Image Formation and Analysis
(Formation et Analyse d'Images)

James L. Crowley

ENSIMAG 3 - MMIS Option MIRV First Semester 2010/2011

Lesson 9 10 Jan 2011

Invariant Image Description

Lesson Outline:

1 Scale Invariant Interest Points

Maximal points in the image derivatives provide keypoints.
In an image scale space, these points are scale invariant.

Example: maxima in the Lapacian as invariant "interest points"

Recall the Laplacian of the image :

$$\nabla^2 P(x,y,s) = P * \nabla^2 G(x,y,\sigma) = P * G_{xx}(x,y,\sigma) + P * G_{yy}(x,y,\sigma) \approx P * \nabla^2 G(x,y,\sigma_1) - P * \nabla^2 G(x,y,\sigma_2)$$

We can compute the Laplacian from a Gaussian Pyramid as a difference of samples at adjactent levels.

DoG: $L(i, j, k) = \nabla^2 P(i, j, k) = P(i, j, k) - P(i, j, k-1)$

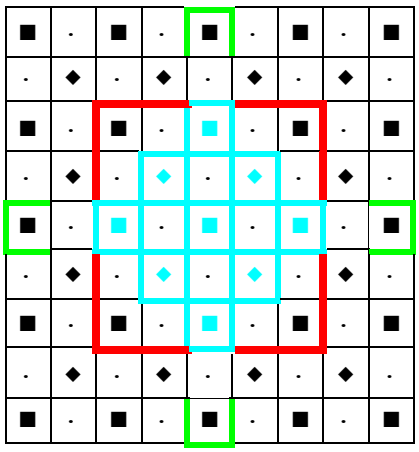
This is referred to as a "Difference of Gaussian" or DoG detector.

We can detect scale invariant interest points as

$$X(i,j,k) = \underset{i,j,k,R}{local - \max}\{L(i,j,k)\}$$

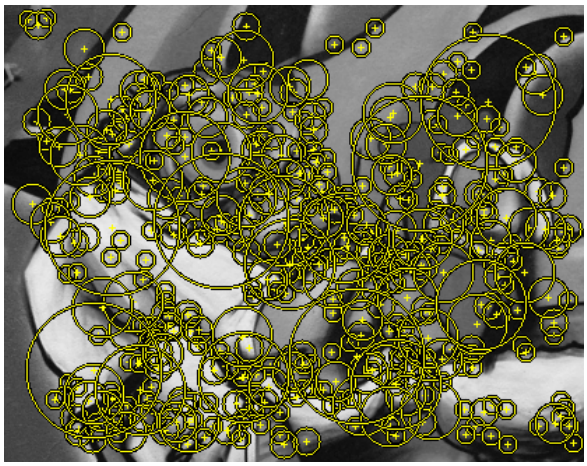
with R=2

Note that because of resampling, $\Delta x = 2^{k/2}$, the neighborhood grows larger as k increases.



Level k-1 - Blue
Level k - Red
level k+1 green.

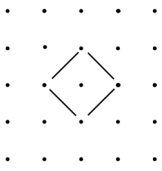
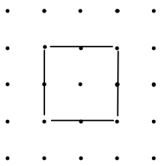
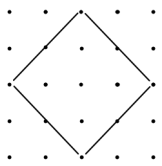
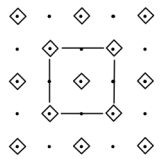
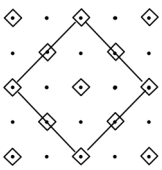
Such points are used for tracking, for image registration, and as feature points for recognition.



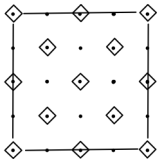
Examples of Local Maxima in the Laplacian Pyramid

Neighborhoods in a Pyramid

Computing a variable Radius Local-Max operator over a $\sqrt{2}$ image pyramid can be somewhat complex.

R=1	4 Nearest Neighbors - Odd (Cartesian) levels :	
R=$\sqrt{2}$	8 Nearest Neighbors - Odd (Cartesian) levels :	
R=2	12 Nearest Neighbors Odd Levels	
R=1	4 Nearest Neighbors - Even ($\sqrt{2}$)levels	
R=$\sqrt{2}$	8 Nearest Neighbors - Even ($\sqrt{2}$)levels	

R=2 12 Nearest Neighbors
Even ($\sqrt{2}$)levels



Other popular interest point detectors.

