Computer Vision Group
Prof. Daniel Cremers

Technische Universität München

Robotic 3D Vision

Lecture 6: Visual Odometry 2 – Indirect Methods cont.

WS 2020/21

Dr. Niclas Zeller
Artisense GmbH

What We Will Cover Today

- Indirect visual odometry methods
  - 2D-to-2D motion estimation
  - 2D-to-3D motion estimation
  - 3D-to-3D motion estimation
- Properties of keypoint detection and matching
- Estimation uncertainty

Recap: Special Euclidean Group SE(n)

- Euclidean transformation matrices have a special structure:

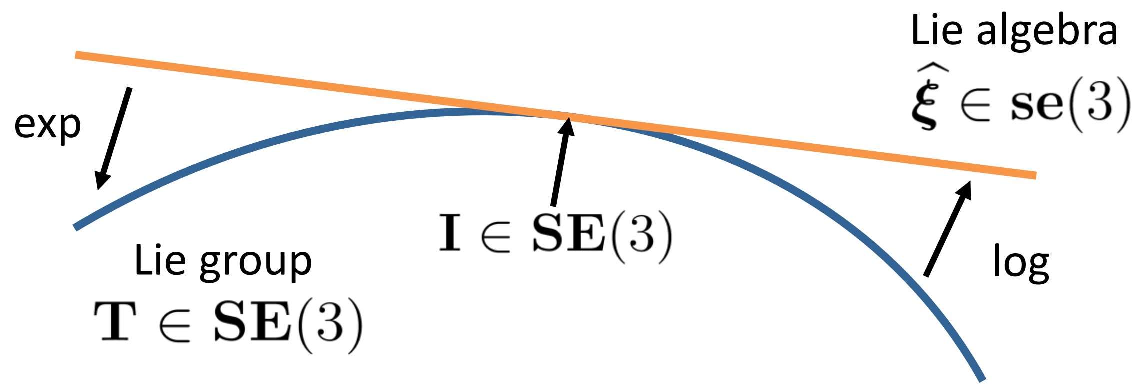
$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \in \mathbf{SE}(3) \subset \mathbb{R}^{4\times 4}$$

  - 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:

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

Recap: Representing Motion using Lie Algebra se(3)



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

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

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

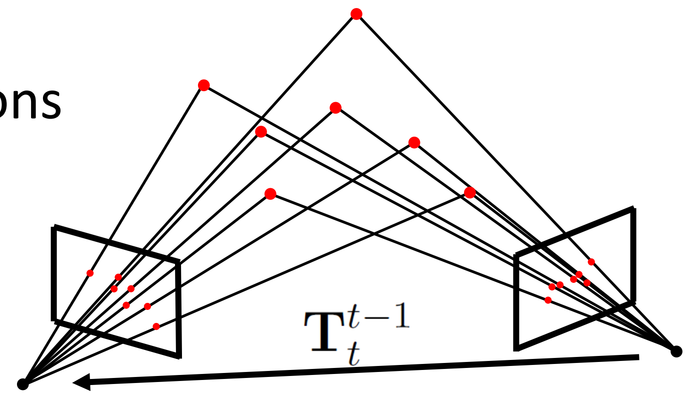
$$\widehat{\boldsymbol{\xi}} := \begin{pmatrix} \widehat{\boldsymbol{\omega}} & \mathbf{v} \\ \mathbf{0} & 0 \end{pmatrix} \in \mathbb{R}^{4\times 4}$$

- $\mathbf{SE}(3)$ 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: Some Notation for Twist Coordinates

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

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

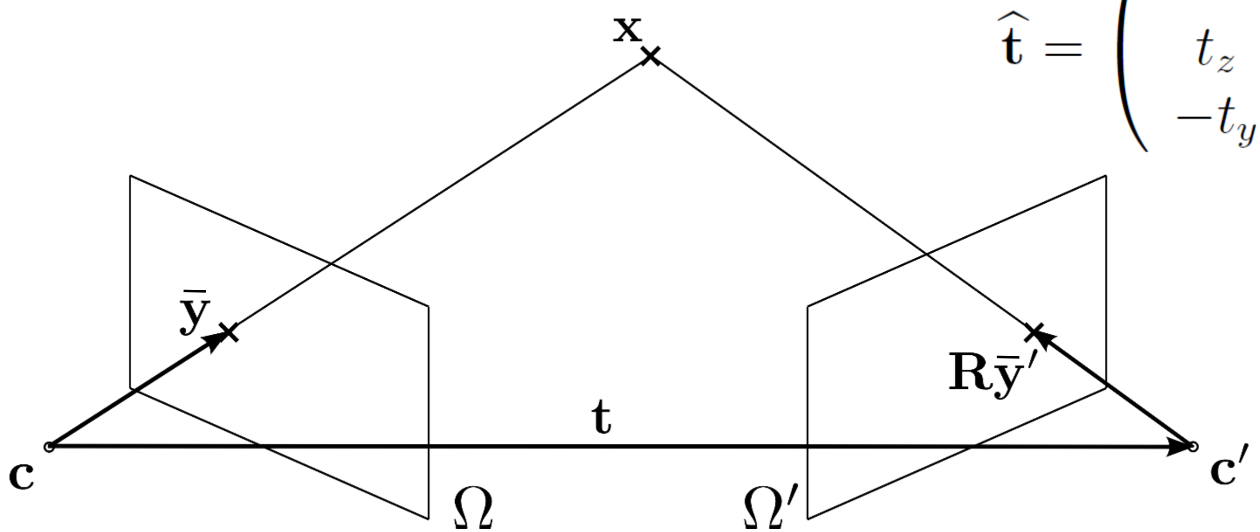
- Naive try: minimize reprojection error using least squares

