Local feature matching and rigid objects recognition

Yann Gousseau
Telecom Paris - IP Paris

Cours IMA201

Introduction

The goal is to decide wether a given object (or part of a scene) is present.
Also called *instance recognition*.
Method :

- which are the similar **geometrical features** between these images ?
- are they coherent ?

Introduction

The goal is to decide wether a given object (or part of a scene) is present.
Also called *instance recognition*.
Method :

- which are the similar **geometrical features** between these images ?
- are they coherent ?

Introduction

Also :
is there any similar **geometrical features** between these images ? Are they coherent ?

Introduction

Motivation and requirements

Problem: finding correspondences between images.

Applications

- comparison of images content;
- research in a database;
- object detection/recognition;
- image stiching;
- 3D reconstruction, stereo...

Usual requirement: invariance or robustness to

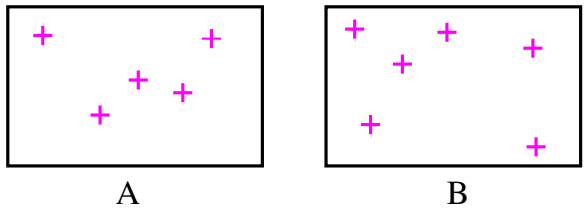
- illumination (contrast changes),
- scale and viewpoint (local similarity, affine or projective transformations),
- occlusion,
- noise

Motivation and requirements

Methodology

Four steps

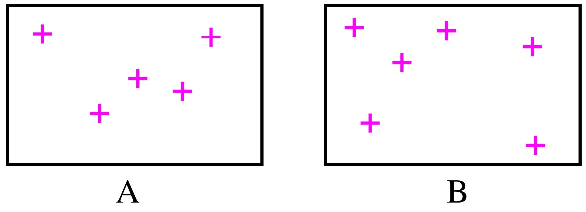
1. Extraction of local features: invariance or robustness requirements;
2. Feature comparison: distance between features;
3. Decision: thresholds on the distance and matching;
4. Grouping of previous matching in coherent rigid transformations.



Methodology

Four steps

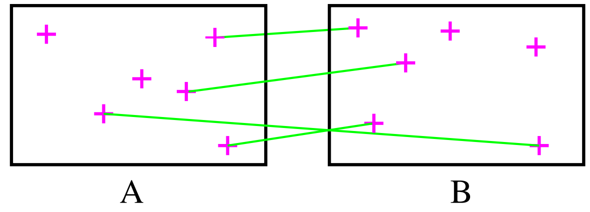
1. Extraction of local features: invariance or robustness requirements;
2. Feature comparison: distance between features;
3. Decision: thresholds on the distance and matching;
4. Grouping of previous matching in coherent rigid transformations.



Methodology

Four steps

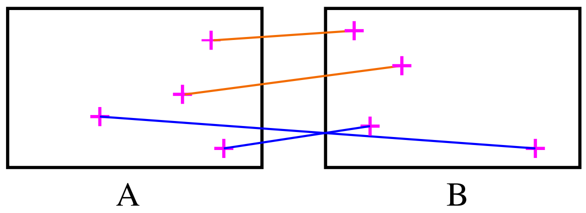
1. Extraction of local features: invariance or robustness requirements;
2. Feature comparison: distance between features;
3. Decision: thresholds on the distance and matching;
4. Grouping of previous matching in coherent rigid transformations.



Methodology

Four steps

1. Extraction of local features: invariance or robustness requirements;
2. Feature comparison: distance between features;
3. Decision: thresholds on the distance and matching;
4. Grouping of previous matching in coherent rigid transformations.