Other popular detectors for scale invariant interest points include:

Gradient Magnitude: $X(i,j,k) = Local_{i,j,k} - \max\{\| P_x(i,j,k), P_y(i,j,k) \|\}$

and Determinant of the Hessian: $X(i,j,k) = Local_{i,j,k} - \max\left\{ \det\begin{pmatrix} P_{xx}(i,j,k) & P_{xy}(i,j,k) \\ P_{xy}(i,j,k) & P_{yy}(i,j,k) \end{pmatrix} \right\}$

$$X(i,j,k) = Local_{i,j,k} - \max\left\{ P_{xx}(i,j,k) P_{yy}(i,j,k) - P_{xy}(i,j,k)^2 \right\}$$

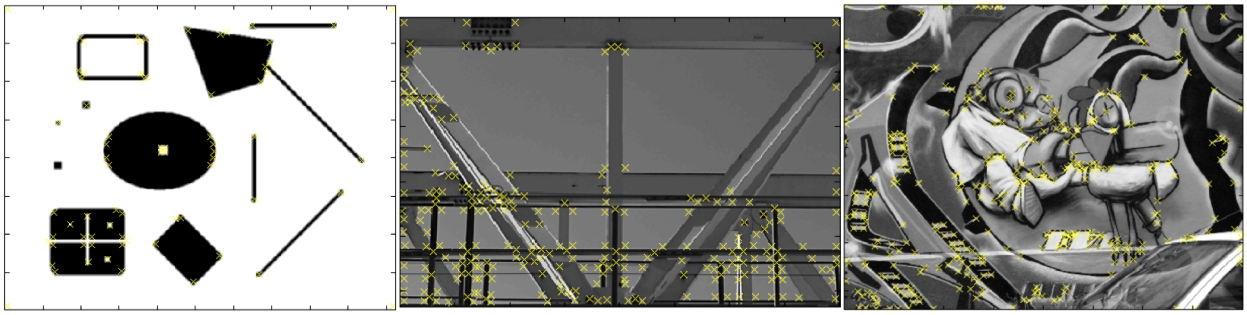


Figure 1. Results from the Hessian Detector proposed by Beaudet 78

and the Harris-Laplace.

$$\text{let } b_2(i,j) = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

$$H_x^2 = b_2 * P_x^2$$
$$H_{xy} = b_2 * P_{xy}$$
$$H_y^2 = b_2 * P_y^2$$

$$H = \begin{pmatrix} H_x^{\,2} & H_{xy} \\ H_{xy} & H_y^{\,2} \end{pmatrix}$$

Harris interest points h(i,j,k) = arg-max{det(H)-Trace(H)}

Figure 1 shows results from a Hessian detector [Beaudet1978]:

Figure 2 shows the Harris detector [Harris1988]:

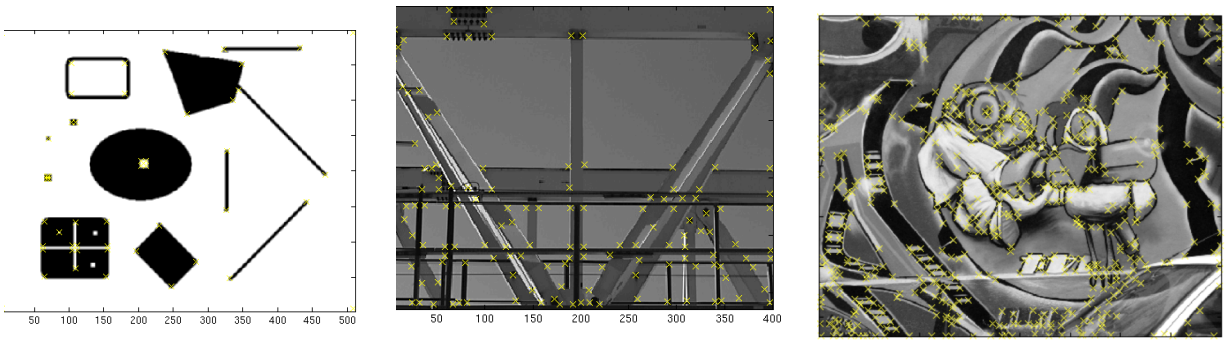


Figure 2 The Harris detector detects mainly corners, but in an image with few corners there are many more features than just corners.

