SLAM

Simultaneous Localization and Mapping

Simultaneous Localization And Mapping

- SLAM – Problematik
- SLAM – Arten
- SLAM – Methoden:
  - (E)KF SLAM
  - GraphSLAM
  - Fast SLAM

SLAM - Problematik

- Ziel: Karte und Pose bestimmen
- Fragen:
  - Wie sieht die Welt aus?
  - Wo bin ich?
- Bekannt:
  - Kontrolldaten, z.B. Odometriedaten (Daten des Vortriebssystems) z_t
  - Sensordaten (erfasste Landmarken) u_t

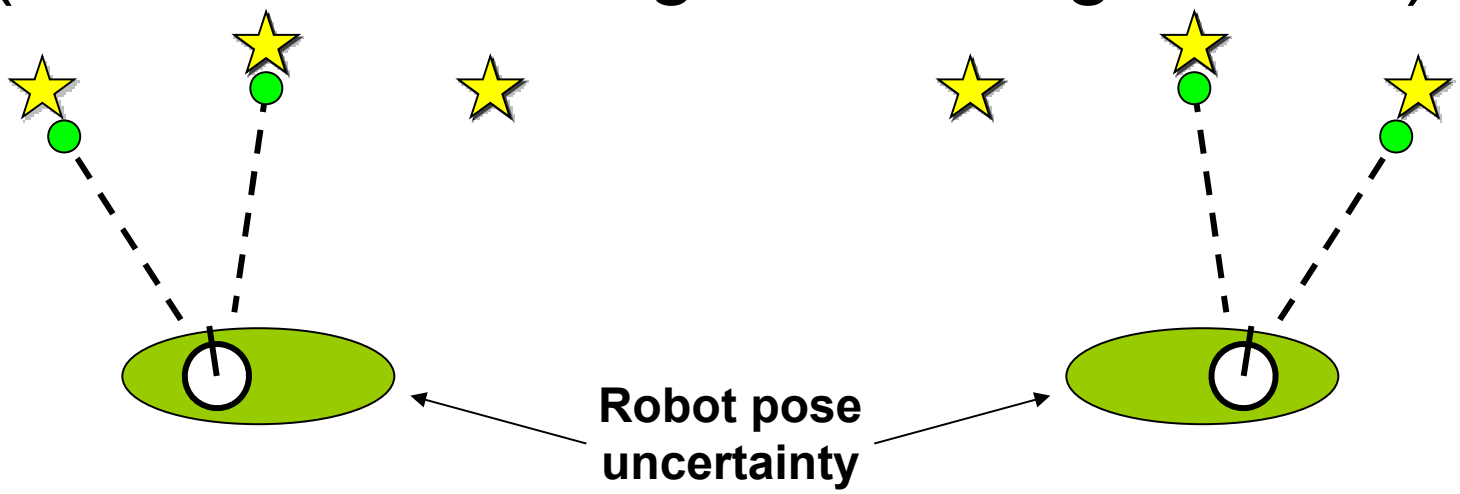
Probleme Odometrie

- Positionsmessung ungenau auf Grund von:
  - Bodenbeschaffenheit
  - Verschleiß
  - Ungleichmäßige Gewichtsverteilung
  - etc.

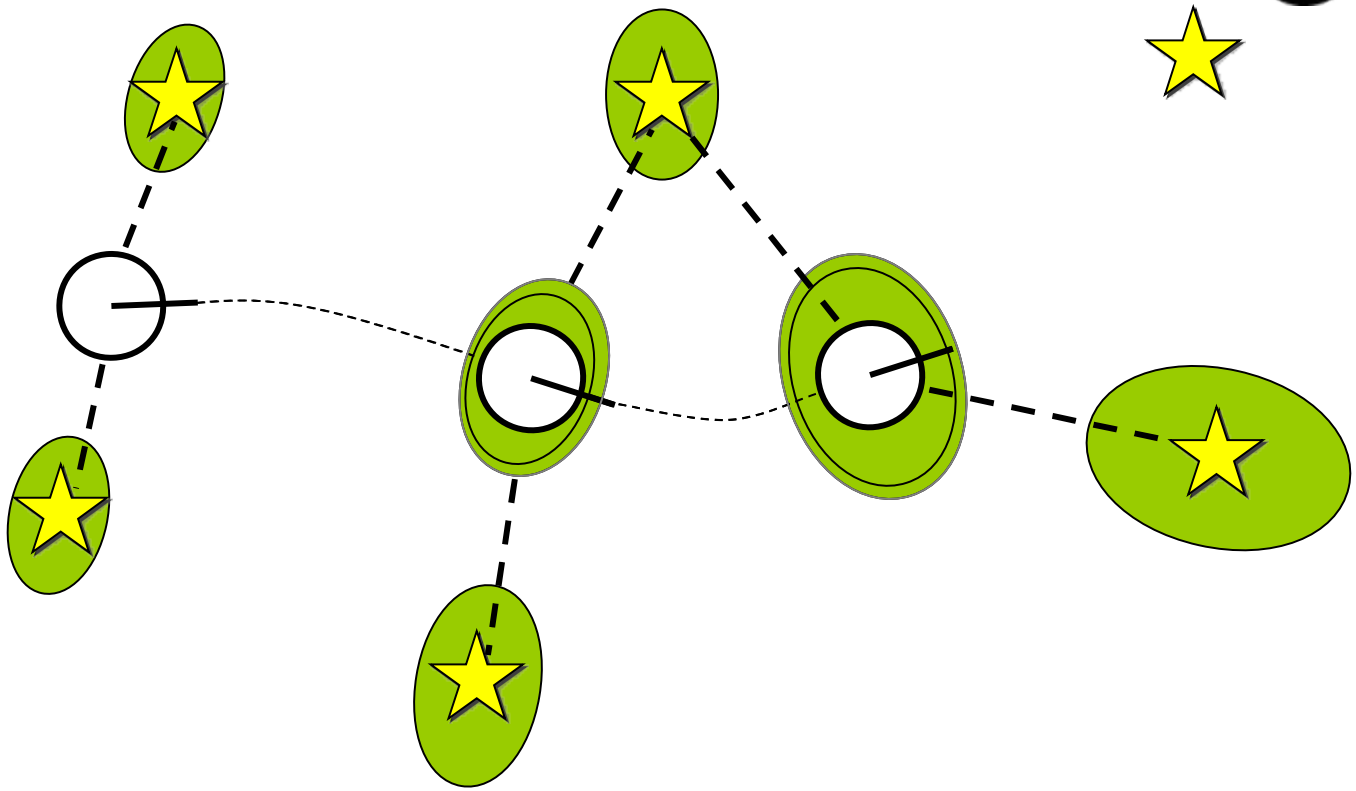
Probleme durch Sensordaten

- Ungenauigkeit der Sensoren

  Noise (Rauschen)
- Gleichartigkeit der Landmarken (Landmarken mit gleichen Signaturen)



Pfad- und Kartenfehler



Pfadfehler und Kartenfehler korrelieren!

Markov Annahme (Markov Assumption)

Statische Welt

- Alle Landmarken sind unbeweglich
- Außer dem Roboter gibt es keine weiteren bewegliche Elemente
- Aktueller Zustand ist eine komplette Zusammenfassung der Vergangenheit (und kann dadurch aus dieser ermittelt werden)

Einführung der Variablen

z_t	Messung (Beobachtung) zum Zeitpunkt t
u_t	Kontrolldaten für Zeitpunkt t -> alle Daten im Intervall]t-1,t]
$z_{t1:t2}$	Menge aller Messungen (Sensordaten) von Zeitpunkt t_1 bis t_2
$u_{t1:t2}$	Menge aller Kontrolldaten von Zeitpunkt t_1 bis t_2
x_t	State, der Zustand des Systems zum Zeitpunkt t (durch die Annahme der statischen Welt beschränkt sich der Zustand auf die Pose des Roboters)
m	Map, die Karte

Arten des SLAM-Problems

- Online SLAM: Schätzt aktuelle Pose und Karte

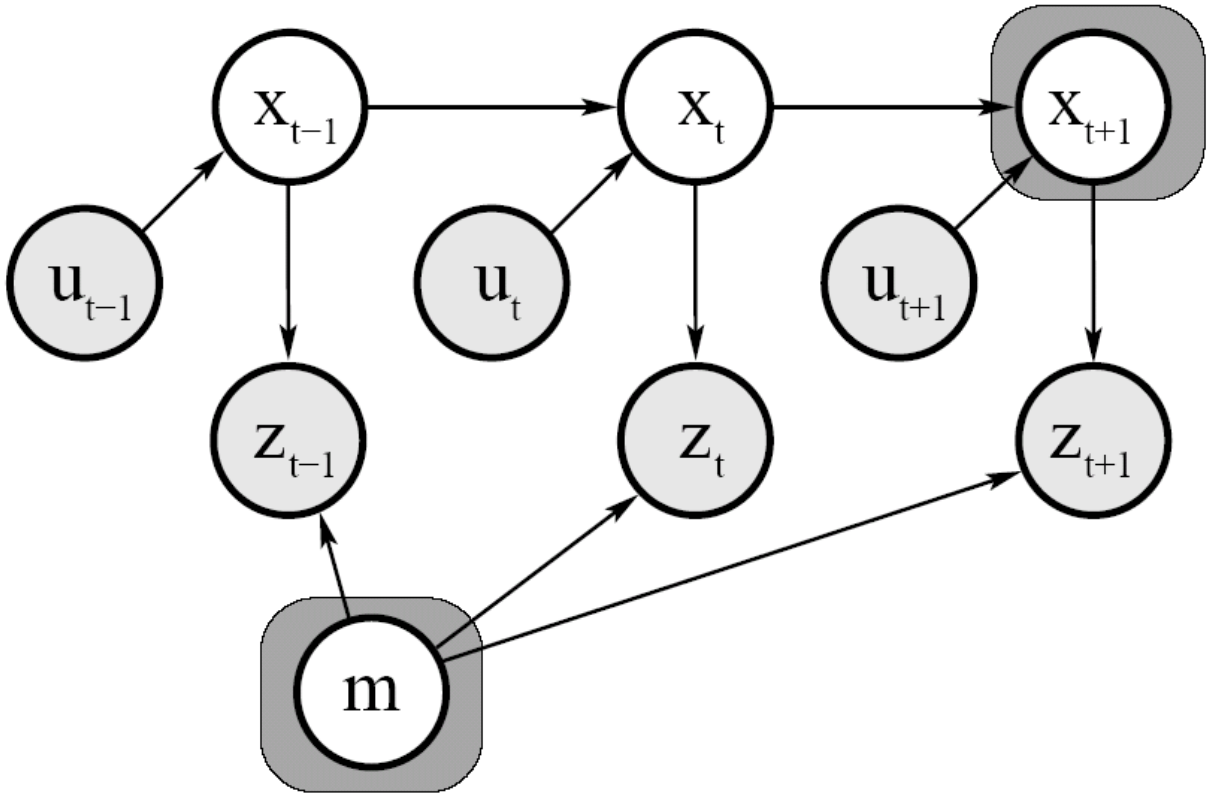
$$p(x_t, m \mid z_{1:t}, u_{1:t}) = \int \int \ldots \int p(x_{1:t}, m \mid z_{1:t}, u_{1:t})\, dx_1 dx_2 \ldots dx_{t-1}$$

