

Formation et Analyse d'Images

James L. Crowley

ENSIMAG 3

Premier Sémestre 2009/2010

Lesson 7

30 novembre 2009

Describing Local Appearance with Gaussian Derivatives and Scale Space

Lesson Outline:

1	Describing Local Appearance.....	2
2	Gaussian Derivative Operators	3
2.1	The Sampled Gaussian Functions.....	3
2.2	Gaussian Digital Filters	4
2.3	Properties of Gaussian Receptive Fields (or Wavelets).....	5
2.4	Using the Gaussian to compute image derivatives	8
2.5	Steerability of Gaussian Derivatives.....	8
3	Gaussian Scale Space	10
3.1	Image Scale Space:.....	10
3.2	Discrete Scale Space	11

1 Describing Local Appearance

Appearance is what you see.

We seek a set of K local basis functions, $d_k(x,y)$ to describe "appearance" around a point in an image. Each basis function, $d_k(x,y)$ gives an image "feature", x_k , describing appearance at the pixel (x_o, y_o) .

$$x_k = \sum_{x=-R}^R \sum_{y=-R}^R p(x_o + x, y_o + y) d_k(x, y)$$

Projection of the image neighborhood $P(x_o, y_o)$ onto this set of functions gives a

"feature" vector for appearance, $\vec{X} = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_K \end{pmatrix}$

Ideally these functions should be orthogonal

$$\sum_{x=-R}^R \sum_{y=-R}^R d_m(x_o + x, y_o + y) d_n(x, y) = 0 \quad \text{if } n \neq m$$

to minimize the number of features. This is not strictly necessary.

Such a basis can be obtained by a Taylor series representation of the image function $P(x)$ around a point, a :

$$f(x) = f(a) + \frac{1}{1!} f_x(x-a) + \frac{1}{2!} f_{xx}(x-a)^2 + \frac{1}{3!} f_{xxx}(x-a)^3 + \dots$$

The basis set for a Taylor series is the series of local derivatives.

This is called the "local jet". The local jet is a basis for describing appearance.

Because the image is a sampled signal, we need a kernel filter to compute the local Jet. The Gaussian function provides a Kernel with many useful properties.

2 Gaussian Derivative Operators

2.1 The Sampled Gaussian Functions

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

Fourier Transform: $F\{e^{-\frac{x^2}{2\sigma^2}}\} = G(\omega, \sigma) = \sqrt{2\pi} \sigma e^{-\frac{1}{2}\sigma^2\omega^2}$

2D Gaussian Kernel: $G(x, y, \sigma) = e^{-\frac{(x^2+y^2)}{2\sigma^2}} = e^{-\frac{x^2}{2\sigma^2}} * e^{-\frac{y^2}{2\sigma^2}}$

Fourier Transform: $F\{e^{-\frac{x^2+y^2}{2\sigma^2}}\} = \frac{\pi}{2\sigma^2} e^{-\frac{1}{2}\sigma^2(u^2+v^2)}$

The sampled Gaussian and its derivatives are:

$$G(n, \sigma) = e^{-\frac{n^2}{2\sigma^2}}$$

$$G_x(n, \sigma) = -\frac{n}{\sigma^2} G(x, \sigma) = -\frac{n}{\sigma^2} e^{-\frac{n^2}{2\sigma^2}}$$

$$G_{xx}(n, \sigma) = \frac{n^2 - \sigma^2}{\sigma^4} G(n, \sigma) = \frac{n^2 - \sigma^2}{\sigma^4} e^{-\frac{n^2}{2\sigma^2}}$$

$$G_{xxx}(n, \sigma) = -\frac{n^3 - n\sigma^2}{\sigma^6} G(n, \sigma) = -\frac{n^3 - n\sigma^2}{\sigma^6} e^{-\frac{n^2}{2\sigma^2}}$$

For the 2D Sampled Gaussian $G(i, j, \sigma) = e^{-\frac{i^2+j^2}{2\sigma^2}}$

(Attention: i, j are INTEGERS, not complex numbers!)

2D Gaussian is separable: $G(i, j, \sigma) = \frac{1}{A} e^{-\frac{(i^2+j^2)}{2\sigma^2}} = e^{-\frac{i^2}{2\sigma^2}} * e^{-\frac{j^2}{2\sigma^2}}$

The normalization factor $A = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} e^{-\frac{(i^2+j^2)}{2\sigma^2}} \approx 2\pi\sigma$

This factor will be slightly smaller if the Gaussian is evaluated over a finite window $W_N(i, j)$ of size $N \times N$ pixels.

