

Smidla Ádám

INVERZ INGA MODELL ALAPÚ SZABÁLYOZÁSA

Témavezető: Dr. Magyar Attila

Műszaki Informatikai Kar, Pannon Egyetem

Pannon Egyetem
Intézményi Tudományos Diákköri Konferencia
2014



Tartalomjegyzék

1. Bevezetés	4
1.1. Motiváció	4
1.2. Célkitűzés	5
1.3. Irodalmi áttekintés	5
2. Modellezés	6
2.1. Az inverz inga felépítése	6
2.2. Egyenáramú motor modellezése	6
2.2.1. Egyenáramú motor fizikai háttere	6
2.2.2. Egyenáramú motor állapotter modellje	9
2.2.3. Motor modell verifikáció	10
2.2.4. Kerék-motor kapcsolata	12
2.3. Mechanikai modell	14
2.4. Nemlineáris állapotter modell	18
2.4.1. Input-affin alakú állapotter	18
3. Modell analízis és szimuláció	20
3.1. Modell verifikáció	20
3.1.1. Motor verifikáció	20
3.1.2. Nemlineáris modell verifikációja	22
3.2. Lokális linearizálás	23
3.2.1. Lokálisan linearizált modell implementációja	26
3.3. Lokális dinamikus analízis	27
3.3.1. Aszimptotikus stabilitás	27
3.3.2. Irányíthatóság	28
3.3.3. Megfigyelhetőség	29
3.4. Paraméterérzékenység vizsgálat	29
3.5. Kísérlettervezés egy későbbi paraméterbecsléshez	30
4. Modell alapú szabályozás	32
4.1. Szabályozási cél	32
4.2. Állapotvisszacsatolás	32
4.2.1. Maradó szabályozási hiba kiküszöbölése	33
4.2.2. Pólusáthelyező	33
4.2.3. LQ szabályozó	36

4.3.	Megfigyelő tervezés	38
4.4.	Alapjel követő szabályozás	42
4.5.	Megfigyelő és szervó szabályozó együttes használata zavarással . .	48
4.6.	Fuzzy szabályozó	49
4.6.1.	Fuzzy	49
4.6.2.	Fuzzy szervó szabályozóval	51
4.7.	PID szabályozó	52
4.7.1.	Megvalósított szabályozó	53
4.8.	Szabályozók összehasonlítása	54
5.	Implementáció és tesztelés	56
5.1.	Rendszerterv	56
5.2.	Választott eszközök	56
5.2.1.	Arduino	56
5.2.2.	Motor	57
5.2.3.	GY-80 kombinált szenzor	57
5.2.4.	Motor vezérlő	58
5.3.	Hardver implementáció	59
5.4.	Szoftver implementáció	59
6.	Összegzés	61
6.1.	Konklúzió	61
6.2.	További munka	61
A.	Modellezés és modellanalízis	64
A.1.	Jelölésjegyzék	64
A.2.	Matlab/Simulink forráskód	64
A.2.1.	Lineáris modell	64
A.2.2.	Nemlineáris modell	66
A.2.3.	Lineáris modell LQ	67
A.2.4.	Nemlineáris modell LQ	69
A.2.5.	Lineáris modell szervó	70
A.2.6.	Nemlineáris modell megfigyelo	71
A.3.	Mathematica kód	73
A.3.1.	Pi körüli linearizálás	73
A.4.	Számítások	76
A.4.1.	Használt motor paramétereinek mérése	76
A.4.2.	0 körül linearizált állapotter modell	77

A.5. Paraméterérzékenység	77
B. Implementáció	84
B.1. Hardverelemek adatlapjai	84
B.1.1. Használt motor adatlapja	84
B.1.2. Maxon-motor katalóguslapja	90
B.1.3. Arduino mikrovezérlő	92
B.1.4. Motorvezérlő	94
B.1.5. Gyorsulásmérő	96
B.1.6. Giroszkóp	98
B.2. Arduino forráskód	100

1. Bevezetés

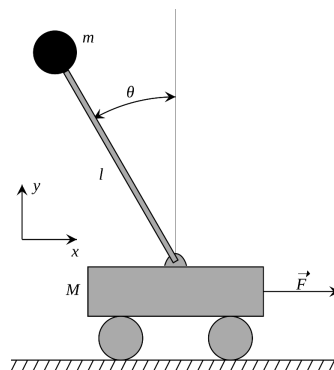
1.1. Motiváció

Az emberiséget mindig foglalkoztatták a saját és a környező világ határai, beleértve a világot leíró fizikai törvényeket is. Ennélfogva időről időre felbukkannak olyan szerkezetek, melyek látszólag ellentmondanak a fizika törvényeinek. Legelső példaként lehet említeni a repülést, amelyről sokáig azt hitték, hogy nem megvalósítható művelet, mindaddig, amíg meg nem jelentek a Wright fivérek. A következő példa az acélból készült hajók lehetnek, amelyek szintén látszólag ellentmondanak a fizikai törvényeknek. Az említett példák között viszonylag friss a Segway personal transporter (1.1 ábra),



1.1. ábra. Segway

amely gyakorlatilag egy inverz inga, amely instabil rendszer lévén (1.2 ábra) a klasszikus és a modern irányításelmélet egyik legelterjedtebb tesztproblémája [1].



1.2. ábra. Inverz inga

1.2. Célkitűzés

A célom egy olyan két kereken egyensúlyozó robot megépítése és szabályozása, amely alkalmas arra, hogy a különböző szabályozási algoritmusokat a diákok számára egyetemi kereten belül be lehessen mutatni. A különféle szabályozási algoritmusokat implementáltam Matlab Simulink környezetben.

A robot egyensúly tartásához szükség van egy modellre, amellyel szabályzót lehet tervezni. Célom az, hogy összehasonlítsak különböző szabályzó algoritmusokat: PID, LQ, Fuzzy, feedback linearizáláson alapuló szabályzó.

Az inverz ingát két darab egyenáramú szervó motor fogja hajtani. Érzékelőként pedig egy gyorsulásmérőt és egy giroszkópot fogok használni. A robot "agya" pedig egy Arduino típusú mikrokontroller lesz.

1.3. Irodalmi áttekintés

Az inverz inga problémája már régóta foglalkoztatja a mérnököket. Az ilyen felépítésű robot a klasszikus kocsirúd kísérleten alapszik, amelynél egy rudat próbálunk egyensúlyban tartani a kocsit előre-hátra mozgásával.

Ma már az ilyen fajta egyensúlyozó járműveket Segway járműként szokást azonosítani, utalva Dean Kamen 2001-es találmányára, a Segway PT-re. Szakdolgozatoktól kezdve [9] egészen tudományos folyóiratban megjelent cikkig [2] terjedően különböző mennyiségű és minőségű anyag található a témában.

Jelen dolgozat fő motivációját egy dolgozat adta [5], amelyben egy kerekeken guruló inverz inga fizikai alapú modellezése és modell alapú szabályozása került megvalósításra. Az abban használt motorok és hardverelemek nagyobbak, mint ami a jelen dolgozatban használok, de nagyon jó alapot szolgált. Az ingában használt olcsó szenzorok fúziójáról [2] cikkben olvashatunk, melyben egy Kalman szűrő és egy *complementary* szűrő használatával sikerül maximális információt kinyerni a gyorsulásmérő és a giroszkóp szenzorokból.

Az inga stabilizálására is rengeteg példa található az irodalomban, példaként említhető [3], melyben a visszalépéses (backstepping) technikát alkalmazzák a stabilizálásra. A már említett [1] cikkben egy robusztus LQR szabályozó kerül alkalmazásra, de számtalan egyéb módszert is találhatunk, amellyel a stabilizálás és az alapjel követése megvalósítható. Egy fuzzy alapú megközelítést tárgyal [4], ami annyiban különbözik a jelen dolgozatban tárgyalt problémától, hogy ott az inga nem kerekeken gurul, hanem egy ún. rotációs inverz inga.

2. Modellezés

A fejezet célja a rendszer dinamikus modelljének fizikai alapon történő felírása. A modell két fő modulból épül fel: a motor, valamint a mechanikus rendszer. Ennek megfelelően a modell két fő modula külön-külön kerül felírásra. A teljes modell a továbbiakban a model analízis, a paraméter érzékenységvizsgálat, valamint a szabályozótervezési lépések alapjául fog szolgálni.

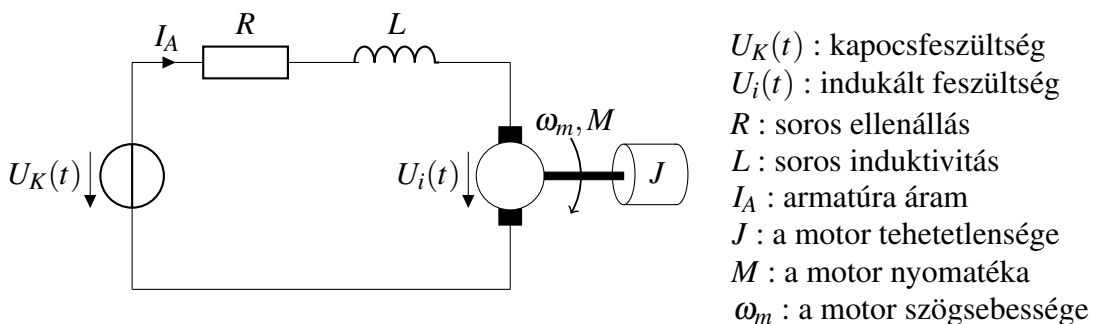
2.1. Az inverz inga felépítése

Az inverz ingám két keréken egyensúlyozó robot lesz, lényegében egy lekicsinyített segway. Első lépésben a robot feladata csupán az egyensúly tartása lesz, később szeretném ha előre-hátra is menne. A két kerékhez csatlakozni fog egy-egy darab egyenáramú motor. A motoroknál feltételezem, hogy ugyanazt a vezérlőjelet kapják meg, vagyis egyszerre fognak mozogni. A motorok méretük ellenére nagy nyomatékot képesek kifejteni, hogy később akár különböző súlyokat is el tudjon szállítani a robot. Azért alkalmaztam egyenáramú motort, mert könnyebb vezérelni, mint például egy aszinkron motort. A inverz inga tartalmazni fog két szenzort, gyorsulásmérőt és giroszkópot.

2.2. Egyenáramú motor modellezése

A motor alapvető fontosságú eleme az inverz inga rendszernek, hiszen rajta keresztül történik a beavatkozás. A motor modellezéséhez a kiindulópontot az alapvető elektromechanikai összefüggések adják.

2.2.1. Egyenáramú motor fizikai háttere



2.1. ábra. Egyenáramú motor modell

Az indukált feszültséget megkapjuk a *Lenz törvény* alapján:

$$U_i = c \phi_p n = K_i n \quad (2.1)$$

Ahol a c egy állandó, a ϕ_p a pólusfluxus, n pedig a motor fordulatszáma. A későbbiek folyamán a $c \phi_p$ szorzatot K_i -vel fogom jelölni.

Az armatúra áram és az indukált feszültség szorzata adja a gép összes villamos teljesítményét ($P_{villamos}$). A gép motorüzemben villamos energiát alakít át mechanikai teljesítménnyé, így felírható az alábbi képlet:

$$P_{mech} = P_{villamos} \quad (2.2)$$

A (2.2) egyenletet tagjait tovább bontva ezt kapjuk,

$$M 2 \pi n = U_i I_a \quad (2.3)$$

ahol M jelenti a motor nyomatékát.

A (2.3) egyenletből kifejezzük a nyomatékot:

$$M = \frac{U_i I_a}{2 \pi n} \quad (2.4)$$

A (2.4) képletbe behelyettesítve a (2.1) a Lenz törvényt az alábbi összefüggést kapjuk, ez lesz a *Biot-Savart* törvény:

$$M = \frac{c \phi_p n I_a}{2 \pi n} = \frac{c \pi p I_a}{2 \pi} = k \phi_p I_a \quad (2.5)$$

A továbbiak folyamán a $k \phi_p$ tagot egy K konstanssal fogom helyettesíteni, amelynek a neve nyomatékállandó.

Az előbbi ábra (2.1) alapján felírható az alábbi huroktörvény:

$$U_k = I_a R + L \dot{I}_a + U_i \quad (2.6)$$

A motor tengelyére ható nyomatékok:

$$M = M_{terheles} + J_m \dot{\omega}_m + b \omega_m \quad (2.7)$$

ahol a J a motor tehetetlenségi nyomatéka, $b \omega$ pedig súrlódási veszteség, amelyet a

későbbi modellezés során elhanyagoltam. Fel lehet írni a következőt:

$$M = J_m \dot{\omega}_m + M_{terheles} \quad (2.8)$$

A következő lépésben kifogom fejteni a motor nyomatékát a kapocsfeszültség és az indukált feszültség függvényeként, mivel erre a kifejezésre később még szükségem lesz.

A motorban elhelyezkedő tekercs induktivitása rendkívül kicsi szokott lenni, így most az egyszerűség kedvéért nullának tekintem az értékét. Így ha a (2.1) ábra alapján felírok egy hurokegyenletet az előbbi egyszerűsítés után, ezt az egyenletet kapom:

$$U_k = U_i + I_a R_a \quad (2.9)$$

A (2.9) kifejezésből kifejezem az armatúra áramot:

$$I_a = \frac{U_k - U_i}{R_a} \quad (2.10)$$

Az előző egyenletet (2.10) felhasználva átírtam a *Biot-Savart* törvényből kapott kifejezést:

$$M = k \phi_p \left(\frac{U_k - U_i}{R_a} \right) \quad (2.11)$$

Így megkaptam a motor nyomatékát a kapocsfeszültség és az indukált feszültség függvényeként.

2.2.2. Egyenáramú motor állapotter modellje

Ebben a fejezetben célom felírni az általam használt motorhoz hasonló motor állapotter modelljét, mivel a későbbi modellezések során szükségem lesz egy pontos motor modellre. Az általam használt motornak sajnos nincsen meg minden paramétere, amely a pontos modellezéshez szükséges, így nem tudnám ellenőrizni egy esetleges jó motor modell működését.

A választásom egy a MAXON cég által gyártott A-max 22 12V/6W típusú motorjára esett, mivel ez méretben, teljesítményben is hasonló az általam használt motorhoz.

A motor állapotter modelljénél alábbi egyenletrendszert szeretném kapni:

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx + Du\end{aligned}\tag{2.12}$$

Jelen esetben a rendszer állapotváltozói az armatúra áram (I_a) és a szögsebesség (ω) lesznek, míg a rendszer bemenetei a terhelőnyomaték ($M_{terhelo}$), és a kapcsolási feszültség (U_k).

2 fő egyenletet tudunk felírni, amelyből levezetjük az állapotter modellt:

$$\begin{aligned}U_k &= I_a R + L \dot{I}_a + U_i \\ J_m \dot{\omega} &= I_a K - M_{terhelo}\end{aligned}\tag{2.13}$$

Ezeket átrendeztem a derivált tagokra:

$$\begin{aligned}\dot{I}_a &= \frac{U_k - I_a R - K_i \omega}{L} \\ \dot{\omega} &= \frac{I_a K - M_{terhelo}}{J_m}\end{aligned}\tag{2.14}$$

A fenti (2.14) egyenletekből felírható a (2.12) alakú állapotter modell:

$$\begin{aligned}\begin{bmatrix} \dot{I}_a \\ \dot{\omega} \end{bmatrix} &= \begin{bmatrix} -\frac{R}{L} & -\frac{K_i}{L} \\ \frac{K}{J_m} & 0 \end{bmatrix} \begin{bmatrix} I_a \\ \omega \end{bmatrix} + \begin{bmatrix} 0 & \frac{1}{L} \\ -\frac{1}{J_m} & 0 \end{bmatrix} \begin{bmatrix} M_{terhelo} \\ U_k \end{bmatrix} \\ \bar{y} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I_a \\ \omega \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} M_{terhelo} \\ U_k \end{bmatrix}\end{aligned}\tag{2.15}$$

2.2.3. Motor modell verifikáció

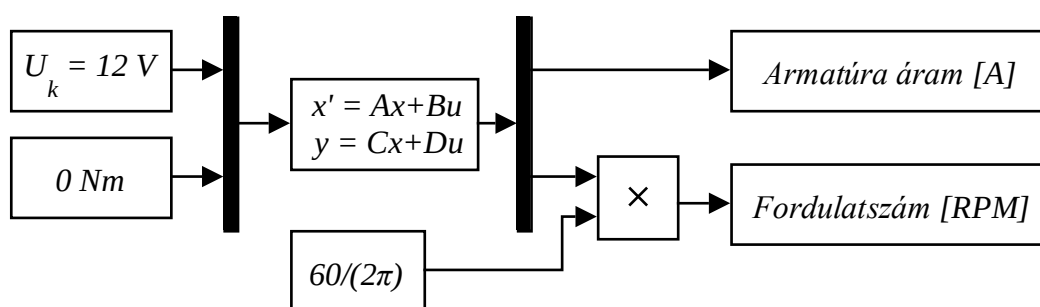
A modell helyességéről való megbizonyosodás egyik lehetséges módja, ha konkrét paraméterértékek mellett megvizsgáljuk, hogy eleget tesz-e az alapvető mérnöki elvárásoknak. Emiatt használok a modell ellenőrzéséhez a korábban megemlített MAXON motort.

A használt motor modellezéshez szükséges paraméterei:

Szimbólum	Mennyiség	Érték	Mértékegység
R	Motor soros ellenállása	6.96	Ω
L	Motor soros induktivitása	$0.445 \cdot 10^{-3}$	H
K_i	Sebességállandó inverze	1/82.73	–
K	Nyomatékállandó	$12.1 \cdot 10^{-3}$	Nm/A
J_m	A motor inerciája	$4.19 \cdot 10^{-7}$	$\text{kg} \cdot \text{m}^2$

2.1. táblázat. Választott motor adatai

A modell implementálása dinamikus szimulációk céljából a Matlab Simulink környezetben történt, a Simulink modell a (2.2) ábrán látható.

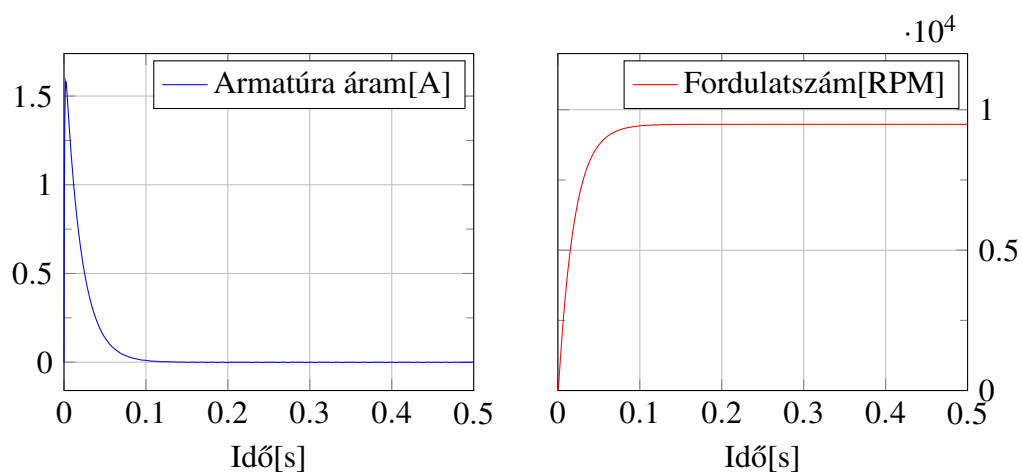


2.2. ábra. Simulink modell

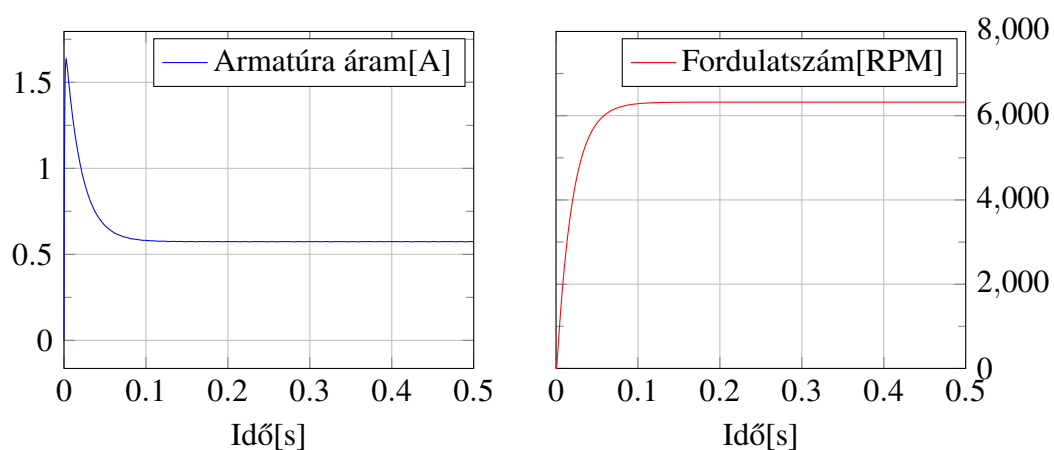
Az első szimulációban üresjárásban járatam a motort, és a cél az, hogy a motor elérje a katalóguslapján szereplő üresjárási fordulatszámot és az armatúra áramot. Mivel a modellemben elhanyagoltam a súrlódást, így a üresjárásban 0 lesz az armatúra áram értéke. A választott motor katalóguslapjából pedig kiderül, hogy a motor üresjárási fordulatszáma 9170 RPM (fordulat/perc).

A szimulációból (2.3) jól látható, hogy az armatúra áram valóban nulla lesz, míg a fordulatszám kicsit nagyobb, mint 9170 RPM, ahogy azt vártuk.

A következő szimuláció (2.4 ábra) során pedig a motor katalóguslapján szereplő névleges nyomatékot adtam meg bemenetként. A katalógus lapból kiderül, hogy erre a nyomatékra a motor armatúra árama 0.613 A, míg a fordulatszáma 6000 RPM. A szimulációm során az előírt fordulatszámnál kicsit nagyobb értéket, míg az armatúra áramra vonatkozóan kisebb értéket várok a súrlódás figyelembe vétele nélkül.



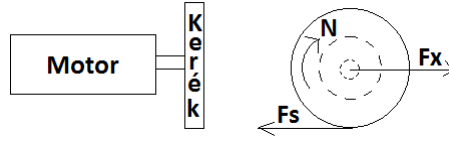
2.3. ábra. Maxon motor szimulálása üresjárásban



2.4. ábra. Maxon motor szimulálása nyomatékkal

A (2.4) ábrán jól látszik, hogy a szimuláció az előzetes elvárások szerint viselkedett. Így megállapítható, hogy a motor modellezésem megfelelő és ezt később felhasználhatom az inverz inga modellezésénél.

2.2.4. Kerék-motor kapcsolata



2.5. ábra. Kerék-motor kapcsolata

Első lépésben felírtam az ábra alapján (2.5) horizontális irányban a Newton törvényt:

$$\sum F = ma \quad (2.16)$$

$$F_x - F_s = ma \quad (2.17)$$

Ahol az F_x jelenti a kerék és a futómű közötti erőt, míg F_s pedig a súrlódási erőt. A következő lépésben felírtam a kerekre ható összes forgatónyomatékokat:

$$J_{kerek} \dot{\omega}_{kerek} = N - F_s r \quad (2.18)$$

ahol N a motor által kifejtett nyomaték, J_{kerek} a kerék tehetetlenségi nyomatéka, míg ω_{kerek} a kerék szögsebessége.

Az inverz ingát úgy építettem meg, hogy nincs semmilyen áttétel a motor és a kerék között, így kijelenthető, hogy a motor szögsebessége meg fog egyezni a kerék szögsebességével:

$$\omega_{kerek} = \omega_m \quad (2.19)$$

Az előbb leírtak szerint felírható az alábbi kifejezés is, vagyis a motor nyomatéka megegyezik a (2.5) ábrán látható N nyomatékkal:

$$N = M \quad (2.20)$$

A következő célom az, hogy átírom a kerekre felírt Newton egyenletet (2.17) úgy, hogy kapcsolatot teremtsen a motor és a kerék között. Ehhez a (2.18) egyenletbe behelyettesítettem a (2.11) kifejezést, vagyis a képletben szereplő nyomatékokat felírtam a kapocsfeszültség és az indukált feszültség függvényeként.

$$J_{kerek} \dot{\omega}_m = k \phi_p \left(\frac{U_k - U_i}{R_a} \right) - F_s r \quad (2.21)$$

A (2.21) egyenletet tovább alakítottam úgy, hogy az indukált feszültség helyére beírtam a *Lenz* törvényt és használtam a K, K_i konstansokat:

$$J_{kerek} \dot{\omega}_m = \frac{U_k K}{R} - \frac{K K_i \omega_m}{R} - F_s r \quad (2.22)$$

Az előző egyenletet (2.22) átrendeztem úgy, hogy a bal oldalon csak a súrlódási erő legyen, így a kapott eredményt majd be lehet helyettesíteni a (2.17) egyenletbe:

$$F_s = \frac{-J_{kerek} \dot{\omega}_m}{r} + \frac{U_k K}{Rr} - \frac{K K_i \omega_m}{Rr} \quad (2.23)$$

A következő lépésben jön a Newton egyenletbe való behelyettesítés, így már látható a kerék és a motor közötti összefüggés:

$$\sum F = F_x - F_s = F_x - \left(\frac{-J_{kerek} \dot{\omega}_m}{r} + \frac{U_k K}{Rr} - \frac{K K_i \omega_m}{Rr} \right) \quad (2.24)$$

Amennyiben a Newton törvényben szereplő gyorsulás helyére beírjuk a kerék pozícióját, vagyis annak a kétszeres deriváltját ($a = \ddot{x}_k$) és a tömeg helyére a kerék tömegét, akkor az alábbi egyenletet kapjuk:

$$m_k \ddot{x}_k = F_x - \left(\frac{-J_{kerek} \dot{\omega}_m}{r} + \frac{U_k K}{Rr} - \frac{K K_i \omega_m}{Rr} \right) \quad (2.25)$$

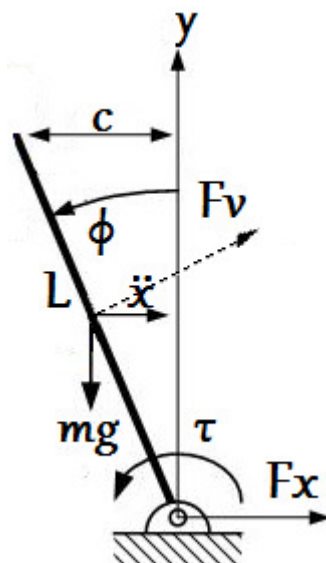
Ez az egyenlet már majdnem jól leírja a kerék-motor közti kapcsolatot, de ez csak egy darab kerékre és a két motor által kifejtett erőre vonatkozik, mivel a motorok által kifejtett nyomatékot a modellezés során egynek veszem. Az előbbieket figyelembe véve így alakul a (2.25) egyenlet:

$$2m_k \ddot{x}_k = F_x + 2 \frac{J_{kerek} \dot{\omega}_m}{r} - 2 \frac{U_k K}{Rr} + 2 \frac{K K_i \omega_m}{Rr} \quad (2.26)$$

2.3. Mechanikai modell

A következő lépésben az inverz inga vázának a fizikai viselkedését vizsgáltam és erre írtam fel különböző összefüggéseket, hogy végül megkapjak egy megfelelő mechanikai modellt.

Első lépésben a vázra ható erőket írtam fel a kerekek és a motor elhagyásával, mint ahogy a következő ábrán (2.6) látszik.



2.6. ábra. Inga vázának a modellje

A vízszintes eredő erők felírásához szükség van egy kis matematikai ügyeskedésre. Először felírom a (2.6) ábrán látható c távolságot, amely nem más, mint az inga vége és a alátámasztási ponttól húzott merőleges egyenes közötti távolság, matematikailag egy szinusz függvényt kapunk:

$$c = L \sin(\phi) \quad (2.27)$$

X_v -vel jelöltem az inga vázának aktuális pozícióját, míg x jelenti az alátámasztási pont pozícióját. Ezeket figyelembe véve, amikor az inga eldől valamelyik irányba, akkor a következő összefüggés áll fenn:

$$x_v = x - L \sin(\phi) \quad (2.28)$$

ahol x jelenti a kezdeti pozíciót, míg $L \sin(\phi)$ azt a kis távolságot, amit le kell vonni,

hogy megkapjuk az inga aktuális pozícióját.

A célom egy Newton törvény felírása, amihez pedig szükségem van a gyorsulásra, amit úgy kapok meg, hogy veszem a (2.28) egyenlet első, majd második deriváltját. Első derivált értéke:

$$\dot{x}_v = \dot{x} - \dot{\phi}L\cos(\phi) \quad (2.29)$$

Második derivált eredménye:

$$\ddot{x}_v = \ddot{x} - \ddot{\phi}L\cos(\phi) + \dot{\phi}^2L\sin(\phi) \quad (2.30)$$

A következő lépésben felírtam az előbb említett Newton törvényt a vízszintes erőkre, amely így néz ki:

$$\sum F_{vzszintes} = m_v a = m_v \ddot{x}_v \quad (2.31)$$

ahol m_v jelenti a váz tömegét. Az előző egyenletbe (2.31) pedig behelyettesítettem az előbb kiszámolt gyorsulást (2.30), így megkapva részletesen a fellépő vízszintes erőket:

$$m_v \ddot{x}_v = m_v \ddot{x} - m_v \ddot{\phi}L\cos(\phi) + m_v \dot{\phi}^2L\sin(\phi) \quad (2.32)$$

A (2.6) ábrát alaposabban megvizsgálva, kijelenthető, hogy az $m_v \ddot{x}$ szorzat nem más, mint a rajzon F_x -el jelölt vízszintes erő. Így tovább egyszerűsödik a (2.32) egyenlet az alábbiak szerint:

$$m_v \ddot{x}_v = F_x - m_v \ddot{\phi}L\cos(\phi) + m_v \dot{\phi}^2L\sin(\phi) \quad (2.33)$$

A (2.6) ábra alapján az inga tehetetlenségi nyomatékára az alábbi összefüggés írható fel:

$$J_{vaz} \ddot{\phi} = -2\tau - 2F_x L\cos(\phi) - 2F_y L\sin(\phi) \quad (2.34)$$

Alaposabban megvizsgálva a (2.6) ábrát látható, hogy az ott fellépő τ -val jelölt nyomaték nem más, mint a motor által kifejtett nyomaték. Így a (2.34) egyenlet átírható az alábbi forma szerint:

$$J_{vaz} \ddot{\phi} = -2M - 2F_x L\cos(\phi) - 2F_y L\sin(\phi) \quad (2.35)$$

A korábban tárgyalt (2.5) törvényt kicsit átalakítva felírható az alábbi egyenlet:

$$M = \frac{U_k K}{R} - \frac{K K_i \omega_m}{R} \quad (2.36)$$

Az előző (2.36) egyenletet behelyettesítettem a (2.35) kifejezés megfelelő helyére, így kapunk egy olyan egyenletet, amelyben már kapcsolatot teremt a motor és a váz között:

$$J_{vaz} \ddot{\phi} = -2 \left(\frac{U_k K}{R} - \frac{K K_i \omega_m}{R} \right) - 2F_x L \cos(\phi) - 2F_y L \sin(\phi) \quad (2.37)$$

A (2.6) ábrán van egy erő, amelyről eddig nem írtam semmit, ez a F_v , amely a váz középpontjára merőlegesen ható erők összessége.

