

# Feature Detectors



## Jiri Matas

Center for Machine Perception,  
Faculty of Electrical Engineering  
Czech Technical University,



## Krystian Mikolajczyk

University of Surrey,  
Guildford, UK



### Slide credits:

- Michal Perdoch (special thanks)
- Darya Frolova, Denis Simakov, The Weizmann Institute of Science
- Martin Urban , Stepan Obdrzalek, Ondra Chum, Dmitro Mishkin, Karel Lenc, Jan Sochman  
Center for Machine Perception Prague
- Matthew Brown, David Lowe, University of British Columbia
- Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, ETH Zurich, KU Leuven
- Noah Snavely, Cornell University

## 1. The Classics.

- methods searching the scale space (Harris, Laplace, Hessian, ...)
- MSERs
- EBRs and line-based and simplex-based distinguished regions

## 2. Talking about Speed.

- SURF
- Learning to Emulate a Detector

## 3. New kids on the block.

## 4. ASIFT: a detector ??

# Widely used feature detectors



## Objectives :

- Location covariance
  - Find distinguished locations, a region must be *at least* distinguishable from *all* its neighbours
  - Choose an operator (e.g. based on 1<sup>st</sup>, 2<sup>nd</sup> order derivatives)
  - Find local extrema
- Scale covariance
  - build a scale space and apply the operator for each scale level,
  - Choose a region in scale space
  - find local extrema
- Affine covariance
  - Find affine shape (ellipse) of the local pattern
    - Segmentation
    - Iterative adaptation

# Harris detector (Harris, 1988)

citations

5400 (2010)

6900 (2012)



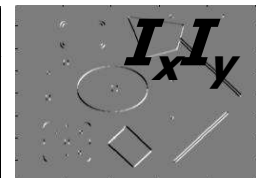
- Second moment matrix / autocorrelation matrix

$$\mu(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

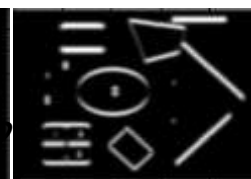
**1. Image derivatives**



**2. Square of derivatives**



**3. Gaussian filter**  $g(\sigma_I)$



**Cornerness function – both eigenvalues are strong**

$$\begin{aligned} har &= \det[\mu(\sigma_I, \sigma_D)] - \alpha[\text{trace}(\mu(\sigma_I, \sigma_D))] = \\ &= g(I_x^2)g(I_y^2) - [g(I_x I_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2 \end{aligned}$$

**Non-maxima suppression**



*har*



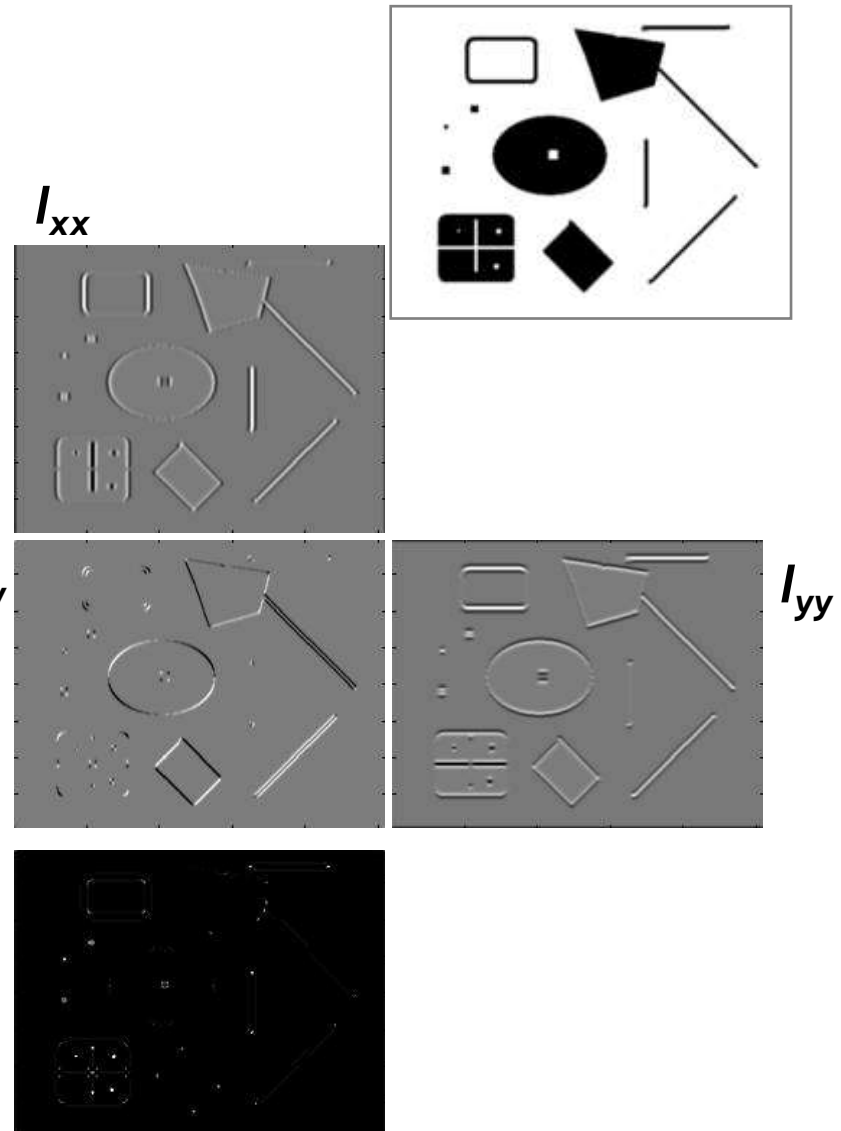
- Hessian determinant

$$\text{Hessian}(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

$$\det(\text{Hessian}(I)) = I_{xx}I_{yy} - I_{xy}^2$$

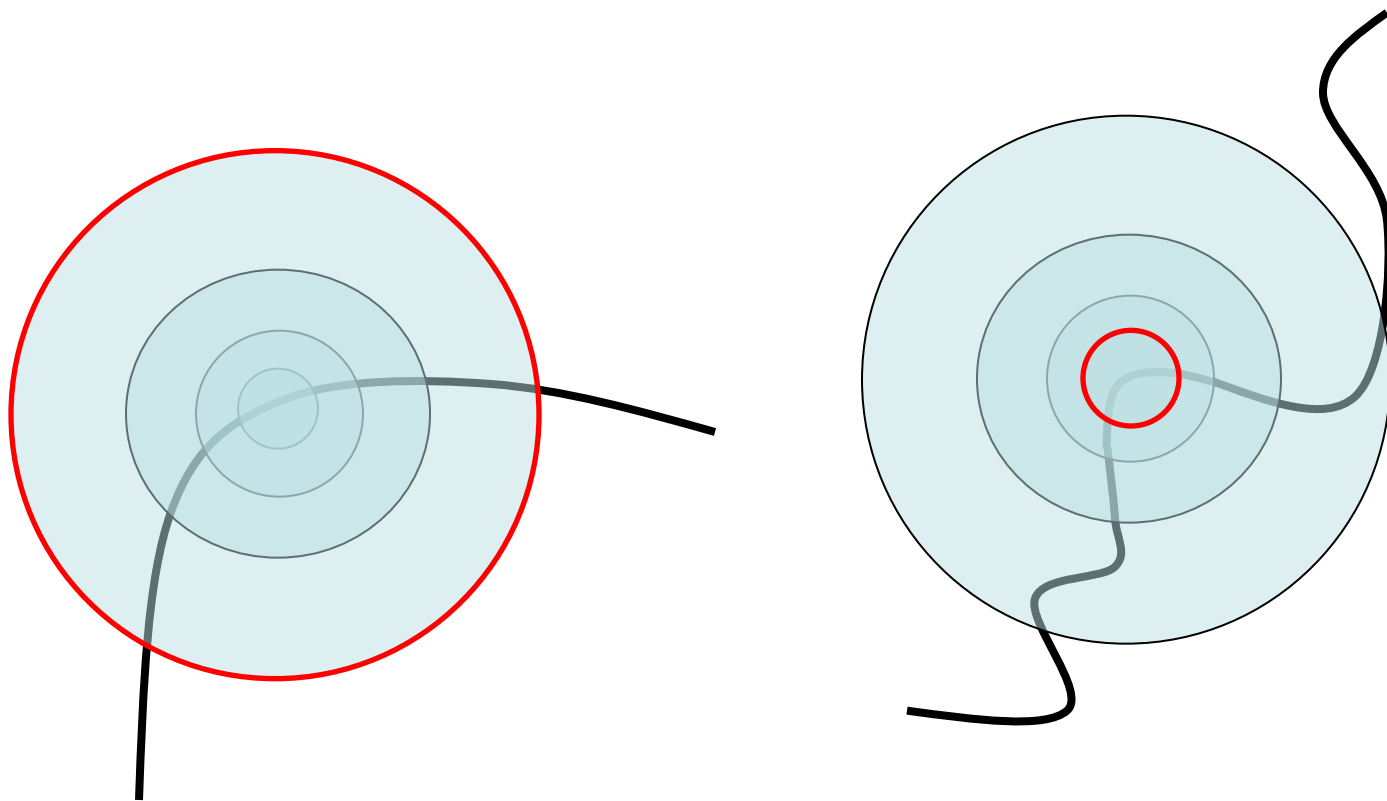
In Matlab:

$$I_{xx}.*I_{yy} - (I_{xy})^2$$



# Scale Covariant Detection

- Not invariant to *image scale*!
- The problem: how do we choose corresponding circles *independently* in each image?



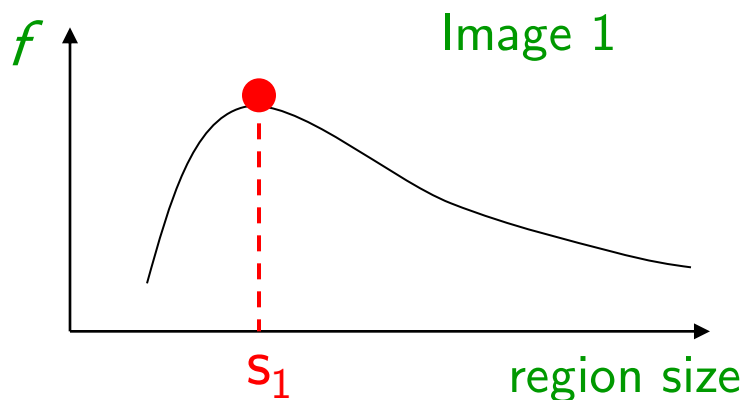
# Scale Covariant Detection


- Common approach:

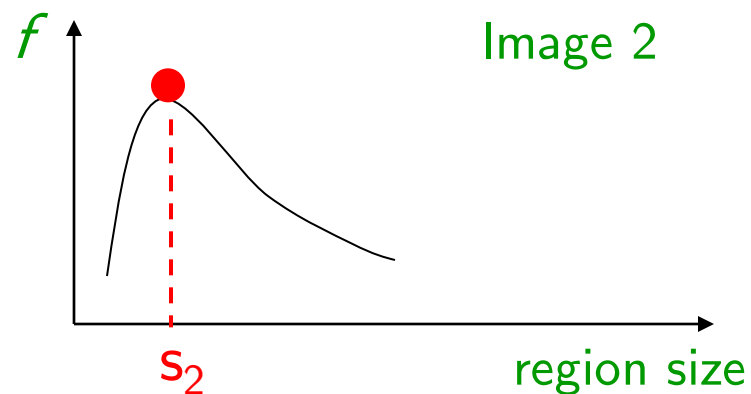
Take a local maximum of some function

Observation: region size, for which the maximum is achieved, should be *invariant* to image scale.

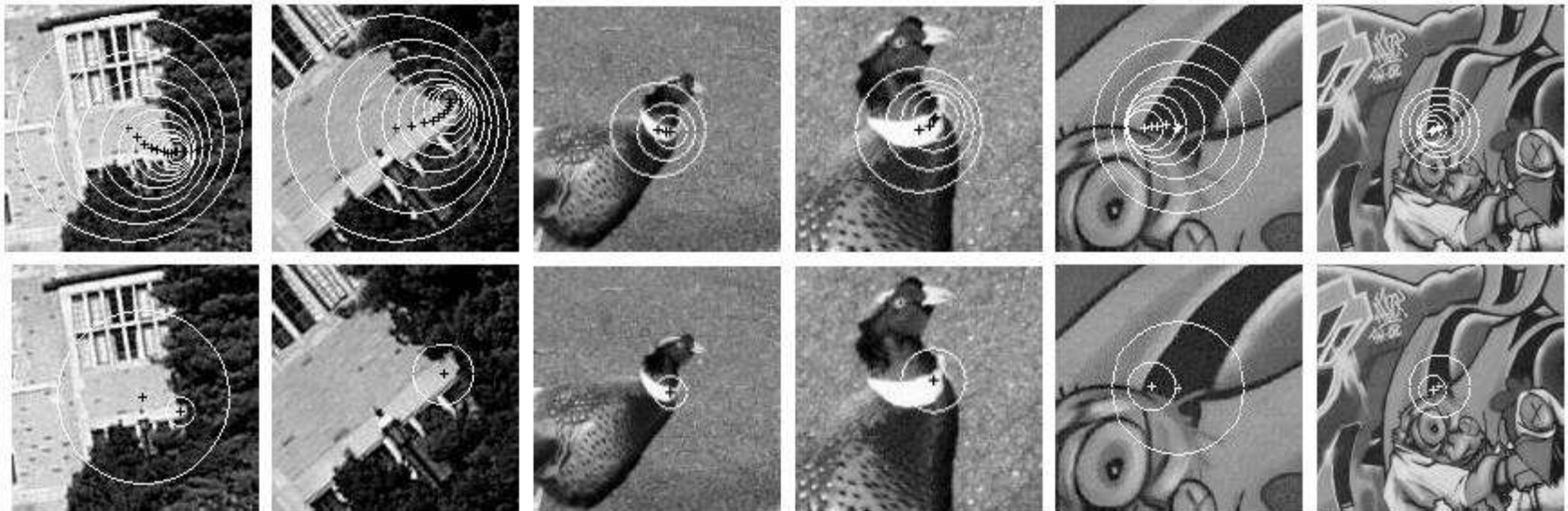
Important: this scale invariant region size is found in each image *independently*.



