Image Features:

# Scale Invariant Interest Point Detection

# Scale Invariant Interest Points

How can we **independently** select interest points in each image, such that the detections are repeatable across different scales?

image 1

image 2

[Source: K. Grauman, slide credit: R. Urtasun]

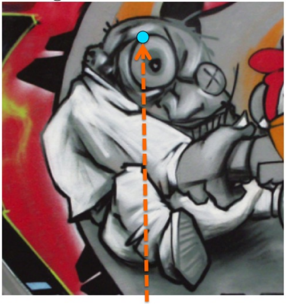
# Scale Invariant Interest Points

How can we **independently** select interest points in each image, such that the detections are repeatable across different scales?

image 1

If I detect an interest point here

image 2



Then I also want to detect one here

[Source: K. Grauman, slide credit: R. Urtasun]

# Scale Invariant Interest Points

How can we **independently** select interest points in each image, such that the detections are repeatable across different scales?

- Extract features at a variety of scales, e.g., by using multiple resolutions in a pyramid, and then matching features at the same level.
- When does this work?

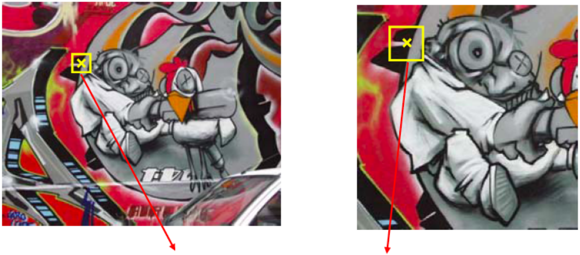
image 1 image 2

If I detect an interest point here Then I also want to detect one here

# Scale Invariant Interest Points

How can we **independently** select interest points in each image, such that the detections are repeatable across different scales?

- More efficient to extract features that are stable in both location and scale.



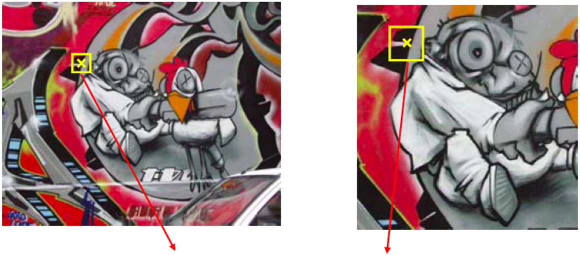
$$f(I_{i_1 \ldots i_m}(x, \sigma)) = f(I_{i_1 \ldots i_m}(x', \sigma'))$$

[Source: K. Grauman, slide credit: R. Urtasun]

# Scale Invariant Interest Points

How can we **independently** select interest points in each image, such that the detections are repeatable across different scales?

- Find scale that gives local maxima of a function $f$ in both position and scale.

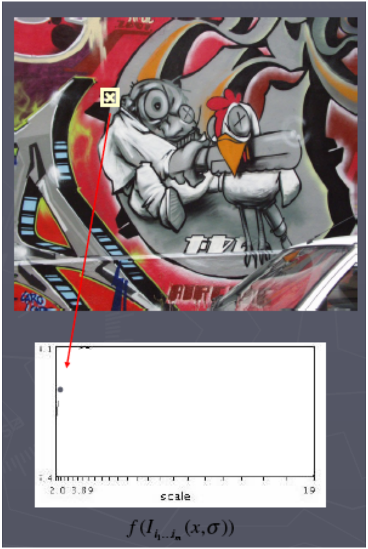


$$f(I_{i_1 \dots i_m}(x, \sigma)) = f(I_{i_1 \dots i_m}(x', \sigma'))$$

[Source: K. Grauman, slide credit: R. Urtasun]

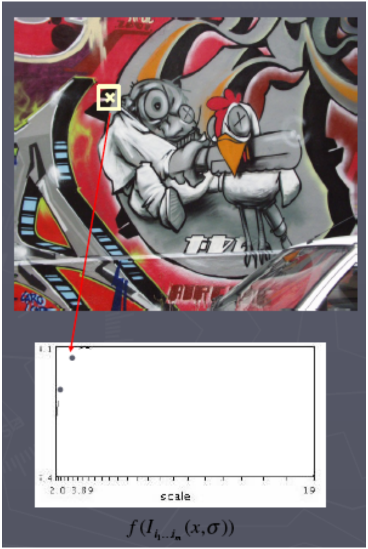
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



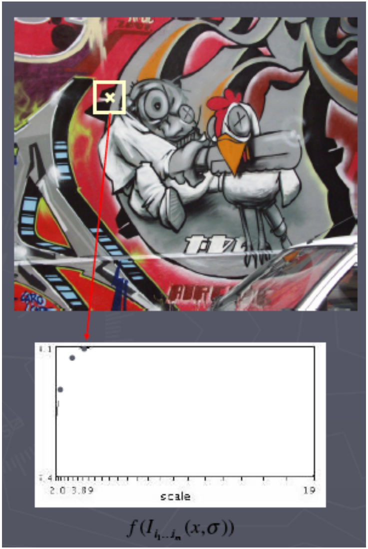
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



# Automatic Scale Selection

Function responses for increasing scale (scale signature).



