TTK4250

Week 10

Recursive SLAM and SLAM-like problems

Edmund Førland Brekke

24. October 2024

Recap from last week

Nominal state

- This is what we predict in dead reckoning.
- Basically a nonlinear ODE.

Error state

- We only propagate the covariance of the error state.
- Normally of lower dimensionality than nominal state.
- Neglecting small terms yields LTV system.

ESKF update and reset

- Estimate the error state and update posterior covariance.
- Inject into nominal state and reset error state.
- Adjust covariance because new error state is relative to new nominal state.

Potential causes of divergence

- Poor initialization.
- Nonlinearities fail to hide in linear noise sources.
- Insufficient bias modeling.
- Bugs and poor ODE solvers.
- Lack of observability?

Lever arm compensation

So far we have tacitly assumed that both the IMU and the GNSS antenna are located at the center of gravity (COG).

Offset of the IMU

- If the IMU is far from COG then rotations of the vehicle may be measured as centripital accelerations.
- ☹ Difficult to model (need derivatives of accelerations).
- ⇒ Keep the IMU as close to the COG as possible!

Offset of the GNSS antenna

- It is seldom possible to place the GNSS antenna at the COG.
- ⇒ Must take the lever arm from the IMU to the GNSS antenna into account in the measurement update.

$$\nu_{\text{GNSS}} = \mathbf{z}_{\text{GNSS}} - \rho - \mathrm{R}\mathbf{a}$$
$$\mathbf{H} = \begin{bmatrix} \mathbf{I} & -\mathrm{RS}(\mathbf{a}) & \mathbf{0} \end{bmatrix}$$

IMU misalignment

Mounting error

Represented by rotation matrix R_M.

Orthogonality error (IMU axes not perfectly orthogonal to each other)

- Represented by matrix $\mathbf{O}_a$ for the accelerometer.
- Represented by matrix $\mathbf{O}_g$ for the gyro.

Scale error

- Represented by diagonal matrix $\mathbf{D}_a$ for the accelerometer.
- Represented by diagonal matrix $\mathbf{D}_g$ for the gyro.

$$
\begin{aligned}
\mathbf{a}_m =& \mathbf{D}_a \mathbf{O}_a \mathrm{R}_M \mathbf{a}_t^b + \mathbf{a}_{bt} + \mathbf{a}_n \\
\mathbf{a}^b =& \mathrm{R}_M^{\mathrm{T}} \mathbf{O}_a^{-1} \mathbf{D}_a^{-1} (\mathbf{a}_m - \mathbf{a}_{bt} - \mathbf{a}_n) = \mathbf{S}_a (\mathbf{a}_m - \mathbf{a}_{bt} - \mathbf{a}_n) \\
\boldsymbol{\omega}_m =& \mathbf{D}_g \mathbf{O}_g \mathrm{R}_M \boldsymbol{\omega}_t^b + \boldsymbol{\omega}_{bt} + \boldsymbol{\omega}_n \\
\boldsymbol{\omega}^b =& \mathrm{R}_M^{\mathrm{T}} \mathbf{O}_g^{-1} \mathbf{D}_g^{-1} (\boldsymbol{\omega}_m - \boldsymbol{\omega}_{bt} - \boldsymbol{\omega}_n) = \mathbf{S}_g (\boldsymbol{\omega}_m - \boldsymbol{\omega}_{bt} - \boldsymbol{\omega}_n)
\end{aligned}
$$

Outline

SLAM and SLAM-like problems

Problems where a robot wants to know something about where it is relative to a stationary environment.

Localization.

Where is the robot relative to known beacons or landmarks?

Mapping.

Robot knows where it is (from e.g. GPS) but does know what is around it.

SLAM.

Robot does not know where it is nor what is around it. Wants to know where it is.

Visual odometry.

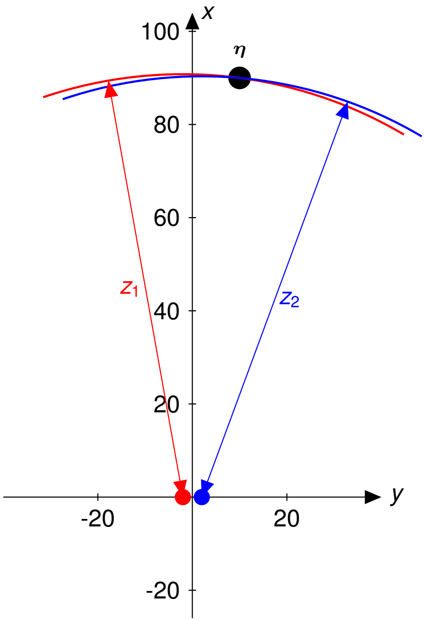
Robot does not know where it is nor what is around it. Wants to know its velocity.

Generalized target tracking.

Robot does not know where it is nor what is around it. Objects around it have unknown motion. Wants to know the whereabouts of the objects relative to itself.

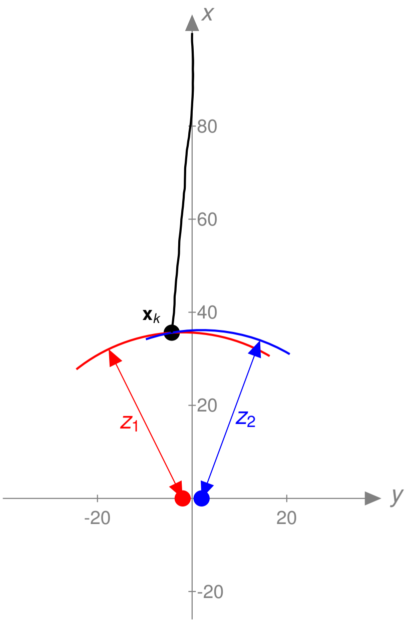
Outline

Example of Localization



Range-only localization using two or more UWB beacons could aid an autonomous ferry during docking operations.

