Computer Vision 2
WS 2018/19

Part 15 – Visual SLAM I
08.01.2019

Prof. Dr. Bastian Leibe

RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de

Important Announcement

Happy New Year everybody!

Other Announcements

- Exam registration should now work…
  - …for students of Computer Science, Media Informatics, SSE (and some others)
- If you are in another degree program and you can't register
  - Please contact your study advisor (Fachstudienberater)
  - They need to create a node in rwth online to connect the exam with an entry in your degree programme ("Prüfungsleistung"), such that it can be properly credited.
  - Only they can do that, we can't.
- If this somehow does not work, don't panic!
  - We'll figure something out…

Course Outline

- Single-Object Tracking
- Bayesian Filtering
- Multi-Object Tracking
- Visual Odometry
  - Sparse interest-point based methods
  - Dense direct methods
- Visual SLAM & 3D Reconstruction
  - Online SLAM methods
  - Full SLAM methods
- Deep Learning for Video Analysis

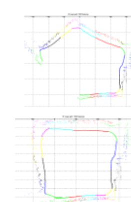


image source: [Clemente et al., RSS 2007]

Topics of This Lecture

- Recap: Direct Methods for Visual Odometry
  - Direct image alignment
  - Lie group se(3) and the exponential map
  - Practical considerations
- Visual SLAM
  - Motivation
  - Probabilistic formulation
- Online SLAM Methods
  - Tracking-and-Mapping
  - Filter-based SLAM
  - EKF SLAM

Recap: Direct vs. Indirect Methods for VO

- Direct methods
  - formulate alignment objective in terms of photometric error (e.g., intensities)

$$p(\mathbf{I}_2 \mid \mathbf{I}_1, \boldsymbol{\xi}) \quad \longrightarrow \quad E(\boldsymbol{\xi}) = \int_{\mathbf{u}\in\Omega} |\mathbf{I}_1(\mathbf{u}) - \mathbf{I}_2(\omega(\mathbf{u}, \boldsymbol{\xi}))| \, d\mathbf{u}$$

- Indirect methods
  - formulate alignment objective in terms of reprojection error of geometric primitives (e.g., points, lines)

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

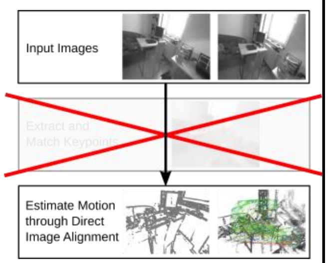
Slide credit: Jörg Stückler

Recap: Direct Visual Odometry Pipeline

- Avoid manually designed keypoint detection and matching
- Instead: direct image alignment

$$E(\boldsymbol{\xi}) = \int_{\mathbf{u}\in\Omega} |\mathbf{I}_1(\mathbf{u}) - \mathbf{I}_2(\omega(\mathbf{u},\boldsymbol{\xi}))|\, du$$

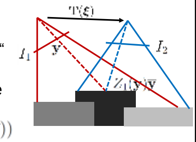
- Warping requires depth
  - RGB-D
  - Fixed-baseline stereo
  - Temporal stereo, tracking and (local) mapping



Recap: Direct Image Alignment Principle

- Idea
  - If we know the pixel depth, we can „simulate" an image from a different viewpoint
  - Ideally, the warped image is the same as the image taken from that pose:

$$I_1(\mathbf{y}) = I_2(\pi(\mathbf{T}(\boldsymbol{\xi})Z_1(\mathbf{y})\bar{\mathbf{y}}))$$



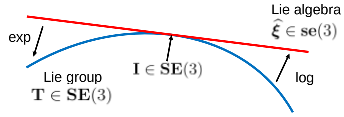
  - Estimate the warp by minimizing the residuals (similar to LK alignment)

$$E(\boldsymbol{\xi}) = \sum_{\mathbf{y}\in\Omega} \frac{r(\mathbf{y},\boldsymbol{\xi})^2}{\sigma_I^2} \qquad r(\mathbf{y},\boldsymbol{\xi}) = I_1(\mathbf{y}) - I_2(\pi(\mathbf{T}(\boldsymbol{\xi})Z_1(\mathbf{y})\bar{\mathbf{y}}))$$

⇒ Non-linear least-squares problem (use second-order tools)

  - Important issue in practice: *How to parametrize the poses?*

Recap: Representing Motion using Lie Algebra se(3)



$$\boldsymbol{\xi} := \begin{pmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{pmatrix} \in \mathbb{R}^6 \qquad \boldsymbol{\omega} \in \mathbb{R}^3 \qquad \mathbf{v} \in \mathbb{R}^3$$

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

- $\mathbf{SE}(3)$ is a smooth manifold, i.e. a Lie group
- 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)

Very Useful Tutorial on Lie Groups

Lie Groups for 2D and 3D Transformations

Ethan Eade

Updated May 20, 2017*

1 Introduction

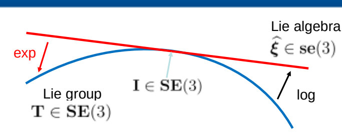
This document derives useful formulae for working with the Lie groups that represent transformations in 2D and 3D space. A Lie group is a topological group that is also a smooth manifold, with some other nice properties. Associated with every Lie group is a Lie algebra, which is a vector space discussed below. Importantly, a Lie group and its Lie algebra are intimately related, allowing calculations in one to be mapped usefully into the other.

This document does not give a rigorous introduction to Lie groups, nor does it discuss all of the mathematical details of Lie groups in general. It does attempt to provide enough information that the Lie groups representing spatial transformations can be employed usefully in robotics and computer vision.

