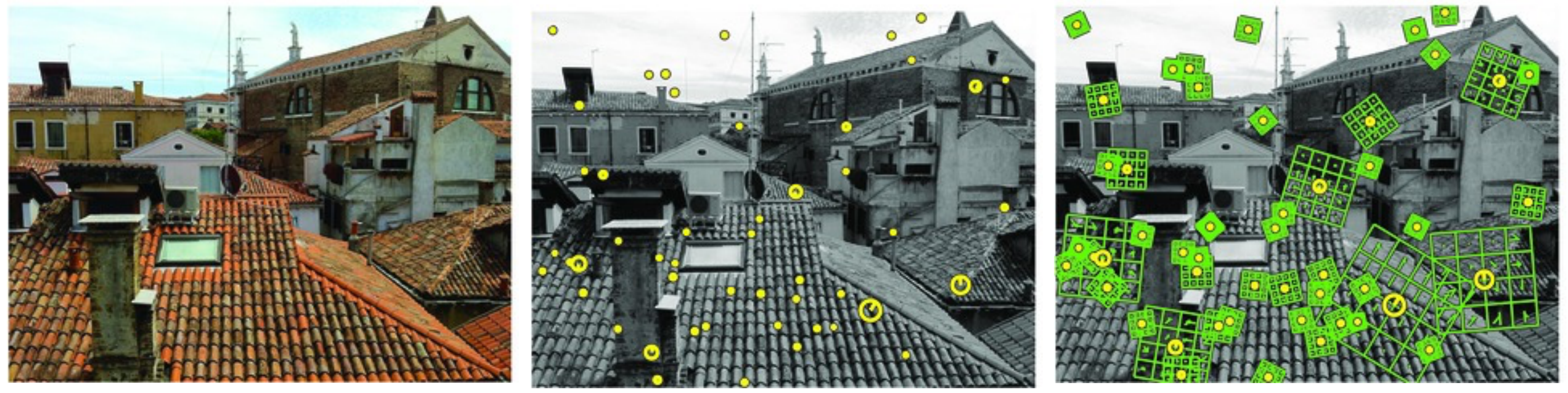
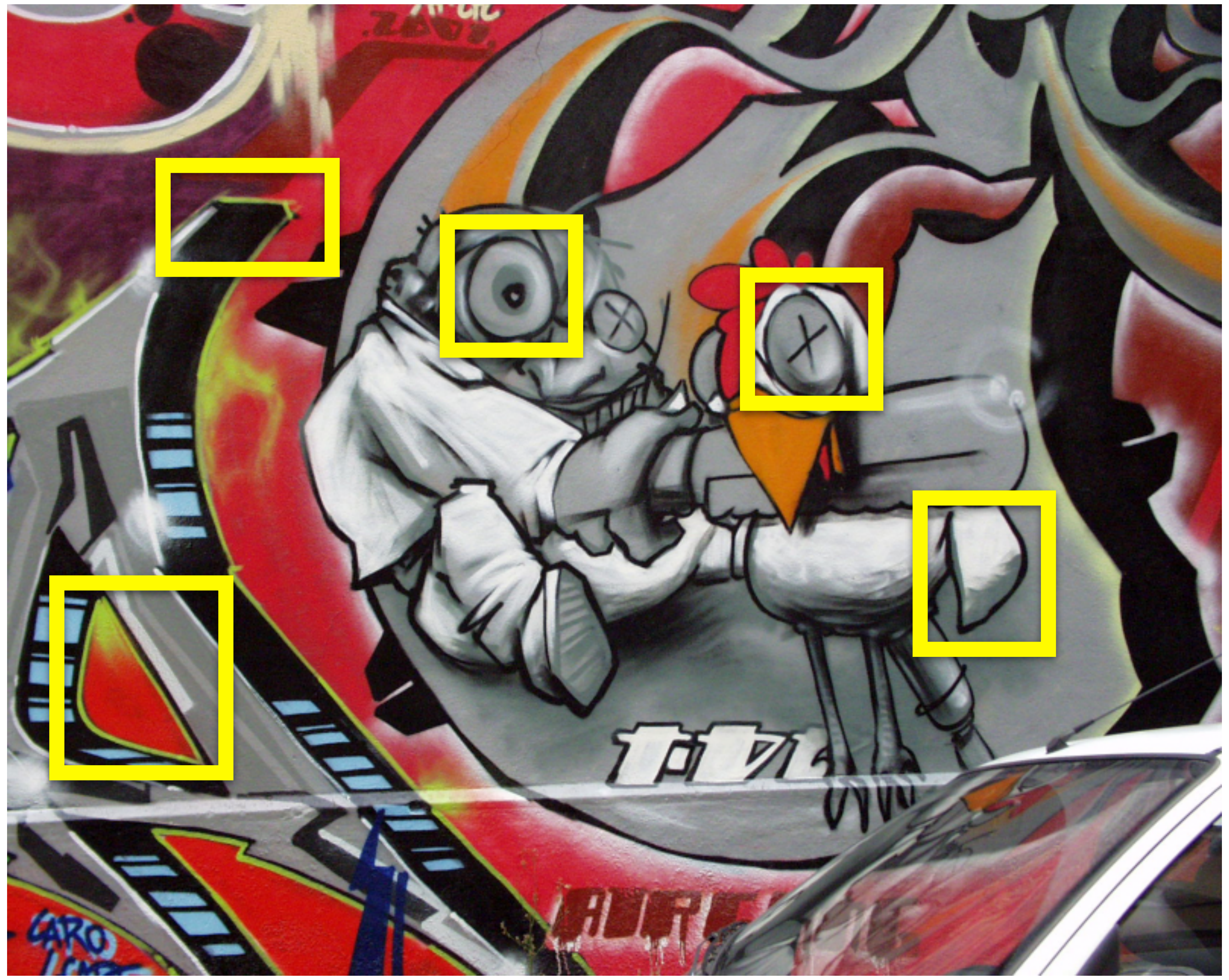
THE UNIVERSITY OF BRITISH COLUMBIA

