MISKOLCI EGYETEM
GÉPÉSZMÉRNÖKI ÉS INFORMATIKAI KAR

ROBOTIZÁLT RENDSZEREK ELMÉLETI ÉS GYAKORLATI ELEMZÉSE

PHD ÉRTEKEZÉS

Készítette:

Cservenák Ákos Dániel
okleveles mechatronikai mérnök (MSc)
okleveles járműmérnök (MSc)

SÁLYI ISTVÁN GÉPÉSZETI TUDOMÁNYOK DOKTORI ISKOLA
GÉPEK ÉS SZERKEZETEK TERVEZÉSE
MECHATRONIKAI RENDSZEREK TERVEZÉSE TÉMACSOPORT

Doktori Iskola vezető:

Vadászné Prof. Dr. Bognár Gabriella
a műszaki tudomány doktora, egyetemi tanár

Témacsoport vezető:

Dr. Szabó Tamás
ny. egyetemi docens

Tudományos témavezető:

Dr. Szabó Tamás
ny. egyetemi docens

Miskolc
2021

TARTALOMJEGYZÉK

NYILATKOZAT

Alulírott Cservenák Ákos Dániel büntetőjogi felelősségem tudatában kijelentem, hogy a Sályi István Gépészeti Tudományok Doktori Iskolába beadott „Robotizált rendszerek elméleti és gyakorlati elemzése” című PhD értekezés önálló munkám eredménye, az irodalmi hivatkozások egyértelműek és teljesek.

Dátum: Miskolc, 2021. március 12.

……

Cservenák Ákos Dániel

TÉMAVEZETŐ AJÁNLÁSA

Cservenák Ákos 2010 szeptemberében került a Miskolci Egyetemre, mint alapszakos mechatronikai mérnök hallgató. A BSc-s és MSc-s hallgatói általános kötelezettségen felül, több TDK dolgozatot adott be, amelyeken sikeresen szerepelt, egyszer II. helyezett, négyszer III. helyezett lett. Az OTDK konferenciákon is eredményesen szerepelt, 2015-ben szekciójában különdíjat kapott. Az alapszakos- és a mesterszakos képzése során Ákos 4 alkalommal is elnyerte a Köztársasági ösztöndíjat és a Kar által adományozott Tanulmányi emlékérem kitüntetés arany fokozatát 4x, ezüst és bronz fokozatát 1-1x.

A tanszéken főként robotos kutatási témában kapcsolódott be, már BSc-s hallgató korától kezdve. Számos publikációja jelent meg, amelyet az MTMT adatbázisa is tartalmaz.

A doktori képzésre 2016/17 I. félévében nyert felvételt. A kutatási témájának a címe: „Robotizált rendszerek elméleti és gyakorlati elemzése”.

A doktori képzés első évében egy egynyomvonalú járműmodell lineáris és nemlineáris dinamikájával foglalkozott, amellyel egy Scopus által referált, Q4-es konferencia kiadványba publikált. A képzés második évében kezdett el foglalkozni a vezető nélküli targoncák témakörével, amelyből első komolyabb eredménye egy Scopus által referált, Q4-es konferenciacikk lett. A PhD képzéssel párhuzamosan elvégezte a győri Széchenyi István Egyetemen a járműmérnöki mesterszakot. Ami átmenetileg ugyan visszafogta a PhD kutatásaiban való intenzív előrehaladását, de a komplex vizsga követelményeinek teljesítésében ez nem okozott fennakadást.

A kutatási fázisba lépve Ákos további irodalom feldolgozásokat végzett, majd a Logisztika Intézetben található vezető nélküli targonca kinematikájával, dinamikájával és vezérlésével kezdett el foglalkozni, amelyet nagy önállósággal végzett. Az eredetileg tervezett négy év után a plusz egy év doktori kutatásának hosszabbítása részben a COVID járvány, illetve az első két évben folytatott járműmérnöki tanulmányokra fektetett energiával magyarázható. Ez utóbbi talán nem volt haszontalan a targonca kutatásainak végzésében sem.

A főbb eredményeit Q2, illetve Q3 minőségű folyóiratokban publikálta. A Manifacturing Technology című Q2 folyóiratban elfogadott cikk a vezető nélküli targonca pálya- és trajektória tervezésének lépéseit írja le 22 oldalas terjedelemben, amely a disszertációjának egyik lényeges részét foglalja magában.

További cikkeket publikált az Academic Journal of Manufacturing Engineering folyóiratba 3 alkalommal, amelyből 1 már megjelent. Ebből 2 cikk a disszertációban tárgyalt modulrendszer grafikusan programozott részét írja le, míg a 3. cikk a mérési rendszer kialakítását tárgyalja.

Az oktatásban is aktívan részt vett hallgató korában demonstrátorként, egy félévig mérnöktanárként és a Phd hallgatói jogviszonya alatt mind a 4 évében. A tanszéki mechatronikai tantárgyak angol és magyar nyelvű gyakorlatának vezetése, valamint hallgatók szakdolgozatának, diplomamunkájának, illetve projektfeladatának konzultálása mellett oktatási segédletet is készített alapszakos hallgatók részére. A doktori kutatómunkáján felül Ákos egy Bosch Power Tool Kft által kezdeményezett robot témájú projektben is részt vett.

A doktori képzésben eltöltött ideje során Ákos publikációs tevékenységével is bizonyította, hogy alkalmas az önálló tudományos kutatómunkára a mechatronika területén. Mint témavezetője megerősítem, hogy a disszertációja az ő saját hiteles eredményeit tartalmazza és formailag megfelel a PhD disszertációkkal szemben támasztott követelményeknek, továbbá javaslom a nyilvános vitára bocsátását, majd elfogadását.

Miskolc, 2021. március 12.

……

Dr. Szabó Tamás
tudományos témavezető
ny. egyetemi docens, óraadó

JELÖLÉSEK JEGYZÉKE

Skalár mennyiségek

Jelölés	Név
X_0, Y_0	Pályagörbe kezdőpontjának *X* és *Y* koordinátája
X_1, Y_1	Pályagörbe 1. vezérlőpontjának *X* és *Y* koordinátája
X_2, Y_2	Pályagörbe 2. vezérlő pontjának *X* és *Y* koordinátája
X_3, Y_3	Pályagörbe végpontjának *X* és *Y* koordinátája
u	Paraméter a görbék esetén szemléletesen a $\boldsymbol{P}_B(u)$ vagy $\boldsymbol{P}_H(u)$ távolságára $\boldsymbol{P}_0$ ponttól $\boldsymbol{P}_3$ irányába
$X_B(u), Y_B(u)$	Bezier-görbe függvényének *X* és *Y* koordinátája
F_0, F_1, F_2, F_3	$\boldsymbol{F}$ vektor 0., 1., 2. és 3. együtthatója
X_{S1}, Y_{S1}	Pályagörbe 1. vezérlővektorának *X* és *Y* koordinátája
X_{2E}, Y_{2E}	Pályagörbe 2. vezérlővektorának *X* és *Y* koordinátája
$X_H(u), Y_H(u)$	Hermite-görbe függvényének *X* és *Y* koordinátája
X_{start}, Y_{start}	Targonca kezdeti pozíciójának *X* és *Y* koordinátája a kerekek felezőpontjánál
φ_{start}	Targonca kezdeti orientációja
X_{end}, Y_{end}	Targonca célpozíciója *X* és *Y* koordinátája a kerekek középvonalánál
φ_{end}	Targonca célorientációja
φ_{S1-SE}	$\boldsymbol{P_{S1}}$ és $\boldsymbol{P_{SE}}$ vektorok által bezárt szög
φ_{2E-SE}	$\boldsymbol{P_{2E}}$ és $\boldsymbol{P_{SE}}$ vektorok által bezárt szög
$X_{start,LIDAR}$, $Y_{start,LIDAR}$	LIDAR szenzor által mért kezdeti pozíció *X* és *Y* koordinátája
$X_{end,LIDAR}$, $Y_{end,LIDAR}$	LIDAR szenzor által mért és beállított célpozíció *X* és *Y* koordinátája
$s_{LIDAR-W}$	Kerekek középvonala és a LIDAR szenzor középpontja közötti távolság
$s_{belt_1-LIDAR}$	1. szállítószalag középpontja és a LIDAR szenzor középpontja közötti távolság
$s_{belt_1-belt_2}$	Két szállítószalag középpontjának távolsága

$X_{end,belt1}, Y_{end,belt1}$	1. szállítószalag középpontjának *X* és *Y* koordinátája
$X_{end,belt2}, Y_{end,belt2}$	2. szállítószalag középpontjának *X* és *Y* koordinátája
s_{belt_1-W}	1. szállítószalag középpontja és kerekek középvonala közötti távolság
s_{belt_2-W}	2. szállítószalag középpontja és kerekek középvonala közötti távolság
u_i	Paraméter útpontsorozat esetén szemléletesen a $X(u_i)$, $Y(u_i)$ távolságára X_{i-1}, Y_{i-1} ponttól X_i, Y_i irányába
i	Útpont sorszáma az útpontsorozaton belül
N	Legutolsó útpontot jelölő szám
$X_{i-1}, Y_{i-1}, \varphi_{i-1}$	Útpontsorozat *i-1*. elemének *X* és *Y* koordinátája és orientációja
X_i, Y_i, φ_i	Útpontsorozat *i*. elemének *X* és *Y* koordinátája és orientációja
X_{ai}, Y_{ai}	Útpontsorozat *i*. elemének 1. vezérlővektorának *X* és *Y* koordinátája
X_{bi}, Y_{bi}	Útpontsorozat *i*. elemének 2. vezérlővektorának *X* és *Y* koordinátája
$X(i), Y(i)$	Útpontsorozat *i*. elemének Bezier-görbe adott pontjának *X* és *Y* koordinátája
$X(u_i), Y(u_i)$	Útpontsorozat *i*. elemének Bezier-görbe függvényének *X* és *Y* koordinátája
S	Szegmens sorszáma
N_S	Szegmensek darabszáma
$X^1_{start,LIDAR}$, $Y^1_{start,LIDAR}$	*1*. szegmens LIDAR szenzor által mért kezdeti pozíció *X* és *Y* koordinátája a LIDAR szenzor középpontjában
$X^S_{end,LIDAR}$, $Y^S_{end,LIDAR}$	*S*. szegmens LIDAR szenzor által mért kezdeti pozíció *X* és *Y* koordinátája a LIDAR szenzor középpontjában
φ^1_{start}	*1*. szegmens LIDAR szenzor által mért kezdeti orientáció
φ^S_{start}	*S*. szegmens számított kezdeti orientáció
$\varphi^1_{end}, \varphi^2_{end}$, $\varphi^{S-1}_{end}, \varphi^S_{end}$	*1*., *2*., *S-1*. és *S*. szegmens célorientációja
X^1_{end}, Y^1_{end} X^S_{end}, Y^S_{end}	*1*. és *S*. szegmens célpozíciójának *X* és *Y* koordinátája a kerekek közötti felezőpontban
j	Adott útpont sorszáma a görbe egyes szegmensein belül
u_{end}	u paraméter intervallumának felosztásából keletkező lépések száma

s^S	*S*. szegmens teljes úthossza
$s^S_{diff}(j)$	Pályagörbe két szomszédos útpontja közötti távolság az *S*. szegmens *j*. útpontjánál
s_{teljes}	Teljes pályagörbe úthossza
t_{01}, t_{23}	Szegmensek *1*. és *3*. szakaszának időtartama
t^S_{12}	*S*. szegmens *2*. szakaszának időtartama
$s^S_{01}, s^S_{12}, s^S_{23}$	*S*. szegmens *1*., *2*. és *3*. szakaszának úthossza
v_{max}, a_{max}	Targonca előírt maximális sebessége és gyorsulása
t^S_{03}	*S*. szegmens teljes, azaz *1-3*. szakaszának időhossza
$s^S_{actual}(1)$	*S*. szegmens *1*. útpontjánál aktuális úthossz
$s^S_{actual}(j)$	*S*. szegmens *j*. útpontjánál aktuális úthossz
k^S_{start}	Iránykoefficiens a maximális sebesség előjelének megállapításához
$t^S(j)$	*S*. szegmens *j*. útpontjánál aktuális idő
$v^S(j), a^S(j)$	*S*. szegmens *j*. útpontjánál aktuális sebesség és gyorsulás
k	Adott útpont sorszáma a teljes, több szegmensből álló görbén belül
$X_{actual}(k)$, $Y_{actual}(k)$	Teljes pályagörbe *k*. útpontjának aktuális *X* és *Y* koordinátája a *k-1*. útponthoz képest
$X_{vertical}, Y_{vertical}$	Függőleges tengelyen elhelyezett 1 egység hosszúságú vektor *X* és *Y* koordinátája
$\varphi_{actual}(k)$	Teljes pályagörbe *k*. útpontjának aktuális orientációja
$\varphi_{diff}(k)$	Teljes pályagörbe *k*. útpontjának orientáció különbsége a *k-1*. útponthoz képest
b	Hajtott kerekek középpontjai közötti távolság
$t(k), t(k-1)$	Teljes pályagörbe *k*. és *k-1*. útpontjának időértéke
$\omega(k)$	Pályagörbe *k*. útpontjának szögsebessége
$v_L(k), v_R(k)$	Teljes pályagörbe *k*. útpontjában bal és jobb oldali kerekek középpontjaiban ébredő sebesség
$v(k)$	Teljes pályagörbe *k*. útpontjában kerekek közötti felezőpontban ébredő sebesség
U_M	Feszültség a DC motor elektrodinamikai modelljéhez
R	DC motor belső ellenállása
i_M	Motoron átfolyó áramerősség a DC motor dinamikai modelljéhez

t	Általános időérték
$k \cdot \phi$	Sebességállandó inverze
ω_M	DC motor szögsebessége a DC motor elektrodinamikai modelljéhez
L	DC motor induktivitása
f_M	Motor belső súrlódási együtthatója
M_g	Gördülésből származó ellennyomaték
J	DC motor tengelyén ébredő tehetetlenségi nyomaték
U_L, U_R	Bal és jobb oldali kereket hajtó DC motor feszültsége
$v_{L_{input}}, v_{R_{input}}$	Bal és jobb oldali kerekek középpontjaiban ébredő bemeneti sebesség a **c**. modul és a **c**-**f**. modulláncolat teszteléséhez
k_H	Kerék és motor közötti hajtőmű hajtóviszonya
r_W, d_W	AGV kerekének sugara és átmérője
U_N	Névleges feszültség a **c**. modul teszteléséhez
v_{input}	Általános bemeneti sebesség a **c**. modul teszteléséhez
M_t	Terhelőnyomaték a DC motor elektrodinamikai modelljéhez
M_s	Motor belső súrlódásából adódó nyomaték
$f_{M,n}$	Motor belső súrlódási együtthatója névleges fordulatszámon
$M_{s,n}$	Motor belső súrlódásából adódó nyomaték névleges fordulatszámon
$\omega_{M,n}$	DC motor szögsebessége névleges fordulatszámon
$n_{M,n}$	DC motor névleges fordulatszáma
F_g	Gördülésből adódó ellenerő
$F_{\mu_{0,G_1}}, F_{\mu_{0,G_2}}$	Rendszeren ébredő gördülési ellenerő az *1*. és *2*. esetben
$F_{húzó_1}, F_{húzó_2}$	Húzóerő az *1*. és *2*. esetben
W	Kerék sorszáma
$F_{\mu_{0,G},W}$	A *W*. sorszámú kereken ébredő gördülési ellenerő
$F_{N,W}$	A *W*. sorszámú kereken ébredő normálerő
F_{N_1}, F_{N_2}	Normálerő az *1*. és *2*. esetben
m_{AGV}	Targonca tömege
g	Gravitációs gyorsulás
μ_{0,G_1}, μ_{0,G_2}	Gördülési ellenállási együttható az *1*. és *2*. esetben
β	Lejtő szöge a gördülési ellenállás mérés *2*. esetében

F_T	Mozgás irányába eső tangenciális erőkomponens
G_2	Gravitációs erő az *1.* és *2.* esetben
$d_{tengely}$	Gördülési ellenállás méréshez használt hajtómű tengelyének átmérője
$v_{targonca}$	Gördülési ellenállás méréshez a targonca sebessége
$i_{Faulhaber}$	Gördülési ellenállás méréshez használt hajtómű áttétele
$\omega_{tengely}$	Gördülési ellenállás méréshez használt hajtómű tengelyének szögsebessége
n_{PSB}	Gördülési ellenállás méréshez használt akkumulátoros csavarozógép fordulatszáma
M_d	Hajtónyomaték a DC motor dinamikai modelljéhez
$\omega_{L_{motor}}, \omega_{R_{motor}}$	Bal és jobb oldali DC motor szögsebessége
ω_L, ω_R	Bal és jobb oldali kerék szögsebessége
$v_{L,modul}, v_{R,modul}$	Bal és jobb oldali kerék sebessége a **d**. modulnál
v_C	AGV kerekek közötti C felezőpontjának sebessége
ω_c	C felezőponton átmenő függőleges tengely körüli szögsebesség
x_C, y_C	Kerekek közötti felezőpont *X* és *Y* koordinátája
φ_C	Kerekek közötti felezőpont elfordulási szöge
i_L, i_R	Bal és jobb oldali kereket hajtó DC motor áramerőssége
n_L, n_R	Bal és jobb oldali kereket hajtó DC motor fordulatszáma
U_L, U_R	Bal és jobb oldali kereket hajtó DC motor feszültsége
U_1	AGV 4db sorba kötött akkumulátor névleges, primer feszültsége
$U_{1_{max}}, U_{2_{max}}$	Maximális bemeneti és kimeneti feszültség a feszültségosztó esetén
R_1, R_2, R_3	Feszültségosztóban használatos névleges ellenállások
$U'_{2\max}$	Feszültségosztó valós szekunder feszültség
R'_1, R'_2, R'_3	Feszültségosztóban használatos valós ellenállások
$U_{Arduino}$	Arduino bemenetén mért feszültségérték
$value_{U,Arduino}$	Arduino programjában levő feszültségérték
$U_{program}$	PC programjában levő feszültségérték
$i_{left,Arduino}$, $i_{right,Arduino}$	Arduino bemenetére kapcsolt Hall-szenzoron mért áramerősség a bal és jobb oldali kerék esetén

