Visual scene real-time analysis for Intelligent Vehicles:

Visual ego-Localization

Pr. Fabien Moutarde
Center for Robotics
MINES ParisTech
PSL Université Paris

Fabien.Moutarde@mines-paristech.fr
http://people.mines-paristech.fr/fabien.moutarde

Acknowledgements

During preparation of these slides, I borrowed significant slide content from several sources, in particular:

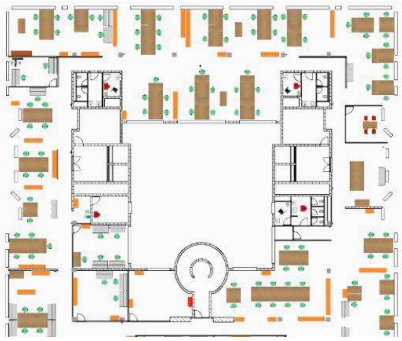
- Davide Scaramuzza (University of Zurich, Robotics and Perception Group, rpg.ifi.uzh.ch) : slides on « Visual Odometry and SLAM » from his IROS'2016 tutorial
  https://www.rsj.or.jp/databox/international/iros16tutorial_2.pdf
- Juan D. Tardos (Univ. Zaragoza) : slides on « Feature-based visual SLAM » from his ICRA'2016 tutorial
  http://www.dis.uniroma1.it/~labrococo/tutorial_icra_2016/
- Akihiko Torii (Tokyo Tech) : slides on « Visual place recognition » from his CVPR'2015 tutorial
  https://sites.google.com/site/lsvpr2015/placerecognition

Outline

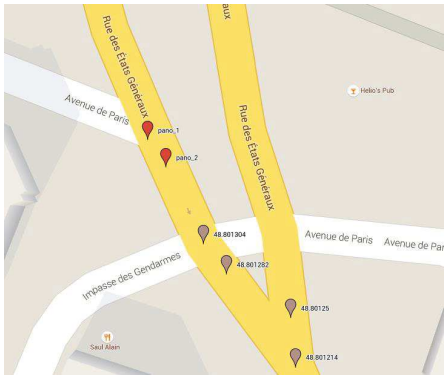
- **Definition, motivations, principle & methods overview**
- **Visual Odometry**
- **Tracked Features: visual keypoints**
- **Visual SLAM**
- **Place visual recognition**
- **Visual ego-localization *with Deep-Learning***

Visual ego-localization problematic

Where am I? (position+bearing)



Where am I? (position+bearing)



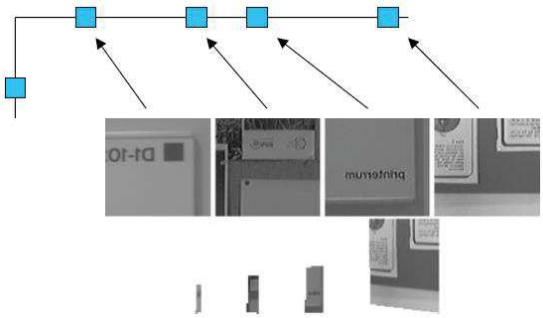
Visual ego-localization motivations

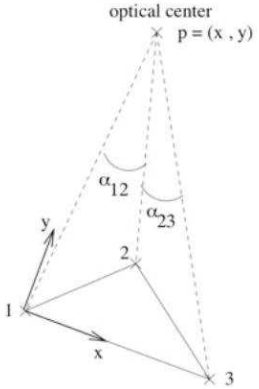
- GPS not always available (indoor, tunnels, underground parkings, « urban canyons »)
- GPS precision quite low (up to 10m error ! [except for differential GPS]
- GPS directly provides position but NOT the orientation (only the local orientation of TRAJECTORY can be estimated over time)
- Odometry is quite imprecise (cf. wheel slip!), and subject to large rapid cumulative errors
- Inertial Measurement Unit (IMU) expansive if precise, and subject to cumulative errors

Visual ego-localization approaches

- Triangulation of visual geo-tagged landmarks
- Visual Odometry
- Visual SLAM
- Place visual recognition
  + Visual ego-localization by keypoints matching with geo-tagged images

Visual ego-localization by triangulation using visual landmarks





$$\angle_{P12} = -\arctan\left(\frac{l_{23}\sin(\alpha_{12})\,(\sin(\alpha_{13})\cos(\angle_{123}) - \cos(\alpha_{13})\sin(\angle_{123}))}{l_{12}\sin(\alpha_{23}) + l_{23}\sin(\alpha_{12})\,(\cos(\alpha_{13})\cos(\angle_{123}) + \sin(\alpha_{13})\sin(\angle_{123}))}\right)$$

$$\angle_{23P} = -\angle_{P12} - \alpha_{13} + \angle_{123}$$

$$x = l_{13}\frac{\tan(\angle_{23P} + \angle_{132})}{\tan(\angle_{P12} + \angle_{213}) + \tan(\angle_{23P} + \angle_{132})}$$

$$y = l_{13}\frac{\tan(\angle_{23P} + \angle_{132})\tan(\angle_{P12} + \angle_{213})}{\tan(\angle_{P12} + \angle_{213}) + \tan(\angle_{23P} + \angle_{132})}$$

But difficult to have enough pre-recorded and geo-tagged landmarks everywhere!

Outline

- **Definition, motivations, principle & methods overview**
- **Visual Odometry**
- **Tracked Features: visual keypoints**
- **Visual SLAM**
- **Place visual recognition**
- **Visual ego-localization *with Deep-Learning***

Visual Odometry

Visual Odometry (VO) = estimating ego-movement of camera by analysis of the video

- **Possible only if:**
  - **Sufficient overlap between consecutive frames**
  - **Dominance of static scene over moving objects in the Field of View**
  - **Enough texture to allow extraction of apparent motion**

Visual Odometry working principle

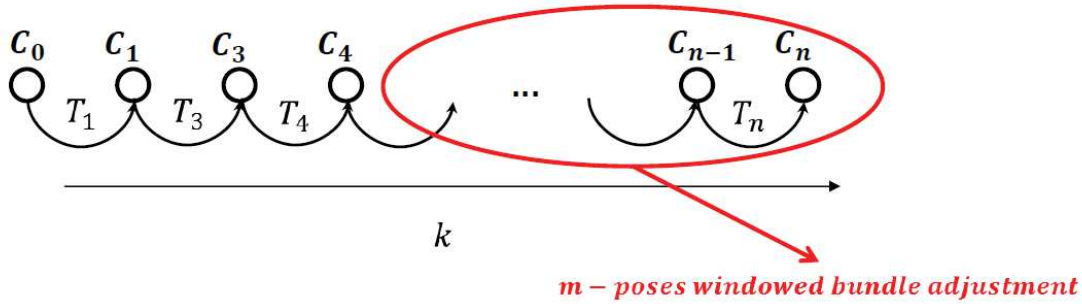
1. Compute the relative motion T_k from images I_{k-1} to image I_k

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}$$

2. Concatenate them to recover the full trajectory

$$C_n = C_{n-1} T_n$$

3. An optimization over the last m poses can be done to refine locally the trajectory (Pose-Graph or Bundle Adjustment)



Davide Scaramuzza – University of Zurich – Robotics and Perception Group - rpg.ifi.uzh.ch

Visual Odometry: estimation of relative motion

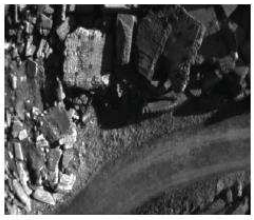
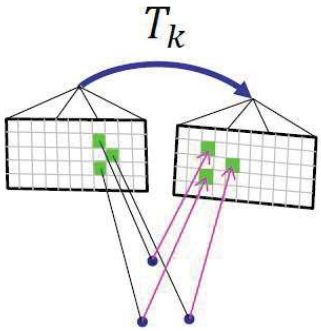


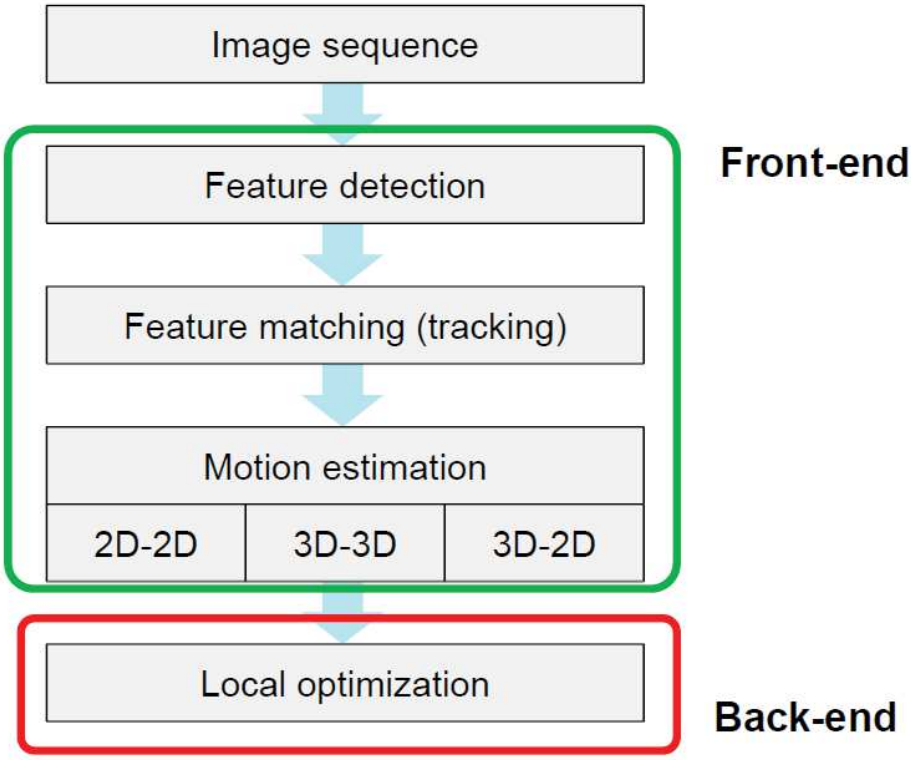
Image I_{k-1}

Image I_k



$$T_k = \arg\min_{\mathbf{T}} \iint_{\bar{R}} \rho \left[I_k\left(\pi\left(\mathbf{T}\cdot\pi^{-1}(\mathbf{u}, d_{\mathbf{u}})\right)\right) - I_{k-1}(\mathbf{u}) \right] d\mathbf{u}$$

