Robotic 3D Vision

Lecture 5: Visual Odometry 2 – Indirect Methods cont.

Prof. Dr. Jörg Stückler
Computer Vision Group, TU Munich
http://vision.in.tum.de

What We Will Cover Today

- Indirect visual odometry methods
  - 2D-to-3D motion estimation
  - 2D-to-3D visual odometry

  - 3D-to-3D motion estimation
  - 3D-to-3D visual odometry

- Properties of keypoint detection and matching
- Robustness and uncertainty propagation
- Probabilistic modelling

Recap: Special Euclidean Group SE(n)

- Euclidean transformation matrices have a special structure:

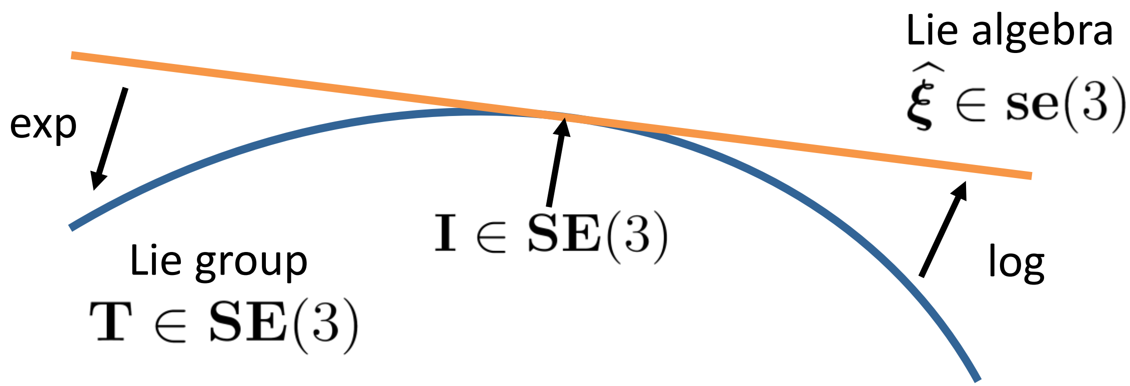
$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \in \mathbf{SE}(3) \subset \mathbb{R}^{4\times 4}$$

  - Translation $\mathbf{t}$ has 3 degrees of freedom
  - Rotation $\mathbf{R} \in \mathbf{SO}(3)$ has 3 degrees of freedom

- They also form a group which we denote as Special Euclidean Group $\mathbf{SE}(3)$. The group operator is matrix multiplication:

$$\begin{aligned} \cdot : \mathbf{SE}(3) \times \mathbf{SE}(3) &\to \mathbf{SE}(3) \\ \mathbf{T}_B^A \cdot \mathbf{T}_C^B &\mapsto \mathbf{T}_C^A \end{aligned}$$

Recap: Representing Motion using Lie Algebra se(3)



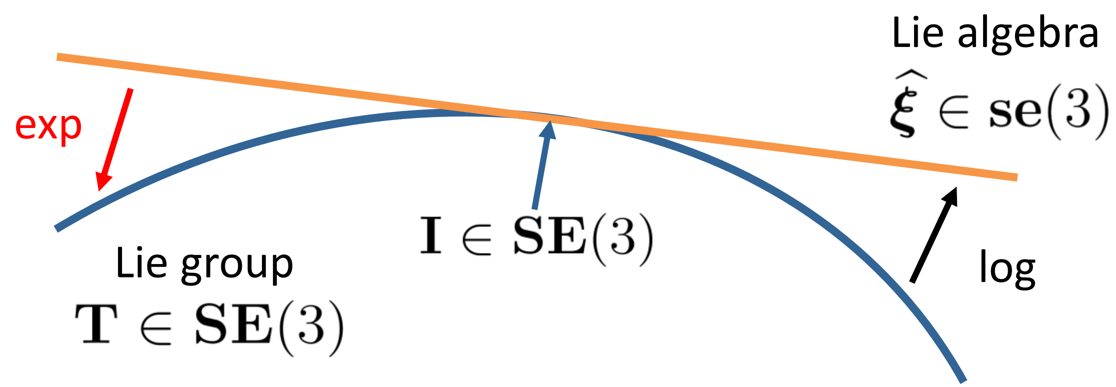
$$\boldsymbol{\xi} := \begin{pmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{pmatrix} \in \mathbb{R}^6$$

$$\boldsymbol{\omega} \in \mathbb{R}^3$$
$$\mathbf{v} \in \mathbb{R}^3$$

$$\widehat{\boldsymbol{\xi}} := \begin{pmatrix} \widehat{\boldsymbol{\omega}} & \mathbf{v} \\ \mathbf{0} & 0 \end{pmatrix} \in \mathbb{R}^{4 \times 4}$$

- $\mathbf{SE}(3)$ is a Lie group, i.e. a smooth manifold with compatible operator, inverse and neutral element
- Its Lie algebra $\mathbf{se}(3)$ provides an elegant way to parametrize poses for optimization
- Its elements $\widehat{\boldsymbol{\xi}} \in \mathbf{se}(3)$ form the tangent space of $\mathbf{SE}(3)$ at identity
- The $\mathbf{se}(3)$ elements can be interpreted as rotational and translational velocities (twists)

Recap: Exponential Map of SE(3)

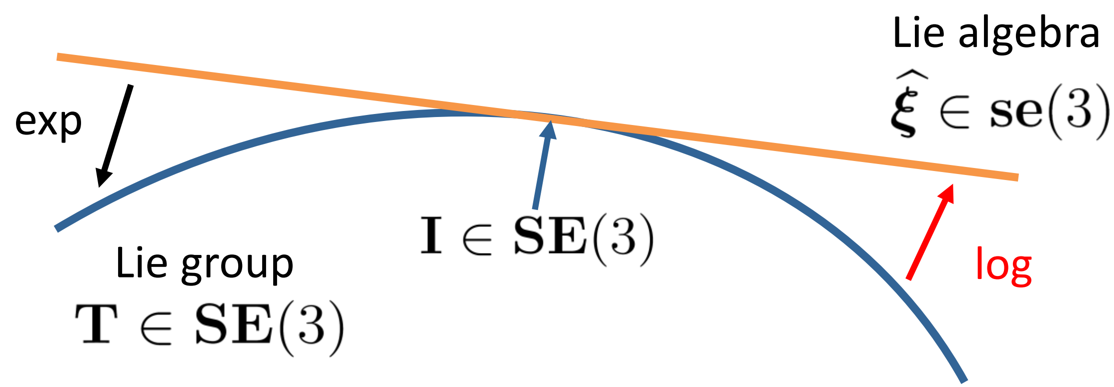


- The exponential map finds transformation matrices for twists:

$$\exp\left(\widehat{\boldsymbol{\xi}}\right) = \begin{pmatrix} \exp\left(\widehat{\boldsymbol{\omega}}\right) & \mathbf{A}\mathbf{v} \\ \mathbf{0} & 1 \end{pmatrix}$$

$$\exp\left(\widehat{\boldsymbol{\omega}}\right) = \mathbf{I} + \frac{\sin|\omega|}{|\omega|}\widehat{\boldsymbol{\omega}} + \frac{1-\cos|\omega|}{|\omega|^2}\widehat{\boldsymbol{\omega}}^2 \qquad \mathbf{A} = \mathbf{I} + \frac{1-\cos|\omega|}{|\omega|^2}\widehat{\boldsymbol{\omega}} + \frac{|\omega| - \sin|\omega|}{|\omega|^3}\widehat{\boldsymbol{\omega}}^2$$

Recap: Logarithm Map of SE(3)



- The logarithm map finds twists for transformation matrices:

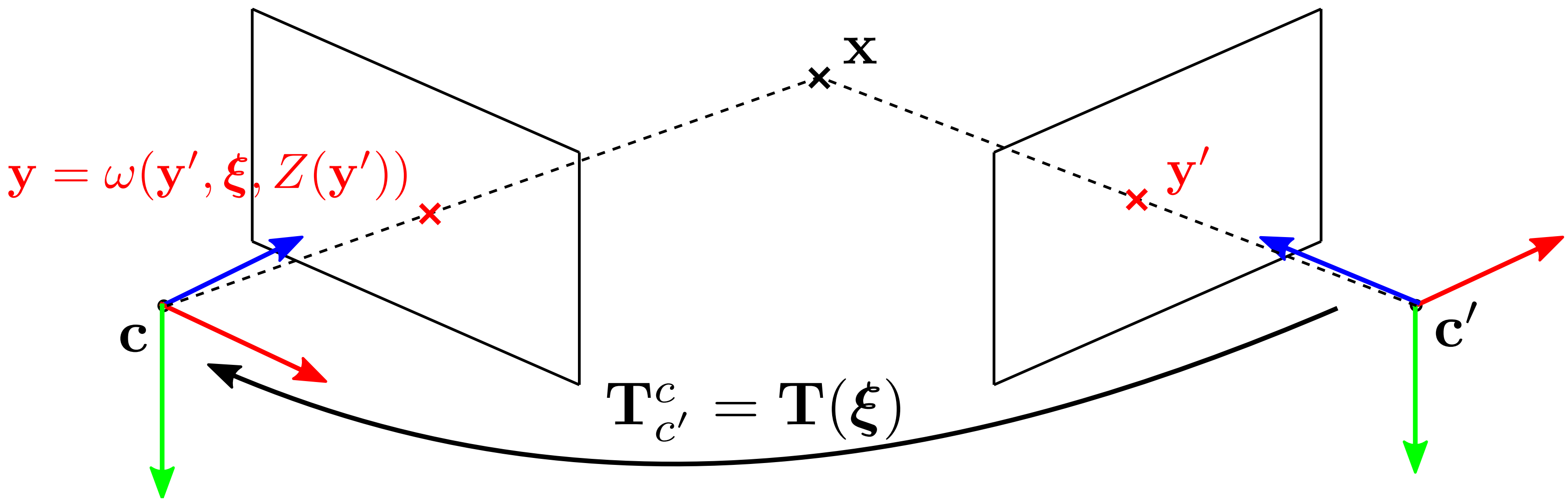
$$\log(\mathbf{T}) = \begin{pmatrix} \log(\mathbf{R}) & \mathbf{A}^{-1}\mathbf{t} \\ \mathbf{0} & 0 \end{pmatrix}$$

$$\log(\mathbf{R}) = \frac{|\omega|}{2\sin|\omega|}\left(\mathbf{R} - \mathbf{R}^T\right) \qquad |\omega| = \cos^{-1}\left(\frac{\mathrm{tr}(\mathbf{R}) - 1}{2}\right)$$

Recap: Some Notation for Twist Coordinates

- Let's define the following notation:
  - Inv. of hat operator: $\begin{pmatrix} 0 & -\omega_3 & \omega_2 & v_1 \\ \omega_3 & 0 & -\omega_1 & v_2 \\ -\omega_2 & \omega_1 & 0 & v_3 \\ 0 & 0 & 0 & 0 \end{pmatrix}^\vee = \left(\omega_1 \ \omega_2 \ \omega_3 \ v_1 \ v_2 \ v_3\right)^\top$
  - Conversion: $\boldsymbol{\xi}\left(\mathbf{T}\right) = \left(\log(\mathbf{T})\right)^\vee \qquad \mathbf{T}\left(\boldsymbol{\xi}\right) = \exp(\widehat{\boldsymbol{\xi}})$
  - Pose inversion: $\boldsymbol{\xi}^{-1} = \log(\mathbf{T}\left(\boldsymbol{\xi}\right)^{-1})^\vee = -\boldsymbol{\xi}$
  - Pose concatenation: $\boldsymbol{\xi}_1 \oplus \boldsymbol{\xi}_2 = \left(\log\left(\mathbf{T}\left(\boldsymbol{\xi}_2\right)\mathbf{T}\left(\boldsymbol{\xi}_1\right)\right)\right)^\vee$
  - Pose difference: $\boldsymbol{\xi}_1 \ominus \boldsymbol{\xi}_2 = \left(\log\left(\mathbf{T}\left(\boldsymbol{\xi}_2\right)^{-1}\mathbf{T}\left(\boldsymbol{\xi}_1\right)\right)\right)^\vee$

Recap: Warping Function



- Normalized image coordinates:

$$\omega(\mathbf{y}', \boldsymbol{\xi}, Z(\mathbf{y}')) = \pi\left(\mathbf{T}(\boldsymbol{\xi}) \overline{Z(\mathbf{y}')\overline{\mathbf{y}}'}\right)$$

- Pixel coordinates:

$$\omega(\mathbf{y}'_p, \boldsymbol{\xi}, Z(\mathbf{y}'_p)) = \mathbf{C}\pi\left(\mathbf{T}(\boldsymbol{\xi}) \overline{Z(\mathbf{y}'_p)\mathbf{C}^{-1}\overline{\mathbf{y}}'_p}\right)$$

Recap: Sensors for Visual Odometry

- **Monocular cameras**
  - Pros: Low-power, light-weight, low-cost, simple to calibrate and use
  - Cons: requires motion parallax and texture, scale not observable

- **Stereo cameras**
  - Pros: depth without motion, less power than active structured light
  - Cons: requires texture, accuracy depends on baseline, resolution, synchronization and extrinsic calibration of the cameras

- **Active RGB-D sensors**
  - Pros: measures geometry directly (geometric alignment), similar to stereo processing
  - Cons: active sensing consumes power, blackbox depth estimation

Image source: IDS, PointGrey, ASUS

Recap: Definition of Visual Odometry

- Visual odometry is the process of estimating the egomotion of an object (robot) using visual inputs from cameras on the object (robot)

- **Inputs**: images at discrete time steps t,
  - Monocular case: Set of images $I_{0:t} = \{I_0, \ldots, I_t\}$
  - Stereo case: Left/right images $I^l_{0:t} = \{I^l_0, \ldots, I^l_t\}$ / $I^r_{0:t} = \{I^r_0, \ldots, I^r_t\}$
  - RGB-D case: Color/depth images $I_{0:t} = \{I_0, \ldots, I_t\}$ / $Z_{0:t} = \{Z_0, \ldots, Z_t\}$

- **Output**: Transformation estimate $\mathbf{T}_t \in \mathbf{SE(3)}$ of camera frame to world frame

- Camera pose integrated up from relative pose estimates
- Example: camera pose $\mathbf{T}_t = \mathbf{T}_0 \mathbf{T}^0_1 \cdots \mathbf{T}^{t-1}_t$ from frame-to-frame transformations $\mathbf{T}^{t-1}_t$

Recap: Indirect vs. Direct VO Methods

- Direct visual odometry methods formulate alignment objective in terms of pixel-wise error (e.g. photometric or geometric error)
  - Two-view case with known depth:

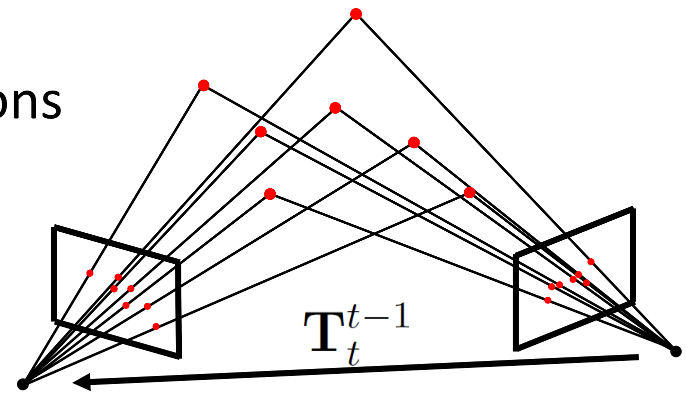
$$p(I_2 \mid I_1, Z_1, \boldsymbol{\xi}) \longrightarrow E(\boldsymbol{\xi}) = \int_{\mathbf{y} \in \boldsymbol{\Omega}} |I_1(\mathbf{y}) - I_2(\omega(\mathbf{y}, \boldsymbol{\xi}, Z_1(\mathbf{y})))| d\mathbf{y}$$

- Indirect visual odometry methods formulate alignment objective in terms of reprojection error of geometric primitives (e.g. points, lines)
  - Two-view case with known depth:

$$p(\mathcal{Y}_2 \mid \mathcal{Y}_1, Z_1, \boldsymbol{\xi}) \longrightarrow E(\boldsymbol{\xi}) = \sum_i |\mathbf{y}_{2,i} - \omega(\mathbf{y}_{1,i}, \boldsymbol{\xi}, Z_1(\mathbf{y}_{1,i}))|$$

  - $\mathcal{Y}_1, \mathcal{Y}_2$: sets of primitives (e.g. keypoints) in image 1 and 2

Recap: 2D-to-2D Motion Estimation



- Given corresponding image point observations

  $$\mathcal{Y}_t = \{\mathbf{y}_{t,1}, \ldots, \mathbf{y}_{t,N}\}$$
  $$\mathcal{Y}_{t-1} = \{\mathbf{y}_{t-1,1}, \ldots, \mathbf{y}_{t-1,N}\}$$

  of unknown 3D points $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
  (expressed in camera frame at time t)
  determine relative motion $\mathbf{T}_t^{t-1}$ between frames

- Naive try: minimize reprojection error using least squares

$$E(\mathbf{T}_t^{t-1}, \mathcal{X}) = \sum_{i=1}^{N} \left\|\overline{\mathbf{y}}_{t,i} - \pi\left(\overline{\mathbf{x}}_i\right)\right\|_2^2 + \left\|\overline{\mathbf{y}}_{t-1,i} - \pi\left(\mathbf{T}_t^{t-1}\overline{\mathbf{x}}_i\right)\right\|_2^2$$

- Convexity? Uniqueness (scale-ambiguity)?

- Alternative algebraic approach

Recap: Eight-Point Algorithm

- First proposed by Longuet and Higgins, Nature 1981
- Algorithm:
  1. Rewrite epipolar constraints as a linear system of equations

     $$\widetilde{\mathbf{y}}_i^\top \mathbf{E} \widetilde{\mathbf{y}}'_i = \mathbf{a}_i \mathbf{E}_s = 0 \longrightarrow \mathbf{A}\mathbf{E}_s = \mathbf{0} \qquad \mathbf{A} = \left(\mathbf{a}_1^\top, \ldots, \mathbf{a}_N^\top\right)^\top$$

     using Kronecker product $\mathbf{a}_i = \widetilde{\mathbf{y}}_i \otimes \widetilde{\mathbf{y}}'_i$ and $\mathbf{E}_s = (e_{11}, e_{12}, e_{13}, \ldots, e_{33})^\top$

  2. Apply singular value decomposition (SVD) on $\mathbf{A} = \mathbf{U}_\mathbf{A} \mathbf{S}_\mathbf{A} \mathbf{V}_\mathbf{A}^\top$ and unstack the 9th column of $\mathbf{V}_\mathbf{A}$ into $\widetilde{\mathbf{E}}$

  3. Project the approximate $\widetilde{\mathbf{E}}$ into the (normalized) essential space: Determine the SVD of $\widetilde{\mathbf{E}} = \mathbf{U}\,\mathrm{diag}\left(\sigma_1, \sigma_2, \sigma_3\right)\mathbf{V}^\top$ with $\mathbf{U}, \mathbf{V} \in \mathbf{SO}(3)$ and replace the singular values $\sigma_1 \geq \sigma_2 \geq \sigma_3$ with $1, 1, 0$ to find

     $$\mathbf{E} = \mathbf{U}\,\mathrm{diag}(1, 1, 0)\mathbf{V}^\top$$

Recap: Error Metric of the Eight-Point Algorithm

- What is the physical meaning of the error minimized by the eight-point algorithm?
- The eight-point algorithm finds E that minimizes

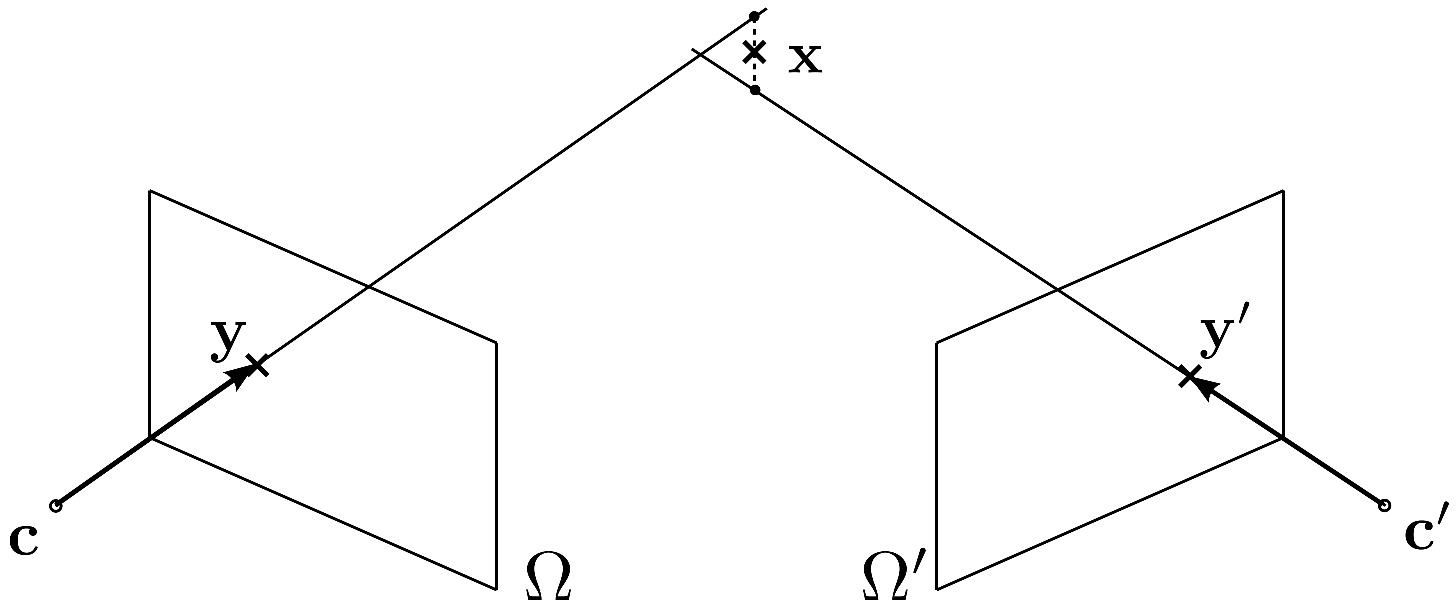
$$\operatorname{argmin}_{\mathbf{E}_s} \|\mathbf{A}\mathbf{E}_s\|_2^2$$

  subject to $\|\mathbf{E}_s\|_2^2 = 1$ through the SVD on A
- We find a least squares fit to the epipolar constraints
- A violated epipolar constraint

$$\widetilde{\mathbf{y}}^\top (\mathbf{t} \times \mathbf{R}\widetilde{\mathbf{y}}') = 0$$

  quantifies the volume spanned by y, t, and Ry'
- No clear interpretation in terms of distance or angular error

Triangulation



Triangulation

- Goal: Reconstruct 3D point $\widetilde{\mathbf{x}} = (x, y, z, w)^\top \in \mathbb{P}^3$ from 2D image observations $\{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ for known camera poses $\{\mathbf{T}_1, \ldots, \mathbf{T}_N\}$

- Linear solution: Find 3D point such that reprojections equal its projections

$$\mathbf{y}'_i = \pi(\mathbf{T}_i \widetilde{\mathbf{x}}) = \begin{pmatrix} \frac{r_{11}x + r_{12}y + r_{13}z + t_x w}{r_{31}x + r_{32}y + r_{33}z + t_z w} \\ \frac{r_{21}x + r_{22}y + r_{23}z + t_y w}{r_{31}x + r_{32}y + r_{33}z + t_z w} \end{pmatrix}$$

  - Each image provides one constraint $\mathbf{y}_i - \mathbf{y}'_i = 0$
  - Leads to system of linear equations $\mathbf{A}\widetilde{\mathbf{x}} = 0$, two approaches:
    - Set $w = 1$ and solve nonhomogeneous system
    - Find nullspace of $\mathbf{A}$ using SVD

- Non-linear solution: Minimize least squares reprojection error (more accurate)

$$\min_{\mathbf{x}} \left\{ \sum_{i=1}^{N} \|\mathbf{y}_i - \mathbf{y}'_i\|_2^2 \right\}$$

Relative Scale Recovery

- Problem: each subsequent frame-pair gives another solution for the reconstruction scale

- Approach:
  - Triangulate corresponding image points $\mathcal{Y}_{t-2}, \mathcal{Y}_{t-1}, \mathcal{Y}_t$ for current and last frame pair using the last and current recovered pose estimates and find their 3D positions

$$\mathcal{X}_{t-2,t-1}, \mathcal{X}_{t-1,t}$$

  - Rescale translation of current relative pose estimate to match the reconstruction scale with the distance ratio between corresponding 3D point pairs

$$r_{i,j} = \frac{\left\| \mathbf{x}_{t-2,t-1,i} - \mathbf{x}_{t-2,t-1,j} \right\|_2}{\left\| \mathbf{x}_{t-1,t,i} - \mathbf{x}_{t-1,t,j} \right\|_2}$$

  - Use mean or robust median over available pair ratios

Algorithm: 2D-to-2D Visual Odometry

Input: image sequence $I_{0:t}$, camera calibration

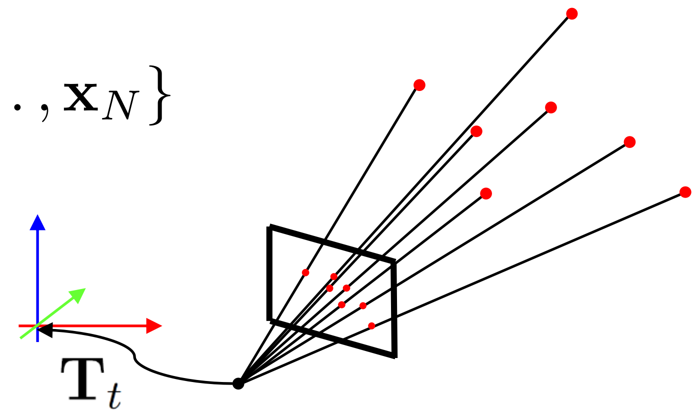
Output: aggregated camera poses $\mathbf{T}_{0:t}$

Algorithm:

For each current image I_k :

1. Extract and match keypoints between I_{k-1} and I_k
2. Compute relative pose $\mathbf{T}_k^{k-1}$ from essential matrix between I_k , I_{k-1}
3. Fine-tune pose estimate by minimizing reprojection error
4. Compute relative scale and rescale translation of $\mathbf{T}_k^{k-1}$
5. Aggregate camera pose by $\mathbf{T}_k = \mathbf{T}_{k-1}\mathbf{T}_k^{k-1}$

2D-to-3D Motion Estimation



- Given a local set of 3D points $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and corresponding image observations

  $$\mathcal{Y}_t = \{\mathbf{y}_{t,1}, \ldots, \mathbf{y}_{t,N}\}$$

  determine camera pose $\mathbf{T}_t$ within the local map
- Minimize least squares geometric reprojection error

$$E(\mathbf{T}_t) = \sum_{i=1}^{N} \left\| \mathbf{y}_{t,i} - \pi(\mathbf{T}_t^{-1} \mathbf{x}_i) \right\|_2^2$$

- A.k.a. Perspective-n-Points (PnP) problem, many approaches exist, f.e.
  - 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)

Direct Linear Transform for PnP

- Goal: determine projection matrix $\mathbf{P} = (\mathbf{R}\ \mathbf{t}) \in \mathbb{R}^{3\times4} = \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{pmatrix}$

- Each 2D-to-3D point correspondence
  3D: $\widetilde{\mathbf{x}}_i = (x_i, y_i, z_i, w_i)^\top \in \mathbb{P}^3$ 2D: $\widetilde{\mathbf{y}}_i = (x'_i, y'_i, w'_i)^\top \in \mathbb{P}^2$
  gives two constraints
  $$\begin{pmatrix} \mathbf{0} & -w'_i\widetilde{\mathbf{x}}_i^\top & y'_i\widetilde{\mathbf{x}}_i^\top \\ w'_i\widetilde{\mathbf{x}}_i^\top & \mathbf{0} & -x'_i\widetilde{\mathbf{x}}_i^\top \end{pmatrix} \begin{pmatrix} \mathbf{P}_1^\top \\ \mathbf{P}_2^\top \\ \mathbf{P}_3^\top \end{pmatrix} = \mathbf{0}$$
  through $\widetilde{\mathbf{y}}_i \times (\mathbf{P}\widetilde{\mathbf{x}}_i) = 0$

- Form linear system of equations $\mathbf{A}\mathbf{p} = \mathbf{0}$ with $\mathbf{p} := \begin{pmatrix} \mathbf{P}_1^\top \\ \mathbf{P}_2^\top \\ \mathbf{P}_3^\top \end{pmatrix} \in \mathbb{R}^9$
  from $N \geq 6$ correspondences

- Solve for $\mathbf{p}$: determine unit singular vector of $\mathbf{A}$ corresponding to its smallest singular value

Algorithm: 2D-to-3D Visual Odometry

Input: image sequence $I_{0:t}$, camera calibration

Output: aggregated camera poses $\mathbf{T}_{0:t}$

Algorithm:

Initialize:

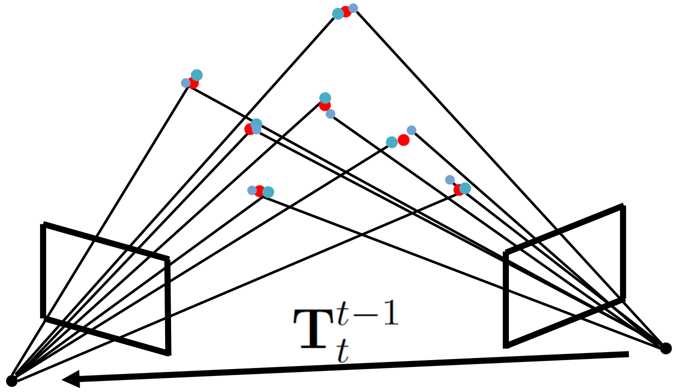
1. Extract and match keypoints between I_0 and I_1
2. Determine camera pose (essential matrix) and triangulate 3D keypoints X_1

For each current image I_k :

1. Extract and match keypoints between I_{k-1} and I_k
2. Compute camera pose $\mathbf{T}_k$ using PnP from 2D-to-3D matches
3. Triangulate all new keypoint matches between I_{k-1} and I_k and add them to the local map X_k

3D-to-3D Motion Estimation

- Given corresponding 3D points in two camera frames
  $\mathcal{X}_{t-1} = \{\mathbf{x}_{t-1,1}, \ldots, \mathbf{x}_{t-1,N}\}$
  $\mathcal{X}_t = \{\mathbf{x}_{t,1}, \ldots, \mathbf{x}_{t,N}\}$
  determine relative camera pose $\mathbf{T}_t^{t-1}$



- Idea: determine rigid transformation that aligns the 3D points
- Geometric least squares error: $E\left(\mathbf{T}_t^{t-1}\right) = \sum_{i=1}^{N} \left\| \overline{\mathbf{x}}_{t-1,i} - \mathbf{T}_t^{t-1} \overline{\mathbf{x}}_{t,i} \right\|_2^2$
- Closed-form solutions available, f.e. Arun et al., 1987
- Applicable f.e. for calibrated stereo cameras (triangulation of 3D points) or RGB-D cameras (measured depth)

3D Rigid-Body Motion from 3D-to-3D Matches

- Arun et al., Least-squares fitting of two 3-d point sets, IEEE PAMI, 1987
- Corresponding 3D points, $N \geq 3$

$$\mathcal{X}_{t-1} = \{\mathbf{x}_{t-1,1}, \ldots, \mathbf{x}_{t-1,N}\} \qquad \mathcal{X}_t = \{\mathbf{x}_{t,1}, \ldots, \mathbf{x}_{t,N}\}$$

- Determine means of 3D point sets

$$\boldsymbol{\mu}_{t-1} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_{t-1,i} \qquad \boldsymbol{\mu}_t = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_{t,i}$$

- Determine rotation from

$$\mathbf{A} = \sum_{i=1}^{N} \left(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1}\right)\left(\mathbf{x}_t - \boldsymbol{\mu}_t\right)^\top \qquad \mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^\top \qquad \mathbf{R}_{t-1}^t = \mathbf{V}\mathbf{U}^\top$$

- Determine translation as $\mathbf{t}_{t-1}^t = \boldsymbol{\mu}_t - \mathbf{R}_{t-1}^t \boldsymbol{\mu}_{t-1}$

Algorithm: Stereo 3D-to-3D Visual Odometry

Input: stereo image sequence $I^l_{0:t}, I^r_{0:t}$, camera calibration (including known pose between stereo cameras)

Output: aggregated camera poses $\mathbf{T}_{0:t}$

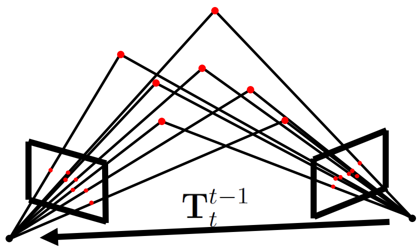
Algorithm:

For each current stereo image I^l_k, I^r_k:

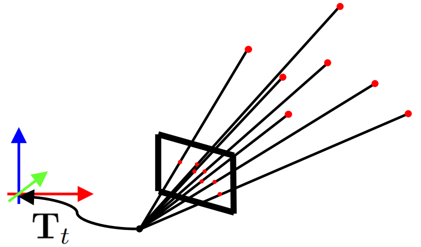
1. Extract and match keypoints between I^l_k and I^l_{k-1}
2. Triangulate 3D points X_k between I^l_k and I^r_k
3. Compute camera pose $\mathbf{T}^{k-1}_k$ from 3D-to-3D point matches X_k to X_{k-1}
4. Aggregate camera pose by $\mathbf{T}_k = \mathbf{T}_{k-1}\mathbf{T}^{k-1}_k$

Motion Estimation from Point Correspondences

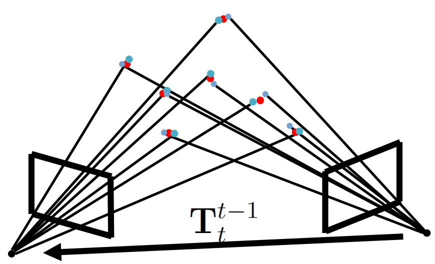
- **2D-to-2D**
  - Reproj. error:
    $$E\left(\mathbf{T}_t^{t-1}, X\right) = \sum_{i=1}^{N} \left\|\overline{\mathbf{y}}_{t,i} - \pi\left(\overline{\mathbf{x}}_i\right)\right\|_2^2 + \left\|\overline{\mathbf{y}}_{t-1,i} - \pi\left(\mathbf{T}_t^{t-1}\overline{\mathbf{x}}_i\right)\right\|_2^2$$
  - Linear algorithm: 8-point



- **2D-to-3D**
  - Reprojection error: $E(\mathbf{T}_t) = \sum_{i=1}^{N} \left\|\mathbf{y}_{t,i} - \pi(\mathbf{T}_t\overline{\mathbf{x}}_i)\right\|_2^2$
  - Linear algorithm: DLT PnP



- **3D-to-3D**
  - Reprojection error: $E\left(\mathbf{T}_t^{t-1}\right) = \sum_{i=1}^{N} \left\|\overline{\mathbf{x}}_{t-1,i} - \mathbf{T}_t^{t-1}\overline{\mathbf{x}}_{t,i}\right\|_2^2$
  - Linear algorithm: Arun‘s method



Further Considerations

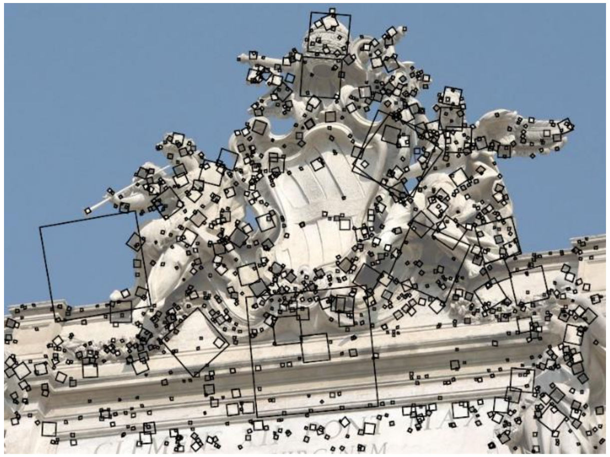
- How to detect keypoints?
- How to match keypoints?
- How to cope with outliers in keypoint matches?
- When to create new 3D keypoints ? Which keypoints to use?
- How to cope with noisy observations?
- 2D-to-2D, 2D-to-3D or 3D-to-3D?
- Optimize over more than two frames?
- …

Keypoint Detection

- Desirable properties of keypoint detectors for visual odometry:
  - High repeatability
  - Localization accuracy
  - Robustness
  - Invariance
  - Computational efficiency

Harris Corners

Image source: Svetlana Lazebnik



DoG (SIFT) Blobs

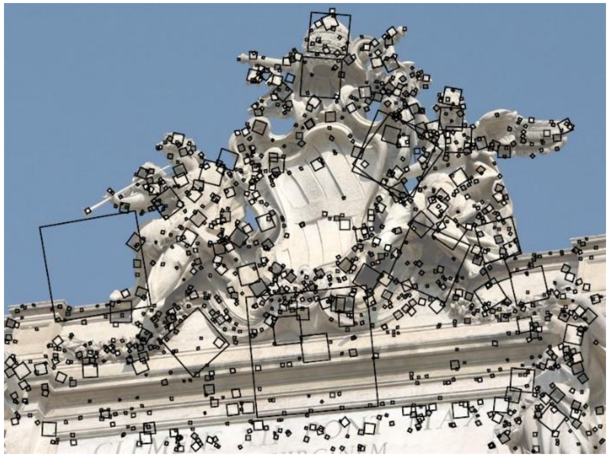
Keypoint Detection

- Corners
  - Image locations with locally prominent intensity variation
- Examples: Harris, FAST

- Blobs
  - Image regions that stick out from their surrounding in intensity/texture
- F.e.: LoG, DoG (SIFT), SURF

Harris Corners

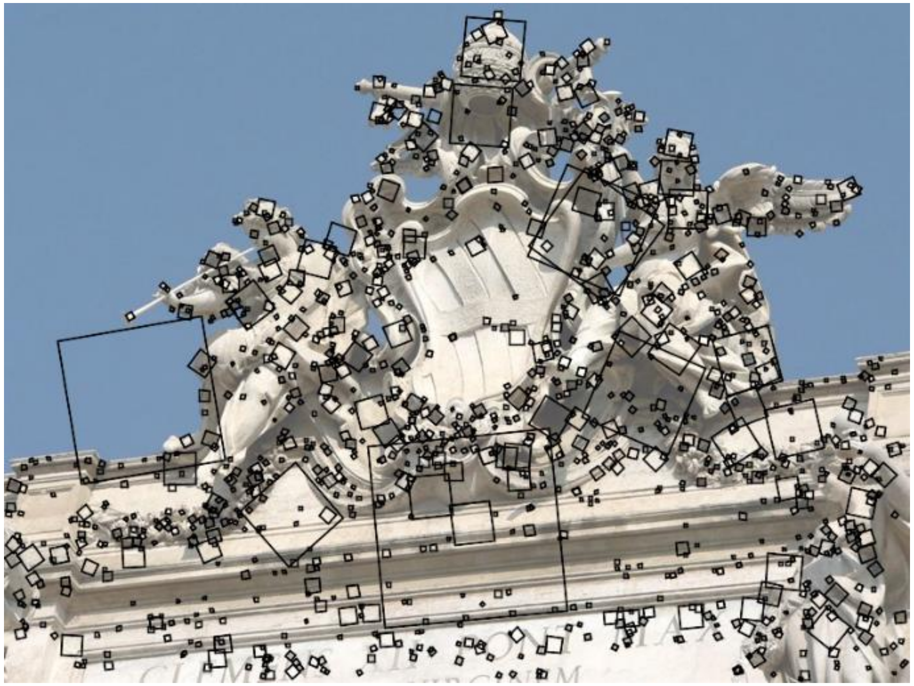
Image source: Svetlana Lazebnik



DoG (SIFT) Blobs

Keypoint Detection

- Invariance for view-point changes
  - Translation
  - Rotation
  - Scale
  - Perspective



DoG (SIFT) Blobs

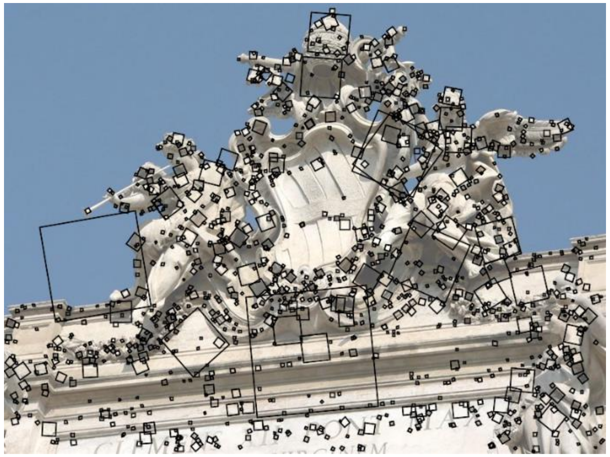
Image source: Svetlana Lazebnik

Keypoint Detection

- Corners vs. blobs for visual odometry:
  - Typically corners provide higher spatial localization accuracy, but are less well localized in scale
  - Corners are typically detected in less distinctive local image regions
  - Highly run-time efficient corner detectors exist (f.e. FAST)

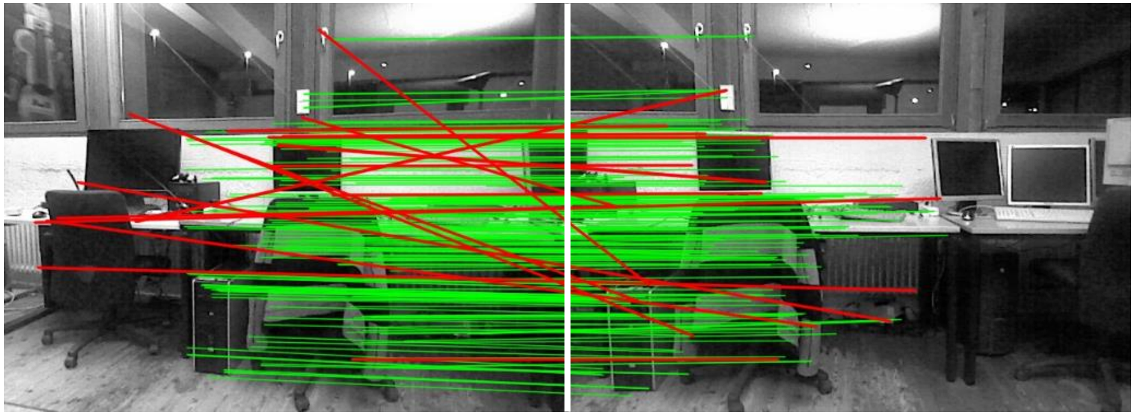
Harris Corners

Image source: Svetlana Lazebnik



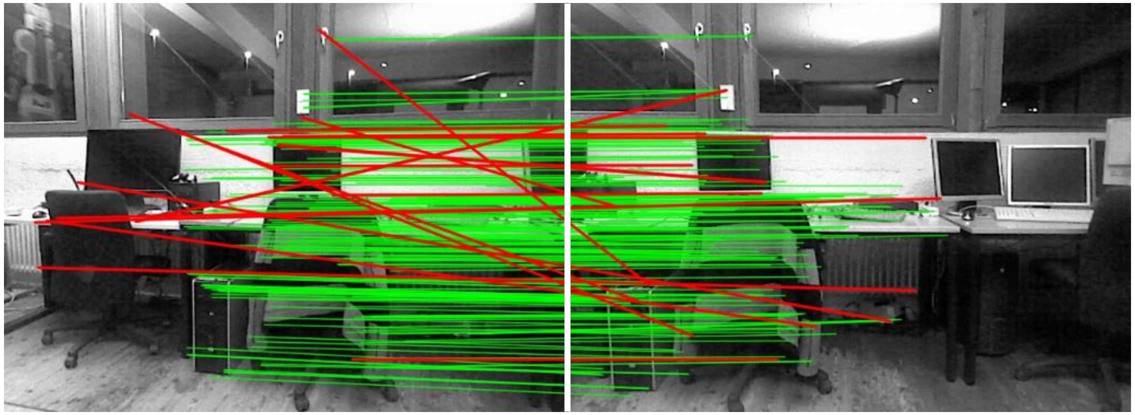
DoG (SIFT) Blobs

Keypoint Matching



- Desirable properties for VO:
  - High recall
  - Precision
  - Robustness
  - Computational efficiency

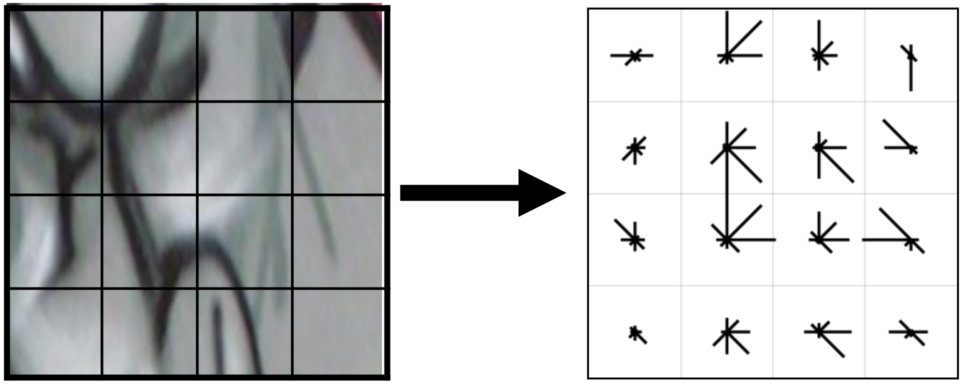
Keypoint Matching



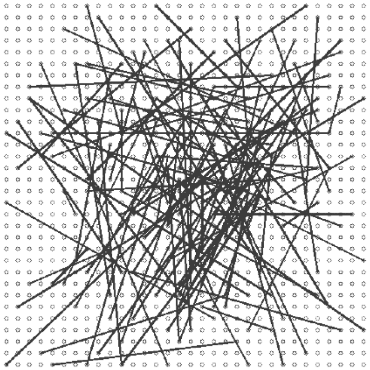
- Data association principles:
  - Matching by reprojection error / distance to epipolar line:
    assumes an initial guess for camera motion (f.e. Kalman filter prediction, IMU, or wheel odometry)
  - Detect-then-track (f.e. KLT-tracker):
    Correspondence search by local image alignment, assumes incremental small (but unknown) motion between images
  - Matching by descriptor:
    scale-/viewpoint-invariant local descriptors
  - Robustness through outlier rejection (f.e. RANSAC) for motion estimation

Local Feature Descriptors

- Desirable properties for VO: 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 test locations

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

First-Order Error Propagation

- Given a non-linear function in a Gaussian variable

$$\mathbf{y} = f(\mathbf{x})$$

- Apply first-order Taylor approximation

$$\mathbf{y} \approx f(\mathbf{x}_0) + \nabla_{\mathbf{x}} f(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0)$$

- Note: Linear transformation $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ of Gaussian variable remains Gaussian: $\mathbf{y} \sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu}_{\mathbf{x}} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{A}^\top)$

- Gaussian approximation of the non-linearly transformed variable

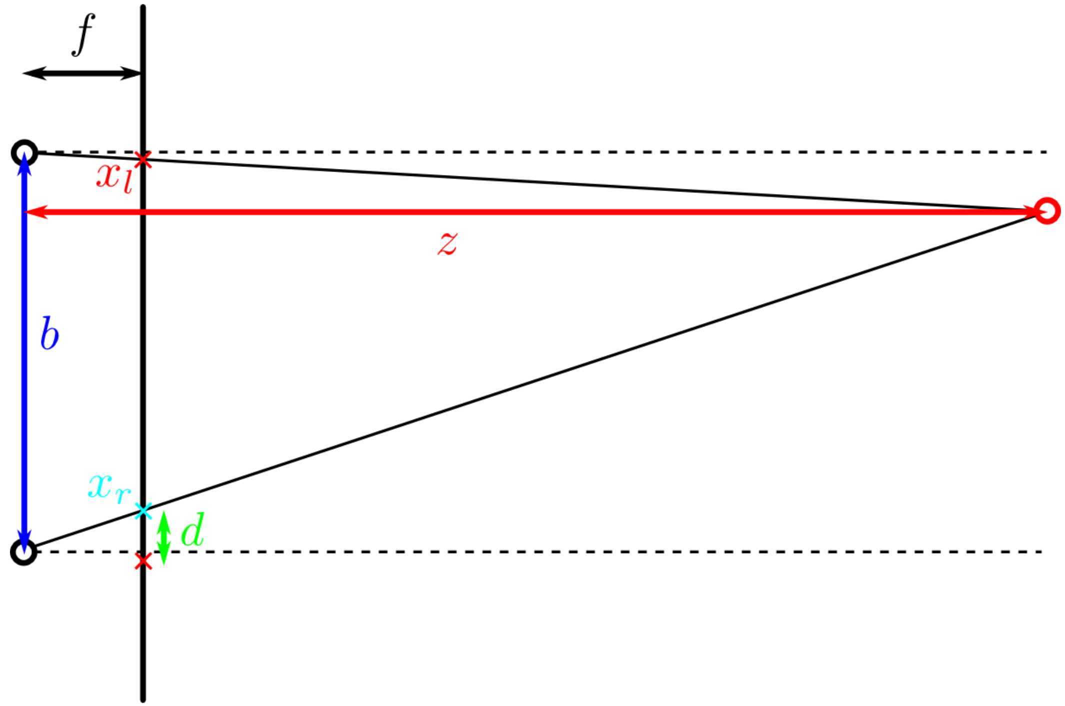
$$\mathbf{y} \sim \mathcal{N}(f(\boldsymbol{\mu}_{\mathbf{x}}), \nabla_{\mathbf{x}} f(\boldsymbol{\mu}_{\mathbf{x}})\boldsymbol{\Sigma}_{\mathbf{x}}\nabla_{\mathbf{x}} f(\boldsymbol{\mu}_{\mathbf{x}})^\top)$$

Disparity and Depth

Similar triangles:

$$\frac{b}{z} = \frac{b-d}{z-f}$$

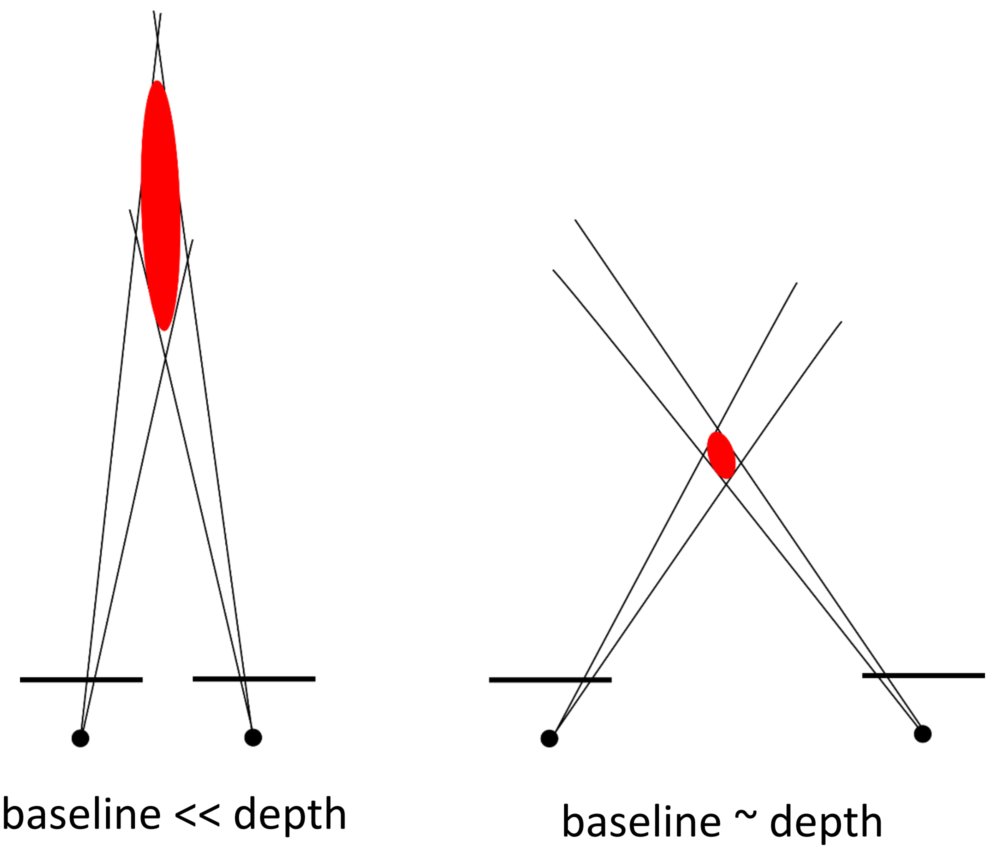
$$\Rightarrow \quad d = \frac{bf}{z}$$



- Let‘s consider a simple case when camera planes are parallel and focal lengths are equal
  - Disparity d is inversely proportional to depth z: The larger the depth, the smaller the disparity
  - Disparity d is proportional to baseline b: The larger the baseline, the larger the disparity

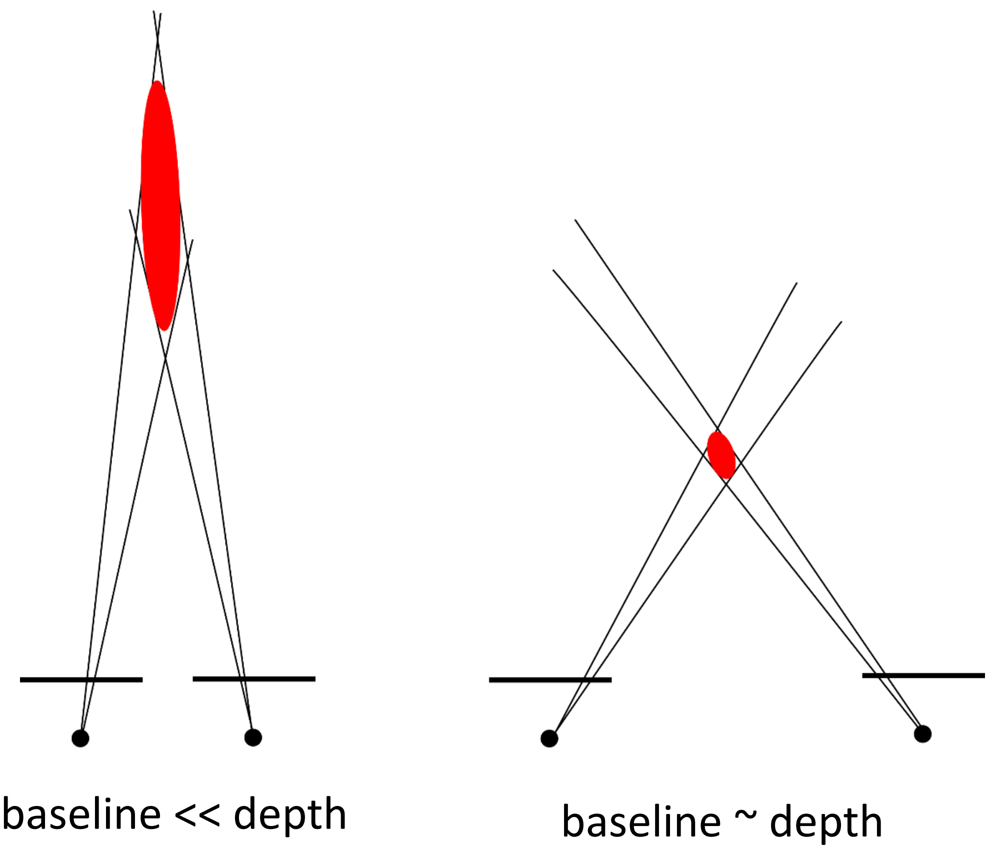
Uncertainty of Depth Estimates

- Given Gaussian uncertainty in the disparity
- Can we quantify the uncertainty in the depth estimate?
  - Blackboard



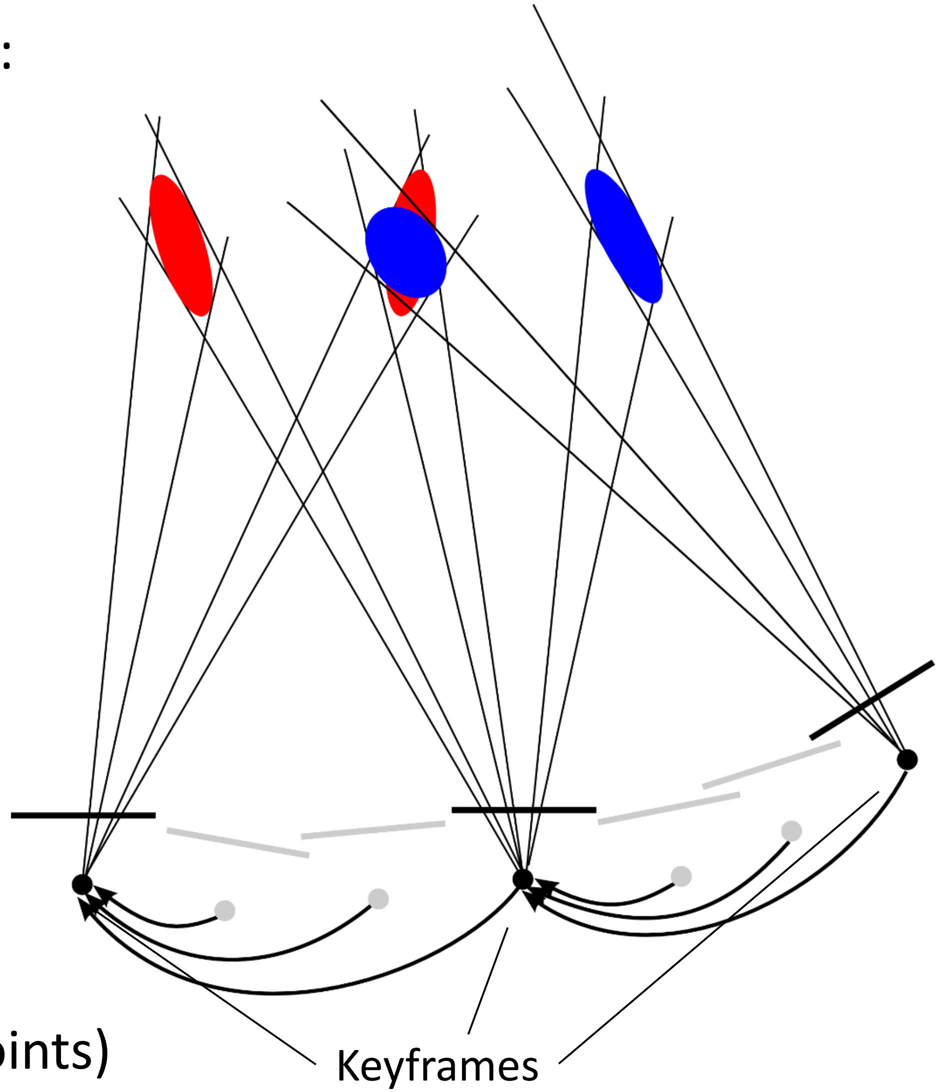
Drift in Motion Estimates

- Since we aggregate pose estimates from relative pose estimates, estimation errors in relative poses accumulate: Drift
- Noisy observations of 2D image point location
- How does uncertainty in motion estimate depend on observation noise?

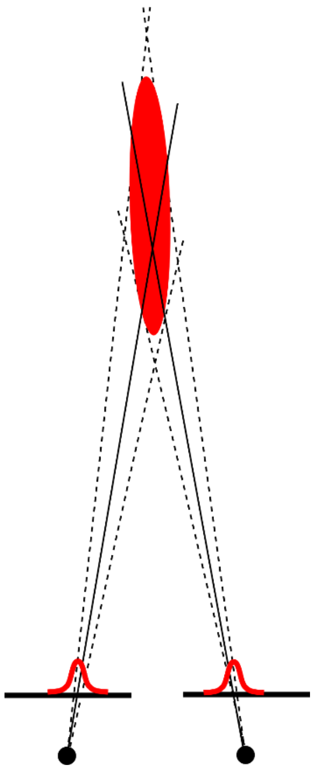


Keyframes

- Popular approach to reduce drift: Keyframes
- Carefully select reference images for motion estimation / triangulation
- Incrementally estimate motion towards keyframe
- If baseline sufficient (and/or image overlap small), create next keyframe (and for instance triangulate 3D positions of keypoints)



Probabilistic Modelling



- Model image point observation likelihood $p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi})$

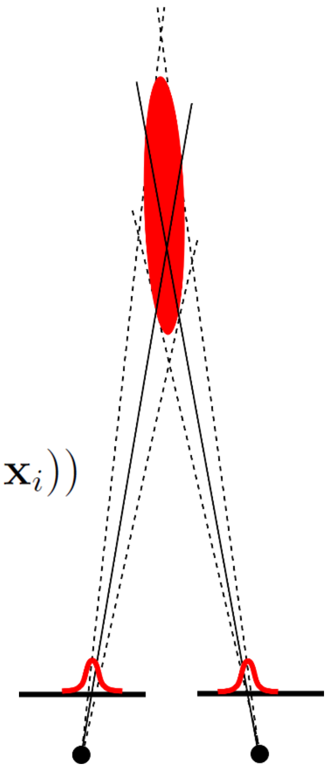
  f.e. Gaussian: $p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi}) = \mathcal{N}(\mathbf{y}_i; \pi(\mathbf{T}(\boldsymbol{\xi})\overline{\mathbf{x}}_i), \boldsymbol{\Sigma}_{\mathbf{y}_i})$

- Optimize maximum a-posteriori likelihood of estimates

  $p(\mathcal{X}, \boldsymbol{\xi} \mid \mathcal{Y}) \propto p(\mathcal{Y} \mid \mathcal{X}, \boldsymbol{\xi})\, p(\mathcal{X}, \boldsymbol{\xi}) = p(\mathcal{X}, \boldsymbol{\xi}) \prod_{i=1}^{N} p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi})$

  Neg. log-likelihood: $E(\mathcal{X}, \boldsymbol{\xi}) = -\log(p(\mathcal{X}, \boldsymbol{\xi})) - \sum_{i=1}^{N} \log(p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi}))$

Probabilistic Modelling



- Gaussian prior and observation likelihood:

$$E(\mathcal{X}, \boldsymbol{\xi}) = \text{const.} + \left(\boldsymbol{\xi} - \boldsymbol{\mu}_{\boldsymbol{\xi},0}\right)^\top \boldsymbol{\Sigma}_{\boldsymbol{\xi},0}^{-1} \left(\boldsymbol{\xi} - \boldsymbol{\mu}_{\boldsymbol{\xi},0}\right) +$$

$$\sum_{i=1}^{N} \left(\mathbf{x}_i - \boldsymbol{\mu}_{\mathbf{x}_i,0}\right)^\top \boldsymbol{\Sigma}_{\mathbf{x}_i,0}^{-1} \left(\mathbf{x}_i - \boldsymbol{\mu}_{\mathbf{x}_i,0}\right) + \left(\mathbf{y}_i - \pi\left(\mathbf{T}(\boldsymbol{\xi})\mathbf{x}_i\right)\right)^\top \boldsymbol{\Sigma}_{\mathbf{y}_i}^{-1} \left(\mathbf{y}_i - \pi\left(\mathbf{T}(\boldsymbol{\xi})\mathbf{x}_i\right)\right)$$

- Gauss-Newton yields Gaussian estimate $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
  - MAP solution $\arg\min_{\mathbf{x}} E(\mathbf{x}) = \frac{1}{2}\mathbf{r}(\mathbf{x})^\top \mathbf{W}\mathbf{r}(\mathbf{x})$ is the mean
  - The inverse Hessian of the Gauss-Newton approximation

$$\boldsymbol{\Sigma} \approx \left(\nabla_{\mathbf{x}}\mathbf{r}(\boldsymbol{\mu})^\top \mathbf{W} \nabla_{\mathbf{x}}\mathbf{r}(\boldsymbol{\mu})\right)^{-1}$$

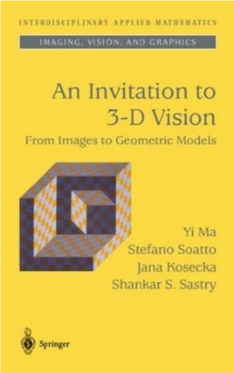
yields an approximate covariance of the estimates

Motion Estimation for Camera Type

Correspondences	Monocular	Stereo	RGB-D
2D-to-2D	X	X	X
2D-to-3D	X	X	X
3D-to-3D		X	X

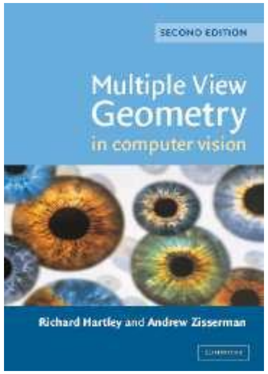
Further Reading

- MASKS and MVG textbooks



MASKS

An Invitation to 3D Vision,
Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry,
Springer, 2004



MVG

Multiple View Geometry in Computer Vision,
R. Hartley and A. Zisserman,
Cambridge University Press, 2004

Lessons Learned Today

- Motion estimation from point correspondences
  - 2D-to-3D correspondences, DLT algorithm for PnP
  - 3D-to-3D correspondences, Arun's method
  - Rudimentary visual odometry approaches based on 2D-to-3D and 3D-to-3D motion estimation (not robust yet)
  - Properties for keypoint detection and matching
  - Probabilistic modeling can incorporate priors and find probabilistic estimates
  - Uncertainty in structure and motion estimation depends on observation noise
  - Visual odometry methods accumulate drift
  - Motion estimation towards keyframes reduces drift

Thanks for your attention!