https://doi.org/10.70750/GEP.2025.2.2

MEGERŐSÍTÉSES TANULÁSON ALAPULÓ ÁGENSEK TELJESÍTMÉNYÉNEK JAVÍTÁSA TAPASZTALAT ALAPÚ KONTEXTUS BEVEZETÉSÉVEL

IMPROVING THE PERFORMANCE OF REINFORCEMENT LEARNING AGENTS USING EXPEPERIENCE-BASED CONTEXT

Farkas Péter[*], *Szőke László*[**], *Dr. Aradi Szilárd*[***], *Dr. Gyurkó Zoltán*[****]

ÖSSZEFOGLALÁS

Ezen publikációban egy tapasztalat-alapú online adaptációs keretrendszert mutatunk be megerősítéses tanuláson alapuló ágensek számára, mely lehetővé teszi, hogy a múltbéli állapot-akció átmenetek felhasználásával képesek legyenek alkalmazkodni változó körülményekhez, nehezen hozzáférhető információk felhasználása nélkül. Megoldásunk működését egy széles dinamikai tartományt lefedő robotmodell irányítási problémáján keresztül értékeljük ki.

ABSTRACT

This paper proposes an experience-based online adaptation framework for reinforcement learning agents, enabling them to adjust to changing conditions by leveraging past state-action transitions, improving their performance in dynamic environments without relying on hard-to-obtain information. The performance of our solution is evaluated through the control problem of a robot model covering a wide dynamic range.

1. BEVEZETÉS

A magasan automatizált rendszerek elterjedése megköveteli a gépi tanuláson alapuló technikák irányítási folyamatokban történő integrálását, különösen a nagy komplexitású rendszerek esetén. Az ilyen algoritmusok közül a megerősítéses tanulás (reinforcement learning, RL) az egyik legígéretesebb megoldás, mely kiemelkedő teljesítményt mutat az autonóm járművek [1][2], a drónok [3], és a robotika területén [4].

Míg a mai fejlett irányító algoritmusok hatékonyan működnek ideális helyzetekben, teljesítményük jelentősen csökkenhet változó körülmények között [5]. Például a modern, magasan automatizált járművek jól teljesítenek optimális vezetési körülmények között, azonban sokszor kihívásokba ütköznek, amikor olyan problémákkal kell szembenézniük, mint a rossz időjárás vagy a megváltozott útviszonyok [6].

A hagyományos irányítási rendszereket gyakran egy adott felhasználási módhoz vagy járműhöz tervezik, melynek eredményeként az ágensek nem rendelkeznek adaptációs képességgel. Ezzel szemben az emberek képesek alkalmazkodni az ideálistól eltérő körülményekhez, így például egy rövid adaptációs folyamat után egy könnyű személygépkocsi és egy teljesen megrakott furgon irányításával is boldogulnak. Ebből az összehasonlításból kiindulva előnyös, ha az irányító ágenseket hasonló adaptációs képességekkel látjuk el, ezzel növelve megbízhatóságukat és robusztusságukat, mely eredményeként sokoldalúbbá és hatékonyabbá válhatnak megváltozott körülmények között is.

1.1. A publikáció célja

A fent ismertetett probléma orvoslására publikációnk egy online adaptációs keretrendszert javasol, mely segíti az ágensek különböző körülményekhez történő alkalmazkodását azáltal, hogy a korábbi állapot-akció átmenetekből kinyert információkat kiegészítő bemenetként használja fel. Ez lehetővé teszi az algoritmus számára, hogy felismerje a változatosokat az irányított rendszerben, és alkalmazkodjon hozzájuk, ezzel fenntartva az optimális teljesítményt. Megoldásunk egy zárt hurkot képez a modellmentes policy-alapú algoritmus számára, melyen keresztül az ágens láthatja, hogy a múltbeli átmenetek során a választott akciói

[*] *PhD hallgató, peterfarkas@edu.bme.hu*
[**] *PhD hallgató, szoke.laszlo@kjk.bme.hu*
[***] *egyetemi docens, aradi.szilard@kjk.bme.hu*
[****] *R&D csoportvezető, zoltan.gyurko@knorr-bremse.com*

milyen állapotokhoz vezettek. Tesztjeink során a javasolt keretrendszer jelentősen javította az ágens teljesítményét és alkalmazkodóképességét.

2. A MEGERŐSÍTÉSES TANULÁS

A gépi tanulás ezen ága lehetővé teszi az ágensek számára, hogy a környezetükkel történő interakciók és az így szerzett tapasztalataik alapján kialakítsanak egy optimális döntéshozatali stratégiát. Ennek köszönhetően a megerősítéses tanulás széles körben használt keretrendszerré vált a dinamikus környezetek irányításában [7][8].

A megerősítéses tanulás alapjait a Markov-döntési folyamat adja, melyet a következő négy elemű vektorral írhatunk le:

$$M = <S, A, R, P_{S,A}> \quad (1)$$

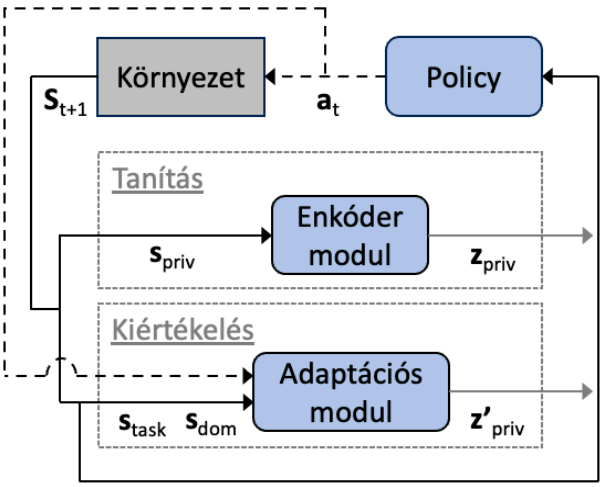
Minden egyes t diszkrét időlépésben az ágens megkapja a környezet $s_t \in S$ reprezentációját és választ egy $a_t \in A$ akciót az aktuális $\pi(a_t|s_t)$ policy alapján. Ezután a környezet $P(s_{t+1}|s_t, a_t)$ valószínűséggel átlép a következő $s_{t+1} \in S$ állapotba, majd az ágens $r_{t+1} \in R$ jutalomban részesül, amely a választott a_t akció minőségét jelzi. Az ágens célja tanulás közben, hogy az epizódok interakciói során a kumulatív várható jutalmat maximalizálja.

A megerősítéses tanulás algoritmusai közül ezen munkában a Proximal Policy Optimization (PPO) metódus került felhasználásra, mely napjaink legkorszerűbb policy-alapú megoldásai közé tartozik irányítási feladatok esetén [9].

3. A JAVASOLT MEGOLDÁS

Ahogy az az 1. fejezetben már hangsúlyozásra került, a megerősítéses tanuláson alapuló ágensek adaptációs képességekkel történő felruházása növelné az ilyen algoritmusok valós körülmények között történő használhatóságát. A tanulási folyamat során modellmentes ágensek megtanulják a környezet átmeneti függvényét, azonban amikor a rendszer viselkedése megváltozik, a döntéshozatali stratégia teljesítménye romolhat, vagy bizonyos eloszláson kívüli minták esetén hibásan működhet.

Ezen kihívások megoldása érdekében módszerünk először kinyeri a múltbeli állapot-akció átmenetekbe ágyazott információkat, majd bevezeti azokat az ágensek kiegészítő bemeneteként. Az ily módon hozzáadott visszacsatolási ág lehetővé teszi az algoritmus számára, hogy észlelje a környezet eltéréseit, és ezt követően működés közben úgy igazítsa döntéshozatali startégiáját, hogy robusztusan tudja kezelni a megváltozott körülményeket, ezzel fenntartva az optimális teljesítményét. A javasolt architektúrát az 1. ábra szemlélteti.



1. ábra: A javasolt architektúra

3.1. Az állapotreprezentáció

A tanult viselkedés és a környezet átmeneti függvényének elválasztása egyszerűsíthető az állapottér elemeinek 3 kategóriába történő csoportosításával: a feladathoz kapcsolódó állapotváltozók ($\boldsymbol{s}_{task}$), a domain-specifikus állapotok ($\boldsymbol{s}_{dom}$), valamint a privilegizált információk ($\boldsymbol{s}_{priv}$). Az $\boldsymbol{s}_{task}$ tartalmaz minden olyan információt, amely az ágens által megoldandó konkrét feladatot írja le, például egy sávtartás esetén az útfelfestések reprezentációját. Az $\boldsymbol{s}_{dom}$ olyan, a szabályozott rendszerrel vagy a környezettel kapcsolatos adatokat ír le, melyek az irányítási folyamat során alapból rendelkezésre állnak (pl. meglévő szenzorok adatai). Ezzel ellentétben az $\boldsymbol{s}_{priv}$ olyan információkból áll, melyek működés nehezen vagy egyáltalán nem hozzáférhetőek, azonban segítenék az ágenst a változó körülményekhez történő alkalmazkodásban. Ilyen többletinformáció lehet a súrlódási együttható, az aktuátorok belső állapota vagy az irányított rendszer fizikai paraméterei.

Azt, hogy egy adott állapotváltozó a domain-specifikus vagy a privilegizált kategóriába tartozik-e, mindig az adott alkalmazás határozza meg. A kategorizálást a rendelkezésre álló szenzorok, az irányított rendszer fizikai paramétereinek ismerete, valamint a környezet állapotának azonosíthatósága befolyásolja.

3.2. Az állapot enkóder modul

A keretrendszer továbbá tartalmaz egy enkóder modult (Enc_s) is, mely a bemenetére kapcsolt $\boldsymbol{s}_{priv}$ privilegizált információkból egy $\boldsymbol{z}_{priv}$ látens reprezentációt hoz létre. Ennek a modulnak a célja, hogy gazdagítsa állapotvektor információtartalmát, valamint, hogy egyszerűsítse a következőkben bemutatott adaptációs modul állapotbecslési feladatát azáltal, hogy eliminálja a pontos fizikai paraméterbecslés szükségességét.

3.3. Az adaptációs modul

A valós működés során, alapvetően az s_{priv} állapotváltozók és ennek megfelelően a z_{priv} látens komponensek nem, vagy csak nehezen hozzáférhetőek, ezért az adaptációs modul fő feladata a múltbeli állapot-akció átmenetekbe ágyazott információkat felhasználva a hiányzó információk előállítása.

Ezen komponens egy Long Short-Term Memory (LSTM) hálózatként van implementálva, amelyet lineáris rétegek követnek. A modul megkapja múltbeli s_{task} és s_{dom} állapotokat, valamint az a_t akciókat, melyek alapján megbecsüli a hiányzó z'_{priv} látens állapotkomponenseket. Mivel a tanítás szimulációban történik, a privilegizált állapotelemek rendelkezésre állnak, és így felügyelt tanítás keretein belül felhasználhatóak az adaptációs modul tanításához az Átlagos Abszolút Hiba által leírt veszteség függvényen keresztül:

$$L^{SL}{}_{\theta} = ÁAH(z_{priv}, z'_{priv}) \qquad (2)$$

Működés során a nem elérhető z_{priv} látens állapotok helyettesíthetőek az előzményalapú adaptációs modul z'_{priv} kimenetével. Ezáltal a javasolt megoldás lehetővé teszi, hogy az ágens alkalmazkodjon különböző működési környezetekhez vagy akár valós szcenáriókhoz is, nehezen hozzáférhető információk felhasználása nélkül.

3.4. A tanítási és kiértékelési fázisok

A bemutatott keretrendszerben a tanítási és kiértékelési fázisok szétválasztásra kerülnek.

Tanítás során a policy a feladattal kapcsolatos (s_{task}) és a domain-specifikus (s_{dom}) állapotok, valamint a privilegizált információ (s_{priv}) látens reprezentációja (z_{priv}) alapján frissül. Mivel a tanítás szimulációban történik, a z_{priv} rendelkezésre áll. Az epizódok között a szabályozott rendszer dinamikája megváltozik, ezzel biztosítva a gazdag állapotátmeneteket. A policy és az Enc_s hálók az eredeti PPO veszteségen keresztül frissülnek, míg az adaptációs modul optimalizálása az 2-es egyenletben megadott felügyelt metrikával történik.

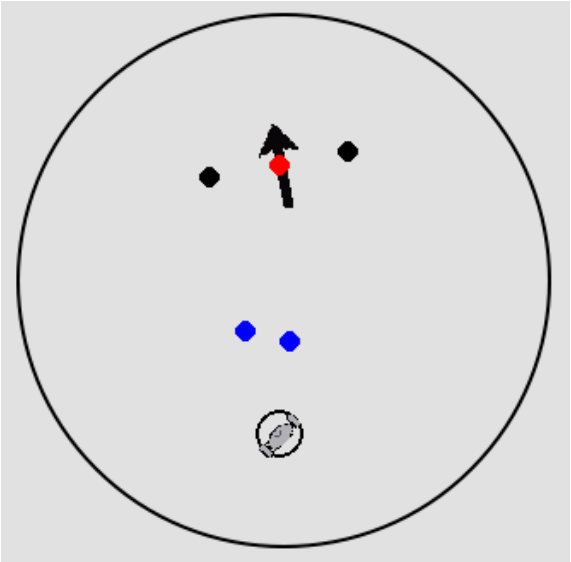
Ezzel szemben kiértékelés/működés során az s_{priv} és így a z_{priv} információ nem érhető el, azonban az adaptációs modul az s_{task} és s_{dom} állapotokból, valamint az a_t akciókból álló múltbéli trajektória alapján képes a z'_{priv} látens változók becslésére. Ezáltal az adaptációs modul elő tudja állítani a hiányzó állapotokat, így segítve az ágenst a környezet eltérésének felismerésében. Ez a folyamat biztosítja, hogy az ágens valós időben tudjon alkalmazkodni a megváltozott körülményekhez, finomhangolás vagy a célkörnyezetben történő további tanítás nélkül.

4. A KIÉRTÉKELÉSI KERETRENDSZER

Módszerünk működését egy széles dinamikai tartományt lefedő robotmodell segítségével, egy összetett irányítási feladaton keresztül. mutatjuk be. A modell dinamikai paraméterei változtathatóak, ami diverz szcenáriókhoz, valamint adaptív szabályozási stratégiák szükségességéhez vezet. Az általunk javasolt keretrendszerben betanított ágens performanciáját a környezetről különböző információkhoz hozzáférő hagyományos RL alapú algoritmusokkal hasonlítjuk össze. A kiértékelés során kétféle beállítást vizsgáltunk: fizikai paramétereken alapuló adaptáció és mozgásállapot-becslés.

4.1. Az RL környezet

Munkánk során az ágensnek egy differenciálhajtású robot ütközésmentes irányítását kell megvalósítania. A robot működését Lagrange egyenletek írják le, melynek elméleti hátterét [10] mutatja be. Az így implementált dinamikai modell továbbá a motorok karakterisztikáját leíró egyenletekkel is kiegészül, így a rendszert közvetlenül feszültségeken keresztül lehet irányítani nyomatékok helyett. A Lagrange-egyenletek állandói a robot mozgását nagy mértékben befolyásoló fizikai paraméterek, melyek a különböző epizódok között változtathatóak. Az ilyen jellemzők közé tartozik például a robot súlya, inerciája, szélessége, kerekeinek átmérője vagy a motorok ellenállása. Ezen paraméterek együttes változtatásával olyan diverz szcenáriók hozhatóak létre, melyek adaptációs képességet igényelnek, ezzel kialakítva egy komplex irányítási problémát. A 2. ábra az implementált szimulációs környezet egy pillanatképét mutatja.



2. ábra: A szimulációs környezet

4.2. Állapottér, akciótér és jutalomfüggvény

A robot a két motor bemeneti feszültségének változtatásával vezérelhető. Ennek megfelelően az ágens egy 2 dimenziós, [-1, 1] közé eső folytonos akciótérben hozhat döntéseket. A környezetben ezen kiválasztott bemeneti feszültségek ±50 Volt közé vannak skálázva, hogy az összes lehetséges motorkonfiguráció vezérlésére lehetőség nyíljon.

Ahogy az az előzőekben már bemutatásra került, az ágens döntéseit magalapozó állapottér elemei különböző kategóriákba csoportosíthatóak. Az adott feladattal kapcsolatos $\boldsymbol{s}_{task}$ információk tartalmazzák a célpont és az akadályok relatív helyzetét. Ezen komponensek ily módon történt definiálása elősegíti a valós körülmények között történő alkalmazhatóságot. Az ágens számára továbbá elérhetőek a robot fizikai jellemzői és a mozgásállapotát leíró változók is. Az $\boldsymbol{s}_{dom}$ és $\boldsymbol{s}_{priv}$ pontos tartalma azonban az adaptációs modul beállításától függ, és a későbbiekben kerül bemutatásra.

Az epizódok során az ágens minden lépés után jutalomban (r_{step}) részesül, mely segíti a robot célpontba történő eljuttatását a komplex dinamikával definiált szcenáriókban:

$$r_{step} = \Delta\, d_{goal} / d_{max} \qquad (3)$$

ahol $\Delta\ d_{goal}$ a célpont robottól vett távolságának változását jelöli két lépés között, d_{max} pedig a kettőjük maximális távolságát jelenti. A jutalomfüggvényt továbbá egy $r_{success}$ taggal is kiegészül, mely az epizód végső pozíció- és szöghiba csökkentésére szolgál:

$$r_{success} = 1 - min(e_{pos} + e_{head}\ ,\ 1) \qquad (4)$$

Itt az e_{pos} és e_{head} rendre a robot epizód végén vett pozíció és szöghiba eltérésének normalizált értékét jelöli. Végezetül a sikertelen epizódok során az ágens -1 jutalomban részesül (tereptárgyakkal való ütközés, túl nagy megtett távolság, a kijelölt terület elhagyása, túl hosszú epizód).

4.3. Adaptációs beállítások

A javasolt keretrendszer működését kétféle adaptációs mechanizmuson keresztül mutatjuk be, melyeket lehetséges valós alkalmazási módok inspiráltak.

Az első felállásban olyan eseteket szimulálunk, ahol a szabályozott rendszer pontos fizikai tulajdonságainak azonosítása kihívást jelenthet (pl. az áruk pontos súlya és mérete egy anyagmozgató robot esetében). Itt az adaptációs modul a robot fizikai paramétereit becsüli meg ($\boldsymbol{z}'_{priv}$), míg feltételezzük, hogy a mozgásállapota különböző érzékelők segítésével megállapítható és így az ágens rendelkezésére bocsátható ($\boldsymbol{s}_{dom}$).

A második kategóriába tartozó szcenáriókat olyan esetek inspirálták, ahol a mozgásállapotok szenzorok hiányában nem mérhetőek. Itt azonban azt feltételezzük, hogy a robot dinamikai tulajdonságai ismertek. Ilyen helyzetekben az adaptációs modul becsüli meg a mozgásállapotot leíró vektor elemeit ($\boldsymbol{z}'_{priv}$), míg a fizikai paraméterek $\boldsymbol{s}_{dom}$ részeként érhetőek el.

4.4. Kiértékelt ágensek

Kiértékelésünk során a javasolt keretrendszer performanciáját 3 másik ágensével hasonlítjuk össze. Módszerünkhöz hasonlóan az összehasonlítás többi résztvevőjének alapjait is az RLLib PPO implementációja [11] adja.

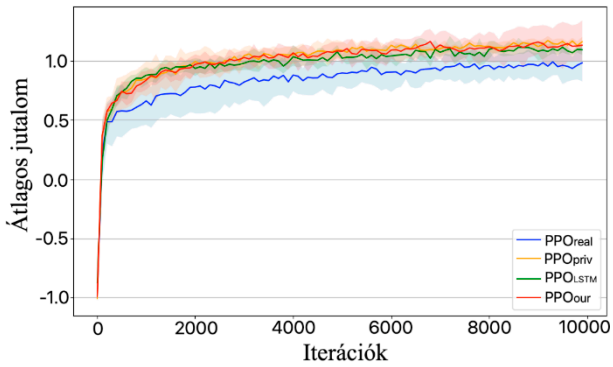
A PPO_{real} egy olyan megvalósítható ágenst szimbolizál, amely csak az alapból is rendelkezésre álló információk felhasználásával kerül betanításra és felhasználásra (azaz nem kapja meg az $\boldsymbol{s}_{priv}$ adatokat egyetlen ponton sem). Ezzel szemben a PPO_{priv} és PPO_{LSTM} a privilegizált ágenseket jelöli, mivel ezek mindig hozzáférnek a $\boldsymbol{s}_{priv}$ kiegészítő információkhoz. Fontos azonban kiemelni, hogy a valós felhasználási körülmények között ezen extra információk nem érhetőek el. A PPO_{priv} megkapja az összes állapotot, míg a PPO_{LSTM} az ezen ágens LSTM hálóval kibővített változatát jelöli, amely az általunk javasolt megoldás időbeli reprezentációs képességét figyelembevéve biztosít fair összehasonlítást. Végül a PPO_{our} pedig a bemutatott keretrendszerrel tanított és kiértékelt ágenst azonosítja. A tanítás során ugyanazt az információt kapja, mint a PPO_{priv} és a PPO_{LSTM}, kiértékelés közben azonban az $\boldsymbol{s}_{priv}$ információk nélkül működik, és az adaptációs modult használja a $\boldsymbol{z}'_{priv}$ becslésére a korábbi állapot-akció átmenetek alapján.

4.5. A tanítási folyamat konfigurálása

A reprodukálhatóság érdekében az összes ágens esetén az RLLib alapértelmezett PPO beállításai kerültek felhasználásra. Kivételt képez a PPO_{our}, ahol az alap veszteségfüggvény kiegészül a (2)-es egyenletben leírt hibával. A policy és a value becslő modulokat Tanh aktiválású Multi-Layer Perceptron (MLP) hálók adják, melyek csak bemeneti dimenziójukban különböznek az aktuális állapottér kialakításától függően. Továbbá a PPO_{our} tartalmazza az adaptációs és az enkóder modulokat is: az előbbi egy 2 rétegű MLP, míg az utóbbi egy LSTM háló, melyet előre csatolt rétegek követnek. A tanítási minták diverzifikálása érdekében a robot minden epizódban különböző dinamikai paraméterekkel kerül inicializálásra.

1. táblázat: Az ágensek teljesítménye fizikai paraméter alapú adaptáció esetén

	Siker [%]	Átl. e_{pos} [m]	Átl. e_{head} [°]
PPO_{real}	72	0,1	8,0
PPO_{priv}	80	0,07	5,6
PPO_{LSTM}	80	0,05	3,8
PPO_{our}	89	0,06	5,2



3. ábra: A tanítások alakulása fizikai paraméter alapú adaptáció esetén

5. EREDMÉNYEK

5.1. Fizikai paraméter alapú adaptáció

Az első kísérletsorozat azokat az eseteket vizsgálja, amikor a szabályozott rendszer tényleges fizikai paraméterei nem állnak rendelkezésre. Ilyen helyzetekben az adaptációs modul a robot dinamikai jellemzőit rekonstruálja.

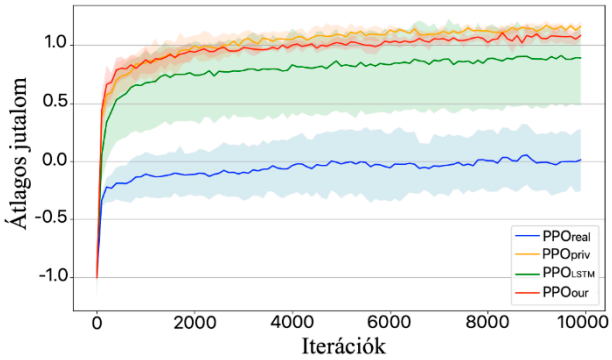
A 3. ábra 50 egymást követő kiértékelési epizód jutalmának alakulását szemléleti az egyes modellek tanítása során. A görbék ágensenként 5 különböző seed átlagolt teljesítményét mutatják, míg a halvány területek ezek szórását jelölik. Amint az várható volt, az információhiány miatt a PPO_{real} érte el a legalacsonyabb átlagos jutalmat. Eközben, ágensünk (PPO_{our}) hiába csak ugyanezeket az állapotokat kapja meg a kiértékelés során, mégis képes a PPO_{priv} és a PPO_{LSTM} ágensek teljesítményére.

Az 1. táblázat az egyes ágensek performanciájának összehasonlítását mutatja be 100 azonos kiértékelési epizód során. Látható, hogy a legjobb összjutalommal rendelkező PPO_{our} tudta a legtöbb epizódot sikeresen teljesíteni, mellyel az extra információhoz hozzájutó ágenseket is felülmúlta. A táblázat tartalmazza továbbá az átlagos pozíció- és szöghibákat is. Ezen metrikák is azt mutatják, hogy az általunk javasolt keretrendszerben tanított ágens képes felülmúlni az azonos információkból dolgozó PPO_{real} ágenst a privilegizált ágensekkel összehasonlítható teljesítményével.

5.2. Mozgásállapot alapú adaptáció

A javasolt módszerünk működésének vizsgálata kiterjed arra az alkalmazásra is, ahol a robot mozgásállapota szenzorok hiányában nem mérhető, viszont a szabályozott rendszer pontos méretei és fizikai paraméterei ismertek.

A 4. ábrán látható, hogy a PPO_{real} nem képes megfelelő teljesítményt nyújtani a mozgásállapotokat leíró változók ismerete nélkül, miközben az általunk javasolt ágens a privilegizált modellek teljesítményéhez közel azonos eredményre jutott. A különböző dinamikai paraméterekkel rendelkező rendszerek irányítása kihívást jelentő feladat az aktuális mozgásállapotuk ismerete nélkül, az adaptációs modul azonban az időbeli reprezentációjának köszönhetően képes a szükséges információk előállítására.



4. ábra: A tanítások alakulása mozgásállapot alapú adaptáció esetén

A 2. táblázat szintén a különböző ágensek teljesítményét hasonlítja össze 100 azonos epizód során. Látható, hogy a mozgásállapotok látens reprezentációjának becslése jelentősen javítja az irányítás minőségét, ezzel megközelítve a PPO_{priv} és a PPO_{LSTM} ágensek teljesítményét. Mindeközben a PPO_{real} csak az epizódok felében járt sikerrel, valamint a mért hibái is jóval elmaradtak a másik 3 ágensétől. Ez tovább igazolja módszerünk működőképességét, mivel megállapítható, hogy a múltbéli átmenetek információtartalmának kinyerésével és felhasználásával akár különböző szenzorok adatai is helyettesíthetőek.

2. táblázat: Az ágensek teljesítménye mozgásállapot alapú adaptáció esetén

	Siker [%]	Átl. e_{pos} [m]	Átl. e_{head} [°]
PPO_{real}	52	0,1	19,6
PPO_{priv}	80	0,07	5,6
PPO_{LSTM}	80	0,05	3,8
PPO_{our}	77	0,06	6,0

6. KONKLÚZIÓ

Összességében elmondható, hogy a javasolt keretrendszer képes javítani a megerősítéses tanuláson alapuló vezérlő ágensek adaptációs képességét. A bevezetett enkóder és adaptációs modulok segítségével olyan nehezen hozzáférhető információkat állíthatunk elő, amelyek egyébként új érzékelők vagy környezeti tényezők hozzáadása nélkül nem állnának rendelkezésre. Az adaptációs modul sikeresen ki tud nyerni olyan, a múltbeli állapot-akció átmenetekbe ágyazott információkat, melyek segítik az ágens döntéshozatalát.

Ez a kiegészítő bemenet lehetővé teszi a javasolt módszerrel tanított és felhasznált ágensek számára, hogy

felismerjék a vezérelt rendszer devianciáit, és ezt követően adaptálják kialakított stratégájukat a megváltozott körülmények robusztus kezelése érdekében. A keretrendszer alkalmazása jelentős javulást eredményez a hagyományos RL-alapú megoldásokhoz képest és a nehezen hozzáférhető információk alapján működő ágensekkel is összehasonlítható teljesítményhez vezet.

7. KÖSZÖNETNYÍLVÁNÍTÁS

A KDP-IKT-2023-900-11-00000957/0000003 számú projekt a Kulturális és Innovációs Minisztérium Nemzeti Kutatási Fejlesztési és Innovációs Alapból nyújtott támogatásával, a KDP-2023 pályázati program finanszírozásában valósult meg.

A kutatás a Bolyai János Kutatási Ösztöndíj támogatásával készült.

8. FELHASZNÁLT IRODALOM

[1] L. Wang et al., „Efficient Reinforcement Learning for Autonomous Driving with Parameterized Skills and Priors," in *Robotics: Science and Systems*, Daegu, R. of Korea, 2023.

[2] Z. Zhang, A. Liniger et al., „End-to-End Urban Driving by Imitating a Reinforcement Learning Coach," in *Proceedings of the IEEE International Conference on Computer Vision*, Montreal, Canada, 2021.

[3] E. Kaufmann, L. Bauersfeld et al., „Champion-level drone racing using deep reinforcement learning," *Nature,* vol. 620, pp. 982-987, 2023.

[4] P. Adolfo et al., „Position/force control of robot manipulators using reinforcement learning," *Industrial Robot: The International Journal of Robotics Research and Application,* vol. 46, pp. 267-280, 2019.

[5] X. B. Peng, M. Andrychowicz et al., „Sim-to-Real Transfer of Robotic Control with Dynamics Randomization," in *IEEE International Conference on Robotics and Automation*, Singapore, 2017.

[6] F.-M. Luo, S. Jiang, Y. Yu et al. „Adapt to Environment Sudden Changes by Learning a Context Sensitive Policy," in *AAAI Conference on Artificial Intelligence*, 2022.

[7] W. Koch, R. Mancuso et al., „Reinforcement learning for UAV attitude control," *ACM Transactions on Cyber-Physical Systems,* vol. 3, no. 2, pp. 1-21, 2019.

[8] J. Baltes et al., „A deep reinforcement learning algorithm to control a two-wheeled scooter with a humanoid robot," *Engineering Applications of Artificial Intelligence, vol.*126, p. 106941 2023.

[9] J. Schulman et al., „Proximal Policy Optimization Algorithms," 2017.
Available: https://arxiv.org/abs/1707.06347.

[10] R. Dhaouadi et al., „Dynamic Modelling of Differential-Drive Mobile Robots using Lagrange and Newton-Euler Methodologies: A Unified Framework," *Advances in Robotics & Automation,* vol. 02, 2013.

[11] Ray, „RLlib: Industry-Grade Reinforcement Learning," [Online].
Available:
https://docs.ray.io/en/latest/rllib/index.html.