Rich Chi Ooi dolgozata alapján[5] az inga vázára, az F_v erőre az alábbi egyenletek állnak fenn:

$$\sum F_v = m_v \ddot{x}_v \cos(\phi) \quad (2.38)$$

$$2F_x \cos(\phi) + 2F_y \sin(\phi) - m_v g \sin(\phi) - m_v L \ddot{\phi} = m_v \ddot{x}_v \cos(\phi) \quad (2.39)$$

Az előző (2.39) egyenletet megszoroztam az inga hosszával ($-L$)-el, hogy később egyszerűbb alakra tudjam hozni az egyenletet:

$$-L2F_x \cos(\phi) + -L2F_y \sin(\phi) + Lm_v g \sin(\phi) + m_v L^2 \ddot{\phi} = -Lm_v \ddot{x}_v \cos(\phi) \quad (2.40)$$

A következő lépésekben pár matematikai átalakítást fogok végrehajtani, egyenleteket vonok össze, amelynek a végső célja, hogy kapjak két olyan egyenletrendszert, amelyből majd fel lehet írni az inverz ingára vonatkozó matematikai modellt.

Ehhez első lépésben az előző (2.40) egyenletbe beírjuk a (2.37) egyenletet:

$$J_{vaz} \ddot{\phi} + 2 \left(\frac{U_k K}{R} - \frac{K K_i \omega_m}{R} \right) + Lm_v g \sin(\phi) + m_v L^2 \ddot{\phi} = -Lm_v \ddot{x}_v \cos(\phi) \quad (2.41)$$

majd pedig a (2.33) egyenletbe beillesztjük a korábban tárgyalt (2.26) kifejezést:

$$2m_k \ddot{x}_k = m_v \ddot{x}_v + m_v \ddot{\phi} L \cos(\phi) - m_v \dot{\phi}^2 L \sin(\phi) + 2 \frac{J_{kerék} \dot{\omega}_m}{r} - 2 \frac{U_k K}{Rr} + 2 \frac{K K_i \omega_m}{Rr} \quad (2.42)$$

Az előbbi két egyenletből felírható az állapottér modell, csak egy kicsit rendezgetni

kell az egyenleteket:

$$\begin{aligned}\ddot{\phi}(J_{vaz} + m_v L^2) + 2\left(\frac{U_k K}{R} - \frac{K K_i \omega_m}{R}\right) + L m_v g \sin(\phi) &= -L m_v \ddot{x}_v \cos(\phi) \\ \dot{\omega}_m(2m_k - 2\frac{J_{kere_k}}{r}) &= m_v \ddot{x}_v + m_v \ddot{\phi} L \cos(\phi) - m_v \dot{\phi}^2 L \sin(\phi) - 2\frac{U_k K}{Rr} + 2\frac{K K_i \omega_m}{Rr}\end{aligned}\quad (2.43)$$

Átrendezve az egyenleteket:

$$\begin{aligned}\ddot{\phi}(J_{vaz} + m_v L^2) &= -2\left(\frac{U_k K}{R} - \frac{K K_i \omega_m}{R}\right) - L m_v g \sin(\phi) - L m_v \ddot{x}_v \cos(\phi) \\ \dot{\omega}_m(2m_k - 2\frac{J_{kere_k}}{r}) &= m_v \ddot{x}_v + m_v \ddot{\phi} L \cos(\phi) - m_v \dot{\phi}^2 L \sin(\phi) - 2\frac{U_k K}{Rr} + 2\frac{K K_i \omega_m}{Rr}\end{aligned}\quad (2.44)$$

Az alábbi összefüggéseket ismerjük:

$$\begin{aligned}\dot{x} &= r\omega = r\dot{\theta} \\ \ddot{x} &= r\ddot{\theta}\end{aligned}\quad (2.45)$$

Az előző (2.45) egyenleteket elfogadva át tudjuk írni a (2.44) egyenletet. Egyszerűség kedvéért ezt írjuk fel:

$$\begin{aligned}\ddot{\phi}(J_{vaz} + m_v L^2) + 2\left(\frac{U_k K}{R} - \frac{K K_i \dot{x}}{Rr}\right) + L m_v g \sin(\phi) &= -L m_v \ddot{x} \cos(\phi) \\ \frac{2U_k K}{Rr} &= m_v L \ddot{\phi} \cos \phi - m_v \dot{\phi}^2 L \sin(\phi) + \frac{2K K_i \dot{x}}{Rr^2} + \ddot{x}(m_v + \frac{2I_k}{r^2} + 2m_k)\end{aligned}\quad (2.46)$$

Mivel az előző egyenletből (2.46) még nem lehet felírni a nemlineáris egyenletet, tovább kell alakítani az alábbiak szerint:

$$\begin{aligned}\ddot{\phi} &= \frac{-((2I_k + r^2(2m_k + m_v))(gL_v r R \sin(\phi) - 2K K_i \dot{x} + 2K r U_k) + L m_v r \cos(\phi)(-2K K_i \dot{x} + 2K r U_k + L m_v r^2 R \dot{\phi}^2 \sin(\phi)))}{(rR((J_v + L^2 m_v)(2I_k + r^2(2m_k + m_v)) - L^2 m_v^2 r^2 \cos^2(\phi)))} \\ \ddot{x} &= \frac{(2L m_v r \cos(\phi)(gL_v r R \sin(\phi) - 2K K_i \dot{x} + 2K r U_k) + 2(J_v + L^2 m_v)(-2K K_i \dot{x} + 2K r U_k + L m_v r^2 R \dot{\phi}^2 \sin(\phi)))}{(R(4I_k(J_v + L^2 m_v) + r^2(2J_v(2m_k + m_v) + L^2 m_v(4m_k + m_v) - L^2 m_v^2 r^2 \cos(2\phi))))}\end{aligned}\quad (2.47)$$

Ez a két egyenlet (2.47) már elegendő ahhoz, hogy Matlab-ban egy megfelelő modellt fel lehessen írni.

2.4. Nemlineáris állapotter modell

Nemlineáris állapotter általános alakja:

$$\begin{aligned}\dot{x} &= f(x, u) \\ y &= g(x, u)\end{aligned}\tag{2.48}$$

Az általam választott állapotváltozók a következők:

$$\begin{aligned}x_1 &= x && \text{(Aktuális pozíció)} \\ x_2 &= \dot{x} && \text{(Aktuális sebesség)} \\ x_3 &= \phi && \text{(Aktuális szög)} \\ x_4 &= \dot{\phi} && \text{(Aktuális szögsebesség)} \\ u &= U_k && \text{(Kapocsfeszültség)}\end{aligned}\tag{2.49}$$

Alkalmazva az előző (2.49) jelölést az alábbi formában lehet felírni a nemlineáris állapotter modellt:

$$\dot{x} = \begin{bmatrix} x_2 \\ \frac{(2Lm_v r \cos(x_3)(gLm_v r R \sin(x_3) - 2KK_i x_2 + 2Kru) + 2(J_v + L^2 m_v)(-2KK_i x_2 + 2Kru + Lm_v r^2 R x_4^2 \sin(x_3)))}{(R(4I_k(J_v + L^2 m_v) + r^2(2J_v(2m_k + m_v) + L^2 m_v(4m_k + m_v) - L^2 m_v^2 r^2 \cos(2x_3)))} \\ x_4 \\ \frac{-((2I_k + r^2(2m_k + m_v))(gL_v r R \sin(x_3) - 2KK_i x_2 + 2Kru) + Lm_v r \cos(x_3)(-2KK_i x_2 + 2Kru + Lm_v r^2 R x_4^2 \sin(x_3)))}{(rR((J_v + L^2 m_v)(2I_k + r^2(2m_k + m_v)) - L^2 m_v^2 r^2 \cos(2x_3)))} \end{bmatrix}\tag{2.50}$$

2.4.1. Input-affin alakú állapotter

A nemlineáris állapottermodellel leírható dinamikus rendszerek egy részhalmazát alkotják az ún. bemenetekben affin, vagy input affin rendszerek, amelyeket az alábbi formában lehet megadni:

$$\begin{aligned}\dot{x} &= f(x) + \sum_{i=1}^m g_i(x) u_i \\ y &= h(x)\end{aligned}$$

ahol

$$f(x) = \begin{bmatrix} f_1(x_1, \dots, x_n) \\ \vdots \\ f_n(x_1, \dots, x_n) \end{bmatrix}, \quad g_i(x) = \begin{bmatrix} g_{i1}(x_1, \dots, x_n) \\ \vdots \\ g_{in}(x_1, \dots, x_n) \end{bmatrix}$$

$$h(x) = \begin{bmatrix} h_1(x_1, \dots, x_n) \\ \vdots \\ h_p(x_1, \dots, x_n) \end{bmatrix} \text{ folytonosan differenciálható leképezések}$$

A kérdés az, hogy létezik-e olyan állapotviszacsatolás vezérlő, amely hatására a nemlineáris rendszer lineáris lesz.

Használva a korábban tárgyalt (2.49) jelölést, az alábbi alakot lehet felírni:

$$f(x) = \begin{bmatrix} x_2 \\ \frac{gL^2m_v^2r^2R\sin(x_3)\cos(x_3)-2\gamma x_2(J_v+L^2m_v)+Lm_vr^2Rx_4^2\sin(x_3)(J_v+L^2m_v)-2\gamma Lm_vrx_2\cos(x_3)}{R((J_v+L^2m_v)(2J_k+r^2(2m_k+m_v))-L^2m_v^2r^2\cos^2(x_3))} \\ x_4 \\ \frac{-2gLm_vrR\sin(x_3)(2J_k+r^2(2m_k+m_v))+8J_k\gamma x_2+4\gamma Lm_vrx_2\cos(x_3)+8\gamma m_kr^2x_2+4\gamma m_vr^2x_p-L^2m_v^2r^3Rx_4^2\sin(2x_3)}{rR(4J_k(J_v+L^2m_v)+r^2(2J_v(2m_k+m_v)+L^2m_v(4m_k+m_v))-L^2m_v^2r^2\cos(2x_3))} \end{bmatrix} \quad (2.51)$$

ahol $\gamma = KK_i$

$$g(x) = \begin{bmatrix} 0 \\ U_k \frac{(2Kr(J_v+L^2m_v)+2KLm_vr^2\cos(x_3))}{R((J_v+L^2m_v)(2J_k+r^2(2m_k+m_v))-L^2m_v^2r^2\cos^2(x_3))} \\ 0 \\ U_k \frac{(-8J_kKr-4KLm_vr^2\cos(x_3)-8Km_kr^3-4Km_vr^3)}{rR(4J_k(J_v+L^2m_v)+r^2(2J_v(2m_k+m_v)+L^2m_v(4m_k+m_v))-L^2m_v^2r^2\cos(2x_3))} \end{bmatrix} \quad (2.52)$$

3. Modell analízis és szimuláció

A modell felírása után a következő feladat annak megvizsgálása, hogy eleget tesz-e az alapvető mérnöki elvárásoknak. A dolgozat későbbi fejezeteiben alkalmazott model alapú szabályzási módszerek nagyrésze nem alkalmas arra, hogy közvetlenül a nemlineáris modellre alkalmazzuk őket, ezért a modell lokálisan linearizált változójának meghatározása is szükséges a linearizált modell analízisével együtt.

3.1. Modell verifikáció

A modell verifikációja alatt azt az ellenőrzési lépést értjük, amikor a modellt kitesszük a józan ész és a mérnöki elvárások próbatételének. Gyakorlatilag egy olyan bemeneti jel (gerjesztés) hatásának tesszük ki, amelyre ismerjük a valódi rendszer (kvalitatív) válaszát.

3.1.1. Motor verifikáció

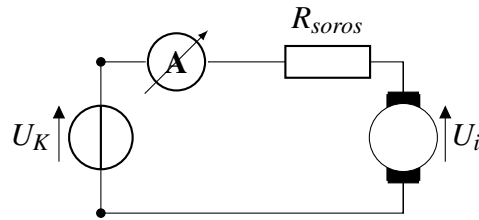
Az inverz ingámhoz két darab egyenáramú motort alkalmaztam, amelyekről csak az alábbi információt sikerült kiderítenem:

Mennyiség	Érték	Mértékegység
Üresjárási fordulatszám 6V-on	175	RPM
Üresjárási armatúra áram 6V-on	250	mA
Indító áram 6V-on	2500	mA
Indító nyomaték 6V-on	0.388	Nm
Üresjárási fordulatszám 12V-on	350	RPM
Üresjárási armatúra áram 12V-on	300	mA
Indító áram 12V-on	5000	mA
Indító nyomaték 12V-on	0.77	Nm

3.1. táblázat. Használt motor adatai

A (3.1) táblázatból látható, hogy a motorról nincsen meg minden szükséges információ, amelyek kellenek a modellezéshez (2.2.1) alfejezet. Szükségem van a motor soros ellenállására R , a K , valamint a K_i konstansokra. Emiatt összeraktam az (3.1) ábrán látható kapcsolást és kimértem a szükséges adatokat. A mérés során a tekercs értékét elhanyagolhatóan kicsinek tekintettem ($L = 0$) továbbá azt feltételeztem, hogy

$$U_k = U_i \quad (3.1)$$



3.1. ábra. A mérés elrendezése

vagyis az indukált feszültség értéke megegyezik a kapocsfeszültség értékével. A motor soros ellenállását az előbbi egyszerűsítésekkel egy egyszerű huroktörvény segítségével fel lehet írni:

$$R_a = \frac{U_i}{I_a} \quad (3.2)$$

Méréseim során a soros ellenállását 12 V-on mértem, mivel a motoromat később ezen a feszültségen fogom járatni. Több mérés után végül a soros ellenállás értékét 46.154Ω vettem.

A következő lépésben fél voltonként megmértem az armatúra áram értékét és a fordulatszámot. A mért értékek a függelékben megtalálhatóak (A.4.1). A (A.4.1) táblázatban szereplő nyomaték az alábbi képletből lettek kiszámolva:

$$P = M\omega \quad (3.3)$$

A teljesítményt és a fordulatszámot tudtam, így nem volt nehéz dolgom kiszámolni a nyomatékot.

A következő lépésben pedig kiszámoltam a két konstans (K és K_i). Egyiket a *Lenz* törvényből (2.1), míg a másikat a *Biot-Savart* törvényből (2.5) számoltam ki. A részletes eredmények a függelékben találhatóak.

A mérések és a számítások után az alábbi értékeket fogom tovább használni a modellezés során:

$$K = 0.306 \text{ Nm/A} \quad (3.4)$$

$$K_i = 1.924 \text{ Vs/rad} \quad (3.5)$$

$$R = 46.154 \text{ } \Omega \quad (3.6)$$

3.1.2. Nemlineáris modell verifikációja

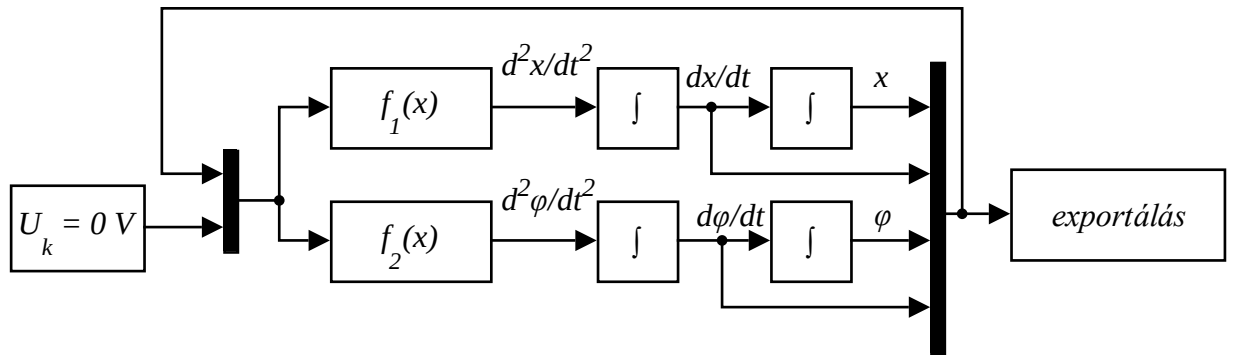
Szükséges volt a nemlineáris modell elkészítése is, mivel így lehet leellenőrizni a későbbi modellek helyességét. Ezen kívül pedig nemlineáris alapú szabályozók is használják ezt a modellt.

A nemlineáris modellem az alábbi két egyenletből indul ki:

$$\ddot{\phi}(J_{vaz} + m_v L^2) + 2\left(\frac{U_k K}{R} - \frac{\dot{x} K K_i}{R r}\right) + L m_v g \sin \phi = -L m_v \ddot{x} \cos \phi \quad (3.7)$$

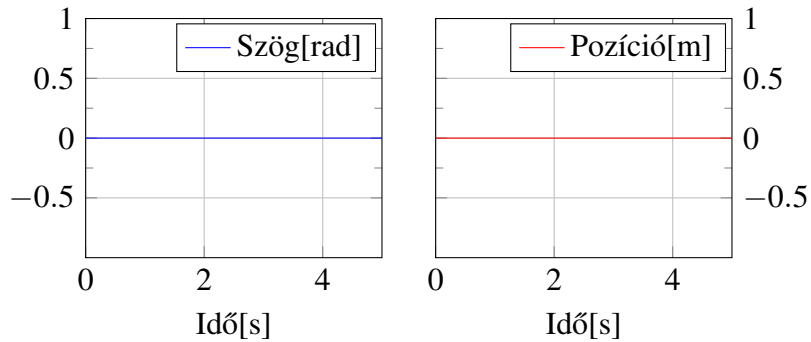
$$\frac{2U_k K}{R r} = m_v L \ddot{\phi} \cos \phi - m_v \dot{\phi}^2 L \sin \phi + \frac{2K K_i \dot{x}}{R r^2} + \ddot{x} \left(m_v + \frac{2J_{kerekek}}{r^2} + 2m_k\right)$$

A modellt összeraktam Matlab Simulinkben (3.2 ábra).

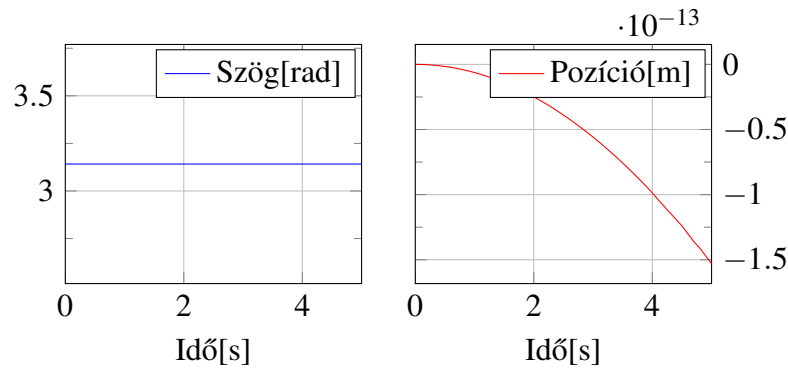


3.2. ábra. Nemlineáris modell simulinkben

Miután összeállítottam a modellt, a helyességét ellenőriztem. Amennyiben π és 0 voltak a kezdeti szög értékek, és a bemenetet szintén nullának vettem, akkor stabil maradt a rendszer, amit láthatunk is a következő ábrákon.



3.3. ábra. Nemlineáris modell 0 kezdeti érték esetén



3.4. ábra. Nemlineáris modell π kezdeti érték esetén

3.2. Lokális linearizálás

A cél az, hogy az inverz inga függőleges helyzetben maradjon. Ehhez szükség van egy szabályozóra, amely az ingát és az egyensúlyi pontot stabilizálja. Ehhez először is meg kell állapítani az egyensúlyi pontokat a (2.50) egyenletrendszer felhasználásával.

Egyensúlyi pontok ott lesznek ahol az $\dot{x} = 0$ egyenlet teljesül, vagyis:

$$\dot{x} = 0 = f(u = 0) \quad (3.8)$$

Így tovább egyszerűsödik a (2.50) egyenletrendszer:

$$\dot{x} = \begin{bmatrix} x_2 \\ \frac{2gL^2mv^2r^2\sin(\phi)\cos(\phi)}{4Jk(Jv+L^2mv)+r^2(2Jv(2mk+mv)+L^2mv(4mk+mv))-L^2mv^2r^2\cos(2\phi)} \\ x_4 \\ \frac{(2Jk+r^2(2mk+mv))(2KKixp-gLmvrR\sin(\phi))-Lmvr\cos(\phi)(Lmvr^2R\phi^2\sin(\phi)-2KKixp)}{rR((Jv+L^2mv)(2Jk+r^2(2mk+mv))-L^2mv^2r^2\cos^2(\phi))} \end{bmatrix} \quad (3.9)$$

Mivel a cél az, hogy a $\dot{x}$ nullával legyen egyenlő (3.8), így már megállapítható, hogy a (3.9) egyenletrendszerben szereplő x_2 és x_4 állapotváltozók értékei nullával egyeznek meg. Ezt elfogadva a (3.9) kifejezés még tovább egyszerűsödik az alábbi forma szerint:

$$\dot{x} = \begin{bmatrix} 0 \\ \frac{2gL^2mv^2r^2\sin(\phi)\cos(\phi)}{4Jk(Jv+L^2mv)+r^2(2Jv(2mk+mv)+L^2mv(4mk+mv))-L^2mv^2r^2\cos(2\phi)} \\ 0 \\ -\frac{gLmv\sin(\phi)(2Jk+r^2(2mk+mv))}{(Jv+L^2mv)(2Jk+r^2(2mk+mv))-L^2mv^2r^2\cos^2(\phi)} \end{bmatrix} \quad (3.10)$$

Az előző egyenletrendszert (3.10) megoldva az alábbi megoldásokat kapjuk, amelyek az alábbiak: A fenti két megoldásból az egyik az amikor a $\phi = 0$, a másik

$$\begin{aligned} \hat{x}^1 &= [0, 0, 0, 0]^T \\ \hat{x}^2 &= [0, 0, \pi, 0]^T \end{aligned} \quad (3.11)$$

esetben $\phi = \pi$. Nekem az utóbbi eredményre van szükségem, mert ebben az esetben fog felfelé állni az inga.

Az A és a B mátrix kiszámolásához az alábbi Jacobi-mátrixot kell kiszámolni:

$$\begin{aligned} A &= \frac{\partial f}{\partial x}(x_0, f_0) \\ B &= \frac{\partial f}{\partial u}(x_0, f_0) \end{aligned} \quad (3.12)$$

Célom (3.2) alakú lineáris időinvariáns állapotter modell létrehozása:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix} [U_k] \quad (3.13)$$

A deriválásokat elvégezve az alábbi eredményeket kaptam:

$$\begin{aligned}
 A_{11} &= A_{13} = A_{14} = A_{21} = 0 \\
 A_{12} &= 1 \\
 A_{22} &= \frac{2KK_i Lmvr - 2KK_i (J_v + L^2 m_v)}{R((J_v + L^2 m_v)(2J_k + r^2(2m_k + m_v)) - L^2 m_v^2 r^2)} \\
 A_{23} &= \frac{gL^2 m_v^2 r^2}{(J_v + L^2 m_v)(2J_k + r^2(2m_k + m_v)) - L^2 m_v^2 r^2} \\
 A_{24} &= A_{31} = A_{32} = A_{33} = 0 \\
 A_{34} &= 1 \\
 A_{41} &= 0 \\
 A_{42} &= \frac{8J_k K K_i - 4K K_i Lmvr + 8K K_i m_k r^2 + 4K K_i m_v r^2}{rR(4J_k(J_v + L^2 m_v) + r^2(2J_v(2m_k + m_v) + L^2 m_v(4m_k + m_v)) - L^2 m_v^2 r^2)} \\
 A_{43} &= \frac{2gLm_v(2J_k + r^2(2m_k + m_v))}{4J_k(J_v + L^2 m_v) + r^2(2J_v(2m_k + m_v) + L^2 m_v(4m_k + m_v)) - L^2 m_v^2 r^2} \\
 A_{44} &= 0
 \end{aligned} \tag{3.14}$$

$$\begin{aligned}
 B_1 &= 0 \\
 B_2 &= \frac{2Kr(J_v + L^2 m_v) - 2KLm_v r^2}{R((J_v + L^2 m_v)(2J_k + r^2(2m_k + m_v)) - L^2 m_v^2 r^2)} \\
 B_3 &= 0 \\
 B_4 &= \frac{-8J_k K r + 4K Lm_v r^2 - 8K m_k r^3 - 4K m_v r^3}{rR(4J_k(J_v + L^2 m_v) + r^2(2J_v(2m_k + m_v) + L^2 m_v(4m_k + m_v)) - L^2 m_v^2 r^2)}
 \end{aligned} \tag{3.15}$$

Tovább alakítva az egyenletrendszert megkapható az alábbi lineáris állapottér modell:

$$\begin{aligned}
 \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{(2KK_i(Lm_v r - L^2 m_v - J_v))}{\beta} & \frac{(gL^2 m_v^2 r^2 R)}{\beta} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{(\frac{2KK_i}{rR}(\frac{Lm_v}{r} - \frac{2I_k}{r^2} - 2m_k - m_v))}{\alpha} & \frac{(gLm_v(-2m_k - m_v - \frac{2I_k}{r^2}))}{\alpha} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \\
 &+ \begin{bmatrix} 0 \\ \frac{(2Kr(J_v + L^2 m_v - Lm_v r))}{\beta} \\ 0 \\ \frac{(\frac{2K}{R}(2m_k + m_v + \frac{2I_k}{r^2} - \frac{Lm_v}{r}))}{\alpha} \end{bmatrix} [U_k]
 \end{aligned} \tag{3.16}$$

Ahol α és β az alábbi értékeket jelenti:

$$\begin{aligned}
 \alpha &= L^2 m_v^2 - (J_v + L^2 m_v)(2m_k + m_v + \frac{2I_k}{r^2}) \\
 \beta &= R(2I_k J_v + 2I_k L^2 m_v + 2J_v m_k r^2 + J_v m_v r^2 + 2L^2 m_k m_v r^2)
 \end{aligned} \tag{3.17}$$

Ez a modell már használható Matlab szimulációkhoz.

3.2.1. Lokálisan linearizált modell implementációja

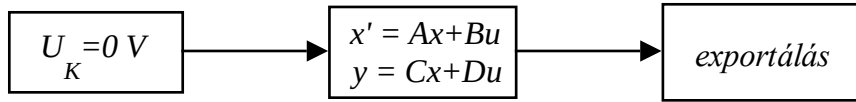
A linearizálás több szabályozóhoz kell. Ilyen például a pólusáthelyező vagy az LQ szabályozó. Ezek később lesznek kifejtve.

Miután a linearizálást végrehajtottam a π körül az alábbi állapotter modellt tudtam felírni:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{(2KK_i(Lm_v r - L^2 m_v - J_v))}{\beta} & \frac{(gL^2 m_v^2 r^2 R)}{\beta} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{(\frac{2KK_i}{rR}(\frac{Lm_v}{r} - \frac{2I_k}{r^2} - 2m_k - m_v))}{\alpha} & \frac{(gLm_v(-2m_k - m_v - \frac{2I_k}{r^2}))}{\alpha} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} +$$

$$+ \begin{bmatrix} 0 \\ \frac{(2Kr(J_v + L^2 m_v - Lm_v r))}{\beta} \\ 0 \\ \frac{(\frac{2K}{R}(2m_k + m_v + \frac{2I_k}{r^2} - \frac{Lm_v}{r}))}{\alpha} \end{bmatrix} \begin{bmatrix} U_k \end{bmatrix}$$

A linearizált modell Matlab Simulinkben így néz ki összehúzalozva:



3.5. ábra. Lineáris modell

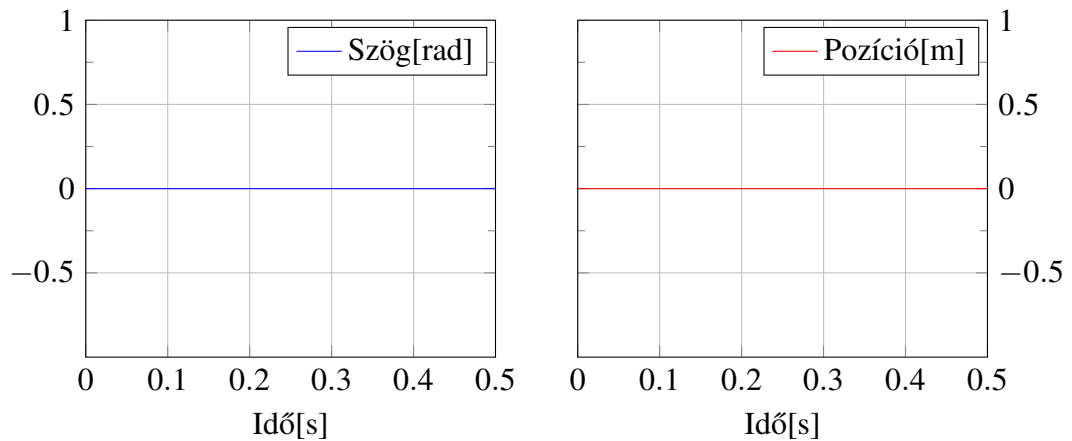
A lineáris modell a következő kezdeti értékek esetén stabilan viselkedik:

$$\begin{aligned} U_k &= 0 \\ x &= 0 \\ \dot{x} &= 0 \\ \phi &= 0 \\ \dot{\phi} &= 0 \end{aligned}$$

Mivel a rendszert már linearizáltam a π -ben, így a 0 kezdetű szögérték π -vel egyezik meg.