Here are the Lie groups that this document addresses:

http://ethaneade.com/lie.pdf

Recap: Exponential Map of SE(3)

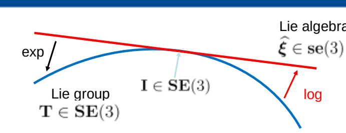


- The exponential map finds the transformation matrix for a twist:

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

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

Recap: Logarithm Map of SE(3)



- The logarithm maps twists to transformation matrices:

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

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

Working with Twist Coordinates

- Let's define the following notation:
  - Inversion 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: $\xi(\mathbf{T}) = (\log(\mathbf{T}))^\vee, \quad \mathbf{T}(\xi) = \exp(\widehat{\xi})$
  - Pose inversion: $\xi^{-1} = \log(\mathbf{T}(\xi)^{-1}) = -\xi$
  - Pose concatenation: $\xi_1 \oplus \xi_2 = (\log(\mathbf{T}(\xi_2)\,\mathbf{T}(\xi_1)))^\vee$
  - Pose difference: $\xi_1 \ominus \xi_2 = (\log(\mathbf{T}(\xi_2)^{-1}\,\mathbf{T}(\xi_1)))^\vee$

Optimization with Twist Coordinates

- Twists provide a minimal local representation without singularities
- Since $\mathbf{SE}(3)$ is a smooth manifold, we can decompose transformations in each optimization step into the transformation itself and an infinitesimal increment

$$\mathbf{T}(\xi) = \mathbf{T}(\xi)\exp\left(\widehat{\delta\xi}\right) = \mathbf{T}(\delta\xi \oplus \xi)$$

- We can then optimize an energy function $E(\xi_i, \delta\xi)$ in order to estimate the pose increment $\delta\xi$, e.g., using Gradient descent

$$\delta\xi^* = \mathbf{0} - \eta\nabla_{\delta\xi}E(\xi_i, \delta\xi)$$

$$\mathbf{T}(\xi_{i+1}) = \mathbf{T}(\xi_i)\exp\left(\widehat{\delta\xi^*}\right)$$

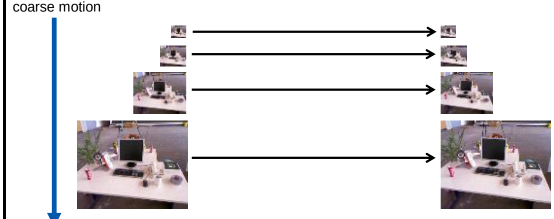
Algorithm: Direct RGB-D Visual Odometry

Input: RGB-D image sequence $I_{0:t}, Z_{0:t}$
Output: aggregated camera poses $\mathbf{T}_{0:t}$

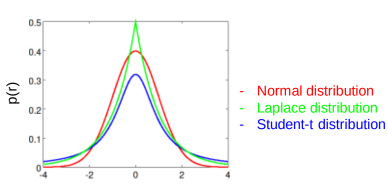
Algorithm:
For each current RGB-D image I_k, Z_k :

1. Estimate relative camera motion $\mathbf{T}_k^{k-1}$ towards the previous RGB-D frame using direct image alignment
2. Concatenate estimated camera motion with previous frame camera pose to obtain current camera pose estimate $\mathbf{T}_k = \mathbf{T}_{k-1}\mathbf{T}_k^{k-1}$

Practical Considerations: Coarse-To-Fine Optimization

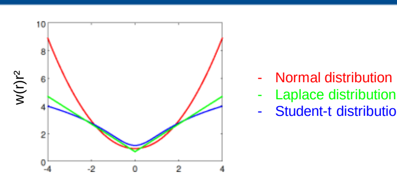


Practical Considerations: Residual Distributions



- Practical advice
  - Gaussian noise assumption on photometric residuals oversimplifies
  - Outliers (occlusions, motion, etc.): Residuals are distributed with more mass on the larger values

Optimizing Non-Gaussian Measurement Noise



- Accommodating different noise distributions
  - Can we change the residual distribution in least squares optimization?
  - For specific types of distributions: yes!
  - Iteratively reweighted least squares: Reweight residuals in each iteration

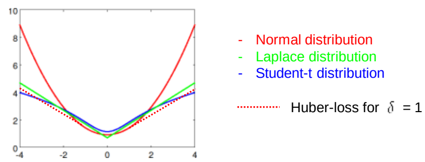
$$E(\xi) = \sum_{\mathbf{y}\in\Omega} w(r(\mathbf{y},\xi))\,\frac{r(\mathbf{y},\xi)^2}{\sigma_I^2}$$

Laplace distribution:
$w(r(\mathbf{y},\xi)) = |r(\mathbf{y},\xi)|^{-1}$

Huber Loss

- Huber-loss „switches“ between Gaussian (locally at mean) and Laplace distribution

$$\|r\|_\delta = \begin{cases} \frac{1}{2}\|r\|_2^2 & \text{if } \|r\|_2 \leq \delta \\ \delta\left(\|r\|_1 - \frac{1}{2}\delta\right) & \text{otherwise} \end{cases}$$



Summary

- Direct image alignment avoids manually designed keypoints, can use all available image information
- Direct visual odometry
  - Dense RGB-D odometry by direct image alignment with measured depth
  - Direct image alignment for monocular cameras requires depth estimation from temporal stereo
- Direct image alignment as non-linear least squares problem
  - Linearization of the residuals requires a coarse-to-fine optimization scheme
  - Gaussian distribution on pose can be obtained
  - SE(3) Lie algebra provides an elegant way of motion representation for gradient-based optimization
  - Iteratively reweighted least squares allows for wider set of residual distributions than Gaussians

Topics of This Lecture

- Recap: Direct Methods for Visual Odometry
  - Direct image alignment
  - Lie group se(3) and the exponential map
  - Practical considerations
- Visual SLAM
  - Motivation
  - Probabilistic formulation
- Online SLAM Methods
  - Tracking-and-Mapping
  - Filter-based SLAM
  - EKF SLAM

What is Visual SLAM ?