scale = 1/2  




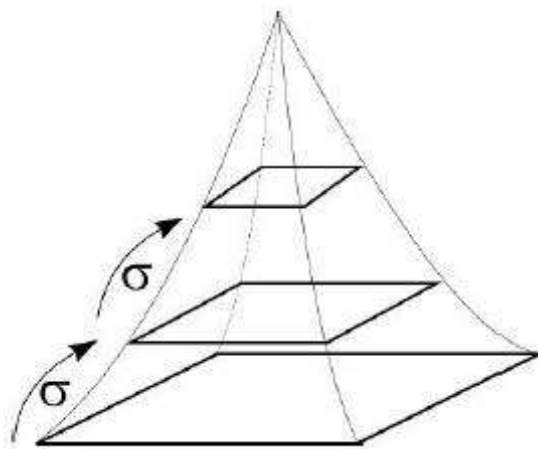
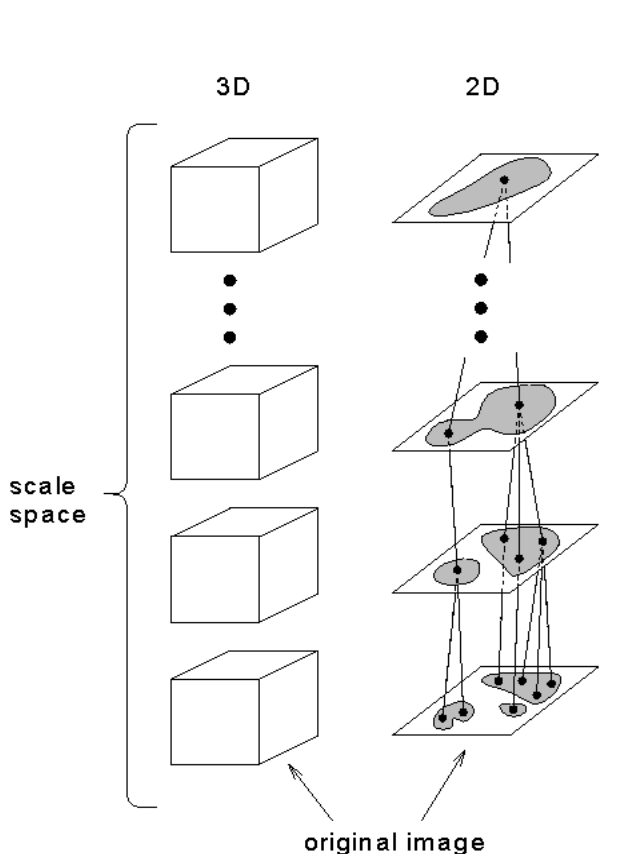
1. Initialization: Multiscale Harris corner detection
2. Scale selection based on Laplacian



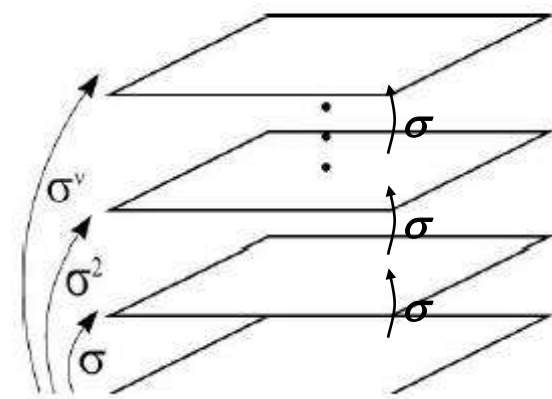


# Scale-space representation

- Exponential increase of kernel size - uniform information change
- Kernel size and sampling interval are directly related.



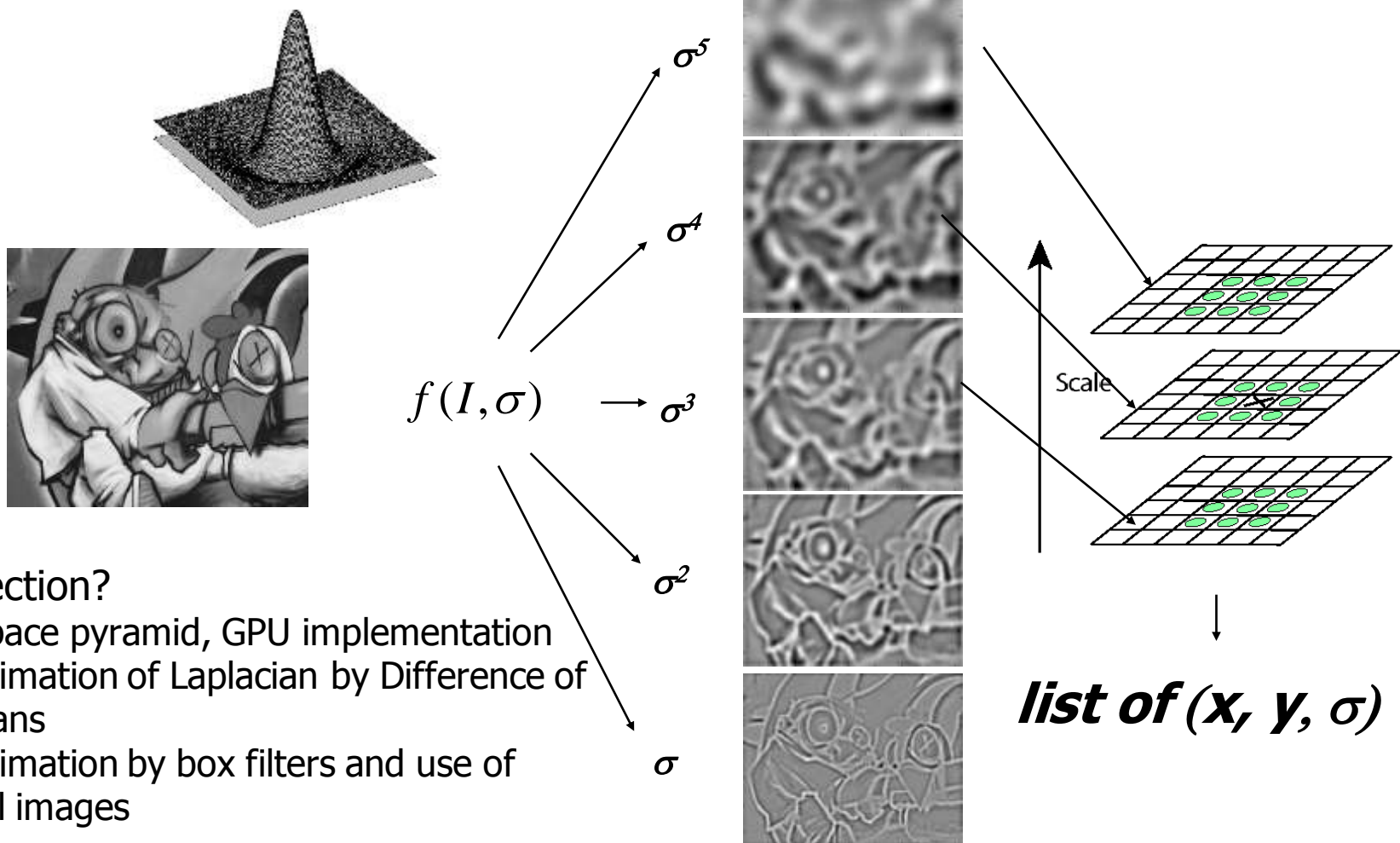
Convolution with kernel of constant size and sampling



Convolution with kernel of increasing size

# Scale invariant detectors

- Scale invariant detectors, find local extrema (both in space and scale) of **Laplacian** and **determinant of Hessian** response in gaussian scalespace.

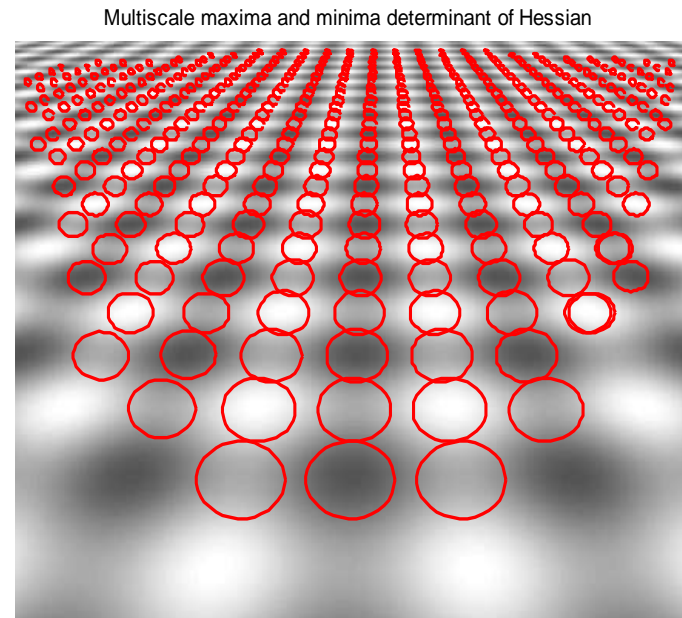
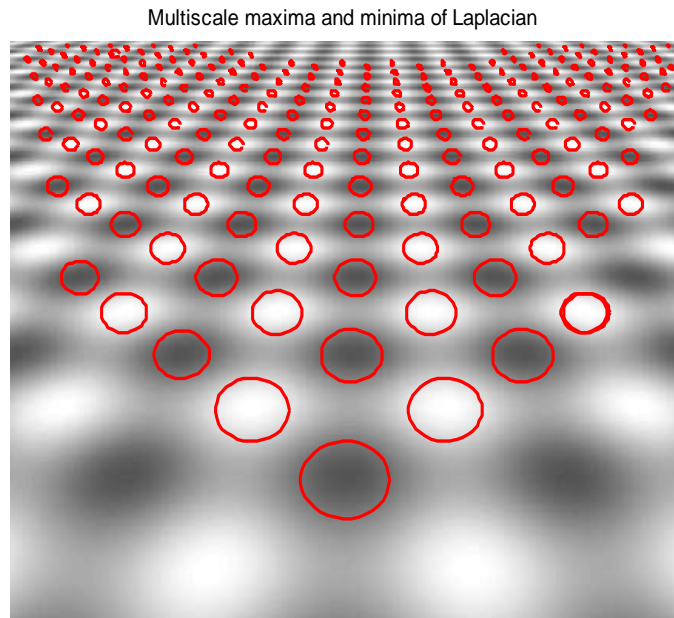


Faster detection?

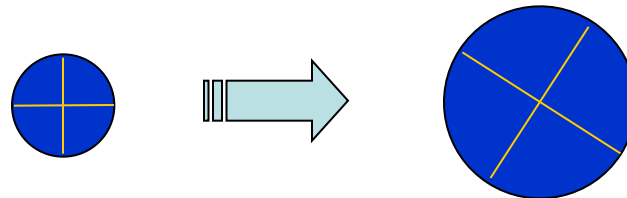
Scalespace pyramid, GPU implementation  
Approximation of Laplacian by Difference of Gaussians  
Approximation by box filters and use of integral images

# Properties

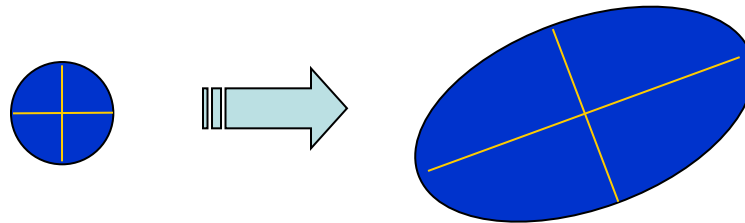
- 😊 scale-invariant
- 😊 simple, efficient scheme
- 😐 laplacian fires more on edges than determinant of hessian



- Above we considered:  
Similarity transform (rotation + uniform scale)



- Now we go on to:  
Affine transform (rotation + non-uniform scale)



# 1D Affine invariance [Tell & Carlsson 00], [Matas & Burianek 00]



- 2D affine transform maps a line to a line

Idea:

- take pairs of points, match intensity profiles along the line segments

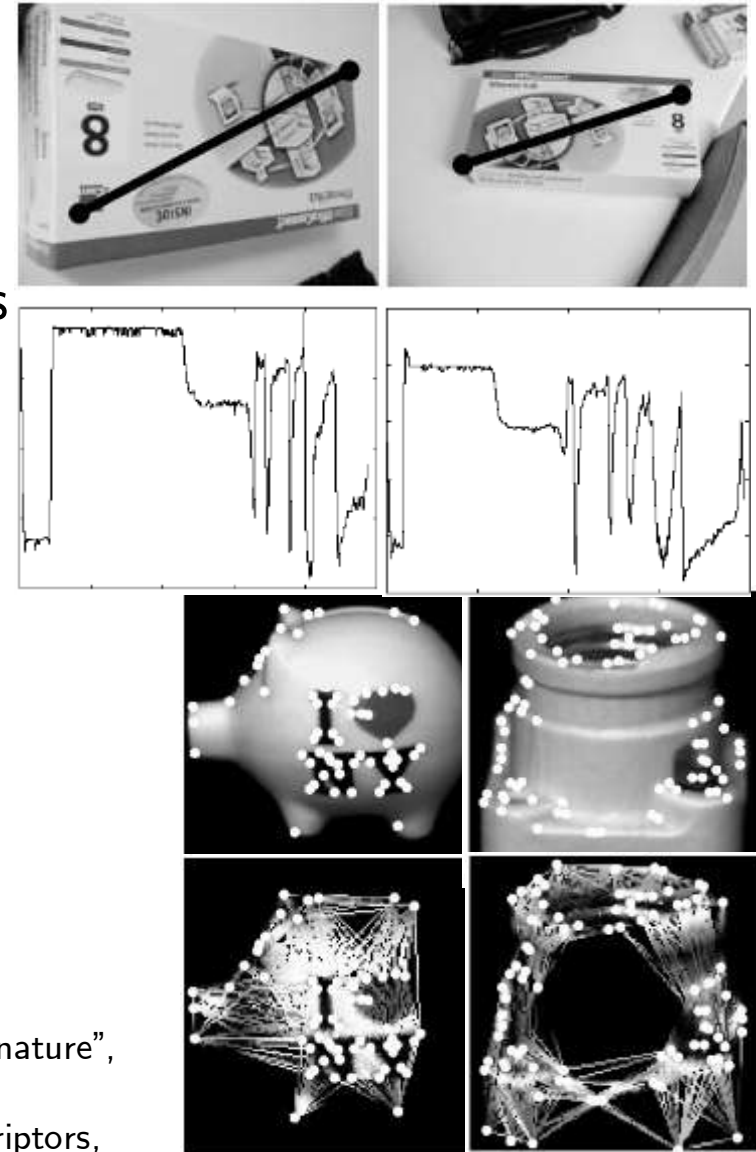
Problems:

- which pairs of points to choose?
- the pre-images of the line segments must be planar
- line segment pre-image is often on discontinuities

Tell, Carlsson: "Wide Baseline Point Matching Using Affine Invariants Computed from Intensity Profiles", ECCV00

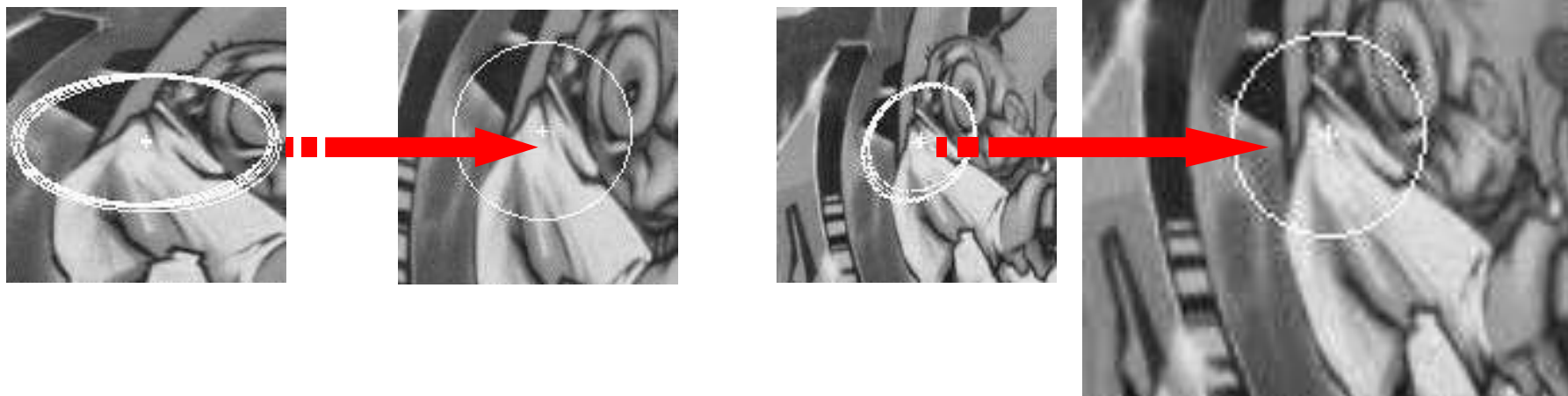
Matas, Burianek: "Object Recognition using the Invariant Pixel-Set Signature", BMVC2000

