Features Matching

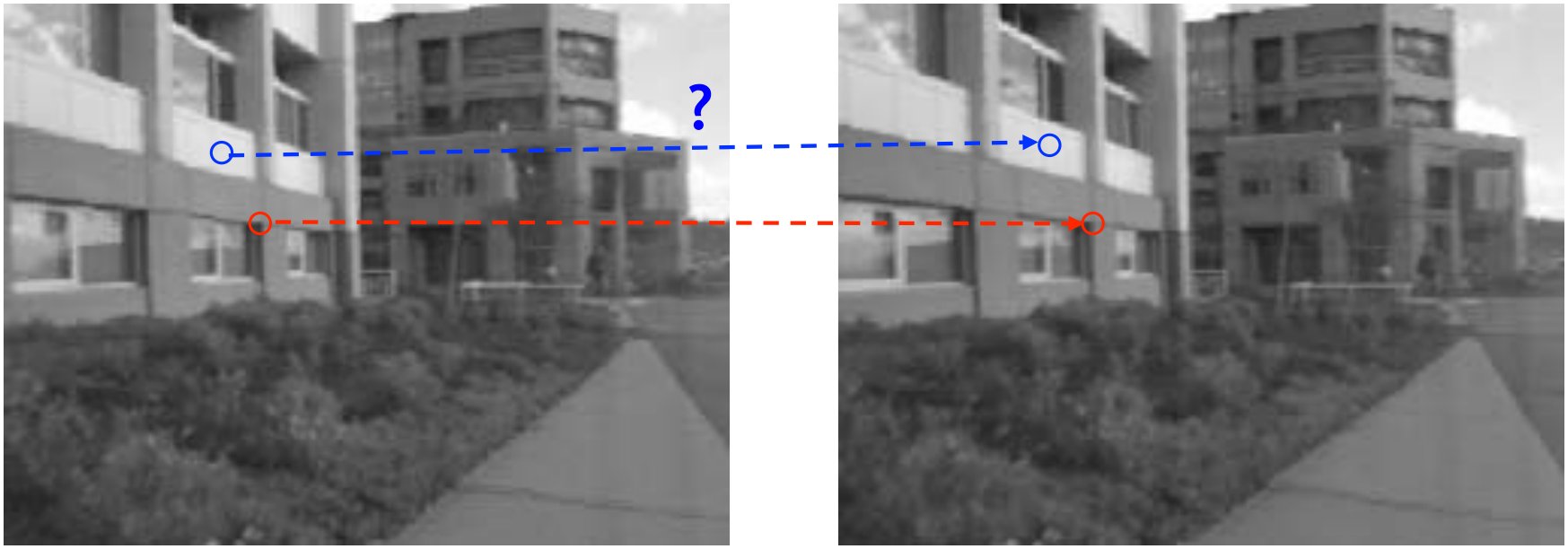
Kuan-Wen Chen
2022/5/26

Features

with slides by Yung-Yu Chuang, Trevor Darrell Cordelia Schmid, David Lowe, Darya Frolova, Denis Simakov, Robert Collins and Jiwon Kim

Features

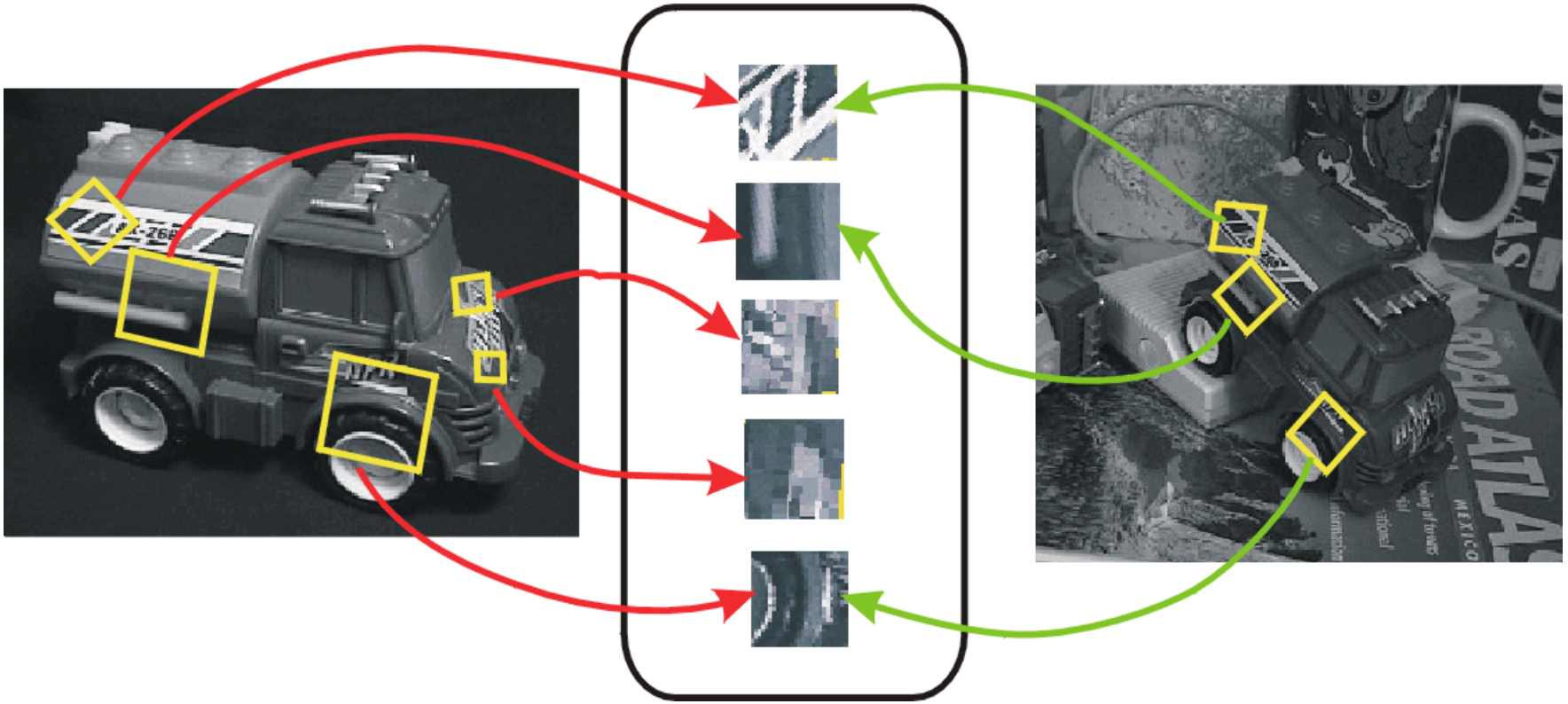
- Also known as interesting points, salient points or keypoints. Points that you can easily point out their correspondences in multiple images using only local information.



Applications

- Object or scene recognition
- Structure from motion
- Stereo
- Motion tracking
- …

Object Recognition



SIFT Features

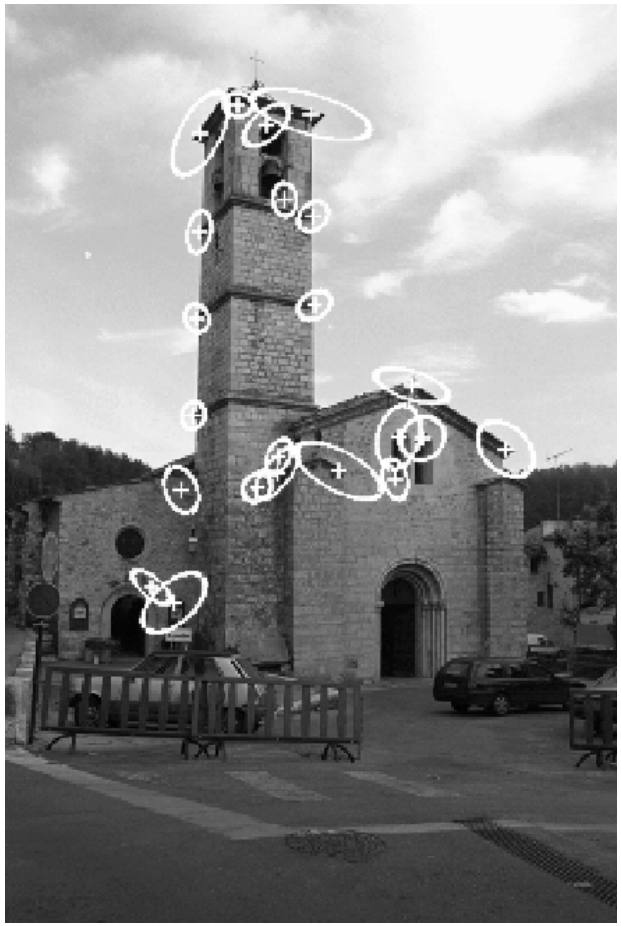
Image retrieval

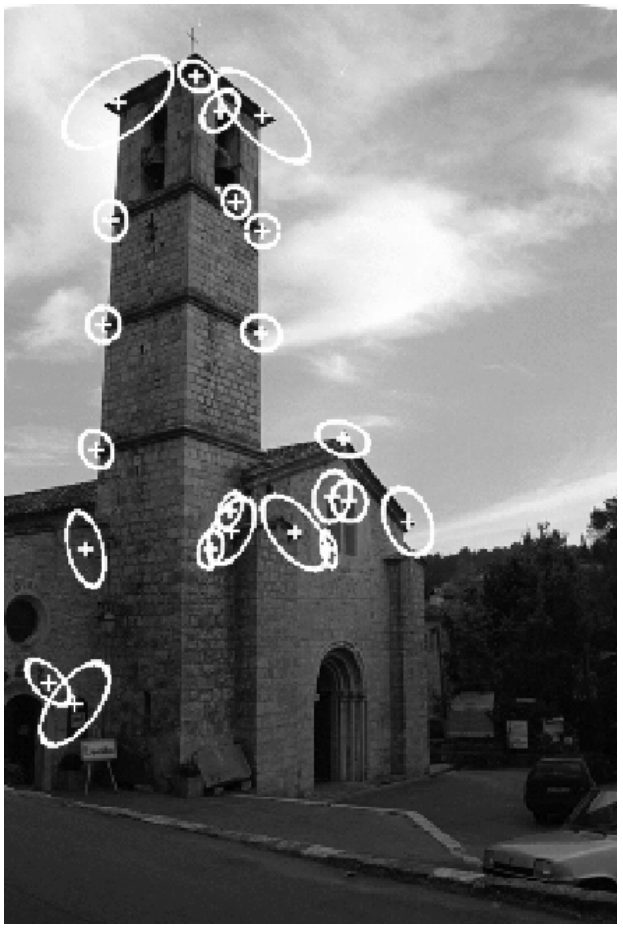
change in viewing angle

• • •

> 5000 images

Image retrieval





22 correct matches

Image retrieval

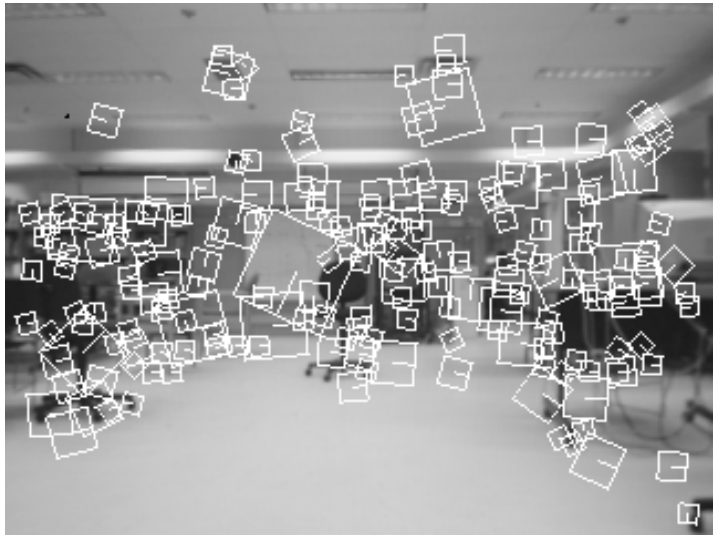
change in viewing angle

+ scale change

• • •

> 5000 images

Robot location



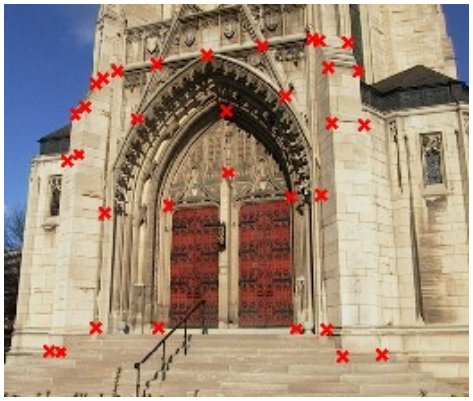
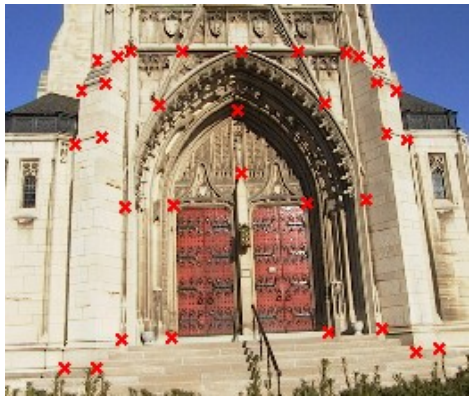
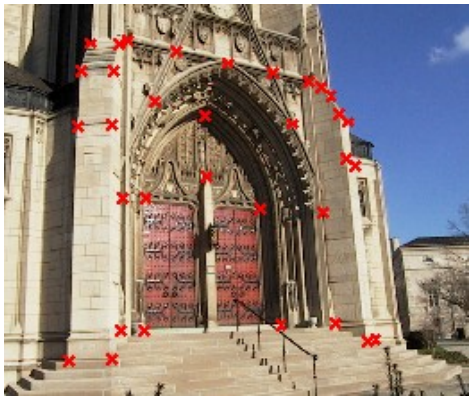
Robotics: Sony Aibo

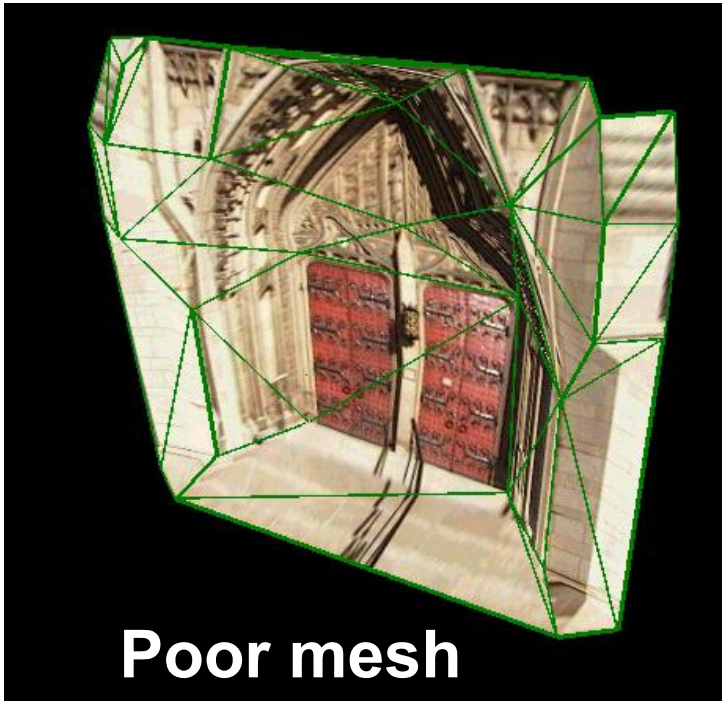
SIFT is used for

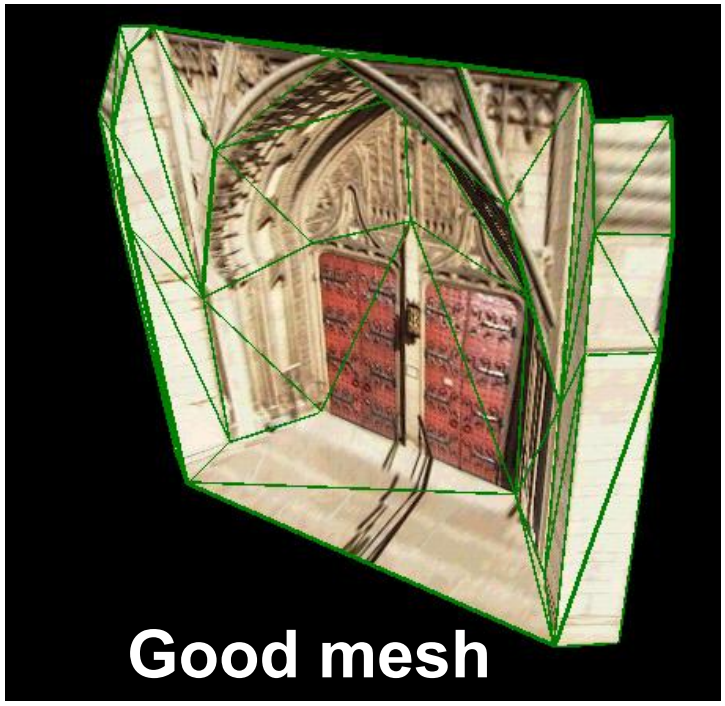
- Recognizing charging station
- Communicating with visual cards
- Teaching object recognition

- soccer

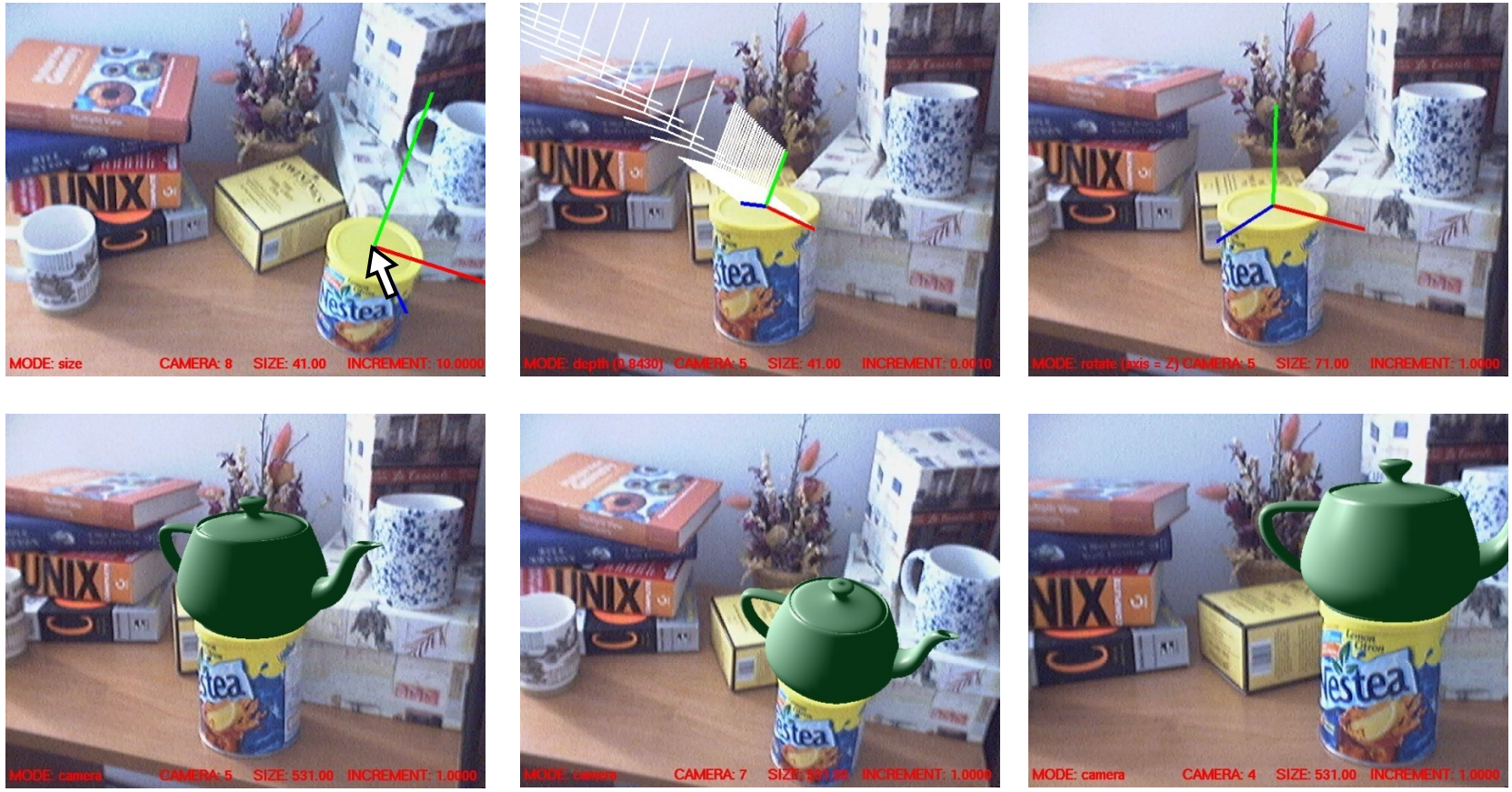
Structure from Motion

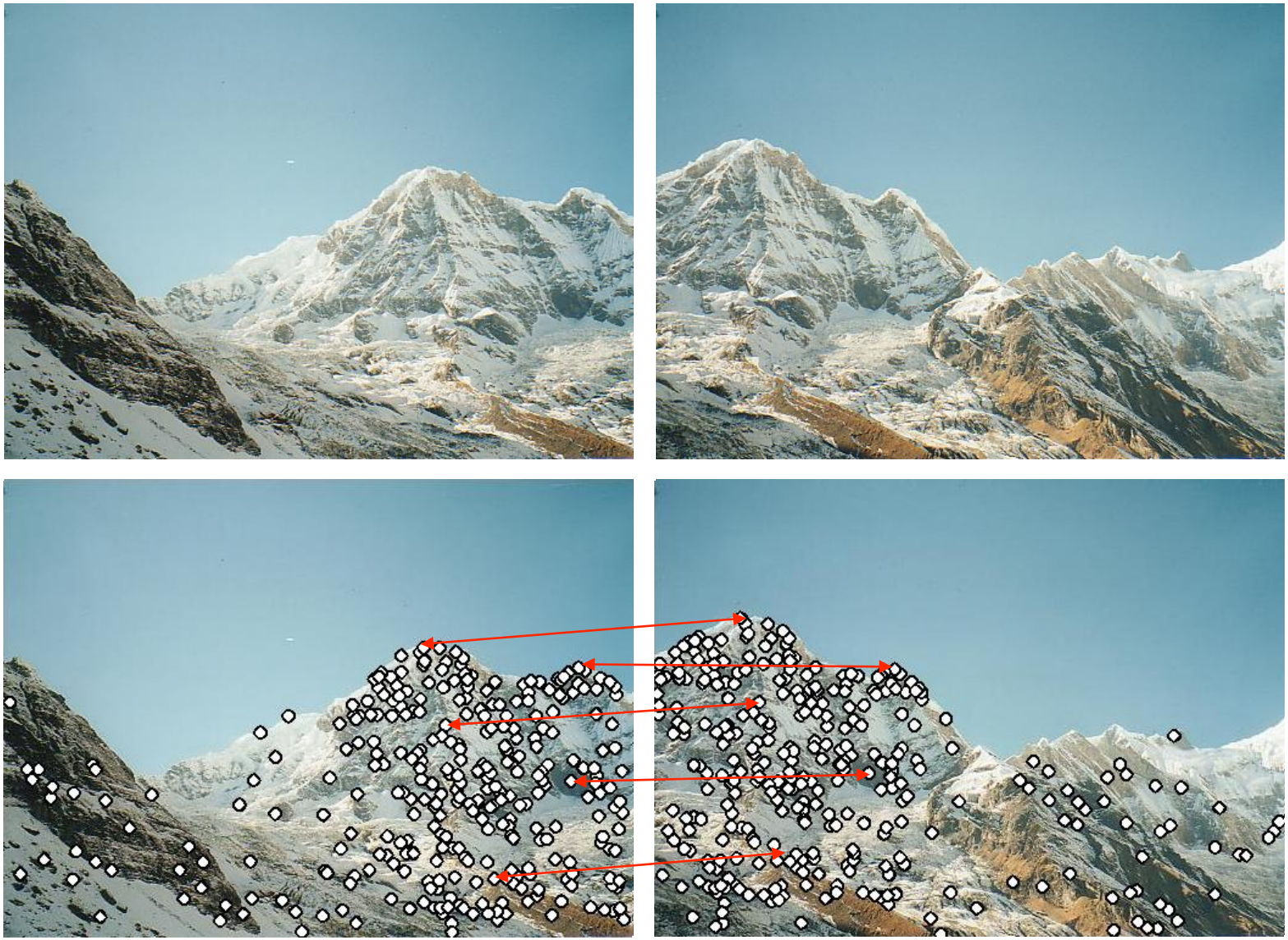




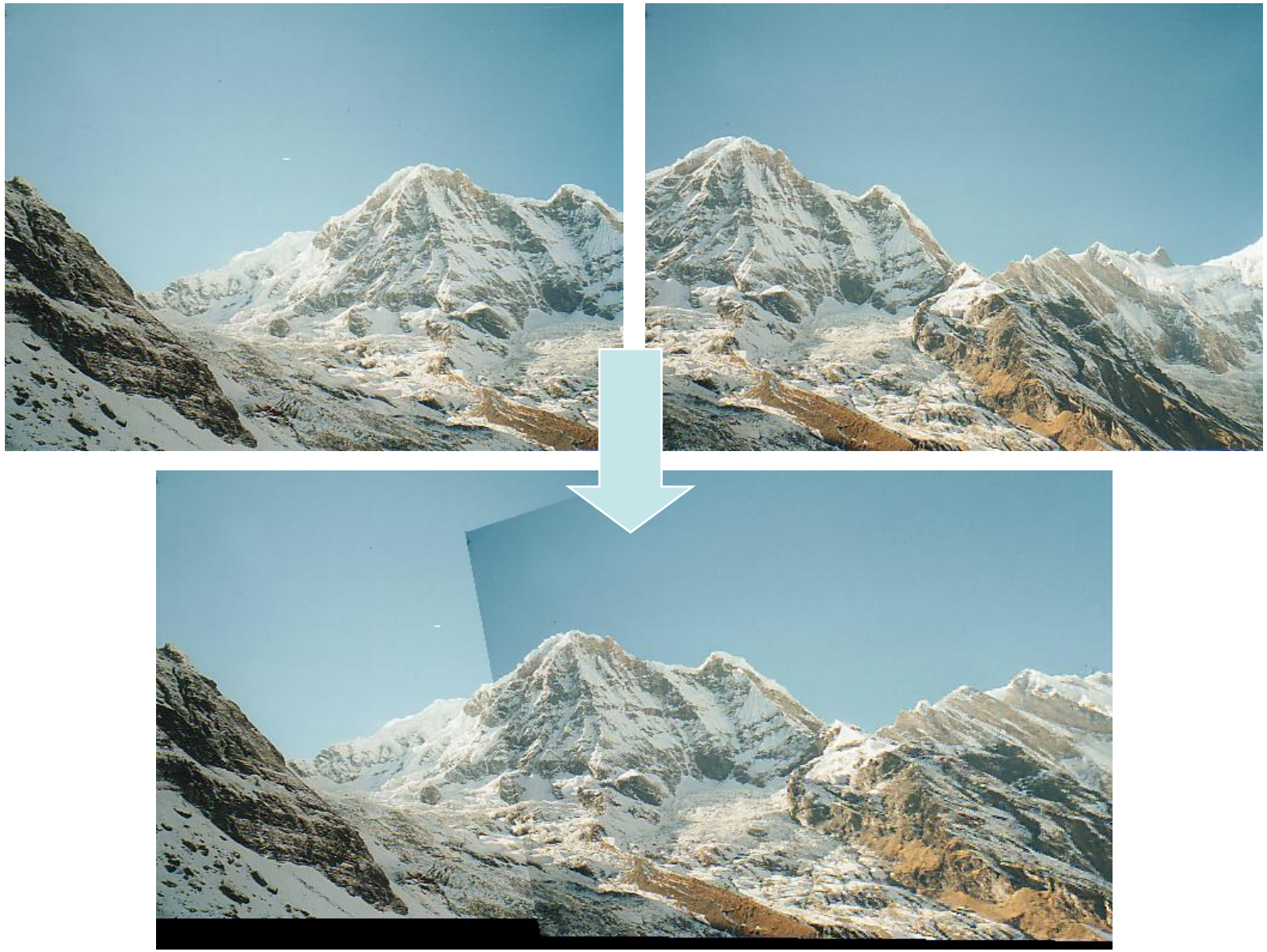
Augmented reality



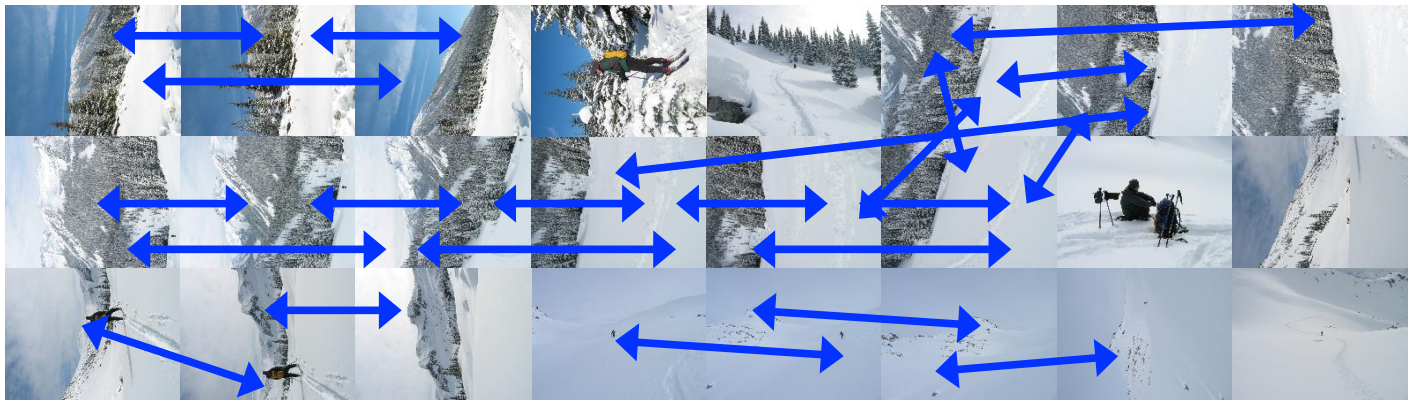
Automatic image stitching



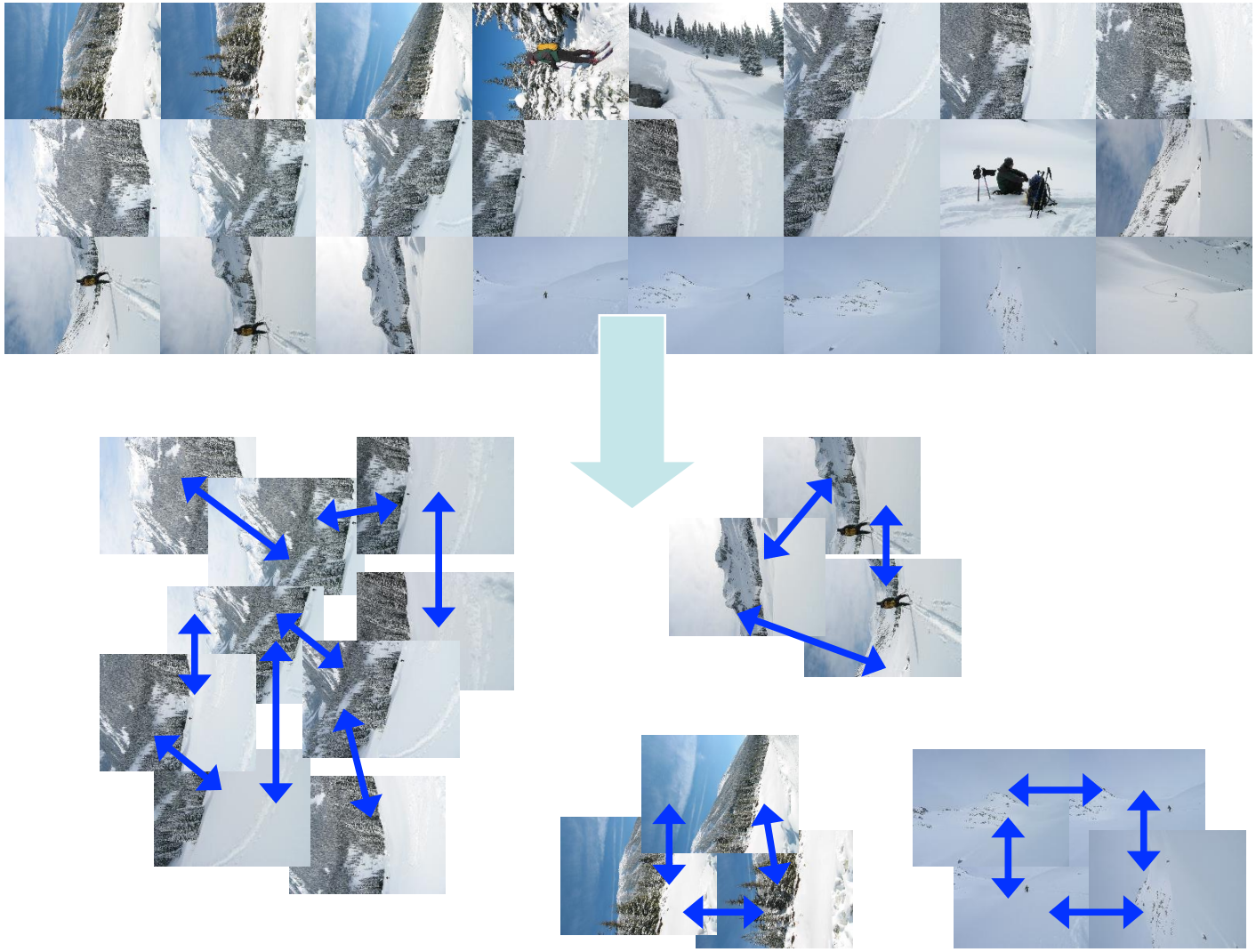
Automatic image stitching



Automatic image stitching



Automatic image stitching



Automatic image stitching

Desired properties for features

- **Distinctive**: a single feature can be correctly matched with high probability.

- **Invariant**: invariant to scale, rotation, affine, illumination and noise for robust matching across a substantial range of affine distortion, viewpoint change and so on. That is, it is repeatable.