Recursive model for range-only localization



Let the measurement model be

$$\mathbf{z}_k = \begin{bmatrix} \sqrt{x^2 + (-2 - y)^2} \\ \sqrt{x^2 + (2 - y)^2} \end{bmatrix} + \mathbf{w}_k$$

where the measurement noise is

$$\mathbf{w}_k \sim \mathcal{N}\left(\mathbf{0}, \mathrm{diag}([1, 1])\right).$$

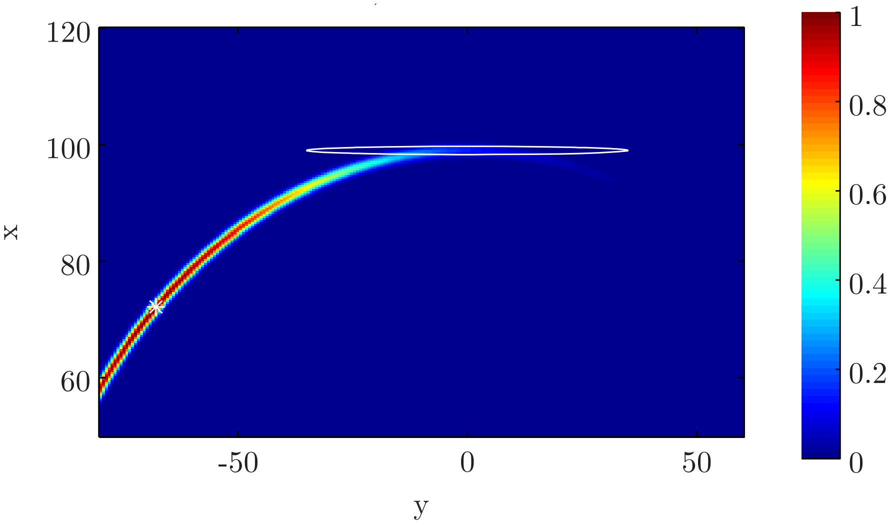
Assume a CV model with state vector $\mathbf{x} = [x, y, v_x, v_y]^\mathsf{T}$:

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_2 \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \mathbf{0} \\ \mathbf{I}_2 \end{bmatrix} \mathbf{v}$$

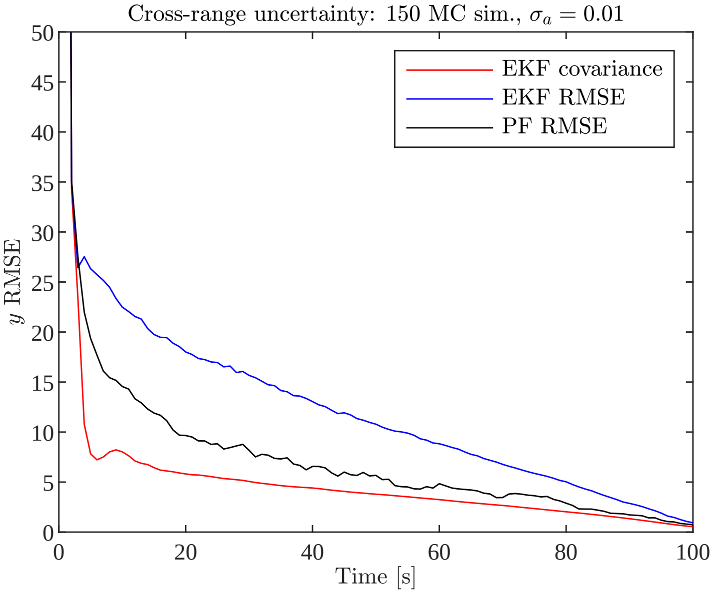
where the process noise is

$$\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \sigma_a^2 \mathbf{I}_2 \delta(t - \tau)).$$

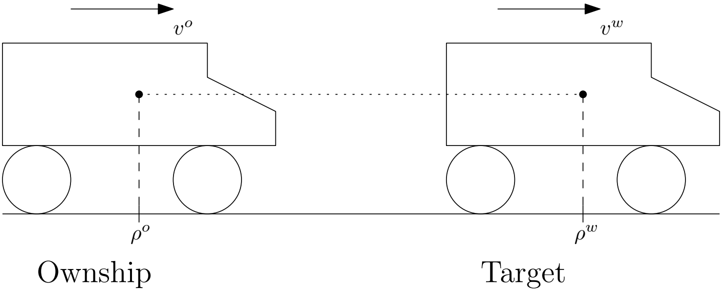
Range-only localization: Likelihood vs Cartesian uncertainty



Some simulation results



Another example: Target tracking with navigation uncertainty



$$\dot{\boldsymbol{\eta}} = \mathbf{A}^o\boldsymbol{\eta} + \mathbf{B}u + \mathbf{G}^o\mathbf{n}^o \qquad \dot{\mathbf{x}}^w = \mathbf{A}\mathbf{x}^w + \mathbf{G}\mathbf{n}$$

$$\Rightarrow \boldsymbol{\eta}_k = \boldsymbol{\Psi}\boldsymbol{\eta}_{k-1} + \mathbf{u}_k + \mathbf{v}_k^o \qquad \mathbf{x}_k^w = \boldsymbol{\Phi}\mathbf{x}_{k-1}^w + \mathbf{v}_k$$

$$\mathbf{y}_k = \mathbf{H}^o\boldsymbol{\eta}_k + \mathbf{w}_k^o \qquad \mathbf{z}_k = \mathbf{H}(\mathbf{x}_k^w - \mathbf{E}\boldsymbol{\eta}_k) + \mathbf{w}_k$$

Where u is ownship acceleration, **v** and **w** are ownship and target noise, and **y** and **z** are ownship and target measurements.

