



M Ű E G Y E T E M 1 7 8 2

BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI
EGYETEM

Gépészmérnöki Kar

Környezeti merevséghez és kommunikációs
időkésítéshez alkalmazkodó szabályozás tervezése
erő-visszacsatolt telemanipulációhoz

Kuti József

Mechatronikai mérnök MSc. hallgató, 2. évf.

Konzulens: Galambos Péter

tudományos segédmunkatárs: BME MANUF, MTA SZTAKI

Budapest, 2012.

Támogatás

A kutatást a Nemzeti Fejlesztési Ügynökség támogatta az (ERC-HU-09-1-2009-0004 MTASZTAK) (OMFB-01677/2009) projekt keretében.

A dolgozat bemutatását a TÁMOP - 4.2.2.B-10/1-2010-0009 "Új tehetséggondozó programok és kutatások a Műegyetem tudományos műhelyeiben" című program támogatta.

Nyilatkozat

Alulírott Kuti József a Budapesti Műszaki és Gazdaságtudományi Egyetem hallgatója kijelentem, hogy ezt a dolgozatot meg nem engedett segítség nélkül, saját magam készítettem, és csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen a forrás megadásával megjelöltem.

Kuti József

Tartalomjegyzék

Ábrák jegyzéke	VII
Tartalmi összefoglaló	IX
Abstract	X
Jelölések és rövidítések jegyzéke	XI
1. Bevezető	1
2. A telemanipuláció rövid bemutatása	2
2.1 Impedancia modell alapú irányítás	2
2.1.1 Impedancia irányítás a robotikában	2
2.1.2 Alkalmazása a telemanipulációban	3
3. TP és TP^τ modell transzformáció	4
3.1 Tenzoralgebrai áttekintés	4
3.1.1 Magasabb rendű szinguláris értékek szerinti felbontás (HOSVD)	7
3.2 A TP függvény HOSVD alapú kanonikus alakja	8
3.3 TP függvény kanonikus alakjának numerikus rekonstrukciója TP modell transz- formációval	9
3.3.1 I. lépés: diszkretizáció	10
3.3.2 II. lépés: a diszkretizált függvényhez TP struktúra előállítása	10
3.3.3 III. lépés: a folytonos súlyfüggvények rekonstrukciója	11
3.3.4 A kanonikus alak approximációs alkalmazásai	12
3.4 LPV, qLPV rendszerek politopikus reprezentációja	13
3.5 Konvex burok manipuláció	14
3.6 TP^τ modell transzformáció	15
3.6.1 Időkéséses rendszer újradefiniálása	15
3.6.2 TP^τ transzformáció lépései	16
I. lépés: diszkretizáció újradefiniálással	16
II. lépés: a diszkretizált tenzor kibontása TP struktúrába	16

III. lépés: folytonos súlyfüggvények rekonstrukciója	17
3.7 Összefoglalás	18
4. Merevségbecslő alkalmazások	19
4.1 Feladatspecifikáció	19
4.2 Nem-parametrikus merevségmegfigyelő	20
4.3 Parametrikus merevségmegfigyelő környezettel érintkező robotrendszerekhez . .	21
4.4 Parametrikus merevségmegfigyelő VSA eszközökhöz	22
4.5 Összefoglalás	23
5. Az irányítási rendszer konvex politopikus reprezentációja	24
5.1 Modellalkotás	24
5.2 Politopikus reprezentáció előállítása TP^r transzformációval	26
5.2.1 Újradefiniálás	26
5.2.2 Paramétertér és rácsháló	28
5.2.3 Diszkretizáció	29
5.2.4 Kanonikus súlyfüggvények	29
5.2.5 Komplexitáscsökkentés rangcsökkentéssel	30
5.2.6 Konvex burok	31
5.3 Validáció	31
5.3.1 Állandó paraméterezés vizsgálata	31
5.3.2 Változó paraméterezés vizsgálata	32
5.4 Összefoglalás	34
6. Szabályozótervezés	35
6.1 A szabályozási struktúra	35
6.2 LMI alapú szabályozótervezés	36
6.2.1 A konvex TP alak és a Takagi-Sugeno fuzzy leírás kapcsolata	36
6.2.2 Szabályozótervezés lineáris mátrix egyenlőtlenségekkel	37
6.2.3 Szabályozótervezés	38
6.3 A szabályozott rendszer vizsgálata numerikus szimulációkkal	39
6.3.1 Konstans időkézés és merevség	39
6.3.2 Változó időkézés és merevség	41
6.4 Merevségbecslő alkalmazása a szabályozókörben	41
6.4.1 Nem-parametrikus merevségmegfigyelő alkalmazása	44
6.4.2 Parametrikus merevségmegfigyelő alkalmazása	44
6.5 Összefoglalás	47

7. Kísérleti vizsgálat	48
7.1 A környezet bemutatása	48
7.1.1 Vezérlőprogram	48
TP szabályozó modul	49
Matematikai modul	50
Főprogram	50
7.2 A rendszer szűrési feladatai	51
7.2.1 A késleltetett pozícióérték szűrése – mozgóátlag szűrő	51
7.2.2 A mester oldali erő szűrése – mozgóátlag szűrő	52
7.2.3 A szolga oldali erő szűrése – Kalman-szűrő	52
7.3 Eredmények	53
7.4 Összefoglalás	54
8. Összefoglalás	62
Irodalomjegyzék	64
Függelék	67
F.1 A TP^r modell transzformáció MATLAB programmal	67
F.2 Konvex politopikus reprezentációk	71
F.2.1 Az "A" modell CNO konvex burka	71
F.2.2 A "B" modell CNO konvex burka	72
F.2.3 A "C" modell CNO konvex burka	73
F.2.4 A "D" modell CNO konvex burka	74
F.2.5 Az "E" modell CNO konvex burka	75
F.2.6 Az "F" modell CNO konvex burka	76
F.3 LMI alapú szabályozótervezés	78
F.4 A merevségbecslővel kiegészített szabályozás numerikus szimulációja	81
F.5 A kísérleti környezet vezérlőprogramja	91

Ábrák jegyzéke

2.1. Az impedancia modell használata robotirányítás (a.,) és telemanipulációs (b.,) célra	3
3.1. Háromdimenziós tenzor 1-, 2- és 3-módú mátrixának képzése	6
3.2. a., a TP alak használata egy rácsponti érték meghatározásához b., a súlyfüggvény értékének meghatározása a rácspontok között	11
4.1. A merevségmegfigyelő elvi felépítése	22
5.1. Telemanipulációs irányítási struktúra közös impedanciaszabályozással	25
5.2. Az elhanyagolások után kapott modell	25
5.3. Az újradefiniáláshoz használt modell	27
5.4. A rendszerparaméterek a merevség és az időkésés függvényében	28
5.5. Az átparaméterezés hatása a transzformációs tér és a rácsháló alakjára	29
5.6. A szinguláris értékek a merevséghez és a relatív időkéséshez	30
5.7. γ_{max} különböző komplexitású rendszerekhez	30
5.8. A "C" CNO burok súlyfüggvényei	31
5.9. A paraméterezés és a core tenzor méretének hatása a sebességválaszra	32
5.10. A különböző modellek sebességhibájának négyzetes integrálja különböző paraméterezések esetén	33
5.11. Időtől függő merevségprofil alkalmazása	33
5.12. Pozíciófüggő merevségprofil alkalmazása	34
6.1. Az új adaptív állapotvisszacsatolási struktúra	36
6.2. A vizsgálatához használt operátorerő-profil	39
6.3. A szabályozott és a szabályozás nélküli rendszer összehasonlítása, különböző merevség és időkésés esetén	40
6.4. Változó merevség és időkésés hatása - a vizsgált paramétertartományban	42
6.5. Változó merevség és időkésés hatása - az instabil tartományban	43
6.6. Nem-parametrikus merevségmegfigyelő alkalmazása, $\alpha = 1500$	45
6.7. Parametrikus merevségmegfigyelő alkalmazása	46

7.1. A kísérleti elrendezés	49
7.2. A kísérleti környezetben megvalósított irányítási rendszer vázlata	51
7.3. A változó időkézés hatása a pozíciójelre	52
7.4. 2140 [N/m] merevségű rugó megfogása kézzel mozgatott mestereszköz esetén . .	55
7.5. 2140 [N/m] merevségű rugó megfogása TP szabályozó működése esetén	56
7.6. 2140 [N/m] merevségű rugó megfogása TP szabályozó nélkül	57
7.7. 6800 [N/m] merevségű rugó megfogása TP szabályozó működése esetén	58
7.8. 6800 [N/m] merevségű rugó megfogása TP szabályozó nélkül	59
7.9. 6300 [N/m] merevségű rugó megfogása TP szabályozó működése esetén	60
7.10. 6300 [N/m] merevségű rugó megfogása TP szabályozó nélkül	61
F.1. Az "A" modell CNO burokának súlyfüggvényei	71
F.2. A "B" modell CNO burokának súlyfüggvényei	72
F.3. A "C" modell CNO burokának súlyfüggvényei	73
F.4. A "D" modell CNO burokának súlyfüggvényei	74
F.5. Az "E" modell CNO burokának súlyfüggvényei	75
F.6. Az "F" modell CNO burokának súlyfüggvényei	76
F.7. A szimulációk Simulink blokk vázlata	81

Tartalmi összefoglaló

Az ember-robot interakció biztonságos megvalósítása és az erő-visszacsatolt telemanipuláció régóta aktívan kutatott területek. A BME Gyártástudomány és –technológia Tanszékén a RESCUER projekt (2004-2007) apropóján indultak el azon ma is folyó, telemanipulációval kapcsolatos kutatások, amelyek témájához ezen dolgozat is illeszkedik.

A dolgozatomban vizsgált irányítási struktúra az impedancia/admittancia szabályozási elvre épül, amely napjaink egyik kulcs technológiája a fizikai robot interakció és a telemanipuláció területén. A struktúra lényege, hogy egy virtuális tömeg-csillapítás mechanikai modell (impedancia modell) mozgását valós időben számítjuk a robot végperemén mért erő és nyomaték alapján. Az így számított pozíciót használjuk fel a manipulátor pozíciószabályozásának referenciájaként. A stabil és transzparens Internet alapú telemanipuláció legnagyobb akadályja az időkéésés, ami a mester és szolga oldal közötti csomagkapcsolt kommunikációból származik. A dolgozatban tárgyalt szabályozási módszer az időkéésés negatív hatásának kiküszöbölésére, azaz a stabil és transzparens működés fenntartására irányul.

Munkám elméleti háttérét a TP^r modell transzformáció alapú szabályozótervezési módszertan képezi, amely a tenzor szorzat (TP) modell transzformáció kiterjesztésével lehetővé teszi a korszerű qLPV és LMI alapú tervezési eljárások alkalmazását időkééséses rendszerekhez.

Munkám elsődleges célja, a konzulensem által korábban állandó merevségű esetre javasolt tervezési eljárás kibővítése olyan módon, hogy az időkéésés mellett a környezet merevségéhez is alkalmazkodóvá váljon.

Dolgozatomban ismertetem a kutatómunkám során kifejlesztett új modellezési és tervezési eljárást, amely lehetővé teszi az időkéésésre és a távoli környezet merevségére nézve is adaptív szabályozó tervezését. A javasolt eljárás további előnye, hogy az állapottér modellben mérhető állapotváltozókat alkalmaz, így a korábbi struktúrával ellentétben nincs szükség állapotmegfigyelőre. Az adaptív viselkedéshez szükséges a merevség aktuális értékének ismerete, ezért online merevségmegfigyelővel egészítettem ki a szabályozót.

A javasolt szabályozási struktúrát és egyúttal a tervezési eljárás egészét numerikus szimulációkkal és valós robotokon végzett kísérletekkel is vizsgáltam. Az elvégzett átfogó validáció igazolja az új módszer hatékonyságát.

Abstract

Safe human-robot interaction and telemanipulation with haptic feedback are under very active investigation since the past decades. At the Department of Manufacturing Sciences and Technology telemanipulation-related research has been running since the RESCUE project (2004-2007). This work also fits in this branch of research.

The control scheme investigated in this essay is based on the principle of impedance/admittance control, which is considered as one key technology in the field of interaction robotics and telemanipulation. The substance of this structure is that the motion of a mass-damper system is computed in real-time based on the force/torque measured at the robot's end-effector. The so computed position is utilised as the reference input of the position controller of the robot. Communication time delay between the master and the slave plants renders the most serious problem in stable and transparent telemanipulation. The control method discussed within this essay aims to eliminate the negative effect of time delay and maintain the stable and transparent operation.

The TP^τ model transformation based control design methodology yields the theoretical background of my work. It enables to use the modern qLPV and LMI based design methods with the extension of tensor product (TP) model transformation to time delay systems.

The primary goal of my work is to extend the design method formerly proposed by my tutor for constant stiffness case in such a way that it becomes adaptive to variable environmental stiffness as well as to the varying communication delay.

In my essay, I introduce the proposed modelling and design method that allows the delay- and stiffness-adaptive control design as a result of my research. A further benefit of this method is that since the applied state space is composed by measurable state variables, the state observer became unnecessary compared to the former structure. The environmental stiffness must be monitored to enable the adaptive behaviour; therefore, I extended the structure with an on-line stiffness estimator.

I investigated the proposed control scheme and the design method via numerical simulations and experiments on real robots. This comprehensive validation proves the efficiency of the novel method.

Jelölések és rövidítések jegyzéke

A dolgozat előtt szükségesnek tartok egy rövid leírást a felhasznált jelölésekről és rövidítésekről. Más betűtípussal fogom jelölni a különböző típusú mennyiségeket. A skalár változókat dőlt betűkkel ($a, b \dots$), a vektorokat félkövér, álló kisbetűk ($\mathbf{a}, \mathbf{b} \dots$), a mátrixokat félkövér, álló nagybetűkkel ($\mathbf{A}, \mathbf{B} \dots$) jelölöm, míg a tenzorokat kalligráfiai karakterekkel ($\mathcal{A}, \mathcal{B} \dots$).

Jelölések

$F_h(t)$	- az operátor által kifejtett erő $[N]$
$F_e(t)$	- a környezet által kifejtett erő $[N]$
$x(t)$	- az impedancia modell pozíciója $[m]$
k	- a környezet merevsége $[N/m]$
m	- tömeg $[kg]$
b	- viszkózus csillapítás $[Ns/m]$
τ	- időkézés
ϑ	- relatív időkézés
$\mathbf{A}, \mathbf{B}$	- az állapottér modell rendszermátrixai
$\mathbf{x}(t)$	- az állapottér modellben az állapotvektor
Ω	- a TP modell transzformációs tere
M	- a TP modell diszkretizációs rácsa
$\mathbf{w}_n(p_n)$	- a TP modell n -dik súlyfüggvényvektora

Rövidítések

LPV – Linear Parameter Varying - lineáris paraméter változós

qLPV – quasi Linear Parameter Varying - kvázi lineáris paraméter változós

LTI – Linear Time Invariant - időfüggetlen lineáris

LMI – Linear Matrix Inequality - lineáris mátrix egyenlőtlenség

SVD – Singular Value Decomposition - szinguláris értékek szerinti felbontás

HOSVD – Higher-Order Singular Value Decomposition - magasabb rendű szinguláris értékek szerinti felbontás

CHOSVD – Compact Higher-Order Singular Value Decomposition - kompakt magasabb rendű szinguláris értékek szerinti felbontás

RHOSVD – Reduced Higher-Order Singular Value Decomposition - redukált magasabb rendű szinguláris értékek szerinti felbontás

TP modell transzformáció – Tensor Product model transformation - tenzor szorzat modell transzformáció

TP^r modell transzformáció – Tensor Product model transformation for Systems with Feedback delay - tenzor szorzat modell transzformáció késleltetett rendszerekhez

TS fuzzy rendszer – Takagi-Sugeno fuzzy rendszer

SN – Sum Normalized - összegre normalizált

NN – Non-Negative - nem negatív

NO – NOrmal - normál

CNO – Close to NOrmal - közelítőleg normál

RLS – Recursive Least Squares - rekurzív legkisebb négyzetek

VSA – Variable Stiffness Actuator - változtatható merevségű aktuátor

1. fejezet

Bevezető

Az erővisszacsatolt telemanipuláció szabályozási kérdései régóta intenzív kutatás tárgyát képezik. A BME Gyártástudomány és -technológia Tanszékén a RESCUER projekt keretében indult el egy erővisszacsatolt megfogó fejlesztése, amely lehetővé teszi a robotkar kezelője számára a biztonságos, stabil és transzparens műveletvégzést. [1] [2]

A vizsgálatok során kiderült, hogy a rendszer időkésése miatt (kommunikációs, mintavételezési idő) kemény anyagok megfogása nehézségekbe ütközik. A folyamat instabillá válhat és ekkor a megfogó elpattan a megfogandó tárgytól. A probléma további kutatásokat motivált a projekt lezáródása után is.

Konzulensem doktori disszertációjában [3] adott merevségű környezetbe adaptív stabilizáló szabályozás tervezését mutatta be, amely az időkésés nagyságától függően avatkozik be. A tervezési módszertan, a TP^τ modell transzformáció alapú szabályozástervezés is ennek a kutatásnak az eredménye.

A dolgozatomban ebben az irányban folytattam a kutatást. Az instabilitás kialakulásában az időkésés mellett a környezet merevsége is jelentős szerepet játszik. A célom olyan adaptív szabályozó tervezése volt, amely az időkésés mellett figyelembe veszi a környezet merevségét.

A dolgozat első részében bemutatom azokat az elméleti alapokat, amelyekre a munkám épül. Ismertetem a telemanipuláció szerepét a robotikában és az alkalmazott közös impedancia irányítási struktúrát. Ezután részletesen bemutatom a TP^τ modell transzformáció elméleti alapjait, amelyre a szabályozótervezés épül.

A merevségre adaptív szabályozó működéséhez szükség van a merevség online becslésére. Ezért külön fejezetben ismertetem az irodalomban fellelhető merevségmegfigyelő módszereket.

A dolgozat második része három fejezetre tagolódik. Az első fejezetben előállítom az időkéséses rendszer időkésésre és merevségre adaptív TP típusú politopikus modelljét a TP^τ modell transzformációval. Ezután a modellhez szabályozót tervezek, majd szimulációkkal vizsgálom meg a szabályozó tulajdonságait és a merevségbecslők alkalmazhatóságát. Végül kísérleti környezetbe implementálom a megtervezett és megfelelő szűrőkkel kiegészített irányítási struktúrát és bemutatom a kísérletek eredményeit.

2. fejezet

A telemanipuláció rövid bemutatása

A telemanipuláció, azaz a távoli beavatkozás alkalmazásának legfőbb motivációi azok a feladatok, amelyek közvetlen jelenléttel nem, vagy csak nehezen megoldhatóak, de végrehajtásukhoz szükséges az emberi intelligencia bevonása. Ezek lehetnek veszélyes környezetben (űrben, szennyezett területen, vagy akár harctéren) [4] megoldandó feladatok, vagy olyan helyzetek, melyekben nem is léphetünk közvetlen interakcióba a környezettel (kis mérettartománybeli műveletek, kulcslyuksebészet). De a terület felhasználható, mint kognitív infokommunikációs forma akár a virtuális valósággal is. [5]

A szabályozási követelmények teljesítésének legnagyobb akadálya a kommunikációs csatorna időkésése és a csomagvesztés. A problémák megoldásához különböző irányítási struktúrákat és a szabályozási módszereket fejlesztettek ki [4, 6]. Ebben a fejezetben a későbbiekben használt impedancia modell alapú irányítási módszert mutatom be.

2.1. Impedancia modell alapú irányítás

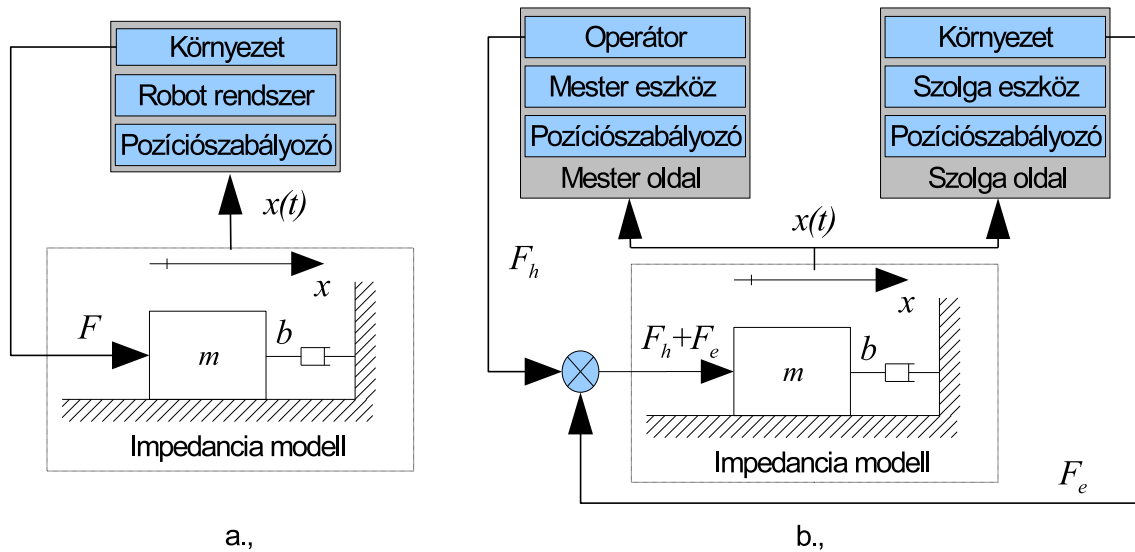
Az utóbbi években az impedancia irányítás fontos szerepet kapott a robotika egyes területein, mint a biztonságos humán-robot interakció, telemanipuláció vagy sebészeti robotika. Hogan átfogó kutatása [7–9], melyben az impedancia irányítás alapjait és alkalmazását kidolgozta, tette ezt a módszert a modern robot irányítás egyik kulcs technológiájává. Az impedancia irányítást a robotika több területén használják, például a telerobotikában, vagy flexibilis csuklójú robotok irányításában.

Robot rendszerekben sok esetben elkerülhetetlen az időkésés, ami nemkívánt hatásokat okozhat. Számos kutatás vizsgálja az impedancia modellek időkésés érzékenységét. [10–12]

2.1.1. Impedancia irányítás a robotikában

Az irányítás során egy virtuális tömeg-csillapítás modell mozgását szimuláljuk, a modell bemenete egy külső erő, kimenete a tömeg pozíciója vagy sebessége. Robotirányítás esetén

a robot végperemére épített erőmérő cella adja a bemenetet, az impedanciamodellel pozícióját pedig a robot pozíciószabályozása kapja alapjelnek.



2.1. ábra. Az impedancia modell használata robotirányítás (a.) és telemanipulációs (b.) célra

Az irányítási struktúra egyszerű vázlatát az 2.1.a ábrán látható. Lehetővé teszi az ember-robot interakció biztonságos megvalósítását, mivel a robotrendszerre a pozíciószabályozás a virtuális tömeg-csillapítás modell mozgását kényszeríti rá.

2.1.2. Alkalmazása a telemanipulációban

A módszer elosztott robotrendszerek, telemanipulációs rendszerek irányítására is használható, a mester és szolga oldalhoz közös impedancia modellt alkalmazva az 2.1.b ábrán látható módon. Az alkalmazások jelentős részében azonban nagy mértékű időkéésés keletkezhet a kommunikációs csatónában. Az időkéésés a rendszer stabilitásvesztését is okozhatja, a kritikus időkéésés a környezeti merevségtől, a mintavételi időtől és az impedancia modell paramétereitől is függ. [2]

A dolgozatban ezt a struktúrát használom fel, és az időkéésés hatásainak kompenzálására időkéésésre és környezeti merevségre adaptív szabályozót tervezek.

3. fejezet

TP és TP^τ modell transzformáció

A TP modell transzformáció egy numerikus eljárás, mellyel folytonos skalár-, vektor-, mátrix- vagy akár tenzorfüggvényeket egységesen TP (tenzor szorzat) típusú politopikus alakra transzformálhatunk. A módszertan lehetőséget biztosít konvex politopikus modell előállítására is.

A lineáris és kvázi lineáris paraméterváltozójú (qLPV) rendszerek rendszermátrixai mátrixfüggvénynek tekinthetők, ezáltal a módszer lehetőséget ad ezek konvex politopikus alakú approximációjára is. Ez az alak közvetlenül felhasználható a korszerű LMI alapú szabályozótervezési eljárásokhoz.

A TP^τ modell transzformáció a TP modell transzformáció kiterjesztése, hogy időkésést tartalmazó rendszerekhez is előállítható legyen a konvex politopikus alak.

A fejezetben először a tárgyaláshoz szükséges mátrix- és tenzoralgebrai ismereteket tekintem át röviden, Lathauwer [13] munkája alapján. Majd bemutatom a TP modell transzformációt, Baranyi P. munkái [14–18] alapján. Az érthetőség kedvéért az eljárást először skalárfüggvényekhez ismertetem, majd külön bemutatom a kiterjesztését mátrixokra, ami lehetővé teszi LPV, qLPV rendszerek kezelését.

Végül konzulensem doktori munkája alapján [3] a TP^τ modell transzformációt ismertetem, amellyel a transzformáció elvégezhető időkésést tartalmazó rendszerekhez is.

3.1. Tenzoralgebrai áttekintés

Először is áttekintem a téma tárgyalásához szükséges definíciókat a mátrixalgebrából és azok kiterjesztését tenzorokra.

3.1. Tétel. *(Mátrixok szinguláris értékek szerinti felbontása (SVD).) Minden valós $(I_1 \times I_2)$ méretű $\mathbf{A}$ mátrix felírható*

$$\mathbf{A} = \mathbf{U}_1 \mathbf{S} \mathbf{U}_2^T \tag{3.1}$$

alakban, melyben

– $\mathbf{U}_1: I_1 \times I_1$, $\mathbf{U}_2: I_2 \times I_2$ négyzetes mátrixok,

– $\mathbf{S}: I_1 \times I_2$ valós mátrix, amely

– pszeudodiagonális:

$$\mathbf{S} = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_{\min(I_1, I_2)}), \quad (3.2)$$

– rendezett:

$$\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(I_1, I_2)} \geq 0. \quad (3.3)$$

3.1. Megjegyzés. Az $\mathbf{S}$ mátrix elemei (a szinguláris értékek) egyértelműen meghatározhatóak: azonosak $(\mathbf{A}^T \mathbf{A})$ mátrixszorzat sajátértékeinek pozitív négyzetgyökével. Ha minden szinguláris érték különbözik, $\mathbf{U}_1$ és $\mathbf{U}_2$ mátrixok egyértelműen meghatározhatóak, csak az előjelek változtathatóak meg. Azonos szinguláris értékekhez $(\sigma_n = \sigma_{n+1})$ tartozó szinguláris vektorok meghatározása nem egyértelmű (hasonlóan, mint azonos sajátértékekhez tartozó sajátvektoroknál).

3.1. Definíció. (Mátrix L_2 normája vagy spektrálnormája.) Egy mátrix L_2 normája alatt SVD felbontásuk legnagyobb szinguláris értékét (σ_1) értjük.

3.2. Megjegyzés. Egy mátrix L_2 normája értelmezhető legnagyobb vektornyújtásként.

3.2. Definíció. (Skaláris szorzat.) Két tenzor $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ skaláris szorzata alatt

$$\langle \mathcal{A}, \mathcal{B} \rangle \equiv \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \dots \sum_{i_N=1}^{I_N} a_{i_1, i_2, \dots, i_N} b_{i_1, i_2, \dots, i_N} \quad (3.4)$$

szorzatösszeget értjük.

3.3. Definíció. (Ortogonalis tulajdonság.) Az $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ tenzorokat ortogonalisnak nevezzük, ha skaláris szorzatuk zérus.

3.4. Definíció. (Frobenius-norma.) Az $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ tenzor Frobenius normája alatt a következő kifejezést értjük:

$$\|\mathcal{A}\|_F \equiv \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}. \quad (3.5)$$

3.3. Megjegyzés. A Frobenius-norma és a szinguláris értékek közötti összefüggés:

$$\|\mathcal{A}\|_F = \sqrt{\sum_{i=1}^{\min(I_1, I_2)} \sigma_i^2}.$$

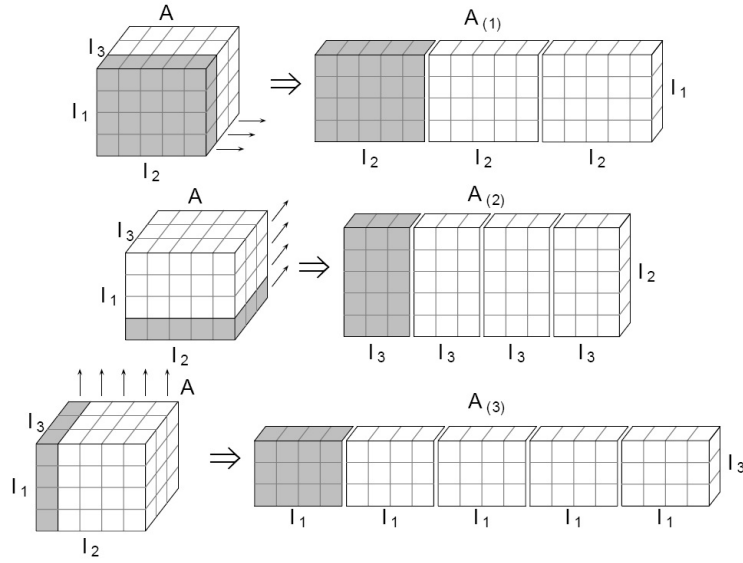
Tenzorokra a fent leírt tulajdonságok a tenzor n-módú mátrixain keresztül terjeszthetők ki.

3.5. Definíció. (Tenzor n -módú mátrixa.) Adott $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ tenzor. Ezen tenzor n -módú mátrixa $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n \times (I_{n+1} I_{n+2} \dots I_N I_1 I_2 \dots I_{n-1})}$, alatt azt a mátrixot értjük, melyben a tenzor $(a_{i_1, i_2, \dots, i_N})$ eleme, a mátrix i_n -dik sorában és

$$(i_{n+1} - 1)I_{n+2}I_{n+3} \dots I_N I_1 I_2 \dots I_{n-1} + (i_{n+2} - 1)I_{n+3}I_{n+4} \dots I_N I_1 I_2 \dots I_{n-1} + \dots + (i_N - 1)I_1 I_2 \dots I_{n-1} + (i_1 - 1)I_2 I_3 \dots I_{n-1} + (i_2 - 1)I_3 I_4 \dots I_{n-1} + \dots + (i_{n-1} - 1)$$

-dik oszlopában helyezkedik el.

A (3.1) ábrán látható egy háromdimenziós tenzor 1-, 2- és 3-módú felbontása.



3.1. ábra. Háromdimenziós tenzor 1-, 2- és 3-módú mátrixának képzése

A tenzor rangját a tenzor n -módú mátrixán keresztül értelmezhetjük, az alábbi definícióval.

3.6. Definíció. (Tenzor n -módú rangja.) Az $\mathcal{A}$ tenzor n -módú rangja alatt az $\mathcal{A}$ tenzor n -módú mátrixának rangját értjük. Jelölése $R_n = \text{rank}_n(\mathcal{A})$.

3.7. Definíció. (Tenzor n -módú szorzata mátrixszal.) Az $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_n \times \dots \times I_N}$ tenzor és $\mathbf{U} \in \mathbb{R}^{J_n \times I_n}$ mátrix n -módú szorzata $(\mathcal{A} \times_n \mathbf{U})$ alatt azt a $I_1 \times I_2 \times \dots \times I_{n-1} \times J_n \times I_{n+1} \times \dots \times I_N$ méretű tenzort értünk, melyre

$$(\mathcal{A} \times_n \mathbf{U})_{i_1, i_2, \dots, i_{n-1}, j_n, i_{n+1}, \dots, i_N} = \sum_{i_n} a_{i_1, i_2, \dots, i_{n-1}, i_n, i_{n+1}, \dots, i_N} u_{j_n, i_n}.$$

Az olyan többszörös tenzorszorzatot, mint $\mathcal{A} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \dots \times_N \mathbf{U}_N$, a továbbiakban a következő rövid alakkal fogjuk jelölni:

$$\mathcal{A} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \dots \times_N \mathbf{U}_N = \mathcal{A} \boxtimes_{n=1}^N \mathbf{U}_n.$$

A definíciókból a következő fontos tulajdonságok adódnak.

3.1. Tulajdonság. Adott $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ tenzor és $\mathbf{F} \in \mathbb{R}^{J_n \times I_n}$, $\mathbf{G} \in \mathbb{R}^{J_m \times I_m}$ mátrixokra, ha $n \neq m$

$$(\mathcal{A} \times_n \mathbf{F}) \times_m \mathbf{G} = (\mathcal{A} \times_m \mathbf{G}) \times_n \mathbf{F} = \mathcal{A} \times_m \mathbf{G} \times_n \mathbf{F}.$$

3.2. Tulajdonság. Adott $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ tenzor és $\mathbf{F} \in \mathbb{R}^{J_n \times I_n}$, $\mathbf{G} \in \mathbb{R}^{K_n \times J_n}$ mátrixokra

$$(\mathcal{A} \times_n \mathbf{F}) \times_n \mathbf{G} = \mathcal{A} \times_n (\mathbf{F}\mathbf{G}).$$

3.1.1. Magasabb rendű szinguláris értékek szerinti felbontás (HOSVD)

Az alfejezetben a szinguláris értékek szerinti felbontás tenzorokra kidolgozott, magasabb rendű általánosítását mutatom be.

3.2. Tétel. (Magasabb rendű szinguláris értékek szerinti felbontás - HOSVD.) Minden valós $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ tenzor felírható

$$\mathcal{A} = \mathcal{S} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \dots \times_N \mathbf{U}_N = \mathcal{S} \boxtimes_{n=1}^N \mathbf{U}_n$$

tenzorszorzatként, melyben

- $\mathbf{U}_n = (\mathbf{u}_{n,1}, \mathbf{u}_{n,2}, \dots, \mathbf{u}_{n,I_n})$, $I_n \times I_n$ négyzetes mátrixok, ahol $n = 1..N$.
- $\mathcal{S}: I_1 \times I_2 \times \dots \times I_N$ valós mátrix, amely $\mathcal{S}_{i_n=\alpha}$ altenzoraira igazak a következő tulajdonságok
 - ortogonalitás: $\mathcal{S}_{i_n=\alpha}$ és $\mathcal{S}_{i_n=\beta}$ altenzorokra minden lehetséges n , α és β értékekre, ha $\alpha \neq \beta$: $\langle \mathcal{S}_{i_n=\alpha}, \mathcal{S}_{i_n=\beta} \rangle = 0$.
 - rendezettség: minden lehetséges n értékre igaz, hogy

$$\|\mathcal{S}_{i_n=1}\| \geq \|\mathcal{S}_{i_n=2}\| \geq \dots \geq \|\mathcal{S}_{i_n=I_n}\| \geq 0.$$

$\mathcal{S}$ tenzor n -dik dimenzió szerinti i -dik altenzorának Frobenius normáját ($\|\mathcal{S}_{i_n=i}\|$) az $\mathcal{S}$ tenzor i -dik n -módú szinguláris értékének nevezzük és röviden $\sigma_i^{(n)} = \|\mathcal{S}_{i_n=i}\|$ módon jelöljük.

3.4. Megjegyzés. A $\mathcal{S}$ tenzor meghatározása egyértelmű. Ha a szinguláris értékek különbözőek, $\mathbf{U}_n$ mátrixok egyértelműen meghatározhatóak, csak az előjelek változtathatóak meg. Azonos $\sigma_i^{(n)} = \sigma_{i+1}^{(n)}$ szinguláris értékek esetén $\mathbf{u}_{n,i}$ és $\mathbf{u}_{n,i+1}$ szinguláris vektorok meghatározása nem egyértelmű.

3.8. Definíció. (Compact HOSVD (CHOSVD)). A HOSVD felbontás minimalizált formáját kapjuk, ha csak a nem-nulla szinguláris értékeket és csak a hozzájuk tartozó szinguláris vektorokat tartjuk meg. A kapott felbontás:

$$\mathcal{A} = \mathcal{D} \boxtimes_{n=1}^N \mathbf{U}_n,$$

ahol $\mathcal{D}: R_1 \times R_2 \times \dots \times R_N$ méretű tenzor, $\mathbf{U}_n: I_n \times R_n$ méretű mátrixok és $R_n = \text{rank}_n(\mathcal{A})$, minden $n = 1..N$ esetén.

3.9. Definíció. (*Reduced HOSVD (RHOSVD)*). Ha a HOSVD felbontásból nem csak a zérus szinguláris értékeket (és a hozzájuk tartozó szinguláris vektorokat) távolítjuk el, a kapott felbontás hibával közelíti az eredeti tenzort, a következő tulajdonsággal.

3.3. Tulajdonság. Adott az $\mathcal{A}$ tenzor HOSVD felbontása a 3.2. tételben leírt módon, mint $\mathcal{A} = \mathcal{D} \boxtimes_{n=1}^N \mathbf{U}_n$, és $\mathcal{A}$ tenzor n -módú rangjait R_n -nel jelöljük, minden $n = 1..N$ esetén. Defináljuk $\hat{\mathcal{A}}$ tenzort, melyben $I'_n \leq R_n$ számú szinguláris értéket tartunk meg, azaz $\sigma_{I'_n+1}^{(n)}, \sigma_{I'_n+2}^{(n)}, \dots, \sigma_{R_n}^{(n)}$ szinguláris értékeket és a hozzájuk tartozó szinguláris vektorokat elhagyjuk minden $n = 1..N$ dimenziókban külön-külön. Ebben az esetben az approximációs hibára a következő összefüggés adódik:

$$\gamma = \|\mathcal{A} - \hat{\mathcal{A}}\| \leq \sqrt{\sum_{i_1=I'_1+1}^{R_1} (\sigma_{i_1}^{(1)})^2 + \sum_{i_2=I'_2+1}^{R_2} (\sigma_{i_2}^{(2)})^2 + \dots + \sum_{i_N=I'_N+1}^{R_N} (\sigma_{i_N}^{(N)})^2}. \quad (3.6)$$

Így lehetőségünk van az eredeti tenzor alacsony rangú becslésre minimális becslési hiba mellett. Az eredeti, esetleg nagy méretű tenzort komplexitását lehetőségünk van csökkenteni. A gyakorlatban a kisebb méretű modell tárolása, kezelése kisebb memóriaigényt, számítási kapacitást igényel.

3.2. A TP függvény HOSVD alapú kanonikus alakja

3.10. Definíció. (*Tenzor szorzat (TP) függvény*). Tekintsük a következő függvényt:

$$y = f(\mathbf{p}), \quad (3.7)$$

ahol y skalár és $\mathbf{p} \in \mathbb{R}^N$. A függvényt TP függvénynek nevezzük, ha megadható a következő formában

$$y = \mathcal{B} \boxtimes_{n=1}^N \mathbf{w}_n(p_n), \quad (3.8)$$

ahol

- $\mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ korlátos, konstans elemeket tartalmazó magtenzor,
- a $\mathbf{w}_n(p_n)$ ($n = 1..N$) súlyfüggvényvektorok egyenként I_n darab egyváltozós, folytonos, súlyfüggvényt tartalmaznak.

Ha elvégezzük a CHOSVD felbontást a TP függvény magtenzorán, a következő jelölésekkel:

$$\mathcal{B} = \mathcal{D} \boxtimes_{n=1}^N \mathbf{U}_n, \quad (3.9)$$

a függvény átírható

$$y = \mathcal{D} \boxtimes_{n=1}^N \tilde{\mathbf{w}}_n(p_n) \quad (3.10)$$

alakban, ahol $\tilde{\mathbf{w}}_n(p_n) = \mathbf{U}_n \mathbf{w}_n(p_n)$ (minden $n = 1..N$ esetén). Ez az alak a TP függvény HOSVD alapú kanonikus alakja.

3.3. Tétel. (TP függvények HOSVD alapú kanonikus alakja.) Tekintsük a következő TP függvényt:

$$y = f(\mathbf{p}) = \mathcal{B} \boxtimes_{n=1}^N \mathbf{w}_n(p_n). \quad (3.11)$$

Elvégezve 3.11 egyenlet $\mathcal{B}$ magtenzorán a CHOSVD eljárást, meghatározhatjuk a TP függvény kanonikus alakját:

$$y = \mathcal{D} \boxtimes_{n=1}^N \tilde{\mathbf{w}}_n(p_n), \quad (3.12)$$

ahol $\mathcal{D}$ tenzor $R_1 \times R_2 \times \dots \times R_N$ méretű.

A kanonikus alak tulajdonságai:

- ha az eredeti súlyfüggvényrendszer $\mathbf{w}_n(p_n)$ ortonormált rendszert alkotott, a kanonikus alak súlyfüggvényei is ortonormált rendszert alkotnak, azaz

$$\int_{a_n}^{b_n} \tilde{w}_{n,i}(p_n) \tilde{w}_{n,j}(p_n) dp_n = \delta_{i,j} \quad (3.13)$$

minden $n = 1..N$ -hez és $1 \leq i, j \leq I_n$ esetén. (Itt $\delta_{i,j}$ a Kronecker-delta, amire $\delta_{i,j} = 1$: ha $i=j$, $\delta_{i,j} = 0$: ha $i \neq j$.)

- $\mathcal{D}$ magtenzor tartalmazza a csúcspontokat,
- a kanonikus alak a HOSVD eljárásból következő ortogonalitási, rendezettségi tulajdonságokkal rendelkezik és $\mathcal{D}$ magtenzor minimális méretű.

3.3. TP függvény kanonikus alakjának numerikus rekonstrukciója TP modell transzformációval

A TP modell transzformáció célja a kanonikus TP alak előállítása folytonos függvényekhez. Ez TP függvények esetén annak egy egyedi alakja, de a későbbiekben látható lesz, hogy a módszer lehetőséget ad nem-TP függvények TP alakú approximációjára is. Az eljárást az érthetőség kedvéért itt skalár függvényre vezetem be, de a továbbiakban látható lesz a kiterjesztése is.

Transzformációs tér (Ω): a függvény értelmezési tartományának azon hiper-téglatest alakú részhalmaza, amelyen a TP modell transzformációt elvégezzük, azaz $\mathbf{p} \in \Omega = [a_1, b_1] \times [a_2, b_2] \times \dots \times [a_N, b_N]$.