Integrationen werden normal schrittweise durchgeführt

- Full SLAM: Schätzt den ganzen Pfad und Karte

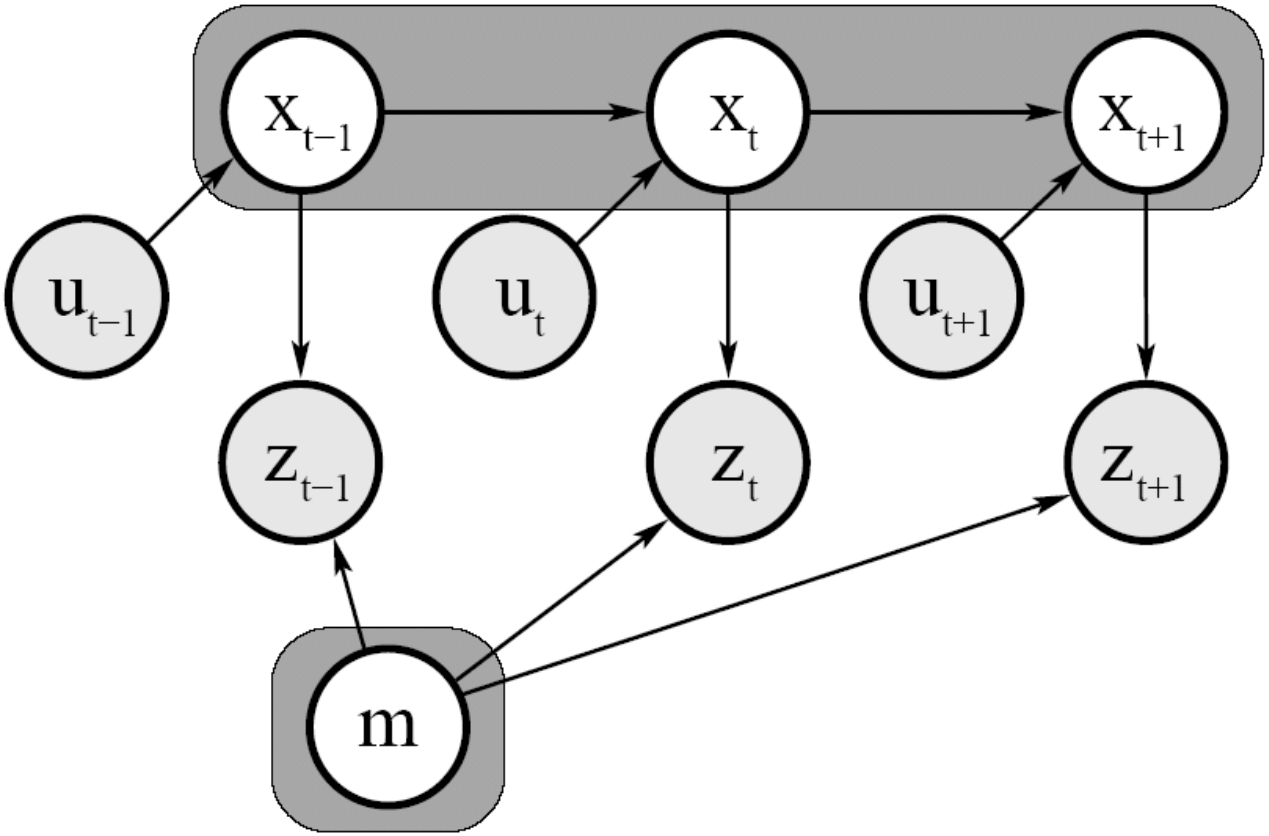
$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

Online SLAM: Graphisches Modell



$$p(x_t, m \mid z_{1:t}, u_{1:t}) = \int \int \ldots \int p(x_{1:t}, m \mid z_{1:t}, u_{1:t})\, dx_1 dx_2 \ldots dx_{t-1}$$

Full SLAM: Graphisches Modell



$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

(E)KF - SLAM

- Verwendet (Extended) Kalman Filter
- Online SLAM Verfahren
- Nutzt nur positive Landmarken-Sichtungen
- Annahme:
  - Rauschen (Noise) ist normal verteilt

Kalman Filter

- Bayes Filter
- Von Swerling (1958) und Kalman (1960) entwickelt
- Belief wird durch den Erwartungswert μ_t und die Kovarianz Σ_t repräsentiert

Kalman Filter: Annahmen

1. Markov Assumption
2. Wahrscheinlichkeitsfunktion der Zustandsübergänge ist linear mit normal verteiltem Rauschen
3. Wahrscheinlichkeitsfunktion der Messungen ist ebenfalls linear mit normal verteiltem Rauschen
4. Initial belief $bel(x_0)$ ist normal verteilt

Komponenten des Kalman Filters

A_t Matrix (nxn), die beschreibt, wie sich der Zustand, ohne Kontrolldaten oder Rauschen, von *t* nach *t-1* entwickelt.

B_t Matrix (nxm), die beschreibt, wie die Kontrolldaten u_t den Zustand von *t* nach *t-1* ändern.

C_t Matrix (kxn), die beschreibt, wie der Zustand x_t auf eine Beobachtung z_t abgebildet wird.

ε_t Zufallsvariablen, die das Rauschen abbilden. Sie sind unabhängig und normalverteilt mit Covarianz R_t and Q_t.

δ_t

Kalman Filter Algorithmus

1. Algorithm **Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):

1. Vorhersage:
2. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
3. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

Vorhersage auf Grund der Kontrolldaten

4. Korrektur:
5. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
6. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
7. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

Korrektur anhand der Beobachtungen (Sensordaten)

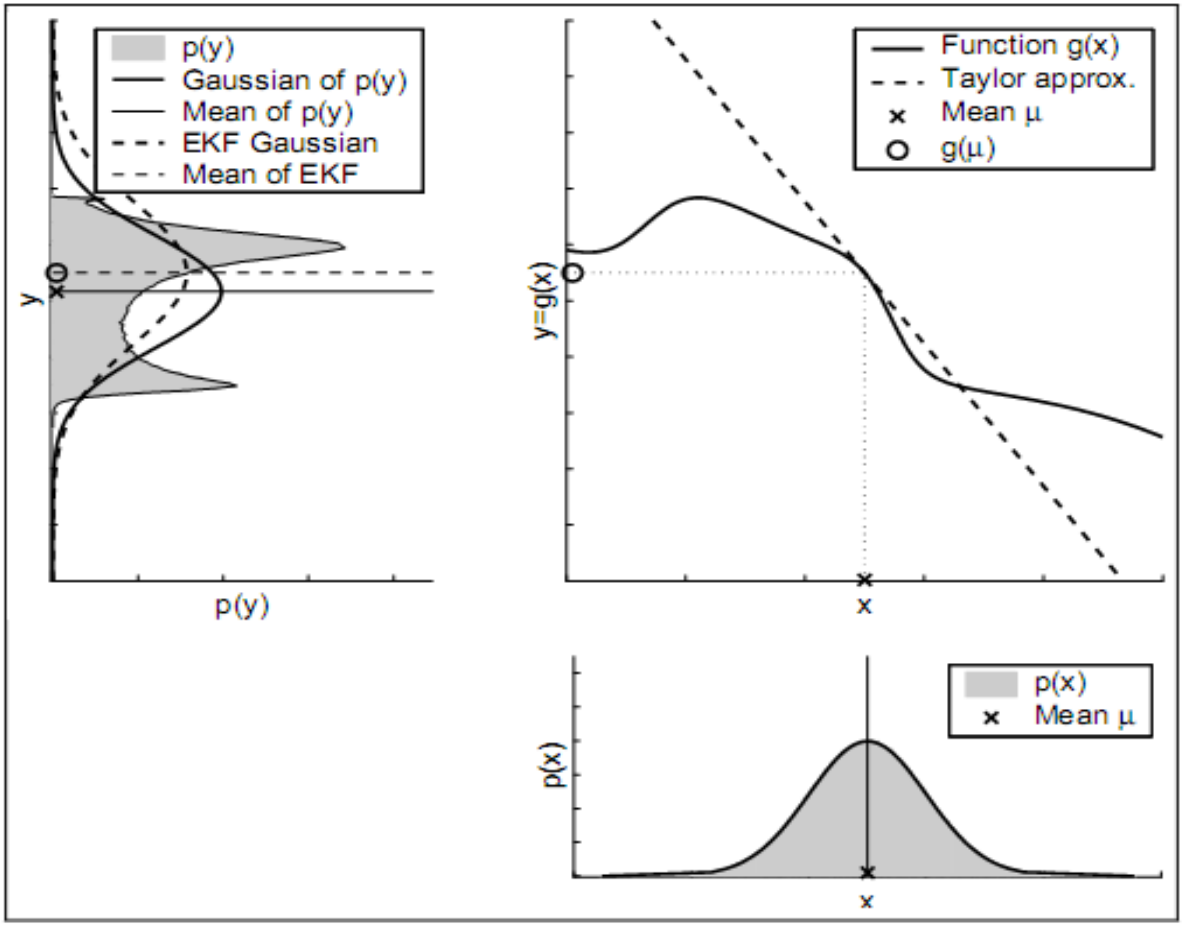
8. Return μ_t, Σ_t

Extended Kalman Filter

- Problem: Kalman Filter benötigt Linearität
- In der Realität ist diese Linearität häufig nicht gegeben.

