x, g_y)$$

  - Symmetrical discrete approxmation:
    $\Delta_i g(i,j) = g(i,j) - g(i-n,j),\ \Delta_j g(i,j) = g(i,j) - g(i,j-n)$
- Sometimes only edge magnitudes without regard to their orientation is important. The Laplacian is the linear differential operator of choice since it is rotationally invariant.

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Discrete approximations

- Assymmetrical 1st derivative discrete approximation (including pixel (i, j)):

$$\begin{cases} \Delta_i g(i,j) &= g(i,j) - g(i-n,j) \\ \Delta_j g(i,j) &= g(i,j) - g(i,j-n) \end{cases}$$

- Symmetrical 1st derivative discrete approximation (excluding pixel (i, j)):

$$\begin{cases} \Delta_i g(i,j) &= g(i+n,j) - g(i-n,j) \\ \Delta_j g(i,j) &= g(i,j+n) - g(i,j-n) \end{cases}$$

- Possible Laplacian discretization

$$\nabla^2 f_{i,j} = (f_{i+1,j} + f_{i-1,j} + f_{i,j+1} + f_{i,j-1}) - f_{i,j}$$

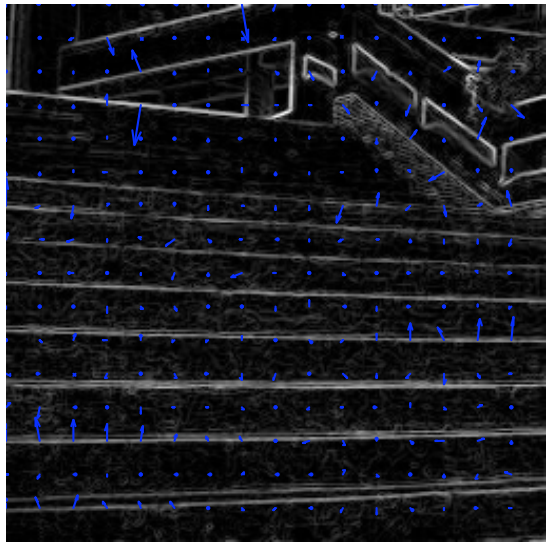
Example

Exercise

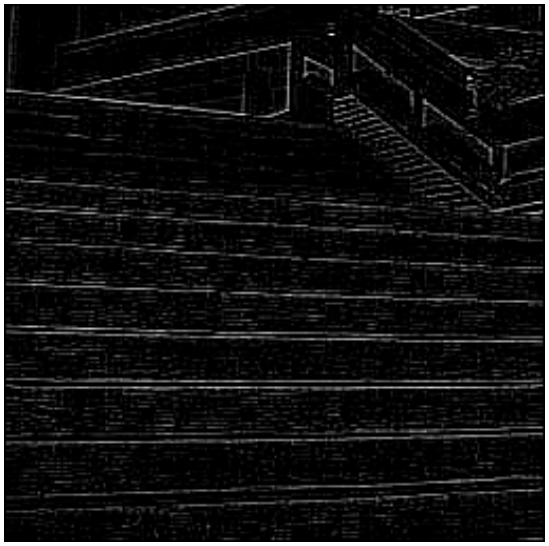
- Read de image *stairs.tiff*
- Use Matlab functions `gradient` and `del2` to compute the gradient and laplacian of image
- Display the resulting images
- Superimpose a *quiver* plot to see the gradient directions

Original

Gradient



Laplacian



Edge sharpening

Exercise

- Compute the image Laplacian (use the *stairs.tiff* image)
- Observe the edge sharpening effects for various values of $C \geq 0$ when computing

$$f_{i,j} = f_{i,j} - C \times L_{i,j}$$

- Use a gaussian filter to compute a blurred version B of the image
- Observe the edge sharpening effects for various values of $C \geq 0$ when computing

$$f_{i,j} = f_{i,j} + C \times (f_{i,j} - B_{i,j})$$

- This operation is a digital version of the so called unsharp mask operation

Differential operators for edge detection

- Operators approximating derivatives of the image function using differences:
  - Rotationally invariant
    - Laplacian
  - First derivative approximation
    - Multi-mask
    - Orientation determined upon a best match criteria
- Operators based on the zero-crossings of the image function second derivative
  - Marr-Hildreth
  - Canny
- Operators which attempt to match an image function to a parametric model of edges.

