Samosprožilno vodenje z nelinearnim modelom na osnovi globokega učenja

Sebastjan Vogrinčič[1]
***Mentor:* doc. dr. Andrej Sarjaš**[1]
[1] **Fakulteta za elektrotehniko, računalništvo in informatiko, Laboratorij za obdelavo signalov in daljinskega vodenja, Koroška cesta 46, 2000, Maribor, Slovenija**
sebastjan.vogrincic@student.um.si, andrej.sarjas@um.si

Abstract

The article presents the problem of modeling nonlinear dynamic systems, using deep learning, and implementation of self-triggered control on an air levitation system. It is a control in which, while the model itself determines the current update of the controller, there is no need for constant sampling of the microcontroller. This is especially interesting for cases where the system operates via network communication, where we don't want to overload the network too often. In the context of modeling, it is necessary to prepare data, and thus to implement communication between the microcontroller, where we capture measurements, and the computer, where we process real data. This is followed by the phase of deep learning and model validation, and then the problem of self-triggered control, where we update the controller according to the model's prediction. The goal of self-triggered control is primarily to reduce the sampling rate and thereby save resources, such as the network in the case of network control, energy consumption, etc. To implement such control, it is necessary to obtain a good open-loop or closed-loop model of the system. The accuracy of this depends mainly on, firstly, the chosen deep learning method and its structure, and secondly, the quality of training and test data.

Kratek pregled prispevka

V članku je predstavljen problem modeliranja nelinearnih dinamičnih sistemov, z uporabo globokega učenja, in implementacija samosprožilnega dogodkovnega vodenja na sistemu zračne levitacije. Gre za vodenje, pri katerem, medtem ko določa sam model trenutke posodabljanja regulatorja, ni potrebnega stalnega vzorčenja mikrokrmilnika. To je še posebej zanimivo za primere delovanja sistema preko omrežne komunikacije, kjer ne želimo obremenjevati omrežja prepogosto. V okviru modeliranja, je potrebna priprava podatkov, in s tem implementacija komunikacije med mikrokrmilnikom, kjer zajemamo meritve, in računalnikom, kjer procesiramo realne podatke. Sledi faza globokega učenja in validacija modela ter zatem še problem samosprožilnega vodenja, kjer posodabljamo regulator glede na predikcijo modela. Cilj samosprožilnega vodenja je predvsem zmanjšanje frekvence vzorčenja in s tem tudi varčevanje z viri, kot je omrežje v primeru mrežnega vodenja, poraba energije itn. Za implementacijo takega vodenja je potrebno pridobiti dober odprto-zančni ali zaprto-zančni model sistema. Natančnost tega je odvisna predvsem od, prvič, izbrane metode globokega učenja in njene strukture, in drugič, kvalitete učnih in testnih podatkov.

1 Uvod

Za spopadanje s pogostimi težavami, ki jih povzroča periodično vzorčenje pri posodobitvi zaprto-zančnega sistema, v sodobnem načinu vodenja, kot je mrežno vodenje, sta bili uvedeni dve metodi posodobitve, imenovani dogodkovno in samosprožilno dogodkovno proženje regulatorja [1]-[4]. Dogodkovno proženje obsega spremljanje izhoda sistema skozi celotni čas vodenja, in posodobitev krmilnega signala le, ko je zaznan nek dogodek. Pri samosprožilnem vodenju, pa gre za napoved dogodka na podlagi modela sistema. Metode, ki temeljijo na dogodkih, sprejemajo odločitve ob zaznavi dogodka in jih tako lahko kategoriziramo, kot reaktivne metode. Nasprotno, samosprožilne metode veljajo za proaktivne, saj napovedujejo čas pojava naslednjega dogodka vnaprej [5].

Cilj obravnavanega vodenja je predvsem zmanjšanje frekvence vzorčenja in s tem tudi varčevanje z viri, kot je uporaba omrežja v primeru mrežnega vodenja ali zmanjšanje porabe virov na sistemu vodenja. Osnovni cilj članka je pridobitev dobrega odprto-zančnega ali zaprto-zančnega modela nelinearnega dinamičnega sistema, kot je sistem zračne levitacije, z namenom testiranja delovanja samosprožilnega vodenja omenjenega sistema.

2 Modeliranje nelinearnih dinamičnih sistemov

Sistemi v realnem svetu so na splošno nelinearni in se obnašajo na različnih ravneh tako v prostoru, kot v času. Prav tako je treba

predpostaviti, da obstaja negotovost v enačbah gibanja, v specifikaciji parametrov in v meritvah sistema [6].

Navedimo primer dinamičnega sistema modeliranega s končnim številom sklopljenih navadnih diferencialnih enačb prvega reda,

$$\begin{aligned} \dot{x}_1 &= f_1(t, x_1, \dots, x_n, u_1, \dots, u_m) \\ \dot{x}_2 &= f_1(t, x_1, \dots, x_n, u_1, \dots, u_m) \\ &\vdots \qquad \vdots \\ \dot{x}_n &= f_n(t, x_1, \dots, x_n, u_1, \dots, u_m), \end{aligned} \tag{1}$$

kjer $\dot{x}_i$ označuje odvod x_i spremenljivke, u_1, u_2,…, u_m so vhodi, x_1, x_2,…, x_n so spremenljivke stanj in f je poljubna gladka funkcija, ki izpolnjuje Lipschitz-ov pogoj [2],[6]. Izraz (1) lahko zapišemo, kot n-dimenzionalno diferencialno enačbo prvega reda,

$$\dot{\boldsymbol{x}} = f(t, \boldsymbol{x}, \boldsymbol{u}). \tag{2}$$

Izraz (2) imenujemo enačba stanja, kjer x označuje stanje, u pa vhod v sistem [7]. Takšna regresijska oblika je primerna za strojno učenje, kjer dinamični model gradimo na osnovi podatkov in meritev sistema. Globoko učenje, natančneje nevronske mreže z regresijsko strukturo, so zmožne učenja in posnemanje kompleksne nelinearne dinamike [6].

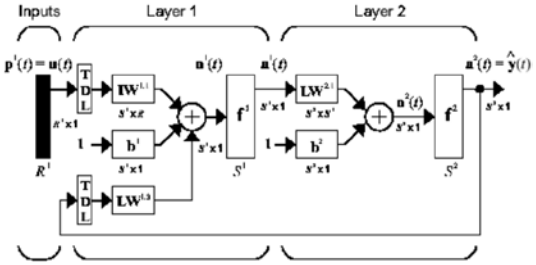
2.1 NARX nevronska mreža

Nelinearna avtoregresivna nevronska mreža z eksogenimi vhodi (angl. *NARX*) je rekurzivna dinamična mreža, s povratnimi povezavami, ki zajema več plasti. Model *NARX* temelji na linearnem modelu *ARX*, ki se običajno uporablja pri modeliranju linearnih časovnih vrst [9]. *NARX* model je definiran kot,

$$y(t) = f\begin{pmatrix} y(t-1), y(t-2), \dots, y(t-n_y), \\ u(t-1), u(t-2), \dots, u(t-n_u) \end{pmatrix}, \tag{3}$$

kjer je naslednja vrednost izhodnega signala *y(t)* regresirana na prejšnje vrednosti izhodnega signala in prejšnje vrednosti neodvisnega (eksogenega) vhodnega signala [7],[8].

Slika 1 prikazuje dvoslojno usmerjeno (angl. Feedforward) NARX nevronsko mrežo. Ta izvedba omogoča tudi vektorski model ARX, kjer sta lahko vhod in izhod večdimenzionalna [8].

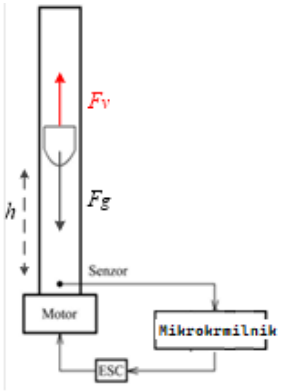


Slika 1: Primer NARX nevronske mreže.

Te nevronske mreže se uporabljajo lahko, kot napovedovalec, za predvidevanje naslednje vrednosti vhodnega signala. Uporablja se lahko tudi za nelinearno filtriranje, pri katerem je ciljni izhod različica vhodnega signala brez šuma [4]. V članku je opisana uporaba NARX mreže za modeliranje nelinearnih dinamičnih sistemov.

3 Sistem zračne levitacije

Pri sistemu zračne levitacije je namen regulacija višine plovca v cevi s pomočjo krmiljenja pretoka zraka.



Slika 2: Sistem zračne levitacije z vsemi gradniki.

Slika 2 shematsko prikazuje tak sistem, kjer elemente sistema vodenja sestavljajo: ToF (angl. Time of flight, čas naleta) senzor razdalje, motor z vgrajenim ventilatorjem, ESC (angl. Electronic speed control, električni krmilnik hitrosti) in krmilnik za vodenje motorja ter senzorja razdalje. Za naš sistem smo uporabili VL53L0X senzor razdalje z dosegom 2m in resolucijo 1mm ter trifazni sinhronski motor s trajnimi magneti, katerega vodimo s pomočjo ZTW A series 12A ESC enoto. Vodenje te poteka s pomočjo PŠM

(pulzno-širinska modulacija) signala, ki ga generira mikrokrmilnik Nucleo-F767ZZI.

3.1 Matematični model sistema

Iz drugega Newtonovega zakona velja,

$$\begin{aligned} m\ddot{h} &= F_v - F_g \\ &= F_v - mg, \end{aligned} \tag{4}$$

kjer je F_v , F_g – sila vzgona in gravitacije [N], m – masa plovca [g], g – gravitacijska konstanta [m/s^2] in $\ddot{h}$ pospešek plovca [m/s^2]. Silo na plovec izračunamo iz enačbe,

$$F_v = \frac{1}{2} C_d \rho A \left(v_\omega - \dot{h}\right)^2, \tag{5}$$

kjer je F_v – sila vzgona [N], Cd – koeficient upora, ρ – gostota zraka [kg/m^3], A – presek plovca [m^2], v_ω – hitrost zraka v cevi [m/s], $\dot{h}$ - hitrost plovca v cevi [m/s]. Matematični model sistema je glede na izraza (4) in (5) definiran kot,

$$m\ddot{h} = \frac{1}{2} \cdot C_d \rho A \left(v_\omega - \dot{h}\right)^2 - mg \tag{6}$$

Model (6) poenostavimo tako, da predpostavimo, da je koeficient upora konstanten, zaradi nizke hitrosti plovca in nespremenljivih razmer vzdolž cevi. Definirajmo konstanto,

$$\beta = \frac{C_d \cdot \rho \cdot A}{2}. \tag{7}$$

V nadaljevanju zgornji izraz (7) vstavimo v prejšnjo enačbo (6) in dobimo,

$$\dot{h} = v \tag{8}$$

$$\dot{v} = \frac{\beta}{m} \cdot \left(v_\omega - \dot{h}\right)^2 - g \tag{9}$$

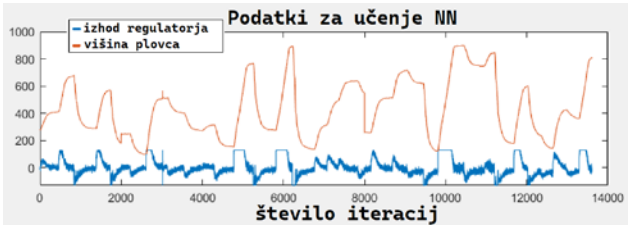
Matematični model nam je v pomoč pri načrtovanju regulatorja, kot je PID, ali nelinearnih regulatorjev kot so npr. zaprto-zančna linearizacija, sestopni (angl. Backstepping) regulator, ali regulator drsnega režima.

4 Priprava podatkov nevronske mreže za učenje dinamike

Prvi izziv je generiranje nabora podatkov. Za ta namen vzpostavimo komunikacijo med mikrokrmilnikom, kjer zajemamo izhod sistema h s senzorjem in računamo vhod sistema tj. izhod regulatorja. Zajemanje podatkov v intervalu 25 ms in procesiranje je bilo izvedeno v programskem okolju MATLAB.

Pridobitev dobrega modela je močno odvisna od kvalitete učnih in testnih podatkov. Pri čemer moramo upoštevati naslednje:

- primerna izbira nabora podatkov, ki opisujejo dinamiko v različnih delovnih točkah,
- filtriranje podatkov,
- normiranje podatkov,
- odstranitev morebitnih odstopajočih vrednosti (angl. Outliers, ubežniki),
- ustrezna razdelitev izmerjenih, in ustrezno obdelanih, podatkov na učne in testne skupine.



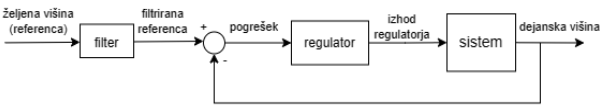
Slika 3: Primer učnega podatka.

Z upoštevanjem prve točke za zagotovitev primernih podatkov učenja in Slike 3, lahko ugotovimo, da bo potrebno predprocesiranje in dodatna obdelava zajetih vrednosti. Izhod regulatorja smo filtrirati z nizkoprepustnim FIR filtrom (angl. Finite impulse response, končni impulzni odziv) Parametre tega smo določili po t.i. 'trial in error' metodi, kjer smo želeli odstraniti neželene visoko frekvenčen komponente signala v obliki šuma in ohraniti prvotno dinamiko primarnega signala. Po filtriranju smo se odločili za t.i. 'detrend' podatkov. Gre za odstranitev učinkov trenda iz nabora podatkov, da se prikažejo samo razlike v vrednostih iz trenda. Trend se običajno nanaša na spremembo srednje vrednosti skozi čas. Ta metoda še vedno ohranja dinamiko in korelacijo podatkov [9]. Vse zgoraj omenjene predpriprave učnih podatkov prispevajo k uspešnejšemu učenju in k natančnejšemu modelu nevronske mreže. Zadnji korak pred učenjem je potrebno podatke razdeliti v učno in validacijsko skupino. Učni podatki zajemajo 70% nabora vseh podatkov, pri čemer je potrebno upoštevati, da obe skupini opisujeta

dinamiko v vseh delovnih točkah, saj le tako lahko zagotovimo uspešnost validacije učenja.

5 Učenje nevronske mreže (NN)

Samosprožilno vodenje sloni na modelu sistema, zato smo tudi sprva začeli z učenjem nevronske mreže. Ko smo pričeli z učenjem nevronske mreže, smo ugotovili, da bodo rezultati zagotovo boljši pri učenju zaprtozančnega modela, saj je v tem primeru korelacija med vhodom in izhodom NN veliko večja in ta model je primeren za izbran način vodenja.



Slika 4: Regulacijska zanka sistema zračne levitacije (brez NN).

5.1 Izbira strukture nevronske mreže in metode učenja

Večje število nevronov in časovnih zakasnitev prispeva k modeliranju kompleksnejšim problemov, manjše pa k kompaktnejšim rešitvam. Iz tega razloga smo težili k manjšemu številu nevronov. Za določitev optimalnih vrednosti smo uporabili več vgnezdenih zank, z namenom preiskati večje število kombinacij nevronov in plasti ter vsako od teh ponoviti večkrat (zaradi narave učenja nevronskih mrež, tj. pridobivanje drugačnih rezultatov ob vsaki ponovitvi). Za kriterij kvalitete rezultatov smo uporabili koeficient determinacije R^2 [6],[10].

Kot drugi kriterij smo uporabili avtokorelacijo pogreška. Avtokorelacija pogreška prikazuje, kako pogrešek NN pri danem trenutku korelira s samim seboj ob različnih časovnih zamikih. Ta graf mora doseči vrh pri zamiku 0 in hitro upadati pri večjih zamikih. Če ne, je to dober znak, da je verjetno potrebnih več nevronov ali število časovnih zakasnitev. Vrednost avtokorelacije pri zamiku 0 je enaka srednjemu kvadratnemu pogrešku (MSE, angl. Mean Squared Error) naučene mreže. Križna korelacija pogreška pa prikazuje kako pogrešek NN pri danem trenutku korelira z vhodom NN ob različnih časovnih zamikih. Vrednosti tega grafa naj ne bi prestopili meje zaupanja [6]. Glede na omenjene kriterije smo se odločili za 4 korake časovne zakasnitve tako vhoda kot izhoda, in število nevronov med 7 in 14, odvisno od velikosti učnih podatkov.

Kot metodo učenja smo sprva uporabili metodo Levenberg-Marquadt, saj je ta najhitrejša. Problem pri tej metodi je potencialno osciliranje izhoda naučene nevronske mreže. To lahko odpravimo z Bajesevo metodo učenja (ang. Bayesian regularization).

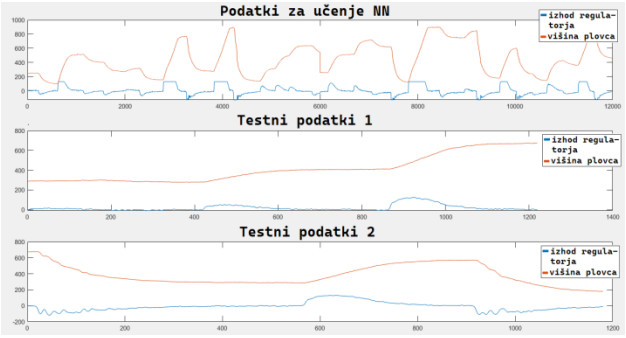
6 Rezultati

Na naslednjih slikah so prikazani najboljši rezultati vsakega postopka reševanja in implementacije problema, ki smo ga opisali. Realni sitem prikazuje slika 5.

Slika 5: Sistem zračne levitacije

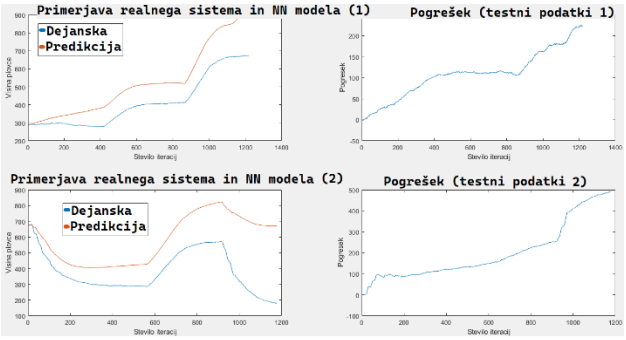
Slika 6 prikazuje učne in testne podatke, ki so filtrirani z izločenimi ubežniki, in nad katerimi je bila opravljena odstranitev trenda. Podatki filtra, ki smo ga uporabili so: frekvenca vzorčenja F_s = 1/60 Hz, N = 3, spodnja meja F_{pass} = 3/20000 Hz, zgornja meja F_{stop} = 1/500 Hz, valovitost R_p = 0.5, zaporno ojačanje A_{stop} = 50.

Slika 7 prikazuje rezultate odpro-zančnega modela z uporabljenimi podatki iz slike 6. Vidimo, da nevronska mreža ni preveč natančna. Predvidevamo lahko, da je to zaradi slabe razdelbe nabora podatkov, ki znaša 3:1 (učni/testni). Slika 8 prikazuje predikcijo nevronske mreže, naučene z razdelitvijo 4:1. Rezultati so boljši, ampak poskusimo za primerjavo še naučiti zaprtozančni sistem, prikazan na sliki 10.

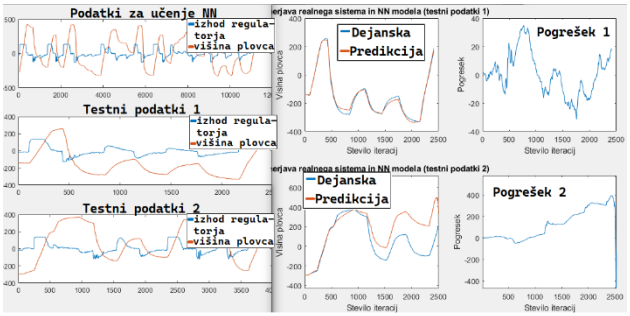


Slika 6: Procesirani podatki.

Rezultati učenja so prikazani na naslednjih slikah.

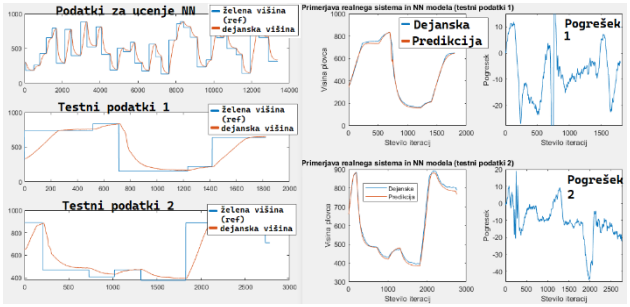


Slika 7: Prvi rezultati učenja.

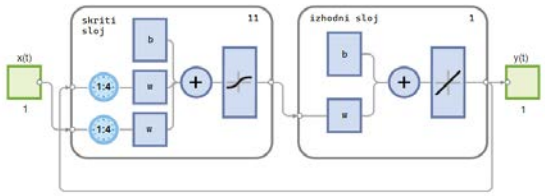


Slika 8: Rezultati po boljši razporeditvi podatkov.

Rezultati iz slike 9 prikazujejo najboljši model, tj. zaprtozančni model. Iz predikcije na grafu na desni strani slike je razvidno, da je za najboljše delovanje samosprožilnega vodenja potrebno uporabiti podatke zaprtozančnega sistema. Uporabljeno strukturo najboljše naučene nevronske mreže prikazuje slika 10.



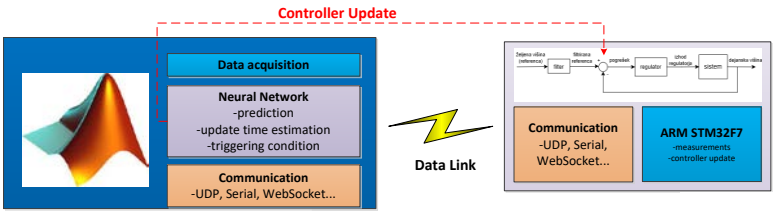
Slika 9: Rezultati zaprtozančnega modela.



Slika 10: NN struktura zaprto-zančnega modela.

6.1 Samosprožilno dogodkovno vodenje

Kot smo že prej omenili, smo se za implementacijo samosprožilnega vodenja odločili za uporabo zaprtozančnega modela. Model mora biti dovolj dober, da lahko z visoko natančnostjo napovemo višino plovca za časovni interval naprej. Struktura vodenja je prikazana na sliki 11.



Slika 11: Samosprožilno vodenje

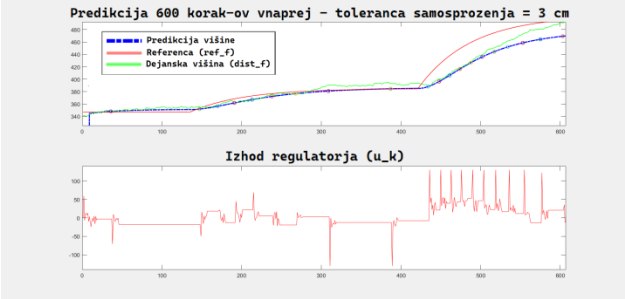
Za zaprto-zančno vodenje je uporabljen klasični PID regulator, katerega v članku nismo obravnavali posebej. PID regulator zagotavlja stabilno vodenje pri času vzorčenja 25 ms. Samosprožilno vodenje deluje na naslednji način:

- ob zagonu sistema zajema meritve za namen priprave predikcije nevronske mreže več korakov vnaprej,
- izvede predikcijo n-korakov vnaprej,
- izračuna časovni interval posodobitve, ko se posodobi regulator,
- regulator se posodobi,

- izklopi tako zajemanje meritev, kot tudi posodabljanje regulatorja, regulator drži zadnjo posodobljeno vrednost,
- po preteku časovnega intervala v slednjem trenutku zajame meritve ter posodobi regulator
- pripravi naslednjo predikcijo in ponovi postopek.

Omenjen časovni trenutek se izračuna tako, da najprej določimo toleranco odstopanja. Gre za razliko odstopanja izhoda v trenutku od zadnje posodobitve regulatorja. Algoritem iz predikcije več korakov vnaprej izračuna trenutek, ko izhod zaprtozančnega modela preseže dano toleranco.

Slika 12 prikazuje rezultate samosprožilnega vodenja, kjer se prvih 8 iteracij posodablja regulator in zajema podatke. V začetni fazi se pripravijo pogoji za predikcijo 600 korakov vnaprej. Krogci označujejo časovne trenutke, ko je, glede na zaprtozančni model (toleranca 3 cm), potrebno posodobiti regulator. Od zadnje posodobitve naprej, regulator posodobimo in zajemamo naslednjih 8 meritev, nato držimo zadnjo vrednost regulatorja in izračunamo naslednji trenutek posodobitve.



Slika 12: Rezultati samosprožilnega dogodkovnega vodenja vetrovnika.

Dejansko višino plovca smo zajemali ves čas, a zgolj zaradi prikaza rezultatov v članku. Delovanje takega vodenja je precej odvisno od vrednosti izhoda regulatorja v trenutku, ko tega ne posodabljamo. Zato smo tudi pustili, da se regulator pri zajemanju 8 meritev posodablja, čeprav v tem primeru priprava podatkov ni potrebna.

7 Sklep

Rezultati ponazorjeni v članku so dober prikaz postopka načrtovanja in izdelave modela sistema za namen izvedbe samosprožilnega vodenja, kot tudi implementacije takšnega vodenja. S člankom smo predvsem dokazali delovanje obravnavanega vodenja na hitrem nelinearnem in nestabilnem sistemu. Seveda pa še to ni optimalna rešitev. Iz tega sledi predlog izboljšave, tj. optimizacija vodenja glede na zasnovan regulator, kakor tudi na horizont predikcije. V nadaljevanju se lahko bolj podrobno lotimo analitične izpeljave predlaganega vodenje ter tako posledično določimo ključne parameter, ki najbolj vplivajo na stabilnost in zanesljivost sistema.

Zanesljivost takega vodenja je predvsem odvisna od natančnosti modela. Samosprožilno vodenje je lahko tvegano, zato bi npr. za industrijsko aplikacijo, bilo potrebno vpeljati dodatne varnostne mehanizme, kot je npr. občasno preverjanje delovanja modela ipd.. V nadaljevanju predlagamo tudi testiranje delovanja samosprožilnega vodenja preko različnih komunikacijskih kanalov in analizo oz. validacijo uspešnosti nadomestitve klasičnih načinov vodenja z obravnavanim.

8 Literatura

[1] Franklin G.,F.; Powell J.,D.; Workman M., L. Digital Control of Dynamic Systems, 3rd edition, Ellis-Kagle Press, 2006.

[2] Shtessel, Y.; Edwards, C.; Fridman, L.; Levant, A. Sliding Mode Control and Observation. Birkhäuser, 2014.

[3] Behera, A., K.; Bandyopadhyay, B. Self-triggering based sliding-mode control for linear systems, *IET Control Theory & Applications*, 2015 9(17), 2541–2547.

[4] Incremona, P.,G; Ferrara, A. Adaptive model-based event-triggered sliding mode control, *International Journal of Adaptive Control and Signal Processing*, 2016, 30, 1298–1316

[5] Tiberi, U., Johansson, K. *On the robustness of self-triggered sampling of nonlinear control systems.* arXiv preprint arXiv: 1510.00564, 2015.

[6] Brunton, Steven L., Kutz, J. Nathan. *Data-driven science and engineering : machine learning, dynamical systems, and control*, 6. Cambridge: Cambridge University Press, 2020.

[7] Khalil, H. K. *Nonlinear control*, 1. Pearson, 2015

[8] *Design Time Series NARX Feedback Neural Networks.* Dostopno na: Design Time Series NARX Feedback Neural Networks - MATLAB & Simulink (mathworks.com)° (9.3.2023).

[9] Kenton, W. *Detrend.* Dostopno na: Detrend Definition (investopedia.com) (9.3.2023).

[10] *Maglev Modeling with Neural Time Series App.* Dostopno na: Maglev Modeling with Neural Time Series App - Video - MATLAB (mathworks.com) (9.3.2023).