- Lösung: Linearisierung
- Die nichtlineare Funktion g(x) wird durch ihre Tangente am Erwartungswert der Normalverteilung linearisiert

EKF: Linearisierung



Kombinierter Zustandsvektor

$$y_t = \begin{Bmatrix} x_t \\ m \end{Bmatrix} = \begin{Bmatrix} x & y & \theta & m_{1,x} & m_{1,y} & s_1 & m_{2,x} & m_{2,y} & s_2 & \dots & m_{N,x} & m_{N,y} & s_N \end{Bmatrix}^T$$

Der kombinierte Zustandsvektor besteht aus der Pose des Roboters und der Karte mit N Landmarken.

Die Pose des Roboters besteht aus x und y Koordinaten sowie der Ausrichtung des Roboters.

Die N Landmarken werden durch ihre Koordinaten und ihre Signatur im Vektor repräsentiert.

EKF-SLAM Algorithmus Teil 1: Motion Update

1: Algorithm EKF_SLAM_known_correspondence$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, c_t)$:

2: $F_x = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \end{pmatrix}$

3: $\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{w_t}\sin(\mu_{t-1,0}) + \frac{v_t}{w_t}\sin(\mu_{t-1,0} + w_t \Delta t) \\ \frac{v_t}{w_t}\cos(\mu_{t-1,0}) - \frac{v_t}{w_t}\cos(\mu_{t-1,0} + w_t \Delta t) \\ w_t \Delta t \end{pmatrix}$

4: $G_t = F_x^T \begin{pmatrix} 0 & 0 & -\frac{v_t}{w_t}\cos(\mu_{t-1,0}) + \frac{v_t}{w_t}\cos(\mu_{t-1,0} + w_t \Delta t) \\ 0 & 0 & -\frac{v_t}{w_t}\sin(\mu_{t-1,0}) + \frac{v_t}{w_t}\sin(\mu_{t-1,0} + w_t \Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_x$

5: $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + F_x^T R_t F_x$

EKF-SLAM Algorithmus Teil 2

$$6: Q_t = \begin{pmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\Phi^2 & 0 \\ 0 & 0 & \sigma_s^2 \end{pmatrix}$$

$$7: \textit{for all observed features } z_t^i = (r_t^i \; \Phi_t^i \; s_t^i)^T \textit{ do}$$

$$8: j = c_t^i$$

$$9: \textit{if landmark } j \textit{ never seen before}$$

$$10: \begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \\ \bar{\mu}_{j,s} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \\ s_t^i \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\Phi_t^i + \bar{\mu}_{t,0}) \\ r_t^i \sin(\Phi_t^i + \bar{\mu}_{t,0}) \\ 0 \end{pmatrix}$$

$$11: \textit{endif}$$

$$12: \delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix}$$

7ff: Iteration über alle Messungen
Messung = Entfernung, Richtung, Signatur

9-11: Initialisierung neuer (bisher unbekannter) Landmarken

EKF-SLAM Algorithmus Teil 3

$$13: q = \delta^T \delta$$

$$14: \hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ atan2(\delta_y, \delta_x) - \mu_{t,0} \\ \mu_{j,s} \end{pmatrix}$$

Berechnung der „erwarteten“ Messung

$$15: F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 1 & 0 & \dots & 0 \end{pmatrix}$$

$$16: H_t^i = \frac{1}{q} \begin{pmatrix} -\sqrt{q}\,\delta_x & -\sqrt{q}\,\delta_y & 0 & \sqrt{q}\,\delta_x & \sqrt{q}\,\delta_y & 0 \\ \delta_y & -\delta_x & -q & -\delta_y & \delta_x & 0 \\ 0 & 0 & 0 & 0 & 0 & q \end{pmatrix} F_{x,j}$$

EKF-SLAM Algorithmus Teil 4

17: Kalman-Gain: $K_t^i = \bar{\Sigma}_t H_t^{iT} (H_t^i \bar{\Sigma}_t H_t^{iT} + Q_t)^{-1}$

