Regeln statt nur steuern

Proportional – Integral – Differential: PID-Regler

Der PID-Regler ist gewissermaßen die hohe Schule der Regelungstechnik. Statt nur wie ein einfacher Regler einen Prozess mit einer einzigen Regelgröße nachzuregeln, kann ein PID-Regler die Vorzüge mehrerer Einzelregelungen in sich vereinigen und so genau auf die anfallenden Anforderungen eingestellt werden bzw. schnell auf Störungsgrößen reagieren. Wir stellen sein Arbeitsprinzip vor und zeigen anhand zahlreicher nachvollziehbarer Projekte auf, wie man eine PID-Regelung in Mikrocontroller-Umgebungen realisieren kann.

Immer im Kreis ...

Warum heißt der Lautstärkeregler am Radio umgangssprachlich Regler? Im Grunde genommen ist er ja eine Steuerung, er teilt im Normalfall dem Endverstärker einen gewissen Signalpegel zu. Zum „Regler" wird er, wenn der Hörer den aus dem Lautsprecher kommenden Istwert mit dem Sollwert, also der gewünschten Lautstärke, vergleicht und den Lautstärkeregler entsprechend nachstellt. Das ist zugegebenermaßen ein sehr triviales Beispiel, aber bereits im normalen Leben begegnen wir täglich der Regelungstechnik. Das fängt bei der Temperaturregelung an, ob im Kühlschrank oder in der Heizung, und setzt sich vielfach fort, so z. B. in der Waschmaschine, bei der ein Regelkreis die vorgegebene Umdrehungszahl des Antriebsmotors auch bei unterschiedlicher Belastung stabil hält.

Bei den Reglern unterscheiden wir grob zwischen Zweipunktreglern, Proportionalreglern und dem PID-Regler. Der einfachste Regler ist dabei der Zweipunktregler, der den Bereich zwischen zwei Sollwerten überwacht und bei Erreichen eines der Sollwerte jeweils den Aktor umschaltet, etwa bei einem elektrischen Heizungsregler.

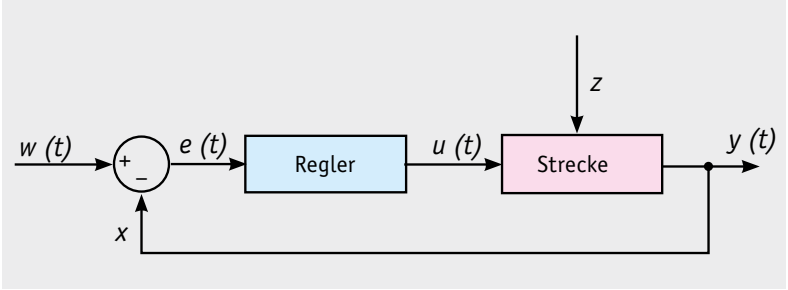

Bild 1: Die Bestandteile einer klassischen Regelstrecke

Bild 1 illustriert eine einfache Regelung. Wir haben hier den Regler, der dafür zuständig ist – nach Aufbereitung des Sollwerts (w, auch Führungsgröße genannt) und des Istwerts (y, Regelgröße) – anhand der hier erkannten Regelabweichung eine Stellgröße (u), also den entsprechend angeglichenen Sollwert auf die Regelstrecke, z. B. den Waschmaschinenmotor, auszugeben. Die den Regler ansteuernde Regelabweichung wird aus der Differenz zwischen Sollwert und Istwert gebildet. Wirkt auf den Motor eine Störgröße (z), z. B. wechselnde Belastung, wird dies wiederum rückgekoppelt über den Istwert (y), den gibt der Tachogenerator an jedem Waschmaschinenmotor aus, und der Regelkreis wird erneut aktiv. So funktioniert im Prinzip jeder Regelkreis: Soll- und Istwert werden verglichen, und per Regler wird die Abweichung zwischen beiden nachgestellt.

Natürlich muss ein solcher Regler schnell arbeiten, um ständig die tatsächlichen Verhältnisse abzubilden. Würde er zu langsam reagieren, käme es zu ständigen Schwankungen der Drehzahl. Dies nennt man Zeitverhalten der Regelung, sie spielt eine wesentliche Rolle, um starke Schwankungen der Regelgröße (Schwingen) zu vermeiden. In der Praxis spielen also genau diese Faktoren, das Zeitverhalten eines Reglers und seine Regelgenauigkeit (Istwert = möglichst genau und konstant eingehaltener Sollwert) die bestimmende Rolle bei der Auslegung eines Regelkreises. Deshalb kommen hier die verschiedensten Reglertypen zum Einsatz. Die klassischen Grundtypen bilden dabei die technische Grundlage für den kombinierten Drei-in-einem-Regler, den detailliert auf die jeweilige Anforderung einstellbaren PID-Regler.

Die klassischen Regler

Die Klassiker unter den Reglern sind drei stetige lineare Reglertypen.

Der Proportional-Regler (P-Regler)

Dieser Regler (Bild 2 stellt ihn in der klassischen analogen Form mit Operationsverstärker dar) ermittelt die Differenz zwischen Soll- und Istwert. An seinem Ausgang erscheint die Abweichung zwischen Soll und Ist, multipliziert mit dem Verstärkungsfaktor des Reglers (*Kp*).

Die vereinfachte Formel für dieses Verhalten lautet daher:

$$y(t) = Kp \cdot e(t)$$

Dieser Regler realisiert die schnellste Regelung, hat aber auch einen Nachteil, er erreicht nie die genaue Vorgabe, also den Sollwert, es bleibt eine bleibende Regelabweichung. Wählt man *Kp* zu hoch, produziert der Regler zunächst zu Beginn ein Überschwingen, bevor er wieder gleichmäßig regelt. Dieser Effekt ist nicht immer gewollt, da er beim Überschwingen eine zu hohe Regelabweichung erzeugt.

In der klassischen Schaltung aus Bild 2 stellt man den Verstärkungsfaktor durch die Division von R2 durch R1 ein.

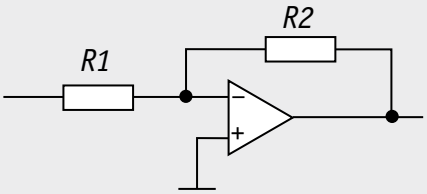

Bild 2: Analoge Schaltung des P-Reglers

Der Integral-Regler (I-Regler)

Hier (Bild 3) kommt der bereits erwähnte Faktor Zeit ins Spiel. Der I-Regler wird auch integrierender Regler genannt, er berücksichtigt den zeitlichen Verlauf in der Regelabweichung, indem er diese über die Zeit aufsummiert und diese Summe mit dem Verstärkungsfaktor multipliziert.

$$y(t) = \frac{1}{Tn} \int_0^t e(\tau)\,d\tau \qquad \left(Ki = \frac{1}{Tn} \right)$$

In der späteren PID-Berechnungspraxis wird der Faktor *Ki* als Kehrwert der Nachstellzeit benutzt. Dauert die Regelabweichung also länger, erhöht der Regler sukzessive den Verstärkungsfaktor. Dieser Vorgang benötigt zwar eine gewisse Zeit (Nachstellzeit *Tn*), aber der Regler realisiert am Schluss genau Sollwert = Istwert.

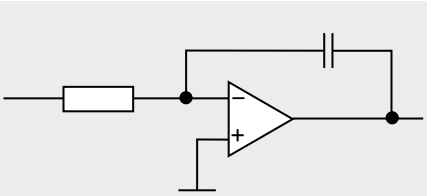

Bild 3: Analoge Schaltung des I-Reglers

Das Differential-Glied (D)

Dieses Regelglied ergänzt die beiden bisher beschriebenen Typen, die auch kombiniert als PI-Regler einsetzbar sind. Es ist eigentlich kein eigener Regler (auch, wenn es später im PID-Regler als solcher bezeichnet wird), sondern es erfasst die Geschwindigkeit der Regelabweichung und beschleunigt die Reaktion

des Reglers durch zeitlich begrenzte Verstärkung der Differenz zwischen Soll- und Istwert – begrenzt, weil es sonst zum Überschwingen der Regelung kommt. Diesen Vorgang bezeichnet man als Differenzierung. Es ist also eine Bewertung der zeitlichen Änderung der Regelabweichung. Denn in der Praxis ist man bemüht, die Verstärkung so weit zu erhöhen, dass man eine minimale Regelzeit erreicht. Dabei kann es wieder zu Überschwingen kommen, und hier kommt die D-Komponente ins Spiel, solange die Steigungen der Sollwert- und der Istwertkurve noch nicht identisch sind – sie bremst den Anstieg. In der zugehörigen Formel finden wir jetzt die Vorhaltzeit, die Zeitkonstante *Tv*, die wir später beim PID-Regler umgewandelt als Parameter *Kd* wiederfinden:

$$y(t) = Tv \frac{d}{dt} e(t)$$

Kennzeichen von Reglern mit D-Glied ist eine sehr schnelle und stabile Regelung.

Kombinierte Regler

Die kombinierten Regler fassen jeweils zwei klassische Regler zusammen und erfordern zur Programmierung jeweils zwei Einstellparameter.

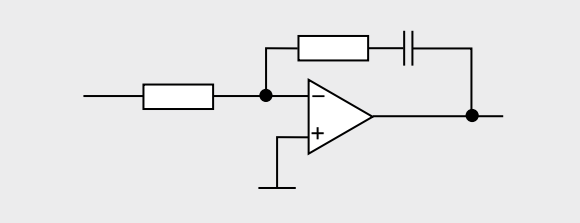

Bild 4: Analoge Schaltung des PI-Reglers

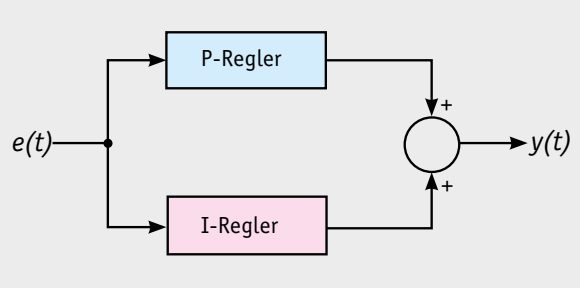

Bild 5: Aufbau des PI-Reglers in Parallelstruktur

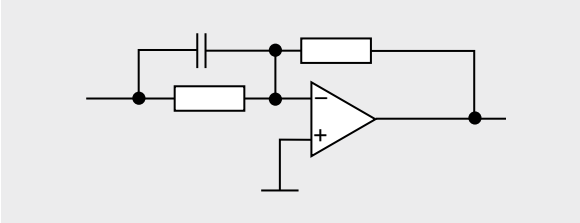

Bild 6: Analoge Schaltung des PD-Reglers

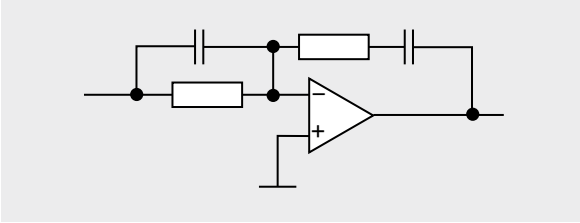

Bild 7: Analoge Schaltung des PID-Reglers

Der PI-Regler

Wie der Name schon sagt, ist der PI-Regler eine Kombination aus P- und I-Regler, hier wird die schnelle Regelung des P-Reglers mit der genauen Soll-/Istwert-Ausregelung des I-Reglers kombiniert:

$$y(t) = Kp \left[e(t) + \frac{1}{Tn} \int_0^t e(\tau) d\tau \right]$$

In der Formel sehen wir die Kombination aus der Wirkung des Verstärkungsfaktors *Kp* und der Zeitkonstante (Nachstellzeit) *Tn*. In der analogen Regelschaltung (Bild 4) findet man nun auch die Kombination aus P- und I-Regler. Im Vergleich zum reinen P-Regler entsteht hier keine bleibende Regelabweichung, nach Ausregelung durch den I-Regler stimmen Soll- und Istwert, also die Führungsgröße überein.

Bild 5 zeigt die Parallelstruktur eines solchen Reglers. Als Einstellparameter in der Praxis agieren hier also die beiden Parameter *Kp* und *Tn* (bzw. dessen aus dem Kehrwert von *Tn* hervorgehender Paramter *Ki*)

Der PD-Regler

Hier wird der P-Regler mit dem D-Glied (auch D-Anteil genannt) kombiniert:

$$y(t) = Kp \left(Tv \frac{d}{dt} e(t) + e(t) \right)$$

Er reagiert z. B. bereits auf eine geringe Sensorveränderung und „spekuliert" quasi mit der Vorhaltzeit auf die mögliche kommende und größere Veränderung. Damit versucht er praktisch, einer kommenden Abweichung zuvorzukommen und erreicht die beschriebene schnelle Regelzeit bis zum stabilen Betrieb. Bild 6 führt hier wieder die entsprechende Analogschaltung dieses Reglertyps auf. Die Einstellparameter sind hier *Kp* und *Tv*.

PID – der Universelle

Der Name sagt es, der PID-Regler fasst die Eigenschaften aller drei klassischen Regler zusammen, somit ist er der flexibelste, genaueste, aber auch nur ein mäßig schneller Regler.

Die analoge Schaltungskombination des PID-Reglers ist in Bild 7 zu sehen.

In der üblichen Blockdarstellung des PID-Reglers sind die drei Einstellparameter Kp, Tn (Ki) und Tv (Kd), als idealer Regler in Parallelstruktur angenommen, wiederzufinden (Bild 8). Dort sind auch die in der zum PID-Regler gehörenden Differenzialgleichung:

$$y(t) = Kp \left[e(t) + \frac{1}{Tn} \int_0^t e(\tau) d\tau + Tv \frac{d}{dt} e(t) \right]$$

gekennzeichneten Blöcke zu sehen. In Bild 9 kann man das Regelverhalten der einzelnen Reglertypen und -anteile sehen und erkennt sogleich, dass der PID-Regler ganz offensichtlich die optimale Regelung darstellt.

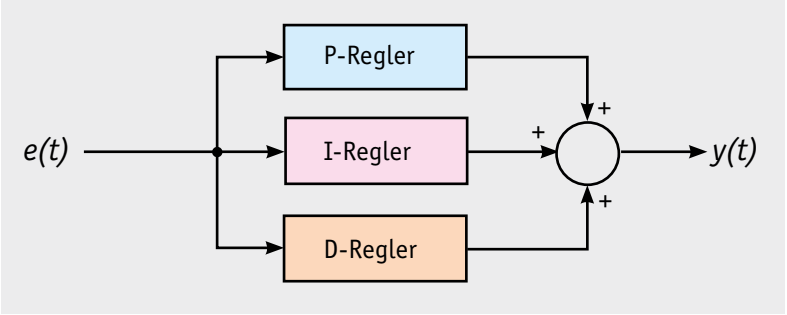

Bild 8: Aufbau des PID-Reglers in Parallelschaltung

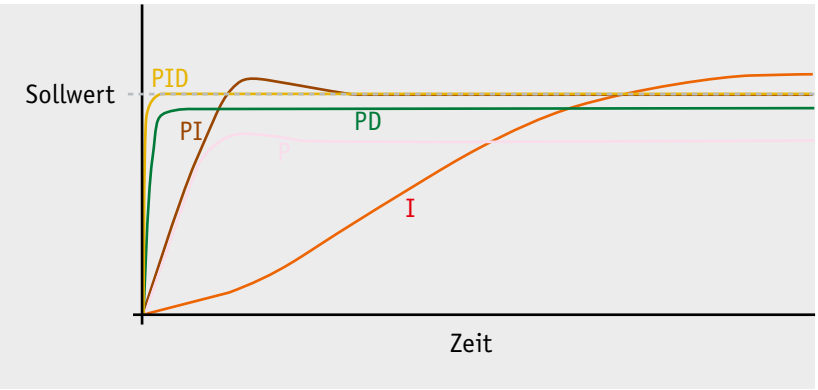

Bild 9: Vergleich des Regelverhaltens verschiedener Regler anhand der Sprungantworten

PID – die Praxis

Heute sind PID-Regler nahezu ausschließlich in Mikrocontroller-Anordnungen integriert, die praktischerweise einen perfekten digitalen Regler abbilden können. Den eigentlichen Regler bilden dabei die CPU und das Anwendungsprogramm, die Verbindung zwischen Controller und Regelstrecke wird durch die im Controller integrierten Digital-analog- und Analog-digital-Wandler hergestellt. Die Ausgabe an die Aktoren erfolgt dabei meist als PWM-Signal.

Je nach gewünschter Anwendung können die drei Parameter berechnet oder rein empirisch ermittelt werden. So spielt bei einer Heizungsregelung zum Beispiel der D-Anteil eine geringere Rolle als der P- und I-Anteil, da hier eine eher träge Regelung gewünscht ist, die auf kurze, schnelle Veränderungen, etwa starke temporäre Absenkung der Außentemperatur bei einem Gewitter, nicht reagieren muss. Bei einer Motorregelung hingegen spielt der D-Anteil eine Rolle, um bei sich anbahnender Lastveränderung die Drehzahl konstant zu halten. Ganz stark ist dieser bei Steuerungen gefragt, die eine Lageregelung realisieren müssen, etwa bei einer Hoverboard-Steuerung.

Eine sehr anschauliche und praktikable, wenn auch vereinfachte Erläuterung zur praktischen Anwendung der Parameter ist zum Beispiel in [1] zu finden. In [2] ist die Berechnung der Parameter, auch anhand von AVR-Programmen praxisorientiert nach den verschiedenen Methoden, also berechnet nach Einstellregeln oder empirisch ermittelt, zusammengetragen. Wir wollen an dieser Stelle nicht detailliert darauf eingehen, die genannten Quellen führen hier sehr umfangreich weiter.

Stattdessen wollen wir auf einfach nachvollziehbare Projekte hinweisen, die auch den Umgang mit einer PID-Regelung schnell veranschaulichen. In [3] findet man eine Arduino-Anwendung zur Steuerung von Servo- und anderen Motoren. In den zugehörigen Arduino-Programmen kann man die Definition und Auswahl der Parameter gut verfolgen.

Eine Anwendung zur Realisierung einer kontollierten Wohnraumlüftung mittels PID ist unter [4] erläutert. Schließlich beschreiben [5] das PID-Tuning für die Heizungsregelung von Extruder und Heizbett bei 3D-Druckern, [6] eine Regelung der Heizung für einen Eierkocher, und in [7] findet man eine weitere Anwendung für eine Lüfterregelung per PID.

Abschließend soll hier auch eine PID-Anwendung in der Hausautomatisierung genannt werden, das PID20-Projekt für die Smart Home Lösung FHEM in [8].

Weitere Infos:

[1] forum.arduino.cc/index.php?topic=433030.0
[2] homepages.uni-regensburg.de/~erc24492/PID-Regler/PID-Regler_mit_uC.html
[3] www.homofaciens.de/technics-computer-arduino-uno_ge.htm
[4] github.com/svenjust/room-ventilation-system
[5] www.kinder-technik.de/3d-drucker-pid-tuning-extruder-heizbett/
[6] www.mikrocontroller.net/articles/Onsen_Eierkocher
[7] www.amateurfunkbasteln.de/arduino_luefter/index.html
[8] wiki.fhem.de/wiki/PID20_-_Der_PID-Regler
[9] playground.arduino.cc/Code/PIDLibrary/
[10] www.robotis.us
[11] www.cmu.edu/me/robomechanicslab/robots.html

- H. Lutz, W. Wendt: Taschenbuch der Regelungstechnik mit MATLAB und Simulink 11. Auflage. Europa-Verlag, 2019, ISBN 978-3-8085-5869-0
- rn-wissen.de/wiki/index.php/Regelungstechnik
- de.wikipedia.org/wiki/Regler
- en.wikipedia.org/wiki/PID_controller

Alle Links finden Sie auch online unter: de.elv.com/elvjournal-links

Bild 10: Die Dynamixel-Servomotoren sind robust, äußerst universell einsetzbar und enthalten eine umfangreiche Elektronik bis hin zu Lagesensoren und integriertem PID-Regler. Sie sind in zahlreichen Größen und Leistungsklassen für Roboteranwendungen verfügbar. Bild: Robotis

Für alle Mikrocontroller-Plattformen gibt es auch fertige, direkt einsetzbare Bibliotheken, so etwa für den Arduino die „PID.h" oder die ebenfalls das Programmieren vereinfachende „PIDController.h", die in einer eigenen Wiki umfangreich erläutert ist. Zusätzlich ist im Arduino-Playground [9] eine ausführliche Abhandlung zum Thema „PID mit Arduino" hinterlegt. Im dortigen Github-Link sind zudem einfache Beispiele zu finden, die ein Verständnis für die Einstellung der Regelparameter in verschiedenen Anwendungen fördern.

PID integriert

Im Rahmen dieses Beitrags über PID-Regler wollen wir am Schluss auf eine interessante Möglichkeit hinweisen, Antriebe mit PID-Steuerung besonders einfach realisieren zu können. Speziell für den Aufbau von Robotern aller Couleur gibt es die Dynamixel-Servomotor-Reihe von Robotis INC. [10]. Diese Motoren, Bild 10 zeigt ein Beispiel aus der X-Serie, weisen neben hoher Robustheit eine Reihe herausragender Merkmale auf, so etwa eine Busansteuerung, aber auch eine integrierte PID-Regelung.

In Bild 11 ist die Architektur der integrierten Kontrolleinheit zu sehen. Hier finden wir den PID-Regler wieder. Die Verbindung mehrerer Motoren mit dem Controller erfolgt im Daisy-Chain-Verfahren, bei dem jeder Motor eine eigene ID zugeordnet bekommt (Bild 12). So muss man dem Motor nur noch die jeweiligen Parameter übergeben und Statuswerte auswerten.

Wir haben mit diesen Motoren ein kleines Open-Source-Projekt, den MiniRHex-Laufroboter des Robomechanics Lab der Carnegie Mellon University [11], nachgebaut (Bild 13).

Mit diesem Laufroboter, der tatsächlich für seine Größe sehr leistungsstark ist und nahezu beliebiges Gelände überwinden und dabei sogar springen kann, ist man in die Lage versetzt, sehr einfach über das im GitHub angebotene Open-Source-Programm (Bild 14), das dort inklusive der zugehörigen Bibliotheken bereitsteht, das Parametrieren des PID-Reglers zu trainieren. Besonders interessant ist hier tatsächlich die Befehlsversion des Springens, die man sehr weit ausfeilen kann, sodass der Laufroboter etwa über höhere oder längere Hindernisse springen kann.

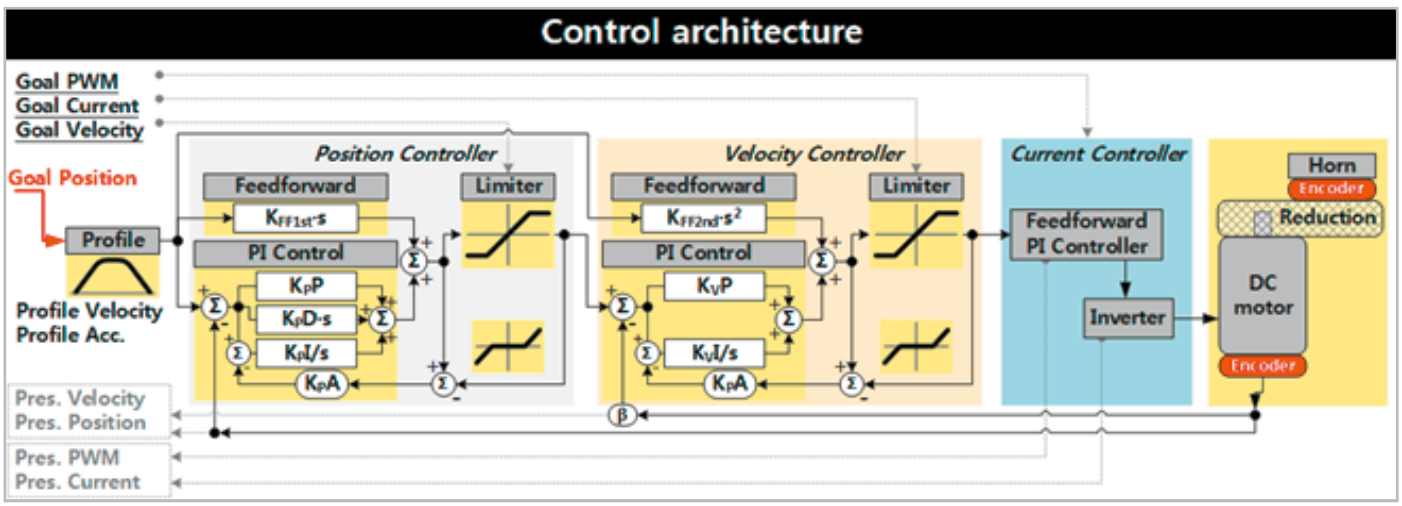

Bild 11: Die interne Steuerungsarchitektur der Dynamixel-Servomotoren mit PID-Steuerung. Bild: Robotis

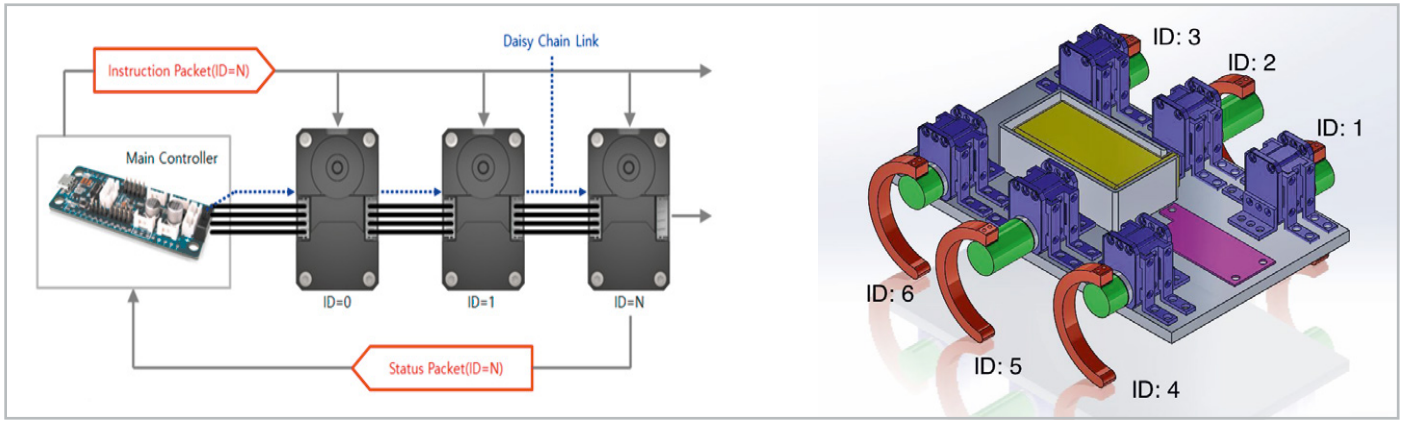

Bild 12: Die Dynamixel-Motoren werden im Daisy-Chain-Verfahren angesteuert, bekommen so eine ID und sind dann entsprechend am Aufbau, hier dem MiniRHex, ansteuerbar. Bild: Robotis/Carnegie Mellon University

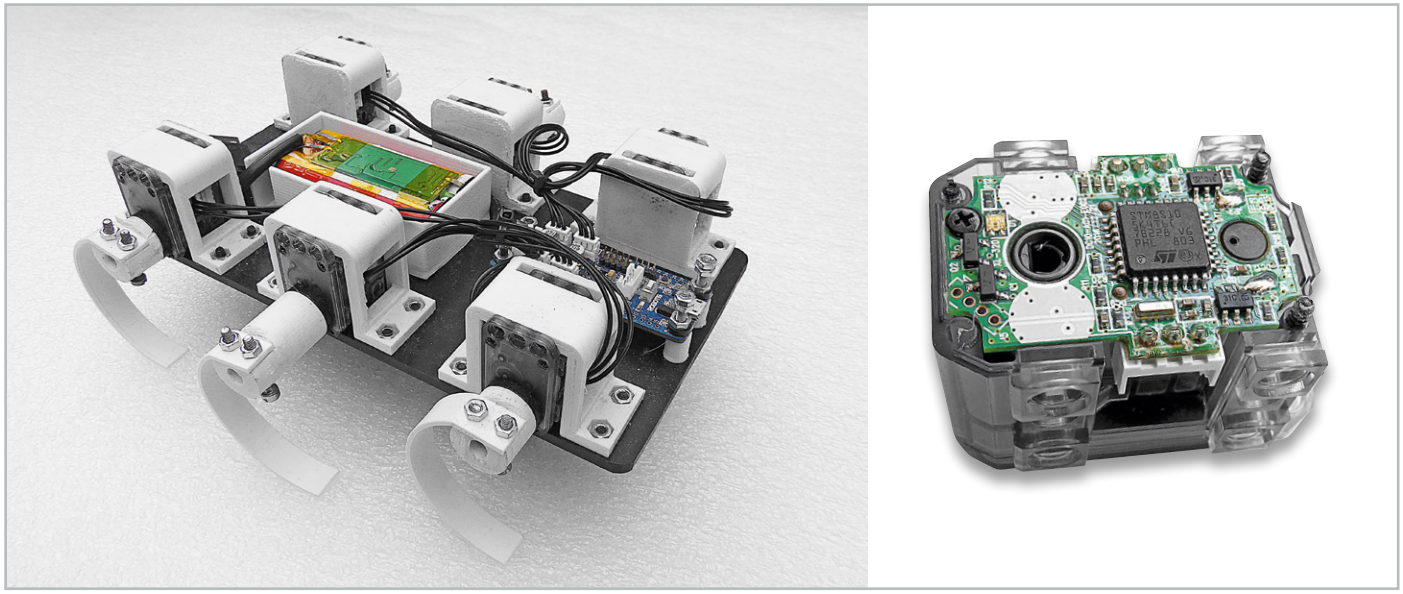

Bild 13: Der aufgebaute MiniRHex und die dabei verwendeten Dynamixel-Antriebe. Im Bild fehlt nur noch die Bluetooth-Option für die drahtlose Fernsteuerung. Rechts ein einzelner Antrieb mit der integrierten Regelelektronik.

Fazit

Fazit unseres kurzen Exkurses durch die Welt der Regler: Hat man sich das Grundverständnis für die Funktion und die Parametrierung der Regler einmal erarbeitet, ist die Realisierung eigener Lösungen relativ einfach. Je nach Aufgabe sind dabei nicht immer aufwendige Berechnungen nötig, manchmal kommt man auch mit der „Trial-and-Error"-Methode zu einer Lösung. ELV

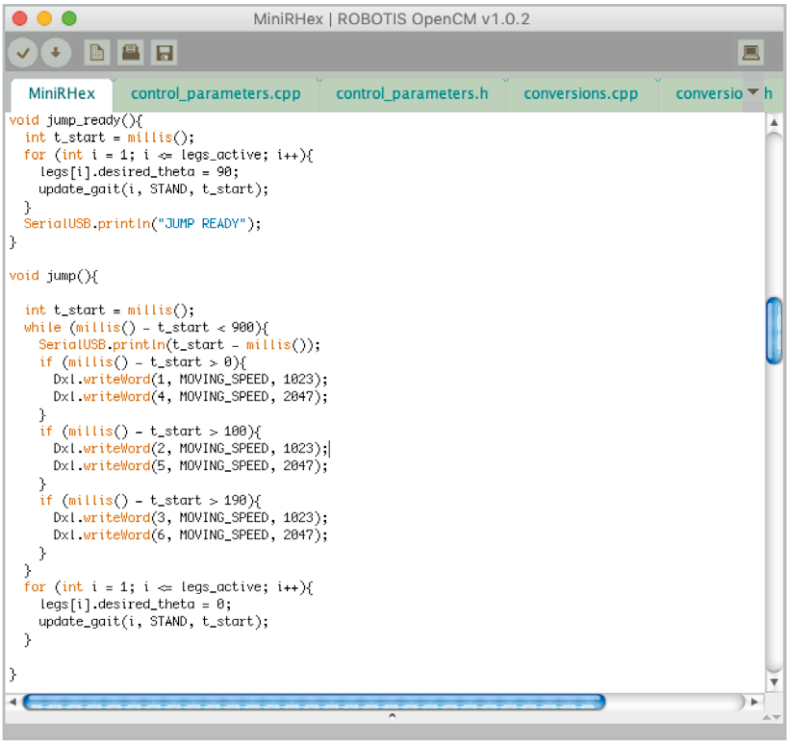

Bild 14: Ein Ausschnitt aus dem Open Source-Programm der Carnegie Mellon University zum MiniRHex zeigt die Parametrierung für einen Sprung des Roboters.