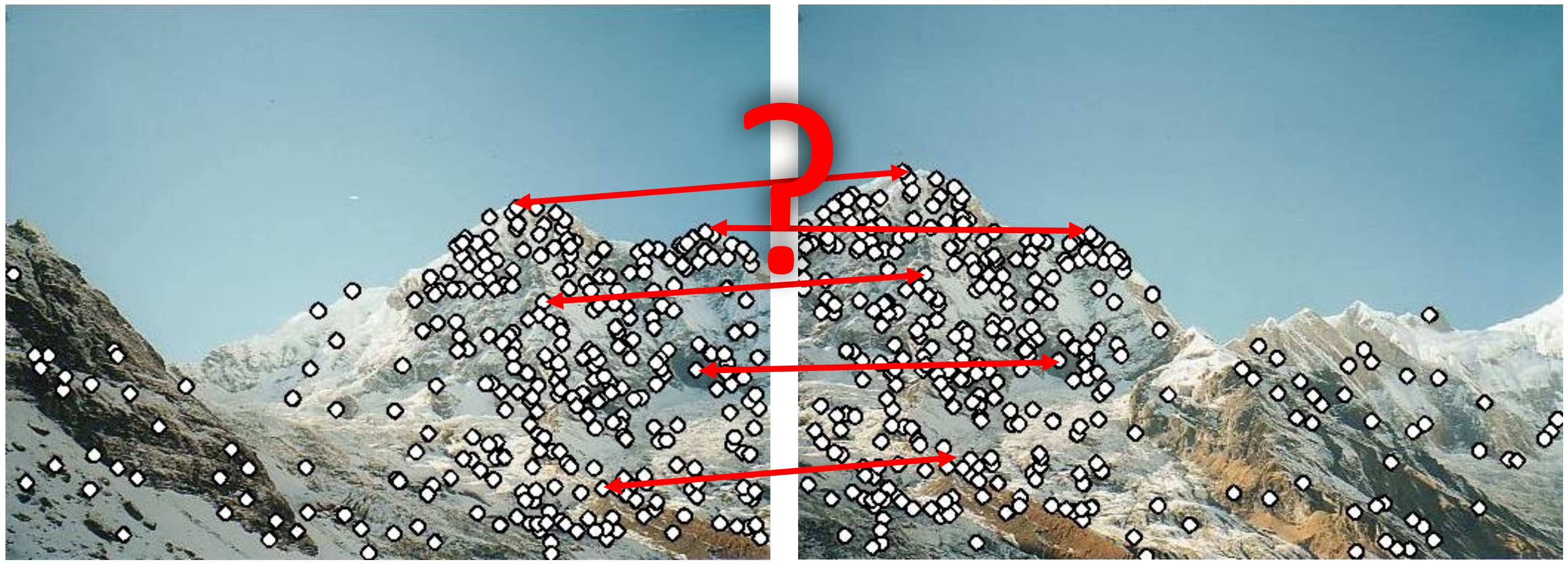
Feature description and matching

Matching feature points

We know how to detect good points

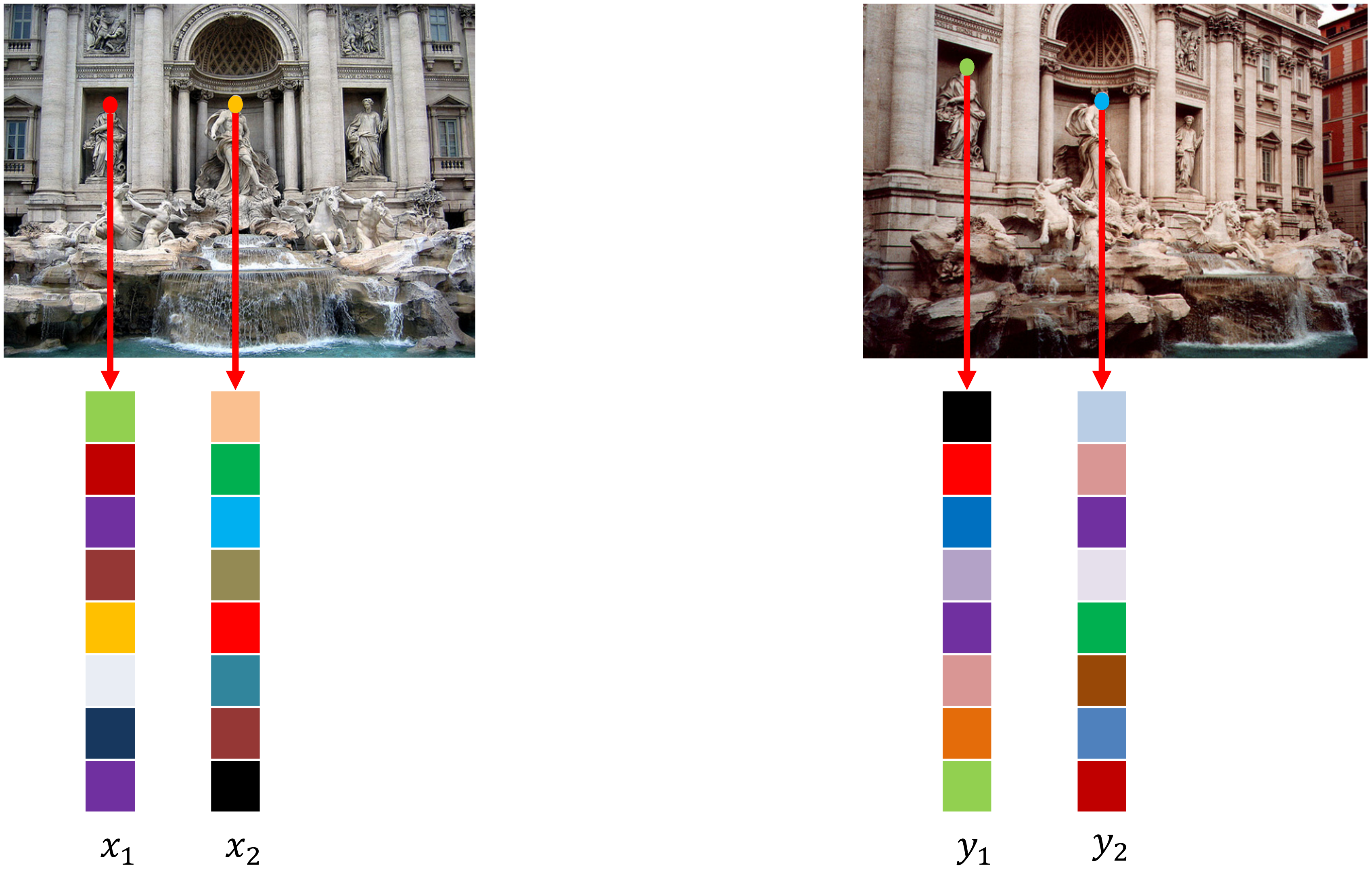
Next question: **How to match them?**



Two interrelated questions:

1. How do we *describe* each feature point?
2. How do we *match* descriptions?

Feature descriptor



Feature matching

- Measure the distance between (or similarity between) every pair of descriptors

	y_1	y_2
x_1	$d(x_1, y_1)$	$d(x_1, y_2)$
x_2	$d(x_2, y_1)$	$d(x_2, y_2)$

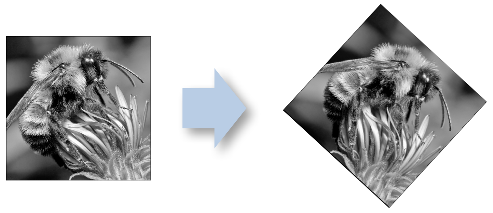
Invariance vs. discriminability

- Invariance:
  - Distance between descriptors should be small even if image is transformed
- Discriminability:
  - Descriptor should be highly unique for each point (far away from other points in the image)

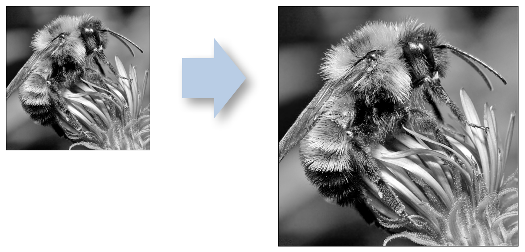
Image transformations

- Geometric

Rotation

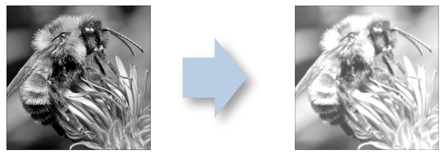


Scale



- Photometric

Intensity change



Invariance

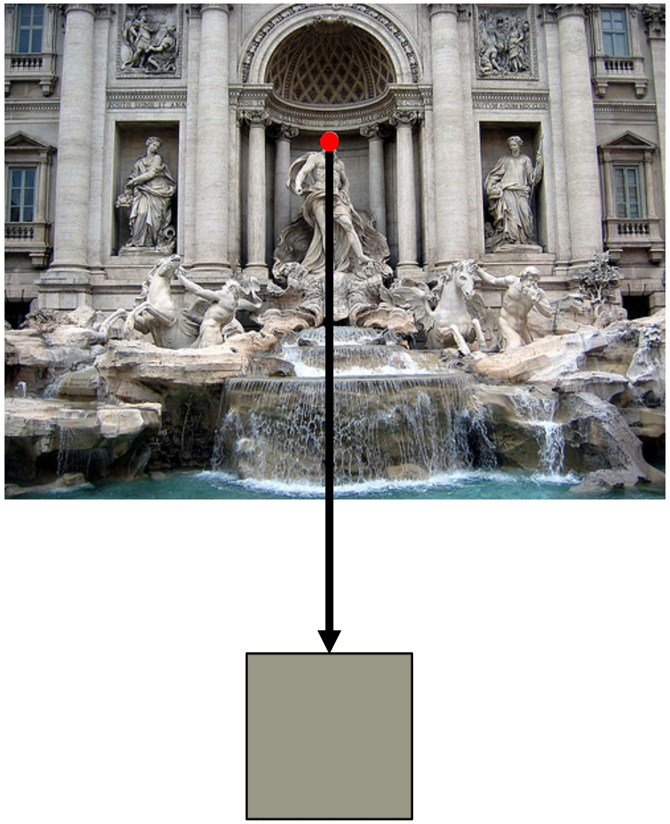
- Most feature descriptors are designed to be invariant to
  - Translation, 2D rotation, scale
- They can usually also handle
  - Limited 3D rotations (SIFT works up to about 60 degrees)
  - Limited affine transformations (some are fully affine invariant)
  - Limited illumination/contrast changes

How to achieve invariance

Design an invariant feature descriptor

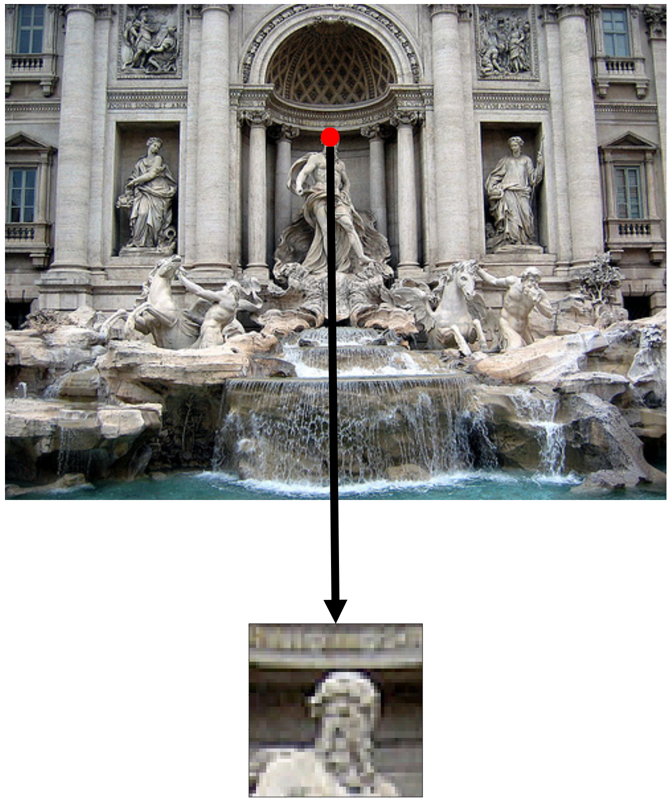
- Simplest descriptor: a single 0
  - What's this invariant to?
  - Is this discriminative?
- Next simplest descriptor: a single pixel
  - What's this invariant to?
  - Is this discriminative?

The aperture problem



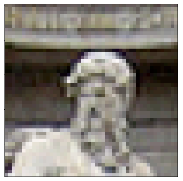
The aperture problem

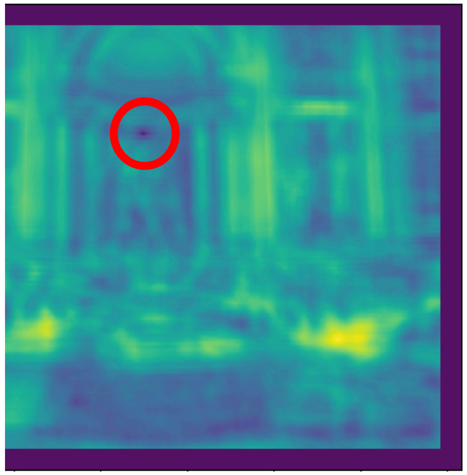
- Use a whole patch instead of a pixel?