Optimal and suboptimal solutions

Filtering

The optimal solution to the filtering part of target tracking with navigation uncertainty is found by generalizing SLAM to deal with moving landmarks. Possible simplifications:

- Maintain correlations without allowing the target measurements to affect the ownship state.
- Ignore correlations, but allow uncertainty in ownship to affect the target state.
- Ignore the ownship uncertainty altogether.

See also the TTK4250 exam 2019 and Brekke & Wilthil: “Suboptimal Kalman Filters for Target Tracking with Navigation Uncertainty in One Dimension”, IEEE Aerospace Conf. 2017.

Data association

- Data association probabilities will in general be integrals over ownship uncertainty.
- Simplifications needed to express them as products over individual terms from the measurements.

See also Brekke et al. “A novel formulation of the Bayes recursion for single-cluster filtering”, IEEE Aerospace Conf. 2014.

Outline

What is SLAM?

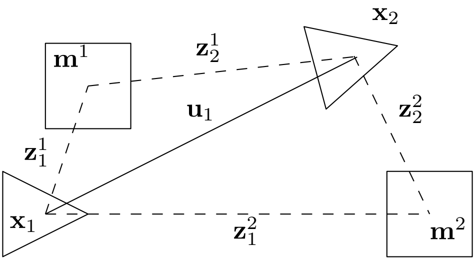
- Simultaneous localization and mapping (SLAM) is the process of building a map of the surroundings of a rover, and locate the rover in this map.
- Has been implemented on robots, underwater and airborne systems as well as planetary rovers.

Figur: Examples of SLAM applications. Left: Stanley, winner of the DARPA Grand Challenge. Right: Swarm exploration using SLAM.

SLAM: some definitions

- **Pose:** Position + orientation of the robot.
- **Online/Recursive SLAM:** Estimate map + last pose.
- **Full SLAM:** Estimate map + trajectory of all historical poses including current pose as well as map.
- **Loop closure:** When the robot revisits a place where it can observe landmarks it has not seen for a while.
- **Feature-based SLAM:** The map is represented as a collection of points (landmarks).
- **Dense SLAM:** More dense map representations such as **occupancy grids** are used.
- **Range-bearing SLAM:** Range and bearing measuremnts of landmarks available.
- **Range-only SLAM:** Only range measurements (from e.g. transponders) available.
- **Bearing-only SLAM:** Only bearing measurements (from e.g. camera) available.
- **Scan-matching:** Estimation of pose displacement from successive sensor scans.

Notation - SLAM as a pure filtering problem



- $\boldsymbol{\eta}_k$ is the state vector at time k. It consists of pose and map.
- $\mathbf{x}_k$ is the pose of the robot at time k. For our purposes it consists of position ρ_k and heading ψ_k.
- $\mathbf{x}_{1:k}$ is the history of robot poses (trajectory).
- $\mathbf{u}_k$ is the control input (odometry) at time k. $\mathbf{u}_{1:k}$ is the history of control inputs.
- $\mathbf{m}^i$ is the position of landmark i. $\mathbf{m}$ is the concatenated vector of all landmarks.
- $\mathbf{z}_k^i$ is an observation of landmark i. $\mathbf{z}_k$ is the concatenated vector of all observations at time k. $\mathbf{z}_{1:k}$ is the history of observations.

Formulation of recursive probabilistic SLAM

We want the joint distribution of both $\mathbf{x}_k$ and $\mathbf{m}$ from the observations and inputs:

$$p(\mathbf{x}_k, \mathbf{m} | \mathbf{z}_{1:k}, \mathbf{u}_{1:k})$$

This is computed recursively using a *motion model* and an *observation model*:

$$p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_k) \qquad\qquad p(\mathbf{z}_k | \mathbf{x}_k, \mathbf{m})$$

As usual, apply the total probability theorem and Bayes' rule to predict and correct the estimate:

$$p(\mathbf{x}_k, \mathbf{m} | \mathbf{z}_{1:k-1}, \mathbf{u}_{1:k}) = \int p(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_k) \times \\ p(\mathbf{x}_{k-1}, \mathbf{m} | \mathbf{z}_{1:k-1}, \mathbf{u}_{1:k-1}) \mathrm{d}\mathbf{x}_{k-1}$$

$$p(\mathbf{x}_k, \mathbf{m} | \mathbf{z}_{1:k}, \mathbf{u}_{0:k}) = \frac{p(\mathbf{z}_k | \mathbf{x}_k, \mathbf{m}) p(\mathbf{x}_k, \mathbf{m} | \mathbf{z}_{1:k-1}, \mathbf{u}_{1:k})}{p(\mathbf{z}_k | \mathbf{z}_{1:k-1}, \mathbf{u}_{1:k})}$$

Outline

EKF-SLAM: Motion model

We consider a planar scenario with pose vector $\mathbf{x}_k = [x_k, y_k, \psi_k]^\mathsf{T}$.