Íme a szimuláció eredménye:

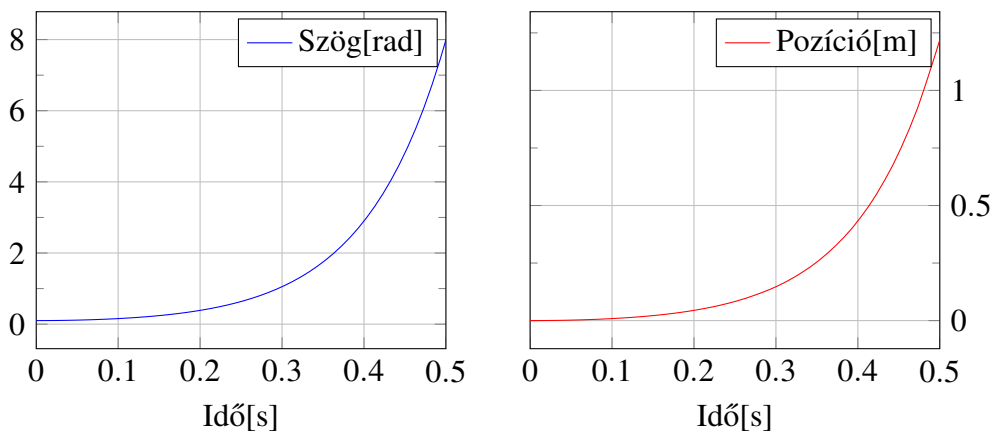
Amennyiben 0 kezdeti szögérték helyett más értéket adunk meg, akkor már nem lesz stabil a modell.



3.6. ábra. Lineáris modell szimulációja 0 kezdeti értékek esetén

A következő szimulációban a kezdeti szögértéke 6° körül volt.

Mint a (3.7) ábrán látható 6° fok kezdeti érték esetén már nem lesz stabil a rendszer,

3.7. ábra. Lineáris modell szimulációja 6° kezdeti értékek esetén

ami várható is volt.

3.3. Lokális dinamikus analízis

A lokális analízis célja, hogy kiderüljön, alkalmazhatók-e a következő fejezetekben lineáris technikák. Az alkalmazhatósági feltételek az irányíthatóság és a megfigyelhetőség, de emellett az aszimptotikus stabilitást is megvizsgálom.

3.3.1. Aszimptotikus stabilitás

Egy lineáris időinvariáns rendszer (LTI) pontosan akkor aszimptotikusan stabil, hogy ha az A mátrix valamennyi sajátértékének valós része negatív. Ebből a szempontból két fő esetünk lehet:

- Az A mátrix minden sajátértékének valós része negatív, ekkor beszélünk: *aszimptotikus stabilitásról*
- Az A mátrixnak van pozitív valós részű sajátértéke, ekkor beszélünk *instabilitásról*

Az általam létrehozott lineáris modell A mátrixának vettem a sajátértékeit a (3.2)-ban kapott egyensúlyi pontokban, és az alábbi eredményt kaptam:

$$\text{eig}(A) = [0; -0.0040; -10.1513; 10.1485]^T \quad (3.18)$$

Mint látható a (3.18) kifejezésben, van egy pozitív tagja az A mátrix sajátértékeinek. Így megállapítható, hogy a Π körül linearizált rendszer nem stabil.

A 0 körül linearizált rendszert esetén az előzetes elvárás az, hogy az inverz inga a 0-ban stabil legyen. A függelékben megtalálható a 0 körül linearizált rendszer A mátrixa. Ennek az A mátrixnak a sajátértékei a következők:

$$\begin{aligned} \text{eig}(A) = [& 0.0000 + 0.0000i \\ & -6.1229 + 0.0000i \\ & -3.4044 + 7.0528i \\ & -3.4044 - 7.0528i] \end{aligned} \quad (3.19)$$

A (3.19) jól látható, hogy a sajátértékek valós része negatív, vagyis a robot a 0-ban aszimptotikusan stabil, ahogy várható is volt.

3.3.2. Irányíthatóság

A rendszer (állapot) irányítható, ha állapotvektora az u irányítás hatására tetszőleges $x(t_0)$ kezdeti állapotból véges $(t_v - t_0)$ idő alatt a tetszőlegesen előírt $x(t_v)$ állapotba vihető át.

Az irányítás feltétele, hogy az irányíthatósági mátrix (M_c) oszlopai lineárisan függetlenek, vagyis az M_c mátrix invertálható.

$$M_c = [B \quad AB \quad \dots \quad A^{n-1}B] \in R^{n \times n} \quad (3.20)$$

Az irányíthatósági mátrix numerikus értéke:

$$M_c = \begin{bmatrix} 0 & 0.0876 & -0.0006 & 3.6550 \\ 0.0876 & -0.0006 & 3.6550 & -0.0493 \\ 0 & 0.2292 & -0.0015 & 23.6146 \\ 0.2292 & -0.0015 & 23.6146 & -0.2235 \end{bmatrix}$$

A mátrix teljes rangú, azaz a (3.2) lineáris modell irányítható. Az invertálást sikerült végrehajtani, így kijelenthető, hogy ez a rendszer irányítható.

3.3.3. Megfigyelhetőség

A megfigyelhetőség arra ad választ, hogy egy ismeretlen állapotú rendszer bemenő- és kimenőjelének bizonyos ideig tartó mérése után rekonstruálható-e a mérés kezdetekor fennálló állapot. A rendszer megfigyelhető, ha a $t_0 < t < t_v$ intervallumban megfigyelt $y(t)$ és $u(t)$ jelekből $x(t_0)$ meghatározható.

A megfigyelhetőség feltétele, hogy a megfigyelhetőségi mátrix (M_o) teljes rangú legyen, azaz annyi a rangja, mint az állapottér dimenziója, jelen esetben 4.

M_o mátrix:

$$M_o = [C \quad CA \quad \dots \quad CA^{n-1}]^T \in \mathbb{R}^{n \times n} \quad (3.21)$$

Megvizsgálva ennek a mátrixnak a rangját, azt kapjuk, hogy az eredmény 4.

$$\text{rank}(M_o) = 4$$

Megállapíthatjuk, hogy a (3.2) linearizált modell a (3.2) kifejezés 2. munkapontjában együttesen irányítható és megfigyelhető.

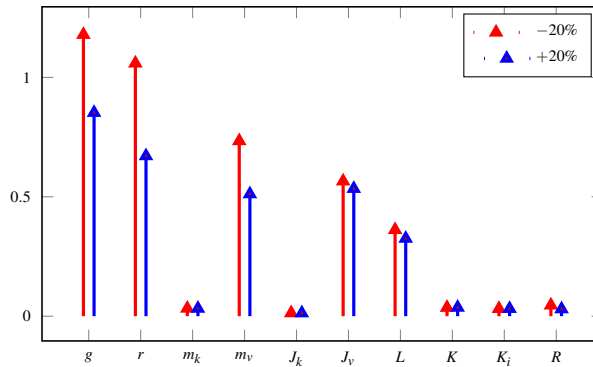
3.4. Paraméterérzékenység vizsgálat

A vizsgálat fő célja, hogy a későbbi paraméter becsléshez kiválasszam azokat a paramétereket, melyeket esélyünk van megbecsülni bemenet-kimenet mérésekből. A modellemben összesen tíz darab paraméter található ($g, r, m_k, m_v, J_k, J_v, L, K, K_i, R$) amelyeknek az értékeit egyesével megváltoztattam először $\pm 20\%$, majd $\pm 30\%$ -al, hogy lássam, mely paraméterekre érzékeny a modellem. A szimulációk során kapott jelek időfüggvényei (A.5) függelékben találhatóak. A jobb átláthatóság miatt mindössze két ábrán mutatom be a paraméterérzékenység vizsgálat eredményeit. Ezt egy alkalmas jelnorma (3.22) segítségével teszem meg

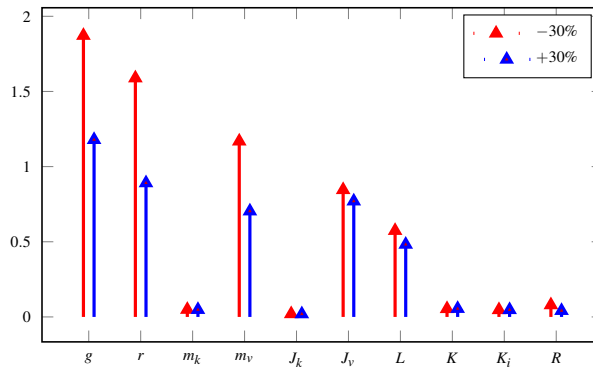
$$V_{ij} = \frac{\sqrt{\int_0^\tau |\hat{x}_{ij}(\tau) - \bar{x}(\tau)|^2 d\tau}}{\sqrt{\int_0^\tau |\bar{x}(\tau)|^2 d\tau}} \quad (3.22)$$

,ahol az $\bar{x}$ jelenti a nominális értéket, míg a $\hat{x}$ jelen esetben a növelt értéket. A (3.22) képletben az i index jelöli a különböző paramétereket ($i \in \{g, r, m_k, m_v, J_k, J_v, L, K, K_i, R\}$),

míg az j index jelöli, hogy csökkentett vagy növelt értékről beszélünk ($j \in \{+, -\}$). A (3.22) képlet integrálja a különbség négyzetét, majd a gyökét veszi, amelyet leosztok a nominális értékkel, így kapva egy egységnyi számot. Ez a szám már alkalmas arra, hogy megfelelő módon jellemezze a paraméter érzékenységét a modellben. A paraméterérzékenység vizsgálatot a nyílt hurkú (vagyis a szabályozó nélküli) rendszerre végeztem el.



3.8. ábra. $\pm 20\%$ esetén a jelnorma alakulása



3.9. ábra. $\pm 30\%$ esetén a jelnorma alakulása

A (3.8) és (3.9) ábra alapján a következő megállapítást tehetjük: a modell érzékeny a gravitációs gyorsulásra (g), a kerék sugarának nagyságára (r), a váz tömegére (m_v), a váz tehetetlenségére (J_v) és végül az inga hosszára (L). A továbbiakban ebből az öt paraméterből csak négyet kell vizsgálnom, mivel a gravitációs gyorsulás értéke nem fog változni a kísérleteim során.

3.5. Kísérlettervezés egy későbbi paraméterbecsléshez

A kísérlettervezés célja a paraméterbecslés szempontjából optimális mérési körülmények megválasztása. A mintavételezési idő megválasztásánál arra törekszünk,

hogy elegendően gyors mintavételezést biztosítsunk, ezért azt úgy célszerű megválasztani, hogy az a leggyorsabb (legkisebb) karakterisztikus időnek legfeljebb az egynegyede legyen. Másrészt a mérési rekordnak elegendően hosszú időt le kell fednie, hogy a leglassabb időállandójú részdinamika is megjelenjen a mérésekben, azaz a mintaelemszám és a mintavételezési idő szorzatát úgy kell megválasztani, hogy az a leglassúbb időállandójának legalább a négyszerese legyen. A rendszerre adott bemenettel szemben az a legfőbb elvárás, hogy ún. elegendő gerjesztést biztosítson [REF], ami a jelen esetben egy álvéletlen bináris jelsorozat (PRBS) lesz. [10] Az (3.19) kifejezésből kiszámoltam a legnagyobb és a legkisebb időállandókat, amelyek a következők:

$$\begin{aligned}t_1 &= 0,163 \text{ sec} \\ t_2 &= 0,2937 \text{ sec}\end{aligned}\tag{3.23}$$

Így az előbb leírtak alapján a javasolt mintavételezési idő minimális értéke:

$$t_{minta} = 0.04075 \text{ sec} \approx 24,54 \text{ Hz}\tag{3.24}$$

A modell alapú szabályozótervezés implementációja előtt mindenképpen szükség lesz a modell paraméterek pontos megbecslésére. Az általam megismert módszerek közül nagy valószínűséggel egy, a nemlineáris modelleknél is használható predikciós hiba minimalizálásán alapuló módszert fogok használni [10].

4. Modell alapú szabályozás

Dinamikus rendszerek szabályozása nem feltétlenül igényli egy matematikai modell meglétét, erre remek példa a klasszikus arányos-integráló-differenciáló (PID) szabályozó, mindazonáltal a szabályozott rendszer dinamikus modelljét alapul vevő szabályzótervezési módszerek sokkal nagyobb esélyt adnak biztosítani arra, hogy jó minőségű szabályozást tevézzünk. A fejezetben két kivétellel modell alapú szabályozók kerülnek megtervezésre, ezek többsége a lineáris állapotter modell paramétereinek ismeretét igényli. Két olyan módszert is tárgyalok, melyek nem modell alapúak; az egyikük a klasszikus PID szabályozó, a másik pedig egy fuzzy szabályozó. Az előbbivel a célom, hogy összehasonlítás alapul szolgáljon. A fuzzy szabályozó alkalmazásának oka pedig egyszerűen csak az, hogy szerettem volna megismerni a lágy számítási algoritmusok irányítástechnikai alkalmazását ezen keresztül.

4.1. Szabályozási cél

A kitűzött feladat az, hogy az inverz inga (3.2) egyensúlyi pont lokális stabilizálása. Ezt különböző szabályozók használatával kívánom elérni. Végző célom, hogy ezek közötti különbséget feltárjam. Használni fogok pólusáthelyező, LQ, Fuzzy, szervó szabályozót. További cél, hogy a zárt rendszer egy időben változó alapjel követésére is alkalmas legyen.

4.2. Állapotviszacsatolás

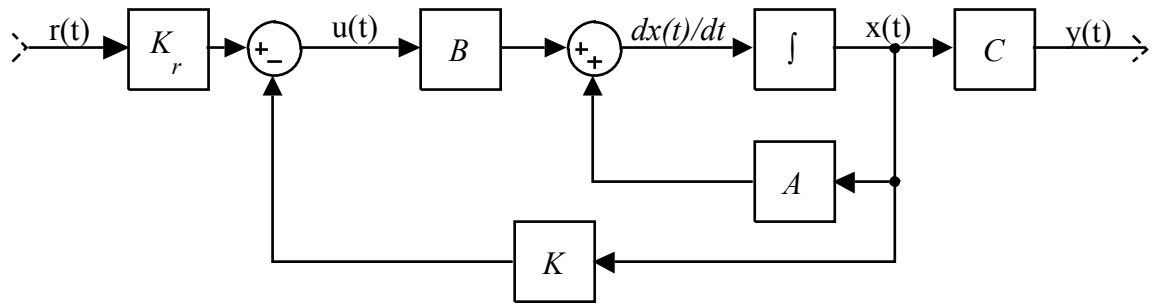
Az állapotviszacsatolást alkalmazó szabályozási körök egyik esete a pólusáthelyezés. De előbb megtárgyaljuk, hogy mit jelent az állapotviszacsatolás. Lineáris állapotviszacsatolás esetén a bemenet így alakul:

$$\begin{aligned} u &= K_R r - Kx \\ \dot{x} &= Ax + BK_R r - BKx \end{aligned} \tag{4.1}$$

Így megkapjuk a zárt rendszer lineáris állapotegyenletét:

$$\begin{aligned} y &= Cx \\ \dot{x} &= (A - BK)x + BK_R r \end{aligned} \tag{4.2}$$

A K állapot visszacsatolási paraméterrel stabilizálható a rendszer, amennyiben az irányítható. A K_r kalibrációs tényezővel a maradó szabályozási hiba küszöbölhető



4.1. ábra. Állapottér blokkvázlata

ki.

4.2.1. Maradó szabályozási hiba kiküszöbölése

Az (4.1) ábra alapján a zárt szabályozási kör r -ről y -ra vonatkozó átviteli függvénye:

$$H_{ry}(s) = \frac{Y(s)}{R(s)} = \frac{C(sI - A)^{-1}BK_R}{1 + K(sI - A)^{-1}B} \quad (4.3)$$

A K_R kalibrációs tényező, amely segítségével lehet beállítani, hogy a H_{ry} erősítése 1 legyen ($H_{ry}(0) = 1$ -ből). A felnyitott kör ugyanis jól láthatóan nem integráló jellegű, így zérus hibát és egységnyi statikus átvitelt nem tud eredményezni.

4.2.2. Pólusáthelyező

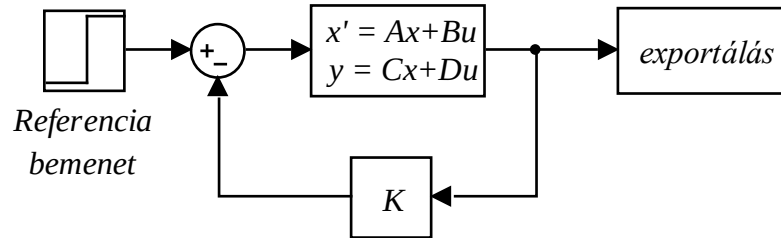
A pólusáthelyezés lényege az, hogy úgy kell megválasztani a visszacsatoló K vektort, hogy a zárt rendszerünk karakterisztikus polinomja az előírt $R(s)$ polinom legyen:

$$R(s) = (s - K_1)(s - K_2) \dots = s^n + r_1 s^{n-1} + \dots + r_{n-1} s + r_n = \prod_{i=1}^n (s - s_i) = \det(sI - A + BK)$$

A megoldás mindig létezik, hogy ha a folyamat irányítható. Irányítás alatt azt értjük, hogy keresünk egy olyan $\{u(t) \mid t_1 < t < t_2\}$ bemenetet, amivel a rendszer x állapota $x(t_1)$ -ből $t_2 - t_1$ idő alatt $x(t_2)$ -be irányítható. Nem irányítható, hogy ha nem tudunk az egyik állapotból a másikba jutni.

Első lépésben a linearizált modellen hajtottam végre a pólusáthelyezést és ezek után tértem át a nemlineáris modellre.

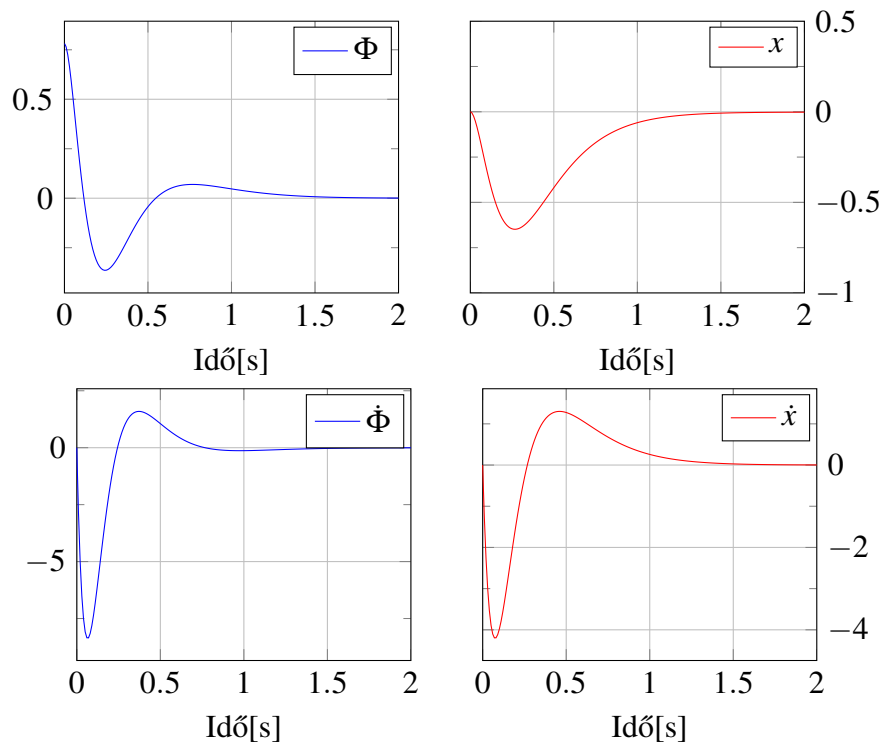
A következő ábra a Matlab-ban összehuzalozott pólusáthelyezőt mutatja:



4.2. ábra. Lineáris modell

Megfelelő visszacsatolást alkalmazva a stabilá tehető a modell. A következő szimulációban a kiindulási szög 45° (0.78 rad) lesz.

Célom a rendszer gyorsítása, ehhez olyan sajátértékekre van szükségem, amelyek a komplex síkon a képzetes tengelytől balra helyezkednek el, mint az eredeti rendszer sajátértékei (A mátrix sajátértékei). Figyelembe kell venni a visszacsatoló K vektor tervezésénél, hogy a választott pólusok jelentősen befolyásolják a zárt rendszer viselkedését. Túl nagy pólusok választása esetén a rendszer nagyobb amplitúdójú beavatkozó jelet kíván meg és a beavatkozó jelnek korlátjai vannak. Nem megfelelő



4.3. ábra. Lineáris modell állapotvisszacsatolással

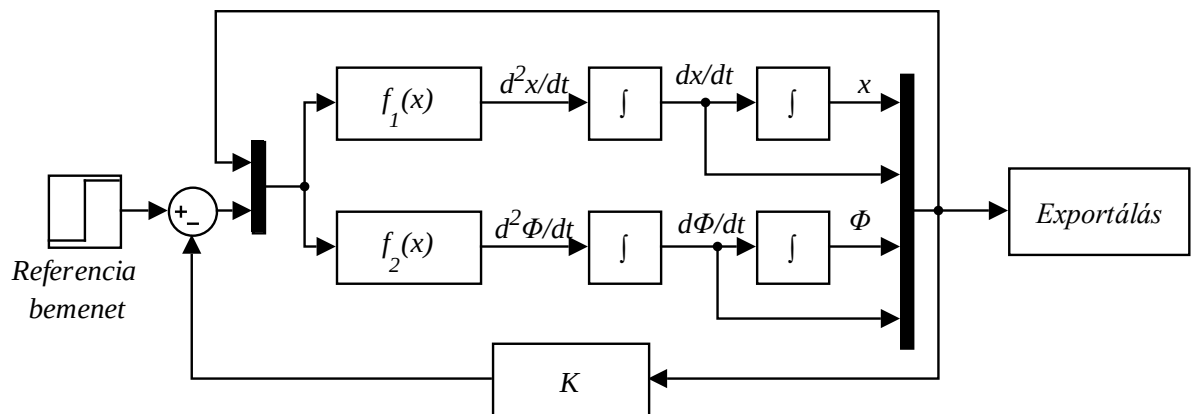
pólusok esetén a beavatkozó jel telítődik (elér egy u_{max} értéket) . Az (4.3) ábránál a választott pólusaim az alábbiak voltak:

$$[-5, -6, -10, -11]$$

A K vektor értékét a Matlab-ba beépített függvénnyel határoztam meg, amely a következő eredményt adta:

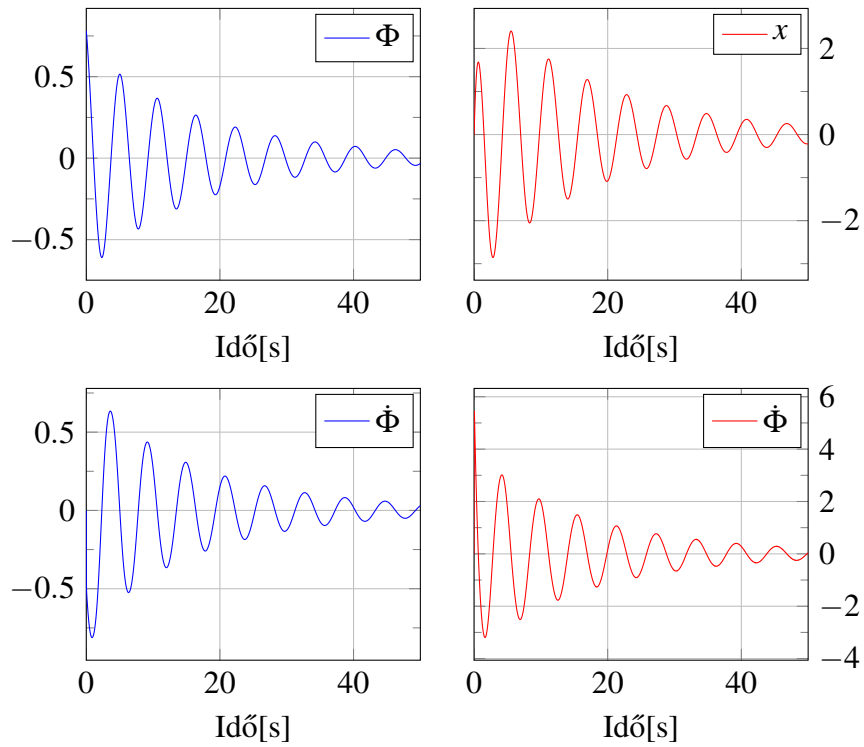
$$[-614.185, -342.532, 2302.756, 270.524] \quad (4.4)$$

A (4.2.2) értékeket tartottam meg, mikor a nemlineáris modellen is verifikáltam a korábban megtervezett pólusáthelyezőt.



4.4. ábra. Nemlineáris modell pólusáthelyezővel

A szimulációban ismét 45° volt a kiindulási érték, ezt kellett tehát stabilizálni. Az ábrán (4.5) jól látható, hogy amennyiben megfelelően választjuk meg a visszacsatoló K vektor értékét, akkor egyensúlyba lehet tartani az inverz ingát. Amennyiben túl nagy pólus értékeket választottam volna, akkor gyors lett volna a beállítás, de nagyobbak lettek volna a lengések.



4.5. ábra. Nemlineáris modell pólusát helyezővel

4.2.3. LQ szabályozó

Az LQ szabályozó az állapot visszacsatolás erősítésének optimális módon való meghatározására szolgáló módszer. Az alábbi rendszert akarjuk szabályozni:

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx\end{aligned}\tag{4.5}$$

Ehhez keresünk egy olyan u -t, ami mellett ez a kifejezés minimális:

$$J(x, u) = \frac{1}{\eta} \int_{t=0}^{\infty} [x^T(t) \cdot Q \cdot x(t) + u^T(t) \cdot R \cdot u(t)] dt\tag{4.6}$$

Ahol Q és R mátrixok súlymátrixok. A Q mátrix súlyozza a rendszer állapotait, míg az R a rendszerbe táplált energiát. Más szóval, ha a $Q > R$, akkor az állapotot jobban büntetjük, vagyis egy jobb minőségű szabályozót kapunk meg. Amennyiben a $Q < R$, akkor pedig a bemenetet büntetjük jobban, így kapva egy olcsóbb megoldást. Keresünk egy olyan $u(t)$ bemenetet a $[0, T]$ intervallumon, ami minimalizálja a $J(x, u)$ költségfüggvényt úgy, hogy közben teljesül az állapotegyenlet.

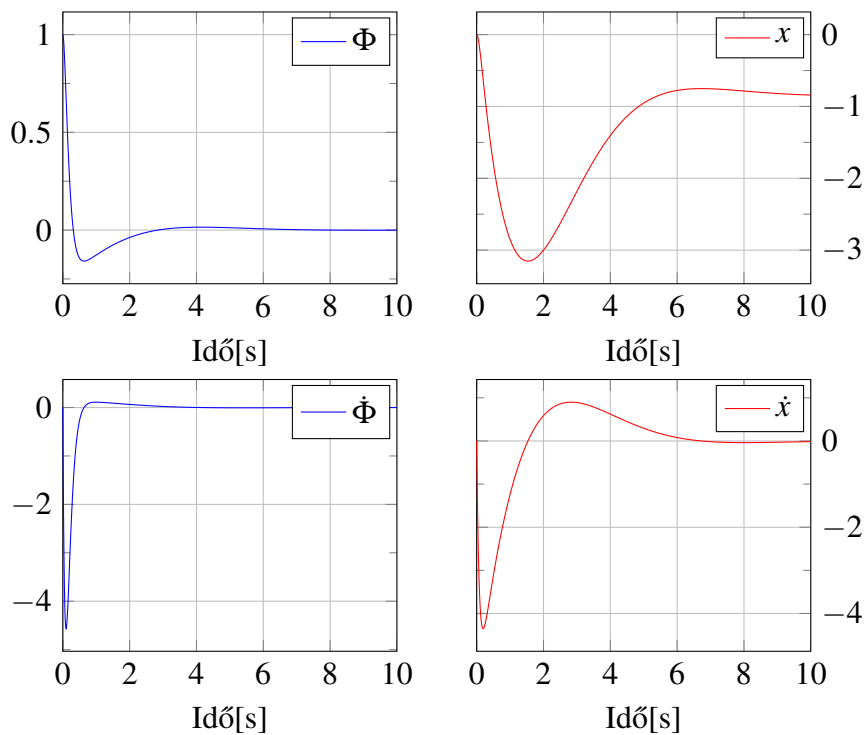
$$u(t) = -R^{-1}B^TKx(t)$$

Az LQ szabályozó a stabilizálás mellett egyidejűleg a tranziens folyamatok minőségét is tervezhetővé teszi.

Az LQ szabályozó használhatóságának feltétele, hogy a modellnek együttesen kell irányíthatónak és megfigyelhetőnek lennie. A lineáris modellem összehuzalozása ugyanúgy néz ki, mint a pólusáthelyező esetén (4.2) ábra. Az LQ szabályozó esetén én a jó minőségre törekedtem, mivel az inverz inga stabilizálásához igen pontos szabályozóra van szükség.