2 HoG: Histogram of Gradients

A local histogram of gradient orientation provides a vector of features image appearance that is equivariant 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 (CR 1 - Laboratoire 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

$$a(i,j,k) = N \cdot Trunc\left\{\frac{\theta(i,j,k)}{\pi}\right\}+1$$

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.

3 Scale Invariant Feature Transform (SIFT)

SIFT uses a scale invariant pyramid to compute a scale invariant DoG interest point detector to detect local scale-invariant interest points.

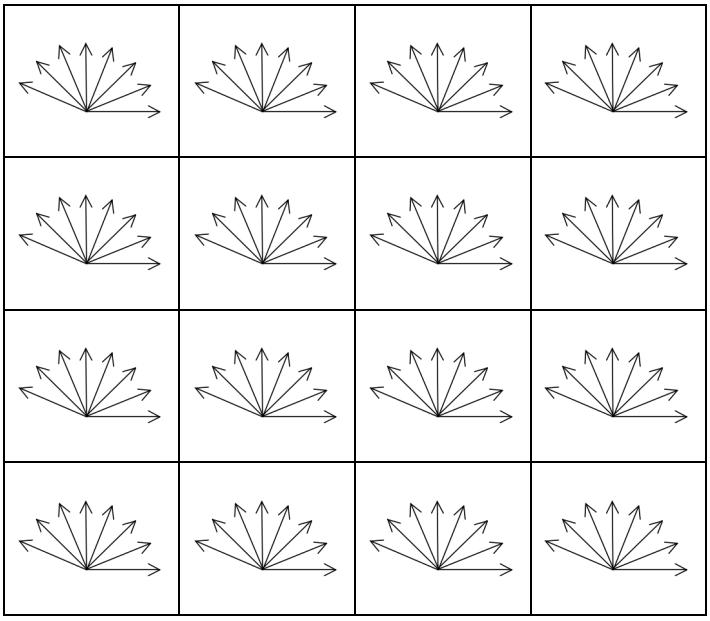
$$X(i,j,k) = \underset{i,j,k,R=2}{Local} - \max\{P(i,j,k) - P(i,j,k-1)\}$$

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 16 = 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 Haar Wavelets and Integral Images

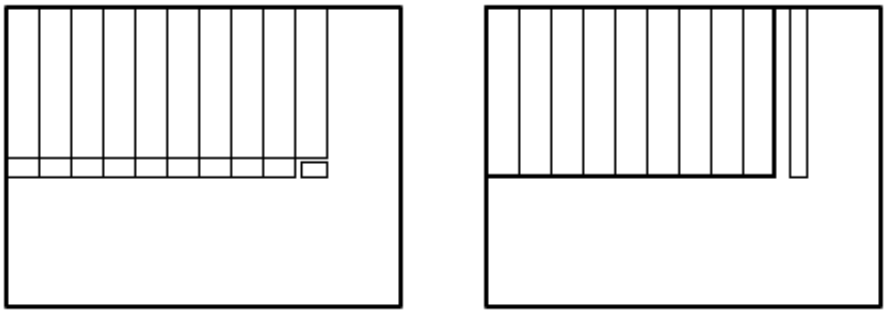
Integral Images.

An integral image is an image where each pixel contains the sum from the upper left corner:

$$ii(u,v) = \sum_{i=1}^{u}\sum_{j=1}^{v} p(i,j)$$

Integral images have been used for decades to compute local energy for normalization of images. An integral image provides a structure for very fast computing local image features.

Fast Integral Image algorithm.