Hundelshausen, [Sukthankar](#), D-Nets: Beyond Patch-Based Image Descriptors, CVPR12



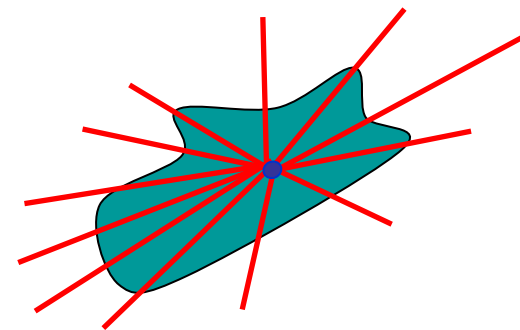
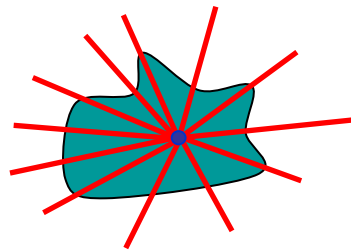
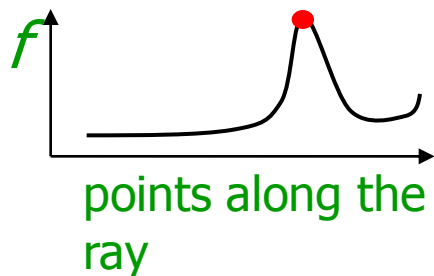
# Affine Normalization

- Scale, stretch, skew and rotate
- Fixed size disk



# Affine Covariant Detection

- Take a local intensity extremum as initial point
  - Go along every ray starting from this point and stop when extremum of function  $f$  is reached
- citations  
365 (2010)  
420 (2012)



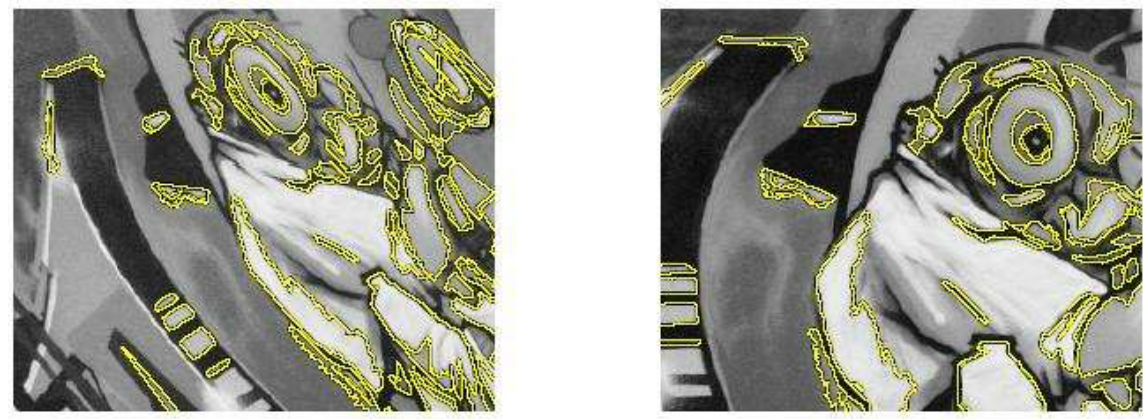
T.Tuytelaars, L.V.Gool. "Wide Baseline Stereo Matching Based on Local, Affinely Invariant Regions". BMVC 2000.



# Maximally Stable Extremal Regions (Matas et al 2002)

- Based on watershed algorithm
  - Consecutive image thresholding by all thresholds
  - Maintain list of Connected Components
  - Regions = Connected Components with stable area (or some other property) over multiple thresholds selected

citations  
 1300 (2010)  
 1620 (2012)





# Properties



Affine invariant



Simple, efficient scheme



High repeatability



Fires on similar features as IBR

(regions need not be convex, but need to be closed)

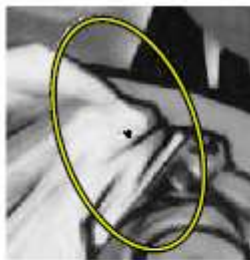


Sensitive to image blur

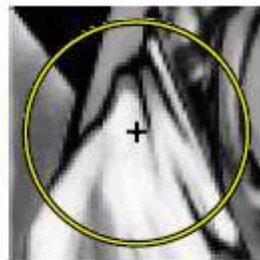
# SMM Affine Adaptation (Lindeberg-Baumberg-Garding)

1. Detect initial region with Harris or Hessian detector and select the scale.
2. Estimate the shape with the second moment matrix
3. Normalize the affine region to the circular one
4. Go to step 2 if the eigenvalues of the second moment matrix for new point are not equal.

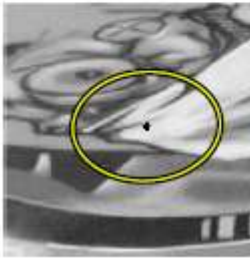
$$\mu(\sigma_I, \sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$



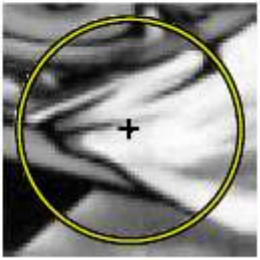
$$\mathbf{x}_L \longrightarrow M_L^{-1/2} \mathbf{x}'_L$$



$$\mathbf{x}'_L \longrightarrow R \mathbf{x}'_R$$



$$\mathbf{x}_R \longrightarrow M_R^{-1/2} \mathbf{x}'_R$$



# Properties

Scale or affine invariant

Detects blob and corner-like structures



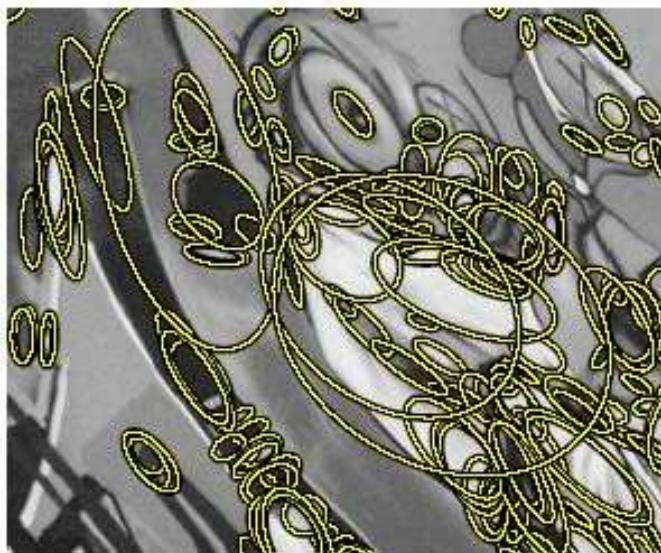
large number of regions



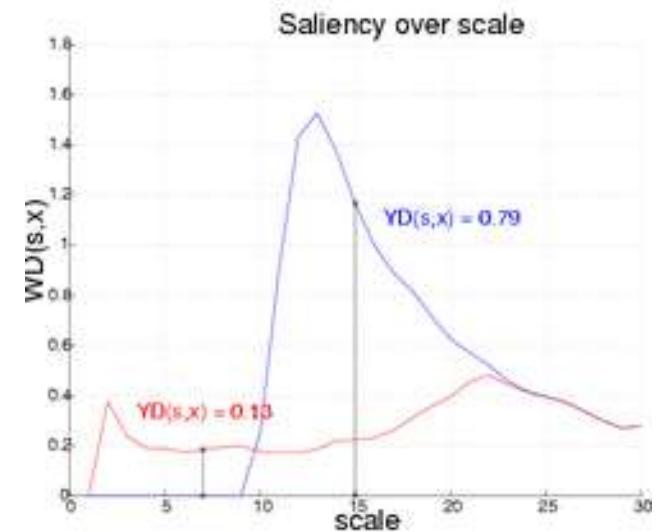
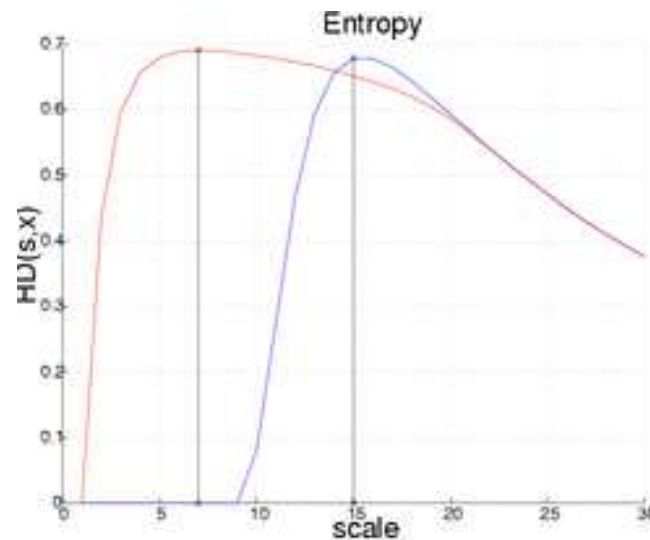
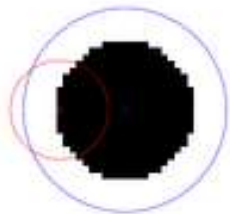
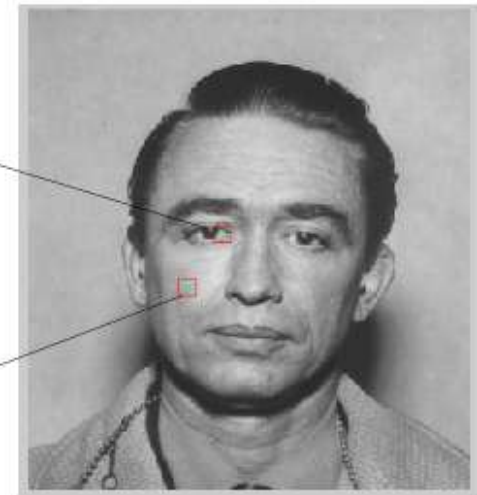
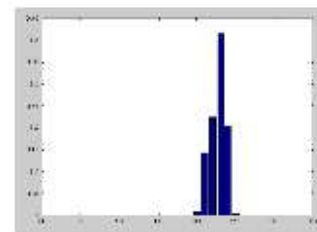
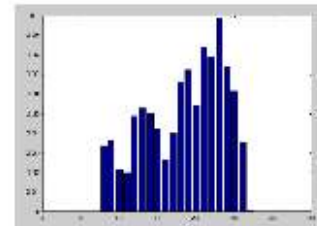
well suited for object class recognition



less accurate than some competitors



- Maxima in entropy
  - combined with inter-scale saliency
  - Extended to affine invariance





# Properties

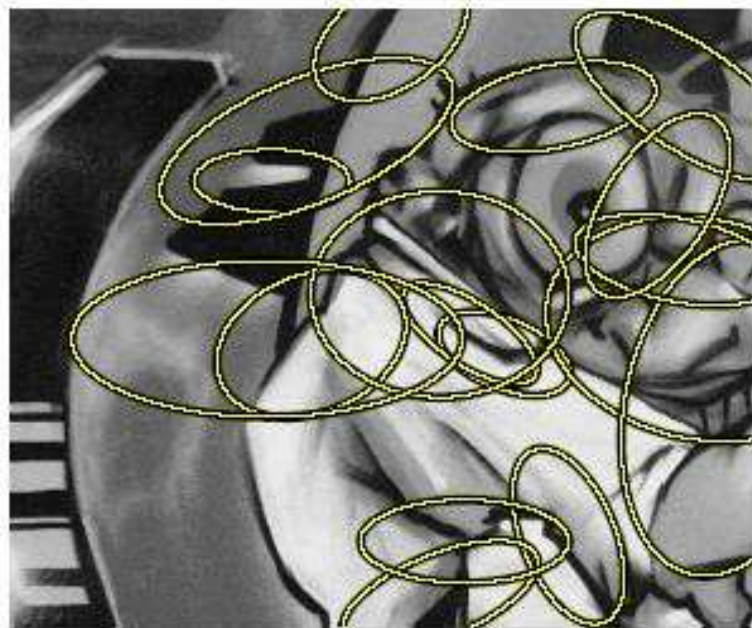
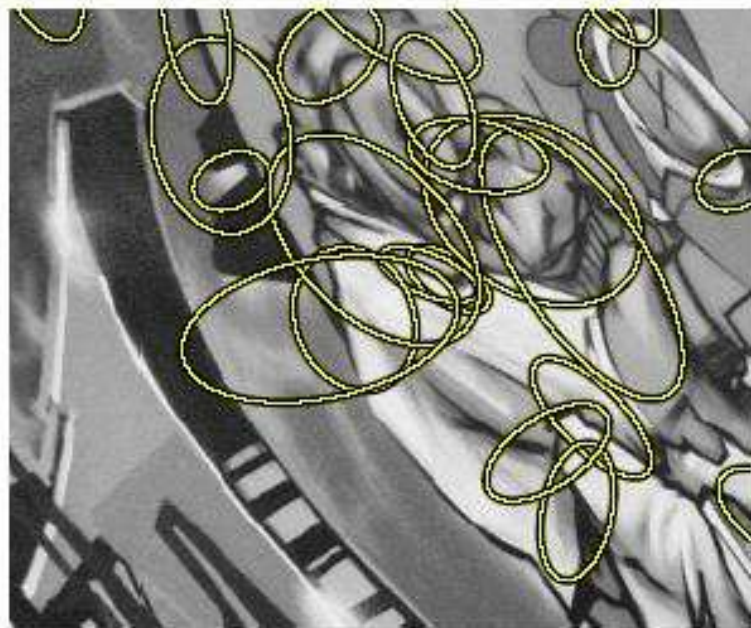
Scale or affine invariant

Detects blob-like structures

😊 very good for object class recognition

😐 limited number of regions

😞 slow to extract



# Talking about Speed

1. Consider that descriptor calculation may take longer than the detection process! Sometimes, “auxiliary calculations” like non-maximum suppression dominates computation time.
2. Consider the required level of invariance:  
in some applications, reduced level invariance is sufficient
3. Consider fast approximations
4. Use fast implementations, e. g. on GPU (GPU SURF, GPU SIFT)

# SURF: Speeded Up Robust Features



## Idea:

- Approximate Hessian + SIFT calculation with a computationally efficient algorithm.

