Lokalisierung in Autonomen Mobilen Systemen

Bruno Hauser
Fachseminar in Verteilte Systeme, ETH Zürich
Betreuerin: Silvia Santini

Ablauf des Vortrags

- **Was sind “Autonome Mobile Systeme”?**
- **Motivation**
- **Ortsbestimmung – Warum / Wie ?**
- **Übersicht über vorgestellte Methoden**
- **Learned Landmarks**
- **Collective Localization**
- **Virtuelle Pheromone**
- **Zusammenfassung**

Was sind “Autonome Mobile Systeme”?

- **Oft auch als “Multi-Roboter Systeme” bezeichnet**

- **Ansammlung von Geräten, die mit einer ..**

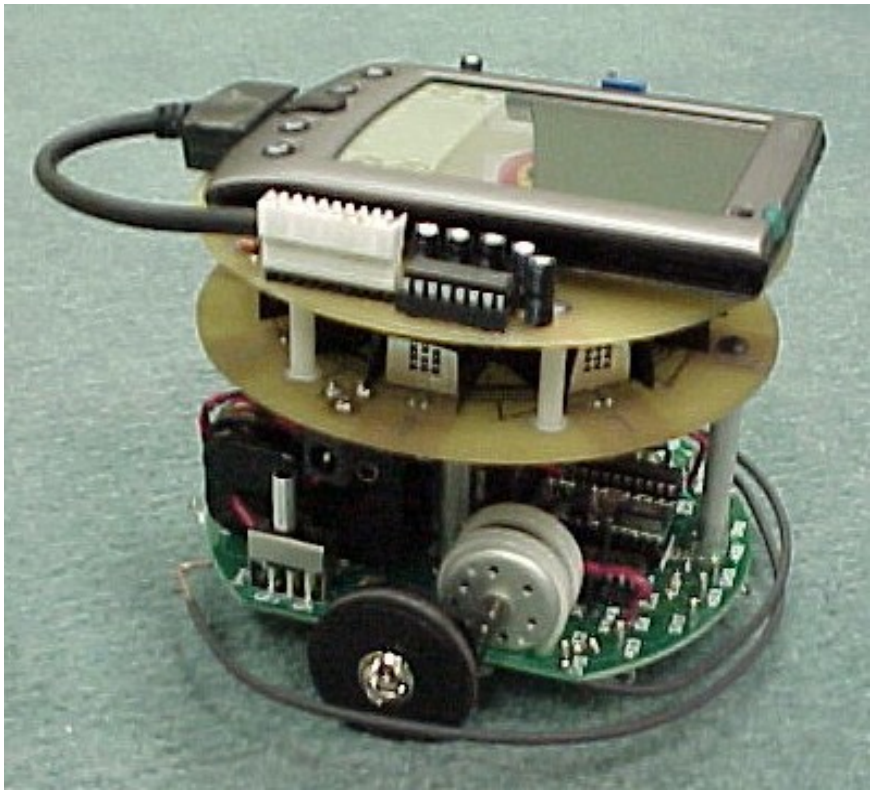
  - Recheneinheit

  - Kommunikationseinheit

  - Sensoreinheit

  - Fortbewegungseinheit

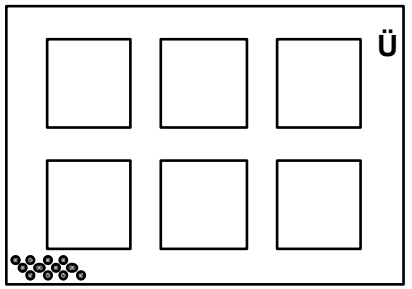
  .. ausgestattet sind



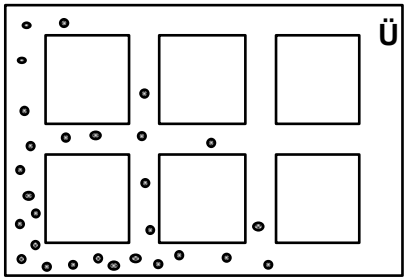
Autonome Mobile Systeme: Warum? - Beispielszenario

- **Rettungsteam will unbekanntes Gebäude nach Überlebenden durchsuchen**

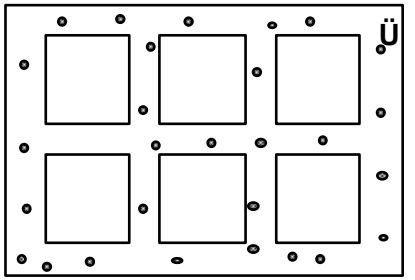
  -> benutzen dafür Verbund von Robotern



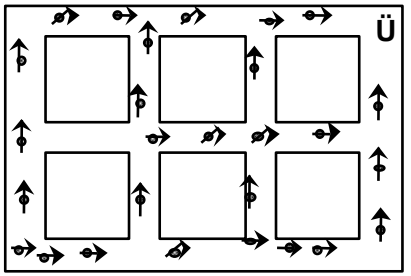
Absetzen der Roboter



Roboter verteilen sich



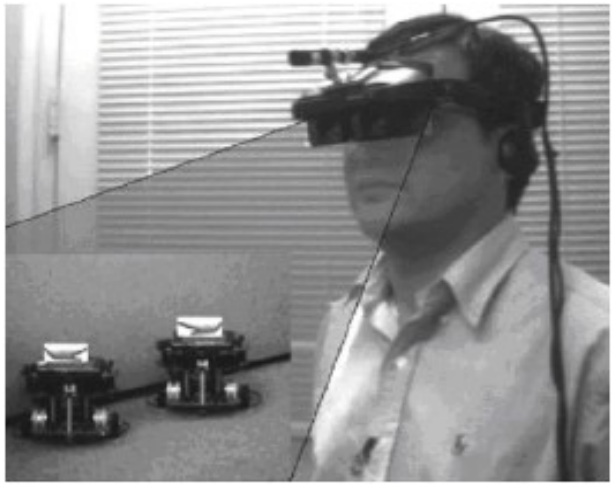
Detektieren Opfer

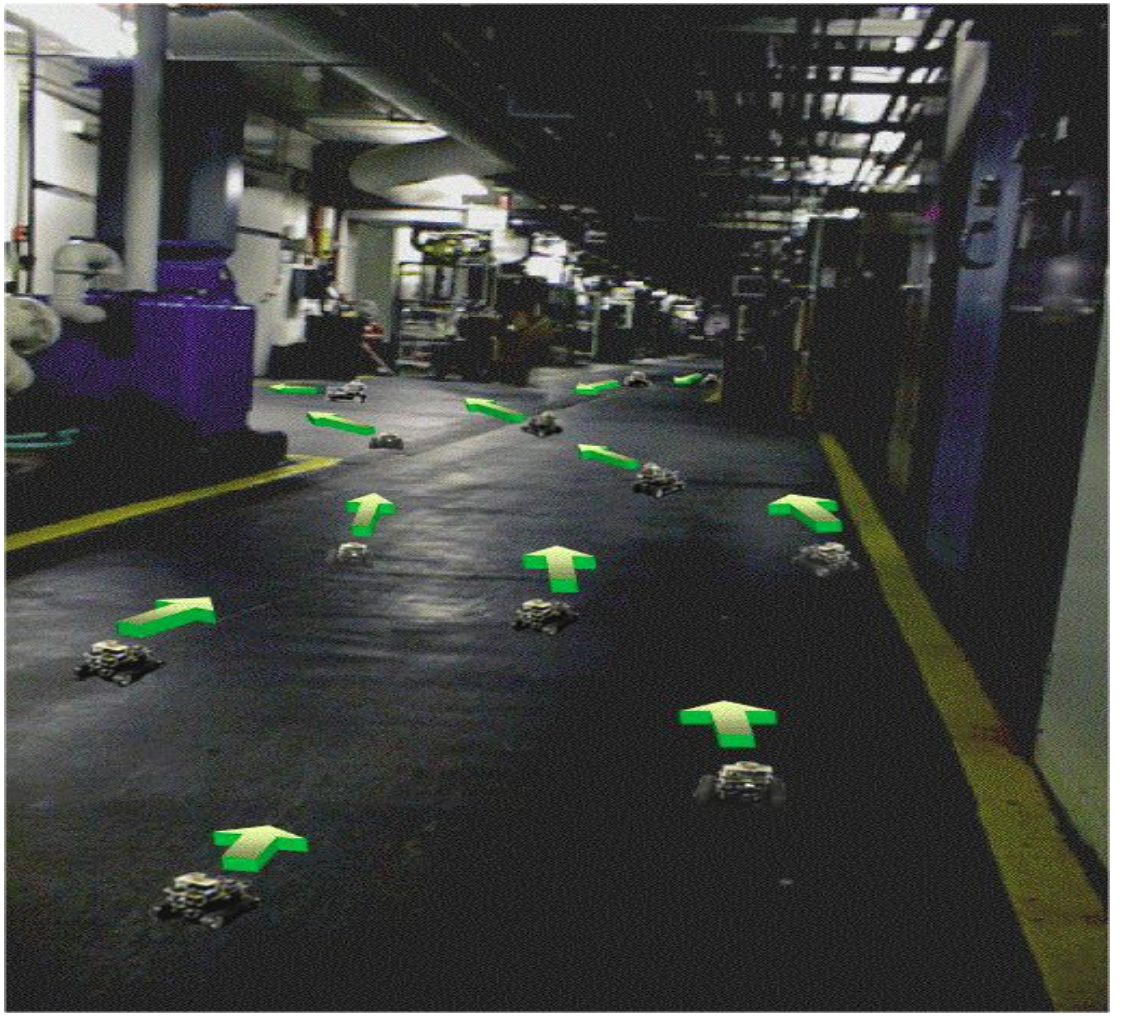


Propagierung der Information

Wo ist der Überlebende?

