

**SAPIENTIA ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR
AUTOMATIKA ÉS ALKALMAZOTT INFORMATIKA SZAK**

**ACROBOT SZABÁLYOZÁSA ÉS
GYAKORLATI MEGVALÓSÍTÁSA
TDK DOLGOZAT**

Témavezető:

Dr. Dávid László, egyetemi tanár

Hallgató:

Lukács Szabolcs

2023

Tartalomjegyzék

KIVONAT.....	3
1 BEVEZETŐ	4
2 GYAKORLATI MEGVALÓSÍTÁSOK	5
3 AZ ACROBOT DINAMIKÁJA	6
3.1 MOZGÁSEGYENLETEK	7
4 SZABÁLYZÓ	9
4.1 LINEÁRIS SZABÁLYZÓ	10
4.1.1 Linearizált modell	10
4.1.2 Diszkrétizálás	10
4.1.3 Szabályozhatóság vizsgálata	11
4.1.4 DLQR szabályzó	12
4.2 ENERGIA ALAPÚ SZABÁLYOZÁS	14
4.3 SWITCH LOGIC	15
5 NEURONHÁLÓ.....	16
5.1 EGYRÉTEGŰ ELŐRECSATOLT NEURONHÁLÓ FELÉPÍTÉSE	17
5.2 MEGVALÓSÍTÁS.....	17
5.2.1 DLQR	18
5.2.2 Swing up	20
6 MODELLEZÉS, SZIMULÁCIÓ	21
6.1 A MODELL FELÉPÍTÉSE	23
6.1.1 Fizikai elemek.....	23
6.1.2 Szabályzó.....	24
6.1.3 Szervomotor.....	25
6.1.4 Forgóérzékelő	26
6.2 EREDMÉNYEK.....	27
7 GYAKORLATI MEGVALÓSÍTÁS	31
7.1 ACROBOT FELÉPÍTÉSE	33
7.2 ÁLLAPOTOK MÉRÉSE	33
7.3 VEZÉRLŐJEL SZÁMÍTÁS	35
7.4 SZERVO MOTOR VEZÉRLÉS	35
7.5 KOMMUNIKÁCIÓ	36
8 GYAKORLATI EREDMÉNYEK.....	37
9 KÖVETKEZTETÉSEK, JÖVŐBENI CÉLOK.....	38
10 IRODALOMJEGYZÉK.....	39
11 ÁBRAJEGYZÉK.....	40

Kivonat

A humanoid típusú robotok, valamint drónok irányításának uralkodó irányzata a rendszer dinamikáját kompenzáló, lineáris, linearizált vagy esetleg állapotfüggő szabályozási módszerek használata. Ezek nagy hátránya, hogy a szabályozási stratégiák nem használják ki a rendszer természetes dinamikája adta lehetőségeket. Az alulaktuált rendszerek vizsgálatakor olyan szabályozók tervezésére szoktunk törekedni, amelyek a rendszer dinamikáját felhasználva gyorsabb, energia-hatékonyabb és robusztusabb szabályozást tesznek lehetővé.

A dolgozatom témája egy olyan nemlineáris, alulaktuált dinamikus rendszer szabályozásának megvalósítása, nevezetesen egy „ACROBOT”, amelyet később laborgyakorlatként is lehet majd használni. Az alulaktuált rendszerek szabályozása, másfajta szabályozást igényel, mint a hagyományos rendszereké. Ilyen lehet a CLF (Control Lyapunov Függvény) valamint egy előre meghatározott optimális pályakövetés. Célom egy olyan kísérleti eszköz elkészítése, amin be lehet mutatni egy ilyen szabályozást.

Az tanulmányozott dinamikus rendszer az „acrobot”, ami egy inverz dupla inga, de csak a 2. csuklóban van beavatkozó eszköz.

A szabályozás fő célja eljuttatni az acrobotot az alsó, stabil egyensúlyi pontjáról, a felső, instabil pontjára, és ott stabilizálni felhasználva egy CLF valamint egy DLQR szabályozási algoritmust. A rendszert pontosan modellezve, és szimulálva megkaptam a második szegmens előírt szögét. Ahhoz, hogy ezt on-line azaz valós időben lehessen implementálni egy neuronálót használtam fel, amely biztosítja, hogy valós idejű állapotok alapján kapjuk meg az előírt szöget.

A megvalósítás során egy szervomotort használok a második szegmens mozgására, és egy inkrementális enkódert, a szögek és szögsebességek mérésére. A jelek feldolgozása és a beavatkozó jel kiszámítása egy hálózatra csatolt arduino fejlesztőlapokkal történik.

1 Bevezető

Az acrobot egy nemlineáris alulaktuált rendszer. Akkor nevezünk egy rendszert alulaktuáltként, ha szabadságfokainak a száma nagyobb, mint a rendszer független beavatkozójának száma. Ez azt jelenti, hogy a rendszernek van néhány "passzív" foka, amelyet nem lehet közvetlenül irányítani az elérhető aktuátorokkal, így nagyon fontos szerepet kap ezen rendszerek szabályozásában a rendszer belső dinamikája.[1] Az alulaktuált rendszerek fontossága abban rejlik, hogy a való életben is számos rendszer alulaktuált. Az alulaktuált rendszerek egyre gyakrabban találhatók meg a modern robotikában, ilyenek például a sétáló robotok, segway-ek, bipedal robotok, űrbéli robotok, flexibilis robotok, a legtöbb emberi mozgást utánózni próbáló robot.[2] Az egyik legegyszerűbb ilyen rendszer, ezen a területen már alappéldának számító acrobot, ami az emberi járás egyik legegyszerűbb modellje.

Az acrobot esetében 2 szabadságfokról beszélhetünk, és egy beavatkozóról. Ha a beavatkozó az első csuklóban helyezkedik el, akkor pendubotnak nevezzük, ha a másodikban, akkor acrobot. Ha egyik csuklóban sincs beavatkozó, abban az esetben dupla inverz ingáról beszélünk.

Az acrobot esetében a klasszikus szabályozás a robot stabilizálását jelenti a fenti egyensúlyi pontja körül. Általában ezt két részre lehet osztani, az ú.n. "Swing-up" és "Balancing" fázisra[3]. A swing-up fázisban a cél a második szegmens eljuttatása az equilibrium pont környezetébe, a stabilizáló fázis feladata ott tartani és az instabil pontja körül stabilizálni a rendszert.

A dolgozatomban célja egy egyszerű acrobot fizikai megvalósítása és annak szabályozása, amely később használható oktatási célra, másfajta szabályozások tanulmányozására. Az acrobot dinamikája eléggé komplex ahhoz, hogy sokféle szabályozási technikát tanulmányozni lehessen rajta.

2 Gyakorlati megvalósítások

Egy acrobot fizikai megvalósítása a legtöbb esetben nehézkes. Két fő részre oszthatók: “Directly driven acrobot” (DDA) és „Remotely driven acrobot” (RDA).[4]

RDA (Remotely Driven Acrobot)

A második szegmens meghajtása egy vezérműszíjon keresztül van meghajtva, így elérve hogy az aktuátor nem mozog együtt az egész rendszerrel, hanem egy fix ponthoz rögzítve tudja vezérelni a második szegmenst. Előnye hogy a motor súlya nem terheli a rendszert, így a szegmensek könnyebb anyagból is készíthetők, és nagyobb nyomaték is kifejthető, az áttételnek köszönhetően. Hátránya a fizikai megvalósításban a bonyolultabb felépítés, az áttételek megoldása és stabilan tartása a robot teljes mozgási tartományán belül. A gyakorlatban ez az elterjedtebb modell az acrobot bemutatására.

DDA (Directly Driven Acrobot)

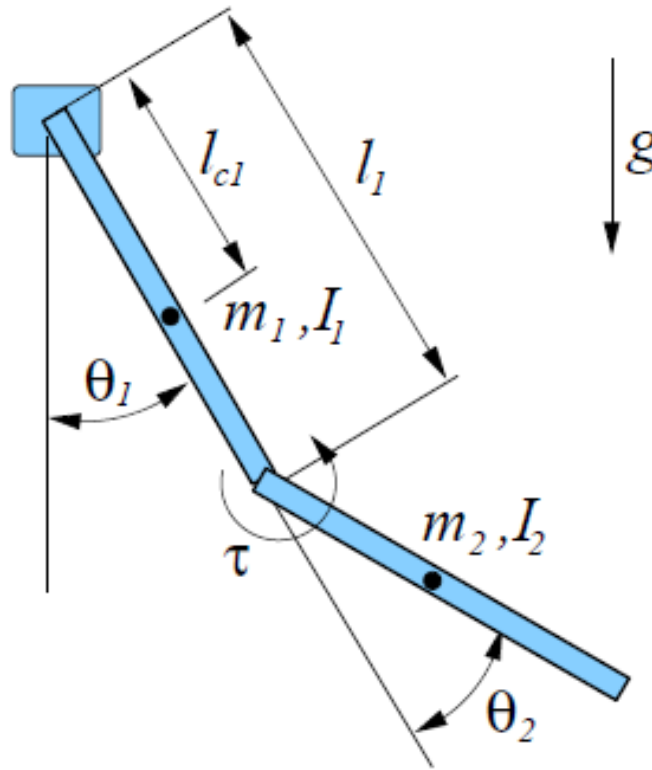
A directly driven acrobot esetében közvetlenül van meghajtva a második szegmens. Ebben az esetben a motor, vagy a beavatkozó közvetlenül hajtja meg a második szegmenst. Kevésbé elterjedt, mivel itt korlátozottak a rendszer paraméterei. Ha túl nagy a motor tömege, akkor a második kar végén lévő súlyt is növelni kell, hogy fel lehessen tornáztatni az inverz állapotába. Ha viszont nagyobb az alsó súly, az általában nehezebb, drágább motorhoz vezet, mivel nagyobb nyomaték lesz szükséges ennek mozgatására. Előnye viszont, hogy egyszerűbb fizikai felépítéssel is kivitelezhető.

A fenti előnyöket és hátrányokat figyelembe véve a DDA mellett döntöttem, mivel manapság már elérhetőek kellően könnyű és viszonylag nagy maximális kifejthető nyomatékkal rendelkező szervomotorok. Így kevésbé komplex mechanikai kivitelezésben is megvalósítható a rendszer.

3 Az acrobot dinamikája

Az acrobot egy két szabadságfokkal rendelkező, úgynevezett dupla inga. A felfüggesztéshez közelebbi csuklót sokszor nevezik vállnak, a távolabbit, ahol az aktuátor jelen van, könyöknek.[5]

A szabályozási nehézséget az adja, hogy figyelembe kell venni az állapotfüggő kapcsolatot az aktív és a passzív szabadságfokok közt.



3.1. ábra. Az acrobot modellje [5]

A szegmensek tehetetlenségi nyomatéka:[6]

$$I = \frac{1}{12}ml^2$$

(1)

τ	nyomaték
l_1	Első szegmens hossza
l_2	Második szegmens hossza
lc_1	Első szegmens tömegközéppontjának hossza
lc_2	Második tömegközéppontjának hossza
m_1	Első szegmens tömege
m_2	Második szegmens tömege
g	Gravitációs állandó
θ_1	Első szegmens relatív elfordulása
θ_2	Második szegmens relatív elfordulása
I_1	Első szegmens inerciája
I_2	Második szegmens inerciája

1. táblázat. Jelmagyarázat

3.1 Mozgásegyenletek

A mozgásegyenletek a könnyebb átláthatóság kedvéért a Lagrange mechanika alapján vannak levezetve.[5]

x_1 x_2 jelöli az abszolút helyét az első és a második szegmens tömegközéppontjának.

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} lc_1 * \sin(q_1) \\ -lc_1 * \cos(q_1) \end{pmatrix} \quad x_2 = x_1 + \begin{pmatrix} l_2 * \sin(q_1 + q_2) \\ -l_2 * \cos(q_1 + q_2) \end{pmatrix} \quad (2)$$

Így az acrobot mozgási energiája P , ami két részből tevődik össze.

$$P_1 = \frac{1}{2} * I_1 * \dot{q}_1^2 \quad (3)$$

$$\begin{aligned} P_2 = & \frac{1}{2} * (m_2 * l_1^2 + I_2 + 2 * m_2 * l_1 * lc_2 * \cos(q_2)) * \dot{q}_1^2 + \frac{1}{2} * I_2 * \dot{q}_1^2 \\ & + (I_2 + m_2 * l_1 * lc_2 * \cos(q_2)) * \dot{q}_1^2 * \dot{q}_2^2 \end{aligned} \quad (4)$$

Mozgási energia

$$P = P_1 + P_2 \quad (5)$$

Potenciális energia

$$K = -m_1 * g * l c_1 * \cos(q_1) - m_2 * g * (l_1 * c_1 + l_2 * \cos(q_1 + q_2)) \quad (6)$$

A Lagrange függvény

$$L = P - K \quad (7)$$

Kiszámítva a parciális deriváltakat: [7]

Majd behelyettesítve az Euler-Lagrange-egyenletbe(8),[8]

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}} \right) - \left(\frac{\partial L}{\partial q} \right) = \tau \quad (8)$$

megkapjuk a mozgási egyenleteket. Ezeket rendszerezve kapjuk az acrobot mozgását leíró mátrixokat.[5]

$$M(q) = \begin{bmatrix} I_1 + I_2 + m_2 l_1^2 + 2m_2 l_1 l c_2 \cos(q_2) & I_2 + m_2 l_1 l c_2 \cos(q_2) \\ I_2 + m_2 l_1 l c_2 \cos(q_2) & I_2 \end{bmatrix} \quad (9)$$

$$C(q, \dot{q}) = \begin{bmatrix} -2m_2 l_1 l c_2 \sin(q_2) \dot{q}_2 & -m_2 l_1 l c_2 \sin(q_2) \dot{q}_2 \\ m_2 l_1 l c_2 \sin(q_2) \dot{q}_1 & 0 \end{bmatrix} \quad (10)$$

$$G(q) = \begin{bmatrix} m_1 g l c_1 \sin(q_1) - m_2 g (l_1 \sin(q_1) + l c_2 \sin(q_1 + q_2)) \\ m_2 g l c_2 \sin(q_1 + q_2) \end{bmatrix} \quad (11)$$

Ahol:

M - tehetetlenségi mátrix

C - Centrifugális és a Coriolis erők vektora

G - a gravitációs erő hatása

Összeségében a következőképpen írható fel az acrobot dinamikus modellje:

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = Bu \quad (12)$$

Ahol acrobot esetében

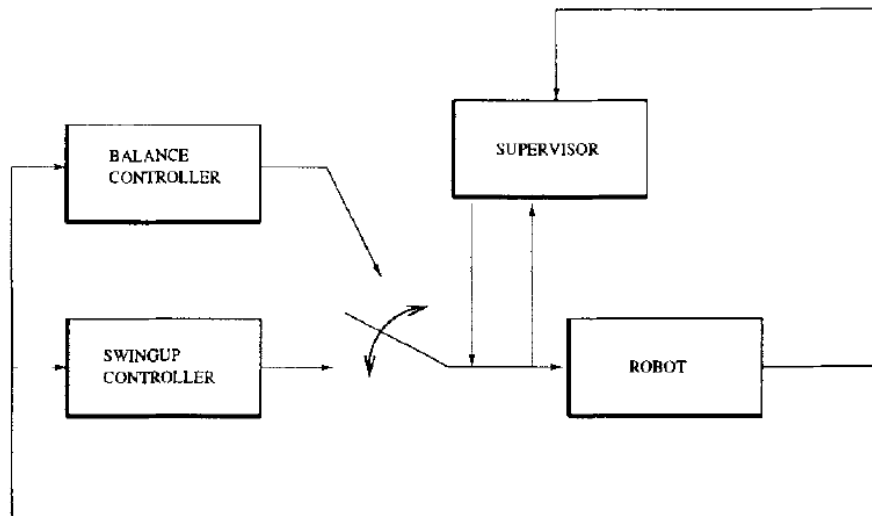
$$B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

(13)

Mivel csak a második az aktív csukló.

