Computer Vision

MoSIG options GVR and UIS

James L. Crowley

Fall Semester 9 November 2017

Lesson 4

Scale Invariant Pyramids, Interest Points and Descriptors

Lesson Outline:

1. Scale Space

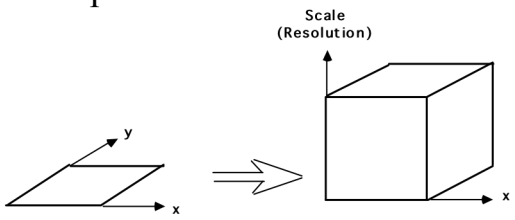
There is no "correct" value for σ. In most scenes there is information at all scales. Scale space techniques can be used to describe information at any scale in an image.

Let *P(x, y)* be a 2-D image where *(x, y)* are real values,
Let *G(x, y, σ)* be a Normalized Gaussian function of scale σ, with real valued σ.

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

The variable, σ, is sometimes referred to as the "scale" of the Gaussian.

If we consider σ to be a free variable we can define a continuous 3D space *P(x, y, s)*. This is referred to as a "Scale Space".



1.1. Definition

Scale space is defined as a convolution of the image with Gaussians over a continuous range of scales, s.

$$P(x,y,s) = P(x,y) * G(x,y,\sigma_s)$$

recall:

$$P(x,y) * G(x,y,\sigma_s) = \iint P(x-u, y-v) \cdot G(u,v,\sigma_s) du, dv$$

For some base value of σ_0, typically $\sigma_0 = 2$. In this case $\sigma_s = 2^s$

Structures can be found in scale space that stay the same when an object moves or rotates in the image, or changes in size. Such structures are said to be invariant with changes in position, rotation and size.

Note that the scale axis is exponential.
An exponential scale axis is necessary for scale "invariant image description".
When an object in an image is made larger or smaller by a factor of $D = 2^d$

$$P(x,y) \rightarrow P(x2^d, y2^d)$$

Then the projection of appearance is translated by d in the scale axis

$$P(x, y, s) \rightarrow P(x,y,s+d)$$

Similarly, translating a pattern by Δx, Δy and the structure translates by $x+\Delta x$, $y+\Delta y$ in $P(x, y, s)$.

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

1.2. Invariant Structures in Scale Space

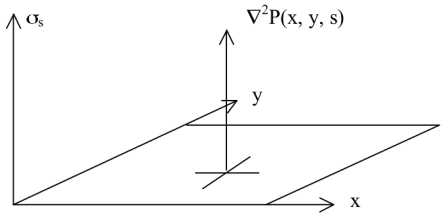
Differential Geometry can be used to find invariant structures for matching and tracking in scale space. Such structures typically correspond to peaks and ridges in the first or second derivatives, but can also be found with zero crossings or with level crossing curves. For example, scale space can be used to extract "interest points" for tracking or matching.

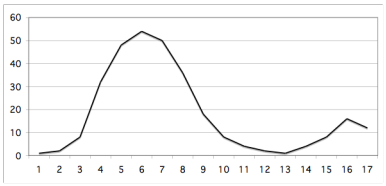
Scale-invariant interest points can be computed using
1) The Gradient Magnitude or
2) The Laplacian (second derivatives).

The Laplacian of the image is $\nabla^2 P(x,y,s) = P * \nabla^2 G(x,y,\sigma_s) = P_{xx}(x,y,s) + P_{yy}(x,y,s)$

A Laplacian profile for an image point is the Laplacian of the image computed over a continuous (exponential) range of scales.

The Laplacian profile is invariant to rotation and translation and equivariant to changes in scale. Since scale is proportional to distance, the profile is equivariant to viewing distance.





A change in viewing distance at x, y shifts the function $\nabla^2 P(x,y,s)$ in s.
The form of the profile translates in scale but remains the same.
Technically this is called a "covariant" .
A Laplacian interest point is found by $(x_i, y_i, s_i) = \underset{x,y,s}{local-\max}\{\nabla^2 P(x,y,s)\}$
Local-Max{} returns any point (x, y, s) for which the value of the function is larger than all neighbors within some distance ε.

2. Image Pyramids

Scale space is an abstract mathematical construct. To compute the scale space representation of an image we must sample scale space in x, y and in s.