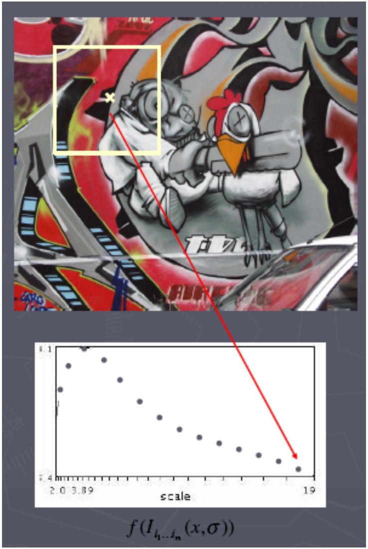
# Automatic Scale Selection

Function responses for increasing scale (scale signature).


$f(I_{i_1..i_m}(x,\sigma))$

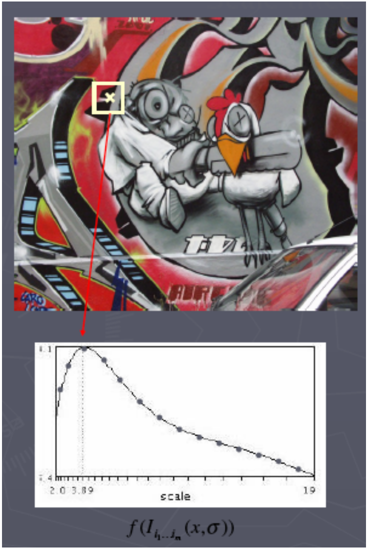
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



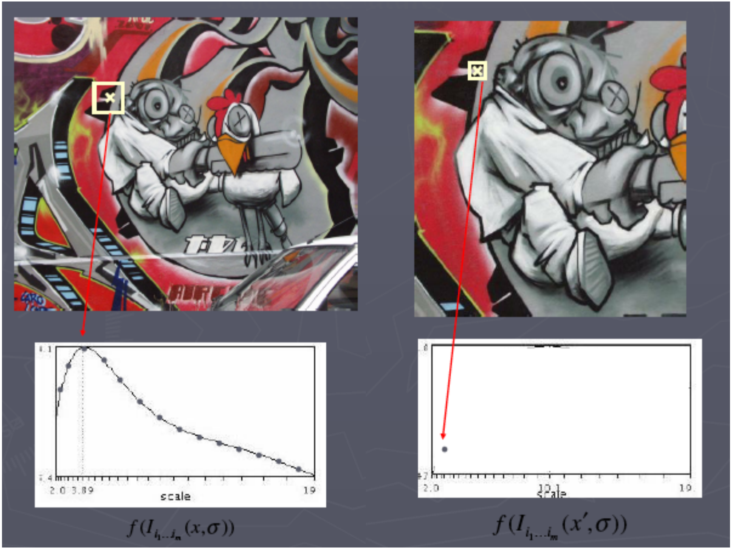
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



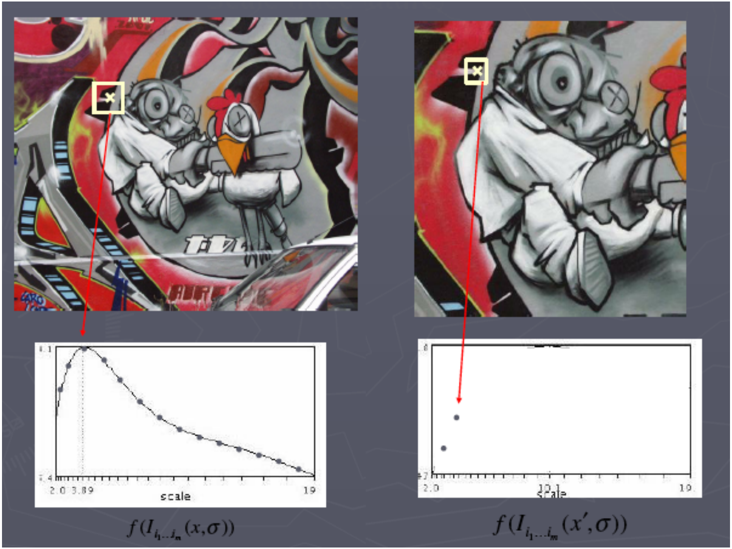
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



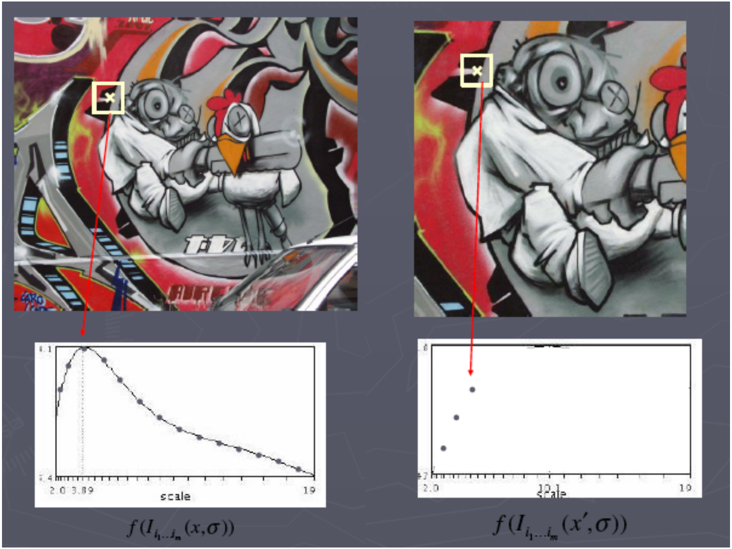
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