Part I

Local descriptors

Local representation choice

Many local features have been proposed in the literature:

- local descriptors: SIFT [Lowe, 1999], GLOH [Mikolajczyk, Schmid, 2005], PCA-SIFT [Ke, Sukthankar, 2004], SURF [Bay *et al*, 2006], [Dalal *et al*, 2005], [Rabin *et al*, 2009], BRIEF [Calonder *et al*, 2012], etc...
- region and boundary descriptors: Shape Context [Belongie, Malik, 2000], shapes [Monasse, Guichard, 2000], MSER [Matas *et al*, 2002], boundary pieces [Muse *et al* 2006], etc.
- Most recent trend : local descriptors are obtained through pre-trained CNNs ([Szegedy 2013, LIFT Yi et al. 2016, Superpoints De Tone et al. 2018, Dusmanu et al. 2019, SuperGlue De Tone et al. 2022])
  $\rightarrow$ IMA 205, IMA206, 3A

SIFT decriptors

In the following, we focus on a generic local features extraction pipeline, as introduced in [Lowe 1999]
SIFT : scale invariant feature transform

- keypoints extraction Relies on the ideas introduced in the first part of the lecture (linear scale-space + differential operators)
- features computation Local histograms of the orientation of the gradient
- features comparison distance + grouping strategy

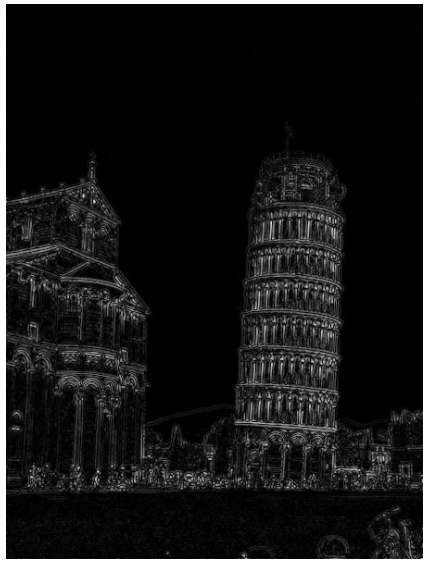
Local feature extraction

1. Discrete image u
2. Linear scale-space representation $\forall \sigma, \quad u_\sigma = g_\sigma * u$
3. Local extrema $(\vec{x}, \sigma)$ in space and scale of $\sigma^2 \Delta u_\sigma$
4. Hessian criterion to eliminate edge points $\rightarrow$ interest points $(\vec{x}, \sigma)$.
5. Main orientations (direction of ∇u_σ) assigned at each point $\rightarrow$ interest points $(\vec{x}, \sigma, \theta)$.

Local feature extraction

1. Discrete image u
2. Linear scale-space representation $\forall \sigma, \quad u_\sigma = g_\sigma * u$
3. Local extrema $(\vec{x}, \sigma)$ in space and scale of $\sigma^2 \Delta u_\sigma$
4. Hessian criterion to eliminate edge points $\rightarrow$ interest points $(\vec{x}, \sigma)$.
5. Main orientations (direction of ∇u_σ) assigned at each point $\rightarrow$ interest points $(\vec{x}, \sigma, \theta)$.

Local feature extraction

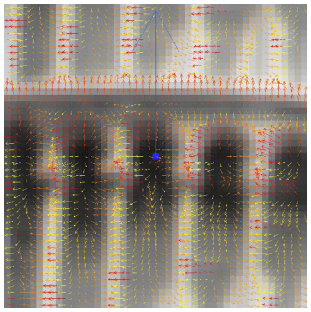


1. Discrete image u
2. Linear scale-space representation $\forall \sigma, \quad u_\sigma = g_\sigma * u$
3. Local extrema $(\vec{x}, \sigma)$ in space and scale of $\sigma^2 \Delta u_\sigma$
4. Hessian criterion to eliminate edge points $\rightarrow$ interest points $(\vec{x}, \sigma)$.
5. Main orientations (direction of ∇u_σ) assigned at each point $\rightarrow$ interest points $(\vec{x}, \sigma, \theta)$.