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

- Convexity? Uniqueness (scale-ambiguity)?
- Alternative algebraic approach

Recap: Essential Matrix

$$\hat{\mathbf{t}} = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix}$$



$$\hat{\mathbf{t}} = [\mathbf{t}]_\times$$

- The rays to the 3D point and the baseline $\mathbf{t}$ are coplanar

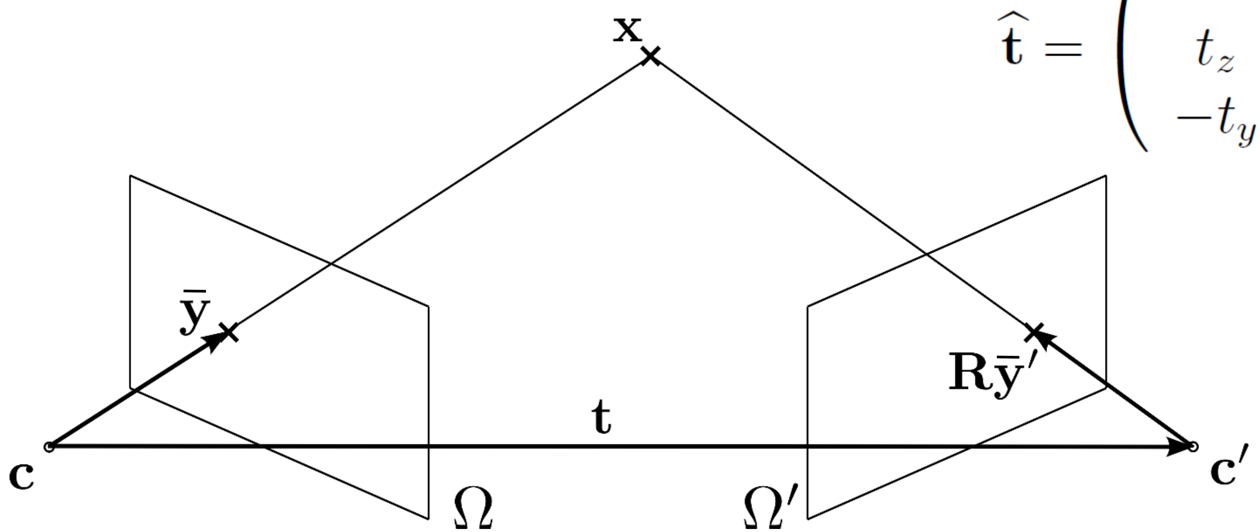
$$\tilde{\mathbf{y}}^\top (\mathbf{t} \times \mathbf{R}\tilde{\mathbf{y}}') = 0 \Leftrightarrow \tilde{\mathbf{y}}^\top \hat{\mathbf{t}} \mathbf{R} \tilde{\mathbf{y}}' = 0$$

- The essential matrix $\mathbf{E} := \hat{\mathbf{t}}\mathbf{R}$ captures the relative camera pose
- Each point correspondence provides an „epipolar constraint"
- 5 correspondences suffice to determine $\mathbf{E}$ (simpler: 8-point algorithm)

Recap: Fundamental Matrix

$$\widehat{\mathbf{t}} = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix}$$

$$\widehat{\mathbf{t}} = [\mathbf{t}]_\times$$



- The rays to the 3D point and the baseline $\mathbf{t}$ are coplanar

$$\tilde{\mathbf{y}}_p^\top \mathbf{C}^{-\top} \widehat{\mathbf{t}} \mathbf{R} \mathbf{C}^{-1} \tilde{\mathbf{y}}_p' = \tilde{\mathbf{y}}_p^\top \mathbf{F} \tilde{\mathbf{y}}_p' = 0$$

- The fundamental matrix $\mathbf{F} := \mathbf{C}^{-\top} \widehat{\mathbf{t}} \mathbf{R} \mathbf{C}^{-1}$ captures the relative camera pose and camera intrinsics
- Each point correspondence provides an „epipolar constraint“
- Can be estimated from at least 7 point correspondences

Some Properties of E and F

- $\mathbf{F} \in \mathbb{R}^{3\times 3}$ is a fundamental matrix iff $\operatorname{rank}(\mathbf{F}) = 2$

- $\mathbf{E} \in \mathbb{R}^{3\times 3}$ is an essential matrix iff $\operatorname{rank}(\mathbf{E}) = 2$ and its non-zero singular values are equal

- $\mathbf{E} \in \mathbb{R}^{3\times 3}$ is a normalized esssential matrix iff $\operatorname{rank}(\mathbf{E}) = 2$ and its non-zero singular values are 1

$$\|\mathbf{E}\| = \left\|\widehat{\mathbf{t}}\right\| = 1$$

- (Normalized) essential space: set of all (normalized) essential matrices

Eight-Point Algorithm

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

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

     using Kronecker product $\mathbf{a}_i = \widetilde{\mathbf{y}}_i \otimes \widetilde{\mathbf{y}}_i'$ and $\mathbf{E}_s = (e_{11}, e_{12}, e_{13}, \dots, e_{33})^\top$
  2. Apply singular value decomposition (SVD) on $\mathbf{A} = \mathbf{U_A S_A V_A^\top}$ and unstack the 9th column of $\mathbf{V_A}$ into $\widetilde{\mathbf{E}}$
  3. Project the approximate $\widetilde{\mathbf{E}}$ into the (normalized) essential space: Determine the SVD of $\widetilde{\mathbf{E}} = \mathbf{U}\,\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)\,\mathbf{V}^\top$ 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

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

Eight-Point Algorithm cont.

- Algorithm (cont.):
  - Determine one of the following 4 possible solutions that intersects the points in front of both cameras:

$$\mathbf{R} = \mathbf{U}\mathbf{R}_Z^\top\left(\pm\frac{\pi}{2}\right)\mathbf{V}^\top \qquad \widehat{\mathbf{t}} = \mathbf{U}\mathbf{R}_Z\left(\pm\frac{\pi}{2}\right)\mathrm{diag}(1,1,0)\mathbf{U}^\top$$

- A derivation of the eight-point algorithm can be found in the „An Invitation to 3-D Vision“ textbook, Ch. 5
- Algebraic solution does not minimize reprojection error
- Refine using non-linear least-squares of reprojection error

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

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

  subject to $\|\mathbf{E}_s\|_2^2 = 1$ through the SVD on $\mathbf{A}$
- We find a least squares fit to the epipolar constraints
- Each epipolar constraint measures the volume spanned by $\mathbf{y}$, $\mathbf{t}$, and $\mathbf{R}\mathbf{y}$‘

Notes on Eight-Point Algorithm

- Points need to be in „general position“ to recover unique E: certain degenerate configurations exists (f.e. points on a plane, specific quadratic surfaces)
- No translation, ideally: $\left\|\widehat{\mathbf{t}}\right\| = 0 \Rightarrow \|\mathbf{E}\| = 0$
- But: for small translations, signal-to-noise ratio of image parallax may be problematic: „spurious“ pose estimate
- Non-linear 5-point algorithm with up to 10 (possibly complex) solutions (D. Nister, An Efficient Solution to the Five-Point Relative Pose Problem, CVPR 2004)

Normalized Eight-Point Algorithm

- Hartley, In Defense of the 8-Point Algorithm, IEEE PAMI 1997
- $\mathbf{A}$ can be numerically ill-conditioned when estimating the fundamental matrix with the eight-point algorithm naively

250906.36	183269.57	921.81	200931.10	146766.13	738.21	272.19	198.81	1.00
2692.28	131633.03	176.27	6196.73	302975.59	405.71	15.27	746.79	1.00
416374.23	871684.30	935.47	408110.89	854384.92	916.90	445.10	931.81	1.00
191183.60	171759.40	410.27	416435.62	374125.90	893.65	465.99	418.65	1.00
48988.86	30401.76	57.89	298604.57	185309.58	352.87	846.22	525.15	1.00
164786.04	546559.67	813.17	1998.37	6628.15	9.86	202.65	672.14	1.00
116407.01	2727.75	138.89	169941.27	3982.21	202.77	838.12	19.64	1.00
135384.58	75411.13	198.72	411350.03	229127.78	603.79	681.28	379.48	1.00

  - Noise attenuates stronger in large pixel coordinates (quadratic dependency)
  - Least squares (SVD) more sensitive to noise in large coordinates
  - „Imbalanced“ since pixel coordinates start at (0,0)

Normalized Eight-Point Algorithm

- Popular approach: Normalize coordinates to zero mean and standard deviation $\sqrt{2}$ in each image separately

$$\bar{\mathbf{z}} = \frac{\sqrt{2}}{\sigma}(\bar{\mathbf{y}}_p - \boldsymbol{\mu})$$

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N}\bar{\mathbf{y}}_p \qquad \boldsymbol{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}\|\bar{\mathbf{y}}_p - \boldsymbol{\mu}\|_2^2$$

- Find $\mathbf{B}$ and $\mathbf{B}'$ to normalize pixel coordinates

$$\bar{\mathbf{z}} = \mathbf{B}\bar{\mathbf{y}}_p \qquad \mathbf{B} = \begin{pmatrix} \frac{\sqrt{2}}{\sigma} & 0 & -\frac{\sqrt{2}}{\sigma}\mu_x \\ 0 & \frac{\sqrt{2}}{\sigma} & -\frac{\sqrt{2}}{\sigma}\mu_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$\bar{\mathbf{z}}' = \mathbf{B}'\bar{\mathbf{y}}'_p$$

Normalized Eight-Point Algorithm

- Apply eight-point algorithm on normalized coordinates with epipolar constraints

$$\overline{\mathbf{y}}_p^\top \mathbf{B}^\top \mathbf{F}' \mathbf{B}' \overline{\mathbf{y}}_p' = 0$$

- Recover $\mathbf{F}$ from $\mathbf{F}'$

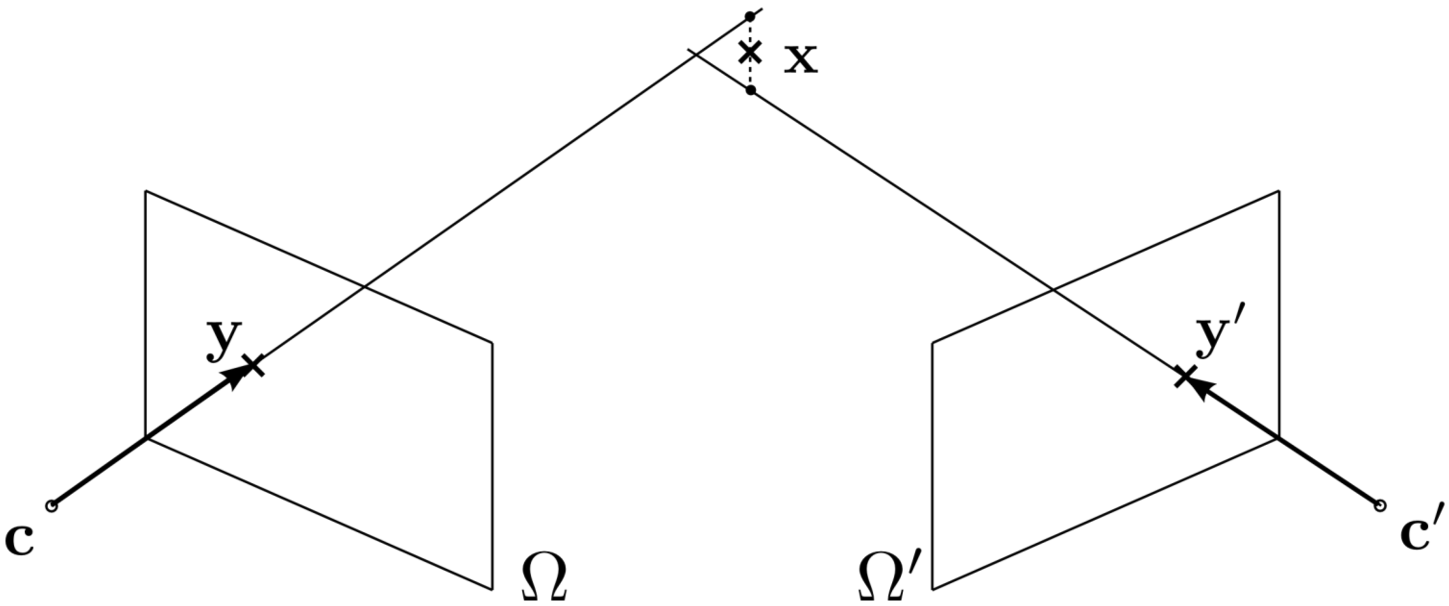
$$\mathbf{F} = \mathbf{B}^\top \mathbf{F}' \mathbf{B}'$$

Eight-Point Algorithm for F

- Calibrated case: we know camera intrinsics, we can estimate E
- Uncalibrated case: we do not know camera intrinsics, we can only estimate F

