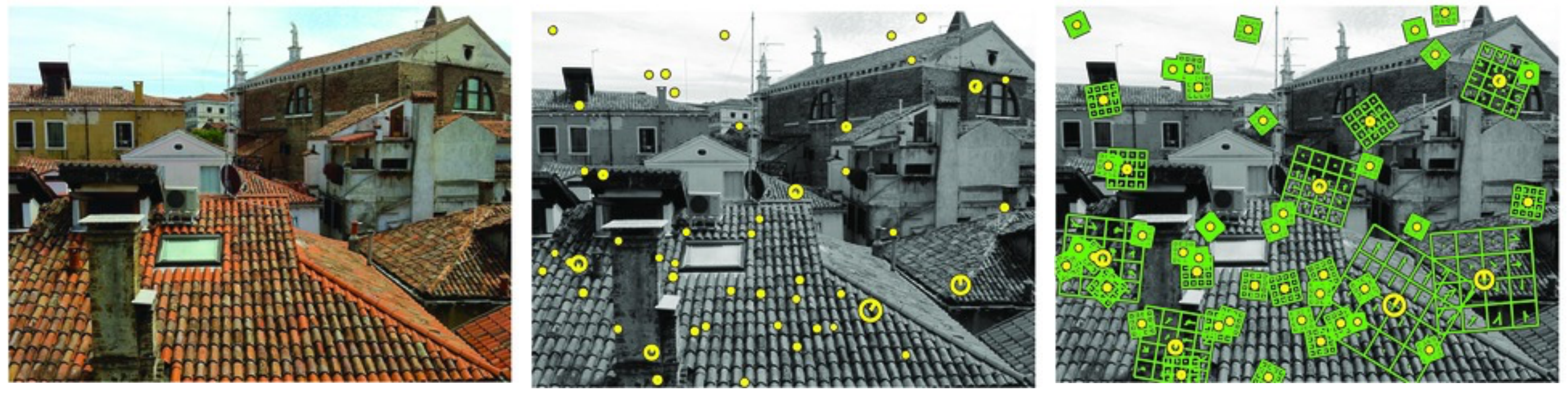
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 18: Scale Invariant Features (SIFT)

Menu for Today (**October 23, 2020**)

Topics:

— Scale Invariant Feature Transform (SIFT)

— SIFT detector, descriptor

Readings:

— **Today's** Lecture: Forsyth & Ponce (2nd ed.) 5.4
"Distinctive Image Features for Scale-Invariant Keypoints

— **Next** Lecture: Forsyth & Ponce (2nd ed.) 10.4.2, 10.1, 10.2

Reminders:

— **Midterm**: last class (we will start grading this weekend)

— **Assignment 3**: Texture Synthesis is due on **October 26th @ 11:59pm**

— **Schedule** for the course

Today's "**fun**" Example: Recognizing Panoramas

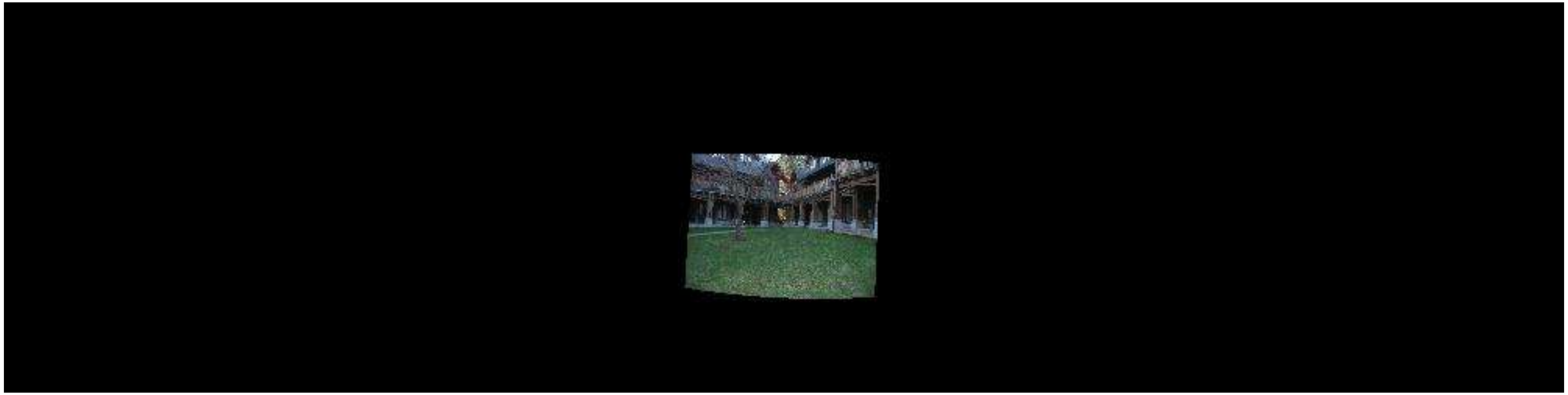


Figure Credit: Matthew Brown and David Lowe

Today's "**fun**" Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe

Today's "**fun**" Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe

Today's "**fun**" Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe

Today's "**fun**" Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe

Today's "**fun**" Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe

Today's "**fun**" Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe

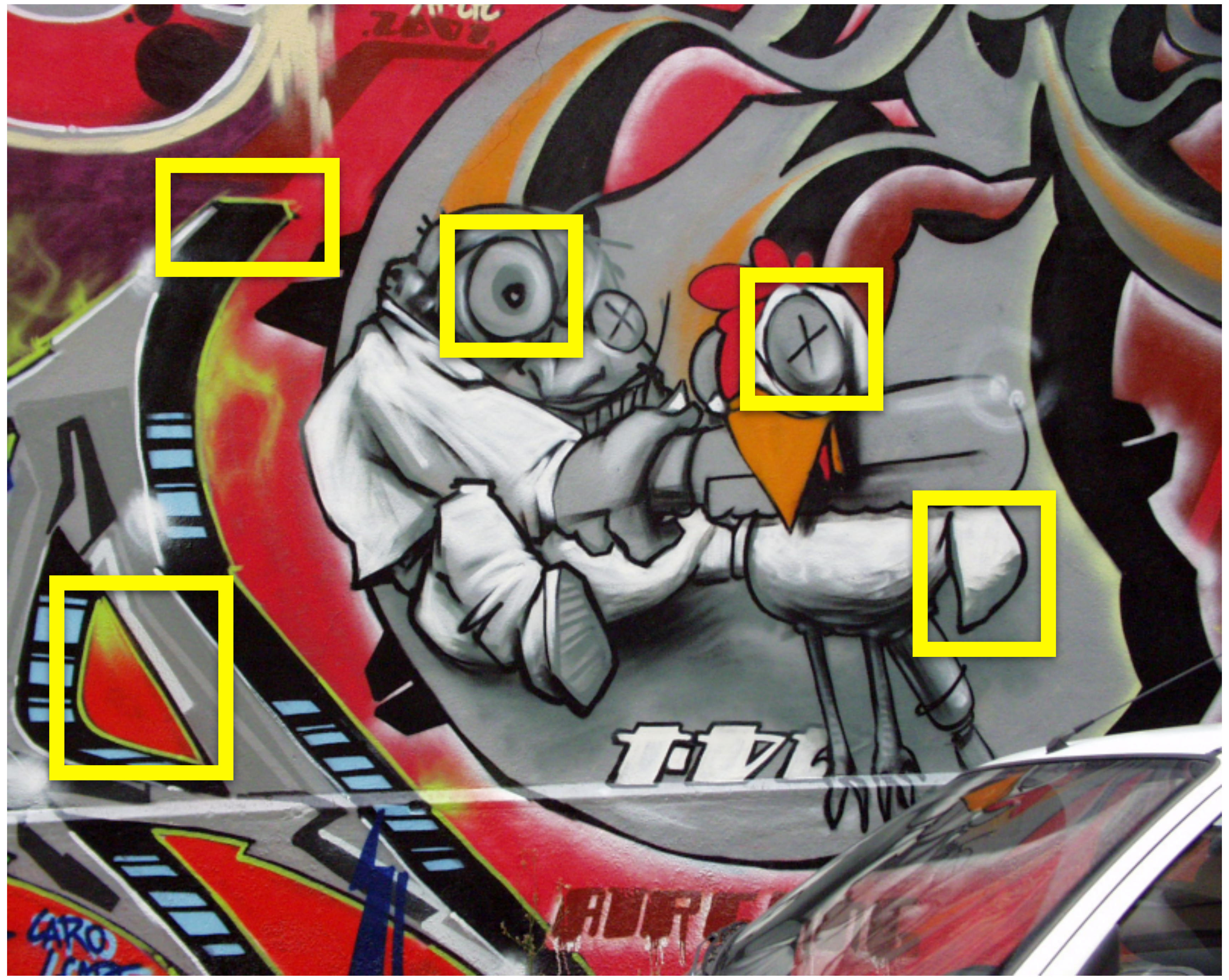
Today's "**fun**" Example: Recognizing Panoramas

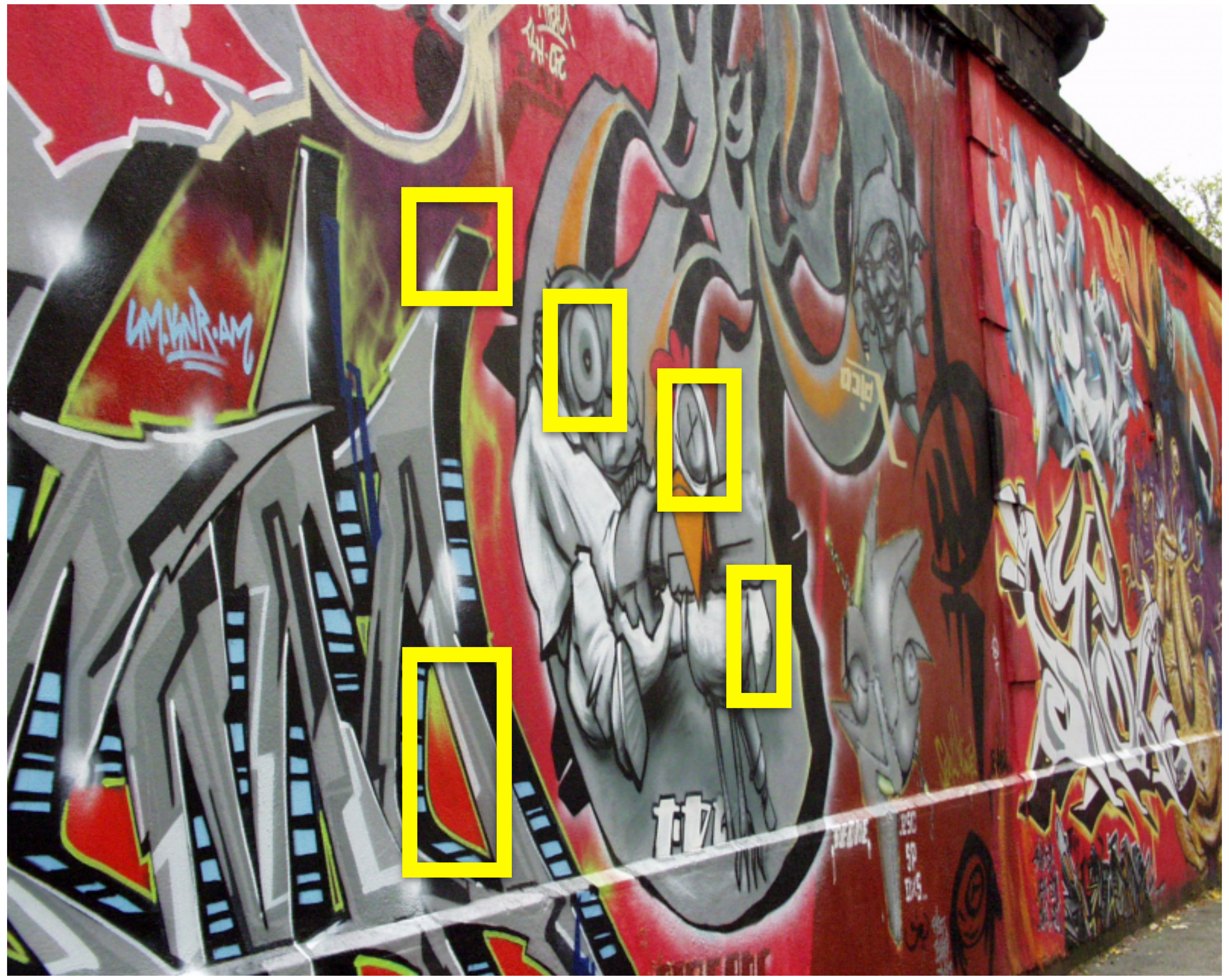
Figure Credit: Matthew Brown and David Lowe

Today's "**fun**" Example: Recognizing Panoramas

Figure Credit: Matthew Brown and David Lowe

Back to **Good Local Features**





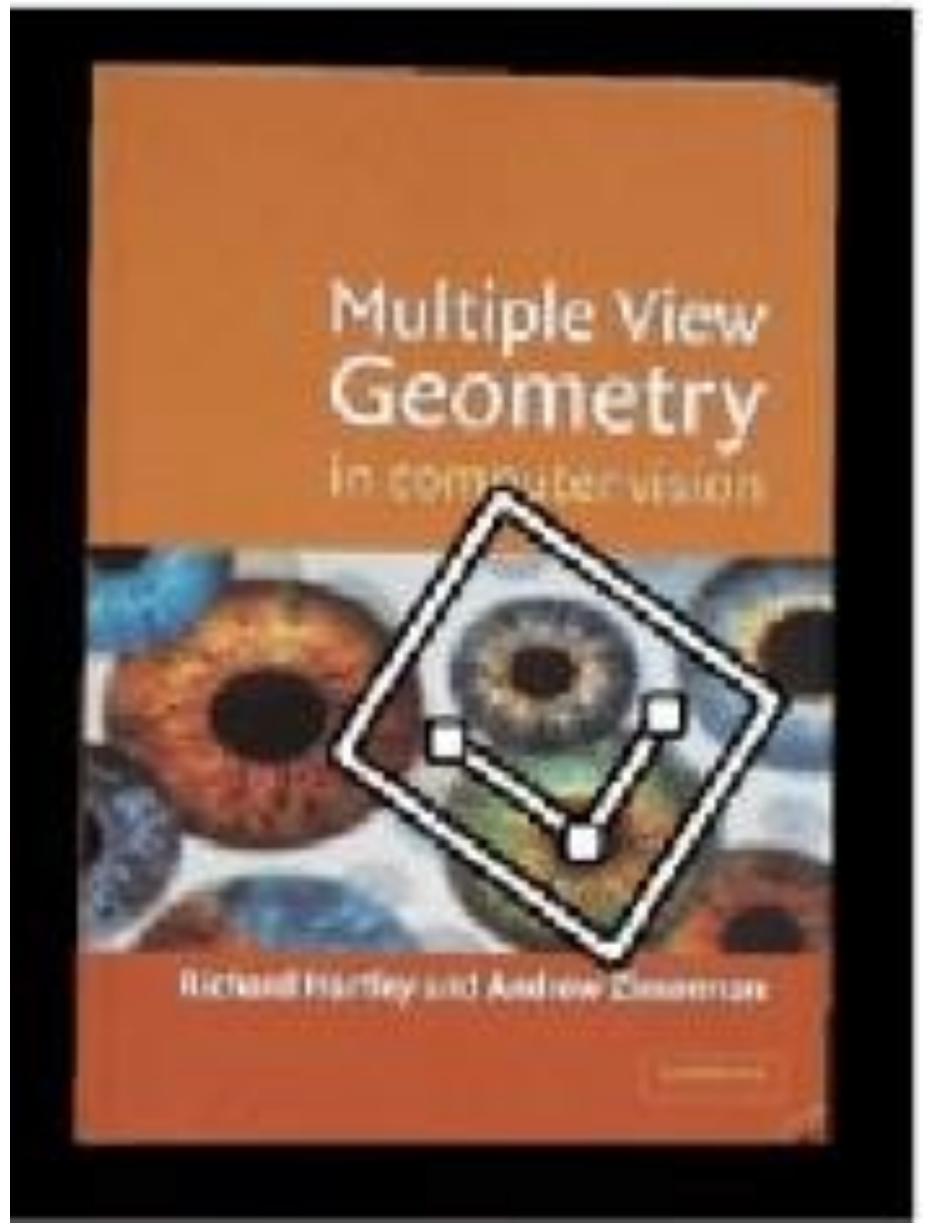
Where are the good features, and
how do we match them?

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?

Back to **Good Local Features**

How do we know which **corner** goes with which?

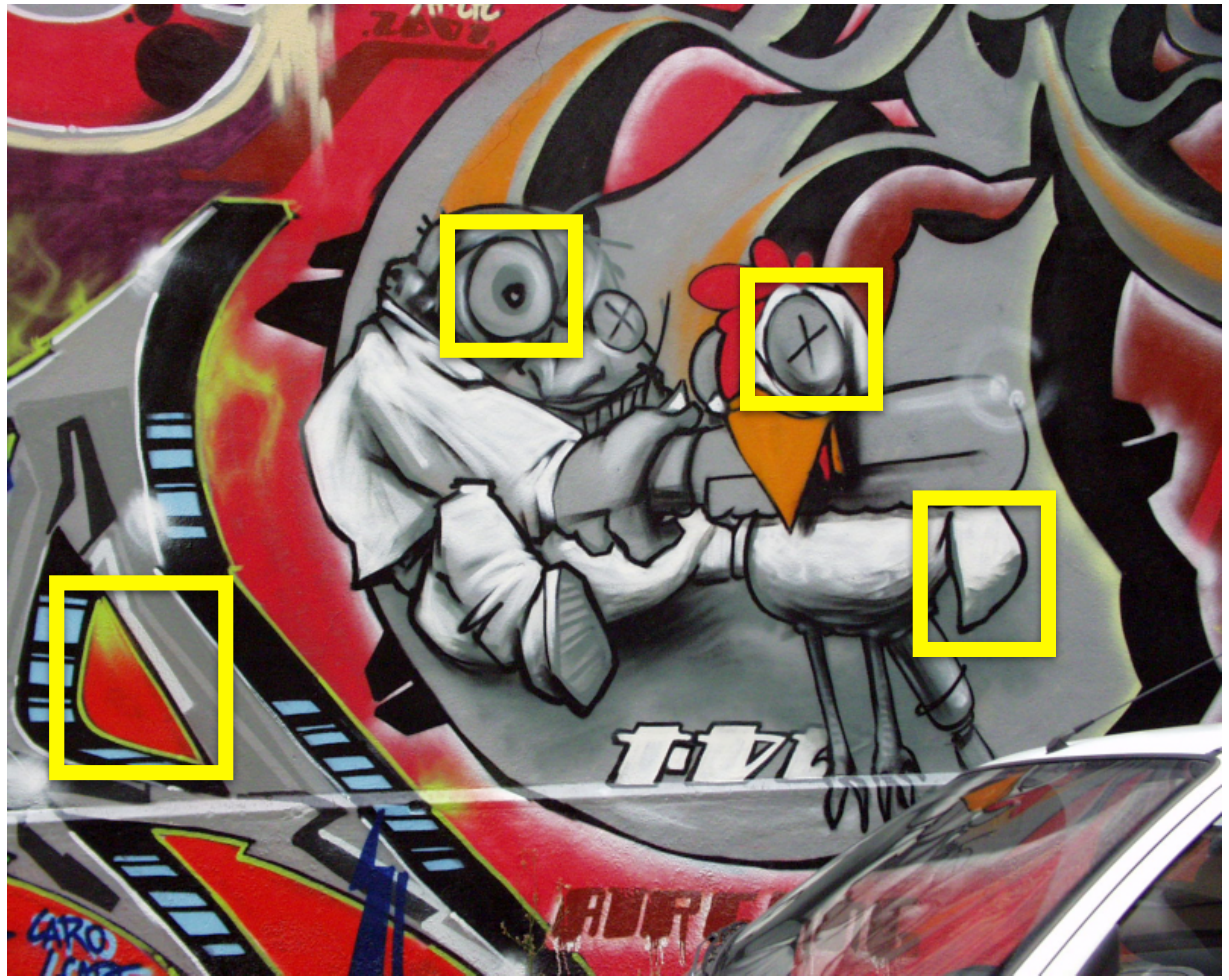
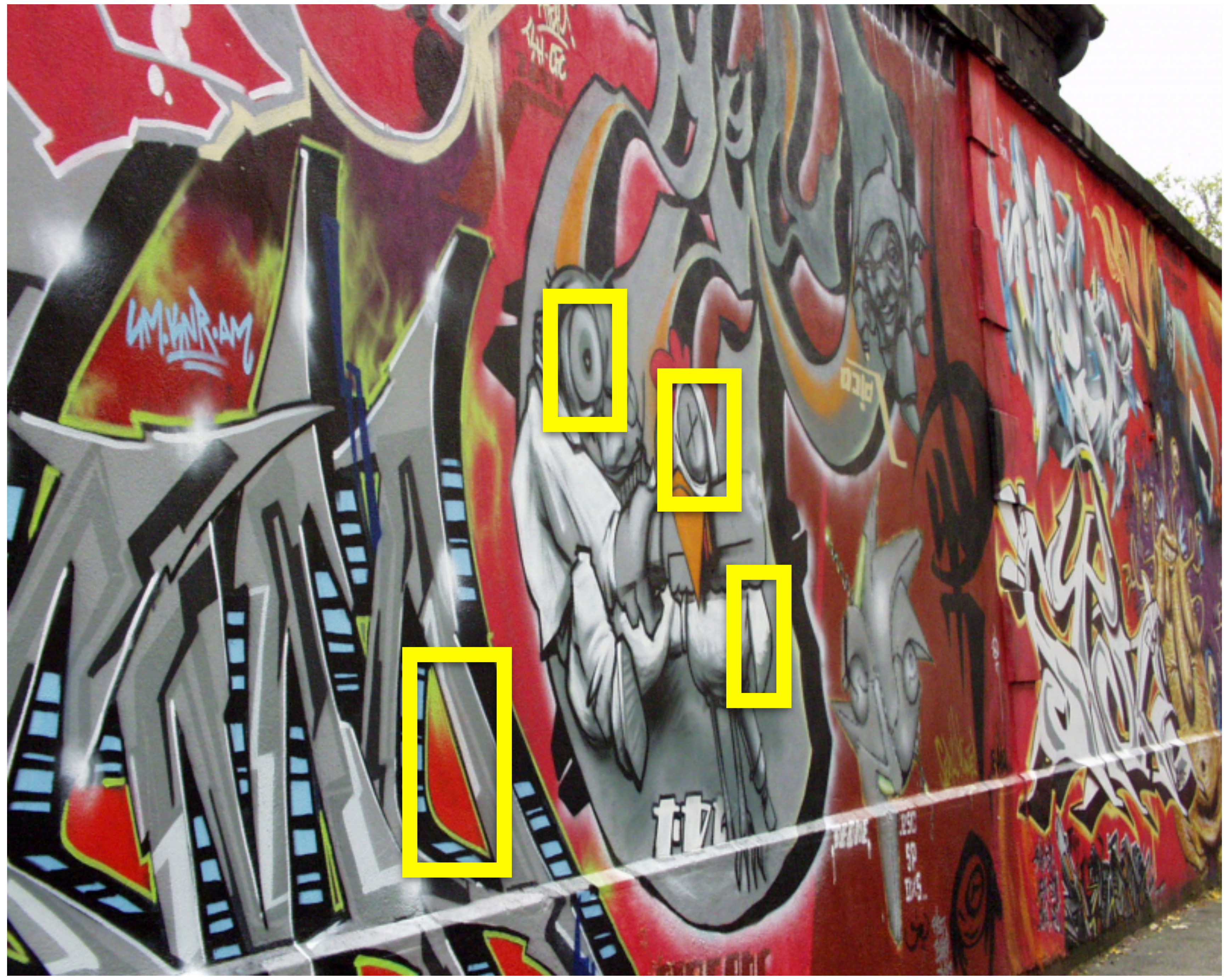
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Back to **Good Local Features**

How do we know which **blob** goes with which?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Back to **Good Local Features**

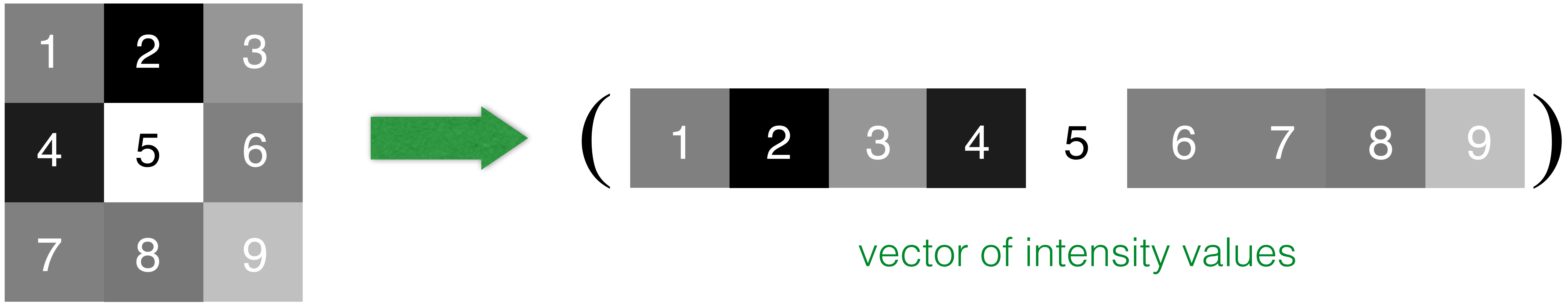
 

Patch around the local feature is very informative

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Intensity Image

Just use the pixel values of the patch



Perfectly fine if geometry and appearance is unchanged

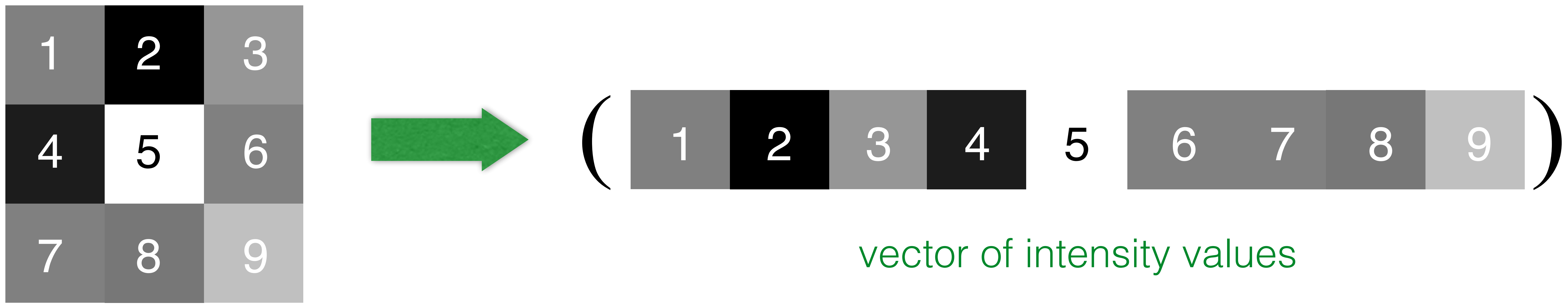
(a.k.a. template matching)

What are the problems?