Edges with first derivatives

- Individual gradient operators that examine small local neighborhoods are convolutions
- Operators which are able to detect edge direction are represented by a collection of masks, each corresponding to a certain direction.
- Roberts operator

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

$$|f(i,j)| = |g(i,j) - g(i+1,j+1)| + |g(i,j+1) - g(i+1,j)|$$

- Simple, fast but very noise sensitive

Compass operators

- The gradient is estimated in eight (for a 3 x 3 convolution mask) possible directions
- The convolution result of greatest magnitude indicates the gradient direction.
- Prewitt

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

- Sobel

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

Compass operators

- Robinson

$$h_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad h_2 = \begin{bmatrix} 0 & 1 & 21 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \quad h_3 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

- Kirsch

$$h_1 = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix} \quad h_2 = \begin{bmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{bmatrix} \quad h_3 = \begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix}$$

Vertical and horizontal edges

Exercise

- Use the image *stairs.tiff*.
- Obtain a horizontal edge function E_h with the Sobel operator h_1
- Obtain a vertical edge function E_v with the Sobel operator h_2
- Obtain a global edge function making

$$E_g = |E_h| + |E_v|$$

- Threshold each filtered image with the IP toolbox functions `graythresh` and finally obtain the binary edge map with `im2bw`
- Repeat the exercise with various instances of zero mean gaussian noise superimposed on the image.

The Laplacian

- Second derivative approximation of ∇^2
- Convolution mask implementation

4-neighboorhood

$$h = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

8-neighboorhood

$$h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- Non rotationally invariant alternatives

$$h = \begin{bmatrix} 2 & 1 & 2 \\ 1 & -4 & 1 \\ 2 & 1 & 2 \end{bmatrix}$$

$$h = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix}$$

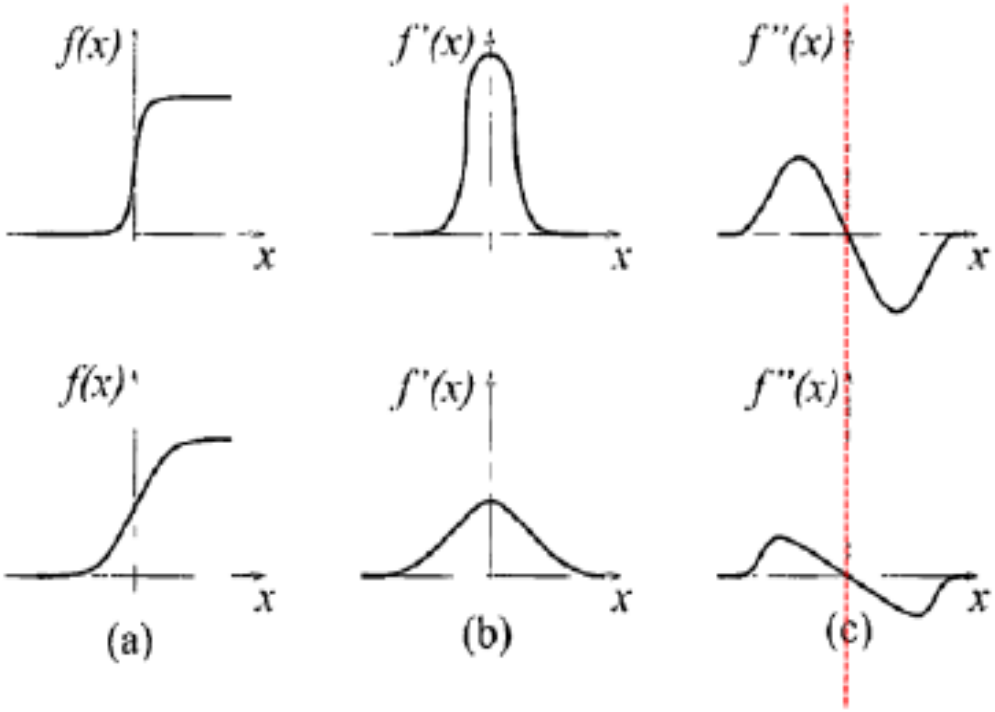
- Problem with Laplacians: double responses in some edge instances

Zero-crossings of the second derivative

- Marr's theory
- Neurophysiological experiments indicate that object boundaries are the most important cues that link an intensity image with its interpretation.
- Edge detection with first derivative approximations over small neighbourhoods are very noise sensitive and object dependent
- An edge detection technique based on the zero-crossings of the second derivative explores the fact that a step edge corresponds to an abrupt change in the image function.
  - The first derivative of the image function should have an extremum at the position corresponding to the edge
  - It is much easier and more precise to find a zero-crossing position than an extremum.