Diszkrétizációs háló (M): az Ω transzformációs téren definiált rácsháló. Az n -dik dimenzióban a pontok számát jelölje M_n , $n = 1..N$. A rácspontok egyenletes eloszlása esetén az n -dik dimenzióban a rácspontok $g_{n,m_n} = a_n + \frac{b_n - a_n}{M_n - 1}(m_n - 1)$ helyeken helyezkednek el, ahol $m_n = 1..M_n$. Az egyes rácspontokat definiáló vektorok ekkor

$$\mathbf{g}_{m_1, \dots, m_N} = \begin{bmatrix} g_{1,m_1} & g_{2,m_2} & \dots & g_{N,m_N} \end{bmatrix}^T. \quad (3.14)$$

3.11. Definíció. (Diszkrétizált függvény). Az $y = f(\mathbf{p})$ függvény diszkrétizált alakja alatt a $\mathcal{F}^{D(\Omega, M)} \in \mathbb{R}^{M_1 \times \dots \times M_N}$ tenzort értjük, melyet a függvény rácspontbeli értékei alkotnak. Az M rácsháló minden rácspontjában meghatározva az $y = f(\mathbf{p})$ függvény értékét, feltölthetők velük a tenzor:

$$\mathcal{F}_{m_1, m_2, \dots, m_N}^{D(\Omega, M)} = f(\mathbf{g}_{m_1, m_2, \dots, m_N}). \quad (3.15)$$

(A felső indexben $D(\Omega, M)$ jelöli a diszkrétizációt Ω transzformációs tér felett, M rácspontjaiban.)

A TP modell transzformáció 3 fő lépésre tagolható, a következő három pontban ezeket ismertetem.

3.3.1. I. lépés: diszkrétizáció

Ebben a lépésben a $\mathcal{F}^{D(\Omega, M)}$ diszkrétizált tenzort határozzuk meg. Először definiáljuk a transzformációs teret, mint $y = f(\mathbf{p})$ függvény értelmezési tartományának egy hiper-téglatest alakú részhalmazát: $\mathbf{p} \in \Omega : [a_1, b_1] \times [a_2, b_2] \times \dots \times [a_N, b_N]$.

Definiáljuk a diszkrétizációs rácsháló méreteit: $M_1, \dots, M_N$.

Végül meghatározzuk a függvény értékeit a rácspontokban, és felépítjük belőlük a $\mathcal{F}^{D(\Omega, M)} \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_N}$ tenzort.

3.3.2. II. lépés: a diszkrétizált függvényhez TP struktúra előállítása

A lépés célja a TP struktúra felépítése. Ehhez elvégezzük $\mathcal{F}^{D(\Omega, M)}$ tenzor HOSVD felbontását, melyből:

$$\mathcal{F}^{D(\Omega, M)} = \mathcal{D} \boxtimes_{n=1}^N \mathbf{U}_n = \mathcal{D} \boxtimes_{n=1}^N \mathbf{W}_n^{D(\Omega, M)}, \quad (3.16)$$

ahol

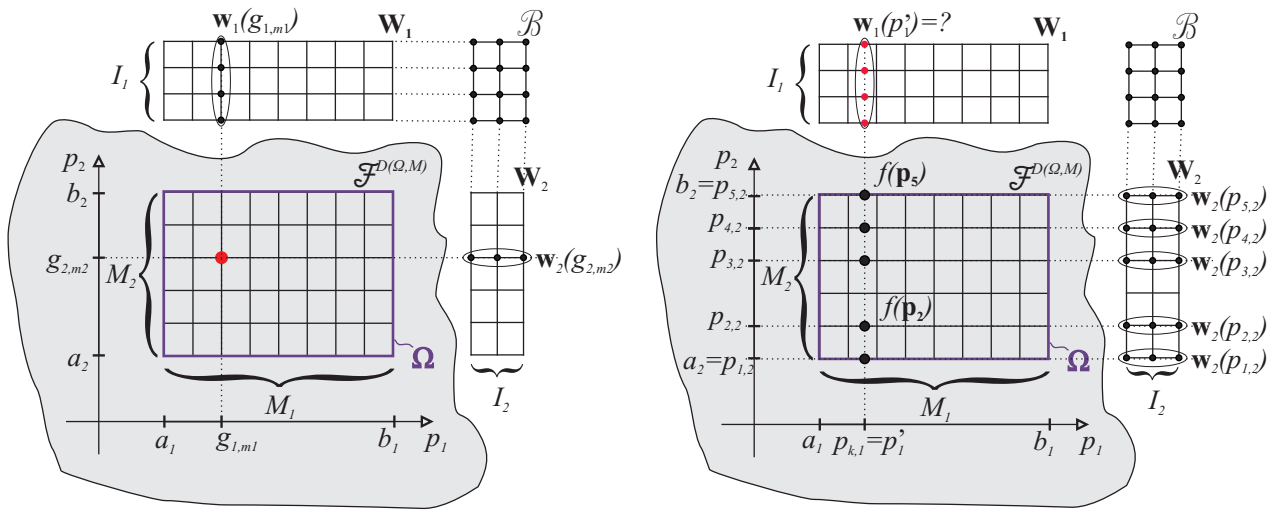
- $\mathcal{D}$ magtenzor méretét $I_1 \times \dots \times I_N$ -vel jelöljük. CHOSVD alkalmazása esetén a magtenzor méretei a diszkrétizált tenzor rangjaival egyeznek meg, azaz $I_n = R_n$ minden $n = 1..N$ -re. RHOSVD alkalmazásával csökkenthetjük a TP alak méretét: szinguláris értékeinek és súlyfüggvényeinek számát, dimenzióként külön-külön. Ekkor legalább egy dimenzióban $I_n < R_n$.
- a HOSVD eljárás során kapott $\mathbf{U}_n$ (szinguláris vektorokat tartalmazó mátrixok) adják a kanonikus alak diszkrétizált $\mathbf{W}_n^{D(\Omega, M)}$ súlyfüggvényeit is. Ezek $M_n \times I_n$ méretű mátrixok. Oszlopaiban az I_n darab súlyfüggvény található M_n pontban mintavételezve. M_n darab sora az I_n értéket tartalmazó súlyfüggvényvektor $\mathbf{w}_n(g_{n, m_n=1..M_n})$ rácsponti értékeinek felel meg.

A magtenzor és a diszkretizált súlyfüggvények ismeretében vissza tudjuk számítani a függvény értékét a rácspontokban a 3.8 definíció alapján. A művelet illusztrációja a 3.2.a ábrán látható kétdimenziós paraméterter esetére. A (pirossal jelölt) $f(\mathbf{g}_{m_1, m_2})$ rácsponti érték számítható a hozzá tartozó súlyfüggvényvektor értékei és a magtenzor tenzorszorzataként:

$$f(\mathbf{g}_{m_1, m_2}) = \mathcal{D} \times_1 \mathbf{w}_1(g_{1, m_1}) \times_2 \mathbf{w}_2(g_{2, m_2}). \quad (3.17)$$

(Matematikailag a magtenzort minden dimenzió mentén a súlyfüggvények értékeinek megfelelően súlyozva adunk össze. Az ábrán a szorzásban részt vevő értékek feketével ki vannak emelve.)

A számítás RHOSVD esetén értelemszerűen csak közelíti az függvény értékét.



3.2. ábra. a., a TP alak használata egy rácsponti érték meghatározásához b., a súlyfüggvény értékének meghatározása a rácspontok között

3.3.3. III. lépés: a folytonos súlyfüggvények rekonstrukciója

A II. lépésben kapott (3.16) struktúra CHOSVD eljárás esetén pontosan követi az eredeti függvény értékeit a rácspontokban. Ebben a lépésben bemutatom, hogyan lehet $[a_n, b_n]$ intervallum bármely pontján értéket adni a súlyfüggvénynek minimális approximációs hiba mellett. Ha tetszőleges pontban meg tudjuk határozni a súlyfüggvényeket, előállíthatóak a folytonos súlyfüggvények.

Keressük az $n = d$ -dik paraméterhez tartozó $\mathbf{w}_d(p_d)$ súlyfüggvényvektor értékét, tetszőleges $p_d = p'_d \in [a_d, b_d]$ helyen, azaz $\mathbf{w}_d(p'_d)$ súlyfüggvényvektorértéket.

Válasszunk olyan paramétervektorokat $(\mathbf{p}_k)$, melyekben a d -dik paraméter értéke azonos: $p_{k,d} = p'_d$ (minden k -ra), a többi változót pedig helyezzük a rács különböző pontjaiba.

A választott paramétervektorokhoz határozzuk meg a függvényértékeket: $f(\mathbf{p}_k)$, majd képezzük az ismert súlyfüggvényvektorokhoz tartozó következő tenzorszorzat d -módú mátrixfel-

bontását is:

$$\left(\mathcal{D} \bigotimes_{n=1, n \neq d}^N \mathbf{w}_n(p_{k,n})\right)_{(d)}, \quad (3.18)$$

ami I_d elemű oszlopvektor.

A keresett súlyfüggvényérték meghatározásához ekkor a következő egyenlet írható fel:

$$\mathbf{w}_d(p'_d) = \mathbf{h} \cdot \mathbf{D}^+ \quad (3.19)$$

ahol

- $\mathbf{h} = [\dots f(\mathbf{p}_k) \dots]$: K elemű sorvektor,
- $\mathbf{D} = \left[\dots \left(\mathcal{D} \bigotimes_{n=1, n \neq d}^N \mathbf{w}_n(p_{k,n})\right)_{(d)} \dots \right]$: $I_d \times K$ méretű mátrix.

A lépés alapja 3.8 átrendezett formája, ami tetszőleges $\mathbf{p}_k$ paramétervektorra:

$$f(\mathbf{p}_k) = \left(\mathcal{D} \bigotimes_{n=1, n \neq d}^N \mathbf{w}_n(p_{k,n})\right) \times_d \mathbf{w}_d(p'_d). \quad (3.20)$$

Képezve a tenzor d -módú mátrixfelbontását az egyenlet mátrixegyenletté egyszerűsíthető:

$$f(\mathbf{p}_k) = \mathbf{w}_d(p'_d) \left(\mathcal{D} \bigotimes_{n=1, n \neq d}^N \mathbf{w}_n(p_{k,n})\right)_{(d)}, \quad (3.21)$$

azonban ebből az egy egyenletből még nem határozható meg az ismeretlen $\mathbf{w}_d(p'_d)$ vektor értéke. Azonban több (K darab) egyenletet felírva lehetőségünk van $\mathbf{w}_d(p'_d)$ meghatározására, ha van közöttük legalább I_d darab lineárisan független:

$$[\dots f(\mathbf{p}_k) \dots] = \mathbf{w}_d(p'_d) \left[\dots \left(\mathcal{D} \bigotimes_{n=1, n \neq d}^N \mathbf{w}_n(p_{k,n})\right)_{(d)} \dots \right], \quad (3.22)$$

röviden:

$$\mathbf{h} = \mathbf{w}_d(p'_d) \cdot \mathbf{D}, \quad (3.23)$$

amelyet $\mathbf{D}$ pszeudoinverzézével beszorozva adódik 3.19, minimális hibát adó összefüggés.

A művelet illusztrációja $N = 2$ esetre a 3.2.b ábrán látható.

Lineáris interpoláció alkalmazása. A legtöbb gyakorlati alkalmazásban a rácspontok olyan sűrűn helyezkednek el, hogy a súlyfüggvények két rácspont között lineárisnak tekinthetők. Ekkor elegendő lehet (a fent ismertetett eljárás helyett) a súlyfüggvény értékét a szomszédos rácsponti súlyfüggvényértékek lineáris interpolációjaként számítani.

3.3.4. A kanonikus alak approximációs alkalmazásai

Nem-TP függvény approximációja TP alakban. A függvényt akkor nevezzük nem-TP függvénynek, ha tetszőlegesen sűrű rácsháló esetén is maximális rangú tenzort kapunk, azaz a függvény végtelen rangú. Mivel csak véges TP alakot tudunk előállítani, az elhagyott szinguláris értékek approximációs hibát okoznak.

TP függvény komplexitásának csökkentése. A gyakorlatban előnyösebb lehet kevesebb csúcspontot tartalmazó TP alak használata. Az egyes dimenziók kiterjedése csökkenthető az utolsó értékek (és a hozzájuk tartozó súlyfüggvények) elhagyásával. A kanonikus alak rendezettsége miatt ezzel minimális hibájú közelítést tudunk előállítani, és az approxmációs hiba az (3.6) összefüggéssel becsülhető.

3.4. LPV, qLPV rendszerek politopikus reprezentációja

A TP modell transzformáció értelemszerűen kiterjeszthető skalárfüggvényekről vektor-, mátrix- vagy tenzorfüggvényekre is. Ezek TP modell transzformációja során a diszkretizált függvény rácsponti elmei és a magtenzor csúcsponti elmei is vektorok, mátrixok illetve tenzorok lesznek.

A módszer irányítástechnikai alkalmazásában, az LPV vagy qLPV rendszerek paraméterfüggő rendszermátrixait hozzuk TP alakra a vizsgált paraméterterén, ami a numerikus eljárások számára könnyebben kezelhető, kompakt forma.

Tekintsük először az LPV és a qLPV rendszerek definícióit.

3.12. Definíció. (LPV, qLPV rendszer.) Tekintsük a következő lineáris paraméterváltozójú (LPV) rendszert:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}(\mathbf{p}(t))\mathbf{x}(t) + \mathbf{B}(\mathbf{p}(t))\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}(\mathbf{p}(t))\mathbf{x}(t) + \mathbf{D}(\mathbf{p}(t))\mathbf{u}(t),\end{aligned}\tag{3.24}$$

ahol az inputvektor: $\mathbf{u}(t) \in \mathbb{R}^m$, az outputvektor $\mathbf{y}(t) \in \mathbb{R}^l$ és az állapotvektor $\mathbf{x}(t) \in \mathbb{R}^k$. A rendszermátrix így:

$$\mathbf{S}(\mathbf{p}(t)) = \begin{pmatrix} \mathbf{A}(\mathbf{p}(t)) & \mathbf{B}(\mathbf{p}(t)) \\ \mathbf{C}(\mathbf{p}(t)) & \mathbf{D}(\mathbf{p}(t)) \end{pmatrix},\tag{3.25}$$

ami szintén egy paraméterváltozós, $O \times I$ méretű objektum, melyben $\mathbf{p}(t) \in \Omega$ időfüggő, N dimenziós vektor, $\Omega = [a_1, b_1] \times [a_2, b_2] \times \dots \times [a_N, b_N]$ egy zárt hipertéglatest. Amennyiben $\mathbf{p}(t)$ vektor tartalmazza $\mathbf{x}(t)$ állapotvektor elemeit is, qLPV rendszerről beszélünk.

3.13. Definíció. (Véges elemű politopikus modell.) Az $\mathbf{S}(\mathbf{p}(t))$ rendszermátrix összeállítható véges számú, LTI típusú $\mathbf{S}_r \in \mathbb{R}^{O \times I}$ rendszermátrix lineáris kombinációjaként:

$$\mathbf{S}(\mathbf{p}(t)) = \sum_{r=1}^R w_r(\mathbf{p}(t))\mathbf{S}_r,\tag{3.26}$$

minden $\mathbf{p}(t) \in \Omega$ esetén.

3.14. Definíció. (Véges elemű, TP típusú politopikus modell.) Ha a (3.26) összefüggés átírható a következő formába:

$$\mathbf{S}(\mathbf{p}(t)) = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \dots \sum_{i_N=1}^{I_N} w_{1,i_1}(p_1(t))w_{2,i_2}(p_2(t)) \dots w_{N,i_N}(p_N(t))\mathbf{S}_{i_1,i_2,\dots,i_N},\tag{3.27}$$

vagy röviden TP alakba:

$$\mathbf{S}(\mathbf{p}(t)) = \mathcal{S} \boxtimes_{n=1}^N \mathbf{w}_n(p_n(t)), \quad (3.28)$$

ahol az $\mathcal{S}$ magtenzor $I_1 \times I_2 \times \dots \times I_N \times O \times I$ méretű, a rendszer véges elemű, TP típusú politopikus modelljét kapjuk.

3.5. Konvex burok manipuláció

A kanonikus TP alakot lehetőségünk van úgy manipulálni, hogy az új TP alak csúcspontjai a leírás konvex burkának csúcspontjai legyenek. Ekkor az $\mathbf{S}(\mathbf{p})$ értékei tetszőleges $\mathbf{p} \in \Omega$ esetén a csúcsponti rendszerek konvex kombinációi lesznek, ahogy a következő definícióban is látható.

3.15. Definíció. A TP függvény konvex, ha a súlyfüggvények kielégítik a következő kritériumokat:

$$w_{n,i_n}(p_n) \in [0, 1] \quad (3.29)$$

$$\sum_{i_n=1}^{I_n} w_{n,i_n}(p_n) = 1 \quad (3.30)$$

minden $n = 1 \dots N$, $i_n = 1 \dots I_n$ és $p_n = [a_n, b_n]$ -re.

Röviden áttekintjük a kidolgozott konvex burok típusokat.

3.16. Definíció. (SN típusú TP függvény.) A konvex TP függvény SN típusú (Sum Normalized), ha minden $p \in \Omega$ -re teljesül 3.30 feltétel.

3.17. Definíció. (NN típusú TP függvény.) A konvex TP függvény SN típusú (Non-Negative), ha a súlyfüggvény értékei minden $p \in \Omega$ esetén nem-negatívak.

3.18. Definíció. (NO/CNO típusú TP függvény.) A konvex TP függvény NO típusú (Normal), ha teljesül (3.29) és (3.30) kritérium és a súlyfüggvények maximuma 1. CNO (Close to Normal) típusúnak nevezzük, ha teljesül (3.29) és (3.30) kritérium és a súlyfüggvények maximumai egy értékűek, vagy tartanak egyhez.

3.19. Definíció. (RNO típusú TP függvény.) A konvex TP függvény RNO típusú (Relaxed Normal), ha teljesül (3.29) és (3.30) kritérium és a súlyfüggvények maximumai azonos értékűek.

3.20. Definíció. (INO típusú TP függvény.) A konvex TP függvény INO típusú (Inverse Normal), ha teljesül (3.29) és (3.30) kritérium és a súlyfüggvények minimumai zérus értékűek.

3.21. Definíció. (IRNO típusú TP függvény.) A konvex TP függvény IRNO típusú (Inverse Normal), ha teljesül (3.29) és (3.30) kritérium, a súlyfüggvények minimumai zérus értékűek és a súlyfüggvények maximumai azonos értékűek.

A konvex burok manipulációval lehetőségünk nyílik qLPV rendszerek konvex politopikus modelljének előállítására, ami közvetlenül felhasználható korszerű LMI alapú szabályozótervezés során. [19]

3.6. TP^τ modell transzformáció

Az irányítási rendszerek gyakori problémája az időkéésés. Időkéésést okoz a mintavételezés, a szabályozó számítási ideje továbbá a kommunikációs csatornán való áthaladás is. A rendszer tulajdonságait jelentősen befolyásolhatja, akár instabillá is teheti.

A fő probléma, hogy az időkéésés nem jelenik meg explicit paraméterként az LPV modellben, hatása csak szimulációval figyelhető meg. Ezért az előző fejezetben ismertetett TP modell transzformáció sem hajtható végre időkééséses rendszerekre.

Galambos Péter [3] munkájában kiterjesztette a TP modell transzformációt, hogy végrehajtható legyen időkééséses rendszerekre is. A TP^τ modell transzformáció végrehajtása után a paramétervektor egy eleme lesz az időkéésés, így időkééséses rendszerek politopikus reprezentációja is előállítható.

A következő részben [3] alapján fogom röviden összefoglalni a TP^τ transzformáció elvét és lépéseit.

3.6.1. Időkééséses rendszer újradefiniálása

Időkééséssel rendelkező rendszerek általánosan a következő, algebrai-differenciálegyenletrendszerrel adhatóak meg:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{A}(\mathbf{p}_0(t))\mathbf{x}(t) + \mathbf{B}_1(\mathbf{p}_0(t))\mathbf{u}(t) + \mathbf{B}_2(\mathbf{p}_0(t))\mathbf{w}(t) \\ \mathbf{y}(t) &= \mathbf{C}_1(\mathbf{p}_0(t))\mathbf{x}(t) + \mathbf{D}_{11}(\mathbf{p}_0(t))\mathbf{u}(t) + \mathbf{D}_{12}(\mathbf{p}_0(t))\mathbf{w}(t) \\ \mathbf{z}(t) &= \mathbf{C}_2(\mathbf{p}_0(t))\mathbf{x}(t) + \mathbf{D}_{21}(\mathbf{p}_0(t))\mathbf{u}(t) + \mathbf{D}_{22}(\mathbf{p}_0(t))\mathbf{w}(t),\end{aligned}\tag{3.31}$$

ahol

$$\mathbf{w}(t) = [z_1(t - \tau) \quad \dots \quad z_L(t - \tau)]^T,\tag{3.32}$$

azaz $z_{1..L}(t)$ állapotváltozók értéke τ időkééséssel hat a rendszerre. Ez átírható a következő kompakt formába is:

$$\begin{bmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \\ \mathbf{z}(t) \end{bmatrix} = \mathbf{S}(\mathbf{p}_0(t)) \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \\ \mathbf{w}(t) \end{bmatrix},\tag{3.33}$$

ahol:

$$\mathbf{S}(\mathbf{p}(t)) = \begin{pmatrix} \mathbf{A}(\mathbf{p}_0(t)) & \mathbf{B}_1(\mathbf{p}_0(t)) & \mathbf{B}_2(\mathbf{p}_0(t)) \\ \mathbf{C}_1(\mathbf{p}_0(t)) & \mathbf{D}_{11}(\mathbf{p}_0(t)) & \mathbf{D}_{12}(\mathbf{p}_0(t)) \\ \mathbf{C}_2(\mathbf{p}_0(t)) & \mathbf{D}_{21}(\mathbf{p}_0(t)) & \mathbf{D}_{22}(\mathbf{p}_0(t)) \end{pmatrix},\tag{3.34}$$

és

$$\mathbf{w}(t) = [z_1(t - \tau) \quad \dots \quad z_L(t - \tau)]^T.$$

Az újradefiniálás célja olyan LTI rendszerek felírása, amelyek egy-egy konstans $\mathbf{p} = \begin{bmatrix} \mathbf{p}_0 \\ \tau \end{bmatrix}$ paraméterezés esetén jól közelítik az eredeti, időkéséses rendszer viselkedését. Az újradefiniált rendszer

$$\begin{bmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{bmatrix} = \mathbf{S}(\mathbf{p}) \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{bmatrix} \quad (3.35)$$

alakban írható fel.

Az újradefiniálást a következő módon jelöljük a továbbiakban:

$$\mathbf{S}(\mathbf{p}) = \text{redef}(f(\mathbf{p}_0, \mathbf{u}(t))). \quad (3.36)$$

3.6.2. TP^τ transzformáció lépései

A TP^τ modell transzformáció abban különbözik a TP modell transzformációtól, hogy nem áll rendelkezésünkre egy explicit függvény, amelybe tetszőleges paraméterezés helyettesíthető, hanem csak újradefiniálással határozhatjuk meg $\mathbf{S}(\mathbf{p})$ értékét. A transzformáció 1. és 3. lépése változik meg, a következő pontokban mindhárom lépést röviden ismertetem.

I. lépés: diszkretizáció újradefiniálással

Ebben a lépésben a cél $\mathcal{S}^{D(\Omega, M)}$ diszkretizált tenzor meghatározása. Definiáljuk Ω paraméterteret, amely már tartalmazza az időkésést is, majd az M rácshálót.

A rácspontokban elvégezzük a rendszer újradefiniálását:

$$\mathbf{S}(\mathbf{p}) = \text{redef}(f(\mathbf{p}, \mathbf{u}(t))), \quad (3.37)$$

majd a kapott $\mathbf{S}(\mathbf{p}) \in \mathbb{R}^{O \times I}$ LTI rendszermátrixokból összeállítjuk az $\mathcal{S}^{D(\Omega, M)} \in \mathbb{R}^{M_1 \times \dots \times M_N \times O \times I}$ diszkretizált tenzort.

II. lépés: a diszkretizált tenzor kibontása TP struktúrába

A lépés célja a TP struktúra felépítése. Ehhez elvégezzük $\mathcal{S}^{D(\Omega, M)}$ tenzor HOSVD felbontását az első N dimenziójában, melyből:

$$\mathcal{S}^{D(\Omega, M)} = \mathcal{D} \boxtimes_{n=1}^N \mathbf{U}_n = \mathcal{D} \boxtimes_{n=1}^N \mathbf{W}_n^{D(\Omega, M)} \quad (3.38)$$

ahol

- $\mathcal{D}$ magtenzor méretét $I_1 \times \dots \times I_N \times O \times I$ -vel jelöljük. CHOSVD alkalmazása esetén a magtenzor méretei I_n a diszkretizált tenzor rangjaival egyeznek meg, azaz $I_n = R_n$ minden $n = 1..N$ -re. RHOSVD alkalmazásával csökkenthetjük a TP alak méretét: szinguláris értékeinek és súlyfüggvényeinek számát, dimenzióként külön-külön. Ekkor legalább egy dimenzióban $I_n < R_n$.
- a HOSVD eljárás során kapott $\mathbf{U}_n$ (szinguláris vektorokat tartalmazó mátrixok) adják a kanonikus alak diszkretizált $\mathbf{W}_n^{D(\Omega, M)}$ súlyfüggvényeit is. Ezek $M_n \times I_n$ méretű mátrixok. Oszlopaiban az I_n darab súlyfüggvény található M_n pontban mintavételezve. M_n darab sora az I_n értéket tartalmazó súlyfüggvényvektor $\mathbf{w}_n(g_{n, m_n=1..M_n})$ rácsponti értékeinek felel meg.

A magtenzor és a diszkretizált súlyfüggvények ismeretében vissza tudjuk számítani a függvény értékét a rácspontokban 3.28 definíció alapján.

III. lépés: folytonos súlyfüggvények rekonstrukciója

A feladat $[a_d, b_d]$ intervallum tetszőleges (p'_d) pontján értéket adni a d -dik súlyfüggvénynek, azaz $\mathbf{w}_d(p'_d)$ -nek. Ha tetszőleges pontban meg tudjuk határozni a súlyfüggvényeket, előállíthatók a folytonos súlyfüggvények.

Válasszunk olyan paramétervektorokat $(\mathbf{p}_k)$, melyekben a d -dik paraméter értéke azonos: $p_{k,d} = p'_d$ (minden k -ra), a többi változót pedig helyezzük a rács különböző pontjaiba.

A választott paramétervektorokhoz újradefiniálással határozzuk meg $\mathbf{S}(\mathbf{p}_k)$ rendszermátrixot, majd képezzük az ismert súlyfüggvényvektorokhoz tartozó következő tenzorszorzat d -módú mátrixfelbontását is:

$$\left(\mathcal{S} \bigotimes_{n=1, n \neq d}^N \mathbf{w}_n(p_{k,n}) \right)_{(d)}, \quad (3.39)$$

ami $I_d \times (O \cdot I)$ elemű mátrix.

A keresett súlyfüggvényérték meghatározásához ekkor a következő egyenlet írható fel:

$$\mathbf{w}_d(p'_d) = \mathbf{h} \cdot \mathbf{D}^+ \quad (3.40)$$

ahol

- $\mathbf{h} = \left[\dots (\mathbf{S}(\mathbf{p}_k))_{(3)} \dots \right] : O \cdot I \cdot K$ elemű sorvektor,
- $\mathbf{D} = \left[\dots \left(\mathcal{D} \bigotimes_{n=1, n \neq d}^N \mathbf{w}_n(p_{k,n}) \right)_{(d)} \dots \right] : I_d \times (O \cdot I \cdot K)$ méretű mátrix.

Lineáris interpoláció alkalmazása. A legtöbb gyakorlati alkalmazásban a rácspontok olyan sűrűn helyezkednek el, hogy a súlyfüggvények két rácspont között lineárisnak tekinthetők. Ekkor elegendő lehet (a fent ismertetett eljárás helyett) a súlyfüggvény értékét a szomszédos rácsponti súlyfüggvényértékek lineáris interpolációjaként számítani.

3.7. Összefoglalás

A TP modell transzformáció egységes és egyértelmű reprezentációra ad lehetőséget LPV és qLPV rendszerekre is. Nem tartalmaz analitikus lépéseket, az eljárás teljesen numerikus. A TP^τ transzformációval az időkézés, egy implicit paraméter is beemelhető a TP struktúra változói közé, és a későbbiekben már explicit változóként kezelhető.

A konvex burok képzése után a TP struktúra hasonlónak válik a Takagi-Sugeno fuzzy leírásokhoz: a súlyfüggvényekből képezhetőek a tagsági függvények és a vertex pontok maguk a szabályok. Lehetőséget adnak multiobjektív LMI alapú szabályozótervezésre, amellyel különböző szabályozási követelményeket állapíthatunk meg.

A TP modell transzformáció eljárásai implementálva elérhetőek az MTA-SZTAKI MATLAB-TPTool toolboxában [20].

4. fejezet

Merevségbecslő alkalmazások

A robotika egyes területein gyakori probléma a környezettel érintkező robot biztonságos és stabil működésének biztosítása. Az érintkezéskor lejátszódó dinamikus folyamatokban a környezet merevsége kiemelkedő szerepet játszik. A környezeti tulajdonságok ismeretében adaptív szabályozási módszerrel javíthatjuk a szabályozás minőségi jellemzőit és megnövelhető a stabil működési tartomány.

A fejezetben bemutatok néhány merevségbecslő módszert az irodalomból, amelyekkel a mért pozíció és kontakterő alapján becsülhető a környezet merevsége, ismeretlen tulajdonságú környezetben is.

4.1. Feladatspecifikáció

A kontakterő egyaránt függhet a pozíciótól, sebességtől és a gyorsulástól, azaz

$$F(t) = F(x, \dot{x}, \ddot{x}), \quad (4.1)$$

ebből a környezet merevsége általánosan $k = -\frac{\partial F}{\partial x}$ parciális deriváltként adható meg, és a következő módon számítható:

$$k(t) = -\frac{\partial F}{\partial x} \Big|_t = -\frac{\partial F / \partial t}{\partial x / \partial t} \Big|_t = -\frac{\dot{F}(t)}{\dot{x}(t)}. \quad (4.2)$$

Ez az összefüggés viszont nem alkalmazható közvetlenül a gyakorlatban, mivel az erőjel általában nagyon zajos, nem deriválható, zérus sebességek esetén nem is értelmezhető és zérus közeli sebesség esetén a zajos mérési eredmények szinguláris eredményeket is okozhatnak.

A merevségmegfigyelők két nagy csoportra oszthatók. A nem-parametrikus merevségmegfigyelők közvetlenül az aktuális merevség értékét becslik. A parametrikus merevségmegfigyelők a környezet erő-elmozdulás karakterisztikáját $(F(x, \dot{x}, \ddot{x}))$ valamilyen paraméteres alakban írják le, az eljárások célja ezen paraméterek becslése, az aktuális merevségérték ezekből a pozíció ismeretében számítható. [21]

Az előbbi eljárások előnye, hogy egyszerű, általános és robosztus eljárást jelentenek. Az utóbbiakban különválik a környezeti modell paramétereinek meghatározása és ezek alapján a merevség értékének számítása. A parametrikus módszerek előnye, hogy a merevség közvetlenül, azonnal számítható a pozíció adatok alapján, hátrányuk lehet, hogy paraméteres környezetmodellt igényelnek, és sok esetben bonyolult nemlineáris módszereket alkalmaznak, amelyekben nehézkes lehet a stabil és optimális dinamika beállítása, ellenőrzése.

Erikson Weber és Sharf, 2003-ban megjelent cikkében [22] a Kelvin-Voigt-féle lineáris anyagmodellel paraméterezi a környezetet, és ezek meghatározásához különböző megfigyelési algoritmusokat hasonlítanak össze: rekurzív legkisebb négyzetek módszere (RLS), modellreferenciás adaptív megfigyelő és indirekt adaptív megfigyelő. Diolaiti, Melchiorri és Stramigioni 2005-ös cikkében [23] a Kelvin-Voigt modell helyett a nem-lineáris Cross-Huntley anyagmodellt javasolják és bemutatnak egy RLS módszerre épülő megfigyelőt is.

A merevségbecslő alkalmazások fejlesztésének egy másik iránya a változtatható merevségű beavatkozó egységek (Variable Stiffness Actuators) fejlesztéséhez kötődik. A VSA eszközök olyan beavatkozók, melyekben lehetséges a pozíció és a merevség egyidejű vezérlése több bemenőjellel. Az irányítás során vagy matematikai modellből számítják az aktuális merevséget, és nyílt körben vezérlik az értékét, vagy zárt szabályozási kört valósítanak meg online merevségmegfigyelővel. Grioli és Bicchi kutatásának eredményeként elsőként nem-parametrikus impedanciamegfigyelő és merevségmegfigyelő eljárást [24], majd parametrikus merevségmegfigyelőt [21] és kapcsolt impedancia megfigyelőt [25] mutattak be.

4.2. Nem-parametrikus merevségmegfigyelő

Grioli és Bicchi [24] cikkében VSA eszközöknél is alkalmazható, nem-parametrikus merevségbecslő eljárásokat mutatnak be. A kontakterő megváltozása felírható mint rugalmas, disszipatív és tehetetlenségi hatások összege:

$$dF(t) = m(t)d\ddot{x}(t) + b(t)d\dot{x}(t) + k(t)dx(t). \quad (4.3)$$

Az ismertetett eljárások célja az aktuális együtthatók becslése.

Változó merevségmegfigyelő. Az eljárás során feltesszük, hogy a kontaktálódó anyag tömegét (m) és a csillapítását (b) ismerjük. Ezek hatása kivonható a kontakterőből, így a rugalmas hatásért: $F_r = F(t) - m\ddot{x}(t) - b\dot{x}(t)$ rész felel.

A merevség becslésére javasolt megfigyelési struktúra:

$$\dot{\hat{k}} = -\alpha(\hat{k} \cdot \dot{x} - \dot{F}_r)sign(\dot{x}), \quad (4.4)$$

láthatóan (4.1 részben tárgyaltaknak megfelelően) nem a merevséghibát számítják, és csatolják vissza, hanem súlyozzák az aktuális sebesség abszolút értékével, amire:

$$(\hat{k} - k) \cdot |\dot{x}| = (\hat{k} \cdot \dot{x} - \dot{F}_r)sign(\dot{x}). \quad (4.5)$$

Ekkor a megfigyelő elméleti hibájára:

$$|\hat{k} - k| > \frac{1}{\alpha} \frac{|\dot{k}|}{|\dot{x}|} \quad (4.6)$$

elméleti hibaminimum adódik.

Lineáris impedancia megfigyelő. Itt a cél (4.3) összefüggés m, b, k paramétereinek online becslése. A módszer során bemutatják, hogy (4.3) alakú közelítésből felépíthető nem lineáris rendszer, amelyben az m, b, k értékeket is állapotváltozóknak tekinthetjük és ezek a változók megfigyelhetők.

A megfigyelésre nemlineáris állapotmegfigyelő módszerek használhatóak, mint például a kiterjesztett Kalman-szűrő.

4.3. Parametrikus merevségmegfigyelő környezettel érintkező robotrendszerekhez

Diolaiti, Melchiorri és Stramiglioni [23] cikkben parametrikus merevségbecslő eljárást javasoltak, mellyel robottal érintkező ismeretlen anyag mechanikai tulajdonságait becsülhetjük.

A bevezetőben bemutatják, hogy a kontaktusok kis deformációja esetén a lineáris model (Kelvin-Voigt) inkonzisztens a mindennapi tapasztalattal, mivel abban a kontakterő még a deformáció megszűnése előtt nullává válik, és a kontaktpontban már a modell szerint olyan irányú erő hat, amely a tárgy felé húzza a testet.

Az ellentmondás feloldására Hunt és Crossley javasolták először, hogy a viszkózus tag is függjön az elmozdulástól:

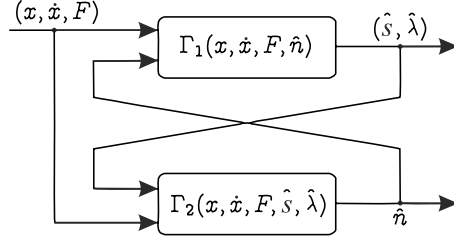
$$F(t) = -x^n(t)(s + \lambda\dot{x}(t)), \quad (4.7)$$

ahol n valós kitevő (rendszerint tart egyhez), k, λ valós konstansok. Ismert paraméterértékek esetén a merevség a következő módon számítható:

$$k(t) = -nx^{n-1}(t)(s + \lambda\dot{x}(t)). \quad (4.8)$$

Az eljárás célja olyan paraméterezés keresése, amely esetén az erőkarakterisztika jól fedi a mérési eredményeket. Ez egy nemlineáris paraméterbecslési feladat, megoldására kettő párhuzamosan kötött RLS (rekurzív legkisebb négyzetek) módszerrel dolgozó lineáris paraméterbecslő egységet javasolnak. A megfigyelő elvi felépítése a 4.1 ábrán látható. Γ_1 megfigyelő s és λ paramétereit becsüli a következő hibaértéket minimalizálva:

$$\epsilon_1(t_i) = \begin{bmatrix} s & \lambda \end{bmatrix} \begin{bmatrix} x^n(t_i) \\ x^n(t_i)\dot{x}(t_i) \end{bmatrix} - F(t_i), \quad (4.9)$$



4.1. ábra. A merevségmegfigyelő elvi felépítése

Γ_2 megfigyelő n kitevőt becsli a következő hibaérték minimalizálásával:

$$\epsilon_2(t_i) = n \log x - \log \frac{F(t_i)}{s + \lambda \dot{x}(t_i)}. \quad (4.10)$$

RLS módszer. Az RLS eljárás során használt jelölések N paraméter becslése esetén:

$$\epsilon(t_i) = \theta^T \mu(t_i) - \varphi(t_i), \quad (4.11)$$

ahol $\mu(t_i)$ N dimenziós, bemenő értékeket tartalmazó vektor, θ N dimenziós paramétervektor, $\varphi(t_i)$ skalár kimenet. A cél $V_j(\theta)$ költségfüggvény minimalizálása:

$$V_j(\theta) = \sum_{i=0}^j \epsilon^2(t_i) \beta^{j-i}, \quad (4.12)$$

ahol $\beta \in [0.99, 1]$ felejtési faktor.

Az implementálható rekurzív összefüggések:

$$\begin{aligned} \mathbf{q}(i+1) &= \mathbf{R}(i) \mu(i+1) [\beta + \mu^T(i+1) \mathbf{R}(i) \mu(i+1)]^{-1} \\ \mathbf{R}(i+1) &= \frac{1}{\beta} [\mathbf{I} - \mathbf{q}(i+1) \mu^T(i+1)] \\ \theta(i+1) &= \theta(i) + \mathbf{q}(i+1) [\varphi(i+1) - \mu^T(i+1) \theta(i)], \end{aligned} \quad (4.13)$$

ahol $\mathbf{q}(i)$: N dimenziós vektor, $\mathbf{R}(i)$: $N \times N$ mátrix és $\mathbf{I}$ az egységmátrix.

4.4. Parametrikus merevségmegfigyelő VSA eszközökhöz

Grioli és Bicchi [21] cikkben egy általános sémát mutat be VSA eszközök merevségbecslésére. VSA eszközökben a rugalmas erő nem csak az x pozíciótól, hanem egyéb változtatható paraméterektől is függ, a dolgozatomban vizsgált rendszerben nincsenek ilyen mérhető paraméterek, ezzel az eljárás lényegesen egyszerűsödik, itt terjedelmi okok miatt csak ezt a változatot mutatom be.

Tegyük fel, hogy az $F(x)$ kontakterő felírható egy függvénytörrel a következő módon:

$$F(x) = \sum_{i=1}^{\infty} c_i f_i(x). \quad (4.14)$$

A függvénytör első N elemét vektoros formába írhatjuk:

$$F(x) = \sum_{i=1}^N c_i f_i(x) + F_{\text{residuum}} = \mathbf{c}^T \mathbf{f}(x) + F_{\text{residuum}}. \quad (4.15)$$

Ekkor a merevség:

$$k(x) = \mathbf{c}^T \frac{\partial \mathbf{f}(x)}{\partial x} + k_{\text{residuum}} = \mathbf{c}^T \mathbf{s}(x) + k_{\text{residuum}} \quad (4.16)$$

Az eljárás célja $\mathbf{c}$ paramétervektor értékeinek meghatározása, amelyekkel az kontakterőt és a merevséget a következő alakban keressük:

$$\hat{k} = \hat{\mathbf{c}}^T \mathbf{s}(x). \quad (4.17)$$

A cikkben javasolt frissítési törvény a következő alakra egyszerűsödik:

$$\dot{\hat{\mathbf{c}}} = -\mathbf{s}(\mathbf{s}^T \mathbf{s})^{-1} \mathbf{A}(\hat{k} \cdot \dot{x} - \dot{F}) \text{sign}(\dot{x}), \quad (4.18)$$

ahol $\mathbf{A}$ együttható mátrix.

4.5. Összefoglalás

Az irodalomkutatás célja az volt, hogy olyan merevségmegfigyelő módszert találjak, amellyel egy ismeretlen környezetben dolgozó telemanipulátor becsülni tudja a megfogott tárgy merevségét az adaptív irányítási algoritmus számára. A megfigyelővel szemben alapvető követelmény a kis hibával és gyors beállással működés.

A fenti algoritmusokhoz elkészítettem a MATLAB Simulink szimulációs környezetbe beilleszthető blokkokat a tulajdonságaik vizsgálatához, és 4.2 részben ismertetett módszert implementáltam a kísérleti környezetbe is, a következő fejezetekben ezeket az eredményeket is ismertetem.

5. fejezet

Az irányítási rendszer konvex politopikus reprezentációja

A fejezet célja a modellezési eljárás bemutatása, amely során 2.1.2 részben ismertetett idő-késéses közös impedancia irányításhoz előállítottam annak konvex politopikus reprezentációját.

Az első részben a irányítási rendszert vizsgálom és bemutatom a használt matematikai modell felírásához szükséges közelítéseket.

A kapott matematikai (időkéséses-LPV) modellhez (3.6 alapján) több különböző konvex politopikus reprezentációt készítettem a TP^r modell transzformációval. A modellezés folyamata több újítást is tartalmaz, a fejezetben ezeket is bemutatom.

A fejezet végén validálom az eredményeket: a konvex politopikus reprezentációk és az idő-késéses rendszer viselkedését numerikus szimulációkkal hasonlítom össze.

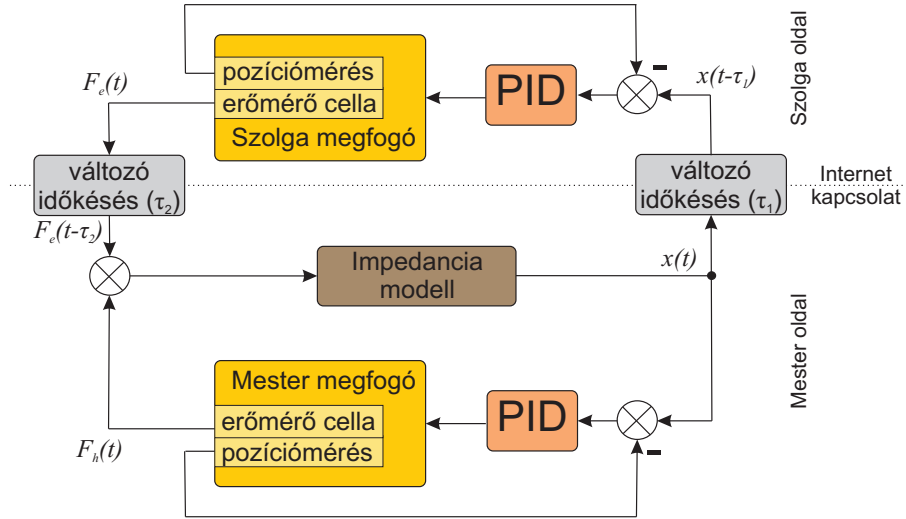
5.1. Modellalkotás

A közös impedancia szabályozás alapú telemanipulációs struktúra részletesen az 5.1 ábrán látható. A mester és a szolga oldal internet kapcsolaton keresztül kommunikál, ami változó időkéséseket okoz a rendszerben.

A modellezés során - figyelembe véve a feladat sajátosságait - a következő elhanyagolásokat tesszük.

1., Pozíciószabályozók idealizálása. A pozíciószabályozásokat ideálisnak tekintjük, mintha a PID szabályozások végtelen sávszélességűek lennének.

2., A távoli környezet modellezése. A telemanipulált környezet fizikai tulajdonságai a legtöbb esetben ismeretlenek. Feltesszük, hogy a környezetben mért erő elsődlegesen a rugalmas hatás következménye, a környezet csillapítását és tehetetlenségét elhanyagoljuk. Ekkor a környezet egyetlen jellemzője a merevsége, ami a környezeti pozíciótól és esetleg az időtől függhet



5.1. ábra. Telemanipulációs irányítási struktúra közös impedanciaszabályozással

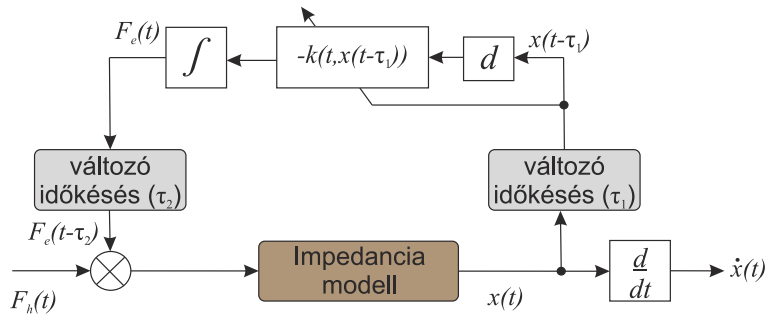
azaz: $k(t) = k(t, x(t - \tau_1))$ és $dF_e(t) = -k(t)dx(t - \tau_1)$.

3., Rendszermmodell. A rendszert az operátor szemszögéből modellezzük: bemenetének a mester oldalon, az operátor által kifejtett erőt választjuk (ez stabilizáló szabályozás esetén kiegészíthető a szabályozó erővel), kimenetnek az operátor által érezhető sebességet.

A fentiek után a rendszer a következő egyenletekkel írható le:

$$\begin{aligned} \ddot{x}(t) &= -\frac{b}{m}\dot{x}(t) + \frac{1}{m}(F_e(t - \tau_2) + F_h(t)), \\ dF_e(t) &= -k(t)dx(t - \tau_1), \end{aligned} \quad (5.1)$$

ahol m , b az impedanciamodellel paraméterei, $k(t) = k(t, x(t - \tau_1))$ a környezet merevsége. A blokkdiagramm a 5.2 ábrán látható.



5.2. ábra. Az elhanyagolások után kapott modell

Algebrai átalakítások után a következő, időkééséseket tartalmazó állapottér modellt írhatjuk

fel:

$$\begin{bmatrix} \ddot{x}(t) \\ \frac{d}{dt}(F_e(t - \tau_2)) \end{bmatrix} = \begin{bmatrix} -\frac{b}{m} & \frac{1}{m} & 0 \\ 0 & 0 & -k(t - \tau_2) \left(1 - \frac{d\tau_1}{dt}\right) \left(1 - \frac{d\tau_2}{dt}\right) \end{bmatrix} \begin{bmatrix} \dot{x}(t) \\ F_e(t - \tau_2) \\ \dot{x}^\tau(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{m} \\ 0 \end{bmatrix} F_h(t),$$

$$\begin{aligned} \dot{x}^\tau(t) &= \dot{x}(t - \tau_1 - \tau_2), \\ k(t - \tau_2) &= k(t - \tau_2, x(t - \tau_1 - \tau_2)). \end{aligned} \quad (5.2)$$

4., Az időkések összevonása. Ha a merevség időinvariáns, vagy változása a jellemző időkések alatt elhanyagolható, $\tau = \tau_1 + \tau_2$ bevezetésével a rendszer megadható a következő időkésekéses LPV modellel:

$$\begin{bmatrix} \ddot{x}(t) \\ \dot{F}_e(t) \end{bmatrix} = \begin{bmatrix} -\frac{b}{m} & \frac{1}{m} & 0 \\ 0 & 0 & -k(t) \left(1 - \frac{d\tau_1}{dt}\right) \left(1 - \frac{d\tau_2}{dt}\right) \end{bmatrix} \begin{bmatrix} \dot{x}(t) \\ F_e(t) \\ \dot{x}^\tau(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{m} \\ 0 \end{bmatrix} F_h(t),$$

$$\begin{aligned} \dot{x}^\tau(t) &= \dot{x}(t - \tau), \\ k(t) &= k(t, x(t - \tau)). \end{aligned} \quad (5.3)$$

A rendszer viselkedését itt már csak négy paraméter: a teljes kör időkése τ , a környezeti merevség k , és az időkések deriváltjai befolyásolják, ezek közül csak a τ időkések nem jelenik meg explicit módon a rendszermátrixban.

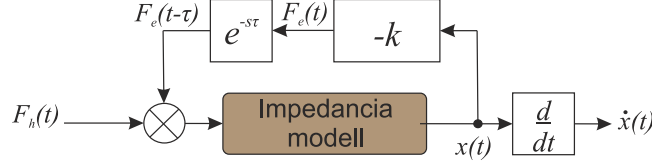
5.2. Politopikus reprezentáció előállítása TP^τ transzformációval

A modellalkotás során az időkésekéses közös impedancia alapú irányítási struktúrához megadtuk (5.3) időkésekéses LPV alakot. A modell lehetőséget ad az időkésekéses deriváltjainak $(\frac{d\tau_1}{dt}, \frac{d\tau_2}{dt})$ figyelembevételére is, így az időkésekéses deriváltjaira is adaptív modell és szabályozó készíthető. Jelenlegi munkám célja az időkésekésre és a környezeti merevségre adaptív politopikus modell bemutatása, $\frac{d\tau_1}{dt} = \frac{d\tau_2}{dt} = 0$ esetén.

5.2.1. Újradefiníálás

Az újradefiníálás során (5.3) rendszermodellt konstans (k, τ) paraméterezés esetén vizsgáljuk, ezzel az 5.3 ábrán látható rendszerhez jutunk. Szimulációval előállítjuk a rendszer sebeségválaszát erő ugrásjelre, és SISO identifikációval 2 tárolós rendszert illesztünk a szimuláció eredményére.

A jelenleg alkalmazott, open-loop SISO identifikációs eljárások hátránya, hogy nem használhatóak instabil, vagy instabil közeli rendszerekre. Ezért figyelembe kell vennünk a rendszer



5.3. ábra. Az újradefiniáláshoz használt modell

paramétereinek és stabilitásának kapcsolatát, amit konzulensem a [26] cikkében tárgyal. Eszerint a rendszer instabillá válik

$$\tau \geq \tau_{crit} = c(m) \frac{b}{k}, \quad (5.4)$$

időkésés esetén, ahol k a rugóként modellezett, időkéséssel visszacsatolt távoli környezet mechanikai merevsége, b az impedanciamodell virtuális csillapítása, $c(m)$ az impedanciamodell virtuális tömegétől is függő konstans. A használt $m = 1[kg]$ tömeg és a $T_s = 1[ms]$ mintavételi idő esetén: $c(m) \approx 1.566$.

A (5.4) korlátot figyelembe véve az identifikáció elvégezhető. A használt ARX identifikáció eredményéből felépíthető a folytonosidejű átviteli függvényt:

$$W(s) = \frac{sX(s)}{F_h(s)} = \frac{b_2s + b_3}{s^2 + a_2s + a_3}. \quad (5.5)$$

A kapott átviteli függvény az elvárásokkal ellentétben nem deriváló jellegű ($b_3 \neq 0$), ami maradó hibát okoz. Az eredmény pontosítására b_3 értékét nullára módosítom, ezzel - az eddigiekkel ellentétben - a modell sebességválaszát már nem terheli maradó hiba.

Harmadik lépésben a deriváló típusú átviteli függvényből állapotter modellt készítünk, amit szemléletesen impedanciamodelleként is megfogalmazhatunk, az alábbi formában:

$$\begin{bmatrix} \ddot{x}(t) \\ \dot{F}_e(t) \end{bmatrix} = \begin{bmatrix} -\frac{b(\mathbf{p})}{m(\mathbf{p})} & \frac{1}{m(\mathbf{p})} \\ -k(\mathbf{p}) & 0 \end{bmatrix} \begin{bmatrix} \dot{x}(t) \\ F_e(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{m(\mathbf{p})} \\ 0 \end{bmatrix} F_h(t). \quad (5.6)$$

Az állapotter modellben csak mérhető változók szerepelnek, és ezek $(\dot{x}, F_e)$ azonosak az (5.3) modell változóival, ezért a modell jól használható változó paraméterek esetén is. Az újradefiniálás eredményeként kapott rendszermátrixok ekkor:

$$[\mathbf{AB}] = \begin{bmatrix} -\frac{b(\mathbf{p})}{m(\mathbf{p})} & \frac{1}{m(\mathbf{p})} & \frac{1}{m(\mathbf{p})} \\ -k(\mathbf{p}) & 0 & 0 \end{bmatrix}. \quad (5.7)$$

A politopikus modellel szembeni elvárásunk, hogy jól reprezentálja a rendszer tulajdonságait és támogassa az alkalmazni kívánt LMI alapú szabályozótervezést. Utóbbi során ugyanis problémát jelenthet, ha a modell csúcspontjaiban „szabályozhatatlan” rendszer áll, vagy numerikusan nem található a kritériumoknak megfelelő megoldás.

A későbbiekben látható lesz, hogy (5.7) alakú újradefiniáláshoz nem tudunk szabályozót tervezni a felhasznált LMI rendszerrel. Különböző állapotter transzformációkat és idő dimenziótlanításokat vezettem be, és ezekkel már lehetségessé vált a szabályozótervezés. A legjobb

eredményeket a következő alakhoz kaptam:

$$\frac{d}{dT} \begin{bmatrix} \dot{x}(T) \\ F_e(T) \sqrt{\frac{m_0}{k(\mathbf{p})}} \end{bmatrix} = \begin{bmatrix} -\frac{b(\mathbf{p})}{m(\mathbf{p})} \sqrt{\frac{m_0}{k(\mathbf{p})}} & \frac{1}{m(\mathbf{p})} \\ -m_0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}(T) \\ F_e(T) \sqrt{\frac{m_0}{k(\mathbf{p})}} \end{bmatrix} + \begin{bmatrix} \frac{1}{m(\mathbf{p})} \sqrt{\frac{m_0}{k(\mathbf{p})}} \\ 0 \end{bmatrix} F_h(t). \quad (5.8)$$

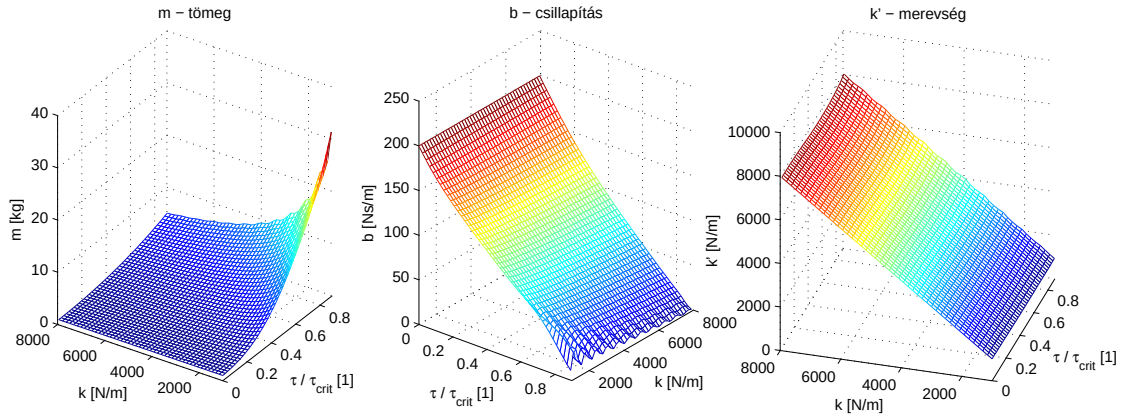
ahol

- T dimeziótlan idő: $T = \frac{t}{\sqrt{m_0/k(\mathbf{p})}}$, $m_0 = 1[kg]$,
- az új állapotvektor $\begin{bmatrix} \dot{x}(t) & F_e(T) \sqrt{\frac{m_0}{k(\mathbf{p})}} \end{bmatrix}$.

Ekkor a rendszermátrix:

$$[\mathbf{AB}] = \begin{bmatrix} -\frac{b(\mathbf{p})}{m(\mathbf{p})} \sqrt{\frac{m_0}{k(\mathbf{p})}} & \frac{1}{m(\mathbf{p})} & \frac{1}{m(\mathbf{p})} \sqrt{\frac{m_0}{k(\mathbf{p})}} \\ -m_0 & 0 & 0 \end{bmatrix}. \quad (5.9)$$

Az időkésés hatásait szemléletesen mutatják be az újradefiniált rendszermátrixban szereplő tömeg, csillapítás és merevségértékek. Ezek a környezet merevségének és τ/τ_{crit} függvényében láthatóak a 5.4 ábrán.

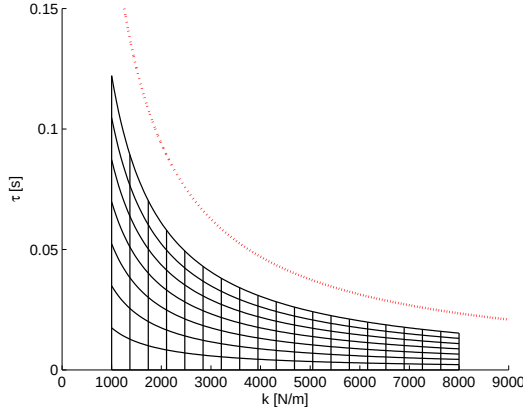


5.4. ábra. A rendszerparaméterek a merevség és az időkésés függvényében

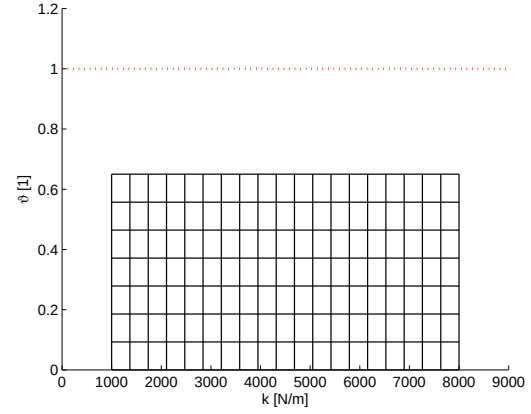
Az eredmények szerint az időkésés a rendszer tehetetlenségérzetét jelentősen megnöveli, és a rendszer disszipatív jellemzői is romlanak, a csillapítás a stabilitási határhoz közelítve fokozatosan nullává válik, és ezzel az identifikáció bizonytalansága is megnő, egyfajta fodrosodás látható itt az értékekben. A rugómerevség értéke pedig gyakorlatilag független az időkéséstől.

5.2.2. Paraméterter és rácsháló

A transzformációs tér kijelölése során figyelembe kell venni az (5.4) összefüggéssel adott stabilitáshatárt. Ezt a 5.5 ábra grafikonjain piros pontozott vonallal jelöltem. A TP^τ modell transzformáció téglalap alakú paramétertartományon hajtható végre, de látható, hogy ilyen



(a) A transzformációs tér alakja $k - \tau$ síkon



(b) A transzformációs tér alakja $k - \vartheta$ síkon

5.5. ábra. Az átparaméterezés hatása a transzformációs tér és a rácsháló alakjára

tartományt nem tudunk illeszteni $k - \tau$ síkon a stabilitási határhoz, hanem az ideális alakú transzformációs tér 5.5(a) ábrán látható alakú lenne. Ezért az időkésés helyett a következő dimenziótlan paramétert vezettem be.

5.1. Definíció. *Relatív időkésés: az időkésés és az analitikusan számítható kritikus időkésés hányadosa:*

$$\vartheta = \frac{\tau}{\tau_{crit}}. \quad (5.10)$$

Az időkésés helyett a relatív időkésést használva paraméternek a paraméterter: $[k_{min}, k_{max}] \times [\vartheta_{min}, \vartheta_{max}]$. Ezzel az 5.5(b) ábrán látható $k - \vartheta$ síkon már ki tudunk jelölni olyan téglalap alakú paramétertartományt, ami minden merevségértéknél azonos mértékben közelíti meg a stabilitási határt.

Válasszuk paraméterternek $\vartheta = 0..0.65$, $k = 1000..8000$ tartományt, rácsméretnek $M_\vartheta = 131$, $M_k = 137$ értékeket. Ezen a paraméterterén még nem jelenik meg a 5.4 ábrán látható fodrosodás, és a nagy rácsméret kellőképpen részletes leírást eredményez.

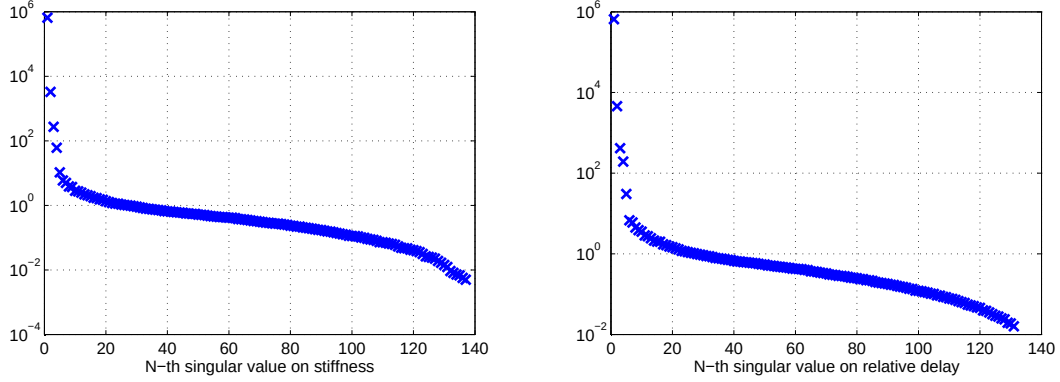
5.2.3. Diszkretizáció

Elvégezve az újradefiniálást, minden rácspontban egy-egy (5.7) vagy (5.9) alakban felírt rendszerátíró kapunk. Ezekből összeállítható $\mathcal{S}^{D(\Omega, M)}$ diszkretizált tenzor, mérete: $137 \times 131 \times 2 \times 3$.

5.2.4. Kanonikus súlyfüggvények

Ha elvégezzük a HOSVD-t a diszkretizációs tenzor k -hoz és ϑ -hoz tartozó dimenziójában, teljes rangú $\mathcal{D}^{CAN} \in \mathbb{R}^{137 \times 131 \times 2 \times 3}$ magtenzort és a hozzá tartozó $\mathbf{W}_k^{CAN}$ és $\mathbf{W}_\vartheta^{CAN}$ súlyfüggvényeket tartalmazó mátrixokat. (A maximális rangnak numerikus okai vannak, az 5.6 áb-

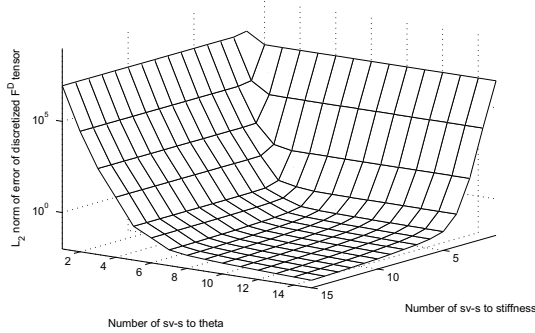
rán látható, hogy a harmadik és a feletti szinguláris értékek csak 0.1%-nál kisebb mértékben módosítják a rendszermátrix elemeinek értékét.) Így ezek elhagyásával a modell komplexitása jelentősen csökkenthető, a pontosság feláldozása nélkül.



5.6. ábra. A szinguláris értékek a merevséghez és a relatív időkéséshez

5.2.5. Komplexitáscsökkentés rangcsökkentéssel

A leírás komplexitása csökkenthető azzal, hogy a kisebb szinguláris értékeket (és a hozzájuk tartozó súlyfüggvényeket) elhagyjuk. Az így kapott approximációkhoz (3.6) alapján számíthatjuk γ_{max} értékét. Ennek értéke $I_k = 1..15$ és $I_\vartheta = 1..15$ esetéhez látható a 5.7. ábrán hálózott megjelenítéssel.



5.7. ábra. γ_{max} különböző komplexitású rendszerekhez

Célunk minimális szinguláris érték megtartása mellett kis hibájú approximáció. Az ábrák alapján azonos számú szinguláris értéket célszerű megtartani a merevséghez és a relatív időkéséshez. A továbbiakban a különböző approximációkat vizsgálom, jelölje ezeket A, B, C, D, E, F .

Az eredeti (5.6) újradefiniálást alkalmazva

- A : csak 1-1 szinguláris értéket tartunk meg: $I_k = 1$ és $I_\vartheta = 1$,
- B : $I_k = 2$ és $I_\vartheta = 2$,
- C : $I_k = 3$ és $I_\vartheta = 3$.

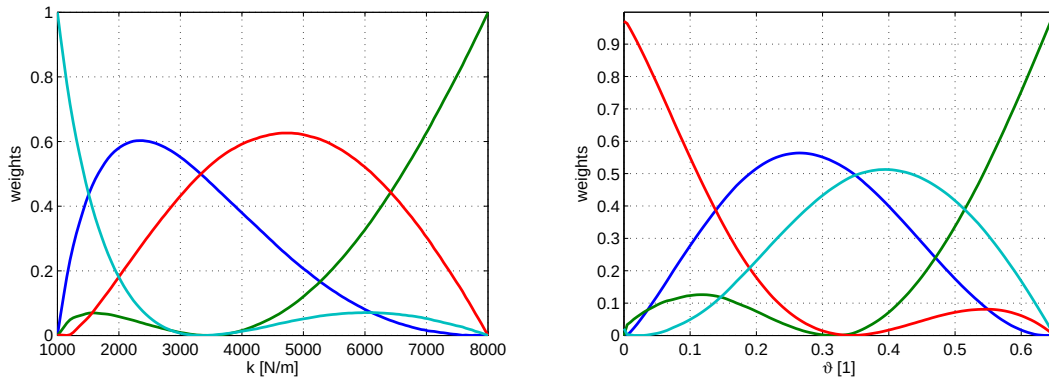
A manipulált (5.8) újradefiniálást alkalmazva

- D : $I_k = 1$ és $I_\vartheta = 1$,
- E : $I_k = 2$ és $I_\vartheta = 2$,
- F : $I_k = 3$ és $I_\vartheta = 3$.

5.2.6. Konvex burok

Esetünkben CNO típusú konvex burok használata célszerű, mivel ez adja a legszűkebb burkot, azaz a csúcsponti rendszerek a modellezett rendszerek közül, vagy azok szűk környezetéből kerülnek ki. A későbbiekben, a szabályozótervezésnél problémát okozhat, ha a csúcsponti rendszerek „messze” esnek a diszkrétizált rendszerektől, mivel szabályozhatatlan csúcsponti rendszerek is adódhatnak.

Különböző approximációkhoz generáltam CNO típusú konvex burkot. Ezek a mellékletben megtalálhatóak. Itt csak a C esethez kapott konvex súlyfüggvényeket (5.8 ábra) mutatom be, de a függelékben A, B, C, D, E, F approximációk konvex burka (csúcsponti rendszerek és súlyfüggvények) is megtalálhatóak.



5.8. ábra. A "C" CNO burok súlyfüggvényei

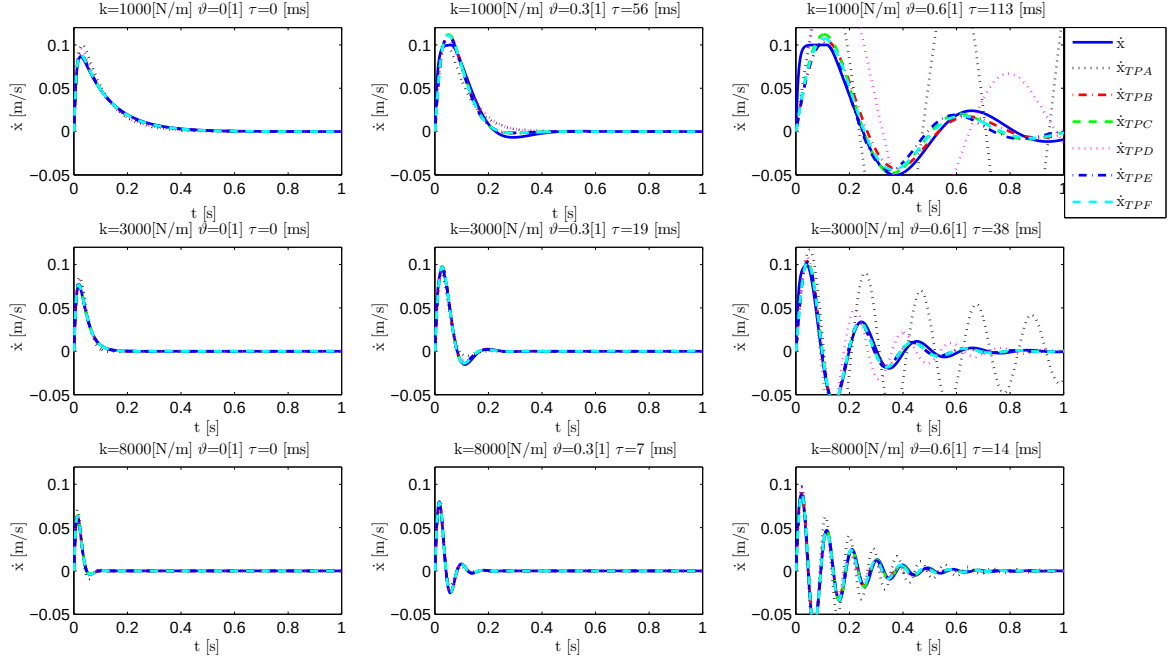
5.3. Validáció

A validáció során numerikus szimulációkkal hasonlítottam össze a CNO típusú politopikus approximációkat egymással és a vizsgált időképes rendszerrel.

5.3.1. Állandó paraméterezés vizsgálata

Először állandó paraméterezés esetén (k, τ : konst.) vizsgáltam meg az A, B, C, D, E, F rendszerek sebességválaszát.

A szimuláció indításától $12[N]$ -os erőgerjesztés hat a rendszerre, a sebességválaszokat rögzítettem. A szimulációt $1.5[s]$ -ig futtattam, ami elegendő idő a tranziensek lefutásához. Az eredményeket az 5.9 ábrán jelenítettem meg.



5.9. ábra. A paraméterezés és a core tenzor méretének hatása a sebességválaszra

Látható, hogy míg az A és D modell viselkedése jelentősen különbözik az időképes rendszertől, C, E, F között már alig látni különbséget, és a B jól közelíti ezeket.

A különböző modellekhez elvégeztem a sebesség átlagos négyzetes eltéréseinek szisztematikus összehasonlítását is. A 5.10 ábrán a függőleges tengelyen látható az $error = \int_{t=0}^{1.5} v_{error}(t)^2 dt / T$ hiba értéke, különböző paraméterértékekhez és modellekhez.

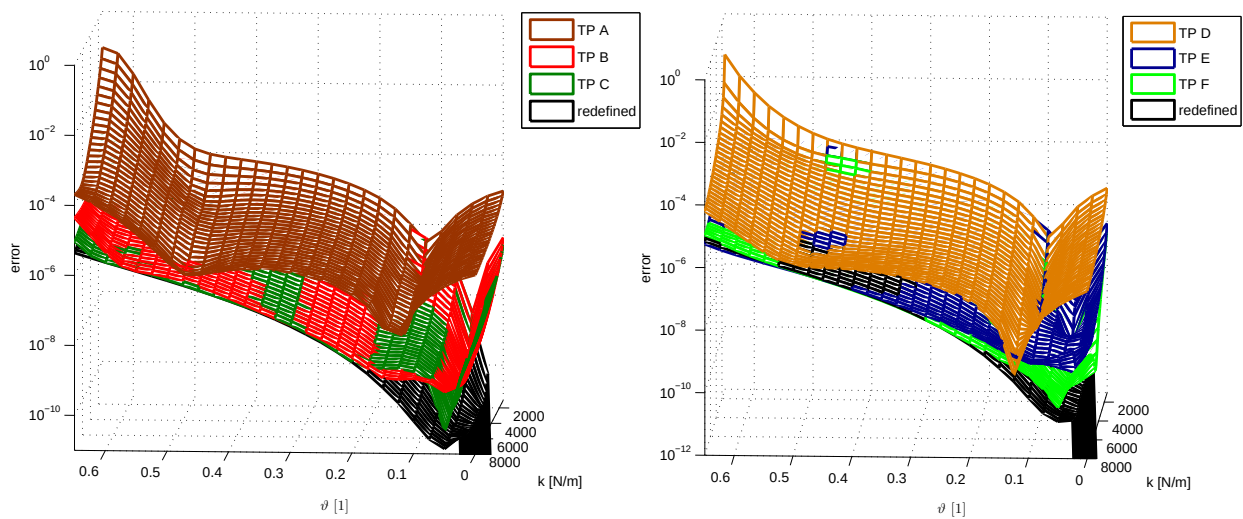
Látható, hogy a több csúcsponthoz tartozó modellek már jól illeszkednek az újradefiniálás eredményére.

5.3.2. Változó paraméterezés vizsgálata

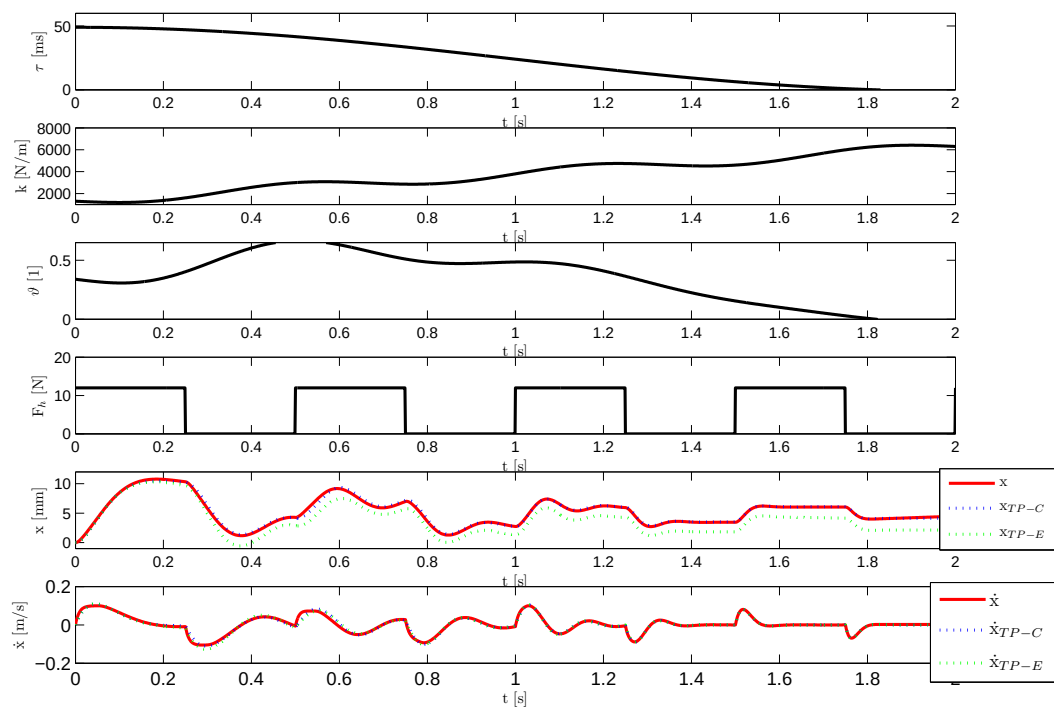
Az előző részben látható volt, hogy konstans paraméterezés esetén a nagyobb konvex burkok már jól approximálják a vizsgált rendszert. Ebben a részben változó paraméterezésű folyamatokhoz vizsgálok meg a modellek viselkedését.

Pozíciófüggő és időfüggő merevségprofilokra is elvégeztem az validációt változó időkézés mellett B és E modellhez. A futási eredmények a 5.12 és 5.11 képeken láthatóak. Mindkettő politopikus modell jól közelíti az időkézéses rendszert.

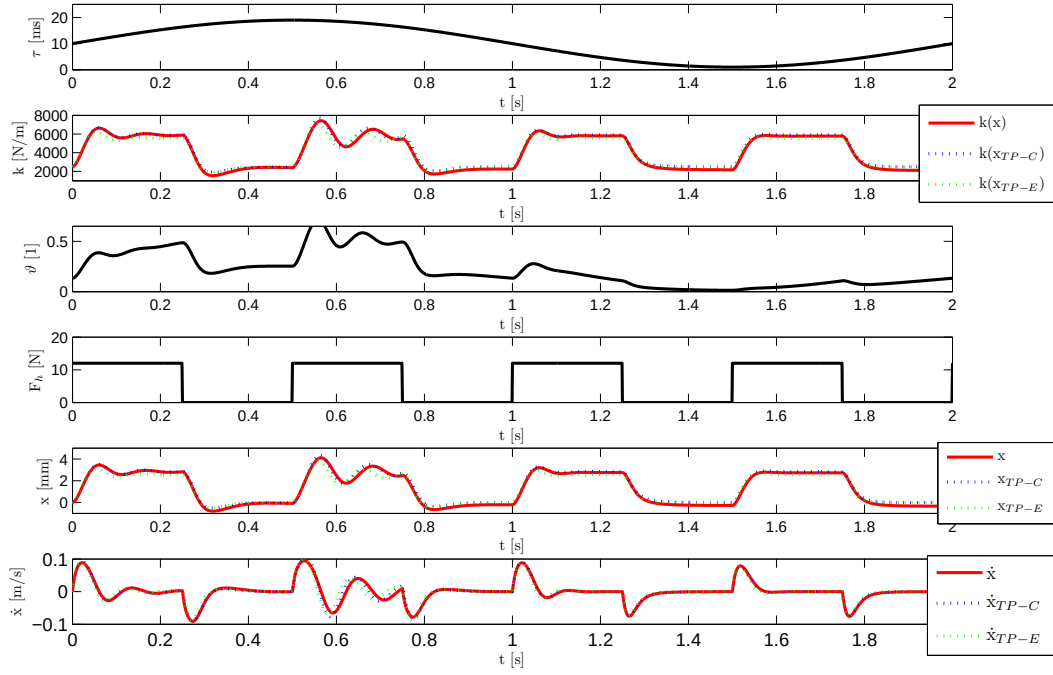
A C és F modellek tovább javítják az approximációt.



5.10. ábra. A különböző modellek sebességhibájának négyzetes integrálja különböző paraméterezések esetén



5.11. ábra. Időtől függő merevségprofil alkalmazása



5.12. ábra. Pozíciófüggő merevségprofil alkalmazása

5.4. Összefoglalás

A fejezetben bemutatam miként kaptam az irányítási struktúra matematikai modelljét, majd részletesen bemutatam a TP^τ modell transzformációt végrehajtását.

Kiterjesztettem a modell érvényességét, hogy különböző időkések és különböző merevségek esetén is jó közelítést adjon. A kutatás ezen eredményeit [27] cikkben publikáltuk. Továbbá bemutattam, hogy a modellezés átparaméterezéssel eredetileg nem téglatest alakú transzformációs téren is végrehajtható.

A feladathoz specializált újradefiniálási módszerrel a rendszernek olyan modelljét állítottam elő, melyben a modell sebességválaszát nem terheli maradó hiba, a modellben csak mérhető/számítható állapotváltozók szerepelnek, és ezért a későbbiekben – az eddigiekkel ellentétben – nincs szükség állapotmegfigyelőre. Ezzel egyszerűsödik az irányítási struktúra és a tervezési feladat.

A kapott konvex politopikus modellekhez átfogó validációt végeztem, ezek igazolták a modell használhatóságát.

6. fejezet

Szabályozótervezés

A modellezés során már láthatóvá váltak az időkésés negatív hatásai, a szabályozás célja ezek kompenzálása. A szabályozótervezés a Takagi-Sugeno fuzzy leírásokhoz kidolgozott PDC szabályozási módszerre épül, amelyben lineáris mátrixegyenlőtlenségek (LMI-k) formájában írhatóak elő a szabályozási kritériumok. A TP alak tekinthető TS fuzzy rendszernek is, azaz a módszertan alkalmazható az előző fejezetben létrehozott TP típusú konvex politopikus modellekre.

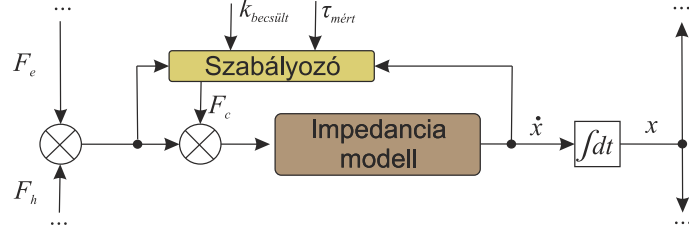
A fejezetben először az új állapotvisszacsatolási struktúrát ismertetem, majd bemutatom a Takagi-Sugeno fuzzy modell és a konvex politopikus modell kapcsolatát, és a felhasznált LMI-ket. Végül numerikus szimulációkkal vizsgálom meg a szabályozott rendszer tulajdonságait és a merevségmegfigyelő algoritmusok alkalmazhatóságát.

6.1. A szabályozási struktúra

A feladat előzménye konzulensem doktori disszertációjában [3] szerepel. A fő különbség, hogy ott a modell nem mérhető állapotváltozót is tartalmaz, ezért működéséhez adaptív állapotmegfigyelő is szükséges.

Az itt ismertetett modellben F_e és $\dot{x}$ értékek mérhetőek, és k , τ ismeretében a felhasznált állapotváltozók számíthatóak, ezért az új szabályozó tervezése és megvalósítása egyszerűbb lesz.

Az új állapotváltozók bevezetésékor az volt az elsődleges kérdés, hogy milyen alapjelhez szabályozzuk az értéküket. A sebességet értelemszerűen elegendő zérusba szabályozni, ezzel növelhetjük a rendszer stabilitását. Ha a rendszer pozícióját választanánk második állapotváltozónak, egyrészt a modellel nehéz lenne egy nemlineáris környezet leírása, másrészt nem egyértelmű a pozíció alapjelének meghatározása sem, mivel a telemanipulációs feladat célja nem egy pozíció elérése, hanem a biztonságos megfogás. Ezért a modellezés során a környezetben mért erőt választottam állapotváltozónak, ehhez a mester oldalon mért erő használható alapjelnek. Ekkor a cél $(F_h(t) + F_e(t - \tau_2))$ zérusba szabályozása, azaz az ideális, transzparens működés. Így a 6.1 ábrán látható szabályozási struktúrához jutunk.



6.1. ábra. Az új adaptív állapotviszacsatolási struktúra

6.2. LMI alapú szabályozástervezés

6.2.1. A konvex TP alak és a Takagi-Sugeno fuzzy leírás kapcsolata

Az ismerttetett konvex TP alakok könnyen átfogalmazhatóak Takagi-Sugeno fuzzy leírássá, így a TS-rendszerekre kidolgozott PDC szabályozási módszer is egyszerűen alkalmazható az elkészített konvex politopikus reprezentációhoz.

6.1. Definíció. *Folytonos idejű rendszer Takagi-Sugeno fuzzy leírása:*

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \sum_{r=1}^R h_r(\mathbf{p}(t))(\mathbf{A}_r \mathbf{x}(t) + \mathbf{B}_r \mathbf{u}(t)), \\ \mathbf{y}(t) &= \left(\sum_{r=1}^R h_r(\mathbf{p}(t)) \mathbf{C}_r \right) \mathbf{x}(t),\end{aligned}\tag{6.1}$$

ahol $\mathbf{A}_r$, $\mathbf{B}_r$, $\mathbf{C}_r$ mátrixok alkotják az r -edik fuzzy szabályt, és $h_r(\mathbf{p})$ az r -edik szabályhoz tartozó tagsági függvény. Minden $\mathbf{p} \in \Omega$ esetén a tagsági függvények pozitívak és összegük egy $\left(\sum_{r=1}^R h_r(\mathbf{p}) = 1 \right)$.

6.2. Definíció. *PDC fuzzy szabályozó:*

$$\mathbf{u}(t) = - \left(\sum_{r=1}^R h_r(\mathbf{p}(t)) \mathbf{F}_r \right) \mathbf{x}(t),$$

ahol $\mathbf{F}_r$ mátrixok alkotják az r -edik fuzzy szabályhoz tartozó visszacsatolást, és $h_r(\mathbf{p})$ az r -edik szabályhoz tartozó tagsági függvény.