- SLAM stands for Simultaneous Localization and Mapping
  - Estimate the pose of the camera in a map, and simultaneously
  - Reconstruct the environment map
- Visual SLAM (VSLAM): SLAM with vision sensors
- Loop-closure: Revisiting a place allows for drift compensation

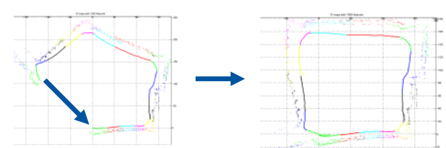
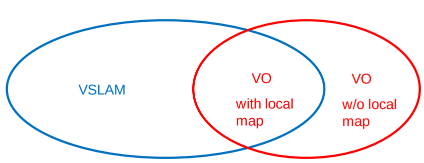


image source: [Clemente et al., RSS 2007]

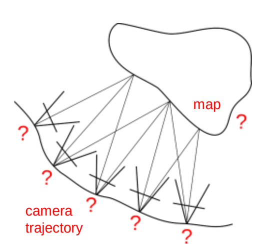
What is Visual SLAM ?



- Global and local optimization methods
  - Global: bundle adjustment, pose-graph optimization. Offline!
  - Local: incremental tracking-and-mapping approaches, visual odometry with local maps. Online! Often designed for real-time.
  - Hybrids: Real-time local SLAM + global optimization in a slower parallel process (e.g., LSD-SLAM)

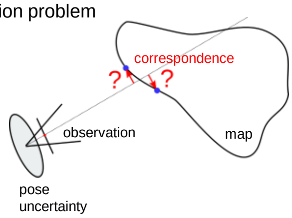
Why is SLAM Difficult?

- Chicken-and-egg problem
  - Camera trajectory and map are unknown and need to be estimated from observations
  - Accurate localization requires an accurate map
  - Accurate mapping requires accurate localization

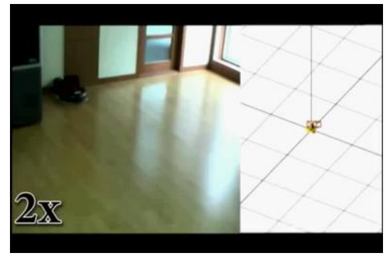


Why is SLAM Difficult?

- Challenging data association problem
  - Correspondences between observations and the map are unknown
  - Wrong correspondences can lead to divergence of trajectory/map estimates
  - Important to model uncertainties of observations and estimates in a probabilistic formulation of the SLAM problem

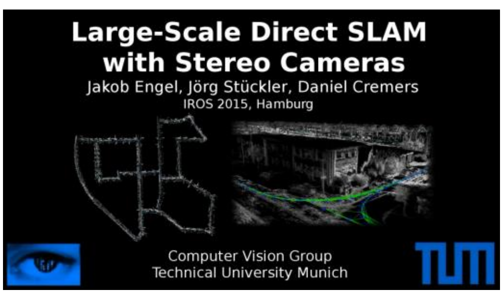


SLAM Examples



Jeong and Lee. CV-SLAM a new ceiling vision-based SLAM technique. IROS 2005

SLAM Examples

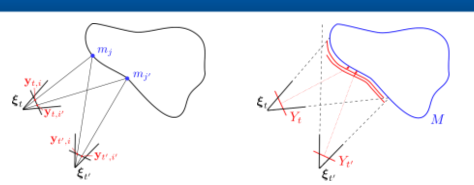


Engel, Stückler, Cremers. Large-scale direct SLAM with stereo cameras. IROS 2015

Definition of Visual SLAM

- Visual SLAM
  - The process of simultaneously estimating the egomotion of an object and the environment map using only inputs from visual sensors on the object
- **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 $I_{0:t} = \{I_0, \ldots, I_t\}$, $Z_{0:t} = \{Z_0, \ldots, Z_t\}$
  - Robotics: **control inputs** $U_{1:t}$
- **Output**:
  - Camera pose estimates $\mathbf{T}_t \in \mathbf{SE}(3)$ in world reference frame. For convenience, we also write $\xi_t = \xi(\mathbf{T}_t)$
  - Environment map M

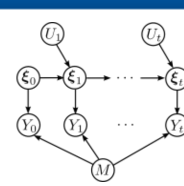
Map Observations in Visual SLAM



With Y_t we denote observations of the environment map in image I_t, e.g.,
- Indirect point-based method: $Y_t = \{\mathbf{y}_{t,1}, \ldots, \mathbf{y}_{t,N}\}$ (2D or 3D image points)
- Direct RGB-D method: $Y_t = \{I_t, Z_t\}$ (all image pixels)
- …

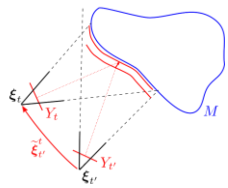
- Involves data association to map elements $M = \{m_1, \ldots, m_S\}$
  - We denote correspondences by $c_{t,i} = j, \ 1 \le i \le N, \ 1 \le j \le S$

Probabilistic Formulation of Visual SLAM



- SLAM posterior probability: $p(\xi_{0:t}, M \mid Y_{0:t}, U_{1:t})$
- Observation likelihood: $p(Y_t \mid \xi_t, M)$
- State-transition probability: $p(\xi_t \mid \xi_{t-1}, U_t)$

