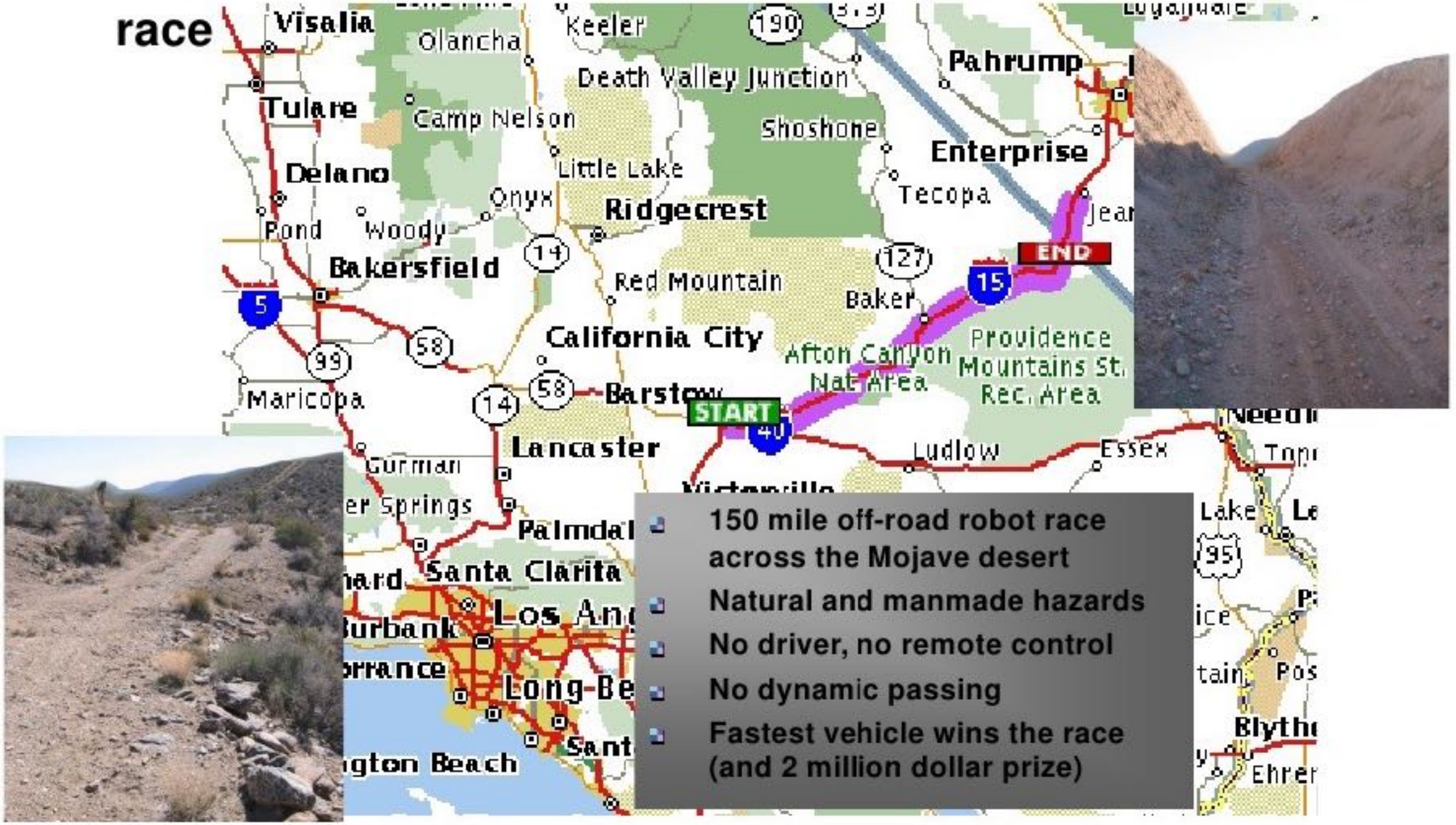
Simultaneous Localisation And Mapping — SLAM

DARPA Grand Challenge

- **The challenge: a driverless car competes for wining the race**



- 150 mile off-road robot race across the Mojave desert
- Natural and manmade hazards
- No driver, no remote control
- No dynamic passing
- Fastest vehicle wins the race (and 2 million dollar prize)

https://www.youtube.com/watch?v=8W2pqtbaybE

https://www.youtube.com/watch?v=lPG0hzoKIto

Sebastian Thrun, Gewinner der DARPA Grand Challenge 2005

SLAM — Problemdefinition

- Ein mobiler Roboter erkundet eine unbekannte Umgebung von einer bekannten Startposition

- Die Bewegungen sind ungenau, d.h. Positionsabschätzung wird mit der Zeit immer ungenauer

- Der Roboter hat eine Sensorik mit der die Umgebung wahrgenommen werden kann

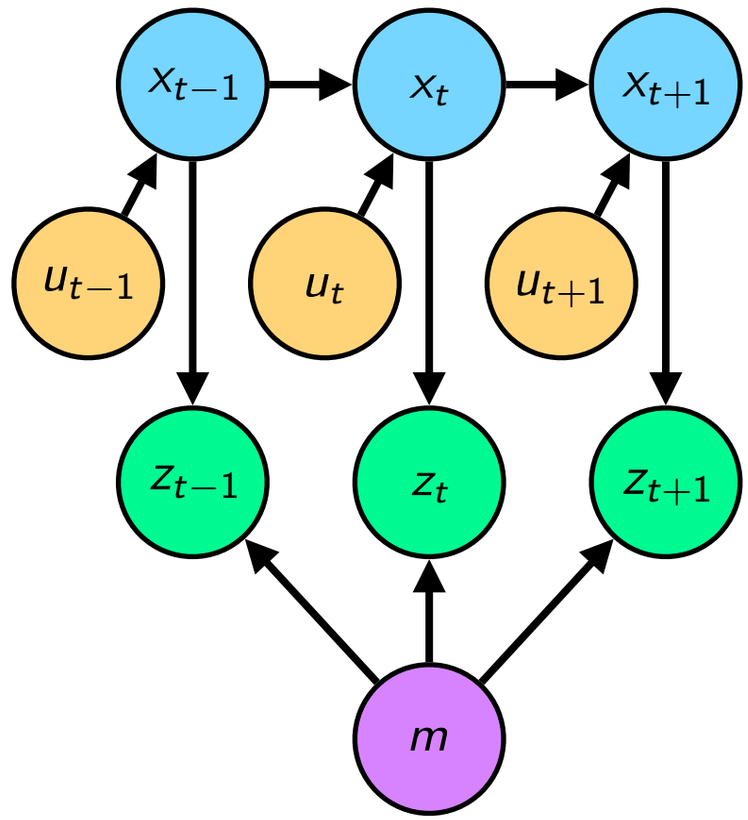
- Aufgabe: Erstellen einer Karte bei gleichzeitiger Lokalisierung in der Karte

- Einer der wichtigsten (teilweise noch ungelösten) Aufgaben in der Robotik

SLAM — Mathematische Beschreibung des Problems

Notation:

Zeit	t
Posture	$x_t = (x, y, \theta)$
Pfad	$x_{1:t} = \{x_0, x_1, \dots, x_t\}$
Initialposture	x_0
Odometrie	u_t, Bewegung von $t-1 \rightarrow t$
Relative Bewegung	$u_{1:t} = \{u_1, u_2, \dots, u_t\}$
	wäre ausreichend, wenn es kein Rauschen gäbe
True map	m (Weltwissen)
Annahme:	Ein Messung pro Zeiteinheit
Sensordaten	$z_{1:t} = \{z_1, z_2, \dots, z_t\}$



SLAM — Unterschiedliche Arten

Full-SLAM (Offline-SLAM)

Batch process, d.h. man berechnet die Karte und Trajektorie anschliessend, basierend auf allen Daten:

$$p(\underbrace{x_{1:t}, m}_{\text{not observerd}} \mid \underbrace{z_{1:t}, u_{1:t}}_{\text{observerd}})$$

Online-SLAM

Inkrementelle Berechnung der aktuellen Posture (wird auch Filter bezeichnet)

$$p(x_t, m | z_{1:t}, u_{1:t})$$

Benötigt 2 Modelle:

$p(x_t \mid x_{t-1}, u_t)$	Odometriemodell
$p(z_t \mid x_{t-1}, m)$	Sensormodell

Beispiel: SLAM mit Landmarken

Annahme:

- Die Welt besteht aus *N* punktförmigen Landmarken, d.h. die Welt läßt sich vollständig durch *2N* Werte beschreiben (2D)
- Der Roboter kann die relative Entfernung, den relativen Winkel und die Identität des Objekts bestimmen

Definitionen:

- Exaktes (rauschfreies) Sensormodell $\hat{z}_t = h(x_t, m)$
- Verrauschtes Sensormodell $p(z_t|x_t, m) \sim \mathcal{N}(h(x_t, m), Q_t)$
- Bewegungsmodell $\hat{x}_t = g(x_{t-1}, u_t)$

$$\mathcal{N}(\mu_t, \Sigma_t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$$\mathcal{N}(\mu_t, \Sigma_t) = \frac{1}{\sqrt{(2\pi)^k \det \Sigma}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

Beispiel: SLAM mit Landmarken

Annahme:

- Die Welt besteht aus *N* punktförmigen Landmarken, d.h. die Welt läßt sich vollständig durch *2N* Werte beschreiben (2D)
- Der Roboter kann die relative Entfernung, den relativen Winkel und die Identität des Objekts bestimmen

Definitionen:

- Exaktes (rauschfreies) Sensormodell $\hat{z}_t = h(x_t, m)$
- Verrauschtes Sensormodell $p(z_t|x_t, m) \sim \mathcal{N}(h(x_t, m), Q_t)$
- Bewegungsmodell $\hat{x}_t = g(x_{t-1}, u_t)$
- Verrauschtes Bewegungsmodell $p(x_t|x_{t-1}, u_t) \sim \mathcal{N}(g(x_{t+1}, u_t), R_t)$

$$\mathcal{N}(\mu_t, \Sigma_t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$$\mathcal{N}(\mu_t, \Sigma_t) = \frac{1}{\sqrt{(2\pi)^k \det \Sigma}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

Kategorisierungen in SLAM

Annahmen und Methoden sollten explizit angeben werden
(z.B. in einer Publikation, Buch, Implementierung, …)

Volumetric vs. Feature-Based

Volumenbasiert: erlaubt fotorealistische Rekonstruktionen der Umgebung

Feature-based: z.B. Landmarken, Liniensegmente

Topological vs. Metric

Topologisch: Qualitative der Objektbeziehungen (Obj. A hinter Obj. B)

Metrisch: Abstandsinformation

Known vs. Unknown Correspondence

Zuordnung von Sensordaten zu Objekten. *Data Association Problem* ist eines der schwierigsten Probleme im SLAM

Kategorisierungen in SLAM

Static vs. Dynamic

Statisch: Umgebung bleibt unverändert mit der Zeit

Dynamisch: Veränderungen sind erlaubt. Aufwendiger, aber auch robuster

Small vs. Large Uncertainty

Algorithmen unterscheiden sich darin, wie gut sie Positionsungenauigkeiten vertragen

Loop-closing Problem, wenn der Pfad des Roboters sich selbst kreuzt

Active vs. Passive

Passive: Roboter wird ferngesteuert

Aktiv: Roboter exploriert eigenständig

Single- vs. Multi-Robot

Multi-Robots noch selten, aber bekommen mehr Aufmerksamkeit

Hauptansätze in SLAM

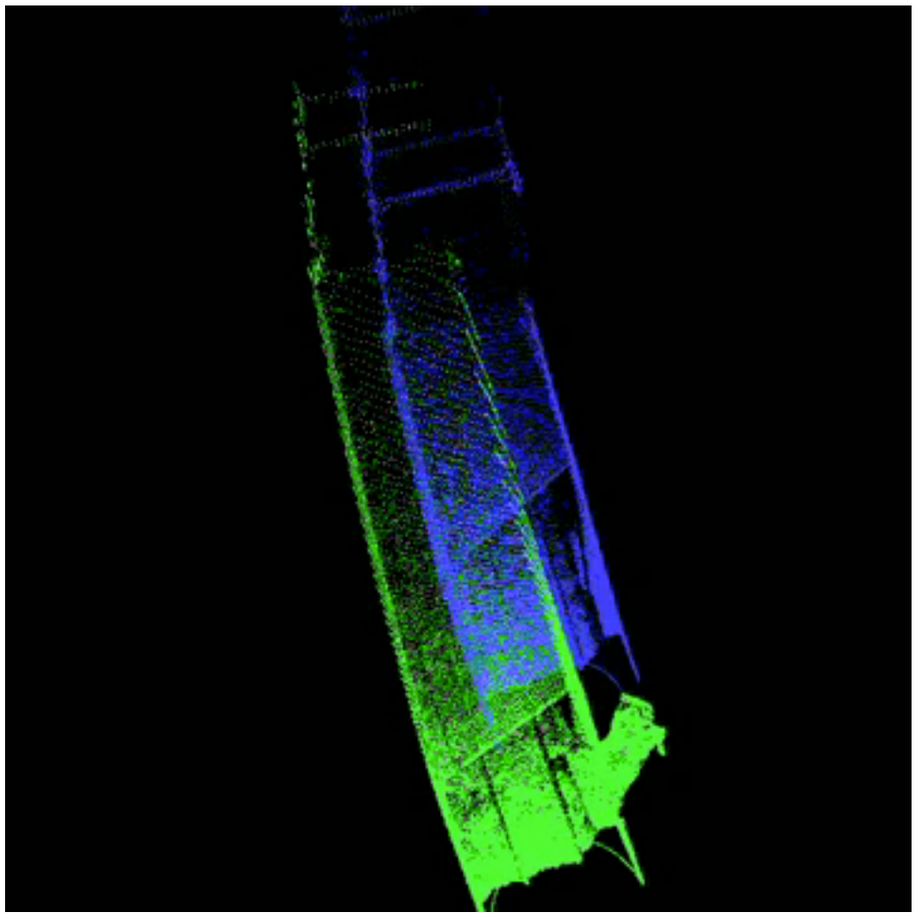
- Extende Kalman-Filter
  - Historisches erstes Verfahren
  - Sehr hoher Rechenaufwand
  - Wird als eigenständiges Verfahren unwichtiger

- Iterative Closest Point
  - Eines der ersten Verfahren (für 3D-Punktmengen)

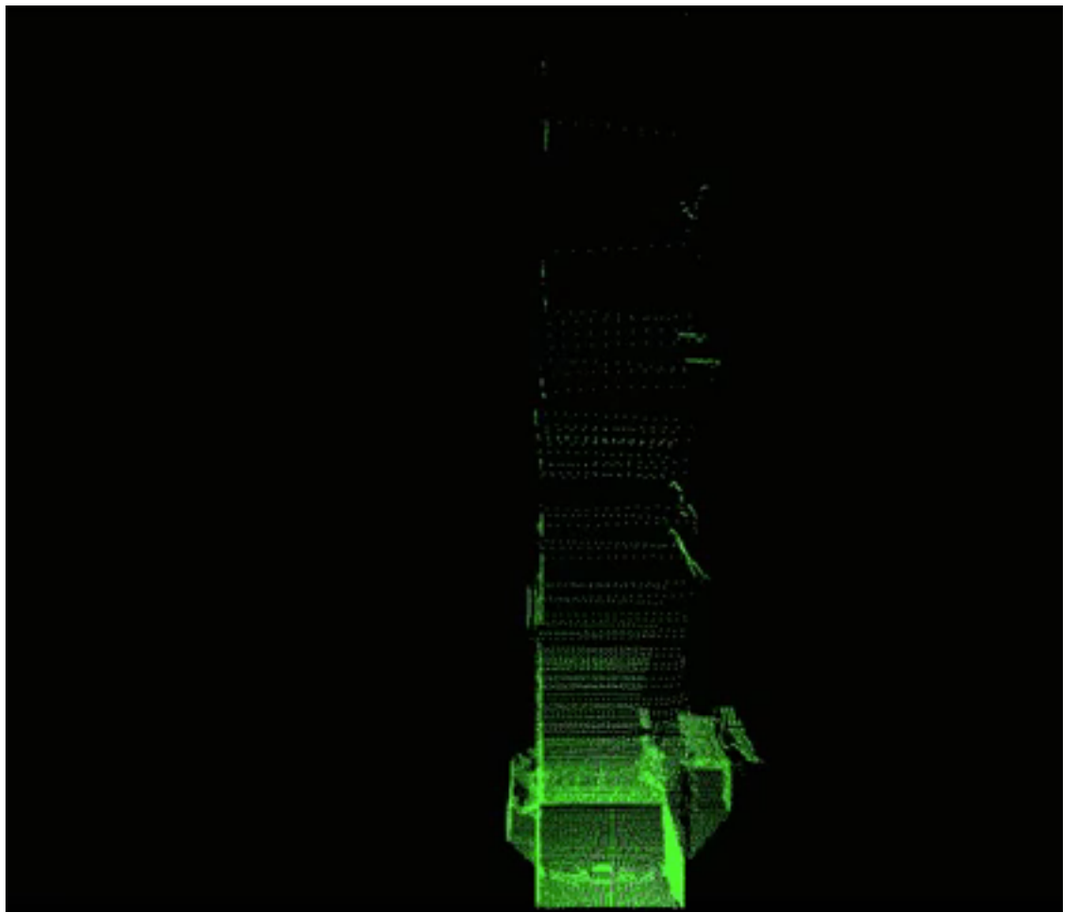
- Graph-basierte Verfahren
  - Hauptmethode für Full-SLAM

- Particle Filter
  - Nonparametric statistical filtering
  - FastSLAM scheint in diesem Bereich gerade zu dominieren
  - Neuer Ansatz für das *Data Association Problem*

Iterative Closest Point (ICP)



https://www.youtube.com/watch?v=qhT1lWCuAm4



https://www.youtube.com/watch?v=I2E9DFLAZ4o

Andreas Nüchter

Iterative Closest Point (ICP)

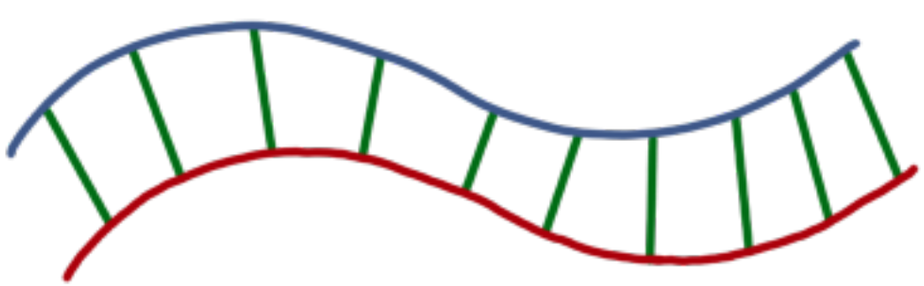
- Gegeben sind zwei Punktmengen

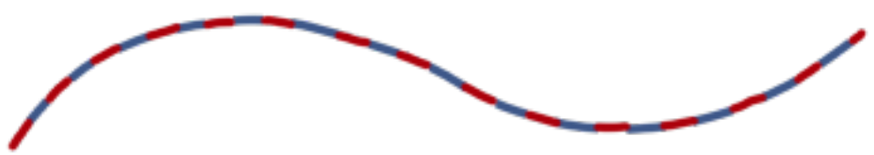
$$X = \{x_1, x_2, \dots, x_n\}, x_i \in \mathbb{R}^n$$
$$Y = \{y_1, y_2, \dots, y_n\}, x_i \in \mathbb{R}^n$$

- Gesucht wird die Rotation R und die Translation t, die den Fehler E minimieren

$$E(R, t) = \frac{1}{N} \sum_{i=1}^{N} \|x_i - Ry_i - t\|^2$$

wobei x_i und y_i korrespondierende Punkte sind.





Iterative Closest Point (ICP)

Wenn die korrespondierenden Punkte bekannt sind, läßt sich die Lösung für $x, y \in \mathbb{R}^3$ direkt berechnen:

1. Massenschwerpunkte der Punktmengen berechnen:

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \text{und} \qquad \mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i$$

$$X' = \{x_i - \mu_x\} = \{x'_i\} \qquad \text{und} \qquad Y' = \{y_i - \mu_y\} = \{y'_i\}$$

2. Singulärwertzerlegung:

$$W = \sum_{i=1}^{N} x'_i y'_i \qquad W = U \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T$$

3. Optimale Lösung:

$$R = UV^T, \qquad t = \mu_x - R\mu_y$$

Iterative Closest Point (ICP)

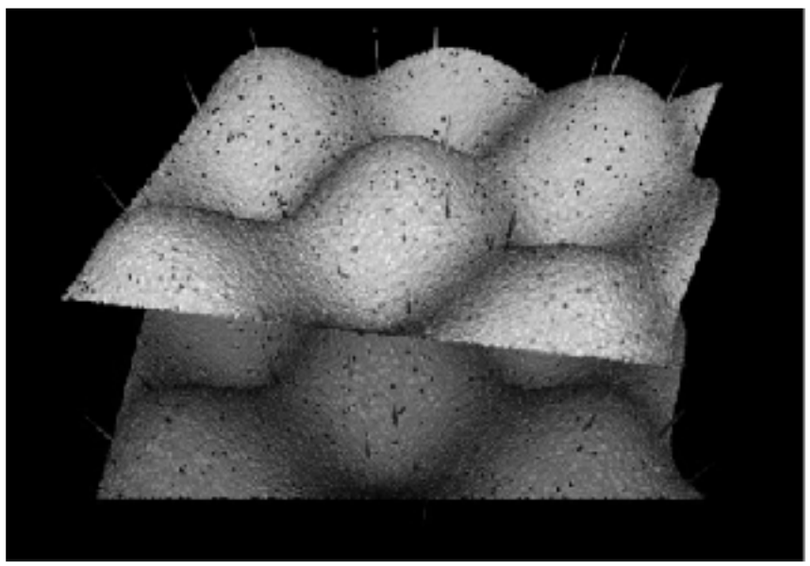
- Idee: Zugehörigkeiten (*correspondences*) iterative finden

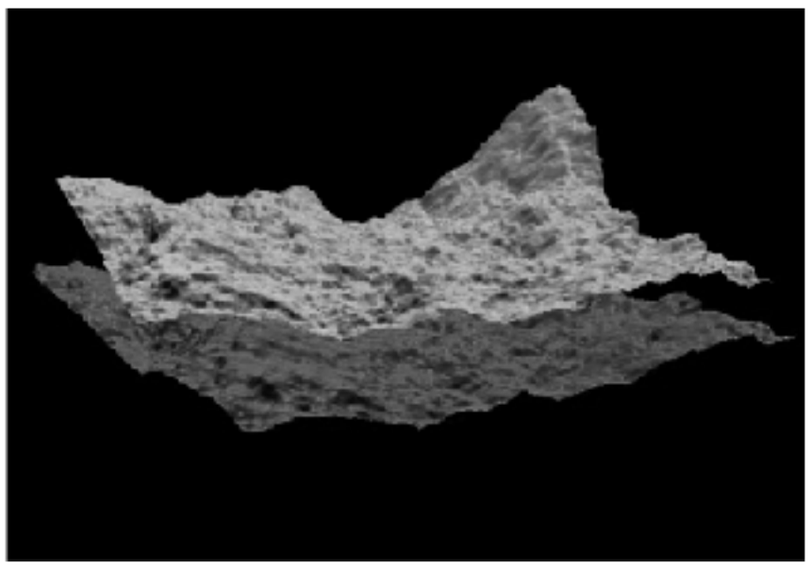
- Varianten von ICP

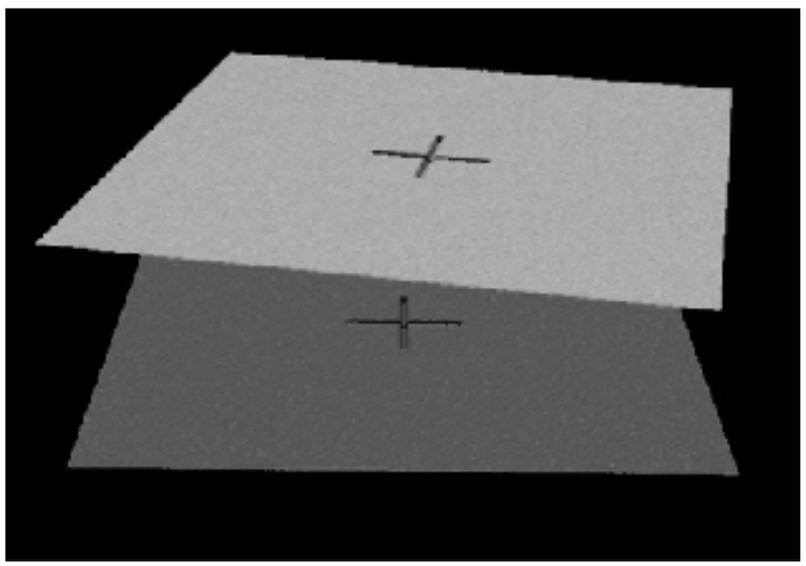
    1. *Selection* — Verteilung der Punkte
    2. *Matching* — Auswahl der Zugehörigkeiten
    3. *Weighting* — Gewichtung der Zugehörigkeiten
    4. *Rejecting* — Auswahl / Ignorieren von Punkten
    5. *Metric* — Abstandsberechnung

Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP)



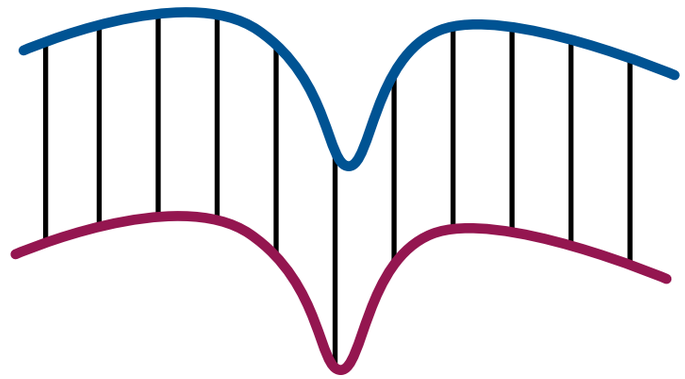




Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP) — Selection

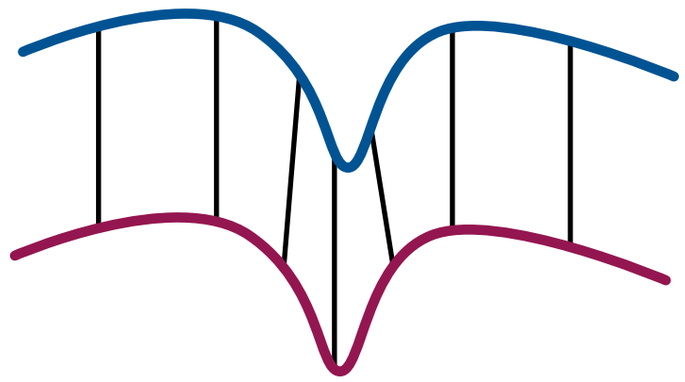
1. Alle verfügbaren Punkte



2. Uniformes sampling

3. Random sampling

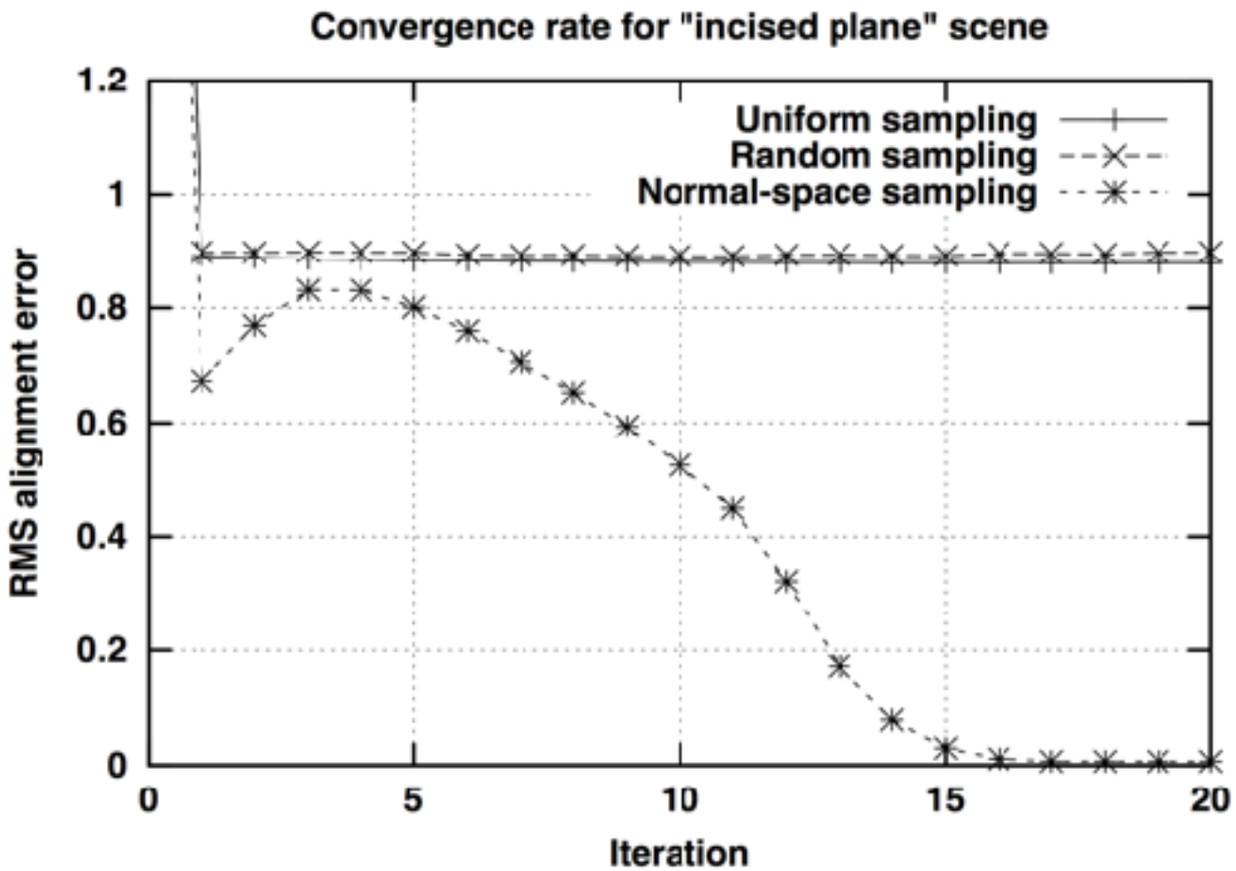
4. Gradientenbasierte Auswahl (z.B. Farbe)

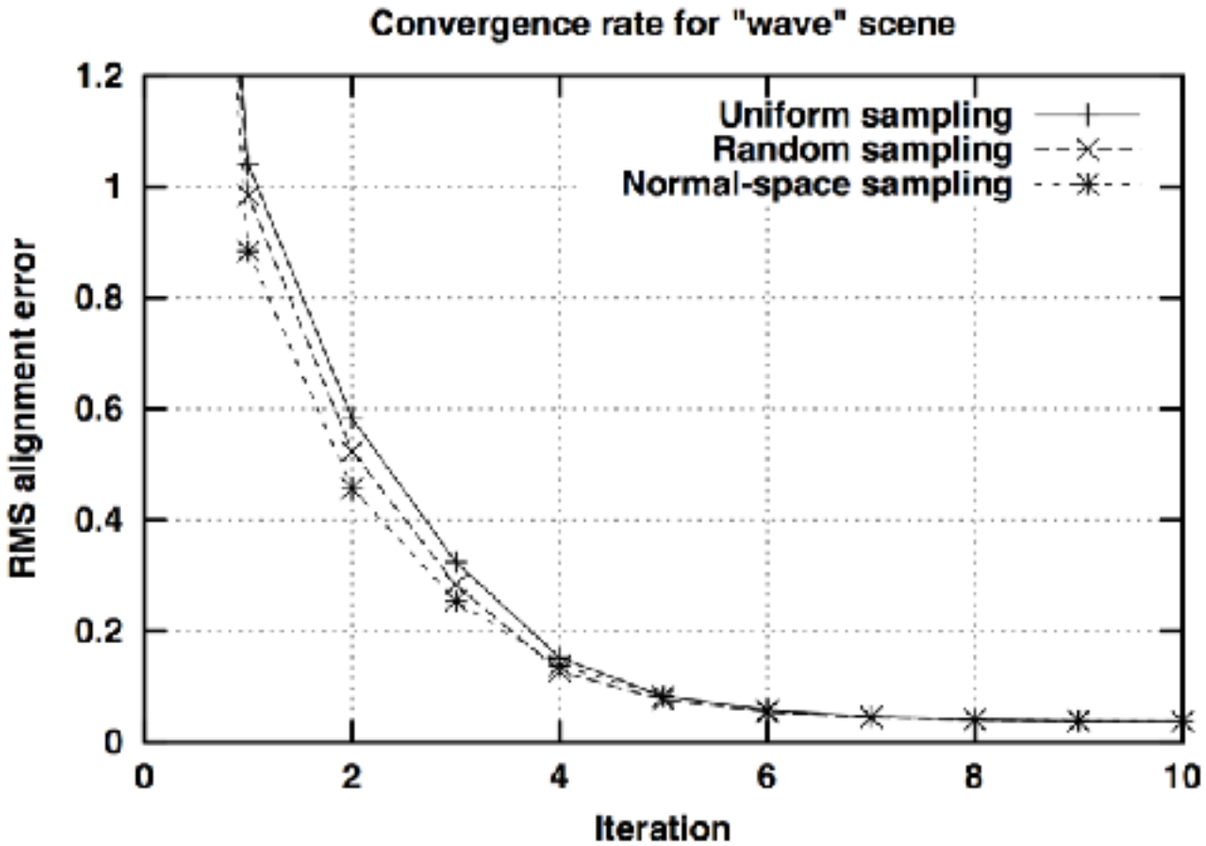


5. Auswahl basierende auf der Flächennormale

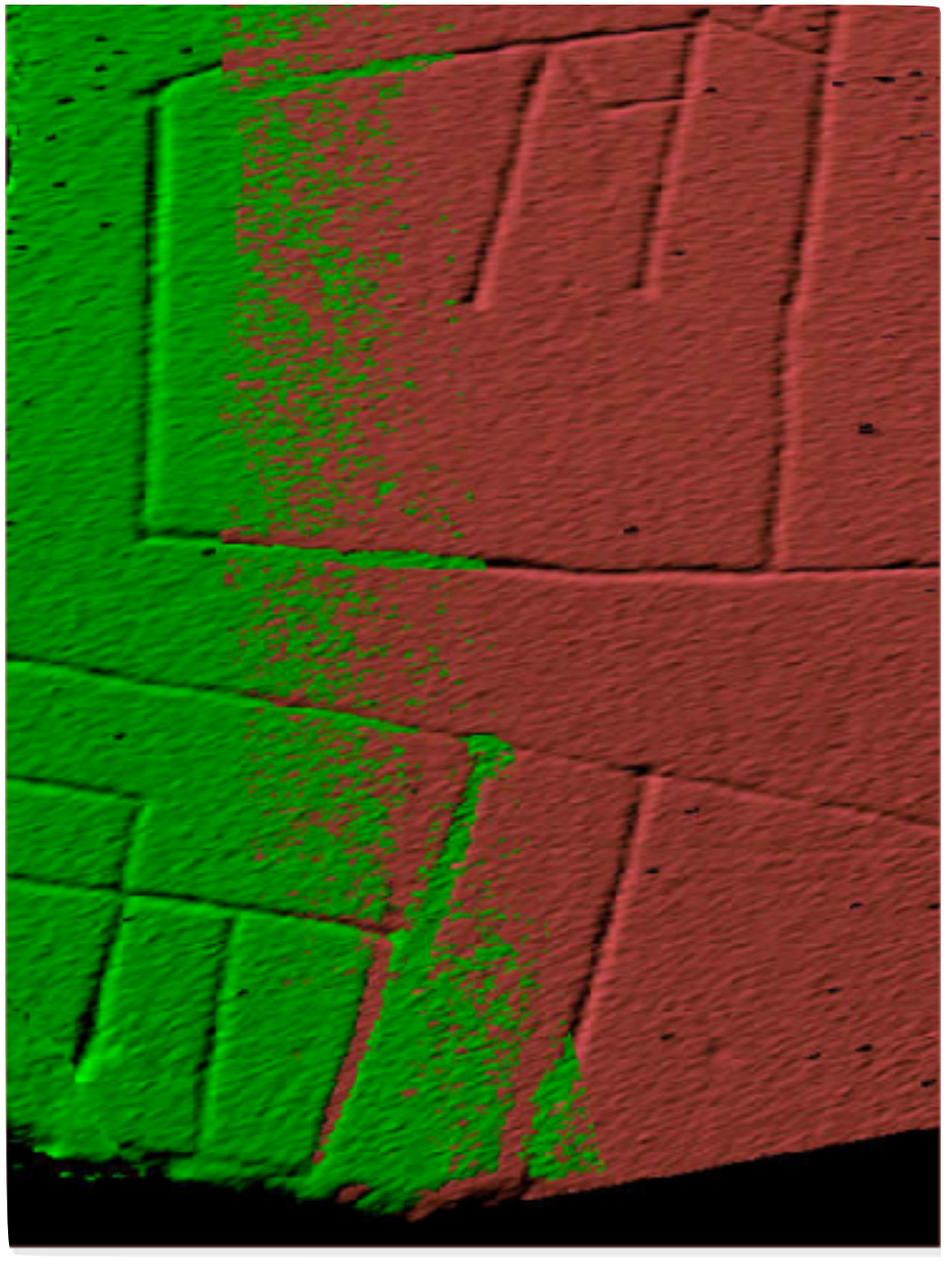
Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP) — Selection

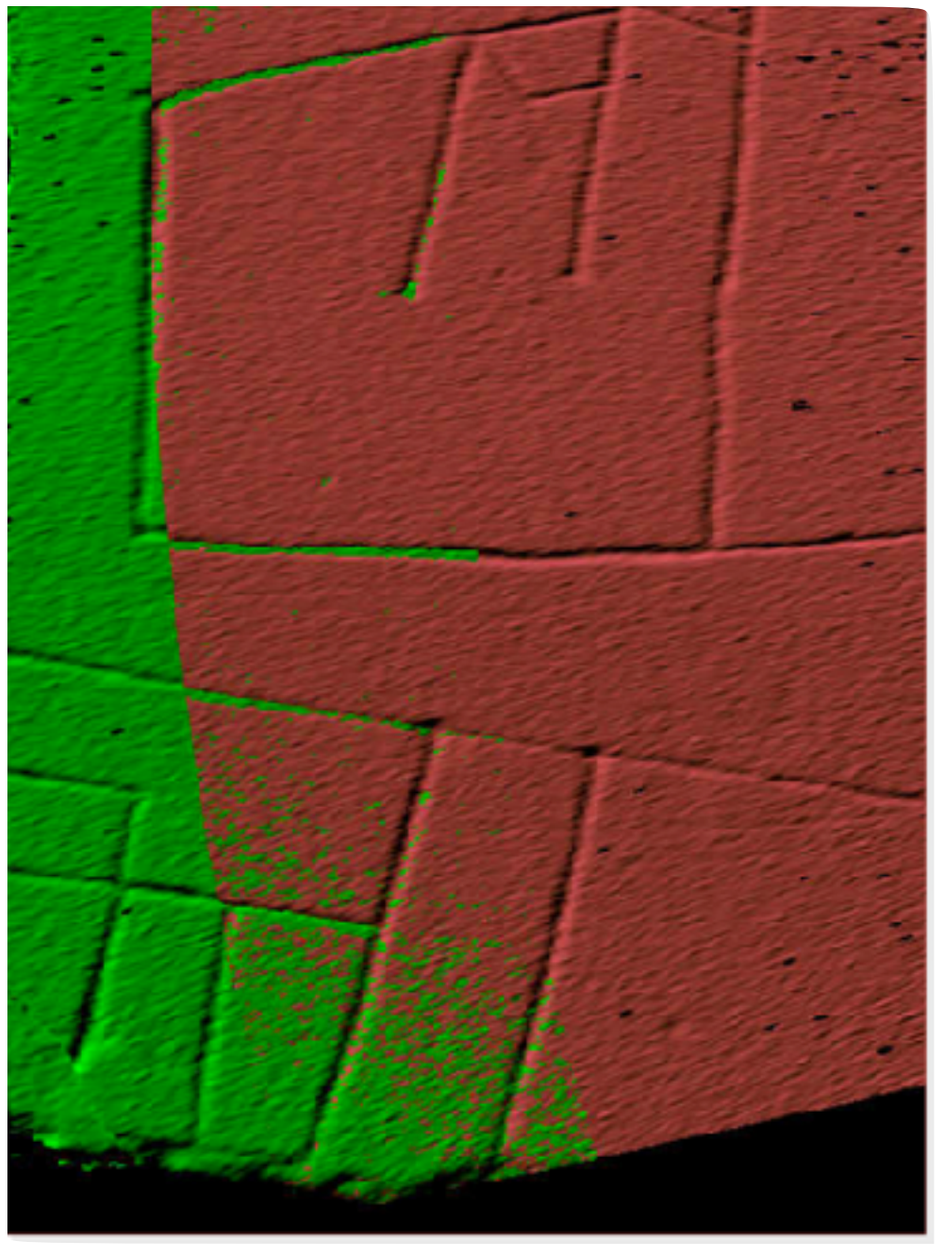




Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001



Random sampling

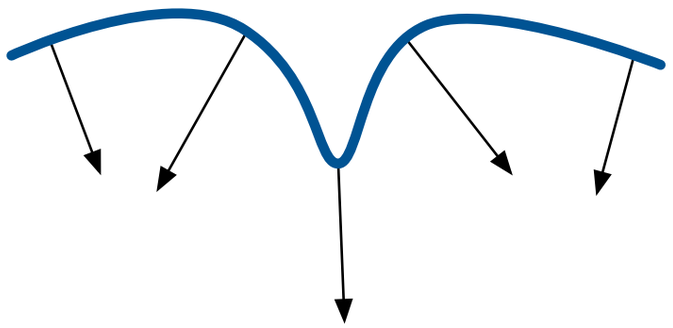


Normal-space sampling

Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

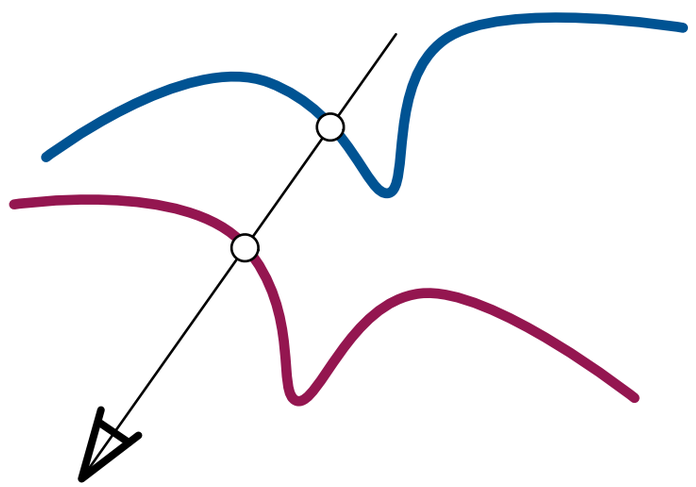
Iterative Closest Point (ICP) — Matching

