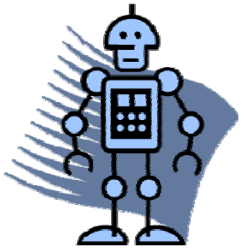
Navigation mobiler Systeme

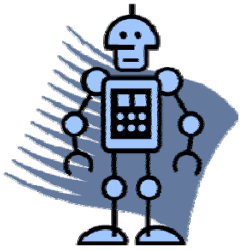
Dipl.-Inform Ingo Boersch
In LV „Autonome Mobile Systeme“



„Navigation ist eine der grundlegendsten Aufgaben eines mobilen Roboters.
Ein mobiler Roboter sollte fähig sein, einen Auftrag in der Art:
*„**Fahre zum Zielort x**“, ausführen zu können.“*

[Roland Stenzel: Steuerungsarchitekturen für autonome mobile Roboter,
Dissertation an der RWTH Aachen, 2002]

Drei Fragen der Navigation



1. Wo bin ich? (Self localization)

- *Relative Positionierung mit internen Sensoren*
- *Absolute Positionierung anhand von Marken*
- *Probabilistische Positionierung*
- *Übung: Lokalisierung und Bewegungsunschärfe*

2. Wohin soll ich gehen? (Task Planing)

3. Wie komme ich dahin? (Path Planing)

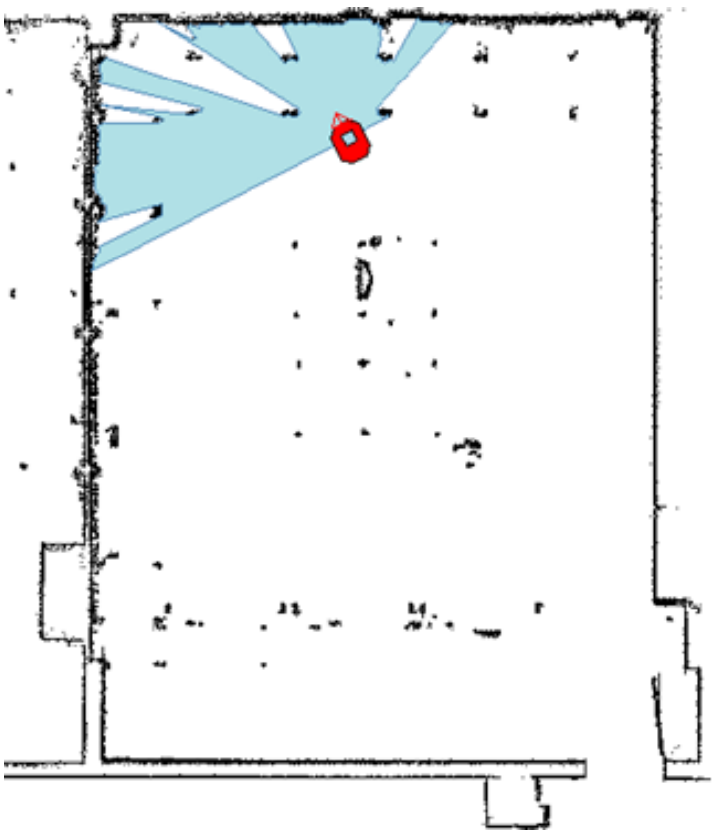
- *Karten*
- *Pfadplanung*
- *Navigationsphänomene bei Lebewesen*
- *Übung: Kartenobjekte und ArActionGoto*

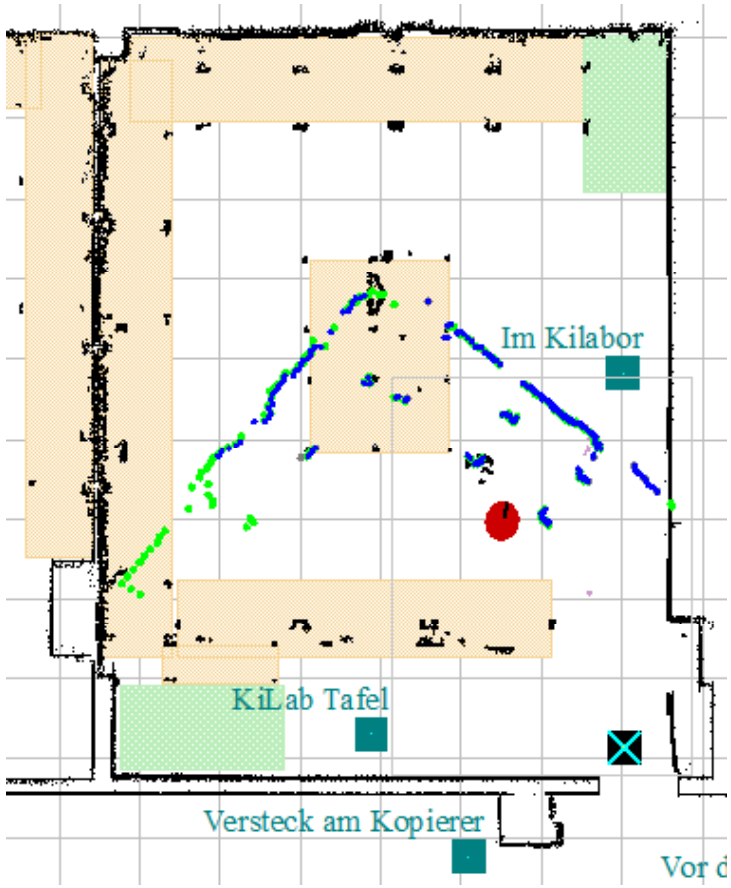
Problemstellung „Wo bin ich“

- Realität

Positionsglaube

- hier Irrglaube



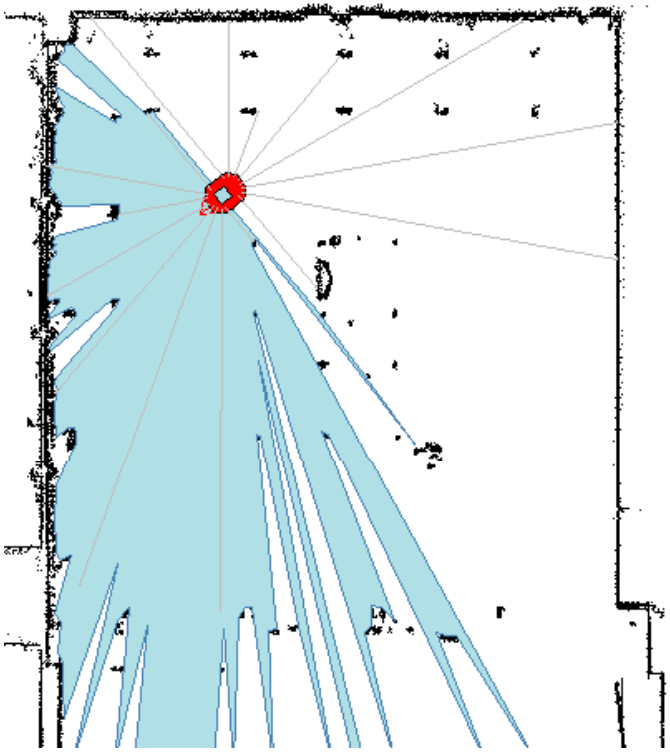


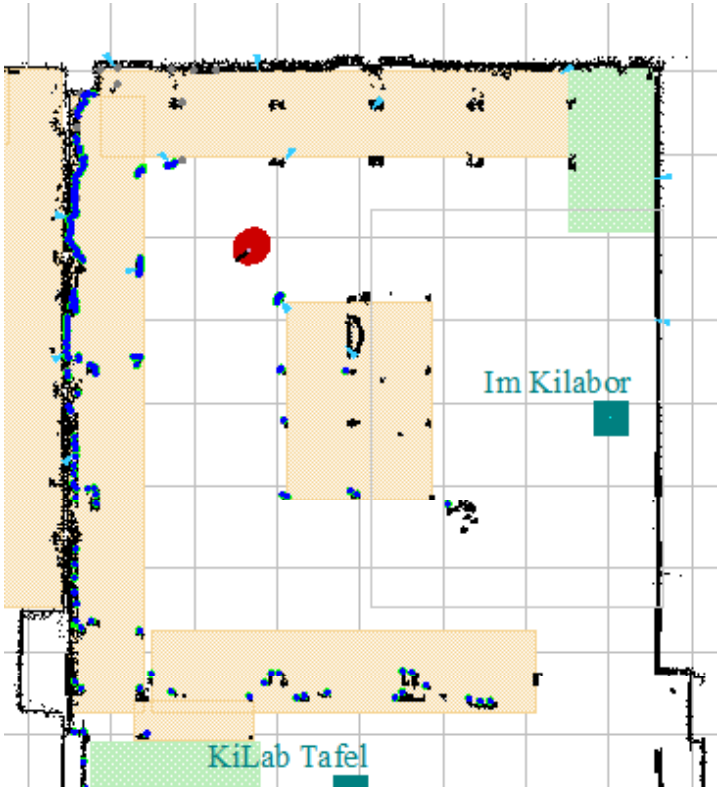
Ziel

- Realität

Positionsglaube

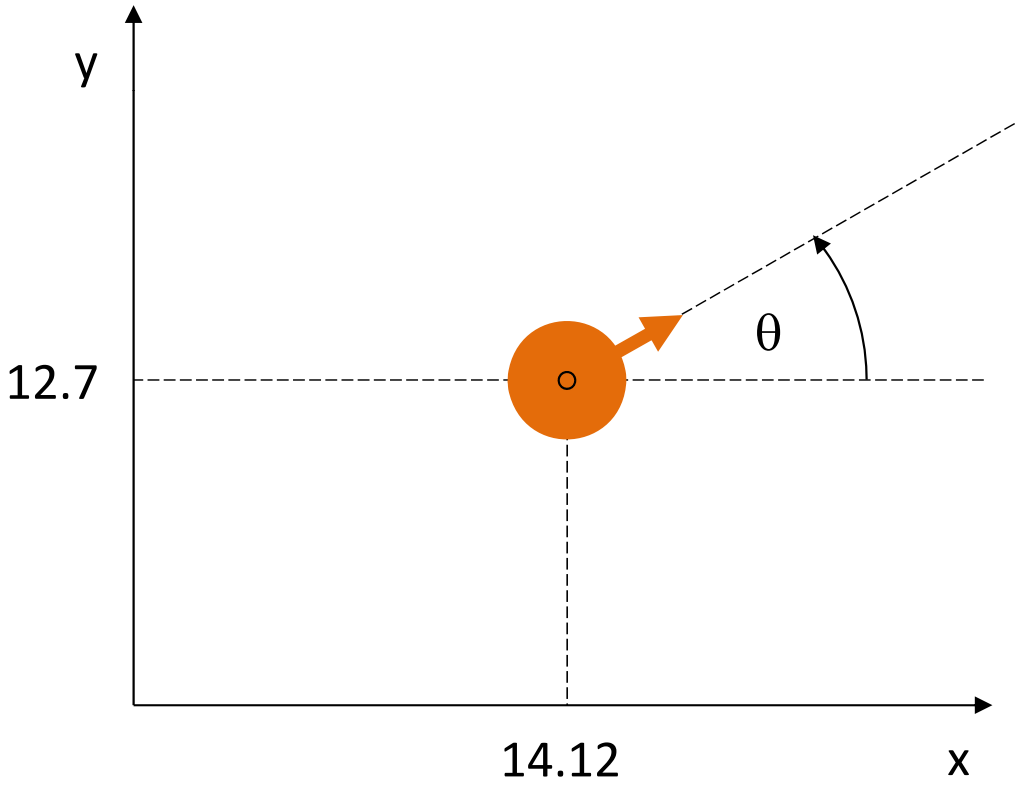
- perfekt lokalisiert





Position in der Ebene

- globales Koordinatensystem
- Roboterposition (x, y, θ) in der realen Welt, 3 Freiheitsgrade
- Position ist nicht direkt messbar.



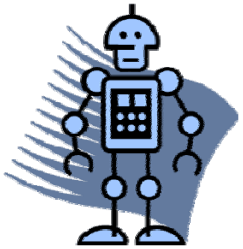
Beispiel:
(14.12, 12.7, 27°) ist die Realität.

Aber was „glaubt" der Roboter?

(Positionsglaube, engl. belief)

- > Wo bin ich?

Selbstlokalisierung, Positionierung



1. Wo bin ich? (Self localization)

Relative Positionierung
Alte Position + Beobachtung der Eigenbewegung => Position

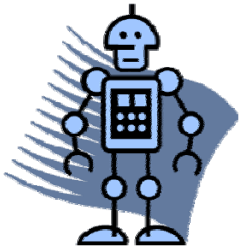
- Odometrie
- Trägheits-navigation

Absolute Positionierung
Beobachtung der Umwelt => Position

Probabilistische Lokalisierung
Synthese

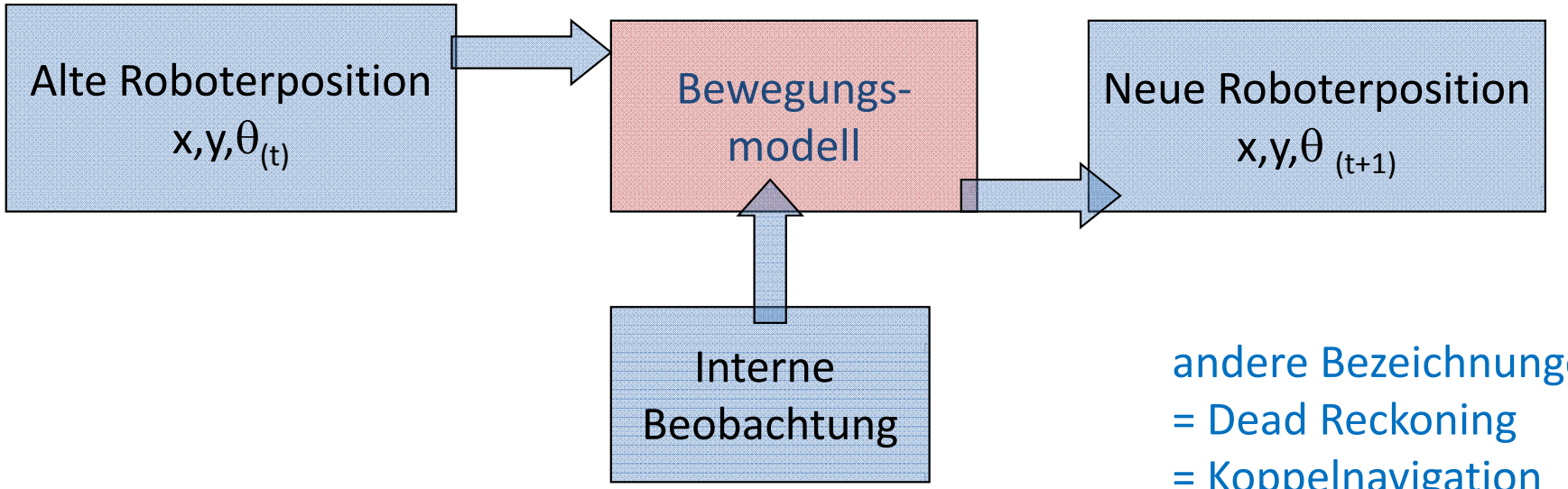
- Monte Carlo Localization in ARIA

Relative Positionierung



- **Def.:** Positionsschätzung mittels Beobachtung der eigenen Bewegung oder der Bewegungsorgane oder des Bewegungsbefehls (ohne Sensorik)

biologisch:	Propriozeptoren, z.B. blindes Laufen
technisch:	Odometer, Trägheitssensoren (a, v, ω, ω) etc.

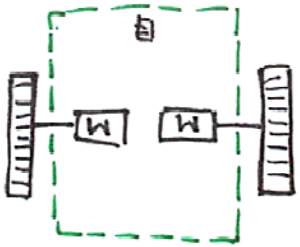


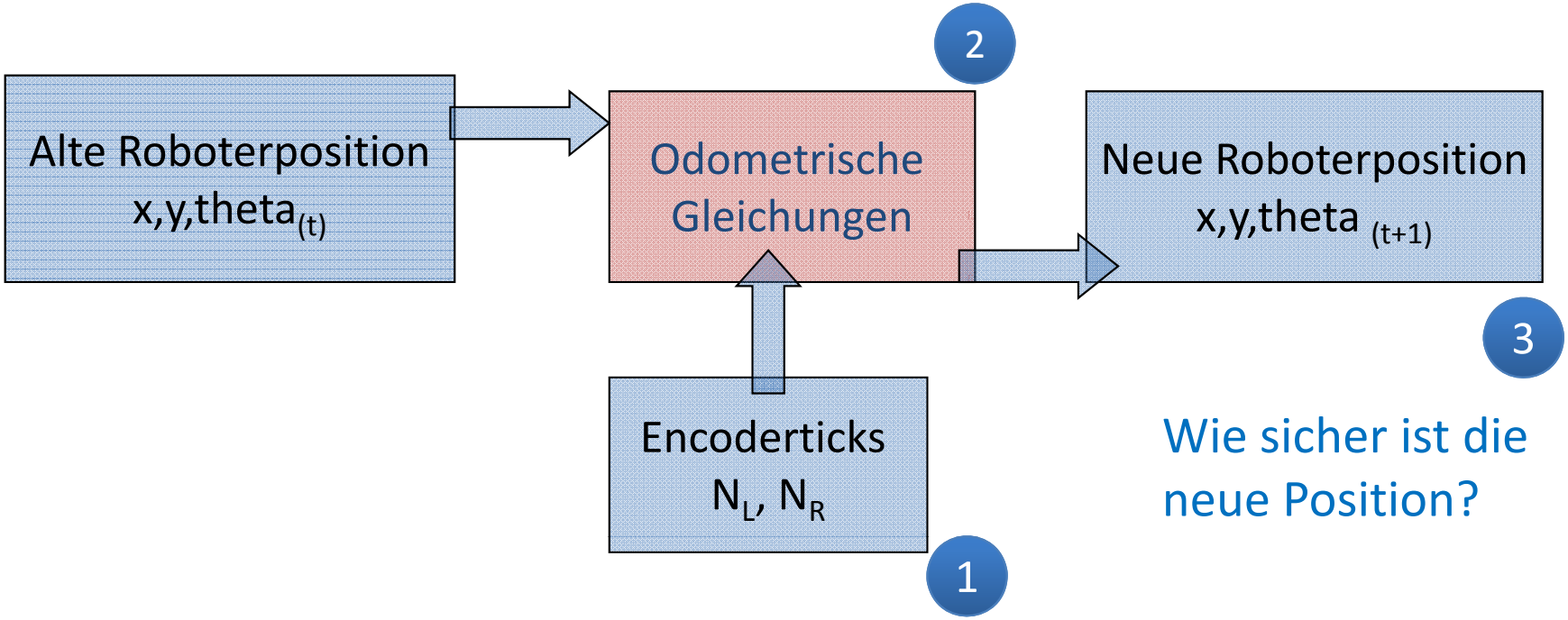
andere Bezeichnungen:
= Dead Reckoning
= Koppelnavigation

Odometrie

- **Def. Odometrie:** Relative Positionierung mittels Beobachtung der Antriebsräder

- Antriebsabhängig
- Bsp: differentieller Antrieb:





Wie sicher ist die neue Position?

Odometrische Sensoren (optische)

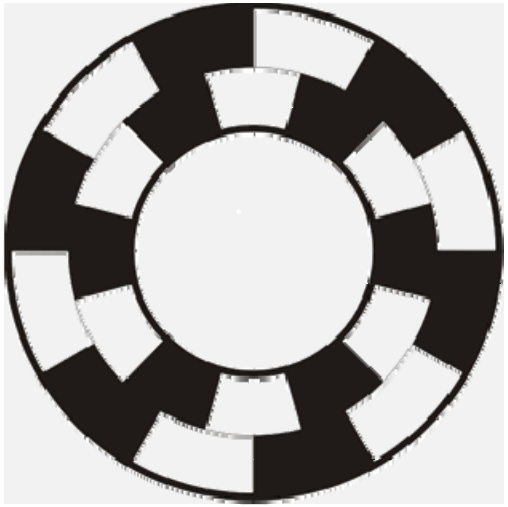
1

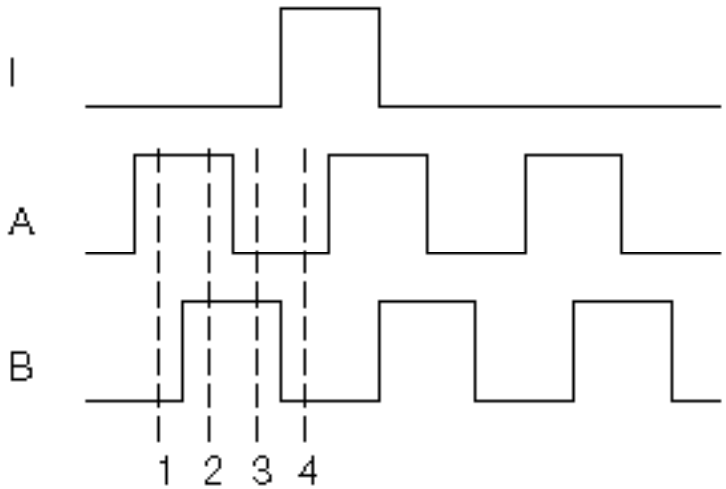
- **Inkrementelle Encoder:**
  - Anzeige der Bewegung einer Achse durch eine Impulsfolge
  - Positionsverlust beim Ausschalten ==> Gelenke beginnen in Endstellung
  - Berechnung notwendig
  - Bsp: Phase-Quadratur-Encoder am Pioneer

- **Absolute Encoder:**
  - Anzeige der Position einer Achse durch eine Postionskodierung
  - 1 Bit/Spur ==> Baugröße abhängig von der Auflösung
  - Bsp: Binärcode-Encoder, Graycode-Encoder

Phase-Quadratur Encoder

- Pioneer-Roboter
- Messprinzip: 2 Spuren, phasenverschoben um 90°
- => doppelte Auflösung + Richtungsinformation
- zwei Licht/IR-Sensoren notwendig





State	Ch A	Ch B
S_1	High	Low
S_2	High	High
S_3	Low	High
S_4	Low	Low

Figure 1.1: The observed phase relationship between Channel A and B pulse trains can be used to determine the direction of rotation with a phase-quadrature encoder, while unique output states S_1 - S_4 allow for up to a four-fold increase in resolution. The single slot in the outer track generates one index pulse per disk rotation [Everett, 1995].

Zustandsdiagramm

Absolutes Odometer

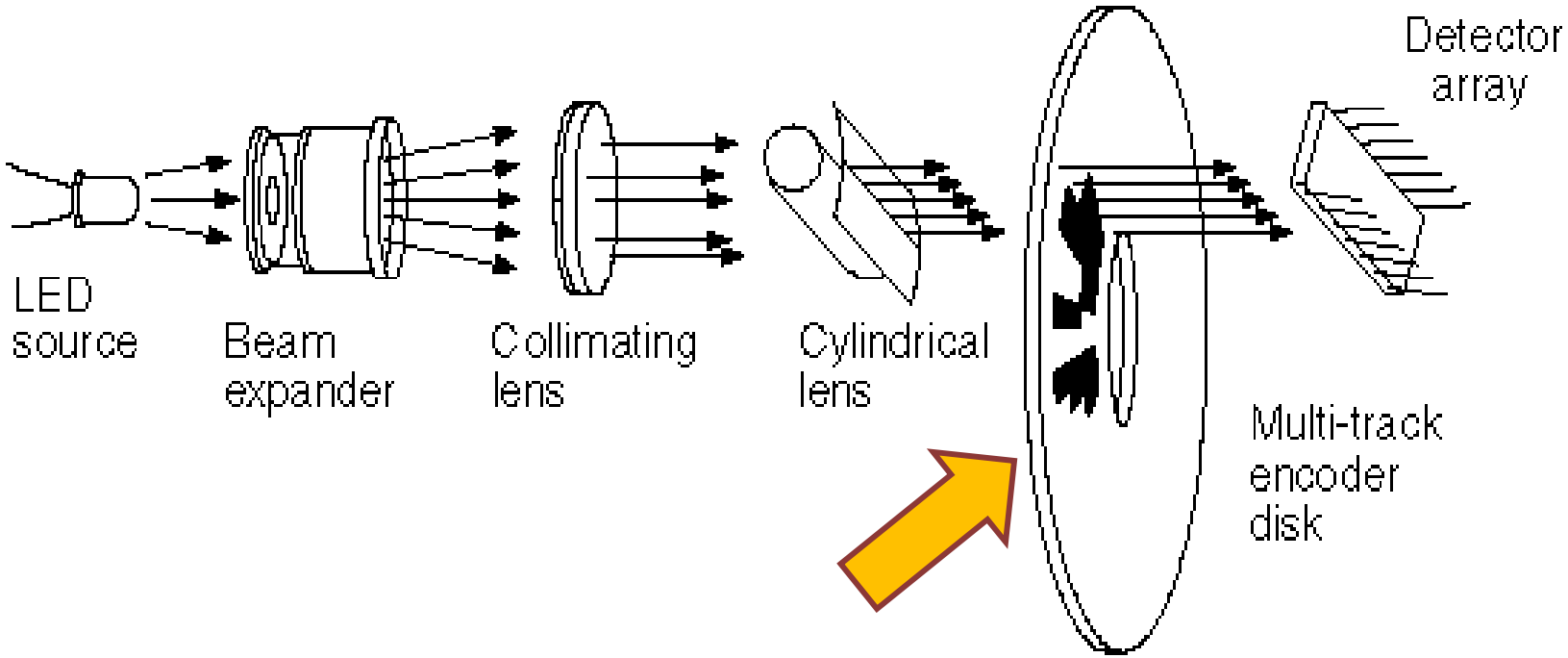
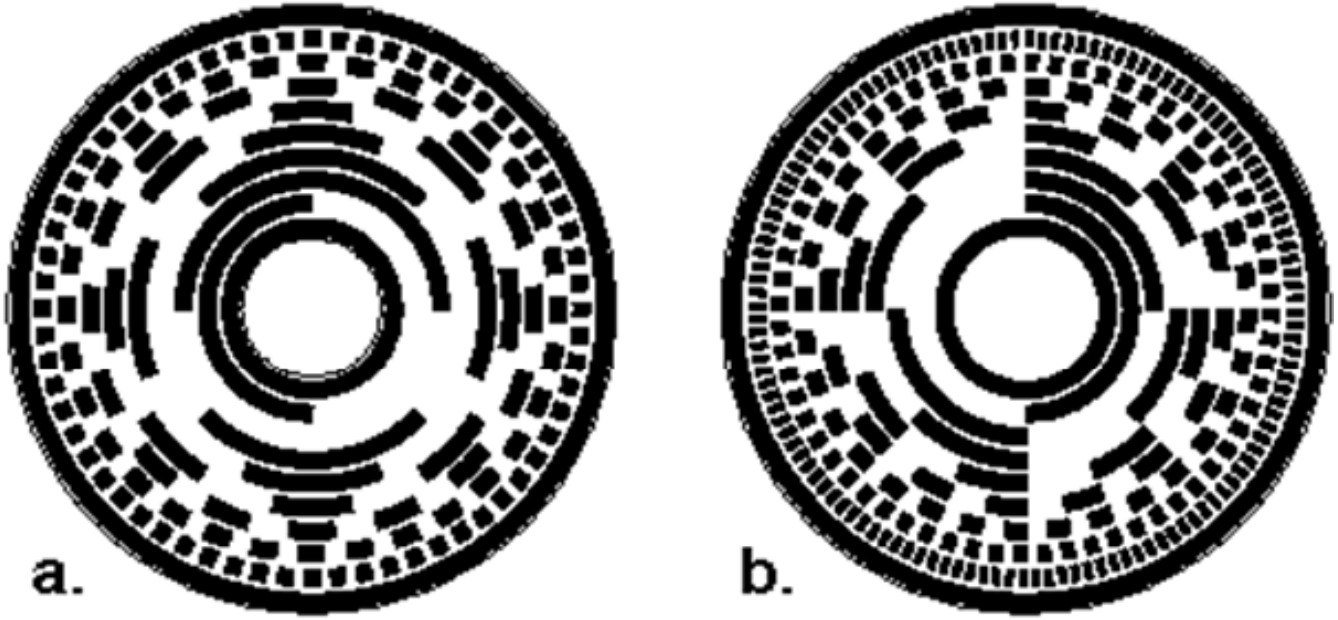


Figure 1.2: A line source of light passing through a coded pattern of opaque and transparent segments on the rotating encoder disk results in a parallel output that uniquely specifies the absolute angular position of the shaft. (Adapted from [Agent, 1991].)

Gray-Code und Binärcode

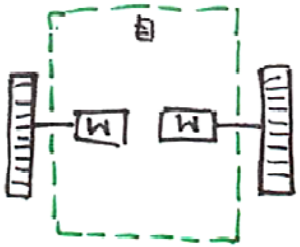


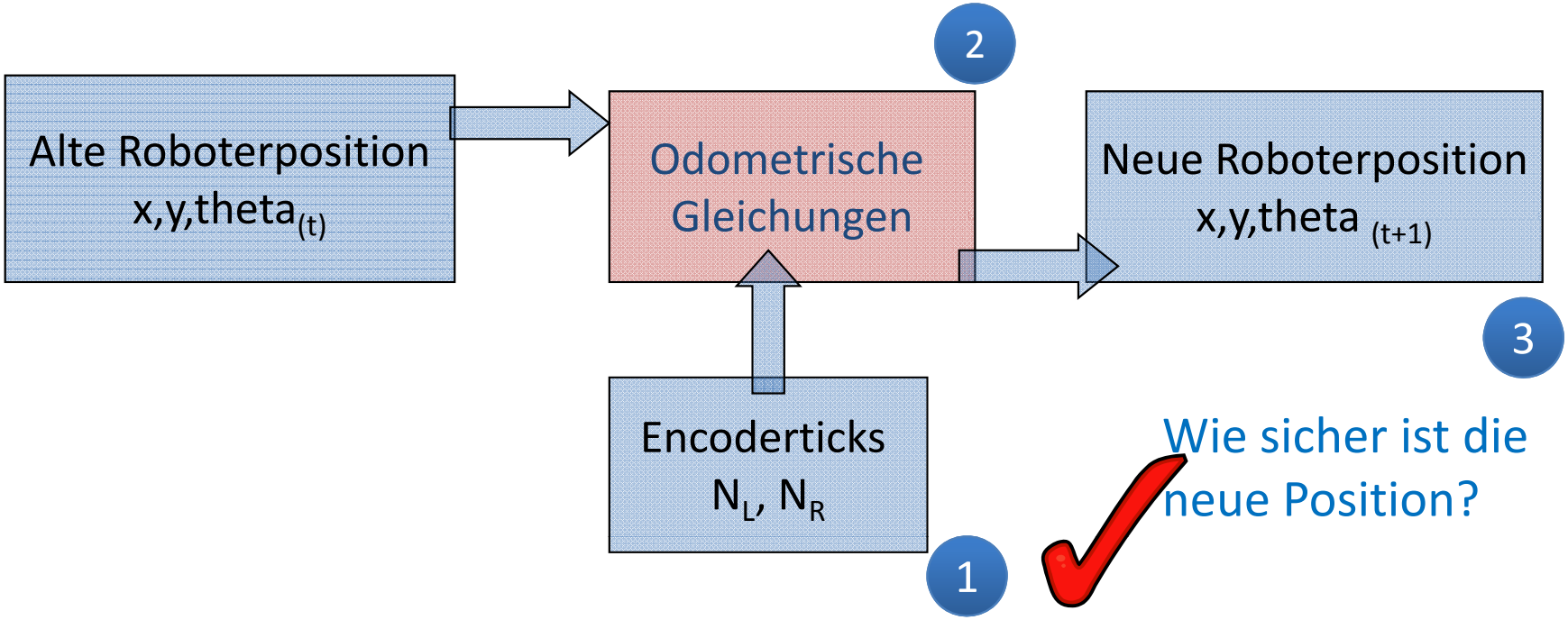
- Welche der Code-Scheiben ist der Gray-Code, welche der Binärcode?
- Warum ist der Gray-Code für absolute Odometer besser geeignet?
- a) Gray-Code b) Binärcode
- Beim Gray-Code ändert sich von Codewort zu Codewort genau ein Bit, so dass keine Hammingklippen (wie bei 12 Uhr des Binärcodes) entstehen, die bei langsamem Drehen zu falsch gelesenen Codeworten führen können.

Odometrie

- **Def. Odometrie:** Relative Positionierung mittels Beobachtung der Antriebsräder

- Antriebsabhängig
- Bsp: differentieller Antrieb:



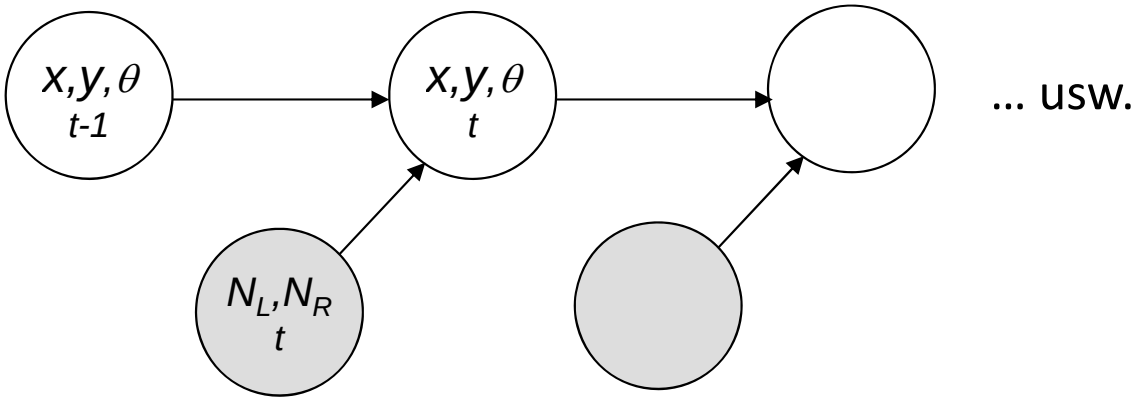


Bewegungsmodell 2

- Berechnung der neuen Position
- Abhängig von Art des Antriebs
- Antrieb der Pioneer-Roboter: differentieller Antrieb

- Annahme: Die aktuelle Position hängt **nur** von der vorhergehenden Position und den gemessenen Ticks ab

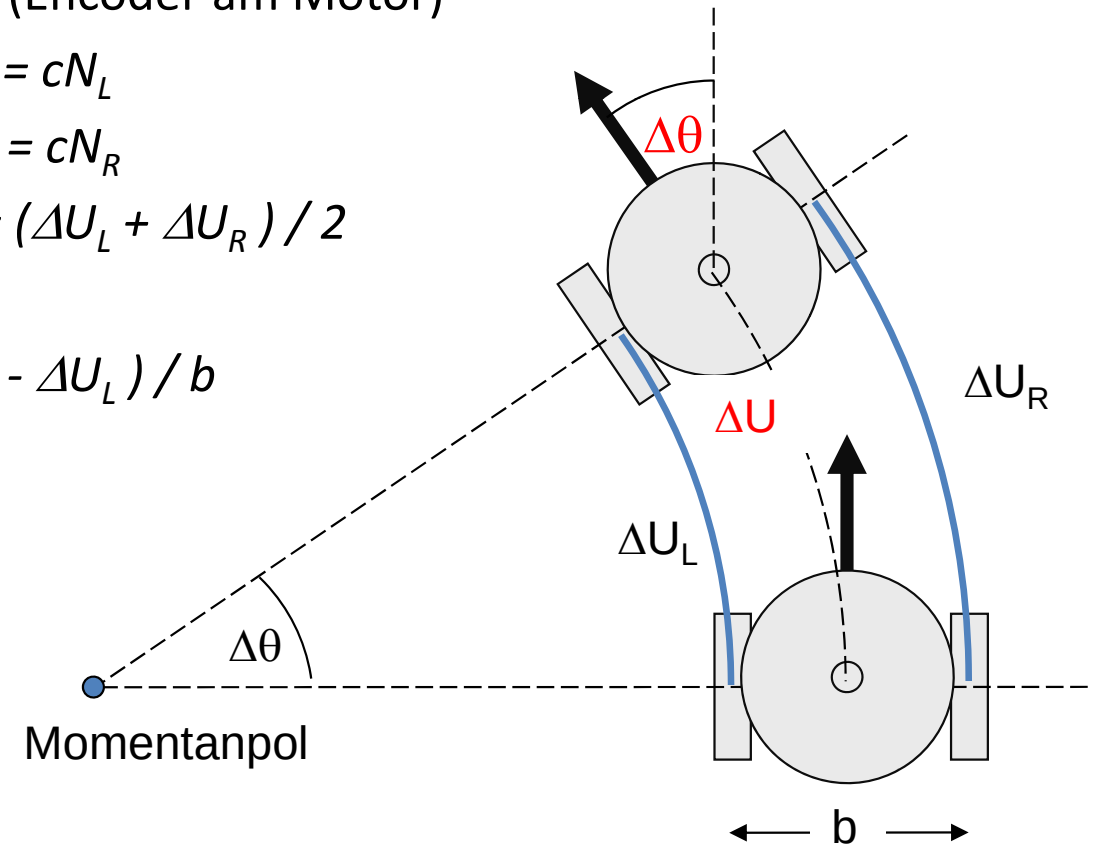
Abhängigkeitsgraph:



Differentieller Antrieb

$N_L, N_R \rightarrow \Delta U, \Delta\theta$

- $c = \pi D / Cn$ = Umrechnungsfaktor Ticks in Weg
  - D = Raddurchmesser
  - C = Encoderauflösung [Ticks pro Umdrehung]
  - n = Getriebeuntersetzung (Encoder am Motor)
- Weg des linken Rades $\Delta U_L = cN_L$
- Weg des rechten Rades $\Delta U_R = cN_R$
- Bogenlänge Mittelpunkt $\Delta U = (\Delta U_L + \Delta U_R) / 2$
- b = Radabstand
- Richtungsänderung $\Delta\theta = (\Delta U_R - \Delta U_L) / b$

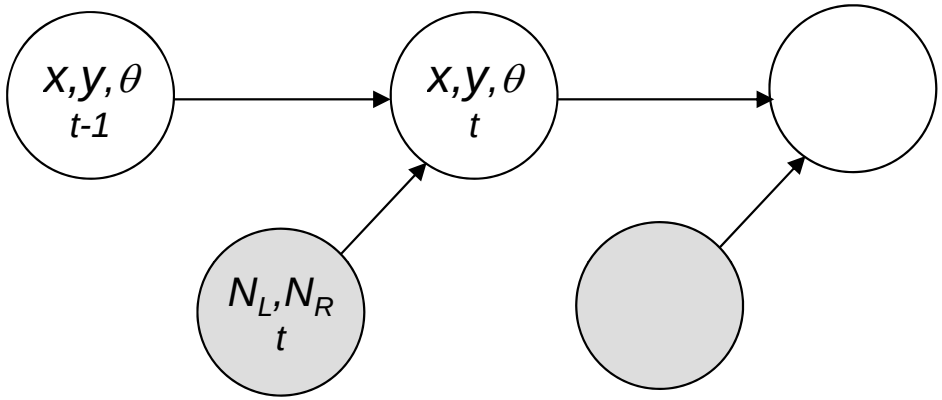


Differentieller Antrieb

$\Delta U, \Delta\theta \rightarrow x, y, \theta$

- Berechnung der **neuen Position $(x,y,\theta)_t$** aus der alten in **sehr kleinen** Zeitintervallen anhand der Bewegung:

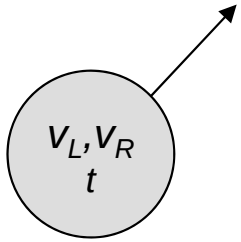
$$\theta_t = \theta_{t-1} + \Delta\theta_t$$
$$x_t = x_{t-1} + \Delta U_t \cos(\theta_{t-1})$$
$$y_t = y_{t-1} + \Delta U_t \sin(\theta_{t-1})$$



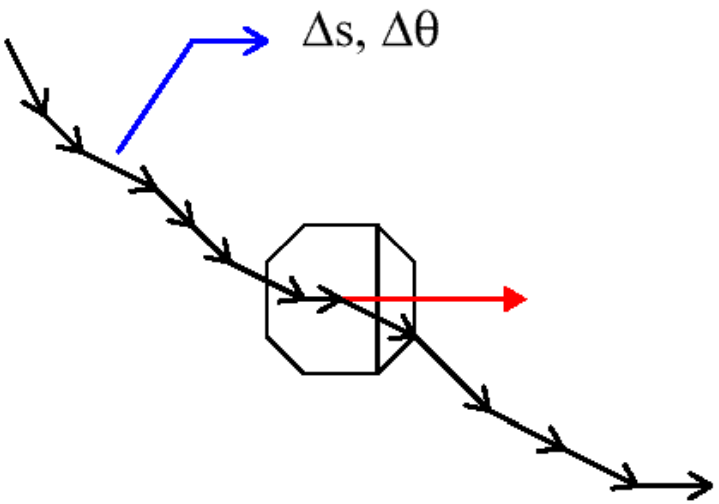
- Nur **Näherung** der tatsächlichen Bewegung
  - Annahme konstanter Radgeschwindigkeiten im Intervall [t-1,t]
  - Bogen ΔU als Gerade in Richtung θ_t

  → Intervalle müssen sehr klein sein!

- auch möglich: statt Ticks die Bewegungsbefehle verwenden



Odometriegleichungen nach Konolige



$$x' = x + \Delta s \cos\theta$$
$$y' = y + \Delta s \sin\theta$$
$$\theta' = \theta + \Delta\theta$$

Figure 2.3 Incremental position information integrated to give global position

$$\Delta\theta = \frac{\Delta r - \Delta l}{d}$$
$$\Delta s = \frac{\Delta r + \Delta l}{2}$$

Equation 2.1 Small wheel movements of a differential-drive base

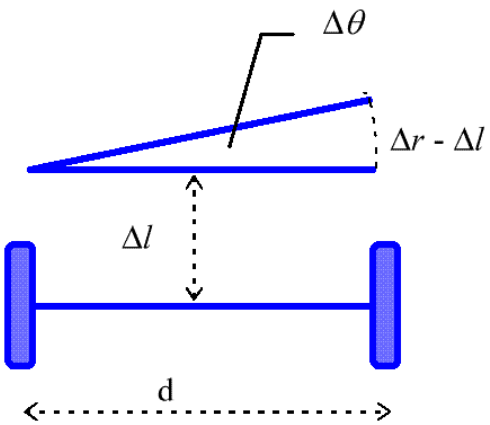
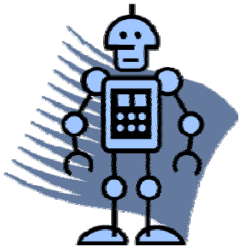


Figure 2.2 Incremental motion of the robot over a 20 ms period

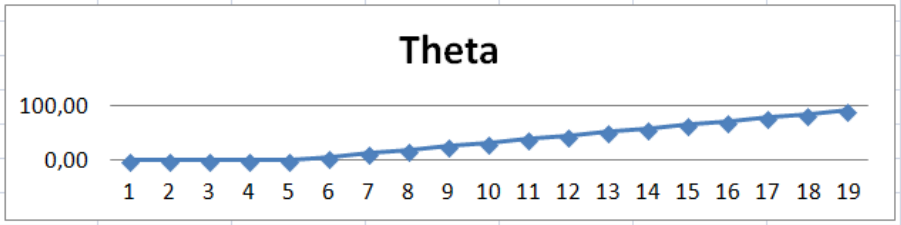
[Kon01] K. Konolige. Robot motion: probabilistic model; sampling and Gaussian implementations; Markov localization. Technical report, AI Center, SRI International, 2001.

Bewegungsmodell

Differentieller Antrieb

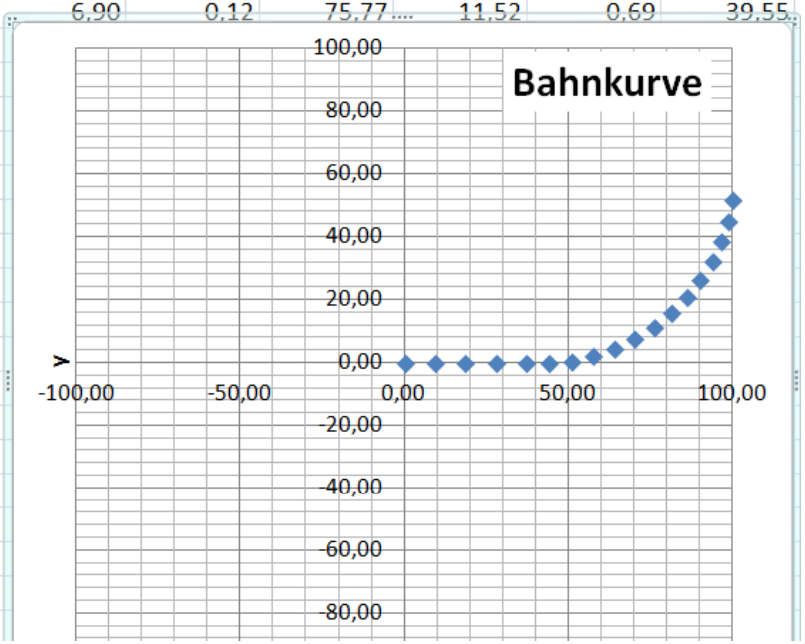


Raddurchmesser D	120	mm
Encoderauflösung C	4096	Ticks pro Umdrehung
Getriebeuntersetzung n	1	
Umrechungsfaktor c	0,092039	mm pro Tick
Radabstand b	200	mm
Eingabe		



c = 2

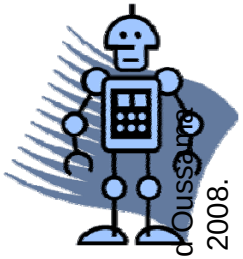
x	y	theta in rad	theta in °	NL	NR	delta_UL	delta_UR	delta_U	delta_theta	x	y	theta in rad	theta in °
0,00	0,00	0,00	0,00	100,00	100,00	9,20	9,20	9,20	0,00	9,20	0,00	0,00	0,00
9,20	0,00	0,00	0,00	100,00	100,00	9,20	9,20	9,20	0,00	18,41	0,00	0,00	0,00
18,41	0,00	0,00	0,00	100,00	100,00	9,20	9,20	9,20	0,00	27,61	0,00	0,00	0,00
27,61	0,00	0,00	0,00	100,00	100,00	9,20	9,20	9,20	0,00	36,82	0,00	0,00	0,00
36,82	0,00	0,00	0,00	**-50,00**	**200,00**	-4,60	18,41	6,90	0,12	43,72	0,00	0,12	6,59
43,72	0,00	0,12	6,59	-50,00	200,00	-4,60	18,41	6,90	0,12	50,58	0,79	0,23	13,18
50,58	0,79	0,23	13,18	-50,00	200,00	-4,60	18,41	6,90	0,12	57,30	2,37	0,35	19,78
57,30	2,37	0,35	19,78	-50,00	200,00	-4,60	18,41	6,90	0,12	63,79	4,70	0,46	26,37
63,79	4,70	0,46	26,37	-50,00	200,00	-4,60	18,41	6,90	0,12	69,98	7,77	0,58	32,96
69,98	7,77	0,58	32,96	-50,00	200,00	-4,60	18,41	6,90	0,12	75,77	11,52	0,69	39,55
75,77	11,52	0,69	39,55	-50,00	200,00	-4,60	18,41						
81,09	15,92	0,81	46,14	-50,00	200,00	-4,60	18,41						
85,87	20,90	0,92	52,73	-50,00	200,00	-4,60	18,41						
90,05	26,39	1,04	59,33	-50,00	200,00	-4,60	18,41						
93,58	32,33	1,15	65,92	-50,00	200,00	-4,60	18,41						
96,39	38,63	1,27	72,51	-50,00	200,00	-4,60	18,41						
98,47	45,21	1,38	79,10	-50,00	200,00	-4,60	18,41						
99,77	51,99	1,50	85,69	-50,00	200,00	-4,60	18,41						
100,29	58,87	1,61	92,29	-50,00	200,00	-4,60	18,41						



Position

Ticks

Relative Positionierung genügt nicht



„Bei bekanntem Bewegungsmodell und der Information über die Bewegung, sollten wir eigentlich in der Lage sein, die Position des Roboters jederzeit genau angeben zu können, ...

ABER ...“

3 Wie sicher ist die neue Position?

„Given such a model and complete knowledge of the control inputs, we should in principle, be able to estimate a robot‘s pose at any time. ... Unfortunately ...“ . Dudek, M. Jenkin, 2008

[DJ08] G. Dudek and M. Jenkin. Inertial Sensors, GPS, and Odometry. In Bruno Siciliano and Oussama Khatib, editors, Springer Handbook of Robotics, pages 477-490. Springer Berlin Heidelberg, 2008.

Fehlerquellen in der Odometrie

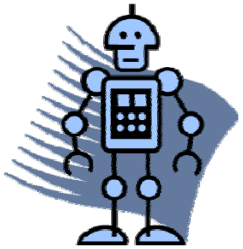
Systematische Fehler (d.h. der Mittelwert der Fehler ist ungleich 0, kalibrierbar!)

- Unequal wheel diameters
- Average of actual wheel diameters differs from nominal wheel diameter
- Actual wheelbase differs from nominal wheelbase
- Misalignment of wheels
- Finite encoder resolution
- Finite encoder sampling rate

Unsystematische Fehler durch eine stochastische Welt

- Travel over uneven floors or unexpected objects
- Wheel-slippage due to:
  - slippery floors
  - overacceleration
  - fast turning (skidding)
  - external forces (interaction with external bodies)
  - internal forces (castor wheels)
- non-point wheel contact with the floor

Modellierung der Fehler



Fehler (Bewegungsunschärfe) wird modelliert für:

1) Probabilistische Lokalisierungsverfahren (Partikelwolke, Vorhersage möglicher Positionen nach einer Bewegung), 2) Simulator

Drei Fehlerarten

- Entfernungsfehler: Fehler beim Zurücklegen einer geraden Strecke
- Drehfehler: Fehler beim Drehen des Roboters
- Driftfehler: Orientierungsfehler beim Zurücklegen einer geraden Strecke

-> TB

Das Bewegungsmodell in ARIA/ARNL bildet genau diese drei Fehlerarten ab:

KMmPerMm (0.2)	When the robot moves linearly, the error in distance is proportional to the distance moved. This error is is given as a fraction in mm per mm
KDegPerDeg (0.2)	When the robot rotates, the error in the angle is proportional to the angle turned. This is expressed as a fraction in degs per deg.
KDegPerMm (0.01)	When the robot moves linearly it can also affect its orientation. This drift can be expressed as a fraction in degs per mm.

Problem der relativen Positionierung

Wachsende Positionsunsicherheit durch

1) Addition neuer Positionsfehler beim Bewegen (stochastische Welt)

2) Vergrößerung bestehender Fehler beim Bewegen

- Beispiel: Auswirkung eines Richtungsfehlers **trotz fehlerloser Bewegung**

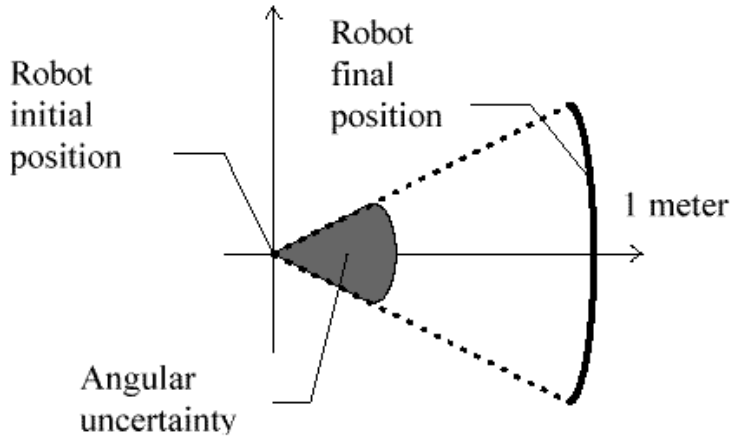


Figure 2.5 A robot's uncertainty in position grows even when it moves with no dead-reckoning error, as long as it has an uncertain orientation.

Unsicherheit wächst mit zunehmender Entfernung vom Startpunkt

[Kon01]

Zunahme der Positionsunsicherheit

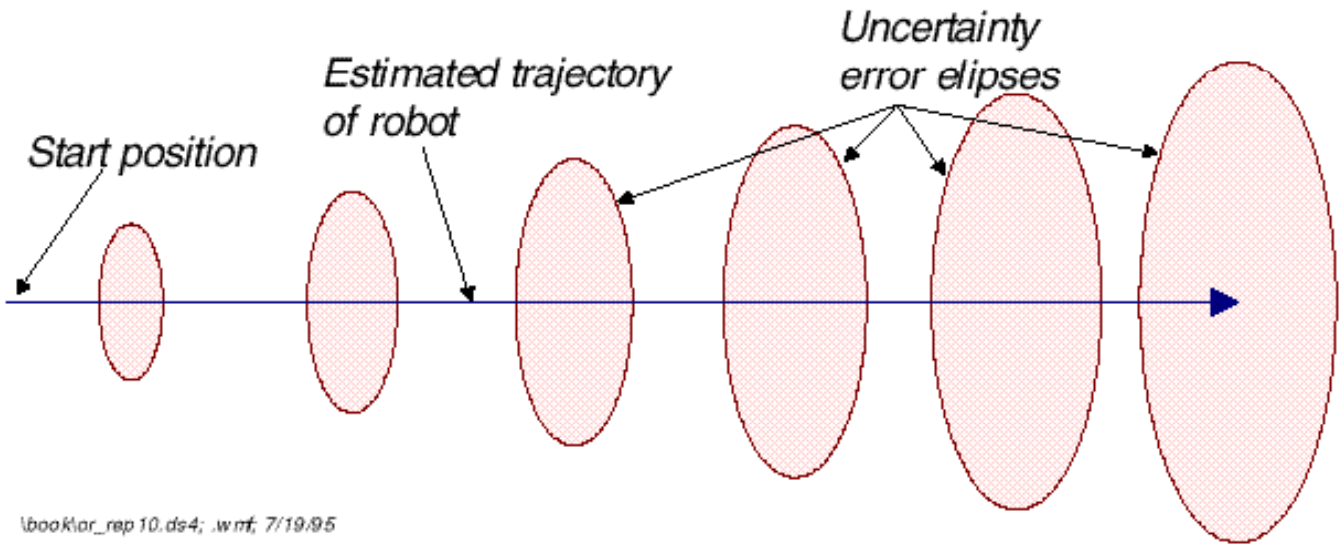
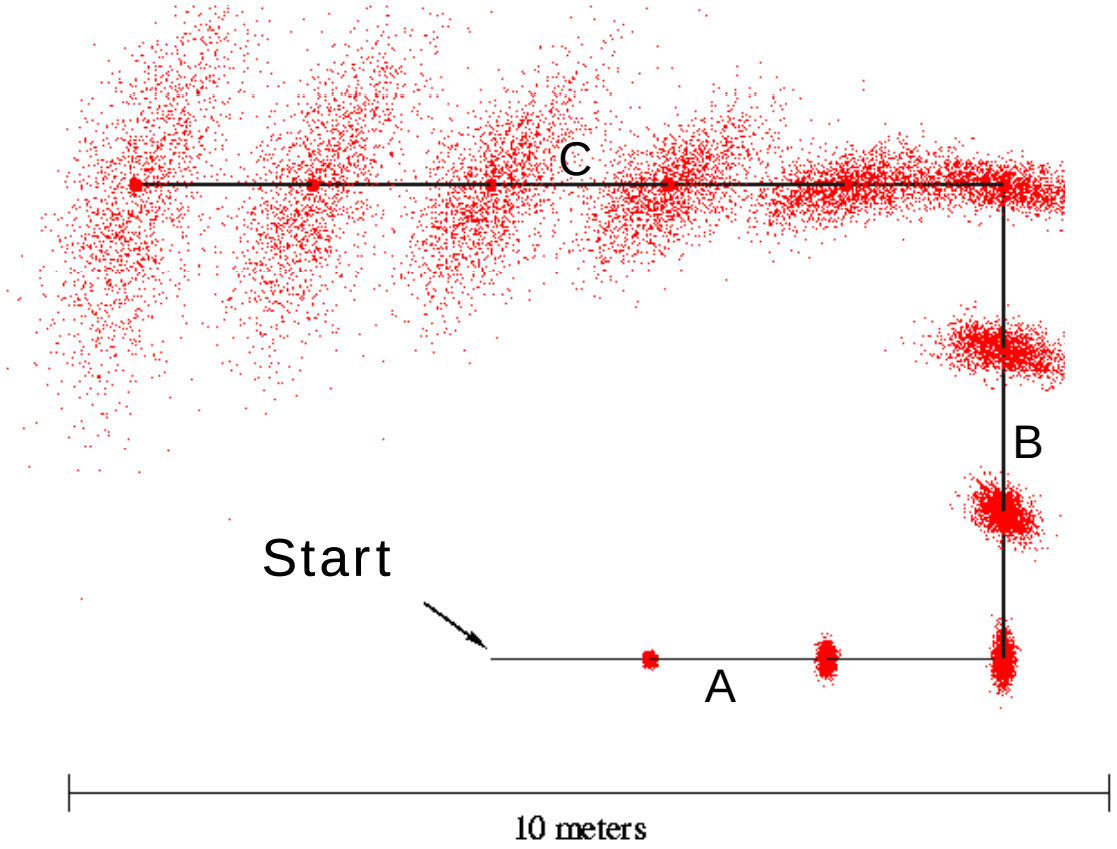


Figure 5.1: Growing "error ellipses" indicate the growing position uncertainty with odometry. (Adapted from [Tonouchi et al., 1994].)

[BEF96]

Zunahme der Positionsunsicherheit bei einer Fahrt ohne Umweltsensorik

Figure 5.10 **Sampling approximation** of the **position belief** for a **non-sensing robot**. The solid line displays the actions, and the samples represent the robot's belief at different points in time.



-> Übung

[Fig. 5.10]

[TBF05] S. Thrun, W. Burgard, and D. Fox. Probabilistic Robotics. MIT Press, 2005.

Beispiel für Auswirkung des Odometriefehlers bei der Kartenerstellung (Mapping)

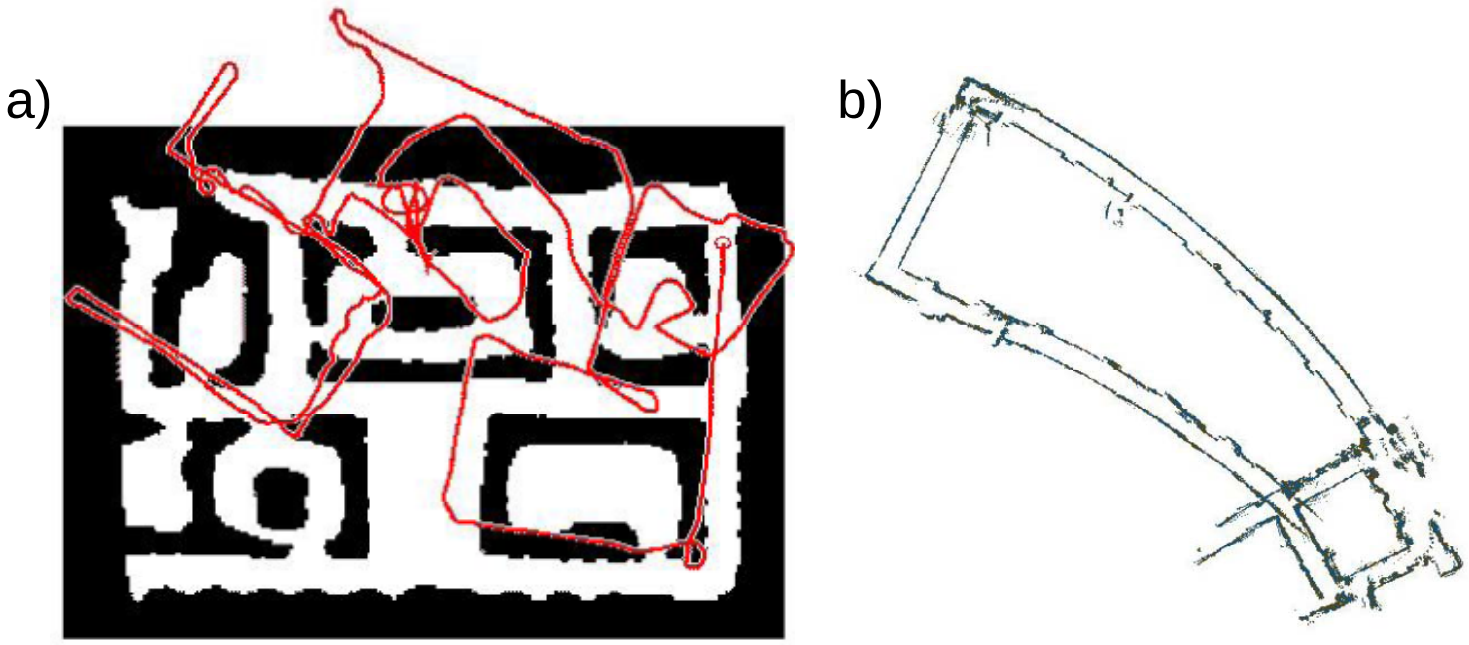


Figure 1: (a) Example of odometry error: Shown here is a robot's path as obtained by its odometry, relative to a given map. Small odometry (or control) errors can have **large effects on later position estimates**. (b) One of the many faces of the correspondence problem: Shown here is a robot traversing a cyclic environment during which it accrues significant odometry error. For building consistent maps the robot has to establish correspondence between the present and past positions, which is particularly difficult when closing a cycle. Establishing this correspondence is one of the most challenging problems in robotic mapping.

[Thr02] Sebastian Thrun. Robotic mapping: A survey. In Exploring Articial Intelligence in the New Millenium. Morgan Kaufmann, 2002.

1. Geeignete Odometrieräder

*... It is widely known that, ideally, wheels used for odometry should be **"knife-edge" thin** and **not compressible** ...*

[Johann Borenstein]

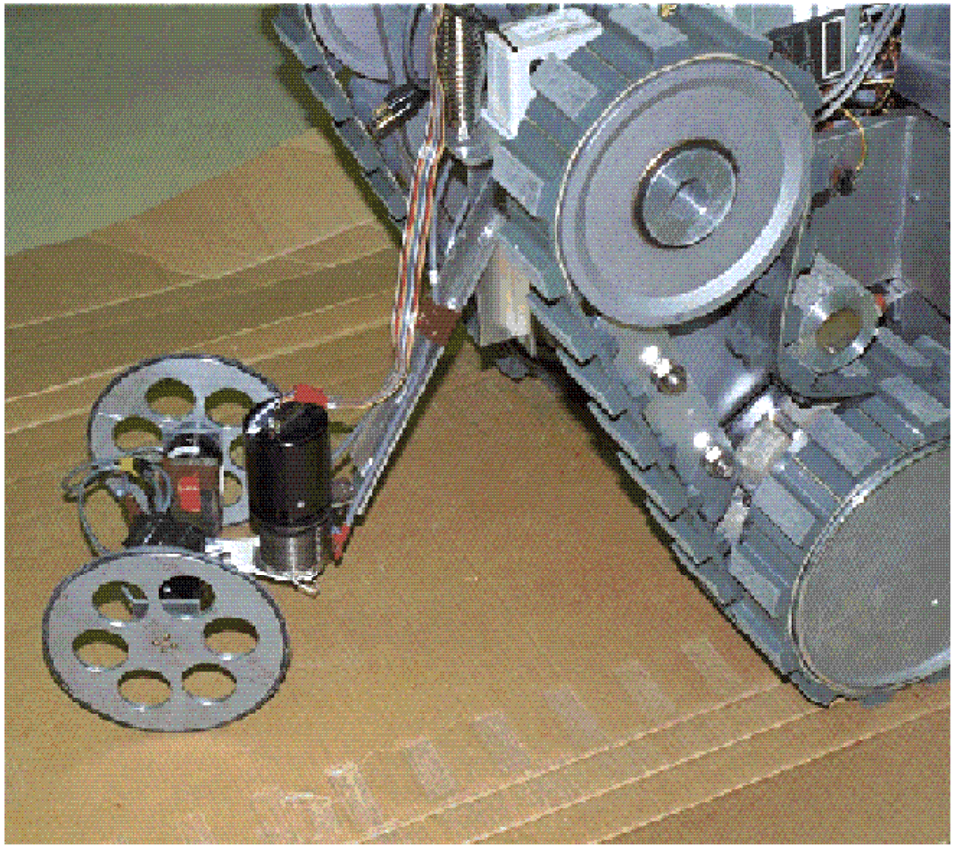


Figure 5.8: A simple encoder trailer. The trailer here was designed and built at the University of Michigan for use with the Remotec's *Andros V* tracked vehicle. (Courtesy of The University of Michigan.)

[BEF96]

2. Kalibrierung

- Reduziert systematische Fehler
- Beispiel: Entfernungsfehler beim Pioneer
  - Mehrmalige Testfahren mit Odometrie- und manueller Wegmessung
  - Stichprobe → fährt im Mittel x% zu weit oder zu kurz
  - Anpassung des Umrechnungsfaktors c (c * Ticks = Weglänge)
  - DistConvFactor in Parameterfiles *.p

3. Verwendung von Richtungssensoren

- Richtung durch <u>relative</u> Messung (Gyroskope, Winkelgeschwindigkeit)

oder besser

- Richtung durch <u>absolute</u> Messung

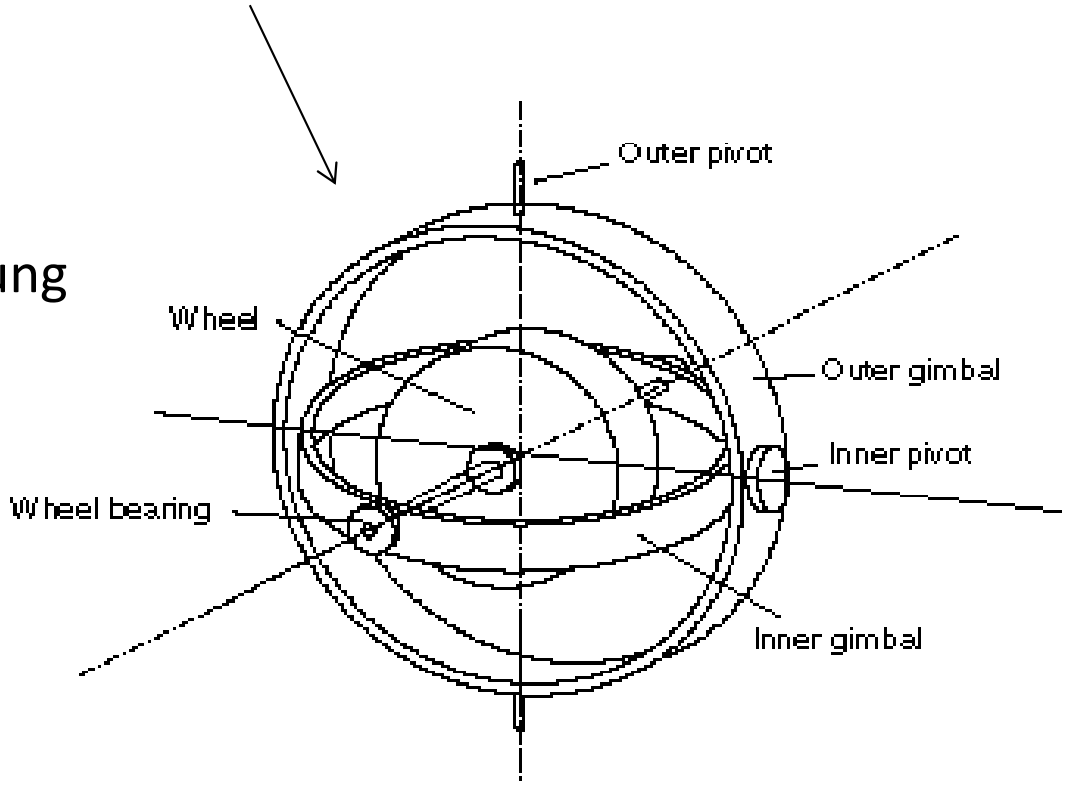
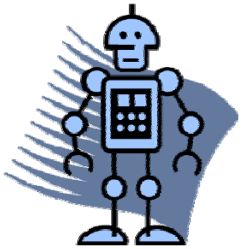


Abb.: 2-achsiges mechanisches Gyroskop, (Kreiselkompass)

[BEF96]

4. Abgleich mit Wahrnehmung



- Trotzdem: **Positionsunsicherheit wächst mit der Zeit**, deshalb

4. Beste Variante:

  - Ständiger Abgleich der geschätzten Position mit aktueller Wahrnehmung der Umgebung
  - Absolute Postionierung, Landmarkennavigation (gleich)
  - Probabilistische Lokalisierung in ARNL/ARIA (gleich danach)

Weitere Formen der relativen Positionierung

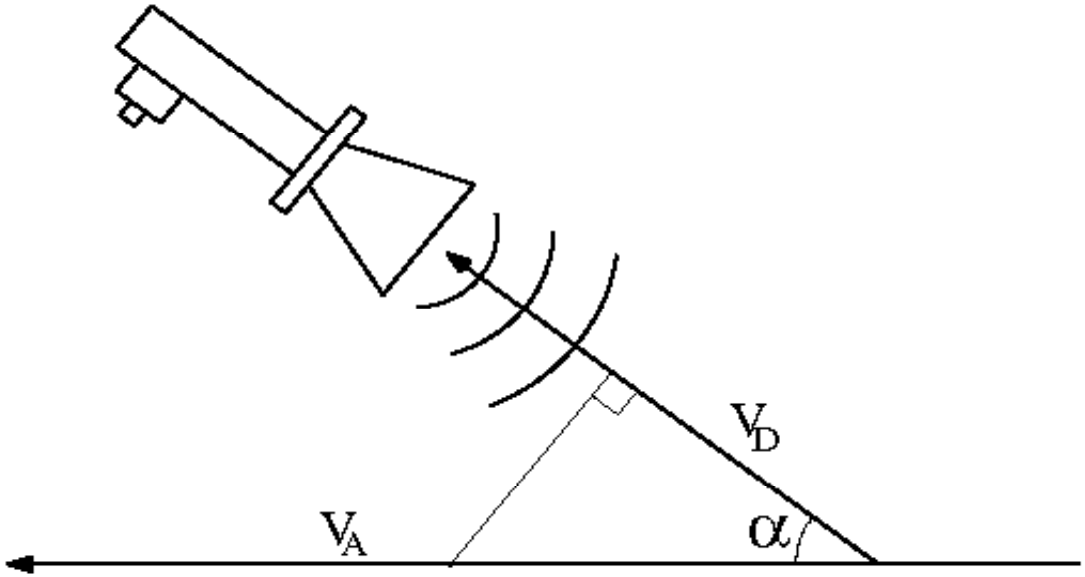
Neben der Odometrie gibt es andere Formen der relativen Positionierung:

Inertial Navigation Systems (INS) ==
Trägheitsnavigation

- Messung von Geschwindigkeit, Beschleunigung, Orientierung

- Beispiele:
  - ohne Räder:
    - Schiffe, Flugzeuge, Drohnen, Humanoide
  - Radbewegung misst kaum Positionsänderung:
    - Offroad, Sumpf

Doppler-Geschwindigkeitssensor



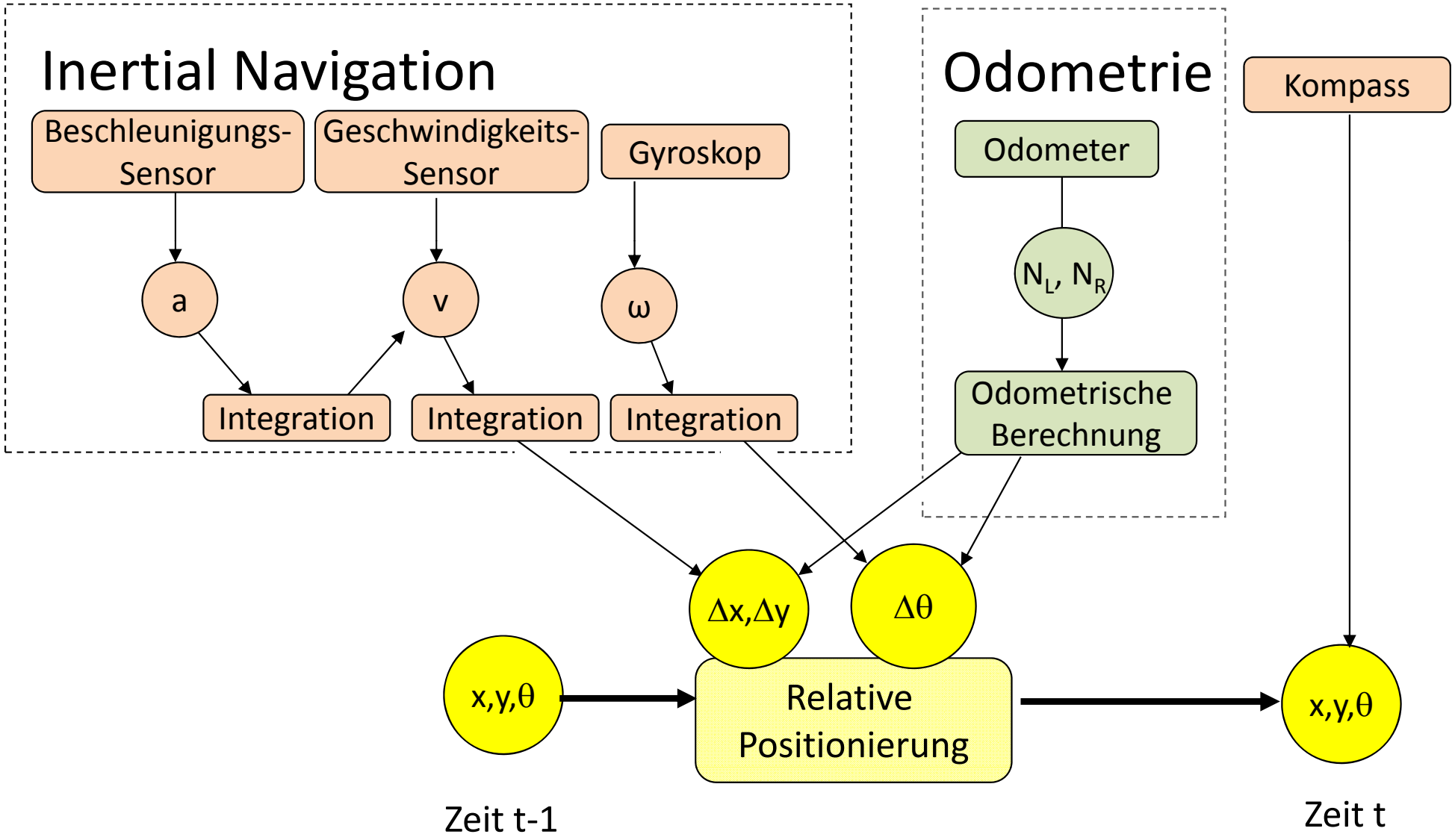
Der Sensor misst die Komponente v_D der Objektgeschwindigkeit v_A. Bei bekanntem Winkel alpha lässt sich daraus v_A bestimmen.

Beispiel: Dopplerradar (Blitzer)

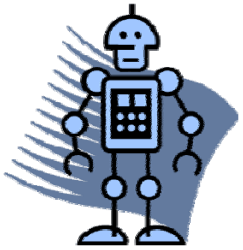
[BEF96]

Zusammenfassung

Relative Positionierung



Absolute Positionierung anhand von Landmarken



Grundidee:

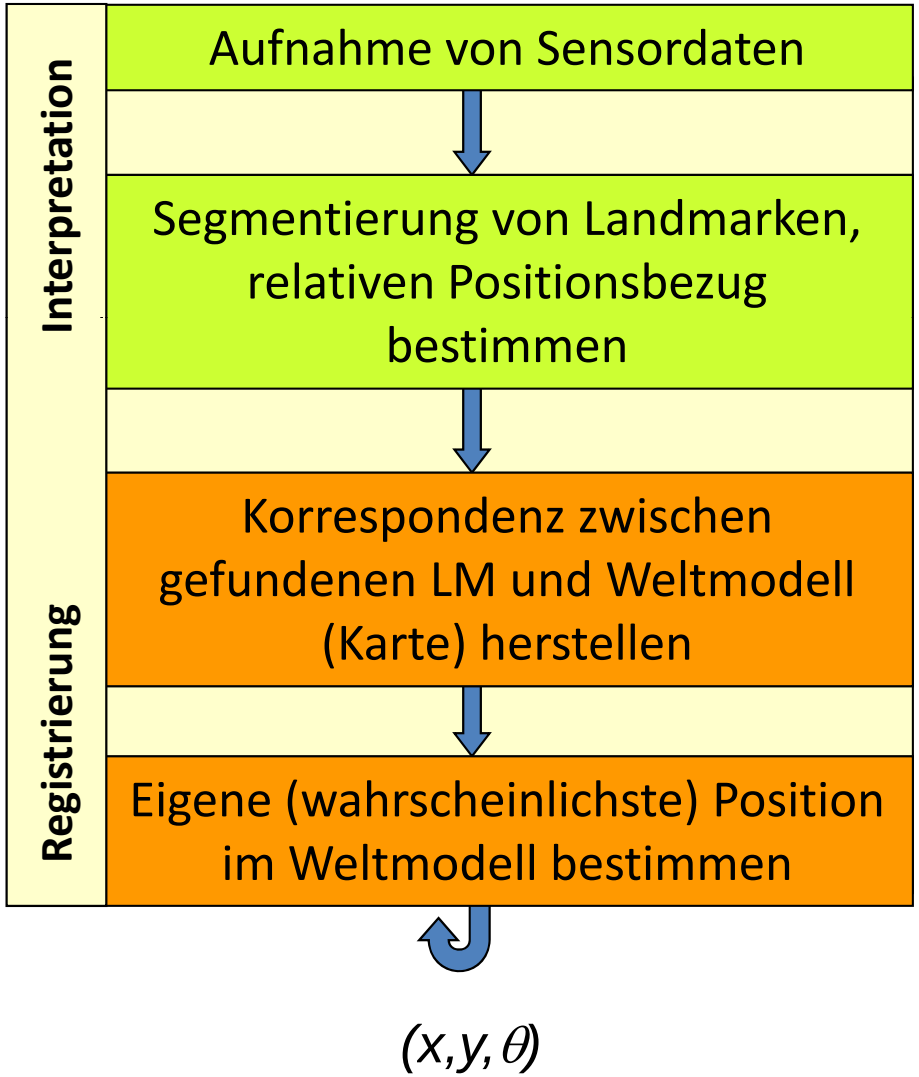
Positionsbestimmung anhand natürlicher oder künstlicher Landmarken mit bekannter Position

Definition Landmarke:

Ein Objekt/Merkmal mit folgenden Eigenschaften:

- Wahrnehmbar (Funktion der Sensorik, Blickwinkelabhängig)
- Informationsgehalt (Eindeutigkeit)
- Bekannte Position im Weltmodell
- Relative Position zum Roboter bestimmbar (bspw. Entfernung und Richtung zur Landmarke, Entfernungsdifferenz zweier Landmarken, Detektionsbereich eines RFID o.ä.)

Positionierung mittels Landmarken



- Modelle der Landmarken,
- oft komplex, bspw. Bildverarbeitung

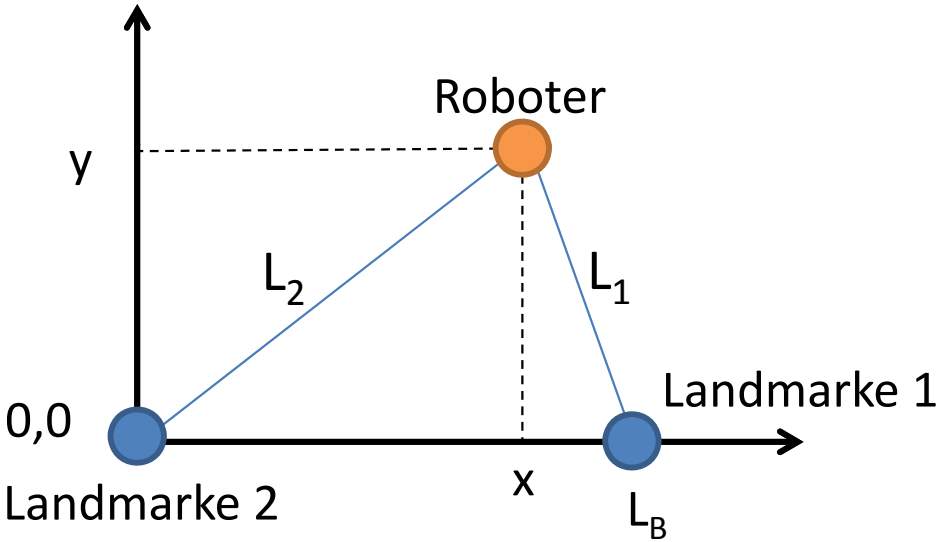
- einfach bei eindeutigen LM
- sonst komplex, bspw. Scan-Matching

- z.B. Trilateration, Triangulation

Trilateration

Positionsbestimmung durch **Entfernungsmessung** zu bekannten Landmarken

Zwei Marken



Es gilt nach Pythagoras:

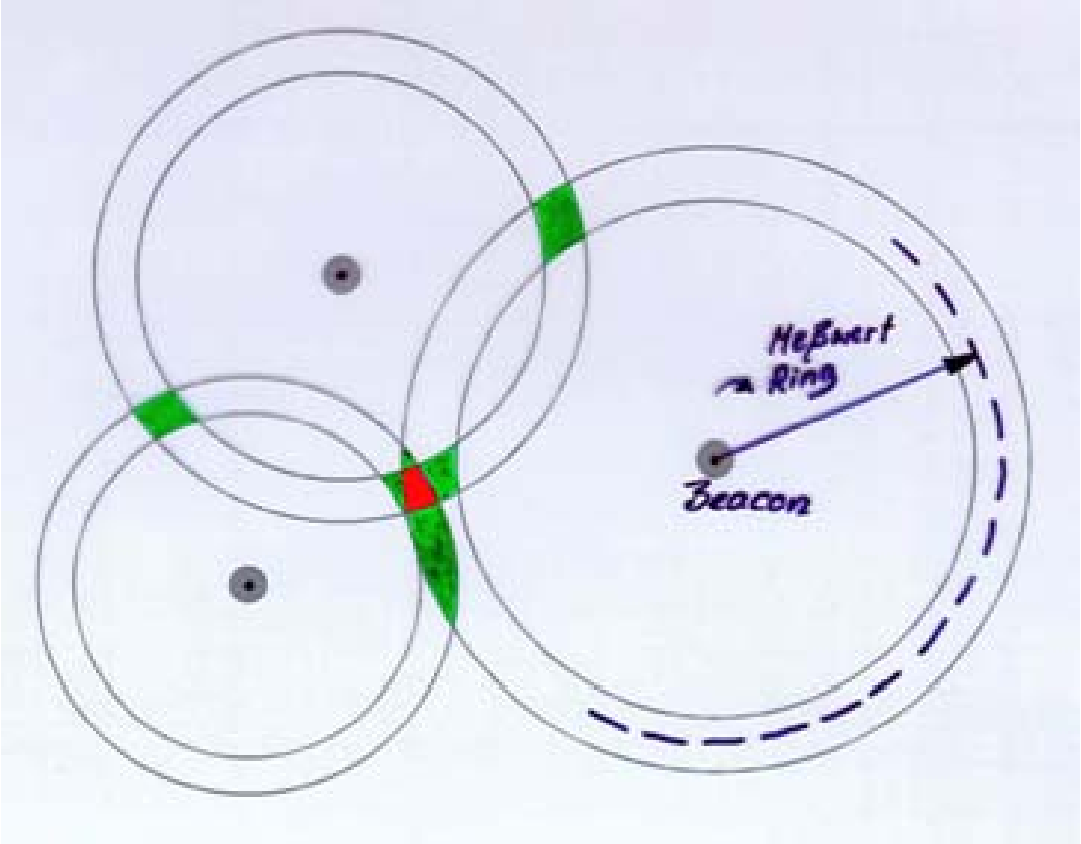
$$L_2^2 - x^2 = L_1^2 - (L_B - x)^2$$

$$\rightarrow x = (L_2^2 - L_1^2 + L_B^2) / (2L_B)$$

Mit bekanntem X zu Y:

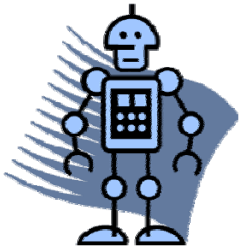
$$y = (L_2^2 - x^2)^{1/2}$$

Drei Marken



Ring durch unscharfe Messung

Arten von Landmarken



Natürliche Landmarken (1)

Ohne explizite Navigationsfunktion

Künstliche Landmarken

Für Navigationszwecke installiert

Passive (2)

- Visuelle Markierungen: Barcodes, Z-Codes
- RFID-Labels
- Flüchtige chemische Markierungen (=> Ameise, olfaktorischer Sensor)
- Magnetpillen

Aktive (sog. Baken oder Beacons) (3)

- Leuchttürme,
- Radiobeacons, Führungsstreifen,
- Induktionsschleifen,
- Laser, US-Baken, IR-Baken
- Thermale Markierungen
- GPS, Differential GPS

Natürliche Landmarken

1

- Bergspitzen, einzelne Bäume,
- **Wände**, Kanalabzweige, Bürotüren, Ecken, Säulen,
- Deckenbeleuchtung, Sterne, Sonne
- Magnetfeld, Flüsse, Strömungen und vieles mehr

- der ‚natürliche' Weg

- wichtigste Sensoren heute:
  - passiv: (Stereo)-Kamera
  - aktiv: Laserscanner, RGB-Depth-Sensor, Radar, (Sonarsensoren)

Scan-Matching

- Umwelt: Wände, Türen etc. – in der Regel nicht eindeutig
- Roboter mit Abstandssensoren: Sonarsensoren, Laserscanner ...
- **Scan-Matching: Finden einer Transformation** (Rotation, Translation) zweier Modelle, so dass sie maximal übereinstimmen.
- Ausrichten der Beobachtung an der Karte nach Korrelation
  → Roboterposition damit bekannt

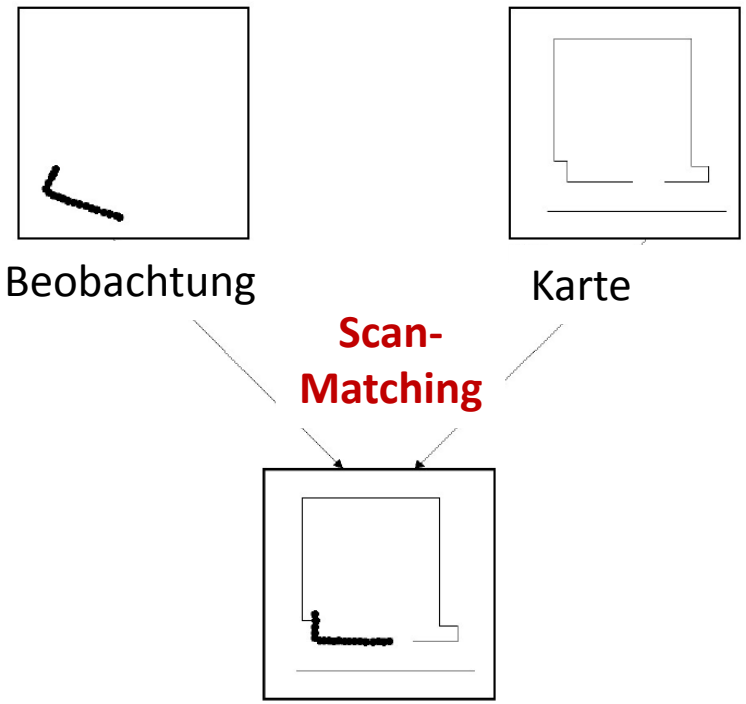


Abb. aus: Jörg Dressler. Navigation mobiler Systeme in Indoor-Umgebungen. Diplomarbeit FH Brandenburg, 2001

,

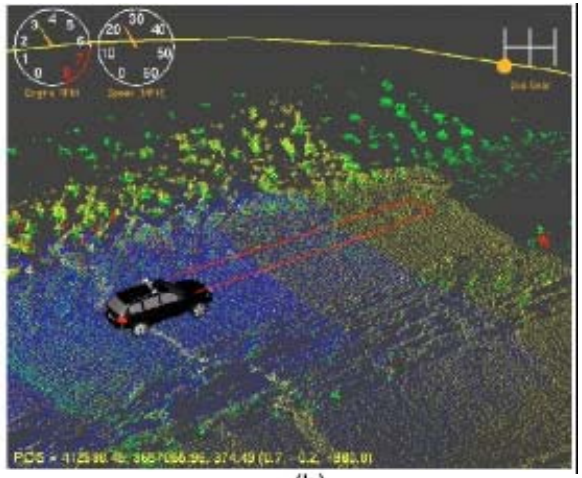
Laserscanner, SICK AG

- Beispiel: **LMS2xx** (auf dem Pioneer)
  - Pulslaufzeitmessung, **Time of Flight (TOF)**, Drehspiegel
  - 8 – 80m, 24V, 20W
  - RS232, RS422
  - Probleme im Gebäude: mehrere Lidar synchronisieren, Kantentreffer
  - Probleme im Freiland: Nebel, Kälte
  - ca. 4500 €
- Sick-Tochter ibeo z.B. **ALASCA XT** >44K€, **mit Objektverfolgung**

IBEO Lux

Stanley mit 5 Laserscannern

(a)



(b)

Bild: Stanley: The Robot that Won the DARPA Grand Challenge (2005)

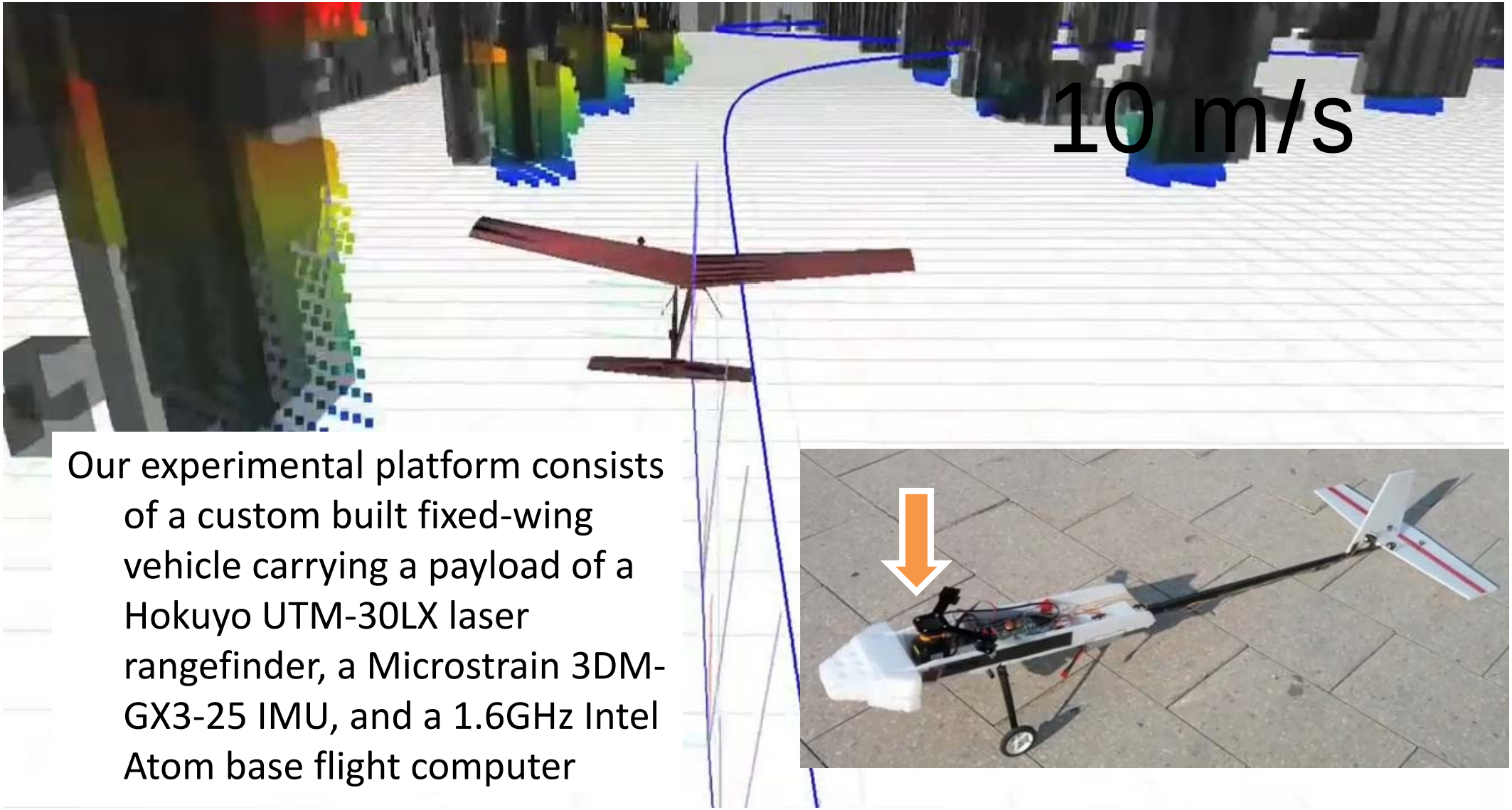
Laserscanner, Hokuyo

- Beispiel: Hokuyo **URG-04LX**
  - Klein, leicht, stromsparend, geringe Reichweite
  - ca. 2.300 €
  - 240°, Auflösung 0.36°
  - RS232, USB
  - 10 Hz
  - **4m**, 500mA@5V (2.5W)
- z.B. im Baumarktroboter Toomas
  - Projekt von MetraLabs GmbH, TU Ilmenau, Toom Baumarkt GmbH

- andere: **UTM-30LX**
  - **30m**, 270°

Toomas, Serviceroboter.
Foto: TU Ilmenau, 2007

Laserscanner, Hokuyo



Our experimental platform consists of a custom built fixed-wing vehicle carrying a payload of a Hokuyo UTM-30LX laser rangefinder, a Microstrain 3DM-GX3-25 IMU, and a 1.6GHz Intel Atom base flight computer

A. Bry, A. Bachrach, N. Roy: State estimation for aggressive flight in GPS-denied environments using onboard sensing. ICRA 2012
Video: http://web.mit.edu/newsoffice/2012/autonomous-robotic-plane-flies-indoors-0810.html

Laserscanner, Velodyne

- Beispiel: High Definition Lidar **HDL-64E**
  - Rotierender Kopf mit **64 Lasern**
  - TOF
  - **360 degree** field of view (azimuth)
  - 15 rpm
  - 0.09 degree angular resolution (azimuth)
  - 26.5 degree field of view (elevation)
  - 5 cm resolution (distance)
  - **50m**@10%, 120m@80% reflectivity
  - 12V, Weatherproof, IP65
  - 100 mbps Ethernet output, UDP
  - ~75K$
- Velodyne Acoustics, Inc. (USA, Kalifornien, Silicon Valley)

Bild: http://www.velodyne.com

Beispielbild HDL-64E

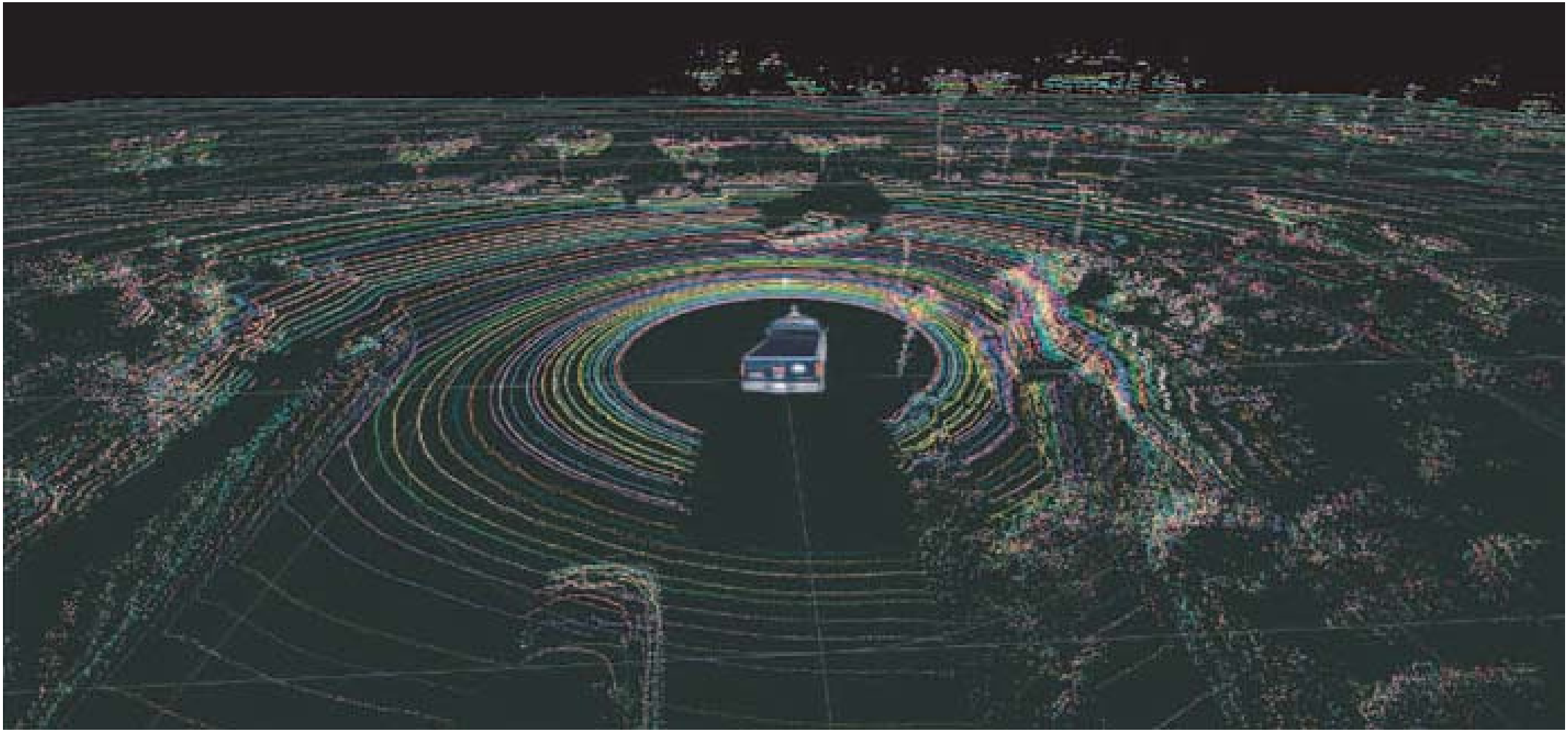


Bild: http://www.velodyne.com

Autonome Konvois aus verschiedenen Fahrzeugen

The U.S. Army Tank-Automotive Research, Development and Engineering Center (TARDEC) and Lockheed Martin have demonstrated the ability of fully autonomous convoys to operate in urban environments with multiple vehicles of different models. The demonstration earlier this month at Fort Hood, Texas, was part of the Army and Marine Corps' Autonomous Mobility Appliqué System (AMAS) program, and marked the completion of the program's Capabilities Advancement Demonstration (CAD).

Quelle: http://defense-update.com/20140201_amas.html

Autonomer Passat MadeInGermany

- BMBF-Projekt an der FU Berlin
- Raúl Rojas
- GPS, INS, Laser (6+1), Odometrie, Radar,
- Kamera (normal, Stereo, IR)
- Fahrbahnerkennung: Hella Aglaia
- Autonome Fahrt auf normalen Straßen Berlins: 09/2011

Foto: Claudia Heinstein

Laserscanner, Velodyne

- Velodyne Puck **VLP-16**
- günstig, leicht, 3D
- 16 Strahlen, 360°, +/-15° up/down
- **>100m**, Genauigkeit im cm-Bereich
- **Ø100mm** x 65mm, 600g, <10W
- ca. 6.200 €
- explizit geeignet für UAV, Drohnen
- VLP-16 auf dem Talon
  (milit. Roboter, QinetiQ)

HDL-64E, HDL-32E und Puck.

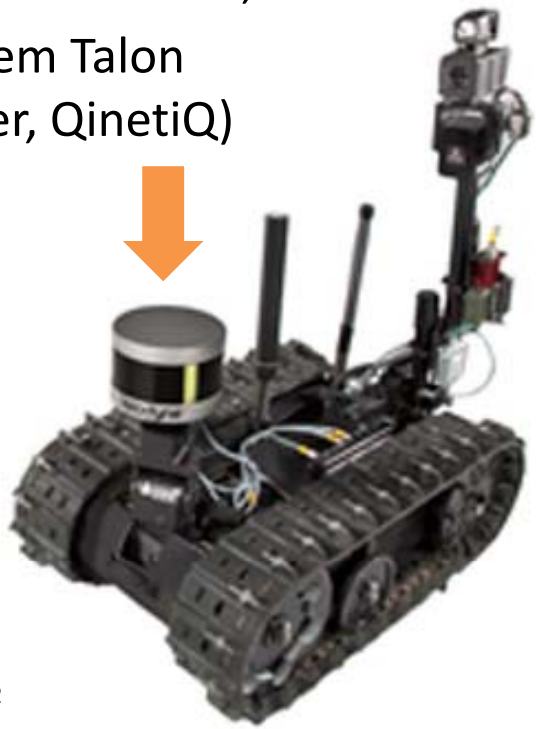


Bild: http://www.velodyne.com

Licht-Lokalisierung

- **Grenzen des Laserscanners:**
  - Glas, spiegelnde Objekte
  - Umgebungen (Lager, Museen, Krankenhäuser ...) mit vielen dynamischen Objekten, die nicht in der Karte verzeichnet sind

In Innenräumen möglich: Licht-Lokalisierung

  - Weitwinkel-Kamera Richtung Decke
  - dynamische Objekte um den Roboter stören nicht
  - keine extra Installation
  - Karte stabil
- NT: Licht nicht ausschalten,
  Blenden durch Sonne, zu wenig Leuchten

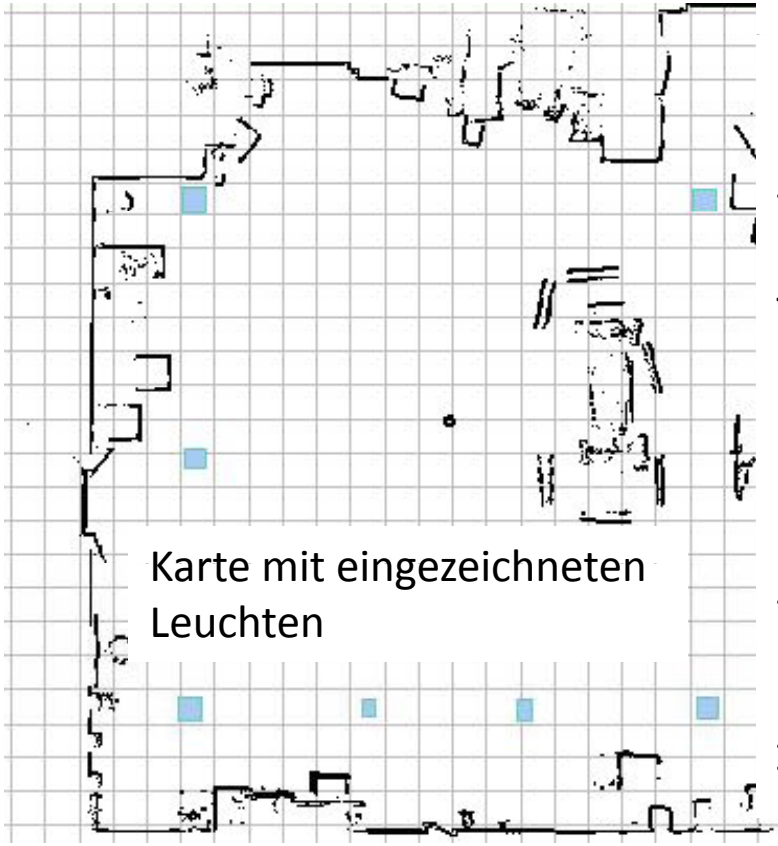
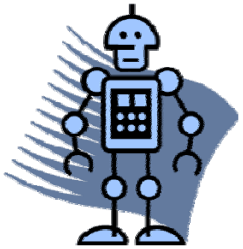


Abb. aus Adept Motivity User's Guide, 2013

Absolute Positionierung



- Natürliche Landmarken ✓
- **Passive künstliche Marken:**
- Aktive künstliche Marken (Beacon-Systeme)

NAMCO

Beispiel

- Landmarken: **Reflektoren**
- Roboter: Schwenkender Laser
- Ausgabe: Entfernung und Lage der LM

- Produkt: Lasernet-Sensor (NAMCO, $3,500-$5,050)

Quelle: NAMCO

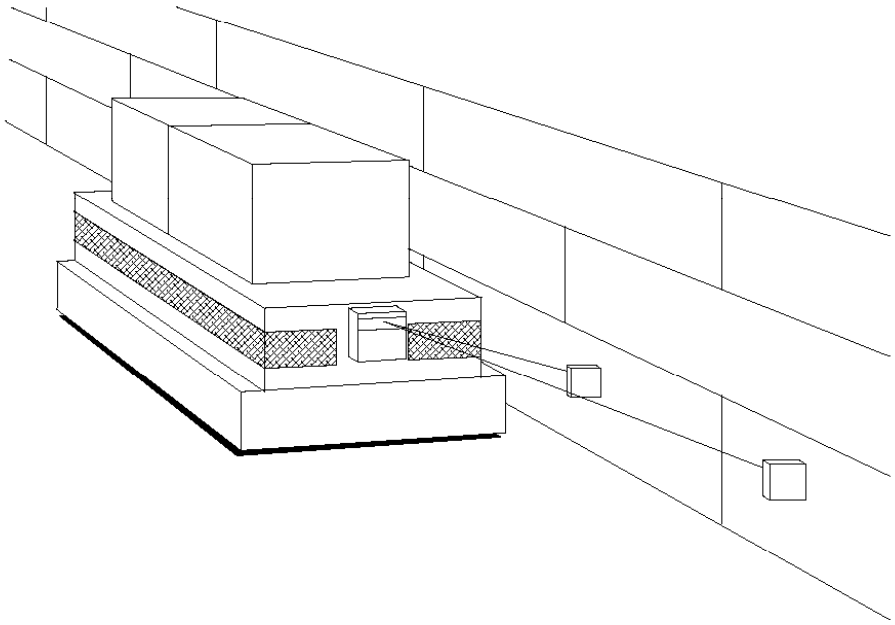


Figure 6.10: The LASERNET system can be used with projecting wall-mounted targets to guide an AGV at a predetermined offset distance. (Courtesy of NAMCO Controls.)

[BEF96]

NAV200

- NAV200 Laser-Positioniersystem zur Navigationsunterstützung
- Positionsbestimmung anhand ortsfester **Reflektoren** bekannter Position
- Messung der Winkel und Abstände zu den Reflektoren
- Drei erkannte Reflektoren zur Positionsermittlung ausreichend.
- Für fahrerlose Transportfahrzeuge (FTF)

- 360° Laserscanner
- Reichweite 28,5m

- Positioniergenauigkeit:4 ... 25 mm
- Winkelgenauigkeit:0,1 °
- ~8K€

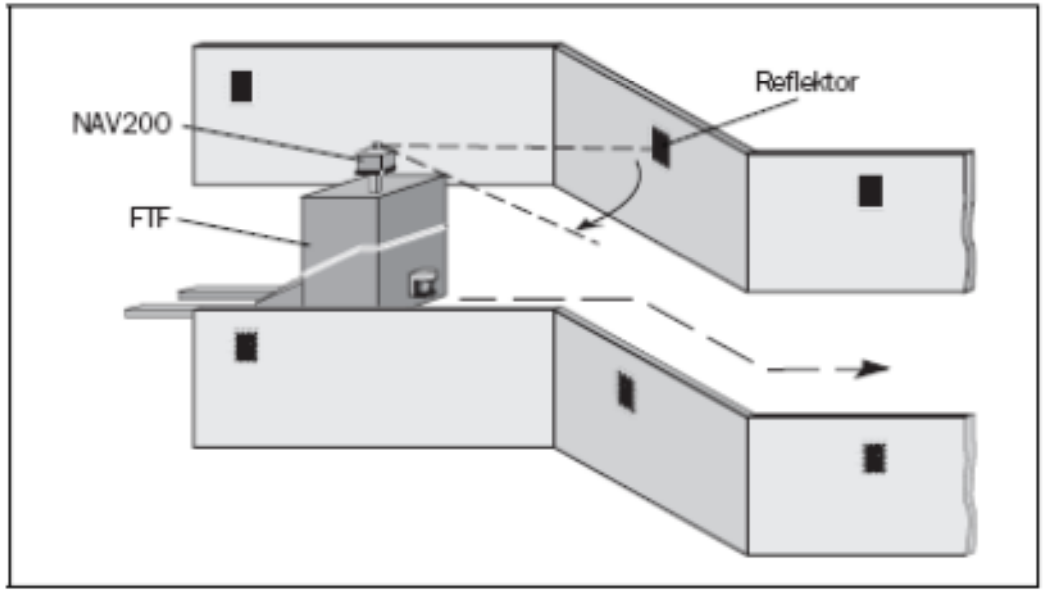


Bild: Betriebsanleitung NAV200, Sick

LaserNav

Beispiel

- Landmarken: **Barcodes**
- Roboter: Rotierender Laser
- Produkt: LaserNav

(Denning Branch International Robotics)

- Ausgabe: Position
- Reichweite: Reflektor 30m, Funkmarken 183 m

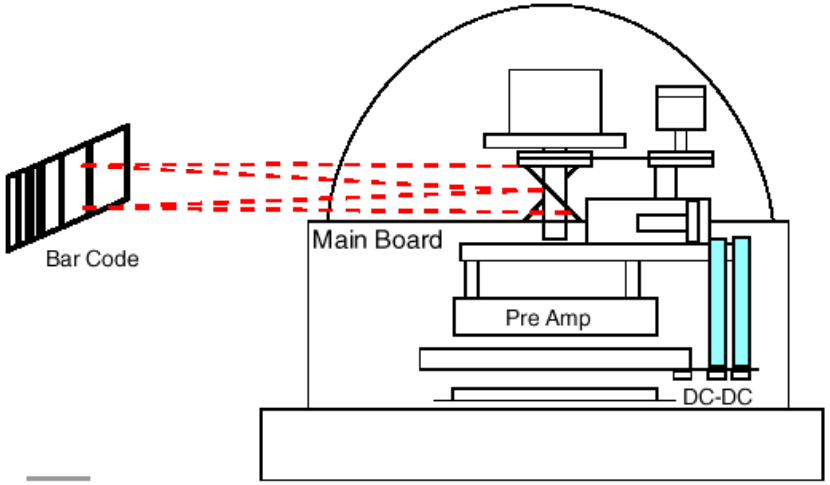


Figure 6.14: Schematics of the Denning Branch International Robotics *LaserNav* laser-based scanning beacon system. (Courtesy of Denning Branch International Robotics.)

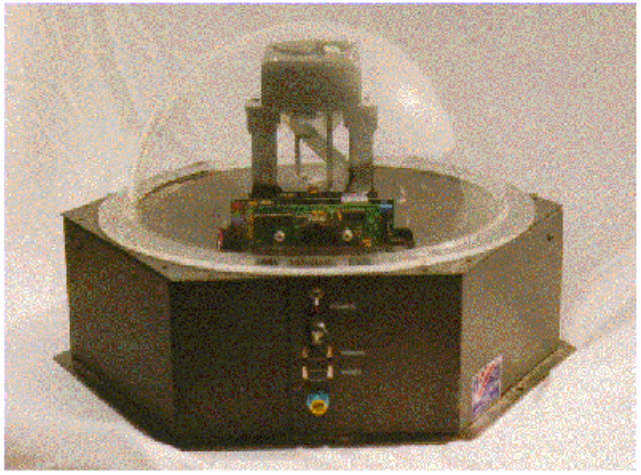


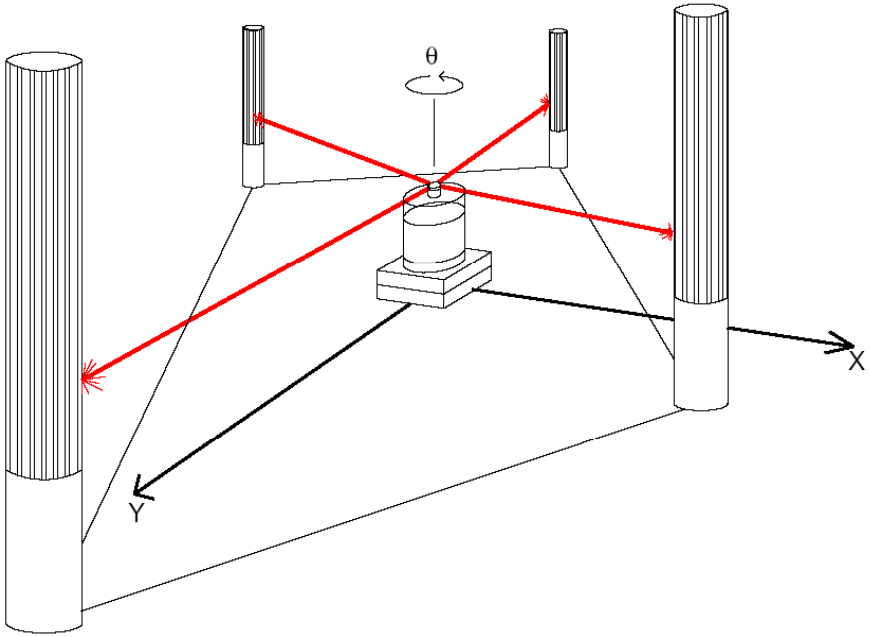
Figure 6.15: Denning Branch International Robotics (DBIR) can see *active targets* at up to 183 meters (600 ft) away. It can identify up to 32 active or passive targets. (Courtesy of Denning Branch International Robotics.)

[BEF96]

Beacon Navigation System

Beispiel

- Umwelt: **reflektierende Säulen**
- Roboter: rotierender Laser
- Hersteller: TRC, Transitions Research Corporation, später Helpmate Robotics Inc.



⇒ Rotierender Laserstrahl erzeugt Reflexionsprofil

⇒ Winkel

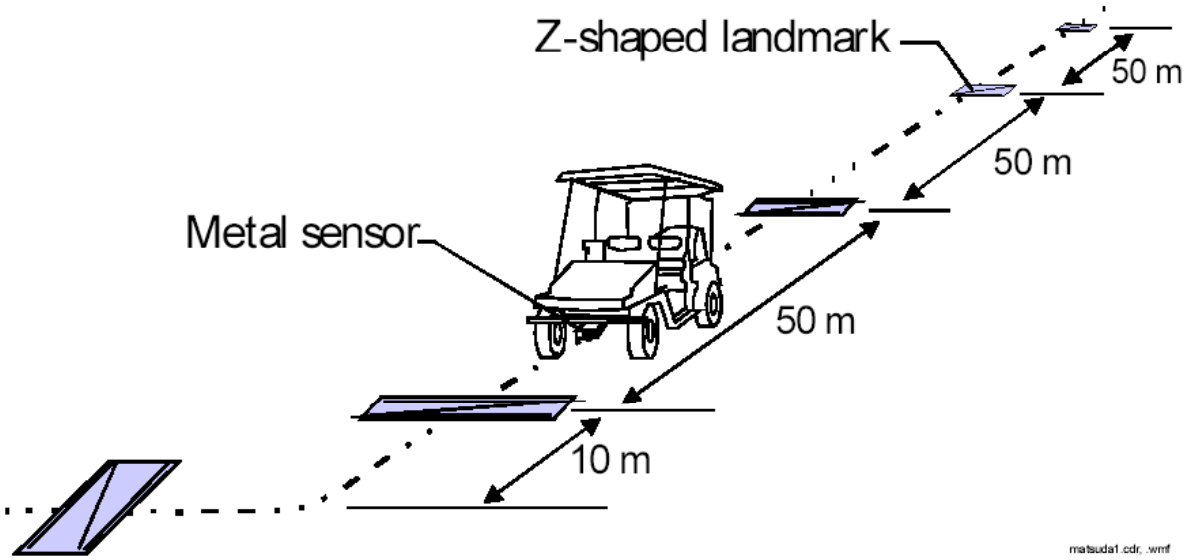
⇒ Positionsschätzung mit **Triangulation**

[BEF96]

Komatsu

Beispiel

- Umwelt: Gummimatten mit Al-Streifen
- Roboter: Metall-Sensor
- Firma: Komatsu Ltd.



[MY89] T. Matsuda and E. Yoshikawa, Z-shaped position correcting landmark for agvs, Proceedings of the 28th SICE Annual Conference, 1989.

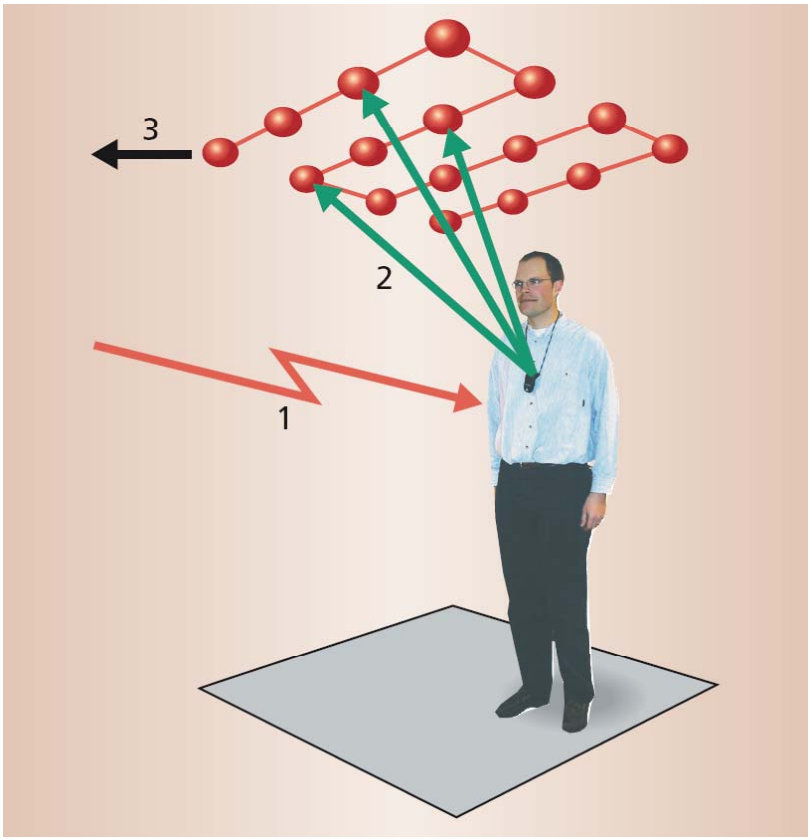
The Bat

- Tragbare Ultraschallsender (Active Bat)

1. Funksignal aktiviert Bat
2. Bat sendet einen Ultraschall-Impuls, in der Decke montierte Empfänger messen die Laufzeit
3. und senden sie zum Server
4. Server bestimmt mit **Trilateration** die Bat-Position

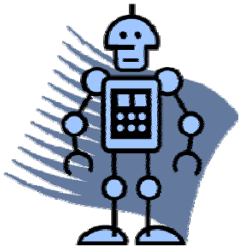
- indoor
- 95% +/-3cm
- 150 Abfragen pro Sekunde,
- 3 Bats gleichzeitig

- Weitere passive Marken??



Cambridge University Computer Laboratory / AT&T: The Bat Ultrasonic Location System, http://www.cl.cam.ac.uk/Research/DTG/attarchive/bat/ ; Stajano, Security for Ubiquitous Computing, 2002, S. 37-40.

Absolute Positionierung



- Natürliche Landmarken ✓
- Passive künstliche Marken: ✓
- **Aktive künstliche Marken (Beacon-Systeme)**

Aktive künstliche Marken

Beispiel

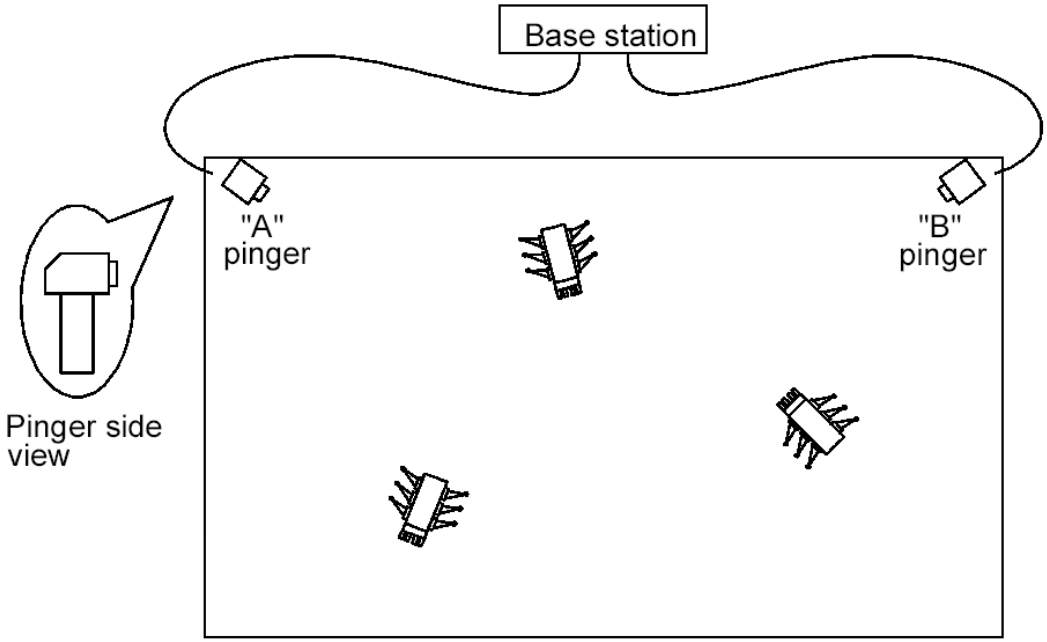
- ISR 2-D Location System für Genghis-Roboter

MIT Genghis

http://www.ai.mit.edu/projects/genghis/genghis.html

ISR 2-D Location System

- IS Robotics (MIT-Ableger)
- Billiges Sonartrilateration-System für MIT Genghis-Roboter
- 10K$
- Genauigkeit im 9x9m-Feld: 12mm
- 1 Basisstation, 2 Beacons (Pinger)



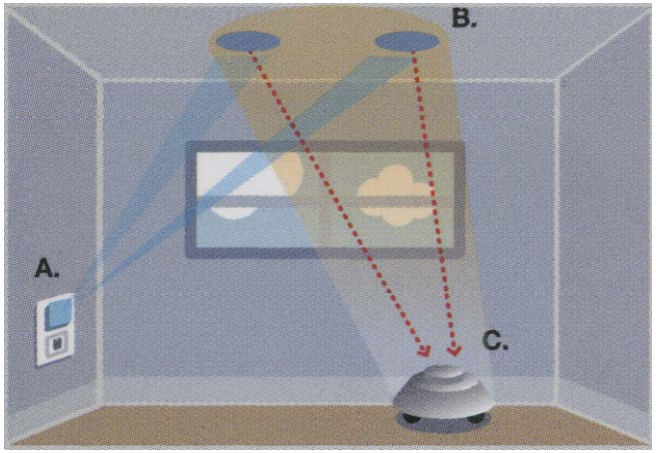
- kegelförmiger Reflektor über den Ultraschallsensoren => omnidirektionale Sensitivität
- Lösung für Zeitsynchronisation:
  Blitz und Donner-Prinzip:
  Senden eines elektromagnetischen Signals (hier Funksignal) <u>zeitgleich</u> mit einer Schallwelle

Northstar (Evolution Robotics, D)

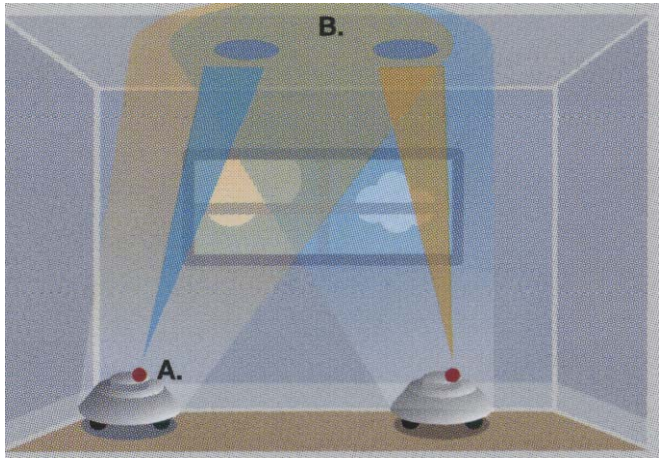
- An die Decke projizierte IR-Punkte (20 verschiedene)
- Triangulation, 10 Hz
- Detektor 3x4x1cm, 12.5g, 0.7W, max. Entfernung zum Lichtpunkt 6m
- Positioniergenauigkeit:10 – 40 mm, Winkelgenauigkeit:1 - 2 °
- Projektor + Detektor ~1.2K€

im Rovio als TrueTrack

- Ortsfester Projektor
- Positionierung im Raum



Projektor auf den Robotern
relative Postion untereinander

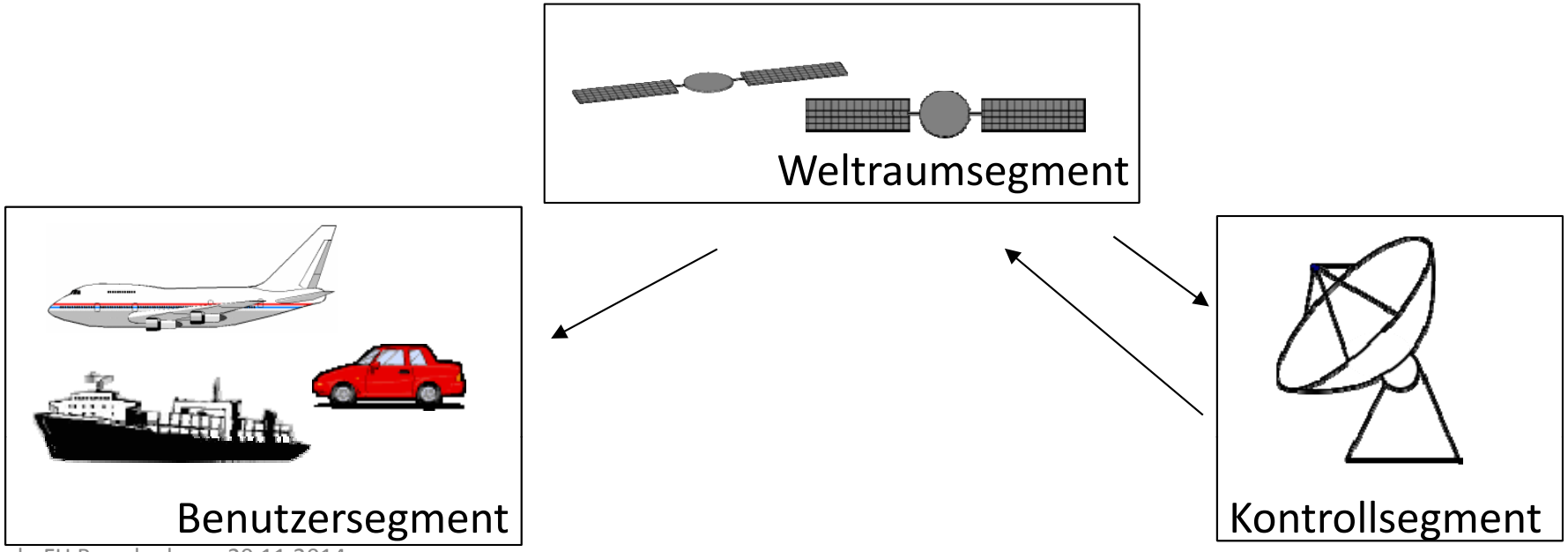


Bilder: www.evolution.com

- Sep 2012: iRobot kauft Evolution Robotics, $74 Million

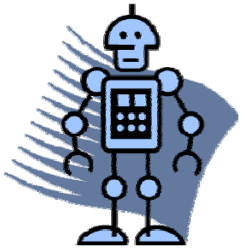
GPS – Global Positioning System

- 3 Segmente:
  - Weltraumsegment: 28 Satelliten, Höhe: 20 180 km
  - Kontrollsegment: Bodenstationen zur Überwachung des Systems
  - Benutzersegment: Anwender (zivil und militärisch)
- Laufzeit der Satelliten-Signale bis zum Empfänger -> Entfernung
- Position der Satelliten bekannt
- 4 Satelliten (einer für Zeitsynchronisation) reichen zur Positionsbestimmung aus



Bilder: www.evolution.com

Vor- und Nachteile von Beaconlösungen



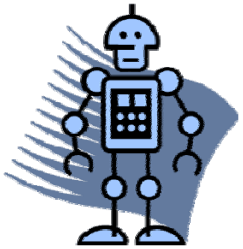
Positiv:

- Hohe Genauigkeit und Zuverlässigkeit
- Hohe Abfragerate
- Einfache Informationsverarbeitung

Negativ

- Umgebung muß modifiziert werden
- Hohe Wartungs- und Installationskosten
- Stromversorgung
- Direkte Sicht (zu mehreren Beacons) notwendig

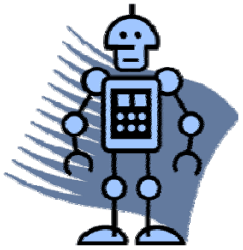
Absolute Positionierung



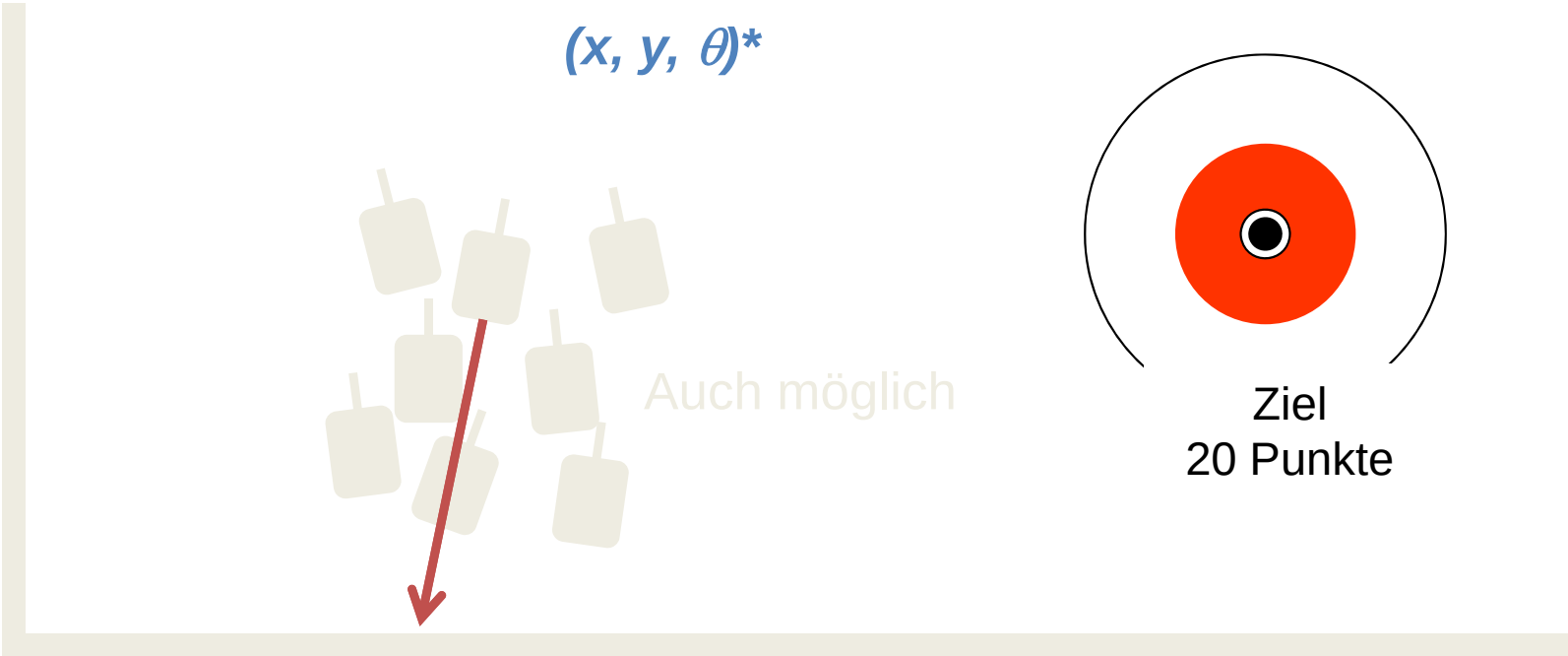
- Natürliche Landmarken ✓
- Passive künstliche Marken: ✓
- Aktive künstliche Marken (Beacon-Systeme) ✓

Absolute Positionierung ohne Sensoren

Absolute Positionierung ohne Sensoren

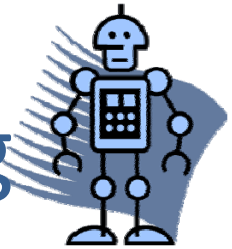


- LEGO League: Roboter ist ungenau positioniert, hat keine geeigneten Sensoren, soll aber genau ein Ziel treffen
- Idee: Lokalisierungsmodul **nicht nur passiv** („Positionssensor“), sondern:
  - **Aktiver Eingriff** in die Bewegung
  - Welche Bewegungen führen zu einer guten Lokalisierung?
  - **Fahre 2 Sekunden rückwärts → y=0**



-> Film Error Force One (2009)

Fazit relative und absolute Positionierung



- Relative Positionierung: auf kurze Distanz genau, aber **zunehmende** Ungenauigkeit
- Absolute Positionierung: prima, aber Landmarken **nicht immer sichtbar oder eindeutig** (langer Korridor)
- Offene Fragen:
  - Wie können sich relative und absolute Positionierung ergänzen?
  - Wie kann die Unsicherheit in der Position repräsentiert werden?

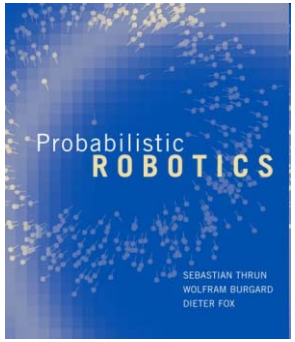
➔ Lösung

Probabilistische Lokalisierung

- leistungsfähiger, verbreiteter Ansatz
- vereint relative und absolute Postionierung als

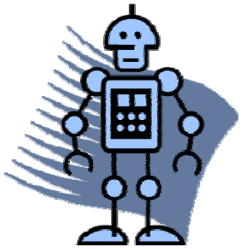
  Bewegungsmodellierung und Wahrnehmung der Umwelt

- ausführlich im Buch „**Probabilistic Robotics**“, S. Thrun, Wolfram Burkhard, Dieter Fox, MIT Press, 2006
  - Webseite: http://robots.stanford.edu/probabilistic-robotics/



- in ARIA umgesetzt in **ARNL** (Advanced Robot Navigation and Localization Software)
  - Methode: **Monte Carlo Localization**

ARNL (Wiederholung)



4 Module:

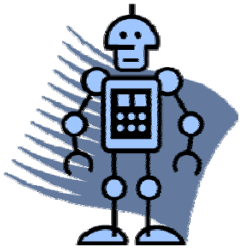
- BaseArnl: Infrastukture, **Pfadplanung, Pfadfolgen**
- **Lokalisierung** mit Sonar (SonArnl), Laserscanner (**Arnl**) und GPS (Mogs)

Hilfsprogramme:

- Kartenzeichnen
  - Mapper3Basic
  - Mapper3 inkluse Kartenerstellung aus aufgezeichneten Laserdaten und Odometriepositionen

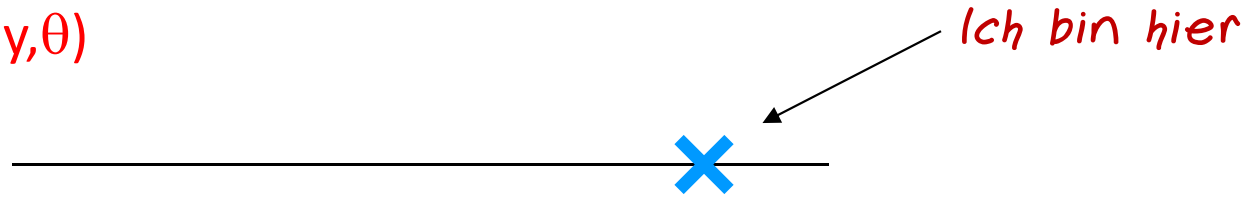
ca. 1k€/Roboter
closed source
SICK, Hokuyo

Probabilistische Lokalisierung



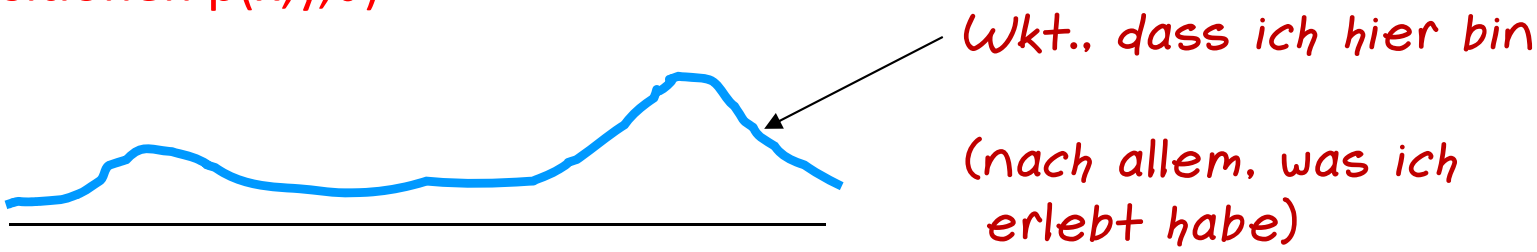
Wie wird im Roboter die Wissen / Vorstellung über die Position (engl.: belief) repräsentiert?

- **Bisher**: eine Position (x,y,θ)



  - Nachteile:
    - Unsicherheit wird nicht repräsentiert
    - nur <u>eine einzige</u> Position repräsentiert
  - Vorteil: Nutzung und Update einfach

- **Besser Probabilistische Lokalisierung**: Wahrscheinlichkeitsverteilung über mögliche Positionen p(x,y,θ)



3 Arten für die Repräsentation der Wahrscheinlichkeitsverteilung

Parametrische Form

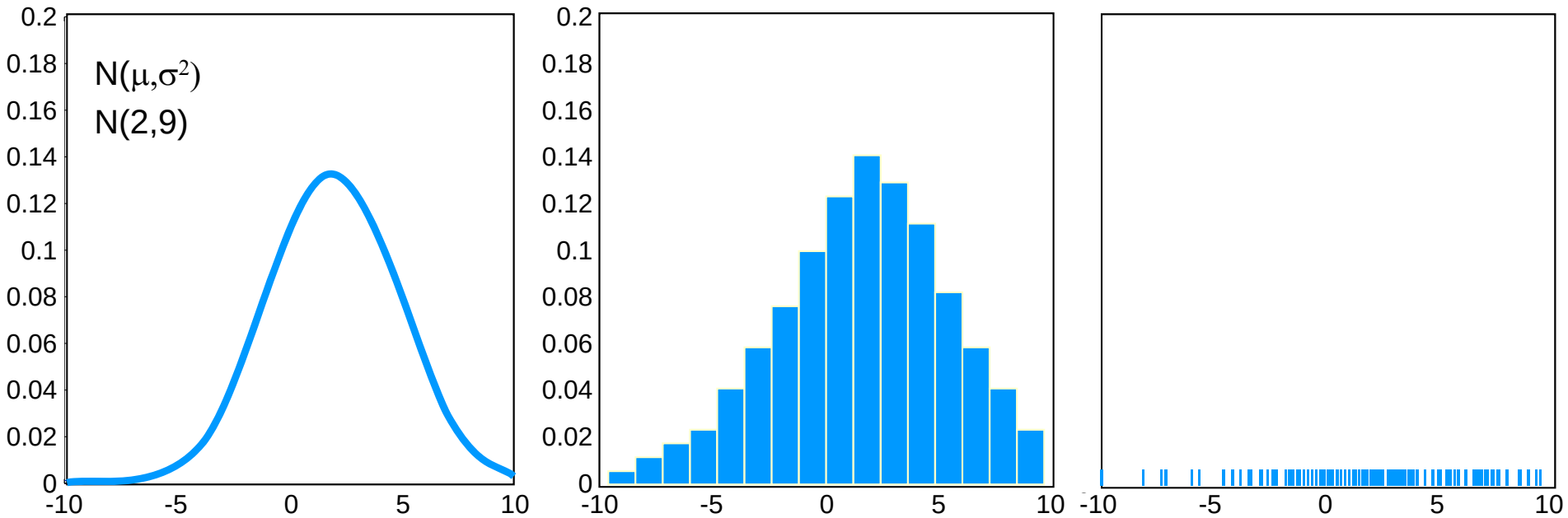
- Bsp. Gaußglocke
  - Kalman-Filter
- Extended KF
- oft unimodal

Histogramm

- sog. **Histogramm-Filter**
- Kacheln notwendig
- jede Kachel erhält eine Wahrscheinlichkeit
- speicherplatzintensiv

Partikelmenge (Samples)

- **Partikel-Filter**
- Monte-Carlo-Lokalisierung (ARNL)
- Dichte der Partikel <-> Wkt.
- Schnell, speicherplatzsparend



hier zur Vereinfachung nur **eindimensionale Position**

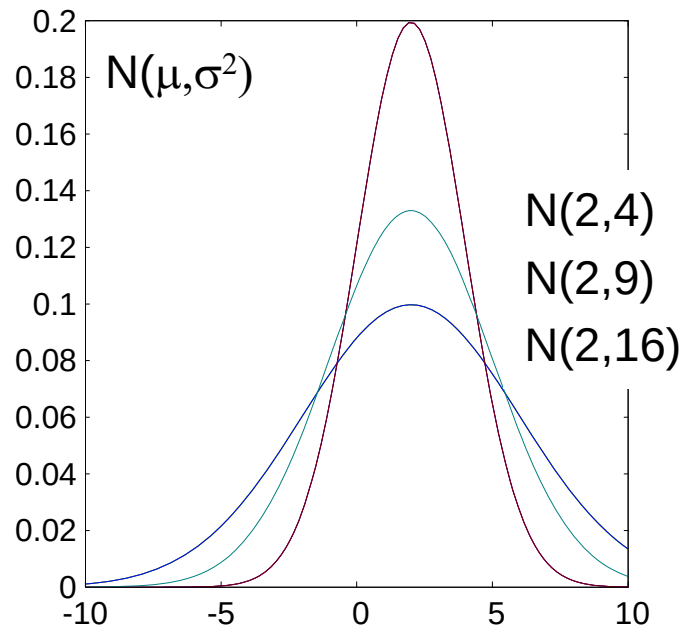
Parametrische Darstellung (in 1D)

- Formel für die Dichtefunktion,
- bei gleicher Formel genügt das Speichern und Updaten von Parametern

Beispiel: Normalverteilung (gaußsche Glockenkurve)

- 2 Parameter im eindimensionalen Fall: Mittelwert μ und Standardabweichung σ

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2} \right)$$



Multivariate Normalverteilung

- x kann auch ein Vektor sein, bspw. bei uns die Position (x,y,θ)
- dann: μ ein Vektor, σ^2 wird zur Kovarianzmatrix Σ

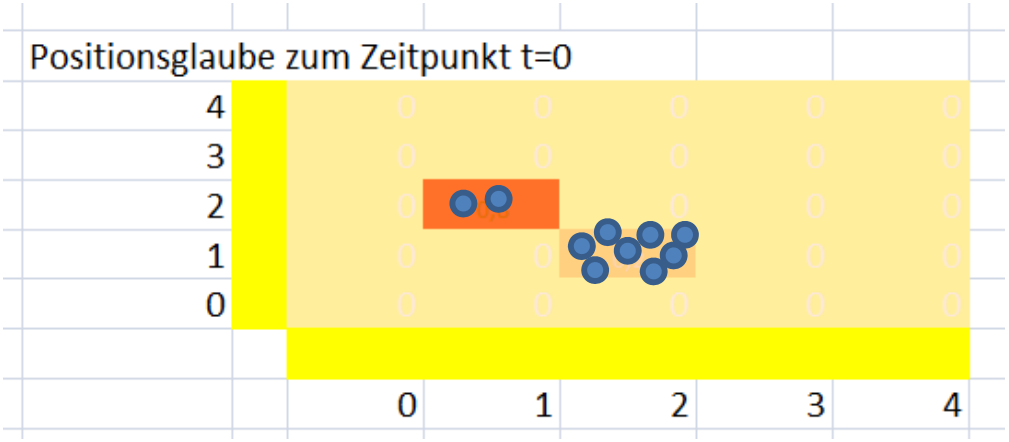
Zwei Dimensionen

- Wahrscheinlichkeitsverteilung als Histogramm

Positionsglaube zum Zeitpunkt t=0

4	0	0	0	0	0
3	0	0	0	0	0
2	0	0,8	0	0	0
1	0	0	0,2	0	0
0	0	0	0	0	0
	0	1	2	3	4

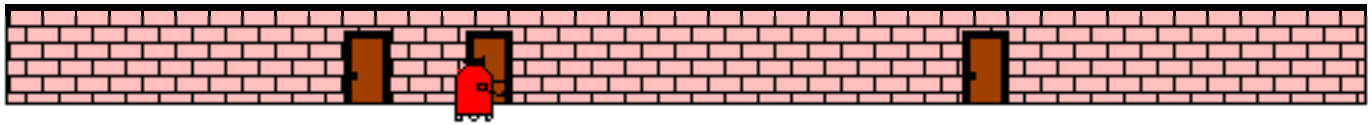
- Wahrscheinlichkeitsverteilung als Partikelmenge



ein Sample =
eine Positionshypothese

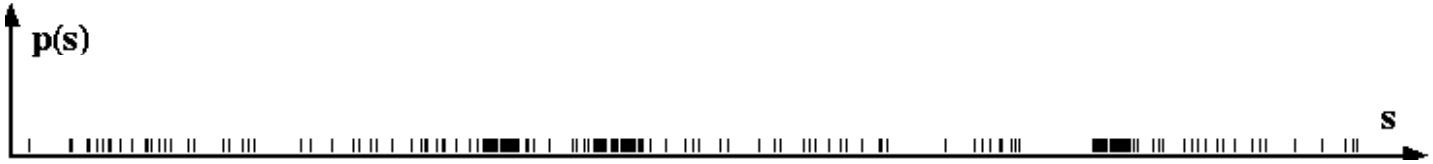
Multiple Hypothesen (in 1D)

Histogramm-Filter und Partikel-Filter können **multiple** Hypothesen repräsentieren:



Histogramm-Filter

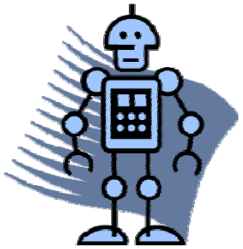
Partikel-Filter



Wie verändert sich der Positionsglaube (diese Verteilung), wenn sich der Roboter bewegt oder wenn er etwas sieht?

[TBF05]

Markov-Annahme



Wo bin ich?
Wahrscheinlichkeitsverteilung über Positionen:
Aktueller Zustand, bspw. (x,y,θ)

Vorheriger Zustand

Letzter Steuerbefehl (oder Odometrieticks bei uns)

$$p(x_t \mid x_{1:t-1}, z_{1:t}, u_{1:t}) = \underbrace{p(x_t \mid x_{t-1}, u_t)}$$

Bewegungsmodell

Was ist das? (Haben wir heute schon bei der Odometrie gehört)

Alle Zustände der Vergangenheit

Alle Beobachtungen der Vergangenheit, bspw. Wände

Alle Steuerbefehle (oder Odometrieticks bei uns) der Vergangenheit

Markov-Annahme

Die aktuelle Position x hängt **nur** von der vorhergehenden Position x und dem Bewegungsbefehl u ab (und nicht von allen bisherigen Positionen, allen Beobachtungen und allen Bewegungsbefehlen).

$$p(x_t \mid x_{1:t-1}, z_{1:t}, u_{1:t}) = p(x_t \mid x_{t-1}, u_t)$$

Bewegungsmodell

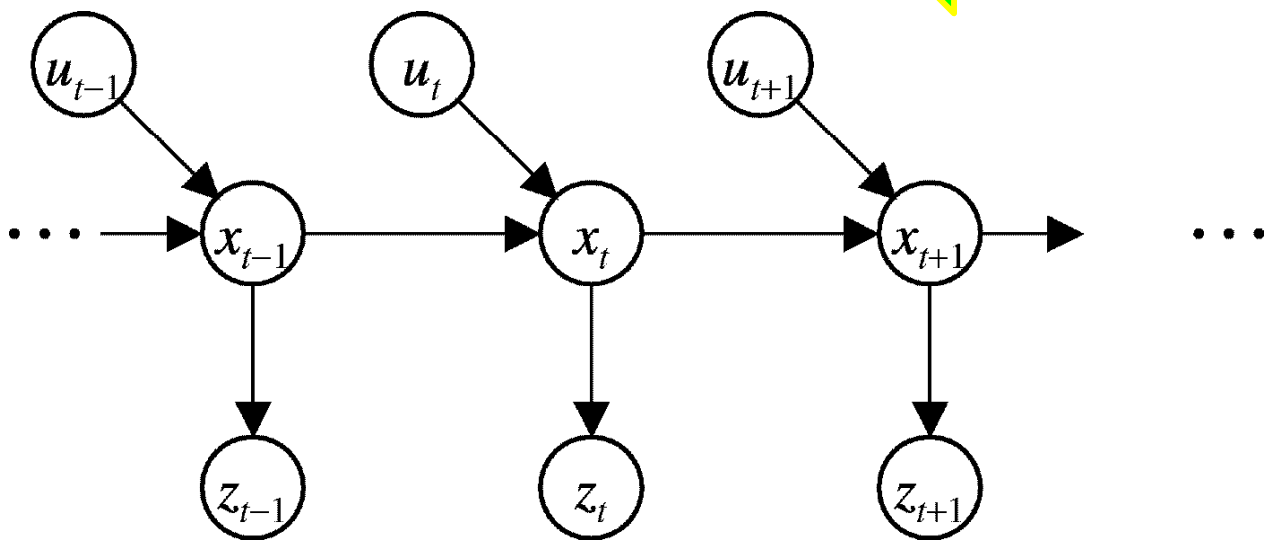
Die aktuelle Beobachtung z hängt **nur** von der aktuellen Position ab.

$$p(z_t \mid x_{0:t}, z_{1:t}, u_{1:t}) = p(z_t \mid x_t)$$

Sensormodell

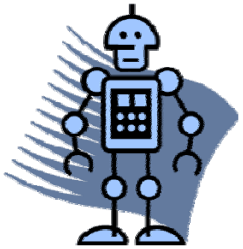
Bayessches Netz,

Knoten sind Zufallsvariablen,
Pfeil bedeutet "beeinflußt"
oder in der Gegenrichtung
"ist abhängig von"



[TBF05]

Beispiel: Unscharfes Bewegungsmodell



Bewegungsmodell

Roboter im Gitter an der Position x,y: Bewegungsbefehl (oder Messung) = ‚Gehe nach Norden', Erfolg (Schritt tatsächlich nach Norden, y->y+1) **mit p = 0.7,** Schritt nach Osten, Westen oder Süden mit je p = 0.1

$$p(x_t \mid x_{t-1}, u_t)$$

p(Pos(t)=(x,**y+1**) | Pos(t-1) = (x,y), u(t) = „Gehe nach Norden") = 0.7
p(Pos(t)=(x,**y-1**) | Pos(t-1) = (x,y), u(t) = „Gehe nach Norden") = 0.1 …

Totale Wahrscheinlichkeit:

$$P(A) = \sum_{i=1}^{n} P(A \cap H_i) = \sum_{i=1}^{n} P(A|H_i) \cdot P(H_i)$$

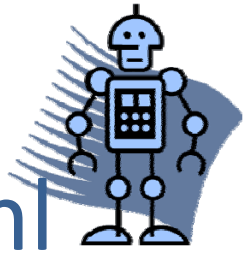
Roboter glaubt: Pos=(2,1) mit p=0.2 und Pos=(1,2) mit p=0.8
Er wählt „Gehe nach Norden",
Wie groß ist jetzt die Wahrscheinlichkeit für die Pos (2,2) ?

Positionsglaube zum Zeitpunkt t=0

4	0	0	0	0	0	
3	0	0	0	0	0	
2	0	0,8	0	0	0	Summe
1	0	0	0,2	0	0	1
0	0	0	0	0	0	
	0	1	2	3	4	

[TBF05]

Beispiele: Prognose-Schritt mit Bewegungsmodell und Bewegungsbefehl



p(Pos(t+1)=(2,2)) = p(Pos(t)=(2,2) | Pos(t-1)=(2,1), u(t)=„Norden“) * p(Pos(t)=(2,1)) + p(Pos(t)=(2,2) | Pos(t-1)=(1,2), u(t)=„Norden“) * p(Pos(t)=(1,2)) + 0 + 0 + ... + 0
= 0.7 * 0.2 + 0.1 * 0.8 = 0.14 + 0.08 = **0.22**

Positionsglaube zum Zeitpunkt t=0

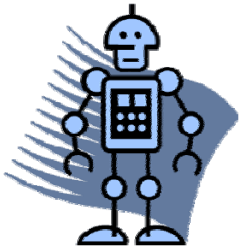
4	0	0	0	0	0	
3	0	0	0	0	0	
2	0	0,8	0	0	0	Summe
1	0	0	0,2	0	0	1
0	0	0	0	0	0	
	0	1	2	3	4	

Positionsglaube zum Zeitpunkt t=1 nach der Aktion "Gehe nach Norden"

4	0	0	0	0	0	
3	0	0,560	0	0	0	
2	0,08	0	0,22	0	0	
1	0	0,1	0	0,02	0	Summe
0	0	0	0,02	0	0	1
	0	1	2	3	4	

[TBF05]

Beispiel: Sensormodell



Bewegungsmodell $p(x_t \mid x_{t-1}, u_t)$

Die neue Wkt für **alle** Positionen x_t berechnen: **jeweils** ein Ausdruck mit einer **Summe über alle Positionen**! (sog. Faltung, engl. convolution)

Aufwändig in der Navigation -> besser beim Partikelfilter (gleich)

Sensormodell $p(z_t \mid x_t)$

Einige Felder sind schwarz, der Roboter hat dafür einen Sensor.

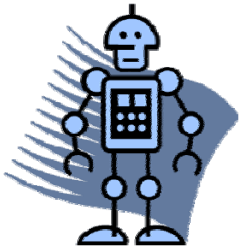
Roboter nimmt die Farbe Schwarz in 90% der Fälle richtig wahr und die Farbe Weiß in 70% der Fälle.

p(Sensor misst = Schwarz | Feld ist = Schwarz) = 0.9

p(Sensor misst = Schwarz | Feld ist = Weiß) = ...

[TBF05]

Beispiel: Korrektur-Schritt mit Sensormodell + Wahrnehmung



- Unser (richtiges) Weltmodell: Nur Feld (1,3) ist schwarz
- Der Farbsensor misst schwarz.

Wie ändert sich die Wahrscheinlichkeitsverteilung?

Positionsglaube zum Zeitpunkt t=1 nach der Aktion "Gehe nach Norden"

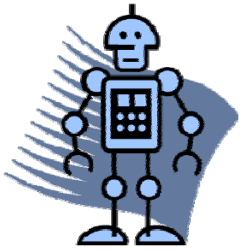
4	0	0	0	0	0	
3	0	0,560	0	0	0	
2	0,08	0	0,22	0	0	
1	0	0,1	0	0,02	0	Summe
0	0	0	0,02	0	0	1
	0	1	2	3	4	

BayesRegel

$$P(H|A) = \frac{P(A|H) \cdot P(H)}{P(A)}$$

- p(Pos=(2,2) | Messung=schwarz) = p(Messung=schwarz | Pos=(2,2)) * p(2,2) / p(Messung=schwarz)
- p(Messung=schwarz) bestimmbar, aber für alle Positionen gleich -> Konstante, muss nicht ausgerechnet werden, wir normieren alle Werte stattdessen auf die Summe 1
- p(Pos=(2,2)| Messung=schwarz) = 0.9 * **0.56** / Konstante = 0.504 / K = **0.792**
- Für alle anderen Felder ist der Faktor 0.3 (Weiß wird als Schwarz gemessen)

Beispiel: Korrektur-Schritt mit Sensormodell + Wahrnehmung



- Korrekur-Schritt: die Wahrscheinlichkeit von Positionen, die zur Wahrnehmung passen, wird angehoben – die der anderen abgesenkt

Positionsglaube zum Zeitpunkt t=1 nach der Aktion "Gehe nach Norden"

	0	1	2	3	4
4	0	0	0	0	0
3	0	0,560	0	0	0
2	0,08	0	0,22	0	0
1	0	0,1	0	0,02	0
0	0	0	0,02	0	0

Summ
1

Multiplikation mit p(z|x)

Feld (1,3) ist schwarz + Wahrnehmung schwarz

	0	1	2	3	4
4	0,000	0,000	0,000	0,000	0,000
3	0,000	0,504	0,000	0,000	0,000
2	0,024	0,000	0,066	0,000	0,000
1	0,000	0,030	0,000	0,006	0,000
0	0,000	0,000	0,006	0,000	0,000

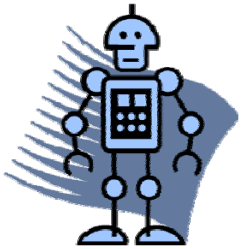
Summe
0,636

Normieren auf 1

	0	1	2	3	4
4	0,000	0,000	0,000	0,000	0,000
3	0,000	0,792	0,000	0,000	0,000
2	0,038	0,000	0,104	0,000	0,000
1	0,000	0,047	0,000	0,009	0,000
0	0,000	0,000	0,009	0,000	0,000

Summe
1

Markov-Lokalisierung



„The central idea of Markov localization is to represent the robots's belief by a ***probability distribution over possible positions****, and use* ***Bayes rule*** *and* ***convolution*** *to* ***update the belief whenever the robot senses or moves****."*

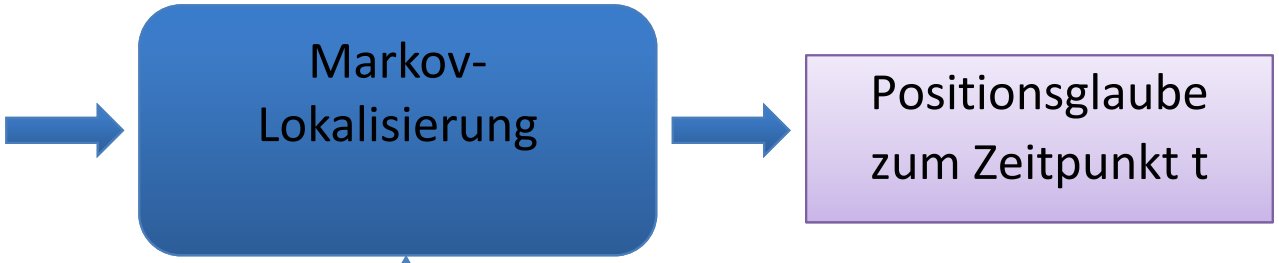
Grundidee:

Belief (Positionsglaube) als Wahrscheinlichkeitsverteilung,

die sich bei Bewegung und Wahrnehmung ändert.

Markov-Lokalisierung

- Positionsglaube zum Zeitpunkt t-1
- Bewegung: Steuerbefehl oder Beobachtung der eigenen Bewegungsorgane
- Umweltwahrnehmung: Landmarken, Laserscan ...



Modellierung der Abhängigkeiten ist **konstant**:

- **Bewegungsmodell**: Ort + Bewegung -> neuer Ort
- **Sensormodell**: Karte + Ort -> Beobachtung

Für 2011 auf Partikelversion umstelle

Markov-Lokalisierung

Markov_localization(Bel(x_{t-1}), u_t, z_t, m) {

for all x_t do:

$$Bel'(x_t) = \sum_{x_{t-1}} p(x_t \mid u_t, x_{t-1}) \cdot Bel(x_{t-1})$$

1. Prognose mit Bewegungsmodell

$$Bel(x_t) = \eta \cdot p(z_t \mid x_t, m) \cdot Bel'(x_t)$$

2. Korrektur mit Sensormodell

endfor

return Bel(x_t)

x = Position bspw. Ort im Raum, z.B. x = ‚Mensa' oder x = (30,40,70°)

Bel(x) = Positionsglaube: Wahrscheinlichkeit, sich an der Position x zu befinden

u = Steuerbefehle oder Odometermessung

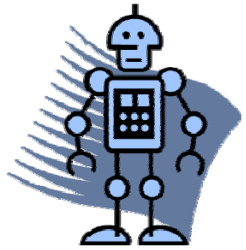
z = Beobachtung, bspw. Laserscan

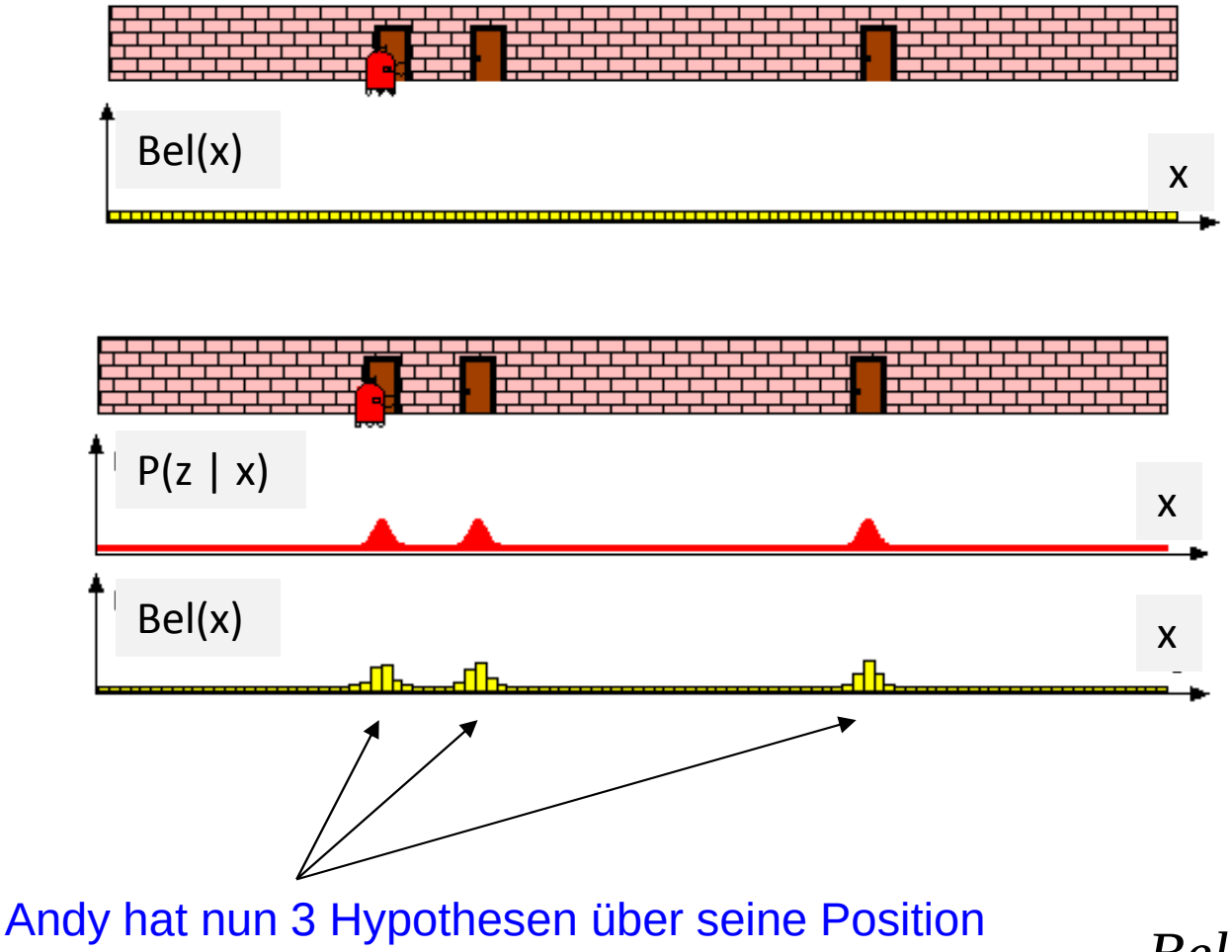
m = Karte

η = Normierungsfaktor, so dass die Summe aller bel(x) 1 ergibt

... Folie auf Partikelversion umstellen

Andy der eindimensionale Roboter präsentiert: Histogramm-Filter





- Histogramm-Filter, Position x

Initialzustand: jede Position x ist gleichwahrscheinlich

Sensormodell P(z | x): Wahrscheinlichkeit, eine Tür zu sehen (Beobachtung z), wenn man sich an der Stelle x befindet

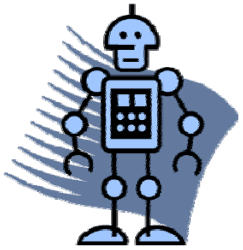
Andy sieht eine Tür ->

Bel(x) wird neu berechnet mit Korrektur-Schritt, die Wahrnehmung korrigiert den Positionsglauben

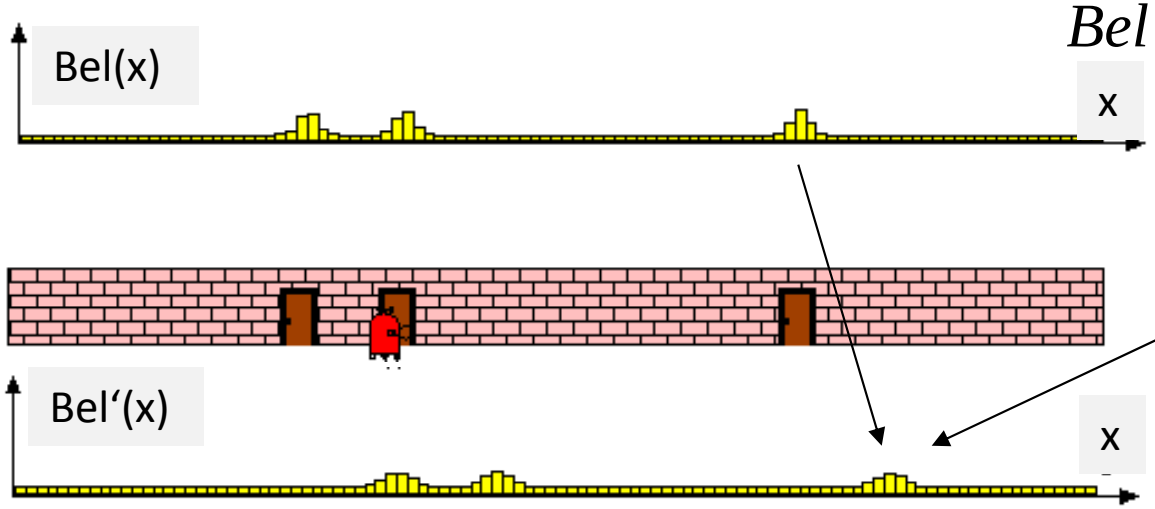
$$Bel(x_t) = \eta \cdot p(z_t \mid x_t, m) \cdot Bel'(x_t)$$

[TBF05]

Andy der eindimensionale Roboter präsentiert: Histogramm-Filter



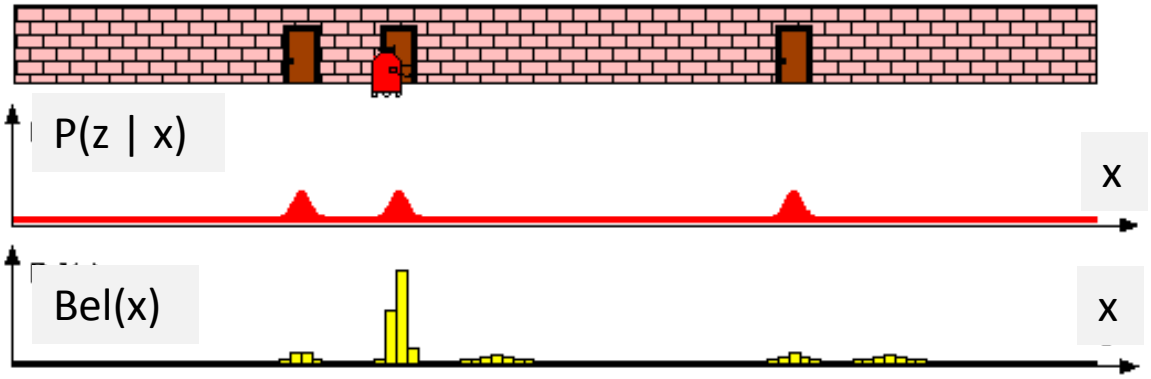
$$Bel'(x_t) = \sum_{x_{t-1}} p(x_t \mid u_t, x_{t-1}) \cdot Bel(x_{t-1})$$



Andy fährt ein Stück

Prognose-Schritt: Bel‘(x) wird mit dem Bewegungsmodell neu berechnet (aufwendig), Positions-Unsicherheit wächst!

Erste Runde vorbei



Andy sieht eine Tür ->

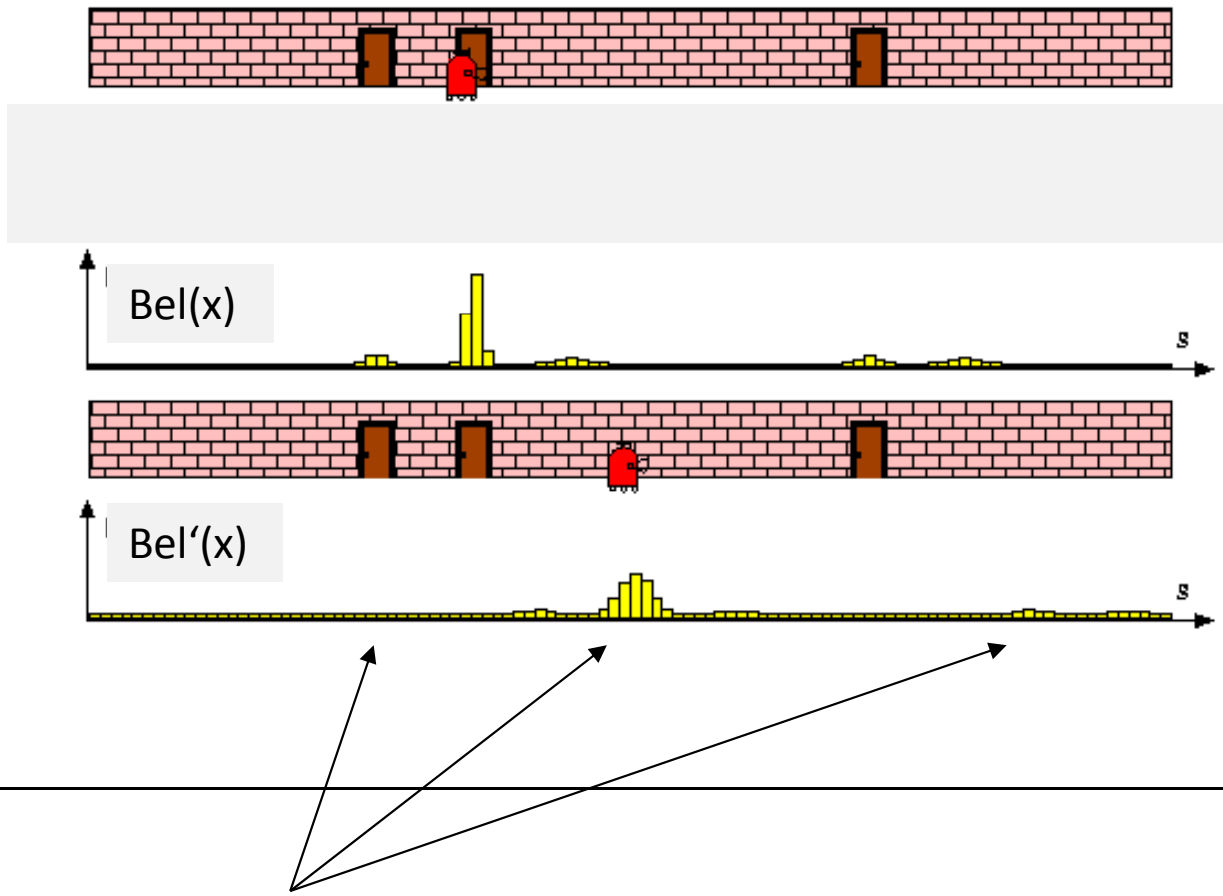
Sensormodell P(z | x): konstant

Bel(x) wird neu berechnet mit Korrektur-Schritt (Multiplikation wie vorher)

[TBF05]

Andy hat nun 1+4 Hypothesen über seine Position

Andy der eindimensionale Roboter präsentiert: Histogramm-Filter



Andy fährt ein Stück,

Bel‘(x) wird mit dem Prognose-Schritt neu berechnet, Positions-Unsicherheit wächst wieder

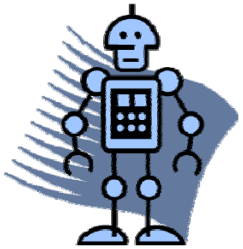
Zweite Runde vorbei

usw. usf.

Andy hat 1 Hypothese, hält aber alles für möglich

[TBF05]

Monte Carlo Localization (MCL)



- Nachteil Histogramm-Filter: Prognose-Schritt (hier) zu aufwändig →
- **Monte Carlo Localization** = Markov-Lokalisierung mit Abbildung der Wahrscheinlichkeit als **Samplemenge** (Partikelwolke)
- Sample = eine Position (x,y,θ)

- Prognose-Schritt: jedes Sample erzeugt ein neues Sample entsprechend unscharfem Bewegungsmodell -> neue Samplemenge bel‘
- Korrektur-Schritt:
  - Wichtung: Jedes Partikel aus Bel‘ erhält ein Gewicht nach dem Sensormodell (wie eine Fitness, Übereinstimmung mit Wahrnehmung)
  - Resampling: Neue Samplemenge Bel wird durch zufällige Selektion anhand der Fitness aus Bel‘ erzeugt

→ ständige Evolution einer Partikelwolke anhand von Bewegungsbefehlen und Sensorwahrnehmung

EA im Master, 2. Semester, KI

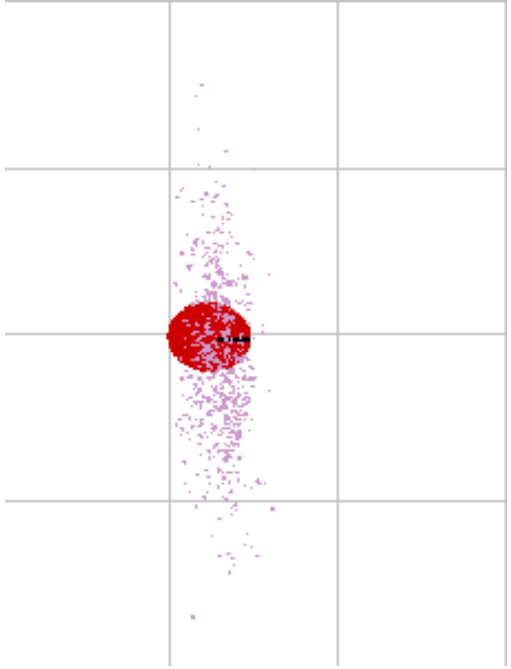
Samplemenge, Partikelwolke,

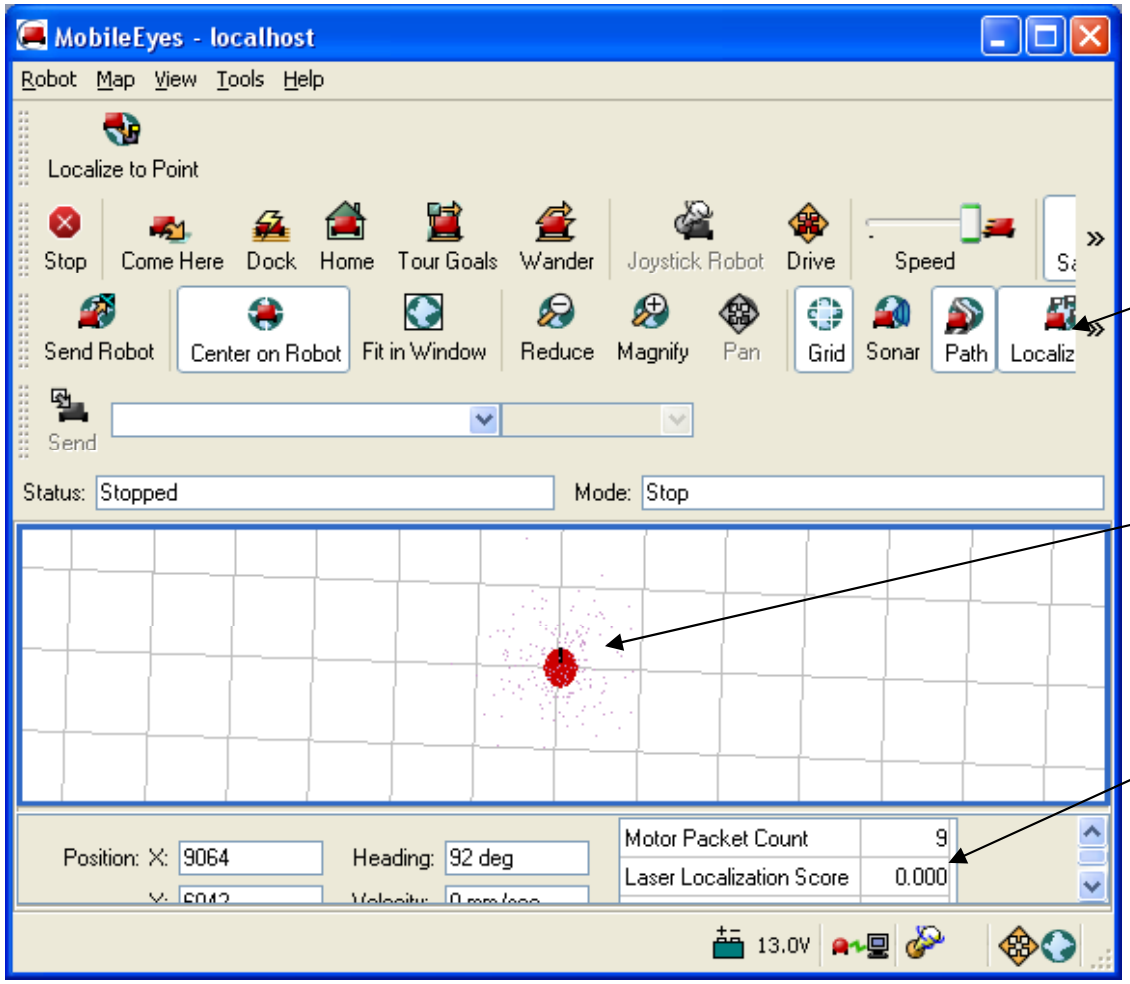
Positionsglaube des Roboters als Partikelwolke

Wie kann diese Verteilung entstanden sein?

[TBF05]

Partikelwolke in MobileEyes



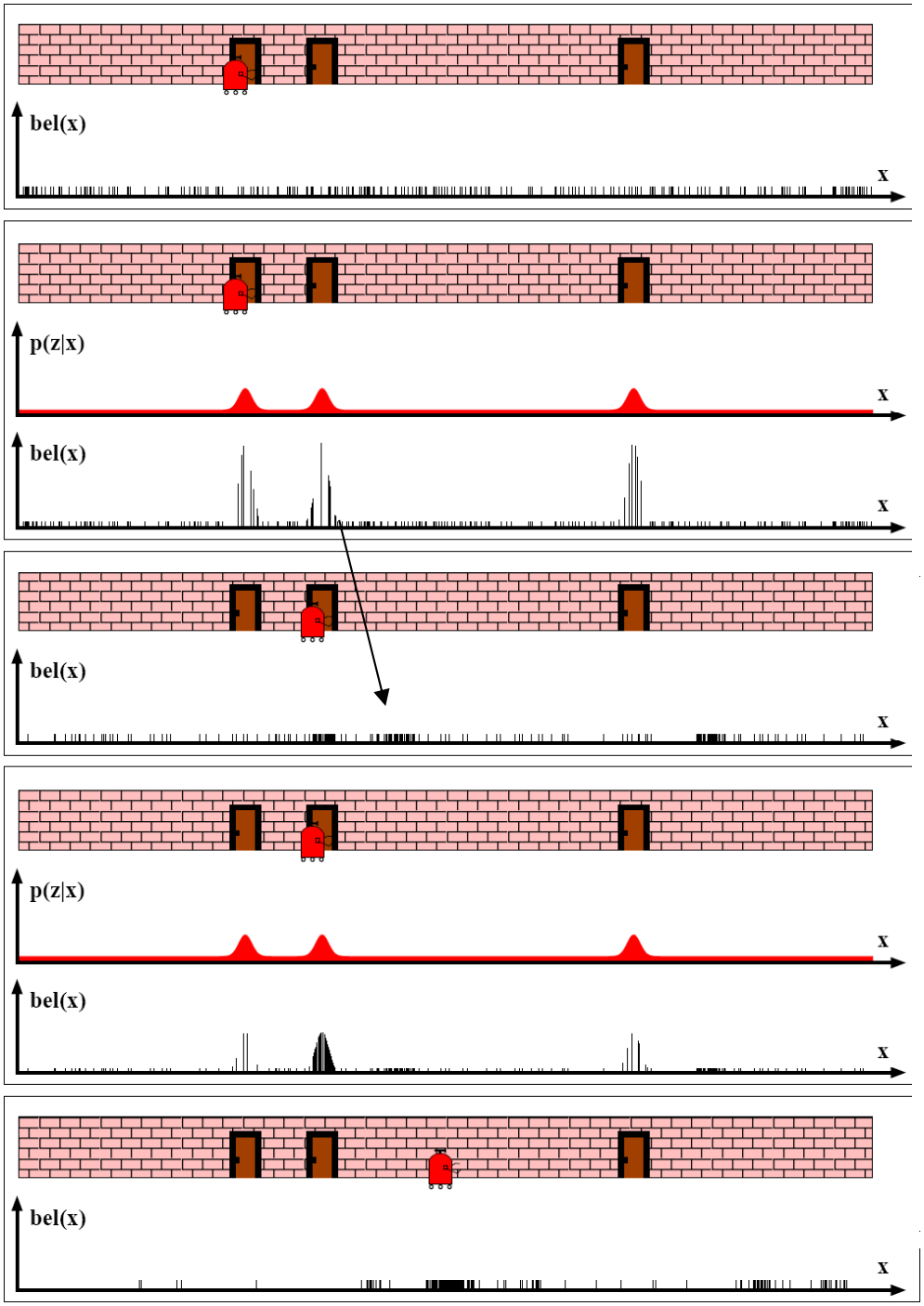


Anzeige Samplemenge einschalten

Samples (sehr klein und violett)

-> Übung

LLS = Laser Localization Score = das aktuelle maximale Gewicht, d.h. wie gut stimmt das beste Sample mit der Sensorwahrnehmung überein, Maximum 1.0



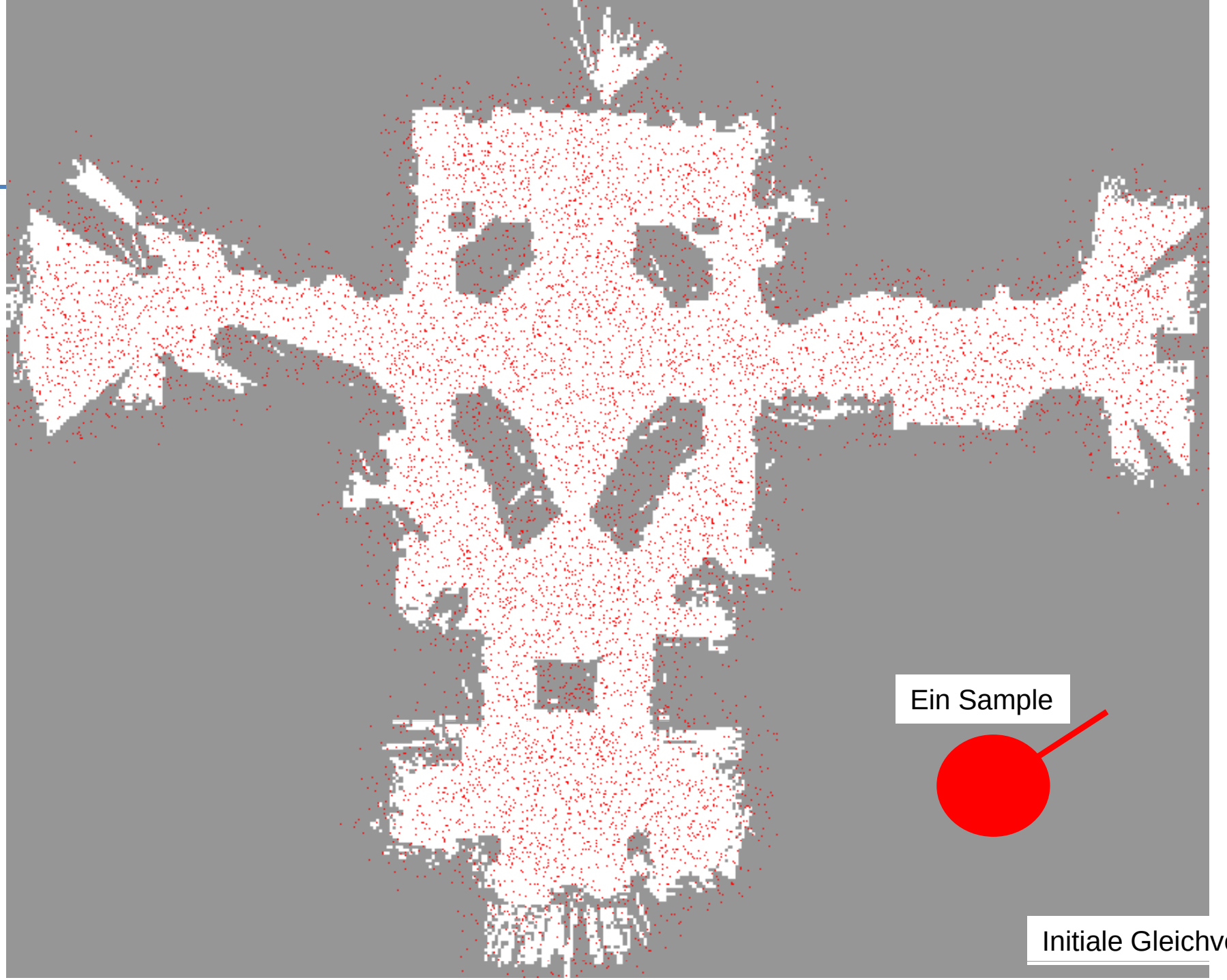
- gleichmäßige Verteilung der Samples
- Sensormodell p(z | x)
- Korrektur-Schritt: Wichtung der Samples
- Resampling: neue Sample-Menge erzeugen, dann Prognose-Schritt
- Korrektur-Schritt: Wichtung der Samples
- Resampling: neue Sample-Menge erzeugen, dann Prognose-Schritt

[TBF05]

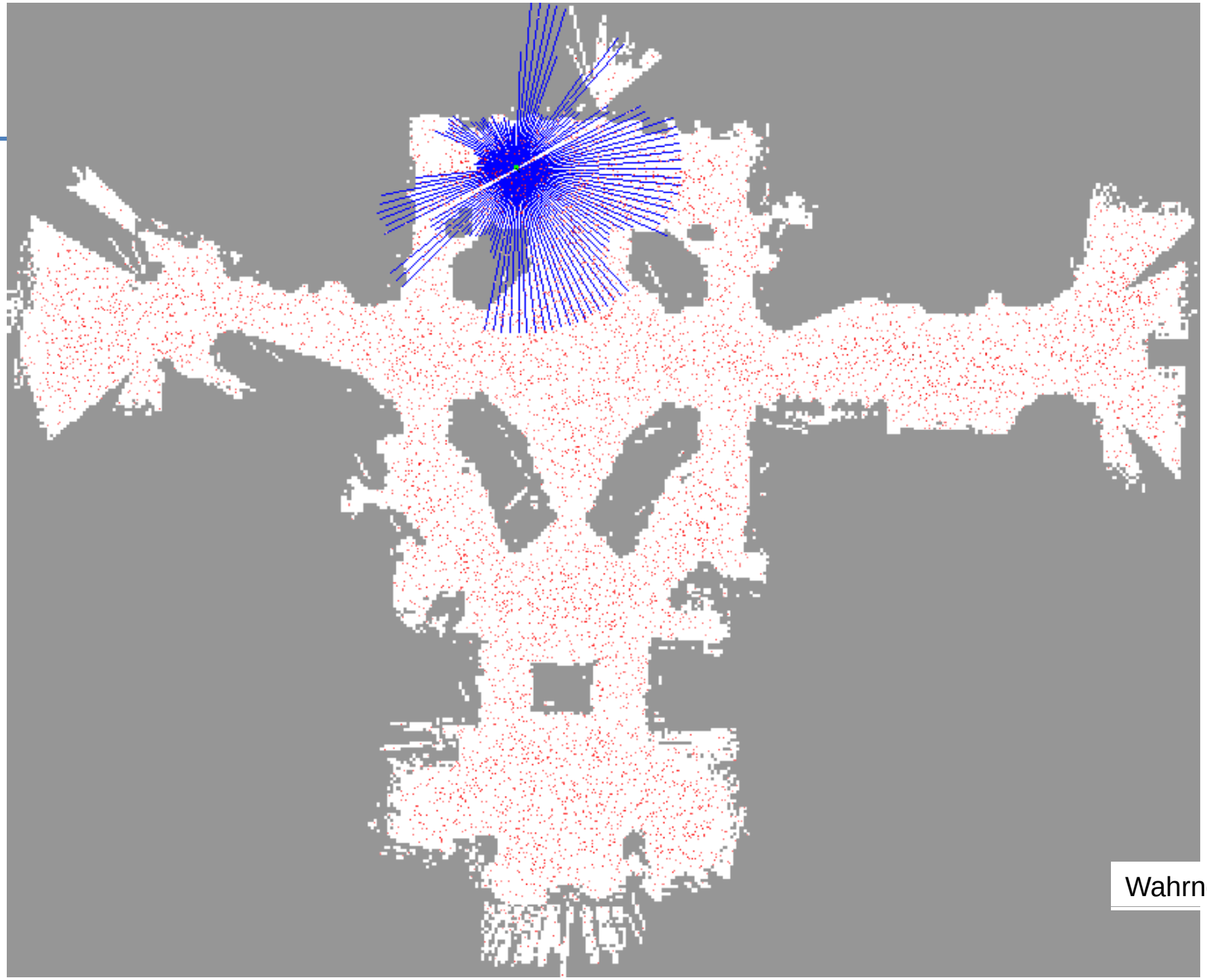
Beispiel MCL

- aus particle-filters.ppt
- http://robots.stanford.edu/probabilistic-robotics/ppt/

[TBF05]

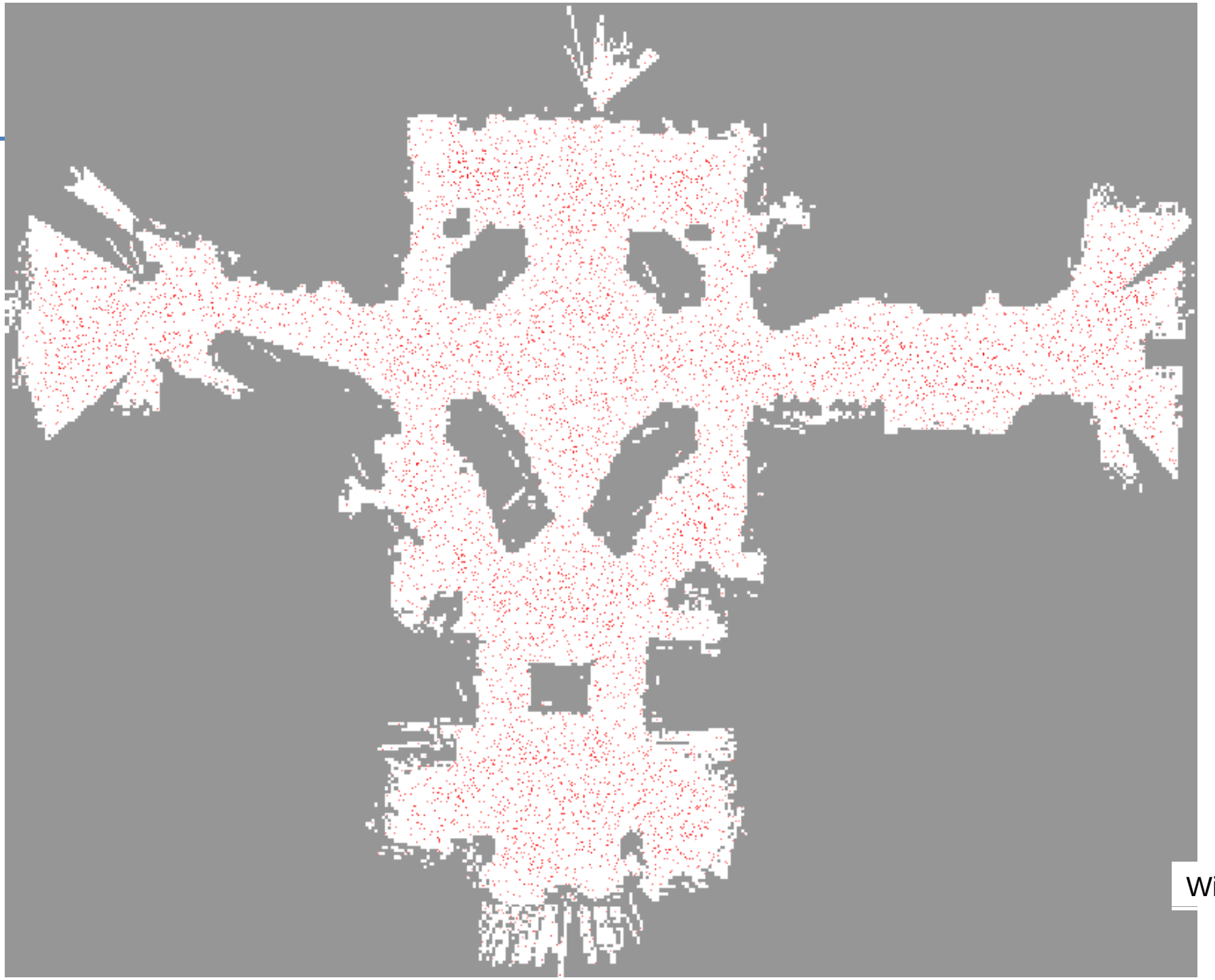


[TBF05]



[TBF05]

Wahrnehmung



[TBF05]

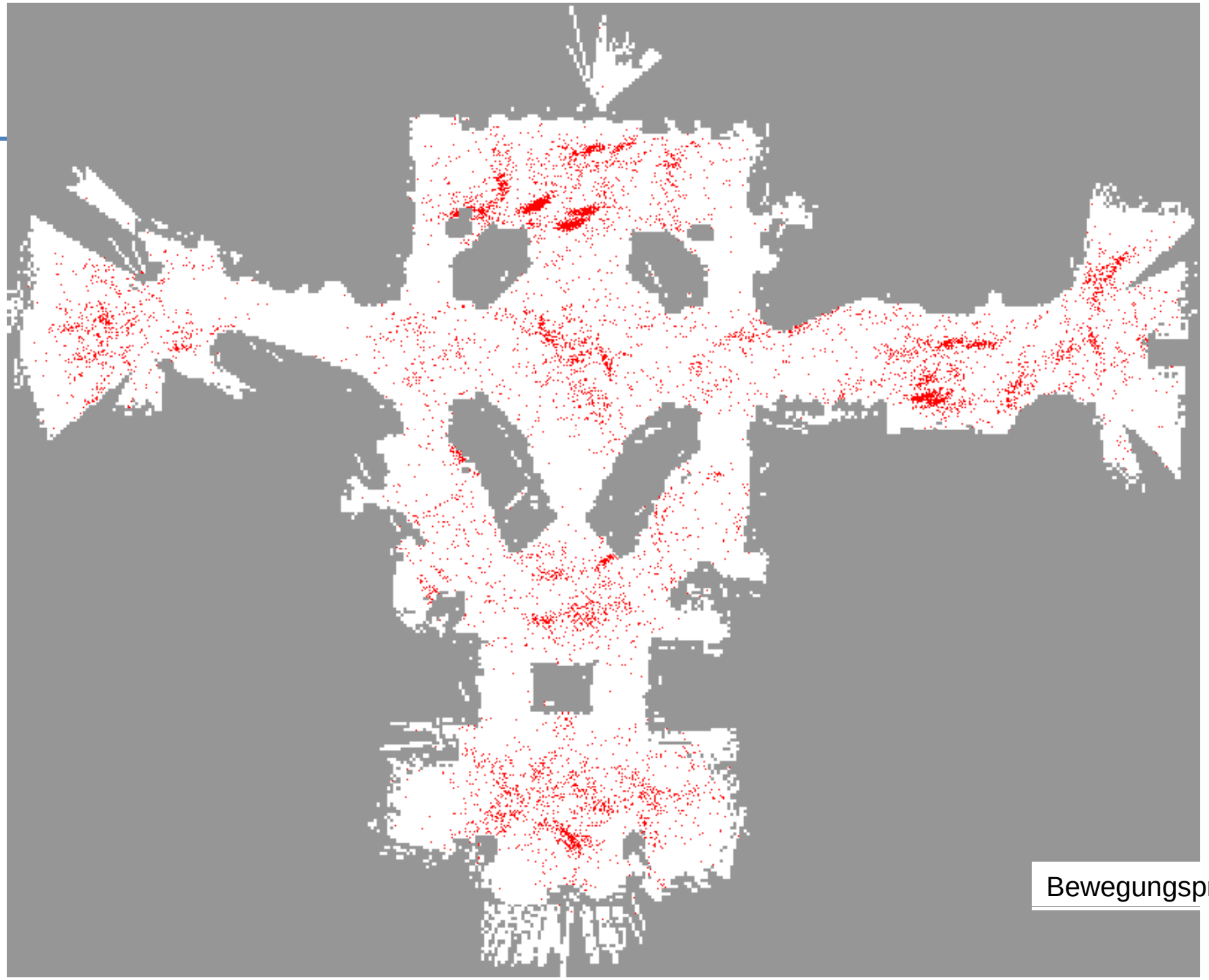
Wichtung

[TBF05]

Bild fehlt,

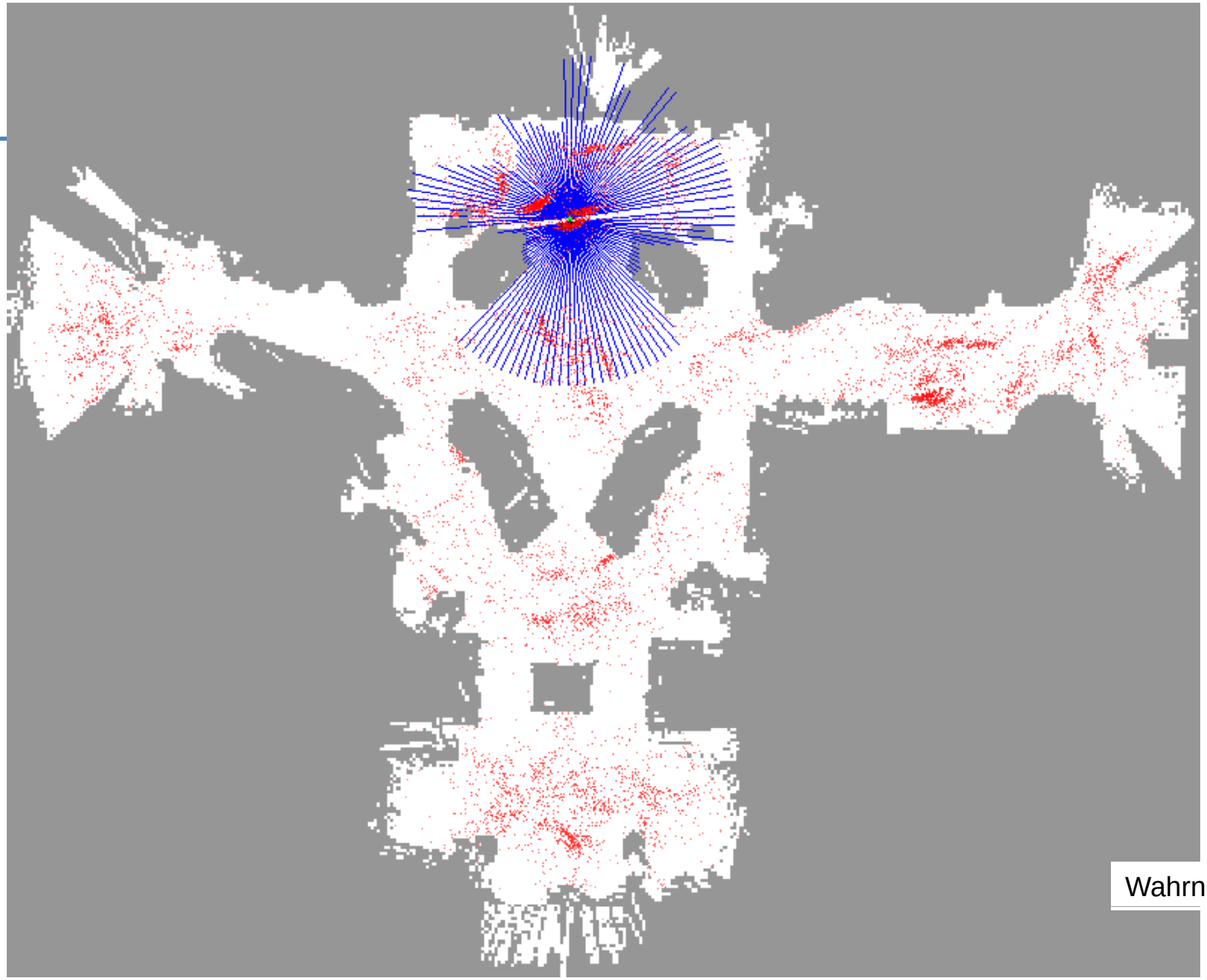
Samplemenge dünnt aus

Resampling



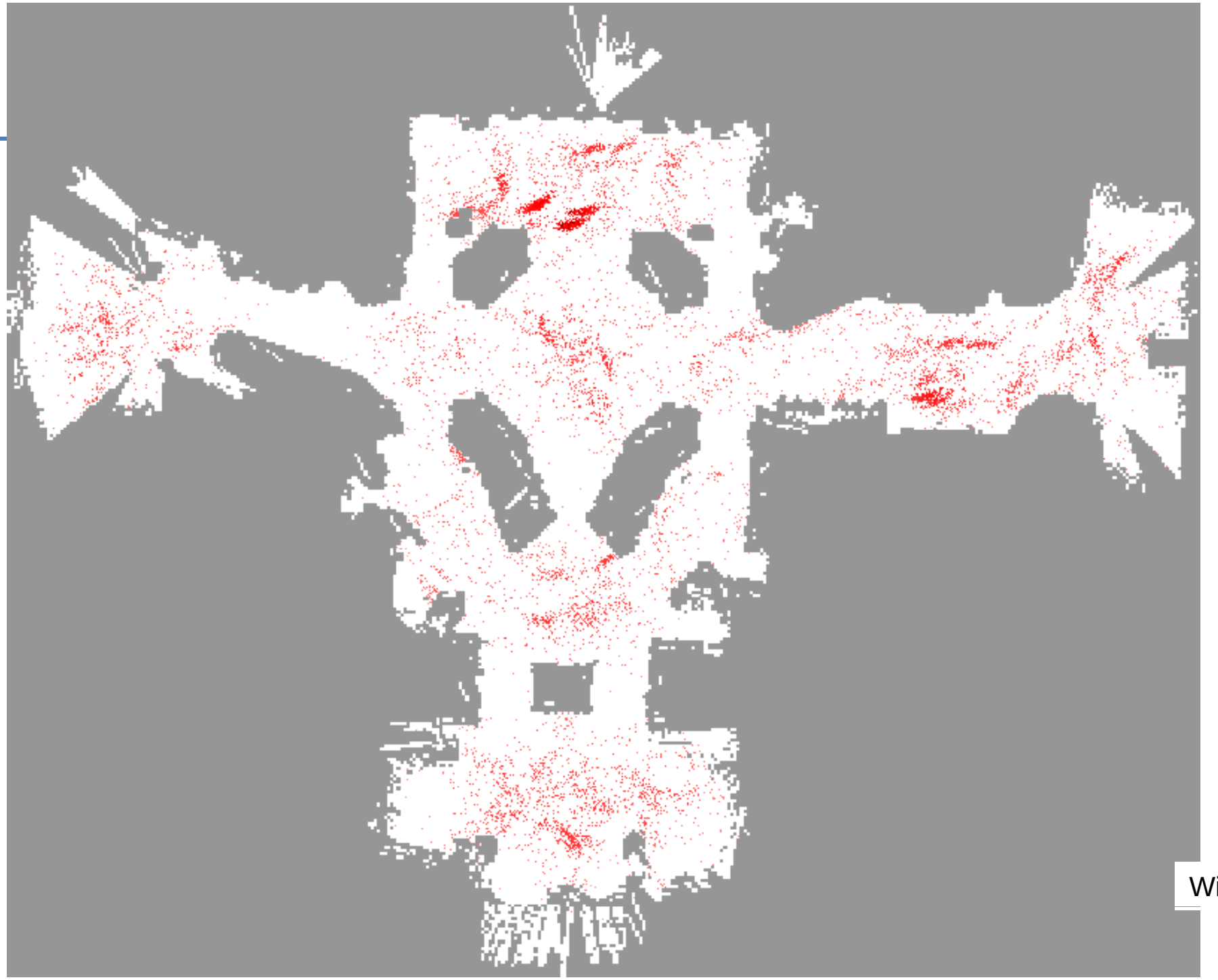
Bewegungsprognose

[TBF05]



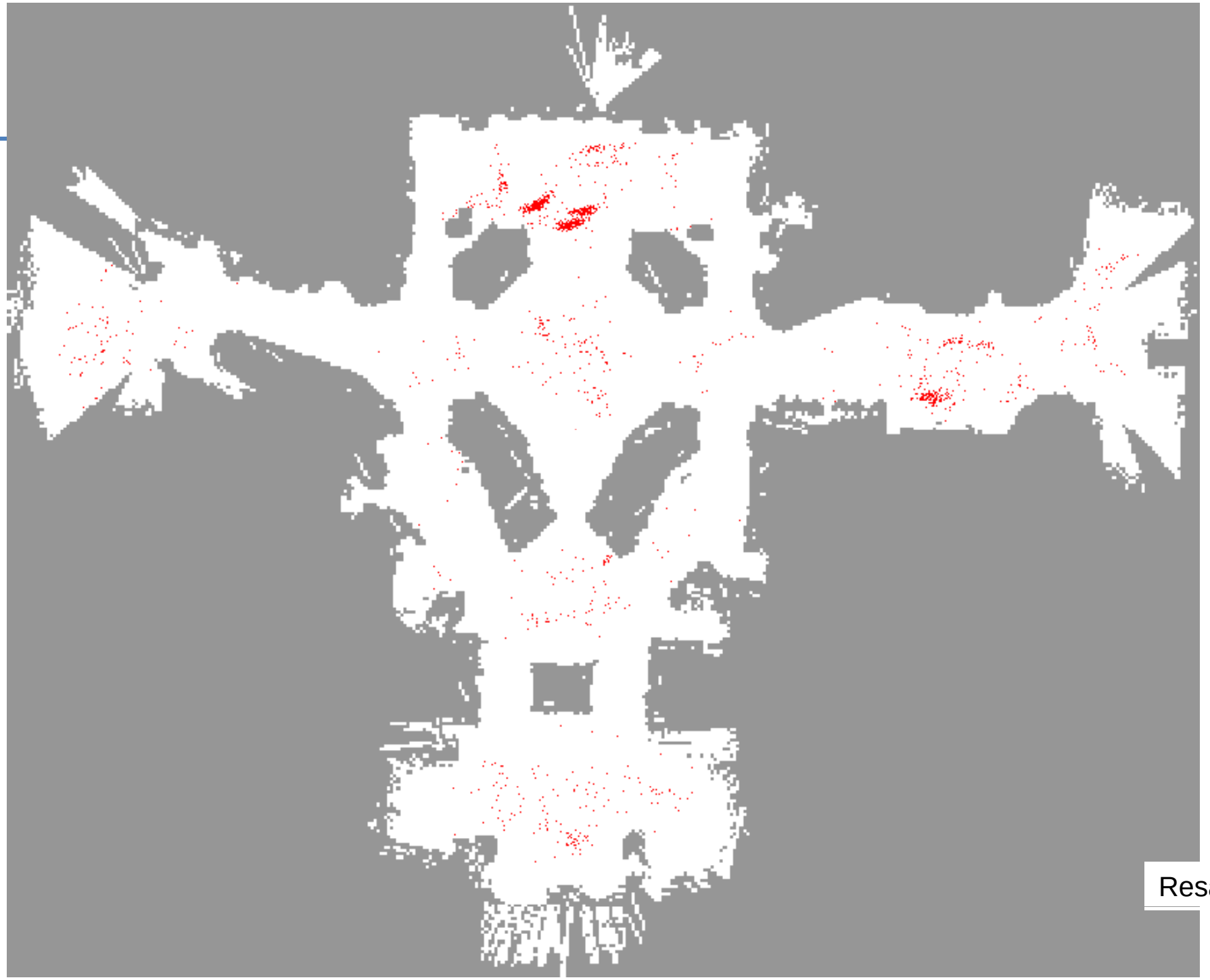
[TBF05]

Wahrnehmung



[TBF05]

Wichtung

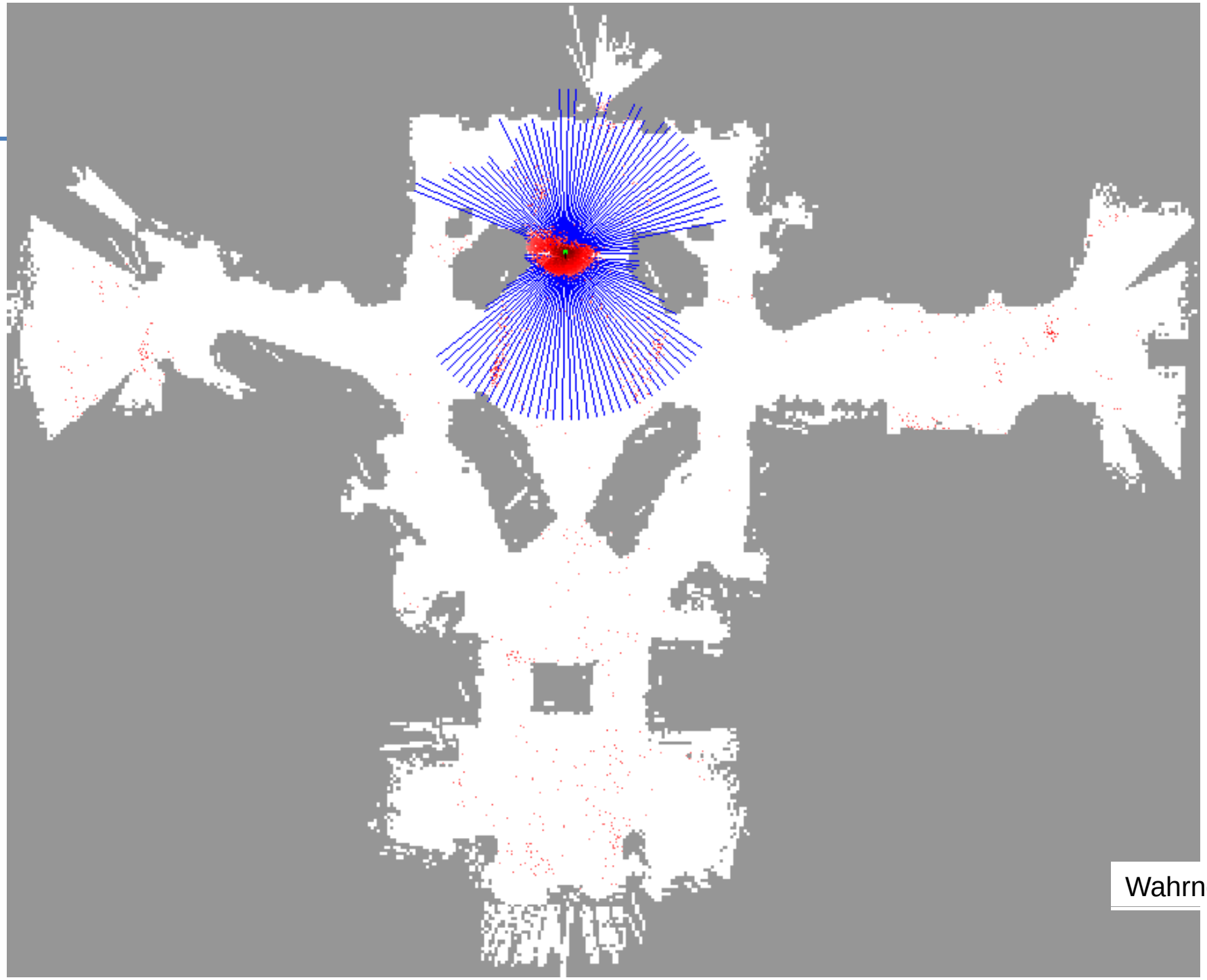


[TBF05]

Resampling

[TBF05]

Bewegungsprognose

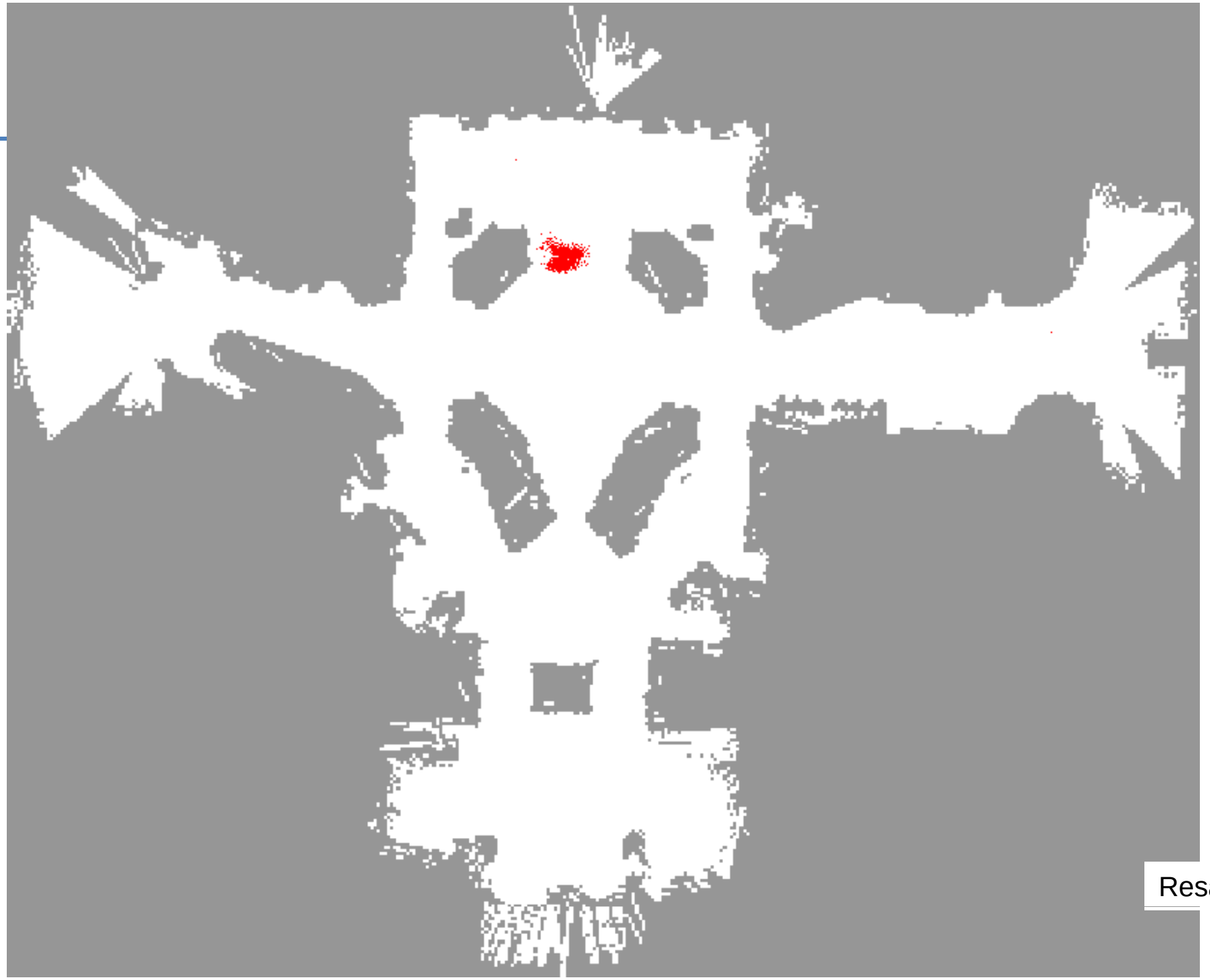


Wahrnehmung

[TBF05]

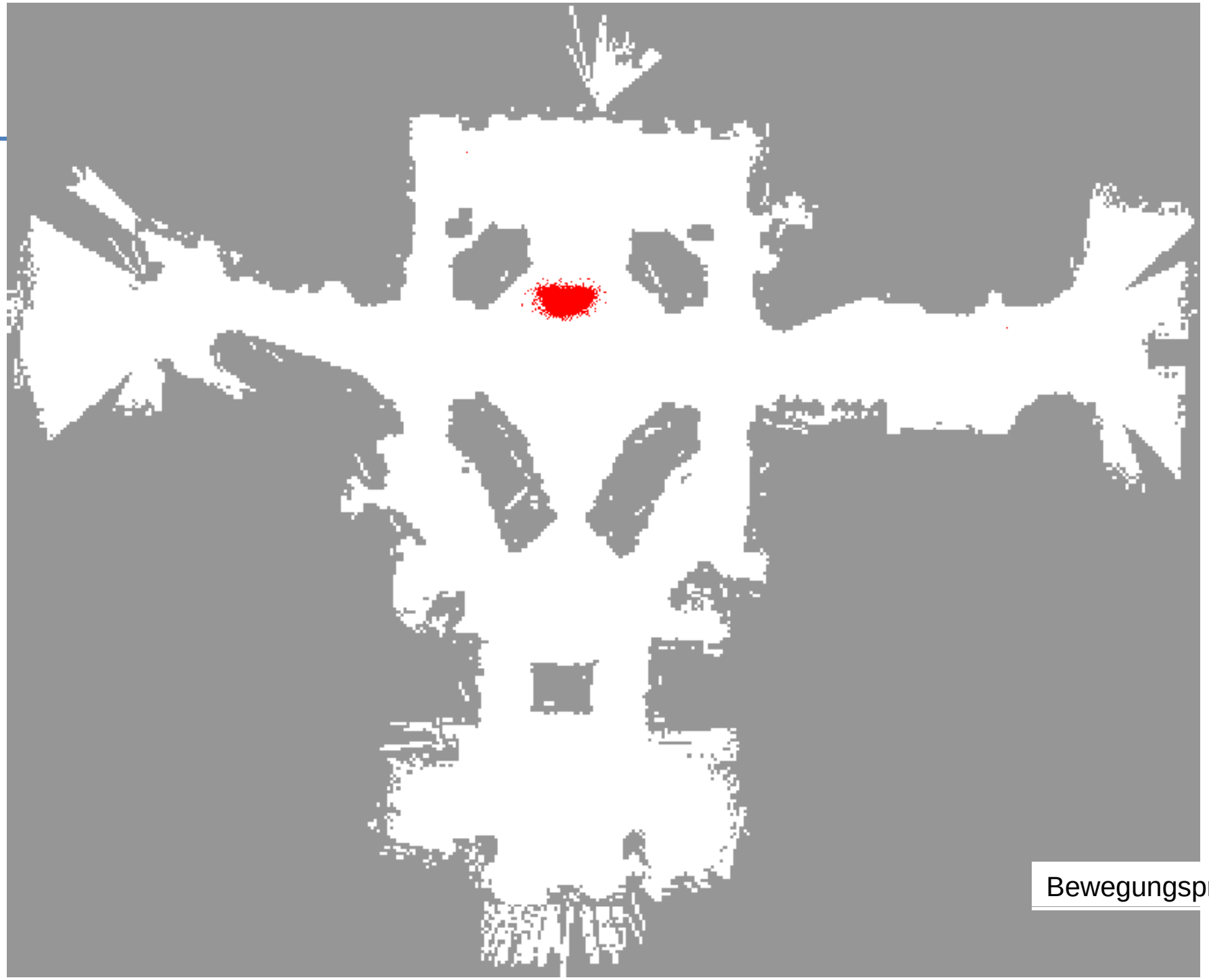
[TBF05]

Wichtung



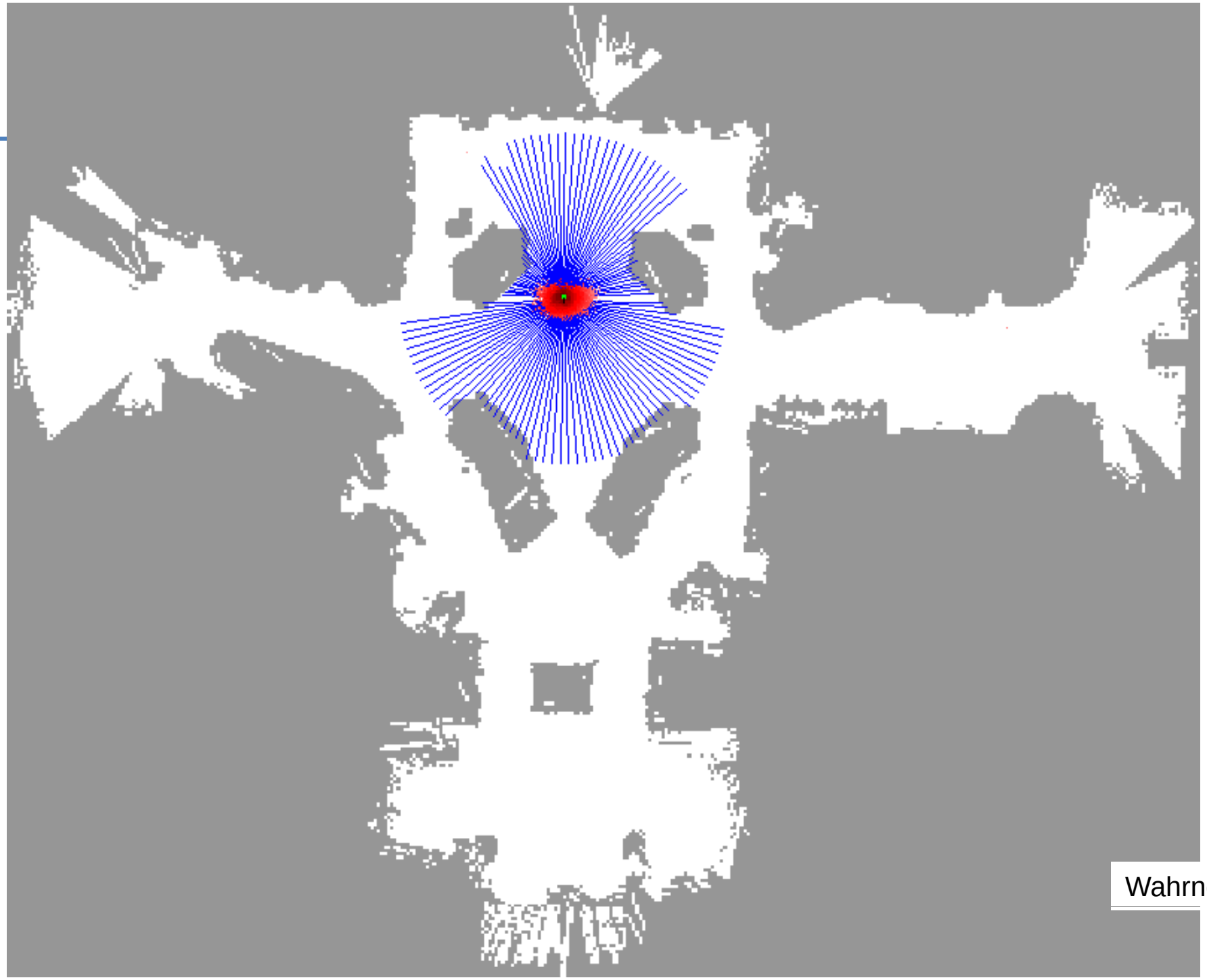
[TBF05]

Resampling



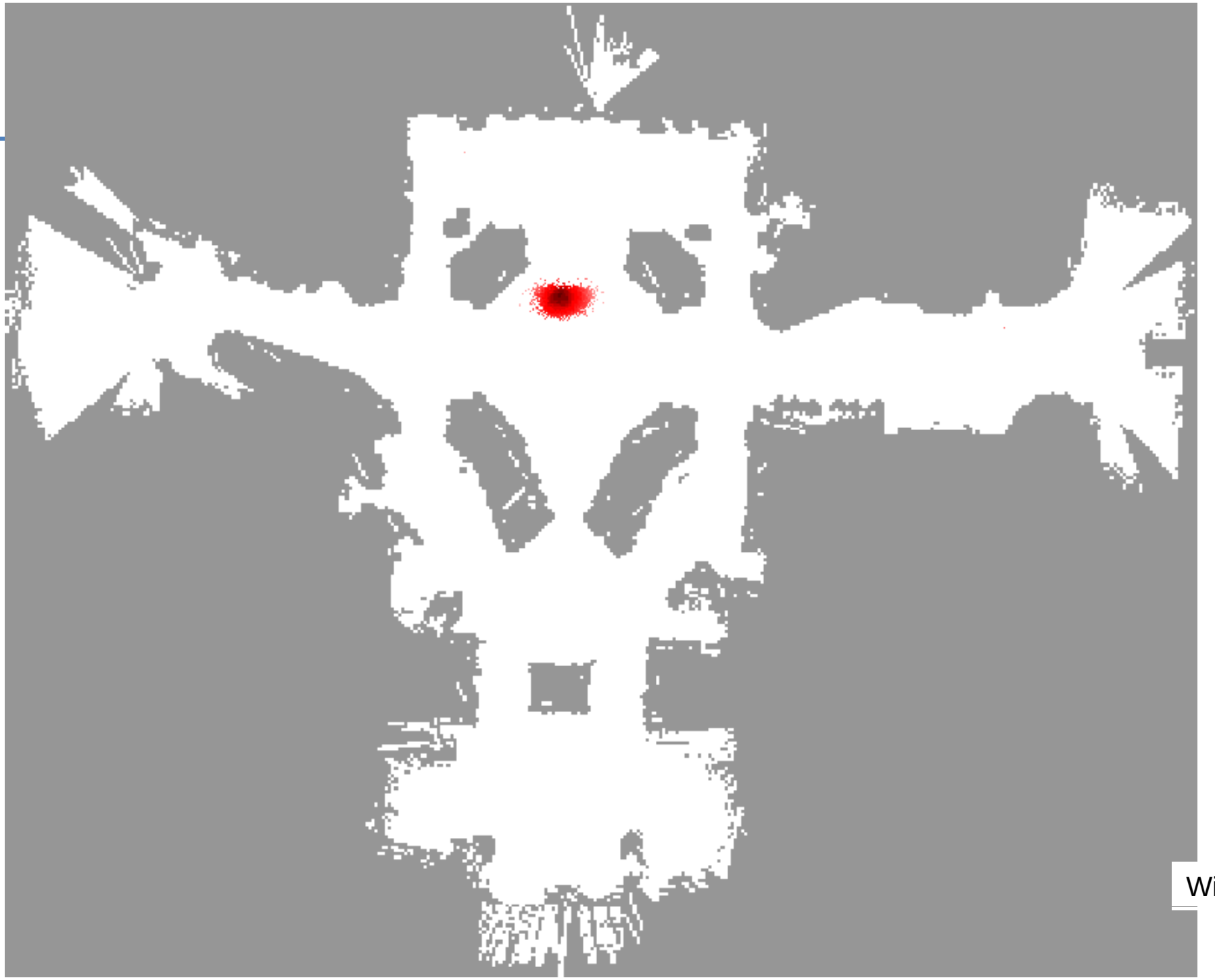
[TBF05]

Bewegungsprognose



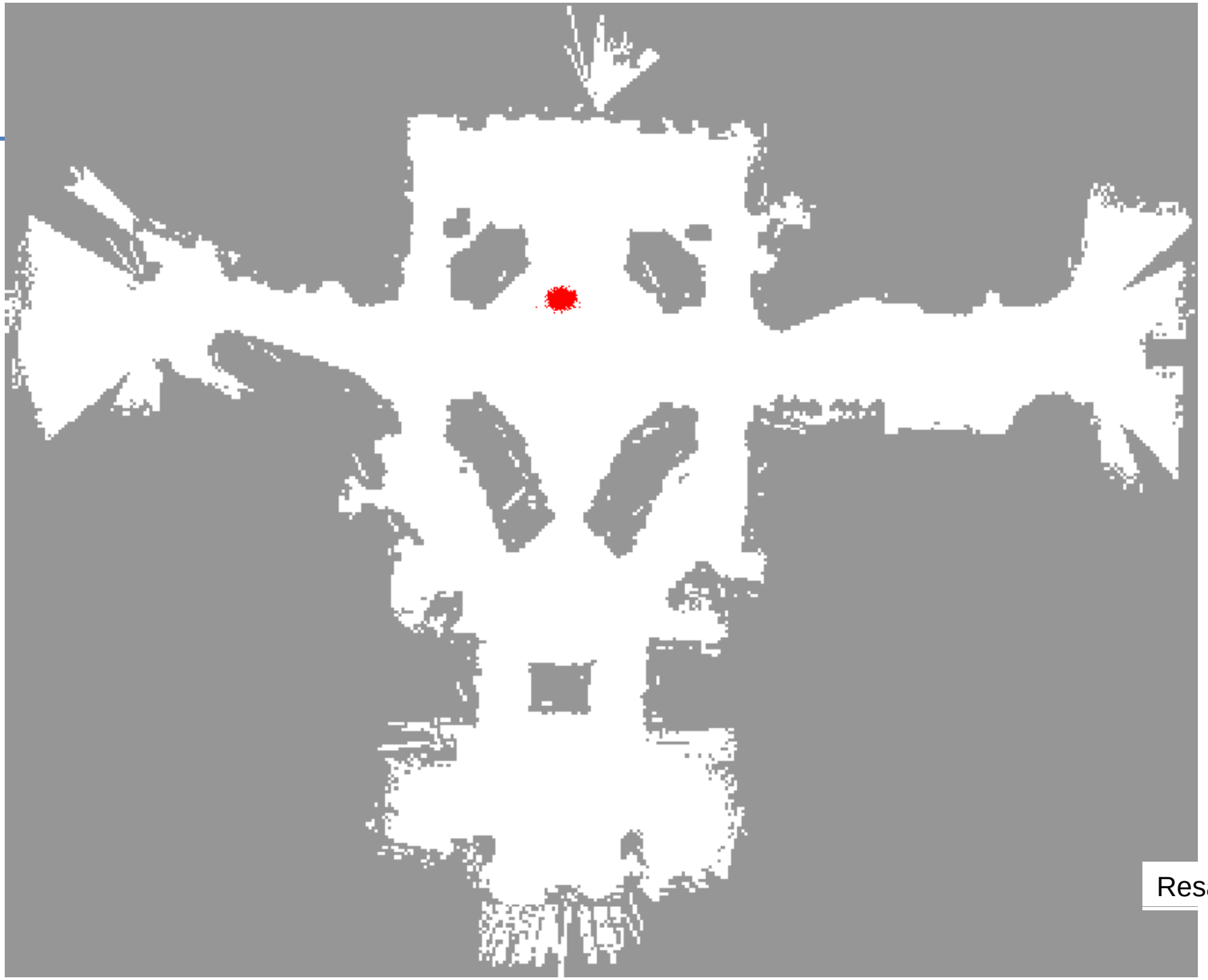
Wahrnehmung

[TBF05]



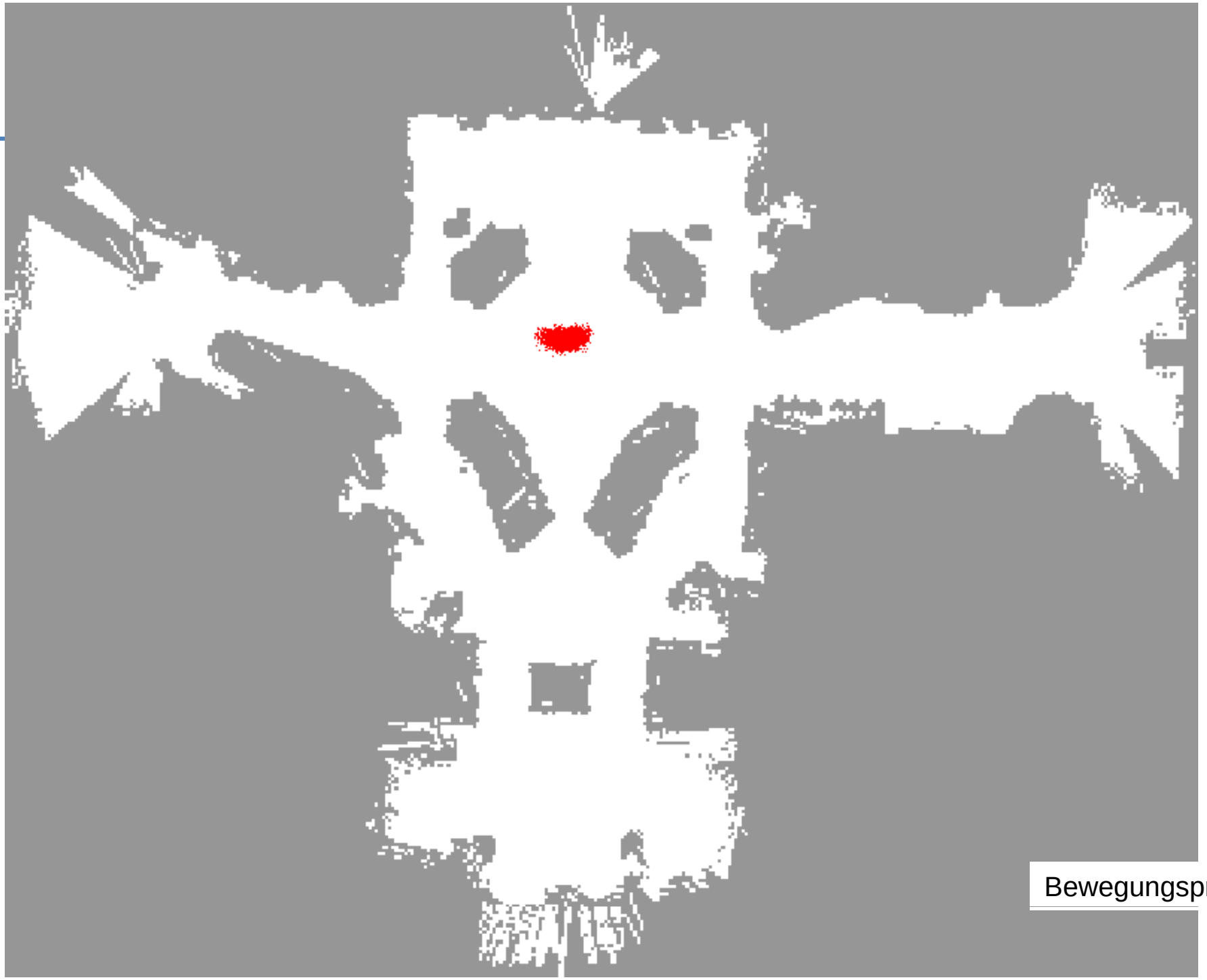
[TBF05]

Wichtung



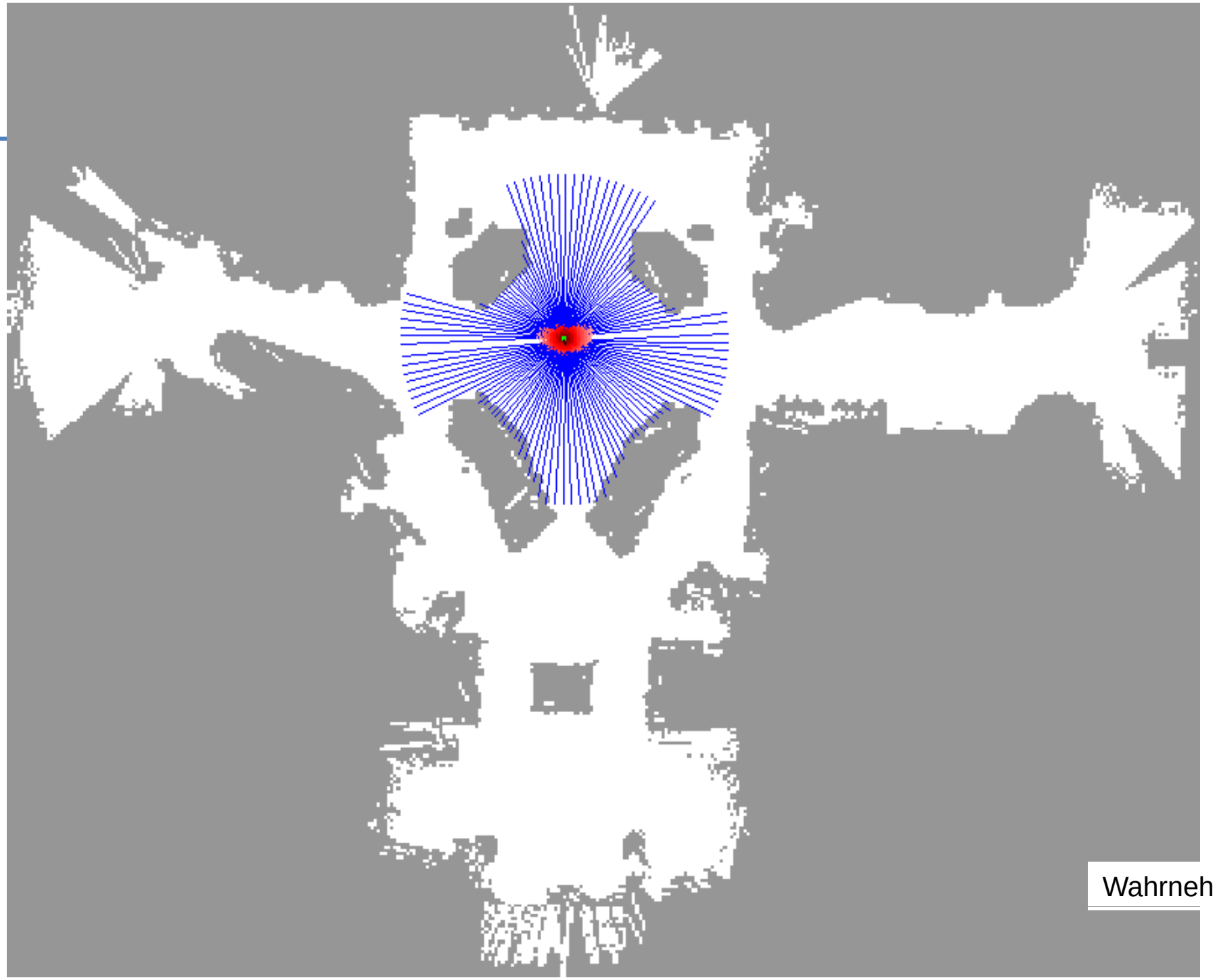
[TBF05]

Resampling



[TBF05]

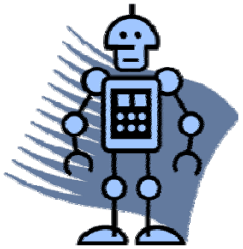
Bewegungsprognose



[TBF05]

Wahrnehmung...

Zusammenfassung



Markov-Lokalisierung

- Grundlage Bayes-Regel, Markov-Annahme
- **nicht nur für Lokalisierung**, auch für Zustandschätzung anderer Art, bspw. in der Medizin
- KI-Methode Bayessche Netze (Master)
- Trennung in Bewegungsmodell und Sensormodell
- Sensormodell nicht auf Abstandssensorik beschränkt, möglich bspw. Bild der Raumdecke, Erschütterungsprofil bei der Fahrt

Monte-Carlo-Lokalisierung

- MCL erlaubt elegante Fusion der Daten verschiedener Sensorik
- MCL skalierbar anhand der Partikelanzahl
- MCL effizient, bspw. dynamische Anpassung der Partikelanzahl (adaptiv)

Ziel der Übung

Funktioniert die Lokalisierung in ARNL?

Wie gut? Und wenn nicht?

Wie verhält sich die Partikelwolke, wenn der Roboter ohne Feedback aus der Umwelt fährt?