Computer Vision

James L. Crowley

M2R MoSIG option GVR

Fall Semester
15 October 2009

Lesson 4

Contrast Description With Edge Features

Lesson Outline:

1 Describing Contrast (Continued) 2
1.1 Difference Operators: Derivatives for Sampled Signals 2
1.2 Smoothing: The Binomial Low pass filter. 3
1.3 Edge Detection using integer coefficient filters 5
1.4 Non-maximum suppression 6
2 Hough Transform 7
2.1 Generalisation of the Hough Transform 8
3 Second Derivatives 9
3.1 Integer Coefficient Second Derivatives 10
3.2 Zero Crossings in the second derivative. 12
4 Image Description Using Gaussian Derivatives 13
4.1 Gaussian Derivatives Operators 13

1 Describing Contrast (Continued)

1.1 Difference Operators: Derivatives for Sampled Signals

For the function, s(x) the derivative can be defined as :

$$\frac{\partial s(x)}{\partial x} = \lim_{\Delta x \to 0}\left\{\frac{s(x+\Delta x) - s(x-\Delta x)}{\Delta x}\right\}$$

For a sampled signal, s(n), an the equivalent is $\frac{\Delta s(n)}{\Delta n}$

the limit does not exist, however we can observe

$$\Delta n = 1: \quad \frac{\Delta s(n)}{\Delta n} = \frac{s(n+1) - s(n-1)}{1} = s(n) * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

This is the operator used by Sobel.

Thus we can define the first "difference" operator as a first order approximation for the derivative of a discrete signal.

$$\Delta_i p(i,j) = \Delta p(i,j)/\Delta i = p(i,j) * [-1, 0, 1]$$

$$\Delta_i p(i,j) = \frac{\Delta p(i,j)}{\Delta i} = p(i,j) * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

$$\Delta_j p(i,j) = \frac{\Delta p(i,j)}{\Delta j} = p(i,j) * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

$$\nabla P(i,j) = \begin{pmatrix} \Delta_i p(i,j) \\ \Delta_j p(i,j) \end{pmatrix}$$

$$E(i,j) = \left\| \nabla P(i,j) \right\|$$

$$\vartheta(i,j) = Tan^{-1}\left(\frac{\Delta_j p(i,j)}{\Delta_i p(i,j)}\right)$$

This works fine, except that such a derivative operator amplifies sampling noise.

1.2 Smoothing: The Binomial Low pass filter.

Sobel uses a filter [1, 2, 1] to smooth. This is also an optimal filter.
It is part of a family of filters generated by the binomial series.

The binomial series is the series of coefficients of the polynomial:

$$(x + y)^n = \sum_{m=0}^{n} b_{m,n} x^{n-m} y^m$$

The coefficients can be computed as $b_{m,n} = b_n(m) = [1, 1]^n$

These are the coefficients of Pascal's Triangle.

Les coefficients du suite binomial sont générés par le triangle de Pascal :

n	sum = 2^n	$\mu = n/2$	$\sigma^2 = n/4$	$\sigma = \sqrt{n/2}$	Coefficients
0	1	0	0	0	1
1	2	0.5	0.25		1 1
2	4	1	0/5		1 2 1
3	8	1.5	0.75		1 3 3 1
4	16	2	1	1	1 4 6 4 1
5	32	2.5	1.25		1 5 10 10 5 1
6	64	3	1.5		1 6 15 20 15 1
7	128	3.5	1.75		1 7 21 35 35 21 7 1
8	256	4	2	$\sqrt{2}$	1 8 29 56 70 56 29 8 1

These coefficients provide a family of low pass filters with remarkable properties.
Notably, these are the best approximation for a Gaussian filter of finite extent.
They also happen to have integer coefficients.

$$b_n(m) = b_1(m)^{*n} = [1, 1]^{*n} = \text{n convolutions of } [1, 1]$$

Gain : $$s_n = \sum_{m=1}^{n} b_n(m) = 2^n$$

Center of gravity is

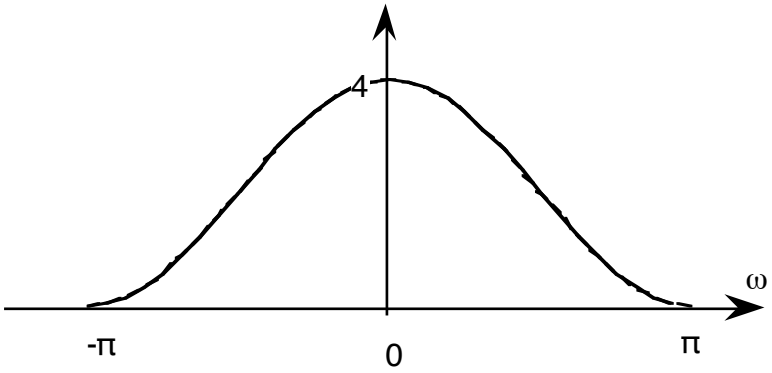
$$\mu_n = \frac{1}{s_n} \sum_{m=1}^{n} b_n(m) \cdot m = \frac{n}{2}$$

