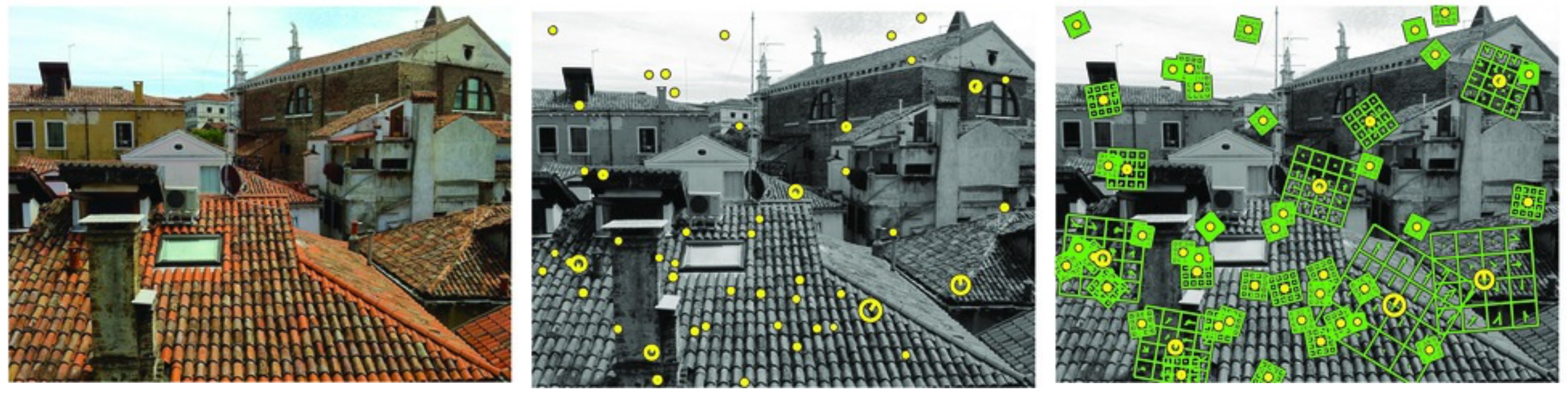
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 13: SIFT cont., HOG, SURF, Object Recognition

Menu for Today (**February 25, 2020**)

Topics:

— SIFT continued

— HOG, SURF descriptors

— Object detection with SIFT

— RANSAC intro

Readings:

— **Today's** Lecture: Forsyth & Ponce (2nd ed.) 5.4, 10.4.2
"Distinctive Image Features for Scale-Invariant Keypoints

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

Reminders:

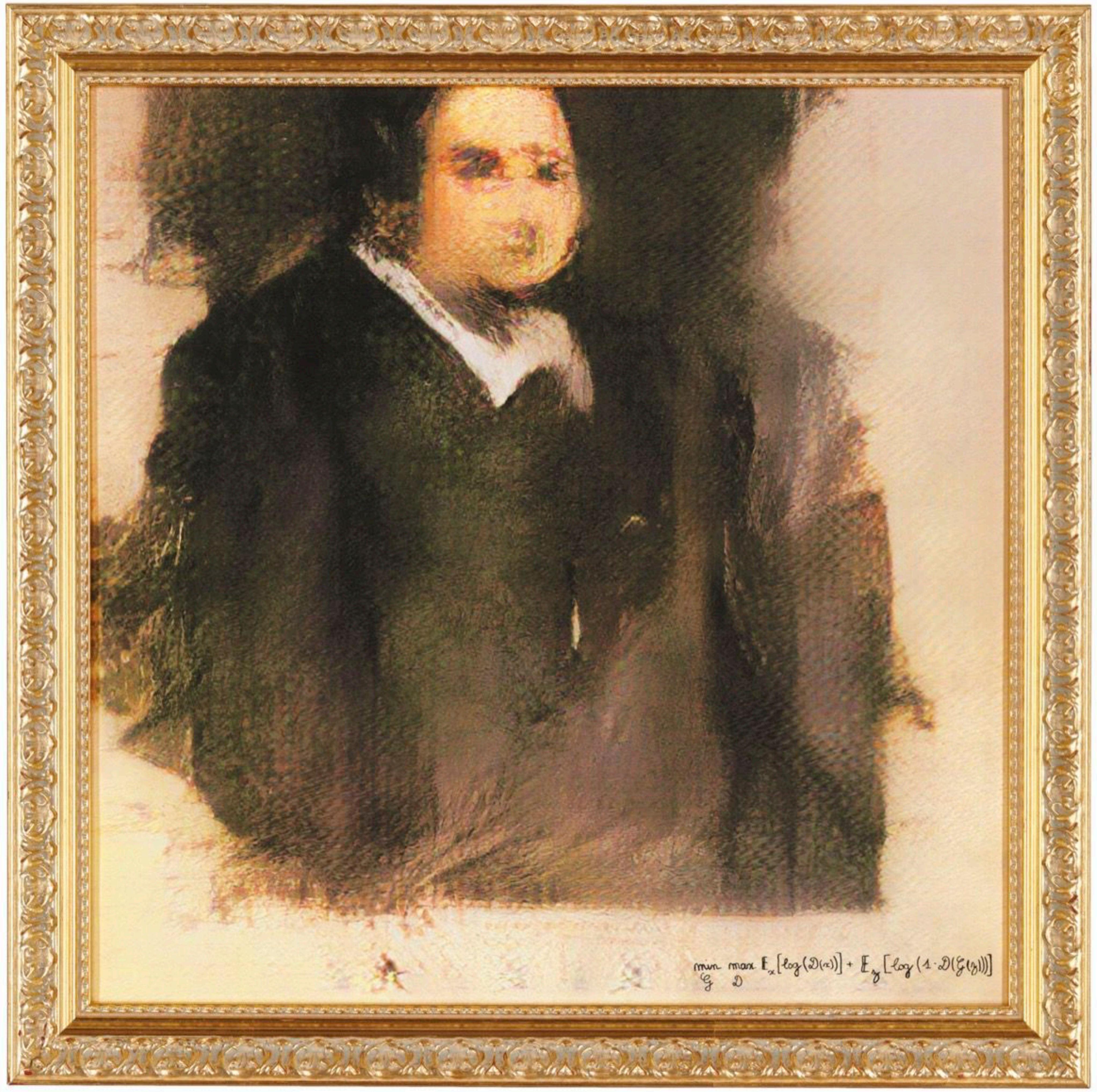
— **Assignment 3**: is due next **Tuesday, March 3rd** (Assignment 2 almost graded)

— **Midterm** is next class, Thursday, February 27th — in class

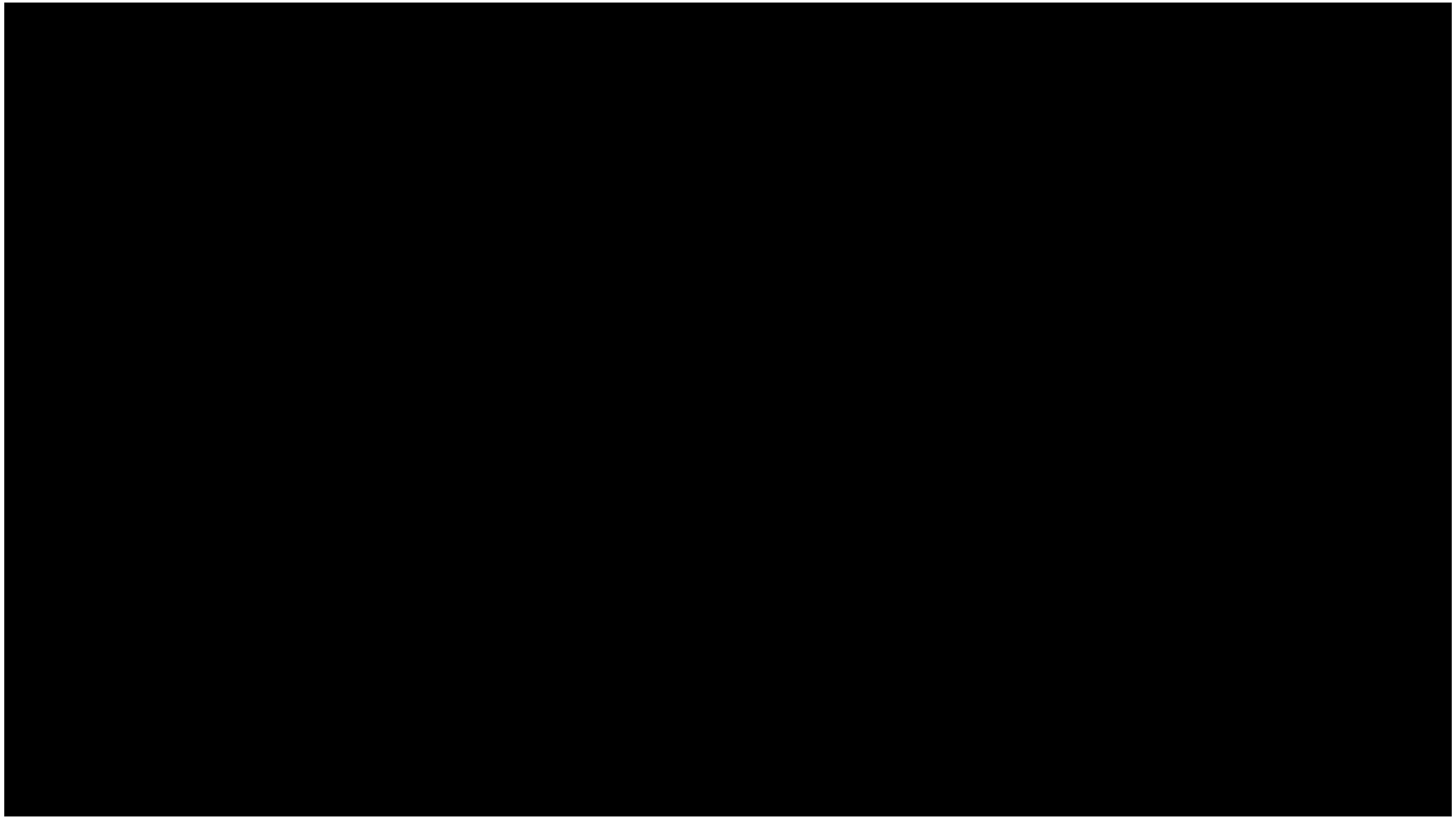
— Office hours **today** (5-7pm @ ICCS 104), **tomorrow** (6-7pm @ ICCS 104)

Today's "**fun**" Example: AI Generated Portrait

Sold in 2018 for $432,500 at British auction house



Today's “**fun**” Example: Sunspring



Lecture 15: Re-Cap

— We motivated SIFT for identifying locally distinct keypoints in an image (**detection**)

— SIFT features (**description**) are invariant to translation, rotation, and scale; robust to 3D pose and illumination

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

Lecture 15: Re-Cap

Keypoint is an image location at which a descriptor is computed

— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.

Lecture 15: Re-Cap

Keypoint is an image location at which a descriptor is computed

— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.

Lecture 15: Re-Cap

Keypoint is an image location at which a descriptor is computed

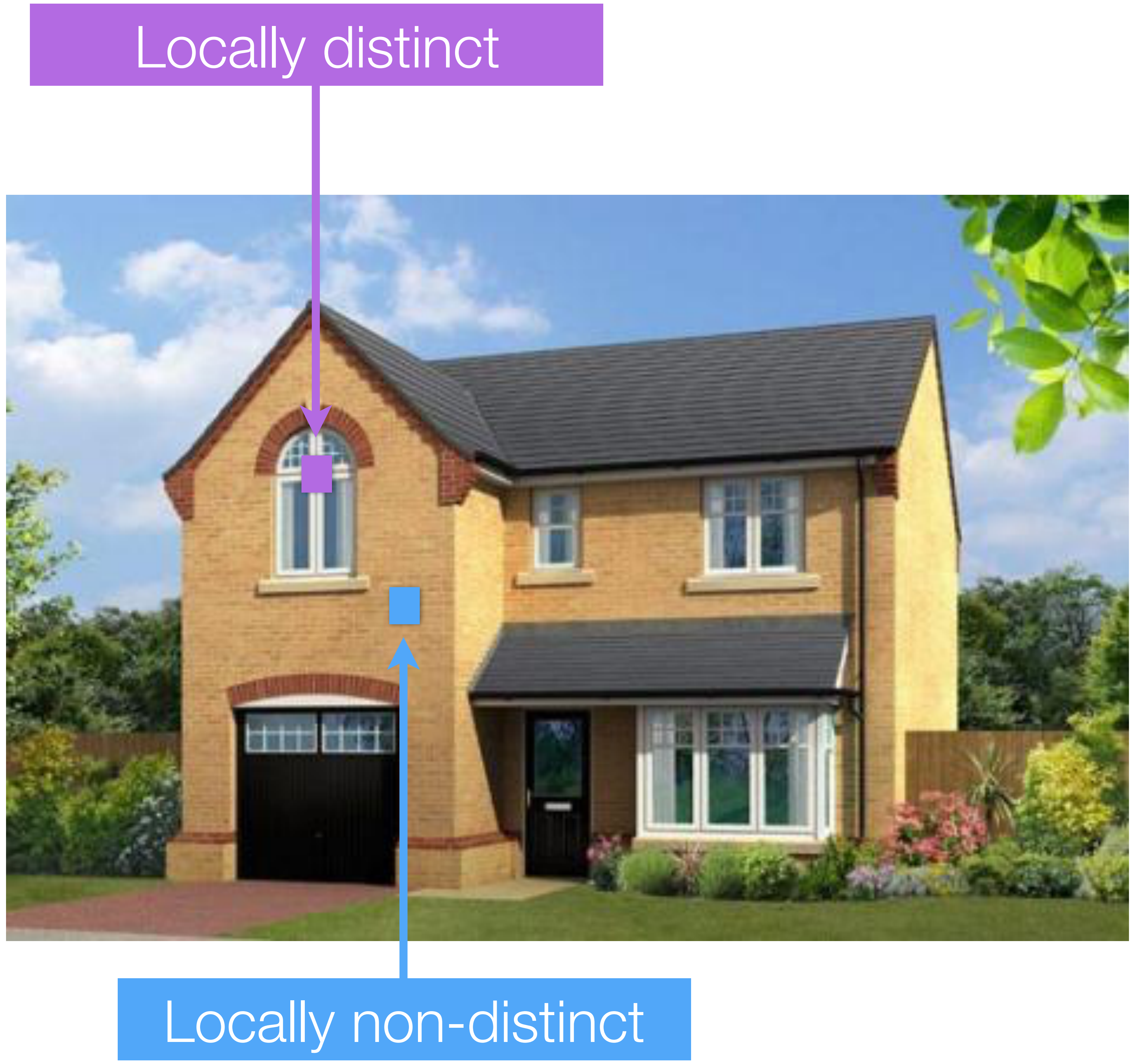
— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.



Lecture 15: Re-Cap

Keypoint is an image location at which a descriptor is computed

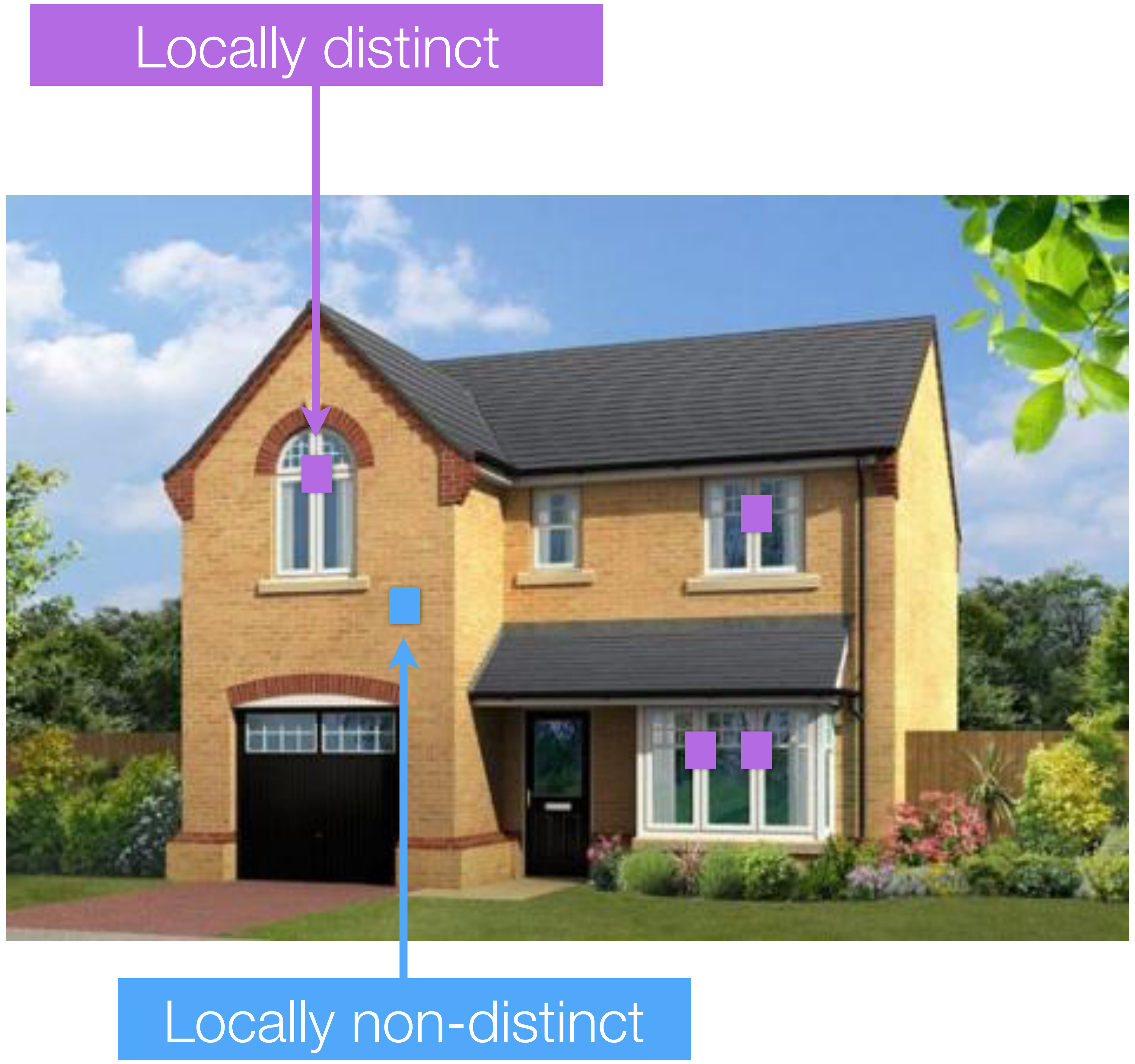
— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.



Lecture 15: Re-Cap

Keypoint is an image location at which a descriptor is computed

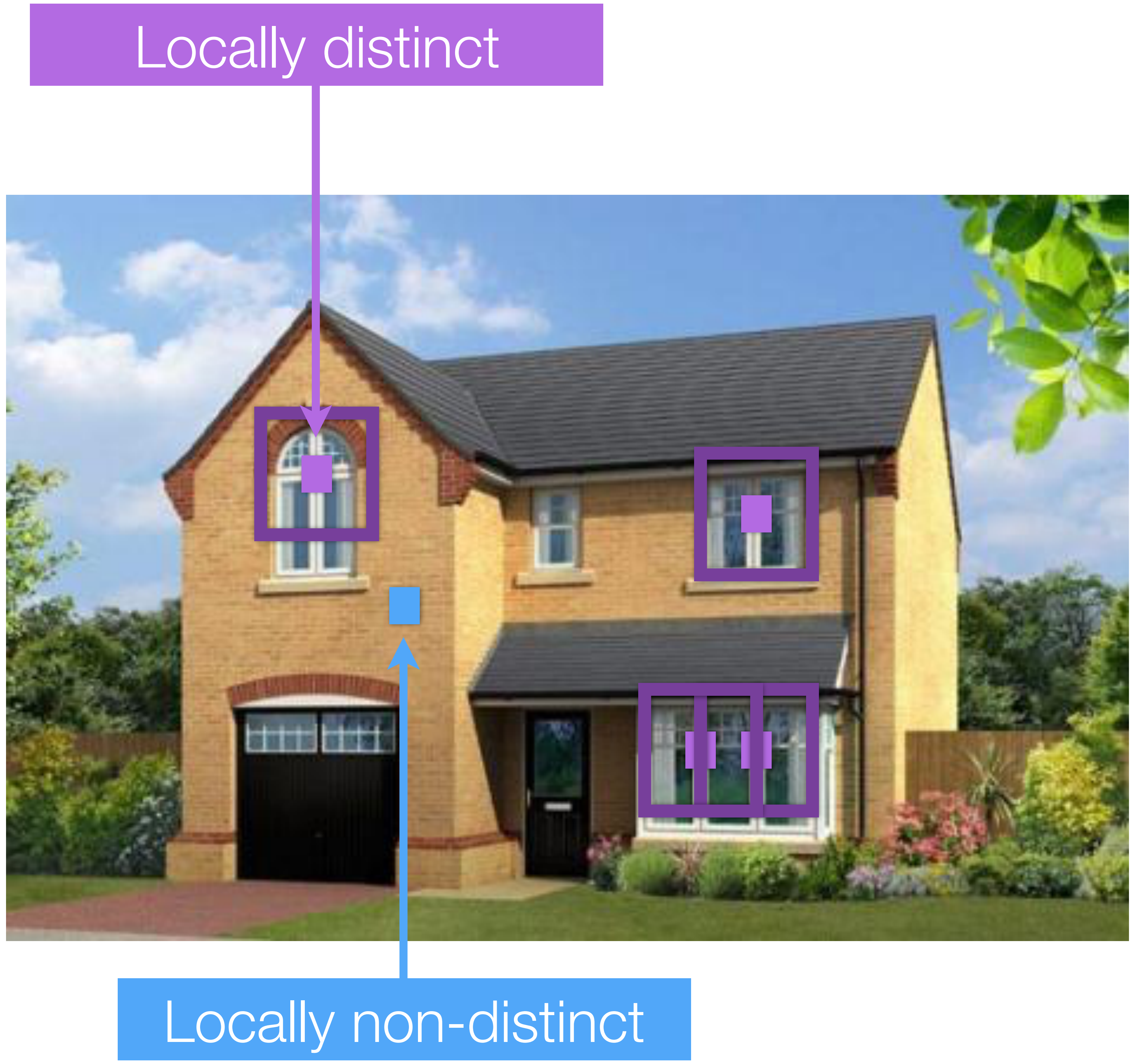
— Locally distinct points

— Easily localizable and identifiable

The feature **descriptor** summarizes the local structure around the key point

— Allows us to (hopefully) unique matching of keypoints in presence of object pose variations, image and photometric deformations

Note, for repetitive structure this would still not give us unique matches.



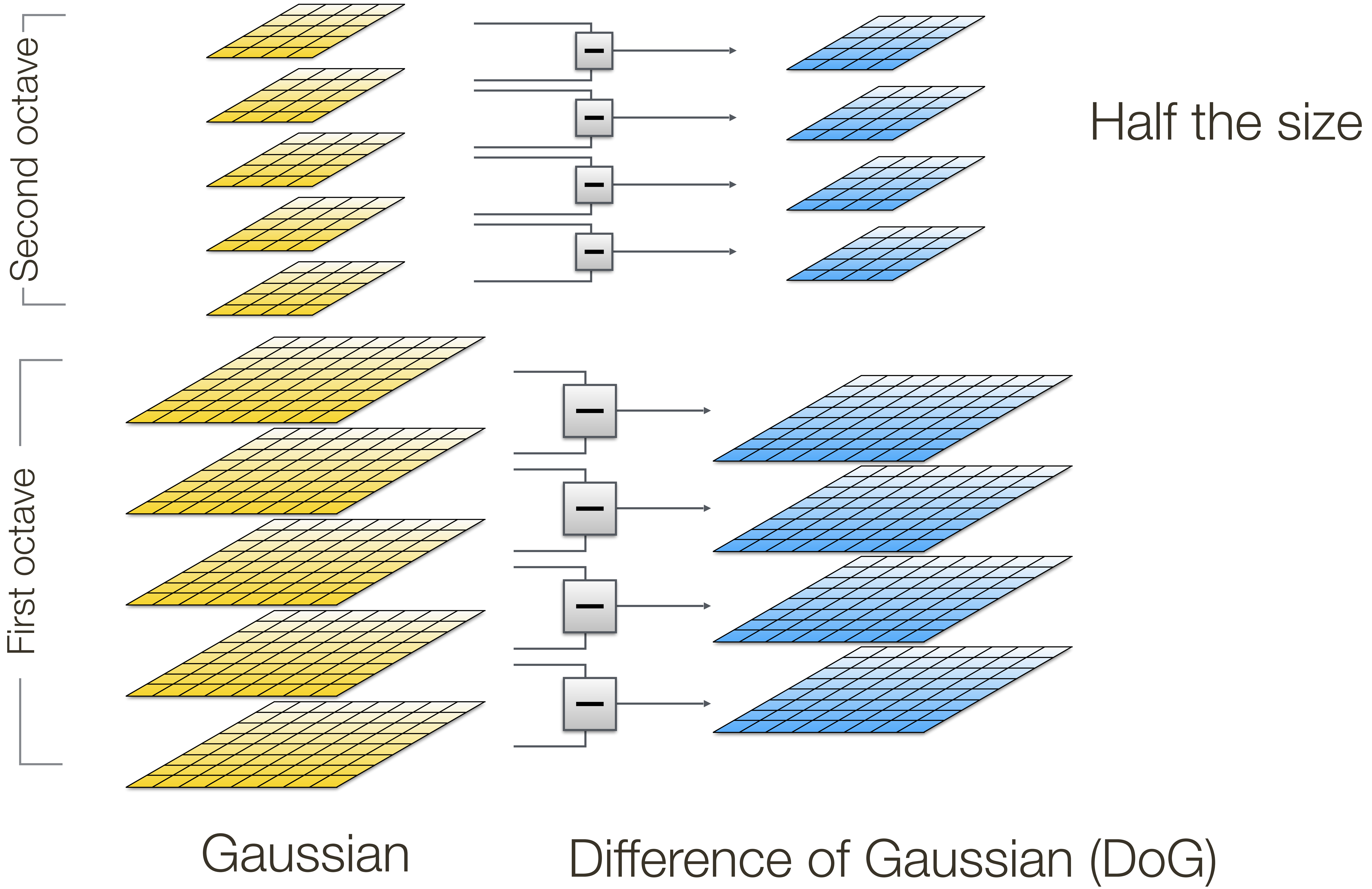
Lecture 18: Re-Cap

— We motivated SIFT for identifying locally distinct keypoints in an image (**detection**)

— SIFT features (**description**) are invariant to translation, rotation, and scale; robust to 3D pose and illumination

