DIS10

Digitale Systemen

HOGESCHOOL ROTTERDAM

Dictaat Digitale Regelsystemen

Versie 1.2

E.H.W. van de Logt
E.M. van den Bergh
J.W. Peltenburg
J.Z.M. Broeders

Versiehistorie

Datum	Versie	Omschrijving	Auteur
04-06-2025	1.2[1]	Open-loop-methoden beter uitgelegd. Issue #20 verwerkt.	BroJZ
08-05-2024	1.1a	Symbolen consequenter gebruikt, bijvoorbeeld n voor samplenummer. Meer uitleg bij numerieke methoden. Verbeterde uitleg nut snelheidsalgoritme. Issue #19 verwerkt.	BroJZ
10-05-2019	1.0	Eerste versie voor DIS10. Voor voorgaande versiehistorie zie DCS01.	BroJZ

[1] Toelichting versiecodering $A.Bc$: A = grote aanpassing, B = kleine aanpassing, c = taal- of wiskundige correcties.

Inhoudsopgave

1

Introductie

Dit dictaat is onderdeel van het vak DIS10 (Digitale Systemen) en is bestemd voor tweedejaars studenten van de opleiding Elektrotechniek. In deze module wordt voortgebouwd op de theorie van het basisvak Regeltechniek (REG10) en op de theorie die al eerder bij Digitale Systemen (DIS10) is behandeld. Waar we bij REG10 nog de nadruk gelegd hebben op de regeltheorie zelf, wordt in dit dictaat de nadruk gelegd op de praktijk: de student is na afloop van DIS10 in staat om regelsystemen te ontwerpen, te realiseren met behulp van een microcontroller en optimaal in te stellen. Met name het optimaal kunnen instellen van een regelaar is een belangrijke competentie voor een HBO-ingenieur. In de beroepspraktijk krijgt deze vaak te maken met regelaars, die tegenwoordig overwegend digitaal zijn.

Dit document bevat de volgende hoofdstukken:

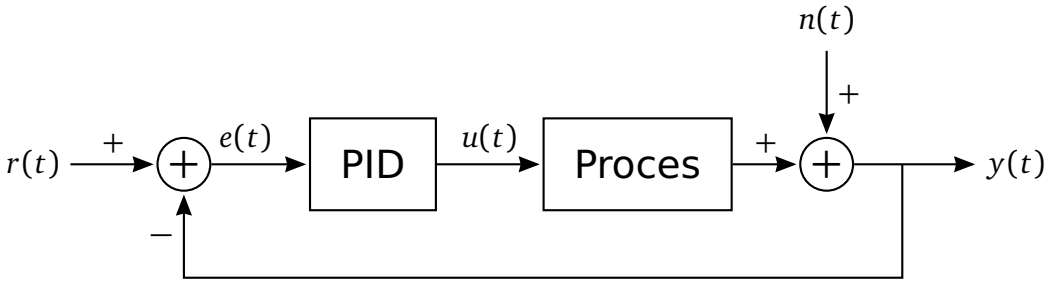
- Hoofdstuk 2: digitale PID-regelaars. Hier worden diverse afleidingen gegeven van discrete PID-regelaars, die afgeleid zijn van de ideale (tekstboek versie) tijdcontinue PID-regelaar.
- Hoofdstuk 3: verbetering aan digitale PID-regelaars. Dit hoofdstuk bevat een aantal verbeteringen die toegevoegd kunnen worden aan de discrete PID-regelaars, zoals die in hoofdstuk 2 afgeleid zijn.
- Hoofdstuk 4: Vinden van de optimale parameters. Hier worden diverse methoden en algoritmen beschreven om de optimale PID-parameters te vinden.

2

Digitale PID-regelaars

Een PID-regelaar probeert de uitgangswaarde van een te regelen proces (in de Engelse literatuur kom je het woord ‘plant’[2] hiervoor regelmatig tegen) op een vooraf ingestelde waarde te houden. Deze waarde wordt ook wel de referentiewaarde of het setpoint (SP) genoemd. De regelaar doet dit door de afwijking van de gewenste waarde te bepalen. Dit signaal noemt men vaak het ‘error’-signaal. Deze errorwaarde is gedefinieerd als het verschil tussen het setpoint en de uitgangswaarde (of ‘output’) van het te regelen proces (bijv. een temperatuurwaarde of een waterhoogte). Deze uitgangswaarde wordt ook vaak de procesvariabele genoemd. In de praktijk bevat de uitgangswaarde ook wat ruis. In sommige systemen wordt hier rekening mee gehouden, maar in theoretische besprekingen van de basisprincipes van regelaars wordt dit vaak weggelaten. Hoe groter de errorwaarde, des te meer zal de regelaar bij moeten sturen. De meest gebruikte regelaar in de industrie is ontegenzeggelijk de PID-regelaar. Hierbij staat de afkorting PID voor Proportioneel, Integrerend en Differentiërend. Met behulp van een drietal parameters (hierover later meer) kan de complete PID-regelaar ingesteld worden.

We kunnen een typisch PID-geregeld proces weergeven met het wiskundige model van figuur 2.1.



Figuur 2.1: Algemeen model van een PID-geregeld proces.

In dit model is:

t	tijdvariabele;
$r(t)$	referentiewaarde oftewel setpoint;
$e(t)$	afwijking of error t.o.v. de gewenste output;

[2] In de betekenis van ‘fabriek’.

$u(t)$ output van de regelaar oftewel input van het proces;

$n(t)$ een ruissignaal oftewel noise;

$y(t)$ de uitgangswaarde, output oftewel procesvariabele.

Alle waarden die je in dit overzicht ziet zijn als functie van de tijd, t, gegeven.

De algemene vergelijking voor een PID-regelaar in het tijdcontinue domein wordt gegeven door:

$$u(t) = K_c\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)\,d\tau + T_d\frac{de(t)}{dt}\right) \tag{2.1}$$

Hierbij is:

K_c Proportionele versterkingsfactor;

T_i Tijdconstante integrerende versterking [s];

T_d Tijdconstante differentiërende versterking [s].

Door middel van een Laplace-transformatie kan deze tijdcontinue functie naar het s-domein getransformeerd worden:

$$U(s) = K_c\left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s\right)E(s) \tag{2.2}$$

Hierbij is s gelijk aan $\sigma + j\omega$. Uitgaande van vergelijking (2.2) kunnen we de overdrachtsfunctie bepalen van deze PID-regelaar in het Laplace-domein ($H_r(s)$):

$$H_r(s) = \frac{U(s)}{E(s)} = K_c\left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s\right) \tag{2.3}$$

Digitale regelaars werken in het tijddiscrete domein in plaats van het tijdcontinue domein, omdat ze werken op samples van signalen en niet met het directe signaal zelf.

Om een tijddiscrete versie te krijgen van deze tijdcontinue PID-regelaar, moeten we de integrerende en differentiërende componenten van vergelijking (2.1) tijd-discreet maken. Hiervoor maken we eerst even een uitstapje naar twee deelparagrafen: eentje over numerieke integratie en eentje over numerieke differentiatie.

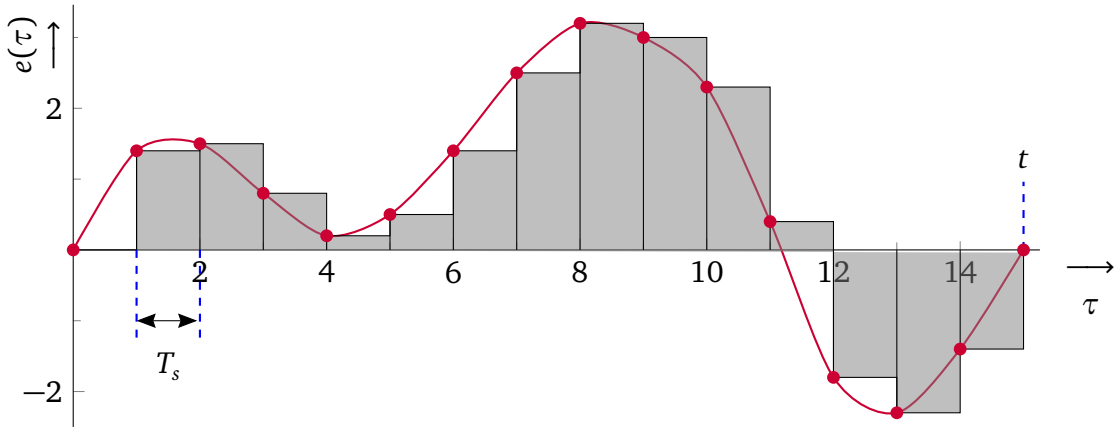
2.1 Numerieke integratie

In deze paragraaf gaan we de integraal uit de algemene vergelijking voor een PID-regelaar in het tijdcontinue domein, zie vergelijkingen (2.1) en (2.4), benaderen met een numerieke methode.

$$\int_0^t e(\tau)\,d\tau \tag{2.4}$$

We moeten het signaal $e(\tau)$[3] dus integreren van 0 tot t. Om deze integraal te berekenen in een digitaal systeem (bijvoorbeeld een microcontroller) moeten we het signaal eerst samplen. De gesamplelde waarden van $e(\tau)$ noemen we $e[\nu]$, met $\tau = \nu \cdot T_s$. Hierbij is T_s de sampletijd en ν het samplenummer. Als we zoals gebruikelijk definiëren dat $t = n \cdot T_s$ dan moeten we dus van samplenummer 0 tot samplenummer n integreren.

Uit de integraalrekening weten we dat het bepalen van een integraal overeenkomt met het bepalen van het oppervlak onder de functie waarover we integreren. Het oppervlak onder de curve kunnen we benaderen door van elk sample $e[\nu]$ een rechthoek te maken en het oppervlak van al deze rechthoeken bij elkaar op te tellen. In figuur 2.2 is dit concept weergegeven. De samples $e[\nu]$ zijn weergegeven als rode stippen.



Figuur 2.2: Forward Rectangular Method (FRM).

Het oppervlak onder een functie $e(\tau)$ kan als volgt benaderd worden:

$$\int_0^t e(\tau)\,d\tau \approx T_s \cdot e[0] + T_s \cdot e[1] + \cdots + T_s \cdot e[n-1] = T_s \sum_{\nu=1}^{n} e[\nu-1] \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \quad (2.5)$$

$n = \left\lfloor \frac{t}{T_s} \right\rfloor$ betekent dat bij deze formule n de naar beneden afgeronde waarde van $\frac{t}{T_s}$ is.

De aanpak voor het benaderen van de integraal gegeven in vergelijking (2.5) wordt ook wel de Forward Rectangular Method genoemd, of de 'ondersom' nemen van de functie.

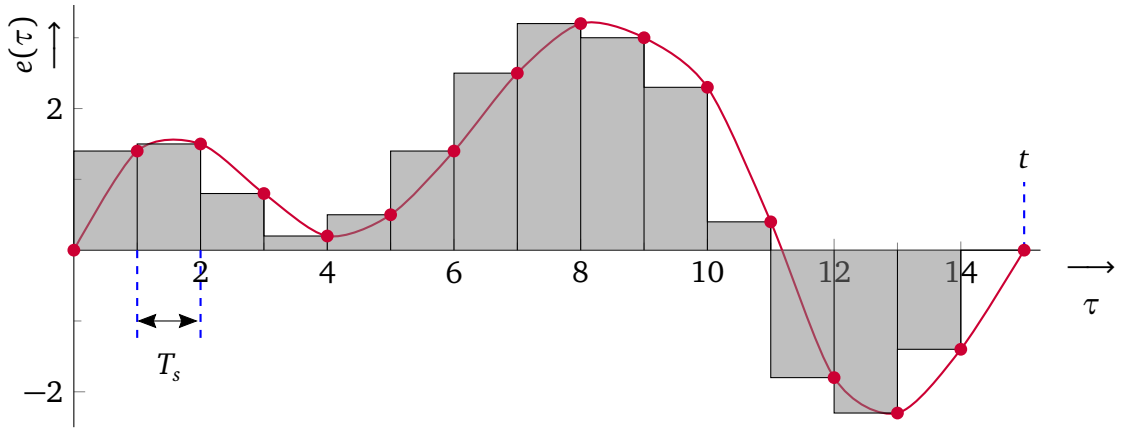
Een alternatieve manier om het oppervlak te bepalen is m.b.v. de 'bovensom'. Dit wordt ook wel de Backward Rectangular Method (BRM) genoemd. Dit is weergegeven in figuur 2.3.

Het oppervlak onder een functie $e(\tau)$ wordt bij deze methode als volgt benaderd:

$$\int_0^t e(\tau)\,d\tau \approx T_s \cdot e[1] + T_s \cdot e[2] + \cdots + T_s \cdot e[n] = T_s \sum_{\nu=1}^{n} e[\nu] \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \quad (2.6)$$

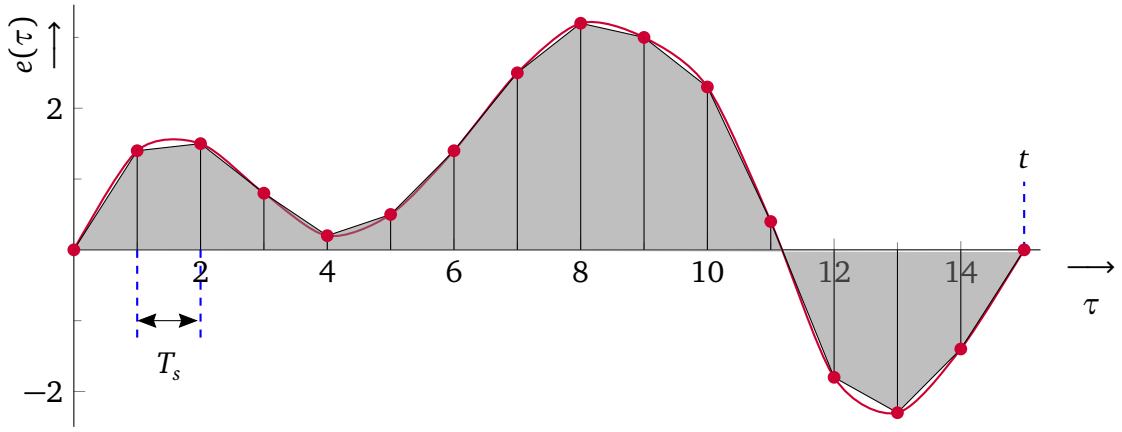
Naast deze twee methoden is er nog een derde methode die vaak gebruikt wordt. Dat is de 'trapeziummethode' (TRAP). Hierin wordt het oppervlak onder een stapje van T_s niet

[3] Omdat de integratiegrens in dit geval t is, moet als argument van de errorfunctie een andere letter gekozen worden. In dit geval is τ gebruikt, τ (tau) is de griekse letter t. De samples van het signaal $e(\tau)$ noemen we $e[\nu]$, ν (nu) is de griekse letter n. Waarbij geldt $\tau = \nu \cdot T_s$ met T_s de sampletijd en ν het samplenummer.



Figuur 2.3: Backward Rectangular Method (BRM).

benaderd met een rechthoekje maar met een rechthoekig trapezium. De oppervlakte van zo'n trapezium is gelijk aan het gemiddelde van 2 opeenvolgende functiewaarden maal de sampletijd. Dit is meestal nauwkeuriger dan het gebruik van de FRM en BRM. Dit is goed te zien in figuur 2.4.



Figuur 2.4: Trapeziummethode (TRAP).

Het oppervlak onder een functie $e(\tau)$ wordt bij deze methode als volgt benaderd:

$$\int_0^t e(\tau)\,d\tau \approx T_s \cdot \frac{e[1]+e[0]}{2} + T_s \cdot \frac{e[2]+e[1]}{2} + \cdots + T_s \cdot \frac{e[n]+e[n-1]}{2}$$
$$= \frac{T_s}{2} \sum_{\nu=1}^{n} e[\nu]+e[\nu-1] \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \tag{2.7}$$

Er zijn nog nauwkeurigere methoden om de integraal te benaderen [9], zoals de Simpson-regel. Deze methoden zijn echter complexer en worden in de praktijk niet vaak gebruikt bij het implementeren van digitale regelaars.

2.2 Numerieke Differentiatie

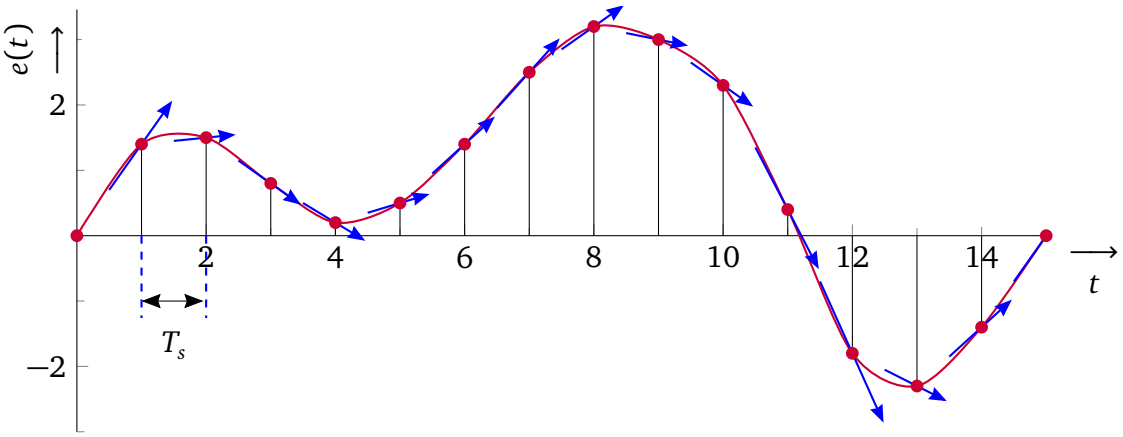
In deze paragraaf gaan we de afgeleide van $e(t)$ uit de algemene vergelijking voor een PID-regelaar in het tijdcontinue domein, zie vergelijkingen (2.1) en (2.8), benaderen met een numerieke methode.

$$\frac{de(t)}{dt} \tag{2.8}$$

Uit de differentiaalrekening weten we de afgeleide in een punt overeenkomt met de richtingscoëfficiënt van de raaklijn in dat punt. Als benadering nemen we het verschil van twee opeenvolgende samples van $e(t)$, dus twee opeenvolgende waarden van $e[n]$, en delen dat door het verschil van de bijbehorende waarden van t, de sampletijd T_s. In een formule:

$$e'(t) = \frac{de(t)}{dt} \approx \frac{e[n]-e[n-1]}{T_s} \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \tag{2.9}$$

Deze zogenoemde tweepuntsbenadering is zichtbaar in figuur 2.5.



Figuur 2.5: Numeriek Differentiëren m.b.v. de tweepuntsbenadering.

In plaats van de relatief eenvoudige tweepuntsbenadering voor het differentiëren (zie vergelijking (2.9)), kan ook gebruik gemaakt worden van een driepunts- of een vierpuntsbenadering voor het differentiëren[4]. In dat geval gaat vergelijking (2.9) over in:

Driepuntsbenadering:

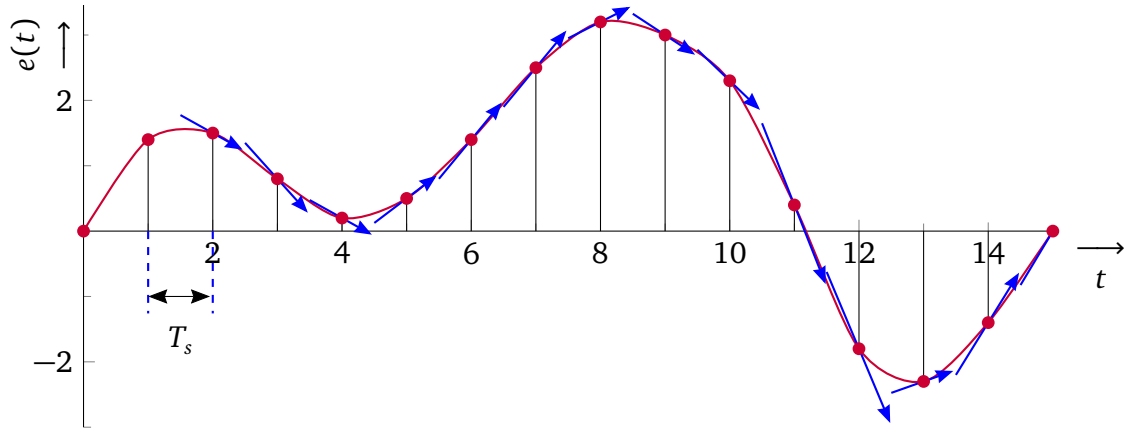
$$e'(t) = \frac{de(t)}{dt} \approx \frac{3\cdot e[n]-4\cdot e[n-1]+e[n-2]}{2\cdot T_s} \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \tag{2.10}$$

Deze driepuntsbenadering is zichtbaar in figuur 2.6.

Vierpuntsbenadering:

$$e'(t) = \frac{de(t)}{dt} \approx \frac{11\cdot e[n]-18\cdot e[n-1]+9\cdot e[n-2]-2\cdot e[n-3]}{6\cdot T_s} \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \tag{2.11}$$

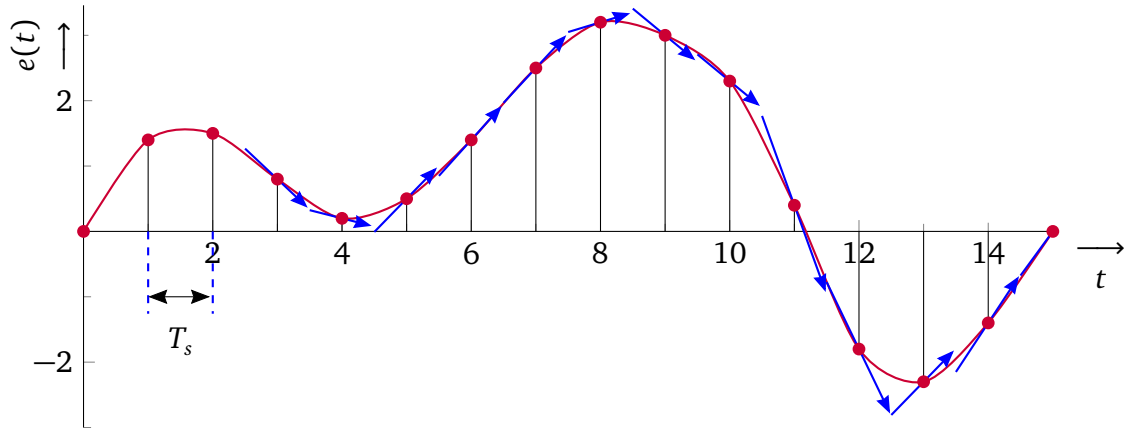
[4] Omdat we alleen over samples uit het verleden beschikken moeten we in dit geval de “backward approximations” gebruiken, zie https://en.wikipedia.org/wiki/Finite_difference_coefficient#Backward_finite_difference.



Figuur 2.6: Numeriek Differentiëren m.b.v. de driepuntsbenadering.

Deze vierpuntsbenadering is zichtbaar in figuur 2.7.

In de praktijk wordt bij digitale regelaars vaak de tweepuntsbenadering gebruikt omdat deze eenvoudig te implementeren is en voldoende nauwkeurig is voor de meeste toepassingen.



Figuur 2.7: Numeriek Differentiëren m.b.v. de vierpuntsbenadering.

2.3 Afleiding van een tijddiscrete PID-regelaar

Met behulp van de bovenstaande vergelijkingen voor numeriek integreren en differentiëren kunnen we de formule van een tijddiscrete PID-regelaar afleiden.

Als voorbeeld maken we gebruik van de Forward Rectangular Method (FRM) (vergelijking (2.5)) om de integrator om te zetten en numerieke differentiatie met de tweepuntsbenadering (vergelijking (2.9)) om de differentiator om te zetten.

Nogmaals gegeven de vergelijking van de uitgang van de regelaar vergelijking (2.1):

$$u(t) = K_c\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)\,d\tau + T_d\frac{de(t)}{dt}\right) \tag{2.1}$$

Als we dit samplen met sampletijd T_s en we passen de bovengenoemde numerieke methoden toe dan krijgen we:

$$u[n] = K_c\left(e[n] + \frac{T_s}{T_i}\sum_{\nu=1}^{n} e[\nu-1] + \frac{T_d}{T_s}\big(e[n]-e[n-1]\big)\right) \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \tag{2.12}$$

Waarbij n de tijdvariabele van het tijddiscrete systeem is, oftewel het samplenummer. De bovenstaande vergelijking (2.12) wordt ook wel het *positiealgoritme*[5] genoemd en kent in de praktijk een aantal nadelen.

Opdracht 1: Andere integratiemethodieken

Leid de uitgang van een tijddiscrete regelaar ook af met behulp van:

a) de BRM;

b) de TRAP-methode.

2.4 Het snelheidsalgoritme

Een nadeel van vergelijking (2.12), de tijddiscrete variant van de uitgang van de regelaar, is te vinden in het gedeelte van de integrator. Als we deze integrator letterlijk implementeren, dan moeten we alle voorgaande waarden van het errorsignaal $e[n]$ ($e[0]$ tot en met $e[n-1]$) opslaan. Dit komt omdat de formule nu zo geschreven is, dat het integrerende deel de som van alle voorgaande waarden van $e[n]$ bepaalt met behulp van een som. De som loopt over alle voorgaande waarden van $e[n]$. In de praktijk is het niet nodig om alle voorgaande waarden van $e[n]$ op te slaan. Het is voldoende om alleen de som van de voorgaande waarden van $e[n]$ bij te houden en daar telkens het laatste de nieuwe errorsample e[n] toe te voegen.

Een ander nadeel is dat een verandering in de proceswaarden K_c of T_i direct leidt tot een verandering in de waarde van de integrerende actie in zijn totaliteit. Dit kan bijvoorbeeld resulteren in overflows of andere dergelijke problemen. In de praktijk is dit vaak niet acceptabel omdat dit instabiliteit kan veroorzaken. [3, 4]

Om deze problemen tegen te gaan, kunnen we gebruik maken van een recursieve variant van vergelijking (2.12), die we het *snelheidsalgoritme*[6]. De afleiding van dit algoritme gaat als volgt.

In plaats van elke keer de nieuwe uitgang van de regelaar, $u[n]$, te bepalen, kunnen we ook de voorgaande waarde van $u[n]$ nemen, d.w.z. $u[n-1]$, en het verschil met de gewenste $u[n]$, genaamd $\Delta u[n]$, er bij optellen zodat:

$$u[n] = u[n-1] + \Delta u[n] \tag{2.13}$$

[5] In het Engels: 'positional algorithm'.

[6] In het Engels: 'velocity algorithm'.

We kunnen nu afleiden dat $\Delta u[n]$ gegeven wordt door:

$$\Delta u[n] = u[n] - u[n-1] \tag{2.14}$$

We nemen nu de $u[n]$ die afgeleid is met de FRM in vergelijking (2.12), nogmaals hieronder getoond:

$$u[n] = K_c\left(e[n] + \frac{T_s}{T_i}\sum_{\nu=1}^{n} e[\nu-1] + \frac{T_d}{T_s}\big(e[n]-e[n-1]\big)\right) \text{ met } n = \left\lfloor \frac{t}{T_s} \right\rfloor \tag{2.12}$$

Door in deze formule voor n telkens $n-1$ in te vullen vinden we $u[n-1]$

$$u[n-1] = K_c\left(e[n-1] + \frac{T_s}{T_i}\sum_{\nu=1}^{n-1} e[\nu-1] + \frac{T_d}{T_s}\big(e[n-1]-e[n-2]\big)\right) \tag{2.15}$$

Nu kunnen we met behulp van vergelijking (2.14) $\Delta u[n]$ bepalen:

$$\Delta u[n] = K_c\left(e[n]-e[n-1] + \frac{T_s}{T_i}e[n-1] + \frac{T_d}{T_s}\big(e[n]-2e[n-1]+e[n-2]\big)\right) \tag{2.16}$$

Nu dat $\Delta u[n]$ bekend is kunnen we dit weer invullen in vergelijking (2.13). De uitgang van de regelaar wordt nu gegeven door:

$$u[n] = u[n-1] + K_c\left(e[n]-e[n-1] + \frac{T_s}{T_i}e[n-1] + \frac{T_d}{T_s}\big(e[n]-2e[n-1]+e[n-2]\big)\right) \tag{2.17}$$

We zien hier een formule met een recursief gedeelte, namelijk $u[n-1]$. Dit heeft als voordeel dat we nu in de integrator geen somreeks met alle voorgaande samples meer terugvinden. De waarde van de integrator zit nu in $u[n-1]$ verwerkt.

Het snelheidsalgoritme kent ook een algemene, kortere notatie, namelijk:

$$u[n] = u[n-1] + q_0 e[n] + q_1 e[n-1] + q_2 e[n-2] \tag{2.18}$$

met:

$$q_0 = K_c\left(1 + \frac{T_d}{T_s}\right) \qquad q_1 = K_c\left(-1 + \frac{T_s}{T_i} - \frac{2T_d}{T_s}\right) \qquad q_2 = K_c\frac{T_d}{T_s} \tag{2.19}$$

Aangezien alle q's (nadat de regelaar is ingesteld) constanten zijn, lijkt vergelijking (2.18) veel op de (reeds bekende) algemene notatie voor een IIR-filter. De PID-regelaar in snelheidsalgoritme (oftewel de recursieve vorm) is dan ook een specifieke implementatie van een IIR-filter!

Er zijn nog een aantal andere aan de praktijk gerelateerde aspecten. Voor al deze implementaties geldt dat de sampletijd T_s klein moet zijn in verhouding tot de tijdconstante van het te regelen proces.

Verder is een belangrijke eigenschap van PID-regelaars dat ze relatief ongevoelig zouden moeten zijn voor wijzigingen in de PID-parameters. De hierboven beschreven regelaars zijn dat nog niet voldoende. Op deze zaken komen we later in dit document nog terug.

Opdracht 2: Het snelheidsalgoritme.

a) Reken de afleiding van $\Delta u[n]$ na en controleer dat vergelijking (2.16) klopt.

b) Pas het snelheidsalgoritme ook toe op de formule voor $u[n]$ die je bij opdracht 1a met behulp van de BRM hebt afgeleid.

c) Pas het snelheidsalgoritme ook toe op de formule voor $u[n]$ die je bij opdracht 1b met behulp van de TRAP-methode hebt afgeleid.

d) Waarom is het snelheidsalgoritme handiger om in software te implementeren dan het positiealgoritme?

Tabel 2.1: Antwoorden van opdracht 2.

Coëfficiënten	FRM	BRM	TRAP
q_0	$K_c\left(1+\frac{T_d}{T_s}\right)$	$K_c\left(1+\frac{T_s}{T_i}+\frac{T_d}{T_s}\right)$	$K_c\left(1+\frac{T_s}{2T_i}+\frac{T_d}{T_s}\right)$
q_1	$K_c\left(-1+\frac{T_s}{T_i}-\frac{2T_d}{T_s}\right)$	$K_c\left(-1-\frac{2T_d}{T_s}\right)$	$K_c\left(-1+\frac{T_s}{2T_i}-\frac{2T_d}{T_s}\right)$
q_2	$K_c\frac{T_d}{T_s}$	$K_c\frac{T_d}{T_s}$	$K_c\frac{T_d}{T_s}$

2.5 Directe transformatiemethoden

Een andere methode om van de tijdcontinue variant naar een tijddiscrete variant van de PID-regelaar over te gaan is door de Laplace-getransformeerde van de PID-regelaar direct om te zetten naar het z-domein. Hiervoor kunnen we verschillende methoden gebruiken waarvan we er hier drie beschrijven.

De meest gebruikte methode is de Bilineaire Transformatie (BLT). In de literatuur, maar ook in MATLAB, wordt deze transformatie ook wel de 'Tustin'-transformatie genoemd. Twee andere methoden zijn de 'Backward Euler'- en 'Forward Euler'-methode.

De BLT heeft de volgende vorm:

$$s \approx \frac{2}{T_s}\cdot\frac{z-1}{z+1} = \frac{2}{T_s}\cdot\frac{1-z^{-1}}{1+z^{-1}} \tag{2.20}$$

De Backward Euler gebruikt de volgende substitutie:

$$s \approx \frac{z-1}{z\cdot T_s} \tag{2.21}$$

De Forward Euler gaat als volgt:

$$s \approx \frac{z-1}{T_s} \tag{2.22}$$

Als we bijvoorbeeld terugkijken naar vergelijking (2.3):

$$H_r(s) = \frac{U(s)}{E(s)} = K_c\left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s\right) \tag{2.3}$$

Dan kunnen we nu s substitueren met een van de bovenstaande vormen. Als voorbeeld gebruiken we de Backward-Euler-methode. We kunnen nu de overdracht bepalen van de regelaar in het z-domein door elke s in vergelijking (2.3) te vervangen door $\frac{z-1}{z \cdot T_s}$:

$$H_r(z) = K_c\left(1 + \frac{1}{T_i \cdot \frac{z-1}{z \cdot T_s}} + T_d \cdot \frac{z-1}{z \cdot T_s}\right) \tag{2.23}$$

Omdat we uiteindelijk de formule in de vorm $U(z) = \ldots$ willen hebben, proberen we alles binnen een enkele breuk te krijgen, aangezien $H_r(z) = \frac{U(z)}{E(z)}$.

Dubbele breuk wegwerken en constanten naar voren halen:

$$H_r(z) = K_c\left(1 + \frac{T_s}{T_i} \cdot \frac{z}{z-1} + \frac{T_d}{T_s} \cdot \frac{z-1}{z}\right) \tag{2.24}$$

Omwerken naar gemeenschappelijke noemer:

$$H_r(z) = K_c\left(1 + \frac{T_s}{T_i} \cdot \frac{1}{1-z^{-1}} + \frac{T_d}{T_s} \cdot \left(1-z^{-1}\right)\right) \tag{2.25}$$

$$= K_c\left(1 + \frac{\frac{T_s}{T_i}}{1-z^{-1}} + \frac{\frac{T_d}{T_s} \cdot \left(1-z^{-1}\right)}{1}\right) \tag{2.26}$$

$$= K_c\left(\frac{1-z^{-1}}{1-z^{-1}} + \frac{\frac{T_s}{T_i}}{1-z^{-1}} + \frac{\frac{T_d}{T_s} \cdot \left(1-z^{-1}\right)^2}{1-z^{-1}}\right) \tag{2.27}$$

$$= K_c\left(\frac{1-z^{-1} + \frac{T_s}{T_i} + \frac{T_d}{T_s} \cdot \left(1-2z^{-1}+z^{-2}\right)}{1-z^{-1}}\right) \tag{2.28}$$

Dus:

$$H_r(z) = \frac{U(z)}{E(z)} = K_c\left(\frac{1-z^{-1} + \frac{T_s}{T_i} + \frac{T_d}{T_s} \cdot \left(1-2z^{-1}+z^{-2}\right)}{1-z^{-1}}\right) \tag{2.29}$$

Als we kruislings vermenigvuldigen vinden we:

$$U(z)-U(z)\cdot z^{-1} = K_c\left(E(z)-E(z)\cdot z^{-1}+\frac{T_s}{T_i}\cdot E(z)+\frac{T_d}{T_s}\cdot\left(E(z)-2E(z)\cdot z^{-1}+E(z)\cdot z^{-2}\right)\right) \quad (2.30)$$

Terugtransformeren van het z-domein naar het tijddomein geeft:

$$u[n]-u[n-1] = K_c\left(e[n]-e[n-1]+\frac{T_s}{T_i}\cdot e[n]+\frac{T_d}{T_s}\cdot\left(e[n]-2e[n-1]+e[n-2]\right)\right) \quad (2.31)$$

Deze vergelijking kan ook geschreven worden als:

$$u[n] = u[n-1]+q_0e[n]+q_1e[n-1]+q_2e[n-2] \quad (2.32)$$

met:

$$q_0 = K_c\left(1+\frac{T_s}{T_i}+\frac{T_d}{T_s}\right) \qquad q_1 = K_c\left(-1-\frac{2T_d}{T_s}\right) \qquad q_2 = K_c\frac{T_d}{T_s} \quad (2.33)$$

Wat valt je op aan vergelijking (2.32) en vergelijking (2.33) in vergelijking met vergelijking (2.18) en de bij opdracht 2 gevonden waarden voor q_0, q_1 en q_2?

Opdracht 3: Forward Euler en BLT.

Maak de bovenstaande afleiding ook met:

a) de Forward-Euler-methode (je kunt een pool toevoegen in de oorsprong om de overdracht causaal te maken);

b) de Bilineaire-Transformatie-methode.

Ga daarbij uit van vergelijking (2.3).

2.6 Samenvatting

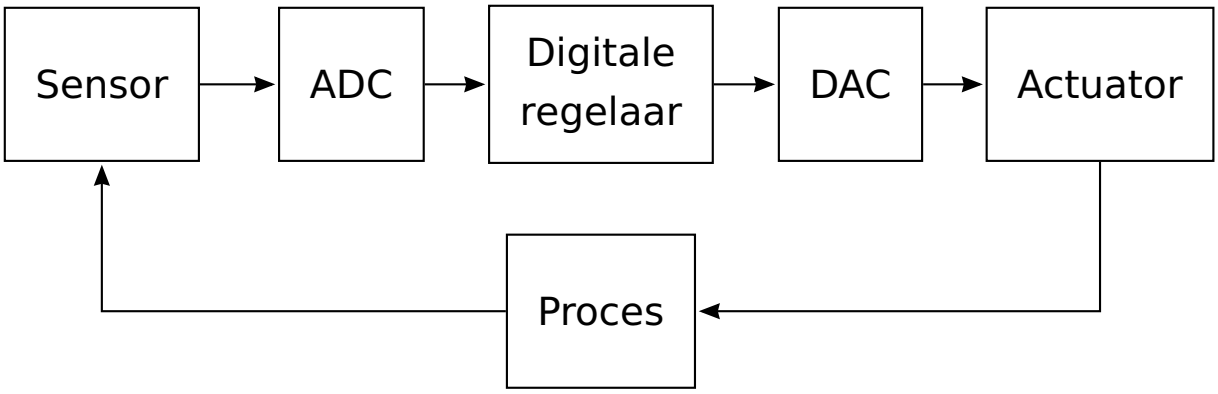
In dit hoofdstuk hebben we gezien hoe we van de tijdcontinue variant van een PID-regelaar kunnen overgaan naar een tijddiscrete variant van een PID-regelaar. Met behulp van numerieke methoden kunnen we het positiealgoritme van de PID-regelaar afleiden. Helaas kent deze vorm een aantal implementatiebeperkingen. Met behulp van het snelheidsalgoritme kunnen we een efficiëntere variant afleiden van de PID-regelaar. Verder kunnen we ook gebruik maken van directe transformatiemethodieken, zoals de Forward Euler, Backward Euler en de Bilineaire Transformatie. Deze methodieken komen van pas in het volgende hoofdstuk, als we verbeteringen aan gaan brengen aan de regelaars.

3

Verbeteringen aan digitale PID-regelaars

Vergelijking (2.1) beschrijft de algemene vergelijking voor een PID-regelaar in het tijdcontinue domein. In het tijddiscrete domein komt dit overeen met het doorrekenen van de bijbehorende differentievergelijking. Bij de praktische implementatie van zo'n regelaar lopen we tegen problemen aan, die bij een analoge regelaar niet altijd op zullen treden.

Een model van een typisch digitaal geregeld proces is gegeven in figuur 3.1.

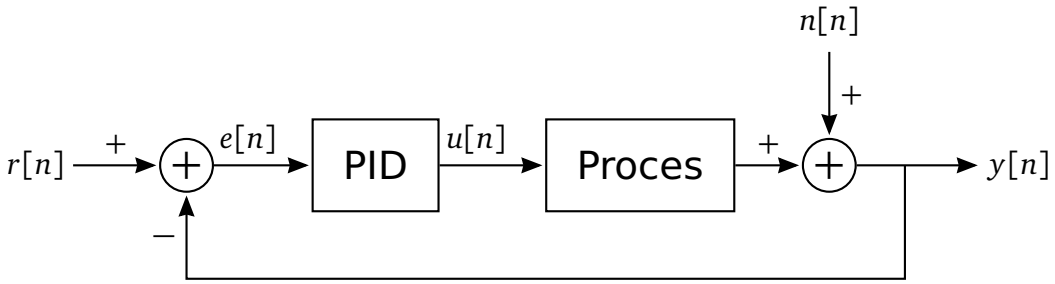


Figuur 3.1: Algemeen model van een digitaal geregeld proces.

Een bepaalde eigenschap die we willen regelen van het proces wordt gemeten d.m.v. een sensor. Deze waarde wordt door een ADC gesampled. Een actuator zorgt voor de nieuwe input van het proces. De actuator wordt aangestuurd vanuit een DAC, die de proceswaarde vanuit de digitale regelaar ontvangt. Vanuit de regelaar gezien is dankzij de DAC en de ADC de hele omgeving tijddiscreet geworden, en kunnen we het in figuur 2.1 getoonde model tekenen zoals in figuur 3.2 gegeven is.

Bij een analoge regelaar zijn de meeste terugkoppellussen op een elektrische manier begrenst in bandbreedte. Hierdoor is het niet mogelijk om zeer grote en snelle veranderingen in bijvoorbeeld het foutsignaal te krijgen. In een digitale regelaar zijn deze snelle veranderingen wel mogelijk, waardoor deze regeling instabiel gedrag kan vertonen.

Het berekenen van de differentievergelijking gebeurt volgens vergelijking (2.32). Grote verandering in bijvoorbeeld een variabele of een parameter kunnen leiden tot grote wijzigingen



Figuur 3.2: Model van een tijddiscreet PID-geregeld proces

in de output van de PID-regelaar. Wat verder ook nog meespeelt, is dat de ingelezen waarde $y[n]$ vaak beïnvloed is door ruis $n[n]$.

Het negatieve effect van grote en snelle wijzigingen in het foutsignaal kan onderdrukt worden door het errorsignaal $e[n]$ een specifieke behandeling te geven, alvorens het door de PID-regelaar gebruikt gaat worden ('preprocessing'). Een andere mogelijkheid is het aanpassen van de differentievergelijking van de PID-regelaar, zodat een deel van deze preprocessing direct in het algoritme van de PID-regelaar gebeurt. Deze mogelijkheden worden in dit hoofdstuk verder uitgewerkt.

Aan het eind van dit hoofdstuk komen nog enkele praktische aspecten aan bod.

3.1 Filteren van de differentiërende actie

Vaak zijn er verstoringen in de gemeten waarden van de procesoutput $y[n]$. Dit zijn vaak relatief hoge frequentiecomponenten. Indien een D-actie gebruikt wordt in de PID-regelaar, dan reageert deze op veranderingen in het errorsignaal. Daarmee wordt deze actie dan ook gevoelig voor deze hoge frequentiecomponenten. Dit kan weer leiden tot een te grote uitsturing aan de uitgang van de PID-regelaar. Dit kan soms een ongewenst effect geven.

De D-actie kan begrensd worden, om een te grote uitsturing te voorkomen. Meer gebruikelijk is het toevoegen van een eenvoudig 1ste- of 2de-orde laagdoorlaatfilter, zodat de versterking lager wordt voor hogere frequenties.

We gaan uit van de in vergelijking (2.3) gegeven Laplace-getransformeerde van de overdracht van een PID-regelaar:

$$H_r(s) = \frac{U(s)}{E(s)} = K_c\left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s\right) \tag{2.3}$$

We kennen de overdracht in het Laplace-domein van een laagdoorlaatfilter (**l**ow**p**ass):

$$H_{lp}(s) = \frac{1}{1 + \gamma \cdot s} \tag{3.1}$$

Als we dit toevoegen aan de D-actie van vergelijking (2.3) resulteert dit in:

$$H_{rl}(s) = \frac{U(s)}{E(s)} = K_c\left(1 + \frac{1}{T_i \cdot s} + T_d \cdot \frac{s}{1 + \gamma \cdot s}\right) \tag{3.2}$$

Hierbij is γ een kleine tijdconstante die in de praktijk meestal ingesteld wordt op ongeveer 10% van de T_d-waarde (bij de meeste industriële PID-regelaars varieert γ tussen 6,25% en 12,5% van de waarde van T_d [1]).

Als we vergelijking (3.2) willen implementeren in een digitale regelaar dan moet deze vergelijking getransformeerd worden naar het z-domein. Als voorbeeld hebben we hier gebruik gemaakt van de directe transformatie via de Backward-Euler-methode, zie vergelijking (2.21).

Uiteindelijk komen we bij deze afleiding uit op de volgende overdracht:

$$H_r(z) = K_c \left(\frac{\gamma + T_s + \frac{\gamma T_s}{T_i} + \frac{T_s^2}{T_i} + T_d - z^{-1}\left(2\gamma + T_s + \frac{\gamma T_s}{T_i} + 2T_d\right) + z^{-2}(T_d + \gamma)}{\gamma + T_s - z^{-1}(T_s + 2\gamma) + \gamma z^{-2}} \right) \tag{3.3}$$

Door dezelfde stappen toe te passen als in de voorgaande voorbeelden kunnen we deze overdracht terugtransformeren naar het tijddiscrete domein. Let er op dat in de bovenstaande vergelijking onder de deelstreep nu factoren bij de verschillende macht-termen van z staan. Als we dit omwerken naar de algemene vorm krijgen we ook coëfficiënten in het recursieve deel. Deze noemen we p.

De uitgang van de regelaar wordt nu gegeven door:

$$u[n] = p_1 \cdot u[n-1] + p_2 \cdot u[n-2] + q_0 \cdot e[n] + q_1 \cdot e[n-1] + q_2 \cdot e[n-2] \tag{3.4}$$

Met de volgende coëfficiënten:

$$p_1 = \frac{T_s + 2\gamma}{T_s + \gamma} \qquad p_2 = \frac{-\gamma}{T_s + \gamma}$$

$$q_0 = K_c\left(1 + \frac{T_s}{T_i} + \frac{T_d}{T_s + \gamma}\right) \qquad q_1 = \frac{-K_c}{T_s + \gamma}\left(T_s + \gamma\left(2 + \frac{T_s}{T_i}\right) + 2T_d\right)$$

$$q_2 = K_c \frac{T_d + \gamma}{T_s + \gamma} \tag{3.5}$$

Als we in plaats van de Backward-Euler-methode de bilineaire directe transformatiemethode toepassen (zie vergelijking (2.20)), dan krijgen we de volgende coëfficiënten:

$$p_1 = \frac{4\gamma}{T_s + 2\gamma} \qquad p_2 = \frac{T_s - 2\gamma}{T_s + 2\gamma}$$

$$q_0 = K_c\left(1 + \frac{T_s}{2T_i} + \frac{2T_d}{T_s + 2\gamma}\right) \qquad q_1 = K_c\left(\frac{\frac{T_s^2}{T_i} - 4\gamma - 4T_d}{T_s + 2\gamma}\right)$$

$$q_2 = K_c \frac{2\gamma - T_s + \frac{T_s^2}{2T_i} - \frac{\gamma T_s}{T_i} + 2T_d}{2\gamma + T_s} \tag{3.6}$$

Opdracht 4: LPF in de D-actie

Transformeer nu vergelijking (3.2) zelf naar het tijddiscrete domein met behulp van de directe transformatie volgens

a) de Backward-Euler-methode. Als het goed is levert dit vergelijking (3.4) op met de coëfficiënten uit vergelijking (3.5);

b) de Forward-Euler-methode;

c) de bilineaire methode. Als het goed is levert dit vergelijking (3.4) op met de coëfficiënten uit vergelijking (3.6).

3.2 PID-regelaar type A, B en C

In paragraaf 2.5 hebben we het snelheidsalgoritme van een PID-regelaar met behulp van Backward Euler afgeleid. De differentievergelijking werd toen:

$$u[n]-u[n-1] = K_c\left(e[n]-e[n-1]+\frac{T_s}{T_i}\cdot e[n]+\frac{T_d}{T_s}\cdot\big(e[n]-2e[n-1]+e[n-2]\big)\right) \tag{2.31}$$

Hierbij is $e[n]$ het errorsignaal. Dit is gelijk aan het setpoint (de referentiewaarde) minus de gemeten proceswaarde $y[n]$. Het probleem bij deze implementatie van een PID-regelaar is dat, als het setpoint op een andere waarde ingesteld wordt, dit direct invloed heeft op de grootte van de proportionele actie, maar ook op de grootte van de differentiërende actie. Hele snelle veranderingen in het setpoint kunnen dus voor ongewenste effecten in de P- en D-actie zorgen! Een type regelaar waarbij dit probleem niet verholpen is (zoals in vergelijking (2.31)) wordt ook wel een 'type A regelaar' genoemd[7].

Deze type A regelaar kan minder gevoelig gemaakt worden voor setpoint veranderingen door het setpoint te verwijderen uit de differentiërende actie. Dit kunnen we als volgt beredeneren; omdat we alleen willen regelen met veranderingen op de lange termijn, kunnen we aannemen dat r[n] (het setpoint) over een korte periode hetzelfde blijft.

Dit betekent dat we kunnen aannemen dat: $r[n] \approx r[n-1] \approx r[n-2] \approx \cdots$.

Aangezien $e[n] = r[n] - y[n]$ kunnen we een substitutie maken in de D-actie. Nu gaat vergelijking (2.31) over in:

$$u[n]-u[n-1] = K_c\left(e[n]-e[n-1]+\frac{T_s}{T_i}e[n]\right)+$$

[7] Zie http://bestune.50megs.com/typeABC.htm.

$$K_c\left(\frac{T_d}{T_s}\Big(r[n]-y[n]-2\big(r[n-1]-y[n-1]\big)+\big(r[n-2]-y[n-2]\big)\Big)\right) \quad (3.7)$$

Observeer nogmaals dat $r[n] \approx r[n-1] \approx r[n-2] \approx \cdots$. We kunnen nu een aantal $r[n]$'s in de D-actie tegen elkaar wegstrepen. Dit resulteert in het wegvallen van het setpoint uit de D-actie. Deze vorm wordt ook wel een 'Type B regelaar' of 'PI-D-regelaar' genoemd [1, pagina 562]:

$$u[n]-u[n-1] = K_c\left(e[n]-e[n-1]+\frac{T_s}{T_i}e[n]+\frac{T_d}{T_s}\big(-y[n]+2y[n-1]-y[n-2]\big)\right) \quad (3.8)$$

We kunnen ditzelfde principe ook toepassen op de proportionele actie. In dat geval gaat de bovenstaande differentievergelijking over in:

$$u[n]-u[n-1] = K_c\left(-y[n]+y[n-1]+\frac{T_s}{T_i}e[n]+\frac{T_d}{T_s}\big(-y[n]+2y[n-1]-y[n-2]\big)\right) \quad (3.9)$$

Een regelaar waarbij het setpoint zowel uit de P- als de D-actie is weggehaald wordt een 'Type C regelaar' of 'I-PD-regelaar' genoemd [1, pagina 562].

3.3 Integrator wind-up

Een ander aandachtspunt bij het implementeren van digitale regelaars is het verschijnsel van de zogenoemde integrator wind-up. Gegeven vergelijking (2.12), die een discrete PID-regelaar voorstelt, die met behulp van FRM is bepaald:

$$u[n] = K_c\left(e[n]+\frac{T_s}{T_i}\sum_{\nu=1}^{n}e[\nu-1]+\frac{T_d}{T_s}\big(e[n]-e[n-1]\big)\right) \text{ met } n=\left\lfloor\frac{t}{T_s}\right\rfloor \quad (2.12)$$

Stel je de situatie voor dat het errorsignaal (om wat voor reden dan ook[8]) een flinke tijd positief is. Dit errorsignaal wordt geïntegreerd in de tijd. In het geval van vergelijking (2.12) betekent dit dat de som bepaald wordt van alle voorgaande errorwaarden. De integrerende actie kan in dit geval behoorlijk groot worden.

Het beste is om dit te illustreren met een voorbeeld: stel dat de errorwaarde $e[i]$ gelijk is aan 5 % voor de laatste 50 waarden. De totale somwaarde van de integrerende actie bedraagt in dit geval 50 * 5 % = 250 %.

Stel nu dat het errorsignaal $e[i]$ vervolgens net iets kleiner wordt dan 0, bijvoorbeeld −1 %. Dan duurt het nog minstens 150 samples voordat het integrerende deel van het outputsignaal weer terug bij 100 % is! Dit is een onacceptabel resultaat, dat kan leiden tot een langdurig overschot in het te regelen proces.

[8] In de praktijk gebeurt dit vaak door zogenoemde actuatorverzadiging. Het proces is dan niet in staat om de output van de regelaar te volgen omdat de actuator al maximaal aangestuurd wordt.

Dit fenomeen staat bekend als integrator wind-up. De oplossing hiervoor is om de totale waarde van de integrerende actie te beperken op de maximale waarde van het outputsignaal. In het voorbeeld heeft het dus zin om de integrerende actie te beperken tot 100 %.

3.4 Beperkte precisie

Binnen een digitaal systeem werken we niet alleen met discrete stappen in de tijd, maar ook met discrete stappen in de waarden van het setpoint, de procesvariabele, enz., omdat de waarden gerepresenteerd worden door een eindig aantal bits.

In de praktijk wordt vaak gerekend met zogenoemde fixed-point getallen omdat fixed-point berekeningen sneller zijn dan floating-point berekeningen en ook minder energie kosten. Bij het gebruik van fixed-point getallen moeten we goed opletten dat we de berekeningen met voldoende precisie uitvoeren.

Een voorbeeld: stel dat we gebruik maken van een 12-bit AD-converter. Iedere stap in het $e[n]$-signaal is nu $\frac{1}{2^{12}} = \frac{1}{4096}$ groot.

Stel nu dat we de volgende vergelijking willen uitrekenen: $\Delta u_i[n] = \frac{K_c \cdot T_s}{T_i} e[n]$, waarbij gegeven is dat $K_c \cdot T_s$ gelijk aan 1 is en T_i gelijk aan 3550 is.

Stel nu dat we dit naïef programmeren met behulp van 16-bits integer getallen als $\Delta u_i[n] = \frac{e[n]}{3550}$. Het errorsignaal zal dan minimaal 87% van de full-scale-waarde (4095 voor een 12-bit ADC) moeten bedragen om voor $\Delta u_i[n]$ een waarde groter dan 0 berekend te krijgen.

Een minder naïeve implementatie zal gebruik maken van 16-bits fixed-point getallen met 12 bits voor en 4 bits na de decimale punt. We moeten dan $e[n]$ eerst vermenigvuldigen met 16 (4 plaatsen naar links schuiven). Er is geen gevaar voor overflow omdat het $e[n]$-signaal afkomstig is van een 12-bit ADC. De berekening wordt dan uitgevoerd als $\Delta u_i[n] = \left(\frac{e[n]*16}{3550}\right)$. Het errorsignaal zal dan slechts minimaal 5,5% van de full-scale-waarde (4095 voor een 12-bit ADC) moeten bedragen om een waarde groter dan 0 berekend te krijgen. Als dit nog steeds niet acceptabel is zal met 32-bits fixed-point getallen gerekend moeten worden waarbij 20 bits voor en 12 bits na de decimale punt gebruikt worden. In dit geval kunnen we de berekening uitvoeren als: $\Delta u_i[n] = \left(\frac{e[n]*4096}{3550}\right)$ en wordt de berekening met de hoogst mogelijke precisie uitgevoerd.

Het werken met fixed-point berekeningen is soms onontkoombaar, bijv. omdat de snelheid of het energiegebruik zeer belangrijk is. Maar met het krachtiger worden van microcontrollers verdient het aanbeveling om te kijken of het mogelijk is om toch met floating-point berekeningen te werken. Het vereenvoudigt het implementeren van de regelaar in software aanzienlijk.

3.5 Industriële regelaars

De ontwikkeling van elektronica en in het bijzonder de computertechnologie (microcontrollers) heeft er in geresulteerd dat vrijwel alle commercieel verkrijgbare regelaars digitaal zijn en niet langer analoge elementen bevatten, zoals dat decennia geleden nog wel gebruikelijk was.

Praktische regelaars bestaan uit een of meerdere PID-regelaars. PID-regelaars zijn vaak beschikbaar als functieblokken in PLC's en regelsystemen. De output van zo'n regelaar is meestal analoog (analoge output, AO) of digitaal (digitale output, DO). Meestal wordt de digitale output gebruikt om een apparaat aan of uit te schakelen. Ook zijn er regelaars met 2 digitale outputs: hierbij wordt de ene output gebruikt om een apparaat aan of uit te schakelen en de andere output wordt gebruikt om de richting aan te geven (omhoog/omlaag, links/rechts, verwarmen/koelen etcetera).

Het analoge outputsignaal genereert meestal een spanningswaarde tussen 0 V en een maximale spanning. Het digitale outputsignaal is meestal een in pulsbreedte gemoduleerd signaal van een bepaalde frequentie. De waarde van de regelaar bepaalt de duty-cycle van het outputsignaal (bijv. als de output van de regelaar 20% is, dan is het signaal 20% van de periodetijd hoog en 80% van de periodetijd laag).

Naast deze standaard outputs is er meestal ook een communicatiepoort aanwezig. Dit is vaak een seriële interface.

Belangrijk bij iedere PID-regelaar zijn de instellingen. Naast de gebruikelijke waarden (K_c, T_i, T_d, T_s en γ voor het D-filter) kunnen vaak ook de boven- en ondergrens van het outputsignaal van de regelaar ingesteld worden.

Daarnaast kan de polariteit van de regelaar vaak ingesteld worden. Hiermee wordt in feite het teken van de versterking ingesteld (+ of -). Hiervoor worden de termen direct-acting (ook wel non-inverting of heating-loop genoemd) en reverse-acting (ook wel inverting of cooling-loop genoemd) gebruikt.

Bij een direct-acting controller is sprake van een positieve versterking. Dit houdt in dat wanneer de output van de PID-regelaar verhoogd wordt, de procesvariabele ook groter zal worden. Bij een verwarmingssysteem zal een grotere waarde van de output van de regelaar meestal leiden tot een hogere temperatuur.

Bij een reverse-acting controller is sprake van een negatieve versterking. Denk hier bijvoorbeeld aan een regelaar voor een koelkast. Indien de output van de PID-regelaar verhoogd wordt, zal de procesvariabele (de koelkast temperatuur) juist lager worden.

Andere veel voorkomende functies/instellingen van de huidige generatie PID-regelaars zijn:

- Kunnen schakelen tussen handmatige bediening en automatisch regelen.
- De mogelijkheid hebben om de regelaar te programmeren, waarbij het setpoint in de tijd geprogrammeerd kan worden (bijv. om een bepaalde cyclus af te lopen).
- De mogelijkheid om meerdere regelaars achter elkaar te zetten (cascade-loop).
- De mogelijkheid om de regelaar zelf de optimale PID-parameters te laten bepalen (auto-tuning).

4

Het vinden van de PID-parameters

Nu we de tijddiscrete regelaars hebben afgeleid en ook enkele interessante eigenschappen en verbeteringen aan de regelaars hebben besproken, rest de vraag: “Hoe stellen we de regelaar eigenlijk in?”. In dit hoofdstuk wordt hier op verschillende manieren (voor simpel te regelen systemen) antwoord op gegeven.

Bij het vinden van de optimale set parameters voor een digitale PID-regelaar denken we al snel aan moderne digitale signaalbewerking. Toch is de basis hiervoor reeds begin jaren ’40 gelegd, in een tijd dat er nog geen digitale regelaars waren. De regelaars in die tijd waren mechanische en pneumatische apparaten.

Het was in 1942 dat de heren Ziegler en Nichols hun wereldberoemde artikel publiceerden: “settings for automatic controllers” [12]. De inhoud van dit artikel wordt nog steeds gebruikt. Sterker nog: in de komende paragrafen zullen we de essentie van hun verhaal behandelen.

4.1 Stap-responsie van een eerste-orde-proces met tijdvertraging

Veel industriële processen, die geregeld dienen te worden, kunnen gemodelleerd worden als een eerste-orde-systeem met tijdvertraging (FOPTD = First Order Process with Time Delay). De overdrachtsfunctie van zo’n eerste-orde-systeem wordt dan voorgesteld door:

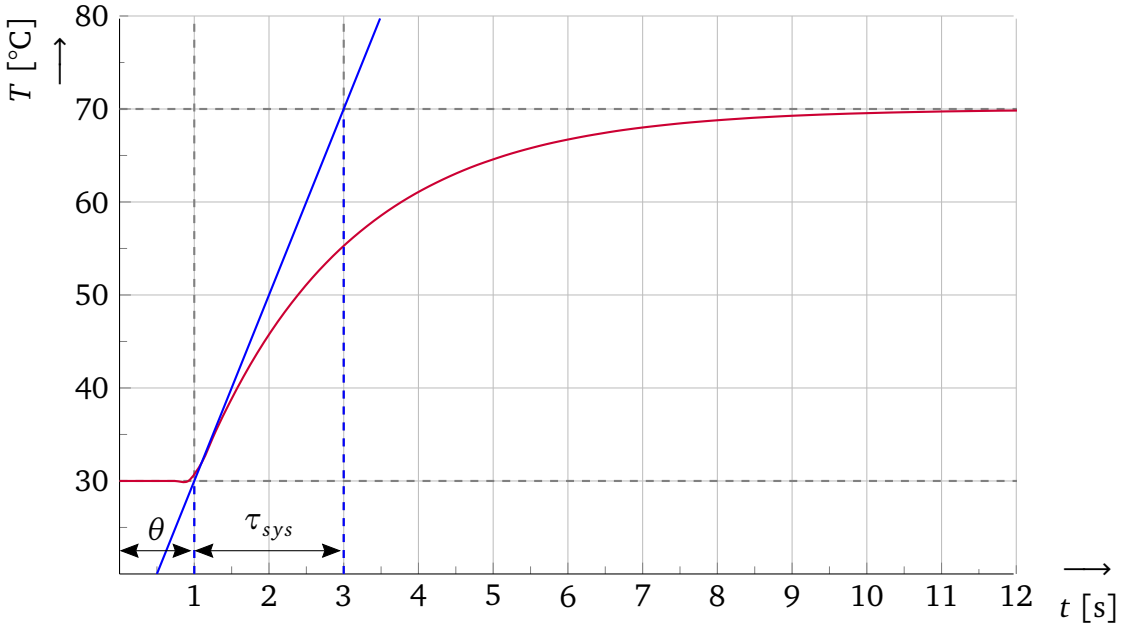
$$G(s) = \frac{Y(s)}{U(s)} = \frac{K_{sys} e^{-\theta s}}{\tau_{sys} s + 1} \tag{4.1}$$

Hierin is:

- $G(s)$: de overdrachtsfunctie van het te regelen proces.
- $Y(s)$: de output van het te regelen proces, dit wordt ook wel de procesvariabele PV genoemd. De eenheid hangt af van het te regelen proces. Stel bijvoorbeeld dat er een temperatuur geregeld moet worden dan is de eenheid bijvoorbeeld °C.
- $U(s)$: de output van de PID-regelaar en de input voor het te regelen proces. Deze waarde wordt vaak uitgedrukt in een percentage van het bereik van het outputsignaal (0 tot 100 %).

- Dode tijd θ: de tijdvertraging van het te regelen proces in seconden.
- K_{sys}: de versterking van het te regelen proces. De eenheid hangt af van het te regelen proces. Stel bijv. dat $Y(s)$ een temperatuur voorstelt, dan is de eenheid van K_{sys} gelijk aan °C/%.
- τ_{sys}: de tijdconstante van het te regelen proces in seconden.

De stapresponsie van een FOPTD met $K_{sys} = 1{,}016\,°\text{C}/\%$, $\tau_{sys} = 2\,\text{s}$ en $\theta = 1\,\text{s}$ op een stap van 27 % naar 63 % is gegeven in figuur 4.1.



Figuur 4.1: De stapresponsie van een FOPTD.

De dode tijd θ is eenvoudig uit de stapresponsie af te lezen. Je ziet dat in figuur 4.1 geldt $\theta = 1\,\text{s}$. Als we de raaklijn tekenen in het eindpunt van de dode tijd (de blauwe lijn in figuur 4.1) dan kunnen we ook τ_{sys} aflezen. Je ziet dat in figuur 4.1 geldt $\tau_{sys} = 2\,\text{s}$.

Bedenk dat vergelijking (4.1) een model is voor een proces dat we met een PID-regelaar willen regelen. Geen enkel praktisch proces zal zich exact zo gedragen als het model. In de volgende paragraaf zullen we de methode bespreken die bedacht is door Ziegler en Nichols waarmee de instellingen voor de PID-regelaar gevonden kunnen worden voor een proces waarvan de response *lijkt* op de response van een FOPTD.

4.2 Open-loop-methode van Ziegler-Nichols

Stel dat we een proces dat zich *ongeveer* gedraagt als een FOPTD willen regelen met een PID-regelaar. We kunnen de instellingen van de PID-regelaar volgens de open-loop-methode[9] van Ziegler en Nichols als volgt vinden[6]:

1. Zorg ervoor dat de regellus (zie figuur 2.1) doorbroken wordt. Dit kan in de praktijk gedaan worden door de PID-regelaar op handbediening te zetten.

[9] De Engelse benaming is 'open-loop method'. Dit kan in het Nederlands vertaald worden als 'openkringmethode'. Omdat de Engelse termen 'open-loop' in de regeltechniek, ook in het Nederlands, vaak voorkomt wordt in dit dictaat de term open-loop-methode gebruikt.

2. Regel het proces handmatig zodanig dat de uitgang een waarde heeft die in de praktijk vaak als setpoint wordt gebruikt en wacht tot de uitgangswaarde constant blijft.
3. Breng nu handmatig een stap op de output van de regelaar (dat is de input van het proces) aan en registreer de stapresponsie. Het kiezen van een geschikte stapgrootte is een kwestie van ervaring. Het verdient in principe de voorkeur om de stapgrootte zo groot mogelijk te nemen, dat is namelijk het meest nauwkeurig. Aan de andere kant willen we het proces dat we regelen in sommige gevallen ook niet te veel verstoren alleen maar om de regelaar in te kunnen stellen. We nemen als voorbeeld een temperatuurregeling (deze gedragen zich in de praktijk vrijwel altijd als een FOPTD) met een K_{sys} van 1,016 °C/% die we handmatig hebben ingesteld op 27 % wat dus overeenkomt met $27 \times 1{,}016 = 30\,^{\circ}\text{C}$. Vervolgens brengen we op $t = 0$ een stap aan van 27 % naar 63 %. De stapgroote in % noemen we Δp. In dit voorbeeld geldt dus $\Delta p = 63\,\% - 27\,\% = 36\,\%$. De stapresponsie is gegeven in figuur 4.2. Je ziet dat deze stapresponsie niet helemaal gelijk is aan de stapresponsie van een FOPTD (zie figuur 4.1) maar dat het er voldoende op lijkt om de FOPTD als model te gebruiken.
4. Uit de stapresponsie moeten we nu 2 waarden aflezen:
   - De genormaliseerde stijging van de stapresponsie. In dit document wordt deze grootheid aangeduid met a^*. Om deze te kunnen vinden moeten we eerst de stijging van de stapresponsie, aangeduid met a vinden. Deze kan gevonden worden door de helling te bepalen van de meest steile raaklijn aan de stapresponsie die we kunnen vinden[10]. In de tijd van Ziegler en Nichols was dit een kwestie van schatten, maar tegenwoordig kunnen we de stapresponse inlezen in een computer en de maximale waarde van de afgeleide opzoeken[11]. We zouden dit kunnen doen m.b.v. van MATLAB, maar moderne digitale regelaars kunnen dit ook zelf. De waarde van a^* is gelijk aan $a/\Delta p$.
   - De dode tijd θ: hiervoor nemen we de tijd tussen de start van de stap en het tijdstip waarop de meest steile raaklijn aan de stapresponsie de beginwaarde van de output snijdt.

Opdracht 5: Bepalen van a^* en θ

Bepaal zelf de waarde van a^* en θ uit de in figuur 4.2 gegeven stapresponsie bij een stap van 27 % naar 63 % op de input.

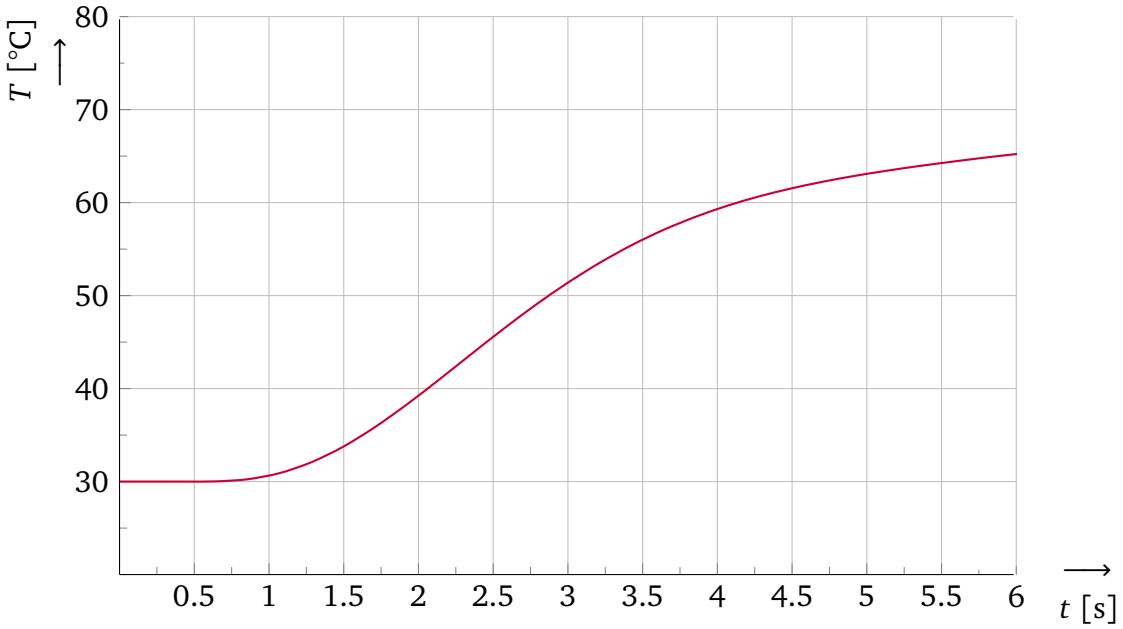
5. Op basis van deze waarden (a^* en θ) waren Ziegler en Nichols destijds al in staat om een PID- of PI-regelaar zodanig in te stellen, dat deze een goed regelgedrag liet zien. De PID-regelparameters en PI-regelparameters voor deze open-loop-methode van Ziegler-Nichols staan in tabel 4.1 vermeld.

[10] We zoeken dus de helling van de raaklijn in het buigpunt van de stapresponsie.

[11] We kunnen het tijdstip waarop de afgeleide maximaal is eenvoudig vinden door de tweede afgeleide van de stapresponsie gelijk te stellen aan nul.

Opdracht 6: Bepalen van de parameters van de PID-regelaar

Bepaal met behulp van de bij opdracht 5 bepaalde waarden van a^* en θ de parameters voor de PID-regelaar van dit proces volgens de open-loop-methode van Ziegler-Nichols.



Figuur 4.2: De stapresponsie van het te regelen proces.

Tabel 4.1: PID-parameters volgens de open-loop-methode van Ziegler-Nichols.

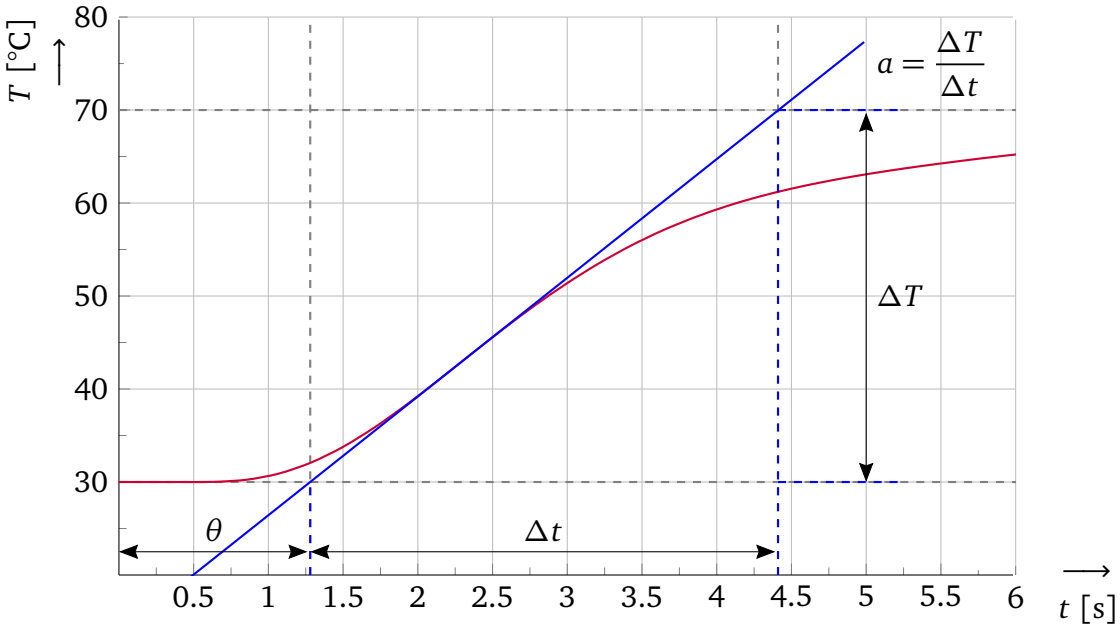
Methode / Parameters	K_c [%/?]	T_i [s]	T_d [s]
Ziegler-Nichols open-loop (PID)	$K_c = \frac{1{,}2}{\theta \cdot a^*}$	$T_i = 2{,}0 \cdot \theta$	$T_d = 0{,}5 \cdot \theta$
Ziegler-Nichols open-loop (PI)	$K_c = \frac{0{,}9}{\theta \cdot a^*}$	$T_i = 0{,}5 \cdot \theta$	–

Het voordeel van deze methode is de eenvoud. Door een stap op het systeem te zetten, kan eigenlijk al heel snel een redelijke instelling bepaald worden voor de PID-regelaar. Van dit principe maken veel auto-tuning functies in commercieel verkrijgbare regelaars dan ook dankbaar gebruik. Merk op dat het bij deze methode *niet* nodig is om de stapresponsie helemaal tot de eindwaarde te laten lopen. Zodra we het buigpunt gezien hebben kunnen we stoppen.

De Ziegler-Nichols-methode is erg eenvoudig om te gebruiken maar levert in de praktijk meestal niet het gewenste gedrag op [7, pagina 32-4]. De regelaar geeft in de meeste gevallen te weinig demping en de regelaar is te gevoelig voor variaties in de procesparameters. Om deze reden zijn er vele alternatieve methoden bedacht waarvan we er enkele zullen bespreken in paragraaf 4.4.

4.3 Antwoorden van opdrachten 5 en 6

In figuur 4.3 zie je hoe de waarden van a en θ gevonden kunnen worden met behulp van de raaklijn door het buigpunt van de stapresponsie.



Figuur 4.3: De stapresponsie van het te regelen proces met de raaklijn door het buigpunt.

Uit figuur 4.3 lezen we af $\theta \approx 1{,}3\,\text{s}$, $\Delta t \approx 4{,}4 - 1{,}3 = 3{,}1\,\text{s}$ en $\Delta T = 70 - 30 = 40\,^\circ\text{C}$. We kunnen nu a berekenen: $a = \frac{\Delta T}{\Delta t} = 40/3{,}1 = 12{,}9\,^\circ\text{C/s}$.

De waarde van a^* kunnen we nu eenvoudig berekenen:

$$a^* = a/\Delta p = 12{,}9/36 = 0{,}358\,^\circ\text{C}/(\text{s}\cdot\%) \tag{4.2}$$

Met behulp van tabel 4.1 kunnen we nu de PID-parameters eenvoudig berekenen:

$$K_c = \frac{1{,}2}{\theta \cdot a^*} = \frac{1{,}2}{1{,}3 \cdot 0{,}358} = 2{,}58\,\%/^\circ\text{C} \tag{4.3}$$

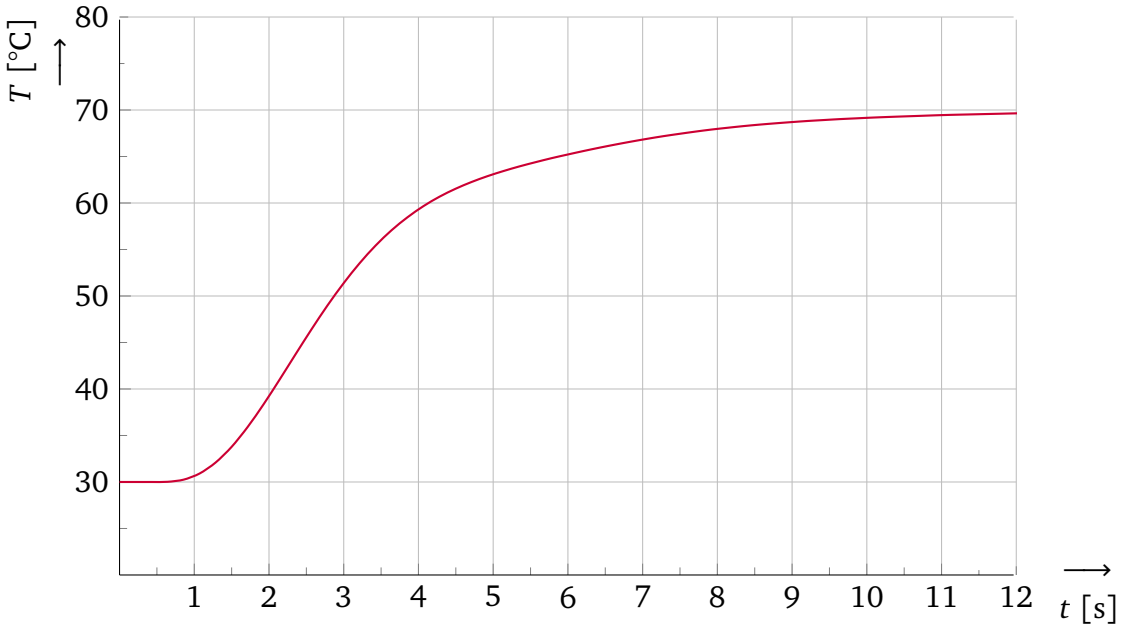
$$T_i = 2{,}0 \cdot \theta = 2{,}0 \cdot 1{,}3 = 2{,}6\,\text{s} \tag{4.4}$$

$$T_d = 0{,}5 \cdot \theta = 0{,}5 \cdot 1{,}3 = 0{,}65\,\text{s} \tag{4.5}$$

4.4 Andere open-loop-methoden

Bij de in paragraaf 4.2 besproken methode schatten we twee procesparameters waarmee we de PID-parameters bepalen. Betere instellingen voor de PID-parameters kunnen gevonden worden door drie in plaats van twee procesparameters te bepalen. We schatten dan de versterking, de dode tijd en de tijdconstante van het systeem. De eersten die dit bedachten waren Cohen en Coon [5] in 1953.

We bepalen weer de stapresponsie van de open-loop, net zoals in de vorige paragraaf. We laten de responsie nu echter wel lopen totdat de eindwaarde bereikt is, zie figuur 4.4. Dit kan overigens erg lang duren, zeker indien het te regelen proces beschikt over een grote tijdconstante.



Figuur 4.4: De stapresponsie van het te regelen proces.

Voor deze meting is een nauwkeurig eindwaarde van groot belang (in figuur 4.4 is dat 70 °C. Hoe nauwkeuriger deze gemeten wordt, des te beter. Op basis hiervan wordt een ΔT bepaald ($\Delta T = T_{eind} - T_{start}$).

De tijdconstante van een systeem (τ_{sys}) is bij een zuiver eerste-orde-systeem het tijdstip waarop de stapresponsie gestegen is met 63,2 % van ΔT[12]. Bij een systeem met dode tijd θ geldt dat het tijdstip waarop de stapresponsie gestegen is met 63,2 % van ΔT gelijk is aan $\theta + \tau_{sys}$. Daarnaast wordt bij een zuiver eerste-orde-systeem een derde van de tijdconstante bereikt op het moment dat de stapresponsie gestegen is met 28,3 % van ΔT. Bij een systeem met dode tijd θ geldt dat het tijdstip waarop de stapresponsie gestegen is met 28,3 % van ΔT gelijk is aan $\theta + \frac{1}{3}\tau_{sys}$.

We bepalen het tijdstip waarbij de temperatuur gestegen is tot $T_1 = T_{begin} + 0.283 \times \Delta T$ af en noemen dat t_1. We bepalen ook het tijdstip waarbij de temperatuur gestegen is tot $T_2 = T_{begin} + 0.632 \times \Delta T$ en noemen dat t_2. Als beide bovengenoemde tijdstippen (t_1 en t_2) bepaald zijn, dan hebben we een stelsel van twee vergelijkingen met twee onbekenden:

$$\begin{cases} t_1 = \theta + \frac{1}{3}\tau_{sys} \\ t_2 = \theta + \tau_{sys} \end{cases} \tag{4.6}$$

[12] De stapresponsie van een zuiver eerste-orde-systeem in procenten van de verandering van het uitgangssignaal is gelijk aan $(1 - e^{\frac{-t}{\tau_{sys}}}) \cdot 100\,\%$. Als we $t = \tau_{sys}$ invullen geeft dit ongeveer 63,2 % en als we $t = \frac{1}{3}\tau_{sys}$ invullen geeft dit ongeveer 28,3 %.

We kunnen de waarden van τ_{sys} en θ bepalen door dit stelsel op te lossen:

$$\begin{cases} \tau_{sys} = \frac{3}{2}(t_2 - t_1) \\ \theta = t_2 - \tau_{sys} \end{cases} \tag{4.7}$$

Om de versterking van het te regelen proces (K_{sys}) te bepalen, wordt de output van het proces gedeeld door de inputwaarde (= de output van de PID-regelaar).

Met behulp van K_{sys}, τ_{sys} en θ kunnen we een aantal nieuwe methoden bekijken. Want naast Ziegler en Nichols zijn vele anderen bezig geweest met het vinden van optimale PID-regelparameters. Een korte selectie van de meest gebruikte methoden staat in tabel 4.2. De IEEE database bevat maar liefst 1446 papers met zowel 'PID' als 'tuning' in de titel[13].

Tabel 4.2: PID-parameters volgens verschillende open-loop-methoden.

Methode / Parameters	K_c [%/?]	T_i [s]	T_d [s]
Ziegler-Nichols open-loop (PID)	$K_c = \frac{1{,}2 \cdot \tau_{sys}}{K_{sys} \cdot \theta}$	$T_i = 2{,}0 \cdot \theta$	$T_d = 0{,}5 \cdot \theta$
Ziegler-Nichols open-loop (PI)	$K_c = \frac{0{,}9 \cdot \tau_{sys}}{K_{sys} \cdot \theta}$	$T_i = 3{,}33 \cdot \theta$	–
Cohen-Coon (PID)	$K_c = \frac{\tau_{sys}}{K_{sys} \cdot \theta} \cdot \left(\frac{\theta}{4 \cdot \tau_{sys}} + \frac{4}{3}\right)$	$T_i = \theta \cdot \frac{32 \cdot \tau_{sys} + 6 \cdot \theta}{13 \cdot \tau_{sys} + 8 \cdot \theta}$	$T_d = \theta \cdot \frac{4 \cdot \tau_{sys}}{2 \cdot \theta + 11 \cdot \tau_{sys}}$
Cohen-Coon (PI)	$K_c = \frac{\tau_{sys}}{K_{sys} \cdot \theta} \cdot \left(\frac{\theta}{12 \cdot \tau_{sys}} + \frac{9}{10}\right)$	$T_i = \theta \cdot \frac{30 \cdot \tau_{sys} + 3 \cdot \theta}{9 \cdot \tau_{sys} + 20 \cdot \theta}$	–
minimum-ITAE (PID)	$K_c = \frac{1{,}357}{K_{sys}} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{-0{,}947}$	$T_i = \frac{\tau_{sys}}{0{,}842} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{0{,}738}$	$T_d = 0{,}381 \cdot \tau_{sys} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{0{,}995}$
minimum-ITAE (PI)	$K_c = \frac{0{,}859}{K_{sys}} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{-0{,}977}$	$T_i = \frac{\tau_{sys}}{0{,}674} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{0{,}680}$	–
AMIGO (PID)	$K_c = \frac{1}{K_{sys}} \cdot \left(0{,}2 + 0{,}45 \cdot \frac{\tau_{sys}}{\theta}\right)$	$T_i = \frac{0{,}4 \cdot \theta + 0{,}8 \cdot \tau_{sys}}{\theta + 0{,}1 \cdot \tau_{sys}} \cdot \theta$	$T_d = \frac{0{,}5 \cdot \theta \cdot \tau_{sys}}{0{,}3 \cdot \theta + \tau_{sys}}$
AMIGO (PI)	$K_c = \frac{0{,}35 \cdot \tau_{sys}}{K_{sys} \cdot \theta}$	$T_i = 13{,}4 \cdot \theta$	–

Hierbij wordt Cohen-Coon vaak voor trage processen (lange dode tijd) gebruikt. De minimum-ITAE [8] zorgt ervoor dat de integraal van de absolute error vermenigvuldigd met de tijd geminimaliseerd wordt (ITAE staat voor: Integral of Time-weighted Absolute Error). Deze methode levert een nauwkeurige regelaar op. Beide methoden zijn echter, net zoals de Ziegler-Nichols-methode, gevoelig voor veranderingen in de procesparameters [7]. De AMIGO (Approximate M-constraint Integral Gain Optimization) -methode [2] is de meest recente van de hier gegeven methoden en is ontwikkeld met het doel om robuste instellingen te vinden waarbij de PID-regelaar minder gevoelig is voor variaties in de procesparameters.

[13] https://ieeexplore.ieee.org/search/searchresult.jsp?action=search&newsearch=true&searchField=Search_All&matchBoolean=true&queryText=((%22Document%20Title%22:PID)%20AND%20%22Document%20Title%22:tuning).

Het zal duidelijk zijn dat de open-loop-methoden alleen gebruikt kunnen worden als het proces zelfregelend is. Als het systeem van zichzelf instabiel is moet een andere methode worden gebruikt om de PID-parameters in te stellen, bijvoorbeeld een van de closed-loop-methoden die in paragraaf 4.6 worden behandeld.

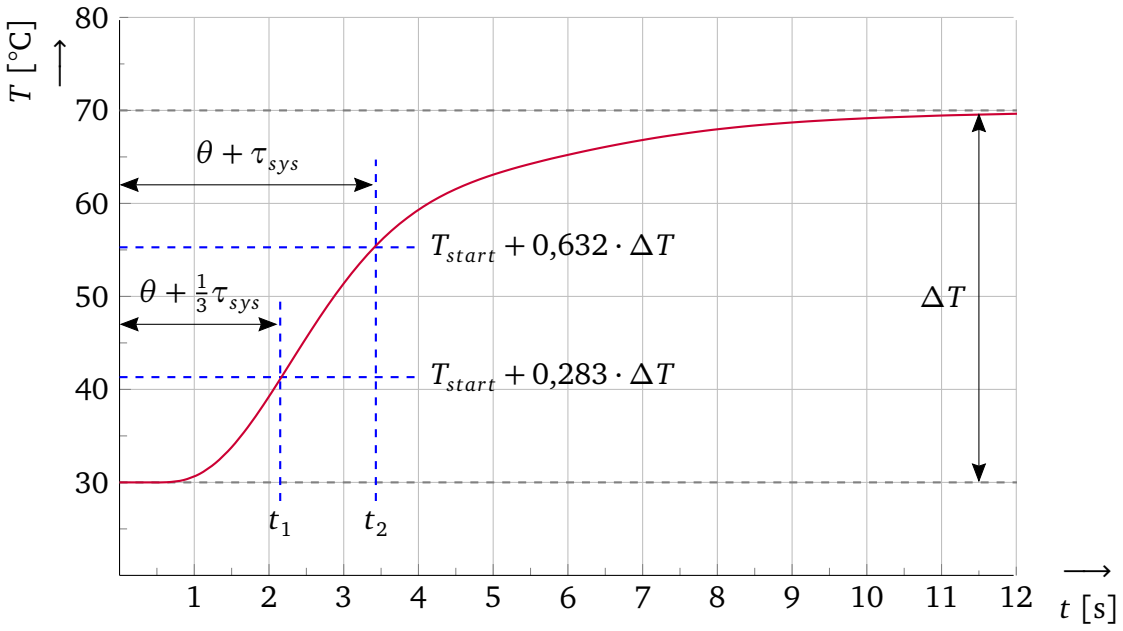
Opdracht 7: Bepalen van de parameters van de PID-regelaar volgens de minimum-ITAE-methode

a) Bepaal zelf de waarde van K_{sys}, τ_{sys} en θ uit de in figuur 4.4 gegeven stapresponsie, bij een stap van 27 % naar 63 % op de ingang, met behulp van de in deze paragraaf beschreven methode.

b) Bepaal met behulp van deze waarden de parameters voor de PID-regelaar van dit proces volgens de minimum-ITAE-methode.

4.5 Antwoorden van opdracht 7

Uit figuur 4.5 lezen we af: $t_1 \approx 2{,}2\,\text{s}$ en $t_2 \approx 3{,}4\,\text{s}$. Met behulp van vergelijking (4.7) bepalen we τ_{sys} en θ: $\tau_{sys} = \frac{3}{2} \cdot (3{,}4 - 2{,}2) = 1{,}8\,\text{s}$ en $\theta = 3{,}4 - 1{,}8 = 1{,}6\,\text{s}$. De waarde van K_{sys} kan als volgt berekend worden $K_{sys} = \frac{\Delta T}{\Delta p} = \frac{70-30}{63-27} = 1{,}016\,°\text{C}/\%$.



Figuur 4.5: De stapresponsie van het te regelen proces. Op tijdstip t_1 heeft de output de waarde $T_{start}+0{,}283\cdot\Delta T$ bereikt en op tijdstip t_2 heeft de output de waarde $T_{start}+0{,}632\cdot\Delta T$ bereikt.

Met behulp van de bepaalde waarden voor K_{sys}, τ_{sys} en θ kunnen we met behulp van tabel 4.2 de parameterwaarden van de PID-regelaar vinden volgens de minimum-ITAE-methode:

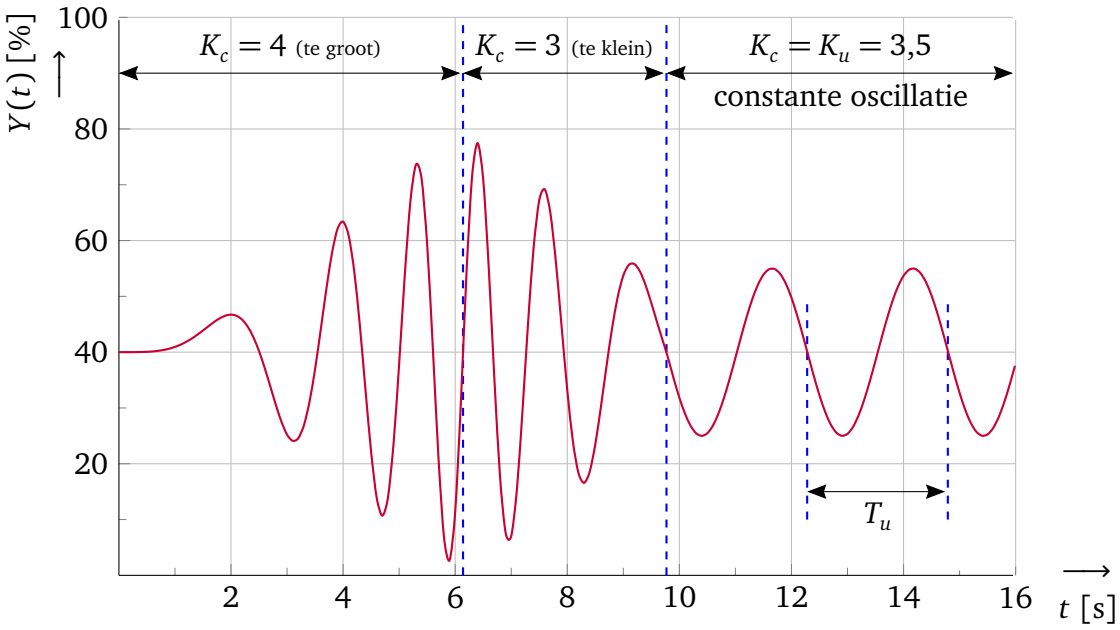
$$K_c = \frac{1{,}357}{K_{sys}} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{-0{,}947} = \frac{1{,}357}{1{,}016} \cdot \left(\frac{1{,}6}{1{,}8}\right)^{-0{,}947} = 1{,}50\ \%/°\text{C} \tag{4.8}$$

$$T_i = \frac{\tau_{sys}}{0{,}842} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{0{,}738} = \frac{1{,}8}{0{,}842} \cdot \left(\frac{1{,}6}{1{,}8}\right)^{0{,}738} = 1{,}96\ \text{s} \tag{4.9}$$

$$T_d = 0{,}381 \cdot \tau_{sys} \cdot \left(\frac{\theta}{\tau_{sys}}\right)^{0{,}995} = 0{,}381 \cdot 1{,}8 \cdot \left(\frac{1{,}6}{1{,}8}\right)^{0{,}995} = 0{,}61\ \text{s} \tag{4.10}$$

4.6 Closed loop response

Een andere methode om de PID-parameters te bepalen is de zogenoemde 'closed-loop'-responsie. Bij deze methode is het de bedoeling om de 'lus te sluiten' en de regelaar te laten regelen. De T_i-waarde wordt zo groot mogelijk gemaakt en de T_d-waarde zo klein mogelijk, waardoor ze effectief uitgeschakeld zijn en er uitsluitend met de K_c-waarde gewerkt wordt. De K_c-waarde van de regelaar wordt vervolgens zodanig opgevoerd, dat het systeem begint te oscilleren. Deze methode wordt daarom ook wel de 'oscillatie methode' genoemd.



Figuur 4.6: De responsie van het te regelen proces in closed loop bij verschillende waarden van K_c.

De K_c-waarde waarbij het totale systeem begint te oscilleren, wordt K_u genoemd. De periodetijd van de oscillaties, T_u, is ook van belang bij deze methode. Uit figuur 4.6 kun je aflezen dat, in dit specifieke geval, de K_c-waarde waarbij het totale systeem begint te oscilleren, 3,5 is en dat de periodetijd T_u ongeveer 2,5 seconden is.

Een direct nadeel van deze methode is dat het in sommige installaties ongewenst is om het systeem te laten oscilleren, het kan zelfs gevaarlijk zijn. Wees je dus altijd bewust van het systeem waarmee je aan het werk bent!

Ook hier zijn weer een aantal vuistregels te geven, hoe een regelaar ingesteld kan worden. Een korte selectie van de meest gebruikte methoden staat in tabel 4.3 [2, 10, 11, 12].

Tabel 4.3: PID-parameters volgens verschillende closed-loop-methoden.

Methode / Parameters	K_c [%/?]	T_i [s]	T_d [s]
Ziegler-Nichols closed loop (PID)	$K_c = \frac{K_u}{1{,}7}$	$T_i = \frac{T_u}{2}$	$T_d = \frac{T_u}{8}$
Tyreus-Luyben	$K_c = \frac{K_u}{2{,}2}$	$T_i = 2{,}2 \cdot T_u$	$T_d = \frac{T_u}{6{,}3}$
Takahashi digitale regelaar	$K_c = 0{,}6 \cdot K_u \cdot \left(1 - \frac{T_s}{T_u}\right)$	$T_i = \frac{T_u - T_s}{2}$	$T_d = \frac{{T_u}^2}{8 \cdot T_u - T_s}$
AMIGO PID	$K_c = \left(0{,}3 - 0{,}1 \cdot \kappa^4\right) \cdot K_u$	$T_i = \frac{0{,}6}{1 + 2 \cdot \kappa} \cdot T_u$	$T_d = \frac{0{,}15 \cdot (1 - \kappa)}{1 - 0{,}95 \cdot \kappa} \cdot T_u$
AMIGO PI	$K_c = 0{,}16 \cdot K_u$	$T_i = \frac{1}{1 + 4{,}5 \cdot \kappa} \cdot T_u$	–

Voor de closed-loop AMIGO-methoden geldt $\kappa = \left(\frac{1}{K_{sys} \cdot K_u}\right)$.

De methode van Takahashi geeft een optimale instelling voor de in hoofdstuk 3 afgeleide type C PID-regelaar, zie vergelijking (3.9). Een voordeel van deze methode is dat ook direct de sampletijd T_s hierin meegenomen is.

4.7 Industriële regelaars

In [7, blz. 32-19] worden 17 industriële controllers opgesomd die automatisch de PID-parameters kunnen bepalen. Tien daarvan gebruiken een open-loop-methode. Een andere methode die veel gebruikt wordt is de zogenoemde 'relay autotuner'. Bij deze methode wordt de PID-regelaar vervangen door een relay met hysteresis. Voor veel processen zorgt dit ervoor dat het outputsignaal gaat oscilleren met periodetijd T_u. De waarde van K_u kan door een amplitudemeting worden bepaald. Als de waarde van T_u en K_u bekend zijn, dan kunnen de PID-parameters met behulp van de formules behorende bij één van de closed-loop-methoden (zie tabel 4.3) bepaald worden.

Bibliografie

[1] Kiam Heong Ang, G. Chong en Yun Li. "PID control system analysis, design, and technology". In: *Control Systems Technology, IEEE Transactions on* 13.4 (jul 2005), p. 559–576. ISSN: 1063-6536. URL: https://core.ac.uk/download/pdf/1394222.pdf.

[2] K.J. Åström en T. Hägglund. "Revisiting the Ziegler–Nichols step response method for PID control". In: *Journal of Process Control* 14.6 (2004), p. 635–650. ISSN: 0959-1524. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.476.1815&rep=rep1&type=pdf.

[3] Karl Johan Åström. *PID Control*. 2002. URL: https://www.cds.caltech.edu/~murray/courses/cds101/fa02/caltech/astrom-ch6.pdf.

[4] Rockwell Automation. *Perform Common Process Loop Control Algorithms*. 2016. URL: https://literature.rockwellautomation.com/idc/groups/literature/documents/wp/logix-wp008_-en-p.pdf.

[5] G.H. Cohen en G.A. Coon. "Theoretical consideration of retarded control". In: *Trans. Asme* 75.1 (1953), p. 827–834.

[6] Finn Haugen. "Ziegler-Nichols' Open-Loop Method". In: (2010). URL: http://home.hit.no/~hansha/documents/control/theory/zn_open_loop_method.pdf.

[7] William S. Levine, red. *The Control Handbook*. Second. CRC Press, 2011.

[8] A.M. Lopez e.a. "Tuning Controllers with Error-Integral Criteria". In: *Instrumentation Technology* (1976), p. 57–62.

[9] Bernd Souvignier. *Les 9 Numerieke integratie*. 2004. URL: https://www.math.ru.nl/~souvi/wiskunde2_04/les9.pdf.

[10] R.H.C. Takahashi, P.L.D. Peres en P.A.V. Ferreira. "Multiobjective H_2/H_∞ guaranteed cost PID design". In: *Control Systems, IEEE* 17.5 (1997), p. 37–47. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=621468.

[11] Bjorn D. Tyreus en William L. Luyben. "Tuning PI controllers for integrator/dead time processes". In: *Industrial & Engineering Chemistry Research* 31.11 (1992), p. 2625–2628.

[12] J. G. Ziegler en N. B. Nichols. "Optimum Settings for Automatic Controllers". In: *Transactions of ASME* 64 (1942), p. 759–768.