Local feature extraction

1. Discrete image u
2. Linear scale-space representation $\forall \sigma, \quad u_\sigma = g_\sigma * u$
3. Local extrema $(\vec{x}, \sigma)$ in space and scale of $\sigma^2 \Delta u_\sigma$
4. Hessian criterion to eliminate edge points $\rightarrow$ interest points $(\vec{x}, \sigma)$.
5. Main orientations (direction of ∇u_σ) assigned at each point $\rightarrow$ interest points $(\vec{x}, \sigma, \theta)$.

Local feature extraction

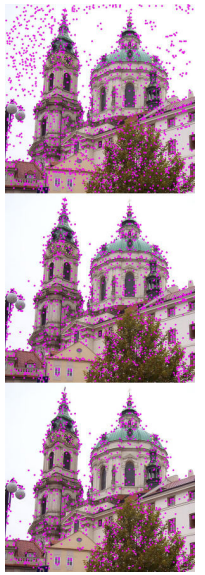


1. Discrete image u
2. Linear scale-space representation $\forall \sigma, \quad u_\sigma = g_\sigma * u$
3. Local extrema $(\vec{x}, \sigma)$ in space and scale of $\sigma^2 \Delta u_\sigma$
4. Hessian criterion to eliminate edge points $\rightarrow$ interest points $(\vec{x}, \sigma)$.
5. Main orientations (direction of ∇u_σ) assigned at each point $\rightarrow$ interest points $(\vec{x}, \sigma, \theta)$.

Keypoints extraction

- Computation of Dog (approximation of LoG) for a sequence $\sigma_k = \sigma_0 r^k$: see the first part of the lecture)
- keypoints are selected as **local scale-space extrema of the DoG** (top image)
- keypoints are further selected by
  - thresholding the DoG response (middle image)
  - applying the Hessian criterion (first part of the lecture) (bottom image)
    $\rightarrow$ edge points are discarded

(Fig. court. of Lukas Mach, wikipedia, cc)



Keypoints extraction

- In practice, images are downsampled at each octave (i.e. by a factor of 2 each time σ_k reaches 2^l, some l)
- This necessitates some refinement of the spatial position

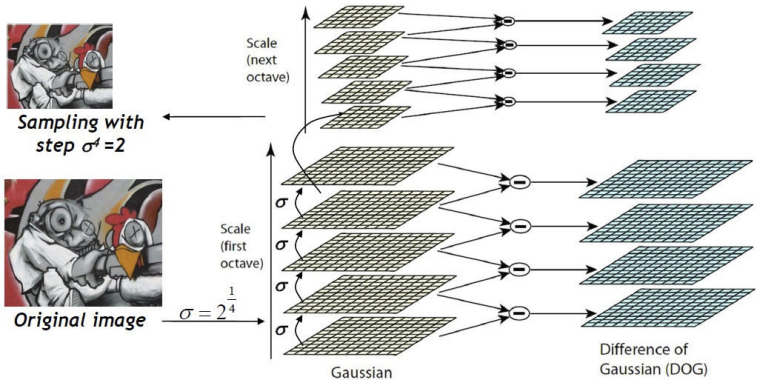
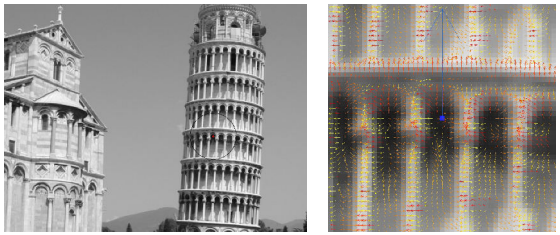


Fig. from D. Lowe 2004, T. Tuytelaars and K. Mikolajczyk 2008

keypoints extraction

- After the previous step : a set of keypoints $(\vec{x}, \sigma)$
- at each point, a **main orientation** is computed as the maximum of a local histogram of gradient orientation : $\theta = \arctan(\frac{u_y}{u_x})$
- a second orientation may be computed if it is strong enough $\rightarrow$ two keypoints are encoded
- **eventually, we get a set of keypoints** $(\vec{x}, \sigma, \theta)$



Important remark : θ is invariant to any contrast change (i.e. $u \rightarrow g \circ u$) (proove it !)