Ekkor a visszacsatolt fuzzy rendszer:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \left(\sum_{i=1}^R \sum_{j=1}^R h_i(\mathbf{p}) h_j(\mathbf{p}) (\mathbf{A}_i - \mathbf{B}_i \mathbf{F}_j) \right) \mathbf{x}(t), \\ \mathbf{y}(t) &= \left(\sum_{i=1}^R h_i(\mathbf{p}) \mathbf{C}_i \right) \mathbf{x}(t)\end{aligned}\tag{6.2}$$

alakban írható fel.

A TP alak és a TS-rendszer kapcsolata:

- a TP modell csúcspontjai alkotják a fuzzy szabályokat,
- a súlyfüggvényekből képezhetőek a tagsági függvények: $h_r(\mathbf{p}) = \prod_{n=1}^N w_{n,i_n}(p_n)$,
- a fuzzy szabályokhoz tervezett visszacsatolásokból felépíthető a TP alakú szabályozó, így a szabályozási törvény:

$$u(t) = - \left(\sum_{r=1}^R h_r(\mathbf{p}) F_r \right) \mathbf{x}(t) = - \left(\mathcal{F} \boxtimes_{n=1}^N \mathbf{w}_n(p_n) \right) \mathbf{x}(t), \quad (6.3)$$

ahol $\mathcal{F}$ tenzor $I_1 \times \dots \times I_N \times k$, és k a visszacsatolt állapotváltozóinak száma.

6.2.2. Szabályozótervezés lineáris mátrix egyenlőtlenségekkel

A PDC szabályozótervezési módszer felhasznált elemeit Tanaka és Wang [28] munkája alapján ismertetem.

6.3. Definíció. A lineáris mátrix egyenlőtlenségek (LMI) a következő formában adhatók meg:

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}_0 + \sum_{m=1}^M x_m \mathbf{F}_m > 0, \quad (6.4)$$

ahol $\mathbf{x} = (x_1 \dots x_M)^T$ az ismeretlen változók, $\mathbf{F}_m \in \mathbb{R}^{n \times n}$ ($m = 0..M$) konstans, szimmetrikus mátrixok, és az egyenlőtlenség szimbóluma alatt itt $\mathbf{F}(\mathbf{x})$ pozitív definit tulajdonságát értjük.

A Lyapunov stabilitási kritériumoknak és különböző szabályozási módszereknek kidolgozták az LMI-s felírását. Az LMI rendszerek megoldására hatékony algoritmusok állnak rendelkezésre, mint a belső pontok módszere.

Lyapunov-stabil szabályozótervezés.

6.1. Tétel. A visszacsatolt fuzzy rendszer egyensúlyi állapotai Lyapunov stabilak tetszőleges tagsági függvényértékek esetén is, ha

$$-\mathbf{X}\mathbf{A}_i - \mathbf{A}_i^T \mathbf{X} + \mathbf{B}_i \mathbf{M}_i + \mathbf{M}_i^T \mathbf{B}_i^T > 0, \quad (6.5)$$

$$-\mathbf{X}(\mathbf{A}_i + \mathbf{A}_j) - (\mathbf{A}_i + \mathbf{A}_j)^T \mathbf{X} + \mathbf{B}_i \mathbf{M}_j + \mathbf{B}_j \mathbf{M}_i + \mathbf{M}_i^T \mathbf{B}_j^T + \mathbf{M}_j^T \mathbf{B}_i^T > 0,$$

minden $i = 1..R$ és $j = 2..(i-1)$ értékhez, ahol $\mathbf{X} > 0$ mátrix, és $\mathbf{M}_i = \mathbf{F}_i \mathbf{X}$.

Zajelnyomás. Tekintsük a következő, zajt tartalmazó fuzzy rendszert:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \sum_{r=1}^R h_r(\mathbf{p}) (\mathbf{A}_r \mathbf{x}(t) + \mathbf{B}_r \mathbf{u}(t) + \mathbf{E}_r \mathbf{v}(t)), \\ \mathbf{y}(t) &= \left(\sum_{r=1}^R h_r(\mathbf{p}) \mathbf{C}_r \right) \mathbf{x}(t), \\ \mathbf{u}(t) &= - \sum_{r=1}^R h_r(\mathbf{p}) \mathbf{F}_r \mathbf{x}(t), \end{aligned} \quad (6.6)$$

ahol $\mathbf{v}(t)$ a bemenet zaja.

Jó zajelnyomáshoz $\sup_{\|\mathbf{v}(t)\| \neq 0} \frac{\|\mathbf{y}(t)\|_2}{\|\mathbf{v}(t)\|_2} \leq \gamma$ kifejezésben γ értéket kell minimalizálnunk.

6.2. Tétel. Azokat a $\mathbf{F}_r$ visszacsatolásokat, amelyek stabilizálják a (fenti) fuzzy modellt, és minimalizálják γ -t, a következő optimalizálási LMI-k megoldásával kaphatjuk meg:

$$\min_{\mathbf{X}, \mathbf{M}_1 \dots \mathbf{M}_R} \gamma^2, \text{ ahol} \quad \begin{bmatrix} -\frac{1}{2}(\mathbf{X}(\mathbf{A}_i + \mathbf{A}_j) + (\mathbf{A}_i + \mathbf{A}_j)^T \mathbf{X} - & -\frac{1}{2}(\mathbf{E}_i + \mathbf{E}_j) & \frac{1}{2}\mathbf{X}(\mathbf{C}_i + \mathbf{C}_j)^T \\ -\mathbf{B}_i \mathbf{M}_j - \mathbf{B}_j \mathbf{M}_i - \mathbf{M}_i^T \mathbf{B}_j^T - \mathbf{M}_j^T \mathbf{B}_i^T) & \gamma^2 \mathbf{I} & 0 \\ -\frac{1}{2}(\mathbf{E}_i + \mathbf{E}_j)^T & 0 & \mathbf{I} \\ \frac{1}{2}(\mathbf{C}_i + \mathbf{C}_j) \mathbf{X} & 0 & \mathbf{I} \end{bmatrix} > 0. \quad (6.7)$$

Szabályozóerő korlátozása.

6.3. Tétel. Tegyük fel, hogy a kezdeti értékek korlátosak $\|\mathbf{x}(0)\|_2 \leq \phi$. A szabályozóerő $\|\mathbf{u}(t)\|_2 \leq \mu$ korlátos lesz minden $t \geq 0$ esetén, ha

$$\begin{bmatrix} \mathbf{X} & \mathbf{M}_i^T \\ \mathbf{M}_i & \mu^2 \end{bmatrix} \geq 0 \quad \text{és} \quad \phi^2 \mathbf{I} \leq \mathbf{X}. \quad (6.8)$$

További lehetőségek. A szabályozótervezési folyamat továbbfejleszthető korszerű, multiobjektív kritériumok H_2 , H_∞ előírásával, amelyeket [28]-ben szintén kidolgozták a PDC szabályozási struktúrához.

A stabilitási feltételt kevésbé konzeratív módon kezelő LMI rendszerek szélesebb körben alkalmazhatóak, ezekkel kiterjeszthető a szabályozótervezés megvalósíthatósága. Az utóbbi években A. Sala és C. Ariño mutatott be ilyen módszereket. [29, 30]

6.2.3. Szabályozótervezés

A szabályozótervezéshez felhasznált MATLAB kód a függelékben megtalálható, ez az előző részben bemutatott LMI rendszert használja fel. A megoldó a „B” és „C” modellhez nem talált megoldást. Az „A” és „D” modellhez kapott eredmény a nagy modellezési hibák miatt nem releváns. Az „E” modell vezetett megfelelő eredményre.

A szabályozási törvény:

$$F_c(t) = -(\mathcal{F} \times_1 \mathbf{w}_k(k) \times_2 \mathbf{w}_\vartheta(\vartheta)) \mathbf{x}(t), \quad (6.9)$$

ahol

- az állapotvektor: $\mathbf{x}(t) = [\dot{x}(t) \quad (F_e(t) + F_h(t))\sqrt{m_0/k(t)}]^T$,

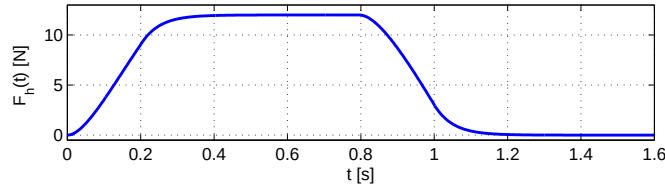
- $\mathcal{F}$: $3 \times 3 \times 2$ méretű, itt számszerűen:

$$\begin{aligned}\mathcal{F}_{i_k=1..3, i_\vartheta=1..3, 1} &= \begin{bmatrix} 242.7128 & -11.0196 & 224.2767 \\ 210.6137 & 220.5476 & 207.6171 \\ 238.4836 & 234.0487 & 226.7619 \end{bmatrix}, \\ \mathcal{F}_{i_k=1..3, i_\vartheta=1..3, 2} &= \begin{bmatrix} -13.6843 & 23.8301 & 11.4214 \\ 26.4360 & 15.5801 & 22.7101 \\ 9.5646 & 10.4713 & 11.5359 \end{bmatrix}.\end{aligned}\quad (6.10)$$

A következő részekben a szabályozóval elért eredményeket ismertetem.

6.3. A szabályozott rendszer vizsgálata numerikus szimulációkkal

A vizsgálatokat MATLAB Simulink környezetben végeztem. A mester oldalon minden esetben egy előre meghatározott profilt használunk, ami egy $\Delta t = 1.1[s]$ hosszúságú megfogási műveletnek felel meg. A szabályozott rendszer viselkedését először állandó (k, τ) paraméterezésekhez mutatom be, majd állandó és nemlineáris merevségű környezetek megfogását szimulálom változó időkézés mellett. Végül a merevségbecslőt is beillesztem a rendszerbe olyan módon, hogy a szabályozó csak a becsült merevségértékhez férjen hozzá – mint a későbbi megvalósítás során is.

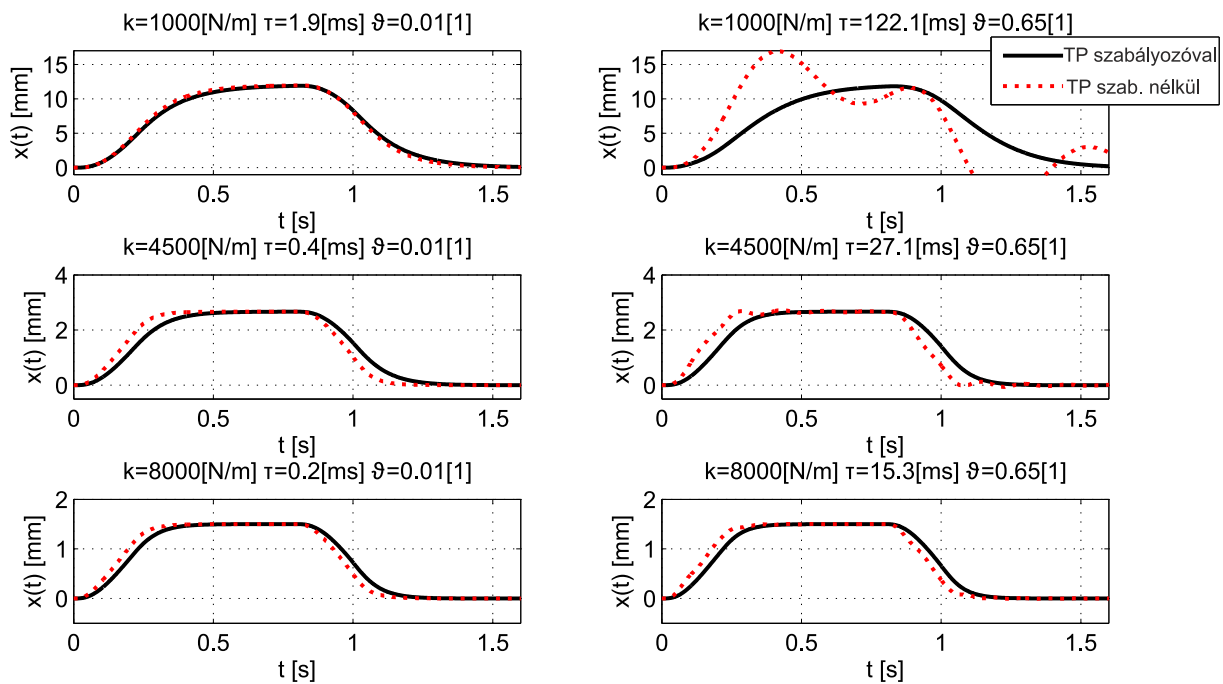


6.2. ábra. A vizsgálathoz használt operátorerő-profil

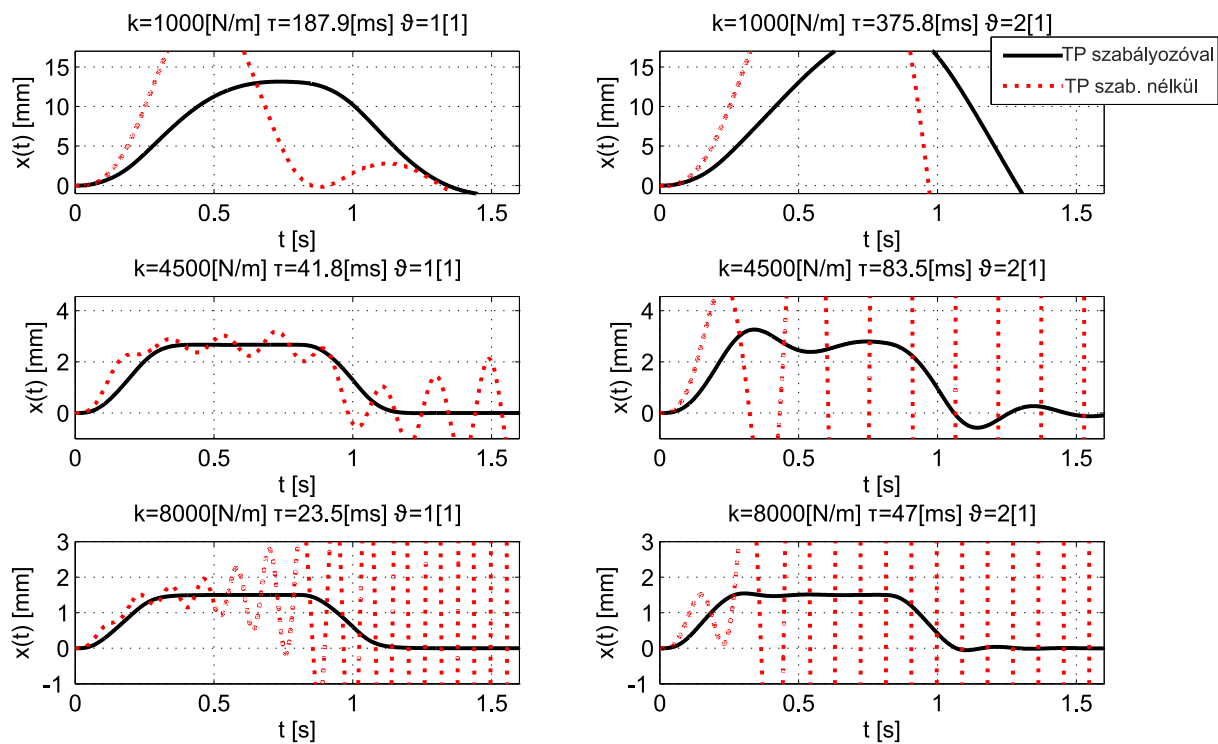
6.3.1. Konstans időkézés és merevség

Az eredményeket az 6.3 ábrán foglaltam össze. Az áttekinthetőség érdekében itt különböző (k, τ, ϑ) paraméterezésekhez csak a pozíció lefutásokat jelenítettem meg, minden grafikonon a TP szabályozóval és anélkül működő rendszerhez. (A folytonos vonal jelzi a szabályozott rendszerhez, a pontozott a szabályozás nélkülihez tartozik.) Mindkét ábrán soronként különböző $(1000[N/m], 4500[N/m]$ és $8000[N/m])$ merevségekhez és oszloponként növekvő relatív időkézésekhez mutatom be a szimulációk eredményét.

A 6.3(a) ábrán a tervezés során figyelembe vett paramétertér $(\Omega = (1000 \dots 8000[N/m]) \times (0..0.65[1]))$ belül vizsgálom a rendszert. Itt azonos merevséghez különböző időkézésekhez tartozó lefutások között nincs számottevő különbség, tehát a szabályozó jól működik. További



(a) A szabályozó hatása a transzformációs tér határain



(b) A szabályozó hatása az instabil tartományban

6.3. ábra. A szabályozott és a szabályozás nélküli rendszer összehasonlítása, különböző merevség és időkézés esetén

fontos jellemző, hogy a szabályozott lefutás alakja kis időkésekknél megegyezik az ideális esettel, azaz az időkésés nélküli szabályozatlan lefutással és nagyobb merevségek esetén közel áll ehhez az alakhoz.

A 6.3(b) ábrán már a transzformációs téren kívüli paraméterezéseket vizsgálom. Látható, hogy a szabályozott rendszer az időkésés növekedésével fokozatosan veszíti el a stabilitását, de még széles tartományban használható marad. Az eredmények azt is jelzik, hogy a modellezés kiterjesztésével az instabil tartományhoz is tervezhető stabilizáló szabályozás.

A szabályozatlan eseteken látható, hogy az időkésés növekedésével hogyan csökken a rendszer a csillapítása 6.3(a), majd válik instabillá 6.3(b).

6.3.2. Változó időkésés és merevség

Itt olyan szimulációk eredményét mutatom be, ahol az időkésés csak szakaszosan konstans, és a merevség is változik. Az 6.4 és 6.5 ábrán a környezeti merevség kontaktus jellegű: a felütközés előtt ($x < 0$ esetén) zérus, utána ($x \geq 0$ esetén) állandó, vagy nemlineáris karakterisztikájú.

Ezek az ábrák a szabályozott folyamat több jellemzőjét is megjelenítettem. A felhasznált időkésésprofil, a merevség értékeit, ezekből a relatív időkésés nagyságát, a pozíciót, a sebességet és az impedanciamodellekre ható erőket: a mester (F_h) és szolga oldali (F_e) erőt, a szabályozóerőt (F_c) és ezek összegét ($\sum F$). Valamint a pozíció grafikonját megjelenítettem a szabályozatlan rendszer lefutását is ugyanilyen paraméterek esetére.

Ezek az ábrák már megfigyelhető az időkéséses rendszer dinamikája és a szabályozó erő beavatkozása. Ha nem lenne időkésés a rendszerben, a környezeti erő és az operátor által kifejtett erő rövid idő alatt egyensúlyba kerülhetne. Viszont ha a környezeti erő csak időkéséssel csatolható vissza, jelentős túllendülés alakulhat ki romlik a rendszer csillapítottsága, akár be is gerjedhet a rendszer. A szabályozóerő a sebesség visszacsatolásával csillapítást ad a rendszerhez, és $F_e + F_h$ visszacsatolásával szintén lassíthatja a folyamatot, ha az nem transzparens.

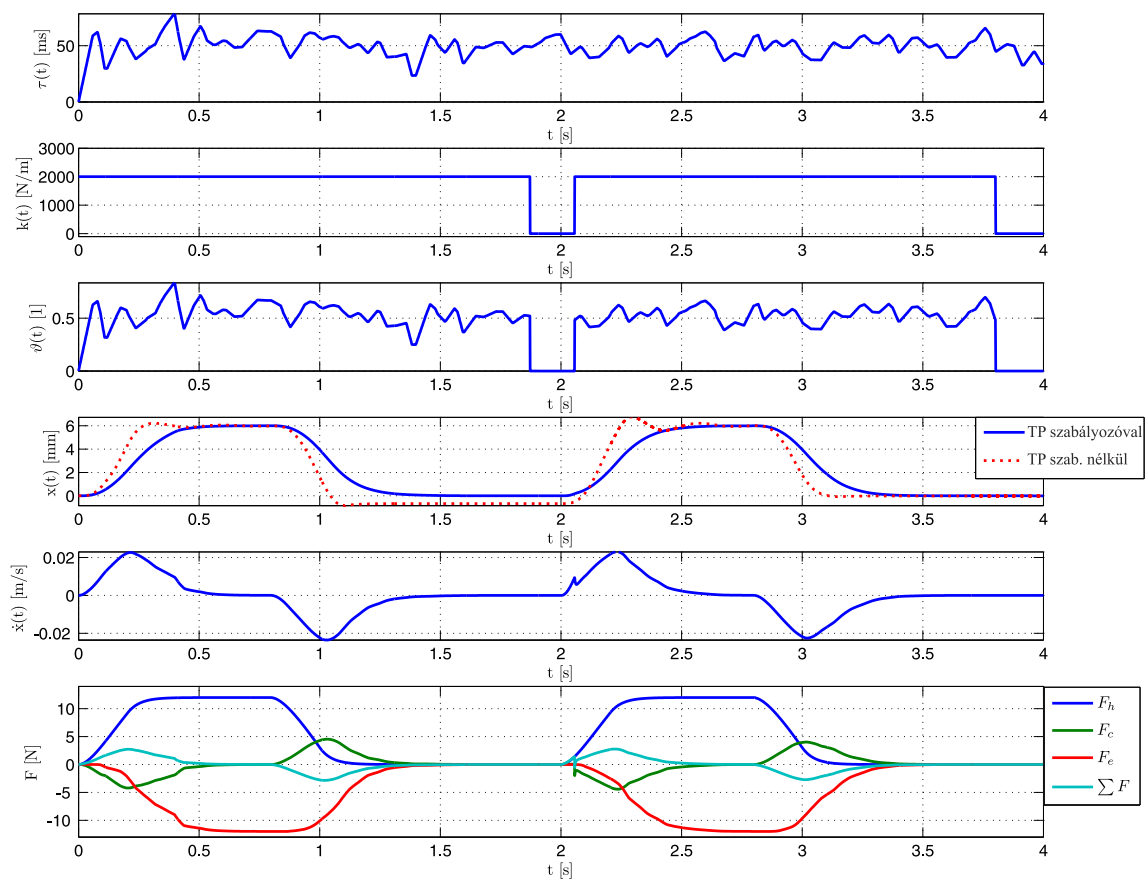
A 6.5(a) ábrán, a szabályozatlan rendszer lefutásán jól látható a gerjedés utáni ellökődés folyamata: a rendszer instabillá válik, begerjed, és csak az érintkezés (és a beavatkozóerő) megszűnte után nyugszik meg.

A vizsgálatok alapján kijelenthető, hogy a szabályozás változó paraméterek esetén is képes megőrizni a stabil működést.

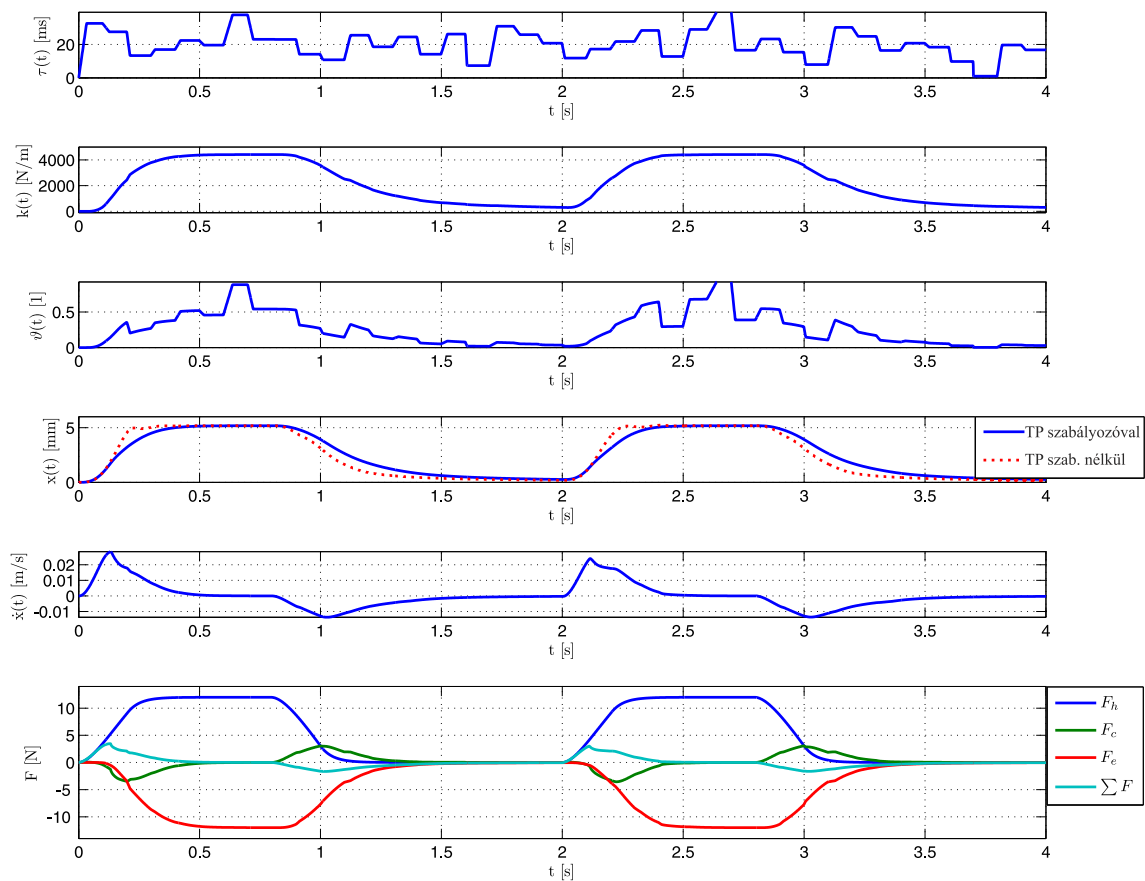
6.4. Merevségbecslő alkalmazása a szabályozókörben

Az eddigi szimulációkban a szabályozó az analitikusan ismert „ k ” értéket használja fel. A megvalósítás során ez csak a 4. fejezetben ismertetett merevségmegfigyelő eszközök alkalmazásával lehetséges.

Ezeknek az algoritmusoknak a hatékonyságát itt szimulációkkal vizsgálom meg. A vizsgálathoz felhasznált Simulink struktúra, a felhasznált TP szabályozó és a merevségbecslő blokkok

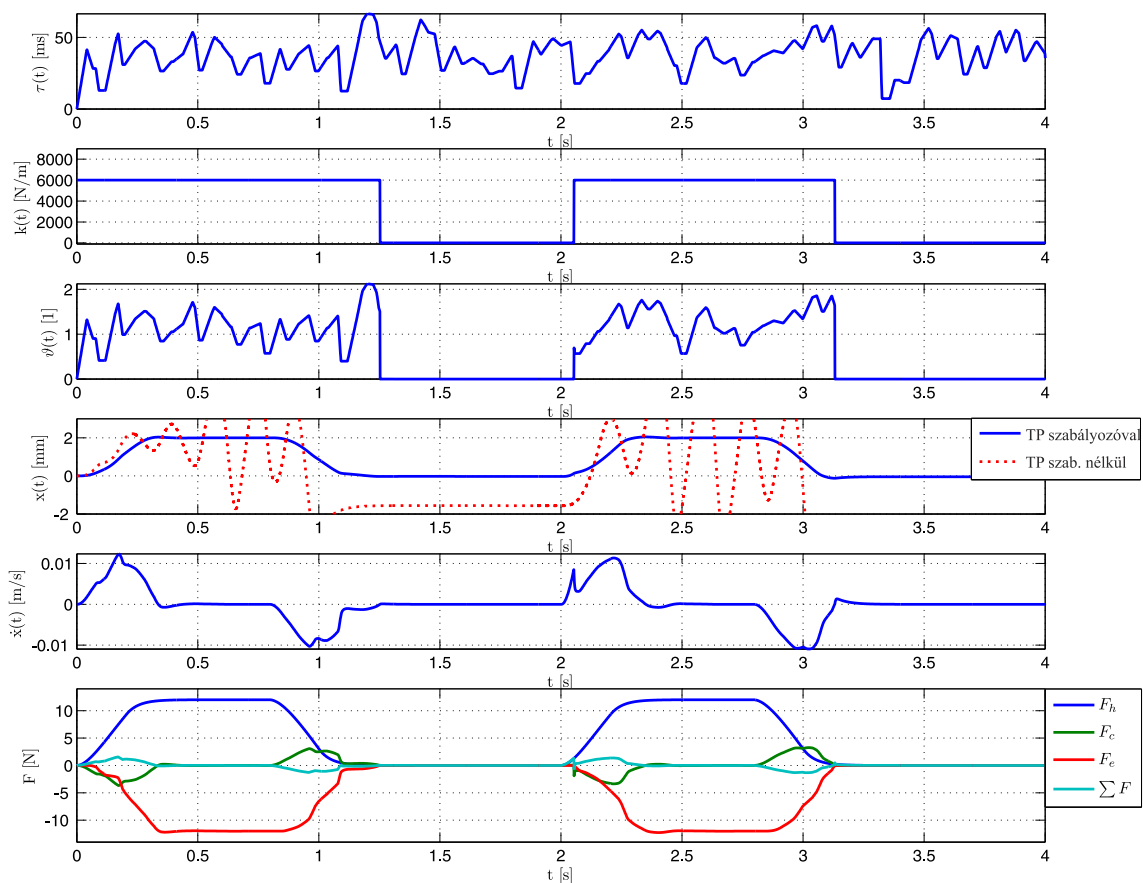


(a) Állandó merevségű környezet megfogása

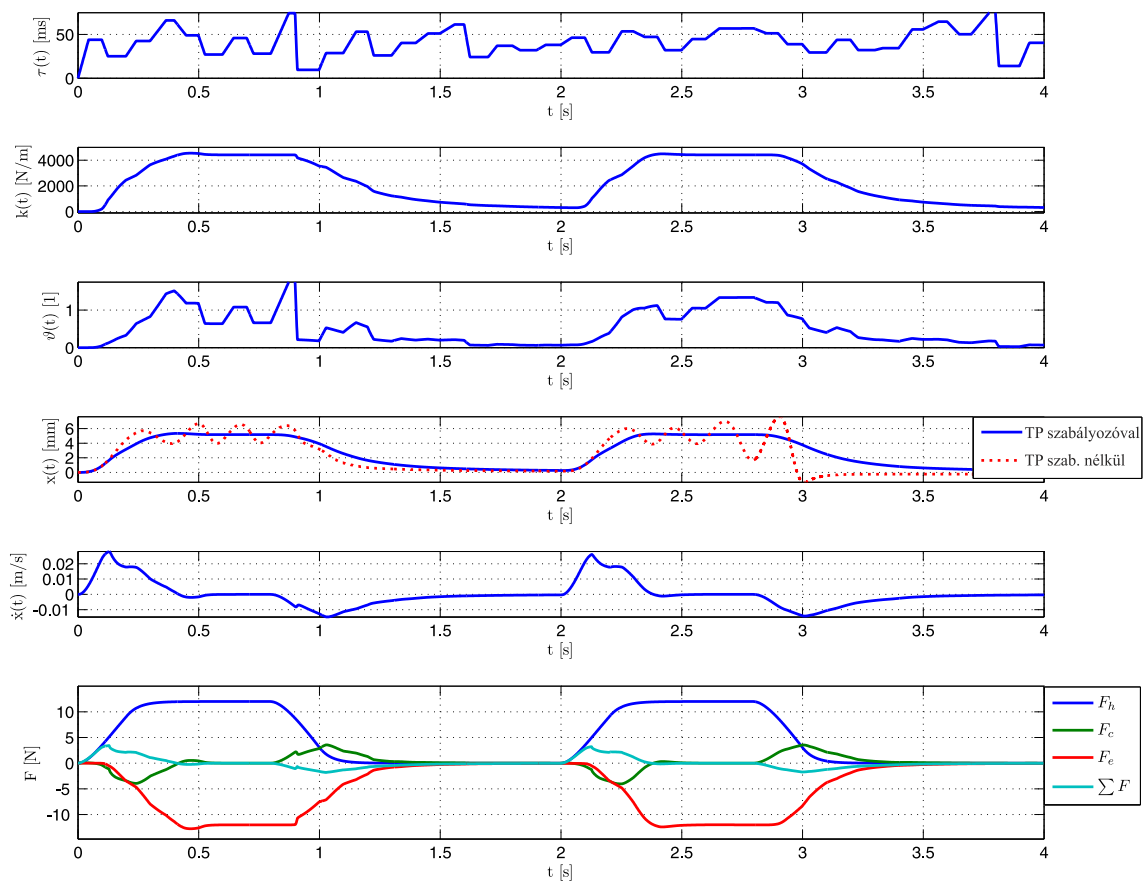


(b) Változó merevségű környezet megfogása

6.4. ábra. Változó merevség és időkéésés hatása - a vizsgált paramétertartományban



(a) Állandó merevségű környezet megfogása



(b) Változó merevségű környezet megfogása

6.5. ábra. Változó merevség és időkéésés hatása - az instabil tartományban

programjai a függelékben megtalálhatók.

(Megjegyzés: a fejezet ábráin a relatív időkésés grafikonjain annak valódi értékét jelenítettem meg, nem azt az értéket, amit a szabályozó a becsült $k_{estimated}$ érték alapján számol.)

6.4.1. Nem-parametrikus merevségmegfigyelő alkalmazása

Itt a 4.2 alfejezetben ismertetett merevségbecslő eljárás alkalmazását fogom bemutatni. A (4.4) megfigyelési törvényben α erősítési tényező optimalizálandó. Növelésével a megfigyelő gyorsítható, de ezzel együtt jár a zaj és a közelítés hibáinak kiemelése. A szimulációk során a szabályozó a merevségmegfigyelő eredményét használja fel, nem az analitikusan ismert merevségértéket. Az 6.6 ábrán látható, hogy a megfigyelő jól teljesít állandó és változó merevségű környezet esetén is $\alpha = 1500$ erősítési tényező esetén.

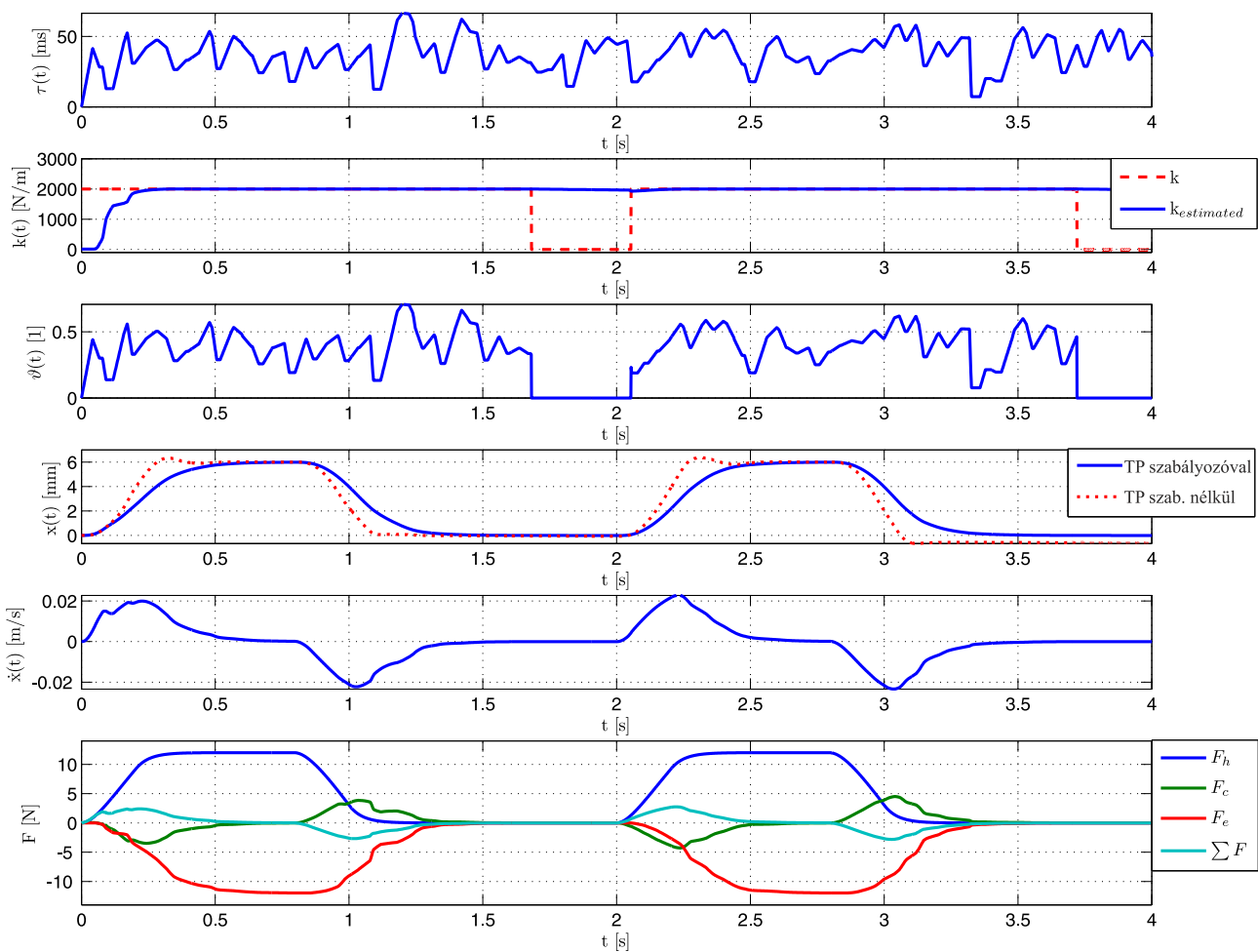
6.4.2. Parametrikus merevségmegfigyelő alkalmazása

A parametrikus merevségmegfigyelők fontos előnye lehet a feladatban, hogy különválnak a környezet paramétereinek megfigyelése és az aktuális pozícióhoz tartozó merevségérték meghatározása. Ezzel lehetőségünk nyílik arra is, hogy míg a paramétereket a szolga oldalon található megfigyelő határozza meg, a merevség értékét a mester oldalon számítjuk az impedanciamodell pozíciójához. Ekkor a merevség értéke gyakorlatilag időkésés nélkül felhasználható a szabályozás során.

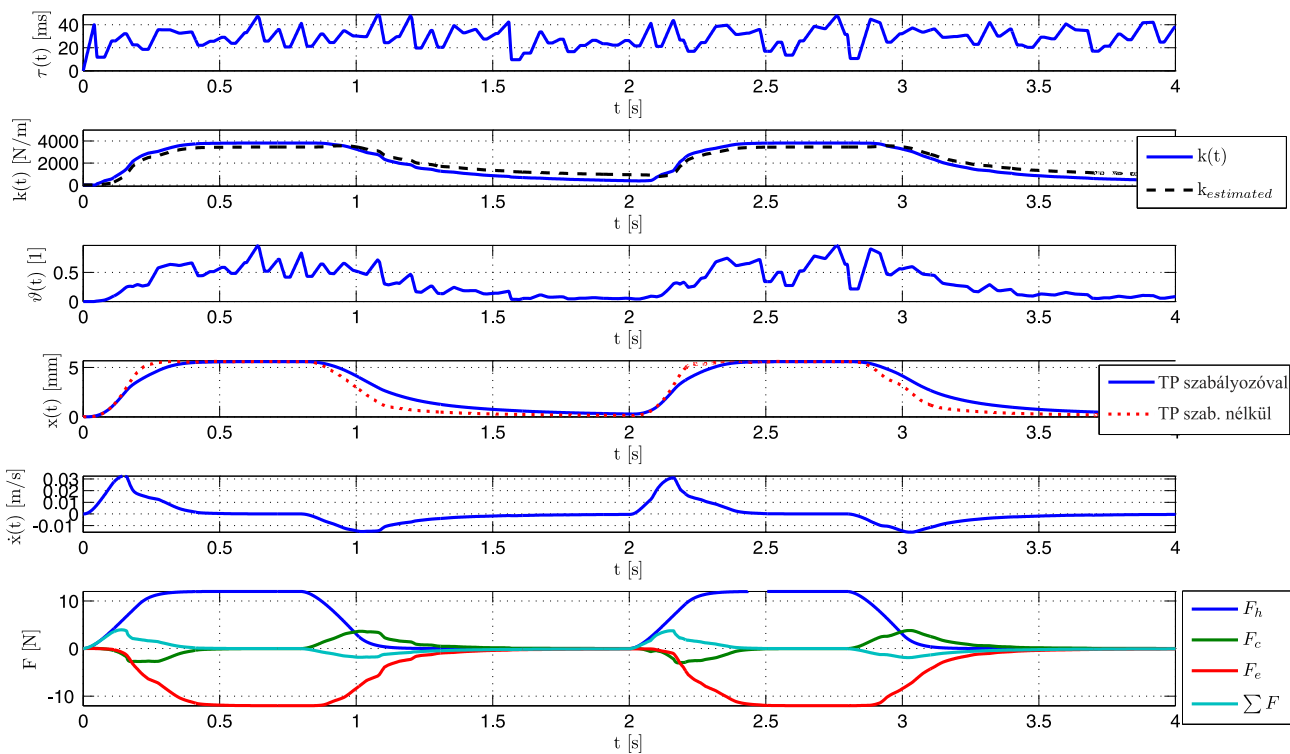
A 4.3 alfejezetben ismertetett módszer alkalmazása esetén a kezdeti értékeket és a β felejtési tényezők értékét kell optimalizálnunk. A működéshez szükséges az érintkezési pont mérése is, a megfigyelő csak a megfelelően eltoltt koordinátarendszerben működik.

Ezt a megfigyelőt is megvizsgáltam lineáris és nemlineáris környezetmodellhez, az eredmények a 6.7 ábrán láthatóak. Az eredmények szerint bár a megfigyelő a paraméterek beállása után jobb közelítést tud adni a nem-parametrikus megfigyelőnél, jóval lassabban állnak be a paraméterei, ami rontja a szabályozás megbízhatóságát.

A 4.4 alfejezetben ismertetett módszer előnye, hogy illeszteni lehet különböző környezetekhez, a vizsgálataim szerint megfelelő illesztés esetén jól működik az eljárás. Ebben a feladatban nincs specifikáció a környezetről, hanem az általánosan jó működést várjuk el, ezért ezt a módszert itt nem tudom alkalmazni.