Visual Odometry flow chart



Davide Scaramuzza – University of Zurich – Robotics and Perception Group - rpg.ifi.uzh.ch

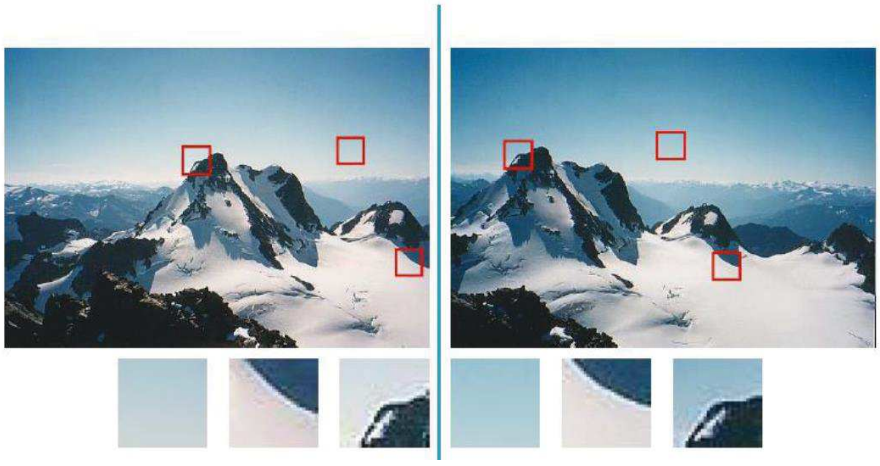
Outline

- **Definition, motivations, principle & methods overview**
- **Visual Odometry**
- **<u>Tracked Features: visual keypoints</u>**
- **Visual SLAM**
- **Place visual recognition**
- **Visual ego-localization *with Deep-Learning***

Features: visual keypoints

What are Good Features to Track ?

Which of the patches below can be matched reliably?



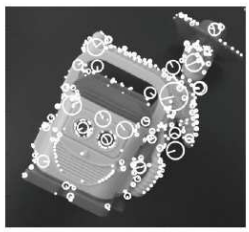
➔ **Choice of keypoints detector is critical !**

Visual keypoints DETECTORS

Corners vs Blob Detectors

- A **corner** is defined as the intersection of one or more edges
  - A corner has high localization accuracy
    - Corner detectors are good for VO
  - It's **less distinctive than a blob**
  - E.g., *Harris, Shi-Tomasi, SUSAN, FAST*

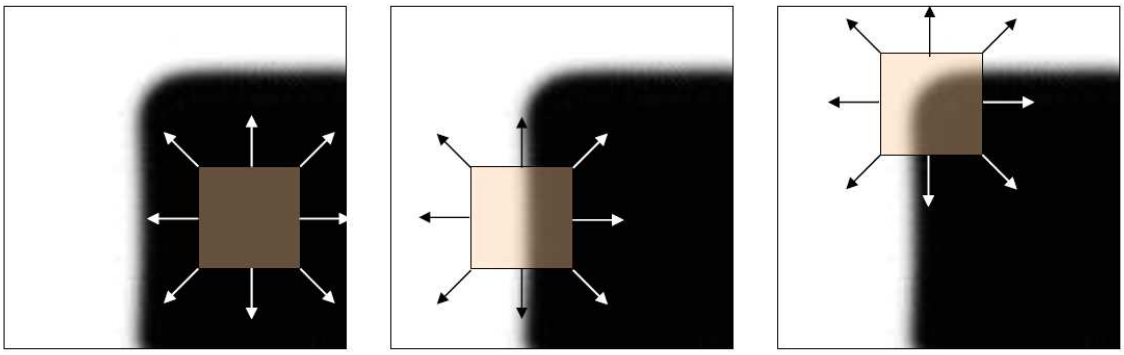
- A **blob** is any other image pattern, **which is not a corner**, that significantly differs from its neighbors in intensity and texture
  - **Has less localization accuracy than a corner**
    - **Blob detectors are better for place recognition**
  - It's **more distinctive than a corner**
  - E.g., *MSER, LOG, DOG (SIFT), SURF, CenSurE*



- ***Descriptor**: Distinctive feature **identifier***
  - ***Standard** descriptor: squared patch of pixel intensity values*
  - ***Gradient** or difference-based descriptors: SIFT, SURF, ORB, BRIEF, BRISK*

Harris corners detector (1988)

- How do we identify corners?
- We can easily recognize the point by looking through a small window
- Shifting a window in **any direction** should give a **large change** in intensity in at least 2 directions



"flat" region: no intensity change

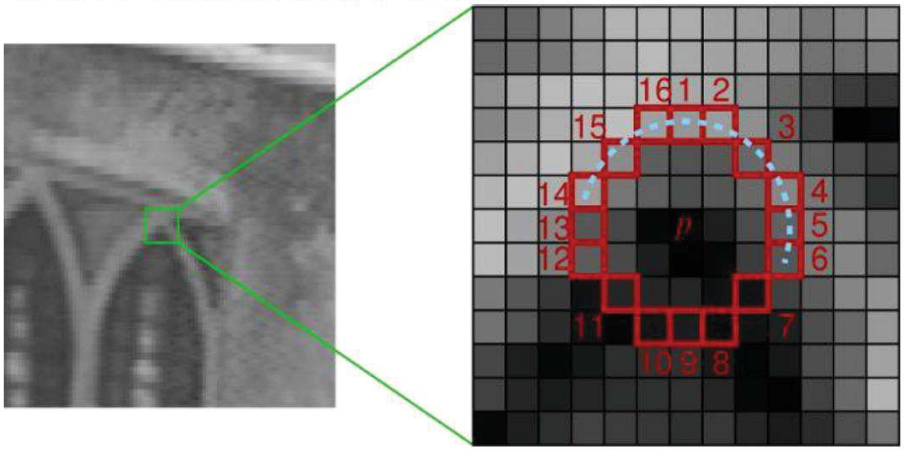
"edge": no change along the edge direction

"corner": significant change in at least 2 directions

FAST corners detector

FAST = Features from Accelerated Segment Test
[Rosten et al., PAMI 2010]

- Studies intensity of pixels on circle around candidate pixel *C*
- *C* is a FAST corner if a set of *N* contiguous pixels on circle are:
  - all brighter than *intensity_of(C)+theshold*, or
  - all darker than *intensity_of(C)+theshold*



- Typical FAST mask: test for 9 contiguous pixels in a 16-pixel circle
- Very fast detector - in the order of 100 Mega-pixel/second

Davide Scaramuzza – University of Zurich – Robotics and Perception Group - rpg.ifi.uzh.ch

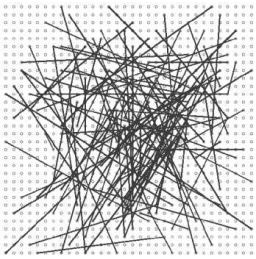
Keypoints DESCRIPTORS for Visual Odometry

BRIEF descriptor [Calonder et. al, ECCV 2010]

- **B**inary **R**obust **I**ndependent **E**lementary **F**eatures
- Goal: high speed (in description and matching)

- **Binary** descriptor formation:
  - Smooth image
  - **for each** detected keypoint (e.g. FAST),
  - ***sample*** 256 intensity pairs **p**=(p_1, p_2) within a squared patch around the keypoint
  - **for each pair p**
    - **if** $p_1 < p_2$ **then *set*** bit **p** of descriptor to **1**
    - **else *set*** bit **p** of descriptor to **0**

- The pattern is generated randomly only once; then, the same pattern is used for all patches


Pattern for intensity pair samples – generated randomly

- Not scale/rotation invariant
- Allows **very fast** Hamming Distance matching: count the number of bits that are different in the descriptors matched

ORB descriptor
[Rublee et al., ICCV 2011]

- ***O***riented FAST and ***R***otated ***B***RIEF
- Alterative to SIFT or SURF, designed for fast computation
- Keypoint detector based on **FAST**
- **BRIEF** descriptors are *steered* according to keypoint orientation (to provide rotation invariance)
- Good Binary features are learned by minimizing the correlation on a set of training patches.