$value_{i_{left},Arduino}$, $value_{i_{right},Arduino}$	Arduino programjában levő áramerősség-érték a bal és jobb oldali kerék esetén
$i_{left,program}$, $i_{right,program}$	PC programjában levő áramerősség-érték a bal és jobb oldali kerék esetén
T_{camera}	Kamerás felvétel két képkocka közötti periódusideje
$n_{W,mérés}$	Kamerás mérésből adódó kerék percenként megtett fordulatszáma
$t_{W,mérés}$	Kamerás mérésből adódó kerék egy fordulatához szükséges idő
$n_{W,mérés,1000}$	Kamerás mérésből adódó kerék percenként megtett fordulatszáma 1000 egységre vetítve
$n_{M,setup}$	Motorvezérlőnek vezérlőprogramban beállított értéke
n_M	Hajtómotor fordulatszáma
n_W	Hajtott kerék fordulatszáma
ω_W	Hajtott kerék szögsebessége
v_W	Hajtott kerék sebessége
$X_{actual,LIDAR}$, $Y_{actual,LIDAR}$	LIDAR szenzorral mért aktuális pozíció *X* és *Y* koordinátája a LIDAR szenzor középpontjában
$\varphi_{actual,LIDAR}$	LIDAR szenzorral mért aktuális orientáció a LIDAR szenzor középpontjában
$X_{actual,center}$, $Y_{actual,center}$	LIDAR szenzorral mért aktuális pozíció *X* és *Y* koordinátája a kerekek közötti felezőpontban
$\varphi_{actual,center}$	LIDAR szenzorral mért aktuális orientáció a kerekek közötti felezőpontban
X_{start}^{1}, Y_{start}^{1}	*1.* szegmens LIDAR szenzor által mért kezdeti pozíció *X* és *Y* koordinátája a kerekek közötti felezőpontban

Vektor- és mátrixmennyiségek

$\boldsymbol{P}_0, \boldsymbol{P}_3$	Pályagörbe kezdő- és végpontja
$\boldsymbol{P}_1, \boldsymbol{P}_2$	Pályagörbe 1. és 2. vezérlőpontja
$\boldsymbol{B}$	Kezdő-, vég-, és vezérlőpontokat magába foglaló mátrix
$\boldsymbol{P}_B(u), \boldsymbol{P}_H(u)$	Bezier- és Hermite-görbe függvénye
$\boldsymbol{F}(u)$	Hermite-görbénél együtthatókat magába foglaló vektor
$\boldsymbol{P}_{S1}, \boldsymbol{P}_{2E}$	Pályagörbe 1. és 2. vezérlővektora Hermite-görbéhez
$\boldsymbol{P}_{start}, \boldsymbol{P}_{end}$	Targonca kezdeti- és célpozíciója a kerekek középvonalánál
$\boldsymbol{P}_1^{"}, \boldsymbol{P}_2^{"}$	Bezier-görbe módosított 1. és 2. vezérlőpontja
$\boldsymbol{P}_{S1}^{"}, \boldsymbol{P}_{2E}^{"}$	Hermite-görbe módosított 1. és 2. vezérlővektora
$\boldsymbol{P}_{start,LIDAR}$	LIDAR szenzor által mért kezdeti pozíció
$\boldsymbol{P}_{end,LIDAR}$	LIDAR szenzor által mért és beállított célpozíció
$\boldsymbol{W}_0, \boldsymbol{W}_1, \boldsymbol{W}_{i-1}, \boldsymbol{W}_i, \boldsymbol{W}_N$	Útpont sorozat *0.*, *1.*, *i-1.*, *i.* és *N.* útpontja
$\boldsymbol{q}_{i-1}, \boldsymbol{q}_i$	Útpont sorozat *i-1.* és *i.* útpontjának pozícióját és orientációját magába foglaló oszlopvektor
$\boldsymbol{P}_0, \boldsymbol{P}_1, \boldsymbol{P}_{n-1}, \boldsymbol{P}_n$	*1.* szegmens pontkészletének *0.*, *1.*, *n-1.* és *n.* tagja
$\boldsymbol{Q}_0, \boldsymbol{Q}_1, \boldsymbol{Q}_n$	*2.* szegmens pontkészletének *0.*, *1.* és *n.* tagja
$\boldsymbol{P}_{start,LIDAR}^1$	*1.* szegmens LIDAR szenzor által mért kezdeti pozíció
$\boldsymbol{P}_{start,LIDAR}^S$	*S.* szegmens LIDAR szenzornál levő számított kezdeti pozíció
$\boldsymbol{P}_{end,LIDAR}^1$	*1.* szegmens LIDAR szenzor által mért és beállított célpozíció
$\boldsymbol{P}_{end,LIDAR}^2$	*2.* szegmens LIDAR szenzor által mért és beállított célpozíció
$\boldsymbol{P}_{end,LIDAR}^S$	*S.* szegmens LIDAR szenzor által mért és beállított célpozíció
$\boldsymbol{P}_{end}^1, \boldsymbol{P}_{end}^2, \boldsymbol{P}_{end}^{S-1}, \boldsymbol{P}_{end}^S$	*1.*, *2.*, *S-1.* és *S.* szegmens célpozíciója a kerekek közötti felezőpontban
$\boldsymbol{P}_{actual}(k)$	Teljes pályagörbe *k.* útpontjának aktuális pozíciója a *k-1.* útponthoz képest
$\boldsymbol{P}_{k-1}, \boldsymbol{P}_k$	Teljes pályagörbe *k-1.* és *k.* útpontjának aktuális pozíciója
$\boldsymbol{P}_{vertical}$	Függőleges tengelyen elhelyezett 1 egység hosszúságú vektor
$\dot{\boldsymbol{x}}$	v_c sebesség *X* és *Y* vetületi koordinátáit és a ω_c szögsebességet magába foglaló oszlopvektor
$\boldsymbol{P}_{start}^1$	*1.* szegmens LIDAR szenzor által mért kezdeti pozíciója a kerekek közötti felezőpontban

Egyéb jelölések

AGV	Automated Guided Vehicle = Automatikusan vezérelt jármű
LIDAR	Light Detection and Ranging = Lézerradar
CAD	Computer-Aided Design = Számítógéppel segített tervezés
PLC	Programmable Logic Controller = Programozható logikai vezérlő
PC	Personal Computer = Személyi számítógép
DoF	Degree-of-Freedom = Szabadságfok
DC	Direct Current = Egyenáram
USB	Universal Serial Bus = Univerzális soros busz
FPS	Frame Per Second = Képkocka per másodperc
RPM	Revolutions Per Minute = Percenkénti fordulatszám

1. BEVEZETÉS

1.1. Kutatási téma jelentősége és célkitűzései

Manapság az Ipar 4.0 meghirdetése révén az automatizálás és mobil eszközök terjedése felgyorsult [1], és ez a gyártástámogatás terén is megjelent [2]. Magyarországon és az Észak-magyarországi régióban is egyre nagyobb számú igény jelentkezik a robotizálásra, automatizálásra. Egy logisztikai folyamat tervezése során mindig a modernizálásra kell törekedni [3]-[6]. A robotika egyre inkább teret hódít az automatizálási eszközök terén [7], mind az ipari robotok [8], mind a mobil robotok terén [9]. Ugyanakkor az ipari robotok terén is szükség van a folyamatos fejlesztésekre, pl. erő-visszacsatolás alapú beszerelési folyamat tervezése [10].

Szintén az Ipar 4.0 hatására az iparban is megjelentek a vezető nélküli járművek, így a szállítótargoncák is, amelyek mobil robotnak [11] vagy AGV-nek (Automated Guided Vehicle) tekinthetők [12]-[13]. Egy szállítójármű automatizálása még manapság is összetett, magas fokú szakértelmet igénylő feladat. Egy ilyen vezető nélküli szállítótargonca vagy AGV a Miskolci Egyetem Logisztikai Intézetének High-Tech laboratóriumában található [16], [17], a targonca az ottani anyagmozgatási rendszer része [14]. Ez a targonca egy egyedi gyártású prototípus, amelyet a Gamma Digital Kft. készített 2009-ben [14]. Ezen AGV a laboratórium költözése után már nem tudott automatikusan működni, mivel a korábbi helyéhez kötötten alakították ki a mozgásvezérlést [15], amely módszer az AGV áttelepítésével már nem megfelelő.

Általában egy AGV pozíciójának meghatározására különböző technológiák állnak rendelkezésre [18], mint például induktív szenzor [19]-[21], képfeldolgozás [22]-[23] vagy LIDAR szenzor. Meglátásom szerint túlnyomórészt a legutóbbi módszer célravezető annak konfigurálhatósága miatt. A disszertációban vizsgált jármű pozícióját szintén a legutóbbi módszer segítségével érzékeli és virtuális pályát használ, mint a [24]-ben kifejlesztett AGV, viszont a LIDAR érzékelők csak egy transzformációs algoritmussal érzékelik a környezetet.

A disszertáció témája egy olyan mozgásvezérlés és szimuláció kialakítása, amely alkalmas az adott AGV mozgását előzetesen megtervezni, majd a szervomotorok vezérlésével a megtervezett pályát a kívánt sebességgel és szögsebességgel bejárni a kisebb áramfelvételre törekedve, ezen felül szimulációval kimutatni az egyes paramétereket.

A két pont közötti mozgásszabályozáshoz és szimulációhoz a következő modulokra van szükség: **a.** pályatervező modul, **b.** trajektóriatervező modul, **c.** sebesség-feszültség átalakító modul a trajektóriatervezőtől kapott sebességek felhasználásával, **d.** mozgásvezérlés és a motor elektrodinamikai modelljének szimulálása az átalakítóból érkező feszültségek felhasználásával, **e.** az út szimulációja és **f.** kommunikációs modul.

A disszertáció egy olyan új pálya-és trajektória tervező megoldás kialakítását mutatja be, amely nem helyiséghez kötött, hanem áttelepítés után újra használható. Robotvezérléssel például a [25] és [26] irodalmakban foglalkoztak, ahol különböző felépítésű mobil robotokra különböző módszerekkel valósították meg a robot motorjainak vezérlését. Az egyik esetben a kevésbé megbízható képfeldolgozáson alapul a vezérlés, míg a másik esetben csak elméletileg tárgyalták a vezérlést.

A disszertáció első célja, hogy az új pályatervező megoldás egyszerre két módszerrel is meghatározza a pálya pontokat a targonca geometriai felépítését figyelembevéve. A disszertáció második célja egy olyan új trajektóriatervező megoldás kifejlesztése, amely a pályapontok geometriai és idő adatai alapján kiszámolja a targonca hajtott kerekeinek sebességét. Ezen felül az új pálya-és trajektória tervező megoldás további célja, hogy a hajtás áramfelvétele szempontjából kedvezőbb megoldást válassza.

Az áramfelvétel mértékének meghatározása a hajtómotorok áramfelvételén alapul, amelyet modellekkel szimulálni, valamint mérőeszközökkel mérni is lehetséges. Az értekezés harmadik célja egy új moduláris rendszer kialakítása a hajtás áramfelvételének meghatározására és a pálya szimulációjára.

A disszertáció utolsó célja egy olyan új AGV állapotfelügyeleti mérőrendszer kifejlesztése, amely méri a rendszer tápfeszültségét, hajtás áramerősségét és fogadja a targonca navigációs adatait.

1.2. PhD értekezés felépítése

A disszertáció a célkitűzésekben megfogalmazott feladatok kidolgozását a következő szerkezetben foglalja össze.

A disszertáció 2. fejezete a PhD kutatás témájához kapcsolódó szakirodalmat foglalja össze.

A 3. fejezet a vezető nélküli targoncát és annak mozgásának vezérléséhez és szimulációjához tervezett hat darab modult ismerteti.

A vezérlés hat modulját a 4. fejezet dolgozza fel Scilab szoftveren alapulva.

A vezető nélküli targonca pályájának tervezését megvalósított program felépítését és működését a 4.1. fejezet írja le. A tervezett pálya alapján a targonca vezérlése elkészíti a sebesség- és szögsebesség-adatokat tartalmazó trajektóriát a 4.2. fejezetben leírtaknak megfelelően. A 4.3. fejezetben részletezett modul a trajektóriatervező modulból kijövő sebesség adatokat konvertálja a DC motor elektrodinamikai modelljéhez szükséges feszültségbemenetté. A 4.4. fejezet a targoncában alkalmazott DC motor elektrodinamikai modelljét tárgyalja, kitérve a terhelőnyomaték targoncában használatos elemeire. A 4.5. fejezetben ismertetett programrészlet a motor modelljéből adódó szögsebesség adatok alapján elvégzi a pálya szimulációját. A 4.6. fejezet a kommunikációs modult mutatja be. A 4.7. fejezetben bemutatott grafikus program a 4.3-4.6. fejezetben bemutatott szimulációs részeket egy modulláncolatba helyezi. A fejezet bemutatja a modulláncolat működését különböző példákon keresztül.

Az 5. fejezet az AGV saját vezérlőrendszerében elvégzett manuális és automatikus mozgatásához szükséges programfejlesztéseket részletezi, amelyek szükségesek a teljesen autonóm mozgásvezérlés sikeres előkészítése és elvégzése érdekében.

A mozgásvezérlés és annak ellenőrzése érdekében szükséges mérési rendszer kialakítását tárgyalja a 6. fejezet, kitérve a navigációs adatok alapján a pozíció- és orientáció mérésre, valamint az ellenőrzés érdekében a meghajtó akkumulátorok feszültségének és hajtó szervomotorok áramerősségének mérésére.

A 7. fejezet feldolgozza a pozíció, orientáció, feszültség és áramerősség mérések eredményeit és levonja a következtetéseket.

A 8. fejezet a disszertáció eredményeit tézisekben összegzi.

A 9. fejezet összefoglaló megállapításokat tesz magyar és angol nyelven.

A további, nem számozott fejezetekben köszönetet mondok a kutatást segítőknek, a felhasznált és saját irodalmakat sorolom fel, valamint ábra- és tartalomjegyzéket helyeztem el.

2. SZAKIRODALMI ÁTTEKINTÉS

A robotok pálya- és trajektória tervezés téma széleskörű irodalmi háttérrel rendelkezik.
Elsőként lehetne említeni [27] irodalmat, amelyben a téma szempontjából a „Path Planning and Trajectory Planning Algorithms: a General Overview" fejezet releváns. Az említett fejezet foglalkozik egy pálya- és trajektóriatervezés algoritmusaival. A Miskolci Egyetemen is foglalkoztak a témával, amelyből egy könyvfejezet született [28]. Algoritmus tervezés szempontjából nem csak a fenti kategóriába osztás, hanem offline-online tervezés is lehetséges [29].
A [27] irodalom a pályatervezés és trajektória tervezés terén két külön csoportja osztja az egyes algoritmusokat. A pályatervezési módszereket a 2.1. alfejezetben, míg a trajektóriatervezési módszereket a 2.2. alfejezetben foglaltam össze.
A disszertációban vizsgált AGV korábbi trajektóriatervezési megoldását a 2.3. alfejezetben részletezem.
A 2.4. alfejezet a pályatervezéshez szükséges interpolációs- és approximációs megoldásait foglalja össze.

2.1. Pályatervezés

A pályatervezés módszerei 3 technikára osztható fel [27]: útiterv technikák (angol „roadmap" kifejezésből); cellára bontási technika (angol „cell decomposition" kifejezésből) és mesterséges potenciál technikája (angol „artifical potential" kifejezésből).

Útiterv technikák

Ez a technika egy n-dimenziós konfigurációs teret redukál le egy egydimenziós útvonallá a keresés céljából, lehetőleg egy gráfban. Ebben a térben az irodalom felosztja egy „C-free" szabad és egy „C-obs" nem szabad vagy ütközési terekre, a kettő tér együtt „C-space" konfigurációs teret alkot.
Egy ilyen technika a [30] irodalomban található, ahol egy ún. láthatósági gráfban mutatja meg, ahol az egyes ütközési terek végpontjait csomópontoknak tekinti, és ezeket összeköti egy-egy vonallal. Ezeken a vonalakon már megvalósítható például egy legrövidebb euklideszi útvonal keresés a konfigurációs térben.
Egy másik megközelítés a Voronoi-diagramok alkalmazása. Itt az akadályokat négyzet formájában jelenítik meg, és a tervezett útvonalak úgy vannak kialakítva, hogy egyforma távolságban legyenek legalább két akadálytól. Példát erre a [31] mutat.

Cellára bontási technika

Ez a második ismert módszer a „C-free” szabad teret osztja fel több régióra, nevezetesen cellákra, és az útvonal bármely két kapcsolódása két szomszédos cella között helyezkedik el. Az ún. kapcsolódási gráf a szomszédos cellák közötti kapcsolódásokat alkotja. Ezután az útvonalkeresés problémája egy gráfkeresési problémává alakul át.

A cellákra való felbontást további két csoportra lehet osztani: pontos cellára felbontás és közelítő cellára felbontás.

Az első esetben a szabad teret poligonokra osztja fel [27], [32]. A poligonok összessége pontosan kitölti a szabad teret. A második esetben háromszögek [33], négyzetek vagy téglalapok [27] segítségével osztja fel a teret. Mivel ezen síkidomok összessége nem tölti ki tökéletesen a teret, ezért csak közelítő a felbontás pontossága. A felbontás lépései [27]:

1. az egész „C-space” teret felosztja 4 egyenlő részre,
2. az algoritmus megvizsgálja, hogy teljesen üres, teljesen tele, vagy vegyesen vannak az egyes cellák,
3. minden vegyes cellát további 4 részre bontja fel, és az algoritmus ezt ismétlően alkalmazza egy bizonyos limitig.

Amennyiben síkról, vagyis 2D-ről van szó, akkor a szabad teret 4 részre tudja felosztani az algoritmus, amennyiben térről, vagyis 3D-ről van szó, akkor 8 részre lehet felosztani a teret.

Mesterséges potenciál technikája

Ez a technika egy másféle megközelítést alkalmaz az útvonal keresés problémájára. Az alapötlet feltételezi, hogy a robot a konfigurációs térben mozgó objektum egy potenciális területen, amelyet a célkonfiguráció és a teljes „C-space” térben levő akadályok generálták, nevezetesen, a célkonfiguráció létrehoz egy attraktív (vonzó, kedvező, csábító) potenciált, míg az akadályok egy repulzív (visszataszító, ellenszenves) potenciált. A kettő összessége a teljes potenciál, amely látható a robot számára egy mesterséges erő által, amely a cél felé vezet az akadályokat elkerülve. Ezt a technikát mobil robotra is lehet alkalmazni [34], azonban meglátásom szerint ezzel a módszerrel nagyon élesen fordul a robot.

Ennek a technikának van egy fő problémája: a lokális minimumok előfordulása, ahol a robot csapdába kerülve találja magát. Erre több megoldás is született, például olyan potenciális függvények használata, amelynek nincs lokális minimuma [35]. Ezek a funkciók a navigációs funkciók.

Egyéb módszer erre a technikára az RPP (Random Path Planners) tervező használata [27], [36]. Ez a lokális minimumokat úgy kerüli el, hogy a mesterséges potenciális területek koncepcióját véletlenszerű keresési technikákkal kombinálja.

