

Lenk-Regelung in der Robotik

von Prof. Dr.-Ing. Detlef Brumbi, Deggendorf

Zusammenfassung

- Für die Lenkregelung eines Fahrzeugs mit 2-Rad-Antrieb ist eine PD-Regelung die beste Lösung.
- Für die Linienerkennung sollte der Sensor einen monotonen Funktionszusammenhang zwischen Linienabweichung und Sensor-Ausgangsgröße besitzen.
- Ein optischer Liniensensor muss mit angemessenem Abstand vor den Antriebsrädern montiert werden.
- Die Regelung lässt sich software-gesteuert leicht realisieren.
- Mit Hilfe einer mathematischen Simulation lässt sich das Verhalten des Regelkreises am effektivsten analysieren.

1. Einleitung

Bei Fahr-Robotern – auch allgemein bei Fahrzeugen – besteht häufig die Notwendigkeit, einer Spur zu folgen oder Begrenzungen beim Fahren zu beachten, z.B.

- einen vorgegebenen Abstand zu einer Wand einzuhalten bzw. nicht zu unterschreiten;
- sich zwischen zwei Begrenzungslinien zu bewegen;
- entlang einer Leitlinie zu fahren.

Hierzu sind Lenk-Eingriffe des Fahrzeugs erforderlich, die bei einer Abweichung vom vorgesehenen Weg das Fahrzeug wieder in die „Spur“ bringen.

Soll sich der Fahr-Roboter autonom bewegen, sind geeignete Sensoren vorzusehen, deren Sensorwerte in einen Regelkreis für die Motor(en)-Bewegung einfließen, z.B.

- Ultraschall- oder LIDAR-Sensoren zur Messung des Abstands zu Begrenzungsobjekten oder einer Wand.
- Optische Reflexions-Sensoren zur Erkennung von Leitlinien auf dem Untergrund.

Die folgenden Abhandlungen beziehen sich auf einen Spiel-Roboter, der bei einem Wettbewerb, z.B. der World Robot Olympiad (WRO), eingesetzt werden kann, können aber verallgemeinert auch auf andere Szenarien übertragen werden.

Bei der WRO befinden sich auf dem Spielfeld verschiedene 2 cm breite, schwarze Linien, an denen sich der Roboter orientieren kann. Beispiel: Spielfeld für WRO RoboMission Senior 2023:

Im Prinzip funktioniert der Regelkreis so: Die Regelgröße wird durch den Sensor erfasst. Ergibt sich hierbei eine Abweichung, hat der Regler die Aufgabe, die Regelstrecke so zu beeinflussen, dass die Regelgröße sich dem gewünschten Wert angleicht. Der Einfluss von Störgrößen sollte gleichzeitig minimiert werden.

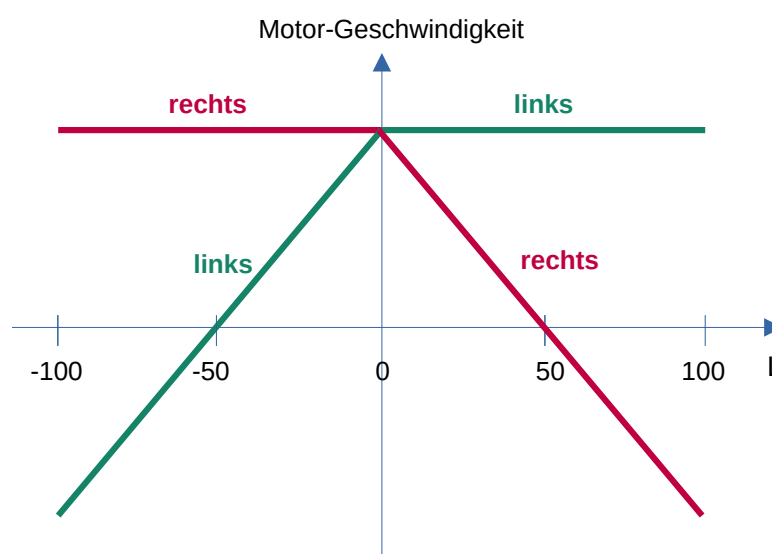
Besonderes Augenmerk ist dabei den drei eingezeichneten Blöcken zu widmen: der Regelstrecke, dem Sensor und dem Regler. Sie werden in den folgenden Abschnitten behandelt.

3. Regelstrecke

Hierunter ist in unserem Fall das System der Antriebsmotoren zu verstehen. Die dynamisch-mechanischen Eigenschaften der Motoren werden in dieser Abhandlung nicht berücksichtigt.

Die Steuerung der Lenkung ist bei einigen Robotik-Baukästen softwareseitig wie folgt festgelegt. Es gibt einen Lenkwert L im Intervall von -100 bis $+100$, der die beiden Antriebsmotoren mit unterschiedlichen Geschwindigkeiten steuert.

- $L = 0$: Roboter fährt geradeaus
- $L > 0$: Roboter fährt eine Rechtskurve
- $L < 0$: Roboter fährt eine Linkskurve
- $L = 50$: Das rechte Rad bleibt stehen und der Roboter dreht sich im Uhrzeigersinn
- $L = -50$: Das linke Rad bleibt stehen und der Roboter dreht sich gegen den Uhrzeigersinn
- $L = 100$: Das rechte Rad dreht nach hinten und das linke Rad nach vorn, so dass sich der Roboter um den Achsenmittelpunkt im Uhrzeigersinn dreht.
- $L = -100$: Das rechte Rad dreht nach vorn und das linke Rad nach hinten, so dass sich der Roboter um den Achsenmittelpunkt gegen den Uhrzeigersinn dreht.
- Zwischenwerte für L führen dazu, dass sich die Räder unterschiedlich schnell drehen und somit eine Kurve gefahren wird.



Für die Bewegungsrichtung der Räder gilt zusammengefasst:

Lenkwert L	Richtung linkes Rad	Richtung rechtes Rad	Drehrichtung Fahrzeug
-100 ... -51	Rückwärts	Vorwärts	Gegen Uhrzeigersinn
- 50	Steht	Vorwärts	Gegen Uhrzeigersinn
-49 ... -1	Vorwärts	Vorwärts	Gegen Uhrzeigersinn
0	Vorwärts	Vorwärts	Geradeaus
1 ... +49	Vorwärts	Vorwärts	Im Uhrzeigersinn
+50	Vorwärts	Steht	Im Uhrzeigersinn
+51 ... +100	Vorwärts	Rückwärts	Im Uhrzeigersinn

Der Lenkwert L wird dann als Parameter in der Roboter-Software eingegeben.

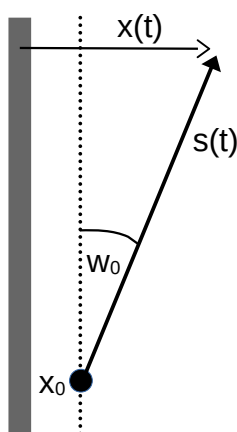
Die technische Charakteristik der Motoren, z.B. Trägheit, Reibung, Dynamik hat ebenfalls Einfluss auf die Regelstrecke, lässt sich aber nur sehr schwierig erfassen. Daher werden diese Faktoren nicht weiter berücksichtigt, zumal der Regler später noch die Aufgabe erhält, diese Einflüsse möglichst auszugleichen.

Dennoch sind die maximale Rotations-Geschwindigkeit und die maximale Beschleunigung durch das Motor-Drehmoment bei der Motoransteuerung einzuhalten. Schrittmotoren bieten gegenüber Gleichstrom-Getriebemotoren den Vorteil, dass sie sehr präzise den Steuerimpulsen folgen, solange das zulässige Drehmoment nicht überschritten wird.

In allen Fällen wird für die Lenk-Regelung hier einzig der Lenkwert L als Eingangsgröße für die Regelstrecke verwendet.

4. Charakteristik des Lenkverhaltens

In Bezug auf eine Linienverfolgung und deren Regelcharakteristik spielt das Bewegungsverhalten des Fahrzeugs durch Lenkeinflüsse eine entscheidende Rolle. Die prinzipielle Wirkung des Lenkwerts L wurde im Abschnitt 3 beschrieben. Im Folgenden werden nur kleine Lenkwerte ($|L| \leq 10$) betrachtet – mit dem Ziel, die Spur des Fahrzeugs einer Linienführung anzunähern.



Betrachtet wird die seitliche Abweichung x von der Linienmitte. Hat das Fahrzeug zum Beginn der Betrachtung ($t = 0$) eine Anfangs-Abweichung x_0 und eine Winkel-Abweichung w_0 , gilt:

$$x(t) = x_0 + s(t) \cdot \sin w_0 \approx x_0 + s(t) \cdot w_0 = x_0 + v_0 \cdot t \cdot w_0$$

Alle Winkel w sind in Radiant (Bogenmaß) einzugeben. Die Näherung gilt für kleine Winkel. s ist die zurückgelegte Fahrstrecke, v_0 die (konstante) Geschwindigkeit.

Ändert sich der Winkel w mit der Zeit t , ist statt dessen ein bestimmtes Integral zu bilden (u ersetzt die Integrationsvariable):

$$x(t) = x_0 + v_0 \cdot \int_0^t w(u) du$$

Führt der Roboter eine kurze Zeitdauer t_1 eine Lenkbewegung mit L_1 aus, ändert sich der Winkel um w_1 : (Gl.10 Roboter-Physik, Kapitel 3)

$$w_1 = \frac{v_0 \cdot t_1}{A} \cdot \frac{L}{50} \quad (\text{mit } A = \text{Achslänge} = \text{Radabstand})$$

Verallgemeinert mit einem zeitabhängigen Lenkwert $L(t)$ (v ersetzt die Integrationsvariable):

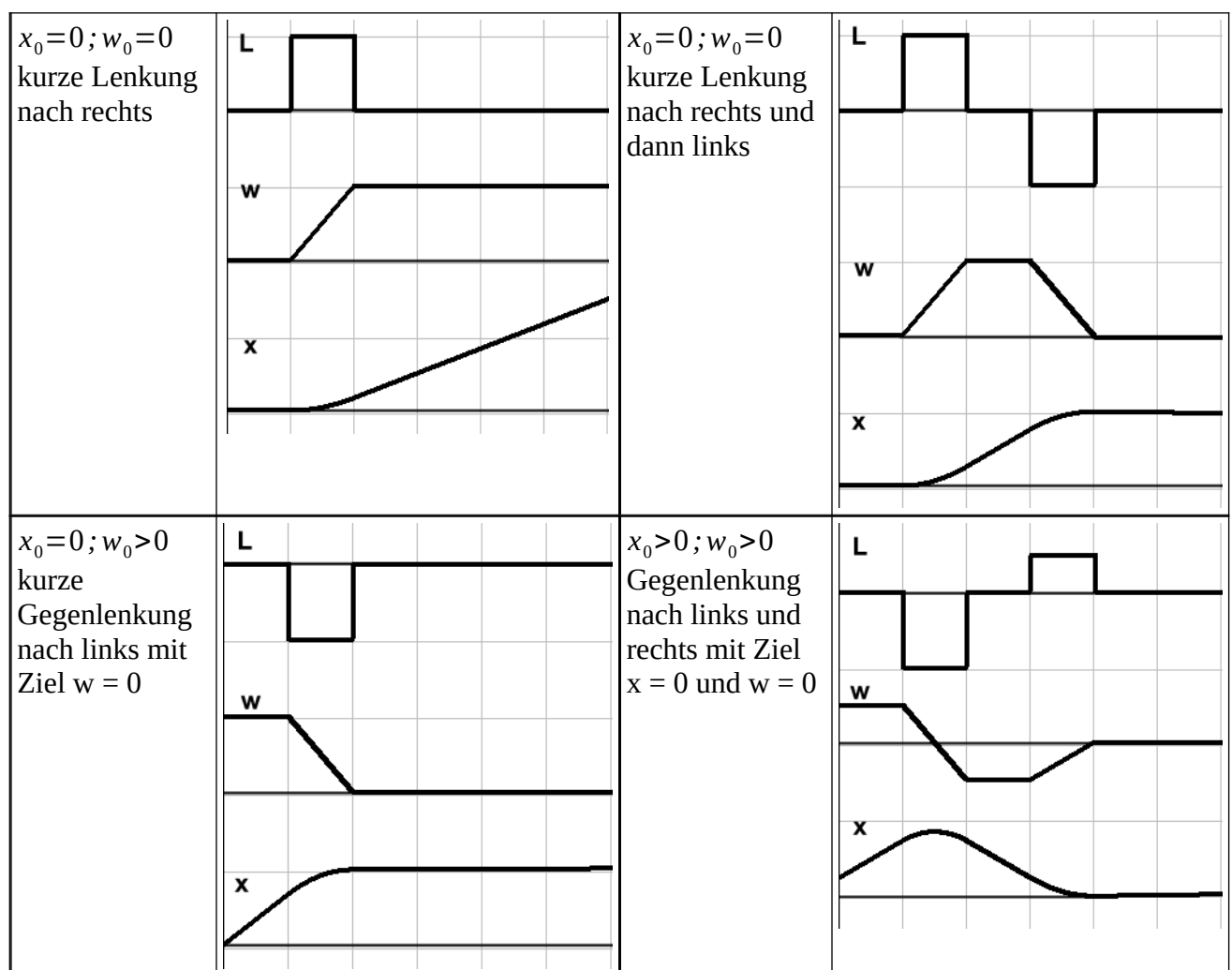
$$w_1(t) = \frac{v_0}{A \cdot 50} \cdot \int_0^t L(v) dv \quad \text{und} \quad w(t) = w_0 + w_1(t)$$

Damit:

$$x(t) = x_0 + w_0 \cdot v_0 \cdot t + \frac{v_0^2}{A \cdot 50} \int_0^t \int_0^u L(v) dv du$$

Fazit: Der Lenkwert L wird zweifach integriert !

Beispiele: ($w = 1.$ Integral , $x = 2.$ Integral)



5. Sensoren

Ultraschall-Sensor:

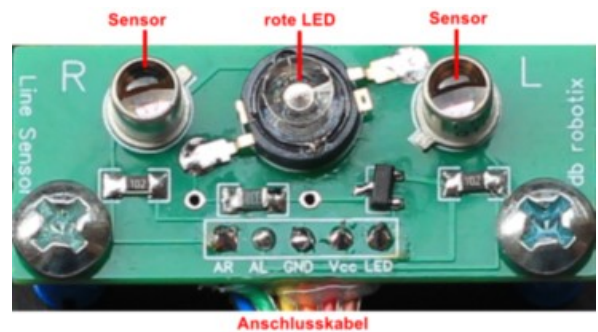
Er kann eingesetzt werden, um den Abstand zu einer Wand zu messen und ihm zu folgen (Wandverfolgung). Der Wandabstand a wird aus der Laufzeit t des Ultraschallsignals mittels Software

ermittelt: $a = \frac{t \cdot c}{2}$ mit der Schallgeschwindigkeit $c = 343 \text{ m/s}$.

In der Praxis ist das Ergebnis nicht sehr genau (typische Abweichung $\pm 5 \text{ mm}$), der praktische Messbereich ist auf etwa 50 cm begrenzt und Objekte, die sich im Ausbreitungsweg befinden, stören die Messung.

Linien-Sensor:

Um einer streifenförmigen Linie auf dem Fahr-Untergrund zu folgen, ist ein Linien-Sensor vorteilhaft, der mindestens zwei nebeneinander liegende Lichtsensoren sowie gegebenenfalls eine Lichtquelle besitzt, wie im nebenstehenden Bild (Quelle: db robotix).



Nehmen wir eine schwarze Linie auf weißem Untergrund an, sieht der Linien-Sensor je nach Position des Roboters diese Bilder.

Roboter befindet sich auf der Linien-Mitte:

Beide Sensoren erhalten gleich viel Licht, die Differenz ist Null.

Der Roboter braucht nicht zu lenken.

Roboter steht links von der Linie:

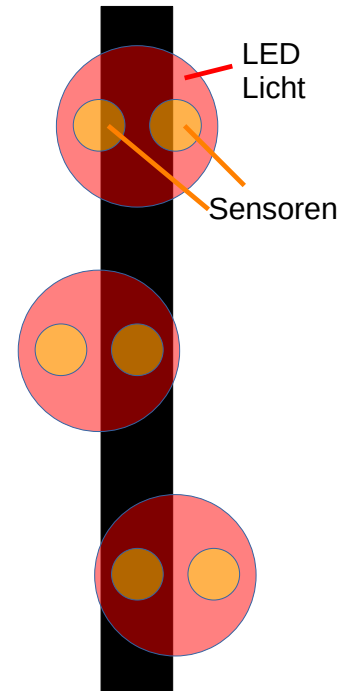
Der linke Sensor erhält mehr Licht als der rechte, die Differenz $L - R$ ist positiv.

Der Roboter muss nach rechts lenken.

Roboter steht rechts von der Linie:

Der rechte Sensor erhält mehr Licht als der linke, die Differenz $L - R$ ist negativ.

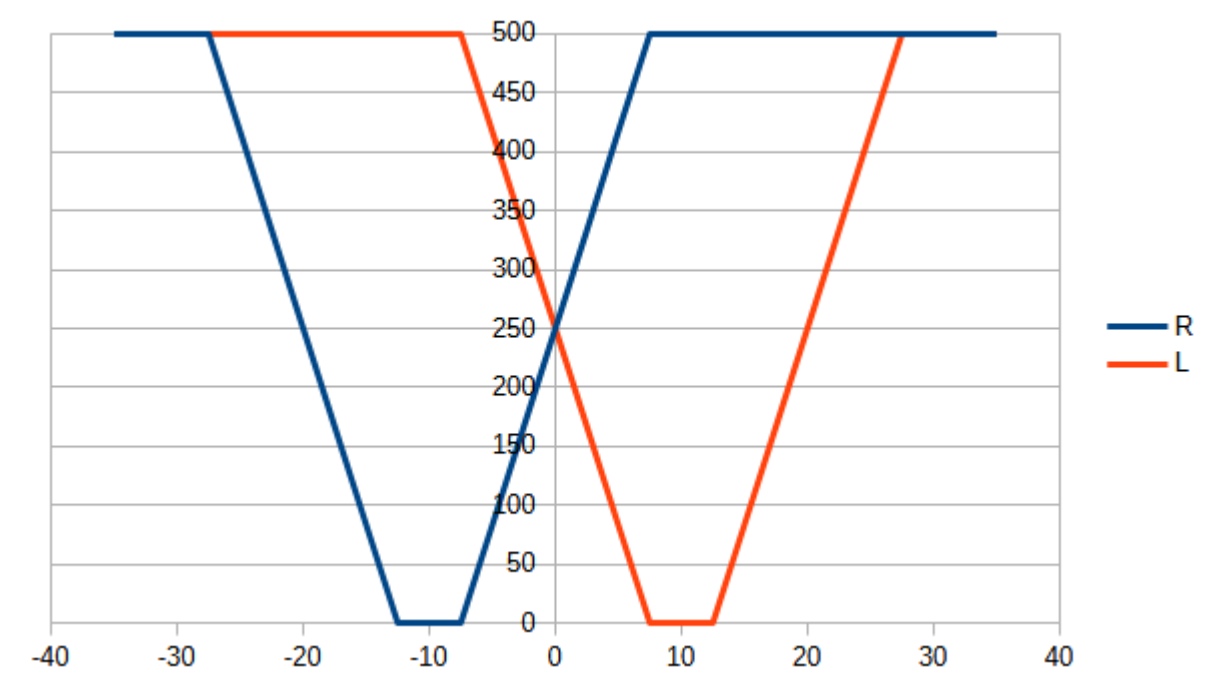
Der Roboter muss nach links lenken.



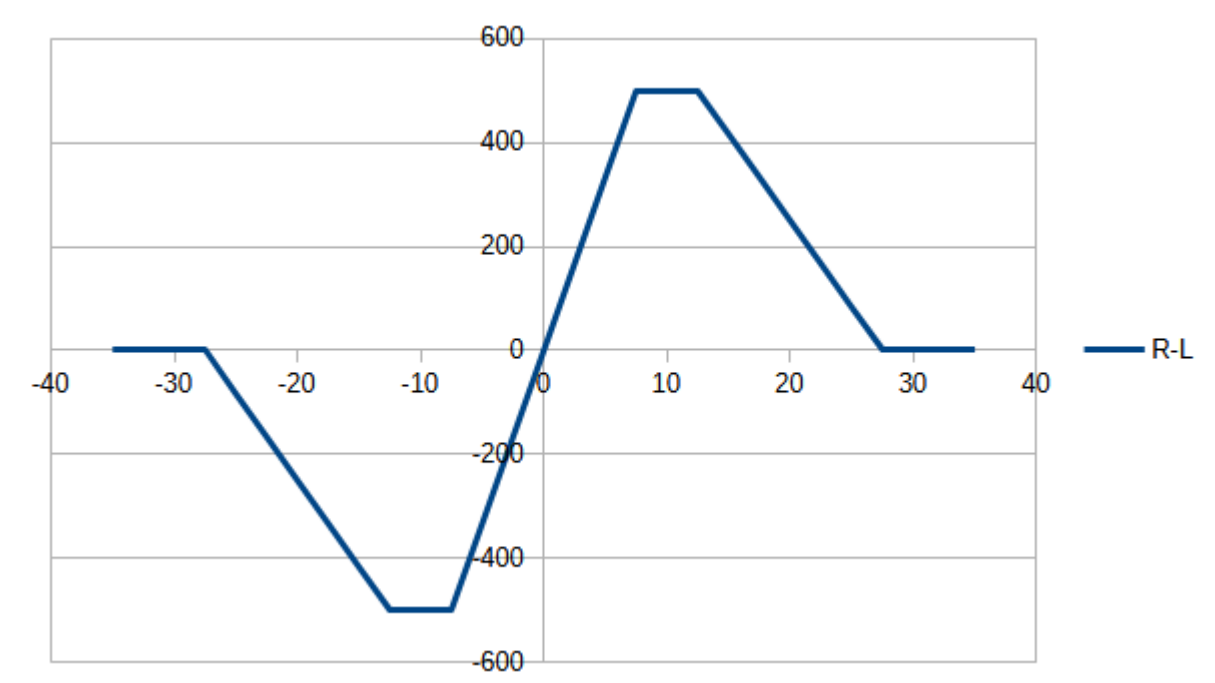
Für die Lenk-Aktion (Linienverfolgung) wird also die Lichtdifferenz der beiden Sensoren ausgewertet.

In den nachfolgenden Diagrammen gibt die horizontale Achse die Mittenabweichung x wieder, die vertikale Achse die Sensorwerte s .

Ist der Detektionsdurchmesser der Lichtsensoren etwas kleiner als die Linienbreite (wie im Bild oben), erhält man für die beiden Sensoren (links und rechts) etwa diese Funktionen in Abhängigkeit von der Spurabweichung (x-Achse), Linienbreite = 20:



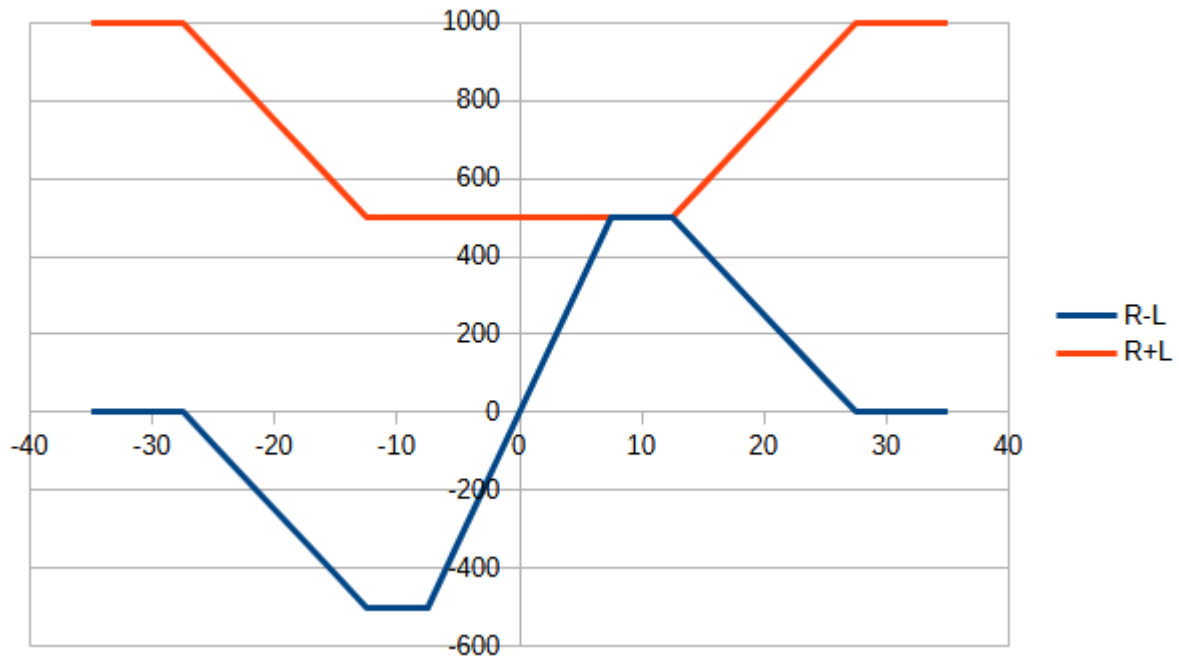
Durch Differenzbildung $R - L$ erhält man eine Funktion, die erkennen lässt, ob sich der Roboter links oder rechts von der Linienmitte ($x = 0$) befindet:



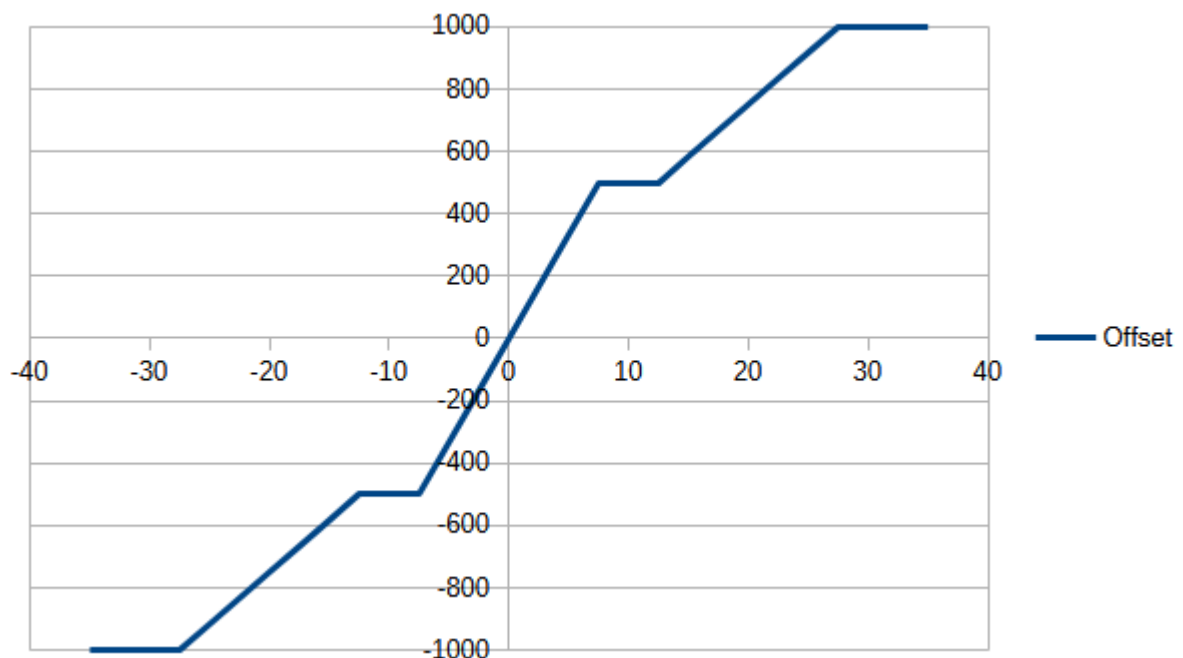
Der Betrag der Lichtdifferenz steigt zunächst mit zunehmendem Abstand von der Linie monoton an. Das ändert sich jedoch, sobald beide Sensoren die Linie verlassen. Befinden sich beide Sensoren über weißem Grund, wird die Lichtdifferenz wieder null. Dann lässt sich nur aus den Absolutwerten der Reflexionen unterscheiden, ob der Roboter genau auf der Linie oder vollständig außerhalb fährt.

Diese Nicht-Monotonie mit Steigungsumkehr führt sogar gegebenenfalls zu Problemen im Regelkreis (mit differentiellem Anteil).

Durch einen geschickten Algorithmus lässt sich die Sensorfunktion verbessern. Sobald sich ein Einzelsignal dem Maximum nähert, d.h. der entsprechende Sensor weiß sieht, wird das Ergebnis auf die Summe $R+L$ bzw. $-(R+L)$ umgeschaltet:



Das Resultat (hier Offset genannt) sieht dann so aus:

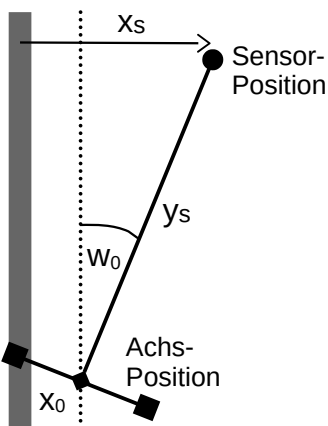


In den Randbereichen, wo beide Sensoren weiß sehen, muss die Historie zusätzlich berücksichtigt werden.

Beispiel für einen C-Algorithmus:

```
int16_t getOffset() {
    int16_t a1, a2;
    int32_t left, right, diff;
    getReflections(a1, a2);
    left = map(a1, blackL, whiteL, 0, 500); // normalize to 0...500
    left = constrain(left, 0, 500);          // limit to 0 ... 500
    right = map(a2, blackR, whiteR, 0, 500); // normalize to 0...500
    right = constrain(right, 0, 500);        // limit to 0 ... 500
    diff = right - left;                     // standard case
    if (left > 450 && right < 450)
        diff = -(left + right); // right sensor on left edge of line
    else if (left < 450 && right > 450)
        diff = left + right;    // left sensor on right edge of line
    else if (left > 450 && right > 450 && lastOffset > 0)
        diff = 1000;           // both sensors on white; consider history
    else if (left > 450 && right > 450 && lastOffset < 0)
        diff = -1000;          // both sensors on white; consider history
    lastOffset = diff;
    return (int16_t)diff;
}
```

Zunächst werden die beiden Sensorwerte $a1$ und $a2$ auf das Intervall $[0 ; 500]$ normiert. Hierzu ist eine einmalige Kalibrierung auf die Weißwerte $whiteL$ und $whiteR$, sowie auf die Schwarzwerte $blackL$ und $blackR$ erforderlich. Sobald einer oder beide Sensorwerte 450 (90% weiß) überschreitet, treten die Fallunterscheidungen in Kraft.



Der Liniensensor sollte in einem nicht zu kleinen Abstand y_s **vor** der Antriebsachse positioniert werden (s. Kapitel 10). Dadurch weicht die x-Sensorposition x_s von der x-Position des Roboters x_0 ab:

Für kleine Winkel (in Radiant) gilt: $x_s = x_0 + y_s \cdot \sin w_0 \approx x_0 + y_s \cdot w_0$

6. Regler

Dem Regler kommt aus verschiedenen Gründen eine bedeutende Rolle zu:

- Er ermöglicht überhaupt erst die Bildung eines Regelkreises.
- Seine Charakteristik beeinflusst das dynamische Verhalten des Regelkreises.
- Bei falscher Dimensionierung kann die Regelgröße schwingen, d.h. der Roboter fährt eine Schlangenlinie um die Bodenlinie.
- Er wirkt mit dem Sensor zusammen (s. Kapitel 7).

Aus der Literatur zur Regelungstechnik sind verschiedene Standard-Regler bekannt, die durch den mathematischen Zusammenhang zwischen Ausgangsgröße und Eingangsgröße des Reglers definiert sind:

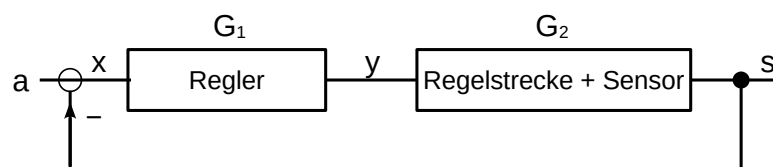
- P-Regler: Die Ausgangsgröße ergibt sich aus der Eingangsgröße, multipliziert mit einem **P**roportionalitätsfaktor, und stellt die einfachste Form eines Reglers dar.
- PD-Regler: Hierbei kommt zum **P**roportionalitätsfaktor noch die zeitliche Ableitung (**D**ifferential) der Eingangsgröße hinzu.
- PI-Regler: Hierbei kommt zum **P**roportionalitätsfaktor noch das **I**ntegral der Eingangsgröße hinzu.
- PID-Regler: Kombination aus **P**roportionalitätsfaktor, **I**ntegral und **D**ifferential. Wird häufig als Universalregler angesehen.
- PDT-Regler: **PD**-Regler mit zusätzlicher Zeitverzögerung (**T**ime Delay) durch gleitende Mittelwertbildung. Verhalten ähnlich dem PID-Regler.

Die zugehörigen mathematischen Funktionen werden im nächsten Kapitel 7 vorgestellt.

Des weiteren wird bei der Lenk-Regelung eines Fahrzeugs meist eine Begrenzung des Regler-Ausgangs verwirklicht, um eine Überreaktion der Regelstrecke, hier der Antriebs-Lenkung, zu vermeiden.

7. Mathematische Behandlung des Regelkreises

Vereinfacht wird der Regelkreis auf zwei Blöcke reduziert. Hierin bedeutet a der Sollwert und s der Sensorwert:



Im allgemeinen Modell kontinuierlicher Größen x und y werden die jeweiligen Zeitfunktionen $x = x(t)$ und $y = y(t)$ benötigt.

Für ein zeitdiskretes System, z.B. bei Realisierung durch Software, werden die Abtastwerte x_n der Eingangsgröße x an den äquidistanten Zeitpunkten $t_n = n \cdot T$ ($n = 0, 1, 2, 3, \dots$) herangezogen. T ist dabei das Abtastintervall, in der Software die Schleifenverzögerung.

Tabelle 7.1: Gleichungen für die **Regler**:

Regler-Typ	Grundgleichung (kontinuierlich)	Zeitdiskretes System (digital)	
P-Regler	$y = k_P \cdot x$	$y_n = KP \cdot x_n$	
PD-Regler	$y = k_P \cdot x + k_D \cdot \frac{dx}{dt}$ $= k_P \cdot \left(x + T_D \cdot \frac{dx}{dt} \right)$	$y_n = KP \cdot x_n + KD \cdot (x_n - x_{n-1})$	
PI-Regler	$y = k_P \cdot x + k_I \cdot \int_0^t x d\tau$ $= k_P \cdot \left(x + \frac{1}{T_I} \cdot \int_0^t x d\tau \right)$	$y_n = KP \cdot x_n + KI \cdot \sum_{i=1}^n x_i$	
PID-Regler	$y = k_P \cdot x + k_I \cdot \int_0^t x d\tau + k_D \cdot \frac{dx}{dt}$ $= k_P \cdot \left(x + \frac{1}{T_I} \cdot \int_0^t x d\tau + T_D \cdot \frac{dx}{dt} \right)$	$y_n = KP \cdot x_n + KI \cdot \sum_{i=1}^n x_i + KD \cdot (x_n - x_{n-1})$	
PDT-Regler	$y + T_1 \frac{dy}{dt} = k_P \cdot x + k_D \cdot \frac{dx}{dt}$ $= k_P \cdot \left(x + T_D \cdot \frac{dx}{dt} \right)$	$y_n + K1 \cdot (y_n - y_{n-1}) = KP \cdot x_n + KD \cdot (x_n - x_{n-1})$ $\Leftrightarrow y_n = \frac{K1 \cdot y_{n-1} + KP \cdot x_n + KD \cdot (x_n - x_{n-1})}{1 + K1}$	

Bei den Gleichungen des kontinuierlichen Reglers können die D- und I-Parameter alternativ als Faktoren k_D, k_I oder als Zeitkonstanten T_D, T_I eingesetzt werden.

Die Parameter des digitalen Reglers KD, KI unterscheiden sich von denen des kontinuierlichen k_D, k_I , da nur die Folge der x-Werte, nicht aber die Abtastzeit oder Programmschleifenzeit berücksichtigt wird.

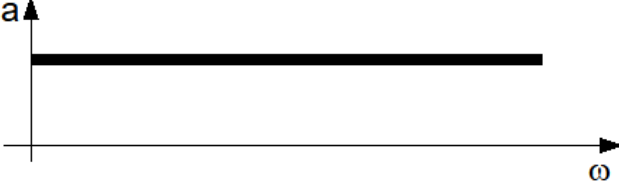
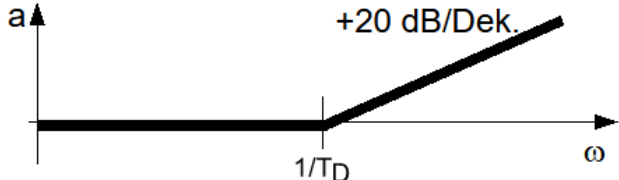
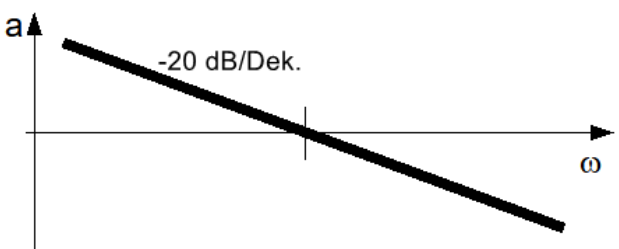
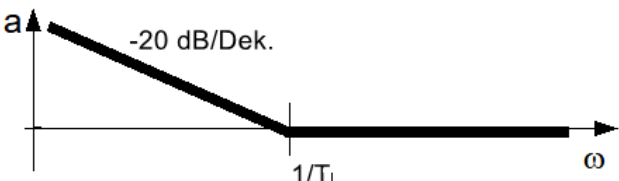
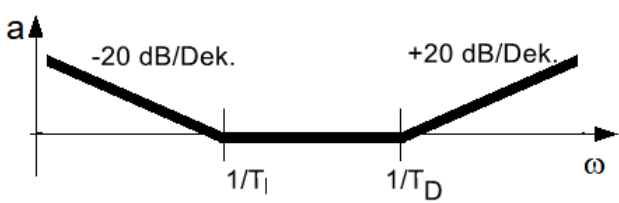
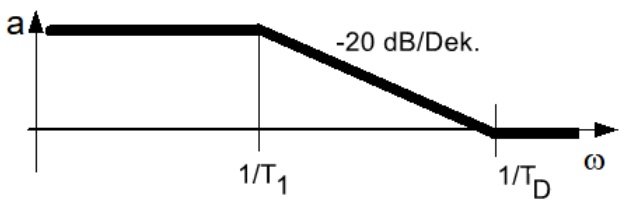
Die folgenden Betrachtungen sind eher für die Spezialisten, ansonsten weiter mit Kapitel 8.

Für die systemtheoretische Behandlung mit Methoden der Regelungstechnik sind allerdings die Zeitfunktionen weniger geeignet, daher werden die Größen gewöhnlich in den Frequenzbereich transformiert, z.B. mit Hilfe der Laplace-Transformation. Die symbolische Frequenz erhält das Formelsymbol s. Regler und Regelstrecke + Sensor besitzen dann die zugehörigen Übertragungsfunktionen $G_1(s)$ und $G_2(s)$, die Einzelgrößen $A(s)$, $X(s)$, $Y(s)$, $S(s)$.

Als Systemgrößen sind definiert:

- Ringverstärkung (bestimmt die Stabilität): $G_R = G_1 \cdot G_2$
- Betriebsverstärkung: $G_B = \frac{S(s)}{A(s)} = \frac{G_1 G_2}{1 + G_1 G_2} = \frac{1}{1 + \frac{1}{G_1 G_2}} = \frac{1}{1 + \frac{1}{G_R}}$

Tabelle 7.2: Es ergibt sich für die Reglertypen:

Regler-Typ	Transformierte Gleichung $G_I(s) = Y(s) / X(s)$	Bode-Diagramm	
P-Regler	k_p		
PD-Regler	$k_p \cdot (1 + s T_D)$		
I-Regler	$\frac{1}{s T_I}$		
PI-Regler	$k_p \cdot (1 + \frac{1}{s T_I})$		
PID-Regler	$k_p \cdot (1 + \frac{1}{s T_I} + s T_D)$		
PDT-Regler	$k_p \cdot \frac{1 + s T_D}{1 + s T_1}$		

Das „Bode-Diagramm“ beschreibt den Verlauf der Übertragungsfunktion in Dezibel (dB) über der Frequenzvariable $s = \omega = 2\pi f$.

8. Stabilität

Ein mögliches Problem bei Regelkreisen besteht darin, dass die Regelgröße nicht den zugeführten Sollwert erreicht (oder wenigstens annähert), sondern weg driftet oder sogar schwingt.

Zur Bewertung der Stabilität muss die Ringverstärkung $G_R = G_1 \cdot G_2$ betrachtet werden. Wendet man das Bodediagramm auf die Ringverstärkung an, müssen die Diagramme für G_1 und G_2 addiert werden. Im Ergebnis sollten keine sehr großen (negativen) Steigungen (Betrag >40 dB/Dekade) in einzelnen Bereichen auftreten.

Wie im Abschnitt 4 hergeleitet, liegt für G_2 prinzipiell ein doppelt integrierendes Verhalten vor. So wie beim I-Regler in Kapitel 7 ergibt sich für eine Integration eine konstante Steigung im Bodediagramm von -20 dB/Dekade, bei doppelter Integration -40 dB/Dekade. Somit würde ein zusätzlicher I-Anteil im Regler sogar eine partielle Steigung von -60 dB/Dekade bewirken, was die Stabilität der Regelung eher negativ beeinflusst.

Daher wird bei der Lenkregelung „nur“ ein PD-Regler eingesetzt. Der P-Anteil wertet Abweichungen von der Linienmitte aus und regelt entsprechend nach. Der D-Anteil verstärkt die Regelung, falls sich der Roboter weiter von der Linie entfernt, bzw. verringert die Regelung, falls der Roboter sich ohnehin der Linie nähert. Zur Einstellung der Parameter k_P und k_D hat sich ein empirisches Vorgehen bewährt:

- Setze $k_D = 0$ und erhöhe k_P so lange, bis das Fahrzeug eine leichte Schlangenlinie fährt.
- Reduziere dann k_P um etwa ein Drittel (es sollte keine Schlangenlinie mehr entstehen).
- Setze einen Anfangswert für k_D und platziere den Roboter etwas schräg auf der Linie. Steuert er zu stark gegen, reduziere k_D . Nähert er sich nur langsam der Linie, erhöhe k_D .

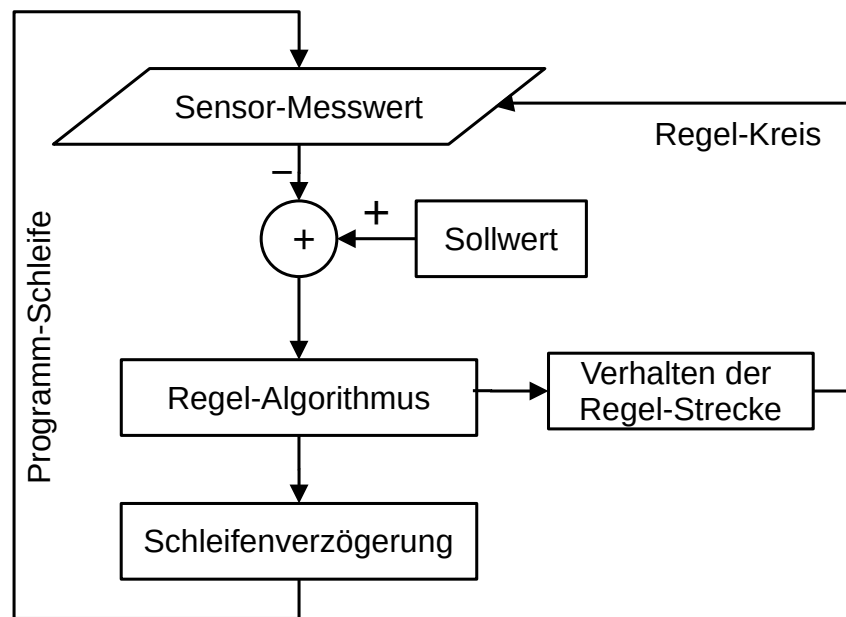
Soll die stationäre Regelabweichung minimiert werden, muss das Produkt $G_R = G_1 \cdot G_2$ für $\omega \rightarrow 0$ groß werden (siehe Betriebsverstärkung). Das ist bei einem integrierenden Anteil immer der Fall.

Anmerkung: Die gewählte Darstellung ist sehr vereinfacht und geht außerdem von einem linearen System aus. Eine genauere Betrachtung erfordert tiefe Kenntnisse der Regelungstechnik!

Gegebenenfalls kann man auch mit mathematischen Simulationen (Kapitel 10) weitere Erkenntnisse gewinnen.

9. Digitale Regelung

Heutiger Stand der Technik ist die weitgehend digitale Steuerung eines Fahrroboters, einschließlich der erforderlichen Regelungs-Mechanismen. Für die Lenkregelung kann man das nachfolgende Ablaufdiagramm als Basis ansetzen:



Der physikalische Regelkreis besteht, wie am Anfang von Kapitel 7 beschrieben, aus dem Sensor, dem Regelalgorithmus und dem Verhalten der Regelstrecke = Lenkverhalten. Der Sensorwert wird jeweils mit dem Sollwert verglichen, das ist der gewünschte Wandabstand bei der Wandverfolgung bzw. Null bei der Linienverfolgung mit einem Reflexions-Differenzsensor.

Programmtechnisch wird die Regelung durch eine Schleife realisiert. Der Abbruch der Schleife, z.B. nach Erreichen einer gewissen Fahrstrecke, ist nicht eingezeichnet.

Um eine definierte Zeitbedingung bei differenzierenden oder integrierenden Algorithmen zu erreichen, ist innerhalb der Schleife eine (möglichst konstante) Verzögerung T erforderlich. Dann ergibt sich z.B. die zeitliche Ableitung des Sensorwerts $s(t)$ mit

$$\frac{ds(t)}{dt} = \frac{\text{aktueller Sensorwert} - \text{letzter Sensorwert}}{T}.$$

Die Algorithmen für die Standardregler sind in der Tabelle 7.1 unter „Zeitdiskretes System“ beschrieben, z.B. für den bevorzugten PD-Regler: $y_n = KP \cdot x_n + KD \cdot (x_n - x_{n-1})$.

Um den gesamten Regel-Kreis zu analysieren, müssen die Charakteristik des Sensors und das Lenkverhalten mit geeigneten Modellen eingebracht werden. Da hier auch Nichtlinearitäten einfließen, ist die Analyse mittels eines Mathematik-Programms oder eines Simulationstools nützlich.

10. Simulation

Es soll das Lenkverhalten mit Hilfe eines Mathematik-Programms analysiert werden. Dazu wird ein Skript für das Open-Source-Programm GNU Octave geschrieben, das weitgehend kompatibel mit dem kommerziellen Programm MATLAB von MathWorks ist.

Um die unterschiedlichen Zeitfunktionen für die Regelung und für die abschließende grafische Ausgabe zwischen zu speichern, werden verschiedene Arrays oder Vektoren definiert:

- n: Zählindex der Programmschleife
- t: diskrete Zeitpunkte
- x: Seitenabweichungen
- y: Zurückgelegte Fahrstrecken
- w: Winkelabweichungen
- L: Lenkwerte (ganzzahlig)
- refl: Sensorwerte

Die Anzahl der Vektor-Elemente hängt von der Simulationszeit und der Schleifenzeit ab.

Man beachte, dass der Zählindex bei Octave und Matlab immer bei 1 beginnt (nicht 0 wie in den meisten Programmiersprachen) und dieser in runde Klammern gesetzt wird. Z.B. ist x(3) das dritte Element im Vektor x.

Folgende roboterspezifische Parameter müssen gesetzt werden:

- A: Radabstand = Achslänge in mm
- ys: Sensorposition in mm vor Achse
- v: Fahrgeschwindigkeit in mm/s (konstant; Beschleunigungsphasen vernachlässigt)

Regelungsspezifische Vorgaben:

- kp: proportionaler Faktor der PD-Regelung
- kd: differentieller Faktor der PD-Regelung
- maxlenk: maximaler Lenkwert (Betrag)
- refSoll: Sollwert des Sensors (bei Linienverfolgung = 0)
- deltaT: Schleifenzeit in s
- fahrzeit: Gesamte Simulationszeit in s

Parameter beim Start (zum Testen unter verschiedenen Bedingungen):

- w0: Winkelabweichung in Grad
- x0: Seitenabweichung in mm

Ein Beispiel-Skript sieht wie folgt aus:

```
1   clc;    # Loeschen des Befehlsfensters
2
3   # Eingabeparameter:
4   kp = 0.05 # P-Faktor Regelung
5   kd = 0.10 # D-Faktor Regelung
6   maxlenk = 10
7   refSoll = 0
8   v = 500   # Fahrgeschwindigkeit mm/s
```

```

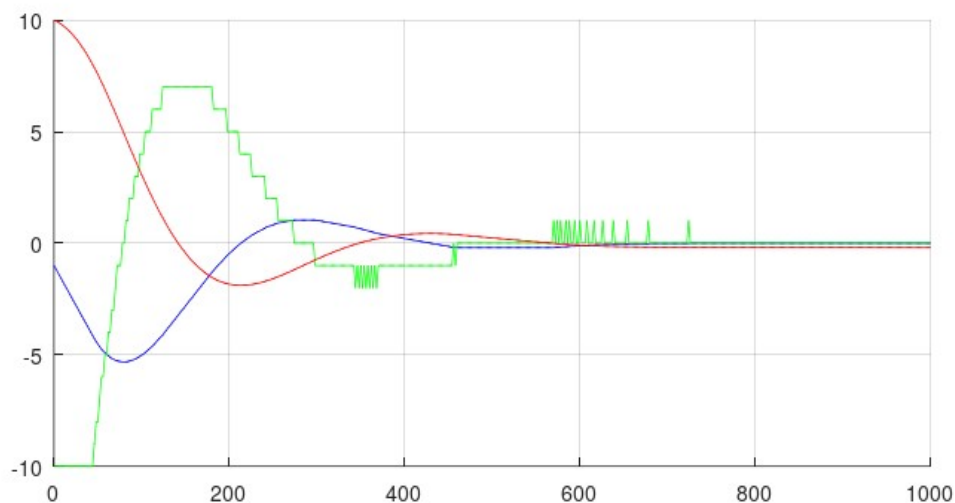
9      A = 160      # Radabstand mm
10     ys = 50      # Sensorposition mm vor Achse
11     deltaT = 4*0.001 # Schleifenzeit s
12     fahrzeit = 2.0      # Simulationszeit s
13     w0 = -1      # Winkelabweichung Grad beim Start
14     x0 = 10      # Seitenabweichung mm beim Start
15
16     # Initialisierte Werte:
17     N = round(fahrzeit / deltaT) # Anzahl Scheifendurchlaeufer
18     n = 1 : N+1;
19     t = deltaT * n;          # diskrete Zeitpunkte ms
20     refl = zeros(1, N+1); # Reflexionswerte
21     w = zeros(1, N+1);      # Fahrtrichtung in rad
22     w(1) = w0 / 57;        # Umrechnung in rad
23     x = zeros(1, N+1);      # Querposition mm
24     x(1) = x0;
25     y = zeros(1, N+1);      # Fahrstrecke mm
26     L = zeros(1, N+1);      # Lenkwerte int
27
28     # Regelschleife:
29     for i = 1 : N
30         # Sensor
31         refl(i) = 50 * (x(i) + ys * w(i)); # Modell: Umrechnung von x in refl
32         if (refl(i) > 1000)
33             refl(i) = 1000;
34         elseif (refl(i) < -1000)
35             refl(i) = -1000;
36         endif
37         # Regler:
38         if (i > 1)
39             L(i) = -round(kp * (refl(i) - refSoll) + kd * (refl(i) - refl(i-1)));
40         else
41             L(i) = -round(kp * (refl(i) - refSoll));
42         endif
43         if (L(i) > maxlenk)
44             L(i) = maxlenk;
45         elseif (L(i) < -maxlenk)
46             L(i) = -maxlenk;
47         endif
48         w(i+1) = w(i) + deltaT * v * L(i) / A / 50;
49         x(i+1) = x(i) + deltaT * v * (w(i) + w(i+1)) / 2;
50         y(i+1) = y(i) + deltaT * v;
51     end
52
53     # Ergebnisausgabe:
54     strecke = y(N+1)
55     #clf;
56     hold on;
57     grid on;
58     plot(y,w*57,"b"); # in Grad
59     plot(y,L,"g");
60     plot(y,x,"r");

```

- In den Zeilen 4-14 werden die Parameter festgelegt.
- Die Zeilen 17-26 erzeugen die Vektoren, teilweise wird deren erstes Element initialisiert.

- In der Regelschleife (Zeilen 29-51) wird zuerst aus der aktuellen x-Position der Sensorwert s nach einem einfachen linearen Modell berechnet und auf das Intervall $[-500; 500]$ begrenzt (Zeilen 31-36). Dabei wird eine Schiefstellung um den Winkel w mit berücksichtigt.
- In Zeile 39 bzw. 41 wird der PD-Algorithmus angewendet. Für $i=1$ existiert $\text{refl}(i-1)$ nicht, daher die Fallunterscheidung. Mit der round-Funktion werden ganzzahlige Werte gebildet.
- Die Zeilen 43-47 begrenzen den gerade berechneten Lenkwert L auf maxlenk (in Zeile 6 definiert).
- In den Zeilen 48-50 werden die Parameter w , x , y für den nächsten Schleifendurchlauf berechnet. Beim x -Wert wird der Mittelwert von letztem Winkel $w(i)$ und neuem Winkel $w(i+1)$ verwendet.
- Zeilen 56-60: Die Ergebnisse werden mit plot grafisch dargestellt, mit der Fahrstrecke y als horizontale Achse. Der letzte plot-Parameter bestimmt die Farbe der Kurven (w blau, L grün, x rot).

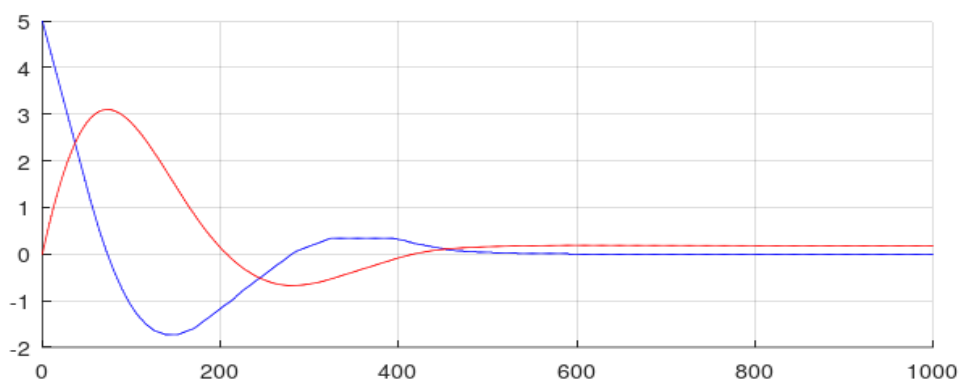
Das Ergebnis dieser Simulation ist:



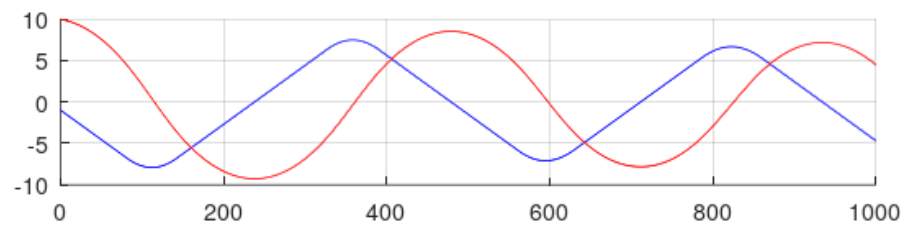
Man erkennt, dass die anfängliche Linienabweichung von 10 mm nach etwa 300 mm weitgehend ausgeglichen wird (rote Kurve), ebenso der anfängliche Fehlwinkel -1° (blaue Kurve). Die Lenkwerte L werden in grün gezeigt und bestehen aus Stufen, da sie nur ganzzahlige Werte haben, was der üblichen Motorensteuerung entspricht.

Durch Änderung der Parameter lässt sich leicht deren Einfluss auf das Regelungsverhalten analysieren.

Selbst eine anfängliche Schiefstellung um 5° ($w_0 = 5$, $x_0 = 0$) wird schnell ausgeglichen:



Es lässt sich auch leicht nachweisen, dass ein gewisser Abstand y_s zwischen Antriebsachse und Liniensensor erforderlich ist. Setzt man $y_s = 0$, ergibt sich unabhängig von den Regelungsparametern immer ein schwingendes Verhalten:



Selbst bei $y_s = 20$ mm dauert der Ausgleichsvorgang sehr lange:

