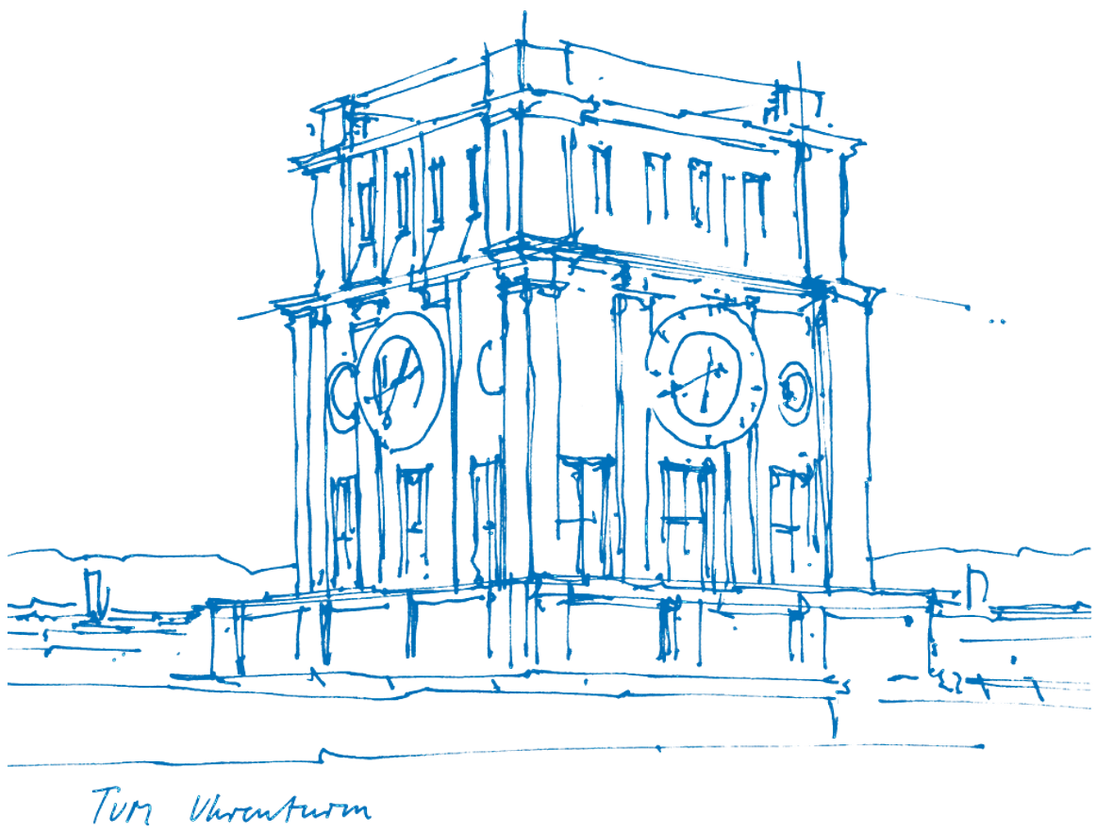
Practical Course: Vision Based Navigation

Lecture 3: Keypoint Detection, Matching and Motion Estimation

Jason Chui, Simon Klenk
Prof. Dr. Daniel Cremers

Version: 07.11.2022



Topics Covered

- Keypoint detection
  - Corner detection
  - Rotation estimation
- Keypoint description
  - Scale-Invariant Feature Transform (SIFT)
  - Binary Features: BRIEF, ORB
- Keypoint matching
- Robust model fitting with RANSAC
- Epipolar constraint
- Keypoint-based motion estimation
- Place recognition with bag of words

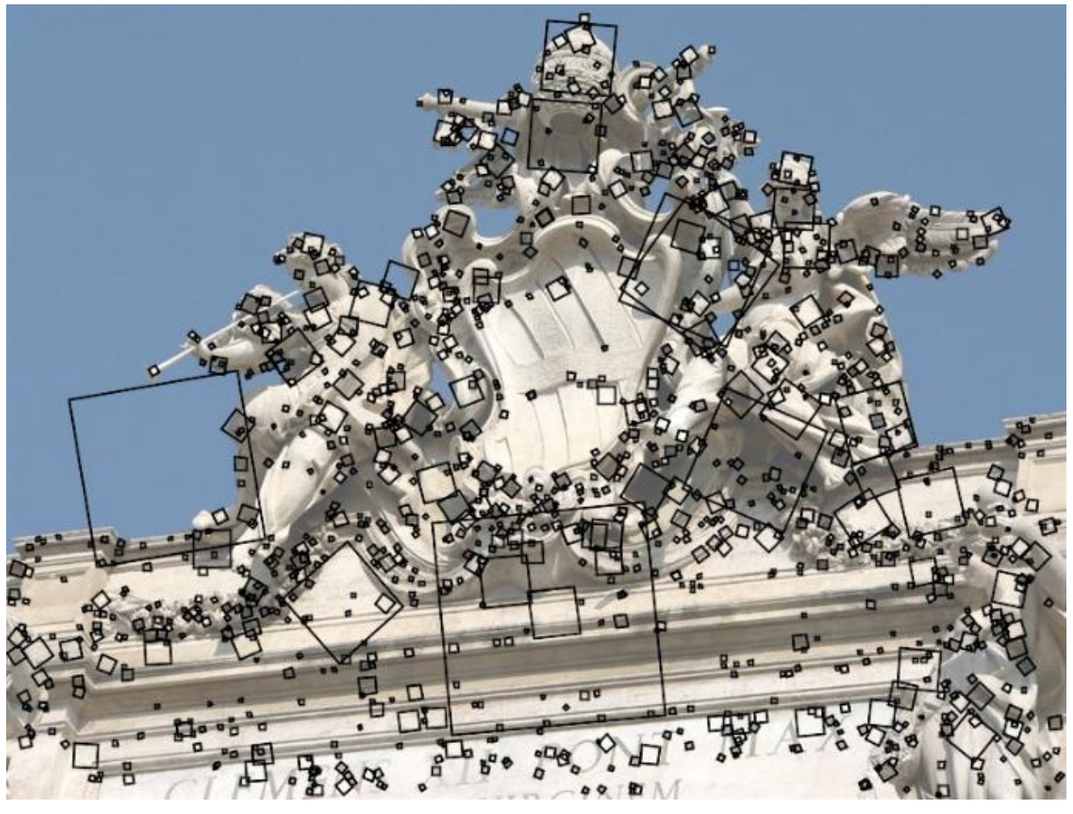
Local Features

Keypoint Detection

Desirable properties of keypoint detectors for visual SLAM / SfM:

- High repeatability
- Localization accuracy
- Robustness
- Invariance
- Computational efficiency

Harris Corners



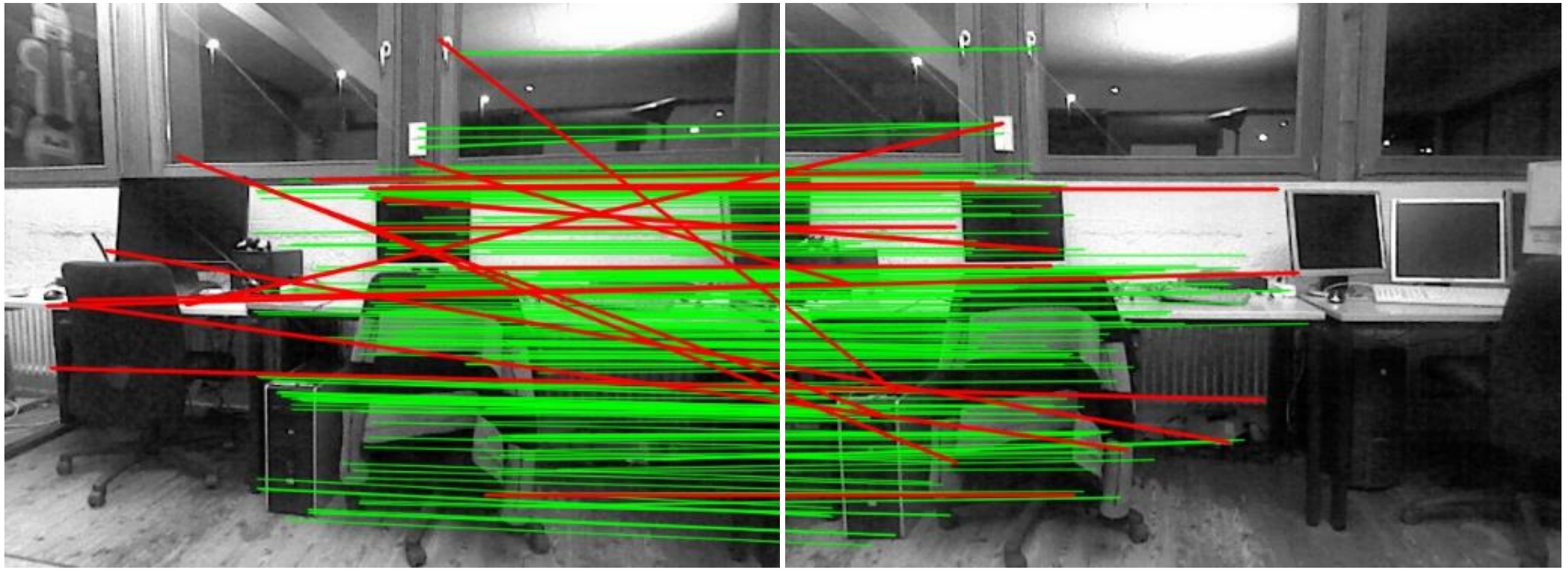
DoG (SIFT) blobs

Image source: Svetlana Lazebnik

Keypoint Matching

Desirable properties of keypoint matching for visual SLAM / SfM:

- High recall
- Precision
- Robustness
- Computational efficiency
- One possible approach to keypoint matching: by descriptor



Advantages of Local Features

Locality

- features are local, so robust to occlusion and clutter

Distinctiveness

- can differentiate a large database of objects

Quantity

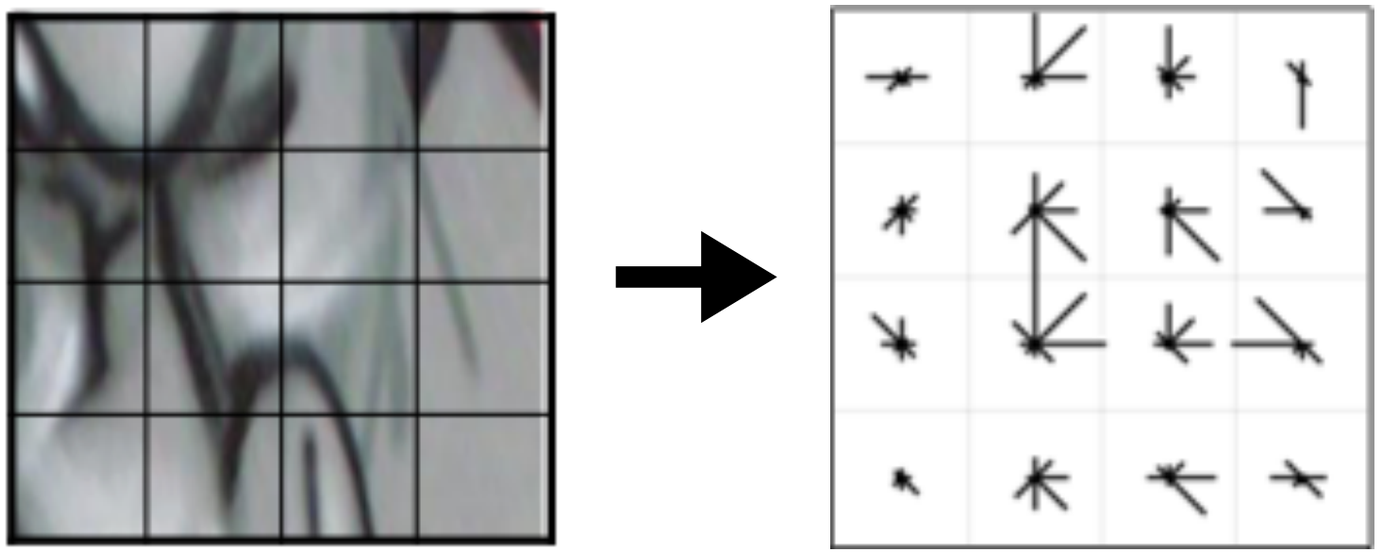
- hundreds or thousands in a single image

Efficiency

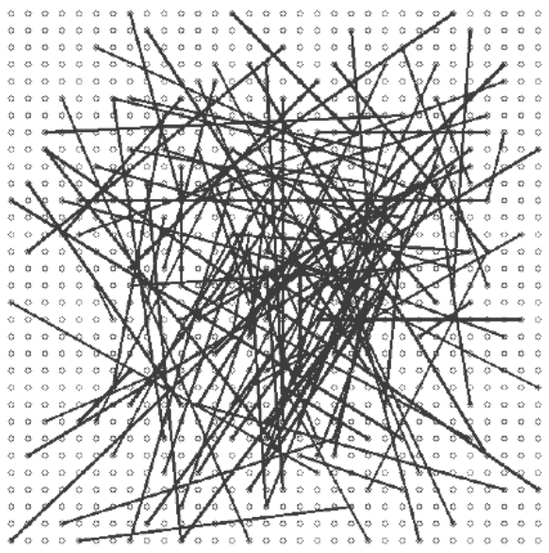
- real-time performance achievable

Local Feature Descriptors

- Desirable properties for SLAM / SfM: distinctiveness, robustness, invariance
- Extract signatures that describe local image regions, examples:
  - Histograms over image gradients (SIFT)
  - Histograms over Haar-wavelet responses (SURF)
  - Binary patterns (BRIEF, BRISK, FREAK, etc.)
  - Learning-based descriptors (f.e. Calonder et al., ECCV 2008)
- Rotation-invariance: Align with dominant orientation in local region
- Scale-invariance: Adapt described region extent to keypoint scale



SIFT gradient pooling



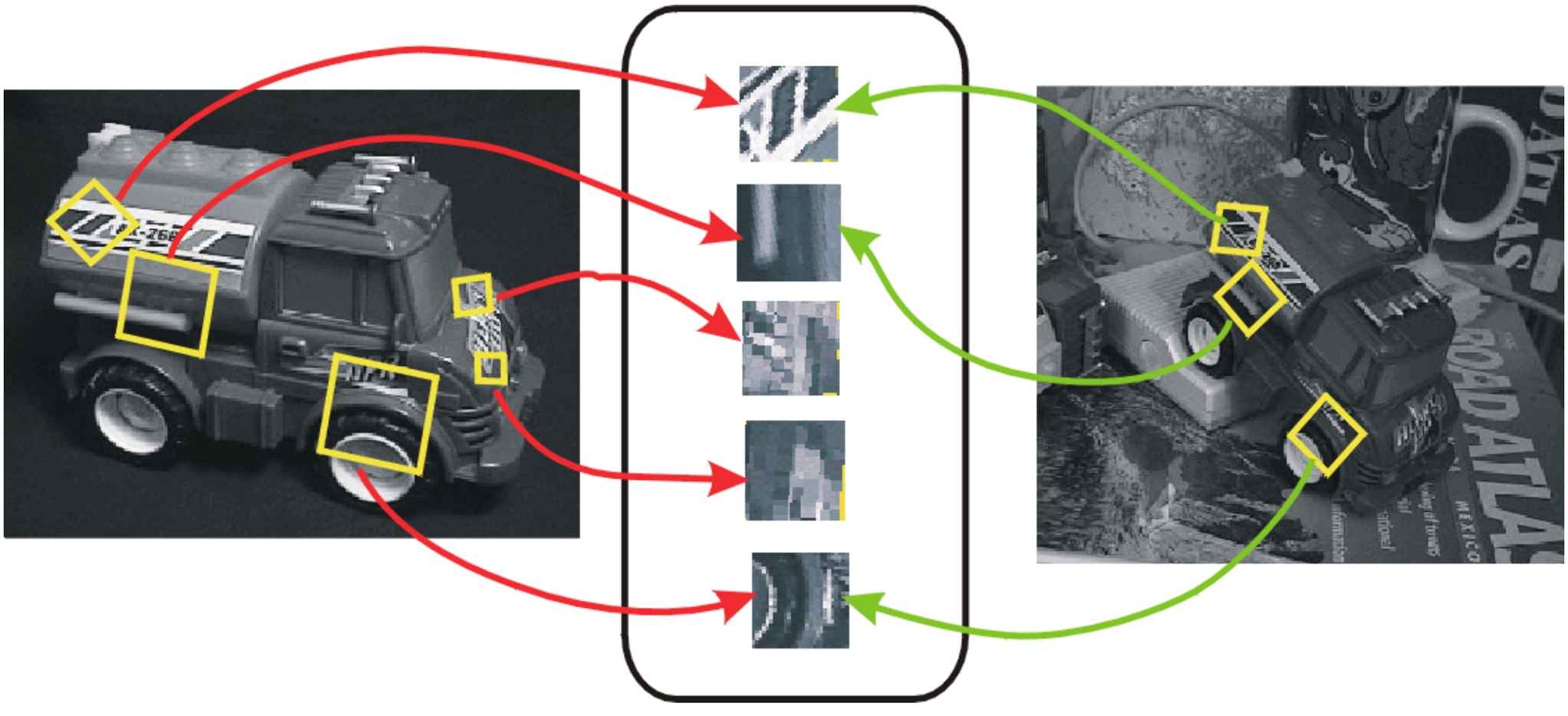
BRIEF binary tests

Image source: Svetlana Lazebnik / Calonder et al., ECCV 2010

Invariant Local Features

Find features that are invariant to transformations

- geometric invariance: translation, rotation, scale
- photometric invariance: brightness, exposure, ...