Az általam választott paraméterek:

$$Q = \begin{bmatrix} 2000 & 0 & 0 & 0 \\ 0 & 20 & 0 & 0 \\ 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \quad R = 10 \quad (4.7)$$



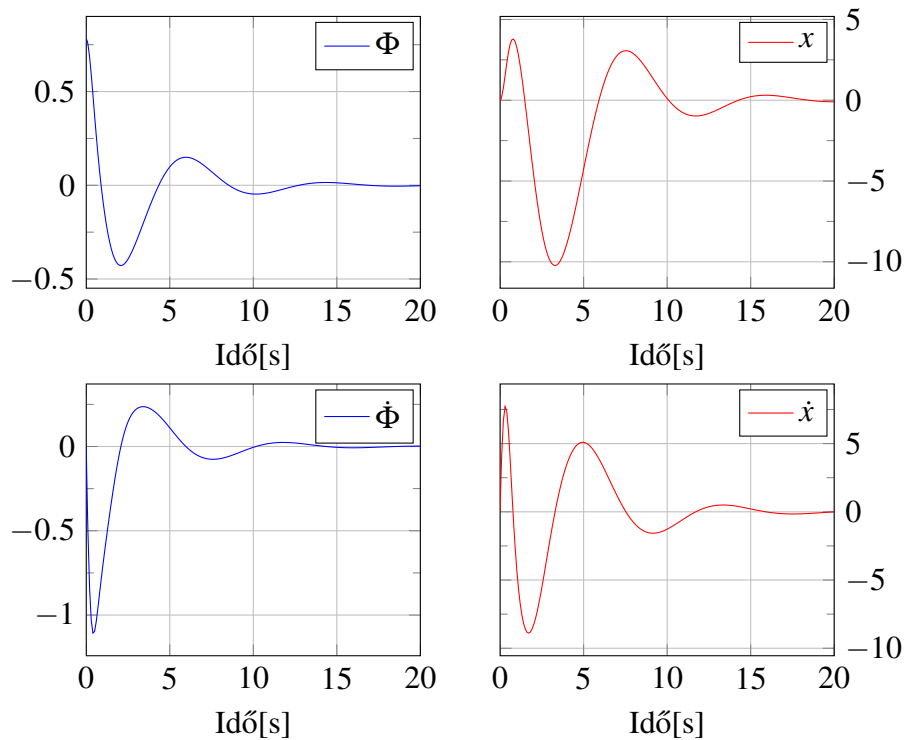
4.6. ábra. Lineáris modell LQ szabályozóval

A (4.6) ábrán jól látható, hogy az inga hamar beáll az egyensúlyi pontba és ott is marad.

Ebben az esetben a K visszacsatoló vektorom értéke az alábbi volt:

$$[-14.142, -26.25, 1015.63, 103.89]$$

Miután megterveztem a lineáris modellre az LQ szabályozót, utána verifikáltam a szabályozót a nemlineáris modellen.

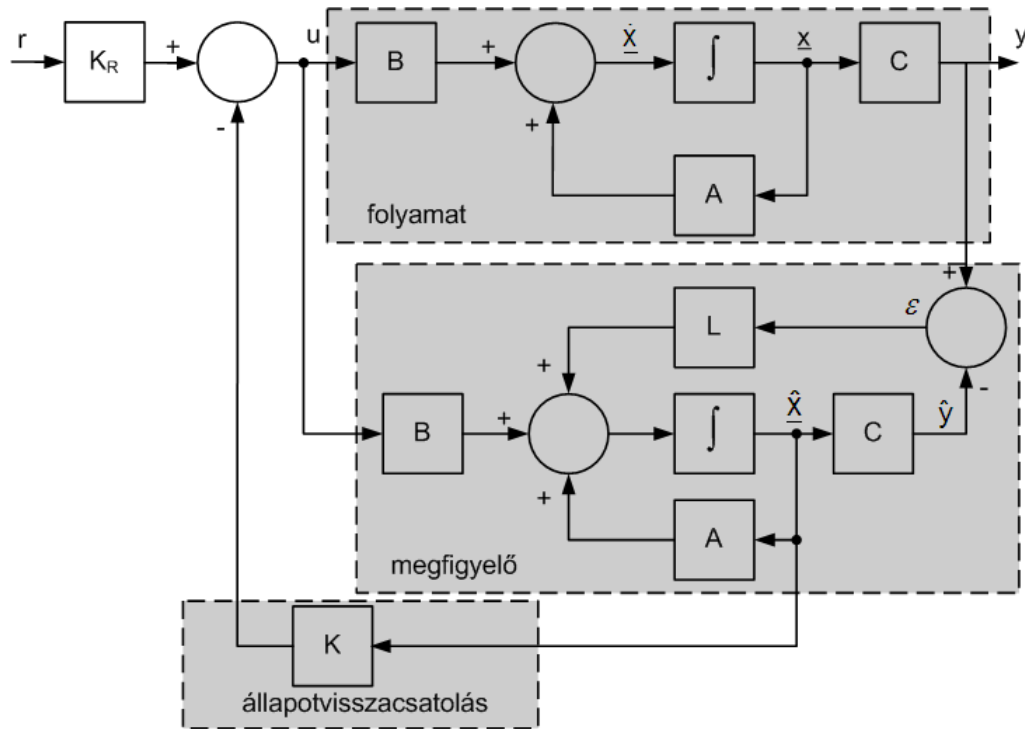


4.7. ábra. Nemlineáris modell LQ szabályozóval

Jól látható a (4.7) ábrán, hogy a szabályozó megfelelően működik, igaz, több idő szükséges a megfelelő állapotba való eljutáshoz. Ez azzal indokolható, hogy inkább olyan szabályozót hoztam létre, amely nem túl nagy bemenettel szabályozza a rendszert, hogy ne jöjjenek létre túl nagy lengések.

4.3. Megfigyelő tervezés

Az állapotvisszacsatolás módszeréhez szükség van a folyamatot leíró állapotegyenlet állapotváltozóira, amelyeket közvetlenül mérni kell. Sajnos ez az eset csak nagyon ritkán teljesül, általában csak alacsony fokszámú dinamika esetén vagy csak nagyon drága módon. Tehát az állapotvisszacsatolás módszere attól függ, hogy lehet-e mérést vagy becslést végezni az állapotváltozókról. Erre dolgozták ki a megfigyelő elvét, amely előállítja az állapotvektort. Ehhez szükség van a rendszermátrixok ismeretére ($A \ B \ C$). A módszer lényege, hogy ezekkel a mátrixokkal előállítjuk a folyamat pontos mását és ugyanolyan gerjesztéseknek kell kitenni. Így az eredeti modell és a megfigyelő előállítja az x , y és az $\hat{x}$, $\hat{y}$ változók becsült értékét. Az állapotvisszacsatolás pedig az $\hat{x}$ felhasználásával valósul meg.[8]



4.8. ábra. Megfigyelő alapú állapotviszacsatolás

Az (4.8) ábrán is látható, hogy az állapotviszacsatolás most az

$$u = K_R - K^T \hat{x} \quad (4.8)$$

formában jelenik meg, tehát egyszerűen x helyébe $\hat{x}$ lépett. Továbbá megvizsgálva a (4.8) ábrát látható, hogy az eredeti modell és a megfigyelő kimeneteinek a különbségéből egy hibajelét képzünk:

$$\varepsilon = y - \hat{y} \quad (4.9)$$

Ezt a hibajelét egy megfelelő L arányos visszacsatoló vektor a megfigyelő integrátorának a bemenetére csatolja vissza. Ezzel azt érjük el, hogy a hibajel csupán addig áll fenn, amíg a folyamat és a megfigyelő kimenete nem egyezik meg. Fontos megjegyezni, hogy a zárt rendszer követési tulajdonsága nem függ az L vektor megválasztásától, mivel a megfigyelő nem irányítható része a teljes zárt rendszernek. A megfigyelő működésének vizsgálatához képezzük az állapothiba

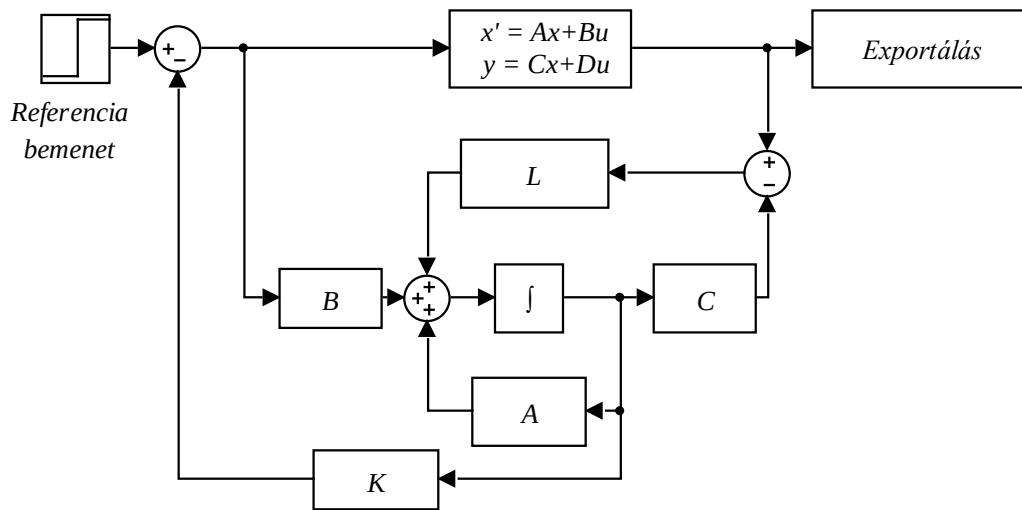
$$\tilde{x} = x - \hat{x} \quad (4.10)$$

A megfigyelők tervezésére az állapotvisszacsatoláshoz nagyon hasonló módszert használnak, ahol az L megválasztásakor az a cél, hogy a (4.10) rendszer dinamikáját a

$$\det(sI - A + LC^T) = F(s) = s^n + f_1s^{n-1} + \dots + f_{n-1}s + f_n \quad (4.11)$$

karakterisztikus polinom biztosítsa. A megoldás mindig létezik, ha a folyamat irányítható. [8] Látható, hogy az F mátrix sajátértékei fogják meghatározni, hogy milyen gyorsan csengjenek le a megfigyelés - tranziensei, azaz hogy milyen pontos legyen a megfigyelésünk.

Matlab Simulink környezetben összehuzaloztam a lineáris modellre a megfigyelő alapú állapotvisszacsatolást, és különböző szimulációkat végeztem el rajta.



4.9. ábra. Lineáris modell megfigyelővel

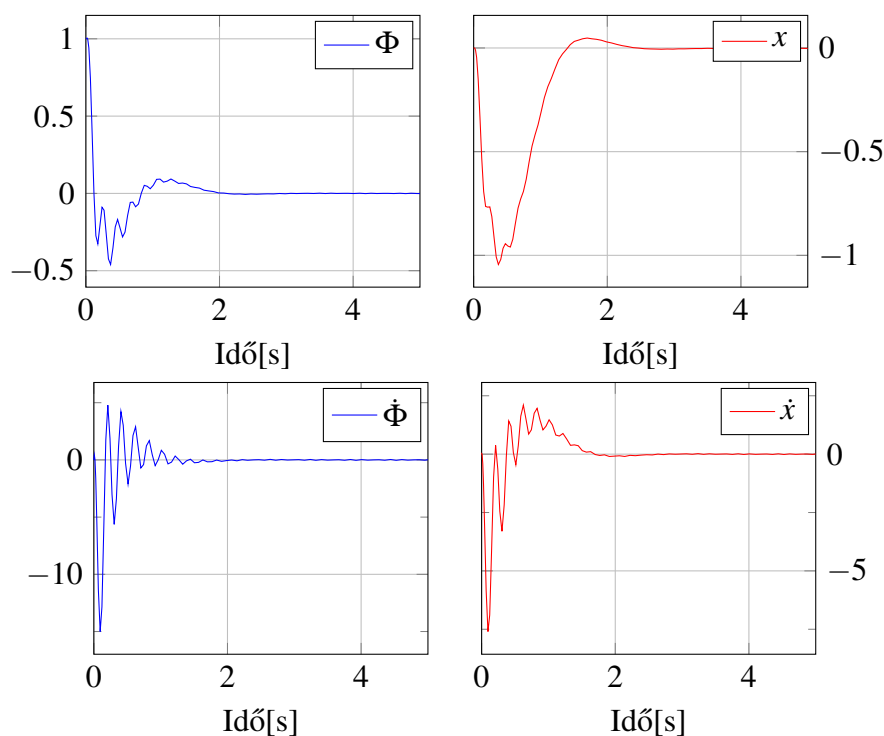
A szimulációk során az inga kezdeti állása 1 radián volt. Az (4.10) ábrán jól látszik, hogy a szabályozó megfelelően működik és hamar beállt a megfelelő egyensúly állapotba az inga. A megfigyelő helyesen működik, amelyet a (4.11) szimuláció mutat be jól. A megfigyelési hiba nem nő, hanem lecseng.

Az általam megalkotott megfigyelő esetén az állapotvisszacsatolásért felelős K vektornak az alábbi pólusokat választottam:

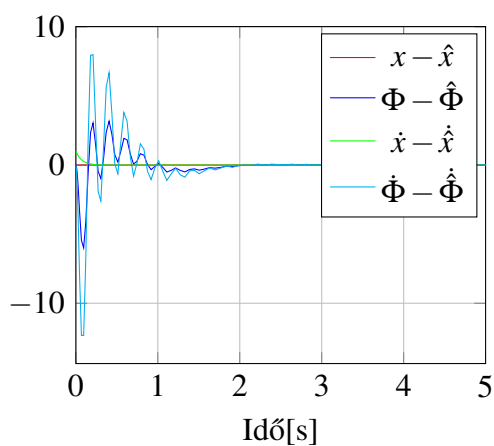
$$[-5, -6, -10, -11]$$

míg az L arányos visszacsatoló vektornak ennél gyorsabb pólusokat választottam, hogy a hibajel minél előbb megszűnjön. Az így választott pólusok:

$$[-80 - 90 - 140 - 150]$$



4.10. ábra. Lineáris modell megfigyelővel

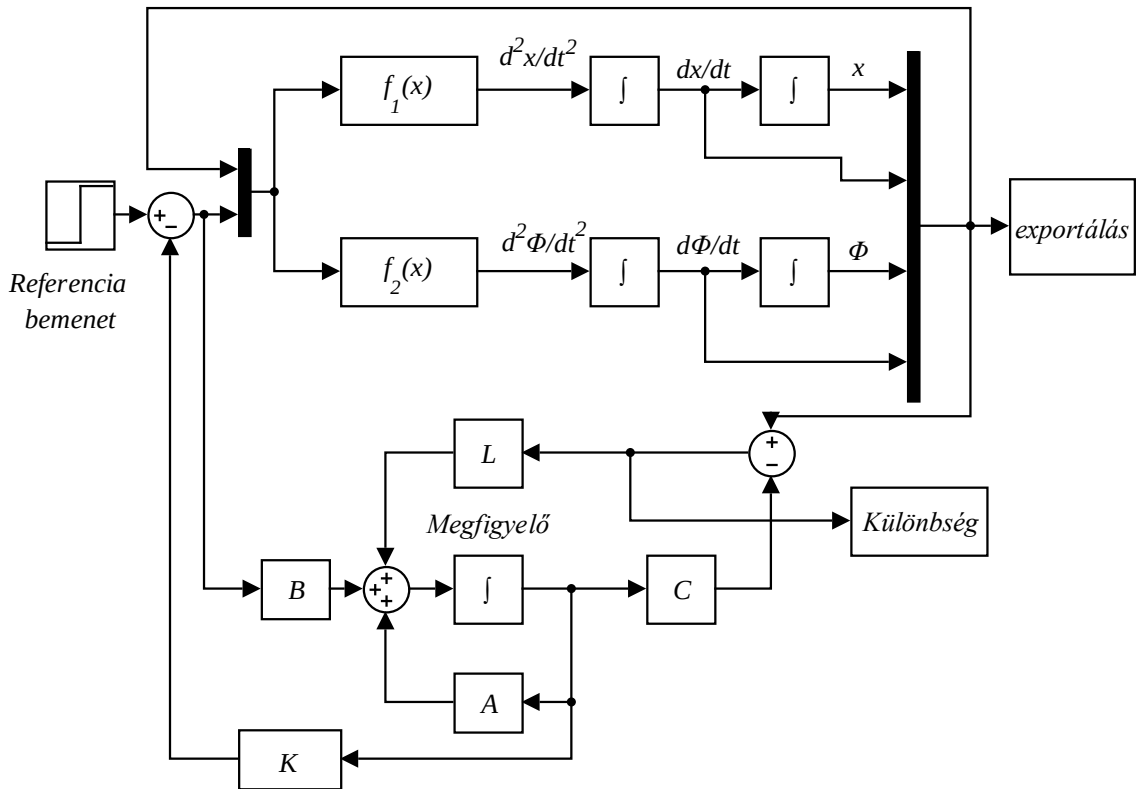


4.11. ábra. A megfigyelési hiba alakulása

Matlab Simulink környezetben továbbá összehuzaloztam a nemlineáris modellre a megfigyelő alapú állapotviszacsatolást, amelyet a lineáris modell alapján terveztem meg.

A szimulációk során megvizsgáltam az állapotváltozók értékeit és a ε jelet. Megfelelő működésű szabályozó esetén a hibajelnek csökkennie kell.

A (4.14) ábrán pedig a hibajel látható, amely csökken.



4.12. ábra. Nemlineáris modell megfigyelővel

4.4. Alapjel követő szabályozás

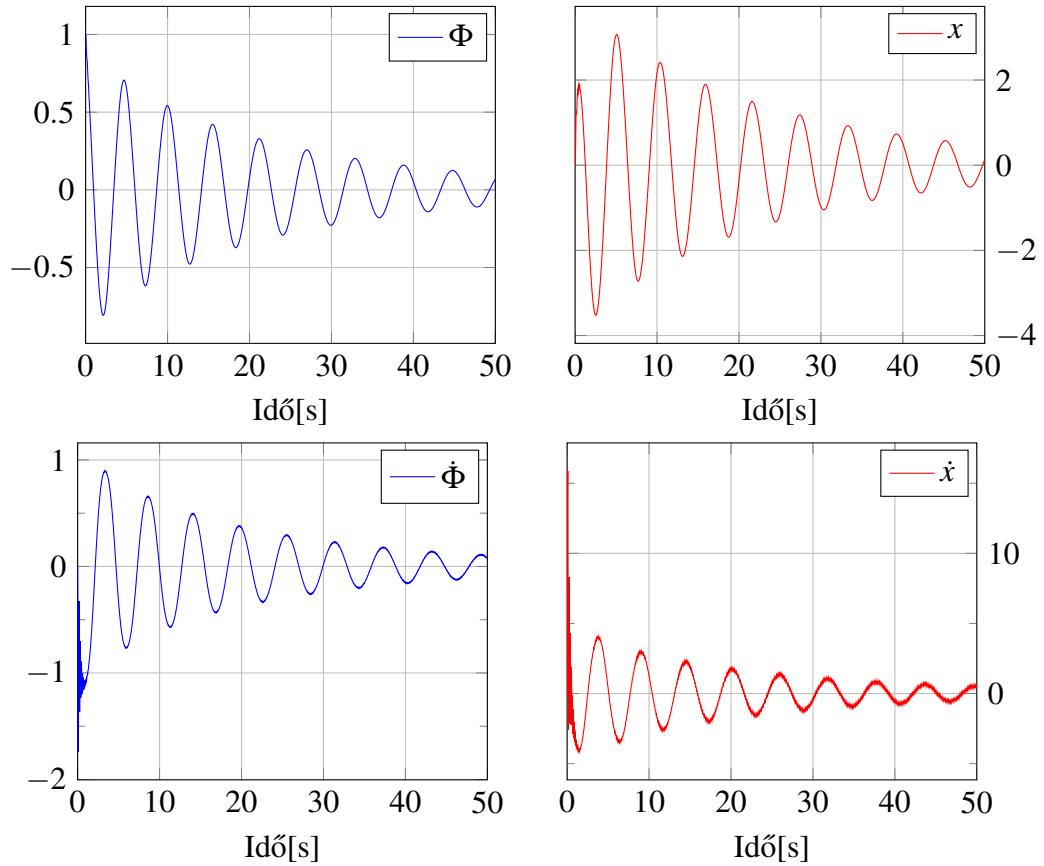
A kétlépcsős tervezési módszer másik elnevezése a szervó szabályozó. Ennek a lényege, hogy egy külső jelet akarunk követni. Az inverz inga esetében például megmondok egy pozíciót, hány métert tegyen meg előre, de fontos, hogy a mozgása közben megtartsa az egyensúlyt.

Az állapotvisszacsatoláson alapuló szabályozónak előnyei közé tartozik, hogy a módszer alkalmazhatósága nem függ attól, hogy a folyamat stabilis vagy labilis.

A nem kedvező tulajdonságai közé azt lehet felsorolni, hogy a zavarelhárítási tulajdonság nem tervezhető közvetlenül és az állapotvisszacsatolás a folyamat zérusait nem tudja megváltoztatni. A kedvezőtlen tulajdonságok miatt egy újabb szabályozási lépcső kerül alkalmazásba, amely egy kaszkád integráló szabályozó lesz, amelynek célja a zavarelnyomás és a paraméterbizonytalanságok kiküszöbölése. Ezt úgy érjük el, hogy új állapotként felvesszük a kimenet integrálját.

A (4.15) ábráról leolvashatjuk, hogy az új állapotváltozó értéke:

$$\delta(t) = \int_0^t r(\tau) - y(\tau) d\tau \quad (4.12)$$



4.13. ábra. Nemlineáris modell állapotváltozói megfigyelő esetén

A (4.12) képlet megadja a hibajel integrálját, ez lesz az új állapotváltozó. Tovább folytatja a vizsgálatot fel lehet írni a zárt rendszer új állapotegyenletét, amely a következőképpen néz ki:

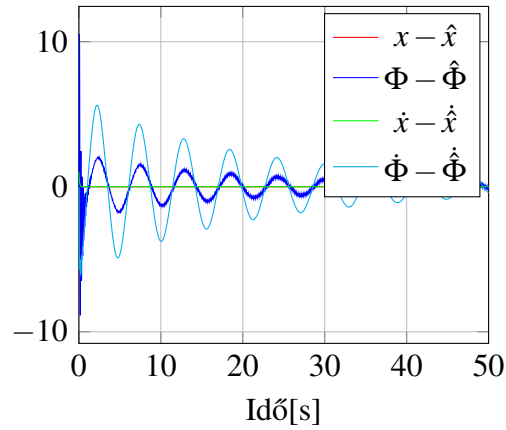
$$\dot{x}^*(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{\delta}(t) \end{bmatrix} = \begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ \delta(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ -1 \end{bmatrix} r(t) \quad (4.13)$$

A (4.13) egyenlet egyszerűsítéséhez az alábbi egyenleteket írjuk fel:

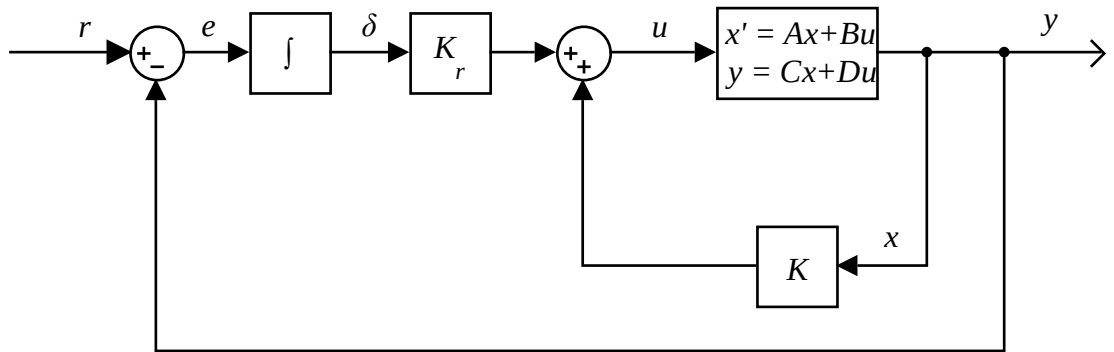
$$A^* = \begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix} \quad B^* = \begin{bmatrix} B \\ 0 \end{bmatrix} \quad x^* = \begin{bmatrix} x(t) \\ \delta(t) \end{bmatrix} \quad v^* = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \quad (4.14)$$

A (4.14) kifejezéseket felhasználva már át tudjuk írni a (4.13) egyenletet a következők szerint:

$$\dot{x}^* = (A^* - B^*K^*)x^*(t) + v^*r(t) \quad (4.15)$$



4.14. ábra. A megfigyelési hiba alakulása

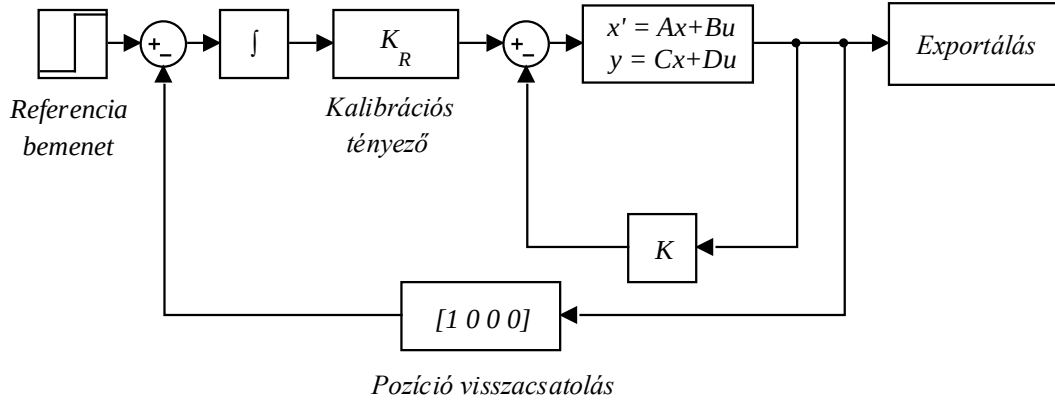


4.15. ábra. Állapotviszacsatolás és integráló szabályozó együttes alkalmazása

A bemenet pedig így fog alakulni:

$$u(t) = - \begin{bmatrix} K & K_R \end{bmatrix} \begin{bmatrix} x(t) \\ \delta(t) \end{bmatrix} = -K^* x^*(t) = K_R \int_0^t e(\tau) d\tau - Kx(t) \quad (4.16)$$

Következő lépésben összeállítottam a lineáris modellre a kétlépcsős tervezési módszert állapotviszacsatolással. A Simulink modelljét az alábbi (4.16) ábra mutatja be:



4.16. ábra. Lineáris modell szervó szabályozóval

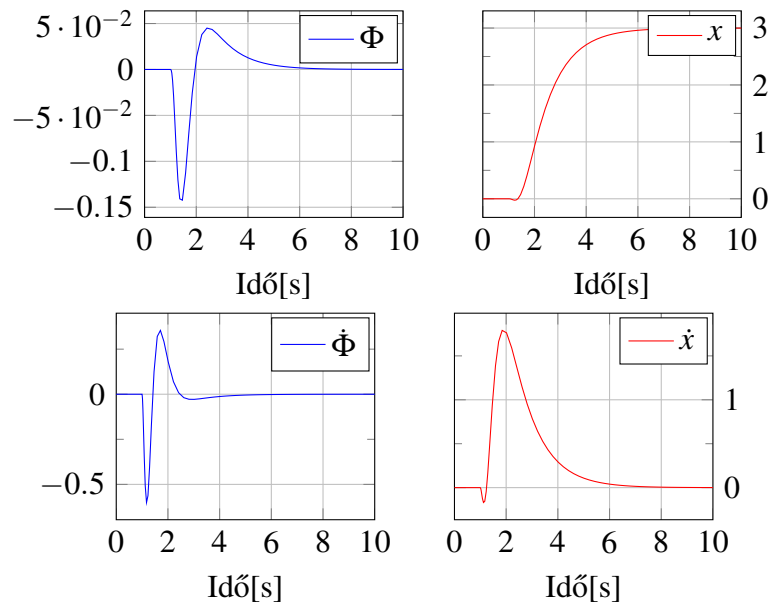
A (3.2) lineáris állapotter modellemet kiegészítettem a hibajel integráljával a (4.13) kifejezés alapján és az alábbi eredményt kaptam:

$$\begin{bmatrix} \ddot{x}^* \\ \ddot{\phi}^* \\ \ddot{\delta}^* \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & \frac{(2KK_i(Lm_v r - L^2 m_v - J_v))}{\beta} & \frac{(gL^2 m_v^2 r^2 R)}{\beta} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \frac{(\frac{2KK_i}{rR}(\frac{Lm_v}{r} - \frac{2I_k}{r^2} - 2m_k - m_v))}{\alpha} & \frac{(gLm_v(-2m_k - m_v - \frac{2I_k}{r^2}))}{\alpha} & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \\ \delta \end{bmatrix} +$$

$$+ \begin{bmatrix} 0 \\ \frac{(2Kr(J_v + L^2 m_v - Lm_v r))}{\beta} \\ 0 \\ \frac{(\frac{2K}{R}(2m_k + m_v + \frac{2I_k}{r^2} - \frac{Lm_v}{r}))}{\alpha} \\ 0 \end{bmatrix} [U_k]$$

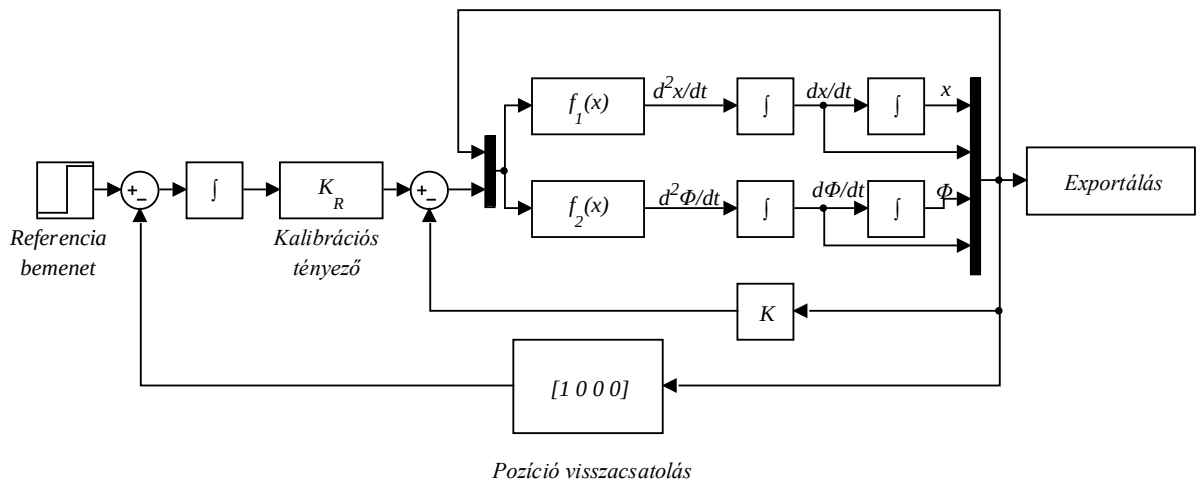
(4.17)

A szimuláció során a követendő jelet egy ugrásfüggvény jelentette, amely 3-ra állt be végül. Így a pozíciónak 3 köré kellett beállnia.