További módszer, amely jelentős eredményeket tud elérni nagyon komplex útvonalkeresési problémáknál, a PRM (Probabilistic Roadmap Planners) tervező [27], [37]. Ez a technika alkalmaz valószínűségszámításon alapuló algoritmusokat, mint a véletlenszerű mintavételezés. Az alapötlet feltételezi egy gráfot, ahol a csomópontok a „C-free" szabad térben véletlenszerű konfigurációk halmaza által adottak. Egy helyi tervező ezután megpróbálja ezeket a konfigurációkat összekapcsolni, ha egy kapcsolódást talál, akkor új csomópontot ad a gráfhoz. Ezáltal a gráf mutatja a „C-free" szabad tér csatlakoztathatóságát. A két konfiguráció közötti elérési út kereséséhez ezeket a konfigurációkat hozzáadják a gráfhoz, majd útvonalkeresési problémát lehet megoldani. Tekintettel a valószínűségen alapuló megoldásra, utómunkálatokra lehet szükség az útvonal minőségének javításához, és véleményem szerint ez növeli a hibák lehetőségét.

2.2. Trajektória tervezés

A trajektória tervezés problémája generálja a referencia bemeneteket a robot vezérlője számára, biztosítva azt, hogy a kívánt mozgás megvalósuljon. Általában a bemenetek a pályatervező által tervezett pálya, és a robot dinamikai és kinematikai kényszerei.

A legfontosabb optimalizálási kritériumok a trajektória tervezéshez [27], [38]: legkisebb végrehajtási idő, legkisebb energia vagy aktuátor erőfeszítés, és legkisebb rántás (gyorsulás idő szerinti deriváltja, idegen nyelven „jerk").

Figyelembe lehet venni a fenti paraméterek közül egyet, vagy hibrid optimalizálási kritériumok esetén többet, mint például idő-energiára optimális trajektória tervezés.

Legkisebb végrehajtási idő

Amennyiben optimálásra van szükség, az első lehetőség, ami adódik, a végrehajtási-, vagy ciklusidő csökkentése. A mai automatizált termelési rendszerekben kulcsfontosságú a végrehajtási idő a magas termelékenység érdekében. Emiatt számos irodalom foglalkozik a végrehajtási időn alapuló trajektória tervezéssel.

Az egyik lehetőség egy algoritmus definiálása pozíció-sebesség fázissíkban [39], [40]. Itt az alapötlet az, hogy a manipulátor dinamikai egyenletét paraméteres formában írják fel a görbe koordinátáiból, amelynél a pálya független paraméter. Itt a rendszer része még a pálya első és második deriváltja, a pszeudo-sebesség és -gyorsulás. Az aktuátorok és nemlineáris

robotdinamika korlátozásai egy vezérlő változóvá alakítható. A pálya minden egyes pontján az end-effektor maximum pszeudo-sebességét a korlátok határozzák meg, ezáltal lehetőség van egy pozíció-sebesség fázissík rajzolására, amely egy sebességkorlát görbe lesz (VLC: Velocity Limit Curve). Az optimális trajektóriát ezután a vezérlő kiszámolja minden egyes pontjára, amelyeknél így a maximális sebesség nem lépheti túl a fázisgörbét.

Egy másik megközelítésben dinamikus programozási technikák alkalmazásából áll [41]. Az alapötlet szerint az állapotteret diszkretizálja pontokból álló rácsra. Ezután a sebesség-, gyorsulás, rántáskorlátokat hozzárendeli az egyes pontokhoz, és definiálja minden egyes megoldás költségét figyelembe véve a mozgáshoz szükséges időt. A költség definiálása egy állandó gyorsulás feltételezésével minden egyes lépésre történik. Végül a dinamikus programozásra alapozott algoritmus generálja a legkisebb idejű trajektóriát.

A fenti módszerek hátránya, hogy az algoritmusok nem folytonos gyorsulásokat és csuklónyomatékokat eredményez, ugyanis a dinamikai modell tökéletesen merev testet vesz figyelembe, ugyanakkor az aktuátor dinamikát nem veszi figyelembe. A valóságban csak folytonos gyorsulások és nyomatékok adódhatnak, így ez késleltetésben jelentkezik és csökkenti a trajektória követés pontosságát. Emiatt a pályaszabályzónak gyakran be kell avatkoznia a trajektória végrehajtásába.

Erre több megoldás is született. Az egyik, hogy a fázissík módszert a nyomatékokra való korlátokkal együtt alkalmazza [42]. A gyakorlati eredmények megmutatják, hogy néhány, pszeudo-rántásra vonatkozó felső korláttal idő-optimális trajektóriát egy hagyományos PID vezérlővel is meg lehet oldani. Egy másik megoldás az, hogy a célfüggvényben nem csak a végrehajtási időt, hanem az energia-hozzájárulást is felhasználja az algoritmus, mint például a teljes pálya mentén négyzetes nyomatékok integrálása [43]. Itt bemutatták, hogy a követendő trajektória pontossága kompenzálhatja a teljes mozgási idő növekedését. Ezek az eredmények csökkentik az aktuátort érő terheléseket.

Lehetőség van spline-interpolációk alkalmazására is. Ekkor a pályát felosztják véges számú szakaszra, és a szakaszok kapcsolódását kerekítik/simítják le. A szakirodalomban számos technikát alkalmaznak, ezek között a különbségek a következők:

- a figyelembe vett kényszerek (vagy kinematikai [44] vagy dinamikai [45]),
- a felhasznált algoritmus az optimális trajektória számításához [46],
- a lehetőség az optimálási probléma kiterjesztésére, figyelembe véve az optimálási kritériumokat [47].

Legkisebb energia vagy aktuátor erőfeszítés

Bizonyos esetekben a legkisebb ciklusidőre való törekvés helyett inkább a legkisebb energiafelhasználásra, -igényre vagy aktuátorra ható erőfeszítés kerül előtérbe. Az energián alapuló trajektória tervezés több esetben is érdekes lehet. Az egyik eset, hogy sima pályákat generál, ezáltal csökkentve az aktuátorokra és mechanikai szerkezetre jutó erőfeszítést. A másik eset, amikor nemcsak gazdasági vagy környezetvédelmi okokból van szükség az energiafelhasználás csökkentésére, hanem sok olyan alkalmazás is lehet, amikor technikai okokból az energiafelhasználás korlátozott, mint például víz alatt kutatások [48] vagy katonai célok.

Egy példát lehet említeni a mobil robotok esetét, amikor az energiaellátás túlnyomórészt akkumulátorok szolgáltatják [49]. Ebben az esetben a működési idő meghosszabbítása esetén törekedni kell a minél kisebb energiafelhasználásra.

Számos irodalom foglalkozik ezzel az optimálási kritériummal, így példának lehet említeni az [50] és [51] irodalmakat, ahol a trajektóriát köbös B-spline-ok segítségével paraméterezik, ahol csuklók fizikai korlátjai lettek hozzáadva a nyomatéki és kinematikai korlátokhoz. Valamint az itt bemutatott célfüggvény elsősorban az energia minimalizálásra törekszik tekintet nélkül a végrehajtási időre.

Egy háromkerekű minden irányú mobil robotra (TOMR = Three-Wheeled Omnidirectional Mobile Robot) alkalmazták a Pontryagin minimum elvet a a trajektória tervező algoritmushoz, az akkumulátorból energiafelvételt a költségfüggvény számításához és egy dinamikai modellt, beleértve az aktuátor dinamikáját és a Coriolis erőt [49], [52]-[53].

A trajektória tervezés egy egyhajtású robot esetén is fennmerül, amelynél egy villanymotor hajtja a járművet egy mechanizmuson át [54]. A Fuzzy és Newton-barrier módszert alkalmazták egy nagyfeszültségű távvezeték vizsgálatára kifejlesztett kétkarú mobil robot trajektória tervezésénél [55].

Egy két kerékkel rendelkező mobil roboton (WMR: Wheeled Mobile Robot) indirekt megoldást adtak az optimal trajektória tervezéshez készített optimális vezérlési stratégiához, amelyhez figyelembe vették a rendszer nemlineáris dinamikai modelljét és anholonom kényszereit [56].

Az [57] irodalomban egy svéd típusú, hajtott és kormányzott kerekekkel felszerelt mobil robotra valósították meg költség indexre optimált mozgásvezérlést, figyelembe véve a direkt és indirekt modellek szingularitását, ennek elkerüléséhez figyelmen kívül hagyva a csúszó mozgásokat, végtelen becslési hibát és lehetetlen vezérlési beavatkozásokat.

Egy komplex és reális környezetben működő humanoid robothoz készített gyors költséghatékony mozgástervezést mutat be az [58] irodalom, amelyben az RRT (Rapidly exploring Random Tree) mintavétel alapú algoritmus különböző variánsait használja fel.
Az [59] irodalomban bemutatott hexapod robot egy kombinált, energiafogyasztásra és munkatérre alapuló, indexet használ a nemlineáris programozás matematikai modelljéhez egyenlőtlen kényszereket alkalmazva.
Az optimális útvonaltervezéshez a Keymeulen/Decuyper folyadék módszer alkalmazását írja le a [60] irodalom egy anholonom járműhöz.
Egy POI (Points of Interest) pontok vizsgálatához készített mobil robot az energiafogyasztás figyelése mellett a kommunikáció biztonságát és költségét is figyelembe veszi a működéshez. Ezt a problémát egy vegyes egész típusú lineáris programként (MILP: Mixed Integer Linear Program) lehet felvetni [61].
Egy két lábon járó robotra a megfelelő járás kiválasztása csökkentheti az energiafogyasztást egy járásszintézis módszeren alapuló genetikus algoritmus felhasználásával [62].
A trajektória tervezésből adódó sebességprofil és útvonal keresése energiára optimalizáltan történik egy autószerű robotnál, amelyben DC motorokat alkalmaznak [63].
A [64] cikk egy differenciális hajtással rendelkező mobil robothoz készített optimális mozgástervezést mutat be, amelyben a modell felhasználható a kinetikus energiaátalakítás energiafogyasztásának megfogalmazására és a csúszási ellenállások leküzdésére.
Egy gömbgördülő (angolul spherical rolling) robot prototípus kifejlesztését és analitikai tanulmányait írja le a [65] irodalom. Itt különböző módszereket dolgoztak ki a robot minimális idejű és minimális energia alapú trajektóriájára is.

Legkisebb rántás

Ez az optimálási kritérium elsősorban az ipari robotokra jellemző, a mobil robotok esetén inkább az előbbi kettőt szokták alkalmazni. Ez a technika az aktuátor nyomatékokban jelentkező diszkontinuitások elkerülésére szolgál.
A [66] irodalomban ismertetett algoritmus intervallum-keresésen alapul. Az itt bemutatott technika a rántás maximum abszolút értékének a minimumát keresi egy pálya mentén, ahol a végrehajtási idő egy prioritást ad. A trajektóriák primitívjai köbös spline-ok, és az intervallumokat a köztes pontok között, és a legalacsonyabb maximum rántási értéket is kiszámítják. Ez az irodalom bemutat egy összehasonlítást a trigonometrikus spline-okon alapuló módszerrel, kitérve a rántás, nyomatékok és nyomatékvariációk legmagasabb értékeire.

2.3. A vizsgált AGV-n korábban elkészített trajektória tervezés

A Logisztikai Intézet laboratóriumban található AGV-n már kialakítottak egy Fuzzy Logic alapú vezérlést [15]. A vezérléshez szükséges részeket ismerteti ez az alfejezet.

Az AGV LIDAR érzékelőjével lehetőség van a targonca pozíciójának és orientációjának meghatározására, valamint opcióként a helyiség kontúrjának kialakítására is. A helyiség kontúrját 2D CAD modellbe le lehet menteni, és megfelelő szoftverrel további módosításokat végrehajtani. Így a modellbe belekerültek a targonca engedélyezett pályája, érkezési irányok és állomások is. Az útvonal egyenes és íves szakaszokból állnak, az íves szakaszokat egyenes szakaszokra történő bontással helyettesítették.

Az automatizált mozgatás vezérléséhez a szerzők a következő lépéseket tették meg:

1. Útvonal tervezés
    a. A helyiség kontúrjának beolvasása a LIDAR szenzorral
    b. Az engedélyezett pálya megtervezése és exportálása AutoCAD szoftverben
2. Az exportált AutoCAD modell feldolgozása, és szakaszok, pontok meghatározása
3. Az útvonal számítása a célig
    a. A robothoz legközelebbi szakasz megtalálása vektorok segítségével
    b. A legközelebbi szakasz felosztása két részre
    c. A legrövidebb útvonal meghatározása A* gráf keresési algoritmus segítségével
4. A robot elmozdulásának számítása, mivel a LIDAR szenzor alacsony mérési frekvenciája nem elegendő információ forrás a vezérlő számára, ezért a robot kinematikája alapján is történik a pozíció meghatározása
5. A Fuzzy vezérlő bemeneteinek (cél távolság és cél szög) számítása
    a. Tagsági funkciók meghatározása a Fuzzy vezérlő számára a két bemenetre és két kimenetre (átlagos szögsebesség és szögsebesség-különbség) történő szabály felhasználásával
    b. Fuzzy szabály készlet létrehozása 6db szabállyal, és ezzel a robot irányának és sebességének szabályzása

Meglátásom szerint a módszer nagy hátránya, hogy a helyiség AutoCAD modellje miatt a vezérlés az adott helyiséghez kötött. Az áttelepítés az AutoCAD modell előkészítése és módosítása időigényes és a szoftverigény miatt költséges feladat, így véleményem szerint ez már nem megfelelő a targonca vezérléséhez. Ennélfogva egy új vezérlés kialakítása szükséges a disszertációban leírtaknak megfelelően.

2.4. Interpolációs és approximációs megoldások

A [67]-[80] az interpolációs és approximációs megoldásokra az 1. táblázat szerinti felsorolás adható, amelyben bal oldalt az egyes megoldások, a felső sorban az egyes irodalmak, az X jelölések pedig az egyes megoldások egyes irodalmakban való előfordulását mutatja. Az egyes megoldásokat jellemzően 3D CAD modellezésben alkalmazott élsimításoknál alkalmazzák, de előfordul mobil robot vezérlésénél is [81].

1. táblázat: Interpolációs megoldások irodalmakban való előfordulás kapcsolata

	[67]	[68]	[69]	[70]	[71]	[72]	[73]	[74]	[75]	[76]	[77]	[78]	[79]	[80]
Lagrange-féle polinom-interpoláció	X		X	X	X	X		X	X	X	X	X	X	X
Newton Divided-Difference képlete	X		X	X	X	X	X	X	X		X			
Hermit polinom interpoláció	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bezier görbék - Berstein polinom	X	X	X	X		X	X		X	X		X	X	X
Természetes köbös spline interpoláció	X	X	X	X	X	X		X	X		X	X		X
Catmull-Rom köbös spline interpoláció				X					X			X	X	X
Köbös B-spline	X	X	X	X	X	X			X	X		X	X	X

Ezek közül négy pontot feltételezve, azaz köbös közelítést alkalmazva azok a megoldások jók a pályatervezés szempontjából, ahol a megoldással származtatott görbe pontjai közül a kezdő- és végpontot interpolálja, azaz megegyezik az eredeti kezdő- és végponttal, míg a közbenső két pontot csak közelíti, tehát nem egyezik meg a korábbi két közbenső ponttal.

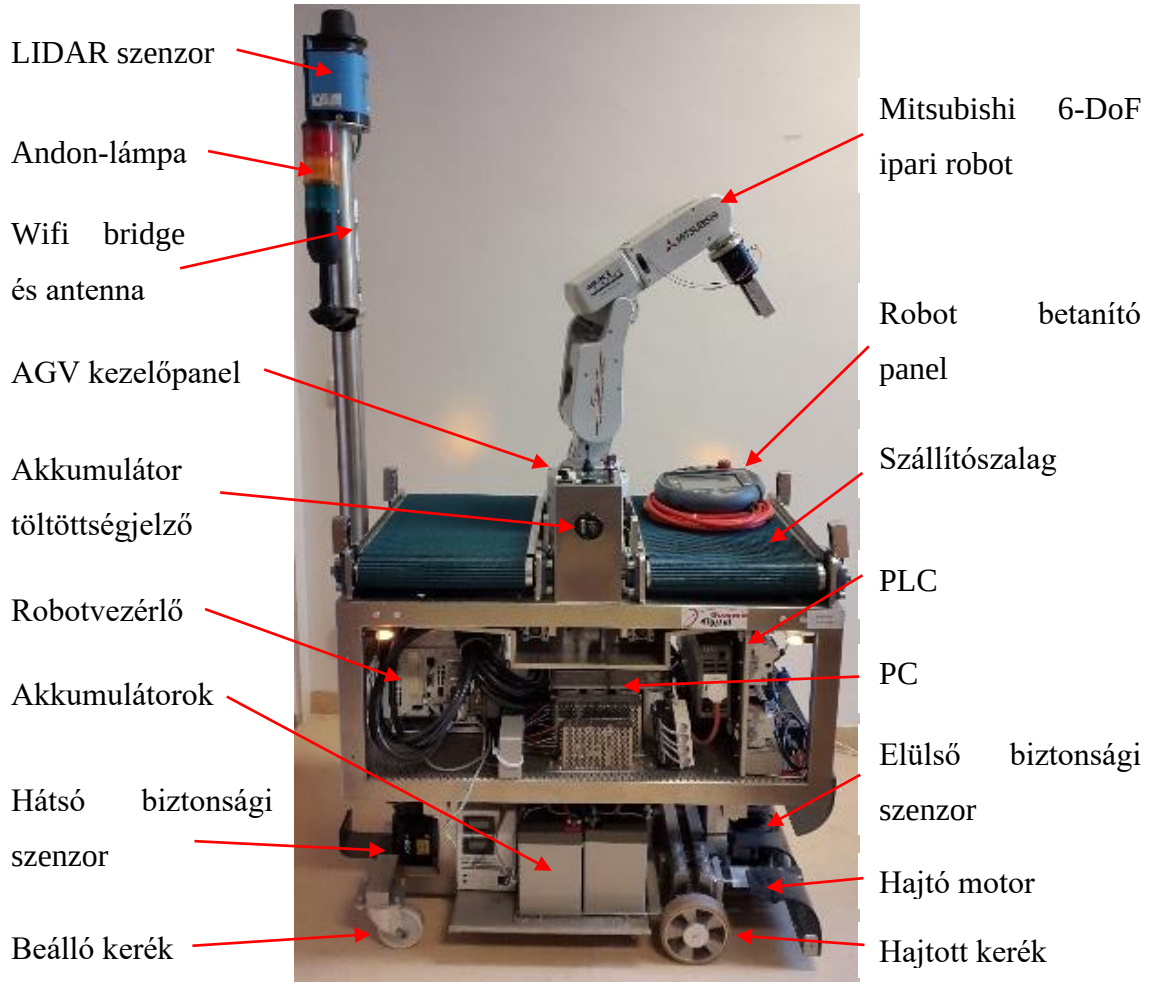
A [67]-[80] szakirodalmakat áttekintve megállapítható, hogy az 1. táblázatban ismertetett megoldások közül csak a Bezier-görbére lesz a fentebbi feltétel igaz. Ugyanakkor az Hermite interpolációs görbénél a kezdőpontból induló és végpontba érkező vezérlővektorok által ez a megoldás is megfelelő lesz a pályavezérléshez, mivel a vezérlővektorok az egyes pontokban levő elfordulási szögekből kiszámíthatók. A többi megoldásnál vagy az összes ponton áthalad az interpolált görbe, ami felesleges plusz utat generál, vagy mindegyik pontot csak közelíti, mint például a B-Spline, ami a pályatervezésnél nem megengedhető.