18: $\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i)$

19: $\bar{\Sigma}_t = (I - K_t^i H_t^i) \bar{\Sigma}_t$

20: *endfor*

21: $\mu_t = \bar{\mu}_t$

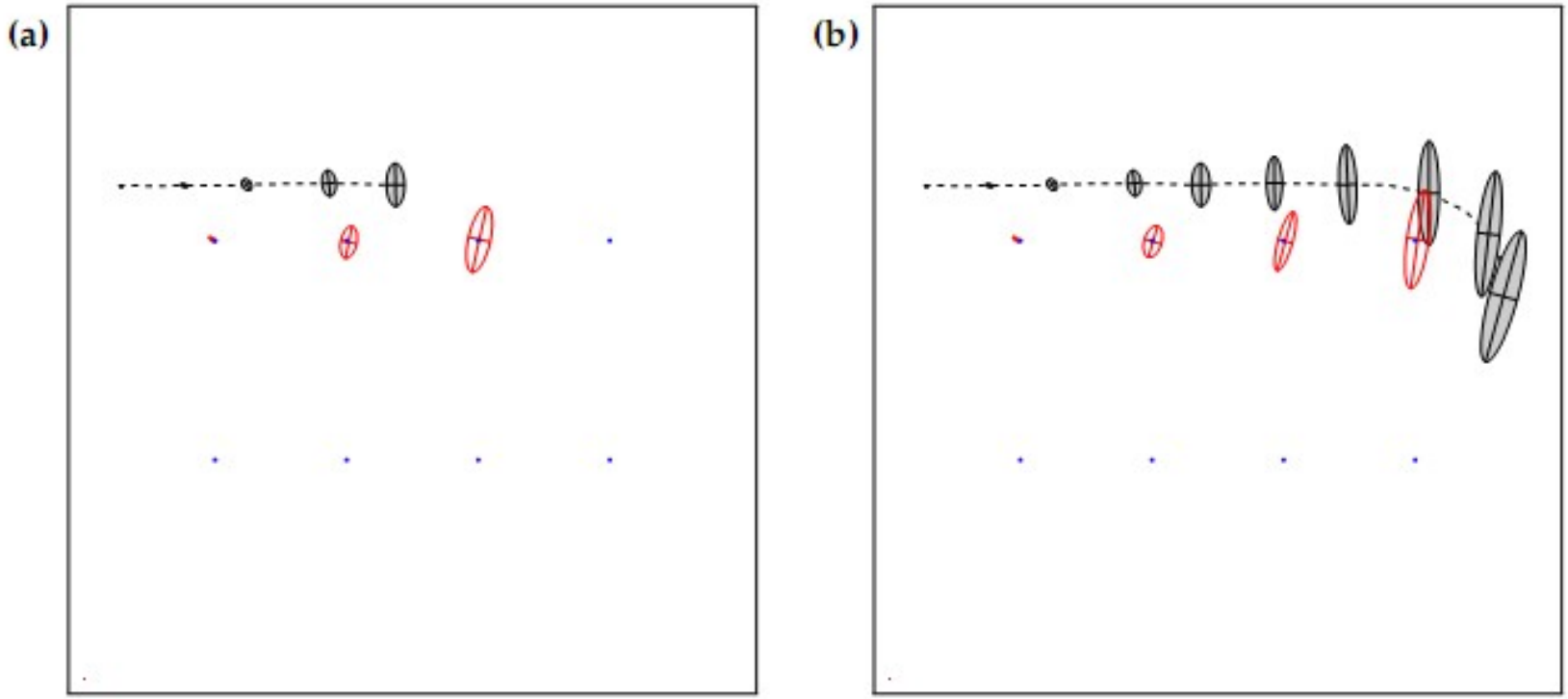
22: $\Sigma_t = \bar{\Sigma}_t$

23: *return* μ_t, Σ_t

17: Berechnung des Kalman-Gain

18,19: Filter-Update

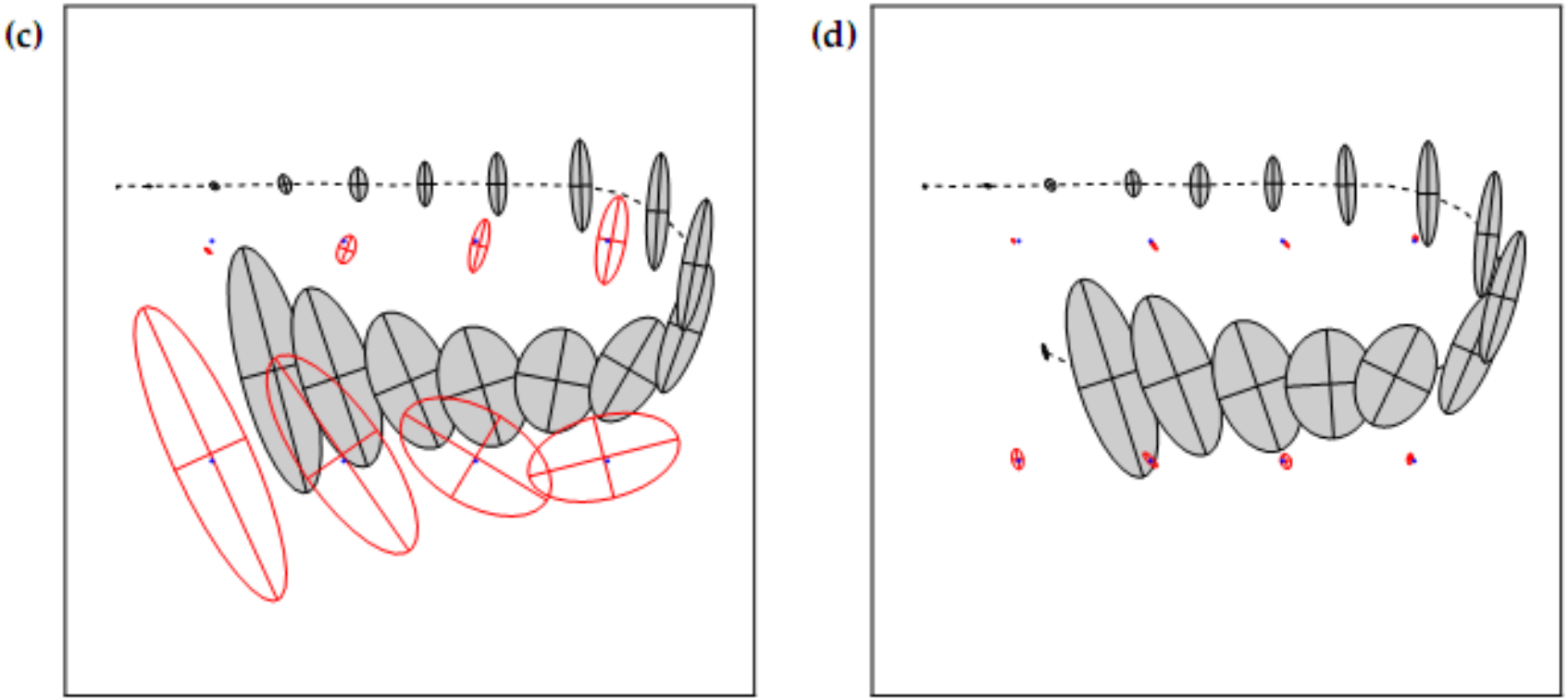
EKF-SLAM: Beispiel



Beispielfahrt eines Roboters in einer Welt mit 8 Landmarken

Mit jedem Schritt steigt die Ungewissheit über die Pose des Roboters sowie der Lage der Landmarken an.

EKF-SLAM Beispiel



Erst die erneute Sichtung der ersten Landmarke führt zu einer Abnahme der Unsicherheiten (für alle Landmarken und die Pose des Roboters)

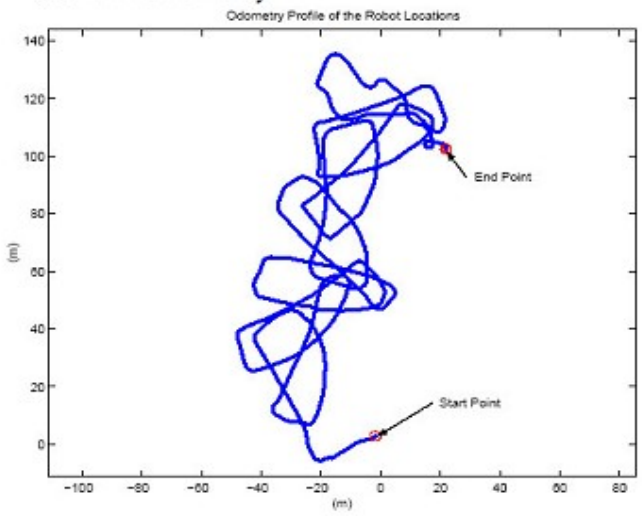
General EKF SLAM

- Korrespondenz oft unbekannt
- Erweiterung des Algorithmus nötig
- Statt einer gegebenen Korrespondenz wird die ML-Korrespondenz (maximum likelihood correspondence) verwendet
- Dem Algorithmus wird statt der Korrespondenz die momentane Größe der Karte übergeben

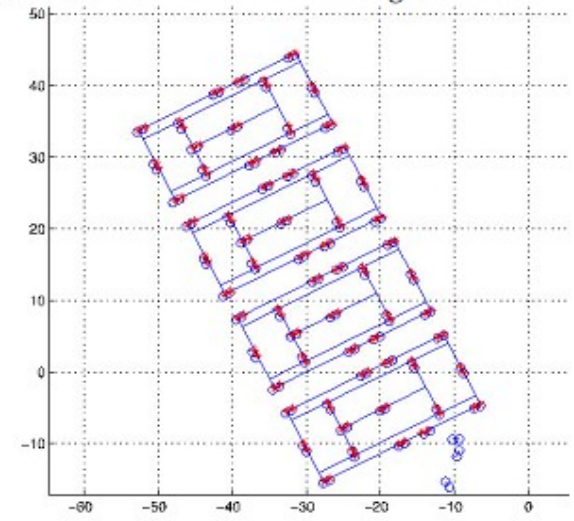
EKF-SLAM Beispiel 2

(a) RWI B21 Mobile robot and testing environment

(b) Raw odometry



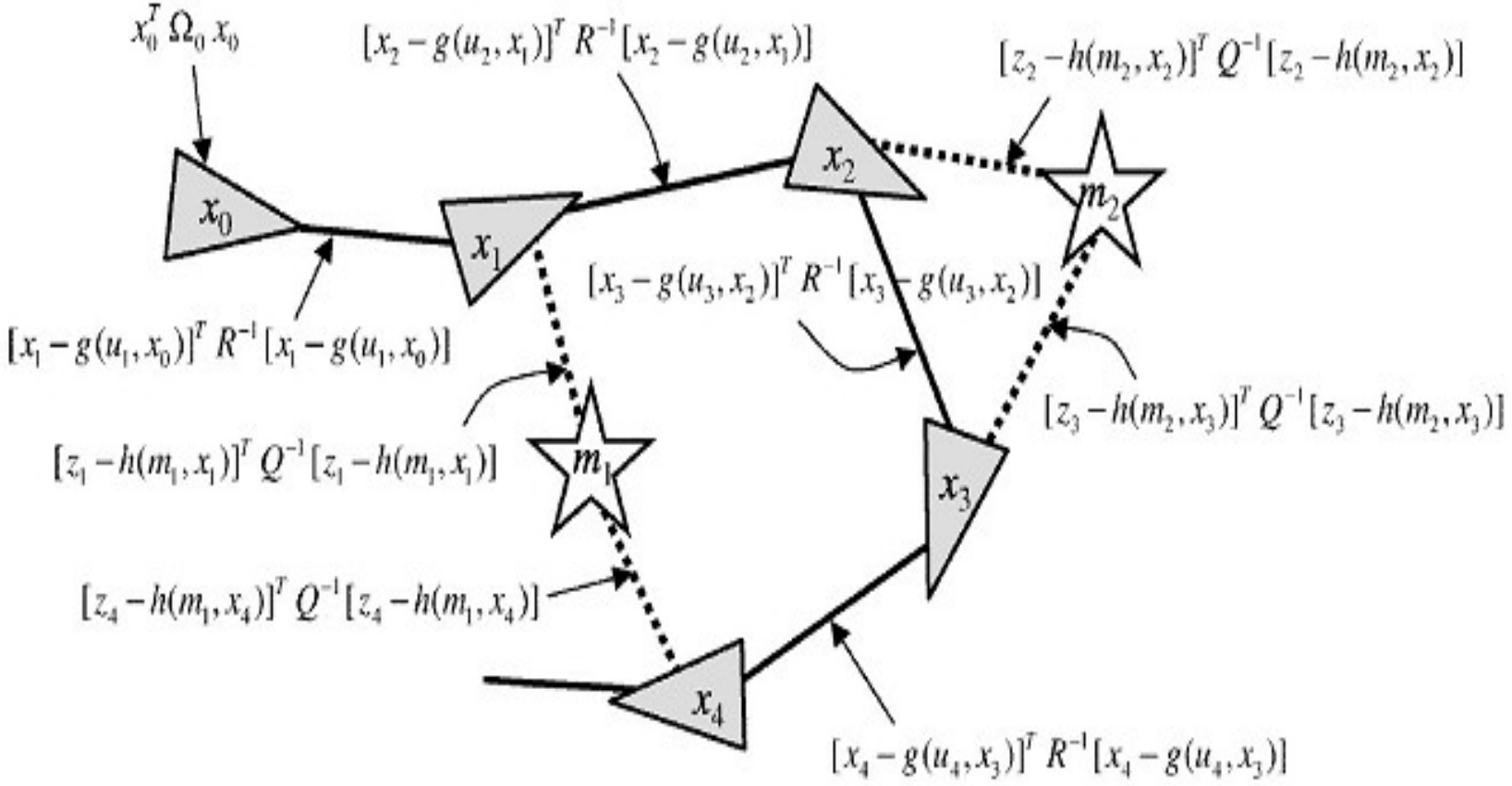
(c) Result of EKF SLAM with ground truth



GraphSLAM

- Full SLAM Verfahren
- Full SLAM Problem formt einen „sparse Graph“ (= Graph mit wenigen Kanten)
- Lazy SLAM (Datenerhebung und -verarbeitung werden getrennt durchgeführt)
- Graph wird reduziert bis er nur noch Roboterposen enthält