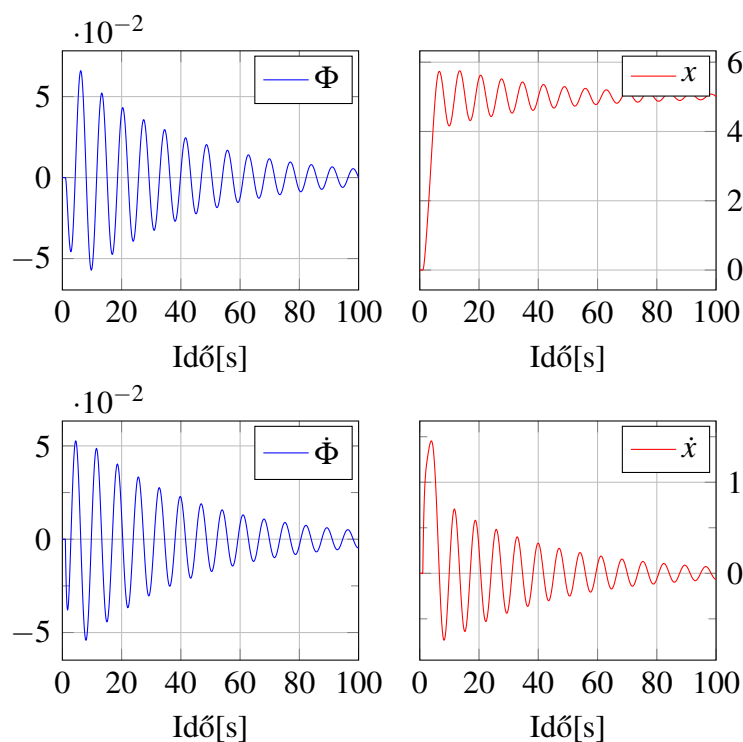
4.17. ábra. Lineáris modell szervó szabályozóval

A nemlineáris modellt összehuzaloztam Simulinkben, amelyet a (4.18) ábra mutat be részletesen:

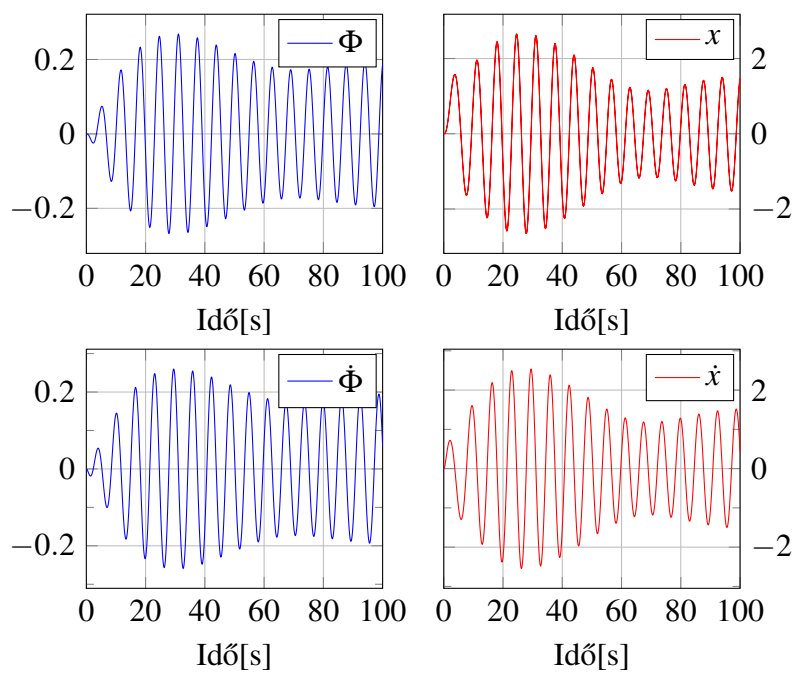


4.18. ábra. Nemlineáris modell szervó szabályozóval

Ezzel a modellel több szimulációt is futtattam különböző bemenetekkel. Elsőként egy egységugrást kell követnie a modellnek, majd egy szinuszjelet.



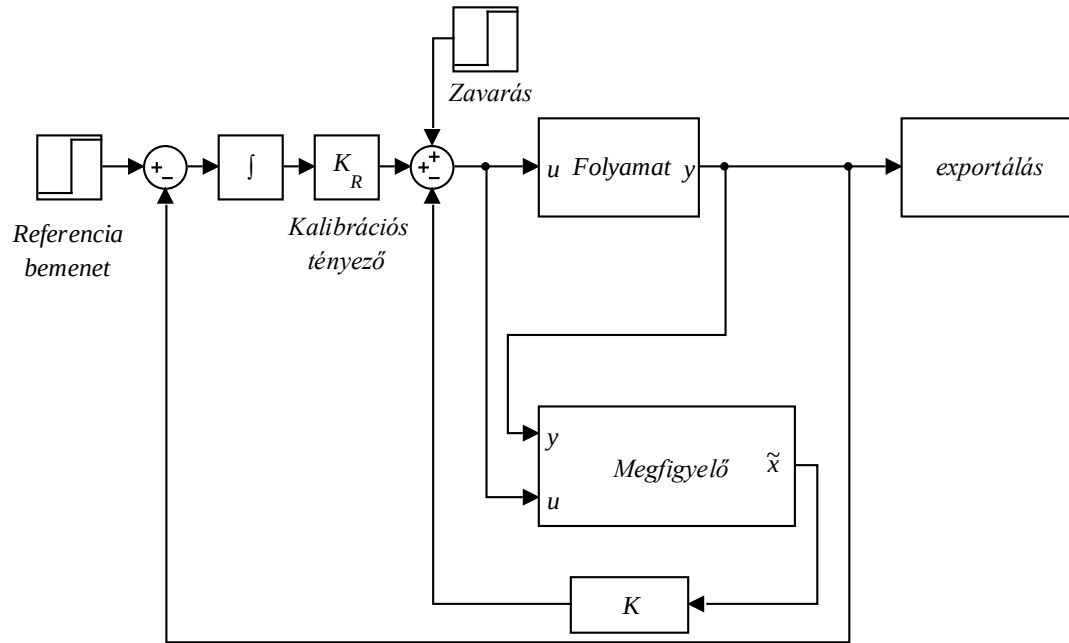
4.19. ábra. Nemlineáris modell szervó szabályozóval egység ugrás bemenettel



4.20. ábra. Nemlineáris modell szervó szabályozóval szinusz bemenettel

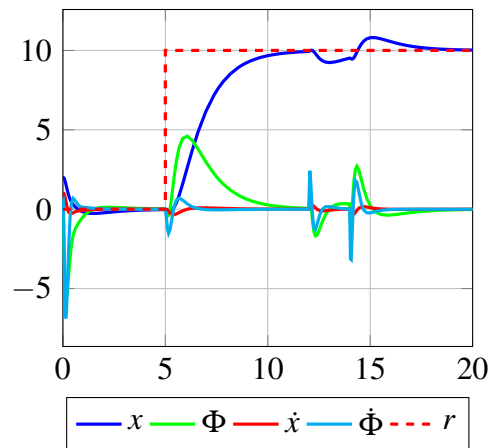
4.5. Megfigyelő és szervó szabályozó együttes használata zavarással

A következőkben együttesen vizsgáltam a megfigyelő és a szervó szabályozó működését különböző zavarások esetén, így kapva egy képet arról, hogy hogyan viselkednek a megtervezett szabályozók a zavarelnyomás folyamán.



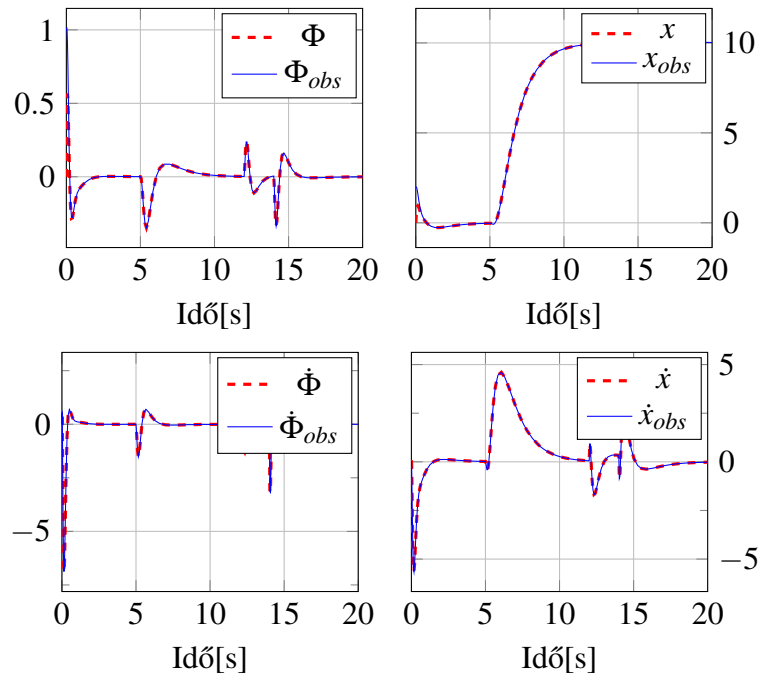
4.21. ábra. Lineáris modell szervó szabályozóval és megfigyelővel

A szimulációk során a kezdeti értékem az alábbiak voltak: $[2, 0, 1, 0]$, vagyis a kezdeti szögérték 1 radián volt, míg a kezdeti pozíció 2 méter volt. A referencia jelem pedig öt másodperc után 10 méterre ugrott, amelyet követnie kellett a rendszernek.



4.22. ábra. Állapotok alakulása a referencia mellett

Az (4.23) ábrán pedig a megfigyelő működése látható:



4.23. ábra. Lineáris modell szervó szabályozóval és megfigyelővel

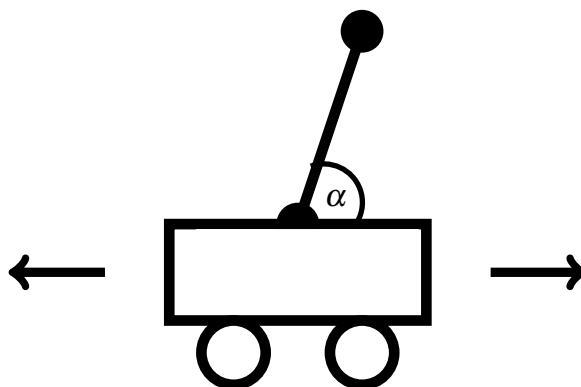
Az (4.22) összehuzalozott modellemben tettem bele zavarást is, amely a 12. és a 14. másodpercben lépnek be a rendszerbe. A (4.23) ábrán is látszik, hogy valamiféle zavarás megjelent, de a szabályozó ezt sikeresen elnyomta.

4.6. Fuzzy szabályozó

4.6.1. Fuzzy

A fuzzy logika és halmazok alapötlete 1965-ből, a kaliforniai Berkeley-n tanító Zadeh-től származik. [6] A fuzzy logika lényege, hogy a nehezen mérhető vagy a bizonytalan információkat matematikai formában lehessen kezelni. Ezzel a logikával olyan folyamatokat lehet egyszerűbben szabályozni, amelyeket a klasszikus szabályozási technikákkal nehezen vagy akár egyáltalán nem lehetne kezelni. A fuzzy halmazok alkalmazása lehetővé teszi a mindennapi életben megszokott, korábban igen nehezen kezelhető nyelvi fogalmak (mint pl. gyors, lassú, szép, fiatal) matematikai leírását.[7]

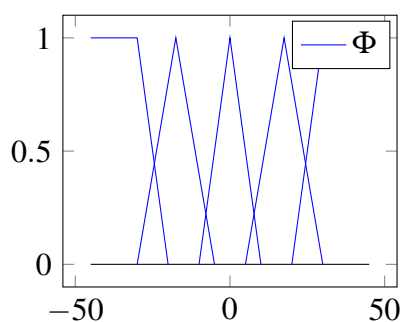
Első lépésben a fellépő mozgásokat kell definiálni (ez a lépés a fuzzifikáció). Az inverz inga vagy előre vagy hátra mozog. Ezt a mozgást még finomítani kell, például előre lassú vagy gyors esettel.



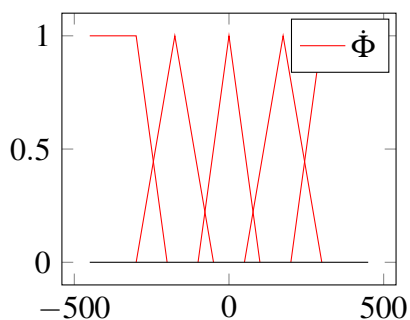
4.24. ábra. Inverz inga leegyszerűsítve

A bemenő jelek a szög és a szögsebesség lesz. Ezeket szintén felosztottam két esetre. A szögnél külön nézem a nagy és a kis szögelfordulást, míg a szögsebességnél a gyorsat és a lassú változást figyelem külön.

Első lépésben felállítottam a halmazokat Matlab környezetben:



4.25. ábra. Szög



4.26. ábra. Szögsebesség

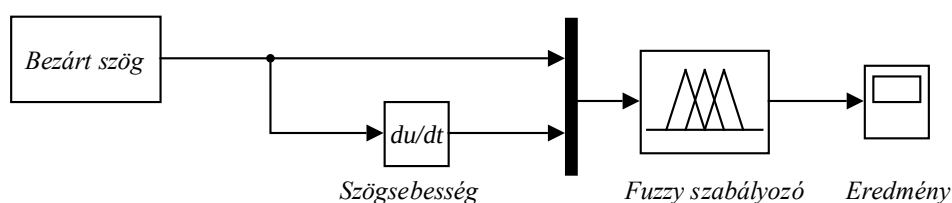
Ezek után a halmazok közötti szabályokat kell felállítani. Ilyen szabály például:

- **Ha** a szög "előre kicsi" és a szögsebesség "előre lassú" **akkor** a sebesség legyen "előre kicsi"

Összesen 11 darab szabályt alkottam, amelyeket táblázatos formában foglalok össze: A szabályokat ideális esetre állítottam össze, vagyis Simulinkben összehuza-

$\dot{x}$		Bezárt szög				
		Hátra nagy	Hátra kicsi	Helyben	Előre kicsi	Előre nagy
Φ	Hátra gyors	Hátra nagy	Hátra nagy			
	Hátra lassú	Hátra nagy	Hátra kicsi	Hátra kicsi		
	Helyben			Helyben		
	Előre lassú			Előre kicsi	Előre kicsi	Előre nagy
	Előre gyors				Előre nagy	Előre nagy

lozott modell így néz ki:



4.27. ábra. Fuzzy simulinkben

4.6.2. Fuzzy szervó szabályozóval

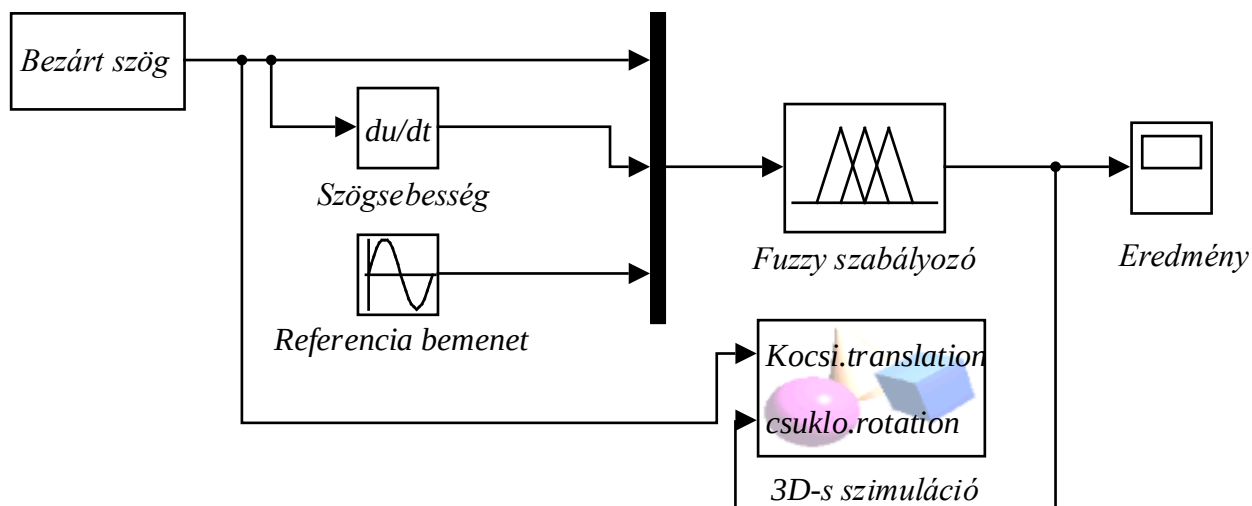
Ebben a fejezetben az általam létrehozott fuzzy szabályozót egészítettem ki szervó szabályozással, így képes lesz már követni egy referencia jelet. Ehhez új szabályokat kellett bevezetnem.

Mivel egy referencia jel követése a cél, lesz egy új bemeneti változó is, amelynek a hiba nevet adtam. Minél nagyobb a hiba, annál nagyobb az eltérés a referencia jeltől, és ennek függvényében kell az ingának előre vagy hátra mozognia. Eddig összesen 11 szabályt definiáltam, amelyet most kiegészítettem további öttel:

- **Ha** a hiba "apró", **akkor** a sebesség "helyben"
- **Ha** a hiba "hátranagy", **akkor** a sebesség "előrekicsi"
- **Ha** a hiba "hátraóriás", **akkor** a sebesség "előrenagy"

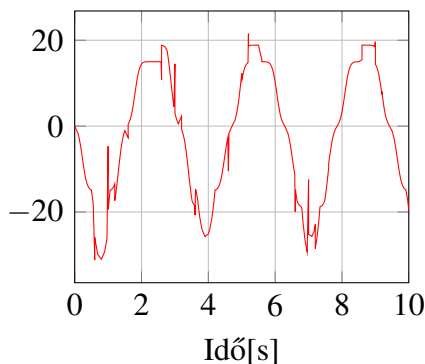
- **Ha** a hiba "előrenagy", **akkor** a sebesség "hátrakicsi"
- **Ha** a hiba "előreóriás", **akkor** a sebesség "hátranagy"

Ezek után összehuzaloztam a Simulinkben a fuzzy szabályozómat.



4.28. ábra. Fuzzy szervó szabályozóval

Az általam választott referencia jel most egy szinusz volt, amelyet a stabilizálás mellett jól követett a rendszer ((4.29) ábra).

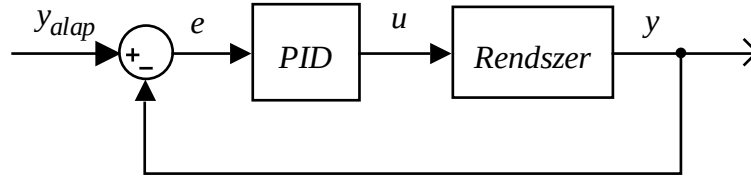


4.29. ábra. Szinuszos referencia bemenet követése

4.7. PID szabályozó

A sok szabályozási algoritmus mellett rendkívül elterjedt szabályozási módszer az egyszerűsége és jó teljesítmőképessége miatt a PID szabályozó. A PID módszer lényege, hogy a szabályozott folyamat kimeneti jelét (y) összehasonlítja egy alapjellel

(y_{alap}). Ebből a két jelből különbségképzés után létrehoz egy hibajelet (e), amely e jel függvényében előállítja az új beavatkozó jelet (u). A cél tehát az, hogy minél gyorsabban érjük az alapjelet. PID szabályozó esetén a beavatkozó jelet az alábbi



4.30. ábra. PID általános felépítése

egyenlet adja meg:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \quad (4.18)$$

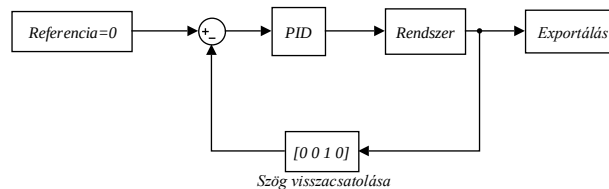
ahol K_p jelenti az erősítést, a K_i az integráló tagot, míg K_d a deriváló tagot.

4.7.1. Megvalósított szabályozó

Matlab-ban összeállítottam egy PID szabályozót és a megfelelő tagokat a Simulink csomagban található beépített program hangolta be az alábbiak szerint:

$$\begin{aligned} P &= 209 \\ I &= 0.08 \\ D &= 65215 \end{aligned} \quad (4.19)$$

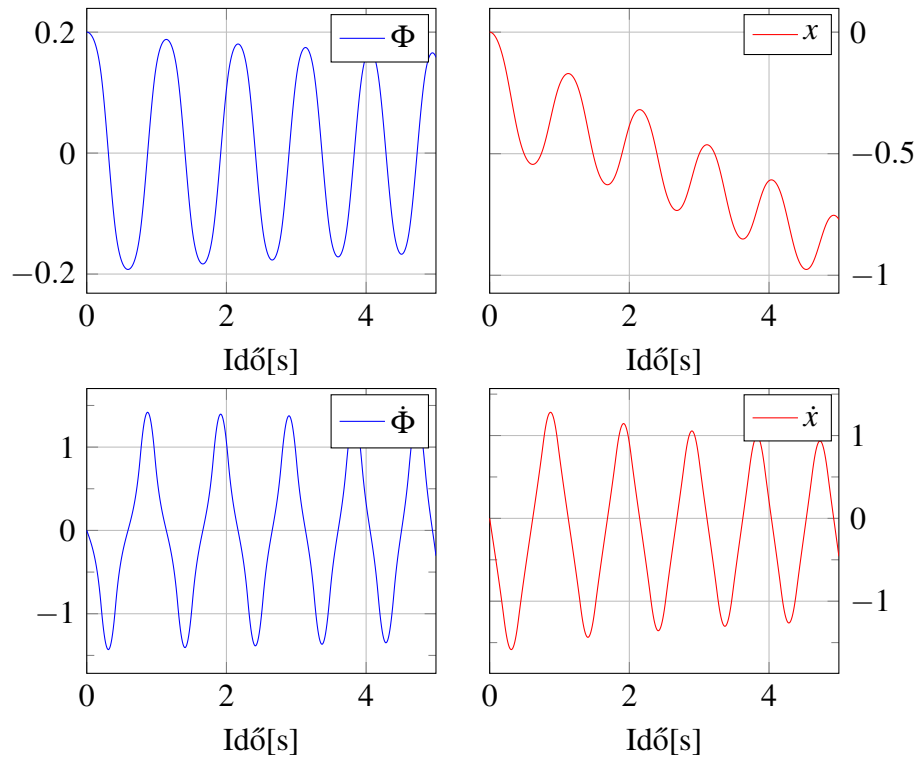
Íme az összeállított modell:



4.31. ábra. PID szabályozó megvalósítása

A következő ábrán a szabályozó működésének eredménye látható. A kezdeti szög érték 0.2 radián ($\approx 12^\circ$) volt.

Az előző ábrán (4.32) látható, hogy az inga nem dől el. A gyenge szabályozást



4.32. ábra. PID szabályozó működése 0.2 radián kezdeti szög esetén

valószínűleg az a tény indokolja, hogy a szabályozott rendszer instabil. A PID szabályozó alapvetően stabil rendszerek szabályozására alkalmas, valószínűleg ez magyarázza a jelenséget.

4.8. Szabályozók összehasonlítása

Ebben a részben az általam megvalósított szabályozókat hasonlítom össze stabilizálás és ahol volt, ott alapjel követés szerint.

Első esetben LQ és pólusáthelyező szabályozókat hasonlítom össze azonos kezdeti értékek esetén. Először a lineáris modellre tervezett szabályozókat vizsgálom. Vizsgálataimhoz a (4.3) és (4.6) ábrákat használtam fel. A pólusáthelyező szabályozó eredményét összehasonlítva az LQ szabályozó eredményével azt kapjuk, hogy az LQ szabályozó sokkal gyorsabban stabilizálta a szög értékét. Ehhez azonban sokkal nagyobb utat tett meg a jármű, amely nem minden esetben lehet hasznos.

A következő lépésben pedig a PID szabályozót hasonlítom össze az LQ szabályozóval. A (4.32) ábrán jól lehet látni, hogy a periodikus lengés csak lassan csökken, vagyis sokkal tovább tart, mire eljut az egyensúlyi pontba az inga. A fentiek alapján kijelenthetem, hogy amennyiben csupán az inga stabilizálása a cél az LQ szabályozó a megfelelő választás.

Alapjel követése esetén pedig a kétlépcsős tervezési módszert hasonlítom össze a Fuzzy szervó szabályozóval, amelyek a stabilizálás mellett alapjelet is követnek. Mindkét szabályozót szimuláltam konstans jel követésével és az eredmények azt mutatták, hogy a kétlépcsős módszer jobban követi az alapjelet, mint Fuzzy logika. Természetesen a Fuzzy logikával még további munkálataim vannak hátra az összeállított hardver segítségével, amellyel tovább tudom hangolni a Fuzzy logika változóit, így javítva működését. A dolgozatom alapján a kétlépcsős tervezési módszert ajánlanám alapjel követésre, mivel már a hardver megléte nélkül is jól követi a referencia jelet zavarások ellenére is (4.22 ábra).

5. Implementáció és tesztelés

5.1. Rendszerterv

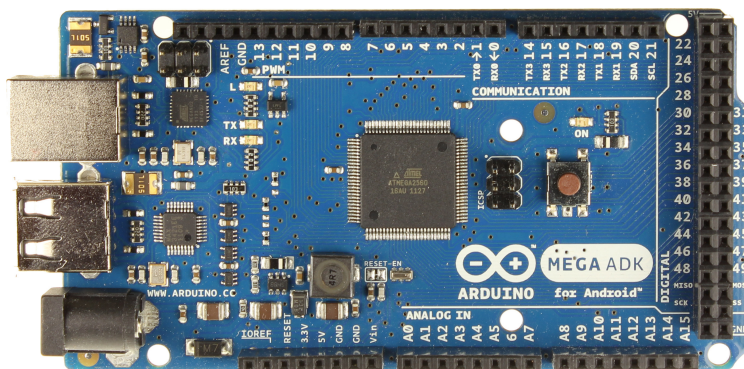
A kitűzött cél, hogy egy kerekeken guruló inverz ingát építsék, ameylen a korábban megtervezett szabályozó algoritmusokat könnyen tudom implementálni. A Matlab SIMulink környezet képes arra, hogy egy modellből automatikus kódot generáljon az Arduino mikrovezérlőcsalád tagjaira. Ez egy nagyon komoly előny, hiszen a folytonos modellek diszkrétizálása így rábízható a Matlab-ra, ami nem kis feladatot jelentene a számomra.

A használt motormodell egy egyenáramú motoré, ez leszűkíti a használandó motorok körét. Méréstechnikai szempontból előnyt jelent, ha a motor enkóderrel ellátott, azaz mérhető a szög/szögsebessége. Az Arduino kártya közvetlenül nem képes a motorok meghajtására, ezért egy megfelelő motorvezérlőre van szükség. Az inga állapotváltozóinak méretősége miatt gyorsulásmérő és giroszkóp alkalmazása is szükséges.

5.2. Választott eszközök

5.2.1. Arduino

Az inverz ingám központi feldolgozó egysége egy Arduino MEGA ADK.



5.1. ábra. Arduino MEGA ADK

Az Arduino MEGA ADK egy ATmega2560-as alaplapú mikrokontrollerek kártya, amely az ATmega2560-as panel tudásán felül képes USB interfészen keresztül csatlakozni Android alapú telefonokhoz, amely elengedhetetlen lesz ahhoz, hogy a későbbi továbbfejlesztések során okostelefonnal lehessen irányítani a robotot.

Néhány fontosabb adat: Részletes információ a kártya katalóguslapján található,

Mikrokontroller	ATmega2560
Üzemi feszültség	5V
Ajánlott bemeneti feszültség	7-12V
Digitális I/O lábak	54 (ebből 15 támogatja a PWM kimenetet)
Analóg bemeneti lábak	16
Órajel	16 MHz

a B.1.3 függelékben.

5.2.2. Motor

Az inga egyensúlyban tartásához két darab beavatkozóra van szükségem, amelyeket a Pololu cég által gyártott két darab egyenáramú motorok valósítanak meg. Íme egy ábra a motorról:



5.2. ábra. 12V-os DC motor

A motor rendelkezik egy beépített enkóderrel (Hall-elem), amely segítségével a motor elfordulása mérhető. Részletes információ a motor katalóguslapján található, a B.1.1 függelékben.

Néhány fontosabb adat:

Áttétel	29:1
Szabad futási sebesség 12V-on	350 RPM
Indító áram 12V-on	5000 mA
Maximális nyomaték	0.77 Nm

5.2.3. GY-80 kombinált szenzor

A GY-80 kombinált 10 tengelyű modul (3 irányú gyorsulásmérő, 3 tengelyű giroszkóp, 3 irányú mágnes térerőmérő és 1 nyomásmérő) felépítése:

- L3G4200D (gyorsulásmérő)
- ADXL345 (giroszkóp)

- HMC5883L (mágneses térerőmérő)
- BMP085 (nyomásmérő)

Az inverz inga szabályozásához nekem csak az első két szenzorra van szükségem.

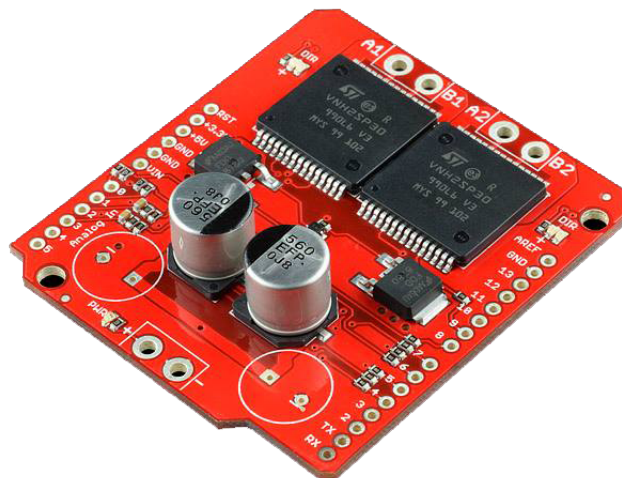


5.3. ábra. GY-80 szenzor

A dedikált I2C buszra közvetlenül, egyetlen címen illeszthető a modul, így lehetőséget biztosít, hogy több érzékelő is közös áramkörön keresztül megszólítható legyen. Üzemi tápfeszültsége 3.3V. Részletes információ a szenzor katalóguslapján található, a B.1.6 függelékben.