1. Punkte mit dem geringsten Abstand



2. Normal shooting

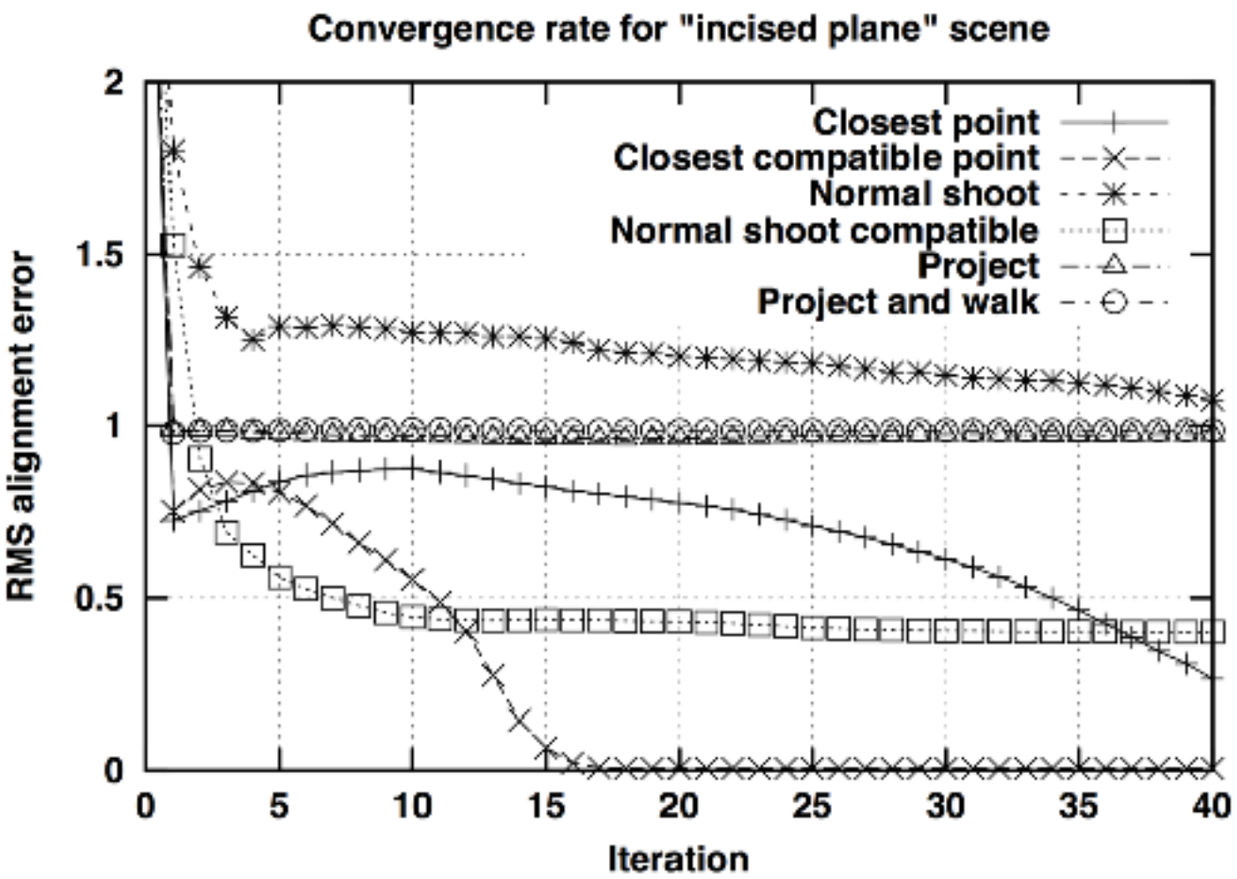
3. View point projection

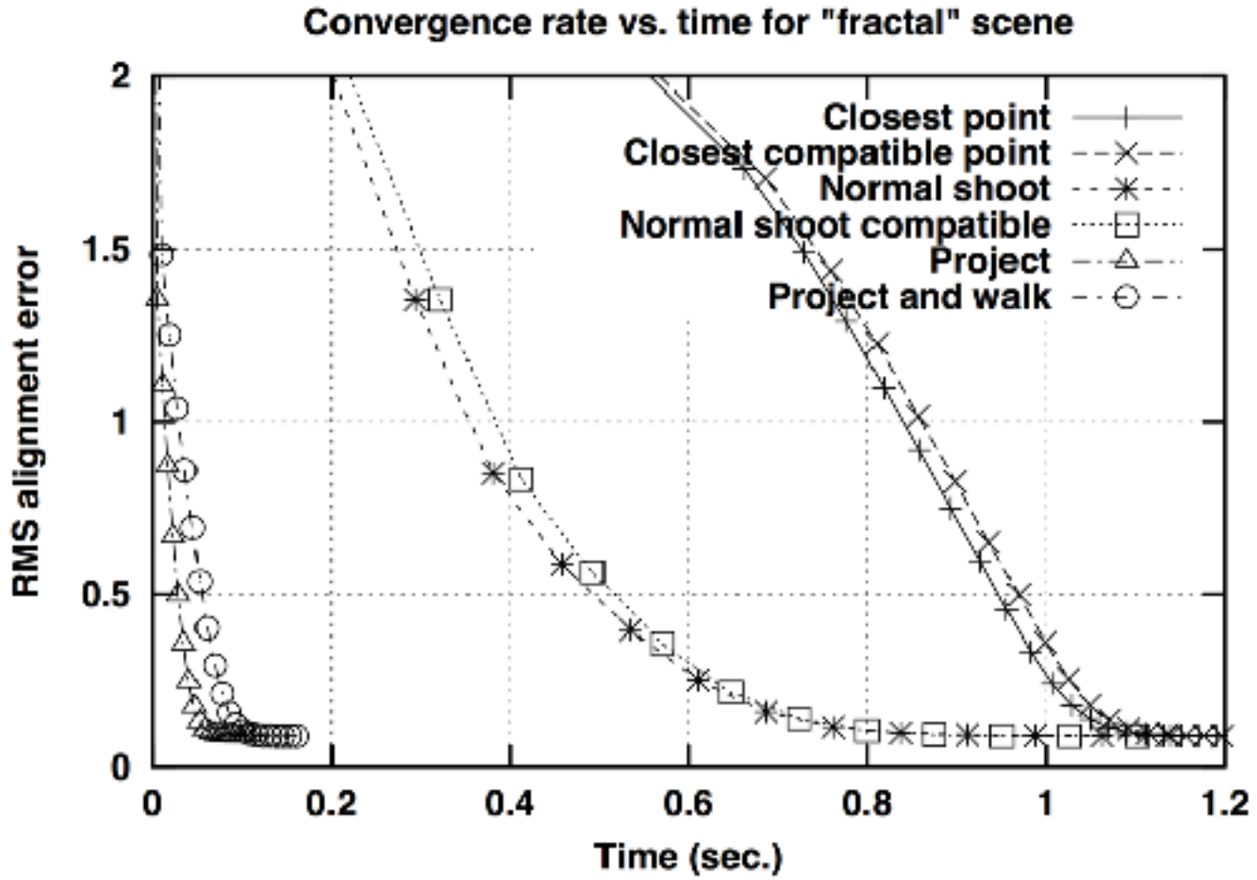


4. Closest compatible point
   (z.B. Normalen, Farbe, Curvature, …)

Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP) — Selection





Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP) — Weighting

1. Konstantes Gewicht

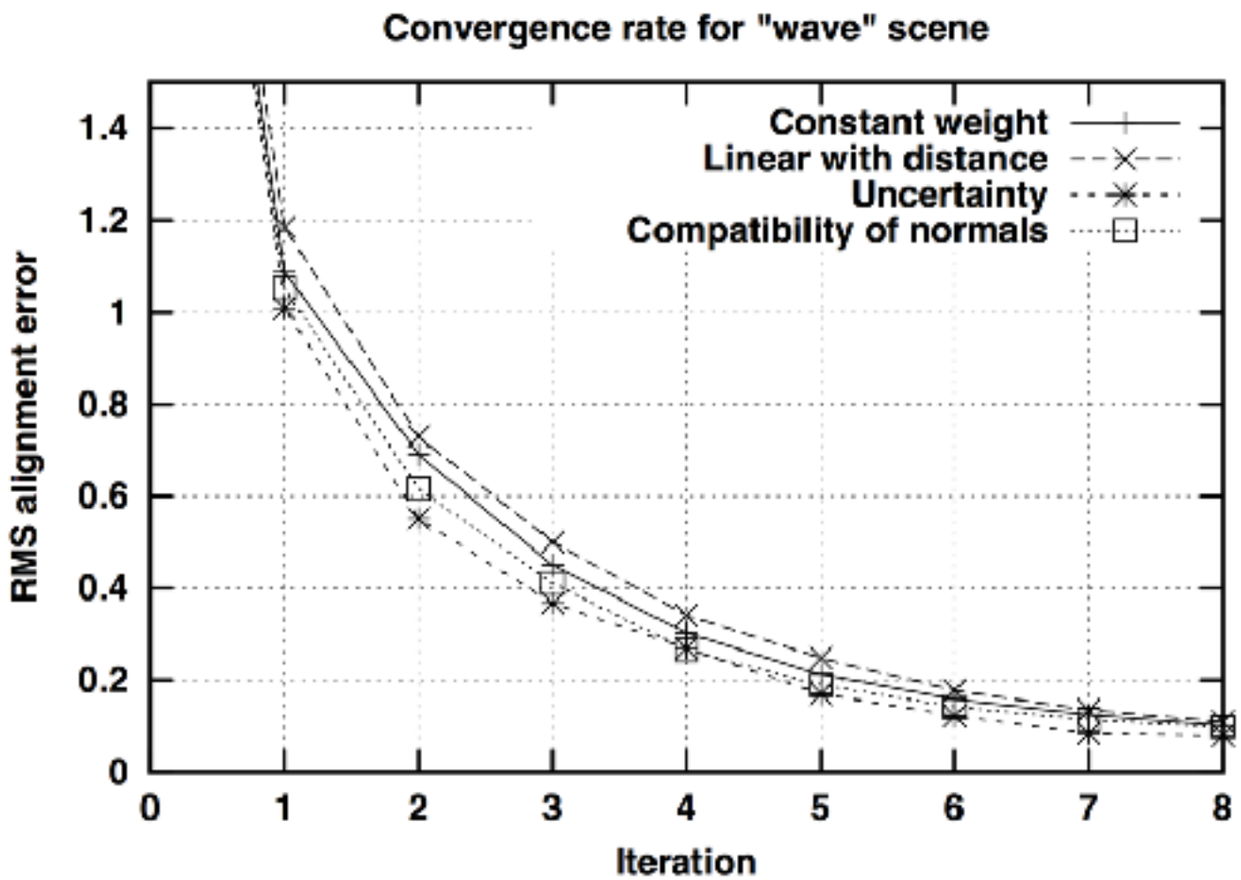
2. Abstandsabhängiges Gewicht $w_{12} = 1 - \frac{\mathrm{Dist}(p_1, p_2)}{\max \mathrm{Dist}(\mathrm{p_i}, \mathrm{p_j})}$

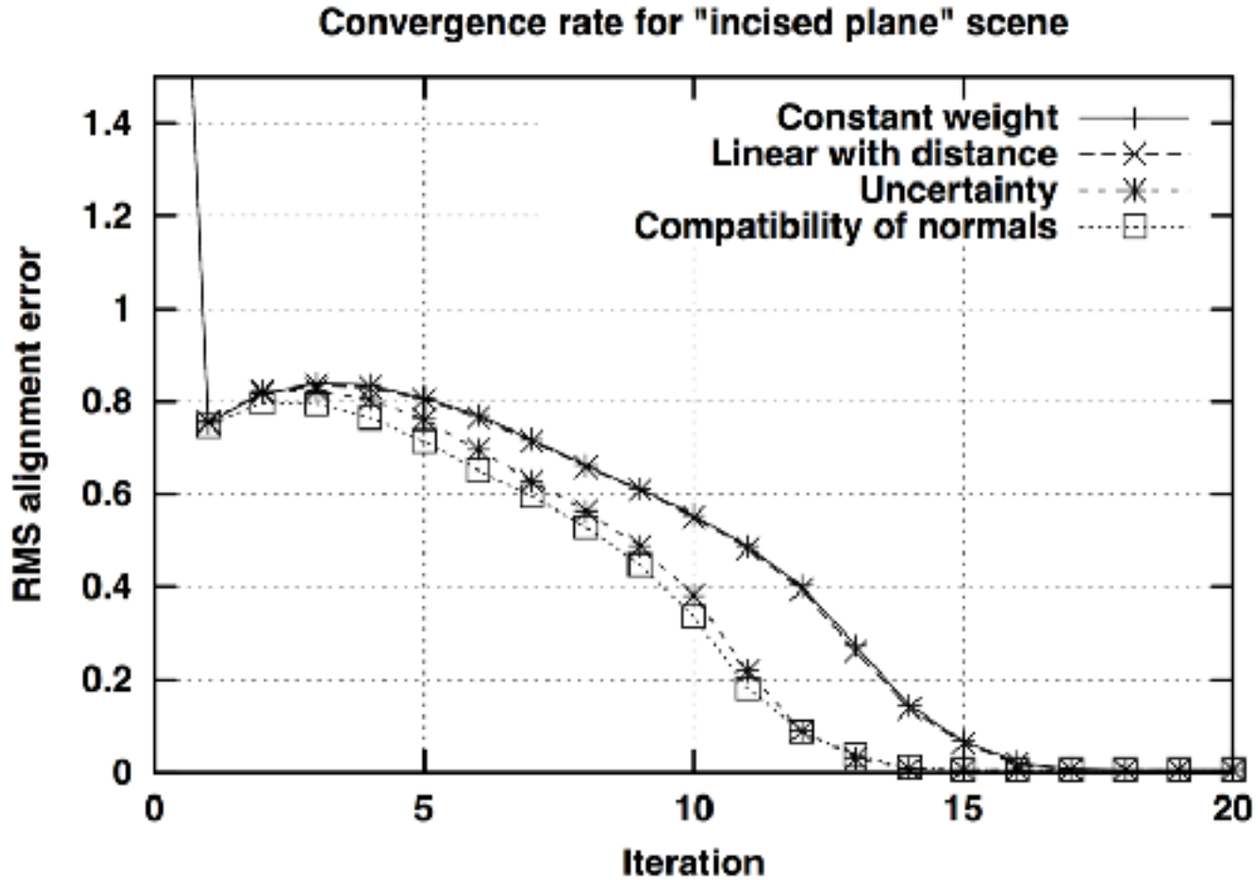
3. Basierend auf den Normalen $w_{12} = n_1 \cdot n_2$

4. Rauschbasierte Gewichtung

Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP) — Selection





Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP) — Rejecting

1. Festgelegter Maximalabstand

2. Ablehnung der n% schlechtesten Matches (z.b. 10% der am weitesten entfernten Punkten)

3. Abstand kleiner als ein Vielfaches der Standardabweichung (z.B. 2.5σ)

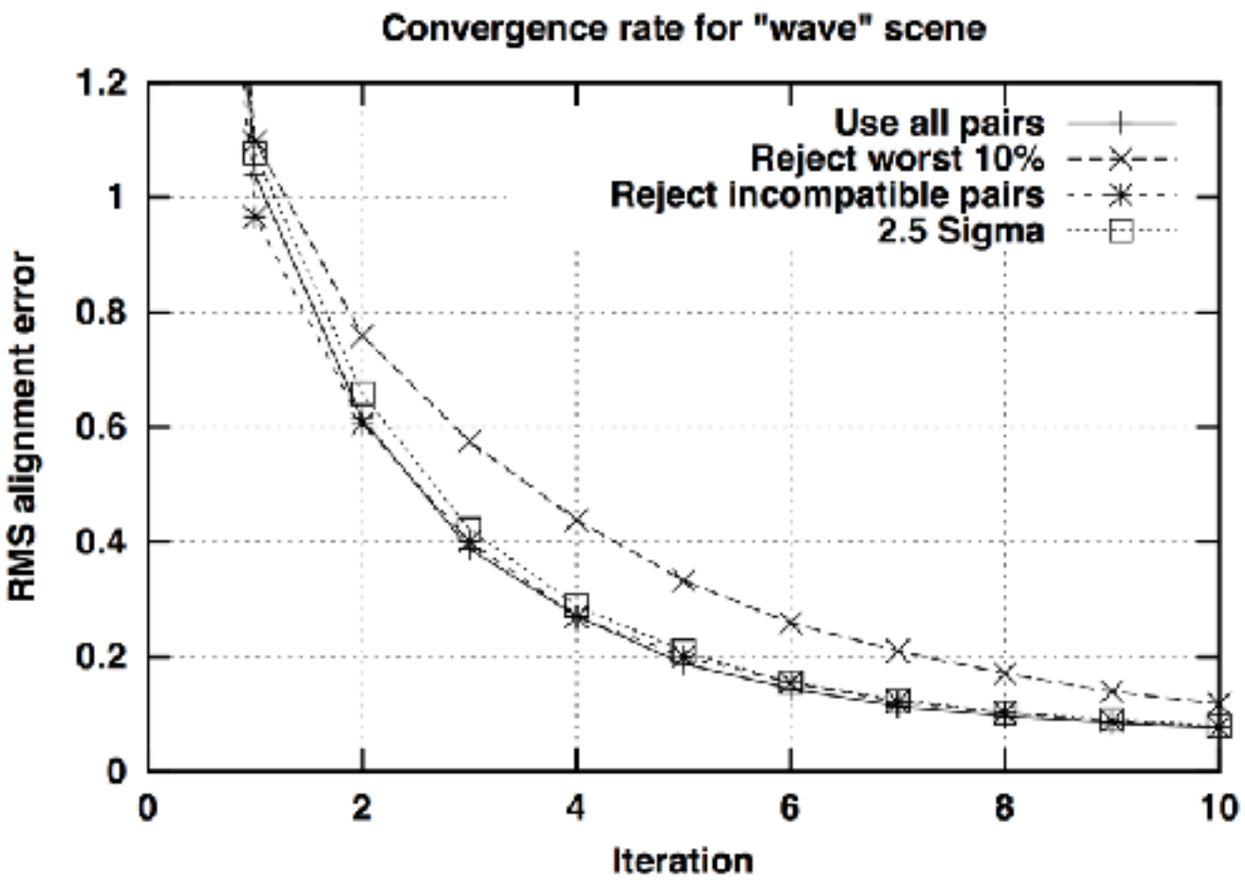
4. Ablehnung, wenn benachbarte Punkte sich zu stark unterscheiden

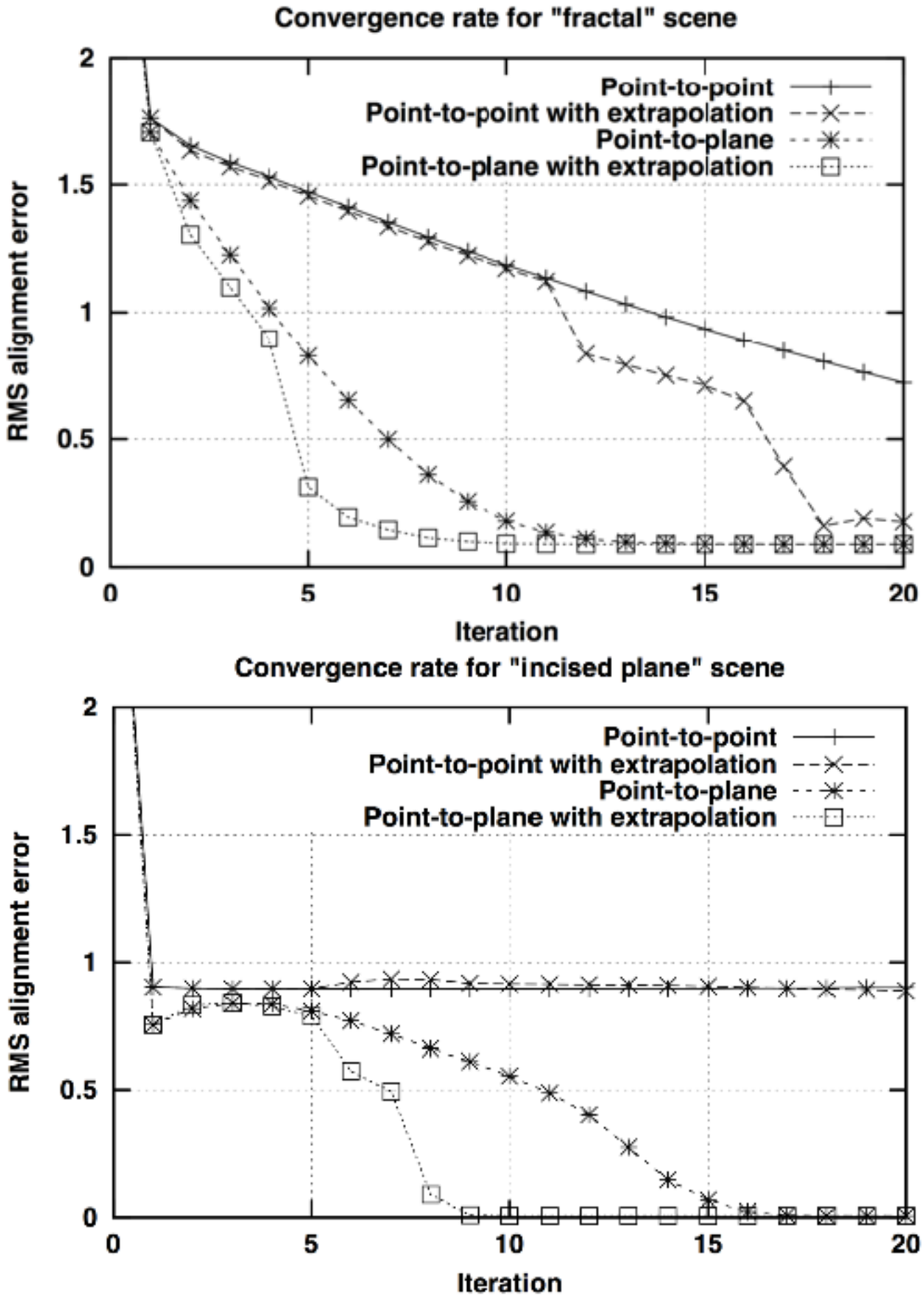
$$(x_1, y_1), (x_2, y_2) : |\text{Dist}(x_1, x_2) - \text{Dist}(y_1, y_2)| > \gamma \max(\text{Dist}(x_i, x_j), \text{Dist}(y_k, y_l)), \quad \gamma = 0.1$$

5. Keine Punkte, die am Rand der Meshes liegen

Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

Iterative Closest Point (ICP) — Selection





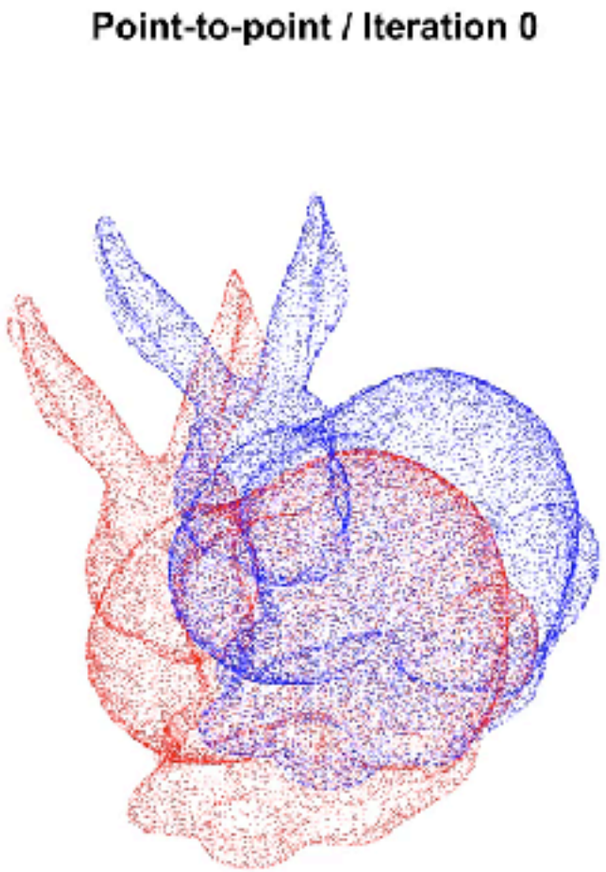
Efficient Variants of the ICP Algorithm, Rusinkiewicz, 2001

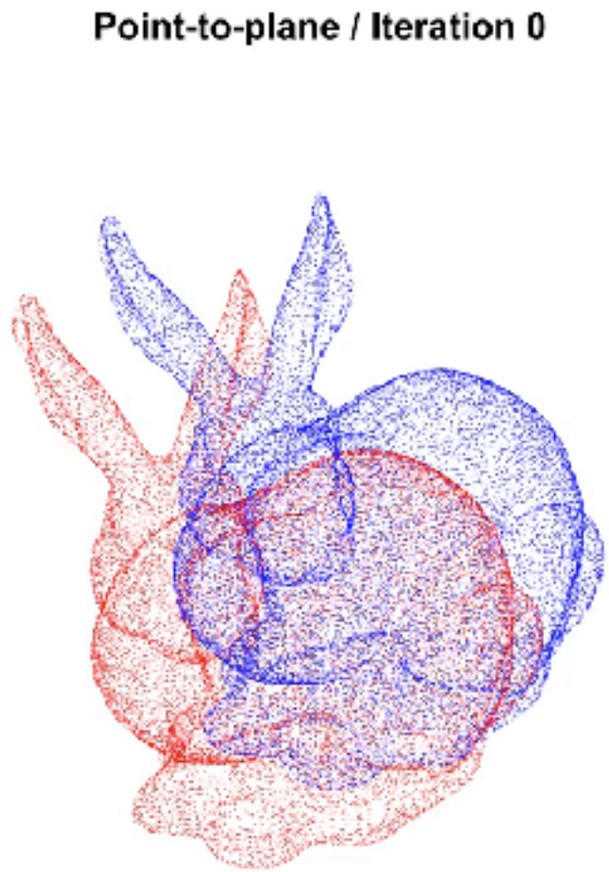
Iterative Closest Point (ICP) — Metric

1. Summe der quadratischen Abstände zusammengehöriger Punkte (SVD)
   Point-to-point

2. Summe der Abstände, geometrisch und Farben

3. Summe der quadratischen Abstände zusammengehöriger Punkte, gewichtet mit der Flächennormalen
   Point-to-plane





https://www.youtube.com/watch?v=LcghboLgTiA

SLAM — Extended Kalman Filter (EKF SLAM)

- Metric, feature-based Verfahren
- Ein Vektor um den Zustand des Roboters und die Umgebung zu repräsentieren
- Wenn neue Features entdeckt werden, wird der Vektor erweitert

$$\hat{y}_k = A\,y_{k-1} + B\,u_k$$
$$\hat{P}_k = A\,P_{k-1}A^T + Q$$
$$K = \hat{P}_k H^T (H\hat{P}_k H^T + R)^{-1}$$
$$y_k = \hat{y}_k + K(z_k - H\hat{y}_k)$$
$$P_k = (\mathbf{1} - KH)\hat{P}_k$$

H – Abbildung der Sensordaten auf den Zustand
A – Modell des Systems
B – Modell der Kontrolle
Q – Covariance des Sensorfehlers
R – Rauschen der Sensoren

konstant

P – Schätzung des Fehlers
y – Zustand
u – Steuerbefehle
x – Messung

aktualisiert

- Problem: Die Covariance-Matrix Q wächst quadratisch
- Repräsentation der Unsicherheit des Zustands erfolgt durch eine multivariate Gauss-Verteilung
- Linearisierung des Modells A mit First-order Taylor Expansion (Jacobian Matrix)

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

EKF-SLAM

$$
\begin{aligned}
p(x_t, m | z_{1:t}, u_{1:t}) &\sim \mathcal{N}(s_t; \mu_t, \Sigma_t) \\
\mu_t &\in \mathbb{R}^{3+2N} && \text{Pose des Roboters + Position aller Objekte} \\
\Sigma_t &\in \mathbb{R}^{(3+2N)\times(3+2N)} && \text{Unsicherheit für jede Koordinate}
\end{aligned}
$$

$$
s_t = \{x_t, x_{m_1}, y_{m_1}, x_{m_2}, y_{m_2}, \dots, x_{m_N}, y_{m_N}\}
$$

$$
\mu_t = \{\mu_{x_t}, \mu_{m_1,t}, \mu_{m_2,t}, \dots, \mu_{m_N,t}\}
$$

$$
\Sigma_t = \begin{bmatrix}
\Sigma_{x_t} & \Sigma_{x_t m_1,t} & \Sigma_{x_t m_2,t} & \cdots & \Sigma_{x_t m_N,t} \\
\Sigma_{m_1 x_t} & \Sigma_{m_1,t} & \Sigma_{m_1 m_2,t} & \cdots & \Sigma_{m_1 m_N,t} \\
 & & \ddots & & \\
\cdots & & & & \\
\Sigma_{m_N x_t} & & \cdots & & \Sigma_{m_N,t}
\end{bmatrix}
$$