This factor can be determined from the filter, and can be ignored in discussing the properties of the Gaussian.

2.2 Gaussian Digital Filters

Computers represent image as 2D sampled digitized signals. Because they are sampled, processing requires convolution with a sampled filter.

To obtain a 2D digital Gaussian filter we must sample the function at a rate of Δx , and Δy over some finite support of radius R .

$$G(x, \sigma)$$

Because we can control scale with σ , we can set the sample size to $\Delta x = \Delta y = 1$. This gives a sampled function

$$G(n, \sigma) = e^{-\frac{n^2}{2\sigma^2}}$$

To represent this in a computer we must also specify the finite support (number of samples), N .

We set $N = 2R + 1$.

This gives us 2 parameters to control:

- 1) The scale of the Gaussian $\sigma/\Delta x$
- 2) the size of the support $N = 2R+1$

Truncating a function to a finite support is equivalent to multiply by a window $W_N(n)$

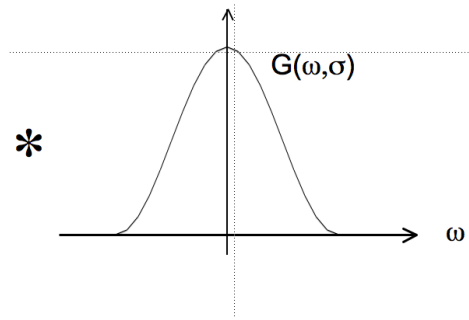
When we limit $G(x, \sigma)$ to a finite support, we multiply by a window

$$G(n, \sigma) = G(n, \sigma) \cdot w_N(n) \text{ where } w_N(n) = \begin{cases} 1 & \text{for } -R \leq n \leq R \\ 0 & \text{otherwise} \end{cases}$$

Multiplying by a finite window is equivalent to convolving with the Fourier transform of the finite window:

$$F\{G(n, \sigma) \cdot w_N(n)\} = G(\omega, \sigma) * W_N(\omega)$$

$$\text{where } W_N(\omega) = \frac{\sin(\omega N/2)}{\sin(\omega/2)} \quad \text{and} \quad G(\omega, \sigma) = \sqrt{2\pi} \sigma e^{-\frac{1}{2}\sigma^2\omega^2}$$



For $N < 7$, the ripples in $WN(\omega)$ dominate the spectrum and corrupt the resulting Gaussian.

At $N=7$ the effect is tolerable but significant.

At $N \geq 9$ the effect becomes minimal

In addition for $\sigma/\Delta x < 1$, the phenomenon of aliasing folds a significant amount of energy at the nyquist frequency, corrupting the quality (and the invariance) of the Gaussian function.

Finally, it is necessary to assure that the "gain" of the Gaussian filter is 1. This can be assured by normalizing so that the sum of the coefficients is 1. If the Gaussian were infinite in extent, then

$$\sum_{x=-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}} = \sqrt{2\pi}\sigma$$

However, because we truncate the Gaussian to an extent $n \pm R$, we must calculate the sum of the coefficients, A:

$$A = \sum_{n=-R}^R e^{-\frac{n^2}{2\sigma^2}}$$

The Gaussian filter is thus normalized by dividing by A to give a unit gain Receptive Field.

$$G(i, j, \sigma) = \frac{1}{A} e^{-\frac{(i^2 + j^2)}{2\sigma^2}}$$

2.3 Properties of Gaussian Receptive Fields (or Wavelets)

Separability:

Continuous Gaussian Kernel: $G(x, y, \sigma) = e^{-\frac{(x^2 + y^2)}{2\sigma^2}} = e^{-\frac{x^2}{2\sigma^2}} * e^{-\frac{y^2}{2\sigma^2}}$

Sampled Gaussian: $W_N(i, j) \cdot G(i, j, \sigma) = W_N(i, j) \cdot e^{-\frac{(j^2 + j^2)}{2\sigma^2}} = (W_N(i) \cdot e^{-\frac{i^2}{2\sigma^2}}) * (W_N(j) \cdot e^{-\frac{j^2}{2\sigma^2}})$