## Properties:

citations

2300 (2010)

4000 (2012)

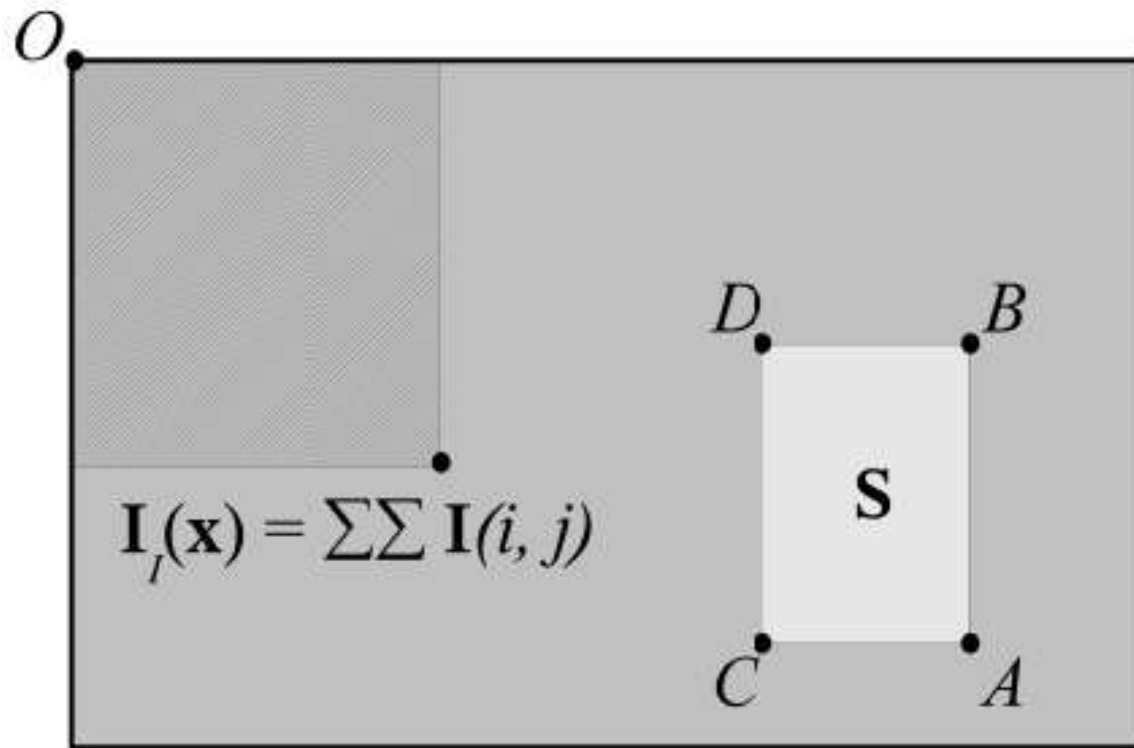
- exploit the integral image
- the SURF detector is an approximation to the Hessian
- reuse the calculations needed for detection in descriptor computation
- maintain robustness to rotation, scale illumination change
- approximately 2x faster than DoG  
10x faster Hessian-Laplace detector

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool , SURF: Speeded Up Robust Features, ECCV 2006.

ECCV 2012 Modern features: ... Detectors.



# The Integral image (Sum Table)



$$S = A - B - C + D$$

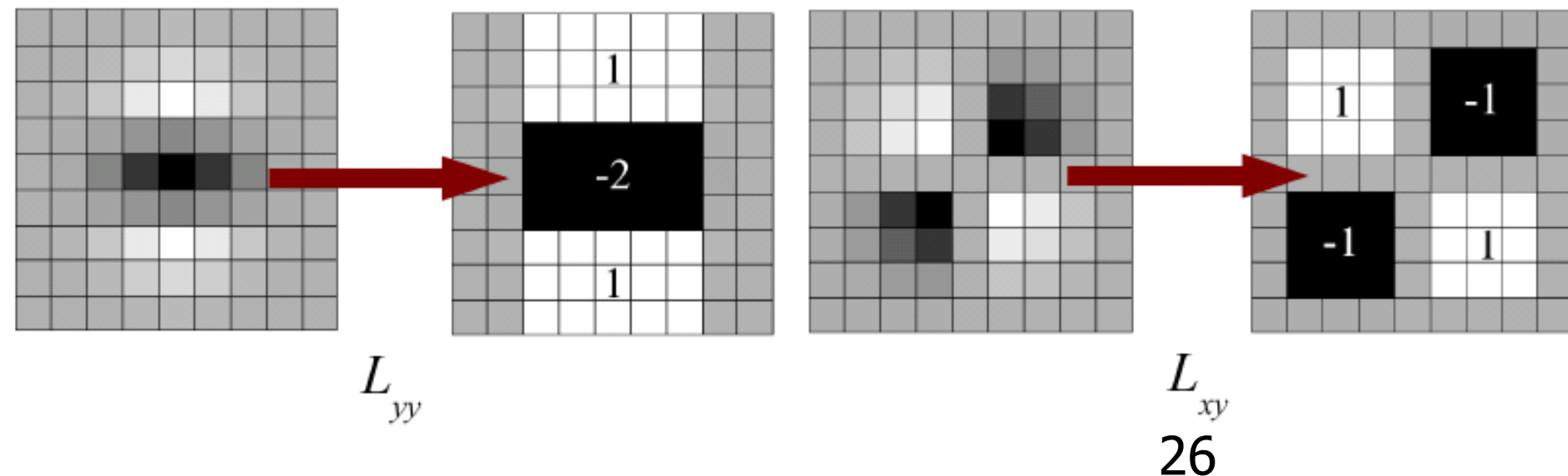
To calculate the sum in the DBCA rectangle, only 3 additions are needed

# SURF Detection

- Hessian-based interest point localization:
- $L_{xx}(x,y,\sigma)$  is the convolution of the *Gaussian* second order derivative with the image

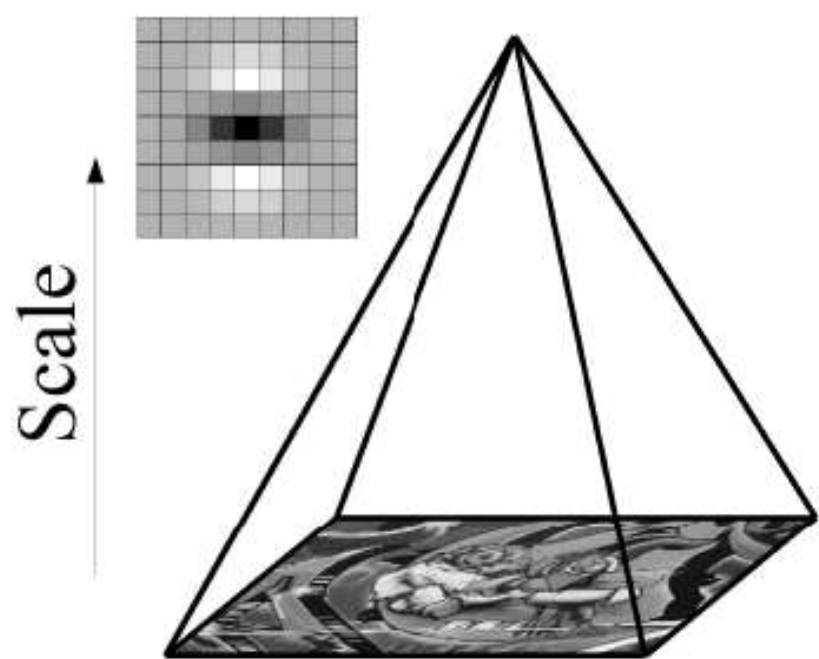
$$H = \begin{bmatrix} L_{xx} & L_{xy} \\ L_{xy} & L_{yy} \end{bmatrix}$$

- Approximate second order derivatives with box filters

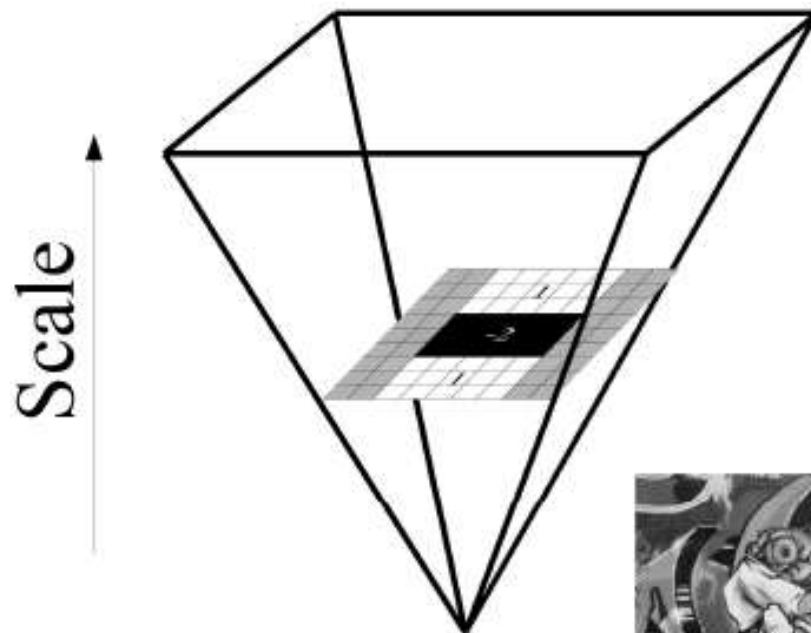


# SURF Detection

- Scale analysis easily handled with the integral image



$9 \times 9, 15 \times 15, 21 \times 21, 27 \times 27 \rightarrow$   
 $1^{st} \text{ octave}$

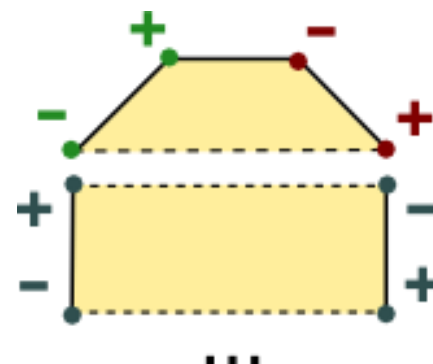


$39 \times 39, 51 \times 51 \dots$   
 $2^{nd} \text{ octave}$

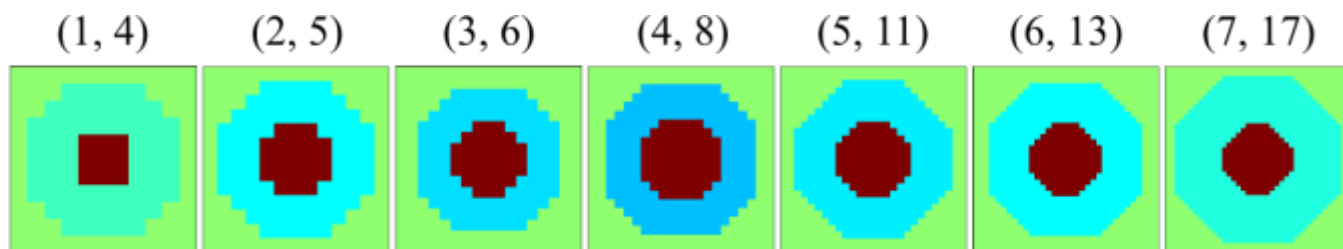


# CenSurE-Oct detector

- Approximation of LoG / DoG by octagonal box filters
- Sum of intensities inside an octagon calculated in  $O(1)$  using 3 integral images:
- Zero DC response requires normalisation
- Constant DC response over scales
- 3x3x3 non-maxima suppression
- Edge responses suppressed using Harris measure
- Scale sampling:



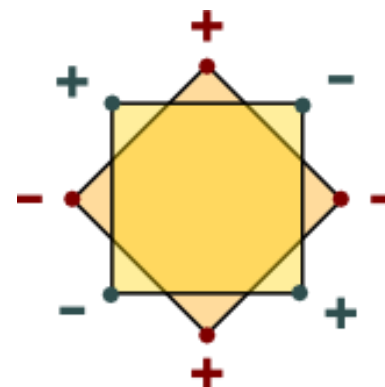
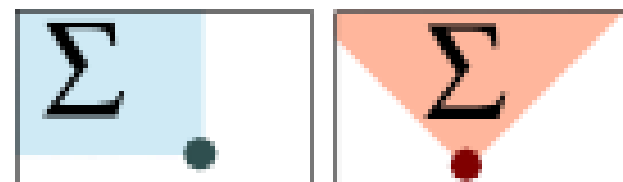
(Inner, outer)  
radius in pixels



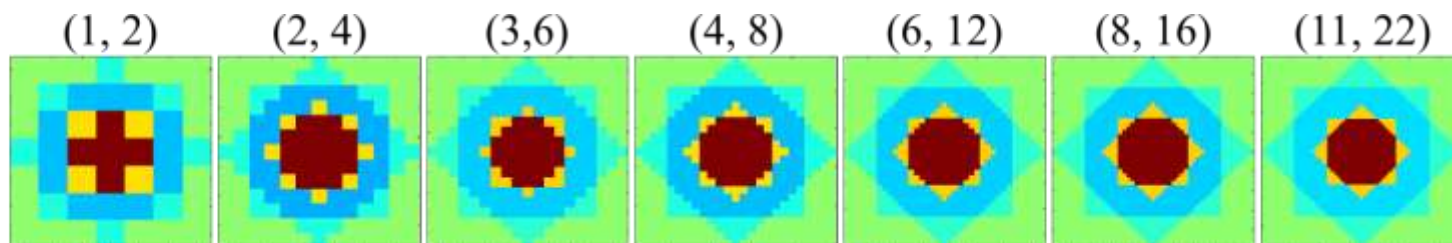
M. Agrawal, K. Konolige, M. R. Blas. CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching ECCV 2008

# STAR detector

- Approximating LoG / DoG using 2 integral images
- 5x5 spatial non-max suppression
- A single response at a point: maximum over scales
- Edge responses suppressed using Hessian



(Inner, outer)  
radius in pixels



K. Konolige et al. View-Based Maps. Journal of Robotics 2010

[http://pr.willowgarage.com/wiki/Star\\_Detector](http://pr.willowgarage.com/wiki/Star_Detector)

# CenSurE and STAR det. vs. DoG

- Small scale features: Only integer locations and scales
- Limited rotation invariance
- Only 2x speedup for the DoG detector

Det. Time [s]	Without SSE instructions	With SSE instructions
VLF SIFT (DoG)	0.34	0.14
STAR	0.16	0.08
CenSurE	0.28	X

# Fast Emulation of A Binary Decision Process



## The Idea

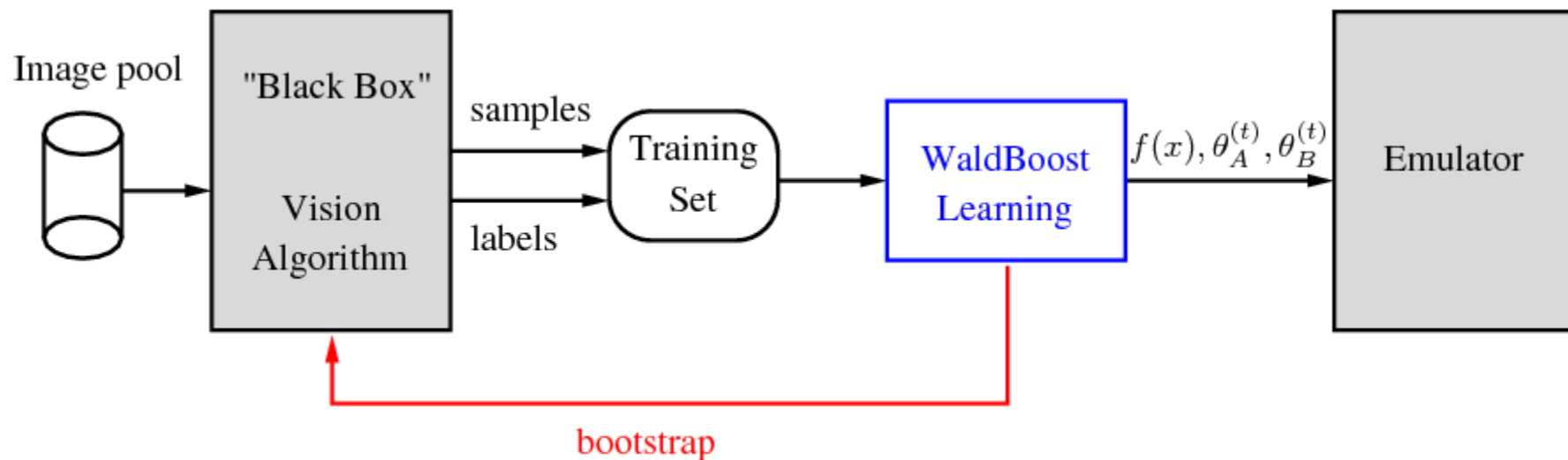
- Given a black box algorithm **A** performing a useful binary (decision) task
- Train a sequential classifier **S** to (approximately) emulate output of algorithm **A** while minimizing time-to-decision
- Allow user to control quality of the approximation

## Advantages

- Instead of spending man-months on code optimization, choose relevant feature class and train sequential classifier **S**
- A (slow) Matlab code can be speeded up this way!