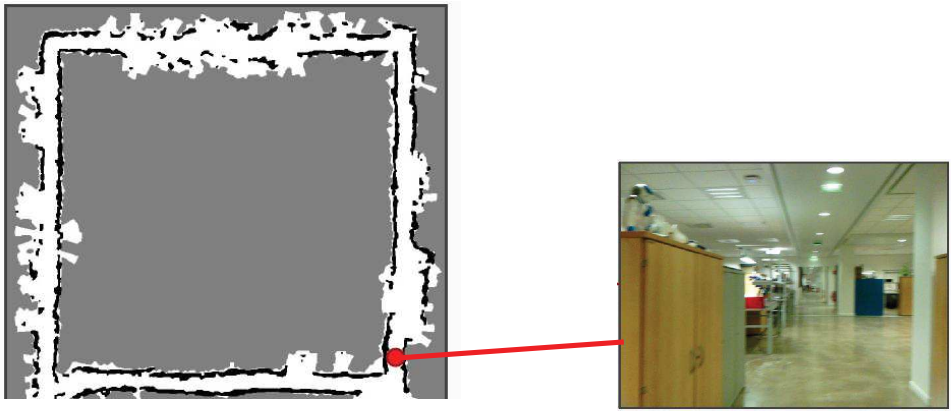
Outline

- Definition, motivations, principle & methods overview
- Visual Odometry
- Tracked Features: visual keypoints
- Visual SLAM
- Place visual recognition
- Visual ego-localization *with Deep-Learning*

Visual ego-localization using SLAM

SLAM = Simultaneous Localization and Mapping

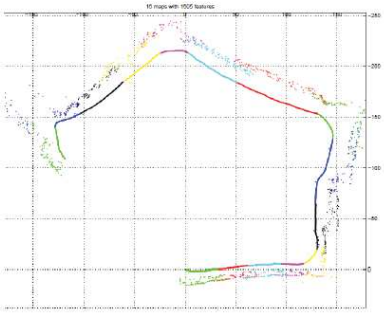
- Progressive creation of a « local » map within which the localization is computed
- Can be done with vision, but also with other sensor (in particular laser/LIDAR)



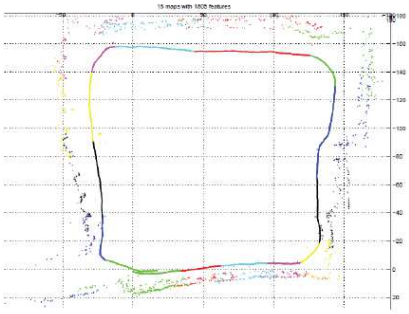
Visual Odometry vs Visual SLAM

- **Visual Odometry**
  - Focus on incremental estimation/**local consistency**
- **Visual SLAM**: Simultaneous Localization And Mapping
  - Focus on **globally consistent** estimation
  - **Visual SLAM = visual odometry + loop detection + graph optimization**

- The choice between VO and V-SLAM depends on the **tradeoff between performance and consistency**, and simplicity in implementation.
- VO **trades off consistency for real-time performance**, without the need to keep track of all the previous history of the camera.



Visual odometry

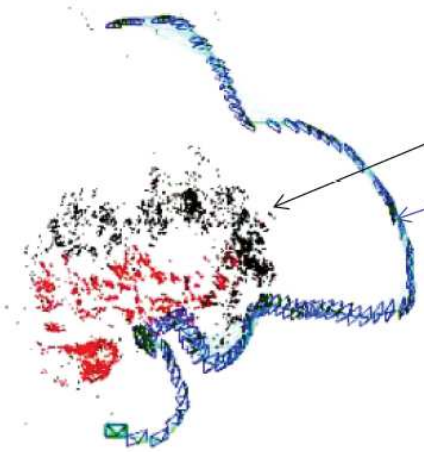


Visual SLAM

Image courtesy from [Clemente et al., RSS'07]

Davide Scaramuzza – University of Zurich – Robotics and Perception Group - rpg.ifi.uzh.ch

Feature-based Visual SLAM



States

$\mathbf{x}_{wj} \in \mathbb{R}^3$ Coordinates of point j

$\mathbf{T}_{iw} \in \mathrm{SE}(3)$ Pose of camera i

Measurements

$\mathbf{u}_{ij} = \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix}$ Observation of point j from camera i

Reprojection error — Projection Function

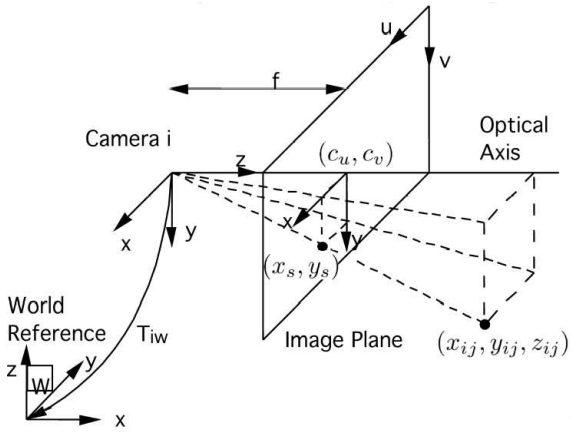
$$\mathbf{e}_{ij} = \mathbf{u}_{ij} - \pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj})$$

From Juan D. Tardos (Univ. Zaragoza) slides

Projection of point j on camera i

$\mathbf{T}_{iw} \in \mathrm{SE}(3)$ $\begin{cases} \mathbf{R}_{iw} \in \mathrm{SO}(3) & \text{Rotation matrix} \\ \mathbf{t}_{iw} \in \mathbb{R}^3 & \text{Translation vector} \end{cases}$ **Camera pose**

$\mathbf{x}_{ij} = \mathbf{R}_{iw}\mathbf{x}_{wj} + \mathbf{t}_{iw}$ Coordinates of point *j* w.r.t. camera *i*



focal length (mm)

$$x_s = f_i \frac{x_{ij}}{z_{ij}}$$

$$u = (s_u f_i)\frac{x_{ij}}{z_{ij}} + c_{i,u} = f_{i,u}\frac{x_{ij}}{z_{ij}} + c_{i,u}$$

horizontal focal length (pixels) — principal point

- In summary:

$$\pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj}) = \begin{bmatrix} f_{i,u}\frac{x_{ij}}{z_{ij}} + c_{i,u} \\ f_{i,v}\frac{y_{ij}}{z_{ij}} + c_{i,v} \end{bmatrix}$$

From Juan D. Tardos (Univ. Zaragoza) slides

Minimization of reprojection errors

Find the state values minimizing the reprojection errors:

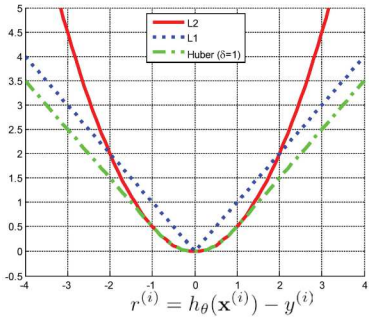
$$\mathbf{e}_{ij} = \mathbf{u}_{ij} - \pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj})$$

Bundle Adjustment

$$\{\mathbf{T}_{1w}..\mathbf{T}_{nw}, \mathbf{x}_{w1}..\mathbf{x}_{wm}\}^\star = \arg\min_{\mathbf{T},\mathbf{x}} \sum_{i,j} \rho_h(\mathbf{e}_{ij}^T \boldsymbol{\Sigma}_{ij}^{-1} \mathbf{e}_{ij})$$

$\Sigma_{ij} = \sigma_{ij}^2 \mathbf{I}_{2\times 2}$ std. dev. typically = 1 pixel * scale

where $\rho_h(\,)$ robust cost function (i.e. Huber cost) to downweight wrong matchings



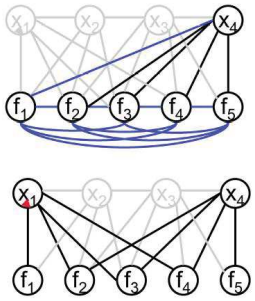
From Juan D. Tardos (Univ. Zaragoza) slides

Bundle Adjustment (BA)

$$\{\mathbf{T}_{1w}..\mathbf{T}_{nw}, \mathbf{x}_{w1}..\mathbf{x}_{wm}\}^{\star} = \arg\min_{\mathbf{T},\mathbf{x}} \sum_{i,j} \rho_h(\mathbf{e}_{ij}^T \boldsymbol{\Sigma}_{ij}^{-1} \mathbf{e}_{ij})$$

- The problem is sparse
  - Not all cameras see all points!
- But still not feasible in real time
  - example: 1k images and 100k points → 1s per LM iteration
- Local BA or sliding-window BA
- BA requires very good initial solutions

From Juan D. Tardos (Univ. Zaragoza) slides



- EKF approach
  - Only keeps the last pose
  - $O(n^2)$ with the number of features
  - Limited to 200-300 features in real-time
- Keyframe approach (PTAM)
  - Uses only a few keyframes for map estimation with non-linear optimization
  - Can handle thousands of points
  - Given the same computational effort is more precise than EKF-SLAM