- In the uncalibrated case, rotation and translation can not be recovered from F due to the unknown camera intrinsics

Triangulation



Triangulation

- Goal: Reconstruct 3D point $\bar{\mathbf{x}} = (x, y, z, 1)^{\mathrm{T}}$ from 2D image observations $\{\mathbf{y}_1, \mathbf{y}_2\}$ for known camera poses $\{\mathbf{T}_1, \mathbf{T}_2\}$
  - Can be extendend to multiple images, as long as scale is known (or scale needs to be estimated)
  - In general we assume $\mathbf{T}_1 = \mathbf{I}$

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

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

  - Each image provides one constraint $\mathbf{y}_i = \pi(\mathbf{T}_i \bar{\mathbf{x}})$ $\qquad \mathbf{y}_i = (x_i, y_i)^{\mathrm{T}}$

$$([\mathbf{T}_i]_1 - x_i[\mathbf{T}_i]_3) \cdot \bar{\mathbf{x}} = 0$$
$$([\mathbf{T}_i]_2 - y_i[\mathbf{T}_i]_3) \cdot \bar{\mathbf{x}} = 0$$

$$([\mathbf{R}_i]_1 - x_i[\mathbf{R}_i]_3) \cdot \mathbf{x} = -(t_x - x_i t_z)$$
$$([\mathbf{R}_i]_2 - y_i[\mathbf{R}_i]_3) \cdot \mathbf{x} = -(t_y - y_i t_z)$$

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

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 :

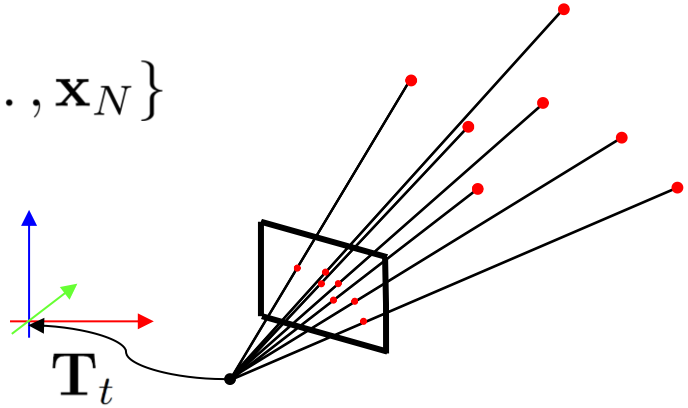
1. Extract and match keypoints between I_{k-1} and I_k
2. Compute relative pose $\mathbf{T}_k^{k-1}$ from essential matrix between I_k , I_{k-1}
3. Fine-tune pose estimate by minimizing reprojection error
4. Compute relative scale and rescale translation of $\mathbf{T}_k^{k-1}$
5. Aggregate camera pose 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 $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and corresponding image observations

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

  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 $\mathbf{P} = (\mathbf{R}\ \mathbf{t}) \in \mathbb{R}^{3\times 4} = \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{pmatrix}$

- 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:

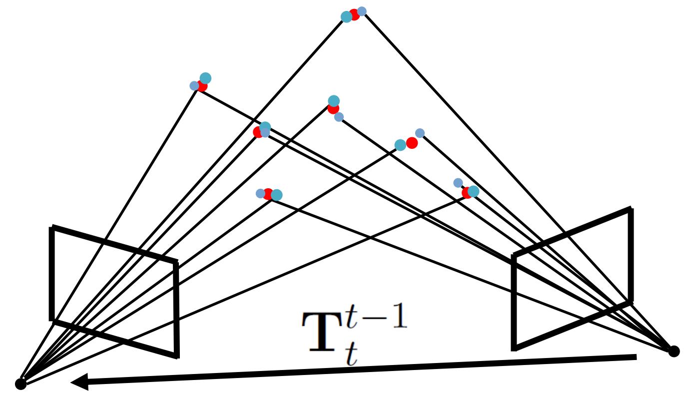
Initialize:

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 new 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 e.g. to RGB-D cameras or also Lidar
  - Should only be used if we have very accurate 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

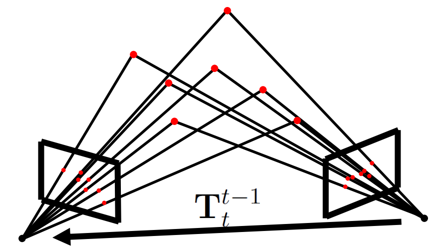
$$\mathbf{A} = \sum_{i=1}^{N} \left(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1}\right)\left(\mathbf{x}_t - \boldsymbol{\mu}_t\right)^\top \qquad \mathbf{A} = \mathbf{USV}^\top \qquad \mathbf{R}_{t-1}^{t} = \mathbf{VU}^\top$$

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

Motion Estimation from Point Correspondences

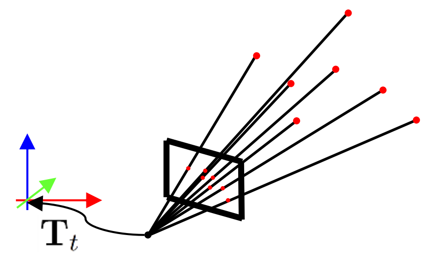
- **2D-to-2D**
  - Reprojection error:

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



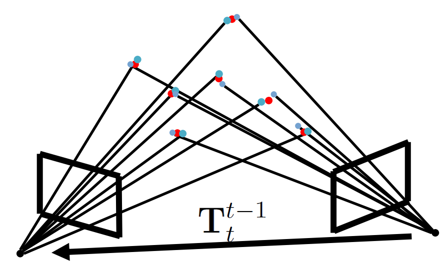
  - Linear algorithm: 8-point

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



  - Linear algorithm: DLT PnP

- **3D-to-3D**
  - 3D geometric error: $E\left(\mathbf{T}_t^{t-1}\right) = \sum_{i=1}^{N} \left\|\bar{\mathbf{x}}_{t-1,i} - \mathbf{T}_t^{t-1}\bar{\mathbf{x}}_{t,i}\right\|_2^2$



  - Linear algorithm: Arun's method

- Allways consider the error distribution (least squares is only optimal for Normal distribution)