Let *P(x, y)* be an image array of size *RxC* pixels, where (x, y) are integers instead of reals. A discrete scale space is:

$$P(x,y,k) = P(x,y) * G(x,y,\sigma_k)$$

where the convolution is defined using a sum:

$$P(x,y) * G(x,y,\sigma) = \sum_{u,y} P(x-u, y-v) \cdot G(u,v,\sigma)$$

For example, we can use a step size of $\sigma_o = 2$ so that

$\sigma_k = 2^k$ For k=0 to K.

At k=0 : $\sigma_k = 2^0 = 1$. $\sigma=1$ is the smallest scale that we can represent.
The largest value, K, is determined by the size of the image.
Let M = min(R, C). $K=\text{Log}_2(M)$

For k > K the scale parameter σ is larger than the image and the scale space signal rapidly converges to a constant value.

2.1. Spatial Resampling and Image Pyramids

Recall that *P(x, y, k)* is defined for integer x, y, k.
Because the Gaussian, *G(x, y, σ)*, is a low pass filter, as σ grows it becomes possible to resample the image with a larger step size without loss of information.

We can replace (x, y) with $(i \cdot \Delta x, j \cdot \Delta y)$.

$$G(x, y, \sigma_k) \rightarrow G(i \cdot \Delta x_k, j \cdot \Delta y_k, \sigma_k)$$

Where Δx_k and Δy_k are determined by σ_k

The images in the discrete scale space can be re-sampled using a sampling operation $S_{\Delta x}\{\}$

$$P(i, j, k) = S_{\Delta xk}\{P(x,y,k)\} = P(i\Delta x_k, j\Delta y_k, k)$$

A resampled scale space is known as a pyramid, and can be written as

$P(i, j, k) = P(i\Delta x_k, j\Delta y_k, k)$

Shannon's sampling theory shows that the sample size at each Δx_k, Δy_k can grow in proportion to σ_k. In the previous lecture we saw that $\sigma \geq 1$. This implies that for any value of k, $\sigma_k \geq \Delta x_k$.

It is common to use $\Delta x_k = \sigma_k = 2^k$ However, while this keeps the algebra simple, using, $\Delta x_k = \sigma_k$ results in substantial noise from aliasing.
This can be remedied by using $\sigma_k = 2^{k+1}$ and $\Delta x_k = 2^k$.

2.2. Optimization of the Pyramid Algorithm

Note that there is no need to compute the convolution at image positions that are removed by resampling. We can substantially reduce the computational cost by skipping these samples during the convolution. This can be written as:

$$P(i,j,k) = \sum_{u=-R_k}^{R_k} \sum_{v=-R_k}^{R_k} P(u - i\Delta x_k, v - j\Delta x_k) G(u,v,\sigma_k)$$

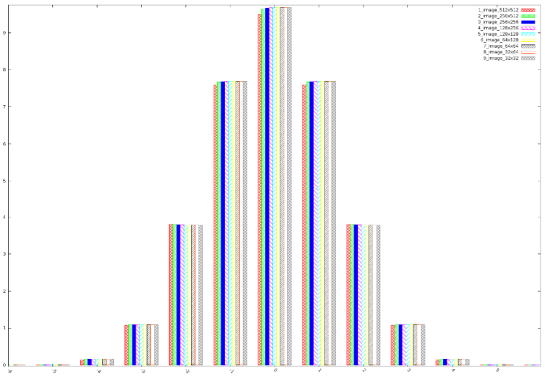
for integer i, j from (0,0) to (2^{K-k} , 2^{K-k}) and k from 0 to K and where R_k is the half size of the Gaussian window at level k. ($N_k = 2R_k + 1$) and K is the top level of the pyramid.

Typical values are $\sigma_k = 2^{k+1}$ and $\Delta x_k = 2^k$.

2.3. Scale Invariant Pyramids

Resampling $P(x, y, k)$ at $\Delta x_k \sim \sigma_k$ results an identical impulse response at each level.

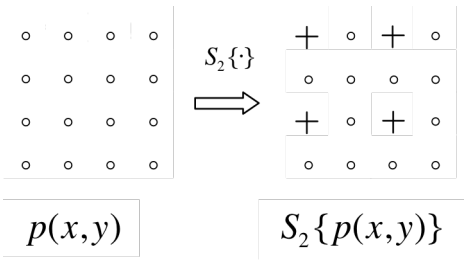
An impulse response is the output of a convolution when the input is a single impulse (Dirac Delta function). The following figure show the "impulse response" for a Scale Invariant pyramid at 6 different scales, computed using a fast O(N) algorithm.