Components

- **Feature detection** locates where they are
- **Feature description** describes what they are
- **Feature matching** decides whether two are the same one

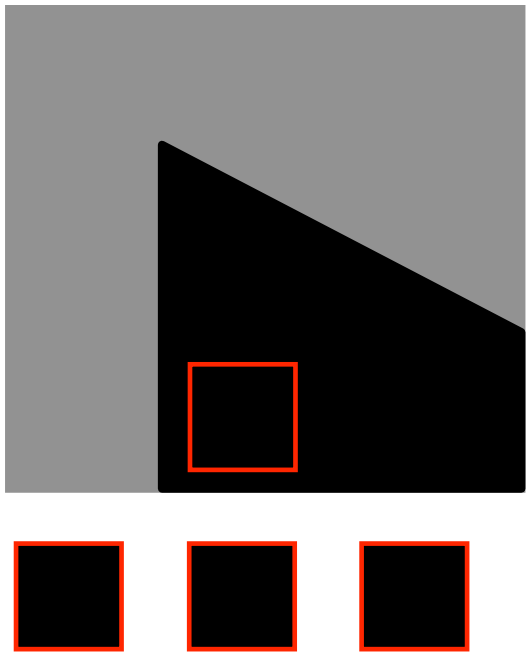
Harris corner detector

with slides by Yung-Yu Chuang, Trevor Darrell Cordelia Schmid, David Lowe, Darya Frolova, Denis Simakov, Robert Collins and Jiwon Kim

Moravec corner detector (1980)

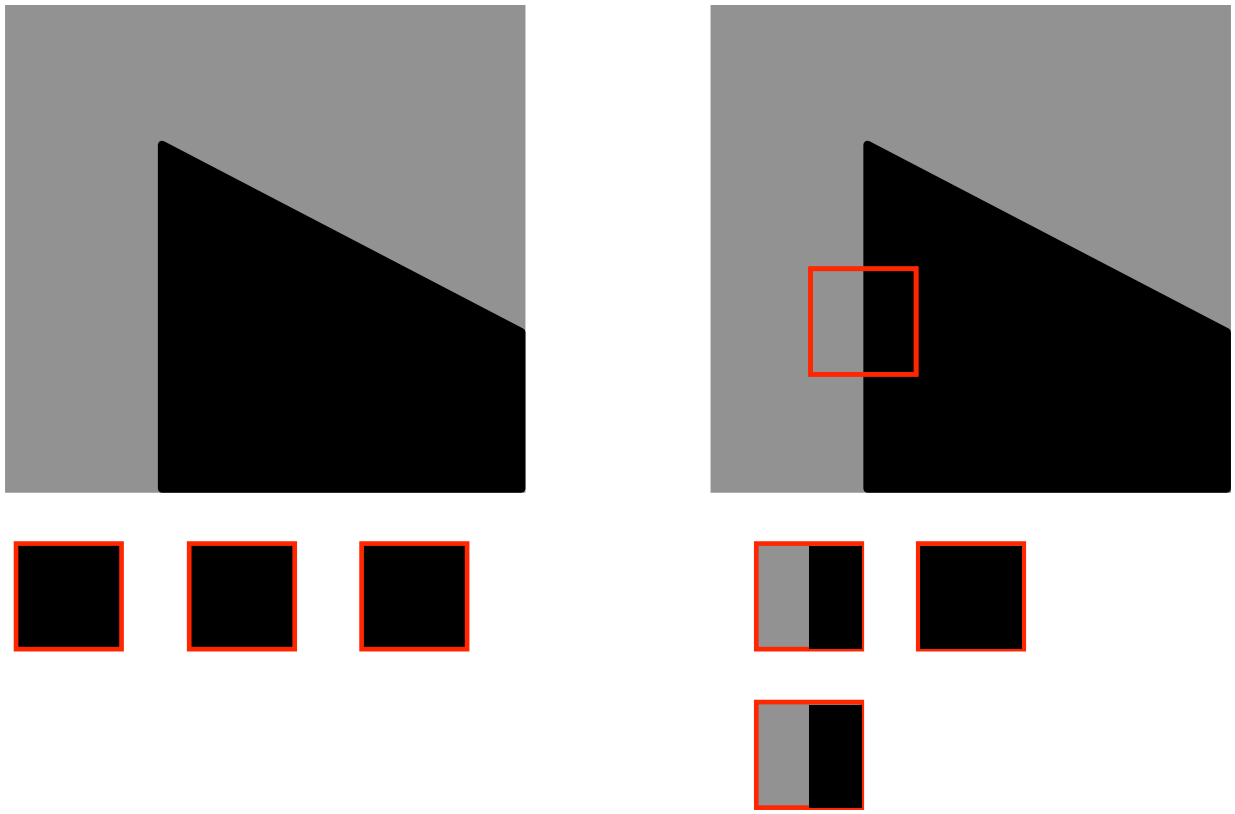
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity

Moravec corner detector



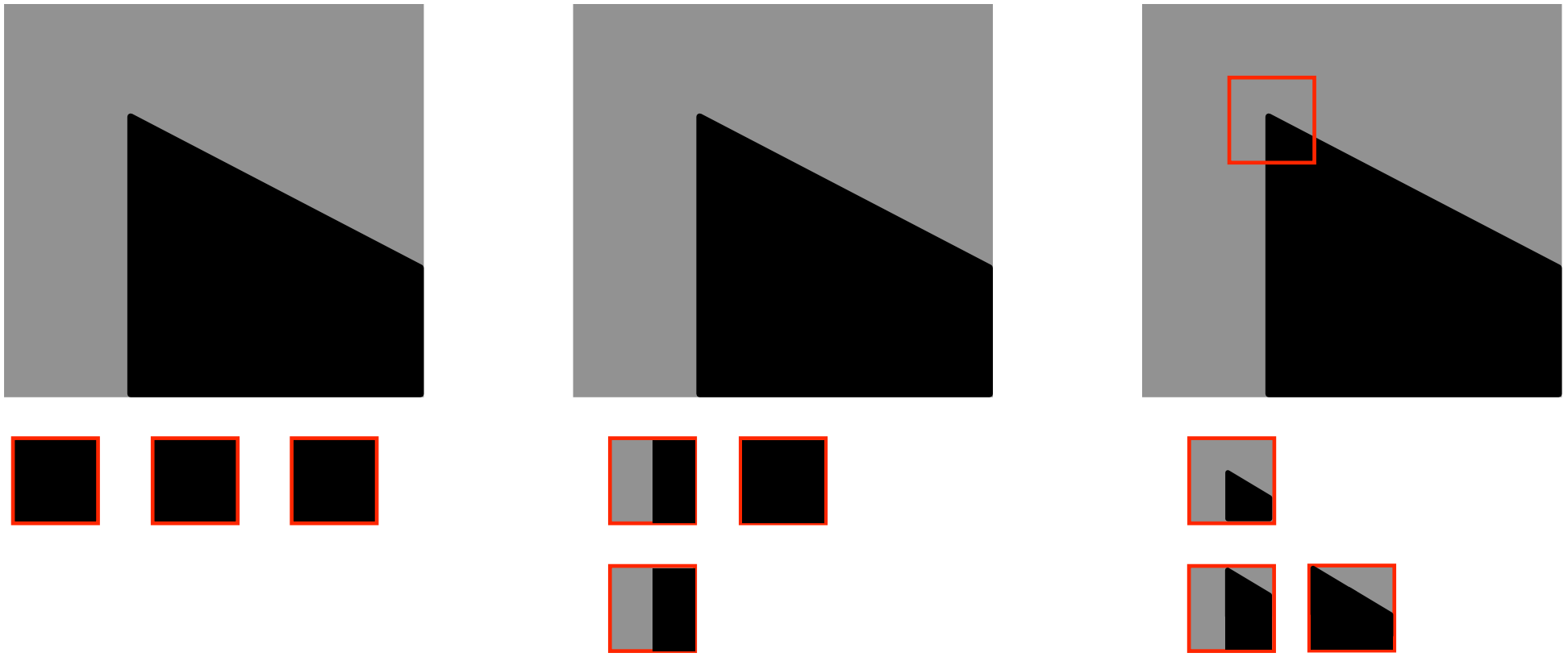
flat

Moravec corner detector



flat edge

Moravec corner detector



flat

edge

corner
isolated point

Moravec corner detector

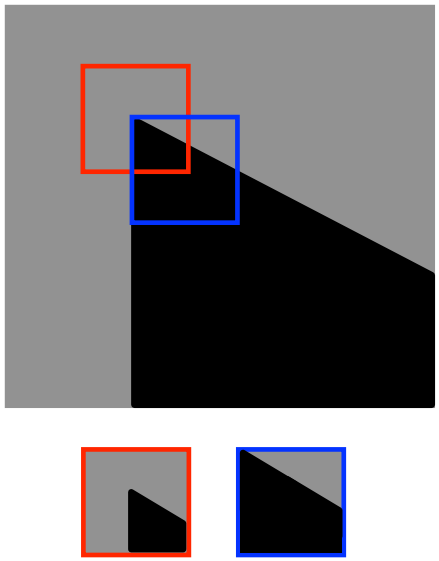
Change of intensity for the shift [*u*,*v*]:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

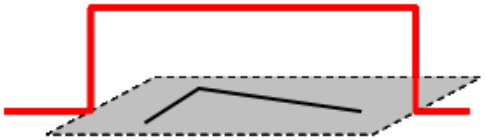
window function

shifted intensity

intensity



Window function $w(x,y)$ =



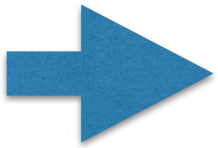
1 in window, 0 outside

Four shifts: (u,v) = (1,0), (1,1), (0,1), (-1, 1)

Look for local maxima in $min\{E\}$

Problems of Moravec detector

- Noisy response due to a binary window function
- Only a set of shifts at every 45 degree is considered
- Only minimum of E is taken into account



Harris corner detector (1988) solves these problems.

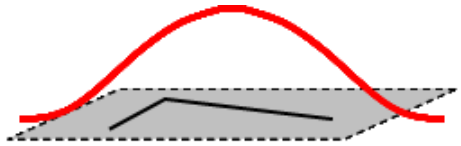
Harris corner detector

- Problem: noisy response due to a binary window function

- Improvement: use a Gaussian function

$$w(x,y) = \exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

Window function $w(x,y)$ =



Gaussian

Harris corner detector

- Problem: only a set of shifts at every 45 degree is considered

- Improvement: consider all small shifts by Taylor's expansion

Small motion assumption

$$E(u,v) = \sum_{(x,y)\in W} [I(x+u, y+v) - I(x,y)]^2$$

Small motion assumption

Taylor Series expansion of I:

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

If the motion (u,v) is small, then first order approx is good

$$I(x+u, y+v) \approx I(x,y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

$$\approx I(x,y) + [I_x \; I_y] \begin{bmatrix} u \\ v \end{bmatrix}$$

$$\text{shorthand: } I_x = \frac{\partial I}{\partial x}$$

Plugging this into the formula on the previous slide…

Feature detection: the math

$$
\begin{aligned}
E(u,v) &= \sum_{(x,y)\in W} [I(x+u,y+v) - I(x,y)]^2 \\
&\approx \sum_{(x,y)\in W} [I(x,y) + [I_x \; I_y] \begin{bmatrix} u \\ v \end{bmatrix} - I(x,y)]^2 \\
&\approx \sum_{(x,y)\in W} \left[[I_x \; I_y] \begin{bmatrix} u \\ v \end{bmatrix} \right]^2
\end{aligned}
$$

This can be rewritten:

$$
E(u,v) = \sum_{(x,y)\in W} [u \; v] \underbrace{\begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} u \\ v \end{bmatrix}
$$

$$
E(u,v) \cong \begin{bmatrix} u & v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}
$$

Harris corner detector

- Problem: only minimum of E is taken into account

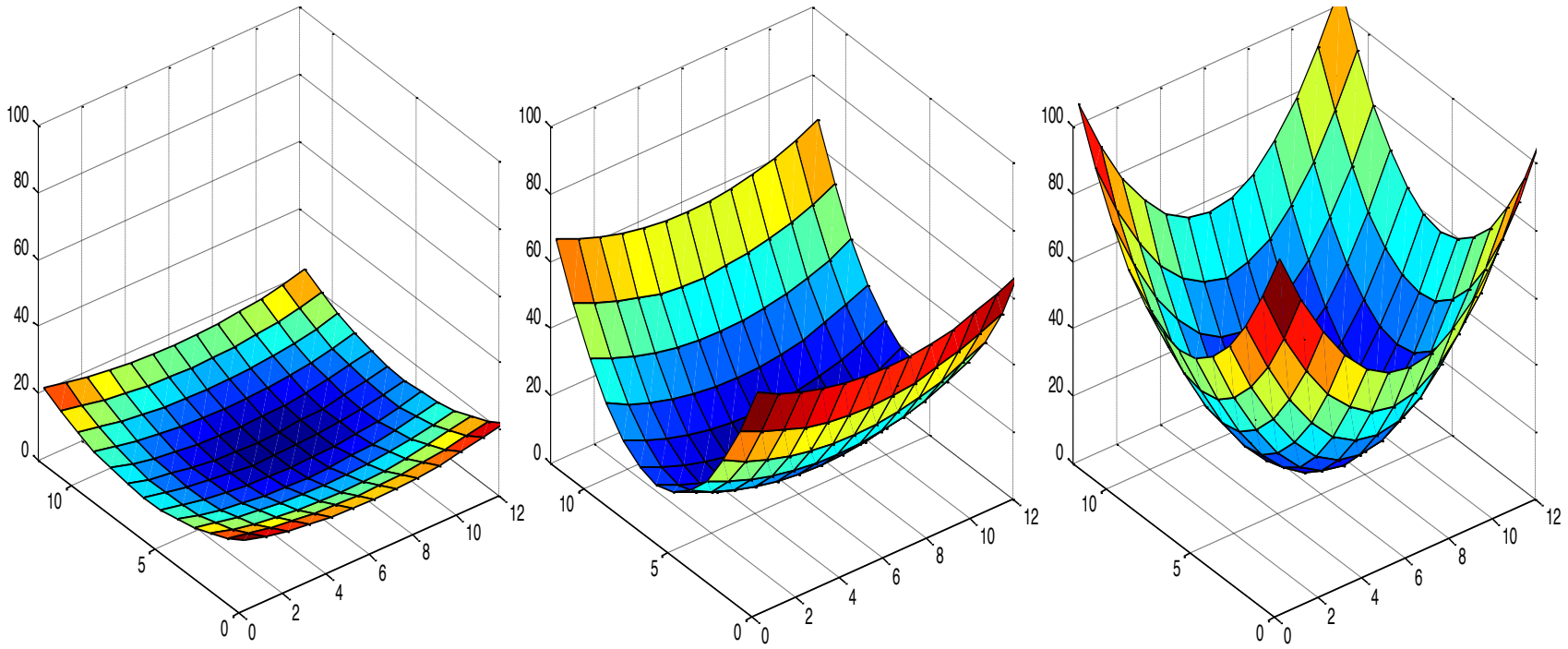
- Improvement: a new corner measurement by investigating the shape of the error function

$$E(u,v) \cong \begin{bmatrix} u & v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

- $\mathbf{u}^T\mathbf{M}\mathbf{u}$ represents a quadratic function; Thus, we can analyze E's shape by looking at the property of $\mathbf{M}$

Harris corner detector

- High-level idea: what shape of the error function will we prefer for features?



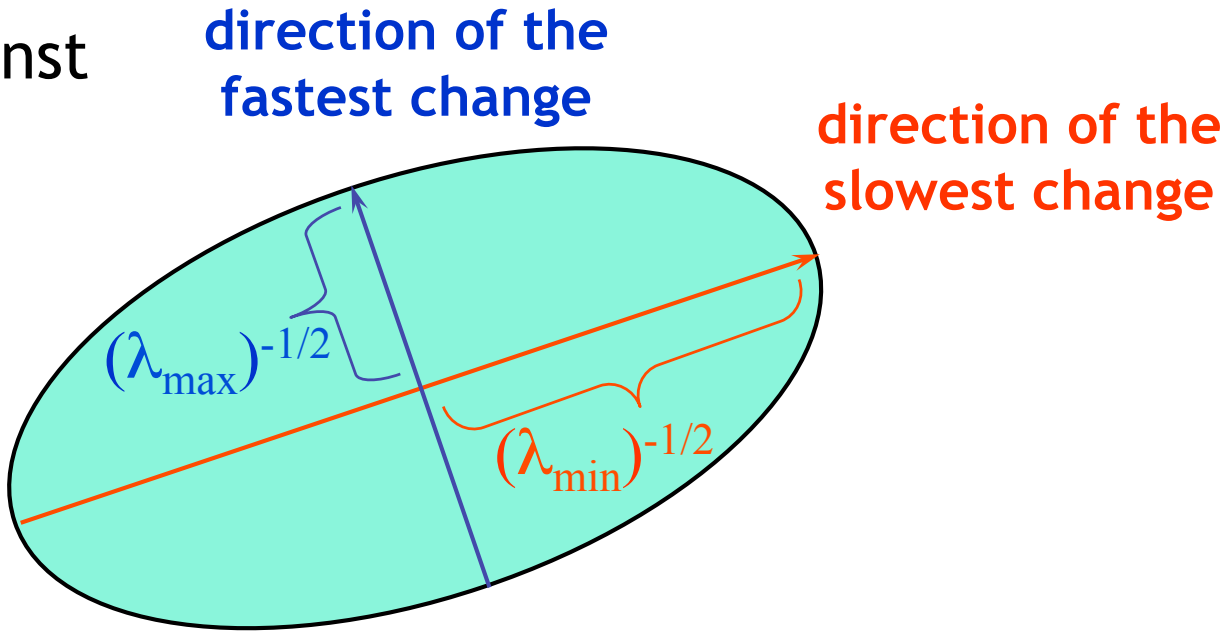
Harris corner detector

- Intensity change in shifting window: eigenvalue analysis

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

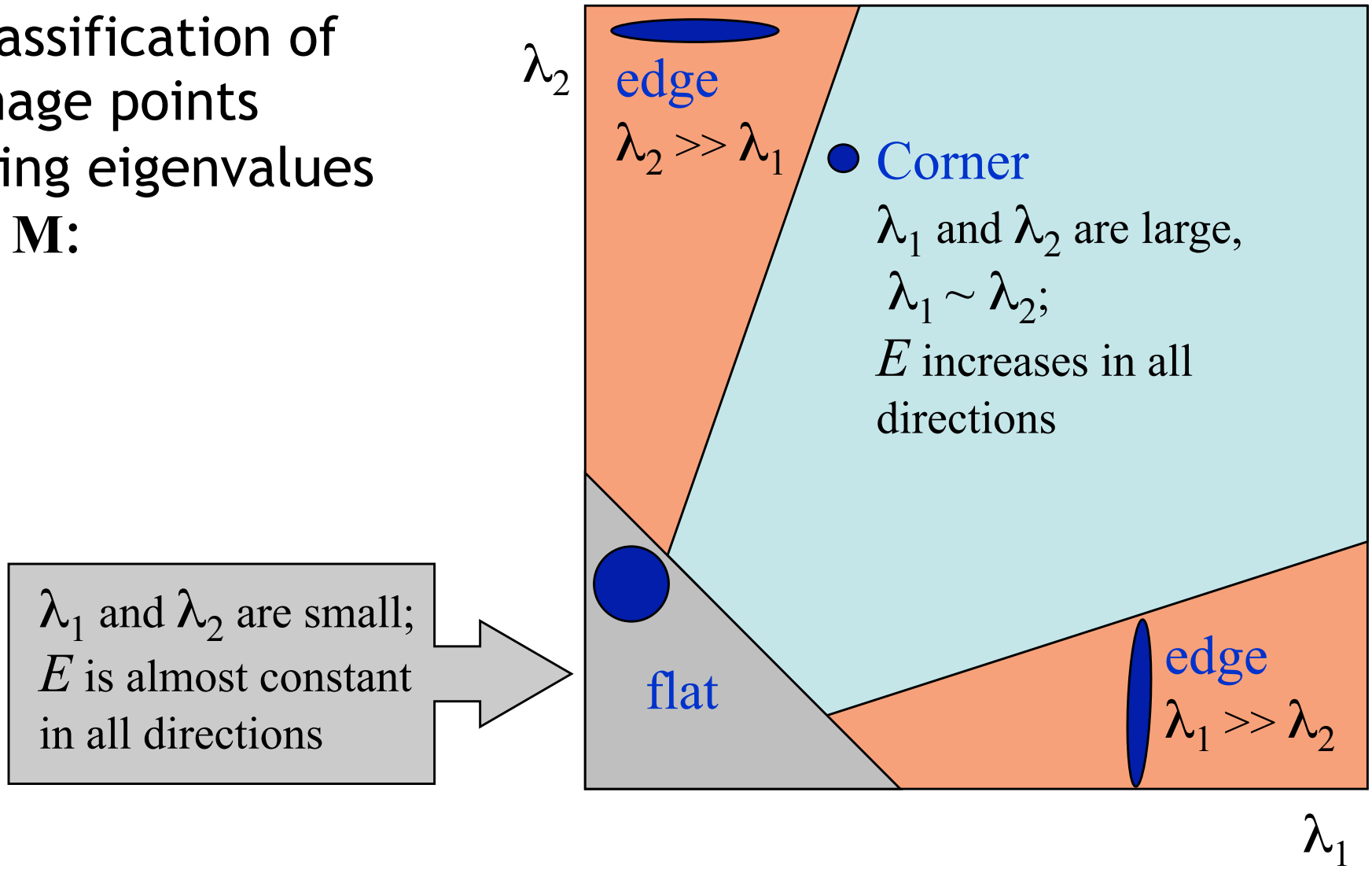
λ_1, λ_2 – eigenvalues of $\mathbf{M}$

Ellipse $E(u,v)$ = const



Harris corner detector

Classification of image points using eigenvalues of **M**:



Harris corner detector

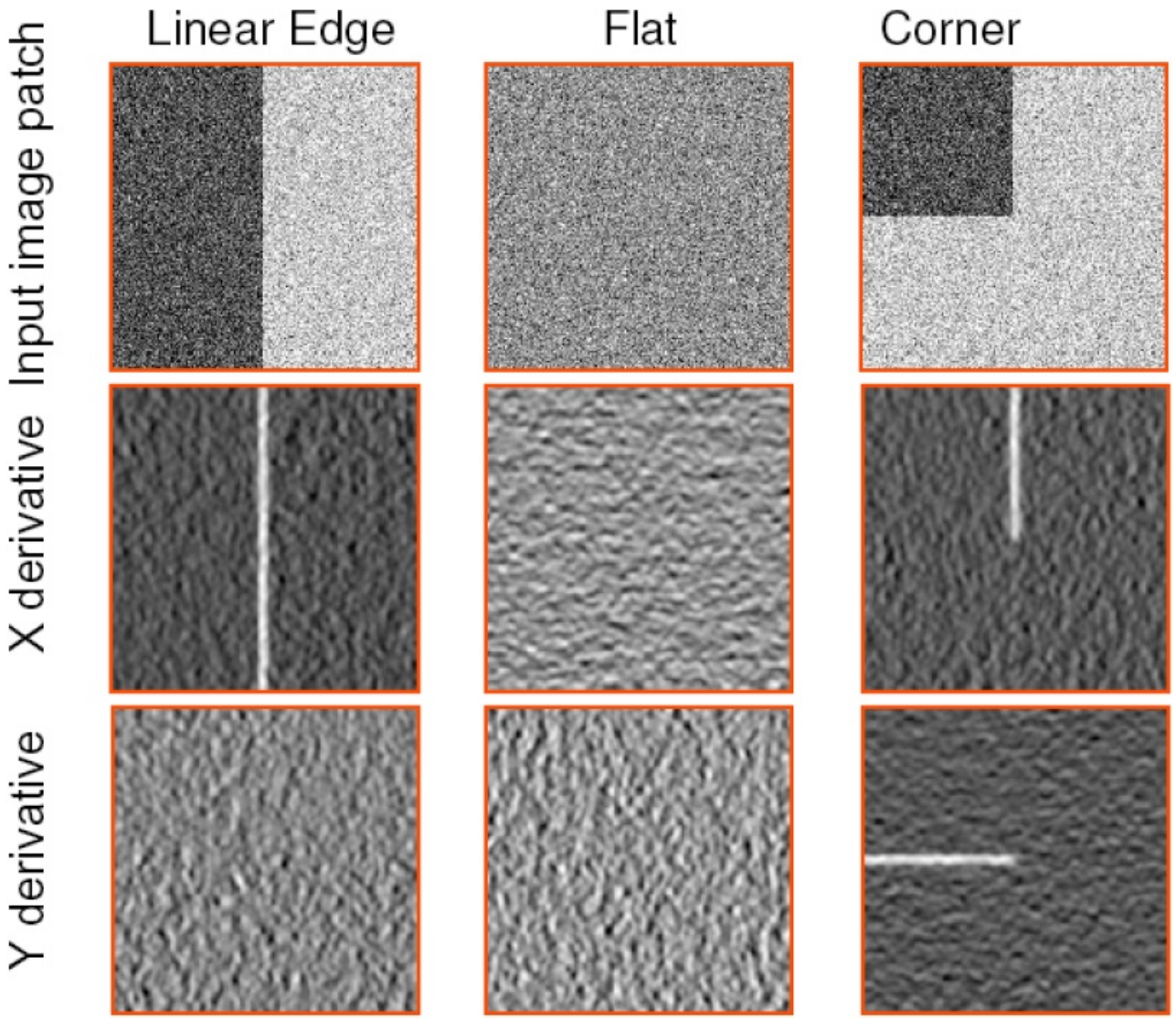
Measure of corner response:

$$R = \det \mathbf{M} - k\left(\text{trace}\,\mathbf{M}\right)^2$$

$$\det \mathbf{M} = \lambda_1 \lambda_2$$
$$\text{trace}\,\mathbf{M} = \lambda_1 + \lambda_2$$

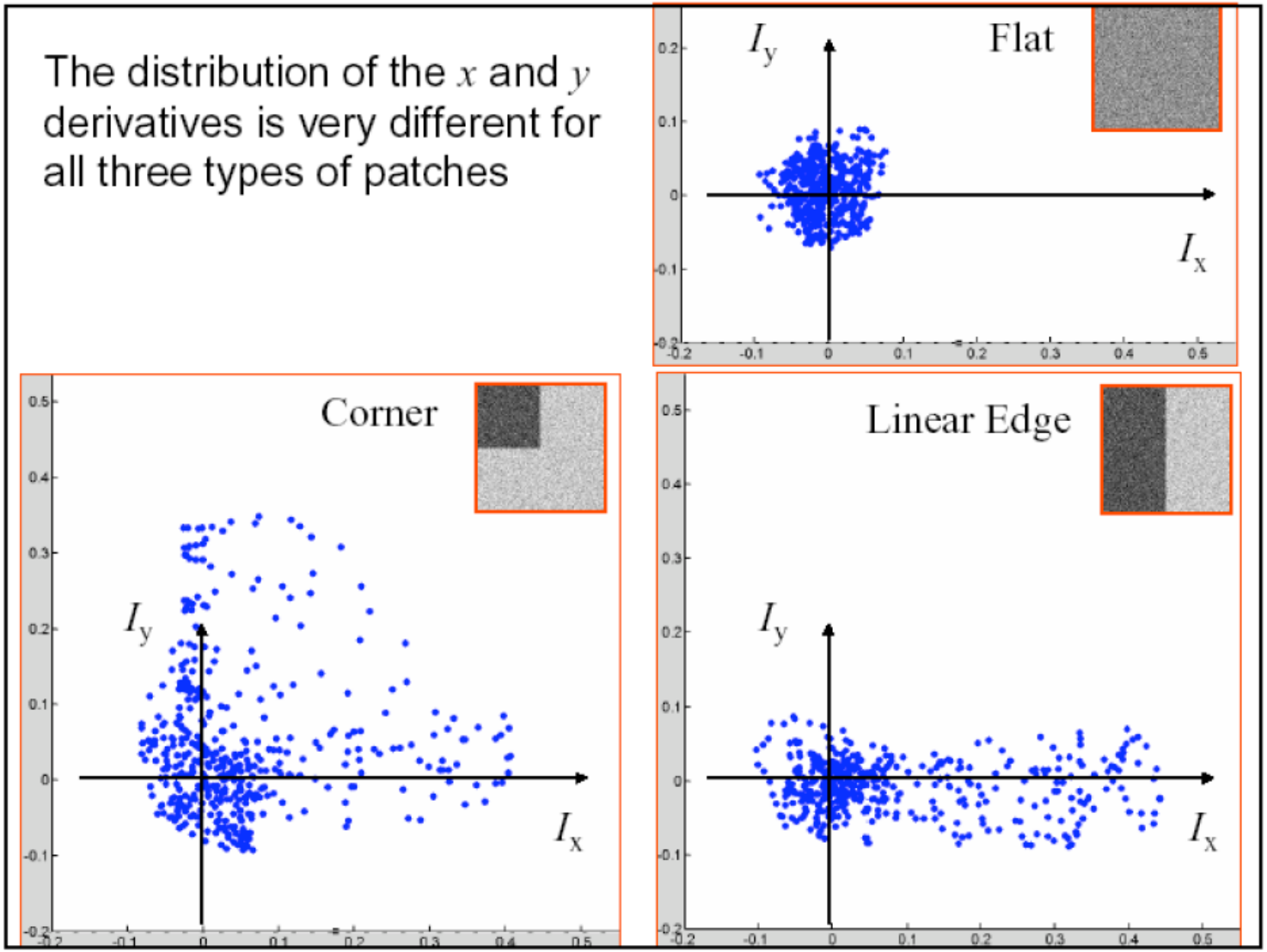
(k – empirical constant, k = 0.04-0.06)

Another view



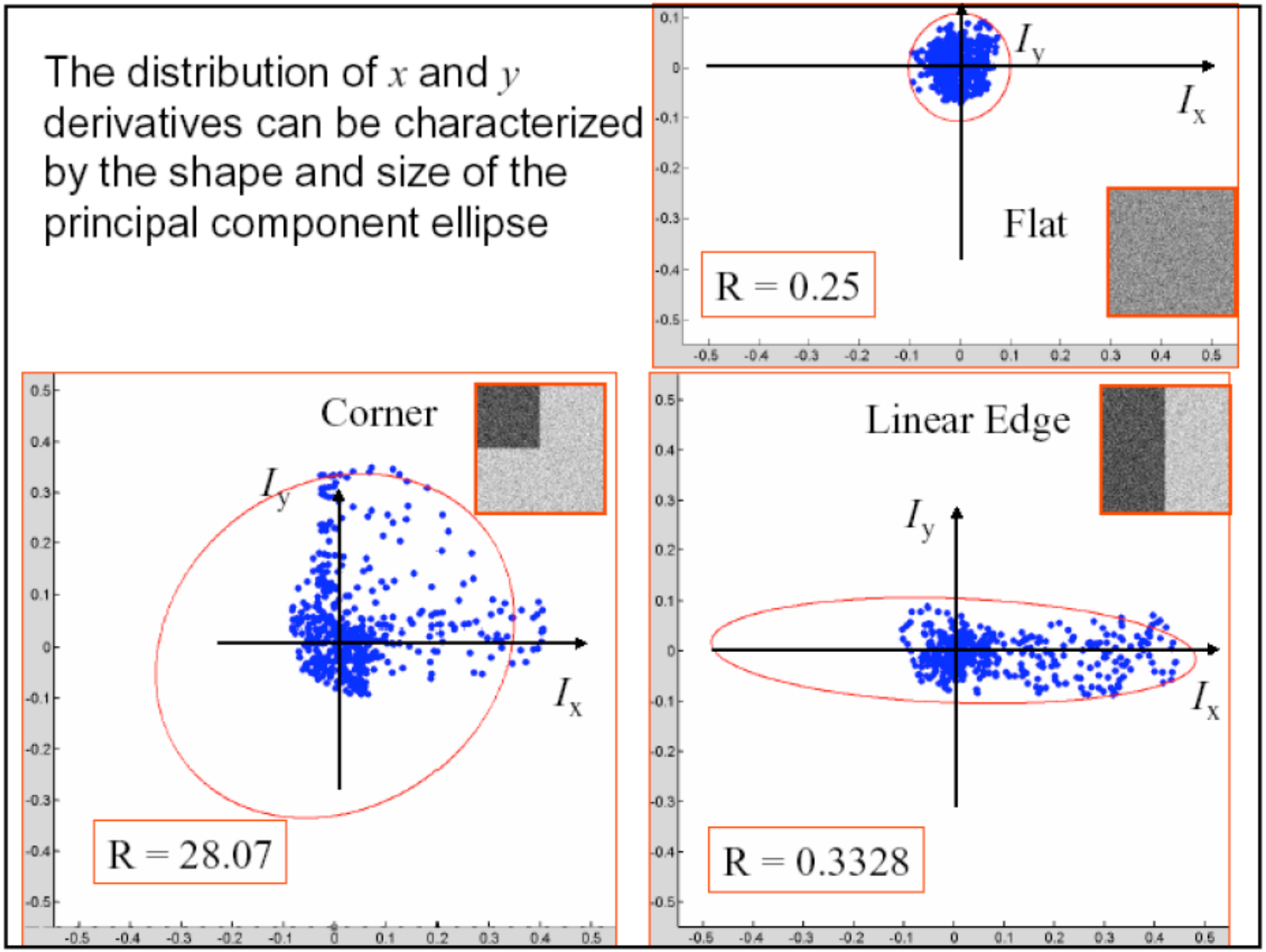
Another view

The distribution of the x and y derivatives is very different for all three types of patches



Another view

The distribution of x and y derivatives can be characterized by the shape and size of the principal component ellipse



Summary of Harris detector

1. Compute x and y derivatives of image

$$I_x = G_\sigma^x * I \qquad I_y = G_\sigma^y * I$$

2. Compute products of derivatives at every pixel

$$I_{x^2} = I_x \cdot I_x \qquad I_{y^2} = I_y \cdot I_y \qquad I_{xy} = I_x \cdot I_y$$

3. Compute the sums of the products of derivatives at each pixel

$$S_{x^2} = G_{\sigma'} * I_{x^2} \qquad S_{y^2} = G_{\sigma'} * I_{y^2} \qquad S_{xy} = G_{\sigma'} * I_{xy}$$

Summary of Harris detector

4. Define the matrix at each pixel

$$M(x,y) = \begin{bmatrix} S_{x^2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y^2}(x,y) \end{bmatrix}$$

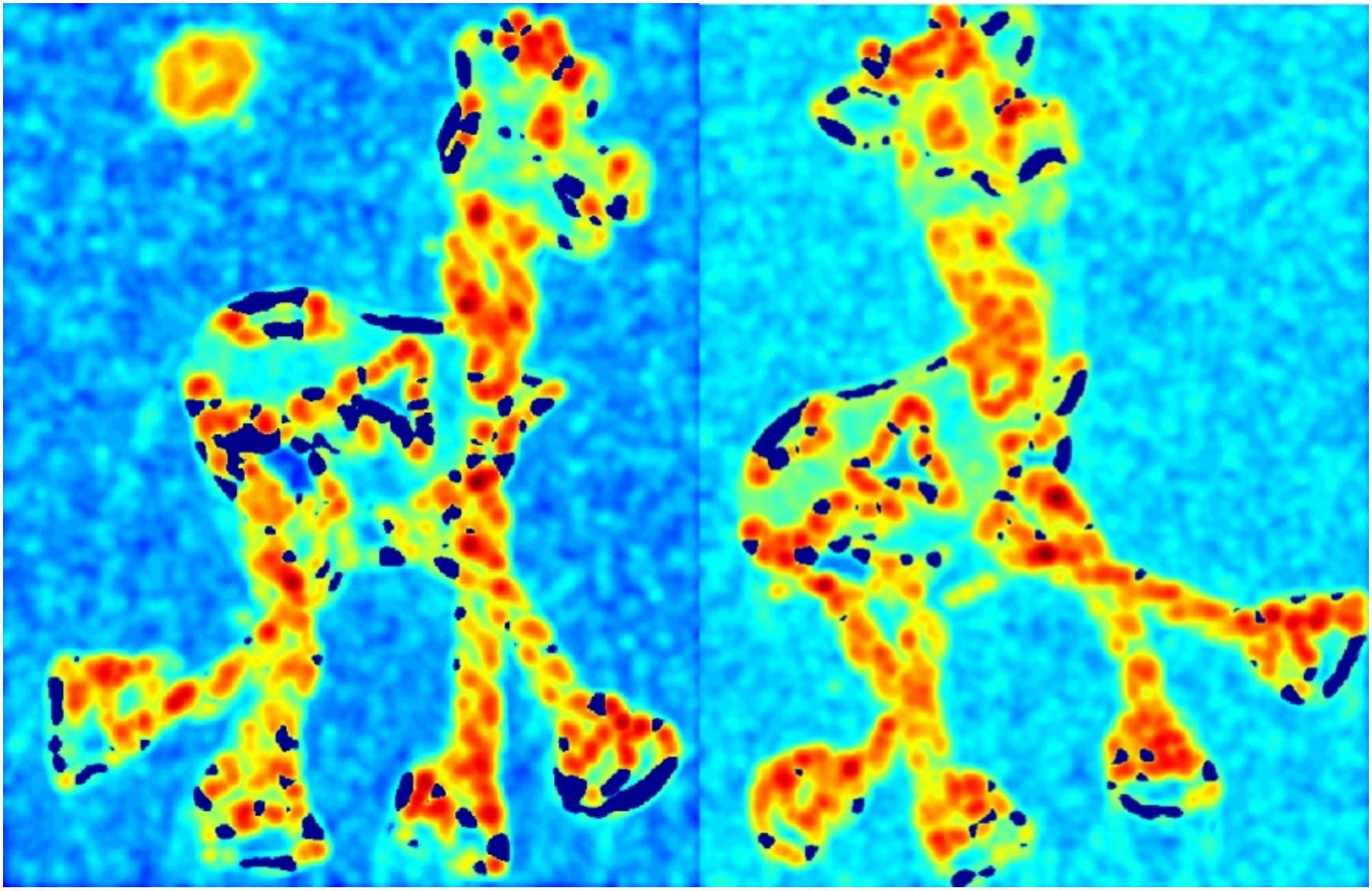
5. Compute the response of the detector at each pixel

$$R = \det M - k\left(\text{trace}M\right)^2$$

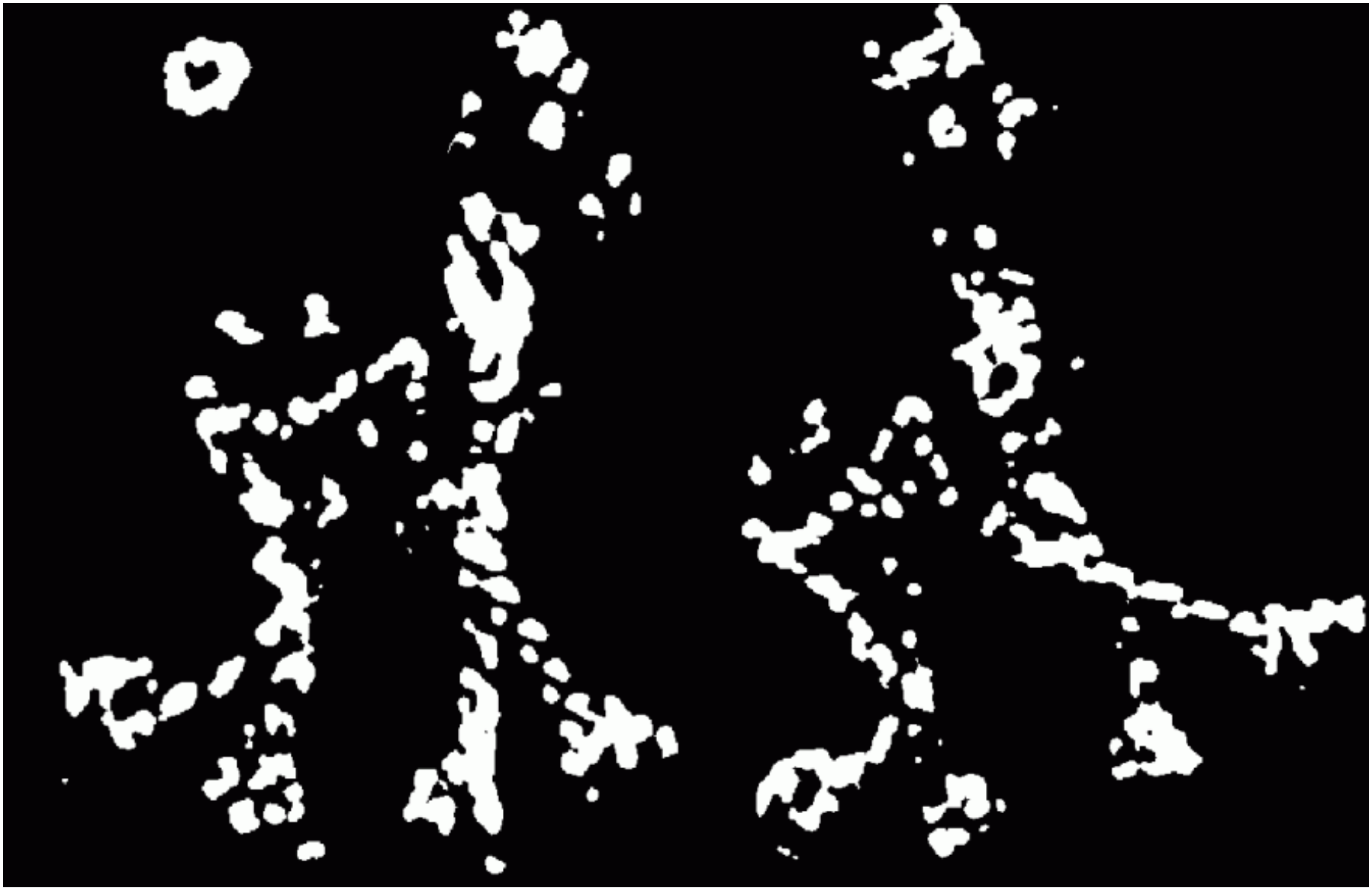
6. Threshold on value of R; compute nonmax suppression.