4 Szabályzó

A rendszer nemlineáris jellegéből adódóan a szabályzási feladatot nem lehet megoldani csak egy lineáris szabályzóval. Ennek megoldására többféle módszer is létezik, például olyan szabályzási algoritmusok, amelyek lefedik a teljes, globális mozgási tartományt, ilyenek például a neuronhálózval megoldott szabályozások, részleges linearizálást alkalmazó szabályzások, vagy két egymástól különböző szabályzó hibrid alkalmazása.



4.1. ábra. A szabályzás tömbvázlata[9, p. 12]

Az elv az, hogy az acrobotot eljuttatjuk az elvárt egyensúlyi pontja közelébe, úgy, hogy megfelelő állapotok szerint érje el azt. Azután egy másik kontrollert veszi át az irányítást, amely ott tartja, stabilizálja az egyensúlyi pontja körül.

4.1 Lineáris szabályzó

Ahhoz, hogy alkalmazni tudjuk az LQR szabályzót, linearizálni kell a rendszert a felső egyensúlyi állapota körül (UEP). Az UEP közvetlen környezetében sikeresen fog tudni működni a lineáris szabályzó, mivel az eltérések a nemlineáris és a lineáris rendszer között minimálisak.[2]

UEP pont:

$$q_1 = p_i, \quad q_2 = 0, \quad \dot{q}_1 = 0, \quad \dot{q}_2 = 0 \quad (14)$$

4.1.1 Linearizált modell

A linearizált modellt a következő alakban tudjuk felírni.

$$\dot{x} = A * x + B * u \quad (15)$$

Ahol az A mátrix a rendszermátrix és B mátrix a bemeneti mátrix.

Linearizálva a rendszert a [10] szerint és behelyettesítve a valós paramétereket:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 28.0744 & -34.4596 & 0 & 0 \\ -31.7716 & 95.9008 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ -168.3284 \\ 407.3343 \end{bmatrix} \quad (16)$$

A C 4x4 egységmátrix és D 4x1 méretű null-mátrix.

4.1.2 Diszkretizálás

Mivel a valóságban diszkrét értékekkel kell megvalósítani a rendszer szabályozását, ezért diszkretizáltam a rendszert. A diszkrét rendszerre való áttérést a nulladrendű tartó módszerrel végeztem. Mintavételezési idő $T_s = 20$ ms.

A kapott diszkrét rendszer:

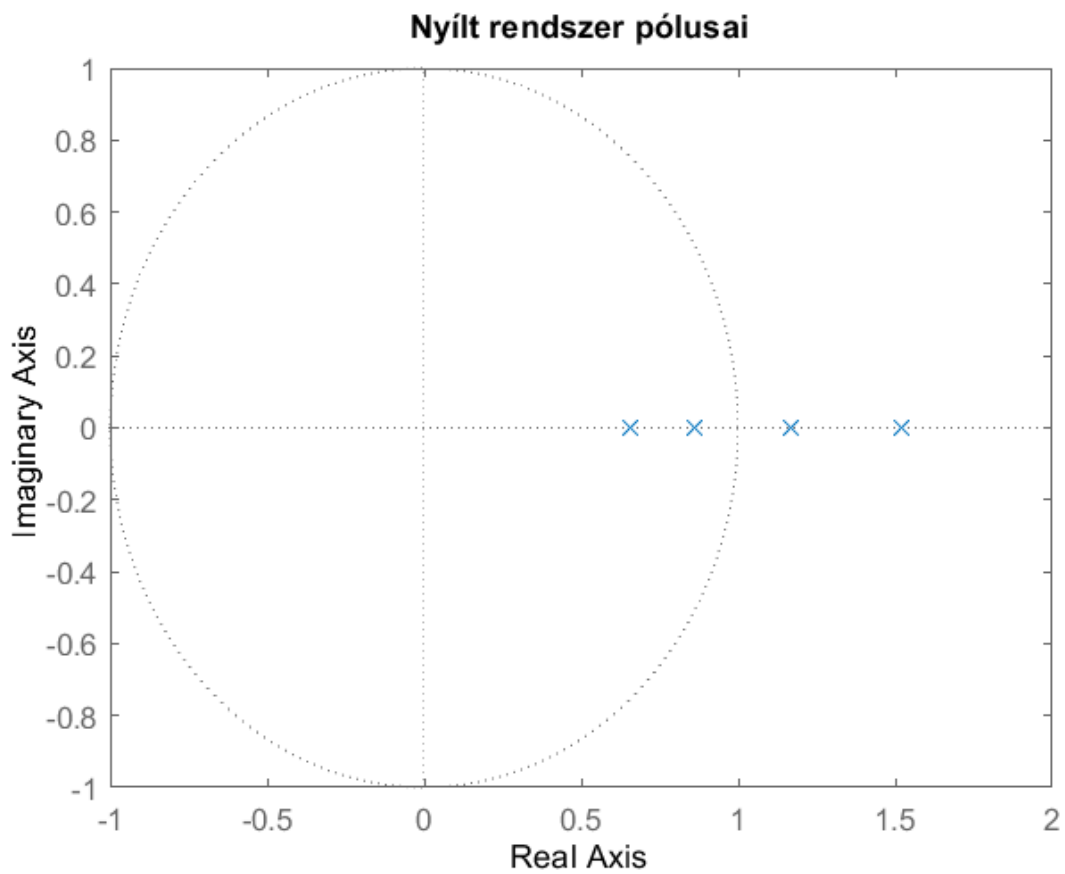
$$A = \begin{bmatrix} 1.02 & -0.02 & 0.04 & 0 \\ -0.02 & 1.078 & 0 & 0.04 \\ 1.14 & -1.42 & 1.02 & -0.02 \\ -1.31 & 3.94 & -0.02 & 1.07 \end{bmatrix} \quad B = \begin{bmatrix} -0.13 \\ 0.33 \\ -6.93 \\ 16.77 \end{bmatrix}$$

(17)

4.1.3 Szabályozhatóság vizsgálata

A nyílt rendszer nem stabil, ami várható volt, mert az instabil egyensúlyi állapota körül linearizáltuk.

A rendszer pólusai közül kettő is az egységsugarú körön kívül van.



4.2. ábra. A nyílt rendszer instabil pólusai

A rendszer teljesen irányítható, ha az irányíthatósági mátrixának rangja egyenlő a rendszermátrix n -el, tudva hogy $A_{n \times n}$ méretű.[11, p. 145]

A rendszer rangja 4, ugyan úgy, mint az irányíthatósági mátrixé tehát irányítható a rendszer.

4.1.4 DLQR szabályzó

A munkapont körül linearizált diszkrét rendszerre lehet alkalmazni a discrete linear quadratic regulator algoritmust. Ehhez meg kell oldani a diszkrét Ricatti egyenletet[12]. A diszkrét Ricatti egyenlet megoldását (P mátrix) a diszkrét Schur módszert alkalmazva kaptam meg.

Ezután a K erősítésvektor megkaphatjuk

$$K = ((R + B'PB)B'PA)^{-1} \quad (18)$$

Ahol Q 4×4 méretű egységmátrix, $R = 100$ választva.

$$K = [-9.21 \quad -3.02 \quad -2.36 \quad -0.91] \quad (19)$$

A szabályzójel számítása:

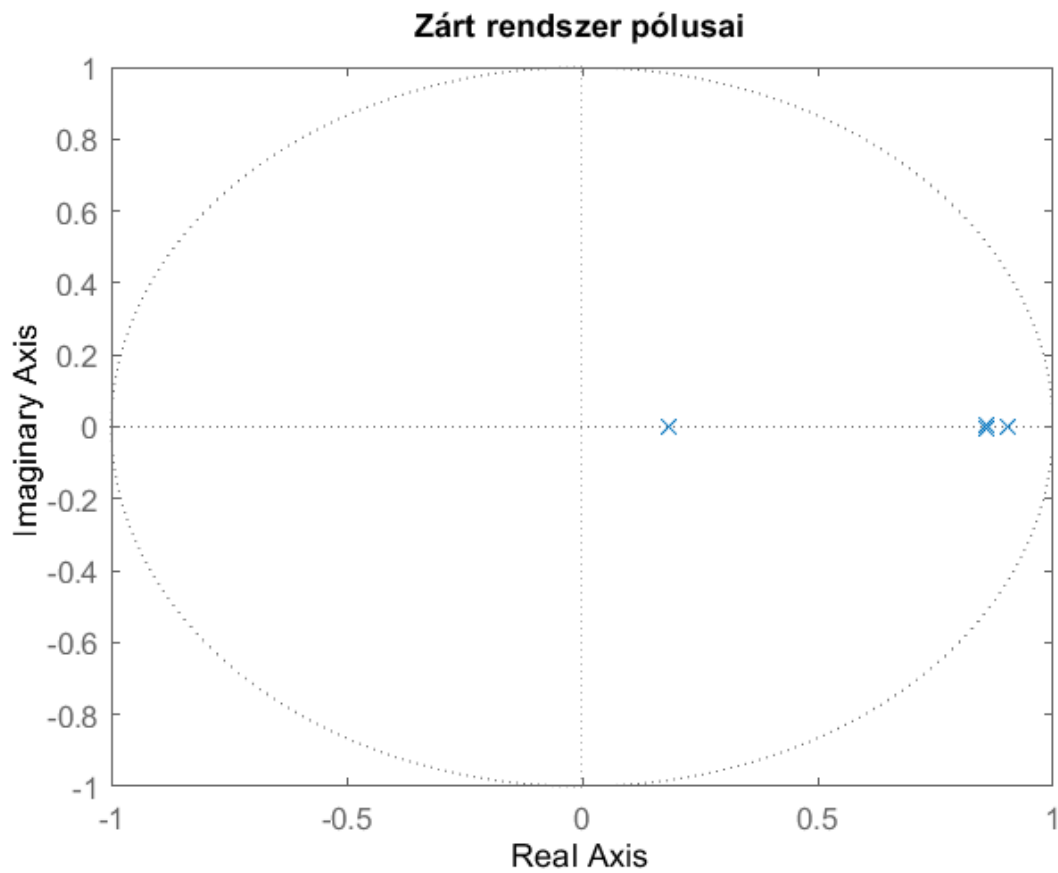
$$u = -K * x \quad (20)$$

Ahol x állapotokat jelöli.

$$x = [q_1 \quad q_2 \quad \dot{q}_1 \quad \dot{q}_2] \quad (21)$$

A zárt szabályzási kört megkaphatjuk, ha az A rendszermátrixból kivonjuk a K erősítésvektor által súlyozott bementi B mátrixot.

A szabályozott rendszer sajátértékei, vagyis pólusai az egység sugarú körön belül vannak, tehát látható, hogy a rendszer stabil.

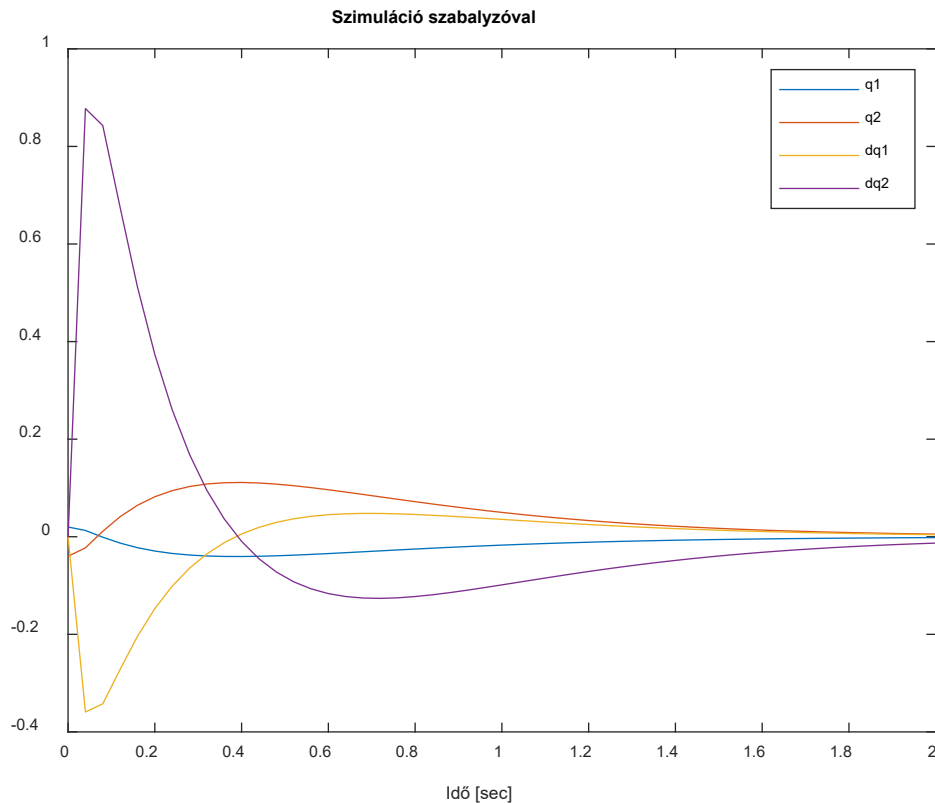


4.3. ábra. A zárt rendszer stabil pólusai

Szimulálva a rendszert látható, hogy az állapotokat a 0-ba vezeti a rendszer.(4.4. ábra.)

A szimulációhoz használt kezdőállapot radiánban:

$$x_0 = [0.02 \quad -0.04 \quad 0 \quad 0]$$



4.4. ábra. A rendszer állapotai szimuláció közben

4.2 Energia alapú szabályozás

Az energia alapú szabályozás elve, hogy a rendszerbe energiát pumpálunk, azzal, hogy a második szegmenst az első szegmens szögelfordulása és gyorsulása alapján kilengessük. Lényegében be kell lengetni a rendszert, addig amíg eléri a felső egyensúlyi pontját. Ezzel a módszerrel folyamatosan mozgási energiát viszünk be a rendszerbe.[3] A feladat, hogy a kezdeti $[0 \ 0 \ 0 \ 0]$ állapotából eljuttassuk a $[\pi \ 0 \ 0 \ 0]$ állapotba.

Az alapötlet az, hogy a belengetjük a második szegmenst egy rögzített $\pm\alpha$ szög között, akkor amikor az fázisban van, úgy, hogy az amplitúdója az első szegmens kilengésének minden lengéssel növekedjen. A legegyszerűbb stratégia akkor lengetni a szegmenst, amikor a $\dot{q}_1$ null-átmenetben van.

Ennek megvalósítására ilyen függvény például a szaturációs függvény[3]. Ennek egy kifinomultabb változata az arcus tangens függvény.

$$q_2^d = (2\alpha/\pi) \tan^{-1}(\dot{q}_1) \quad (23)$$

Ezt vezérlőjelként kiadva k_p és k_d erősítési tényezőkkel végezzük, ahol a k_p az ideális pálya és az aktuális pálya közti eltérést erősíti, k_d pedig a gyorsulást.[3]

$$u = k_p(q_2^d - q_2) - k_d\dot{q}_2 \quad (24)$$

Ahhoz, hogy a belengetés időben is hatékony legyen, és a felső inverz pontot is közel nulla sebességgel érje el a következő értékeket alkalmaztam a szabályzáshoz:

$$\alpha = 1 [rad], \quad k_p = 450 \quad k_d = 0.3 \quad (25)$$

4.3 Switch logic

A két szabályzó között csak a megfelelő állapotban szabad váltani. Fontos hogy az UEP pontot lehetőleg minimális szöggyorsulás mellett érje el. A feltételt ahol a szabályzó átválthat a legtöbb esetben empirikus módon határozzák meg. A következő feltételt vettem alapul.[13, p. 15]

$$|q_1| + |q_2| + 0.1|\dot{q}_1| + 0.1|\dot{q}_2| < \xi \quad (26)$$

Sokszor azonban az átváltás pillanatában a lineáris szabályzó kezdeti beavatkozó jele nagyon nagy lehet, amit a fizikai rendszerben már nem lehet megvalósítani. Ennek elkerülésére én figyelembe vettem a lineáris szabályzó jelét is. Ahhoz, hogy a rendszer sikeresen tudjon váltani a két szabályzó között a DLQR szabályzó jele is egy határon belülre kell kerüljön. A feltétel a váltásra így a következőképpen alakult

$$|q_1| + |q_2| + 0.1|\dot{q}_1| + 0.1|\dot{q}_2| + |u| < \xi \quad (27)$$

Ahol az u a lineáris szabályzó által kiszámított szabályzójel. Az ξ értéke, amelynél a szabályzó sikeresen átválthat 1.6.

Fontos, hogy az átváltás után megtörténhet, hogy a fenti feltétel újra nem teljesül kilengések miatt, ügyelni kell arra, hogy ne váltsunk vissza a feltornáztatáshoz használt szabályzóra. Különbséget kell tenni a sikertelen szabályozás és az ideiglenes kilengések között úgy, hogyha sikertelen, akkor térjen vissza az acrobot feltornáztatásához. Ehhez egy felsőbb limitet választottam $\xi = 15$. Ezt meghaladva a váltó logika sikertelennek veszi a lineáris tartományban a szabályzást és visszavált az első fázisban használt szabályzóra.

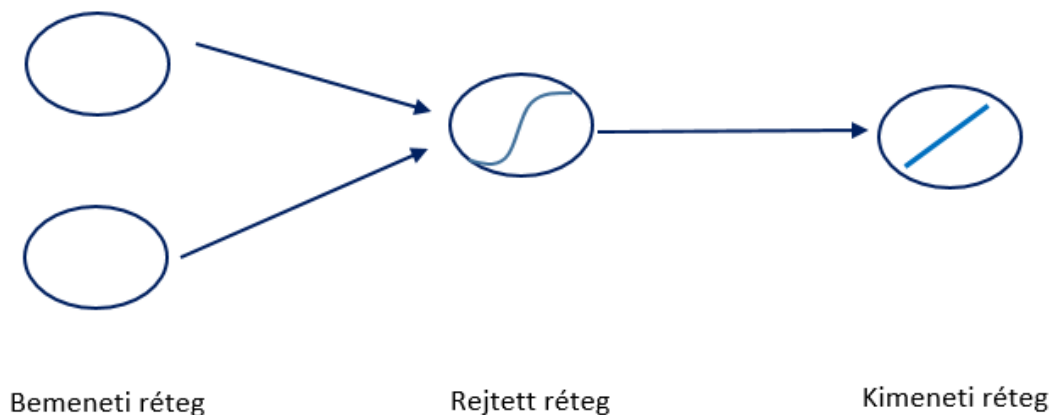
5 Neuronháló

Mivel a szabályozást valós környezetben is meg kell valósítani, így törekedtem az egyszerű kivitelezésre, olyan számításokra, amelyeket egy mikroprocesszoron is el lehet végezni. A szervomotornak egy előírt szöget kell megadni és nem nyomatékot, tehát a szabályzó jel kiszámítása után (ami egy erő, nyomatékban kifejezve) ahhoz, hogy megkapjam az előírt szöget, valós időben kéne szimulálni a rendszert, kiintegrálni a mozgási egyenleteket. A szimuláció alapján kigenerált szabályzó jelet kiadva a valós rendszernek csak off-line irányítást kapnánk, mivel nincs visszacsatolás, nem vesszük figyelembe az állapotokat. Ahhoz, hogy on-line szabályozás legyen, és elkerüljem a nagy számítási igényt az előírt szög kiszámítására, ezért alkalmaztam egy MLP típusú neuronhálót. A neuronhálót a szimuláció alapján tanítottam, a mért q_1 és $\dot{q}_1$ állapotok szerint tudja kiszámítani a második szegmens előírt szögét. Két neuronhálót alkalmaztam külön a két szabályzóra.

5.1 Egyrétegű előrecsatolt neuronháló felépítése

A neuronháló 3 fő részre osztható, az első a bemeneti réteg, amely továbbítja az adatokat a háló többi részéhez. A második a rejtett réteg, amely egy neuront tartalmaz, és a harmadik a kimeneti réteg. Két bemeneti neuron van, mivel két állapotot mérünk, a szögelfordulást és a szögsebességet.

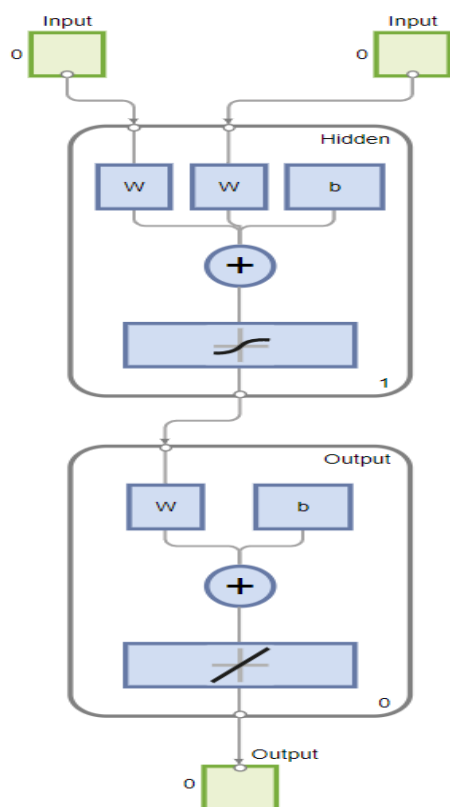
Egy kimeneti neuron található, mivel egy vezérlőjelet kell kiszámítsunk. A neuronháló bemenetén az adatokat előfeldolgozni kell, így az eltolt, és skálázott értékek kerülnek a rejtett rétegre.



5.1. ábra. Egy rétegű, egy neuront tartalmazó neuronháló felépítése, az aktivációs függvényeket is feltüntetve

5.2 Megvalósítás

A megvalósítás a Matlab feedforwardnet[14] beépített függvénnyel és az ahhoz kapcsolódó funkciókkal történt. A tanításhoz a Levenberg-Marquardt algoritmus használtam, a hibaszámolás az átlagos négyzetes eltérés szerint történik. A rejtett réteg aktivációs függvénye tansig, a kimeneti rétegé purelin aktivációs függvény.

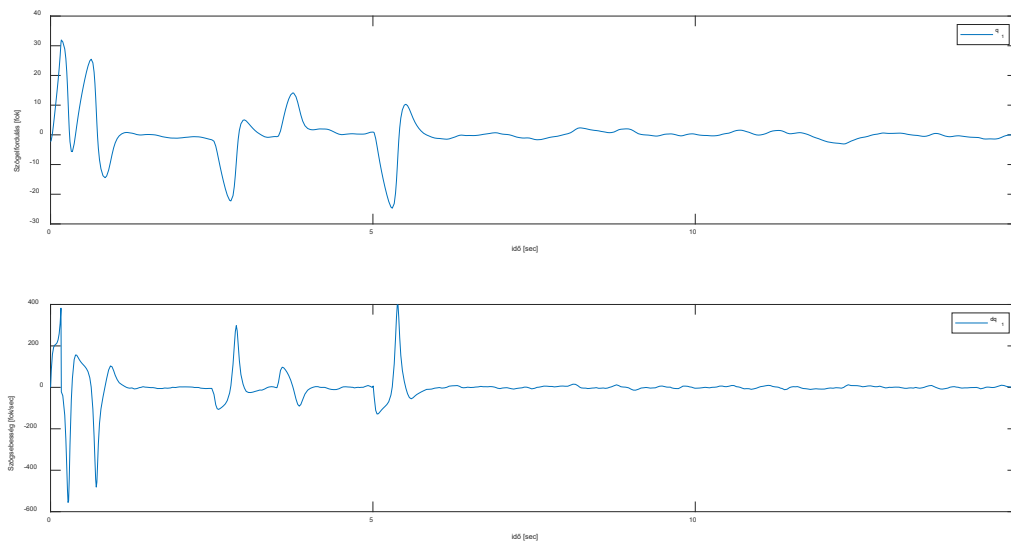


5.2. ábra. Matlabban megvalósított neuronháló tömbvázlata

5.2.1 DLQR

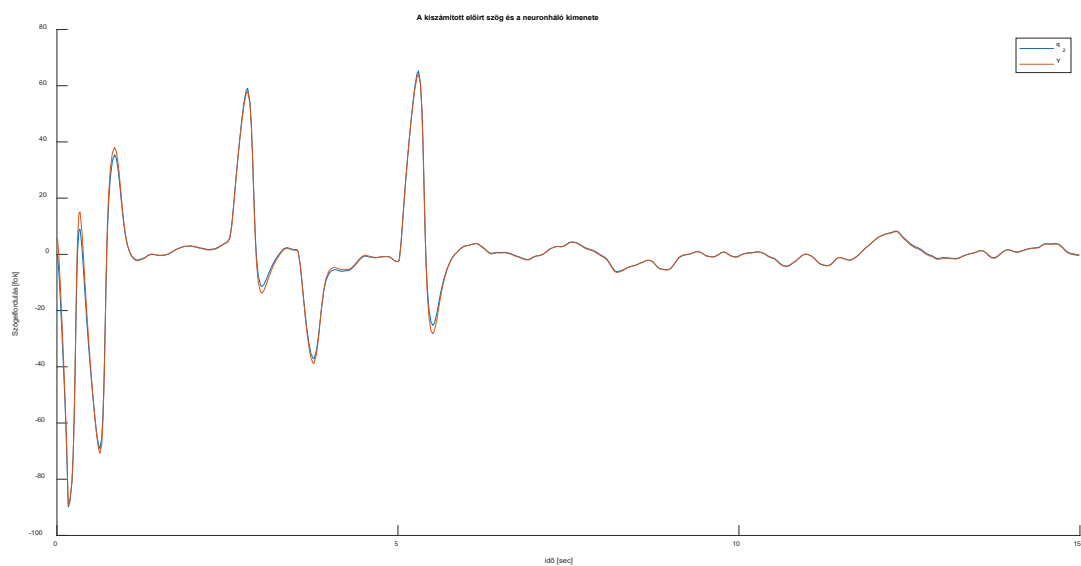
A 6. fejezetben szimuláltam le a szabályozást. A kezdőállapotokat a lineáris tartomány szélére érdemes választani, amelyekre még sikeres a lineáris szabályzó, hogy a tesztjelek, amikre tanítom a neuronhálót, minél nagyobb tartományt fedjenek le. A rendszerre egy konstans fehér zajt és 3 előre meghatározott pontba egy nagyobb zajt is adtam, amelyek célja az volt, hogy valósághűbb legyen a szimulált szabályozás és minél aktívabb legyen a szabályzó, így több hasznos adatot generálva a tanítóhalmazhoz.

Az (5.3. ábra.) ábrán látható a DLQR szabályzót alkalmazva az első szegmens időbeli szögelfordulása és szögsebessége. Mivel a DLQR szabályozás a (a 3.1. ábra szerinti koordinata rendszerben) π rad – értékre kell eljuttassa az első állapotot, ezért a neuronhálónak is így adtam meg az adatokat, π radiánnal eltolva. A könnyebb átláthatóság kedvéért ezeket az értékeket ábrázolásakor a továbbiakban átalakítottam fok, illetve fok/másodperc mértékegységekké.