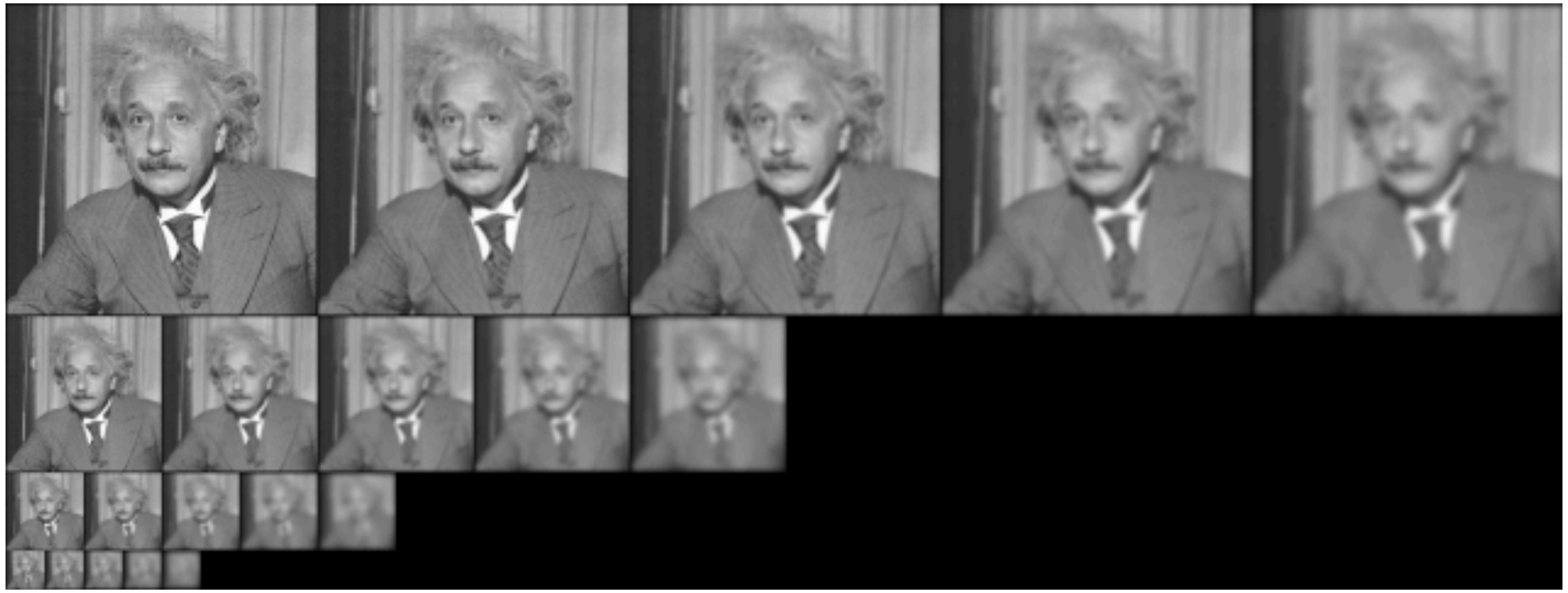
1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

1. Multi-scale Extrema Detection



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

1. Multi-scale Extrema Detection



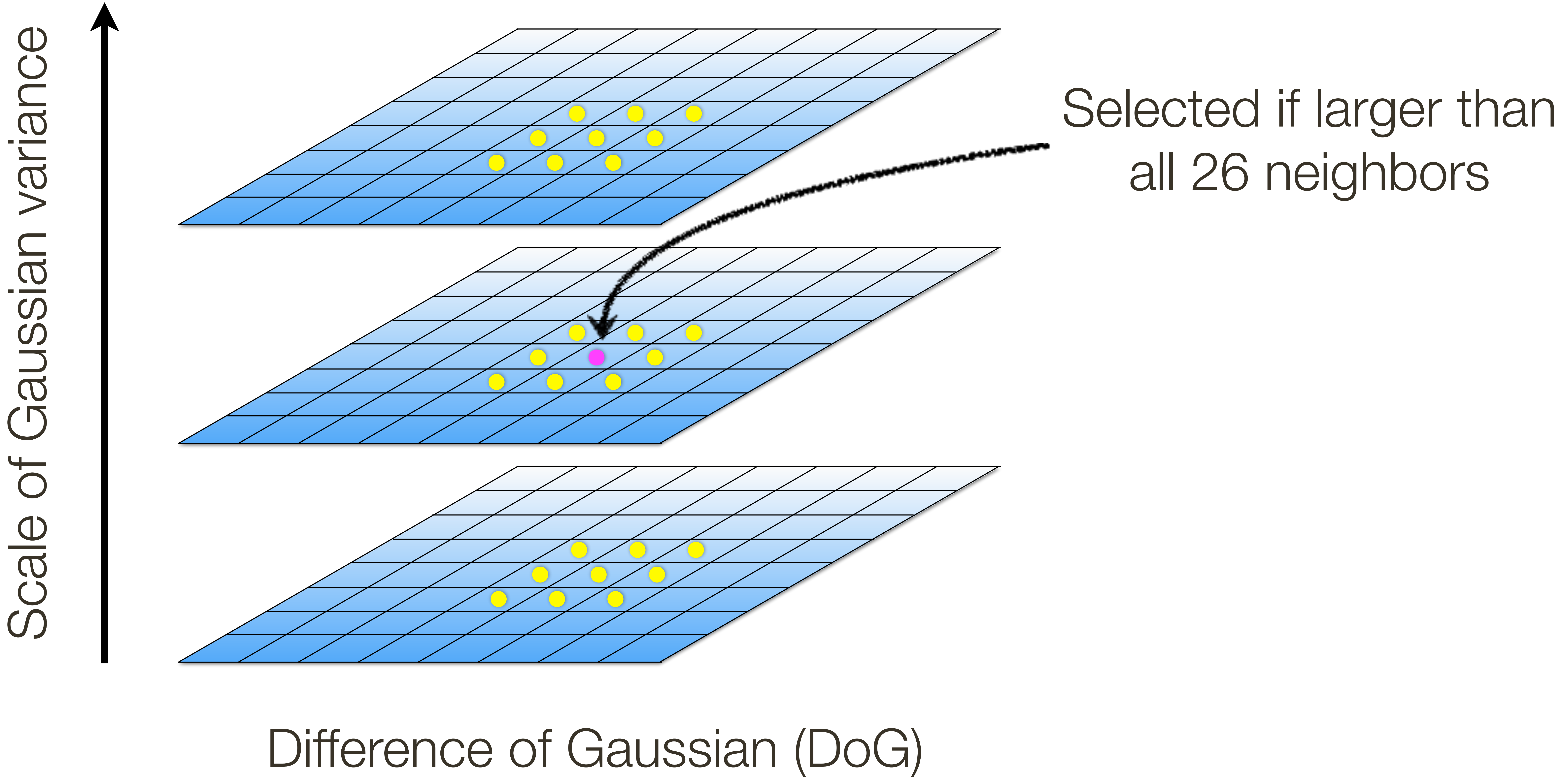
Gaussian

Laplacian

1. Multi-scale Extrema Detection

Detect maxima and minima of Difference of Gaussian in scale space

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



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

1. Multi-scale Extrema Detection

Detect maxima and minima of Difference of Gaussian in scale space

— Responds to blob-line and corner-like structures

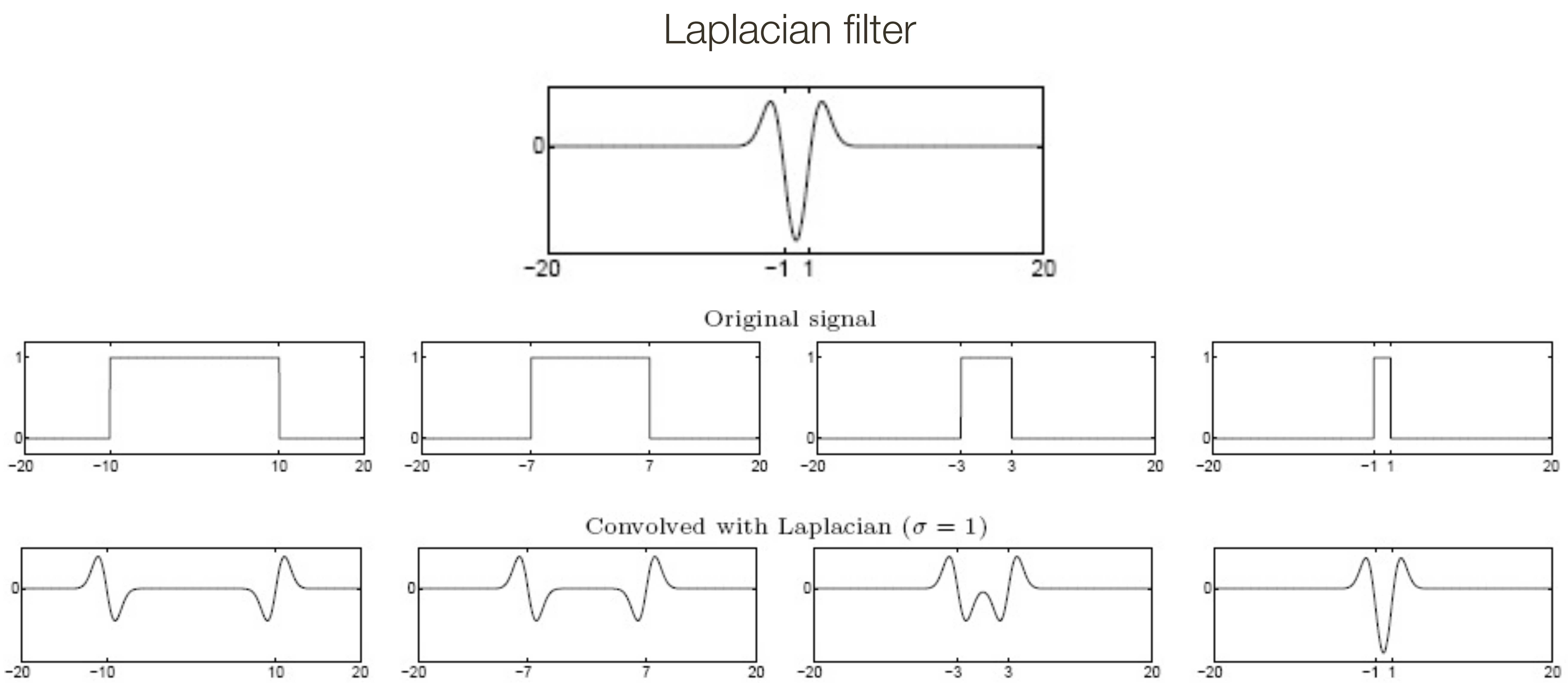
— Could also give strong responses at edges

1. Multi-scale Extrema Detection

Detect maxima and minima of Difference of Gaussian in scale space

— Responds to blob-line and corner-like structures

— Could also give strong responses at edges



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?

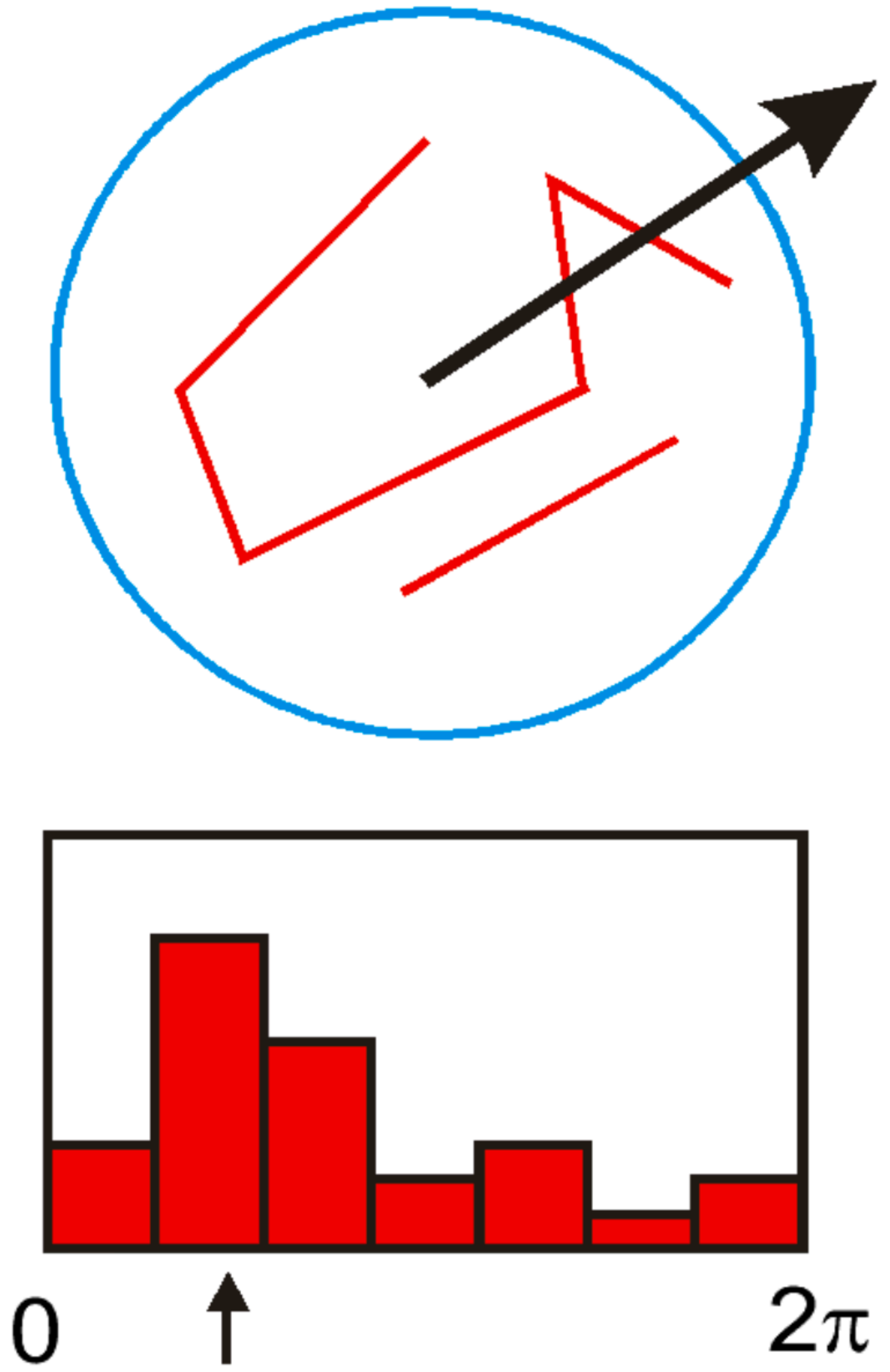
$$C = \begin{bmatrix} \sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\ \sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y \end{bmatrix}$$

3. Orientation Assignment

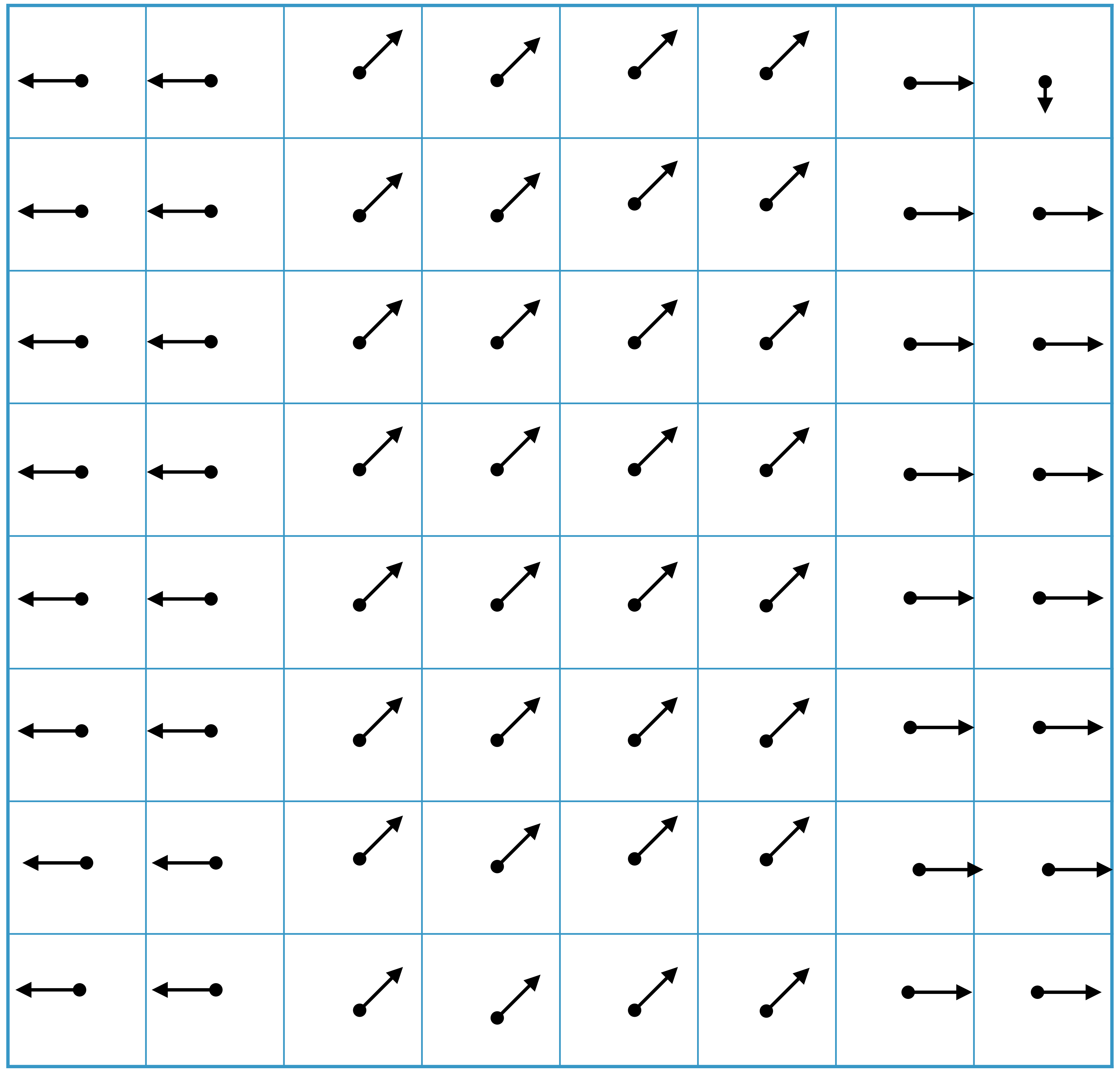
— Create **histogram** of local gradient directions computed at selected scale

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

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

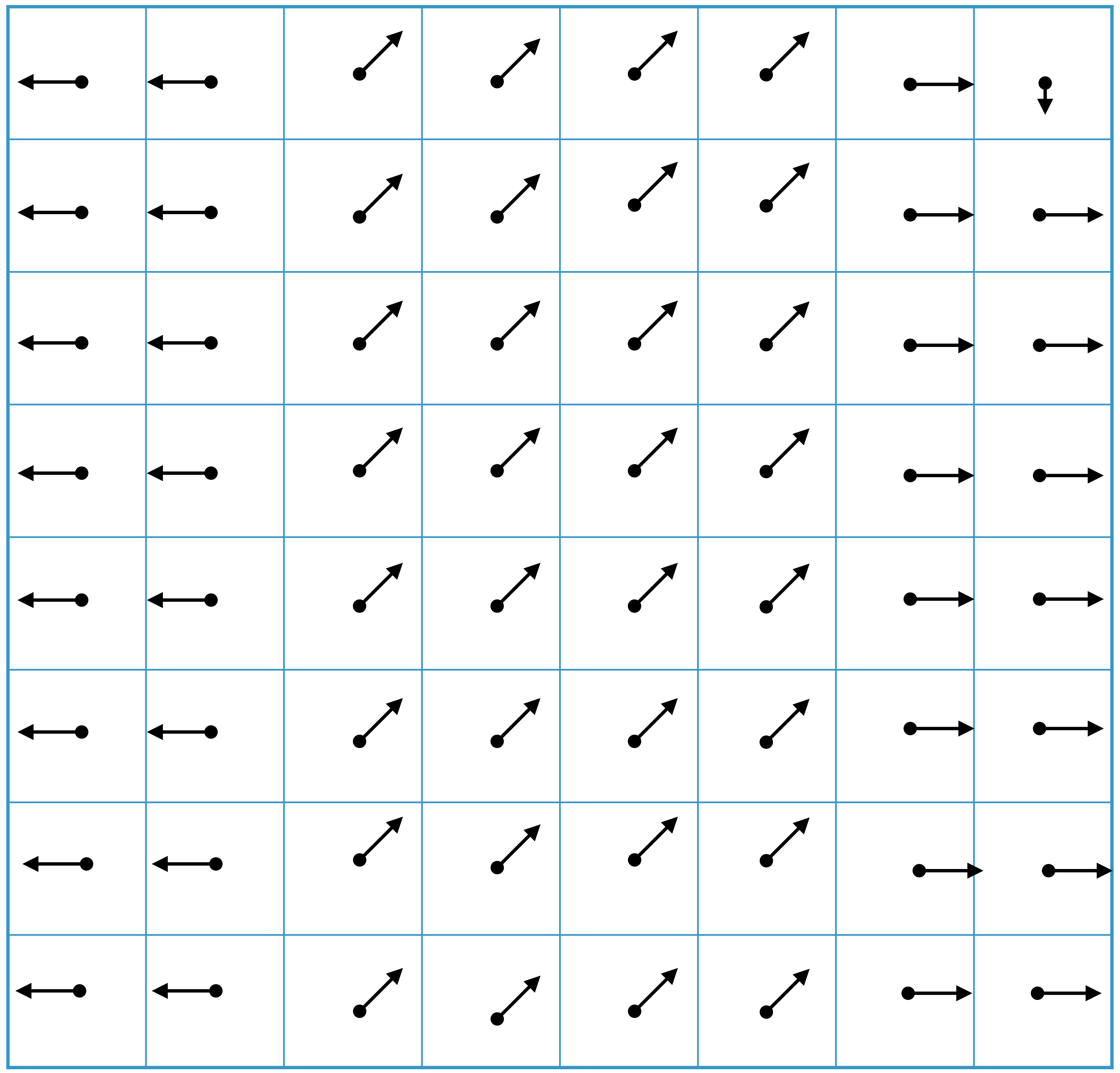


3. Orientation Assignment

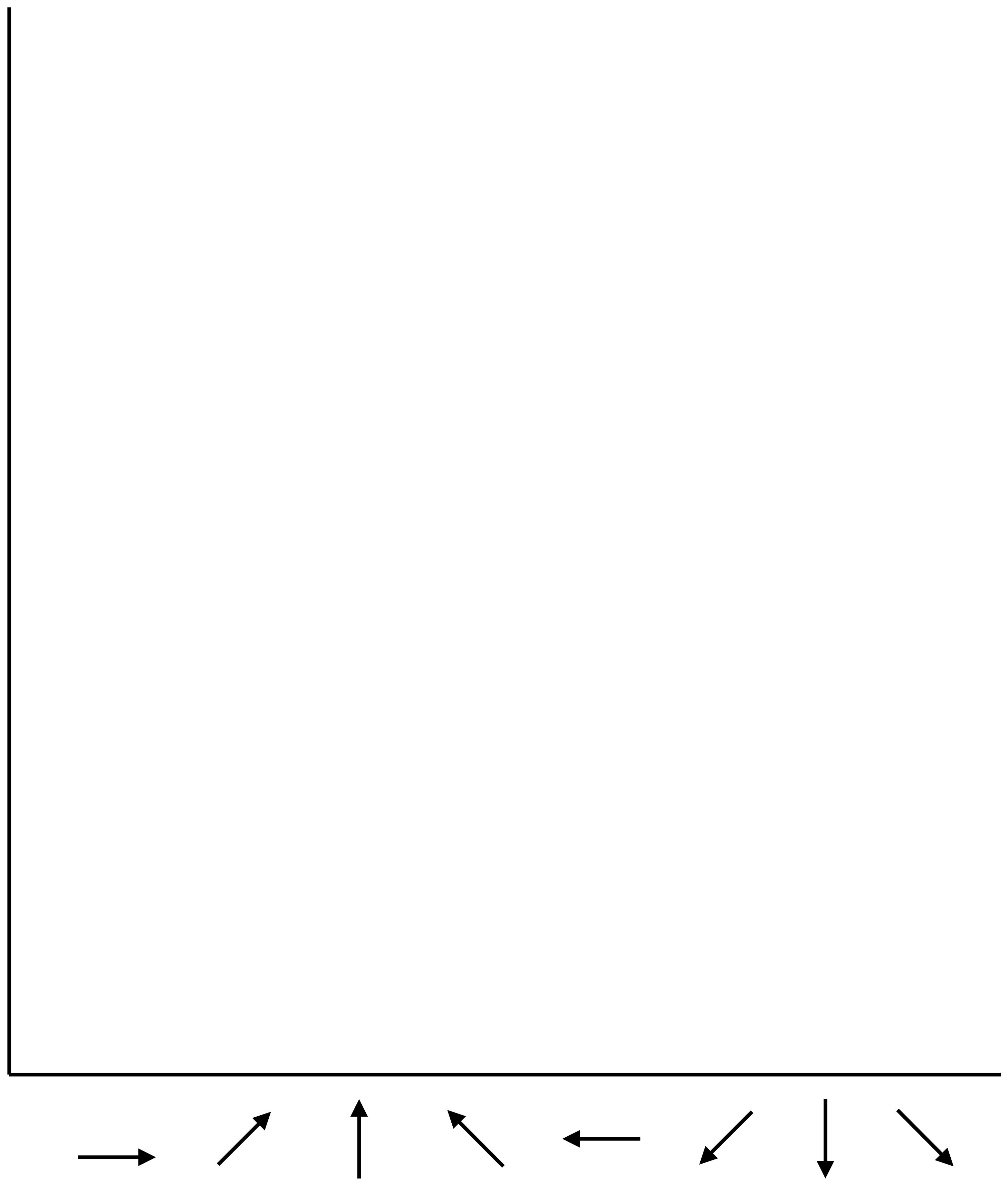


Arrows illustrate **gradient orientation** (direction)
and **gradient magnitude** (arrow length)