- [1] J. Sochman and J. Matas.  
Waldboost - Learning For Time Constrained Sequential Detection. CVPR 2005
- [2] J. Sochman and J. Matas.  
Learning fast emulators of binary decision processes. IJCV 83(1):149 - 163, 2009.

# Black-box Generated Training Set



- WaldBoost Combines AdaBoost training (which selects measurements) with Wald's sequential decision making theory (for time quasi-optimal decisions)
- The Emulator approximates the behavior of the black-box algorithm
- The black-box algorithm can potentially provide almost unlimited number of training samples
  - Efficiency of training is important
  - Suitable for incremental or online methods



# Emulation of Hessian-Laplace and Kadir Detectors

## Motivation

- Hessian-Laplace is close to state of the art
- Kadir's detector very slow (100x slower than Difference of Gaussians)
- Reference: SURF is a fast implementation of SIFT, natural competitor
- Executables of both methods from robots.ox.ac.uk
- Both detectors are scale-invariant which is easily implemented via a scanning window + a sequential test

## Implementation

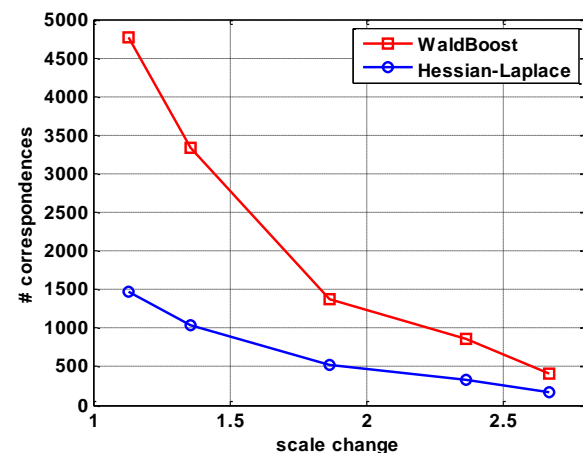
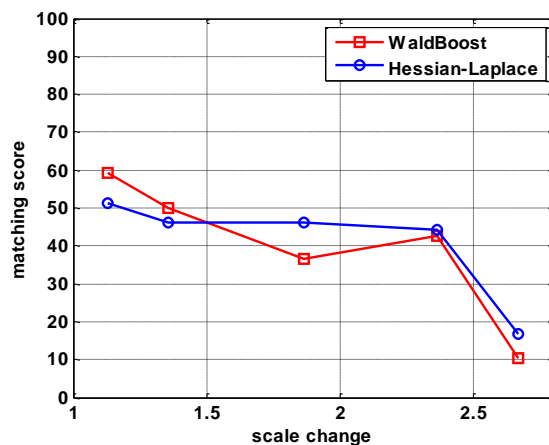
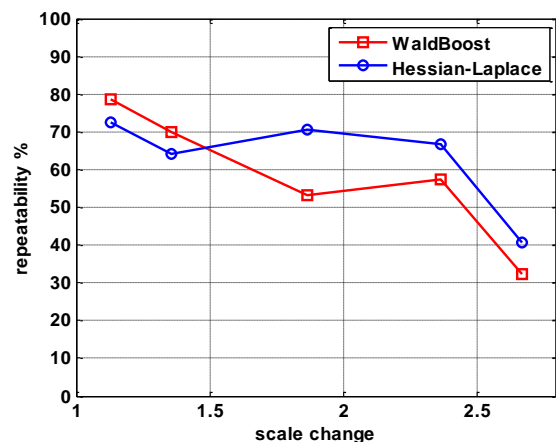
- Choice of  $\mathcal{H}$ : various filters computable with integral images - difference of rectangular regions, variance in a window, ...
- Positive samples: Patches twice the size of original interest point scale

[1] K. Mikolajczyk, C. Schmid.

Scale and Affine Invariant Interest Point Detectors. ICCV 2004.

[2] T. Kadir, M. Brady. Saliency, Scale and Image Description. IJCV 2001.

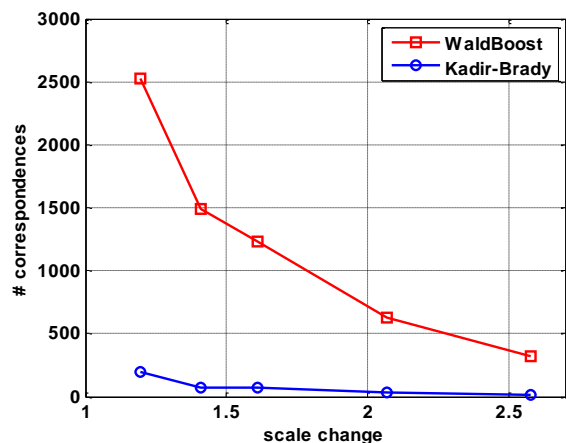
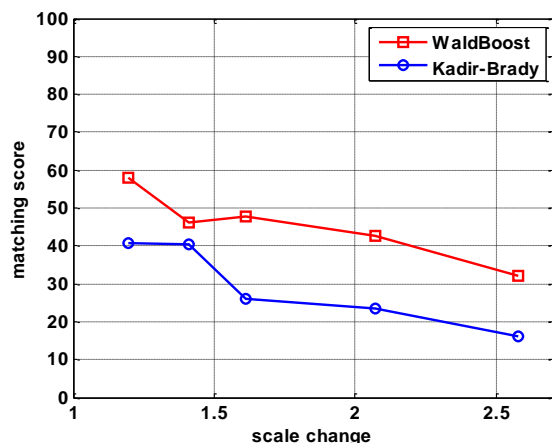
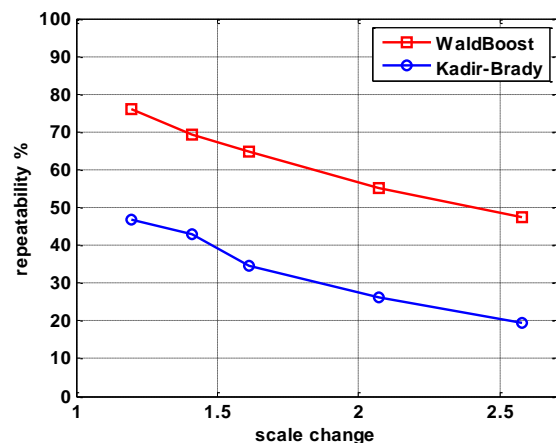
# Results: Hessian-Laplace - Boat sequence



- Repeatability comparable (left graph)
- Matching score almost identical (middle graph)
- Higher number of correspondences and correct matches in WaldBoost.
- Speed comparison (850x680 image)

Original	0.9 sec
WaldBoost	0.1 sec

# Results: Kadir-Brady - east-south sequence



- Repeatability slightly higher (left graph)
- Matching score slightly higher (middle graph)
- Higher number of correspondences and correct matches in WaldBoost.
- Speed comparison (850x680 image)

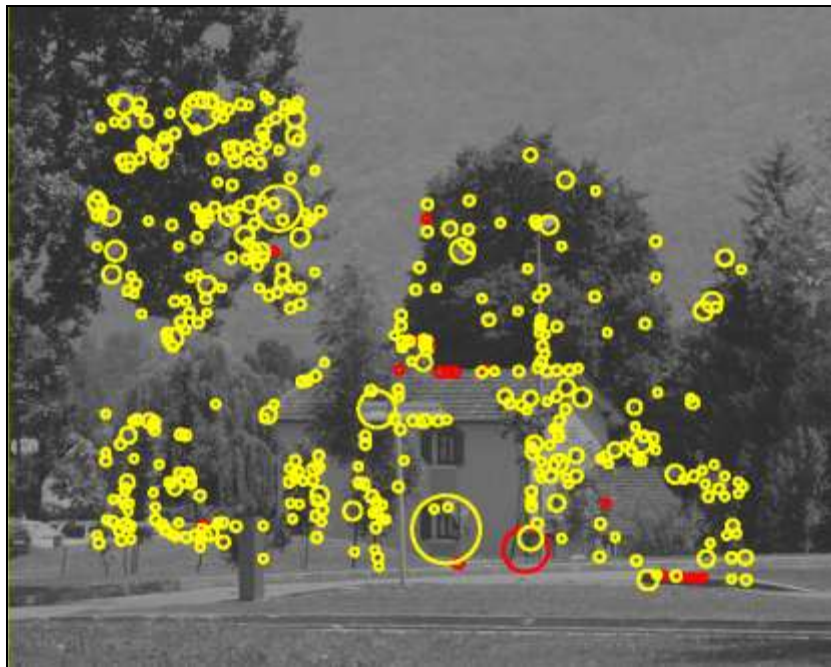
Original

1 min 48 sec

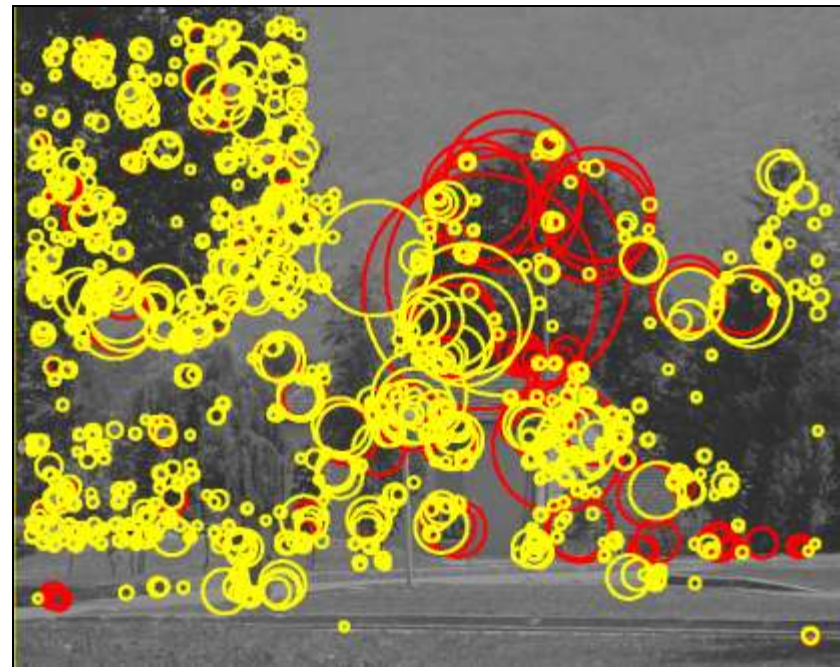
WaldBoost

0.76 sec

# WaldBoost Emulator: Approximation Quality



Kadir-Brady: 96% coverage



Hessian-Laplace: 85% coverage

**Yellow circles:** repeated detections

**Red circles:** original detections not found by the WaldBoost detector

# WaldBoost Emulator: The Speed Up

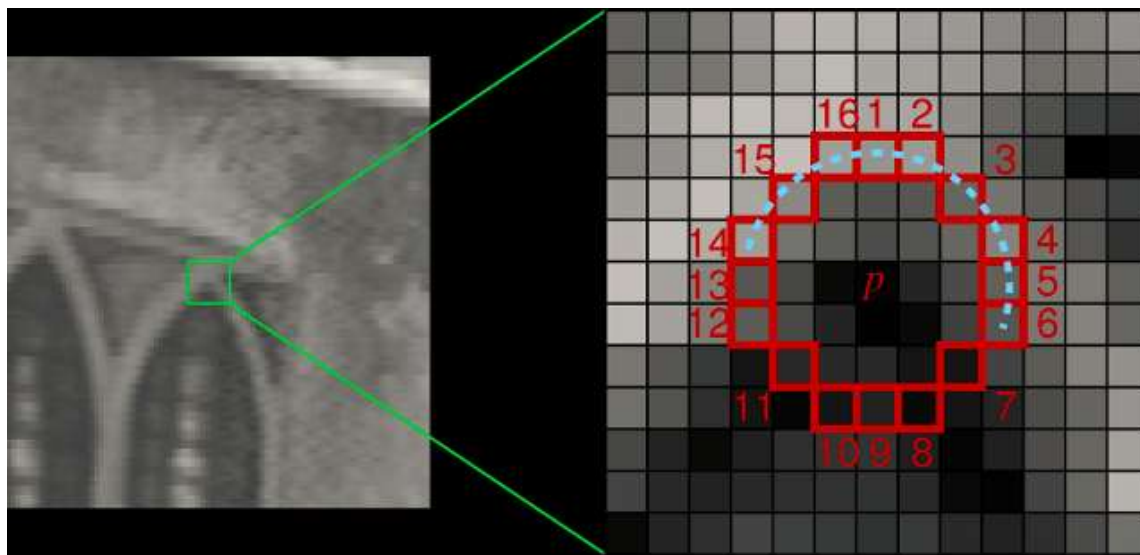


	Hessian Laplace	Kadir Brady
<b>original</b>	0.9s	1m 48s
<b>SURF</b>	0.09s	—
speed-up	10×	—
$T_{s^*}$	3	—
<b>WaldBoost</b>	0.10s	0.76s
speed-up	9×	142×
$T_{s^*}$	1.7	2.2

citations  
140 (2012)

# Fast-9 and Fast-ER (E. Rosten)

- in some situations (controlled lighting, tracking), invariance/robustness is less important than speed
- simple detector based on intensity comparisons could be very fast, and yet “repeatable enough”



citations:  
730 (2012)

- detection: 12 contiguous pixels are darker/brighter than the central pixel by at least  $t$ .
- <http://www.edwardrosten.com/work/fast.html>



# Fast-9 and Fast-ER (E. Rosten)

Detector	Opteron 2.6GHz		Pentium III 850MHz	
	ms	%	ms	%
Fast $n = 9$ (non-max suppression)	1.33	6.65	5.29	26.5
Fast $n = 9$ (raw)	1.08	5.40	4.34	21.7
Fast $n = 12$ (non-max suppression)	1.34	6.70	4.60	23.0
Fast $n = 12$ (raw)	1.17	5.85	4.31	21.5
Original FAST $n = 12$ (non-max suppression)	1.59	7.95	9.60	48.0
Original FAST $n = 12$ (raw)	1.49	7.45	9.25	48.5
Harris	24.0	120	166	830
DoG	60.1	301	345	1280
SUSAN	7.58	37.9	27.5	137.5

**Table 1.** Timing results for a selection of feature detectors run on fields ( $768 \times 288$ ) of a PAL video sequence in milliseconds, and as a percentage of the processing budget per frame. Note that since PAL and NTSC, DV and 30Hz VGA (common for webcams) have approximately the same pixel rate, the percentages are widely applicable. Approximately 500 features per field are detected.