3. VEZETŐ NÉLKÜLI TARGONCA ÉS VEZÉRLÉSÉNEK FELÉPÍTÉSE

Jelen fejezet a kutatás során használt vezető nélküli targoncát részletezi, valamint kitér annak vezérlésének felépítésére.

Kutatás során vizsgált vezető nélküli targonca

A disszertációban vizsgált targonca a 3.1. ábrán látható. A targoncatípusok közül ezt a targoncát szállítótargoncának szokás nevezni, hiszen található rajta két darab szállítószalag, amelyek tárolófelületként is funkcionálnak.



3.1. ábra: Vizsgált vezető nélküli szállítótargonca

Egy szállítótargonca egy anyagmozgatási rendszerben különböző állomások között képes mozogni [16]. A targonca jellemzően egy bejövő áruhelynél felveszi az árut, elviszi például egy görgőspályarendszer valamelyik állomásához, és ott leadja az árut. Ugyanezt a mozgást ellentétes irányban is meg tudja tenni. Általában nem csak két állomás található egy ilyen rendszerben, így a targonca több különböző útvonalon is közlekedhet.

A targoncán a következő fontosabb elemek találhatók, ahogy a 3.1. ábrán látható:

- LIDAR Navigáció,
- 6 szabadságfokú ipari robot, betanító panel és robotvezérlő,
- szállítószalagok,
- biztonsági érzékelők elől-hátul,
- PLC és PC,
- 30V DC hajtó szervomotorok, 1:25 hajtóviszonyú hajtóművel hajtott kerekek,
- beálló kerekek.

Az ipari robot egy Mitsubishi márkájú RV-2SDB típusú robot, amely elektromos megfogója segítségével képes darabáruk vagy egységrakomány manipulálására.

A szállítószalag egységrakomány átadásával és elvételével képes a laboratóriumban található görgőspálya rendszerrel fizikailag kapcsolatba lépni.

A targonca elején és hátulján található Sick márkájú biztonsági érzékelők hivatottak a környezet érzékelésére. Az érzékelőknek két zónája van: figyelmeztető zóna, ekkor még nem áll meg a targonca, ezt az adatot fel lehet használni programozás során; valamint a tiltási zóna, ahol az érzékelő egy relé segítségével bontja a motorokra jutó áramellátást, ezáltal rögtön megáll a targonca.

A PLC a targoncán található egyszerűbb kimeneti és bemeneti adatok kezeléséért, a PC pedig a mozgásért felelős algoritmus futtatásért és az ehhez szükséges matematikai számításokért felelős. A PC feladata ezeken felül a hajtómotorok és szállítószalag motorok vezérlése párhuzamos porton keresztül.

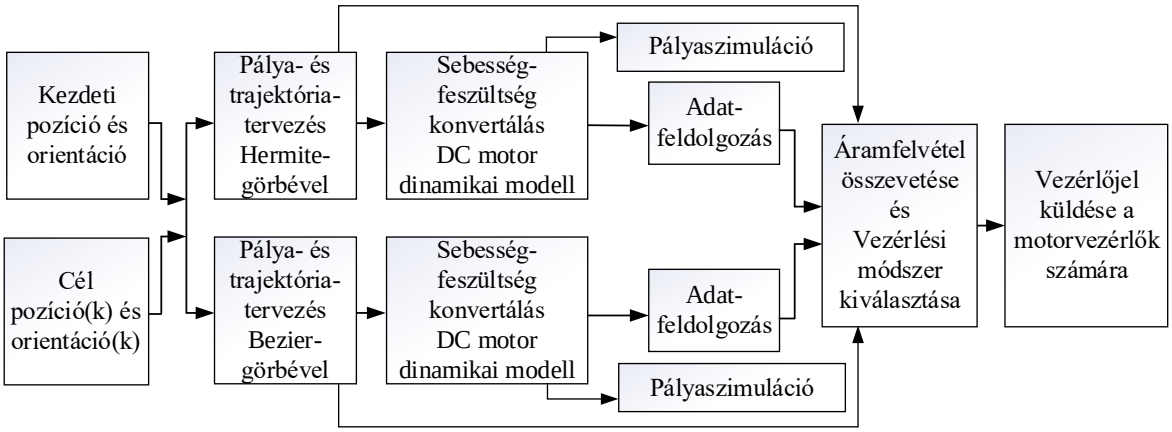
A hajtómotorok egyenáramú szervomotorok, amelyek egyenként 1:25 hajtóviszonyú hajtóművön keresztül hajtják a kerekeket. A két kerék hajtása független egymástól, így differenciális hajtásról beszélhetünk. A targonca további két kereke beálló kerék.

Az egyes, hálózatra is kapcsolható eszközök egy switch segítségével egymással, egy wifi bridge segítségével pedig a külvilággal is képesek kommunikálni.

A régebbi targoncákra jellemző a pályához kötöttség, amelyek valamilyen fizikai pályát követnek, mint például mágnescsíkot, vagy mágnespontokot, illetve optikai nyomvonalkövetés a jellemző még. A vizsgált targonca viszont már a modernebb módot használja, azaz egy virtuális pályán halad, a pozicionáláshoz csak a LIDAR navigációt és ahhoz szükséges reflexiós prizmákat használja.

Targonca mozgásának vezérlésének működése

A targonca általam megvalósított vezérlésének felépítését és főprogramjának működését a 3.2. ábra részletezi. Első lépésben a rendszer fogadja a navigációs adatokat, így a pozíciót és orientációt. Ehhez kapcsolódik hozzá az előre bevitt célpozíciók és orientációk, amelyeken áthalad a targonca, vagy célba érkezik. Ezen adatok alapján a főprogramban a kétféle pálya- és trajektóriatervező módszerrel előállnak a kerekek sebességei. A kerekek sebességadatait továbbítja a konvertáló modulba és a DC motor elektrodinamikai modell moduljába. Ez utóbbi modul egyik ágon továbbítja a szögsebesség értékeket, míg a másik ágra az áramerősség-értékeit továbbítja az adattovábbítás és áramfelvétel meghatározása érdekében. Az áramfelvételek összevetésével a főprogram kiválasztja a vezérlési módszert és előállítja a vezérlőjelet a motorvezérlők számára.



3.2. ábra: Targoncához kifejlesztett tervezési- és vezérlési rendszer felépítése

A targoncát bemutató ismeretanyag az [S7] publikációban, a vezérlés elemeit bemutató rész az [S3] és [S5] publikációkban került megírásra.

A következő fejezet az egyes modulokat ismerteti részletesebben.

4. AGV VEZÉRLÉSHEZ ALKALMAZOTT MODULRENDSZER

Ez a fejezet részletezi az AGV vezérléshez kifejlesztett modulrendszer hat darab modulját sorrendben: **a.** pályatervező modul előállítja a mozgáshoz szükséges pályapontokat (4.1. fejezet); **b.** trajektóriatervező modul a pályapontok geometriai és idő adataiból generálja a hajtott kerekek sebességét (4.2. fejezet), **c.** sebesség-feszültség átalakító modul a trajektóriatervezőtől kapott sebességek felhasználásával generálja a DC motor meghajtásához szükséges feszültséget (4.3. fejezet), **d.** DC motor elektrodinamikai modelljének szimulálása az átalakítóból érkező feszültségek felhasználásával (4.4. fejezet), **e.** pálya szimulációs modul a motorok szögsebességei alapján (4.5. fejezet) és **f.** kommunikációs modul adattovábbításhoz (4.6. fejezet).

A 4.7. fejezet a modulrendszer grafikus úton programozott, **c**-**f**. modulokból álló láncolatának tesztelését ismerteti különböző példákon keresztül.

4.1. Pályatervezési modul

A pályatervezési modulnak szüksége van a kiindulópont navigációs adataira és a kívánt áthaladási és/vagy érkezési pontokra. A pontok ismeretében a vezérléshez kifejlesztendő algoritmus a pálya pontjainak generálja a Bezier-görbe és Hermite-görbe alkalmazásával.

4.1.1. Pályatervezési megoldások köbös görbék felhasználásával

Jelen alfejezet azokat a megoldásokat ismerteti, amelynél a kezdő- és végpontot interpolálja, a többi pontot viszont csak közelíti. A targonca csak vízszintesen közlekedik, ezért kétdimenziós teret feltételezek a pálya polinomokkal való leírásánál.

Bezier-görbék Berstein polinommal

A Bezier-görbéket kétféleképpen is fel lehet írni a [70] irodalom szerint, a Pascal háromszög és binomiális elvvel, illetve a Berstein formával. Mivel számos szakirodalom ([68]-[70],[72]-[73],[75]-[76],[78]-[80]) leggyakrabban az utóbbit alkalmazza, ezért a Berstein formát részesítem előnyben, amelyet az alábbiakban ismertetek.

A Bezier-görbe előállításához 4 db pontra van szükség, amelyek vektorformában jelennek meg: $\boldsymbol{P}_0 = (X_0, Y_0), \boldsymbol{P}_1 = (X_1, Y_1), \boldsymbol{P}_2 = (X_2, Y_2)$ és $\boldsymbol{P}_3 = (X_3, Y_3)$. A $\boldsymbol{P}_0$ a kezdőpont, a $\boldsymbol{P}_3$ a végpont, a $\boldsymbol{P}_1, \boldsymbol{P}_2$ pontok pedig a görbe vezérlőpontjainak vektora.

Megállapításom szerint a [70] irodalom írja le a legjobban az interpolációs megoldásokat, ezért a továbbiakban ezen irodalom alapján ismertetem Bezier görbe előállítását.

A 4 db pontra fel lehet írni a Bezier-görbét u intervallumra:

$$\boldsymbol{P}_B(u) = (u^3, u^2, \mathrm{u}, 1)\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} \boldsymbol{P}_0 \\ \boldsymbol{P}_1 \\ \boldsymbol{P}_2 \\ \boldsymbol{P}_3 \end{pmatrix}. \quad (1)$$

Az összefüggést másképpen felírva, amely már könnyen programozható:

$$\boldsymbol{P}_B(u) = (1-u)^3\boldsymbol{P}_0 + 3u(1-u)^2\boldsymbol{P}_1 + 3u^2(1-\mathrm{u})\boldsymbol{P}_2 + u^3\boldsymbol{P}_3. \quad (2)$$

Kétdimenzióban felírható az x és y tengely menti koordináták:

$$X_B(u) = (1-u)^3X_0 + 3u(1-u)^2X_1 + 3t^2(1-u)X_2 + u^3X_3, \quad (3)$$

$$Y_B(u) = (1-u)^3Y_0 + 3u(1-u)^2Y_1 + 3t^2(1-u)Y_2 + u^3Y_3. \quad (4)$$

Hermite interpolációs görbe

Az Hermite görbe egyenlete kiindulópontként az alábbiak szerint írható fel [70]:

$$\boldsymbol{P}_H(u) = \boldsymbol{F}(u)\boldsymbol{B}, \quad (5)$$

ahol az $\boldsymbol{F}(u)$ az u szerinti együtthatókat magába foglaló vektor

$$\boldsymbol{F}(u) = \big(F_0(u), F_1(u), F_2(u), F_3(u)\big). \quad (6)$$

A $\boldsymbol{B}$ mátrix a kezdeti pont vektorát $\big(\boldsymbol{P}_0 = (X_0, Y_0)\big)$, végpont vektorát $\big(\boldsymbol{P}_3 = (X_3, Y_3)\big)$, a kezdőpontból irányított vezérlővektor $\big(\boldsymbol{P}_{S1} = (X_{S1}, Y_{S1})\big)$, valamint a végpontba irányított vezérlővektor $\big(\boldsymbol{P}_{2E} = (X_{2E}, Y_{2E})\big)$ koordinátáit magába foglaló mátrix:

$$\boldsymbol{B} = (\boldsymbol{P}_0, \boldsymbol{P}_{\mathrm{S1}}, \boldsymbol{P}_{\mathrm{2E}}, \boldsymbol{P}_3)^T. \quad (7)$$

Síkbeli esetben a [70] irodalomban szerint az $\boldsymbol{F}$ vektor együtthatóira az alábbi összefüggések adódnak:

$$F_0(u) = (2u^3 - 3u^2 + 1), \quad (8)$$

$$F_1(u) = (-2u^3 + 3u^2), \quad (9)$$

$$F_2(u) = (u^3 - 2u^2 + \mathrm{u}), \quad (10)$$

$$F_3(u) = (u^3 - u^2). \quad (11)$$

Az együtthatókkal kifejezhető az $\boldsymbol{F}(u)$ vektor:

$$\boldsymbol{F}(u) = (u^3, u^2, \mathrm{u}, 1)\begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}. \quad (12)$$

A Bezier-görbéknél alkalmazott jelölésekhez hasonlóan az Hermite-görbe is kifejezhető az u paraméter szerinti, ahol az $u \in [0,1]$:

$$\boldsymbol{P}_H(u) = \boldsymbol{F}(u)\boldsymbol{B} = (u^3, u^2, \mathrm{u}, 1)\begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} \boldsymbol{P}_0 \\ \boldsymbol{P}_3 \\ \boldsymbol{P}_{\mathrm{S1}} \\ \boldsymbol{P}_{\mathrm{2E}} \end{pmatrix}. \quad (13)$$

A (13) kifejezés a programban használt alakban is felírható:

$$\boldsymbol{P}_H(u) = (2u^3 - 3u^2 + 1)\boldsymbol{P}_0 + (-2u^3 + 3u^2)\boldsymbol{P}_3 + (u^3 - 2u^2 + t)\boldsymbol{P}_{\mathrm{S1}} + (u^3 - u^2)\boldsymbol{P}_{\mathrm{2E}}. \quad (14)$$

A (14) alapján az x és y skalár koordináták is kifejezhetők:

$$X_H(u) = (2u^3 - 3u^2 + 1)X_0 + (-2u^3 + 3u^2)X_3 + (u^3 - 2u^2 + t)X_{S1} + (u^3 - u^2)X_{2E}, \quad (15)$$

$$Y_H(u) = (2u^3 - 3u^2 + 1)Y_0 + (-2u^3 + 3u^2)Y_3 + (u^3 - 2u^2 + t)Y_{S1} + (u^3 - u^2)Y_{2E}. \quad (16)$$

4.1.2. A pályatervezés kiindulási adatai

A LIDAR szenzor a következő három adatot szolgáltatja:

- $\boldsymbol{P}_{start} = (X_{start}, Y_{start})$ kezdeti pozíció és
- φ_{start} kezdeti orientáció.

A targonca előírt célpozíciójára a felhasználónak a következő három adatot kell megadnia:

- $\boldsymbol{P}_{end} = (X_{end}, Y_{end})$ célpozíció és
- φ_{end} célorientáció.

Megjegyzés: A targoncára szerelt LIDAR szenzor a szögértéket óramutató járásának ellentétes irányába értelmezi, ahogy a 4.1. ábrán az *Y* tengelyen látható.

A targonca mozgásának megvalósítása érdekében ezen két pont között kell megtervezni a pályagörbét, amely a vezérlőpontok segítségével írható le.

A következő lépésben definiálásra kerülnek $\boldsymbol{P}_1$ és $\boldsymbol{P}_2$ vezérlőpontok, amely a köbös görbék előállításához szükséges, a kezdeti pozícióból és orientációból, valamint a célpozícióból és orientációból származtatva. A kezdeti pont ($\boldsymbol{P}_{start}$) és az első vezérlőpont ($\boldsymbol{P}_1$) között származtatott vezérlővektor legyen $\boldsymbol{P}_{S1}$. Ezen vektor hosszúsága a számítások egyszerűsége miatt legyen $|\boldsymbol{P}_{S1}| = 1\mathrm{m}$. A második vezérlőpont ($\boldsymbol{P}_2$) és a célpont ($\boldsymbol{P}_{end}$) között származtatott vezérlővektor legyen $\boldsymbol{P}_{2E}$, szintén $|\boldsymbol{P}_{2E}| = 1\mathrm{m}$ vektorhosszúsággal. Az orientációból a $\boldsymbol{P}_1$ pont *X* és *Y* koordinátáját ki lehet számítani egyszerű trigonometrikus összefüggésekkel az egységnyi vektorhosszúság feltételezésével:

$$X_1 = X_{start} + sin(\varphi_{start}),\ Y_1 = Y_{start} + cos(\varphi_{start}). \quad (17)$$

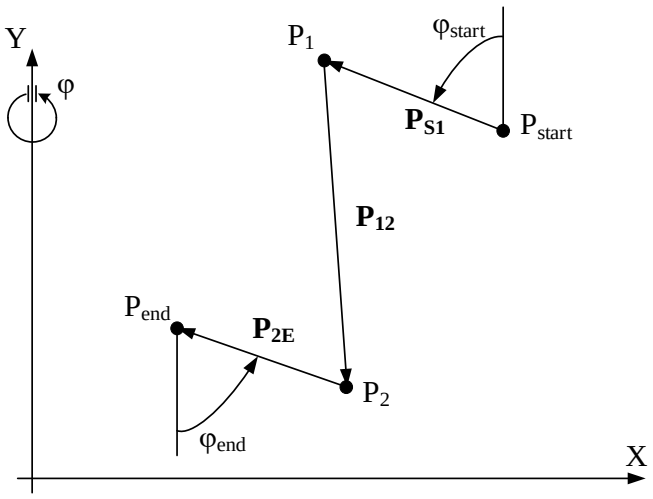
A $\boldsymbol{P}_2$ pont *X* és *Y* koordinátáját is ki lehet számítani az orientációból:

$$X_2 = X_{end} - sin(\varphi_{end}),\ Y_2 = Y_{end} - \cos(\varphi_{end}). \quad (18)$$

A két vezérlőponthoz a vezérlővektorok felírhatók:

$$\boldsymbol{P}_{S1} = \boldsymbol{P}_1 - \boldsymbol{P}_{start},\ \boldsymbol{P}_{2E} = \boldsymbol{P}_{end} - \boldsymbol{P}_2, \quad (19)$$

ahol $\boldsymbol{P}_{start}$, $\boldsymbol{P}_{S1}$, $\boldsymbol{P}_{2E}$ és $\boldsymbol{P}_{end}$ az origóból húzott helyvektorok, helyzetüket a 4.1. ábra mutatja.