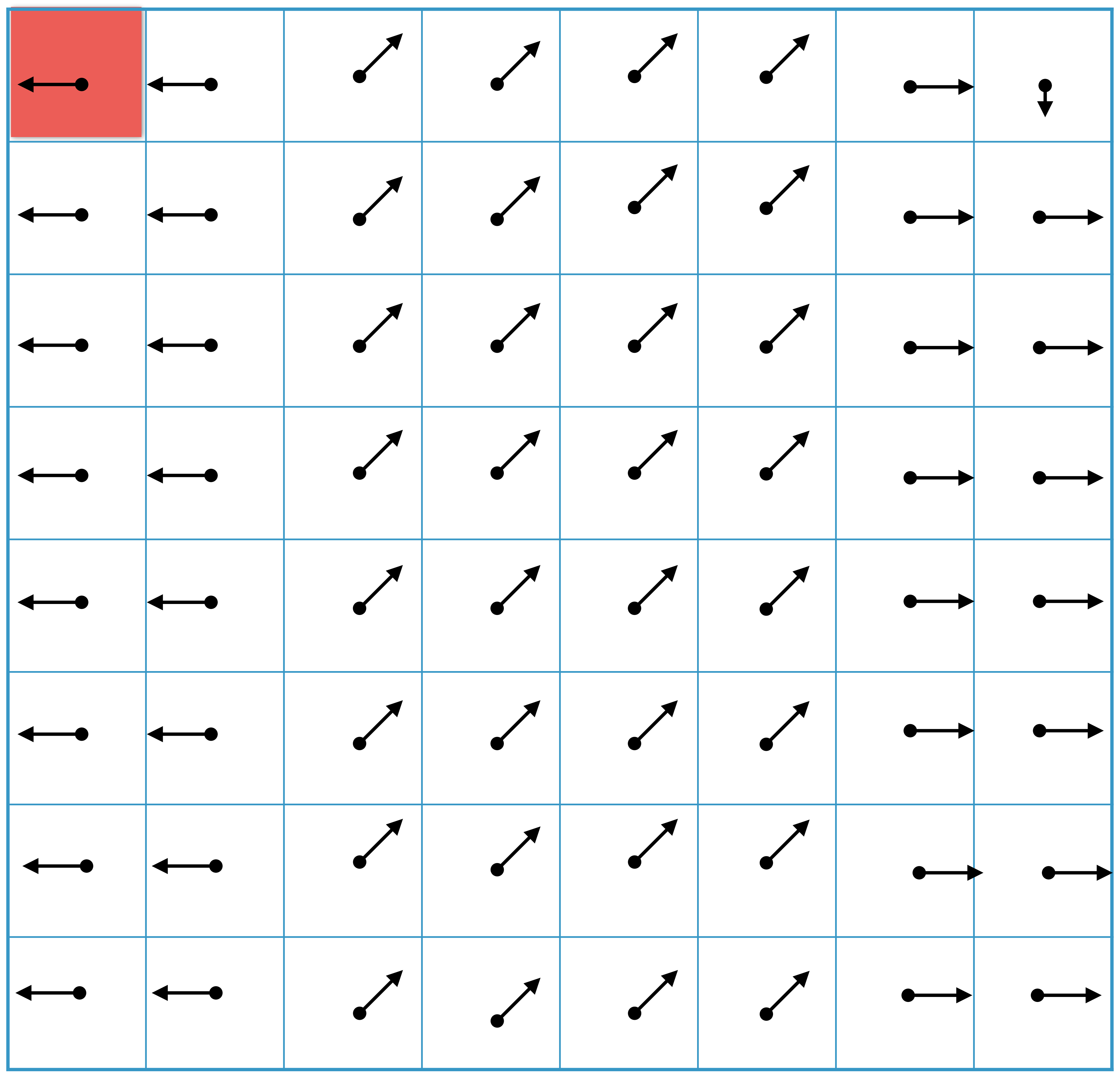
3. Orientation Assignment



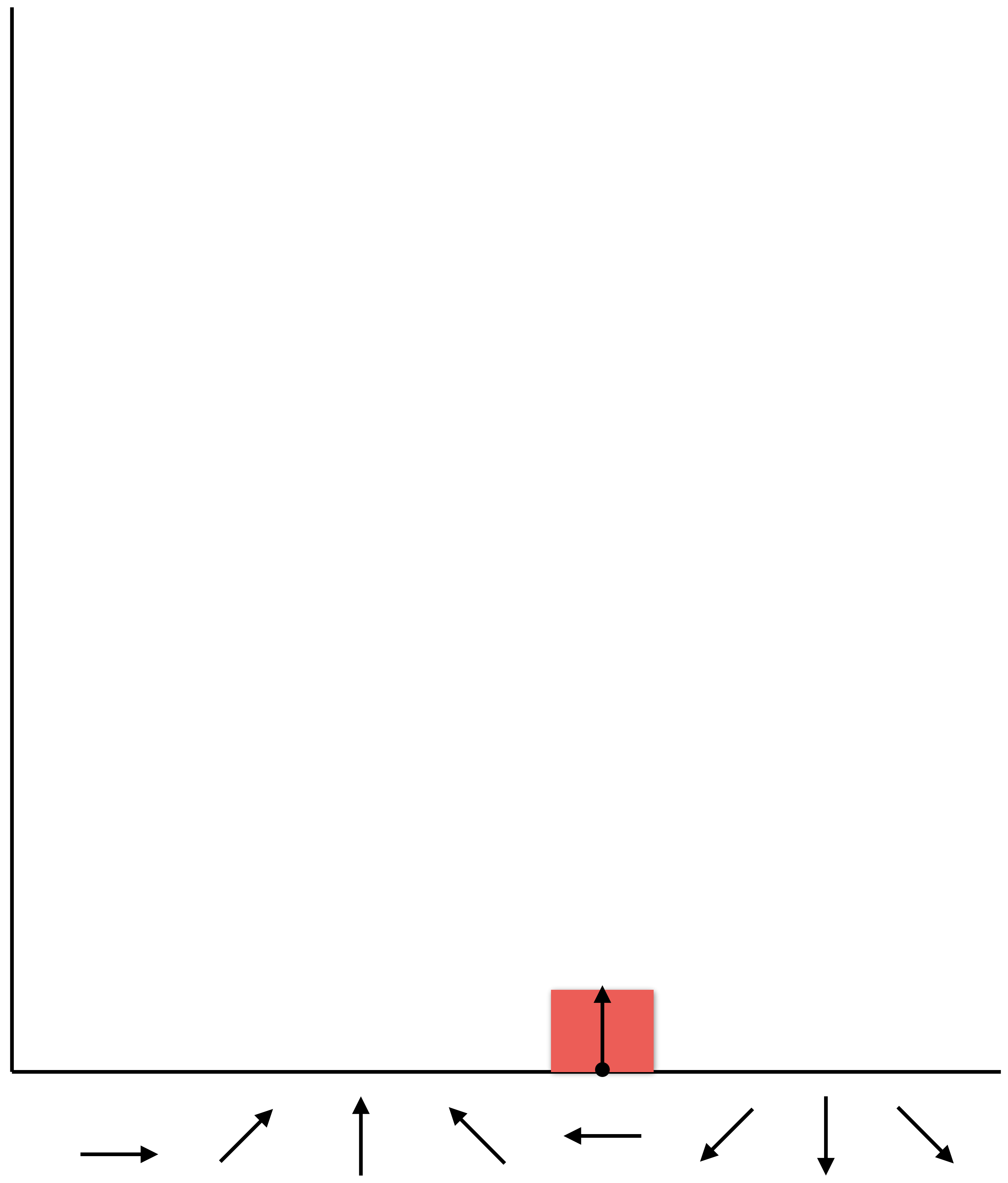
Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)



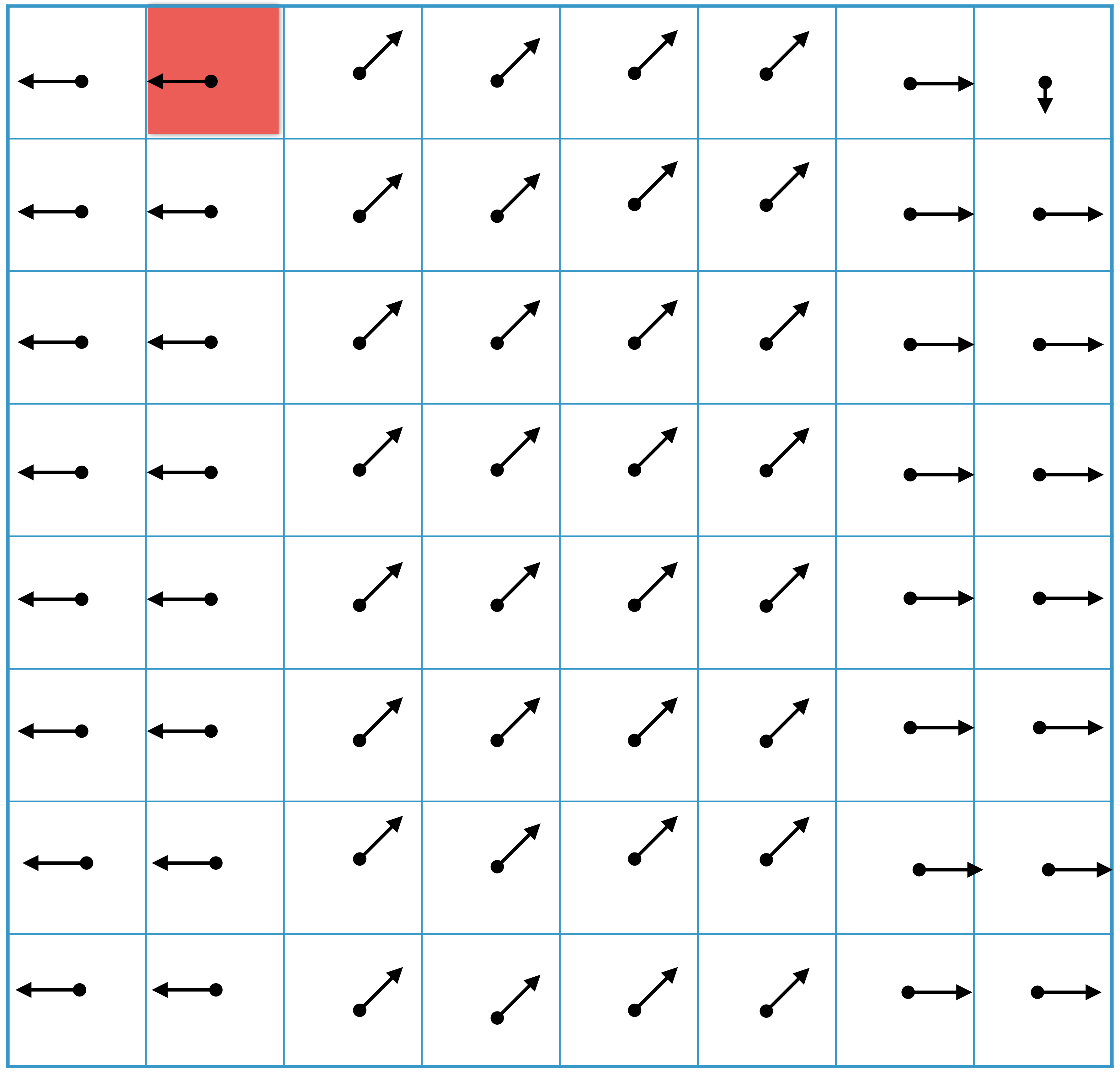
3. Orientation Assignment

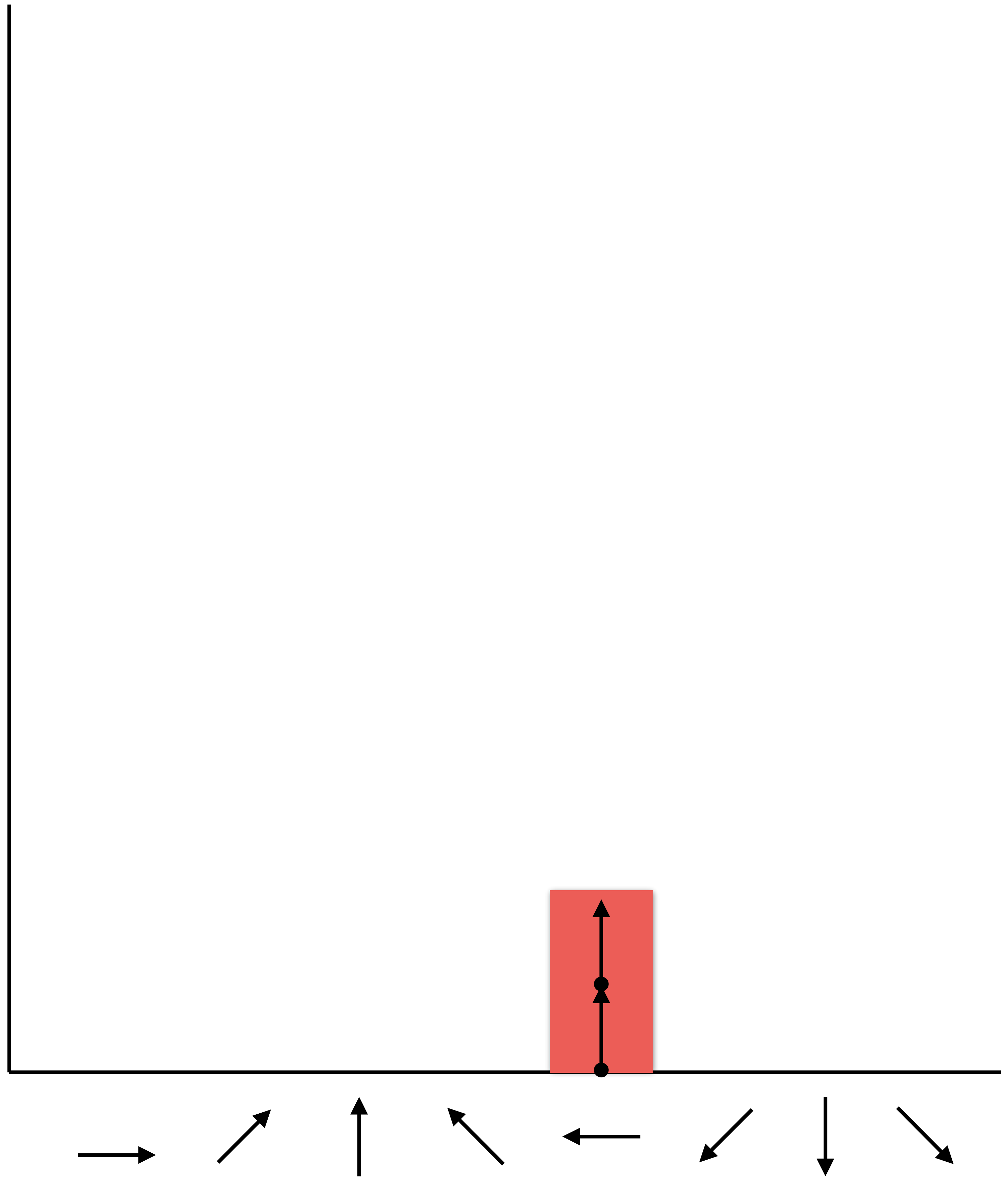


Arrows illustrate **gradient orientation** (direction)
and **gradient magnitude** (arrow length)



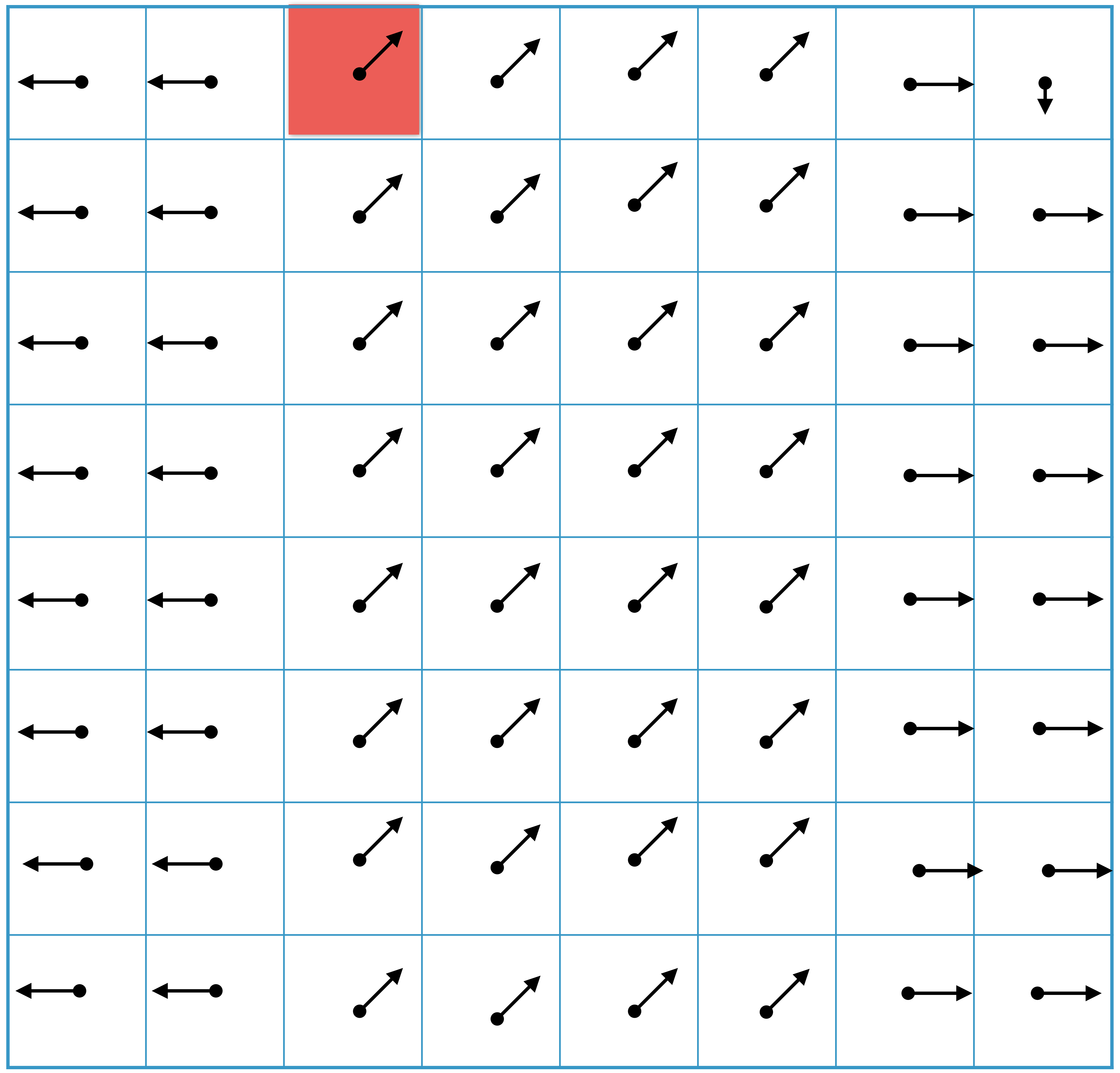
3. Orientation Assignment

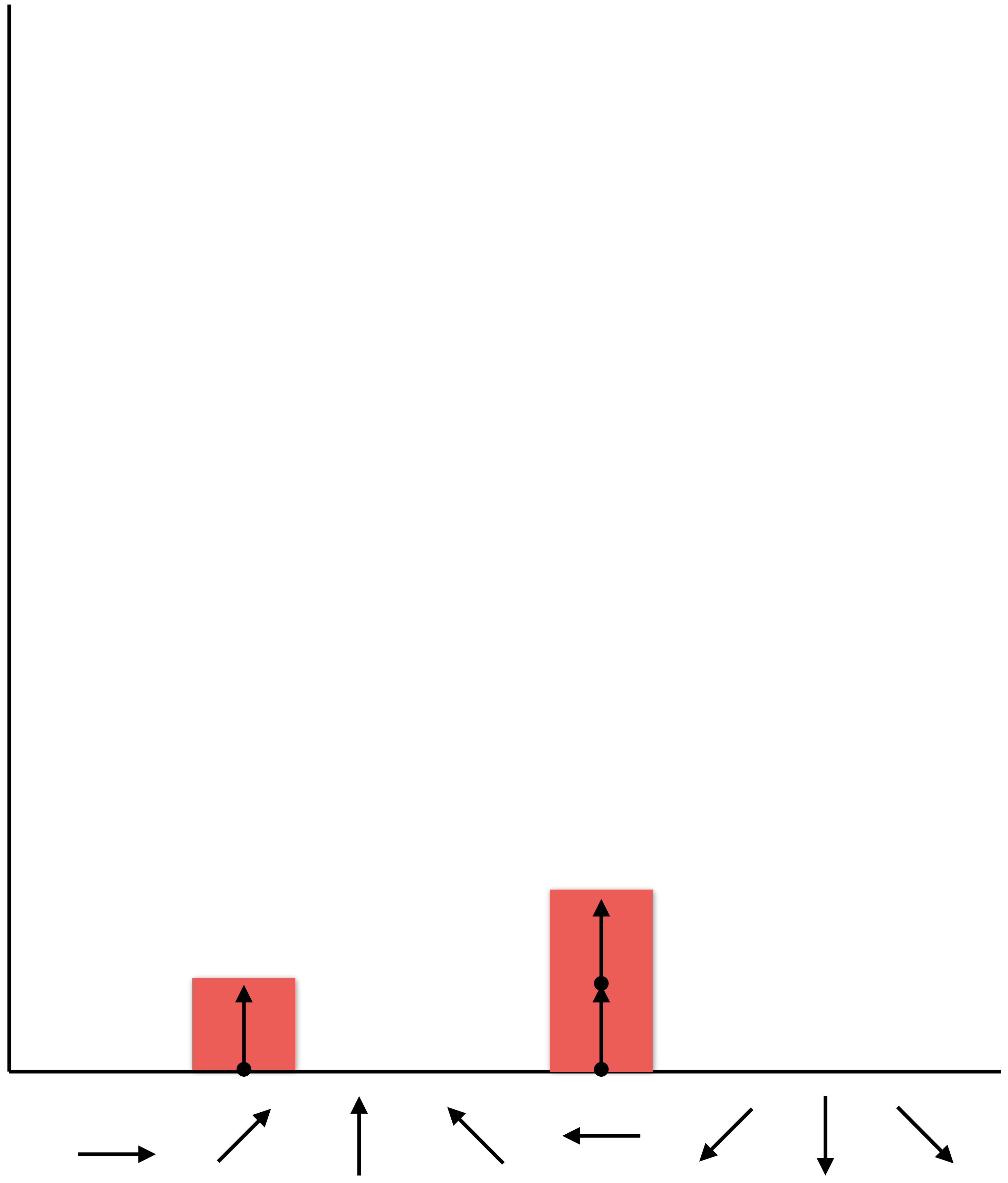




Arrows illustrate **gradient orientation** (direction)
and **gradient magnitude** (arrow length)

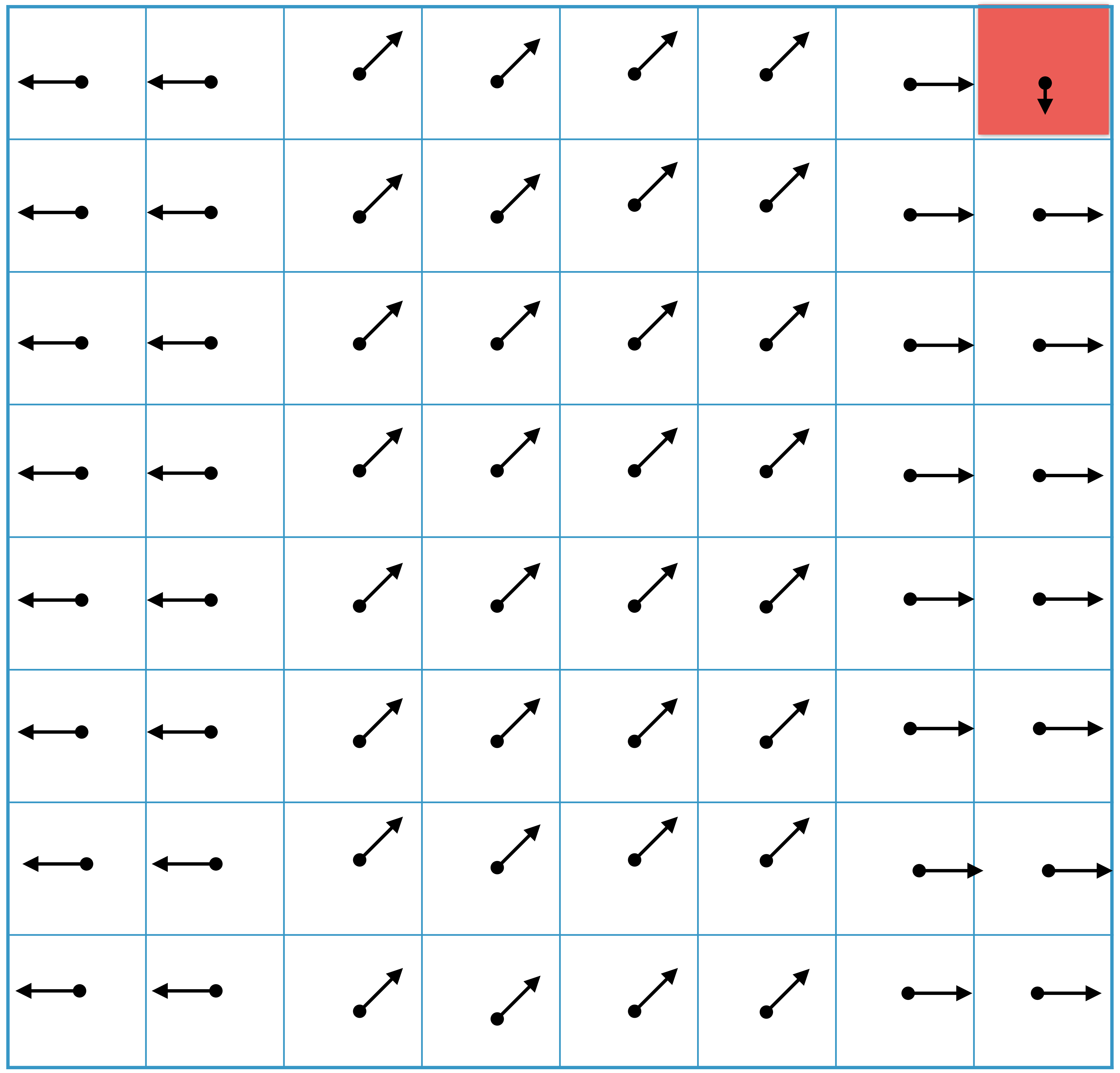
3. Orientation Assignment

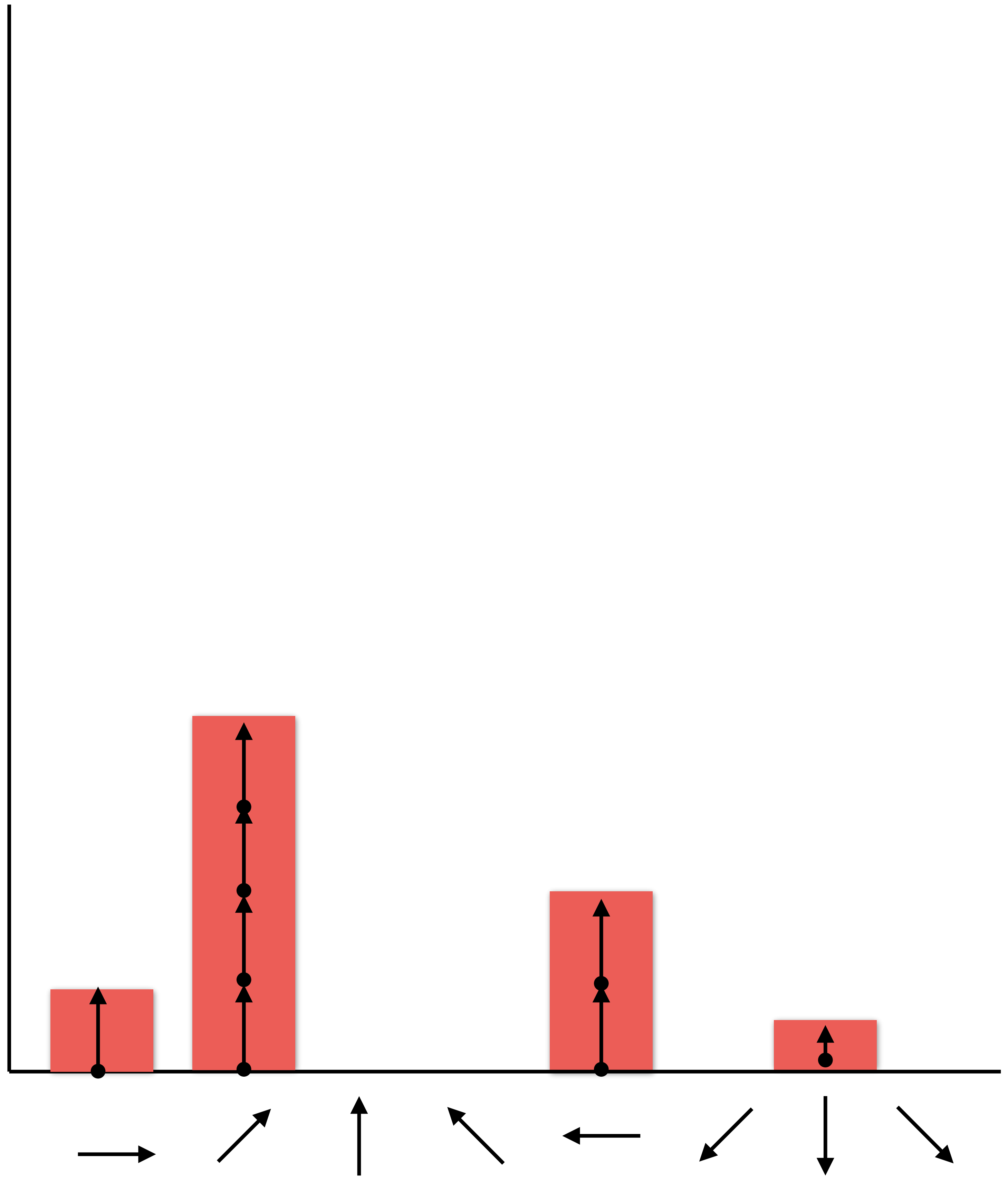




Arrows illustrate **gradient orientation** (direction)
and **gradient magnitude** (arrow length)