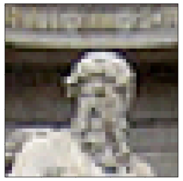
SSD

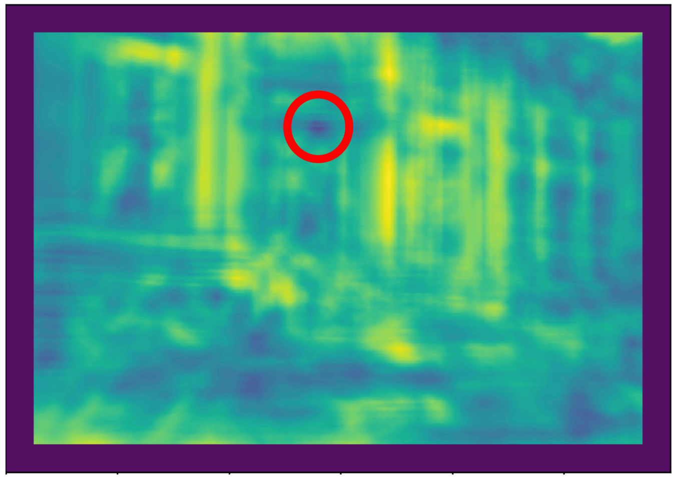
- Use as descriptor the whole patch
- Match descriptors using euclidean distance
- $d(x, y) = ||x - y||^2$



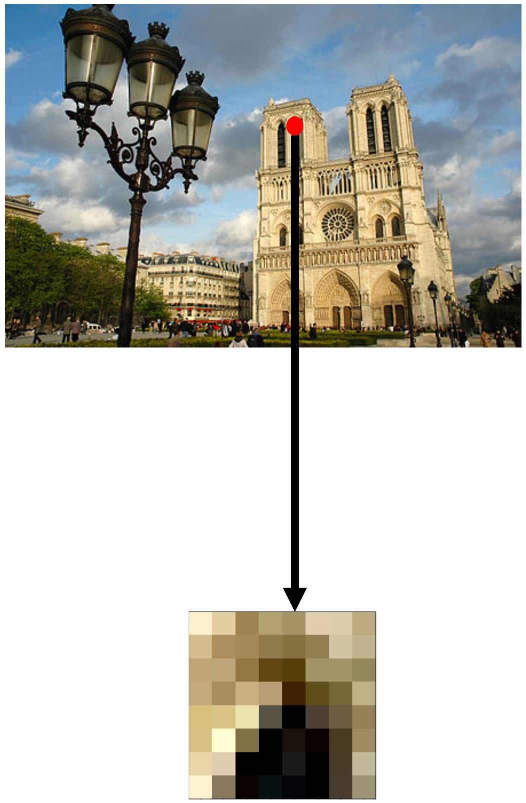


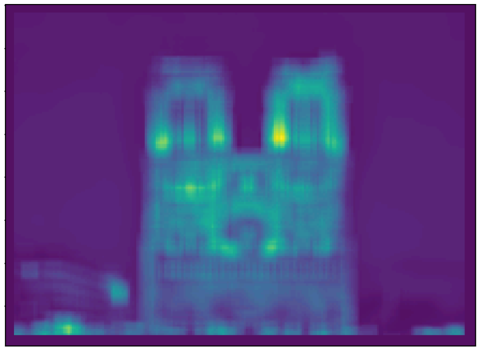
SSD





SSD

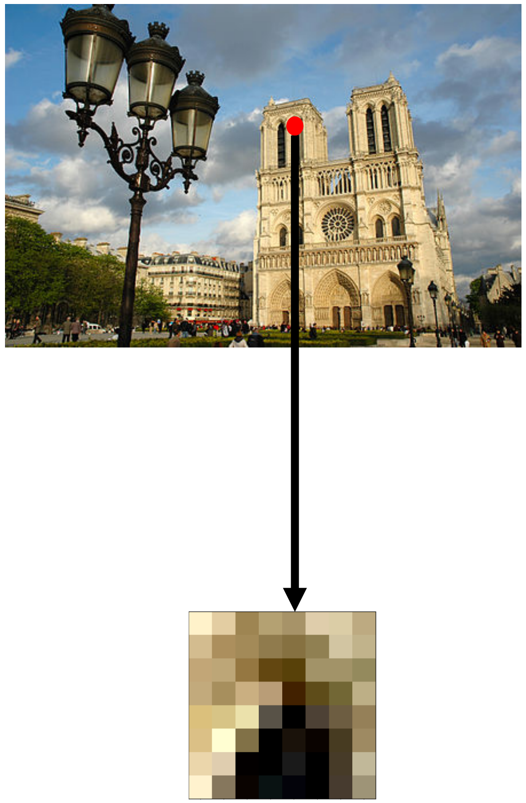


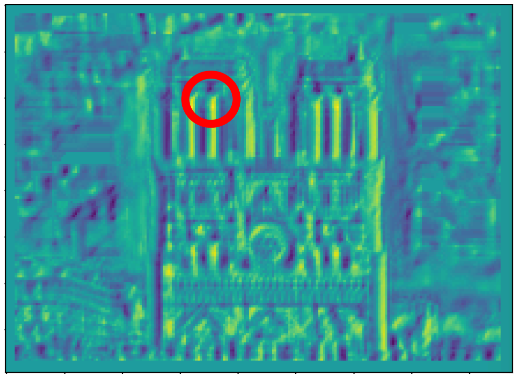


NCC - Normalized Cross Correlation

- Lighting and color change pixel intensities
- Example: increase brightness / contrast
- $I' = \alpha I + \beta$
- Subtract patch mean: invariance to β
- Divide by norm of vector: invariance to α
- $x' = x - < x >$
- $x'' = \frac{x'}{||x'||}$
- *similarity* = $x'' \cdot y''$

NCC - Normalized cross correlation





Basic correspondence

- Image patch as descriptor, NCC as similarity
- Invariant to?
  - Photometric transformations?
  - Translation?
  - Rotation?

Rotation invariance for feature descriptors

- Find dominant orientation of the image patch
  - This is given by $\mathbf{x}_{\mathbf{max}}$, the eigenvector of **M** corresponding to $\lambda_{\max}$ (the *larger* eigenvalue)
  - Rotate the patch according to this angle

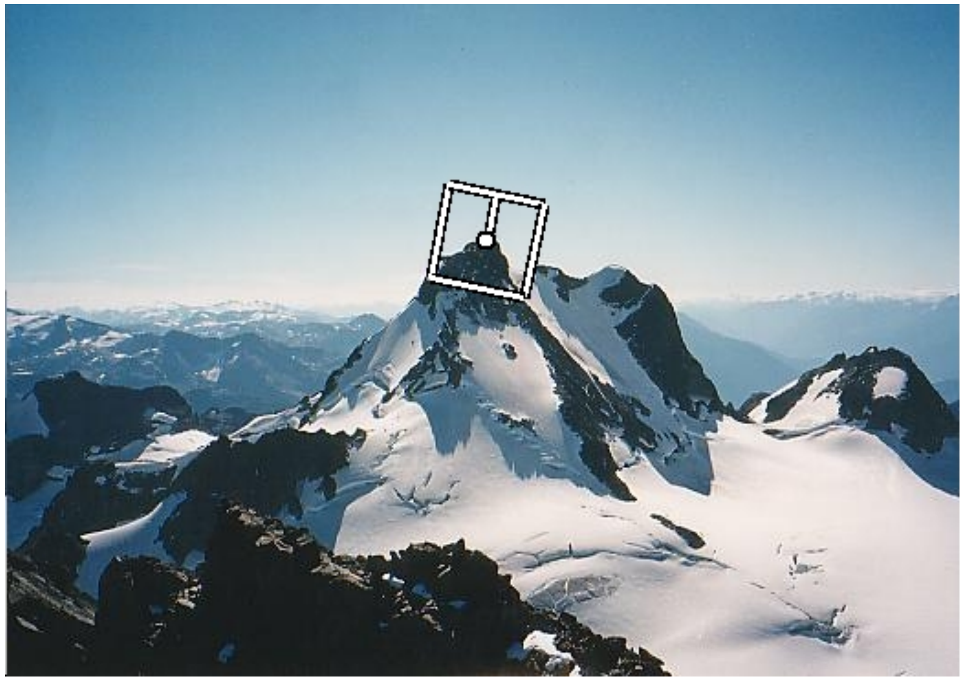
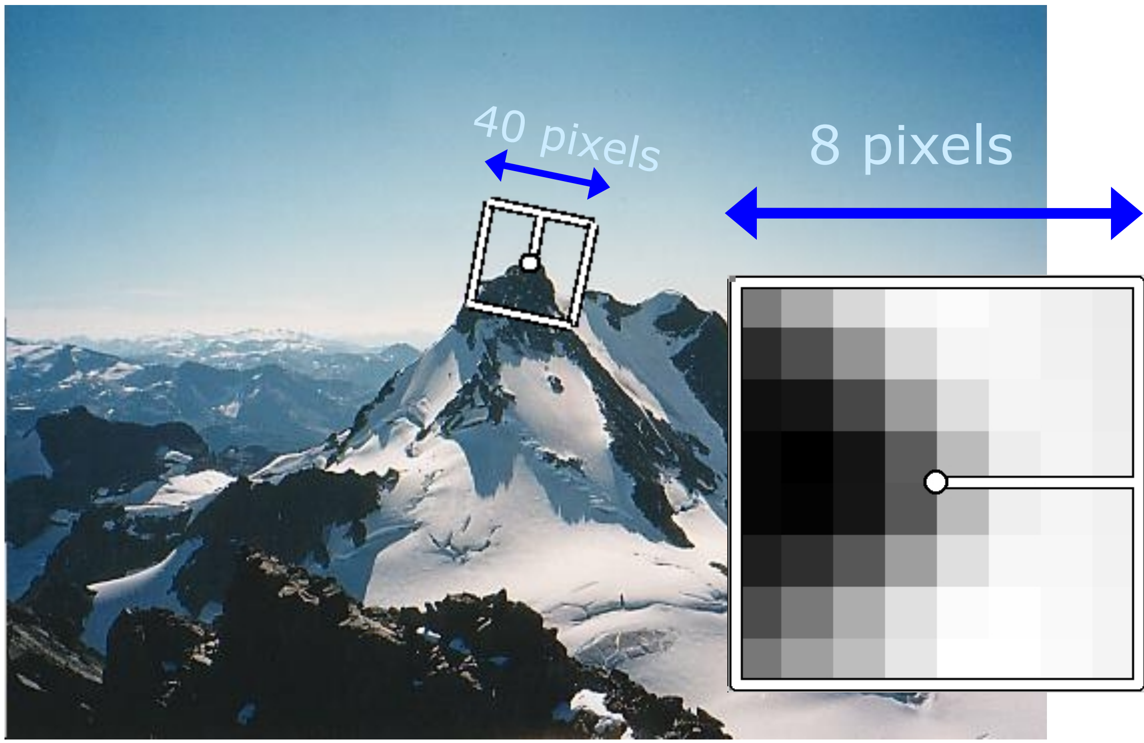


Figure by Matthew Brown

Multiscale Oriented PatcheS descriptor

Take 40x40 square window around detected feature

- Scale to 1/5 size (using prefiltering)
- Rotate to horizontal
- Sample 8x8 square window centered at feature
- Intensity normalize the window by subtracting the mean, dividing by the standard deviation in the window



Adapted from slide by Matthew Brown

Detections at multiple scales

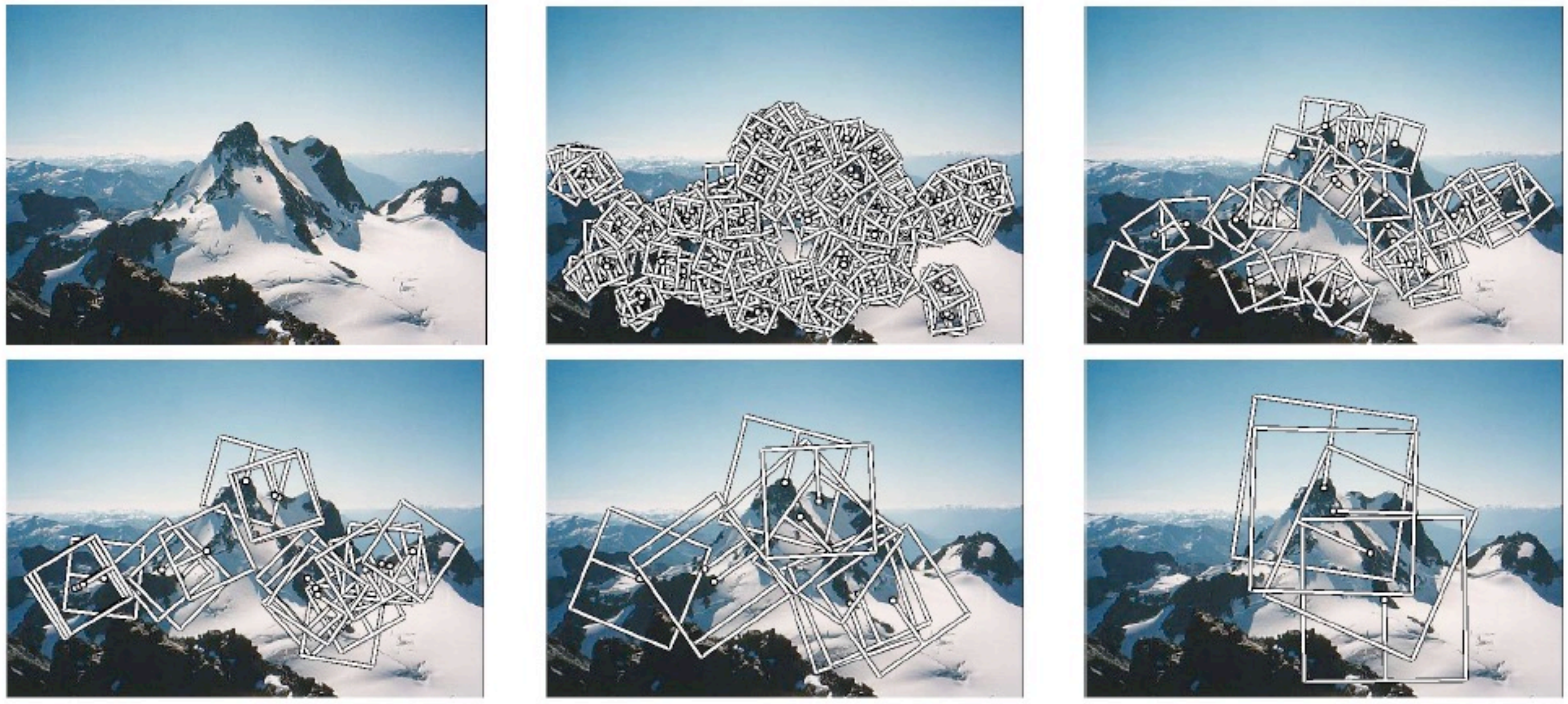


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

Rotation invariance for feature descriptors

- Find dominant orientation of the image patch
  - This is given by $\mathbf{x}_{\mathbf{max}}$, the eigenvector of **M** corresponding to λ_{max} (the *larger* eigenvalue)
  - Rotate the patch according to this angle