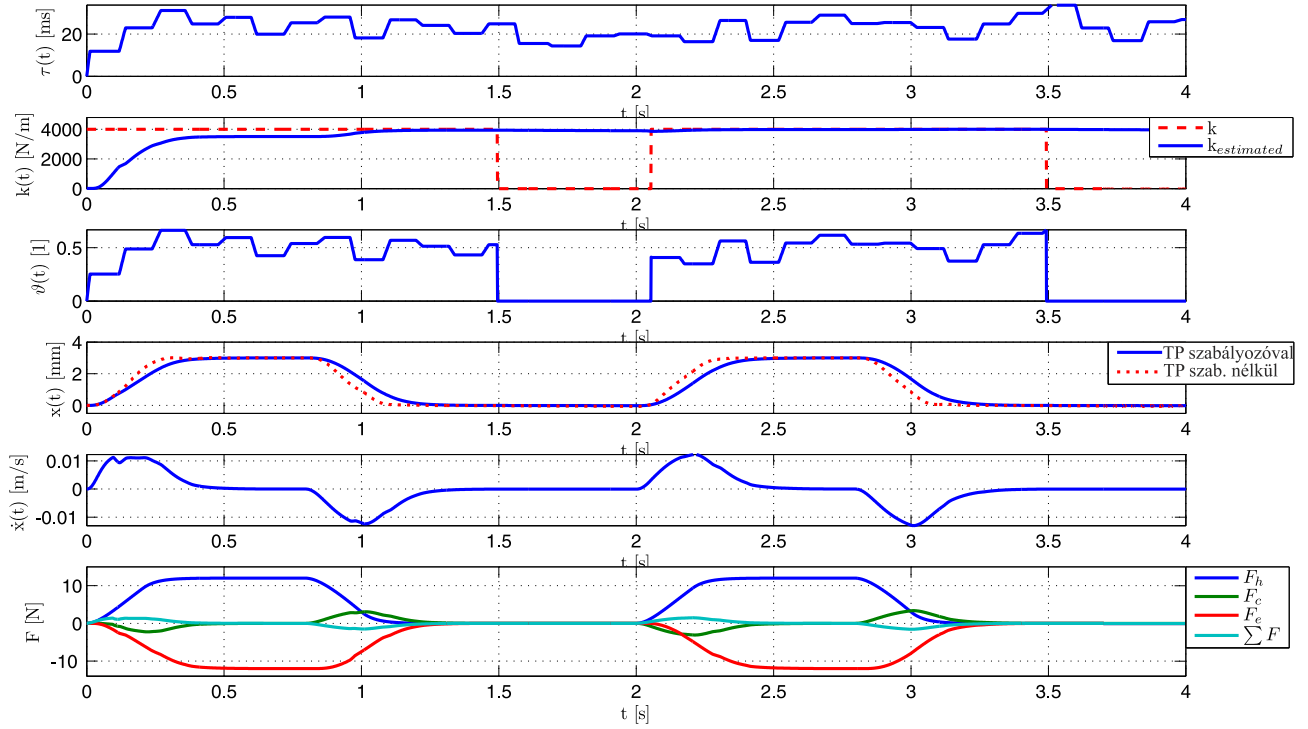


(a) Állandó merevségű környezet megfogása

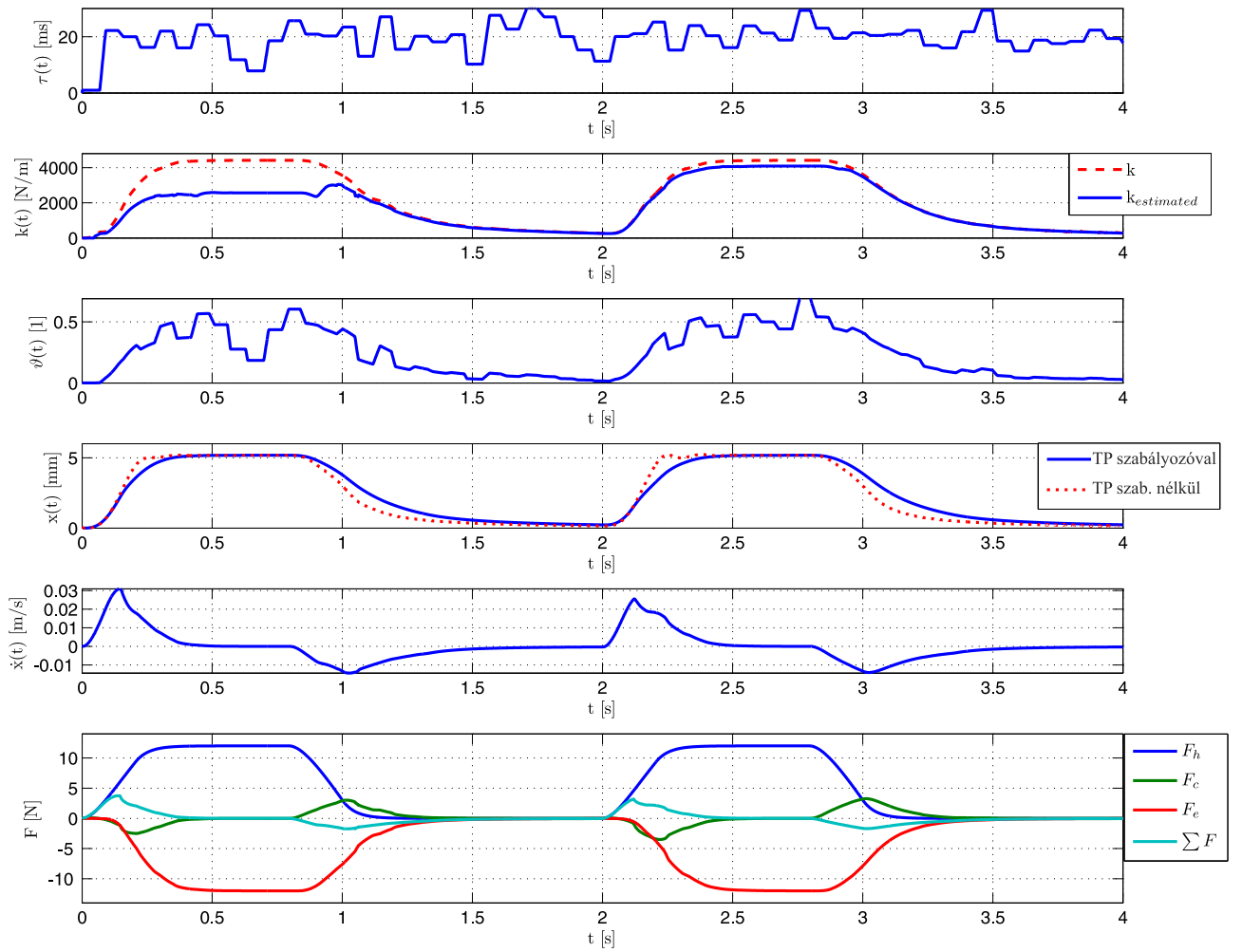


(b) Változó merevségű környezet megfogása

6.6. ábra. Nem-parametrikus merevségmegfigyelő alkalmazása, $\alpha = 1500$



(a) Állandó merevségű környezet megfogása



(b) Változó merevségű környezet megfogása

6.7. ábra. Parametrikus merevségmegfigyelő alkalmazása

6.5. Összefoglalás

A fejezetben bemutattam a szabályozótervezés folyamatát, és a szabályozott rendszer tulajdonságait. Az állandó paraméterezések vizsgálata során láthatóvá vált, hogy az adaptív szabályozó a teljes paramétertéren stabil és jó dinamikájú működést biztosít. A feladat jellegzetessége, hogy a paraméterek: az időkézés és a környezet merevsége is gyorsan változhat, de mérésük, megfigyelésük időt és olyan speciális módszereket igényel, mint a merevségmegfigyelő algoritmusok. A szimulációk szerint mindezek ellenére a szabályozó stabil megfogást tud biztosítani.

Láthatóvá váltak a különböző merevségmegfigyelő módszerek előnyei/hátrányai is. Egy későbbi kutatás témája lehet olyan hibrid megoldás vizsgálata, amely ötvözi a módszerek előnyeit. Ebben nem-parametrikus és parametrikus megfigyelőket is futtatnánk, de a nem-parametrikus módszer eredményét csak átmenetileg használnánk fel, amíg nagy a parametrikus megfigyelő hibája. Ezzel biztonságosan csökkenthető lenne a merevség értékének megfigyelési és kommunikációs időkézése.

7. fejezet

Kísérleti vizsgálat

A fejezetben a kísérleti megvalósítás során felhasznált hardver és szoftver eszközökkel foglalkozom. Bemutatom az elvégzett fejlesztéseket, majd a kísérleti vizsgálatok eredményeit.

7.1. A környezet bemutatása

A kísérletekhez a BME G épületében található descartes koordinátás Hirata robotokat használtam fel. A robotokat csak 1-1 tengely mentén működtettem, az egyik volt a mester-, a másik szolgál oldali eszköz. A robotok végperemére erőmérő cellák vannak felszerelve, a kísérleti elrendezés 7.1 ábrán látható.

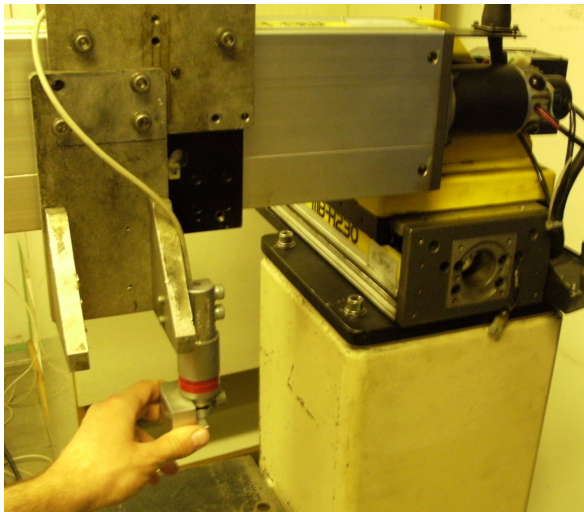
A felhasznált tengelyek főbb paraméterei:

- hajtás: DC motorral működtetett golyósorsó,
- lökethossz: $520[mm]$,
- menetemelkedés: $20[mm/ford]$,
- az inkrementális jeladó felbontása: $4000 [1/ford]$.

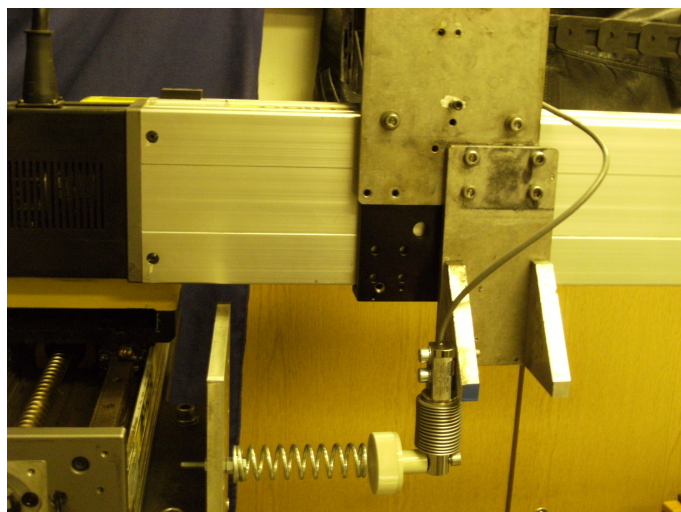
A vezérlőeszközöket Juhász András készítette diplomamunkája [31] során. A vezérlést és pozíciószabályozások beavatkozó jeleinek számítását egy PC végzi. A PC és a mikrokontrollerek közötti kommunikáció ISA buszon keresztül történik. A PWM jelet PIC18F4431 mikrovezérlővel állítjuk elő, ez optikai módon le van választva a motorba táplált teljesítményjeltől.

7.1.1. Vezérlőprogram

A TP szabályozóval kiegészített közös impedancia modell, a mester és szolgál oldali PID szabályozások, az erősűrőssel kiegészített merevségmegfigyelő és a kommunikációs időkésések szimulációja mind a PC-n fut, DOS operációs rendszeren, Turbo C 2.0 nyelven írt programon.



(a) A kézzel mozgatott mester eszköz



(b) A rugót megfogó szolgálka eszköz

7.1. ábra. A kísérleti elrendezés

Az eredeti vezérlőprogramot Galambos Péter és Juhász András készítette a RESCUER projekt kapcsán, ezen Fliszár Dániel diplomamunkájában [32] vizsgálta a TP szabályozást. Ez utóbbi program szolgált a fejlesztések alapjául. Az új programot terjedelmi okok miatt csak a függelékben ismertettem, itt csak röviden bemutatok fontosabb tapasztalatokat és fejlesztéseket.

A program jelentős részét a perifériák (monitor, erőmérő cellák, PIC mikrokontrollerek) inicializálása és kezelése teszi ki. Ezeket (a hiba ellenőrzési és kezelési funkciók miatt) nem módosítottam. A TP szabályozót különálló programmodulként valósítottam meg olyan formában, hogy a felhasznált interface egyszerűen lehetővé tegye különböző szabályozók közötti könnyű váltást. (Ezzel a régi [32]-ben bemutatott szabályozó is felhasználható az új főprogrammal.) A főprogram könnyebb áttekinthetőségének érdekében a matematikai függvények (mozgóátlag szűrő, időkésés szimulációjához készített *struct* típus, stb) szintén külön állományba kerültek.

TP szabályozó modul

Egy külön forrásállomány tartalmazza a szabályozó működéséhez szükséges változókat, egy inicializáló függvényt, amely fájlokból beolvassa ezek értékét, és a TP szabályozást végző függvényt. A függvény négy skalár bemenettel rendelkezik: az időkésés, a környezeti merevség, az impedanciamodell sebessége és a mért erők összege alapján határozza meg a szabályozóerőt, ami a rutin kimenete.

Első lépésként interpolációval meghatározza a megadott merevséghez és a relatív időkéséshez tartozó súlyfüggvényvektorokat. Majd elvégezve a tenzorszorzásokat megkapja az erősítéseket, amelyekkel a szabályozóerő már számítható.

A modulban alkalmazott általános interface-szel, egy külön állományban megadható a régi (csak az időkésésre adaptív, adaptív állapotmegfigyelőt is tartalmazó) TP szabályozás is. Így a

szabályozók közötti váltás megoldható a főprogramba beillesztett állományok váltásával.

Matematikai modul

A modul néhány egyszerű matematikai függvény kódja mellett, a mozgóátlag szűrő és az időkésés szimuláció kódjait tartalmazza. Az egyszerű kezelhetőség érdekében mindkét eljárásban elegendő egy inicializáló függvényt meghívni, ami beállítja a kezdeti értékeket és lefoglalja a megfelelő méretű memóriaterületet. A használatuk is egy-egy függvény meghívásával történik, bemenetei: a kezelendő objektum címe, az új elem és egyéb paraméterek, a függvények kimenete a késleltetett/szűrt jel értéke.

Főprogram

A főprogram két részre tagolódik: a `main()` eljárásra és a `interrupt()` függvényre. A `main()` eljárás gyakorlatilag három részre osztható.

1. A rendszer (perifériák, programmodulok, változók) inicializálása.
2. Egy végtelen ciklus, amely alatt a rendszer számára engedélyezett az interruptok végrehajtása. Ez a ciklus ESC billentyűvel szakítható meg, egyetlen feladata kettő másodpercenként kijelezni a fontosabb adatokat a monitoron.
3. A futás közben naplózott adatokat file-ba írjuk, a mikrokontrollereket alaphelyzetbe állítjuk, és felszabadítjuk a dinamikus memóriát.

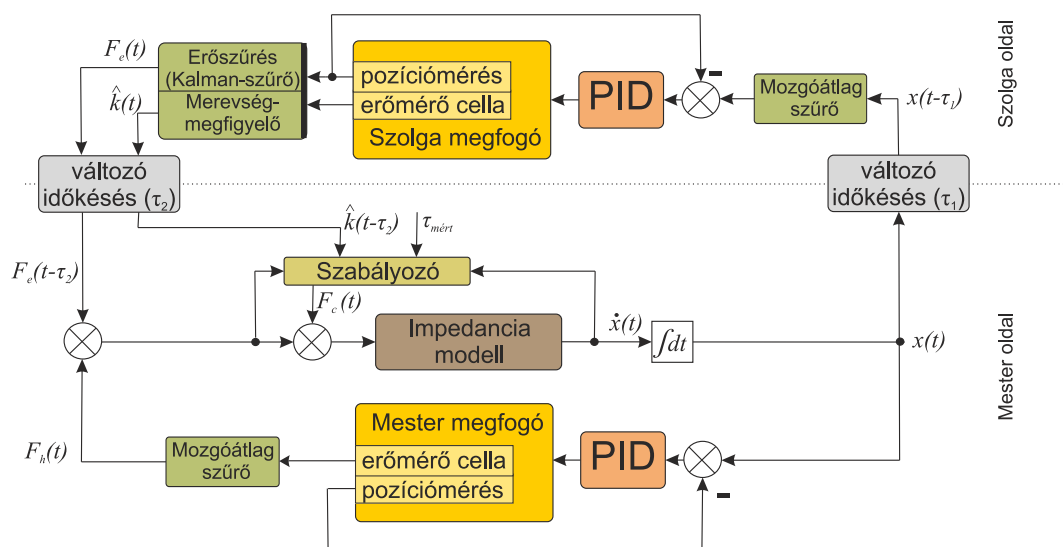
A mikrokontrollerek 1 miliszekundumonként generálnak megszakítást. Ekkor a program `interrupt()` függvénye hajtódik végre. Először beolvassuk a perifériák adatait. Az erősztűrés és a szolga oldali merevségmegfigyelő szintén itt kerültek implementálásra. A mester oldali erősztűrésre egy mozgóátlag szűrőt használunk, a szolga oldali erő szűréséhez Kalman-szűrőt terveztem, a programban tesztelési célra elérhető a mozgóátlag szűrő és szűrés nélküli futtatás is.

A definíciók között beállítható a programfuttatás módja. Ezen módok egyike a telemanipulációs mód, a függvény ekkor végrehajtja a megfelelő rutint. Az interrupt függvény zárásaként a beavatkozó jelek értékét továbbítjuk a mikrokontrollereknek.

Telemanipulációs rutin. A rutin meglehetősen kompakt, a kommentekkel együtt is csak 70 soros eljárás. Működése a következőképpen írható le:

1. Az előírt időkésések beolvasása és számítása a megadott profilból.
2. A szolga oldali szűrt erő és becsült merevségérték késleltetése.
3. A beállításoktól függően a szabályozóerő számítása TP szabályozó rutinnal.

4. A közös impedancia modell gyorsulásának számítása, a mért erők és a kiszámított szabályozóerő alapján. Téglány integrátorokkal az elvárt sebesség és pozíció értékének meghatározása.
5. A szolga oldali pozíció értékének előírásához a pozícióérték késleltetése és szűrése mozgóátlag szűrővel.
6. A mester és szolga oldalon mért és előírt pozíciók alapján kiszámítjuk a PID szabályozók beavatkozó jelét.



7.2. ábra. A kísérleti környezetben megvalósított irányítási rendszer vázlata

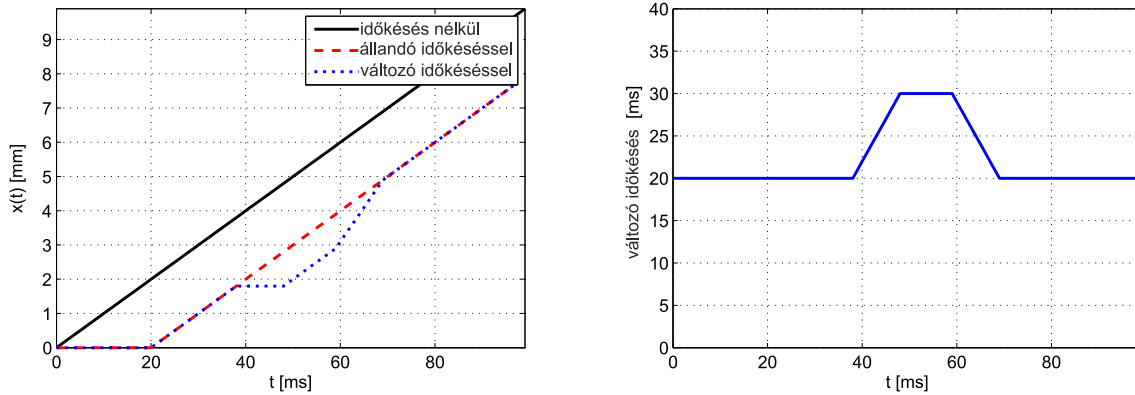
7.2. A rendszer szűrési feladatai

A fejlesztés során megvizsgáltam milyen szűrések szükségesek a kísérleti környezetben, a következő alfejezetben az alkalmazott módszereket ismertetem.

7.2.1. A késleltetett pozícióérték szűrése – mozgóátlag szűrő

A pozícióérték váltakozó nagyságú késleltetésével egy zavaró jelenség figyelhető meg a berendezésen. A szolga oldali megfogó szabálytalan időközönként akadozni kezd: hirtelen megtorpan, majd megindul, a jelenlegi roboton mindezt hangos koppanásszerű hangokkal kísérve. A jelenség megértéséhez nézzük a 7.3 ábrát. Egy egyenletes mozgás idő-elmodulás grafikonját, és ennek különböző módon késleltetett változatait jelenítettem meg. Látható, hogy állandó időkézés esetén megmarad az egyenletes mozgás, de változó időkézés esetén már nem. Az időkézés növekedése azt jelenti, hogy a szolga oldal nem kap új pozícióértéket, azaz az eszköz megáll. Ha ismét állandóvá válik az időkézés, folytatódik a mozgás, és az időkézés csökkenésével a mozgás felgyorsul.

Megjegyzendő, hogy ha a változó időkézéses csatornán pozíció helyett sebességértékeket küldünk, nem alakulnának ki a kopogást okozó gyorsuláscsúcsok. Azonban a pozíció használata biztonságosabb, mivel ekkor a kommunikáció megszakadása esetén megáll a szolgálatszám.



7.3. ábra. A változó időkézés hatása a pozíciójelre

A változó időkézés nagy gyorsulásértékeket ír elő a rendszernek, ezek elsimítására egy mozgóátlag szűrőt illesztettem a rendszerbe. A kísérletek szerint minél nagyobb sebességgel mozgatjuk a megfogót, annál nagyobb ablak szükséges a jó minőségű manipulációhoz. A megfogásra jellemző kis sebességekhez elegendő lehet egy $10ms$ -os ablak, a teszteléshez $30ms$ -os ablakot használtam.

7.2.2. A mester oldali erő szűrése – mozgóátlag szűrő

Az erőmérések zajos jelet adnak, ezért egy $20ms$ -os ablakú mozgóátlag szűrőt helyeztem el a mester oldalon.

7.2.3. A szolga oldali erő szűrése – Kalman-szűrő

A szolga oldalon a szűrés kiemelten fontos, mivel a merevségmegfigyelő az erő deriváltját használja fel. Fontos tényező, hogy a szűrés lehetőleg ne okozzon itt további időkézést, ezért a szolga oldalon nem egyszerű mozgóátlag szűrőt használtam.

A merevségbecslő megfelelő működése esetén a környezet modellje jól ismert, ennek a felhasználásával Kalman-szűrő is illeszthető a rendszerhez. A szolga oldal folytonosidejű modellje:

$$\begin{aligned}\dot{F}_e(t) &= 0 \cdot F_e(t) - k(t)\dot{x}_e(t) + v, \\ F_e^{mert}(t) &= 1 \cdot F_e(t) + z,\end{aligned}\tag{7.1}$$

ahol v , z zajok. Ebből a diszkrétidejű modell:

$$\begin{aligned}F_e(T) &= F_e(T-1) - k(T-1)\Delta x_e(T) + v, \\ F_e^{mert}(T) &= 1 \cdot F_e(T) + z.\end{aligned}\tag{7.2}$$

ahol $\Delta x_e = x_e(T) - x_e(T - 1)$, és T a diszkrét időt jelképezi.

A Kalman-szűrő alkalmazásával $F_e^{mért}$, Δx_e és rendszer modellje (k) alapján olyan becslés adható F_e értékére, amely a jelről eltávolíthatja a zajokat és nem okoz akkora időkésést, mint egy hasonló tulajdonságú mozgóátlag szűrő. A kísérleti környezetbe implementált Kalman-szűrő és merevségmegfigyelő összefüggései:

$$\begin{aligned}\bar{F}_e(T) &= \hat{F}_e(T - 1) - \hat{k}(T - 1)\Delta x_e, \\ \hat{F}_e(T) &= \bar{F}_e(T) + G \cdot [F_e^{mért}(T) - \bar{F}_e(T)], \\ \hat{k}(T) &= \hat{k}(T - 1) - \alpha \cdot [\hat{k}(T - 1)\Delta x_e + \Delta \hat{F}_e] \cdot \text{sign}(\Delta x_e),\end{aligned}\tag{7.3}$$

ahol

- $\Delta x_e = x_e(T) - x_e(T - 1)$,
- $\Delta \hat{F}_e = \hat{F}_e(T) - \hat{F}_e(T - 1)$,
- $G = 0.045$ és $\alpha = 1500$.

A fejlesztés során a környezetben mért erő és elmozdulás adatokat MATLAB környezetbe importáltam. Simulink segítségével teszteltem a különböző szűrési eljárások és megfigyelők együttes alkalmazásának lehetőségeit és teljesítményét. Az eredményeket a kísérleti környezetben is vizsgáltam, az ismertetett paraméterek ennek a vizsgálatnak az eredményei.

A Kalman-szűrő G együtthatójának növelésével az erőn megjelenik a zaj, csökkentésével időkésés alakul ki a méréskor. A merevségmegfigyelő $\alpha = 1500$ paramétere kis merevségtartományban zajos értéket ad, de ez nem zavarja a rendszer működését, viszont lehetővé teszi, hogy gyorsan (ca. $20 - 30ms$) alatt beálljon a merevségmegfigyelő $6 - 7000N/m$ merevségű környezethez is.

7.3. Eredmények

A következő oldalakon a rendszer működését szemléltetem. A megfogási műveletet TP szabályozó alkalmazásával a 7.4(a), a szabályozás nélkül esetet a 7.4(b) ábrán jelenítettem meg. (Megjegyzés: a kísérleti környezetben $x < 0$ irányban történik a megfogás, ekkor $F_e > 0$.) Látható, hogy a szabályozatlan esetben kisebb lengés jelentkezik, de nehéz a két esetet összehasonlítani, mert nem tudunk azonos megfogási folyamatot végezni. Ezért a további vizsgálatok során a mester oldali erőhöz a mért erő helyett egy előre meghatározott profilt használtam fel.

A vizsgálatokhoz 2140 , 6300 és $6800 [N/m]$ merevségű rugókat használtam. Mindegyik rugóhoz $\vartheta \sim 0.6$ és 1.2 relatív időkéséseket, és ezek esetén a szabályozott és szabályozatlan lefutásokat vizsgáltam. Az eredmények a 7.5–7.10 ábrán láthatóak. A szabályozó nélküli lefutások során rendszerint instabillá válik a rendszer, emiatt több esetben idő előtt leállítottam a vezérlést.

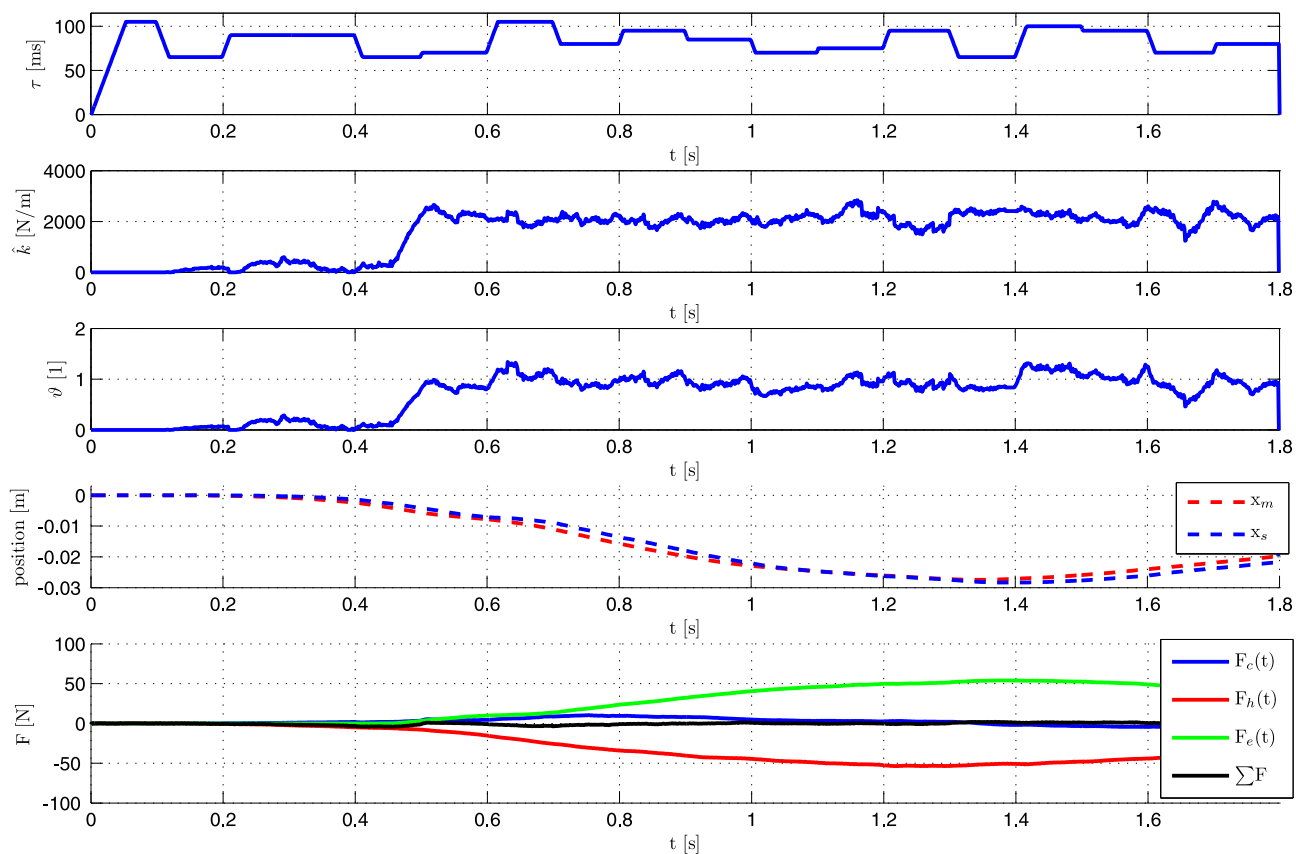
A szabályozott esetekben látható, hogy a felhasznált erősítési tényezőkkel merevség értéke gyors felfutás után körülbelül 15%-os túllendüléssel áll be a valódi értékhez. A feladatban nagyon fontos, hogy minél előbb beálljon a merevségmegfigyelő, ez a túllendülés pedig csak biztonságosabbá teszi a tranziens szakaszt. A pozíció grafikonjain is látható a szabályozott folyamatok kezdetén egy meglódulás (ami a szabályozatlan esetekben begerjedéssel folytatódik), aminek az oka, hogy még nem áll a szabályozó rendelkezésére a környezeti merevség értéke. Látható, hogy a szabályozás képes megfogni ezt, sőt a kézzel mozgatott lassabb folyamatok esetén nem is látható a jelenség.

A szabályozatlan eseteken látható, hogy a gyors lengések közben történő felütközések hatására belenghet a megfigyelt merevség értéke is, és a megfogások végén látható a megfogó elpattanása a rugótól.

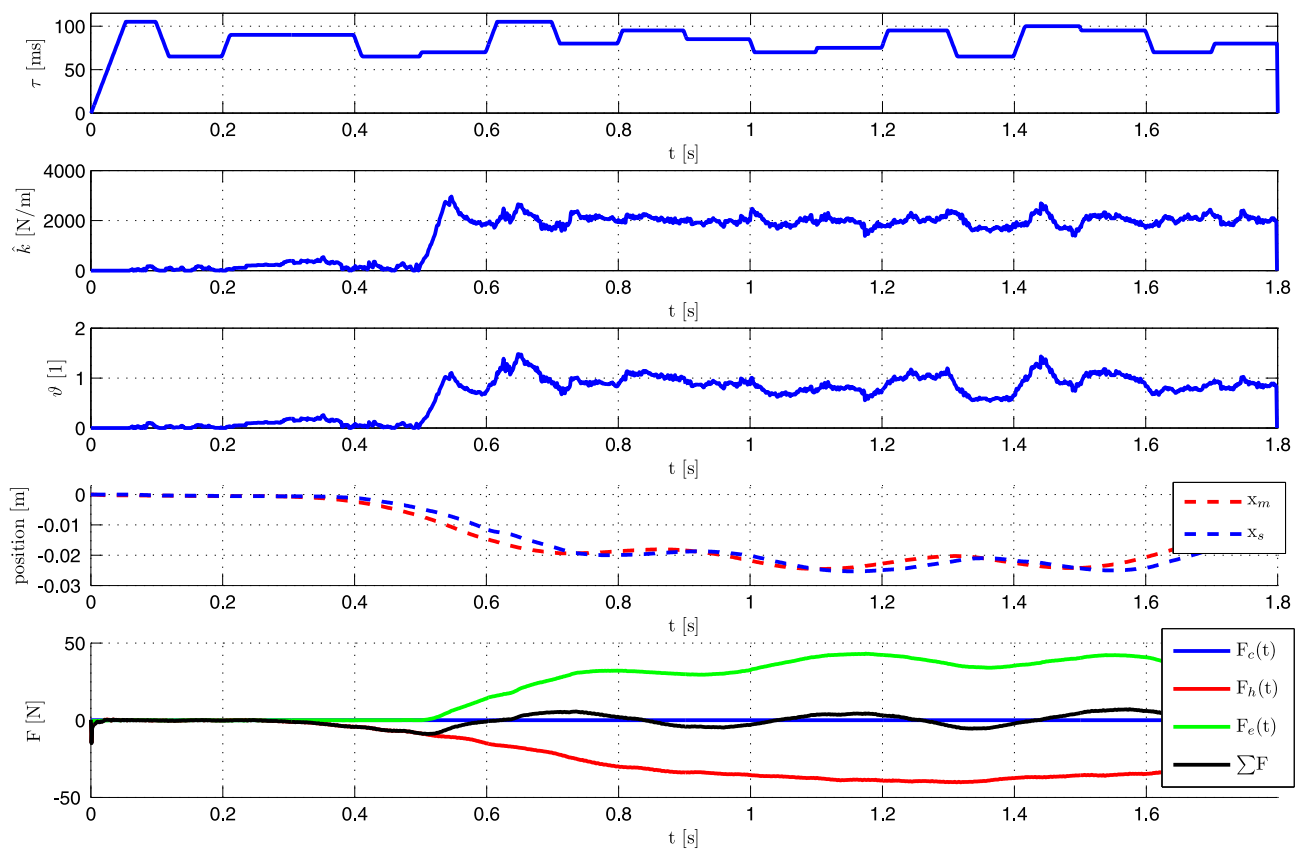
7.4. Összefoglalás

A fejezetben bemutattam a kísérleti környezet felépítését és a vezérlőprogram működését, amit a függelékben mellékeltem is. Részletesen ismertettem milyen szűrésekkel kell kiegészíteni a szabályozási struktúrát. A szolga oldalon a merevségbecslő eljárást Kalman szűrővel egészítettem ki. A szűrő és a merevségbecslő paramétereit numerikus szimulációk, majd kísérleti vizsgálatok során optimalizáltam.

Az eredményeket a naplózott adatokkal szemléltettem, bemutatva a folyamatokat a szabályozó alkalmazása nélküli esetre is. Látható hogy a szabályozás és a merevségmegfigyelő a jó dinamikai tulajdonságok megőrzése mellett stabil lefutást képes biztosítani.

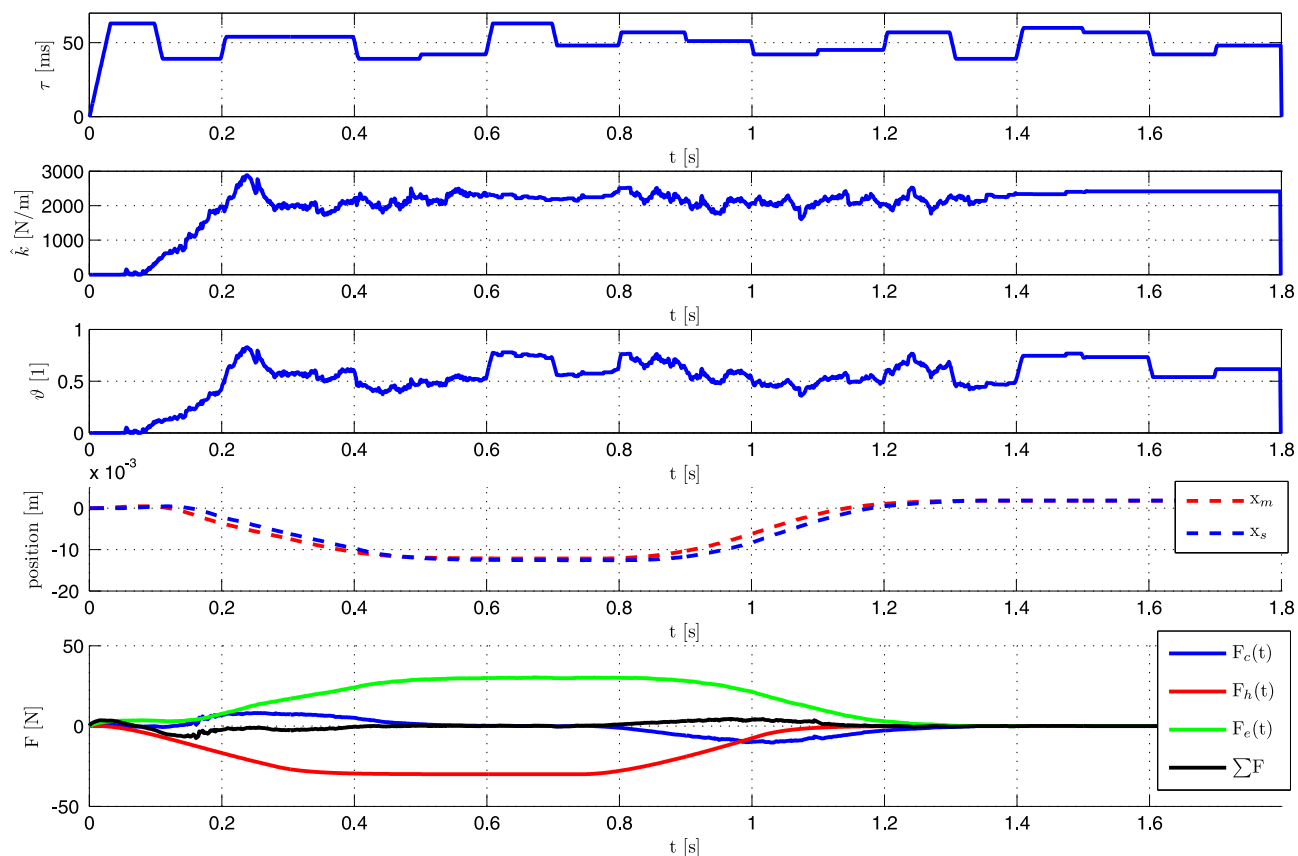


(a) TP szabályozó alkalmazásával

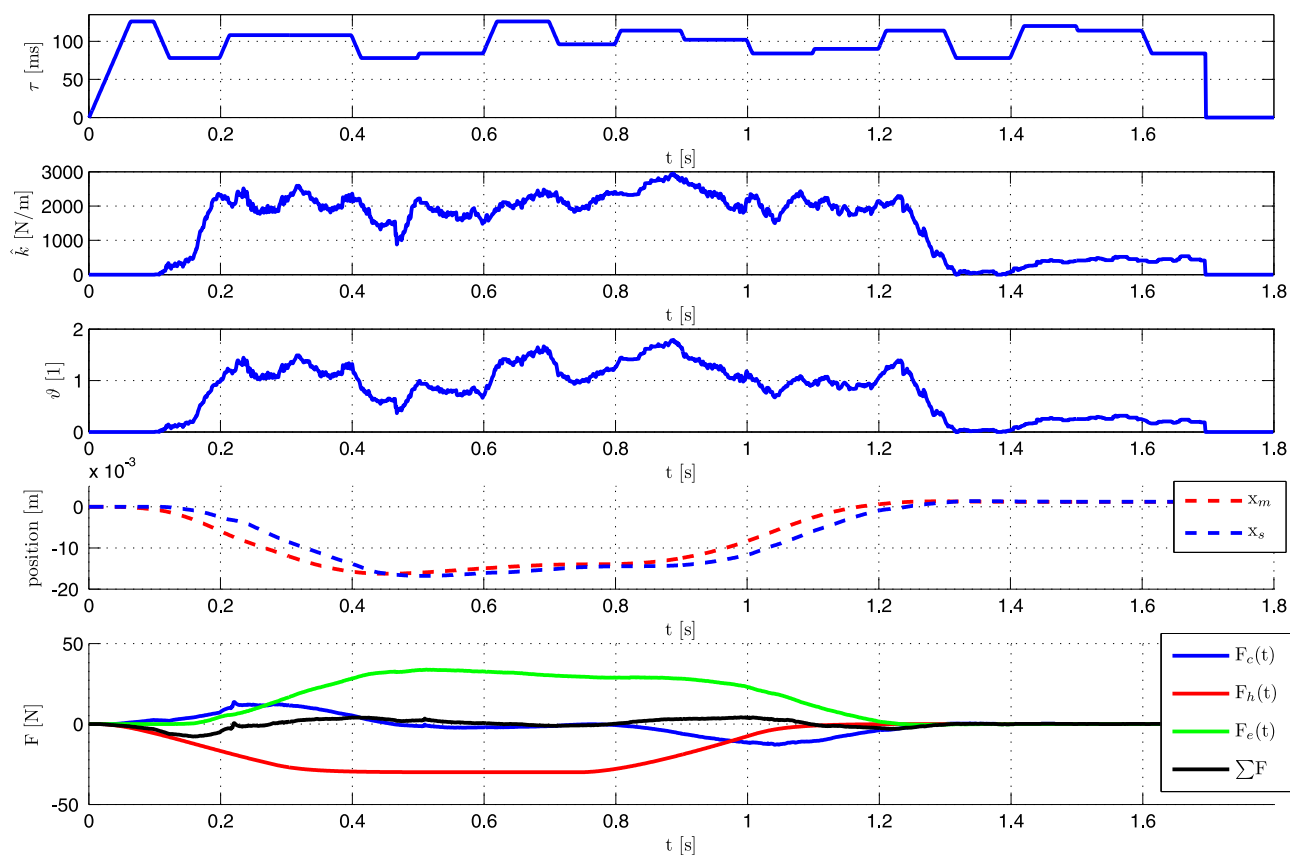


(b) TP szabályozó nélkül

7.4. ábra. 2140 [N/m] merevségű rugó megfogása kézzel mozgatott mesterszkoz esetén