- Let the odometry vector be of the form

$$\mathbf{u}_k = \begin{bmatrix} u_k \\ v_k \\ \varphi_k \end{bmatrix} = \begin{bmatrix} \text{forward displacement} \\ \text{sideways displacement} \\ \text{heading change} \end{bmatrix}.$$

- We express the process model for the pose as

$$\mathbf{x}_k = f_{\mathbf{x}}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{v}_k = \mathbf{x}_{k-1} \oplus \mathbf{u}_k + \mathbf{v}_k$$

where the "$\oplus$" operator now is given by

$$\begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \psi_{k-1} \end{bmatrix} \oplus \begin{bmatrix} u_k \\ v_k \\ \varphi_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + u_k \cos \psi_{k-1} - v_k \sin \psi_{k-1} \\ y_{k-1} + u_k \sin \psi_{k-1} + v_k \cos \psi_{k-1} \\ \psi_{k-1} + \varphi_k \end{bmatrix}.$$

- There is no need to specify a process model for the landmarks.

EKF-SLAM: Measurement model

Let $\boldsymbol{\rho}_k$ denote position, so that the pose vector can be decomposed as $\mathbf{x}_k = [\boldsymbol{\rho}_k^\mathsf{T}, \psi_k]^\mathsf{T}$.

- For planar range-bearing SLAM, the measurement model is given by

$$\mathbf{z}_k = \mathbf{h}(\boldsymbol{\eta}_k) + \mathbf{w}_k = \begin{bmatrix} \mathbf{h}(\mathbf{x}_k, \mathbf{m}^i) \\ \vdots \\ \mathbf{h}(\mathbf{x}_k, \mathbf{m}^m) \end{bmatrix} + \mathbf{w}_k$$

  where the state-to-measurement-mapping consists of m landmark contributions, each given by

$$\mathrm{h}(\mathrm{x}_k, \mathrm{m}^i) = \mathrm{c2p}\left(\mathbf{R}(-\psi)(\mathrm{m}^i - \boldsymbol{\rho}_k)\right) = \begin{bmatrix} \|\mathbf{m}^i - \boldsymbol{\rho}_k\| \\ \angle \mathrm{R}(-\psi)(\mathbf{m}^i - \boldsymbol{\rho}_k) \end{bmatrix}.$$

- The measurement noise for reach landmark observation is i.i.d. Gaussian with covariance

$$\mathbf{R}^i = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}.$$

  The covariance of $\mathbf{w}_k$ is then $\mathbf{R} = \mathbf{I}_m \otimes \mathbf{R}^i$.

EKF-SLAM: Prediction

The EKF prediction is given by

$$\begin{aligned} \hat{\boldsymbol{\eta}}_{k|k-1} &= \mathbf{f}_{\boldsymbol{\eta}}(\hat{\boldsymbol{\eta}}_{k-1}, \mathbf{u}_k) \\ \mathbf{P}_{k|k-1} &= \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^{\mathsf{T}} + \mathbf{G}\mathbf{Q}\mathbf{G}^{\mathsf{T}} \end{aligned}$$

where

$$\mathbf{f}_{\boldsymbol{\eta}}\left(\begin{bmatrix}\mathbf{x}_{k-1} \\ \mathbf{m}\end{bmatrix}, \mathbf{u}_k\right) = \begin{bmatrix} f_{\mathbf{x}}(\mathbf{x}_{k-1}, \mathbf{u}_k) \\ \mathbf{m} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{k-1} \oplus \mathbf{u}_k \\ \mathbf{m} \end{bmatrix}$$

and

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{\mathbf{x}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{2m} \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \mathbf{I}_3 \\ \mathbf{0}_{2m\times 3} \end{bmatrix}, \quad \mathbf{F}_{\mathbf{x}} = \left.\frac{\partial f_{\mathbf{x}}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1}}$$

EKF-SLAM: Update

The EKF update is given by

$$
\begin{aligned}
\mathbf{S}_k &= \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^{\mathsf{T}} + \mathbf{R} \\
\mathbf{W} &= \mathbf{P}_{k|k-1}\mathbf{H}^{\mathsf{T}}\mathbf{S}_k^{-1} \\
\boldsymbol{\nu}_k &= \mathbf{z}_k - \mathbf{h}(\boldsymbol{\eta}_k) \\
\hat{\boldsymbol{\eta}}_k &= \hat{\boldsymbol{\eta}}_{k|k-1} + \mathbf{W}\boldsymbol{\nu}_k \\
\mathbf{P}_k &= (\mathbf{I} - \mathbf{W}\mathbf{H})\mathbf{P}_{k|k-1}
\end{aligned}
$$

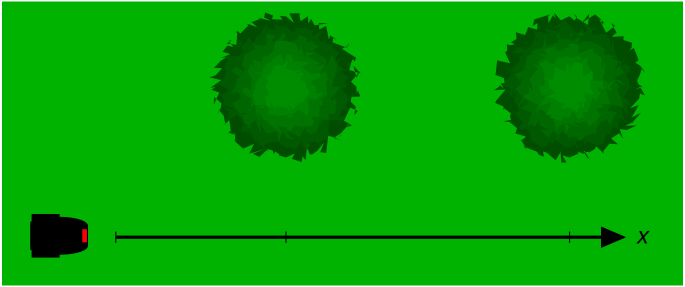
where

$$
\mathbf{H} = \begin{bmatrix} \mathbf{H}_{\mathbf{x}}^{1} & \mathbf{H}_{\mathbf{m}}^{1} & & \\ \vdots & & \ddots & \\ \mathbf{H}_{\mathbf{x}}^{m} & & & \mathbf{H}_{\mathbf{m}}^{m} \end{bmatrix}
$$

$$
\mathbf{H}_{\mathbf{x}}^{i} = \left.\frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{m}^{i})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}, \mathbf{m}=\hat{\mathbf{m}}_{k-1}} \quad \text{and} \quad \mathbf{H}_{\mathbf{m}}^{i} = \left.\frac{\partial \mathbf{h}(\mathbf{x}, \mathbf{m}^{i})}{\partial \mathbf{m}^{i}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}, \mathbf{m}=\hat{\mathbf{m}}_{k-1}}
$$

Denseness of the covariance matrix

Consider a scenario involving a lawn-mowing robot and two trees.