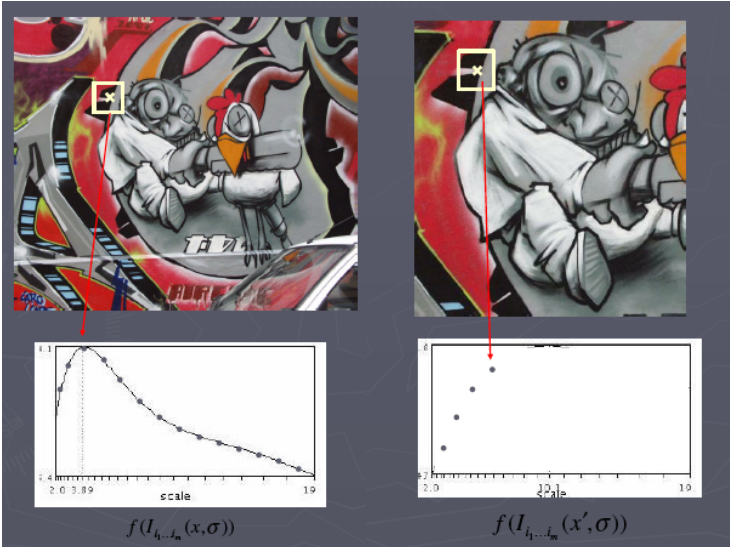
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



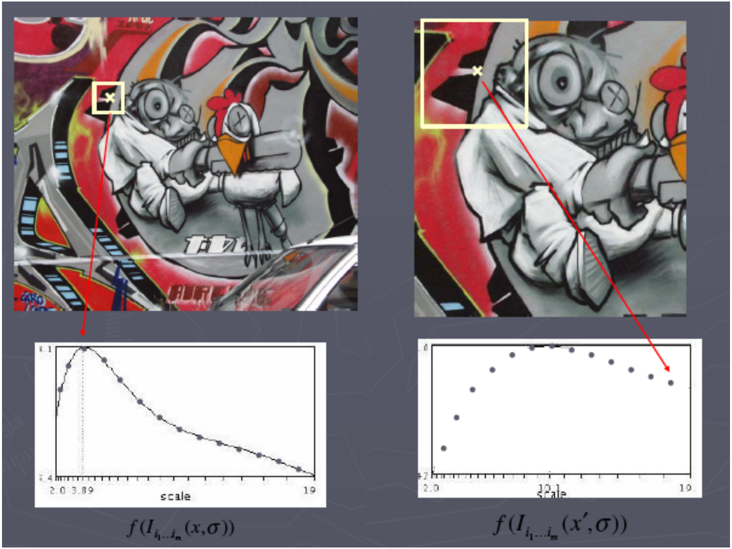
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



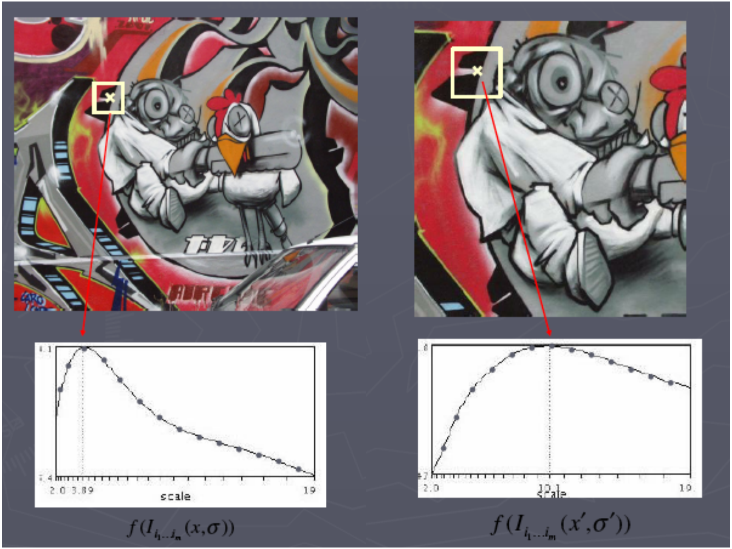
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