Harris corner detector (input)

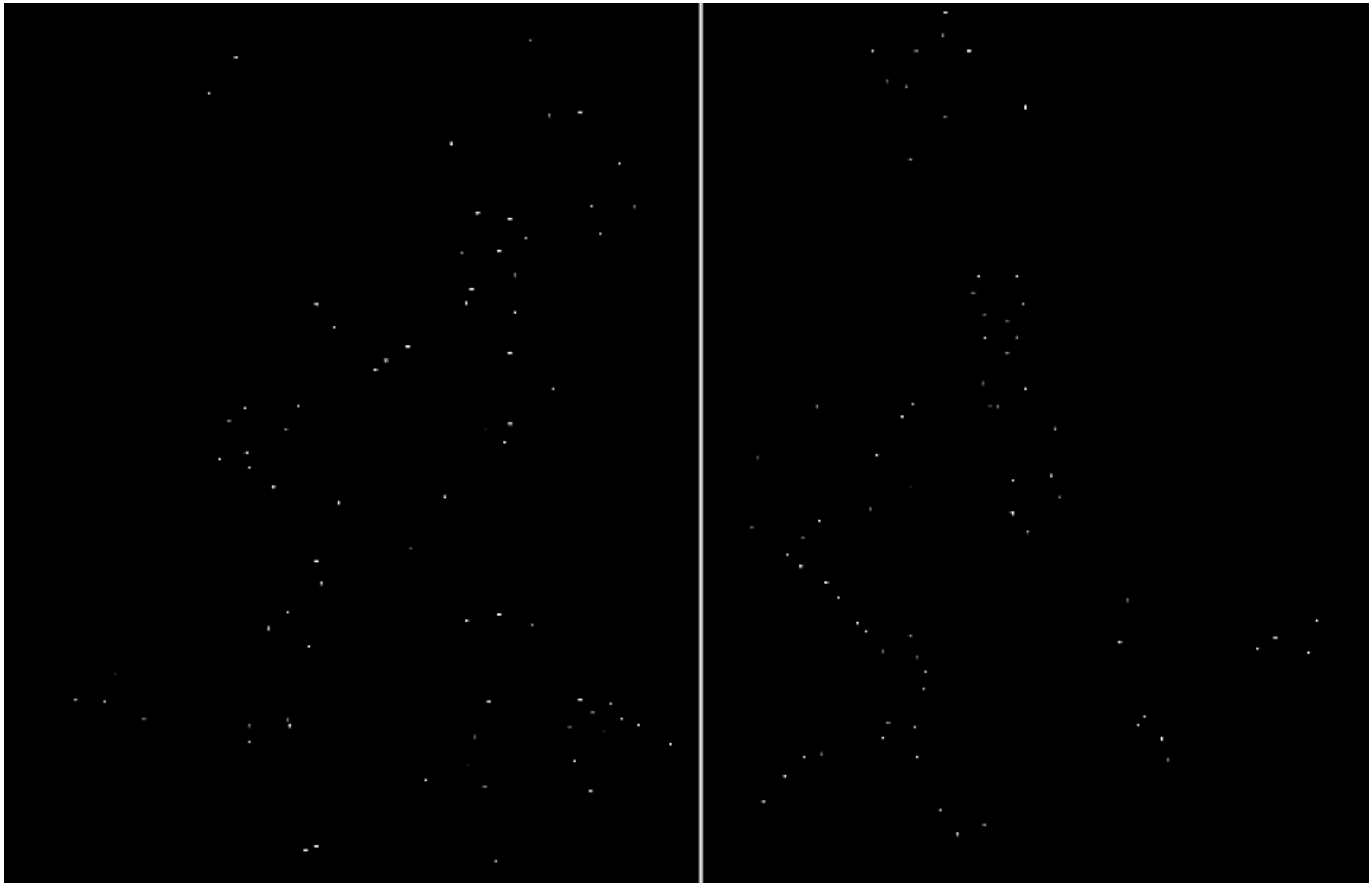
Corner response R



Threshold on R



Local maximum of R



Harris corner detector

Harris detector: summary

- Average intensity change in direction [*u*,*v*] can be expressed as a bilinear form:

$$E(u,v) \cong \begin{bmatrix} u,v \end{bmatrix} \mathbf{M} \begin{bmatrix} u \\ v \end{bmatrix}$$

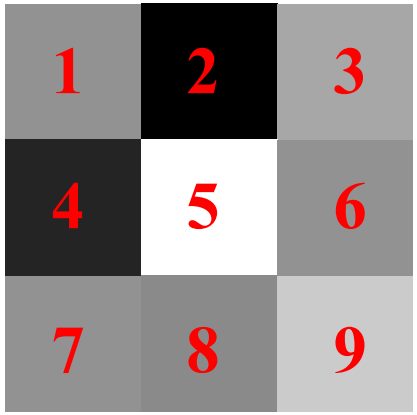
- Describe a point in terms of eigenvalues of *M*: *measure of corner response*

$$R = \lambda_1 \lambda_2 - k\left(\lambda_1 + \lambda_2\right)^2$$

- A good (corner) point should have a *large intensity change* in *all directions*, i.e. *R* should be large positive

Now we know where features are

- But, how to match them?
- What is the descriptor for a feature? The simplest solution is the intensities of its spatial neighbors. This might not be robust to brightness change or small shift/ rotation.

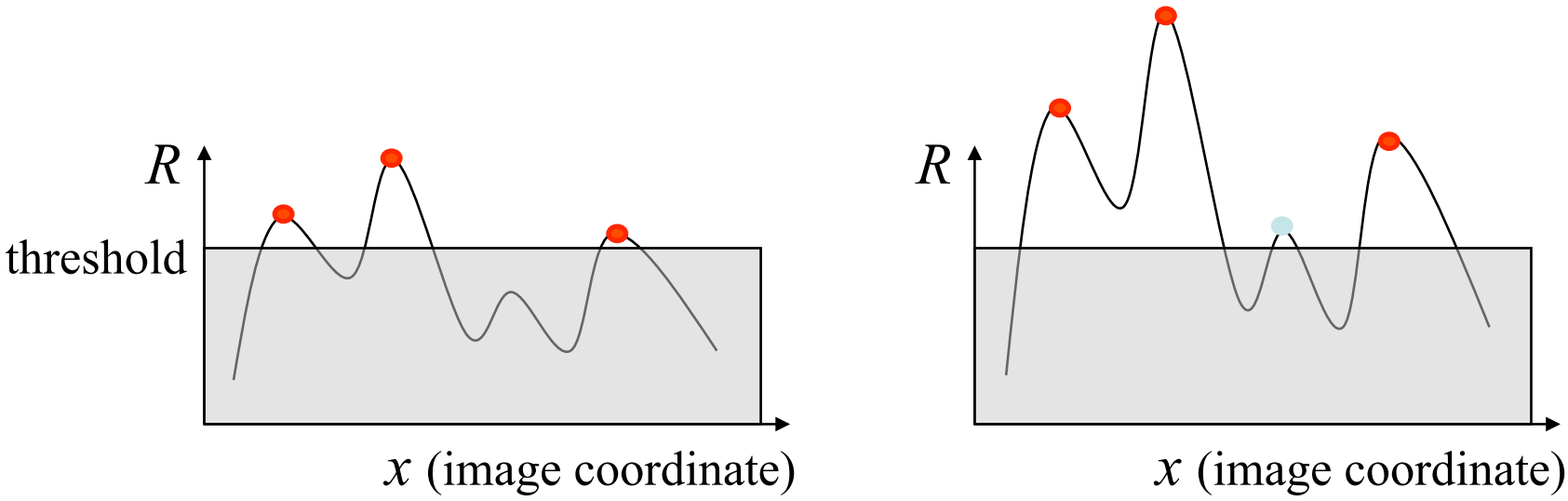


Harris detector: some properties

- Partial invariance to *affine intensity* change

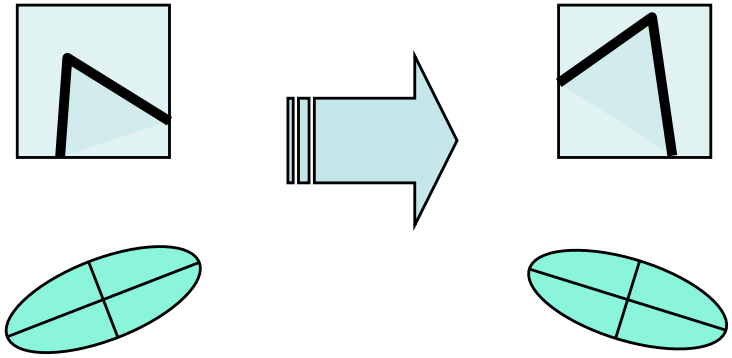
✓ Only derivatives are used =>
invariance to intensity shift $I \rightarrow I + b$

✓ Intensity scale: $I \rightarrow a\, I$



Harris detector: some properties

- Rotation invariance



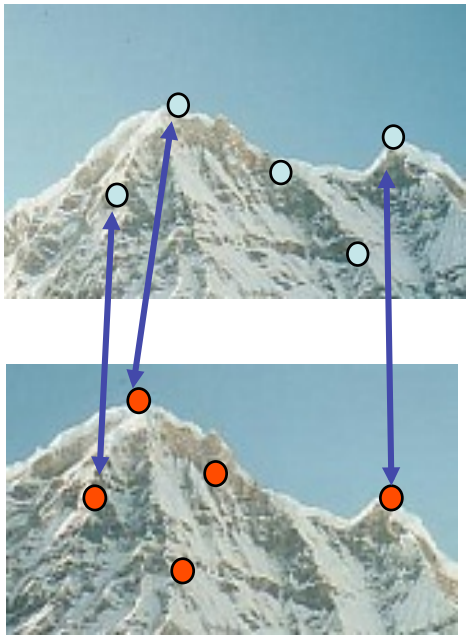
Ellipse rotates but its shape (i.e. eigenvalues) remains the same

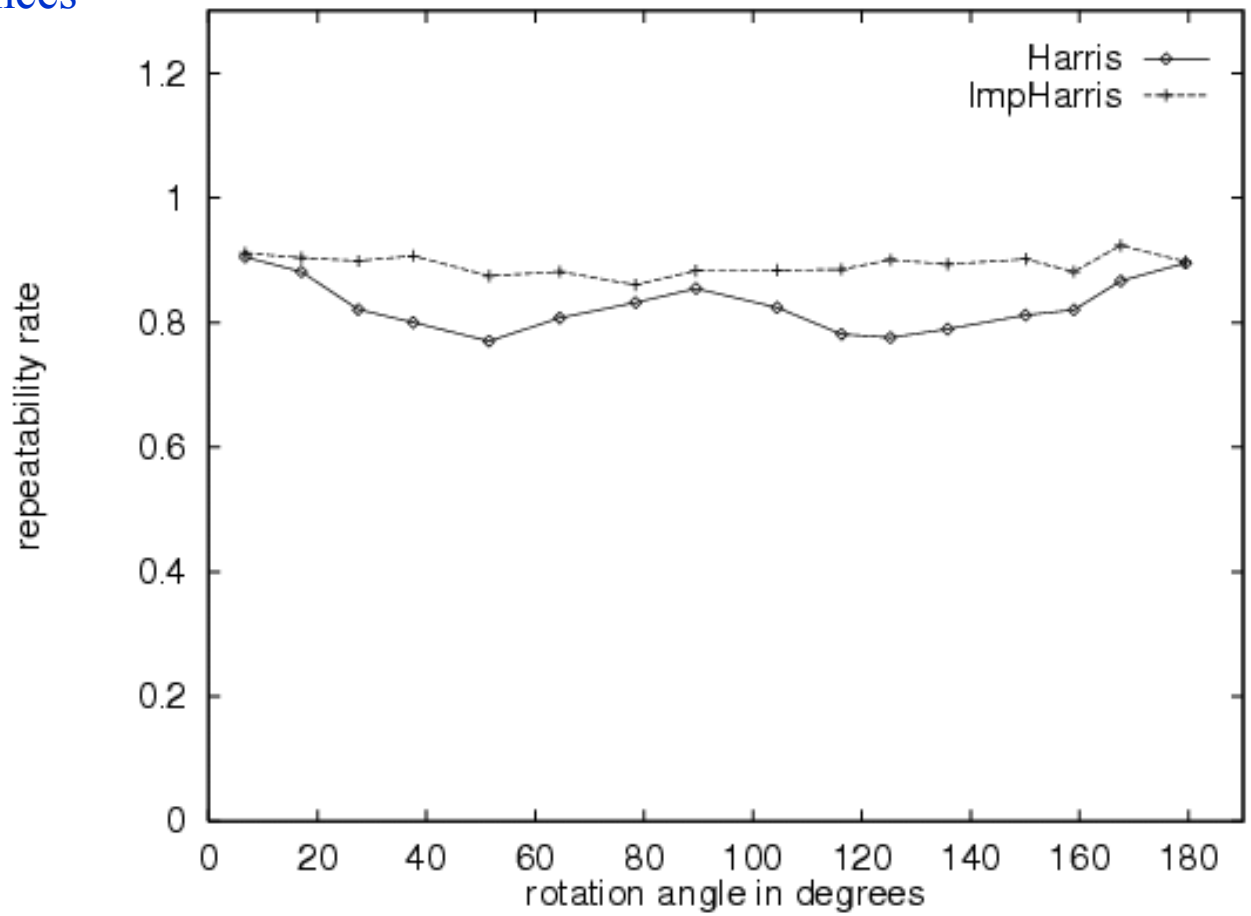
Corner response R is invariant to image rotation

Harris Detector is rotation invariant

Repeatability rate:

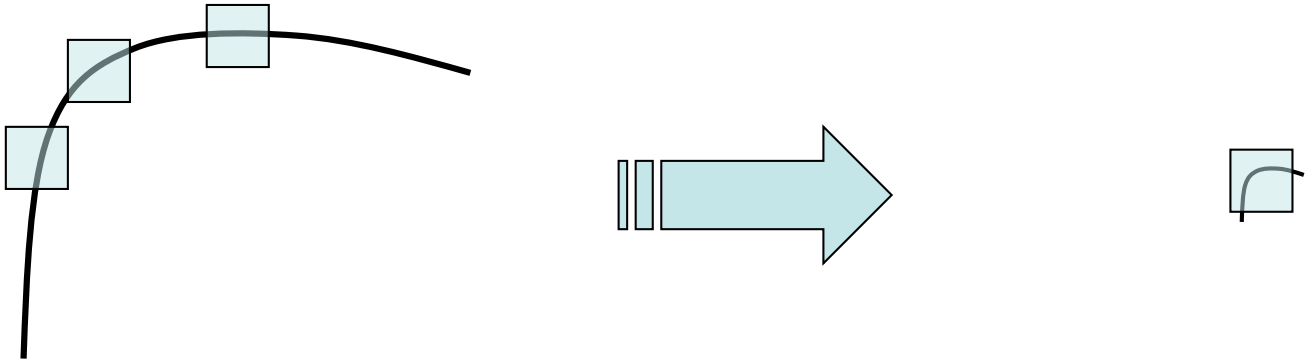
$$\frac{\text{\# correspondences}}{\text{\# possible correspondences}}$$





Harris detector: some properties

- But: not invariant to *image scale*!



All points will be classified as edges

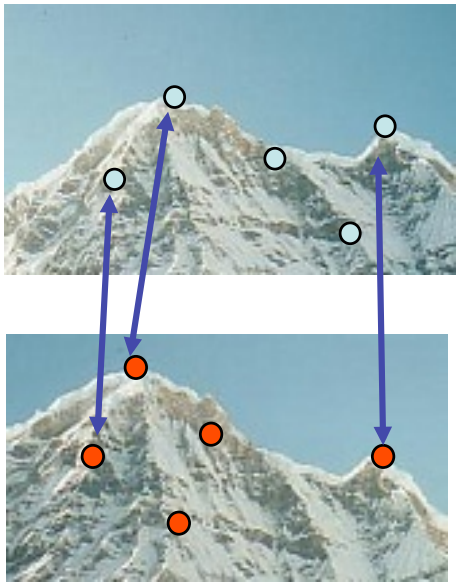
Corner !

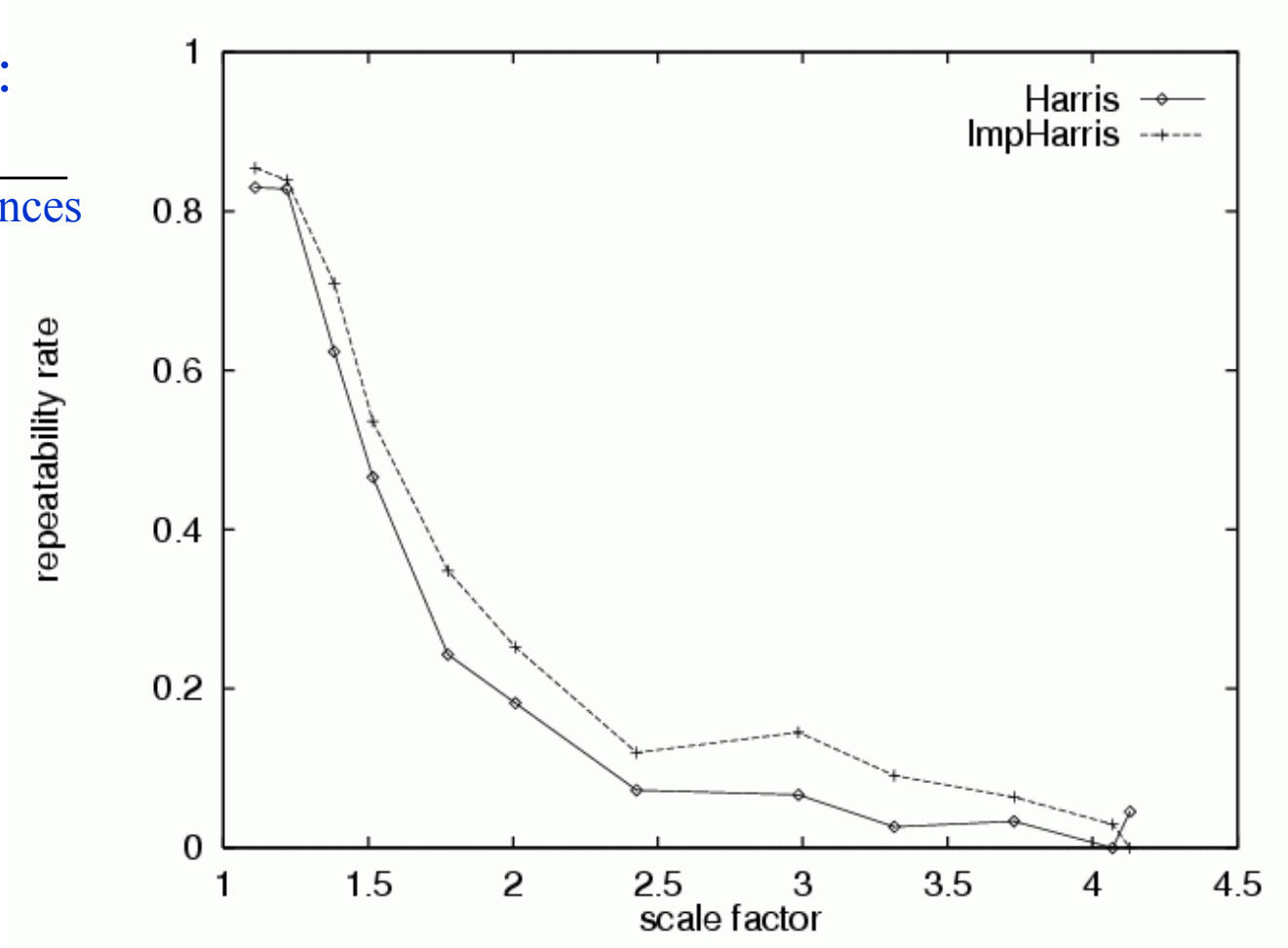
Harris detector: some properties

- Quality of Harris detector for different scale changes

Repeatability rate:

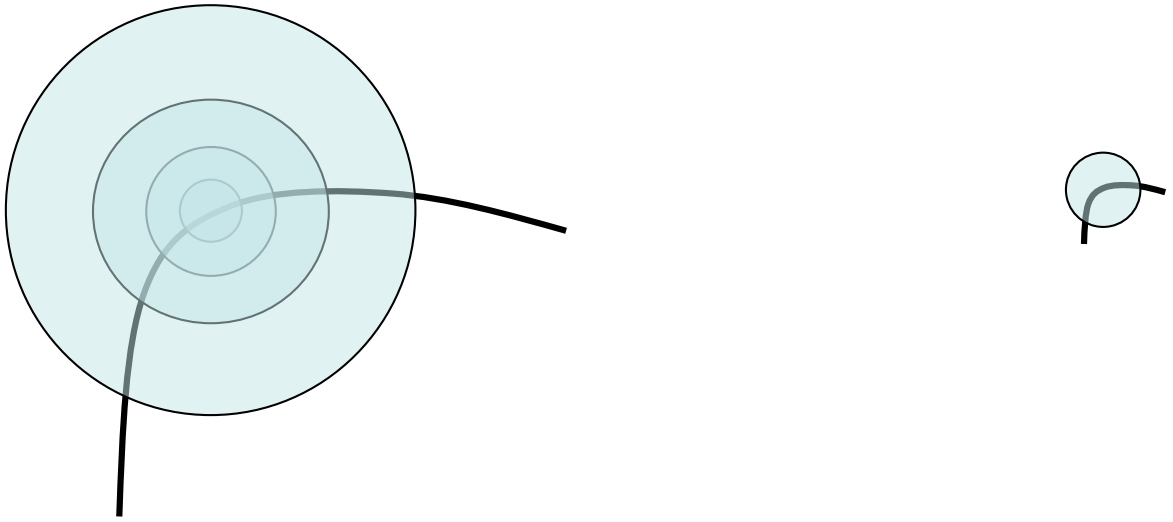
$$\frac{\text{\# correspondences}}{\text{\# possible correspondences}}$$





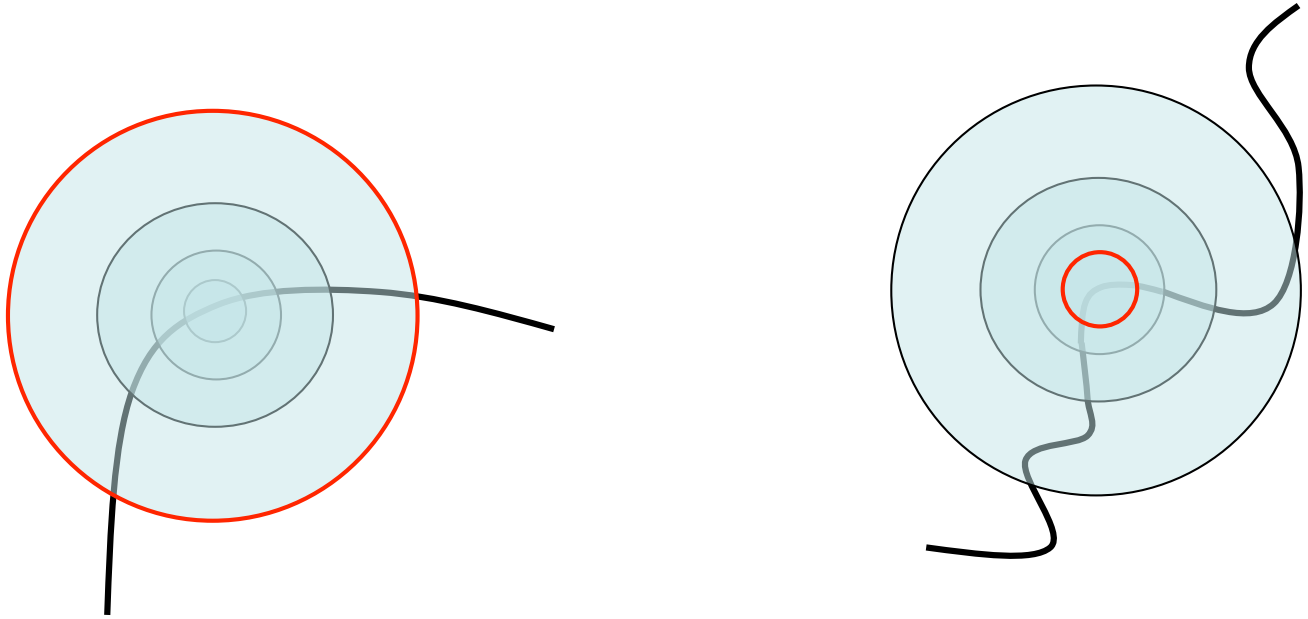
Scale invariant detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images



Scale invariant detection

- The problem: how do we choose corresponding circles ***independently*** in each image?
- Aperture problem



SIFT
(Scale Invariant Feature Transform)

D.G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,”
International Journal of Computer Vision (IJCV), 2004.
(cited number: 66848 from Google)

with slides by Yung-Yu Chuang, Trevor Darrell Cordelia Schmid, David Lowe, Darya Frolova, Denis Simakov, Robert Collins and Jiwon Kim

SIFT

- SIFT is an carefully designed procedure with **empirically determined parameters** for the invariant and distinctive features.

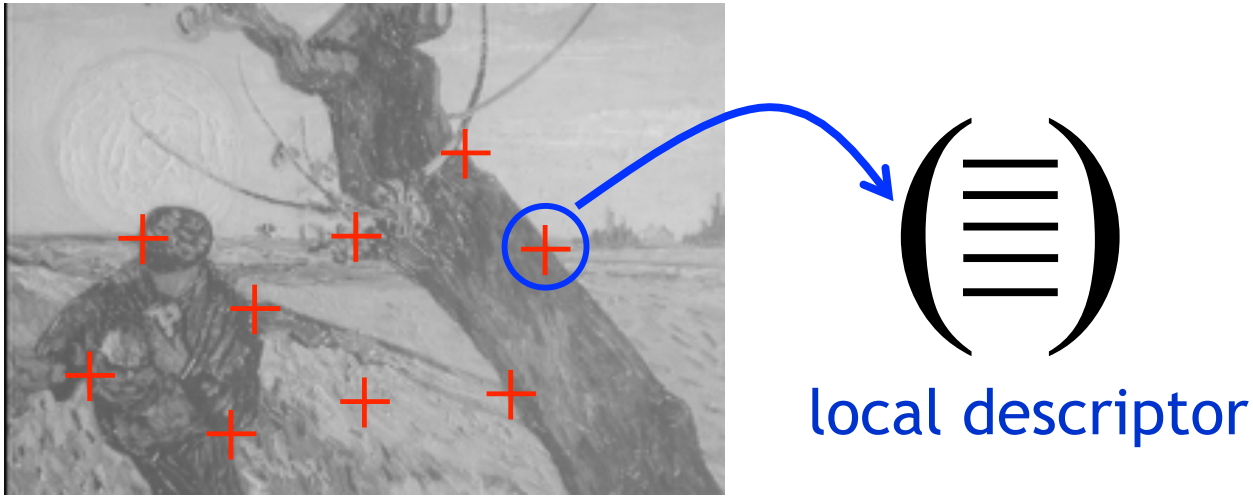
SIFT stages

- Scale-space extrema detection
- Keypoint localization

detector

- Orientation assignment
- Keypoint descriptor

descriptor



A 500x500 image gives about 2000 features

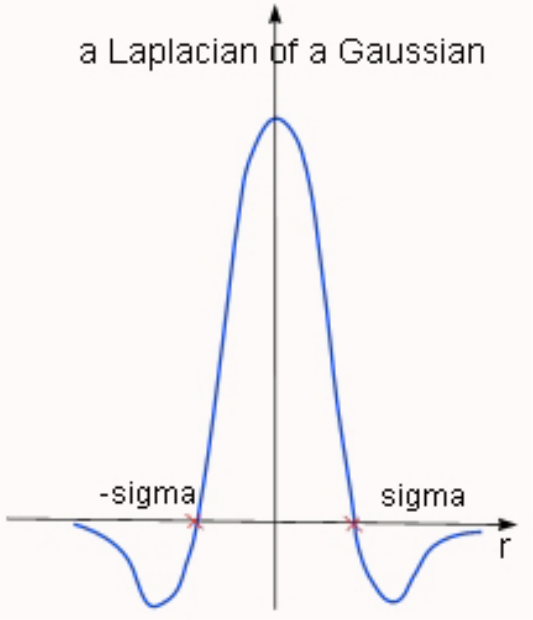
1. Detection of scale-space extrema

- For scale invariance, search for stable features across all possible scales using a continuous function of scale, scale space.

  - Pyramids
    - Divide width and height by 2
    - Take average of 4 pixels for each pixel (or Gaussian blur)
    - Repeat until image is tiny
    - Run filter over each size image and hope its robust

1. Detection of scale-space extrema

- SIFT uses DoG filter (Difference-of-Gaussian) for scale space because
  - it is efficient
  - it is an approximation to the scale-normalized Laplacian of Gaussian.



DoG filtering

Convolution with a variable-scale Gaussian

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

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

Difference-of-Gaussian (DoG) filter

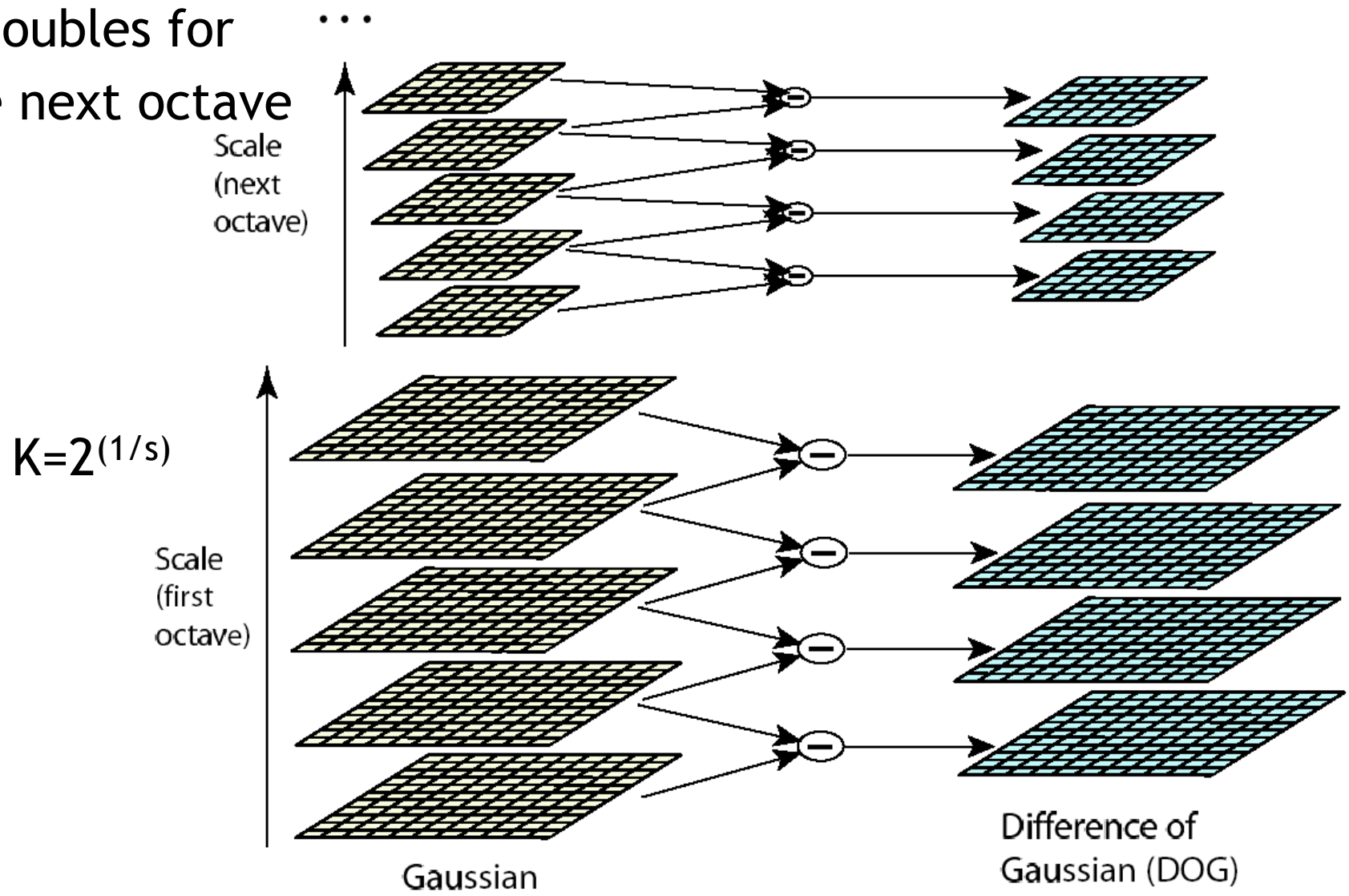
$$G(x, y, k\sigma) - G(x, y, \sigma)$$

Convolution with the DoG filter

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

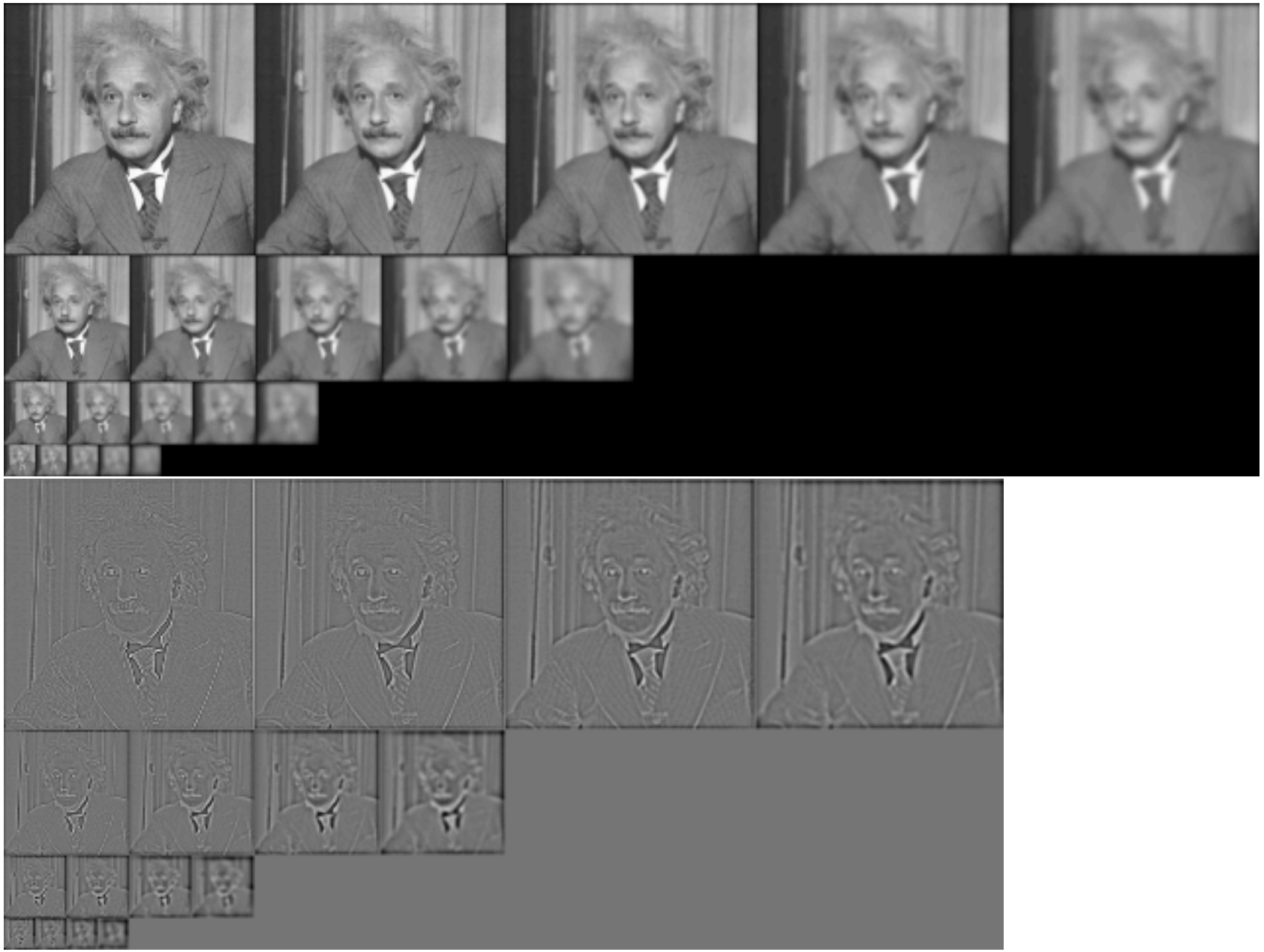
Scale space processed one octave at a time

σ doubles for the next octave

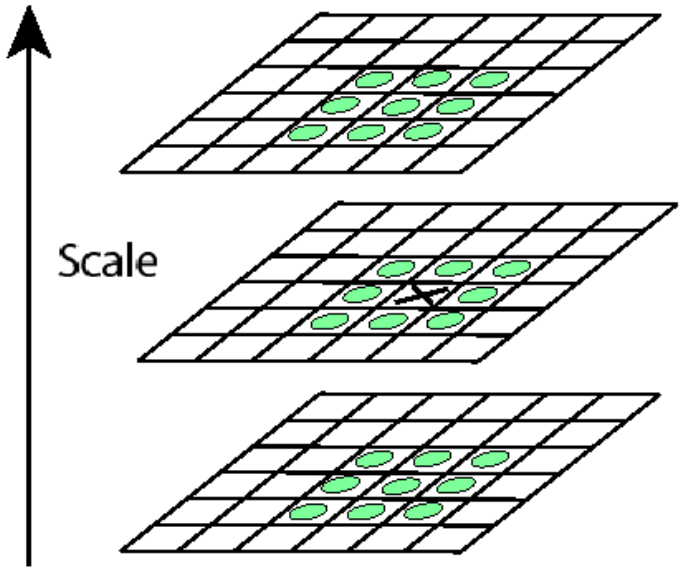


Dividing into octave is for efficiency only.

Detection of scale-space extrema



Keypoint localization

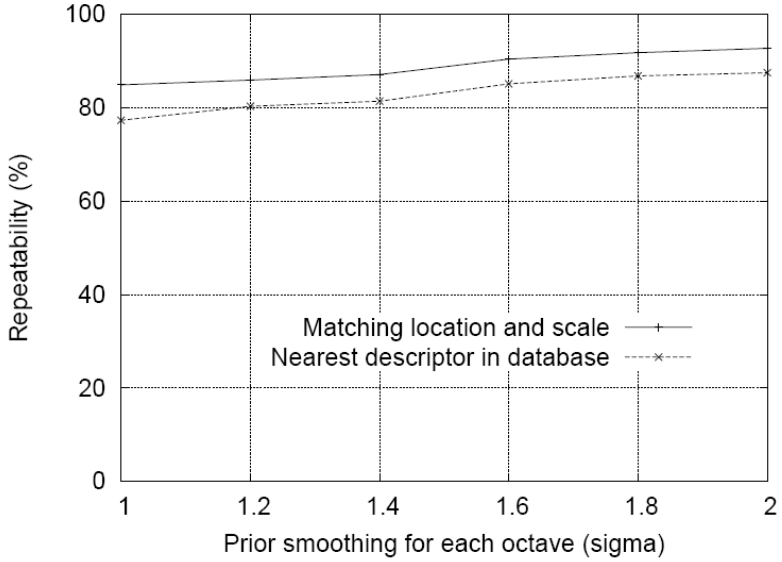


- Detect maxima and minima of difference-of-Gaussian in scale space
  - X is selected if it is larger or smaller than all 26 neighbours

Decide scale sampling frequency

- It is impossible to sample the whole space, tradeoff efficiency with completeness.