Relative Pose Observations in Visual SLAM



- SLAM can also be formulated using relative pose observations $Y^t_{t'} = \tilde{\xi}^t_{t'} = \xi\left(\tilde{T}^t_{t'}\right)$ deduced from images I_t,
  - Indirect point-based motion estimate
  - Direct image alignment result
  - …

Summary

- Simultaneous Localization and Mapping is a chicken-and-egg problem
  - Unknown map and trajectory
  - Unknown correspondences
- Probabilistic formulation to explicitly model uncertainties in observations and estimates
- Map and relative pose observations
- Many use-cases, e.g., in
  - Robotics
  - Augmented/virtual reality
  - Photogrammetry
  - Autonomous driving

Topics of This Lecture

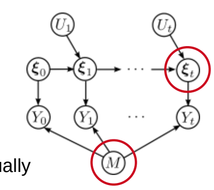
- Recap: Direct Methods for Visual Odometry
  - Direct image alignment
  - Lie group se(3) and the exponential map
  - Practical considerations
- Visual SLAM
  - Motivation
  - Probabilistic formulation
- Online SLAM Methods
  - Tracking-and-Mapping
  - Filter-based SLAM
  - EKF SLAM

Online SLAM Methods

- Marginalize out previous poses

$$p\left(\xi_t, M \mid Y_{0:t}, U_{1:t}\right) =$$

$$\int \cdots \int p\left(\xi_{0:t}, M \mid Y_{0:t}, U_{1:t}\right) d\xi_{t-1} \ldots d\xi_0$$



- Poses can be marginalized individually in a recursive way
- Variants:
  - Tracking-and-Mapping: Alternating pose and map estimation
  - Probabilistic filters, e.g., EKF-SLAM

Tracking-and-Mapping

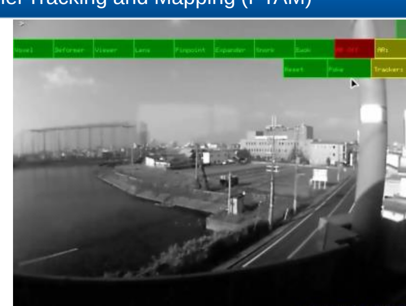
- Alternating optimization of pose estimation and mapping

$$p\left(\xi_t \mid Y_{0:t}, U_{1:t}, M\right) \qquad p\left(M \mid Y_{0:t}, U_{1:t}, \xi_t\right)$$

Tracking Mapping

- Examples
  - Semi-dense direct visual odometry
  - KinectFusion
  - Parallel Tracking and Mapping (PTAM) [G. Klein and D. Murray, Parallel Tracking and Mapping for Small AR Workspaces, ISMAR 2007]

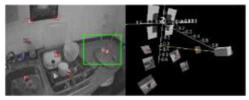
Parallel Tracking and Mapping (PTAM)



G. Klein and D. Murray, Parallel Tracking and Mapping for Small AR Workspaces, ISMAR 2007

Filter-based Visual SLAM

- Historical development
  - ~2000: Structure-and-motion with EKF [Chiuso et al., Structure from Motion Causally Integrated Over Time, TPAMI 2002]
    - Limited to a small set of high-contrast keypoints and no revisiting of keypoints after a loop, evaluated at small scale (object on a turntable, short trajectories)
  - ~2004: A.J. Davison's MonoSLAM
    - EKF-SLAM with loop-closing,
    - Real-time SLAM in small workspaces (room-scale), AR/robotics demos



  - ~2004-~2010: Various extensions to improve scalability, robustness, accuracy
- Filter-based approaches are also popular for sensor-fusion, e.g., visual-inertial odometry/SLAM

Recap: Bayesian Filter

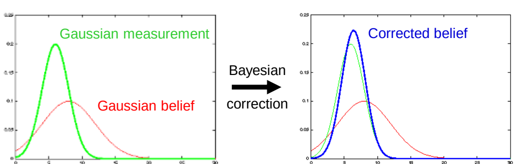
Prediction

$$p(X_t \mid Y_{0:t-1}, U_{1:t}) = \int p(X_t \mid X_{t-1}, U_t)\, p(X_{t-1} \mid Y_{0:t-1}, U_{1:t-1})\, dX_{t-1}$$

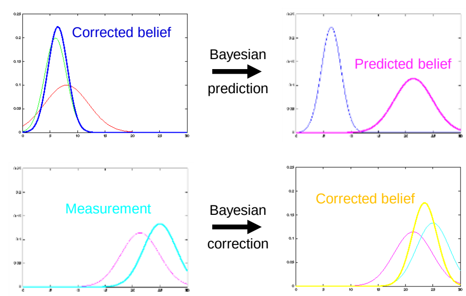
Correction

$$p(X_t \mid Y_{0:t}, U_{1:t}) = \eta\, p(Y_t \mid X_t)\, p(X_t \mid Y_{0:t-1}, U_{1:t})$$

Recap: Kalman Filter



Recap: Kalman Filter



Recap: Extended Kalman Filter (EKF)

- Non-linear models with Gaussian noise

$$\mathbf{x}_t = g(\mathbf{x}_{t-1}, \mathbf{u}_t) + \boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{d_t})$$
$$\mathbf{y}_t = h(\mathbf{x}_t) + \boldsymbol{\delta}_t, \boldsymbol{\delta}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{m_t})$$

- Prediction

$$\mathbf{x}_t^- = g(\mathbf{x}_{t-1}^+, \mathbf{u}_t)$$
$$\boldsymbol{\Sigma}_t^- = \mathbf{G}_t \boldsymbol{\Sigma}_{t-1}^+ \mathbf{G}_t^\top + \boldsymbol{\Sigma}_{d_t} \qquad \mathbf{G}_t = \nabla_{\mathbf{x}} g(\mathbf{x}, \mathbf{u}_t)|_{\mathbf{x}=\mathbf{x}_{t-1}^+}$$