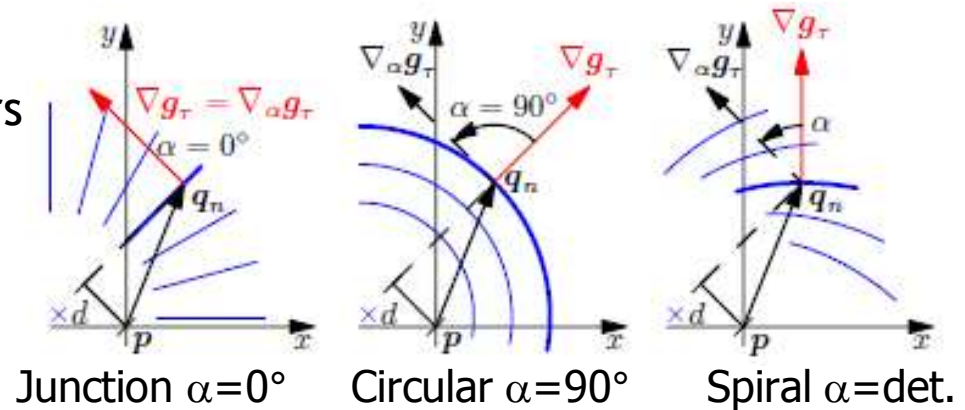
# New Kids On the Block

## an ad hoc selection

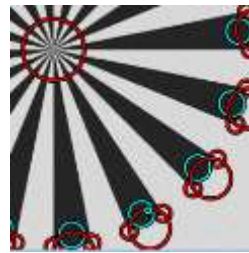


# SFOP Interpretable and Accurate Keypoints

- The image model:
  - *generalization* of Harris-like detectors based on the second moment matrix
  - minimizes the position covariance of  $p$  given the model of a point
  - model selected automatically or fixed to  $\alpha=0^\circ$  or  $90^\circ$



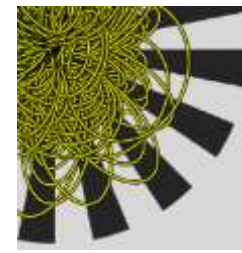
- Properties:
  - Similarity covariant, interpretable keypoints
  - Computationally, not much more complex than Harris detector per single scale level
  - Competitive performance (repeatability, feature number) to both blob and corner detectors



SFOP: junctions (red), circular features (cyan)



EBR



IBR



MSER



Harris affine



Hessian affine



DoG (Lowe)

- CPU and GPU implementation available

citations: 18 (2012)

Wolfgang Förstner, Timo Dickscheid and Falko Schindler: Detecting Interpretable and Accurate Scale-Invariant Keypoints, International Conference on Computer Vision (ICCV'09), Kyoto, 2009

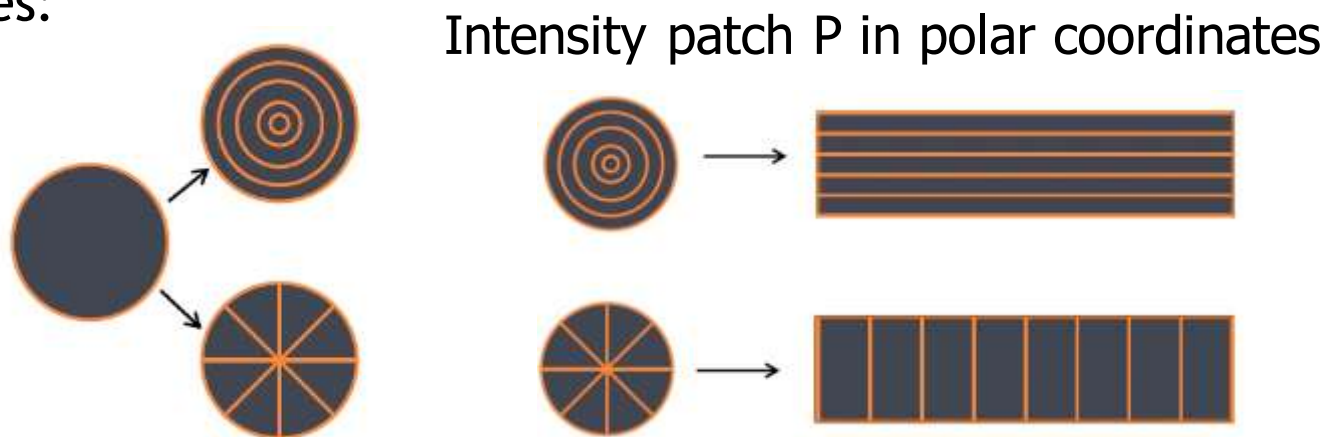
# Self-Similarity Feature Detector [Maver 10]



Jasna Maver: Self-Similarity and Points of Interest.

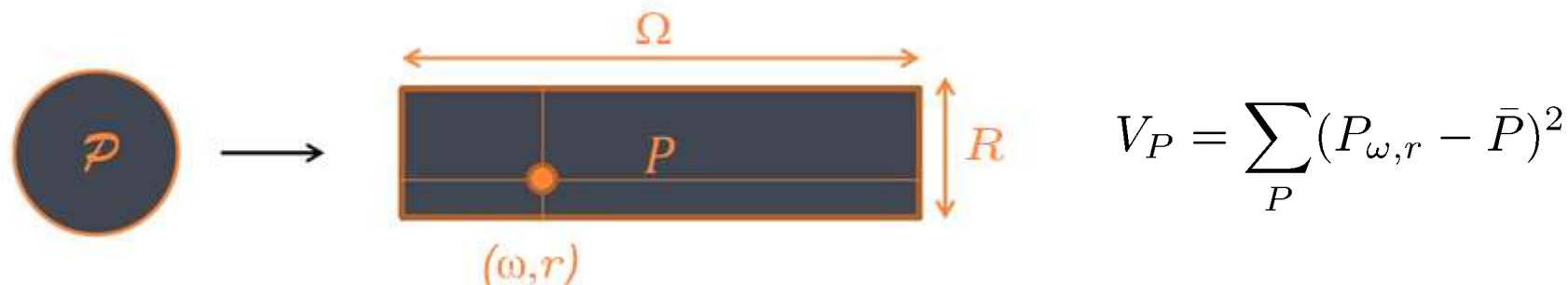
IEEE Trans. Pattern Anal. Mach. Intelligence 32(7): 1211-1226 (2010)

- The detector is based on “saliency” of a circular region  $P$
- $P$  is divided into annuli and circular sectors and sampled regularly in polar coordinates:



- Two measures, row (tangential) and column (radial) saliency, are computed as the variance in  $P$  and the row and column variances in annuli and circular sectors.
- *Generalization* of LoG (~ difference of intensities of two annuli) or Hessian saddle points (~ difference of intensities in four angular sectors)
- $S_t$  and  $S_r$  are evaluated in every pixel and multiple scales, features are detected by finding local maxima in saliency maps  $S_t$ ,  $S_r$  and  $S_r - S_t$

# Self-Similarity Feature Detector [Maver 10]



- Tangential saliency:

- normalized variance of sector intensity sums

$$S_t = \frac{1}{V_P R} \sum_{\omega=1}^{\Omega} (P_{\omega} - \bar{P}_{\omega})^2 \quad \text{where} \quad P_{\omega} = \sum_{r=1}^R P_{\omega, r}$$



- Radial saliency:

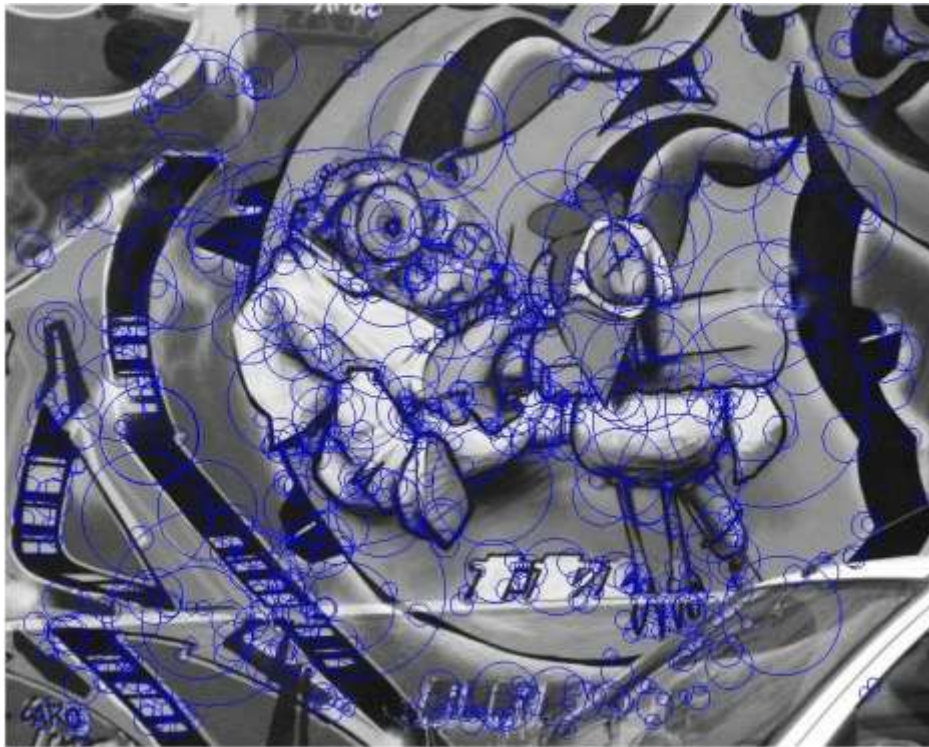
- normalized variance of annulus intensity sums

$$S_r = \frac{1}{V_P \Omega} \sum_{r=1}^R (P_r - \bar{P}_r)^2 \quad \text{where} \quad P_r = \sum_{\omega=1}^{\Omega} P_{\omega, r}$$





# Self-Similarity Feature Detector [Maver 10]



DoG features

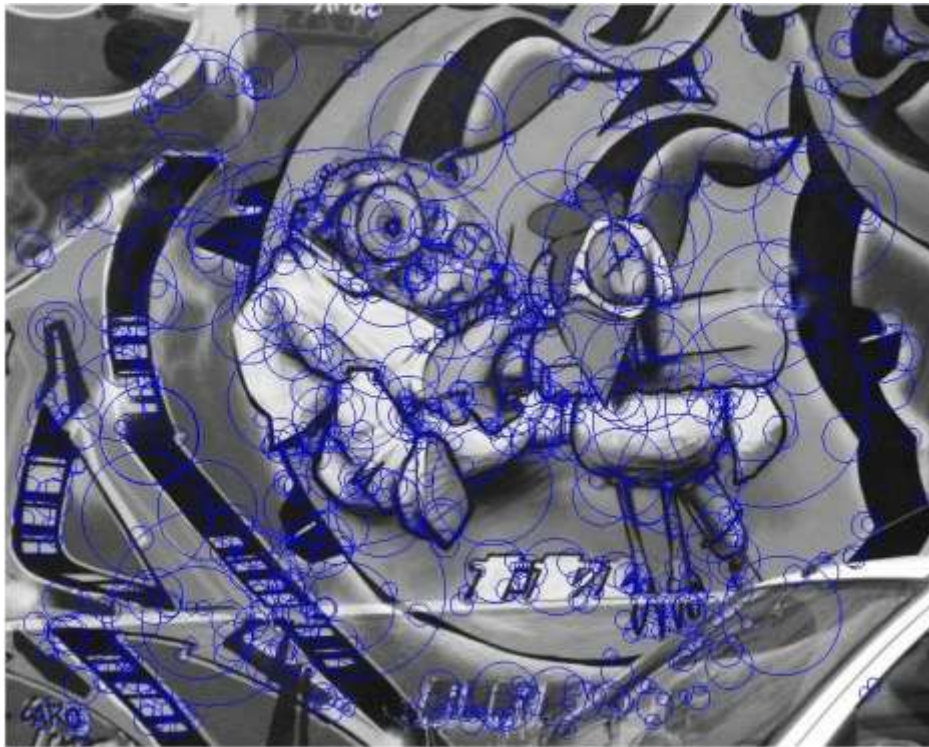


Radial features :

- (cyan) with overlap less than 80% with DoG features (cyan)
- (red), i.e. “DoG-like” R features



# Self-Similarity Feature Detector [Maver 10]



DoG features

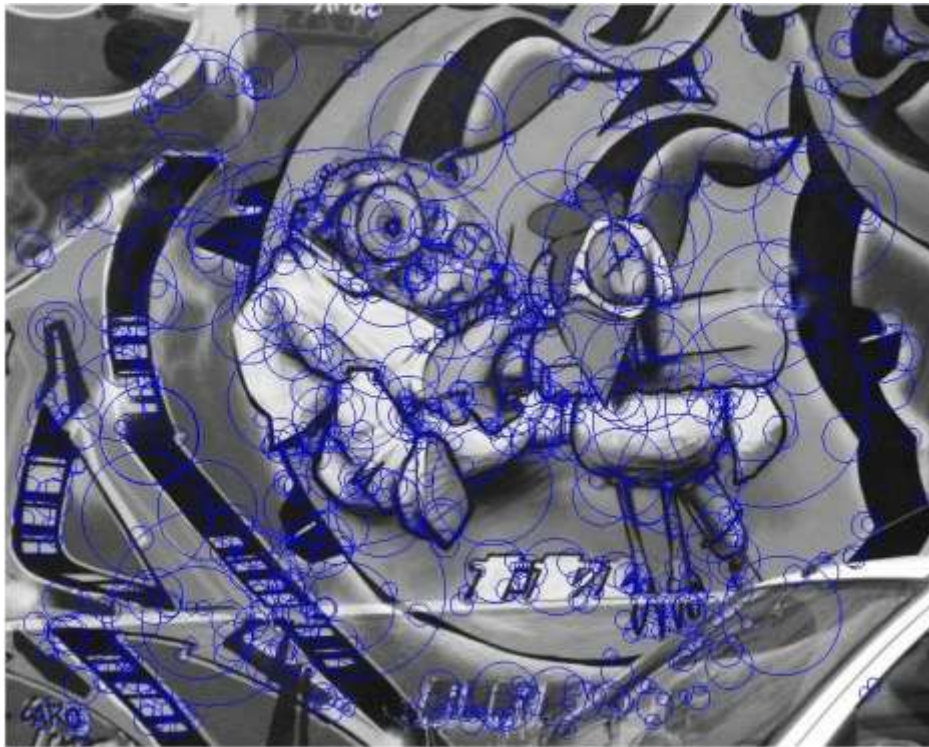


Tangential features :