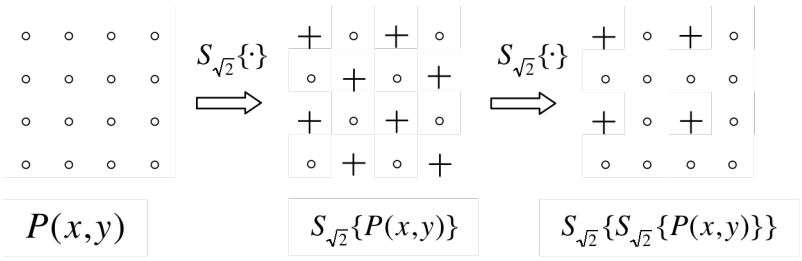
Pyramid samples are at discrete positions ($i\Delta x_k$, $j\Delta x_k$) for integer values of i, j:

$P(i, j, k) = P(i\Delta x_k, j\Delta x_k, k)$

The position in the original image of a sample from level k is $x = i\Delta x_k$ and $y = j\Delta y_k$
If we sampling at a scale step of $\Delta x_k = 2^k$ this gives a "full octave" pyramid.



It is also possible to build a scale invariant pyramid with a step size of $\Delta x_k = 2^{k/2}$ using $\Delta x_k = 2^{k/2}$ This is known as a "half-octave" pyramid.



The Half-octave pyramid requires a re-sampling algorithm that is beyond the scope of this class. For today we will use a full octave pyramid to simplify the explanation.

Let N be the number of pixels in the image. Normally computing an image pyramid for an image composed of N = RxC pixels costs O(N Log N) operations. However a very fast, O(N), recursive algorithm exists.

2.4. Computing Image Derivatives with a Gaussian Pyramid

It is possible to use the Gaussian Pyramid to compute image derivatives at scale using sums and differences. These are very similar to the receptive fields observed in the visual cortex of mammals.

Let *P(x, y)* be an image array of size *RxC* pixels, where (x, y) are integers,

A full octave Gaussian pyramid of the image is a resampled set of images resulting from the convolution of image with Gaussians at an exponential set of scales.

$P(x, y, k) = P * G(x, y, \sigma_k)$

$P(i, j, k) = P(i\Delta x_k, j\Delta y_k, k)$

where $\Delta x_k = \Delta y_k = 2^k$ and $\sigma_k = 2^{k+1}$ and $x = i\Delta x_k$ and $y = j\Delta y_k$

For any sample *p(i, j, k)* the position in the original image is $x = i\Delta x_k$ and $y = j\Delta y_k$

A pyramid of image derivatives can be defined as the convolution of image with derivatives of Gaussians

$$P_x(x,y,k) = P * G_x(x,y,\sigma_k)$$

With the Gaussian Pyramid, we can obtain a fast approximation for Gaussian derivatives by sum and difference of the samples of the Gaussian pyramid.

Then can approximate the Gaussian Derivatives as:

$$P_x(i,j,k) \approx P(i+1,j,k) - P(i-1,j,k) = P(i,j,k) * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

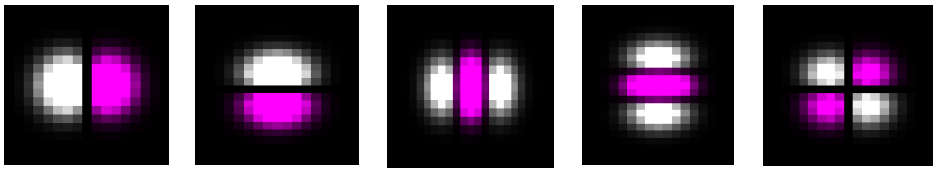
$$P_y(i,j,k) \approx P(i,j+1,k) - P(i,j-1,k) = P(i,j,k) * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

$$P_{xx}(i,j,k) \approx P(i+1,j,k) - 2P(i,j,k) + P(i-1,j,k) = P(i,j,k) * \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$