An integral image, *ii(x,y)* of a window *p(x,y)* is the sum of all pixels from the upper left corner (1,1) to the current pixel (x,y).

$$ii(x,y) = \sum_{u=1,v=1}^{x,y} p(u,v)$$

A recurrence formula may be used to compute the integral image in 2 operations (memory access and additions) per pixel. This operation starts by computing an intermediate sum for each row:

$$s(x,y) = s(x,y-1) + i(x,y)$$

This intermediate sum is then used to compute the sum of rectangles.

$$ii(x,y) = ii(x-1,y) + s(x,y)$$

The fast algorithm involves a row buffer that contains the sum of each row

For j = 1 to I
For i=1 to J
r(i) := s(i) + p(i,j)
ii(i,j) = ii(i-1,j)+s(i)

Thus the total cost of computing ii(x,y) for a window of size W·H is 2WH adds.

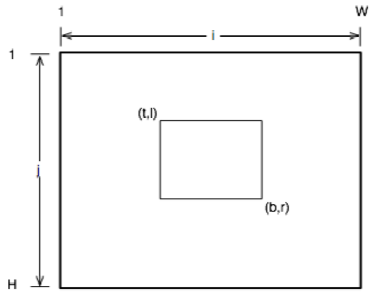
Integral images can be used to provide VERY fast computation of "box" features. These can be used to compute Haar wavelets.

Fast Box Features from Integral Images.

A box feature is the sum of pixels over a rectangle from top (t) to left (l) and bottom (b) to right (r). This may be represented by a vector (t, l, b, r).

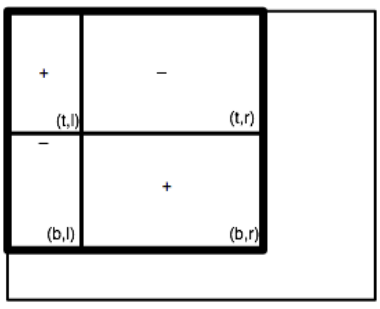
With the constraints : top < bottom and right > left.

$$b(t,l,b,r) = \sum_{i=l}^{r}\sum_{j=t}^{b} p(i,j)$$



For an window of size WxH there are $N=W^2H^2/4$ possible boxes.

$$N = \frac{W^2}{2}\cdot\frac{H^2}{2}$$



Any box feature can be computed with 4 operations (additions/subtractions).

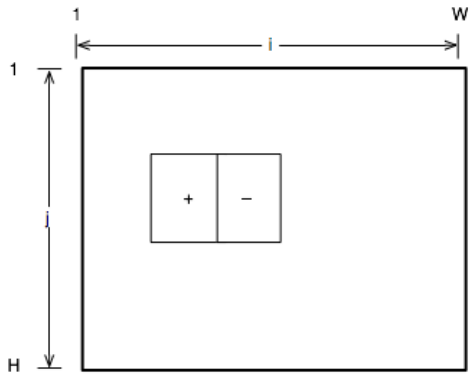
box(t,l,b,r)=ii(b,r)–ii(t,r)–ii(b,l)+ii(t,l)

Difference of Boxes

A first order Difference of Boxes (DoB) feature is a difference of two boxes $box(t1,l1,b1,r1)$.

$$DoB(t1,l1,b1,r1,t2,l2,b2,r2) = box(t1,l1,b1,r1) - box(t2,l2,b2,r2)$$

There are N^2 possible 1st order (2 box) DoB features in an image
There are N^3 possible 2nd order (3 box) DoB features in an image.
Not all DoBs are useful.



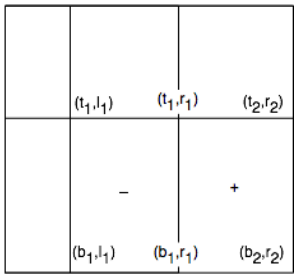
An interesting subclass are Difference of Adjacent Boxes where the sum of pixels is 0. These are a form of Haar wavelet.