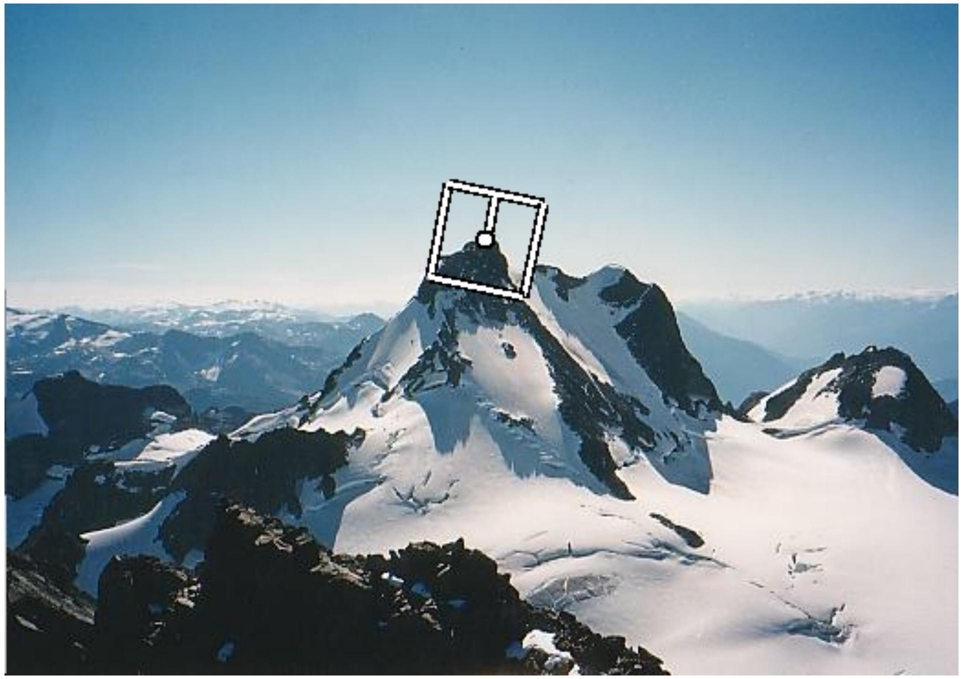


Figure by Matthew Brown

Detour: Image transformations

- What does it mean to rotate a patch?
- Each pixel has coordinates (x,y)
- Rotation represented by a matrix R
- Pixel’s new coordinates:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = R \begin{bmatrix} x \\ y \end{bmatrix}$$

- $I'(x',y') = I(x,y)$

Detour: Image transformations

- What if destination pixel is fractional?
- Flip computation: for every destination pixel figure out source pixel
  - Use interpolation if source location is fractional

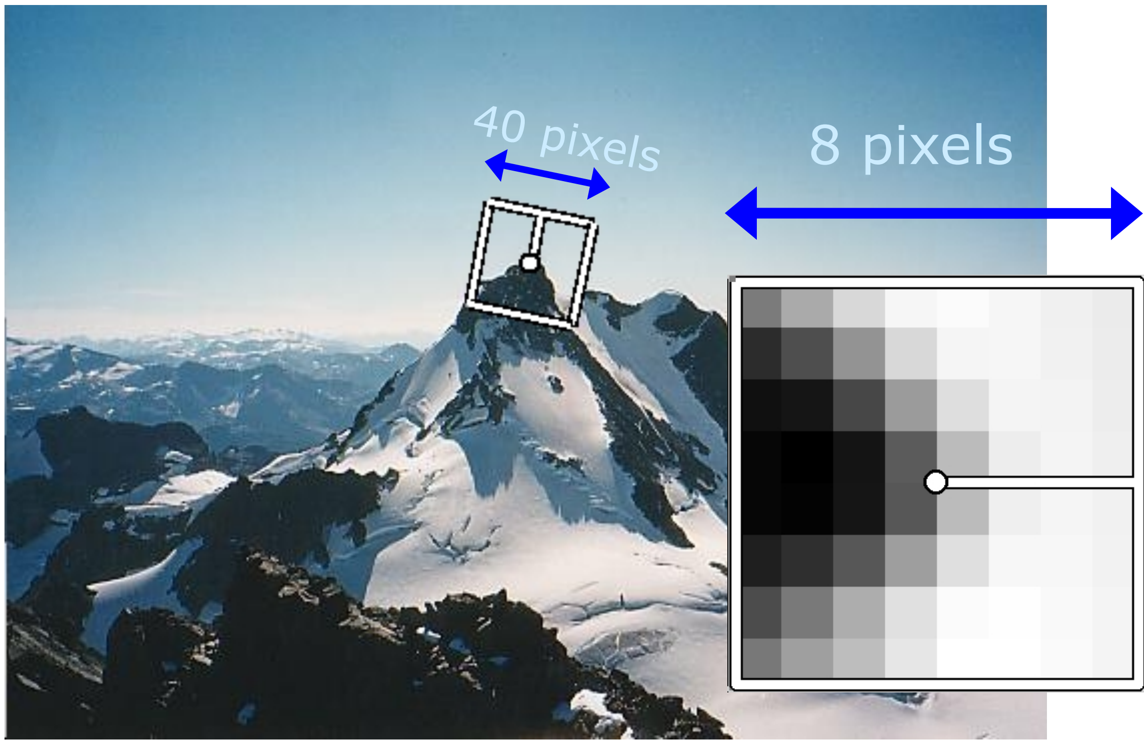
$$\begin{bmatrix} x \\ y \end{bmatrix} = R^{-1} \begin{bmatrix} x' \\ y' \end{bmatrix}$$

- I'(x', y') = I(x,y)

Multiscale Oriented PatcheS descriptor

Take 40x40 square window around detected feature

- Scale to 1/5 size (using prefiltering)
- Rotate to horizontal
- Sample 8x8 square window centered at feature
- Intensity normalize the window by subtracting the mean, dividing by the standard deviation in the window

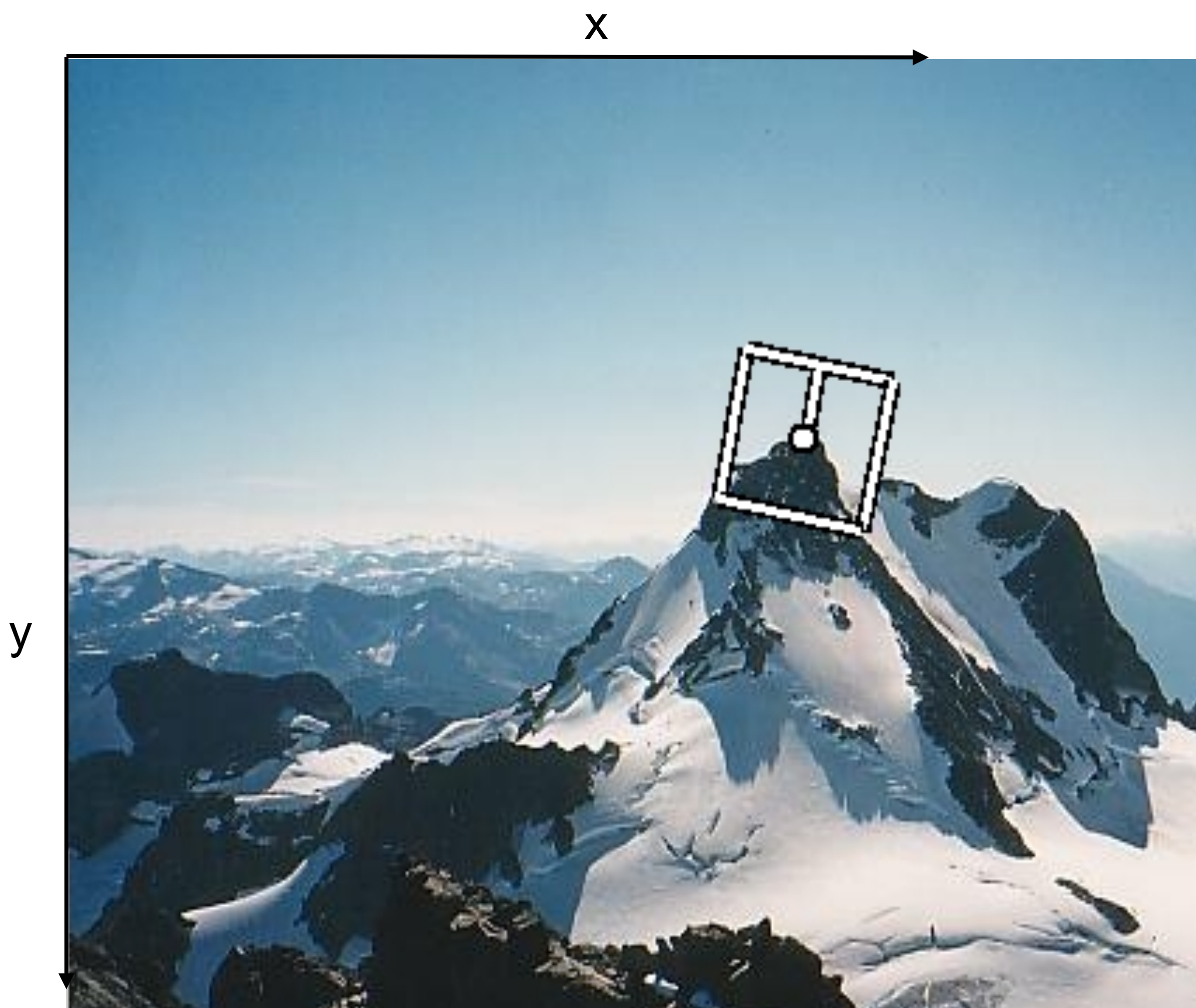


Adapted from slide by Matthew Brown

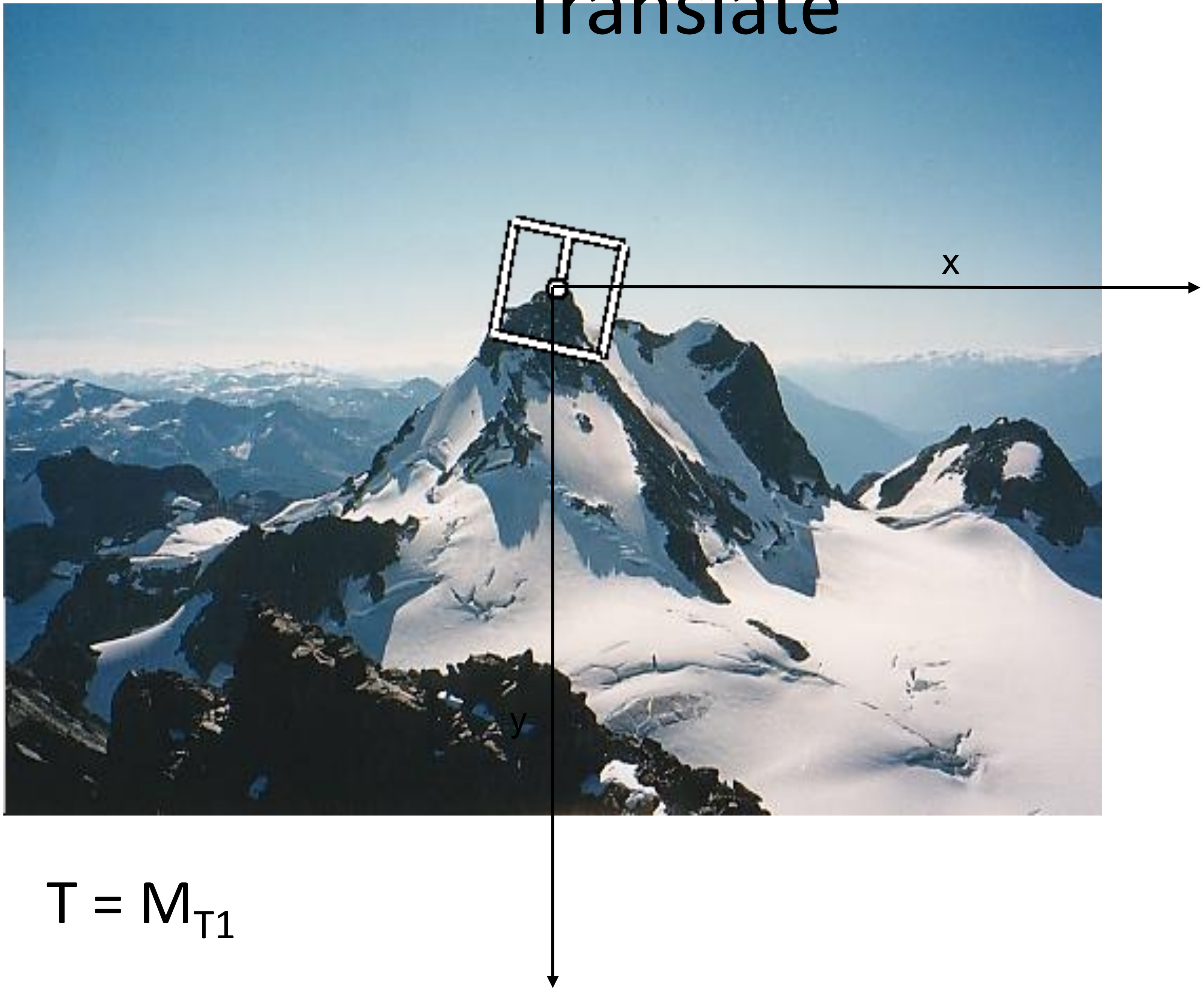
MOPS descriptor

- You can combine transformations together to get the final transformation

T = ?