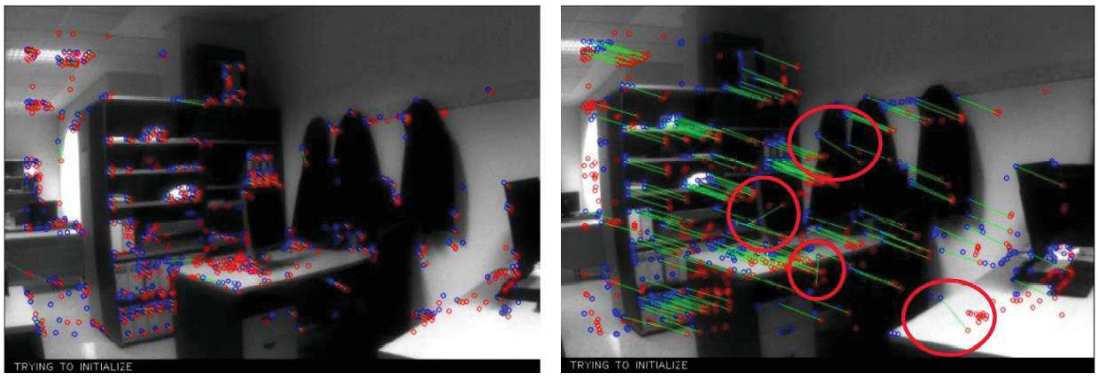
Choice of features (keypoints type)

Detector	Descriptor	Rotation Invariant	Automatic Scale	Accuracy	Relocation & Loops	Efficiency
Harris	Patch	No	No	++++	-	++++
Shi-Tomasi	Patch	No	No	++++	-	++++
SIFT	SIFT	Yes	Yes	++	++++	+
SURF	SURF	Yes	Yes	++	++++	++
FAST	BRIEF	No	No	+++	+++	++++
ORB	ORB	Yes	No	+++	+++	++++

From Juan D. Tardos (Univ. Zaragoza) slides

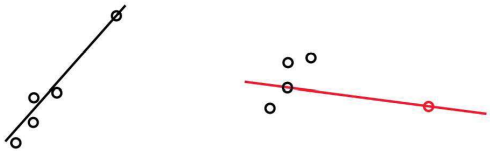
As for Visual Odometry,
usual SIFT or SURF are clearly not the best choice!

Feature matching



- Compare descriptors
- Spurious matchings

- Least-squares is very sensitive to spurious data
- A single spurious match may to ruin the estimation



➔ Search for consensus with a robust technique: RANSAC

From Juan D. Tardos (Univ. Zaragoza) slides

RANSAC

= RANdom SAmple Consensus

- **General statistical method for robust estimation in presence of outliers**

- **Principle: iterate the steps below**
  - **randomly select a subset of points to estimate a model**
  - **compute the # of other points compatible with the model**
  - **If enough inliers, re-estimate model with all, and compute error**
  - **If lower than current-best, replace it with new model (& inliers)**

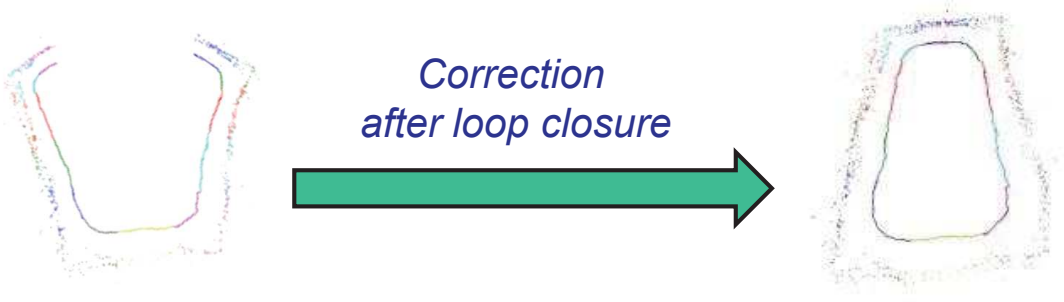
Loop closure

- Loop closing problem

  SLAM is working, and you come back to a previously mapped area

  ➢ Loop detection: to avoid map duplication

  ➢ Loop correction: to compensate the accumulated drift



Requires a place recognition technique

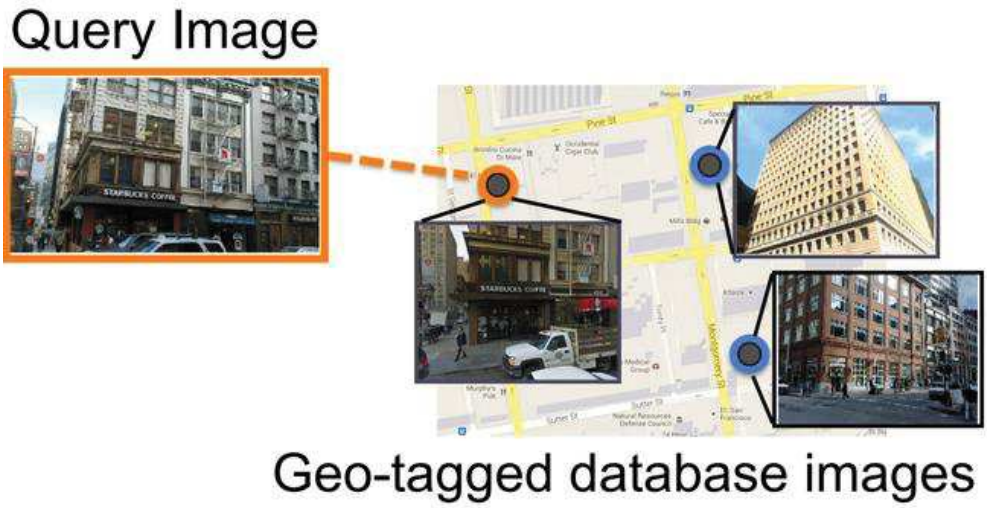
Place recognition also necessary for relocation
(« kidnapped robot » problem, …)

Outline

- **Definition, motivations, principle & methods overview**
- **Visual Odometry**
- **Tracked Features: visual keypoints**
- **Visual SLAM**
- **Place visual recognition**
- **Visual ego-localization *with Deep-Learning***

Visual place recognition

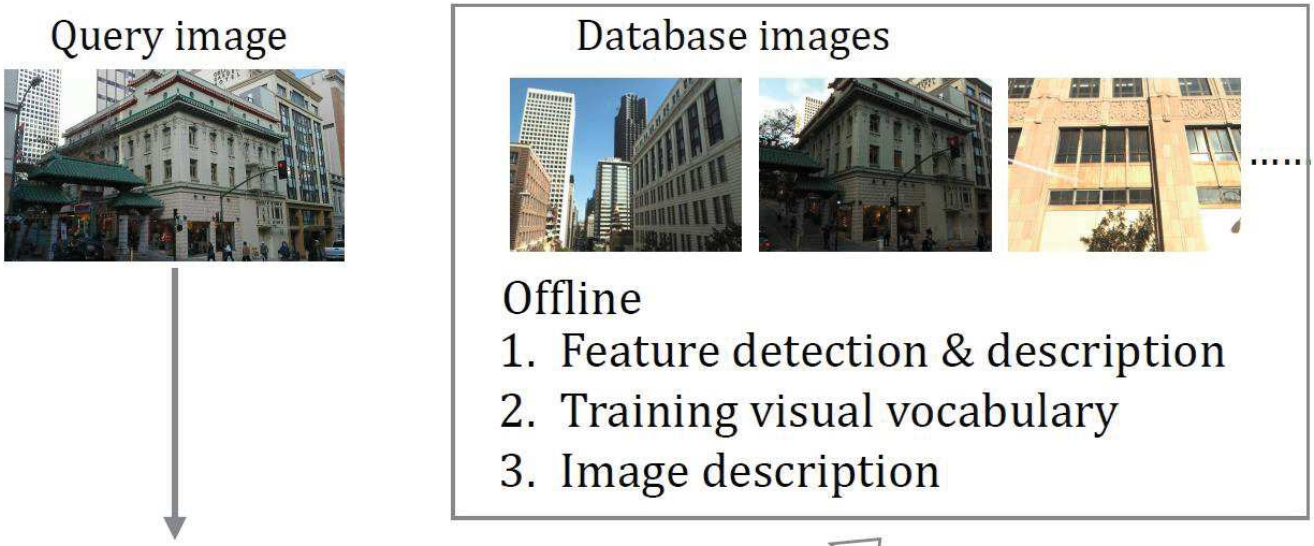
Coarse visual localization



From Akihiro Torii (Tokyo Tech) slides

➔ **This is an Image Retrieval particular case**

Image Retrieval with BoVW



From Akihiro Torii (Tokyo Tech) slides

Bag-of-Visual-Words (BoVW)

Inspired from text analysis in which a piece of text is represented by a sparse vector of the number of occurrences of each word of a dictionary

Adapted to images using *keypoints descriptors* as a representation of image content:

- **descriptor vectors are quantized (usually by K-means partitioning) into a codebook of « visual words**
- **An (sub-)image is represented by an histogram of codebook occurences**