GraphSLAM

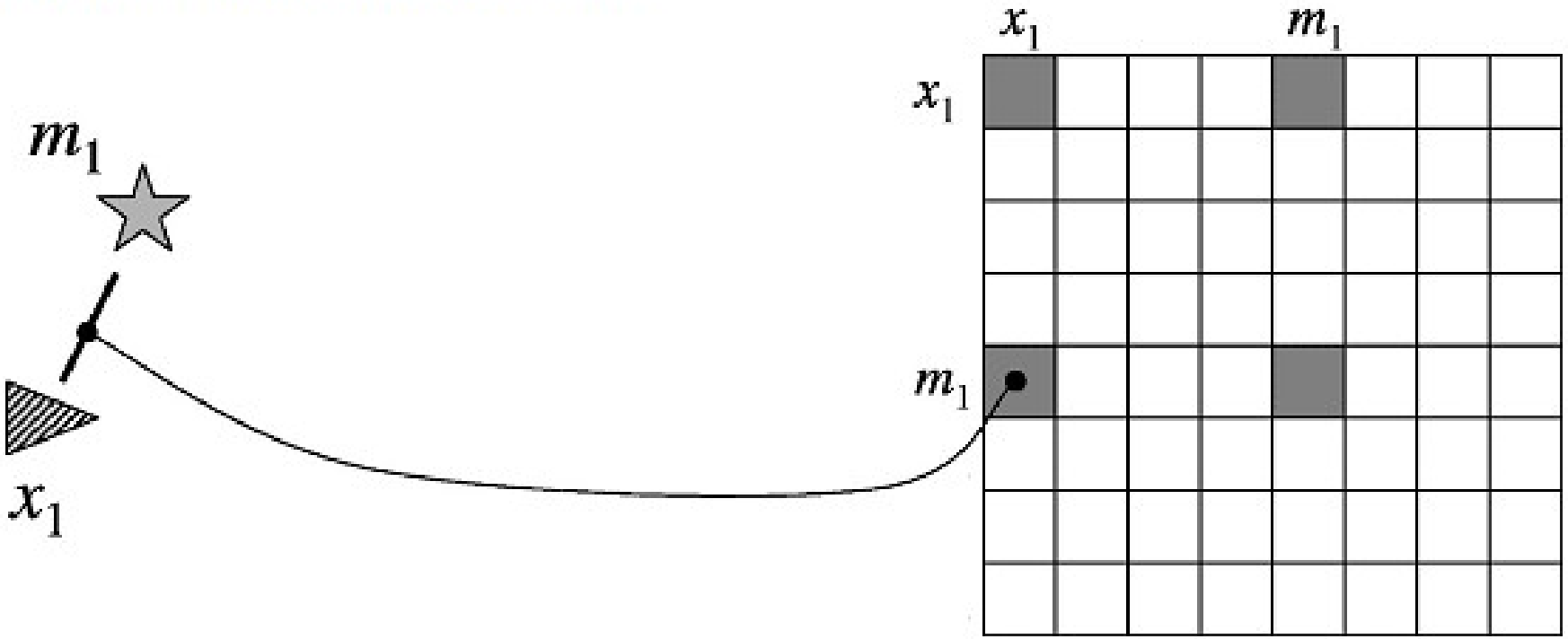

$x_0^T \Omega_0 x_0$
$[x_2 - g(u_2, x_1)]^T R^{-1} [x_2 - g(u_2, x_1)]$
$[z_2 - h(m_2, x_2)]^T Q^{-1} [z_2 - h(m_2, x_2)]$
x_0
x_1
x_2
m_2
$[x_3 - g(u_3, x_2)]^T R^{-1} [x_3 - g(u_3, x_2)]$
$[x_1 - g(u_1, x_0)]^T R^{-1} [x_1 - g(u_1, x_0)]$
$[z_3 - h(m_2, x_3)]^T Q^{-1} [z_3 - h(m_2, x_3)]$
$[z_1 - h(m_1, x_1)]^T Q^{-1} [z_1 - h(m_1, x_1)]$
m_1
x_3
$[z_4 - h(m_1, x_4)]^T Q^{-1} [z_4 - h(m_1, x_4)]$
x_4
$[x_4 - g(u_4, x_3)]^T R^{-1} [x_4 - g(u_4, x_3)]$

GraphSLAM Algorithmus: Hauptschritte

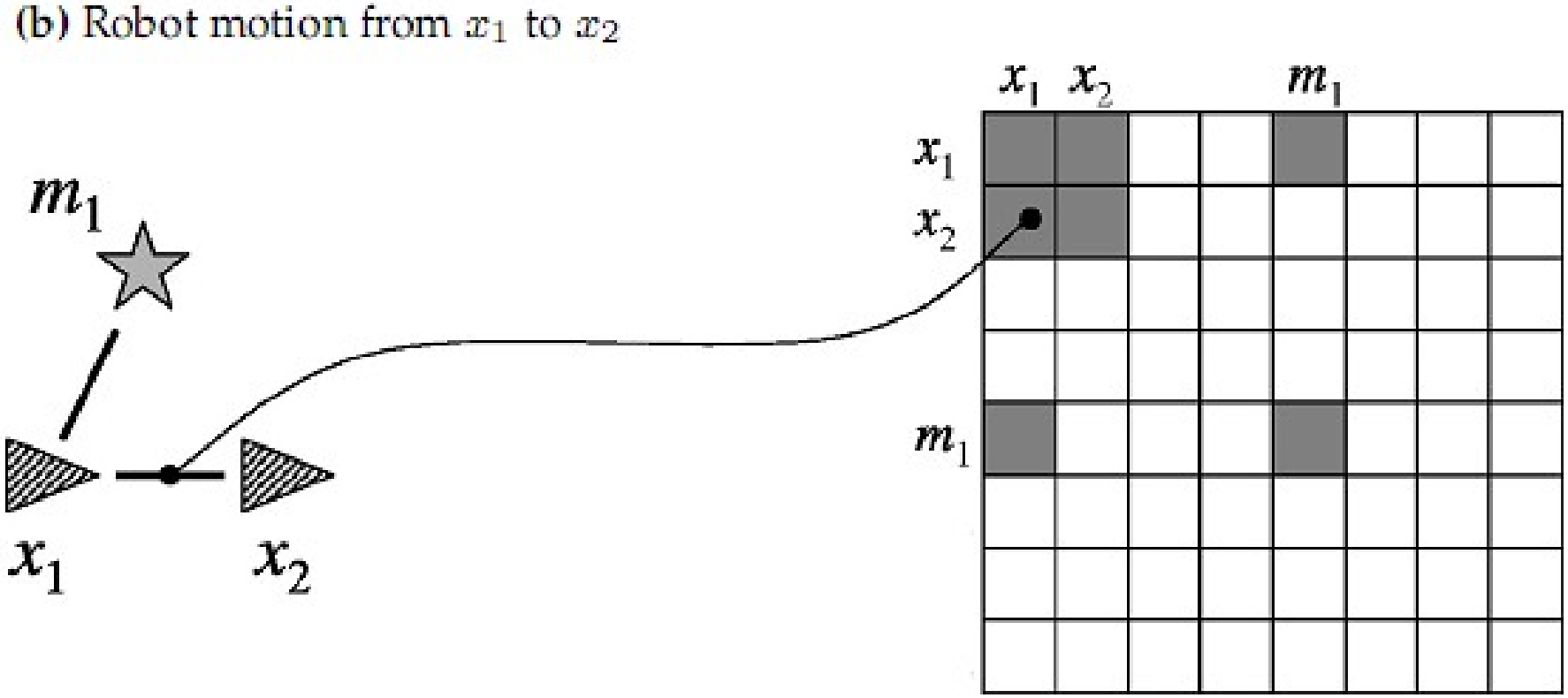
1.	initialize	Initialisierung des Posenerwartungswertvektors
2.	*repeat*	
3.	linearize	Erstellung von Informationsmatrix und -vektor
4.	reduce	Reduzieren der Matrix und des Vektors
5.	solve	Update μ (auch Erwartungswerte der Features)
6.	*until convergence*	
7.	*return* μ	