Edge profiles revisited

- Consistency for various edges profiles



Adapted from Image Processing , Analysis, and Machine Vision 3Ed, Sonka et al

- How to compute the second derivative robustly?

Laplacian of Gaussian - LoG

- The LoG operator integrates
  - A low-pass smoothing operator to minimize noise effects and control the influence of neighbour pixels
  - The Laplacian operator to compute de second derivative
- Continuous model:

$$\mathrm{LoG}\left[f(x,y)\right] = \nabla^2\left[G(x,y,\sigma) * * f(x,y)\right]$$

  - The order of performing differentiation and convolution can be interchanged because of the linearity of the operators involved

$$\mathrm{LoG}\left[f(x,y)\right] = \left[\nabla^2 G(x,y,\sigma)\right] * * f(x,y)$$

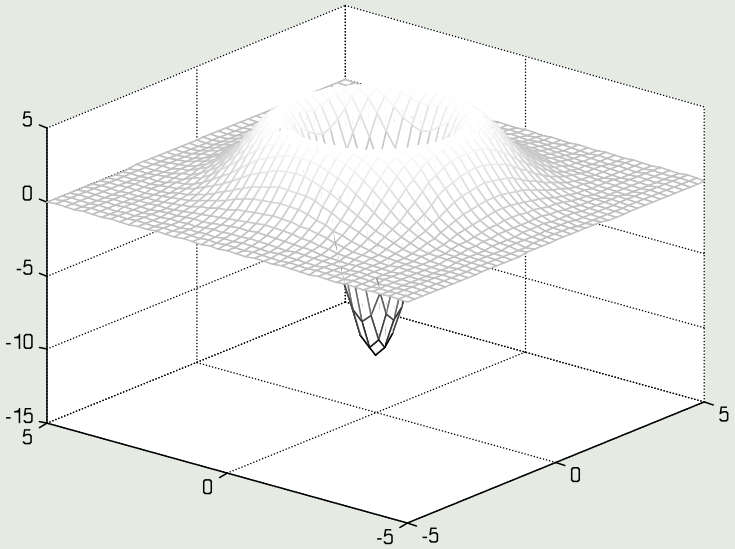
  - Gaussian is rotationally invariant so it is useful to make

$$r^2 = x^2 + y^2, \;\; G(x,y,\sigma) = G(r,\sigma) = e^{-r^2/\sigma^2}$$

The Mexican Hat function

Exercise

- Compute the second derivative $\nabla^2 G(r, \sigma)$
- Show that returning to the original co-ordinates x, y and introducing a normalizing multiplicative coefficient c, we get an inverted Mexican Hat function
- Make a 3D plot



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

(Reversed) Kernel Approximation

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

LoG issues

- Gaussian smoothing effectively suppresses the influence of the pixels that are more than a distance 3σ from the current pixel
  - A significant area surrounding the current pixel is taken into account; the influence of more distant points decreases according to the 3σ of the Gaussian.
- The Laplace operator is an efficient and stable measure of changes in the image.
- Convolution masks become large for larger σ for example, a $\sigma = 4$ needs a mask about 40 pixels wide.
- Separable decomposition of $\nabla^2 G$ exists to speed up the filtering process
- Be aware of sharp corner missing and fake close loops of edges

DoG: Difference of Gaussians

- The practical implication of Gaussian smoothing is that edges are found reliably.
  - Increase the value of σ if only globally significant edges are required
- The $\nabla^2 G$ operator can be very effectively approximated by a convolution with a mask that is the difference of two Gaussian averaging masks with substantially different σ
  - Difference of Gaussians (DoG) method
  - The ratio of standard deviations is an issue
- Coarse implementation $\nabla^2 G$
  - Filter the image twice with two different smoothing gaussian masks and obtain the difference

Looking for Zero Crossings

- Brute force zero search fails
- Naive search over interval produce piecewise disconnected edges
- Necessary to have a robust zero-crossing detector
  - Scan the LoG or DoG image with a 2x2 moving window
    - Assign an edge label to any one corner pixel, say the upper left, if LoG/DoGimage values of both polarities occur in the 2x2 window.
    - No edge label would be given if values within the window are either all positive or all negative.
  - Use first derivative (gradient support)
    - Only those zero-crossings for which there is sufficient edge evidence from a first-derivative edge detector.