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)

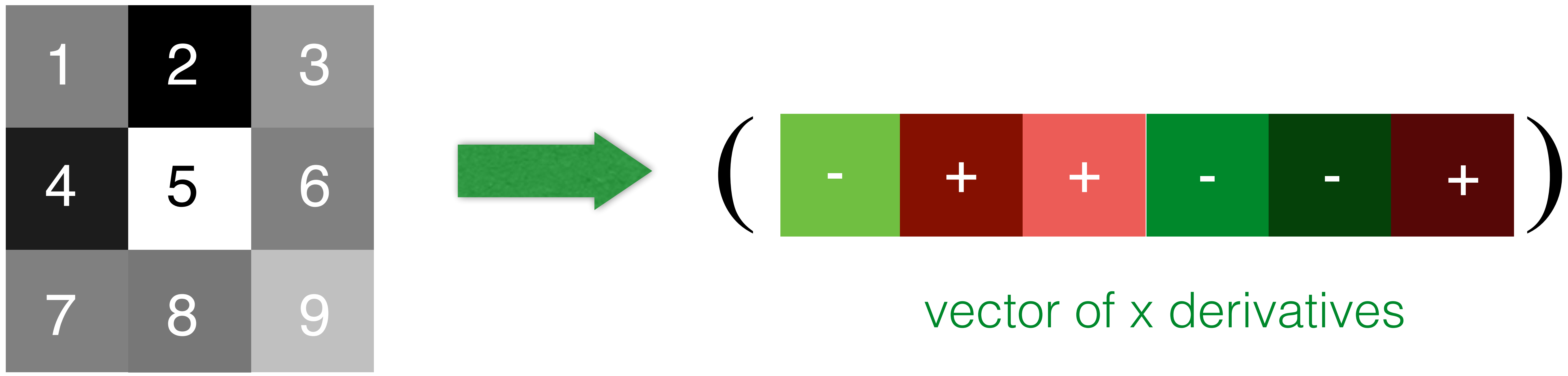
What are the problems?

How can you be less sensitive to absolute intensity values?

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

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

Use pixel differences



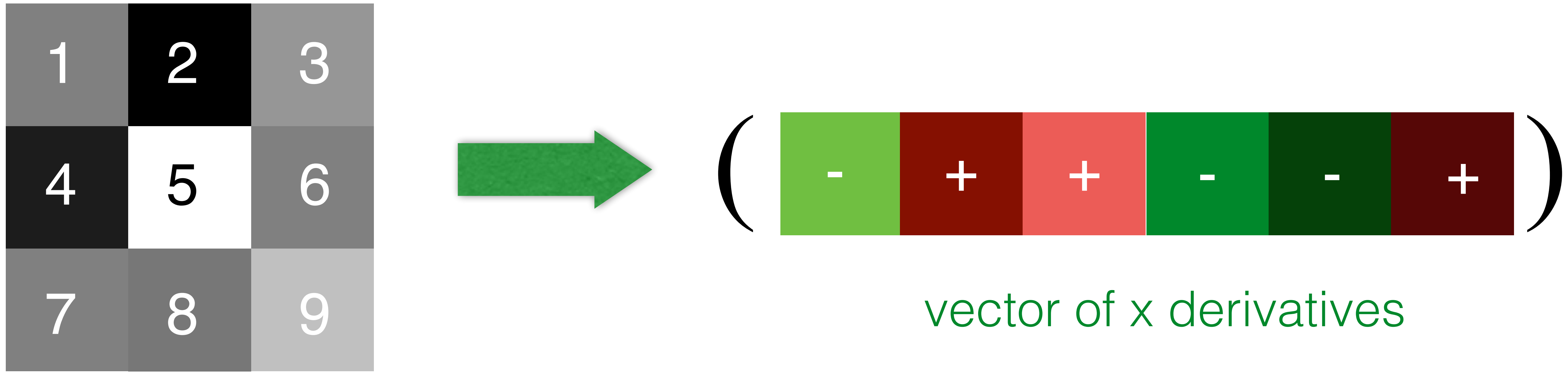
Feature is invariant to absolute intensity values

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	$\bigtriangledown^2 G$, Canny
“corner” / “blob”	sparse (0D)	locally distinct features	Harris, SIFT

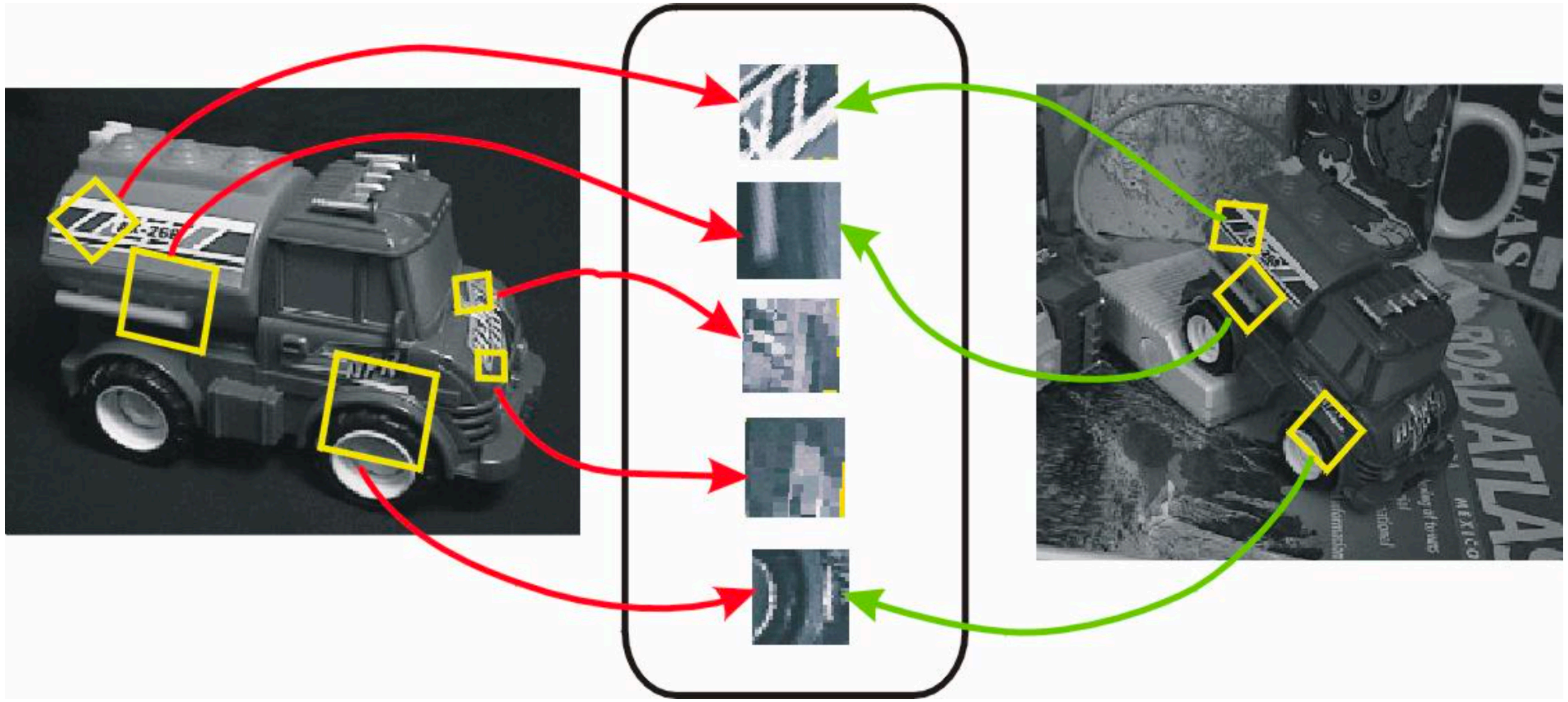
Object **Recognition** with Invariant Features

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

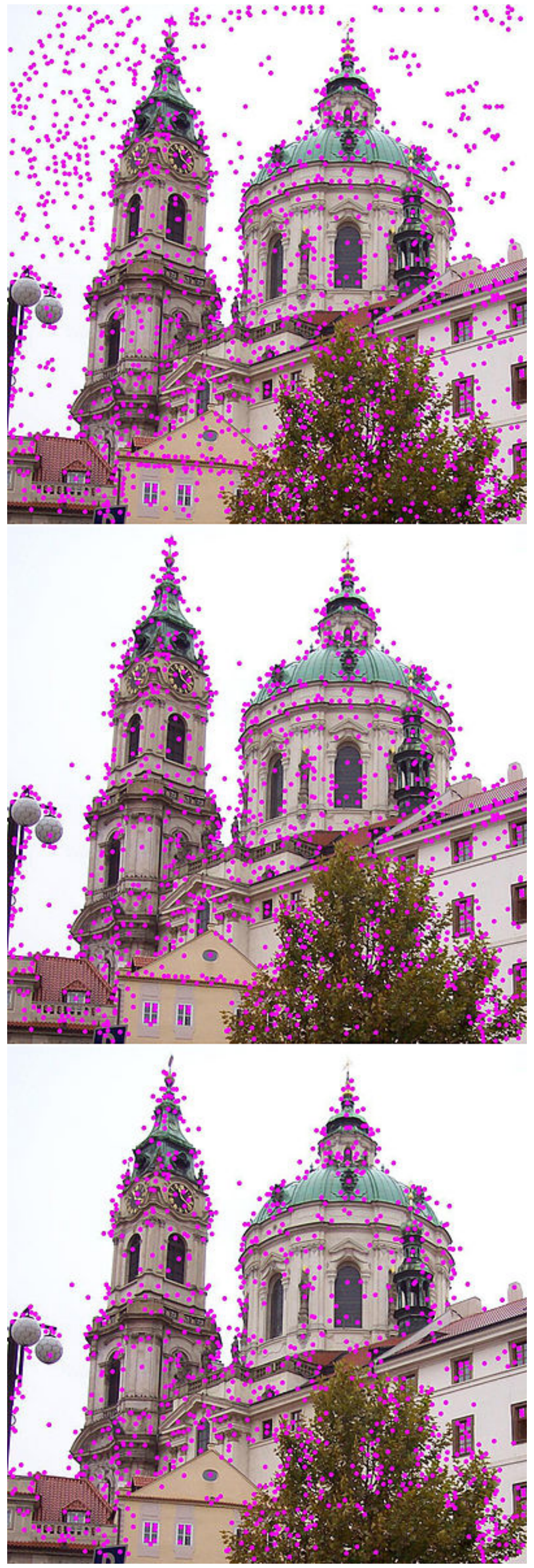
Locality: features are local, so robust to occlusion and clutter (no prior segmentation)

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**)

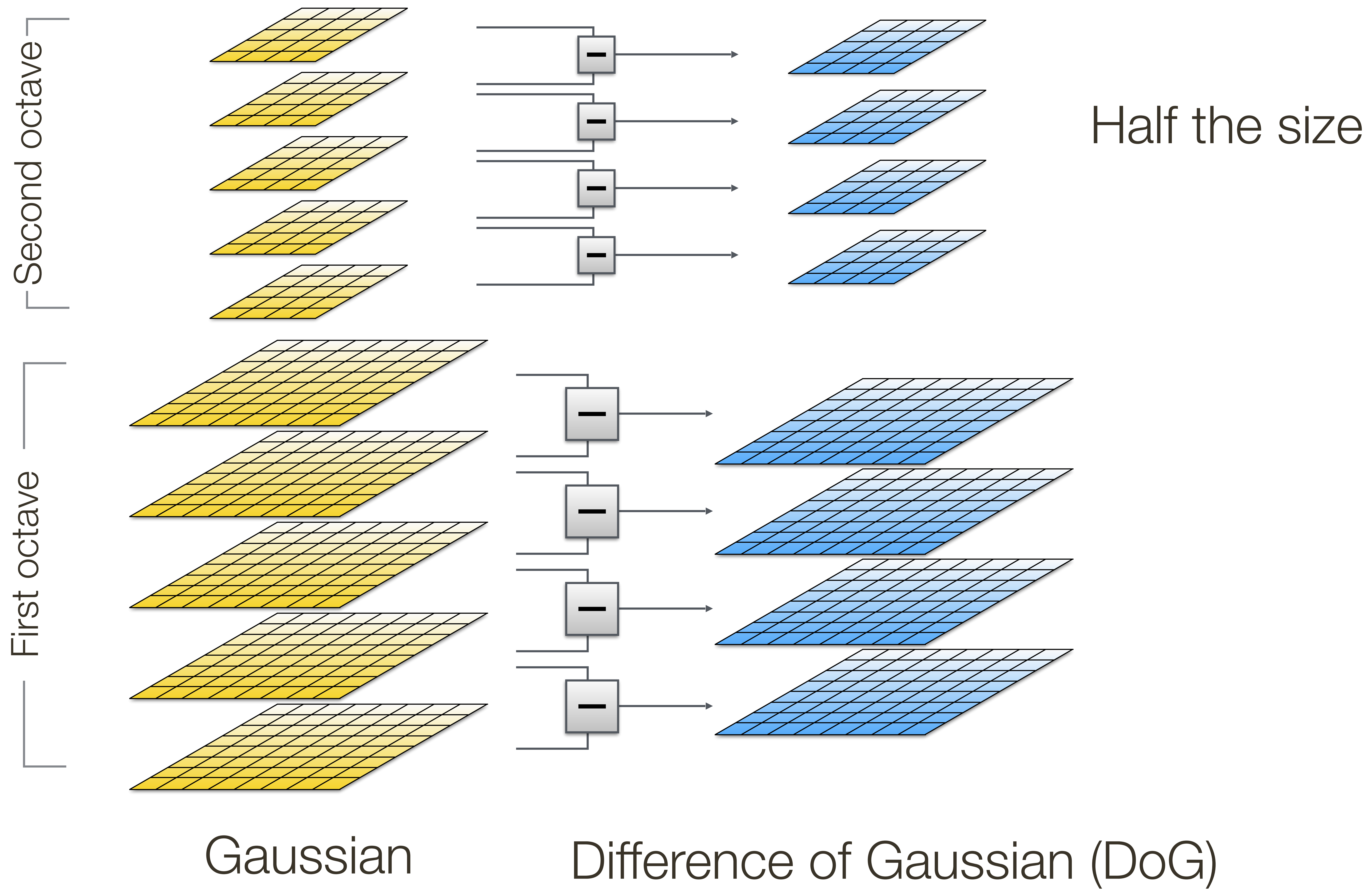


SIFT describes both a **detector** and **descriptor**

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

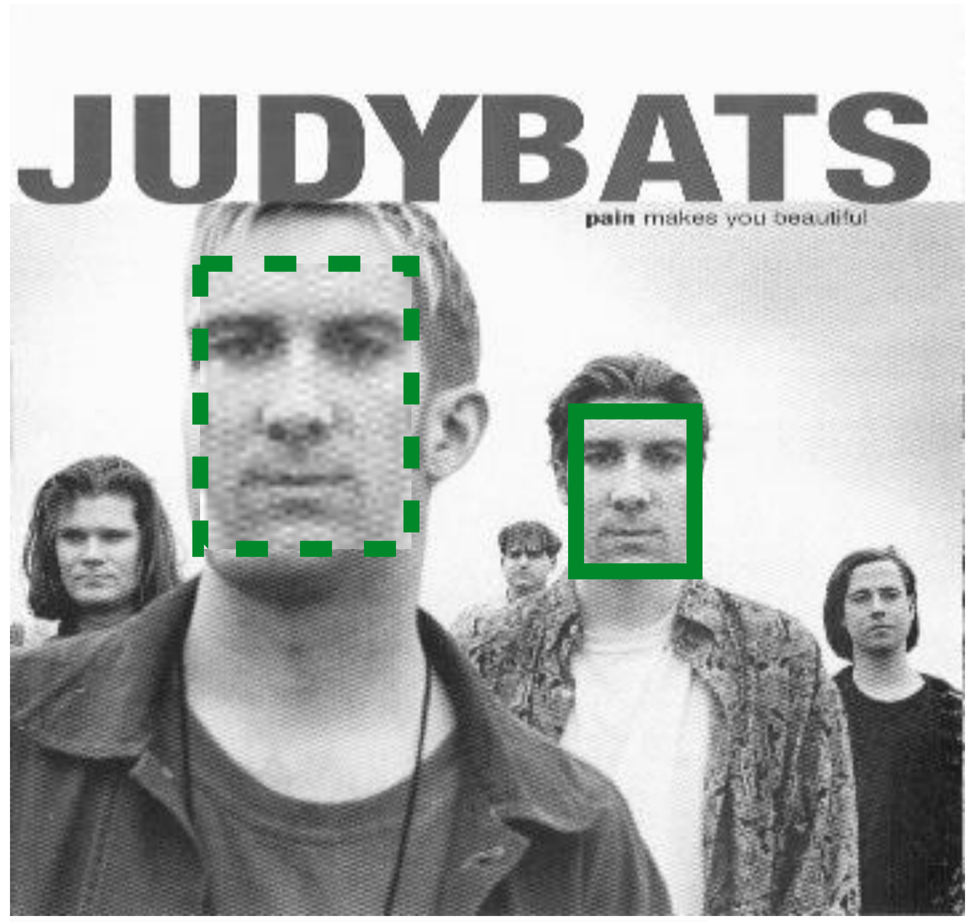


Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

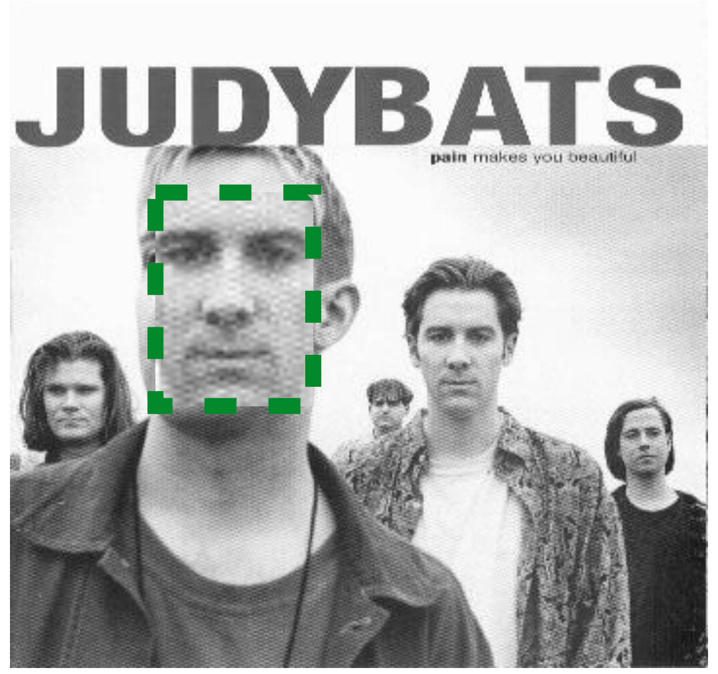
Recall: Template matching

Level | **Image** Pyramid (s) | **Template**

0

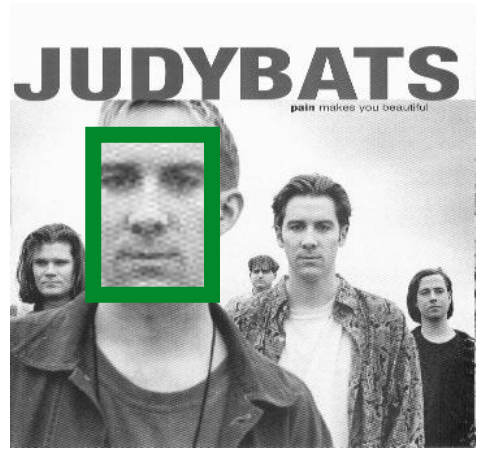


1



. . .

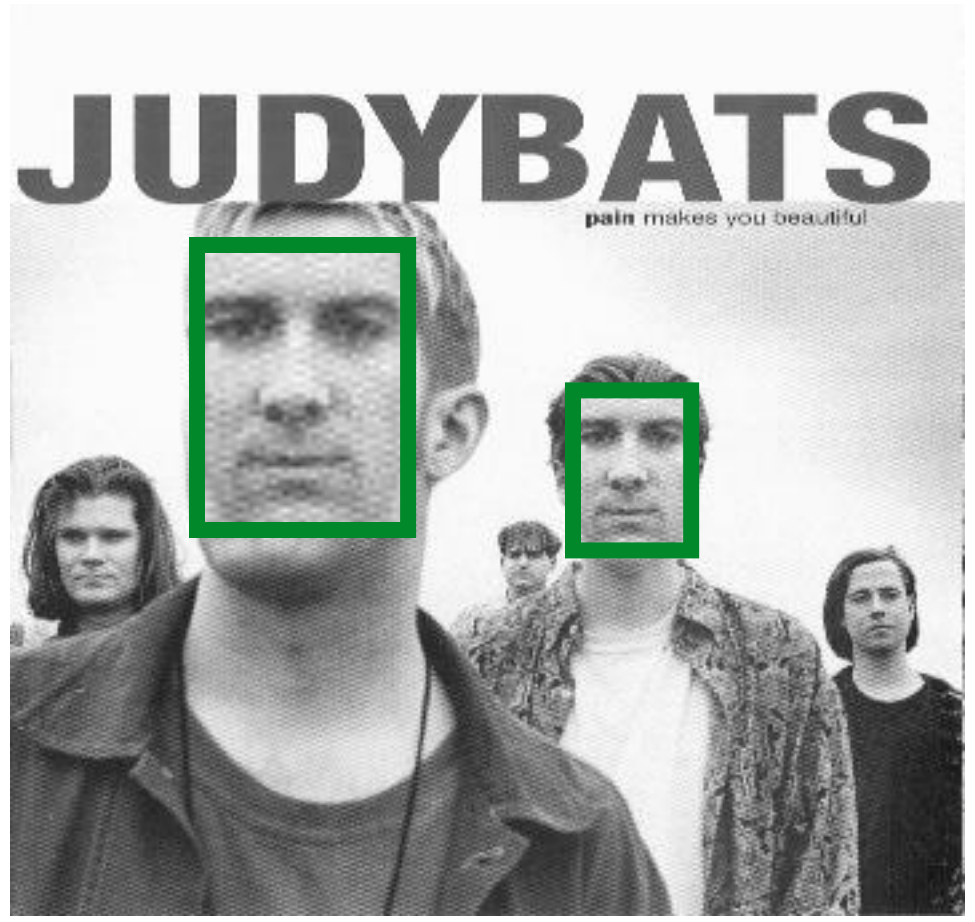
L



Template Pyramid (1/s)

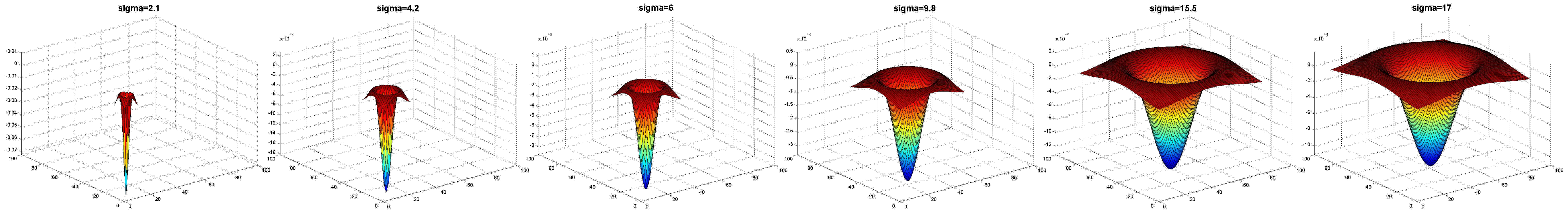
. . .