GraphSLAM Illustration: Erstellung der Informationsmatrix

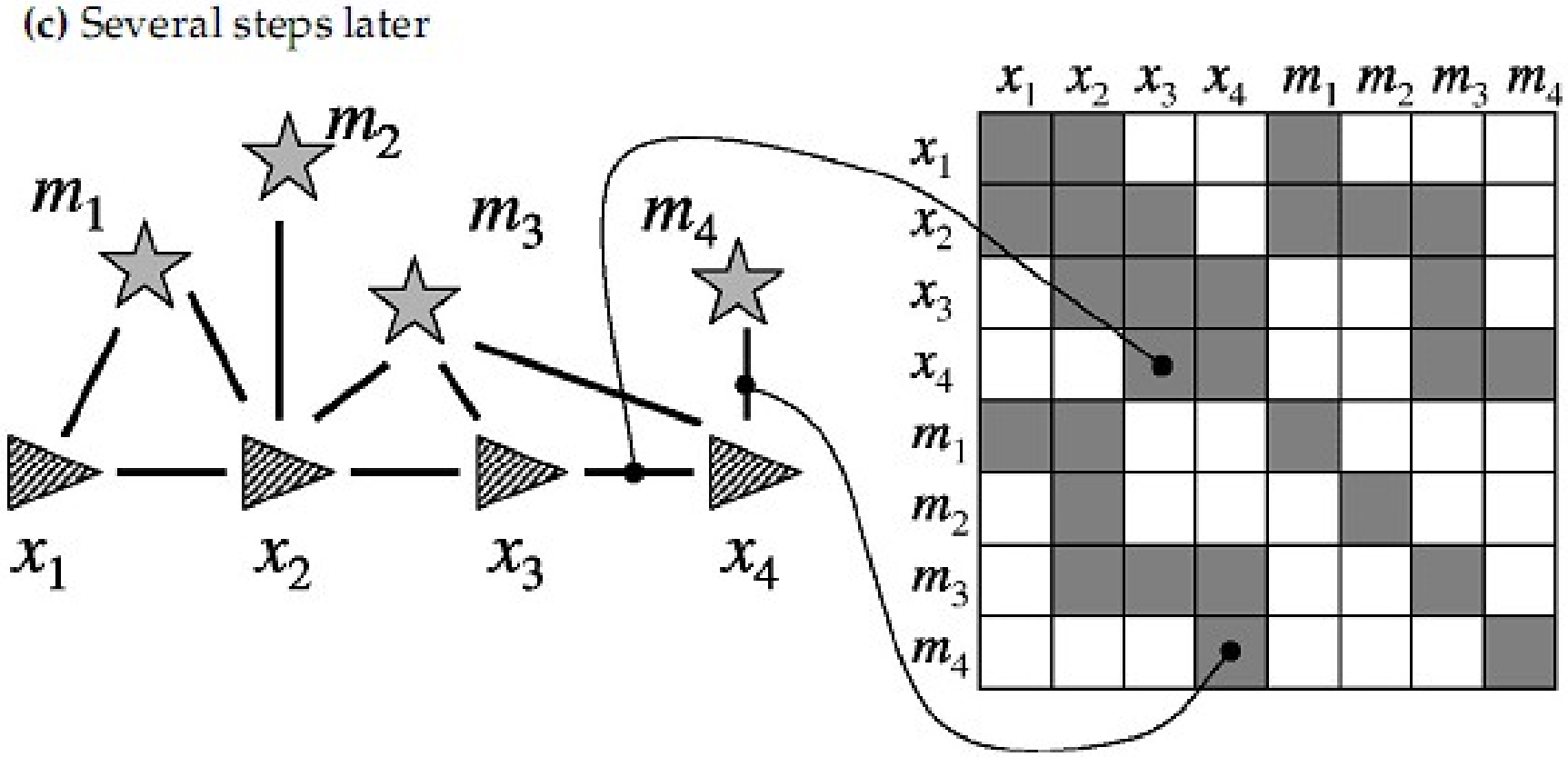
(a) Observation ls landmark m_1



GraphSLAM Illustration: Erstellung der Informationsmatrix

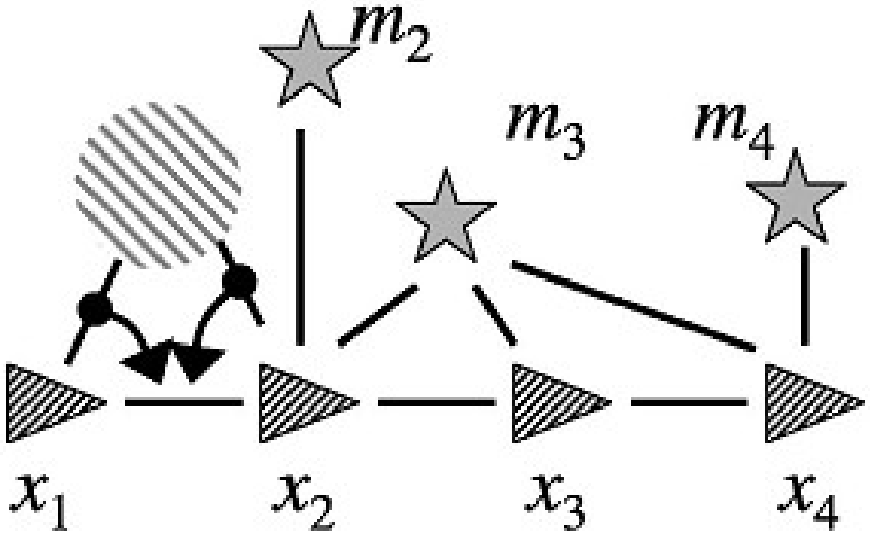


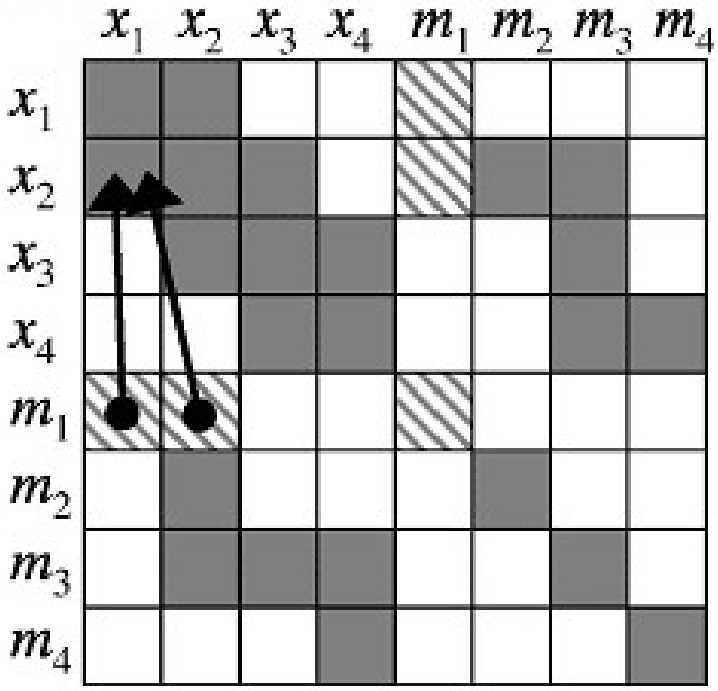
GraphSLAM Illustration: Erstellung der Informationsmatrix



GraphSLAM Illustration: Reduzieren der Informationsmatrix

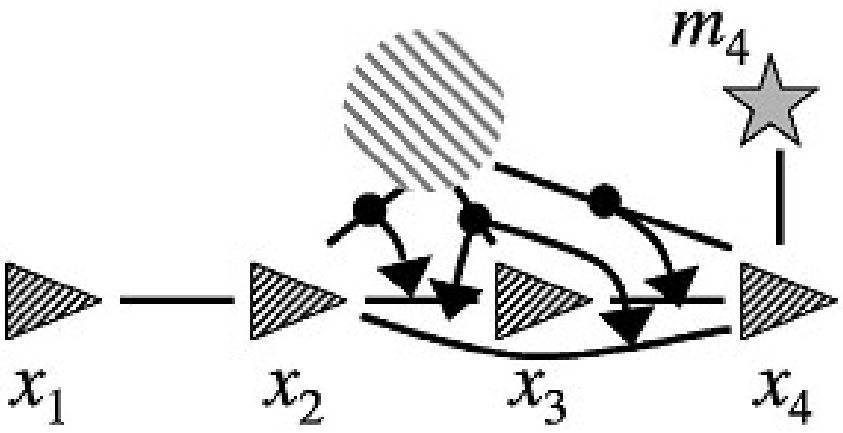
(a) The removal of m_1 changes the link between x_1 and x_2

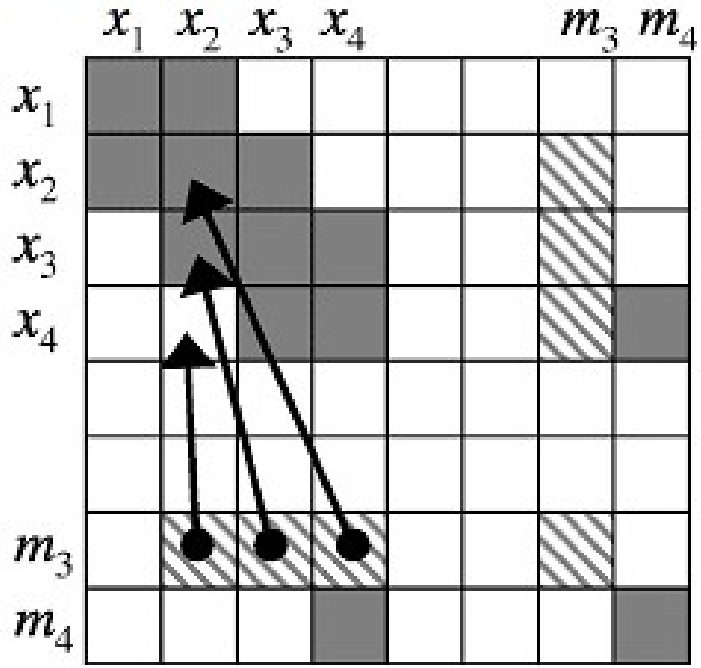




GraphSLAM Illustration: Reduzieren der Informationsmatrix

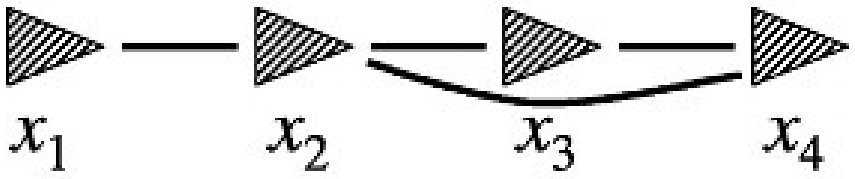
(b) The removal of m_3 introduces a new link between x_2 and x_4

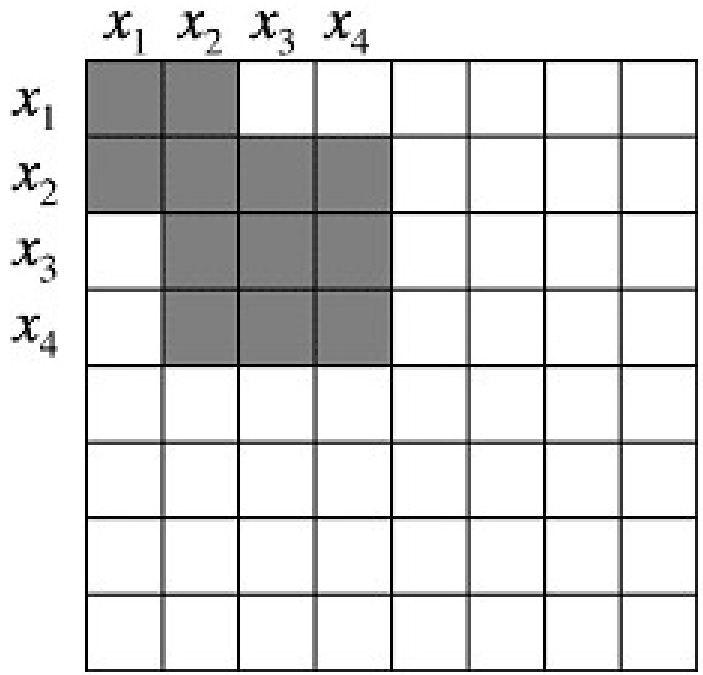




GraphSLAM Illustration: Ergebnis

(c) Final Result after removing all map features





Fast SLAM

- Nutzt Partikelfilter
- Direkte Anwendung von Partikelfiltern auf das SLAM-Problem nicht möglich
  - ➢ exponentieller Aufwand
- Als Online oder Full SLAM Verfahren einsetzbar

Fast SLAM „Trick“

Bei gegebener Pose besitzt das Full SLAM Problem mit bekannter Korrespondenz eine bedingte Unabhängigkeit zwischen 2 disjunkten Featuremengen.