4.1. ábra: Pontok és vektorok a pályatervezéshez

4.1.3. Mintapélda a Bezier-görbével generált pályára

A fent ismertetett algoritmust egy mintapálya generálására mutatom be. A pálya tervezése az alábbi kiinduló adatokra épül: $\boldsymbol{P}_{start} = (0m, 1m), \varphi_{start} = 255°$ és $\boldsymbol{P}_{end} = (-0.5m, -1m), \varphi_{end} = 255°$.

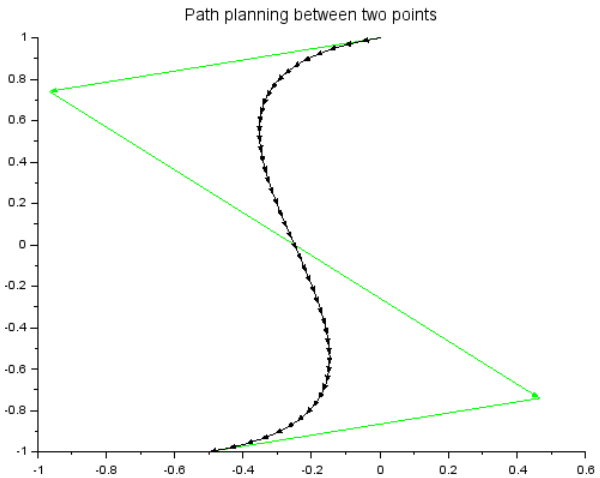
Értelemszerűen az origóból a kezdeti- és végponthoz, és a két vezérlőponthoz húzott vektorok az alábbiak szerint írhatók fel:

$$\boldsymbol{P}_0 = \boldsymbol{P}_{start}\,, \boldsymbol{P}_3 = \boldsymbol{P}_{end}, \tag{20}$$

$$\boldsymbol{P}_1 = (X_1, Y_1) = (X_{start} + sin(\varphi_{start})\,, Y_{start} + cos(\varphi_{start})), \tag{21}$$

$$\boldsymbol{P}_2 = (X_2, Y_2) = (X_{end} - sin(\varphi_{end})\,, Y_{end} - cos(\varphi_{end})). \tag{22}$$

Az u paraméter ebben az esetben 0-tól 1-ig változik 0,02 egyenletes lépésközzel, így a görbét 51 pont alkotja. A Scilab szoftverrel kirajzolt görbe a 4.2. ábrán látható. A Bezier-görbét az 51 ponttal a fekete vonal írja le, míg zöld színű egyenesek jelölik a $\boldsymbol{P}_{S1}$, $\boldsymbol{P}_{2E}$ és $\boldsymbol{P}_{SE}$ vektorokat.



4.2. ábra: Pályatervezés Bezier-görbével

4.1.4. Hermite-görbével tervezett mintapálya

Az 4.1.3. alfejezetben a Bezier-görbéhez megadott kiindulási adatokkal generálom az Hermite-görbét.

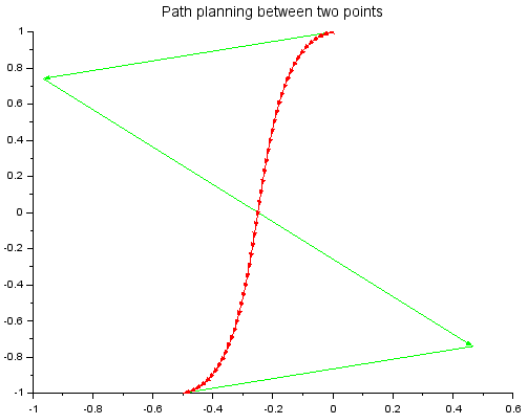
Jelen esetben az origóból a kezdeti- és végponthoz húzott vektorok, valamint a vezérlővektorok az alábbiak szerint írhatók fel:

$$\boldsymbol{P}_0 = \boldsymbol{P}_{start}\,, \boldsymbol{P}_3 = \boldsymbol{P}_{end}, \tag{23}$$

$$\boldsymbol{P}_{S1} = (sin(\varphi_{start})\,, cos(\varphi_{start})), \boldsymbol{P}_{2E} = (sin(\varphi_{end})\,, cos(\varphi_{end})). \tag{24}$$

Az u paraméter most is 0-tól 1-ig változik 0,02 egyenletes lépésközzel.

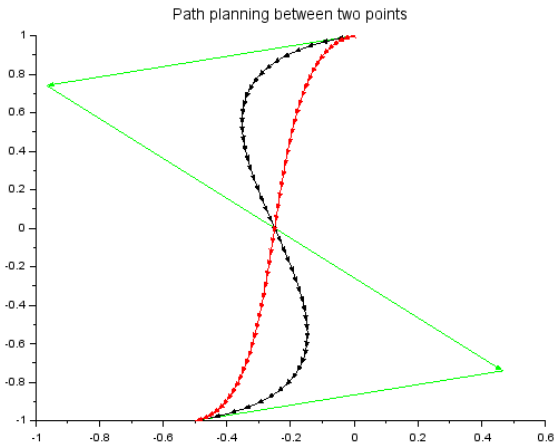
A Scilab szoftverrel kirajzolt görbe a 4.3. ábrán látható. A piros görbe jelöli az Hermite-görbét, míg a zöld színnel jelölt vektorok a $\boldsymbol{P}_{S1}$, $\boldsymbol{P}_{2E}$ és $\boldsymbol{P}_{SE}$ vektorokat mutatják.



4.3. ábra: Pályatervezés Hermite-görbével

4.1.5. Két görbe összevetése

Az előző két alfejezetben ismertetett görbéket veti össze a 4.4. ábra. Ahogy az előző fejezetekben, a piros görbe az Hermite-görbét, a fekete görbe a Bezier-görbét jelöli. Amint megállapítható a szimulációs ábrából, az Hermite-görbe lényegesen rövidebb utat eredményez, ugyanakkor a kezdeti- és végpont környékén kisebb ívű kanyarodást hajt végre.



4.4. ábra: Pályatervezési módszerek összevetése

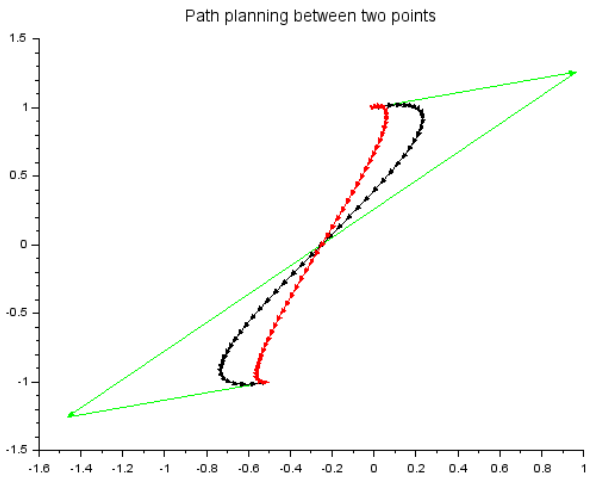
4.1.6. Pályatervezés során felmerülő megoldandó feladatok

Az alfejezet a targonca szerkezeti felépítéséből adódóan az alábbi megoldandó feladatokat ismerteti:

- *Ráállásból adódó többletút:* a targonca szimmetriájából adódóan, ha előremenet helyett hátramenetben indul el, adott esetben rövidebb út adódhat.
- *Hajtott kerekek közötti felezőpontra történő útvonal tervezés:* a targonca induló- és érkezőpontjainak felvétele a LIDAR szenzor helyzetére történik, azonban ez a két pont között távolság van, amit figyelembe kell venni a mozgás megtervezésénél.
- *Végső pozícióra ráállásból adódó probléma:* irányfordítás után a szimuláció ábrázolásában nem fordul át a targonca.
- *Szállítószalagokra való ráállásból adódó helyzet:* biztosítani kell, hogy a két szállítószalag közül az előre definiáltra álljon rá a targonca.
- *Több szegmens összekötése:* bonyolult pálya esetén az útvonal több szegmensből állhat, pl. ütközés elkerülése miatt, ezért ennek a megvalósítása is egy feladat.

Helytelen ráállásból adódó többletút

A jelenlegi példában a szögértékeknél az előzőkben felhasználthoz képest 180°-kal elforgatva indul el és érkezik meg a jármű, változatlan kiinduló- és végpont koordináták mellett, azaz $\boldsymbol{P}_{start} = (0m, 1m), \varphi_{start} = 75°$ és $\boldsymbol{P}_{end} = (-0.5m, -1m), \varphi_{end} = 75°$. Ebben az esetben a 4.5. ábrán látható görbék adódnak, megállapítható, hogy a kezdeti- és végső irányra helytelen ráállás miatt Hermite-görbénél ~3,1%, Bezier-görbénél ~12,75% többletút adódik az előző verzióhoz képest.



4.5. ábra: Pályatervezési módszerek összevetése irányváltás nélkül

A targonca szimmetrikus felépítése azonban megengedi azt, hogy a targonca irány váltásával az eredetihez képest ellentétes irányba induljon el vagy érkezzen meg, és ezáltal rövidebb

útra törekedjen. A program ezt egy feltételvizsgálattal oldja meg. A Bezier-görbe számításának alapjául szolgáló $\boldsymbol{P_1}$ és $\boldsymbol{P_2}$ vektorok esetén történik vizsgálat és szükség esetén az irányváltás:

$$\boldsymbol{P_1^{"}} = \begin{cases} \boldsymbol{P_1^{"}} = \boldsymbol{P_1}, ha\ \varphi_{S1-SE} < 90° \\ \boldsymbol{P_1^{"}} = \boldsymbol{P_{start}} - \boldsymbol{P_{S1}}, ha\ \varphi_{S1-SE} > 90° \end{cases}, \tag{25}$$

$$\boldsymbol{P_2^{"}} = \begin{cases} \boldsymbol{P_2^{"}} = \boldsymbol{P_2}, ha\ \varphi_{2E-SE} < 90° \\ \boldsymbol{P_2^{"}} = \boldsymbol{P_{end}} + \boldsymbol{P_{2E}}, ha\ \varphi_{2E-SE} > 90° \end{cases}. \tag{26}$$

azaz a kezdeti pontból húzott $\boldsymbol{P_1^{"}}$ vektor vagy a végpontba húzott $\boldsymbol{P_2^{"}}$ vektor irányát fordítja meg, ha a vizsgált $\boldsymbol{P_{S1}}$ vagy $\boldsymbol{P_{2E}}$ vektorok által bezárt szög nagyobb, mint 90 fok. A φ_{S1-SE} szög a $\boldsymbol{P_{S1}}$ és $\boldsymbol{P_{SE}}$ vektorok, míg φ_{2E-SE} szög a $\boldsymbol{P_{2E}}$ és $\boldsymbol{P_{SE}}$ vektorok által bezárt szög.

A feltételvizsgálat az Hermite-görbe alkalmazásánál $\boldsymbol{P_{S1}}$ és $\boldsymbol{P_{2E}}$ vezérlővektorokkal fejezhető a módosítás:

$$\boldsymbol{P_{S1}^{"}} = \begin{cases} \boldsymbol{P_{S1}^{"}} = \boldsymbol{P_{S1}}, ha\ \varphi_{S1-SE} < 90° \\ \boldsymbol{P_{S1}^{"}} = -\boldsymbol{P_{S1}}, ha\ \varphi_{S1-SE} > 90° \end{cases}, \tag{27}$$

$$\boldsymbol{P_{2E}^{"}} = \begin{cases} \boldsymbol{P_{2E}^{"}} = \boldsymbol{P_{2E}}, ha\ \varphi_{2E-SE} < 90° \\ \boldsymbol{P_{2E}^{"}} = -\boldsymbol{P_{2E}}, ha\ \varphi_{2E-SE} > 90° \end{cases}. \tag{28}$$

azaz a $\boldsymbol{P_{S1}^{"}}$ vagy $\boldsymbol{P_{2E}^{"}}$ vektorok irányát fordítja meg, ha a vizsgált $\boldsymbol{P_{S1}}$ vagy $\boldsymbol{P_{2E}}$ vektorok által bezárt szög nagyobb, mint 90 fok.

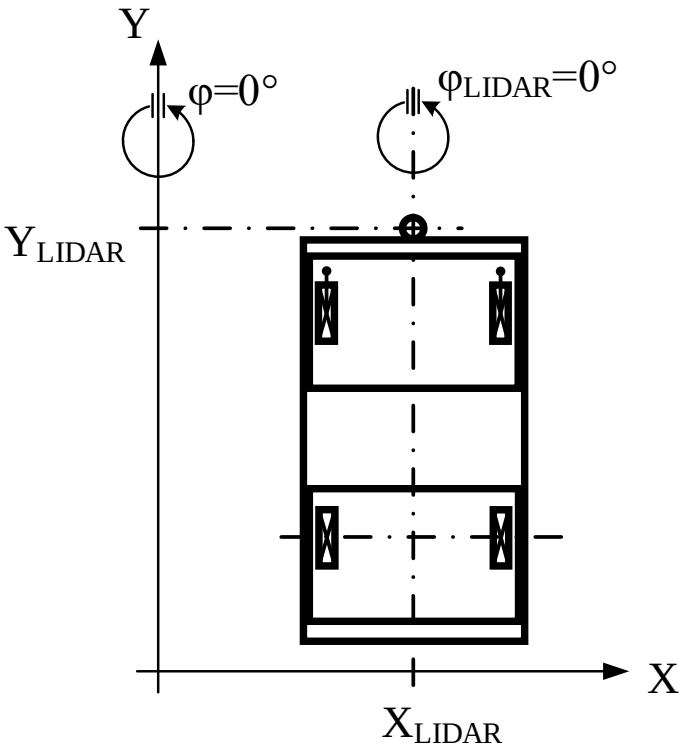
Az irányfordítás révén a 4.5. ábrán látható görbe a 4.4. ábrán látható útvonalra módosul, mivel a feltételvizsgálat során a vizsgált vektorok által bezárt szög minden esetben nagyobb a vizsgált értéknél.

Hajtott kerekek közötti felezőpontra történő útvonal tervezés

A fentiekben a generált útvonal számítása során a targoncát pontszerű testként feltételeztem. A valóságban azonban a targoncának felülnézetből téglalap alakú kiterjedése van, ezért a kezdeti értékek a programban a targonca kiterjedése miatt a következő adatokkal adhatók meg:

- $\boldsymbol{P}_{start,LIDAR} = (X_{start,LIDAR}, Y_{start,LIDAR})$: LIDAR szenzor által mért pozíció,
- φ_{start}: LIDAR szenzor által mért orientáció,
- $\boldsymbol{P}_{end,LIDAR} = (X_{end,LIDAR}, Y_{end,LIDAR})$: LIDAR szenzorral mért és beállított pozíció,
- φ_{end}: LIDAR szenzor által mért és beállított orientáció.

A munkatérben felvett koordináta-rendszerben elhelyezett targonca nullpontjának helyzetét és orientációját a 4.6. ábra szemlélteti.



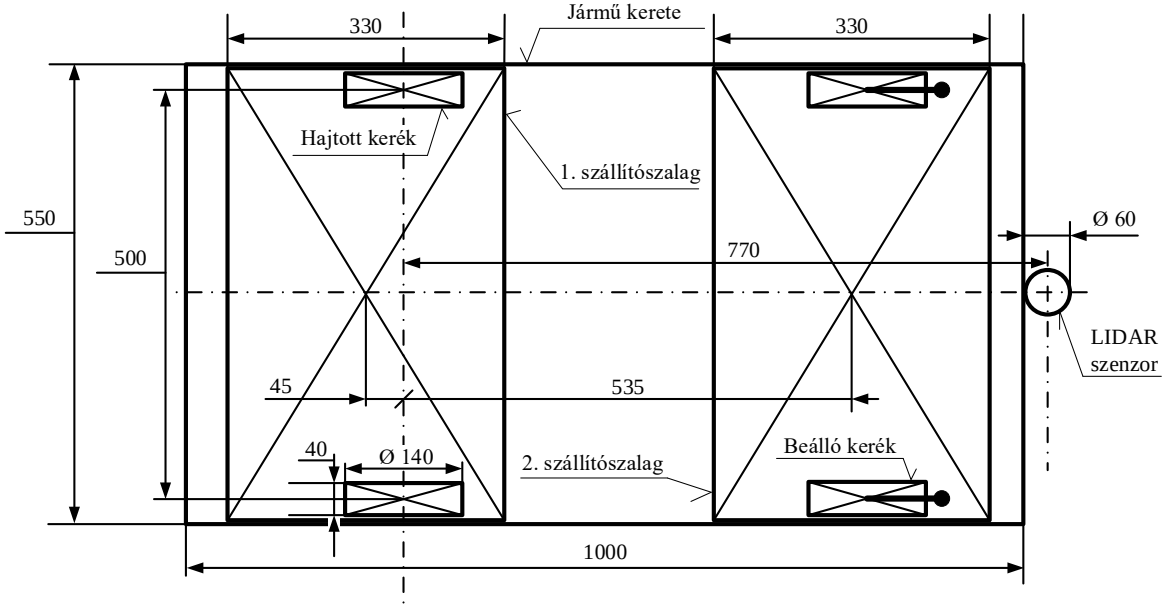
4.6. ábra: LIDAR szenzor helyzete a targoncához és a koordinátarendszerhez képest

Ezekből az értékekből kell származtatni az útvonalra tervezéshez használt $\boldsymbol{P}_{start}$ kezdeti- és $\boldsymbol{P}_{end}$ végpontot, felhasználva a kerekek középvonalának és a LIDAR szenzor középpontja közötti távolságot ($s_{LIDAR-W} = 0{,}770m$), amely a 4.7. ábrán látható.

$$\boldsymbol{P}_{start} = \left(X_{start,LIDAR} - s_{LIDAR-W} \cdot \sin\left(\varphi_{start}\right), Y_{start,LIDAR} + s_{LIDAR-W} \cdot \cos\left(\varphi_{start}\right)\right), \quad (29)$$

$$\boldsymbol{P}_{end} = \left(X_{end,LIDAR} - s_{LIDAR-W} \cdot \sin(\varphi_{end}), Y_{end,LIDAR} + s_{LIDAR-W} \cdot \cos(\varphi_{end})\right). \quad (30)$$

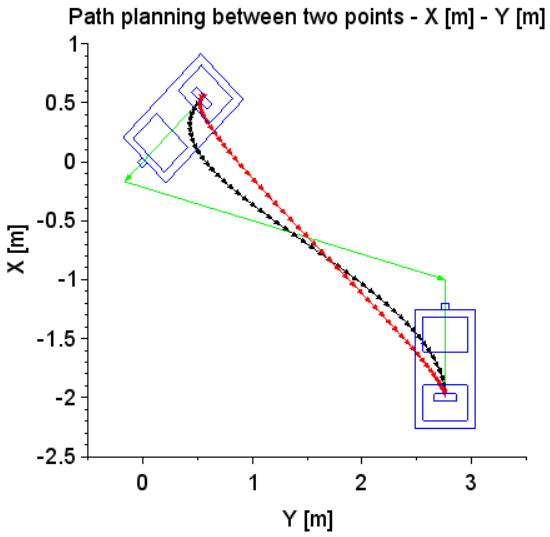
A targonca geometriai méretei a 4.7. ábrán láthatók.