Image



Both allow **search over scale**

Recall: Applying **Laplacian** Filter at Different **Scales**



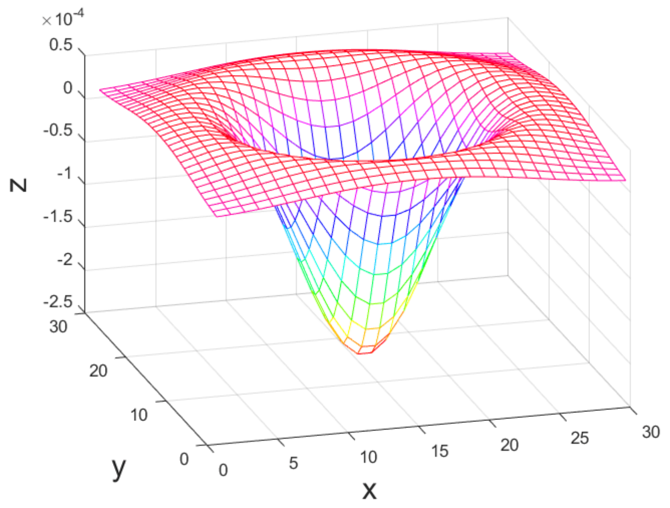
Full size

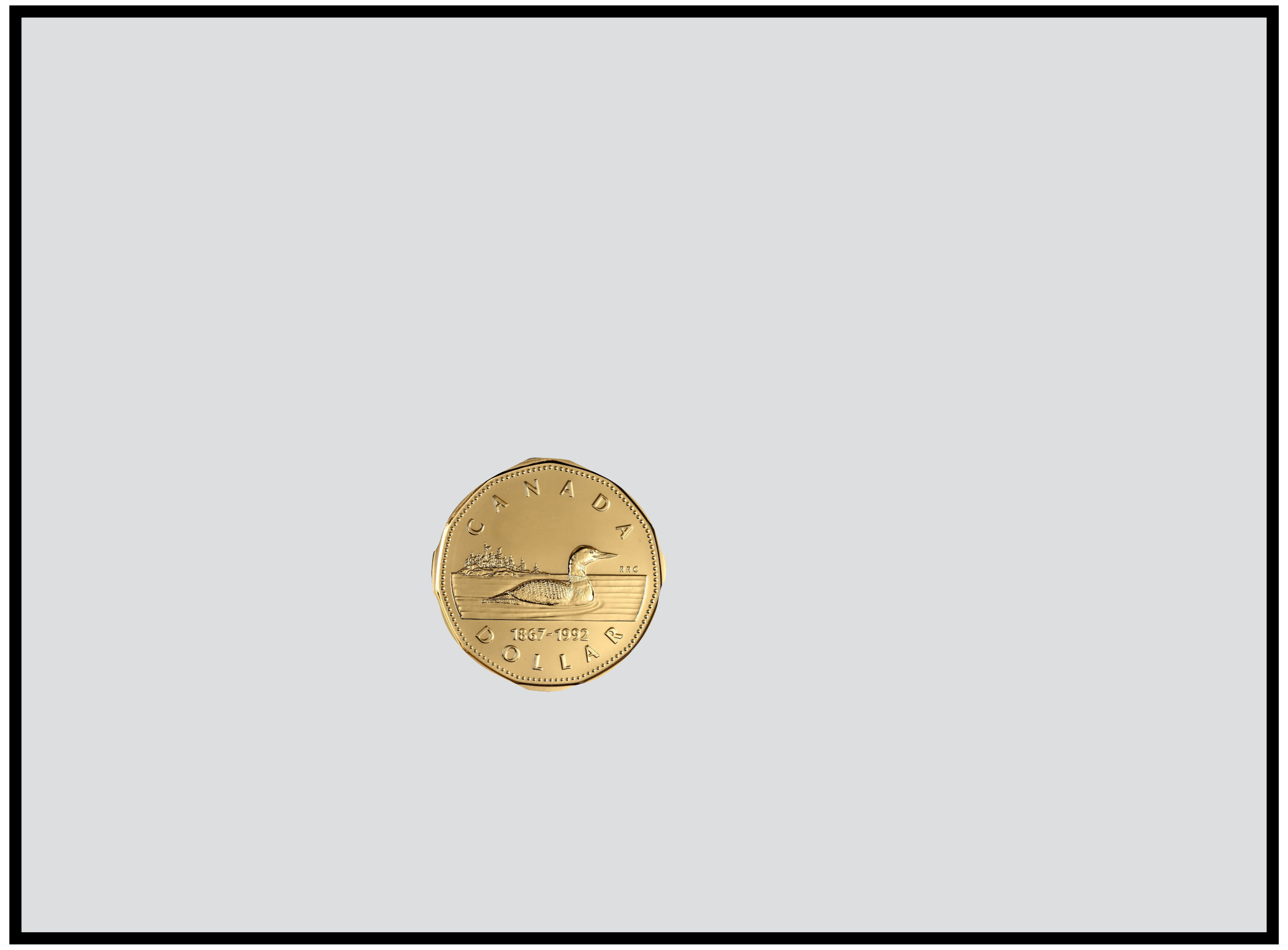
3/4 size

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Searching over **Scale**-space