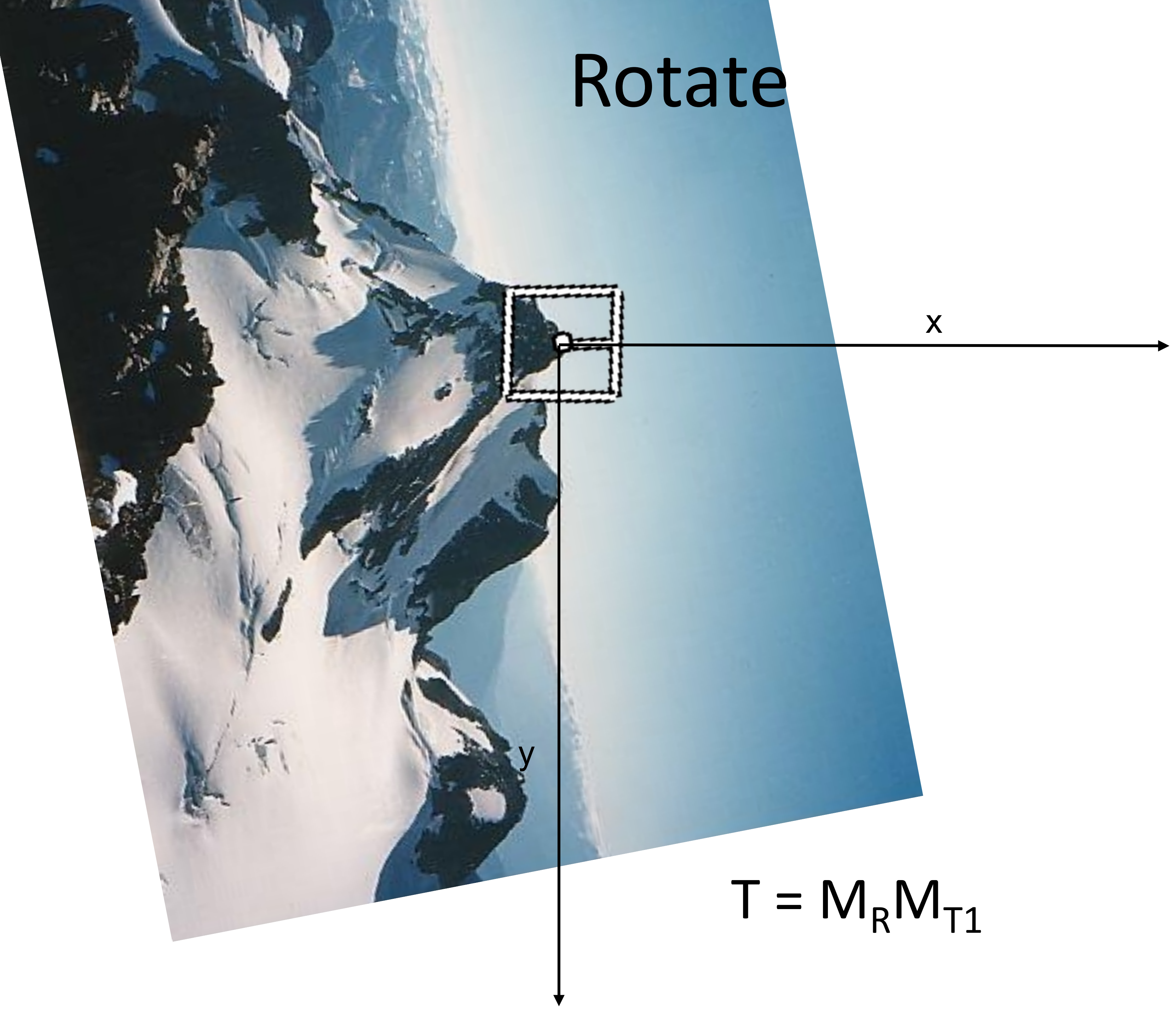


Translate



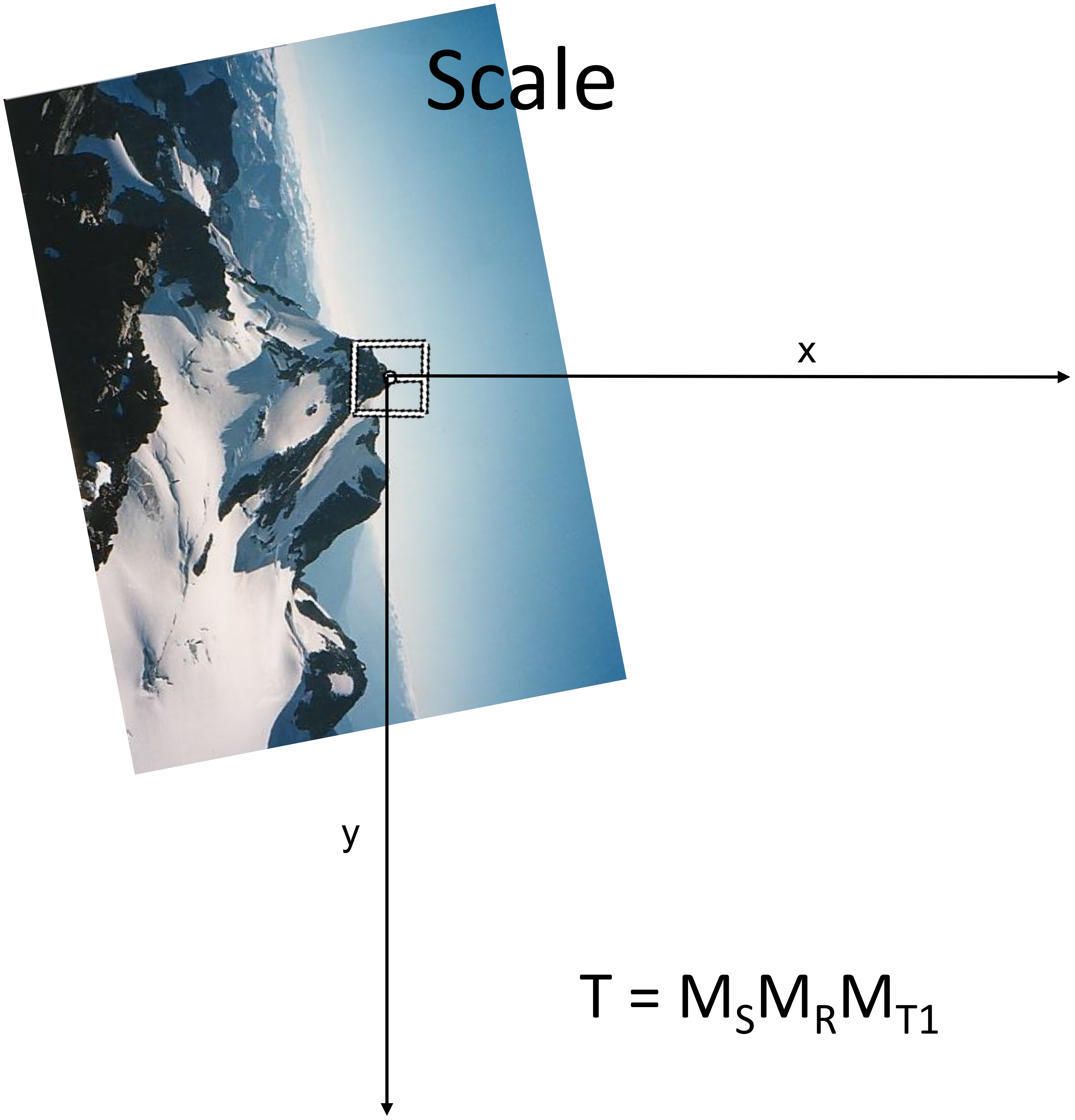
$T = M_{T1}$

Rotate



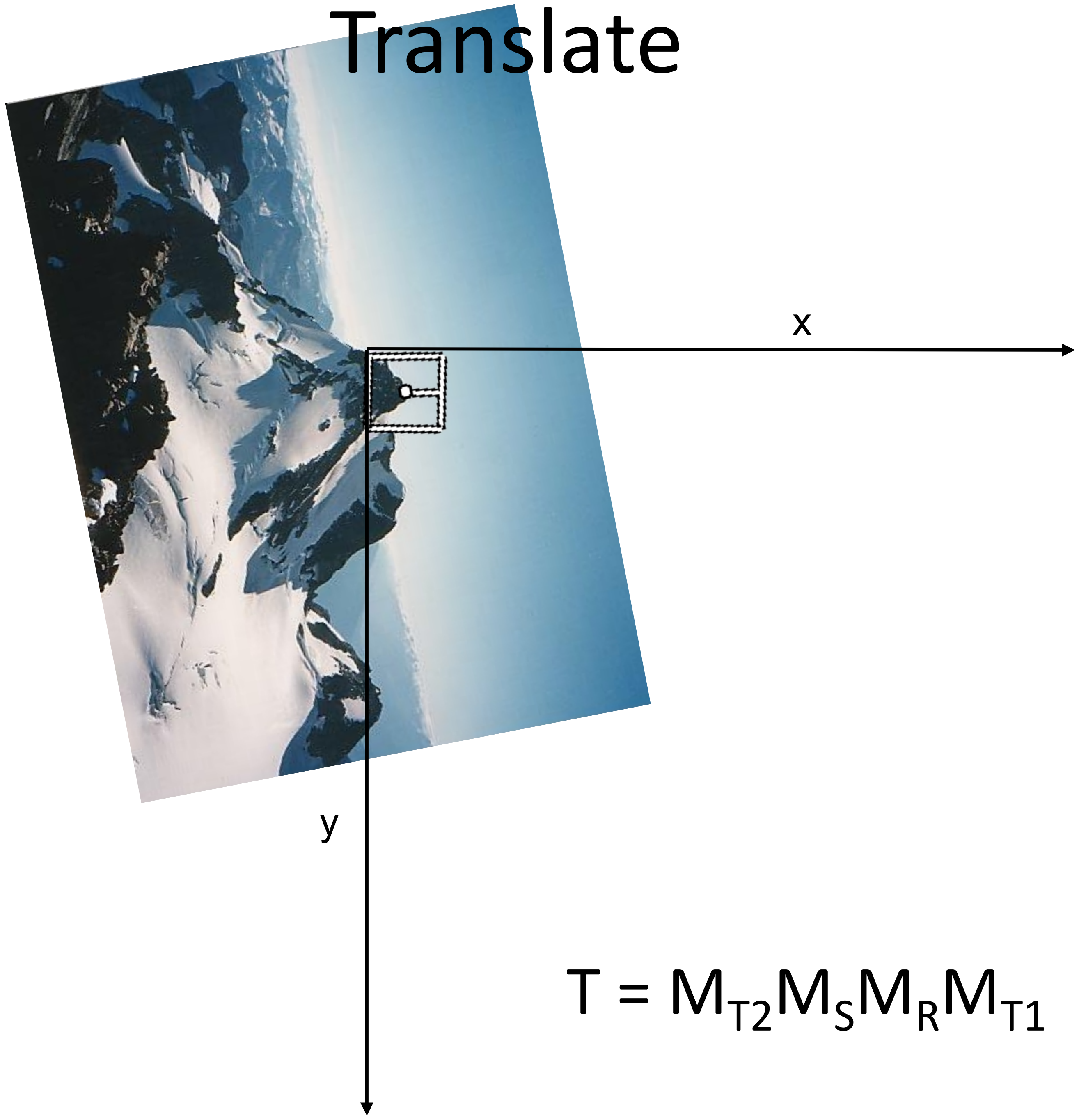
$T = M_R M_{T1}$

Scale



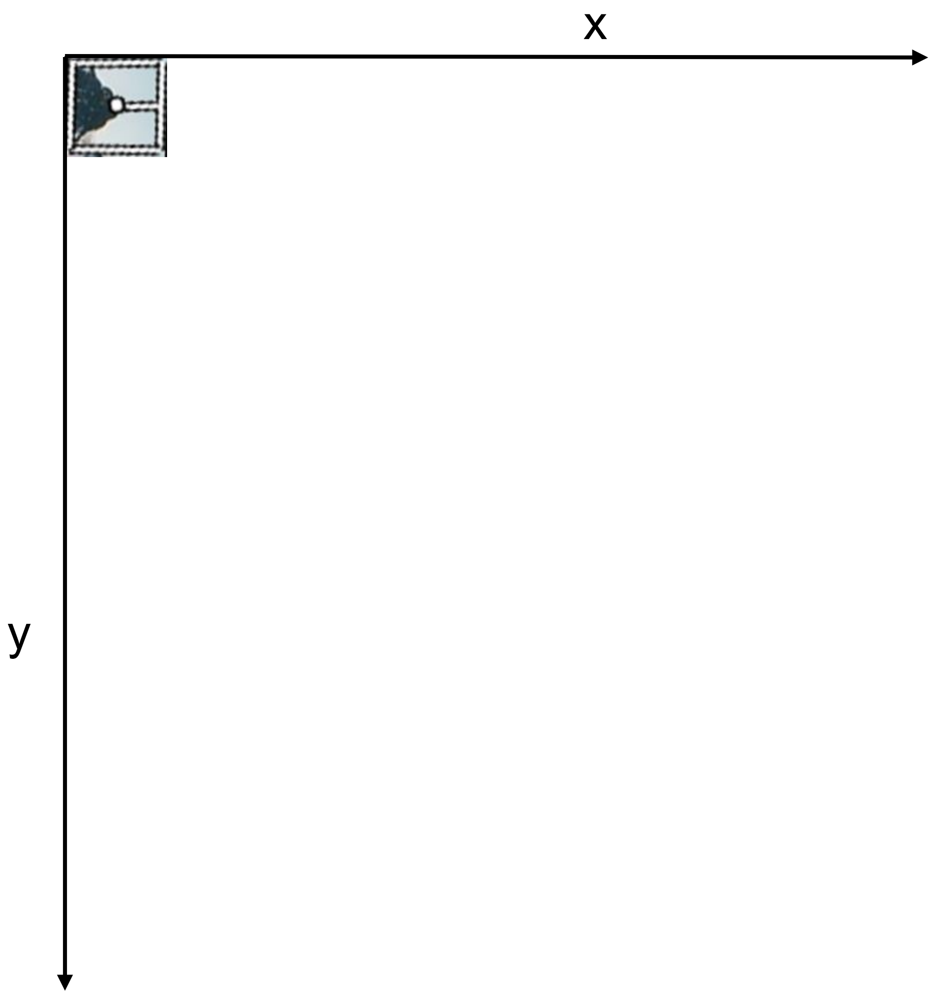
$$T = M_S M_R M_{T1}$$

Translate



$$T = M_{T2} M_S M_R M_{T1}$$

Crop



Detections at multiple scales

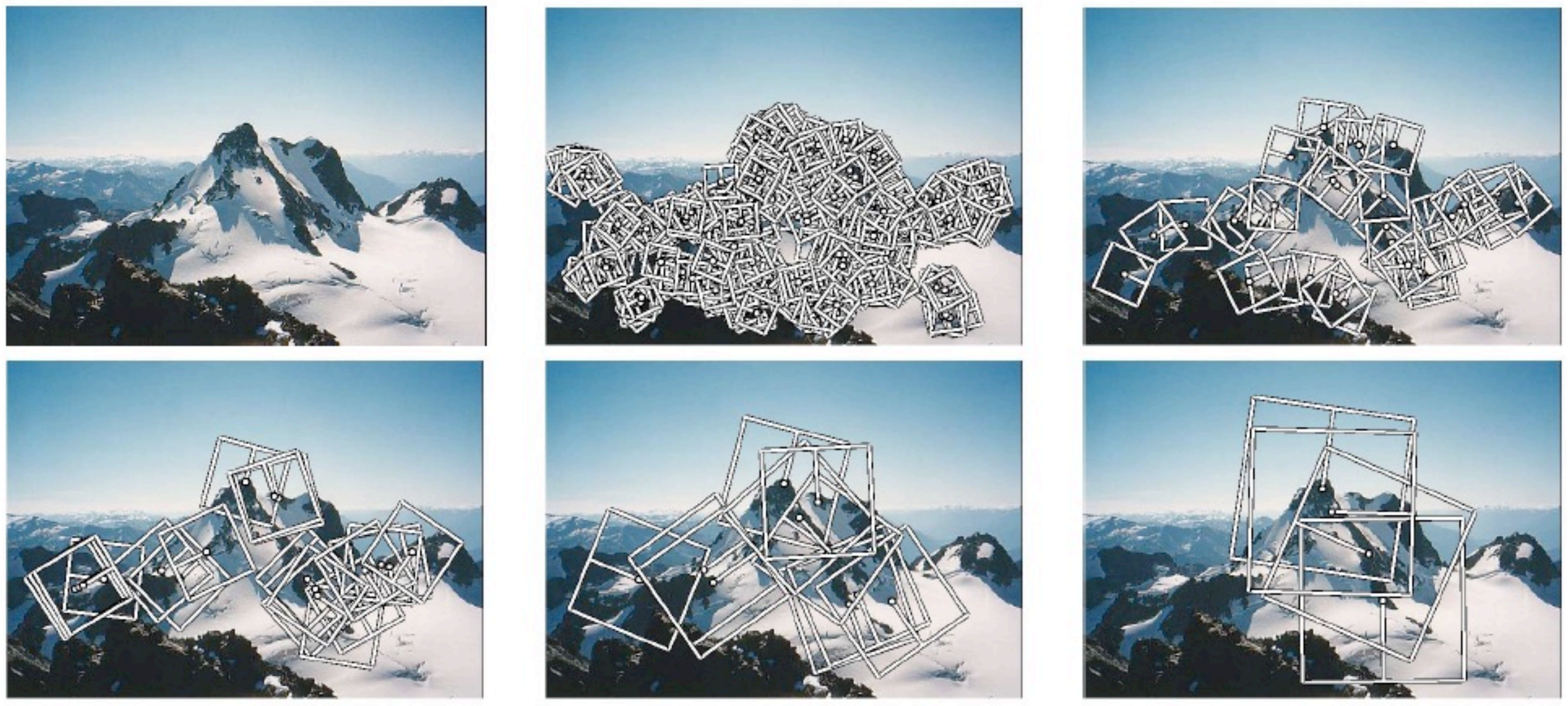
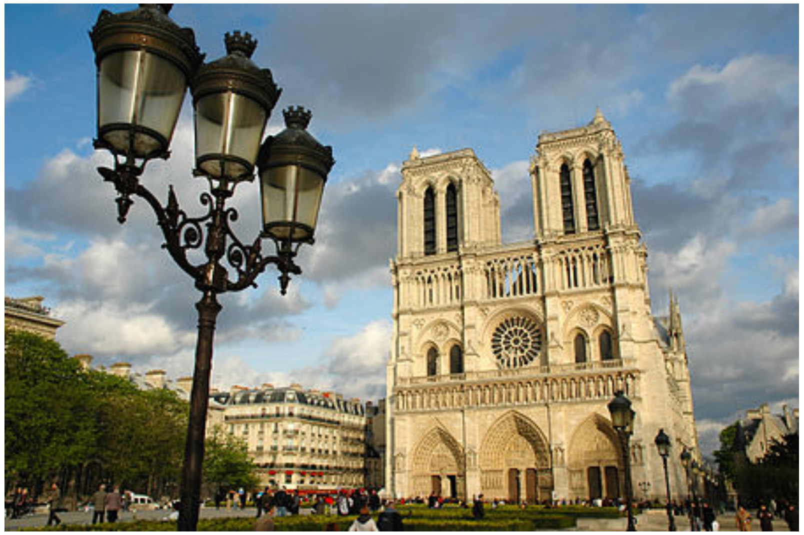


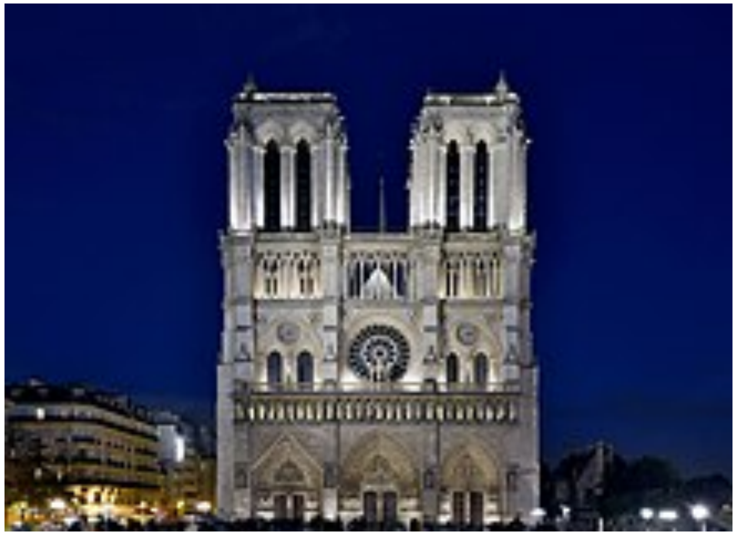
Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

Invariance of MOPS

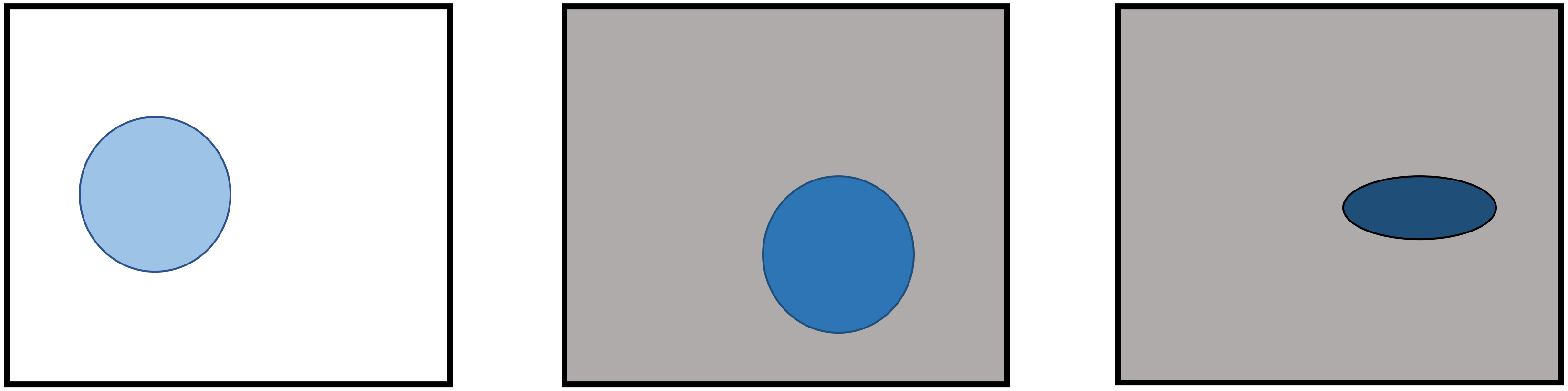
- Intensity
- Scale
- Rotation

Color and Lighting

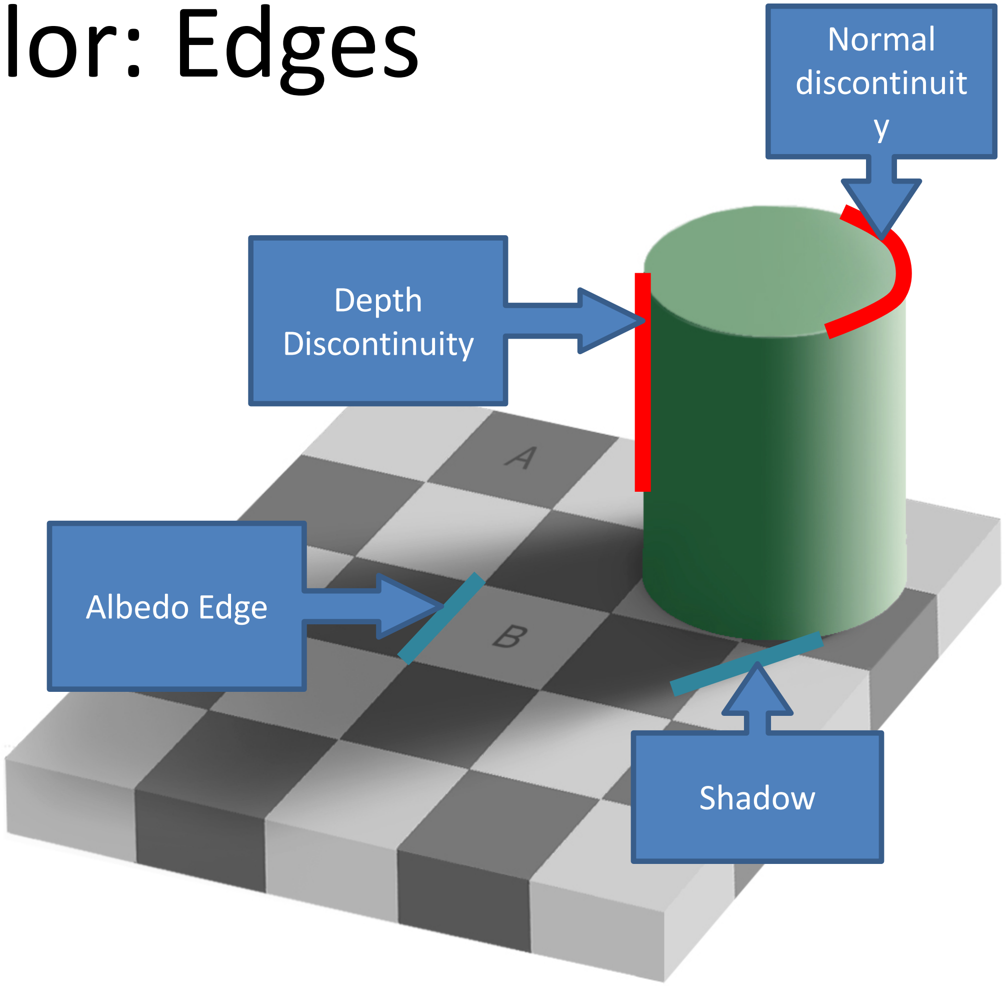




Out-of-plane rotation


Out-of-plane rotation

Better representation than color: Edges

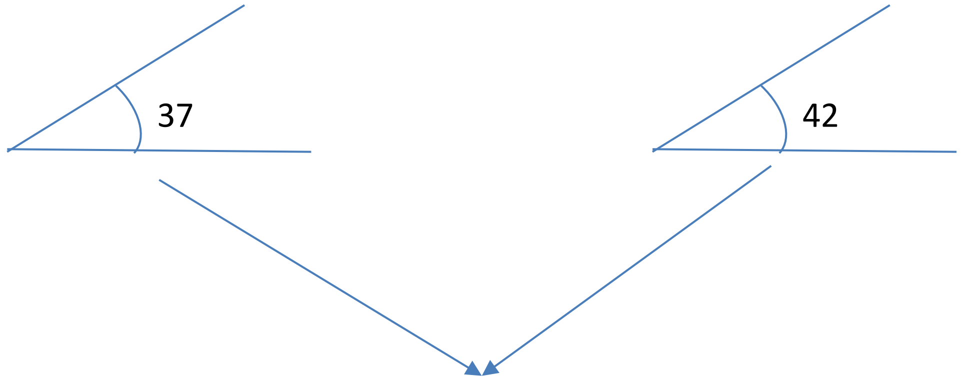


Towards a better feature descriptor

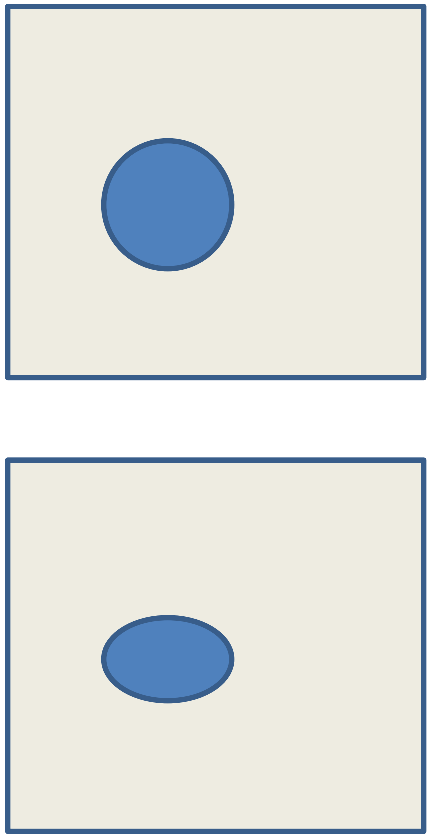
- Match *pattern of edges*
  - Edge orientation – clue to shape