- Correction

$$\mathbf{K}_t = \boldsymbol{\Sigma}_t^- \mathbf{H}_t^\top \left(\mathbf{H}_t \boldsymbol{\Sigma}_t^- \mathbf{H}_t^\top + \boldsymbol{\Sigma}_{m_t}\right)^{-1} \qquad \mathbf{H}_t = \nabla_{\mathbf{x}} h(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_t^-}$$
$$\mathbf{x}_t^+ = \mathbf{x}_t^- + \mathbf{K}_t \left(\mathbf{y}_t - h\left(\mathbf{x}_t^-\right)\right)$$
$$\boldsymbol{\Sigma}_t^+ = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t)\, \boldsymbol{\Sigma}_t^-$$

Topics of This Lecture

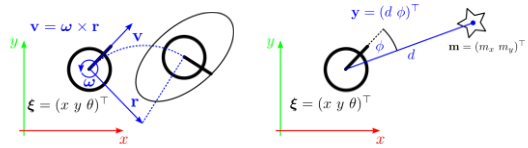
- Recap: Direct Methods for Visual Odometry
  - Direct image alignment
  - Lie group se(3) and the exponential map
  - Practical considerations
- Visual SLAM
  - Motivation
  - Probabilistic formulation
- Online SLAM Methods
  - Tracking-and-Mapping
  - Filter-based SLAM
  - EKF SLAM

SLAM with Extended Kalman Filters

- Detected keypoint y_i in an image observes „landmark“ position m_j in the map $M = \{m_1, \ldots, m_S\}$.
- Idea: Include landmarks into state variable

$$\mathbf{x}_t = \begin{pmatrix} \boldsymbol{\xi}_t \\ \mathbf{m}_{t,1} \\ \vdots \\ \mathbf{m}_{t,S} \end{pmatrix} \qquad \boldsymbol{\Sigma}_t = \begin{pmatrix} \Sigma_{t,\xi\xi} & \Sigma_{t,\xi m_1} & \cdots & \Sigma_{t,\xi m_S} \\ \Sigma_{t,m_1\xi} & \Sigma_{t,m_1 m_1} & \cdots & \Sigma_{t,m_1 m_S} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{t,m_S\xi} & \Sigma_{t,m_S m_1} & \cdots & \Sigma_{t,m_S m_S} \end{pmatrix} = \begin{pmatrix} \Sigma_{t,\xi\xi} & \Sigma_{t,\xi \mathbf{m}} \\ \Sigma_{t,\mathbf{m}\xi} & \Sigma_{t,\mathbf{mm}} \end{pmatrix}$$

EKF-SLAM in a 2D World



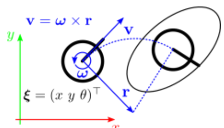
- For simplicity, let's assume
  - 3-DoF camera motion on a 2D plane
  - 2D range-and-bearing measurements of 2D landmarks
  - Only one measurement at a time
  - Known data association

2D EKF-SLAM State-Transition Model

- State/control variables

$$\boldsymbol{\xi}_t = (x_t\ y_t\ \theta_t)^\top \qquad \mathbf{m}_{t,j} = (m_{t,j,x}\ m_{t,j,y})^\top$$

$$\mathbf{u}_t = (v_t\ \omega_t)^\top = (\|\mathbf{v}\|_2\ \ \|\boldsymbol{\omega}\|_2)^\top$$



- State-transition model
  - Pose:

$$\boldsymbol{\xi}_t = g_\xi(\boldsymbol{\xi}_{t-1}, \mathbf{u}_t) + \boldsymbol{\epsilon}_{\xi,t} \qquad \boldsymbol{\epsilon}_{\xi,t} \sim \mathcal{N}(0, \Sigma_{d_t,\xi})$$

$$g_\xi(\boldsymbol{\xi}_{t-1}, \mathbf{u}_t) = \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta_{t-1} + \frac{v_t}{\omega_t}\sin(\theta_t + \omega_t \Delta t) \\ \frac{v_t}{\omega_t}\cos\theta_{t-1} - \frac{v_t}{\omega_t}\cos(\theta_t + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}$$

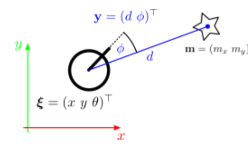
  - Landmarks: $\mathbf{m}_t = g_\mathbf{m}(\mathbf{m}_{t-1}) = \mathbf{m}_{t-1}$
  - Combined:

$$\mathbf{x}_t = g(\mathbf{x}_{t-1}, \mathbf{u}_t) + \boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{d_t}) \qquad g(\mathbf{x}_{t-1}, \mathbf{u}_t) = \begin{pmatrix} g_\xi(\boldsymbol{\xi}_{t-1}, \mathbf{u}_t) \\ g_\mathbf{m}(\mathbf{m}_{t-1}) \end{pmatrix} \qquad \boldsymbol{\Sigma}_{d_t} = \begin{pmatrix} \Sigma_{d_t,\xi} & 0 \\ 0 & 0 \end{pmatrix}$$

2D EKF-SLAM Observation Model

- State/measurement variables

$$\mathbf{y}_t = (d_t\ \phi_t)^\top \qquad \mathbf{m}_{t,j} = (m_{t,j,x}\ m_{t,j,y})^\top$$