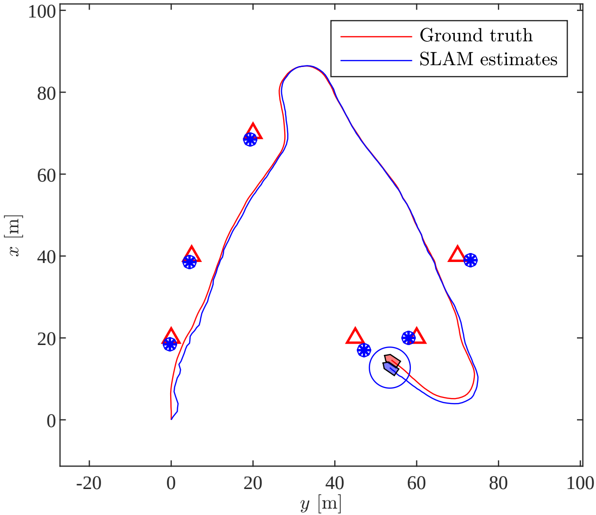


- Assume that $\mathbf{P}_{k|k-1}$ is block-diagonal (and thus **sparse**).
- Then the two trees are observed. The further to the right tree 1 is, the further to the right the lawn mower is. Similarly for tree 2.
- Thus there is a correlation between the lawn mower position and both tree positions, and also between the two tree positions.

$\Rightarrow$ After the measurement update, the covariance matrix is **dense**.

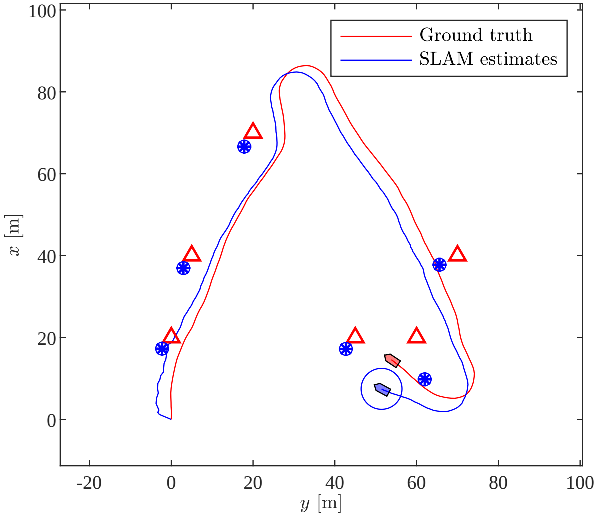
EKF-SLAM in action: Perfect odometry

The robot doesn't know that the odometry is perfect, though.



EKF-SLAM in action: Noisy odometry

Inconsistency becomes a major concern.



Improving the consistency of EKF-SLAM

Robocentric mapping

- Represent the landmarks in the body frame and not in the world frame.

+ Measurement update of landmark position becomes linear.

\- Need to use a nonlinear process model for the landmarks.

Observability-constrained Jacobians

- Planar range-bearing SLAM has an unobservable subspace of dimension 3.
- Linearization tricks the EKF into believe that the subspace has dimension 2.

$\Rightarrow$ The EKF makes use of fake information.

- A possible solution is first-estimate Jacobians: Use the very first estimates of the landmarks as linearization points. This breaks the destructive feedback cycle that makes the EKF diverge.

The Laplace approximation

Use curvature to get a better evaluation of the covariance matrix than the EKF gives.

The Laplace approximation

Rationale

We want our approximation of $p(\boldsymbol{\eta}_k \,|\, \mathbf{z}_{1:k})$ to match the shape of $p(\boldsymbol{\eta}_k \,|\, \mathbf{z}_{1:k})$ near its peak (where most of its probability mass presumably is).

Covariance and curvature

- Define $f(\boldsymbol{\eta}_k) = -\ln p(\boldsymbol{\eta}_k \,|\, \mathbf{z}_{1:k})$.
- Calculate the Hessian matrix H of $f(\boldsymbol{\eta}_k)$ with respect to $\boldsymbol{\eta}_k$.
- We use H^{-1} instead of $\mathbf{P}_k$ in the filter.

Improved state estimation

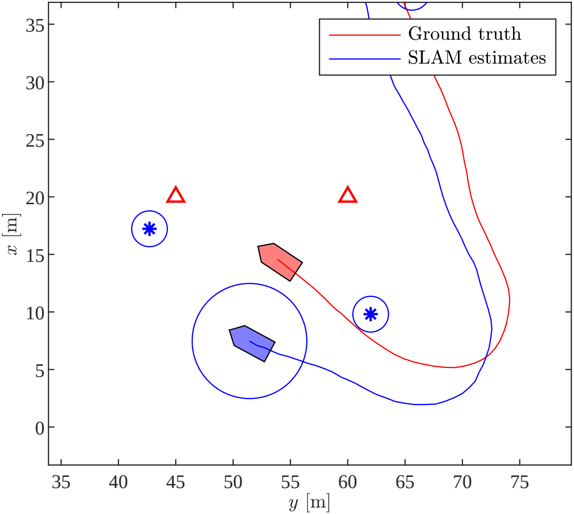
We use one or several Newton-Raphson iterations of the form

$$\hat{\boldsymbol{\eta}}_{k,(i+1)} = \hat{\boldsymbol{\eta}}_{k,(i)} - \mathrm{H}(\hat{\boldsymbol{\eta}}_{k,(i)})^{-1}\mathrm{D}(\hat{\boldsymbol{\eta}}_{k,(i)})^{\mathsf{T}}.$$