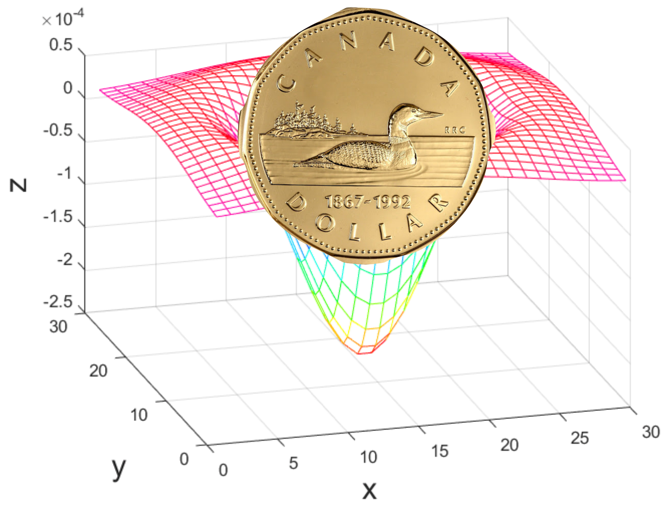
σ

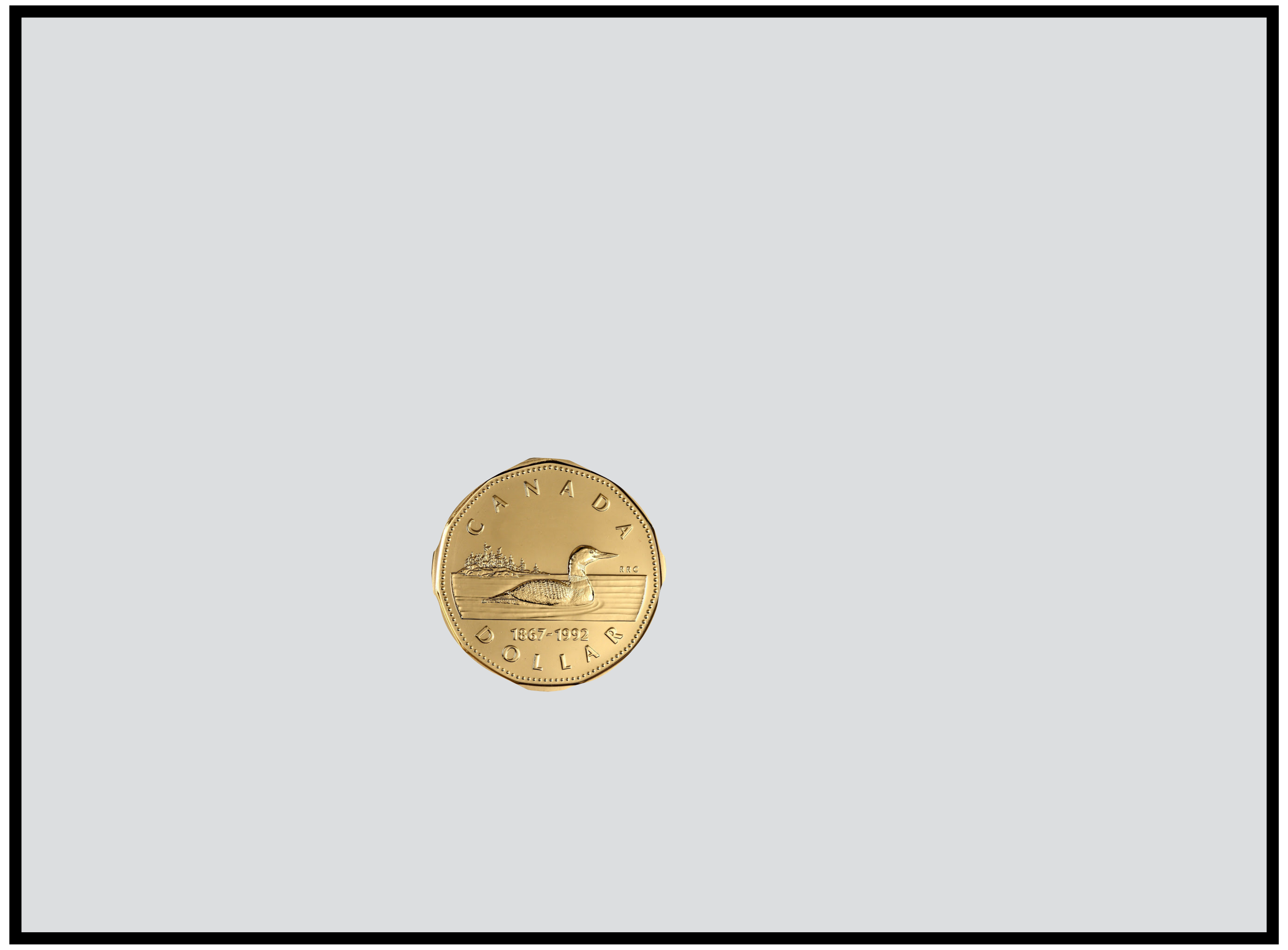




Searching over **Scale**-space

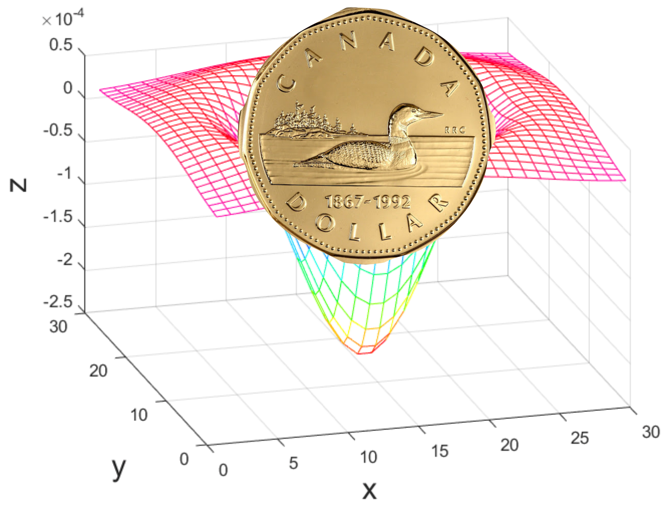
σ

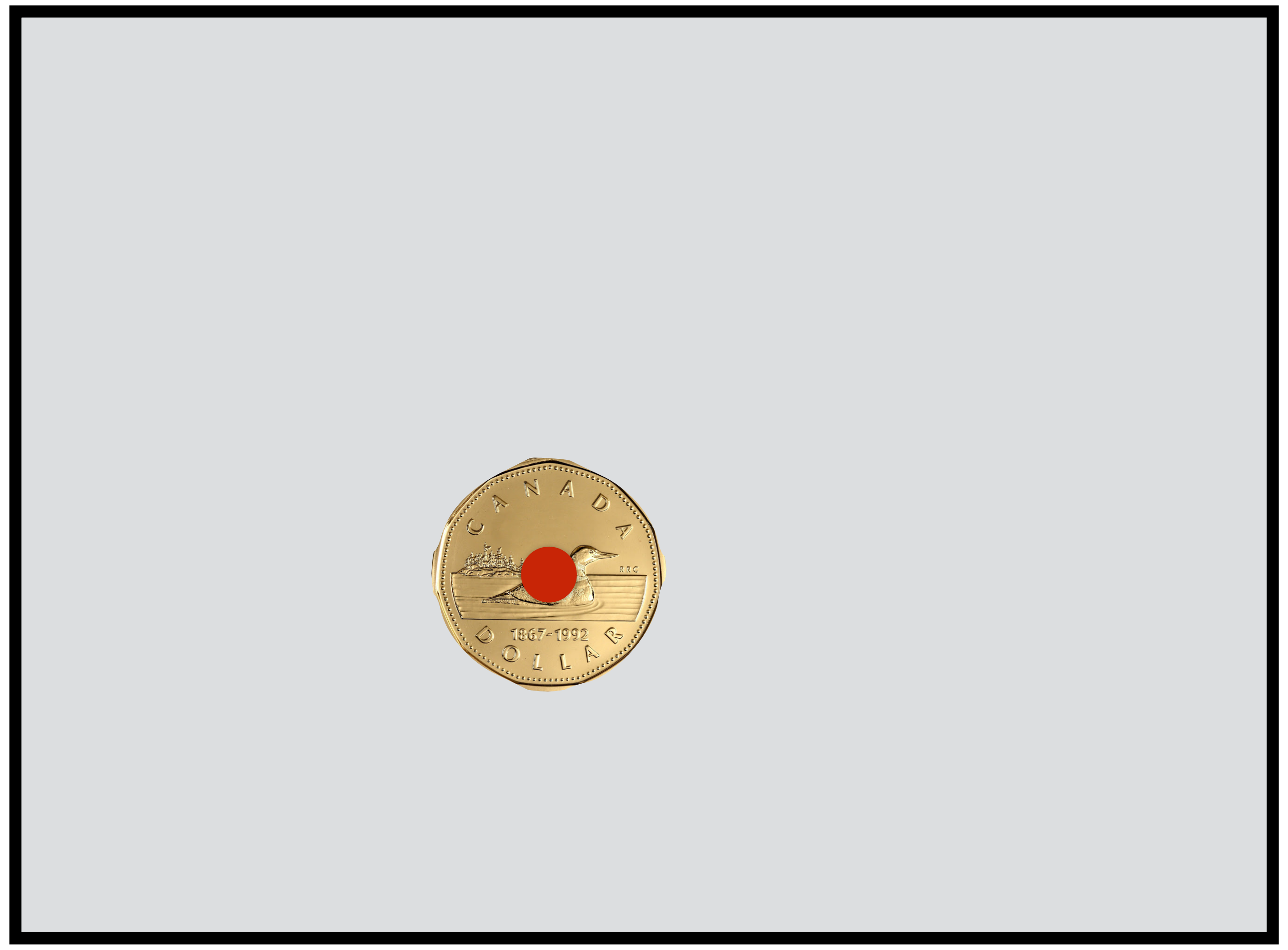




Searching over **Scale**-space

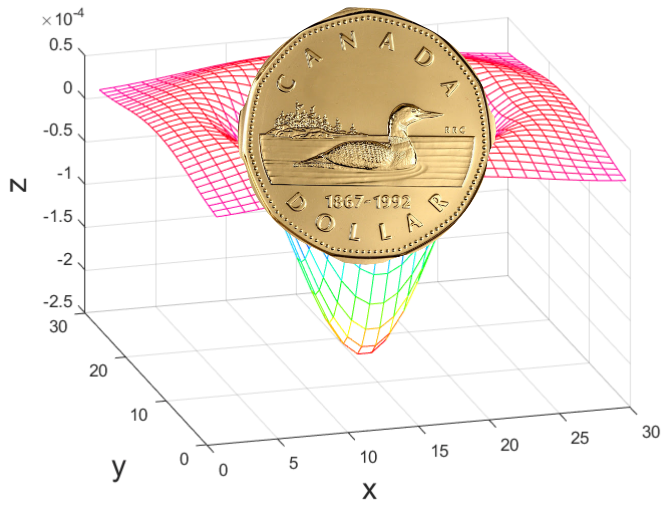
σ

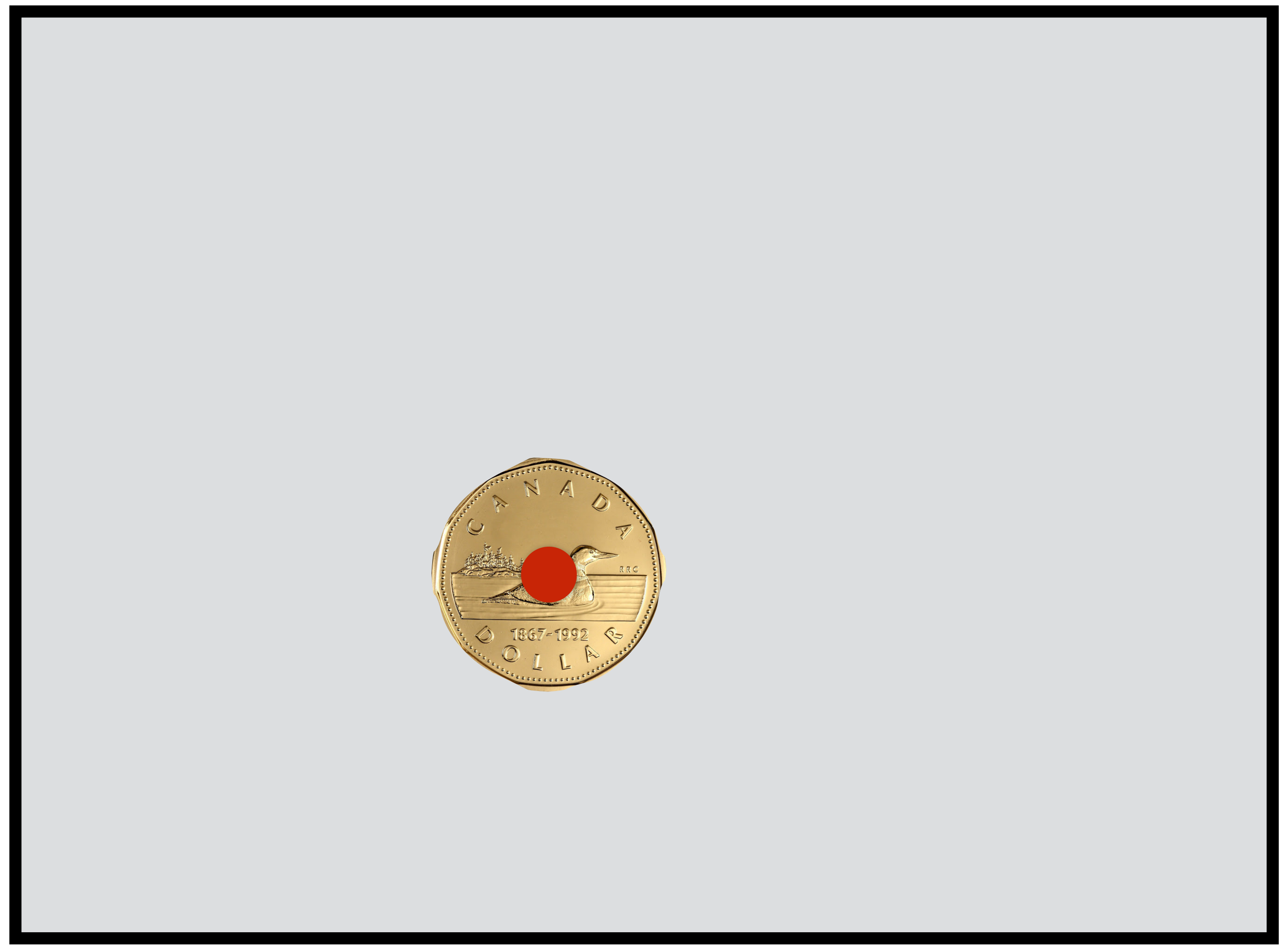




Searching over **Scale**-space

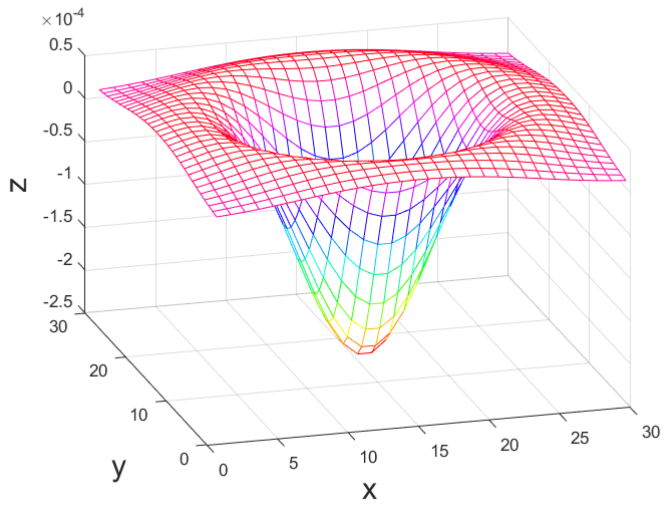
σ



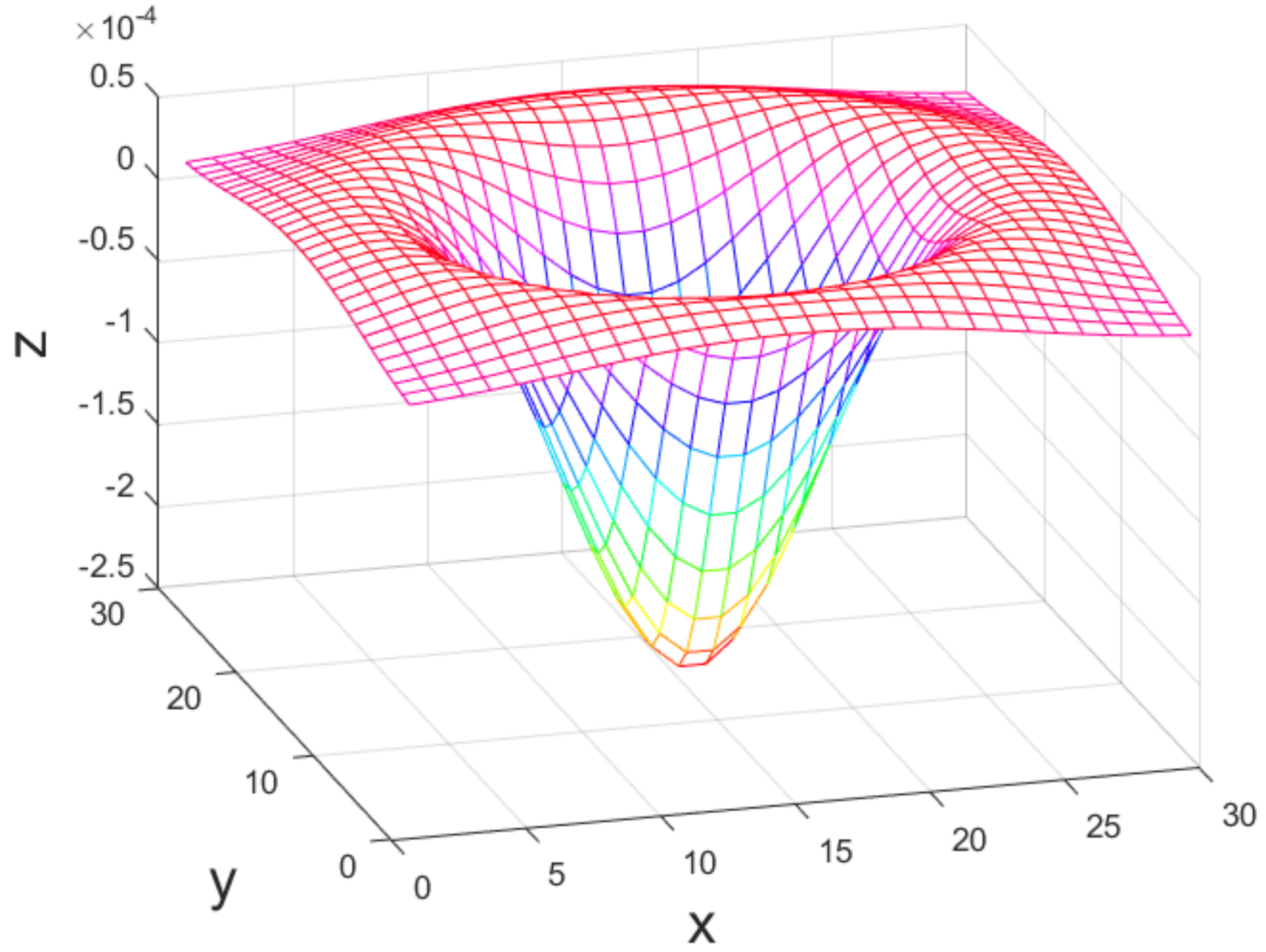


Searching over **Scale**-space

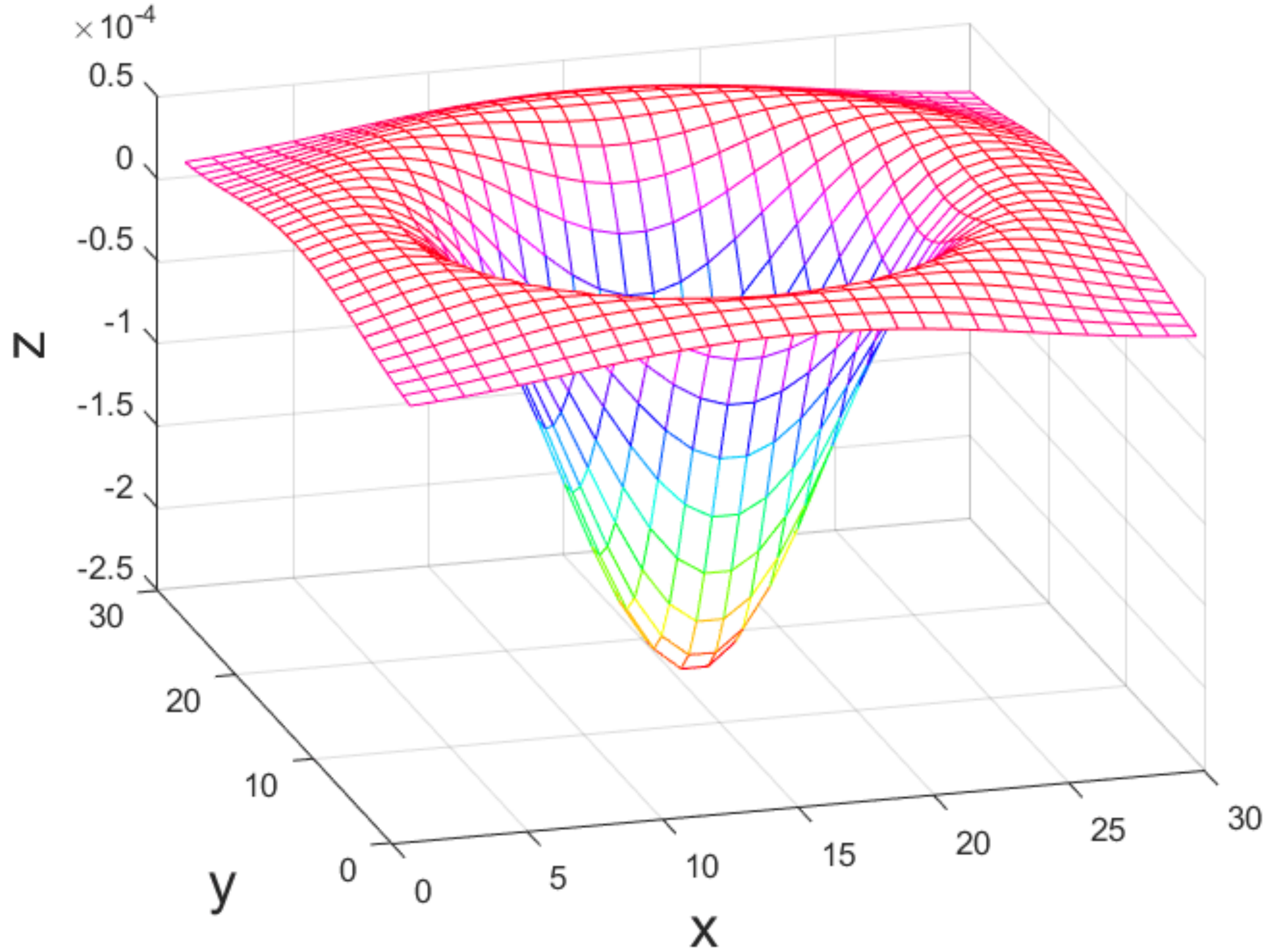
σ

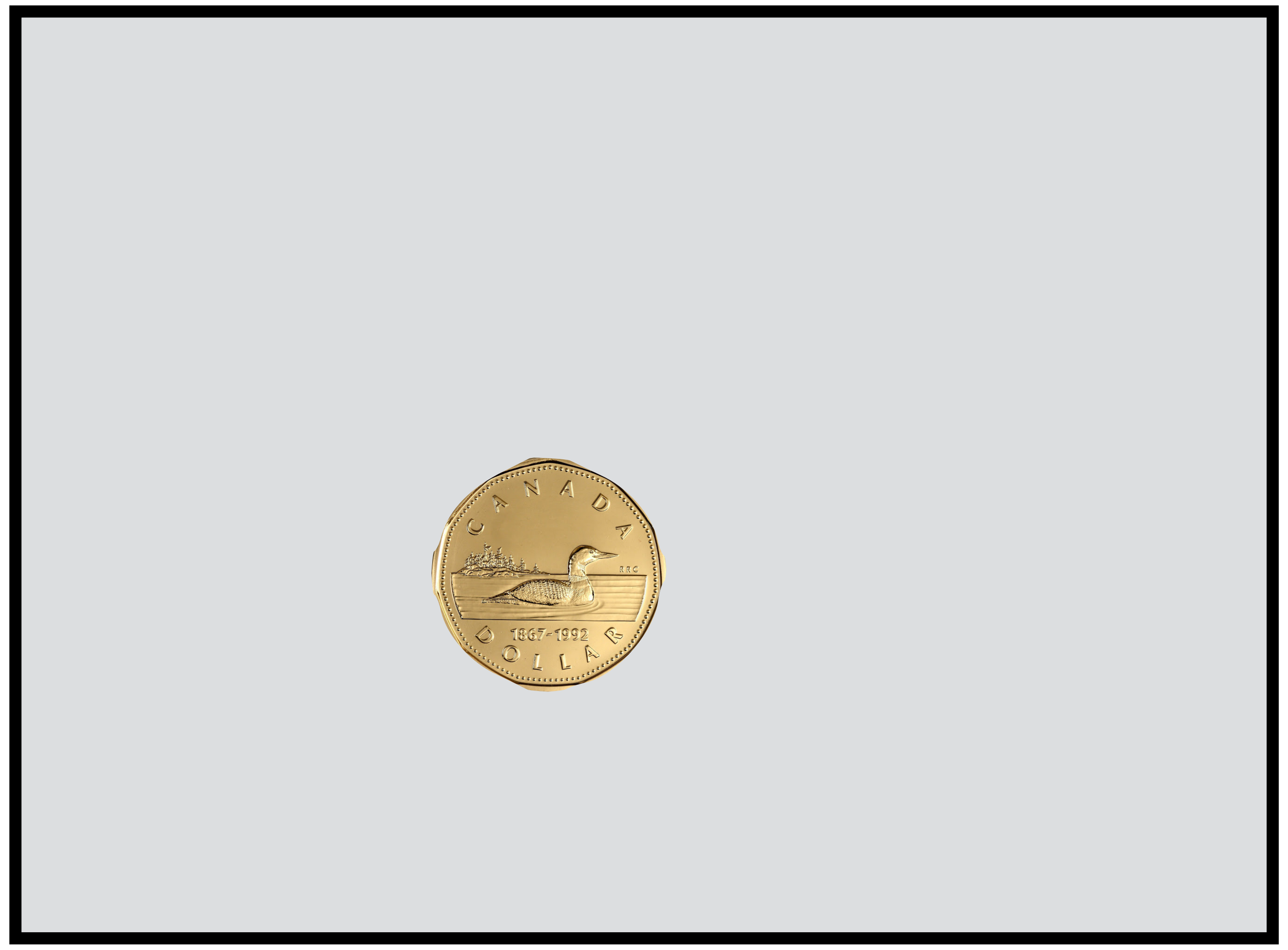


$\sigma' = 2\sigma$



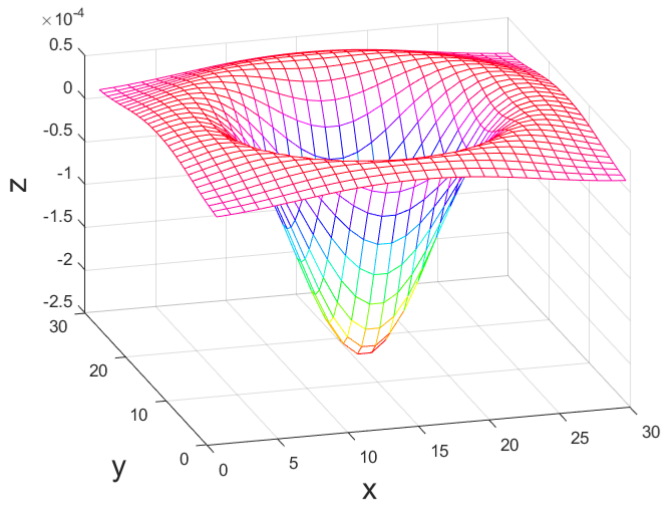
$\sigma' = 3\sigma$



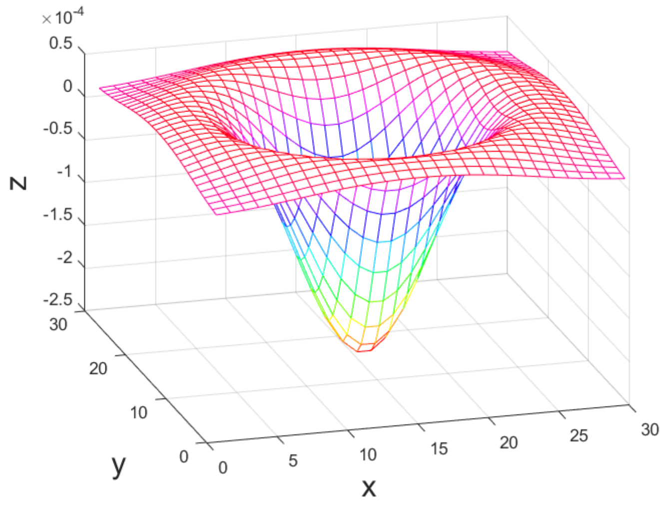


Searching over **Scale**-space

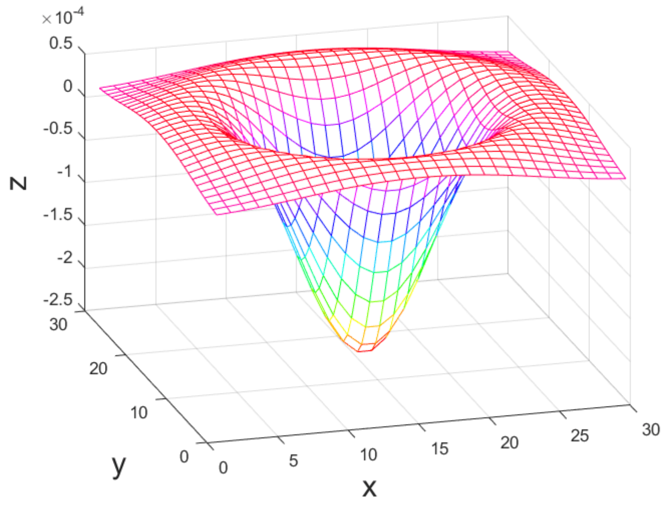
σ

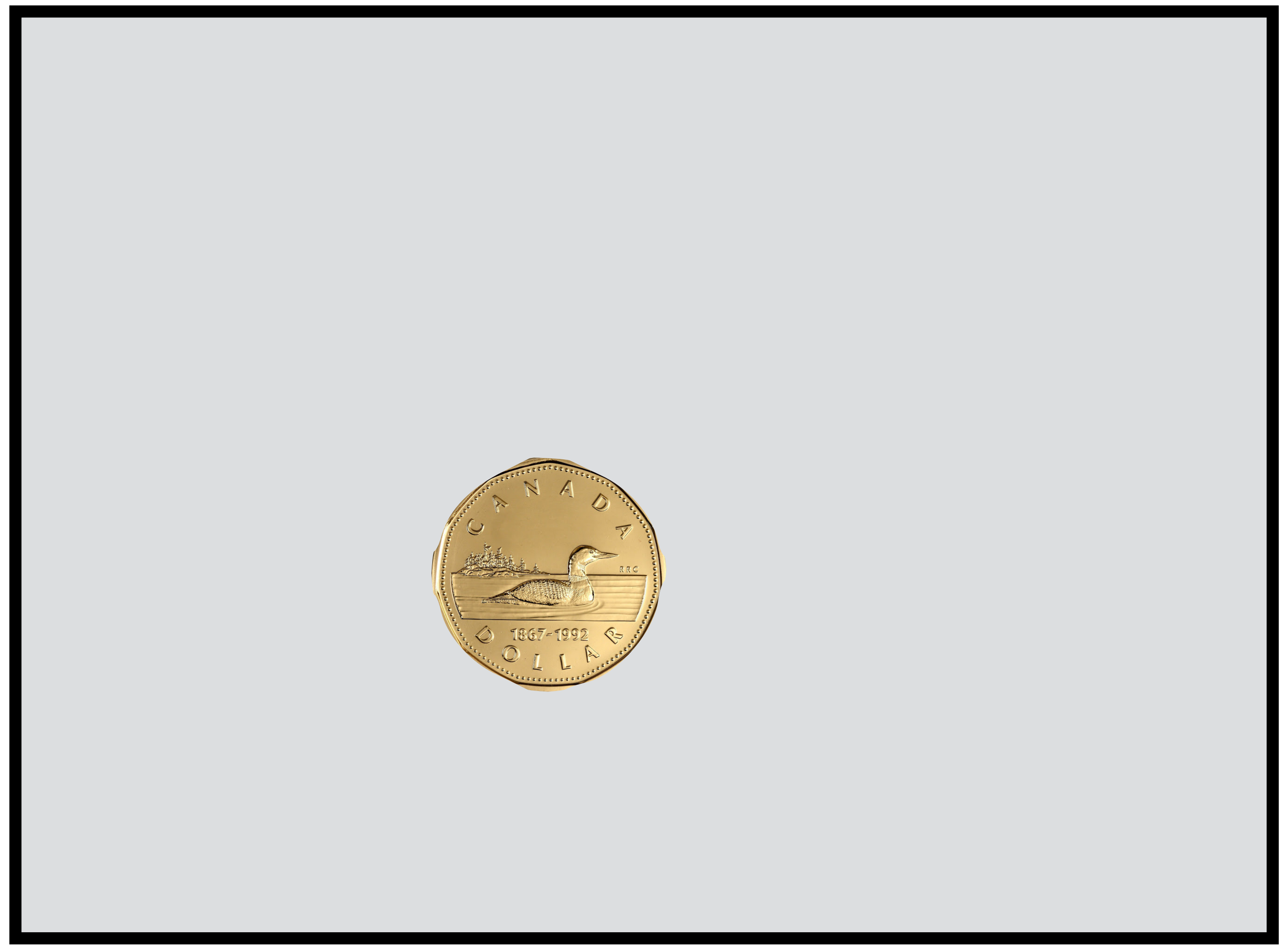


σ



σ

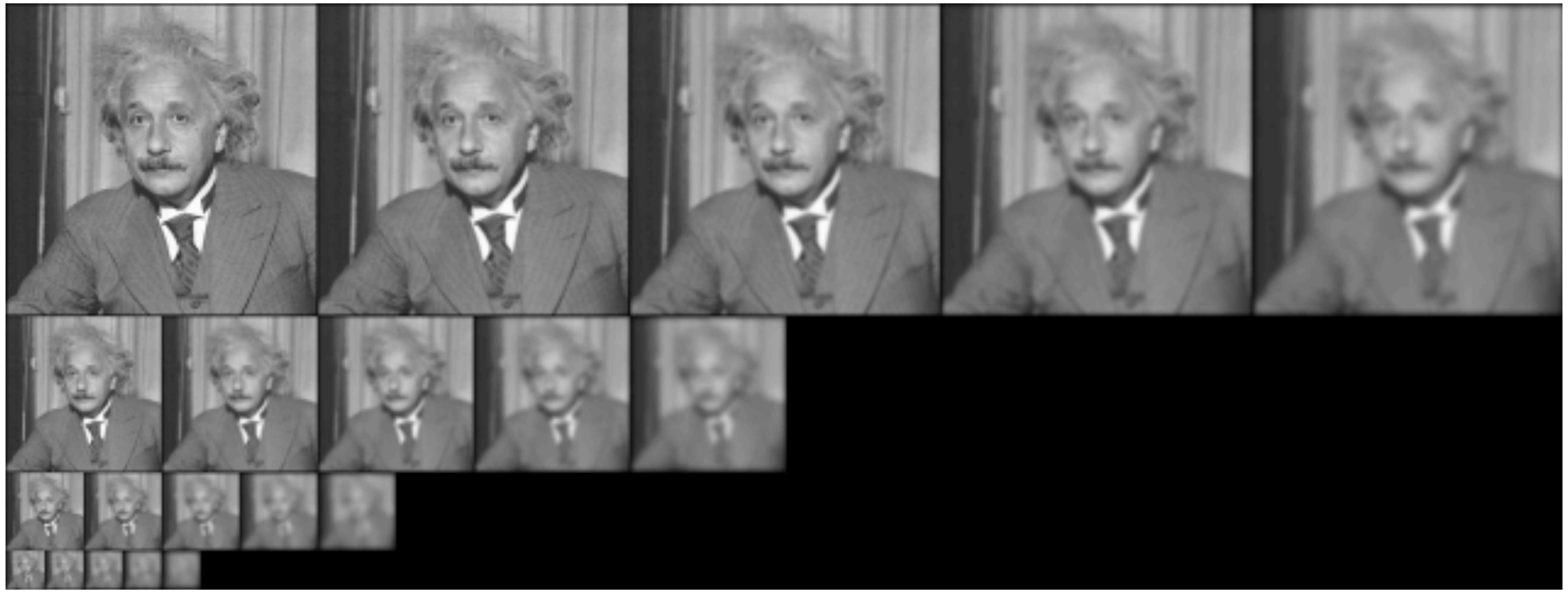




$s = 0.5$

$s = 0.33$

1. Multi-scale Extrema Detection



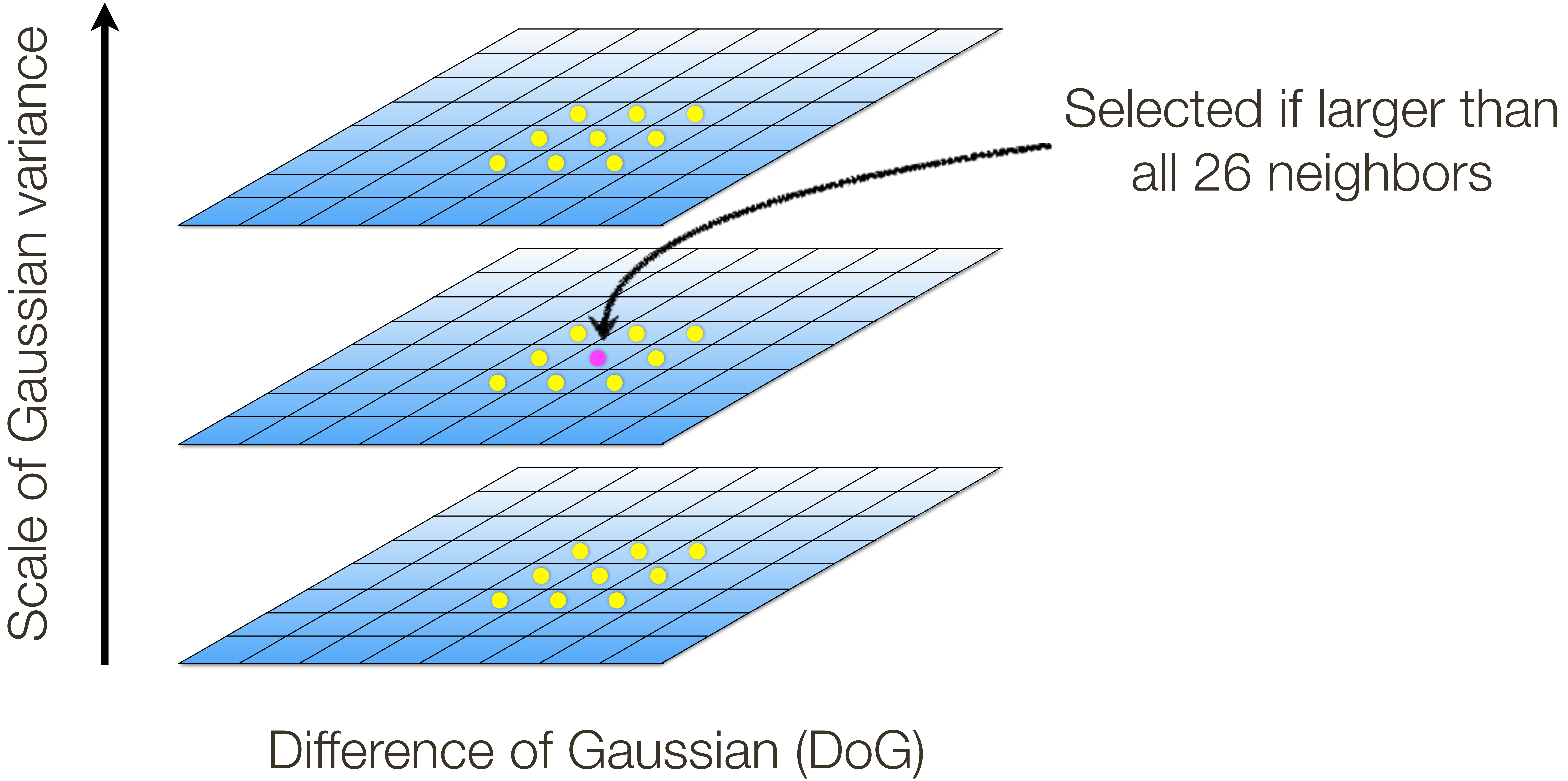
Gaussian

Laplacian

1. Multi-scale Extrema Detection

Detect maxima and minima of Difference of Gaussian in scale space

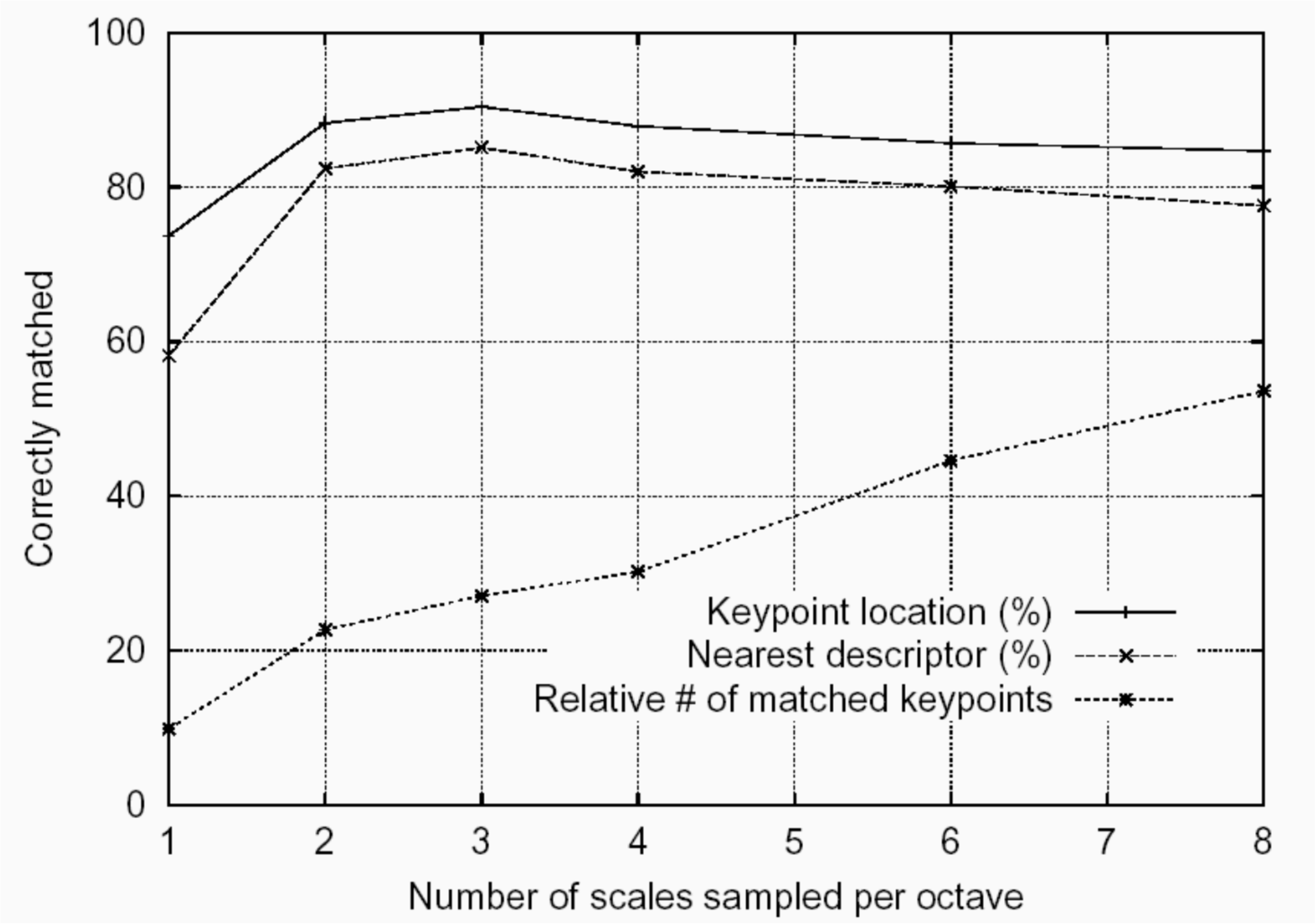
$$\sigma = 2^{1/s}$$



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?

