School of Engineering

Lab 9: Tracking eines Roboters mit Differentialantrieb

1. Einleitung

Mobile Roboter mit Differentialantrieb (differential wheeled robots) bestehen aus zwei separat angetriebenen Rädern, die auf einer Achse aufgehängt sind. Derartige Roboter kommen in verschiedensten Gebieten zum Einsatz, z.B. als Putzroboter.

Abbildung 1: Putzroboter, welcher mittels SLAM (simultaneous localization and mapping) navigiert.

Durch Kenntnis der Radabstands L und des Radius R der Räder können Motorregler die Geschwindigkeit und Drehrate der Plattform einstellen.

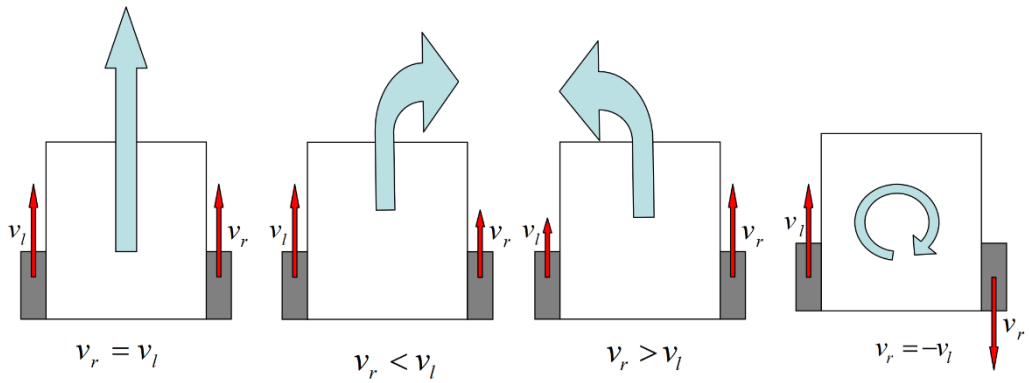

Abbildung 2: Indem man die Bewegung der Räder steuert, kann man die externe Bewegung des Fahrzeugs beeinflussen.

Für eine zielgerichtete Navigation ist es für den Roboter essenziell seine Position $[x, y]$ and seine Orientierung (in 2D Heading genannt) zu kennen. Das Heading beschreibt «wohin die Nase zeigt». Wir modellieren den Zustand des dynamischen Systems (Zustandsraummodell) daher als

$$\underline{x} = \begin{pmatrix} x_k \\ y_k \\ \phi_k \end{pmatrix},$$

wobei x_k und y_k die Verschiebung zum Ursprung angeben und der Winkel ϕ_k das Heading.

Die Geschwindigkeitsnorm (engl. speed) v und die Winkelgeschwindigkeit $\omega = \omega_z$ sind Kontrollinputs, also Inputs, die den Zustand $\underline{x}$ (Position und Heading) des Systems beeinflussen und bis auf eine gewisse Unsicherheit (z.B. Aufgrund der Ungenauigkeit der Modellparameter L und R oder des Motorreglers) bekannt sind. Der vektorwertige Input lautet:

$$\underline{u} = \begin{pmatrix} v \\ \omega \end{pmatrix}.$$

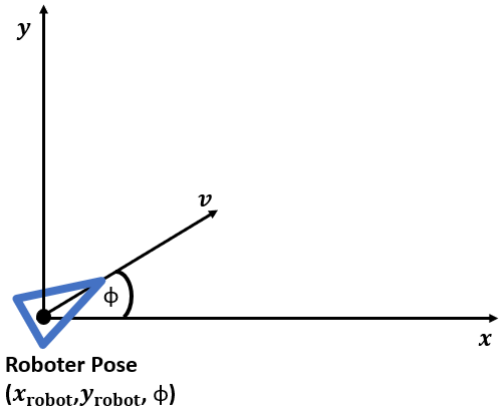

Abbildung 3: Roboterpose, welche aus der Position in 2D und dem Heading besteht.

Dead Reckoning (Koppelnavigation)

Falls der Anfangszustand, also die Anfangsposition und die Anfangsorientierung (perfekt) bekannt sind, könnte man theoretisch durch Integration der Geschwindigkeit und Winkelgeschwindigkeit den Zustand des mobilen Roboters bestimmen (dead reckoning). Dies funktioniert in der Praxis jedoch nur für sehr kurze Zeit, da die Position und die Orientierung aufgrund von Fehlern in den Kontrollsignalen, die aufintegriert werden, «wegdriften».

Orientierung und Positionsmessungen

Als Abhilfe bedient man sich **Distanz-** und **Winkelmessungen** anhand verschiedener Sensoren, z.B. Ultraschall Time-of-Flight oder LiDAR. Im Folgenden nehmen wir an, dass wir durch derartige Sensoren verrauschte Distanz und Winkelmessungen zu einem fixen Landmark mit der Position $p_L = (x_L, y_L)$ erhalten.

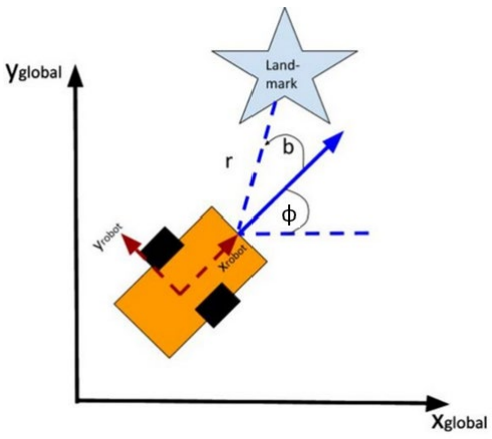

Abbildung 4: Distanz und Peilung.

Aufgaben

1. Verifizieren Sie anhand der Skizze in Abbildung 3, dass die Differentialgleichungen der Bewegung des Roboters wie folgt lauten:

$$\begin{aligned}\dot{x}(t) &= v(t) \cdot \cos(\phi(t)) \\ \dot{y}(t) &= v(t) \cdot \sin(\phi(t)) \\ \dot{\phi}(t) &= \omega(t)\end{aligned}$$

   Hier stellt die skalare Grösse $v(t) = \|\underline{v}(t)\|$ die Geschwindigkeitsnorm (engl. speed) dar.

2. Leiten Sie die zeitdiskrete (mit Samplingperiode Ts) Bewegungsgleichung her.

$V_k = \sqrt{\dot{x}_k + \dot{y}_k}$

$$\underline{\boldsymbol{x}}_{k+1} = \begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \phi_{k+1} \end{pmatrix} = \cdots = \underline{\boldsymbol{f}}(x_k, y_k, \phi_k, v_k, \omega_k) = \underline{\boldsymbol{f}}\left(\underline{\boldsymbol{x}}_k, \underline{u}_k\right)$$

$$\underline{x}_{k+1} = \begin{pmatrix} x_k + v_k \cdot \cos(\phi_k) \cdot T_s \\ y_k + v_k \sin(\phi_k) \cdot T_s \\ \phi_k + \omega_k \cdot T_s \end{pmatrix}$$

3. Bestimmen Sie die Jacobimatrix F an der Stelle $\underline{\boldsymbol{x}}_k$

$$F = \frac{\underline{\boldsymbol{f}}\left(\underline{\boldsymbol{x}}_k, \underline{u}_k\right)}{\partial \underline{\boldsymbol{x}}_k} = \begin{pmatrix} \frac{\partial \underline{\boldsymbol{f}}(\underline{\boldsymbol{x}}_k)}{\partial x_k} & \frac{\partial \underline{\boldsymbol{f}}(\underline{\boldsymbol{x}}_k)}{\partial y_k} & \frac{\partial \underline{\boldsymbol{f}}(\underline{\boldsymbol{x}}_k)}{\partial \phi_k} \end{pmatrix}$$

$$F = \begin{pmatrix} 1 & 0 & -v_k \cdot \sin(\phi_k) \cdot T_s \\ 0 & 1 & v_k \cdot \sin(\phi_k) \cdot T_s \\ 0 & 0 & 1 \end{pmatrix}$$

4. Betrachten Sie Abbildung 4. Das zeitdiskrete Beobachtungsmodell lautet

$$\boldsymbol{h}(\underline{\boldsymbol{x}}_k) = \begin{pmatrix} \sqrt{(x_L - x_k)^2 + (y_L - y_k)^2} \\ \text{atan2}(y_L - y_k, x_L - x_k) - \phi_k \end{pmatrix} = \begin{pmatrix} r_k \\ b_k \end{pmatrix},$$

wobei r_k die Distanz darstellt. Den zum Landmark $p_L = (x_L, y_L)$ = [5,5] (in Meter) gemessenen Winkel (de: relative Peilung und engl. relative bearing) bezeichnen wir als b_k. Beispielsweise würde $b_k = 0$ bedeuten, dass die Landmarke auf dem aktuellen Kurs des Roboters liegt.

Bestimmen Sie die Jacobimatrix z.B. mit Mathematica oder der abgespeckten online Version davon https://www.wolframalpha.com

$$H = \frac{\partial \boldsymbol{h}(\underline{\boldsymbol{x}}_k)}{\partial \underline{\boldsymbol{x}}_k}$$

$$H = \begin{pmatrix} \frac{x_L - x_k}{r_k} & \frac{y_L - y_k}{r_k} & 0 \\ \frac{y_L - y_k}{(x_L - x_k)^2 + (y_L + y_k)^2} & \frac{x_L - x_k}{(x_L + x_k)^2 + (y_L + y_k)^2} & -1 \end{pmatrix}$$

5. Implementieren Sie einen **extended Kalman Filter,** um die Position und Orientierung (den Zustand) des mobilen Roboters zu tracken. Wenden Sie ihren Algorithmus auf die zur Verfügung gestellten Daten an. Diese liegen als numpy npz-File vor: `mobile_robot_measurements_with_state.npz`

6. Welche Bewegung führt der Roboter aus?