Canny Edge Detection

- Canny proposed an approach to edge detection that is optimal for step edges corrupted by white noise
- Criteria
  - The detection criterion expresses the fact that important edges should not be missed and that there should be no spurious responses.
  - The localization criterion says that the distance between the actual and located position of the edge should be minimal.
  - The one response criterion minimizes multiple responses to a single edge.

Canny Edge Detection: Fundamentals

- A step edge is given by its position, orientation, and possibly magnitude (strength).
- The driving ideas:
  - Convolve the image with a symmetric 2D Gaussian filter
  - Differentiate in the direction of the gradient (perpendicular to the edge direction) responses.
  - Note that LoG/DoG zero-crossing operator does not give information about edge direction, as it uses a Laplacian filter).
- A direccional derivative of the Gaussian filter is given by

$$G_n = \frac{\partial G}{\partial \mathbf{n}} = \mathbf{n} \nabla G$$

- The direction $\mathbf{n}$ should be oriented perpendicular to the edge

Canny Edge Detection: Edge estimation

- A good estimate of the perpendiculcar direction to the edge can be obtained by smoothed gradient direction

$$\mathbf{n} = \frac{\nabla (G * f)}{|(G * f)|}$$

- The edge location correspondonds to the local maximum of the image f convolved with the operator $G_{\mathbf{n}}$ in the direction $\mathbf{n}$

$$G_n = \frac{\partial (G * f)}{\partial \mathbf{n}} = 0 \therefore \frac{\partial^2 (G * f)}{\partial \mathbf{n}^2} = 0$$

- Spurious responses to the single edge caused by noise usually create a 'streaking' problem
- Canny's procedure include threshold with hysteresis

Canny Edge Detection: Multi-scale edge prediction

- Multi-scale predicition of edge strength
  - What's the correct scale of the operation?
  - Not known a priori
  - Canny proposed a feature synthesis approach.
  - Start by marking all significant edges from the operator with the smallest scale
  - Synthesize edges of a hypothetical operator with larger σ
  - Compare the edge response to the actual response for larger σ
  - Additional edges are marked only if they have a significantly stronger response than that predicted from synthetic output.
  - Repeat for various scales
  - Build a cumulative edge map

Canny Edge Detection: Algorithm

1. Convolve an image f with a Gaussian of scale σ.
2. Estimate local edge normal directions $\mathbf{n}$ for each pixel in the image
3. Find the location of the edges using non-maximal suppression
4. Compute the magnitude of the edge
5. Threshold edges in the image with hysteresis to eliminate spurious responses.
6. Repeat steps 1) through 5) for ascending values of the standard deviation σ .
7. Aggregate the final information about edges at multiple scale using the 'feature synthesis' approach.

Experiments

Exercise

- Study the IP toolbox function edge
- Read the images *winchester.tiff* and *pentagon.tiff*
- Perform edge detection with
  - First derivative operators
  - Zero-crossing LoG operators
  - Canny operator with several values of σ.
  - Repeat for several levels of zero mean gaussian noise.

Line Detection

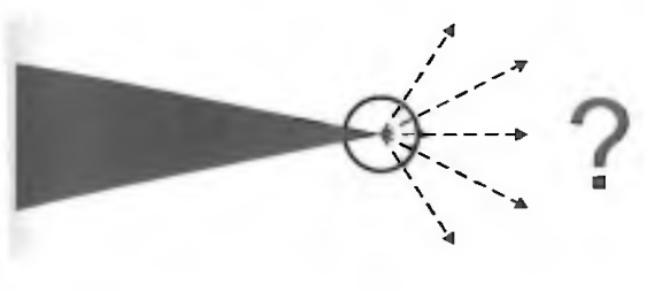
- Same rationale of detecting "roof" like profiles along "strategic" orientations: $0°, 45°, 90°, 135° \ldots$
- Convolution Kernels

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

- Lines detected this way are collections of edges. Most of the time non single pixel wide edges.
- Necessary to introduce line thining algorithms (ahead in this course)