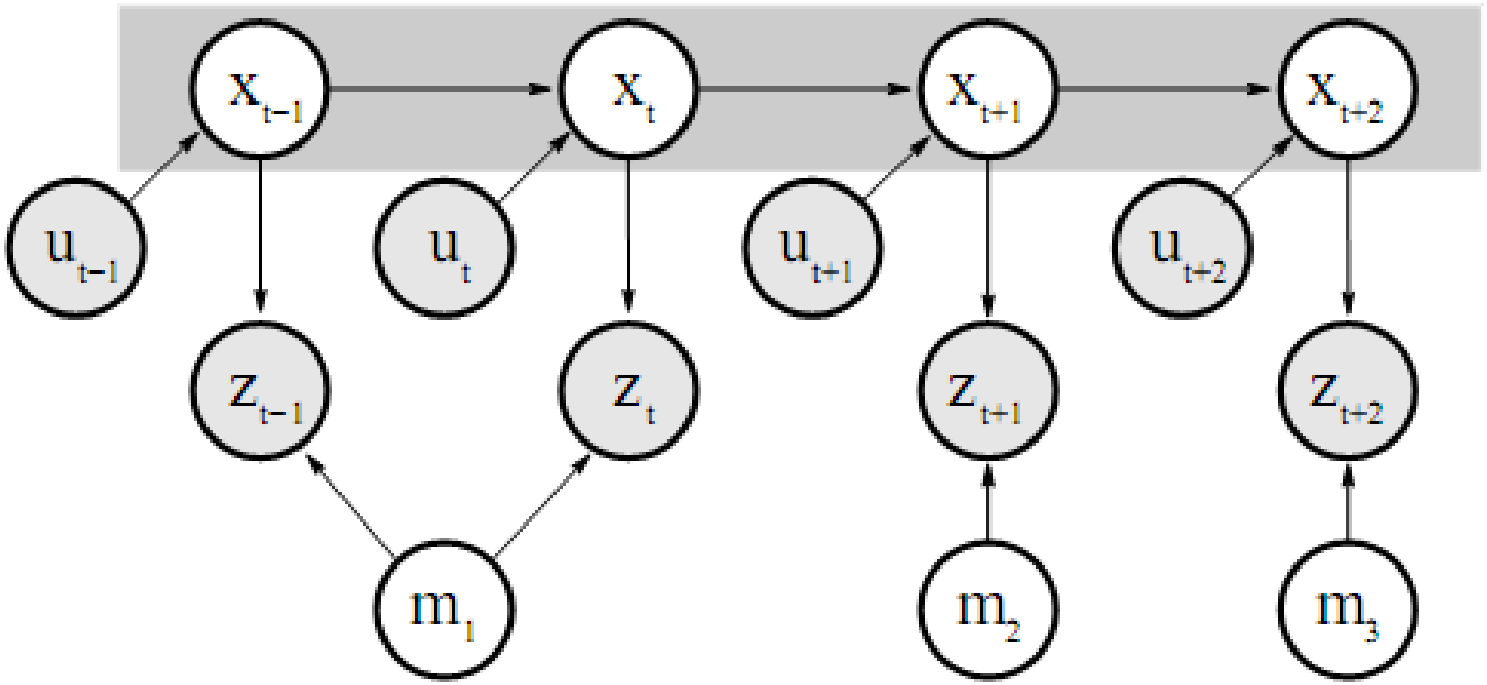
Abhängigkeiten entstehen nur durch die Unsicherheit der Pose.

Diese Beobachtung führt zur Möglichkeit „Rao-Blackwellized particle filters“ einzusetzen

FastSLAM „Trick“

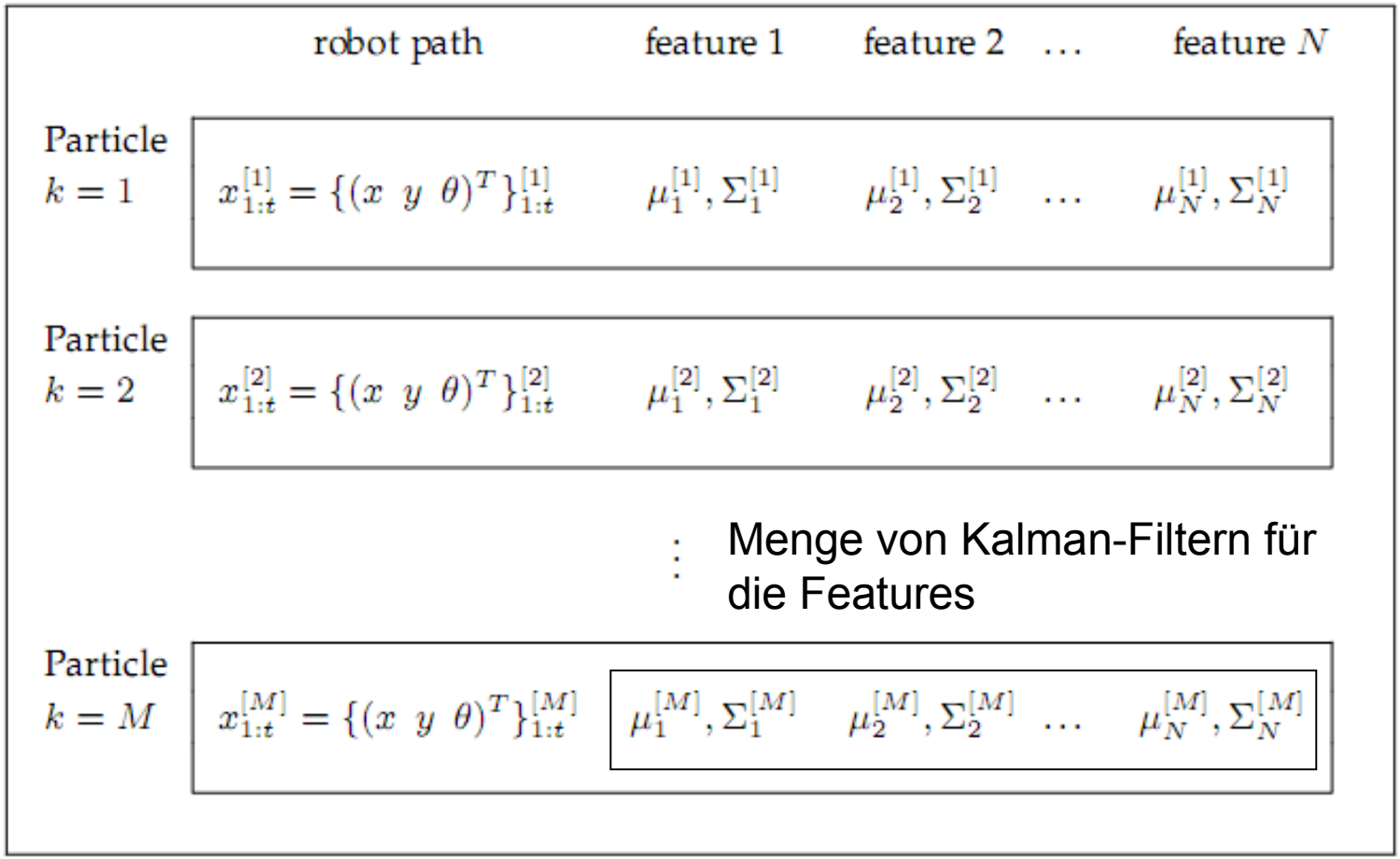
- Jeder Partikel behandelt einen ausgewählten Pfad
- Der Pfad wird ausgeklammert (je Partikel ein Pfad) und die Features werden behandelt als ob sie unabhängig wären
- Jedes Feature wird durch eine eigene Normalverteilung repräsentiert

FastSLAM Graph



Features werden getrennt von einander betrachtet. Die Gesamtkarte m besteht aus Teilkarten m_1 bis m_3.