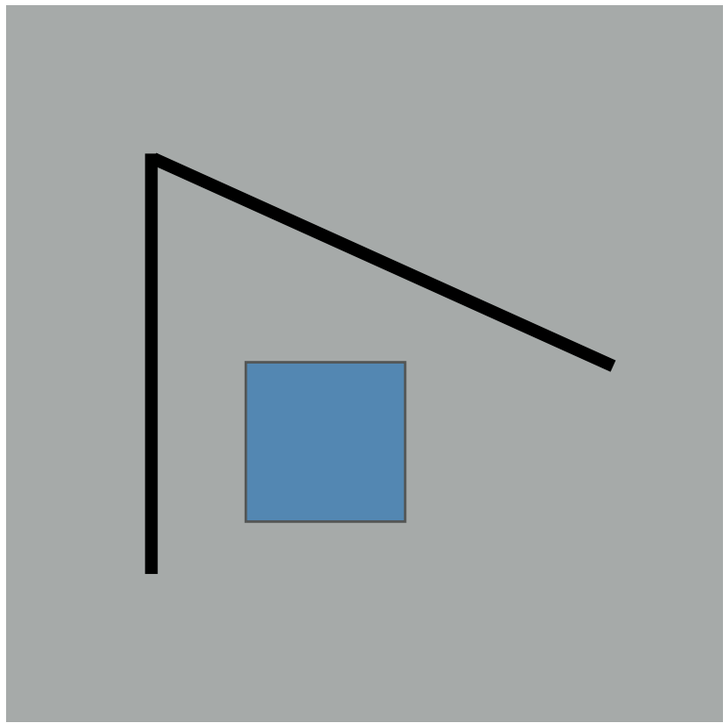
Feature Descriptors

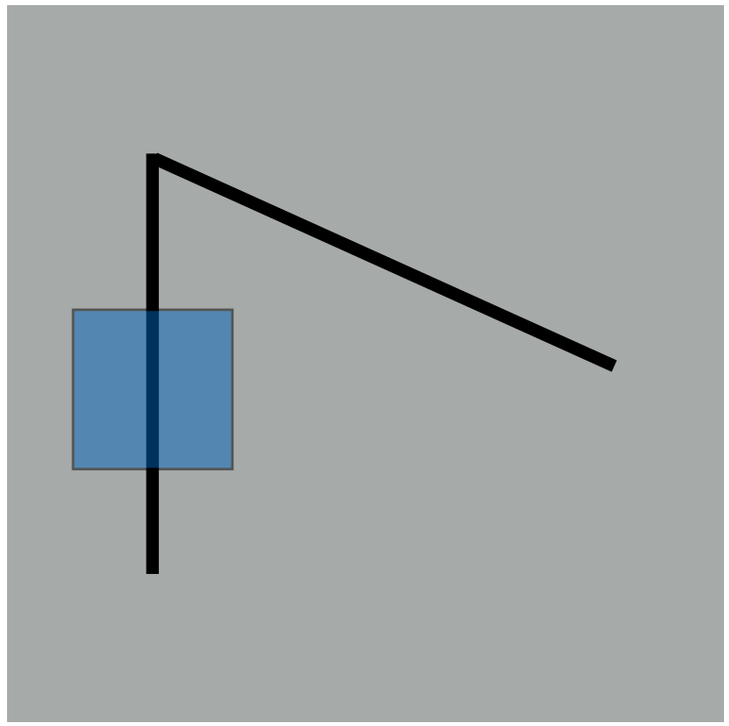
Keypoint Detection

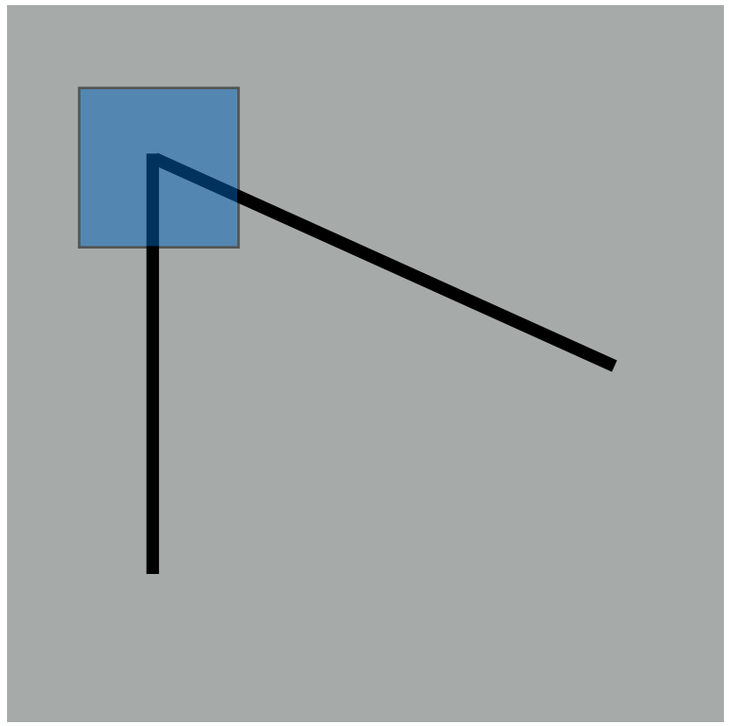
Local Measure of Uniqueness

Suppose we only consider a small window of pixels

- What defines whether a feature is well localized and unique?

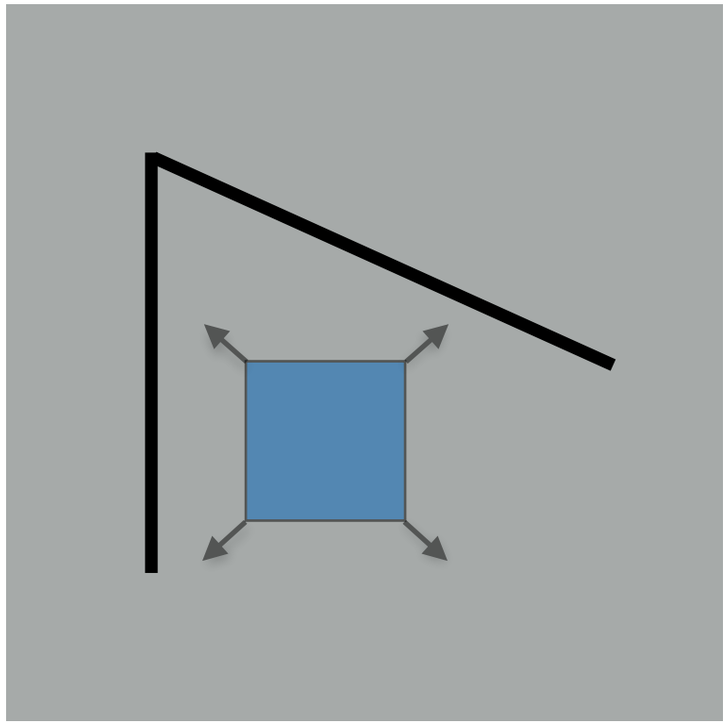


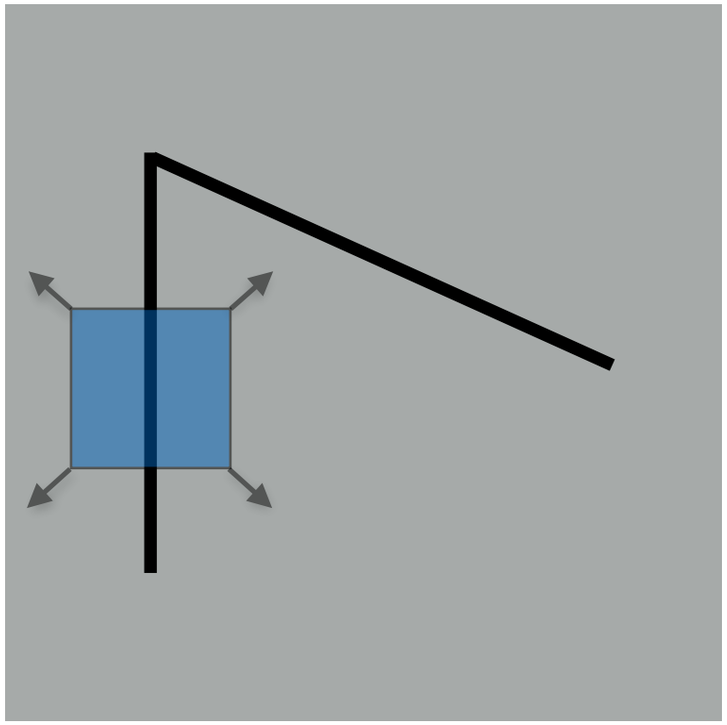


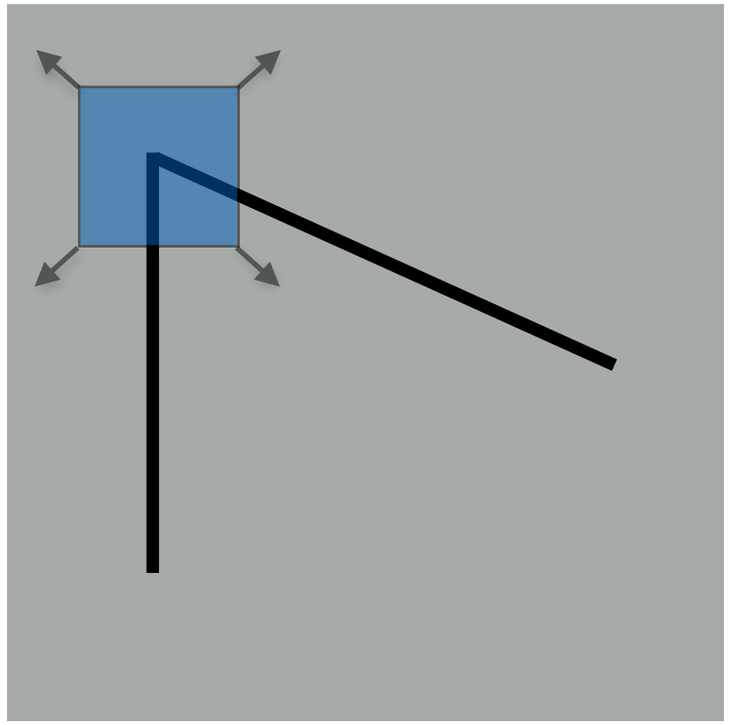


Local Measure of Uniqueness

How does the window change when you shift by a small amount?







"flat" region:
no change in any directions

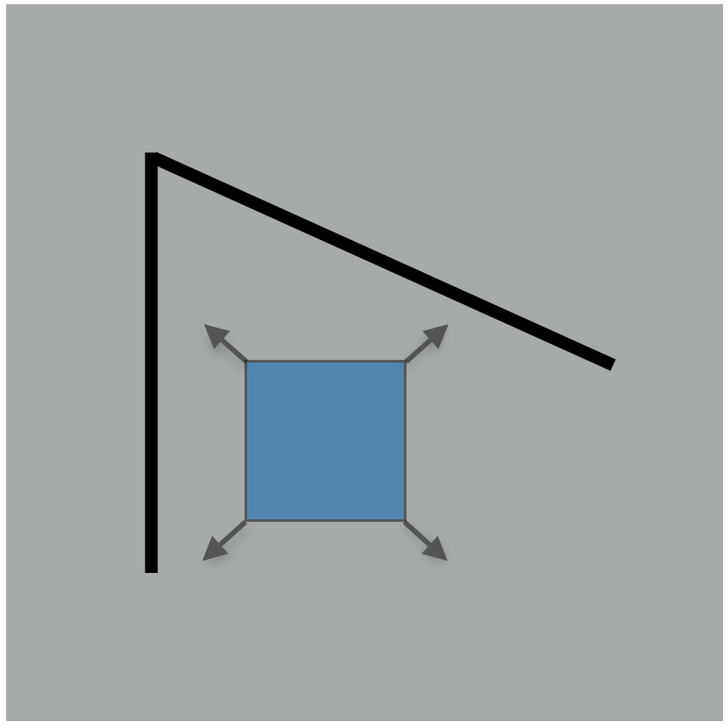
"edge":
no change along the edge direction

"corner":
significant change in all directions

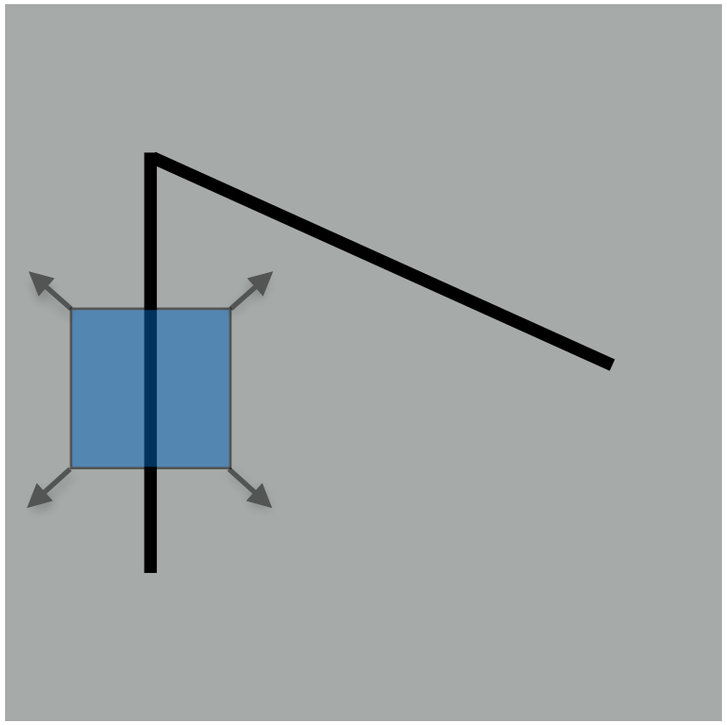
Local Measure of Uniqueness

Define:

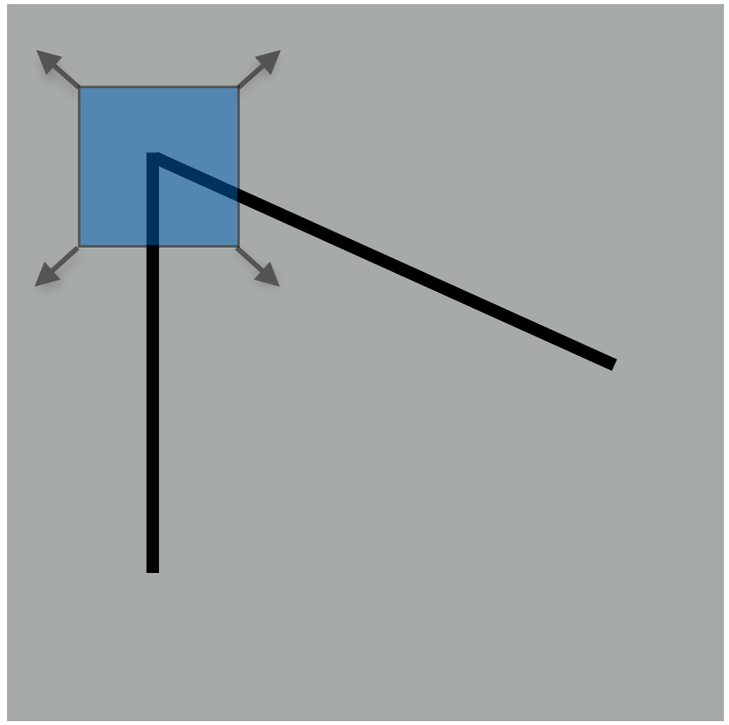
$E(u, v)$ = amount of change when you shift the window by (u, v)



$E(u, v)$ is small
for all shifts



$E(u, v)$ is small
for some shifts



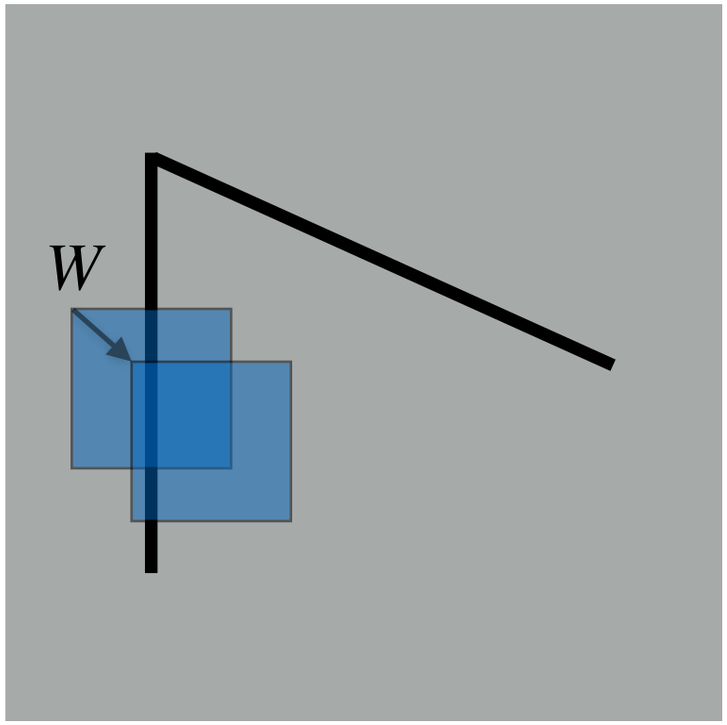
$E(u, v)$ is small
for no shifts

What do we want $\min_{u,v} E(u, v)$ to be?