5.3. ábra. Tanítóhalmaz (DLQR szabályozás fehér zajjal)

A (5.4. ábra) ábrán látható a tanítás eredménye. Látható, hogy sikeresen megtudta tanulni az összefüggést az első szegmens állapotai és az előírt szöge között. A legnagyobb eltérések 3-5 ° közöttiek, amelyek csak a gyors, intenzív beavatkozásoknál jelennek meg, viszonylag kevés ideig.

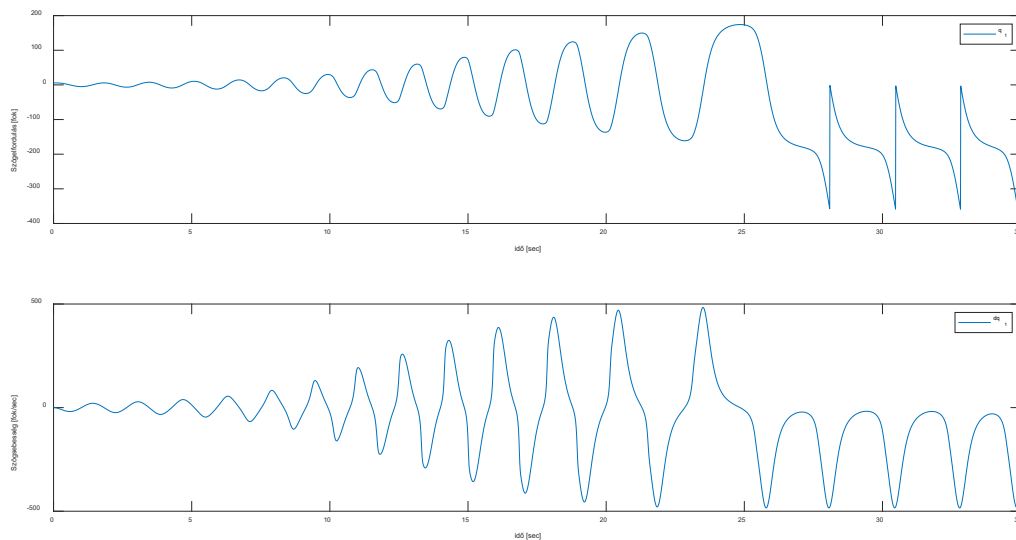


5.4. ábra. A neuronháló kimenete összehasonlítva a kiszámított előírt szöggel

5.2.2 Swing up

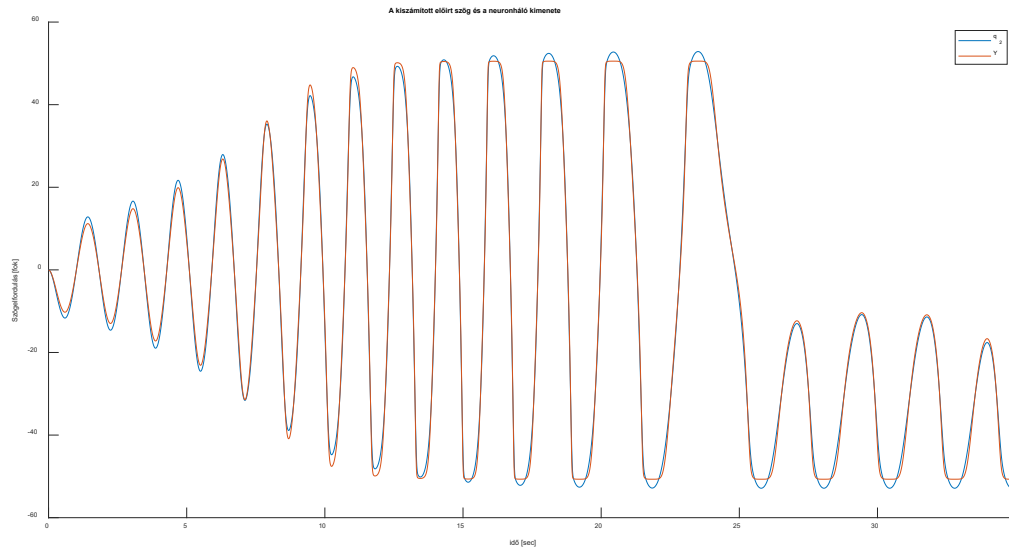
A swing-up, vagyis feltornáztató fázisra is ugyan azt a neuronháló felépítést és tanítási algoritmusokat alkalmaztam. Leszimulálva egy sikeres feltornásztatást, és nem átkapcsolva a DLQR szabályzóra (4.1.4) megkaptam egy az 5.5. ábra alsó felében látható q_2 pályát időben. Ezt a pályát követve a valós rendszer is fel kell tudja tornáztatni magát az UEP pontjának közelébe. Az 5.5. ábra első grafikonján látható, hogy a szabályzó sikeresen eléri a 180° -ot 25 másodperc után, viszont ott nem tudja megtartani magát, ilyenkor kell majd alkalmazni, az ezen a tartományon pontosabb DLQR szabályzót.

Mivel a valós rendszeren is a $[-2\pi, 2\pi]$ tartományban dolgozik a szabályzó, így a tanítóhalmazban is ezen a tartományon belülre korlátoztam a szögeket. Ilyen abban az esetben fordulhatott elő, ha akár többször is körbe fordult az acrobat. Ez látható a 28. másodperc után.



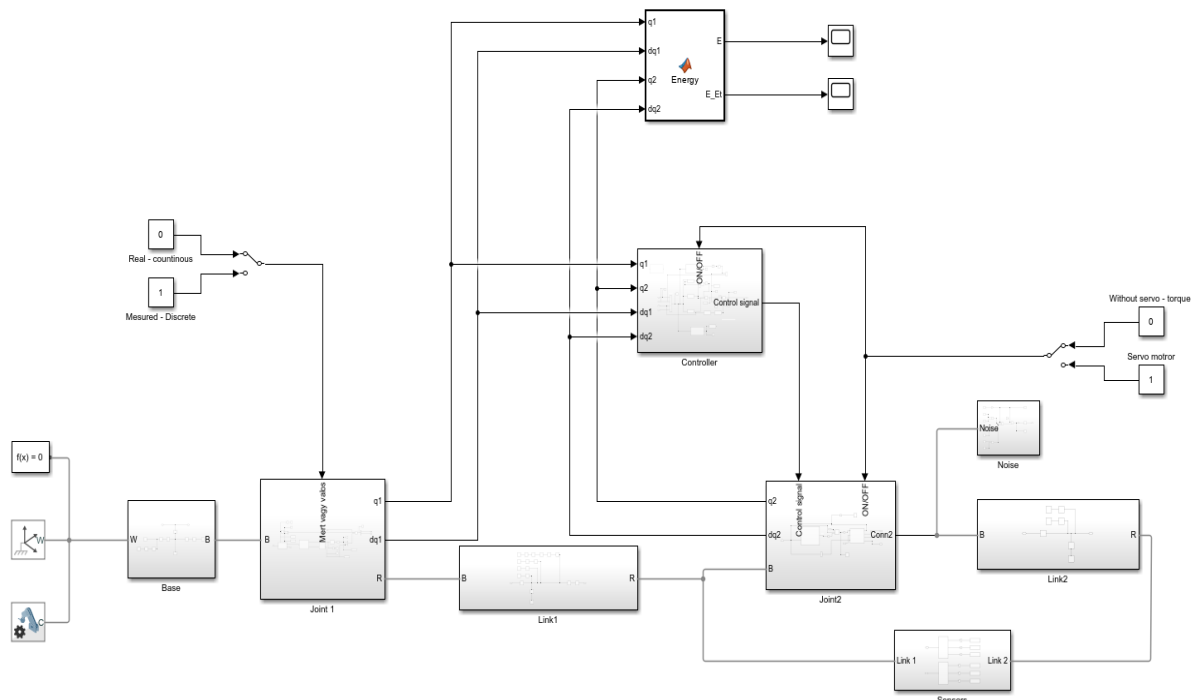
5.5. ábra. Tanítóhalmaz a feltornáztatáshoz (Swing up szabályozás)

A tanítás eredménye látható a (5.6. ábra) ábrán. A legnagyobb különbség az előírt q_2 szögpálya és a neuronháló által kiszámított között, 2° , tehát a tanulás itt is sikeresnek bizonyult.



5.6. ábra. A neuronháló kimenete és a kiszámított q_2 szög összehasonlítása

6 Modellézés, szimuláció



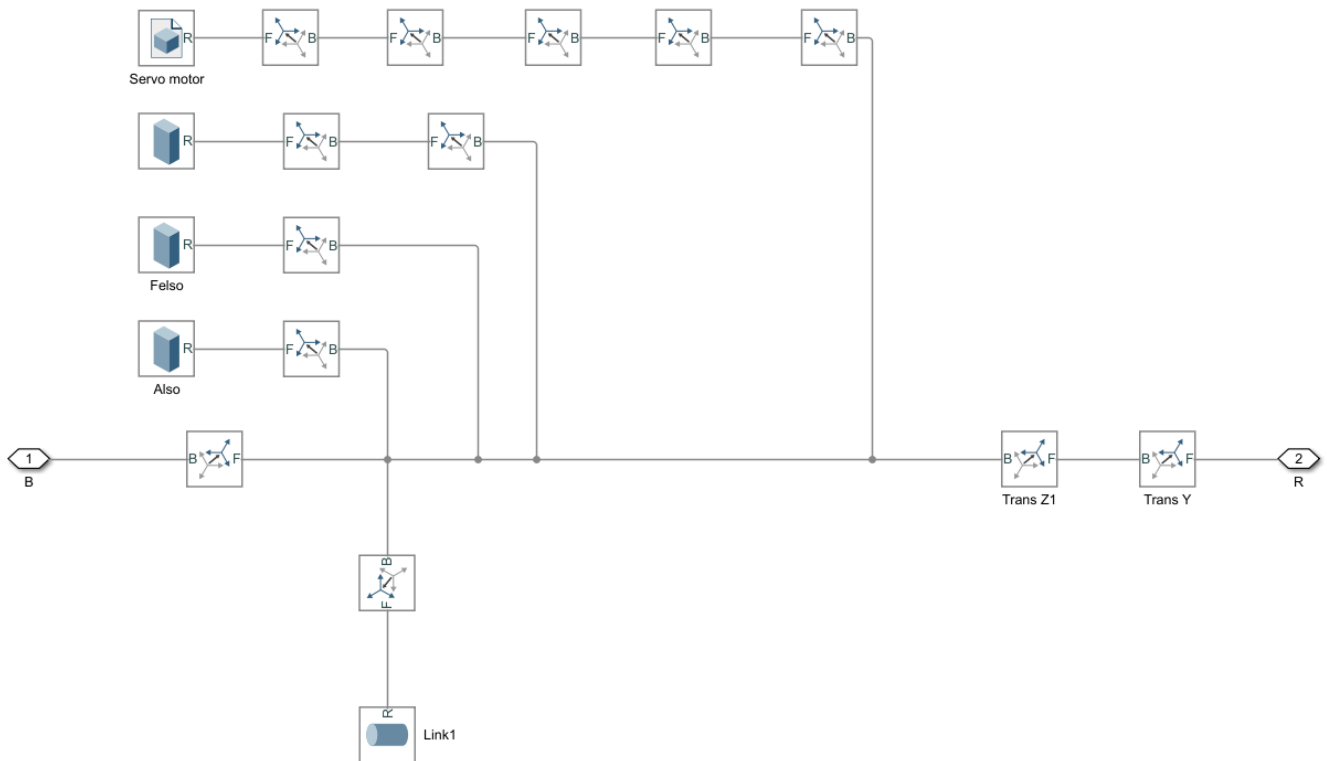
6.1. ábra. Az acrobot és a szabályzás Simulink modellje

A modellezésre a Simulink-hez tartozó Simscape[15] csomagot használtam, simulink elemekkel egybekötve. A Simscape egy különálló szoftvercsomag a Simulinken belül, amely a fizikai elemek modellezésére szolgál. Ennek segítségével hoztam létre az acrobot 3D-s modelljét, megadva a pontos paramétereket. Ezek a következő alrendszerek: Base, Joint1, Link1, Joint2, Link2, amelyek a (6.1. ábra) ábrán láthatóak. A Sensors nevű alrendszerben mérem külön a két szegmens egyes paramétereit, mint például az össztömeg vagy az inercia. A Noise alrendszerben találhatóak a zaj generáláshoz szükséges elemek, amelyek a második szegmensre hatnak. Látható, hogy a Controller tömb megkapja a rendszer állapotait, majd a vezérlőjelet kiszámítva hat a második, aktív csuklóra. A jobb oldalon látható kapcsolóval lehet választani, hogy a második csuklót direktben nyomatékkal vezéreljük, vagy beiktassuk a szervomotort, azáltal megvezérelve a második csuklót. A baloldalon található kapcsolóval lehet állítani, hogy a szabályzó a Simulink által mért szögeket és szöggyorsulásokat használja, vagy a diszkért, forgójeladó (incremental rotary encoder) által kiszámított értéket. Fent látható a rendszer energiáját kiszámító Simulink tömb, amely a [13, p. 3] szerint számolja ki a rendszer pillanatnyi energiáját. A második kimenetén található a cél energia (0 mozgási energia és potenciális energia a felső invertált helyzetében) és az aktuális energiája közti eltérés. A szimuláció az ode23t[16] módszert használja numerikus integrálásra, amely egy időlépéses, adaptív integráló algoritmus, amely ebben az esetben hatékonyabb, mint az ode45.