Corner Detection

- Corners in images can be located using local detectors;
  - Input to the corner detector is the gray-level image
  - Output is the image in which values are proportional to the likelihood that the pixel is a corner. Interest points are obtained by thresholding the result of the corner detector.
- Edge detectors themselves are not stable at corners.
  - Gradient at the tip is ambiguous



Adapted from Image Processing , Analysis, and Machine Vision 3Ed, Sonka et al

  - The corner in the image can also be defined as a pixel in its small neighborhood where there are two dominant and different edge directions. (Not very precise definition)

Corner Detectors

- Moravec Detector:
  - Auto-Correlation based

$$\mathrm{MO}(i,j) = \frac{1}{8} \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} |f(k,l) - f(i,j)|$$

  - Compare the MO discriminant according to a predefined threshold
- Zuniga-Haralick:
  - Approximate the function $f(i,j)$ by a $2D$ cubic polynomial

$$f(i,j) = c_1 + c_2 x + c_3 y + c_4 x^2 + c_5 xy + c_6 y^2 + c_7 x^3 + c_8 x^2 y + c_9 xy^2 + c_{10} y^3$$

  - Consider the rate of change in gradient direction and build a discriminant $\mathrm{ZH}(i,j)$

$$\mathrm{ZH}(i,j) = \frac{-2(c_2^2 c_6 - c_2 c_3 c_5 + c_3^2 c_4)}{(c_2^2 + c_3^2)^{3/2}}$$

Harris corner detector

- Auto-correlation based
- Improvement upon Moravec's corner detector
- Use a sliding window W patch and estimate the sum of square differences of the discriminant function

$$S_w(\Delta x, \Delta y) = \sum_{x \in W} \sum_{y \in W} \left(f(x_i, y_j) - f(x_i - \Delta x, y_j - \Delta y)\right)^2$$

- A corner point must have a high response of $S_w(\Delta x, \Delta y)$ for all $\Delta x, \Delta y$.
- With a first order Taylor approximation near (i, j)

$$f(x_i - \Delta x, y_j - \Delta y) \approx f(x_i, y_j) + \left[\frac{\partial f(x_i, y_j)}{\partial x}, \frac{\partial f(x_i, y_j)}{\partial y}\right] \begin{bmatrix}\Delta x \\ \Delta y\end{bmatrix}$$

Harris corner detector

- After some algebra the analytical maximum of $S_w(\Delta x, \Delta y)$ is given by

$$[\Delta x, \Delta y]\, A_W(x_i, y_j) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

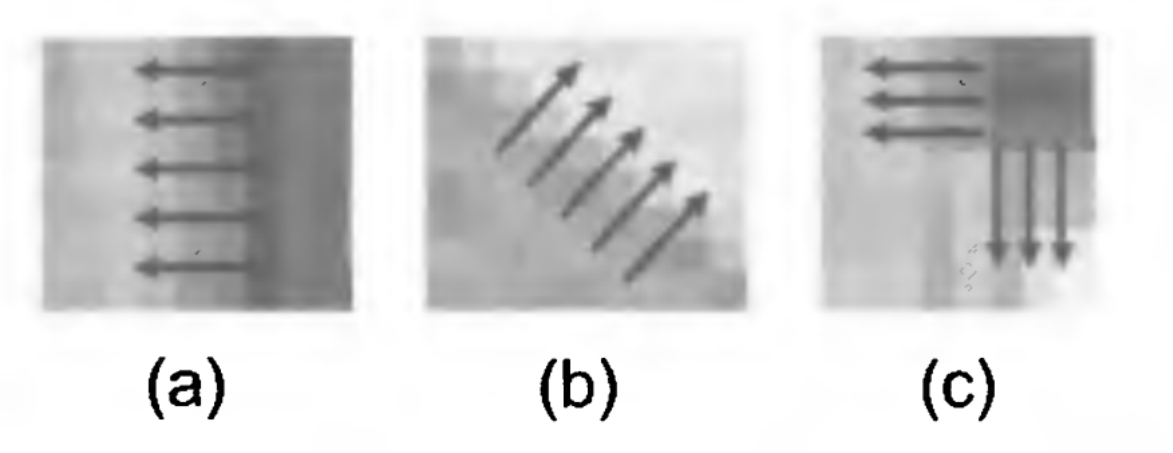
- Where $A_W(x_i, y_j)$ denotes the Harris matrix containing the second derivatives around the central point of S_w