- Observation model:

$$\mathbf{y}_t = h(\boldsymbol{\xi}_t, \mathbf{m}_{t,c_t}) + \boldsymbol{\delta}_t \qquad \boldsymbol{\delta}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{m_t})$$

$$h(\boldsymbol{\xi}_t, \mathbf{m}_{t,c_t}) = \begin{pmatrix} \left\|\mathbf{m}^{\mathrm{rel}}_{t,c_t}\right\|_2 \\ \operatorname{atan2}\left(\mathbf{m}^{\mathrm{rel}}_{t,c_t,y}, \mathbf{m}^{\mathrm{rel}}_{t,c_t,x}\right) \end{pmatrix}$$

$$\mathbf{m}^{\mathrm{rel}}_{t,c_t} := \mathbf{R}(-\theta_t)\left(\mathbf{m}_{t,c_t} - (x_t\ y_t)^\top\right)$$

State Initialization

- First frame:
  - Anchor reference frame at initial pose $\quad \mathbf{x}_0 = \mathbf{0}$
  - Set pose covariance to zero $\quad \boldsymbol{\Sigma}^-_{0,\xi\xi} = \mathbf{0}$
- New landmark:
  - Initial position unknown $\quad \boldsymbol{\Sigma}^-_{0,\xi\mathbf{m}} = \boldsymbol{\Sigma}^-_{0,\mathbf{m}\xi} = \mathbf{0}$
  - Initialize mean at zero
  - Initialize covariance to infinity (large value) $\quad \boldsymbol{\Sigma}^-_{0,\mathbf{mm}} = \infty\mathbf{I}$

Evolution of State Estimate on Prediction

- How is the state estimate modified on a state-transition?
- Recap: EKF Prediction

$$\mathbf{G}_t = \nabla_\mathbf{x} g(\mathbf{x}, \mathbf{u}_t)|_{\mathbf{x}=\mathbf{x}^+_{t-1}}$$

$$\mathbf{x}^-_t = g(\mathbf{x}^+_{t-1}, \mathbf{u}_t) \qquad \boldsymbol{\Sigma}^-_t = \mathbf{G}_t \boldsymbol{\Sigma}^+_{t-1} \mathbf{G}_t^\top + \boldsymbol{\Sigma}_{d_t} \qquad \mathbf{G}_{t,\xi} := \nabla_\xi g_\xi(\boldsymbol{\xi}, \mathbf{u}_t)|_{\xi=\xi^+_{t-1}}$$

↓

$$\mathbf{x}^-_t = \begin{pmatrix} g_\xi(\boldsymbol{\xi}^+_{t-1}, \mathbf{u}_t) \\ \mathbf{m}^+_{t-1} \end{pmatrix}$$

only the mean pose is updated!

$$\mathbf{x}_t = \begin{pmatrix} \boldsymbol{\xi}_t \\ \mathbf{m}_{t,1} \\ \vdots \\ \mathbf{m}_{t,S} \end{pmatrix}$$

Evolution of State Estimate on Prediction

- How is the state estimate modified on a state-transition?
- Recap: EKF Prediction

$G_t = \nabla_x g(x, u_t)|_{x=x^+_{t-1}}$

$x^-_t = g(x^+_{t-1}, u_t) \qquad \Sigma^-_t = G_t \Sigma^+_{t-1} G^\top_t + \Sigma_{d_t} \qquad G_{t,\xi} := \nabla_\xi g_\xi(\xi, u_t)|_{\xi=\xi^+_{t-1}}$

$$\begin{pmatrix} G_{t,\xi} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} \Sigma^+_{t-1,\xi\xi} & \Sigma^+_{t-1,\xi m} \\ \Sigma^+_{t-1,m\xi} & \Sigma^+_{t-1,mm} \end{pmatrix} \begin{pmatrix} G^\top_{t,\xi} & 0 \\ 0 & I \end{pmatrix} + \begin{pmatrix} \Sigma_{d_t,\xi} & 0 \\ 0 & 0 \end{pmatrix} =$$

$$\begin{pmatrix} G_{t,\xi}\Sigma^+_{t-1,\xi\xi}G^\top_{t,\xi} + \Sigma_{d_t,\xi} & G_{t,\xi}\Sigma^+_{t-1,\xi m} \\ \Sigma^+_{t-1,m\xi}G^\top_{t,\xi} & \Sigma^+_{t-1,mm} \end{pmatrix}$$

covariances are transformed to the new pose!

$$\Sigma_t = \begin{pmatrix} \Sigma_{t,\xi\xi} & \Sigma_{t,\xi m_1} & \cdots & \Sigma_{t,\xi m_S} \\ \Sigma_{t,m_1\xi} & \Sigma_{t,m_1m_1} & \cdots & \Sigma_{t,m_1m_S} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{t,m_S\xi} & \Sigma_{t,m_Sm_1} & \cdots & \Sigma_{t,m_Sm_S} \end{pmatrix}$$

Evolution of State Estimate on Correction

- How is the state estimate modified on a landmark measurement?
- Recap: EKF Correction

$K_t = \Sigma^-_t H^\top_t \left(H_t \Sigma^-_t H^\top_t + \Sigma_{m_t}\right)^{-1} \qquad H_t = \nabla_x h(x)|_{x=x^-_t}$

$x^+_t = x^-_t + K_t\left(y_t - h\left(x^-_t\right)\right)$

$\Sigma^+_t = (I - K_t H_t)\Sigma^-_t$

- How do correlations propagate onto mean and covariance through the Kalman gain?