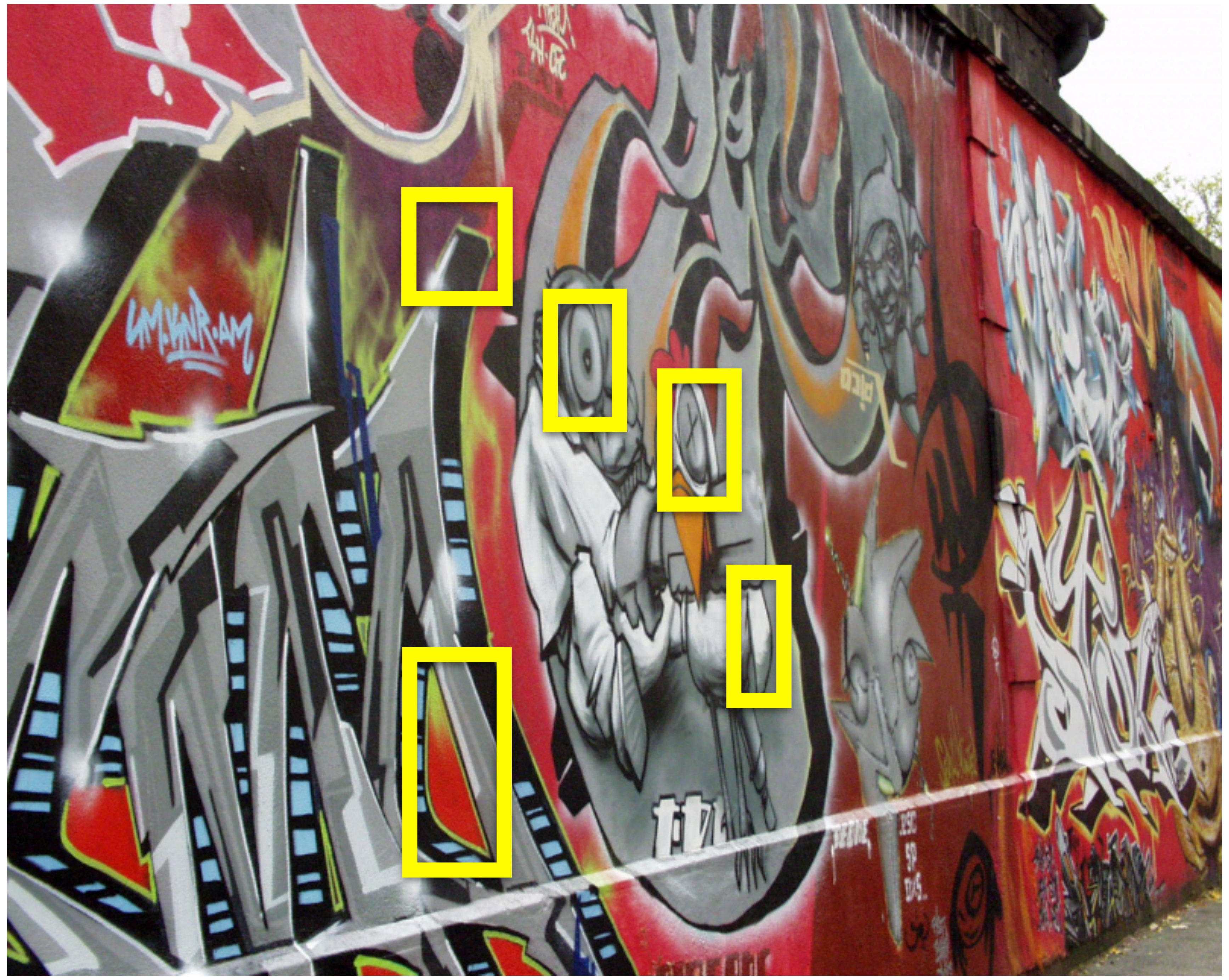
CPSC 425: Computer Vision



Lecture 12: Scale Invariant Features (SIFT)