6.1 A modell felépítése

6.1.1 Fizikai elemek

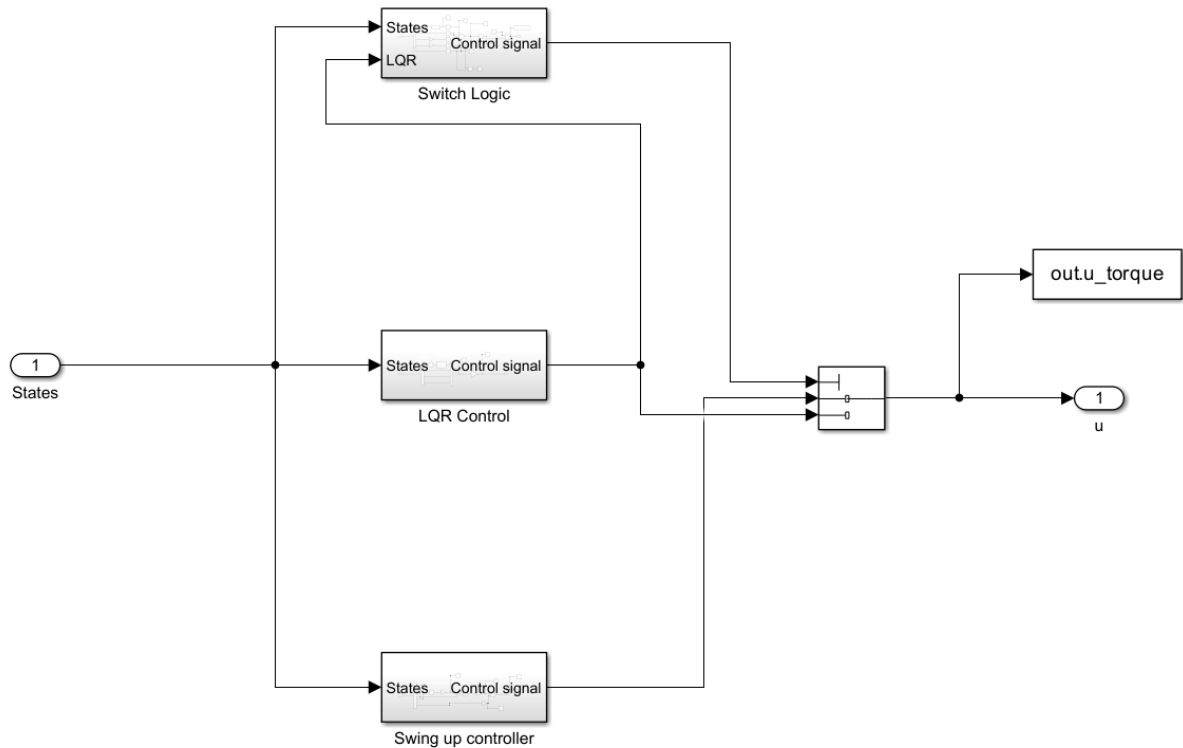
A Simscape csomag által könnyen meg lehet adni a fizikai elemek tulajdonságait, mint a méretük, tömegük. Könnyedén és alrendszerre bontva le lehet modellezni az egész robotot ezáltal.



6.2. ábra. A második szegmens Simscape modellje

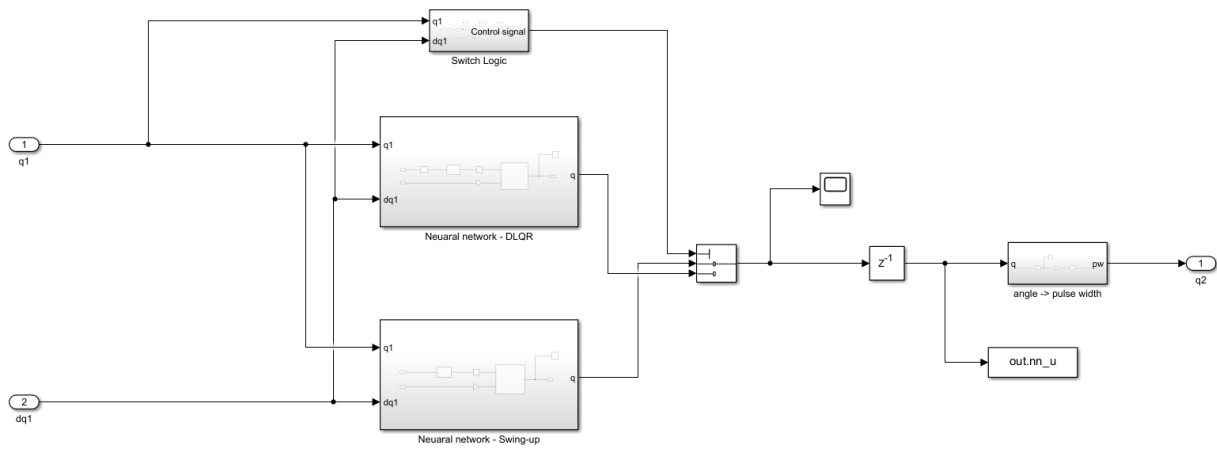
A (6.2. ábra) ábrán látható a második szegmens fizikai modellje. Megtalálható a rúd, a végén lévő összekötő elemek, és a szervomotor 3D-s modellje is.[17]

6.1.2 Szabályzó



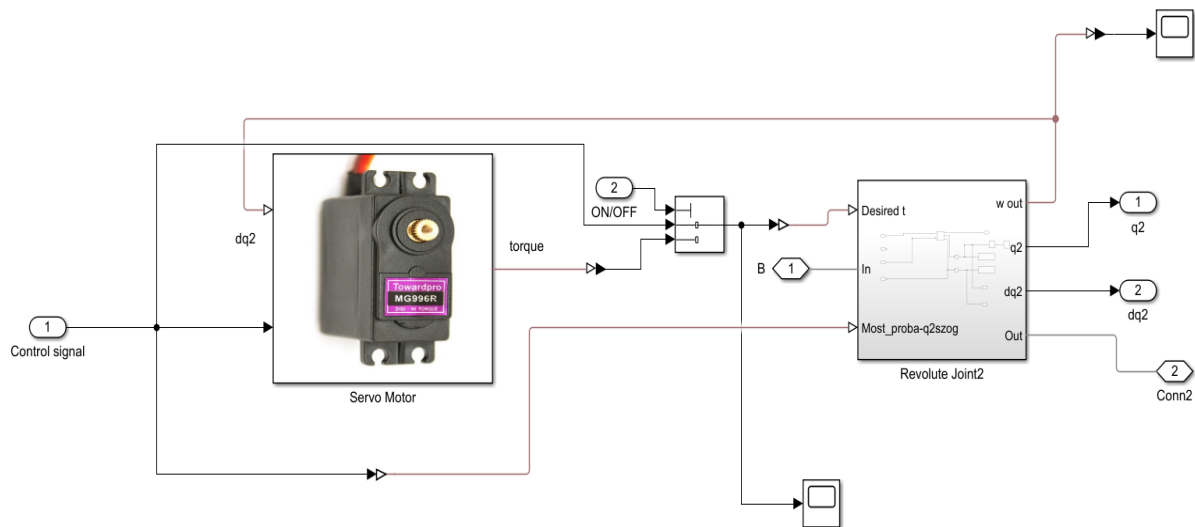
6.3. ábra. A szabályzó tömbvázlata

A szimulációban megtalálható mind a DLQR és energia alapú szabályzó (6.3. ábra), mind ennek a neuronhálóval való implementálása (6.4. ábra). A neuronhálók Simulink modelljét, vagyis az adatok feldolgozását, majd a súlyokkal való szorzását a tanítás után a gensim beépített függvénnyel generáltam ki.



6.4. ábra. A neuronhálóval megvalósított szabályzás

6.1.3 Szervomotor

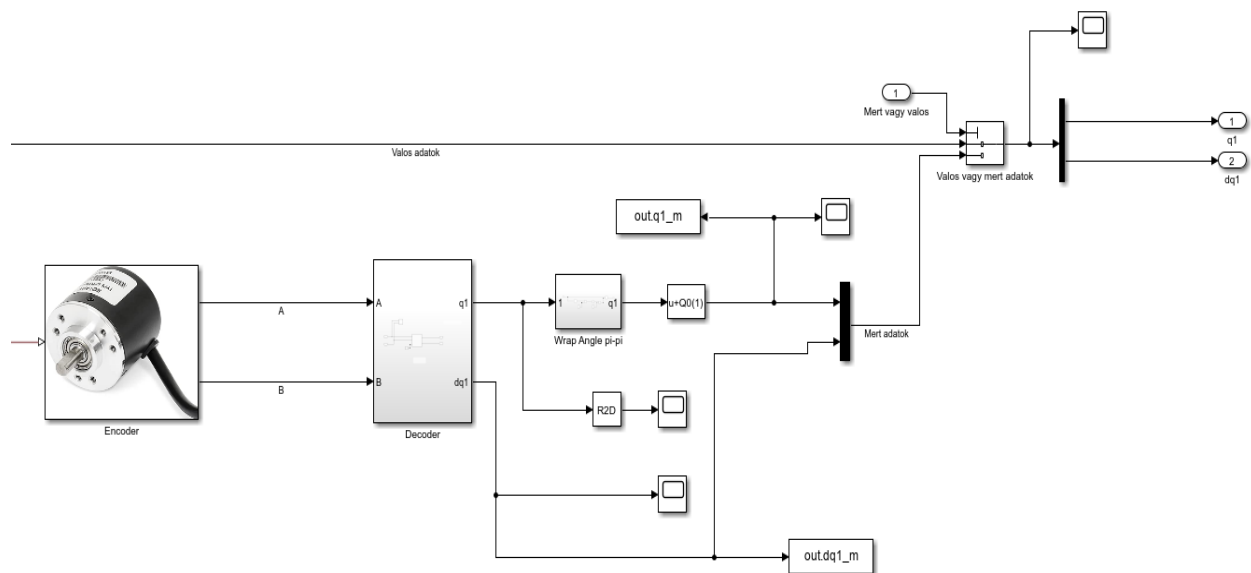


6.5. ábra. A szervomotor és a második csukló Simscape modellje

A Servo Motor alrendszerben található a szervomotor Simscape modellje, a valóságban is használt szervomotor katalógus adatai szerint. Itt látható az ON/OFF kapcsoló szerepe is, amely vagy közvetlen a vezérlő jellel hat a csuklóra, ami nyomatékban van kifejezve, vagy a servo motor által szolgáltatott nyomaték mozditja el a második szegmenst. Ebben az esetben a vezérlőjel egy előírt szög kell legyen, amit majd a servo motor követ.

6.1.4 Forgóérzékelő

Az első csuklóban található a forgóérzékelő, vagyis rotary encoder.

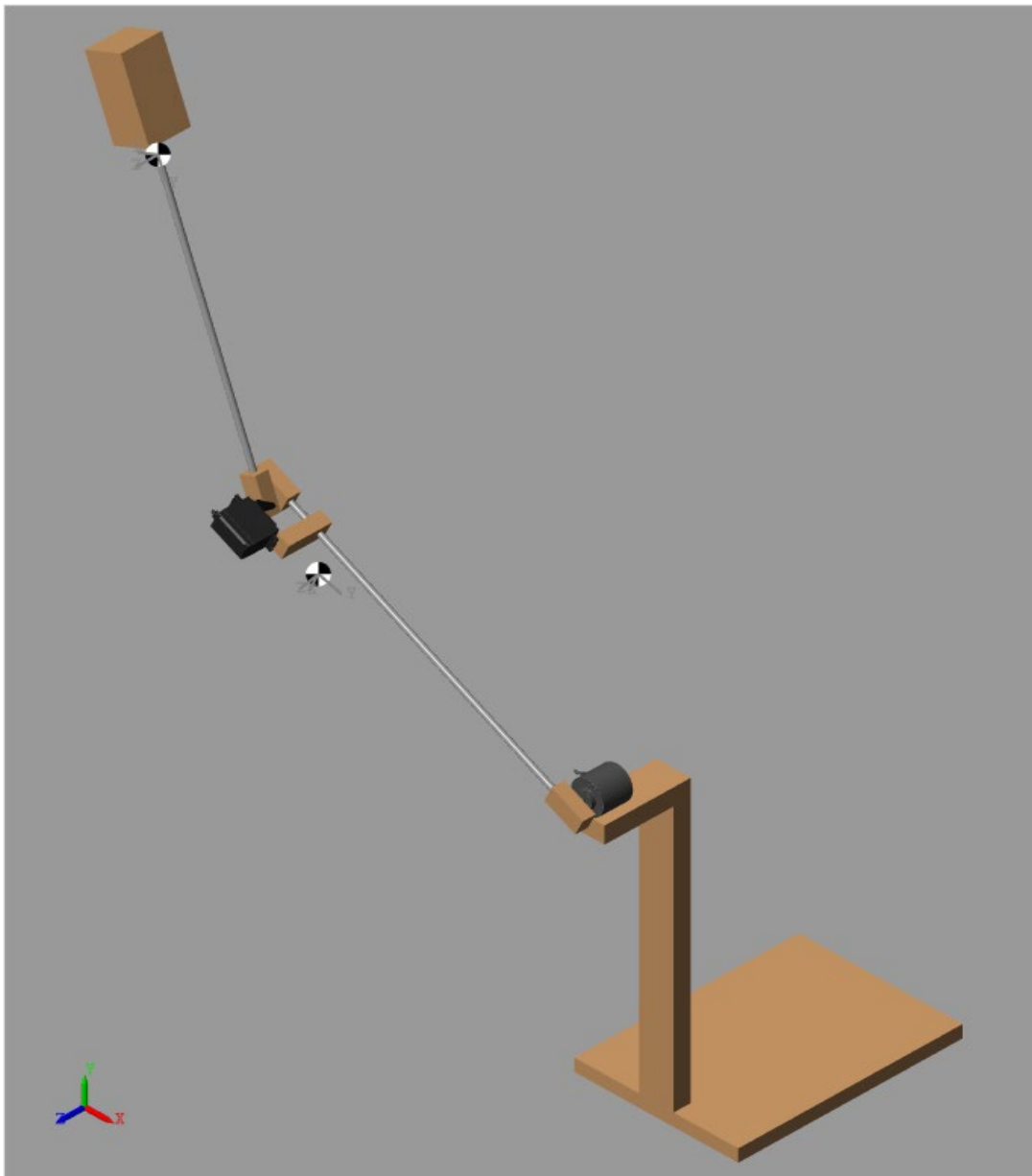


6.6. ábra. Az enkóder és az első csukló Simscape modellje

Itt is megtalálható a forgóérzékelő Simscape modellje[18], illetve egy dekóder is, ami az A és B négszög jeleket dolgozza fel és számolja ki a diszkrét szöget és szögsebességet. Látható továbbá itt is a második kapcsoló, amivel válthatunk a Simulink által generált adatok és a forgóérzékelő és dekóder által szolgáltatott adatok között.