Evolution of State Estimate on Correction

- Let's have a closer look at the Kalman gain

$K_t = \Sigma^-_t H^\top_t \left(H_t \Sigma^-_t H^\top_t + \Sigma_{m_t}\right)^{-1} \qquad H_t = \nabla_x h(x)|_{x=x^-_t}$

- The Jacobian of the observation function is only non-zero for the pose and the measured landmark:

$H_t = \begin{pmatrix} H_{t,\xi} & 0 & \ldots & 0 & H_{t,m_{c_t}} & 0 & \ldots & 0 \end{pmatrix}$

$H_{t,\xi} = \nabla_\xi h\left(\xi, m_{t,c_t}\right)|_{\xi=\xi^-_t} \qquad H_{t,m_{c_t}} = \nabla_{m_{c_t}} h\left(\xi_t, m_{c_t}\right)|_{m_{c_t}=m^-_{t,c_t}}$

Evolution of State Estimate on Correction

- Let's have a closer look at the Kalman gain

$K_t = \Sigma^-_t H^\top_t \left(H_t \Sigma^-_t H^\top_t + \Sigma_{m_t}\right)^{-1} \qquad H_t = \nabla_x h(x)|_{x=x^-_t}$

- The matrix $H_t \Sigma^-_t H^\top_t$ only involves covariances between pose and the measured landmark:

$H_t \Sigma^-_t H^\top_t$

$= H_{t,\xi}\Sigma^-_{t,\xi\xi}H^\top_{t,\xi} + H_{t,m_{c_t}}\Sigma^-_{t,m_{c_t}\xi}H^\top_{t,\xi} + H_{t,\xi}\Sigma^-_{t,\xi m_{c_t}}H^\top_{t,m_{c_t}} + H_{t,m_{c_t}}\Sigma^-_{t,m_{c_t}m_{c_t}}H^\top_{t,m_{c_t}}$

$$\Sigma_t = \begin{pmatrix} \Sigma^-_{t,\xi\xi} & \Sigma^-_{t,\xi m_1} & \cdots & \Sigma^-_{t,\xi m_{c_t}} & \cdots & \Sigma^-_{t,\xi m_S} \\ \Sigma^-_{t,m_1\xi} & \Sigma^-_{t,m_1m_1} & \cdots & \Sigma^-_{t,m_1m_{c_t}} & \cdots & \Sigma^-_{t,m_1m_S} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \Sigma^-_{t,m_{c_t}\xi} & \Sigma^-_{t,m_{c_t}m_1} & \cdots & \Sigma^-_{t,m_{c_t}m_{c_t}} & \cdots & \Sigma^-_{t,m_{c_t}m_S} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \Sigma^-_{t,m_S\xi} & \Sigma^-_{t,m_Sm_1} & \cdots & \Sigma^-_{t,m_Sm_{c_t}} & \cdots & \Sigma^-_{t,m_Sm_S} \end{pmatrix}$$

Evolution of State Estimate on Correction

$K_t = \Sigma^-_t H^\top_t \left(H_t \Sigma^-_t H^\top_t + \Sigma_{m_t}\right)^{-1} \qquad H_t = \nabla_x h(x)|_{x=x^-_t}$

- The matrix $\Sigma^-_t H^\top_t$ stacks the covariances between the pose/the measured landmark and all state variables (pose+landmarks)

$$\Sigma^-_t H^\top_t = \begin{pmatrix} \Sigma^-_{t,\xi\xi}H^\top_{t,\xi} + \Sigma^-_{t,\xi m_{c_t}}H^\top_{t,m_{c_t}} \\ \Sigma^-_{t,m_1\xi}H^\top_{t,\xi} + \Sigma^-_{t,m_1m_{c_t}}H^\top_{t,m_{c_t}} \\ \vdots \\ \Sigma^-_{t,m_S\xi}H^\top_{t,\xi} + \Sigma^-_{t,m_Sm_{c_t}}H^\top_{t,m_{c_t}} \end{pmatrix}$$

$$\Sigma_t = \begin{pmatrix} \Sigma^-_{t,\xi\xi} & \Sigma^-_{t,\xi m_1} & \cdots & \Sigma^-_{t,\xi m_{c_t}} & \cdots & \Sigma^-_{t,\xi m_S} \\ \Sigma^-_{t,m_1\xi} & \Sigma^-_{t,m_1m_1} & \cdots & \Sigma^-_{t,m_1m_{c_t}} & \cdots & \Sigma^-_{t,m_1m_S} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \Sigma^-_{t,m_{c_t}\xi} & \Sigma^-_{t,m_{c_t}m_1} & \cdots & \Sigma^-_{t,m_{c_t}m_{c_t}} & \cdots & \Sigma^-_{t,m_{c_t}m_S} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \Sigma^-_{t,m_S\xi} & \Sigma^-_{t,m_Sm_1} & \cdots & \Sigma^-_{t,m_Sm_{c_t}} & \cdots & \Sigma^-_{t,m_Sm_S} \end{pmatrix}$$

Evolution of State Estimate on Correction

- Hence, the Kalman gain distributes information onto all state dimensions that are correlated with the pose or the measured landmark

$K_t = \Sigma^-_t H^\top_t \left(H_t \Sigma^-_t H^\top_t + \Sigma_{m_t}\right)^{-1}$

$$\Sigma^-_t H^\top_t = \begin{pmatrix} \Sigma^-_{t,\xi\xi}H^\top_{t,\xi} + \Sigma^-_{t,\xi m_{c_t}}H^\top_{t,m_{c_t}} \\ \Sigma^-_{t,m_1\xi}H^\top_{t,\xi} + \Sigma^-_{t,m_1m_{c_t}}H^\top_{t,m_{c_t}} \\ \vdots \\ \Sigma^-_{t,m_S\xi}H^\top_{t,\xi} + \Sigma^-_{t,m_Sm_{c_t}}H^\top_{t,m_{c_t}} \end{pmatrix}$$

- The correction step updates all state dimensions in the mean that are correlated with the pose or measured landmark

$x^+_t = x^-_t + K_t\left(y_t - h\left(x^-_t\right)\right)$

Evolution of State Estimate on Correction

- How is the state covariance updated in the correction step?