Back to **Good Local Features**





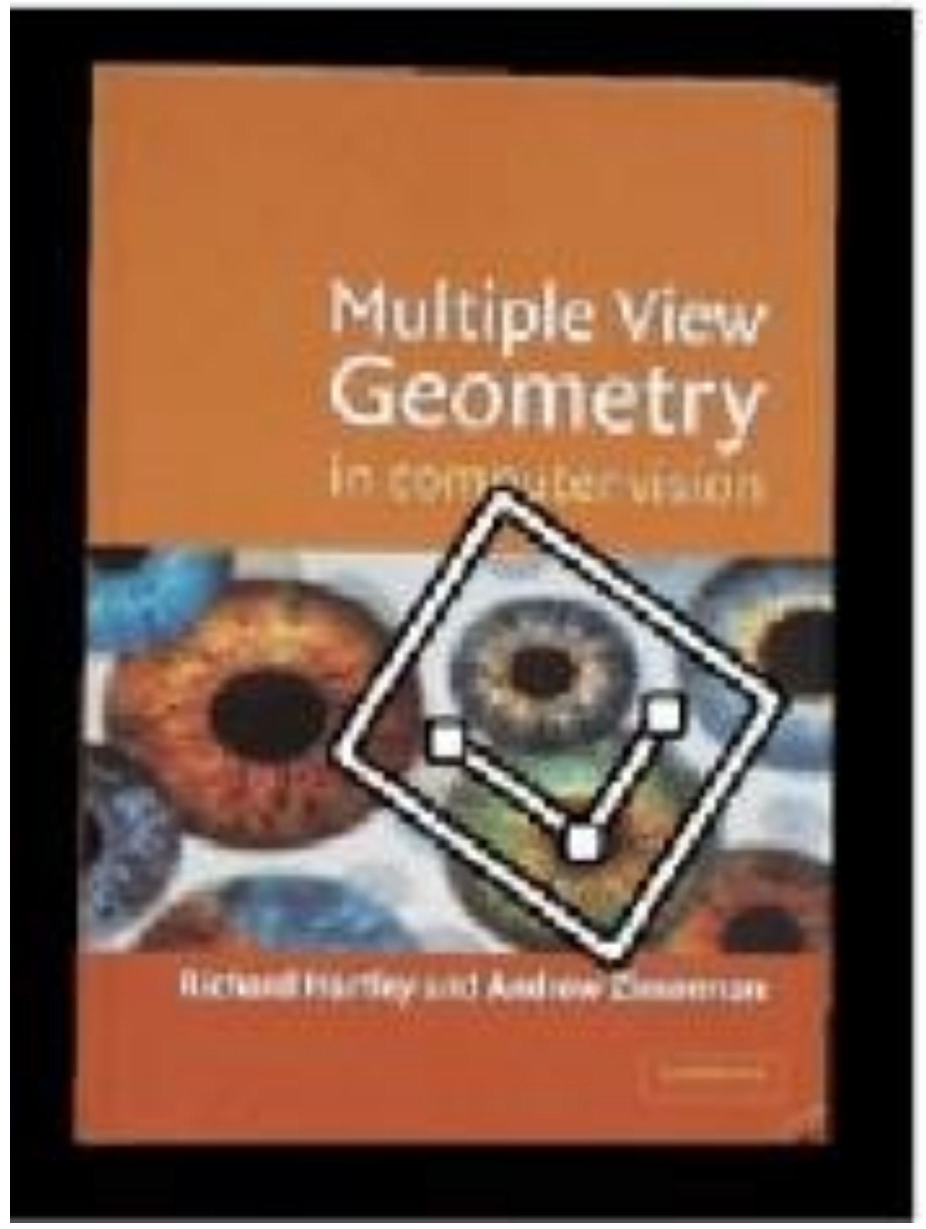
Where are the good features, and
how do we match them?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Photometric Transformations

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Geometric Transformations



objects will appear at different scales,
translation and rotation

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lets assume for the moment we can figure out where the good features (patches) are … how do we **match** them?

Intensity Image

Just use the pixel values of the patch

Perfectly fine if geometry and appearance is unchanged

(a.k.a. template matching)

What are the problems?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Intensity Image

Just use the pixel values of the patch

Perfectly fine if geometry and appearance is unchanged
(a.k.a. template matching)

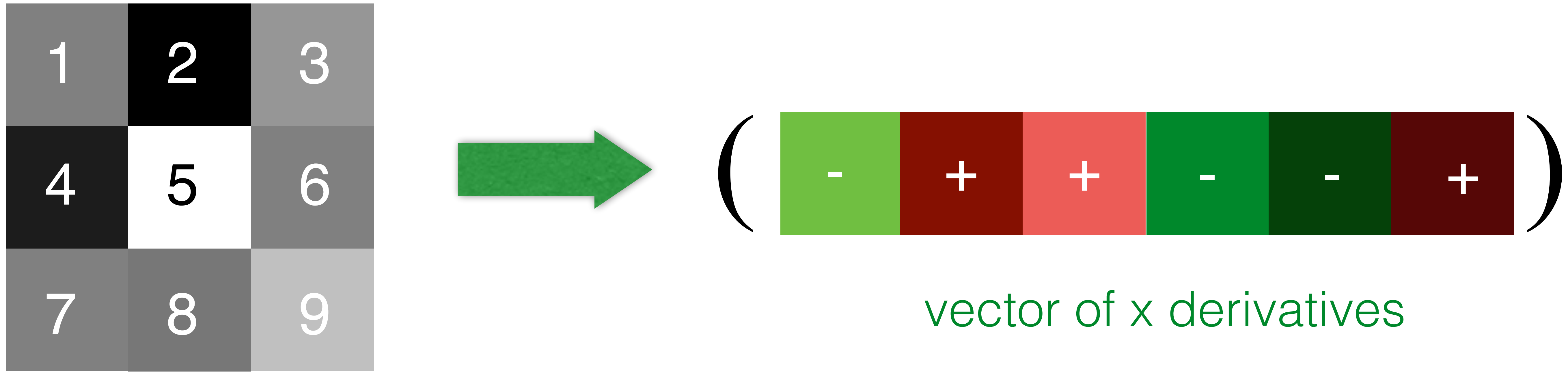
What are the problems?