4.7. ábra: Targonca geometriai méretei

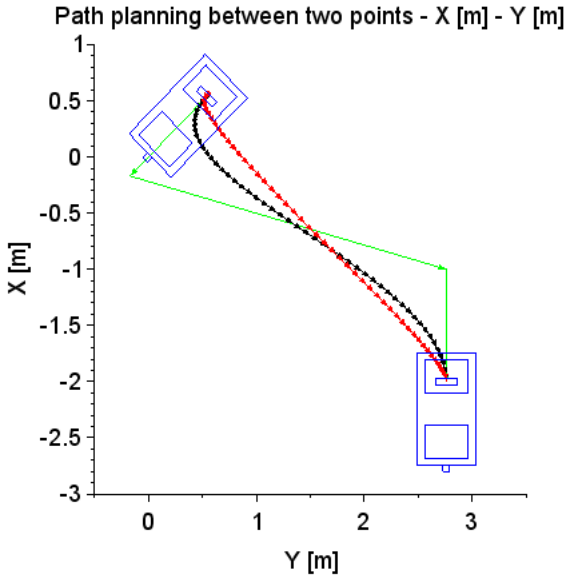
Végső pozícióra való ráállásnál felvetődő probléma

Az irányfordításnál figyelni kell, hogy a kezdeti pozícióban meghatározott mozgási irány a végső pozícióban is megvalósuljon a szimulációban.



4.8. ábra: Pályatervezési útvonalak inkorrekt érkezési szöggel

Amennyiben az irányfordításkor az egyes görbéknél levő feltételvizsgálat során csak az egyik esetben történik módosítás, akkor előfordulhat, hogy a targonca nem a megfelelő irányban érkezik meg a szimulációban (lásd 4.8. ábra), ahol a pozíciók és orientációk: $\boldsymbol{P}_{start} = (0m, 0m), \varphi_{start} = 315°$ és $\boldsymbol{P}_{end} = (2m, -2m), \varphi_{end} = 180°$.
Erre megoldást nyújt a végső pozíció elfordulási szögének módosítása 180°-kal, az eredményt a 4.9. ábra mutatja.



4.9. ábra: Pályatervezési útvonalak korrekt érkezési szöggel

Szállítószalagra való ráállásból adódó helyzet

Az előző alfejezetben részletezett problémán túl foglalkozni kell azzal, hogy a targonca célorientációja mellett a célpozíciója is változhat attól függően, hogy a targonca melyik szállítószalagját kell pozícionálni a görgőspálya rendszerhez. A célpozíció szállítószalagokhoz való kötésével biztosítani lehet azt, hogy a targonca át tudjon adni vagy el tudjon venni egységrakományt egy anyagmozgató rendszerre vagy rendszerről.

A célpozíció meghatározásánál a targoncát, ha úgy oda kell mozgatni, hogy az első szalagról történhessen a levétel, akkor leolvasható a célpozíció LIDAR szenzor által mért értéke.

Az első szállítószalag helyzete a LIDAR szenzor pozíciójához viszonyítva (lásd 4.7. ábra: $s_{belt_1-LIDAR} = 0{,}850m$):

$$X_{end,belt1} = X_{end,LIDAR} - s_{belt_1-LIDAR} \cdot \sin(\varphi_{end}), \tag{31}$$

$$Y_{end,belt1} = Y_{end,LIDAR} + s_{belt_1-LIDAR} \cdot \cos(\varphi_{end}). \tag{32}$$

Amennyiben az első szalag helyett a második szalagra történő ráállásra lenne szükség, egy eltolás jelenik meg a $\boldsymbol{P}_{end}$ célpozíciót tekintve, felhasználva a két szalag középpontjának távolságát (lásd 4.7. ábra: $s_{belt_1-belt_2} = 0{,}580m$):

$$X_{end} = X_{end,belt2} = X_{end,belt1} - s_{belt_1-belt_2} \cdot \sin(\varphi_{end}), \tag{33}$$

$$Y_{end} = Y_{end,belt2} = Y_{end,belt1} + s_{belt_1-belt_2} \cdot \cos(\varphi_{end}). \tag{34}$$

További megoldandó feladat az irányváltásnál az új célpozíció meghatározása mindkét szalagra, felhasználva az egyes szalagok középpontjának hosszirányú távolságát a kerekek középpontjának távolságától (lásd 4.7. ábra: $s_{belt_1-W} = 0{,}045m$, $s_{belt_2-W} = 0{,}535m$):

1. szalag esetén, amennyiben irányváltás szükséges:

$$X_{end} = X_{end,belt1} - 2 \cdot s_{belt_1-W} \cdot \sin(\varphi_{end}), \tag{35}$$

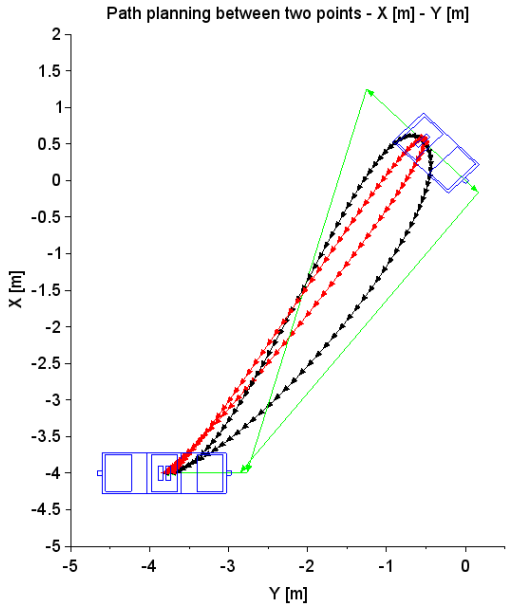
$$Y_{end} = Y_{end,belt1} + 2 \cdot s_{belt_1-W} \cdot \cos(\varphi_{end}), \tag{36}$$

2. szalag esetén, amennyiben irányváltás szükséges:

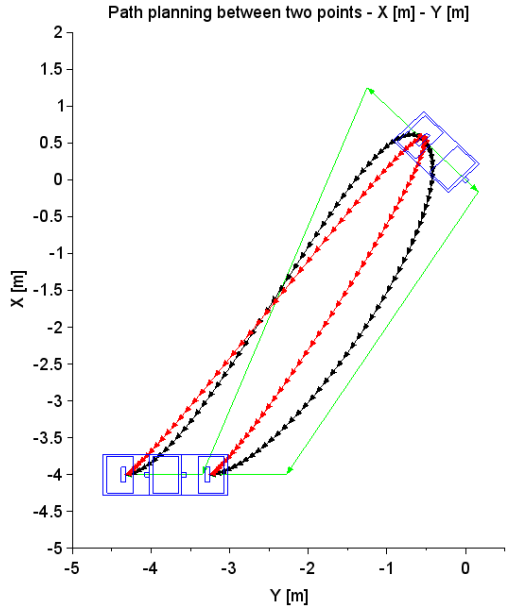
$$Y_{end} = Y_{end,belt2} - 2 \cdot s_{belt_2-W} \cdot \sin(\varphi_{end}), \tag{37}$$

$$Y_{end} = Y_{end,belt2} + 2 \cdot s_{belt_2-W} \cdot \cos(\varphi_{end}). \tag{38}$$

Az első szalagra a 4.10. ábra, a második szalagra a 4.11. ábra mutatja az előzőkben leírt összefüggéseket felhasználó megoldásokat, piros színnel az Hermite-görbét, fekete színnel a Bezier-görbét szemléltetve az irányváltás nélküli és az irányváltásos eseteket.



4.10. ábra: Útvonalak 1. szalagra ráállással, irányváltással vagy anélkül



4.11. ábra: Útvonalak 2. szalagra ráállással, irányváltással vagy anélkül

4.1.7. Több szegmens összekötése

Az eddigiek során az útvonaltervezés csak 1 db szegmenssel történt, azonban gyakran előfordulhat olyan eset, amikor az útvonalat több szegmensből kell összeállítani. Ilyenkor jellemzően az előre ismert akadályok elkerülése a cél. Több szegmens összekapcsolására, illetve használatára a [64] és [70] irodalmak adnak ajánlást, amelyet röviden ismertetek.

A [64] irodalom $\boldsymbol{W}_0, \boldsymbol{W}_1, \dots, \boldsymbol{W}_i, \dots, \boldsymbol{W}_N$ útpontsorozatot definiál, ahol $i = 1, \dots, N$ és N a legutolsó útpontot jelölő szám. A generált útvonal $\boldsymbol{W}_{i-1}$ és $\boldsymbol{W}_i$ szomszédos útpontokat kapcsolja össze. A $\boldsymbol{W}_{i-1}$ és $\boldsymbol{W}_i$ útpontok pozícióját és orientációját $\boldsymbol{q}_{i-1} =$

$[X_{i-1}, Y_{i-1}, \varphi_{i-1}]^T$ és $\boldsymbol{q}_i = [X_i, Y_i, \varphi_i]^T$ összefüggéssel jelöli. A Bezier-görbe segítségével a pálya a következőképpen állítható elő:

$$X(u_i) = (1-u_i)^3 X_{i-1} + 3u_i(1-u_i)^2 X_{ai} + 3u_i^2(1-u_i)X_{bi} + u_i^3 X_i, \quad (39)$$

$$Y(u_i) = (1-u_i)^3 Y_{i-1} + 3u_i(1-u_i)^2 Y_{ai} + 3u_i^2(1-u_i)Y_{bi} + u_i^3 Y_i, \quad (40)$$

ahol $X(u_i)$ és $Y(u_i)$ a pálya X- és Y-irányú függvényei, ahol $u_i \in [0,1]$; (X_{ai}, Y_{ai}) és (X_{bi}, Y_{bi}) meghatározandó paraméterek. Ha u_i 0-tól 1-ig változik, akkor $\big(x(u_i), y(u_i)\big)$ függvények értékei (X_{i-1}, Y_{i-1}) útponttól (X_i, Y_i) útpontig terjednek. Minden $\boldsymbol{W}_{i-1}\boldsymbol{W}_i$ szegmensre igaz az, hogy a szegmens $\boldsymbol{W}_i$ végső pontja a következő $\boldsymbol{W}_i\boldsymbol{W}_{i+1}$ szegmens kezdő pontja lesz. A (X_{ai}, Y_{ai}) és (X_{bi}, Y_{bi}) paraméterek meghatározását ugyan leírja az irodalom, de véleményem szerint ez már túl nehézkes, ugyanis további, ismeretlen paraméterek meghatározását igényli.

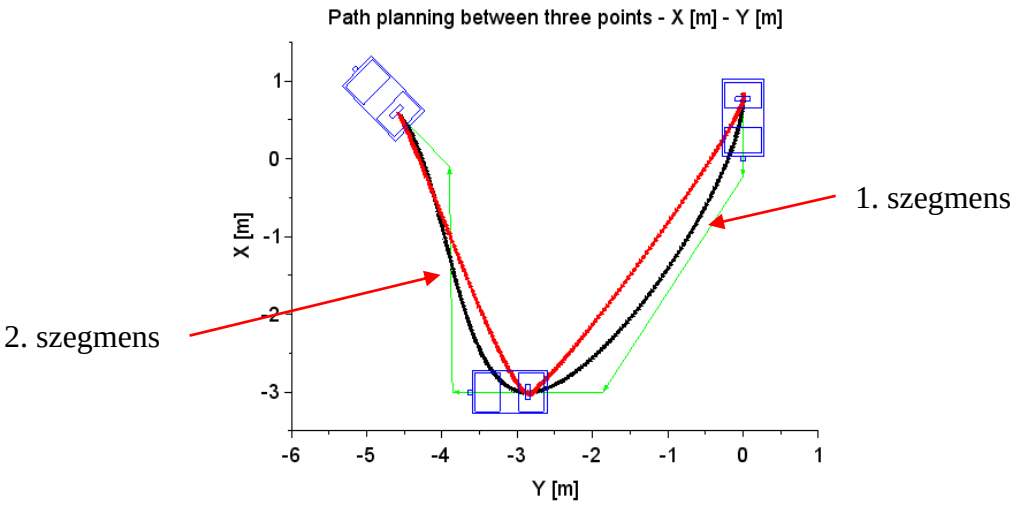
A [70] irodalom leírja, hogy egy n-dimenziós Bezier-görbe számítása számításigényes lenne, ezért érdemes több Bezier-szegmenst összekapcsolni, mindegyiket 4-6 ponttal definiálva. Két szegmens összekapcsolásához az irodalom $\boldsymbol{P}_0, \boldsymbol{P}_1, \dots, \boldsymbol{P}_n$ és $\boldsymbol{Q}_0, \boldsymbol{Q}_1, \dots, \boldsymbol{Q}_n$ pontkészletet definiál, ahol $n = 4 \div 6$. A kapcsolódás érdekében a $\boldsymbol{P}_n$ pontnak $\boldsymbol{Q}_0$ ponttal meg kell egyeznie. Az irodalom ezen felül megállapítja, hogy a $\boldsymbol{P}_{n-1}, \boldsymbol{P}_n$ és $\boldsymbol{Q}_1$ pontoknak egy vonalba kell esnie, ezáltal teljesül a sima kapcsolódás a szegmensek között. Az irodalom ettől nagyobb számú szegmens összekötésére már nem ad megoldást.

A disszertációban alkalmazott megoldásban a szegmens sorszámát jobb felső indexként jelölöm. A program kezdeti értékként a legelső szegmens $\boldsymbol{P}^1_{start,LIDAR} = (X^1_{start,LIDAR}, Y^1_{start,LIDAR})$ mért kezdeti pozícióját és φ^1_{start} mért kezdeti orientációját, valamint minden egyes szegmensre az egyes pontok $\boldsymbol{P}^S_{end,LIDAR} = (X^S_{end,LIDAR}, Y^S_{end,LIDAR})$ mért végső pozícióját és φ^S_{end} mért orientációját használja fel, ahol $S = 1,2, \dots, N_S$ az aktuális szegmens sorszámát jelöli és maximálisan az előre definiált szegmensek darabszáma (N_S) lehet. A legelső szegmens $\boldsymbol{P}^1_{end} = (X^1_{end}, Y^1_{end})$ végső pont pozíciója és φ^1_{end} orientációja az előző alfejezetekben ismertetett módon számítható ki. A második szegmenstől kezdve az egyes szegmensek kezdő pozíciója és orientációja az előző szegmens célpozícióját és célorientációját kapja meg az alábbi módon:

$$\boldsymbol{P}^{\mathrm{S}}_{start,LIDAR} = \boldsymbol{P}^{S-1}_{end}, \text{ ha } S > 1, \quad (41)$$

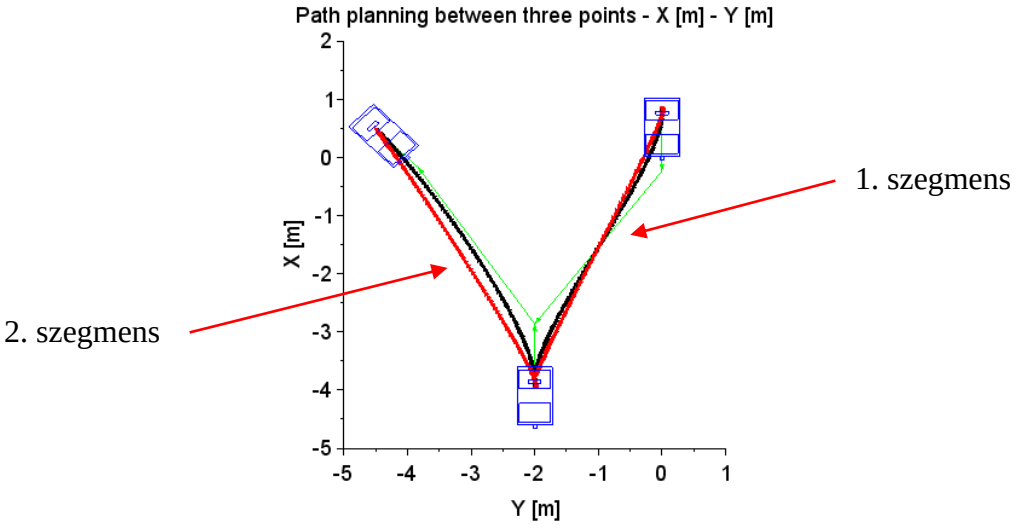
$$\varphi^{\mathrm{S}}_{start} = \varphi^{S-1}_{end}, \text{ ha } S > 1. \quad (42)$$

Ezután az egyes szegmensek $\boldsymbol{P}^S_{end} = (X^S_{end}, Y^S_{end})$ célpozíciója és φ^S_{end} célorientációja a (33)-(38) összefüggésekkel határozható meg.



4.12. ábra: Útvonaltervezés 3 pont közé 2 szegmenssel irányfordítás nélkül

A 4.12. ábra példát mutat egy két szegmenses útvonaltervezésre. A kezdeti ponthoz tartozó $\boldsymbol{P}^1_{start,LIDAR} = (0m, 0m)$, $\varphi^1_{start} = 0°$ pozícióval és orientációval, az első szegmenshez tartozó $\boldsymbol{P}^1_{end,LIDAR} = (-2m, -3m)$, $\varphi^1_{end} = 90°$ célpozícióval és célorientációval és a második szegmenshez tartozó $\boldsymbol{P}^2_{end,LIDAR} = (-4m, 0m)$, $\varphi^2_{end} = 45°$ célpozícióval és célorientációval. Ebben az esetben az első és a második szegmens határán közös az érintő. Előfordulhat azonban olyan eset, amikor a köztes határpontból nem ugyanolyan irányba folytatja az útját a jármű, hanem az előzővel ellentétes irányba. Erre példát az 4.13. ábra mutat. Itt a 2 szegmenses útvonaltervezésre, a kezdeti ponthoz tartozó $\boldsymbol{P}^1_{start,LIDAR} = (0m, 0m)$, $\varphi^1_{start} = 0°$ pozíció és orientáció, a 1. szegmenshez tartozó $\boldsymbol{P}^1_{end,LIDAR} = (-2m, -3m)$, $\varphi^1_{end} = 180°$ célpozíció és célorientáció és a 2. szegmenshez tartozó $\boldsymbol{P}^2_{end,LIDAR} = (-4m, 0m)$, $\varphi^2_{end} = 45°$ célpozíció és célorientáció adatokat használtam fel.



4.13. ábra: Útvonaltervezés 3 pont közé 2 szegmenssel irányfordítással

1. TÉZIS: Kidolgoztam egy olyan új pályatervező megoldást, amely egyszerre két módszerrel (Hermite-görbe és Bezier-görbe) határozza meg egy két szállítószalagos vezető nélküli szállítótargonca mozgásának vezérléséhez szükséges pályapontokat a kezdeti- és cél pozíciók és -orientációk felhasználásával.