5.2.4. Motor vezérlő

A két darab nagy nyomatékú motorom vezérlésére egy "dual VN2SP30" típusú motor vezérlő modult használtam, amely két darab motort képes egyidejűleg vezérelni.



5.4. ábra. VN2SP30 motorvezérlő

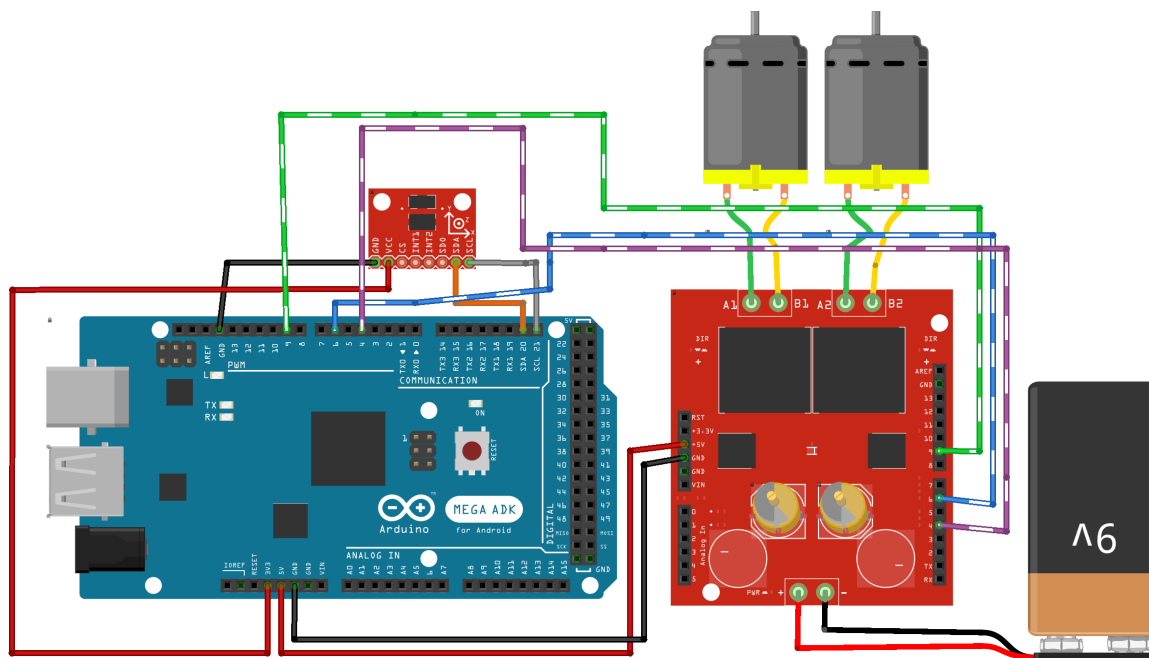
Néhány fontosabb adat: Részletes információ az eszköz katalóguslapján található,

Üzemi tápfeszültség	5.5 – 16 V
Maximális áram terhelhetőség	30 A
Maximális PWM frekvencia	20 kHz
MOSFET ellenállás a lábakon	19mΩ

a B.1.4 függelékben.

5.3. Hardver implementáció

Az első kapcsolási rajz az alapvető összeszerelést mutatja be, amely alapján lehet már mozgatni az inverz ingát. A (5.5) ábrán 9V-os elem látható, mivel a szerkesztő

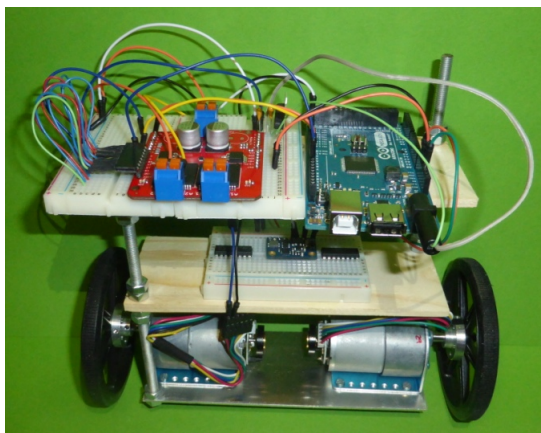


5.5. ábra. Első kapcsolási rajz

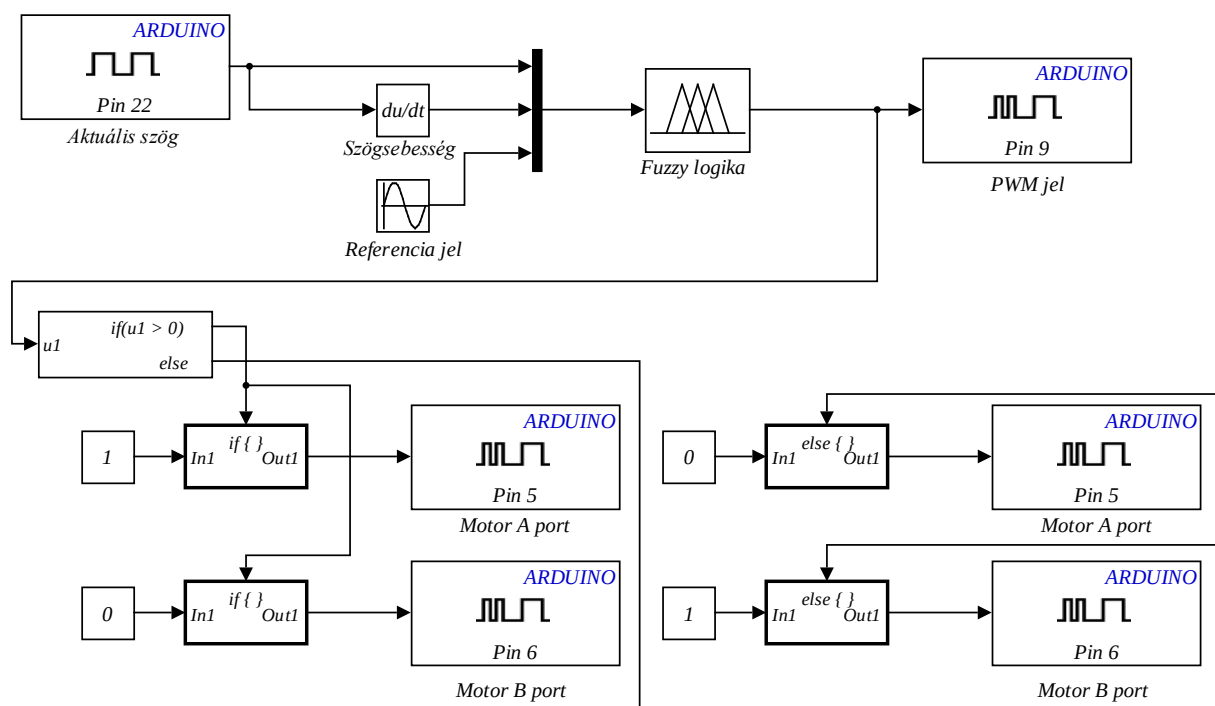
programban nem tudtam 12V-os tápegységet választani. Ebből kitalálható, hogy az ingám akkumulátor helyett még tápegységről járatom. Az Arduino számára az 5V-ot egy LF50CV típusú feszültségszabályozó IC biztosítja.

5.4. Szoftver implementáció

A szoftveres implementáció egyelőre még várat magára, az alábbi ábrán a fuzzy alapú szervó szabályozó Simulink modellje látható, amelyből az Arduino kód automatikusan generálódik.



5.6. ábra. Összerakott inverz inga



5.7. ábra. Arduino program Simulinkben

6. Összegzés

6.1. Konklúzió

A kitűzött feladatom egy guruló inverz inga szabályozása volt, mind elméleti, mind pedig gyakorlati szempontból. Jelen munkában elsősorban az elméleti munkával foglalkoztam, a párhuzamosan készülő szakdolgozatomban a hardveres implementáció egy kicsivel nagyobb hangsúllyal szerepel.

A dolgozatomban először az egyenáramú motor modelljét készítettem el. Az elkészült modellt utána egy általam választott valós motor paramétereivel szimuláltam. A szimulálás során látszott, hogy a modell jól lett összeállítva, mivel a választott motor katalógus lapjának megfelelően viselkedett.

A következő lépésben elkészítettem az inga teljes modelljét fizikai alapokon, amely tartalmazza a motor modellt. A legtöbb időt ez vette igénybe, mivel teljesen a nulláról indultam a modell építéskor. Először megvizsgáltam a kerék-motor kapcsolatot, majd áttértem a váz modellezésére. Miután azzal végeztem, már feltudtam írni a nemlineáris egyenletet, amelyet lokális linearizálás útján linearizáltam.

Ezek után a létrehozott modellre különféle szabályozókat terveztem, többek között pólusáthelyezőt, LQ, Fuzzy, PID és szervó szabályozót. Mindegyik szabályozót összeállítottam Matlab Simulink környezetben és különféle szimulációkat futtatam, amelyeket később ki is értékeltem. A végén pedig még össze is hasonlítottam a szabályozókat egymással, amely alapján kiválasztottam, hogy melyiket fogom implementálni a hardveren.

6.2. További munka

A legközelebbi elérendő cél, hogy a hardveres változat is megfelelően működjön, ehhez már meglehetősen közel állok. A modell alapú módszerek alkalmazása úgy korrekt, ha a valós rendszert megfelelően illesztjük a modellhez, ehhez bemenet-kimenet méréseken alapuló paraméterbecslés szükséges, aminek az előkészületeit megtettem a paraméterérzékenység vizsgálattal.

A mai világban egyre jobban teret nyernek az android alapú programok, emiatt és mert érdekel is, egy okos telefon segítségével lehet majd távirányítani az inverz ingát. Emiatt választottam mikrokontrollernek az Arduino Mega ADK-t, amely segítségével később képes leszek megvalósítani a távirányítást. Az irányítás mellett majd a szabályozó algoritmust is a telefonon keresztül lehet változtatni.

Az inga egyenlőre csak egy dimenziós (x-tengely) irányban képes mozogni. Ezt

később szeretném két dimenziós (x,y-tengelyek) tartományba növelni. Amennyiben ez elkészül az inga már képes lesz fordulásra és összetettebb feladatok elvégzésére is. A megtervezett szervó szabályozót szeretném majd éles helyzetben is tesztelni, valóban működik-e.

Irodalomjegyzék

- [1] E. Vinodh Kumara, Jovitha Jeromeb. *Robust LQR Controller Design for Stabilizing and Trajectory Tracking of Inverted Pendulum*. Procedia Engineering, 64 (2003) 169-178
- [2] Hyungjik Lee, Seul Jung. *Balancing and navigation control of a mobile inverted pendulum robot using sensor fusion of low cost sensors*. Mechatronics 22 (2012) 95–105
- [3] Chih-Hui Chiu, Ya-Fu Peng, You-Wei Lin. *Intelligent backstepping control for wheeled inverted pendulum*. Expert Systems with Applications, 38(4), April 2011, 3364–3371
- [4] Stelian-Emilian Oltean. *Swing-up and Stabilization of the Rotational Inverted Pendulum Using PD and Fuzzy-PD Controllers*. Procedia Technology 12 (2014) 57–64
- [5] Rich Chi Ooi. *Balancing a Two-Wheeled Autonomous Robot*. The University of Western Australia School of Mechanical Engineering, 2003.
- [6] Kóczy T. László, Tikk Domonkos. *Fuzzy rendszerek*.
- [7] Kovács Szilveszter. *Fuzzy logic control*. M.Phil. theses, Technical University of Budapest, Faculty of Informatics and Electrical Engineering, Budapest, Branch of Computer Science, p.116, (1993).
- [8] Keviczky László, Bars Ruth, Hetthéssy Jenő, Barta András, Bányász Csilla. *Szabályozástechnika* Kiadja az UNIVERSITAS-GYŐR Kht. Felelős kiadó dr. Szily István.
- [9] Kádár Bence, Pethe Szabolcs
Egyensúlyozó robot építése és modellezése Debreceni Egyetem Informatikai Kar, 2011.
- [10] Hangos Katalin, Szederkényi Gábor *Dinamikus rendszerek paramétereinek becslése* Egyetemi Kiadó, Veszprém

A. Modellezés és modellanalízis

A.1. Jelölésjegyzék

Szimbólum	Mennyiség	Mérték-egység
M_{motor}	Motor nyomatéka	Nm
I_a	Armatúra áram	A
R	Motor soros ellenállása	Ω
K	Nyomatékállandó	Nm/A
U_i	Indukált feszültség	V
K_i	Back-EMF konstans	-
ω_m	A motor szögsebessége	rad/s
L	Az inga hossza	m
m_k	A kerék tömege	kg
r	A kerék sugara	m
θ	A kerék szögelfordulása	rad
$\dot{\theta}$	A kerék szögsebessége	rad/s
$\ddot{\theta}$	A kerék szöggyorsulása	rad/s ²
m_v	Az inga vázának a tömege	kg
U_k	Kapocsfeszültség	V
J_m	A motor tehetlenségi nyomatéka	kgm ²
$M_{terhelo}$	A motorra ható külső nyomaték	Nm
ϕ	Az inga dőlésszöge	rad

A.1. táblázat. Jelölések

A.2. Matlab/Simulink forráskód

A.2.1. Lineáris modell

```

g=9.81; %(m/s ^2)
r=0.0900 %(m)
mk=0.044; %(kg)
mv= 0.673; %(kg)
Jk=0.00044; %(kg*m^2)
Jv=0.00673; %(kg*m^2)
L=0.2; %(m)
K=0.006123; %Motor nyomatek konstans
Ki=0.0069203; %Back EMF konstans
R=3; %Motor ellenallas

```

```
alpha = (L^2*mv^2-(Jv+L^2*mv)*(2*mk+mv+(2*Jk)/r^2));
beta = (R*(2*Jk*Jv+2*Jk*L^2*mv+2*Jv*mk*r^2+Jv*mv*r^2+
+2*L^2*mk*mv*r^2));
A = [0 1 0 0;
      0 ((2*K*Ki*(L*mv*r-L^2*mv-Jv))/beta
      (g*L^2*mv^2*r^2*R)/beta 0;
      0 0 0 1;
      0 ((2*K*Ki)/(r*R))*((L*mv)/
      /r-(2*Jk)/r^2-2*mk-mv))/alpha
      (g*L*mv*(-2*mk-mv-
      -(2*Jk)/r^2))/alpha 0]

B = [0;
      (2*K*r*(Jv+L^2*mv-L*mv*r))/beta;
      0;
      (((2*K)/R)*(2*mk+mv+(2*Jk)/r^2-(L*mv)/r))/alpha]
C = [1 0 0 0;
      0 1 0 0;
      0 0 1 0;
      0 0 0 1]
D = [0;
      0;
      0;
      0]
K=place(A,B,[-5 -6 -10 -11])
sim('linearis_modell1_simulink',2);
%rajzolas
subplot(4,1,1);
plot(szog.time, szog.signals.values)
grid on;
xlabel('Ido[s]');
ylabel('Inga dolesszoge');
legend('Szog[rad]');
subplot(4,1,2);
plot(szog_derivalt.time, szog_derivalt.signals.values)
grid on;
xlabel('Ido[s]');
ylabel('Inga szogsebessége');
```

```
legend( 'Szog_gyorsulas[rad/s] ');  
subplot(4,1,3);  
plot(pozicio.time, pozicio.signals.values)  
grid on;  
xlabel( 'Ido[s] ');  
%ylabel( 'Kerek szogelfordulasa' );  
legend( 'Pozicio[m] ');  
subplot(4,1,4);  
plot(pozicio_derivalt.time, pozicio_derivalt.signals.values)  
grid on;  
xlabel( 'Ido[s] ');  
%ylabel( 'Motor szogsebessége' );  
legend( 'Gyorsulas[m/s] ');
```

A.2.2. Nemlineáris modell

```
g=9.81; %(m/s^2)  
r=0.0900 %(m)  
mk=0.044; %(kg)  
mv= 0.673; %(kg)  
Jk=0.00044; %(kg*m^2)  
Jv=0.00673; %(kg*m^2)  
L=0.2; %(m)  
K=0.006123; %Motor nyomatek konstans  
Ki=0.0069203; %Back EMF konstans  
R=3; %Motor ellenallas  
sim( 'nemlinearis_modell1_simulink_vissza',20);  
%rajzolas  
subplot(4,1,1);  
plot(szog.time, szog.signals.values)  
grid on;  
xlabel( 'Ido[s] ');  
%ylabel( 'Inga dolesszoge' );  
legend( 'Szog[rad] ');  
subplot(4,1,2);  
plot(szog_derivalt.time, szog_derivalt.signals.values)  
grid on;  
xlabel( 'Ido[s] ');
```

```

%ylabel('Inga szogsebessege ');
legend('Szog_gyorsulas[rad/s]');
subplot(4,1,3);
plot(pozicio.time, pozicio.signals.values)
grid on;
xlabel('Ido[s]');
%ylabel('Kerek szogelfordulasa ');
legend('Pozicio[m]');
subplot(4,1,4);
plot(pozicio_derivalt.time, pozicio_derivalt.signals.values)
grid on;
xlabel('Ido[s]');
%ylabel('Motor szogsebessege ');
legend('Gyorsulas[m/s]');

```

A.2.3. Lineáris modell LQ

```

g=9.81; %(m/s^2)
r=0.0900 %(m)
mk=0.044; %(kg)
mv= 0.673; %(kg)
Jk=0.00044; %(kg*m^2)
Jv=0.00673; %(kg*m^2)
L=0.2; %(m)
K=0.006123; %Motor nyomatek konstans
Ki=0.0069203; %Back EMF konstans
R=3; %Motor ellenallas
alpha = (L^2*mv^2-(Jv+L^2*mv)*(2*mk+mv+(2*Jk)/r^2));
beta = (R*(2*Jk*Jv+2*Jk*L^2*mv+2*Jv*mk*r^2+Jv*mv*r^2+
+2*L^2*mk*mv*r^2));
A = [0 1 0 0;
0 ((2*K*Ki*(L*mv*r-L^2*mv-Jv))/beta
(g*L^2*mv^2*r^2*R)/beta 0;
0 0 0 1;
0 ((2*K*Ki)/(r*R))*((L*mv)/
/r-(2*Jk)/r^2-2*mk-mv))/alpha
(g*L*mv*(-2*mk-mv- -(2*Jk)/r^2))/alpha 0]

```

```
B = [0;
      (2*K*r*(Jv+L^2*mv-L*mv*r))/beta;
      0;
      (((2*K)/R)*(2*mk+mv+(2*Jk)/r^2-(L*mv)/r))/alpha]
C = [1 0 0 0;
      0 1 0 0;
      0 0 1 0;
      0 0 0 1]
D = [0;
      0;
      0;
      0]
Q=[2000 0 0 0;
    0 20 0 0;
    0 0 1000 0;
    0 0 0 10]
R=10;
K2=lqr(A,B,Q,R);
sim('linearis_modell1_simulink_LQ',10);
%rajzolas
subplot(4,1,1);
plot(szog.time, szog.signals.values)
grid on;
xlabel('Ido[s]');
ylabel('Inga dolesszoge');
legend('Szog[rad]');
subplot(4,1,2);
plot(szog_derivalt.time, szog_derivalt.signals.values)
grid on;
xlabel('Ido[s]');
ylabel('Inga szogsebessege');
legend('Szog_gyorsulas[rad/s]');
subplot(4,1,3);
plot(pozicio.time, pozicio.signals.values)
grid on;
xlabel('Ido[s]');
ylabel('Kerek szogelfordulasa');
```

```
legend( ' Pozicio [m] ' );  
subplot( 4 , 1 , 4 );  
plot( pozicio_derivalt.time , pozicio_derivalt.signals.values )  
grid on;  
xlabel( ' Ido [s] ' );  
%ylabel( ' Motor szogsebessége ' );  
legend( ' Gyorsulas [m/s] ' );
```

A.2.4. Nemlineáris modell LQ

```
g=9.81; %(m/s^2)  
r=0.0900 %(m)  
mk=0.044; %(kg)  
mv= 0.673; %(kg)  
Jk=0.00044; %(kg*m^2)  
Jv=0.00673; %(kg*m^2)  
L=0.2; %(m)  
K=0.006123; %Motor nyomatek konstans  
Ki=0.0069203; %Back EMF konstans  
R=3; %Motor ellenallas  
Q=[2000 0 0 0;  
    0 20 0 0;  
    0 0 1000 0;  
    0 0 0 10]  
R=10;  
K2=lqr(A,B,Q,R);  
sim( 'nemlinearis_modell_simulink_LQ' ,10);  
%rajzolas  
subplot( 4 , 1 , 1 );  
plot( szog.time , szog.signals.values )  
grid on;  
xlabel( ' Ido [s] ' );  
%ylabel( ' Inga dolesszoge ' );  
legend( ' Szog [rad] ' );  
subplot( 4 , 1 , 2 );  
plot( szog_derivalt.time , szog_derivalt.signals.values )  
grid on;  
xlabel( ' Ido [s] ' );
```

```

%ylabel('Inga szögsebessége');
legend('Szöggyorsulás[rad/s]');
subplot(4,1,3);
plot(pozicio.time, pozicio.signals.values)
grid on;
xlabel('Ido[s]');
%ylabel('Kerek szögelfordulása');
legend('Pozicio[m]');
subplot(4,1,4);
plot(pozicio_derivalt.time, pozicio_derivalt.signals.values)
grid on;
xlabel('Ido[s]');
%ylabel('Motor szögsebessége');
legend('Gyorsulás[m/s]');

```

A.2.5. Lineáris modell szervo

```

g=9.81; %(m/s^2)
r=0.0900 %(m)
mk=0.044; %(kg)
mv= 0.673; %(kg)
Jk=0.00044; %(kg*m^2)
Jv=0.00673; %(kg*m^2)
L=0.2; %(m)
K=0.006123; %Motor nyomatek konstans
Ki=0.0069203; %Back EMF konstans
R=3; %Motor ellenallas
A22=(-2*K*Ki*(Jv + L^2*mv) + 2*K*Ki*L*mv*r)/((( -L^2)*mv^2*r^2+
+ (Jv + L^2*mv)*(2*Jk + (2*mk + mv)*r^2))*R);
A23=(g*L^2*mv^2*r^2)/((-L^2)*mv^2*r^2 + (Jv + L^2*mv)*
*(2*Jk + (2*mk + mv)*r^2));
A42=(8*Jk*K*Ki - 4*K*Ki*L*mv*r + 8*K*Ki*mk*r^2 + 4*K*Ki*mv*r^2)/
/(r*(4*Jk*(Jv + L^2*mv) - L^2*mv^2*r^2+(2*Jv*(2*mk + mv)+
+ L^2*mv*(4*mk + mv))*r^2)*R);
A43=(2*g*L*mv*(2*Jk + (2*mk + mv)*r^2))/(4*Jk*(Jv + L^2*mv)-
-L^2*mv^2*r^2 + (2*Jv*(2*mk + mv) + L^2*mv*(4*mk + mv))*r^2);
B2=(2*K*(Jv + L^2*mv)*r - 2*K*L*mv*r^2)/((( -L^2)*mv^2*r^2 +
+(Jv + L^2*mv)*(2*Jk + (2*mk + mv)*r^2))*R);

```

```

B4=(-8*Kk*K*r + 4*K*L*mv*r^2 - 8*K*mk*r^3 - 4*K*mv*r^3)/
    /(r*(4*Kk*(Jv + L^2*mv) - L^2*mv^2*r^2 +(2*Jv*(2*mk + mv)+
    + L^2*mv*(4*mk + mv))*r^2)*R);
A = [0 1 0 0;
      0 A22 A23 0;
      0 0 0 1;
      0 A42 A43 0]
B = [0;
      B2;
      0;
      B4]
C = [1 0 0 0;
      0 1 0 0;
      0 0 1 0;
      0 0 0 1]
D = [0;
      0;
      0;
      0]
nullak=[0;0;0;0];
Ap=[A nullak];
C_csillag=[1 0 0 0];
Cp=[C_csillag 0];
A_csillag=[Ap ;Cp]
B_csillag=[B;0]
K_csillag=place(A_csillag,B_csillag,[-1 -5 -6 -7 -8])
Kx=K_csillag(1:4);
Kdelta=K_csillag(5);
sim('linearis_modell1_simulink_szervo',10);

```

%rajzolas

A.2.6. Nemlineáris modell megfigyelo

```

clear;
clc;
close all;
g=9.81; %(m/s^2)

```

```

r=0.0900 %(m)
mk=0.044; %(kg)
mv= 0.673; %(kg)
Jk=0.00044; %(kg*m^2)
Jv=0.00673; %(kg*m^2)
L=0.2; %(m)
K=0.006123; %Motor nyomatek konstans
Ki=0.0069203; %Back EMF konstans
R=3; %Motor ellenallas
alpha = (L^2*mv^2-(Jv+L^2*mv)*(2*mk+mv+(2*Jk)/r^2));
beta = (R*(2*Jk*Jv+2*Jk*L^2*mv+2*Jv*mk*r^2+Jv*mv*r^2+
+2*L^2*mk*mv*r^2));
A = [0 1 0 0;
0 ((2*K*Ki*(L*mv*r-L^2*mv-Jv)))/beta
(g*L^2*mv^2*r^2*R)/beta 0;
0 0 0 1;
0 ((2*K*Ki)/(r*R))*((L*mv)/
/r-(2*Jk)/r^2-2*mk-mv))/alpha
(g*L*mv*(-2*mk-mv-
-(2*Jk)/r^2))/alpha 0]

B = [0;
(2*K*r*(Jv+L^2*mv-L*mv*r))/beta;
0;
(((2*K)/R)*(2*mk+mv+(2*Jk)/r^2-(L*mv)/r))/alpha]
C = [1 0 0 0;
0 1 0 0;
0 0 1 0;
0 0 0 1]
D = [0;
0;
0;
0]
K=place(A,B,[-5 -6 -10 -11])
L2=place(A,C,[-80 -90 -140 -150])
%kezdeti ertekek
x0=[5,0,1,0];
sim('linearis_modell_simulink_megfigyelovel2.slx');

```

```
figure
plot(ido,y,'LineWidth',2);
grid on
hold on
plot(ido,r,'r—','LineWidth',2);
legend('y(1)','y(2)','y(3)','y(4)','r(t)')
figure
subplot(4,1,1)
plot(ido,x(:,1),'LineWidth',2);
grid on
hold on
plot(ido,x_obs(:,1),'r—','LineWidth',2);
legend('x_1(t)','x^{obs}_1(t)')
subplot(4,1,2)
plot(ido,x(:,2),'LineWidth',2);
grid on
hold on
plot(ido,x_obs(:,2),'r—','LineWidth',2);
legend('x_2(t)','x^{obs}_2(t)')
subplot(4,1,3)
plot(ido,x(:,3),'LineWidth',2);
grid on
hold on
plot(ido,x_obs(:,3),'r—','LineWidth',2);
legend('x_3(t)','x^{obs}_3(t)')
subplot(4,1,4)
plot(ido,x(:,4),'LineWidth',2);
grid on
hold on
plot(ido,x_obs(:,4),'r—','LineWidth',2);
legend('x_4(t)','x^{obs}_4(t)')
```