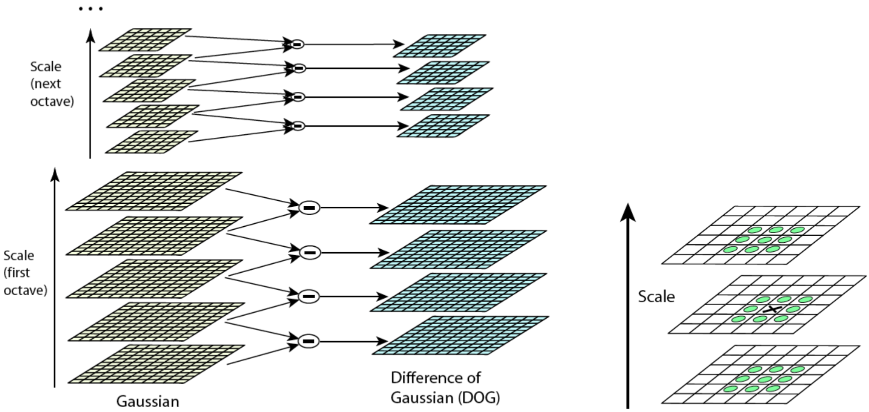
# Automatic Scale Selection

Function responses for increasing scale (scale signature).



# What Can the Signature Function Be?

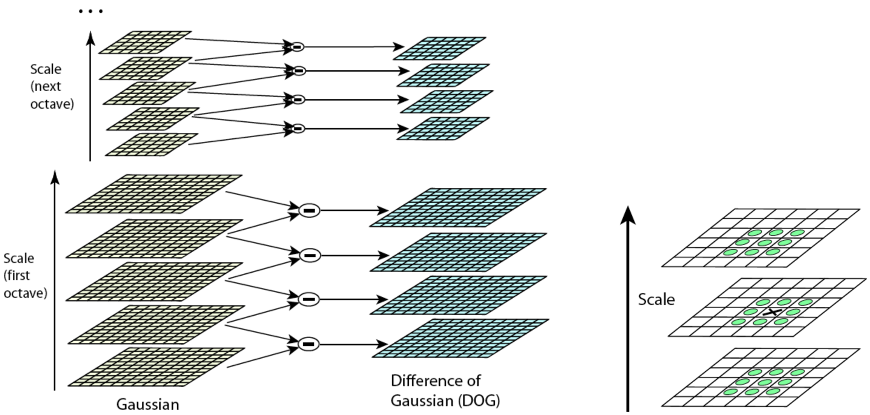
- Lindeberg (1998): extrema in the Laplacian of Gaussian (LoG).
- Lowe (2004) proposed computing a set of sub-octave Difference of Gaussian filters looking for 3D (space+scale) maxima in the resulting structure.




[Source: R. Szeliski, slide credit: R. Urtasun]

# What Can the Signature Function Be?

- Lindeberg (1998): extrema in the Laplacian of Gaussian (LoG).
- Lowe (2004) proposed computing a set of sub-octave Difference of Gaussian filters looking for 3D (space+scale) maxima in the resulting structure.



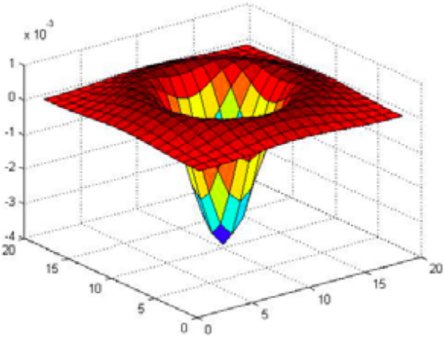

[Source: R. Szeliski, slide credit: R. Urtasun]

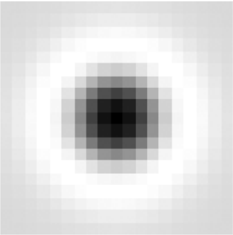
# Blob Detection – Laplacian of Gaussian

- Laplacian of Gaussian: We mentioned it for edge detection

$$\nabla^2 g(x, y, \sigma) = \frac{\partial^2 g(x, y, \sigma)}{\partial x^2} + \frac{\partial^2 g(x, y, \sigma)}{\partial y^2}, \qquad \text{where } g \text{ is a Gaussian}$$

- It is a circularly symmetric operator (finds difference in all directions)
- It can be used for 2D blob detection! How?

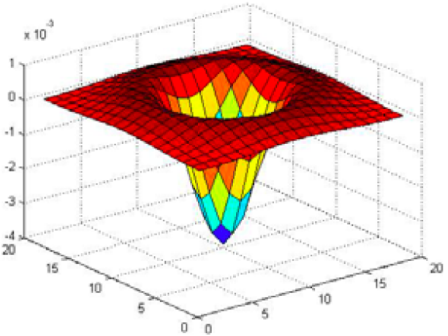


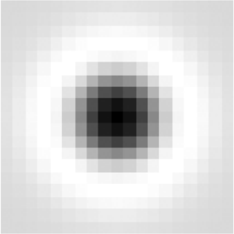



- Laplacian of Gaussian: We mentioned it for edge detection

$$\nabla^2 g(x, y, \sigma) = -\frac{1}{\pi\sigma^4}\left(1 - \frac{x^2 + y^2}{2\sigma^2}\right) \exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

- It is a circularly symmetric operator (finds difference in all directions)
- It can be used for 2D blob detection! How?

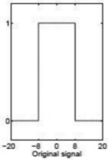


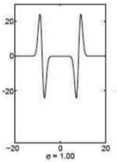
# Blob Detection – Laplacian of Gaussian

- It can be used for 2D blob detection! How?