Scale property: $G(x, y, \sqrt{2}\sigma) = G(x, y, \sigma) * G(x, y, \sigma)$

Discrete form: $G(i, j, \sqrt{2}\sigma) = G(i, j, \sigma) * G(i, j, \sigma)$

Derivative Filters:

$$G_x(i, j, \sigma) = -\frac{i}{\sigma^2} G(i, j, \sigma)$$

$$G_{xx}(i, j, \sigma) = \frac{i - \sigma^2}{\sigma^4} G(i, j, \sigma)$$

$$G_{xy}(i, j, \sigma) = \frac{ij}{\sigma^4} G(i, j, \sigma)$$

$$G_{xxx}(i, j, \sigma) = -\frac{i^3 - i\sigma^2}{\sigma^6} G(i, j, \sigma)$$

$$\nabla^2 G_x(i, j, \sigma) = G_{xx}(i, j, \sigma) + G_{yy}(i, j, \sigma)$$

Diffusion Equation: $\nabla^2 G_x(i, j, \sigma) = G_{xx}(i, j, \sigma) + G_{yy}(i, j, \sigma) = \frac{\partial G(i, j, \sigma)}{\partial \sigma}$

As a consequence: $\nabla^2 G(i, j, \sigma) \approx G(i, j, \sigma_1) - G(i, j, \sigma_2)$

This typically requires $\sigma_1 \geq 1.4 \sigma_2$

Thus it is common to use:

$$\nabla^2 G(x, y, \sigma) \approx G(x, y, \sqrt{2}\sigma) - G(x, y, \sigma)$$

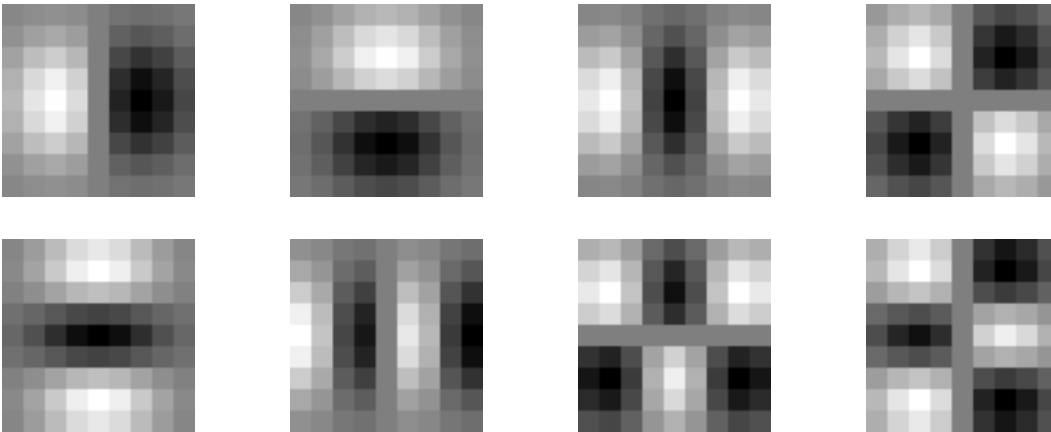
But note that :

$$G(x, y, \sqrt{2}\sigma) \approx G(x, y, \sigma) * G(x, y, \sigma)$$

so that $\nabla^2 G(x, y, \sigma) \approx G(x, y, \sigma) * G(x, y, \sigma) - G(x, y, \sigma)$

We can use these sampled functions to create a basis set of receptive fields:

$$\vec{G}_a = (G_x, G_y, G_{xx}, G_{xy}, G_{yy}, G_{xxx}, G_{xxy}, G_{xyy}, G_{yyy})$$



The Gaussian receptive fields $G_x, G_y, G_{xx}, G_{xy}, G_{yy}, G_{xxx}, G_{xxy}, G_{xyy}, G_{yyy}$.

Note that there is only one parameter: σ . This determines the limit of the resolution for the position of a contrast point.

These can be used to compute a local description known as the "local jet".

$$\vec{L}(i, j, \sigma) = P * \vec{G}(i, j, \sigma)$$

The Local jet can be steered to a normalization angle θ .

Steerability:

$$G_\theta^1(x, y, \sigma) = \cos(\theta) \cdot G_x(x, y, \sigma) + \sin(\theta) \cdot G_y(x, y, \sigma)$$

$$G_\theta^2(x, y, \sigma) = \cos(\theta)^2 G_{xx}(x, y, \sigma) + \cos(\theta)\sin(\theta) G_{xy}(x, y, \sigma) + \sin(\theta)^2 G_{yy}(x, y, \sigma)$$

$$G_\theta^3(x, y, \sigma) = \cos(\theta)^3 G_{xxx}(x, y, \sigma) + \cos(\theta)^2 \sin(\theta) G_{xxy}(x, y, \sigma) + \cos(\theta)\sin(\theta)^2 G_{xyy}(x, y, \sigma) + \sin(\theta)^3 G_{yyy}(x, y, \sigma)$$

Note the scale parameter σ determines the "resolution" of the derivatives.