- **Richtungsanzeige der Roboter gibt dem Rettungsteam den Weg an**





Ortsbestimmung - Warum?

Damit mobile Roboter autonom agieren können:

Präzise Ortsbestimmung nötig!

Ortsbestimmung - Wie?

- **Odometrisch**

  - **Fehleranfällig (Durchdrehen der Räder, Kollisionen etc.)**
  - **Fehler vergrössert sich stetig wenn keine Rekalibrierung**

- **Umgebungssensoren (Kombination mit odometrischen Messungen)**

  - **Ranging Sensors (Sonar, Laser)**
  - **Vision-Based (Landmarks)**
  - **GPS, andere**

- **Zusätzlich: Schätzfunktion, reduziert Ungenauigkeiten der Sensormessungen**

Alleine oder im Team?

- **Forschung: von “Single” zu “Multi”, Algorithmen übernommen**

- **Allein:**

  Jeder Roboter im Verbund löst Lokalisierungsproblem komplett für sich (nur eigene Sensorinformationen)

  -> Kein Informationsaustausch

- **Im Team:**

  Jeder misst unabhängig, tauscht danach Messinformationen aus

  Bei ständiger Kommunikation untereinander:

  -> Jeder hätte max. gleiche Positionsungewissheit wie Roboter mit den besten Lokalisierungsresultaten (ohne Informationsaustausch)

Übersicht über vorgestellte Methoden

- **Learned Landmarks**

  - **Single-Robot Methode, Anwendung/Kombination in Multi-Robot System möglich**
  - **Optischer Ansatz**
  - **Kanten-Maximas von Kamera-Bildern als Angelpunkte**

- **Collective Localization (Kalman Filter)**

  - **Multi-Robot Lokalisierungsmethode**
  - **Einsatz verschiedener Sensoren möglich**
  - **Austauschen von Sensor-Informationen verbessert Genauigkeit beider Positionsabschätzungen**

Lokalisierung mit Landmarks

[6]

- **Abbild von Umgebungsansicht**
- **Landmarks: Visuelle Charakteristiken, die spezielle Eigenschaften des Bilds beschreiben**

Probleme bei der Lokalisierung mit Landmarks

- **Künstliche Landmarks**
- **Abhängig von Annahmen über Umgebung**
- **Autonomie ?**

Learned Landmarks (I)

- **Sensor: Digitale Kamera**

  -> billig, grosse Datenmenge mit wenig Rechenaufwand

  Im Gegensatz zu

  - **Sonar: störbehaftet**
  - **Laser: teuer, schwierig zu kalibrieren**

- **Landmarks werden vom Roboter selber extrahiert**

  - **Keine künstlich eingefügten Landmarks nötig**
  - **Keine speziell strukturierte Umgebung nötig**

Learned Landmarks (II)

- **Lokalisierung beruht auf zweistufigem Prozess:**

  a **Offline-Phase: (1x)**

  - **Repräsentation der Umgebung wird in Datenbank abgelegt**
  - **Extrahieren der Landmarks**

  b **Online-Phase: (nach Bedarf)**

  - **Vergleich der neuen Landmarks mit denjenigen in DB**
  - **Berechnung von Positionsabschätzungen aus Vergleichen**

Offline Phase

1. Bilder **2. Lokale Maximas** **3. Matching oder Neu?**

1. **Bilder von verschiedenen Blickwinkeln der Umgebung aufnehmen**

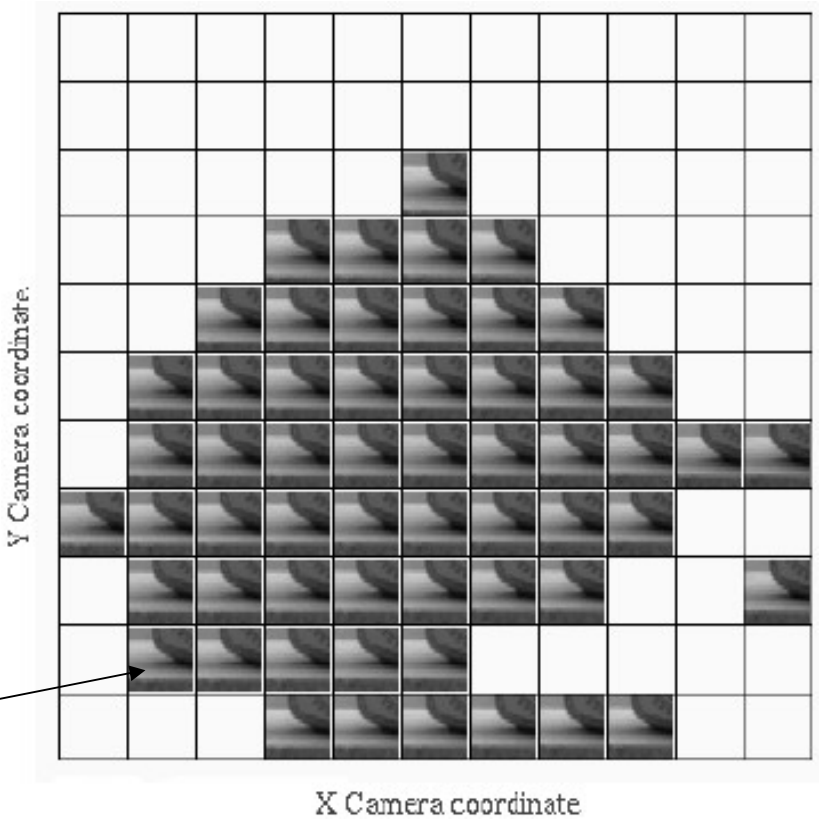
2. **Bestimmen der Candidate Landmarks anhand lokalen Kantendichte-Maximas**

3. **Tracking Algorithmus**
   -> speichere Ergebnisse in Datenbank

A Tracked Landmark

Kamera-Raum

- **Bildausschnitt im x-y Gitter entspricht dem Candidate Landmark, der aufgrund eines lokalen Kanten-Maximas im Roboter-Foto an der Stelle (x,y) im Kamera-Raum entdeckt wurde**



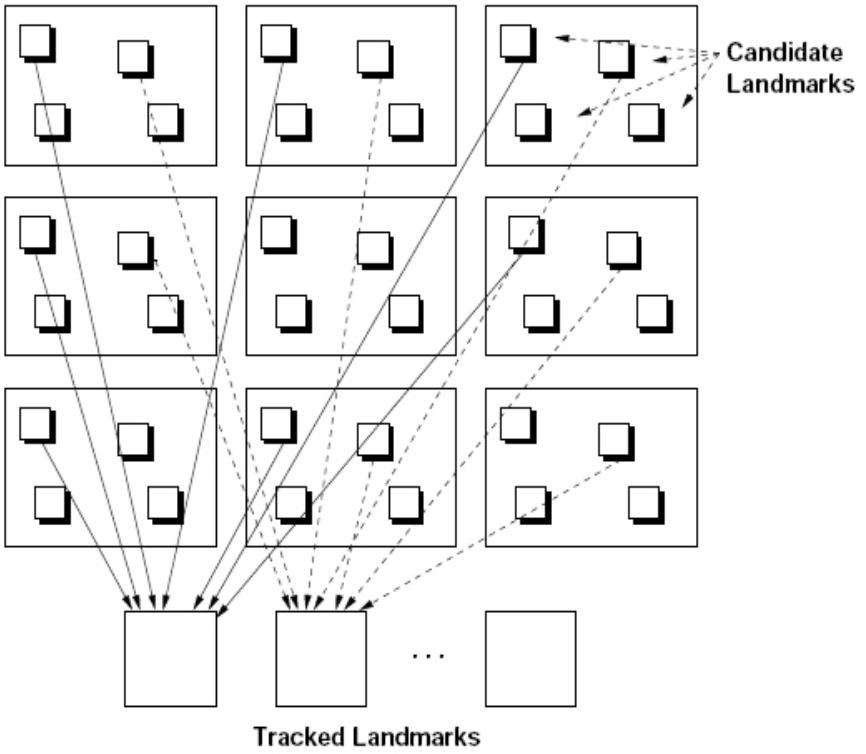
Candidate Landmark an Stelle (1,1)

Sicht

Fotografierte Bildszene

Tracking Algorithmus

- **Vergleiche alle Candidate Landmarks *cl* mit den Tracked Landmarks tl**
- **Suche nach besseren Candidates in Umgebung der *cl's* -> tauschen**
- **Wenn in Übereinstimmung cl/tl in bestimmten Bereich liegt**
  - **“Bester” cl dem tl hinzufügen**
  - **Ansonsten kreiere neuen tl aus *cl***



Online Phase

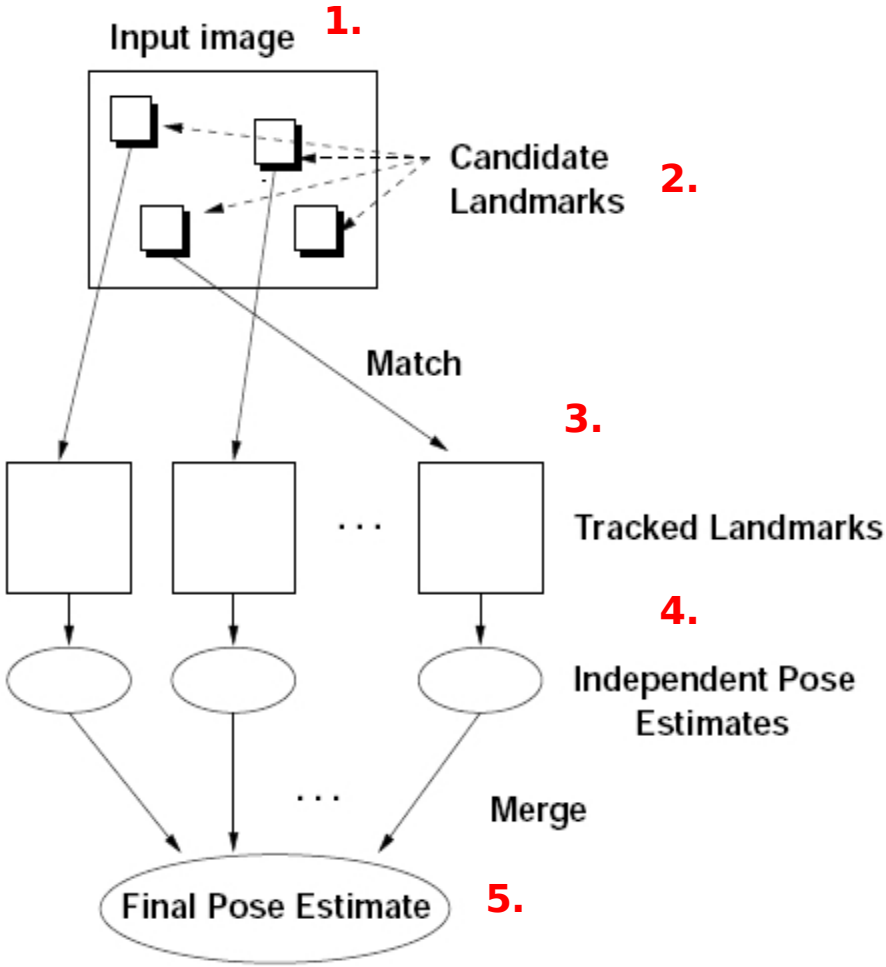
1. **“Brauche Position!”**
   -> Bild der aktuellen Sicht aufnehmen

2. **Extrahiere Candidate Landmarks *CL* aus Bild (analog Offline Phase)**

3. **Vergleich der *CL* mit den Tracked Landmarks (analog Offline Phase)**

4. **Interpolation der Positionen (der Sichten) der *CL* in den Tracked Landmarks**
   -> Positionsabschätzung der *CL*

5. **Mitteln der Abschätzungen von 4.**
   -> Positionsabschätzung des Roboters



Experiment

- **Roboter mit Kamera 1m von Beispiel-Szene entfernt**
- **Odometrie-Genauigkeit: ~1/10 mm**
- **Aufnahme von Bildern alle 2cm, Ablage in 30cm x 30cm Gitter**
- **Zusätzlich 100 Bilder aus zufälligen Positionen**

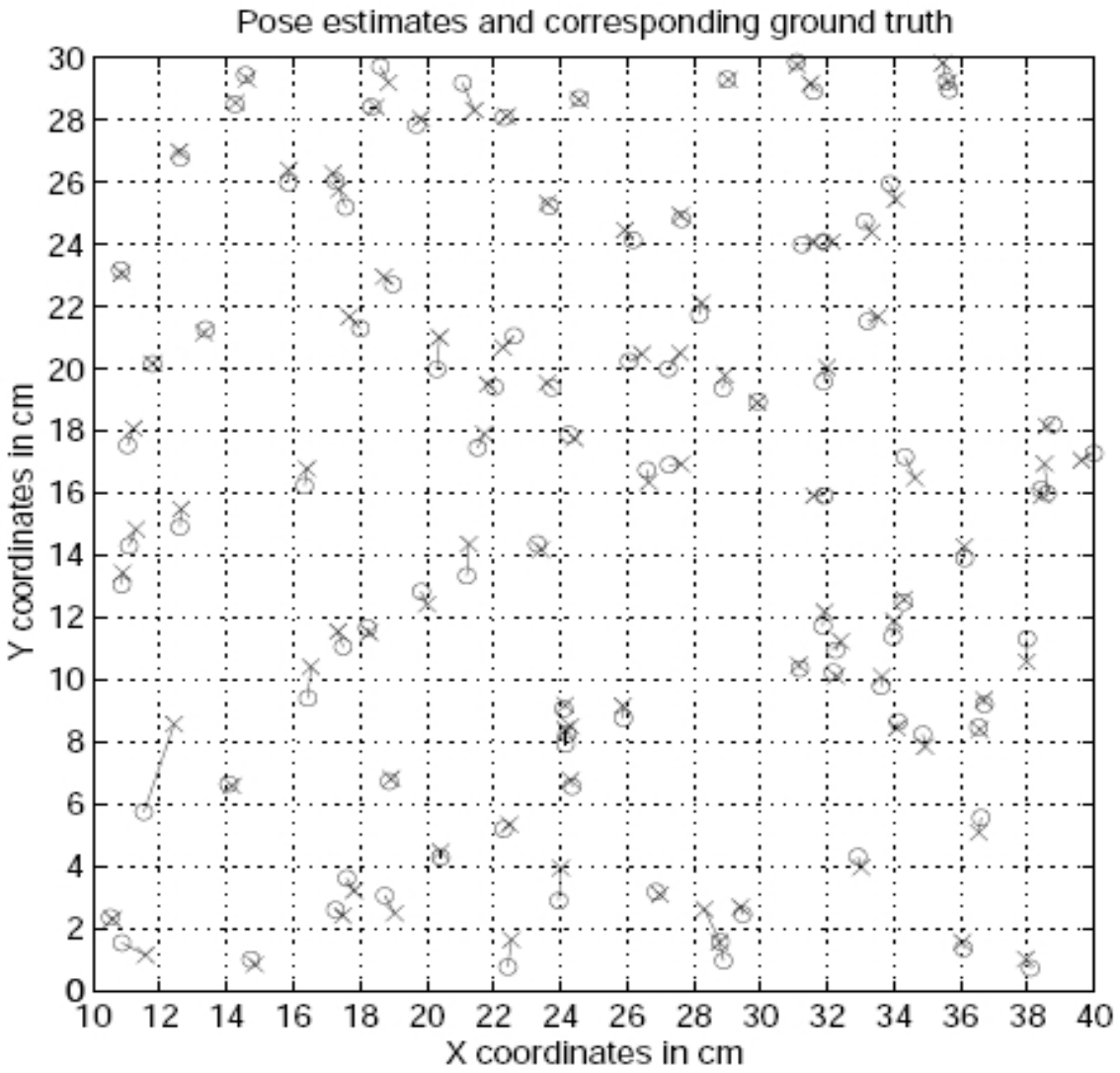
Resultate aus Experimenten

Durchschnittliche Abweichung:

~3.8mm

3.8mm < 20% Bildfrequenz(2cm)

(x) tatsächliche Position
(o) geschätzte Position



Sicht

Fotografierte Bildszene

Übersicht über vorgestellte Methoden

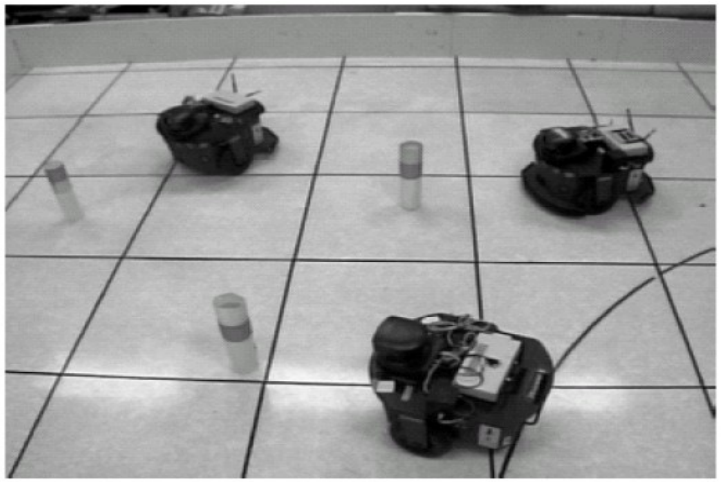
- **Learned Landmarks**

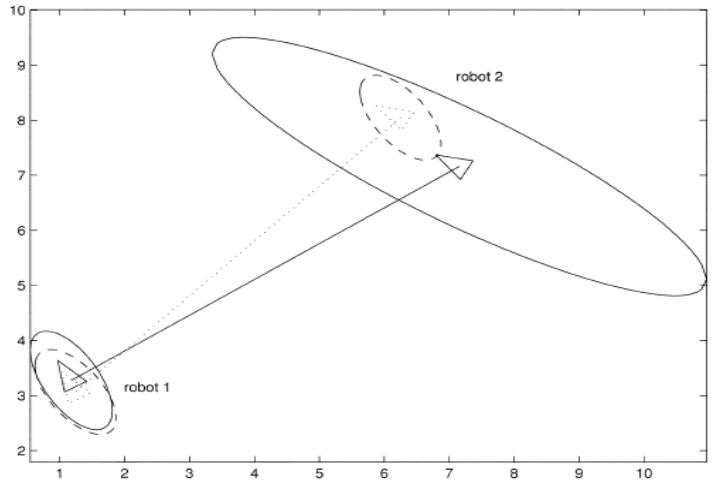
  - **Single-Robot Methode, Anwendung/Kombination in Multi-Robot System möglich**
  - **Optischer Ansatz**
  - **Kanten-Maximas von Kamera-Bildern als Angelpunkte**

- **Collective Localization (Kalman Filter)**

  - **Multi-Robot Lokalisierungsmethode**
  - **Einsatz verschiedener Sensoren möglich**
  - **Austauschen von Sensor-Informationen verbessert Genauigkeit beider Positionsabschätzungen**

Collective Localization (I)



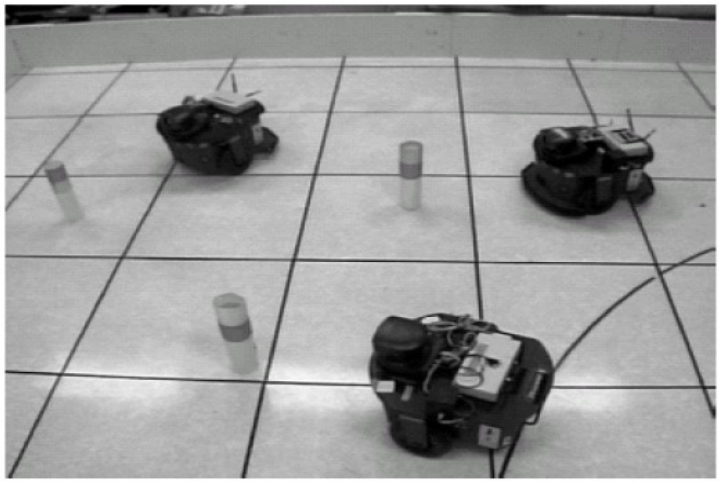


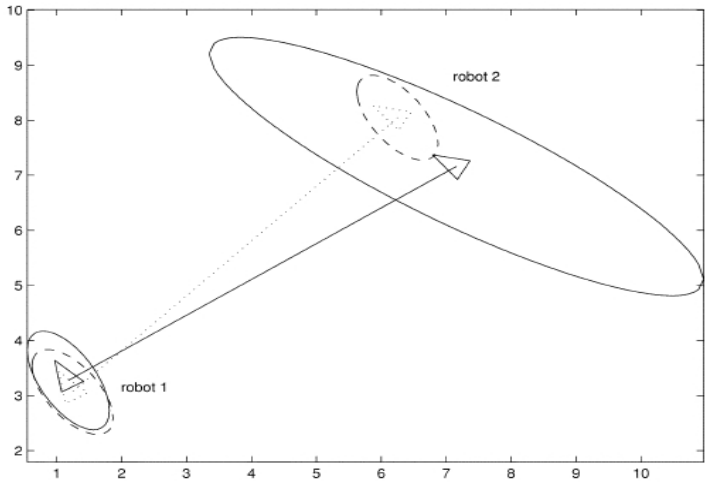
- **Gruppe voneinander unabhängiger Roboter, messen relativen Abstand**
- **Odometrie-Messungen werden dem Gegenüber bei Sichtkontakt mitgeteilt**
- **Schätzfunktion (Kalman-Filter):**

  - Sammelt Daten

  - Errechnet daraus Positionsabschätzung der Gruppenmitglieder

Collective Localization (II)





- **Keine Annahmen über Umgebung nötig (Karten etc.)**
  Andere Roboter sind “Landmarks”

- **Alle Roboter des Systems können sich gleichzeitig bewegen**
  -> kein stationärer Roboter / Basisstation nötig

- **Keine ständige Kommunikation nötig (Informationsaustausch nur bei Sichtkontakt)**

Kalman-Filter

- **Weit verbreiteter Algorithmus zur Zustandsschätzung**
- **Zustände des Systems werden aufgrund z.T. redundanter, von Rauschen überlagerten Messungen iterativ geschätzt**

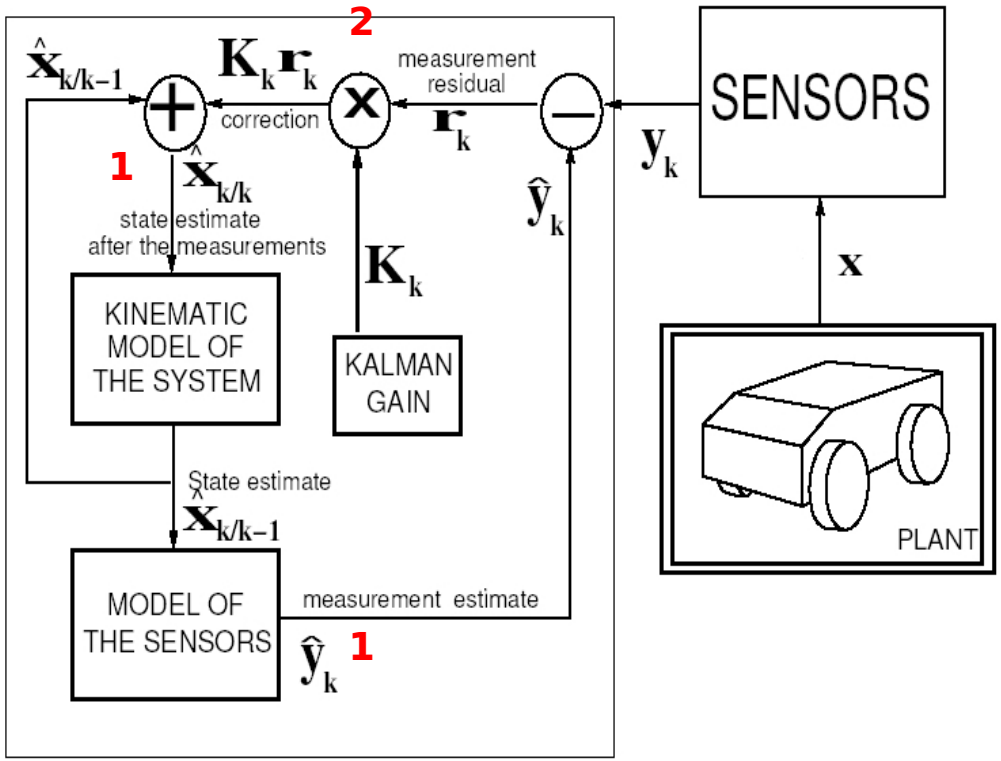
1 Wahrscheinlichster Ausgangswert wird aufgrund Eingangsdaten vorausgesagt

2 Differenz zu realem Ausgangs-wert wird linear gewichtet, Aktueller Zustand korrigiert

x = Zustand

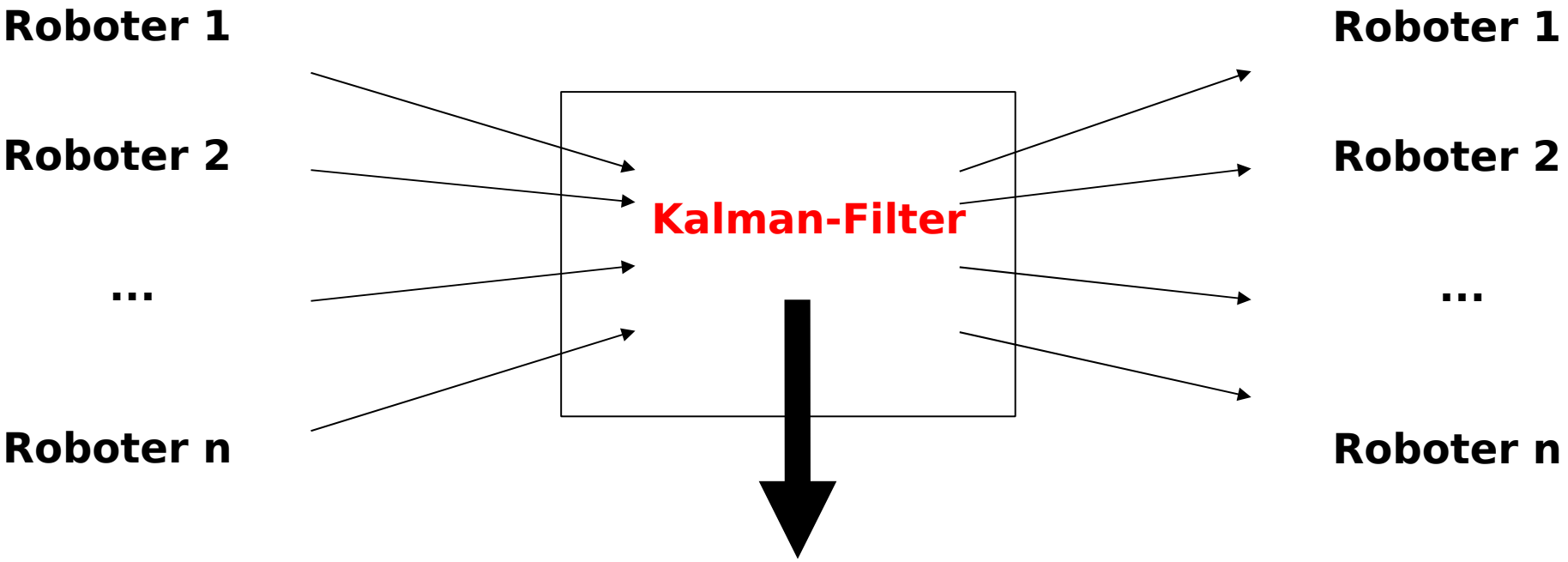
y = Messung

K = Kalman-Gain (bestimmt Gewichtung der Messung gegenüber Prädiktion)



[7]

Verteilter Kalman-Filter



-> Die bei der Zustandsschätzung involvierten (Block-)Matrizen sind diagonal

-> Bei Neuberechnung: Nur Blöcke der involvierten Roboter verändern sich

Verteilte, dezentrale Berechnung möglich

Collective Localization mit verteiltem Kalman-Filter

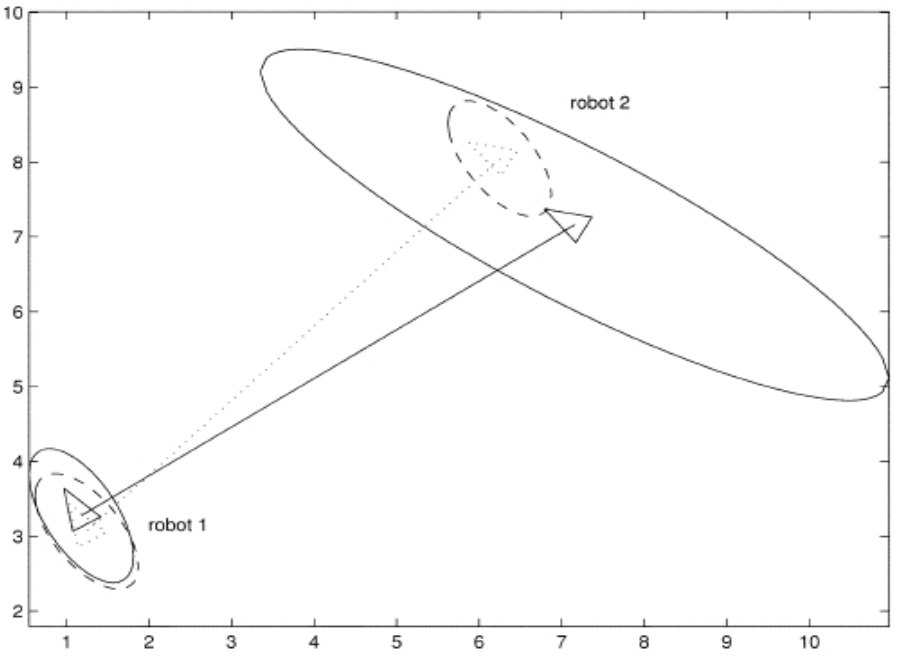
- **Anfangs kennen Roboter nur eigene Position inkl. Fehler**

- **Bei Sichtkontakt/Treffen mit anderem Roboter:**

  - Messung relativer Position (Sensor)

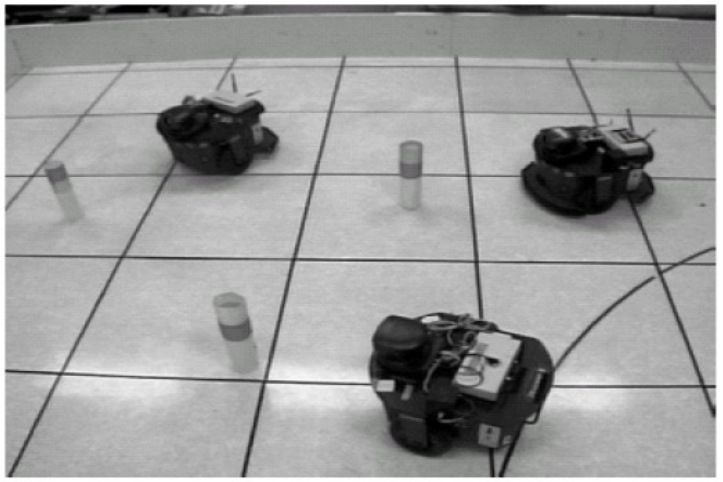
  - Austausch Zustandsschätzungen

  - Neuberechnen der Filter-Parameter und Positionsabschätzungen



-> Informationssharing verbessert Positionsabschätzungen beider Parteien

Experiment



- **3 Roboter mit verschiedenen Startpunkten manövrieren in gleichem Umfeld**

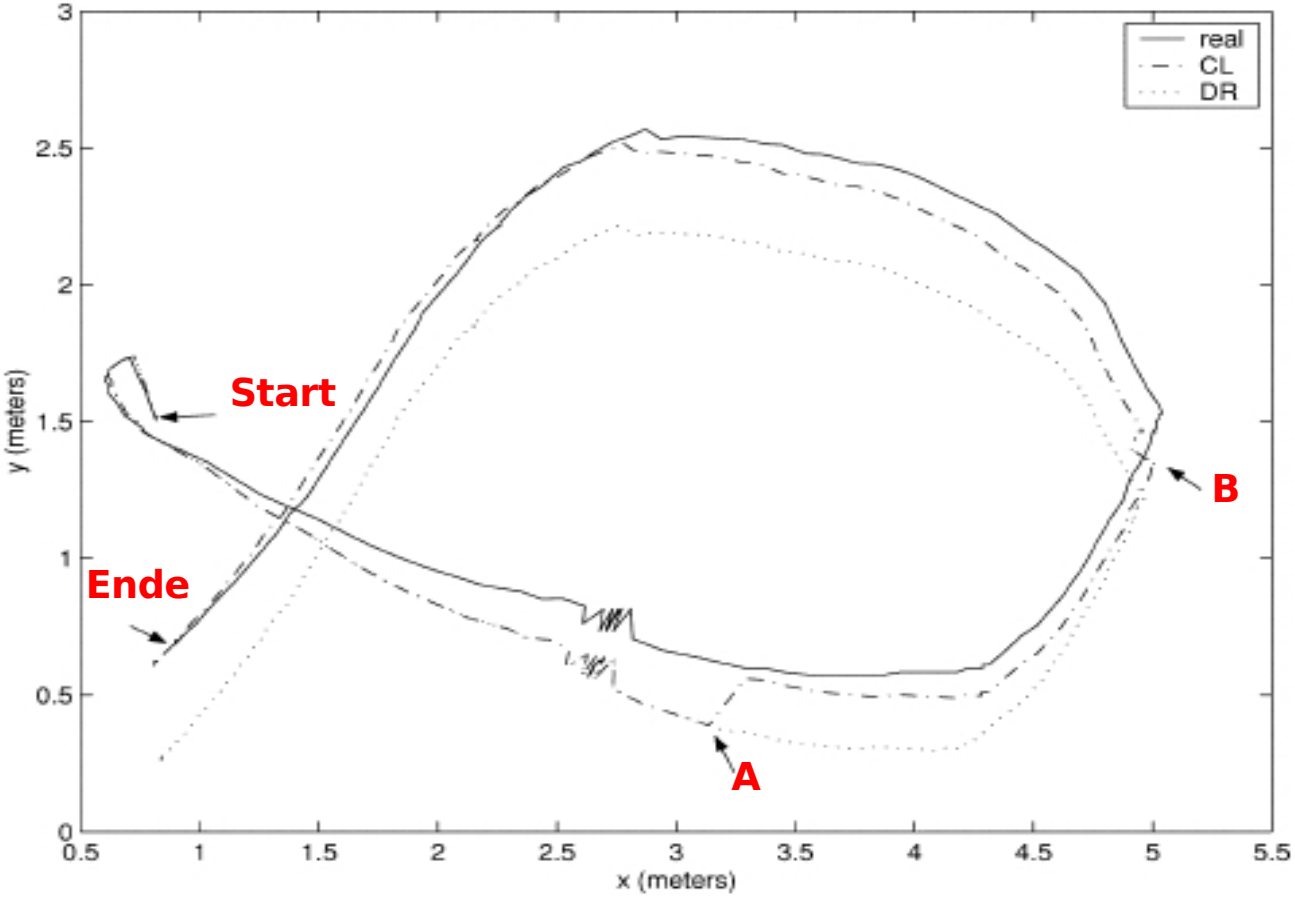
- **Kamera über Testfeld simuliert Sensoren der Roboter (relative Positionsmessung)**

  -> Genauigkeit der Sensoren veränderbar

  Hier: Genauigkeit bei Experiment: 300mm respektive 34°

Experiment mit Positionsaustausch I

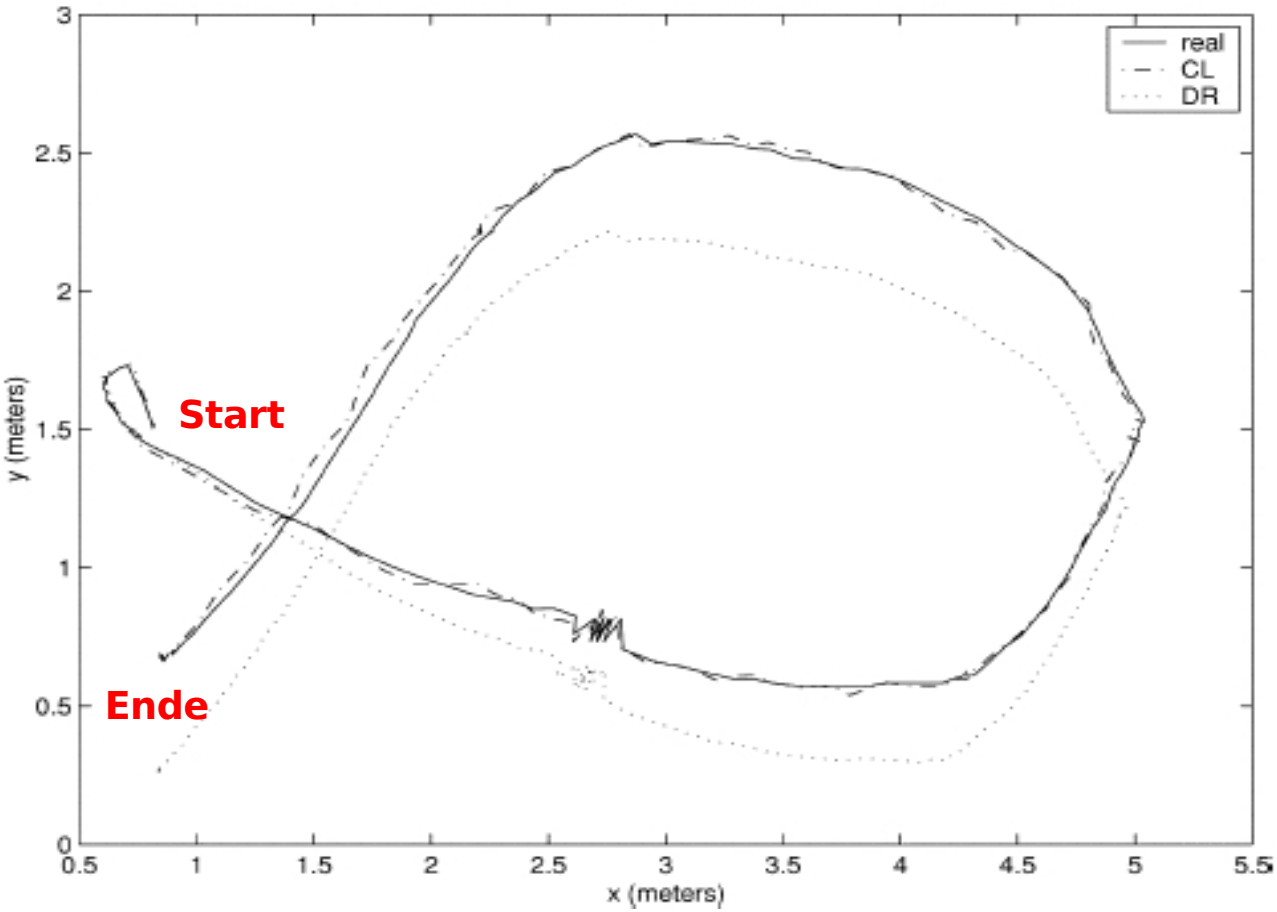
- 3 Roboter tauschen nur bei Zusammentreffen (A,B) relative Position aus
- Trajektion Roboter 2
- Fehler x-Richtung:
  ~ 224mm



- Reale Position
- Collective Localization
- Odometrisch

Experiment mit Positionsaustausch II

- **3 Roboter tauschen andauernd relative Position aus**
- **Trajektion Roboter 2**
- **Fehler x-Richtung: ~ 39mm**



Reale Position

Collective Localization

Odometrisch

Experiment mit Positionsaustausch III

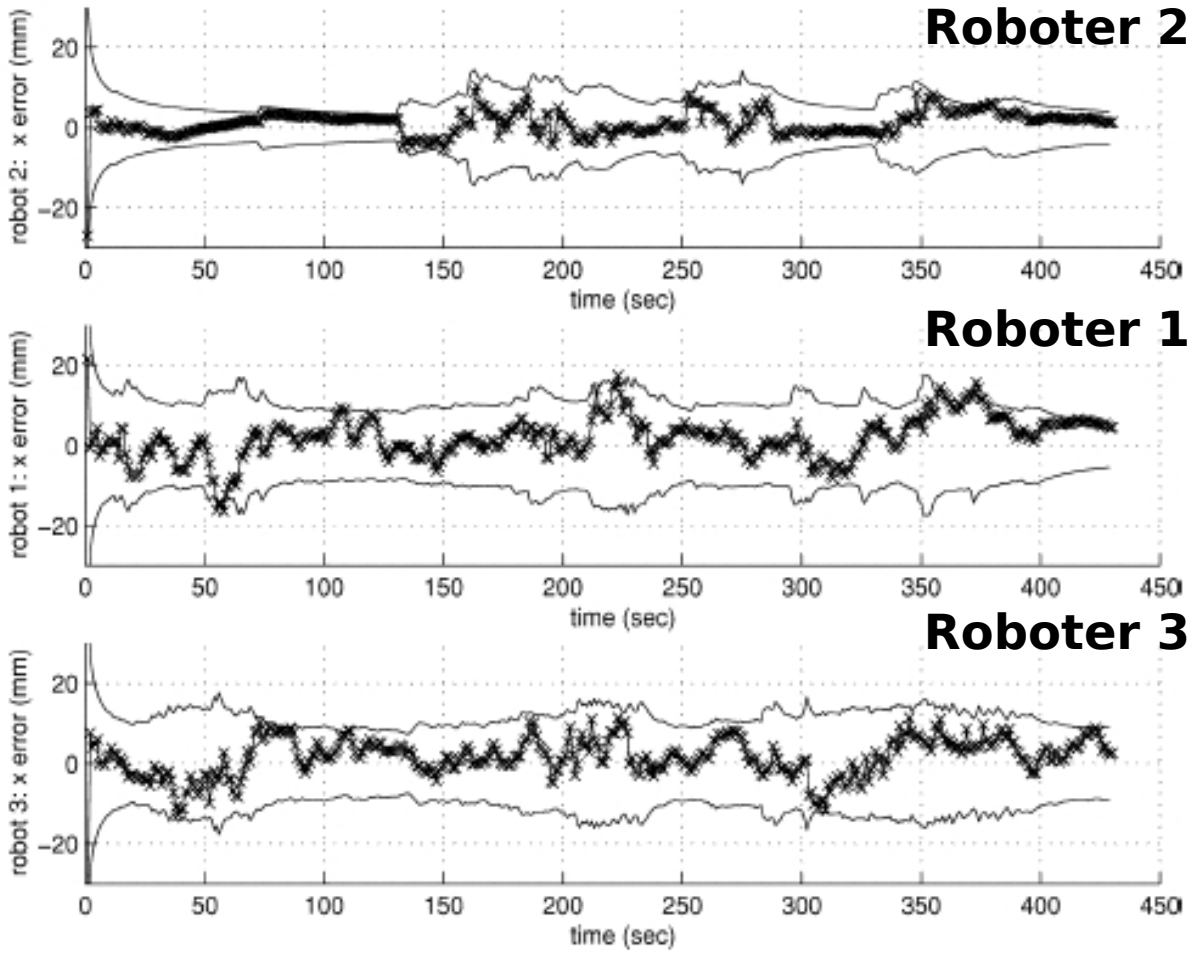
- **3 Roboter tauschen andauernd relative Position aus**

- **Roboter 2 (oben) mit absoluter Positions-information**

- **Roboter 1 & 3 provitieren von 2**

- **Fehler in x-Richtung:**

  Roboter 2: < 20 mm
  Roboter 1: < 20 mm
  Roboter 3: < 20 mm



Experiment ohne Posistionsaustausch

- **Roboter tauschen keine Positionsinformation aus**

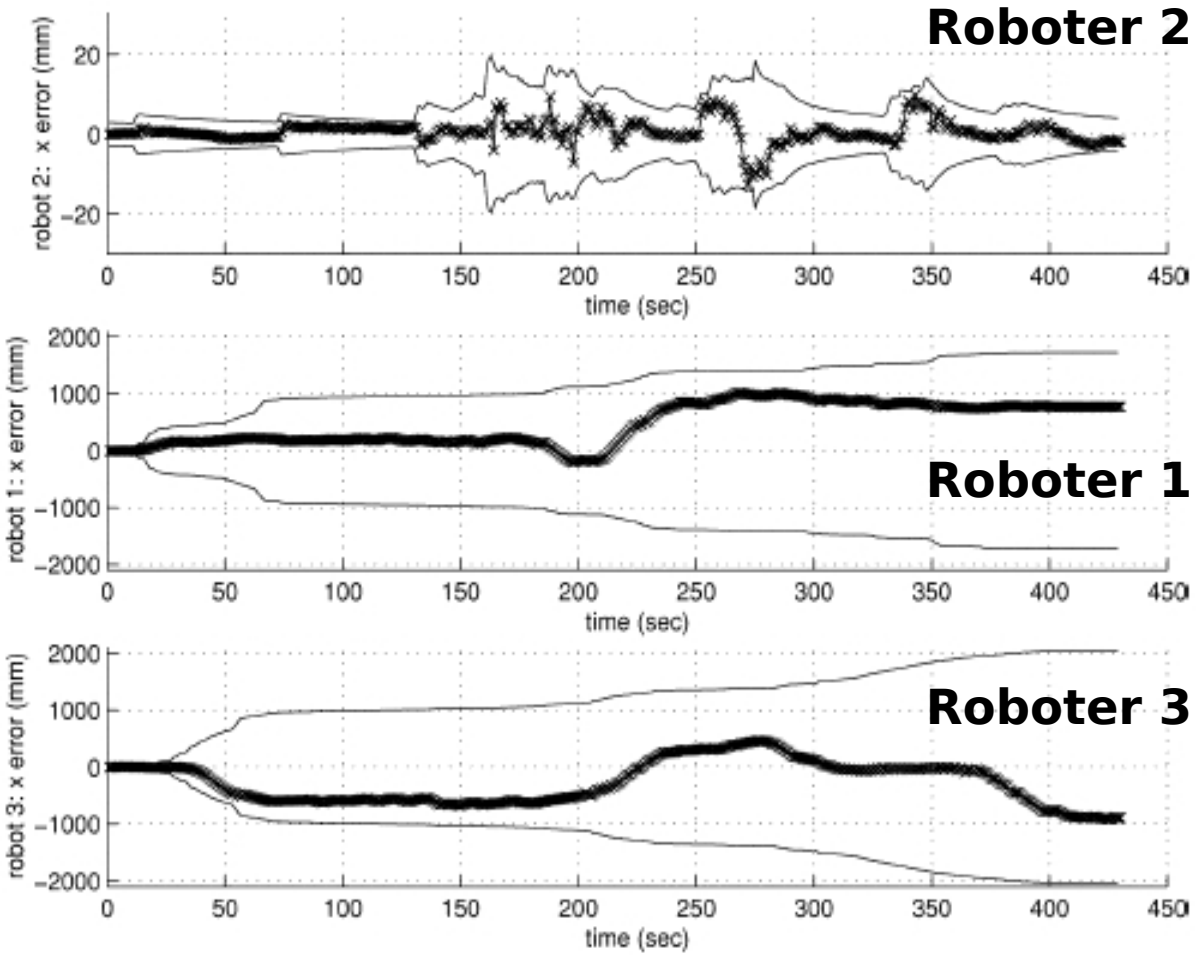
- **Roboter 2 (oben) mit absoluter Positions-information**

- **1 & 3 messen nur Eigenbewegung**

  -> Fehler summiert sich ständig

- **Fehler in x-Richtung**

  Roboter 2: < 20 mm
  Roboter 1: 767 mm
  Roboter 3: -891 mm



Virtuelle Pheromone

- **Beispielanwendung, die auf einer möglichen Implementierung der relativen Abstandsmessung (Infrarot) aufbaut**

- **Vision:**
  Autonomes Mobiles System als Art “Superorganismus” bestehend aus kleinen autonomen Elementen, die als Ganzes agieren

- **Virtuelle Pheromone: Biologisch inspiriertes Konzept für Informationsaustausch**

Virtuelle Pheromone

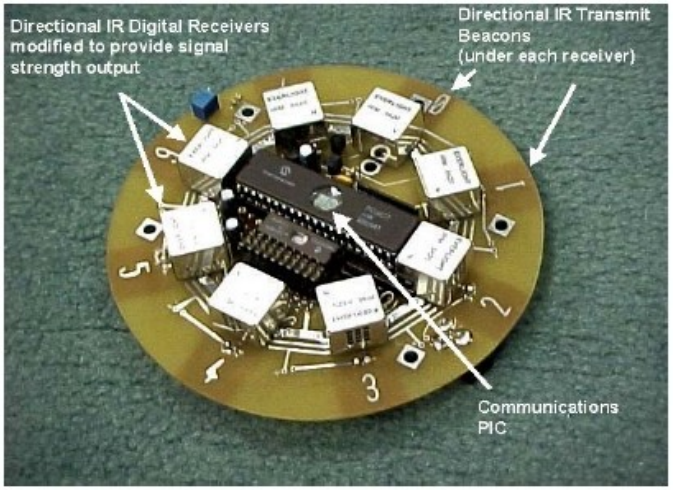
- **Verbund von Robotern, ausgestattet mit einem Ring von Infrarot-Sensoren**

- **Optische Signale als virtuelle Pheromon Implementierung**

  Infrarot, weil:

  - Gerichtet (Kodierung der Richtung)

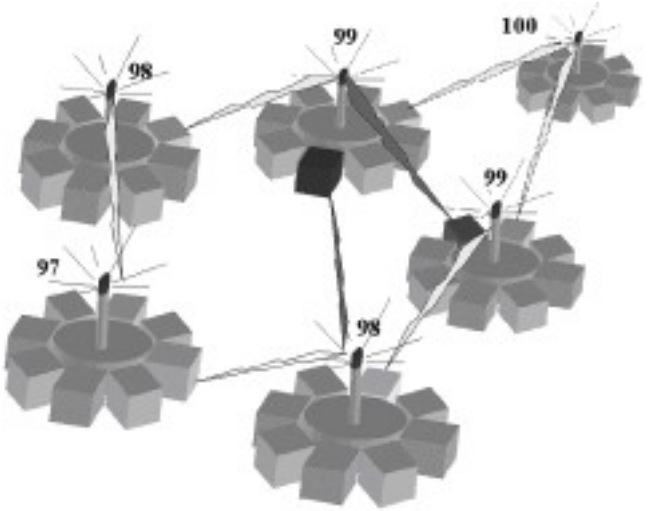
  - Intensitätsverlust (Distanz Abschätzung)



- **PalmV PDA als Recheneinheit sowie für In-/Output**

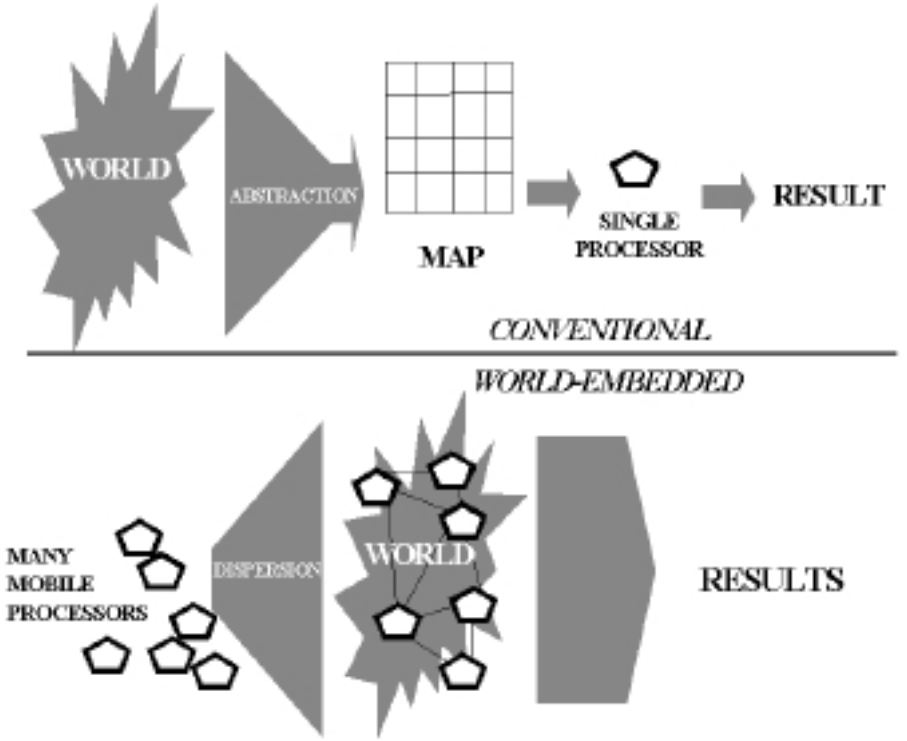
- **Signal-Intensität und Orientierung den einkommenden Nachrichten hinzufügen -> Distanzabschätzung, Detektierung von Objekten**

Virtuelle Pheromone



- **Virtuelle Pheromon Nachricht:**
  - **Typen Feld, Hop-Count Feld Daten Feld**
- **Nachrichten Verteilung:**
  - **Dekrementieren des Hop-Counts bei Empfang**
  - **Weitersenden der Nachricht in alle/ausgewählte Richtungen**
  - **Bei mehrfachem Empfang: Ignoriere Nachrichten mit Hop-Count <= bereits empfangenem Hop-Count**
  - **Sender braucht sich (meist) nicht um Empfang zu kümmern**
- **Einzelne Roboter bilden Knoten in “Superorganismus” mit lokalen Sensor-Fähigkeiten**

Roboter-System Ansatz

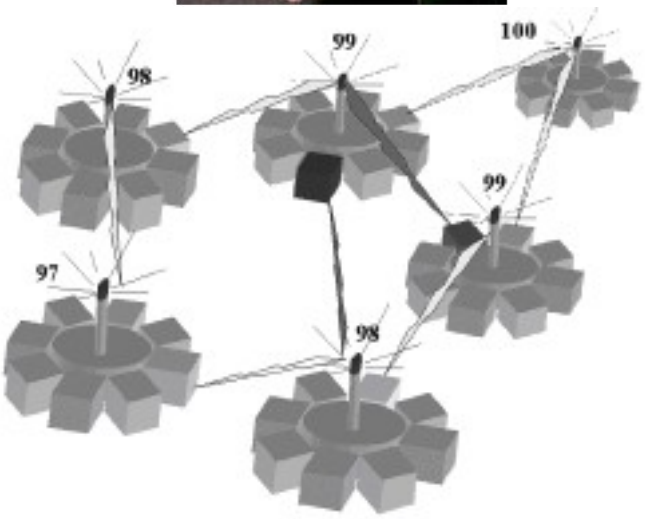


- **Virtuelle Pheromone ermöglichen Analyse der Umgebung**
- **Jedes Mitglied im “Superorganismus” liefert Beitrag zu Berechnungen**

  -> Bsp. Kürzeste Route, Blockierte Wege, Ausweichpläne können verteilt berechnet werden

Anwendungsbeispiele

- **Selbständige Verteilung in unbekanntem Raum**
  - **Wegbewegung von Hindernissen und anderen Robotern**
  - **Anziehende/Abstossende Kräfte um in gewünschte Distanz zu gelangen**



- **Virtuelle Weganzeiger**
- **Zusammenarbeit von heterogenen Robotern**

Sobald spezieller Roboter benötigt wird:

Source:	**Sende *Search-Pheromon***
Angesprochene Roboter:	**Sende *Distance-Pheromon***
	(Source-Entfernung wird hizugefügt)

-> wirkt hemmend auf Roboter die weiter weg von Source sind

Zusammenfassung

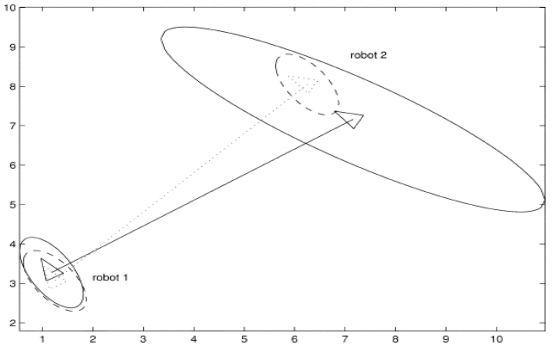
- **Lokalisierung in Autonomen Mobilen Systemen**

  -> Präzise Ortsbestimmung

- **Verschiedene Methoden**

  Allein -> Bsp. *Learned Landmarks*

  Team -> Bsp. *Collective Localization*



- **Applikationen**

  Virtuelle Pheromone

Referenzen

- [1] Tamio Arai, Enrico Pagello, and Lynne E. Parker: Advances in Multi-Robot Systems, IEEE Transactions on Robotics and Automation, 18(5): 655-661, 2002.
- [2] Mobile robot localization from learned landmarks, Robert Sim and Gregory Dudek, Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems (IROS), Victoria, BC, Oct 1998
- [3] Stergios I. Roumeliotis, George A. Bekey. Distributed Multirobot Localization. IEEE Transactions on Robotics and Automation, Vol. 18, No. 5, 2002
- [4] Payton, D., R. Estkowski, and M. Howard (2004). Pheromone robotics and the logic of virtual pheromones. In Proc. SAB 2004 Int'l Workshop Swarm Robotics, Volume 3342 of LNC
- [5] http://www.cim.mcgill.ca/~simra/publications/Thesis/
- [6] http://swarm-robotics.org/SAB04/presentations/payton-review.ppt
- Die meisten Bilder von [1], [2], [3], [4], [5], [6], sowie
  [7] http://www.art-elektronik.dk/Acer_CS-5530.gif
  [8] http://www.cs.cmu.edu/~robosoccer/cmrobobits/lectures/Kalman.ppt

Backup - Tracking Algorithmus

1. **Für alle Candidate Landmarks *cl* eines Bildes:**

   a) **Für alle Tracked Landmarks *tl* in der DB:**

      i) **Suche in Umgebung von *cl* nach einem besseren Match *cl'* zu *tl***

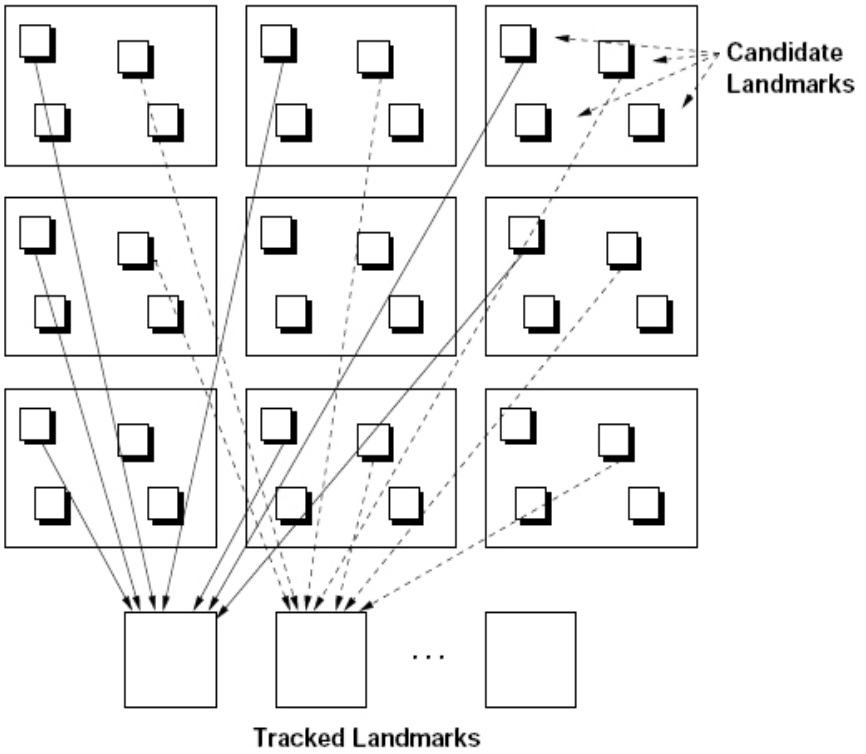
      Wenn ein *cl'* gefunden wurde, tausche *cl'* mit *cl* als Kandidat für *tl*

   b) **Wähle denjenigen *tl* aus, der in a) am besten mit *cl* übereinstimmte**

2. **Wenn *cl* bester Match für *tl* von allen anderen cls und Match innerhalb eines Schwellbereichs liegt:**

   - füge *cl* *tl* hinzu

   - ansonsten kreiere neues Set aus *cl*



Backup – Positionsabschätzung Candidate Landmarks

- **Learned Landmark *l*, der mit Tracked Landmark *T* übereinstimmt**
- **Feature Vektor *f* : | k p c |**
  k = Kodierung des Landmarks
  p = Bildposition von *l*
  c = Kameraposition von der *l* aufgenommen wurde
- **Anfangs c initialisiert als Mittelwert der c's aller anderen *l*'s in *T***
- ***F* mit *fi's* als Kolonnenvektoren, *Uf* aus Singulärwertzerlegung**
  g* = *Uf* ' *fi
  fi_est* = *U g
- **Iteriere bis Schätzung *fi_est* nicht mehr ändert**
- **c = Positionsabschätzung**