Jelen tézis tartalma az [S1] Scopus által indexált Q2-es minőségű folyóiratban került ismertetésre.

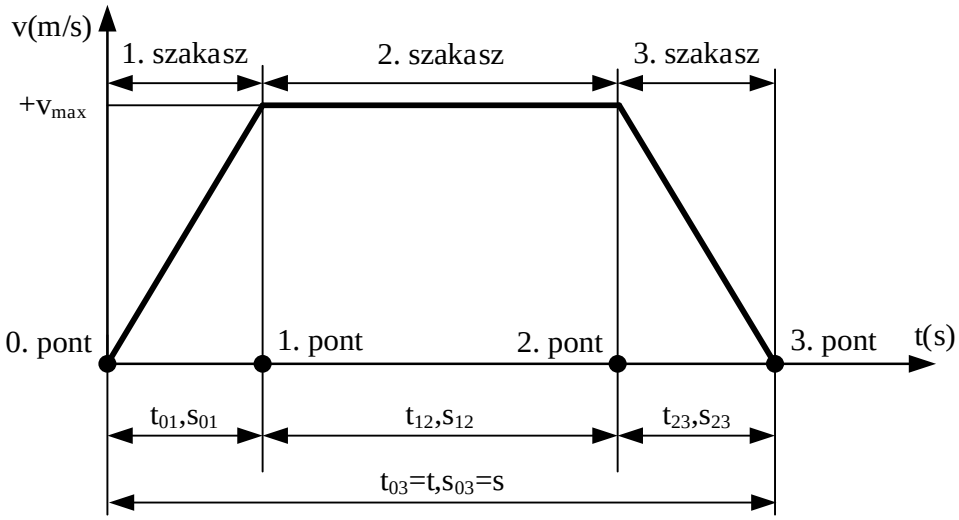
4.2. Trajektóriatervezés

A targonca mozgásvezérlésének trajektóriatervező része (**b**. modul) a pályatervező rész (**a**. modul) által kiszámított pályaadatokat használja fel, és az egyes pályapontokhoz hozzárendeli a forgáspont sebességét és a targonca forgáspontján átmenő függőleges tengely körüli szögsebességét az időadatok felhasználásával.

4.2.1. A trajektóriatervezéshez kezdetben rendelkezésre álló adatok

A [64] irodalom egy előre definiált sebesség- és szögsebességprofilt használ a trajektóriatervezéshez. Az irodalomban bemutatott kísérlet során a robot a kívánt sebességre gyorsul ($0{,}3\frac{m}{s}$ sebességre) 3 másodpercen belül (1. szakasz), majd fenntartja ezt a sebességet (2. szakasz), emellett a hatodik másodpercnél a szögsebesség egyenletesen növekszik, elérve a $\frac{\pi}{2}\frac{rad}{s}$ értéket a kilencedik másodpercnél. Végül mind a haladó-, mind a szögsebesség a tizedik másodpercnél kezd csökkenni, majd végül a tizenharmadik másodpercnél eléri a nullát (3. szakasz). Amint megfigyelhető, a szögsebesség itt adott volt, azonban az általam használt megoldásban a szögsebesség a pályatervező algoritmus által előállított görbéből határozza meg. A szögsebesség függvény előállítása előtt azonban a sebesség függvényt kell meghatározni a görbék útpontjaiból.

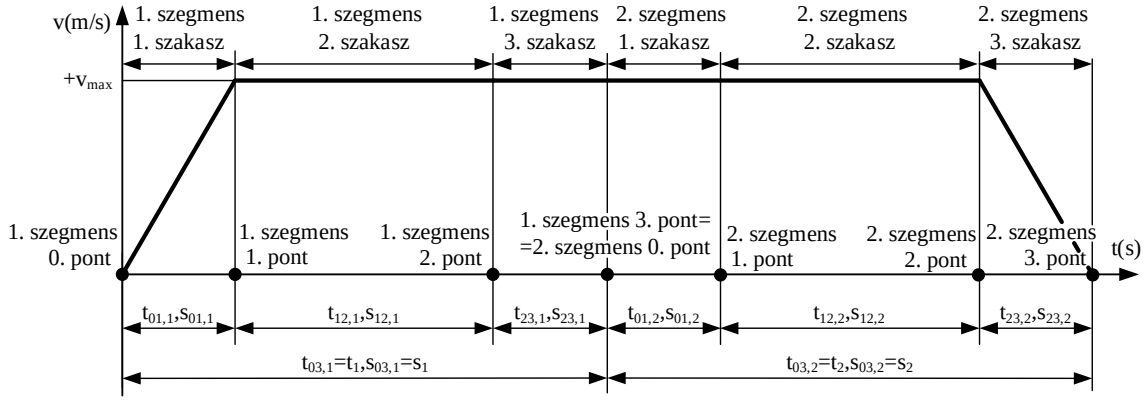
A sebességprofilt illetően az irodalomban is használt, 4.14. ábrán látható függvényt követem. Egy darab szegmens esetén a targonca $0.5s$ alatt gyorsul fel a kívánt sebességre (1. szakasz), majd a végén $0.5s$ alatt lassul le nullára (3. szakasz), a kettő között állandó értéken tartja a sebességet (2. szakasz). Az irodalomban a köztes időt előre definiálták, azonban a disszertációban a trajektóriatervező algoritmus az időt az úthosszból számítja ki.



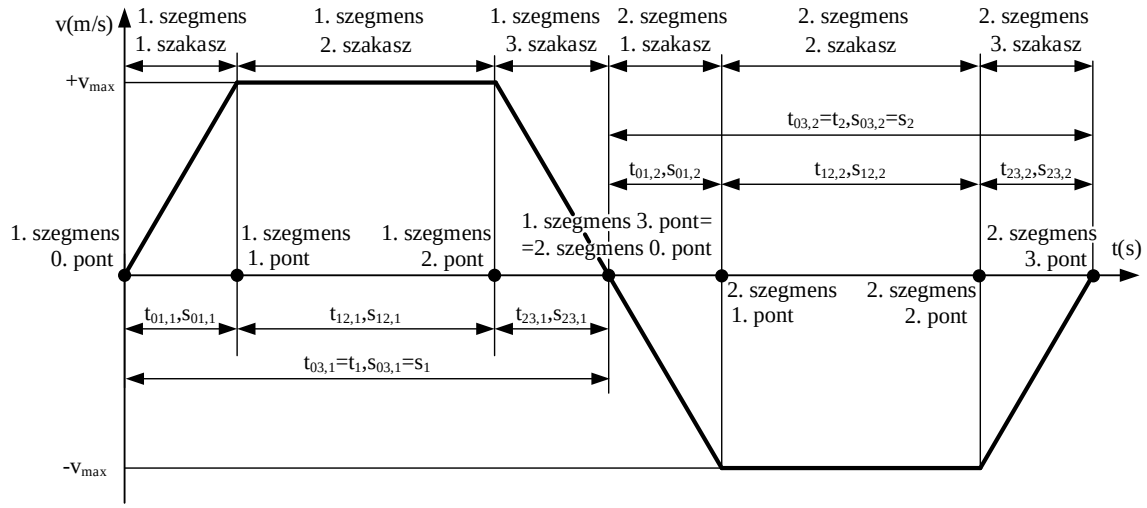
4.14. ábra: Sebességprofil 1 szegmens esetén

További megoldandó feladat a sebességprofil megvalósítása több szegmensre. Itt alapvetően két eset lehetséges szegmensváltások között:

- ha ugyanazon irányba (példák kettő szegmensre a 4.12. és a 4.15. ábra) azonos sebességgel folytatja az útját a targonca, akkor nem kell lelassítania meg felgyorsítania,
- ha ellenkező irányba (példák kettő szegmensre a 4.13. és a 4.16. ábra) irányváltással folytatja az útját a targonca, akkor le kell lassítania nullára, majd felgyorsítania az ellenkező irányú sebességre.



4.15. ábra: Sebességprofil 2 szegmens esetén irányváltás nélkül



4.16. ábra: Sebességprofil 2 szegmens esetén irányváltással

Ennek figyelembevételével változnak a szegmensek szakaszaira eső úthosszak és időértékek. A szakaszok adataihoz először meg kell határozni a szegmensek teljes úthosszát:

$$s^S = \sum_{j=1}^{u_{end}} s_{diff}^S(j)\ [m], \tag{43}$$

$$s_{diff}^S(\mathrm{j}) = \sqrt{\left(X_j^S - X_{\mathrm{j}-1}^S\right)^2 + \left(Y_j^S - Y_{\mathrm{j}-1}^S\right)^2}\ [m], \text{ ha } j \geq 2, \tag{44}$$

ahol s^S az adott S-ik szegmens teljes hossza (ahol $S = 1,2, \dots, N_S$ görbe sorszáma) és $s_{diff}^S(j)$ pedig a görbe j-ik és $(j-1)$-ik útpontjai közötti távolság (ahol $j = 1,2, \dots, u_{end}$).

A pálya s_{teljes} teljes hosszát az egyes szegmensek teljes úthosszának összege szolgáltatja:

$$s_{teljes} = \sum_{s=1}^{N_S} s^S\ [m]. \tag{45}$$

4.2.2. A trajektóriatervezéshez szükséges szakaszidők és úthosszak számítása

A szegmensek úthossz- és időadatai a következők szerint definiálhatók:

1. szakasz: $t_{01} = 0.5s$, minden szegmensre igaz, 0. és 1. pontja közötti időhossz:
   a. $s_{01}^S = v_{max} \cdot t_{01}\ [m]$, minden szegmensre igaz, 0. és 1. pontja közötti úthossz, **ha** a szomszédos szegmens-irányok megegyeznek;
   b. $s_{01}^S = a_{max} \cdot \frac{(t_{01})^2}{2}\ [m]$, minden szegmensre igaz, 0. és 1. pontja közötti úthossz, **ha** a szomszédos szegmens-irányok nem egyeznek meg, pl. irányváltás vagy első szegmens esetén;
2. szakasz: $s_{12}^S = s^S - s_{01}^S - s_{23}^S\ [m]$, adott szegmens 1. és 2. pontja közötti úthossz, és ebből $t_{12}^S = \frac{s_{12}^S}{v_{max}}[s]$, szegmens 1. és 2. pontja közötti időhossz;

3. szakasz: $t_{23} = 0.5s$, minden szegmensre igaz, 2. és 3. pontja közötti időhossz:

   a. $s_{23}^S = v_{max} \cdot t_{23}\ [m]$, minden szegmensre igaz, 2. és 3. pontja közötti úthossz, **ha** a szomszédos szegmens-irányok megegyeznek;

   b. $s_{23}^S = v_{max} \cdot t_{23} - a_{max} \cdot \frac{(t_{23})^2}{2}\ [m]$, minden szegmensre igaz, 2. és 3. pontja közötti úthossz, **ha** a szomszédos szegmens-irányok nem egyeznek meg, pl. irányváltás vagy utolsó szegmens esetén.

Végül meghatározható a szegmens 0. és 3. pont közötti, azaz az adott szegmens teljes ideje:

$$t^S = t_{03}^S = t_{01} + t_{12}^S + t_{23}\ [s]. \tag{46}$$

4.2.3. A trajektóriatervezés során adódó sebességprofil előállítása

A 4.2.1. alfejezetben ismertetett kiindulási adatok, és a 4.2.2. alfejezetben ismertetett módon kiszámított időértékek és úthosszak szükségesek a sebességadatok számításához. Az útpontokhoz rendelt $s_{actual}^S(j)$ aktuális úthossz a következőképpen számítható a Bezier-görbénél és az Hermite-görbénél is előállított görbék pontjainak X és Y koordinátájából:

$$s_{actual}^S(1) = s^{S-1}\ [m], \text{ ha } S \geq 2, \tag{47}$$

$$s_{actual}^S(j) = s_{actual}^S(j-1) + s_{diff}^S(j)\ [m], \text{ ahol } j = 2, \dots, u_{end}. \tag{48}$$

Az egyes szegmensekben a maximális sebesség előjele az iránytól függően változhat, ezt a k_{start}^S iránykoefficiens írja le. Az útpontokhoz rendelt idők, sebességek és gyorsulások a jól ismert négyzetes úttörvény és az egyes szakaszoknál a kezdeti időértékek és úthosszak határozzák meg:

1. szakasz: adott szegmens 0. és 1. pontja közötti idő, sebesség és gyorsulás:

   a. $t^S(j) = t_0^S + \frac{s_{actual}^S(j) - s_{actual}^S(1)}{v_{max}}\ [s];\ v^S(j) = k_{start}^S \cdot v_{max}\ \left[\frac{m}{s}\right]; a^S(j) = 0\ \left[\frac{m}{s^2}\right]$, S. szegmensen belül a j. útponthoz tartozó aktuális idő, sebesség és gyorsulás, **ha** a szomszédos szegmens-irányok megegyeznek;

   b. $t^S(j) = t_0^S + \sqrt{\frac{2 \cdot (s_{actual}^S(j) - s_{actual}^S(1))}{a_{max}}}\ [s]; v^S(j) = k_{start}^S \cdot a_{max} \cdot (t^S(j) - t_0^S)\ \left[\frac{m}{s}\right]$; $a^S(j) = k_{start}^S \cdot a_{max}\ \left[\frac{m}{s^2}\right]$, S. szegmensen belül a j. útponthoz tartozó aktuális idő, sebesség és gyorsulás, **ha** a szomszédos szegmens-irányok nem egyeznek meg, pl. irányváltás vagy első szegmens esetén;

2. szakasz: $t^S(j) = t_0^S + t_{01} + \frac{s_{actual}^S(j) - s_{01}^S - s_{actual}^S(1)}{v_{max}}\ [s];\ v^S(j) = k_{start}^S \cdot v_{max}\ \left[\frac{m}{s}\right]$; $a^S(j) = 0\ \left[\frac{m}{s^2}\right]$, S. szegmens 1. és 2. pontja közötti aktuális idő, sebesség és gyorsulás;

3. szakasz: adott szegmens 2. és 3. pontja közötti aktuális idő, sebesség és gyorsulás:

a. $t^S(j) = t_0^S + t_{01} + t_{12}^S + \frac{s_{actual}^S(j) - s_{12}^S - s_{01}^S - s_{actual}^S(1)}{v_{max}}$ $[s]$; $v^S(j) = k_{start}^S \cdot v_{max}$ $\left[\frac{m}{s}\right]$;

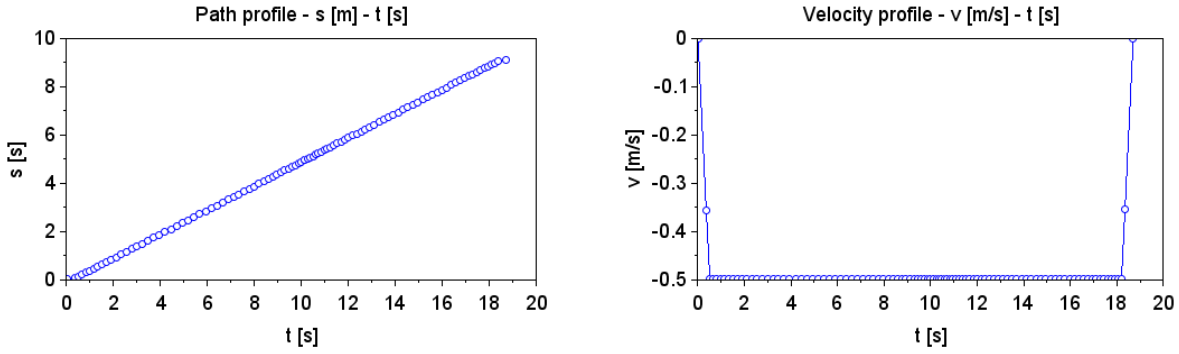
$a^S(i) = 0$ $\left[\frac{m}{s^2}\right]$, S. szegmensen belül a j. útpontokhoz tartozó aktuális idő, sebesség és gyorsulás, **ha** a szomszédos szegmens-irányok megegyeznek;

b. $t^S(j) = t_0^S + t_{01} + t_{12}^S + \frac{v_{max} - \sqrt{v_{\max}^2 - 2 \cdot \left(s_{actual}^S(j) - s_{12}^S - s_{01}^S - s_{actual}^S(1)\right) \cdot a_{max}}}{a_{max}}$ $[s]$;

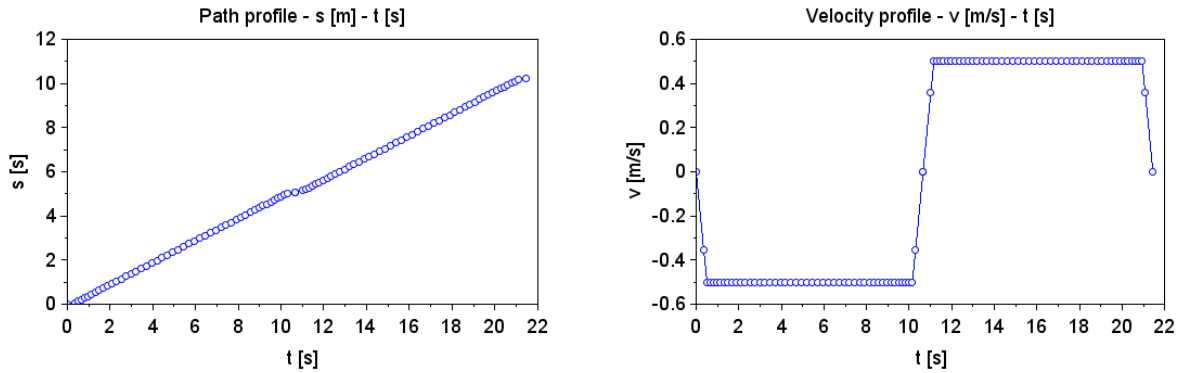
$v^S(j) = k_{start}^S \cdot \left(\mathrm{v}_{\max} - a_{max} \cdot (t^S(j) - t_{01} - t_{12}^S - t_0^S)\right)$ $\left[\frac{m}{s}\right]$;

$a^S(j) = k_{start}^S \cdot a_{max}$ $\left[\frac{m}{s^2}\right]$, S. szegmensen belül a j. útpontokhoz tartozó aktuális idő, sebesség és gyorsulás, **ha** a szomszédos szegmens-irányok nem egyeznek meg, pl. irányváltás vagy utolsó szegmens esetén.

Az algoritmus alapján a 4.1.7. alfejezetben bemutatott két példára Scilab szoftverben szimulált úthossz-idő és sebesség-idő diagramjai rendre a 4.17. és a 4.18. ábrákon láthatók. Mivel az első szegmens esetén a hátramenet miatt a mozgás iránya negatív, így adódik a 4.15. és a 4.16. ábrákhoz képest ellentétes görbe.