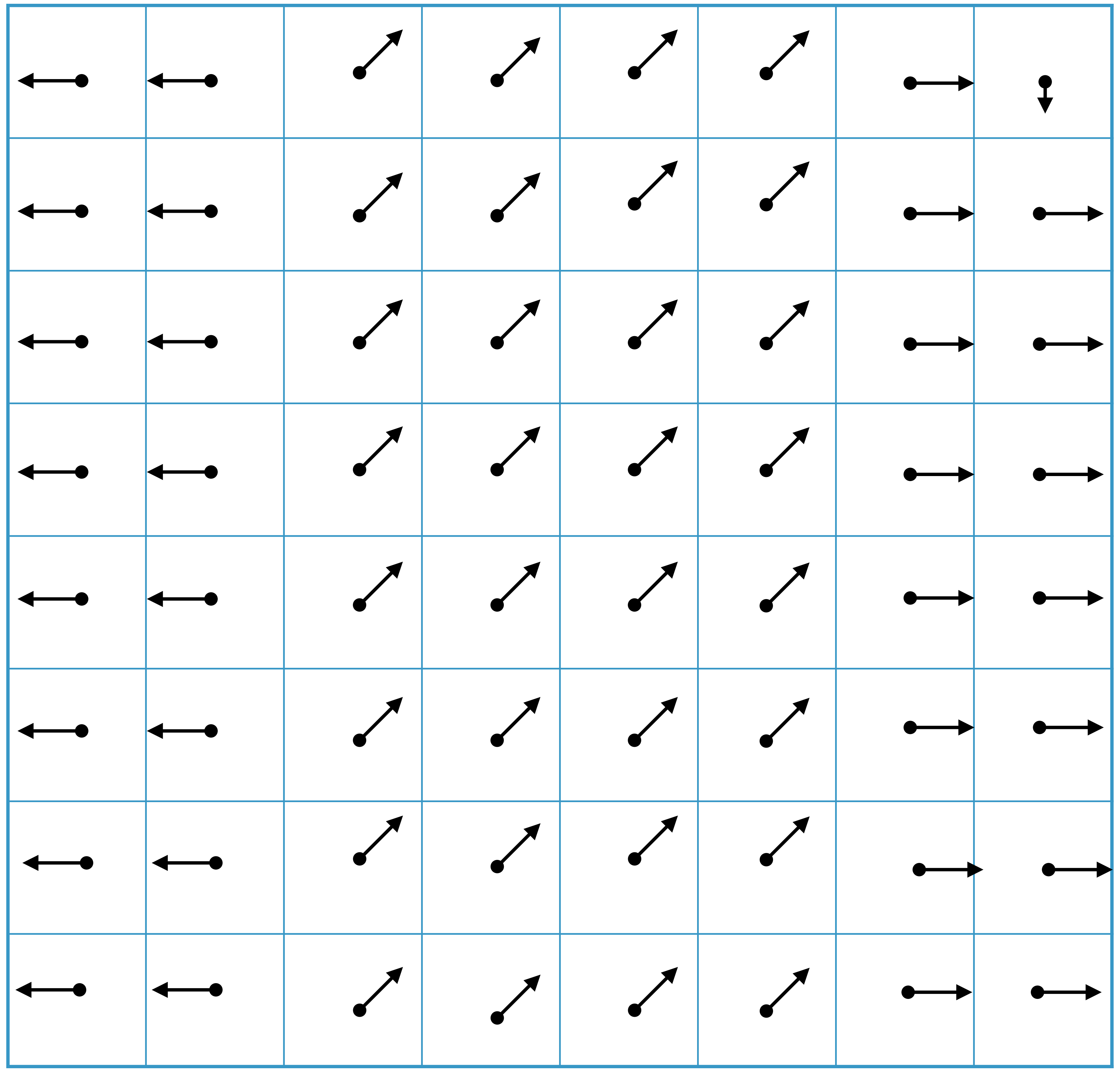
3. Orientation Assignment



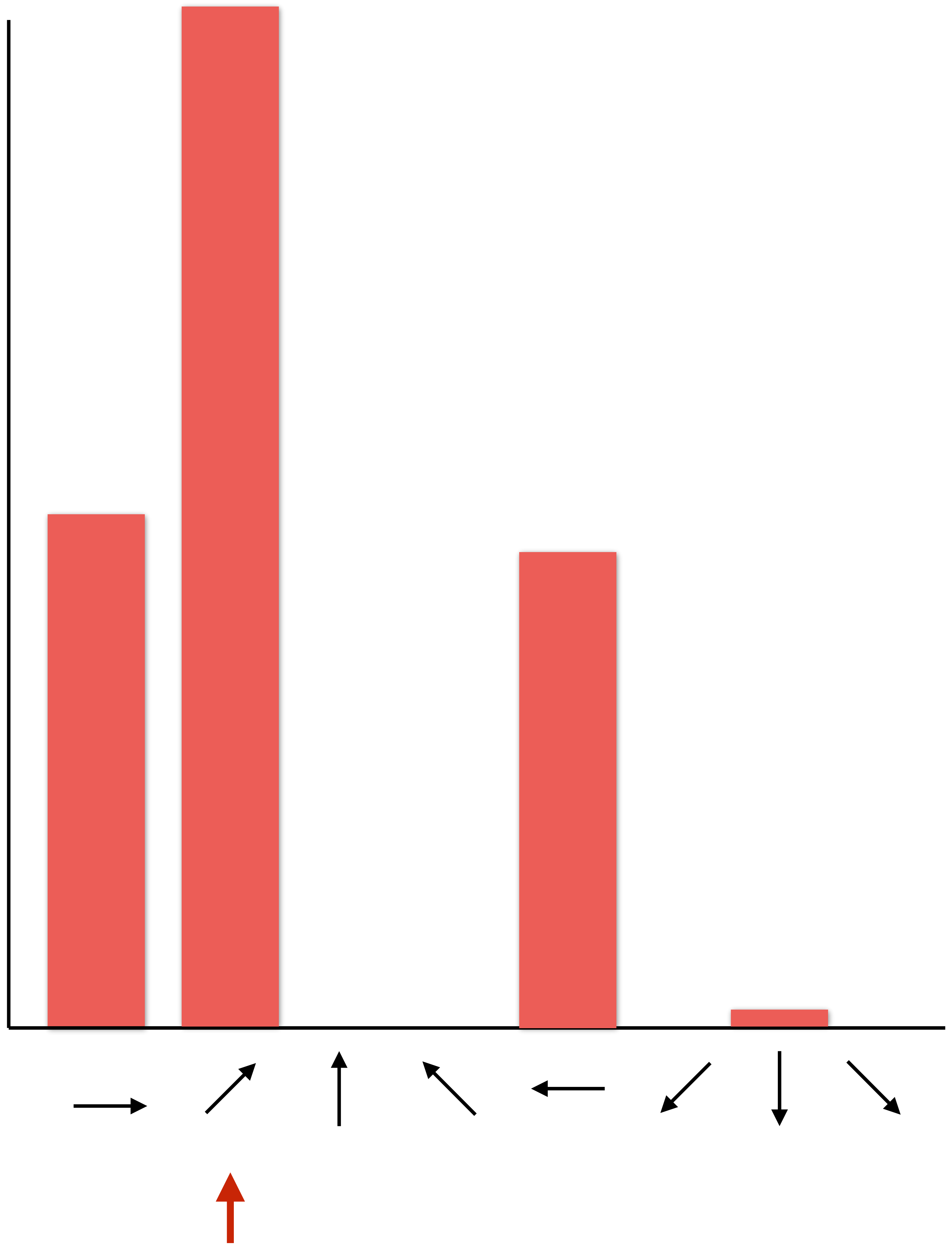


Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

3. Orientation Assignment

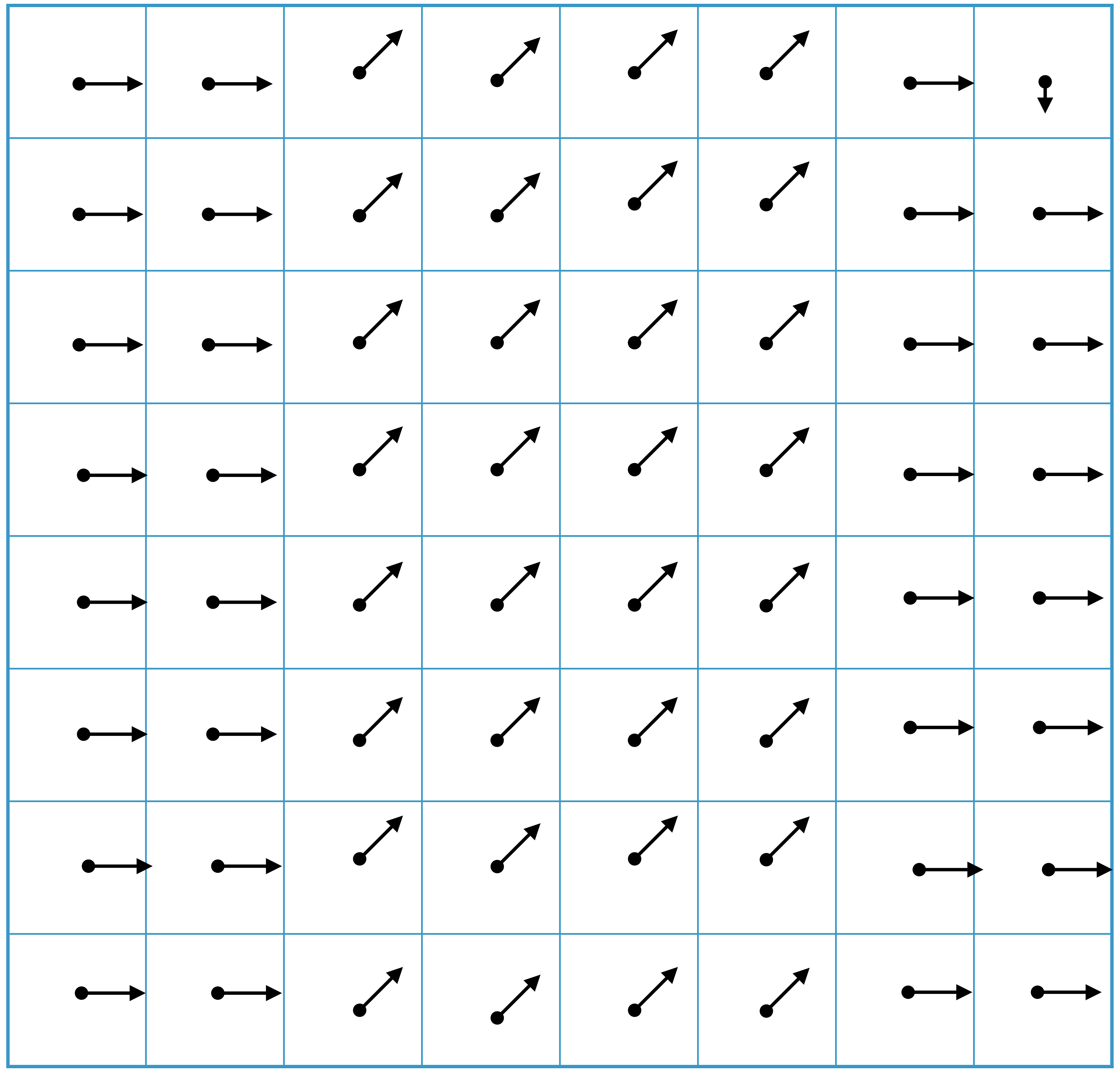


Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

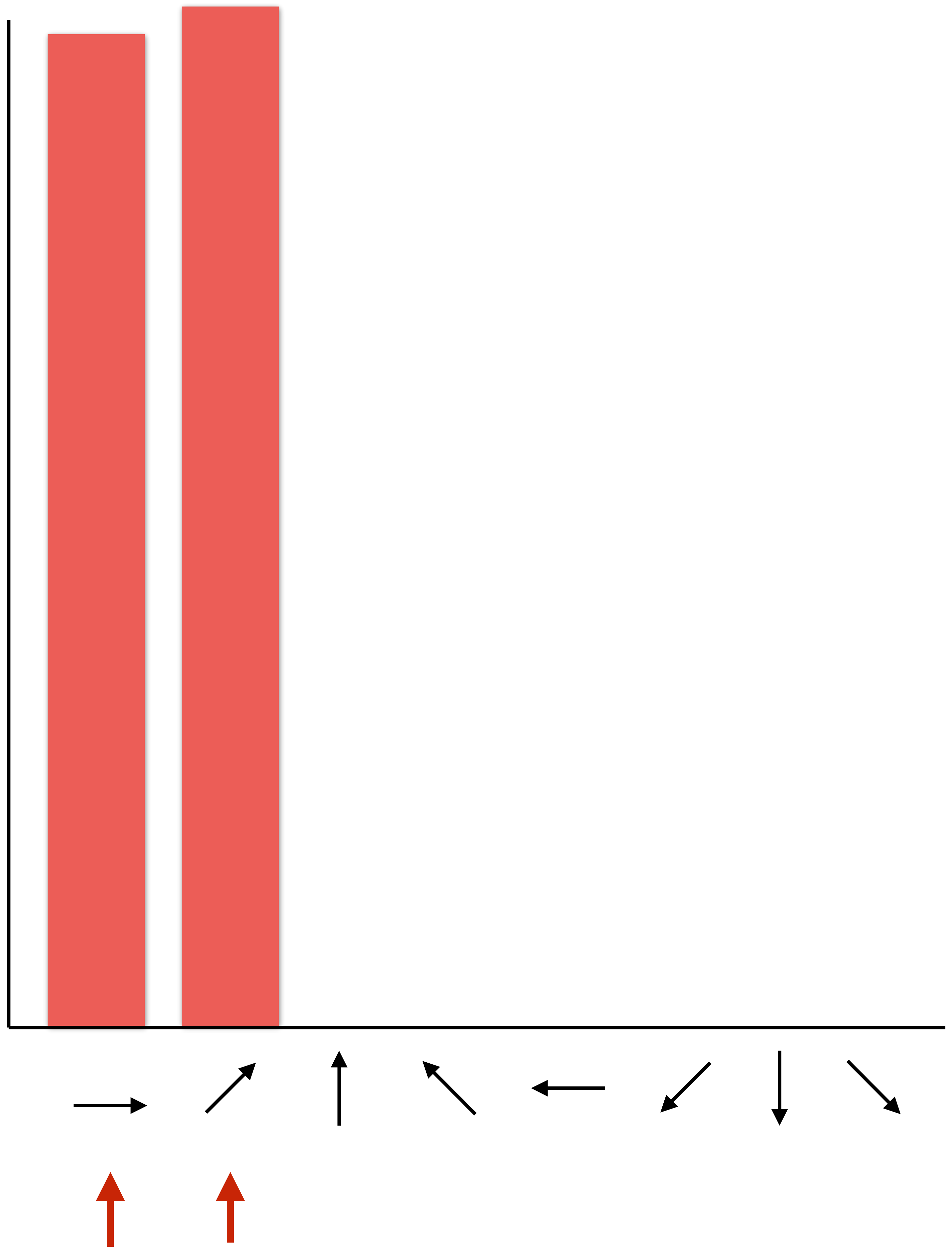


Assigned Orientation

3. Orientation Assignment

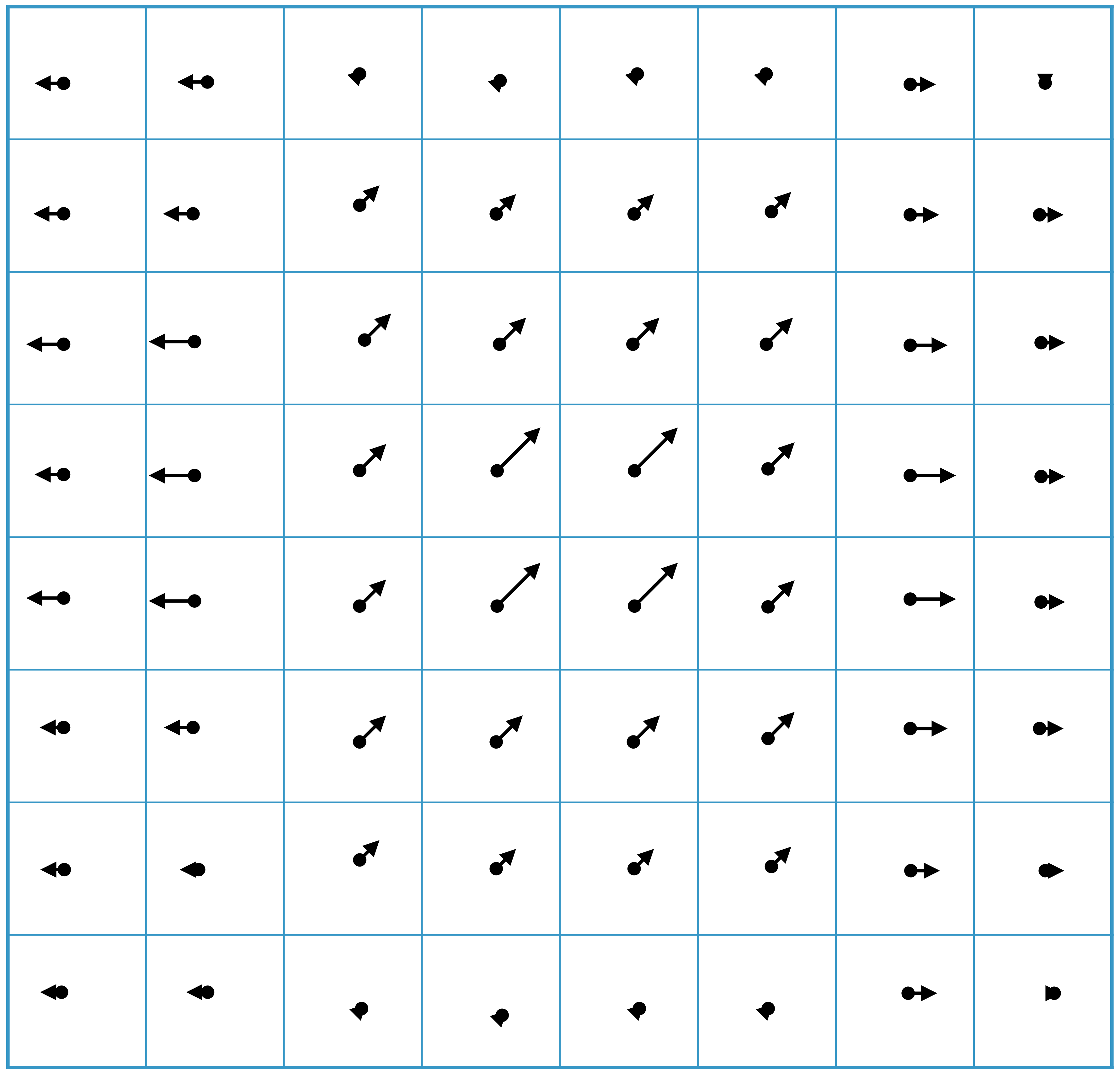


Arrows illustrate **gradient orientation** (direction) and **gradient magnitude** (arrow length)