- Decide the best sampling frequency by experimenting on 32 real image subject to synthetic transformations. (rotation, scaling, affine stretch, brightness and contrast change, adding noise…)
  - s=3 is the best
  - σ=1.6



2. Accurate Keypoint Localization & Filtering

- Now we have much less points than pixels by detecting maxima and minima of difference-of-Gaussian in scale space.
- However, still lots of points…
  - With only pixel-accuracy at best
  - And this includes many bad points

- Reject points with low contrast (flat) and poorly localized along an edge (edge)
- Fit a 3D quadratic function for sub-pixel maxima

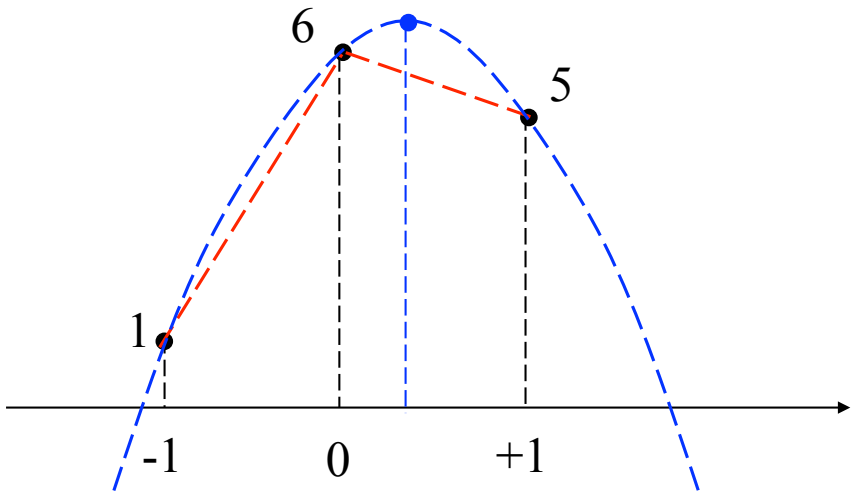
Keypoint Filtering

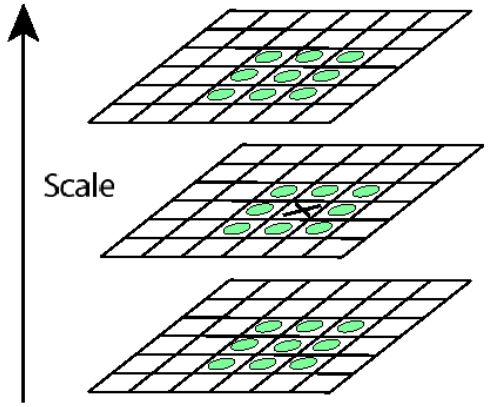
- **Reject points with bad contrast:**
  - DoG smaller than 0.03 (image values in [0,1])

- **Reject edges**
  - Similar to the Harris detector; look at the autocorrelation matrix

Accurate Keypoint Localization

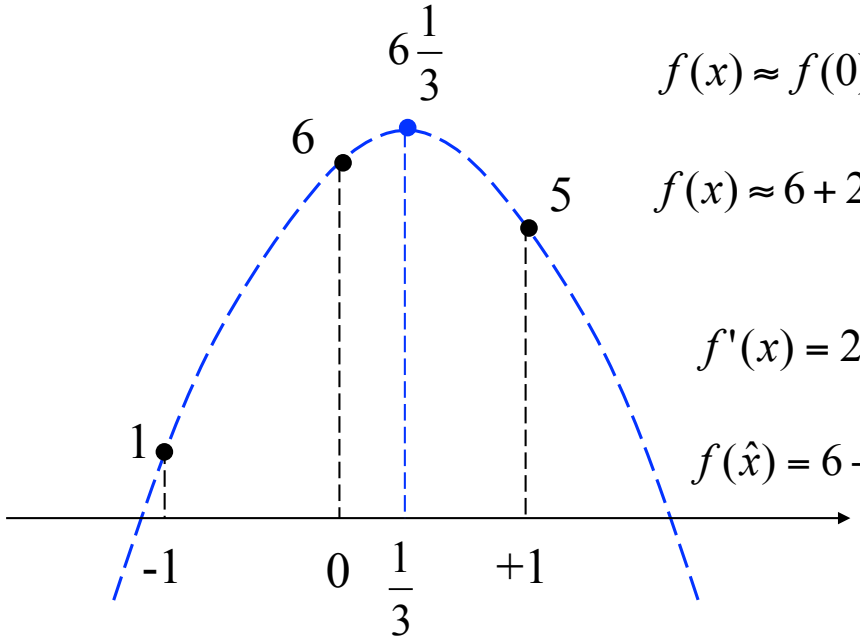
- **Fit a 3D quadratic function for sub-pixel maxima**





Accurate Keypoint Localization

- **Fit a 3D quadratic function for sub-pixel maxima**



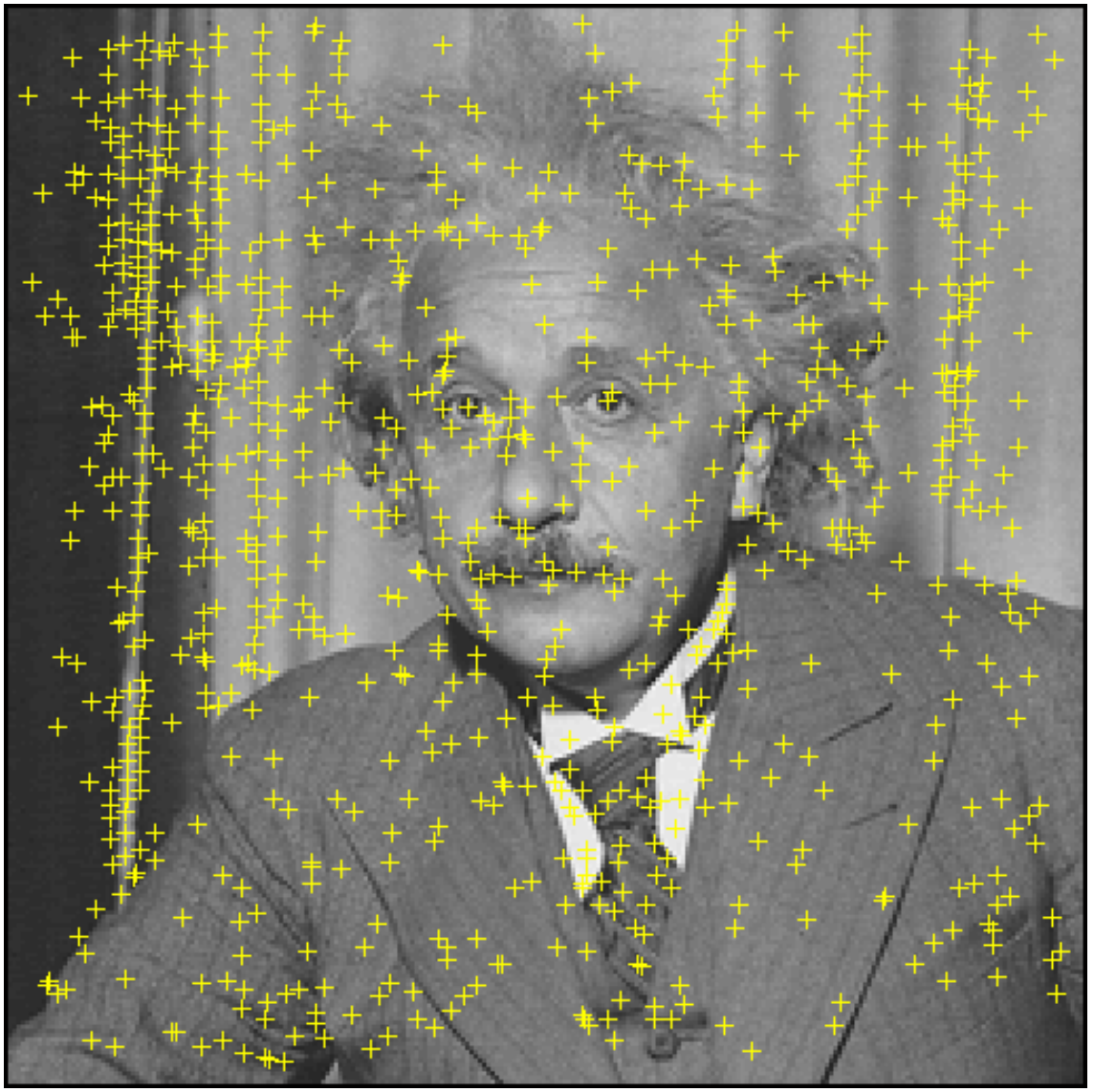
$$f(x) \approx f(0) + f'(0)x + \frac{f''(0)}{2}x^2$$

$$f(x) \approx 6 + 2x + \frac{-6}{2}x^2 = 6 + 2x - 3x^2$$

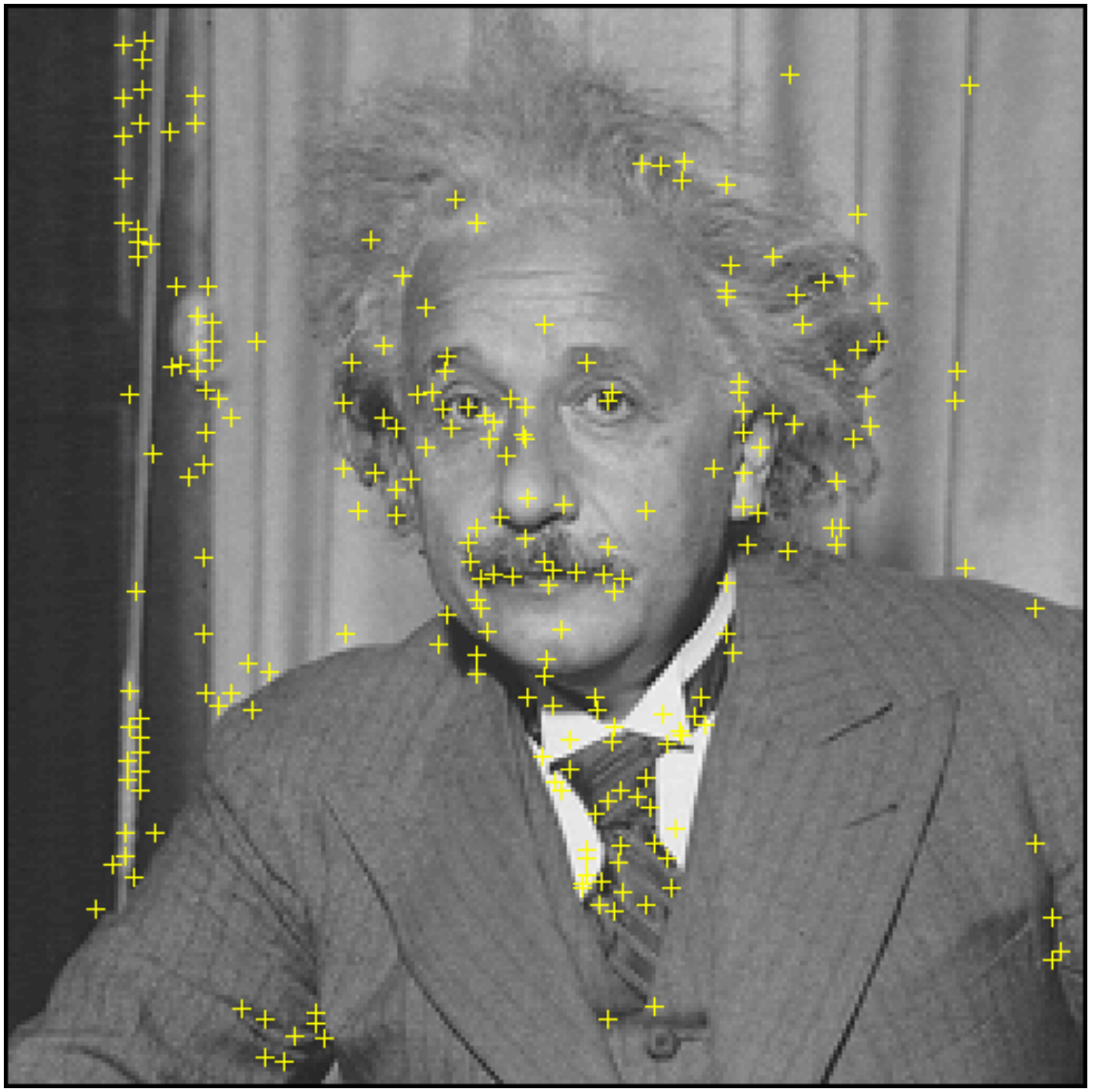
$$f'(x) = 2 - 6x = 0 \longrightarrow \hat{x} = \frac{1}{3}$$

$$f(\hat{x}) = 6 + 2 \cdot \frac{1}{3} - 3 \cdot \left(\frac{1}{3}\right)^2 = 6\frac{1}{3}$$

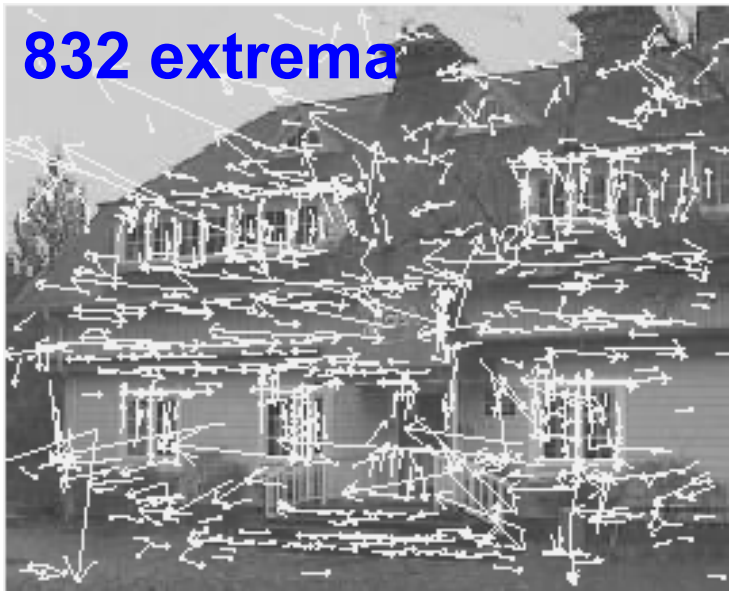
Maxima in D

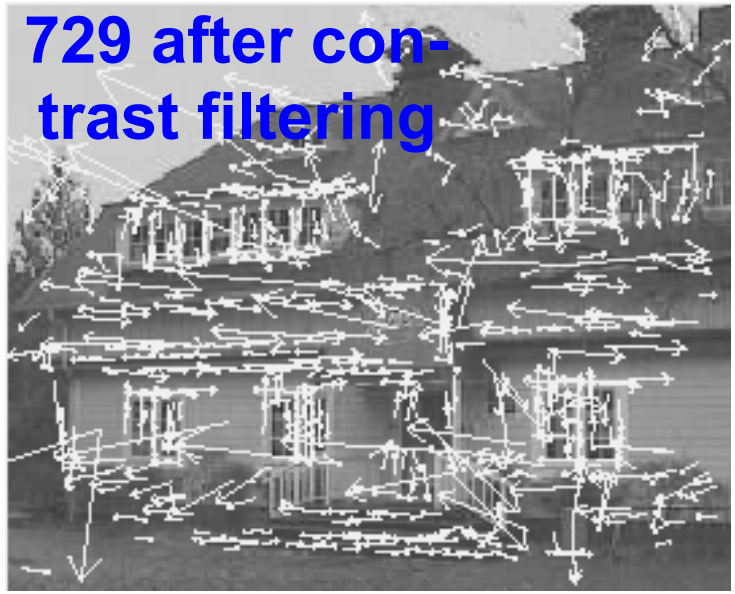


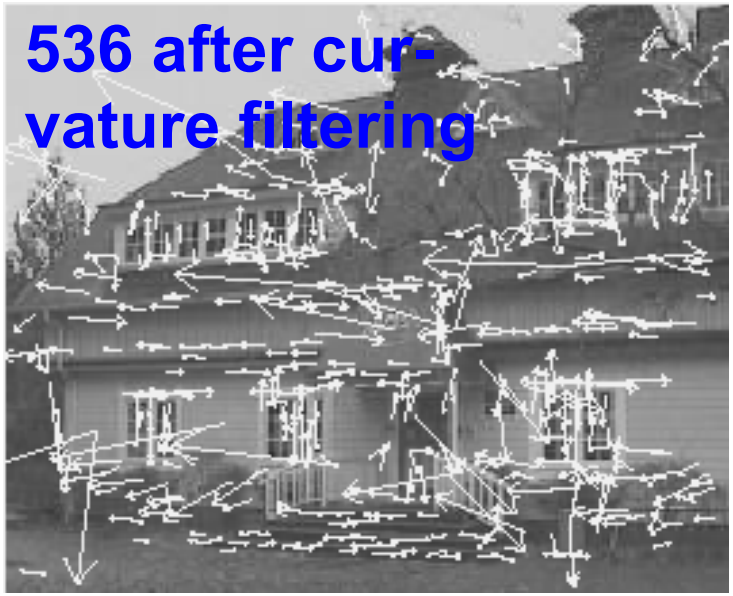
Remove low contrast and edges



Keypoint detector

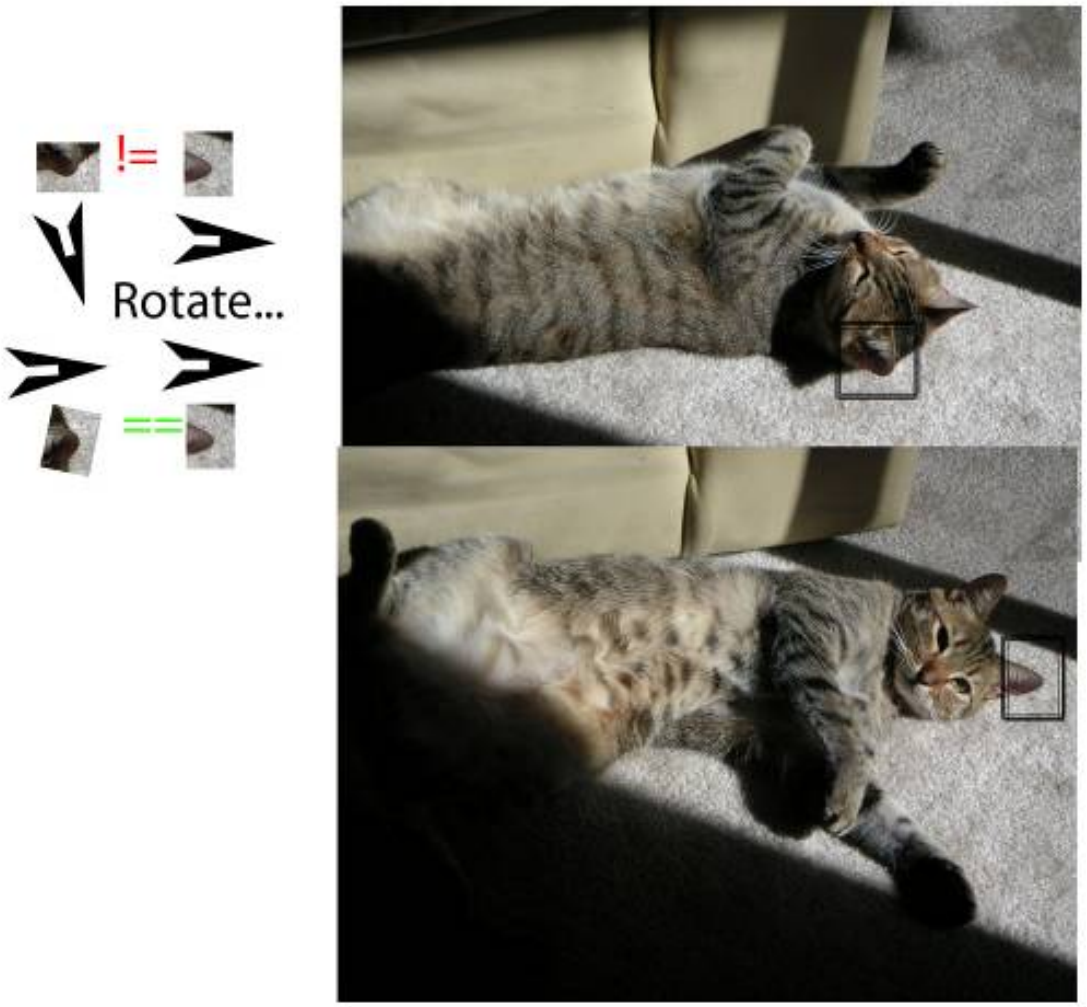






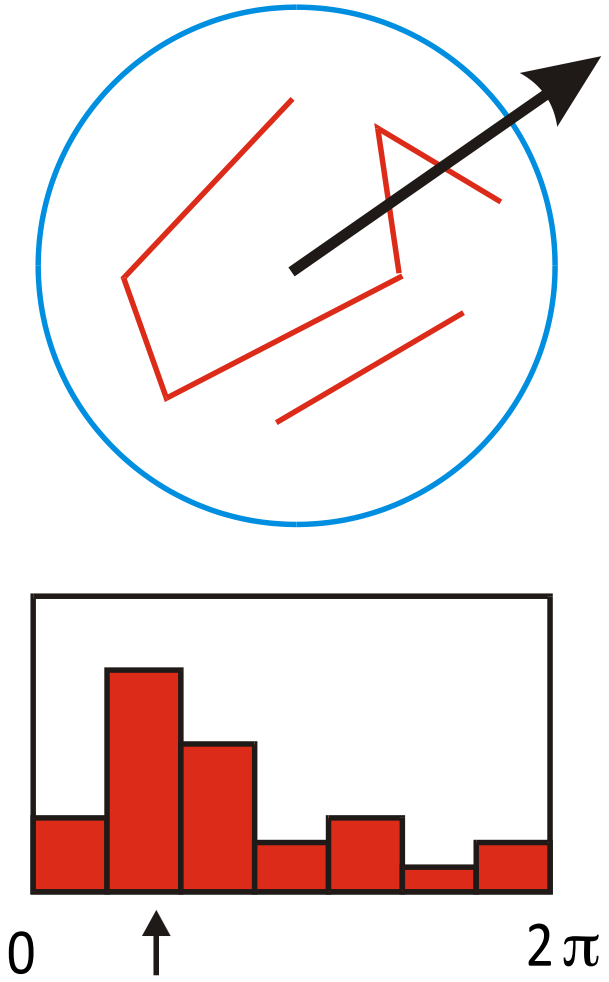
3. Orientation assignment

- For rotation invariance



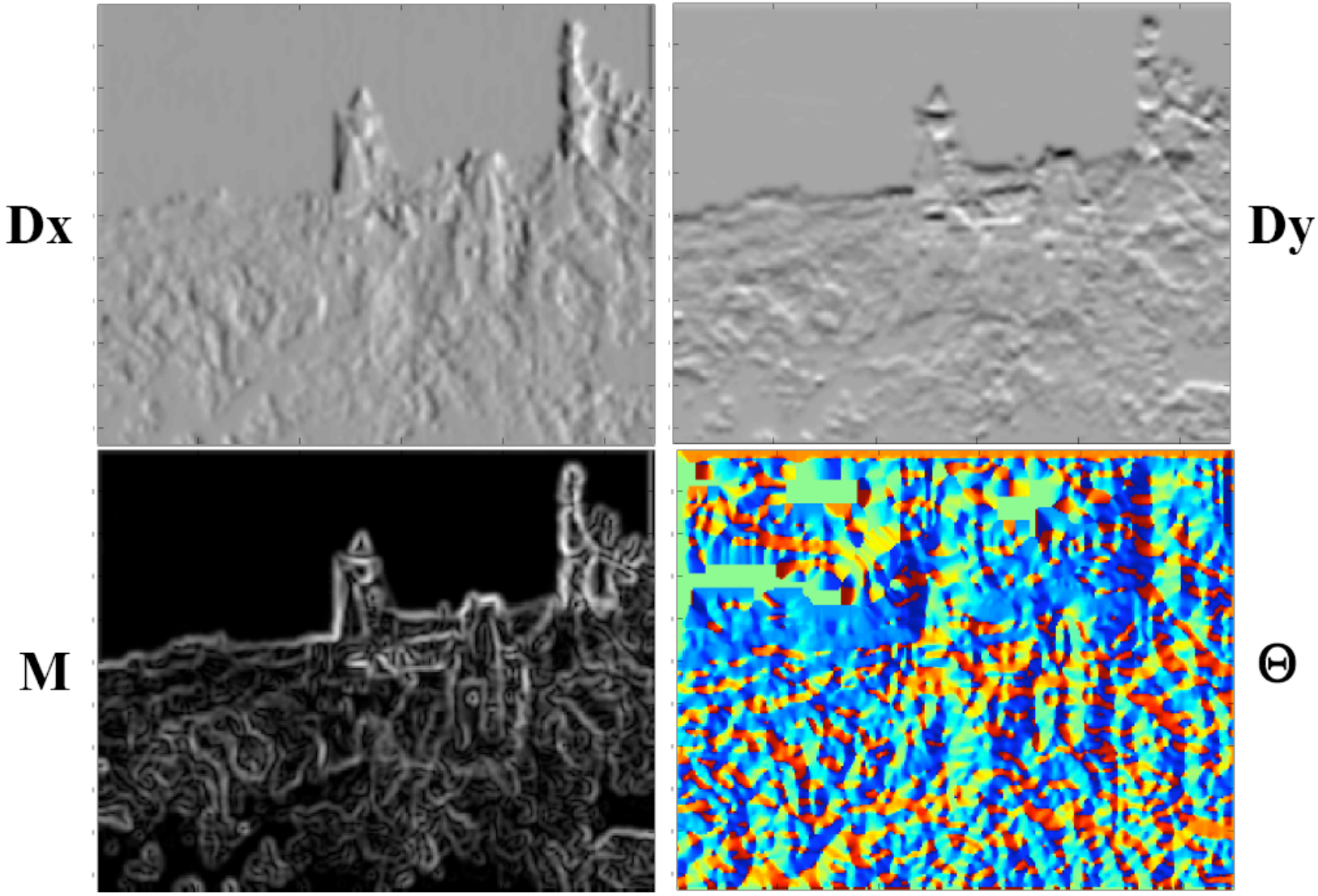
Select canonical orientation

- Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)

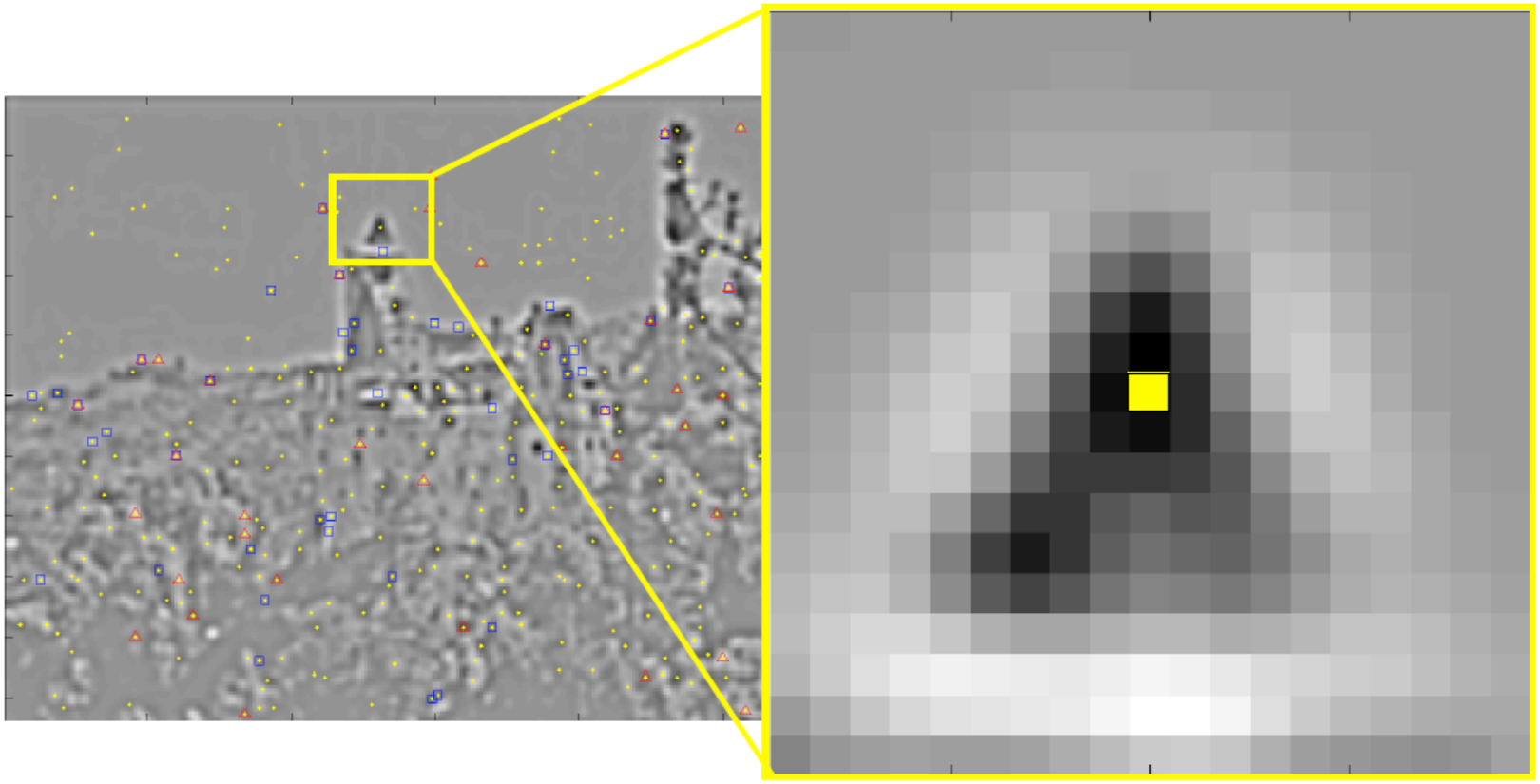


orientation histogram (36 bins)

Orientation assignment

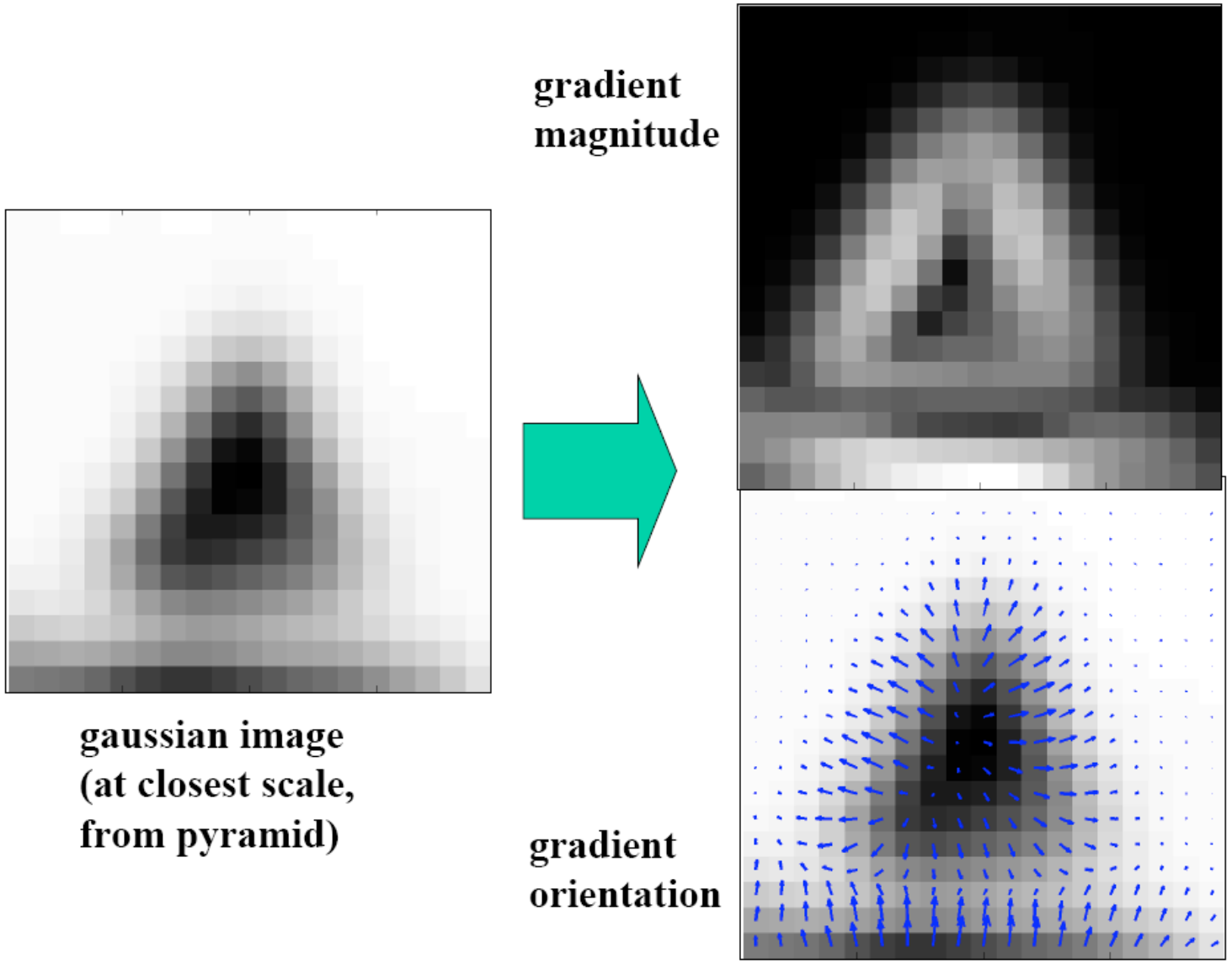


Orientation assignment

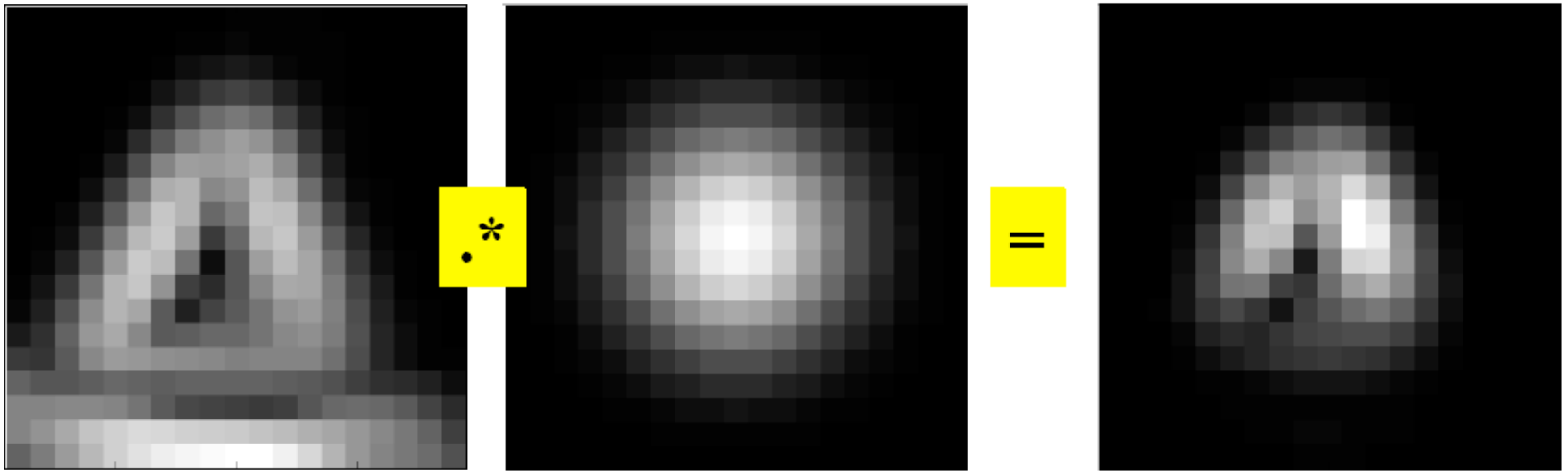


- **Keypoint location = extrema location**
- **Keypoint scale is scale of the DOG image**

Orientation assignment



Orientation assignment



gradient magnitude

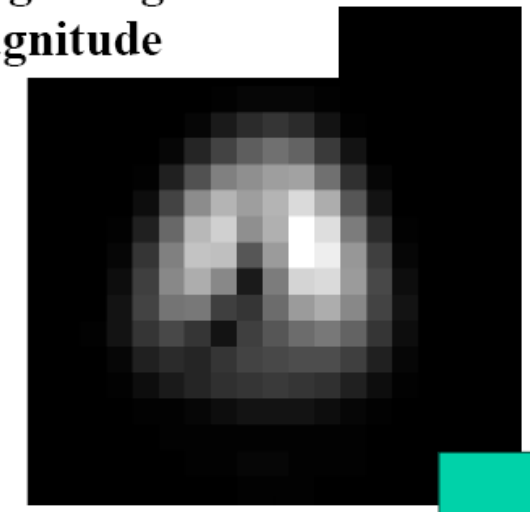
weighted by 2D gaussian kernel

weighted gradient magnitude

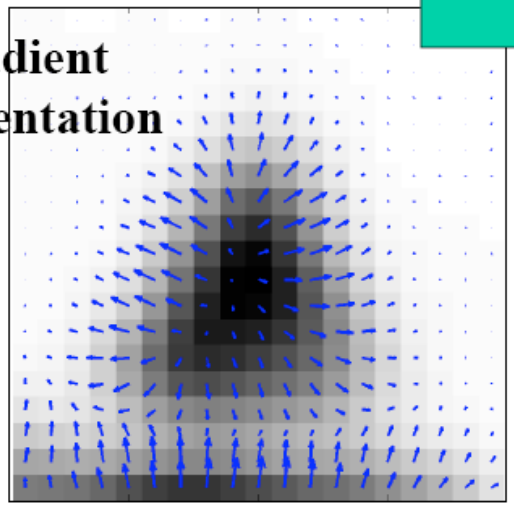
σ=1.5*scale of the keypoint

Orientation assignment

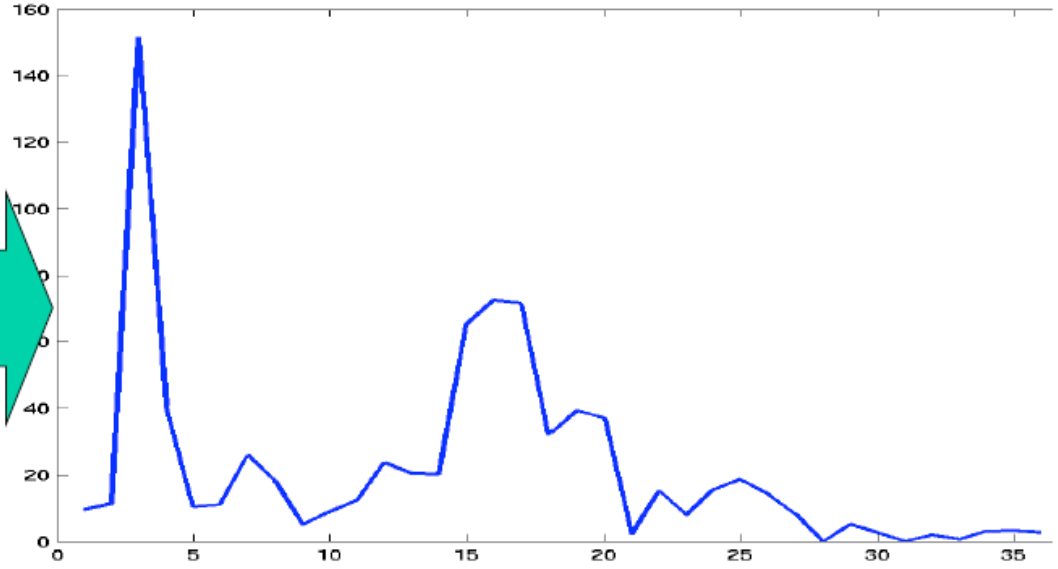
weighted gradient magnitude



gradient orientation

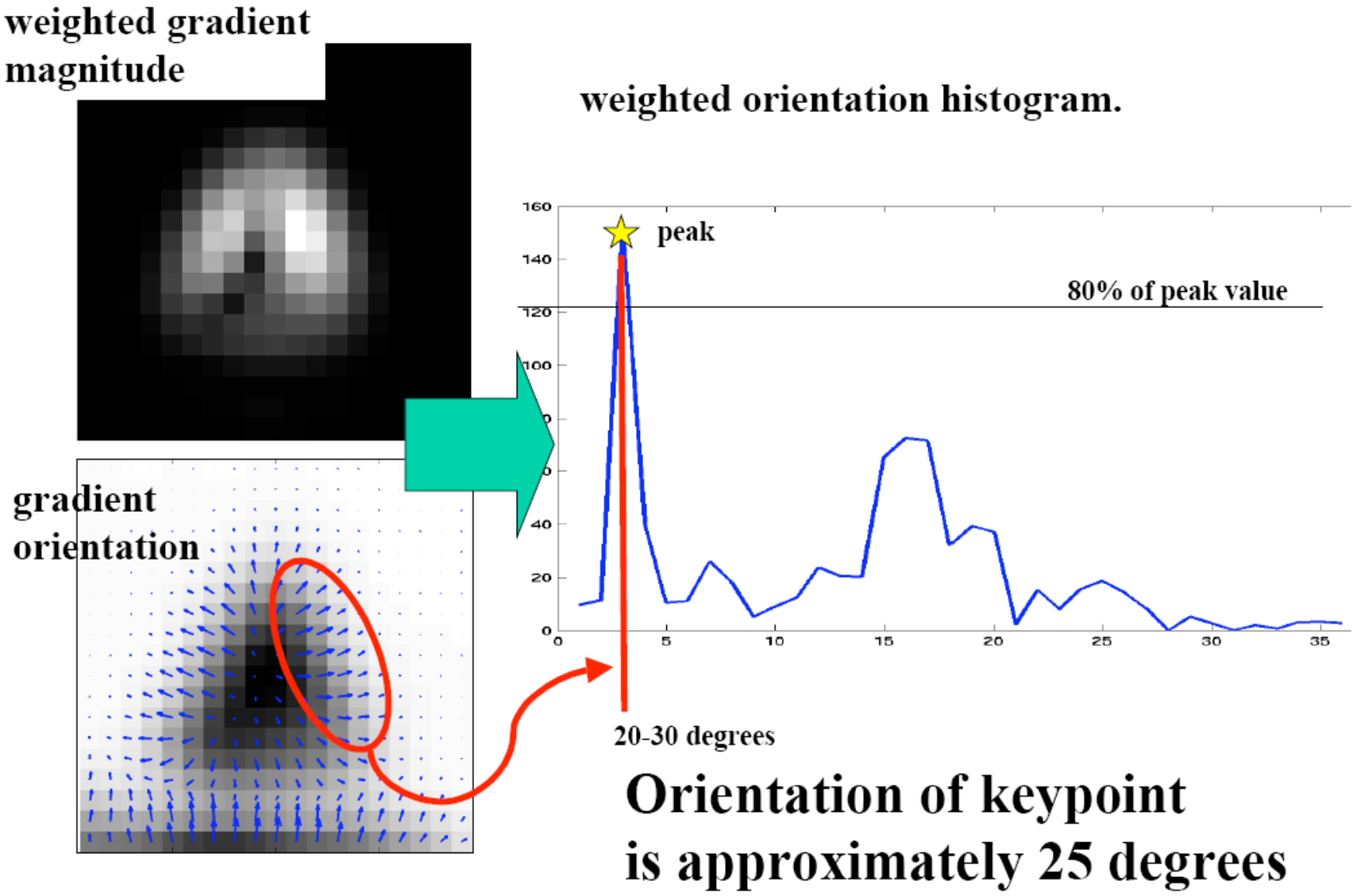


weighted orientation histogram.
Each bucket contains sum of weighted gradient magnitudes corresponding to angles that fall within that bucket.