Computation of the SIFT features

Construction of a local descriptor a at each interest point $(\vec{x}, \sigma, \theta)$.

- Over a Mask around $\vec{x}$:
  - M sectors ($M = 4 \times 4 = 16$ in the original SIFT),
  - size proportional to σ: **scale invariance**.
  - orientations computed with respect to θ: **rotation invariance**
- Local descriptor $a = (a_1, \ldots a_M)$

  a_m = histogram of the gradient orientation weighted by the gradient magnitude, in the m^{th} sector (8 bins in the original SIFT).
- Eventually, the local descriptor a ($16 \times 8 = 128$ in the original SIFT) is normalized (sum to one): **invariance to local affine contrast change**

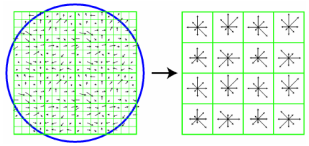
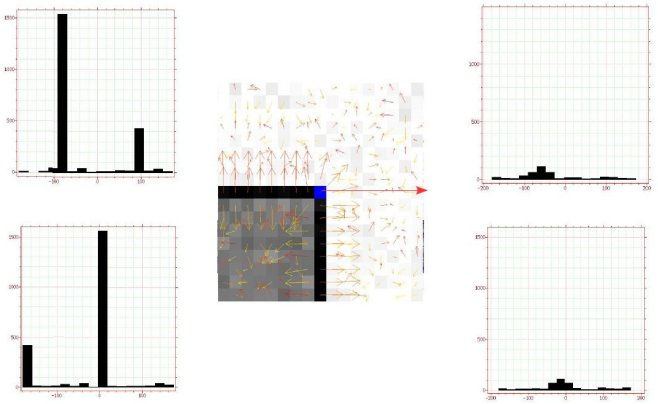


Fig. from Pham et al. 2008

Example with $M = 2 \times 2$



Part II

Dissimilarity measure

Features comparison

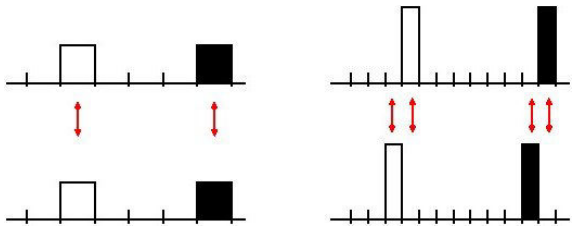
Two features a and b are simply compared using the euclidean distance in $\mathbb{R}^n$ ($n = 128$ for SIFT)

$$d^2(a,b) = \sum_{i=1}^{n} (a_i - b_i)^2$$

Can we do better than Euclidean distance ?

A small digression ....

Bin-to-bin distances (Euclidean, L^1...) not robust to histogram quantization and small angular shifts.

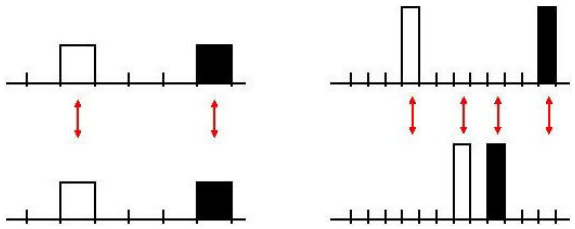


Dissimilarity measure

Can we do better than Euclidean distance ?

A small digression ....

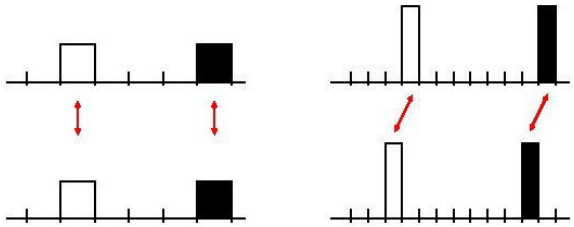
Bin-to-bin distances (Euclidean, L^1...) not robust to histogram quantization and small angular shifts.