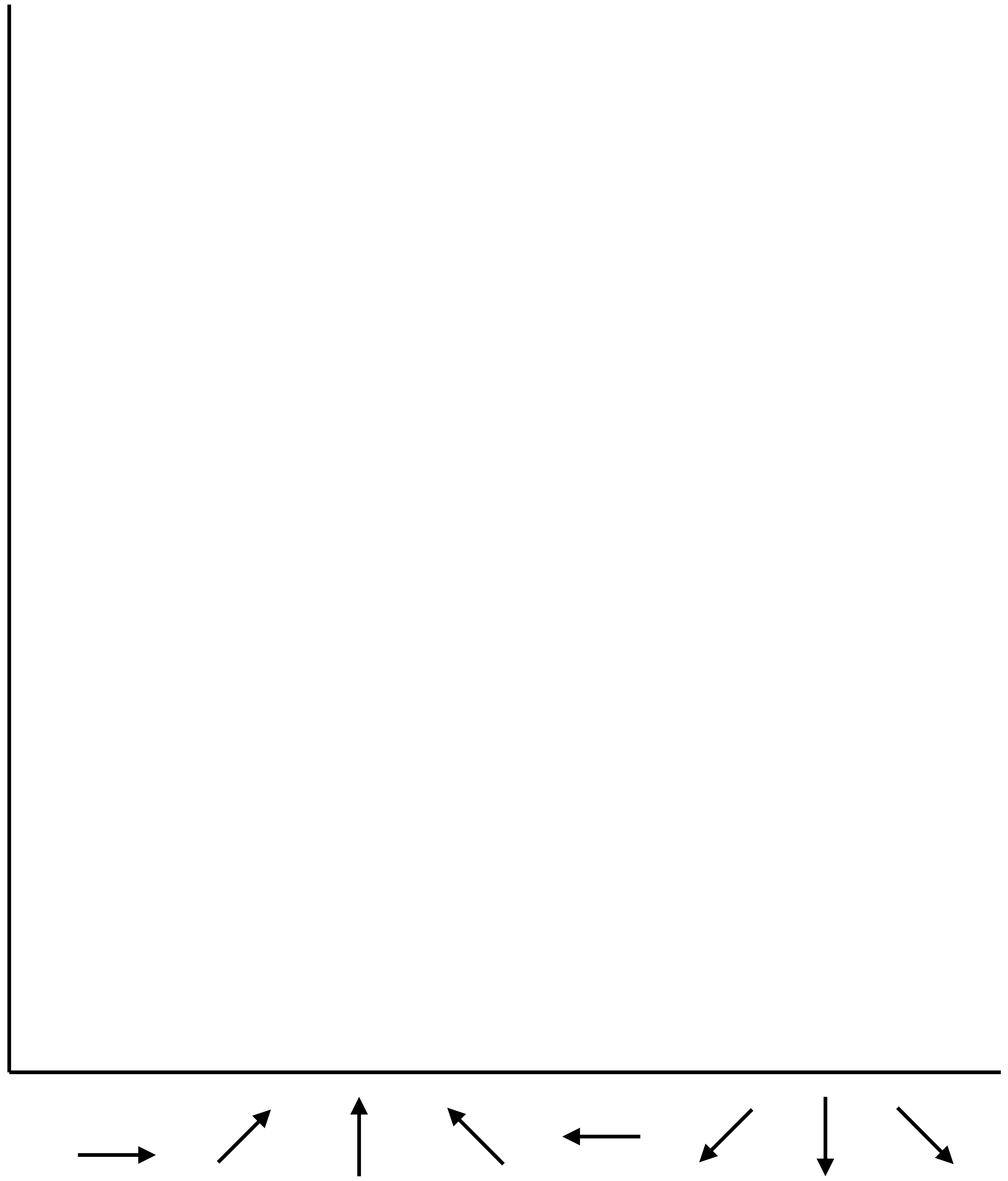


3. Orientation Assignment

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



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

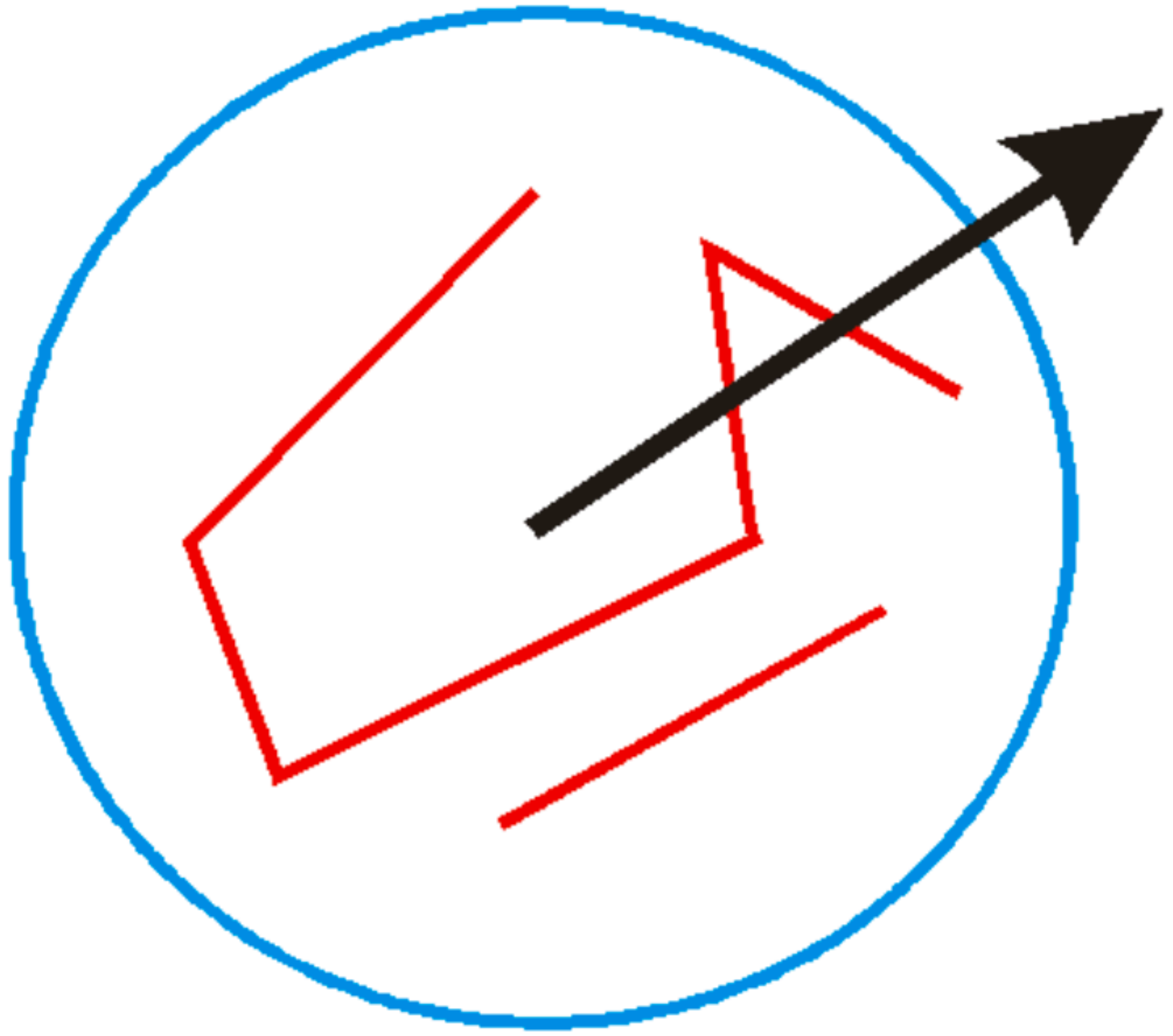


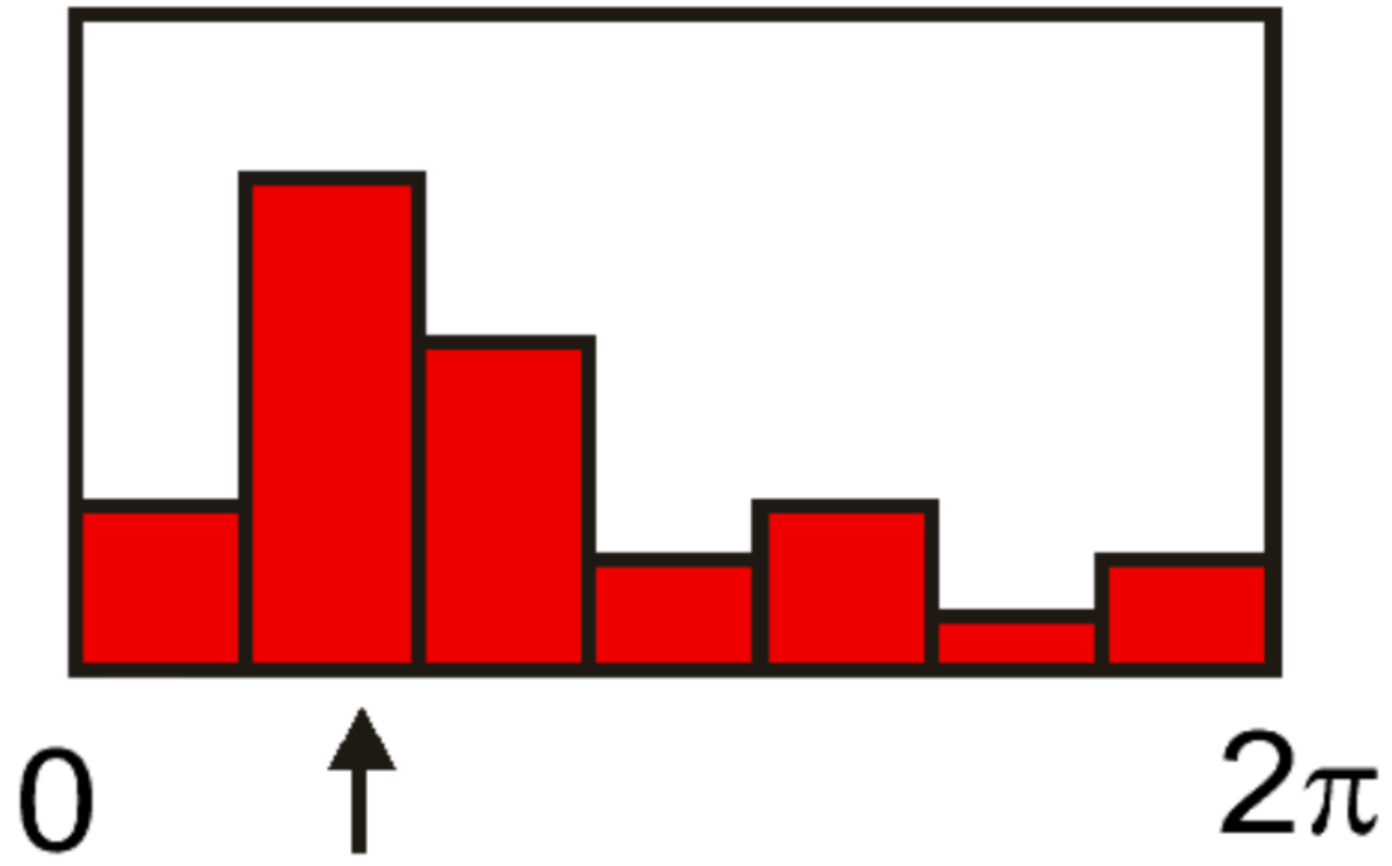
3. Orientation Assignment

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

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

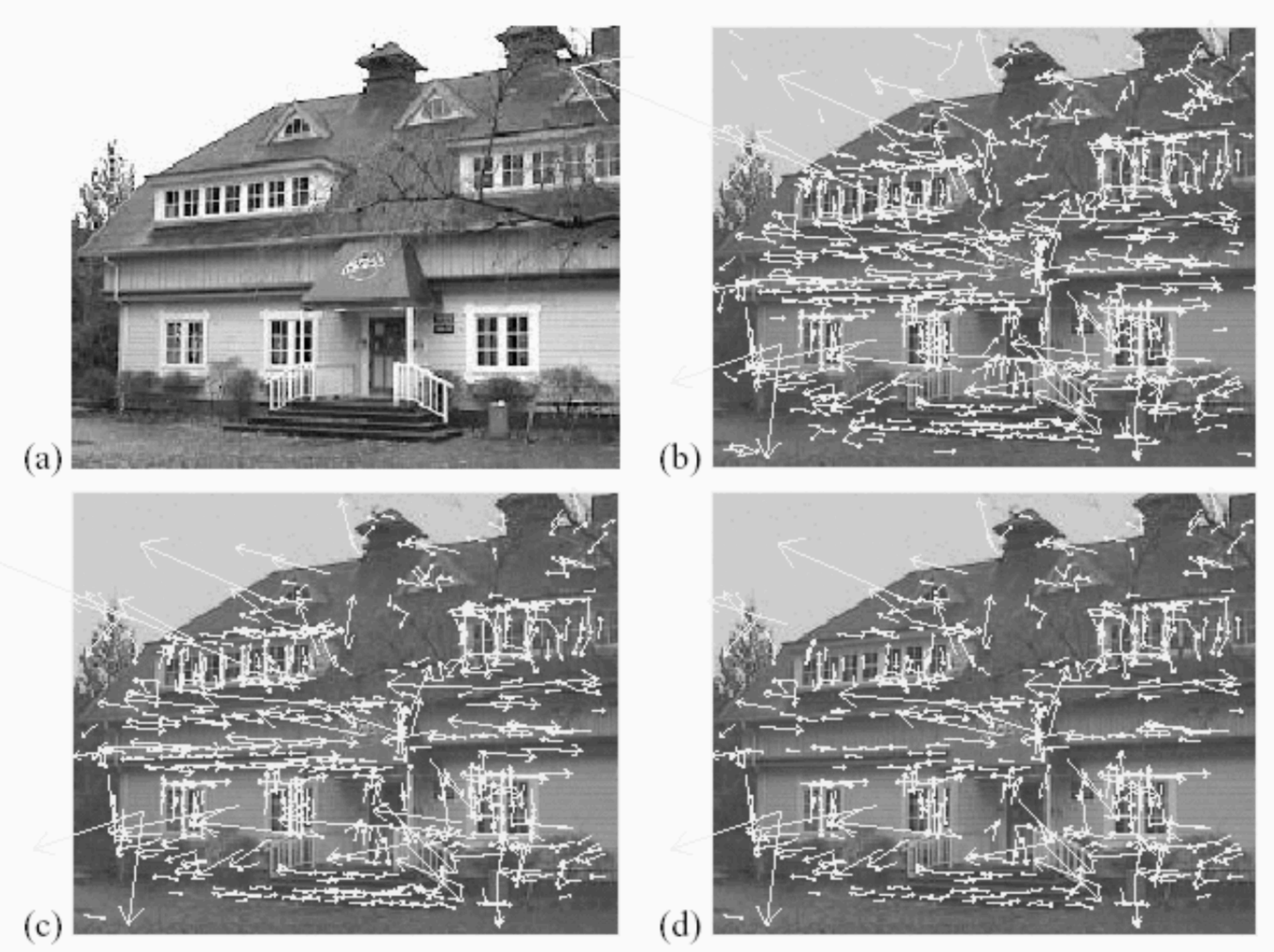
— Each key specifies stable 2D coordinates (x , y , scale, orientation)





3. Keypoint Localization

Example:



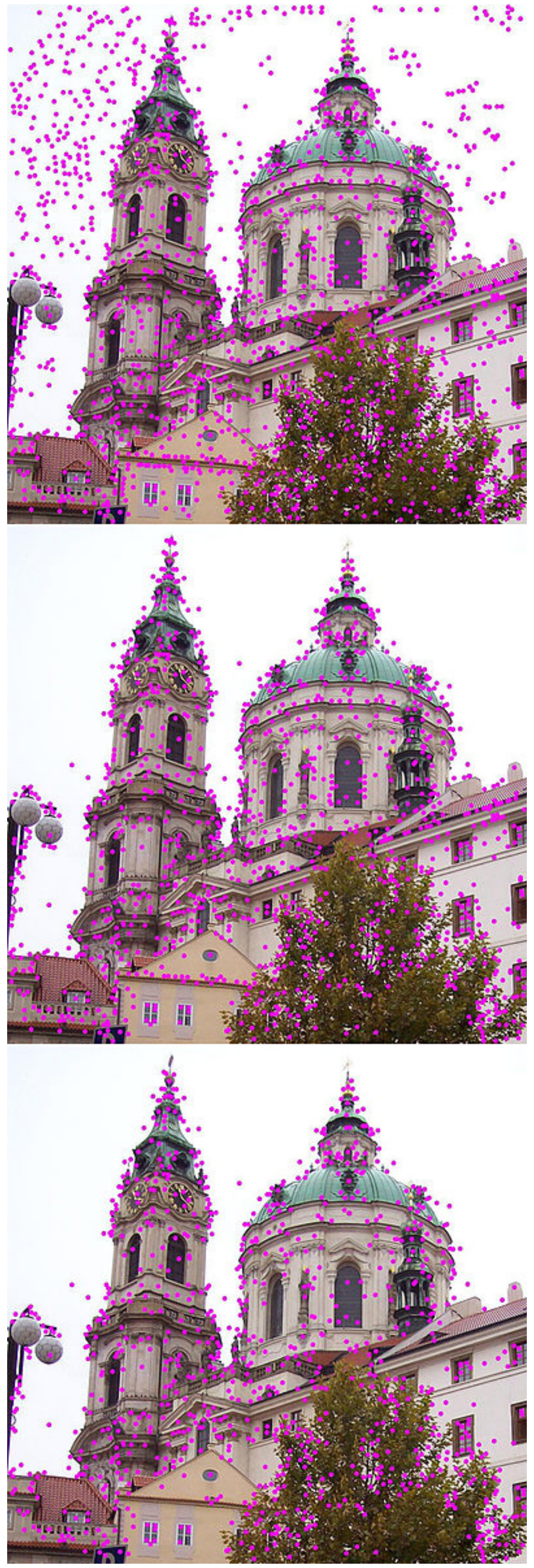
(a) 233×189 image

(b) 832 DOG extrema

(c) 729 left after peak value threshold

(d) 536 left after testing ratio of principal curvatures

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)

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

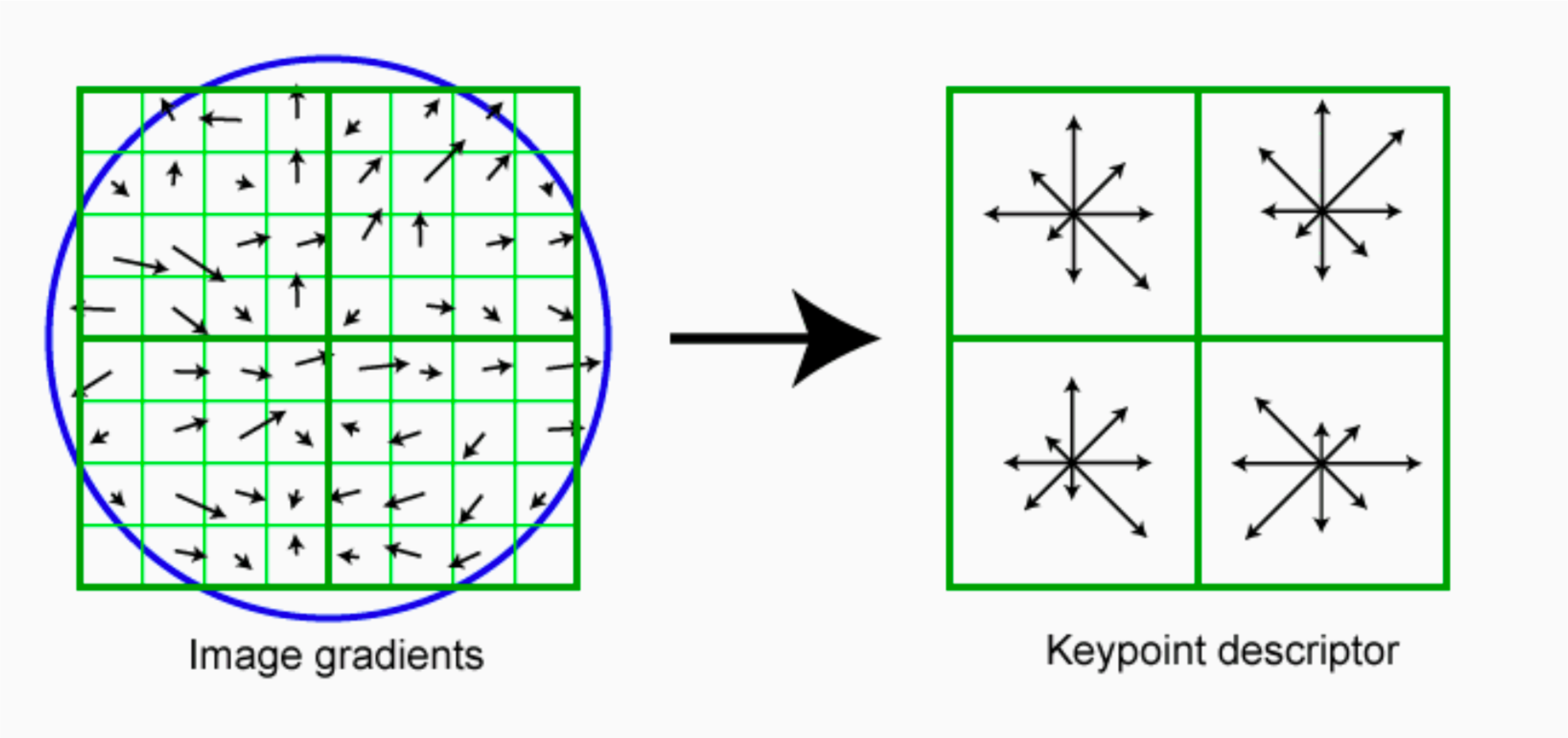
— Keypoint detection is not the same as keypoint description, e.g. some applications skip keypoint detection and extract SIFT descriptors on a regularly spaced grid

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

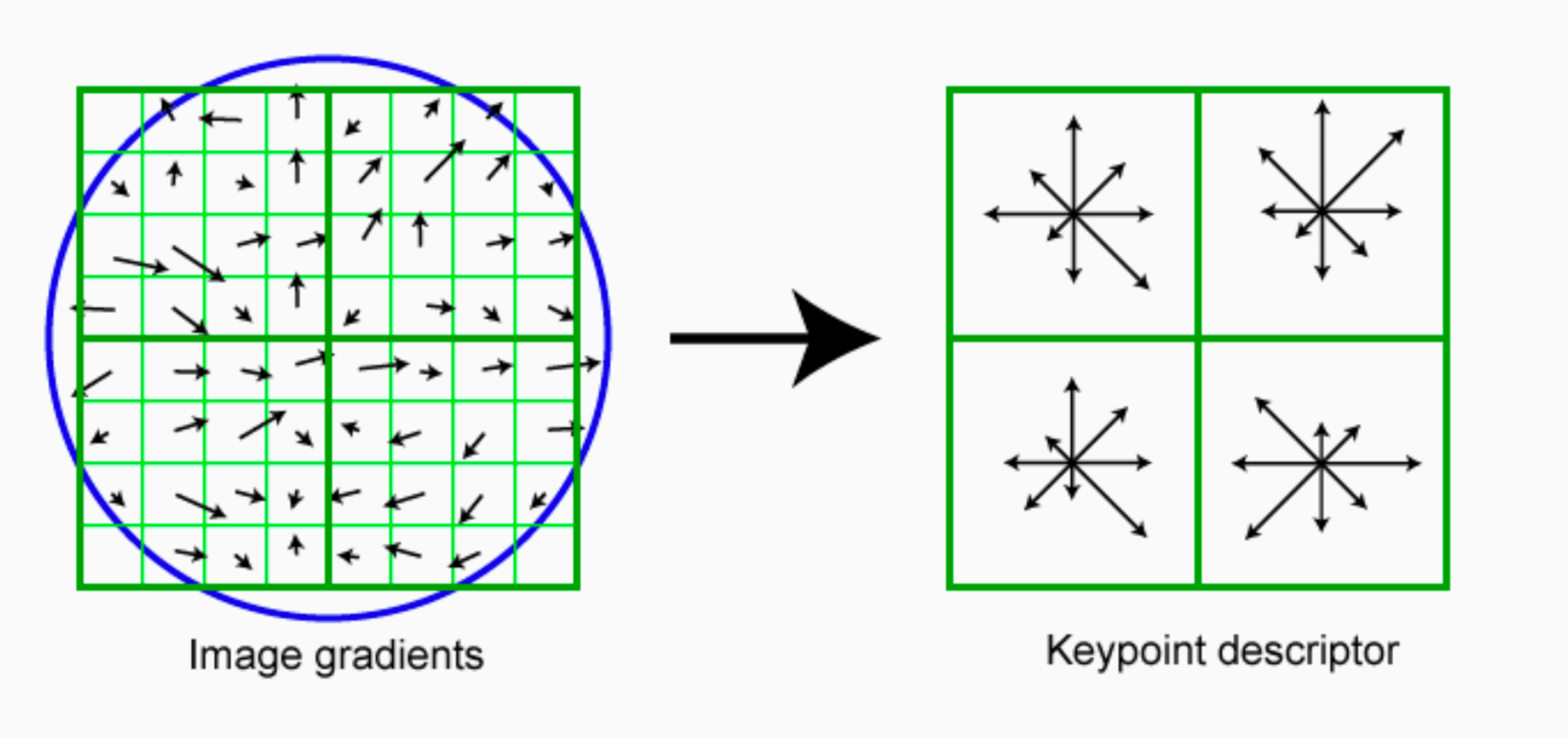


Demo

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

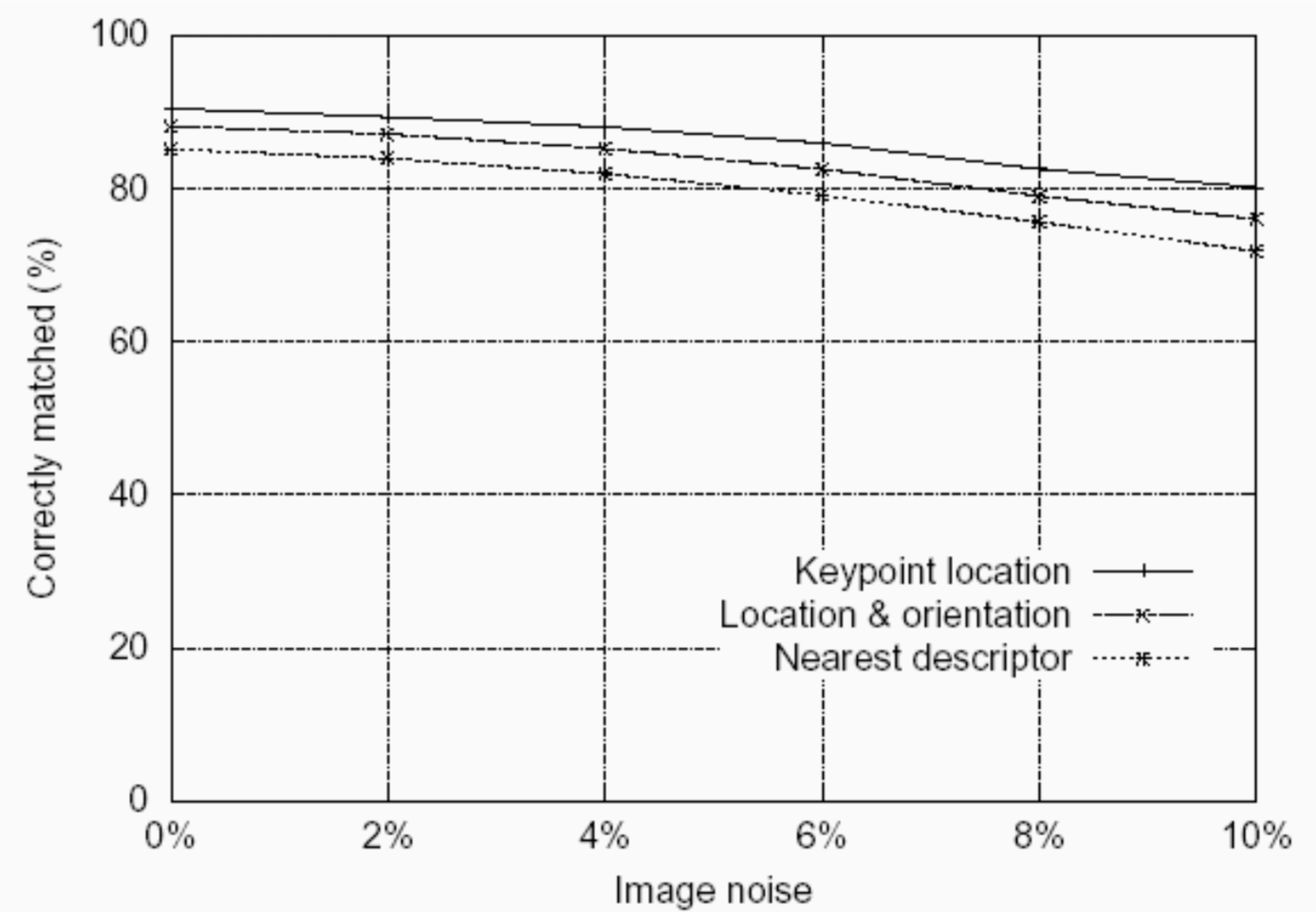
— if brightness values are scaled (multiplied) by a constant, the gradients are scaled by the same constant, and the normalization cancels the change

— 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

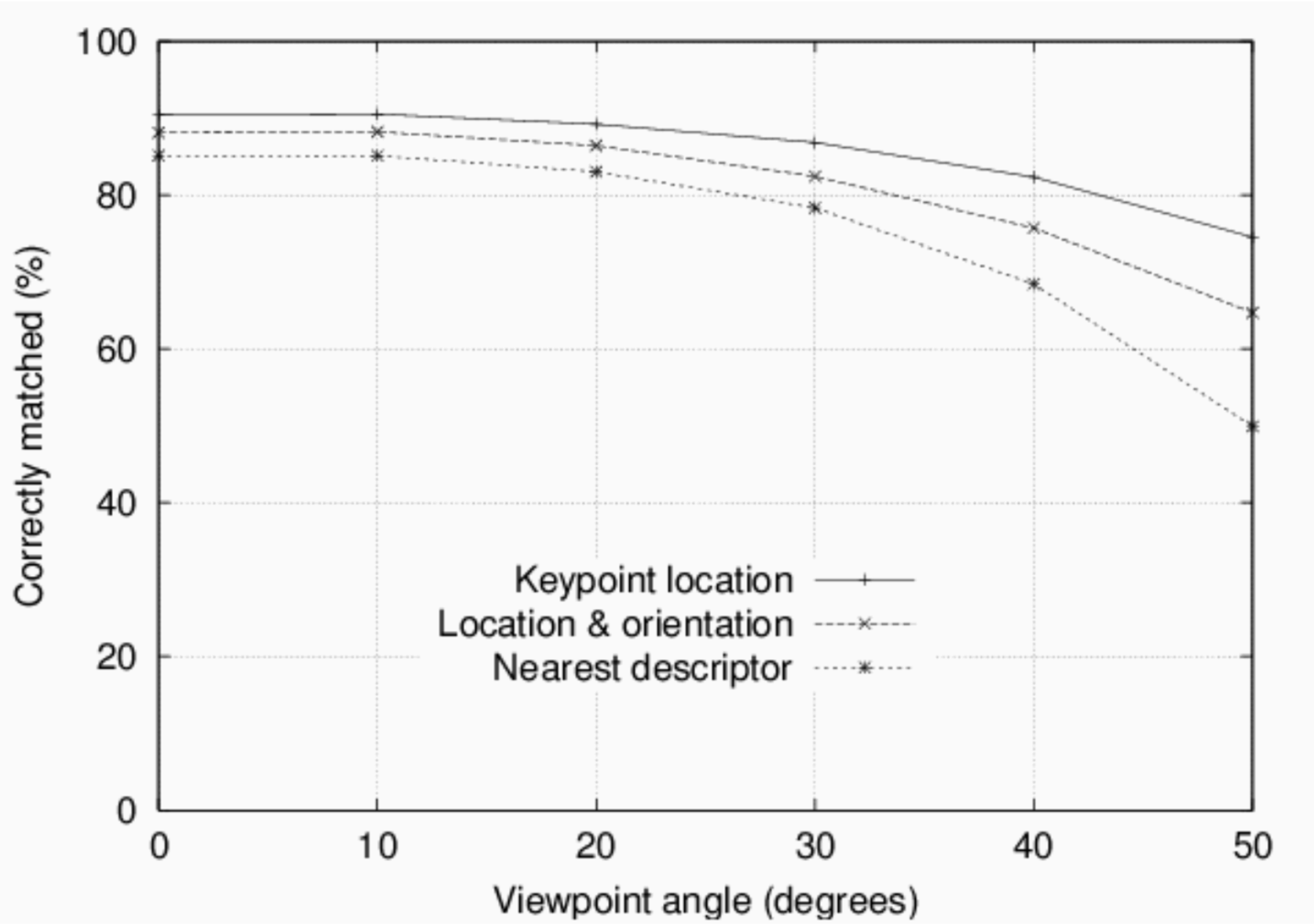
Find nearest neighbour in database of 30,000 features