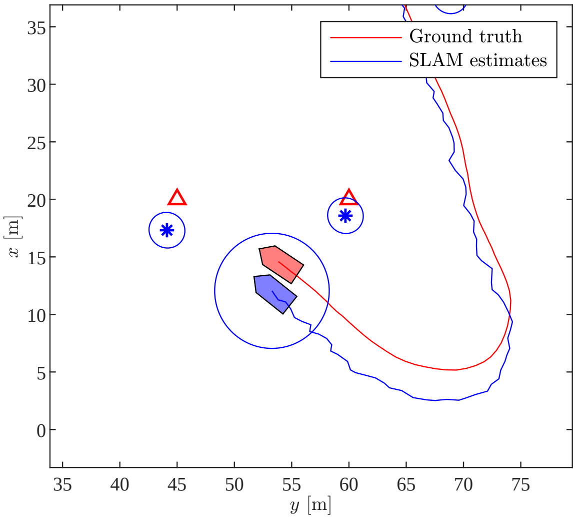
where $\mathrm{D}(\hat{\boldsymbol{\eta}}_{k,(i)})$ is the gradient of $f(\boldsymbol{\eta}_k)$.

Beware: Robocentric approaches and first-estimate Jacobians are the established solutions, while the Laplace approximation appears not to have been seriously considered by anyone else than myself.

Basic EKF: Final time step zoom-in



Laplace approximation: Final time step zoom-in



The Gauss-Newton method

Caveats of the Laplace approximation

- Hessian may not be positive definite.
- Evaluation and inversion of the Hessian may be computationally prohibitive.
- Would more global curvature evaluators such as UKF be preferable?

The Gauss-Newton method

We approximate the Hessian of a function $f(\boldsymbol{\eta}) = (\mathbf{a} - \mathbf{g}(\boldsymbol{\eta}))^{\mathsf{T}}\mathbf{C}(\mathbf{a} - \mathbf{g}(\boldsymbol{\eta}))$ as

$$\mathrm{H}(\boldsymbol{\eta}) \approx \mathbf{G}^{\mathsf{T}}(\boldsymbol{\eta})\mathbf{C}\mathbf{G}(\boldsymbol{\eta}).$$

This leads to the iterative optimization scheme

$$\hat{\boldsymbol{\eta}}_{k,(i+1)} = \hat{\boldsymbol{\eta}}_{k,(i)} - \left(\mathbf{G}^{\mathsf{T}}(\boldsymbol{\eta}_{k,(i)})\mathbf{C}\mathbf{G}(\boldsymbol{\eta}_{k,(i)})\right)^{-1}\mathbf{G}^{\mathsf{T}}(\boldsymbol{\eta}_{k,(i)})\mathbf{C}(\mathbf{a} - \mathbf{g}(\boldsymbol{\eta}_{k,(i)})).$$

It can be shown that this is actually the same as an EKF update.
$\Rightarrow$ Gauss-Newton is the same as an iterated EKF.

Outline

Data association in SLAM versus MTT

Why data association is easier in SLAM than in MTT

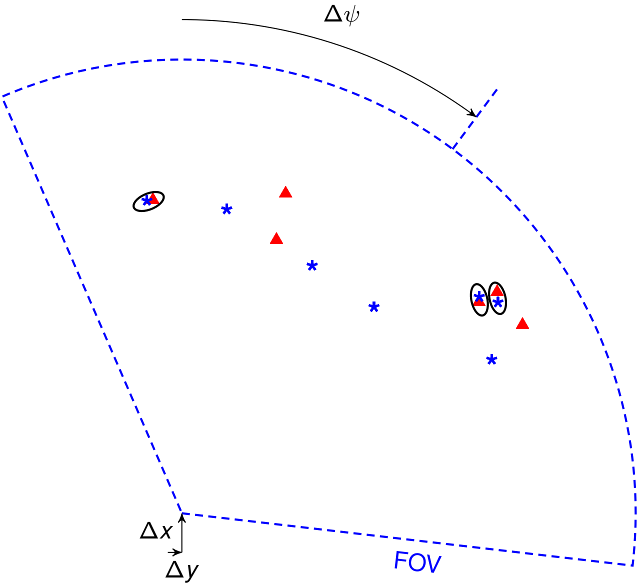
- Stationary landmarks as opposed to moving targets.
- It may be sufficient to only match a percentage of all the landmarks with their measurements.

Why data association is easier in MTT than in SLAM

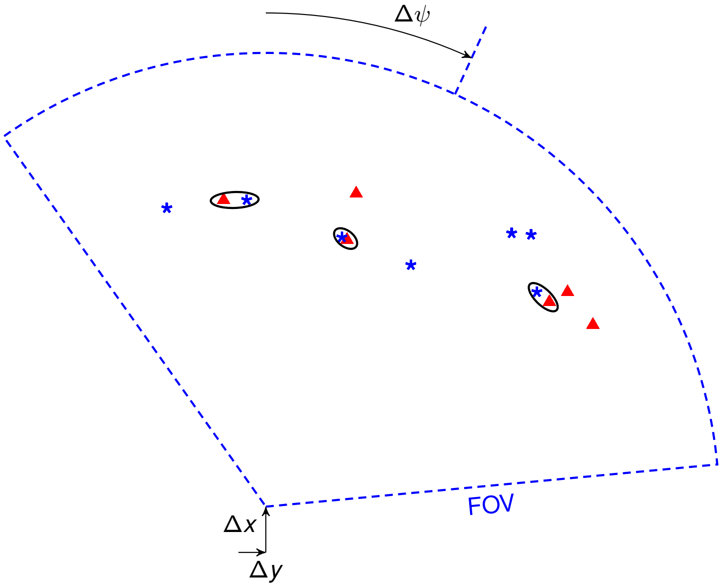
- The pose of the sensor is known.