- Be resilient to *small deformations*
  - Deformations might move pixels around, but slightly
  - Deformations might change edge orientations, but slightly

Invariance to deformation by quantization



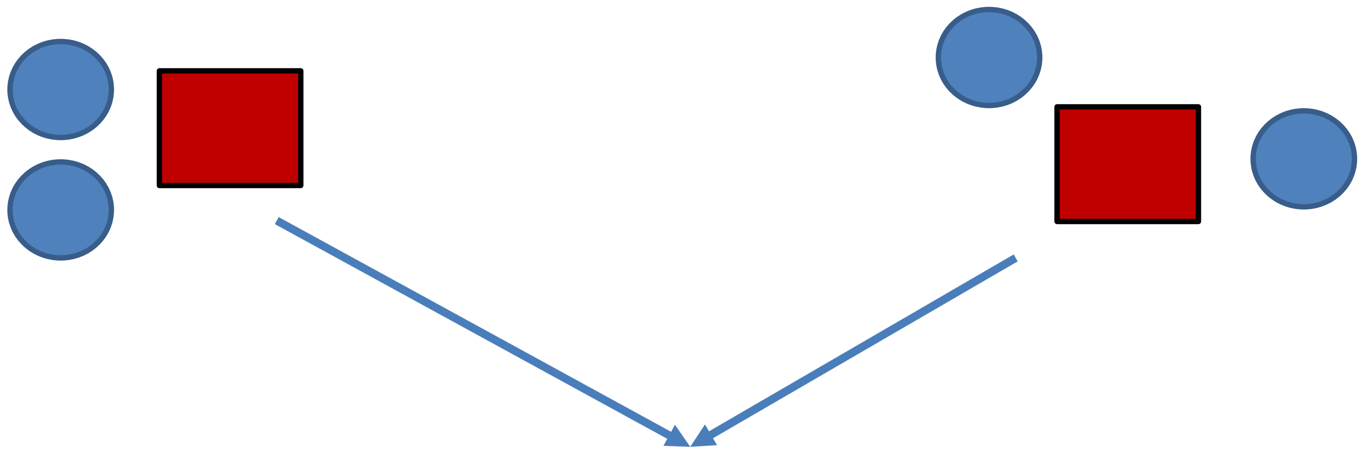
Between 30 and 45



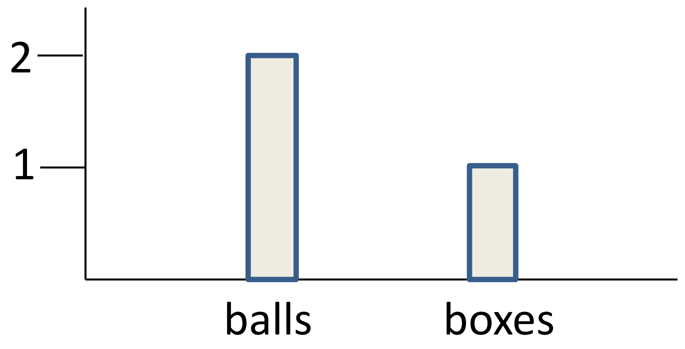
Invariance to deformation by quantization

$$g(\theta) = \begin{cases} 0 & \text{if } 0 < \theta < 2\pi/N \\ 1 & \text{if } 2\pi/N < \theta < 4\pi/N \\ 2 & \text{if } 4\pi/N < \theta < 6\pi/N \\ & \dots \\ N-1 & \text{if } 2(N-1)\pi/N \end{cases}$$

Spatial invariance by histograms



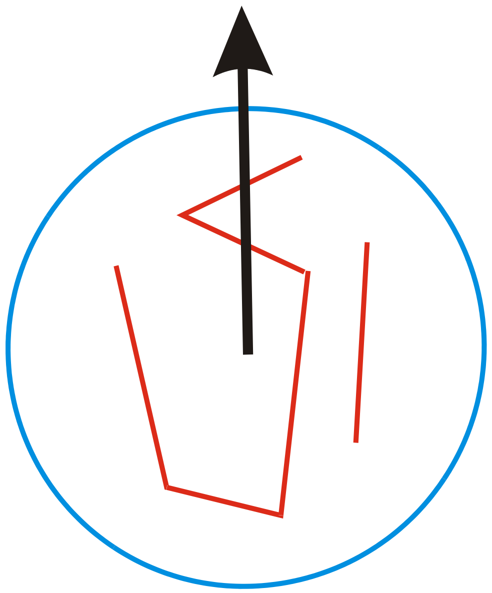
2 blue balls, one red box

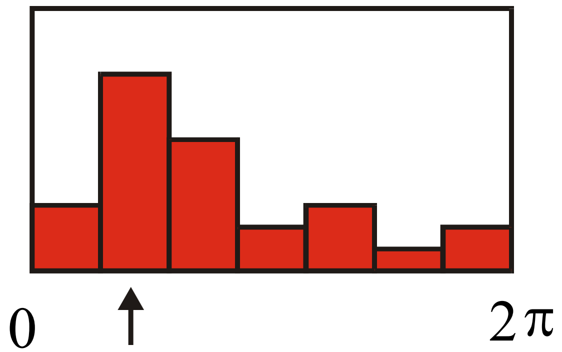


Rotation Invariance by Orientation Normalization

[Lowe, SIFT, 1999]

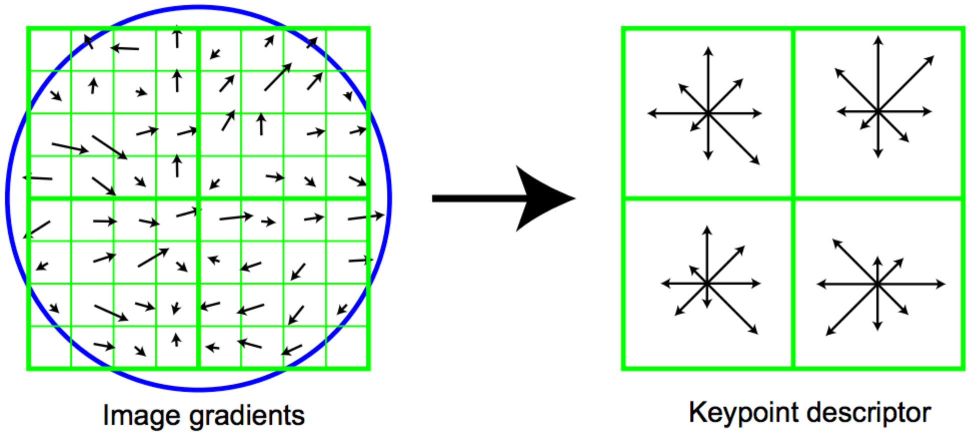
- Compute orientation histogram
- Select dominant orientation
- Normalize: rotate to fixed orientation





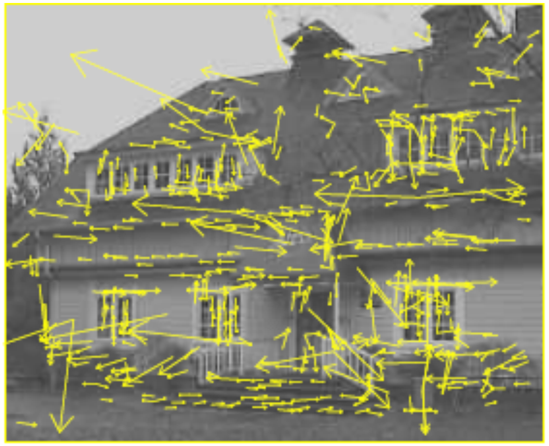
The SIFT descriptor

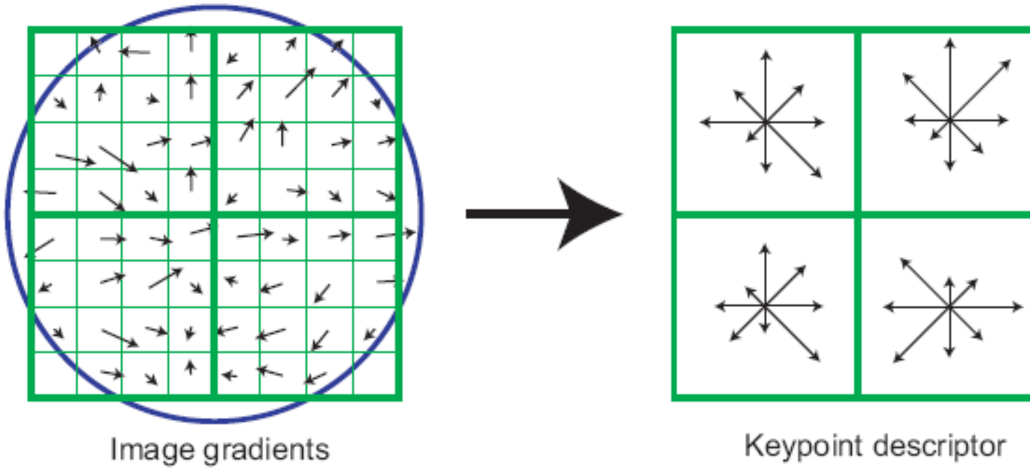
- Compute edge magnitudes + orientations
- Quantize orientations *(invariance to def)*
- Divide into spatial cells
- Compute orientation histogram in each cell *(spatial invariance)*



Distinctive Image Features from Scale-Invariant Keypoints. Lowe. In IJCV 2004

The SIFT descriptor





SIFT – Lowe IJCV 2004

Scale Invariant Feature Transform

Basic idea:

- DoG for scale-space feature detection
- Take 16x16 square window around detected feature
  - Compute gradient orientation for each pixel
  - Throw out weak edges (threshold gradient magnitude)
  - Create histogram of surviving edge orientations

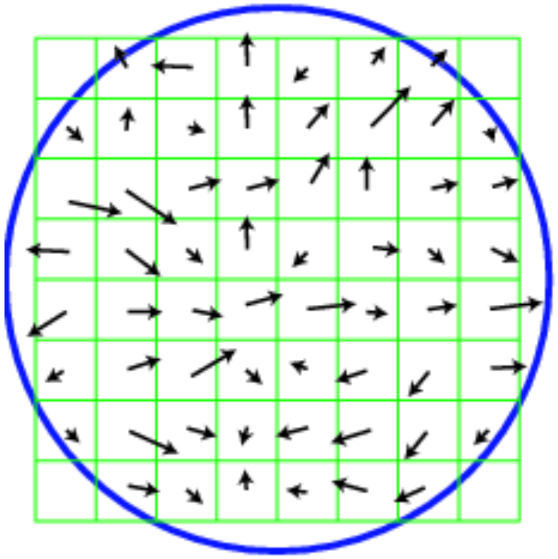
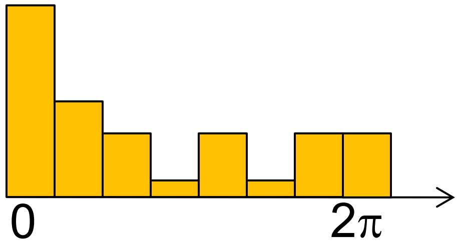
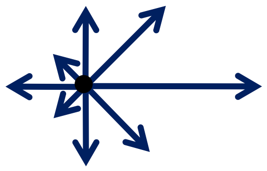


Image gradients



angle histogram

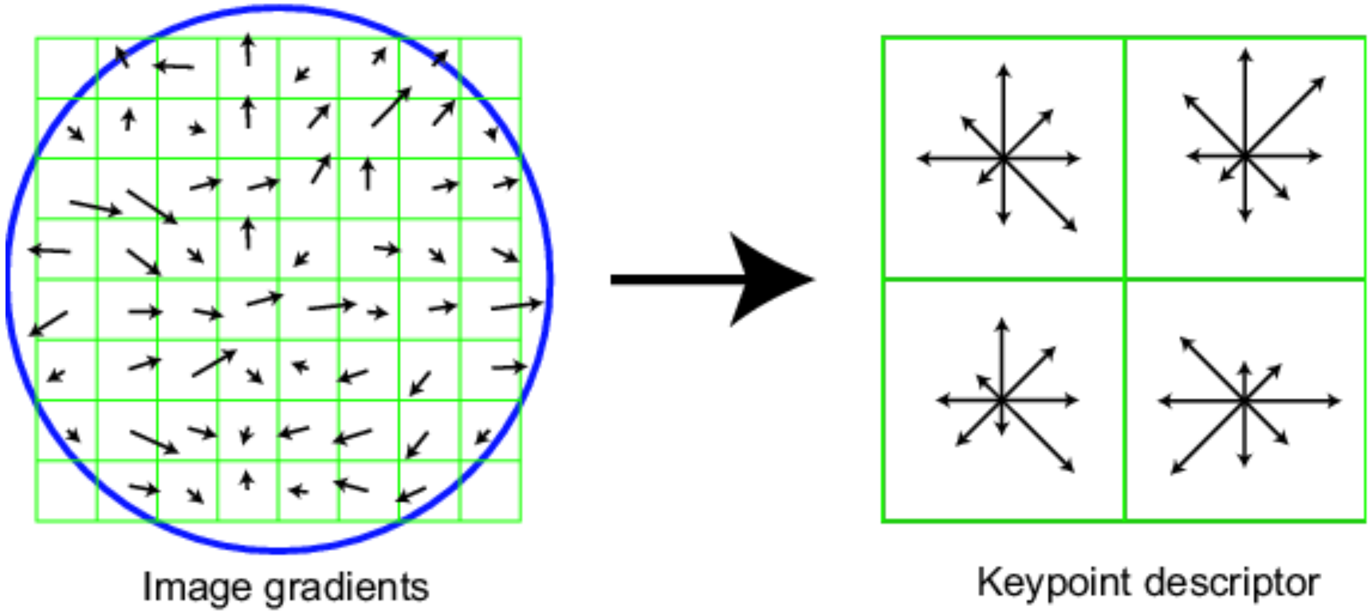

Keypoint descriptor

Adapted from slide by David Lowe

SIFT descriptor

Create histogram

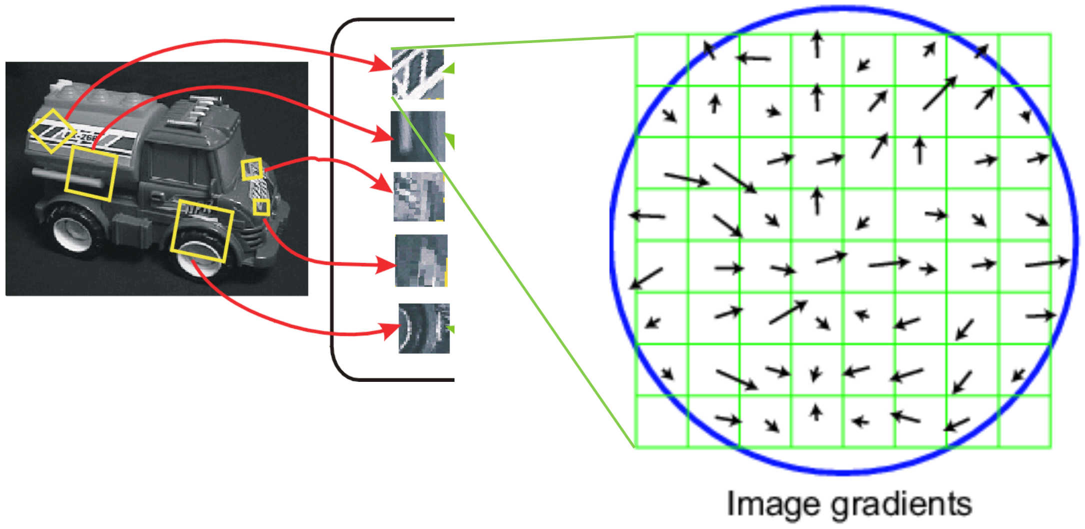
- Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
- Compute an orientation histogram for each cell
- 16 cells * 8 orientations = 128 dimensional descriptor



Adapted from slide by David Lowe

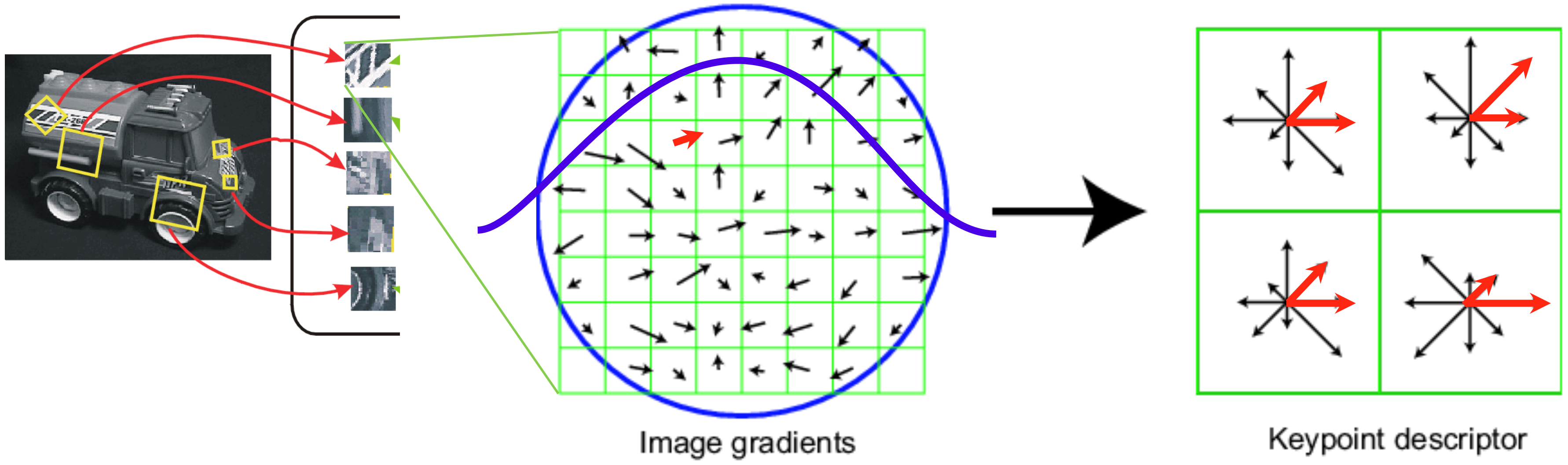
SIFT vector formation

- Computed on rotated and scaled version of window according to computed orientation & scale
  - resample the window
- Based on gradients weighted by a Gaussian



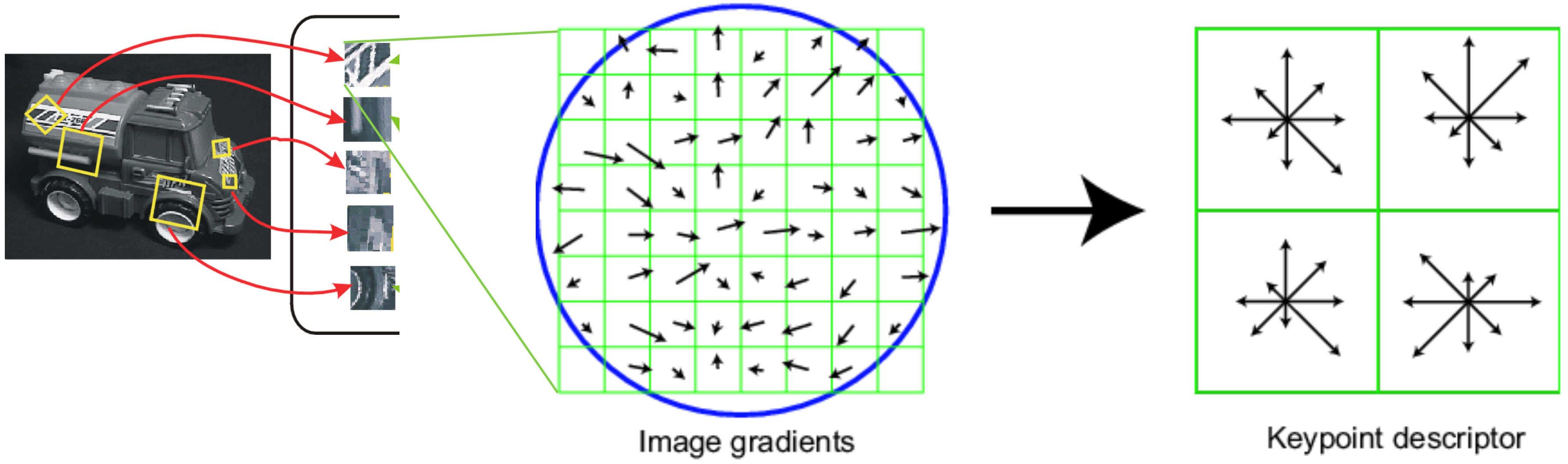
Ensure smoothness

- Trilinear interpolation
  - a given gradient contributes to 8 bins: 4 in space times 2 in orientation



Reduce effect of illumination

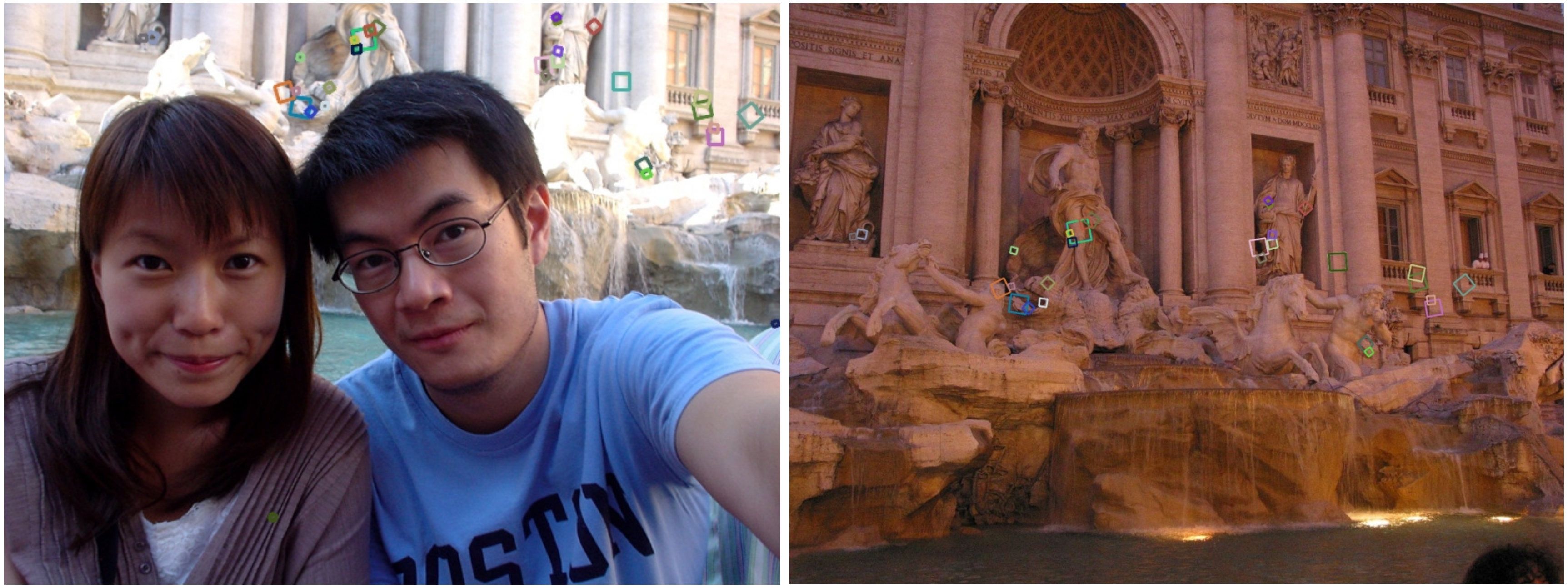
- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
  - after normalization, clamp gradients >0.2
  - renormalize



Properties of SIFT

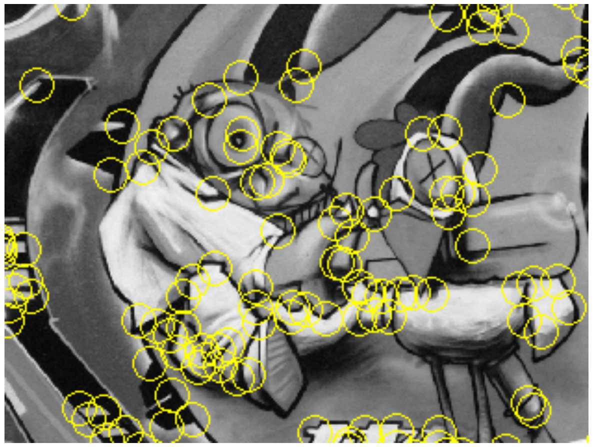
Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time
- Lots of code available: http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT

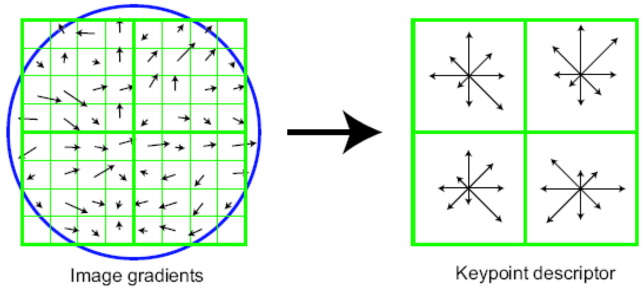


Summary

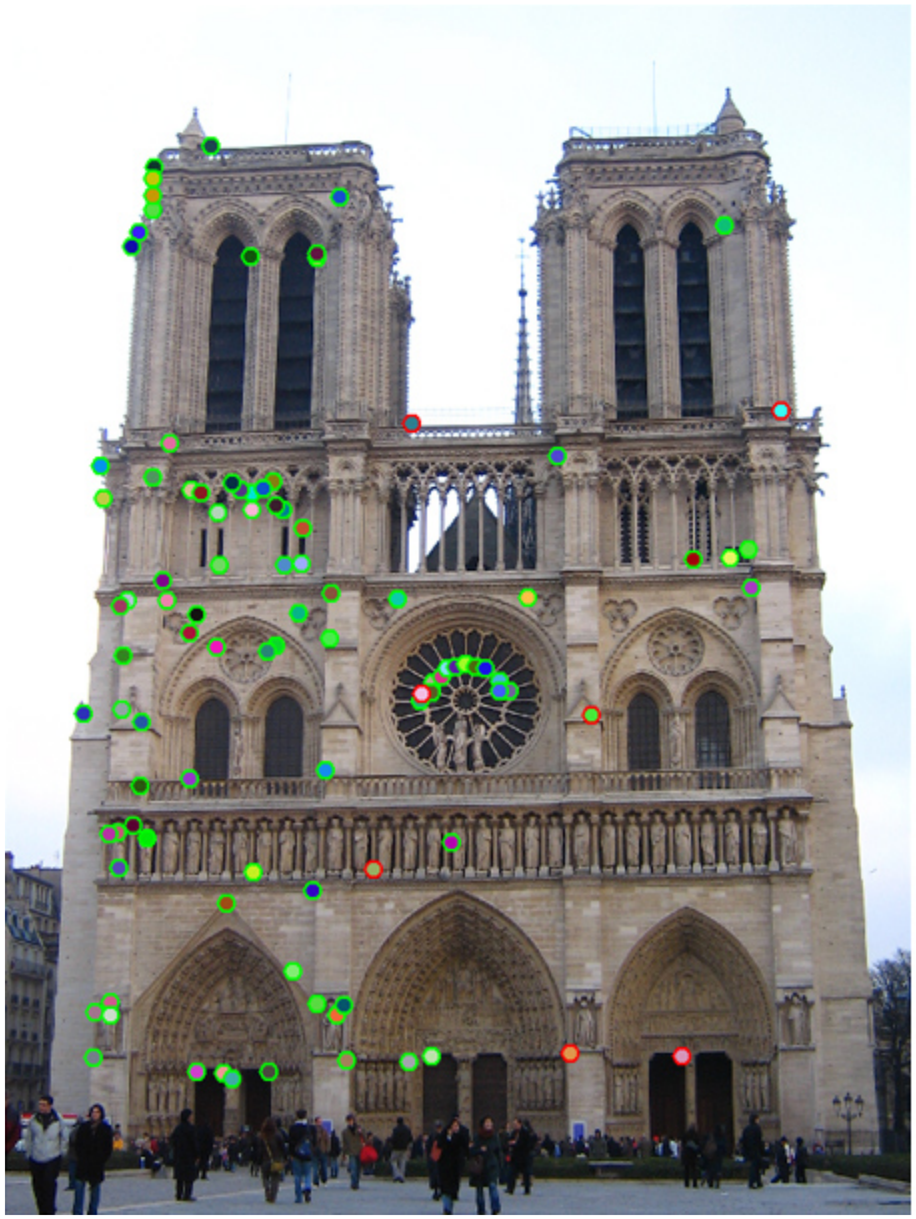
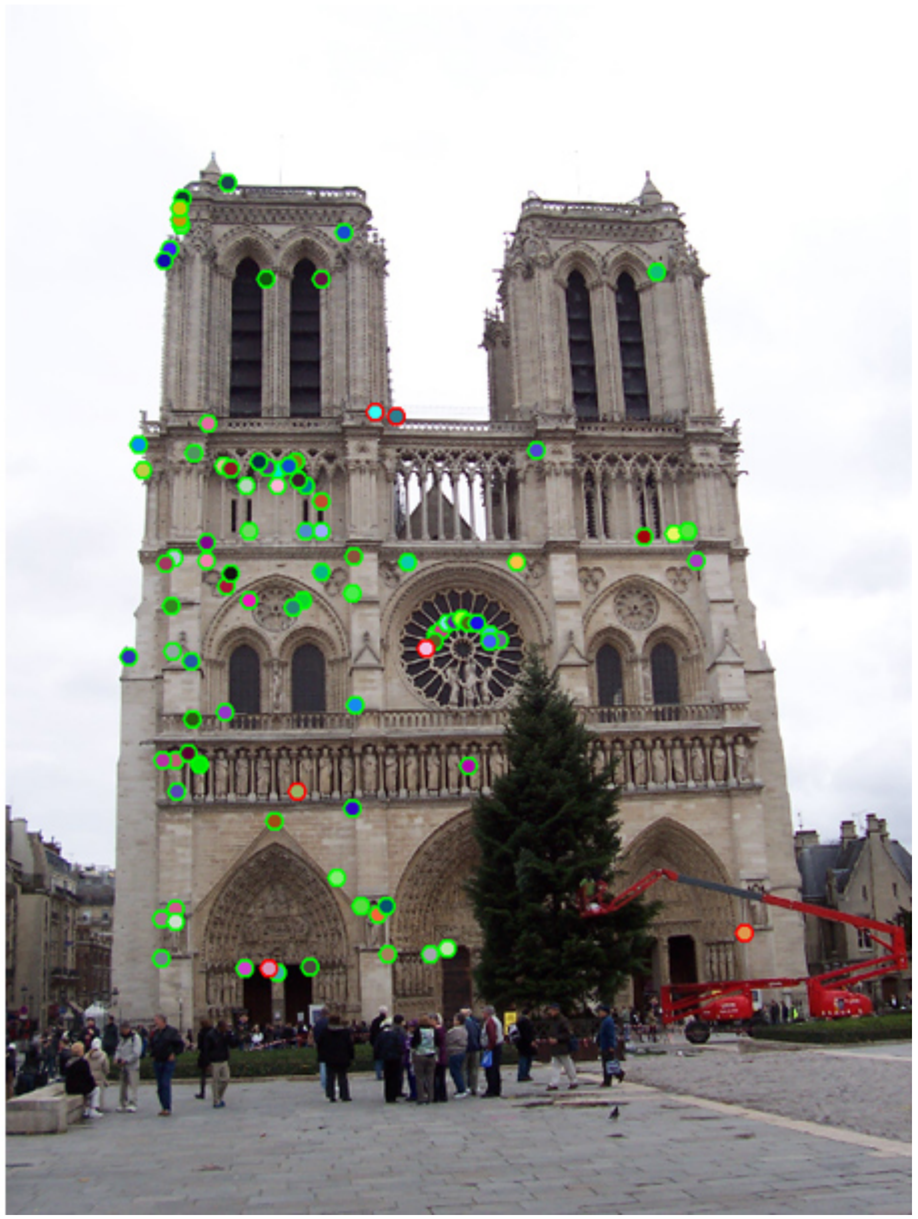
- Keypoint detection: repeatable and distinctive
  - Corners, blobs, stable regions
  - Harris, DoG



- Descriptors: robust and selective
  - spatial histograms of orientation
  - SIFT and variants are typically good for stitching and recognition
  - But, need not stick to one



Which features match?

Feature matching

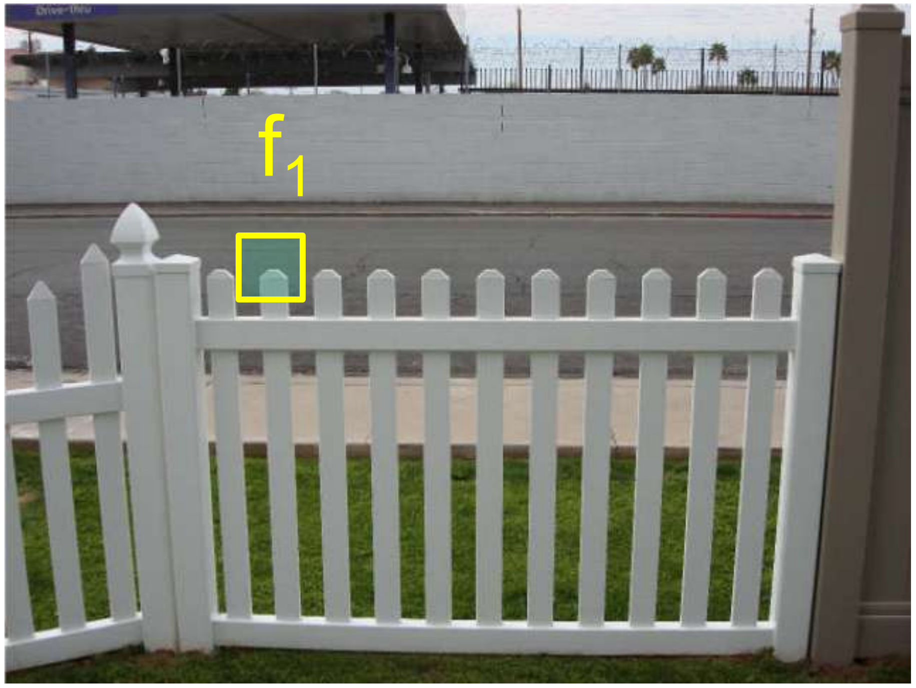
Given a feature in I_1, how to find the best match in I_2?

1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with min distance

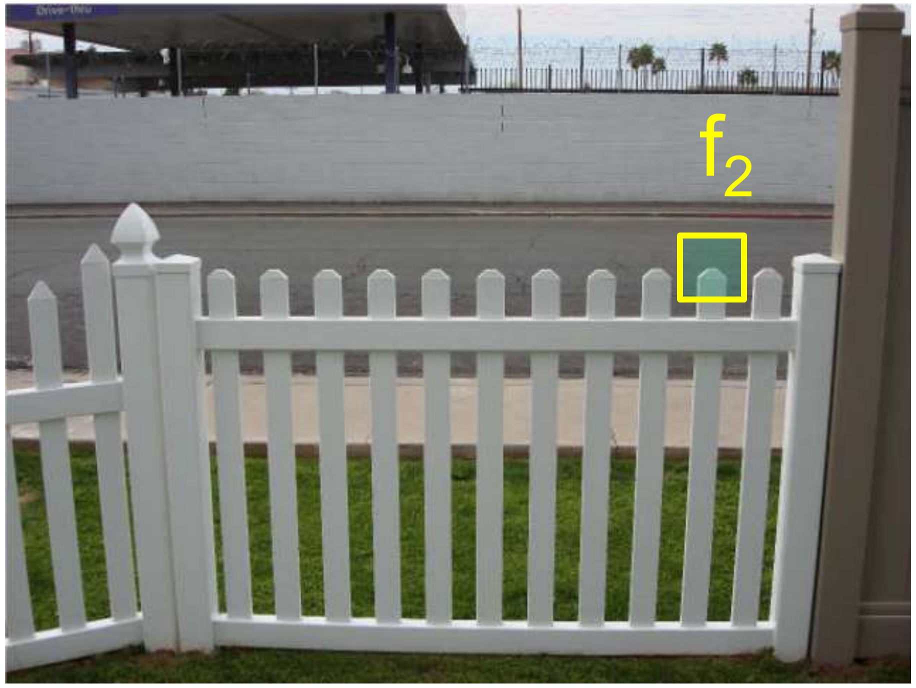
Feature distance

How to define the difference between two features f_1, f_2?

- Simple approach: L_2 distance, $||f_1 - f_2||$
- can give good scores to ambiguous (incorrect) matches



I_1

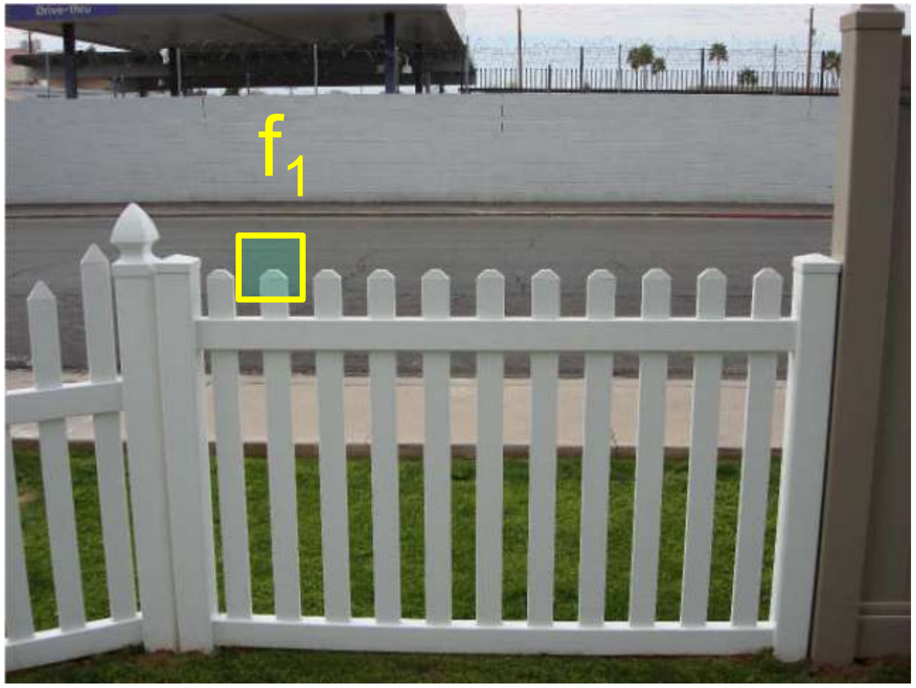


I_2

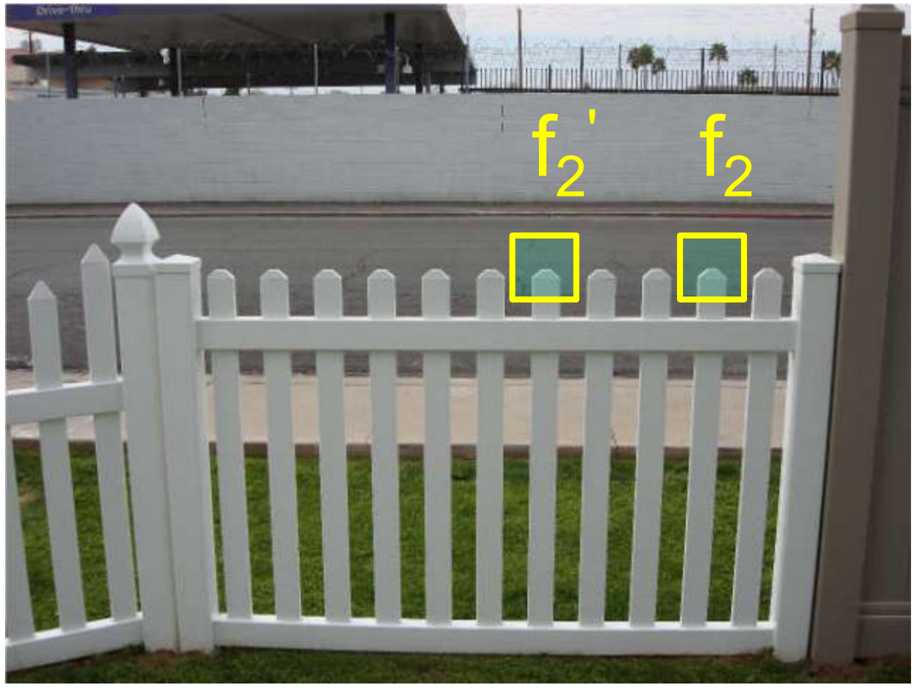
Feature distance

How to define the difference between two features f_1, f_2?

- Better approach: ratio distance = $||f_1 - f_2|| / ||f_1 - f_2'||$
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2nd best SSD match to f_1 in I_2
  - gives large values for ambiguous matches



I_1



I_2