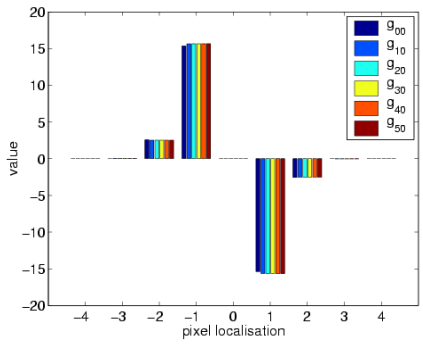
$$P_{yy}(i,j,k) \approx P(i,j+1,k) - 2P(i,j,k) + P(i,j-1,k) = P(i,j,k) * \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

$$P_{xy}(i,j,k) \approx P(i+1,j+1,k) - P(i-1,j+1,k) - P(i+1,j-1,k) + P(i-1,j-1,k) = P(i,j,k) * \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}$$

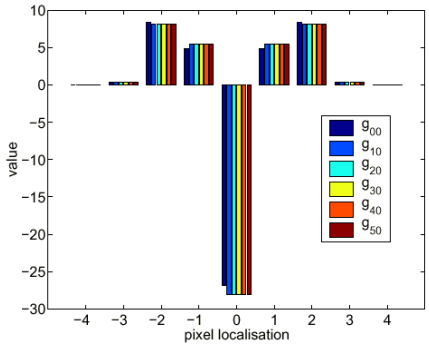
The following are some numerically evaluated examples published in the Scale Space Conference of 2003. [Crowley-Riff 2003]



$G_x(i,j,k)$ $G_y(i,j,k)$ $G_{xx}(i,j,k)$ $G_{yy}(i,j,k)$ $G_{xy}(i,j,k)$



Impulse Response for $G_x(i,j,k)$



Impulse Response for $G_{xx}(i,j,k)$

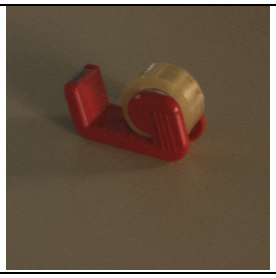
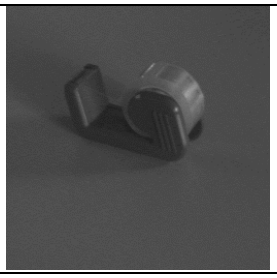
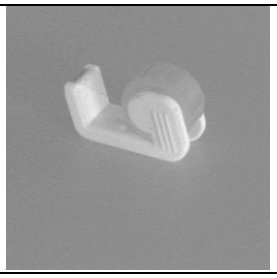
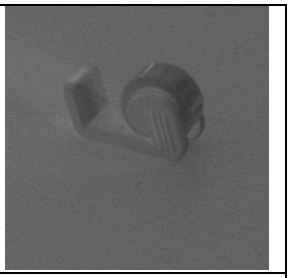
Impulse response for Gaussian derivatives for pyramid levels k=0,1,2,3,4,5.

2.5. Color Opponent Receptive Fields

A color opponent space is useful for illumination invariance

$$(R, G, B) \Rightarrow (L, C_1, C_2) \qquad \begin{pmatrix} L \\ C_1 \\ C_2 \end{pmatrix} = \begin{pmatrix} 0.33 & 0.33 & 0.33 \\ -0.5 & -0.5 & 1 \\ 0.5 & -0.5 & 0 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

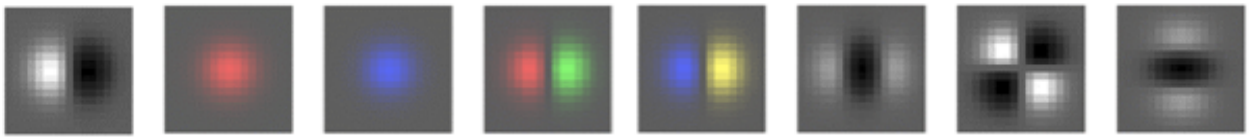
This representation separates luminance and chrominance.

			
RGB	B-W	R-G	R+G-B

Color opponent space can be used to build color opponent receptive fields.

$$\begin{pmatrix} L \\ C_1 \\ C_2 \end{pmatrix} = \begin{pmatrix} 0.33 & 0.33 & 0.33 \\ -0.5 & -0.5 & 1 \\ 0.5 & -0.5 & 0 \end{pmatrix} \begin{pmatrix} \alpha_1 R \\ \alpha_2 G \\ \alpha_3 B \end{pmatrix}$$

We then compute 3 pyramids: $L(i, j, k)$, $C_1(i, j, k)$, and $C_2(i, j, k)$,



Examples of color opponent receptive fields.