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

2. Keypoint Localization

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

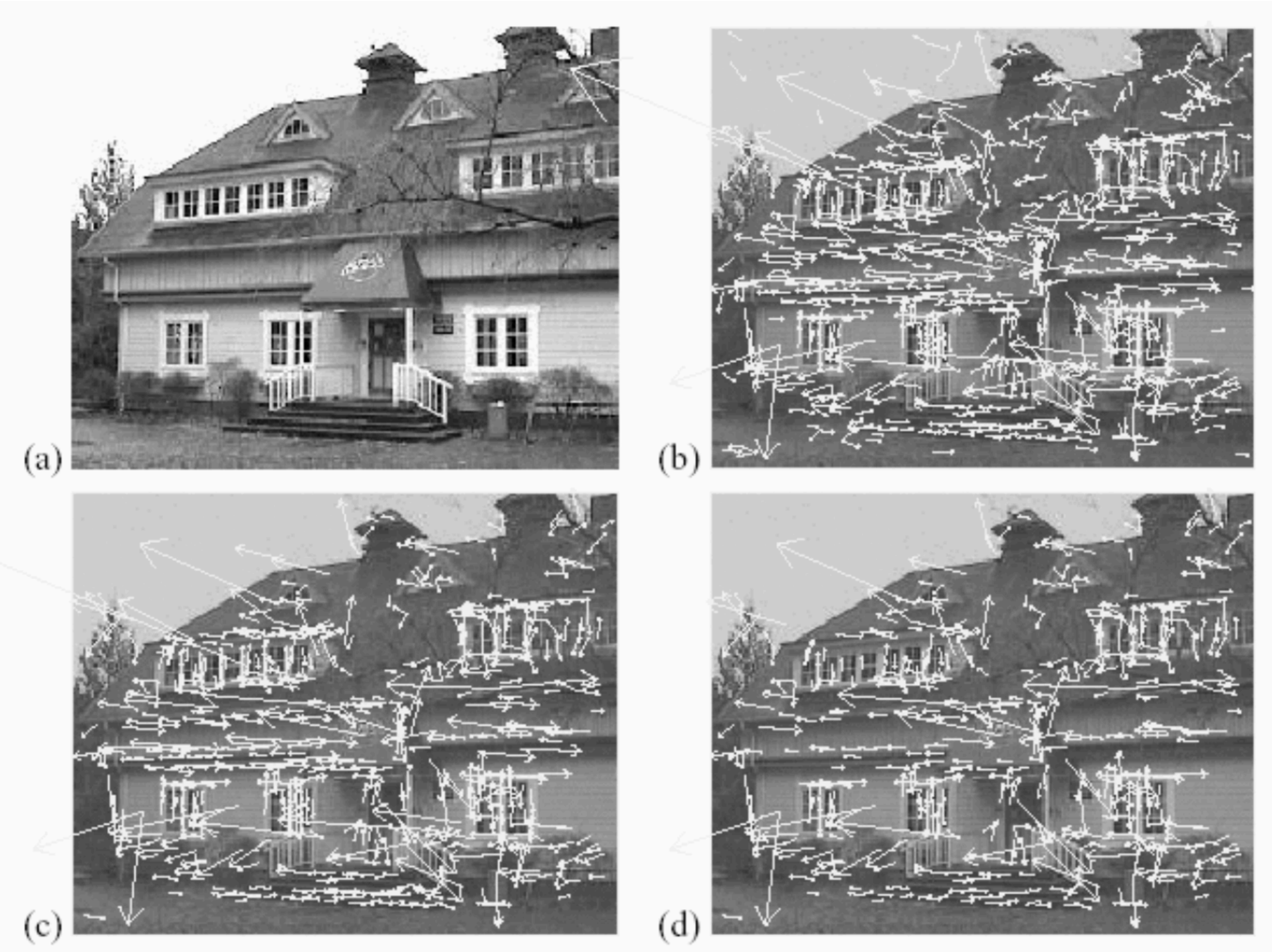
How do we decide whether a keypoint is poorly localized, say along an edge, vs. well-localized?

— 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

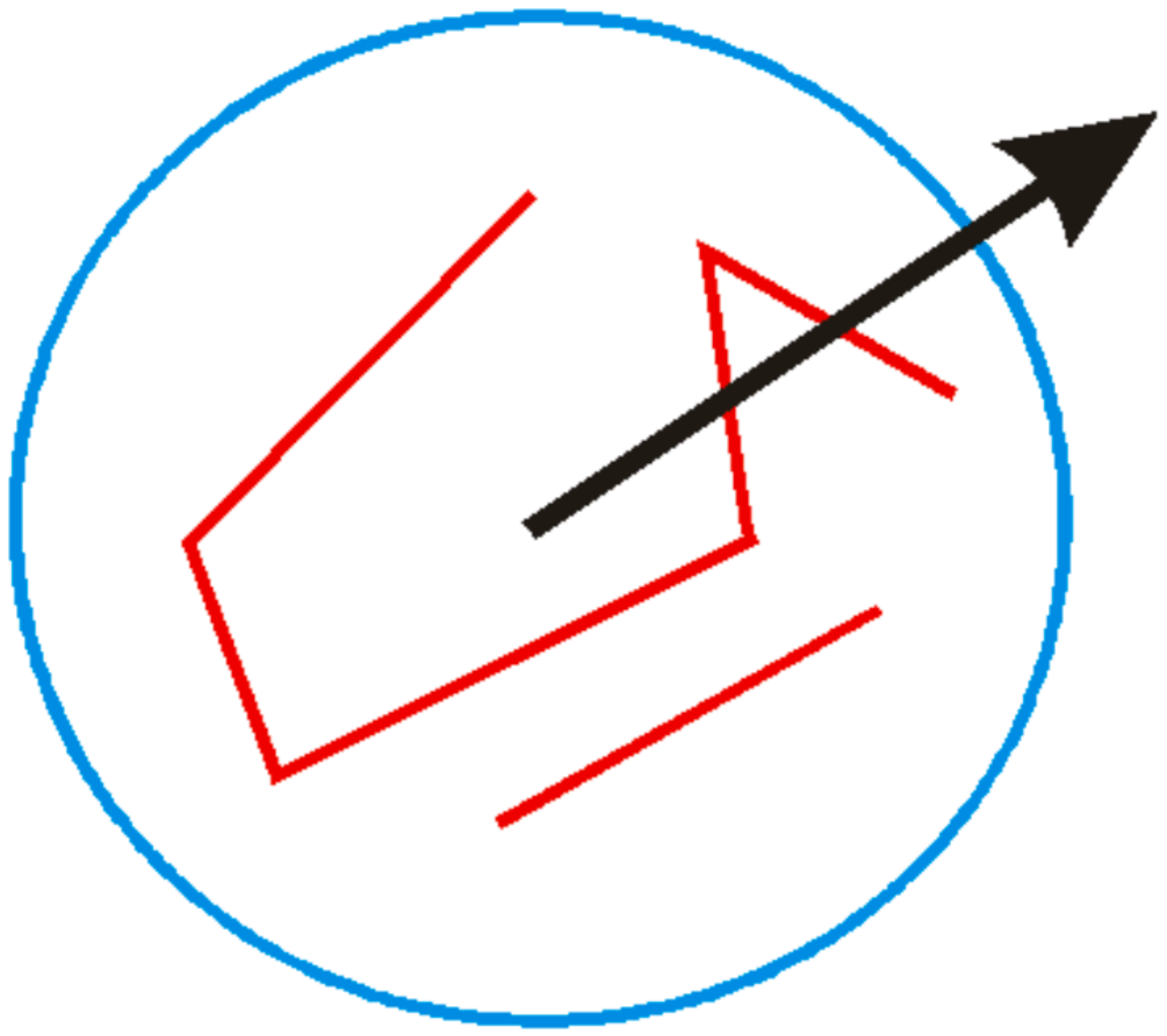
(d) 536 left after testing ratio of principal curvatures

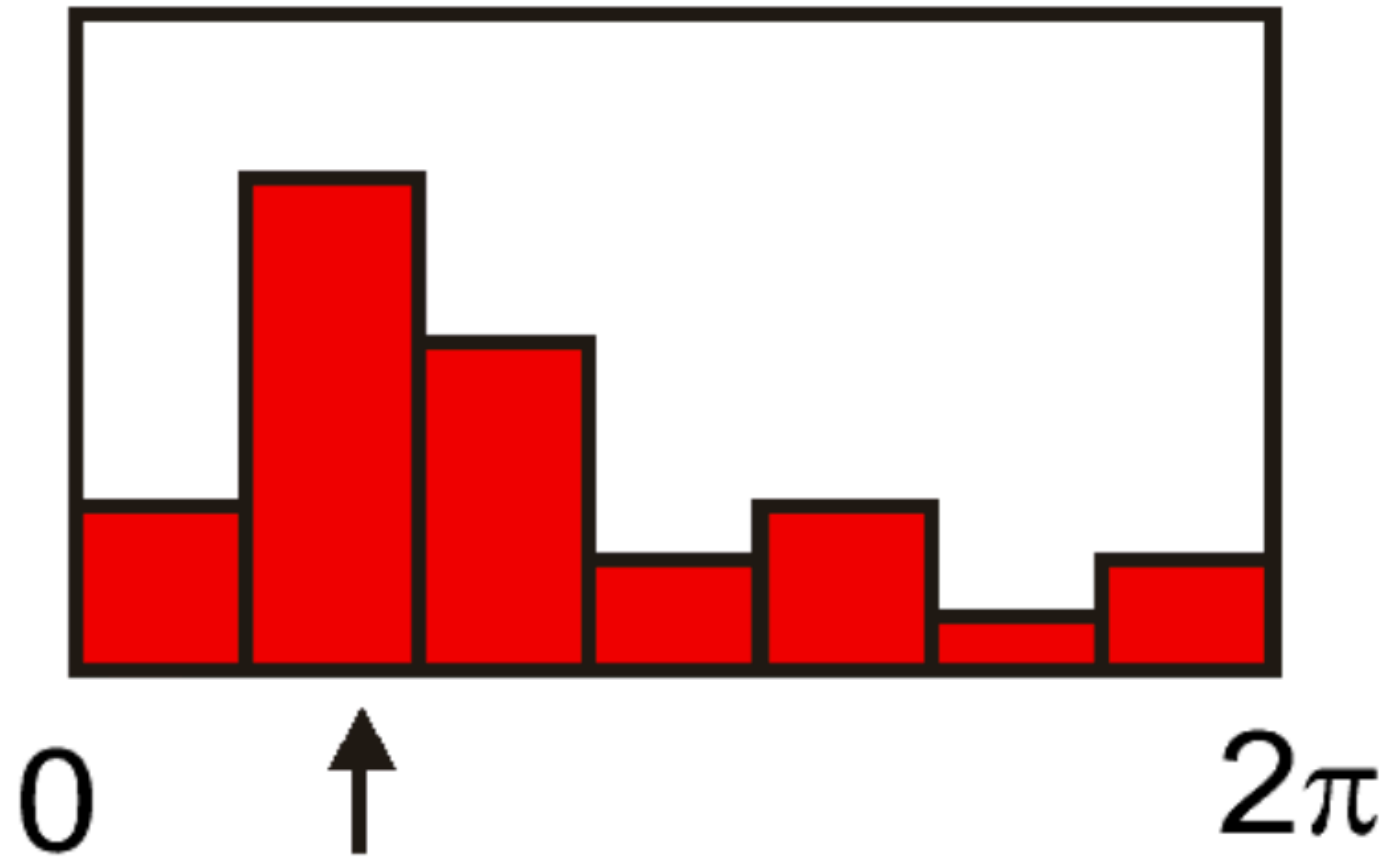
3. Orientation Assignment

— Create **histogram** of local gradient directions computed at selected scale

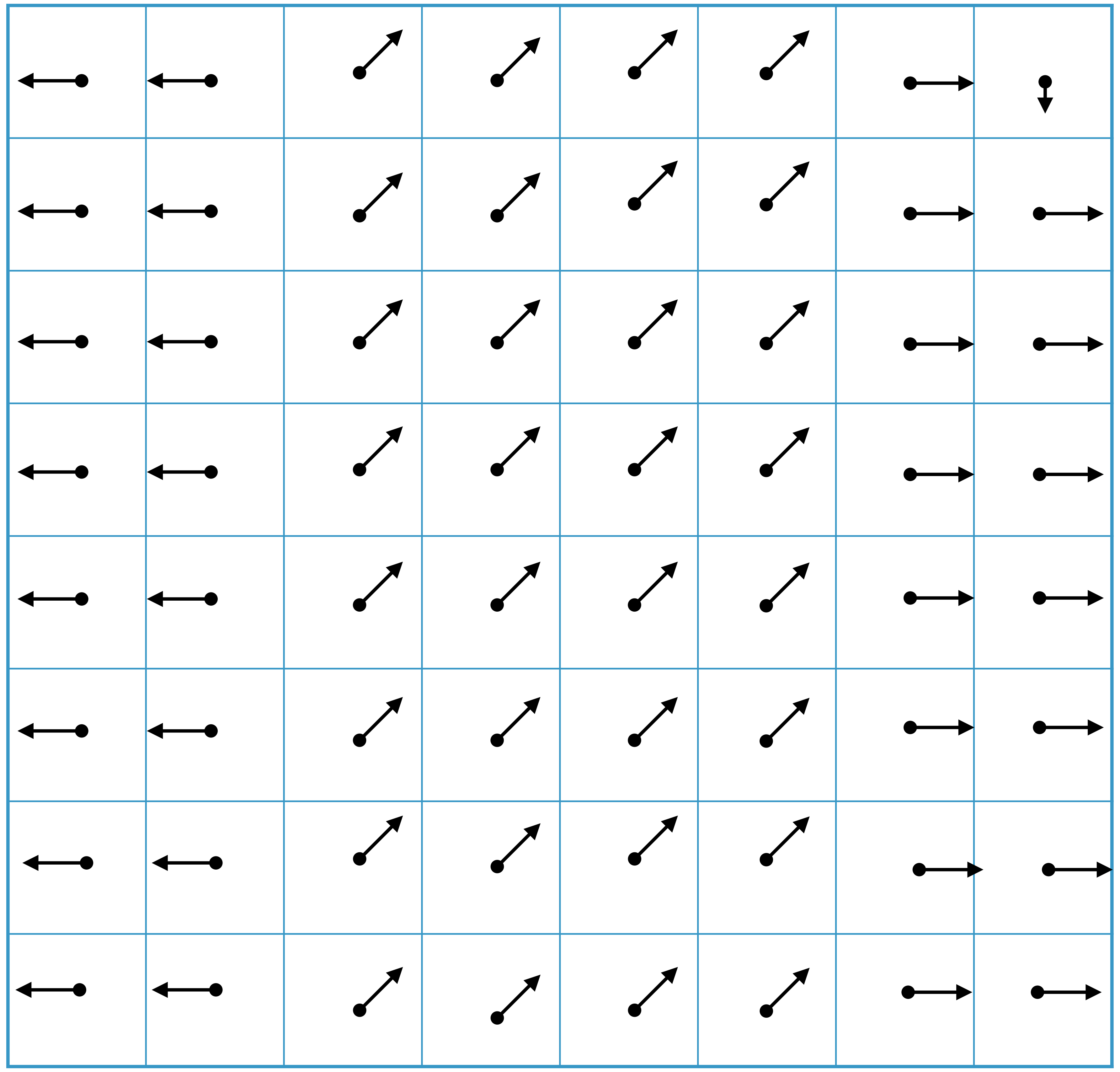
— Assign **canonical orientation** at peak of smoothed histogram

— Each key specifies stable 2D coordinates (x , y , scale, orientation)



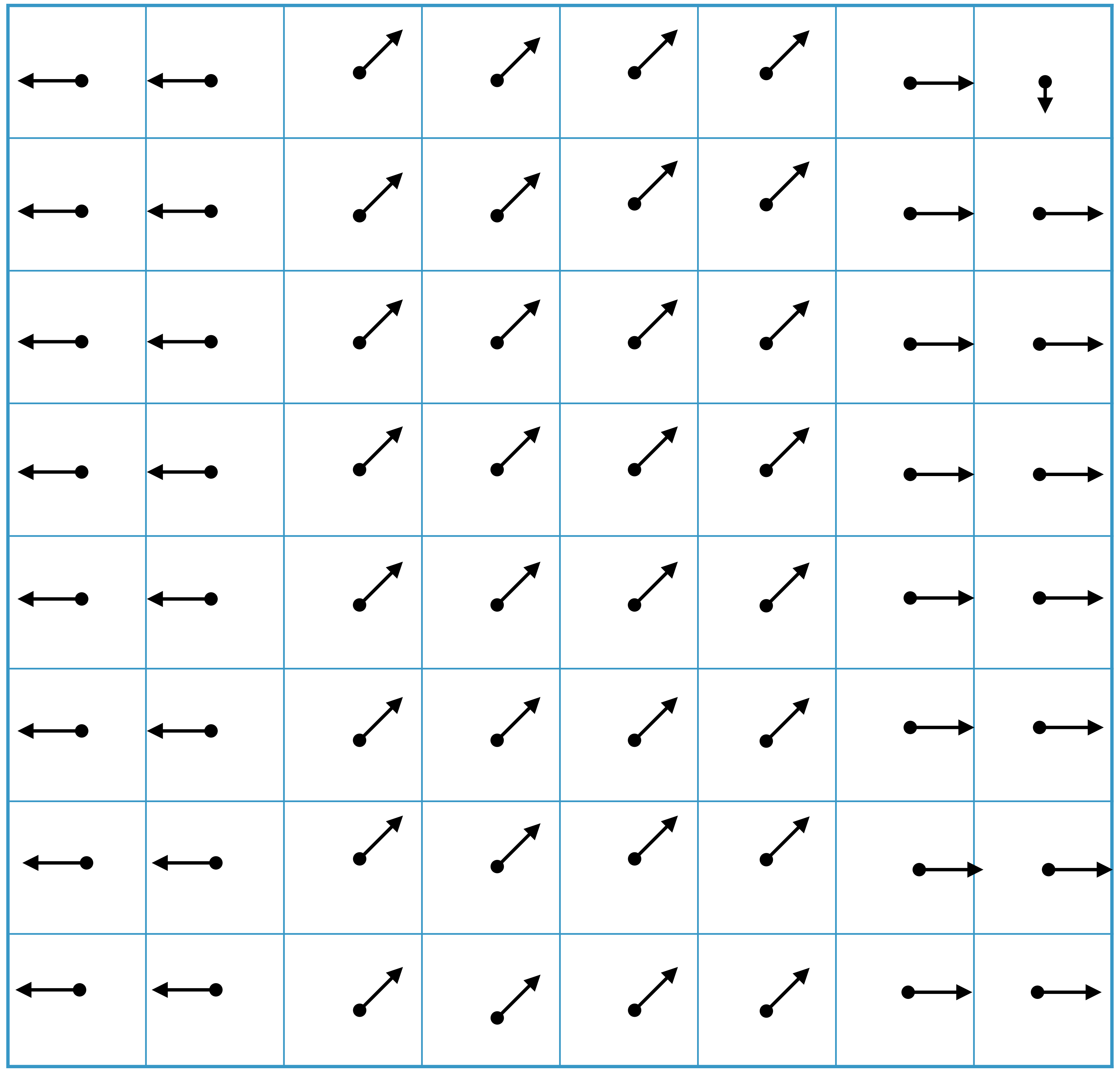


3. Orientation Assignment

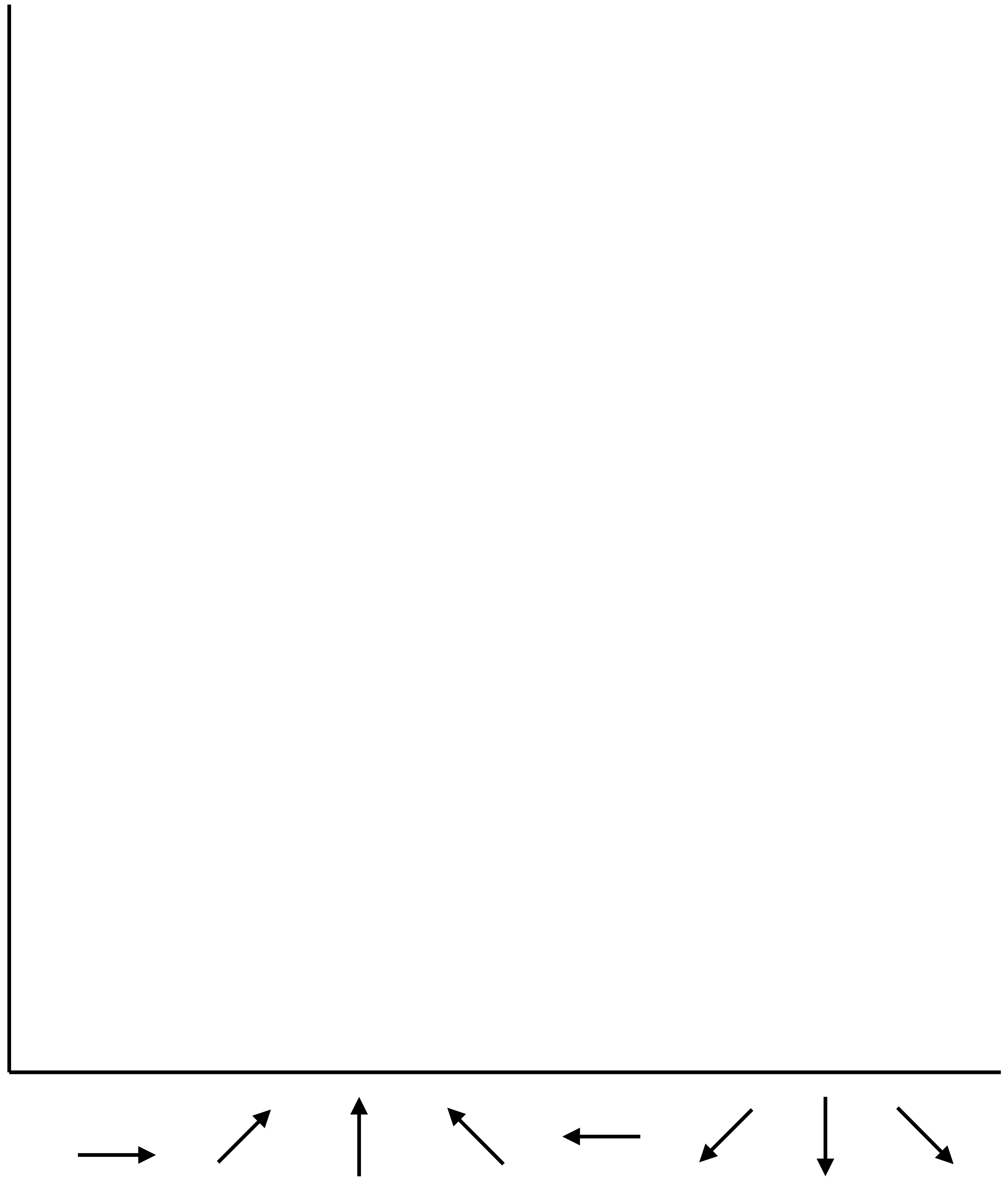


Arrows illustrate **gradient orientation** (direction)
and **gradient magnitude** (arrow length)

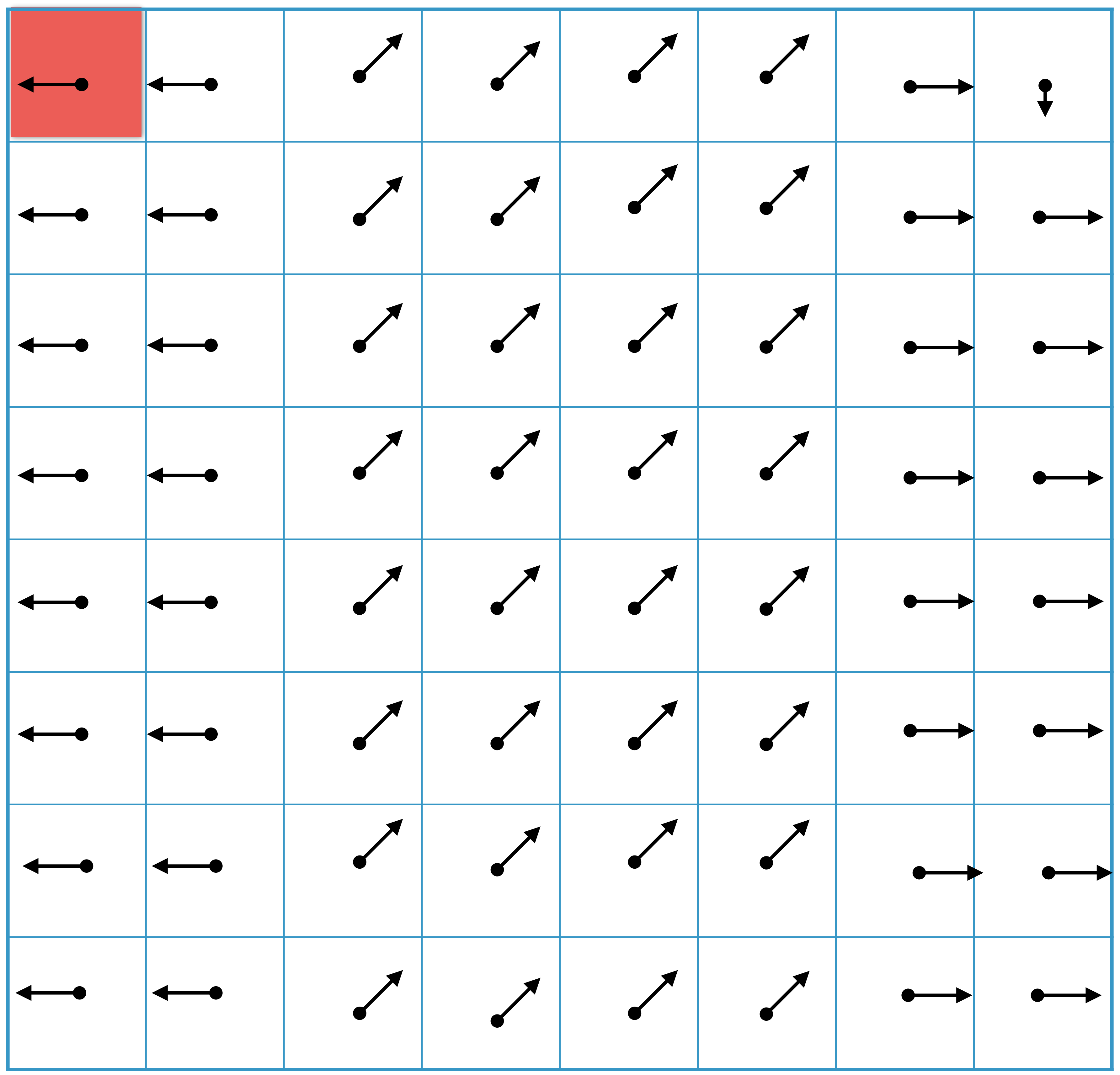
3. Orientation Assignment

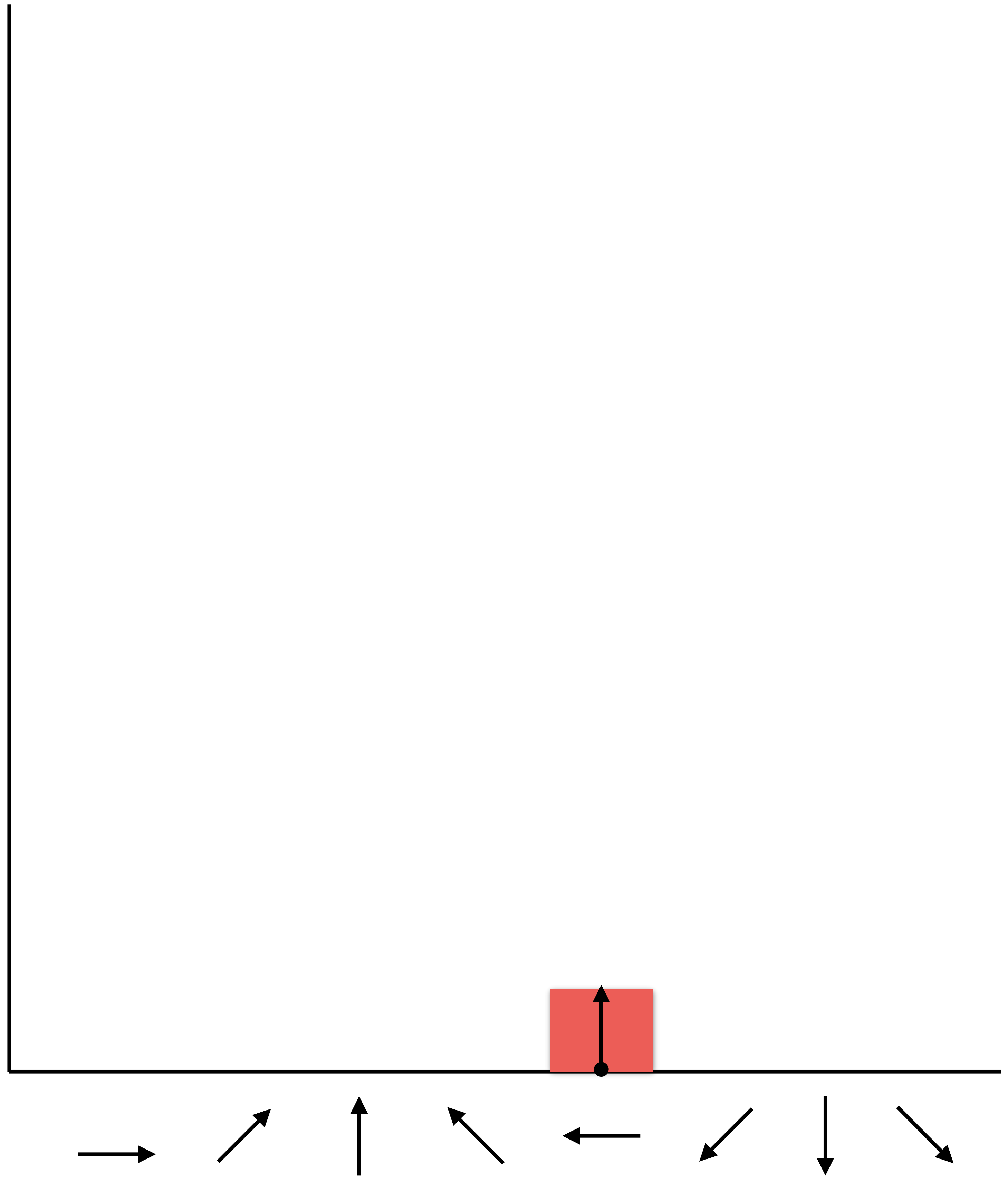


Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)