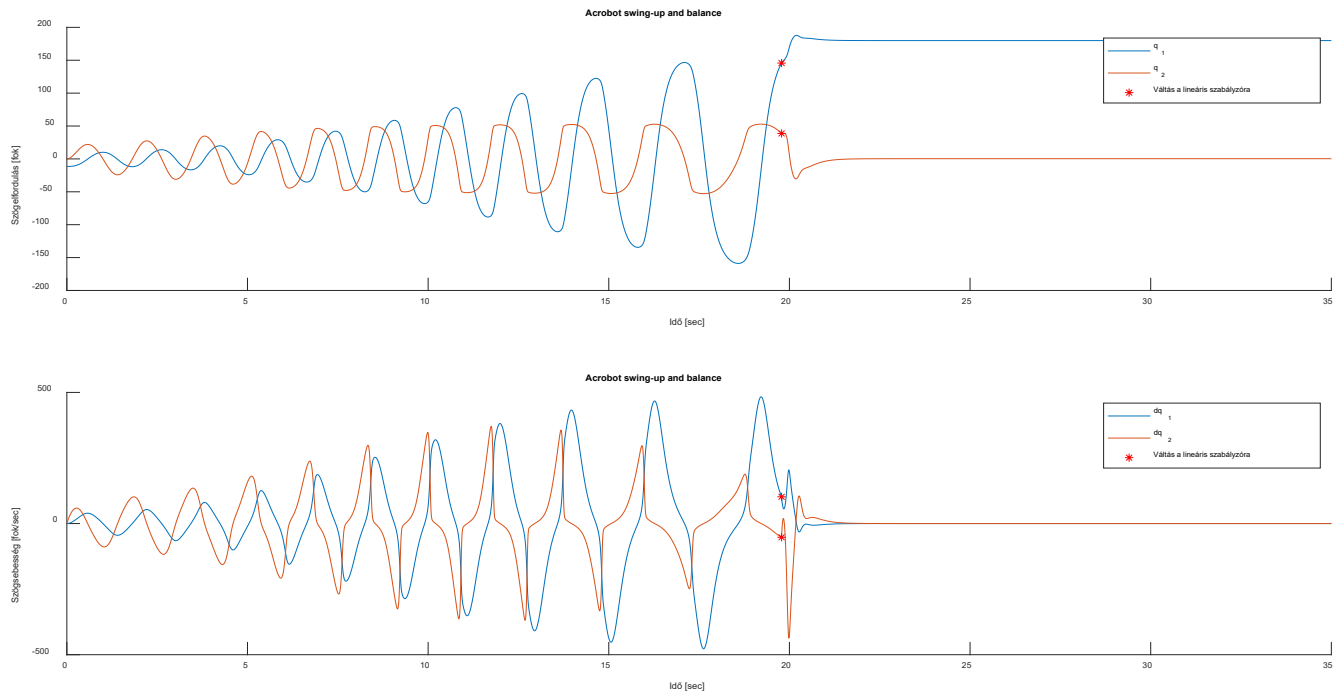
6.2 Eredmények

A valós rendszer paramétereit elvégeztem a szimulációt, a kezdőállapot: $X_0 = [0.1 \ 0 \ 0 \ 0]$. Ha a rendszer kezdőállapotát a $[0 \ 0 \ 0 \ 0]$ -ban határozzuk meg, akkor nem fog onnan kimozdulni, tehát érdemes megadni egy minimális 0-tól különböző kezdőállapotot. Ezzel a kezdeti feltornáztató szakasz időbeli hossza is rövidül. A fejlesztőkörnyezetben nyomon lehetett követni a rendszer viselkedését (6.7. ábra).



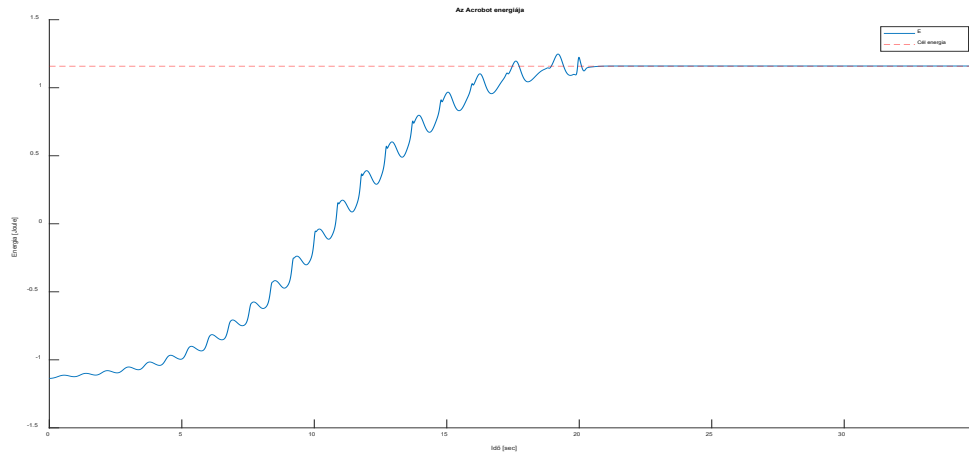
6.7. ábra. Az acrobat 3D-s modellje szimuláció közben

A (6.8. ábra) ábrán láthatunk egy sikeres feltornárást és stabilizálást az UEP pont körül, csillaggal jelölve a váltás időpontját a feltornárást végző szabályzóról a stabilizálást végző lineáris szabályzóra.

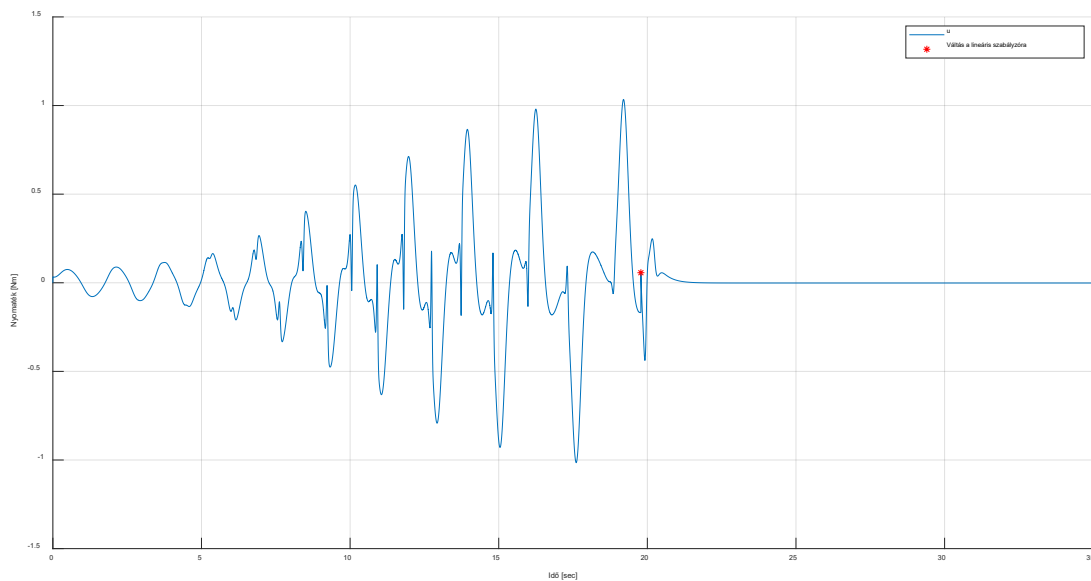


6.8. ábra. Az acrobat szögeinek és szögsebességeinek változása időben egy sikeres szabályzás alatt

A (6.9. ábra) ábrán láthatjuk a rendszer összenergiáját a szabályozás ideje alatt. A célenergia értéke 1.16 J, vagyis csakis a rendszer potenciális energiája az inverz helyzetében.

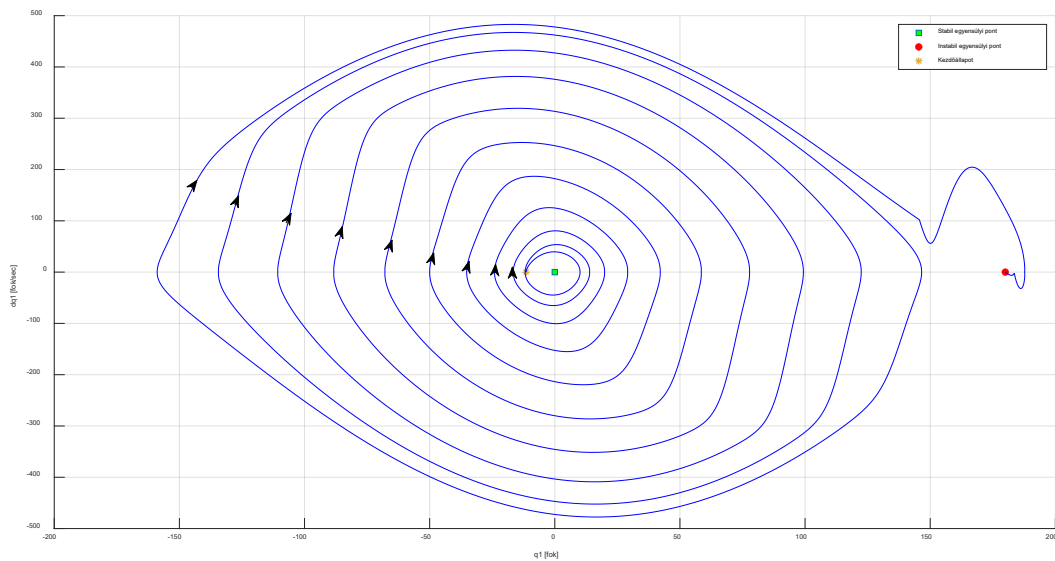


6.9. ábra. A teljes rendszer energiája és a cél energia a szabályozás alatt



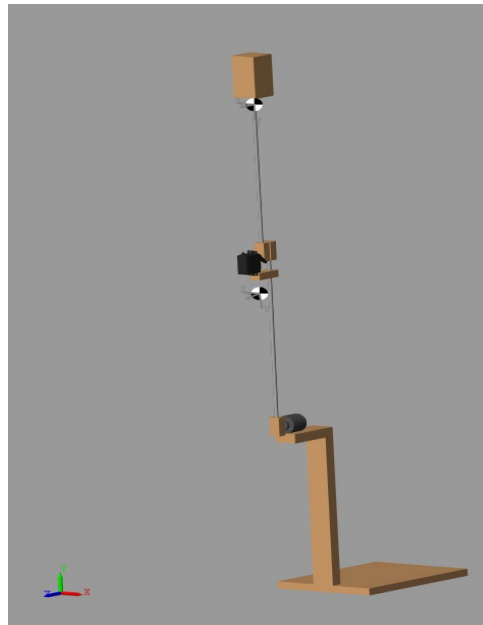
6.10. ábra. A szabályzó jel változása Nm-ben a szabályozás közben

A (6.10. ábra) látható a szabályzó jel, vagyis a második csuklóra ható nyomaték. Ezt a fizikai rendszeren nem tudjuk közvetlenül ráadni a rendszerre a szervomotorral, ott az előírt pályát követi a rendszer.



6.11. ábra. Az acrobot fázisportréja, az első szegmens szögelfordulása és szögsebesség függvényében

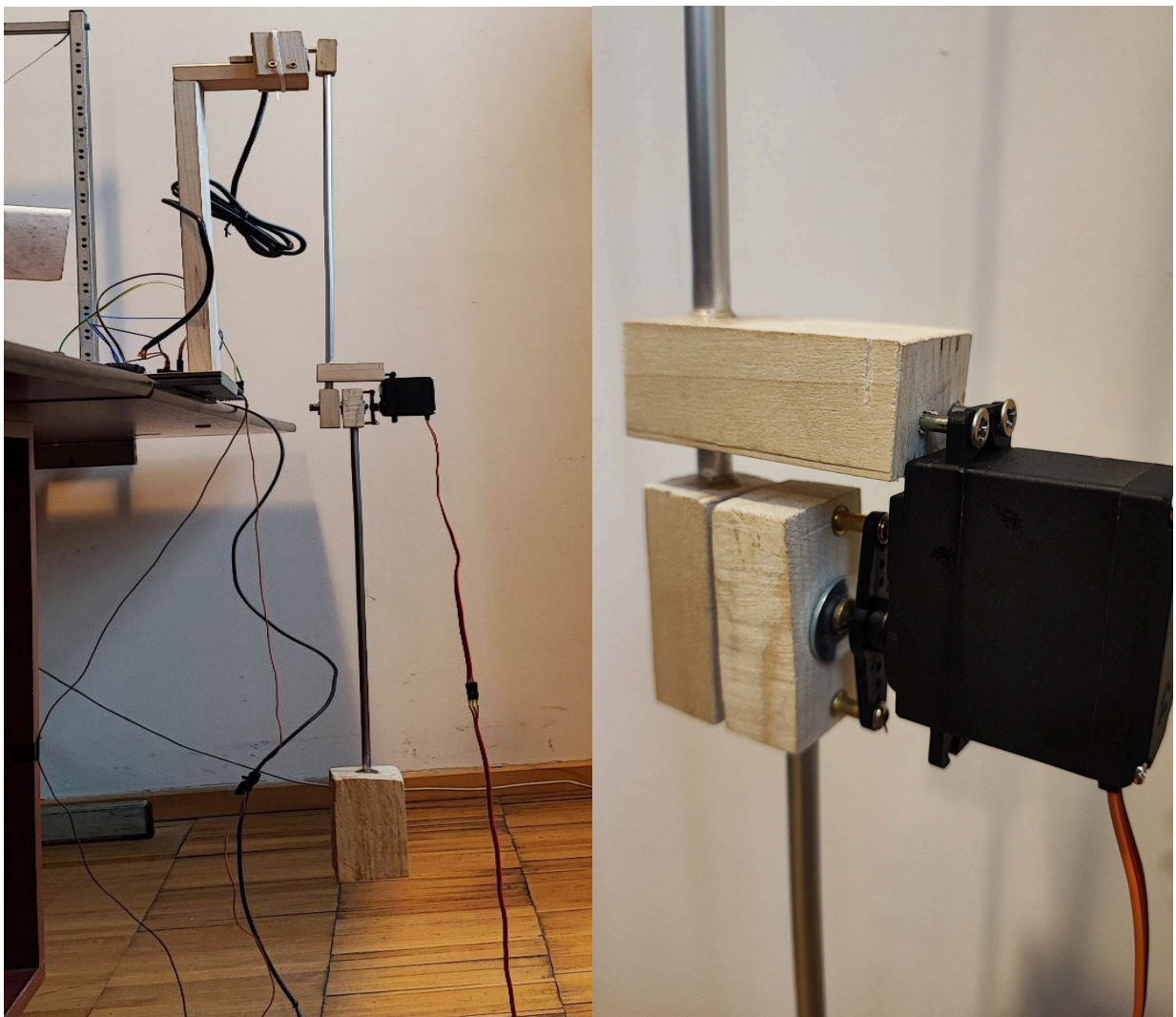
A (6.11. ábra)-n látható a fázisportré egy része a teljes szabályozási szakasz alatt. Az első szegmens szögsebessége látható a szögelfordulás függvényében. Látható, ahogy a 0 közeli kezdeti állapotból eléri az instabil egyensúlyi helyzetét, 180° -ot, 0 sebességgel.



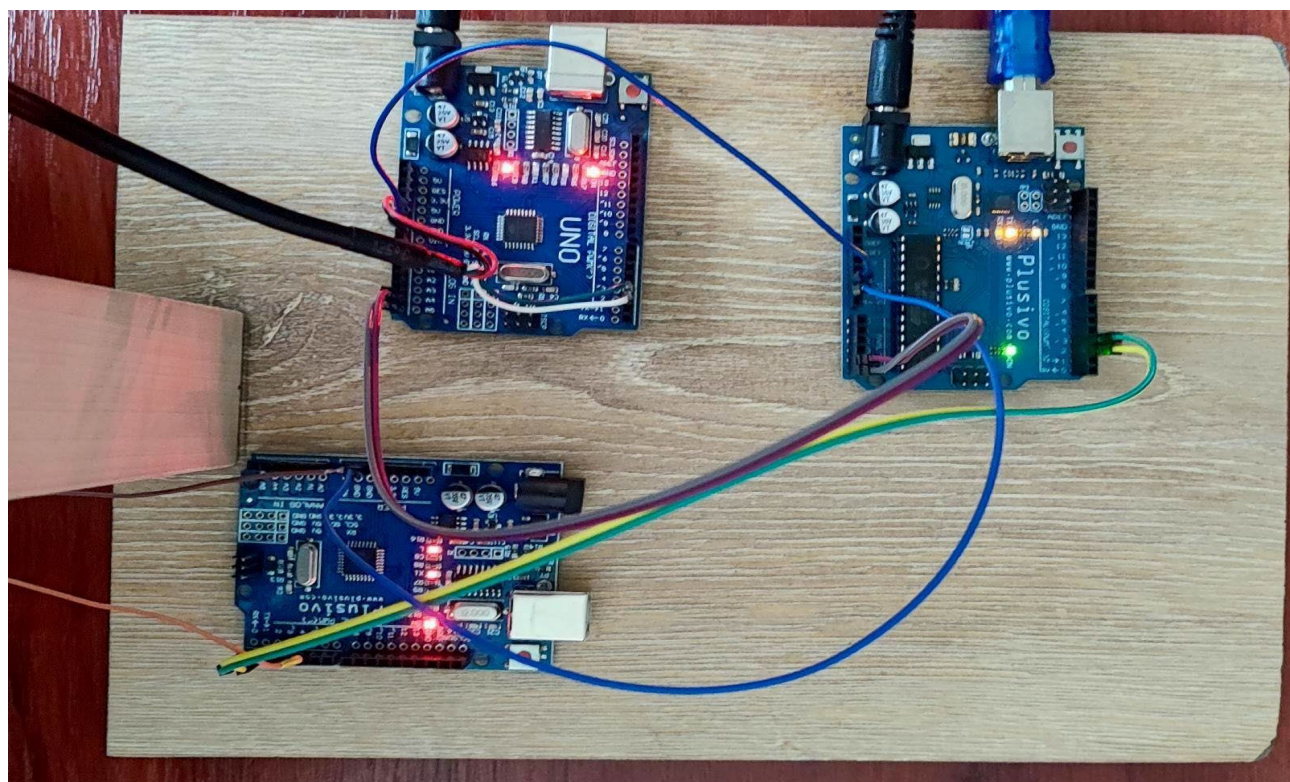
6.12. ábra. Az acrobot 3D-s modellje a felső inverz pontjában

7 Gyakorlati megvalósítás

A (7.1. ábra) ábrán látható az acrobot gyakorlati megvalósítása. A tartószerkezet alsó felében található a szabályozási algoritmust futtató arduino hálózat (7.2. ábra). A tartószerkezet felső felében található a forgóérzékelő, amelyhez rögzítve van maga az acrobot. A szegmensek alumínium rudakból állnak, amelyeknek a végén találhatóak az összekötő elemek. A második szegmens végén található a súly, a nagyobb tehetetlenség elérése érdekében, aminek a tömege 86 gramm. A szervo motorhoz szükséges vezetékek alól vannak elvezetve, mivel nagy mozgási tartományt fed le működés közben.



7.1. ábra. Az acrobot gyakorlati megvalósítása, jobb oldalt a szervomotor és a második csukló



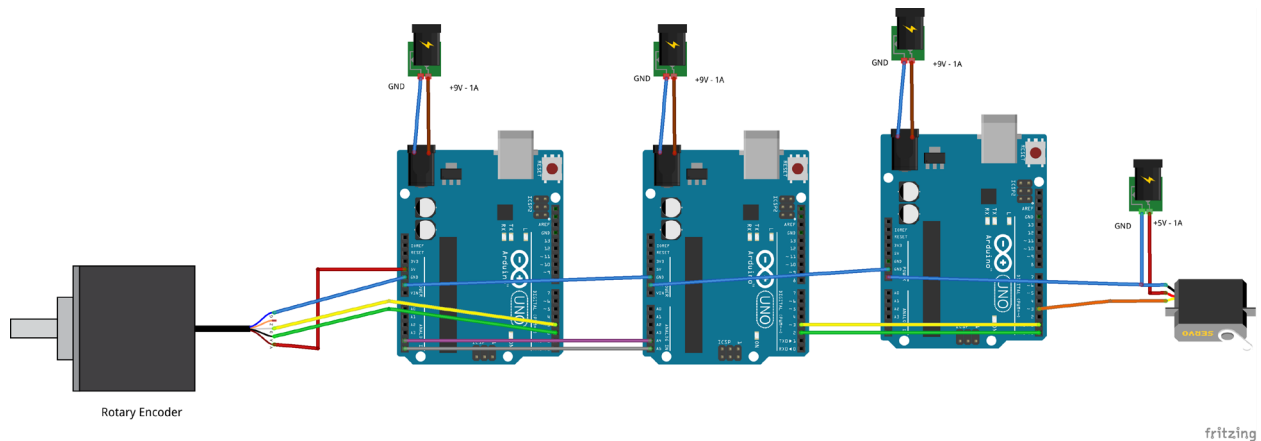
7.2. ábra. Arduino fejlesztőlapokkal megvalósított szabályozás

Jelölés	Érték	Mértékegység
l_1	0.4	m
l_2	0.425	m
lc_1	0.2885	m
lc_2	0.34	m
m_1	0.1012	kg
m_2	0.1203	kg
I_1	0.0015	kg*m ²
I_2	0.0018	kg*m ²

2. táblázat. A valós rendszer paraméterei

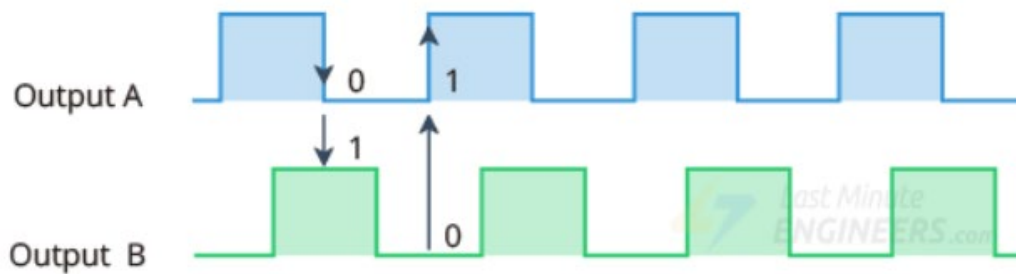
7.1 Acrobot felépítése

Az ábrán látható a szabályozás tömbvázlata. Az 1. arduino feladata a forgóérzékelőtől érkező adatok fogadása és azok feldolgozása. A második feladata a szabályzó jel kiszámítása a beérkező szög, szögsebesség és a neuronhálók segítségével. A harmadik arduino feladata a szervomotor irányítása.

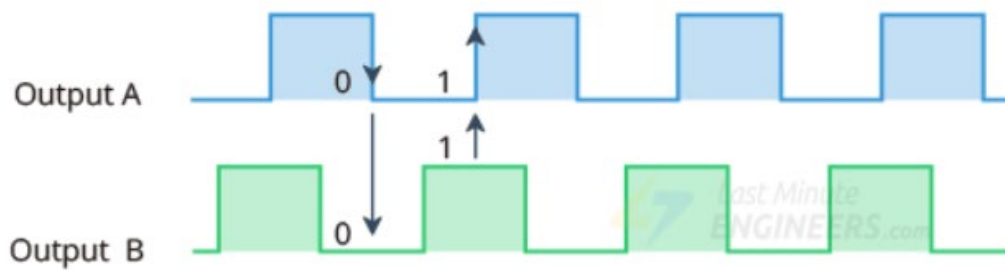


7.3. *ábra.* A gyakorlati szabályozás tömbvázlata

7.2 Állapotok mérése



7.4. *ábra.* A forgóérzékelő által küldött A és B jelforma, amikor a forgás az óramutató járásával megegyező irányban történik[19]



7.5. ábra. A forgóérzékelő által küldött A és B jelforma, amikor a forgás az óramutató járásával megegyező irányban történik[19]

A szög és szögsebesség mérése egy 600 impulzusos forgóérzékelővel történik. Egy teljes fordulat alatt 600 impulzust ad le, A és B fázisban, melyek 90°-kal vannak eltolva egymáshoz képest. A feldolgozás során a két jel helyzetéből kitudom számítani a forgás irányát. Az A fázis lefutó élére nézem meg a B értékét. Ha az 1, akkor az óra mutató irányába forog, ha 0 akkor azzal ellentétesen. Ezt a logikát alkalmazom a B jelre is, ott az A jel értékét figyelem, a B jel lefutó élére, ezzel megduplázva a felbontást. Mindegyik lefutó él egy megszakítást generál az arduinon, ezzel elérve, hogy mindegyik jelet figyelembe vegye. A processzor minden 20ms-onként kiszámolja a relatív szögelfordulást és a szögsebességet. Mivel ebben az esetben egy fordulat 1200 impulzust tartalmaz, ezért minden lefutó él 0.3°-t jelent. A sebesség mérése a (28) képlettel történik, ahol a mintavételezési idő 20ms.[20]

$$\omega = \frac{2\pi n}{Nt}$$

(28)

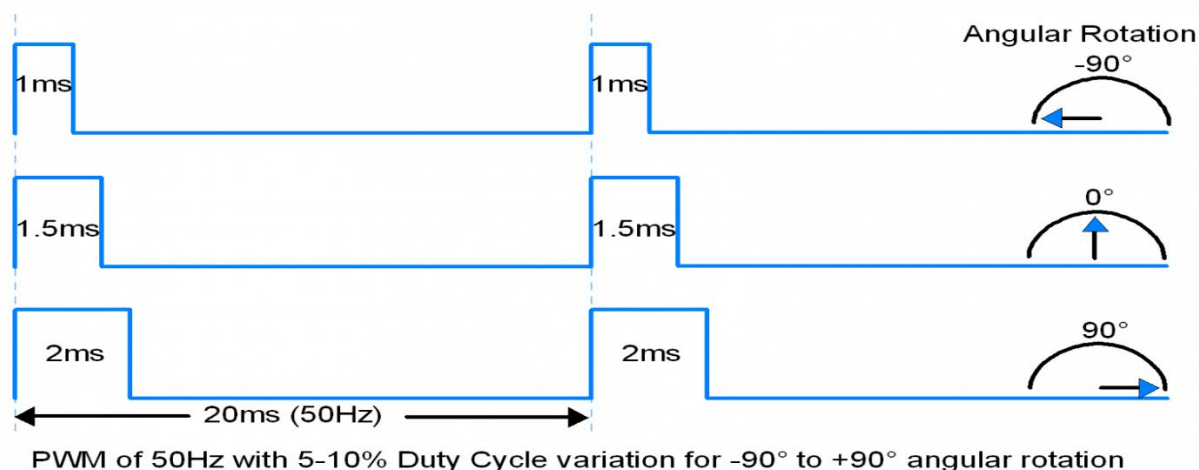
7.3 Vezérlőjel számítás

A számításokat külön arduino végzi. Itt került implementálásra a két szabályozásnak megfelelő egyszerű neuronháló és a köztük való váltást eldöntő logika. A neuronháló megvalósítása:

- Preprocess, adatok skálázása
- Inger kiszámítása
- Aktivációs függvény alkalmazása
- Rejtett réteg ingerének kiszámítása
- Purelin aktivációs függvény alkalmazása
- Process output, végeredmény skálázása

7.4 Szervo motor vezérlés

A szervo motor belsejében található egy belső szabályzó áramkör. Ennek segítségével képes pontos szögpozícióra beállni. Ennek az irányítása egy pwm jellel történik, amely 50 Hz-es frekvenciájú. A kitöltési tényező változtatásával lehet vezérelni a motort. A szervo motor 180°-t képes fordulni. Az 1ms-os kitöltési tényező a minimális -90°-k felel meg, 2ms a 90°-k. A megadott kitöltési tényező és a szervo motor szöge egyenesen arányos a két szélsőérték között, tehát ennek alapján bármilyen szöget be tudunk állítani 5 microsecundumos pontossággal. Az arduinon ezt a Servo.h könyvtár segítségével oldottam meg. Itt megadva a kívánt szöget, automatikusan kiszámítja a szögnek megfelelő kitöltési tényezőt, és beállítja az adott timereket. A Timer1-et az 50Hz-es jel létrehozásához, a Timer2-t a kitöltési tényező pontos megadásához használja a rendszer.



7.6. ábra. 50Hz-es PWM jel a szervomotor irányításához, feltüntetve a két szélső és a középső állapotához szükséges kitöltési tényezőt[21]

7.5 Kommunikáció

A szabályozás megvalósításához 3 külön egységet használtam. Ennek a főbb okai: a feladatok elkülönítése, jobb átláthatóság, a Timerek közötti konfliktusok, ütközések elkerülése.

Az első és második arduino között i2C kommunikációs protokollt használok, a Wire.h könyvtárat használva.[22] Az első, forgóérzékelőtől érkező jeleket feldolgozó egység a Master. 20ms-ként küld el egy union típusú adatot, ami tartalmaz 2 float típusú számot, 8 byte-t küld el összesen.

