

Učenje in vgrajena izvedba nevronskega regulatorja za stabilizacijo dvokolesnega robota z odpravljanjem motenj

Jon Tofant¹, Andrej Sarjaš¹

¹University of Maribor, Faculty of Electrical Engineering and Computer Science
Koroška cesta 46, 2000 Maribor, Slovenia
E-pošta: jon.tofant@um.si

Learning and Embedded Implementation of a Neural Controller for Disturbance Rejection in a Two-Wheeled Self-Balancing Robot

Traditional control design approaches for two-wheeled self-balancing robots typically use linearized PID or LQR design for high-frequency, disturbance-resistant control. We train a neural controller in Isaac Lab with model-free reinforcement learning using Proximal Policy Optimization in the absence of a stabilizing controller, with a five-stage curriculum where pushes and offset loads are gradually added, and domain randomization for sim-to-real transfer. The obtained compact network with 1410 parameters is converted to ONNX format and implemented in a low-cost STM32 microcontroller. The controller was tested on Isaac Lab, MuJoCo and the physical robot with tilt disturbances from 5–15°. In all scenarios, the controller was able to stabilize the robot and show real-time disturbance rejection on limited hardware.

1 Uvod

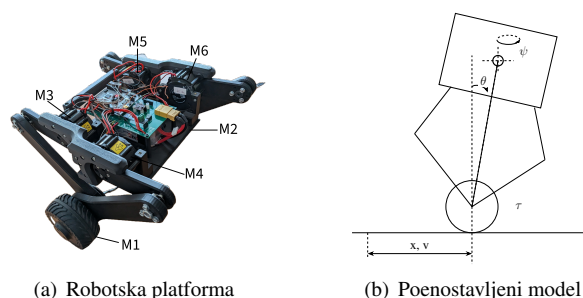
Dvokolesni robot za učinkovito odpravljanje zunanjih motenj v spreminjajočem se okolju potrebuje robusten regulacijski algoritem s kratkim časom vzorčenja. Pregled razvoja dvokolesnih robotov in pripadajočih pristopov vodenja je podan v delu [1]. Klasične metode vodenja za sintezo regulatorja uporabljajo linearni ali nelinearni model v prostoru stanj, ki je pogosto natančneje opisan v bližini delovne točke [2]. Avtorji [3] so pokazali, da je optimalni regulator stanj (LQR), ki upošteva poln vektor stanj, učinkovitejši od večznančne kaskadne strukture linearnih regulatorjev. Raziskave na področju kompleksnih dinamičnih gibov se ukvarjajo s skoki, seskoki, vožnjo po enem kolesu, delovanjem v razgibanem okolju in prevajanjem različnih bremen [4]. Avtorji [5, 6] so uporabljali metode učenja za ohranjanje stabilnosti robota, pri čemer so za učenje potrebovali začetni regulator, ki že stabilizira robota. Prispevka članka sta uporaba spodbujevanega učenja za stabilizacijo in vožnjo brez matematičnega modela ter izvedba kompaktne nevronske mreže na nizkocenovnem mikrokrmilniku. Cilj je mreža z majhnim številom uteži in plasti, ki upošteva celoten vektor stanj in omogoča predpisane gibe. Izbrana mreža ima 1410 učljivih parametrov. Prenos na realni sistem omogočimo z naključnim spreminjanjem fizikalnih parametrov znotraj znanega

območja med učenjem.

2 Robotska platforma

2.1 Mehanska in elektronska zasnova

Robot na sliki 1(a) je zgrajen kot dvokolesna platforma z dvema pogonskima kolesoma DDSM115 s tokovnim vodenjem (M1, M2). Položaj in hitrost koles beremo z vgrajenih enkoderjev vsakih 15 ms. Trup robota se giba s pomočjo štirih motorjev Xiaomi CyberGear (M3–M6). Motorji CyberGear imajo stalne referenčne kote 0 rad; PD-regulatorji z $K_P = 20,0$ in $K_D = 0,9$ vzdržujejo lego in blažijo gibanje sklepov ob motnjah. Inercialna merilna enota BNO08x je postavljena na sredini platforme robota. Naklon določamo iz vektorja težnosti, kotne hitrosti pa iz kalibriranega žiroskopa (oba pri 100 Hz); odklon določamo iz kvaterniona pri 20 Hz. Platforma uporablja vgrajeni mikrokrmilnik STM32F446RE. Pogoni se napajajo neposredno iz baterije LiPo 6S, mikrokrmilnik pa prek napetostnega regulatorja. V modelu MuJoCo znašajo polmer koles 0,050 m, razdalja med središčema koles 0,330 m in skupna masa 6,90 kg. Sklepi M3–M6 so v tem modelu togi, zato njihova podajnost ni zajeta.



Slika 1: Robotska platforma in njen poenostavljeni dinamični model.

2.2 Merjene veličine

Na poenostavljenem modelu na sliki 1(b) je x relativni položaj robota vzdolž podlage, v njegova linearna hitrost, θ naklon telesa (angl. pitch), $\dot{\theta}$ kotna hitrost naklona, ψ odklon okoli navpične osi (angl. yaw), $\dot{\psi}$ kotna hitrost odklona in τ navor pogonskega kolesa. Nevronskega regulatorja je v regulacijskem koraku k na voljo opazovalni

vektor

$$\mathbf{o}_k = [x \ v \ \theta \ \dot{\theta} \ \psi_{\text{pogrešek}} \ \dot{\psi} \ I_{L,k-1} \ I_{R,k-1}]^T. \quad (1)$$

Indeksa L in R označujeta levo in desno kolo. Položaj in hitrost se izračunata kot $x = \frac{R}{2}(\varphi_L + \varphi_R)$ in $v = \frac{R}{2}(\dot{\varphi}_L + \dot{\varphi}_R)$, kjer je R polmer kolesa, φ_L in φ_R sta kotni legi koles, $\dot{\varphi}_L$ in $\dot{\varphi}_R$ pa njuni kotni hitrosti. Odklon se iz kvaterniona $\mathbf{q} = [q_w, q_x, q_y, q_z]^T$ izračuna kot $\psi = \text{atan2}(2(q_w q_z + q_x q_y), 1 - 2(q_y^2 + q_z^2))$. Pogrešek odklona $\psi_{\text{pogrešek}} = \psi - \psi_{\text{ref}}$, kjer je ψ_{ref} referenčni odklon, se omeji na interval $[-\pi, \pi]$. Naklon in kotna hitrost naklona sta izmerjena z inercialno merilno enoto, $I_{L,k-1}$ in $I_{R,k-1}$ pa sta tokovna ukaza (referenčni vrednosti tokov) levega in desnega kolesa iz prejšnjega regulacijskega koraka. Vse veličine so dostopne na robotu.

3 Predlagani nevronski regulator

Nevronski regulator vsebuje 1410 učljivih parametrov. Pri uporabi 32-bitnih števil s plavajočo vejico uteži zasedejo približno 5,5 kB pomnilnika, kar je bistveno manj od razpoložljivega pomnilnika mikrokrmilnika STM32F446RE, ki vsebuje 512 kB pomnilnika Flash in 128 kB pomnilnika SRAM.

3.1 Struktura regulatorja

Nevronski regulator na sliki 2 ima na vходу merljiva stanja sistema ter prejšnji vrednosti tokov obeh koles. Vhod v mrežo je normalizirani opazovalni vektor $\bar{\mathbf{o}}_k$. Mreža ima dva polno povezana skrita sloja z 32 nevroni in aktivacijsko funkcijo ReLU. Na izhodu sta dva nevrona, katerih vrednosti se s funkcijo tanh skalirajo med -2 A in 2 A . Izhoda neposredno predstavljata referenčna tokova posameznih koles.

Opazovalni vektor $\mathbf{o}_k$ iz enačbe 1 normaliziramo z deljenjem posameznih komponent s konstantami κ_i in omejitvijo rezultata:

$$\bar{o}_{k,i} = \min \left(10, \max \left(-10, \frac{o_{k,i}}{\kappa_i} \right) \right), \quad i = 1, \dots, 8.$$

Delitelji v vrstnem redu komponent opazovalnega vektorja so

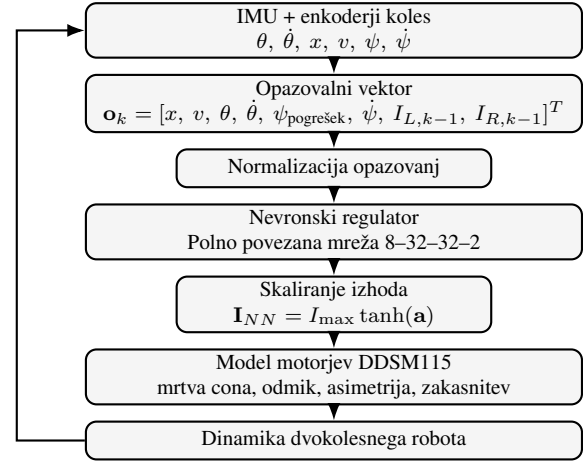
$$\kappa = [1 \text{ m}, 1 \text{ m s}^{-1}, 25^\circ, 4 \text{ rad s}^{-1}, 180^\circ, 4 \text{ rad s}^{-1}, 2 \text{ A}, 2 \text{ A}]^T.$$

Kotni veličini in njuna delitelja so pri izračunu izraženi v radianih. Vrednosti NaN pred omejitvijo nadomestimo z nič. Pri učenju in vgrajeni izvedbi uporabljamo iste konstante in enako omejitve.

3.2 Opazovalni in akcijski prostor

Regulator prejme opazovalni vektor $\mathbf{o}_k$, definiran z enačbo 1. Njegov izhod je akcijski vektor tokovnih ukazov levega in desnega kolesa v trenutnem regulacijskem koraku:

$$\mathbf{u}_k = [I_{L,k} \ I_{R,k}]^T. \quad (2)$$



Slika 2: Struktura nevronskega regulatorja: iz normaliziranih opazovanj mreža določi tokovna ukaza pogonskih koles.

3.3 Omejitev toka

Surovi izhod nevronske mreže je vektor $\mathbf{a} = [a_L, a_R]^T$, kjer sta a_L in a_R surova izhoda za levo in desno kolo. Tokovna ukaza dobimo z omejitvijo in skaliranjem tega vektorja:

$$\mathbf{u}_k = \mathbf{I}_{NN} = I_{\max} \tanh(\mathbf{a}), \quad (3)$$

pri čemer se funkcija tanh uporabi po komponentah, $I_{\max} = 2 \text{ A}$ pa je največji dovoljeni tok pogonov.

4 Učenje regulatorja v simulaciji

4.1 Simulacijsko okolje

Učenje izvajamo v simulacijskem okolju Isaac Lab (NVIDIA Isaac Sim) z algoritmom PPO. Fizikalni korak simulacije je 1 ms, regulacijski korak pa 20 ms, kar ustreza frekvenci izvajanja prekinitvene rutine regulatorja na mikrokrmilniku. Epizoda traja 8 s (400 regulacijskih korakov) in se konča predčasno, če naklon telesa v petih zaporednih korakih preseže 25° . Prav tako končamo učenje, če relativni odmik platforme preseže 1 m. Učenje poteka v petih zaporednih stopnjah z naraščajočo težavnostjo (angl. curriculum). Stopnje so zapisane v tabeli 1.

PPO (RSL-RL) uporablja 4096 vzporednih okolij, 64 zbranih korakov na okolje v iteraciji ter 4 učne prehode na iteracijo, vsakega s 4 manjšimi učnimi paketi. Začetna stopnja učenja je 5×10^{-4} s prilagajanjem glede na ciljno divergenco KL 0,01. Diskontni faktor je $\gamma = 0,99$, parameter GAE $\lambda = 0,95$, parameter omejevanja razmerja verjetnosti 0,2, utež izgubne funkcije kritika 1,0, utež entropijskega člana 0,002, največja dovoljena norma gradienta 0,5 in začetni standardni odklon šuma akcij 0,2.

4.2 Nagradna funkcija

$$\begin{aligned} r = & 1 - w_\theta \theta^2 - w_\omega \dot{\theta}^2 \\ & - w_v v^2 - w_x x^2 - w_{\psi_p} \psi_{\text{pogrešek}}^2 \\ & - w_{\dot{\psi}} \dot{\psi}^2 - w_I (I_L^2 + I_R^2) \\ & - w_{\Delta I} (\Delta I_L^2 + \Delta I_R^2) \end{aligned} \quad (4)$$

Pokončna lega ne zahteva ničelnega položaja x ali pogreška odklona $\psi_{\text{pogrešek}}$. Nagradna funkcija kaznuje naklon, kotno hitrost naklona, pogrešek odklona in hitrost baze

Tabela 1: Petstopenjska učna shema za čisto nevronske regulacije. Za stopnjo 1 velja $\Delta\theta_0 = \pm 3^\circ$, $\Delta\dot{\theta}_0 = \pm 0,2 \text{ rad s}^{-1}$, $\Delta v_0 = \pm 0,05 \text{ m s}^{-1}$; za stopnje 2–5 pa $\Delta\theta_0 = \pm 8^\circ$, $\Delta\dot{\theta}_0 = \pm 0,8 \text{ rad s}^{-1}$, $\Delta v_0 = \pm 0,15 \text{ m s}^{-1}$.

Stopnja	Iteracije	Motnje
1	100	brez
2	100	brez (samo večja inicializacija)
3	150	enojni/dvojni sunek
4	150	tovor / premik masnega središča
5	200	kombinirane motnje

Tabela 2: Verjetnosti motenj po stopnjah učenja v %; – pomeni ničelno verjetnost.

Motnja	Stopnji 1–2	3	4	5
brez	100	50	40	30
enojni sunek	–	40	–	–
dvojni sunek	–	10	–	–
sunek	–	–	–	30
tovor / premik masnega središča	–	–	40	25
tovor + sunek	–	–	20	15

v prostoru. Robota kaznujemo tudi za preveliko porabo in sunkovito spremembo toka. Spremenljivka w_x je v tabeli 3 nastavljena na 0, saj želimo hitrostno vodenje namesto položajnega, kljub temu pa lahko podatek o položaju pomaga regulatorju med učenjem.

4.3 Motnje med učenjem

V simulacijskem okolju med učenjem injiciramo motnje. Motnje so fizikalno utemeljene. Potisk z roko simuliramo s silo, ki na telo deluje kot kratkotrajni sunek. Dolgotrajni navor na telo simulira objekt, ki je postavljen na platformo zunaj masnega središča. Motnje se uvajajo postopoma od tretje stopnje učenja dalje, kot prikazuje tabela 1. Amplituda sunka znaša $F_x \in -6 - -6 \text{ N}$, $F_y \in -1,5 - -1,5 \text{ N}$ in $M_z \in -0,15 - -0,15 \text{ N m}$, amplituda tovara oziroma premika težišča pa $M_x \in -0,25 - -0,25 \text{ N m}$ (tabela 2).

4.4 Naključno spreminjanje parametrov

Razlike v fizikalnem modelu motorja, trenju med kolesom in tlemi ter masi vplivajo na uspešnost prenosa naučenega algoritma iz simulacije na vgrajeni sistem. Med učenjem upoštevamo napake pri modeliranju in znotraj določenega območja spreminjamo parametre sistema. Razponi so podani v tabeli 4. Znotraj tega območja želimo zajeti realne

Tabela 3: Utežni koeficienti nagradne funkcije

Člen	Simbol	Vrednost
Nagrada za preživetje	–	1,0
Kazen za naklon	w_θ	10,0
Kazen za kotno hitrost naklona	w_ω	0,4
Kazen za hitrost	w_v	0,15
Kazen za pogrešek odklona	w_{ψ_p}	0,05
Kazen za kotno hitrost odklona	$w_{\dot{\psi}}$	0,05
Kazen za tok	w_I	0,003
Kazen za spremembo toka	$w_{\Delta I}$	0,03
Kazen za položaj	w_x	0

Tabela 4: Naključno spreminjanje parametrov za prenos iz simulacije na realni sistem

Parameter	Razpon
Motorno ojačenje L/R (neodvisno)	(0,8; 1,2)
Mrtva cona motorja	0,03–0,20 A
Tokovni odmik	–0,08–0,08 A
Časovna konstanta tokovne zanke	0,005–0,020 s
Tokovna omejitev	1,6–2,4 A
Odmik naklona	$\pm 0,5^\circ$
Šum naklona	$\sigma = 0,15^\circ$
Šum kotne hitrosti naklona	$\sigma = 0,02 \text{ rad s}^{-1}$
Šum hitrosti	$\sigma = 0,01 \text{ m s}^{-1}$
Zakasnitev opazovanja	0 ali 1 vzorec
Trenje kolesnega sklepa	(0,007; 0,013)
Viskozno dušenje kolesa	0,006–0,014 N m s rad ^{–1}
Statično trenje tal	(0,5; 1,1)
Dinamično trenje tal	(0,4; 0,9)

parametre brez potrebe po natančni analizi in meritvah.

5 Vgrajena izvedba

5.1 Pretvorba modela

Z orodjema Isaac Lab in PyTorch učimo model, pri čemer uteži in odmike shranimo v formatu PyTorch. Model pretvorimo v format ONNX in preverimo skladnost. Model vgradimo v mikrokrmilnik STM32 z uporabo orodja STM32Cube.AI. Okolje generira knjižnice naučene nevronske mreže v jeziku C. Orodje STM32Cube.AI optimizira aktivacijske funkcije, zato je potrebno ponovno preverjanje skladnosti.

5.2 Izvajanje na STM32

Znotraj enega koraka, ki traja 20 ms, najprej preberemo vrednosti inercialne merilne enote in žiroskopa ter trenutno hitrost in kot koles robotske platforme. Sestavimo osemdimenzionalni opazovalni vektor, ki ga normaliziramo s skalirnimi faktorji. Izhod nevronske mreže skaliramo in uporabimo za vodenje motorja DDSM115. Izmerjeni čas izračuna nevronske mreže na mikrokrmilniku znaša 0,09 ms.

6 Rezultati

6.1 Eksperimentalni protokol

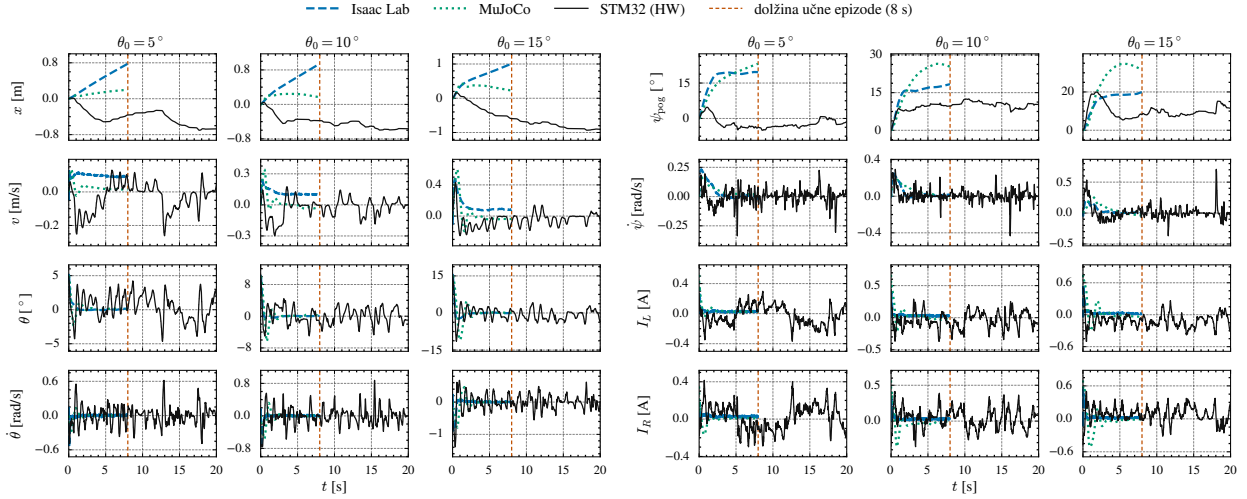
Za vrednotenje regulatorja uporabljamo naslednje metrike: največji absolutni naklon $\max |\theta|$, efektivno (RMS) vrednost naklona, efektivno vrednost tokovnih ukazov levega in desnega motorja (RMS I_L , RMS I_R), srednjo in efektivno vrednost preostalega naklona proti koncu testa ter končni položaj x_{kon} .

6.2 Rezultati simulacije in vgrajene izvedbe

Regulator smo preizkusili v učnem okolju Isaac Lab, neodvisnem simulatorju MuJoCo (prenos sim2sim) in na realnem robotu z mikrokrmilnikom STM32 (prenos sim2real) pri začetnih naklonih 5° , 10° in 15° . Na realnem robotu smo začetni naklon nastavili ročno in ob začetku merjenja robota spustili brez dodajanja sile.

Tabela 5: Odzivi pri različnih začetnih naklonih v oknu 0–8 s od spusta ($t = 0, x = 0$); x_{kon} je položaj pri 8 s. Tokovna ukaza sta referenčni vrednosti tokov, ne izmerjena tokova.

Scenarij	Platforma	$\max \theta [^\circ]$	$\text{RMS } \theta [^\circ]$	$\bar{\theta}_{\text{res}} [^\circ]$	$\theta_{\text{res,RMS}} [^\circ]$	$x_{\text{kon}} [\text{m}]$	$\text{RMS } I_L [\text{A}]$	$\text{RMS } I_R [\text{A}]$
5°	Isaac Lab	5,13	0,68	0,07	0,09	0,778	0,038	0,033
5°	MuJoCo	5,00	0,95	0,15	0,15	0,200	0,068	0,049
5°	STM32 (HW)	4,86	1,78	1,23	1,81	-0,362	0,118	0,132
10°	Isaac Lab	10,12	1,34	0,06	0,07	0,943	0,051	0,043
10°	MuJoCo	10,00	2,25	0,19	0,23	0,168	0,118	0,111
10°	STM32 (HW)	10,44	2,44	-0,73	1,90	-0,373	0,131	0,140
15°	Isaac Lab	15,45	2,31	-0,02	0,06	1,006	0,075	0,064
15°	MuJoCo	15,00	4,16	0,19	0,20	0,226	0,183	0,199
15°	STM32 (HW)	15,42	3,43	-0,76	1,69	-0,598	0,165	0,165



Slika 3: Odzivi v okoljih Isaac Lab in MuJoCo ter na robotu (STM32) pri začetnih naklonih 5°, 10° in 15°. Levo: položaj, hitrost, naklon in kotna hitrost naklona; desno: pogrešek odklona, kotna hitrost odklona ter tokovna ukaza motorjev. Simulaciji sta prikazani do 8 s, robot do 20 s; črtkana črta označuje dolžino učne epizode (8 s).

Preostali naklon ovrednotimo v oknu 6–8 s s srednjo in efektivno vrednostjo:

$$\bar{\theta}_{\text{res}} = \frac{1}{N} \sum_{i=1}^N \theta_i, \quad \theta_{\text{res,RMS}} = \sqrt{\frac{1}{N} \sum_{i=1}^N \theta_i^2}. \quad (5)$$

Rezultati so prikazani v tabeli 5 in na sliki 3. Regulator stabilizira naklon platforme pri vseh treh začetnih naklonih. Na realnem robotu se ravnotežje ohranja tudi po koncu časovnega obsega učne epizode, do 20 s, pri čemer ostaja prisotno nihanje naklona. Položaj robota se med meritvijo spreminja, zato stabilizacija naklona ne pomeni tudi zaustavitve. Pogrešek odklona se v prikazanem časovnem oknu ne odpravi v celoti. Med simulacijama in realnim sistemom so razlike v poteku položaja, pogrešku odklona in preostalem naklonu. Simulacijska odziva sta prikazana do 8 s, zato je daljše delovanje prikazano le za realni sistem.

7 Zaključek

Predstavili smo kompakten nevronskega regulator, naučen s spodbujevanim učenjem, in njegovo vgradnjo v mikro-krmilnik STM32. Robustnost smo povečali s senzorskim šumom, motnjami in naključnim spreminjanjem mrtve cone koles, trenja tal, tokovne omejitve in dušenja koles med učenjem. Regulator stabilizira naklon v simulacijah

in na realnem robotu. Nadaljnje delo vključuje optimizacijo nagradne funkcije in arhitekture mreže.

Literatura

- [1] X. Liu *et al.*, “Development of wheel-legged biped robots: A review,” *Journal of Bionic Engineering*, vol. 21, pp. 607–634, Mar. 2024.
- [2] N. Mao, J. Chen, E. Spyarakos-Papastavridis, and J. S. Dai, “Dynamic modeling of wheeled biped robot and controller design for reducing chassis tilt angle,” *Robotica*, vol. 42, pp. 2713–2741, Aug. 2024.
- [3] J. Yang, Q. Wu, S. Li, Y. Ye, and C. Luo, “Integrated modeling and control optimization of biped wheel-legged robot,” *IEEE Robotics and Automation Letters*, vol. 10, pp. 1465–1472, Feb. 2025.
- [4] G. Pyo, J. H. Hwang, Y. Ko, J. W. Lee, and Y.-J. Park, “JuM-BRo: A model predictive control based two-wheeled self-balancing robot capable of jumping over obstacles,” *Control Engineering Practice*, vol. 166, p. 106620, Jan. 2026.
- [5] L. Cui, S. Wang, J. Zhang, D. Zhang, J. Lai, Y. Zheng, Z. Zhang, and Z.-P. Jiang, “Learning-based balance control of wheel-legged robots,” *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7667–7674, 2021.
- [6] R. Hernández, R. García-Hernández, and F. Jurado, “Stabilization of a two-wheeled self-balancing robot using reinforcement learning,” *Proceedings of the 26th Robotics Mexican Congress, COMRob 2024*, pp. 117–122, 2024.