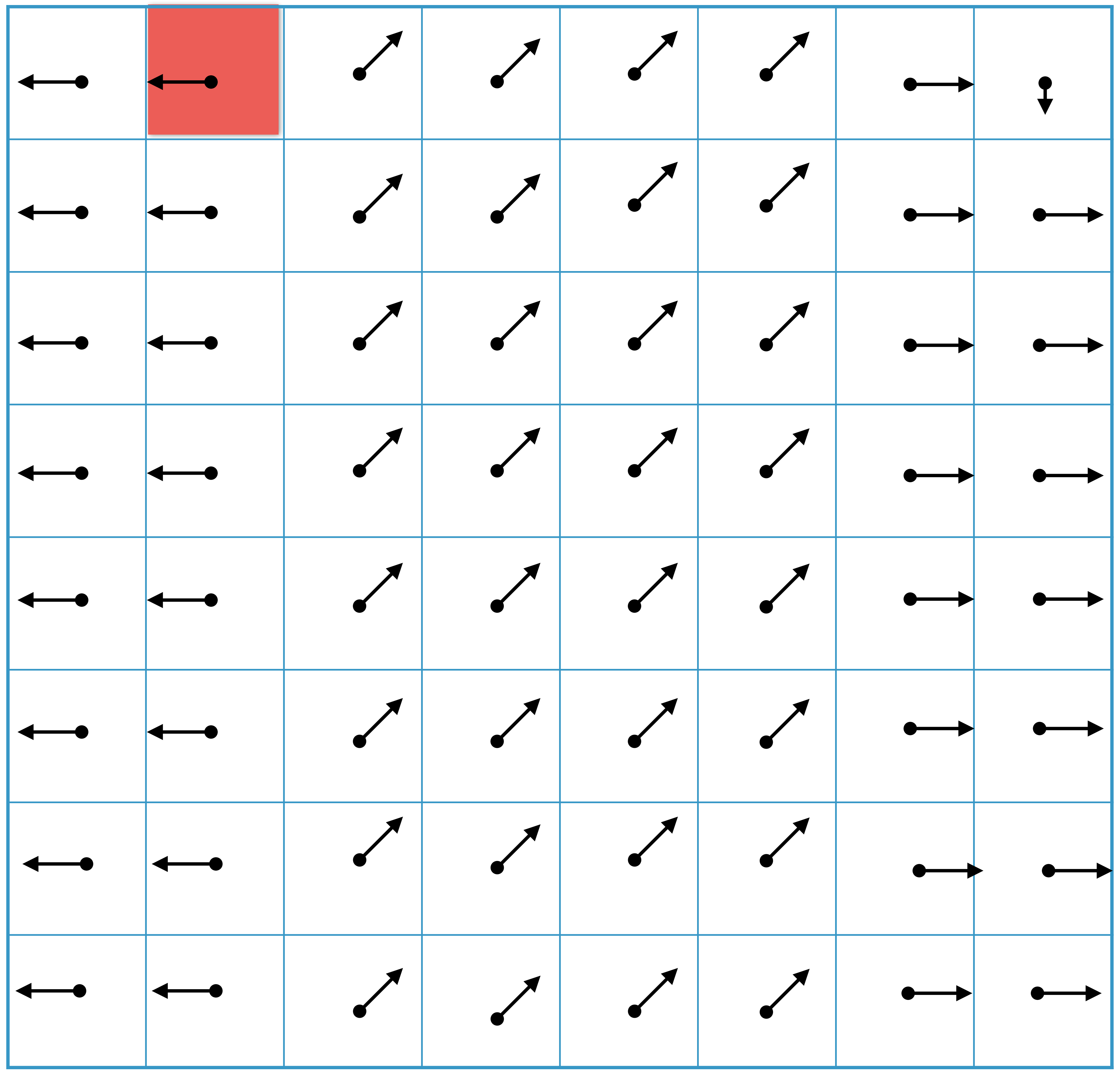
3. Orientation Assignment

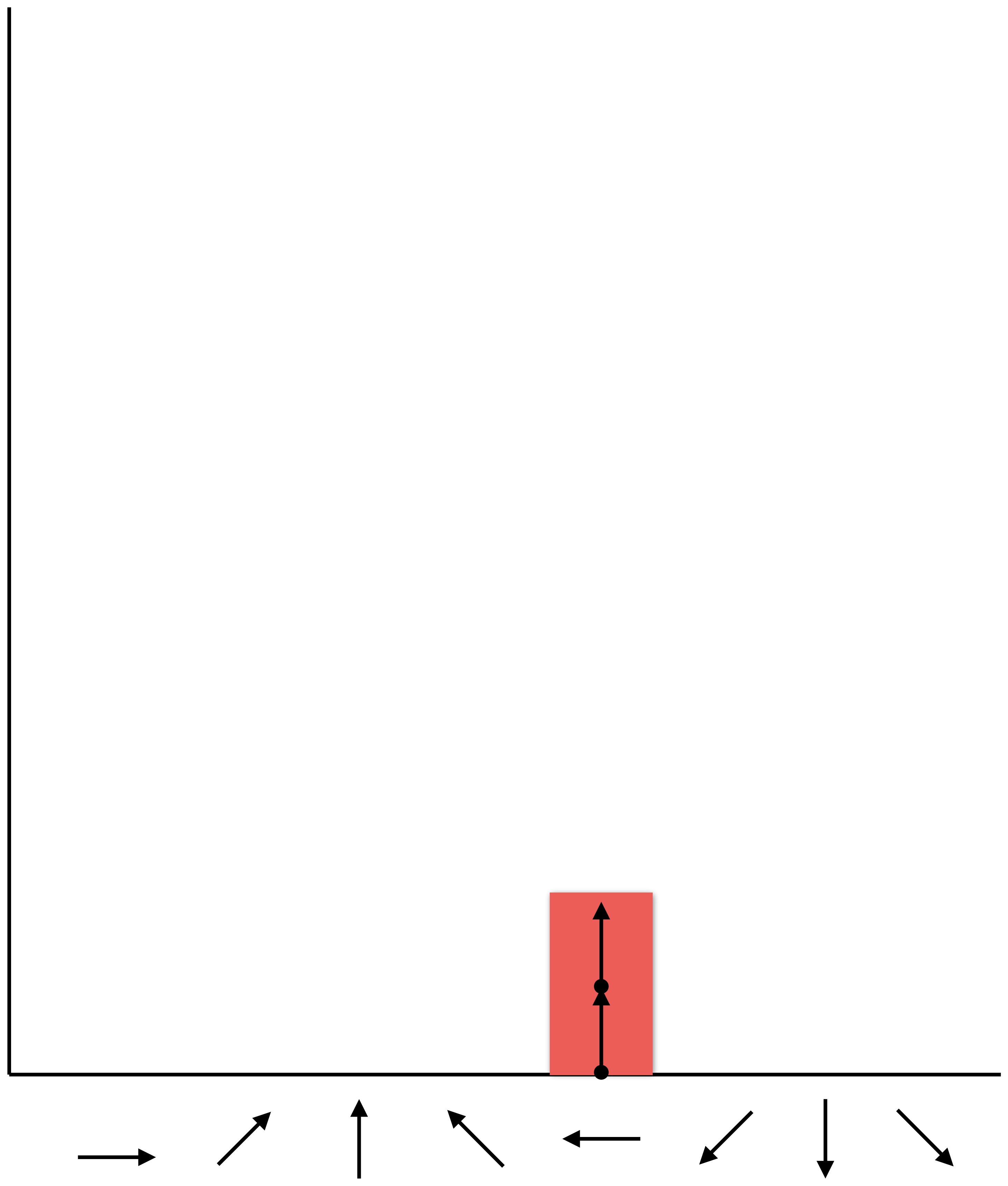




Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

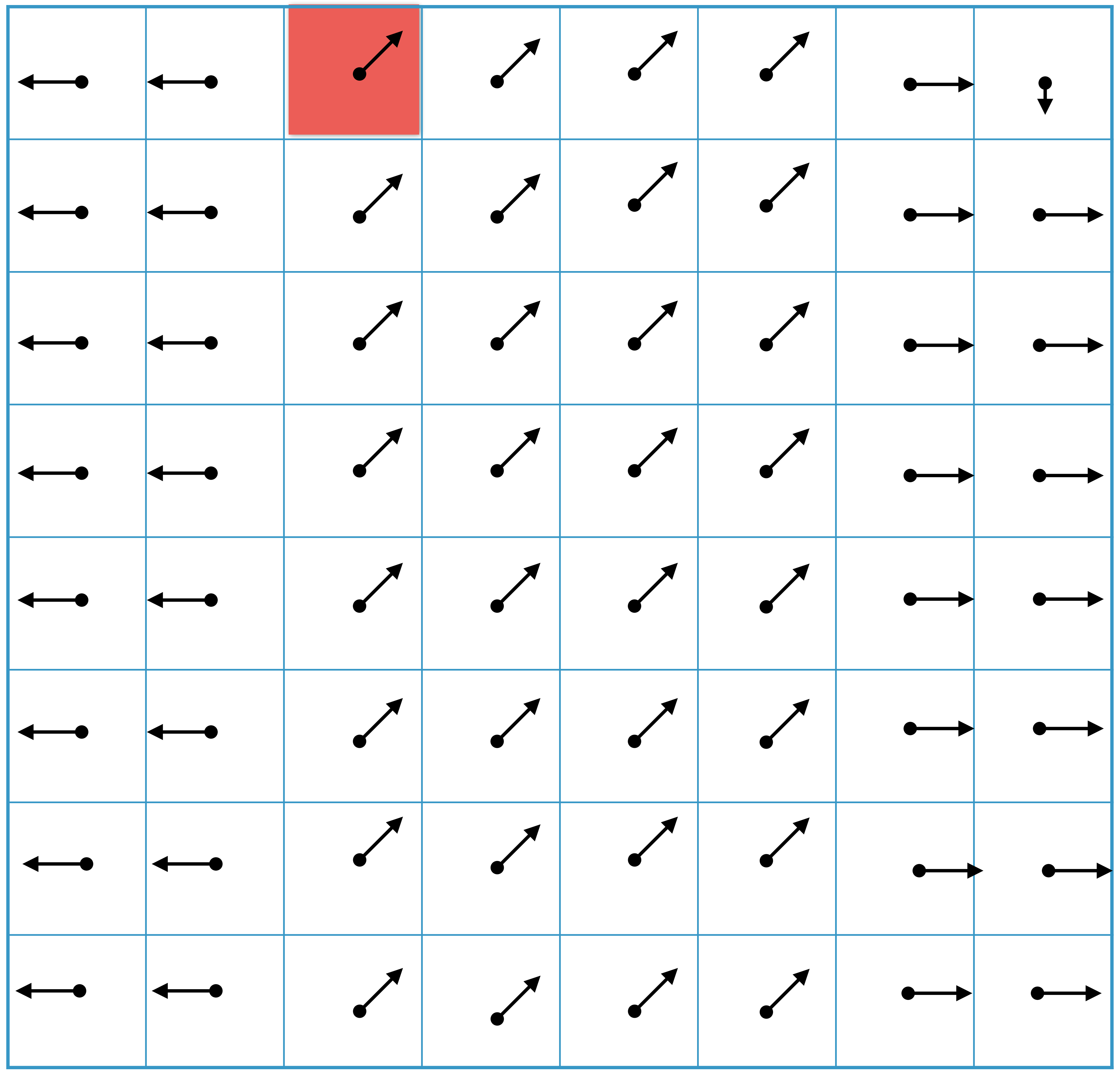
3. Orientation Assignment

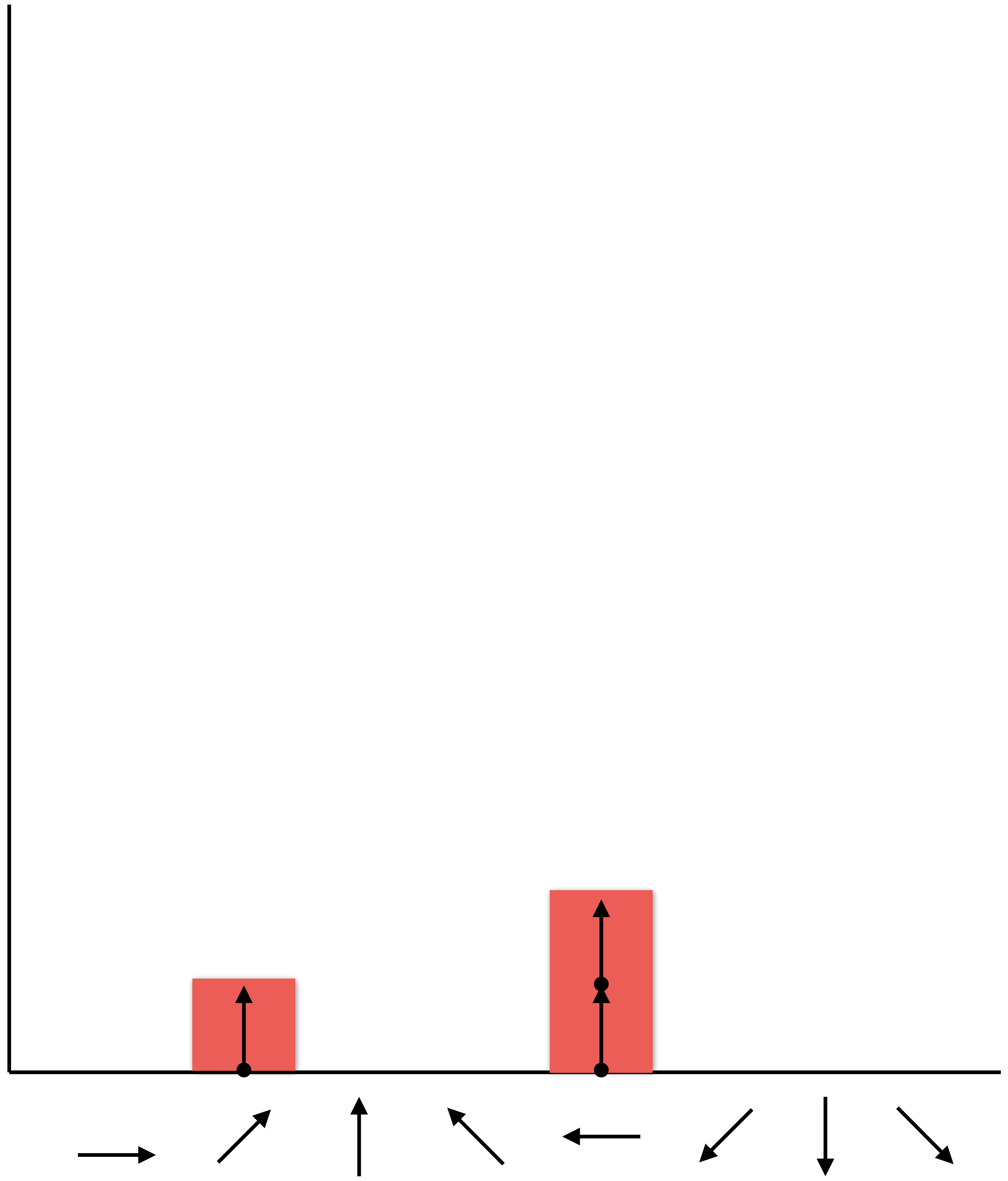




Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

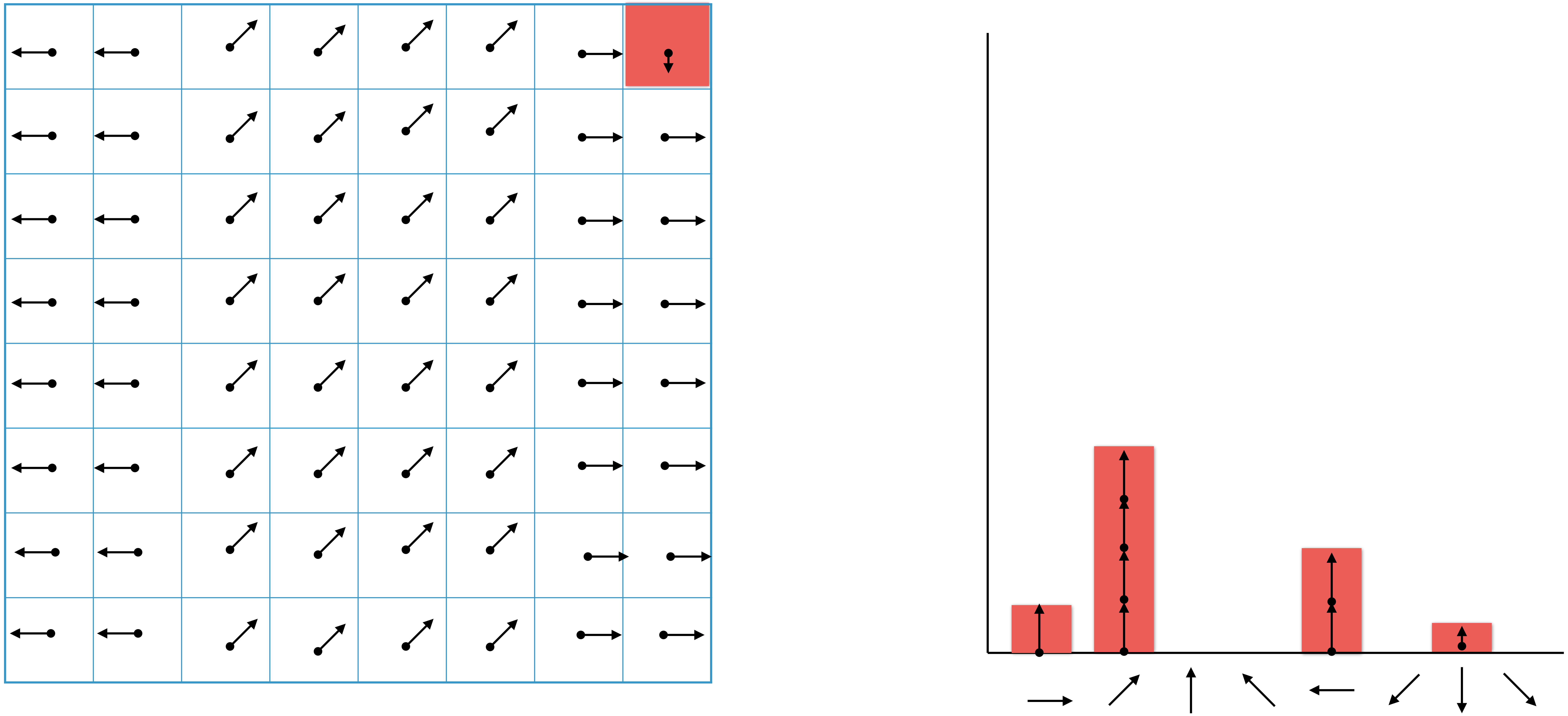
3. Orientation Assignment





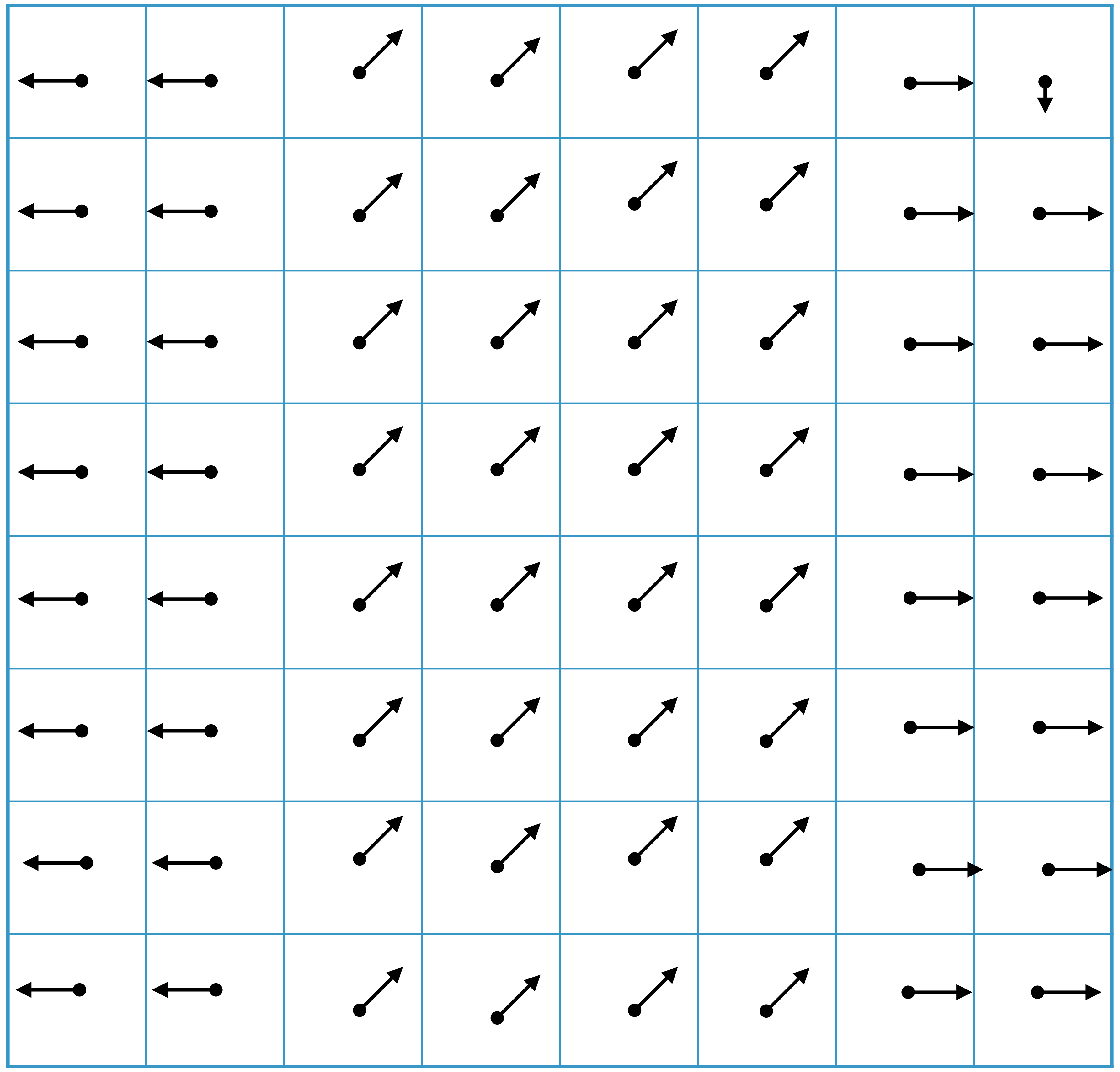
Arrows illustrate **gradient orientation** (direction)
and **gradient magnitude** (arrow length)