Further Considerations

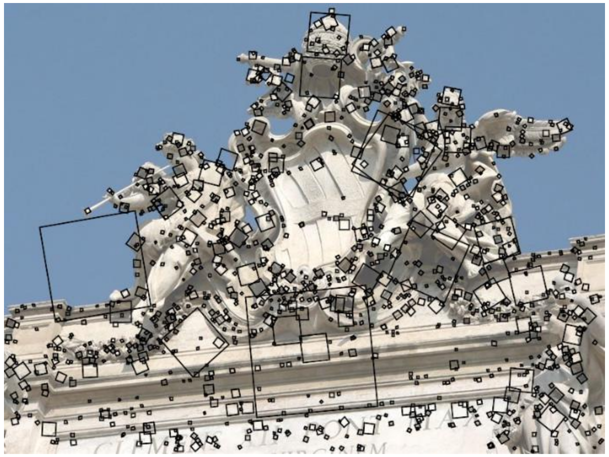
- How to detect keypoints?
- How to match keypoints?
- How to cope with outliers in keypoint matches?
- When to create new 3D keypoints ? Which keypoints to use?
- 2D-to-2D, 2D-to-3D or 3D-to-3D?
- Optimize over more than two frames?
- …

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

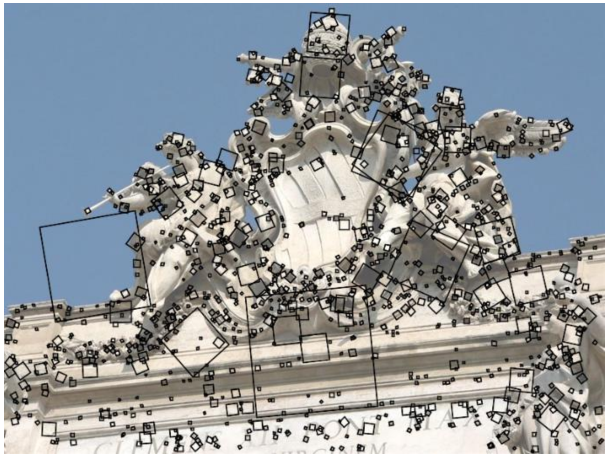
Keypoint Detection

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

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

Harris Corners

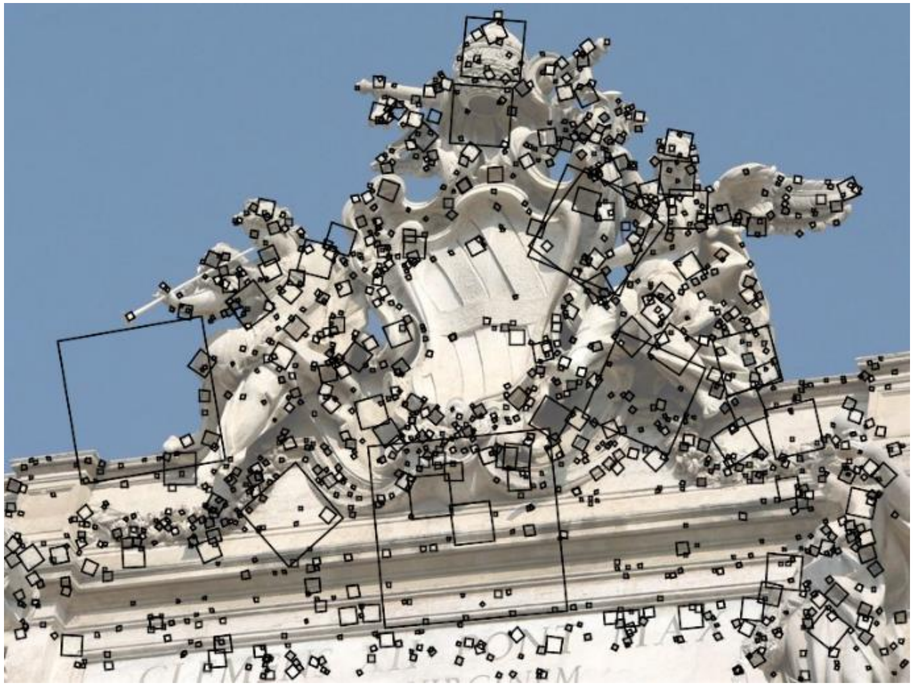
Image source: Svetlana Lazebnik



DoG (SIFT) Blobs

Keypoint Detection

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



DoG (SIFT) Blobs

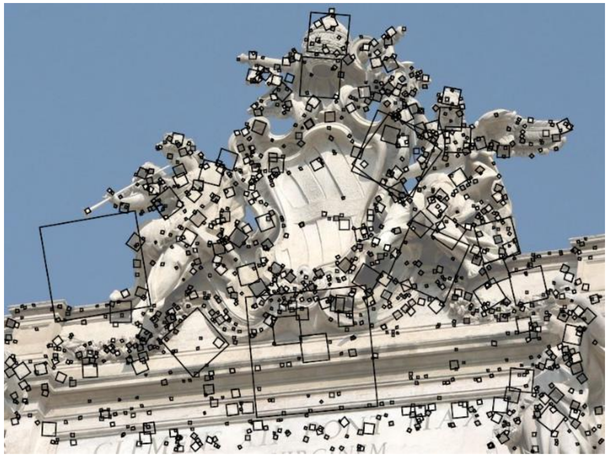
Image source: Svetlana Lazebnik

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)

