Tampereen yliopisto

Jussi Leimu

KASKADISÄÄDÖN ANTI-WINDUP-TOTEUTUS MATLABIN S-FUNKTIOLLA

Tekniikan ja luonnontieteiden tiedekunta
Kandidaatintyö
Syyskuu 2020

TIIVISTELMÄ

Jussi Leimu: "Kaskadisäädön anti-windup-toteutus MATLABin S-funktiolla"
Kandidaatintyö
Tampereen yliopisto
Teknisten tieteiden TkK-tutkinto-ohjelma, Automaatiotekniikka
Syyskuu 2020

Kaskadisäätö on kehittynyt säätöpiirirakenne, jota käytetään yleisesti esimerkiksi prosessiteollisuudessa. Rakenne koostuu kahdesta tai useammasta itsenäisestä säätimestä ja sen hyöty perustuu kasvaneeseen järjestelmästä saatavan mittaustiedon määrään. Anti-windup viittaa toimilaitteiden rajallisista kyvyistä johtuvien ongelmien huomioimiseen. Windup-ilmiön mahdollisuus on huomioitava kaikissa säätöpiirin säätimissä siten, että yhdessä säätimessä tapahtuva erikoistilanne ei vaikuta negatiivisesti muiden piirissä olevien säätimien toimintaan.

Tässä työssä toteutetaan kaskadisäätöpiirin simulaattori käyttäen MATLABin S-funktiota PI-tyyppisten säätimien luomiseen. Toteutetussa kaskadipiirissä on anti-windup-toiminto, joka on luotu käyttäen tracking back-calculation -rakennetta. Tässä työssä huomioitavat erikoistilanteet ovat kaskadipiirin sekundäärisäätimen paikallisen asetusarvon kytkentä ja sekundäärisäätimen manuaalinen ohjaus.

S-funktioilla toteutetun simulaattorin oikeellinen toiminta varmistetaan tekemällä samanlainen kaskadipiirin simulaattori alkeislohkoja käyttäen. Toteutusten eroja vertaillaan ajamalla simulaattorien läpi testiskenaario, joka simuloi toimilaitteen saturoitumista, sekundäärisäätimen paikallisen asetusarvon, sekundäärisäätimen manuaaliohjauksen käyttöä sekä askelmaisen häiriön vaikutusta säätöpiirin ulostuloon. Häiriöitä kohdistetaan sekä simulaattorissa ohjattavan järjestelmän primääri- että sekundääriprosessiin. Simulaattorissa ohjattava järjestelmä ei perustu mihinkään reaaliseen järjestelmään ja sen ominaisuudet on valittu mielivaltaisesti.

Työssä toteutetun simulaattorin pohjalta on mahdollista luoda simulaattori mitä tahansa kaksitasoista kaskadisäätöpiiriä varten.

Avainsanat: kaskadisäätö, anti-windup, MATLAB S-funktio.

ALKUSANAT

Kiitos Veli-Pekka Pyrhöselle kandidaatintyöni ohjaamisesta. Kiitokset myös läheisille tuesta kirjoitusprosessin aikana.

Tampereella, 25.9.2020

Jussi Leimu

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

Lyhenteet

D-osa	Derivointiosa
I-osa	Integrointiosa
MATLAB	Matrix laboratory, The Mathworksin ohjelmointiympäristö
MIMO	Multiple-Input-Multiple-Output; järjestelmä, jossa on useita sisääntuloja ja ulostuloja
PD	Proportional-Derivative
PI	Proportional-Integral
PID	Proportional-Integral-Derivative
P-osa	Proportionaaliosa
Simulink	MATLABin simulointiympäristö

Merkinnät

A	Tilamatriisi
Aw_p	Anti-windup-toiminnon ulostulo
B	Ohjausmatriisi
C	Ulostulomatriisi
D	Myötäkytkentämatriisi
e	Erosuure
e_p	Primäärisäätimen erosuure
e_s	Sekundäärisäätimen erosuure
$ISmax$	Input-skaalauksen yläraja
$ISmin$	Input-skaalauksen alaraja
K_a	Anti-windup-vahvistus
K_{aw}	Sekundäärisäätimen anti-windup-vahvistus
K_i	Integrointiosan vahvistus
K_p	Proportionaaliosan vahvistus
Kp_p	Primäärisäätimen proportionaaliosan vahvistus
Kp_s	Sekundäärisäätimen proportionaaliosan vahvistus
Kt	Tracking-takaisinkytkennän vahvistus
r	Säätöpiirin asetusarvo
tau_p	Primääriprosessin aikavakio
tau_s	Sekundääriprosessin aikavakio
T_d	Derivointiaika
T_i	Integrointiaika
Ti_p	Primäärisäätimen integrointiaika
Ti_s	Sekundäärisäätimen integrointiaika
TR	Tracking-signaali
T_t	Anti-windup-rakenteen aikavakio
u	Ohjaussignaali
u_max	Saturoitunut ohjaussignaali
u_min	Saturoitunut ohjaussignaali
u_p	Primäärisäätimen käsittelemätön ohjaussignaali
$u\_p(sat)$	Primäärisäätimen saturoitunut ohjaussignaali
$u(sat)$	Saturoitunut ohjaussignaali
u_sek	Sekundäärisäätimen käsittelemätön ohjaussignaali
x	Tilamuuttuja
$\dot{x}$	Tilamuuttujan x derivaatta
y	Säätöpiirin ulostulo

ys	Sekundääriprosessin ulostulo

1. JOHDANTO

Tässä työssä käsitellään windup-ilmiön korjaamista kaskadisäätöpiirin yhteydessä. Windup-ilmiö johtuu järjestelmien fyysisistä rajoituksista, joten sen tapahtumiseen tulee varautua kaikissa todellisissa järjestelmissä. Työn viitekehyksenä on Proportional-Integral-Derivative-säätimin (PID-säädin) toteutettu säätöpiiri. PID-säädin koostuu proportionaali-, integrointi- ja derivointiosista. Säädin käyttää saamansa asetusarvon ja prosessin ulostulon välistä erosuuretta, joka syötetään erikseen säätimen eri osiin. Osien ulostulojen summa on säätimen prosessiin syöttämä ohjaussignaali. [1, pp. 64-72] Tässä työssä säätöpiirin molemmissa säätimissä derivointiosa on kytketty pois päältä eikä se esiinny lohkokaavioissa.

Kaskadisäätö on kehittynyt, mutta suhteellisen yksinkertainen säätöpiirirakenne. Tyypillinen käyttöympäristö kaskadisäädölle on prosessiteollisuus. Windup-ilmiö voi esiintyä missä tahansa säätöpiirissä ja se on huomioitava aina säätöpiiriä suunniteltaessa. Simulaattorien hyödyntäminen suunnitteluprosessissa on tyypillistä, sillä varsinaisia prosesseja, joihin säätöpiiriä ollaan suunnittelemassa, ei välttämättä ole vielä olemassa, se on jatkuvassa käytössä tai virhetilanteiden riskit ovat liian suuret säätöpiirin testaamiseen reaaliprosessissa. Esimerkkejä tällaisista skenaarioista ovat järjestyksessään, uuden prosessilaitoksen rakentaminen, vanhan automaatiojärjestelmän uudistaminen ja ydinvoimalaitosympäristöissä kaikki automaatiojärjestelmien muutostilanteet.

Simulaattori toteutetaan MATLABin Simulink-ympäristössä käyttäen lopullisen kaskadisäätöpiirin luomiseen Level-2 MATLAB® S-function -ohjelmointirajapintaa. Simulaattorissa ohjattava prosessi koostuu kahdesta mielivaltaisesti valitusta sarjaan kytketystä osaprosessista. Ohjattavan prosessin ei ole tarkoitus mallintaa mitään tiettyä järjestelmää, vaan sen ominaisuudet on valittu puhtaasti siten, että simulaattorin vasteesta on helposti todennettavissa halutut toiminnot ja kaskadipiirin ominaispiirteet.

Tämän työn avulla voidaan toteuttaa kahden säätimen kaskadisäätöpiirin simulaattori. Tässä työssä toteutetussa säätöpiirissä on anti-windup-toiminto ja säätöpiiri on toteutettu yhtenä S-funktio-lohkona. Työssä toteutettuun simulaattoriin on mahdollista muokata lukijan tarpeisiin sopivia ominaisuuksia, ja simulaattorin pohjalta voidaan tuottaa simulaattori kolmi- tai useampitasoiselle kaskadirakenteelle.

2. TAUSTA

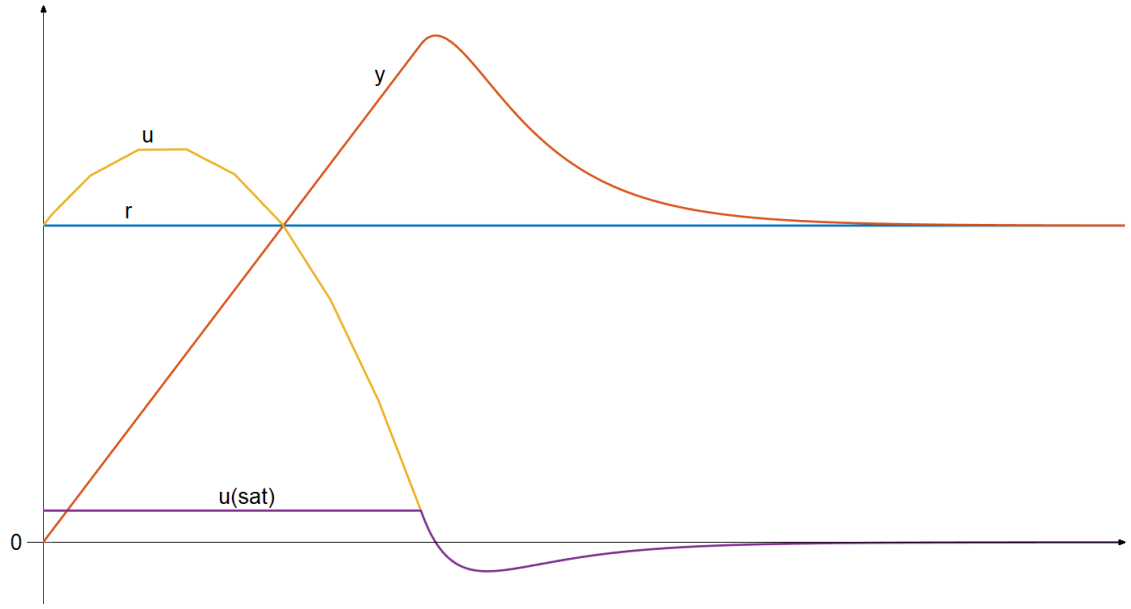
Tässä luvussa tarkastellaan työn keskeisimpiä käsitteitä, windup-ilmiötä ja kaskadisäätöä. Windup-ilmiöstä käsitellään sen synty ja tapoja siitä aiheutuvien ongelmien kompensoimiseksi (anti-windup). Kaskadisäädöstä esitellään siihen liittyvää käsitteistöä ja rakennetta sekä millaisissa järjestelmissä sen käyttöä on syytä harkita, mitä edellytyksiä käytölle on ja millaista hyötyä voidaan odottaa. Anti-windup-toiminnon toteutus kaskadirakenteen yhteydessä eroaa yhden säätimen toteutuksesta, josta johtuvat muutokset, kuten tracking-toiminto, käydään läpi. Lisäksi luvun lopussa esitellään MATLABin S-funktio.

2.1 Windup-ilmiö

PID-säätimen integrointiosaa käytetään vakiohäiriöiden eliminointiin [2, p. 270]. Integrointiosa kasvattaa ulostuloaan niin kauan kuin säätimelle saapuva erosuure poikkeaa nollasta. Windup-ilmiö tapahtuu, kun säädettävä toimilaite on toiminta-alueensa rajalla, esimerkiksi kun venttiili on täysin avattu tai täysin suljettu. Säädin pyrkii ohjaamaan toimilaitetta sen käyttöalueen ulkopuolelle pienentääkseen erosuuretta. Tällöin toimilaite on saturoitunut. Kun prosessin ulostulo ei vastaa ohjauksen muutokseen, säädin kasvattaa ohjaustaan. Toimilaite reagoi ja järjestelmän ulostulo hidastuu vasta, kun ilmiön aiheuttama ylimäärä integrointiosan ulostulossa on säätimen muiden osien ulostuloja pienempi. Tämä johtaa asetusarvon ylitykseen ja vasteen oskilloimiseen. [3]

Ilmiötä on havainnollistettu kuvassa 1, jossa r on asetusarvo, y on systeemin ulostulo, u on säätimen tuottama ohjaussignaali ja $u(sat)$ on toimilaitteen tuottama ohjaus. Kuvan esimerkkitilanteessa systeemin ulostulon kasvu hidastuu vasta kun ohjaussignaali saapuu toimilaitteen toiminta-alueelle.

Ilmiön aiheuttamat ongelmat on perinteisesti ohitettu ylimitoittamalla järjestelmän toimilaitteet, mikä johtaa järjestelmän kokonaismassan kasvuun. Esimerkiksi polttoainetehokkuutta vaativissa sovelluksissa, kuten ilma- ja avaruusaluksissa, tämä ei ole suotavaa. Myös modernien säätimien tapauksissa windup-ilmiöllä tarkoitetaan toimilaitteen saturoitumisesta aiheutuvia ongelmia, vaikka niissä ei edellä kuvailtua integrointiosaa olekaan. [3]



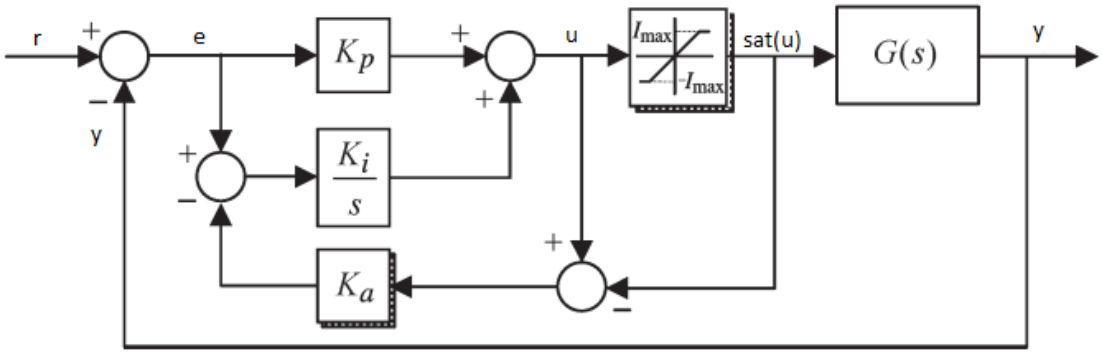
Kuva 1. *Windup-ilmiö havainnollistettuna.*

Windup-ilmiön estämiseksi säätöpiiri on rakennettava niin, että säädin reagoi toimilaitteen saturoitumiseen [3]. Tämä voidaan toteuttaa joko ottamalla toimilaite huomioon säädintä viritettäessä tai lisäämällä säätimeen anti-windup-toiminto, jolloin säädin voidaan virittää perinteisesti. Anti-windup-toiminnon käyttö on yleisempää juuri virittämisen suhteellisen helppouden takia. [4]

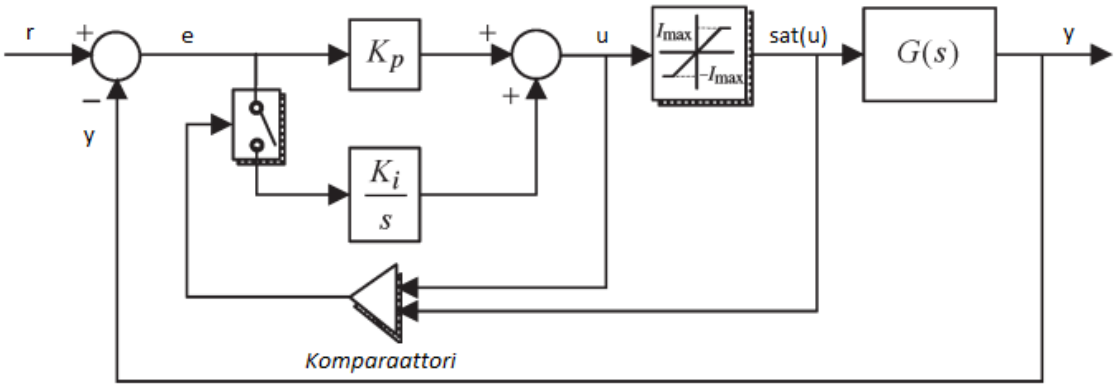
Perinteiset windup-ilmiön vastaiset anti-windup-rakenteet ovat ehdollinen integrointi (eng. conditional integration) ja tracking back-calculation. Viimeksi mainittua käytettäessä säätimessä on ylimääräinen takaisinkytkentä, joka seuraa toimilaitteen saaman ohjauksen ja toimilaitteen tuottaman ulostulon erotusta. Tracking back-calculation -rakenne on esitetty lohkokaaviona kuvassa 2 osana PI-säädintä. Lohkokaaviossa esiintyvä K_a on anti-windup-toiminnon vahvistus, $G(s)$ on säädettävä prosessi, K_p on P-osan vahvistus ja K_i on integrointiosan vahvistus. [5] Eräissä lähteissä [1, pp. 79-81] merkitään integrointiosan ja anti-windup-toiminnon vahvistuksia murtolukuina $\frac{K_p}{T_i}$ ja $\frac{1}{T_t}$, joiden nimittäjät ovat integrointiaika T_i ja anti-windup-aika T_t. Näissä lähteissä kuvatuissa säätimissä on myös D-osa. Åström ja Hägglund [1, p. 80] esittävät, että anti-windup-toiminnon aikavakion T_t tulee olla suurempi kuin D-osan derivointiaika T_d ja pienempi kuin T_i.

Ehdollista integrointia hyödyntävä anti-windup-toiminto pitää integrointiosan ulostulon vakiona, kun tietyt ehdot täyttyvät [1]. Hyväksi ehdoksi on havaittu toimilaitteen saturoituminen ja tarkentavana ehtona voidaan vaatia, että erosuureen ja ohjaussignaalin pitää olla saman merkkiset [4]. Ehdollinen integrointi osana PI-säädintä on esitetty kuvassa 3, jossa esiintyvä komparaattori vertailee säätimen ulostuloa sekä saturoitunutta ohjaussignaalia, ja kun säätimen ulostulo ja saturoitunut ohjaussignaali

ovat samat, pitää komparaattori integrointiosan päälle kytkettynä [5]. Kuvissa 2 ja 3 esitetyt anti-windup-rakenteet muistuttavat suuresti toisiaan, mutta ehdollisen integroinnin lohkokaavioesitys riippuu integroinnille asetetuista ehdoista, joten toisiaan vastaavia ne eivät ole. Back-calculation-rakenteen etuna voidaan pitää sen dynaamista luonnetta verrattuna ehdolliseen integrointiin, joka katkaisee integrointiosan toiminnan täysin [1].



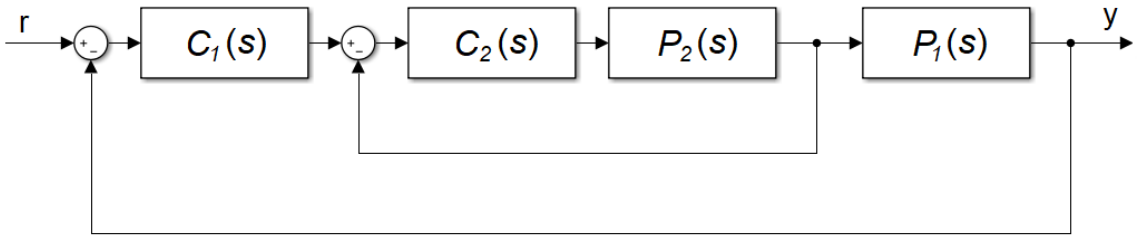
Kuva 2. *PI-säädin, jossa back-calculation rakenne, mukailtu lähteestä [5].*



Kuva 3. *PI-säädin, jossa ehdollinen integrointi, mukailtu lähteestä [5].*

2.2 Kaskadisäätöpiiri

Kaskadisäätö on yleisesti käytetty, kehittynyt säätöpiirirakenne, jossa on kaksi tai useampia säätimiä ohjaamassa järjestelmän toimintaa. Kaskadirakenteen toiminta perustuu ylimääräisiin mittauksiin ja sen myötä lisääntyneeseen prosesseista saatavaan informaatioon. Tällöin voidaan reagoida sekä järjestelmän hitaaseen että nopeaan dynamiikkaan samanaikaisesti. Kahden säätimen kaskadipiirissä toinen säädin ohjaa koko prosessin ulostuloa. Tätä kutsutaan primääri- tai pääsäätimeksi. Toinen säädin, jota kutsutaan sekundääri- tai orjasäätimeksi, ohjaa jotakin prosessin osa-aluetta. Primäärisäätimen ulostulo toimii sekundäärisäätimen asetusarvona. [2, p. 476]

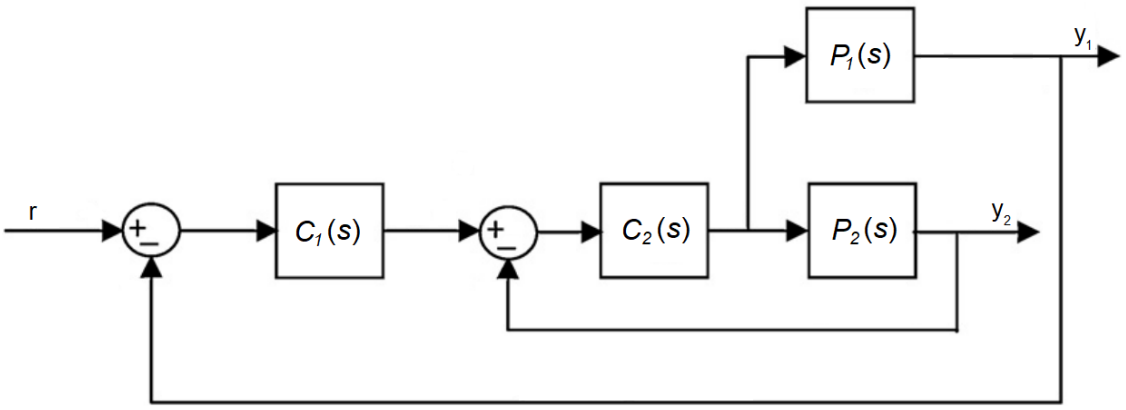


Kuva 4. *Lohkokaavioesitys kaskadisäätöpiiristä sarjassa oleville prosesseille tai osaprosesseille.*

Kaskadirakenteella saavutetaan merkittäviä etuja prosesseissa, joissa viiveet ovat pitkiä ja/tai häiriöt suuria. Tällaisissa prosesseissa säädettävänä suureena on usein paine, virtaus tai lämpötila. [6] Kaskadirakennetta ei voida käyttää, jos järjestelmässä ei ole mitattavissa kahta tai useampaa prosessisuuretta [1, p. 373]. Teollisuudessa kolmen tai neljänkään mittauksen kaskadirakenteet eivät ole epätavallisia. Teoriassa takaisinkytkentöjen lukumäärällä ei ole ylärajaa. [2, p. 492]

Vanhimmat aihetta käsittelevät julkaisut [7, 8] ovat 1950- ja 1960-luvulla kirjoitettuja, mutta kaskadipiiri saattaa olla keksintönä vieläkin vanhempi [2]. Franks ja Worley [7] osoittivat jo vuonna 1956, että kaskadisäätöpiirin ohjattavuus on samaan järjestelmään kytketyn yhden säätimen piirin ohjattavuutta parempi. He määrittivät ohjattavuuden paremmuuden vertailemalla kumulatiivista virhettä säädettävän suureen ja asetusarvon välillä järjestelmän transienttivasteen aikana eri säätöpiirirakenteita käytettäessä. Järjestelmä, jonka transienttivasteen aikana esiintyi vähiten kumulatiivista virhettä, määritettiin parhaiten ohjattavaksi. He tutkivat säätöpiirityyppien ohjattavuutta seitsemässä eri järjestelmässä. Kaikissa järjestelmissä oli määritettävissä kaksi osaprosessia. Ohjattavuutta tutkittiin tilanteissa, joissa järjestelmään vaikutti joko asetusarvohäiriö, primääriprosessin häiriö tai sekundääriprosessin häiriö. Kaskadisäätöpiirin ohjattavuuden todettiin olevan kahdessa ensin mainitussa häiriötilanteessa 0–5 ja 0–8 kertaa parempi kuin yksittäisellä säätimellä toteutetun säätöpiirin ohjattavuus. Kun häiriö kohdistui piirin sekundääriprosessiin, kaskadirakenne on kuitenkin ylivoimainen, koska ohjattavuuden havaittiin parantuneen jopa 2000-kertaiseksi verrattuna yhden säätimen toteutukseen.

Kaskadipiirin prosessit voivat olla sarjassa, kuten Franks ja Worley [7] esittivät, tai rinnakkain. Ensimmäiset julkaisut kaskadipiirin käytöstä rinnakkaisille prosesseille ovat 1970-luvulta. Luyben [9] osoitti tuolloin, että kaskadipiirin käyttö parantaa systeemin häiriönsietoa sekä sarjassa että rinnakkain olevien prosessien tapauksissa. Kaskadirakenne sarjassa oleville prosesseille on esitetty kuvassa 4, jossa $C_1(s)$ on primäärisäädin, $C_2(s)$ on sekundäärisäädin, $P_2(s)$ on sekundääriprosessi ja $P_1(s)$ on primääriprosessi. Lohkokaavioesitys rinnakkaisten prosessien tapaukselle on kuvassa 5. Siinä esiintyvät muuttujat y_1 ja y_2 ovat primääri- ja sekundääriprosessien ulostulot.



Kuva 5. *Kaskadisäätöpiiri rinnakkaisille prosesseille tai osaprosesseille. Mukailtu lähteestä [10].*

Kaskadisäätöpiiriä suunniteltaessa ohjattavan prosessin toiminnan ja prosessimallien tunteminen on avainasemassa. Huonosti valittu sekundäärinen mittaus ei paranna prosessin ohjattavuutta. Marlin [2, pp. 479-480] esittää, että sekundäärimittauksen tulee täyttää kolme kriteeriä. Ensiksi mitattavan suureen pitää ilmaista prosessiin vaikuttavan häiriön vaikutusta. Toiseksi primääri- ja sekundäärisuureen välillä tulee olla syy-seuraussuhde. Lisäksi sekundäärisuureen dynamiikka pitää olla primäärisuureen dynamiikkaa nopeampi, jotta sekundäärisäädin ehtii korjaamaan häiriön vaikutuksen ennen kuin häiriö vaikuttaa primäärisuureeseen. Yleisohjeeksi on esitetty, että sekundäärisuureen asettumisajan tulee olla korkeintaan kolmasosa primäärisuureen asettumisajasta, mutta luku vaihtelee lähteen mukaan [1, p. 375, 2, p. 480].

Sekä primääri- että sekundäärisäätimessä voidaan käyttää mitä tahansa PID-konfiguraatiota [2, p. 487]. Usein sekundäärisäätimeksi riittää P- tai PD-säädin, mutta integrointiosasta voi olla hyötyä, jos prosessiin vaikuttaa matalataajuisia häiriöitä. Primäärisäädin on tyypillisesti PI- tai PID-säädin. [1, pp. 375-376] Jos prosesseja ajetaan myös erikoistilanteissa, joissa primäärisäädin ei ole toiminnassa, sekundäärisäädin on rakenteeltaan sama kuin primäärisäädin. [2, p. 487]

Kaskadipiirissä windup-ilmiön huomiointi on äärimmäisen tärkeää, koska piirissä on kaksi itsenäistä säädintä. Riskinä on windup primäärisäätimessä, kun sekundäärisäätimessä tapahtuu jotakin poikkeuksellista. Åströmin ja Hägglundin [1, pp. 376-377] mukaan seuraavat poikkeustilanteet on tuotava primäärisäätimen tietoon: toimilaitteen saturoituminen, sekundäärisäätimen paikallisen asetusarvon päälle kytkentä ja sekundäärisäätimen manuaalisen ohjauksen päälle kytkentä. Tämä voidaan toteuttaa esimerkiksi back-calculation-rakenteen tracking-toiminnolla, jolloin primäärisäätimelle tuodaan seurantasignaali, joka korvaa säätöpiirin ulkoisen asetusarvon poikkeustilan vallitessa. Seu-

rantasignaalina toimii poikkeustilanteen mukaan joko saturoitunut ohjaussignaali, paikallinen asetusarvo tai manuaalisesti asetettu ohjaussignaali. Muulloin seurantasignaali on nolla. [1, p. 377]

Yleinen käyttöympäristö kaskadisäädölle on prosessiteollisuus, jossa sitä käytetään esimerkiksi höyrykattiloiden tulistimissa ja pneumaattisten venttiilien asemoinnissa [1, p. 377, 2, pp. 494-495]. Venttiilin asemoinnissa kaskadipiiri rakentuu siten, että primäärisäädin ohjaa varsinaista prosessisuuretta ja sekundäärisäädin ohjaa sitä venttiiliä, jonka läpi virtaavalla aineella vaikutetaan prosessisuureeseen. Tämä parantaa säätöpiirin tarkkuutta, koska pneumaattisissa venttiileissä voi esiintyä usean prosenttiyksikön suuruisia kuolleita alueita, joiden takia suhteellisesti pienien venttiilin asennonmuutosten tekeminen on haasteellista. Kaskadipiirin käyttömahdollisuudet eivät kuitenkaan rajoitu pelkästään prosessiteollisuuteen, vaan sitä voidaan käyttää esimerkiksi kilpamoottoripyörän sähköisessä jarrujärjestelmässä [11]. Kyseessä olevan artikkelin [11] tapauksessa primäärisenä mittaussuureena on pääjarrusylinterissä vallitseva paine ja sekundäärisenä mittaussuureena pääjarrusylinterissä olevan männän asento.

2.3 MATLABin S-funktio

MATLAB S-funktio on MATLAB-ohjelmiston työkalu, jonka avulla voidaan luoda käyttäjän tarpeisiin räätälöityjä lohkoja ohjelmiston Simulink-simulointiympäristöön. Funktion luoma systeemilohko voi mallintaa aikajatkuvia, -diskreettejä ja näiden yhdistelmiä eli hybridisysteemejä. [12] Tässä työssä käytetty S-funktion implementointitapa on Level-2 MATLAB S-function -ohjelmointirajapinta. Funktio kirjoitetaan käyttäen MATLABin omaa ohjelmointikieltä. Simulink-ympäristössä S-funktio-lohkoon määritetään funktiossa käytettävät parametrit. Level-2 MATLAB S-function -ohjelmointirajapinta mahdollistaa muun muassa multiple-input and multiple-output-systeemin (MIMO-systeemin) mallintamisen ja se pystyy käsittelemään sekä reaalisia että kompleksisia signaaleja [13]. S-funktio voidaan luoda myös käyttäen C, C++ ja Fortran -ohjelmointikieliä [12].

Yksinkertaisimmillaan level-2 MATLAB S-funktion kooditiedostossa on vain kaksi callback metodia: Setup ja Outputs. Ensin mainitussa määritetään muun muassa lohkon sisään- ja ulostulojen ominaisuudet sekä lohkon viritysparametrien lukumäärä. Outputs-metodissa tapahtuu lohkon tuottamien ulostulojen laskenta. Mahdollisia S-funktioon sisällytettäviä metodeja on olemassa 24. [14] Esimerkiksi aikajatkuvan tilamuuttujan alustamiseen tarvitaan InitializeConditions-metodi [15].

3. SIMULAATTORI JA SÄÄTÖPIIRIN S-FUNKTIO-TOTEUTUS

Tässä luvussa käsitellään kaksitasoista kaskadisäätöpiiriä mallintavan simulaattorin toteutus. Simulointiin käytetään MATLABin Simulink-lisäosaa, joka on mallintamiseen ja simuloimiseen tarkoitettu ohjelmointiympäristö. Simulinkin graafinen käyttöliittymä perustuu lohkokaavioesityksen tekemiseen. Kaikki tässä työssä käytetyt lohkot kuuluvat ympäristön peruskirjastoon. Säätöpiiriin toteutettavat ominaisuudet ovat anti-windup ja input-skaalaus. Anti-windup toteutetaan back-calculation-rakenteella, jossa on tracking-toiminto.

Ensin toteutetaan simulaattori käyttäen säätimien luomiseen alkeislohkoja, jonka jälkeen ajetaan simulaattorin läpi eräs testiskenaario. Seuraavaksi säätimet luodaan käyttäen S-funktio-lohkoja. Lopuksi molemmat säätimet luodaan käyttäen yhtä S-funktio-lohkoa. Näiden toteutusten läpi ajetaan sama testiskenaario kuin alkeislohkototeutuksenkin läpi. Toteutusten toimintaa tarkastelemalla todennetaan, että S-funktio-toteutukset toimivat halutulla tavalla.

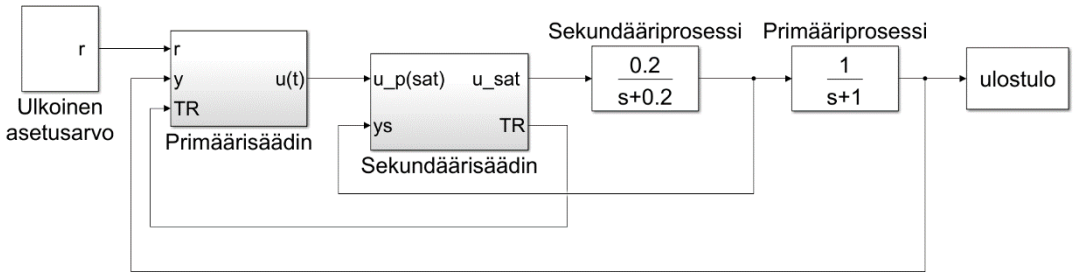
3.1 Simulaattori ja testiskenaario

Testaussimulaattorissa on kahdesta osaprosessista, primääri- ja sekundääriprosessista, koostuva yksinkertainen järjestelmä. Osaprosessien mallintamiseen käytetään State Space -lohkoja. Tilamallin yleisessä muodossa

$$\begin{cases} \dot{x} = A \cdot x + B \cdot u \\ y = C \cdot x + D \cdot u \end{cases} \quad (1)$$

esiintyvät tilamuuttuja x ja sen derivaatta $\dot{x}$. Tilamallissa esiintyvät matriisit A, B, C ja D ovat järjestyksessään tilamatriisi, ohjausmatriisi, ulostulomatriisi ja myötäkytkentämatriisi. Testisimulaattorin osaprosesseissa kaikki edellä mainitut matriisit ovat skalaareja. Primääriprosessin tilamallissa A on $\frac{-1}{tau\_p}$ ja B on $\frac{1}{tau\_p}$. Sekundääriprosessin tilamallissa A on $\frac{-1}{tau\_s}$ ja B on $\frac{1}{tau\_s}$. Vakiot tau_p ja tau_s on valittu siten, että sekundääriprosessi on dynamiikaltaan primääriprosessia nopeampi, ja täten kaskadipiirin implementointi prosessikokonaisuuteen on aliluvussa 2.2 kuvailtujen ehtojen mukaisesti järkevää. Primääriprosessin tilamalli on kaavan (2) mukainen. Sekundääriprosessin tilamalli on kaavan (3) mukainen.

$$\begin{cases} \dot{x} = -0{,}2 \cdot x + 0{,}2 \cdot u \\ y = x \end{cases} \quad (2)$$



Kuva 6. *Simulaattorin lohkokaavioesitys*

$$\begin{cases} \dot{x} = -x + u \\ y = x \end{cases} \quad (3)$$

Kaikki simulaattorissa käytetyt muuttujat selitteineen ja arvoineen on koottu taulukkoon 1. Molempien osaprosessien tilamalleissa C on 1 ja D on 0. Kaikkien simulaattorissa käytettyjen integrator- ja state space-lohkojen alkuarvo (eng. initial condition) on 0. Testaussimulaattori on esitetty yksinkertaistettuna kuvassa 6, jossa osaprosessien lohkoihin on merkitty niiden siirtofunktiot. Sekundääriprosessin siirtofunktio on $\frac{0,2}{s+0,2}$ ja primääriprosessin siirtofunktio on $\frac{1}{s+1}$.

Taulukko 1 *Simulaattorissa käytetyt vakiot.*

Symboli	Selite	Arvo
tau_p	Primääriprosessin aikavakio	5,0
tau_s	Sekundääriprosessin aikavakio	1,0
u_min	Ohjaussignaalin skaalattu minimiarvo	0,0
u_max	Ohjaussignaalin skaalattu maksimiarvo	1,0
$ISmin$	Input-skaalauksen alaraja	0,0
$ISmax$	Input-skaalauksen yläraja	1,0
Kp_p	Primäärisäätimen vahvistus	0,5
Ti_p	Primäärisäätimen integrointiaika	7,0
Kt	Primäärisäätimen tracking-vahvistus	1,0
Kp_s	Sekundäärisäätimen vahvistus	2,0
Ti_s	Sekundäärisäätimen integrointiaika	6,0
Kaw	Sekundäärisäätimen anti-windup-vahvistus	0,1

Molemmat simulaattorissa käytettävät säätimet ovat konfiguraatioltaan PI-säätimiä. Primäärisäätimen sisääntuloina ovat ulkoisen asetusarvon ja primääriprosessin ulostulon erosuure $e(t)$ ja tracking-signaali TR. Primäärisäätimen ulostulo on primääriohjaus u_pri. Sekundäärisäätimen sisääntuloina ovat primäärisäätimen ulostulo u_pri ja sekundääriprosessin ulostulo ys. Ulostuloina sekundäärisäätimellä on sekundääriohjaus u_sat

ja tracking-signaali TR, joka lähtee takaisinkytkentänä aiemmin mainitusti primäärisäätimen sisääntuloksi. Sekundäärisäätimeen on mahdollista kytkeä paikallinen asetusarvo, ja sekundäärisäätimen ohjaus voidaan asettaa manuaalisesti. Molemmat aktivoivat tracking-toiminnon. Säätimissä on input-skaalaus, joka tarkoittaa sitä, että sisääntulojen arvot skaalataan halutulle välille. Tässä työssä arvot skaalataan välille 0–1.

Simulaattorissa ajettava testiskenaario koostuu viidestä osiosta, jotka testaavat säätöpiirin anti-windup-toteutuksen toimivuutta, tracking-toiminnon aktivoitumista sekä säädön häiriönkompensointikykyä. Testiskenaarion pituus on 900 aikayksikköä. Ensimmäisessä osiossa järjestelmän ulkoinen asetusarvo nostetaan askelmaisesti arvosta 0 arvoon 1,2, jolloin toimilaite saturoituu, koska ohjaussignaalin maksimiarvoksi on määritetty 1,0. Ajanhetkellä 100 asetusarvo lasketaan askelmaisesti arvoon 0,6. Jos anti-windup on toteutettu oikein, säätöpiirin ulostulo lähtee laskuun välittömästi asetusarvon muuttuessa. Ulkoinen asetusarvo pysyy arvossa 0,6 testiskenaarion loppuun asti.

Toisessa ja kolmannessa osiossa testataan tracking-toiminnon aktivoituminen ja oikeellinen toiminta. Toisessa osiossa aktivoidaan sekundäärisäätimen paikallinen asetusarvo ajanhetkellä 200 ja deaktivoidaan ajanhetkellä 250. Kolmannessa osiossa sekundäärisäädintä ohjataan manuaalisesti aikavälillä 300–400. Molemmissa osioissa muutos on askelmainen, mutta ei aiheuta toimilaitteiden saturoitumista. Molemmissa osioissa ulostulo reagoi välittömästi muutokseen sekä tracking-toiminnon aktivoituessa että deaktivoituessa.

Neljäs ja viides osio testaavat säätöpiirin vastetta, kun prosessiin vaikuttaa askelmainen häiriö. Neljännessä osiossa häiriö vaikuttaa sekundääriprosessiin ja viidennessä osiossa primääriprosessiin. Kummassakin osiossa vaikuttavat häiriöt ovat vahvuudeltaan samat. Tarkoituksena on demonstroida kaskadipiirille ominaista hyvää sekundääriprosessiin vaikuttavan häiriön kompensointia.

3.2 Kaskadipiirin säädinten S-funktio-toteutus

S-funktion kooditiedoston tuottaminen aloitettiin MATLAB:n tarjoamalle tiedostopohjalle ”msfuntmpl_basic.m”, joka kuuluu Simulinkin perusasennukseen ja on löydettävissä MATLAB-komennolla

```
edit([matlabroot,'/toolbox/simulink/blocks/msfuntmpl_basic.m']).
```

Primäärisäätimessä on sisääntuloina ulkoisen asetusarvon ja primääriprosessista mitatun ulostulon erosuure sekä tracking-signaali. Säätimen ainut ulostulo on sen tuottama ohjaussignaali. Sisääntulot ja ulostulo ovat ominaisuuksiltaan toisiaan vastaavia, kaikki

ovat arvoiltaan reaalisia skalaareja. Datatyyppiksi valitaan double, joka on MATLABin oletusdatatyyppi numeeriselle datalle [16]. Näytteenottomoodiksi asetetaan ”Sample”.

Sisään- ja ulostuloporttien ominaisuudet alustetaan setup-metodissa, jossa määritetään myös kuusi S-funktion parametria: säätimen integraattorin alkuarvo, säätimen vahvistus, säätimen integrointiaika, input-skaalauksen minimi- ja maksimiarvo sekä tracking-vahvistus. Integraattori on aikajatkuva systeemi, jolle alustetaan aikajatkuva tilamuuttuja setup-metodissa. Tämän tilamuuttujan alkuarvoksi asetetaan S-funktioon parametrina syötetty integraattorin alkuarvo käyttäen InitializeConditions-metodia. Lisäksi primäärisäätimen S-funktioon tarvitaan Derivatives- ja Outputs-metodit.

Outputs-metodissa tapahtuu säätimen tuottaman ohjauksen laskenta ja toimilaiterajojen huomiointi. Yksinkertaisuudessaan säätimen saturoitumattoman ohjauksen laskenta tapahtuu kaavalla

$$u\_p = Kp\_p \cdot i + Kp\_p \cdot e\_p, \tag{4}$$

jossa i on säätimen integraattorin tila ja e_p on ulkoisen asetusarvon r ja primäärimittauksen y välinen input-skaalattu erosuure. Tästä saadaan primäärisäätimen saturoitunut ohjaus $u\_p(sat)$ rajoittamalla u_p välille $[u\_min, u\_max]$ käyttäen ehtolauserakennetta. Erosuureen laskenta ja sen input-skaalaus tapahtuu alla olevan kaavan (5) mukaisesti.

$$e\_p = \frac{r-y}{(ISmax-ISmin)} \tag{5}$$

Derivates-metodissa lasketaan aikajatkuvan tilan derivaatta eli tässä tapauksessa säätimen integraattorin arvon muutosnopeus. Integraattorin tilaan vaikuttavat integrointiajan ja erosuureen lisäksi anti-windup-toiminto. Integraattorin tilan muutosnopeus lasketaan kaavan

$$\frac{1}{Ti\_p} \cdot e\_p + Aw\_p \tag{6}$$

mukaisesti. Kaavassa (6) esiintyvä muuttuja Aw_p on anti-windup-toiminnon vaikutus. Normaali tilanteessa kyseinen vaikutus on nolla, sillä anti-windup-toiminto aktivoituu vasta kun säätimen tuottama ohjaus on saturoitunut. Derivaatan laskemiseksi metodissa pitää siis laskea säätimen saturoitunut ulostulo ja laskea sen perusteella anti-windup-toiminnon vaikutus. Anti-windup-toimintoon liittyvä laskenta on esitetty ohjelmassa 1, jossa ehtolausekkeilla huomioidaan mahdollinen tracking-signaali ja toimilaitteen saturoituminen.

```
2     if TR ~= 0
        Aw_p = Kt*(TR-u_p);
4     else
        if PI_out_p > 1
6          Aw_p = Kt*(1-u_p);
        elseif PI_out_p < 0
8          Aw_p = Kt*(0-u_p);
        else
10         Aw_p = 0;
        end
      end
```

Ohjelma 1. *Primäärisäätimen anti-windup-toiminnon laskenta.*

Sekundäärisäädin on primäärisäätimeen verrattuna monimutkaisempi ja sen setup-metodi on esitetty typistettynä ohjelmassa 2. Ohjelmasta 2 on toiston välttämiseksi esitetty vain yhden sisääntuloportin ominaisuuksien alustaminen riveillä 11–16 ja yhden ulostuloportin ominaisuuksien alustaminen riveillä 18–21. Vastaavat koodirivit pitää olla jokaiselle sisään- ja ulostuloportille. Sekundäärisäätimeen tehdään sisääntuloportit primäärisäätimen tuottamalle ohjaukselle, paikalliselle asetusarvolle, manuaaliohjaukselle ja prosessin mittaukselle. Ulostuloina säädin tuottaa prosessiin syötettävän ohjaussignaalin ja primäärisäätimelle takaisinkytkettävän tracking-signaalin, jota varten S-funktioon alustetaan yksi aikadiskreettitila. Parametreina S-funktioon syötetään integraattorin alkuarvo, säätimen vahvistus, säätimen integrointiaika, input-skaalauksen minimi- ja maksimiarvo sekä anti-windup-vahvistus.

```
function setup(block)

  % Alustetaan input- ja outputporttien määrä
  block.NumInputPorts = 4;
  block.NumOutputPorts = 2;

  % Alustetaan porttien ominaisuudet dynaamisesti periytyviksi
  block.SetPreCompInpPortInfoToDynamic;
  block.SetPreCompOutPortInfoToDynamic;

  % Määritetään tietyt porttien ominaisuudet vakioiksi
  block.InputPort(1).Dimensions = 1;
  block.InputPort(1).DatatypeID = 0; % double
  block.InputPort(1).Complexity = 'Real';
  block.InputPort(1).DirectFeedthrough = false;
  block.InputPort(1).SamplingMode = 'Sample';

  block.OutputPort(1).Dimensions = 1;
  block.OutputPort(1).DatatypeID = 0; % double
  block.OutputPort(1).Complexity = 'Real';
  block.OutputPort(1).SamplingMode = 'Sample';

  % Alustetaan S-funktion parametrien lukumäärä, aikajatkuvien
  % tilojen lukumäärä ja näytteenottoväli
  block.NumDialogPrms = 6;
  block.NumContStates = 1;
  block.SampleTimes = [0 0];

  % Alustetaan miten Simulink suhtautuu S-funktiolohkoon
  block.SimStateCompliance = 'DefaultSimState';

  % Rekisteröidään S-funktion metodit
  block.RegBlockMethod('PostPropagationSetup', @DoPostPropSetup);
  block.RegBlockMethod('InitializeConditions',  ...
     @InitializeConditions);
  block.RegBlockMethod('Start', @Start);
  block.RegBlockMethod('Outputs' @Outputs);
  block.RegBlockMethod('Update', @Update);
  block.RegBlockMethod('Derivatives', @Derivatives);

  endfunction;
```

Ohjelma 2. *Sekundäärisäätimen kooditiedostosta mukailtu setup-metodi.*

Sekundäärisäätimen S-funktion kooditiedostoon tehdään samat metodit kuin primäärisäätimenkin kooditiedostoon. Lisäksi aikadiskreetin tilan hallintaan tarvitaan DoPostPropSetup-, Start- ja Update-metodit. Kaksi ensin mainittua metodia ovat aikadiskreetin tilamuuttujan alustamista varten ja Update-metodissa tapahtuu varsinainen tracking-signaalin tuottaminen, johon käytetään kolmiosaista ehtolauserakennetta. Järjestys, jossa ehtoja käydään läpi, perustuu säätöpiirin toimintojen määrittelyihin: paikallinen asetusarvo ohittaa primäärisäätimen syöttämän ohjauksen, ja manuaaliohjaus ohittaa paikallisen asetusarvon. Tämän hierarkian takia viimeisessä ehtolausekkeessa valitaan manuaaliohjauksen ja kaiken muun väliltä. Keskimmäinen ehtolauseke huomioi paikallisen

asetusarvon ja ensimmäinen ehtolauseke huomioi ohjaussignaalin saturoitumisen. Saturaation huomioimisessa on otettu huomioon laskennan tarkkuudesta johtuva värähtely käyttämällä tuhannesosan suuruista toleranssiarvoa. Ehtolauserakenne toteutettuna s-funktion kooditiedostossa on esitetty ohjelmassa 3.

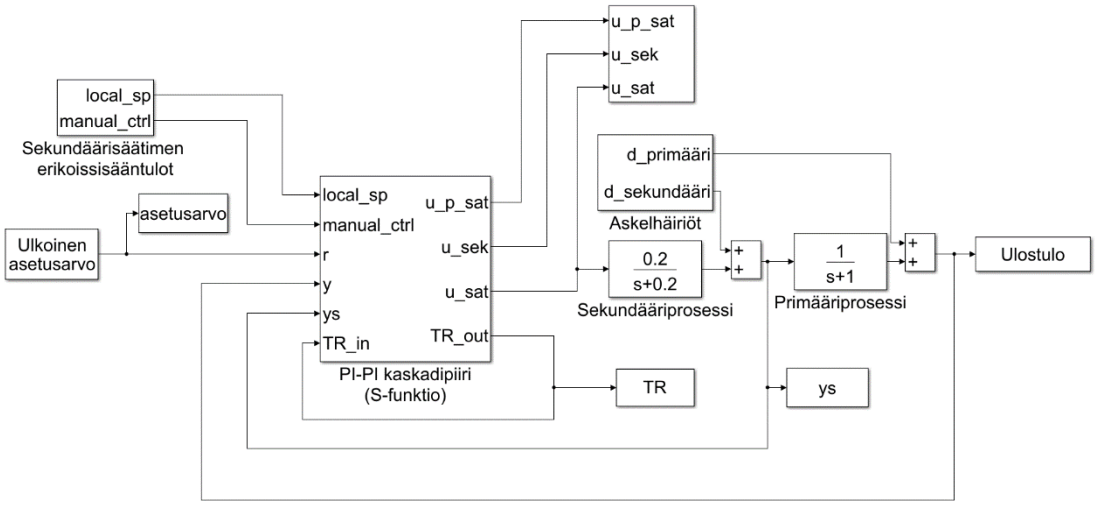
```
% TR-signaalin tuottaminen.
% Saturaation huomiointi tallentamalla mahdollinen saturoitunut
% ohjaussignaali apumuuttujaan log1:
if abs(u_sat-u_sek) > 0.001
     log1 = u_sat;
else
     log1 = 0;
end

% Paikallisen ohjauksen huomiointi tallentamalla apumuuttujaan
% log2 joko paikallinen asetusarvo tai apumuuttuja log1:
if local_sp > 0
    log2 = local_sp;
else
    log2 = log1;
end

% Manuaalisen ohjauksen huomiointi:
if man_ctrl > 0
    TR = u_sat;
else
    TR = log2;
end
```

Ohjelma 3. *Tracking-signaalin määrittäminen.*

Kaskadipiirin toteutus yhdellä S-funktiolla tapahtuu käytännössä liittämällä yllä kuvailtujen primäärisäätimen S-funktion ja sekundäärisäätimen S-funktion kooditiedostot yhteen. Ainoat eroavaisuudet ovat S-funktion alustuksessa. Tässä toteutuksessa S-funktioon alustetaan kaksi aikajatkuvaa tilaa, yksi aikadiskreetti tila, kuusi sisääntuloa, neljä ulostuloa ja kymmenen parametria. Sisääntuloina on sekundäärisäätimen manuaaliohjaus, sekundäärisäätimen paikallinen asetusarvo, säätöpiirin ulkoinen asetusarvo, primääriprosessin ulostulon mittaustulos, sekundääriprosessin ulostulon mittaustulos ja tracking-signaali. S-funktion ulostuloina on saturoituneen ohjaussignaalin lisäksi primäärisäätimen tuottama ohjaussignaali, sekundäärisäätimen saturoitumaton ohjaussignaali ja tuotettu tracking-signaali. Kolme viimeksi mainittua ulostuloa ovat S-funktion oikeellisen toiminnan todentamista ja mahdollisten ongelmien havainnointia ja korjaamista varten. Lisäksi Tracking-signaalin kierrättäminen funktion ulkopuolelta takaisin sisääntuloksi yksinkertaistaa S-funktiota siten, että tracking-signaalia ei tarvitse laskea erikseen sekä Update- että Derivatives-metodissa.

4. TULOKSET

Tässä luvussa esitellään S-funktiolla toteutetun simulaattorin toiminta, kun simulaattorissa suoritetaan luvussa 3 kuvailtu testiskenaario. Tämän jälkeen käsitellään toteutuksessa ilmenneet ongelmat. Lopuksi arvioidaan toteutuksen heikkouksia ja mahdollisia parannuskohteita. Kuvassa 7 on esitetty työssä toteutettu simulaattori täydellisenä.



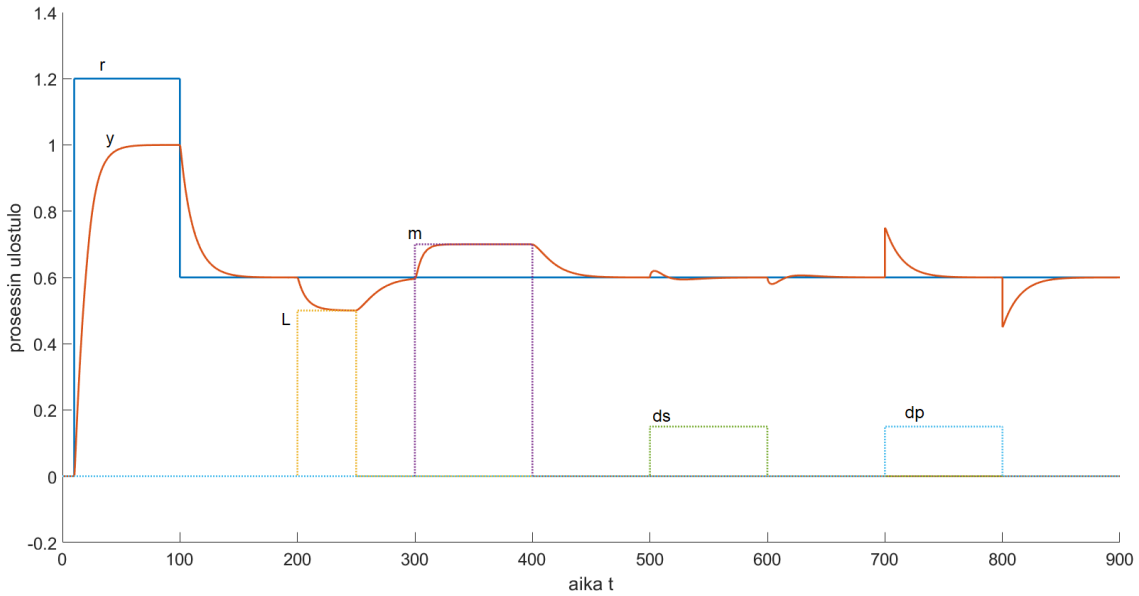
Kuva 7. *Valmis simulaattori.*

4.1 S-funktio-toteutuksen onnistuminen

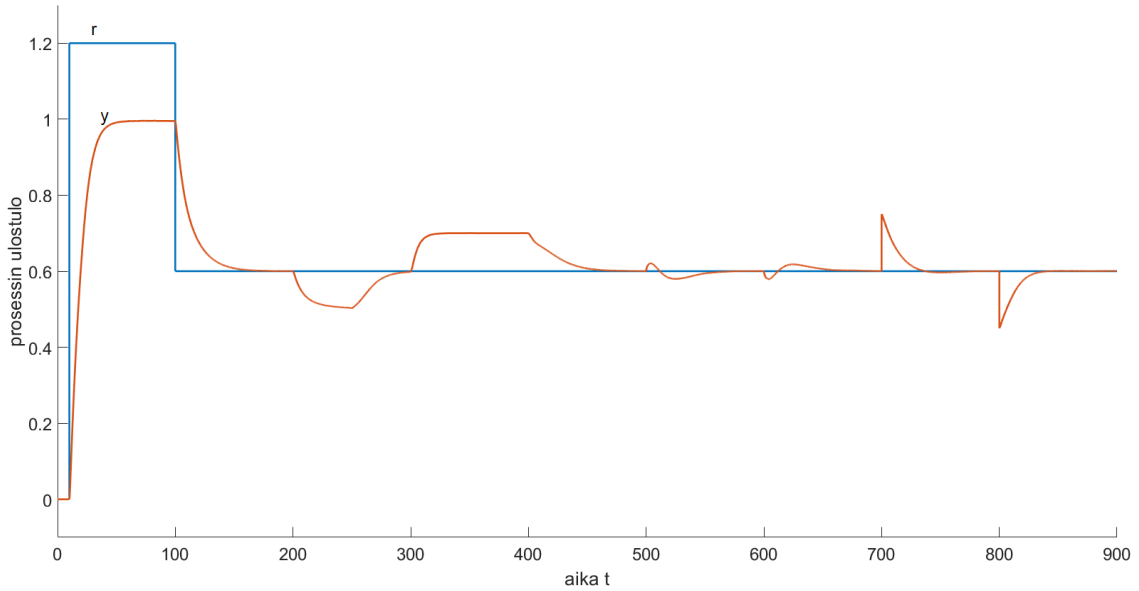
Alkeislohkoilla toteutetun kaskadisäätösimulaattorin vaste luvussa 3 kuvaillun testiskenaarion aikana on esitetty kuvassa 8, jossa L on sekundäärisäätimen paikallinen asetusarvo, m on sekundäärisäätimen manuaaliohjaus, ds on sekundääriprosessin häiriö ja dp on primääriprosessin häiriö. Vasteesta voidaan todeta säätöpiirin toimivan anti-windup-toteutuksen toteutuksen osalta halutusti. Säätöpiirissä ei esiinny viivettä ulostulon ja ohjauksen muutoksien välillä, vaan ulostulo hakeutuu asetusarvoon heti asetusarvomuutoksen jälkeen sekä toimilaitteen saturoituessa että tracking-toiminnon aktivoituessa ja deaktivoituessa. Säätöpiiri kykenee myös eliminoimaan sekä primääri- että sekundääriprosessiin vaikuttavan häiriön aiheuttaman säätövirheen, mutta primääriprosessiin vaikuttava häiriö aiheuttaa huomattavasti suuremman muutoksen säätöpiirin ulostulossa kuin sekundääriprosessiin vaikuttava samansuuruinen häiriö.

Kuvassa 9 on esitetty yhdellä S-funktiolla toteutetun simulaattorin vaste testiskenaarion aikana. S-funktio-toteutuksen vaste ei ole täysin identtinen kuvassa 8 esitetyn alkeislohkototeutuksen vasteen kanssa, vaan S-funktiolla tuotettu vaste on laadultaan heikompi.

S-funktiosimulaattorin ulostulo ei saavuta maksimiarvoa 1 testiskenaarion ensimmäisessä vaiheessa, jossa toimilaite on saturoitunut. S-funktio-toteutuksen ulostulon arvo myös oskilloi enemmän kuin alkeislohkototeutus sekundääriprosessin häiriön vaikutuksen alkaessa ja päättyessä.



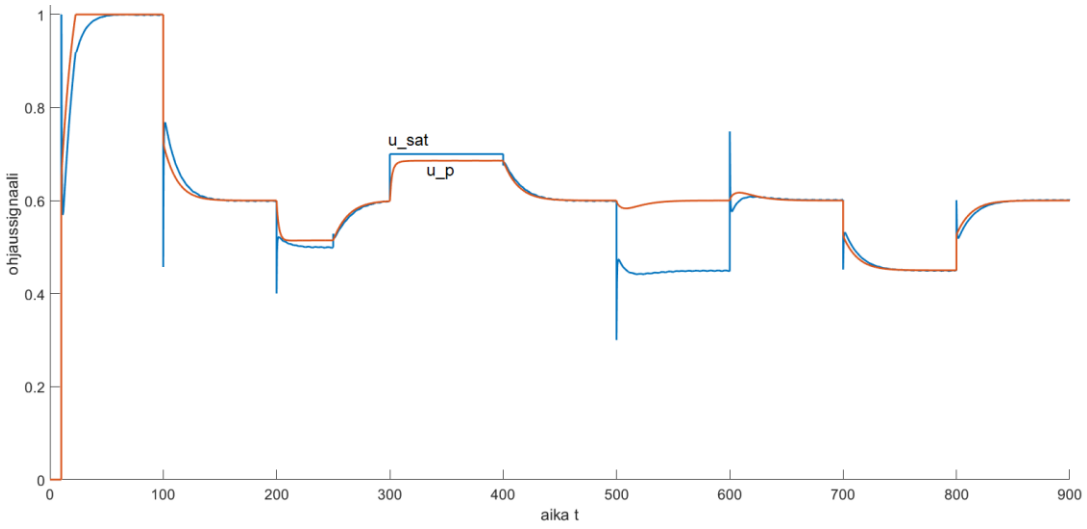
Kuva 8. *Alkeislohkosimulaattorin vaste testiskenaarion aikana.*



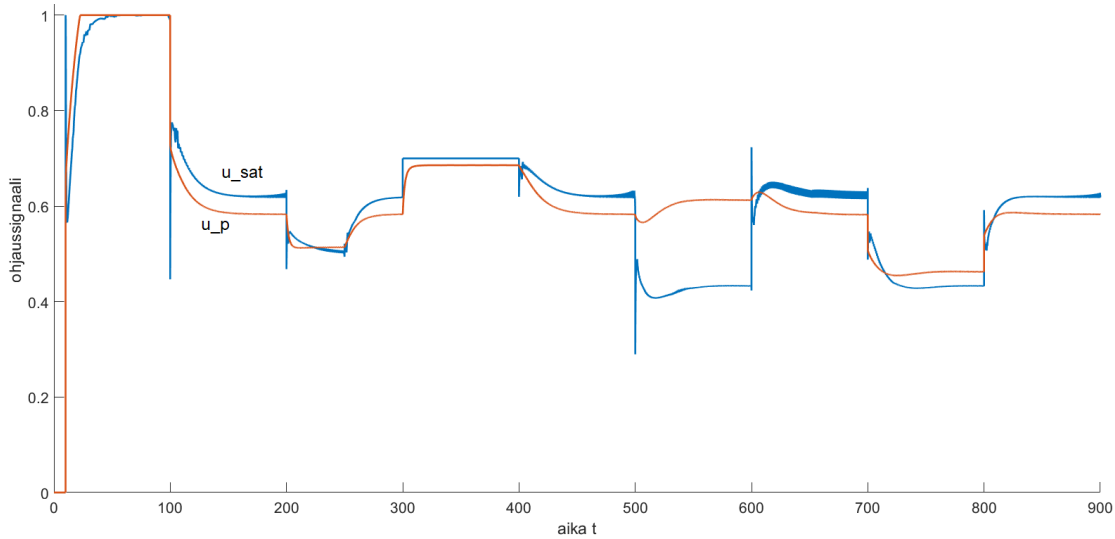
Kuva 9. *S-funktiolla toteutetun simulaattorin vaste testiskenaarion aikana.*

S-funktio-toteutuksen ja alkeislohkototeutuksen ohjaussignaaleja vertailemalla huomataan, että S-funktio-toteutuksessa sekä primääri- että sekundäärisäätimen ohjaussignaalissa esiintyy enemmän kohinaa kuin alkeislohkototeutuksen vastaavissa signaa-

leissa. Suurinta kohina on S-funktio-toteutuksen sekundäärisäätimen ulostulossa. Kohinan voimakkuus on heikointa transienttien aikana ja voimistuu ulostulon asettuessa. Alkeislohkototeutuksen tuottamat primäärisäätimen saturoitunut ohjaussignaali ja sekundäärisäätimen saturoitunut ohjaussignaali ovat esitettynä kuvassa 10. S-funktio-toteutuksen tuottamat vastaavat ohjaussignaalit ovat esitetty kuvassa 11.

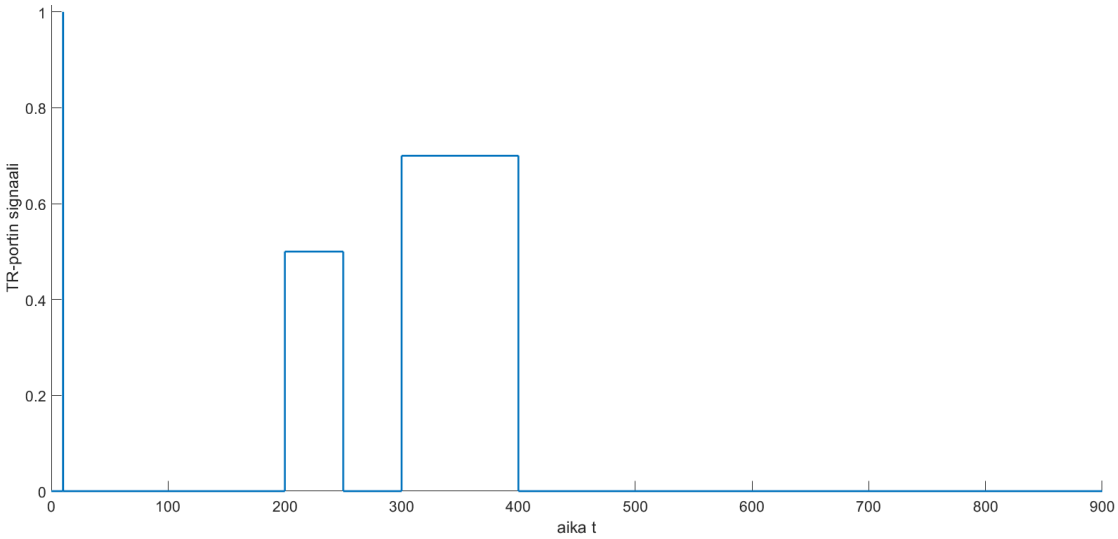


Kuva 10. *Alkeislohkototeutuksen tuottamat ohjaussignaalit.*

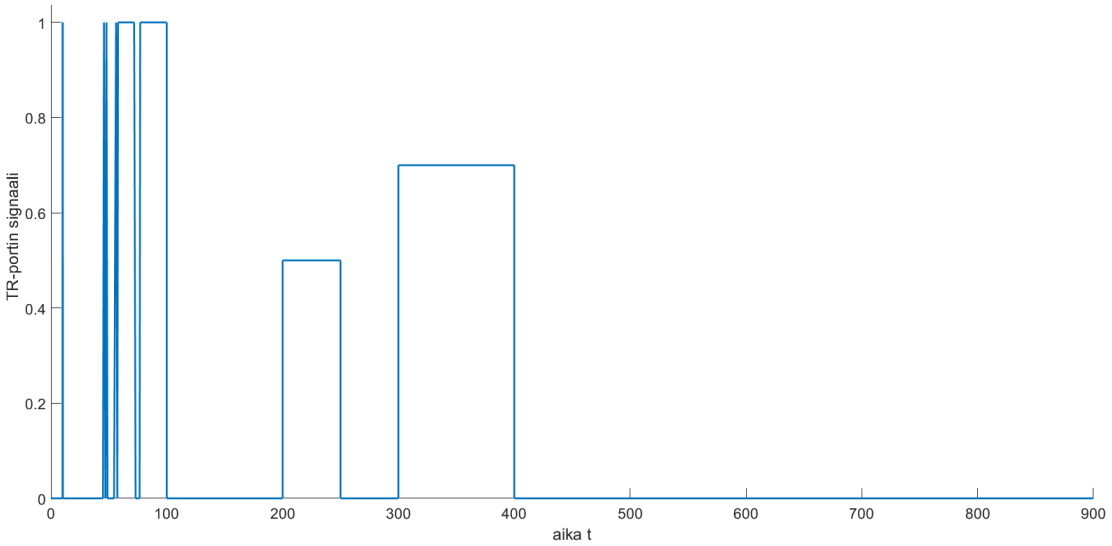


Kuva 11. *S-funktio-toteutuksen tuottamat ohjaussignaalit.*

Toteutusten tracking-signaaleja vertailemalla huomataan, että S-funktio-toteutuksessa tracking-signaali aktivoituu ohjauksen saturoitumisen aikana useita kertoja, kun alkeislohkototeutuksessa se aktivoituu vain kerran transientin aikana. Alkeislohkototeutuksen tracking-signaali on esitetty kuvassa 12 ja S-funktio-toteutuksen tracking-signaali on esitetty kuvassa 13.



Kuva 12. *Alkeislohkototeutuksen tracking-signaali.*



Kuva 13. *S-funktio-toteutuksen tracking-signaali.*

4.2 Toteutuksen arviointi

S-funktio-toteutuksen simulaattori tuotti testiskenaarion aikana MATLABin työtilaan (eng. workspace) 2254 datapistettä jokaiselle työtilaan tuodulle muuttujalle. Alkeislohkoilla toteutettu simulaattori tuotti saman testiskenaarion aikana 784 datapistettä. Jokainen datapiste on keskenään yhtä tarkka, koska käytetty datatyyppi on molemmissa simulaattoreissa sama, mutta lähes kolminkertainen näytteenotto tiheys on mahdollinen syy S-funktio-toteutuksen ohjaussignaaleissa havaitulle kohinalle. Todellista järjestelmää kuvaavaa mallia simuloitaessa kohinan alkuperä on syytä tutkia tarkemmin, sillä tarkempi laskenta saattaa paljastaa säätöpiirin olevan epästabiili.

S-funktion kooditiedostossa toistuvat samat koodirivit eri metodeissa, sillä samoja arvoja, kuten erosuuretta, tarvitaan monessa eri metodissa ja ne pitää laskea jokaisessa metodissa erikseen. Tämä yhdessä pienemmän näytteenottovälin kanssa aiheuttaa sen, että S-funktio-toteutuksen simulointi kestää huomattavasti alkeislohkototeutuksen simulointia pidempään. Testiskenaarion simulointi alkeislohkoilla toteutetussa simulaattorissa kesti allekirjoittaneen käyttämällä laitteistolla noin puoli sekuntia, kun S-funktio toteutusten simulointi kesti 12 sekuntia. Käytännössä kumpikin toteutus on nopea simuloida, mutta jos simuloitava järjestelmä olisi monimutkaisempi tai testiskenaario pidempi, simulointiin kuluva aika saattaisi muuttua relevantiksi tekijäksi toteutustapaa valittaessa. Monimutkaisemman simulaattorin tapauksessa olisi merkittävää skaalautuuko simulointiin kuluva aika lineaarisesti vai epälineaarisesti ja tapahtuuko skaalautuminen molempien toteutustapojen kohdalla samanlaisesti.

5. YHTEENVETO

Työn tarkoituksena oli toteuttaa kahden PI-säätimen muodostaman kaskadisäätöpiirin simulaattori käyttäen MATLABin LEVEL-2 MATLAB S-function ohjelmointirajapintaa ja siinä onnistuttiin. Kaskadisäätöpiirissä on anti-windup-toiminto, joka on toteutettu tracking back-calculation -rakenteella. Tracking-toiminto mahdollistaa paikallisen asetusarvon kytkemisen kaskadipiirin sekundäärisäätimeen tai sekundäärisäätimen ulostulon ohjaamisen manuaalisesti.

Työssä toteutettiin kaskadisäätöpiirin simulaattori kahdessa vaiheessa. Ensin kaskadipiiri toteutettiin alkeislohkoilla. Sen jälkeen kaskadipiiri toteutettiin siten, että primääri- ja sekundäärisäädin oli toteutettu erillisinä S-funktioina. Lopuksi kaskadipiirin säätimen tuotiin yhteen S-funktioon. Lopullisen S-funktio-toteutuksen sekä alkeislohkototeutuksen läpi ajettiin tietty testiskenaario ja näistä simulaatioista saatuja vasteita vertailtiin toisiinsa. Toteutuksien vasteissa huomattiin eroavaisuuksia, merkittävimpänä S-funktiototeutuksen saturoituneessa ohjaussignaalissa esiintynyt voimakas kohina. Kohinan alkuperää ei tässä työssä selvitetty.

Tämän työn avulla voi luoda kaksitasoisen kaskadisäätöpiirin simulaattorin. Tässä työssä toteutetussa simulaattorissa on anti-windup-toiminto ja säätöpiiri on toteutettu yhtenä S-funktio-lohkona. Työssä toteutettuun simulaattoriin on mahdollista muokata lukijan tarpeisiin sopivia ominaisuuksia, tai simulaattorin pohjalta voidaan tuottaa simulaattori kolmi- tai useampitasoiselle kaskadirakenteelle. Lisäksi tätä työtä voi käyttää caseharjoituksena MATLABin S-funktioon tutustuttaessa.

LÄHTEET

[1] K. J. Åström and T. Hägglund, Advanced PID Control, Research Triangle Park, NC: ISA, 2006.

[2] T. E. Marlin, Process Control: Designing Processes and Control Systems for Dynamic Performance, New York: McGraw-Hill, 1995.

[3] S. Galeani, S. Tarbouriech, M. Turner and L. Zaccarian, A Tutorial on Modern Anti-windup Design, European Journal of Control, vol. 15, no. 3-4, pp. 418-440, 2009.

[4] A. Visioli, "Modified anti-windup scheme for PID controllers," IEE Proceedings: Control Theory and Applications, vol. 150, no. 1, pp. 49 - 54, 2003.

[5] J. Choi and S. Lee, Antiwindup Strategy for PI-Type Speed Controller, IEEE Transactions on Industrial Electronics, vol. 56, no. 6, pp. 2039-2046, 2009.

[6] İ. Kaya, N. Tan and D. P. Atherton, Improved cascade control structure for enhanced performance, Journal of Process Control, vol. 17, no. 1, pp. 3-16, 2007.

[7] R. G. Franks and C. W. Worley, Quantitative Analysis of Cascade Control, Industrial & Engineering Chemistry, pp. 1074 - 1079, 1956.

[8] P. U. Webb, Effect of secondary loop configuration on over-all response of a cascade control system, dissertation, Oklahoma State University, 1960, p. 71.

[9] W. L. Luyben, Parallel cascade control, Industrial & Engineering Chemistry Fundamentals, vol. 12, no. 4, pp. 463-467, 1973.

[10] D. G. Padhan and S. Majhi, An improved parallel cascade control structure for processes with time delay, Journal of Process Control, vol. 22, no. 5, pp. 884-898, 2012.

[11] F. Todeschini, M. Corno, G. Panzani, S. Fiorenti and S. M. Savaresi, Adaptive Cascade Control of a Brake-By-Wire Actuator for Sport Motorcycles, IEEE/ASME Transactions on Mechatronics, vol. 20, no. 3, pp. 1310-1319, June 2015.

[12] MathWorks, What Is an S-Function?, verkkosivu. Saatavissa (viitattu 31.3.2020): https://se.mathworks.com/help/simulink/sfg/what-is-an-s-function.html

[13] MathWorks, Create and Configure MATLAB S-Functions, verkkosivu. Saatavissa (viitattu 31.3.2020): https://se.mathworks.com/help/simulink/matlab-s-functions-1.html

[14] MathWorks, Write Level-2 MATLAB S-Functions, verkkosivu. Saatavissa (viitattu 5.5.2020): https://se.mathworks.com/help/simulink/sfg/writing-level-2-matlab-s-functions.html

[15] MathWorks, InitializeConditions, verkkosivu. Saatavissa (viitattu 5.5.2020): https://se.mathworks.com/help/simulink/sfg/initializeconditions.html

[16] MathWorks, double, verkkosivu. Saatavissa (viitattu 28.5.2020): https://se.mathworks.com/help/matlab/ref/double.html