3. Orientation Assignment

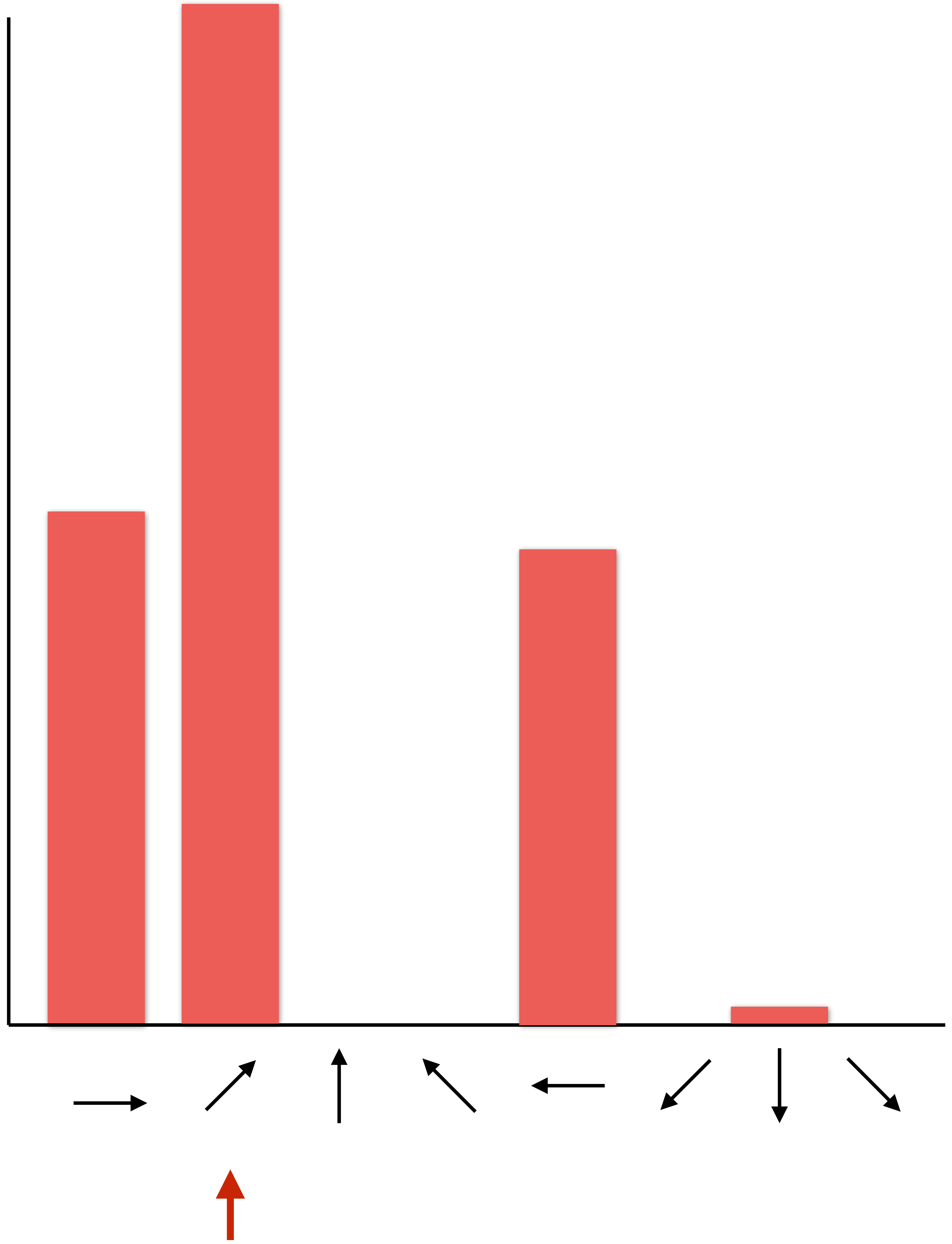


Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

3. Orientation Assignment

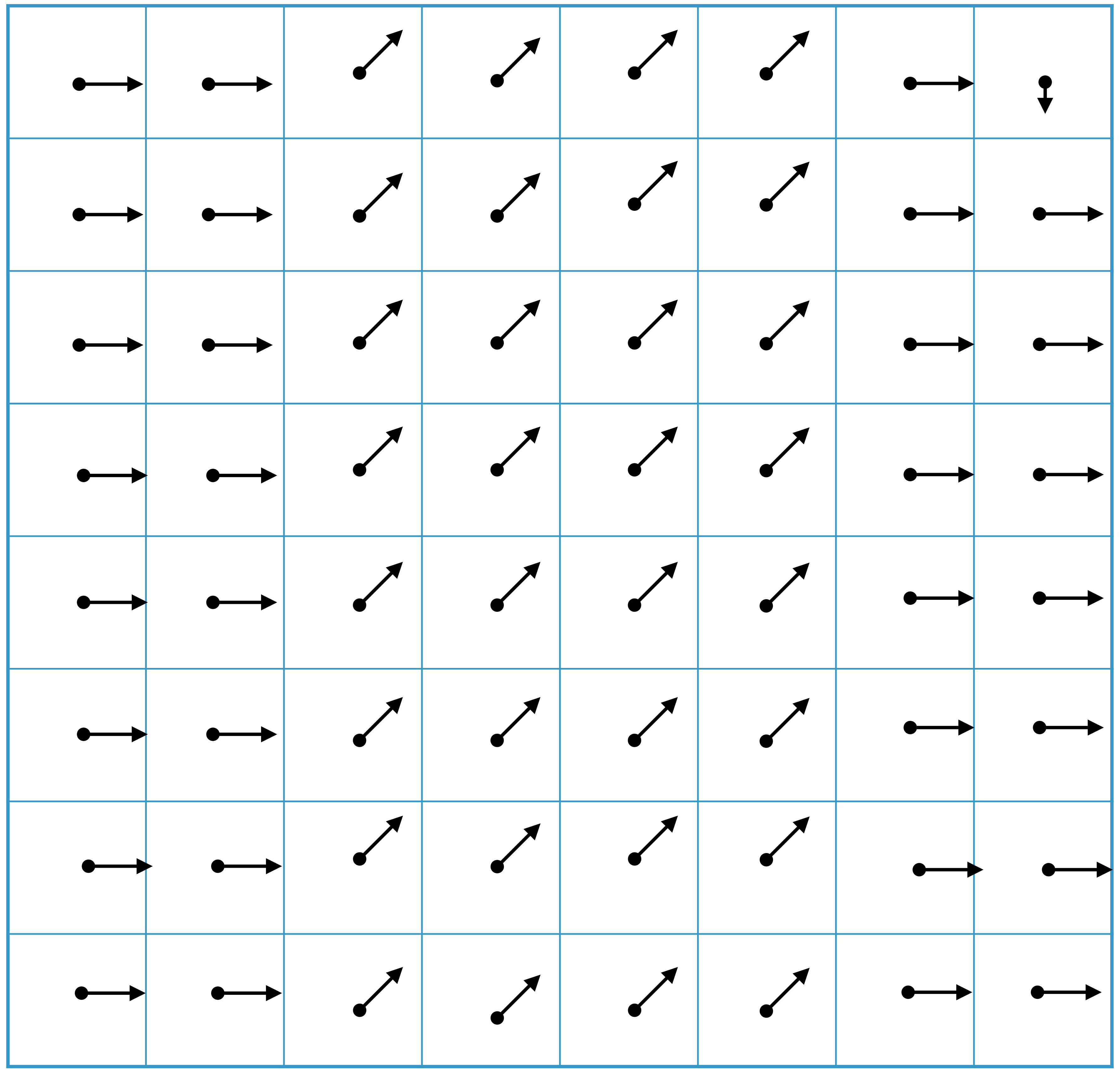


Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

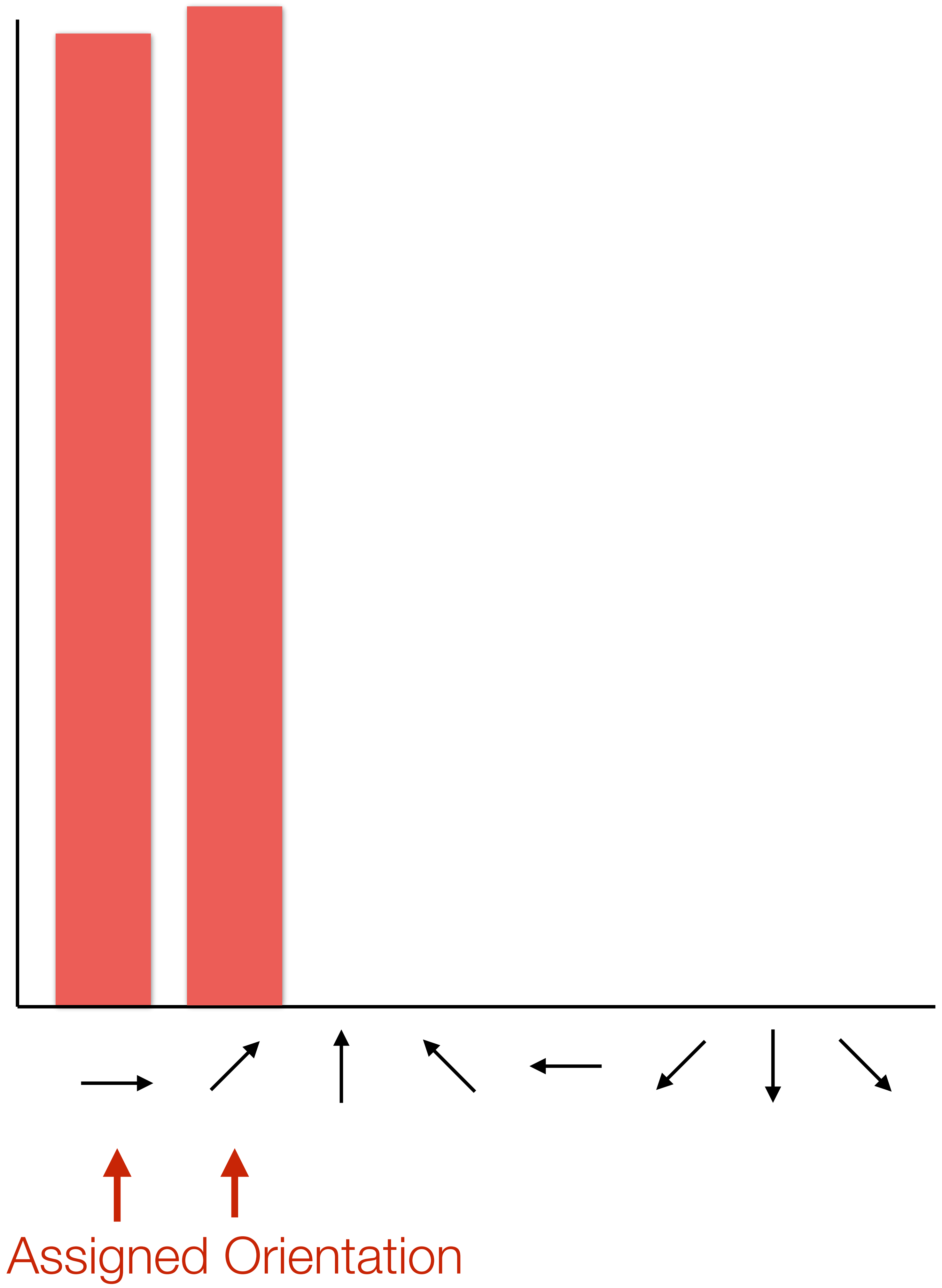


Assigned Orientation

3. Orientation Assignment

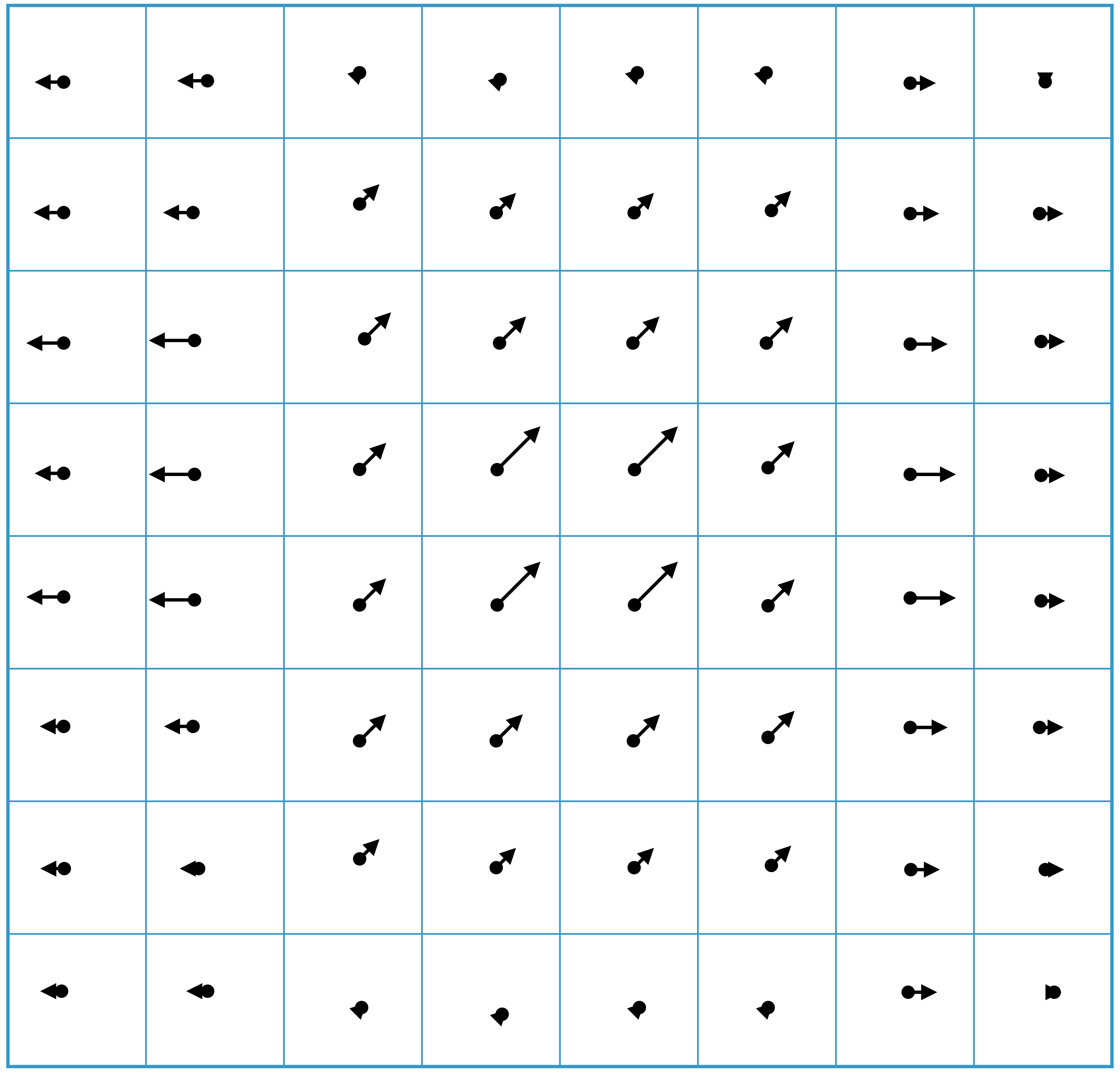


Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

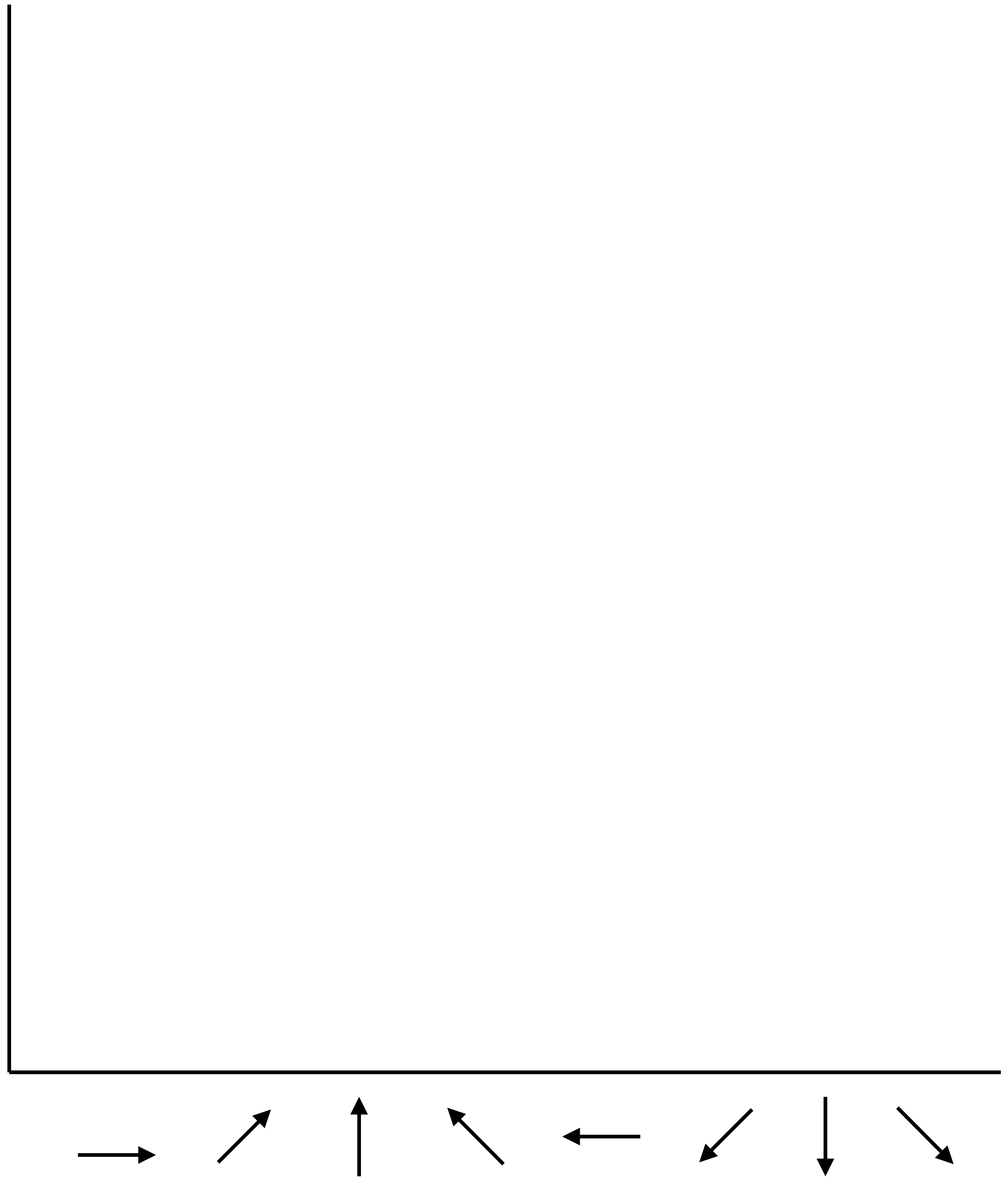


3. Orientation Assignment

Multiply **gradient magnitude** by a **Gaussian** kernel



Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)



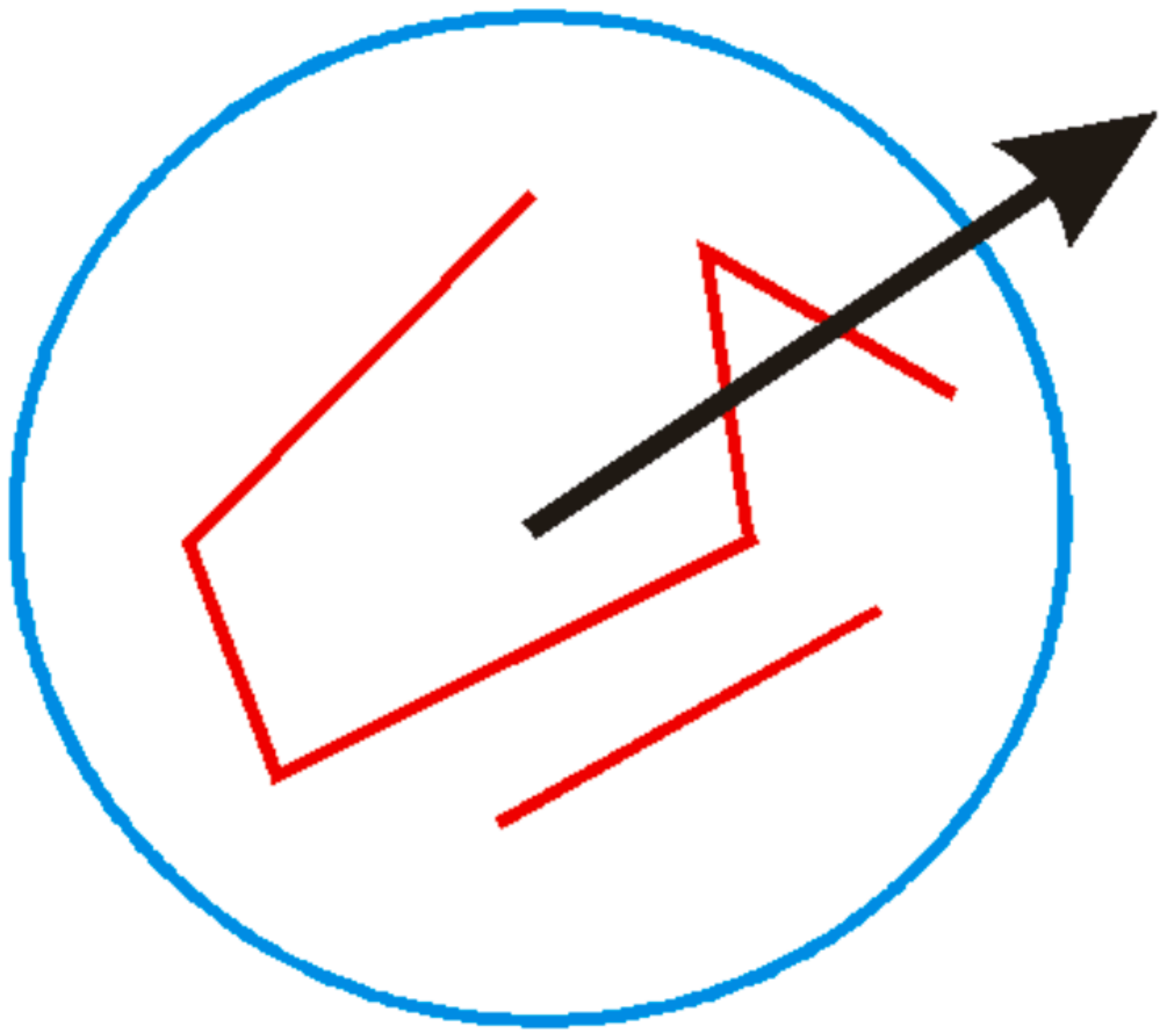
3. Orientation Assignment

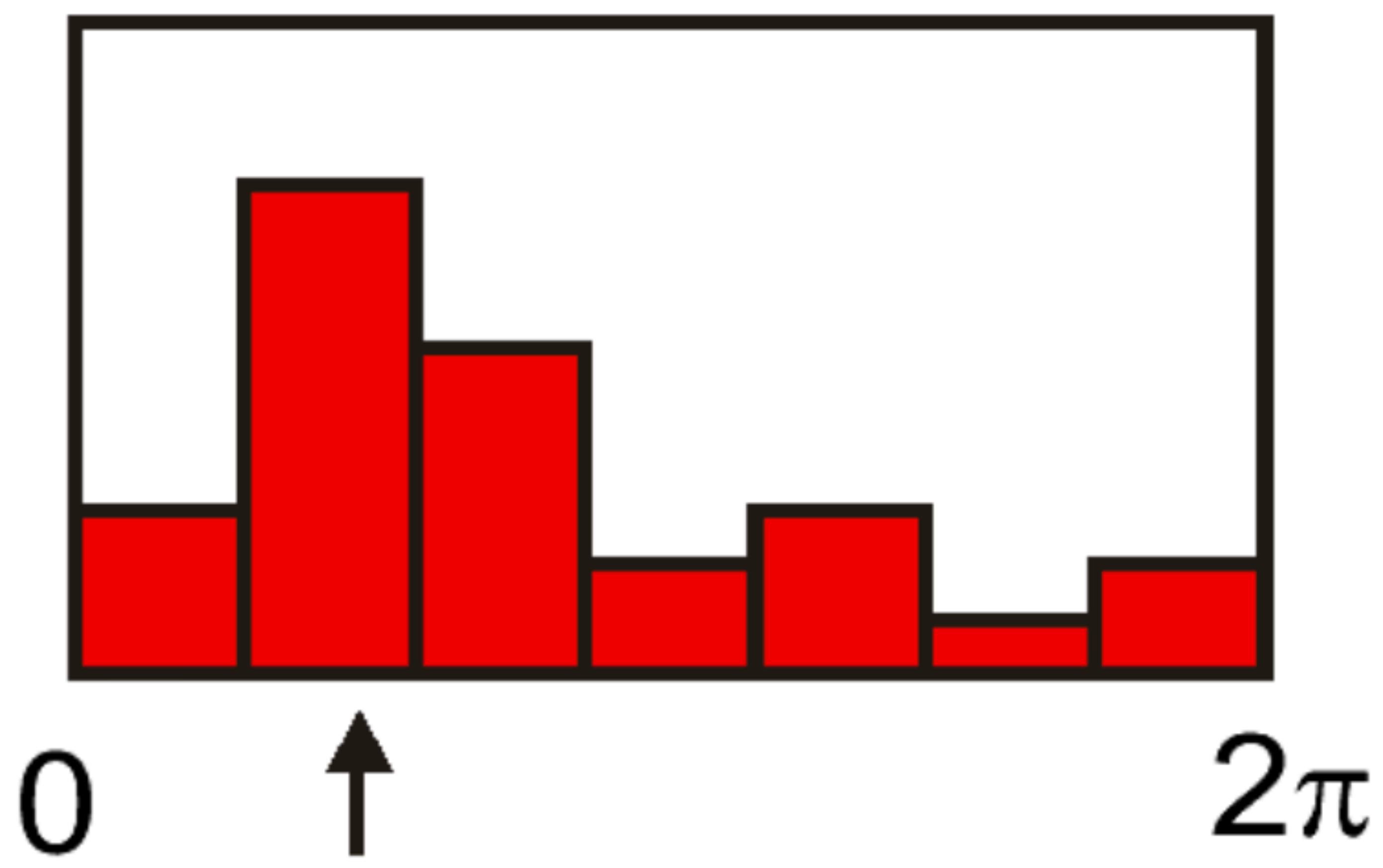
— **Histogram** of 36 bins (10 degree increments)

