

# Computer Vision

MoSIG options GVR and UIS  
James L. Crowley

Fall Semester

15 November 2016

## Lesson 5

### Scale Invariant Pyramids, Interest Points and Descriptors

#### Lesson Outline:

1. Gaussian, Gradient and Laplacian Pyramids .....	2
1.1. Scale Space .....	2
1.2. Discrete Scale Space .....	4
1.3. Spatial Resampling and Image Pyramids.....	4
1.4. Using a Gaussian Pyramid to compute image derivatives at scale .....	6
1.5. Color Opponent Scale Space .....	7
1.6. Scale Invariant Interest Points and the Laplacian profile .....	8
1.7. Natural Interest points from the Laplacian at half octave scales.....	10
1.8. Interest points from Gradient Magnitude .....	10
2. HOG: Histogram of Oriented Gradients .....	12
3. Scale Invariant Feature Transform (SIFT).....	13
4. Harris Corner Detector .....	14
5. Ridge Detection.....	16

# 1. Gaussian, Gradient and Laplacian Pyramids

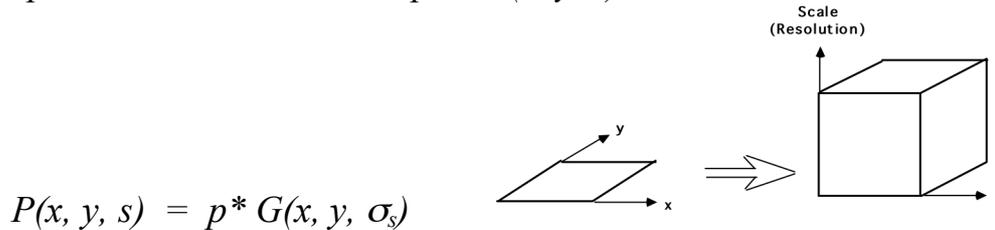
## 1.1. Scale Space

Let  $P(x, y)$  be a 2-D image where  $(x, y)$  are real values,

Let  $G(x, y, \sigma_s)$  by a Normalized Gaussian function of scale  $\sigma_s$

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

Scale Space is a continuous 3D space  $P(x, y, s)$



The scale axis ( $s$ ) in scale space is logarithmic

At  $s=0$ , the scale is defined by a Gaussian with parameter  $\sigma_0$   
(typically  $\sigma_0=1$ , but other values are possible. )

The scale axis is exponential with  $\sigma_s = \sigma_0^s$

$$\text{Log}(\sigma_s) = s \cdot \text{Log}(\sigma_0)$$

or 
$$s = \frac{\text{Log}(\sigma_s)}{\text{Log}(\sigma_0)}$$

An exponential axis is necessary for scale “covariance”.

With “covariance”, structures in scale space maintain a local distance metric through changes in scale.

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

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

It is common to use base 2 logarithms and  $\sigma_0 = 1$

In this case  $\sigma_s = 2^s$  and  $s = \text{Log}_2(\sigma_s)$  and the scale axis ranges from 0 to  $S_{\text{max}}$ .  
The value of  $S_{\text{max}}$  is determined by the number of pixels in the image.

Scale space is covariant (equivariant) in position, scale and rotation

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

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

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

Scale space is an ideal mathematical construct. In a computer we must sample scale space in  $x, y$  and  $s$ .

## 1.2. Discrete Scale Space

Let  $P(x, y)$  be an image array of size  $R \times C$  pixels, where  $(x, y)$  are integers

A discrete scales space is:

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

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

$$\sigma_k = 2^k \quad \text{For } k=0, \text{ to } K.$$

At  $k=0$  :  $\sigma_0 = 2^0 = 1$ . where  $\sigma_0 = 1$  is the smallest scale that we can represent.

let  $M = \min(R, C)$  and  $K = \text{Log}_2(M)$  at  $k=K$ ,  $\sigma_K = 2^K = 2^{\log_2(M)} = M = \min(R, C)$

so for  $k > K$  the scale parameter  $\sigma$  is larger than the image and the scale space images rapidly converges to a constant value with larger scale.

## 1.3. Spatial Resampling and Image Pyramids

Consider  $P(x, y, k)$  for integer  $x, y, k$ , where  $\sigma_k = 2^k$

Because the Gaussian,  $G(x, y, \sigma_k)$ , is a low pass filter, as  $\sigma_k$  grows it becomes possible to resample the image with a larger step size without loss of information.