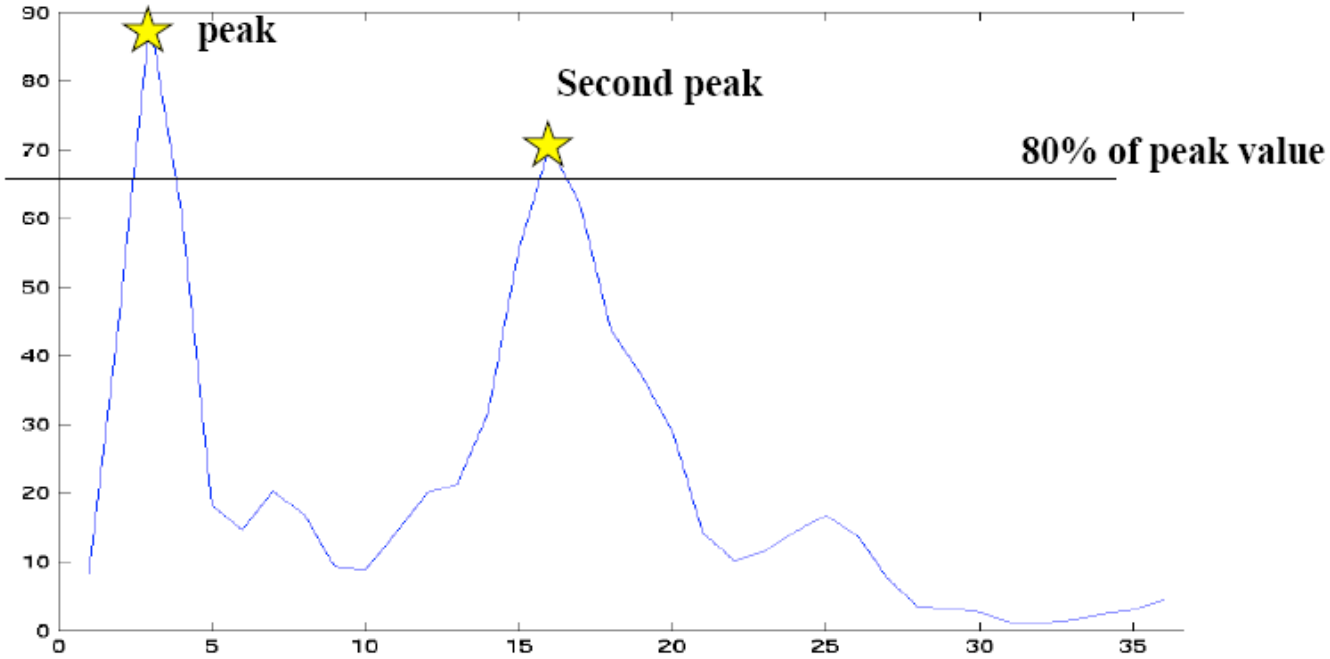
36 buckets
10 degree range of angles in each bucket, i.e.
0 <=ang<10 : bucket 1
10<=ang<20 : bucket 2
20<=ang<30 : bucket 3 …

Orientation assignment



Orientation assignment

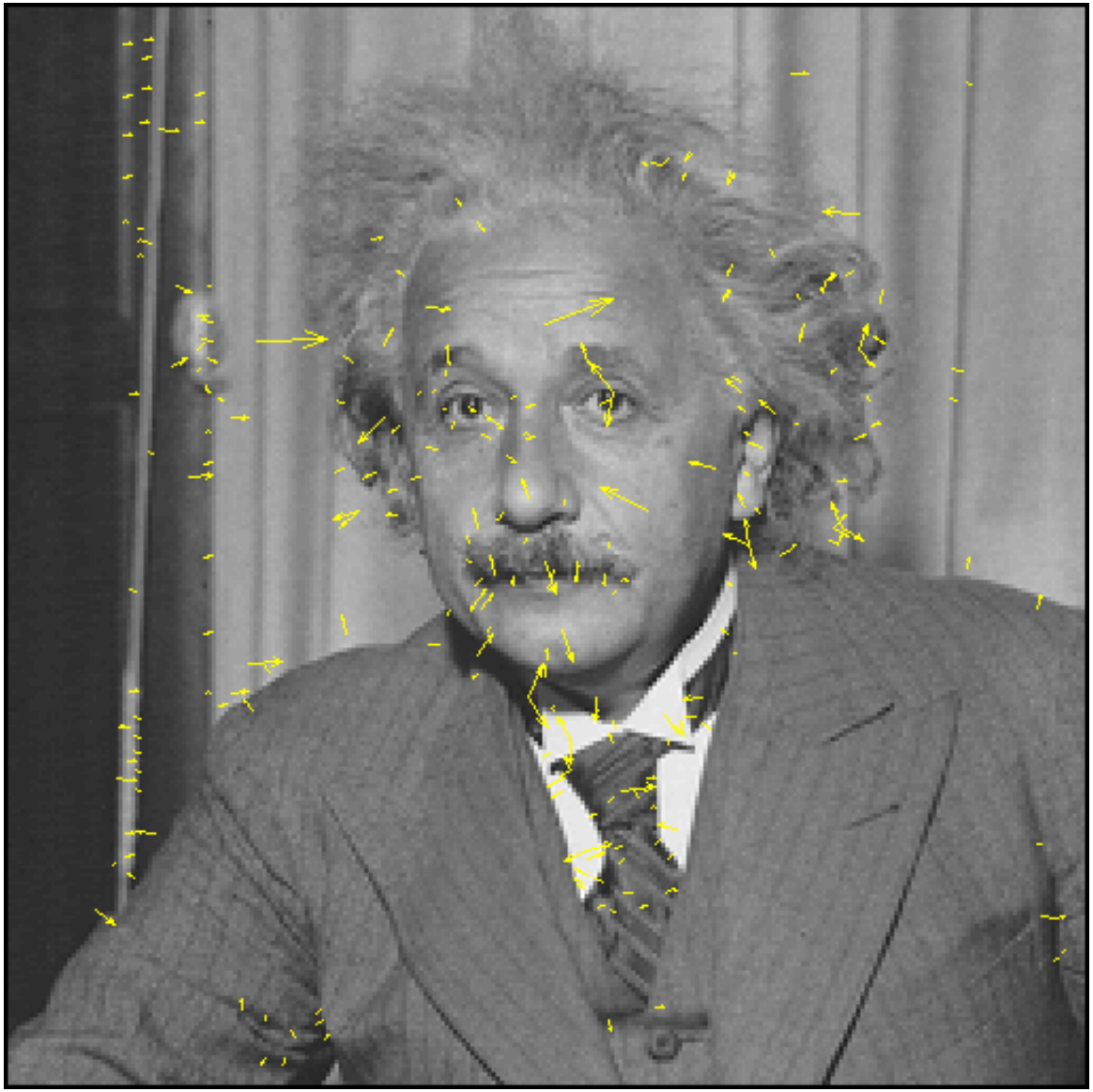
There may be multiple orientations.



In this case, generate duplicate keypoints, one with orientation at 25 degrees, one at 155 degrees.

Design decision: you may want to limit number of possible multiple peaks to two.

SIFT descriptor



4. Local image descriptor

- 16x16 Gradient window is taken (w.r.t. key orientation). Partitioned into 4x4 subwindows.
- Gaussian weighting around center(σ is 0.5 times that of the scale of a keypoint)
- Histogram of 4x4 samples in 8 directions
- 4x4x8 = 128 dimensional feature vector

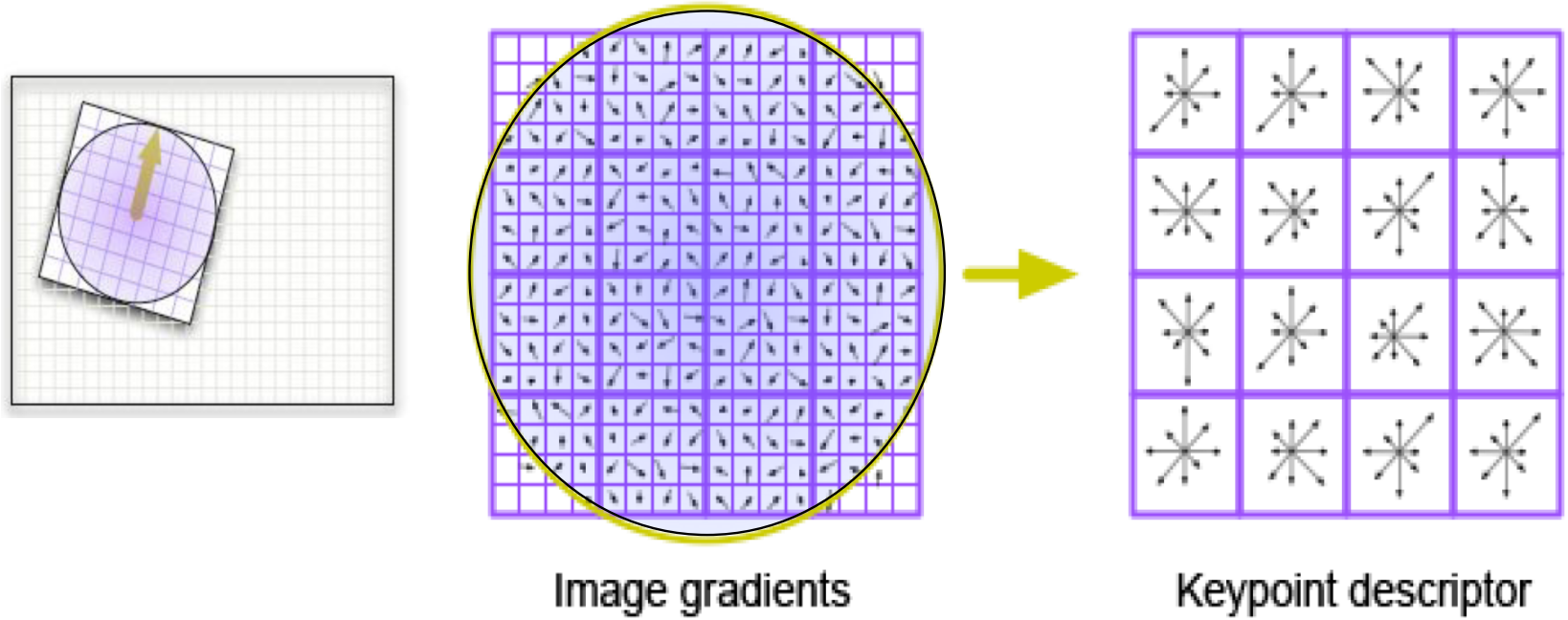
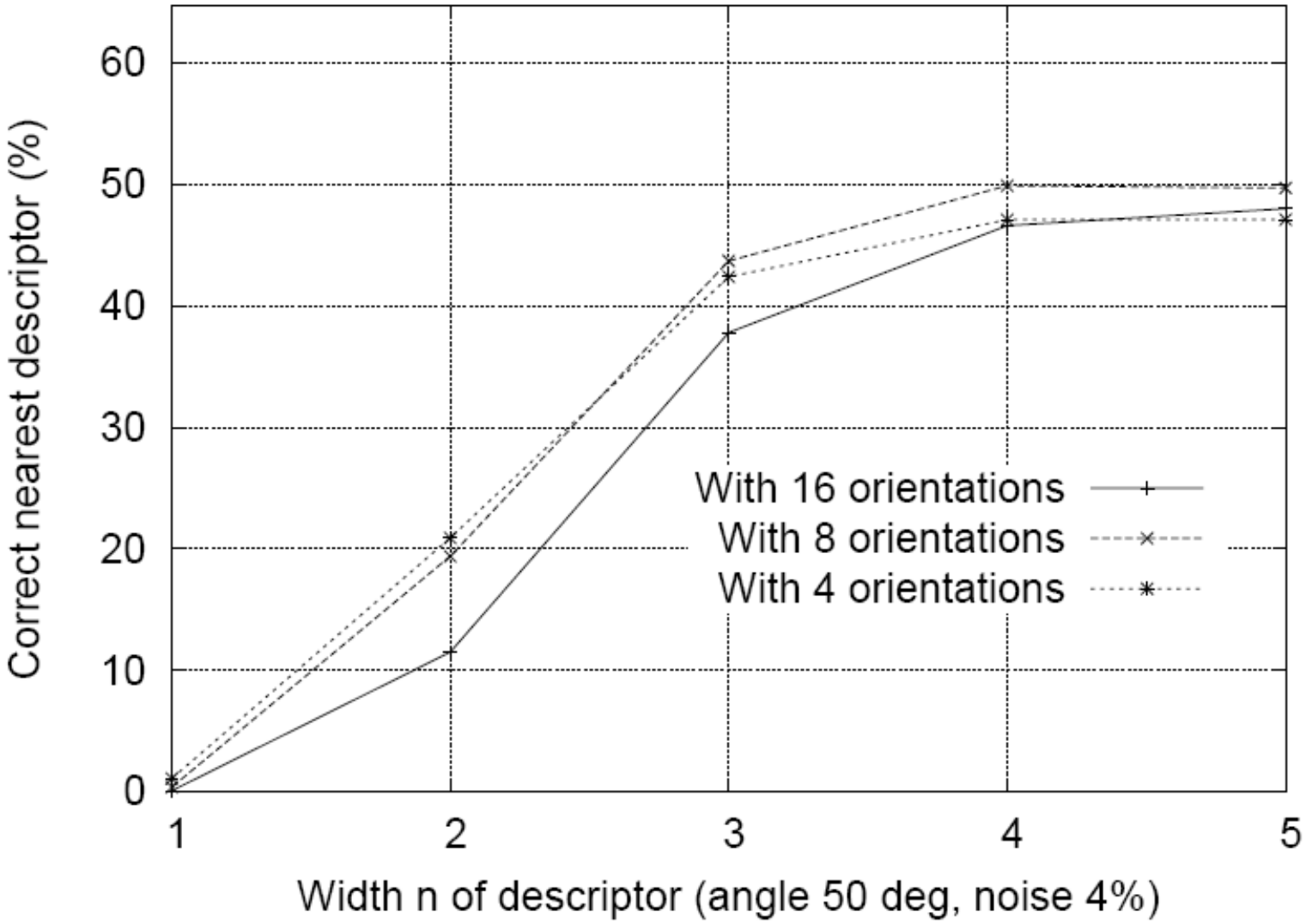
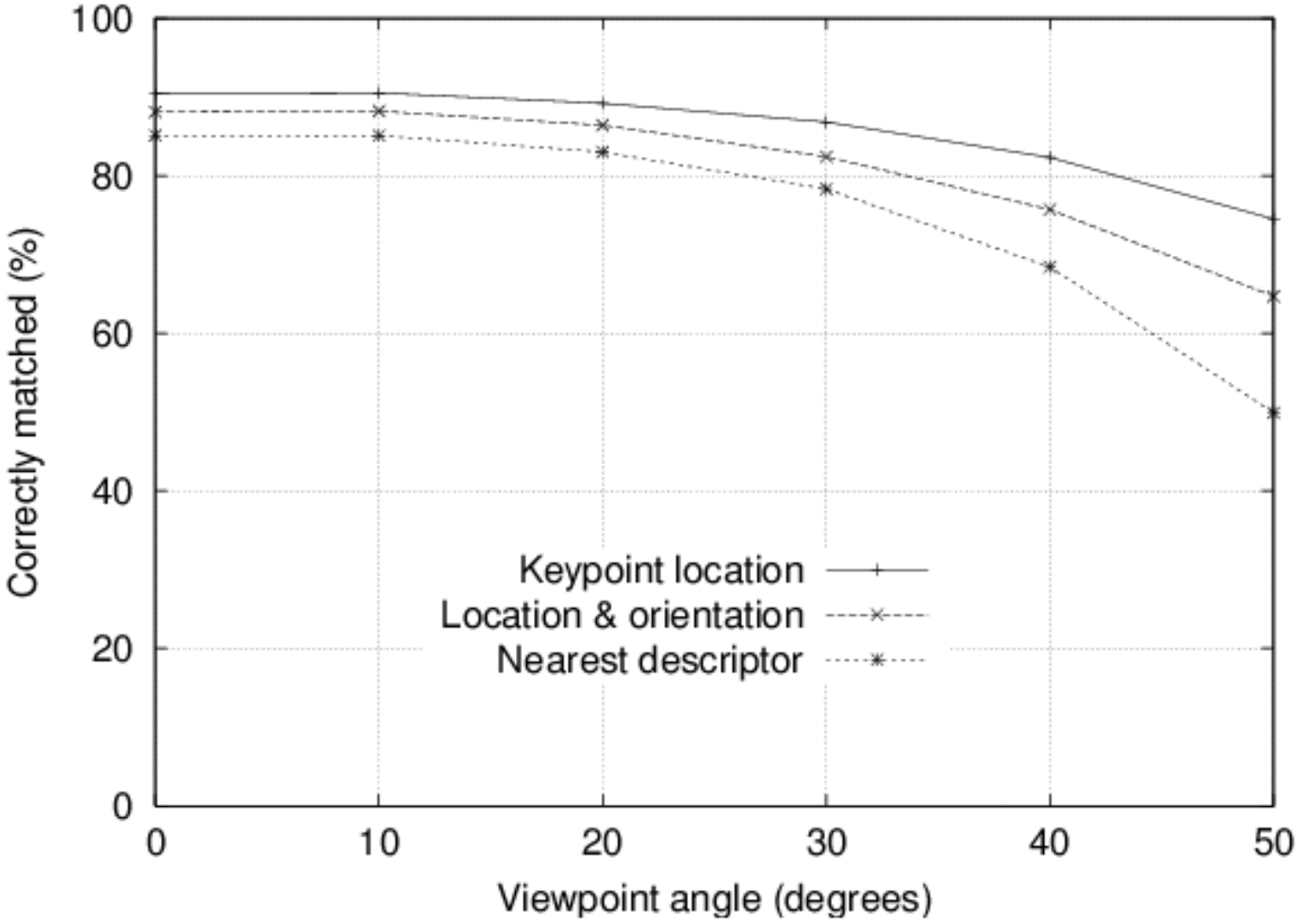


Image from: Jonas Hurrelmann

Why 4x4x8?


Correct nearest descriptor (%)
60
50
40
30
20
10
0
1
2
3
4
5
Width n of descriptor (angle 50 deg, noise 4%)
With 16 orientations
With 8 orientations
With 4 orientations

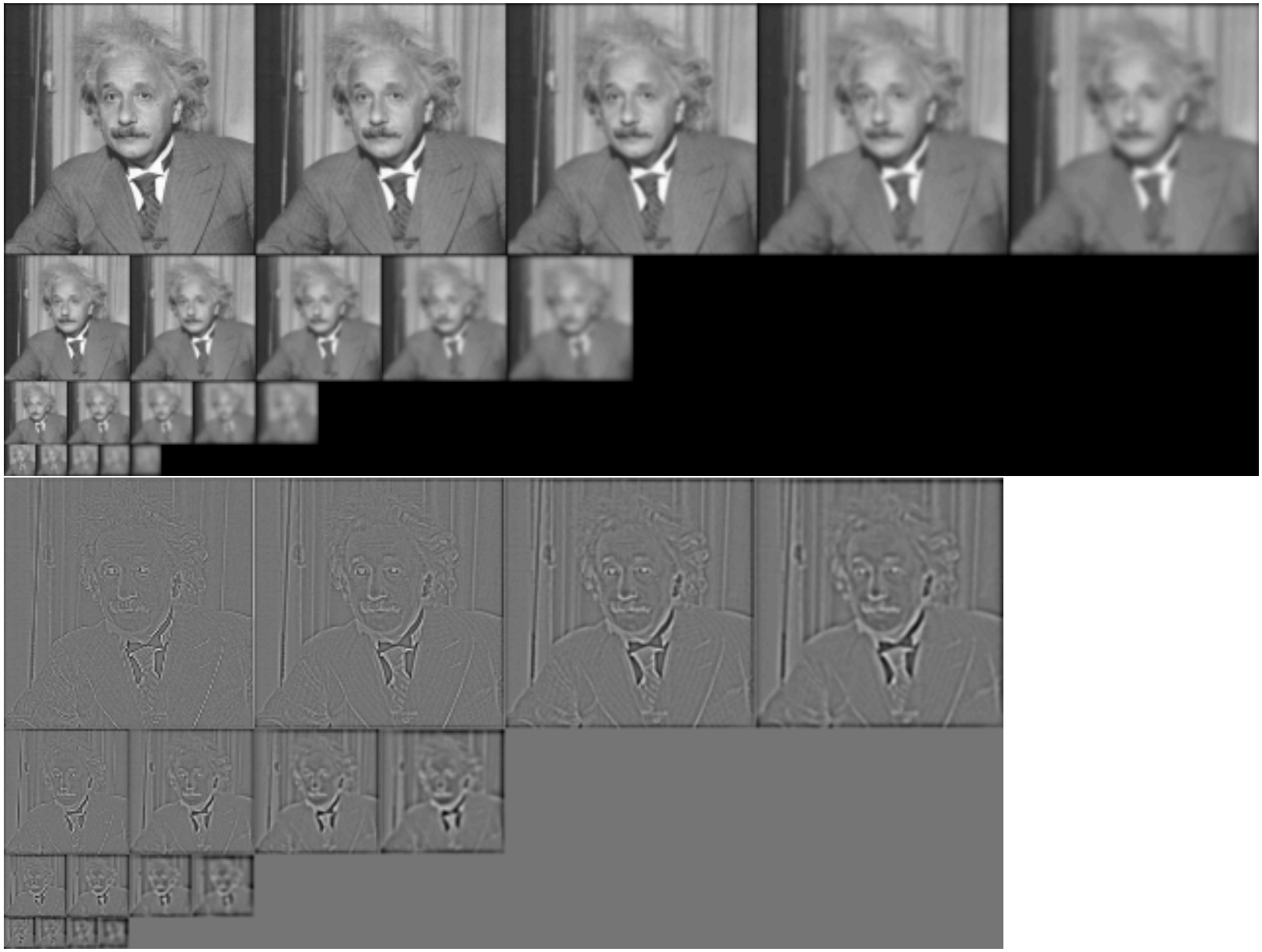
Sensitivity to affine change



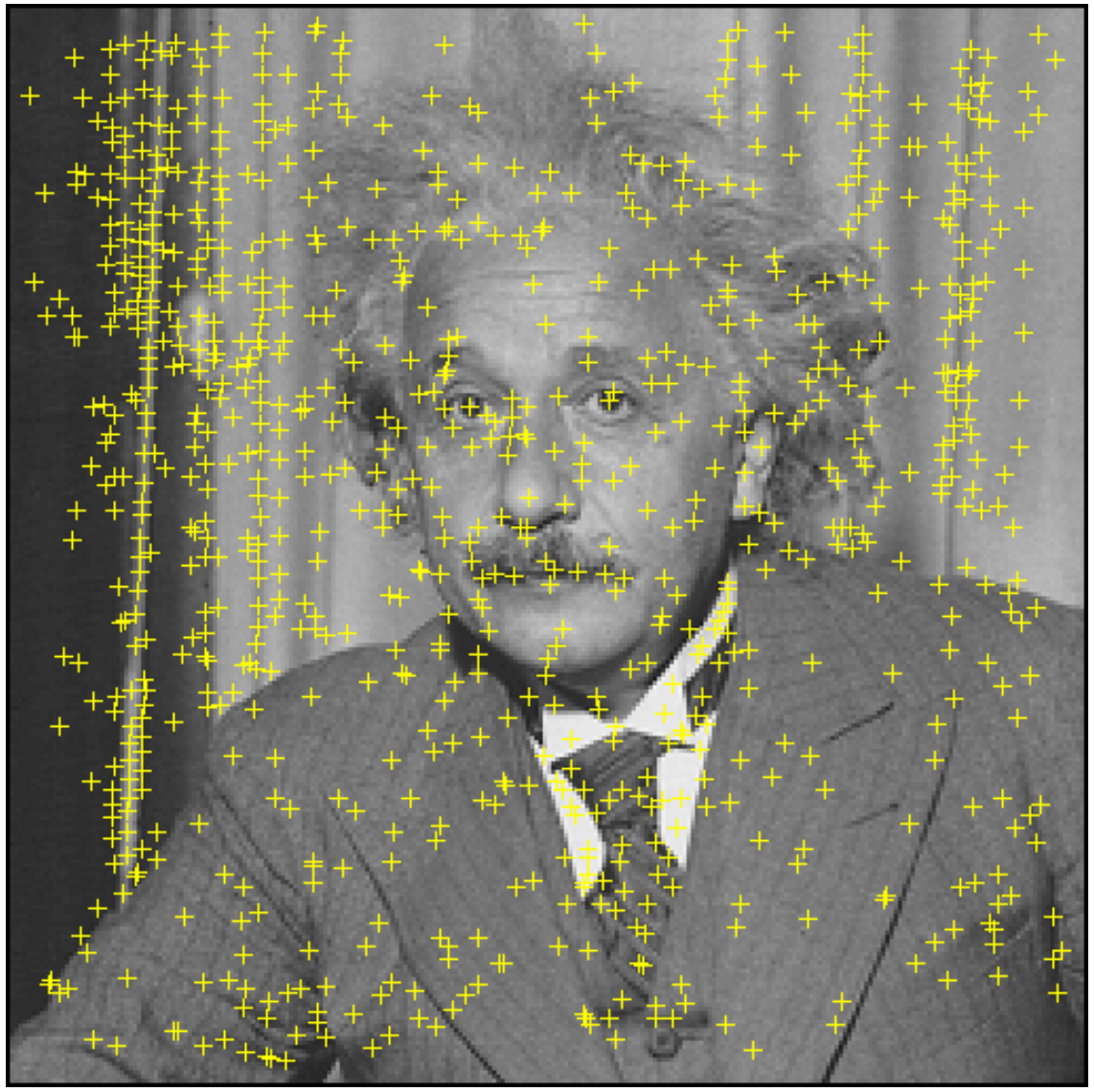
Feature matching

- for a feature x, he found the closest feature x_1 and the second closest feature x_2. If the distance ratio of $d(x, x_1)$ and $d(x, x_2)$ is smaller than 0.8, then it is accepted as a match.

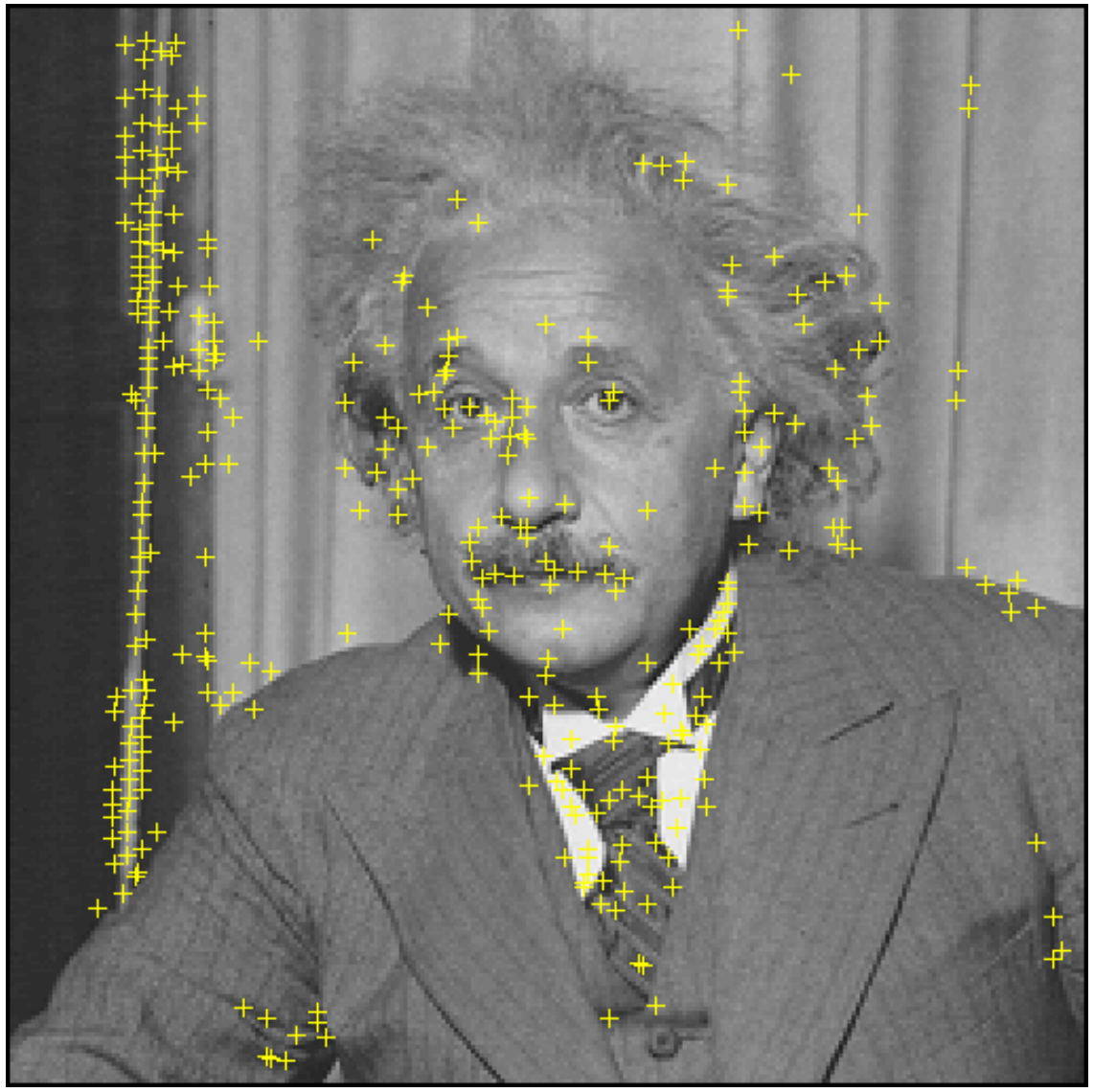
SIFT flow



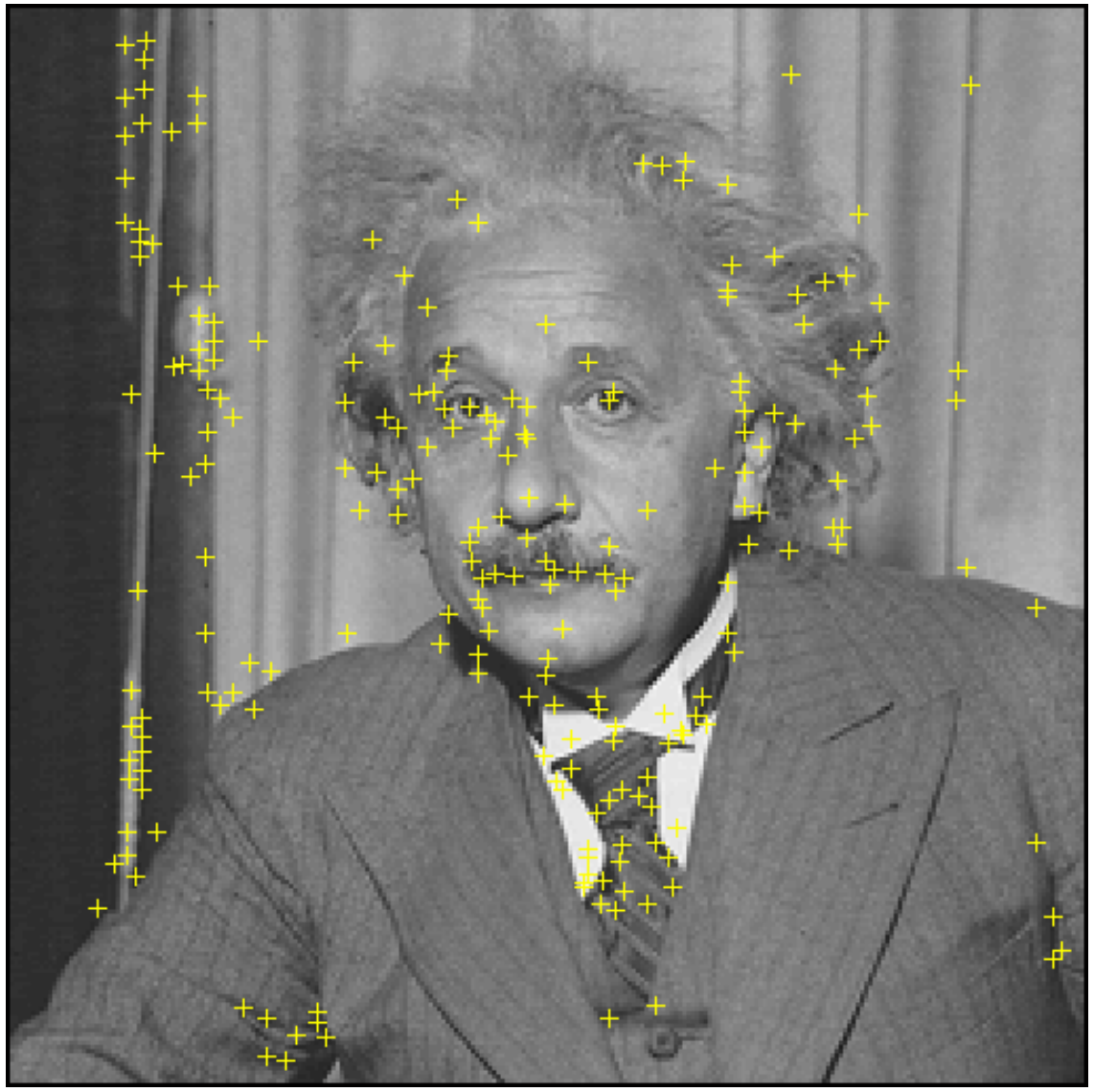
Maxima in D



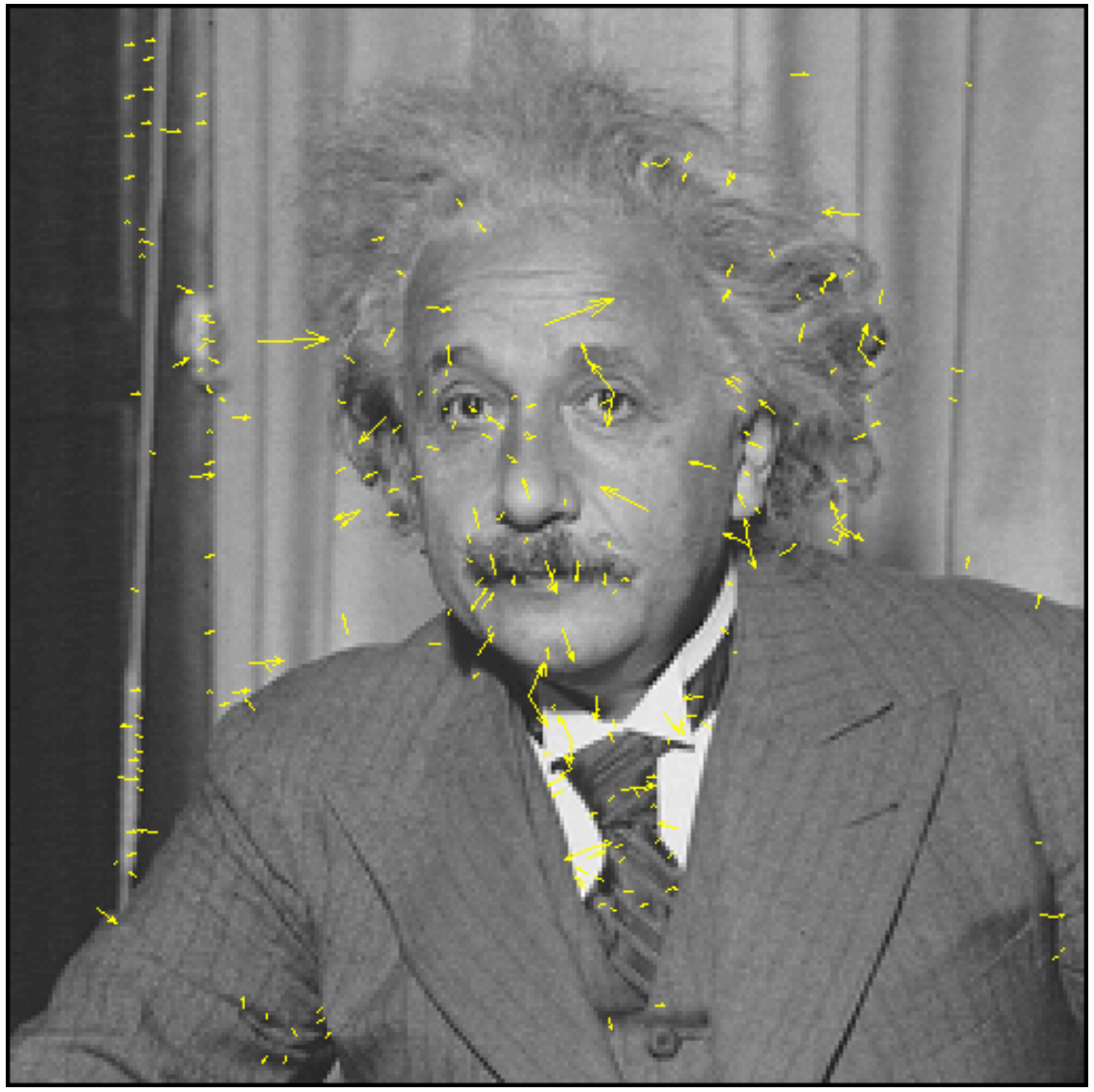
Remove low contrast



Remove edges



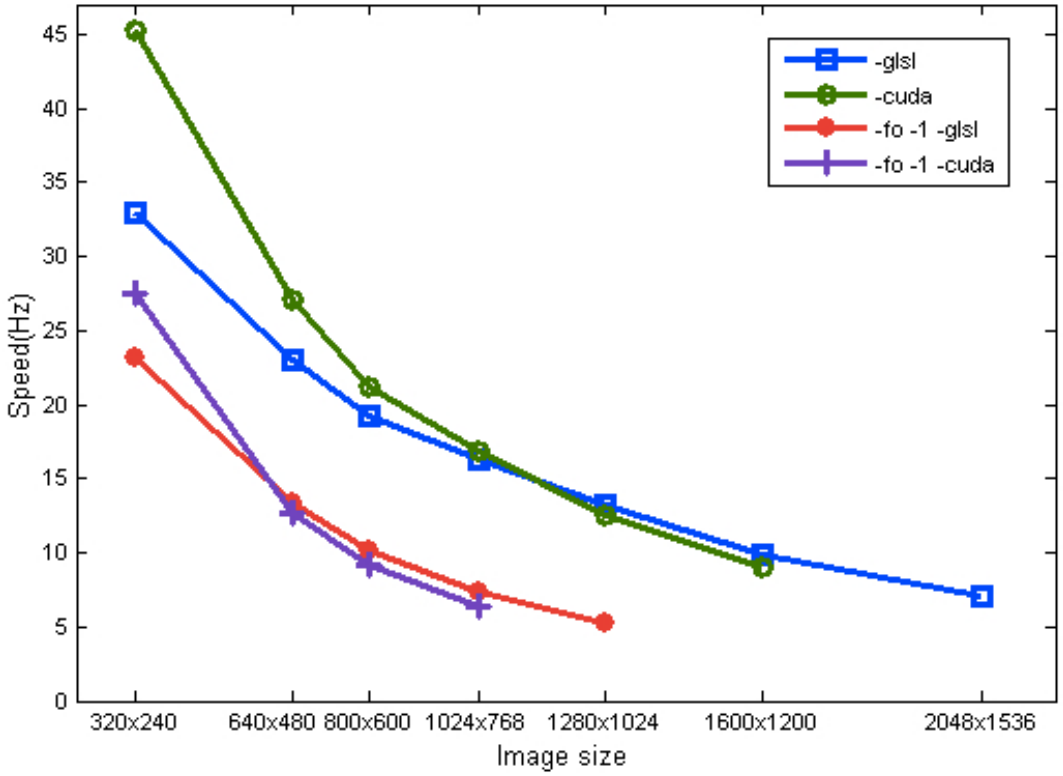
SIFT descriptor



Real-time SIFT by GPU

http://www.cs.unc.edu/~ccwu/siftgpu/

SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT)
Changchang Wu



parameter: -m -s -fo 0		
image size,	glsl,	cuda
320x240,	33.0	45.2
640x480,	23.0	27.1
800x600,	19.2	21.2
1024x768,	16.3	16.8
1280x1024,	13.2	12.5
1600x1200,	9.9	9.0
2048x1536,	7.0	X

parameter: -m -s -fo -1		
image size,	glsl,	cuda
320x240,	23.2	27.5
640x480,	13.4	12.6
800x600,	10.2	9.1
1024x768,	7.3	6.4
1280x1024,	5.2	X

Other features

- The following features are also well-known and can be found in OpenCV
  - **SURF** (Speeded-Up Robust Features)
  - **FAST** Algorithm for Corner Detection
  - **BRIEF** (Binary Robust Independent Elementary Features)
  - **ORB** (Oriented FAST and Rotated BRIEF)

SURF (Speeded Up Robust Features)

H. Bay, T. Tuytelaars, and L. Gool, “SURF: Speeded Up Robust Features,” *ECCV*, 2006.
(cited number: 20723 from Google)

SURF
(Speeded Up Robust Features)

- Why SURF?
  - SIFT is really good, but not fast enough, so people came up with a speeded-up version called SURF.

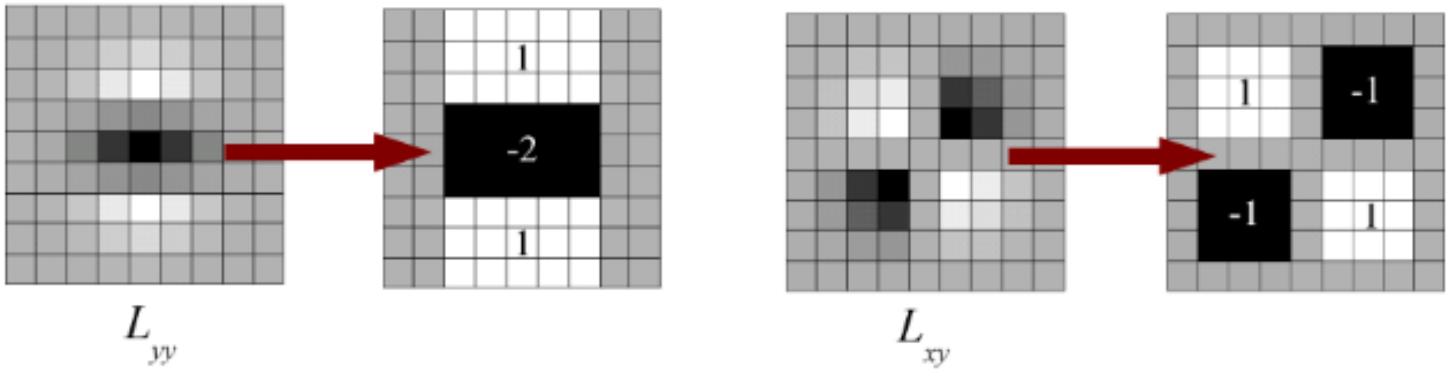
- Properties of SURF
  - idea from SIFT, but much more simplified
  - faster computation and faster matching

Feature Detection: Fast-Hessian

- For faster computation
  - Lowe uses difference of Gaussian to approximate Laplacian of Gaussian. (SIFT)
  - Here, SURF uses Hessian - Laplacian to approximate Laplacian of Gaussian, to improve calculation speed.

Feature Detection: Fast-Hessian

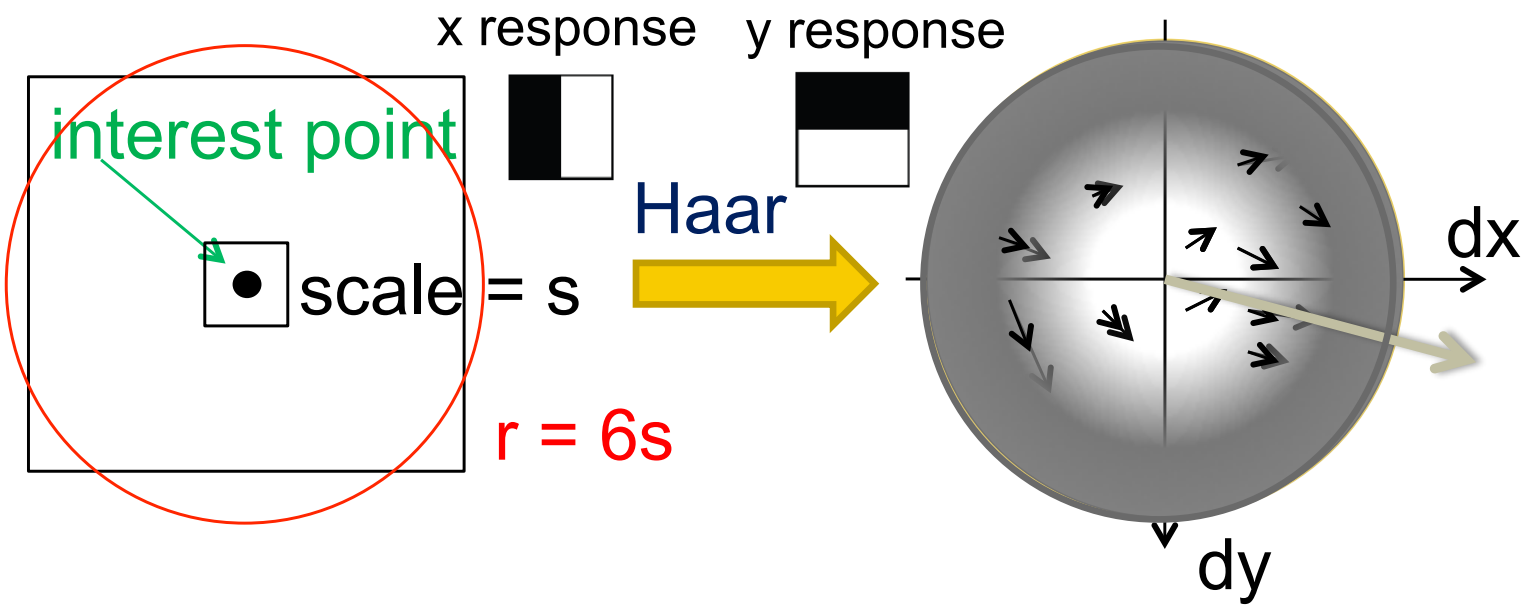
- based on Hessian matrix (Laplacian of Gaussian, LoG)
- SURF uses “Fast-Hessian Detector”: approximation with box filters



box filters → very fast for arbitrary size and can be easily calculated with the help of integral images.

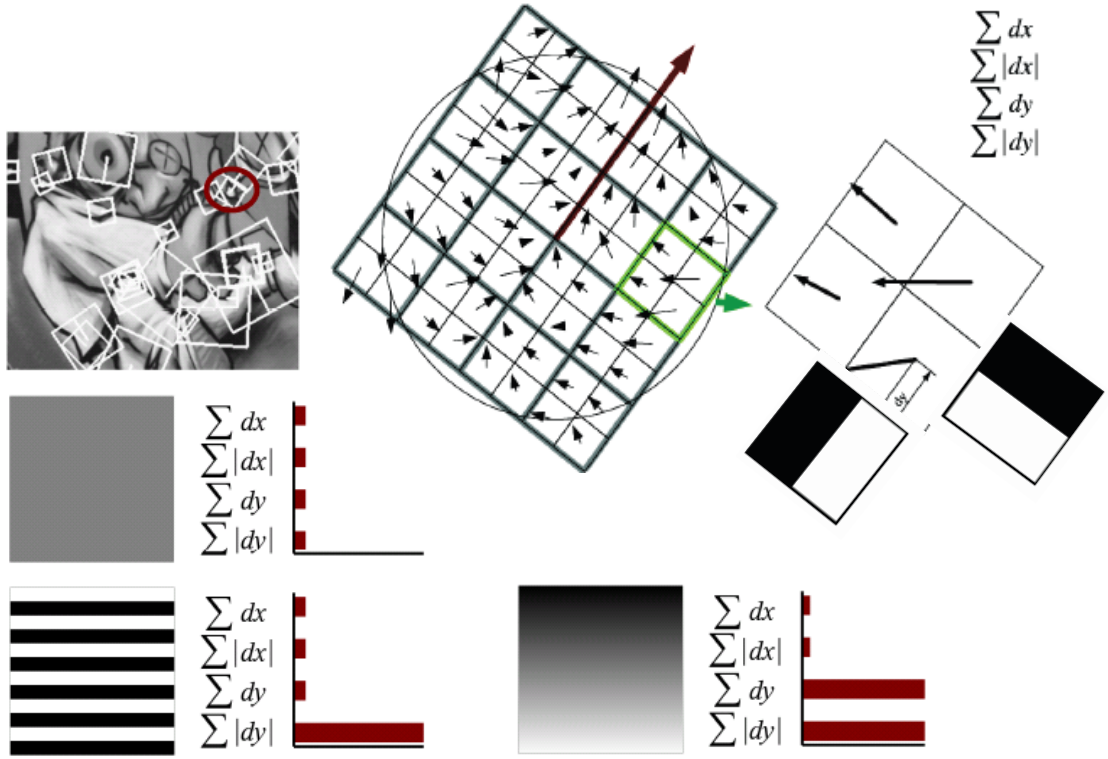
Orientation Assignment

- The Haar wavelet responses are represented as vectors
- Sum all responses within a sliding orientation window covering an angle of 60 degree
- The longest vector is the dominant orientation



Descriptor

- Split the interest region (20s x 20s) up into 4 x 4 square sub-regions.
- Calculate Haar wavelet response d_x and d_y and weight the response with a Gaussian kernel.
- Sum the response over each sub-region for d_x and d_y, then sum the absolute value of response.



Descriptor

- Split the interest region (20s x 20s) up into 4 x 4 square sub-regions.
- Calculate Haar wavelet response d_x and d_y and weight the response with a Gaussian kernel.
- Sum the response over

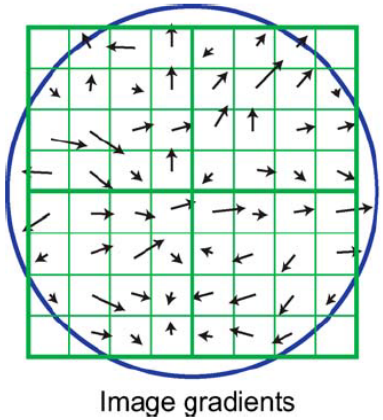
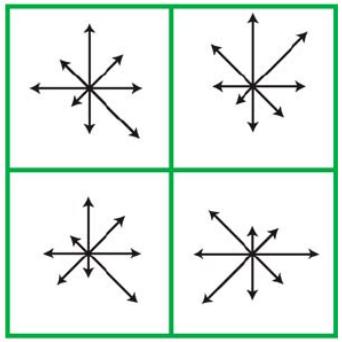
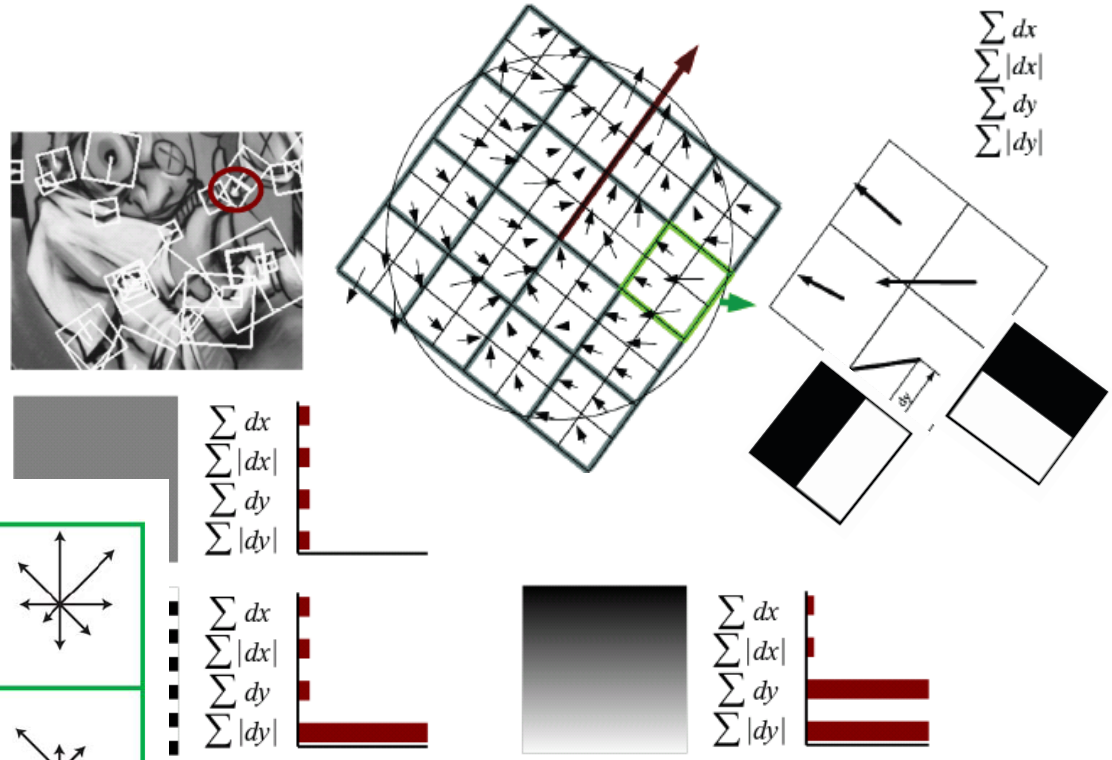


Image gradients


Keypoint descriptor



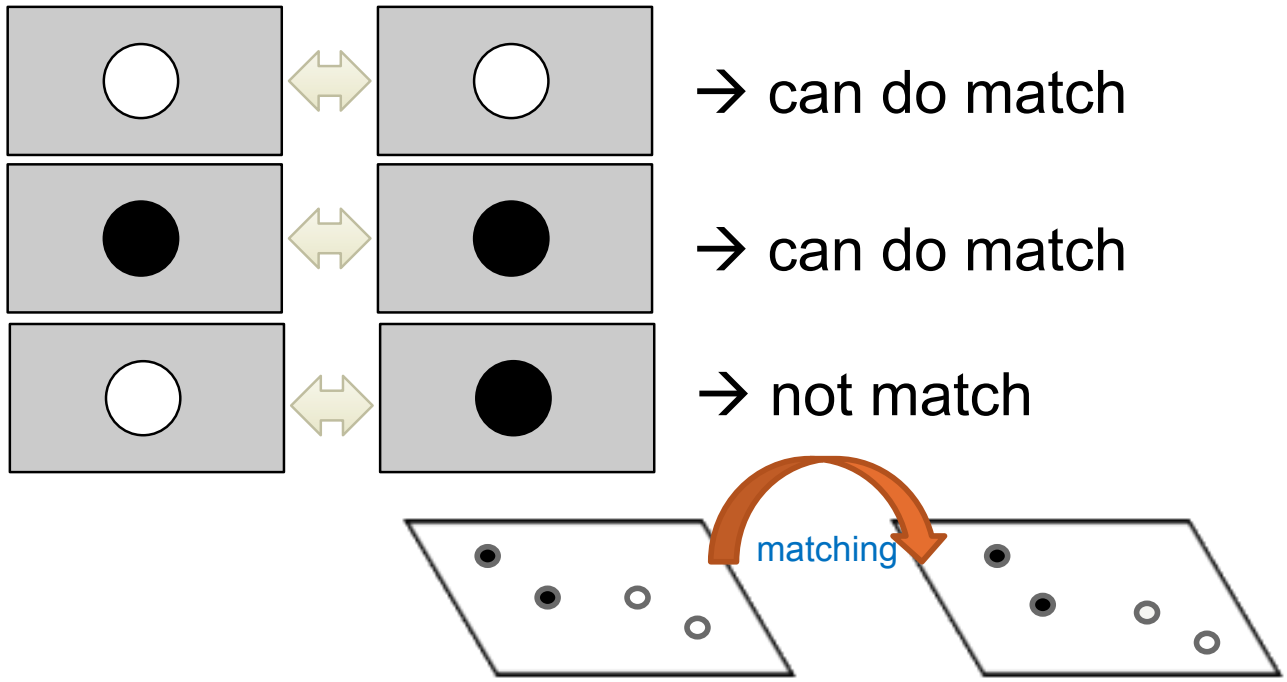
4x4x4 = 64 dimensions

Matching

- Fast indexing through the sign of the Laplacian for the underlying interest point

The sign of trace of the Hessian matrix

Trace = $L_{xx} + L_{yy}$



SIFT vs. SURF

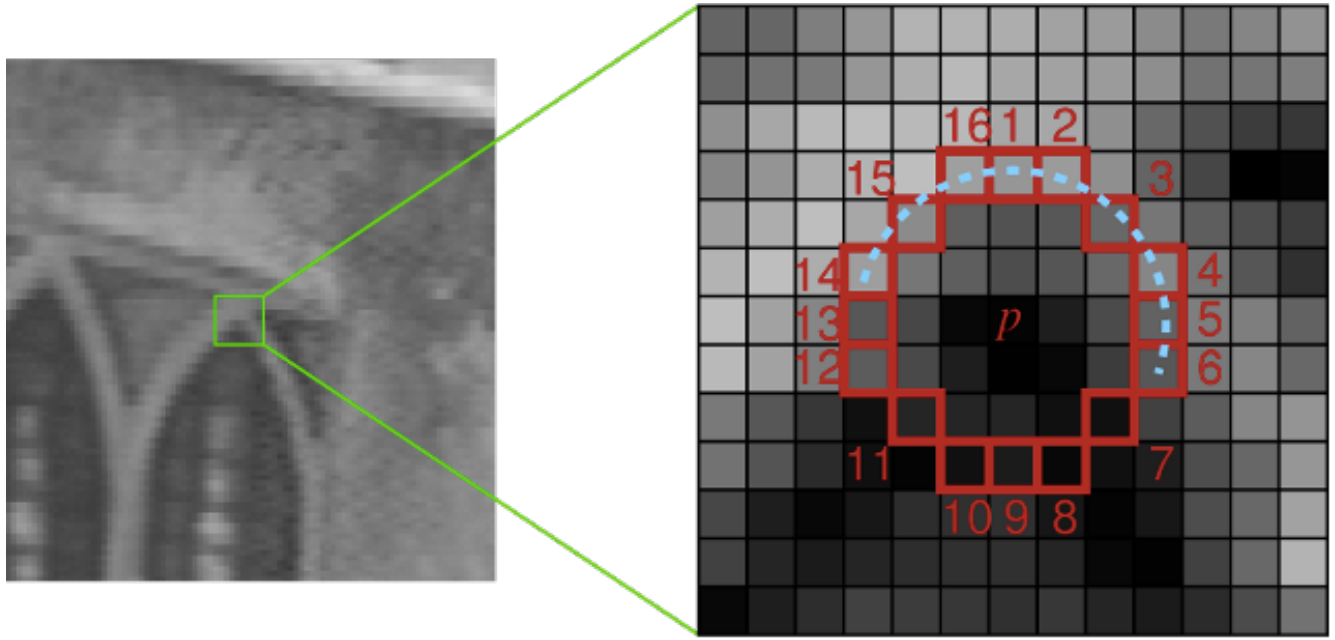
- SIFT can detect more number of features compared to SURF
- SURF is faster than SIFT by 3 times, and has recall precision not worse than SIFT.
- SURF is good at handling image with blurring or rotation.
- SURF is poor at handling image with viewpoint change.

Still not fast enough, so we need other features

FAST
(Features from Accelerated Segment Test)

E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” *ECCV*, 2006.
(cited number: 5858 from Google)

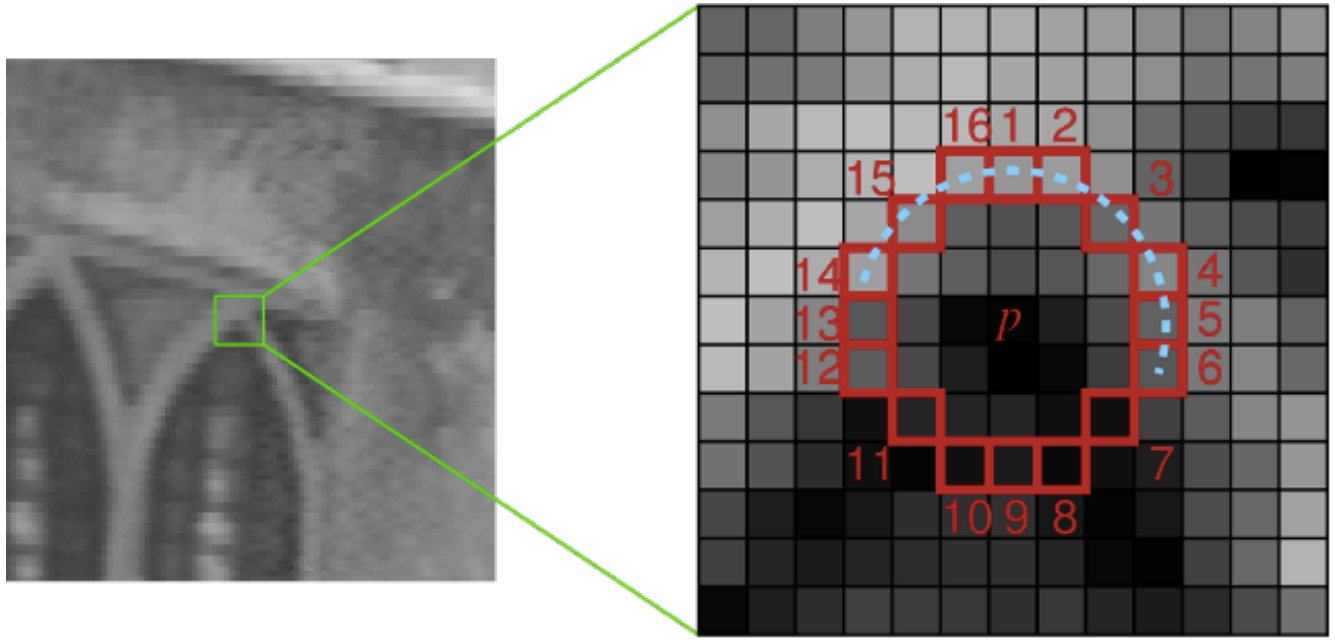
Fast Corner Detector



- What is corner?
  - Contiguous arc of *n* or more pixels:
    - All much brighter than *p* (brighter than *p* + *t*)

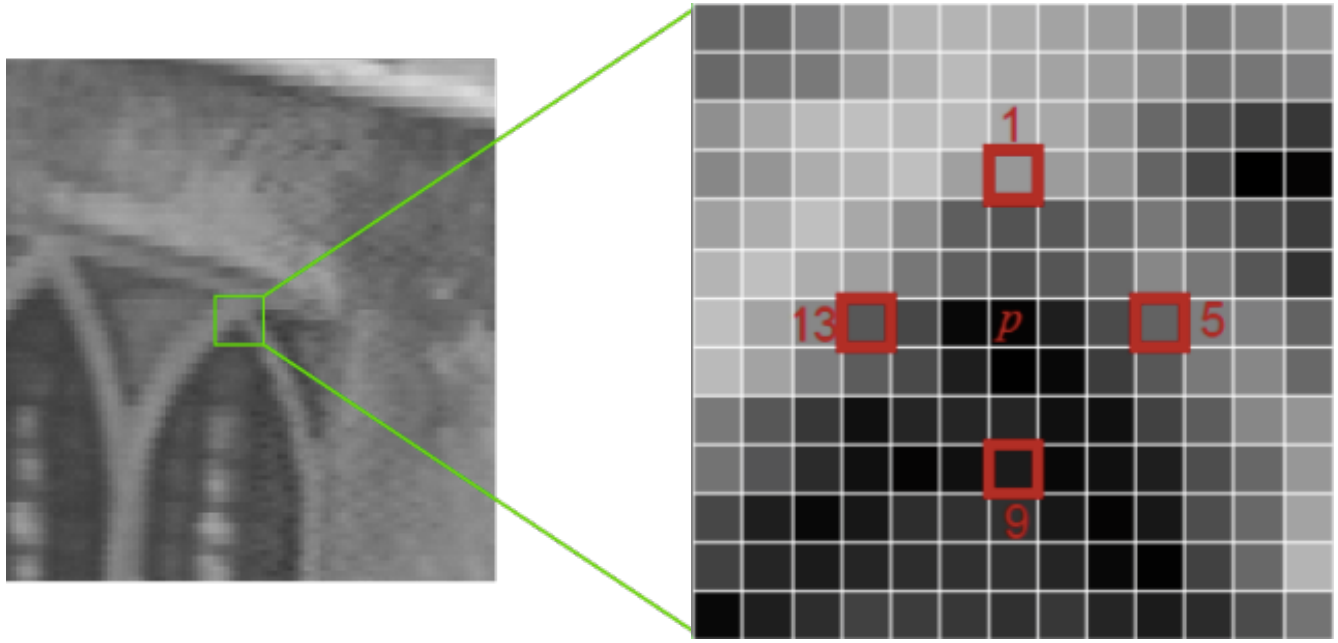
  or
    - All much darker than *p* (brighter than *p* - *t*)

Fast Corner Detector



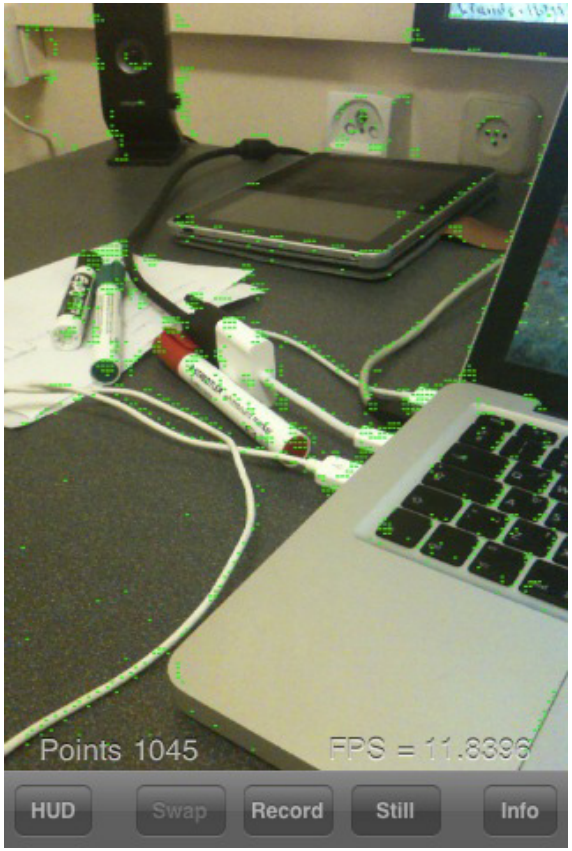
- For each pixel, consider a circle of 16 pixels around the pixel under test.
  - *p* is a corner if there exists a set of *n* contiguous pixels in the circle
  - *n* = 12

Fast Corner Detector



- For each pixel, consider a circle of 16 pixels around the pixel under test.
  - *p* is a corner if there exists a set of *n* contiguous pixels in the circle
  - *n* = 12
- **rapid rejection** by testing 1, 9, 5, 13
  (First 1 and 9 are tested if they are too brighter or darker. If so, then checks 5 and 13)

Results and Summary



- Pros
  - very fast (several times faster than other existing corner detectors)
  - high quality feature detection
- Cons
  - not robust to high levels noise
  - dependent on a threshold

BRIEF (Binary Robust Independent Elementary Features)

M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: Binary Robust Independent Elementary Features,” *ECCV*, 2010.
(cited number: 4516 from Google)

BRIEF feature descriptor

- SIFT uses 128-dim vector for descriptors.
  - floating point
  - 512 Bytes

- SURF uses 64-dim vector for descriptors.
  - floating point
  - 256 Bytes

They take large memory and longer time for matching

BRIEF feature descriptor

- BRIEF is a **feature descriptor**, it doesn't provide any method to find the features.
- BRIEF is a **faster** method feature descriptor calculation and matching.
- BRIEF also provides **high recognition rate** unless there is large in-plane rotation.

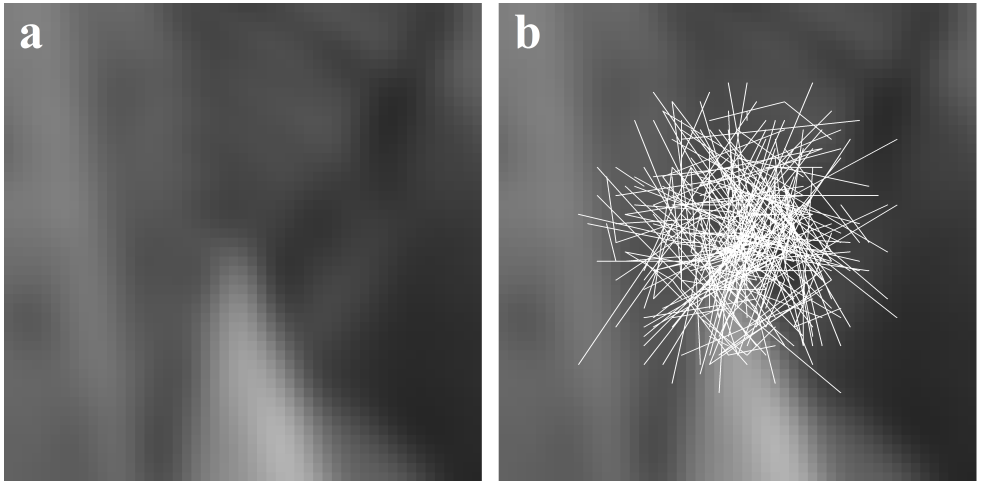
BRIEF feature descriptor

1. smooth the input image with a 9×9 Gaussian filter

2. choose 256 pairs of points within a 35×35 pixel area, following a Gaussian distribution with σ = 7 pixels.

3. for every pair of points, if intensity at the first point is greater than at the second one, add a 0 to the bit string, otherwise add a 1.

4. The resulting bit string of length 256 is the descriptor for point c



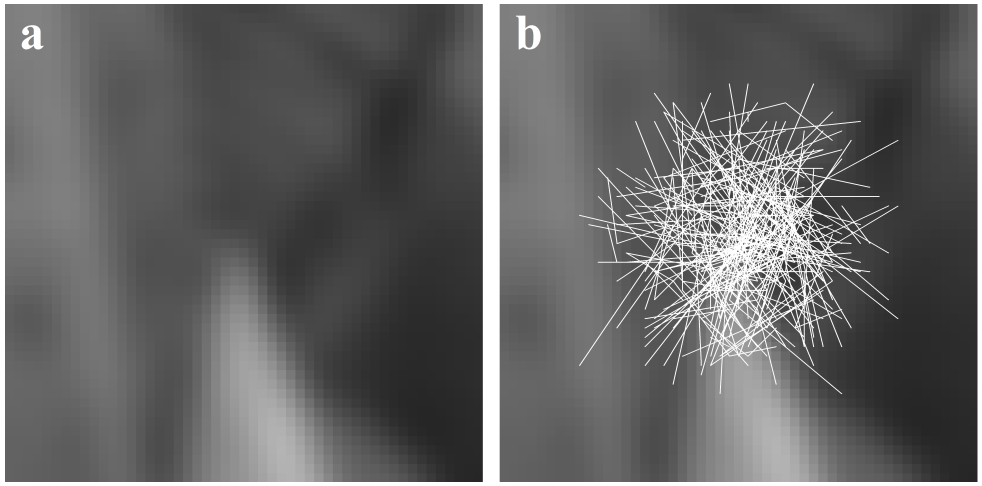
256 binary ==> 32 Byte

can use Hamming Distance to match

Summary

- Pros:
  - Compact, easy-computed, highly discriminative
  - Fast matching using Hamming distance
  - Good recognition performance

- Cons:
  - More sensitive to image distortions and transformations, in particular to in-plane rotation and scale change



ORB
(Oriented FAST and Rotated BRIEF)

E. Rublee, V. Rabaud, K. Konolige and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” *ICCV*, 2011.
(cited number: 9764 from Google)

Detector: Oriented FAST

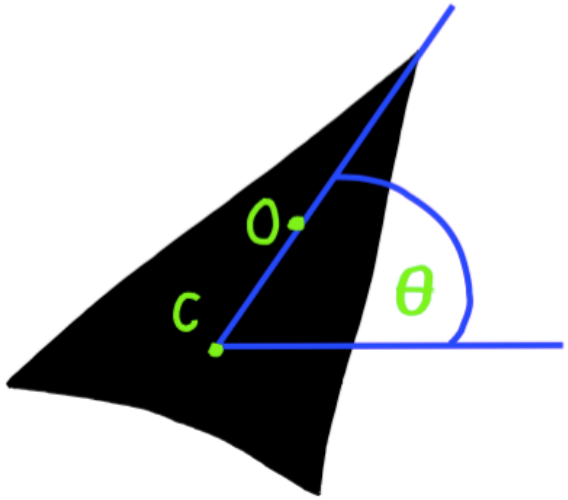
- shortcomings of FAST approach:

1. no quality measure (“cornerness”)
   solution: use Harris cornerness measure - apply Harris corner measure to find top *N* points among them.

2. not scale invariant
   solution: use scale pyramid (like SIFT)

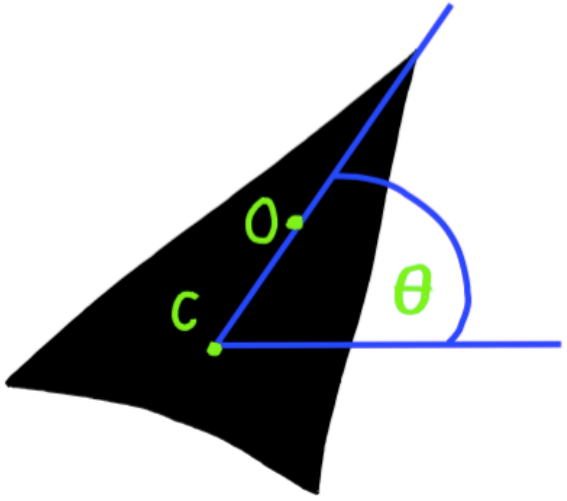
3. not rotation invariant
   solution: calculate intensity centroid

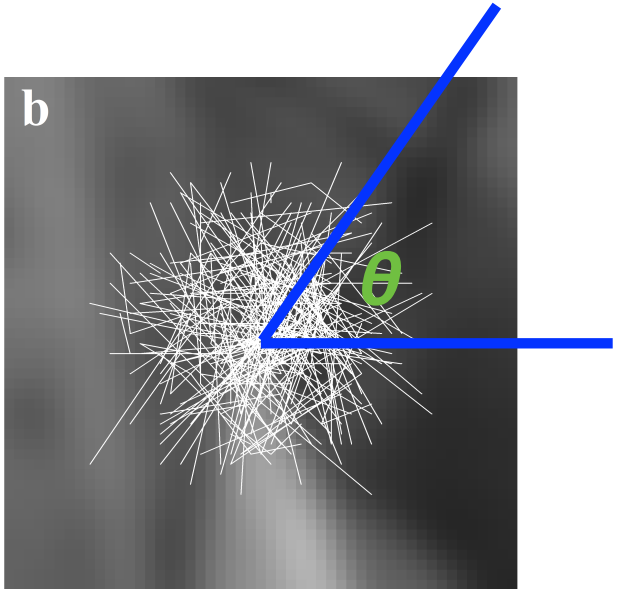
   The direction of the vector from this corner point to centroid gives the orientation.



Descriptor: Rotated BRIEF

- “steer” BRIEF according to the orientation of keypoints.
  - from oriented FAST, we have detected the direction of the feature
  - rotate the patch to get steered (rotated) version of BRIEF





Summary

- The paper says ORB is much faster than SURF and SIFT and ORB descriptor works better than SURF. ORB is a good choice in low-power devices for panorama stitching etc.

Comparison

Table 1. Averaged computation times for the different detectors

Detector	Run time [ms.]	Speed-up [-]	# keypoints
SURF	176	1.9	2 911
DoG	338	1.0	1 552
FAST	**2**	169.0	**5 158**
STAR	17	19.9	849
MSER	60	5.6	483
BRISK	**10**	33.8	1 874
ORB	**7**	48.3	594

Miksik, O., & Mikolajczyk, K. (2012, November). Evaluation of local detectors and descriptors for fast feature matching. In Pattern Recognition (ICPR), 2012 21st International Conference on (pp. 2681-2684). IEEE.

Comparison

Table 2. Computation times for the different descriptors for 1000 SURF keypoints

Descriptor	Run time [ms.]	Speed-up [-]
SURF	117.1	3.83
SIFT	448.6	1.00
BRIEF	**3.8**	118.05
BRISK	**10.6**	42.32
ORB	**4.2**	106.80
LIOP	1 801.1	0.25
MROGH	2 976.8	0.15
MRRID	5 625.1	0.08

Miksik, O., & Mikolajczyk, K. (2012, November). Evaluation of local detectors and descriptors for fast feature matching. In Pattern Recognition (ICPR), 2012 21st International Conference on (pp. 2681-2684). IEEE.

Comparison

Table 4. Precision/Recall for the different descriptors and $N = 40$, $\epsilon = 3$

Detector	Descriptor	Precision	Recall	MAP
SURF	SURF	0.485	0.513	0.334
SURF	SIFT	0.525	0.533	0.491
SURF	BRIEF	0.517	0.546	0.514
SURF	ORB	0.448	0.470	0.437
SURF	LIOP	0.581	0.597	0.568
SURF	MROGH	0.540	0.567	0.527
SURF	MRRID	0.550	0.569	0.510
SURF	BRISK	0.536	0.553	0.530
BRISK	BRISK	0.504	0.527	0.492
ORB	ORB	0.493	0.495	0.463
FAST	SIFT	0.366	0.376	0.336

Miksik, O., & Mikolajczyk, K. (2012, November). Evaluation of local detectors and descriptors for fast feature matching. In Pattern Recognition (ICPR), 2012 21st International Conference on (pp. 2681-2684). IEEE.

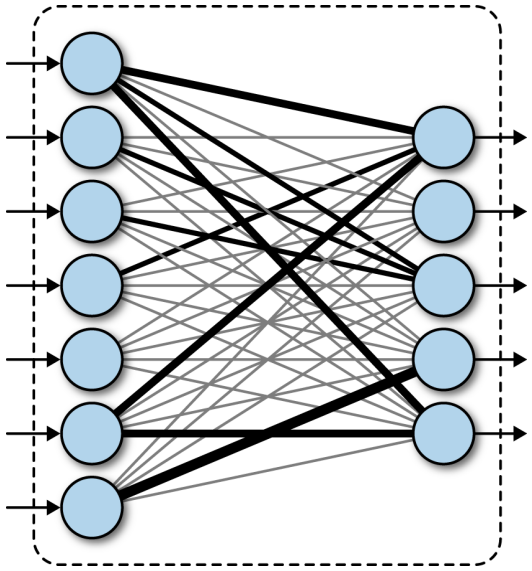
Feature Matching with Deep Learning

Learning-based feature descriptor

Learning-based features utilizing CNN to learn the descriptor

Two categories:

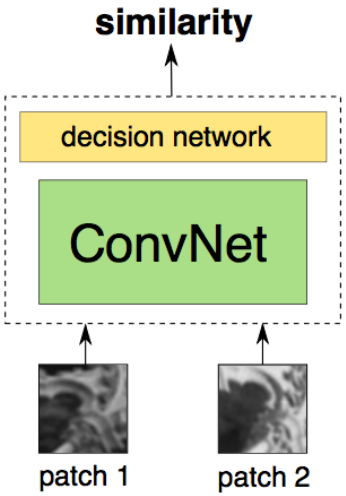
- With metric layer
- Without metric layer

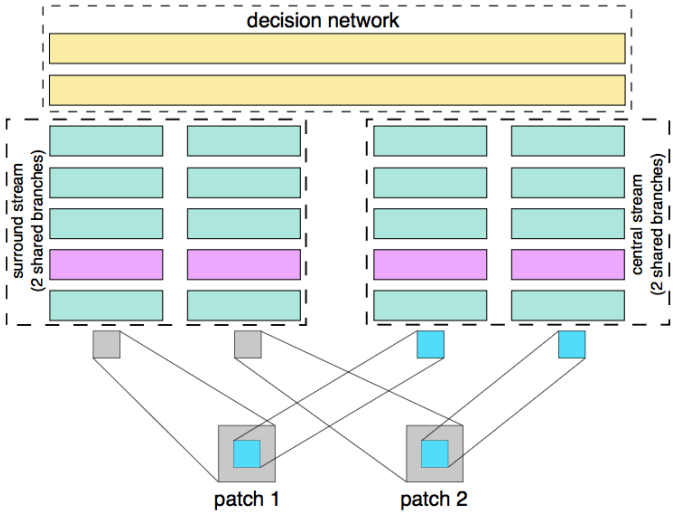


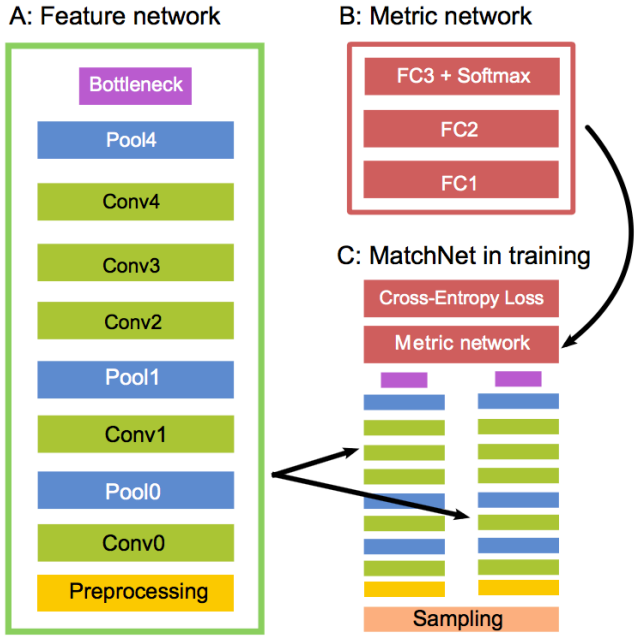
A fully connected layer is considered as the metric layer

with Metric Layer

- **“MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching”**
  - X. Han, T. Leung, Y. Jia, R. Sukthankar, A.C. Berg , (CVPR, 2015)
- **“Learning to Compare Image Patches via Convolutional Neural Networks”**
  - Sergey Zagoruyko, Nikos Komodakis, (CVPR, 2015)



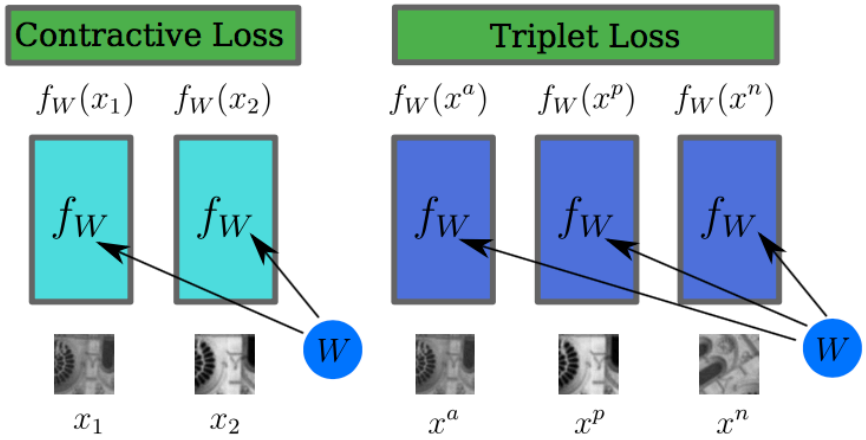


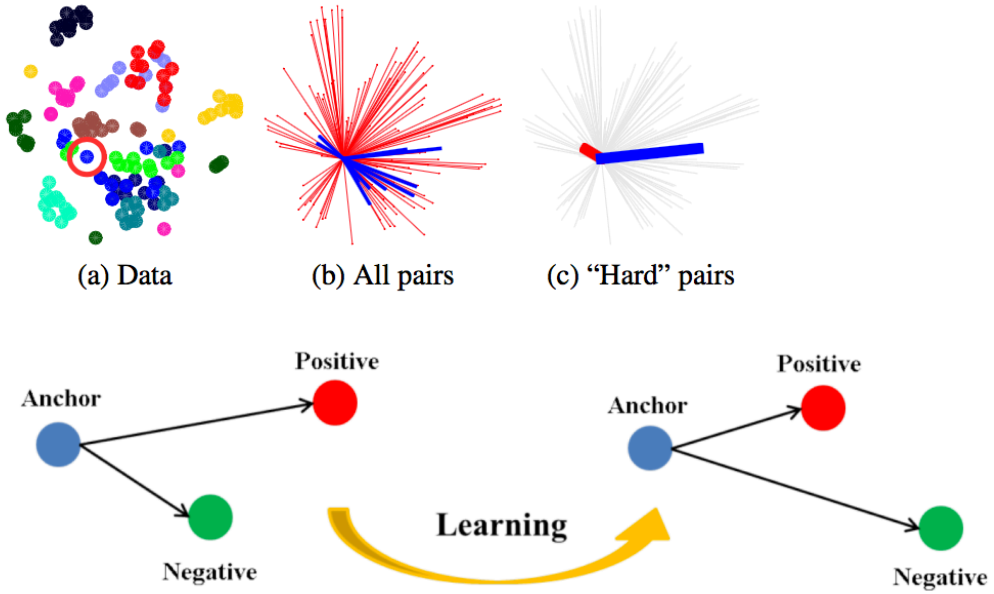


Problems of metric layer: can not perform nearest neighbor search (NNS) => binary classification case

without Metric Layer

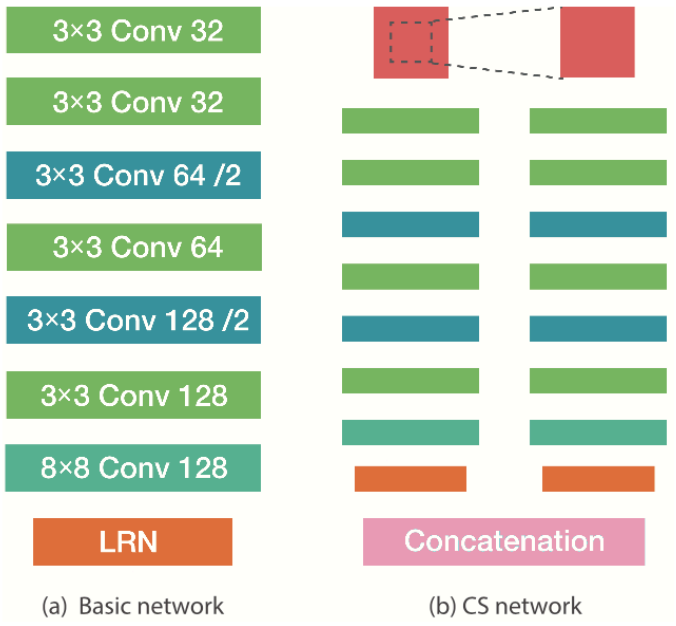
- **“Discriminative learning of deep convolutional feature point descriptors”**
  - Edgar Simo-Serra, Eduard Trulls, (ICCV, 2015)
- **“Euclidean and Hamming Embedding for Image Patch Description with Convolutional Networks”**
  - Zishun Liu, Zhenxi Li, Juyong Zhang, Ligang Liu, (CVPR, 2016)

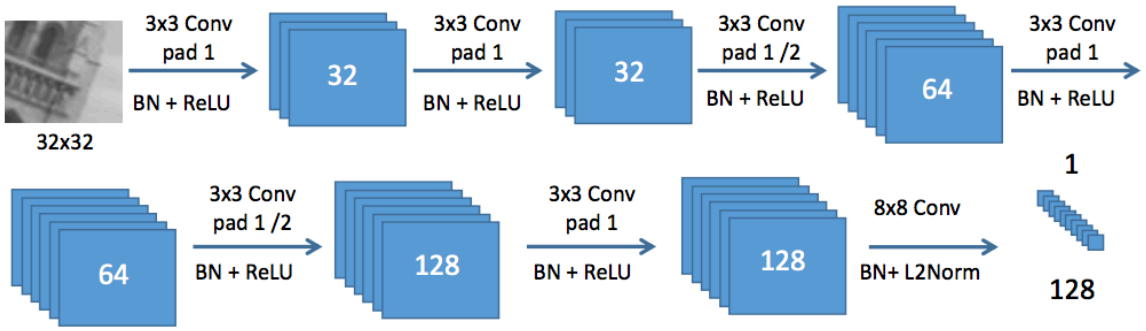




without Metric Layer

- **“L2-Net: Deep learning of discriminative patch descriptor in euclidean space”**
  - Bin Fan Yurun Tian and Fuchao Wu. (CVPR, 2017)
- **“Working hard to know your neighbor’s margins”**: Local descriptor learning loss (HardNet)
  - Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenovic´, Jirˇi Matas (NIPS, 2017)





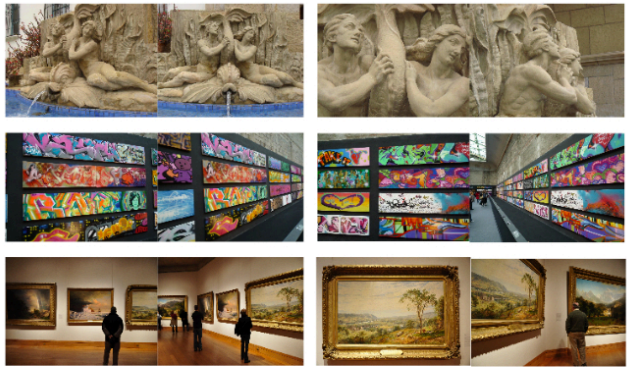
Some existing datasets

- **Multi-view Stereo Correspondence Dataset**
  - S. Winder and M. Brown. ”**Learning Local Image Descriptors**.” *CVPR2007*
  - 3 scenes : Statue of Liberty , Notre Dame, and Half Dome, 2,400,000 matching pairs
  - The data is taken from ***Photo Tourism*** reconstructions.

Some existing datasets

- **PS-Dataset**
  - Rahul Mitra, Nehal Doiphode, Utkarsh Gautam, Sanath Narayan, Shuaib Ahmed, Sharat Chandran, Arjun Jain, **"A Large Dataset for Improving Patch Matching."** arXiv:1801.01466, 2018.
  - 30 scenes, 20,600,000 matching pairs
  - images from ***Microsoft PhotoSynth***, adapt SFM to create ground truth pairs of correspondence.

(a) Image pairs showing illumination variation.



(b) Image pairs showing viewpoint variation.

(c) Image pairs showing scale variation.

Problems: existing datasets do not have sufficiently large illumination variation for training

Dataset we collected

AMOS-crop dataset:

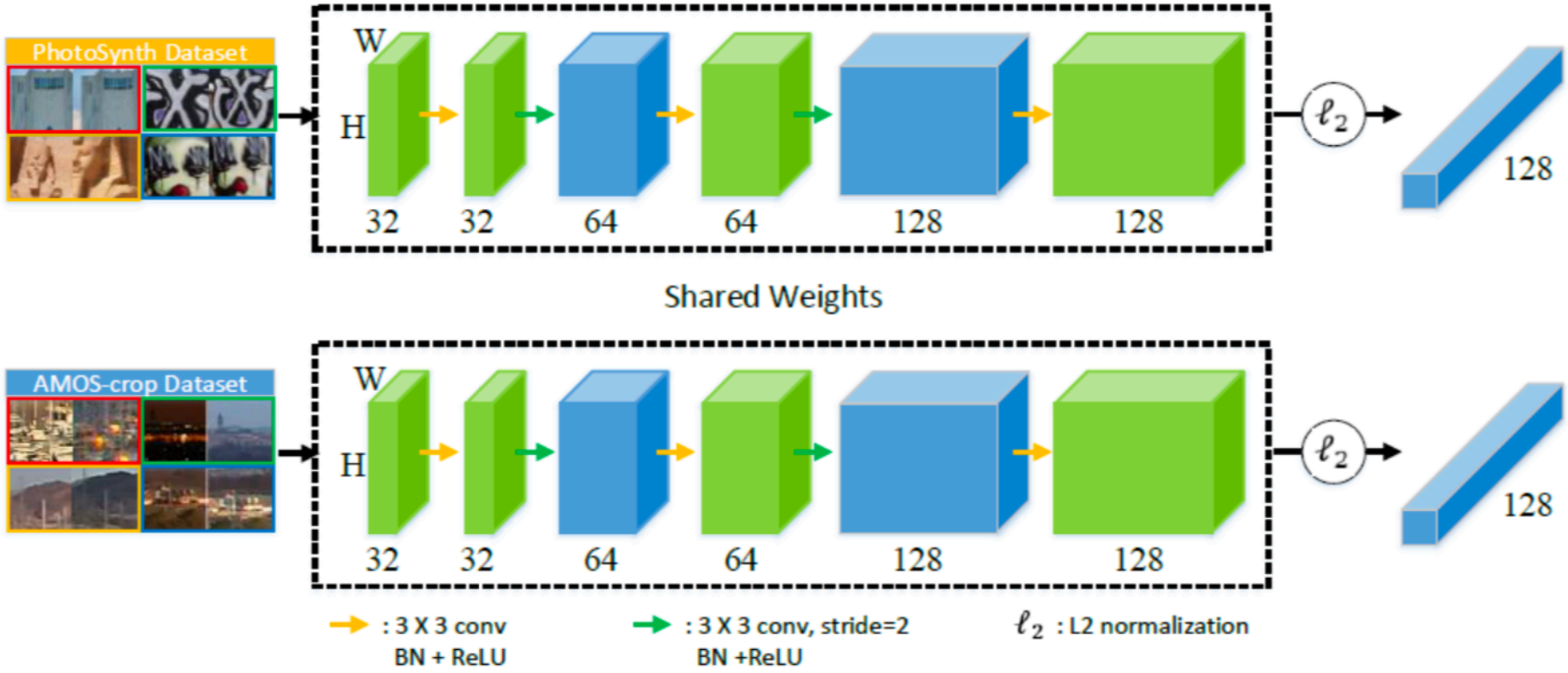
1. Inspiration AMOS dataset [1] (time-lapse+ fixed camera position).
2. SIFT, SURF, ORB to detect feature points.
3. Total about 1 million patches.

[1] N. Jacobs; N. Roman; and R. Pless. Consistent Temporal Variations in Many Outdoor Scenes, In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2007

IF-Net

Illumination-invariant Feature Network (IF-Net):

- Separation training scheme
- Reduce-Outlier-Inlier (ROI) loss function



Po-Heng Chen, Zhao-Xu Luo, Zu-Kuan Huang, Chun Yang, and Kuan-Wen Chen, “IF-Net: An Illumination-invariant Feature Network,” ICRA 2020.

IF-Net

- **ROI loss function:**

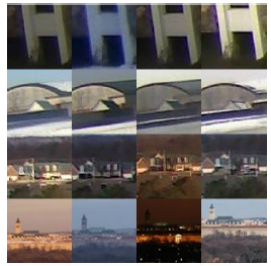
a) $Loss\ funtion : \sum_{i=1}^{n} \max(W_{original} * d(a_i, p_i) - W_{light} * d(a_i, n_i) + 1, 0)$

b) $W_{original}, W_{light} = Softmax\left(\frac{1}{N}\sum_{i=1}^{n} d(a_i, p_i), \frac{1}{N}\sum_{i=1}^{n} d(a_i, n_i)\right)$

c) $Calculate\ L_{original}, L_{light}\ from\ PS\ dataset\ and\ Amos\ crop\ dataset$

d) $L_{total} = L_{original} + L_{light}$

PS dataset



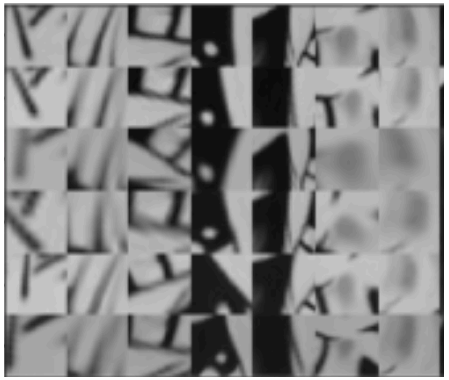
Amos-crop dataset

- **Design idea:**
  - Anchor patch may not choose well → call it outlier
  - By multiplying weights in loss function → reduce impact of outliers
  - Reduce contribution of potential outliers in training

Po-Heng Chen, Zhao-Xu Luo, Zu-Kuan Huang, Chun Yang, and Kuan-Wen Chen, “IF-Net: An Illumination-invariant Feature Network,” ICRA 2020.

Experiment Results: HPatches Dataset

- Quantitative results (accuracy%):
  - The higher the better
- HPatches (CVPR, 2017):
  - A public benchmark for patch matching



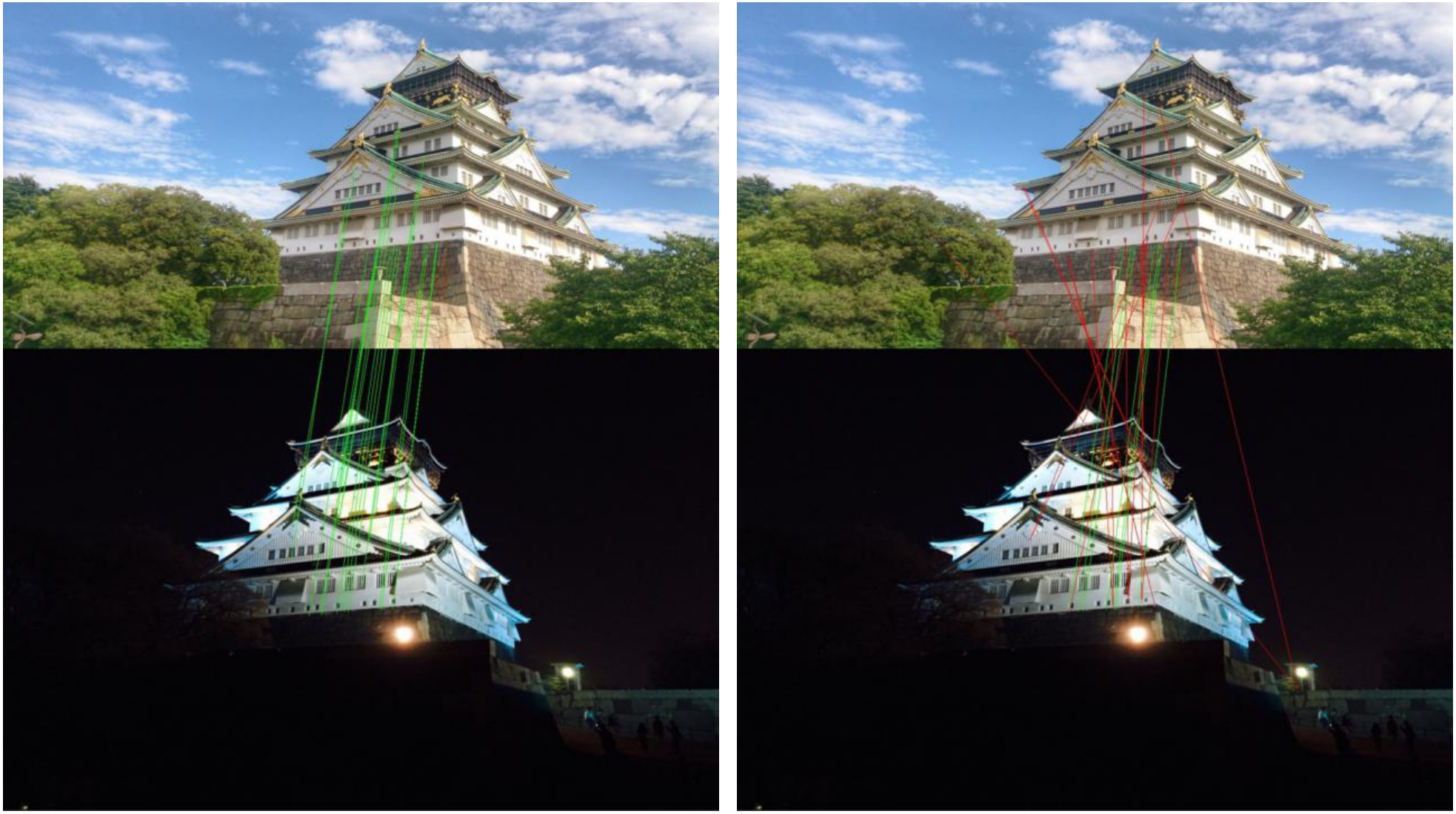
		HardNet	IF-Net
Verification	Inter	**93.85**	93.83
	Intra	93.53	**93.57**
Retrieval		60.85	**67.31**
Matching		41.64	**48.65**

Experiment Results: IMD

- Proposed **Illumination Matching Dataset (IMD)**
- Contain *Easy* and *Hard* levels:
  - Easy: small illumination changes
  - Hard: large illumination & viewpoint changes

	Scene Level	HardNet	IF-Net
Total	Easy	70.9%	**79.7%**
	Hard	55.3%	**62.4%**
Top-40	Easy	86.6%	**94.2%**
	Hard	72.7%	**83.2%**

Experiment Results: IMD

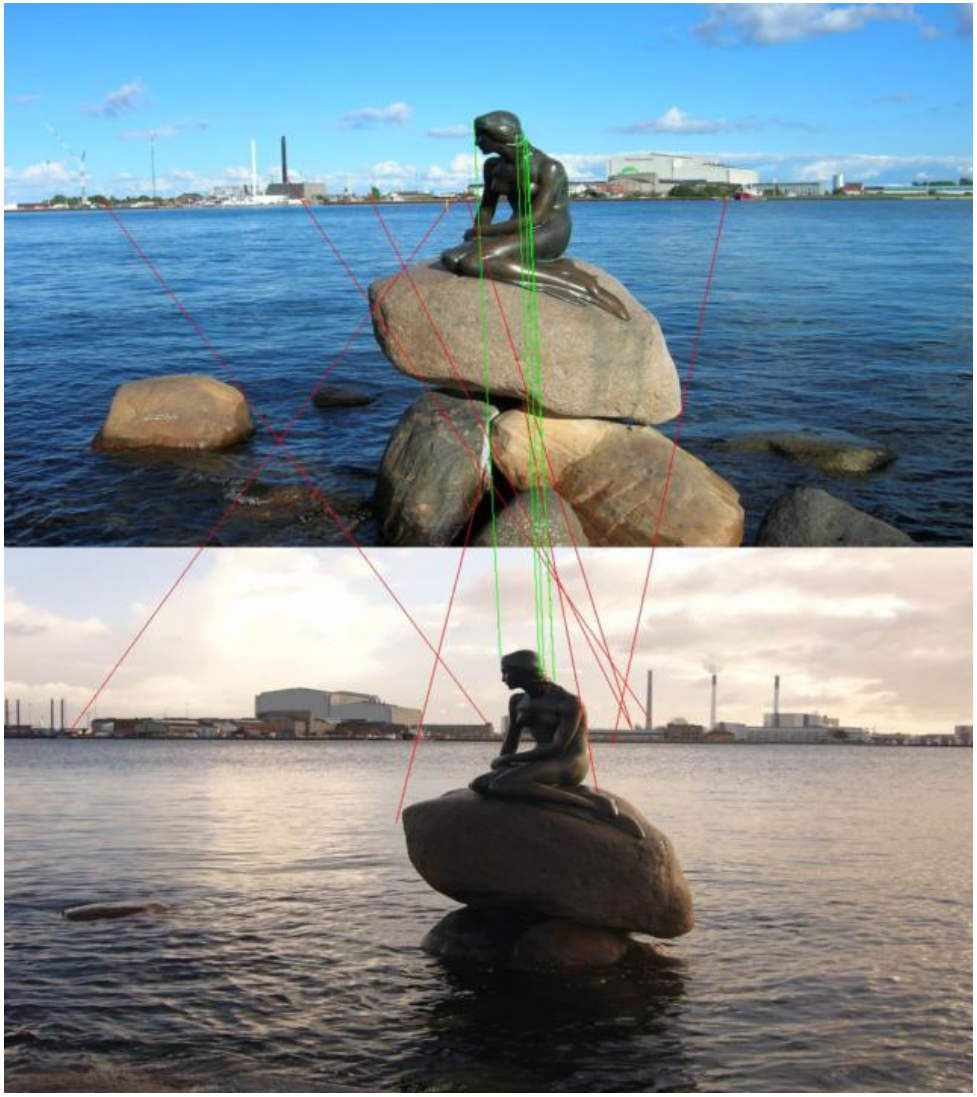


IF-Net

HardNet

Experiment Results: IMD

IF-Net



HardNet

Experiment Results: IMD

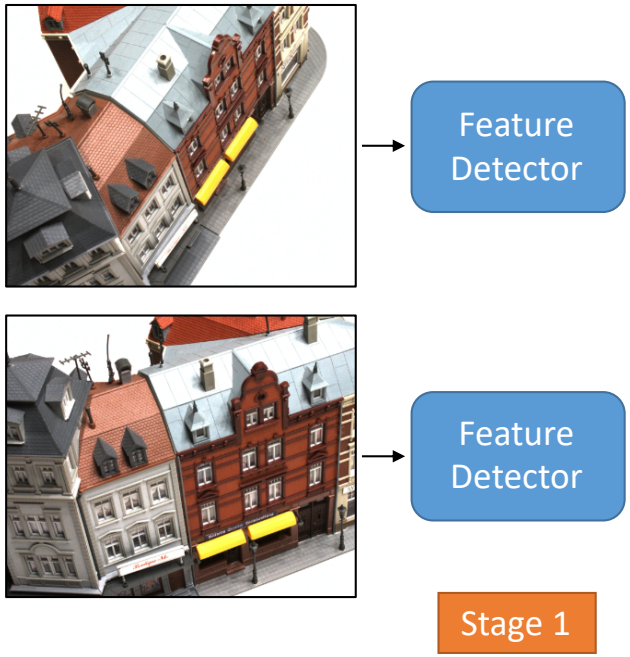
IF-Net

HardNet

An end-to-end learning for detector and descriptor simultaneously

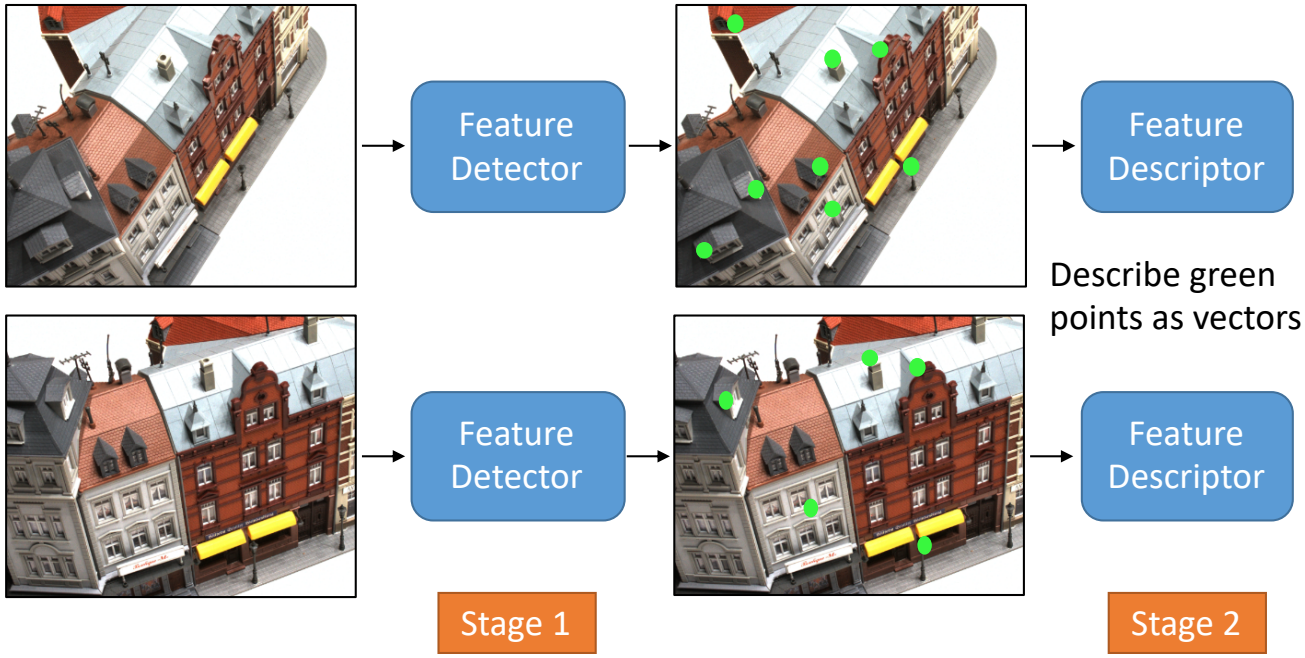
Why end-to-end?

- Traditional feature matching pipelines:
  - Feature detection and description are separated tasks



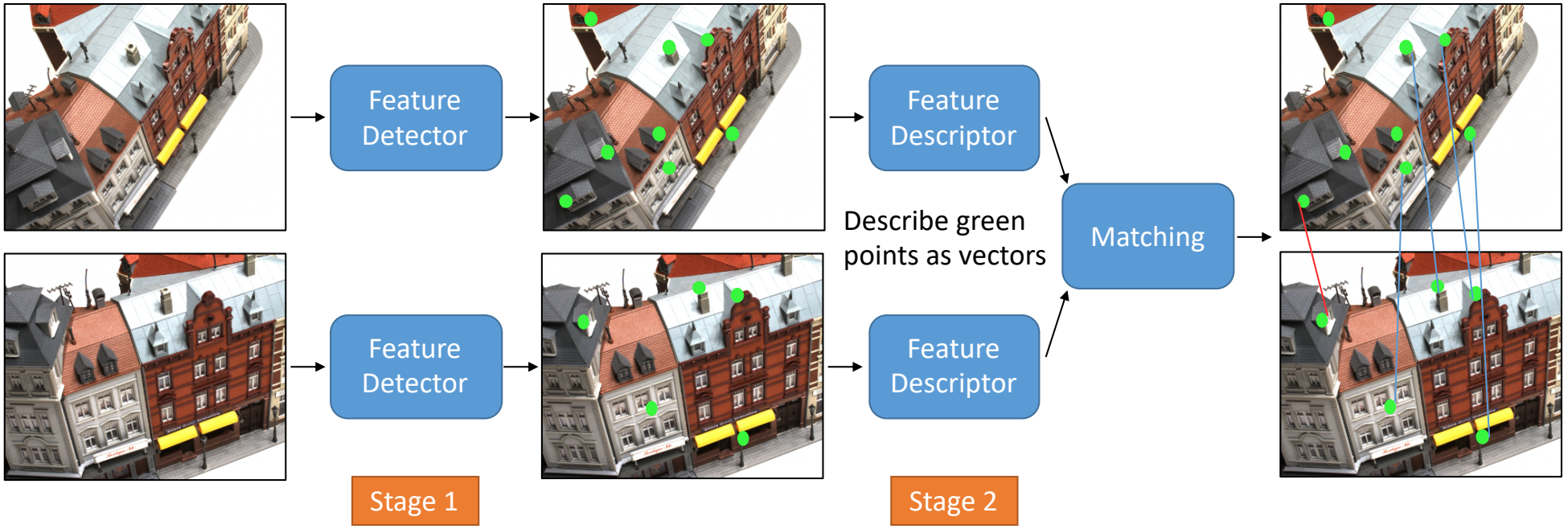
Why end-to-end?

- Traditional feature matching pipelines:
  - Feature detection and description are separated tasks

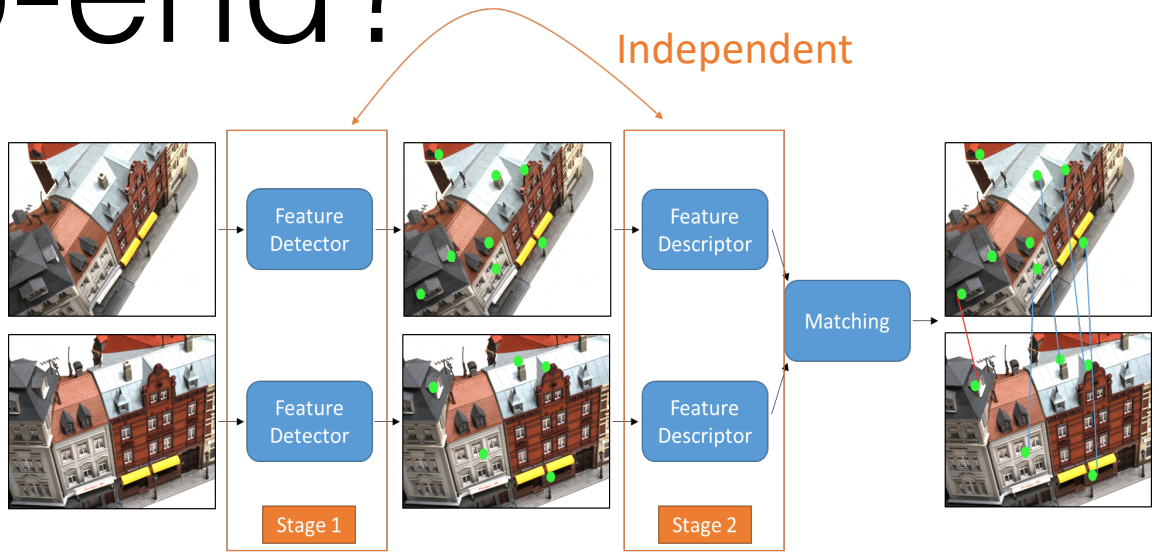


Why end-to-end?

- Traditional feature matching pipelines:
  - Feature detection and description are separated tasks



Why end-to-end?

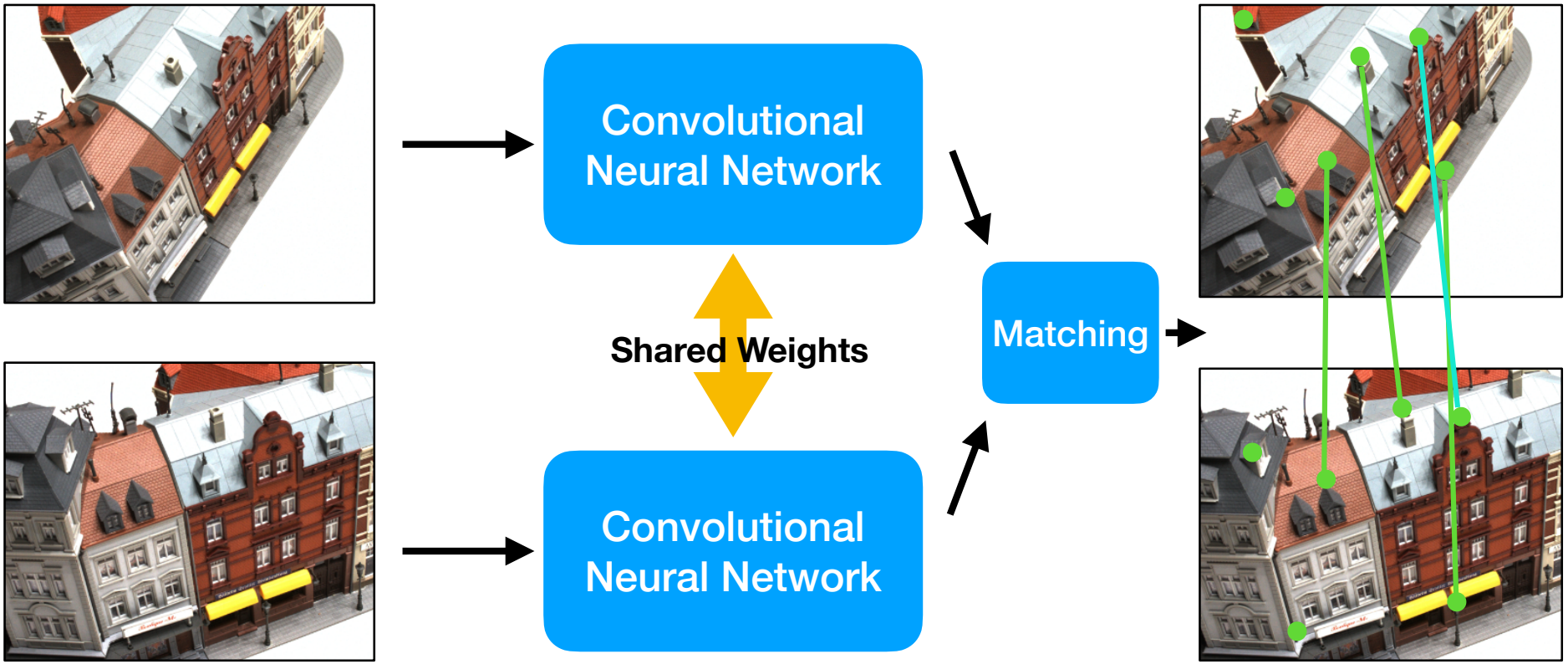


- Traditional feature matching pipelines:
  - Feature detection and description are separated tasks

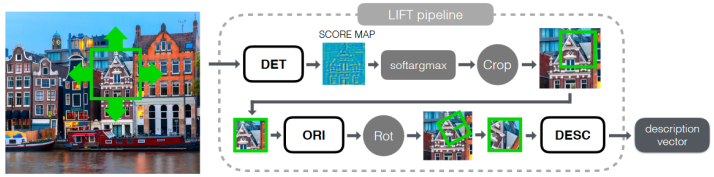
- However, such pipeline exists two drawbacks:
  - **Detector and descriptor are independent**
    - Detector should detect keypoints that are distinctive for descriptor to describe
    - Descriptor should learn to describe features that are repeatable
  - **The receptive field of detector and descriptor is different**
    - Detector consider low-level information (e.g., corners, edges, …)
    - Descriptor consider high-level information (e.g., semantic features)

Why end-to-end?

Integrate feature detection and description to one-stage pipeline

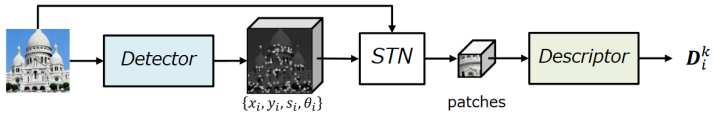


Existing End-to-End Feature Learning



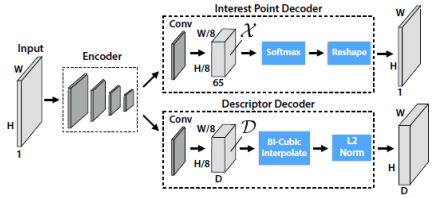
ECCV' 16

LIFT



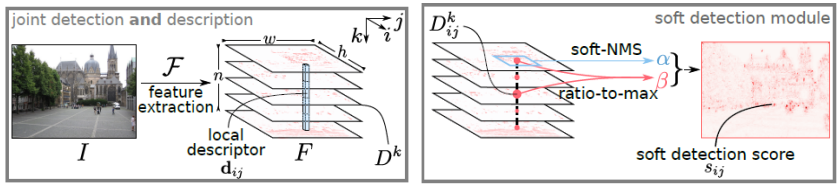
NIPS' 18

LF-Net



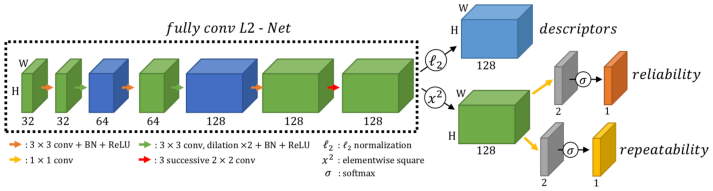
CVPR' 18

SuperPoint



CVPR' 19

D2-Net



NIPS' 19

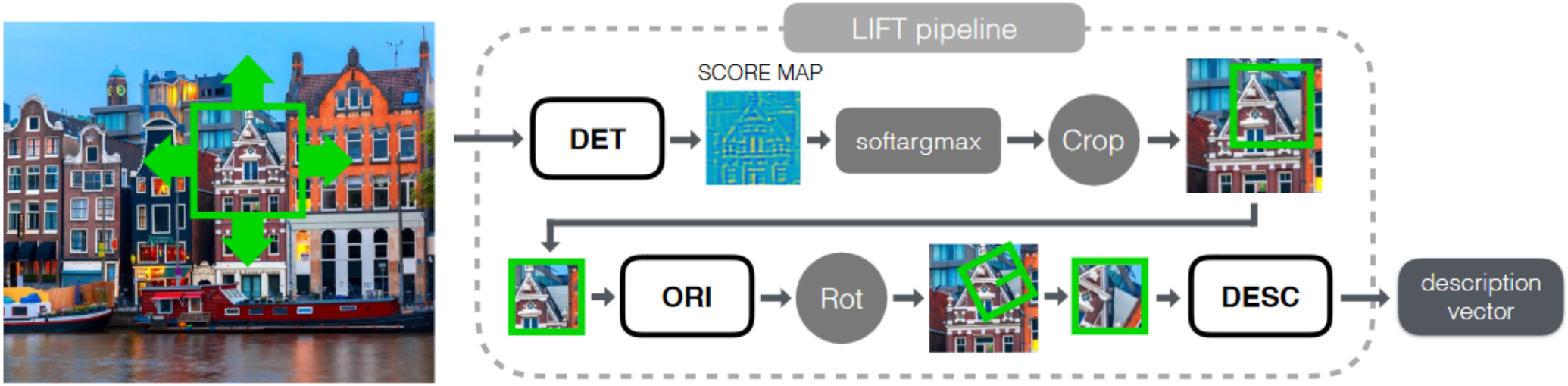
R2D2

End-to-End Methods

- **LIFT: Learned Invariant Feature Transform, ECCV'16**
  - The first work to combine detection and description using CNN
- **SuperPoint: Self-Supervised Interest Point Detection and Description, CVPR'18**
  - The first work for dense feature description
- **D2-Net: A Trainable CNN for Joint Detection and Description of Local Features, CVPR'19**
  - State-of-the-art performance

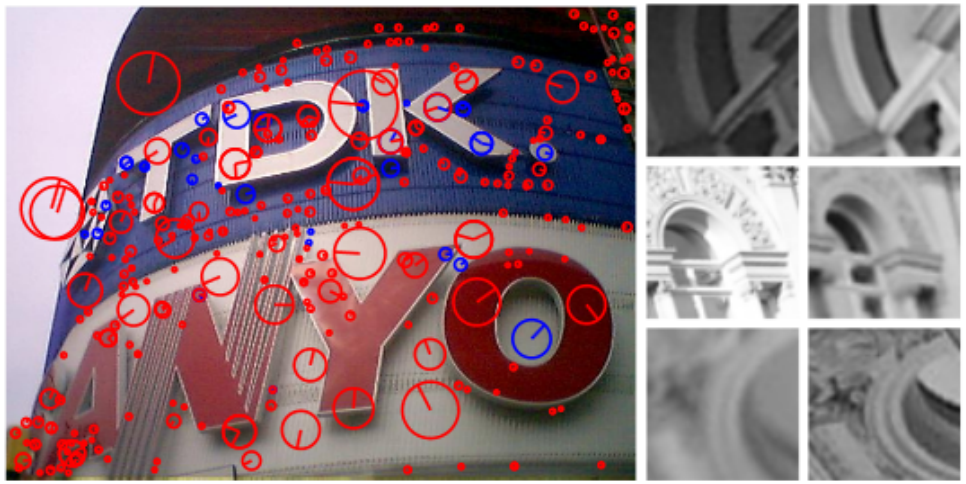
LIFT, ECCV' 16

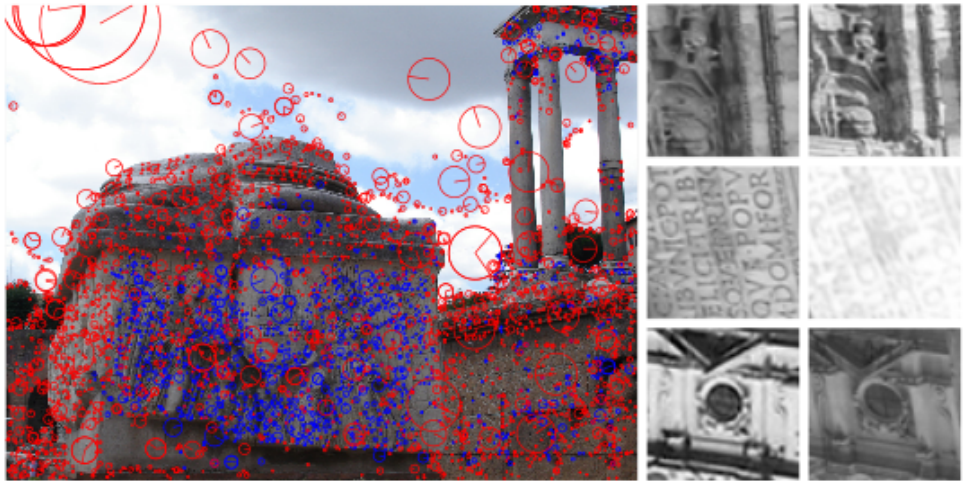
- First end-to-end matching framework consists of:
  - Feature detector
  - Orientation estimator
  - Feature descriptor



LIFT, ECCV' 16

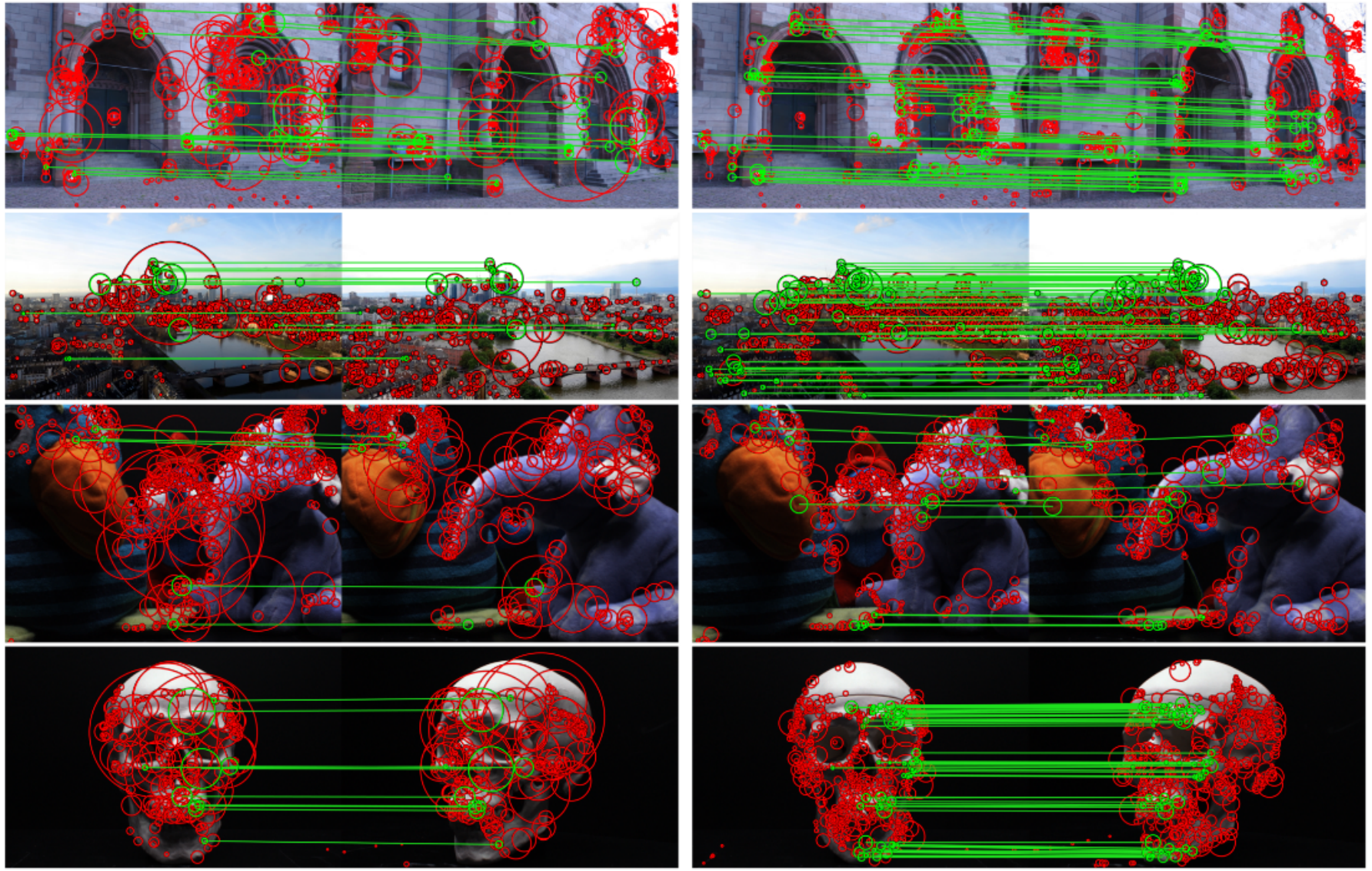
- The training data is collected from Structure-from-Motion (SfM) pipeline:
  - The inputs are patches, not image





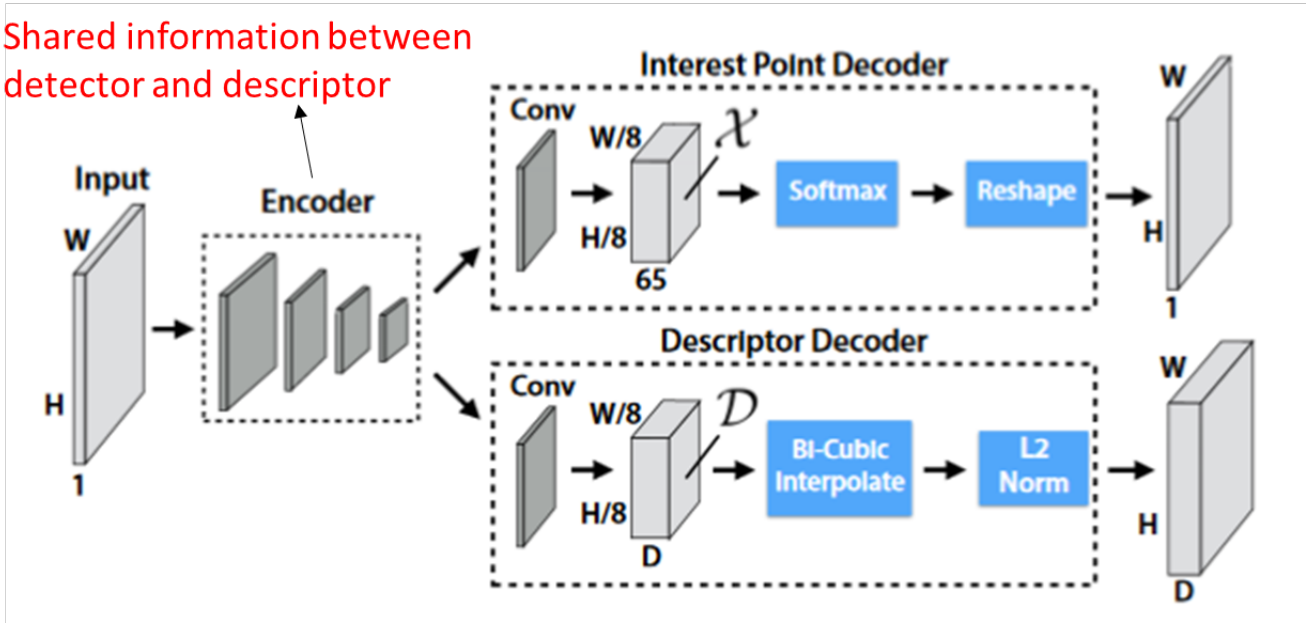
Keypoints survive from SfM are blue, the rest are red

LIFT, ECCV' 16



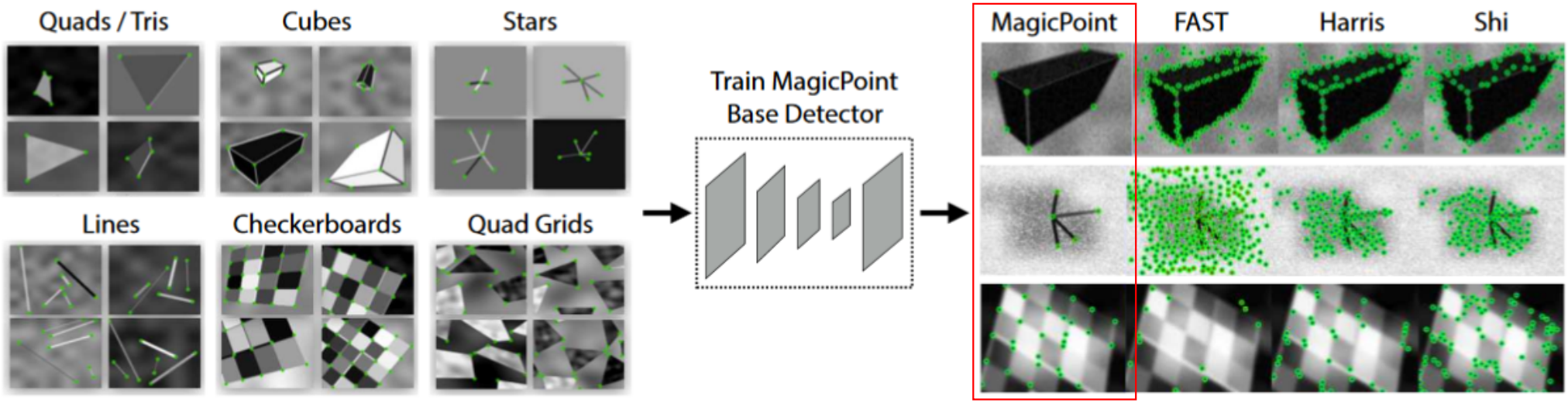
SuperPoint, CVPR' 18

- Instead of using patch as input, they use whole image as input
  - First pre-train interest points detector → form base detector
  - Using base detector with homography warping to label more training data
  - Joint training detector & descriptor



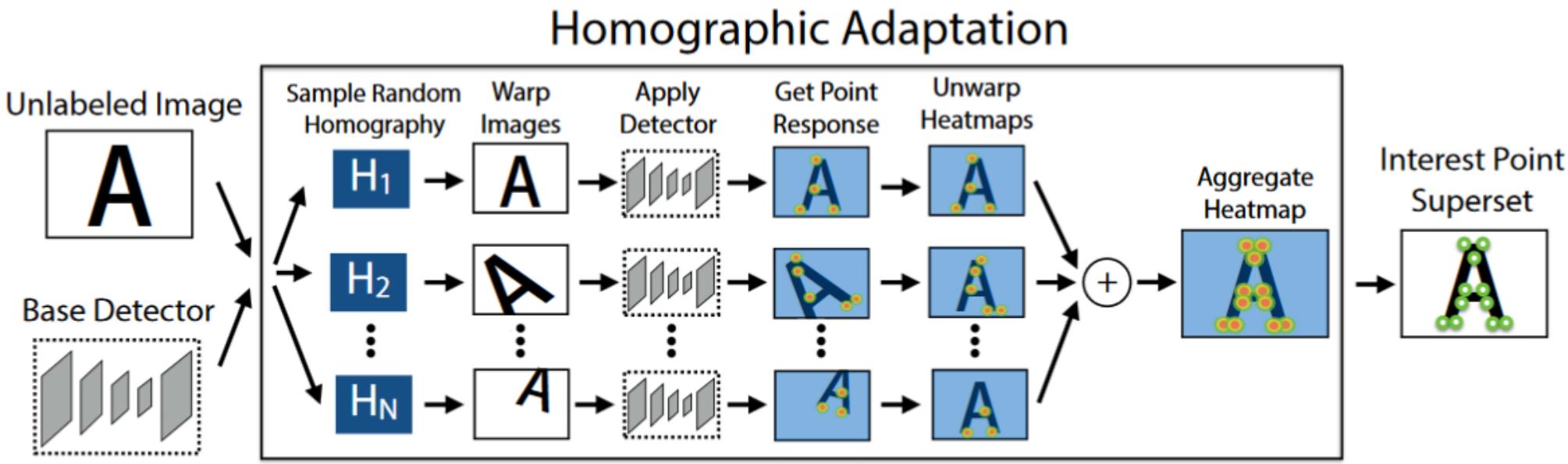
SuperPoint, CVPR' 18

- Interest points detector pre-training
  - Using synthetic dataset (generated by the authors)
  - Make the detector to learn basic keypoints (e.g., corners, …)



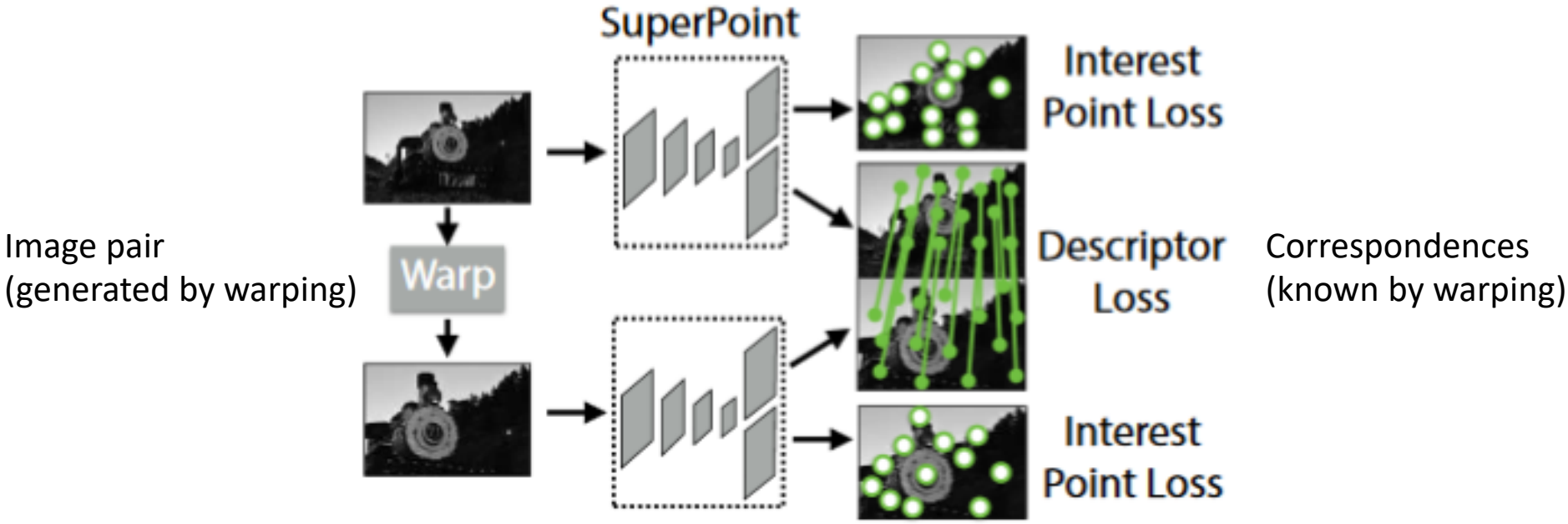
SuperPoint, CVPR' 18

- Using base detector with homography warping to label more training data

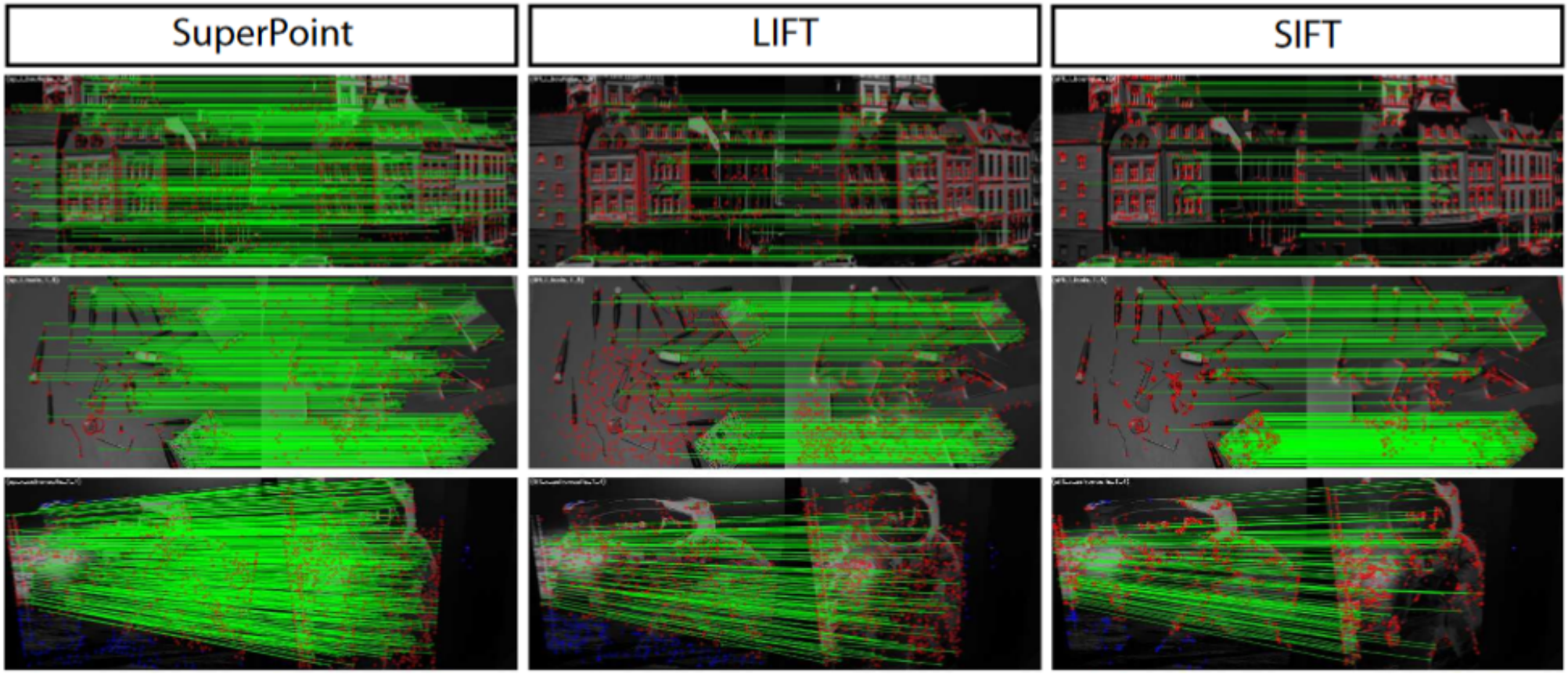


SuperPoint, CVPR' 18

- Final step: joint training
- By warping → we will know the ground truth correspondences



SuperPoint, CVPR' 18

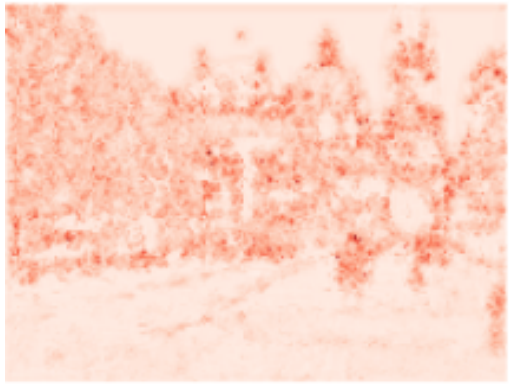


D2-Net, CVPR' 19

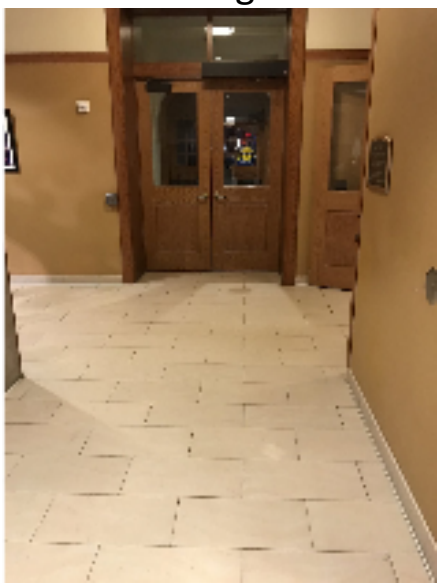
- Current state-of-the-art performance
- Using pre-trained VGG16 to generate **score maps**
  - Using score maps to train the network

Image

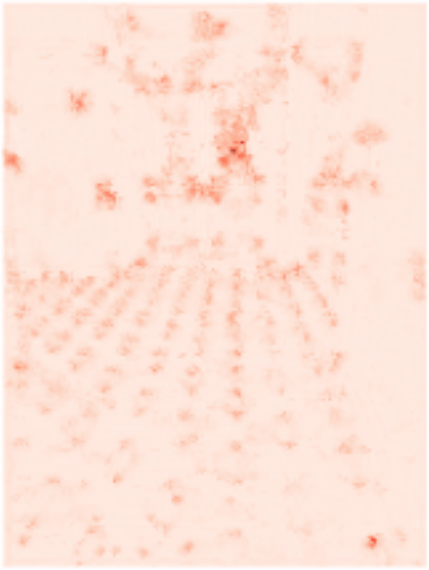
Score map



Image

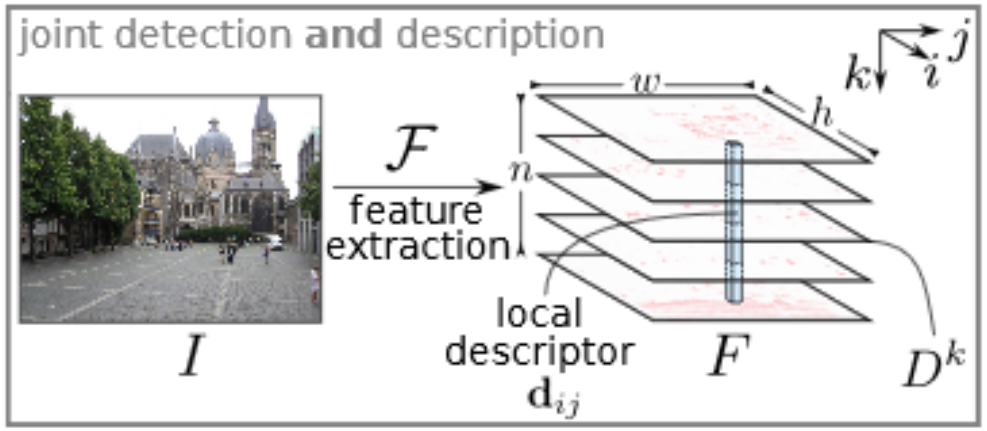


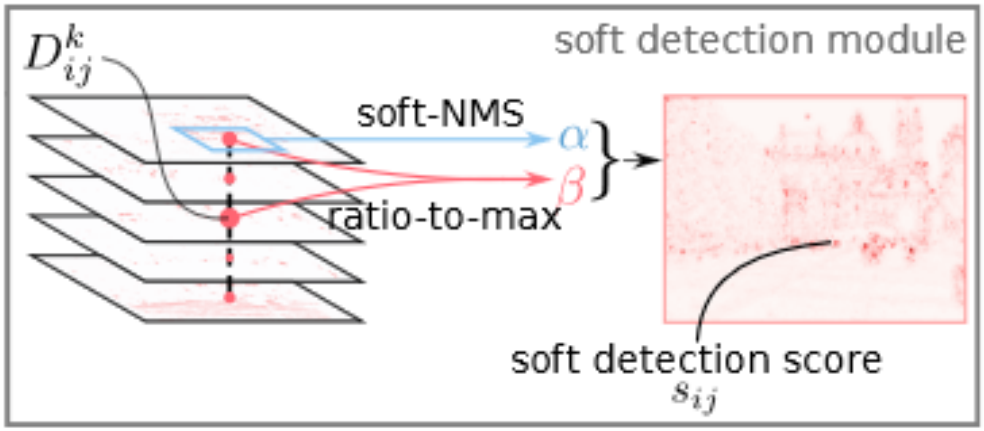
Score map



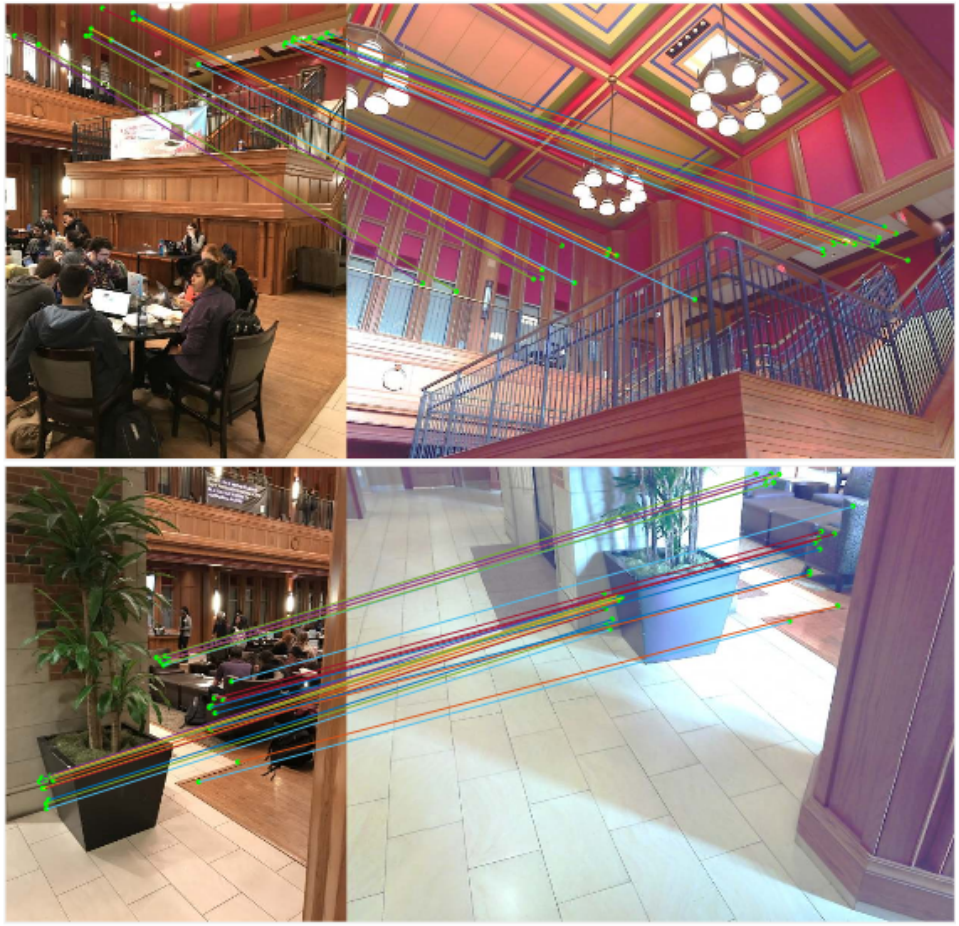
D2-Net, CVPR' 19

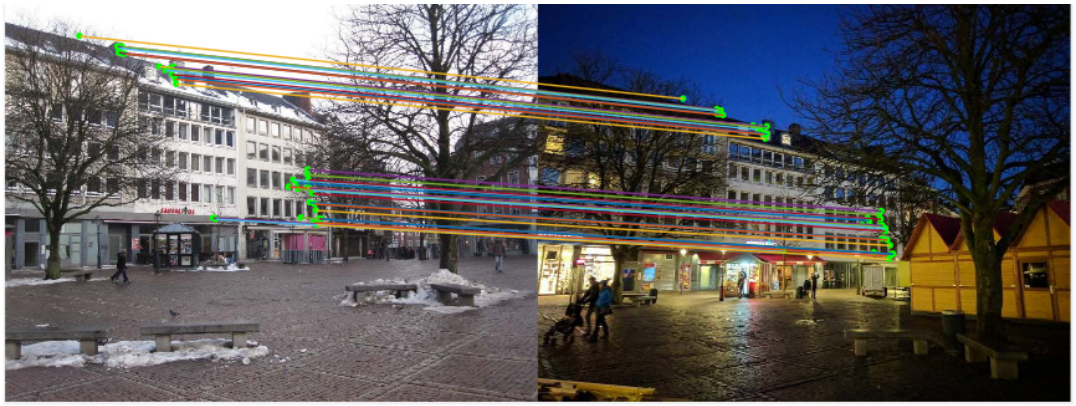
- Descriptor is extracted from VGG16 across channel dimension
- Detector
  - During training: soft-detection → generate score maps
  - During testing: hard-detection → satisfy spatial max **&** channel max





D2-Net, CVPR’ 19





Features Matching

Kuan-Wen Chen
2022/5/26