— Size of the **window** is 1.5 scale (recall the Gaussian filter)

— Gaussian-weighted **voting**

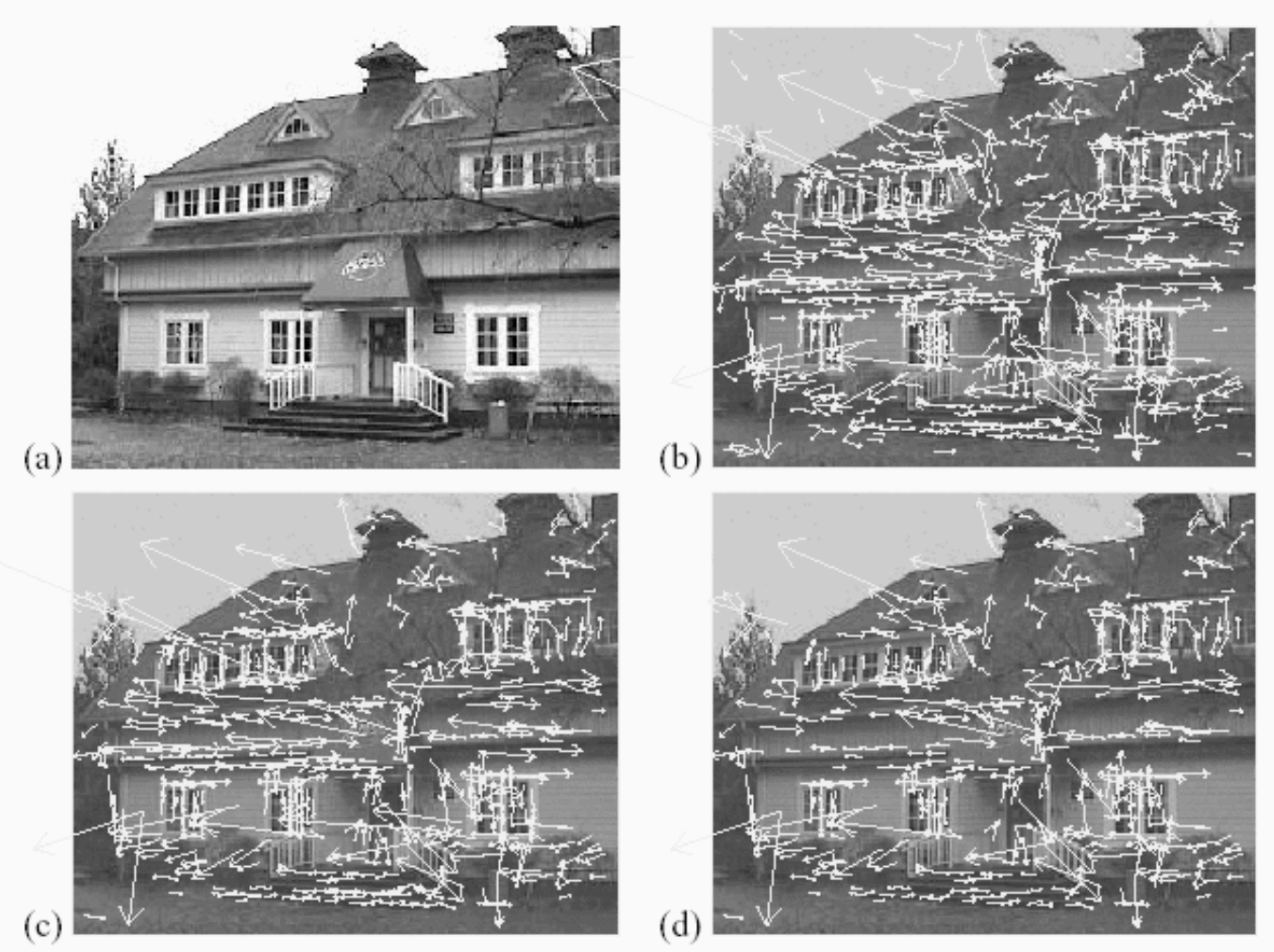
— Highest **peak** and peaks above 80% of highest also considered for calculating dominant orientations





3. 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

4. Keypoint Description

We have seen how to assign a location, scale, and orientation to each key point — **keypoint detection**

— The next step is to compute a **keypoint descriptor**: should be robust to local shape distortions, changes in illumination or 3D viewpoint

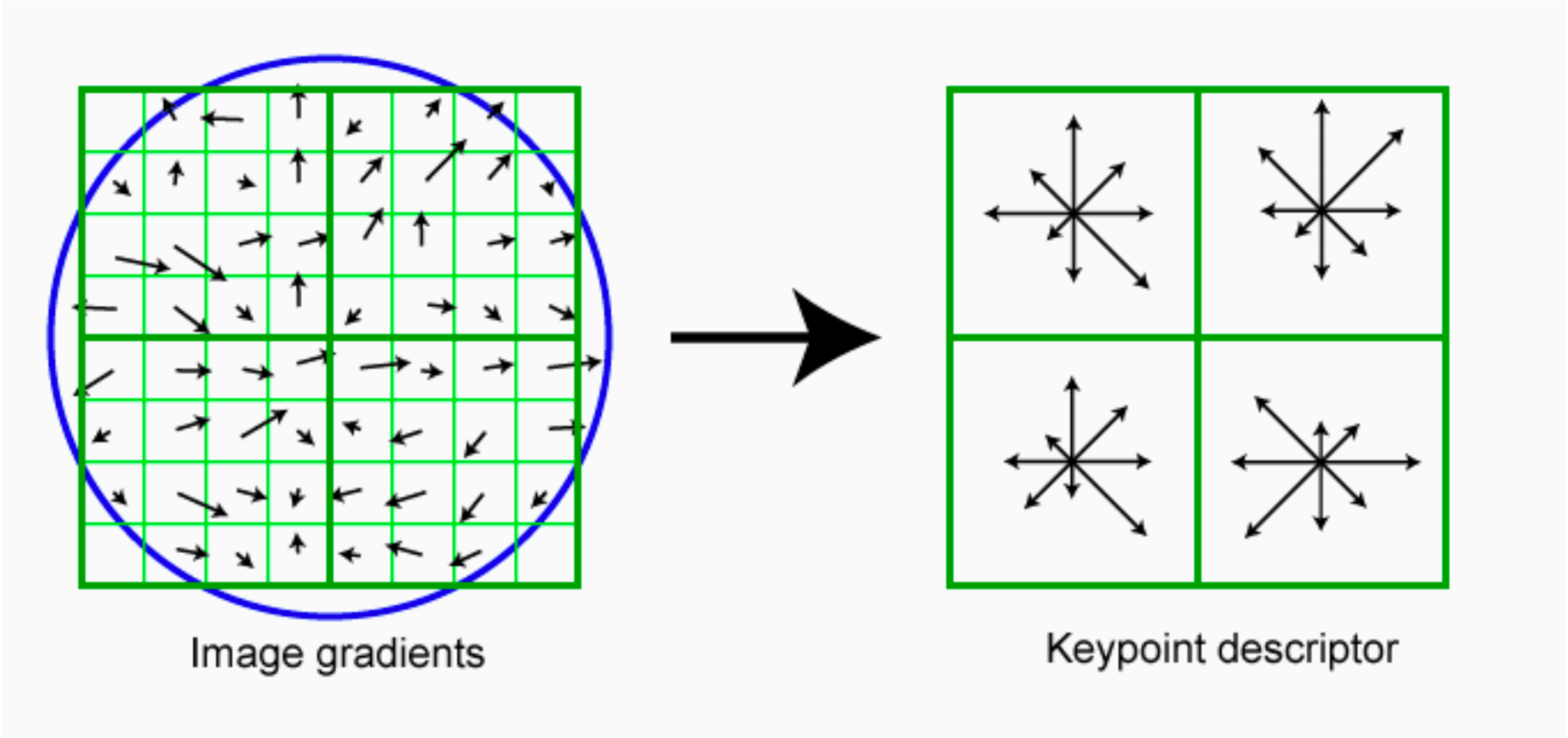
— Keypoint detection is not the same as keypoint description, e.g. some applications skip keypoint detection and extract SIFT descriptors on a regularly spaced grid

4. SIFT Descriptor

— Thresholded image gradients are sampled over 16 × 16 array of locations in scale space (weighted by a Gaussian with sigma half the size of the window)

— Create array of orientation histograms

— 8 orientations × 4 × 4 histogram array

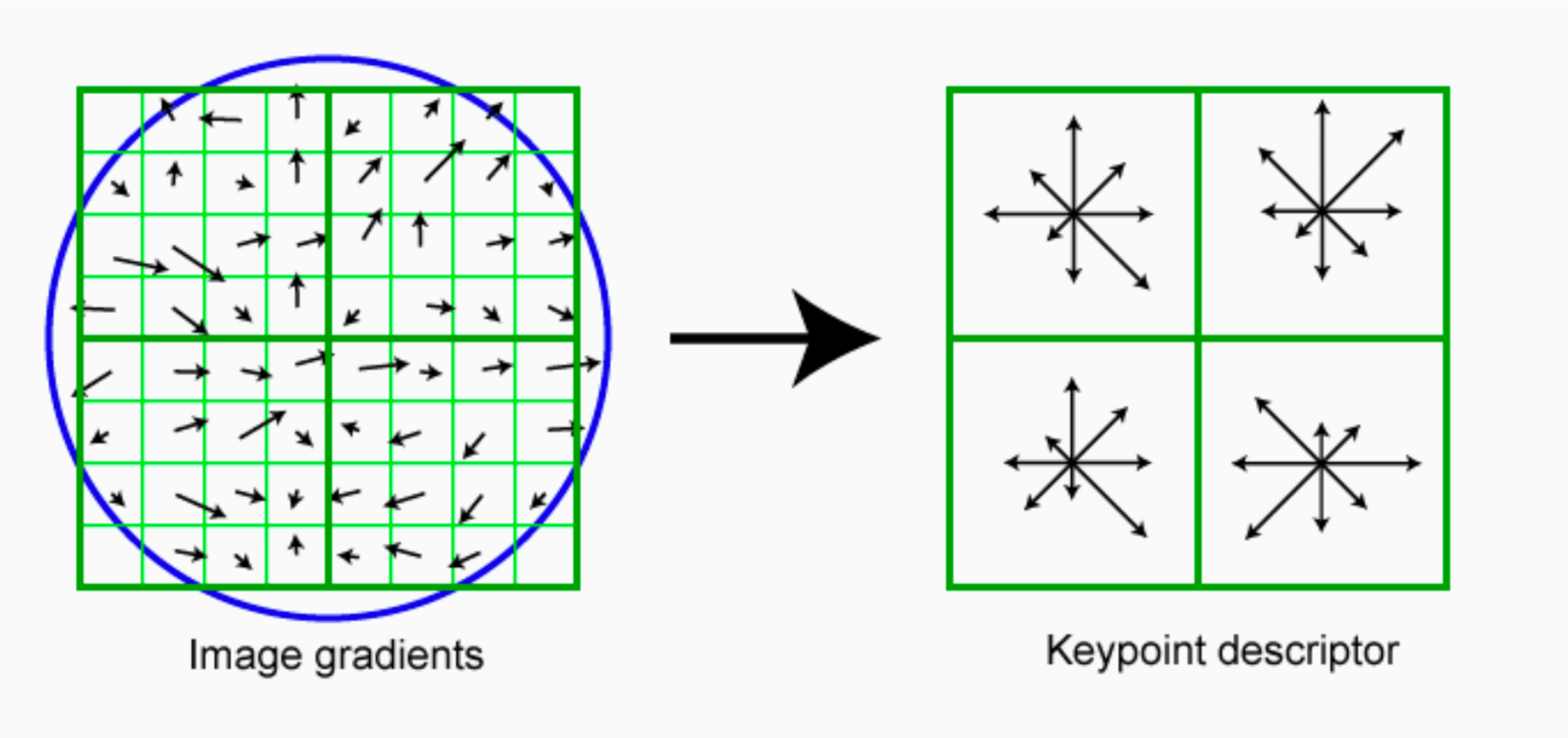


Demo

4. SIFT Descriptor

How many dimensions are there in a SIFT descriptor?

(**Hint**: This diagram shows a 2 x 2 histogram array but the actual descriptor uses a 4 x 4 histogram array)



4. SIFT Descriptor

Descriptor is **normalized** to unit length (i.e. magnitude of 1) to reduce the effects of illumination change

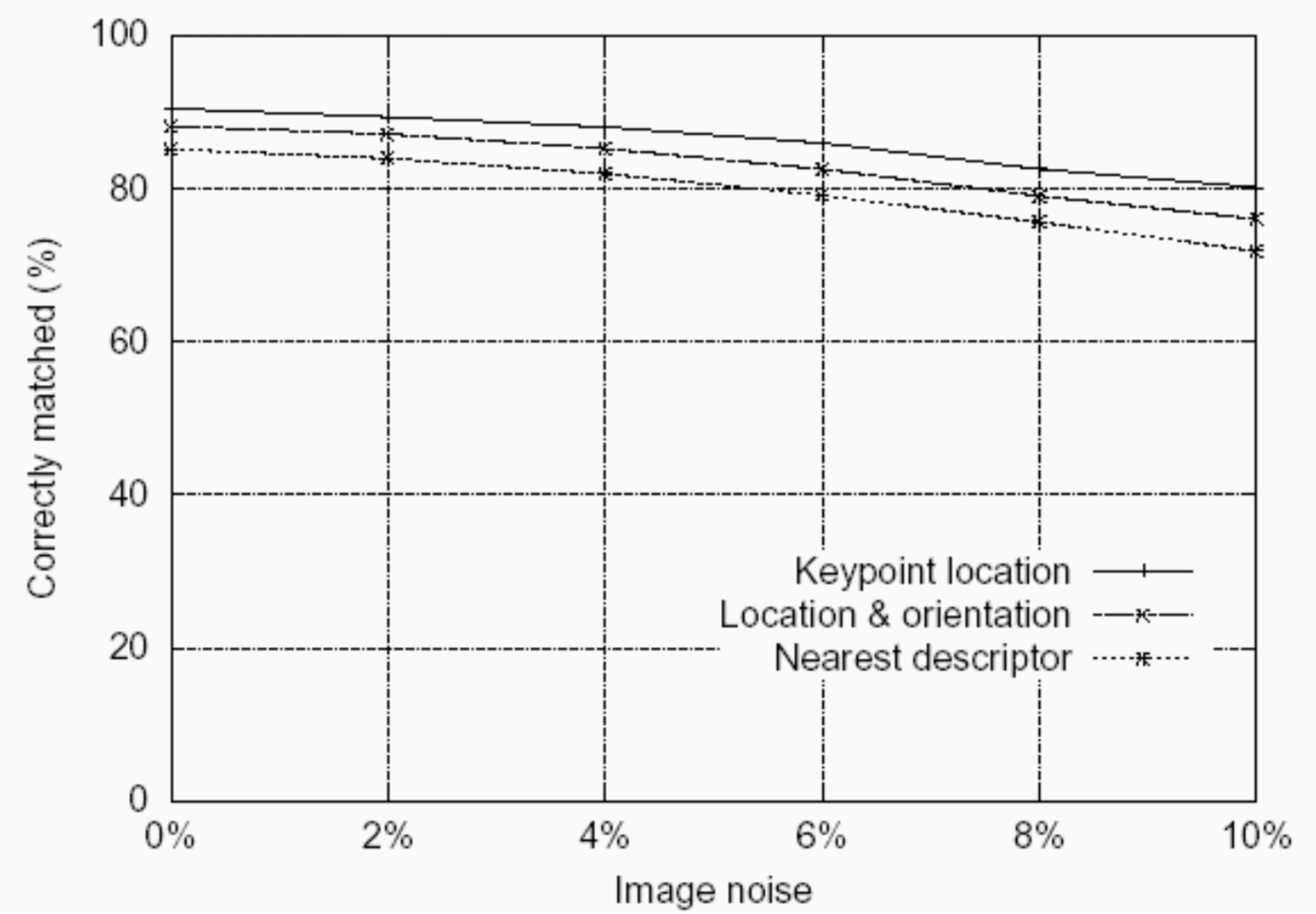
— if brightness values are scaled (multiplied) by a constant, the gradients are scaled by the same constant, and the normalization cancels the change

— if brightness values are increased/decreased by a constant, the gradients do not change

Feature Stability to **Noise**

Match features after random change in image scale & orientation, with differing levels of image noise

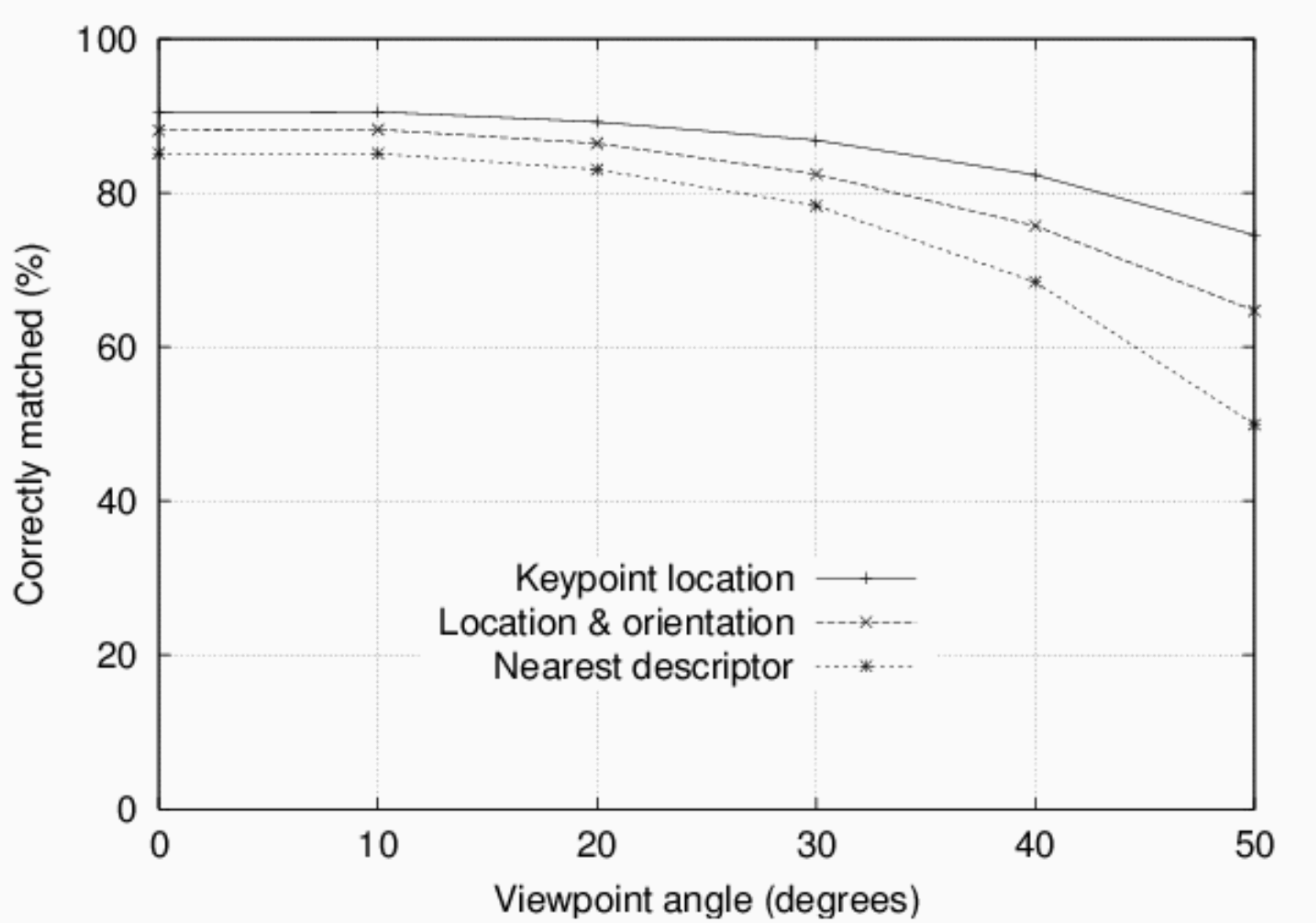
Find nearest neighbour in database of 30,000 features



Feature Stability to **Affine Change**

Match features after random change in image scale & orientation, with differing levels of image noise

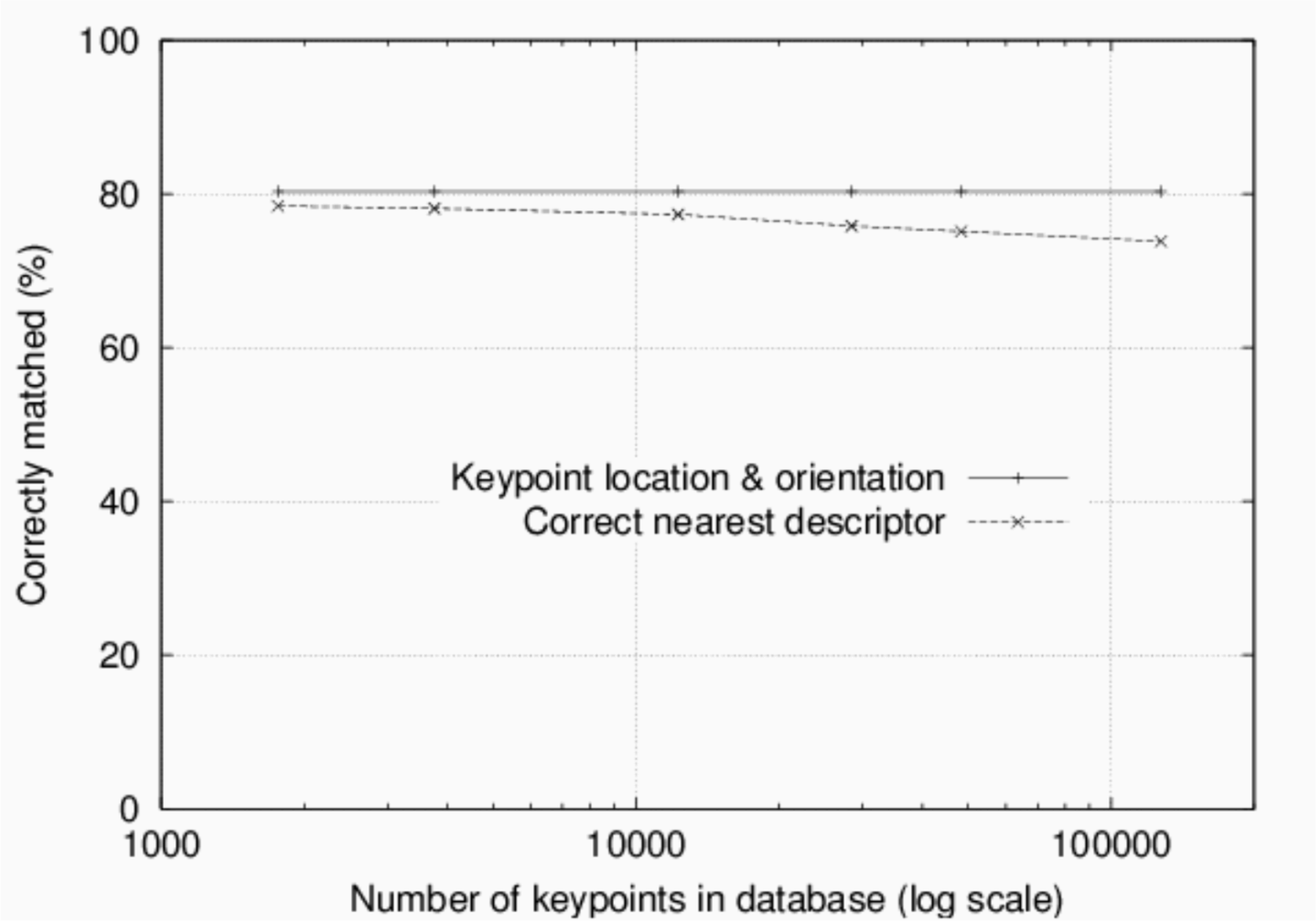
Find nearest neighbour in database of 30,000 features



Distinctiveness of Features

Vary size of database of features, with 30 degree affine change, 2% image noise

Measure % correct for single nearest neighbour match



Summary

Four steps to SIFT feature generation:

1. **Scale-space representation and local extrema detection**
   — use DoG pyramid
   — 3 scales/octave, down-sample by factor of 2 each octave
2. **Keypoint localization**
   — select stable keypoints (threshold on magnitude of extremum, ratio of principal curvatures)
3. **Keypoint orientation assignment**
   — based on histogram of local image gradient directions
4. **Keypoint descriptor**
   — histogram of local gradient directions — vector with $8 \times (4 \times 4) = 128$ dim
   — vector normalized (to unit length)