Original signal



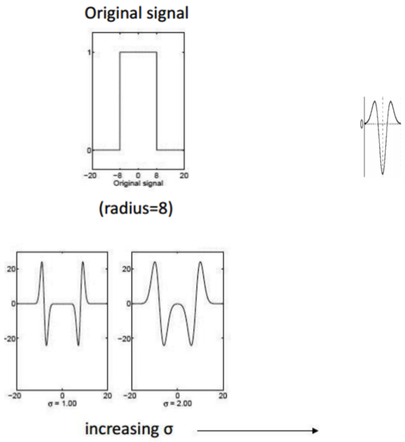

(radius=8)




[Source: F. Flores-Mangas]

# Blob Detection – Laplacian of Gaussian

- It can be used for 2D blob detection! How?

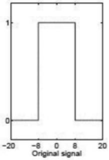



[Source: F. Flores-Mangas]

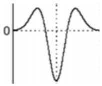
# Blob Detection – Laplacian of Gaussian

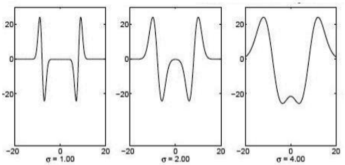
- It can be used for 2D blob detection! How?

Original signal




(radius=8)



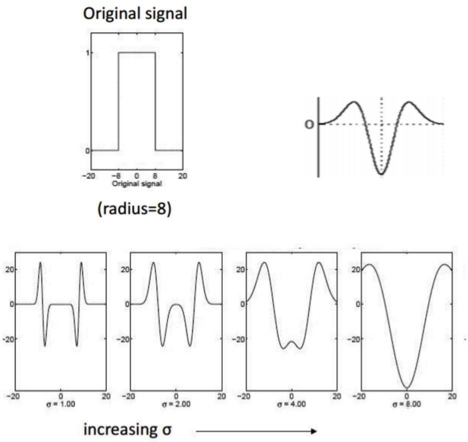




increasing σ

[Source: F. Flores-Mangas]

# Blob Detection – Laplacian of Gaussian

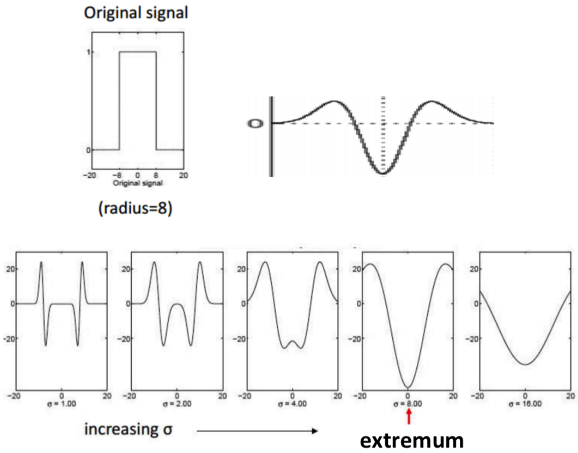
- It can be used for 2D blob detection! How?




[Source: F. Flores-Mangas]

# Blob Detection – Laplacian of Gaussian

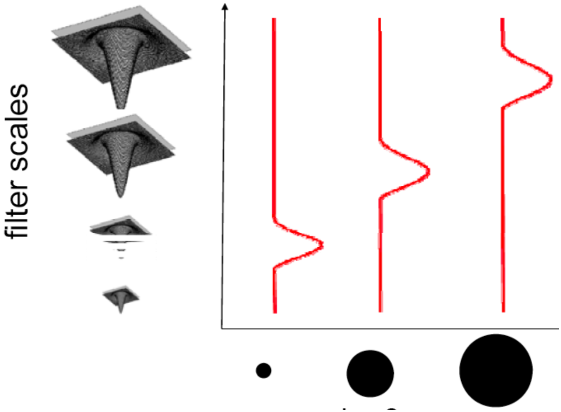
- It can be used for 2D blob detection! How?




[Source: F. Flores-Mangas]

# Blob Detection in 2D: Scale Selection

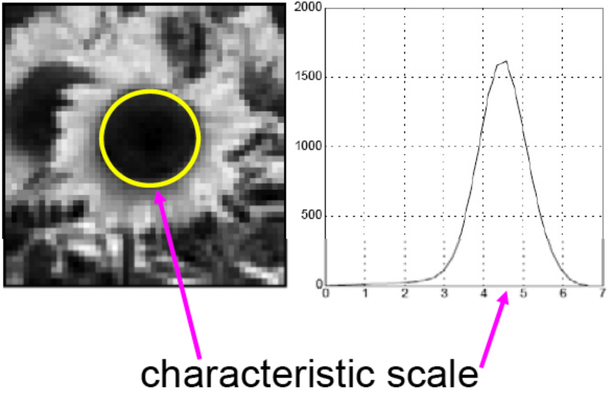
Laplacian of Gaussian = blob detector




[Source: B. Leibe, slide credit: R. Urtasun]

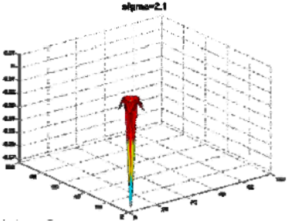
# Characteristic Scale

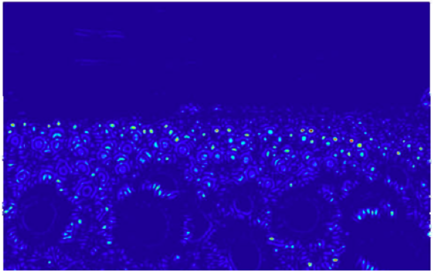
- We define the **characteristic scale** as the scale that produces peak (minimum or maximum) of the Laplacian response




[Source: S. Lazebnik]

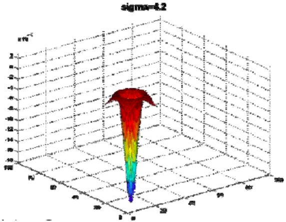
# Example

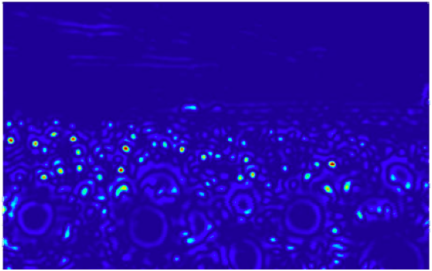


[Source: K. Grauman]

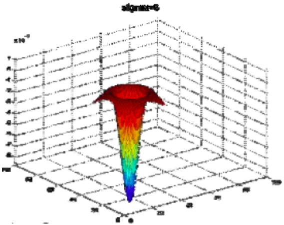
# Example

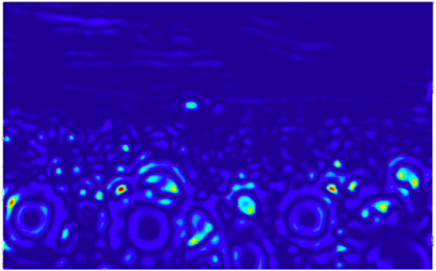




[Source: K. Grauman]

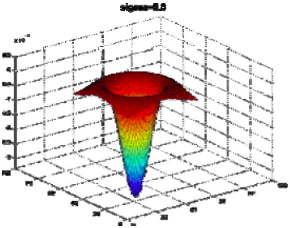
# Example

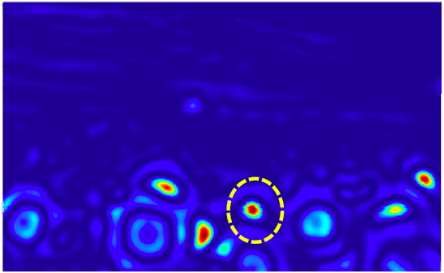




[Source: K. Grauman]

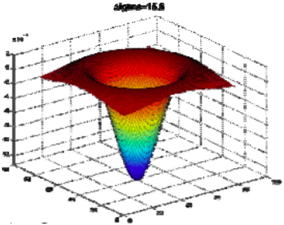
# Example

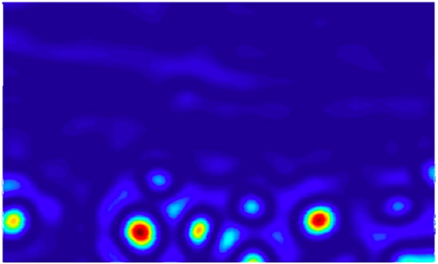




[Source: K. Grauman]

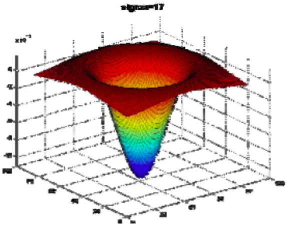
# Example

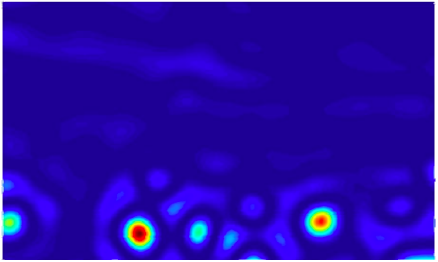




[Source: K. Grauman]

# Example

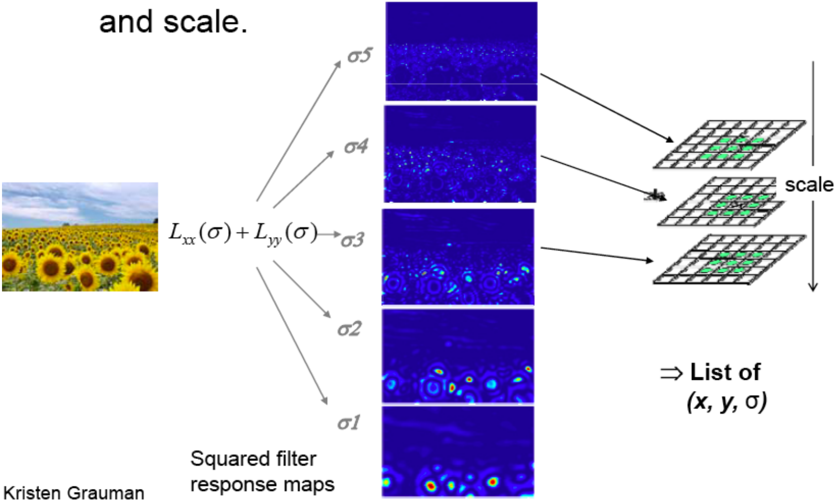




[Source: K. Grauman]

# Scale Invariant Interest Points

Interest points are local maxima in both position and scale.