Feature Stability to **Affine Change**

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

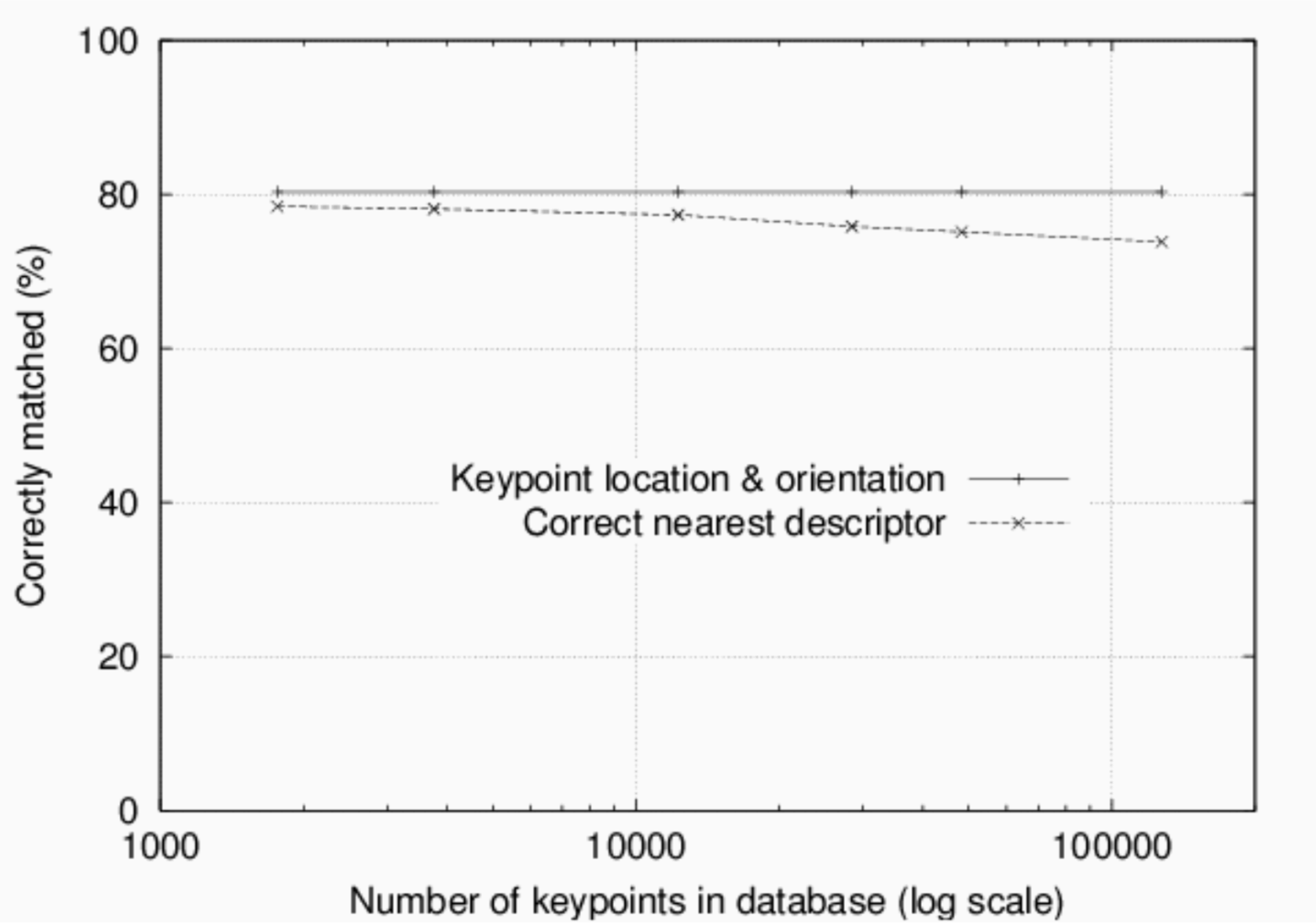
Find nearest neighbour in database of 30,000 features



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

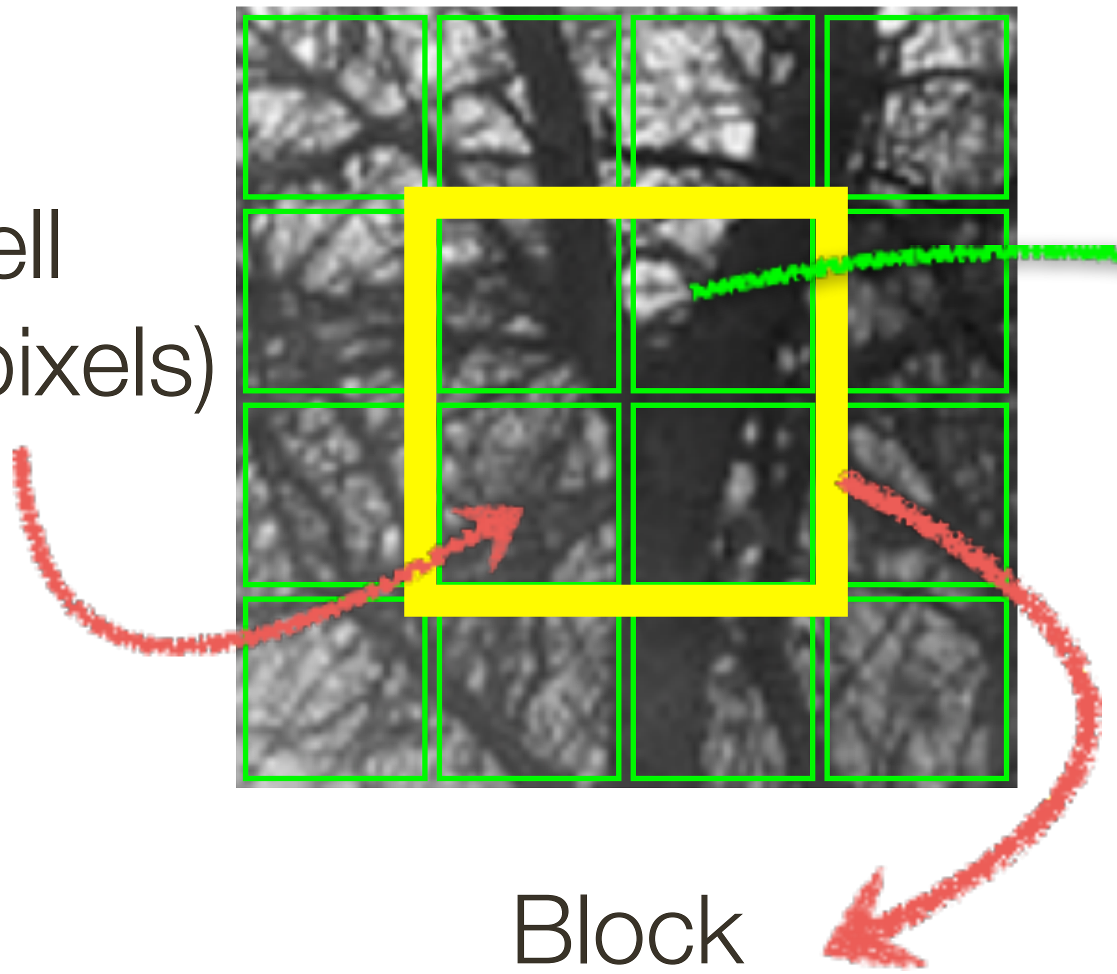
Four steps to SIFT feature generation:

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)

Histogram of Oriented Gradients (**HOG**) Features

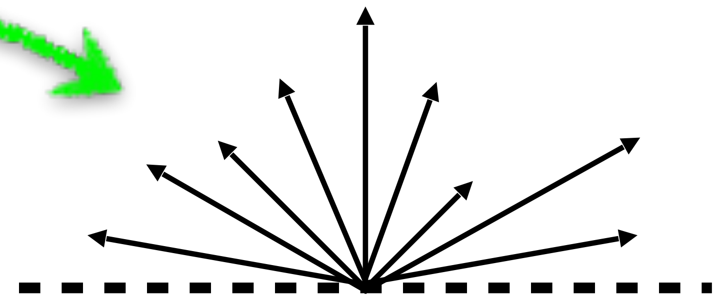
Dalal, Triggs. Histograms of Oriented Gradients for Human Detection. CVPR, 2005

Cell
(8x8 pixels)



Block
(2x2 cells)

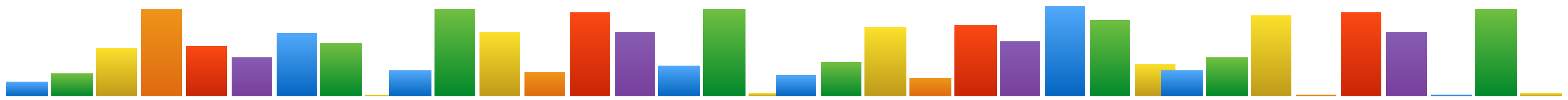
histogram of
'unsigned'
gradients



soft binning

gradient magnitude histogram
(one for each cell)

Concatenate and L-2 normalization



Single scale, no dominant orientation

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Histogram of Oriented Gradients (**HOG**) Features

1 cell step size

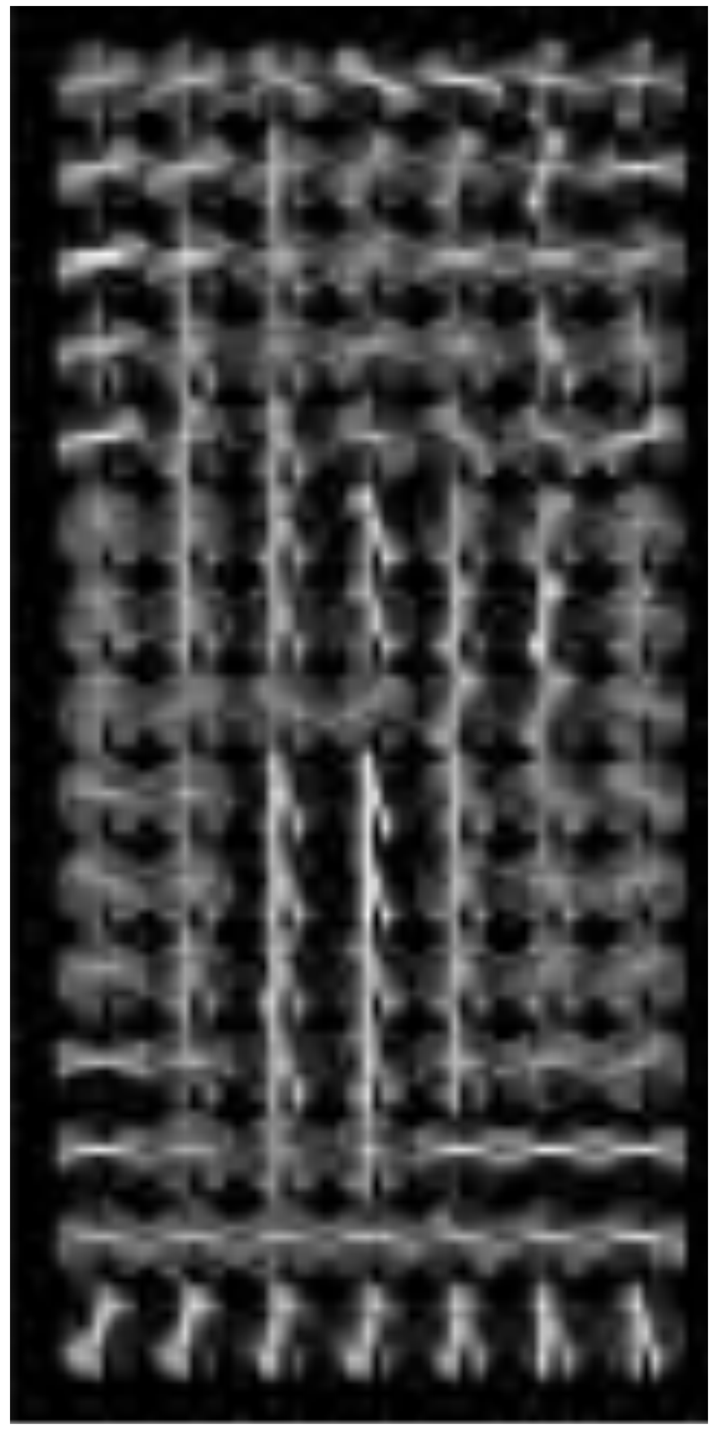
Pedestrian detection

128 pixels
16 cells
15 blocks

64 pixels
8 cells
7 blocks

15 x 7 x 4 x 9 =
3780

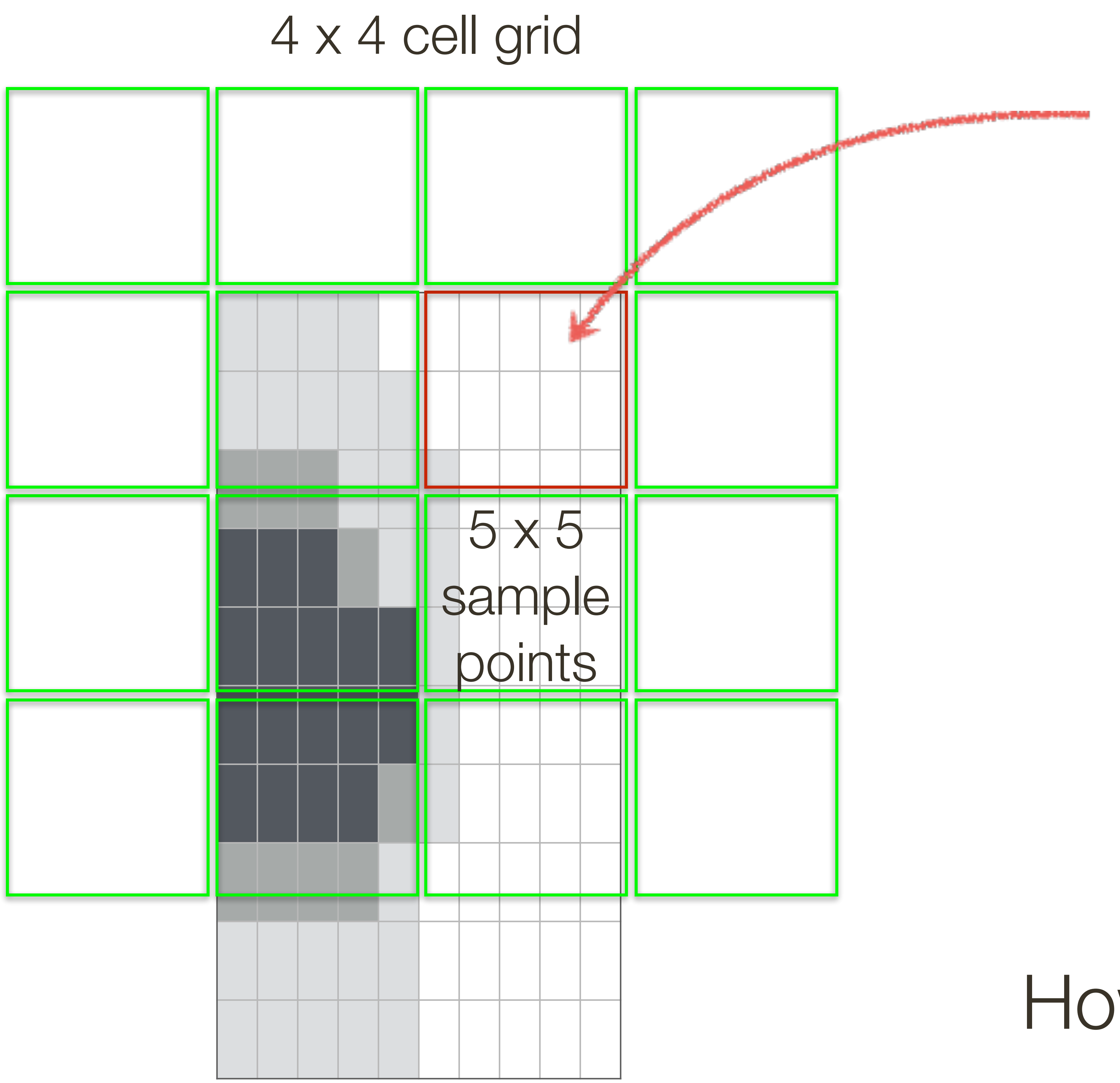
visualization



Redundant representation due to overlapping blocks

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

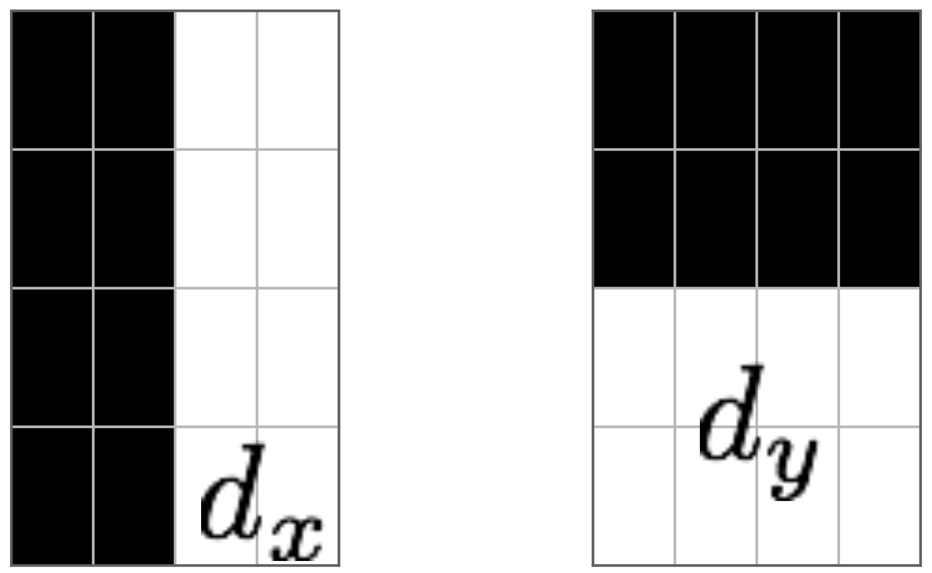
‘Speeded’ Up Robust Features (**SURF**)



Each cell is represented by 4 values:

$$\left[\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right]$$

Haar wavelets filters
(Gaussian weighted from center)

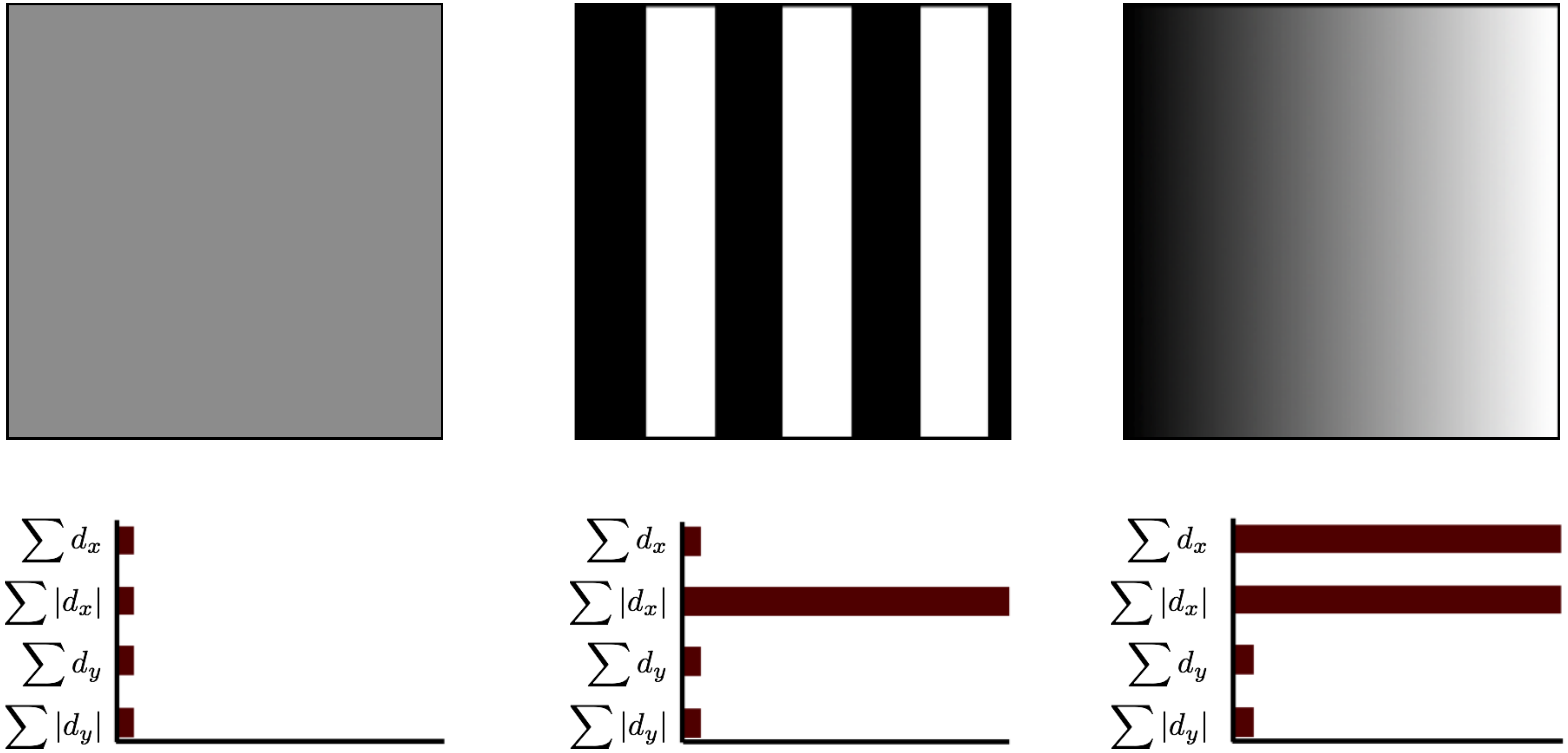


How big is the SURF descriptor?

64 dimensions

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

‘Speeded’ Up Robust Features (**SURF**)



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

SIFT and **Object Recognition**

Object recognition requires us to first match each keypoint independently to the database of keypoints

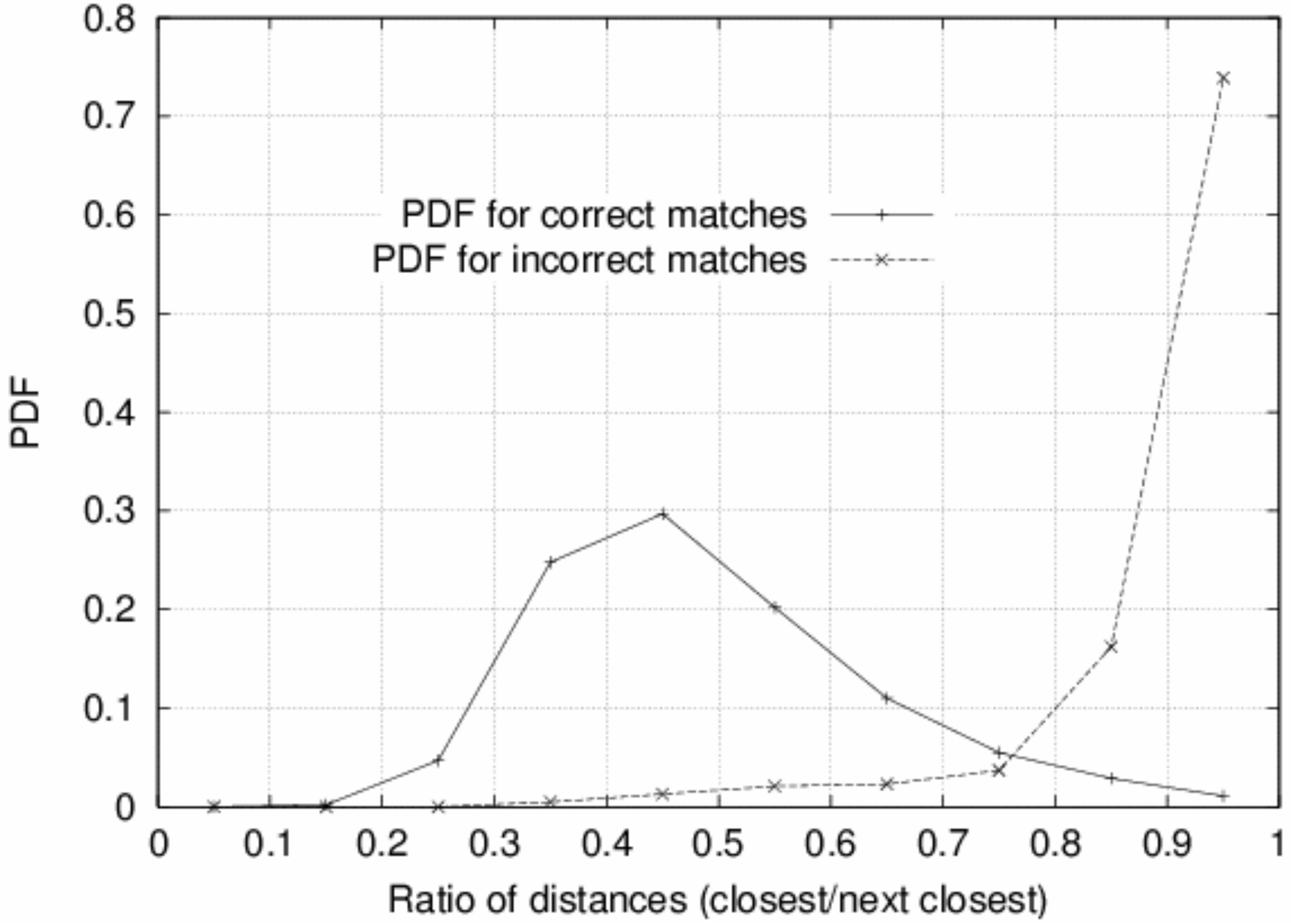
Many features will not have any correct match in the database because they arise from background clutter

It would be useful to have a way to **discard features** that do not have any good match

Probability of **Correct** Match

Compare ratio of distance of **nearest** neighbour to **second** nearest neighbour (from different object)

Threshold of 0.8 provides excellent separation

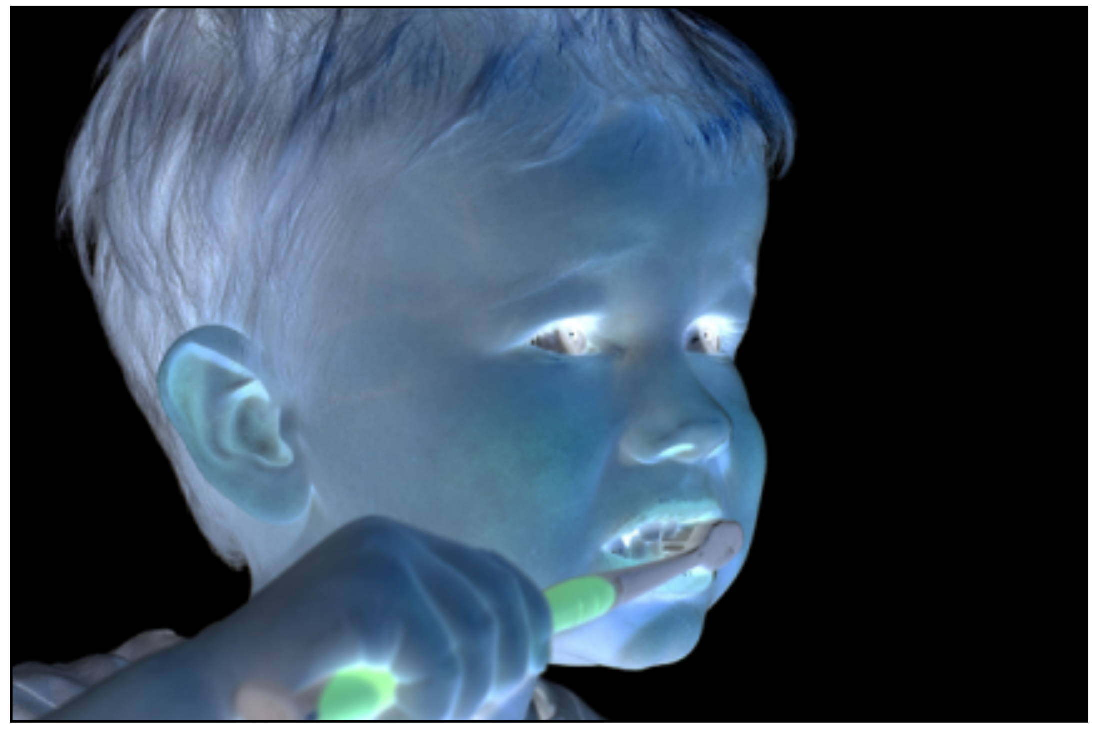


What types of **transformations** can we do?