EKF-SLAM

m_5
μ_5
Σ_5
m_3
μ_3
Σ_3
m_2
μ_2
Σ_2
m_1
μ_1
Σ_1
m_6
μ_6
Σ_6
m_7
μ_7
Σ_7

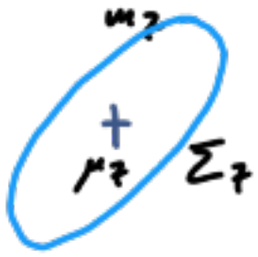
EKF-SLAM

$$s_t = \{x_t, x_{m_1}, y_{m_1}\}$$

$$\mu_t = \{\mu_{x_t}, \mu_{m_1,t}\}$$

$$\Sigma_t = \begin{bmatrix} \Sigma_{x_t} & \Sigma_{x_1^t,t} \\ \Sigma_{m_1 x_t} & \Sigma_{m_1,t} \end{bmatrix}$$

EKF-SLAM



$$s_t = \{x_t, x_{m_1}, y_{m_1}, x_{m_2}, y_{m_2}\}$$

$$\mu_t = \{\mu_{x_t}, \mu_{m_1,t}, \mu_{m_2,t}\}$$

$$\Sigma_t = \begin{bmatrix} \Sigma_{x_t} & \Sigma_{x_t m_1,t} & \Sigma_{x_t m_2,t} \\ \Sigma_{m_1 x_t} & \Sigma_{m_1,t} & \Sigma_{m_1 m_2,t} \end{bmatrix}$$

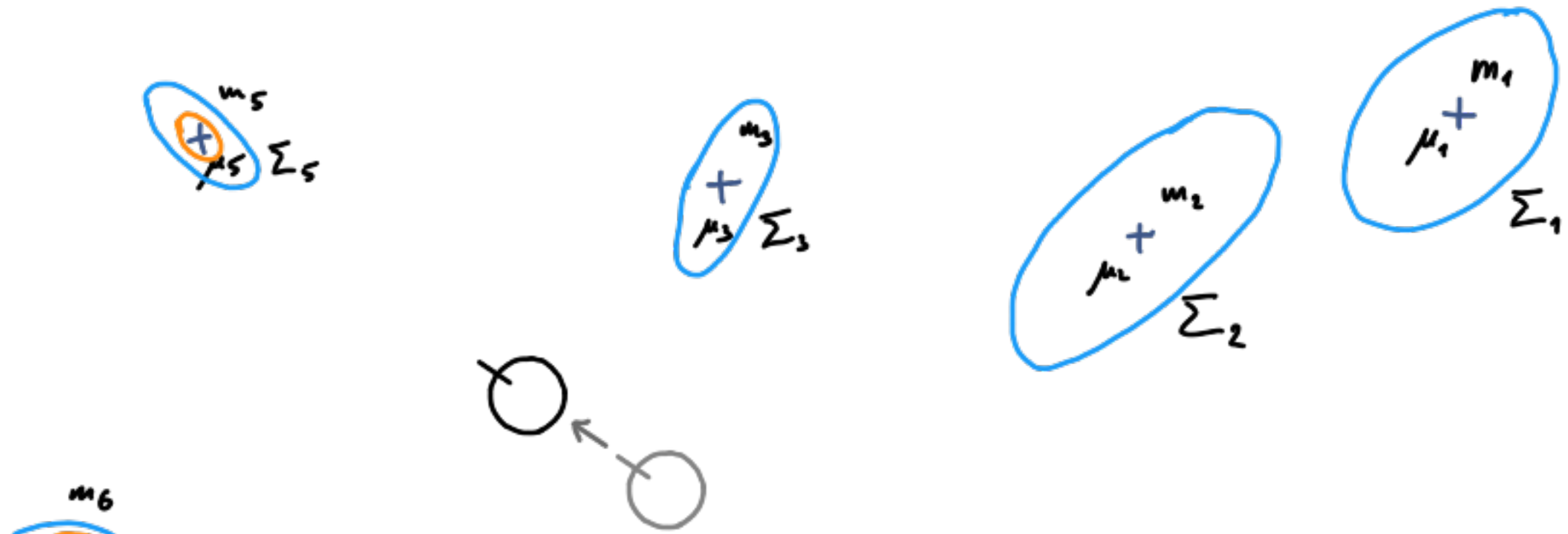
EKF-SLAM

$$s_t = \{x_t, x_{m_1}, y_{m_1}, x_{m_2}, y_{m_2}, \dots, x_{m_7}, y_{m_7}\}$$

$$\mu_t = \{\mu_{x_t}, \mu_{m_1,t}, \mu_{m_2,t}, \dots, \mu_{m_7,t}\}$$

$$\Sigma_t = \begin{bmatrix} \Sigma_{x_t} & \Sigma_{x_t m_1,t} & \Sigma_{x_t m_2,t} & \cdots & \Sigma_{x_t m_7,t} \\ \Sigma_{m_1 x_t} & \Sigma_{m_1,t} & \Sigma_{m_1 m_2,t} & \cdots & \Sigma_{m_1 m_7,t} \\ \cdots & & \ddots & & \\ \Sigma_{m_7 x_t} & & \cdots & & \Sigma_{m_7,t} \end{bmatrix}$$

EKF-SLAM

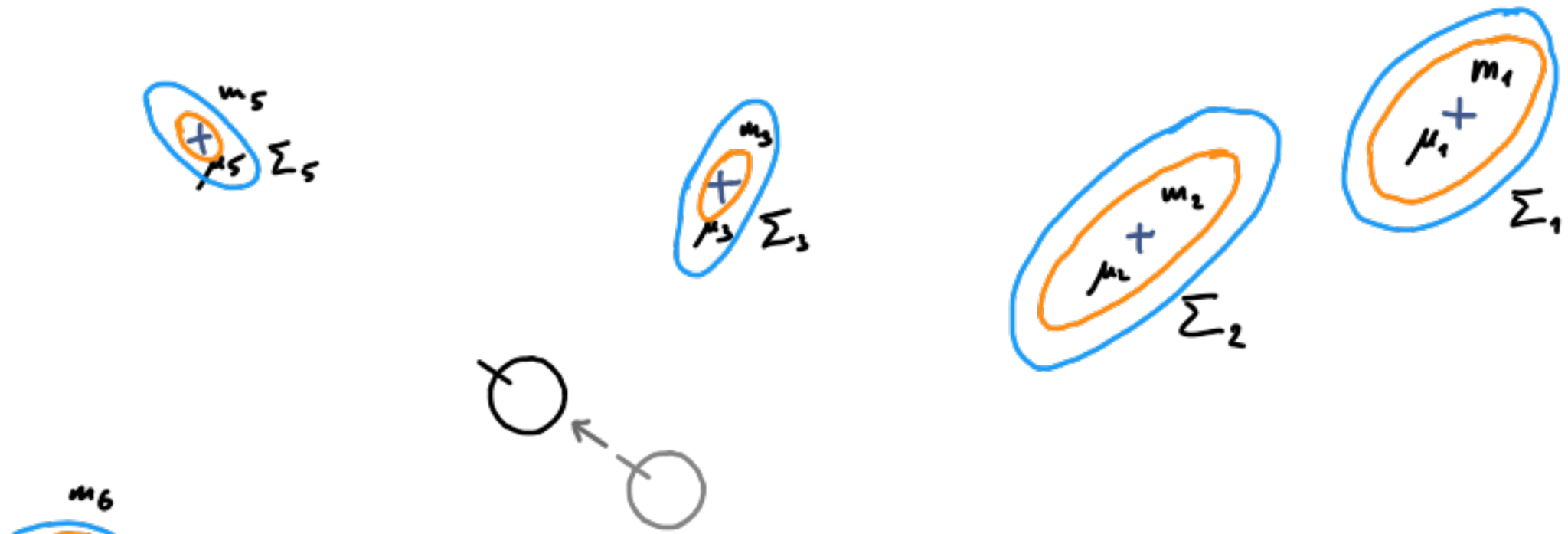


$$s_t = \{x_t, x_{m_1}, y_{m_1}, x_{m_2}, y_{m_2}, \dots, x_{m_7}, y_{m_7}\}$$

$$\mu_t = \{\mu_{x_t}, \mu_{m_1,t}, \mu_{m_2,t}, \dots, \mu_{m_7,t}\}$$

$$\Sigma_t = \begin{bmatrix} \Sigma_{x_t} & \Sigma_{x_t m_1,t} & \Sigma_{x_t m_2,t} & \cdots & \Sigma_{x_t m_7,t} \\ \Sigma_{m_1 x_t} & \Sigma_{m_1,t} & \Sigma_{m_1 m_2,t} & \cdots & \Sigma_{m_1 m_7,t} \\ & & \ddots & & \\ \cdots & & & & \\ \Sigma_{m_7 x_t} & & \cdots & & \Sigma_{m_7,t} \end{bmatrix}$$

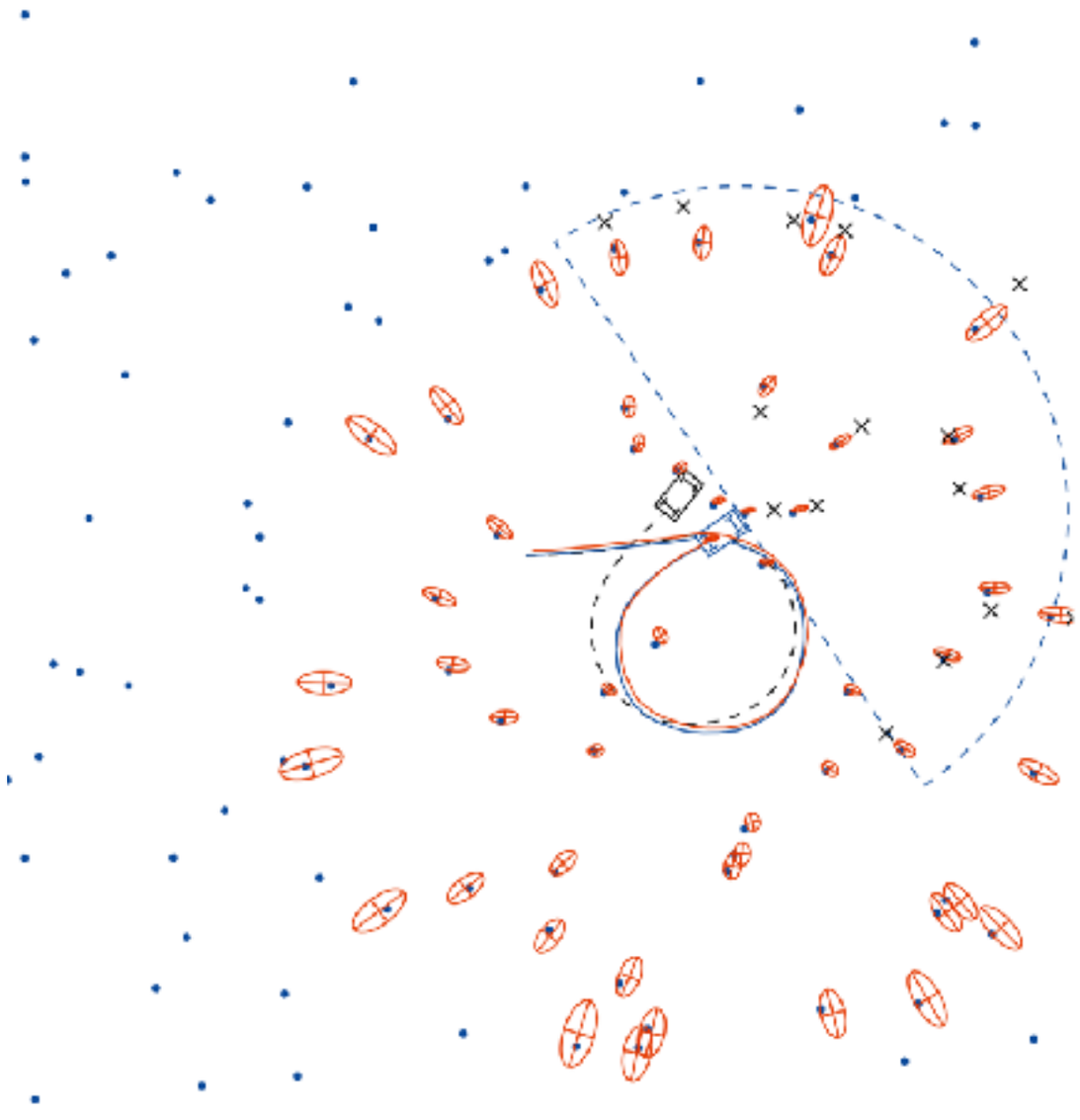
EKF-SLAM

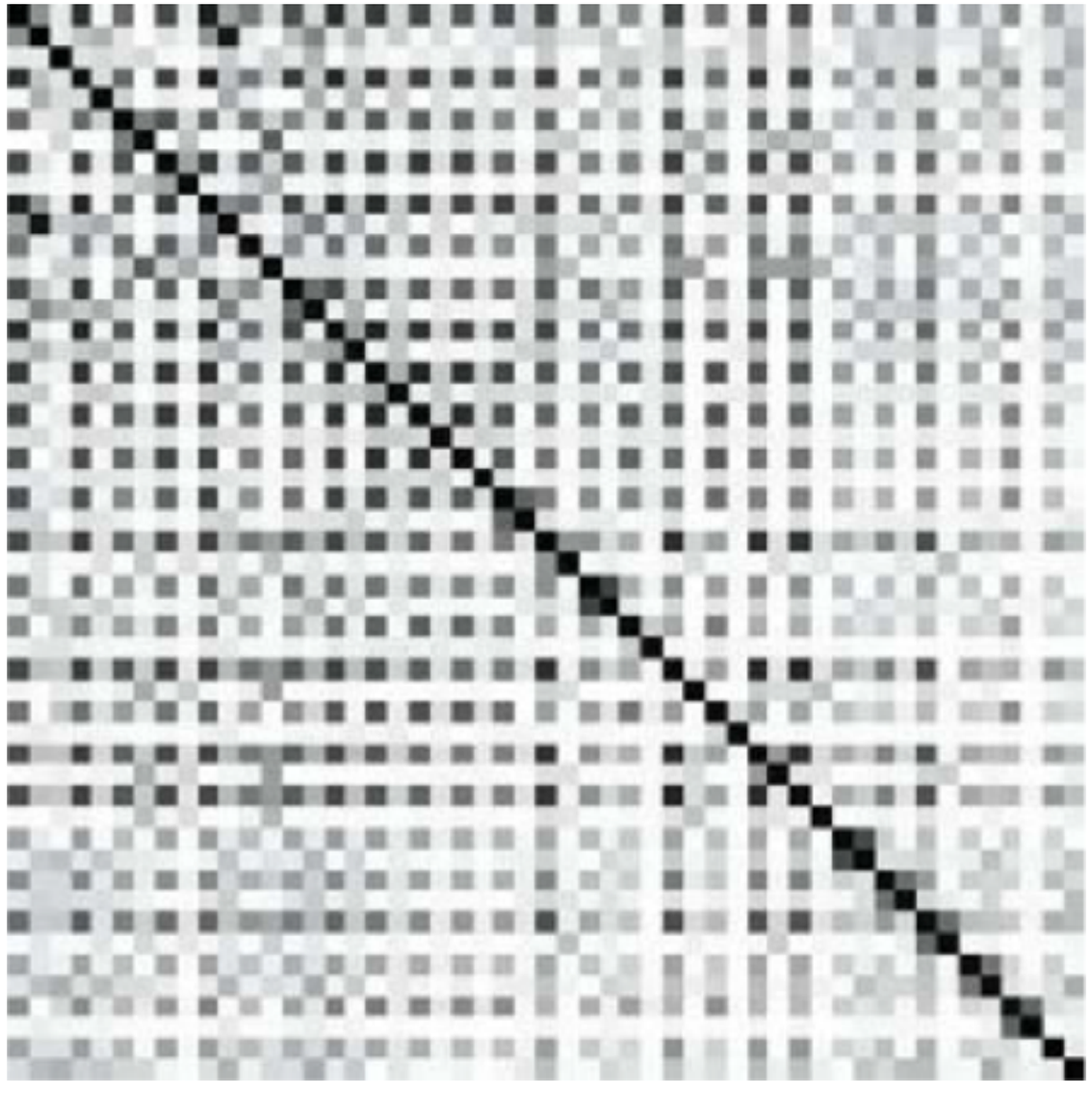


$$s_t = \{x_t, x_{m_1}, y_{m_1}, x_{m_2}, y_{m_2}, \dots, x_{m_7}, y_{m_7}\}$$

$$\mu_t = \{\mu_{x_t}, \mu_{m_1,t}, \mu_{m_2,t}, \dots, \mu_{m_7,t}\}$$

$$\Sigma_t = \begin{bmatrix} \Sigma_{x_t} & \Sigma_{x_t m_1,t} & \Sigma_{x_t m_2,t} & \cdots & \Sigma_{x_t m_7,t} \\ \Sigma_{m_1 x_t} & \Sigma_{m_1,t} & \Sigma_{m_1 m_2,t} & \cdots & \Sigma_{m_1 m_7,t} \\ \cdots & & \ddots & & \\ \Sigma_{m_7 x_t} & & \cdots & & \Sigma_{m_7,t} \end{bmatrix}$$





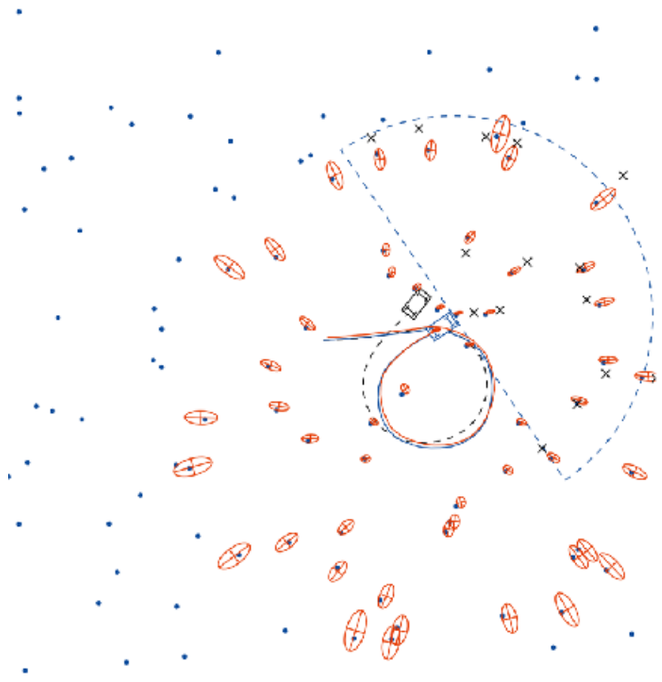
Montemerlo, Thrun, FastSLAM, 2007, Springer

Fast-SLAM

- Nutzt die strukturellen Eigenschaften aus, die in EKF-SLAM nicht beachtet werden

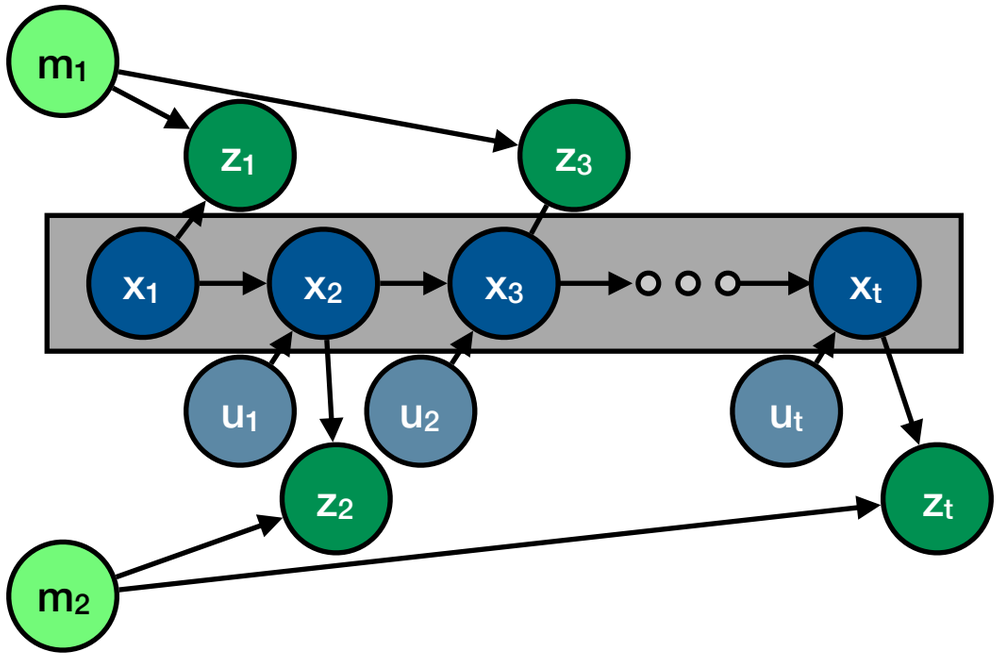
- Die Landmarken sind erst dann vollständig korreliert, wenn deren Positionen gesichert sind

- EKF macht davon keinen Gebrauch, da keine Annahmen über die Umgebung gemacht werden



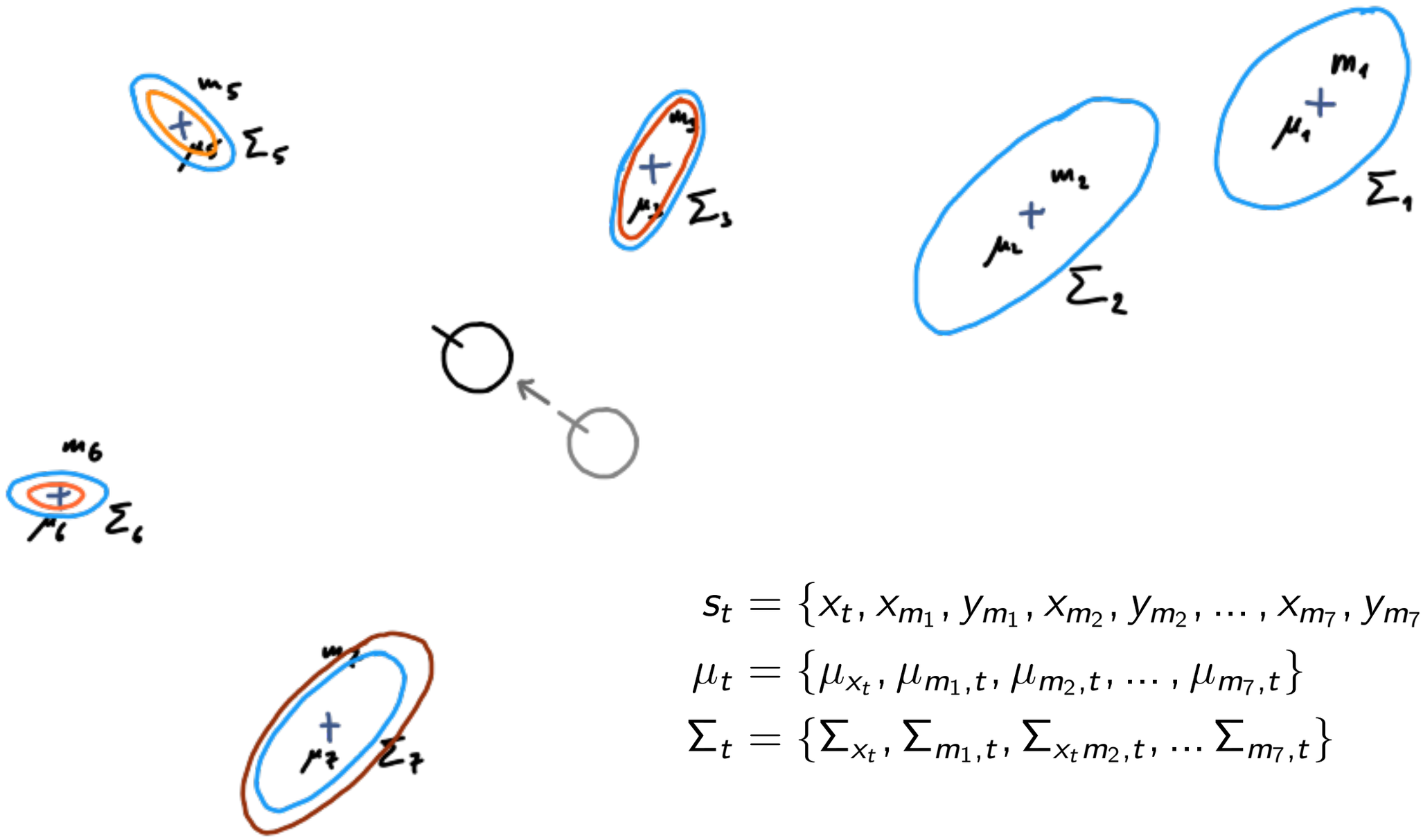
FastSLAM

- SLAM Posterior: $p(x_t, m | z_{1:t}, u_{1:t}, n_{1:t})$
- FastSLAM Posterior: $p(x_{1:t}, m | z_{1:t}, u_{1:t}, n_{1:t})$
- Erlaubt die Faktorisierung des Posteriors in einfachere Terme