How can you be less sensitive to absolute intensity values?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Image **Gradients / Edges**

Use pixel differences



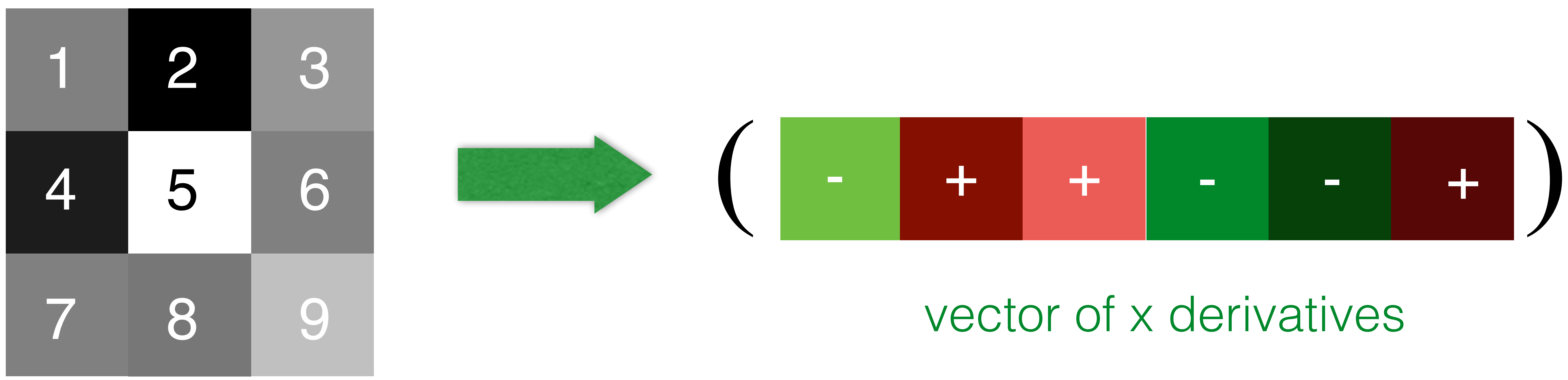
Feature is invariant to absolute intensity values

What are the problems?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Image **Gradients** / **Edges**

Use pixel differences



Feature is invariant to absolute intensity values

What are the problems?

How can you be less sensitive to deformations?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Where does **SIFT** fit in?

Representation	Result is. . .	Approach	Technique
intensity	dense (2D)	template matching	(normalized) correlation, SSD
edge	relatively sparse (1D)	derivatives	$\triangledown^2 G$, Canny
"corner"	sparse (0D)	locally distinct features	Harris, SIFT

Object **Recognition** with Invariant Features

Task: Identify objects or scenes and determine their pose and model parameters

Applications:

— Industrial automation and inspection

— Mobile robots, toys, user interfaces

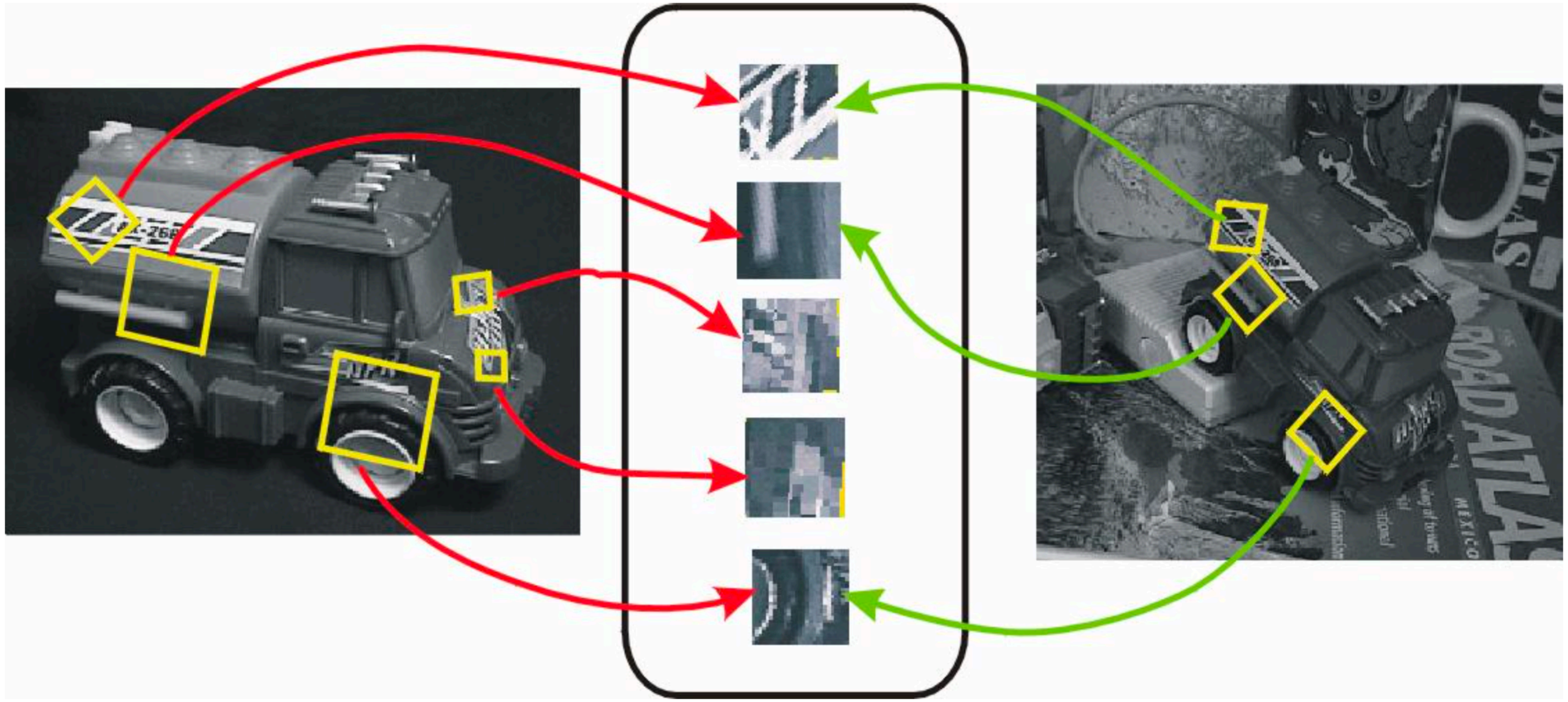
— Location recognition

— Digital camera panoramas

— 3D scene modeling, augmented reality

David Lowe's Invariant Local Features

Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



SIFT Features

Advantages of Invariant Local Features

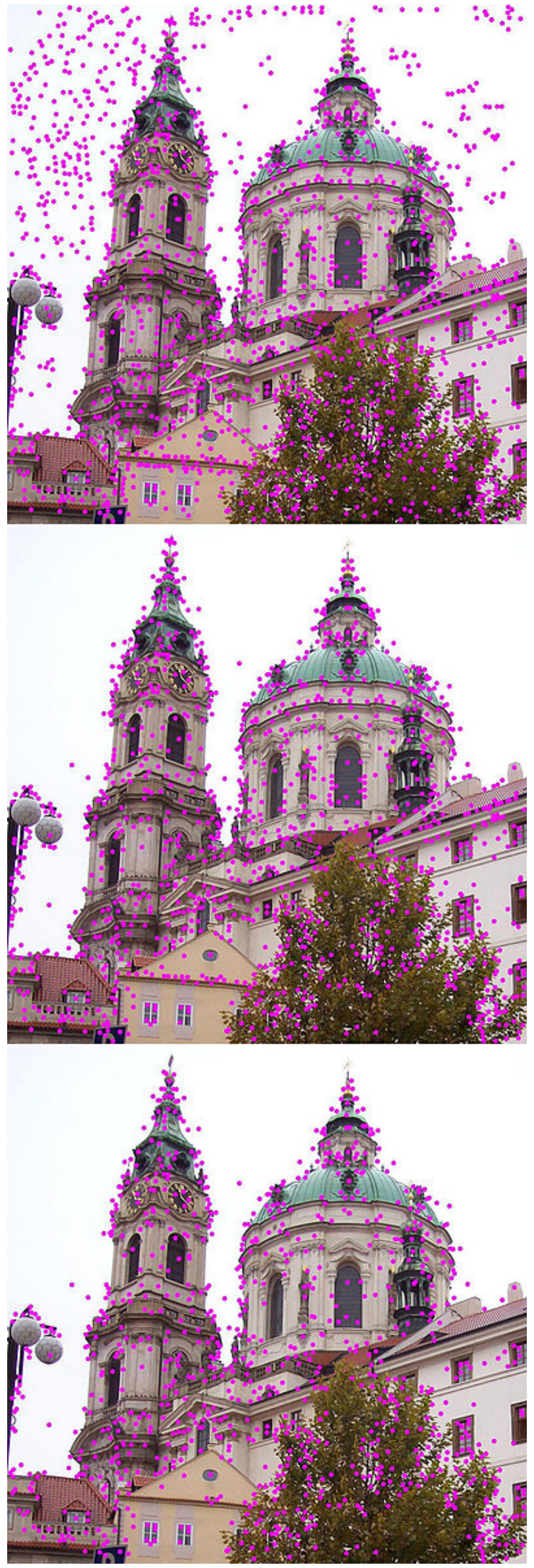
Locality: features are local, so robust to occlusion and clutter (no prior segmentation)

Distinctiveness: individual features can be matched to a large database of objects

Quantity: many features can be generated for even small objects

Efficiency: close to real-time performance

Scale Invariant Feature Transform (**SIFT**)