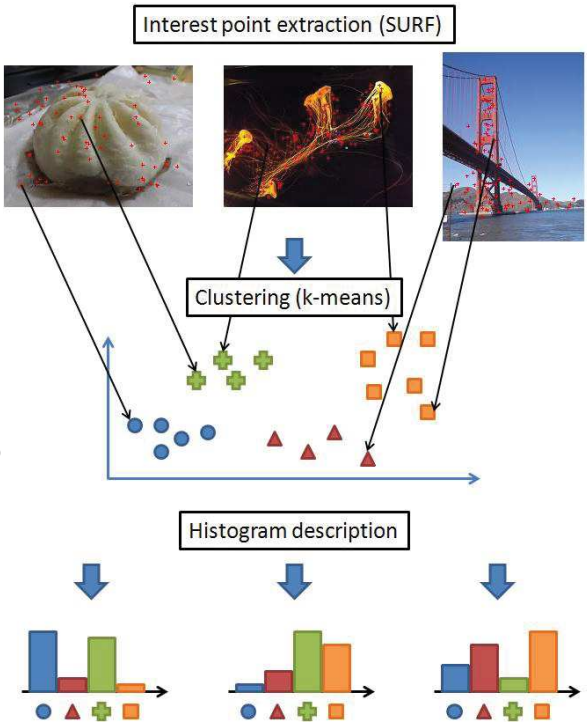
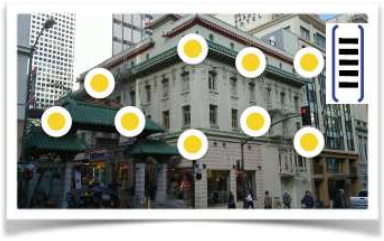


Image description



Quantization (aggregation) → Bag of visual words (VLAD, FV)

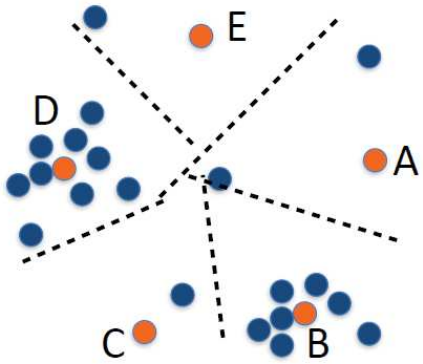
1. Feature detection & description, e.g. SIFT
2. Represent the set of features
   - as a sparse histogram (BoVW)
   - as an aggregated vectors (VLAD, FV)

From Akihiro Torii (Tokyo Tech) slides

Building visual « vocabulary » (offline)

Training data

Detect & describe features (e.g. SIFT)



Find representative features
(e.g. approximate k-means)

Visual words (centroids)

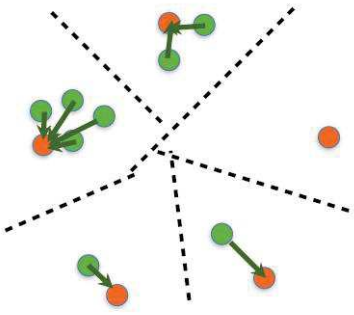
- Feature (128D)
- Centroid (128D)

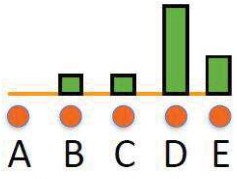
From Akihiro Torii (Tokyo Tech) slides

Computing BoVW sparse histogram for an image

For each image,

1. we assign features to the visual word closest in the feature space.
2. we build a histogram by voting.





Histogram

Visual words (centroids)

tf-idf weighted and L2 normalized

Design of the weights directly impacts the retrieval results!

See also [Jegou-CVPR09, Zheng-CVPR13]

- Feature (128D)
- Centroid (128D)

From Akihiro Torii (Tokyo Tech) slides

TF-IDF

= Term Frequency-Inverse Document Frequency

For a term i in document j:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

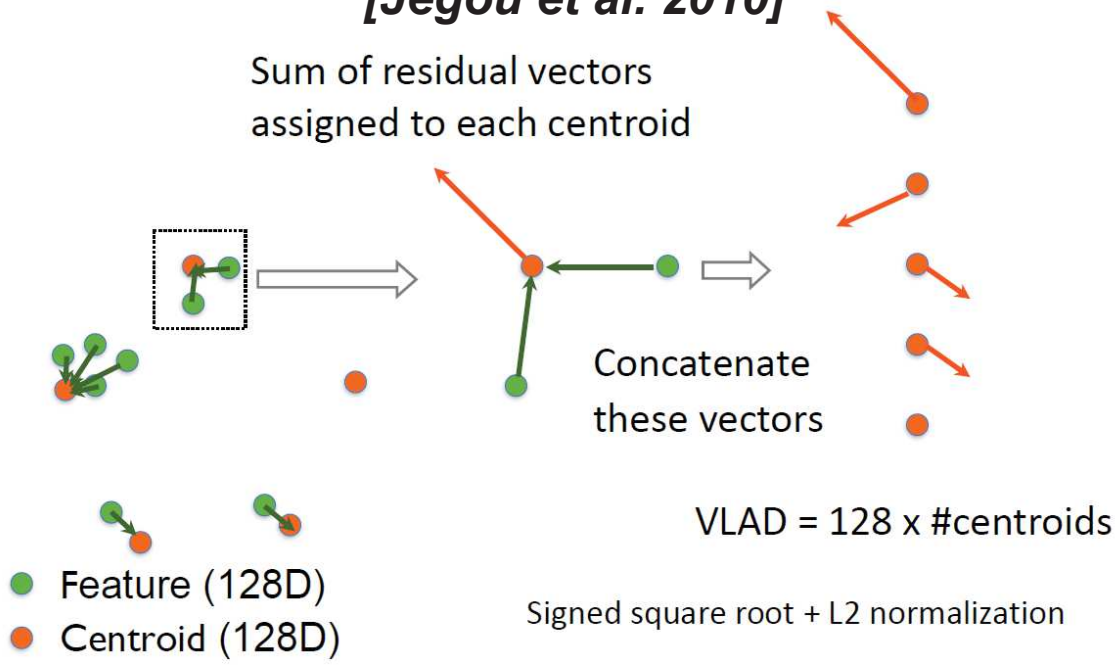
$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Giving larger weights to uncommon words

Alternative to BoVW: VLAD

= Vector of Locally Aggregated Descriptors

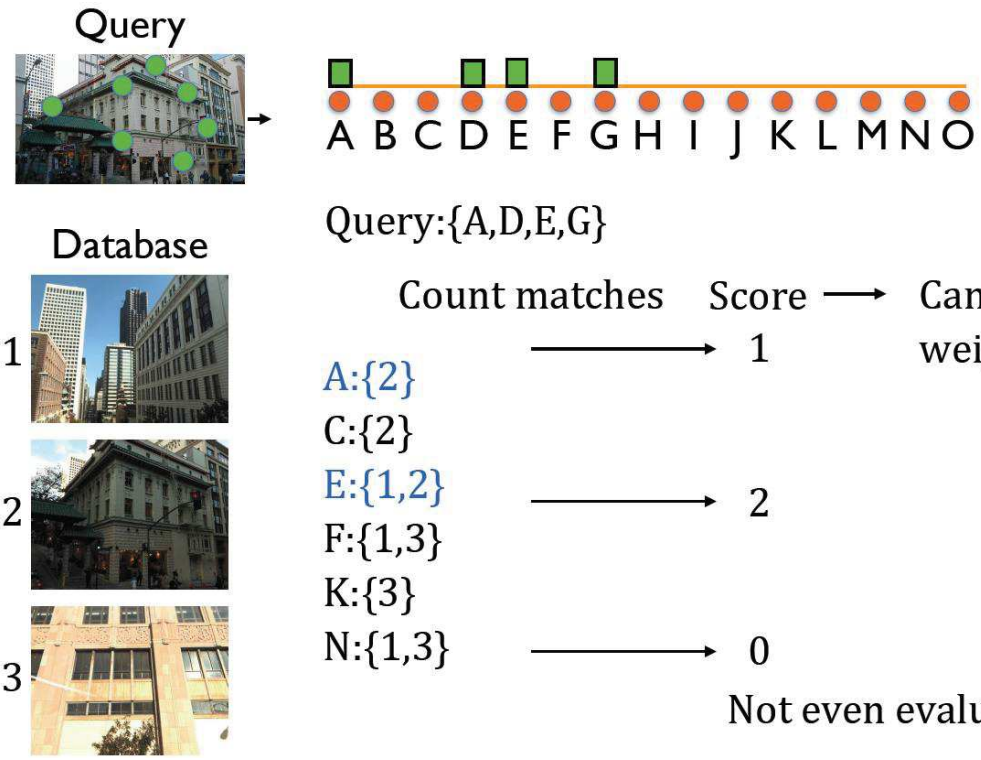
[Jégou et al. 2010]