The variance is:

$$\sigma_n^2 = \frac{1}{s_n}\sum_{m=1}^{n} b_n(m)\cdot(m-\mu)^2 = \frac{n}{4}$$

The Fourier transform for $b_2(m) = [1, 2, 1]$ is

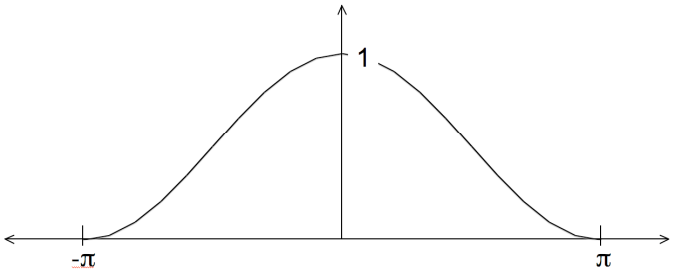
$$B_2(\omega) = \sum_{m=-1}^{1} b_2(m)e^{-j\omega m}$$
$$B_2(\omega) = 1e^{-j\omega(-1)} + 2e^{-j\omega 0} + 1e^{-j\omega(1)}$$
$$B_2(\omega) = 2 + e^{j\omega} + e^{-j\omega}$$
$$B_2(\omega) = 2 + 2\cos(\omega)$$



If we normalize the gain: $b_2(m) = (1/4)[1, 2, 1]$

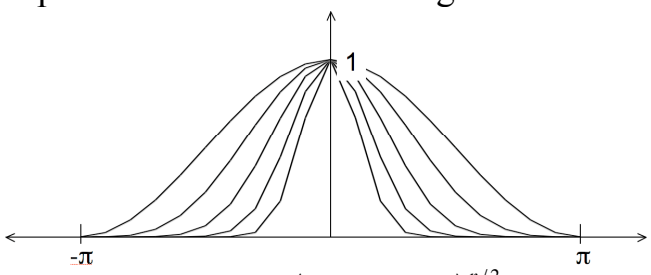
$$B_2(\omega) = \frac{1}{2} + \frac{1}{2}\cos(\omega)$$

Which is a cosine on a platform



$$B_2(\omega) = \frac{1}{2} + \frac{1}{2}\cos(\omega)$$

Repeated convolution makes generates



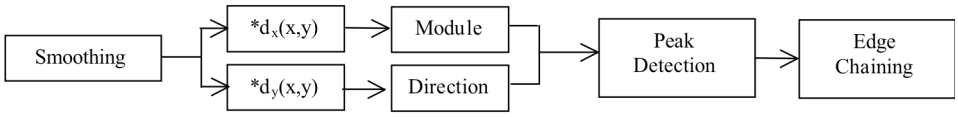
$$B_n(\omega) = \left(\frac{1}{2} + \frac{1}{2}\cos(\omega)\right)^{n/2}$$

The binomial coefficients provide a series of low pass filters with no ripples.

In 2D, the filters provide separable filters that are nearly circularly symmetric

$$\text{2-D} \quad b2(i,j) = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

1.3 Edge Detection using integer coefficient filters



$$\nabla P(i,j) = \begin{bmatrix} \frac{\Delta p(i,j)}{\Delta i} \\ \frac{\Delta p(i,j)}{\Delta j} \end{bmatrix} = \begin{bmatrix} m_{1*}p(i,j) \\ m_{2*}p(i,j) \end{bmatrix} = \begin{bmatrix} E_1(i,j) \\ E_2(i,j) \end{bmatrix}$$

Gradient:

$$E(i,j) = \left\| \vec{E}(i,j) \right\| = \sqrt{E_1(i,j)^2 + E_2(i,j)^2}$$

Direction of maximum contrast

$$\varphi(i,j) = Tan^{-1}\left(\frac{E_2(i,j)}{E_1(i,j)} \right)$$

Steps:
1) Smoothing - Suppress high frequency noise
2) Gradient - Compute first derivatives in row and column
3) Detection - Non-maximum suppression with double threshold
4) Chaining - Assembly of connected points above threshold. Elimination of chains where one of the points is not above a second threshold.
5) Polygonal approximation (multiple algorithms exist).

1.4 Non-maximum suppression.

Contrast points are local maxima in E(i, j).

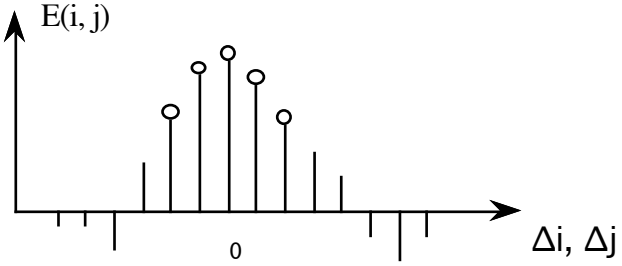
les points de contraste : C(i, j)
pour le gradient de la magnitude E(i, j) et orientation Φ(i, j)

For each point :

1) Determine the direction of maximum gradient:

$$\Delta i = \frac{\Delta_i P(i,j)}{\|\nabla P(i,j)\|} \qquad \Delta j = \frac{\Delta j P(i,j)}{\|\nabla P(i,j)\|}$$

2) Compare the gradient to its neighbors in this direction.



$$c(i,j) = \begin{cases} E(\mathrm{i},\mathrm{j}) \text{ if } \mathrm{E}(\mathrm{i}-\Delta\mathrm{i},\mathrm{j}-\Delta\mathrm{j}) \le \mathrm{E}(\mathrm{i},\mathrm{j}) \ge \mathrm{E}(\mathrm{i}+\Delta\mathrm{i},\mathrm{j}+\Delta\mathrm{j}) \\ 0 \text{ Otherwise} \end{cases}$$

Construct a list of connected points for which E(i,j) ≠ 0.

Techniques:

1) Line scan edge chaining algorithm
2) Edge following
3) Hough Transform

2 Hough Transform

The Hough transform is an "optimal" statistical detector for estimating parametric functions from discrete samples. This method was invented for interpreting bubble chamber images in particle physics. It is based on "voting" for possible parameters.

This transform was invented by
P.V.C. Hough, Machine Analysis of Bubble Chamber Pictures, Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

It was patented in a crude form by IBM in 1962 using $y = mx+c$.

It was made popular by Duda and Hart :
Duda, R. O. and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," Comm. ACM, Vol. 15, pp. 11–15 (January, 1972)

Consider the line equation

$$x \cos(\theta) + y \sin(\theta) + c = 0$$

In the image, for each x,y (free parameters) we need to determine (c, θ)

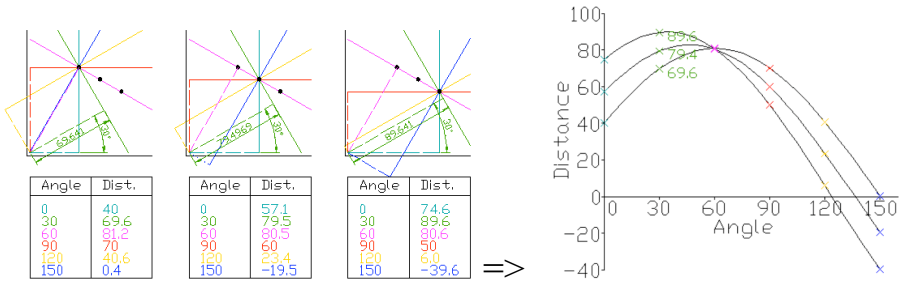
In the Hough transform, we will create a dual space in which (c, θ) are free parameters.
We will estimate lines as peaks in this dual space. To find peaks we build an accumulator array : $h(c, \theta)$.

Let the c be an integer $c \in [0, D]$ where D is the "diagonal distance of the image.
Let θ be an integer $\theta \in [0, 179]$

Algorithm:

```
allocate a table  h(c, θ) initially set to 0.
For each x, y of the image
        for  θ from 0 to 179
                c = –x cos(θ) – ysin(θ)
                h(c, θ) = h(c, θ) + E(x, y)
        End
End
```

The resulting table accumulates contrast.
Peaks in $h(c, \theta)$ correspond to line segments in the image.



Because we know θ(x, y), we can limit the evaluation to θ(x, y)+/- Δθ

2.1 Generalisation of the Hough Transform

We can represent a circle with the equation:

$$(x - a)^2 + (y - b)^2 = r^2$$

We can use this to create a Hough space h(a, b, r) for limited ranges of r.

The ranges of a and b are the possible positions of circles.

Algorithm

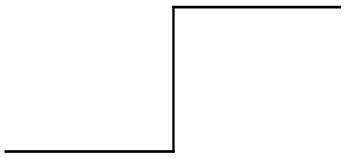
Algorithm:

allocate a table h((a, b, r) initially set to 0.
For each x, y of the image
for r from r_{min} to r_{max}
for a from 0 to a_{max}
b = –y–sqrt(r^2 – (x - a)^2)
h(a,b,r) = h(a,b,r) + E(x,y).
End
End
End

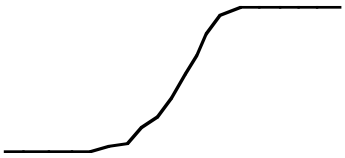
3 Second Derivatives.

An alternative to the gradient is to detect edges as zero crossings in the second derivative.

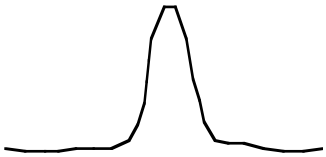
Contrast :



Smoothing :



1st derivative



Second derivative

3.1 Integer Coefficient Second Derivatives

The second derivative is a form of Laplacian operator:

$$\text{Laplacien: } \nabla^2 p(i,j) = \frac{\Delta^2 P(i,j)}{\Delta i^2} + \frac{\Delta^2 P(i,j)}{\Delta j^2}$$

$$\Delta_i^2 = [\ 1\ \ -1] * [1\ \ -1] = [\ -1\ \ 2\ \ -1]$$

$$\Delta_i^2 = \boxed{-1\ 2\ -1} \qquad \Delta_j^2 = \boxed{\begin{matrix} -1 \\ 2 \\ -1 \end{matrix}}$$

$$\text{Laplacian : } \nabla^2 p(i,j) = \Delta_i^2 * p(i,j) + \Delta_j^2 * p(i,j)$$

There are several possible discrete forms:

$$\boxed{\begin{matrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{matrix}} = \boxed{\begin{matrix} -1 \\ 2 \\ -1 \end{matrix}} + \boxed{-1\ 2\ -1}$$

$L(u,v) = 4 - 2\cos(u) - 2\cos(v)$
The best is :

$$\boxed{\begin{matrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{matrix}} + 2 \boxed{\begin{matrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{matrix}} = \boxed{\begin{matrix} -1 & -2 & -1 \\ -2 & 12 & -2 \\ -1 & -2 & -1 \end{matrix}}$$

$$\text{Gradient : } \| \nabla\ p(i,j) \| = \sqrt{\left(\frac{\partial P(i,j)}{\partial i}\right)2 + \left(\frac{\partial P(i,j)}{\partial j}\right)2}$$

Laplacien : $$\nabla^2 p(i,j) = \frac{\partial^2 P(i,j)}{\partial i^2} + \frac{\partial^2 P(i,j)}{\partial j^2}$$

3.2 Zero Crossings in the second derivative.

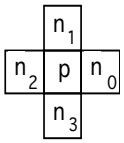
In theory

1) Zero crossings give closed contours
2) Zero crossings can be easily interpolated for high precision.

In practice

Zero crossings detect many small unstable contours.

Neighborhood test:



$$C(i,j) = \begin{cases} 1 & if(sign(n_0) \neq sign(n_2)) \; and \; |n_0 n_2 \rhd 0| \\ & or \text{ if } (signe(n_1) \neq signe(n_3)) \; and \; |n_1 n_3 \rhd 0 \\ 0 & \text{Otherwise} \end{cases}$$

alternatively :

$$C(i,j) = \begin{cases} 1 & \text{if } (\text{sign}(n_0) \neq \text{sign}(n_2)) \\ & \quad \text{and } |n_0 \; n_2| > 0| \\ & \quad \text{and } |n_0 - n_2| > \text{Threshold} \\ & \text{or if } (\text{sign}(n_1) \neq \text{sign}(n_3)) \\ & \quad \text{and } |n_1 n_3 \rhd 0 \\ & \quad \text{and } |n_1 - n_3 \rhd \text{Threshold} \\ 0 & \text{Otherwise} \end{cases}$$

4 Image Description Using Gaussian Derivatives

4.1 Gaussian Derivatives Operators

The Gaussian Function is $G(x,\sigma) = e^{-\frac{x^2}{2\sigma^2}}$

The Gaussian function is invariant to affine transformations.

$$T_a\{G(x, \sigma)\} = G(T_a\{x\}, T_a\{\sigma\})$$

Recall from lesson 2 we saw that $x_r = x_c \frac{F}{z_c}$

The apparent size of an object is inversely proportional to its distance

A change in size (or scale) is a special case of an affine transform:

$$T_S\{G(x, \sigma)\} = G(T_S\{x\}, T_S\{\sigma\}) = G(sx, s\sigma)$$

This is just one of the many interesting properties of the Gaussian function.