- Die Position der Landmarken sind unabhängig voreinander, gegeben des Roboterpfads

Faktorisierung der Landmarken

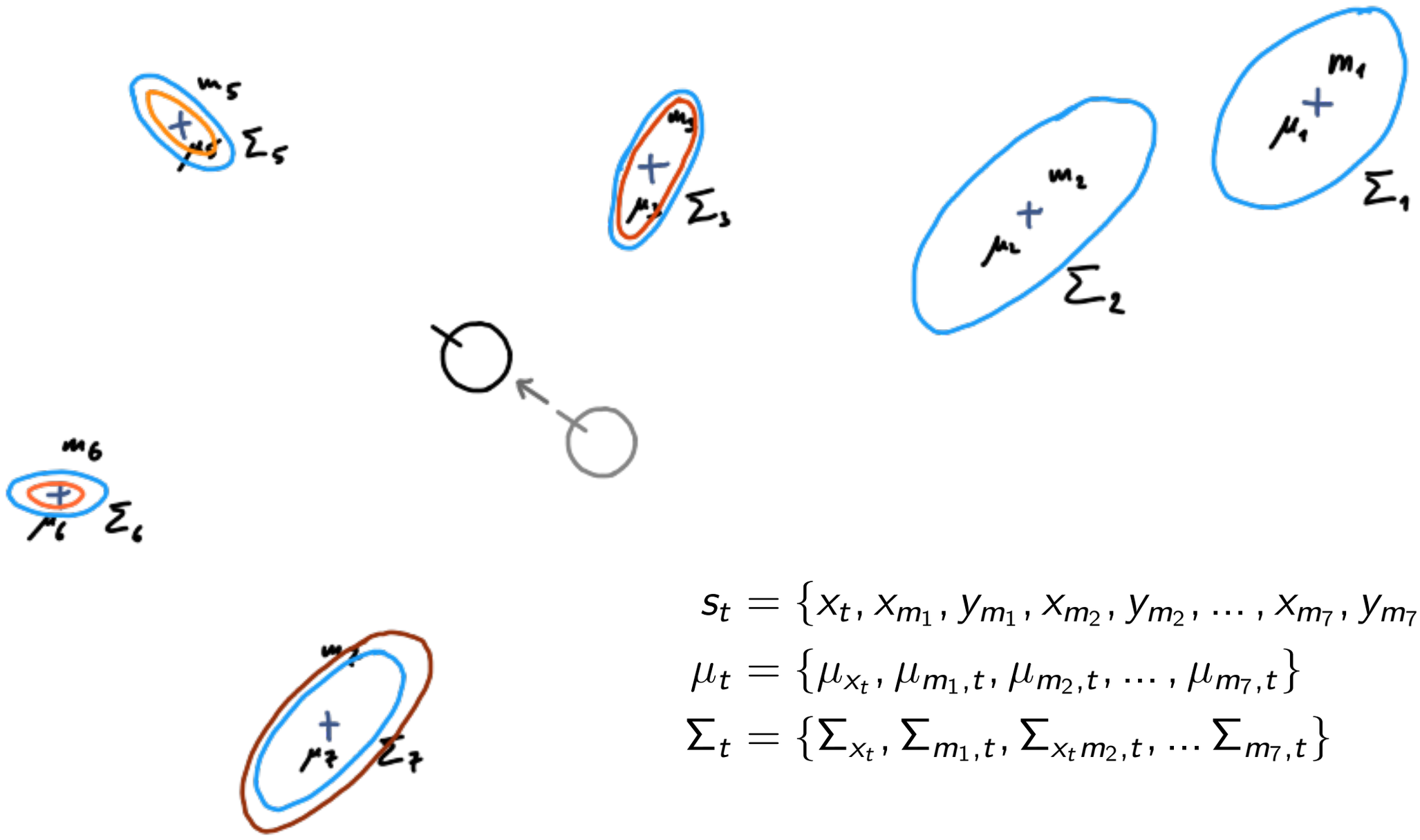


$$s_t = \{x_t, x_{m_1}, y_{m_1}, x_{m_2}, y_{m_2}, \dots, x_{m_7}, y_{m_7}\}$$

$$\mu_t = \{\mu_{x_t}, \mu_{m_1,t}, \mu_{m_2,t}, \dots, \mu_{m_7,t}\}$$

$$\Sigma_t = \{\Sigma_{x_t}, \Sigma_{m_1,t}, \Sigma_{x_t m_2,t}, \dots \Sigma_{m_7,t}\}$$

Faktorisierung der Landmarken



$$
\begin{aligned}
s_t &= \{x_t, x_{m_1}, y_{m_1}, x_{m_2}, y_{m_2}, \dots, x_{m_7}, y_{m_7}\} \\
\mu_t &= \{\mu_{x_t}, \mu_{m_1,t}, \mu_{m_2,t}, \dots, \mu_{m_7,t}\} \\
\Sigma_t &= \{\Sigma_{x_t}, \Sigma_{m_1,t}, \Sigma_{x_t m_2,t}, \dots \Sigma_{m_7,t}\}
\end{aligned}
$$

$$
p(x_{1:t}, m | z_{1:t}, u_{1:t}, n_{1:t}) = p(x_{1:t} | z_{1:t}, u_{1:t}, n_{1:t}) \prod_{n=1}^{N} p(m_n | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t})
$$

FastSLAM — Herleitung

$$p(x_{1:t}, m | z_{1:t}, u_{1:t}, n_{1:t}) = p(x_{1:t} | z_{1:t}, u_{1:t}, n_{1:t}) \prod_{n=1}^{N} p(m_n | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t})$$

Es gilt: $p(x_{1:t}, m | z_{1:t}, u_{1:t}, n_{1:t}) = p(x_{1:t} | z_{1:t}, u_{1:t}, n_{1:t}) p(m | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t})$

Zu zeigen: $p(m | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) = \prod_{n=1}^{N} p(m_n | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t})$

Zwischenschritt 1:

$$p(m_{n_t} | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) \overset{Bayes}{=} \frac{p(z_t | m_{n_t}, x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})}{p(z_t | x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})} p(m_{n_t} | x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})$$

$$p(m_{n_t} | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) = \frac{p(z_t | m_{n_t}, x_{1:t}, n_{1:t})}{p(z_t | x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})} p(m_{n_t} | x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1})$$

$$p(m_{n_t} | x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1}) = \frac{p(z_t | x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})}{p(z_t | m_{n_t}, x_{1:t}, n_{1:t})} p(m_{n_t} | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t})$$

Zwischenschritt 2:

$$p(m_{n \neq n_t} | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) = p(m_{n \neq n_t} | x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1})$$

FastSLAM — Beweis per Induktion

Induktionshypothese für t-1:

$$p(m|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1}) = \prod_{n=1}^{N} p(m_n|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1})$$

Trivial für t = 1: Da keine Beobachtungen gemacht wurden ist die Faktorisierung richtig.

Für t > 1:

$$p(m|x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) = \frac{p(z_t|m, x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})}{p(z_t|x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})} p(m|x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})$$

$$\text{[Markov]} \quad = \frac{p(z_t|m_{n_t}, x_t, n_t)}{p(z_t)|x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})} p(m|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1})$$

$$\text{[Induktionshypothese]} \quad = \frac{p(z_t|m_{n_t}, x_t, n_t)}{p(z_t)|x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t})} \prod_{n=1}^{N} p(m_n|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1})$$

$$\text{[Zwischenergebnisse]} \quad = p(m_{n_t}|x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) \prod_{n \neq n_t}^{N} p(m_n|x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t})$$

$$= \prod_{n \neq n_t}^{N} p(m_n|x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) \qquad \square$$

FastSLAM — Faktorisierung

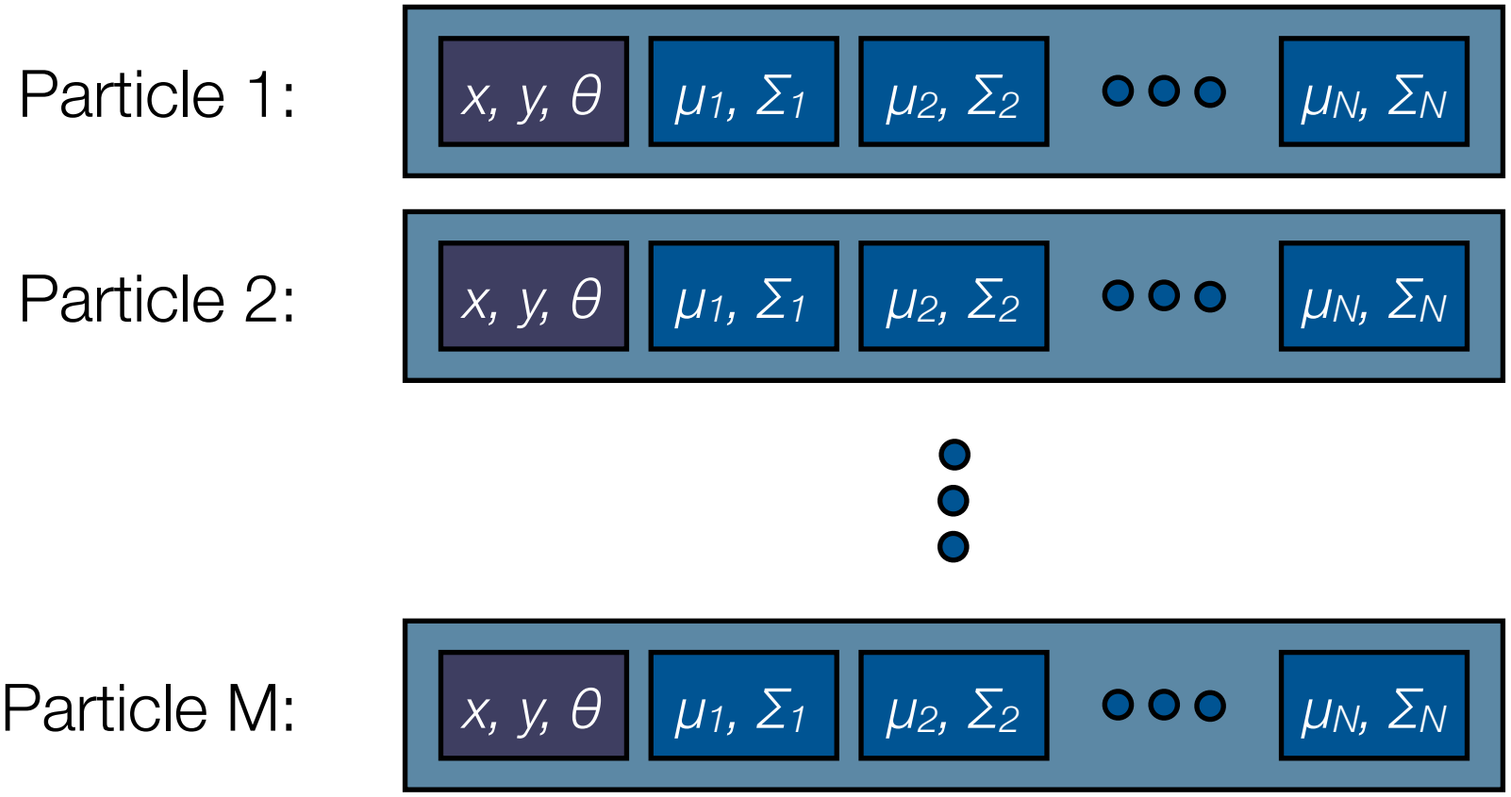
- Erlaubt die Schätzung der Position und Umgebung, ohne die Korrelationen zwischen den Landmarken explizit zu beachten

- Es werden nur *N + 1* Filter benötigt, jeder mit einer erheblichen geringeren Komplexität (Dimension)

- Erlaubt die Anwendung des *Particle Filters* für SLAM

- Jedes Particle enthält den Pfad des Roboters und ein EKF für jede Landmarke

FastSLAM — Particles & Algorithmus



1. Bestimme (*sample*) neue Roboterpose für jedes Partikel gemäß der Kontrolle *u*
2. Updaten der Landmarken-EKFs für die aktuellen Sensordaten *z*
3. Berechnen der *Importance Weights* für jedes Particle
4. Updaten der Partikel aufgrund der *Importance Weights*

FastSLAM — Neue Roboterpose

- *Importance Sampling*: Verwenden einer bekannte Verteilung zum Generieren von neuen Particles (Hypothesen), weil wir die eigentliche Verteilung nicht kennen

- Gewichten die neuen Partikels nach der Aussagekraft, die sie haben

- *Proposal Distribution*: $x_t^{[m]} \sim p(x_t | u_t, x_{t-1}^{[m]})$

- Bewegungsmodell kann jede Form (z.B. nicht-linear) annehmen

- Üblicherweise:

$$v_t' \sim \mathcal{N}(v_t, \alpha_1 v_t + \alpha_2)$$
$$\omega_t' \sim \mathcal{N}(\omega_t, \alpha_3 \omega_t + \alpha_4)$$

FastSLAM — Landmarken Update

- Landmarkenschätzungen $p(m_n|x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t})$ werden als low-dimensional EKF repräsentiert

$$\begin{aligned} p(m_n|x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) &= \eta\, p(z_t|m_{n_t}, x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t}) p(m_{n_t}|x_{1:t}, z_{1:t-1}, u_{1:t}, n_{1:t}) \\ &= \eta\, \underbrace{p(z_t|m_{n_t}, x_t, z_t, n_t)}_{(1)} \underbrace{p(m_{n_t}|x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1})}_{(2)} \end{aligned}$$

(1) $p(z_t|m_{n_t}, x_t, z_t, n_t) \sim h(x_t, m_{n_t})$

$$\begin{aligned} \hat{z}_t &= h(x_t^{[m]}, \mu_{n_t,t-1}) \\ G_{m_{n_t}} &= \nabla_{m_{n_t}} h(x_t, m_{n_t})|_{x_t=x_t^{[m]}; m_{n_t}=\mu_{n_t,t-1}^{[m]}} \\ h(x_t, m_{n_t}) &\approx \hat{z}_t + G_m(m_{n_t} - \mu_{n_t,t-1}^{[m]}) \end{aligned}$$

$$p(z_t|m_i, x_t, n_t) \sim \mathcal{N}(z_t, \hat{z}_t + G_m(m_{n_t} - \mu_{n_t,t-1}^{[m]}), R_t)$$

FastSLAM — Landmarken Update

(2) $p(m_i|s_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1}) \sim \mathcal{N}(m_{n_t}; \mu^{[m]}_{n_t,t-1}, \Sigma^{[m]}_{n_t,t-1})$

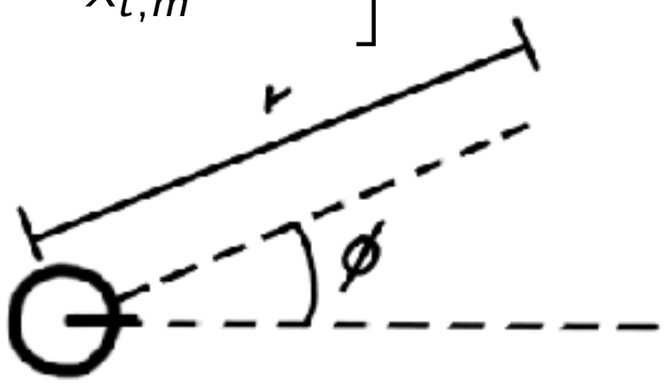
$$\hat{z}_t = h(x_t^{[m]}, \mu_{n_t,t-1})$$
$$G_{m_{n_t}} = \nabla_{m_{n_t}} h(x_t, m_{n_t})|_{x_t=x_t^{[m]}; m_{n_t}=\mu^{[m]}_{n_t,t-1}}$$
$$Z_{n,t} = G_{m_{n_t}} \Sigma^{[m]}_{n_t,t-1} G^T_{m_t} + R_t$$
$$K_t = \Sigma^{[m]}_{n_t,t-1} G^T_{m_{n_t}} Z^{-1}_{n,t}$$
$$\mu^{[m]}_{n_t,t} = \mu^{[m]}_{n_t,t-1} + K(z_t - \hat{z}_t)$$
$$\Sigma^{[m]}_{n_t,t} = (I - K_t G_{m_{n_t}})\Sigma^{[m]}_{n_t,t-1}$$

$$\hat{y}_k = A\, y_{k-1} + B\, u_k$$
$$\hat{P}_k = A\, P_{k-1} A^T + Q$$
$$K = \hat{P}_k H^T (H \hat{P}_k H^T + R)^{-1}$$
$$y_k = \hat{y}_k + K(z_k - H\hat{y}_k)$$
$$P_k = (\mathbf{1} - KH)\hat{P}_k$$

$$h(x_t, m_{n_t}) = \begin{bmatrix} r(x_t, m_{n_t}) \\ \phi(x_t, m_{n_t}) \end{bmatrix} = \begin{bmatrix} \sqrt{(m_{n_t,y} - x_{t,x})^2 + (m_{n_t,y} - x_{t,y})^2} \\ \tan^{-1}\left(\frac{m_{n_t,y} - x_{t,y}}{m_{n_t,x} - x_{t,x}}\right) - x_{t,m} \end{bmatrix}$$

$$G_{m_{n_t}} = \begin{bmatrix} \frac{m_{n_t,x} - x_{t,x}}{\sqrt{q}} & \frac{m_{n_t,y} - x_{t,y}}{\sqrt{q}} \\ -\frac{m_{n_t,y} - x_{t,y}}{q} & \frac{m_{n_t,x} - x_{t,x}}{q} \end{bmatrix}$$

$$q = (m_{n_t,x} - x_{t,x})^2 + (m_{n_t,y} - x_{t,y})^2$$



FastSLAM — *Importance Weights*

- Proposal Distribution: $p(x_{1:t}^{[m]} | z_{1:t-1}, u_{1:t}, n_{1:t-1})$

- Desired Posterior: $p(x_{1:t}^{[m]} | z_{1:t}, u_{1:t}, n_{1:t})$

$$\omega_t^{[m]} = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{p(x_{1:t}^{[m]} | z_{1:t}, u_{1:t}, n_{1:t})}{p(x_{1:t}^{[m]} | z_{1:t-1}, u_{1:t}, n_{1:t-1})}$$

- Zähler mit Bayes entwickeln (Normalisierung erfolgt beim Resampling)

$$\begin{aligned}
\omega_t^{[m]} &= \frac{p(z_t | x_{1:t}^{[m]}, z_{1:t-1}, u_{1:t}, n_{1:t}) p(x_{1:t}^{[m]} | z_{1:t-1}, u_{1:t}, n_{1:t})}{p(x_{1:t}^{[m]} | z_{1:t-1}, u_{1:t}, n_{1:t-1})} \\
&= \frac{p(z_t | x_{1:t}^{[m]}, z_{1:t-1}, u_{1:t}, n_{1:t}) p(x_{1:t}^{[m]} | z_{1:t-1}, u_{1:t}, n_{1:t-1})}{p(x_{1:t}^{[m]} | z_{1:t-1}, u_{1:t}, n_{1:t-1})} \\
&= p(z_t | x_{1:t}^{[m]}, z_{1:t-1}, u_{1:t}, n_{1:t})
\end{aligned}$$

FastSLAM — *Importance Weights*

- Berechnung der Importance Weights

$$\omega_t^{[m]} = \frac{1}{\sqrt{|2\pi Z_{n_t,t}|}} \exp\left(-\frac{1}{2}(z_t - \hat{z}_{n_t,t})^T (Z_{n_t,t})^{-1}(z_t - \hat{z}_{n_t,t})\right)$$

- Abhängigkeit vom Roboterpfad $x_{1:t}$ kann ignoriert werden, da alle Funktionen für das Update der Landmarken nur von x_t abhängen