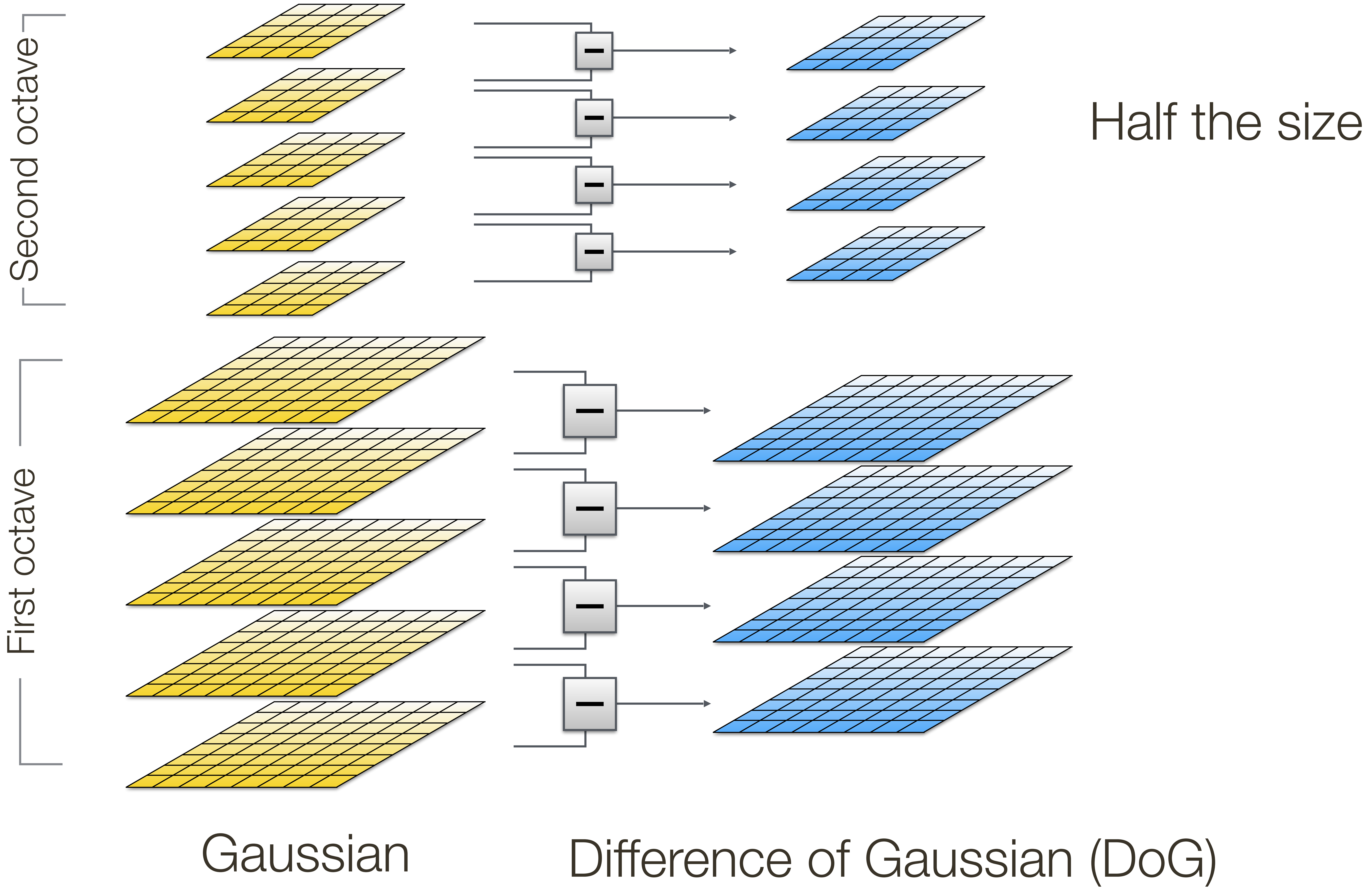


SIFT describes both a **detector** and **descriptor**

1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

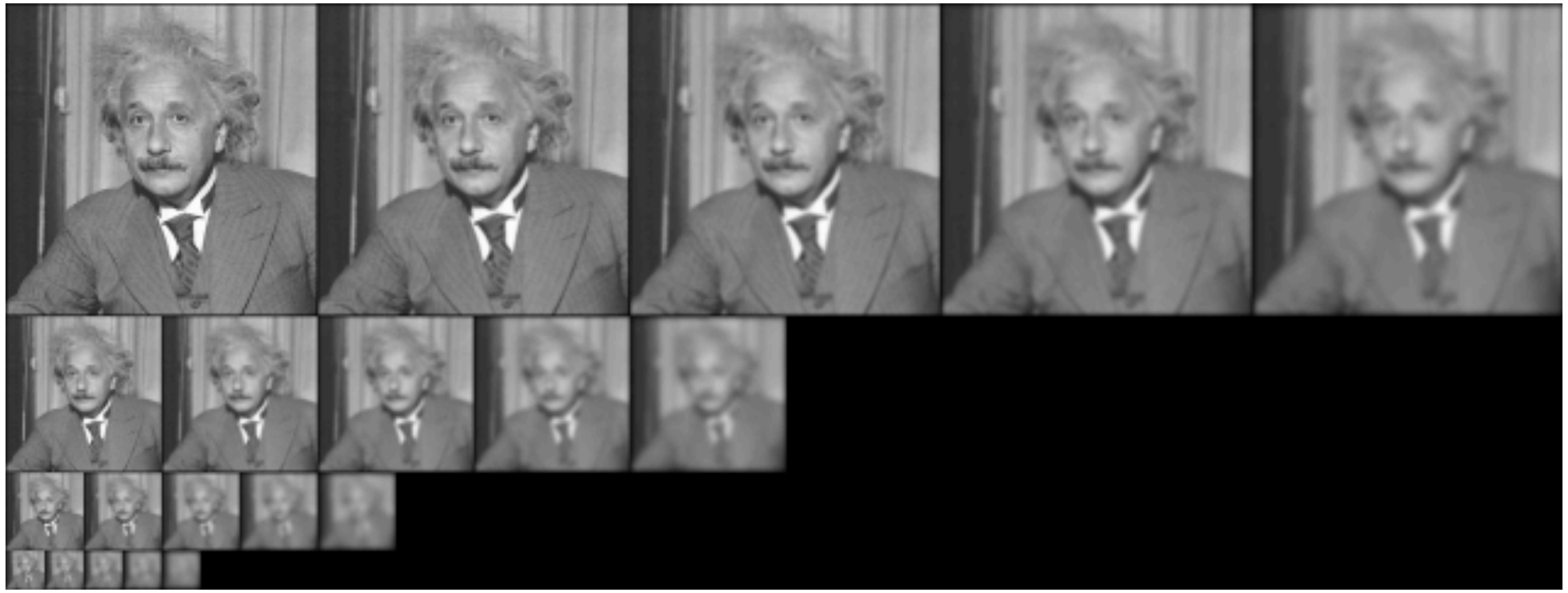
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