Corner Detection

Consider shifting the window W by (u, v)

- how do the pixels in W change?
- compare each pixel before and after by Sum of the Squared Differences (SSD)
- this defines an SSD “error”:



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

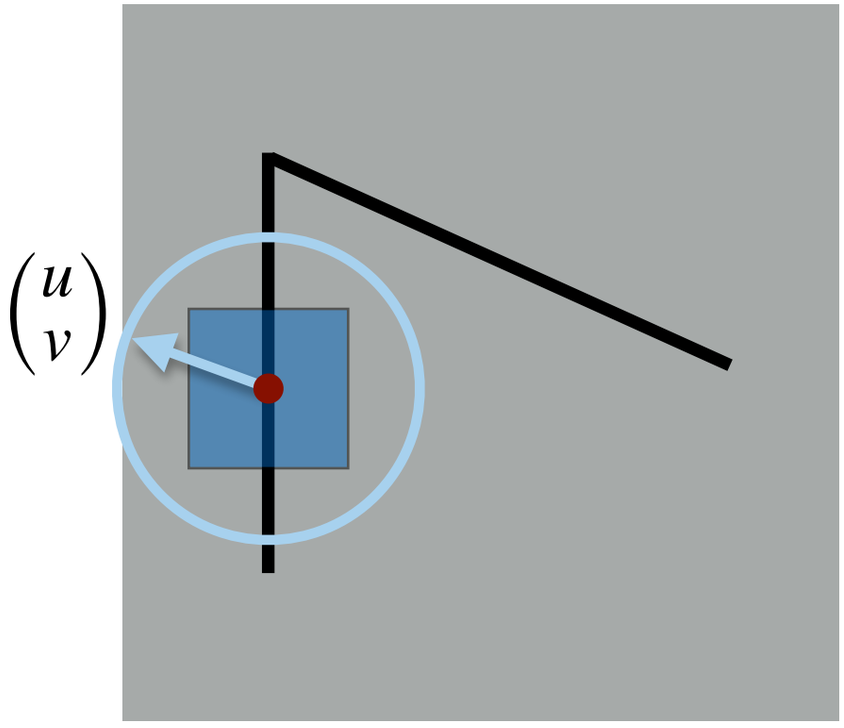
Corner Detection: Structure Tensor

This can be rewritten:

$$E(u,v) \approx \sum_{(x,y)\in W} \left[(I_x \; I_y) \begin{pmatrix} u \\ v \end{pmatrix} \right]^2$$

$$= \sum_{(x,y)\in W} (u \; v) \underbrace{\begin{pmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{pmatrix}} \begin{pmatrix} u \\ v \end{pmatrix}$$

“structure tensor” H



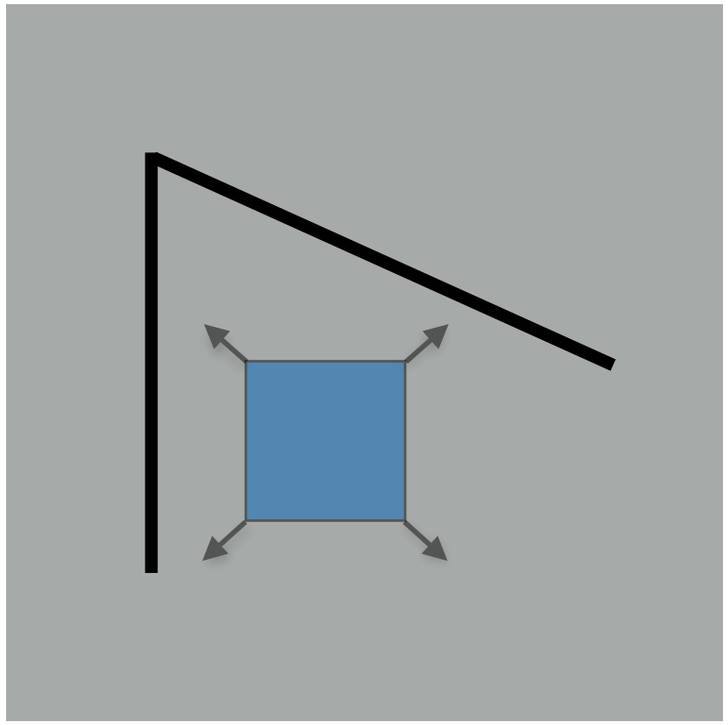
For the example above:

- You can move the center of the window to anywhere on the blue unit circle
- Which directions will result in the largest and smallest E values?
- We can find these directions by looking at the eigenvectors of H

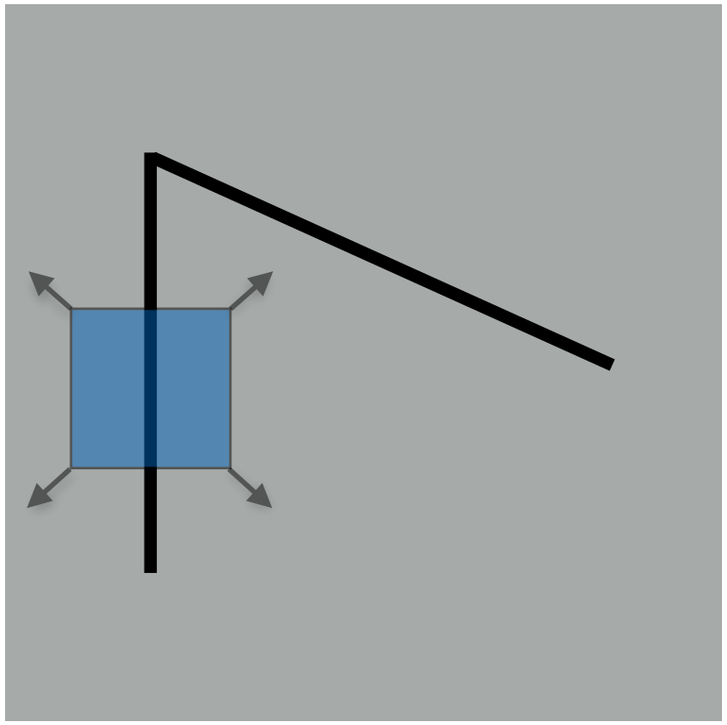
Corner Detection

Define:

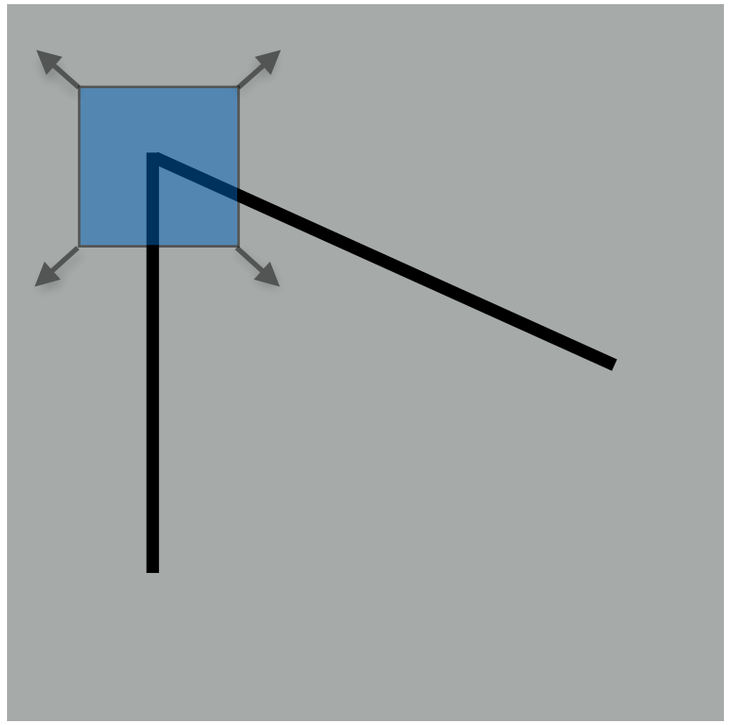
$E(u, v)$ = amount of change when you shift the window by (u, v)



$E(u, v)$ is small
for all shifts



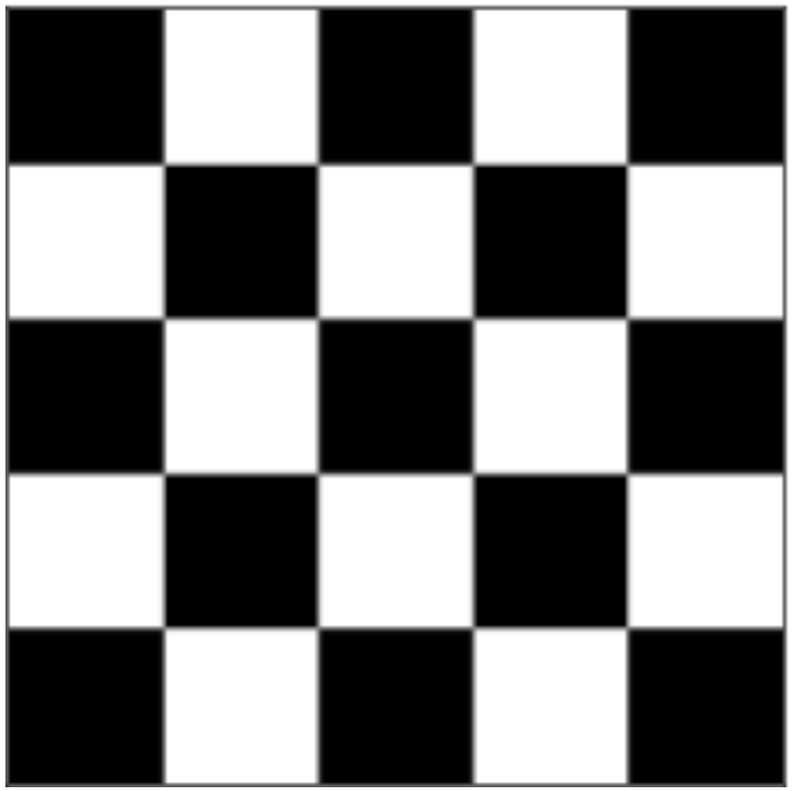
$E(u, v)$ is small
for some shifts



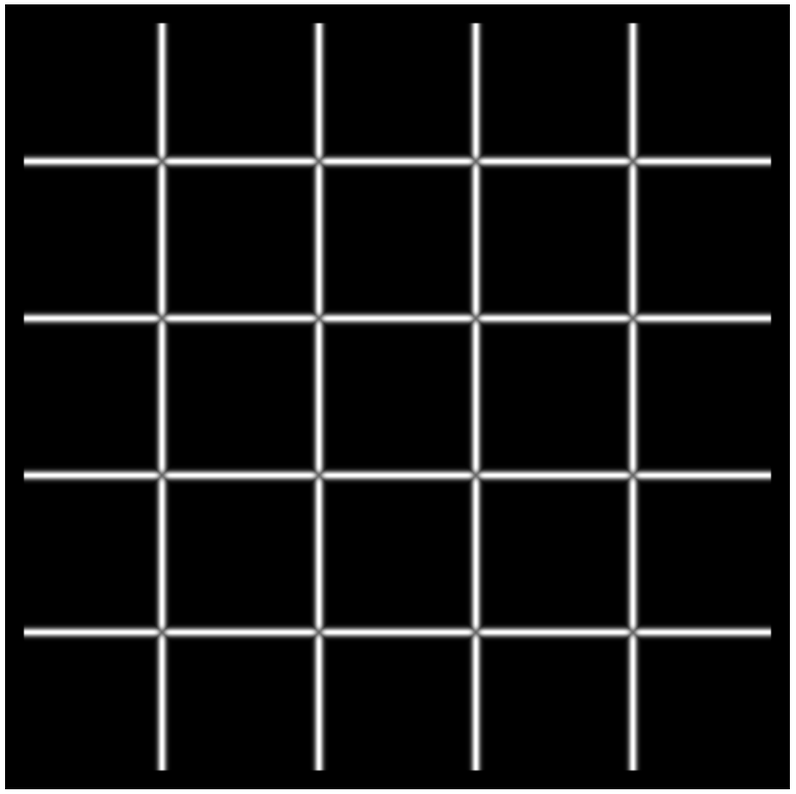
$E(u, v)$ is small
for no shifts

Corner Detection Recipe

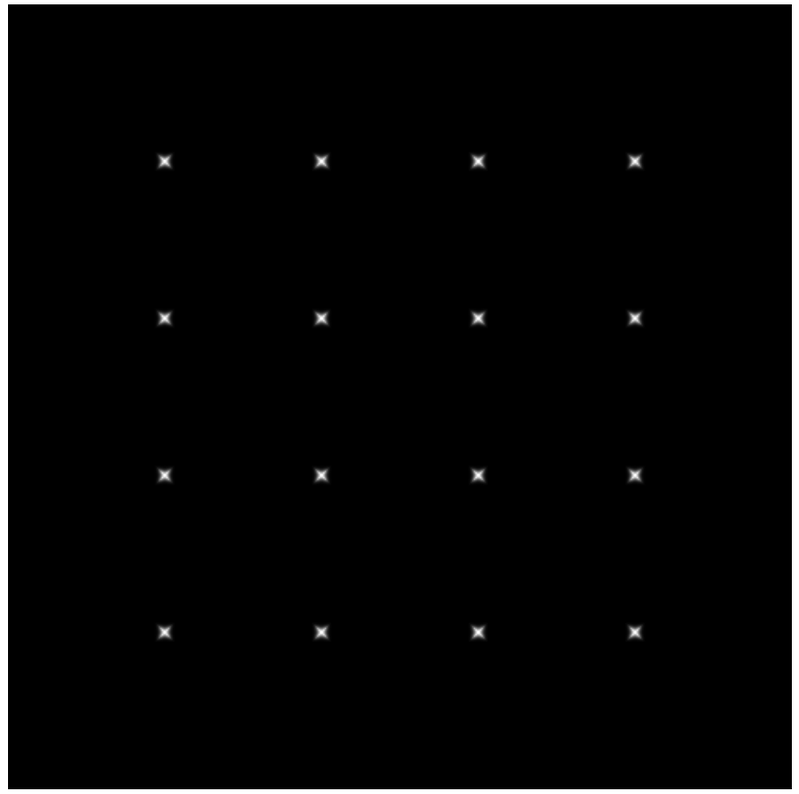
- Compute the gradient at each point in the image.
- Create the H matrix from the entries in the gradient.
- Compute the eigenvalues.
- Find points with large response (λ_- > threshold).
- Choose those points where λ_- is a local maximum as features.