From Akihiro Torii (Tokyo Tech) slides

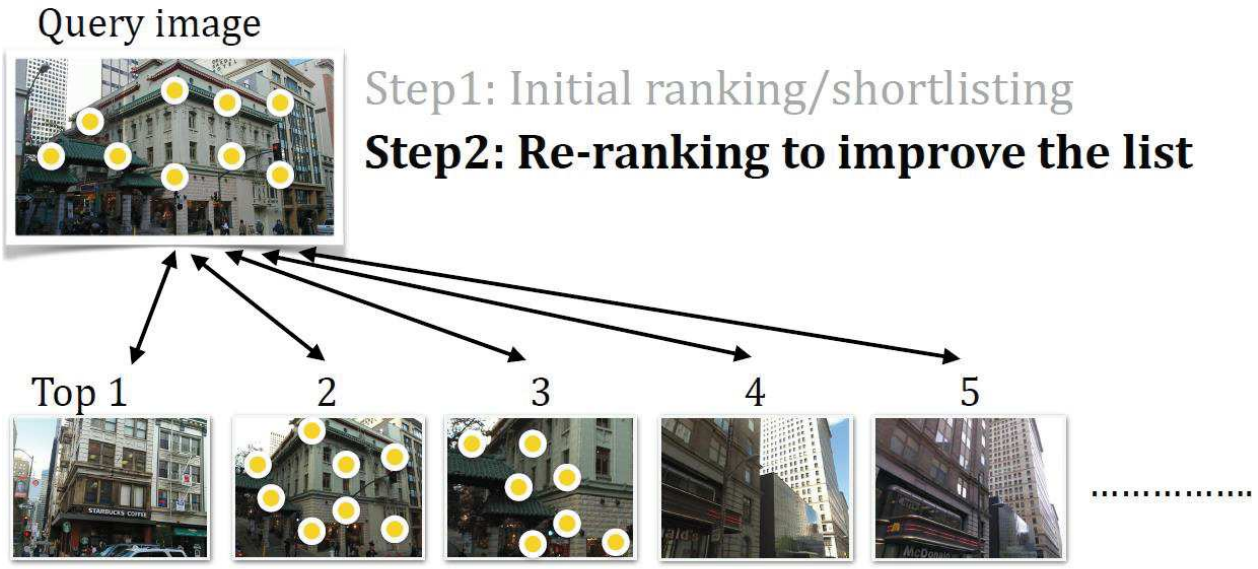
Use a sum of residuals instead of a TF-IDF histogram

Retrieval from query: inverted file indexing



From Akihiro Torii (Tokyo Tech) slides

Re-ranking by geometric verification

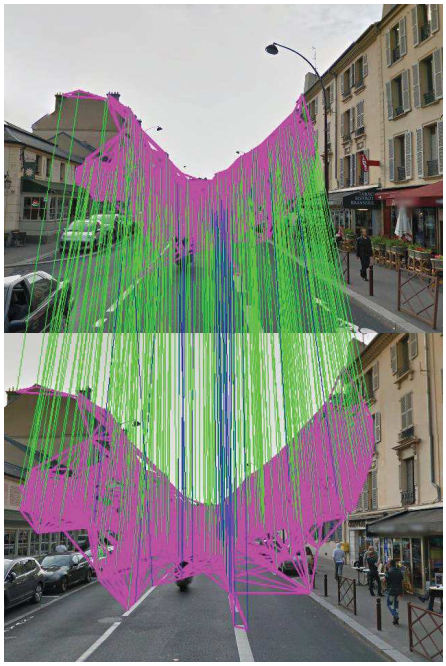


- Generate tentative matches of features
- Verify the tentative matches by fitting geometric transformations (affine, homography, ...), i.e. RANSAC
- Re-rank the shortlisted images by the number of verified matches

From Akihiro Torii (Tokyo Tech) slides

Visual fine ego-localization using keypoints

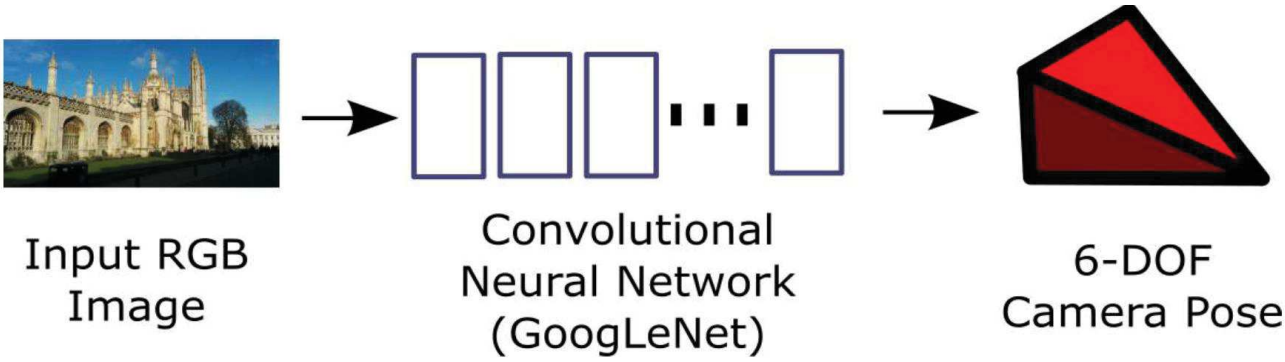
- Estimation of translation+rotation by multiple matches of keypoint descriptors (SIFT, SURF, ORB, vLAD, …) between query and match
- Requires elimination of outliers by RANSAC



Outline

- Definition, motivations, principle & methods overview
- Visual Odometry
- Tracked Features: visual keypoints
- Visual SLAM
- Place visual recognition
- Visual ego-localization *with Deep-Learning*

PoseNet: 6-DoF camera pose regression with Deep-Learning

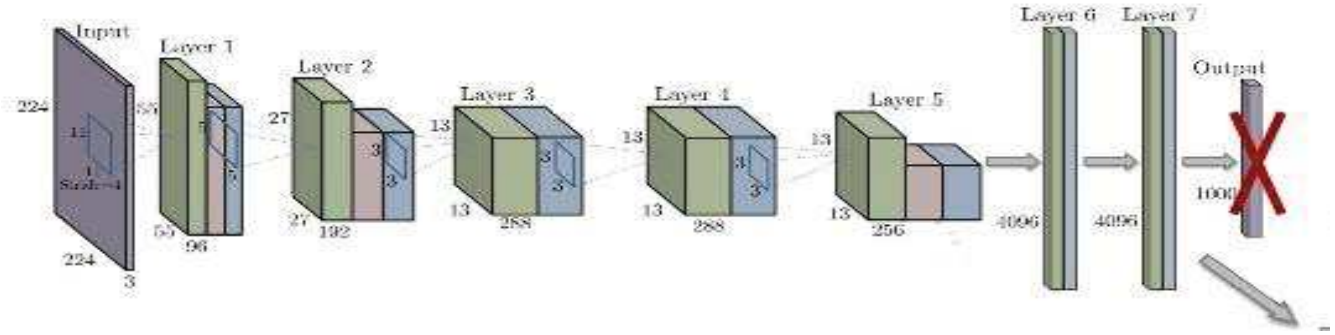


Trained with a naïve end-to-end loss function to regress camera position, **x**, and orientation, **q**

$$\text{loss}(I) = \|x - \hat{x}\|_2 + \beta \left\| q - \frac{\hat{q}}{\|\hat{q}\|} \right\|_2$$

*[**A. Kendall, M. Grimes & R. Cipolla,** "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization« , ICCV'2015, pp. 2938-2946]*

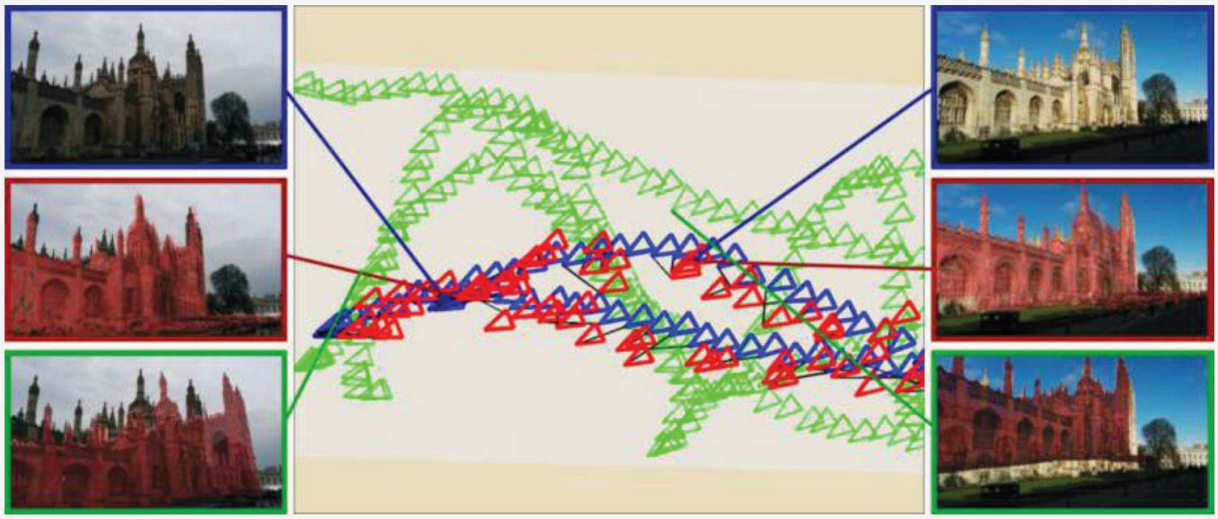
PoseNet applies Transfer learning for a task totally different from classification!



By <u>removing *last layer(s)*</u> (those for classification) of a convNet trained on ImageNet, one obtains a <u>*transformation of any input image into a semi-abstract representation,*</u> which can be used for learning SOMETHING ELSE (« <u>transfer learning</u> ») by creating new convNet output and perform <u>learning of new output layers + fine-tuning of re-used layers</u>

PoseNet training data and test results

training data in green, test data in blue, PoseNet results in red



Alex Kendall, Matthew Grimes and Roberto Cipolla. **PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization.** ICCV, 2015.

PoseNet results on other tests

King's College *Street* *Old Hospital* *Shop Façade* *St Mary's Church*

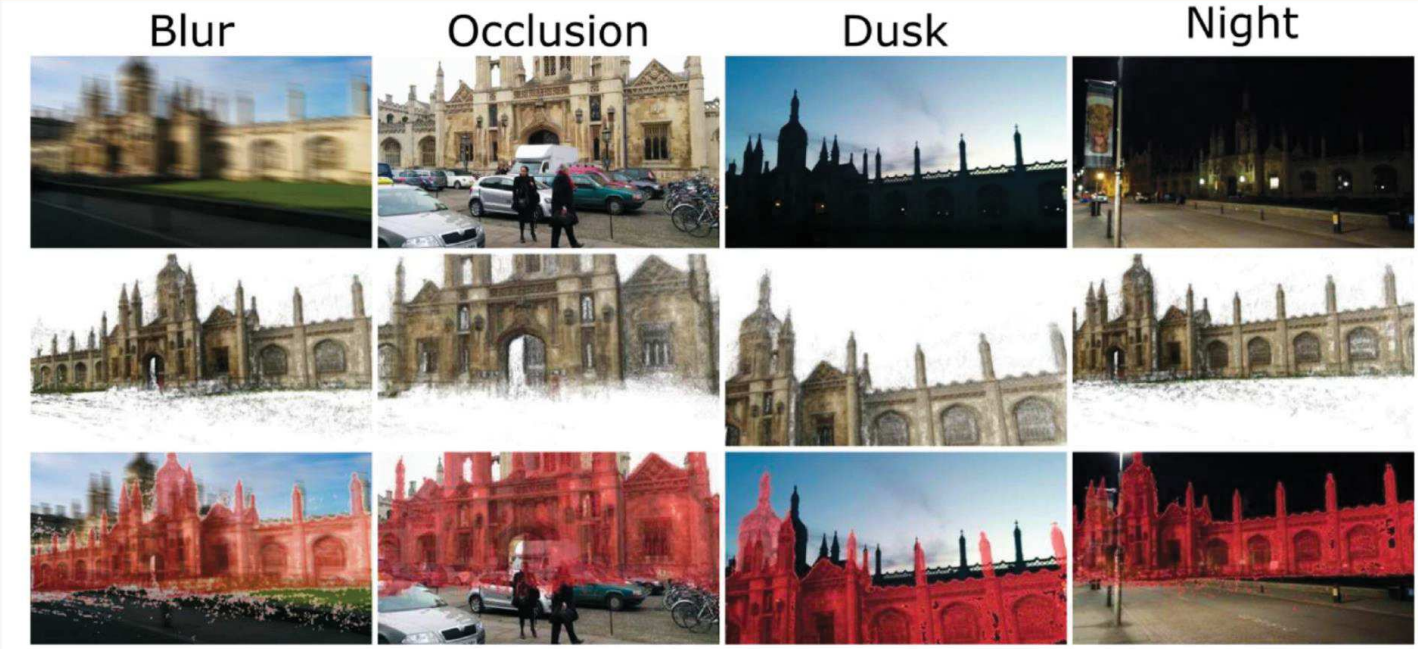
Figure 4: **Map of dataset** showing training frames (green), testing frames (blue) and their predicted camera pose (red). The testing sequences are distinct trajectories from the training sequences and each scene covers a very large spatial extent.

PoseNet results summary

Scene	# Frames Train	# Frames Test	Spatial Extent (m)	SCoRe Forest (Uses RGB-D)	Dist. to Conv. Nearest Neighbour	PoseNet	Dense PoseNet
King's College	1220	343	140 x 40m	N/A	3.34m, 2.96°	1.92m, 2.70°	1.66m, 2.43°
Street	3015	2923	500 x 100m	N/A	1.95m, 4.51°	3.67m, 3.25°	2.96m, 3.00°
Old Hospital	895	182	50 x 40m	N/A	5.38m, 4.51°	2.31m, 2.69°	2.62m, 2.45°
Shop Façade	231	103	35 x 25m	N/A	2.10m, 5.20°	1.46m, 4.04°	1.41m, 3.59°
St Mary's Church	1487	530	80 x 60m	N/A	4.48m, 5.65°	2.65m, 4.24°	2.45m, 3.98°
Chess	4000	2000	3 x 2 x 1m	0.03m, 0.66°	0.41m, 5.60°	0.32m, 4.06°	0.32m, 3.30°
Fire	2000	2000	2.5 x 1 x 1m	0.05m, 1.50°	0.54m, 7.77°	0.47m, 7.33°	0.47m, 7.02°
Heads	1000	1000	2 x 0.5 x 1m	0.06m, 5.50°	0.28m, 7.00°	0.29m, 6.00°	0.30m, 6.09°
Office	6000	4000	2.5 x 2 x 1.5m	0.04m, 0.78°	0.49m, 6.02°	0.48m, 3.84°	0.48m, 3.62°
Pumpkin	4000	2000	2.5 x 2 x 1m	0.04m, 0.68°	0.58m, 6.08°	0.47m, 4.21°	0.49m, 4.06°
Red Kitchen	7000	5000	4 x 3 x 1.5m	0.04m, 0.76°	0.58m, 5.65°	0.59m, 4.32°	0.58m, 4.17°
Stairs	2000	1000	2.5 x 2 x 1.5m	0.32m, 1.32°	0.56m, 7.71°	0.47m, 6.93°	0.48m, 6.54°

PoseNet robustness

Tolerance to environment, unknown intrinsics, weather, etc.



Alex Kendall, Matthew Grimes and Roberto Cipolla. **PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization.** ICCV, 2015.

PoseNet summary: robust to scene change + very fast

✓Robust to lighting, weather, dynamic objects

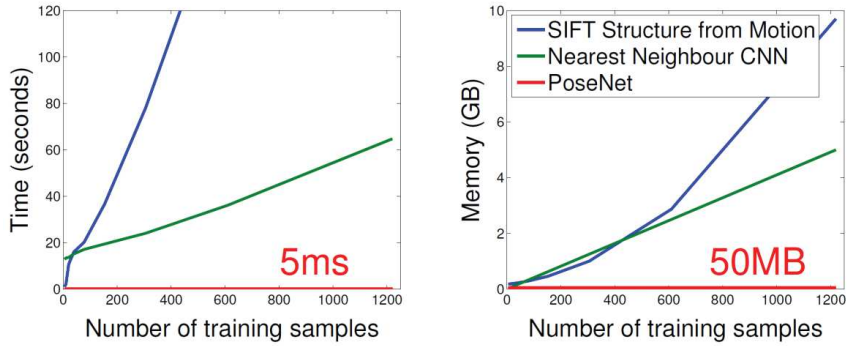
✓Fast inference, <2ms per image on Titan GPU

✓Scale not dependent on number of training images

✗Coarse accuracy

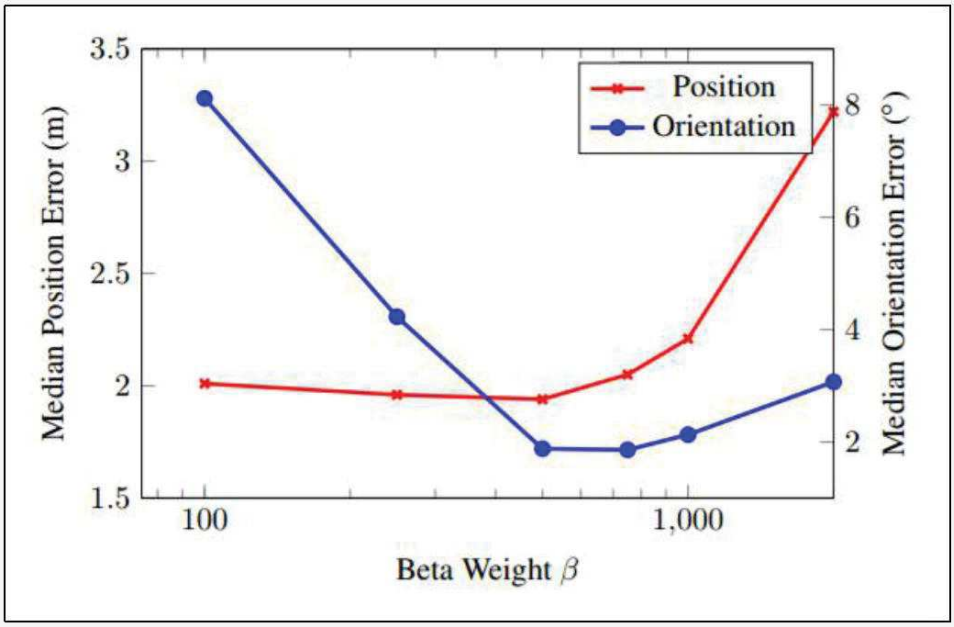
✗Difficult to learn both position vs orientation

Alex Kendall, Matthew Grimes and Roberto Cipolla. **PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization.** ICCV, 2015.