A Slave-ben minden küldéskor aktiválódik egy megszakítás, ami elmenti a küldött tömb első, és második 4 byte-át külön változóba, amik jelentik a kiszámított aktuális szögelfordulást és szögsebességet.

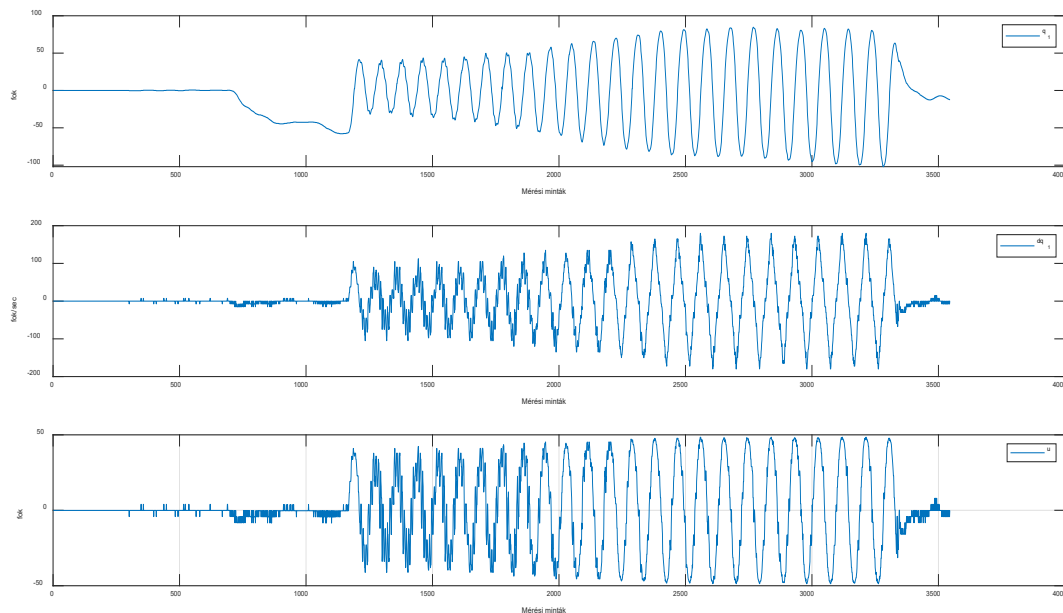
A második és harmadik egység között soros kommunikációt használok. Az arduinon ezeket az RX és TX pinek segítségével lehet megvalósítani. Az eszköz ezek segítségével kommunikál a számítógéppel is, ezért a mérések elvégzése érdekében a számításokat végző arduinon szoftveresen hoztam létre az RX és TX pineknek megfelelő 2,3-as digitális pineket, a SoftwareSerial könyvtárral.[23] A protokoll csomagokat állít össze, amely tartalmaz egy start bitet, az adatot, paritás bitet, stop bitet, majd bitekké darabolva küldi el.

Miután a fogadó fél megkapta a biteket, visszaállítva azokat az UART bufferben tárolja, onnan kiolvasva tárolja el egy változóban. A küldött adat a kiszámított előírt szög float típus, 4 bájtban ábrázolva.

8 Gyakorlati eredmények

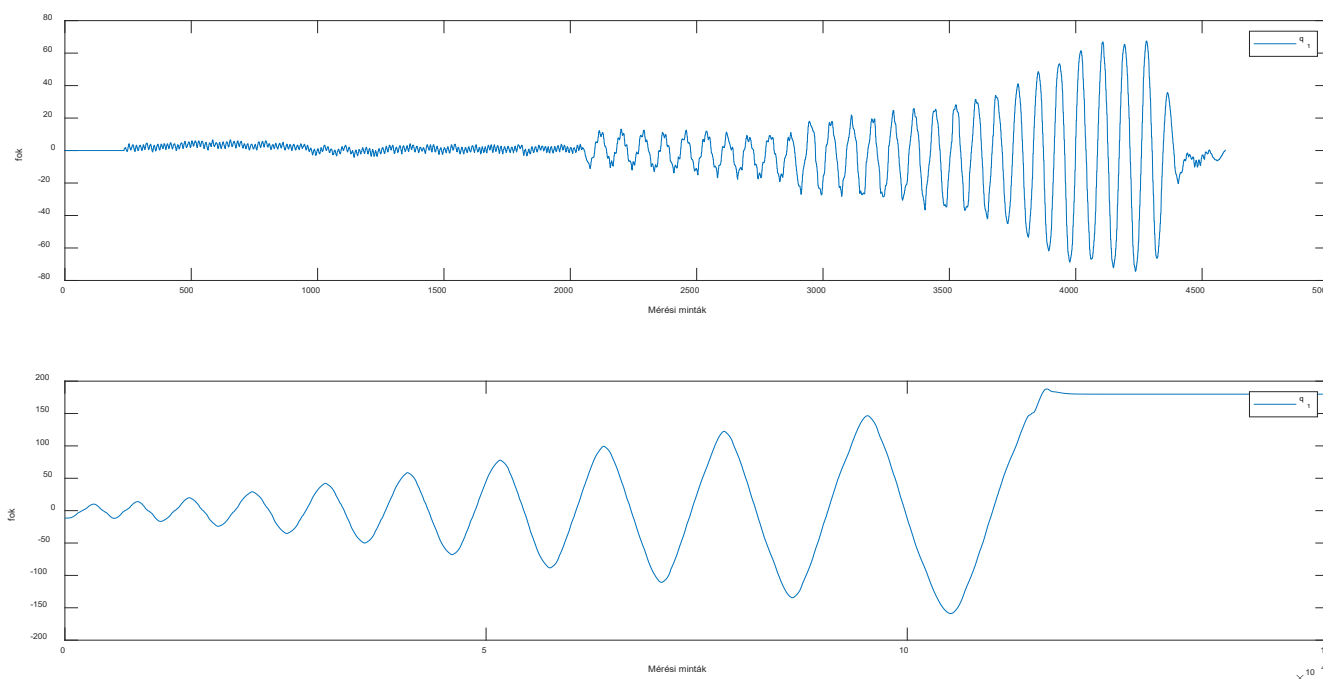
A számításokat végző arduinot a számítógéphez csatlakoztatva tudtam mérni az állapotokat, és a kiszámolt vezérlő jelet. A maximális szög, amit elér a feltornászási fázisban 100.8° (8.1. ábra). A mérést -57° kezdőpozícióból kezdtem, mivel a 0 körüli szögekre, a szervó motor nem tudja időben lekövetni az előírt pályáját, ezért beremeg a rendszer és időben csak később kezdődik el a swing-up fázis.

A rendszer nem tudja elérni az UEP pontját, viszont látható, hogy a lengések amplitúdója fokozatosan nő. Legfőképp a fizikai határai miatt nem tudja elérni az UEP pontját, a továbbiakban szükséges lesz a vezetékek meghosszabbítása, és a tartószerkezet stabilabbá tétele, rögzítése, hogy minimális legyen annak az energia elnyelése.



8.1. ábra. A megvalósított rendszer mérési eredményei szabályozás közben

A (8.2. ábra) ábrán látható összehasonlítva a leszimulált rendszer és a valós mérés. Megfigyelhető, hogy csak több idő után tud kilépni a 0° környékéről a valós rendszer, mivel berezeg.



8.2. ábra. Feltornáztatási fázis összehasonlítása.

Valós rendszer (felső). Szimulált, elméleti modell (alsó)

9 Következtetések, jövőbeni célok

A dolgozat célja egy alulaktuált rendszer, az acrobot szabályozása, és annak bemutatása. Látható, hogy e rendszerek szabályozása a valóságban sokkal nehezebb, mint például az acroboté is.

Jövőbeni cél a szerkezet stabilabbá tétele, minimalizálva a másfajta mozgásokból eredő energiavesztést. Másik cél a motor vezetékeinek a megoldása, mivel egy 40 cm sugarú kört ír le a teljes mozgási tartományban így sokkal hosszabb vezetékek kellenek, ezáltal a teljes feltornáztatás is elérhető lesz.

10 Irodalomjegyzék

- [1] Bodor Bálint, 2015. Alulaktuált robotok szabályozásának eltérő megközelítései TDK dolgozat
- [2] Mark W. Spong. 1995. The Swing Up Control Problem For The Acrobot.
- [3] M. W. Spong. 1994. Swing Up Control of the Acrobot. IEEE 1050-4729/9
- [4] X. Xin and T. Yamasaki, 2012 Energy-based swing-up control for a remotely driven acrobot: Theoretical and experimental results. *IEEE Transactions on Control Systems Technology*, vol. 20, no. 4, pp. 1048–1056, doi: 10.1109/TCST.2011.2159220.
- [5] R. Tedrake. 2009. Underactuated Robotics: Learning, Planning, and Control for Efficient and Agile Machines Course Notes for MIT 6.832.
- [6] R Nave, “Moment of Inertia: Rod.” <http://hyperphysics.phy-astr.gsu.edu/hbase/mi2.html#irod> (accessed Apr. 17, 2023).
- [7] Russ Tedrake. Multi-Body Dynamics. http://underactuated.mit.edu/multibody.html#double_pendulum (accessed Apr. 17, 2023).
- [8] Márton Lőrinc, Robotokarok Dinamikus Modellezes Palyatervezes PDG PID.
https://www.ms.sapientia.ro/~martonl/Docs/Lectures/Robotokarok_Dinamikus_Modellezes_Palyatervezes_PDG_PID.pdf
- [9] M. W. Spong. n.d. Underactuated Mechanical Systems.
- [10] X. Xin and M. Kaneda. 2001. A Robust Control Approach to the Swing up Control Problem for the Acrobot.
- [11] D. N. Kutasi and L. Márton. 2010. *Rendszerelmélet laboratóriumi gyakorlatok*. Scientia
- [12] György Katalin (n.d.) Diszkrét LQR állapot utáni szabályzó laborgyakorlat.
- [13] X. Xin and M. Kaneda. Nov. 2007. Analysis of the energy-based swing-up control of the Acrobot. *International Journal of Robust and Nonlinear Control*, vol. 17, no. 16, pp. 1503–1524, , doi: 10.1002/rnc.1184.
- [14] <https://www.mathworks.com/help/deeplearning/ref/feedforwardnet.html> (accessed Apr. 17, 2023).
- [15] <https://www.mathworks.com/products/simscape.html> (accessed Apr. 17, 2023).
- [16] <https://www.mathworks.com/help/matlab/ref/ode23t.html>(accessed Apr. 17, 2023).
- [17] https://grabcad.com/library/mg996r-servo/details?folder_id=218498(accessed Apr. 17, 2023).
- [18] https://grabcad.com/library/optical-rotary-encoder-1/details?folder_id=13312851 (accessed Apr. 17, 2023).

- [19] <https://lastminuteengineers.com/rotary-encoder-arduino-tutorial/> (accessed Apr. 17, 2023).
- [20] <https://www.motioncontroltips.com/how-are-encoders-used-for-speed-measurement/> (accessed Apr. 17, 2023).
- [21] <https://www.electronicwings.com/sensors-modules/servo-motor> (accessed Apr. 17, 2023).
- [22] <https://www.circuitbasics.com/how-to-set-up-i2c-communication-for-arduino/> (accessed Apr. 17, 2023).
- [23] <https://www.circuitbasics.com/how-to-set-up-uart-communication-for-arduino/> (accessed Apr. 17, 2023).

11 Ábrajegyzék

3.1. ábra. Az acrobot modellje [5].....	6
4.1. ábra. A szabályzás tömbvázlata[9, p. 12]	9
4.2. ábra. A nyílt rendszer instabil pólusai	11
4.3. ábra. A zárt rendszer stabil pólusai	13
4.4. ábra. A rendszer állapotai szimuláció közben	14
5.1. ábra. Egy rétegű, egy neuront tartalmazó neuronháló felépítése, az aktivációs függvényeket is feltüntetve.....	17
5.2. ábra. Matlabban megvalósított neuronháló tömbvázlata	18
5.3. ábra. Tanítóhalmaz (DLQR szabályozás fehér zajjal)	19
5.4. ábra. A neuronháló kimenete összehasonlítva a kiszámított előírt szöggel	19
5.5. ábra. Tanítóhalmaz a feltornáztatáshoz (Swing up szabályozás)	20
5.6. ábra. A neuronháló kimenete és a kiszámított q_2 szög összehasonlítása	21
6.1. ábra. Az acrobot és a szabályzás Simulink modellje.....	21
6.2. ábra. A második szegmens Simscape modellje	23
6.3. ábra. A szabályzó tömbvázlata	24
6.4. ábra. A neuronhálóval megvalósított szabályzás	25
6.5. ábra. A szervomotor és a második csukló Simscape modellje	25
6.6. ábra. Az enkóder és az első csukló Simscape modellje	26
6.7. ábra. Az acrobot 3D-s modellje szimuláció közben.....	27
6.8. ábra. Az acrobot szögeinek és szögsebességeinek változása időben egy sikeres szabályzás alatt.....	28
6.9. ábra. A teljes rendszer energiája és a cél energia a szabályozás alatt	29
6.10. ábra. A szabályzó jel változása Nm-ben a szabályozás közben	29
6.11. ábra. Az acrobot fázisportréja, az első szegmens szögelfordulása és szögsebesség függvényében.....	30
6.12. ábra. Az acrobot 3D-s modellje a felső inverz pontjában	30
7.1. ábra. Az acrobot gyakorlati megvalósítása, jobb oldalt a szervomotor és a második csukló	31

7.2. ábra. Arduino fejlesztőlapokkal megvalósított szabályozás.....	32
7.3. ábra. A gyakorlati szabályozás tömbvázlata.....	33
7.4. ábra. A forgóérzékelő által küldött A és B jelforma, amikor a forgás az óramutató járásával megegyező irányban történik[19]	33
7.5. ábra. A forgóérzékelő által küldött A és B jelforma, amikor a forgás az óramutató járásával megegyező irányban történik[19]	34
7.6. ábra. 50Hz-es PWM jel a szervomotor irányításához, feltüntetve a két szélső és a középső állapotához szükséges kitöltési tényezőket[21]	36
8.1. ábra. A megvalósított rendszer mérési eredményei szabályozás közben.....	37
8.2. ábra. Feltornáztatási fázis összehasonlítása.	38