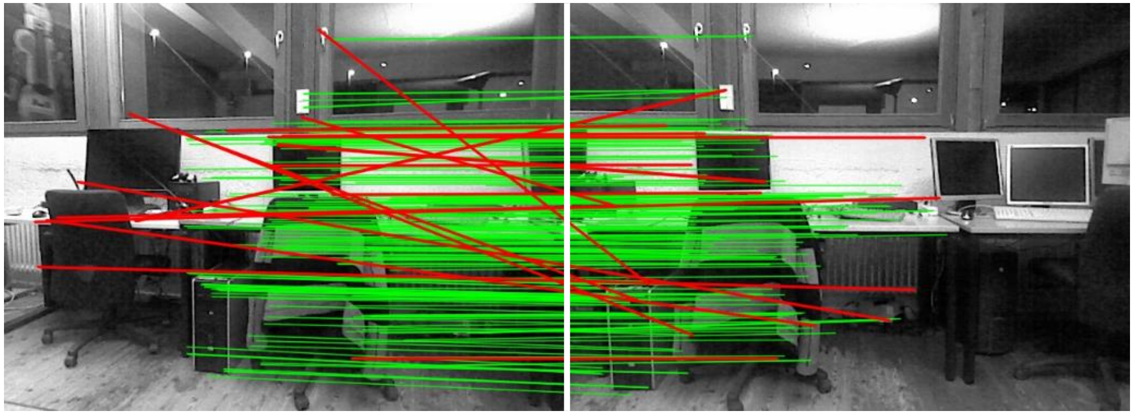
Harris Corners

Image source: Svetlana Lazebnik



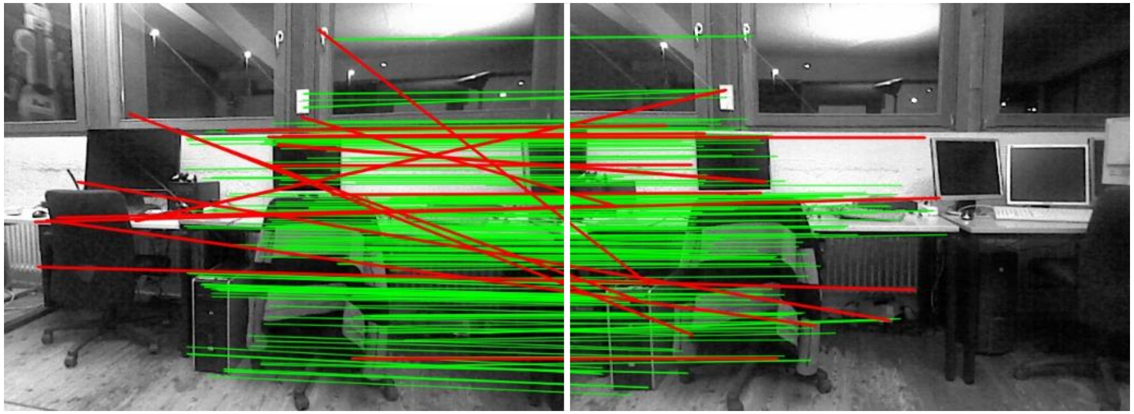
DoG (SIFT) Blobs

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.)
  - Learned descriptors (SuperPoint, etc.)
- Rotation-invariance: Align with dominant orientation in local region
- Scale-invariance: Extract descriptor from different scales

Uncertainty 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

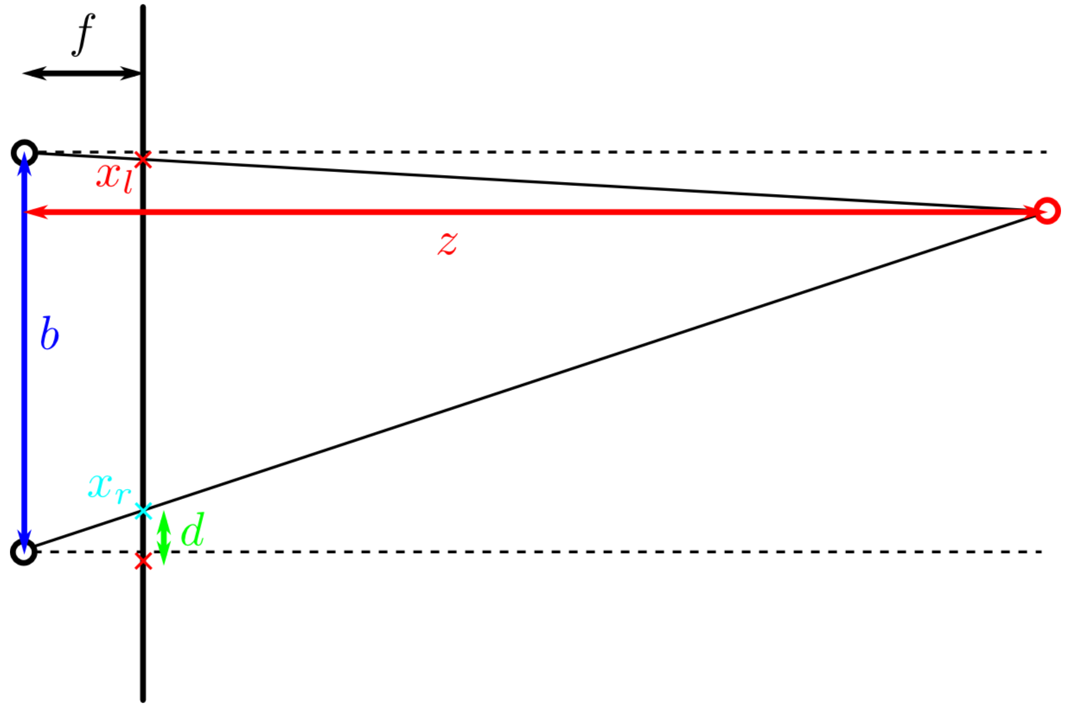
$$\mathbf{y} \sim \mathcal{N}(f(\boldsymbol{\mu}_{\mathbf{x}}), \nabla_{\mathbf{x}} f(\boldsymbol{\mu}_{\mathbf{x}})\boldsymbol{\Sigma}_{\mathbf{x}}\nabla_{\mathbf{x}} f(\boldsymbol{\mu}_{\mathbf{x}})^\top)$$