- ***Data Association Problem*:**

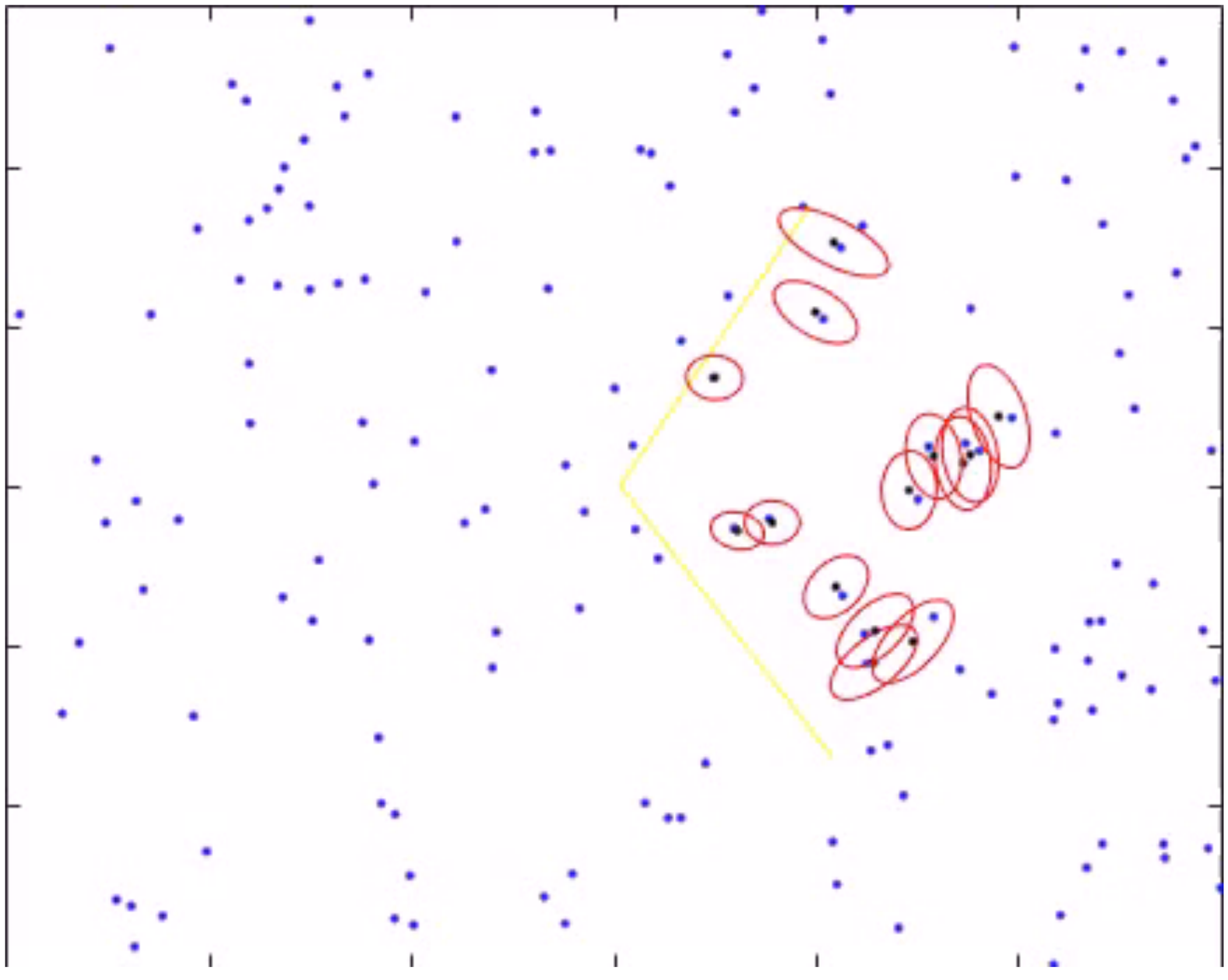
$$\hat{n}_t = \arg\max_{n_t} p\left(z_t \middle| n_t, \hat{n}_{1:t-1}, x_{1:t}, z_{1:t-1}, u_{1:t}\right)$$

input: X_{t-1}, z_t, R_t, u_t
$X_t = X_{\text{aux}} = \{\}$
for $m = 1$ to M **do**
  retrieve m-th particle $\langle x_{t-1}^{[m]}, N_{t-1}^{[m]}, \mu_{1,t-1}^{[m]}, \Sigma_{1,t-1}^{[m]}, \ldots, \mu_{N_{t-1}^{[m]},t-1}^{[m]}, \Sigma_{N_{t-1}^{[m]},t-1}^{[m]} \rangle$
  from X_{t-1}
  draw $x_t^{[m]} \sim p(x_t | s_{t-1}^{[m]}, u_t)$
  for $n = 1$ to $N_{t-1}^{[m]}$ **do**
    $\hat{z}_t = h(x_t^{[m]}, \mu_{n_t,t-1})$
    $G_{m_{n_t}} = \nabla_{m_{n_t}} h(x_t, m_{n_t})|_{x_t = x_t^{[m]}; m_{n_t} = \mu_{n_t,t-1}^{[m]}}$
    $Z_{n,t} = G_{m_{n_t}} \Sigma_{n_t,t-1}^{[m]} G_{m_t}^T + R_t$
    $p_{n,t}^{[m]} = \frac{1}{\sqrt{|2\pi Z_{n_t,t}|}} \exp\left(-\frac{1}{2}(z_t - \hat{z}_{n_t,t})^T (Z_{n_t,t})^{-1} (z_t - \hat{z}_{n_t,t})\right)$
  end for
  $p_{N_{t-1}^{[m]}+1,t}^{[m]} = p_0$
  $\hat{n}_t = \arg\max_n p_{n,t}^{[m]}$ or draw $\hat{n}_t$ with prob. $\sim p_{n,t}^{[m]}$

if $\hat{n}_t = N_{t-1}^{[m]} + 1$ **then**
  $N_t^{[m]} = N_{t-1}^{[m]} + 1$
  $\mu_{\hat{n}_t,t}^{[m]} = h^{-1}(x_t^{[m]}, \hat{z}_{\hat{n}_t,t})$
  $\Sigma_{\hat{n}_t,t}^{[m]} = (G_{m_{n_t}}^T R^{-1} G_{m_{n_t}})^{-1}$
else
  $N_t^{[m]} = N_{t-1}^{[m]}$
  $K_{\hat{n}_t,t} = \Sigma_{\hat{n}_t,t-1}^{[m]} G_{m_{n_t}}^T Z_{n,t}^{-1}$
  $\mu_{\hat{n}_t,t}^{[m]} = \mu_{\hat{n}_t,t-1}^{[m]} + K_{\hat{n}_t,t}(z_t - \hat{z}_{\hat{n}_t,t})$
  $\Sigma_{\hat{n}_t,t}^{[m]} = (I - K_{\hat{n}_t,t} G_{m_{n_t}}) \Sigma_{\hat{n}_t,t}^{[m]}$
end if
for $n = 1$ to $N_t^{[m]}$ **do**
  if $n \neq \hat{n}_t$ **then**
    $\mu_{\hat{n}_t,t}^{[m]} = \mu_{\hat{n}_t,t-1}^{[m]}$
    $\Sigma_{\hat{n}_t,t}^{[m]} = \Sigma_{\hat{n}_t,t-1}^{[m]}$
  end if
end for
$\omega_t^{[m]} = p_{\hat{n}_t,t}^{[m]}$
add new particle to X_{aux}
end for

for $m = 1$ to M **do**
  draw random particle from X_{aux} with prob. $\sim \omega_t^{[m]}$
  add new particle to X_t
end for

FastSLAM



https://www.youtube.com/watch?v=ATj-DrwrHx0

FastSLAM

Graph-based SLAM

- Offline, Full-SLAM, feature-based Algorithmus

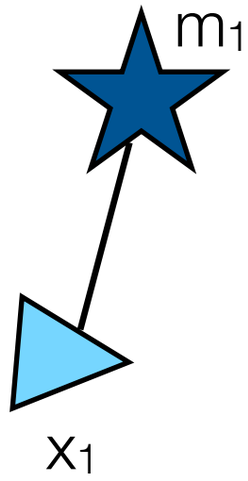
- Idee

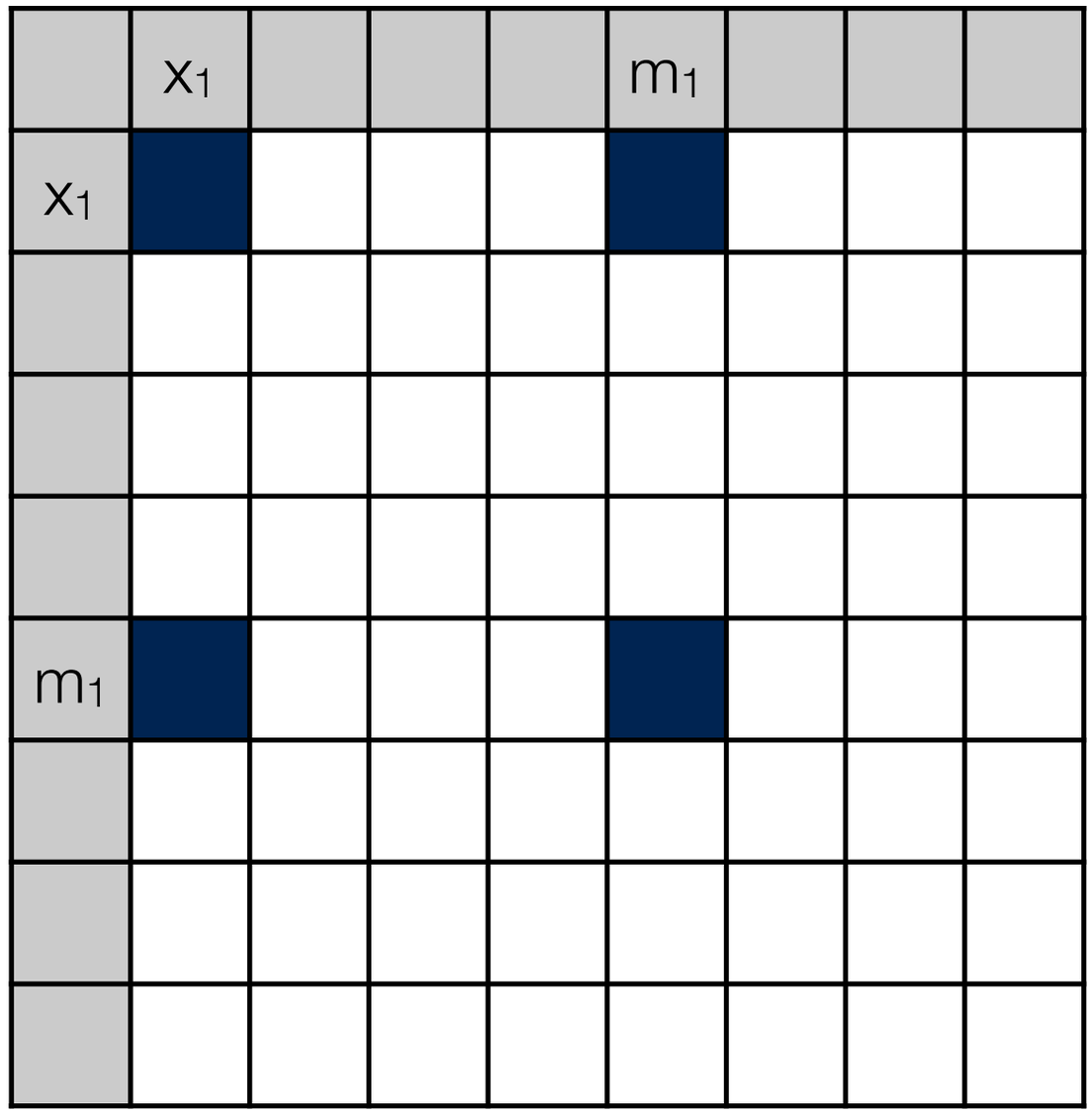
    - Position und Landmarken sind Knoten in einem Graph

    - Jeder Übergang von x_{t-1} nach x_t (Roboterposition) wird durch Kanten in dem Graphen repräsentiert

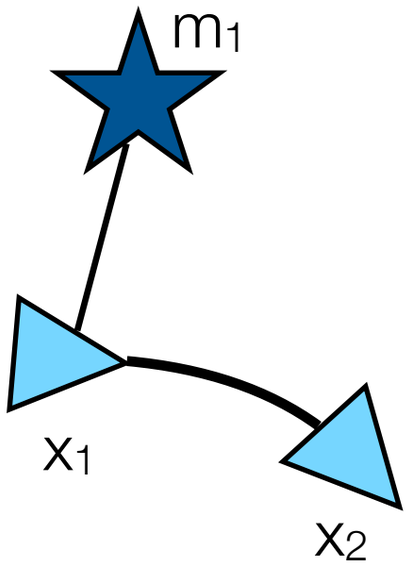
    - Jede Messung eines Objekts m_i wird ebenfalls mit dem Zustand x_t durch eine Kante verknüpft, wenn m_i zum Zeitpunkt t beobachtet wurde

Graph-based SLAM



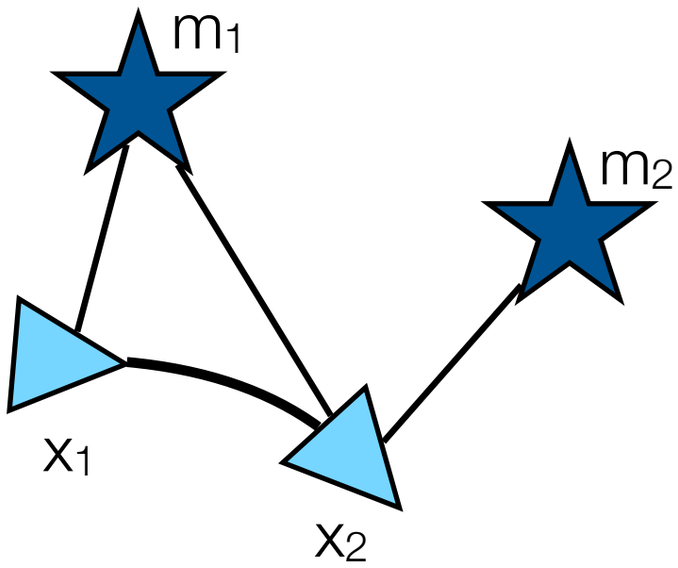


Graph-based SLAM



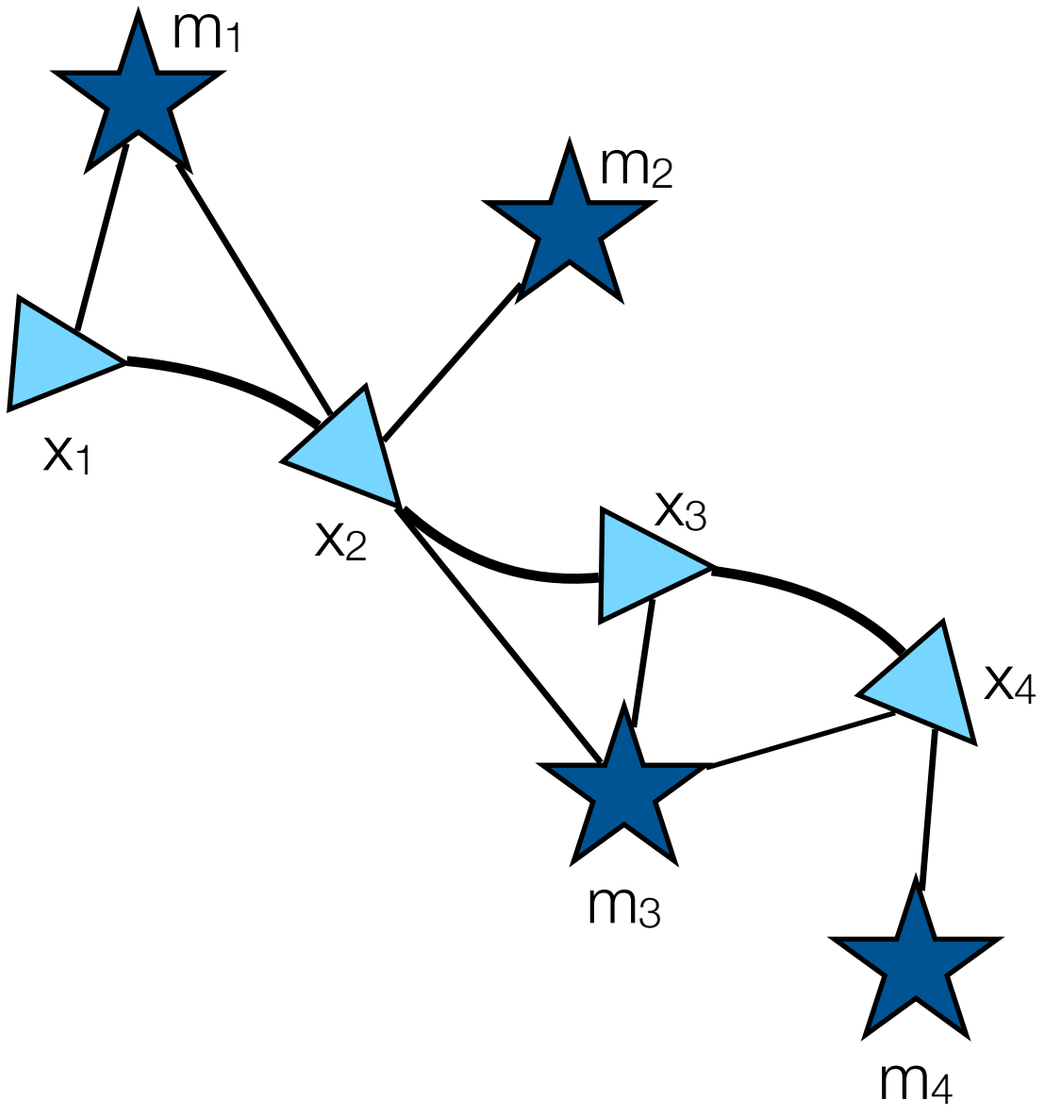
	x_1	x_2	x_3	x_4	m_1	m_2	m_3	m_4
x_1	■	■			■			
x_2	■	■						
x_3								
x_4								
m_1	■				■			
m_2								
m_3								
m_4								

Graph-based SLAM



	x_1	x_2			m_1	m_2		
x_1	■	■			■			
x_2	■	■			■	■		
m_1	■	■			■			
m_2		■				■		

Graph-based SLAM



	x_1	x_2	x_3	x_4	m_1	m_2	m_3	m_4
x_1	■	■			■			
x_2	■	■	■		■	■	■	
x_3		■	■	■			■	■
x_4			■	■				
m_1	■	■			■			
m_2		■				■		
m_3		■	■	■			■	
m_4				■				■

Graph-based SLAM

- Stellen uns den Graphen als ein Feder-Massen-System vor

- Lösung des SLAM Problems entspricht dem Gleichgewichtszustand (minimale Energie des Systems)

- Graph korrespondiert zu dem Logarithmus der Posterior-Verteilung:

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

- Ohne Herleitung: $\log p(x_{1:t}, m | z_{1:t}, u_{1:t}) = \text{const}$

[Kante im Graph: Bewegung des Roboters] $+ \sum_t \log_t p(x_t | x_{t-1}, u_t)$

[Kante im Graph: Sensormessung] $+ \sum_t \log_t p(z_t | x_t, m)$

- SLAM: Maximierung der Gleichung $x^*_{1:t}, m^* = \arg\max_{x_{1:t}, m} \log p(x_{1:t}, m | z_{1:t}, u_{1:t})$

Graph-based SLAM

$$\log p(x_{1:t}, m | z_{1:t}, u_{1:t}) = \text{const}$$
$$+ \sum_t \left[x_t - g(x_{t-1}, u_t)\right]^T R_t^{-1} \left[x_t - g(x_{t-1}, u_t)\right]$$
$$+ \sum_t \left[z_t - h(x_t, m)\right]^T Q_t^{-1} \left[z_t - h(x_t, m)\right]$$

- Lösungen kann durch Standardmethoden gefunden werden, z.b. Gradientenverfahren, Simulated Annealing

ATLAS — Hybrides SLAM Verfahren

- Graphenbasiertes, metrischen Verfahren
- Funktioniert mit Scan-matching und Feature-based SLAM-Verfahren
- Explicit Loop-closing via effizientes Map-Matching Verfahren

- Grundlegende Idee:
  - Knoten sind lokale Koordinatensysteme
  - Kanten beschreiben die Translationen zwischen den Knoten und die Unsicherheiten eines Knoten in Bezug des anderen

Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework,
Bosse, Newman, Leonard, Teller, 2004

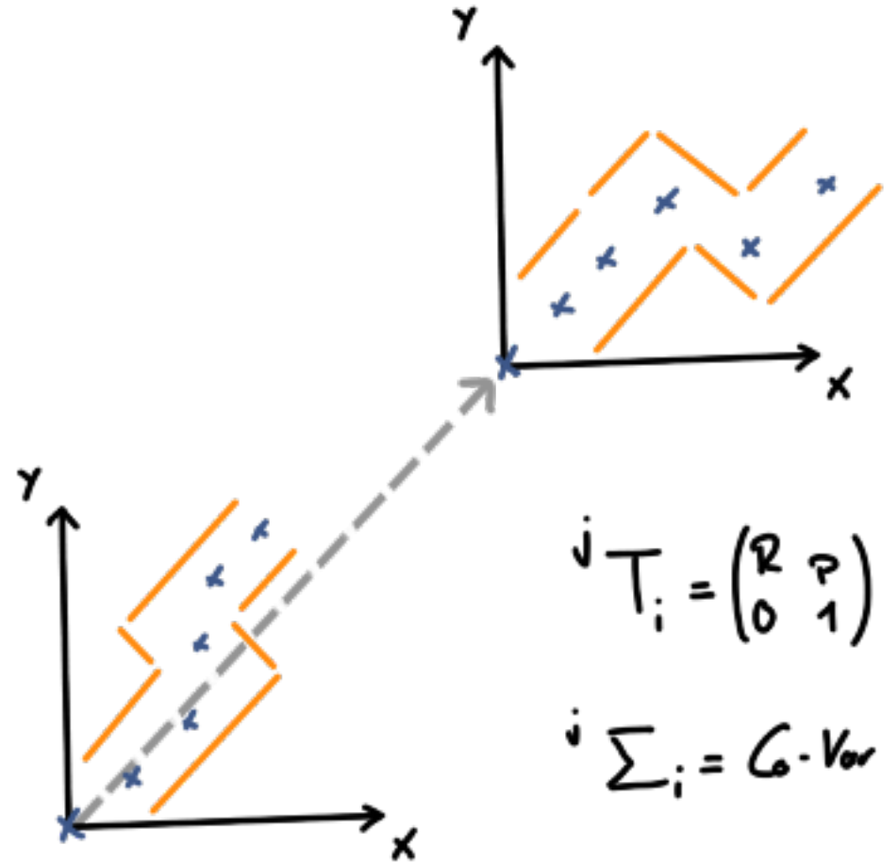
ATLAS — Algorithmus

- In jeder Iteration des Algorithmus müssen 4 Schritte ausgeführt werden

1. Local Map Iteration
   - Lokales SLAM-Verfahren um Daten zu sammeln
2. Hypothesis State Transition
   - Generieren und testen von Hypothesen über den aktuellen Zustand
3. Map Projection Iteration
   - Aktualisieren der lokalen Map-unsicherheiten und -positionen
4. Map Matching
   - Loop-closing / Reduction

ATLAS — Komponenten

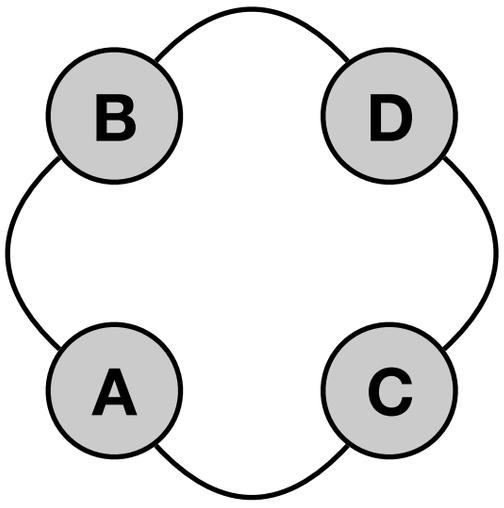
1. Uncertainty Projection
2. Competing Hypotheses
3. Generation of new maps
4. Loop closing
5. Instantiation and evaluation of new maps

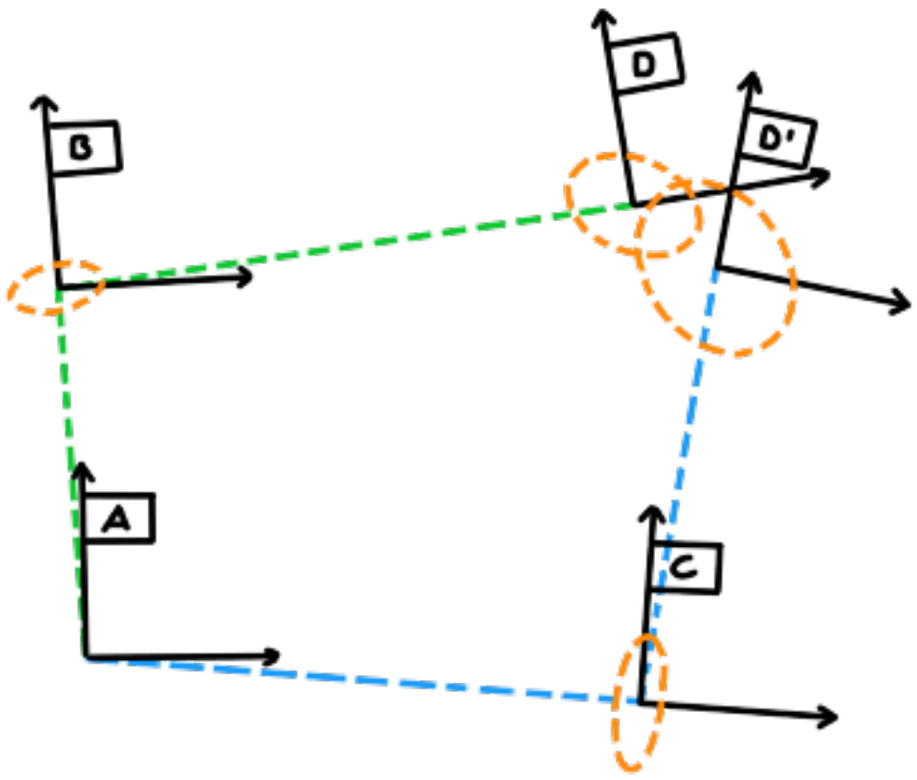


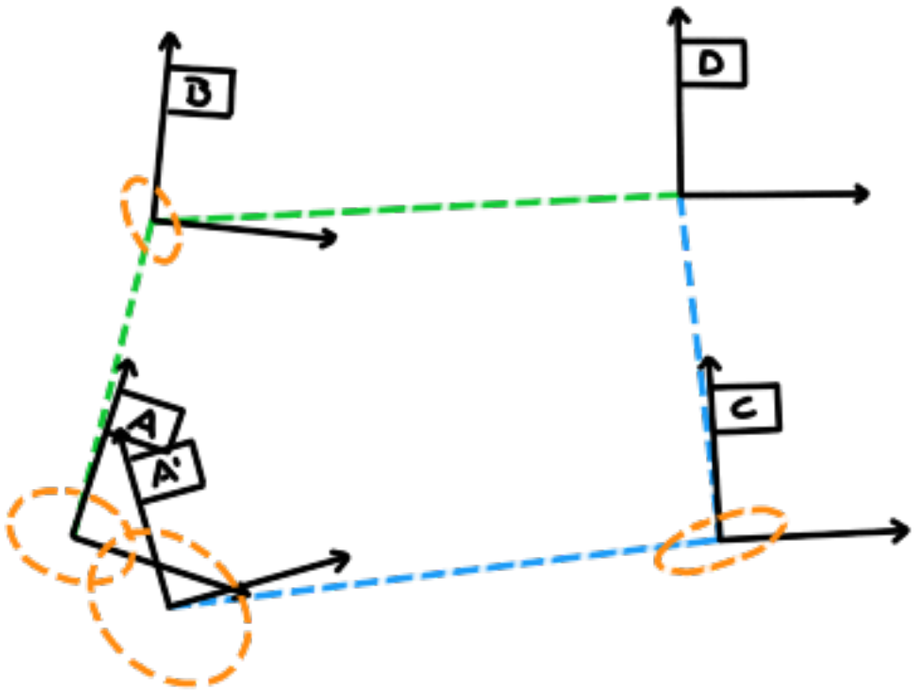
ATLAS — Uncertainty Projection

- In einem zyklischen Graph existieren mehrere Pfade zwischen Knoten

- Für die Berechnung der Unsicherheiten wird der “kürzeste” Pfad genommen, d.h. der Pfad mit der geringsten projizierten Unsicherheiten

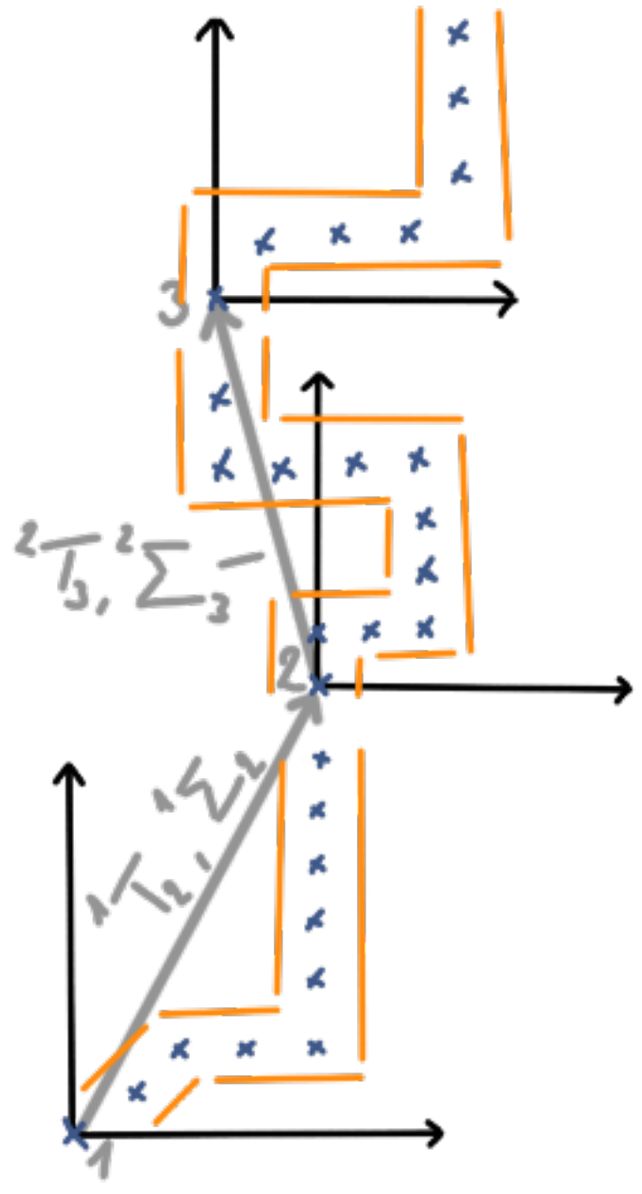






ATLAS — Genesis

- Jede lokale Map hat eine maximale Komplexität
- In den Anwendungen sind das maximal 15 Positionen oder Features
- Der Graph muß daher durch neue Maps erweitert werden



1. Aktuelle Position des Roboters definiert den Ursprung des neuen Koordinatensystems
2. Posture wird auf Null gesetzt
3. Unsicherheit wird auf Null gesetzt
4. Unsicherheit des neuen Koordinatensystems wird relativ zur Source angeben

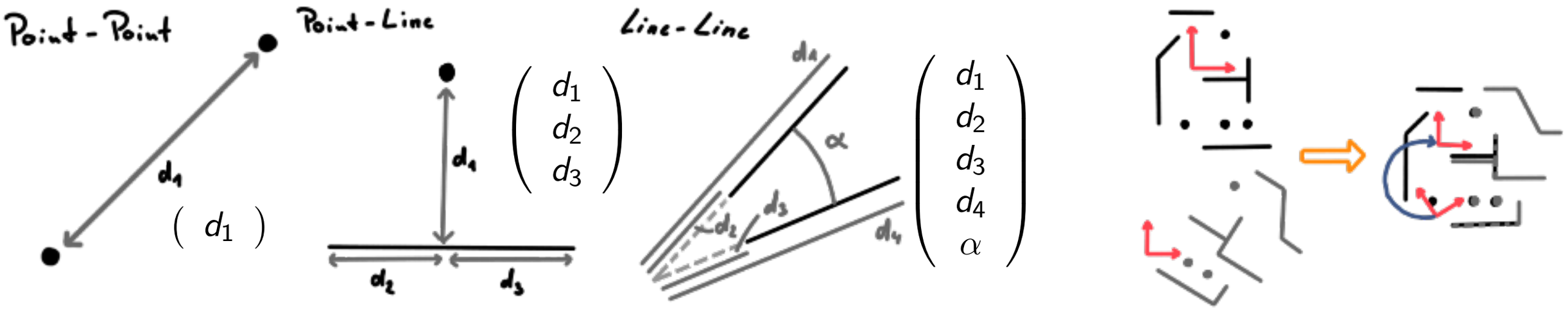
ATLAS — Map Matching / Loop-closing

- In zyklischen Umgebungen, werden Bereiche mehrfach besucht
- *Map-matching* bedeutet, dass man diese Bereiche automatisch identifiziert
- 3 Schritte:

1. Durch die *Uncertainty Projection* erhalten wir eine Liste von möglichen Matches
2. Für jede der möglichen Matches, wird die “Ähnlichkeit” bestimmt
3. Update des Graphen

ATLAS — Feature-based Map Matching

- Erstellen einer Signatur, die invariant gegenüber Transformationen ist



- <u>Self-Matching</u>: Entfernen von repetitive Features (z.B. Türen in einem Flur)

- <u>Vergleich & Zuordnung</u>: Jeder Match beschreibt eine mögliche Transformation zwischen den beiden Maps

- Jede Transformation wird evaluiert, nach der Anzahl der Features, die in Einklang gebracht werden

- Die besten *n* Transformationen ergeben die Gesamttransformation & Co-variance Matrix

$$\Sigma_{ij} = \mathbb{E}\left[(X_i - \mu_i)(X_j - \mu_j)\right]$$

ATLAS — Cycle Verification

- Zwei mögliche Probleme beim Schliessen von Zyklen

    1. *False Positives* — führt zu falschen Loops

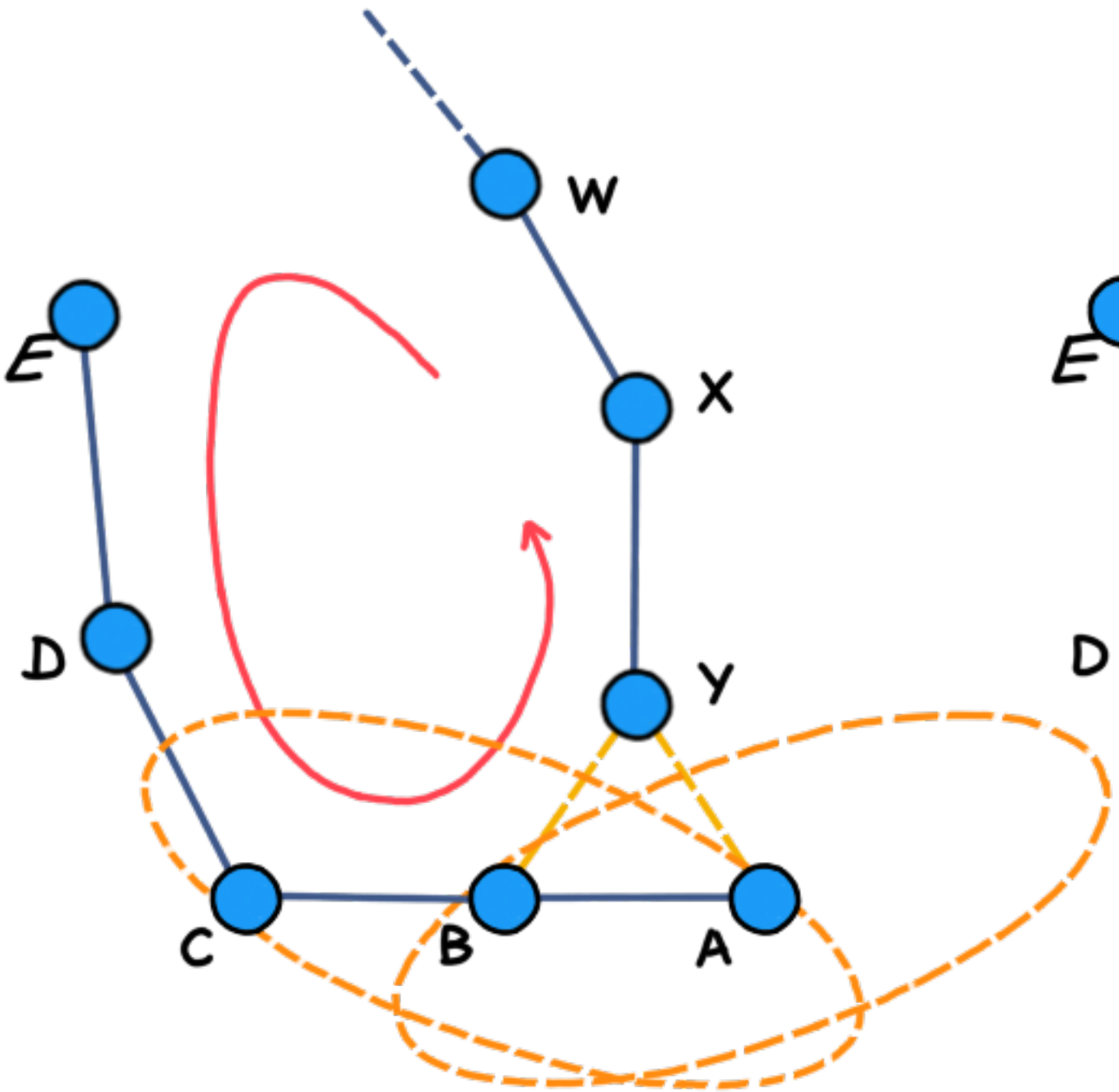
    2. *False Negatives* — führt zu Loops, die nicht geschlossen werden

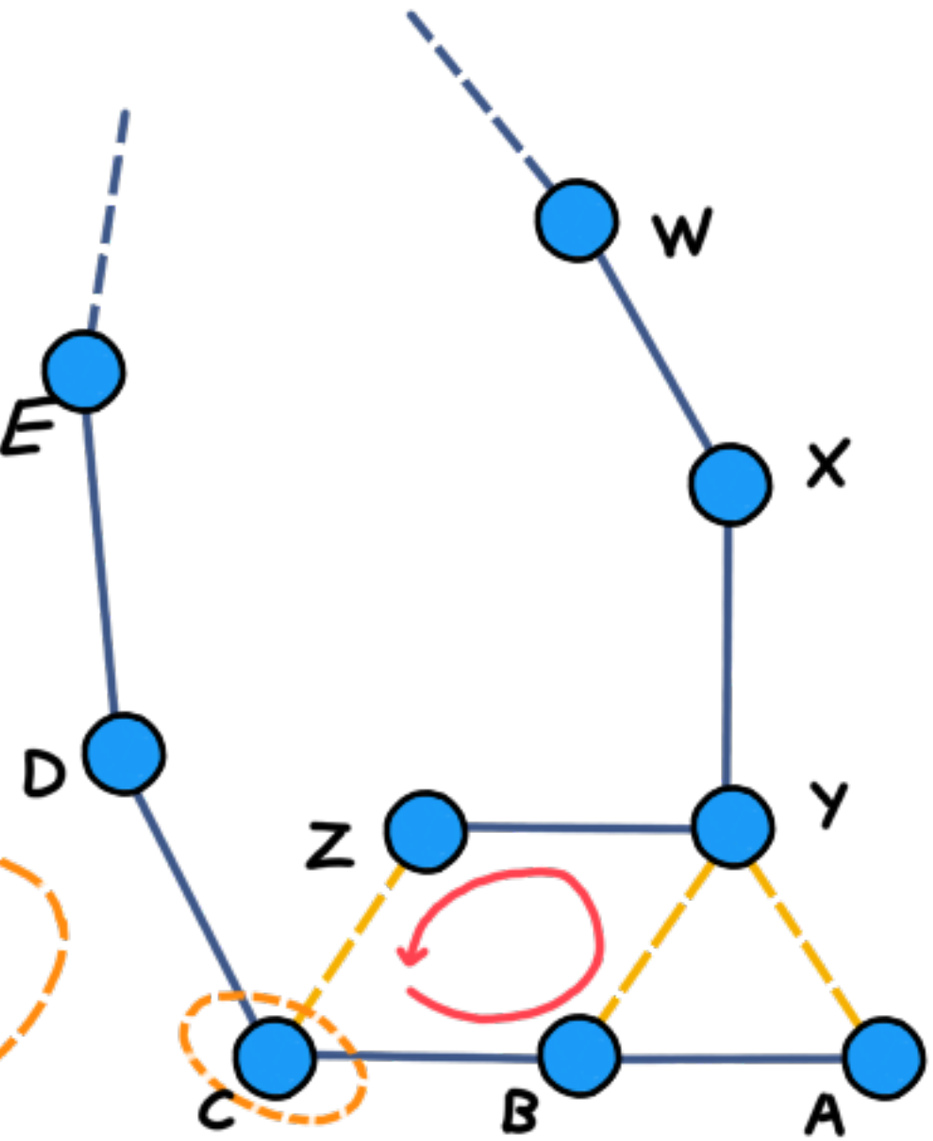
- Ansatz:

    - Entscheidung eine Loop zu schliessen wird erst dann getroffen, wenn genug Informationen vorhanden sind

    - Schliessen von kleinen Loops wird bevorzugt

ATLAS — Cycle Verification





ATLAS — *Traversal with Competing Hypotheses*

- Wenn der Roboter eine Map verlässt, muss entweder einer anderen, angrenzenden Map zugeordnet werden oder es muss eine neue Map erzeugt werden

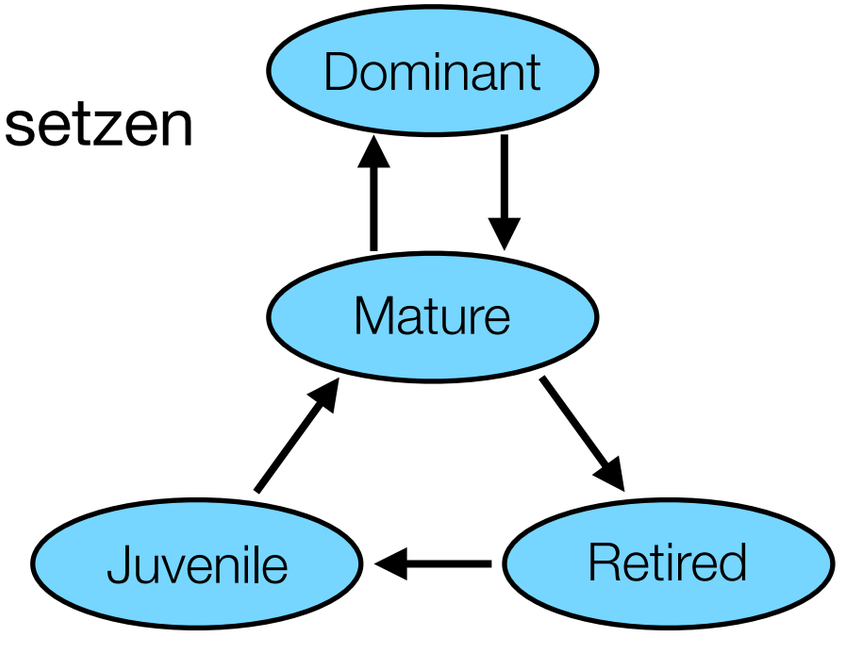
- Die Entscheidung geschieht aufgrund von Hypothesen, die mit den Sensordaten verglichen werden

- Es wird zu jedem Zeitpunkt eine Menge von Hypothesen verwaltet, um die Position in der Map zu bestimmen

- Es gibt vier Arten von Hypothese, *Dominant, Mature, Juvenile, Retired*

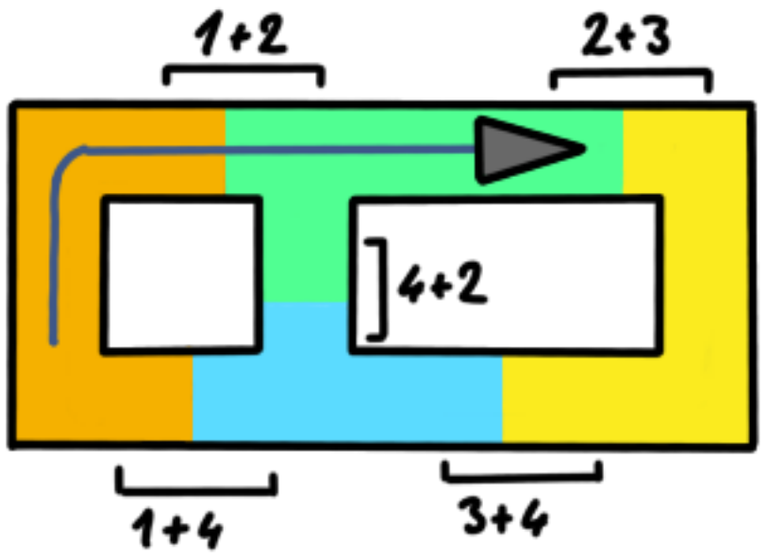
ATLAS — Hypotheses states

- Mature Map:
  - Kann den Status angrenzender Maps auf Juvenile setzen
  - Kann neue Maps erzeugen
- Dominant Map:
  - Mature Map mit der besten Metric (gegeben der Sensordaten)
  - Ausgabe des ATLAS Frameworks
- Juvenile Map:
  - Testet wie gut eine angrenzende Map die aktuellen Messungen erklären kann
  - Wird nach einer festen Zeit (z.B. 2 Sekunden) gelöscht
- Retired Map:
  - Mature Maps, die die aktuellen Daten nicht erklären können
  - Können als Juvenile Map reaktiviert werden
  - Markieren inaktive Bereiche der Karte
- Wenn es nur eine Mature Map gibt, die aber die Daten nicht erklären kann, wird eine neue Map erzeugt

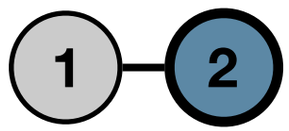


ATLAS — Traversal

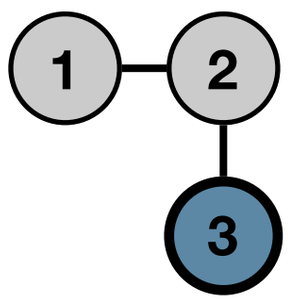
Generation



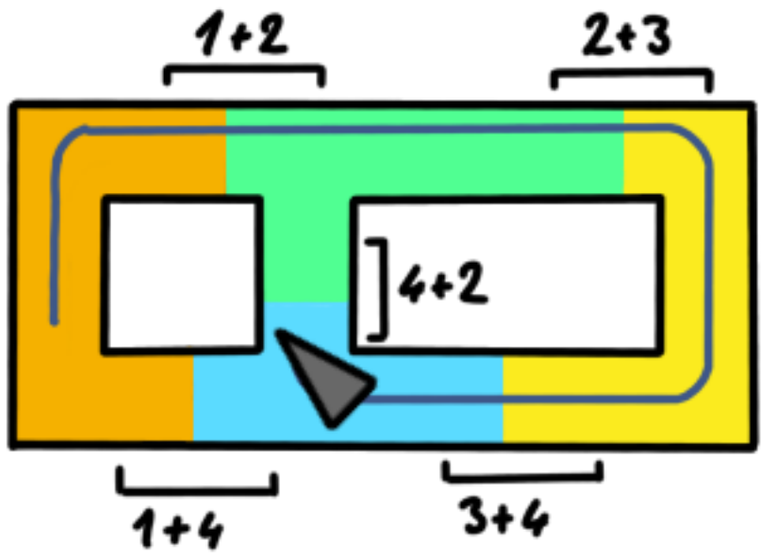
Prior



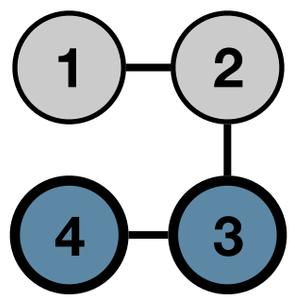
Posterior



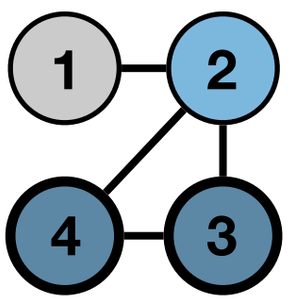
Map-Matching



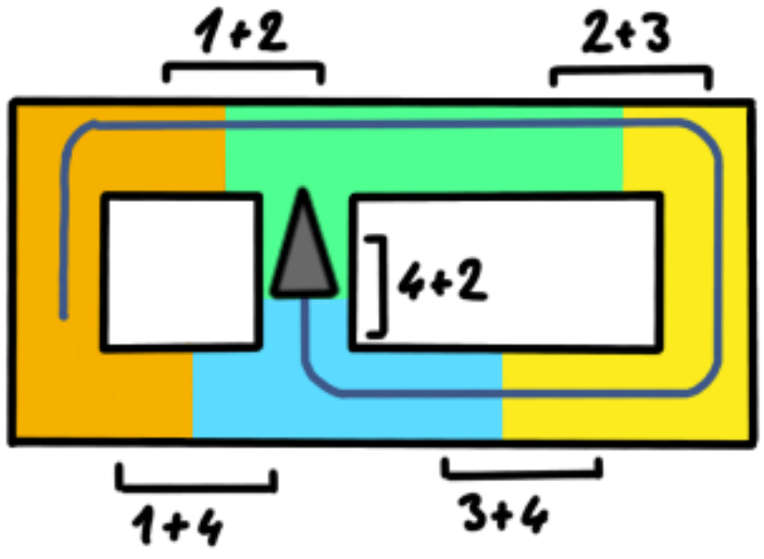
Prior



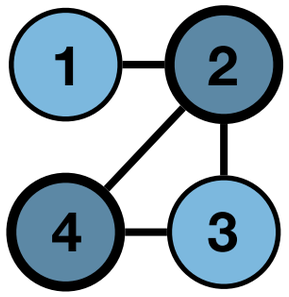
Posterior



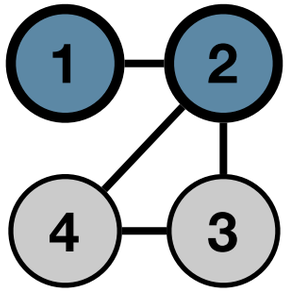
Hypothesenauswahl



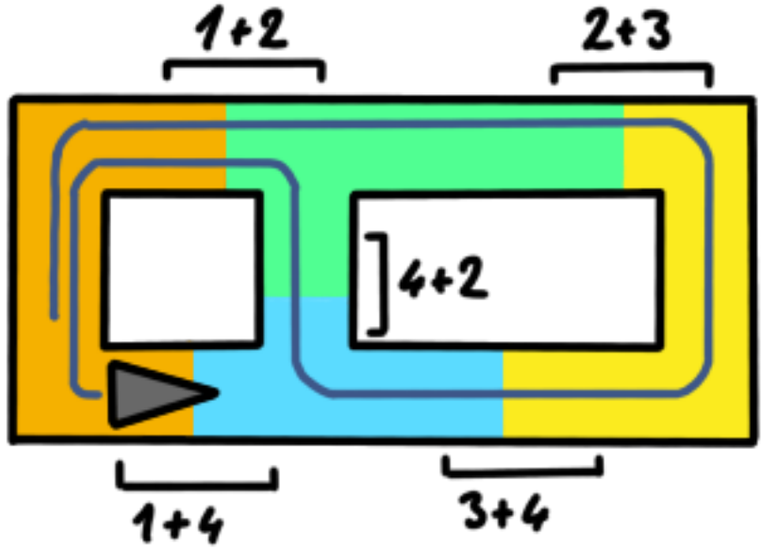
Prior



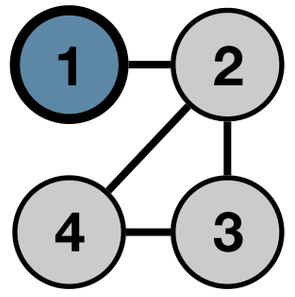
Posterior



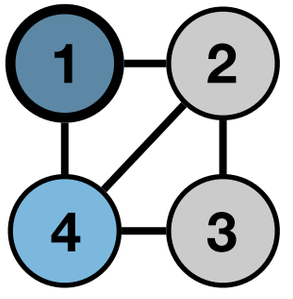
Loop-closing



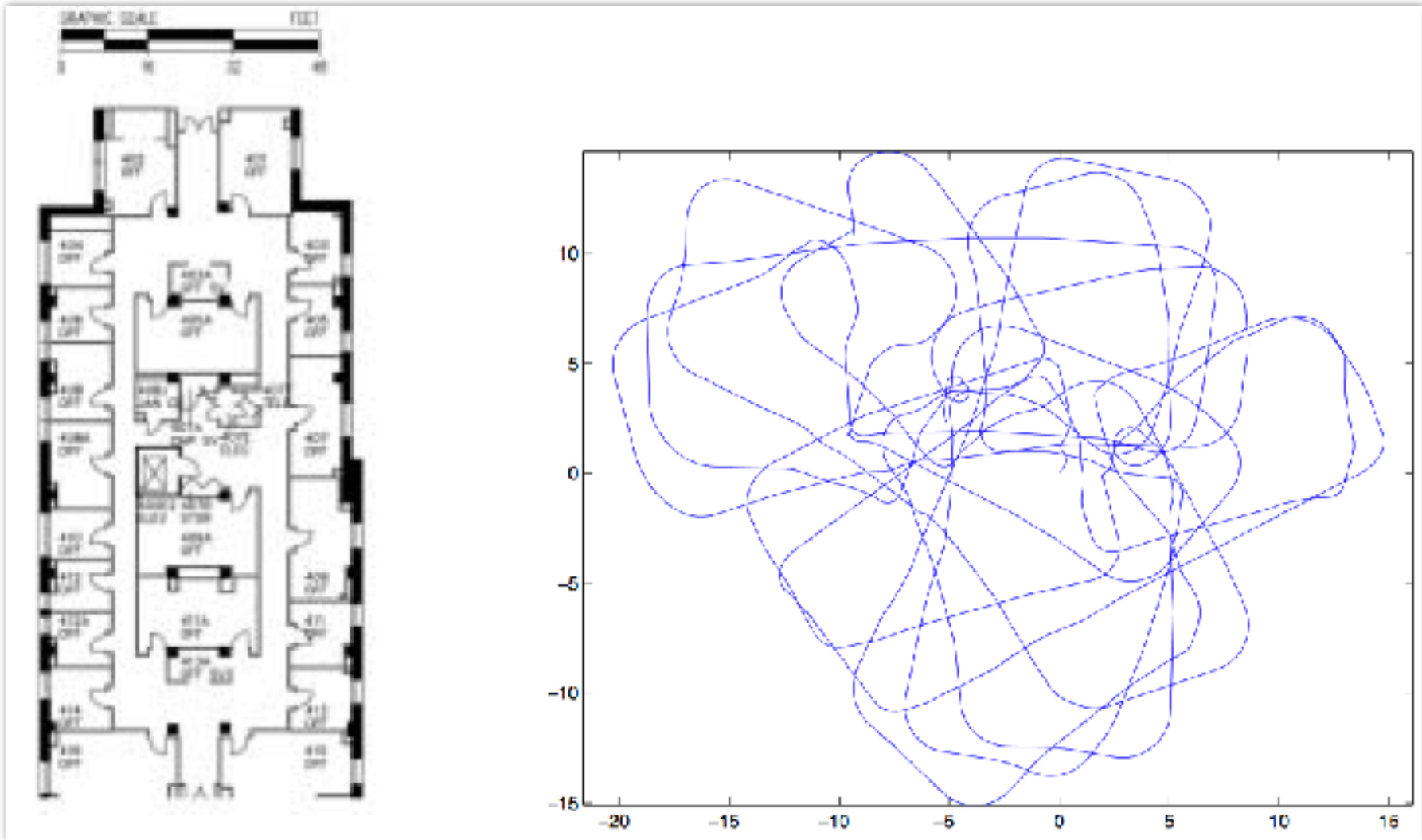
Prior

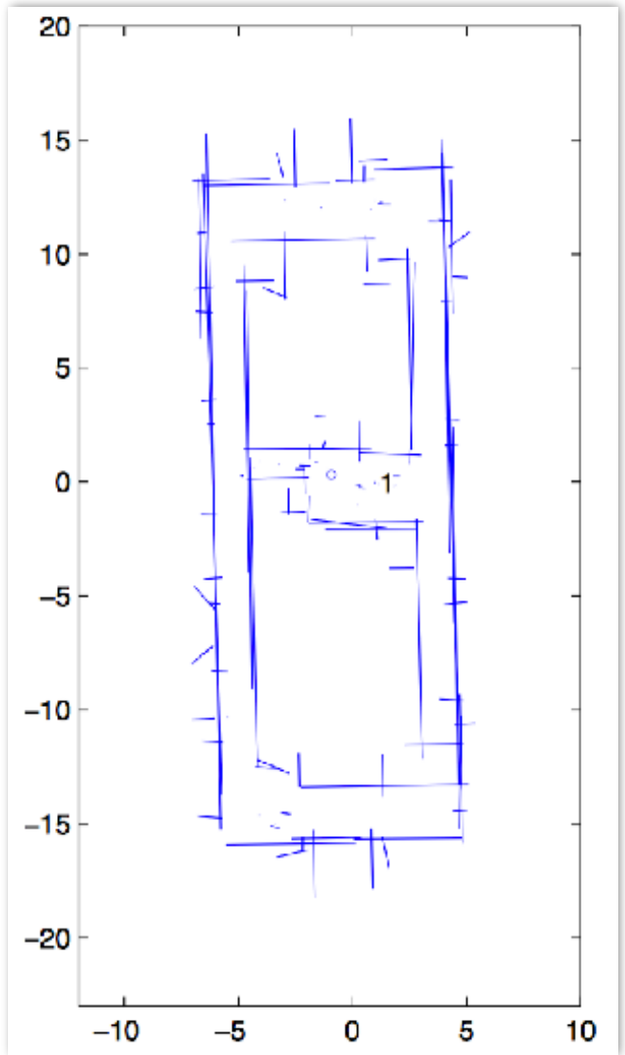


Posterior



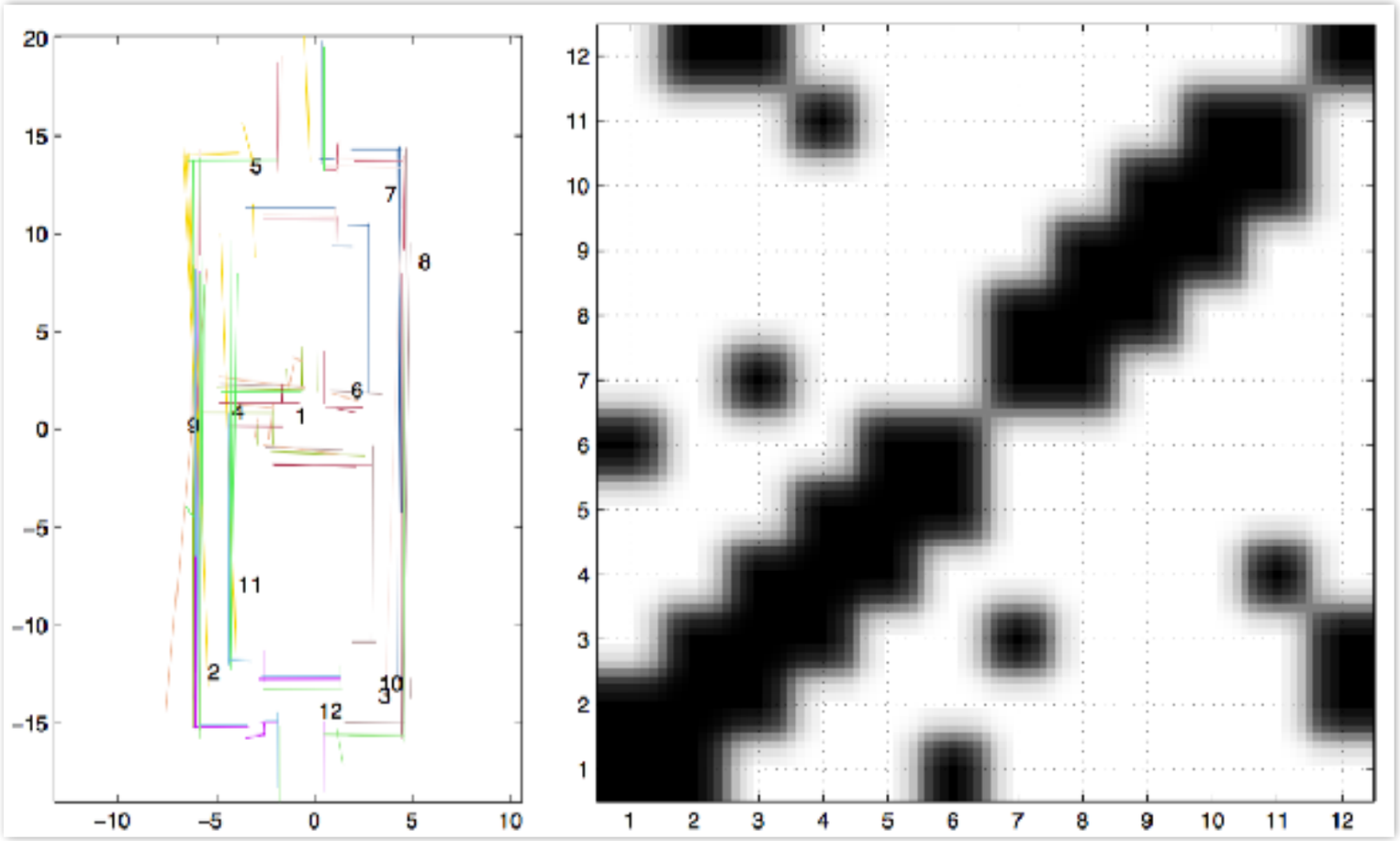
ATLAS — Erstellen einer globalen Karte

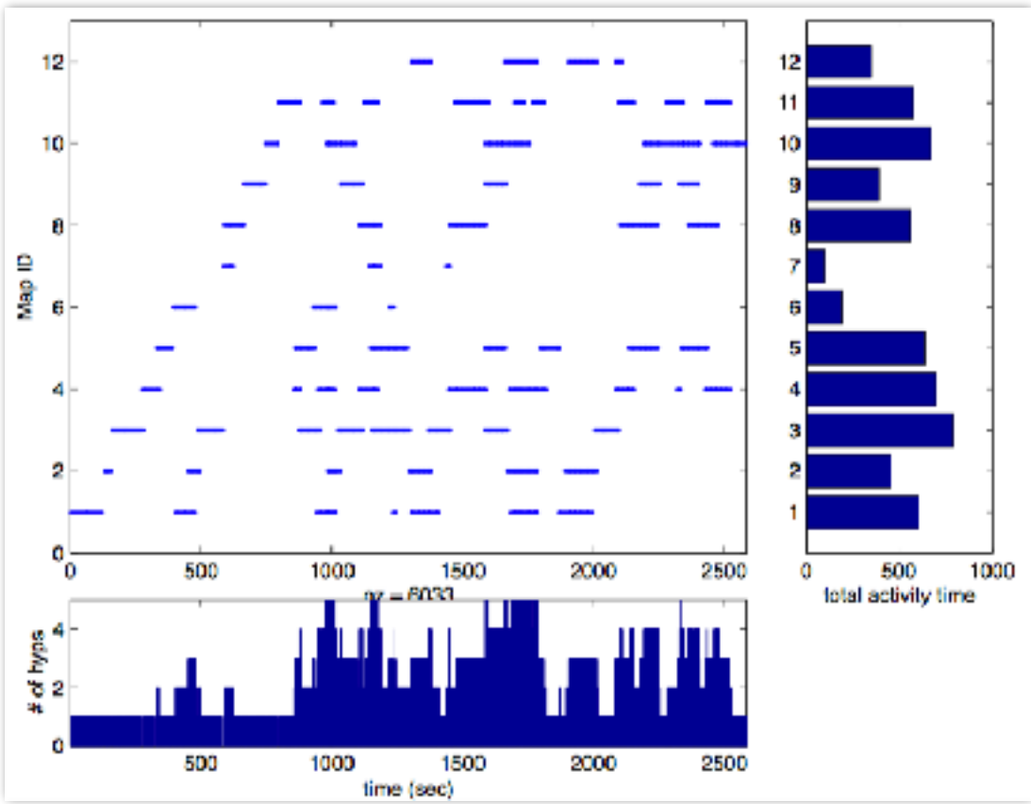




Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework,
Bosse, Newman, Leonard, Teller, 2004

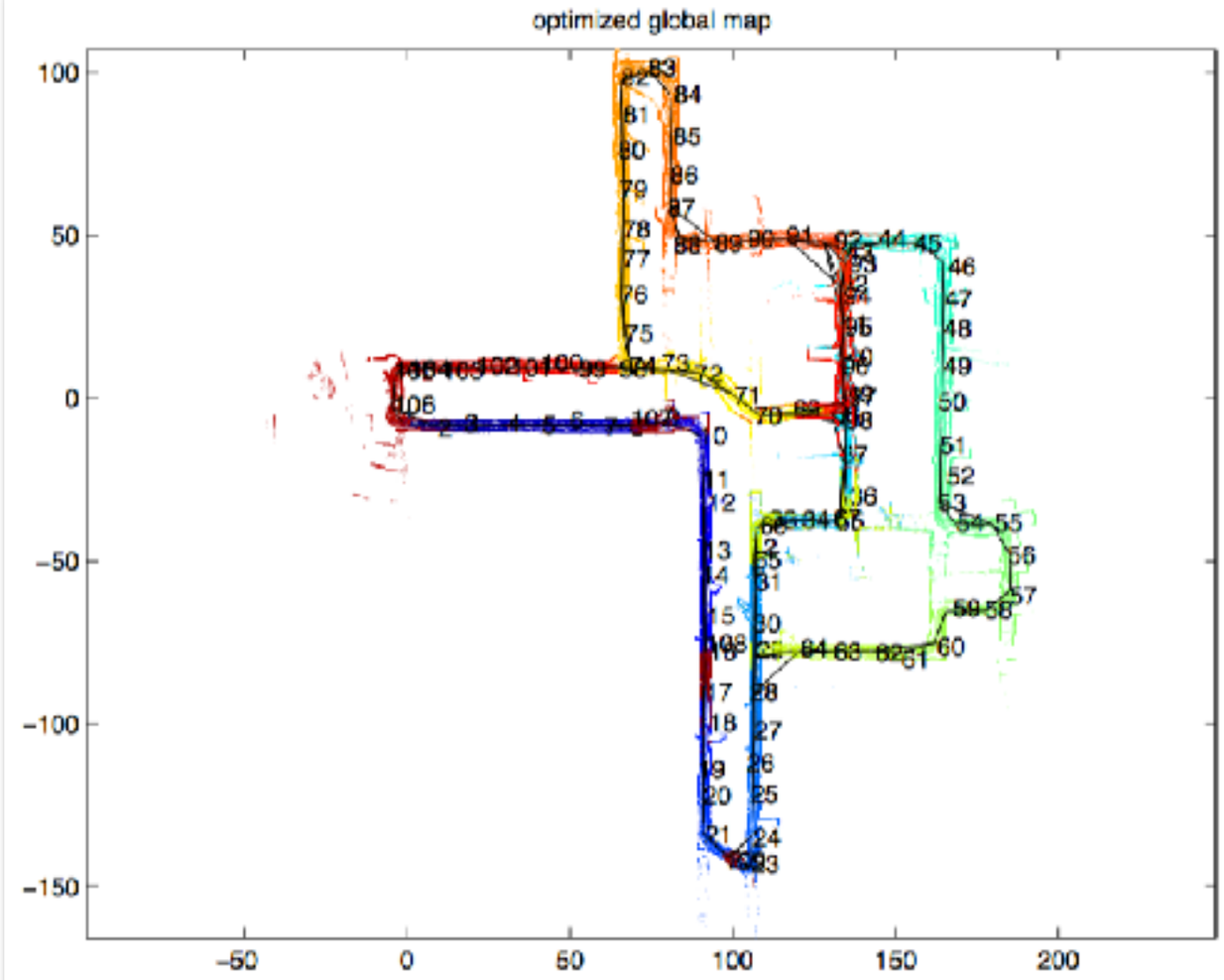
ATLAS — Erstellen einer globalen Karte

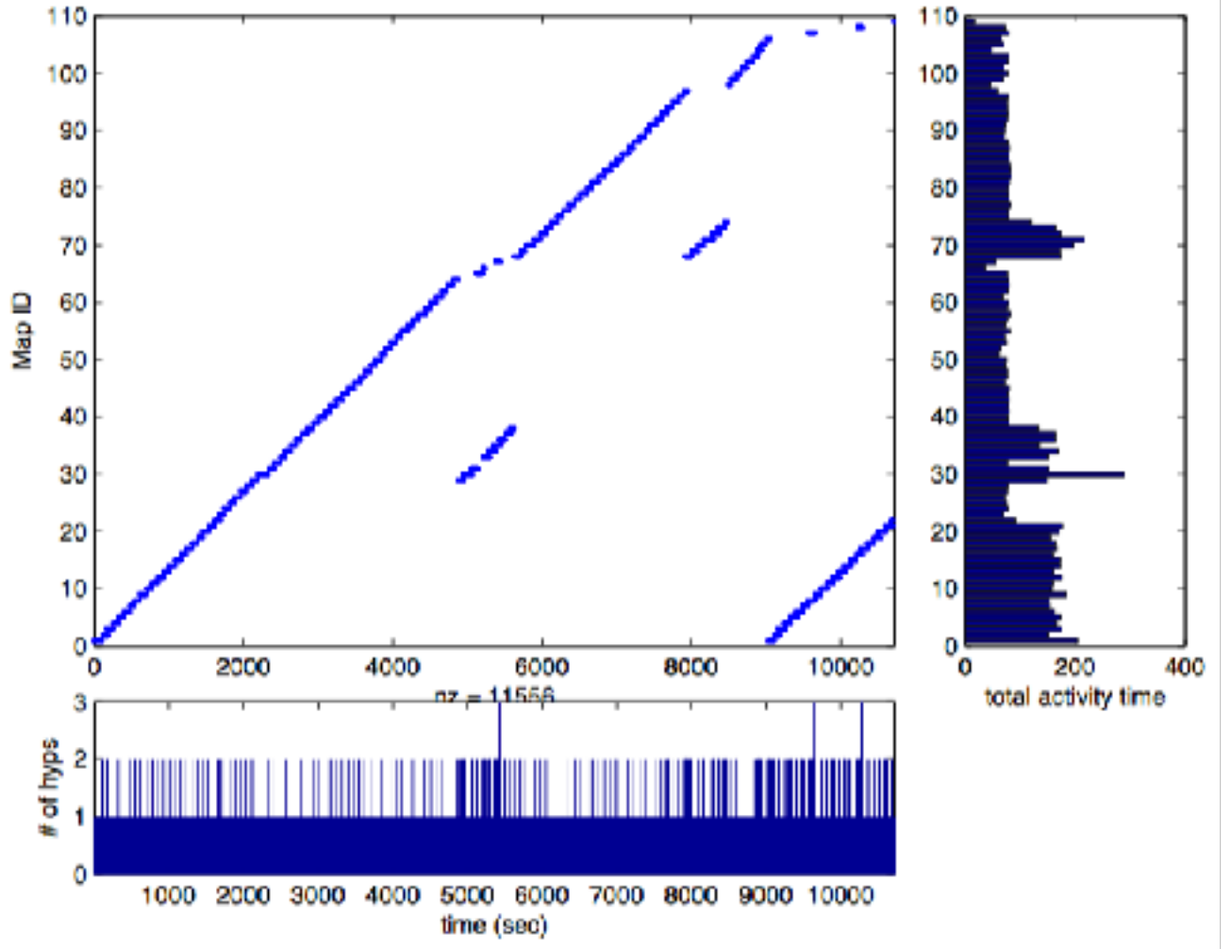




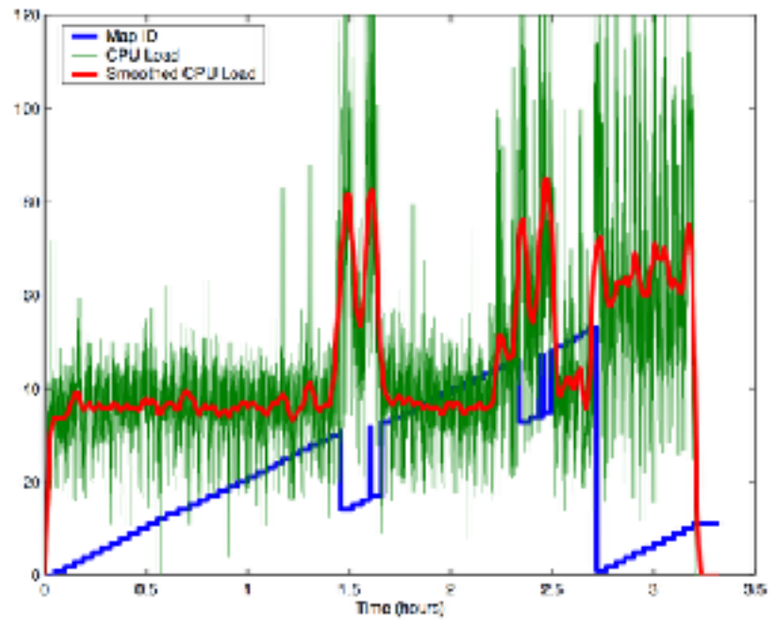
Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework, Bosse, Newman, Leonard, Teller, 2004

ATLAS — Erstellen einer globalen Karte





Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework, Bosse, Newman, Leonard, Teller, 2004



Large-Scale 2D Laser-Based Simultaneous Localization and Mapping

Autonomous Systems Laboratory
CSIRO ICT Centre

SLAM — Offene Probleme

- Dynamische Umgebungen
- Multi-robot SLAM
- Standardimplementierungen auf den Niveau industrieller Anwendungen. Beispiel OpenCV
- Long-term SLAM (Tage, Wochen, Monate, Jahre)