$$A(x_i, y_j) = \begin{bmatrix} \sum\limits_{x_i, y_j \in W} \sum \frac{\partial^2 f(x_i, y_j)}{\partial x^2} & \sum\limits_{x_i, y_j \in W} \sum \frac{\partial f(x_i, y_j)}{\partial x} \frac{\partial f(x_i, y_j)}{\partial y} \\ \sum\limits_{x_i, y_j \in W} \sum \frac{\partial f(x_i, y_j)}{\partial x} \frac{\partial f(x_i, y_j)}{\partial y} & \sum\limits_{x_i, y_j \in W} \sum \frac{\partial^2 f(x_i, y_j)}{\partial y^2} \end{bmatrix}$$

  - Usually the derivatives are computed upon a Gaussian window
  - Matrix A is symmetric, positive semi-definite (non-negative eigenvalues)
  - Corner information is associated with the main variation modes of the Harris matrix

Eigen-values of Harris Matrix

- Relative weight of λ_1, λ_2,eigen-values correspond to the variation along the orthogonal directions
  - Both eigen-values are small. This means that image f is flat in the examined pixel, there are no edges or corners in this location.
  - One eigen-value is small and the second one large. The local neighborhood is ridge-shaped. Significant change of image f occurs if a small movement is made perpendicularly to the ridge.
  - Both eigen-values are rather large. A small shift in any direction causes significant change of image f. A corner is found.



Adapted from Image Processing , Analysis, and Machine Vision 3Ed, Sonka et al

Implementation issues

- Exact computation of eigen-values is not trivial
- Harris proposed a response function

$$R(A) = \det(A) - k.\mathrm{trace}^2(A)$$

- k is a heuristic coefficient, $k \in [0.04, 0.15]$

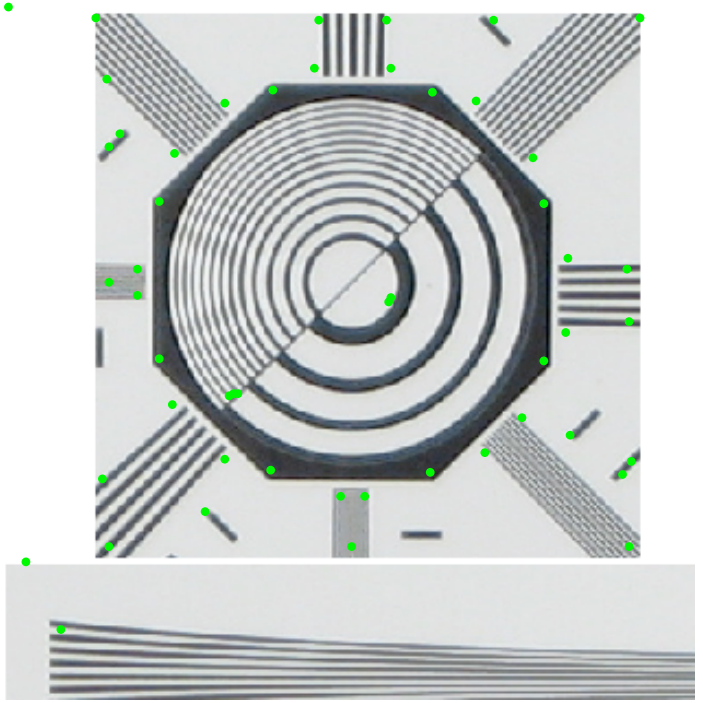
Algorithm

1. Filter the image with a Gaussian.
2. Estimate intensity gradient in two perpendicular directions for each pixel. Use successive $1D$ convolutions with the kernel approximating the derivatives
3. For each pixel and a given neighborhood window:
   1. Calculate the local structure matrix A
   2. Evaluate the response function $R(A)$.
   3. Choose the best candidates for corners by selecting a threshold on the response function $R(A)$ and perform non-maximal suppression.

Harris corner detection: remarks

- Advantages
  - Insensitivity
    - 2D shift and rotation
    - Small illumination variations
    - Small viewpoint change
  - Low computational requirements.
- Disadvantages
  - Not invariant to
    - large scale change
    - Large viewpoint changes
    - significant changes in contrast.

Harris corner detection: examples



Corners at image limits are not removed