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

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

A resampled scale space is known as a pyramid,  $P(i, j, k) = P(i \Delta x_k, j \Delta y_k, k)$

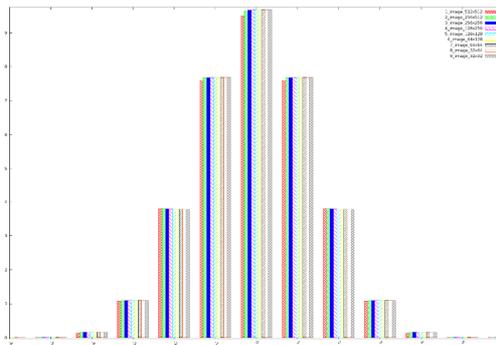
Shannon's sampling theory shows that the sample size at each  $\Delta x_k, \Delta y_k$  can grow exactly as  $\sigma_k$ .

In the previous lecture we saw that  $\sigma \geq 1$ .

This implies that for any value of  $k$ ,  $\sigma_k \geq \Delta x_k$ .

For example, it is common to use  $\Delta x_k \leq \sigma_k = 2^k$

Resampling  $P(x, y, k)$  at  $\Delta x_k = \sigma_k = 2^k$  results an identical impulse response at each level. This property is called "scale invariance".



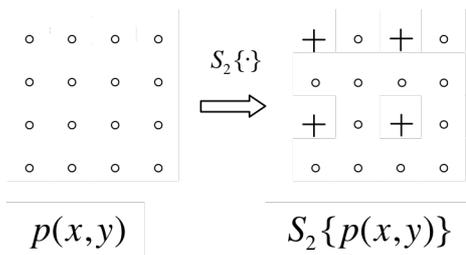
A scale invariant impulse is necessary for a scale covariant pyramid.

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

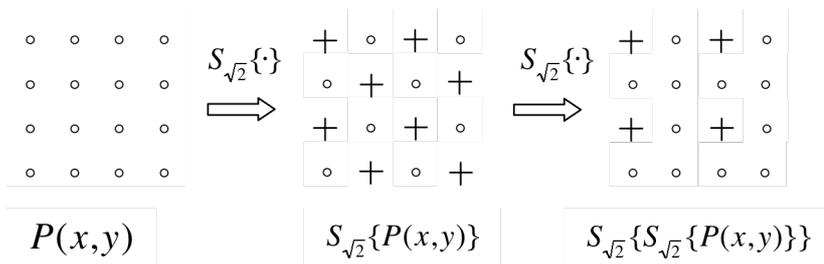
$$P(i, j, k) = P(i\Delta x_k, j\Delta y_k, k)$$

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  $\sigma_k = 2^k$  this gives a full "octave" pyramid.



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



The Half-octave pyramid requires a complicated resampling 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 scale space costs  $O(N \log N)$  operations.  
 However a very fast,  $O(N)$ , recursive algorithm exists

## 1.4. Using a Gaussian Pyramid to compute image derivatives at scale

It is possible to use the Gaussian Pyramid to compute image derivatives at scale using sums and differences.

Let  $P(x, y)$  be an image array of size  $R \times C$  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.

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

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

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

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

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

These are sometimes referred to as "Receptive Fields" because they are similar to the receptive fields observed in the visual cortex of mammals.

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

Then can approximate the Gaussian Derivatives as:

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

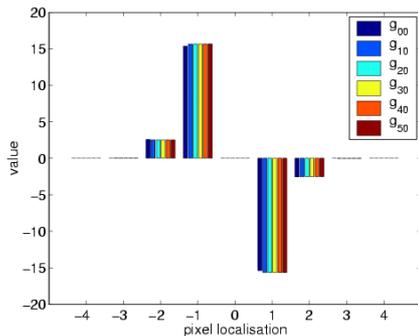
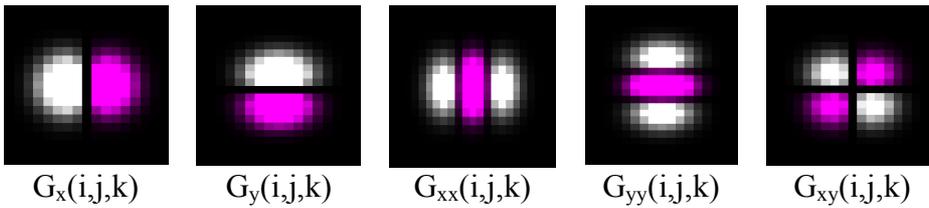
$$P_y(i, j, k) \approx P(i, j+1, k) - P(i, j-1, k) = P(i, j, k) * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

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

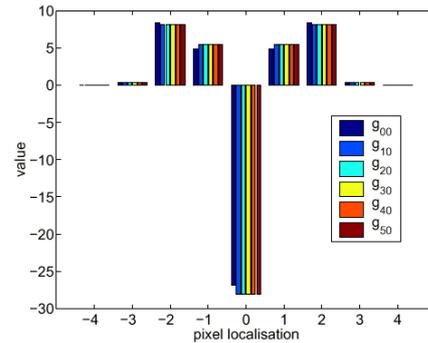
$$P_{yy}(i, j, k) \approx P(i, j+1, k) - 2P(i, j, k) + P(i, j-1, k) = P(i, j, k) * \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