An arbitrary 1st order difference of boxes costs 8 ops.

$$\begin{aligned} DoB(t_1,l_1,b_1,r_1,t_2,l_2,b_2,r_2) = {} & ii(b_1,r_1)–ii(t_1,r_1)–ii(b_1,l_1)+ii(t_1,l_1) \\ & – ii(b_2,r_2)–ii(t_2,r_2)–ii(b_2,l_2)+ii(t_2,l_2) \end{aligned}$$

However, a 1st order Haar wavelet costs only 6 ops because $r_1=l_2$ and thus

$ii(t_1,r_1) = ii(t_2,l_2)$ and $ii(b_1,r_1)= ii(b_2,l_2)$

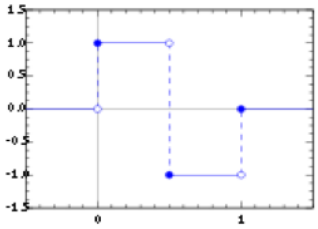


$$Haar(t_1,l_1,b_1,r_1,b_2,r_2) = ii(b_2,r_2)–2ii(b_1,r_1)+ii(b_1,l_1)–ii(t_2,r_2)+2ii(t_1,r_1)–ii(t_1,l_1)$$

Haar Wavelets

Haar A. Zur Theorie der orthogonalen Funktionensysteme, Mathematische Annalen, 69, pp 331–371, 1910.

The Haar wavelet is a difference of rectangular Windows.



$$h(t) = \begin{cases} 1 & \text{for } 0 \le t < 0.5 \\ -1 & \text{for } 0.5 \le t < 1 \\ 0 & \text{for } t < 0 \text{ and } t \ge 1 \end{cases}$$

The Haar wavelet may be shifted by d and scaled by s

$$h(t;s,d) = h(t/s - d)$$

Note that the Haar Wavelet is zero gain (zero sum).

$$G = \int_{-\infty}^{\infty} h(t)dt = 0$$

The Digital (discrete sampled) form of Haar wavelet is

$$h(n;d,k) = \begin{cases} 1 & \text{for d} \le n < d + k/2 \\ -1 & \text{for d} + \text{k}/2 \le n < d + k \\ 0 & \text{for } n < d \text{ and } n \ge d + k \end{cases}$$

Haar wavelets can be used to define an orthogonal transform analogous to the Fourier basis. This can be used to define an orthogonal transform (the Walsh-Hadamard Transform). The basis is

$$H_0 = +1$$

$$H_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H_2 = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

…

$$H_m = \frac{1}{\sqrt{2}}\begin{bmatrix} H_{m-1} & H_{m-1} \\ H_{m-1} & -H_{m-1} \end{bmatrix}$$

Haar Functions, and the Walsh-Hadamard transform have been used in Functional Analysis and signal processing for nearly a century.

In the 1980s the Wavelet community re-baptized the Haar functions as "wavelets" and demonstrated that the Walsh-Hadamard transform is the simplest form of wavelet transform.

A 2-D form of Walsh-Hadamard transform may be defined using DoB features.

Viola-Jones Face Detector

In 2001, Paul Viola and Mike Jones at MERL (Misubishi Research Labs) showed that Haar wavelets could be used for real time face detection using a cascade of linear classifiers.

They computed the Haar Wavelets for a window from integral images.

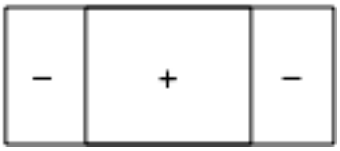
The innovation in the Viola-Jones face detector resulted from

1) A very large number of very simple features (Haar wavelets).
2) The use of the Ada boost algorithm to learn an arbitrariy good detector.

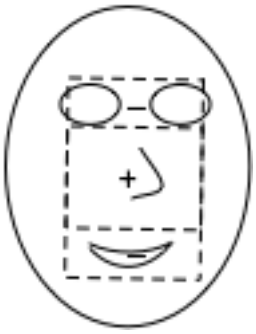
HAAR wavelets are computed using difference of Boxes, with Integral Images.

A WxH image contains $N = W^2H^2/4$ possible 1st order Haar wavelets.
(difference of adjacent boxes of same size).

Similarly, any 2nd Haar wavelet can be computed with 8 ops.



A WxH window contains $W^2H^2/8$ possible 1st and 2nd order Haar wavelets.
These provide a space of $N = W^2W^2/8$ features for detecting Faces.
Each feature, F_n is defined as a difference of boxes.



For a give position (u,v) and scale (s) any window W(u, v, s) that contains a face is a point "+" in this very N-dimensional space.