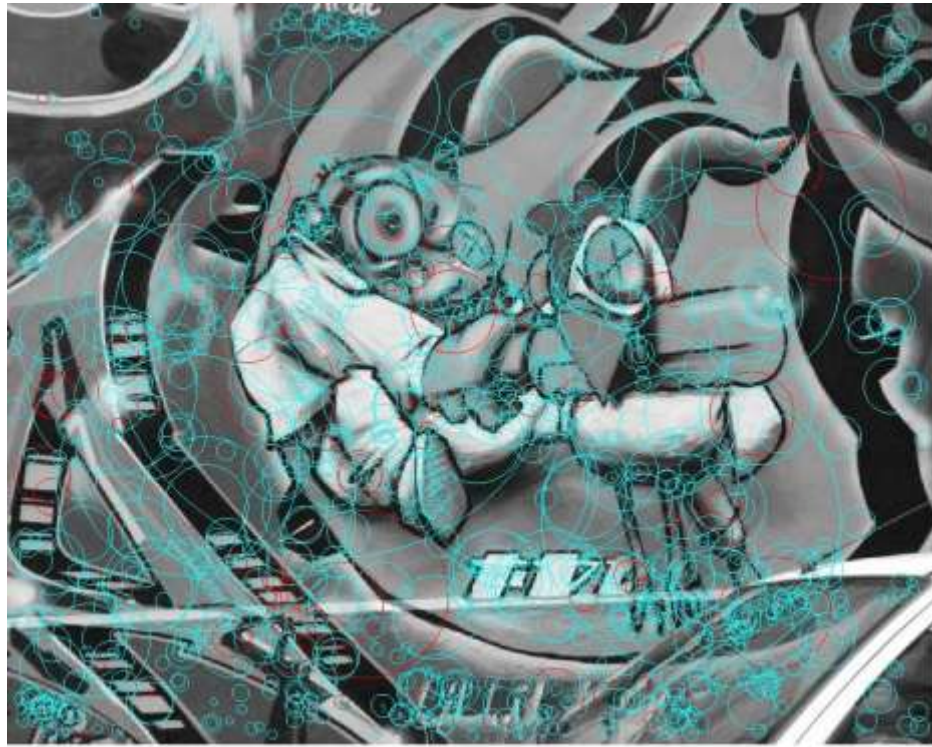
- (cyan) with overlap less than 80% with DoG features (cyan)
- (red), i.e. “DoG-like” T features



# Self-Similarity Feature Detector [Maver 10]



DoG features



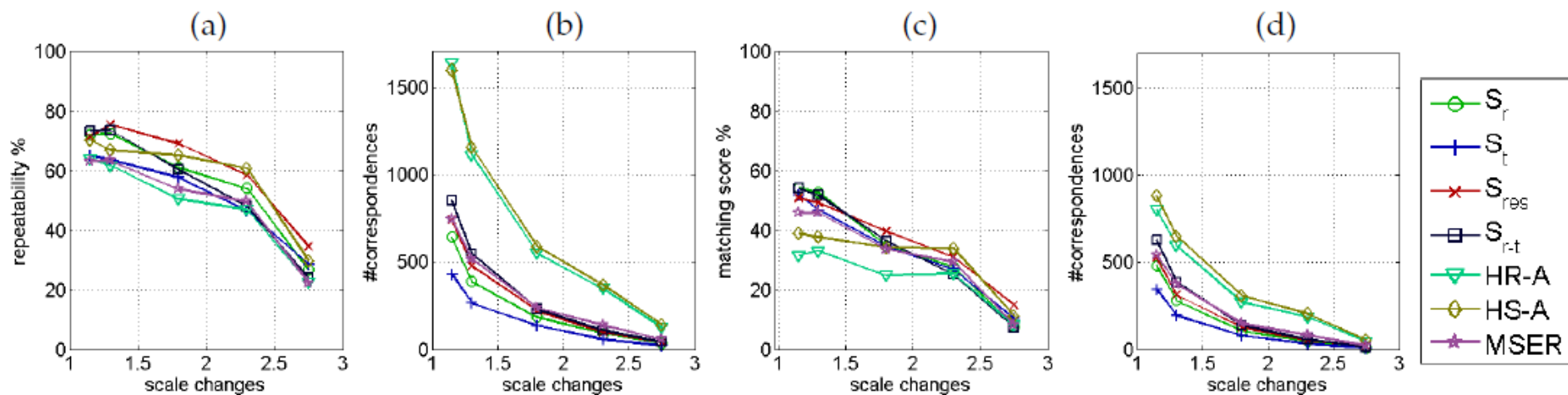
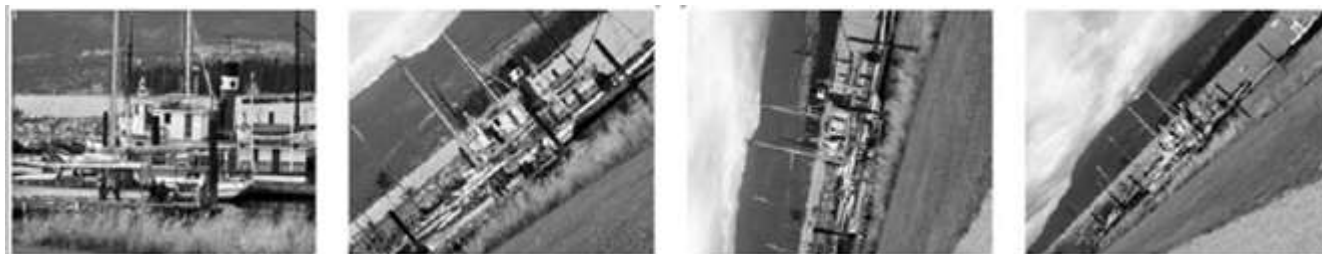
Radial-Tangential features :

- (cyan) with overlap less than 80% with DoG features (cyan)
- (red), i.e. "DoG-like" R-T features

# Self-Similarity Feature Detector [Maver 10]



- Scale change sequence - Performance



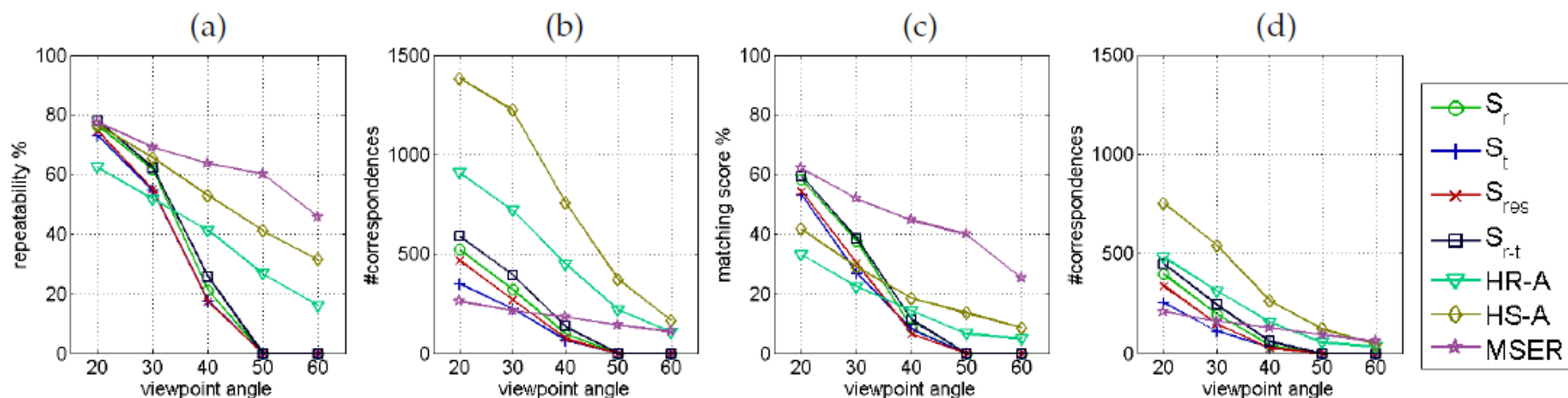
(a) Repeatability score. (b) Number of corresponding regions.  
(c) Matching score. (d) Number of correct nearest matches.



# Self-Similarity Feature Detector [Maver 10]



- Viewpoint change sequence - Performance



(a) Repeatability score. (b) Number of corresponding regions.  
(c) Matching score. (d) Number of correct nearest matches.

# Self-Similarity Feature Detector [Maver 10]



## Properties:

- features rarely overlap with *any* of the blob features (DoG, LoG, Hessian), *the detector is complementary*

	Harris like [%]	Hessian like [%]	DoG like [%]	LoG like [%]	Unique [%]
Rad	0.02	6.31	8.34	7.18	78.14
Tan	0.63	0	0	0	99.38
R-T	0.02	5.59	5.98	5.98	82.43

- invariant to rotation, linear transformation of intensity
- covariant with rotation and scaling
- runs ~5s on 1024x768 image  
(recent CMP implementation, before orders of magnitude longer)

## Notes:

- no evaluated (proposed) a version of the Maver detector with the Baumberg-Lindeberg-Garding iteration for affine deformation modelling
- 10 cites in Google Scholar only (because it was so sloooow?)

# Local Symmetry Features



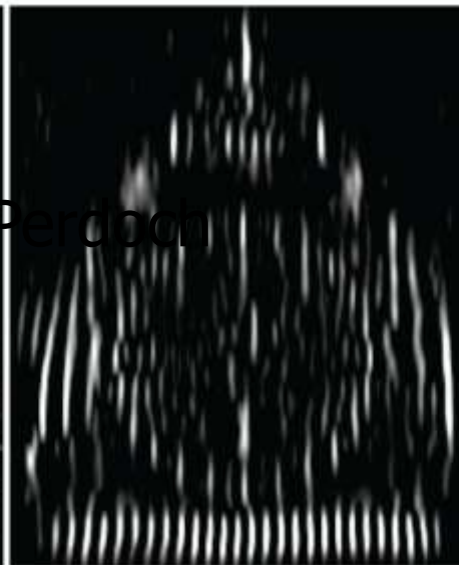
Hypothesis: symmetries survive a dramatic change in illumination, ...  
Symmetries might be useful in long-temporal baseline matching



slide credit:  
Noah Snavely



Horizontal  
symmetry  
score



Vertical  
symmetry  
score



Points with  
multiple  
symmetries

Haugge, Snavely:  
Image matching using local symmetry features. CVPR 2012: 206-213