Color opponent receptive fields can be steered in color to provide color invariance. This gives us a feature vector for local appearance:

$$\vec{A}(x,y,k) = \begin{bmatrix} G_x^{L\sigma_k} \\ G^{C_1\sigma_k} \\ G^{C_2\sigma_k} \\ G_x^{C_1\sigma_k} \\ G_x^{C_2\sigma_k} \\ G_{xx}^{L\sigma_k} \\ G_{xy}^{L\sigma_k} \\ G_{yy}^{L\sigma_k} \end{bmatrix}$$

This can be generalized to include multiple scales and higher order derivatives. The result is tensor who dimensions as x, y, c, σ, θ, d

where c represents the color channel (L, C_1 or C_2) and d represents the derivative order.

2.6. Scale Invariant Interest Points and the Laplacian profile

Scale invariant interest points (or feature points) can be compute with a Gradient (First derivative) or a Laplacian (Second derivative).

Recall that the Laplacian of the image is

$$\nabla^2 P(x,y,s) = P * \nabla^2 G(x,y,\sigma_s) = P_{xx}(x,y,s) + P_{yy}(x,y,s)$$

A Laplacian profile for an image point is the Laplacian of the image computed over a continuous (exponential) range of scales.

A Laplacian interest point is $(x_i, y_i, s_i) = \underset{x,y,s}{local - \max}\{\nabla^2 P(x,y,s)\}$

Such interest points can be computed directly from a difference of levels in the Gaussian pyramid. For a Gaussian Scale Space, we can show that:

$$\nabla^2 G_x(x,y,\sigma) = G_{xx}(x,y,\sigma) + G_{yy}(x,y,\sigma) = \frac{\partial G(x,y,\sigma)}{\partial \sigma}$$

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

This is called a "Difference of Gaussians" (DoG) and requires $\sigma_1 \geq \sqrt{2}\ \sigma_2$
Thus the Laplacian of the image can be approximated as the difference at adjacent pyramid levels from a Gaussian Pyramid.

$$\nabla^2 P(x, y, k) = P(x, y, k) - P(x, y, k-1)$$

s

Thus, we can detect scale invariant interest points local maxima in the Laplacian.

$$x_i, y_i, s_i = \underset{x,y,k}{local-\max}\{P(x,y,k) - P(x,y,k-1)\}$$

Note that the points must be at the same image position (x,y). If the pyramid is resampled then we can not simply use *(i,j)*. We must use $x = i\Delta x_k$ and $y = j\Delta y_k$ to assure that the sample are at the same image position.

A difference in scale in the pyramid levels is $\Delta\sigma=2$ is not precise.
A value of $\Delta\sigma=\sqrt{2}$ is more accurate.

2.7. Natural Interest points from the Laplacian at half octave scales.

David Lowe used the scale invariant pyramid to define "Natural" interest points. These are used in the SIFT image descriptor.
(SIFT = Scale Invariant Feature Transform)

To obtain a natural interest point with a scale precision of less than $\Delta\sigma=2$ we can use cascade convolution within each pyramid level.

Consider a pyramid image at level k: $P(i, j, k)$ with σ_k

Recalle that $G(i, j, \sqrt{2}) = G(i, j, 1) * G(i, j, 1)$

we compute:

$P(i, j, k)$	σ_k
$P_1(i, j, k) = P(i, j, k) * G(i, j, 1)$	$\sigma_{k1} = \sqrt{2}\ \sigma_k$
$P_2(i, j, k) = P_1(i, j, k) * G(i, j, 1) * G(i, j, 1)$	$\sigma_{k2} = 2\ \sigma_k$
$P_3(i, j, k) = P_2(i, j, k) * G(i, j, 2)$	$\sigma_{k2} = 2\sqrt{2}\ \sigma_k$
$P_4(i, j, k) = P_3(i, j, k) * G(i, j, 2) * G(i, j, 2)$	$\sigma_{k2} = 4\ \sigma_k$