A.3. Mathematica kód

A.3.1. Pi körüli linearizálás

```

eq1 :=
  
$$\phi'' (Jv + mv L^2) + 2 \left( (Uk K) / R - (xp K Ki) / (R r) \right) + L mv g \sin[\phi] = -L mv xpp \cos[\phi]$$


eq2 := (2 Uk K) / (r R) == mv L  $\phi'' \cos[\phi]$  -
  
$$mv \phi'^2 L \sin[\phi] + (2 K Ki xp) / (R r^2) + xpp (mv + (2 Jk) / (r^2) + 2 mk)$$


Solve[{eq1 && eq2}, {xpp,  $\phi''$ }]


$$\left\{ \left\{ xpp \rightarrow - \left( -L mv \cos[\phi] \left( 2 \left( \frac{K Uk}{R} - \frac{K Ki xp}{r R} \right) + g L mv \sin[\phi] \right) - \right. \right.$$


$$\left. \left( Jv + L^2 mv \right) \left( \frac{2 K Uk}{r R} - \frac{2 K Ki xp}{r^2 R} + L mv \phi'^2 \sin[\phi] \right) \right) / \right.$$


$$\left. \left( - \left( Jv + L^2 mv \right) \left( -2 mk - mv - \frac{2 Jk}{r^2} \right) - L^2 mv^2 \cos[\phi]^2 \right) \right\},$$


$$\phi'' \rightarrow \frac{\sec[\phi] (2 K r Uk - 2 K Ki xp + L mv r^2 R \phi'^2 \sin[\phi])}{L mv r^2 R} -$$


$$\left( \left( -2 mk - mv - \frac{2 Jk}{r^2} \right) \sec[\phi] \left( -L mv \cos[\phi] \left( 2 \left( \frac{K Uk}{R} - \frac{K Ki xp}{r R} \right) + g L mv \sin[\phi] \right) - \right. \right.$$


$$\left. \left( Jv + L^2 mv \right) \left( \frac{2 K Uk}{r R} - \frac{2 K Ki xp}{r^2 R} + L mv \phi'^2 \sin[\phi] \right) \right) \right) /$$


$$\left( L mv \left( - \left( Jv + L^2 mv \right) \left( -2 mk - mv - \frac{2 Jk}{r^2} \right) - L^2 mv^2 \cos[\phi]^2 \right) \right) \right\}$$


Simplify[%]


$$\left\{ \left\{ xpp \rightarrow (L mv r \cos[\phi] (2 K r Uk - 2 K Ki xp + g L mv r R \sin[\phi]) + \right. \right.$$


$$(Jv + L^2 mv) (2 K r Uk - 2 K Ki xp + L mv r^2 R \phi'^2 \sin[\phi])) /$$


$$(R ((Jv + L^2 mv) (2 Jk + (2 mk + mv) r^2) - L^2 mv^2 r^2 \cos[\phi]^2)),$$


$$\phi'' \rightarrow - (8 Jk K r Uk + 8 K mk r^3 Uk + 4 K mv r^3 Uk - 8 Jk K Ki xp -$$


$$8 K Ki mk r^2 xp - 4 K Ki mv r^2 xp + 4 K L mv r (r Uk - Ki xp) \cos[\phi] +$$


$$2 g L mv r (2 Jk + (2 mk + mv) r^2) R \sin[\phi] + L^2 mv^2 r^3 R \phi'^2 \sin[2 \phi]) / (r R$$


$$(4 Jk (Jv + L^2 mv) + (2 Jv (2 mk + mv) + L^2 mv (4 mk + mv)) r^2 - L^2 mv^2 r^2 \cos[2 \phi])) \right\}$$


kif1 := (L mv r Cos[ $\phi$ ] (2 K r Uk - 2 K Ki xp + g L mv r R Sin[ $\phi$ ]) +
  (Jv + L2 mv) (2 K r Uk - 2 K Ki xp + L mv r2 R  $\phi'^2 \sin[\phi]$ )) /
  (R ((Jv + L2 mv) (2 Jk + (2 mk + mv) r2) - L2 mv2 r2 Cos[ $\phi$ ]2))

kif2 := - (8 Jk K r Uk + 8 K mk r3 Uk + 4 K mv r3 Uk - 8 Jk K Ki xp -
  8 K Ki mk r2 xp - 4 K Ki mv r2 xp + 4 K L mv r (r Uk - Ki xp) Cos[ $\phi$ ] +
  2 g L mv r (2 Jk + (2 mk + mv) r2) R Sin[ $\phi$ ] + L2 mv2 r3 R  $\phi'^2 \sin[2 \phi]$ ) /
  (r R (4 Jk (Jv + L2 mv) + (2 Jv (2 mk + mv) + L2 mv (4 mk + mv)) r2 - L2 mv2 r2 Cos[2  $\phi$ ]))

A11 := D[kif1, x] /. { $\phi \rightarrow \text{Pi}$ ,  $\phi' \rightarrow 0$ , x  $\rightarrow 0$ , xp  $\rightarrow 0$ , Uk  $\rightarrow 0$ }

A11
0

A22 := D[kif1, xp] /. { $\phi \rightarrow \text{Pi}$ ,  $\phi' \rightarrow 0$ , x  $\rightarrow 0$ , xp  $\rightarrow 0$ , Uk  $\rightarrow 0$ }

```

A22

$$\frac{-2 \text{K Ki} \left(\text{Jv} + \text{L}^2 \text{mv} \right) + 2 \text{K Ki} \text{L mv r}}{\left(-\text{L}^2 \text{mv}^2 \text{r}^2 + \left(\text{Jv} + \text{L}^2 \text{mv} \right) \left(2 \text{Jk} + \left(2 \text{mk} + \text{mv} \right) \text{r}^2 \right) \right) \text{R}}$$

A23 := D[kif1, ϕ] /. {ϕ → Pi, ϕp → 0, x → 0, xp → 0, Uk → 0}**A23**

$$\frac{\text{g L}^2 \text{mv}^2 \text{r}^2}{-\text{L}^2 \text{mv}^2 \text{r}^2 + \left(\text{Jv} + \text{L}^2 \text{mv} \right) \left(2 \text{Jk} + \left(2 \text{mk} + \text{mv} \right) \text{r}^2 \right)}$$

A41 := D[kif2, x] /. {ϕ → Pi, ϕp → 0, x → 0, xp → 0, Uk → 0}**A41**

0

A42 := D[kif2, xp] /. {ϕ → Pi, ϕp → 0, x → 0, xp → 0, Uk → 0}**A42**

$$\frac{\left(8 \text{Jk K Ki} - 4 \text{K Ki} \text{L mv r} + 8 \text{K Ki} \text{mk r}^2 + 4 \text{K Ki} \text{mv r}^2 \right) /}{\left(\text{r} \left(4 \text{Jk} \left(\text{Jv} + \text{L}^2 \text{mv} \right) - \text{L}^2 \text{mv}^2 \text{r}^2 + \left(2 \text{Jv} \left(2 \text{mk} + \text{mv} \right) + \text{L}^2 \text{mv} \left(4 \text{mk} + \text{mv} \right) \right) \text{r}^2 \right) \text{R} \right)}$$

A43 := D[kif2, ϕ] /. {ϕ → Pi, ϕp → 0, x → 0, xp → 0, Uk → 0}**A43**

$$\frac{2 \text{g L mv} \left(2 \text{Jk} + \left(2 \text{mk} + \text{mv} \right) \text{r}^2 \right)}{4 \text{Jk} \left(\text{Jv} + \text{L}^2 \text{mv} \right) - \text{L}^2 \text{mv}^2 \text{r}^2 + \left(2 \text{Jv} \left(2 \text{mk} + \text{mv} \right) + \text{L}^2 \text{mv} \left(4 \text{mk} + \text{mv} \right) \right) \text{r}^2}$$

kif1

$$\frac{\left(\text{L mv r Cos}[\phi] \left(2 \text{K r Uk} - 2 \text{K Ki xp} + \text{g L mv r R Sin}[\phi] \right) + \left(\text{Jv} + \text{L}^2 \text{mv} \right) \left(2 \text{K r Uk} - 2 \text{K Ki xp} + \text{L mv r}^2 \text{R } \phi^2 \text{Sin}[\phi] \right) \right) /}{\left(\text{R} \left(\left(\text{Jv} + \text{L}^2 \text{mv} \right) \left(2 \text{Jk} + \left(2 \text{mk} + \text{mv} \right) \text{r}^2 \right) - \text{L}^2 \text{mv}^2 \text{r}^2 \text{Cos}[\phi]^2 \right) \right)}$$

Collect[kif2, Uk]

$$\frac{\text{Uk} \left(-8 \text{Jk K r} - 8 \text{K mk r}^3 - 4 \text{K mv r}^3 - 4 \text{K L mv r}^2 \text{Cos}[\phi] \right)}{\text{r R} \left(4 \text{Jk} \left(\text{Jv} + \text{L}^2 \text{mv} \right) + \left(2 \text{Jv} \left(2 \text{mk} + \text{mv} \right) + \text{L}^2 \text{mv} \left(4 \text{mk} + \text{mv} \right) \right) \text{r}^2 - \text{L}^2 \text{mv}^2 \text{r}^2 \text{Cos}[2 \phi] \right)} +$$

$$\frac{\left(8 \text{Jk K Ki xp} + 8 \text{K Ki} \text{mk r}^2 \text{xp} + 4 \text{K Ki} \text{mv r}^2 \text{xp} + 4 \text{K Ki} \text{L mv r xp Cos}[\phi] - 2 \text{g L mv r} \left(2 \text{Jk} + \left(2 \text{mk} + \text{mv} \right) \text{r}^2 \right) \text{R Sin}[\phi] - \text{L}^2 \text{mv}^2 \text{r}^3 \text{R } \phi^2 \text{Sin}[2 \phi] \right) /}{\left(\text{r R} \left(4 \text{Jk} \left(\text{Jv} + \text{L}^2 \text{mv} \right) + \left(2 \text{Jv} \left(2 \text{mk} + \text{mv} \right) + \text{L}^2 \text{mv} \left(4 \text{mk} + \text{mv} \right) \right) \text{r}^2 - \text{L}^2 \text{mv}^2 \text{r}^2 \text{Cos}[2 \phi] \right) \right)}$$

A.4. Számítások

A.4.1. Használt motor paramétereinek mérése

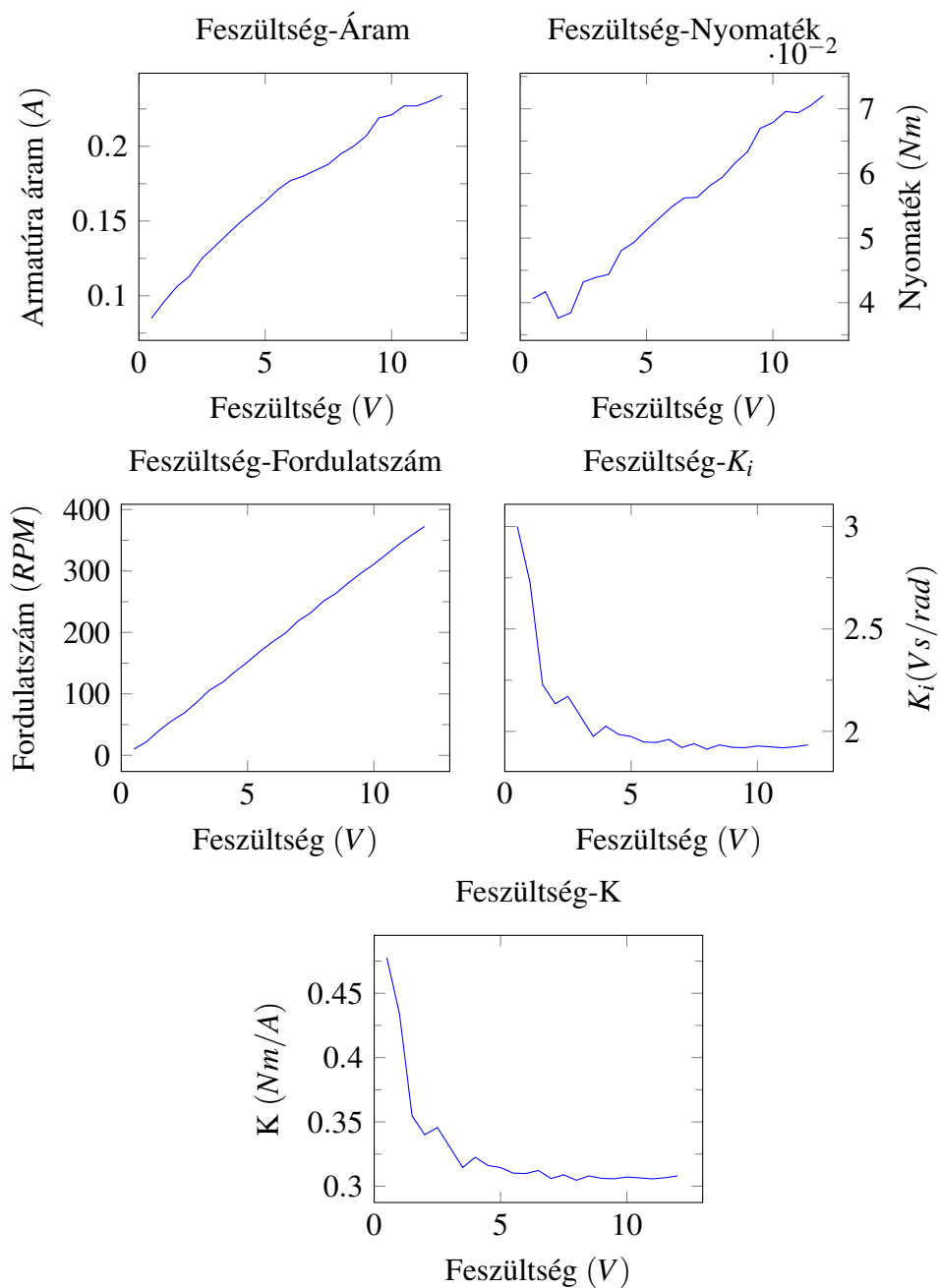
Feszültség (V)	Armatúra áram (A)	Fordulatszám (RPM)	Teljesítmény (P)	Nyomaték (Nm)
0,50	0,085	10	0,0425	0,040605
1,00	0,096	22	0,096	0,041691
1,50	0,106	40,4	0,159	0,037602
2,00	0,113	56,2	0,226	0,038421
2,50	0,125	69,1	0,3125	0,043208
3,00	0,133	86,8	0,399	0,043918
3,50	0,141	106,3	0,4935	0,044355
4,00	0,149	118,5	0,596	0,048053
4,50	0,156	136	0,702	0,049316
5,00	0,163	151,9	0,815	0,051262
5,50	0,171	169,4	0,9405	0,053044
6,00	0,177	185	1,062	0,054846
6,50	0,180	198,9	1,17	0,056201
7,00	0,184	218,6	1,288	0,056293
7,50	0,188	232	1,41	0,058066
8,00	0,195	250,9	1,56	0,059404
8,50	0,200	263,7	1,7	0,061593
9,00	0,207	280,9	1,863	0,063365
9,50	0,219	296,8	2,0805	0,066972
10,00	0,221	311,1	2,21	0,067871
10,50	0,227	327,3	2,3835	0,069576
11,00	0,227	343,8	2,497	0,069391
11,50	0,230	358,4	2,645	0,07051
12,00	0,234	372,3	2,808	0,07206

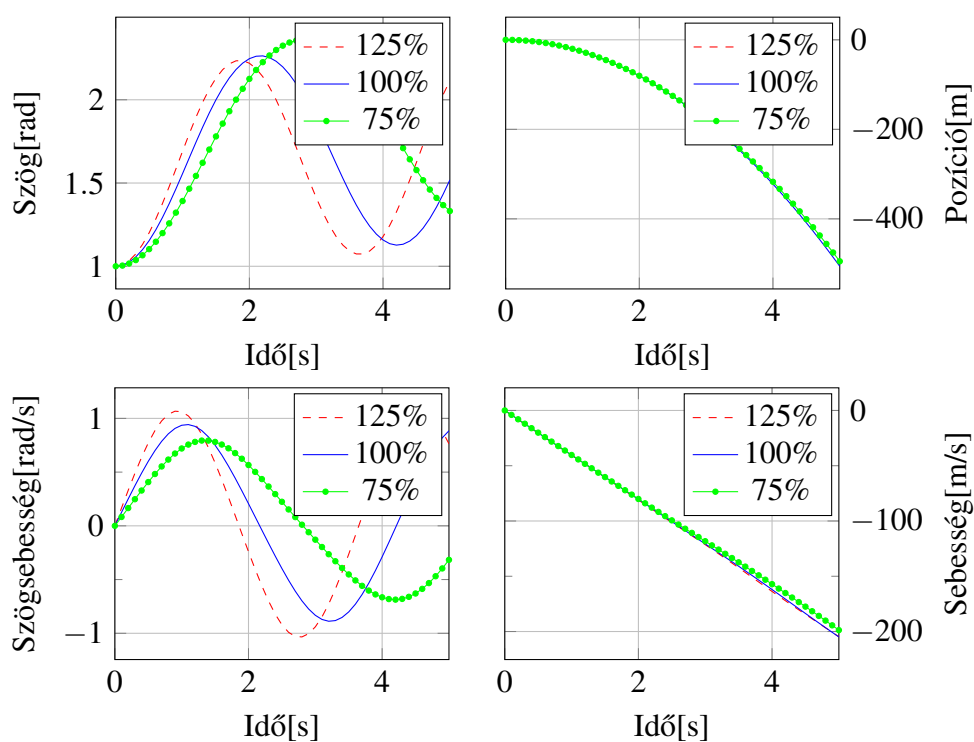
A.2. táblázat. Motor paramétereinek mérései

A.4.2. 0 körül linearizált állapotter modell

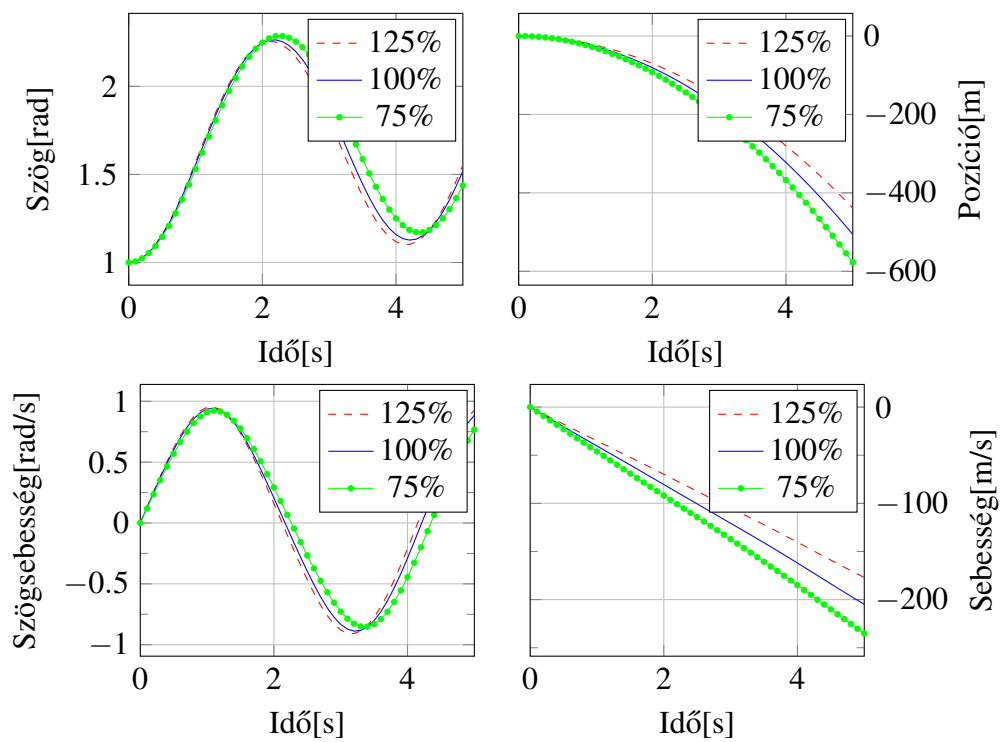
$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & -12.9317 & 15.9452 & 0 \\ 0 & 0 & 0 & 1.0000 \\ 0 & 60.0000 & -103.0210 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0.604913 \\ 0 \\ -2.81371 \end{bmatrix} \left[U_k \right]$$

A.5. Paraméterérzékenység

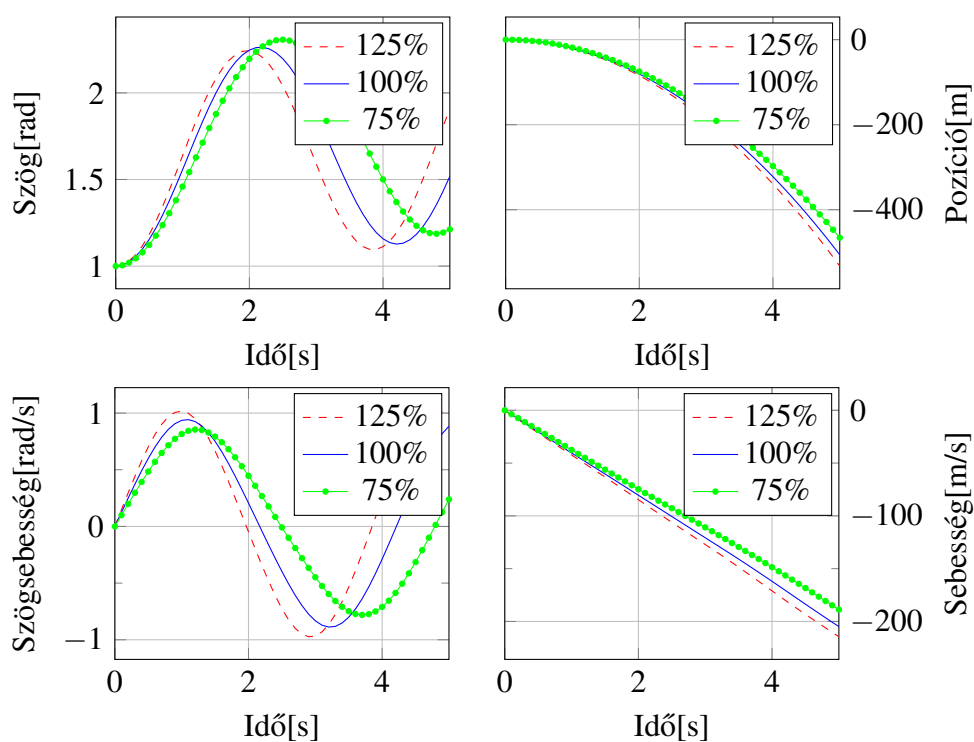




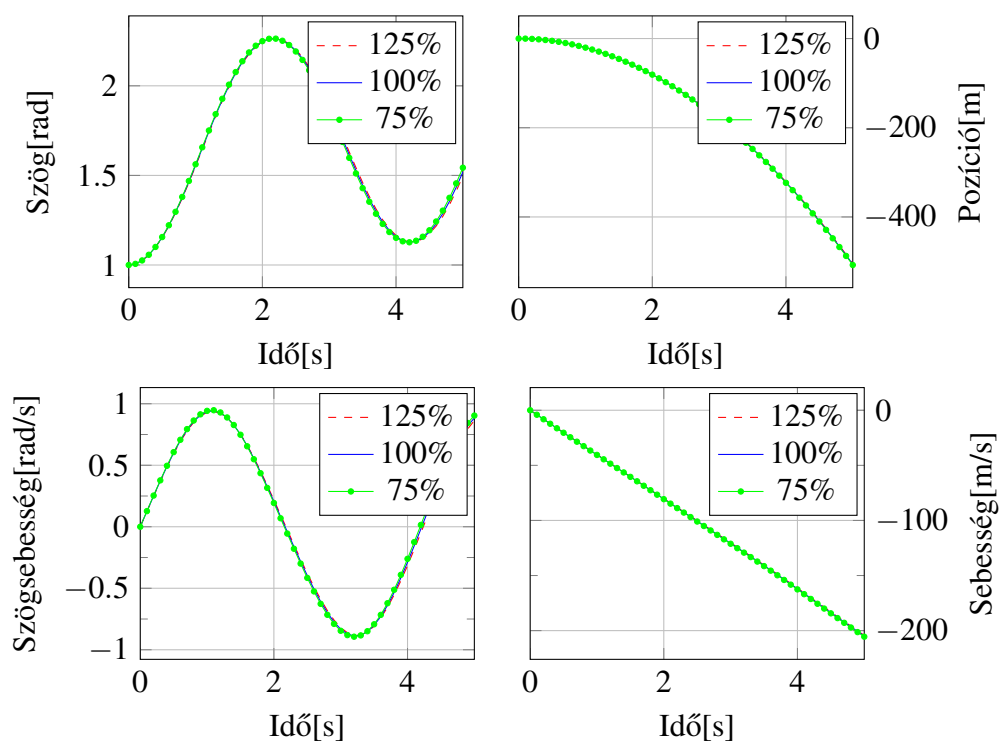
A.1. ábra. Kerék sugarának vizsgálata



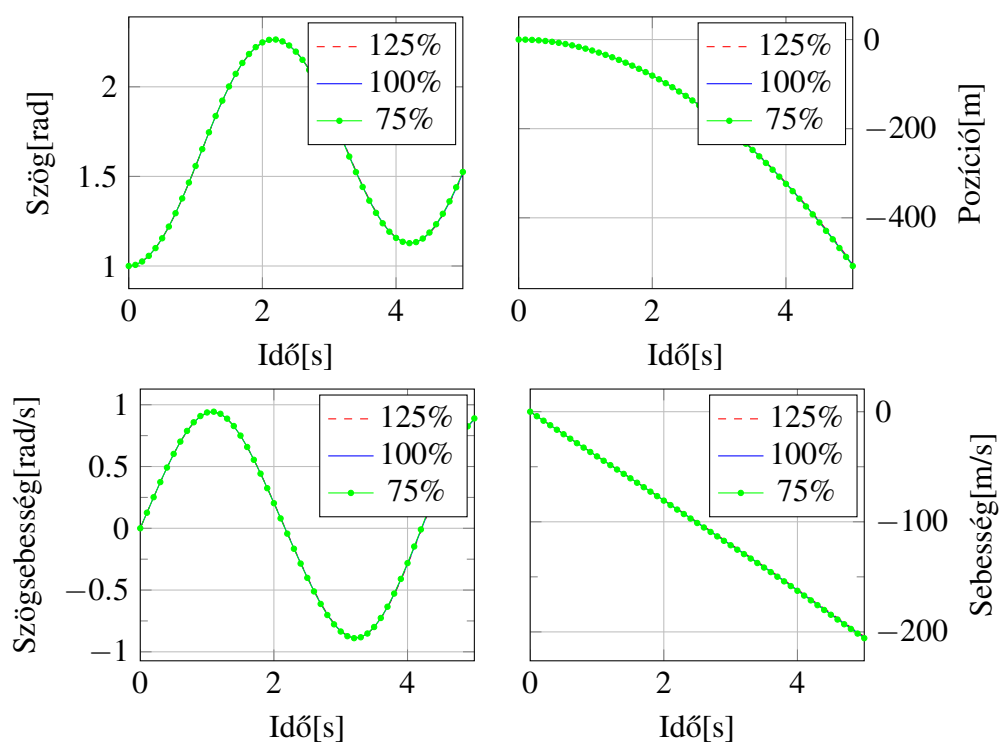
A.2. ábra. Inga hosszának a vizsgálata



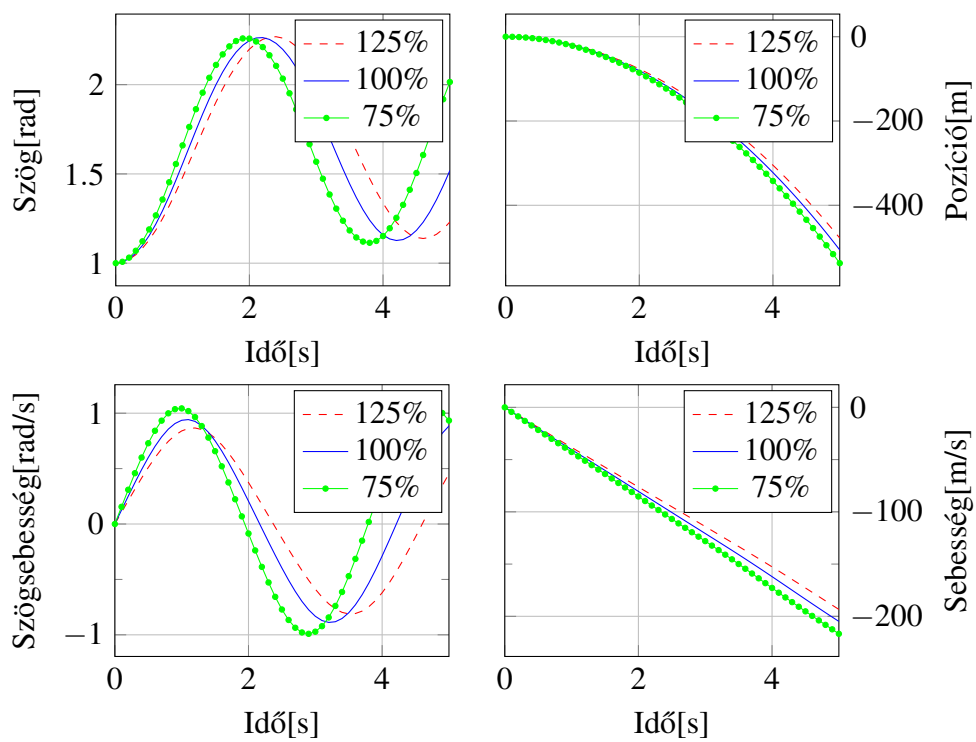
A.3. ábra. Inga tömegének a vizsgálata



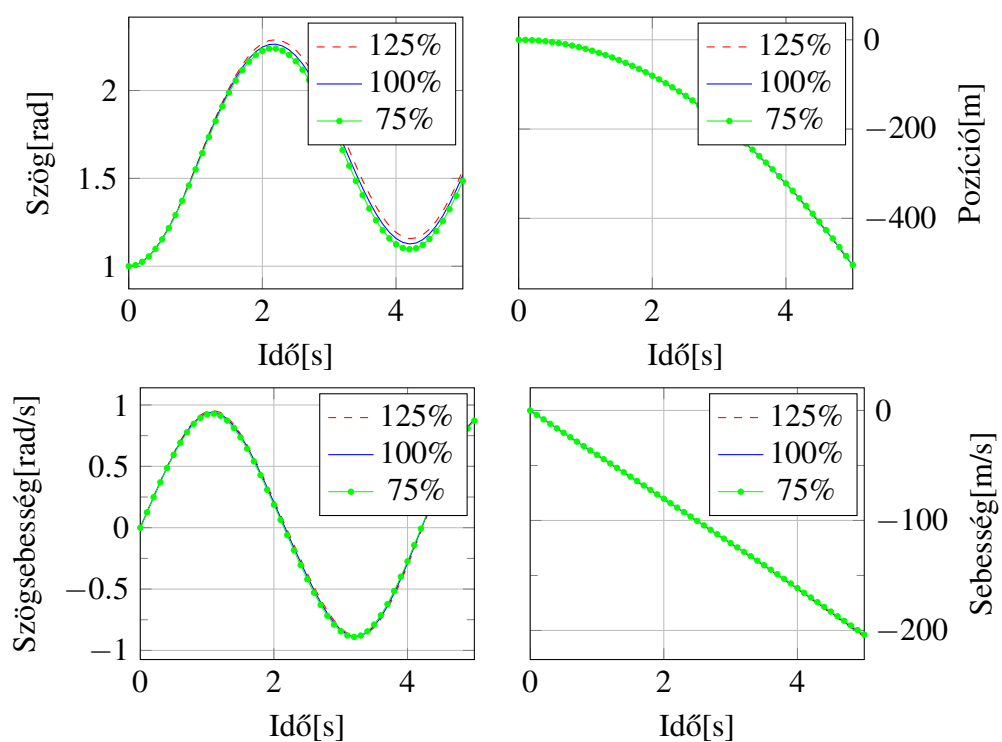
A.4. ábra. A kerék tömegének a vizsgálata