Can we do better than Euclidean distance ?

A small digression ....

Earth Mover's (Wasserstein) distance.



Part III

Matching Experiments

Different decision criteria

Usual criteria used to validate matches:

1. DT: Global threshold (independent of a^i) on the distance D;
2. NN-DT: Global threshold + restriction to the nearest neighbor (avoids many false detections);
3. NN-DR [Lowe]: threshold on the ratio between the distances to the two nearest neighbors.
   Let b^{i_1} and b^{i_2} be the closest and second-closest neighbor of a^i, the match between a^i and b^{i_1} is validated **if**

$$\frac{D(a^i, b^{i_1})}{D(a^i, b^{i_2})} \leqslant r.$$

The third criterion is very efficient, but strongly limited in the case of repetitive structures (e.g. buildings) .

Can experiment

DT matching criterion used with distance CEMD.

Can experiment

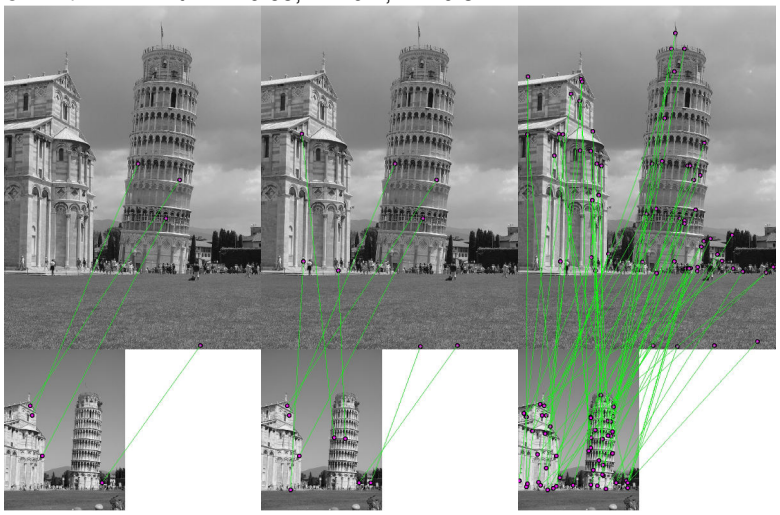
Euclidean distance + NN-DR matching criterion at $r = 0.8$.

Can experiment

Euclidean distance + NN-DT matching criterion.