You MUST specify σ . The smallest σ is not always the best.

Many computer vision algorithms give unpredictable results because the researchers forget to specify the scale σ at which the algorithm was validated.

2.4 Using the Gaussian to compute image derivatives

Consider an image: $P(i, j)$,

The Gradient $\vec{\nabla}P(i, j)$ is calculated by $P(i, j) * \vec{\nabla}G(i, j, \sigma)$

where $\vec{\nabla}G(i, j, \sigma) = \begin{pmatrix} G_x(i, j, \sigma) \\ G_y(i, j, \sigma) \end{pmatrix}$

Gradient: $\vec{\nabla}P(i, j) \approx \vec{\nabla}(P * G(i, j, \sigma)) = P * \vec{\nabla}G(i, j, \sigma) = \begin{pmatrix} P * G_x(i, j, \sigma) \\ P * G_y(i, j, \sigma) \end{pmatrix}$

Laplacien: $\nabla^2 P(i, j) = P * \nabla^2 G(i, j, \sigma) = \begin{pmatrix} P * \nabla^2 G(i, j, \sigma) \\ P * \nabla^2 G(i, j, \sigma) \end{pmatrix} \approx P * \nabla^2 G(i, j, \sigma_1) - P * \nabla^2 G(i, j, \sigma_2)$

where typically $\frac{\sigma_1}{\sigma_2} \geq \sqrt{2}$

2.5 Steerability of Gaussian Derivatives.

For each pixel, one can calculate the orientation of maximal gradient. This orientation is equivariant with rotation. One can use this as an "intrinsic" orientation to normalize the receptive fields at any point in the image.

Local orientation: $\theta_i(x, y, \sigma) = \text{Tan}^{-1}\left(\frac{G_y \cdot P(x, y, \sigma)}{G_x \cdot P(x, y, \sigma)}\right)$

It is possible to synthesize an oriented derivative at any point as a weighted sum of derivatives in perpendicular directions. The weights are given by sine and cosine functions.

$$G_1^\theta(x, y, \sigma) = \cos(\theta) \cdot G_x(x, y, \sigma) + \sin(\theta) \cdot G_y(x, y, \sigma)$$

By steering the Gaussian response to the local orientation, we obtain an "invariant" measure of local contrast:

Thus:

$$P_1(x, y; \theta, \sigma) = \text{Cos}(\theta) \langle G_x(x, y; \sigma) \cdot P(x, y) \rangle + \text{Sin}(\theta) \langle G_y(x, y; \sigma) \cdot P(x, y) \rangle$$

a similar measure can be obtained for the second and third derivatives:

$$G_2^\theta(x, y, \sigma) = \text{Cos}(\theta)^2 G_{xx}(x, y, \sigma) + \text{Cos}(\theta)\text{Sin}(\theta) G_{xy}(x, y, \sigma) + \text{Sin}(\theta)^2 G_{yy}(x, y, \sigma)$$

$$G_3^\theta(x, y, \sigma) = \text{Cos}(\theta)^3 G_{xxx}(x, y, \sigma) + \text{Cos}(\theta)^2 \text{Sin}(\theta) G_{xxy}(x, y, \sigma) + \text{Cos}(\theta)\text{Sin}(\theta)^2 G_{xyy}(x, y, \sigma) + \text{Sin}(\theta)^3 G_{yyy}(x, y, \sigma)$$

thus

$$P_{xx}(x,y;\theta,\sigma) = \text{Cos}(\theta)^2 \langle G_{xx}(x,y;\sigma) \cdot P(x,y) \rangle + \text{Sin}(\theta)^2 \langle G_{yy}(x,y;\sigma) \cdot P(x,y) \rangle \\ + 2\text{Cos}(\theta)\text{Sin}(\theta) \langle G_{xy}(x,y;\sigma) \cdot P(x,y) \rangle$$

and

$$P_{xxx}(x,y,\sigma,\theta) = \text{Cos}(\theta)^3 \langle G_{xxx}(x,y,\sigma) \rangle + \text{Cos}(\theta)^2 \text{Sin}(\theta) \langle G_{xxy}(x,y,\sigma) \rangle + \text{Cos}(\theta) \text{Sin}(\theta)^2 \langle G_{xyy}(x,y,\sigma) \rangle + \text{Sin}(\theta)^3 \langle G_{yyy}(x,y,\sigma) \rangle$$

By steering the Gaussian response to the local orientation, we obtain an "invariant" measure of local contrast. We can also "steer" in scale to obtain invariance to size.