I



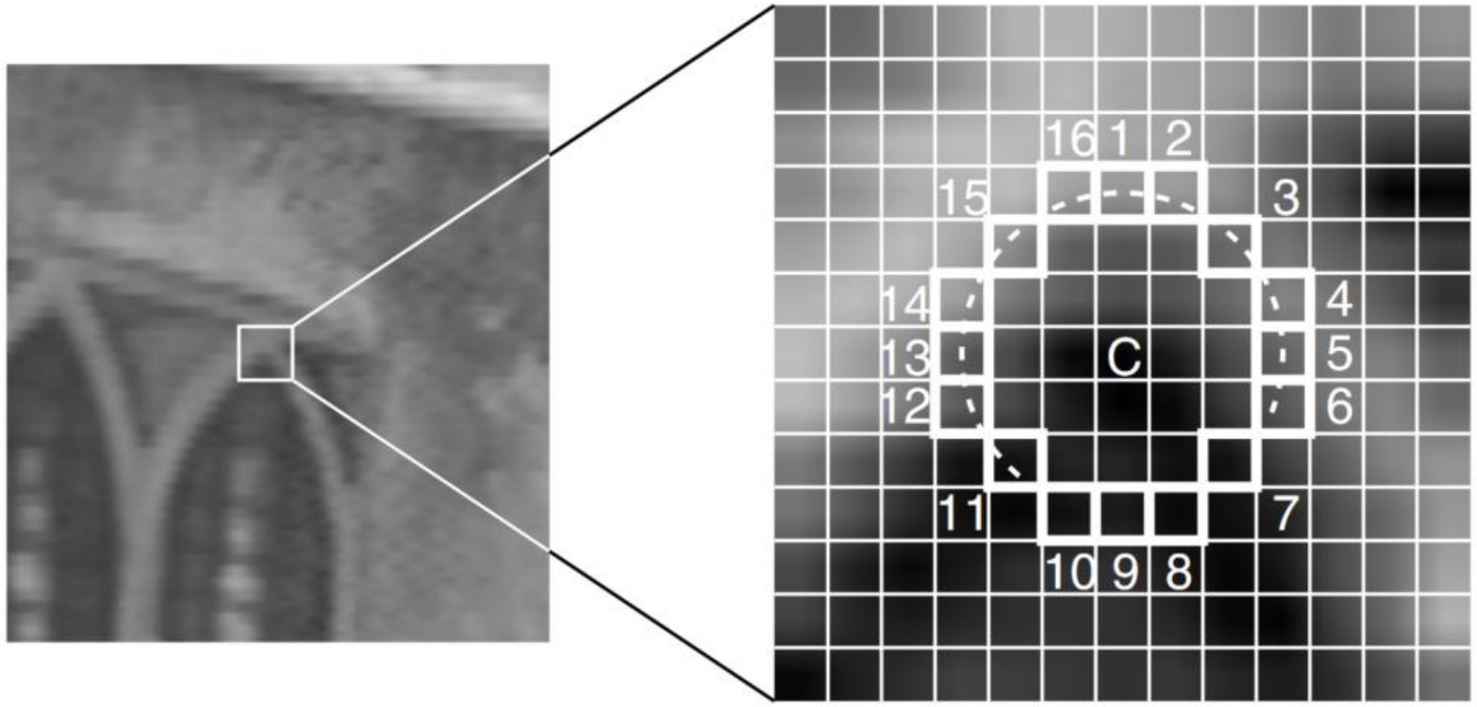
λ_+



λ_-

FAST Detector

- Features from Accelerated Segment Test
- Check relation of brightness values to center pixel along circle
- Specific number of contiguous pixels brighter or darker than center
- Very fast corner detection

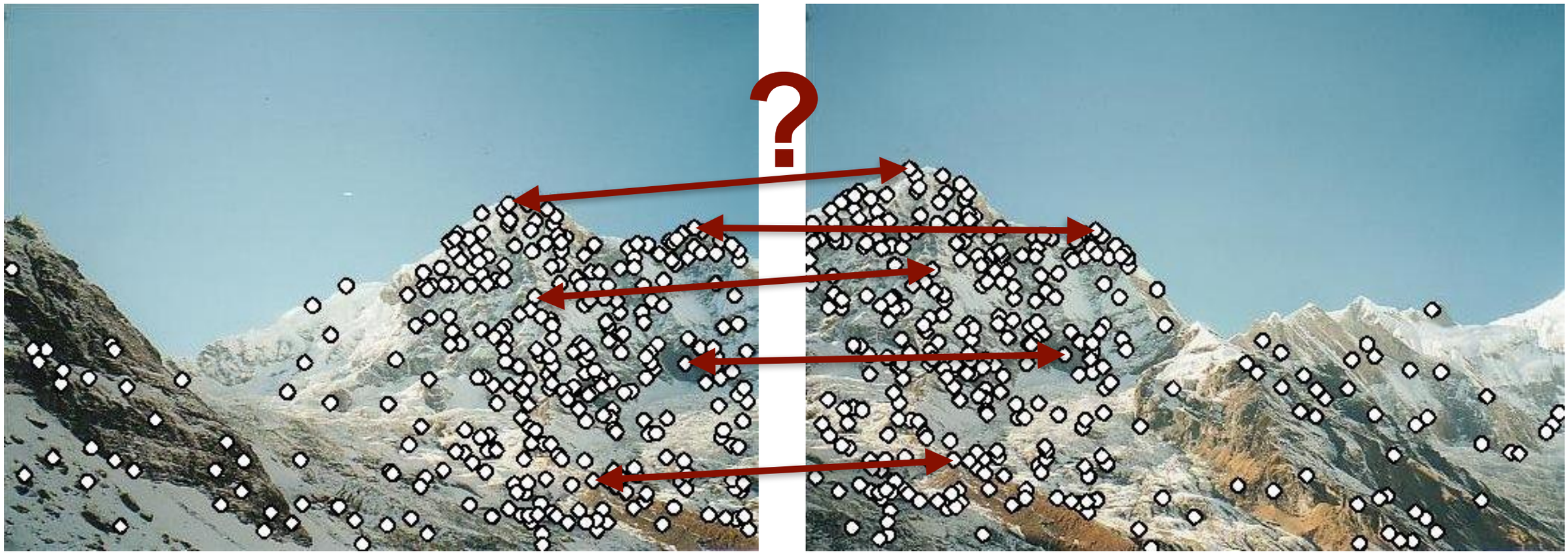


Rosten, Drummond, Fusing Points and Lines for High Performance Tracking, ICCV 2005

Keypoint Descriptors

Keypoint Descriptors

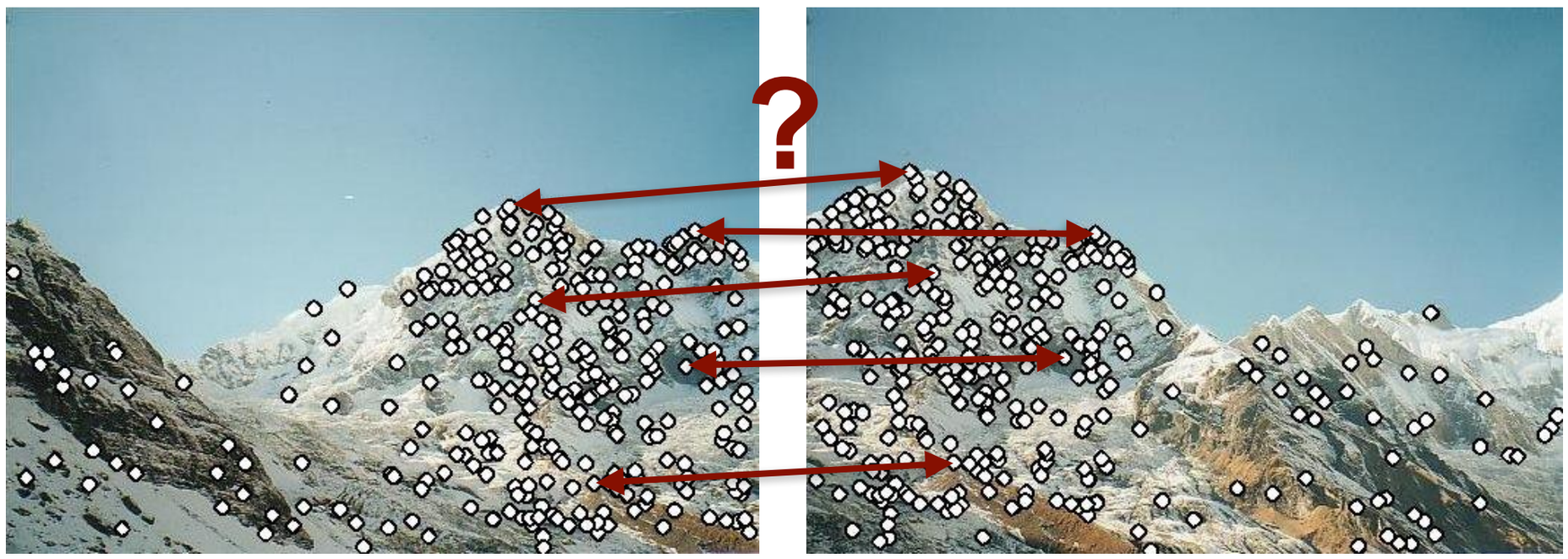
- We know how to detect good points.
- Next question: How to match them?



- Idea: extract distinctive descriptor vector from a local patch around the keypoint.

Invariance

- Goal: match keypoints regardless of image transformation.
  - This is called transformational invariance.
- Most keypoint detection and description methods 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



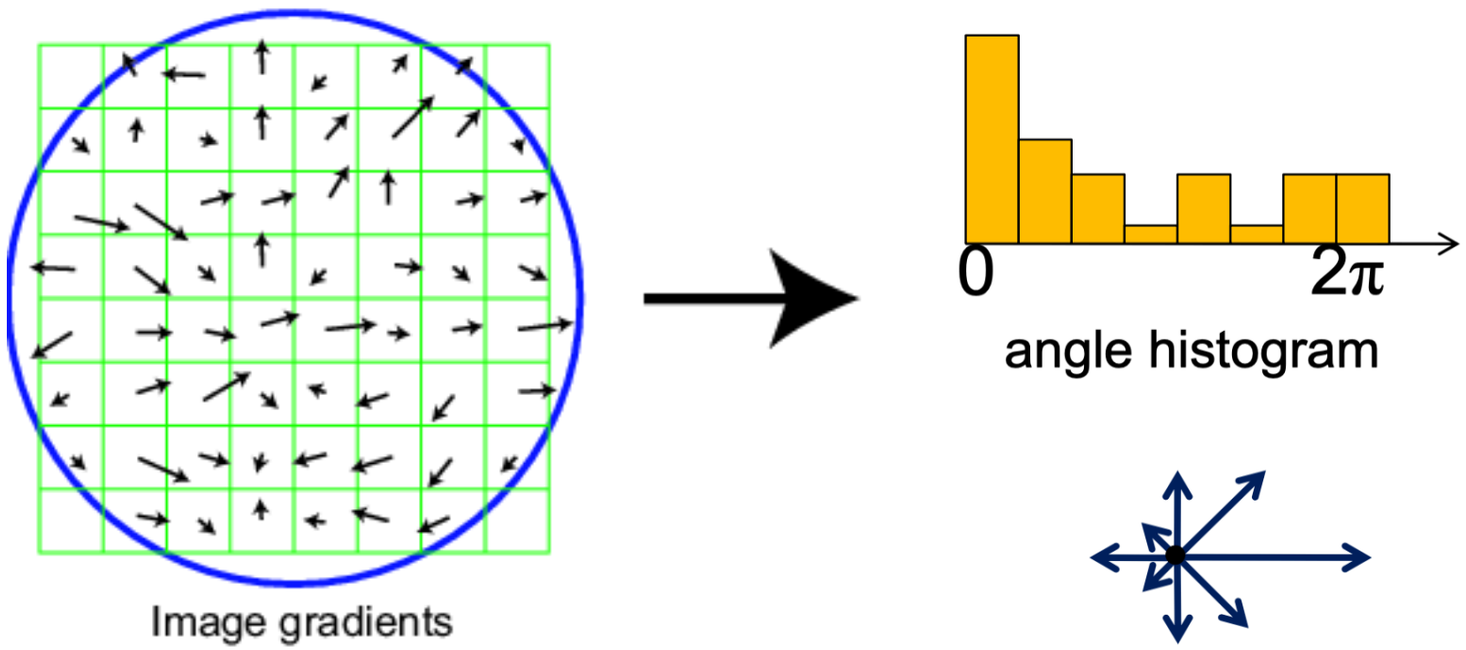
2D Rotation Invariance

- Idea: align the descriptor with a dominant 2D orientation
- Example approach: Use the eigenvector of H corresponding to larger eigenvalue

Figure by Matthew Brown

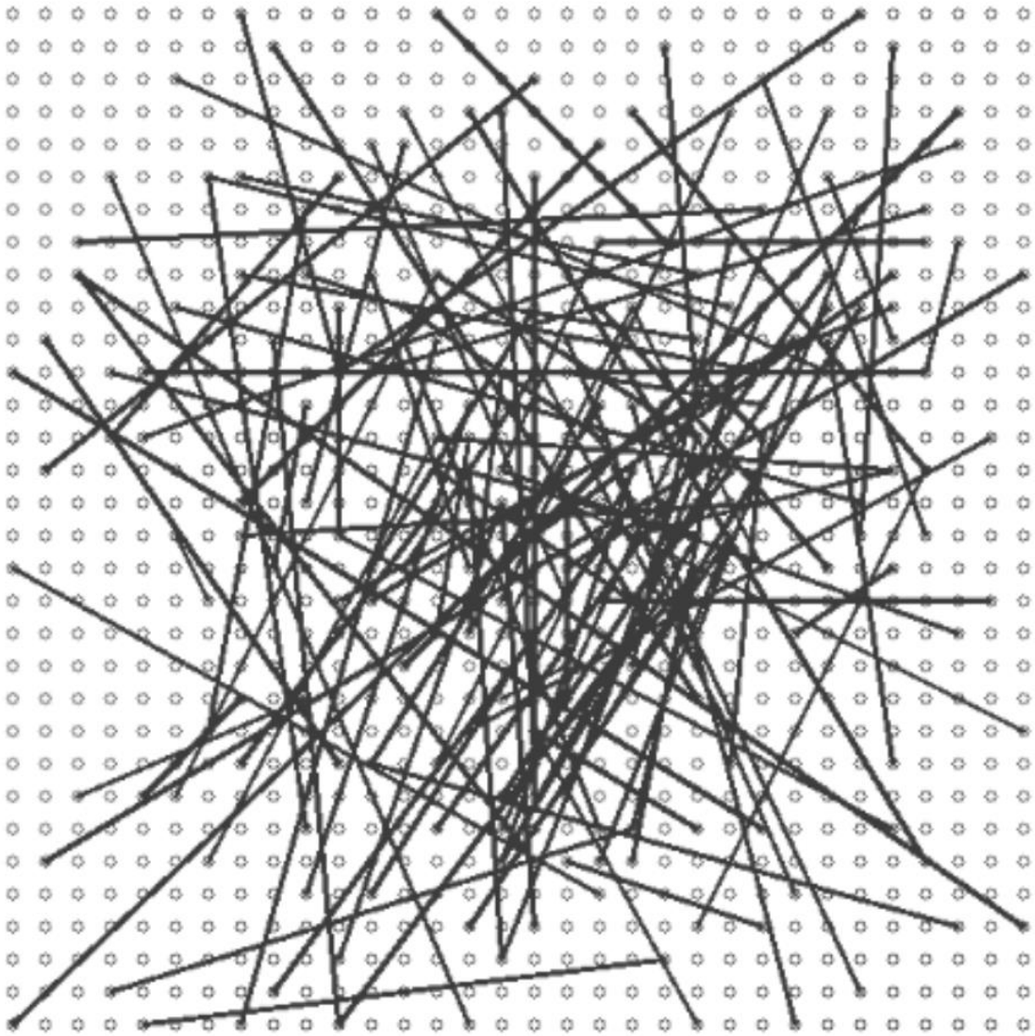
2D Rotation Invariance: SIFT

- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations
- Select strong local orientation maxima and create one or more descriptors



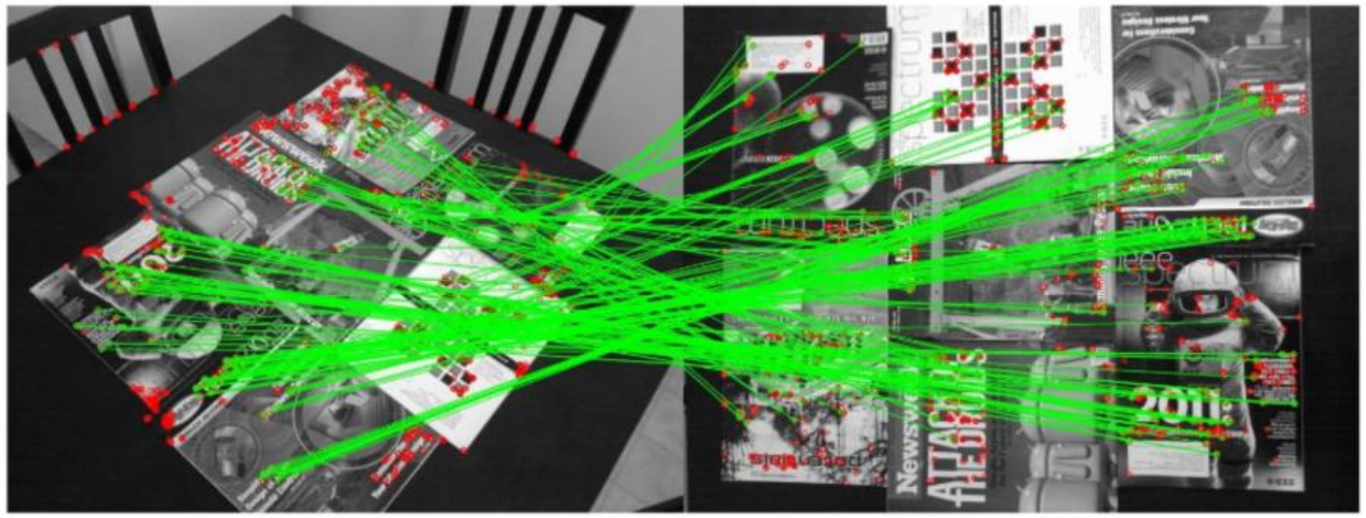
Binary Descriptors: BRIEF

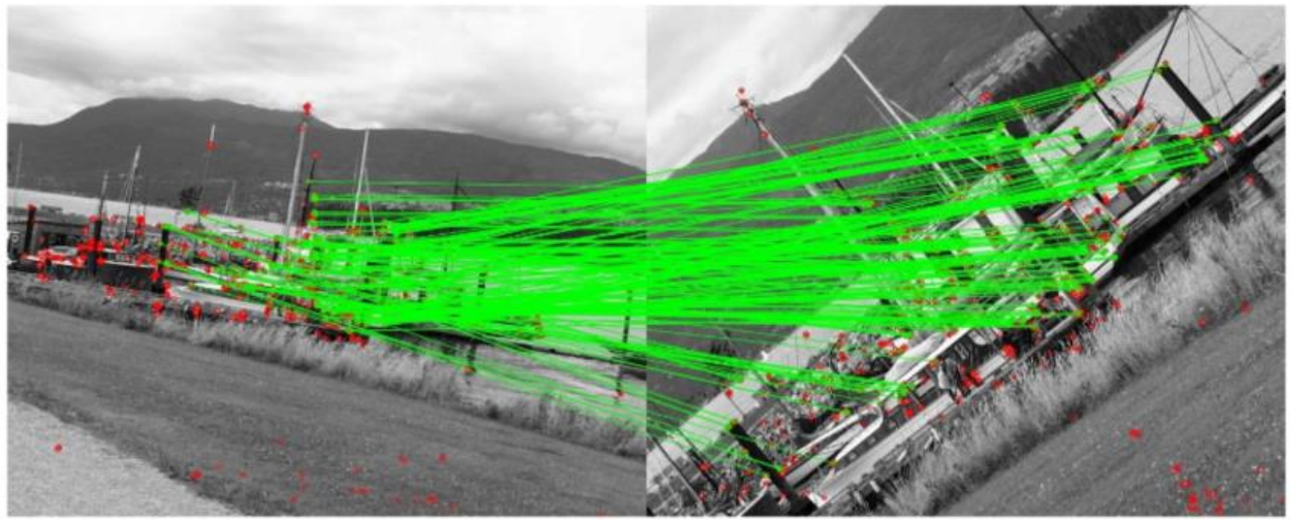
- “Binary Robust Independent Elementary Features” (Calonder et al. ECCV 2010)
- Binary descriptor from intensity comparisons at sample positions
- Very efficient to compute
- Fast matching distance through Hamming distance (xor and popcount CPU instructions)