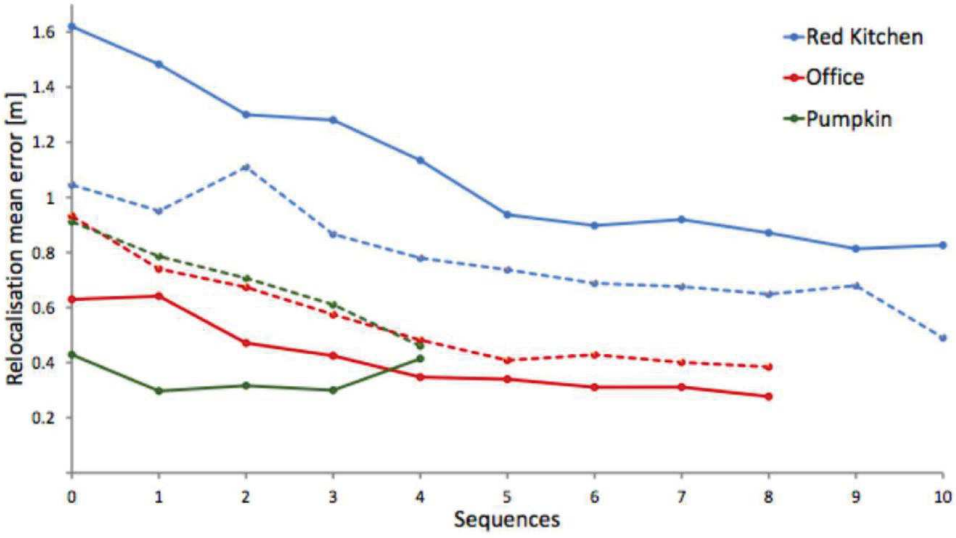


PoseNet: importance of relative weighting of position-orientation errors

$$\text{loss}(I) = \|x - \hat{x}\|_2 + \beta \left\| q - \frac{\hat{q}}{\|\hat{q}\|} \right\|_2$$



PoseNet performance improves with more data

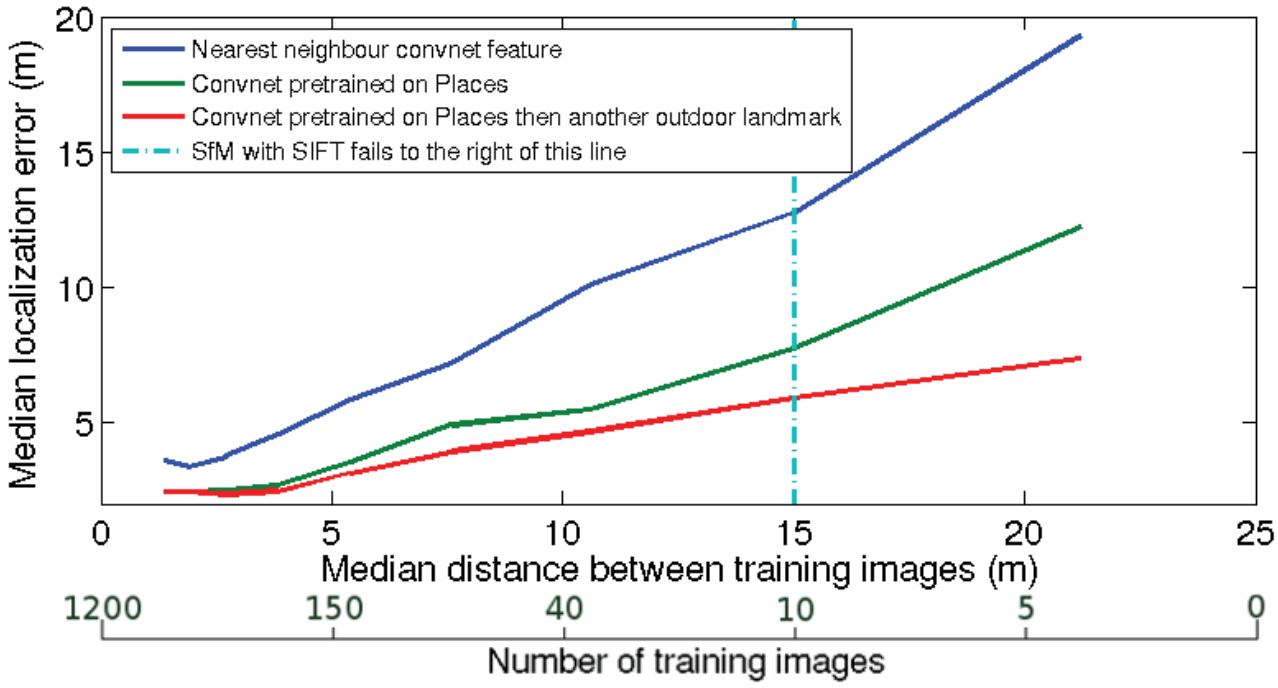


Scales very well:

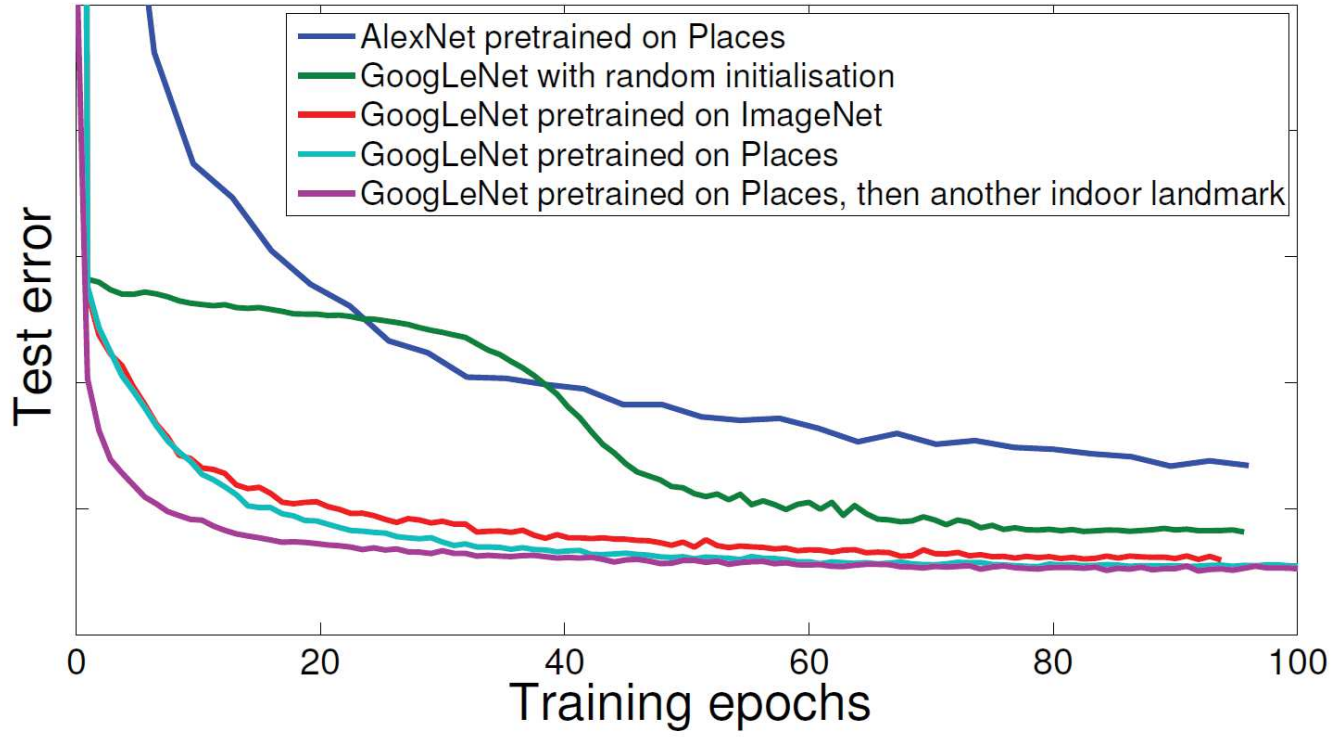
- Constant inference time (single forward pass of the network)
- Constant memory (~5 MB of neural network weights)

Contreras, Luis, and Walterio Mayol-Cuevas. **Towards CNN Map Compression for camera relocalisation.** arXiv:1703.00845, 2017.

PoseNet: graceful degradation with increased spacing of training images



PoseNet: importance of transfer learning



PoseNet *vs.* traditional methods

Dataset	PoseNet with Geometry [1]	Active Search (SIFT + Geometry) [2]
King's College	0.88m, 1.04°	**0.42m, 0.55°**
Resolution	**256 x 256 px**	1920 × 1080 px
Inference Time	**2 ms**	78 ms

PoseNet less precise, but much faster and can work with much smaller images