$I(X,Y)$

Filtering

$I'(X,Y)$

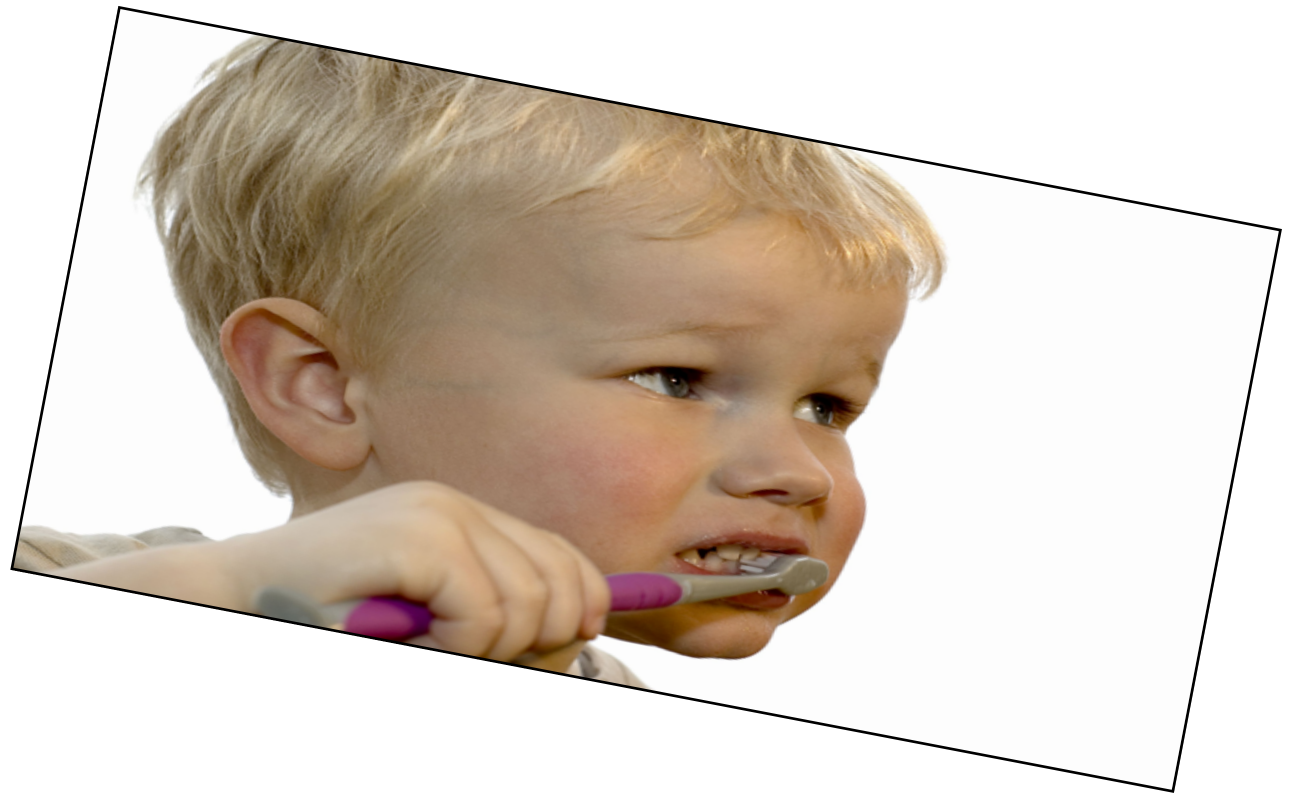


changes range of image function

$I(X,Y)$

Warping

$I'(X,Y)$

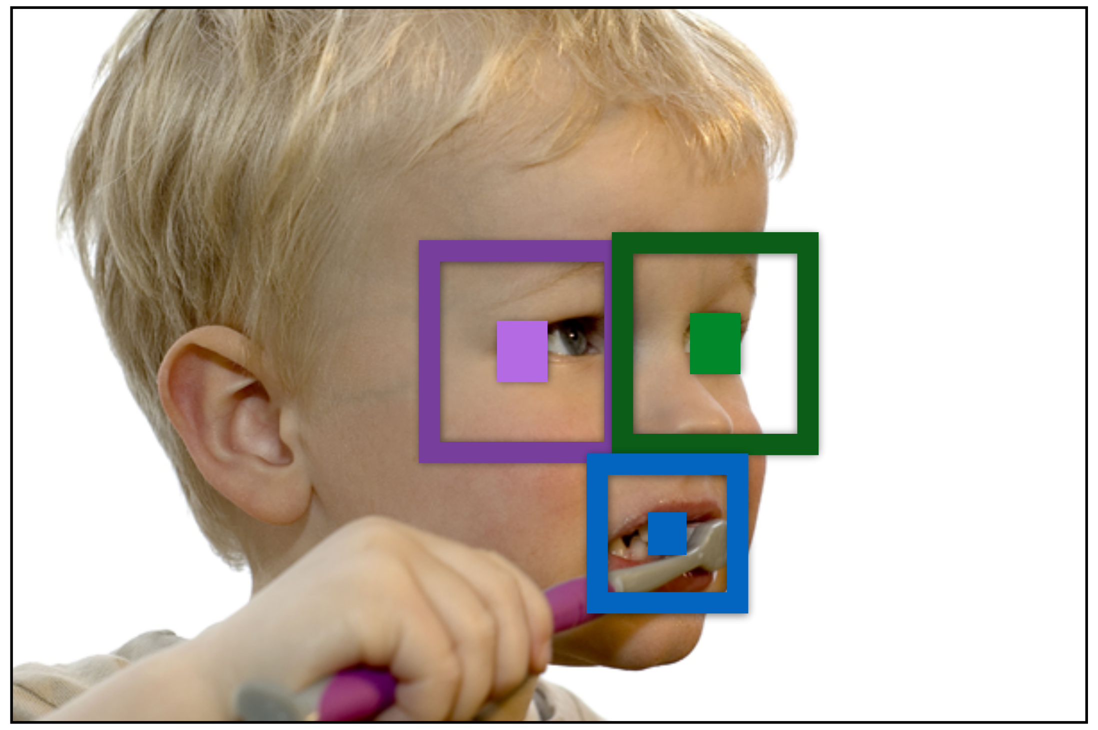


changes domain of image function

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

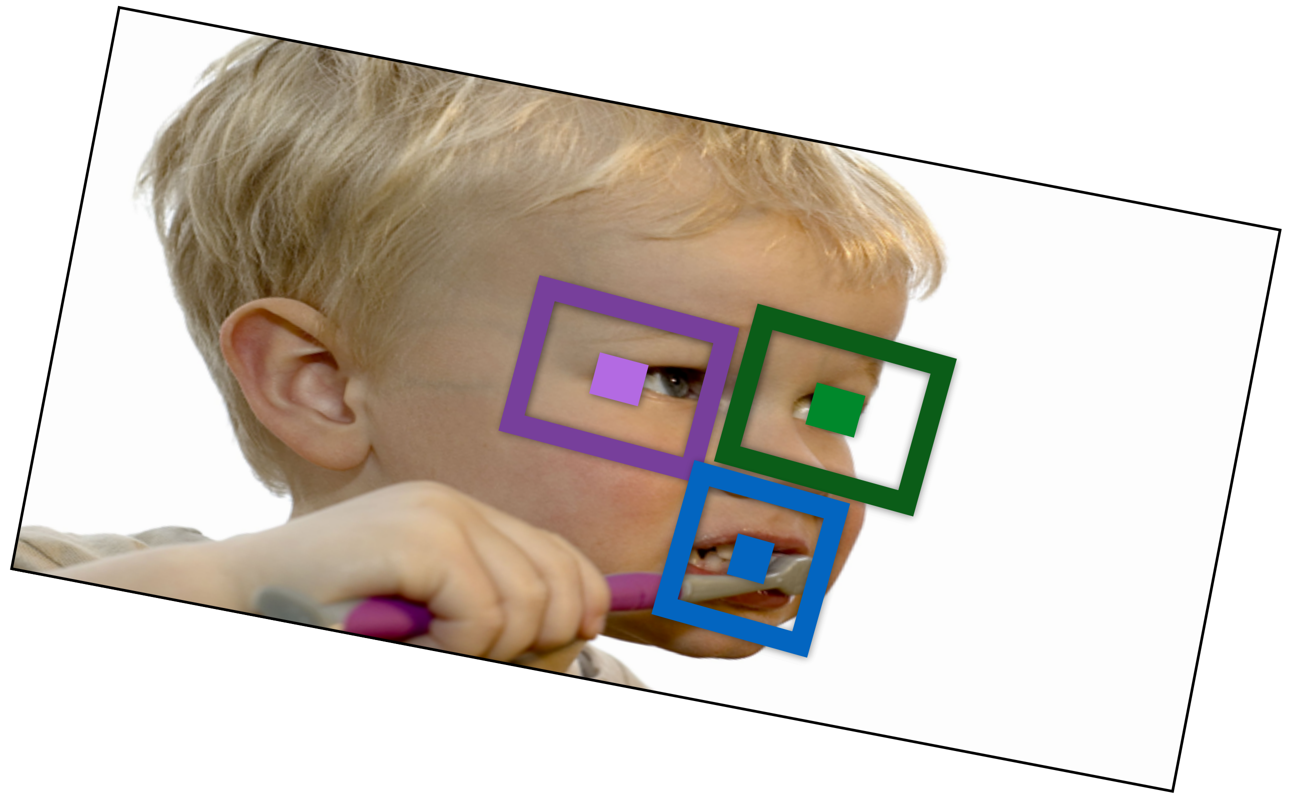
What types of **transformations** can we do?

$I(X,Y)$



Warping

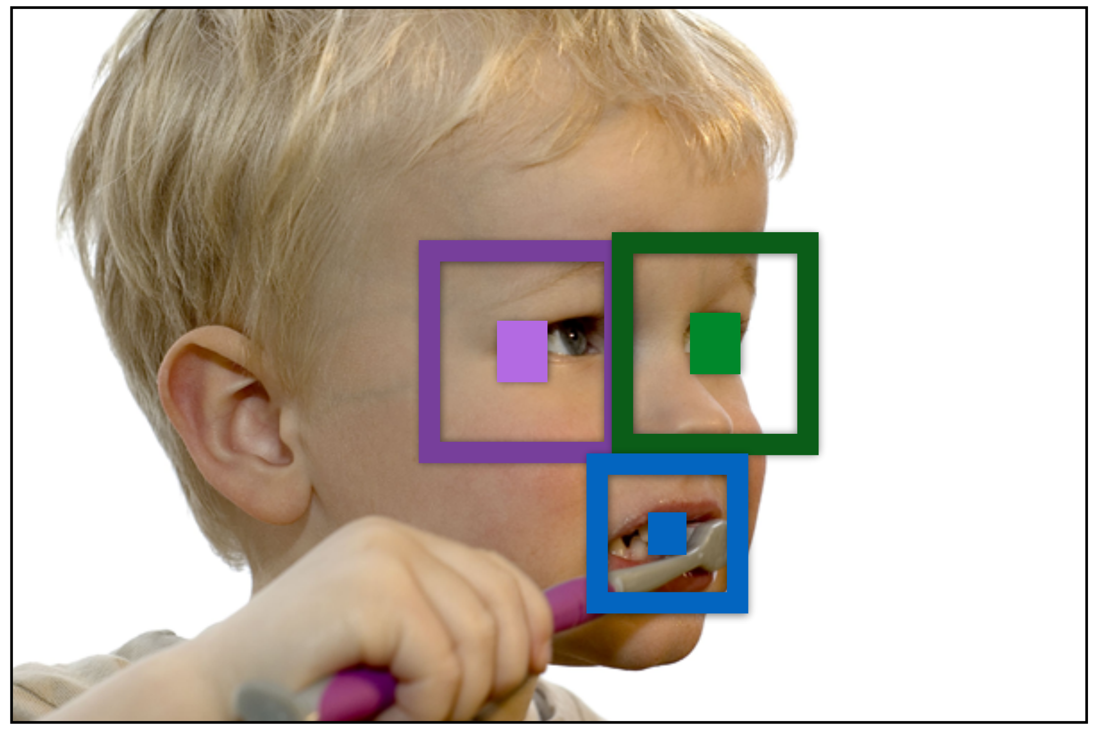
$I'(X,Y)$



changes domain of image function

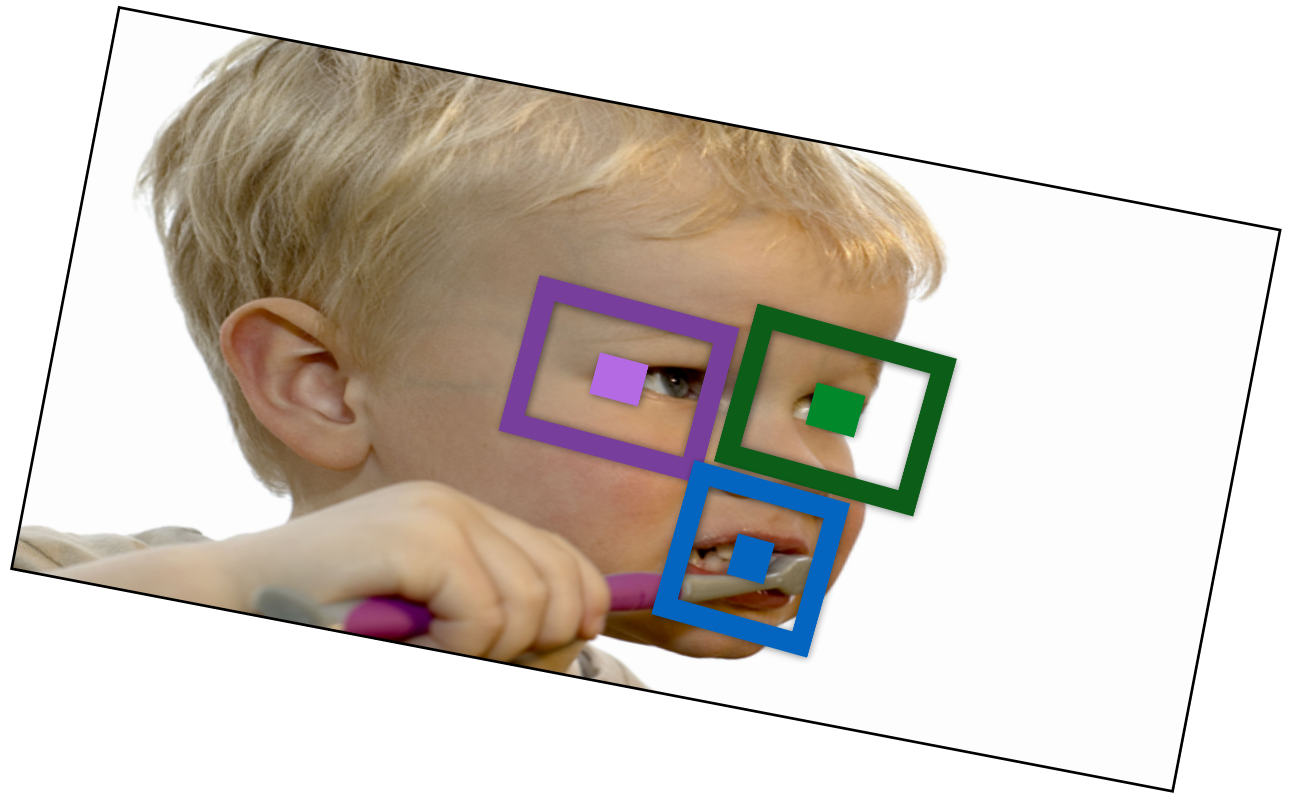
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

What types of **transformations** can we do?



$I(X,Y)$

Warping



$I'(X,Y)$

changes domain of image function

We will call this
"**Warping**" a "**Model**"

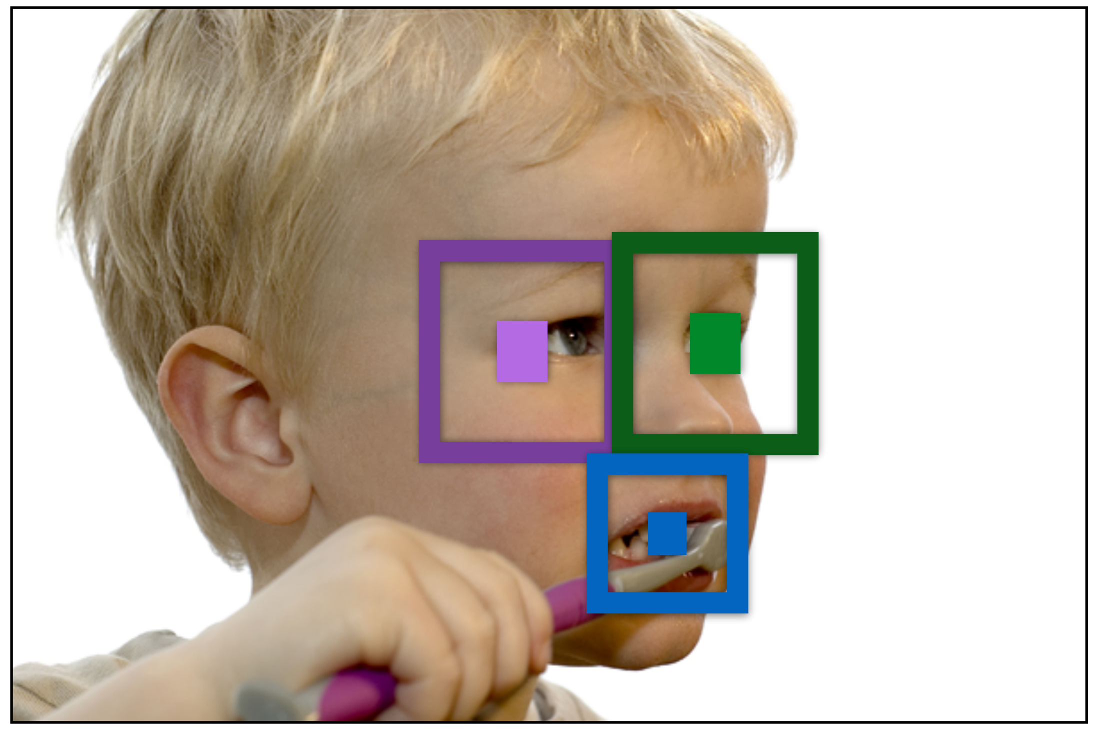
$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = M \begin{bmatrix} X \\ Y \end{bmatrix}$$

$$I'(X',Y') = I(X,Y)$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

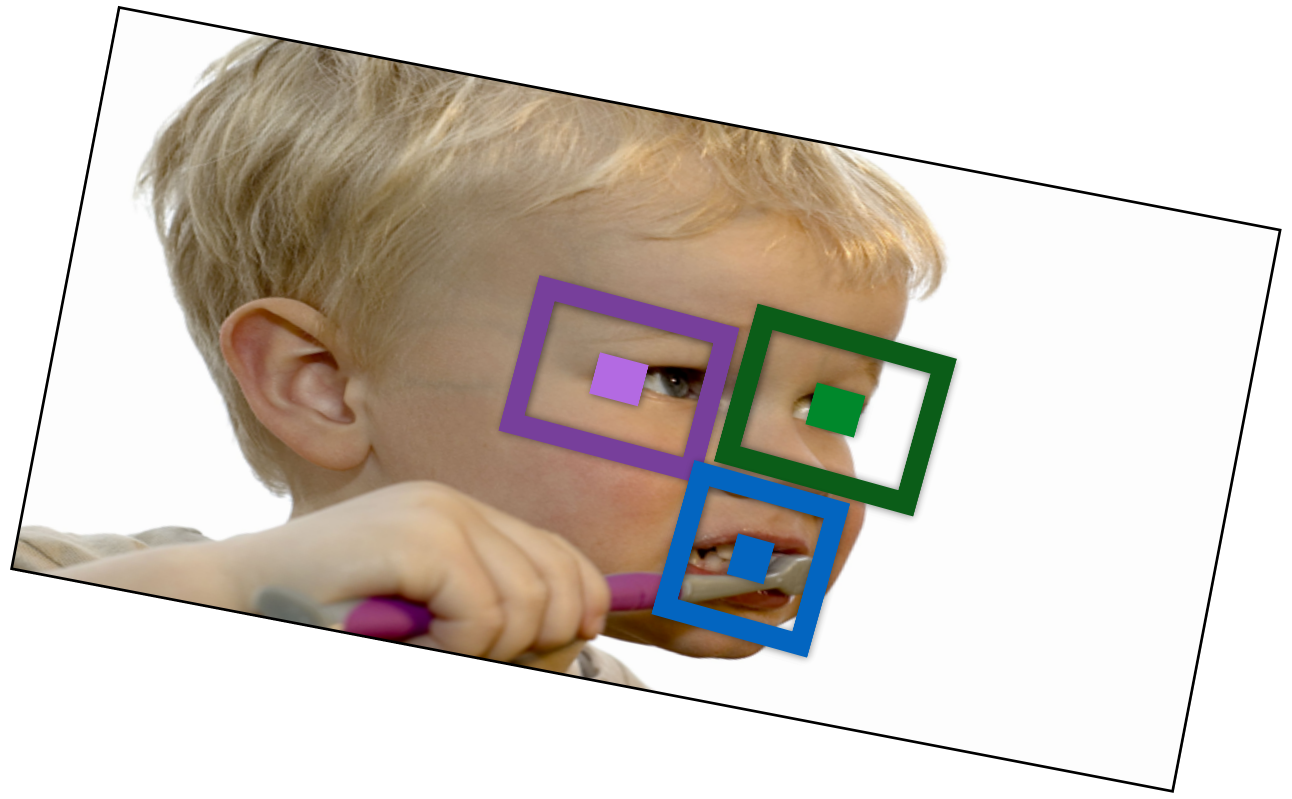
What types of **transformations** can we do?

$I(X, Y)$



Warping

$I'(X, Y)$



changes domain of image function

We will call this
"**Warping**" a "**Model**"

$$\begin{bmatrix} X' \\ Y' \\ 1 \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$I'(X', Y') = I(X, Y)$$

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Nearest-Neighbor Matching to Feature Database

Hypotheses are generated by **approximate nearest neighbour** matching of each feature to vectors in the database

— Use best–bin–first (Beis & Lowe, 97) modification to k-d tree algorithm

— Use heap data structure to identify bins in order by their distance from query point

Result: Can give speedup by factor of 1,000 while finding nearest neighbour (of interest) 95% of the time

Identifying **Consistent** Features

We have matched keypoints to a database of known keypoints extracted from training images

Next we identify **clusters of at least 3 features** that agree on an object and its pose

— a typical image contains 2,000+ features → detecting less than 1% inliers among 99% outliers!