Binary Descriptors: ORB

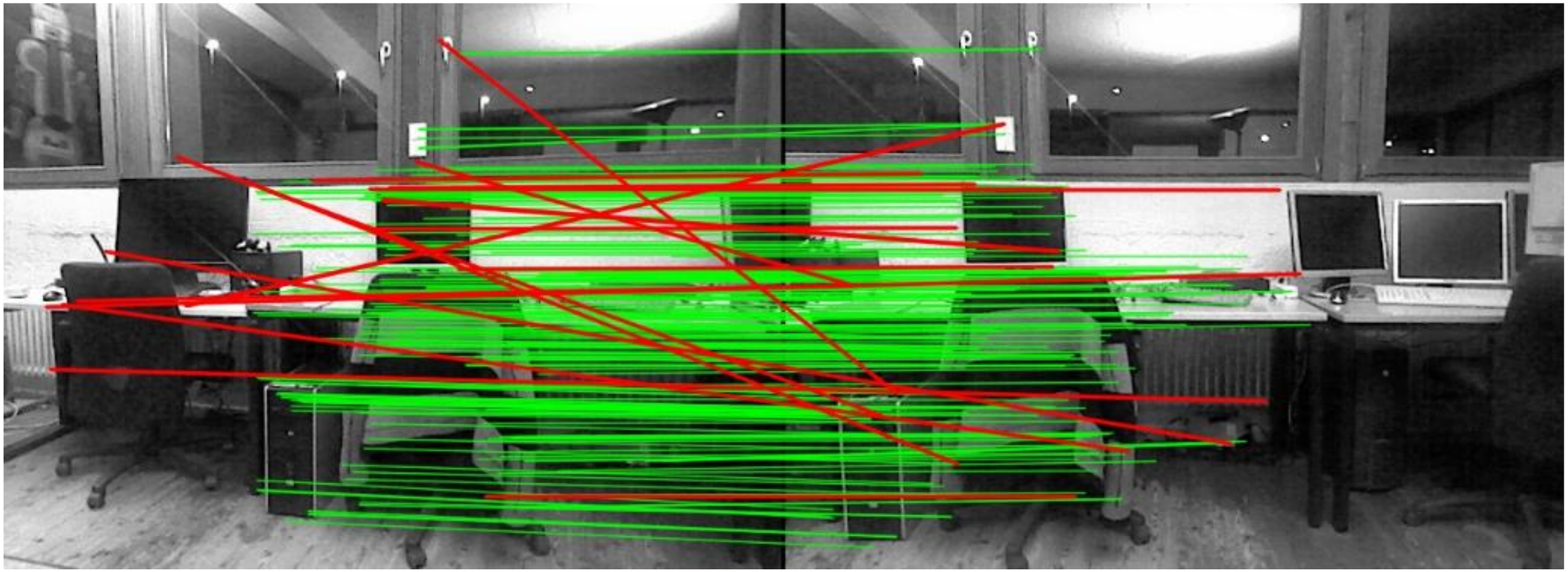
- “Oriented Fast and Rotated BRIEF” (Rublee et al. ICCV 2011)
- Combination of FAST detector and BRIEF descriptor
- Rotation-invariant BRIEF: Estimate dominant orientation from patch moments (intensity centroid)
- Improved binary pattern
- Very popular for SLAM / VO





Keypoint Matching

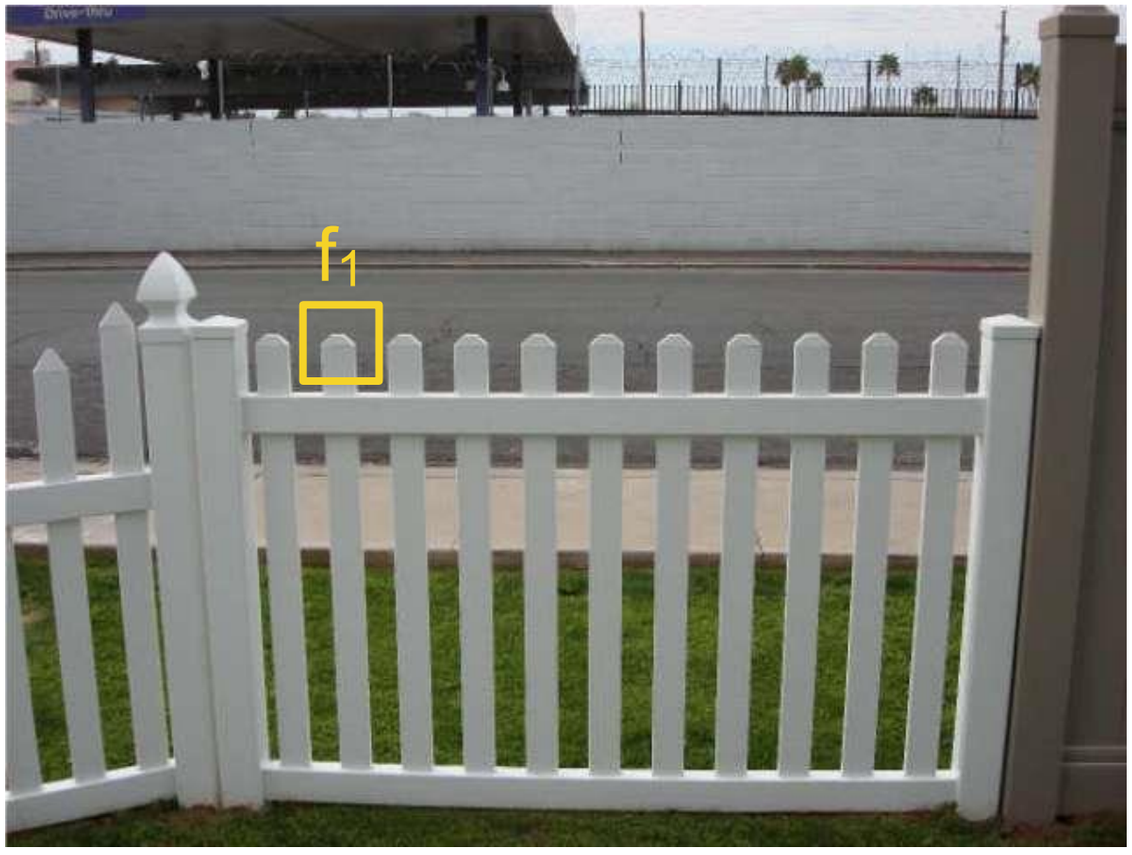
Keypoint Matching



- Main idea: Match keypoints with similar descriptors

Matching Distance

- How to define the difference between two descriptors f_1, f_2?
- Simple approach is to assign keypoints with minimal sum of square differences SSD(f_1, f_2) between entries of the two descriptors

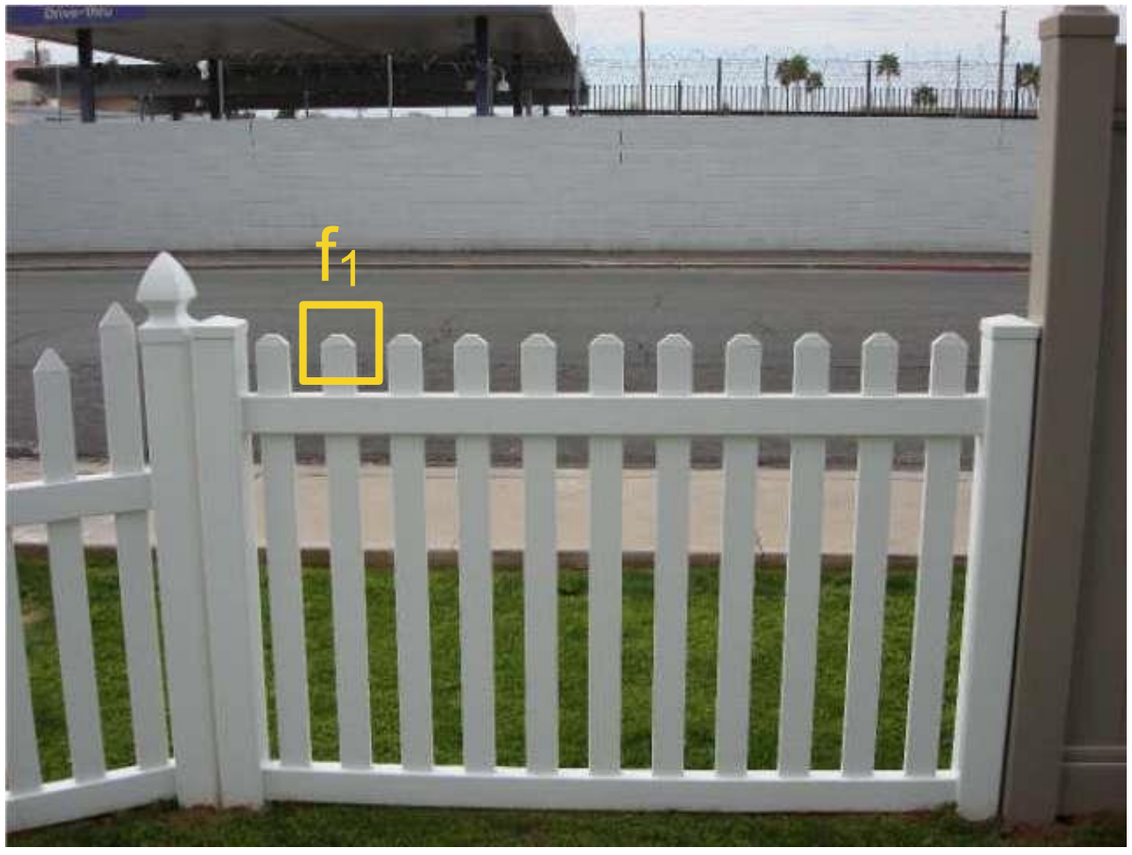


I_1

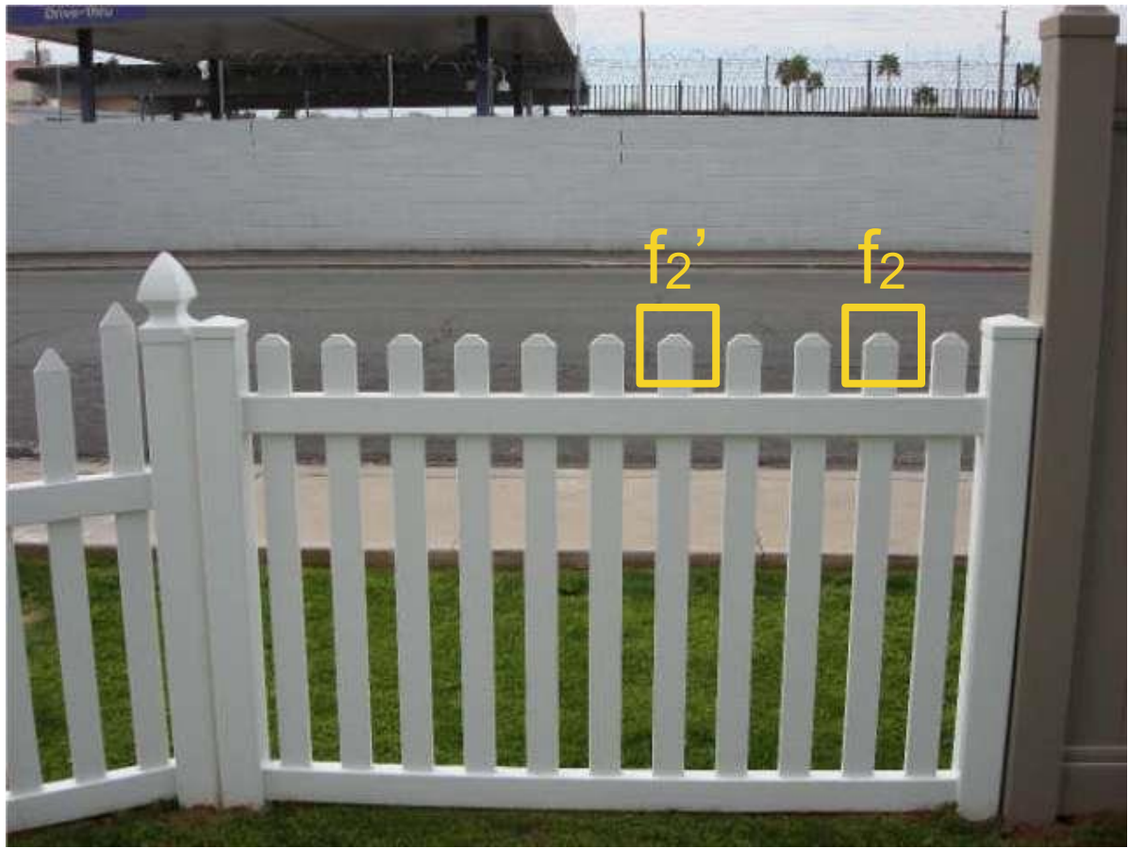
I_2

Matching Distance

- Better approach: best to second best ratio distance = SSD(f_1, f_2) / SSD(f_1, f_2’)
  - f_2 is best SSD match to f_1 in I_2
  - f_2’ is second best SSD match to f_1 in I_2
- Match should be “unique”