$\Rightarrow$ We can decompose the data association problem into local clusters that are independent of each other.

Data association in feature-based SLAM



Data association in feature-based SLAM



Data association for EKF-SLAM - Strategies

Joint compatibility branch and bound (JCBB)

Find the maximal number of landmark-measurement correspondences that have a high enough likelihood.

- Deterministic depth first search.

RANSAC

Sample a minimal set of compatible correspondences and check how many more correspondences can be added.

- Stochastic iteration until better matches are considered too unlikely.

Probabilistic techniques

- Calculating posterior hypothesis probabilities (very expensive).
- Variations of JPDA and multi-dimensional assignment (questionable validity).
- Particle filters (e.g., FastSLAM).

Hypothesis representation in JCBB

Consider a case with 6 landmarks and 7 measurements. Possible hypotheses could be

$$\begin{Bmatrix} 2 & 3 & 4 \\ 7 & 4 & 2 \end{Bmatrix}, \begin{Bmatrix} 1 & 3 & 6 \\ 2 & 3 & 5 \end{Bmatrix}, \begin{Bmatrix} i_1 & i_2 & \cdots & i_n \\ j_1 & j_2 & \cdots & j_n \end{Bmatrix}$$

In the JPDA-style notation they would be

$$\begin{bmatrix} 0 & 7 & 4 & 2 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 2 & 0 & 3 & 0 & 0 & 5 \end{bmatrix}.$$

In Neira-Tardos style notation the would be

$$\begin{bmatrix} 0 & 4 & 0 & 3 & 0 & 0 & 2 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 3 & 0 & 6 & 0 & 0 \end{bmatrix}.$$

The innovation and covariance for a given hypothesis a

$$\boldsymbol{\nu}(a) = \begin{bmatrix} \mathbf{z}_k^{j_1} - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}, \hat{\mathbf{m}}_{k|k-1}^{i_1}) \\ \mathbf{z}_k^{j_2} - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}, \hat{\mathbf{m}}_{k|k-1}^{i_2}) \\ \vdots \\ \mathbf{z}_k^{j_n} - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}, \hat{\mathbf{m}}_{k|k-1}^{i_n}) \end{bmatrix}, \quad \mathbf{S}_k(a) = \begin{bmatrix} S_{i_1,i_1} & S_{i_1,i_2} & \cdots & S_{i_1,i_n} \\ S_{i_2,i_1} & S_{i_2,i_2} & \cdots & S_{i_2,i_n} \\ \vdots & \vdots & \ddots & \vdots \\ S_{i_n,i_1} & S_{i_n,i_2} & \cdots & S_{i_n,i_n} \end{bmatrix}$$

Individual versus joint compatibility

Individual compatibility

Landmark i is compatible with measurement j if

$$(\mathbf{z}^j - \mathbf{h}^i)^\mathsf{T} s_{ii}^{-1} (\mathbf{z}^j - \mathbf{h}^i) < \mathrm{chi2inv}(1-\alpha, n_\mathbf{z}).$$

Joint compatibility

The landmarks in the hypothesis a are compatible with the measurements in a if

$$\boldsymbol{\nu}^\mathsf{T}(a)\mathbf{S}_k^{-1}(a)\boldsymbol{\nu}(a) < \texttt{chi2inv}(1-\alpha, d)$$

where d is the dimension of $\boldsymbol{\nu}(a)$ and α is the required significance level.

Motivation for joint compatibility:

- We recall that correlations make the covariance ellipses narrower.

$\Rightarrow$ Even if $(\mathbf{m}^{i_1}, \mathbf{z}^{j_1})$ and $(\mathbf{m}^{i_2}, \mathbf{z}^{j_2})$ both are individually compatible, they are not necessarily jointly compatible.

JCBB pseudo code

```
Function JCBB_R(a,j) is
    if all observations are accounted for then
        if #(a) > #(a_best) then
            a_best ← a ;
        else
            if mahalanobis(a) < mahalanobis(a_best) then
                a_best ← a ;
            end
        end
    else
        for i ∈ {Landmarks compatible with measurement j} do
            if jointly_compatible([a, i]) then
                JCBB_R([a, i], j + 1) ;
            end
        end
        if ∃ enough more potential matches to beat a_best then
            JCBB_R([a, 0], j + 1) ;
        end
    end
    return a_best;
end
```

Outline

Matrix derivatives

Motivation

Since linearization and gradient-based search techniques are extensively used in estimation, we need a systematic approach to differentiation, that can provide chain rules and product rules for expressions involving vectors and matrices.

Differential of a scalar function

The differential $d\phi(c; u)$ of a scalar function $\phi(c)$ is a linear function of u whose slope is equal to the slope of $\phi(\cdot)$ at c:

$$d\phi(c; u) = u\frac{\mathrm{d}\phi(c)}{\mathrm{d}c}.$$

The theory of this lecture is mainly based on Magnus & Neudecker (1999): "Matrix Differential Calculus with Applications in Statistics and Econometrics", 2nd edition, Wiley.