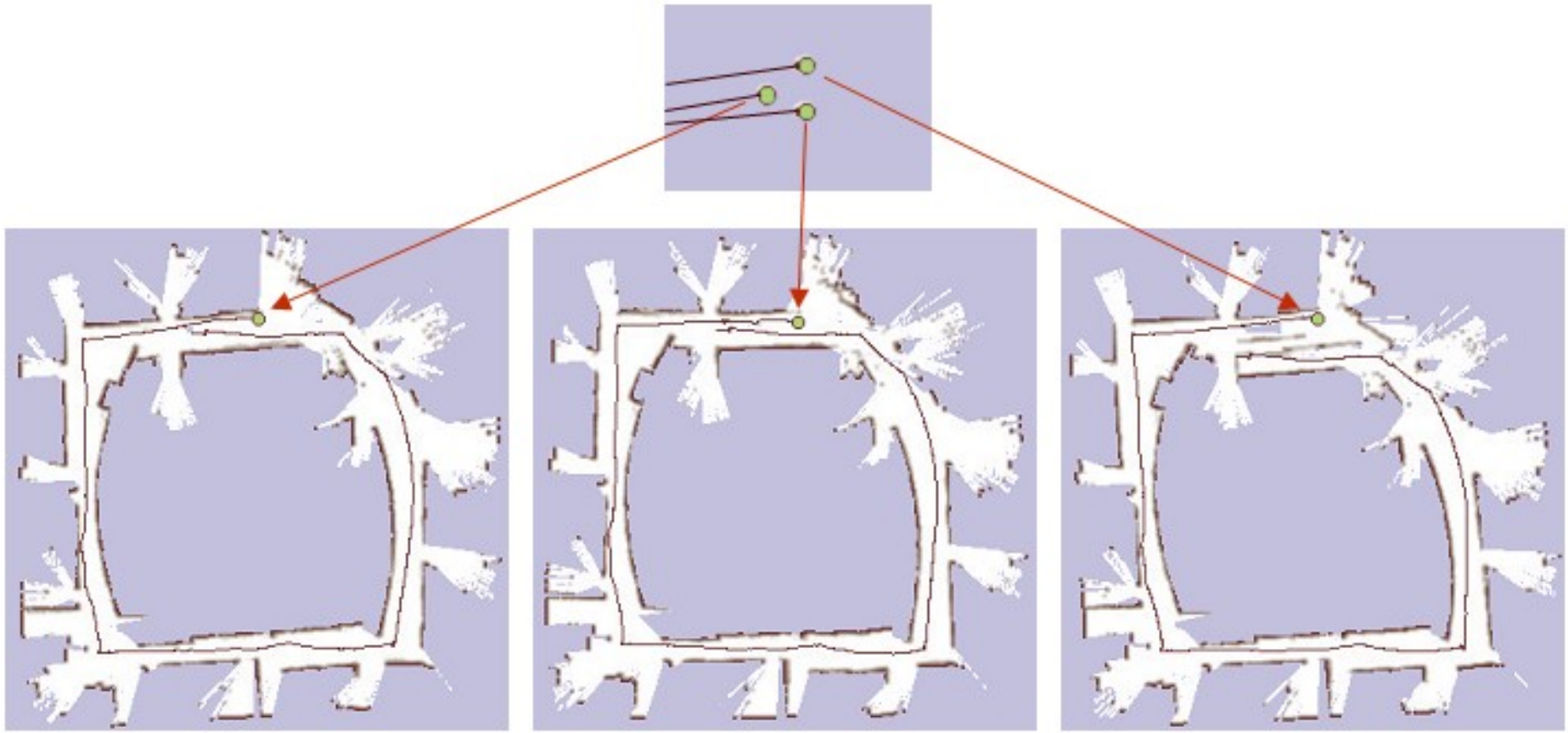
FastSLAM Partikel



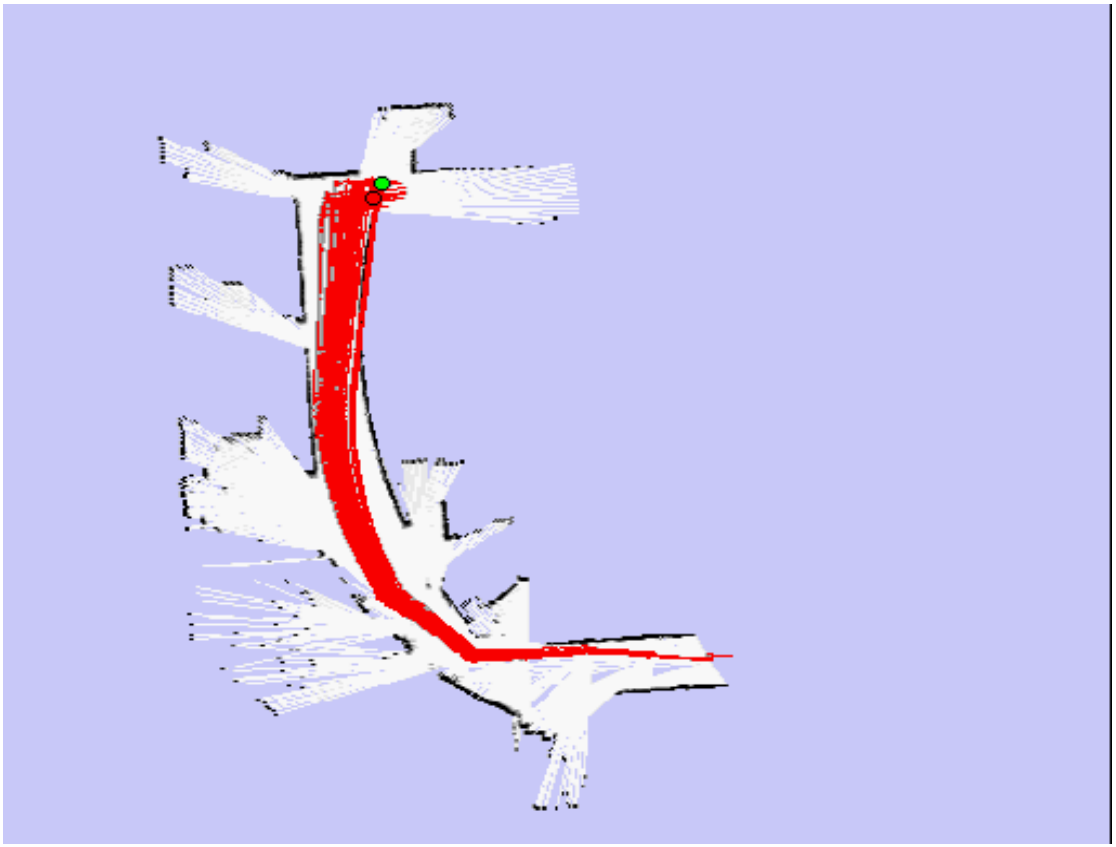
FastSLAM Algorithmus: Hauptschritte

- Do the following M times:
  - **Retrieval.** Retrieve a pose $x_{t-1}^{[k]}$ from the particle set Y_{t-1}.
  - **Prediction.** Sample a new pose $x_t^{[k]} \sim p(x_t \mid x_{t-1}^{[k]}, u_t)$.
  - **Measurement update.** For each observed feature z_t^i identify the correspondence j for the measurement z_t^i, and incorporate the measurement z_t^i into the corresponding EKF, by updating the mean $\mu_{j,t}^{[k]}$ and covariance $\Sigma_{j,t}^{[k]}$.
  - **Importance weight.** Calculate the importance weight $w^{[k]}$ for the new particle.
- **Resampling.** Sample, with replacement, M particles, where each particle is sampled with a probability proportional to $w^{[k]}$.

FastSLAM Beispiel (grid-based)



FastSLAM Beispiel Video



Feature Elimiation

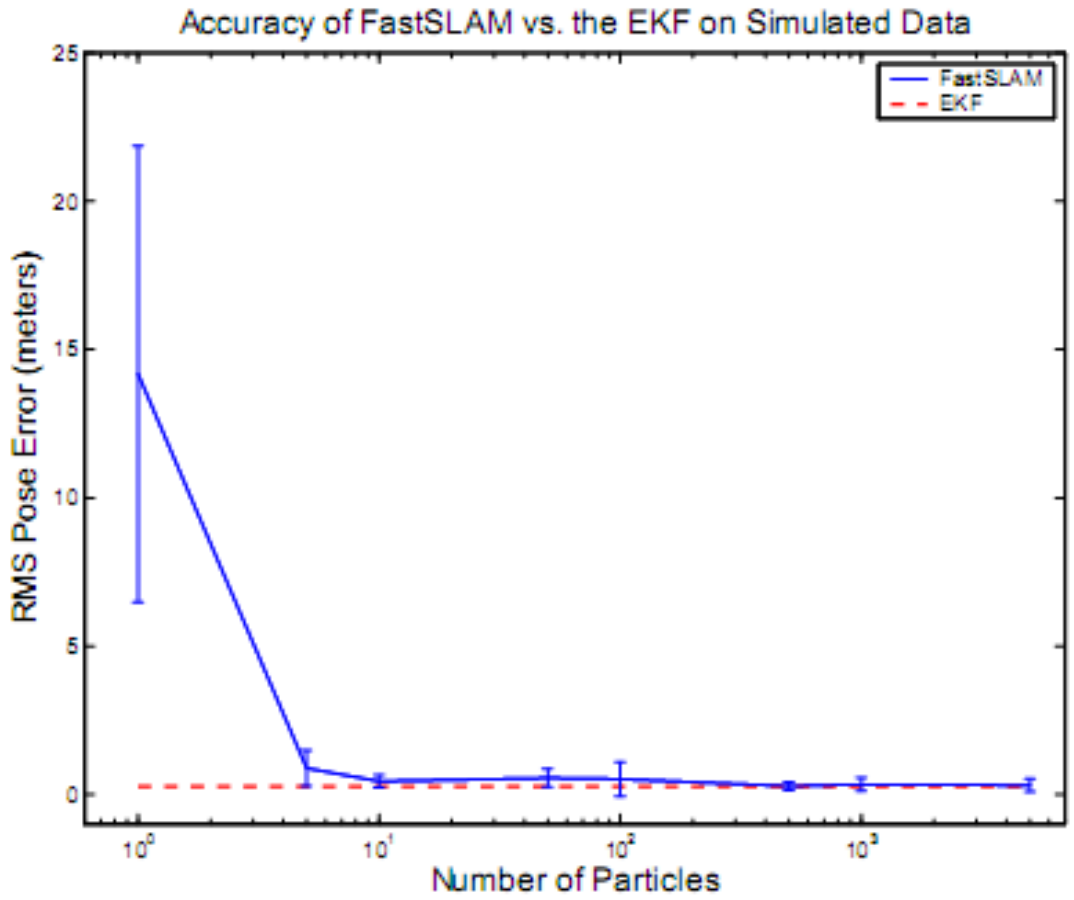
- FastSLAM kann die Abwesenheit von Features nutzen
- Dadurch können zweifelhafte/falsche Features (bewegliche Objekte die bei der ersten Beobachtung als Feature erkannt wurden) eliminiert werden

Feature Elimination Beispiel

(a) Map without feature elimination

(b) Map with feature elimination

Vergleich: FastSLAM & EKF



Quellen

- Literatur:

Sebastian Thrun, Wolfram Burgard, Dieter Fox: Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The Mit Press, 2005, ISBN 978-0-262-20162-9

- Abbildungen

http://robots.stanford.edu/probabilistic-robotics/ppt/

http://robots.stanford.edu/probabilistic-robotics/fig.pdf

- Website zum Buch

http://www.probabilistic-robotics.org/