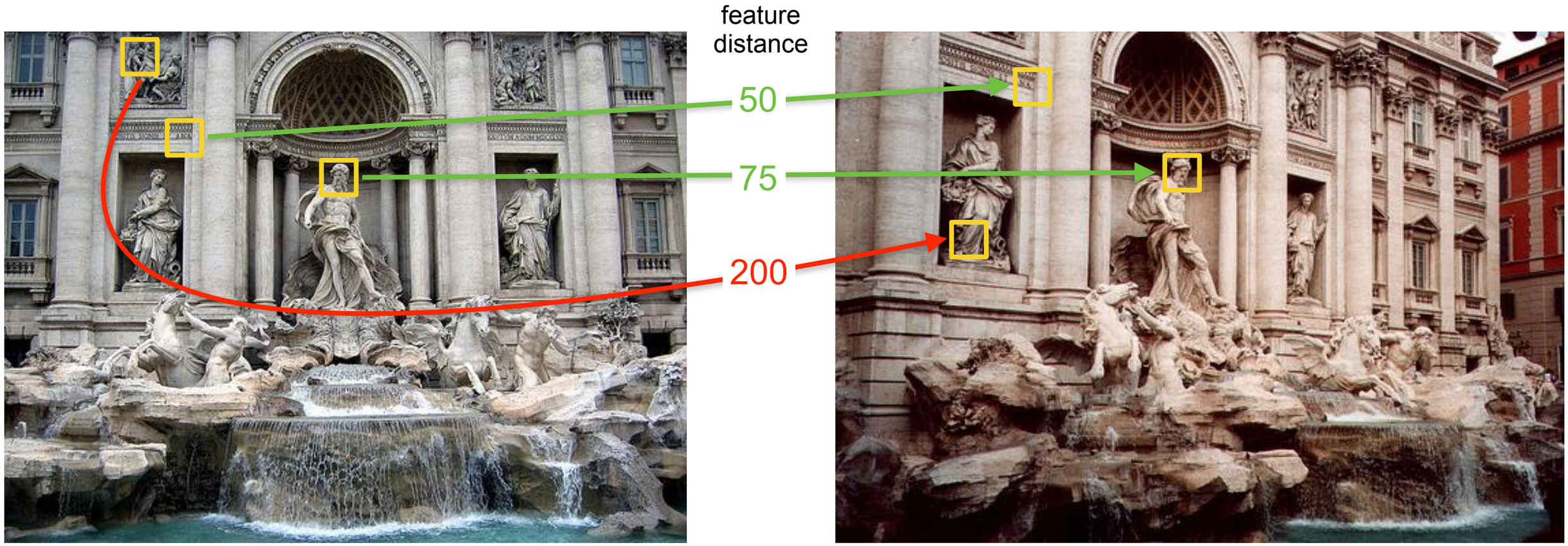


I_1



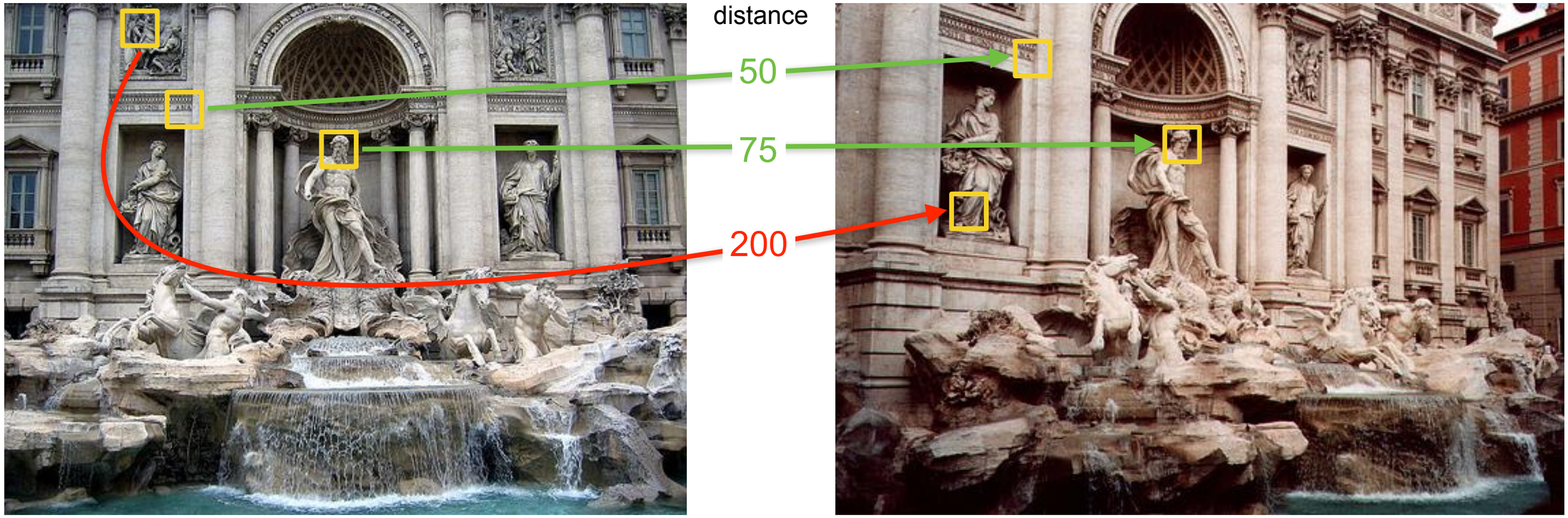
I_2

Eliminating Bad Matches



- Only accept matches with distance smaller a threshold.
- How to choose the threshold?

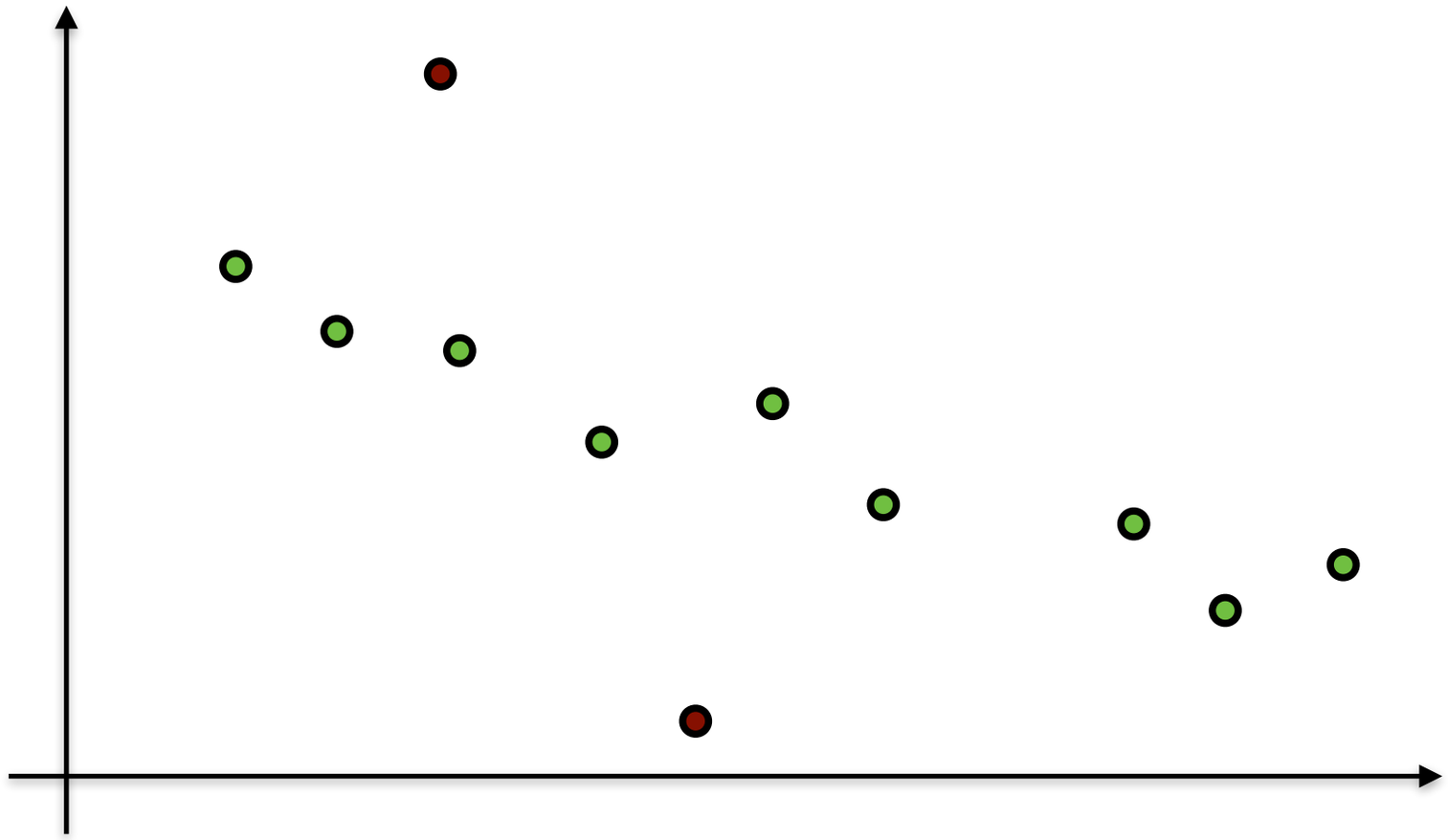
True/False Positives



- Choice of threshold affects performance
  - Too restrictive: less false positives, but also less true positives
  - Too lax: more true positives, but also more false positives
- Can we do more? Yes: Robust model fitting with RANSAC!

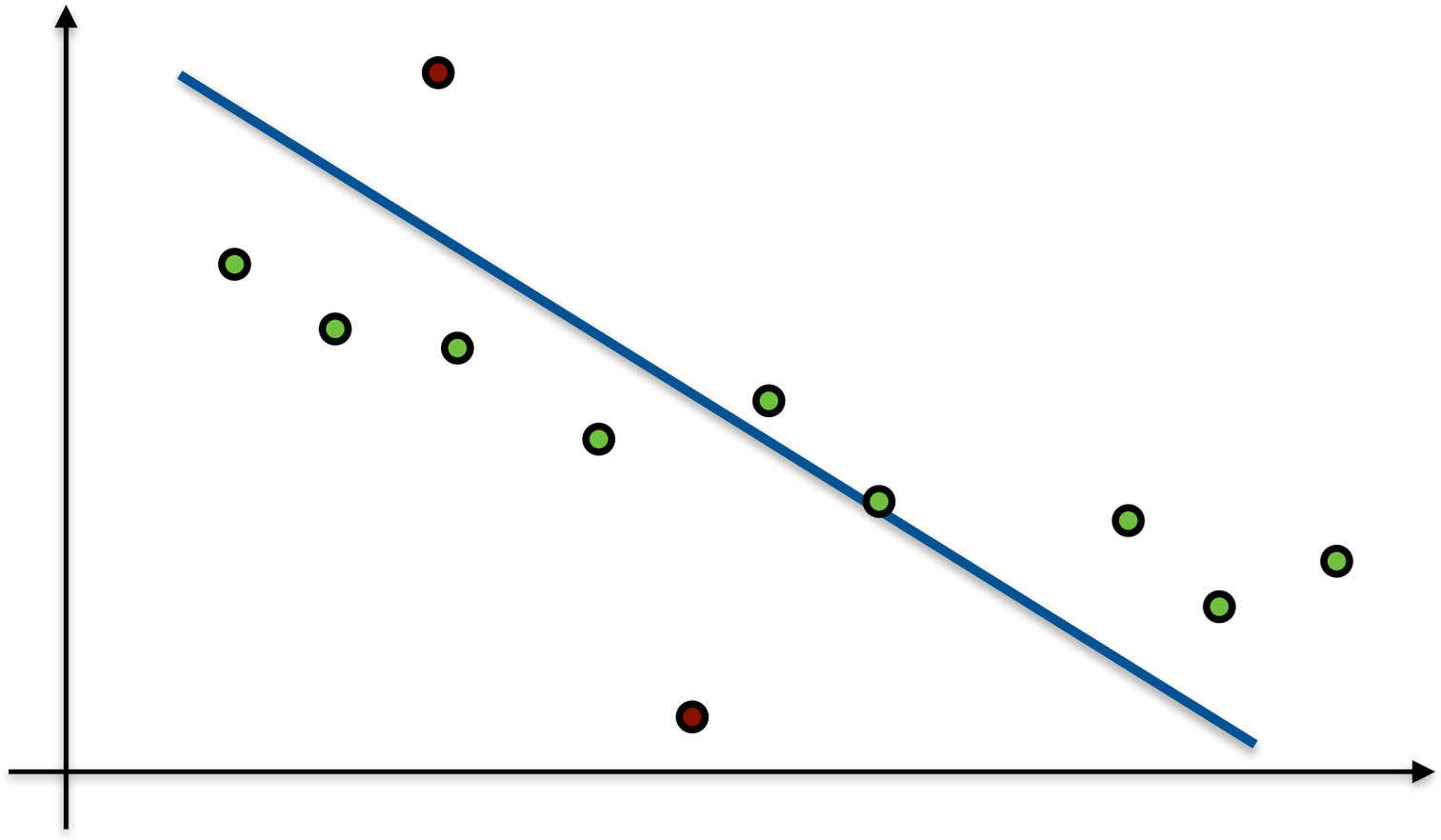
Random Sample Consensus (RANSAC)

- Model fitting in presence of noise and outliers
- Example: fitting a line through 2D points



Random Sample Consensus (RANSAC)

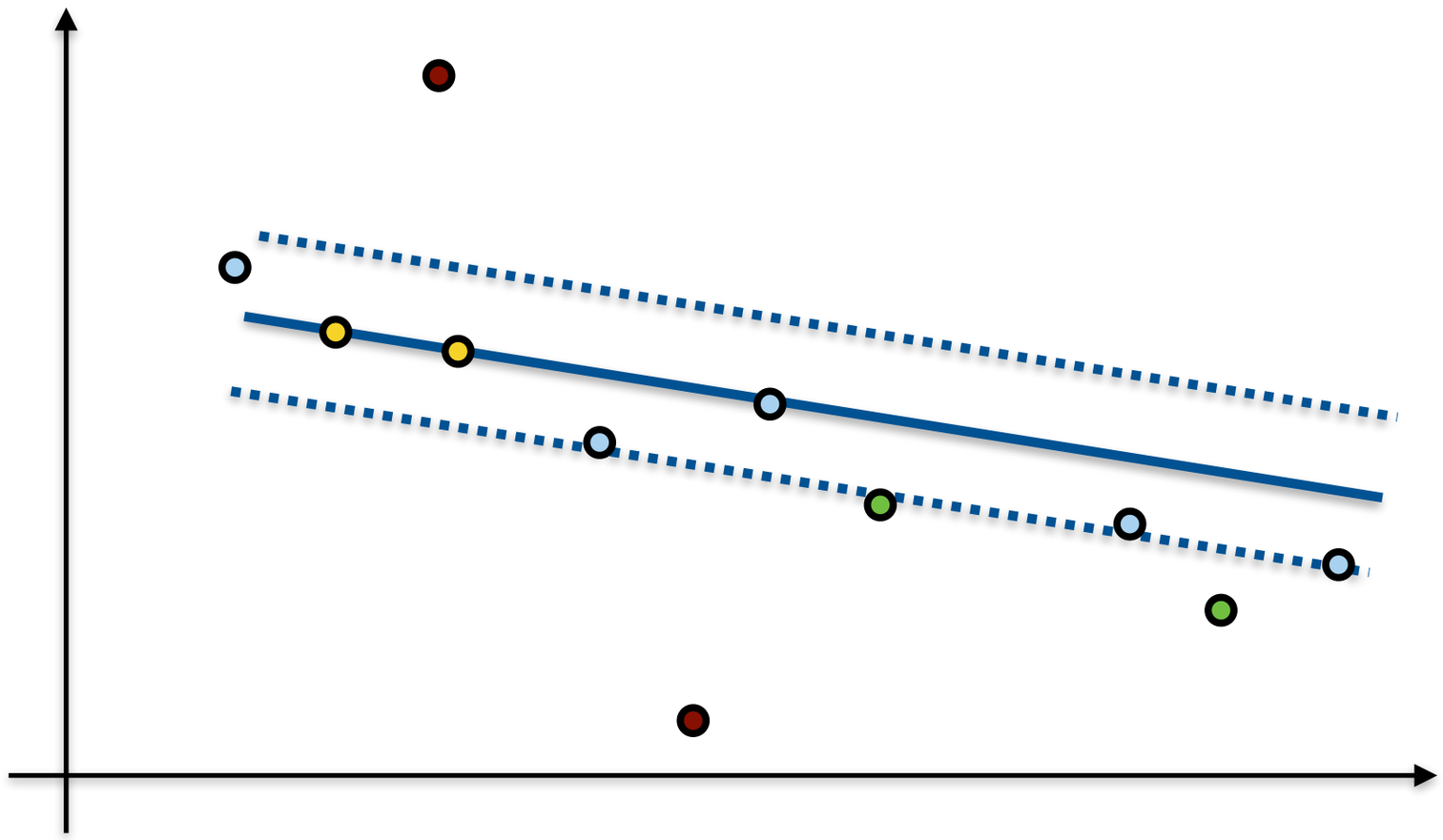
- Least-squares solution, assuming constant noise for all points



Bad!

Random Sample Consensus (RANSAC)

- We only need 2 points to fit a line. Let’s try 2 random points.

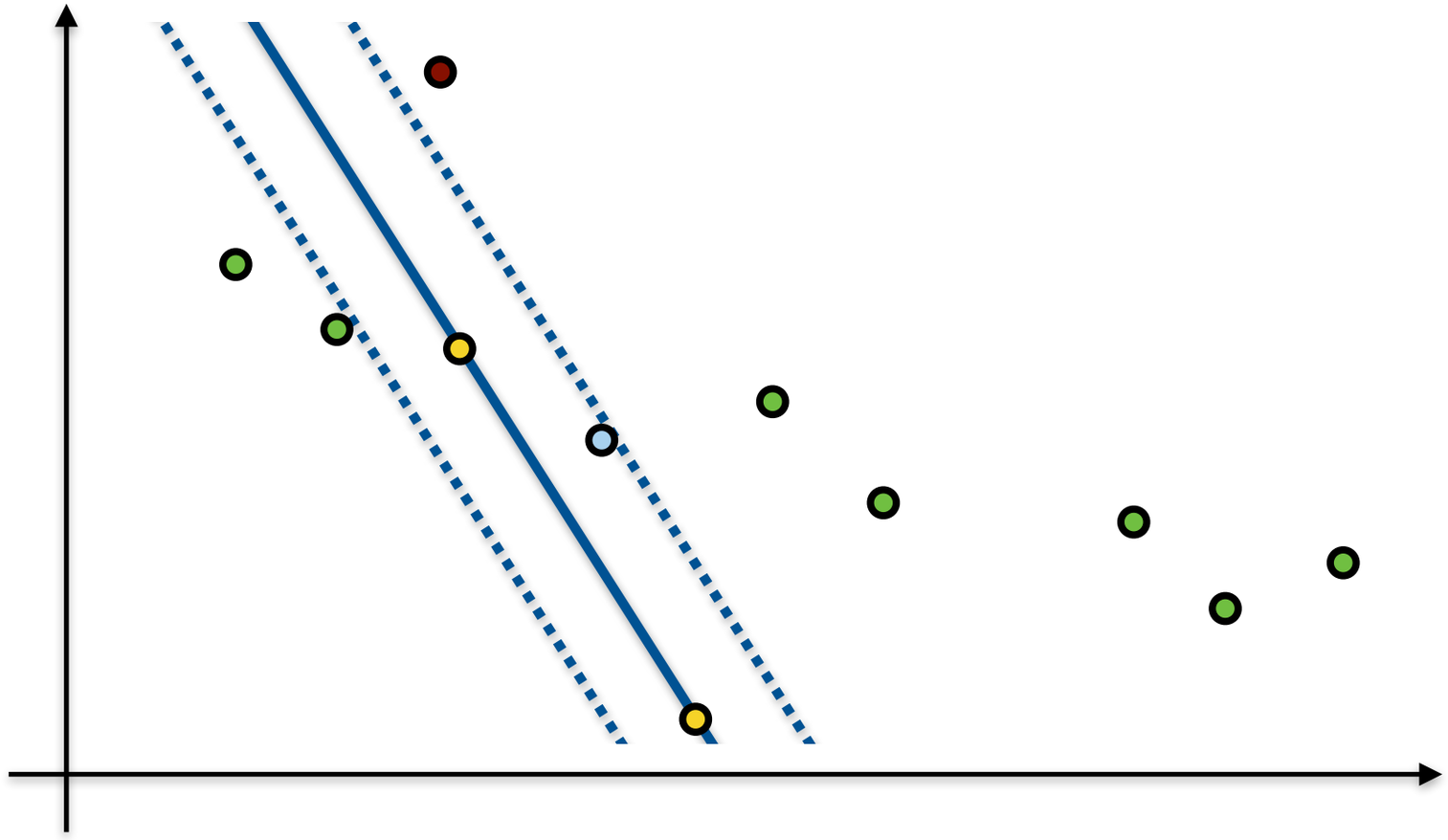


Quite ok…

7 inliers
4 outliers

Random Sample Consensus (RANSAC)

- Let’s try 2 other random points.