# Local Symmetry Features



video credit:  
Noah Snavely

## Local symmetries detected at a larger scale



Horizontal  
symmetry  
score



Vertical  
symmetry  
score

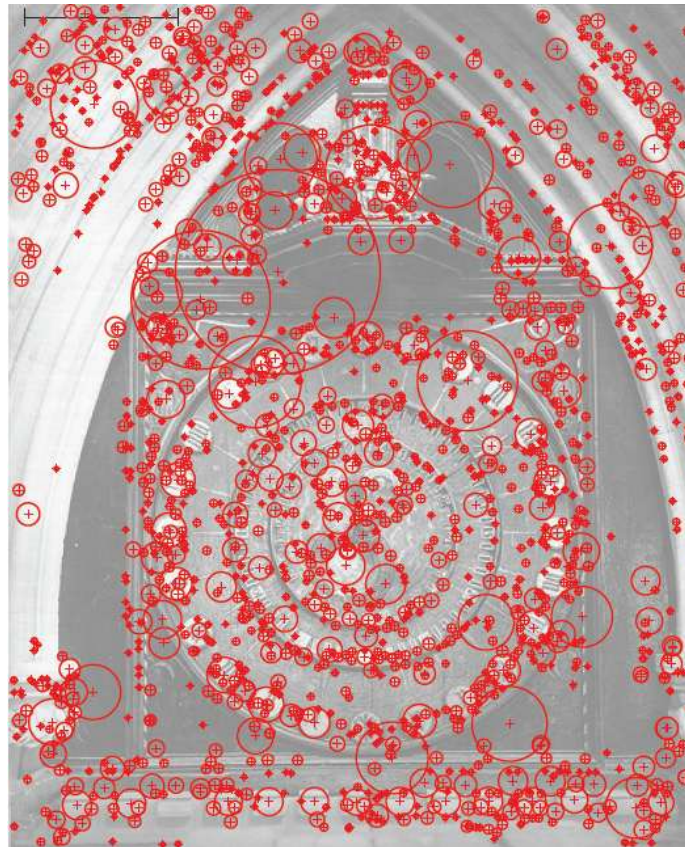


Points with  
multiple  
symmetries

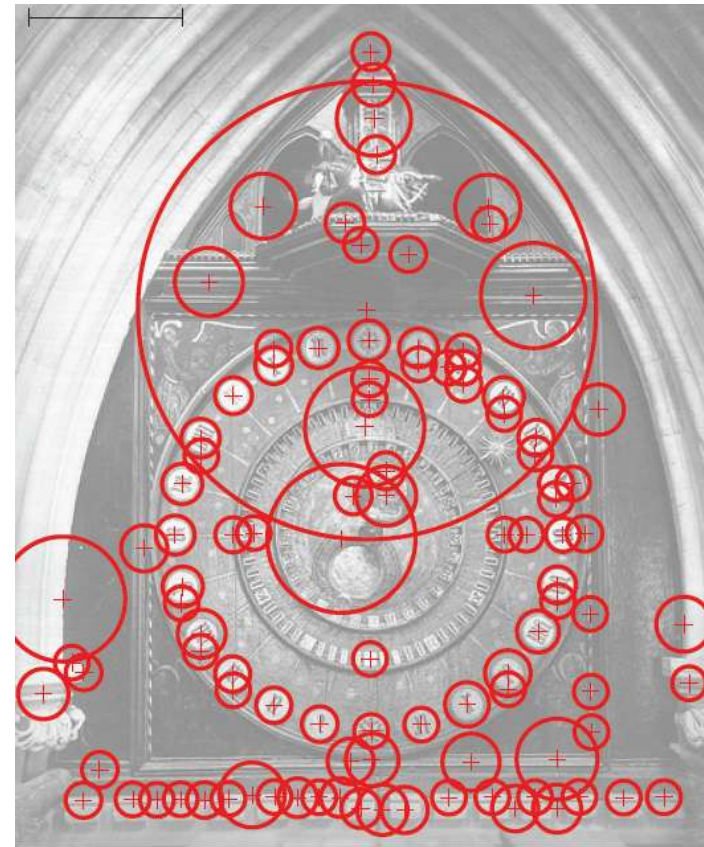


# Local Symmetry Features

video credit:  
Noah Snavely

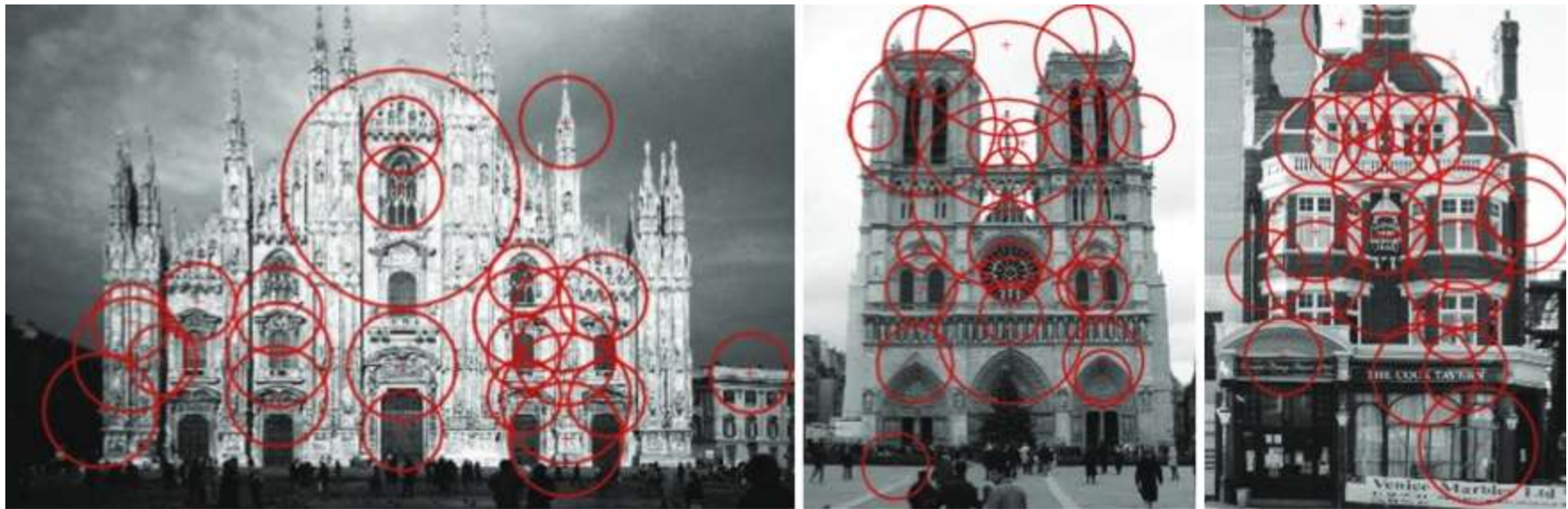


DoG



Local symmetry features

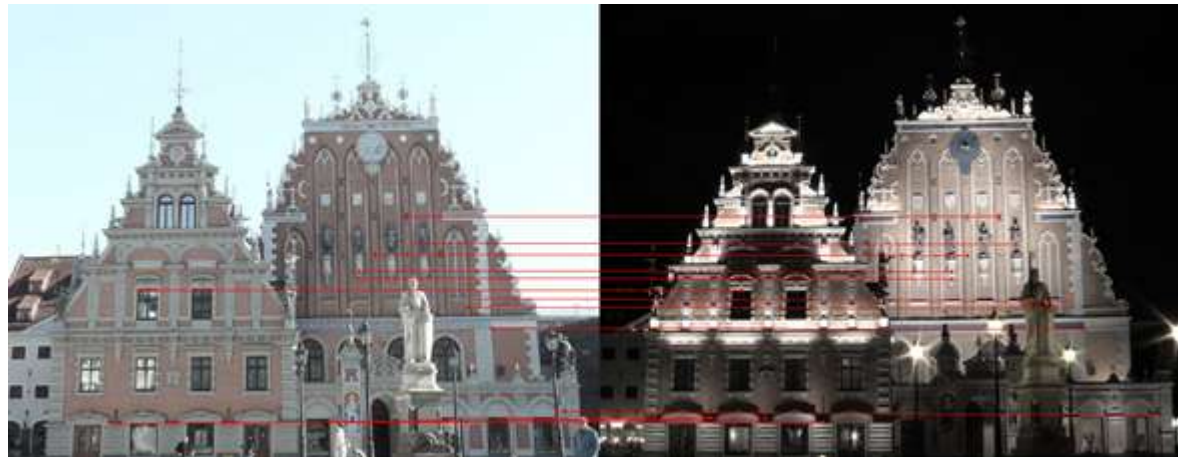
# Local Symmetry Features” More examples



Local symmetry scores used for detection *and* description



# Local Symmetry Features” More examples

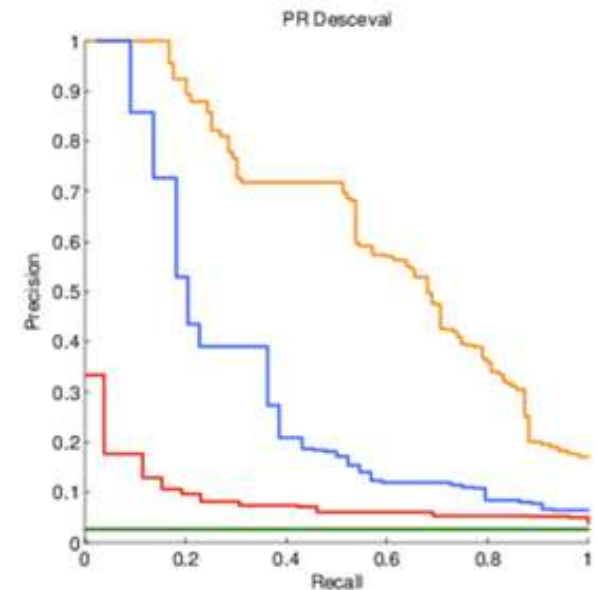


SIFT feature matches  
(after fitting homography)



Symmetry feature matches

— GRID + SIFT (0.02)  
— GRID + SYM (0.62)  
— GRID + Self-Similarity (0.29)  
— GRID + SIFT-SYM (0.07)

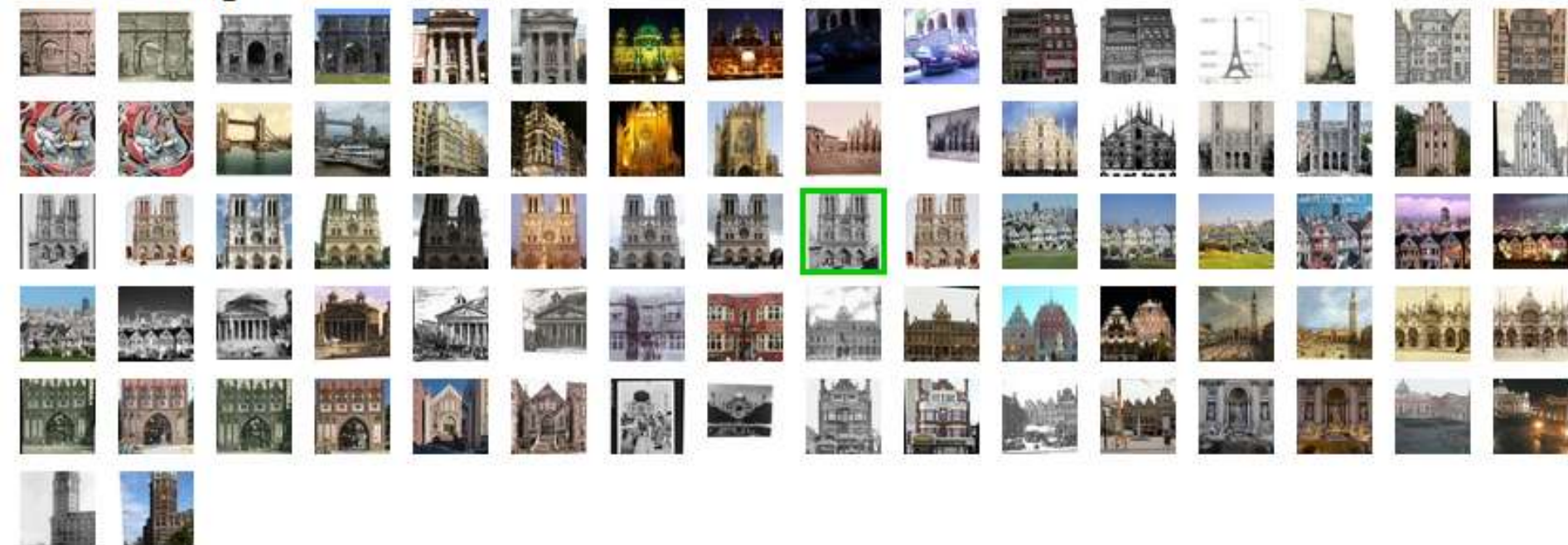




# Benchmark for difficult image matching problems



## Dataset Images



<http://www.cs.cornell.edu/projects/symfeat/>

# ASIFT - a new affine invariant detector?



Idea:

- (due to Lepetit et al. and others) Synthesize warped views of both images in two view matching
- Match all pairs of synthesized images.
- Impose a geometric constraint to prune the tentative correspondences
- Positives:
  - yes, more correct correspondences are found =>
  - some (very) difficult matching problems solvable
- Negatives:
  - detection time goes up significantly
  - matching time goes up even more significantly (quadratically)
  - problematic use in e.g. retrieval
  - issue with evaluation (not all reported matches are inliers)

Guoshen Yu, Jean-Michel Morel: A fully affine invariant image comparison method. ICASSP 2009: 1597-1600

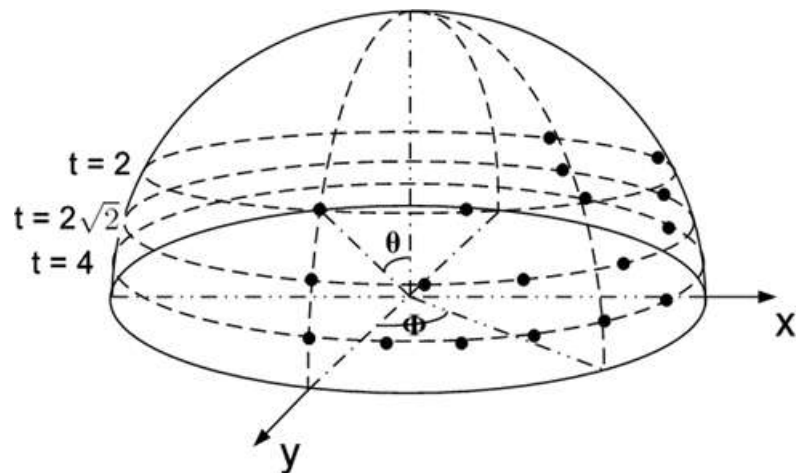
G.Yu and J.M. Morel, ASIFT: An Algorithm for Fully Affine Invariant Comparison, Image Processing On Line, 2011.

ECCV 2012 Modern features: ... Detectors.

# ASIFT - Algorithm

“detection” phase (in both images)

1. Sample tilt  $t$  and longitude  $\phi$
2. For each simulated viewpoint detect DoG features and describe them using SIFT descriptor
3. Back project features back to frontal image



“matching” phase

1. Find matches between all simulated viewpoints from both images
2. Remove duplicates and inconsistent matches

- ASIFT is NOT a detector - rather a “matching” scheme
  - generates “redundant” representation, that slows down the matching significantly
  - any detector may benefit from this “matching” scheme

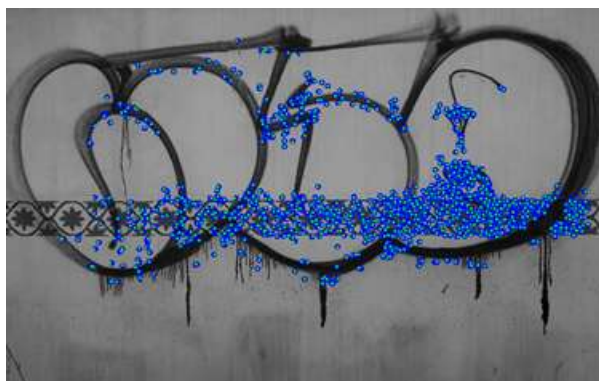
# Affine Invariance by Sampling Viewsphere



- Why use DoG (“SIFT”)?  
Replacing DoG by HessianAffine or MSER is beneficial and more efficient!
- HessianAffine matches from direct matching:



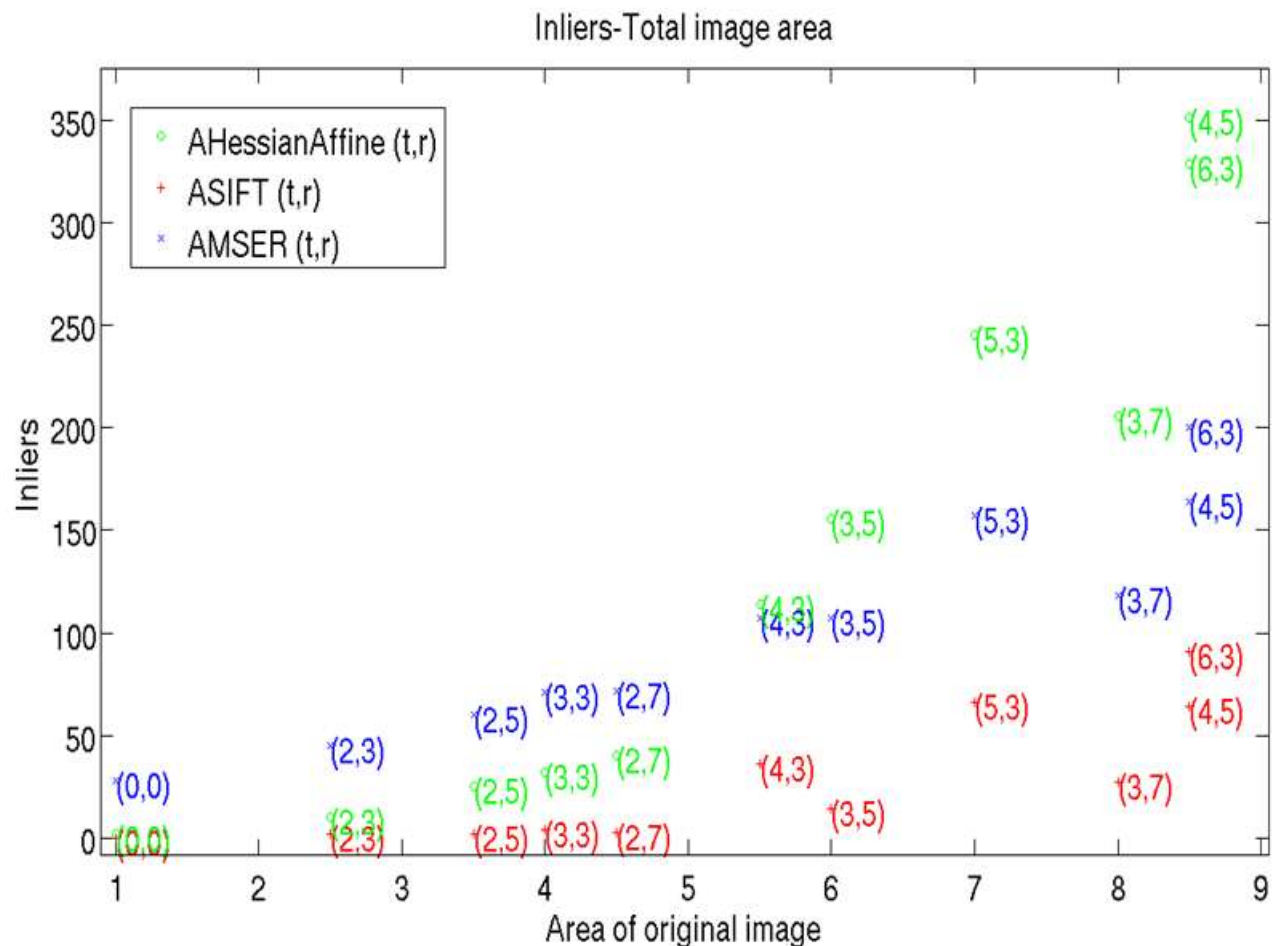
- HessianAffine matches with synthesized images (viewsphere sampling)



- Conclusions: 1. generating synthesized view work 2. no reason to use DoG

# Affine Invariance by Sampling Viewsphere

## ASIFT, AMSER and AHessianAffine detectors comparison



Comparison on Graffiti and other Mikolajczyk et al. IJCV 05 images (t,r) - the number of synthesized tilted and rotated images, Moreover AMSER and AHessianAffine have higher repeatability.

# Questions, please?