# Example


[Source: S. Lazebnik]

# Blob Detection – Laplacian of Gaussian

- That's nice. But can we do faster?
- Remember again the Laplacian of Gaussian:

$$\nabla^2 g(x, y, \sigma) = \frac{\partial^2 g(x, y, \sigma)}{\partial x^2} + \frac{\partial^2 g(x, y, \sigma)}{\partial y^2}, \qquad \text{where } g \text{ is a Gaussian}$$

# Blob Detection – Laplacian of Gaussian

- That's nice. But can we do faster?
- Remember again the Laplacian of Gaussian:

$$\nabla^2 g(x, y, \sigma) = \frac{\partial^2 g(x, y, \sigma)}{\partial x^2} + \frac{\partial^2 g(x, y, \sigma)}{\partial y^2}, \quad \text{where } g \text{ is a Gaussian}$$

- So computing our interest points means two convolutions (one for each derivative) **per scale**
- Larger scale ($\sigma$), larger the filters (more work for convolution)
- Can we do it faster?

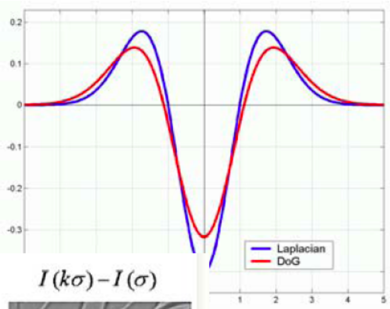
# Approximate the Laplacian of Gaussian

$$L = \sigma^2 \left( G_{xx}(x,y,\sigma) + G_{yy}(x,y,\sigma) \right)$$

(Laplacian)

$$DoG = G(x,y,k\sigma) - G(x,y,\sigma)$$

(Difference of Gaussians)

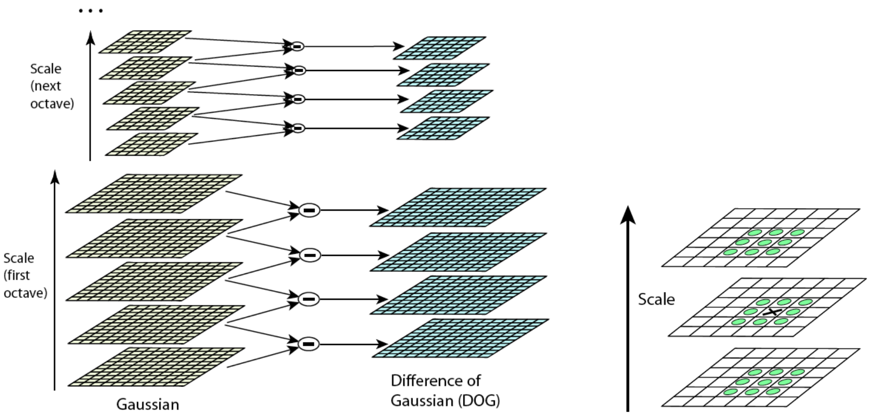



$I(k\sigma)$ $\quad$ $I(\sigma)$ $\quad$ $I(k\sigma) - I(\sigma)$


[Source: K. Grauman]

# Lowe's DoG

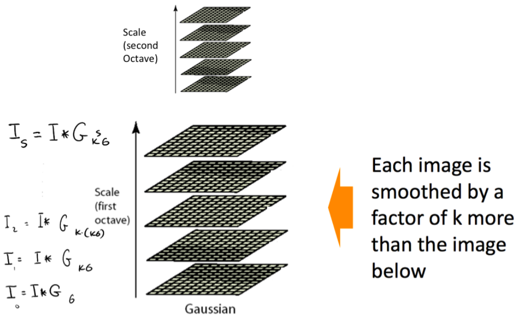
- Lowe (2004) proposed computing a set of sub-octave Difference of Gaussian filters looking for 3D (space+scale) maxima in the resulting structure




[Source: R. Szeliski, slide credit: R. Urtasun]

# Lowe's DoG

- First compute a Gaussian image pyramid

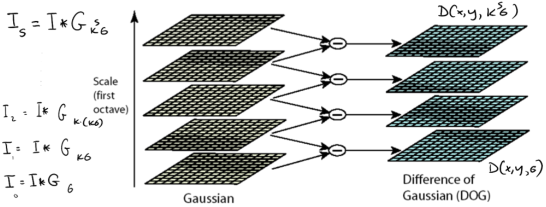



[Source: F. Flores-Mangas]

