Advanced Edge Detection[1]

Lecture 4

See Sections 2.4 and 1.2.5 in
Reinhard Klette: Concise Computer Vision
Springer-Verlag, London, 2014

[1] See last slide for copyright information.

Agenda

1. LoG and DoG
2. Phase-Congruency Model
3. Embedded Confidence

Laplacian of Gaussian (LoG) Edge Detector

Convolution theorem for LoG: $\nabla^2 (G * I) = I * \nabla^2 G$

Follows from applying twice the general rule of convolution:
$D(F * H) = D(F) * H = F * D(H)$ where D is a derivative

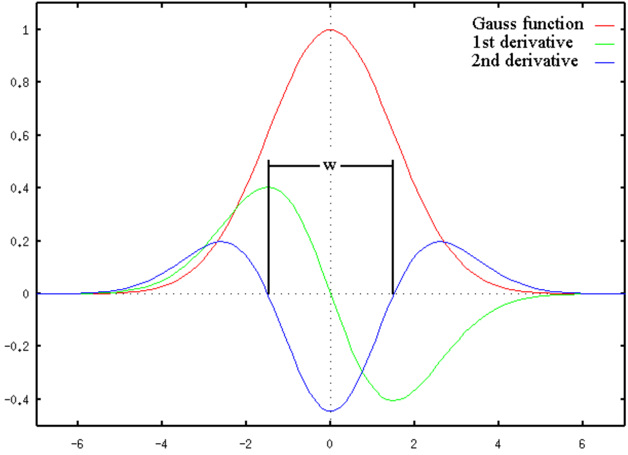
Conclusion: only one convolution needed with $\nabla^2 G$

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

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

$$\nabla^2 G(x, y) = \frac{1}{2\pi\sigma^4} \left(\frac{x^2+y^2-2\sigma^2}{\sigma^2}\right) e^{-\left(x^2+y^2\right)/2\sigma^2}$$

The Mexican Hat as Filter Kernel



$w = |x_1 - x_2| = 2\sqrt{2}\sigma$; sample a $3w \times 3w$ kernel

$\sigma = 1$, thus $3w = 8.485\ldots$, thus 9×9 (i.e., $k = 4$)

Difference of Gaussians (DoG)

Approximation of LoG for reduced run time

Scale $s = 2\sigma^2$

Level function

$$L(x, y, s) = [I * G_s](x, y)$$

DoG for initial scale s and scaling factor $k > 1$:

$$D_{s,k}(x, y) = L(x, y, s) - L(x, y, ks)$$

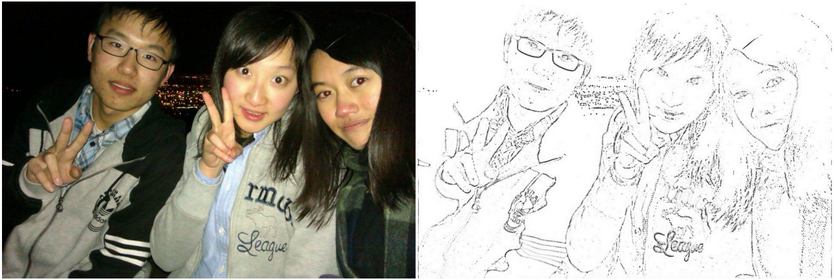
We have that $D_{s,1.4}(x, y) \approx [\nabla^2(G_s * I)](x, y)$

Edges detected by zero-crossings (as for the LoG)

Agenda

1. LoG and DoG
2. Phase-Congruency Model
3. Embedded Confidence

Kovesi Edge Map



Edge map resulting when applying the Kovesi algorithm

Recall: Magnitude and Phase of Complex Numbers

DFT maps real-valued images into a complex-valued Fourier transform

$z = a + \sqrt{-1}b$ defined in polar coordinates by

magnitude $||z||_2 = r = \sqrt{a^2 + b^2}$ and phase $\alpha = \mathrm{arctan}(b/a)$

Local Fourier Transform

$p = (x, y)$ in image I and $(2k+1) \times (2k+1)$ filter kernel

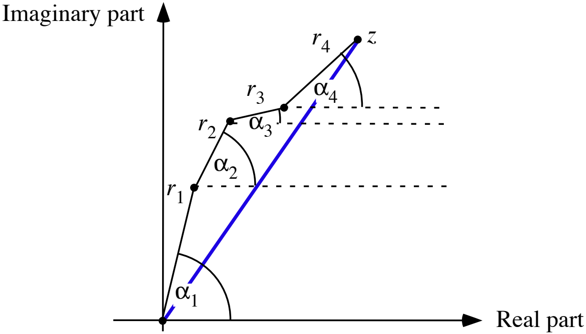
$$\mathbf{J}(u, v) = \frac{1}{(2k+1)^2} \sum_{i=0}^{2k} \sum_{j=0}^{2k} I(x+i, y+j) \cdot W_{2k+1}^{-iu} \cdot W_{N_{rows}}^{-yv}$$