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 σ_d^2

- Inverse depth z^{-1} will also be Gaussian

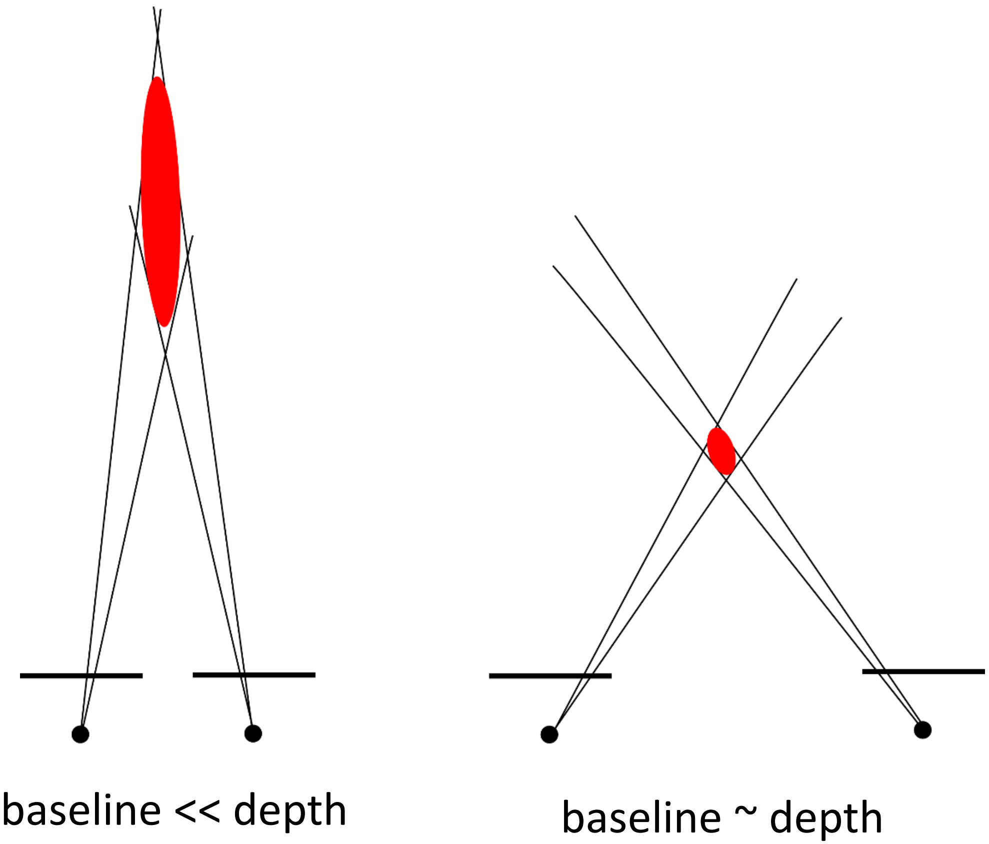
$$z^{-1} = \frac{d}{bf}$$

$$\sigma_{z^{-1}}^2 = \frac{1}{(bf)^2} \sigma_d^2$$

- Uncertainty in depth z can be approximate by a Gaussian with

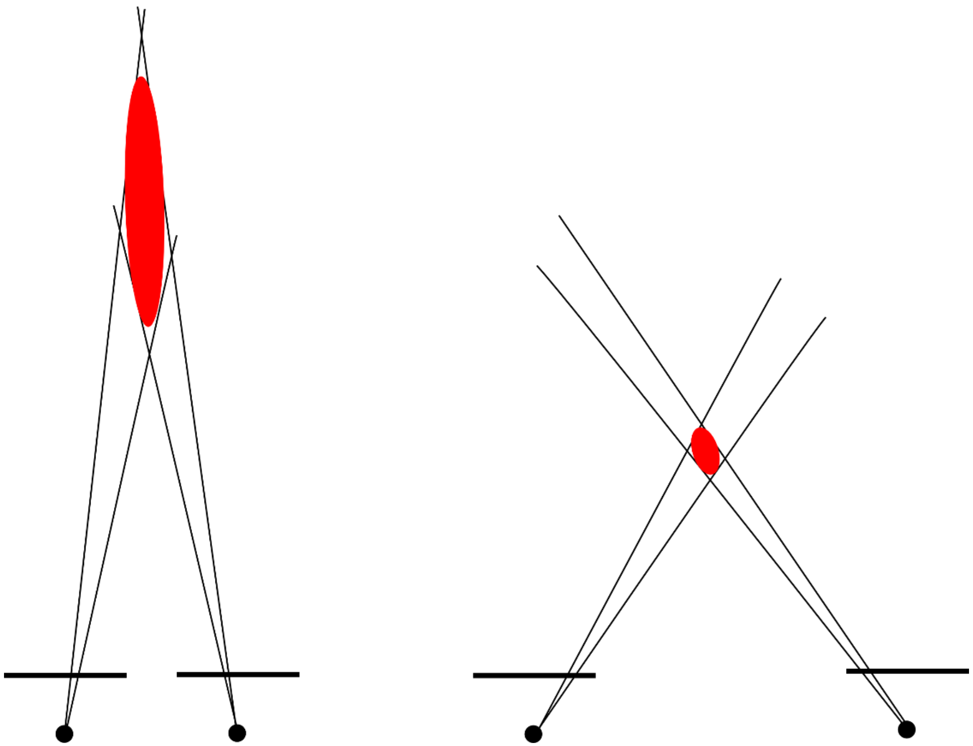
$$\sigma_z^2 = \left|\frac{\partial z}{\partial d}\right|^2 \sigma_d^2 = \frac{(bf)^2}{d^4} \sigma_d^2$$