3 Gaussian Scale Space

$$p(t,s) = x(t) * k\left(\frac{t}{s}\right)$$

$x(t)$ A signal (t is time or space)

$k(t)$ A kernel function

$p(t,s)$ A multi-scale representation of $x(t)$ (s is scale)

Scaling $x(t)$ translates $p(t,s)$ in scale

However, to obtain "equvariant" scaling, we need to use $\text{Log}(s)$.

$$p(t,s) = x(t) * k\left(\frac{t}{2^s}\right)$$

In this way, doubling the size of $x(t)$ translates the signal by $s+1$.

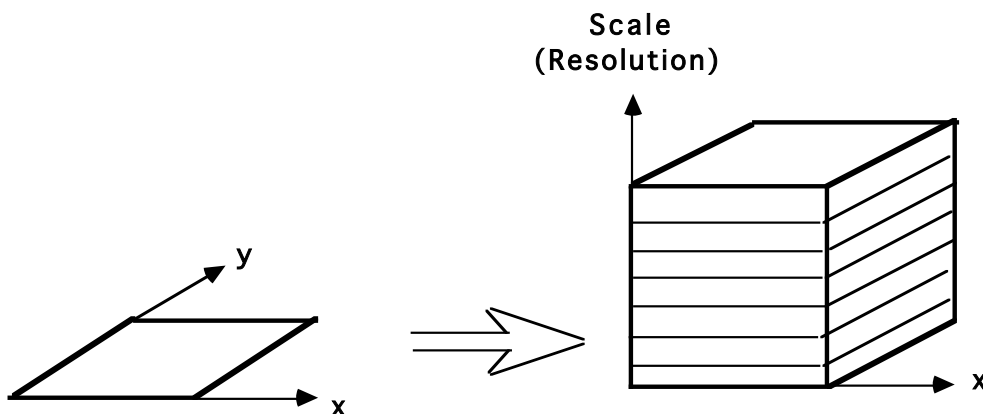
3.1 Image Scale Space:

Continuous Case.

Let $P(x,y)$ be the image.

Let $G(x, y, 2^s)$ by a Gaussian function of scale $\sigma=2^s$

Continuous x, y, s : $P(x, y, s) = P(x,y) * G(x, y, 2^s)$



The appearance of a pattern in the image results in a unique structure in $P(x, y, s)$.

This structure is "equvariant" in position, scale and rotation.

Translate the pattern by $\Delta x, \Delta y$ and the structure translates by $\Delta x, \Delta y$ in $P(x, y, s)$.

Rotate by θ in x,y and the structure rotates by θ in $P(x, y, s)$.

Scale by a factor of 2^s , and the structure translates by s in $P(x, y, s)$.

Scale space :

Separates global structure from fine detail.

Provides context for recognition.

Provides a description that is invariant to position, orientation and scale.

3.2 Discrete Scale Space

Let $P(i,j)$ be an image of size $M \times M$ pixels

We propose to sample scale a $\Delta\sigma = 2^{1/2}$ so that $\sigma_k = 2^{k/2}$

Let $G(i,j,k)$ be a kernel filter for $\sigma_k = 2^{k/2}$

The filter support is $R_k = 4\sigma_k$ so that $N_k = 2R_k + 1 = 2(2^2)2^{k/2} + 1$

$$N_k = 2^{k/2+3} + 1$$

Pyramid Definition: $P(i,j,k) = P(i,j) * G(i,j, 2^{k/2})$

for $1 \leq k \leq M-4$

Diffusion Equation: $\nabla^2 G_x(i,j,\sigma) = G_{xx}(i,j,\sigma) + G_{yy}(i,j,\sigma) = \frac{\partial G(i,j,\sigma)}{\partial \sigma}$

As a consequence: $\nabla^2 G(i,j,\sigma) \approx G(i,j,\sigma_1) - G(i,j,\sigma_2)$

This typically requires $\sigma_1 \geq \sqrt{2} \sigma_2$

Thus it is common to use:

$$\nabla^2 P(i,j,k) = \langle p(i,j), \nabla^2 G(i,j,\sigma_k) \rangle \approx P(i,j,k) - P(i,j, k-1)$$