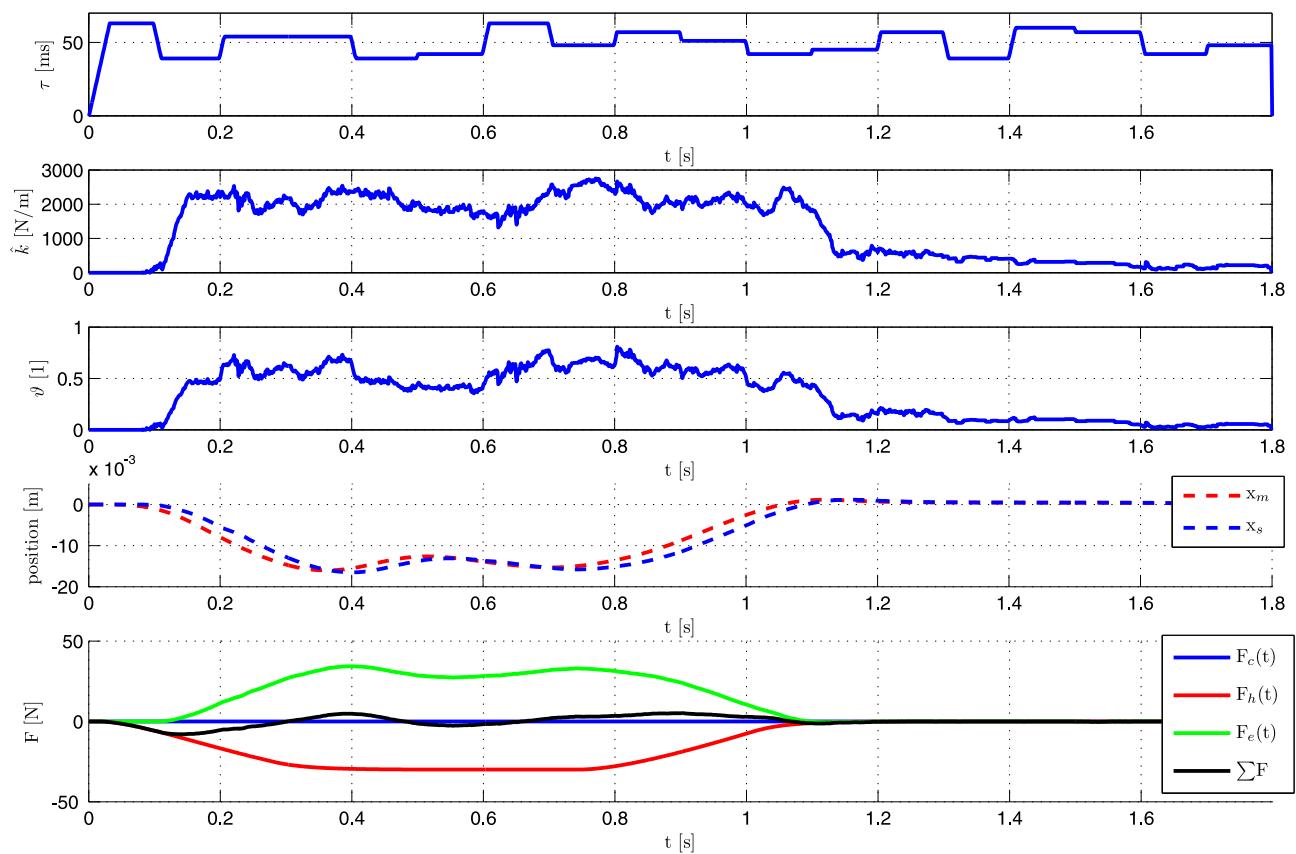


(a) $\tau \sim 55\text{ms}$

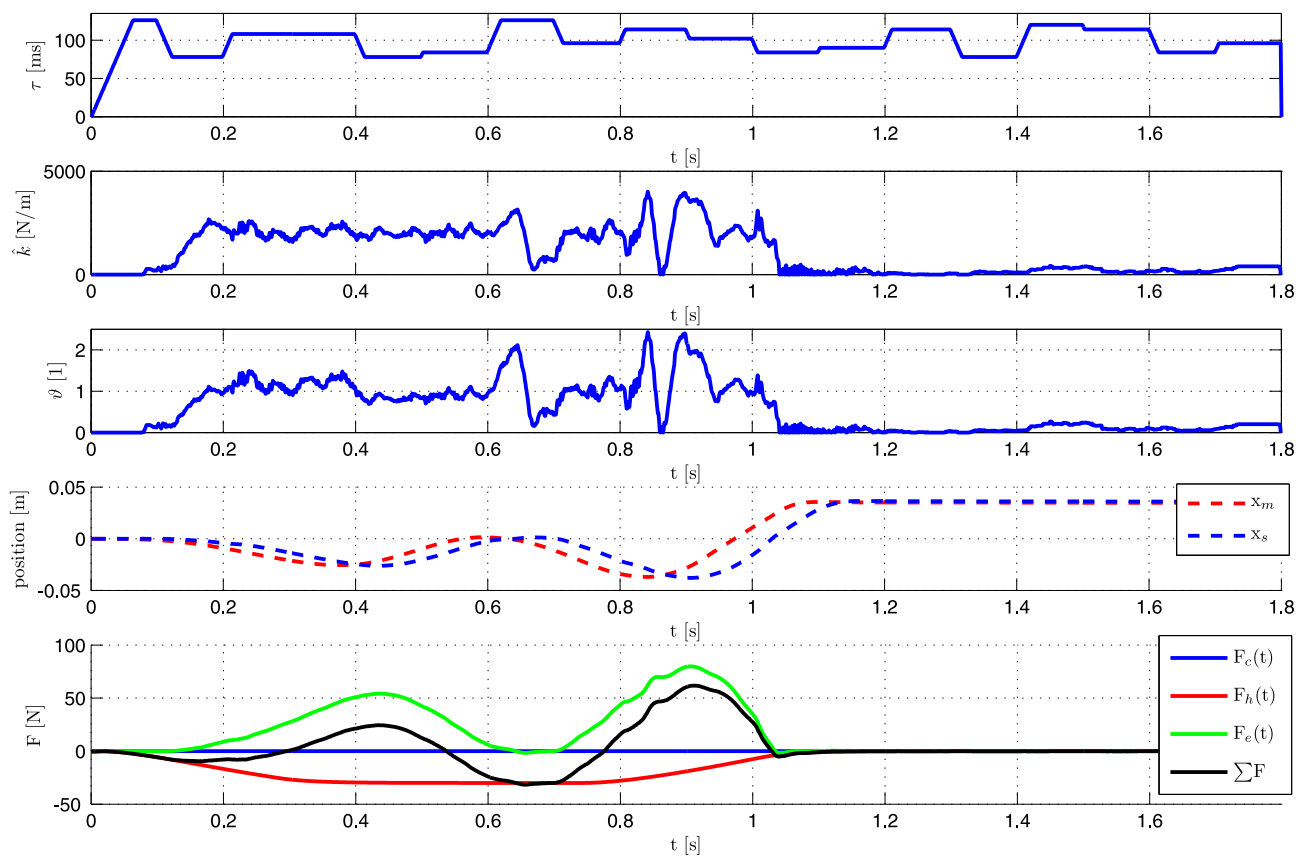


(b) $\tau \sim 110\text{ms}$

7.5. ábra. 2140 [N/m] merevségű rugó megfogása TP szabályozó működése esetén

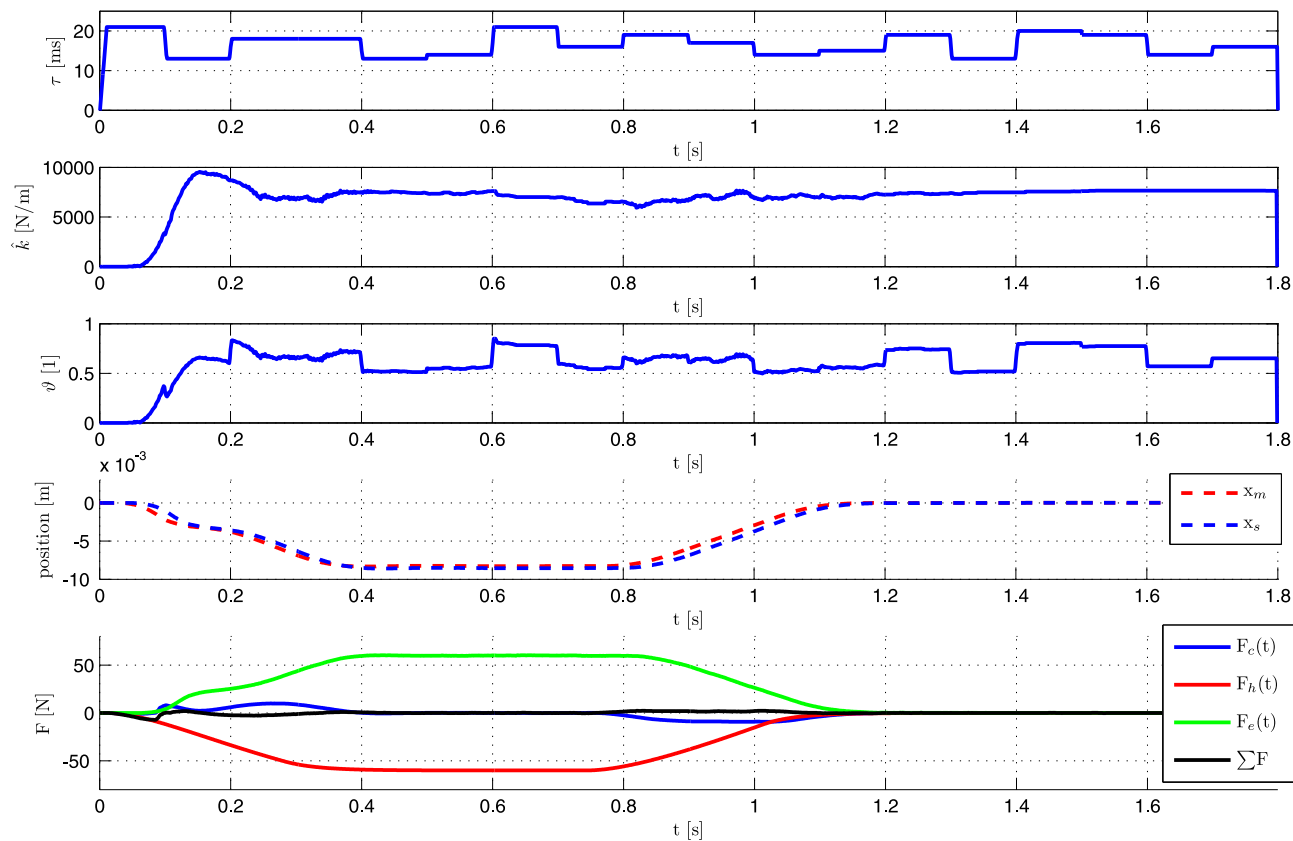


(a) $\tau \sim 55\text{ms}$

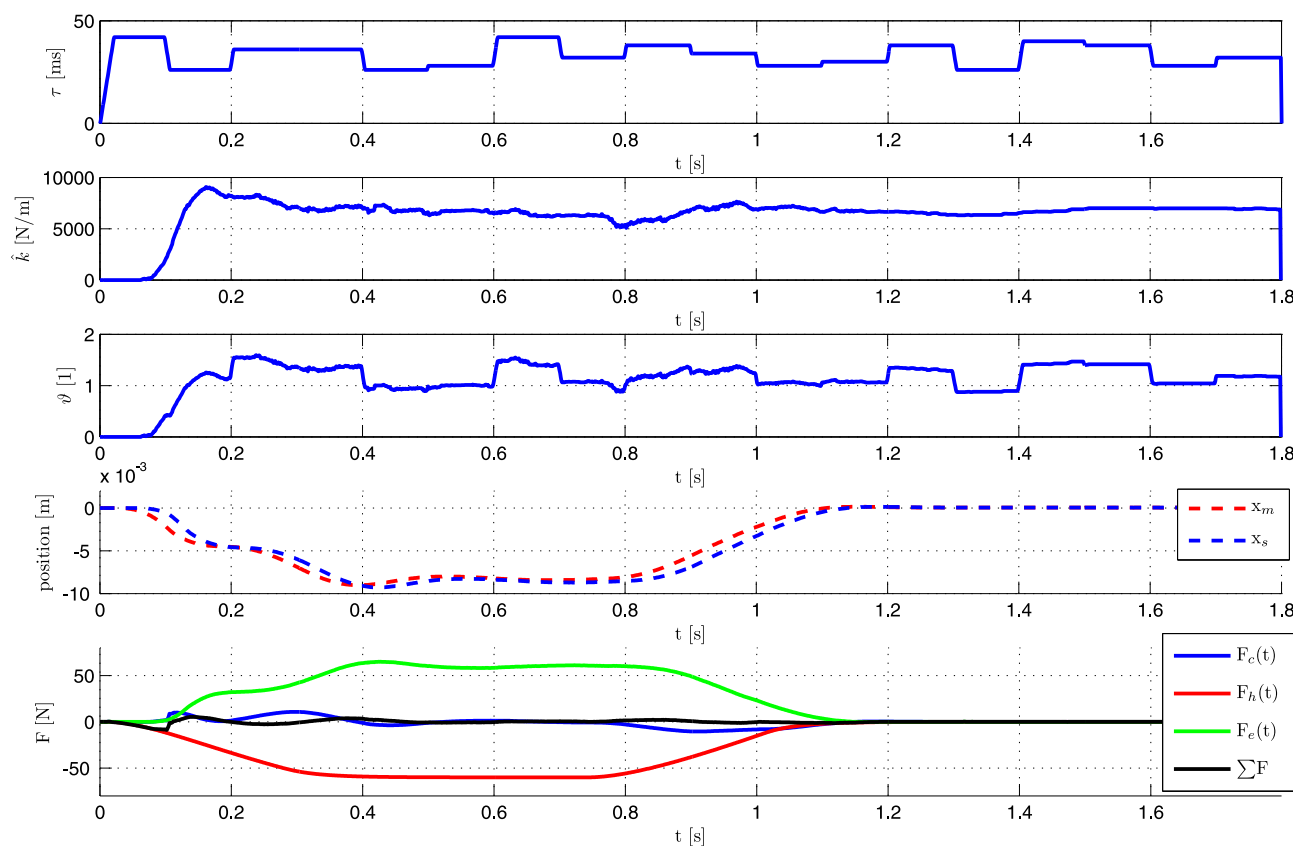


(b) $\tau \sim 110\text{ms}$

7.6. ábra. 2140 [N/m] merevségű rugó megfogása TP szabályozó nélkül

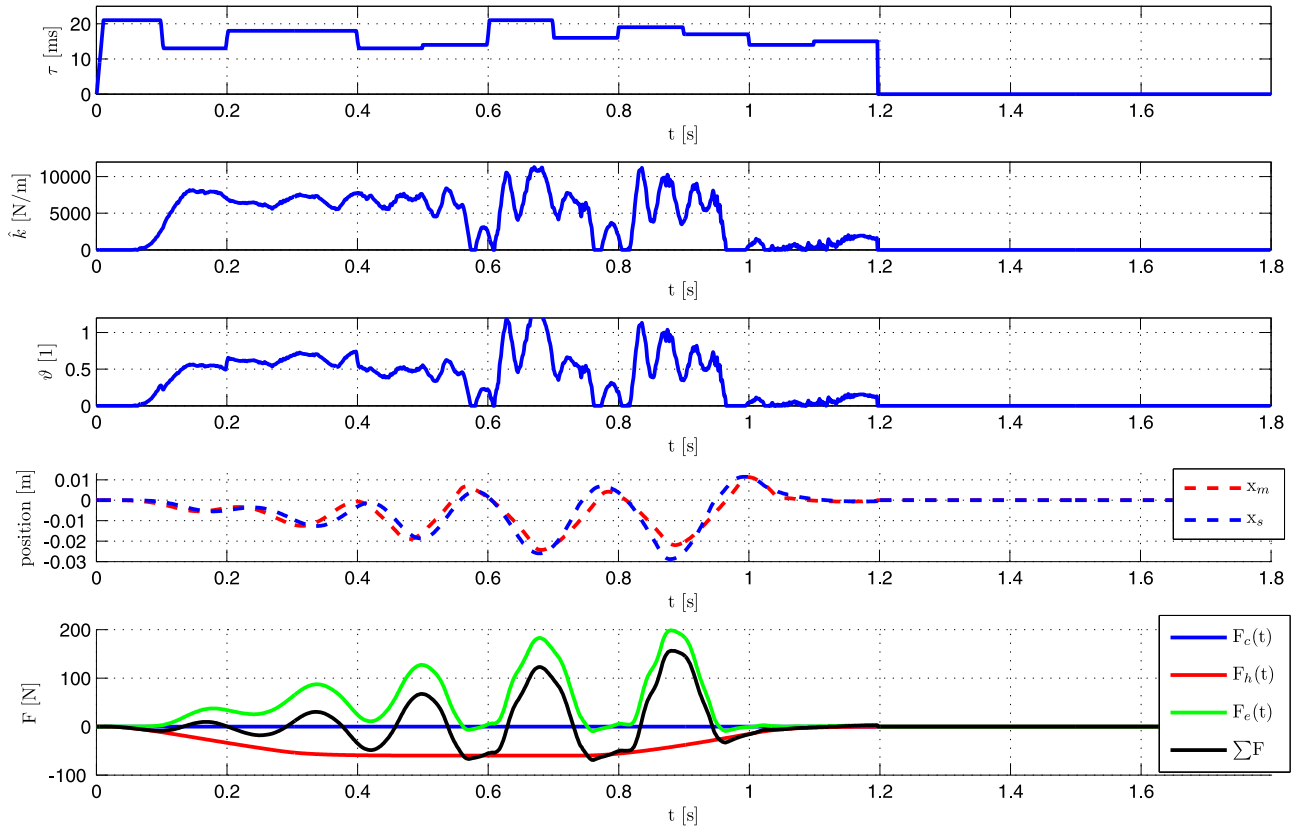


(a) $\tau \sim 18\text{ms}$

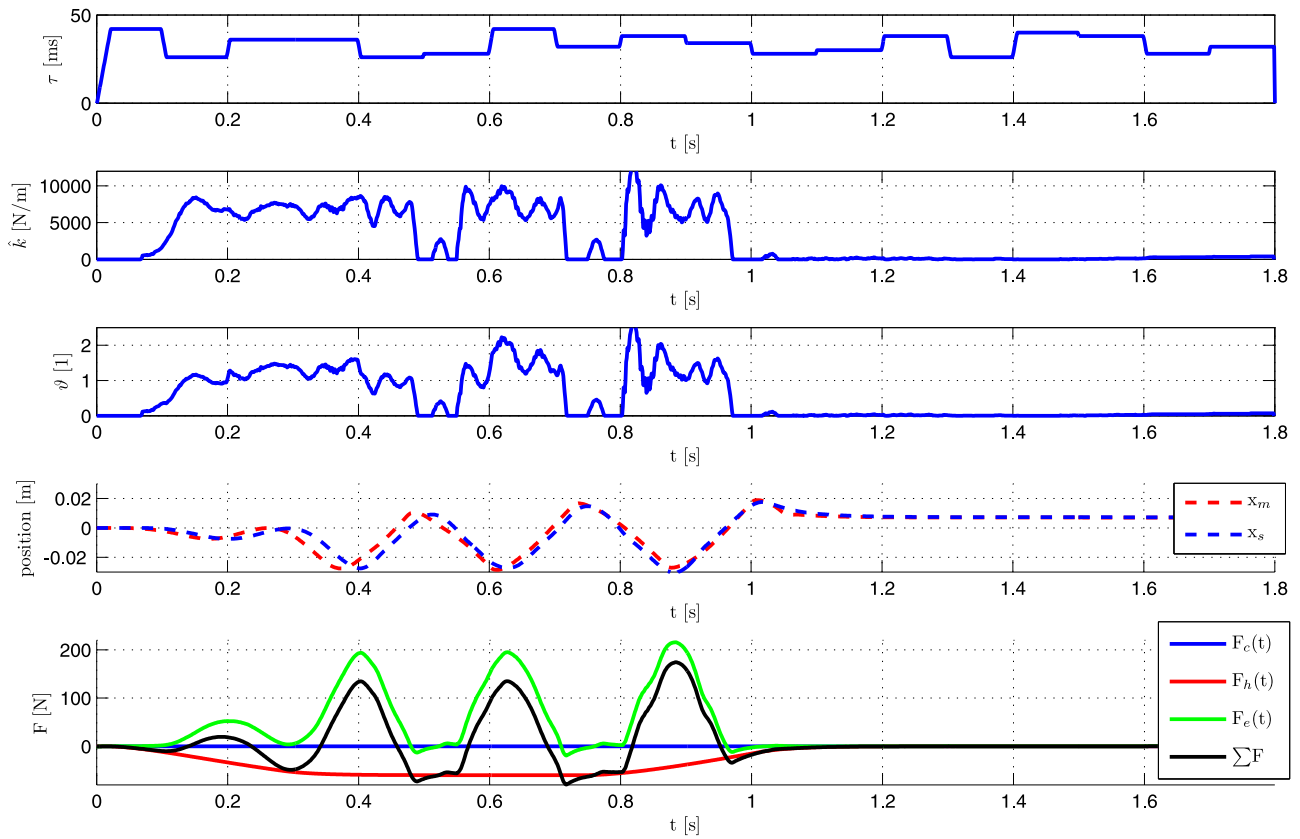


(b) $\tau \sim 35\text{ms}$

7.7. ábra. 6800 [N/m] merevségű rugó megfogása TP szabályozó működése esetén

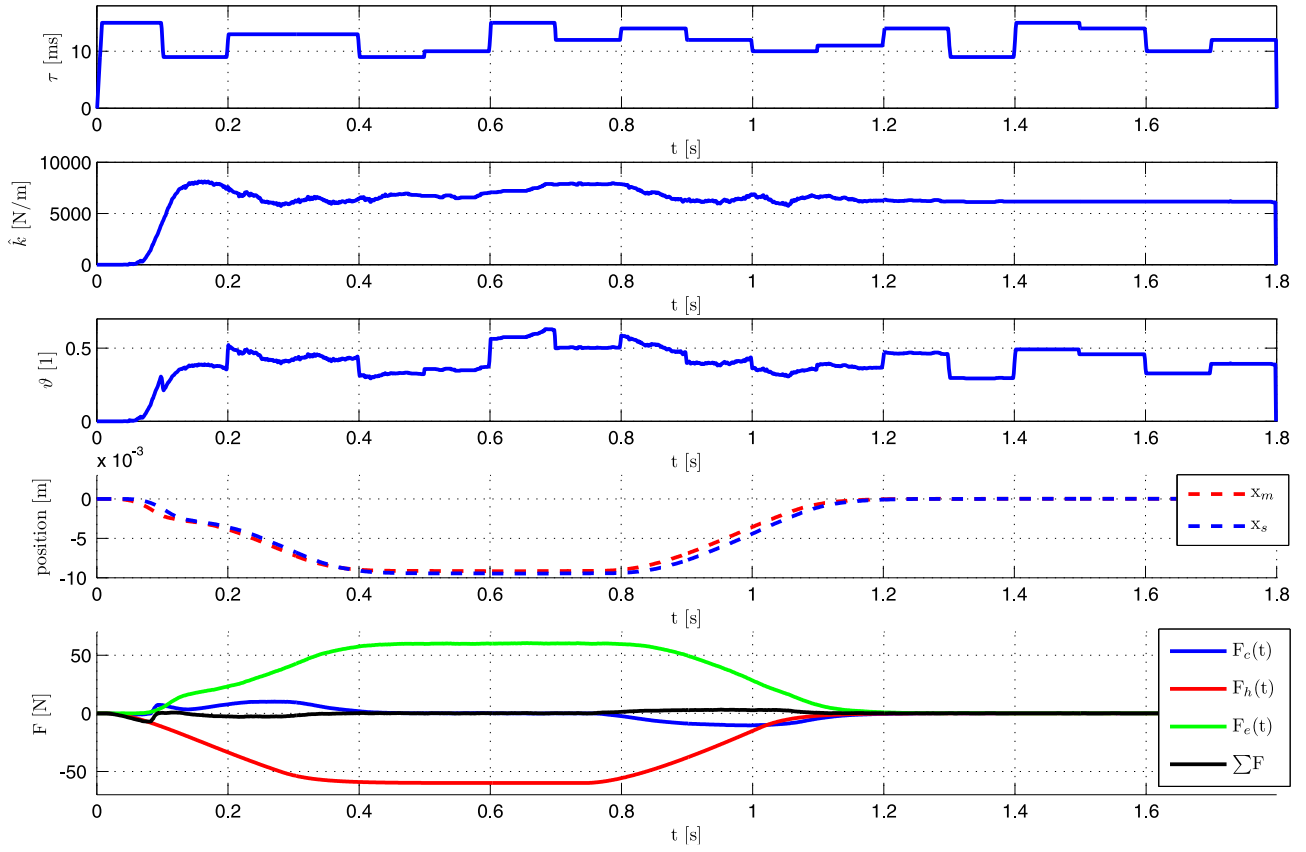


(a) $\tau \sim 18\text{ms}$

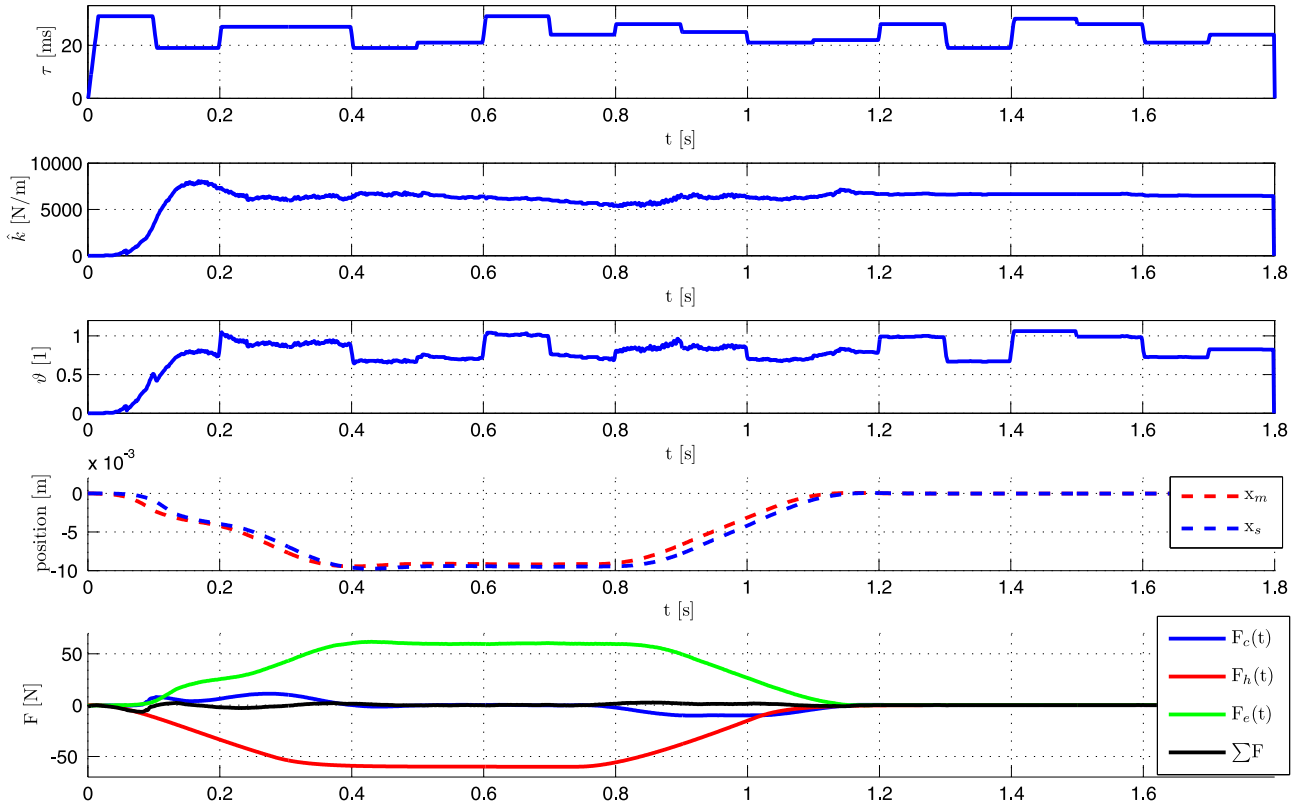


(b) $\tau \sim 35\text{ms}$

7.8. ábra. 6800 [N/m] merevségű rugó megfogása TP szabályozó nélkül

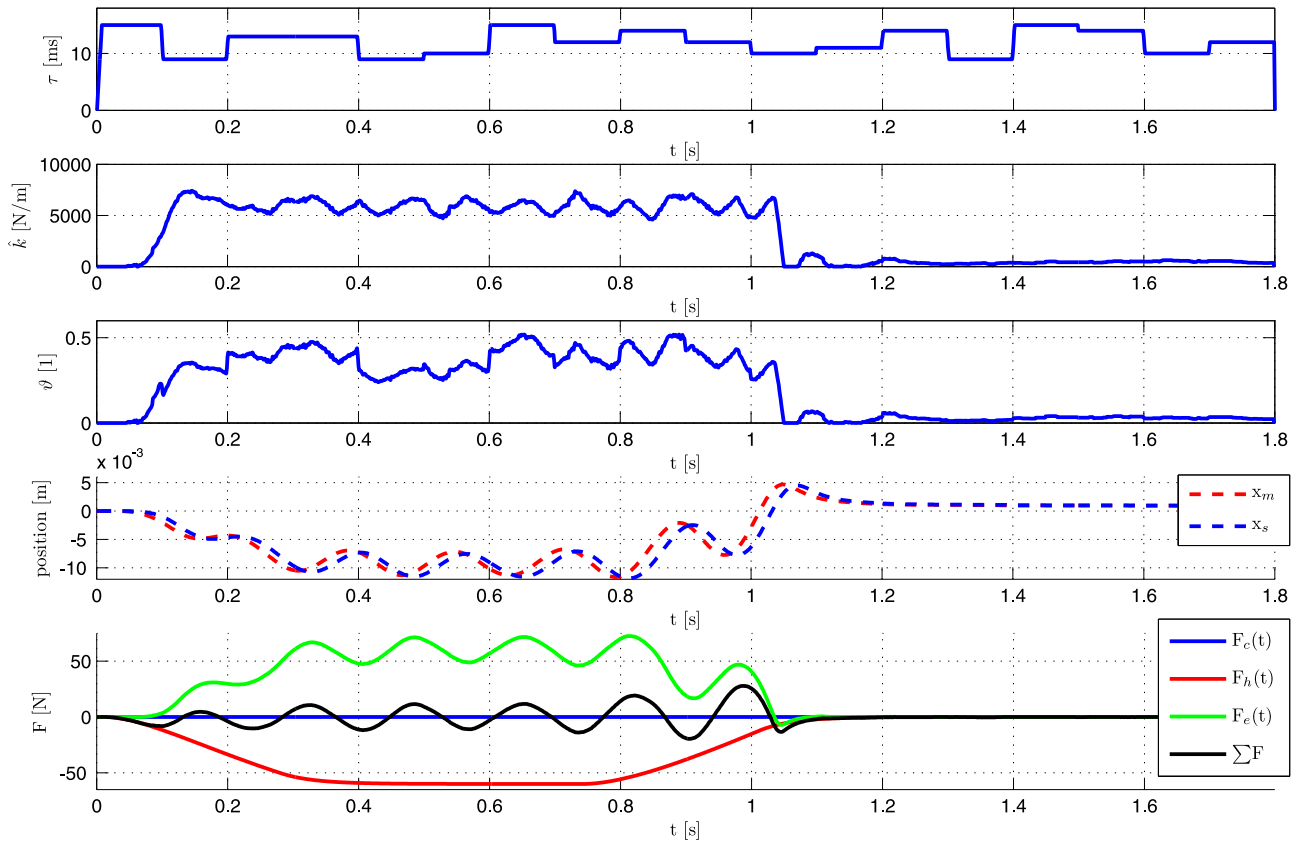


(a) $\tau \sim 12\text{ms}$

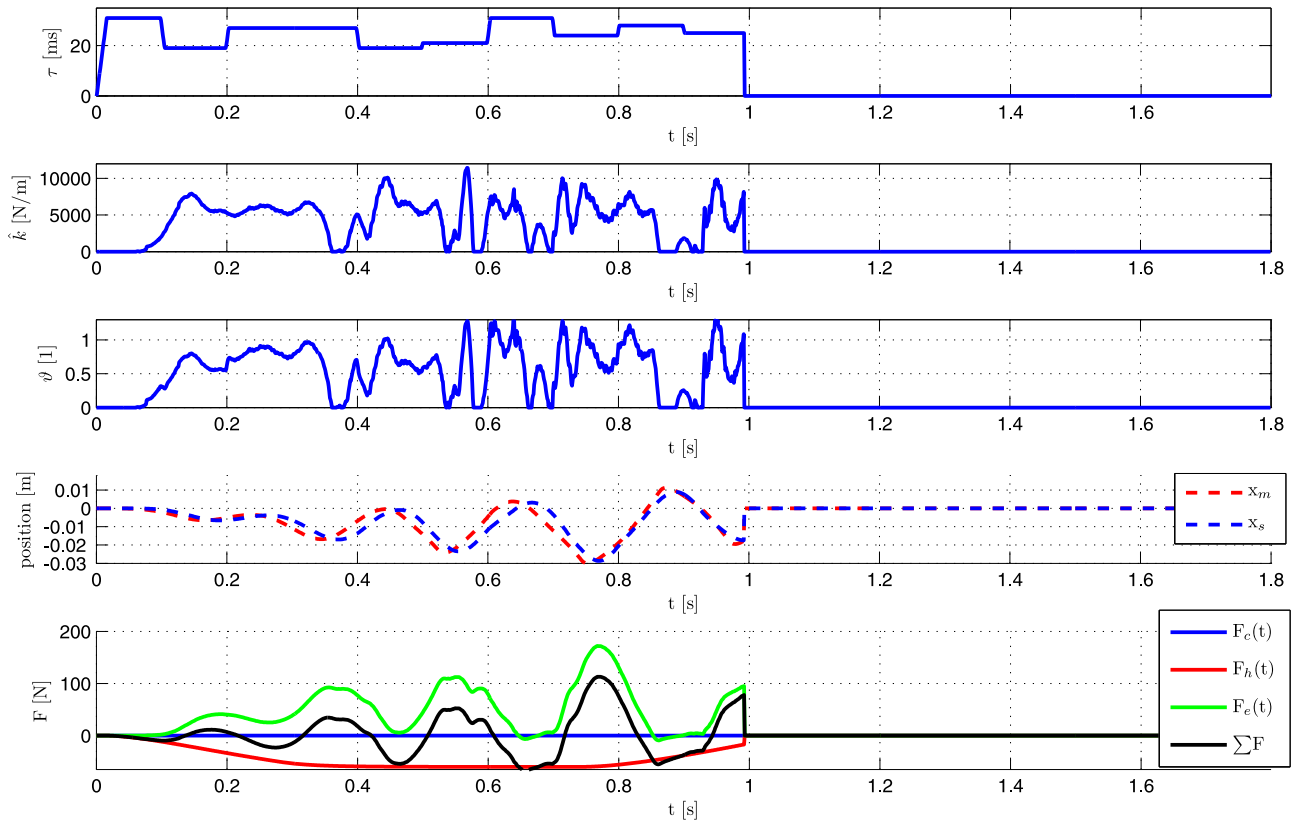


(b) $\tau \sim 24\text{ms}$

7.9. ábra. 6300 [N/m] merevségű rugó megfogása TP szabályzó működése esetén



(a) $\tau \sim 12\text{ms}$



(b) $\tau \sim 24\text{ms}$

7.10. ábra. 6300 [N/m] merevségű rugó megfogása TP szabályozó nélkül

8. fejezet

Összefoglalás

Jelen TDK dolgozatban összefoglalt kutatómunkám során az erővisszacsatolt telemanipulációs megfogás időkésés-érzékenységevel foglalkoztam. A megfogás, mint dinamikai folyamat stabilitása a mester és a szolga oldal közötti időkéséstől és a megfogott tárgy merevségétől függ. Munkám fő célja a konzulensem által javasolt, időkésésre nézve adaptív szabályozási struktúra és tervezési eljárás kiterjesztése volt oly módon, hogy az időkésésen túl a távoli környezet merevségét is figyelembe vegye.

Munkám során több új elméleti eredmény is született:

- A paraméter tér megfelelő átalakításával (dimenziótlan paraméter bevezetésével) a TP^τ modell transzformációt egy eredetileg nem hiper-téglatest alakú transzformációs téren hajtottam végre. Az itt javasolt technika általánosan alkalmazható hasonló esetekben.
- Az TP^τ modell transzformáció első lépésében alkalmazott újradefiniálás specializálásával sikerült az eddigieknél pontosabb és szemléletesebb modellt előállítanom, amely mérhető és visszacsatolható állapotváltozókat alkalmaz, így elkerülhető a korábbi megoldásban szükséges állapotmegfigyelő alkalmazása.
- Az dolgozatban áttekintett merevségbecslő eljárások szimulációs és kísérleti vizsgálatával megmutattam, hogy telemanipulációs rendszerek irányításához kielégítő pontossággal online becsülhető a környezeti merevség.

Az elvégzett numerikus szimulációk és valós robotokon végrehajtott kísérletek szerint a javasolt fejlesztésekkel a telemanipulációs rendszer viselkedése már nem függ az időkéséstől a tervezési-paramétertartományban, és azon túl is csak fokozatosan veszíti el a stabilitását. A szabályozott rendszer jó tulajdonságokat mutat változó paraméterek: nemlineáris merevégű környezet és változó időkésés esetén is.

A szabályozótervezéssel kapcsolatos kutatás következő kihívása, a tervezés során figyelembe vett időkéséstartomány kiterjesztése az instabil tartományra. A merevségbecslő eljárás optimalizálása is fontos feladat. A különböző merevség-megfigyelő eljárások ötvözésével és optimalizálásával egy új merevségbecslő modul kifejlesztését tartom célszerűnek, ami egyesíti a parametrikus és nem-parametrikus módszerek előnyeit és más robot-interakciós irányítási feladatok során is alkalmazható.

Irodalomjegyzék

- [1] C. Beltran-Gonzalez, A. Gasteratos, A. Amanatiadis, D. Chrysostomou, R. Guzman, A. Toth, L. Szollosi, A. Juhasz, and P. Galambos, „Methods and techniques for intelligent navigation and manipulation for bomb disposal and rescue operations,” in *Safety, Security and Rescue Robotics, 2007. SSR 2007. IEEE International Workshop on*, pp. 1–6, 2007.
- [2] P. Galambos, M. Boleraczki, A. Juhasz, A. Tóth, J. Puspoki, and G. Arz, „Development of the prototype of a master-slave servoelectrical gripper with forcefeedback for telemanipulation tasks,” in *Proceedings of the Fifth Conference on Mechanical Engineering*, (Budapest, Hungary), May 2006.
- [3] P. Galambos, „TP model transformation based control design for time-delay systems: Application in telemanipulation - PhD disszertáció vázlat,” *Hungary: Budapest, MTA-SZTAKI, BME*, 2012.
- [4] J. Cui, S. Tosunoglu, R. Roberts, C. Moore, and D. W. Repperger, „A review of teleoperation system control,” in *Proceedings of the 2006 Florida conference recent advances in robotics (FCRAR)*, *Florida Atlantic University, FL*, 2003.
- [5] P. Galambos, „Vibrotactile feedback for haptics and telemanipulation: Survey, concept and experiment,” *Acta Polytechnica Hungarica*, vol. 9, no. 1, pp. 41–65, 2012.
- [6] P. F. Hokayem and M. W. Spong, „Bilateral teleoperation: An historical survey,” *Automatica*, vol. 42, pp. 2035–2057, Dec. 2006.
- [7] N. Hogan, „Impedance control: An approach to manipulation: Part I—Theory,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, pp. 1–7, Mar. 1985.
- [8] N. Hogan, „Impedance control: An approach to manipulation: Part II—Implementation,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, pp. 8–16, Mar. 1985.
- [9] N. Hogan, „Impedance control: An approach to manipulation: Part III—Applications,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, pp. 17–24, Mar. 1985.

- [10] T. Hulin, C. Preusche, and G. Hirzinger, „Stability boundary for haptic rendering: Influence of physical damping,” in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, p. 1570–1575, 2006.
- [11] J. J. Gil, E. Sánchez, T. Hulin, C. Preusche, and G. Hirzinger, „Stability boundary for haptic rendering: Influence of damping and delay,” in *Robotics and Automation, 2007 IEEE International Conference on*, p. 124–129, 2007.
- [12] T. Hulin, C. Preusche, and G. Hirzinger, „Stability boundary for haptic rendering: Influence of human operator,” in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, p. 3483–3488, 2008.
- [13] L. De Lathauwer, B. De Moor, and J. Vandewalle, „A multilinear singular value decomposition,” *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, p. 1253–1278, 2000.
- [14] P. Baranyi, „TP model transformation as a way to LMI based controller design,” *IEEE Transaction on Industrial Electronics*, vol. 51, pp. 387–400, April 2004.
- [15] P. Baranyi, „Convex hull generation methods for polytopic representations of LPV models,” in *Applied Machine Intelligence and Informatics, 2009. SAMI 2009. 7th International Symposium on*, pp. 69–74, IEEE, 2009.
- [16] P. Baranyi, Z. Petres, P. Korondi, Y. Yam, and H. Hashimoto, „COMPLEXITY RELAXATION OF THE TENSOR PRODUCT MODEL TRANSFORMATION FOR HIGHER DIMENSIONAL PROBLEMS,” *Asian Journal of Control*, vol. 9, pp. 195–200, Oct. 2008.
- [17] P. Baranyi, L. Szeidl, P. Várlaki, and Y. Yam, „Definition of the HOSVD-based canonical form of polytopic dynamic models,” in *3rd International Conference on Mechatronics (ICM 2006)*, (Budapest, Hungary), pp. 660–665, July 3-5 2006.
- [18] P. Baranyi, L. Szeidl, P. Várlaki, and Y. Yam, „Numerical reconstruction of the HOSVD-based canonical form of polytopic dynamic models,” in *10th International Conference on Intelligent Engineering Systems*, (London, UK), pp. 196–201, June 26-28 2006.
- [19] P. Gróf, P. Baranyi, and P. Korondi, „Convex hull manipulation based control performance optimisation,” *WSEAS Transactions on Systems and Control*, vol. 5, pp. 691–700, August 2010.
- [20] „TP tool for MATLAB.”
- [21] G. Grioli and A. Bicchi, „A real-time parametric stiffness observer for VSA devices,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, p. 5535–5540, 2011.