For each pixel local, we can then calculate 3 Laplacian values:

$$L_{k0} = P_1(i, j, k) - P(i, j, k)$$
$$L_{k1} = P_2(i, j, k) - P_1(i, j, k)$$
$$L_{k2} = P_3(i, j, k) - P_2(i, j, k)$$
$$L_{k3} = P_4(i, j, k) - P_3(i, j, k)$$

If $L_{k0} < L_{k1} > L_{k2}$ then the point $P(i, j, k)$ is a natural interest point at with $\sigma = \sqrt{2}\ \sigma_k$
If $L_{k1} < L_{k2} > L_{k3}$ then the point $P(i, j, k)$ is a natural interest point with $\sigma = 2\ \sigma_k$

The position of the natural interest point is $x = i \cdot 2^k$, $y = j \cdot 2^k$

This is the method used to find natural interest points in the SIFT detector described below.

2.8. Interest points from Gradient Magnitude

We can also compute an intrinsic scale for the Gradient magnitude.

The Gradient $\vec{\nabla}P(x,y,s) = \begin{pmatrix} P_x(x,y,s) \\ P_y(x,y,s) \end{pmatrix} = \begin{pmatrix} P * G_x(x,y,s) \\ P * G_y(x,y,s) \end{pmatrix}$

For any image point *(x,y)* the intrinsic scale can be computed from

$$s_i = \underset{s}{\text{local}-\max}\{\left\|\vec{\nabla}P(x,y,s)\right\|\}$$

These are positions in the image that can serve as landmarks for tracking or recognition.

In a scale-invariant pyramid, the gradient is available at any sample in the pyramid as

$$\vec{\nabla}P(i,j,k) = \begin{pmatrix} P_x(i,j,k) \\ P_y(i,j,k) \end{pmatrix} = \begin{pmatrix} P(i+1,j,k) - P(i-1,j,k) \\ P(i,j+1,k) - P(i,j-1,k) \end{pmatrix}$$

For the image gradient, a scale invariant interest point is

$$i_i, j_i, k_i = \underset{i,j,k}{\text{Local}-\max}\{\left\|\vec{\nabla}P(i,j,k)\right\|\}$$

3. Histogram of Oriented Gradients (HOG)

A local histogram of gradient orientation provides a vector of features image appearance that is relatively robust to changes in orientation and illumination.

HOG gained popularity because of its use in the SIFT feature point detector (described next). It was subsequently explored and made popular by Navneet Dalal (M2R GVR 2003) and Bill Triggs (currently at UGA Labo LJK).

Recall: The orientation of a gradient at pyramid sample (i,j,k) is:

$$\theta(i,j,k) = Tan^{-1}\left\{\frac{p_y(i,j,k)}{p_x(i,j,k)}\right\}$$

This is a number between 0 and π. We can quantize it to a value between 1 and N value by

$$a(i,j,k) = Trunc\left\{N \cdot \frac{\theta(i,j,k)}{\pi}\right\}$$

We can then build a local histogram for a window of size WxH, with upper left corner at i_o, j_o, k. We allocate a table of N cells: h(a). Then for each pixel i,j in our window:

$$\forall_{i=1}^{W} \forall_{j=1}^{H} h(a(i+i_o, j+j_o, k)) = h(a(i+i_o, j+j_o, k)) + 1$$

The result is a local feature composed of N values.
Recall that with histograms, we need around 8 samples per bin to have a low RMS error. Thus a good practice is to have N=W=H. For example N=4, W=4 and H=4. Many authors ignore this and use values such as N=8, W=4, H=4, resulting in a sparse histogram.

Remark: A fast version when N=4 replaces the inverse tangent by computing the diagonal derivatives with differences:

$$P_{\frac{\pi}{4}}(i,j,k) = P(i+1,j+1,k) - P(i-1,j-1,k)$$
$$P_{\frac{\pi}{2}}(i,j,k) = P(i,j+1,k) - P(i,j-1,k)$$
$$P_{\frac{3\pi}{4}}(i,j,k) = P(i+1,j-1,k) - P(i-1,j+1,k)$$
$$P_{\pi}(i,j,k) = P(i+1,j,k) - P(i-1,j,k)$$

To determine a(i,j,k) simply choose the maximum.

4. Scale Invariant Feature Transform (SIFT)

SIFT uses a scale invariant pyramid to compute scale invariant interest points as shown above.