$$\boldsymbol{\Sigma}_t^+ = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\,\boldsymbol{\Sigma}_t^-$$

$$\begin{pmatrix} \mathbf{K}_{t,\xi}\mathbf{H}_{t,\xi} & 0 & \dots & 0 & \mathbf{K}_{t,\xi}\mathbf{H}_{t,\mathbf{m}_{c_t}} & 0 & \dots & 0 \\ \mathbf{K}_{t,\mathbf{m}_1}\mathbf{H}_{t,\xi} & 0 & \dots & 0 & \mathbf{K}_{t,\mathbf{m}_1}\mathbf{H}_{t,\mathbf{m}_{c_t}} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{K}_{t,\mathbf{m}_S}\mathbf{H}_{t,\xi} & 0 & \dots & 0 & \mathbf{K}_{t,\mathbf{m}_S}\mathbf{H}_{t,\mathbf{m}_{c_t}} & 0 & \dots & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\Sigma}_{t,\xi\xi}^- & \boldsymbol{\Sigma}_{t,\xi\mathbf{m}_1}^- & \cdots & \boldsymbol{\Sigma}_{t,\xi\mathbf{m}_{c_t}}^- & \cdots & \boldsymbol{\Sigma}_{t,\xi\mathbf{m}_S}^- \\ \boldsymbol{\Sigma}_{t,\mathbf{m}_1\xi}^- & \boldsymbol{\Sigma}_{t,\mathbf{m}_1\mathbf{m}_1}^- & \cdots & \boldsymbol{\Sigma}_{t,\mathbf{m}_1\mathbf{m}_{c_t}}^- & \cdots & \boldsymbol{\Sigma}_{t,\mathbf{m}_1\mathbf{m}_S}^- \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\Sigma}_{t,\mathbf{m}_{c_t}\xi}^- & \boldsymbol{\Sigma}_{t,\mathbf{m}_{c_t}\mathbf{m}_1}^- & \cdots & \boldsymbol{\Sigma}_{t,\mathbf{m}_{c_t}\mathbf{m}_{c_t}}^- & \cdots & \boldsymbol{\Sigma}_{t,\mathbf{m}_{c_t}\mathbf{m}_S}^- \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{\Sigma}_{t,\mathbf{m}_S\xi}^- & \boldsymbol{\Sigma}_{t,\mathbf{m}_S\mathbf{m}_1}^- & \cdots & \boldsymbol{\Sigma}_{t,\mathbf{m}_S\mathbf{m}_{c_t}}^- & \cdots & \boldsymbol{\Sigma}_{t,\mathbf{m}_S\mathbf{m}_S}^- \end{pmatrix}$$

- Covariance change for a landmark that is not the measured landmark:

$$\boldsymbol{\Sigma}_{t,\mathbf{m}_1\mathbf{m}_{c_t}}^+ =$$

$$\mathbf{K}_{t,\mathbf{m}_1} = \left(\boldsymbol{\Sigma}_{t,\mathbf{m}_1\xi}^-\mathbf{H}_{t,\xi}^\top + \boldsymbol{\Sigma}_{t,\mathbf{m}_1\mathbf{m}_{c_t}}^-\mathbf{H}_{t,\mathbf{m}_{c_t}}^\top\right)\left(\mathbf{H}_t\boldsymbol{\Sigma}_t^-\mathbf{H}_t^\top + \boldsymbol{\Sigma}_{m_t}\right)^{-1}$$

non-zero!

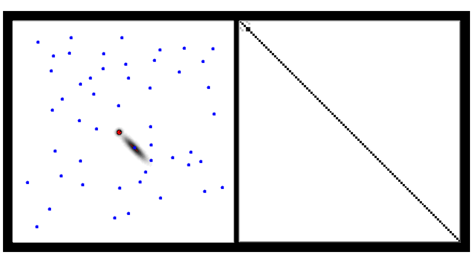
Evolution of State Estimate on Correction

- The correction step updates all state dimensions in the state covariance that correlate with the pose or measured landmark

$$\boldsymbol{\Sigma}_t^+ = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\,\boldsymbol{\Sigma}_t^-$$

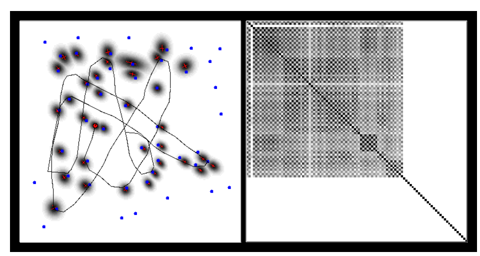
- Since all landmarks are correlated with pose, all landmark correlations with the measured landmark get updated
- Hence, all state variables become correlated: The state covariance is dense!
- Measurement information propagates on all landmarks along the trajectory

Example Evolution of the Covariance



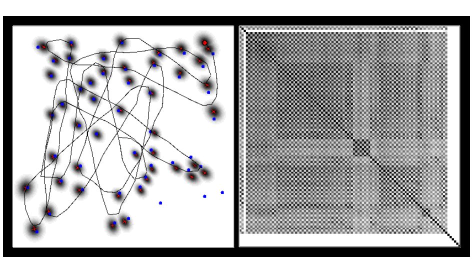
Pose and map Correlation matrix

Example Evolution of the Covariance



Pose and map Correlation matrix

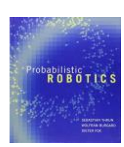
Example Evolution of the Covariance



Pose and map Correlation matrix

References and Further Reading

- Probabilistic Robotics textbook



Probabilistic Robotics, S. Thrun, W. Burgard, D. Fox, MIT Press, 2005