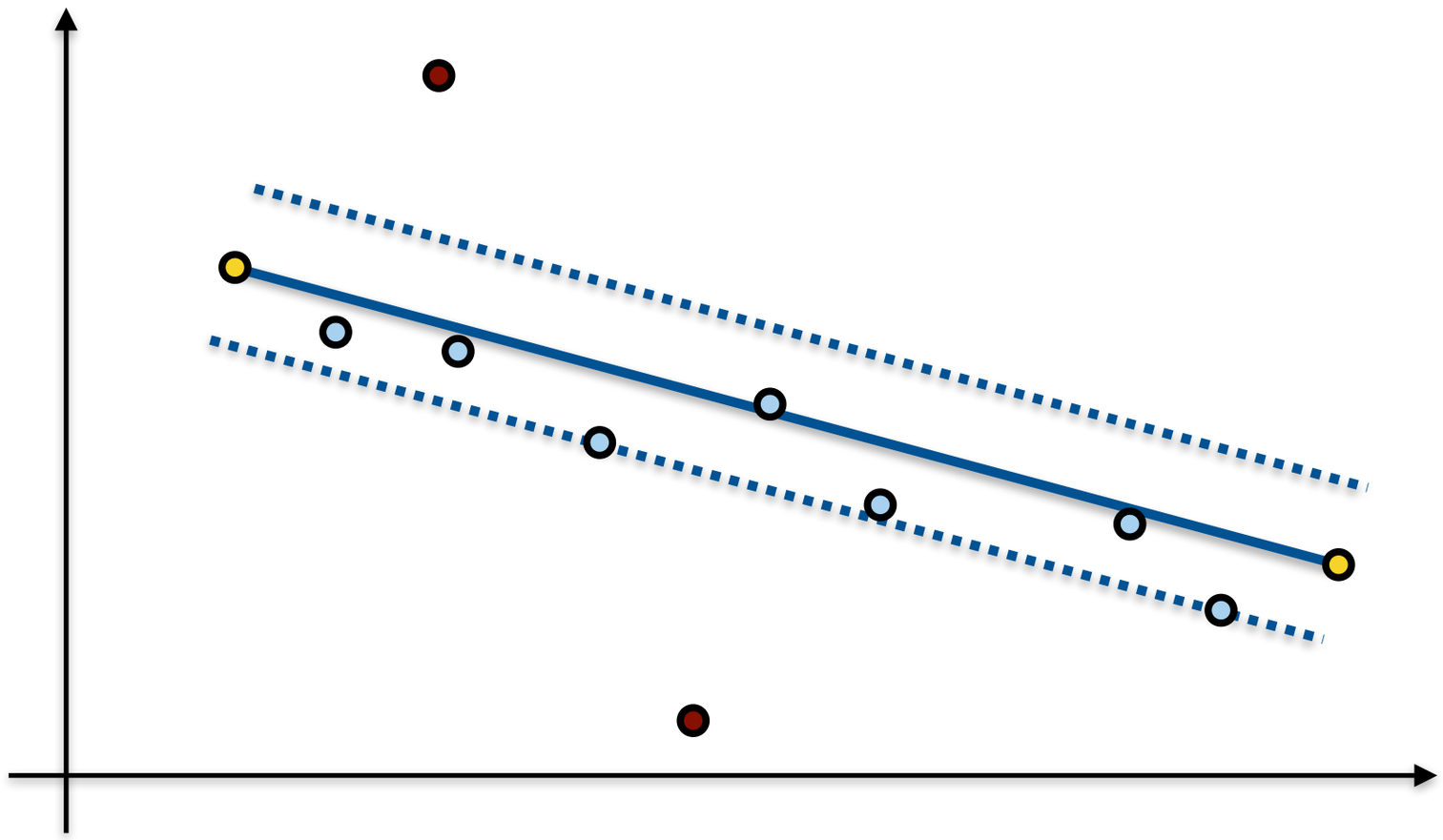


Quite bad..

3 inliers
8 outliers

Random Sample Consensus (RANSAC)

- Let’s try yet another 2 random points.

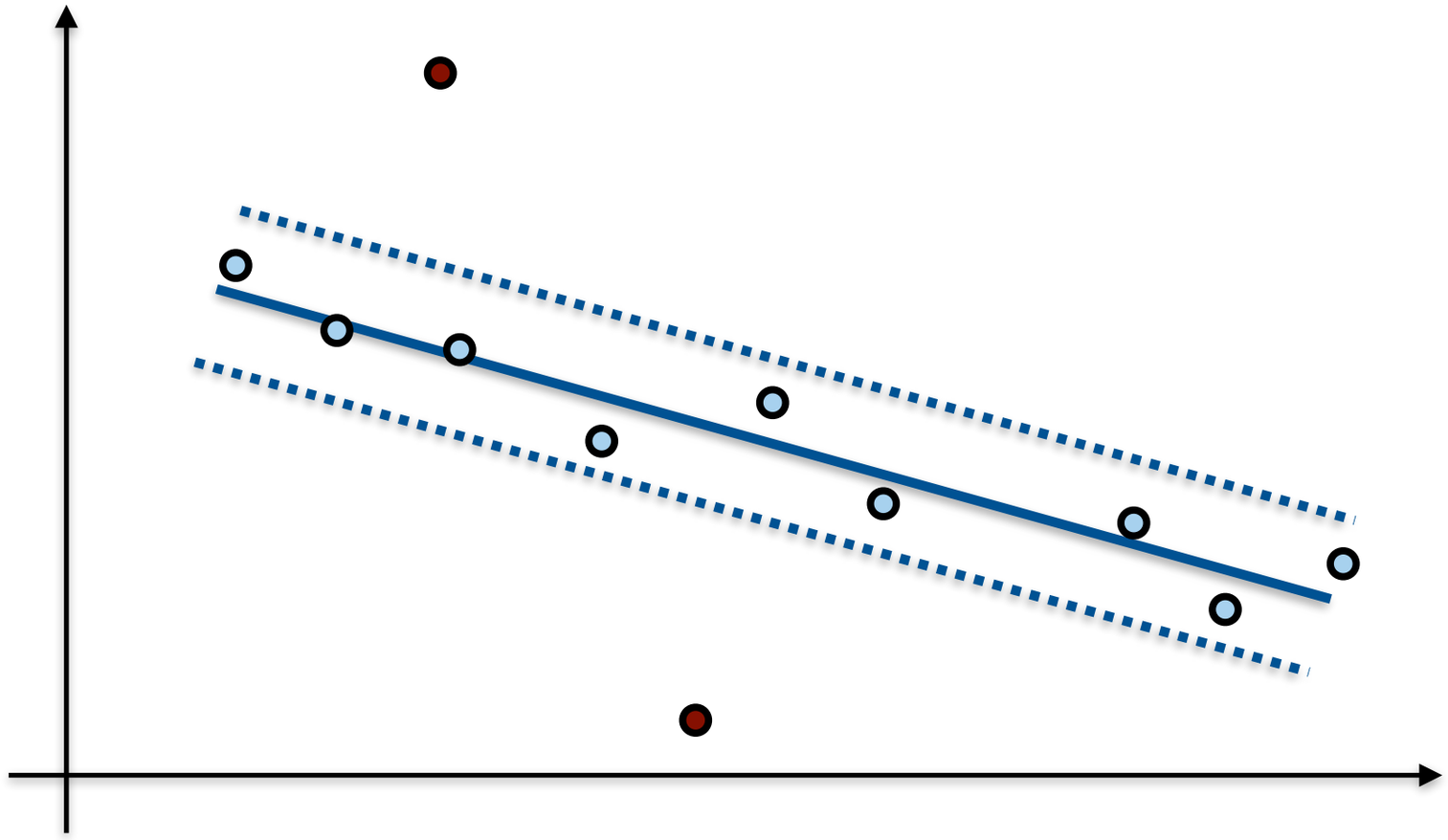


Quite good!

9 inliers
2 outliers

Random Sample Consensus (RANSAC)

- Let’s use the inliers of the best trial so far to perform least squares fitting.



Even better!

Epipolar Constraint

Epipolar Constraint on the Image Plane

- The epipolar constraint can be computed for a pair of matched 2D image points when the **relative camera pose is known**, here encoded by R, T (e.g. stereo pair)
- The **epipolar plane** is spanned by $\mathbf{x}_\mathrm{L}$ and the two camera centers O_L and O_R
- The **epipolar line** is the intersection of the epipolar plane and the right image plane
- The **epipolar constraint** encodes that $\mathbf{x}_\mathrm{R}$ must lie on the epipolar line in the right image

$$\mathbf{x}_L^\top \hat{T} R \, \mathbf{x}_R = 0$$

- $E = \hat{T} R$ is called the **Essential Matrix**
- Note that the epipolar constraint is not sufficient to guarantee a correct match. A wrong match my still lie elsewhere on the epipolar line. However, in many cases outliers can be correctly filtered.

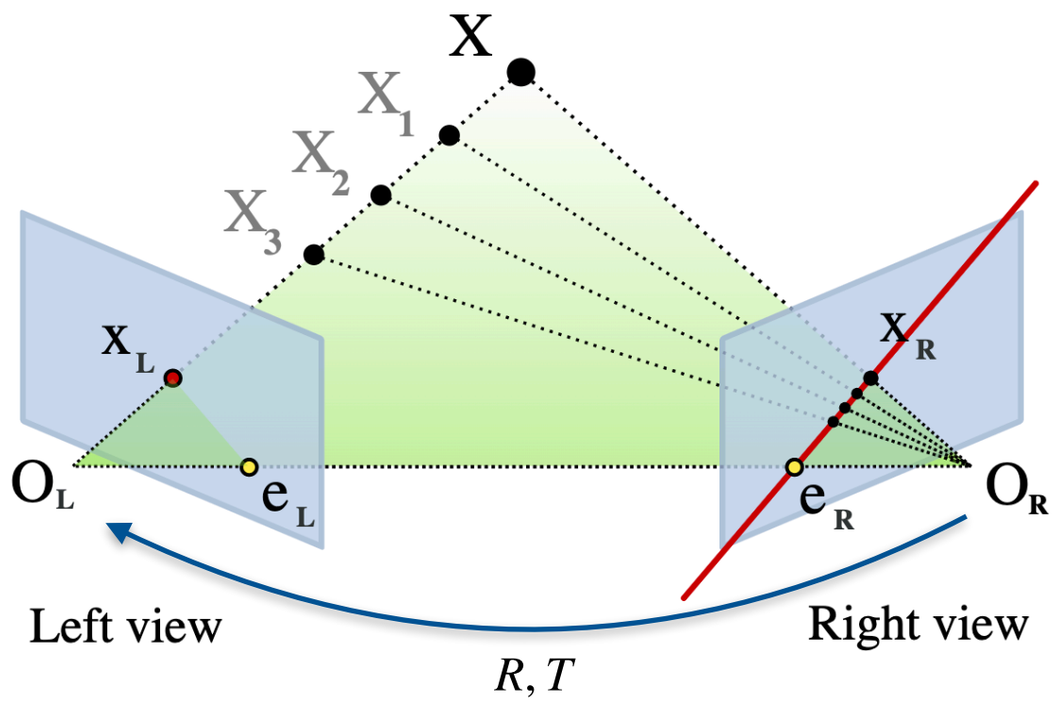


Image: “Epipolar geometry” by Arne Nordmann (norro) used under CC BY-SA 3.0 / relative pose R, T added

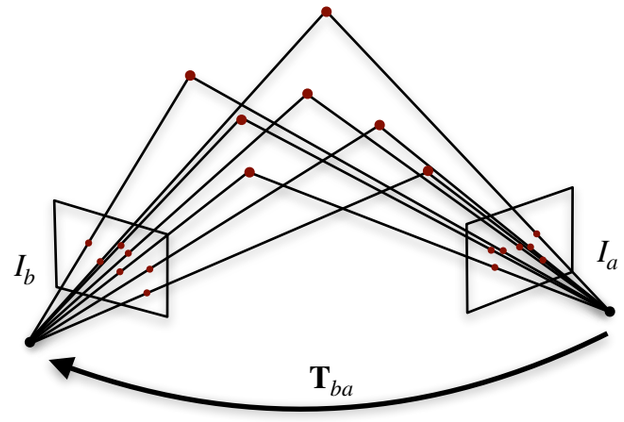
Motion Estimation

Motion Estimation from Point Correspondences

- **2D-to-2D**
  - Reprojection error:

$$E\left(\mathbf{T}_{ba}, \mathcal{X}\right) = \sum_{i=1}^{N} \left\| \mathbf{y}_{a,i} - \pi\left(\mathbf{X}_i\right) \right\|^2 + \left\| \mathbf{y}_{b,i} - \pi\left(\mathbf{T}_{ba}(\mathbf{X}_i)\right) \right\|^2$$

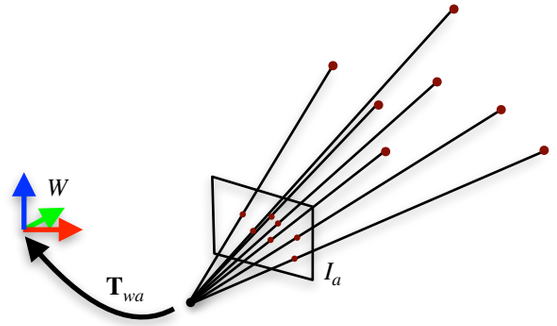
  - Linear Algorithm: 8-point, 5-point



- **2D-to-3D**
  - Reprojection error:

$$E\left(\mathbf{T}_{wa}, \mathcal{X}\right) = \sum_{i=1}^{N} \left\| \mathbf{y}_{a,i} - \pi\left(\mathbf{T}_{wa}^{-1}(\mathbf{X}_i)\right) \right\|^2$$

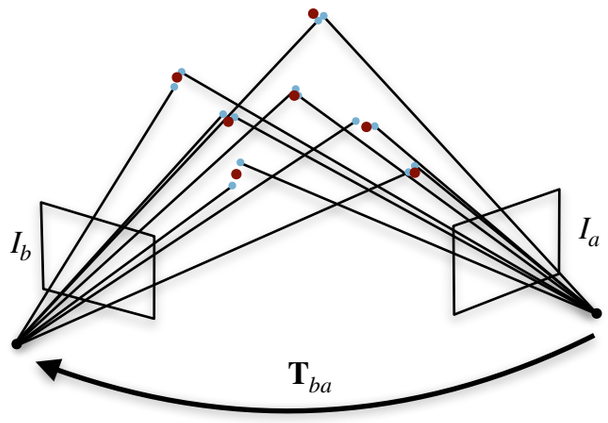
  - Linear Algorithm: DLT PnP



- **3D-to-3D**
  - 3D geometric error:

$$E\left(\mathbf{T}_{ab}\right) = \sum_{i=1}^{N} \left\| \mathbf{Y}_{a,i} - \mathbf{T}_{ab}(\mathbf{Y}_{b,i}) \right\|^2$$