$0 \leq u, v \leq 2k+1$

Vector Sum of Complex Numbers

J composed of $n = (2k+1)^2$ complex numbers z_h, for $1 \leq h \leq n$

Each z_h defined by $r_h = ||z_h||_2$ and α_h



Addition of four complex numbers $z_h = (r_h, \alpha_h)$

The Phase-Congruency Model for Edges

$$0 \leq \mathcal{C}_{phase}(p) = \frac{||z||_2}{\sum_{h=1}^{n} r_h} \leq 1$$

$\mathcal{C}_{phase}(p) = 1$ defines *perfect congruency*

$\mathcal{C}_{phase}(p) = 0$ for perfectly opposing phases and magnitudes

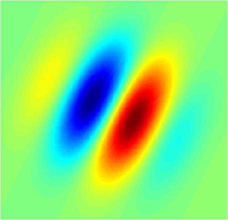
Local phase congruency identifies features in images

High phase congruency at adjacent pixels defines *edges*

Kovesi Algorithm

1. Apply n Gabor-filter functions for an approximate local DFT
2. Quantify phase congruency for resulting (r_h, α_h), $1 \leq h \leq n$ also incorporating noise compensation
3. If phase congruency $\geq T$ then edge pixel

Only one threshold parameter T if set of Gabor functions is fixed



Example of a Gabor function (also known as Gabor wavelet)

Agenda

1. LoG and DoG
2. Phase-Congruency Model
3. Embedded Confidence

Confidence Measure for a Feature Detector

A *confidence measure* is

quantified information derived from calculated data,

to be used for deciding about the existence of a particular feature;

if calculated data match the underlying model of the feature detector reasonably well

then this should correspond to high values of the confidence measure.

Meer-Georgescu Algorithm

for every pixel p in image I **do**
    estimate gradient magnitude $g(p)$ and edge direction $\theta(p)$;
    compute the confidence measure $\eta(p)$;
end for
for every pixel p in image I **do**
    determine value $\rho(p)$ in the cumulative distribution of gradient magnitudes;
end for
generate the $\rho\eta$ diagram for mage I;
perform non-maxima suppression;
perform hysteresis thresholding;

Four parameters: gradient magnitude $g(p) = ||\mathbf{g}(p)||_2$, gradient direction $\theta(p)$, edge confidence $\eta(p)$, percentile ρ_k of cumulative gradient magnitude distribution

Gradient Magnitude and Gradient Direction Estimators

Trace $\mathrm{Tr}(\mathbf{A})$ of $n \times n$ matrix $\mathbf{A} = (a_{ij})_{ij}$
is the sum $\sum_{i=1}^{b} a_{ii}$ of (main) diagonal elements

$\mathbf{A}$ a matrix representation of $(2k+1) \times (2k+1)$ window $W_p(I)$

$\mathbf{W}$ a row-/column-symmetric $(2k+1) \times (2k+1)$ matrix of weights

$d_1 = \mathrm{Tr}(\mathbf{WA})$ and $d_2 = \mathrm{Tr}(\mathbf{AW}^\top)$

$$g(p) = \sqrt{d_1^2 + d_2^2} \quad \text{and} \quad \theta(p) = \arctan\left(\frac{d_1}{d_2}\right)$$

Note: not in x- and y-direction, but 45° rotated directions; but we could also use estimators as defined earlier

Percentile and Confidence

Gradient-magnitudes $g_{[1]} < \ldots < g_{[k]} < \ldots < g_{[N]}$ in image I with

$$\rho_k = Prob\left[g \leq g_{[k]}\right]$$

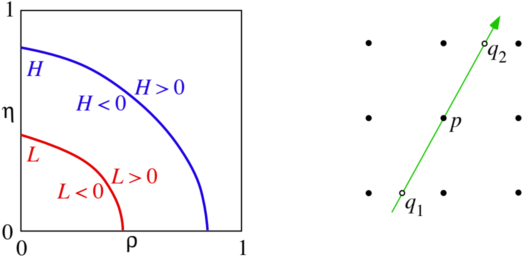
for $1 \leq k \leq N$. If $g_{[k]}$ the closest real to edge magnitude $g(p)$ then

Percentile: $\rho(p) = \rho_k$ between 0 and 1

Let $\mathbf{A}_{ideal}$ be a $(2k+1) \times (2k+1)$ matrix representing a template of an ideal step edge having gradient direction $\theta(p)$.