$$= \frac{z^4}{(bf)^2} \sigma_d^2$$



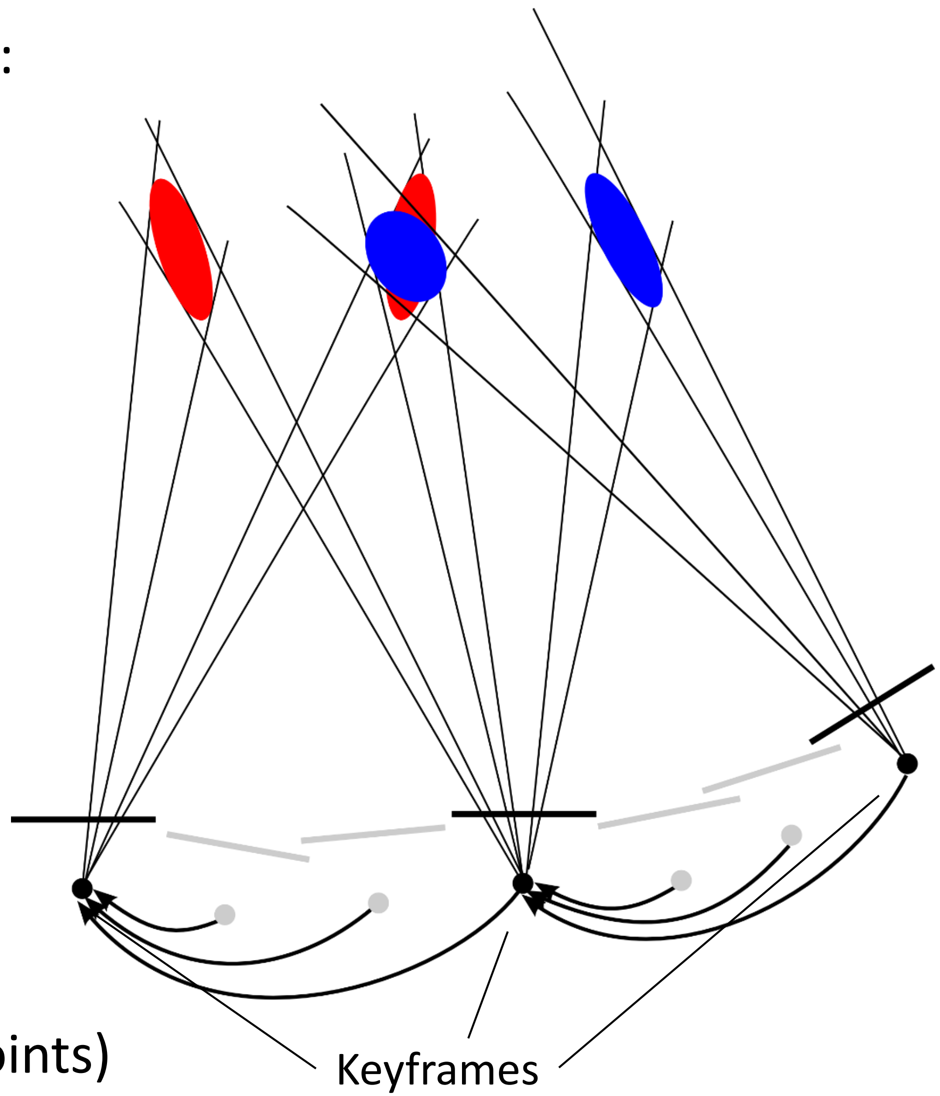
Drift in Motion Estimates

- 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?

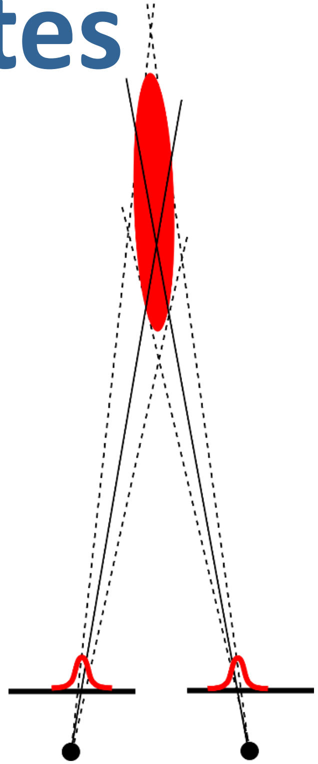


Keyframes

- 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)



Uncertainty in Pose Estimates



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

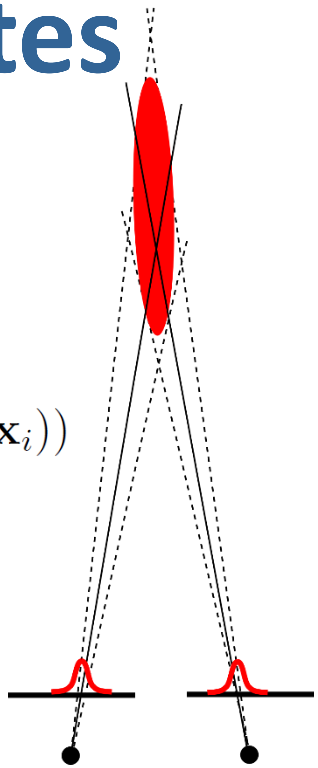
  f.e. Gaussian: $p(\mathbf{y}_i \mid \mathbf{x}_i, \boldsymbol{\xi}) = \mathcal{N}(\mathbf{y}_i; \pi(\mathbf{T}(\boldsymbol{\xi})\overline{\mathbf{x}}_i), \boldsymbol{\Sigma}_{\mathbf{y}_i})$

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

Uncertainty in Pose Estimates



- Gaussian prior and observation likelihood:

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

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

- We use Gauss-Newton to find

$$\arg\min_{\mathbf{x}} E(\mathbf{x}) = \tfrac{1}{2}\mathbf{r}(\mathbf{x})^\top \mathbf{W}\mathbf{r}(\mathbf{x})$$

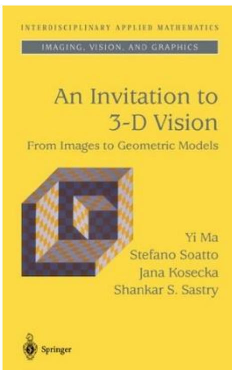
  - $\mathbf{W}$ models inverse covariances of observations and priors
  - The inverse Hessian of the Gauss-Newton approximation

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

yields an approximate covariance of the estimates

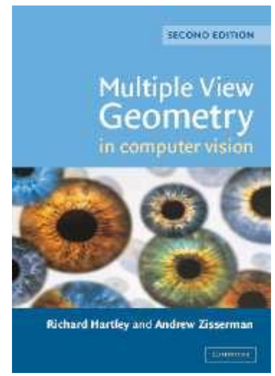
Further Reading

- 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-2D correspondences, eight-point algorithm
  - 2D-to-3D correspondences, DLT algorithm for PnP
  - 3D-to-3D correspondences, Arun's method
- Properties for keypoint detection and matching
- Uncertainty in structure and motion estimation depends on observation noise

Thanks for your attention!

Slides Information

- These slides have been initially created by Jörg Stückler as part of the lecture “Robotic 3D Vision” in winter term 2017/18 at Technical University of Munich.
- The slides have been revised by myself (Niclas Zeller) for the same lecture held in winter term 2020/21
- Acknowledgement of all people that contributed images or video material has been tried (please kindly inform me if such an acknowledgement is missing so it can be added).