Visual scene real-time analysis for Intelligent Vehicles: Visual ego-Localization

Pr. Fabien Moutarde  
Center for Robotics  
MINES ParisTech  
PSL Research University

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 and methods overview
- Visual Odometry
- Tracked Features: visual keypoints
- Visual SLAM
- Place visual recognition
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])
- 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 localization approaches

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

Outline

- Definition motivations and principle
- **Visual Odometry**
- Tracked Features: visual keypoints
- **Visual SLAM**
- Place visual recognition

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

VO 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)
VO: estimation of relative motion

\[ T_K = \arg \min_T \iint \rho \left[ l_k \left( \pi \left( T \cdot \pi^{-1}(u), d_u \right) \right) - l_{k-1}(u) \right] du \]

VO flow chart

1. Image sequence
2. Feature detection
3. Feature matching (tracking)
4. Motion estimation
   - 2D-2D
   - 3D-3D
   - 3D-2D
5. Local optimization

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

- Definition motivations and principle
- Visual Odometry
- Tracked Features: visual keypoints
- Visual SLAM
- Place visual recognition

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 $\text{intensity}_\text{of}(C)+\text{threshold}$, or
  - all darker than $\text{intensity}_\text{of}(C)+\text{threshold}$

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

Keypoints DESCRIPTORS for VO

BRIEF descriptor [Calonder et. al, ECCV 2010]
- Binary Robust Independent Elementary Features
- 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
- 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]
- Oriented FAST and Rotated BRIEF
- Alternative 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 and principle
- Visual Odometry
- Tracked Features: visual keypoints
- Visual SLAM
- Place visual recognition

Visual 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)
**VO 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.

**Feature-based Visual SLAM**

- **States**
  - $x_{wj} \in \mathbb{R}^3$: Coordinates of point $j$
  - $T_{iw} \in SE(3)$: Pose of camera $i$

- **Measurements**
  - $u_{ij} = \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix}$: Observation of point $j$ from camera $i$

**Reprojection error**
- $e_{ij} = u_{ij} - \pi_i(T_{iw}, x_{wj})$

**Projection Function**

*From Juan D. Tardos (Univ. Zaragoza) slides*
Projection of point $j$ on camera $i$

$$\mathbf{T}_{iw} \in \text{SE}(3) \quad \begin{cases} \mathbf{R}_{iw} \in \text{SO}(3) \\ \mathbf{t}_{iw} \in \mathbb{R}^3 \end{cases}$$

Camera pose

$$\mathbf{x}_{ij} = \mathbf{R}_{iw} \mathbf{x}_{wj} + \mathbf{t}_{iw}$$

Coordinates of point $j$ w.r.t. camera $i$

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

$$\{\mathbf{T}_{1w}, \ldots, \mathbf{T}_{nw}, \mathbf{x}_{w1}, \ldots, \mathbf{x}_{wm}\}^* = \arg \min_{\mathbf{T}, \mathbf{x}} \sum_{i,j} \rho_h(\mathbf{e}_{ij}^T \Sigma_{ij}^{-1} \mathbf{e}_{ij})$$

$$\Sigma_{ij} = \sigma_{ij}^2 \mathbf{I}_{2 \times 2}$$

std. dev. typically = 1 pixel * scale

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

From Juan D. Tardos (Univ. Zaragoza) slides
Bundle Adjustment (BA)

\[
\{T_{1w}, T_{nw}, x_{w1}..x_{wm}\}^* = \arg \min_{T,x} \sum_{i,j} \rho_h(e_{ij}^T \Sigma_{ij}^{-1} e_{ij})
\]

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

- 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)

<table>
<thead>
<tr>
<th>Detector</th>
<th>Descriptor</th>
<th>Rotation Invariant</th>
<th>Automatic Scale</th>
<th>Accuracy</th>
<th>Relocation &amp; Loops</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harris</td>
<td>Patch</td>
<td>No</td>
<td>No</td>
<td>+++</td>
<td>-</td>
<td>+++</td>
</tr>
<tr>
<td>Shi-Tomasi</td>
<td>Patch</td>
<td>No</td>
<td>No</td>
<td>+++</td>
<td>-</td>
<td>+++</td>
</tr>
<tr>
<td>SIFT</td>
<td>SIFT</td>
<td>Yes</td>
<td>Yes</td>
<td>++</td>
<td>+++</td>
<td>+</td>
</tr>
<tr>
<td>SURF</td>
<td>SURF</td>
<td>Yes</td>
<td>Yes</td>
<td>++</td>
<td>+++</td>
<td>++</td>
</tr>
<tr>
<td>FAST</td>
<td>BRIEF</td>
<td>No</td>
<td>No</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
</tr>
<tr>
<td>ORB</td>
<td>ORB</td>
<td>Yes</td>
<td>No</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
</tr>
</tbody>
</table>

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 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

Correction after loop closure

Requires a place recognition technique

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

Outline

- Definition motivations and principle
- Visual Odometry
- Tracked Features: visual keypoints
- Visual SLAM
- Place visual recognition
Visual place recognition

Coarse visual localization

Query Image

Geo-tagged database images

From Akihiro Torii (Tokyo Tech) slides

→ This is an Image Retrieval particular case

Image Retrieval with BoVW

Query image

Database images

Offline
1. Feature detection & description
2. Training visual vocabulary
3. Image description

4. Feature detection & description
5. Image description
6. Initial ranking
7. Re-ranking with geometric verification

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 occurrences

---

Image description

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)

Detect & describe features (e.g. SIFT)

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

Visual words (centroids)

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

Computing BoVW sparse histogram for an image

Design of the weights directly impacts the retrieval results!

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]

Sum of residual vectors assigned to each centroid

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

VLAD = 128 x #centroids

Signed square root + L2 normalization

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

From Akihiro Torii (Tokyo Tech) slides
Visual fine 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