  - Linear Algorithm: Arun, Horn



2D-to-2D Motion Estimation

- Given corresponding image point observations

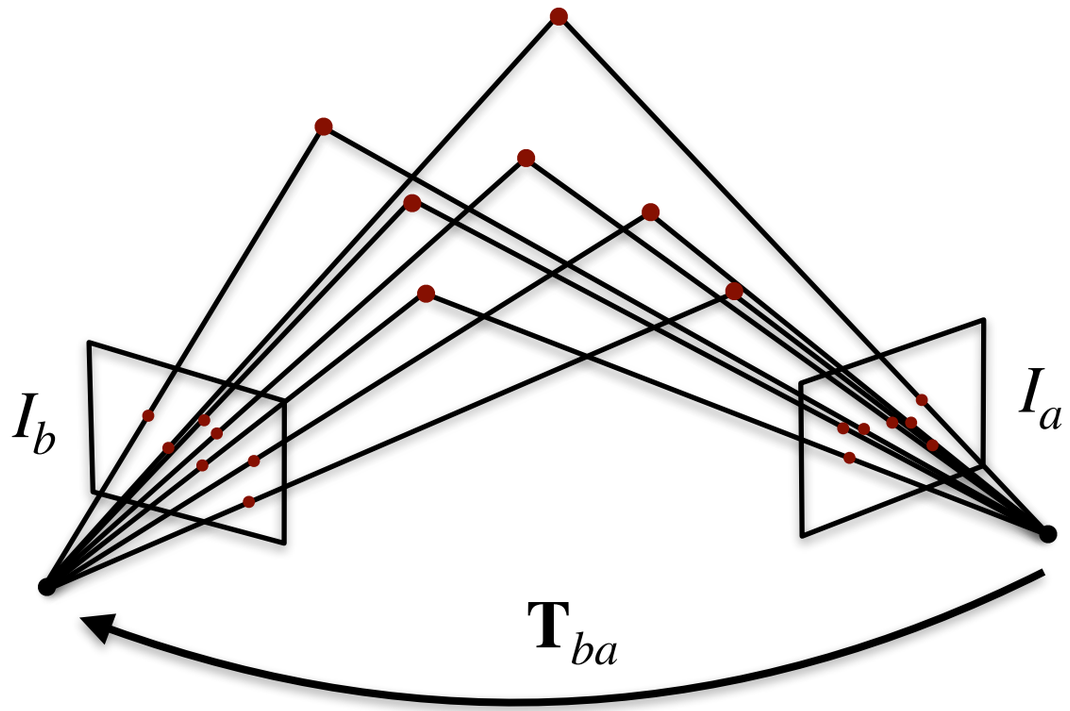
  $\mathcal{Y}_a = \{\mathbf{y}_{a,1}, \ldots, \mathbf{y}_{a,N}\}$

  $\mathcal{Y}_b = \{\mathbf{y}_{b,1}, \ldots, \mathbf{y}_{b,N}\}$

  of unknown 3D points $\mathcal{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$

  (expressed in camera frame a)

  determine relative motion $\mathbf{T}_{ba}$ between the frames



- Reprojection Error (Bundle Adjustment):

$$E\left(\mathbf{T}_a^b, \mathcal{X}\right) = \sum_{i=1}^{N} \left\| \mathbf{y}_{a,i} - \pi\left(\mathbf{X}_i\right) \right\|^2 + \left\| \mathbf{y}_{b,i} - \pi\left(\mathbf{T}_{ba}(\mathbf{X}_i)\right) \right\|^2$$

  Non-linear optimization requires good initialization. Non-convex, non-unique (scale ambiguity).

- Algebraic approach based on epipolar geometry to recover relative pose (up to scale) without explicitly recovering 3D point location: 8-point, 5-point algorithm
- Applications:
  - Filtering pairwise feature matches with RANSAC
  - Monocular SLAM / SfM initialisation

2D-to-3D Motion Estimation

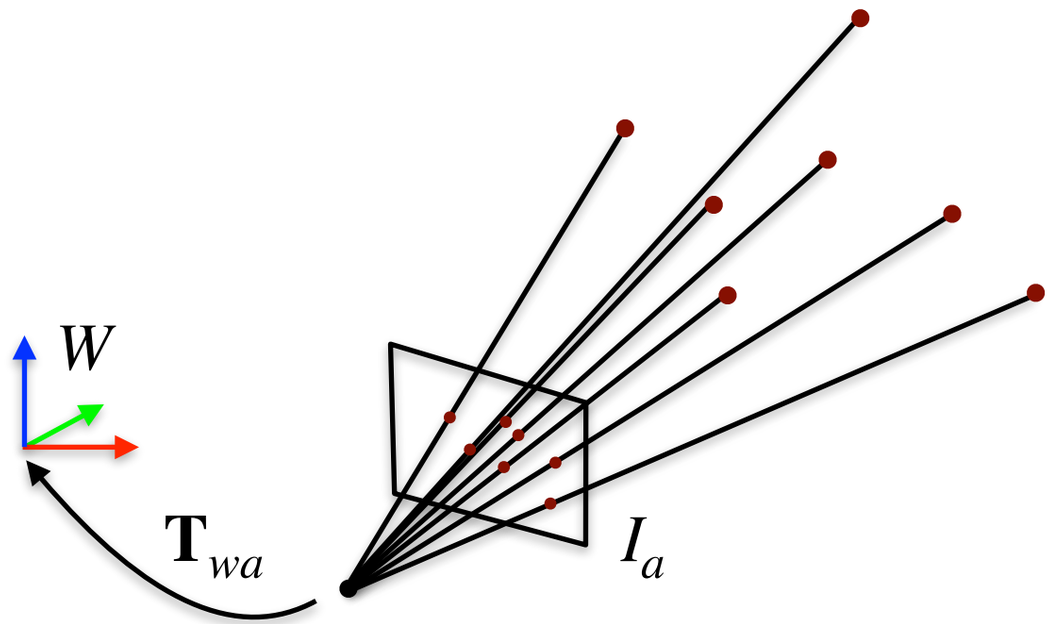
- Given a set of 3D points in world frame W

  $\mathcal{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$

  and corresponding image observations

  $\mathcal{Y}_a = \{\mathbf{y}_{a,1}, \ldots, \mathbf{y}_{a,N}\}$

  determine camera pose $\mathbf{T}_{wa}$ in world frame



- Reprojection Error (Pose-only Bundle Adjustment):

$$E\left(\mathbf{T}_{wa}, \mathcal{X}\right) = \sum_{i=1}^{N} \left\| \mathbf{y}_{a,i} - \pi\left(\mathbf{T}_{wa}^{-1}(\mathbf{X}_i)\right) \right\|^2$$

  Non-linear optimization requires good initialization. Non-convex.

- A.k.a. Perspective-n-Points (PnP) problem, many approaches exist, e.g.
  - Direct linear transform (DLT)
  - EPnP (Lepetit et al., An accurate O(n) Solution to the PnP problem, IJCV 2009)
  - OPnP (Zheng et al., Revisiting the PnP Problem: A Fast, General and Optimal Solution, ICCV 2013)
- Applications:
  - Localize camera in local keypoint map (with RANSAC)

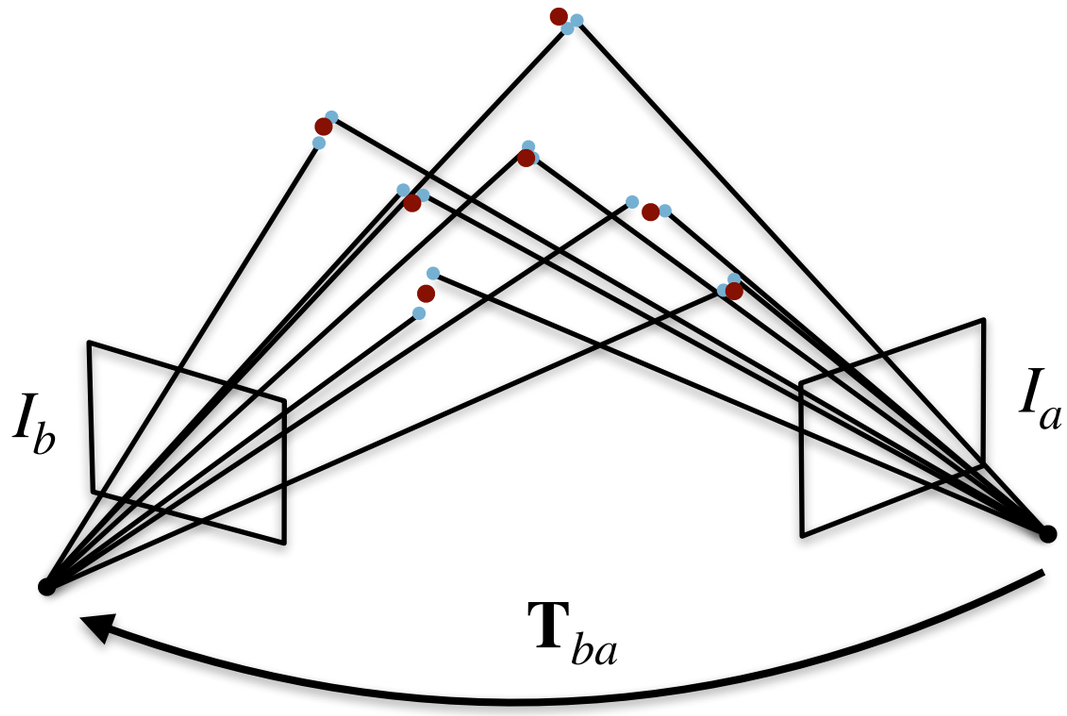
3D-to-3D Motion Estimation

- Given corresponding 3D points in two camera frames

$$\mathcal{Y}_a = \{\mathbf{Y}_{a,1}, \ldots, \mathbf{Y}_{a,N}\}$$

$$\mathcal{Y}_b = \{\mathbf{Y}_{b,1}, \ldots, \mathbf{Y}_{b,N}\}$$

  determine the relative camera pose $\mathbf{T}_{ab}$



- 3D geometric error:

$$E(\mathbf{T}_{ab}) = \sum_{i=1}^{N} \| \mathbf{Y}_{a,i} - \mathbf{T}_{ab}(\mathbf{Y}_{b,i}) \|^2$$

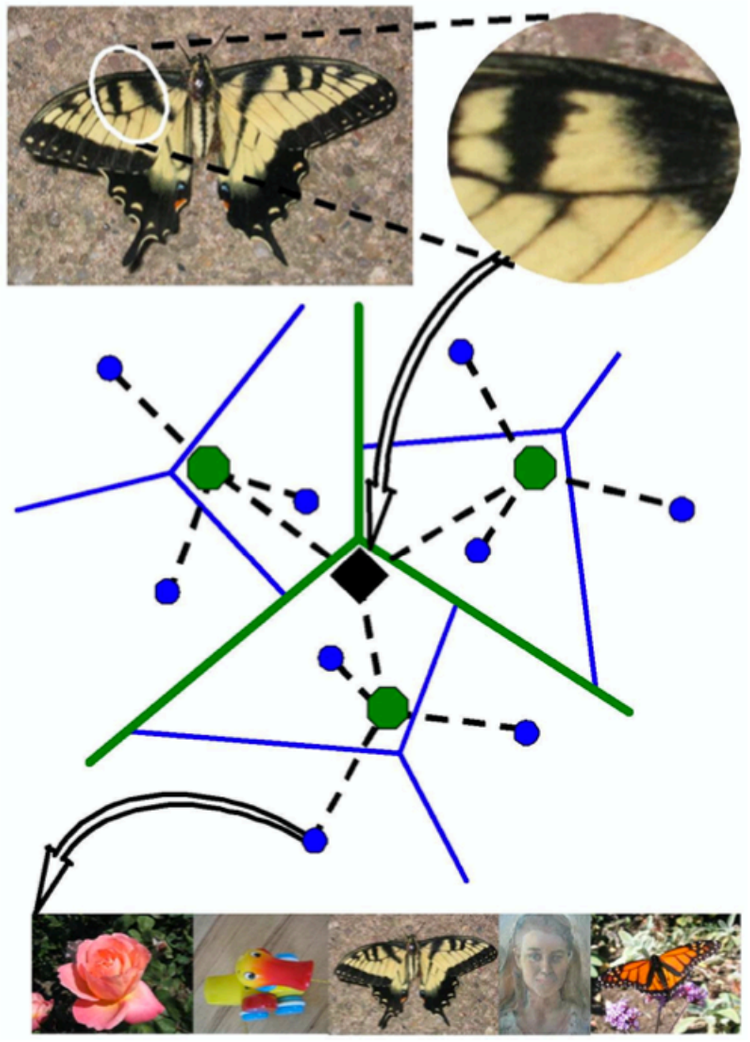
  Corresponds to least-squares point cloud alignment.

- Closed-form solutions available, e.g. Arun et al., 1987
- Applications:
  - Relative pose for calibrated stereo cameras (triangulation of 3D points) or RGB-D cameras (measured depth)
  - Loop-closure correction (variant with scale estimate available for monocular SLAM)

Place Recognition

Place Recognition with Bag of Words

- Place recognition aims to find similar images to a given query image
  - SfM: Which images to match?
  - SLAM: Detect loops
- Idea: Discretize the feature-descriptor space by hierarchical clustering in a “vocabulary tree”
  - visual “words” correspond to leaf-nodes
  - words are weighted by distinctiveness: e.g. “inverse document frequency” $\log(N/N_i)$
- Image comparison:
  - each feature is assigned to a word by passing it down the tree
  - for an image, count occurrence of each word (“term frequency”): bag-of-words vector
  - to compare images, compute the distance of (normalized) bag-of-words vectors
  - close bag-of-words vectors correspond to potentially similar images
- Vocabulary tree and weights are built offline from large collection of features
- False positives possible: Combine with geometric, temporal, … consistency checks



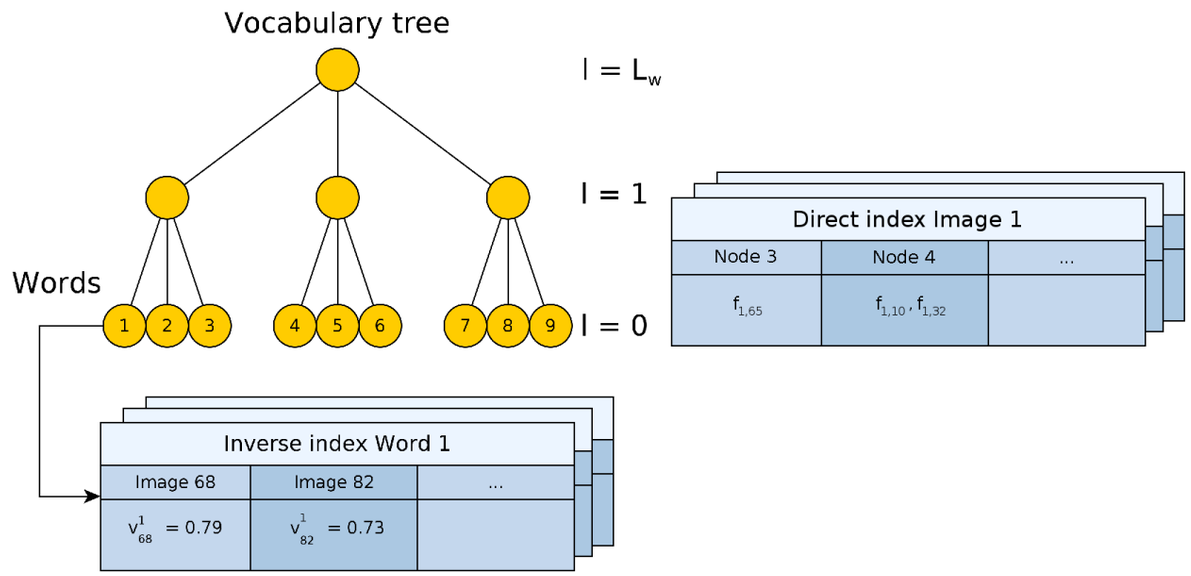
Nistér & Stewénius, CVPR 2006

Efficient Query and Feature Matching

- Image query with “inverse index”:
  - For each word store list of images, and for each image cache the word count.
  - During query, consider only images from the inverse index of each word
- Efficient BoW distance (scoring):
  - BoW vectors are sparse (most entries are 0)
  - For normalised vectors, use

$$\|\mathbf{q}-\mathbf{d}\|_p^p = 2 + \sum_{i|q_i\neq 0, d_i\neq 0} \left(\left|q_i - d_i\right|^p - \left|q_i\right|^p - \left|d_i\right|^p \right)$$

  - L1-norm is often used ($p = 1$)



Gálvez-López & Tardós, T-RO 2012

- Approximate nearest neighbour feature matching using “direct index”
  - For each image, store occurrence of nodes at each tree level and list of corresponding features
  - When matching images, only consider features from direct image at given level

Lessons Learned

- Keypoint detection, description and matching is a well researched topic
- Highly performant corner and blob detectors exist
- Descriptors can be invariant to translation, rotation, scale, viewpoint, brightness, … (e.g. SIFT)
- Binary descriptors are highly efficient and effective (e.g. ORB) and thus popular for real-time
- Keypoint matching by descriptor distance
- Robust matching based on model fitting using RANSAC
- Motion estimation from 2D-to-2D, 2D-to-3D, 3D-to-3D matches
- Image retrieval and loop-closure detection with bag-of-words

Exercise Sheet 3

In the third exercise sheet you will:

- Implement rotation estimation and descriptor matching for a simple variant of ORB.
- Filter inlier matches for stereo pairs using the epipolar constraint.
- Filter inlier matches for arbitrary pairs using RANSAC with the 5-point algorithm.
- Implement bag-of-words for efficient image retrieval using an inverse index.