See also `http://www2.stat.duke.edu/~zo2/shared/resources/matrixc1.pdf` and Brekke & Chitre (2015): "A multi-hypothesis solution to data association for the two-frame SLAM problem", IJRR, vol. 34, no. 1, pp. 43-63.

Matrix derivatives

Differential of a vector function

The differential $df(c; u)$ of a vector function $f(c)$ is a vector function of the vector u of the form

$$df(c; u) = A(c)u$$

where $A(c)$ is the matrix of partial derivatives of $f(\cdot)$ evaluated at c.

Differential of a matrix function

Let $F(C)$ be a matrix function of the form $F : \mathbb{R}^{\times q} \to \mathbb{R}^{m \times p}$. Let $A(C)$ be matrix containing partial derivatives of $\mathrm{vec}\, F(C)$ with respect to $\mathrm{vec}\, C$.

The differential of $F(\cdot)$ at C is the matrix function $dF(C; U)$ of U such that

$$\mathrm{vec}\, dF(C; U) = A(C)\mathrm{vec} U.$$

Jacobian matrices

- We refer to the matrix A as a Jacobian matrix.
- The vec operator enables generalization to matrix functions.
- The notation $\mathrm{D}f(x)$ or $\mathrm{D}F(X)$ is often used to denote the Jacobian.

Examples

Differential and Jacobian of a quadratic form

Let $\phi(x) = x^\mathsf{T} A x$. Then

$$\mathrm{d}\phi(x) = \mathrm{d}(x^\mathsf{T} A x) = (\mathrm{d}x)^\mathsf{T} A x + x^\mathsf{T} A\, \mathrm{d}x = ((\mathrm{d}x)^\mathsf{T} A x)^\mathsf{T} + x^\mathsf{T} A\, \mathrm{d}x = \underbrace{x^\mathsf{T}(A^\mathsf{T} + A)}_{\mathrm{D}\phi(x)}\, \mathrm{d}x$$

Differential and Jacobian of a determinant

Let $\phi(X) = |X|$. The differential of this matrix-to-scalar function is

$$\mathrm{d}\phi(x) = |X| \mathrm{tr}(X^{-1} \mathrm{d}X)$$

while its Jacobian is

$$\mathrm{D}\phi(X) = |X| (\mathrm{vec}(X^{-1})^\mathsf{T})^\mathsf{T}$$

Chain rule and product rule

The chain rule

Let f and g be two functions, possibly vector-valued or matrix-valued, and possibly with vectors or matrices as input. Define the composite function $h(x) = g(f(x))$. Assume that f is differentiable at c, and that g is differentiable at $b = f(x)$. Then

$$\mathrm{D}h(c) = \mathrm{D}g(b)\mathrm{D}f(c)$$

Example: Derivative of quadratic form

Let $g(x) = f(x)Af^{\mathsf{T}}(x)$. Then the Jacobian of g with respect to x is

$$\mathrm{D}g(x) = f^{\mathsf{T}}(x)(A + A^{\mathsf{T}})\mathrm{D}f(x)$$

Fackler's product rule

Let $f : \mathbb{R}^n \to \mathbb{R}^{m\times p}$ and $g : \mathbb{R}^n \to \mathbb{R}^{p\times q}$ be two matrix-valued functions. If $h(x) = f(x)g(x)$, the its Jacobian is

$$\mathrm{D}h(x) = \left(g(x)^{\mathsf{T}} \otimes \mathbf{I}_m\right) \mathrm{D}f(x) + (\mathbf{I}_q \otimes f(x))\, \mathrm{D}g(x)$$

EKF-SLAM Jacobians

Example: The Jacobian matrix $\mathbf{H}^i_\mathbf{x}$ in EKF-SLAM

This is the derivative of the function

$$\mathbf{h}(\mathbf{x}, \mathbf{m}^i) = \begin{bmatrix} \|\mathbf{u}\| \\ \angle\mathbf{u} \end{bmatrix} \quad \text{where} \quad \mathbf{u} = \mathrm{R}(-\psi)(\mathbf{m}^i - \boldsymbol{\rho}).$$

The chain rule yields

$$\mathrm{D}_\mathbf{x} \begin{bmatrix} \|\mathbf{u}\| \\ \angle\mathbf{u} \end{bmatrix} = \begin{bmatrix} \frac{1}{\|\mathbf{u}\|}\mathbf{u}^\mathsf{T} \\ \frac{1}{\|\mathbf{u}\|^2}\mathbf{u}^\mathsf{T}\mathrm{R}(\pi/2) \end{bmatrix} \mathrm{D}_\mathbf{x}\mathbf{u}$$

Linearity of the derivative yields $\mathrm{D}_{\boldsymbol{\rho}}\mathbf{u} = -\mathrm{R}(-\psi)$, and the product rule yields

$$\mathrm{D}_\psi\mathbf{u} = \mathrm{J}(\psi)(\mathbf{m}^i - \boldsymbol{\rho}) \;\; \text{where} \;\; \mathrm{J}(\psi) = \begin{bmatrix} -\sin\psi & -\cos\psi \\ \cos\psi & -\sin\psi \end{bmatrix}.$$

Stitching it all together yields the result

$$\mathbf{H}^i_\mathbf{x} = \begin{bmatrix} \frac{1}{\|\mathbf{m}^i - \boldsymbol{\rho}\|}(\mathbf{m}^i - \boldsymbol{\rho})^\mathsf{T} & 0 \\ \frac{1}{\|\mathbf{m}^i - \boldsymbol{\rho}\|^2}(\mathbf{m}^i - \boldsymbol{\rho})^\mathsf{T}\mathrm{R}(\pi/2) & 1 \end{bmatrix}$$