Lowe's solution uses the generalized **Hough transform**

— vote for each potential match according to model ID and pose

— insert into multiple bins to allow for error in similarity approximation

— (more on Hough transforms later)

Model **Verification**

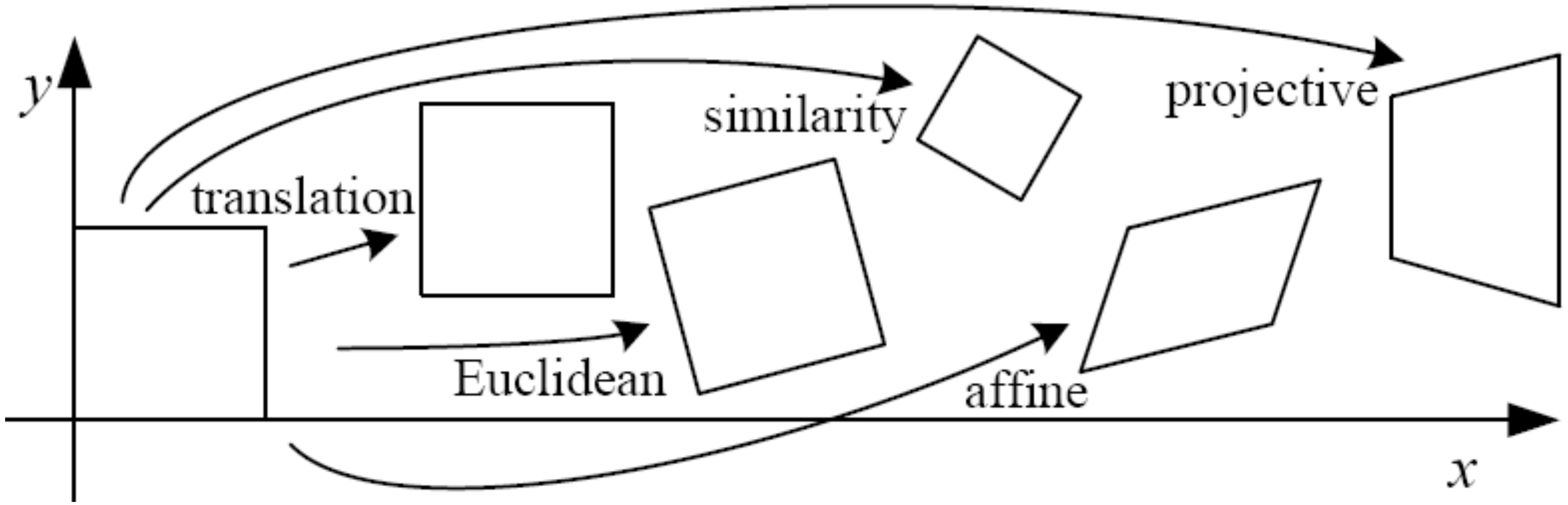
1. Examine all clusters with at least 3 features

2. Perform least-squares affine **fit to model**

3. **Discard outliers** and perform top-down check for additional features

4. Evaluate probability that match is correct
   — Use Bayesian model, with probability that features would arise by chance if object was not present (Lowe, CVPR 01)

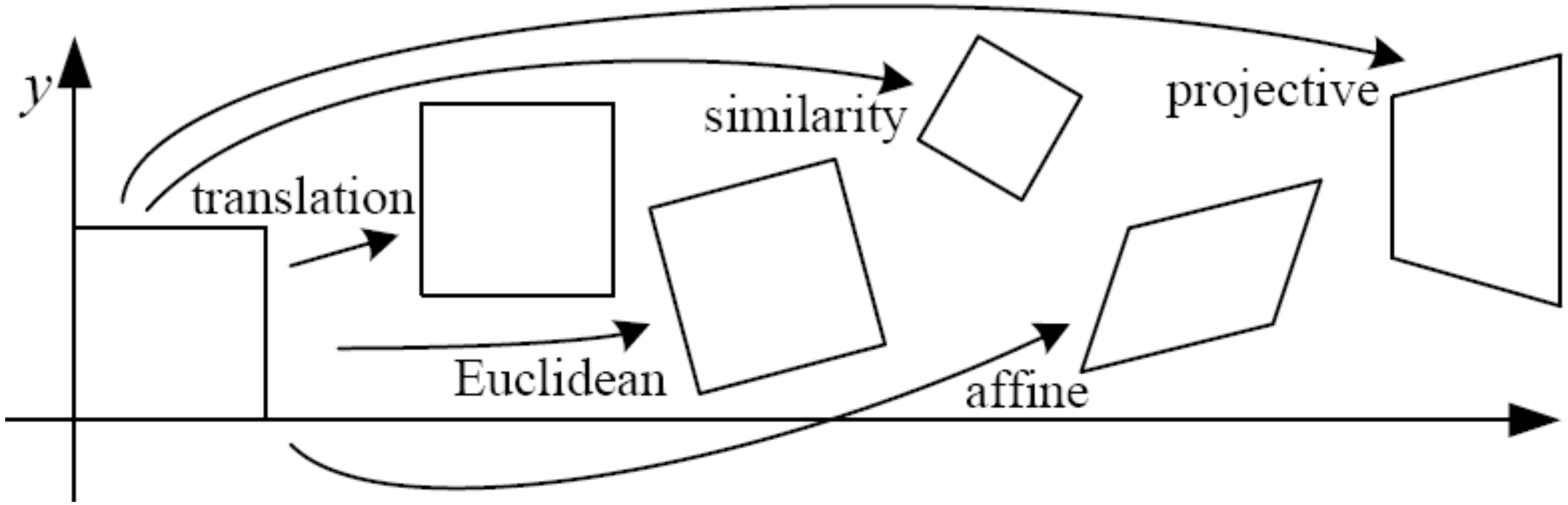
Aside: Classification of 2D Transformations



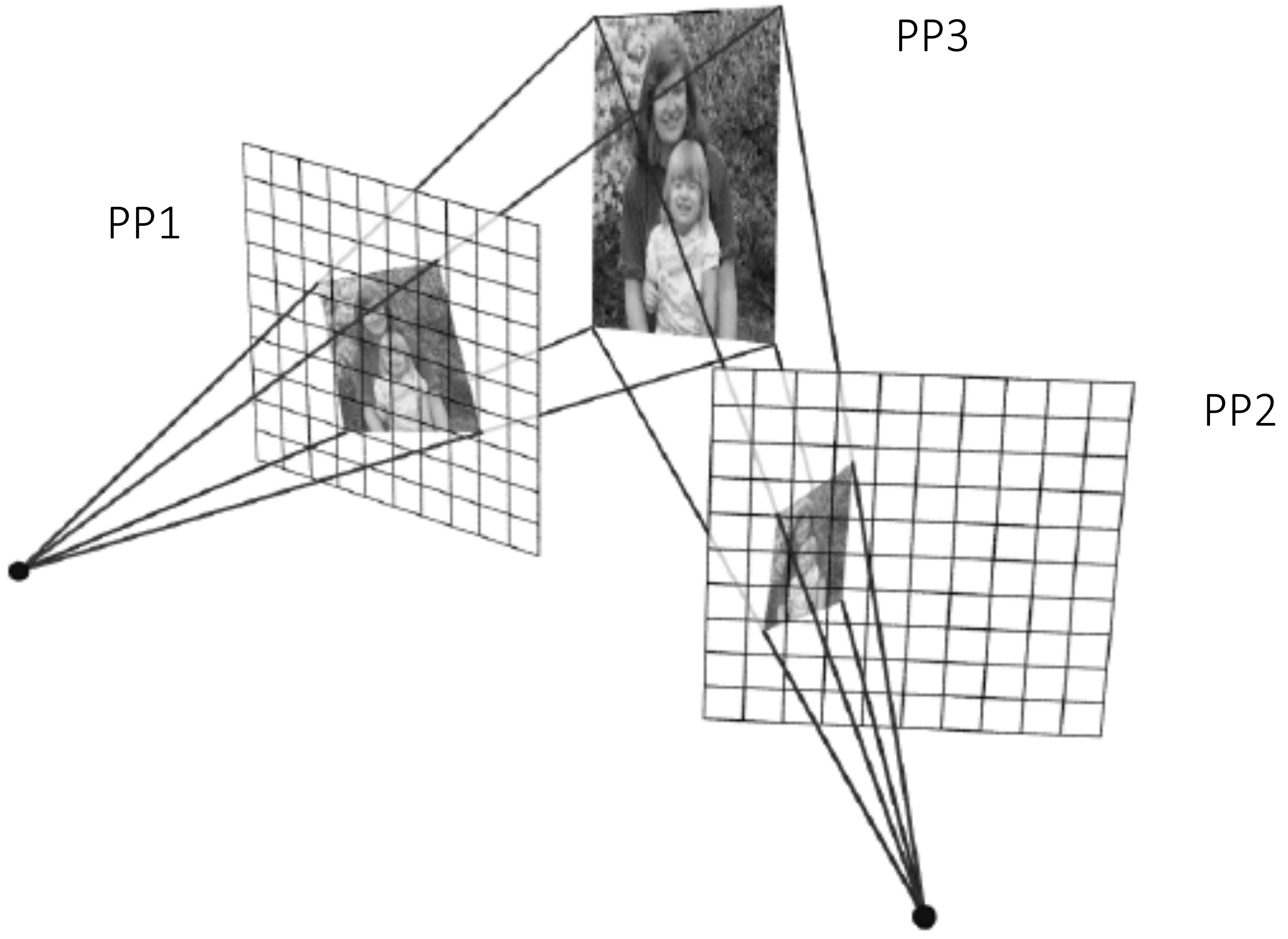
Name	Matrix	# D.O.F.
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: Classification of 2D Transformations

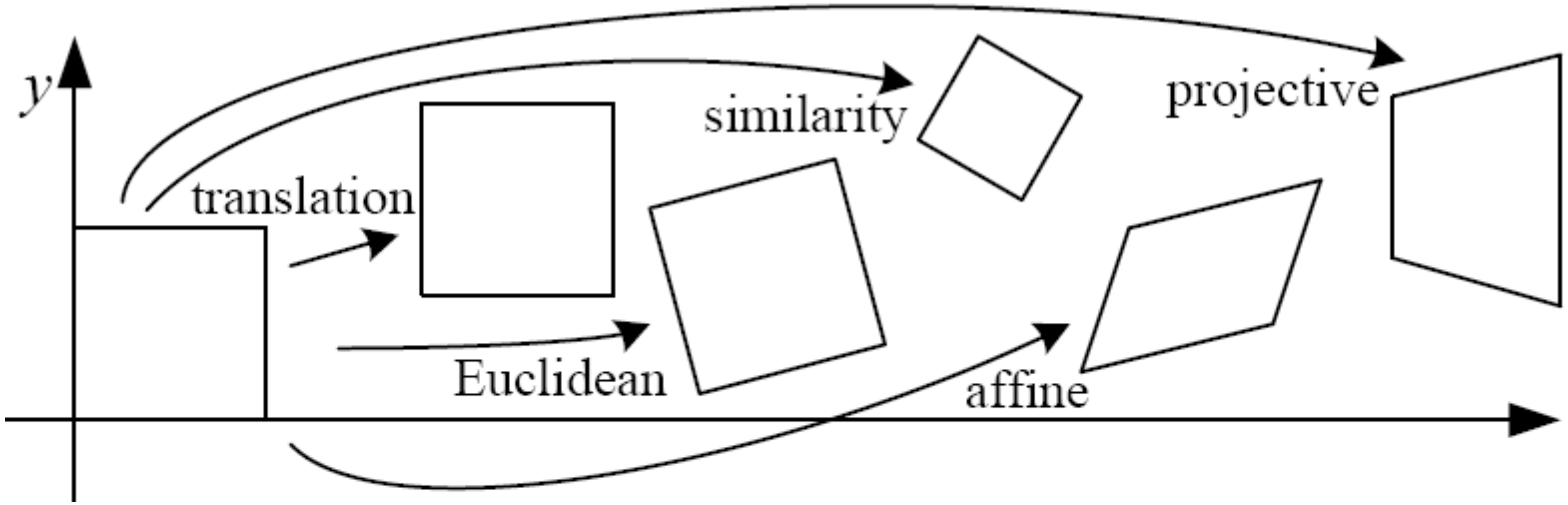


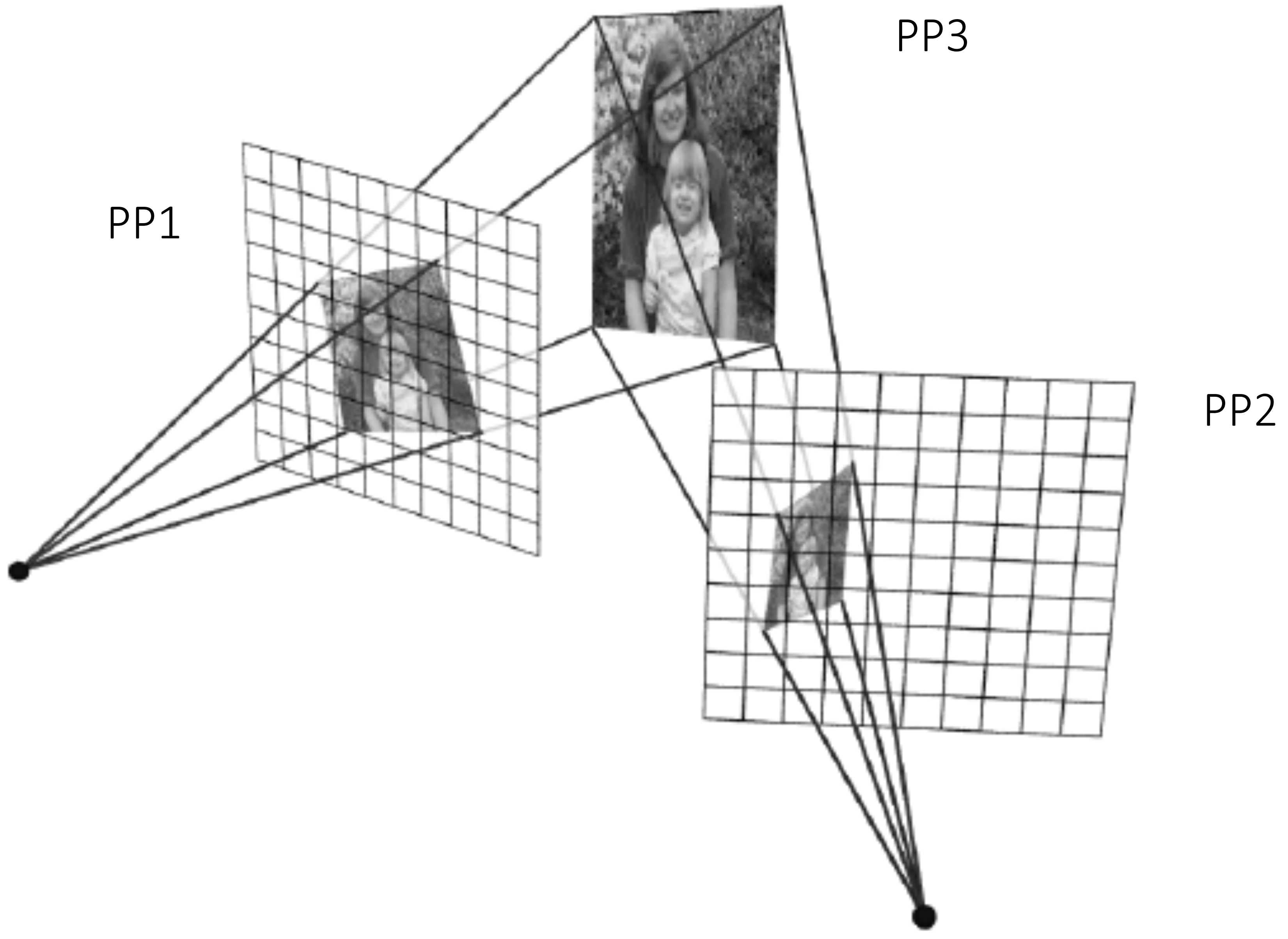
Which kind **transformation** is needed to warp projective plane 1 into projective plane 2?



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: Classification of 2D Transformations





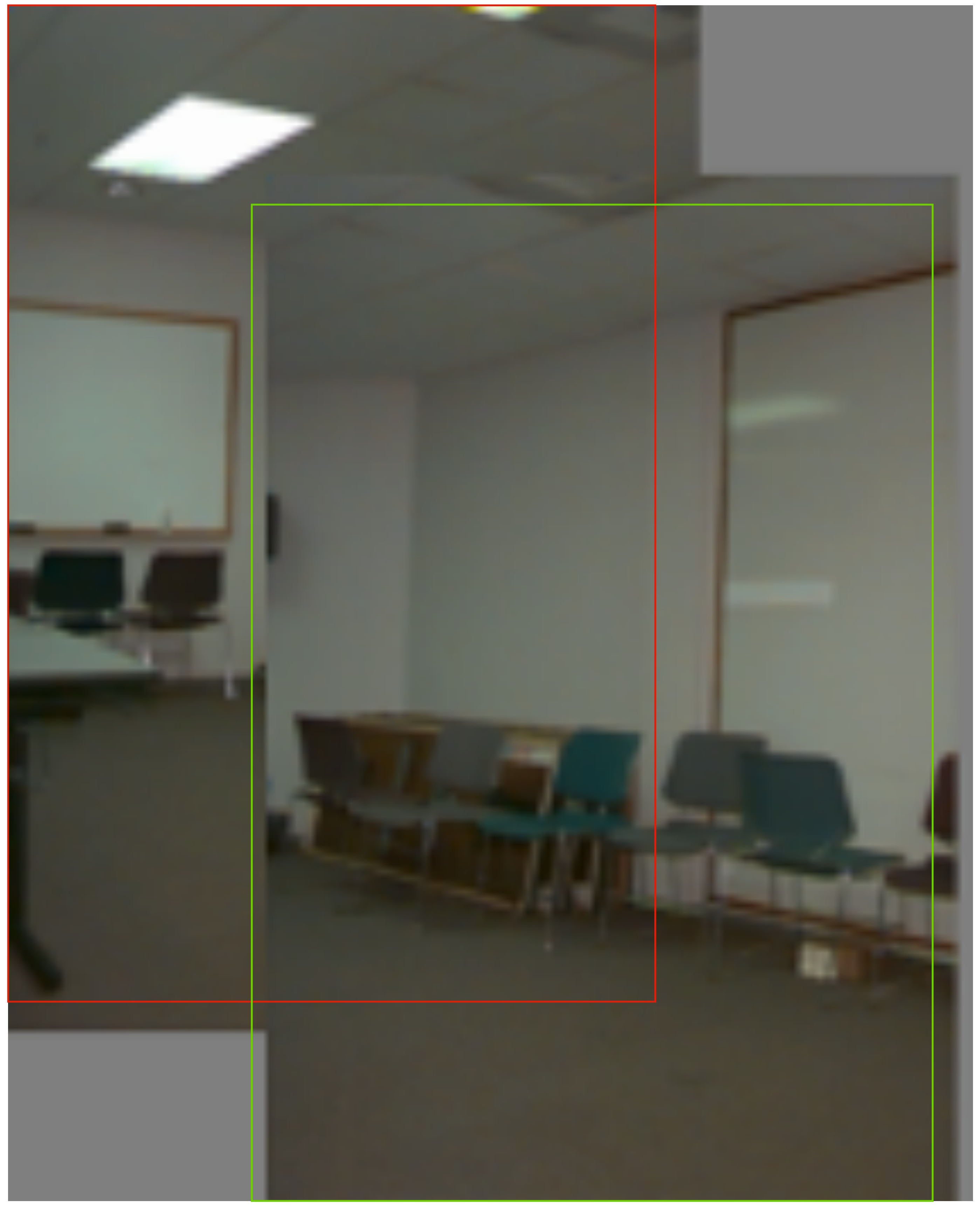
Which kind **transformation** is needed to warp projective plane 1 into projective plane 2?

— A **projective** transformation (a.k.a. a homography).

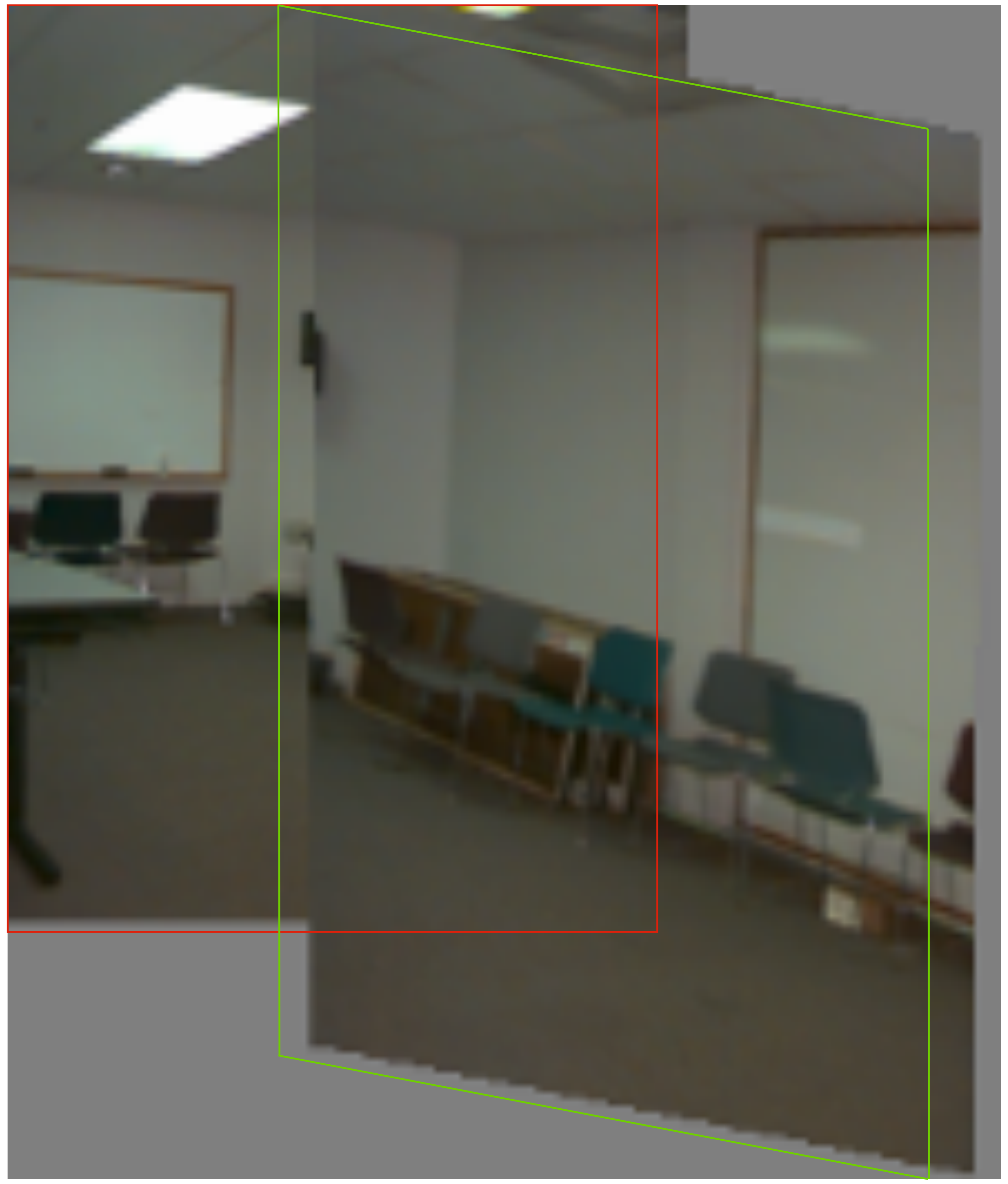
Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: Warping with Different Transformations

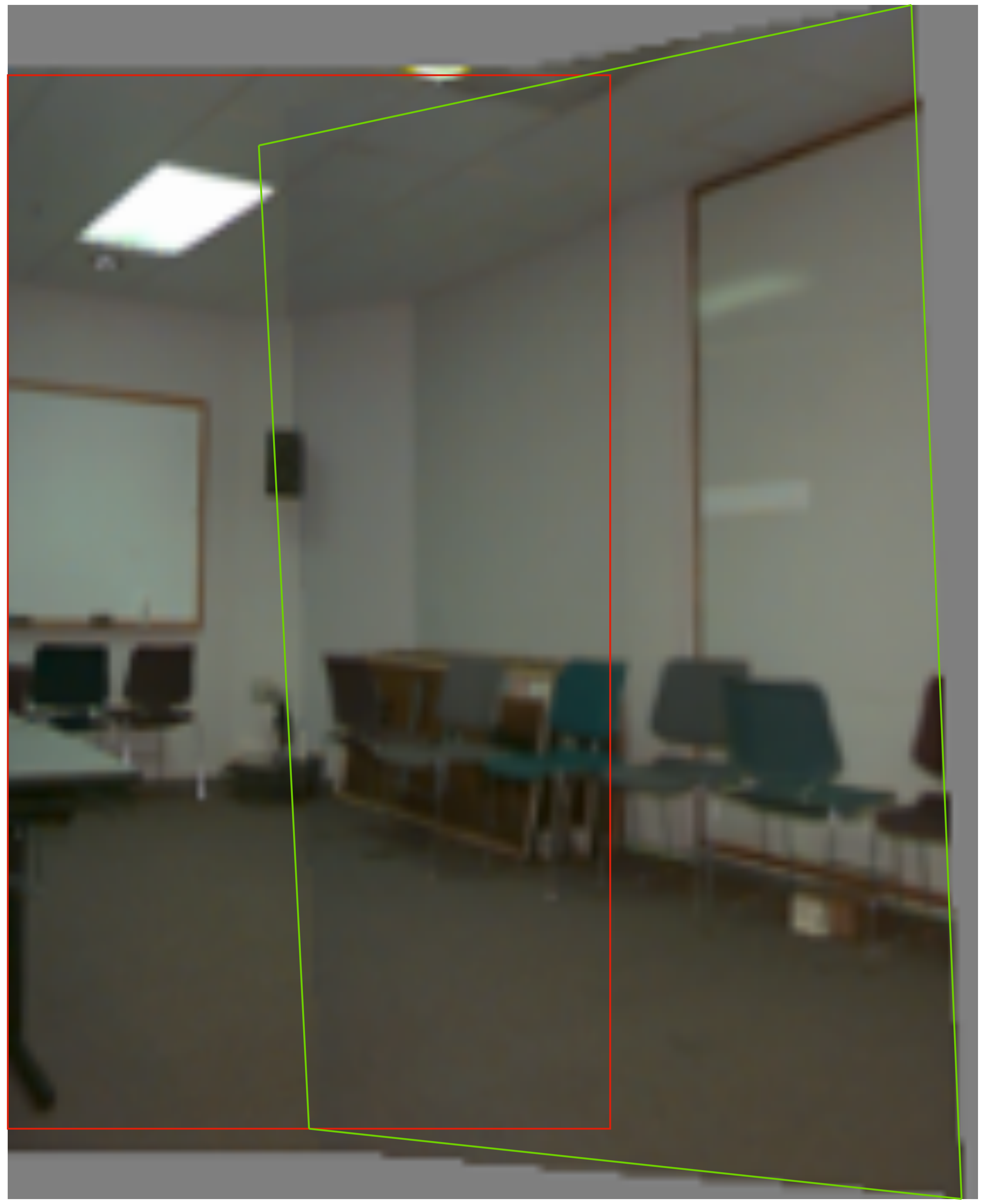
Translation



Affine



Projective
(homography)



Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: We can use homographies when …

1.… the scene is planar; or

2.… the scene is very far or has small (relative) depth variation → scene is approximately planar

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Aside: We can use homographies when …

3.… the scene is captured under camera rotation only (no translation or pose change)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)