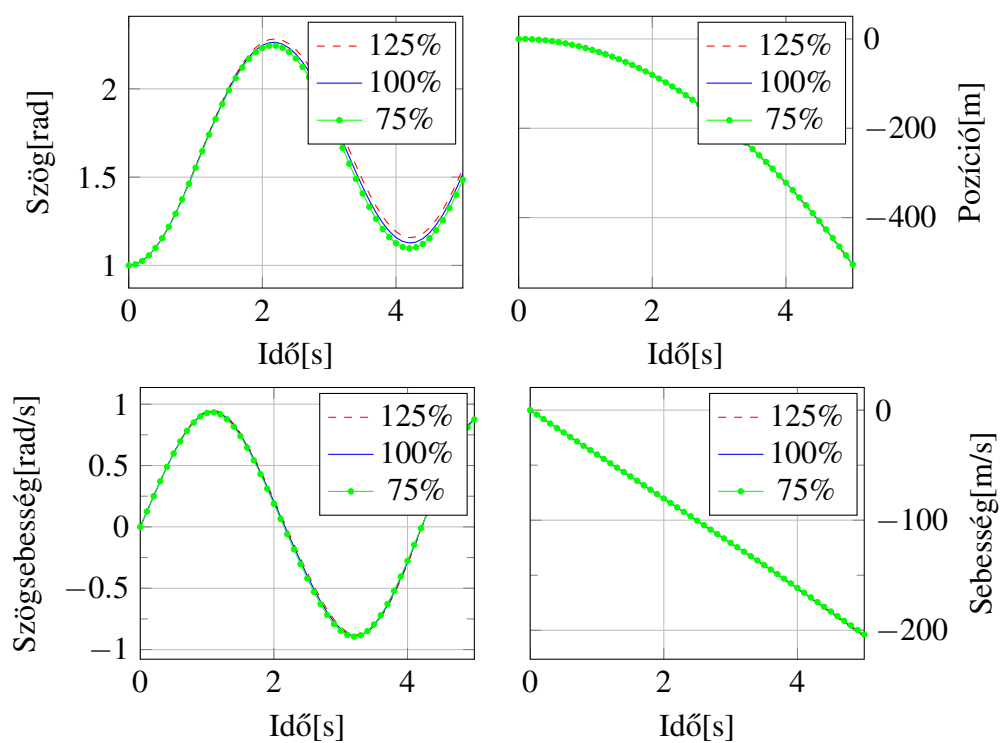
A.5. ábra. A kerék tehetetlenségének a vizsgálata



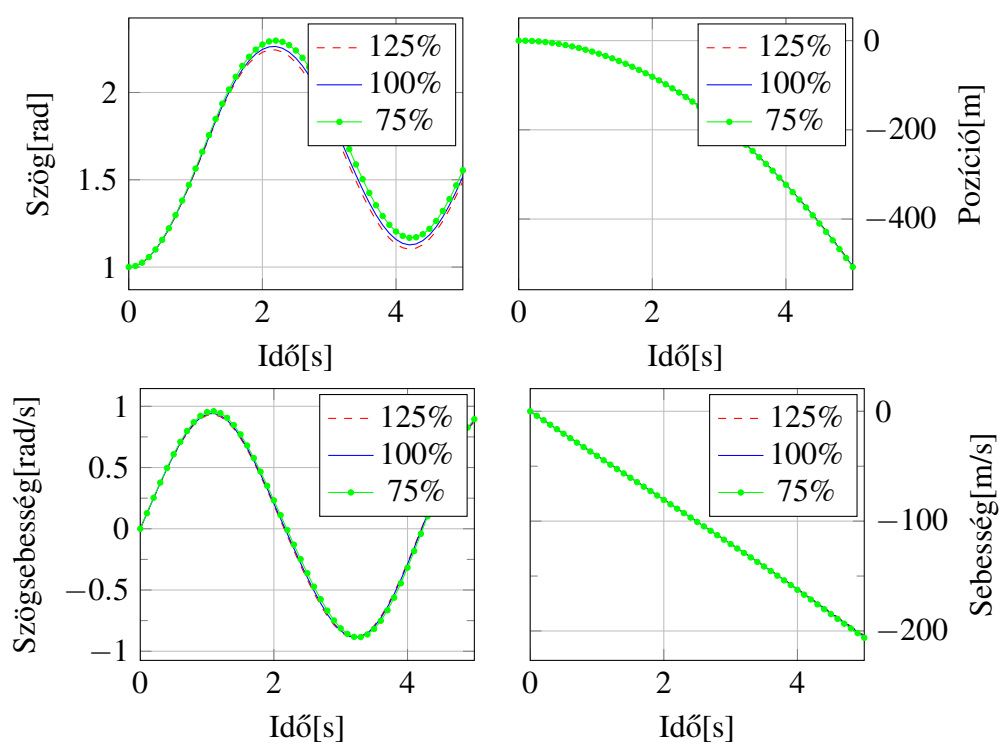
A.6. ábra. A váz tehetetlenségének a vizsgálata



A.7. ábra. K konstans vizsgálata



A.8. ábra. K_i konstans vizsgálata



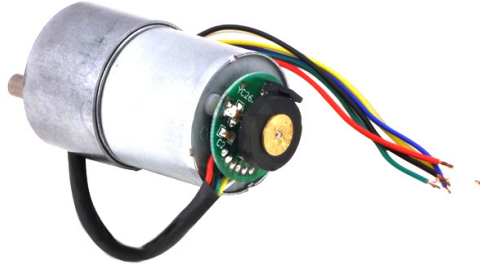
A.9. ábra. R konstans vizsgálata

B. Implementáció

B.1. Hardverelemek adatlapjai

B.1.1. Használt motor adatlapja

RB-Pol-124
Pololu 12V, 29:1 Gear Motor w/Encoder



This 2.54" × 1.45" × 1.45" gearmotor is a powerful 12V motor with a **29:1** metal gearbox and an integrated quadrature encoder that provides a resolution of 64 counts per revolution of the motor shaft, which corresponds to **1856 counts per revolution** of the gearbox's output shaft. These units have a 0.61"-long, 6 mm-diameter D-shaped output shaft. This gearmotor is also available without an encoder.

Key specs at **12 V**: 350 RPM and 300 mA free-run, 110 oz-in (8 kg-cm) and 5 A stall.

Gearmotor Options

This powerful gearmotor is available in six different gear ratios and features an integrated quadrature encoder with 64 counts per revolution (CPR) of the motor shaft. A version without the encoder is also available.

Gear Ratio	With Encoder	Without Encoder
19:1	Metal Gearmotor 37Dx52L mm	Metal Gearmotor 37Dx52L mm
29:1	Metal Gearmotor 37Dx52L mm	Metal Gearmotor 37Dx52L mm
50:1	Metal Gearmotor 37Dx54L mm	Metal Gearmotor 37Dx54L mm
67:1	Metal Gearmotor 37Dx54L mm	Metal Gearmotor 37Dx54L mm
100:1	Metal Gearmotor 37Dx57L mm	Metal Gearmotor 37Dx57L mm
131:1	Metal Gearmotor 37Dx57L mm	Metal Gearmotor 37Dx57L mm

Note: The version we used to call “122:1 metal gearmotor 37Dx57L mm” is now “131:1”. The gear ratio is now more accurately specified, but the other specifications have not changed.

These motors are intended for use at 12 V, though the motor can begin rotating at voltages as low as 1 V.

Gearmotor Dimensions

The face plate has six mounting holes evenly spaced around the outer edge threaded for M3 screws. These mounting holes form a regular hexagon and the centers of neighboring holes are 15.5 mm apart. You can use our custom 37D mm metal gearmotor bracket (shown in the left picture below) to mount the gearmotor to your project via these mounting holes and the screws that come with the bracket.

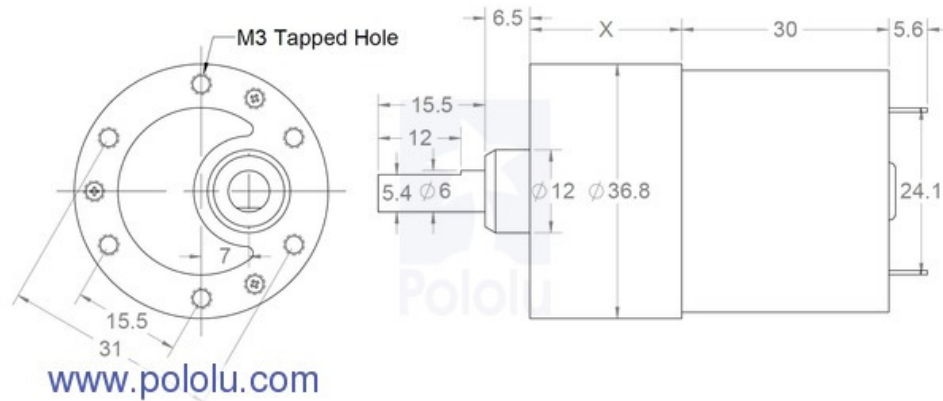


Gearmotor with bracket and hub.

37D mm metal gear motor with 64 CPR encoder and Pololu 90×10mm wheel.

Please note that unlike our smaller metal gearmotors, these 37D mm gearmotors have output shafts with a diameter of 6 mm. The Pololu universal aluminum mounting hub for 6mm shafts can be used to mount our larger Pololu wheels (80mm- and 90mm-diameter) or custom wheels and mechanisms to the gearmotor's output shaft (see the right picture above).

The diagram below shows the dimensions (in mm) of the 37D mm line of gearmotors. The value of x is **22 mm** for the 19:1 37Dx52L mm and 29:1 37Dx52L mm versions, **24 mm** for the 50:1 37Dx54L mm and 67:1 37Dx54L mm versions, and **26.5 mm** for the 100:1 37Dx57L mm and 131:1 37Dx57L mm versions. Note that the encoder PCB and magnetic disc are not shown in this dimension diagram. The encoder assembly extends an additional 12.5 mm beyond the rear of the motor.



37D mm metal gearmotor dimensions (units in mm).

Warning: Do not screw too far into the mounting holes as the screws can hit the gears. We recommend screwing no further than 3mm (1/8") into the screw hole.

Using the Encoder

A two-channel Hall effect encoder is used to sense the rotation of a magnetic disk on a rear protrusion of the motor shaft. The quadrature encoder provides a resolution of 64 counts per revolution of the motor shaft. To compute the counts per revolution of the gearbox output, multiply the gear ratio by 64. The motor/encoder has six color-coded, 11" (28 cm) leads:



37D mm metal gear motor with 64 CPR encoder.

Color	Function
Black	motor power
Red	motor power
Blue	Hall sensor Vcc (3.5 – 20 V)
Green	Hall sensor GND
Yellow	Hall sensor A output
White	Hall sensor B output

These leads have stripped, unterminated ends that can be soldered or crimped to make solder free connectors. The Hall sensor requires an input voltage, Vcc, between 3.5 and 20 V and draws a maximum of 10 mA. The A and B outputs are square waves from 0 V to Vcc approximately 90° out of phase. The frequency of the transitions tells you the speed of the motor, and the order of the transitions tells you the direction. The following oscilloscope capture shows the A and B (yellow and white) encoder outputs using a motor voltage of 12 V and a Hall sensor Vcc of 5 V:

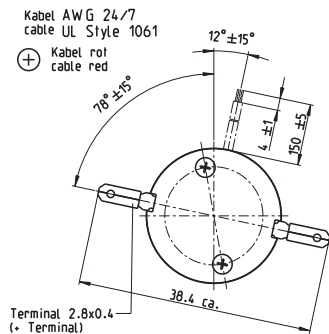


Encoder A and B outputs for 37D mm metal gearmotor with 64 CPR encoder (motor running at 12 V).

By counting both the rising and falling edges of both the A and B outputs, it is possible to get 64 counts per revolution of the motor shaft. Using just a single edge of one channel results in 16 counts per revolution of the motor shaft, so the frequency of the A output in the above oscilloscope capture is 16 times the motor rotation frequency.

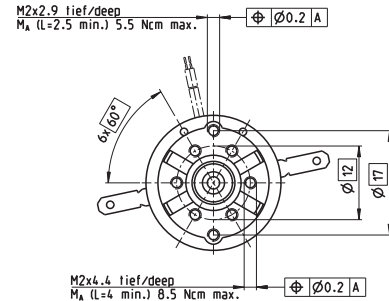
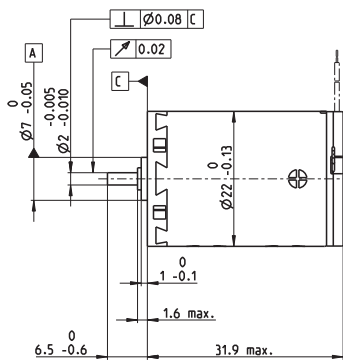
B.1.2. Maxon-motor katalóguslapja

A-max 22 Ø22 mm, Graphite Brushes, 6 Watt



M 1:1

- Stock program
- Standard program
- Special program (on request)



maxon A-max

Part Numbers

with terminals	110143	110145	110146	110147	110148	110149	110150	110151	110152	110153	110154	110155
with cables	139840	353017	199807	320206	323856	108828	199424	202921	267433	325492	313302	353019

Motor Data

Values at nominal voltage			6	9	9	12	12	15	18	24	24	36	48	48
1 Nominal voltage	V		6	9	9	12	12	15	18	24	24	36	48	48
2 No load speed	rpm		9240	9690	8500	10200	9170	10000	9770	10500	8480	9630	9110	8210
3 No load current	mA		83.1	57.9	49.6	45.8	40.5	36	29	23.7	18.4	14.2	9.99	8.84
4 Nominal speed	rpm		6240	6530	5350	7060	6000	6890	6600	7380	5270	6420	5840	4940
5 Nominal torque (max. continuous torque)	mNm		5.91	6.88	7.04	6.96	6.95	6.93	6.92	6.9	6.97	6.86	6.75	6.86
6 Nominal current (max. continuous current)	A		1.08	0.859	0.77	0.681	0.613	0.534	0.432	0.347	0.283	0.21	0.147	0.135
7 Stall torque	mNm		19.4	22.1	19.8	23.7	20.9	22.9	22	23.7	18.9	21.1	19.2	17.6
8 Starting current	A		3.29	2.59	2.04	2.17	1.72	1.65	1.29	1.12	0.721	0.606	0.393	0.325
9 Max. efficiency	%		67	70	69	72	70	72	72	73	70	72	71	70
Characteristics														
10 Terminal resistance	Ω		1.82	3.48	4.42	5.53	6.96	9.09	14	21.5	33.3	59.4	122	148
11 Terminal inductance	mH		0.106	0.223	0.288	0.363	0.445	0.585	0.891	1.37	2.1	3.69	7.3	8.97
12 Torque constant	mNm/A		5.9	8.55	9.73	10.9	12.1	13.9	17.1	21.2	26.2	34.8	48.9	54.3
13 Speed constant	rpm/V		1620	1120	981	875	790	689	558	450	364	274	195	176
14 Speed / torque gradient	rpm/mNm		500	454	446	444	455	452	457	456	461	468	487	479
15 Mechanical time constant	ms		20.9	20.2	20.1	19.9	19.9	19.7	19.7	19.8	19.7	19.9	19.8	
16 Rotor inertia	gcm ²		4	4.25	4.3	4.29	4.19	4.2	4.13	4.13	4.09	4.02	3.9	3.94

Specifications

Thermal data		
17 Thermal resistance housing-ambient	20 K/W	
18 Thermal resistance winding-housing	6.0 K/W	
19 Thermal time constant winding	9.43 s	
20 Thermal time constant motor	314 s	
21 Ambient temperature	-30...+85°C	
22 Max. permissible winding temperature	+125°C	

Mechanical data (sleeve bearings)		
23 Max. permissible speed	9800 rpm	
24 Axial play	0.05 - 0.15 mm	
25 Radial play	0.012 mm	
26 Max. axial load (dynamic)	1 N	
27 Max. force for press fits (static)	80 N	
28 Max. radial loading, 5 mm from flange	2.8 N	

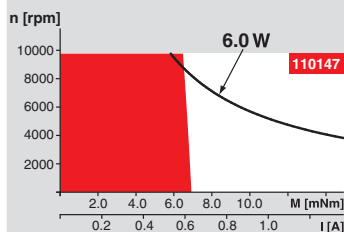
Mechanical data (ball bearings)		
23 Max. permissible speed	9800 rpm	
24 Axial play	0.05 - 0.15 mm	
25 Radial play	0.025 mm	
26 Max. axial load (dynamic)	3.3 N	
27 Max. force for press fits (static)	45 N	
28 Max. radial loading, 5 mm from flange	12.3 N	

Other specifications		
29 Number of pole pairs	1	
30 Number of commutator segments	9	
31 Weight of motor	54 g	

Values listed in the table are nominal.
Explanation of the figures on page 71.

Option
Ball bearings in place of sleeve bearings

Operating Range



Comments

- Continuous operation**
In observation of above listed thermal resistance (lines 17 and 18) the maximum permissible winding temperature will be reached during continuous operation at 25°C ambient.
= Thermal limit.
- Short term operation**
The motor may be briefly overloaded (recurring).
- Assigned power rating**

maxon Modular System

Overview on page 20 - 25

Planetary Gearhead

Ø22 mm
0.1 - 0.6 Nm
Page 248/249

Planetary Gearhead

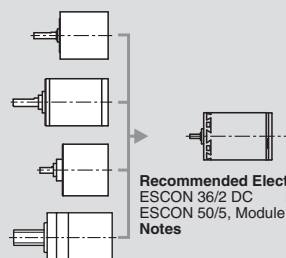
Ø22 mm
0.5 - 2.0 Nm
Page 250/251

Spur Gearhead

Ø24 mm
0.1 Nm
Page 256

Spindle Drive

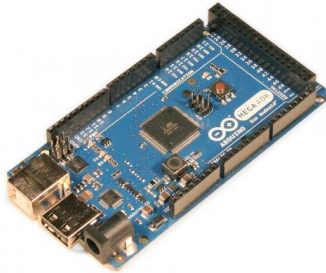
Ø22 mm
Page 284/285



Recommended Electronics:
ESCON 36/2 DC Page 320
ESCON 50/5, Module 50/5 321
Notes 22

B.1.3. Arduino mikrovezérlő

Arduino ADK Rev3



The Arduino ADK R3 is a microcontroller board based on the ATmega2560 ([datasheet](#)). It has a USB host interface to connect with Android based phones.

It is compatible with Android's Accessory Development Kit examples. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

The ADK is based on the [Mega 2560](#). Plus it has an USB Host circuit that enable this board to communicate with USB Devices, and give them power supply.

Additional features coming with the R3 version are:

- ATmega16U2 instead 8U2 as USB-to-Serial converter.
- 1.0 pinout: added SDA and SCL pins for TWI communication placed near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board and the second one is a not connected pin, that is reserved for future purposes.
- stronger RESET circuit.

Technical Specifications

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	9V
Input Voltage (limits)	7-18V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

B.1.4. Motorvezérlő



VNH2SP30-E

Automotive fully integrated H-bridge motor driver

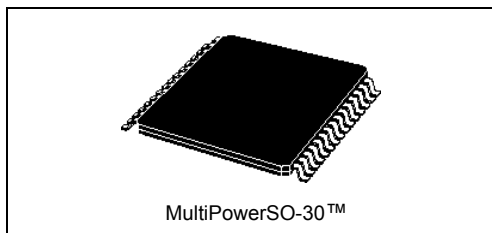
Features

Type	$R_{DS(on)}$	I_{out}	V_{CCmax}
VNH2SP30-E	19m Ω max (per leg)	30A	41V

- 5V logic level compatible inputs
- Undervoltage and overvoltage shut-down
- Overvoltage clamp
- Thermal shut down
- Cross-conduction protection
- Linear current limiter
- Very low stand-by power consumption
- PWM operation up to 20 kHz
- Protection against loss of ground and loss of V_{CC}
- Current sense output proportional to motor current
- Package: ECOPACK®

Description

The VNH2SP30-E is a full bridge motor driver intended for a wide range of automotive applications. The device incorporates a dual monolithic high side driver and two low side switches. The high side driver switch is designed using STMicroelectronics' well known and proven proprietary VIPower™ M0 technology which permits efficient integration on the same die of a true Power MOSFET with an intelligent signal/protection circuitry.



The low side switches are vertical MOSFETs manufactured using STMicroelectronics' proprietary EHD ('STripFET™') process. The three die are assembled in the MultiPowerSO-30 package on electrically isolated leadframes. This package, specifically designed for the harsh automotive environment offers improved thermal performance thanks to exposed die pads. Moreover, its fully symmetrical mechanical design allows superior manufacturability at board level. The input signals IN_A and IN_B can directly interface to the microcontroller to select the motor direction and the brake condition. The $DIAG_A/EN_A$ or $DIAG_B/EN_B$, when connected to an external pull-up resistor, enable one leg of the bridge. They also provide a feedback digital diagnostic signal. The normal condition operation is explained in [Table 12: Truth table in normal operating conditions on page 14](#). The motor current can be monitored with the CS pin by delivering a current proportional to its value. The speed of the motor can be controlled in all possible conditions by the PWM up to 20 kHz. In all cases, a low level state on the PWM pin will turn off both the LS_A and LS_B switches. When PWM rises to a high level, LS_A or LS_B turn on again depending on the input pin state.

Table 1. Device summary

Package	Order codes	
	Tube	Tape and Reel
MultiPowerSO-30	VNH2SP30-E	VNH2SP30TR-E

B.1.5. Gyorsulásmérő



3-Axis, $\pm 2\text{ g}/\pm 4\text{ g}/\pm 8\text{ g}/\pm 16\text{ g}$ Digital Accelerometer

Data Sheet

ADXL345

FEATURES

Ultralow power: as low as 23 μA in measurement mode and 0.1 μA in standby mode at $V_S = 2.5\text{ V}$ (typical)
Power consumption scales automatically with bandwidth
User-selectable resolution
Fixed 10-bit resolution
Full resolution, where resolution increases with g range, up to 13-bit resolution at $\pm 16\text{ g}$ (maintaining 4 mg/LSB scale factor in all g ranges)
Patent pending, embedded memory management system with FIFO technology minimizes host processor load
Single tap/double tap detection
Activity/inactivity monitoring
Free-fall detection
Supply voltage range: 2.0 V to 3.6 V
I/O voltage range: 1.7 V to V_S
SPI (3- and 4-wire) and I²C digital interfaces
Flexible interrupt modes mappable to either interrupt pin
Measurement ranges selectable via serial command
Bandwidth selectable via serial command
Wide temperature range (-40°C to $+85^\circ\text{C}$)
10,000 g shock survival
Pb free/RoHS compliant
Small and thin: 3 mm $\times$ 5 mm $\times$ 1 mm LGA package

APPLICATIONS

Handsets
Medical instrumentation
Gaming and pointing devices
Industrial instrumentation
Personal navigation devices
Hard disk drive (HDD) protection

GENERAL DESCRIPTION

The ADXL345 is a small, thin, ultralow power, 3-axis accelerometer with high resolution (13-bit) measurement at up to $\pm 16\text{ g}$. Digital output data is formatted as 16-bit twos complement and is accessible through either a SPI (3- or 4-wire) or I²C digital interface.

The ADXL345 is well suited for mobile device applications. It measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (3.9 mg/LSB) enables measurement of inclination changes less than 1.0° .

Several special sensing functions are provided. Activity and inactivity sensing detect the presence or lack of motion by comparing the acceleration on any axis with user-set thresholds. Tap sensing detects single and double taps in any direction. Free-fall sensing detects if the device is falling. These functions can be mapped individually to either of two interrupt output pins. An integrated, patent pending memory management system with a 32-level first in, first out (FIFO) buffer can be used to store data to minimize host processor activity and lower overall system power consumption.

Low power modes enable intelligent motion-based power management with threshold sensing and active acceleration measurement at extremely low power dissipation.

The ADXL345 is supplied in a small, thin, 3 mm $\times$ 5 mm $\times$ 1 mm, 14-lead, plastic package.

FUNCTIONAL BLOCK DIAGRAM

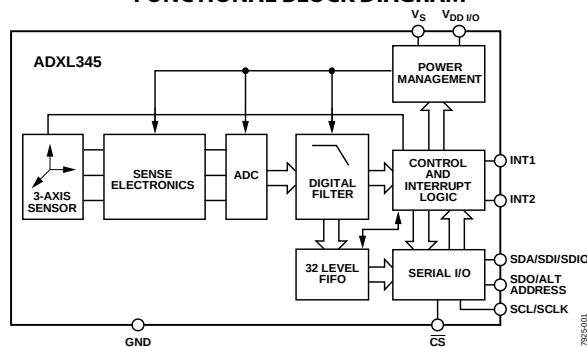


Figure 1.

Rev. D

Document Feedback

Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use, nor for any infringements of patents or other rights of third parties that may result from its use. Specifications subject to change without notice. No license is granted by implication or otherwise under any patent or patent rights of Analog Devices. Trademarks and registered trademarks are the property of their respective owners.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 ©2009–2013 Analog Devices, Inc. All rights reserved.
[Technical Support](#) www.analog.com

B.1.6. Giroszkóp



L3G4200D

MEMS motion sensor: ultra-stable three-axis digital output gyroscope

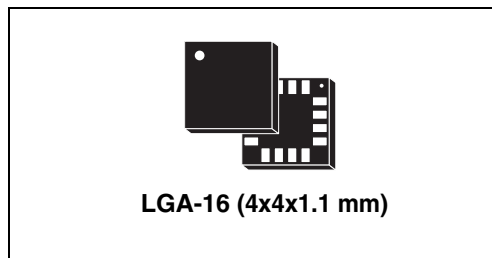
Preliminary data

Features

- Three selectable full scales (250/500/2000 dps)
- I²C/SPI digital output interface
- 16 bit-rate value data output
- 8-bit temperature data output
- Two digital output lines (interrupt and data ready)
- Integrated low- and high-pass filters with user-selectable bandwidth
- Ultra-stable over temperature and time
- Wide supply voltage: 2.4 V to 3.6 V
- Low voltage-compatible IOs (1.8 V)
- Embedded power-down and sleep mode
- Embedded temperature sensor
- Embedded FIFO
- High shock survivability
- Extended operating temperature range (-40 °C to +85 °C)
- ECOPACK[®] RoHS and "Green" compliant

Applications

- Gaming and virtual reality input devices
- Motion control with MMI (man-machine interface)
- GPS navigation systems
- Appliances and robotics



Description

The L3G4200D is a low-power three-axis angular rate sensor able to provide unprecedented stability of zero rate level and sensitivity over temperature and time. It includes a sensing element and an IC interface capable of providing the measured angular rate to the external world through a digital interface (I²C/SPI).

The sensing element is manufactured using a dedicated micro-machining process developed by STMicroelectronics to produce inertial sensors and actuators on silicon wafers.

The IC interface is manufactured using a CMOS process that allows a high level of integration to design a dedicated circuit which is trimmed to better match the sensing element characteristics.

The L3G4200D has a full scale of $\pm 250/\pm 500/\pm 2000$ dps and is capable of measuring rates with a user-selectable bandwidth.

The L3G4200D is available in a plastic land grid array (LGA) package and can operate within a temperature range of -40 °C to +85 °C.

Table 1. Device summary

Order code	Temperature range (°C)	Package	Packing
L3G4200D	-40 to +85	LGA-16 (4x4x1.1 mm)	Tray
L3G4200DTR	-40 to +85	LGA-16 (4x4x1.1 mm)	Tape and reel

B.2. Arduino forráskód

```
#include <Wire.h>

#define Register_ID 0
#define Register_2D 0x2D
#define Register_X0 0x32
#define Register_X1 0x33
#define Register_Y0 0x34
#define Register_Y1 0x35
#define Register_Z0 0x36
#define Register_Z1 0x37

int ADXAddress = 0xA7 >> 1;
int reading = 0;
int val=0;
int X0,X1,X_out;
int Y0,Y1,Y_out;
int Z1,Z0,Z_out;
double Xg,Yg,Zg;
double pTerm,iTerm,dTerm;
double Kp,Ki,Kd;
double integrated_error;
int GUARD_GAIN=10;
int last_error;
double K;

const int motor1Pin=7;
const int motor2Pin=8;
const int motorPWM1=9;
double PID;

void setup()
{
    Wire.begin();
    Serial.begin(9600);
```

```
pinMode ( motor1Pin ,OUTPUT);
pinMode ( motor2Pin ,OUTPUT);
pinMode (motorPWM1 ,OUTPUT);

digitalWrite ( motor1Pin ,LOW);
digitalWrite ( motor2Pin ,LOW);

Wire . beginTransmission ( ADXAddress );
Wire . write ( Register_2D );
Wire . write ( 8 );
Wire . endTransmission ( );
K=1;
Kp=1.3;
Ki=1.4;
Kd=1.2;
}
void loop ()
{

Wire . beginTransmission ( ADXAddress );
Wire . write ( Register_X0 );
Wire . write ( Register_X1 );
Wire . endTransmission ( );
Wire . requestFrom ( ADXAddress , 2 );
if ( Wire . available () <= 2 )
{
    X0 = Wire . read ( );
    X1 = Wire . read ( );
    X1=X1<<8;
    X_out=X0+X1;
}

Wire . beginTransmission ( ADXAddress );
Wire . write ( Register_Y0 );
Wire . write ( Register_Y1 );
Wire . endTransmission ( );
Wire . requestFrom ( ADXAddress , 2 );
```

```
if(Wire.available()<=2)
{
    Y0 = Wire.read();
    Y1 = Wire.read();
    Y1=Y1<<8;
    Y_out=Y0+Y1;
}
Wire.beginTransmission(ADXAddress);
Wire.write(Register_Z0);
Wire.write(Register_Z1);
Wire.endTransmission();
Wire.requestFrom(ADXAddress,2);
if(Wire.available()<=2)
{
    Z0 = Wire.read();
    Z1 = Wire.read();
    Z1=Z1<<8;
    Z_out=Z0+Z1;
}
Xg=X_out/256.0;
Yg=Y_out/256.0;
Zg=Z_out/256.0;
Serial.print("X= ");
Serial.print(Xg);
Serial.print("          PID ertek:");
PID=updatePid(0,Xg*100);
Serial.println(PID);
analogWrite(motorPWM1,abs(PID));
    if (PID >= 0)
    {
        digitalWrite(motor1Pin, HIGH);
        digitalWrite(motor2Pin, LOW);
    }
else
{
    digitalWrite(motor1Pin, LOW);
    digitalWrite(motor2Pin, HIGH);
}
```

```
    delay(100);

}

int updatePid(int targetPosition, int currentPosition)
{
    int error = targetPosition - currentPosition;
    pTerm = Kp * error;
    integrated_error += error;
    iTerm = Ki * constrain(integrated_error, -GUARD_GAIN, GUARD_GAIN);
    dTerm = Kd * (error - last_error);
    last_error = error;
    return -constrain(K*(pTerm + iTerm + dTerm), -255, 255);
}
```