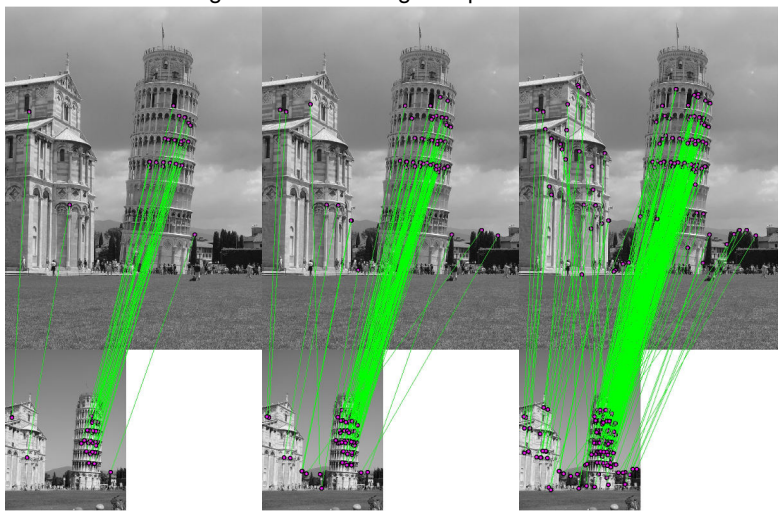
Pisa

CEMD+NN-DR with $r = 0.65$, $r = 0.7$, $r = 0.8$

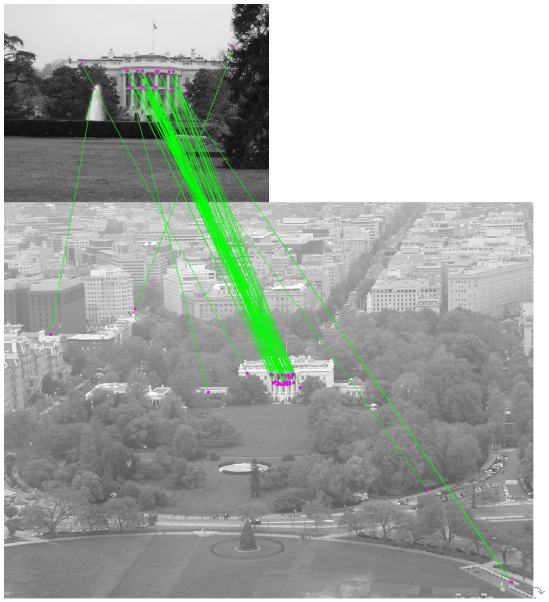


frame

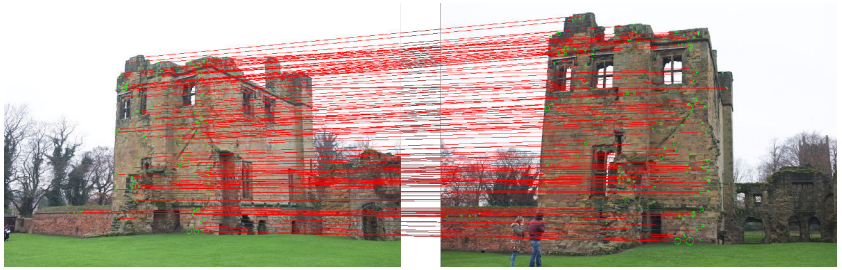
Alternative matching criterion enabling multiple matches



White House



Another example

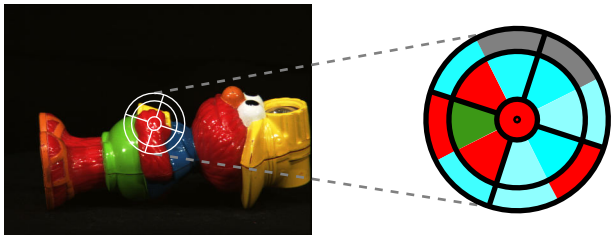


Extracted from IPOL demos
→ "Anatomy of the SIFT method", IPOL online publication and demo

Part IV

And what about color ?

Descripteur local couleur



Without color

With color

Part V

Grouping matches

How to group matches ?

Two different approaches:

1. grouping matches in parameter space **Hough transform**
   - \+ : Multiple group detection
   - – : Sampling, time complexity, “Ghost” group
2. grouping matches by transformation consensus: **RANSAC** algorithm
   - \+ : Fast
   - – : Parameter tuning, single group detection

How to group matches ?

Two different approaches:

1. grouping matches in parameter space **Hough transform**
   - $+$: Multiple group detection
   - $-$: Sampling, time complexity, “Ghost” group
2. grouping matches by transformation consensus: **RANSAC** algorithm
   - $+$: Fast
   - $-$: Parameter tuning, single group detection

How to group matches ?

Two different approaches:

1. grouping matches in parameter space **Hough transform**
   - $+$: Multiple group detection
   - $-$: Sampling, time complexity, “Ghost” group
2. grouping matches by transformation consensus: **RANSAC** algorithm
   - $+$: Fast
   - $-$: Parameter tuning, single group detection

RANSAC

Let $\{(x_i, y_i)\}_{i=1,\dots,N}$ be matches between images.

Assume that n pairs enables to estimate a transform T.

Examples : translations $n = 1$, affine transforms $n = 3$.
Residuals are defined as $r_i = \max(d(x_i, T^{-1}y_i), d(Tx_i, y_i))$.

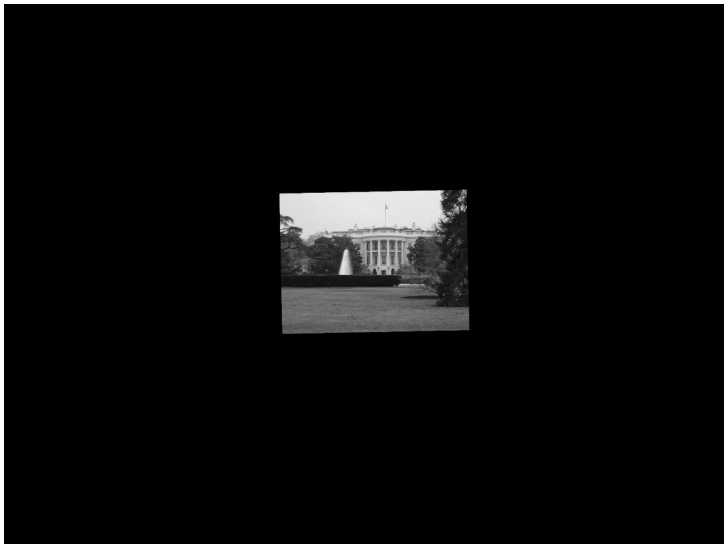
Fix a residual threshold D

Then, iterate

- draw n pairs and infer transformation T
- Let M be the number of pairs for which $r_i \leqslant D$ (a group)
- T is the best candidate if M is bigger than the biggest group found

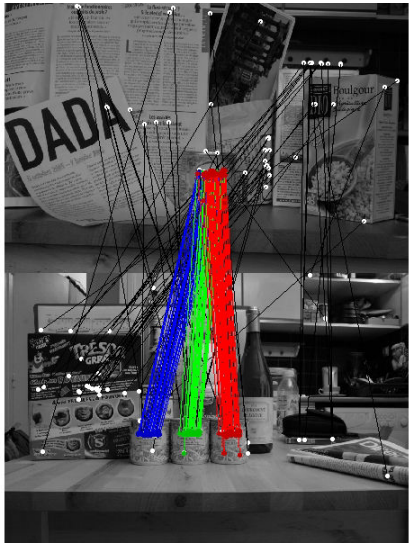
White house

White house



White house

Can experiment



Can experiment

Can experiment

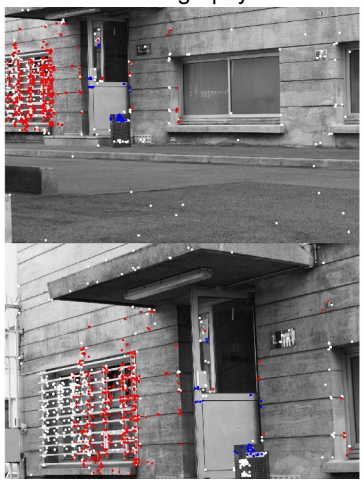
Can experiment

Pisa tower

Another example (homography)

Another example (homography)

Homography



Another example (homography)

Another example (homography)

Another example (homography)

Epipolar geometry

Deep features

- **Local features** can also be extracted by **deep neural networks** (Yi et al. 2016 “LIFT”, DeTone et al. 2018 “Superpoints”, etc.)
- The **matching and pose estimation** problem can also be solved by deep networks (DeTone et al. 2022, "Superglue", etc.)
- The **object detection** problem is efficiently adressed by the combination of classifying neural architecture and heuristics for the proposal of candidate object locations (Girschick et al. 2014, “RCNN”, Ren et al. 2016, “faster RCNN”, Redmon et al. 2016, 2018, 2022 “Yolo”, etc.)

$\rightarrow$ IM1205, IMA206, 3rd year, etc.