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

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



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



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

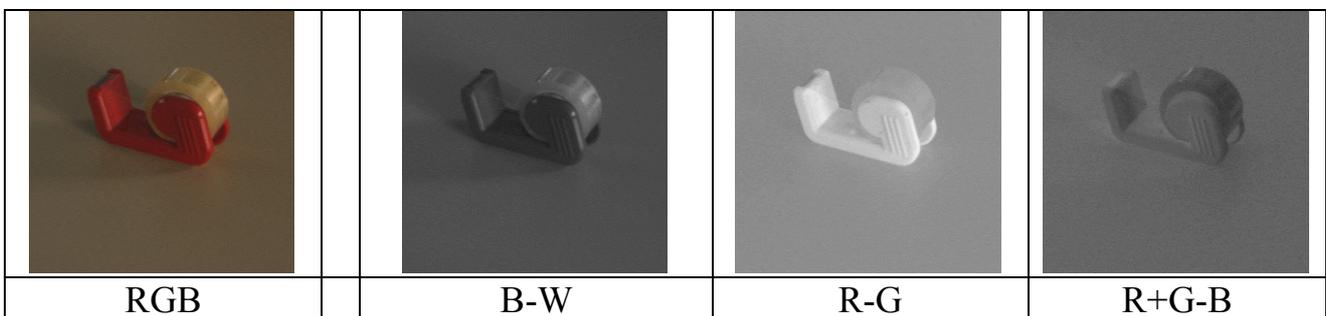
Impulse response for Gaussian derivatives for pyramid levels  $k=0,1,2,3,4,5$ .

### 1.5. Color Opponent Scale Space

A color opponent space is useful for illumination invariance

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

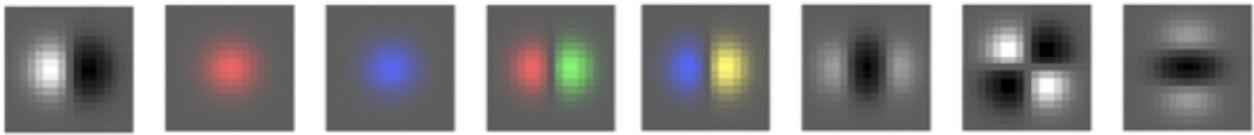
This representation separates luminance and chrominance.



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.

## 1.6. Scale Invariant Interest Points and the Laplacian profile

Differential Geometry can provide invariant features for matching and tracking. These can be with local maxima or zero crossings.

Such local maxima are called "interest points". Interest points are typically chosen to be invariant to changes in position, size, orientation, illumination and other phenomena.

Invariant interest points can be computed using

- 1) The Gradient Magnitude or
- 2) The Laplacian (second derivatives).

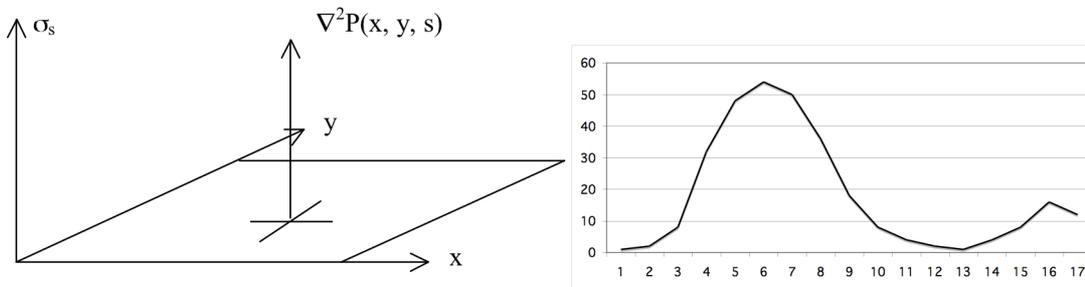
In an image scale space, these points provide landmarks for scale invariant image description.

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.

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 function remains the same. Thus the maximum is a local invariant.

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

Local-Max  $\{ \}$  returns any point  $(x, y, s)$  for which the value of the function is larger than all neighbors within a distance  $\epsilon$ . (note that we need a distance metric in scale)

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.

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

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.

### 1.7. Natural Interest points from the Laplacian at half octave scales.

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  $\sigma_k = 2^k$

we compute:

$$\begin{aligned} P_1(i, j, k) &= P(i, j, k) * G(i, j, 1) & \sigma_{k1} &= 2^{k+1/2} \\ P_2(i, j, k) &= P_1(i, j, k) * G(i, j, 1) & \sigma_{k2} &= 2^{k+1} \\ P_3(i, j, k) &= P_2(i, j, k) * G(i, j, 2) & \sigma_{k2} &= 2^{k+2} \\ P_4(i, j, k) &= P_3(i, j, k) * G(i, j, 4) & \sigma_{k2} &= 2^{k+4} \end{aligned}$$

For each pixel local, we can then calculate 4 Laplacian values. :

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

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  $\sigma = 2^{k+1}$

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.