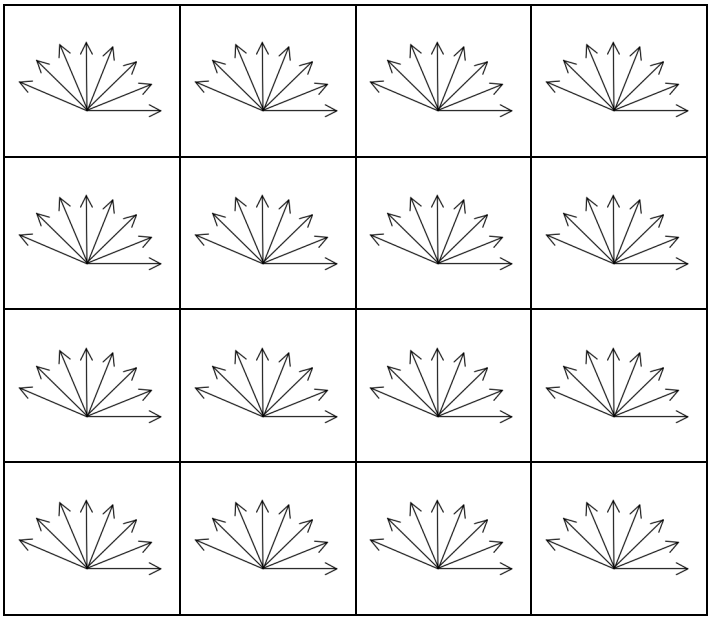
$$L_{k0} = p_1(i, j, k) - p(i, j, k)$$
$$L_{k1} = p_2(i, j, k) - p_1(i, j, k)$$
$$L_{k2} = p_3(i, j, k) - p_2(i, j, k)$$
$$L_{k3} = p_4(i, j, k) - p_3(i, j, k)$$

If $L_{k0} < L_{k1} > L_{k2}$ then the point p(i, j, k) is a natural interest point at with $\sigma = 2^{k+1/2}$
If $L_{k1} < L_{k2} > L_{k3}$ then the point p(i, j, k) is a natural interest point with $\sigma = 2^{k+1}$

For each interest point, it then computes a U x V grid of HOG detectors with N=8, W=4, H=4 at the level k
Typically U=V=4.

At level k, $\Delta i = \Delta j = 2^{k/2}$

This gives 16 x 8 = 128 features at each interest point.
This feature vector is invariant to changes in position and scale and very robust with changes in image plane rotation and illumination intensity.



Various authors experiment with other grid sizes.
For example, let the grid size be G.

G=4, W=4, H=4, N=4

Gives 64 features.

5. Harris Corner Detector

Harris, Chris, and Mike Stephens. "A combined corner and edge detector." Alvey vision conference. Vol. 15. 1988.

The Harris-Stevens Corner detector is inspired from the Moravec Interest Point detector proposed in 1973 by Hans Moravec for stereo matching. Moravec used the Sum of Squared Difference (SSD) between adjacent small patches to detect interest points. In 1988, Harris and Stevens observed that this is equivalent to an auto-correlation of the image.

$$S(x,y) = \sum_{u,v} w(u,v)\big(I(u+x,v+y) - I(u,v)\big)^2$$

where $I(x,y)$ is the image,
$w(x,y)$ is some window function, typically Gaussian.

$I(u+x,v+y)$ can be approximated as a local Taylor Series:

$$I(u+x,v+y) \approx I(u,v) + I_x(u,v)x + I_y(u,v)y$$

where $I_x(x,y)$ and $I_y(x,y)$ are the local x and y derivatives

Giving $S(x,y) = \sum_{u,v} w(u,v)\big(I_x(u,v)x + I_y(u,v)y\big)^2$

Which can be written in Matrix form as: $S(x,y) \approx \begin{pmatrix} x & y \end{pmatrix} A \begin{pmatrix} x \\ y \end{pmatrix}$ where A is the "Structure Tensor"

$$A = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

With our Gaussian pyramid this is simply: $A = \begin{bmatrix} P_x^2 & P_x P_y \\ P_x P_y & P_y^2 \end{bmatrix}$

Compute the Eigenvectors of A: $\begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = R\,A\,R^T$

where λ_1 is the maximum gradient, λ_2 is the minimum gradient.