1. Multi-scale Extrema Detection



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

1. Multi-scale Extrema Detection



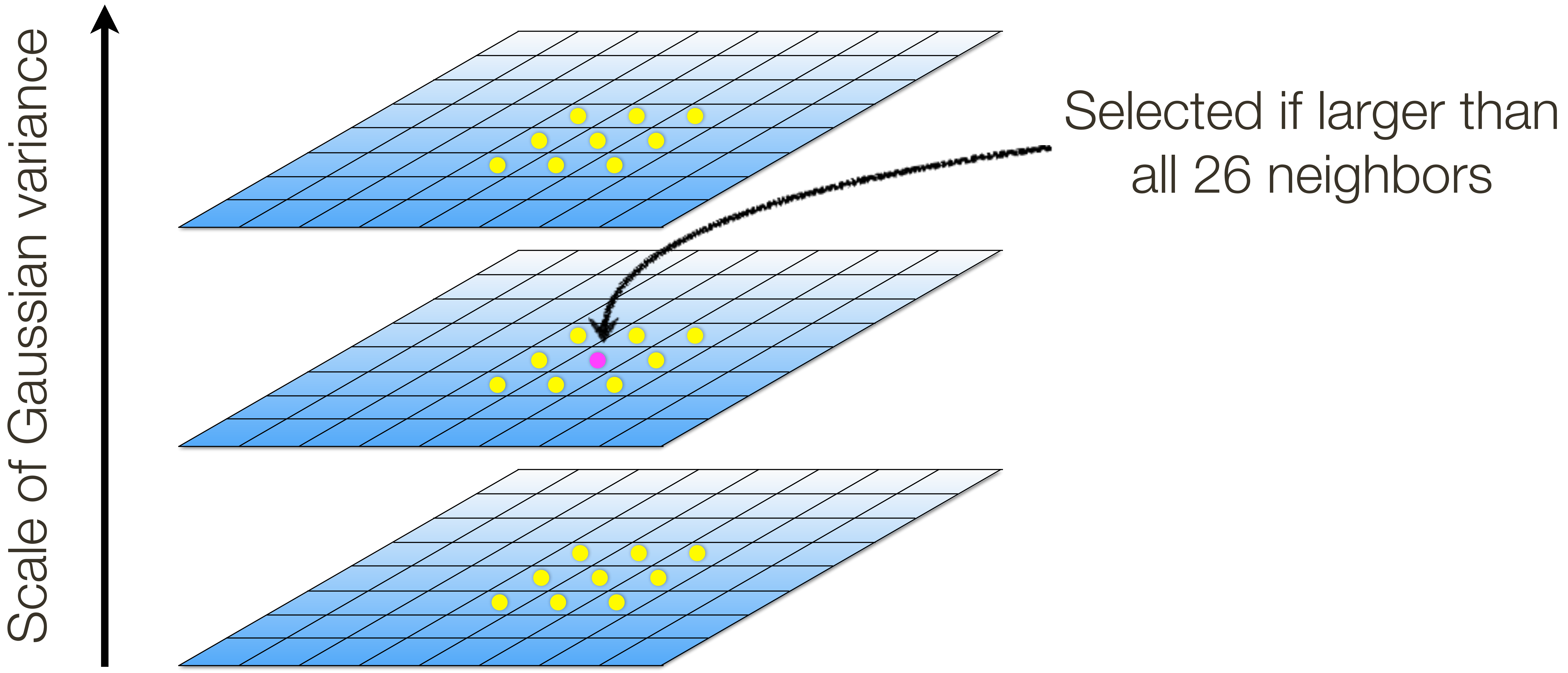
Gaussian

Laplacian

1. Multi-scale Extrema Detection

Detect maxima and minima of Difference of Gaussian in scale space

$$\sigma = 2^{1/s}$$

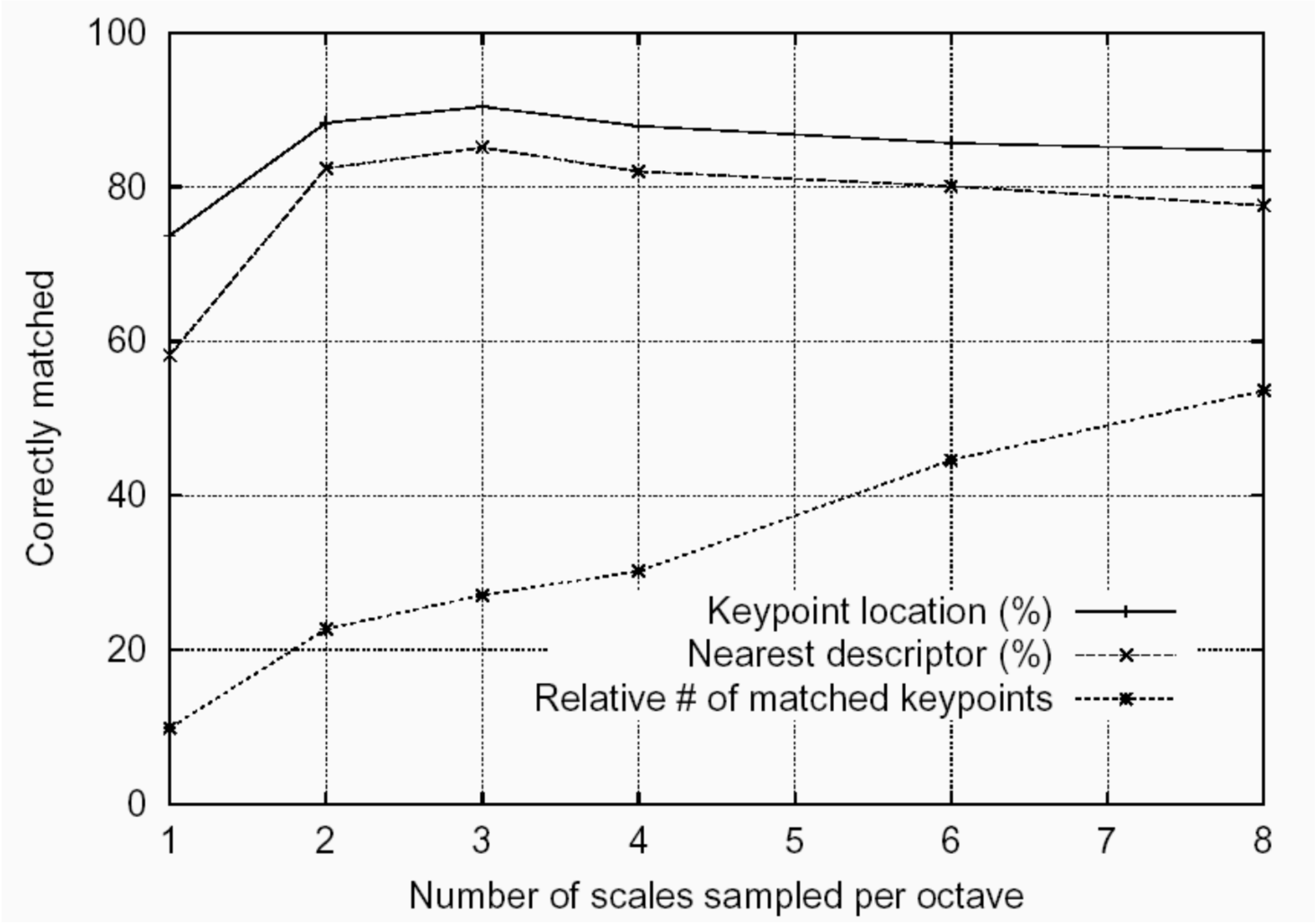


Difference of Gaussian (DoG)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

1. Multi-scale Extrema Detection — Sampling Frequency

More points are found as sampling frequency increases, but accuracy of matching decreases after 3 scales/octave



2. Keypoint Localization

— After keypoints are detected, we remove those that have **low contrast** or are **poorly localized** along an edge

2. Keypoint Localization

— After keypoints are detected, we remove those that have **low contrast** or are **poorly localized** along an edge

How do we decide whether a keypoint is poorly localized, say along an edge, vs. well-localized?

2. Keypoint Localization

— After keypoints are detected, we remove those that have **low contrast** or are **poorly localized** along an edge

How do we decide whether a keypoint is poorly localized, say along an edge, vs. well-localized?

$$C = \begin{bmatrix} \sum\limits_{p \in P} I_x I_x & \sum\limits_{p \in P} I_x I_y \\ \sum\limits_{p \in P} I_y I_x & \sum\limits_{p \in P} I_y I_y \end{bmatrix}$$

2. Keypoint Localization

— After keypoints are detected, we remove those that have **low contrast** or are **poorly localized** along an edge

How do we decide whether a keypoint is poorly localized, say along an edge, vs. well-localized?

— Lowe suggests computing the ratio of the eigenvalues of **C** (recall Harris corners) and checking if it is greater than a threshold

— Aside: The ratio can be computed efficiently in fewer than 20 floating point operations, using a trick involving the trace and determinant of **C** - no need to explicitly compute the eigenvalues

2. Keypoint Localization

Example:

(a) 233×189 image

(b) 832 DOG extrema

(c) 729 left after peak value threshold

(d) 536 left after testing ratio of principal curvatures

3. Orientation Assignment

— 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)