# Lowe's DoG

- First compute a Gaussian image pyramid
- Compute Difference of Gaussians

$$D(x, y, \rho) = I(x, y) * (G(x, y, k\rho) - G(x, y, \rho))$$
$$\text{for } \rho = \{\sigma,\ k\sigma,\ k^2\sigma, \ldots, k^{s-1}\sigma\}, \quad k = 2^{1/s}$$

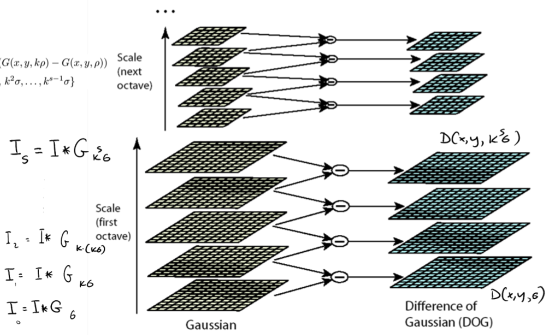



[Source: F. Flores-Mangas]

# Lowe's DoG

- First compute a Gaussian image pyramid
- Compute Difference of Gaussians
- At every scale

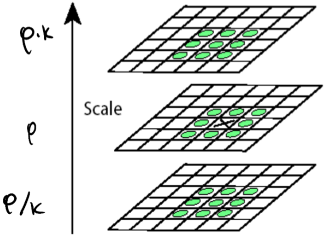
$$D(x,y,\rho) = I(x,y) * (G(x,y,k\rho) - G(x,y,\rho))$$
$$\text{for } \rho = \{\sigma, k\sigma, k^2\sigma, \ldots, k^{s-1}\sigma\}$$




[Source: F. Flores-Mangas]

# Lowe's DoG

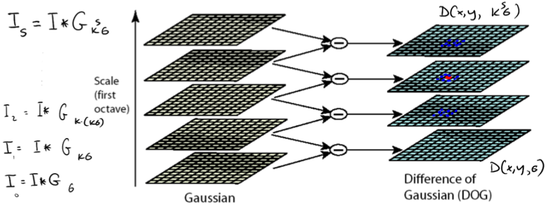
- First compute a Gaussian image pyramid
- Compute Difference of Gaussians
- At every scale
- Find local maxima in scale
- A bit of pruning of bad maxima and we're done!




[Source: F. Flores-Mangas]

# Lowe's DoG

- First compute a Gaussian image pyramid
- Compute Difference of Gaussians
- At every scale
- Find local maxima in scale
- A bit of pruning of bad maxima and we're done!




[Source: F. Flores-Mangas]

# Other Interest Point Detectors (Many Good Options!)

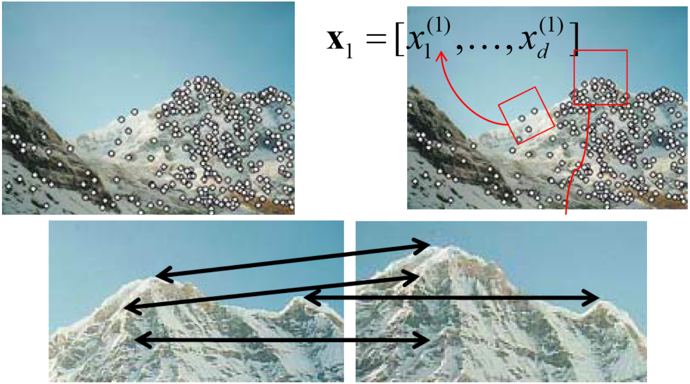
- Lindeberg: Laplacian of Gaussian
- Lowe: DoG (typically called the SIFT interest point detector)
- Mikolajczyk & Schmid: Hessian/Harris-Laplacian/Affine
- Tuyttelaars & Van Gool: EBR and IBR
- Matas: MSER
- Kadir & Brady: Salient Regions

# Summary – Stuff You Should Know

- To match the same scene or object under different viewpoint, it's useful to first detect **interest points** (keypoints)
- We looked at these interest point detectors:
  - Harris corner detector: translation and rotation but not scale invariant
  - Scale invariant interest points: Laplacian of Gaussians and Lowe's DoG
- Harris' approach computes $I_x^2$, $I_y^2$ and $I_x I_y$, and blurs each one with a Gaussian. Denote with: $A = g * I_x^2$, $B = g * (I_x I_y)$ and $C = g * I_y^2$. Then $M_{xy} = \begin{pmatrix} A(x,y) & B(x,y) \\ B(x,y) & C(x,y) \end{pmatrix}$ characterizes the shape of $E_{WSSD}$ for a window around $(x, y)$. Compute "cornerness" score for each $(x, y)$ as $R(x, y) = \det(M_{xy}) - \alpha \, \mathrm{trace}(M_{xy})^2$. Find $R(x, y) > \mathrm{threshold}$ and do non-maxima suppression to find corners.
- Lowe's approach creates a Gaussian pyramid with $s$ blurring levels per octave, computes difference between consecutive levels, and finds local extrema in space and scale

# Local Descriptors – Next Time

- **Detection**: Identify the interest points.
- **Description**: Extract a feature descriptor around each interest point.
- **Matching**: Determine correspondence between descriptors in two views.




[Source: K. Grauman]