### 1.8. Interest points from Gradient Magnitude

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

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

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

$$s_i = \text{local} - \max_s \{ \|\vec{\nabla}P(x, y, s)\| \}$$

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

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

## 2. HOG: Histogram of Oriented Gradients

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 (CNRS Labo LJK).

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

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

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

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

We can then build a local histogram for a window of size  $W \times H$ , 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:

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

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.

### 3. 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  $\sigma = 2^{k+1}$

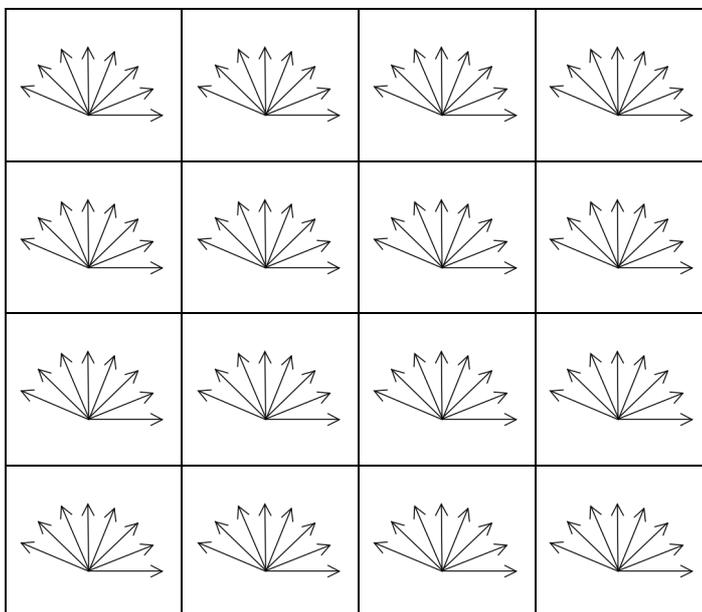
For each interest point, it then computes a  $U \times V$  grid of HOG detectors with  $N=8$ ,  $W=4$ ,  $H=4$  at the level  $k$

Typically  $U=V=4$ .

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

This gives  $16 \times 8 = 128$  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.

## 4. 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.

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

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

Which can be written in Matrix form as:  $S(x,y) \approx (x \ y) A \begin{pmatrix} x \\ y \end{pmatrix}$  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  $\lambda_1$  is the maximum gradient,  $\lambda_2$  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  $\lambda_2 \gg 0$  then the point is a horizontal edge

if  $\lambda_1 \gg 0$  and  $\lambda_2 \approx 0$  then the point is a vertical edge

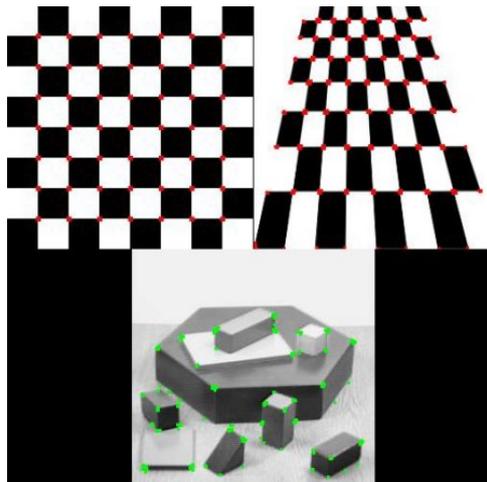
if  $\lambda_1 \approx \lambda_2 \gg 0$  then the point is corner

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

$$M_c = \det(A) - \kappa \cdot \text{Trace}^2(A) = \lambda_1 \lambda_2 - \kappa (\lambda_1 + \lambda_2)^2$$

where  $\kappa$  is a tunable sensitivity parameter.

Examples of Harris-Stevens Corners:



## 5. 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  $(x, y, s)$

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

$P_{ss}$  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

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

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

$$\begin{aligned} P_r &= \cos(\theta)P_x - \sin(\theta)P_y \\ P_s &= \sin(\theta)P_x + \cos(\theta)P_y \end{aligned}$$

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

$$P_r = 0 \text{ and } P_{rr} \leq 0 \text{ and } |P_{rr}| \geq |P_{ss}|$$

A negative ridge is any point for which

$$P_r = 0 \text{ and } P_{rr} \geq 0 \text{ and } |P_{rr}| \leq |P_{ss}|$$

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  $\text{local}_s - \max\{\nabla^2 P(x, y, s)\}$

Examples of Maximal Ridge points:

