Regelung eines Differentialantriebes

Richard Schroedter

2010

Inhaltsverzeichnis

1 Steuerung

1.1 Trajektoriengenerator

1.1.1 für lineare Strecken

1.1.1.1 lineare Rampe

Es wird eine lineare Rampe für die Anfangswerte: Anfangsgeschwindigkeit v_0, Endgschwindigkeit v_1 und Maximalgeschwindigkeit v_{max} sowie Startbesschleunigung a_1 und Bremsbeschleunigung a_2 gesucht.
Die folgenden Gleichungen beschreiben die Trajektorie:

Weg:

$$s = sign(s) \begin{cases} s_0 + v_0 t + \frac{a_1}{2} t^2 & \text{für} \quad t < t_1 \\ s_0 + v_0 t_1 + \frac{a_1}{2} t_1^2 + v_{max}(t - t_1) & \text{für} \quad t_1 < t < t_2 \\ s_0 + v_0 t_1 + \frac{a_1}{2} t_1^2 + v_{max}(t - t_1) + \frac{a_2}{2}(t - t_3)^2 & \text{für} \quad t_2 < t < t_3 \end{cases} \tag{1.1}$$

Geschwindigkeit:

$$v = sign(s) \begin{cases} v_0 + a_1 t & \text{für} \quad t < t_1 \\ v_{max} & \text{für} \quad t_1 < t < t_2 \\ v_{max} + a_2(t - t_2) & \text{für} \quad t_2 < t < t_3 \end{cases} \tag{1.2}$$

Beschleunigung:

$$a = sign(s) \begin{cases} a_1 & \text{für} \quad t < t_1 \\ 0 & \text{für} \quad t_1 < t < t_2 \\ a_2 & \text{für} \quad t_2 < t < t_3 \end{cases} \tag{1.3}$$

wobei s_0 der Anfangsweg ist und gilt: $a_2 < 0$. Zur Berechnung Streckenzeiten t_1, t_2, t_3 werden die Gleichungen ineinander eingesetzt.

$$t_1 = \frac{v_{max} - v_0}{a_1} \tag{1.4}$$

$$t_2 = \frac{(a_2 + a_1)v_{max}^2 - 2a_2 v_0 v_{max} - a_1 v_1^2 + a_2 v_0^2 + 2a_1 a_2 |d|}{2a_1 a_2 v_{max}} \tag{1.5}$$

$$t_3 = \frac{(a_2 - a_1)v_{max}^2 + 2(a_1 v_1 - a_2 v_0)v_{max} - a_1 v_1^2 + a_2 |v_0| + 2a_1 a_2 |d|}{2a_1 a_2 v_{max}} \tag{1.6}$$

Zur Überprüfung, ob die Maximalgeschwindigkeit erreicht wird, kann der Schnittpunkt der beiden Anstiegsgeraden verwendet werden. Dazu wird geprüft, ob gilt:

$$v_{lim} = \sqrt{\frac{a_1 v_1^2 - a_2 v_0^2 - 2a_1 a_2 |d|}{a_1 - a_2}} > v_{max} \tag{1.7}$$

Da nicht trapezförmige Rampen ohne konst. Geschwindigkeit den doppelten Sprung in der Beschleunigung verursachen, wird durch Herabsenken auf ca. 70% der Maximalgeschwindigkeit v_{max} stets eine trapezförmige Rampe berechnet.

2 Regelung

Die Regelung besteht aus einer 3-stufigen Kaskaden-Regelung. Geregelt werden Kraft mit einem Deadbeat-Regler und Geschwindigkeit sowie Position mit einem allgemeinen PID-Regler als $\mathcal{Z}$-Transformierte. In der Realität hat sich außerdem eine Vorsteuerung bis zu 100 % als günstig erwiesen. Die Schnelligkeit der gesamten Kaskade kann dadurch deutlich erhöht werden.

2.1 Deadbeat-Regler

Da die Umrechnung zwischen der Kraft die auf das Fahrzeug über die Räder wirkt und den Motorstrom linear ineinander umgerechnet werden können, kann anstatt eines Stromreglers auch ein Kraftregler implementiert werden. Die Kraft als Regelgröße ist für diese Anwendung sinnvoll und anschaulicher. Hierfür wird der Motorstrom gemessen. Der Deadbeat-Regler ist ein Regler für zeitdiskrete Systeme und garantiert, dass in einer endlichen Anzahl von Abtastschritten der Sollwert erreicht wird. In [Froe08] wurde ein solcher Regler und Vorfilter hergeleitet. Die Vorfilter-Übertragungsfunktion liegt näherungsweise bei eins wenn das Verhältnis von Induktivität L und Widerstand R_M des Motors sehr viel kleiner ist als die Abtastzeit T_A und kann vernachlässigt werden.

$$\frac{L}{R_M} << T_A \tag{2.1}$$

Die Reglerfunktion in der $\mathcal{Z}$-Ebene lautet:

$$D(z) = \frac{R_M}{1 - z^{-1}} \tag{2.2}$$

Der Ausgang des Reglers ist also direkt abhängig vom aktuellen Motorwiderstand. Für den Maxon-Motor RE30 wurde daher die nichtlineare Widerstands-Strom-Kennlinie aufgenommen.

Die Testergebnisse haben für den Deadbeat-Regler im Vergleich zum einfachen, und für Stromregler üblichen, I-Regler bessere Ergebnisse erzielt. Im Test wurde die Ausregelzeit bei verschiedenen Lastsprüngen verglichen.

2.2 Digitaler PID-Regler in $\mathcal{Z}$-Ebene

Es soll ein digitaler PID-Regler der Form

$$G_z(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{a_0 + a_1 z^{-1} + a_2 z^{-2}}$$

entworfen werden. Dazu wird zunächst die Übertragungsfunktion eines allgemeinen PID-Reglers als Laplace-Transformierte herangezogen

$$C(s) = K_R \left(1 + \frac{1}{T_I s} + T_D s\right)$$

wobei K_R der proportionelle, T_I der integrale und T_D der differetielle Verstäkungsfaktor ist. Nach [1] wird das s im Integralanteil mit der Tustin-Formel

$$s \approx \frac{2}{T_A} \frac{z-1}{z+1}$$

substituiert. Das s im Differentialanteil wird hingegen mit

$$s \approx \frac{z-1}{T_A z}$$

ersetzt. T_A entspricht hierbei der Abtastzeit des Analog-Digital-Wandler. Dadurch entsteht der diskretisierte PID-Regler

$$G_z(z) = \frac{K_R\left(1+\frac{T_A}{2T_I}+\frac{T_D}{T_A}\right)+K_R\left(-1+\frac{T_A}{2T_I}-2\frac{T_D}{T_A}\right)z^{-1}+K_R\left(\frac{T_D}{T_A}\right)z^{-2}}{1-z^{-1}}$$

der Form

$$G_z(z) = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{a_0+a_1z^{-1}+a_2z^{-2}}$$

mit den Koeffizienten

$$\begin{aligned} a_0 &= 1 \\ a_1 &= -1 \\ a_2 &= 0 \\ b_0 &= K_R\left(1+\frac{T_A}{2T_I}+\frac{T_D}{T_A}\right) \\ b_1 &= K_R\left(-1+\frac{T_A}{2T_I}-2\frac{T_D}{T_A}\right) \\ b_2 &= K_R\left(\frac{T_D}{T_A}\right) \end{aligned}$$

Nun kann man die Übertragungsfunktion $G_z(z) = \frac{U_z(z)}{E_z(z)}$ als zeitdiskrete Funktion in Abhängigkeit des Schritts k schreiben. Man erhält mit der $\mathcal{Z}$-Rücktransformation:

$$a_0u[k]+a_1u[k-1]+a_2u[k-2] = b_0e[k]+b_1e[k-1]+b_2e[k-2]$$

Umstellen nach $u[k]$ liefert den gesuchten Ausgang mitdem man in jedem Abtastschritt den Regelausgang u berechnen kann.

$$u[k] = \frac{b_0}{a_0}e[k]+\frac{b_1}{a_0}e[k-1]+\frac{b_2}{a_0}e[k-2]-\frac{a_1}{a_0}u[k-1]-\frac{a_2}{a_0}u[k-2]$$

Durch Einsetzen der Koeffizienten a_0, a_1, a_2 vereinfacht sich die Formel zu

$$u[k] = b_0e[k]+b_1e[k-1]+b_2e[k-2]+u[k-1]$$

2.3 Reglerkaskade

Für die Reglerkaskade werden nun die mathematischen Beziehungen benötigt. Regelt man die Motorspannung mittels PWM-Signal so erhält man bei Berechnung des Effektivwertes U_A aus der Versorgungsspannung U_{CC} die Gleichung

$$U_A = \frac{1}{T}\int_0^{T_{an}} U_{CC}dt = DU_{CC}$$

wobei T die Periode, T_{an} die Anschaltzeit und $D = \frac{T_{an}}{T}$ das Tastverhältnis des periodischen Rechtecksignals sind. Das gewünschte Tastverhältnis lässt sich also mit

$$D = \frac{U_A}{U_{CC}}$$

berechnen.
Für die Ankerspannung U_A wiederum gilt

$$U_A = (R_M + Ls)I_A$$

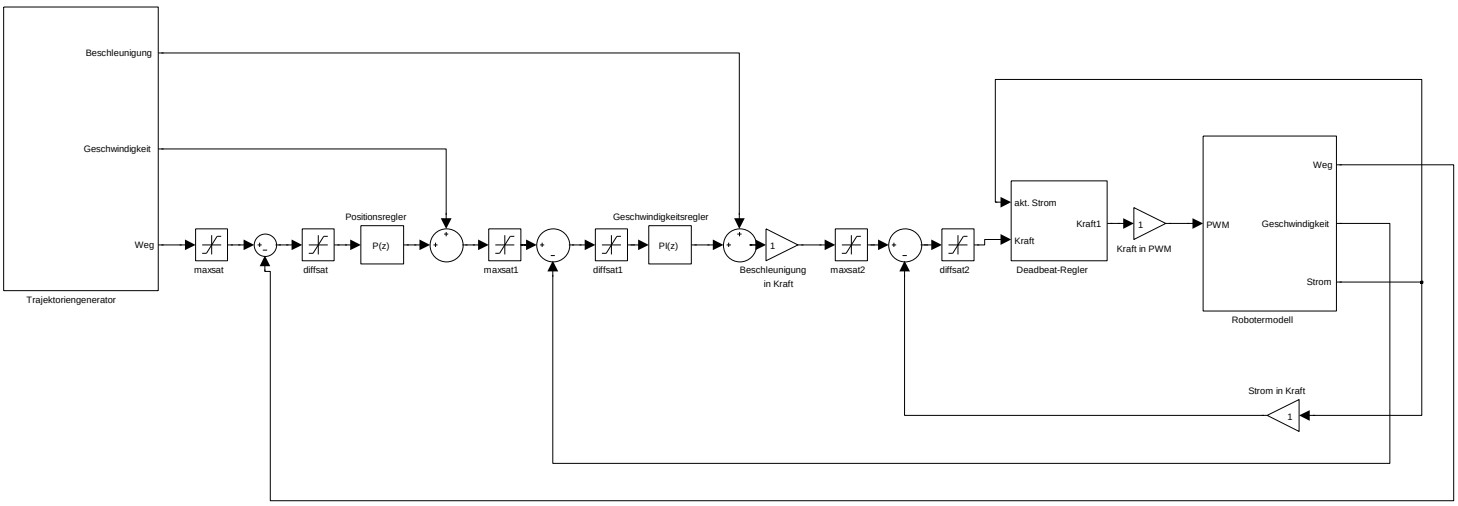
und für das Motormoment

$$M = k_M I_A$$

sowie

$$M = m\frac{v}{\Delta t}r_{Rad} \quad \text{für Translation}$$

$$M = \frac{J}{R^2_{\text{mittl. Trägheit}}}\frac{v}{\Delta t}r_{Rad} \quad \text{für Rotation}$$


Beschleunigung
Geschwindigkeit
Weg
Trajektoriengenerator
maxsat
diffsat
Positionsregler
P(z)
maxsat1
diffsat1
Geschwindigkeitsregler
PI(z)
1
Beschleunigung
in Kraft
maxsat2
diffsat2
akt. Strom
Kraft
Kraft1
Deadbeat-Regler
1
Kraft in PWM
PWM
Weg
Geschwindigkeit
Strom
Robotermodell
Strom in Kraft
1

2.4 Splines

3 Implementierung

3.1 motion_control.c

Die *motion_control.c* startet die drei Hauptthreads. Darin werden jeweils in einer while(1)-Schleife die Funktionen *do_control*,*do_odometry* alle 10 ms und die *do_communication* alle 30 ms ausgeführt. Während die *do_odometry* alle Odometrie-Messwerte auswertet, werden in der *do_control* wird die Steuerung und Regelung des Roboters realisiert. Der *communication*-Thread bildet die Kommunikationsschnittstelle mit der System Control und anderen Einheiten, d. h. in der *do_communication* werden Befehle über CAN-Bus gesendet und empfangen.

3.2 control.c

Die *control.c* beinhaltet Steuerung und Regelung des Roboters. Sie nimmt die Fahrbefehle mit der *drive*-Funktion an, berechnet daraus die neuen Stellwerte. Diese werden in die Regelung in der *feedback_control_cascade.c* gegeben. Die berechneten PWM-Werte gehen schließlich an den Motortreiber. Es stehen folgende Fahrbefehle zur Verfügung:

- CONTROL_DISTANCE: Eine Strecke fahren. Dabei wird auf die Distanz und Winkel Null geregelt.
- CONTROL_ANGLE: Einen Winkel fahren. Dabei wird auf die Winkel und Distanz Null geregelt.
- CONTROL_TRANSLATION: Eine bestimmt Strecke mit gegebener Maximalgeschwindigkeit fahren. Dabei wird nur auch die translatorische Geschwindigkeit geregelt.
- CONTROL_ROTATION: Eine bestimmten Winkel mit gegebener Maximalgeschwindigkeit fahren. Dabei wird nur auch die rotatorische Geschwindigkeit geregelt.
- CONTROL_FORCE: Eine bestimmte Kraft in translatorische Richtung aufbringen.
- CONTROL_TORQUE: Eine bestimmte Kraft in rotatorische Richtung aufbringen.
- CONTROL_STOP: Aus gegebener Geschwindigkeit so schnell wie möglich bremsen. Hierbei die kürzeste Trajektorie für Translation und Rotation berechnet, die zum Halt führt.
- CONTROL_HOLD_POSITION: Eine Position halten. Hierbei wird auf aktuelle Distanz und Winkel geregelt.
- CONTROL_LINE: Mit dem nichtlinearen Ansatz von Kanayama-Kimura auf eine Gerade/Strecke regeln.
- CONTROL_ARC: Einen Kreisbogen fahren.
- CONTROL_SPLINE: Mit dem nichtlinearen Ansatz von Kanayama-Kimura auf einen Spline regeln.

- CONTROL_VELOCITY: Eine bestimmte Geschwindigkeit fahren. Achtung: Dieser Befehl ist ohne Beschleunigungs- und Bremsphase.
- CONTROL_PWM: Eine bestimmte Spannung am Motor anlegen.
- CONTROL_FLAT: Mit dem flachheitsbasierten Ansatz eine Trajektorie fahren.

Mit der *drive*-Funktion können diese Befehle und andere, die diese kombinieren, ausgeführt werden.

3.3 feedback_control_cascade.c

Die Regelung besteht aus einer 3-stufigen Kaskade mit 6 Reglern, also 2 Reglern pro Stufe, da translatorische und rotatorische Regler getrennt sind. Man kann Stellwerte für folgende Kaskadenstufen vorgeben:

- FEEDBACK_CONTROL_POSE: Positionsregelung auf Distanz und Winkel
- FEEDBACK_CONTROL_VELOCITY: Geschwindigkeitsregelung auf Translation und Rotation
- FEEDBACK_CONTROL_FORCE: Kraftregelung auf tangentiale Kraft und Moment

Dazu werden in der Funktion *fetch_desired_value* zunächst die Sollwerte für den Reglereingang und für die Vorsteuerung getrennt. Der Ausgang eines Regler ist bis auf Umrechnungen der Stellwert für die nächste Kaskade bzw. für das PWM-Signal des Motors. Als sichere Methode zum Abfangen von Programmierfehlern oder sonstigen Defekten hat sich die Begrenzung des Reglerfehlers (Reglereingang e) sowie die Begrenzung des Stellwertes erwiesen. Kommt ein Regler in die Begrenzung so wird auf einen PD-Regler ohne Integralanteil umgeschalten und eine Meldung an die Steuerung gegeben. Wenn ein Regler in einer inneren Kaskadenstufe begrenzt ist so müssen auch alle darüberliegenden Regler ihren Integralanteil deaktivieren.

Literaturverzeichnis

[1] Heinz Unbehauen: *Regelungstechnik*, volume II. Vieweg (2007)

[2] Norbert Fröhlich: *Studienarbeit*, (2008)