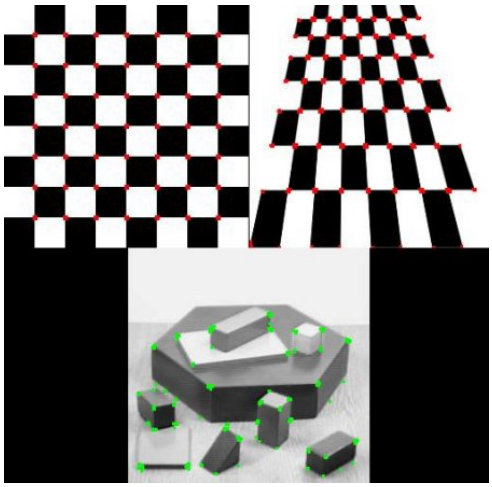
if $\lambda_1 \approx 0$ and $\lambda_2 \approx 0$ then the point is of no interest
if $\lambda_1 \approx 0$ and $\lambda_2 >> 0$ then the point is a horizontal edge
if $\lambda_1 >> 0$ and $\lambda_2 \approx 0$ then the point is a vertical edge
if $\lambda_1 \approx \lambda_2 >> 0$ then the point is corner

To avoid computing the eigenvalues (requires a square root), we can define a measure for “corner-ness”:

$$M_c = \det(A) - \kappa \cdot Trace^2(A) = \lambda_1\lambda_2 - \kappa\left(\lambda_1 + \lambda_2\right)^2$$

where κ is a tunable sensitivity parameter.

Examples of Harris-Stevens Corners:



6. Ridge Detection.

The Eigenvalues of the Hessian provide a popular ridge detector.

The Hessian at scale s is $H(x,y,s)=\begin{pmatrix} P_{xx}(x,y,s) & P_{xy}(x,y,s) \\ P_{xy}(x,y,s) & P_{yy}(x,y,s) \end{pmatrix}$

The Eigenvalues are found by diagonalizing the Hessian.
For any point in scale space *(x, y, s)*

$$\begin{pmatrix} P_{rr} & 0 \\ 0 & P_{ss} \end{pmatrix} = R\,H\,R^T \qquad \text{where} \qquad R=\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

P_{ss} is the largest value in second derivative, while P_{rr} is the smallest.
On a ridge point, P_{rr} will be the second derivative along the ridge (close to zero) while P_{ss} will be the 2nd derivative perpendicular to the ridge.

For any 2D Matrix, the principal directions can be computed directly as

$$\cos(\theta)=\sqrt{\frac{1}{2}\left(1+\frac{P_{xx}-P_{yy}}{\sqrt{\left(P_{xx}-P_{yy}\right)^2+4P_{xy}^2}}\right)}, \qquad \sin(\theta)=\operatorname{sgn}(P_{xy})\sqrt{\frac{1}{2}\left(1-\frac{P_{xx}-P_{yy}}{\sqrt{\left(P_{xx}-P_{yy}\right)^2+4P_{xy}^2}}\right)}$$

Recall that the gradient is $\vec{\nabla}P(x,y,s)=\begin{pmatrix} P_x(x,y,s) \\ P_y(x,y,s) \end{pmatrix}=\begin{pmatrix} P*G_x(x,y,s) \\ P*G_y(x,y,s) \end{pmatrix}$

for any point *(x, y, s)*, the Gradient can be aligned with the ridge using

$$P_r=\cos(\theta)P_x-\sin(\theta)P_y$$
$$P_s=\sin(\theta)P_x+\cos(\theta)P_y$$

A positive ridge point is any point, *R(x,y,s)* that satisfies:

$P_r = 0$ and $P_{rr} \le 0$ and $|P_{rr}| \ge |P_{ss}|$

A negative ridge is any point for which

$P_r=0$ and $P_{rr} \ge 0$ and $|P_{rr}| \le |P_{ss}|$

of course, P_r will rarely be exactly zero, so we use form of approximation $P_r \approx 0$

The ridge direction at (x, y, s) is: $\cos(\theta) = \frac{P_x}{\sqrt{P_x^2 + P_y^2}}$ $\sin(\theta) = \frac{P_y}{\sqrt{P_x^2 + P_y^2}}$

A Maximal ridge is a ridge point $R(x,y,s)$ for which $\text{local}-\max_s\{\nabla^2 P(x,y,s)\}$

Examples of Maximal Ridge points: