

OTDK-dolgozat

Herczeg Ágnes

Harkó Csanád

Hatlábú robotok irányítása oszcilláló neuronokkal

Szerzők:

Herczeg Ágnes

Babeş–Bolyai Tudományegyetem, Fizika Kar, Fizika szak, III. év

Harkó Csanád

Babeş–Bolyai Tudományegyetem, Fizika Kar, Mérnöki fizika szak, IV. év

Témavezető:

dr. Sándor Bulcsú, egyetemi adjunktus

Babeş–Bolyai Tudományegyetem, Fizika Kar



UNIVERSITATEA BABEŞ-BOLYAI
BABEŞ-BOLYAI TUDOMÁNYEGYETEM
BABEŞ-BOLYAI UNIVERSITÄT
BABEŞ-BOLYAI UNIVERSITY
TRADITIO ET EXCELLENTIA

Kivonat

Napjainkban a robotok vezérlésének elmélete rohamosan fejlődik. A klasszikus robotika, valamint a mesterséges intelligenciára épülő módszerek mellett újabb, az önszerveződő mechanizmusokkal foglalkozó irányzat is kezd egyre fontosabbá válni. A hagyományos módszerek esetében a lépkedő robotok helyváltoztatása jól koordinált mozgásokat feltételez, mely során az egyes csuklók közötti fázisviszonyokat a megfelelő paraméterek beállításával pontosan tudjuk szabályozni. Számos eredmény azonban arra utal, hogy a mozgásokhoz szükséges dinamika önszerveződő módon is kialakulhat.

Dolgozatunkban egy PhantomX típusú hatlábú robot egy lehetséges vezérlési modelljét vizsgáljuk. A tizennyolc csuklója közül tizenkettőt osszcilláló neuronokkal irányítunk, melyek direkt módon, valamint a lábak csuklószögeinek mérésén keresztül vannak csatolva. A modellt először egy számítógépen szimulált fizikai környezetbe helyezzük, majd egy valós roboton is teszteljük. A neuronok közötti csatolás erősségének függvényében mindkét esetben többféle lépésmintázat is létrehozható, melyek részben önszerveződő módon alakulnak ki.

XXV. ERDÉLYI TUDOMÁNYOS DIÁKKÖRI KONFERENCIA
CONFERINȚA ȘTIINȚIFICĂ STUDENȚEASCĂ DIN TRANSILVANIA (CȘST), EDIȚIA A XXV-A
25TH TRANSYLVANIAN STUDENTS' SCIENTIFIC CONFERENCE (TSSC)
reál- és humántudományok • științe reale și umane • real and human sciences

KOLOZSVÁR, 2022. MÁJUS 19–22.

Kolozsvári Magyar Diákszövetség • Uniunea Studențească Maghiară din Cluj
400083 Kolozsvár, Petőfi Sándor/Avram Iancu utca 21 • tel./fax: 0755116251
e-mail: etdk@kmdsz.ro, etdk.kolozsvar@gmail.com • honlap/web: <https://etdk.kmdsz.ro>

Témavezetői igazolás

Alulírott *dr. Sándor Bulcsú* igazolom, hogy *Herczeg Ágnes és Harkó Csanád* hallgatók a(z) *Hatlábú robotok irányítása oszcilláló neuronokkal* című dolgozatot az alulírott szakmai irányításával készítették el, és javaslom az említett tudományos munka bemutatását a XXV. Reál- és Humántudományi Erdélyi Tudományos Diákköri Konferencián (ETDK).

dr. Sándor Bulcsú, egyetemi adjunktus
Babeș–Bolyai Tudományegyetem, Fizika Kar

Kolozsvár, 2022. szeptember 30.

Tartalomjegyzék

1. Bevezető	1
1.1. Klasszikus robotika	1
1.1.1. Direkt és inverz kinematika	2
1.2. Robotok irányítása dinamikai rendszerekkel	4
1.2.1. Dinamikai rendszerek	4
2. A hatlábú robot felépítése	6
2.1. A robot teste és lábai	7
2.2. Hatlábú robot irányításának alapjai	7
3. Oszcilláló neuronmodell robotok vezérléséhez	10
3.1. Fázisoszcillátorok	10
3.1.1. Taszító kuramoto-modell	11
3.2. Oszcilláló neuron visszacsatolással	11
3.2.1. Neurális modellek dióhéjban	12
3.2.2. Homeosztázis	14
3.2.3. Oszcilláló neuron	14
3.3. Túlcillapított mozgás vezérlése	15
3.3.1. A rendszer dinamikája	15
3.3.2. Fixpontok, bifurkáció	18
3.4. Csillapított rugó-tömb rendszer vezérlése	19
3.4.1. A rendszer dinamikája	20
3.4.2. Fixpontok, bifurkáció	21
3.4.3. Rezonancia	23
3.5. Vezérlés csatolt neuronokkal	24
3.5.1. A rendszer dinamikája	25
3.6. Hatlábú robot numerikus szimulációja	26
4. Kísérleti eredmények	29
4.1. Hatlábú robot irányítása inverz kinematikával	30
4.2. Vezérlés fázisoszcillátorokkal	31
4.3. Vezérlés csatolt neuronokkal	33

Következtetések és továbbfejlesztési lehetőségek	38
A. Függelék	40
A.1. Bővebben a PhantomX - ről	40
A.1.1. A robot vezérlése, programozása	40
A.1.2. A LiPo akkumulátor	41
A.1.3. Az okos szervomotorok	41
A.2. A robot numerikus szimulációjának részletezése	45

1. Bevezető

Dolgozatunkban dinamikai rendszerekkel vezérlünk egy hatlábú robotot. Célunk már létező modellek tanulmányozása és reprodukálása, majd egy saját neuronmodell felépítése. Kezdetben alapfogalmakat tisztázunk, rövid robotikai alapfogalmakat vezetünk be, majd a hatlábú robot modellekhez szükséges paramétereit írjuk le. Ezt követően tárgyaljuk a már létező klasszikus és komplex elméleti modelleket, melyekkel hatlábú robotokat irányítottak [4, 5, 3]. A későbbiekben ezeket alkalmazzuk a hatlábú roboton és tanulmányozzuk a váltakozó háromlábú lépésmintázatot. A saját neuronmodellt lépésenként építjük fel, mely már létező elméletek továbbfejlesztése. [6, 11, 7]. Megvizsgáljuk numerikusan és kísérletileg is, hogy milyen önszerveződő dinamikák jöhetnek létre a változtatható paraméterek függvényében. Megjelennek különböző lépésmintázatok, fixpontok és kaotikus dinamikák. Az elméleti és kísérleti eredmények összehasonlítása során látható majd a minőségi egyezés, a hexapod robot lábainak spontán szinkronizálódása során különböző típusú mozgásformák figyelhetők meg mindkét módszer esetén.

1.1. Klasszikus robotika

A klasszikus robotika keretében a robotok elsődleges célja a feladatvégzés. Erre a legegyszerűbb példa egy tárgy felemelése majd egy más pozícióba való áthelyezése. A feladatot leggyakrabban robotkarok végzik, melyek *szegmensekből* és *csuklókból* állnak. A szegmensek alatt a robotkar váza értendő. Ha a robotot emberhez hasonlítanánk, a szegmensek jelképeznék a csontokat. A szegmenseket csuklók kapcsolják egymáshoz, melyek ebben az esetben forgató *aktuátorok* (mechanikai elfordulás produkálására alkalmas eszközök). Ezek lesznek az emberi testben a forgó ízületek. Egy rögzített robotkart, mely pár szegmensből és pár csuklóból áll, általánosan manipulátor robotnak, míg egy mozgó, kerekekkel rendelkező robotot, mobil robotnak hív a szakirodalom. A robotkar szabadon mozgó vége, amivel fog vagy amire támaszkodik, az *effektor* vagy *végberendezés*.

A manipulátor a feladatait általánosan a háromdimenziós fizikai térben kell elvégezze. Egy merev test a 3D-ben való leírásához azonban elengedhetetlen tudni annak *pozícióját* és *helyzetét*. A pozíciót egy 3 elemű vektor határozza meg az adott koordináta-rendszerben, ahol a merev testet vizsgáljuk. A 3 elemű vektor a három irány-menti koordinátát tartalmazza. Ebben a

pozícióban a test elfordulását írja le az orientáció, mely ugyancsak egy 3 elemű vektor. Ezt általában három egymás után alkalmazott elemi forgatás szögeivel (Euler-, Cardano-szögek) szokták jellemezni. Ezeket összegezve belátható, hogy egy merev test térbeli helyzetének leírásához hat paraméterre van szükség, ez a rendszer *szabadsági fokainak száma*.

1.1.1. Direkt és inverz kinematika

A manipulátor-feladatok klasszikus megoldási módszerei közé tartozik az inverz kinematika módszere is. Egy n csuklós, három dimenzióban mozgó robotkar esetén az effektor pozíciójának meghatározására geometriai műveletekre van szükség. Egyszerűsített esetben az n csuklót és n szegmens helyzetét és orientációját felírhatjuk $2n$ darab 3 elemű vektorral. Ezen vektorok összessége egy *operátort* eredményez, mely leírja a kapcsolatot két koordináta-rendszer között ($\{A\}$ és $\{B\}$), és a következőképpen jelöljük: ${}^A\xi_B$. Ez kifejezi $\{A\}$ koordináta-rendszerben $\{B\}$ pozícióját és orientációját egyaránt. Általános alakban egy ilyen operátor a *homogén transzformációs mátrix*, mely egy t vektor szerinti translációból (eltolás), és egy R forgatási mátrixból áll:

$${}^A\mathbf{T}_B = \begin{bmatrix} {}^A\mathbf{R}_B & {}^A\mathbf{t}_B \\ 0_{1 \times 3} & 1 \end{bmatrix} \quad (1)$$

Robotkaroknál általában nem használnak gömbcsuklókat, így egy csukló matematikai operátora megfelel egy kiválasztott tengely körüli forgatásnak, mely egy ortonormált forgatási mátrix:

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \quad (2)$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad (3)$$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4)$$

Összetettebb forgatásokat, helyzeteket leíró operátorokra többféle reprezentáció is létezik [1]. A jelen dolgozat keretében a fentiek segítségével azonban már egyszerűen leírható például az effektor $\mathbf{x}$ pozíciója a robotkar bármely $\mathbf{q}$ csuklókoordináta-beállítására:

$$\mathbf{x} = \mathbf{K}(\mathbf{q}), \quad (5)$$

ahol a $\mathbf{K}$ egy nemlineáris függvény. Ezt nevezzük direkt kinematikai feladatnak.

A fordított, úgynevezett inverz kinematikai feladat során meg kell határozni, mely $\mathbf{q}$ csuklósögek esetén érhető el az effektorral egy adott $\mathbf{x}$ térbeli koordináta (és orientáció, de ennek a tárgyalásától most eltekintünk). Az inverz kinematikai feladat analitikusan már általánosan nem mindig megoldható a következő okok miatt [1]:

- az egyenletek nem lineárisak, nincs mindig egzakt megoldásuk;
- legtöbbször egy problémára több lehetséges megoldás is létezik (például egy 2 csuklós 2D - s robotkar, mely esetén egy adott effektor pozíció beállításához két csuklósög páros is megfelel);
- vagy olyan eset is előfordulhat, amikor nem létezik megoldás.

Ezen problémák numerikus módszerekkel kiküszöbölhetők. A már említett n csuklós robotkar csuklósögei meghatározhatóak például numerikus inverz kinematika segítségével, az alábbi differenciálegyenlet alapján (részletekért lásd [1]):

$$\dot{\mathbf{q}}(t) = \alpha \cdot \mathbf{J}^T \cdot \Delta(\mathbf{K}(\mathbf{q}(t)), \xi_E^*), \quad (6)$$

ahol $\Delta(\mathbf{K}(\mathbf{q}(t)), \xi_E^*) = (\mathbf{x}^* - \mathbf{x})$ az effektor $\mathbf{x}$ pillanatnyi és az $\mathbf{x}^*$ célpozíciója közötti 3 elemű eltolásvektor, $\mathbf{J}^T$ a robotkar Jacobi mátrixának transzponáltja, $\dot{\mathbf{q}}(t)$ pedig a csuklósögek idő szerinti deriváltja, valamint α egy állandó paraméter. A Jacobi mátrix három csuklós rendszer esetén:

$$\mathbf{J} = \frac{d\mathbf{K}(\mathbf{q}(t))}{d\mathbf{q}} = \begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \frac{\partial x}{\partial q_3} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \frac{\partial y}{\partial q_3} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \frac{\partial z}{\partial q_3} \end{bmatrix} \quad (7)$$

A fenti egyenlet diszkrét idejű rendszerré alakítása után az alábbi Euler-módszerrel

$$\mathbf{q}(t + dt) = \mathbf{q}(t) + dt \cdot d\mathbf{q}, \quad (8)$$

ahol $d\mathbf{q} = \alpha \cdot \mathbf{J}^T \cdot (\mathbf{x}^* - \mathbf{x}(\mathbf{q}(t)))$ addig iterálható az aktuális csuklókoordináta, míg az egy megadott tolerancia-értéknél közelebb kerül a célpozícióhoz. Ekkor az effektor célbaértnek tekinthető.

1.2. Robotok irányítása dinamikai rendszerekkel

Amint az előző fejezetben is láttuk egy robotkar vezérlése esetében, a robot irányítása az inverz kinematika használatával a csuklók szögeinek interációjával történik, amíg az effektor el nem éri a kívánt célpozíciót. Általános esetben, amikor nem csak egy adott célpozíció elérése a feladat, hanem valamilyen mozgásmintázat létrehozása, akkor a célpozíció egy időben változó, előre megszerkesztett időbeli függvénnyel adható meg. Alternatív megoldásként a robotokat vezérelhetjük úgy is, hogy a csuklókoordináták valamilyen önszerveződő dinamikai rendszerek megoldásaként kapjuk [7].

1.2.1. Dinamikai rendszerek

Dinamikai rendszerek segítségével általában olyan rendszerek időbeni evolúcióját vizsgáljuk, melyek kevés változó mennyiség segítségével leírhatóak. A változó mennyiségek jellemezhetnek különböző fizikai, kémiai, biológiai, vagy akár gazdasági jelenségeket is, ezeknek száma adja meg a rendszer dimenzióját [9]. A dinamikai rendszerek megoldása általában analitikusan nem kivitelezhető, mivel a mozgásegyenletek nem mindig lineárisak, viszont számos numerikus módszer áll rendelkezésünkre [2], melyek segítségével könnyedén leírható a vizsgált rendszer, akár véletlenszerű kezdőértékek esetén is.

Egy adott rendszer állapotát az n dimenziós fázistér egy pontja, a *fázispont* határozza meg: $\mathbf{x} = [x_1, x_2, \dots, x_n]$. Matematikailag a fázispont változása kétféle lehet: folytonos és diszkrét. Folytonos idejű rendszerek esetén az egyenletek elsőrendű, közönséges differenciálegyenletek.

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}); \quad \mathbf{f} = [f_1, f_2, \dots, f_n], \quad (9)$$

ahol $\dot{\mathbf{x}} = \frac{d\mathbf{x}}{dt}$ az idő szerinti deriváltat, $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$ az egyenletrendszer jobb oldalát jelentik. Diszkrét idejű rendszerek esetén leképezések segítségével írható le a rendszer állapota egymás

utáni t és $t + \Delta t$ időpillanatokban (általában $\Delta t = 1$).

$$\mathbf{x}_{t+\Delta t} = \mathbf{f}(\mathbf{x}_t); \quad \mathbf{f} = [f_1, f_2, \dots, f_n] \quad (10)$$

Annak alapján, hogy a dinamikai rendszerek csupán változóikban, vagy expliciten is tartalmaznak-e időfüggést, megkülönböztetünk *autonóm* és *nem autonóm* rendszereket. Autonóm rendszerek esetében a viselkedés a kezdőértékektől függ. Ha a rendszer nem autonóm, a kezdeti feltételek mellett a környezet aktuális állapotától is függ az időbeni evolúció, az egyenlet jobb oldala a t függvénye is.

A fázispont fázistérbeli pályáját, ami a tér egy alhalmaza, *trajektóriának* nevezzük. Annak függvényében, hogy a fázistér hogyan alakul a dinamika során, a rendszer lehet disszipatív, konzervatív, táguló és adaptív.

Ha kezdeti értékektől függetlenül a rendszer a fázistér egy alterébe asszimptotikusan tart, ezt az alteret *attraktornak* nevezzük. Az attraktor lehet egy pont (fixpont), vagy egy zárt görbe (határciklus), esetleg egy kaotikus, fraktálszerű görbe (kaotikus attraktor). Az attraktorok helye és száma a dinamikai rendszertől és annak paramétereitől függenek, autonóm rendszerek esetében időben változatlanok.

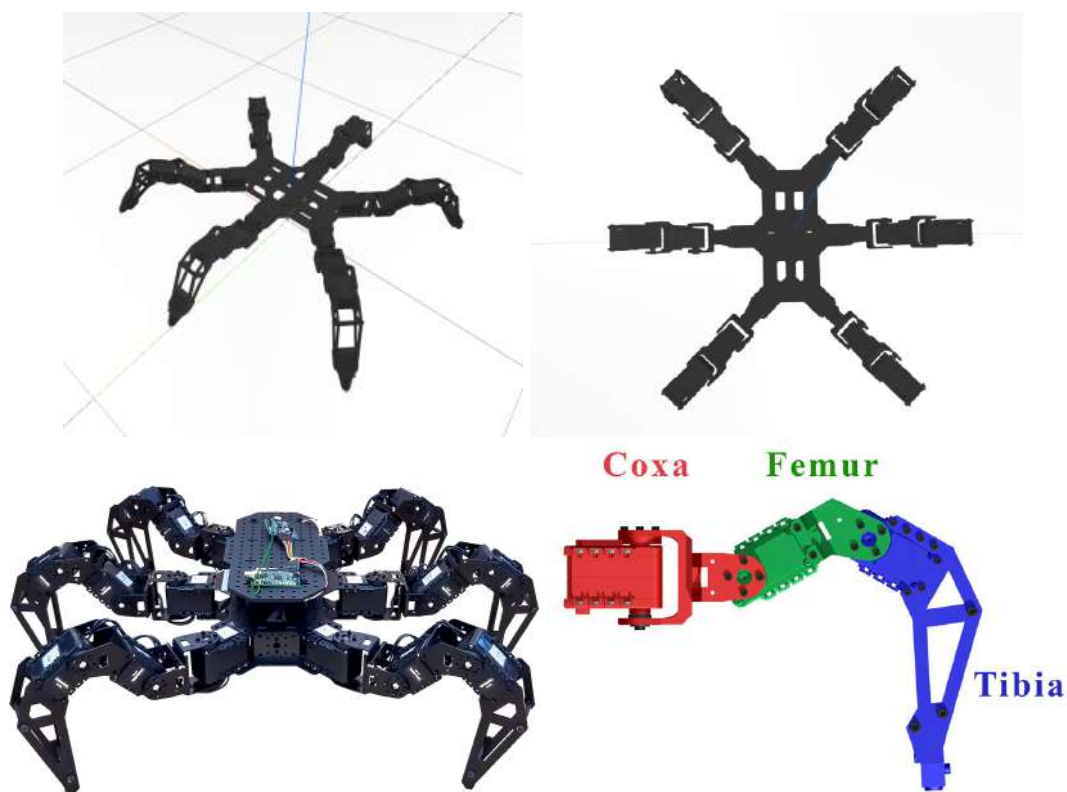
Ha változtatjuk a rendszert jellemző állandó mennyiségeket, az attraktorok helye és minősége is változhat. Minőségi változás esetén beszélünk *bifurkációkról*. A bifurkációtípusok az minőségi átmenetet jellemzik. A leggyakoribb bifurkációtípusok a nyeregpon-, vasvilla-, Hopf-féle, homoklinikus és perióduskettőző bifurkációk [9].

2. A hatlábú robot felépítése

A kutatás során egy hatlábú robot irányítását először numerikus szimulációk segítségével vizsgáljuk, majd kísérletileg ellenőriztük egy kutatási célokra kifejlesztett roboton. A hat láb mindegyike két csuklóval rendelkezik, és további eggyel csatlakozik a robot testéhez, tehát egy láb három, a teljes robot pedig összesen 18 csuklóval bír.

A robotot a Julia programozási nyelv RigidBodyDynamics.jl csomagjának segítségével szimuláljuk, és a MeshCatMechanisms.jl csomaggal vizualizáljuk (lásd a 2.1. ábra felső képeit).

A kísérletek során egy **PhantomX AX Metal Hexapod MK-III** hatlábú robottal dolgoztunk (lásd 2.1. ábra alsó képeit), melyet az *Interbotix Labs* fejleszt. A robotot darabokban árulja a gyártó, így használat előtt egy részletes "step by step" útmutató segítségével lehetséges az összerakása, mely átlagos műszaki készségeket és sok türelmet igényel. Mindegyik csuklót egy speciális szervomotor mozgat. A robotot egy Arduino-kompatibilis mikrokontroller vezérli, melynek áramellátása egy nagy teljesítményű LiPo (lítium - polimer) akkumulátorról történik. A robot össztömege pár kilogramm, a legnagyobb általa elfoglalt terület az összes láb teljes kinyújtásával érhető el, ekkor a hexapod belefér egy 40 cm sugarú körbe.



2.1. ábra. **Felső sor:** A szimulációban tanulmányozott hatlábú robot a MeshCatMechanisms.jl csomaggal vizualizálva. **Bal alsó:** PhantomX AX Metal Hexapod MK-III robot szemből fényképezve. **Jobb alsó:** Egy láb szegmensei [10].

2.1. A robot teste és lábai

A numerikus szimulációkban használt robot paraméterei nagyon hasonlóak a kísérletekben használt robotéival.

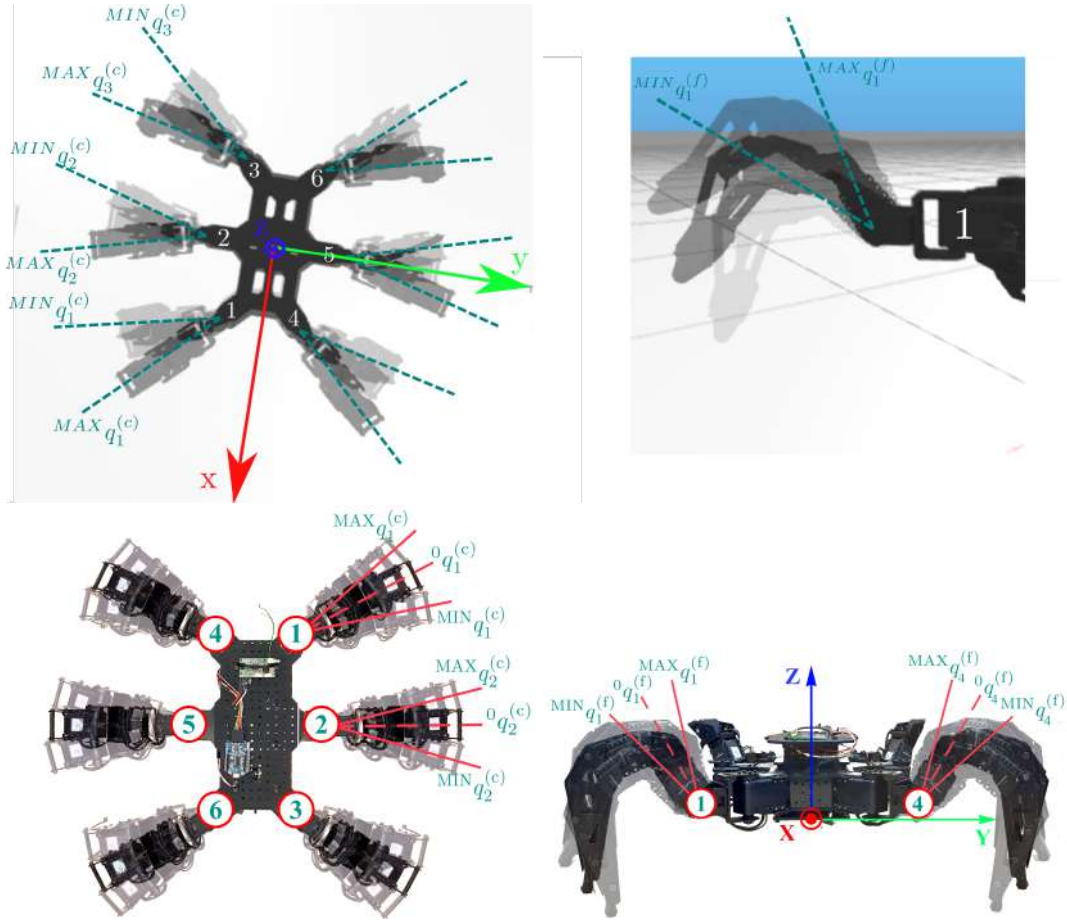
A lábak egyenként egy három dimenzióban mozgó, három csuklós robotkarnak felelnek meg. A lábak három szegmensre vannak osztva, melyek neveit a rovar láb részek ihlettek: coxa, femur és tibia (csípő, comb és sípcsont). A szegmensek elhelyezkedését az 2.1. ábra szemlélteti. Ez alapján egy láb három csuklója is ezeket a neveket viseli és azok első betűs rövidítéseit.

A PhantomX robot váza többnyire erős alumínium lemezekből épül fel, melyeket acél csavarok és anyák kapcsolnak össze. Kisebb elemek és távolságtartók műanyagból készülnek, melyek a kisterhelésű zónákban vannak elhelyezve. A test belsejében kap helyet az akkumulátor, mely ez által védve van a külső mechanikai veszélyek elől, továbbá a robot "agya" is itt kap helyet. Opcionálisan több felső lemez is szerelhető a testre így lehetőség nyílik rádiók, robotkarok, kamerák, szenzorok és további számítógépek elhelyezésére. A lábak effektora egy -a tapadás növelésének érdekében- gumival bevont félgömb felületű lábvég.

2.2. Hatlábú robot irányításának alapjai

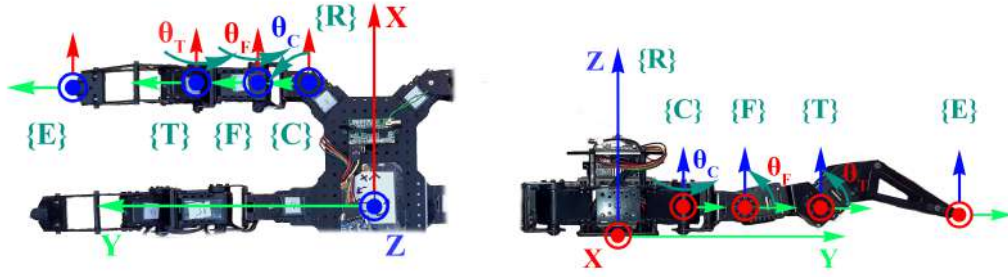
A robot irányításához szükséges pár jelölés és paraméter beállítása. A lábak számozása megegyezik a 2.2. ábrán szereplő értékekkel. Minden esetben a tibia csuklókat fix pozícióban tartjuk, $\mathbf{q}^{(t)} = {}^0\mathbf{q}^{(t)} = \text{konstans}$. A coxa csuklók $\mathbf{q}^{(c)}$ és a femur csuklók $\mathbf{q}^{(f)}$ értékei a $[\text{MIN } \mathbf{q}^{(c)}, \text{MAX } \mathbf{q}^{(c)}]$ és $[\text{MIN } \mathbf{q}^{(f)}, \text{MAX } \mathbf{q}^{(f)}]$ intervallumokból vehetnek fel értékeket, az intervallumok határai konstansak a dolgozatban tárgyalt modellek esetén, a nyugalmi ${}^0\mathbf{q}^{(c,f)}$ szögeket mi választjuk meg, ezen értékek összessége jelenti a láb alapállapotát. A $\mathbf{q}^{(l)}$ vektor 6 elemű, a q_i elem az i -edik láb csuklószögét jelöli. A 2.2. ábrán a megengedett csuklósög-tartományok vannak feltüntetve.

A robot testéhez rögzítünk egy jobbsodrású koordinátarendszert, a továbbiakban $\{R\}$, melynek xOy síkja a robot alsó lemeze által meghatározott síkkal egybeesik, az origó pedig a robot középpontjában található. Az "előre" haladási irány az Ox tengelynek felel meg, a láb femur általi felemelése pedig az Oz irányban történik. A maximális coxa értékek a lábak teljesen előre fordulását jelentik, míg a minimumok a teljesen hátra fordulást, így az Oz tengely szerinti pozitív elfordulás a robot jobb oldalán a maximális érték irányába való elmozdulást, a bal oldalán a minimum irányba történő elmozdulást jelenti. A maximális femur értékek a lábak teljes felemelését írják le, ekkor helyezkedik el a robot teste a legközelebb a földhöz.



2.2. ábra. A robot coxa- és femurszögeinek határai a szimulációs környezetben (**felső sor**) valamint a valódi robot (**alsó sor**) esetében: a $[MIN q_i^{(c)}, MAX q_i^{(c)}]$ és $[MIN q_i^{(f)}, MAX q_i^{(f)}]$ intervallumok a hatlábú roboton bejelölve. Az i a robot lábait számláló index. A robothoz csatolt koordinátarendszer jobbsodrású. **Bal:** A robot lábainak számozása és a coxa csuklók elfordulási intervallumai $MAX q_i^{(c)} - MIN q_i^{(c)} = \pi/6$. **Jobb:** $MAX q_i^{(f)} - MIN q_i^{(f)} = \pi/5$.

A minimum femur csukló értékek a lábak teljes lenyomását jelentik, ekkor a robot teste a legmagasabban áll a földhöz viszonyítva. Ez alapján az Ox tengely szerinti pozitív elfordulás a robot jobb oldalán a minimális értékek felé való elmozdulást, a robot bal oldalán a maximum irányba történő elmozdulást jelenti. A coxa és femur csuklók forgatásai ábrázolva vannak a 2.3. ábrán.



2.3. ábra. Egy tetszőleges láb esetén a robothoz, a csuklókhoz és az effektorhoz rögzített koordináta rendszerek ($\{R\}, \{C\}, \{F\}, \{T\}, \{E\}$) és azok elforgatási szögei ($\theta_{C,F,T}$), ezek színei megegyeznek a tengely színével, mely körül történik az elforgatás. **Bal:** Felülnézet, a coxa csuklók elfordulása mindig az xOy síkban történik, Oz tengely körül. **Jobb:** Az x tengely irányából nézve a femur csuklók elfordulása mindig az előzőleg meghatározott koordinátarendszerek yOz síkjaiban történik, Ox tengely körül.

A robot lábainak végpontjai, az effektorok pozíciója kiszámolható a robot saját koordináta rendszerében. Ez tiszta eltolások és tiszta forgatások megfelelő sorrendben való alkalmazásaként is felírható:

$${}^R\mathbf{T}_E = {}^R\mathbf{T}_C(\mathbf{t}_C) {}^R\mathbf{T}_C(\theta_C) \cdot {}^C\mathbf{T}_F(\mathbf{t}_F) {}^C\mathbf{T}_F(\theta_F) \cdot {}^F\mathbf{T}_T(\mathbf{t}_T) {}^F\mathbf{T}_T(\theta_T) \cdot {}^T\mathbf{T}_E(\mathbf{t}_E), \quad (11)$$

ahol $\mathbf{t}_{C,F,T,E}$ eltolás vektorok a robot és a lábak valós méreteivel, a csuklók és az effektor pozícióját tartalmazzák mindig az előző koordináta-rendszerhez képest; $\theta_{C,F,T}$ az adott coxa (C), femur (F) vagy tibia (T) csukló Oz vagy Ox tengely körüli elforgatás szöge, mely a következő alapján számolható: $\theta_{C,F} = {}^0q^{(c,f)} - q^{(c,f)}$; $\mathbf{T}$ homogén transzformációs mátrix.

Hőtérképen ábrázolva az effektor pozíziójának z komponensét lábanként az idő függvényében kirajzolódik a lépésmintázat (lásd a következő fejezetekben).

3. Oszcilláló neuronmodell robotok vezérléséhez

A szakirodalomban már számos különböző dinamikai rendszert dolgoztak ki hatlábú robotok vezérléséhez. Az egyik legegyszerűbb ilyen kollektív viselkedést is mutató modell a csatolt fázisoszcillátorokból álló Kuramoto-rendszer [4, 5]. A dolgozat célja ezzel szemben egy valódi határciklusokat használó, de ugyanakkor egyszerű neurális hálózati modell kifejlesztése, melynek segítségével egy hatlábú robotot tudunk irányítani. A következő fejezetben tehát először bemutatjuk a Kuramoto-modell egy általunk módosított verzióját, majd lépésről-lépésre, az egyszerűtől a komplex felé haladva felepítünk egy neurális modellt, amelynek az alapegységeit oszcilláló neuronok alkotják.

3.1. Fázisoszcillátorok

Szinkronizációs jelenségek modellezésére gyakran használunk fázisoszcillátorokat. Szinkronizáció alatt jelen dolgozatban olyan jelenségeket értünk, amely során csatolt oszcillátorok fázisai között kialakul egy időben állandó fáziskülönbség. Hagyományosan a 0 vagy 2π többszörösének megfelelő fáziskülönbséget nevezzük szinkron állapotnak, a nemzéró de időben állandó különbséget pedig fázisblokkolt állapotnak. Ilyen jelenségek például a vastaps, vagy az a jelenség, amikor az utcán egymás mellett sétáló emberek egyszerre vagy ellentétesen lépkednek, de szinkronizálhatnak (állandó fáziskülönbséget létrehozva) egy hatlábú robot lábai is, kialakítva így egy olyan dinamikát, ami eredményes járást produkál. A fázisoszcillátorok reprezentációjaként használt körbeforgó vektort a szakirodalomban a *rotátornak* nevezik. Ennek körbeforgási sebessége, a szögsebessége állandó, $\omega = \text{konstans}$. Az elfordulás szöge, amit a rotátor fázisának is nevezünk θ .

Több rotátort szimulálva, ha létezik közöttük valamilyen kölcsönhatás, akkor az őket jellemezhető egyetlen mennyiség, a szögsebességük megváltozik. Egy ilyen többrotátoros rendszer a *Kuramoto-modell*, melyet Kuramoto Josiki fejlesztett ki [8]. Általánosítva n db összekapcsolt rotátor rendszerre, a következő matematikai egyenletrendszer írható fel:

$$\dot{\theta}_i = \omega_i + \sum_{j=1}^n K_{ij} \sin(\theta_j(t) - \theta_i(t)), \quad (12)$$

ahol i futó index, az oszcillátorok sorszámát jelöli. ω_i a rotátor sajátfrekvenciája, K_{ij} -t pedig csatolási mátrixnak nevezzük. Ez a csatolási mátrix a rotátorok közötti kölcsönhatás erősségét jellemzi, minél nagyobb K_{ij} , annál jobban befolyásolja egymást az i . és a j .

rotátor. A klasszikus modellben minden oszcillátor minden más oszcillátorral ugyanolyan erősségű csatolással van kapcsolva. Elegendően erős csatolás esetén ez azt eredményezi, hogy a rendszer arra törekszik, hogy mindegyik oszcillátor ugyanolyan állandó frekvenciával és fázissal rendelkezzen. Így jön létre a szinkronizálódás.

Gyakorlatba ültetve a modellt, a Kuramoto modell segítségével könnyedén vezérelhetőek lépkedő robotok. Komplexebb vezérléshez szükség lehet a rotátorok szögsebességének szabályozására a fázis függvényében. A viselkedés leírásához a (12) egyenletben ω lineáris tagja helyett bevezethető egy koszinuszos tag, így az egyenlet a következőképpen alakul:

$$\tau \dot{\theta}_i = 1 + a \cos(\theta_i) + \sum_{j=1}^n K_{ij} \sin(\theta_j(t) - \theta_i(t)), \quad (13)$$

ahol az a kontrollparaméter, a rotátorok gyorsulásgörbéjét simítja. Kis a értékre a sebesség változása folytonosabb lesz, míg magas a értékre szaggatottab. A τ egy időállandó, másnéven sajátidő.

3.1.1. Taszító kuramoto-modell

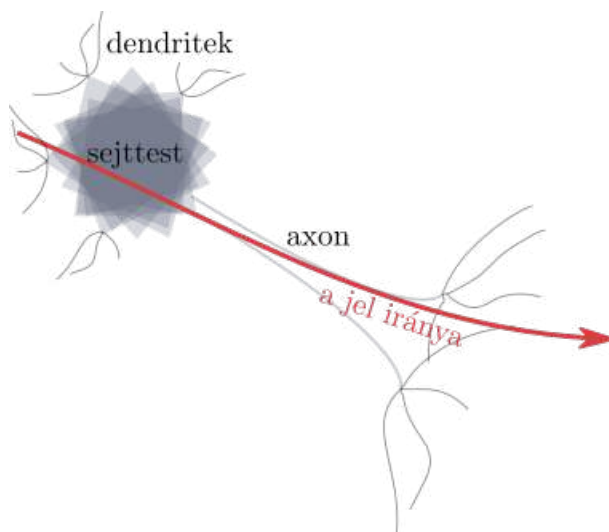
Lépkedő robotok irányításánál, ahogyan az embernél is, bizonyos lábaknak ellenfázisban kell lenni, tehát amíg egy láb a levegőben van, addig a másik a földön tartja a robot testét. Az általános Kuramoto-rendszerben a rotátorok a szinkron kialakulása után teljesen egyszerre forognak, ennek felbontása érdekében bevezethető egy taszító csatolás, $K_{ij} < 0$. Ennek eredményeképpen az egymást taszítva kölcsönható oszcillátorok arra törekszenek, hogy elfordulási szögeik között egy állandó fáziskülönbség legyen, vagyis $\Delta\theta = \text{konstans} \neq 0$. Így jön létre a fázisblokkolt állapot. Megfelelő csatolási mátrix kiválasztásával egyes rotátorok fázisban, míg mások ellenfázisban forognak. A modell adaptációját hatlábú robotok irányításához a kísérleti eredmények bemutatásánál tárgyaljuk.

3.2. Oszcilláló neuron visszacsatolással

Fázisoszcillátorok helyett olyan magasabb dimenziós dinamikai rendszerek is használhatók, amelyek rendelkeznek legalább egy határciklus attraktorral. Ilyen dinamikai rendszerek neuronmodellek segítségével is kialakíthatók.

3.2.1. Neurális modellek dióhéjban

Egy emberi agyban nagyjából $10^9 - 10^{11}$ idegsejt, más néven neuron található (lásd 3.1. ábra). Ezek lehetnek például érző (szenzorként felfogják a környezetből érkező ingereket), motorikus (a mozgásért felelnek), vagy interneuronok (az ingert továbbítják).



3.1. ábra. A neuron vázlatos rajza. Az inger a dendritek irányából érkezik a sejttestbe, majd az axonon keresztül távozik, a soron következő neuron dendritjeihez.

Egy ilyen idegsejt egyik tulajdonsága az ingerlékenység. A fizikában ezt a tulajdonságot a membránpotenciállal jellemezzük. Ha a neuron nyugalomban van, a potenciálja nagyjából -80 mV . Az akciós potenciál -40 mV , a neuron ezen értéken egy impulzust bocsájt ki, más néven tüzel [1]. Szinapszinak nevezünk két neuron közötti funkcionális kapcsolatot, mely lehet kémiai vagy elektromos kölcsönhatás. Ez az egyik neuron axonja és a másik dendritje között történik. Egy neuron önmagával is létesíthet szinapszist.

Legegyszerűbben neurális csoportok együttesét vizsgálva ezek működését átlagolva adhatjuk meg egy átlagos neuron membránpotenciáljának dinamikáját:

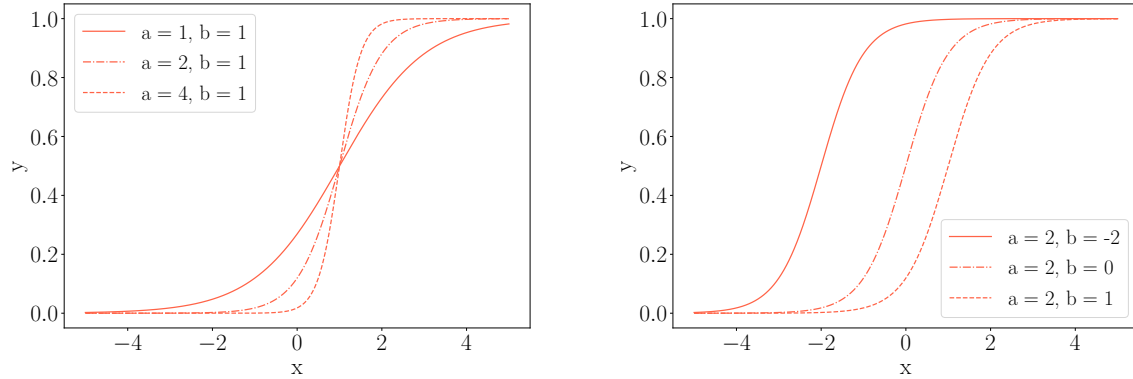
$$\tau_x \dot{x} = -x + I \quad (14)$$

Az egyenletben szereplő x az átlagos membránpotenciál, τ_x egy időskála vagy sajátidő, I a neuronba érkező input, vagyis a membránpotenciált befolyásoló elektromos és vagy kémiai jelek összessége. Ezen modell keretében a neuron által kibocsátott jel erősségét a tüzelési rátával jellemezhetjük, amely egy aktivációs függvényként is tekinthető, és a következőképpen írható

fel:

$$y(x) = \frac{1}{1 + e^{a(b-x)}} \quad (15)$$

Amint látható, y a membránpotenciál függvénye a b aktivációs küszöbértéken és az a aktiválási meredekségen keresztül (lásd 3.2. ábra).



3.2. ábra. Az $y(x)$ szigmoid függvény különböző a paraméterek és b változók esetén. **Bal:** az aktiválási meredekség a lépcsőszerű ugrás meredekségét határozza meg. **Jobb:** az aktiválási küszöbérték az ugrás helyét jelöli.

Neurális hálózatok esetén, egy olyan neuronra, amelyik több másikkal létesít szinapszist, az átlagos membránpotenciál legegyszerűbben úgy írható fel, mint

$$\tau_{xi} \dot{x}_i = -x + \sum_j w_{ij} y_j(x_j) \quad (16)$$

Ebben az esetben inputként mindenik neuron aktivitása jelenik meg az egyes membránpotenciálok változásában, egy-egy w_{ij} csatolási állandóval szorozva, amit egyszerű súlyként értelmezünk. Az adaptív neuronok matematikai modelljében az a és b értékek sem állandóak időben:

$$\begin{aligned} \dot{a}_i &= f(a_i, b_i, x_i) \\ \dot{b}_i &= f(a_i, b_i, x_i) \end{aligned} \quad (17)$$

Ha a hálózatok csatolási állandói is változnak az idővel, tanuló hálózatokról beszélhetünk. Ekkor

$$\dot{w}_{ij} = f_{ij}(x_1, x_2, \dots, x_n, a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_n, \dots). \quad (18)$$

3.2.2. Homeosztázis

A neuron homeosztázisa az idegsejt optimális működési tartomány körüli létét jelenti. Ahhoz, hogy a kellő feltételeket megteremtse, a neuronok belső adaptáció során változtatják az aktivációs küszöbértéküket:

$$\begin{aligned}\tau_{xi}\dot{x}_i &= -x + \sum_j w_{ij}y_j(x_j) \\ \tau_{bi}\dot{b}_i &= y - 0.5\end{aligned}\tag{19}$$

ahol τ_{bi} az egyes neuronok aktivációs küszöbértékének sajátidei. A b változása révén a modell egy biológiailag realiztikusabb rendszert ír le. Alacsony, $y < 0.5$ aktivitás esetén csökken az aktivációs küszöbérték, míg magas aktivitás esetén nő, így átlagosan 0.5 optimális tüzelési ráta jellemzi a dinamikát (részletekért lásd [11]).

3.2.3. Oszcilláló neuron

Az egyik legegyszerűbb, oszcillációra képes neurális modell egy önmagával szinapszist létesítő, homeosztázist megvalósító neuron modellje:

$$\begin{aligned}\tau_x\dot{x} &= -x + w_y y \\ \tau_b\dot{b} &= y - 0.5\end{aligned}\tag{20}$$

w_y a neuron a csatolási állandója. Numerikus adaptáció során a paramétereket a következőképpen választjuk meg: $\tau_x = 0.05$, $\tau_b = 0.5$, $a = 2$

A rendszer fixpontja $(x^*, b^*) = (\frac{w_y}{2}, \frac{w_y}{2})$, ekkor $y = 0.5$. A Jacobi mátrix alakja

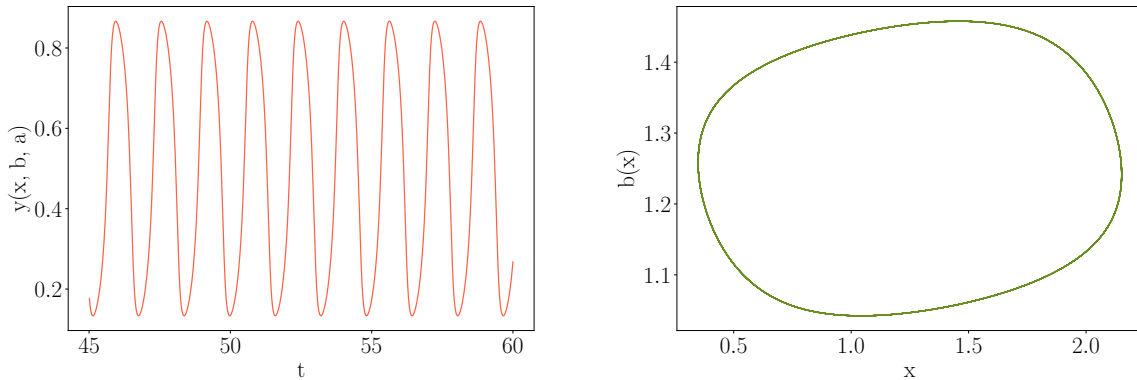
$$\mathbf{J} = \begin{bmatrix} \left. \frac{\partial \dot{x}}{\partial x} \right|_* & \left. \frac{\partial \dot{x}}{\partial b} \right|_* \\ \left. \frac{\partial \dot{b}}{\partial x} \right|_* & \left. \frac{\partial \dot{b}}{\partial b} \right|_* \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau_x}(\frac{aw_y}{4} - 1) & -\frac{1}{\tau_x} \frac{aw_y}{4} \\ \frac{1}{\tau_b} \frac{a}{4} & -\frac{1}{\tau_b} \frac{a}{4} \end{bmatrix}\tag{21}$$

A mátrix determinánsa és nyoma

$$\det(J) = \frac{a}{4\tau_x\tau_b} > 0, \quad \text{tr}(J) = \frac{1}{\tau_x}(\frac{aw_y}{4} - 1) - \frac{1}{\tau_b} \frac{a}{4}.\tag{22}$$

Bifurkáció akkor történik következik be a w_y változtatásának függvényében, ha $\text{tr}(J) = 0$. Ekkor $w_y = \frac{\tau_x}{\tau_b} + \frac{4}{a} = 2.1$. A $\text{tr}(J) > 0$ bármilyen $w_y > 2.1$ esetén, vagyis a rendszer sajátértékeinek valós részei minden ilyen értékre pozitívak, tehát a fixpont instabil és így

határciklus jön létre. Például a $w_s = 2.5$ paraméterérték esetén a rendszer trajektóriája határciklusba tart, az aktivitás értéke pedig periodikus, 0.5 körül oszcillál (lásd 3.3. ábra).



3.3. ábra. Oszcilláló neuronmodell $w_y = 2.5$ paraméterre. **Bal:** a 0.5 körül oszcilláló periodikus aktivitásérték az idő függvényében. **Jobb:** határciklus $b - x$ fázistérben.

3.3. Túlszillapított mozgás vezérlése

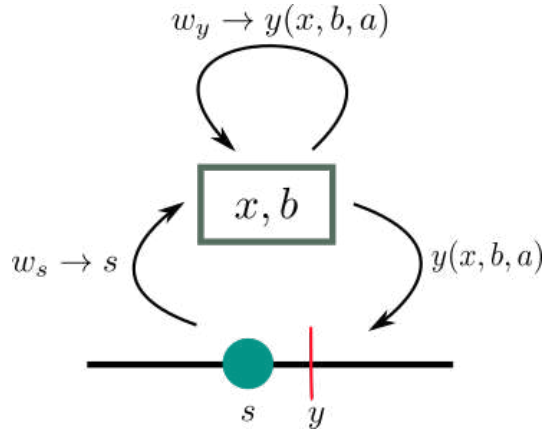
Az előző fejezetben bevezett neuronmodellt felhasználhatjuk egy rúdon csúszkáló túlszillapított test vezérlésére, például úgy, hogy a neuron y aktivitását tekintjük a test célpozíciójának. Továbbá, hogy legyen visszacsatolás is a test s aktuális pozíciójáról, feltételezzük, hogy annak értékét a neuron megkapja bemenő jelként:

$$\begin{cases} \tau_x \dot{x} = -x + w_s s + w_y y \\ \tau_b \dot{b} = y - \frac{1}{2} \\ \tau_s \dot{s} = y - s \end{cases}, \quad y = \frac{1}{1 + e^{a(b-x)}} \quad (23)$$

Az s a test pozíciója, így jellemzi a változó a környezetet, visszaadja a külső rendszerben lévő test pillanatnyi pzícióját (lásd 3.4. ábra). A τ_s a külső rendszer változójának sajátideje, w_s és w_y az első egyenletben megjelenő összegben szereplő súlyok értékei.

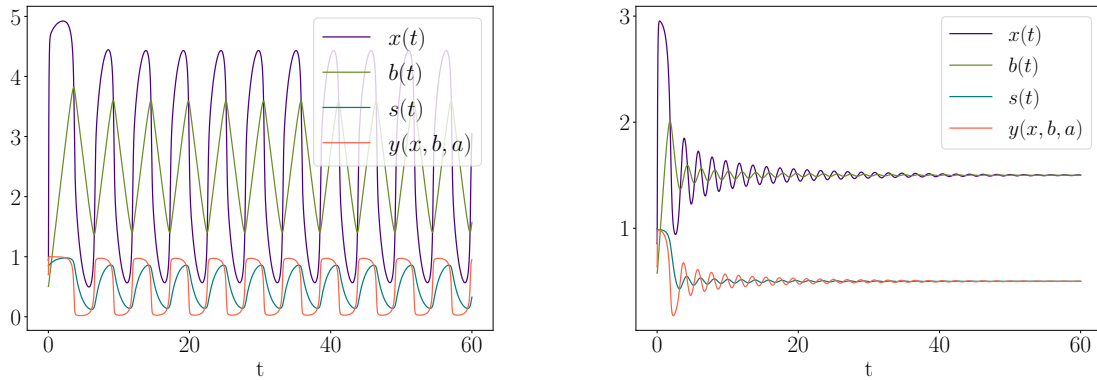
3.3.1. A rendszer dinamikája

A dinamikát az általunk befolyásolható paraméterek függvényében tanulmányozhatjuk. A rendszerben megjelenő sajátidők annak karakterisztikájából fakadóak, kívülről nem változtathatjuk őket. Rögzítjük az $a = 2$ paraméter értékét, így a dinamikai rendszert w_s és w_y súlyok függvényében vizsgáljuk.



3.4. ábra. A túlszabályozott egy neuronnal irányított rendszer szemléltető ábrája: A motor neuronja w_s súllyal kapja vissza kívülről az aktuális pozíciót, mellette w_y súllyal küldi pedig saját aktivitását. Ezeket feldolgozva szolgáltatja a rendszernek az y célpozíciót, amit a test kis késéssel ugyan, de próbál követni. A mozgás célja, hogy s az y -ba kerüljön.

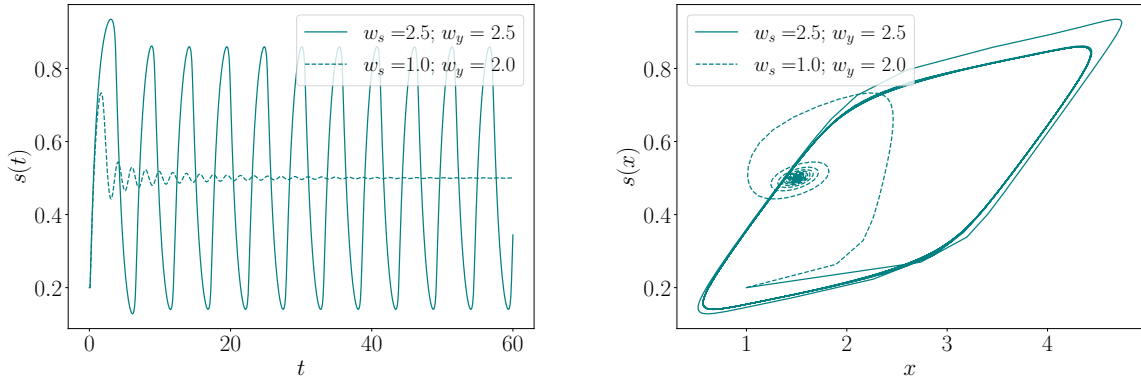
Megfigyelhető, hogy különböző (w_s, w_y) párra, véletlenszerű kezdőértékekre más-más a dinamika minősége (lásd 3.5. ábra). Megfigyelhető a homeosztázis: növekvő membránpotenciál esetén a küszöbaktivitás is nő, így védekezve a túltüzelés ellen, megőrizve a neuron épségét. Ha a potenciál csökkenni kezd, vele csökken le a küszöbaktivitás is. Ugyanakkor remekül látszik az s és y kapcsolat is, miszerint a rendszer pozíciója lemaradva követi a pillanatnyi célpozíciót.



3.5. ábra. A rendszer változóinak idősorai különböző (w_s, w_y) értékek esetén, véletlenszerű kezdőértékekkel. **Bal:** periodikus oszcilláció $(w_s, w_y) = (2.5, 2.5)$ paraméterekre. **Jobb:** $(w_s, w_y) = (1, 2)$, a dinamikai rendszer fixpontba kerül.

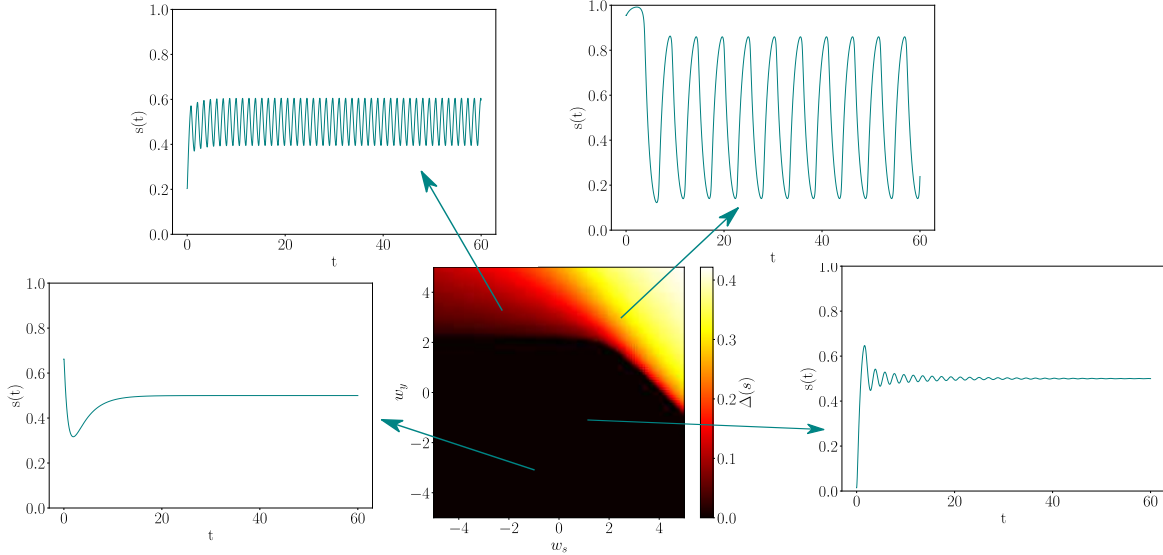
Ugyanabból a kezdőpontból indítva a rendszert más-más változtatható paraméterpárosra is tesztelhetjük. Eltérő (w_s, w_y) esetén ugyancsak különböző dinamikát lehet megfigyelni (lásd 3.6. ábra).

A kontrollparaméterek függvényében fázistérkép mutatja a rendszer paraméterektől függő minőségét. Ha elég sok idő telik el a rendszer működésének kezdetétől számítva, az vagy



3.6. ábra. $(x_0, b_0, s_0) = (1, 0.5, 0.2)$ kezdőfeltételből induló rendszer különböző dinamikai, ha a paramétereket különbözőképpen választjuk meg. **Bal:** a külső rendszer pozíciójának idősorai, **Jobb:** $s - x$ fázistéren fixpontba vagy határciklusba tartó dinamika. A paraméterek az ábrán olvashatóak.

határciklusba, vagy fixpontba jut. Az előbbi esetén a rendszerhez tartozó test pozíciója egy idő után már nem változik, ezért az s változó szórása $\Delta s = 0$. Az utóbbi esetben a határciklus miatt ez a változó nem állandósul, így szórása $\Delta s \neq 0$. A $w_s, w_y \in [-5, 5]$ térre készített térkép mutatja, hogy hol történik oszcilláló, és hol jut fixpontba a rendszer (lásd 3.7. ábra).



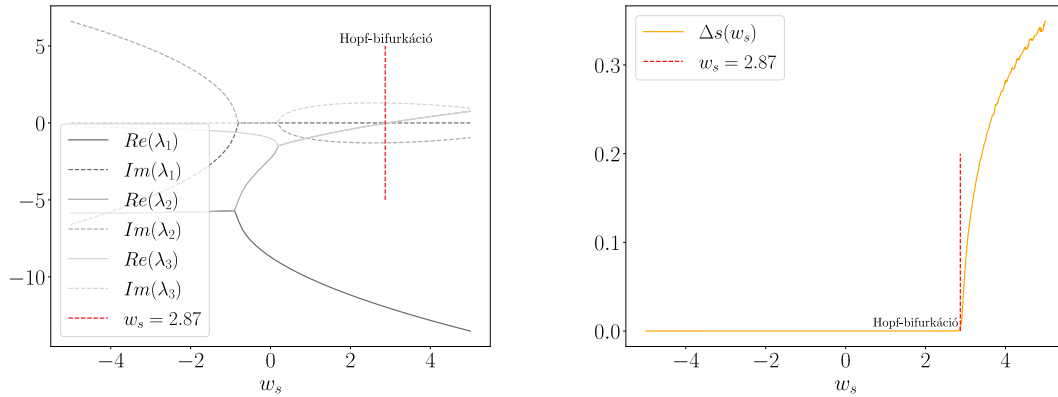
3.7. ábra. A fázistérkép, és egy-egy pontjában megjelölt dinamika: A sötét mező zéró szórást, vagyis fixpontot mutat, a vörös és sárga színek s szórását jelölik, azon a tartományon határciklusba jut a trajektória. Az idősorokat random kezdőértékekre futtatjuk, a fókusz a rendszer hosszútávú viselkedésén van.

3.3.2. Fixpontok, bifurkáció

A rendszer fixpontja $(x^*, b^*, s^*) = (\frac{w_s + w_y}{2}, \frac{w_s + w_y}{2}, 0.5)$, ekkor $y^* = 0.5$. A hozzá tartozó Jacobi mátrixot a következőképpen írjuk fel:

$$\mathbf{J} = \begin{bmatrix} \left. \frac{\partial \dot{x}}{\partial x} \right|_* & \left. \frac{\partial \dot{x}}{\partial b} \right|_* & \left. \frac{\partial \dot{x}}{\partial s} \right|_* \\ \left. \frac{\partial \dot{b}}{\partial x} \right|_* & \left. \frac{\partial \dot{b}}{\partial b} \right|_* & \left. \frac{\partial \dot{b}}{\partial s} \right|_* \\ \left. \frac{\partial \dot{s}}{\partial x} \right|_* & \left. \frac{\partial \dot{s}}{\partial b} \right|_* & \left. \frac{\partial \dot{s}}{\partial s} \right|_* \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau_x} \left(\frac{aw_y}{4} - 1 \right) & -\frac{1}{\tau_x} \frac{aw_y}{4} & \frac{w_s}{\tau_x} \\ \frac{1}{\tau_b} \frac{a}{4} & -\frac{1}{\tau_b} \frac{a}{4} & 0 \\ \frac{1}{\tau_s} \frac{a}{4} & -\frac{1}{\tau_s} \frac{a}{4} & -\frac{1}{\tau_s} \end{bmatrix} \quad (24)$$

A mátrix sajátértékeit numerikusan keressük, a Jacobi-mátrix alapján. Rögzített $w_y = 1$ esetén vizsgáljuk a rendszer sajátértékeinek valós, illetve képzetes részeit w_s függvényében (lásd 3.8. ábra). Ahogyan nő ez az érték, kezdetben mindhárom sajátérték valós, a $w_s^{(H)} \approx 2.87$ bifurkációs pontot követően két sajátérték komplexszé alakul át. Itt történik meg a szuperkritikus Hopf bifurkáció, amely során kis amplitúdóval rendelkező stabil határciklus jön létre.



3.8. ábra. **Bal:** A túlszillapított rendszer (24)-es Jacobi-mátrixának 3 sajátértéke a w_s kontrollparaméter függvényében $w_y = 1$ esetén: folytonos vonallal a sajátértékek valós, szaggatott vonallal pedig képzetes részeit ábrázolva. **Jobb:** A rendszer s pozíciójának szórása w_s függvényében, ha $w_y = 1$. A piros szaggatott vonallal jelölt bifurkációs értéknél nagyobb w_s értékekre bekövetkező nem nulla szórás határciklus megjelenését jelenti.

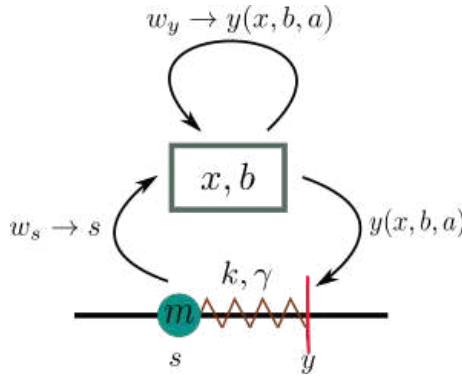
3.4. Csillapított rugó-tömb rendszer vezérlése

Ha a korábbi oszcilláló neuront egy csillapított harmonikus oszcillátor irányítására adaptáljuk, az első két egyenletet nem módosítjuk, viszont a harmadik, környezetet jellemző egyenlet Newton második törvényévé alakul: $\ddot{s} = \frac{\sum F}{m}$. Ekkor a rendszer modelljében a neuronnal irányított, rúdon ide-oda csúszkáló, tömeggel rendelkező, környezetével surlódó golyóra egy rugalmas erő hat a célpozíció irányába (lásd 3.9. ábra), és az egyenletek a következő alakot öltik:

$$\begin{cases} \tau_x \dot{x} = -x + w_s s + w_y y \\ \tau_b \dot{b} = y - \frac{1}{2} \\ \dot{s} = v \\ \dot{v} = -\frac{k}{m}(s - y) - \frac{\gamma}{m}v \end{cases}, y = \frac{1}{1 + e^{a(b-x)}} \quad (25)$$

m a test tömege, k a rugalmassági állandó, γ a csillapítási tényező. A rendszer így négy dimenzióval rendelkezik, és a mennyiségeket adimenzionálisnak tekintjük. Az egyszerűség kedvéért a test tömegét és a rugóállandót egységnyinek vesszük, így az utolsó egyenlet felírható, mint

$$\dot{v} = -(s - y) - \gamma v \quad (26)$$



3.9. ábra. A rendszer szemléltető ábrája: A motor neuronja w_s súllyal kapja vissza kívülről az információt, mellette w_y súllyal küldi a saját aktivitását. Ezeket feldolgozva szolgáltatja a rendszernek az y célpozíciót, aminek irányába az m tömegű golyót egy k rugalmassági állandójú csillapított rugó húzza oda.

Ha a rendszer túlcillapított, vagyis $\gamma v \gg \dot{v}$, a (26) egyenlet bal oldala elhanyagolhatóan kicsi, így

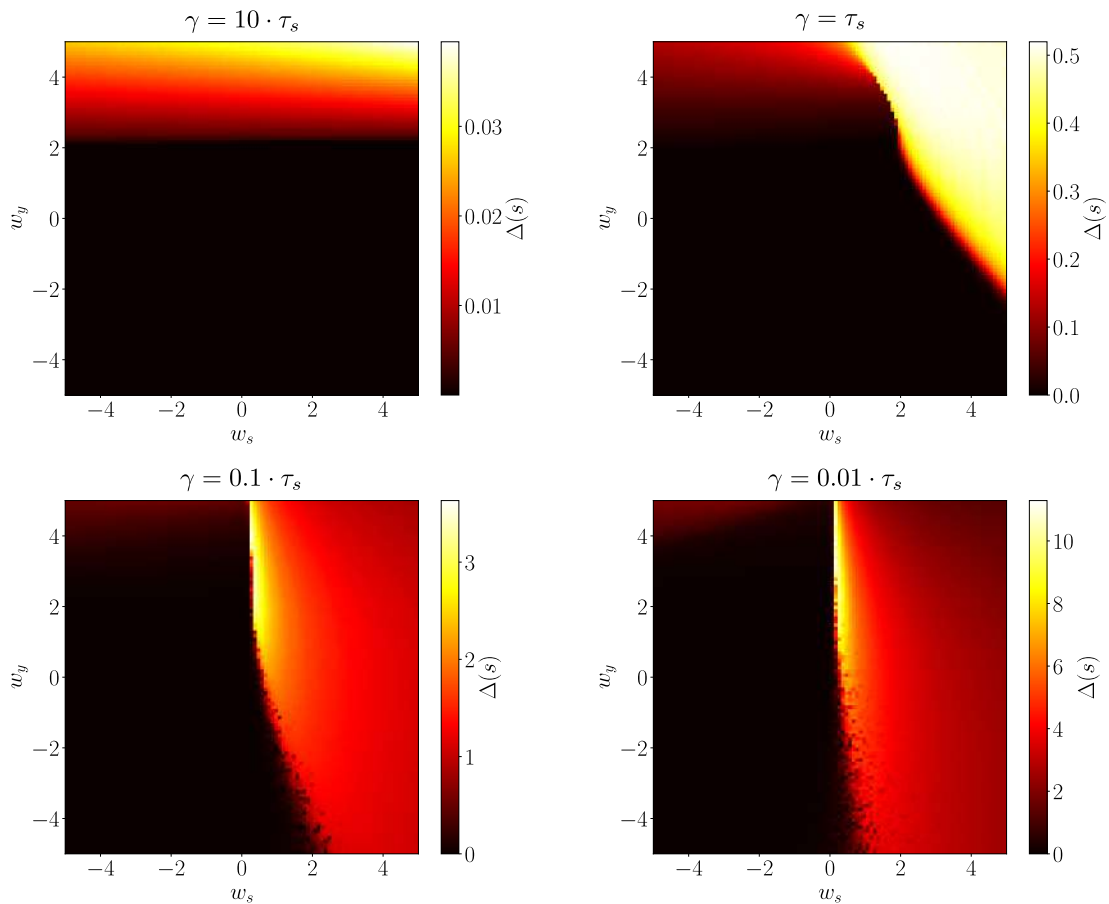
$$\dot{s} = v = \frac{y - s}{\gamma} \quad (27)$$

Innen belátható, hogy túlcillapítás esetén, ha $\gamma = \tau_s$, a rendszer visszavezethető a 3.3.

fejezetben tárgyalt rendszerre.

3.4.1. A rendszer dinamikája

A következőkben rögzítjük a szimulációs paramétereket: $a = 2$, $\tau_x = 0.05$, $\tau_b = 0.5$, $\tau_s = 1$ és a rendszer fázistérképét nem csak a (w_s, w_y) függvényében, hanem különböző csillapítási tényezőkre is vizsgáljuk (lásd 3.10. ábra). Következtetésként levonható, hogy nagyon nagy csillapítási tényezőre, ha $\gamma = 10 \cdot \tau_s$, a rendszer működését nem módosítja a neuron által lekért, külső rendszert jellemző pozíció, fontosabb az aktivitása. Ugyanakkor elegendően kicsi, $\gamma = 0.01 \cdot \tau_s$ esetén a dinamika minősége csak a rendszerben lévő test aktuális helyzetétől függ, az aktivitás befolyása minimális. Amint várható volt, a $\gamma = \tau_s$ esetén a fázistérkép a korábbi 3D-s modellre kapott fázistérképre hasonlít. Megfigyelhető, hogy a pozíció szórásának felvett értékei nőnek a csillapítási tényező csökkenésével.



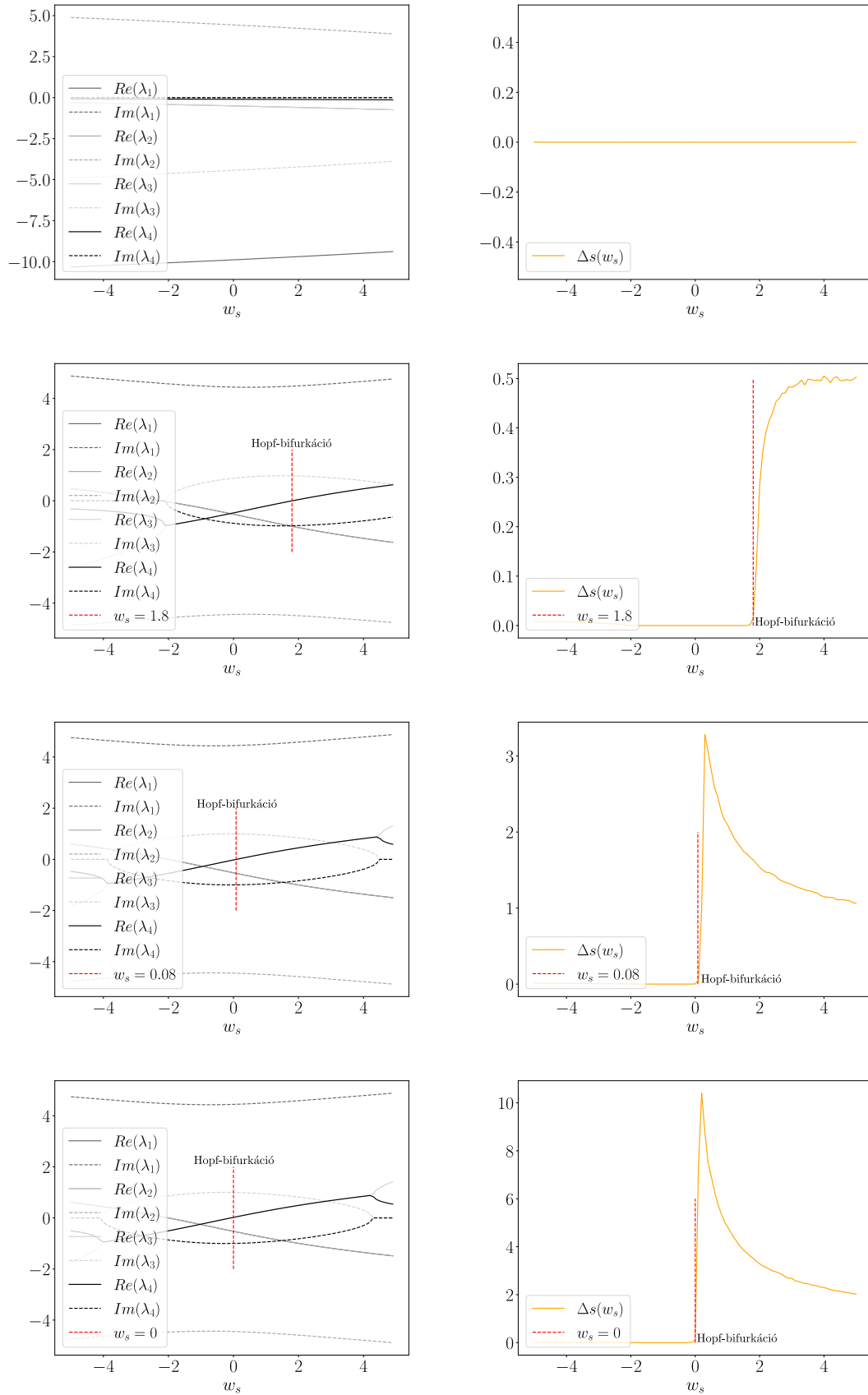
3.10. ábra. Négy különböző γ csillapítási tényezőre vizsgált fázistérképek a $w_y - w_s$ paramétertérben. A színek a pozíció $\Delta(s)$ szórását jelölik.

3.4.2. Fixpontok, bifurkáció

A rendszer fixpontja $(x^*, b^*, s^*, v^*) = (\frac{w_s+w_y}{2}, \frac{w_s+w_y}{2}, 0.5, 0)$, ekkor $y^* = 0.5$. A Jacobi mátrixot a következőképpen írjuk fel:

$$\mathbf{J} = \begin{bmatrix} \left. \frac{\partial \dot{x}}{\partial x} \right|_* & \left. \frac{\partial \dot{x}}{\partial b} \right|_* & \left. \frac{\partial \dot{x}}{\partial s} \right|_* & \left. \frac{\partial \dot{x}}{\partial v} \right|_* \\ \left. \frac{\partial \dot{b}}{\partial x} \right|_* & \left. \frac{\partial \dot{b}}{\partial b} \right|_* & \left. \frac{\partial \dot{b}}{\partial s} \right|_* & \left. \frac{\partial \dot{b}}{\partial v} \right|_* \\ \left. \frac{\partial \dot{s}}{\partial x} \right|_* & \left. \frac{\partial \dot{s}}{\partial b} \right|_* & \left. \frac{\partial \dot{s}}{\partial s} \right|_* & \left. \frac{\partial \dot{s}}{\partial v} \right|_* \\ \left. \frac{\partial \dot{v}}{\partial x} \right|_* & \left. \frac{\partial \dot{v}}{\partial b} \right|_* & \left. \frac{\partial \dot{v}}{\partial s} \right|_* & \left. \frac{\partial \dot{v}}{\partial v} \right|_* \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau_x} \left(\frac{aw_y}{4} - 1 \right) & -\frac{1}{\tau_x} \frac{aw_y}{4} & \frac{w_S}{\tau_x} & 0 \\ \frac{1}{\tau_b} \frac{a}{4} & -\frac{1}{\tau_b} \frac{a}{4} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{a}{4} & -\frac{a}{4} & -1 & -\gamma \end{bmatrix} \quad (28)$$

A mátrix sajátértékeit numerikusan keressük, a Jacobi-mátrix alapján. A bifurkáció, hasonlóan a túlszillapított test vezérléséhez, Hopf típusú, viszont nagy, $\gamma = 10 \cdot \tau_s$ csillapítási tényező esetén nem történik meg. A bifurkáció a λ_4 sajátérték valós részének előjelváltásakor történik meg (lásd 3.11. ábra).

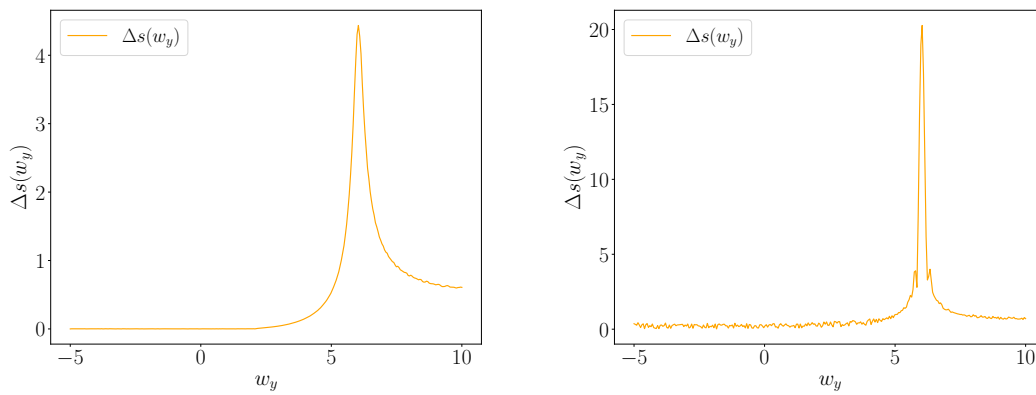


3.11. ábra. Bifurkációk rögzített $w_y = 2$ esetén a 4D-s dinamikai rendszerben. **Bal oldalon** a rendszer sajátértékeinek valós, illetve képzetes részei a w_s függvényében. **Jobb oldalon** a rendszer s pozíciójának szórása a w_s függvényében. **Első sor:** $\gamma = 10\tau_s$. **Második sor:** $\gamma = \tau_s$. **Harmadik sor:** $\gamma = 0.1\tau_s$. **Negyedik sor:** $\gamma = 0.01\tau_s$. A piros szaggatott vonal a Hopf-bifurkáció helyét jelöli.

3.4.3. Rezonancia

Abban az esetben, ha a külső rendszer kapcsolási állandója $w_s = 0$, a rendszer első két egyenlete a rendszer többi egyenletétől függetlenné válik, és visszaadja az oszcilláló neuron-modellt (3.2.3. fejezet). Ekkor a neuron a külső környezetben lévő test helyzetétől függetlenül ugyanazt a periodikus jelet produkálja (3.3. ábra).

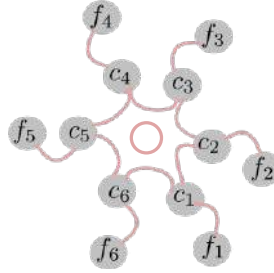
Kis csillapítási tényező esetén a fázistérképen a $w_s = 0$ függőleges egyenes mentén egy kis tartományban kiugróan magas s szórás figyelhető meg (3.12. ábra). A w_y függvényében ábrázolt szórás grafikonon rezonancia-jelenséget figyelhetünk meg.



3.12. ábra. Rögzített $w_s = 0$ esetén vizsgáljuk a környezet s állapotának szórását. **Bal oldal:** $\gamma = 0.1\tau_s$. **Jobb oldal:** $\gamma = 0.01\tau_s$

3.5. Vezérlés csatolt neuronokkal

A hexapodnak 12 neuron irányítja a mozgását: hat db mozgatja a femurokat, újabb 6 a coxákat (lásd 3.13. ábra). Ezeket az idegsejteket elsőként még robot nélkül, a korábbiakhoz hasonlóan túlcillapított testeket vezérelve szimuláljuk. A neuronok egymással csatolva vannak, még hozzá úgy, hogy a coxa neuronok visszatérítik egymásnak az aktivitásukat, ugyanazzal a w_k csatolási állandóval. Ezen kívül minden neuron lekéri a saját aktivitását, és hozzá tartozó csillapított test helyzetét is. A femurokhoz tartozó neuronok működésére csupán a saját aktivitásuk, és a hozzájuk tartozó coxa neuronok aktivitása van befolyással, utóbbi szintén a w_k csatolási állandón keresztül. A neuronok homeosztázisa is megvalósul, ezért mindenik neuron



3.13. ábra. A rendszer vázlatos rajza. A coxa neuronok egymással, a femurt irányító neuronok csak a coxákkal vannak kapcsolatban.

aktivitási küszöbértéke változik az idővel. A hálózat működését a (29)-es egyenletrendszer írja le.

$$\begin{cases} \tau_x^{(c)} \dot{x}_i^{(c)} = -x_i^{(c)} + w_y y_i^{(c)} + w_k \sum_{i \neq j} y_j^{(c)} + w_s s_i^{(c)} \\ \tau_b \dot{b}_i^{(c)} = y_i^{(c)} - \frac{1}{2} \\ \tau_x^{(f)} \dot{x}_i^{(f)} = -x_i^{(f)} + w_k y_i^{(c)} + w_y y_i^{(f)} \\ \tau_b \dot{b}_i^{(f)} = y_i^{(f)} - \frac{1}{2} \end{cases} \quad (29)$$

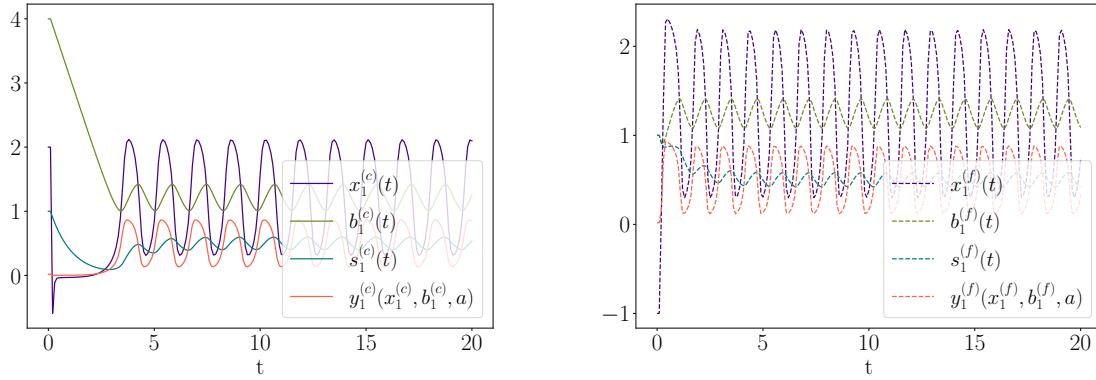
$$y_i^{(c,f)} = \frac{1}{1 + e^{a(b_i^{(c,f)} - x_i^{(c,f)})}} \quad (30)$$

$\tau_x^{(c)}$	$\tau_x^{(f)}$	τ_b	a	w_k	w_y	w_s
0.05	0.025	0.5	2.0	-0.4	[-5, 5]	[-5, 5]

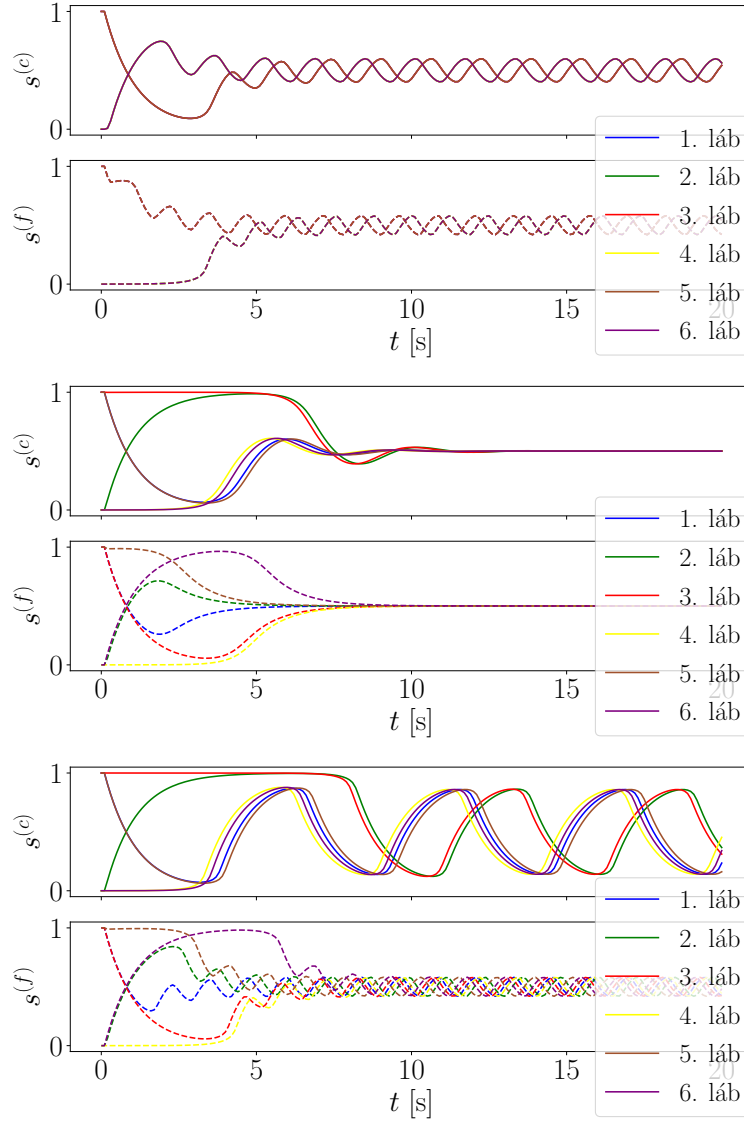
1. táblázat. A fenti paramétereket rögzítjük a továbbiakban

3.5.1. A rendszer dinamikája

A rendszer első coxa és femur neuronjára vizsgáljuk a hozzá tartozó tulajdonságok időbeni evolúcióját és a csillapított test s pozíciójának idősorát, különböző paraméterek esetén (lásd 3.14. ábra). Emellett minden coxa- és femur-neuronhoz tartozó külső rendszer helyzetének idősorát is vizsgáljuk (lásd 3.15. ábra).



3.14. ábra. $(w_s, w_y) = (0, 2.5)$ **Bal:** c_1 coxa változóinak idősora **Jobb:** f_1 femur változóinak idősora



3.15. ábra. A coxák és femurok által mozgatott testek s helyzete az idő függvényében. **Felső:** $(w_s, w_y) = (0, 2.5)$, a páros és páratlan neuronok esetén az idősorok megegyeznek, a megfelelő lábak fázisban vannak. Minden s periodikus, a páros és páratlan helyzetek idősorai egymáshoz képest el vannak tolódva. A coxák és femurok nagyjából ugyanolyan periódusú oszcillációt eredményeznek. **Középső:** $(w_s, w_y) = (1.5, 1.5)$, a rendszer fixpontba kerül a kezdeti kaotikus dinamika után. **Alsó:** $(w_s, w_y) = (2.5, 2.5)$, oszcilláló a dinamika, de a páros és páratlan testek nincsenek fázisban. A femurok által mozgatott testek mozgásperiódusa sokkal kisebb, mint a coxák által mozgatott testek esetén.

3.6. Hatlábú robot numerikus szimulációja

A 2. fejezetben bemutatott robotot a 12 neuronból álló rendszerrel vezéreljük. Az idegsejtek működését leíró egyenletek változatlanok maradnak, a robot lábainak mozgását megadó nyomatók pedig egy PD kontrollernak megfelelően a következőképp adhatóak meg:

$$\hat{M}_i^{(c,f)} = K_p \Delta q_i^{(c,f)} - K_d v_i^{(c,f)}, \quad (31)$$

ahol $i = 1, 2, \dots, 12$. K_p a rugóerőből származtatott erőnyomaték együtthatója, K_d a súrlódásból származtatott nyomaték paramétere, $\Delta q_i^{(c,f)}$ a láb coxa és femur célszögeinek és aktuális csuklósögeinek különbségét jelöli. Ez egy csuklósög-különbség, mely megadja, hogy az adott neuron által irányított láb elérni kívánt csuklósögétől mennyire tér el a *julia* beépített függvényének segítségével lekért pillanatnyi csuklósög. A célpozíciót $y_i^{(c,f)}$ -ből számoljuk úgy, hogy a $[0, 1]$ intervallumot, amit $y_i^{(c,f)}$ felvehet, átskálázzuk minden egyes láb coxa és femur szögének $[\text{MIN } q_i^{(c,f)}, \text{MAX } q_i^{(c,f)}]$ intervallumára:

$$\begin{cases} q_i^{(c)} = \text{MIN } q_i^{(c)} + y_i^{(c)} (\text{MAX } q_i^{(c)} - \text{MIN } q_i^{(c)}) \\ q_i^{(f)} = \text{MIN } q_i^{(f)} + y_i^{(f)} (\text{MAX } q_i^{(f)} - \text{MIN } q_i^{(f)}) \end{cases} \quad (32)$$

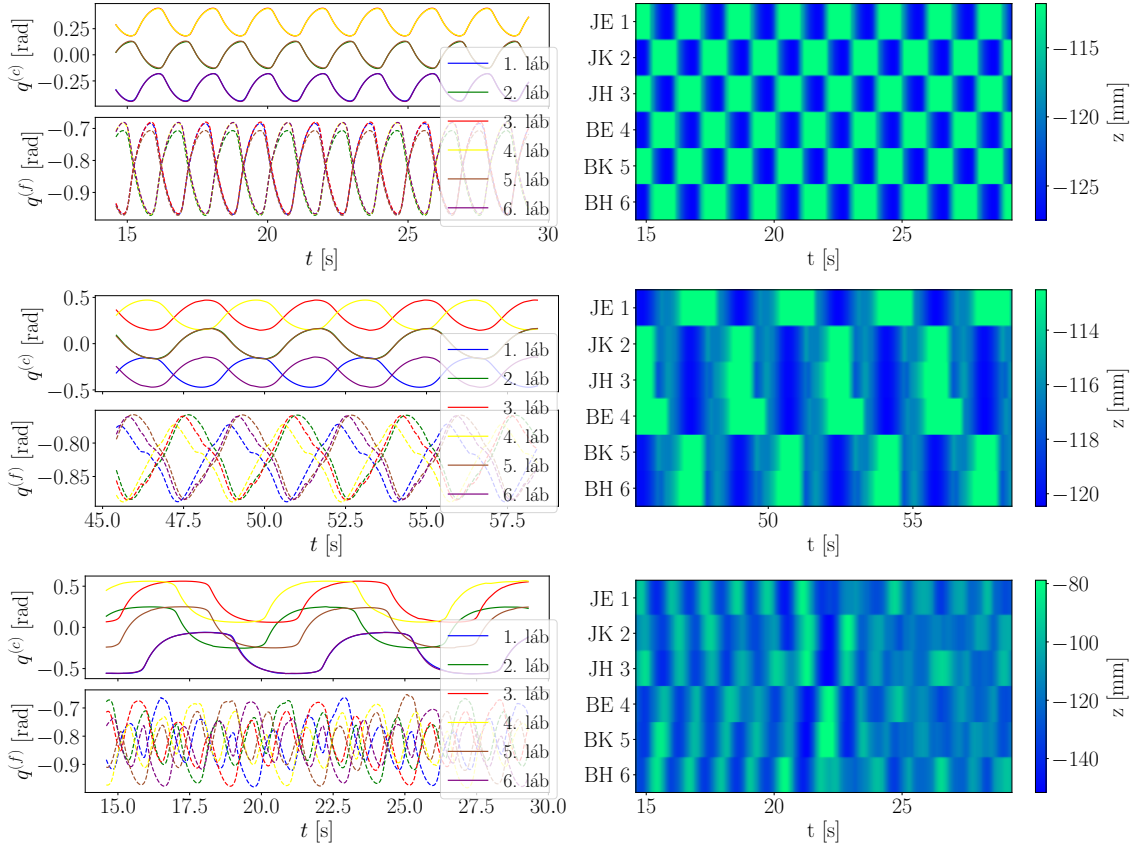
A robotot a Julia programozási nyelv RigidBodyDynamics.jl csomagjának segítségével szimuláljuk, és a MeshCatMechanisms.jl csomaggal vizualizáljuk (lásd a A.2 kiegészítést). A szimulációs közegben létrehozott robotot a robothoz tartozó URDF (Unified Robotics Description Format) fájl segítségével építjük meg.

A numerikus szimuláció lépései a következők:

- inicializálások: beállítjuk a dinamikai rendszer paramétereit, az egyes neuronok változóinak kezdőértékeit
- $dt = 10^{-4}$ időlépésenként léptetjük az időt
- minden időpillanatban megnézzük, hogy a szimuláció kezdete óta, vagy az előző nyomaték megadásától számítva eltelt-e $\Delta t = 10^{-3}$ idő. Ha igen, Euler-módszer segítségével számoljuk ki a neuronok változóinak értékét.
- megadjuk a csuklók célpozícióit, majd a nyomatékokat
- mentjük az adott pillanatban a robot valamennyi tulajdonságát: az időt, csuklósögeket, pillanatnyi aktivitásokat, aktivációs küszöbértékeket és a tibiák effektorjainak magasságát, a robot testének magasságát (a földhöz viszonyítva).

A szimulációban lévő ciklust a RigidBodyDynamics.jl beépített *simulate()* függvényével végeztetjük.

A robot járása önszerveződően alakul ki, vagyis mi csupán a kezdőértékeket, a különböző paramétereket és a csuklósögek határait adjuk meg a rendszernek. A dinamikában megjelenő szinkron a csatolás során valósul meg, mindenféle külső hatás nélkül.

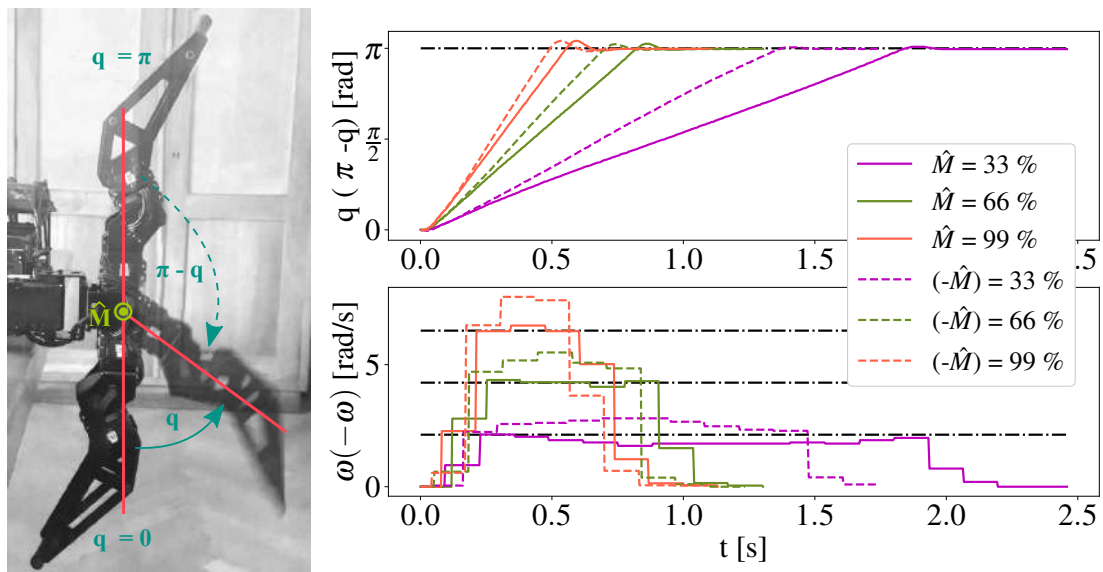


3.16. ábra. Soronként az egyes coxa és femur csuklászögek idősora és a robot lépésmintázata (amikor egy láb kék akkor azon támaszkodik a robot, mikor zöld a levegőben van; a függőleges tengelyen a láb pozíciójának rövidítése található, például: jobb első, bal középső) különböző (w_s, w_y) párosokra, továbbá $K_p = 15$ és $K_d = 2\sqrt{K_p}$. **Első sor:** $(w_s, w_y) = (0, 2.5)$. **Második sor:** $(w_s, w_y) = (1.5, 1.5)$. **Harmadik sor:** $(w_s, w_y) = (2.5, 2.5)$.

A csuklászögek kirajzolása mellett elkészítjük a robot lépésmintázatát is (lásd 3.16. ábra), mely remekül szemlélteti az egyébként matematikailag komplex rendszert. A robot testének aktuális, földfelszíntől mért magasságából (z_0) kivonjuk a lábak effektorjainak magasságát, melyek a tibiák végpontjai ($z_i^{(t)}$). Minden láb esetén a szimuláció minden pillanatában elvégezzük a műveletet. Ha a $z = z_0 - z_i^{(t)}$ negatív érték abszolút értéke nagy, az a test és a láb végpontja közötti nagy távolságot jelenti, ekkor van a láb effektorja a földön, amit a lépésmintázaton késsel jelölünk. Abba az esetben, ha a különbség modulusza egy kisebb szám, az azt jelenti, hogy a láb végpontja közelebb van a robot testéhez, így az felemelt állapotban van, ami a mintázaton a zöld színű jelölés.

4. Kísérleti eredmények

A PhantomX robot Dynamixel AX-12-es típusú okos szervomotorokkal rendelkezik (angolul "smart servo", részletekért lásd a Függelék A részét). Tényleges nyomaték ennek a típusú aktuátornak, úgy ahogy az a 3.6 fejezetben leírt numerikus tanulmányozás során lehetséges, itt nem állítható, csak a kifejtett nyomaték felső határa. A 4.1. ábrán a maximális nyomatékhoz tartozó sebesség vizsgálatának kísérlete tekinthető meg. A kísérlet során a robotkar (a robot lába) π szögelfordulást végez különböző maximális nyomatékbeállítások esetén (33%, 66%, 99% relatív értékek) a gravitáció ellenében, valamint a gravitációs gyorsulás irányában. A mozgás során mérjük a csukló q szögét valamint a hozzá tartozó ω szögsebességet. Következtetésként elmondható, hogy a szervomotor forgási sebessége nagy elfordulási szög esetén közel állandó a mozgás nagy részében, és szabályozható a nyomaték maximális értékével. A lefele mozgás során súlyerő gyorsító nyomatéka miatt a sebesség nagyobb, mint amikor a nyomaték lassítja a mozgást. A robot vezérléséhez a tényleges nyomaték állításának hiányában célpozíciókat állítunk az algoritmusokban a következő kísérletek során.



4.1. ábra. Egy szervomotor sebességének tanulmányozása π szögű elfordulás esetén különböző maximális nyomaték értékekre, úgy hogy a szervomotornak a célpozíció van állítva és a sebesség nincs korlátozva. **Bal:** A kísérletet szemléltető ábra az elfordulás irányáról (felfele: $q = 0 \rightarrow \pi$, lefele: $q = \pi \rightarrow 0$). **Jobb:** A csukló szüge és szögsebessége az idő függvényében. A lefele mozgás zárójellel van jelezve a feliratokban. A nyomaték felső korlátja százalékban kifejezve, $\hat{M} = M/M_{max}$.

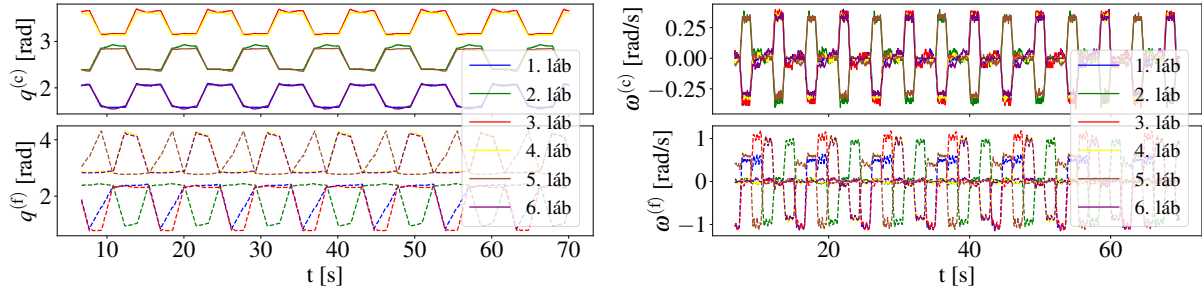
4.1. Hatlábú robot irányítása inverz kinematikával

Az effektorok a robot koordináta rendszerbeli pozíciójának meghatározásából visszaszámolható az ehhez a pozícióhoz szükséges csuklók szögei. Ennek segítségével az összes láb mozgása kiszámolható, melyek megfelelő párosításával lépésmintázat hozható létre. Egy egyszerű váltakozó háromlábú mintázat, a továbbiakban tripod mintázat, felbontható 6 különböző lépés egymás után való ismétlésére. Ezek a lépések Ox irány menti vagy Oz irány menti elmozdulások. Ez a következő képpen van megtervezve:

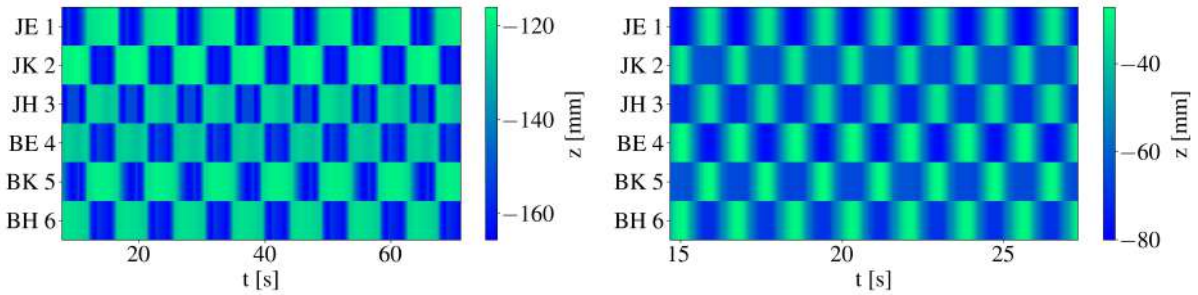
1. lépés: 1,3,5 -ös lábak felemelése ($q_{1,3,5}^{(f)} \rightarrow \text{MAX } q_{1,3,5}^{(f)}$)
2. lépés: 1,3,5 -ös lábak előre, miközben a 2,4,6 - os lábak hátra tolása ($q_{1,3,5}^{(c)} \rightarrow \text{MAX } q_{1,3,5}^{(c)}$;
 $q_{2,4,6}^{(c)} \rightarrow \text{MIN } q_{2,4,6}^{(c)}$)
3. lépés: 1,3,5 -ös lábak leeresztése ($q_{1,3,5}^{(f)} \rightarrow \text{MIN } q_{1,3,5}^{(f)}$)
4. lépés: 2,4,6 - os lábak felemelése ($q_{2,4,6}^{(f)} \rightarrow \text{MAX } q_{2,4,6}^{(f)}$)
5. lépés: 1,3,5 -ös lábak hátra, miközben 2,4,6 - os lábak előre tolása ($q_{1,3,5}^{(c)} \rightarrow \text{MIN } q_{1,3,5}^{(c)}$;
 $q_{2,4,6}^{(c)} \rightarrow \text{MAX } q_{2,4,6}^{(c)}$)
6. lépés: 2,4,6 - os lábak leeresztése ($q_{2,4,6}^{(f)} \rightarrow \text{MIN } q_{2,4,6}^{(f)}$)

Minden egyes lépéshez tehát kiszámolandó a (6) egyenlet alapján az összes mozgó csukló $q^{(c)}, q^{(f)}$ új értéke. Az effektor helyzete ebben az esetben nem releváns számunkra mivel az egy félgömb felület, így csak a pozíciójára koncentrálva több lehetséges csukló szög párt is eredményez a (6) egyenlet megoldása, ez a lépésmintázatot nem befolyásolja.

A lábak coxa és femur csuklóinak idősorait ábrázolva tanulmányozható azok pozíció és sebesség változásai (lásd 4.2. ábra). Mind a 6 lábra kiszámolva a (11) - os egyenlet által adott koordináta hármast, abból az Oz tengely menti időbeli evolúciót ábrázolva hőtésképen, szépen láthatóvá válik a tripod mintázat (lásd 4.3. ábra).



4.2. ábra. Idősorok az inverz kinematikával vezérelt esetben. **Bal:** Pozíció időszora, a lépésmintázat tervezésénél felírt lépésekből az 1,6 és 3,4 összevonhatóak, ez a négy lépés különül az idősorokon is. A pozíció lineáris változása a A.1.3 fejezetben vizsgált állandó fordulási sebességet támasztják alá. **Jobb:** Sebesség időszora, hasonlóan kiolvasható a szögletességből az állandó sebesség.



4.3. ábra. Lépésmintázatok **Bal:** Inverz kinematika lépésmintázat. Az ábrán látható mintázathoz az effektorok, a robot koordináta rendszerében számolva, 4 cm - t mozdultak Ox és Oz mentén. **Jobb:** Kuramoto lépésmintázat, megfigyelhető a nem lineáris tagok jelenléte, mivel jóval összemosódottabb az ábra mint az inverz kinematikával vezérelt esetben

4.2. Vezérlés fázisoszcillátorokkal

Hat lábú robot vezérléséhez 6 darab csatolt fázisoszcillátorból álló taszító kuramoto-rendszert alkalmazunk. A 6 rotátor egy-egy lábat irányít, mind a coxa mind a femur csuklásszögek a rotátorok fázisából van kiszámolva. Egy oszcillátort leíró egyenlet a (13) egyenlet szerint alakul, mely a következővel azonos:

$$\tau \dot{\theta}_i = 1 + a \cos(\theta_i) + \sum_{i \neq j} K_{ij} \sin(\theta_j - \theta_i), \quad (33)$$

ahol a kontrollparaméter és τ időkonstans értékét rögzítjük, $a = 0.4$ és $\tau = 0.3$.

Az 6 darab oszcillátor egymáshoz csatolását a (34)-es taszító csatolási mátrix írja le. Minden egyes lábhoz rendelt oszcillátor a legközelebbi szomszédjaival van ellentétesen csatolva, például: 1 - es láb a 2 - essel és a 4 - essel.

$$K_{ij} = -0.6 \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \quad (34)$$

A folytonos idejű (33) - es rendszer megoldásához azt diszkért idejű rendszerré alakítjuk:

$$\theta_i(t + dt) = \theta_i(t) + dt \cdot \frac{1}{\tau} [1 + a \cos(\theta_i(t)) + \sum_{i \neq j} K_{ij} \sin(\theta_j(t) - \theta_i(t))], \quad (35)$$

ahol dt az egy ciklus elvégzéséhez szükséges idő, ez az első lépésben 40ms, majd a következőben a robot által mért értékkel egyenlő. Kísérleteinkben ez átlagosan 33ms volt.

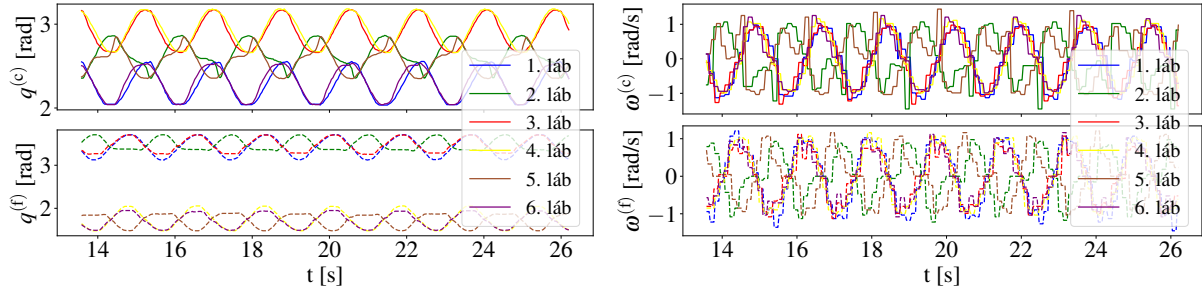
A célpozíció kiszámítás az oszcillátorok fázisának átskálázásával történik. Megjelenik a coxa csuklók késleltetése a femurokhoz képest (36) egyenletrendszer, ennek célja, hogy lépéskor előbb emelje a lábát a robot mielőtt azzal haladni próbál, így az nem súrlódik a földön.

$$\begin{cases} q_i^{(c)}(t) = \text{MIN} q_i^{(c)} + \frac{1}{2}(1 + \cos(\theta_i(t) - \frac{3\pi}{2}))(\text{MAX} q_i^{(c)} - \text{MIN} q_i^{(c)}) \\ q_i^{(f)}(t) = \text{MIN} q_i^{(f)} + \frac{1}{2}(1 + \cos(\theta_i(t))) (\text{MAX} q_i^{(f)} - \text{MIN} q_i^{(f)}) \end{cases} \quad (36)$$

Egy ciklus a mikrokontrolleren a következő lépéseket tartalmazza:

- $\theta_i(t + dt)$ értékek kiszámolása
- $q_i^{(c,f)}$ csuklőszögek kiszámolása
- csuklőszögek küldése a szervomotoroknak
- csuklőszögek és csuklósebességek lekérése és az idővel együtti küldése a számítógépre soros porton
- dt ciklus idő kiszámítása

Az ehhez a kísérlethez tartozó idősorok 4.4. ábrán; a lépésmintázat a 4.3. ábrán látható.



4.4. ábra. Idősorok a fázisoszcillátorokkal vezérelt esetben. Megfigyelhető, hogy a pozíció itt már nem lineárisan változik, ahogyan a sebesség sem. Ez a (33) egyenletbeli $1 + a_i \cos(\theta_i)$ nem lineáris tag jelenlétével magyarázható. Ennek eredményeként a robot mozgása jóval folytonosabb lesz. **Bal:** Megfigyelhető a femur pozíciók változásánál, hogy hiányzik a csúcsozódás minimum vagy maximum értékek közelében. Ennek oka, a lábak roboton való elhelyezkedése, közepén. **Jobb:** Sebesség idősora, folyamatosan gyorsuló/lassuló csuklókat ír le. Mielőtt csukló elér a maximum vagy minimum tartományába lelassul, ez azt eredményezi, hogy simább lesz a robot haladása.

4.3. Vezérlés csatolt neuronokkal

A numerikus szimulációhoz hasonlóan 12 neuronból álló rendszerrel irányítjuk a robotot. Az egyenletek megoldása itt is Euler módszerrel van számolva, ehhez a (29) egyenleteket diszkrét idejűvé rendszerré redukáljuk:

$$\begin{cases} x_i^{(c)}(t + dt) &= x_i^{(c)}(t) + dt \cdot \frac{1}{\tau_x^{(c)}} [-x_i^{(c)}(t) + w_y y_i^{(c)} + w_k \sum_{i \neq j} y_j^{(c)} + w_s s_i^{(c)}] \\ x_i^{(f)}(t + dt) &= x_i^{(f)}(t) + dt \cdot \frac{1}{\tau_x^{(f)}} [-x_i^{(f)}(t) + w_k y_i^{(c)} + w_y y_i^{(f)}] \\ b_i^{(c,f)}(t + dt) &= b_i^{(c,f)}(t) + dt \cdot \frac{1}{\tau_b} [y_i^{(c,f)} - \frac{1}{2}] \end{cases} \quad (37)$$

ahol dt az egy ciklus elvégzéséhez szükséges idő, az első lépésben ez 60ms majd a következő lépésekben a mikrokontroller által mért idő lesz. Kísérleteinkben ez átlagosan 42ms volt.

A coxa neuronok saját aktivitása $s_i^{(c)}$ a (32) egyenletek fordítottjaival számolódik (38). A szervomotor által visszaküldött pozíció (A.1.3. fejezet) beskálázódik a minimum és maximum értékek segítségével a $[0,1]$ intervallumra.

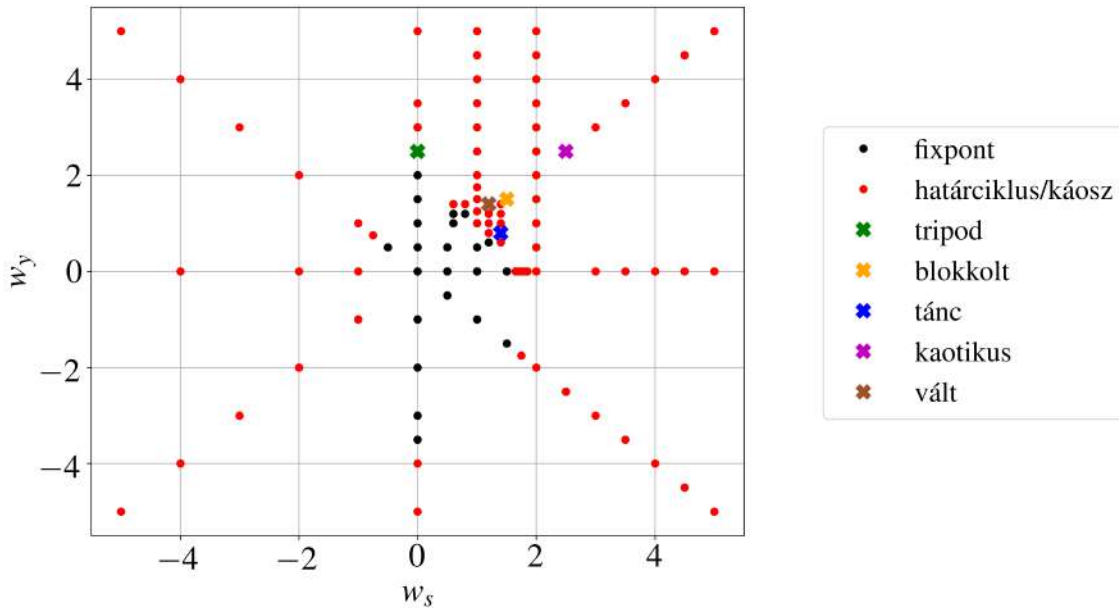
$$s_i^{(c)} = \frac{q_i^{(c)} - \text{MIN } q_i^{(c)}}{\text{MAX } q_i^{(c)} - \text{MIN } q_i^{(c)}} \quad (38)$$

Egy ciklus a mikrokontrolleren a következő lépéseket tartalmazza:

- $x_i^{(c,f)}(t + dt)$, $b_i^{(c,f)}(t + dt)$ értékek kiszámolása

- $q_i^{(c,f)}$ csuklósögek kiszámolása az $y^{(c,f)}$ segítségével
- csuklósögek küldése a szervomotoroknak
- csuklósögek lekérése és a saját aktivitás $s^{(c,f)}$ kiszámolása
- sebességek lekérése és csuklósögekkel együtt küldése a számítógépre soros porton
- dt idő kiszámítása

A paraméterek a 1. táblázat szerint rögzítve vannak, csupán a neuronok aktivitásainak súlyát változtatjuk és a membránpotenciál valamint az aktivációs küszöbérték kezdőértékeit. Ezek vagy egy háromlábú mintázathoz szükséges, 3 láb a másik 3 - al ellentétes fázisban, értékek, vagy pedig véletlenszerűen választjuk őket a $[-5,5]$ intervallumból. A teljes fázistérkép feltérképezése kísérletileg lehetetlen, viszont 154 paraméter párosra megvizsgáltuk a mozgás dinamikáját (lásd 4.5. ábra).

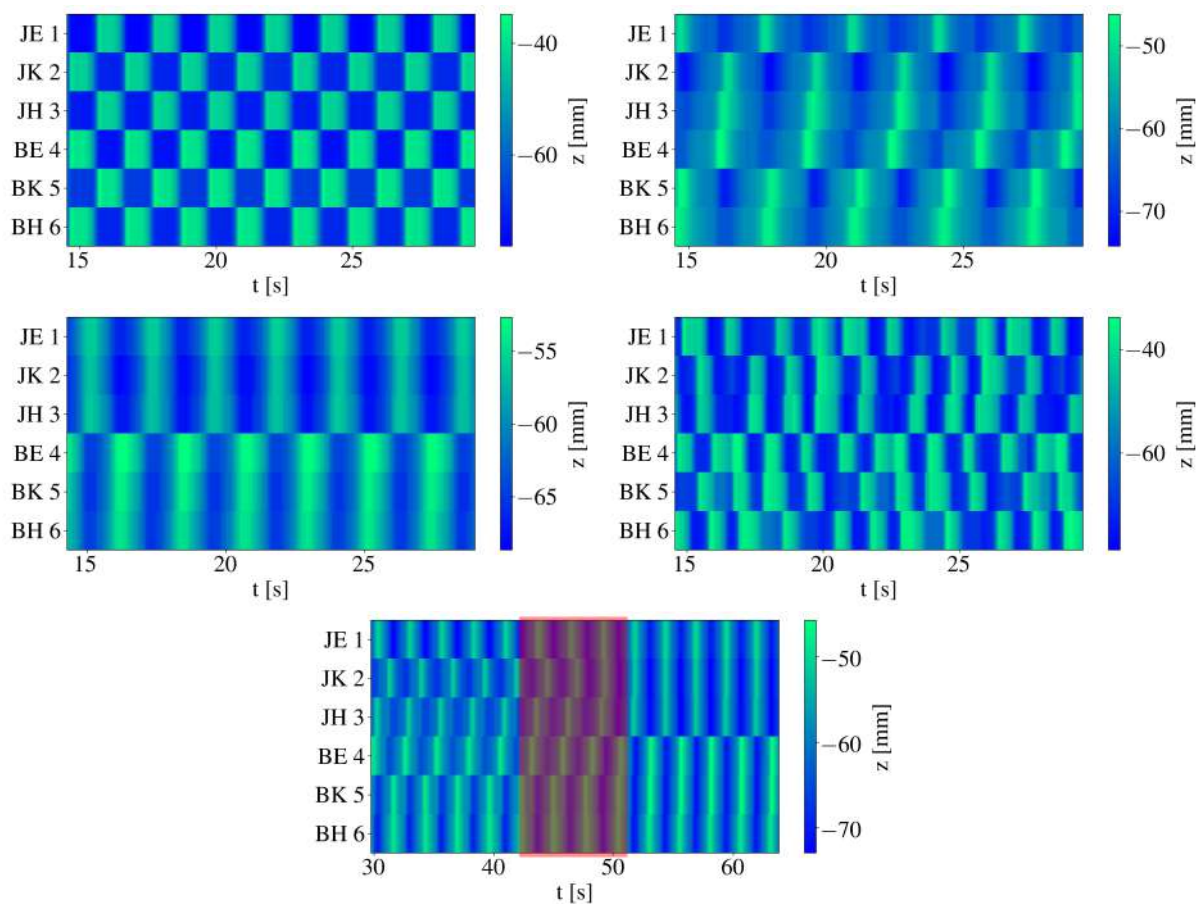


4.5. ábra. Fázistérkép, határciklus dinamika jelenik meg a legtöbb helyen, viszont tényleges előre haladó, lépegető mozgás ritkán alakul ki, mivel a rendszer nagyon érzékeny a kezdeti feltételekre és a változtatható paraméterekre. 2 (w_s, w_y) párost kiemelve (tripod, blokkolt) önszerveződő szabályos dinamika jön létre mellyel a robot számottevő sebességgel halad előre. A tánc és kaotikus viselkedések ugyancsak kivannak emelve, ezek gyakrabban létrejönnek. Bizonyos paraméter párosnál lépésmintázat váltást vélünk felfedezni.

A **tripod** mintázat a már ismert háromlábú váltakozó lépkedés, ebben az esetben $(w_s, w_y) = (0.0, 2.5)$, és fix kezdőértékekről indult az iteráció. A hozzátartozó idősorok a 4.7. ábra első sorában vannak feltüntetve. Szoros kapcsolatot lehet vonni a 4.4. ábrán

szereplő fázisoszcillátorokkal irányított tripod mintázat között, viszont itt a gyorsulás jóval változatosabb. A lépésmintázat a 4.6. ábra első sor bal sarkában található.

A **blokkolt** dinamika elnevezése a középső és hátsó lábak egy oldalon való együtt állásából származik, ebben az esetben $(w_s, w_y) = (1.5, 1.5)$ és véletlenszerű értékekről indult az iteráció. Ez is egy háromlábú mintázat, viszont más sorrendben csoportosulnak a lábak. A hozzátartozó idősorok a 4.7. ábra második sorában vannak feltüntetve, melynek pozíció idősora és sebesség idősora is hasonló a tripod mintázathoz, viszont látszik a lábak egymáshoz képesti eltolódása. Ezzel a lépésmintázattal is halad a robot előre. A lépésmintázat a 4.6. ábra első sor jobb sarkában található.



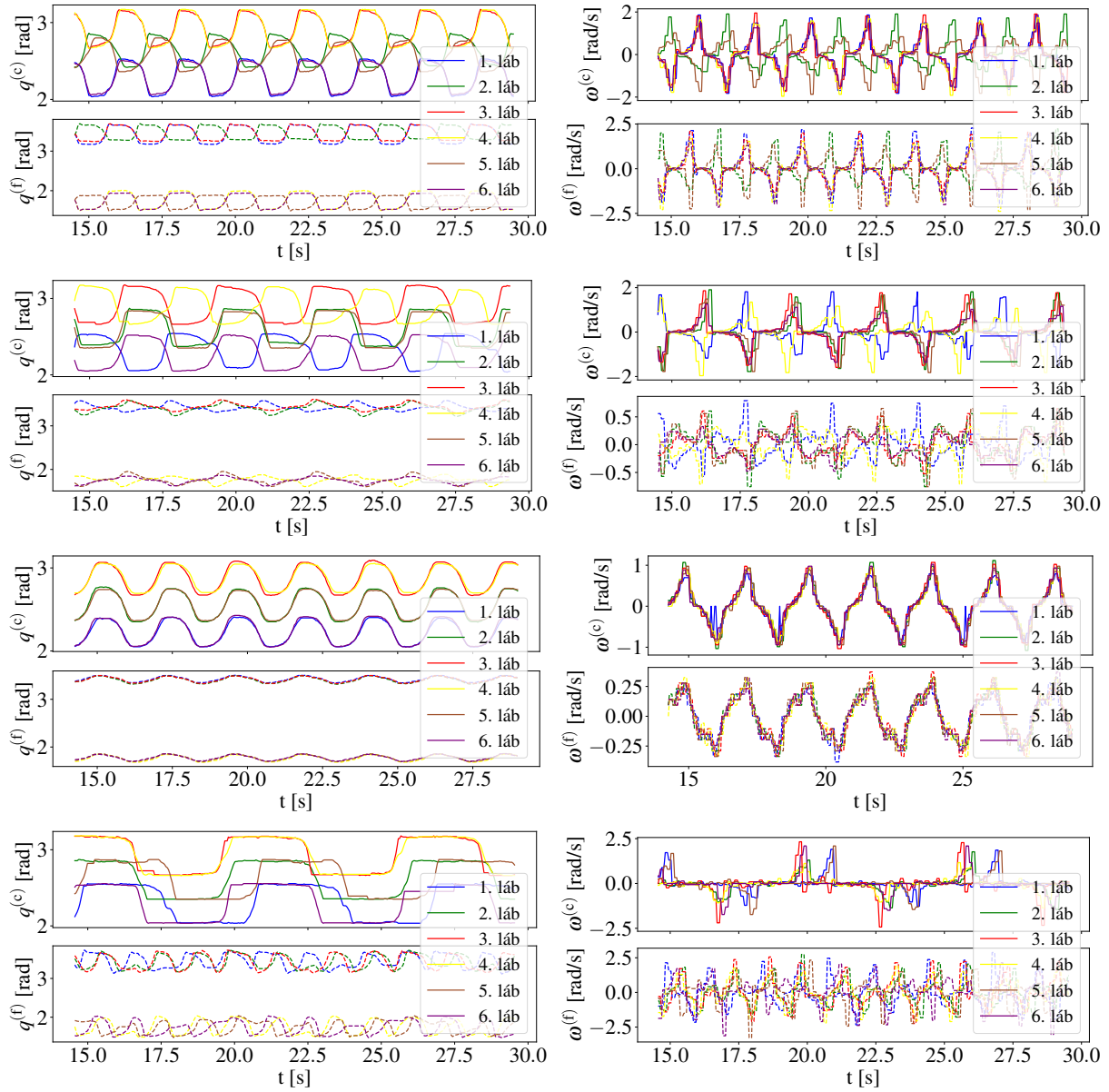
4.6. ábra. Lépésmintázatok a csatolt neuronokkal vezérelt esetben. **Első sor bal:** Tripod lépésmintázat **Első sor jobb:** Blokkolt lépésmintázat. **Második sor bal:** Tánc lépésmintázat. **Második sor jobb:** Kaotikus lépésmintázat. **Harmadik sor:** Lépésmintázat váltás a piros tartományban.

A **tánc** mintázat egy vonzó fázisoszcillátor rendszerhez hasonló, mivel a lábak tökéletesen egyszerre fordulnak, nincsen közöttük fázis különbség, ebben az esetben $(w_s, w_y) = (1.4, 0.8)$

és véletlenszerű kezdőértékekről indult az iteráció. A tánc elnevezés oka, hogy a robot ezen dinamika közben nem halad semerre, az összes effektora a földön van, teste Oz tengely körül oszcillál. A hozzátartozó idősorok a 4.7. ábra harmadik sorában vannak feltüntetve. A lépésmintázat a 4.6. ábra második sorában bal oldalt található.

A **kaotikus** mintázat ahogyan a nevéből is értetendő egy kaotikus viselkedés, ebben az esetben $(w_s, w_y) = (2.5, 2.5)$ és véletlenszerű kezdőértékekről indult az iteráció. A hozzátartozó idősorok a 4.7. ábra alsó sorában vannak feltüntetve. A lépésmintázat a 4.6. ábra második sorában jobb oldalt található.

A $(w_s, w_y) = (1.2, 1.4)$ paraméterpáros kezdetben egy blokkolt típusú haladó mozgást produkál, viszont egy bizonyos idő után mintázat **vált**ást tapasztalunk (lásd 4.6. ábra harmadik sor), a robot átvált a tánc mintázatba. Ennek oka valamilyen külső zaj, jelen esetben a föld egyenetlensége, mivel a blokkolt mintázat egy metastabil állapot, így az attraktor könnyen átkerül egy más határciklusba. Ebben az esetben az iteráció véletlenszerű membrán potenciál és aktivációs küszöbérték kezdőfeltételekről indult.



4.7. ábra. Idősorok a csatolt neuronokkal vezérelt esetben. **Első sor:** Tripod mintázat pozíció és sebesség időszora. **Második sor:** Blokkolt mintázat pozíció és sebesség időszora. **Harmadik sor:** Tánc mintázat pozíció és sebesség időszora. **Negyedik sor:** Kaotikus mintázat pozíció és sebesség időszora.

Következtetések és továbbfejlesztési lehetőségek

Munkánk során a már meglévő elméleti és kísérleti modelleket tanulmányoztuk, majd felépíthettünk egy olyan, oszcilláló neuronok matematikai modelljén alapuló dinamikai rendszert, mely segítségével mozgattuk a hexapod robotot. A szimuláció elvégzését követően gyakorlatban is megvalósítottuk a vezérlést, a robot elméleti és kísérleti lépésmintázatai minőségi egyezést mutatnak.

A numerikus szimuláció és a robot kísérleti vizsgálata során kapott eredmények közötti eltérés többek között annak tudható be, hogy a szimulált robot tömege és tömegeloszlása nem egyezik meg a kísérletekben használt robottal. A csuklókat mozgó szervomotorok, a processzor, a különböző vezetékek és alkatrészek a szimuláció során nem jelennek meg, csupán a robot váza van implementálva. Továbbá, a numerikus szimulációban a fizikai környezet tökéletes, ellenben a valósággal, ahol például a padló kissé egyenetlen. Mindezek mellett a csuklók nyomatékait is eltérő módon állítjuk be, így a felsoroltak összessége miatt tér el a szimulált és valós robot dinamikája.

A modell oszcilláló neuronok hálózatából áll, amelyek egymással teljesen szimmetrikusan vannak csatolva. A neuronok közötti gátló (taszító) csatolás azonban fázisblokkolt állapotokat eredményez. A mérések és szimulációk során kapott lépésmintázatok spontán szimmetriatorés eredményei lehetnek, hiszen a teljesen szimmetrikus csatolás ellenére mégis több különböző lépésmintázatot találtunk a kísérletek elvégzésével.

A robot megépítése, megismerése, a szimuláció megírása, majd a mérési és numerikus adatok feldolgozása során felfedeztük, hogy ha a hexapod robot lábainak két-két csuklósögét egyenként vezéreljük egy oszcilláló neuronnal, és az idegsejtek közötti szinapszist biztosítjuk az aktivitáson keresztül, önszerveződő dinamika alakul ki a különböző csatolási együtthatók és a megfelelő kezdőfeltételek függvényében. A rendszerünkben használt 12 csatolt neuron mindegyike a valós robot egy csuklóját irányítja a speciális szervomotorokon keresztül. A létrjevövő változatos mozgástípusok segíthetnek megérteni a rovarok helyváltoztatás közben produkált lépésmintázatait. Két fő haladó mozgást eredményező lépésmintázatot figyeltünk meg: tripod és blokkolt mintázatokat.

A rendszer továbbfejlesztéseként szeretnénk, ha a jövőben az idegsejtek tulajdonságait leíró egyenletek bemenetében szerepelne majd az egyes femurok aktuális csuklósöge is. A w_k paraméter értékét eddig állandónak vettük minden egyes láb esetén, viszont érdekes lenne vizsgálni azt az esetet is, hogy mi történik akkor, ha a különböző coxa, illetve femur neuronok

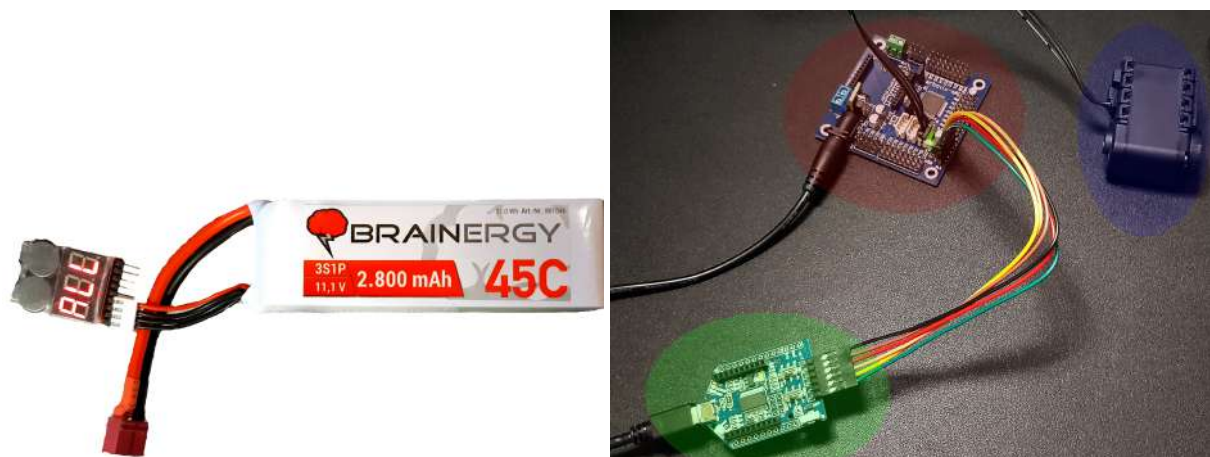
egymásról különböző csatolási együtthatókkal kapnák meg a többiek aktivitását.

A. Függelék

A.1. Bővebben a PhantomX - ról

A.1.1. A robot vezérlése, programozása

A robot irányító központja egy **ArbotiX-M Robocontroller** (lásd A.1. ábra), mely egy sokoldalú vezérlési rendszer Bioloid és Dynamixel aktuátorokhoz, 2010 óta tart a fejlesztése. Arduino kompatibilis így rentgeteg nyílt forráskódú könyvtárhoz és példához ad hozzáférést. A kártya számítógépe egy ATMEGA644p mikrokontroller, 64 kilóbájt írható memóriát és 4 kilóbájt statikus memóriát (SRAM) tartalmaz, mely bőven elegendő ilyen kaliberű projektekhez. A robotvezérlő 3 TTL Dynamixel típusú csatlakozót tartalmaz az okos szervomotorok vezérlésére, továbbá 28 digitális és 8 analóg csatlakozóval rendelkezik hagyományos léptetőmotorok és szenzorok részére. Áramfogyasztása elenyésző az aktuátorokéhoz képest.



A.1. ábra. **Bal:** Brainenergy 11.1V 2800mAh 45C LiPo akkumulátor T konnectorral és hozzácsatlakoztatva a feszültség ellenőrző. **Jobb:** Piros: ArbotiX-M Robocontroller, Zöld: FTDI-USB átalakító, Kék: AX-12A okos szervomotor, az aktuátor áramellátása érdekében a robovezérlő kártya egy külső 12V-os áramforráshoz van csatlakoztatva

A kártya programozása soros porton keresztül történik, melyhez egy FTDI-USB átalakító szükséges. A kártyához az Arduino IDE szoftver segítségével lehet csatlakozni számítógéppel, programozása is itt történik az Arduino sajátos C++ alapú programozási nyelvén. Kezdetben fontos pár alapkönyvtár telepítése mely biztosítja az okos szervomotorokhoz való hozzáférést és irányítást. Ezen csomagok bőséges példa kódot tartalmaznak, a programozás megtanulásához. Egy kód alapstruktúrája 1. kódrészletben tekinthető meg, mely 4 főrésze osztható.

A.1.2. A LiPo akkumulátor

A robot energiafogyasztása hatalmas tud lenni, ha mind a 18 csukló aktív, ezért egy nagy teljesítményű lítium-polimer akkumulátor felel az áramellátásért. Ez egy 3 cellából álló, cellánkénti 3.7V - os feszültségű, maximális 126A áramerősséget leadni képes akkumulátor (lásd A.1. ábra). Tekintve a nagy teljesítmény speciális töltő jár hozzá, mely úgy tölti az akkumulátort, hogy a cellák feszültsége egyensúlyban marad. Nagyon veszélyes, könnyedén kigyúlnak vagy akár fel is robbanhatnak, ezért óvatosan kell vele bánni, nem szabad teljesen lemeríteni (alsó határ $3 \times 3.3V$), legideálisabb töltöttsége a 11V. Használat közben egy vészjelzővel van ellátva, mely folyamatosan monitorizálja a cellák feszültségét.

A.1.3. Az okos szervomotorok

A robot aktuátorai **Dynamixel AX-12A** típusú okosszervomotorok (lásd A.1. ábra), ebből 18 darab található. Okosságuk abban nyilvánul meg, hogy a megszokott egyszerű léptetőmotorokhoz képest jóval precízebb DC motorok, szenzorok és egy mikrokontroller található bennük. Kompakt méretük ellenére nagy nyomatékot és sebességet képesek kifejteni a feladatok végrehajtásához. Visszajelzést tudnak küldeni az aktuális hőmérsékletéről, pozíciójukról, sebességükről, aktuális terhelésükről, az őket ellátó áram feszültségéről és saját memóriával is rendelkeznek. Többnyire jó minőségű műanyagból készülnek mind a külső borítások, mind a belső szerkezetek.

Műszaki szempontból optimális működési feltételek között a következő technikai paraméterekkel jellemezhetőek az AX-12A típusú aktuátorok.

- bemeneti feszültség: 11.1V
- maximális áramfelvétel: 1.5A
- maximális sebesség: 6.17 rad/s
- maximális nyomaték: 1.5 Nm
- maximális kommunikációs sebesség: 1Mbps
- kommunikációs protokoll: Fél duplex, aszinkron

- csatlakozó típusa: TTL, 3 pinű konnektor, mely közül 1 pin a GND, 1 pin VDD (bemeneti feszültség), 1 adat pin a kommunikációhoz

A kommunikáció a robotvezérlő és az aktuátor között adat csomagok küldésével és fogadásával történik. Minden egyes instrukciókat tartalmazó adatcsomag fogadása után az aktuátor válasz adat csomagot küld a kontrollernek. Az adat csomagok összeállítását a 2. táblázat tartalmazza. Minden egyes aktuátorhoz egy ID rendelhető, így azok sorba köthetők (minden Dynamixel típusú szervomotoron két csatlakozási pont található, egy bement és egy kimenet). Így egy robotvezérlő kártyával 254 (8 bit) egymásután kapcsolt aktuátor vezérélhető.

0xFF	0xFF	ID	HOSSZ	PARANCS	PARAMÉTER1	...	PARAMÉTER N
0xFF	0xFF	ID	HOSSZ	ERROR	PARAMÉTER1	...	PARAMÉTER N

2. táblázat. **Felső:** Parancs adatcsomag, a kontroller küldi a szervomotoroknak. **Alsó:** Válasz adatcsomag, a szervomotor küldi a kontrollernek

A szervomotor saját biztonsági rendszerrel van ellátva, ennek célja a belső szerkezet sérüléseinek megakadályozása, például ha menet közben a robot egyik lába beakad akkor egy hagyományos léptetőmotor egyszerűen leégne, mivel az nem tud visszajelzést küldeni a fellépő hibáról. Az aktuátor 7 féle különböző problémát képes észlelni, amelyek a következők: instrukció hiba, túlterhetség, adatcsomag hiba, tartományon kívüli hiba, túlmelegedés, pozíció hiba, bemeneti feszültség hiba. Bármelyik hibajelzés aktívvá válik a szervomotor egyből veszély üzemmódba kapcsol és megszűnik a nyomaték, ezt egy a külső borításon elhelyezett pirosan villogó LED jelzi. Az aktuátor újraindítása, a feszültség ki-be kapcsolása, után az összes hiba kód visszal eredeti inaktív állapotába.

A Dynamixel AX-12A szervomotor két típusú memóriával rendelkezik: EEPROM és RAM, melyeken összesen 50 byte-nyi adat fér el, ez regiszterekre van felosztva. Az EEPROM memória részleg a hosszantartó memória, az aktuátor újraindítása után is ugyanazon regisztereken ugyanazon értékeket fogja tartalmazni. Ide tartozik például a modell szám, az ID, a kommunikáció sebessége, a válasz adatcsomag várakozási ideje, a megengedett elfordulási intervallum, bemeneti feszültség intervalluma stb. A megengedett elfordulási intervallum aktuátoronként beállítható a roboton elfoglalt pozíciójuk alapján, mely fizikailag elérhetetlen pozíciókat határoz meg. A RAM memóriában tárolt regiszterek adatai minden újraindítás után

visszaállnak az eredeti értékükre, melyek az EEPROM-ba vannak eltárolva, ilyen például a cél pozíció, aktuális pozíció, mozgási sebesség, aktuális sebesség, aktuális terhelés, aktuális hőmérséklet, maximálisan kifejthető nyomaték stb. Minden regiszternek különböző sajátos paraméterei vannak, méretük 1 vagy 2 byte, van amelyik csak olvasható, van amelyik csak írható, de vannak kombináltak is. Például a maximálisan kifejthető nyomatékot 2 byte-on lehetséges állítani, mely $[0,1023]$ intervallumból vehet fel értékeket, hasonlóan a cél és aktuális pozícióhoz. Az aktuális sebesség $[0,2047]$ intervallumon értelmezett, ahol a $[0,1023]$ a oZ tengely körüli pozitív iránynak felel meg, míg a $[1024,2047]$ a negatív iránynak.

```

//könyvtárak importálása, globális változók deklarálása, makrók létrehozása
#include <BioloidController.h>
#include <ax12.h>
#define SERVO_NR 18
//-----
//ez a függvény minden indítás után egyszer fut le
void setup(){
    //csuklók számának megadása a bioloid könyvtár számára
    bioloid.setup(SERVO_NR);
    //Soros kommunikáció elindítása, a megadott bit rátával
    Serial.begin(115200);
    //további inicializációk, például paraméterek megadása
}
//-----
//ez a függvény egy végtelen ciklus, a setup() után ez fut a kikapcsolásig
void loop(){
    //a robot tényleges irányítása, pozíciók folyamatos állítása
    //csuklósögek kiolvasása, egyenletek megoldása
}
//-----
//további függvények (opcionális)
int összead(int a, int b){
    return a + b;
}

```

1. kódrészlet. A kísérletben használt, Arduino kompatibilis, robotvezérlő alapprogramja C++ - ban

A.2. A robot numerikus szimulációjának részletezése

- beolvassuk az *URDF* fájlt és felépítjük a robotot(A.2. ábra)

```
#run(git(["clone", "https://github.com/HumaRobotics/phantomx_description.git"]))
hexapod = parse_urdf("phantomx.urdf", floating=true) #beolvassa az urdf-et
mvis_hexapod = MechanismVisualizer(hexapod, URDFVisuals("phantomx.urdf", package_path=["phantomx"])) #vizualizáció
```

A.2. ábra. Kódrészlet: a robot megépítése az *URDF*-file segítségével

- a szimulációs közegben létrehozunk egy vízszintes fizikai felületet, ez tölti be a talaj/padló szerepét(A.3. ábra)
- láthatóvá tesszük a felületet egy vékony téglatest segítségével(A.3. ábra)
- a robot tibiáinak végére kontaktpontokat helyezünk el, ezek a pontok teszik lehetővé, hogy a robot ne essen le a vízszintes felületről, ez tartja meg a robotot a talajon(A.3. ábra)

```
#padló
function add_flat_ground!(mechanism::Mechanism)
    add_environment_primitive!(mechanism, RigidBodyDynamics.HalfSpace3D(Point3D(root_frame(mechanism), 0., 0, 0), FreeVector3D(root_frame(mechanism), 0., 0, 1)))
end

#kontaktpontok
function default_contact_model()
    SoftContactModel(hunt_crossley_hertz(k = 500e3), ViscoelasticCoulombModel(1.0, 20e3, 100.))
end

function add_contacts!(mechanism::Mechanism)
    for body in bodies(mechanism)
        if occursin("tibia", body.name)
            frame = default_frame(body)
            point2 = Point3D(frame, 0, 0.16, 0.020)
            add_contact_point!(body, ContactPoint(point2, default_contact_model()))
        end

        if occursin("alpha", body.name)
            frame = default_frame(body)
            point1 = Point3D(frame, 0, 0, 0.01)
            add_contact_point!(body, ContactPoint(point1, default_contact_model()))
        end
    end
end

end

add_flat_ground!(hexapod) #ez a fizikai felület, kell egy vizuális elem is
#vizuális elem
MeshCat.setobject!(mvis_hexapod.visualizer[:ground], HyperRectangle(Vec(-500, -500, -0.01), Vec(1000, 1000, 0.01)), MeshPhongMaterial(color=RGB(0.99, 0.99, 0.99)))
add_contacts!(hexapod)
```

A.3. ábra. Kódrészlet: a felület elhelyezése, a kontaktpontok robotra rögzítése

- inicializálások: beállítjuk a dinamikai rendszer paramétereit, az egyes neuronok változóinak kezdőértékeit, csuklószőgek kezdőértékeit, szöghatárokat
- feltesszük a robotot a vízszintes felületre, a beállított kezdőpozícióban(A.4. ábra)

```
set_configuration!(mvis_hexapod, configuration(state))
```

A.4. ábra. Kódrészlet: a robot kezdőpozíciójának megadása

- $dt = 10^{-4}$ időlépésenként léptetjük az időt(A.5. ábra)

```
t0 = 0
ts, qs, vs = simulate(state, 80, simple_control!; Δt = 0.0001)
```

A.5. ábra. Kódrészlet: a szimuláció parancssora

- minden időpillanatban megnézzük, hogy a szimuláció kezdete óta, vagy az előző nyomaték megadásától számítva eltelt-e $\Delta t = 10^{-3}$ idő: ha igen, Euler-módszer segítségével számoljuk ki a rendszer változóinak értékeit(A.6. ábra)

```
function (dr::differe)(t)
    if t0 + t - dr.t > 0.001
        dt = t0 + t - dr.t
        yi = y(dr.x, dr.b, dr.a)
        s = []
        for i in 1:6
            push!(s, (state.q[i + 7] - smin[i]) / (smax[i] - smin[i]))
        end

        ysum = sum(yi[1:6])

        for i in 1:6
            yj = ysum - yi[i]
            dr.x[i] = dr.x[i] + (-1 * dr.x[i] + dr.wS * s[i] + dr.wy * yi[i] + dr.wk * yj) / dr.tx * dt
            dr.b[i] = dr.b[i] + (yi[i] - 0.5) / dr.tb * dt

            dr.st[i] = yi[i] * (smax[i] - smin[i]) + smin[i]
        end

        for i in 7:12
            dr.x[i] = dr.x[i] + (-1 * dr.x[i] + dr.wy * yi[i] + dr.wk * yi[i - 6]) / (dr.tx / 2) * dt
            dr.b[i] = dr.b[i] + (yi[i] - 0.5) / dr.tb * dt

            dr.st[i] = yi[i] * (smax[end] - smin[end]) + smin[end]
        end

        dr.t = t0 + t
        write_in_file("2525nagymatrix.txt", dr.t, state.q, state.v, dr.x, dr.b, yi, dr.st)
    end
end
```

A.6. ábra. Kódrészlet: a változók kiszámolása Euler-módszerrel, kiírás

- mentjük az adott pillanatban a szimulációban mérhető valamennyi mennyiséget: az időt, csuklósögeket, pillanatnyi aktivitásokat, aktivációs küszöbértékeket és a tibiák effektorjainak magasságát a talajtól számítva, a robot testének magasságát a talajtól számítva (A.6. ábra)
- megadjuk a csuklók célpozícióit, majd a nyomatékokat (A.7. ábra)

```

function simple_control!(torques::AbstractVector, t, state::MechanismState)

    r(t)
    q0t = -0.68
    Kp = 15
    Kd = 2 * sqrt(Kp)

    for i in 1:6
        torques[i + 6] = (r.st[i] - state.q[i + 7]) * Kp - state.v[i + 6] * Kd
    end

    for i in 7:12
        torques[i + 6] = (r.st[i] - state.q[i + 7]) * Kp - state.v[i + 6] * Kd
    end

    for i in 19:24
        torques[i] = (q0t - state.q[i + 1]) * Kp - state.v[i] * Kd
    end
end

```

A.7. ábra. Kódrészlet: nyomatékok megadása a robot csuklói számára.

Hivatkozások

- [1] Peter Corke. *Robotics, Vision and Control, Fundamental Algorithms in MATLAB*. Springer International Publishing AG, 2017.
- [2] Edward Gerjuoy. *Complex and adaptive dynamical systems*.
- [3] Pablo Kaluza és Teodor Cioaca. „Phase oscillator neuralnetwork as artificial central pattern generator for robots”. *Neurocomputing* (2012).
- [4] Dai Owaki és mások. „A Minimal Model Describing Hexapedal Interlimb Coordination: The Tegotae-Based Approach”. *Frontiers in Neurorobotics* (2017).
- [5] Dai Owaki és mások. „Simple robot suggests physical interlimb communication is essential for quadruped walking”. *J R Soc Interface* (2012).
- [6] Bulcsú Sándor. „The world of dynamical systems: multistability, neural dynamics and robotic locomotion”. *PhD thesis, Goethe University Frankfurt* (2017).
- [7] Bulcsú Sándor és mások. „Kick Control: Using the Attracting States Arising Within the Sensorimotor Loop of Self-Organized Robots as Motor Primitives”. *Frontiers in Neurorobotics* (2018).
- [8] Steven H. Strogatz. *Nonlinear Dynamics and chaos*. Perseus Books Publishing, 1994.
- [9] Tamás Tél és Márton Gruiz. „Kaotikus dinamika”. *Nemzeti tankönyvkiadó, Budapest* (2002).
- [10] TrossenRobotics. *Phantom X Getting started guide*. URL: <https://learn.trossenrobotics.com/10-interbotix/crawlers/phantomx-hexapod/172-phantomx-hexapod-metal-assembly-guide.html>.
- [11] Hendrik Wernecke, Bulcsú Sándor és Claudius Gros. „Attractor metadynamics in terms of target points in slow-fast systems: adiabatic versus symmetry protected flow in a recurrent neural network”. *Journal of physics Communications* (2018).