StuCoSReC

Proceedings
of the 2018
5th Student
Computer
Science
Research
Conference

StuCoSReC

Proceedings of the 2018 5th Student Computer Science Research Conference

Edited by

Iztok Fister jr. and Andrej Brodnik

Reviewers and Programme Committee

Klemen Berkovič • University of Maribor, Slovenia
Lucija Brezočnik • University of Maribor, Slovenia
Patricio Bulić • University of Ljubljana, Slovenia
Zoran Bosnić • University of Ljubljana, Slovenia
Borko Bošković • University of Maribor, Slovenia
Janez Brest • University of Maribor, Slovenia
Andrej Brodnik • University of Primorska and University
of Ljubljana, Slovenia
Haruna Chiroma • Gombe State University, Nigeria
Mojca Ciglarič • University of Ljubljana, Slovenia
Jani Dugonik • University of Maribor, Slovenia
Iztok Fister • University of Maribor, Slovenia
Iztok Fister Jr. • University of Maribor, Slovenia
Matjaž Gams • Jozef Stefan Institute, Slovenia
Tao Gao • North China Electric Power University, China
Andres Iglesias • Universidad de Cantabria, Spain
Branko Kavšek • University of Primorska, Slovenia
Miklos Kresz • University of Szeged, Hungary
Niko Lukač • University of Maribor, Slovenia
Uroš Mlakar • University of Maribor, Slovenia
Marjan Mernik • University of Maribor, Slovenia
Eneko Osaba • University of Deusto, Spain
Vili Podgorelec • University of Maribor, Slovenia
Peter Rogelj • University of Primorska, Slovenia
Xin-She Yang • Middlesex University, United Kingdom
Damjan Vavpotič • University of Ljubljana, Slovenia
Grega Vrbančič • University of Maribor, Slovenia
Aleš Zamuda • University of Maribor, Slovenia
Nikolaj Zimic • University of Ljubljana, Slovenia
Borut Žalik • University of Maribor, Slovenia

Published by

University of Primorska Press
Titov trg 4, SI-6000 Koper

Editor-in-Chief

Jonatan Vinkler

Managing Editor

Alen Ježovnik

Koper, 2018

ISBN 978-961-7055-26-9 (pdf)

www.hippocampus.si/ISBN/978-961-7055-26-9.pdf

ISBN 978-961-7055-27-6 (html)

www.hippocampus.si/ISBN/978-961-7055-27-6/index.html

DOI: <https://doi.org/10.26493/978-961-7055-26-9>

© University of Primorska Press

Kataložni zapis o publikaciji (CIP) pripravili v Narodni in
univerzitetni knjižnici v Ljubljani

COBISS.SI-ID=296769024

ISBN 978-961-7055-26-9 (pdf)

ISBN 978-961-7055-27-6 (html)

Preface

It is no longer necessary to emphasize that computer science is omnipresent today. Terms as digitalization, Industry 4.0 etc. are frequently heard in the mass media and are becoming a part of our everyday life. The rapid advances in computer science require a large investment in research and development. In particular, it is important to educate young scientists, which is one of the objectives of the StuCoSReC conference.

In front of you is the proceedings of the fifth StuCoSReC conference. The StuCoSReC is intended for masters & PhD students to show their research achievements. Socializing is also important and the conference is an excellent opportunity for students to get to know each other. The conference connects students of computer science and goes beyond the invisible limits of faculties. The uniqueness of StuCoSReC conference is that it is organized each year by different higher education institution. The University of Ljubljana - Faculty of Computer and Information Science is proud to host the conference this year.

Twelve papers addressed this conference, covering several topics of the computer science. All the papers were reviewed by two international reviewers and accepted for the oral presentation.

ref. prof. dr. Nikolaj Zimic

Preface

II

Papers

<i>Regulacija plovca v zračnem tunelu (zračna levitacija) na osnovi mehke logike</i> • Primož Bencak, Dominik Sedonja, Andrej Picej, Alen Kolman, Matjaž Bogša	5–13
<i>Razpoznavanje literature v dokumentih in določanje njihovih sklicev v besedilu</i> • Matej Moravec, David Letonja, Jan Tovornik, Borko Boškovič	15–19
<i>Klasifikacija samomorilskih pisem</i> • Dejan Rupnik, Denis Ekart, Gregor Kovačevič, Dejan Orter, Alen Verk, Borko Boškovič	21–25
<i>Načrtovanje poprocesorja za pretvorbo NC/ISO G-kode v translacije za 3/5 osnega robota ACMA ABB XR701</i> • Gregor Germadnik, Timi Karner, Klemen Berkovič, Iztok Fister	27–33
<i>Using constrained exhaustive search vs. greedy heuristic search for classification rule learning</i> • Jamolbek Mattiev, Branko Kavšek	35–38
<i>Real-time visualization of 3D digital Earth</i> • Aljaž Jeromel	39–42
<i>Towards an enhanced inertial sensor-based step length estimation model</i> • Melanija Vežočnik, Matjaž Branko Jurič	43–46
<i>Breast Histopathology Image Clustering using Cuckoo Search Algorithm</i> • Krishna Gopal Dhal, Iztok Fister Jr., Arunita Das, Swarnajit Ray, Sanjoy Das	47–54
<i>Time series classification with Bag-Of-Words approach post-quantum cryptography</i> • Domen Kavran	55–59
<i>The problem of quantum computers in cryptography and post-quantum cryptography</i> • Dino Vlahek	61–64
<i>Named Entity Recognition and Classification using Artificial Neural Network</i> • Luka Bašek, Borko Boškovič	65–70
<i>Context-dependent chaining of heterogeneous data</i> • Rok Gomišček, Tomaž Curk	71

Regulacija plovca v zračnem tunelu (zračna levitacija) na osnovi mehke logike

Primož Bencak
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
primoz.bencak@
student.um.si

Dominik Sedonja
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
dominik.sedonja@
student.um.si

Andrej Picej
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
andrej.picej@
student.um.si

Alen Kolman
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
alen.kolman@
student.um.si

Matjaž Bogša
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
matjaz.bogsa@
student.um.si

POVZETEK

Teorija mehke logike je prisotna že dobrih 40 let, razvita pa je bila z namenom preprostega in intuitivnega vodenja sistemov. Mehka logika namreč posnema človeško sklepanje ter sprejemanje odločitev na podlagi dejstev. V preteklih desetletjih se je uveljavila kot učinkovita in procesorsko nezahtevna alternativa klasičnim tehnikam vodenja. Čeprav se pojavljajo pomisleki o njeni uporabnosti na zahtevnejših aplikacijah, je vendarle našla svoj prostor v mnogih segmentih našega življenja.

Mehka logika nam omogoča, da lahko sami določimo karakteristiko regulatorja, napram linearnemu PID regulatorju, kjer je le-ta vedno linearna in zato v večini primerov neprimerna za vodenje nelinearnih sistemov, razen v omejenih območjih delovanja. V članku je tako predstavljena regulacija valjastega plovca znotraj zračnega tunela na osnovi mehke logike z dolgim mrtvim časom (angl. *dead time*). Opravljene so bile simulacije omenjenega sistema v MATLAB/Simulink-u, izdelano je bilo preizkuševališče za vodenje plovca ter izvedena primerjava med klasično tehniko vodenja in tehniko vodenja z mehko logiko. Preizkuševališče je namenjeno učenju sistemov daljinskega vodenja in preizkušanju različnih tipov regulatorjev.

Ključne besede

zračna levitacija, mehka logika, mikrokrmilniki, LabVIEW

1. UVOD

Zračna levitacija predstavlja mirovanje (v nadaljevanju: lebdenje) telesa v neki točki dvignjeni od tal, zaradi umetno ustvarjene sile zračnega toka, ki je nasprotno enaka sili teže telesa [3]. Lebdenje objekta (plovca) v cevi je s stališča regulacije zahtevnejši problem, saj gre za nelinearen dinamični sistem drugega reda. Za vodenje takega sistema moramo zelo dobro nastaviti parametre PID regulatorja ali se poslužiti drugega načina vodenja, kot je mehka logika. Sistem je bil najprej zgrajen in preizkušen v simulacijskem okolju, nato pa je bil prenešen v realno okolje in za dane razmere optimiziran.

Ideja za izdelavo sistema lebdenja izhaja iz članka [3], v katerem avtorji s PID regulatorjem uspešno vodijo takšen sistem. Izrazito nelinearen sistem avtorji linearizirajo v okolici delovne točke, sledi identifikacija modela in nato iz dobljenih formul izpeljejo prenosno funkcijo, ki je vstavljena v uporabniški vmesnik, preko katerega vodijo sistem.

Dušan Fister pa predlaga uporabo naprednih optimizacijskih algoritmov po vzoru iz narave za nastavljanje parametrov PID regulatorja, da dosežemo najboljše razmerje med vzporskim časom, prenihajem in statičnim pogreškom, ter ga preizkusi na dvoosnem SCARA robotu [4].

V članku je predstavljena rešitev, kjer postopka linearizacije in določanja parametrov PID regulatorja ni potrebno izvesti, saj ni potrebno poznati niti funkcije sistema. Uporabili smo mehki regulator, pri čemer kot vhodna podatka uporabimo le oddaljenost od točke *ekvilibrium*a oz. referenčne točke (točka v prostoru stanj sistema, kjer je doseženo ravnotežno stanje sistema) in odvod te razdalje, torej hitrost premikanja objekta.

Mehka logika se uporablja na veliko različnih področjih kot so: aeronavtična industrija [10], avtomobilska industrija (predstavljanje menjalnika [5], strategija nadzora električnega vozila [9]), medicina (večnamenska kontrola anestezije, nadzorovanje arterijskega tlaka med anestezijo) [17], nadzorovanje

polnjenja in praznjenja baterij [12], procesiranje slik v industriji [1] in še na mnogih drugih.

Struktura članka v nadaljevanju je naslednja: drugo poglavje opisuje teoretično ozadje lebdenja v zraku in mehke logike, tretje poglavje govori o simulaciji sistema v MATLAB/-Simulink-u, v četrtem poglavju je predstavljen postopek načrtovanja fizičnega dela sistema (angl. *hardware*), v petem poglavju je govora o programskem delu (angl. *software*), sledijo še rezultati in na koncu sklep s podanimi možnostmi za izboljšavo obstoječega sistema.

2. LEBDENJE V ZRAKU

Lebdenje v zraku je v osnovi preprosto opisati, saj mora veljati le: $\sum F = 0$. To pomeni, da je potrebno izpolniti pogoj $F_g = F_z$. Vendar, ko želimo za izbrani sistem na ta način izračunati vrednost F_z , ki predstavlja silo ustvarjeno s pomočjo pretoka zraka v cevi, naletimo na težavo, saj moramo za optimalno vodenje poznati razne koeficiente, ki niso konstante (težavo predstavljajo koeficient potiska – odvisen od Reynoldsovega števila; gostota zraka – odvisna od temperature, nadmorske višine, ... in hitrost zraka v cevi). Iz drugega Newtonovega zakona lahko zapišemo ravnotežno enačbo lebdenja v zraku (1) in nato razčlenimo posamezno silo (2).

Enačbe:

$$F = F_z - F_g, \quad (1)$$

$$F = \frac{1}{2} \cdot C_d \cdot \rho \cdot A \cdot (v_w - z')^2 - m \cdot g, \quad (2)$$

pri čemer je C_d koeficient potiska, ρ gostota zraka, A presek objekta na katerega deluje zrak, v_w hitrost zraka v cevi, z višina na kateri se nahaja objekt v cevi, m masa objekta in g težnostni pospešek.

Slika 1: Delovanje sil na objekt v cevi. Vir [3].

2.1 Mehka logika

Začetki mehke logike segajo v leto 1985, ko sta to metodo predstavila Takagi in Sugeno. Pri mehki logiki gre za manj matematično zapletene zapise relacij med vhodnimi in izhodnimi spremenljivkami, saj so spremenljivke lahko lingvistične (podane z besedami naravnega jezika – velik, majhen, hiter, počasen, ...). Operacije nad mehkiimi spremenljivkami izvajamo z mehkiimi operatorji (unija, presek in komplement). Sestaviti pa je potrebno tudi bazo pravil, ki je v osnovi pri mehkih sistemih zgrajena iz "če-potem" (angl. "if-then") mehkih pravil, ki opisujejo vzročno – posledično dogajanje v sistemu. Glede na izhodno spremenljivko ločimo več tipov mehkih pravil, in sicer Mamdani [6] sisteme (vhodi in izhodi so lingvistični), Takagi-Sugeno (mehka pravila, kjer je izhod ostra vrednost) [11] in Singleton (izhod

je namesto funkcije kar število). Pisanje baze pravil mehke logike za posamezen sistem ni trivialno kakor nakazuje metoda, saj je potrebno dobro poznavanje sistema, zato pravila pišejo strokovnjaki (angl. *expert*) dotičnega področja. Strokovnjak mora določiti tudi vhodne in izhodne pripadnostne funkcije. V kompleksnejših sistemih lahko število funkcij in pravil preseže zmožnost človeškega uma in strokovnjak ne more zapisati ustreznega števila pravil [13].

V sisteme regulacije s pomočjo mehke logike velikokrat vstopa t.i. ostra vrednost, v obliki podatka iz senzorja (npr. temperatura zraka). To informacijo je potrebno mehčati (angl. *fuzzification*), kar pomeni, da informacijo spremenimo v mehko vrednost (besedno oz. številčno). Ta postopek poteka s pomočjo pripadnostnih funkcij (funkcije, ki določajo pripadnost elementov iz senzorja posamezni mehki množici in lahko zavzamejo vrednost med 0 in 1). Poznamo več vrst pripadnostnih funkcij, in sicer Z, S, π , sigmoidno, Gaussovo, zvončasto, odsekoma linearno, trikotno in trapezoidno pripadnostno funkcijo. Od sistema in strokovnjaka je odvisno katere pripadnostne funkcije bodo uporabljene v posamezni situaciji. Mehko sklepanje se izvede na podlagi baze pravil. Na ta način se vhodni podatki pretvorijo v izhodne, ki pa so mehke vrednosti. Z mehko vrednostjo računalnik oz. krmilnik ne more operirati, zato je mehko vrednost potrebno ponovno ostriti (angl. *defuzzification*), da dobimo numerično vrednost, s katero lahko računalnik računa oz. na podlagi le-te proži določeno akcijo na aktuatorju. Poznamo več metod ostrenja, in sicer metodo največje vrednosti, težiščno metodo in poenostavljeno težiščno metodo. V praksi se pogosto uporablja poenostavljena težiščna metoda, ki je računsko dokaj preprosta in ne zahteva prevelike procesorske moči [13].

Slika 2: Shematski prikaz mehkega inferenčnega stroja. Vir [13].

Ugotovimo lahko, da se na ta način elegantno izognemo zapletenemu matematičnemu zapisu in reševanju diferencialnih enačb zgolj z dobrim poznavanjem procesa. Kljub temu se pojavljajo pomisleki, ki odvrčajo od uporabe mehke logike. Argumenti proti se nanašajo predvsem na to, da ni sistematičnega pristopa k načrtovanju tovrstnih sistemov, pri kompleksnejših sistemih lahko mehka logika postane ovira in ogroža stabilnost sistema [8], ter da predstavlja manjvreden inženirski pristop k vodenju [13].

3. SIMULACIJA LEBDENJA V ZRAKU

Za uspešno simulacijo modela je potrebno implementirati naslednje sestavne dele:

- plovec, ki predstavlja reguliran objekt

- regulacijo (regulator in Smithov prediktor)

Prenosna funkcija reguliranega sistema je sestavljena iz dveh sklopov - ventilatorja in reguliranega objekta (plovca). Prenosno funkcijo ventilatorja zapišemo s členom prvega reda, plovca pa s členom drugega reda [3]. Slednjo razvijemo na podlagi ravnotežnih enačb, ki smo jih zapisali zgoraj (enačbi (1) in (2)). Enačbo (2) preuredimo tako, da lahko iz rezultirajoče sile F izrazimo pospešek z'' . Enačbo lineariziramo v okolici delovne točke in nad njo izvedemo Laplaceovo transformacijo. Izrazimo izhod ($\Delta z(s)$ - prirastek višine od ravnovesne točke) proti vходу ($\Delta v(s)$ - prirastek hitrosti zraka od ravnovesne lege) ter tako dobimo prenosno funkcijo $Z(s)$. Ventilator modeliramo kot člen prvega reda, z ojačenjem k_v in vzponskim časom τ_s . Celotna izpeljava funkcije je podana v članku [3], zaradi nazornosti pa so tukaj podani samo bistveni poudarki.

Prenosna funkcija ventilatorja je tako podana kot:

$$V(s) = \frac{v(s)}{u(s)} = \frac{k_v}{\tau_s s + 1}, \quad (3)$$

kjer je k_v konstanta ventilatorja, podana v $\frac{m/s}{V}$ in τ_s vzponski čas sistema (čas, da ventilator doseže nazivno hitrost pretoka zraka).

Prenosno funkcijo objekta v vetrovniku po izpeljavi zapišemo kot:

$$Z(s) = \frac{\Delta z(s)}{\Delta v(s)} = \frac{1}{s} \cdot \frac{a}{s(s+a)}, \quad (4)$$

Parameter a , je podan kot:

$$a = \frac{2 \cdot g}{v_{eq}}, \quad (5)$$

v_{eq} pa je potrebna hitrost zraka, da vzpostavimo ravnovesno lego objekta v vetrovniku (dosežemo *ekvililibrium*):

$$v_{eq} = \sqrt{\frac{g \cdot m}{\frac{1}{2} \cdot C_d \cdot \rho \cdot A}} \quad (6)$$

Skupna prenosna funkcija sistema H_s je podana kot:

$$H(s) = V(s) \cdot Z(s) \quad (7)$$

in je tretjega reda. Integratorski člen v prenosni funkciji $Z(s)$ nakazuje na potencialno nestabilnost sistema. Objekt v vetrovniku ima namreč tendenco, da odleti iz cevi, v kolikor se ne poslužimo pravega pristopa pri načrtovanju regulatorja. Prav tako zaradi integralskega člena v prenosni funkciji le-tega ne potrebujemo v regulatorju, zato je teoretično dovolj le regulacija s PD regulatorjem (v okolici delovne točke). Kljub temu, se v praksi doda zelo mala vrednost integralnega (I) člena z namenom izboljšanja odziva.

Simulacije v MATLAB-u so bile narejene še preden so bili znani vsi parametri sistema (C_d , hitrost zraka, hitrost zajema podatkov,...), da je bilo okvirno preverjeno delovanje sistema. Tudi če bi uspelo natančno izmeriti vse potrebne parametre, se modelirani sistem zaradi raznih motenj iz okolice, ki jih je prezahtevno modelirati, ne bi odzval enako kot realni sistem. Treba je poudariti tudi to, da naj bi izračunano delovanje regulatorja zaradi linearizacije, veljalo le v

območju okoli delovne točke. Za nastavitve parametrov regulatorja je bilo uporabljeno MATLAB-ovo orodje *pidtool*. Regulacija oziroma zaprto zankno vodenje vsebuje povratno informacijo izhoda s pomočjo senzorjev in povratne zanke sistema.

V osnovi je bila ideja vodenje na daljavo, kar pomeni, da se algoritem regulacije ne izvaja na istem krmilniku, ki krmili ventilator. Posledično so podaljšani časi do prihoda podatka v regulator. V kolikor se sistem obnaša zelo dinamično (časovne konstante sistema so majhne), lahko pride do težav pri regulaciji. Zaradi je PID regulatorju bil dodan Smithov prediktor, ki glede na matematični model regulirane proge predvidi obnašanje regulatorja med časom, ko ni na voljo novega podatka. V vseh realnih sistemih prihaja v povratni zanki do zakasnitev, ki jih je potrebno za uspešno regulacijo upoštevati. Smithov prediktor uporablja model predikcije zakasnitve izhoda procesa, primerja razliko med predvidenim in željenim izhodom ter določi potrebno korekcijo [7]. Prednost dodanega Smithovega prediktorja v primerjavi s samim PID regulatorjem je hitrejši odziv in manjši prenihaj pri odzivu na vrednost nič, kot prikazuje slika 3.

Slika 3: Stopnični odziv PID regulatorja brez in z uporabo Smithovega prediktorja.

Pri dejanskem nastavljanju parametrov regulatorja simulacije niso bile v veliko pomoč, kar je tudi eden izmed glavnih razlogov, da je bilo uporabljeno vodenje z mehko logiko. Z enakimi pripadnostnimi funkcijami in enakimi pravili, ki so bila uporabljena na realnem sistemu, je bil simuliran še odziv na stopnico z regulatorjem na osnovi mehke logike. Rezultati so napram pridobljenim s PID regulatorjem mnogo boljši. Vzponski čas je padel za okoli 15%, mehki regulator pa je prav tako izničil prenihaj, ki se pojavlja pri PID regulatorju, kar je vidno na sliki 4.

Pri načrtovanju mehkega regulatorja se tako nismo ukvarjali z matematičnim modelom in konstantami sistema, ampak so bile pripadnostne funkcije postavljene glede na dejanski odziv sistema, hitrost v_{eq} pa določena empirično.

Slika 4: Stopnični odziv mehkega in PID regulatorja.

4. IZVEDBA REALNEGA SISTEMA

V nadaljevanju bo predstavljena priprava strojne in programske opreme za preizkuševališče in reguliran objekt, tj. plovec.

4.1 Priprava strojne opreme

Teoretično delovanje mehkega regulatorja je bilo potrebno preveriti tudi v praksi, zato smo zbrali oz. izdelali vse potrebne komponente za regulacijo. Sistem sestavlja:

- plovec, ki predstavlja reguliran objekt (vsebuje MPU-6050, mikrokrmilnik STM32F103, Bluetooth modul HC-05, 3.7 V LiPo baterijo)
- tunel oz. regulacijska proga (cev iz pleksi stekla, usmernik, ventilator Dr. Mad Thrust KV3300)
- zunanje vezje za regulacijo (STM32F407 Discovery Board, optični senzor VL53L0X, krmilnik za ventilator ESC 20A)

Slika 5: Shematski prikaz regulacijske proge.

Prvotna ideja regulacije plovca v cevi je zajemala regulacijo preko kotnega zasuka oz. pospeška. Plovec, tj. telo z aerodinamično obliko, naj bi zato vseboval vse komponente potrebne za zajem, obdelavo in posredovanje podatkov gostujočemu sistemu. V ta namen je bila v plovec vgrajena mikrokrmilniška plošča STM32F103C8T6 (t.i. *Blue Pill*), kombinirani senzor MPU-6050, baterijsko napajanje ter vmesnik Bluetooth HC-05. Senzor MPU-6050 vsebuje integrirani MEMS (Mikro Elektro Mehanski Sistem) pospeškometer in žiroskop. Ta sočasno zajema podatke v X, Y in Z osi, ter je preko komunikacije I²C povezan s krmilnikom. Prikaz, nadzor in regulacija potekajo na računalniku, kamor se obdelani podatki pošiljajo preko Bluetooth komunikacije. Zaradi potrebe po napajanju plovec vsebuje 3.7 V LiPo baterijo s kapaciteto 270 mAh, ki naj bi zagotavljala energijo celotnemu sistemu do 4 ure.

Plovec je zaradi potrebe po čim bolj stabilnem gibanju in vrtenju okoli svoje osi, izdelan v obliki kaplje z dodanimi lopaticami. Ohišje plovca je bilo načrtovano v programu SolidWorks in natisnjeno s 3D tiskalnikom. Potrebni so bili nekaj iteracij, da so bile zagotovljene dimenzijske tolerance in usklajene velikosti delov. V programu SolidWorks je bila prav tako opravljena simulacija pretoka zraka po cevi in izračun težišča plovca z vsemi komponentami (slika 6).

Slika 6: 3D model plovca z vgrajenimi komponentami.

Regulacijsko progo predstavlja votla cev, skozi katero ventilator piha zrak (slika 18). Za ventilatorjem je nameščen usmernik, ki skrbi za enakomeren laminarni pretok zraka, s tem pa je preprečeno nastajanje vrtnicev, ki neugodno vplivajo na gibanje in regulacijo plovca. Za ventilator je bil izbran Dr. Mad Thrust KV3300, ki ga poganja brezkrtačni elektromotor z nazivno močjo 400 W in ga krmili ESC z maksimalnim tokom 20 A. Regulacija hitrosti se izvaja preko pulzno širinske modulacije, pri čemer mora biti frekvenca preklapljanja 50 Hz. Zaradi težav s pridobivanjem koristnih informacij za regulacijo s senzorja MPU-6050, je bil sistemu dodan optični senzor za merjenje razdalje. Širina plastične cevi ni omogočala povečanja plovca, ki je že bil na meji svoje velikosti, zato je bil senzor nameščen izven plovca na nosilec, pritrjen na vrhu cevi. Izvajanju regulacije je tako namenjeno zunanje vezje, ki ga sestavljata krmilnik STM32F407 Discovery Board in senzor za merjenje razdalje. Krmilnik je namenjen generiranju pulzno širinske modulacije za krmiljenje ventilatorja in posledično pretoka zraka skozi cev, kar za regulacijo predstavlja vhodno spremenljivko. Sprva je

bil kot merilnik razdalje uporabljen infrardeči senzor Sharp GP2Y0A21YK0F (10 – 80 cm), ki je bil zaradi prevelikih nihanj v izmerjeni razdalji, zamenjan z optičnim senzorjem STM VL53L0X. Slednji ima popolnoma linearno karakteristiko ter višji dolet (do 2 m), vendar komunicira preko protokola I^2C . Regulacija se izvaja na računalniku, ki je s krmilnikom povezan preko USB komunikacije.

Slika 7: 3D model zgornjega dela preizkuševališča.

4.2 Priprava programske opreme

Za namene projektne naloge so bili napisani programi, ki so se izvajali na uporabljenih krmilnikih, kot tudi program, ki se je izvajal na osebem računalniku in je skrbel za regulacijo. Za mikrokrmilnik z oznako STM32F103C8T6, ki se je nahajal v plovcu, ter za mikrokrmilnik serije STM32F407VG Discovery Board, je bil uporabljen programski jezik C. Pri programiranju je bil v pomoč namenski, brezplačni program CubeMX, ki omogoča lažjo konfiguracijo vseh enot mikrokrmilnika, pripravi vse potrebne knjižnice in ustvari grobo zasnovo vseh programov [16]. Pisanje programa za oba mikrokrmilnika pa je bilo opravljeno v integriranem razvojnem okolju (IDE) μ Vision programskega paketa Keil. Oba omenjena programa sta namenjena delu in programiranju mikrokrmilnikov z Arm Cortex procesorji [2]. Pri programiranju krmilnikov so bile uporabljene tudi posebne visokonivojske knjižnice, t.i. knjižnice HAL (angl. *Hardware Abstraction Layer*), ki so posebej namenjene programiranju krmilnikov serije STM32. HAL knjižnice olajšajo in pospešijo programiranje, saj omogočajo delo z lažje razumljivimi ukazi [15]. Za mikrokrmilnik v plovcu (STM32F103C8T6) je bil napisan program, ki je omogočal branje pospeškov in kotnih hitrosti iz pospeškometra. Za komunikacijo med senzorjem in krmilnikom je skrbel protokol I^2C . Komunikacija je bila implementirana s pomočjo že napisanih prosto dostopnih knjižnic. Vsi podatki so bili ustrezno obdelani in s pomočjo Bluetooth modula HC-05 poslani na računalnik. Podatki so bili sprva prikazani s pomočjo programa LabVIEW, kasneje pa je bilo načrtovano te podatke uporabiti v Kalmanovem filtru pri izračunu višine plovca v vetrovniku. Ta naloga se je izkazala za zelo zahtevno, saj se za pravilno delovanje Kalmanovega filtra priporoča zelo natančen model sistema, poleg tega so podatki iz senzorja MPU-6050 vsebovali preveč motenj in šuma, da bi se iz njih dalo razbrati uporabne meritve, kar bi

povzročalo težave pri obdelavi podatkov in regulaciji. Kotna hitrost se je spreminjala naključno zaradi trkov v stene plastične cevi, zato ni bilo mogoče pridobiti uporabnih podatkov, pospeškometer pa vrača zelo šumne meritve, ki so same po sebi neprimerne za dvojno integracijo. Pri mikrokrmilniku (STM32F407VG), ki je skrbel za branje podatkov iz senzorja časa preleta (VL53L0X), je bila prav tako uporabljena komunikacija I^2C . Tudi ti podatki so bili poslani na računalnik, tokrat preko USB povezave. Na računalniku so bili podatki nato uporabljeni v regulaciji. Rezultat regulacije je bil podatek za hitrost vrtenja ventilatorja, ki je bil nato poslan nazaj na isti mikrokrmilnik. Na mikrokrmilniku je bil napisan tudi program za generiranje pulzno-širinskega signala, ki je bil namenjen krmilniku hitrosti vrtenja (ESC) motorja, glej sliko 8.

Slika 8: Shematski prikaz regulacijske proge.

Kot orodje za prejemanje, pošiljanje in obdelavo podatkov na računalniku je bil uporabljen program LabVIEW. Za prejemanje in pošiljanje podatkov je bilo v njem uporabljeno orodje VISA, ki omogoča branje in pisanje podatkov iz/na računalniški komunikacijski vmesnik. To orodje je bilo uporabljeno pri prejemanju podatkov iz Bluetooth povezave, kot tudi pri prejemanju in pošiljanju podatkov preko USB povezave s krmilnikom STM32F407. Sprva je bil program za regulacijo sestavljen s PID regulatorjem, ki ga je enostavno sestaviti z ustreznim blokom v programu LabVIEW, nato pa je bil zaradi težav implementacije ter pravilnega delovanja na vetrovniku ustvarjen mehki regulator tipa Mamdani.

Slika 9: Blok diagram regulacijske zanke v LabVIEW-u.

V programu LabVIEW, glej sliko 9, je bil takšen regulator ustvarjen s pomočjo orodja imenovanega *Fuzzy System*

Designer. S tem orodjem se ustvari mehki regulator, kateremu določimo število in oblike pripadnostnih funkcij ter napišemo bazo pravil. Kot je razvidno iz slike 10, vhodni množici v mehki regulator predstavljata pogrešek med dejansko in željeno višino plovca v vetrovniku, ter njegov odvod.

Mehki regulator spreminja vrednost pulzno-širinskega signala v malem območju okoli vrednosti, pri kateri plovec lebdi v vetrovniku. Hitrost v_{eq} potrebna za levitacijo pa je odvisna od mase plovca in zmožnosti premikanja po cevi – če se plovec zatika ob cev, je potrebna višja hitrost in obratno.

Slika 10: Regulacijska proga z vključenim mehkih regulatorjem. Veličina y predstavlja višino, na kateri se nahaja plovec.

5. SESTAVLJANJE KOMPONENT

Sestava komponent vetrovnika je bila sorazmerno preprosta, vendar je bilo kljub temu med gradnjo sistema potrebno rešiti nekaj težav. Optični senzor VL53L0X komunicira s krmilnikom STM32F407 preko I^2C komunikacije, zato je bila omejena dolžina kabla, ki povezuje senzor s krmilnikom. Krmilnik je bilo zato potrebno namestiti v neposredno bližino senzorja. V ta namen je bil izdelan preprost nosilec, ki je bil z vročim lepilom pritrjen na cev vetrovnika, kot prikazuje slika 17. Na spodnji strani je bila cev skupaj z usmernikom zraka vstavljena v osnovo, natisnjeno s 3D tiskalnikom. Iz spodnje strani je bil vanjo nameščen še ventilator ter v škatlo ob osnovi krmilnik motorja, ki je z dvema žicama povezan na mikrokrmilnik (slika 16). Prvi testi so pokazali, da se v sistemu pojavi precej vibracij, ki neugodno vplivajo na premikanje plovca po cevi. Oblika plovca, predvsem širina, je omejena s komponentami, ki se nahajajo v njem, poleg tega pa še z notranjim premerom cevi. Preširok plovec se zaradi preslabega obtoka zraka ni vrtil in se je zatikal ob cev. Ožji plovec je bil s stališča vrtenja ob zagonu boljša izbira, vendar zaradi notranjih komponent in načina izdelave, njegovega težišča ni bilo lahko spraviti na os vrtenja, zato je kmalu po začetku vrtenja postal nestabilen. Plovec je bil še dodatno obtežen, saj so se na ta način povečale časovne konstante sistema in spustile zahtevo po visokem vzorčevalnem času. Pri končni izvedbi so združene dobre lastnosti obeh prej omenjenih izvedb. Na začetku je plovec ožji in zagotavlja zadosten obtok zraka, zadnji del plovca pa je širši. Opletanje plovca je rešeno s tem, da je razlika med širino plastične cevi in zadnjega dela plovca le 1 mm, istočasno pa je s krilci omogočeno njegovo vrtenje.

Čeprav je bila prvotna ideja o regulaciji preko podatkov žiroskopa in pospeškometra opuščena, pa je bilo vrtenje plovca pomembno, saj se je plovec ob konstantnem vrtenju lažje premikal ter lažje dosegel željeno vrednost višine. Vibracije,

Slika 11: Plovec aerodinamične oblike.

ki jih povzroča ventilator, so odpravljene tako, da je plastična osnova privita na debelejši in težji kos podlage. Za pravilno delovanje sistema, je zelo pomemben tudi nemoten, predvsem pa zadosten dotok zraka, ki služi tudi hlajenju krmilnika ESC. Ta se lahko zaradi visokih nazivnih tokov pri krmiljenju ventilatorja temu primerno segreje. Napajanje sistema je možno izvesti na več načinov, z uporabo laboratorijskega napajalnika (0-50V, 20 A), računalniškega napajalnika ali s pomočjo LiPo baterije 3S, ki je pogosto uporabljena v modelarstvu. Na ta način je zagotovljena tudi prenosljivost preizkuševališča, ki je primerno za demonstracijsko rabo v pedagoškem procesu.

6. REZULTATI

V končnem izdelku nam PID regulatorja ni uspelo implementirati predvsem zaradi:

- predolgega časa zajema položaja plovca. Čas potreben za pridobitev novega podatka o višini je trajal preko 100 ms, kar je bilo za regulacijo sistema občutno preveč. Dodatno je potrebno upoštevati še čas do nove nastavitve hitrosti, ki ga zaradi USB komunikacije v obe smeri, ter obdelave algoritma ocenjujemo na vsaj 10 ms. Težava s posredovanjem novega podatka sicer ni bila v samem senzorju, saj le-ta zmora zagotoviti novo vrednost vsakih 20 ms v režimu hitrega zajema podatkov, temveč v izvajanju prekinitvene rutine. Najnižji, še mogoč čas proženja prekinitve preko časovnika je znašal 100 ms. Natančnega razloga, zaradi katerega nismo mogli doseči hitrejšega izvajanja prekinitve, ne poznamo, predvidevamo pa, da je bil procesor v tem času preobremenjen.
- nezmožnosti PID regulatorja, da mu nastavimo karakteristiko, saj se v celotnem področju regulator obnaša enako. PID regulator, ki je bil sposoben držati plovec v bližini referenčne točke smo sicer izdelali, vendar z rezultatom nismo bili zadovoljni.

V obeh omenjenih pomanjkljivostih PID regulatorja pa ima uporaba mehkega regulatorja prednost. Uporaben je pri počasnejših časih vzorčenja, njegova največja prednost pa je ta, da omogoča bolj transparentno nastavljanje njegovih parametrov [13]. Strokovnjak, ki načrtuje regulator ima tako izboljšano predstavo o relaciji med vhodnimi in izhodnimi podatki, kar pa je za PID težko trditi. Glavni parametri, od katerih je odvisno delovanje mehkega regulatorja so šte-

vilo in položaj pripadnostnih funkcij ter pravila, ki povežejo vhodne in izhodne pripadnostne funkcije.

Končni regulator je imel 5 pripadnostnih funkcij za vsako vhodno in izhodno množico, med sabo pa jih je povezovalo 25 pravil. Vse pripadnostne funkcije so trikotne nezvezne oblike, kakor so prikazane na sliki 12.

Slika 12: Pripadnostne funkcije mehkega regulatorja in karakteristika - nelinearna ravnina.

Z uporabo mehkega regulatorja, smo se prav tako izognili uporabi Smithovega prediktorja, ki sicer v nekoliko spremenjeni obliki obstaja tudi za mehki regulator [14].

Težave z dolgimi časi zajema (T_s) smo rešili tudi na način, da smo v končni regulaciji uporabili težji plovec, zaradi česar je bil odziv sistema počasnejši in posledično je mehki regulator lažje in bolje reguliral sistem. Ugotovili smo tudi, da je mehki regulator manj občutljiv na spremembe reguliranega objekta – plovca, je pa potrebno poudariti, da telesa z nizko maso (žogica za namizni tenis) na ta način ni mogoče regulirati, ker so časovne konstante tega objekta precej nižje od plovca, uporabljenega v naši nalogi.

Slika 13: Uporabljeni plovci za testiranje preizkuševališča.

Pred začetkom regulacije je bilo potrebno poiskati le vrednost hitrosti zraka v cevi, pri kateri plovec v vetrovniku lebdi (v_{eq}), kar naredimo z enostavnim povečevanjem hitrosti vrtenja motorja z drsnikom na nadzorni plošči v programu LabVIEW (slika 15). Takšen način zagona vetrovnika je bil uporabljen zlasti zaradi uporabljenih različnih plovcev (slika 13), pri katerih so se ravnovesne vrednosti vrtenja motorja med seboj. Ko je ta vrednost zabeležena in vpisana, lahko regulator ustrezno regulira višino plovcev različnih oblik in tež.

Slika 14: Stopnični odziv mehkega regulatorja na realnem sistemu. Uporabljen je bil zeleni plovec z maso 25 g.

Kot je razvidno iz stopničnega odziva na sliki 14, se plovec med skrajnima višinama premakne v približno dveh sekundah. Ob željenem manjšem nastavitvenem času pride do večjih prenihajev, kar lahko pomeni, da plovec odleti iz vetrovnika in zadane senzor, ki je nameščen nad vetrovnikom ter ga poškoduje. Podobno se zgodi pri spuščanju plovca, kjer lahko plovec pri večjih prenihajih poškoduje usmernik zračnega pretoka. Iz teh razlogov je bil narejen kompromis med prenihajem in nastavitvenim časom pri odzivu mehkega regulatorja.

Slika 15: LabVIEW kontrolna plošča.

Sistem v taki obliki je tudi stabilen in odporen na motnje. Preizkuševališče (slike 16, 17, 18) lahko dvignemo ali mu spremenimo naklon, pri čemer regulator še vedno odlično opravlja svoje delo.

Slika 16: Spodnji del preizkuševališča.

Slika 17: Zgornji del preizkuševališča.

7. SKLEP

Članek zajema izvedbo regulirane levitacije plovca v zračnem tunelu, pri čemer gre za sistem daljinskega vodenja. Regulacija se izvaja na oddaljenem sistemu, v našem primeru na osebnem računalniku. Višino plovca v cevi nam je uspelo regulirati ne glede na visoke mrtve čase, kar nam sicer s PID regulatorjem ni povsem uspelo. Aplikacija na osebnem računalniku (LabVIEW) nam omogoča, da nastavimo želeno višino, sistem pa se samodejno odzove. Poleg tega lahko v njej istočasno spremljamo realne vrednosti pospeškov in kotnih hitrosti plovca, v kolikor uporabimo omejeno izvedbo z vgrajenim krmilnikom. Tekom dela smo naleteli na več težav, največ težav je predstavljala strojna oprema in implementacija I^2C komunikacije med optičnim senzorjem (VL53L0X) in mikrokontrolnikom (STM32F407). Dodatne težave, ki jih nismo povsem odpravili so vibracije cevi, ki vplivajo na vrtenje plovca in ga občasno popolnoma zaustavijo, saj se ta namesto da bi se vrtel, prične odbijati od stene. Problem je tudi trenje med plovcem in steno vetrovnika, ki prav tako zavira vrtenje, ter pa opletanje plovca, ker ta ni povsem centriran. Vsekakor pa so nekatere izmed teh težav privedle do tega, da smo se odločili za uporabo mehke logike.

Težave, ki so ostale, predstavljajo možnosti nadaljevanja dela na projektu in dodatnih izboljšav sistema. Boljše rezultate obeta uporaba širše cevi, tako bi bili nekoliko manj omejeni z oblikami in dimenzijami plovcov. Regulator, ki trenutno nadzira sistem je bil v primerjavi s PID, izdelan v zelo kratkem času, vendar je že v začetni fazi testiranja dal spodbudne rezultate, vendar pa smo zaradi pomanjkanja časa morali izpustiti napredno optimizacijo procesa. Do-

Slika 18: Celotni sistem.

kazali smo, da lahko z mehko logiko vodimo kompleksnejši sistem, ki ima dolge čase zajema in bi ga sicer s PID regulatorjem ob enakih pogojih zelo težko vodili brez dodanega Smithovega prediktorja ali nasploh. Za nastavitve regulatorja smo porabili veliko manj časa, kot smo ga sicer za poskus implementacije PID regulatorja, ki nam ni dajal pravih rezultatov. Ob predpostavki, da se mase plovca, na katerem je bil mehki regulator nastavljen ne spremenijo bistveno, regulator ohranja solidne odzive, bolj kot sama masa pa na rezultate bolj vpliva hrapavost in stabilnost plovca v cevi.

ZAHVALA

Iskreno se zahvaljujemo rednemu profesorju dr. Dušanu Gleichu ter asistentu mag. Blažu Pongracu s Fakultete za elektrotehniko, računalništvo in informatiko v Mariboru za pomoč pri izvedbi simulacij in načrtovanju programske opreme.

REFERENCES

- [1] C. G. Amza and D. T. Cicic. Industrial image processing using fuzzy-logic. *Procedia Engineering*, 100:492–498, 2015.
- [2] armKEIL. MDK Microcontroller Development Kit. Dostopno na: <http://www2.keil.com/mdk5/>. Dostopano: 16.9.2018.
- [3] J. Chacon, J. Saenz, L. d. l. Torre, J. M. Diaz, and F. Esquembre. Design of a Low-Cost Air Levitation System for Teaching Control Engineering. *Sensors*, 17(10):2321, 2017.
- [4] D. Fister, R. Šafaric, and I. Fister. Nastavljanje parametrov regulatorja z optimizacijskimi algoritmi. In *StuCoSReC: Proceedings of the 2015 2nd Student Computer Science Research Conference*, pages 13–17,

2015.

- [5] A. M. Gavgani, A. Sornioti, J. Doherty, and C. Cavallino. Optimal gearshift control for a novel hybrid electric drivetrain. *Mechanism and Machine Theory*, 105:352 – 368, 2016.
- [6] I. Iancu. A mamdani type fuzzy logic controller. In *Fuzzy Logic-Controls, Concepts, Theories and Applications*. InTech, 2012.
- [7] A. Ingimundarson and T. Hägglund. Robust tuning procedures of dead-time compensating controllers. *Control Engineering Practice*, 9(11):1195–1208, 2001.
- [8] K. Ngorex. Slideshare. Dostopno na: <https://www.slideshare.net/poerslide/lecture7-ml-machines-that-can-learn>. Dostopano: 7.8.2018.
- [9] O. Kraa, A. Aboubou, M. Becherif, M. Ayad, R. Saadi, M. Bahri, and H. Ghodbane. Fuzzy logic maximum structure and state feedback control strategies of the electrical car. *Energy Procedia*, 50:178 – 185, 2014. Technologies and Materials for Renewable Energy, Environment and Sustainability (TMREES14 – EUMISD).
- [10] R. Li, Y. Guo, S. K. Nguang, and Y. Chen. Takagi-Sugeno fuzzy model identification for turbofan aero-engines with guaranteed stability. *Chinese Journal of Aeronautics*, 31(6):1206–1214, June 2018.
- [11] K. Mehran. Takagi-sugeno fuzzy modeling for process control. *Industrial Automation, Robotics and Artificial Intelligence (EEE8005)*, 262, 2008.
- [12] N. Rai and B. Rai. Control of fuzzy logic based PV-battery hybrid system for stand-alone DC applications. *Journal of Electrical Systems and Information Technology*, 2018.
- [13] R. Šafarič and A. Rojko. *Inteligentne regulacijske tehnike v mehatroniki*. Fakulteta za elektrotehniko, računalništvo in informatiko, 2007.
- [14] A. Sakr, A. M. El-Nagar, M. El-Bardini, and M. Sharaf. Fuzzy smith predictor for networked control systems. In *Computer Engineering & Systems (ICCES), 2016 11th International Conference on*, pages 437–443. IEEE, 2016.
- [15] ST. Description of STM32F4 HAL and LL drivers. Dostopno na: https://www.st.com/resource/en/user_manual/dm00105879.pdf. Dostopano: 16.9.2018.
- [16] ST. STM32CubeMX. Dostopno na: <https://www.st.com/en/development-tools/stm32cubemx.html#sw-tools-scroll>. Dostopano: 16.9.2018.
- [17] Tutorialspoint. Fuzzy Logic Applications. Dostopno na: https://www.tutorialspoint.com/fuzzy_logic/fuzzy_logic_applications.htm. Dostopano: 7.8.2018.

University of Primorska Press
www.hippocampus.si
ISBN 978-961-7055-26-9