- [22] D. Erickson, M. Weber, and I. Sharf, „Contact stiffness and damping estimation for robotic systems,” *The International Journal of Robotics Research*, vol. 22, no. 1, p. 41–57, 2003.
- [23] N. Diolaiti, C. Melchiorri, and S. Stramigioli, „Contact impedance estimation for robotic systems,” *IEEE Transactions on Robotics*, vol. 21, pp. 925–935, Oct. 2005.
- [24] G. Grioli and A. Bicchi, „A non-invasive real-time method for measuring variable stiffness,” *Proc. Robotics Science and Systems (RSS 2010), Zaragoza, E*, 2010.
- [25] A. Serio, G. Grioli, I. Sardellitti, N. G. Tsagarakis, and A. Bicchi, „A decoupled impedance observer for a variable stiffness robot,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, p. 5548–5553, 2011.
- [26] P. Galambos, „Stability boundary of impedance controlled robots: effect of stiffness, damping, friction and delay,” in *Proceedings of the 15th WSEAS international conference on Systems*, p. 247–252, 2011.
- [27] J. Kuti, P. Galambos, and P. Baranyi, „Delay and stiffness dependent polytopic LPV model of impedance controlled robot interaction,” in *Fifth Győr Symposium and First Hungarian–Polish Joint Conference on Computational Intelligence*, pp. 174–175, 2012.
- [28] K. Tanaka and H. O. Wang, *Fuzzy Control Systems Design and Analysis: A Linear Matrix Inequality Approach*. John Wiley & Sons, Inc., 2001.
- [29] A. Sala and C. Ariño, „Relaxed stability and performance LMI conditions for takagi–sugeno fuzzy systems with polynomial constraints on membership function shapes,” *IEEE Transactions on Fuzzy Systems*, vol. 16, pp. 1328–1336, Oct. 2008.
- [30] C. Ariño and A. Sala, „Relaxed LMI conditions for closed-loop fuzzy systems with tensor-product structure,” *Engineering Applications of Artificial Intelligence*, vol. 20, no. 8, pp. 1036 – 1046, 2007.
- [31] A. Juhász, „Hirata DC szervó motor hajtásának tervezése és megvalósítása,” *Budapesti Műszaki és Gazdaságtudományi Egyetem, Gépgyártástechnológia Tanszék*, 2005.
- [32] D. Fliszár, „Impedancia szabályozás stabilitásának vizsgálata a telemanipulációban,” *Budapesti Műszaki és Gazdaságtudományi Egyetem, Gyártástudomány és -technológia Tanszék*, 2011.

Függelék

F.1. A TP^r modell transzformáció MATLAB programmal

Inicializáció. Először megadom az impedanciamodell paramétereit, a transzformációs tér és a diszkrétizációs rács méretét.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                               Initialization                               %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc;
clear all;
echo off;

%parameters of impedance model
model.m = 1; %[kg] mass
model.b = 120; %[Ns/m] damping

%transformation space
model.kMin = 1000; %[N/m] minimal stiffness
model.kMax = 8000; %[N/m] maximal stiffness
%theta=Tau/TauCrit, where TauCrit=1.566*b/k
model.thetaMax=0.65; %[] Maximum delay/critical delay:

%grid sizes
model.kN=137;
model.thetaN=131;

%gridpoints to k
model.kGP=linspace(model.kMin,model.kMax,model.kN);
%gridpoints to theta
model.thetaGP=linspace(0,model.thetaMax,model.thetaN);

% Mass-Damper system
model.A = [-model.b/model.m 0; 1 0];
model.B = [1/model.m; 0];
model.C = [0 1];
model.C_ = [1 0]; % C in the identified system
model.D = [0];
MD = ss(model.A,model.B,model.C,model.D);
```

Diszkretizáció. Előállítunk egy periódikus négyszögjelet. Ezzel a bemenőjellel minden rácsponti paraméterezéshez lefuttatjuk az időképes rendszer szimulációját. Majd sebességválaszra ARX típusú identifikációval kéttárolós rendszert illesztünk.

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                               Input Signal for Identification                               %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%sampling time to identification
model.dt=0.001;                                     %secundum
T=[0:model.dt:20];
U=zeros(1,length(T));
Period = 4;

for i = 1 : length(T)
    if(mod(T(i),Period)<Period/2)
        U(i)=100;
    else
        U(i)=-100;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                               Identification of delayed systems                               %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:model.kN;
    i/model.kN*50
    for j=1:model.thetaN;
        k = model.kGP(i);
        tau = model.thetaGP(j)*1.566*model.b/k;
        RS = ss(k,'Inputdelay',tau); % Remote Stiffnes
        Cloop = feedback(MD,RS);
        [A0,B1,B2,C1,C2,D11,D12,D21,D22,E,tau] = getDelayModel(Cloop,'mat');
        if (j==1)
            RS = ss(k); % Remote Stiffnes
            Cloop = feedback(MD,RS);
            Cloop.c=[1 0];
        else
            Cloop=setDelayModel(A0,B1,B2,[1 0],[0 1],D11,D12,D21,D22,tau);
        end
    end

    %Identification 1
    Y = lsim(Cloop,U,T);
    yu = iddata(Y,U',model.dt);
    IDCloop{i,j} = arx(yu,[2 2 1],'Cov','None','Focus','Sim');
end
end

```

Az átviteli függvényt folytonos időbe transzformáljuk, majd ebből kisebb módosítások után felépíthetők a rendszermátrixok és a diszkrétizált tenzor.

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Creating State Space structure from the discrete transfer fn.%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
model.SSPParameters=zeros(model.kN,model.thetaN,3);
for i = 1:model.kN;
    for j = 1:model.thetaN;
        a=d2c(IDCloop{i,j});
        m=1/a.B(end-1);
        b=a.A(2)*m;
        k=a.A(3)*m;
        model.SSPParameters(i,j,:)=[m b k];
    end
end

for i = 1:model.kN;
    for j = 1:model.thetaN;
        m=model.SSPParameters(i,j,1);
        b=model.SSPParameters(i,j,2);
        k=model.SSPParameters(i,j,3);

        if model.typ==2
            model.SYSD(:,:,i,j)=[-b/m 1/m 1/m; -k 0 0];
        end;

        if model.typ==7
            model.SYSD(:,:,i,j) =[-b/m/sqrt(k) 1/m 1/m/sqrt(k); -1 0 0];
        end;
    end
end
end

```

Kanonikus alak képzése. A `model.SYSD` diszkrétizált tenzor megfelelő dimenzióiban végrehajtott szinguláris értékek szerinti felbontással megkapjuk a kanonikus alak `model.SYSSCAN` magtenzorát és `model.SYSUCAN{3}`, `model.SYSUCAN{4}` súlyfüggvényeket tartalmazó mátrixokat.

```

%% HOSVD
[model.SYSSCAN model.SYSUCAN model.SYSsv model.SYSTol]...
= hosvd(model.SYSD, [0 0 1 1]);

```

Konvex burok előállítása. A CNO típusú konvex burok a TPTool MATLAB toolbox `genhull()` függvényével történik, a kapott magtenzor `model.SYSSCNO` a súlyfüggvénymátrixokat `model.SYSUCNO` tartalmazza.

```
model.hullN0=[i_k i_theta];

SYSUCANR{3}=model.SYSUCAN{3}(1:end,1:model.hullN0(1));
SYSUCANR{4}=model.SYSUCAN{4}(1:end,1:model.hullN0(2));

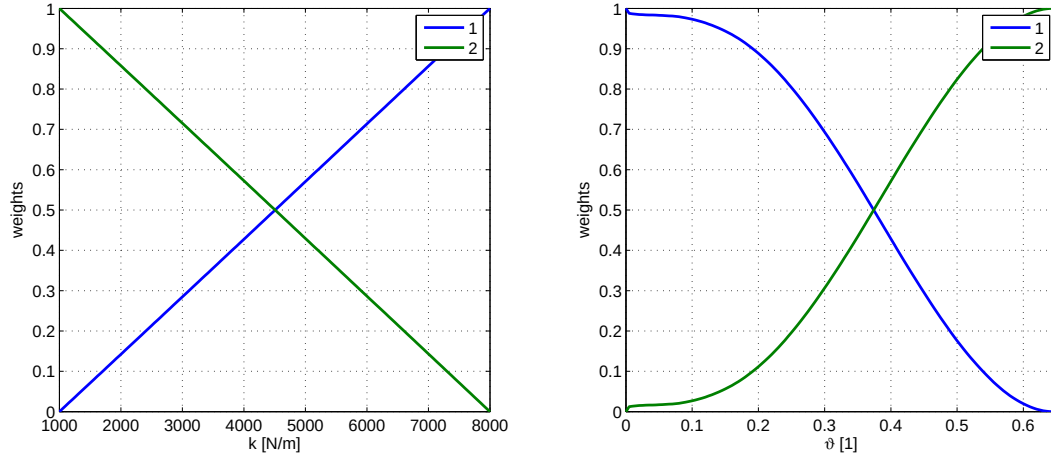
%Getting the CNO weighting function of SYS
model.SYSUCNO = genhull(SYSUCANR, 'cno');

%Pseudoinverse of the CNO weighting funcs
SYSPUCNO_3 = pinv(model.SYSUCNO{3});
SYSPUCNO_4 = pinv(model.SYSUCNO{4});

%CNO core matrix for SYS
model.SYSSCNO = tprod1(model.SYSD,SYSPUCNO_3,3);
model.SYSSCNO = tprod1(model.SYSSCNO,SYSPUCNO_4,4);
```

F.2. Konvex politopikus reprezentációk

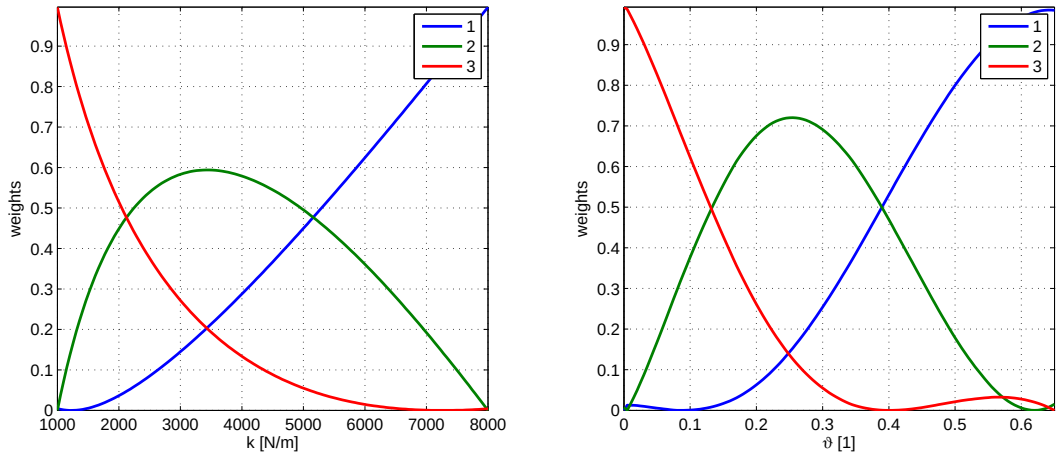
F.2.1. Az "A" modell CNO konvex burka



F.1. ábra. Az "A" modell CNO burokának súlyfüggvényei

$$\begin{aligned}
 [\mathbf{AB}]_{i_k=1, i_\vartheta=1} &= 10^3 \begin{bmatrix} -0.0973 & 0.0010 & 0.0010 \\ -8.0086 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=1} &= \begin{bmatrix} -85.5686 & 0.8738 & 0.8738 \\ -998.5784 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=2} &= 10^3 \begin{bmatrix} -0.0081 & 0.0006 & 0.0006 \\ -7.8730 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=2} &= \begin{bmatrix} 2.2066 & 0.1716 & 0.1716 \\ -953.8198 & 0 & 0 \end{bmatrix}
 \end{aligned}$$

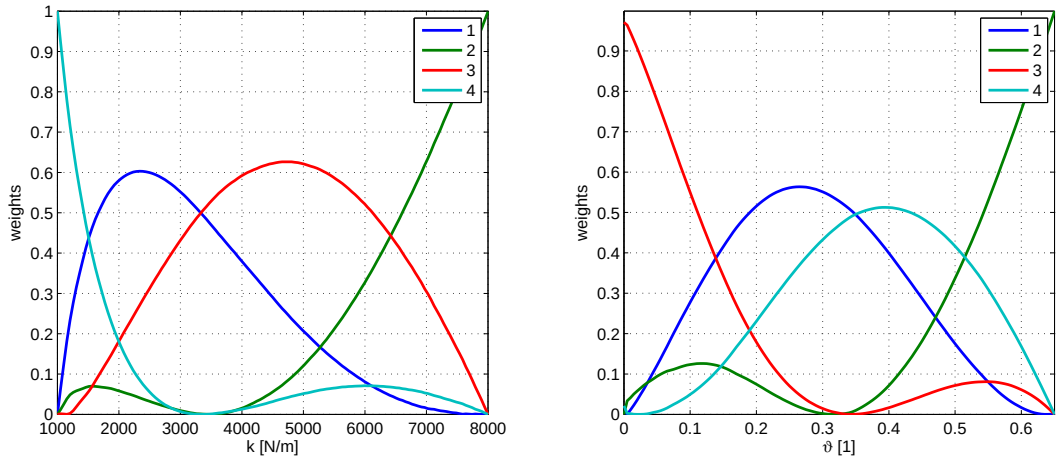
F.2.2. A "B" modell CNO konvex burka



F.2. ábra. A "B" modell CNO burkának súlyfüggvényei

$$\begin{aligned}
 [\mathbf{AB}]_{i_k=1, i_\vartheta=1} &= 10^3 \begin{bmatrix} -0.0115 & 0.0006 & 0.0006 \\ -7.8993 & 0 & 0 \\ -2.6134 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=1} &= 10^3 \begin{bmatrix} -0.0104 & 0.0004 & 0.0004 \\ -2.6134 & 0 & 0 \\ -3.2763 & 0.1259 & 0.1259 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=1} &= \begin{bmatrix} -3.2763 & 0.1259 & 0.1259 \\ -934.1801 & 0 & 0 \\ -39.0760 & 0.5547 & 0.5547 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=2} &= 10^3 \begin{bmatrix} -0.0674 & 0.0009 & 0.0009 \\ -8.0334 & 0 & 0 \\ -0.0618 & 0.0009 & 0.0009 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=2} &= 10^3 \begin{bmatrix} -0.0618 & 0.0009 & 0.0009 \\ -2.6773 & 0 & 0 \\ -39.0760 & 0.5547 & 0.5547 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=2} &= \begin{bmatrix} -39.0760 & 0.5547 & 0.5547 \\ -945.4508 & 0 & 0 \\ -0.1189 & 0.0010 & 0.0010 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=3} &= 10^3 \begin{bmatrix} -0.1189 & 0.0010 & 0.0010 \\ -8.0216 & 0 & 0 \\ -0.1203 & 0.0010 & 0.0010 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=3} &= 10^3 \begin{bmatrix} -0.1203 & 0.0010 & 0.0010 \\ -2.7067 & 0 & 0 \\ -120.4222 & 1.0478 & 1.0478 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=3} &= \begin{bmatrix} -120.4222 & 1.0478 & 1.0478 \\ -972.2104 & 0 & 0 \end{bmatrix}
 \end{aligned}$$

F.2.3. A "C" modell CNO konvex burka

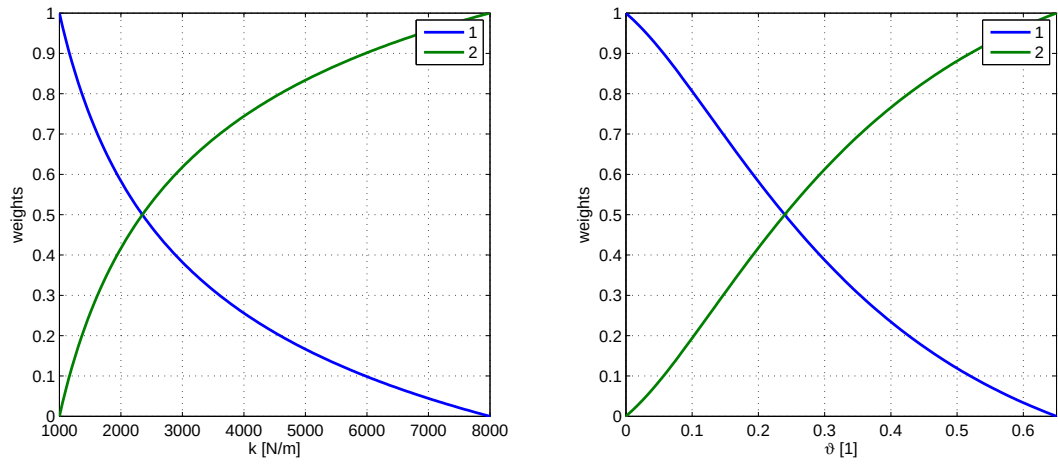


F.3. ábra. A "C" modell CNO burokának súlyfüggvényei

$$\begin{aligned}
 [\mathbf{AB}]_{i_k=1, i_\vartheta=1} &= \begin{bmatrix} -91.8121 & 1.0904 & 1.0904 \\ -379.1835 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=1} &= 10^3 \begin{bmatrix} -0.1124 & 0.0011 & 0.0011 \\ -8.0758 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=1} &= 10^3 \begin{bmatrix} -0.1150 & 0.0012 & 0.0012 \\ -6.3674 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=4, i_\vartheta=1} &= 10^3 \begin{bmatrix} -0.0812 & 0.0010 & 0.0010 \\ -1.0086 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=2} &= \begin{bmatrix} -6.0958 & 0.1759 & 0.1759 \\ -347.2675 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=2} &= 10^3 \begin{bmatrix} -0.0096 & 0.0006 & 0.0006 \\ -7.8745 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=2} &= 10^3 \begin{bmatrix} -0.0114 & 0.0005 & 0.0005 \\ -6.1664 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=4, i_\vartheta=2} &= \begin{bmatrix} -3.4352 & 0.1570 & 0.1570 \\ -968.9871 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=3} &= \begin{bmatrix} -123.7084 & 1.0609 & 1.0609 \\ -367.5746 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=3} &= 10^3 \begin{bmatrix} -0.1230 & 0.0010 & 0.0010 \\ -8.0136 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=3} &= 10^3 \begin{bmatrix} -0.1230 & 0.0010 & 0.0010 \\ -6.2926 & 0 & 0 \end{bmatrix}
 \end{aligned}$$

$$\begin{aligned}
[\mathbf{AB}]_{i_k=4, i_\vartheta=3} &= 10^3 \begin{bmatrix} -0.1234 & 0.0011 & 0.0011 \\ -1.0027 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=1, i_\vartheta=4} &= \begin{bmatrix} 28.3207 & -0.1442 & -0.1442 \\ -277.6349 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=2, i_\vartheta=4} &= 10^3 \begin{bmatrix} 0.0173 & 0.0005 & 0.0005 \\ -7.8605 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=3, i_\vartheta=4} &= 10^3 \begin{bmatrix} 0.0169 & 0.0005 & 0.0005 \\ -6.1141 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=4, i_\vartheta=4} &= \begin{bmatrix} 28.4018 & -0.1681 & -0.1681 \\ -937.3831 & 0 & 0 \end{bmatrix}
\end{aligned}$$

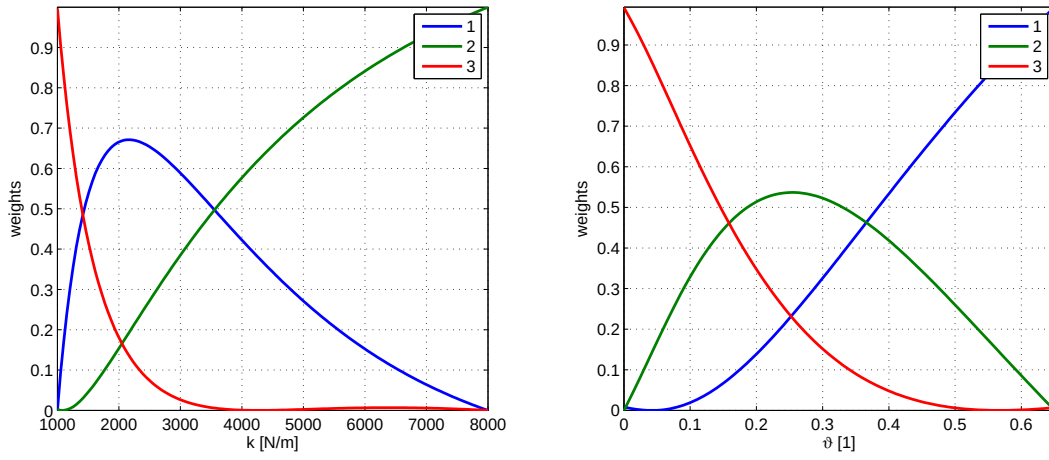
F.2.4. A "D" modell CNO konvex burka



F.4. ábra. A "D" modell CNO burokának súlyfüggvényei

$$\begin{aligned}
[\mathbf{AB}]_{i_k=1, i_\vartheta=1} &= \begin{bmatrix} -3.4564 & 1.0575 & 0.0326 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=2, i_\vartheta=1} &= \begin{bmatrix} -1.3424 & 1.0880 & 0.0119 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=1, i_\vartheta=2} &= \begin{bmatrix} 0.0173 & 0.0633 & 0.0051 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=2, i_\vartheta=2} &= \begin{bmatrix} -0.1660 & 0.5989 & 0.0075 \\ -1.0000 & 0 & 0 \end{bmatrix}
\end{aligned}$$

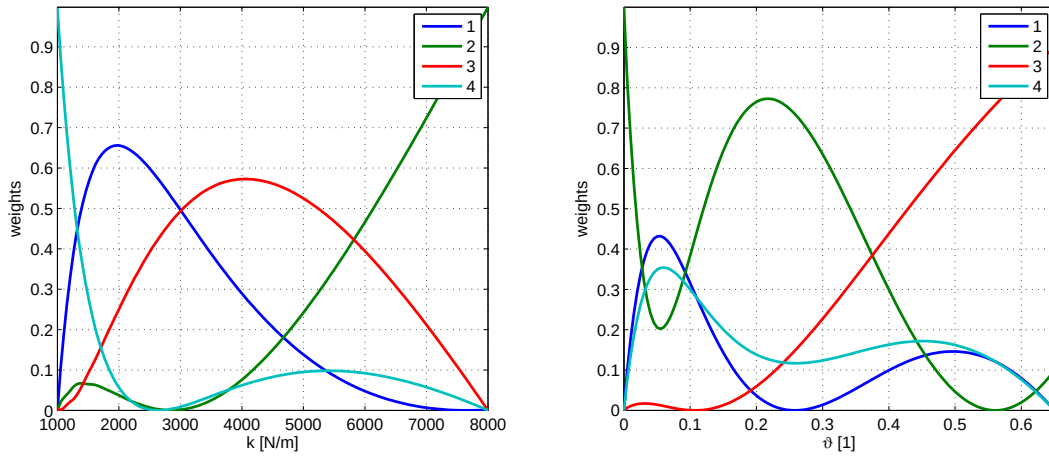
F.2.5. Az "E" modell CNO konvex burka



F.5. ábra. Az "E" modell CNO burokának súlyfüggvényei

$$\begin{aligned}
 [\mathbf{AB}]_{i_k=1, i_\vartheta=1} &= \begin{bmatrix} -0.1645 & 0.1715 & 0.0055 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=1} &= \begin{bmatrix} -0.1069 & 0.5407 & 0.0064 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=1} &= \begin{bmatrix} -0.1463 & 0.1416 & 0.0039 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=2} &= \begin{bmatrix} -1.0825 & 0.7335 & 0.0177 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=2} &= \begin{bmatrix} -0.7634 & 0.9860 & 0.0111 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=2} &= \begin{bmatrix} -0.9015 & 0.5062 & 0.0165 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=3} &= \begin{bmatrix} -2.7095 & 1.0329 & 0.0229 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=3} &= \begin{bmatrix} -1.3176 & 0.9949 & 0.0110 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=3} &= \begin{bmatrix} -3.8687 & 1.0503 & 0.0333 \\ -1.0000 & 0 & 0 \end{bmatrix}
 \end{aligned}$$

F.2.6. Az "F" modell CNO konvex burka



F.6. ábra. Az "F" modell CNO burokának súlyfüggvényei

$$\begin{aligned}
 [\mathbf{AB}]_{i_k=1, i_\vartheta=1} &= \begin{bmatrix} -12.5262 & 3.0196 & 0.0746 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=1} &= \begin{bmatrix} -3.9440 & 1.2955 & 0.0143 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=1} &= \begin{bmatrix} -2.2881 & 1.0441 & -0.0023 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=4, i_\vartheta=1} &= \begin{bmatrix} -15.5644 & 3.2867 & 0.1027 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=2} &= \begin{bmatrix} -3.2897 & 1.0348 & 0.0281 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=2} &= \begin{bmatrix} -1.3267 & 0.9896 & 0.0110 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=2} &= \begin{bmatrix} -1.0600 & 0.9885 & 0.0085 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=4, i_\vartheta=2} &= \begin{bmatrix} -3.7965 & 1.0299 & 0.0326 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=1, i_\vartheta=3} &= \begin{bmatrix} 0.1687 & 0.0119 & 0.0021 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=2, i_\vartheta=3} &= \begin{bmatrix} 0.0224 & 0.5068 & 0.0057 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
 [\mathbf{AB}]_{i_k=3, i_\vartheta=3} &= \begin{bmatrix} -0.0358 & 0.4716 & 0.0069 \\ -1.0000 & 0 & 0 \end{bmatrix}
 \end{aligned}$$

$$\begin{aligned}
[\mathbf{AB}]_{i_k=4, i_\vartheta=3} &= \begin{bmatrix} 0.2357 & 0.0297 & 0.0010 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=1, i_\vartheta=4} &= \begin{bmatrix} 8.9607 & -1.5530 & -0.0323 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=2, i_\vartheta=4} &= \begin{bmatrix} 2.1468 & 0.6427 & 0.0073 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=3, i_\vartheta=4} &= \begin{bmatrix} 0.5855 & 0.9713 & 0.0234 \\ -1.0000 & 0 & 0 \end{bmatrix} \\
[\mathbf{AB}]_{i_k=4, i_\vartheta=4} &= \begin{bmatrix} 11.8118 & -1.9123 & -0.0587 \\ -1.0000 & 0 & 0 \end{bmatrix}
\end{aligned}$$

F.3. LMI alapú szabályozótervezés

A tervezés a következő rövid eljárással történik.

```
S=model.SYSSCNO;

% Add C and D to the core tensor
for i=1:size(S,3)
    for j=1:size(S,4)
        S(3,1:2,i,j)=[1 0];
        S(3,3,i,j)=0;
    end
end

S=permute(S,[3,4,1,2]);

if model.typ==7
    LMI_CNO = lmi_ControllerConstrain(S,2,350,350,0,500);
end;

model.KLMI_CNO=LMI_CNO.F; % Controller gain
```

A programban a Tanaka és Wang által [28] kidolgozott LMI rendszert használjuk fel az `lmi_ControllerConstrain()` függvénnyel, amit szintén mellékelek.

```
function obscont = lmi_ControllerConstrain(S,Asize,mu1,mu2,phi,gamma)
%LMI_CONTROLLER Design controller with asymptotic stability constraints
%from Galambos
%
% S - core tensor of a TP model
% Asize - size of matrix A in the system matrix (S = [A B; C D])
% mu1,mu2 - Control signal constrain
% phi - norm of the initial state
% obscont.F - feedback gain tensor

I = size(S);
Bsize = I(end) - Asize;
Csize = I(end-1) - Asize;

Iomega = I(1:end-2);
R = prod(Iomega);
S = reshape(S,[R I(end-1) I(end)]);
```

```

A = S(:, 1:Asize, 1:Asize);
B = S(:, 1:Asize, Asize+1:Asize+Bsize);
C = S(:, Asize+1:Asize+Csize, 1:Asize);

X = sdpvar(Asize, Asize, 'symmetric');
M = cell(1, R);
for r = 1:R
    M{r} = sdpvar(Bsize, Asize, 'full');
end

% X > 0
f = set(X > 0, 'poz def');

%% Asymptotic stability (controller)
%  $X \cdot A_i' - M_i' \cdot B_i' + A_i \cdot X - B_i \cdot M_i < 0$  %Stable Feedback gain
for i = 1:R
    Ai = reshape(A(i, :, :), [Asize Asize]);
    Bi = reshape(B(i, :, :), [Asize Bsize]);
    f = f + set(X*Ai' - M{i}'*Bi' + Ai*X - Bi*M{i} < 0, '');
end

%  $X \cdot A_i' - M_j' \cdot B_i' + A_i \cdot X - B_i \cdot M_j + X \cdot A_j' - M_i' \cdot B_j' + A_j \cdot X - B_j \cdot M_i < 0$  %Stable Feedback gain
for i = 1:R
    for j = i+1:R
        Ai = reshape(A(i, :, :), [Asize Asize]);
        Aj = reshape(A(j, :, :), [Asize Asize]);
        Bi = reshape(B(i, :, :), [Asize Bsize]);
        Bj = reshape(B(j, :, :), [Asize Bsize]);
        f = f + set(X*Ai' - M{j}'*Bi' + Ai*X - Bi*M{j} + X*Aj' ...
            -M{i}'*Bj' + Aj*X - Bj*M{i} < 0, '');
    end
end

%% Disturbance rejection
% X > 0
%  $[-1/2 \cdot \{X \cdot A_i' - M_j' \cdot B_i' + A_i \cdot X - B_i \cdot M_j + X \cdot A_j' - M_i' \cdot B_j' + A_j \cdot X - B_j \cdot M_i\},$   $-1/2 \cdot (E_i + E_j), 1/2 \cdot X \cdot (C_i + C_j)';$ 
%  $-1/2 \cdot (E_i + E_j)',$   $\gamma^2 \cdot I,$   $0;$ 
%  $1/2 \cdot (C_i + C_j) \cdot X],$   $0,$   $I] \geq 0$ 
for i = 1:R
    for j = i+1:R
        Ai = reshape(A(i, :, :), [Asize Asize]);
        Aj = reshape(A(j, :, :), [Asize Asize]);
        Bi = reshape(B(i, :, :), [Asize Bsize]);
        Bj = reshape(B(j, :, :), [Asize Bsize]);
        Ci = reshape(C(i, :, :), [Csize Asize]);
        Cj = reshape(C(j, :, :), [Csize Asize]);

```

```

    Ei = reshape(B(i,:,:), [Asize Bsize]);
    Ej = reshape(B(j,:,:), [Asize Bsize]);

    f = f + set([ -(1/2)*(X*Ai' -M{j}'*Bi' +Ai*X -Bi*M{j})...
        +X*Aj' -M{i}'*Bj' +Aj*X -Bj*M{i}),...
        -(1/2)*(Ei+Ej), (1/2)*X*(Ci+Cj)';...
        -(1/2)*(Ei+Ej)', gamma^2*eye(Bsize),0;...
        (1/2)*(Ci+Cj)*X,0,eye(Bsize)]>0, '3');

    end
end

%% Constraints on the control value
% phi^2 I < X
f = f + set(phi^2*eye(Asize) < X, 'phi^2 I < X');

% [X, Mr'; Mr, mu^2 I] > 0
for r = 1:R
    mu=mu2;
    f = f + set([X M{r}'; M{r} mu^2*eye(Bsize)] > 0, '4');
end

%% solve
solvesdp(f);
checkset(f)

%% OUTPUT

% Obtain controller gain tensor
F = zeros(R, Bsize, Asize);
for i = 1:R
    F(i,:,:)= double(M{i})/double(X);
end

F = reshape(F, [Iomega Bsize Asize]);
obscont.F=F;

```


TP szabályozó. Első lépésként meghatározzuk a paraméterekhez tartozó súlyfüggvényvektorok értékét.

```
function u = TPcontrollerLMI3sv_(k,tau,x)
echo off;
global model;
        %size(model.KLMI_CNO)=[kN thetaN Bsize Asize]
KCORE=permute(model.KLMI_CNO,[4 3 1 2]);
        %size(KCORE)=[Asize Bsize kN thetaN]

theta=tau/(1.566*model.b/k);

%weighting vector
SYSU=cell(2,1);
SYSU{1}=model.SYSUCNO{3};
SYSU{2}=model.SYSUCNO{4};

W = queryw1(SYSU,[model.kGP(1) model.kGP(end);...
        model.thetaGP(1) model.thetaGP(end)], [k theta]);
```

A kapott vektorokkal súlyozzuk a visszacsatolásokat tartalmazó magtenzort.

```
%controller gain vector
K = tprod1(KCORE,W{1},3);
K = tprod1(K,W{2},4);

k2=k;
if k2<700 k2=700; end
```

Utolsó lépés az állapotvisszacsatolás megvalósítása. (A modellben használt állapotvektoroktól függően ez különféle lehet.)

```
if model.typ==1 || model.typ==2
    u=-K(1:2)'*x;
end;

if model.typ==7
    u=-K(1:2)'*[1 0; 0 1/sqrt(k2)]*x;
end;
```

Merevségmegfigyelők. A merevségmegfigyelőket „Level-2 M file S function” blokkal valósítottam meg. Ezek előnye, hogy testreszabhatóak a kimenő és bemenő portok, valamint a blokk memóriával rendelkezik. Hátrányuk, hogy ezek beállítása és inicializálása terjedelmes kóddal történik. A kód az inicializálásokkal kezdődik majd a kezdeti értékek beállításával folytatódik és végül maga a program következik.

```
function full_est_S(block)
% Level-2 M file S-Function
% Variable stiffness estimator &
% Hunt-Crossley model based stiffness estimator.
% inputs: t,x,dx/dt,F,dF/dt
% outputs:k
    setup(block);
function setup(block)
    %% Register dialog parameter: nums of parameter
    block.NumDialogPrms = 1;
    block.DialogPrmsTunable = {'Tunable'};
    %% Regieste number of input and output ports
    block.NumInputPorts = 5;
    block.NumOutputPorts = 1;
    %% Setup functional port properties to dynamically inherited.
    block.SetPreCompInpPortInfoToDynamic;
    block.SetPreCompOutPortInfoToDynamic;

    %InputPort(1): t: Real
    block.InputPort(1).Complexity = 'Real';
    block.InputPort(1).DataTypeId = 0;
    block.InputPort(1).SamplingMode = 'Sample';
    block.InputPort(1).Dimensions = 1;
    %InputPort(2): x(t): Real
    block.InputPort(2).Complexity = 'Real';
    block.InputPort(2).DataTypeId = 0;
    block.InputPort(2).SamplingMode = 'Sample';
    block.InputPort(2).Dimensions = 1;
    %InputPort(3): dx(t)/dt: Real
    block.InputPort(3).Complexity = 'Real';
    block.InputPort(3).DataTypeId = 0;
    block.InputPort(3).SamplingMode = 'Sample';
    block.InputPort(3).Dimensions = 1;
    %InputPort(4): F(t): Real
    block.InputPort(4).Complexity = 'Real';
    block.InputPort(4).DataTypeId = 0;
    block.InputPort(4).SamplingMode = 'Sample';
    block.InputPort(4).Dimensions = 1;
```

```

    %InputPort(5): dF(t)/dt: Real
    block.InputPort(5).Complexity    = 'Real';
    block.InputPort(5).DataTypeId    = 0;
    block.InputPort(5).SamplingMode  = 'Sample';
    block.InputPort(5).Dimensions    = 1;
    %OutputPort(1): stiffness(t): Real
    block.OutputPort(1).Complexity    = 'Real';
    block.OutputPort(1).DataTypeId    = 0;
    block.OutputPort(1).SamplingMode  = 'Sample';
    block.OutputPort(1).Dimensions    = 1;

    %% Register methods
    block.RegBlockMethod('CheckParameters', @CheckPrms);
    block.RegBlockMethod('ProcessParameters', @ProcessPrms);
    block.RegBlockMethod('PostPropagationSetup', @DoPostPropSetup);
    block.RegBlockMethod('Start', @Start);
    block.RegBlockMethod('WriteRTW', @WriteRTW);
    block.RegBlockMethod('Outputs', @Outputs);
    %% Block runs on TLC in accelerator mode.
    block.SetAccelRunOnTLC(true);
%endfunction

function CheckPrms(block)
;
%endfunction

function DoPostPropSetup(block)
    %% Setup Dwork
    block.NumDworks = 7;
    block.Dwork(1).Name = 'y'; %% y(h-1)
    block.Dwork(1).Dimensions    = 1;
    block.Dwork(1).DatatypeID    = 0;
    block.Dwork(1).Complexity    = 'Real';
    block.Dwork(1).UsedAsDiscState = true;

    block.Dwork(2).Name = 'l'; %% lambda(h-1)
    block.Dwork(2).Dimensions    = 1;
    block.Dwork(2).DatatypeID    = 0;
    block.Dwork(2).Complexity    = 'Real';
    block.Dwork(2).UsedAsDiscState = true;

    block.Dwork(3).Name = 'k'; %% k(h-1)
    block.Dwork(3).Dimensions    = 1;
    block.Dwork(3).DatatypeID    = 0;
    block.Dwork(3).Complexity    = 'Real';
    block.Dwork(3).UsedAsDiscState = true;

```

```

block.Dwork(4).Name = 'n'; %% n(h-1)
block.Dwork(4).Dimensions = 1;
block.Dwork(4).DatatypeID = 0;
block.Dwork(4).Complexity = 'Real';
block.Dwork(4).UsedAsDiscState = true;

block.Dwork(5).Name = 'R_n';      %% R(t-1) to n ( P(t-1) matrix )
block.Dwork(5).Dimensions = 1;
block.Dwork(5).DatatypeID = 0;
block.Dwork(5).Complexity = 'Real';
block.Dwork(5).UsedAsDiscState = true;

block.Dwork(6).Name = 'R_kl';      %% R(t-1)to k&lambda ( P(t-1) matrix )
block.Dwork(6).Dimensions = 2*2;
block.Dwork(6).DatatypeID = 0;
block.Dwork(6).Complexity = 'Real';
block.Dwork(6).UsedAsDiscState = true;

block.Dwork(7).Name = 't';      %% Forgetting factor to k&l estimator
block.Dwork(7).Dimensions = 1;
block.Dwork(7).DatatypeID = 0;
block.Dwork(7).Complexity = 'Real';
block.Dwork(7).UsedAsDiscState = true;

%% Register all tunable parameters as runtime parameters.
block.AutoRegRuntimePrms;

%endfunction

function ProcessPrms(block)
    block.AutoUpdateRuntimePrms;
%endfunction

function Start(block)
    %% Initialize Dwork
    block.Dwork(1).Data = 0;
    block.Dwork(2).Data = 0;
    block.Dwork(3).Data = 10;
    block.Dwork(4).Data = 1;
    ro=10000;
    block.Dwork(5).Data = ro;      %%R(t=0) (P(0))
    block.Dwork(6).Data = reshape(ro*eye(2), [1 2*2]);      %%R(t=0) (P(0))
    block.Dwork(7).Data = 0;
%endfunction

```

```

function Outputs(block)
    if 2==block.DialogPrm(1).Data;
        % Hunt crossley model based estimator

        scala_m = 200; %ha scala_m=1000, mm-ben számol
        scala_s = 50000; %időskálázás

        k=block.Dwork(3).Data;
        l=block.Dwork(2).Data;
        n=block.Dwork(4).Data;
        F=block.InputPort(4).Data;
        y0=block.InputPort(2).Data;
        y=y0*scala_m;
        dy=block.InputPort(3).Data*scala_m/scala_s;

        %% Estimator to k & l parameters
        if y0>0.0001
            beta=0.99;
            R=reshape(block.Dwork(6).Data,[2 2]);
            teta=[k; l];
            mu=y^n*[1; dy];
            fi=F;

            % RLS method
            Q = R * mu / (beta + mu'*R*mu);
            R1= 1/beta * (eye(2)-Q*mu') *R;
            tetal = teta + Q*(fi - mu'*teta); %%new parameter vector
            epszilon=fi-mu'*tetal; %%epszilon(t) - error

            % instrictly variable update
            k2=tetal(1);
            block.Dwork(3).Data=k2;
            l2=tetal(2);
            block.Dwork(2).Data=l2;
            block.Dwork(6).Data = reshape(R1,[1 2*2]);
        end;

        %% Estimator to n parameter
        if (y0>0.0001 && abs(k+l*dy)>0.0001) && F/(k+l*dy)>0.0001 %&& 1>2
            %beta=0.7;
            R=block.Dwork(5).Data;
            teta=n;
            fi=log( F / (k+l*dy) ); % !
            mu=log(y);
        end;
    end;
end

```

```

        % RLS method
        Q = R * mu / (beta + mu'*R*mu);
        R1= 1/beta * (1-Q*mu') *R;
        teta1 = teta + Q*(fi - mu'*teta); %%new parameter vector
        epszilon=fi-mu'*teta1;          %%epszilon^(t) - error

        % instricly variable update
        n2=teta1;
        block.Dwork(4).Data=n2;
        block.Dwork(5).Data = R1;
    end

    %% To output
    stiffness=n*y^(n-1)*(k+l*dy)*scala_m;
    if y0>0.0001
        stiffness=n*y^(n-1)*(k+l*dy)*scala_m;
        block.OutputPort(1).Data = stiffness;
    end;
end

if 1==block.DialogPrm(1).Data;
%% Non-parametric variable stiffness observer

    t=block.InputPort(1).Data;
    dx=block.InputPort(3).Data;
    dF=block.InputPort(5).Data;

    k0=block.Dwork(3).Data;
    dt=t-block.Dwork(7).Data;
    alpha=1500;
    dk    =-alpha*(k0*dx-dF)*sign(dx);

    k=k0+dk*dt;
    block.Dwork(7).Data=t;
    block.Dwork(3).Data=k;
    block.OutputPort(1).Data = k;
end;

```

Változó merevségű környezet. Itt is „Level-2 M file S-Function”-t használtam, hogy a portok kényelmesen felhasználhatóak legyenek a Simulink környezetben, de a tényleges számítást egy külső függvény végzi, itt mindkettőt ismertetem.

```

function Variable_stiffness_S(block)
% Level-2 M file S-Function
% inputs: t,x[2],dx/dt[2], outputs: F[2]

    setup(block);
%endfunction

function setup(block)
    %% Register dialog parameter: nums of parameter
    block.NumDialogPrms = 1;
    block.DialogPrmsTunable = {'Tunable'};
    %% Regieste number of input and output ports
    block.NumInputPorts = 3;
    block.NumOutputPorts = 2;
    %% Setup functional port properties to dynamically inherited.
    block.SetPreCompInpPortInfoToDynamic;
    block.SetPreCompOutPortInfoToDynamic;

    %InputPort(1): time: Real
    block.InputPort(1).Complexity = 'Real';
    block.InputPort(1).DataTypeId = 0;
    block.InputPort(1).SamplingMode = 'Sample';
    block.InputPort(1).Dimensions = 1;
    %InputPort(2): x: Real (displacement)
    block.InputPort(2).Complexity = 'Real';
    block.InputPort(2).DataTypeId = 0;
    block.InputPort(2).SamplingMode = 'Sample';
    block.InputPort(2).Dimensions = 2;
    %InputPort(3): dx/dt: Real (velocity)
    block.InputPort(3).Complexity = 'Real';
    block.InputPort(3).DataTypeId = 0;
    block.InputPort(3).SamplingMode = 'Sample';
    block.InputPort(3).Dimensions = 2;
    %OutputPort(1): forces(t): Real
    block.OutputPort(1).Complexity = 'Real';
    block.OutputPort(1).DataTypeId = 0;
    block.OutputPort(1).SamplingMode = 'Sample';
    block.OutputPort(1).Dimensions = 2;
    %OutputPort(2): stiffnesses(t): Real
    block.OutputPort(2).Complexity = 'Real';
    block.OutputPort(2).DataTypeId = 0;
    block.OutputPort(2).SamplingMode = 'Sample';
    block.OutputPort(2).Dimensions = 2;

```

```

%% Register methods
block.RegBlockMethod('CheckParameters', @CheckPrms);
block.RegBlockMethod('ProcessParameters', @ProcessPrms);
block.RegBlockMethod('PostPropagationSetup', @DoPostPropSetup);
block.RegBlockMethod('Start', @Start);
block.RegBlockMethod('WriteRTW', @WriteRTW);
block.RegBlockMethod('Outputs', @Outputs);
%% Block runs on TLC in accelerator mode.
block.SetAccelRunOnTLC(true);
%endfunction

function Start(block)
;
%endfunction

function CheckPrms(block)
;
%endfunction

function DoPostPropSetup(block)
%% Setup Dwork
block.NumDworks = 0;
%% Register all tunable parameters as runtime parameters.
block.AutoRegRuntimePrms;
%endfunction

function ProcessPrms(block)
block.AutoUpdateRuntimePrms;
%endfunction

function Outputs(block)

t =block.InputPort(1).Data;
x =block.InputPort(2).Data;
dx =block.InputPort(3).Data;

out=Variable_stiffness(t,x,dx);
f=out(1:2);
k=out(3:4);
%To output
block.OutputPort(1).Data = f;
block.OutputPort(2).Data = k;

```

```

function out = Variable_stifness(time,x,dx)
f=zeros(2,1);
k=zeros(2,1);

%linear spring
if 1==1
    k=4000;
    f=k.*x;
    k=[k; k];
end;

%Polinomial spring
if 1==2
    k=((x*100).^3) *8000;
    f=((x*100).^4)/4/100*8000;
end;

%Nonlinear Hunt-Crossley contactmodel
if 1==2
    xMax=0.01;
    kMin=1000; %<- =k(x~0)
    kMax=8000; %<- =k(x=xMax)
    n=1+log(kMax/kMin)/log(10); %k(x=xMax)=kMax, k(x=xMax/10=kMin)
    k0=kMax/n/xMax^(n-1);
    lambda=0;
    if x(1)>=0.00
        f(1)=x(1)^n*(k0+lambda.*dx(1));
        k(1)=n*x(1)^(n-1)*(k0+lambda.*dx(1));
    end;
    if x(2)>=0.00
        f(2)=x(2)^n*(k0+lambda.*dx(2));
        k(2)=n*x(2)^(n-1)*(k0+lambda.*dx(2));
    end;
end;

if x(1)<0
    f(1)=0;    k(1)=0;
end;
if x(2)<0
    f(2)=0;    k(2)=0;
end;
out=[f; k];

```

F.5. A kísérleti környezet vezérlőprogramja

Matematikai modul.

```
/*=====MATH UTILS=====*/
long betw(long mn,long val,long mx)
{
    if (val<mn) return mn;
    else if (mx<val) return mx;
    else return val;
}
double fbetw(double mn,double val,double mx)
{
    if (val<mn) return mn;
    else if (mx<val) return mx;
    else return val;
}
double sgn(double v)
{
    if (v>0.) return( 1.);
    if (v<0.) return(-1.);
    return(0.);
}
```

A mozgóátlag szűrők alkalmazását egy általánosan felhasználható struct típus és három függvény segíti. A főprogramban elegendő deklarálni egy filter típusú változót, majd az inicializálni a Init() függvénnyel, megadva a változó címét és a szűrő méretét. A szűrő használata mafilter() függvénnyel történik. Bemenetei: a változó címe, az új szűrendő érték, kimenete a szűrt érték. A program végén a dinamikus memória felszabadítása Destruct() függvénnyel történik.

```
/*-----Moving averages filters -----*/
#define FS      30      /* Maximal moving average filter length*/
typedef struct filter_ /* Typdef of moving average filter structure*/
{
    int num,max;        /* Currently and maximum number of used elements*/
    double *filt;       /* Elements: i=0:new, i=(max-1):old...*/
} filter;

void Init(filter *maf, int Size)
{
    if (Size<FS) maf->max = Size;
    else maf->max=FS;
    maf->filt = (double*)malloc(sizeof(double)*maf->max);
    maf->num=0;
}
```

```

double mafilter(filter *maf,double vnew)
{
    int i;
    double out=vnew;
    if (maf->max!=0)
    {
        if (maf->num==maf->max) maf->num--;
        for (i=maf->num;i>0;i--) maf->filt[i]=maf->filt[i-1];
        maf->filt[0]=vnew;
        if (maf->max!=0) maf->num++;
        for (i=0;i<maf->num;i++) out+=maf->filt[i];
        out=out/maf->num;
    }
    return out;
}

void Destruct(filter *maf)
{
    free(maf->filt);
    maf->max = 0;
}

```

Az időkésések a mozgóátlag szűrőhöz hasonló módon vannak kódolva. Minden érték késleltetéséhez szükséges egy Tdelay típusú változó deklarálása. Az `init_delay()` függvény beállítja a kezdeti értékeket, a `f_delay()`

```

/*-----Time delay-----*/
#define MAXDELAY 100
typedef struct Tdelay_ /* Typdef of time delay structure*/
{
    int i; /* Index of actual element:*/
    /* array[i]:f(t), array[i-1]=f(t-dt)*/
    double array[MAXDELAY]; /* Elements*/
    int ctau; /* Currently time delay [ms]*/
} Tdelay;

double f_delay(Tdelay *x,double vnew,int reqtau)
{
    /*Index frissítése és új elem beillesztése a pufferbe*/
    x->i++;
    if (x->i==MAXDELAY) x->i=0;
    x->array[x->i]=vnew;
    /*Compute new currently delay (=ctau)*/
    if (x->ctau < reqtau && x->ctau<MAXDELAY-1) x->ctau++;
    if (x->ctau > reqtau && x->ctau>0) x->ctau--;
    /*Return with delayed value*/
    if (x->i - x->ctau >=0 ) return x->array[x->i - x->ctau];
    else return x->array[x->i - x->ctau + MAXDELAY];
}

```

```

void init_delay(Tdelay *x)
{
    x->i=-1;
    x->ctau=-1;
}

```

TP szabályozó modul.

```

/*-----The full TP controller--model type=7-----*/
/*Inputs: ctau, IV, IFe_d+IFh, Ik_d)*/
/*Output: return with IFc*/
/*Syntax:
        Init_TP();
        ...
        IFc=tp(IV, IFm+IFe, ck, ctau); /*v, F, k-SI; tau-ms;*/
*/

static float ctheta=0;          /*Current theta value*/

static float GP_th[131]; /*Gridpoints of theta values*/
static float GP_k[137]; /*Gridpoints of stiffness values*/
static float u_k[137][3];      /*Weighting functions*/
static float u_th[131][3];     /*Weighting functions*/
static float w[3][2]; /*Computed weighting vectors: w_k[:,0], w_th[:,1] */
static float kcno[3][3][2];
static float k[2]; /*Controller gain vector*/

static int t, t2, q, i_k, i_th;

void Init_TP(void)
{
    FILE *f;

    /*-----*/
    f=fopen("TPKJT7/v_k.txt", "rt");
    for(t = 0; t<137; t++)
        fscanf(f, "%e\n", &GP_k[t]);
    fclose(f);
    /*-----*/
    f=fopen("TPKJT7/v_theta.txt", "rt");
    for(t = 0; t<131; t++)
        fscanf(f, "%e\n", &GP_th[t]);
    fclose(f);
    /*-----*/
    f=fopen("TPKJT7/ucno3_k.txt", "rt");
    for (t=0; t<137; t++)
        fscanf(f, "%e,%e,%e\n", &u_k[t][0], &u_k[t][1], &u_k[t][2]);
    fclose(f);
}

```

```

/*-----*/
f=fopen("TPKJT7/ucno3_th.txt","rt");
for (t=0;t<131;t++)
    fscanf(f,"%e,%e,%e\n", &u_th[t][0],&u_th[t][1],&u_th[t][2]);
fclose(f);
/*-----*/
f=fopen("TPKJT7/klmi.txt","rt");
for (t=0;t<3;t++)
    for (t2=0;t2<3;t2++)
        fscanf(f,"%e,%e,%e,%e\n",&kcno[t2][t][0],&kcno[t2][t][1]);
fclose(f);
}

static float ck2=0;

static double tp(double IV,double IFu,double ck,int ctau)
{
    ck2=(float)ck;
    if (ck2<700.F) ck2=700.F;
    ctheta=ctau/(1.566*120/ck2)/1000;

    /*-----Interpolation (queryw1)-----*/
    if (ck2<=GP_k[0]) for (t=0;t<3;t++) w[t][0]=u_k[0][t];
    else if (ck2>GP_k[136]) for (t=0;t<3;t++) w[t][0]=u_k[136][t];
    else
    {
        q=0;
        while (ck>GP_k[q]) q++;
        for (t=0;t<3;t++) w[t][0]=u_k[q-1][t]+((ck2-GP_k[q-1])*
            ((u_k[q][t]-u_k[q-1][t])/(GP_k[q]-GP_k[q-1])));
    }

    if (ctheta<=GP_th[0]) for (t=0;t<3;t++) w[t][1]=u_th[0][t];
    else if (ctheta>GP_th[130]) for (t=0;t<3;t++) w[t][1]=u_th[130][t];
    else
    {
        q=0;
        while (ctheta>GP_th[q]) q++;
        for (t=0;t<3;t++) w[t][1]=u_th[q-1][t]+((ctheta-GP_th[q-1])*
            ((u_th[q][t]-u_th[q-1][t])/(GP_th[q]-GP_th[q-1])));
    }
}
}

```

```

/*-----TP Controller-----*/
/*Tensor product*/
for (t=0;t<2;t++)
{
    k[t]=0.F;
    for (i_k=0;i_k<3;i_k++)
        for (i_th=0;i_th<3;i_th++)
            k[t]=k[t]+kcno[i_k][i_th][t]*w[i_k][0]*w[i_th][1];
}

return (double) (-k[0]*(float) IV-k[1]/sqrt(ck2)*(float) IFu);

```

Főprogram A főprogram elején a forrásállományok linkelése és a változók deklarálása történik meg. Több különböző TP szabályozó közül választhatunk a forrásállomány választás során.

```

/*****
/*
            Hirata TP controller 2012.10.20.
            */
/*
            KUTI JOZSEF TDK - with delayed position
            */
*****/

#include <anda.h>
#include <stdio.h>
#include <stdlib.h>
#include <math08.h>
#include <TPKJT7.h>
/*TPGP.h, TPKJT3.h, TPKJT7.h*/

#define MOD          2    /*1:direct PWM control; 2:telemanipulation*/
#define TP_MOD       1    /*0:TP off;                1: TP on*/
#define Fm_MOD       0    /*0:interaction;            1: Fm Fm from file*/
#define tau_MOD      1    /*0:constant;                1: tau from file*/
#define tau0         0    /*if (tau:constant) tau=tau0 [ms]*/
#define tau2p0       0.5 /*tau2=tau*tau2p0 */
#define filter_MOD   2    /*0: no filter, 1: MA filter, 2: Kalman filter*/

#define FEKETE 0
#define SZURKE 2
#define KEK 4

#define ILPT 0x0D
#define ICM1 0x0C
#define ICM2 0x0B
#define PORT 0x220 /*PIC Base addresse*/
#define PCM1 0x2f8 /*Serial port 1 base address*/
#define PCM2 0x3f8 /*Serial port 2 base address*/

```

```

#define RBOS 1E7                /* Hz */
#define RBPT 1000              /* timer */
#define RBPS 10                /* post scale */
#define RBTH 20                /* mm/ford */
#define RBTO 4000              /* imp/ford */
#define RBTF (RBOS/RBPT/RBPS) /* 1/Ts [Hz] */
#define DT 0.001               /* Control time base */
#define PI 3.14159             /* Pi*/

#define KORD 2                 /*Number of controlled axis*/

#define SMPM 1800              /*Size of debug register*/

/*Structure for debug*/
typedef struct {
    float deb1;
    float deb2;
    float deb3;
    float deb4;
    float deb5;
    float deb6;
    float deb7;
} Tdeb;

static void interrupt (*Rlpt)(void)=NULL;

/*-----PIDs-----*/
/* I**: data used in interrupt function, F**: data used in main funtcion */
static long Ipos[KORD],Fpos[KORD]; /*Current position from PIC*/
static long Ipsc[KORD]; /*Request position in [1]*/
static long Idps[KORD]; /*Error of position*/
static long IP=80001,II=1601,ID=320001; /*PID coefficitents*/
static long Ipro[KORD],Iitg[KORD],Idif[KORD];
/*Impropius, integral, differencial:
    Ipro=IP*Ipsc; Idif=ID*(Ipsc(t)-Ipsc(t-1)); Iitg=sum_{t} II*Ipsc(t)*/
static long Ipwm[KORD],Fpwm[KORD]; /*Computed PWM = Ipro+Iitg+Idif*/

/*----- communication with PIC-s-----*/
static byte Ichk[KORD],Ista[KORD],Fchk[KORD],Fsta[KORD]; /*PIC state display*/
static byte Icba,Fcba;
static kvad Ierr,Icnt,Ferr=01,Fcnt=01; /*Counter to controller*/
static word Ipwa[KORD],Ipwb[KORD]; /*To communication with PIC*/

const word PICE=1,IRQE=2,PICM=4|8,IRPH=128;
const word CBA=PORT+0,CAB=PORT+1;
const word PIC0=PORT+2,PIC1=PORT+3,PIC2=PORT+4;

```

```

/*-----Force sensors-----*/
static short Iem1,Iem2,Iref1,Iref2;          /*Parameters of force sensors*/
static double Iero1,Iero2;                   /*Slave & master force*/
static kvad Icnt1,Icnt2,Fcnt1=01,Fcnt2=01; /*Counters to force sensors*/

/*-----User Interface Variables-----*/
static Tdeb *Deb;
static word Dp;                               /*Current index of Deb array*/

static int  Ft,Ftau,FtauF,FtauPOS;
static float FFc,Ferol,Fero2,Fv;
static double Fk;

/*-----Virtual mechanic---all use SI-----*/
double IM=2.;                                /* mass */
double IB=120.;                              /* damping */
double IC=0.0 ;                             /* coulomb friction coefficient */
double IE=0.1 ;                             /* threshold */
static double IA=0. ,IV=0. ,IS=0.; /* acceleration, velocity, position*/
#define MSEB 0.40;                          /* Bound of velocity*/
#define MPOS 0.20;                          /* Bound of position*/
static double IFh,IFe,IFe_d,IFc=0;          /* Forces*/
static double IS_d;                          /*Delayed position to slave site*/
static double IS_em;                        /*Measured position on slave site*/

/*-----Time delay-----*/

static double gentauv[4000];                /*Request tau profile from file [ms]*/
static int i_gentauv=0;                     /*Index for gentauv*/

#define MAXDELAY 200
static double taug, ctau;                   /*Request and current summa
of variable delay [ms]: (taug=gentauv[i_gentauv])*/
static double taug1,taug2;                 /*Request variable delays: env.
force, and pos. [ms]*/
static Tdelay POSDELAY;                    /*Time delay of slave position*/
static Tdelay FDELAY,KDELAY;               /*Time delay of slave force
& environment stiffness*/

/*-----Master force-----*/
static double mf[2001];                    /*Vector of master forces*/
static int i_mf=1800;                      /*Index for Master force profil*/

/*-----Files-----*/
FILE *gentau;
FILE *mforce;

```

```

/*-----Filters-----*/

static filter Fefilt;
static filter xfilt,Fhfilt;

static double Ik=0,Ik_d; /*Estimated and deklayed stiffness*/

static double est_x,est_dx,est_F1,est_F0,est_Fap;

void Init_stiff(void)
{
    est_dx  = 0.;
    est_x   = 0.;
    est_Fap = 0.;
    est_F0  = 0.;
}

```

A program a telemanipulációs rutinnal folytatódik. Itt először az aktuális időkések értékeit határozzuk meg az előírt profil alapján.

```

/*=====*/
/*-----Telemanipulation-----*/
/*=====*/
static int i;
static void telemanipulation(void)
{
    /* Compute actual tau */
    if (tau_MOD==1)
    {   taug=gentaup[i_gentaup]*2; /*20*_ ms*/
        if(i_gentaup==3999) i_gentaup=0; else i_gentaup++;
    } else taug=tau0;
    taug2=(int) (taug*tau2p0);
    taug1=taug-taug2;
}

```

Majd a szolga oldali, szűrt erőt és a megfigyelt merevségértékét késleltetjük, és kiszámítjuk a jelenlegi időkésést.

```

/* Computation of the delayed slave force */
IFe_d=f_delay(&FDELAY,IFe,(int)taug1);
Ik_d=f_delay(&KDELAY,Ik,(int)taug1);
ctau=FDELAY.ctau+POSDELAY.ctau;

```

A futtatás módjától függően meghatározzuk a szabályozóerőt a TP szabályozó modul `tp()` függvényével. Majd kiszámítjuk az impedanciamodell gyorsulását. Téglány integrátorok felhasználásával ebből meghatározható az új sebesség és pozíció. Biztonsági okokból ezek értékét

korlátozzuk.

```
if (TP_MOD==1) IFc=tp(IV, (IFe+IFh), Ik_d, ctau);
else IFc=0;

/*Impedanciamodell: compute acceleration (IA),
                    velocity (IV) & position (IS) */
IA=(double) ((IFh+IFe_d+IFc)/IM)-(double) (IB*IV/IM);

IV+=IA/RBTF;
if (IV<-MSEB) IV=-MSEB;
if (IV> MSEB) IV= MSEB;

IS+=IV/RBTF;
if (IS<-MPOS) IS=-MPOS;
if (IS> MPOS) IS= MPOS;
```

A szolga oldal csak a késleltetett pozícióértékhez fér hozzá. Ezt mozgóátlag szűrővel simítjuk el.

```
/* Computation of the delayed slave position */
IS_d=f_delay(&POSDELAY, IS, (int)taug2);

/*Moving average filtering of delayed slave position*/
IS_d=mafilter(&xfilt, IS_d);
```

Mentjük a folyamat fontosabb jellemzőit.

```
/*Nyomkovetes mentese*/
if (Dp<SMPM-1)
{
    Deb[Dp].deb1=Ipos[0]; Deb[Dp].deb2=Ipos[1]; Deb[Dp].deb3=IFe;
    Deb[Dp].deb4=IFh;      Deb[Dp].deb5=IFc;      Deb[Dp].deb6=ctau;
    Deb[Dp].deb7=Ik;
    Dp++;
}
```

A mester és szolga oldali pozíció alapjelekhez, meghatározzuk a PID szabályozók beavatkozó jelét, majd a PWM jel kitöltési tényezőjét.

```

/*Compute reference positons (in [1])*/
Ipsc[0]=(long) (IS_d*1000*RBTO/RBTH);
Ipsc[1]=(long) (IS*1000*RBTO/RBTH);

/*PID controller*/
for (i=0; i<KORD; i++)
{
    Ipwm[i]=0l;

    Idif[i] =-ID*Idps[i];
    Idps[i] =Ipsc[i]-Ipos[i];
    Ipro[i] =IP*Idps[i];
    Idif[i]+=ID*Idps[i];
    Iitg[i]+=II*Idps[i];

    Ipro[i]=betw(-1024l*RBPT,Ipro[i],1024l*RBPT);
    Iitg[i]=betw(-1024l*RBPT,Iitg[i],1024l*RBPT);
    Idif[i]=betw(-1024l*RBPT,Idif[i],1024l*RBPT);

    Ipwm[i]=betw(-1000l*RBPT,Ipro[i]+Iitg[i]+Idif[i],1000l*RBPT)/256;
}
}

```

A mentett adatok *.txt fájlba írása.

```

void debmem(void)
{
    word i;
    FILE *fp;

    fp=fopen("dipl1133b.txt","w");
    for (i=0; i<SMPPM; i++)
        fprintf(fp,"%d\t%3.0f\t%3.0f\t%4.4f\t%4.4f\t%4.4f\t%4.4f\t%4.4f\n",
            i,Deb[i].deb1,Deb[i].deb2,Deb[i].deb3,Deb[i].deb4,
            Deb[i].deb5,Deb[i].deb6,Deb[i].deb7);
    fclose(fp);
}

```

Az interrupt függvény első felében beolvassuk az erőmérőcellák mérési eredményeit és a PIC kontrollerektől az enkóderek értékét.

```

static void interrupt lptirp(void)
{
    static byte a,picm,deb;
    static int i;
    if (Ierr>0) goto irpv;
    /* Interrupt kezdes */
    if (Icba==0) Icba=inportb(CBA)&1;
    if (Icba!=0)
    {
        outportb(CAB,~0);
        goto irpv;
    }
    outportb(CAB,~(PICE|IRQE|PICM|IRPH));
    Icnt++;
    /* Beolvasas a kontrollertol */
    for (picm=PICM ,i=0; (inportb(CBA)&PICM)!=PICM&&i<1000; i++);
    if (i>=1000) Ierr|=1;
    Ichk[0]=inportb(PIC0);
    Ichk[1]=inportb(PIC1);
    for (picm^=PICM,i=0; (inportb(CBA)&picm)!=0 &&i<1000; i++);
    if (i>=1000) Ierr|=2|256;

    if (PICM& 4) Ista[0]=inportb(PIC0);
    if (PICM& 8) Ista[1]=inportb(PIC1);

    if (PICM& 4) Ipos[0] =inportb(PIC0);
    if (PICM& 8) Ipos[1] =inportb(PIC1);

    if (PICM& 4) Ipos[0]+=(long)inportb(PIC0)<<8;
    if (PICM& 8) Ipos[1]+=(long)inportb(PIC1)<<8;

    if (PICM& 4) Ipos[0]+=(long)inportb(PIC0)<<16;
    if (PICM& 8) Ipos[1]+=(long)inportb(PIC1)<<16;

    if (PICM& 4) Ipos[0]+=(long)inportb(PIC0)<<24;
    if (PICM& 8) Ipos[1]+=(long)inportb(PIC1)<<24;

    for (picm^=PICM,i=0; (inportb(CBA)&PICM)!=picm&&i<1000; i++);
    if (i>=1000) Ierr|=2|512;

    /*-----Beolvasas az eromeroltol-----*/
    /*PCM1-es portrol olvaso eromerő van kalibrálva*/
    for (; (inportb(PCM1+5)&1)!=0; Icnt1++)
    {
        a=inportb(PCM1);
        if ((a&192)==128) {
            Iem1=64*(int)(a&63);

```

```

        continue;
    }
    if ((a&192)== 64) {
        Iem1+=(int) (a&63);
        Iero1=(double) (Iref1-Iem1)/3.772+3.9;
        continue;
    }
}

/*-----Beolvasas az eromero2tol-----*/
/*PCM2-es portról olvasó erőmérő van kalibrálva*/
for (; (inportb(PCM2+5)&1)!=0; Icnt2++)
{
    a=inportb(PCM2);
    if ((a&192)==128) {
        Iem2=64*(int) (a&63);
        continue;
    }
    if ((a&192)== 64) {
        Iem2+=(int) (a&63);
        Iero2=(double) (Iref2-Iem2)*0.337875;
        continue;
    }
}

```

Az erőszűrések és a merevségbecslő algoritmus is itt került megvalósításra az általános felhasználhatóság érdekében. A mért mester oldali erő helyett lehetőségünk van fájlból beolvasott erőértékek használatára.

```

/*----- Fe filter & Stiffness estimator-----*/
IFe=Iero1;
IS_em=(double) (Ipos[0])/1000/RBTO*RBTH;
est_dx = IS_em - est_x;
est_x = IS_em;

if (filter_MOD==1)    IFe=mafilter(&Fefilt,IFe);
if (filter_MOD==2)
{
    if (Ik>500) est_Fap = est_F0 - Ik * est_dx * 1.00;
    else est_Fap = est_F0;
    est_F1 = est_Fap + 0.045*( IFe - est_Fap );
    IFe=est_F1;
}

Ik      = Ik - 1500 * ( Ik * est_dx + (IFe - est_F0))*sgn(est_dx);
est_F0  = IFe;
Ik      = fbetw(0.,Ik,18000.);

```

```

/*-----Force threshold-----*/
if (IFe>-IE && IFe<IE) IFe=0.;
    else if (IFe>= IE) IFe-=IE;
        else if (IFe<=-IE) IFe+=IE;

/*-----Master force-----*/
if (Fm_MOD==1)
{ IFh=-mf[i_mf]*2;
    i_mf++;
    if ( i_mf > ( sizeof(mf) / sizeof(double) ) ) i_mf=0;
}
else
{
    IFh=Iero2;
    IFh=mafilter(&Fhfilt, IFh);
}

/*=====*/
/*-----Program mods-----*/
/*=====*/

/*-----DIREKT PWM KIADAS-----*/
if (MOD==1)
{
    Ipwm[0]=0; IFe_d=IFe;
}

/*-----Telemanipulation -----*/

if (MOD==2) telemanipulation();

```

A kiszámított beavatkozó jelek értékének továbbítása a kontrollereknek, majd az interruptok engedélyezése.

```

/*-----Kiiras a szabalyozoba-----*/

for (i=0; i<KORD; i++)
{
    Ipwa[i]=0;
    Ipwb[i]=0;

    if (Ipwm[i]>-40&&Ipwm[i]<40) Ipwm[i]=01;
    if (Ipwm[i]<-3900) Ipwm[i]=-3900;
    if (Ipwm[i]> 3900) Ipwm[i]= 3900;
    if (Ipwm[i]<0) Ipwa[i]=labs(Ipwm[i]);
    if (Ipwm[i]>0) Ipwb[i]=labs(Ipwm[i]);
}

```

```

if (PICM& 4) outportb(PIC0, (byte) 0xa5);
if (PICM& 8) outportb(PIC1, (byte) 0x5a);

for (picm^=PICM, i=0; (inportb(CBA)&PICM)!=picm&&i<1000; i++);
if (i>=1000) Ierr|=2|1024;

if (PICM& 4) outportb(PIC0, (byte) Ipwa[0]);
if (PICM& 8) outportb(PIC1, (byte) Ipwa[1]);
if (PICM& 4) outportb(PIC0, (byte) (Ipwa[0]>>8));
if (PICM& 8) outportb(PIC1, (byte) (Ipwa[1]>>8));
if (PICM& 4) outportb(PIC0, (byte) Ipwb[0]);
if (PICM& 8) outportb(PIC1, (byte) Ipwb[1]);
if (PICM& 4) outportb(PIC0, (byte) (Ipwb[0]>>8));
if (PICM& 8) outportb(PIC1, (byte) (Ipwb[1]>>8));

for (picm^=PICM, i=0; (inportb(CBA)&PICM)!=picm&&i<1000; i++);
if (i>=1000) Ierr|=2|2048;

outportb(CAB, ~(PICE|IRQE|PICM|deb));

/*-----Interrupt vege-----*/

irpv:
    outportb(0x20, 0x20);
}

```

Képernyőkezelés.

```

int frissit(void)
{
    static int j;

    disable();
    Ferr=Ierr;    Fcba=Icba;    Fcnt=Icnt;    Fcnt1=Icnt1;    Fcnt2=Icnt2;
    for (j=0; j<KORD; j++)
        Fpos[j]=Ipos[j], Fpwm[j]=Ipwm[j], Fchk[j]=Ichk[j], Fsta[j]=Ista[j];
    Ft=Dp; FFc=IFc; Ftau=ctau; FtauF=FDELAY.ctau;
    Fero1=IFe_d; Fero2=IFh; Fv=IV; FtauPOS=POSDELAY.ctau;
    Fk=Ik;
    enable();

    sprintf(buf1, "%9ld    %9ld [1]", Fpos[0], Fpos[1]);
    asawoutc(32, 10, 5, 10, ABLAK);
    asawout(buf1, 10, 5, ABLAK);
}

```

```

sprintf(buf1,"%9.0ld      %9.0ld",Fpwm[0],Fpwm[1]);
asawoutc(32 ,10, 6,10,ABLAK);
asawout (buf1,10, 6,   ABLAK);

sprintf(buf1,"%9.3lf      %9.3lf [N]",Fero1,Fero2);
asawoutc(32 ,10, 7,10,ABLAK);
asawout (buf1,10, 7,   ABLAK);

sprintf(buf1,"%9.3lf [N]",FFc);
asawoutc(32 ,10, 9,10,ABLAK);
asawout (buf1,10, 9,   ABLAK);

sprintf(buf1,"%9.3lf [N]",Fero1+Fero2);
asawoutc(32 ,10,10,10,ABLAK);
asawout (buf1,10,10,   ABLAK);

sprintf(buf1,"%9.3lf [m/s]",Fv);
asawoutc(32 ,10,11,10,ABLAK);
asawout (buf1,10,11,   ABLAK);

sprintf(buf1,"%9.3lf [N/m]",Fk);
asawoutc(32 ,10,12,10,ABLAK);
asawout (buf1,10,12,   ABLAK);

sprintf(buf1,"%2.3lf [-]",Fk*Ftau/1.566/120/1000);
asawoutc(32 ,10,13,10,ABLAK);
asawout (buf1,10,13,   ABLAK);

sprintf(buf1,"%3.0d [ms]",Ftau);
asawoutc(32 ,10, 14,10,ABLAK);
asawout (buf1,10, 14,   ABLAK);

sprintf(buf1,"%3.0d      %3.0d [ms]",FtauF,FtauPOS);
asawoutc(32 ,10,15,10,ABLAK);
asawout (buf1,10,15,   ABLAK);

sprintf(buf1,"%9.0d [ms]",Ft);
asawoutc(32 ,10,16,10,ABLAK);
asawout (buf1,10,16,   ABLAK);

/*Plot of pic state*/
sprintf(buf1,"%10lu",Fcnt);
/*asawoutc(32 ,65, 0, 6,MABLAK);*/
asawoutd(buf1,65, 0, 6,ABLAK);

sprintf(buf1,"%10lu",Ferr);
asawoutc(32 ,65, 1, 6,ABLAK);
asawout (buf1,65, 1,   ABLAK);

```

```

    sprintf(buf1,"%3lu",Fcnt&255);
    asawoutc(32 ,65, 2, 6,ABLAK);
    asawout (buf1,65, 2,    ABLAK);

    sprintf(buf1,"%3d %3d %3d",Fchk[0],Fchk[1],Fchk[2]);
    asawoutc(32 ,65, 3,11,ABLAK);
    asawout (buf1,65, 3,    ABLAK);

    sprintf(buf1,"%3d %3d %3d",Fsta[0],Fsta[1],Fsta[2]);
    asawoutc(32 ,65, 4,11,ABLAK);
    asawout (buf1,65, 4,    ABLAK);

    sprintf(buf1,"%10lu",Fcnt1);
    asawoutc(32 ,65, 5, 6,ABLAK);
    asawout (buf1,65, 5,    ABLAK);

    sprintf(buf1,"%10lu",Fcnt2);
    asawoutc(32 ,65, 6, 6,ABLAK);
    asawout (buf1,65, 6,    ABLAK);

    return (Fcba&1);
}

void kikep(void)
{
    winalap("DC Servo");
    aswsetwin(1,3,78,21);
    asawcls(32,ABLAK);
}

```

Interruptbeállító eljárások.

```

void ilptbe(void)
{
    byte a;
    if (Rlpt!=NULL) return;
    disable();
    Rlpt=getvect (ILPT);
    setvect (ILPT,lptirp);
    a=inportb(0x21);
    a&=~0x20;
    outportb(0x21,a);
    enable();
}

```

```

void ilptki(void)
{
    byte a;
    disable();
    a=inportb(0x21);
    a|=0x20;
    outportb(0x21,a);
    if (Rlpt!=NULL) setvect(ILPT,Rlpt);
    Rlpt=NULL;
    enable();
}

```

A main() függvény.

```

/*=====*/
/*-----MAIN-----*/
/*=====*/
void main(void)
{   int ch,p,j;
    static int t;
    long i;
    byte a,d,d0,d1;

    /*=====*/
    /*-----System init-----*/
    /*=====*/
    INITASP
    winbelep (VM80x25);
    att[ABLAK]=16*SZURKE+FEKETE;
    kikep();

    /*=====*/
    /*-----Clearing variables-----*/
    /*=====*/
    Icba=0,
    Ierol=0.;
    for (j=0; j<KORD; j++)
        Ipos[j]=0,Ipwa[j]=0,Ipwb[j]=0,Ipsc[j]=0,Idps[j]=0,Iitg[j]=0,Ipwm[j]=0;
    Ierr=0,Icnt=Icnt1=Icnt2=0;
    Deb=NULL;
    Dp=0;
    Deb=calloc(SMPM,sizeof(Tdeb));
    if (Deb==NULL) goto HB;
    Init(&Fefilt,30);      /*Init & reset MA filters*/
    Init(&xfilt,40);
    Init(&Fhfilt,20);

```

```

Init_stiff();          /*Reset variables of stiffness estimator*/
Init_TP();             /*Init variables of TP controller*/
init_delay(&POSDELAY); /*Init timedelays*/
init_delay(&KDELAY);
init_delay(&FDELAY);

/*=====*/
/*-----Reading tau and Fm profiles from files-----*/
/*=====*/
gentau=fopen("mod6315b/gt6315b.txt","rt");
for(t = 0;t<4000;t++)
{
    fscanf(gentau,"%lf\n", &gentauv[t]);
    gentauv[t]*=1000;          /*from [s] to [ms]*/
}
fclose(gentau);
/*-----*/
mforce=fopen("mforce15.txt","rt");
for(t = 0;t<2001;t++)  fscanf(mforce,"%le\n", &mf[t]);
fclose(mforce);

/*=====*/
/*-----Controller init-----*/
/*=====*/
outportb(CAB,~(0));
outportb(PIC0,(byte)0);
outportb(PIC1,(byte)0);

asawoutd("PIC bekapcsolasa <ENTER>",5, 0, 70,ABLAK);
for (; asinkey()!=ENT;) outportb(CAB,~(0));

outportb(CAB,~(PICE));

for (i=0; i<40; i++) {
    outportb(CAB,~(PICE));
}
outportb(CAB,~(PICE|IRPH));
for (d0=(inportb(CBA)&PICM),d1=0,i=3001; i>0; i--) {
    d=inportb(CBA)&PICM;
    d1=(~d0)&d;
    d0=d;
    if (d1&4  && KORD>0 ) Ichk[0]++;
    if (d1&8  && KORD>1 ) Ichk[1]++;
    if (d1&16 && KORD>2 ) Ichk[2]++;
}
outportb(CAB,~(PICE));
sprintf(buf1,"%d %d %d", (KORD>0?Ichk[0]:0), (KORD>1?Ichk[1]:0),
(KORD>2?Ichk[2]:0));

```

```

asawoutd(buf1,5,1,20,ABLAK);

j=0;
if (PICM&4 &&KORD>0) if (Ichk[0]!=3) j=1;
if (PICM&8 &&KORD>1) if (Ichk[1]!=3) j=1;
if (PICM&16&&KORD>2) if (Ichk[2]!=3) j=1;
if (j>0) {
    asawoutd("A PIC nem talalhato <ESC>",5, 0, 70,ABLAK);
    for (; asinkey()!=ESC;);
    goto VG;
}

/*=====*/
/*-----Eromero1 inic-----*/
/*=====*/
outportb(PCM1+3,0x00);
outportb(PCM1+1,0x00);
outportb(PCM1+2,0x00);
outportb(PCM1+4,0x00);
outportb(PCM1+3,0x80); /* DLA */
outportb(PCM1+1,0x00);
outportb(PCM1+0,0x03);
outportb(PCM1+3,0x03); /* 8,N,1 */
outportb(PCM1+1,0x01);
outportb(PCM1+2,0x47);
outportb(PCM1+4,0x00);

for (j=0; j<100; j++) inportb(PCM1);
p=10000;
Iref1=0;
for (ch=0,j=0; ch<5&&j<p; ch++) {
    Iem1=0;
    for (; j<p; j++) {
        if ((inportb(PCM1+5)&1)==0) continue;
        a=inportb(PCM1);
        if ((a&192)==128) {
            Iem1=64*(int) (a&63);
            break;
        }
    }
    for (; j<p; j++) {
        if ((inportb(PCM1+5)&1)==0) continue;
        a=inportb(PCM1);
        if ((a&192)== 64) {
            Iem1+=(int) (a&63);
            break;
        }
    }
}

```

```

        Iref1+=Iem1;
    }

    if (j<p) {
        Iref1=Iref1/ch;
    } else {
        asawoutd("Az eromero1 nem talalhato  <ESC>",5, 0, 70,ABLAK);
        for (; asinkey()!=ESC);
        goto VG;
    }

    /*sprintf(buf,"Irf1=%d  <ESC>",Irf1);
    asawoutd(buf,5,0,70,ABLAK);
    for(;asinkey()!=ESC);*/
    Iref1=1963;

    /*=====*/
    /*-----Eromero2 inic-----*/
    /*=====*/
    outportb(PCM2+3,0x00);
    outportb(PCM2+1,0x00);
    outportb(PCM2+2,0x00);
    outportb(PCM2+4,0x00);
    outportb(PCM2+3,0x80); /* DLA      */
    outportb(PCM2+1,0x00);
    outportb(PCM2+0,0x03);
    outportb(PCM2+3,0x03); /*      8,N,1 */
    outportb(PCM2+1,0x01);
    outportb(PCM2+2,0x47);
    outportb(PCM2+4,0x00);

    for (j=0; j<100; j++) inportb(PCM2);
    p=10000;
    Iref2=0;
    for (ch=0,j=0; ch<5&&j<p; ch++)
    {   Iem2=0;
        for (; j<p; j++)
        {   if ((inportb(PCM2+5)&1)==0) continue;
            a=inportb(PCM2);
            if ((a&192)==128) {
                Iem2=64*(int) (a&63);
                break;
            }
        }
    }
    for (; j<p; j++) {
        if ((inportb(PCM2+5)&1)==0) continue;
        a=inportb(PCM2);

```

```

        if ((a&192)== 64) {
            Iem2+=(int) (a&63);
            break;
        }
    }
    ref2+=Iem2;
}

if (j<p) {
    Iref2=Iref2/ch;
} else {
    asawoutd("Az eromero2 nem talalhato  <ESC>",5, 0, 70,ABLAK);
    for (; asinkey()!=ESC;);
    goto VG;
}

/*=====*/
/*-----Erosito inic-----*/
/*=====*/
asawoutd("Kapcsold be az erositot ! <ESC> megsem",5, 0, 70,ABLAK);

for (; asinkey()!=ESC;) {
    d=inportb(CBA)&1;
    if (d==0) break;
}
if (d!=0) goto VG;

/*=====*/
/*-----Vezerles inditasa-----*/
/*=====*/

asawoutd("Vezerles inditasa <ENTER>  <ESC> megsem",5, 0, 70,ABLAK);
for (; (ch=asinkey())!=ENT&&ch!=ESC;);
if (ch!=ENT) goto VG;

asawoutc(32,5, 0,70,ABLAK);

if (MOD==2)
{
    asawout("  TELEMANIPULATION", 5, 1,ABLAK);
    if (TP_MOD==0) asawout("w/o TP control", 5, 2,ABLAK);
    else asawout("with TP control", 5, 2,ABLAK);
    if (Fm_MOD==0) asawout("Interaction!", 21, 2,ABLAK);
    else asawout("Fm from file!", 21, 2,ABLAK);
}

```

```

asawout ("          slave          master", 5, 4, ABLAK);
asawout ("POS   :", 5, 5, ABLAK);
asawout ("PWM   :", 5, 6, ABLAK);
asawout ("F     :", 5, 7, ABLAK);

asawout ("Fc    :", 5, 9, ABLAK);
asawout ("Fe+Fh:", 5, 10, ABLAK);
asawout ("v     :", 5, 11, ABLAK);
asawout ("k     :", 5, 12, ABLAK);
asawout ("theta:", 5, 13, ABLAK);
asawout ("tau   :", 5, 14, ABLAK);
asawout ("tauFX:", 5, 15, ABLAK);
asawout ("t_DEB:", 5, 16, ABLAK);

```

A végtelen cilus, amely során engedélyezett az interruptok végrehajtása, csak az ESC billentyűvel megszakítható.

```

outportb(CAB, ~(PICE | IRQE | PICM));
ilptbe();
for (; asinkey() != ESC;)
{
    frissit();
    delay(2000);
}
outportb(CAB, ~(PICE));
delay(1);
ilptki();
delay(1);

```

A kommunikáció befejezése, a naplózott értékek fájlba írása, majd a dinamikus memória felszabadítása.

```

VG:
    for (j=0; j<KORD; j++) {
        outportb(PIC0+j, 0x0c);
        Fpos[j]=inportb(PIC0+j);
        sprintf(buf, "%3d", Fpos[j]);
        asawoutd(buf, 65, 10+j, 3, ABLAK);
    }
    outportb(CAB, ~0);
    debmem();
HB:
    if (Deb!=NULL) free(Deb);
    Destruct(&xfilt); Destruct(&Fefilt); Destruct(&Fhfilt);
    winkilep();
}

```