Confidence: $\eta(p) = |Tr(\mathbf{A}_{ideal}^{\top}\mathbf{A})|$ between 0 and 1

$\rho\eta$ Diagram



Left: Implicitly given curves $L(\rho, \eta) = 0$ and $H(\rho, \eta) = 0$ separate square into points with positive or negative signs.

Right: Virtual neighbors q_1 and q_2 in estimated gradient direction

Non-Maxima Suppression

Define a curve $X(\rho, \eta) = 0$ in $\rho\eta$ space

1. For current pixel p, determine virtual neighbors q_1 and q_2
2. Determine ρ and η values for q_1 and q_2 by interpolation of values at adjacent pixel locations
3. p describes with respect to X a *maximum* if both virtual neighbors q_1 and q_2 have a negative sign for curve X

Non-maxima suppression in Step 9:
only remaining pixels are candidates for the edge map.

Hysteresis Thresholding

Hysteresis thresholding is a general technique to decide in a process based on previously obtained results, attempting to continue as before.

Have two *hysteresis thresholds* L and H in $\rho\eta$ space.
Pixel p with values ρ and η stays on the edge map if

1. $L(\rho, \eta) > 0$ and $H(\rho, \eta) \geq 0$ or
2. p is adjacent to a pixel in the edge map and satisfies $L(\rho, \eta) \cdot H(\rho, \eta) < 0$

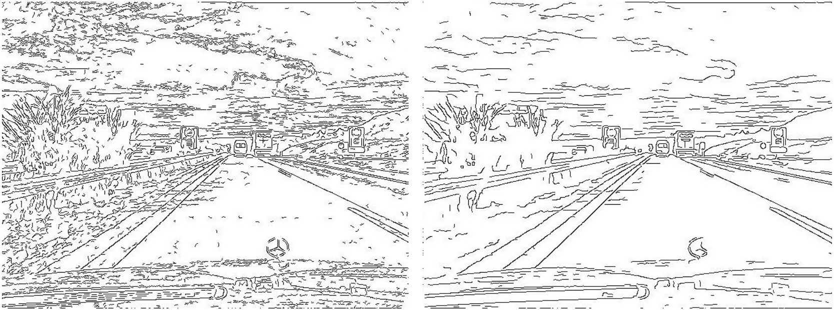
Second condition describes hysteresis thresholding; it is applied recursively.

Options for the Meer-Georgescu Algorithm

The Meer-Georgescu algorithm can be a

1. Canny edge detector of the gradient magnitudes if the two hysteresis thresholds are vertical lines, and
2. a confidence only detector if the two lines are horizontal

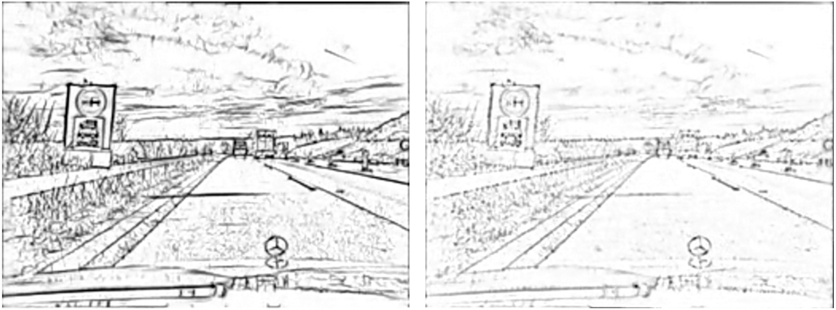
Results



Results of the Meer-Georgescu algorithm with a larger (*left*) or a smaller (*right*) filter kernel

Original Image `Set1Seq1`

Kovesi Edge Detector



Results of the Kovesi algorithm using different thresholds T

Sobel and Canny Edge Detector

Results of Sobel detector (*left*) and of Canny operator (*right*)

Adaptation, and Model Diversity versus Simplicity

There is no “best edge detector”, it all depends on the application context.

Adaptation is often the word - operator and parameters need to be adjusted to the given data.

Design ideas such as step-edge or phase-congruency model, combination of 1st and 2nd order derivatives in the first case, accumulated evidence based on results at adjacent pixels (hysteresis thresholding), confidence measures, thinning of resulting edges (non-maxima suppression) can be used for going towards adapted solutions.

A simple method such as the Sobel operator is still often useful because it does not modify data at this early processing stage based on some model which might not be true in general.

Copyright Information

This slide show was prepared by Reinhard Klette
with kind permission from Springer Science+Business Media B.V.

The slide show can be used freely for presentations.
However, *all the material* is copyrighted.

In case of citation: just cite the book, that's fine.