4.17. ábra: Trajektóriatervezés 2 szegmensre irányfordítás nélkül – úthossz-idő és sebesség-idő diagramja



4.18. ábra: Trajektóriatervezés 2 szegmensre irányfordítással – úthossz-idő és sebesség-idő diagramja

4.2.4. A trajektóriatervezés során adódó szögsebességprofil előállítása

A targonca haladási sebessége mellett szükség van a targonca forgáspontján átmenő függőleges tengely körüli szögsebesség meghatározására is, ugyanis a targonca túlnyomórészt nemcsak egyenes vonalban, hanem íven is halad. A szögsebesség független a sebességtől, hanem az Hermite- és Bezier-görbék adataiból fog közvetlenül származni. A továbbiakban $k = 1,2, \dots, N_S \cdot u_{end}$ a teljes, több szegmensből álló görbe összes útpontjainak sorszámát jelenti.

A szögsebességprofil meghatározásához először az egyes útpontokhoz tartozó szögekre van szükség. Az egyes szögértékeket a szomszédos útpontok között definiált vektorokból lehet meghatározni:

$$\boldsymbol{P}_{actual}(k) = [X_{actual}(k), Y_{actual}(k)] = \boldsymbol{P}_k - \boldsymbol{P}_{k-1}, \text{ ha } k \geq 2. \tag{49}$$

A vektoroknak az Y tengelyen elhelyezett $\boldsymbol{P}_{vertical} = [X_{vertical}; Y_{vertical}] = [0; 1]$ vektorhoz képest bezárt szöge a koszinusztételből számítható:

$$\varphi_{actual}(k) = \arccos\left(\frac{X_{actual}(k) \cdot X_{vertical} + Y_{actual}(k) \cdot Y_{vertical}}{|\boldsymbol{P}_{actual}(k)| \cdot |\boldsymbol{P}_{vertical}|}\right). \tag{50}$$

Mivel $X_{vertical} = 0$ és $|\boldsymbol{P}_{vertical}| = 1$, azaz a $\boldsymbol{P}_{vertical}$ vektor hossza egy, ezért az összefüggés az alábbiak szerint egyszerűsödik:

$$\varphi_{actual}(k) = \arccos\left(\frac{Y_{actual}(k) \cdot Y_{vertical}}{|\boldsymbol{P}_{actual}(k)|}\right), \tag{51}$$

ahol $\varphi_{actual}(k) \in [0°, 180°]$.

Az (51) képlettel csak 0° és 180° közötti szögérték számítható, ezen probléma feloldása céljából az alábbi feltétel kerül bevezetésre:

$$\varphi_{actual}(k) = \begin{Bmatrix} \varphi_{actual}(k) = \varphi_{actual}(k), \text{ha } X_{actual}(k) < 0 \\ \varphi_{actual}(k) = 360° - \varphi_{actual}(k), \text{ha } X_{actual}(k) > 0 \end{Bmatrix}, \tag{52}$$

ahol $\varphi_{actual}(k) \in [0°, 360°]$.

Ahhoz, hogy az egyes pontokban levő szögsebességet számítani lehessen, a közben eltelt időre és az egyes pontok közötti szögeltérésre van szükség:

$$\varphi_{diff}(k) = \varphi_{actual}(k) - \varphi_{actual}(k-1), \tag{53}$$

ahol $k \geq 2$ és $\varphi_{diff}(k) \in [-360°, 360°]$.

Ennek feloldására egy feltételrendszert építettem be a programba az egyes pontok közötti szögeltérésre vonatkozóan:

$$\varphi_{diff}(k) = \begin{cases} \varphi_{diff}(k) = \varphi_{diff}(k), \text{ha } \varphi_{diff}(k) \in [-180°, 180°] \\ \varphi_{diff}(k) = \varphi_{diff}(k) - 360°, \text{ha } \varphi_{diff}(k) \in (180°, 360°] \\ \varphi_{diff}(k) = \varphi_{diff}(k) + 360°, \text{ha } \varphi_{diff}(k) \in [-360°, -180°) \end{cases}, \tag{54}$$

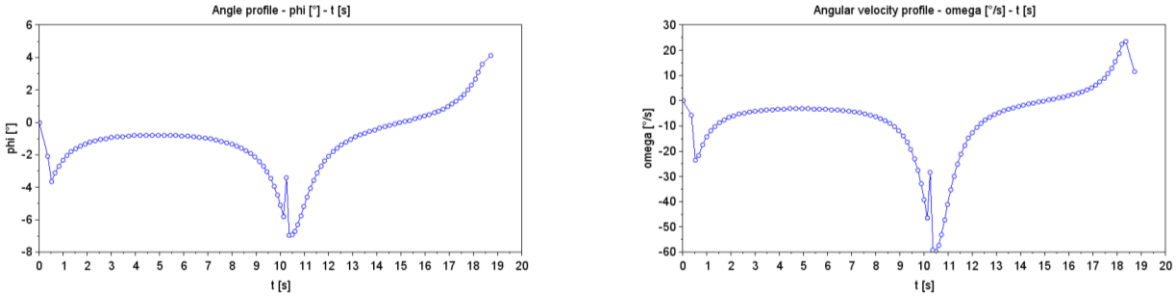
ahol így már $\varphi_{diff}(k) \in [-180°, 180°]$ lesz.

Végül az egyes útpontokhoz tartozó targonca forgáspontján átmenő függőleges tengely körüli szögsebesség már kiszámítható:

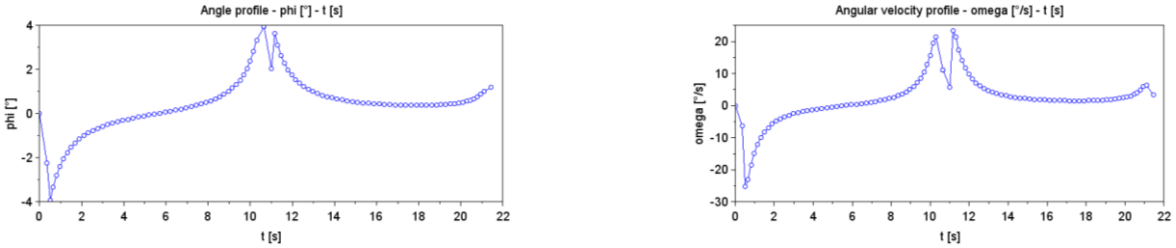
$$\omega(k) = \frac{\varphi_{diff}(k)}{t(k)-t(k-1)}, \tag{55}$$

ahol $k > 1$, az $\omega(k)$ szögsebesség kifejezhető $\left[\frac{rad}{s}\right]$-ban és $\left[\frac{^\circ}{s}\right]$-ban is.

Az eddigiek alapján a 4.12. és a 4.13. ábrákon mutatott példák elfordulási szög-idő és szögsebesség-idő diagramjai rendre a 4.19. és a 4.20. ábrákon láthatók. Megállapítható, hogy a szögsebesség-idő görbék és a 4.17. és a 4.18. ábrákon látható függvények jellegükben eltérnek.



4.19. ábra: Trajektóriatervezés 2 szegmenssel irányfordítás nélkül – elfordulási szög-idő és szögsebesség-idő diagramja



4.20. ábra: Trajektóriatervezés 2 szegmenssel irányfordítással – elfordulási szög-idő és szögsebesség-idő diagramja

4.2.5. A trajektóriatervezés során adódó keréksebességek előállítása

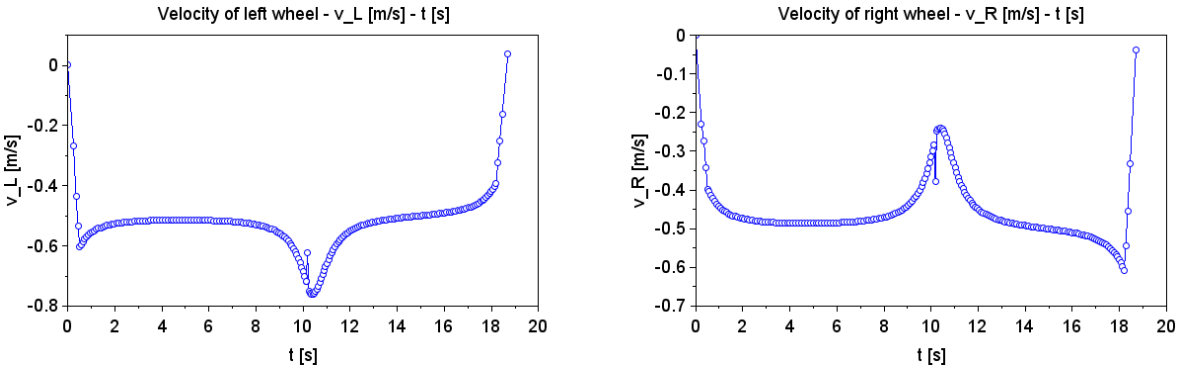
A targonca kerekei közötti felezőpont sebessége és a targonca szögsebessége ismeretében a jobb és bal oldali kerekek sebessége meghatározható:

$$v_R(k) = v(k) - \frac{b}{2} \cdot \omega(k) \left[\frac{m}{s}\right], \tag{56}$$

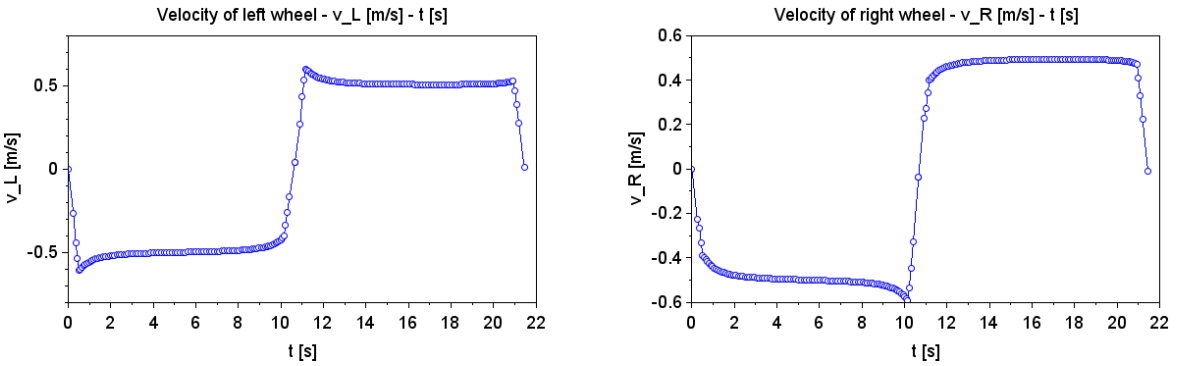
$$v_L(k) = v(k) + \frac{b}{2} \cdot \omega(k) \left[\frac{m}{s}\right], \tag{57}$$

ahol $v_R(k)$ és $v_L(k)$ rendre a jobb és bal oldali kerék középpontjainak sebessége, a $b = 500mm = 0{,}5m$ az egyes kerekek középpontjai közötti távolság.

Az (56) és (57) összefüggések alapján a 4.12. és a 4.13. ábrán mutatott példák keréksebesség-idő diagramjai a jobb oldali kerékre a 4.21. ábrán, a bal oldali kerékre pedig a 4.22. ábrán láthatók.



4.21. ábra: Trajektóriatervezés irányfordítás nélkül – keréksebesség-idő diagramja



4.22. ábra: Trajektóriatervezés irányfordítással – keréksebesség-idő diagramja

2. TÉZIS: Kidolgoztam egy olyan új trajektóriatervező megoldást, amely egyszerre két módszerrel is (Hermite-görbe és Bezier-görbe) meghatározza egy két szállítószalaggal rendelkező vezető nélküli szállítótargonca hajtott kerekeinek sebességét a pályatervező pályapontok geometriai és idő adatai alapján.

Jelen tézis tartalma az [S1] Scopus által indexált Q2-es minőségű folyóiratban került ismertetésre.

4.3. Targonca hajtómotor-tápfeszültségeinek vezérlése a hajtott keréksebességek alapján

Az AGV mozgásvezérlő rendszernek a 4.2. fejezetben ismertetett **b**. modulja hajtott keréksebességi kimeneteket ad, amelyek bemenetként szolgálnak a **c**. modulhoz.

A $v_{L_{input}}$ és $v_{R_{input}}$ sebességek a bal és a jobb kerék középpontjához tartoznak. A sebességből feszültségre történő konvertáláshoz egy egyenáramú motor dinamikus modelljét

alkalmazom. A leíró differenciál-egyenletrendszer különböző formákban található meg a szakirodalomban [82]-[84]. Az AGV vezérléséhez az alábbi elektromechanikai egyenletrendszer kerül alkalmazásra:

$$U_M - R \cdot i_M - k \cdot \phi \cdot \omega_M = L \cdot \frac{di_M}{dt}, \tag{58}$$

$$k \cdot \phi \cdot i_M - f_M \cdot \omega_M - M_g = J \cdot \frac{d\omega_M}{dt}. \tag{59}$$

Az AGV haladását a továbbiakban egyenletes sebesség jellemzi.

Állandó sebesség feltételezése mellett $\frac{d\omega_M}{dt}$, $\frac{di_M}{dt}$ egyaránt 0 lesz. Ezzel a feltételezéssel az egyenáramú motor dinamikus modelljének elektromos egyenlete egyszerűsödik:

$$U_M - R \cdot i_M - k \cdot \phi \cdot \omega_M = 0. \tag{60}$$

A bal és a jobb kerekeket meghajtó motorokra vonatkozó egyenletek alapja megegyezik, ezért itt csak a bal kerékre vonatkozóan részletezem:

$$U_L = R \cdot i_M + k \cdot \phi \cdot \frac{v_{L_{input}}}{r_W \cdot k_H}, \tag{61}$$

$$k \cdot \phi \cdot i_M - f_M \cdot \omega_M - M_g = 0. \tag{62}$$

A (62) egyenletből az áramerősség:

$$i_M = \frac{f_M \cdot \omega_M + M_g}{k \cdot \phi}\ [A]. \tag{63}$$

Visszahelyettesítve a (61) egyenletbe, és konstans és nem konstans részekre rendezve a motort vezérlő feszültség:

$$U_L = v_{L_{input}} \cdot \left(\frac{R \cdot f_M}{k \cdot \phi \cdot r_W \cdot k_H} + \frac{k \cdot \phi}{r_W \cdot k_H}\right) + \frac{R \cdot M_g}{k \cdot \phi}\ [V]. \tag{64}$$

Értelemszerűen a jobb oldali motor U_R feszültségére a $v_{R_{input}}$ sebességet kell behelyettesíteni.

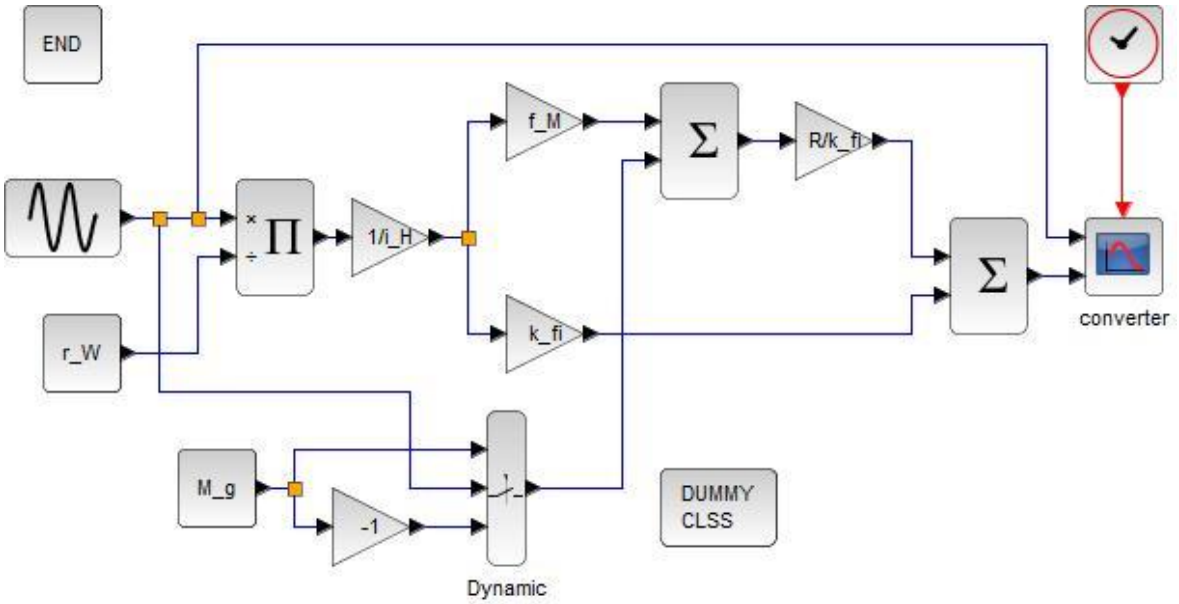
Szimulációs modul

A szimulációt a Scilab szoftverrendszer Xcos grafikus programozási részén terveztem meg és a 2. táblázat adatait alkalmaztam a motorkatalógus szerint [85].

2. táblázat: A sebesség-feszültség konverter szimulációs paraméterei

$\boldsymbol{U_N}$ $[V]$	$\boldsymbol{R}$ $[\Omega]$	$\boldsymbol{k \cdot \phi}$ $\left[\frac{V \cdot s}{rad}\right]$	$\boldsymbol{M_g}$ $[Nm]$	$\boldsymbol{f_M}$ $[Nms]$	$\boldsymbol{k_H}$ $[-]$	$\boldsymbol{r_W}$ $[m]$
24	$\leq 0{,}7$	$1/20$	0.26464	$3{,}3215 \cdot 10^{-5}$	1:25	$\frac{d_W}{2} = 70mm = 0.07m$

A sebesség-feszültség átalakítóból készült Xcos szimulációs program a 4.23. ábrán látható.

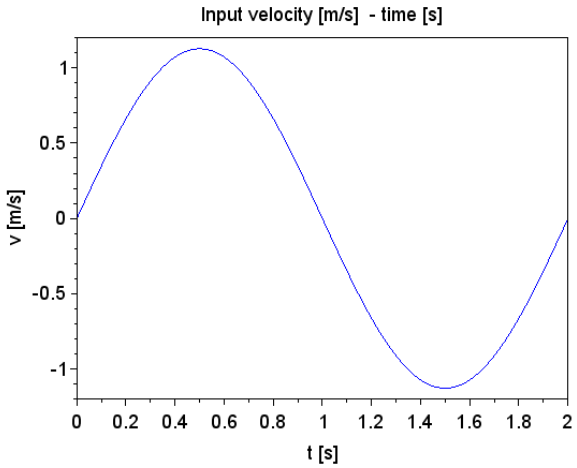


4.23. ábra: Sebesség-feszültség átalakító Xcos szimulációs programja

Példa és eredmények

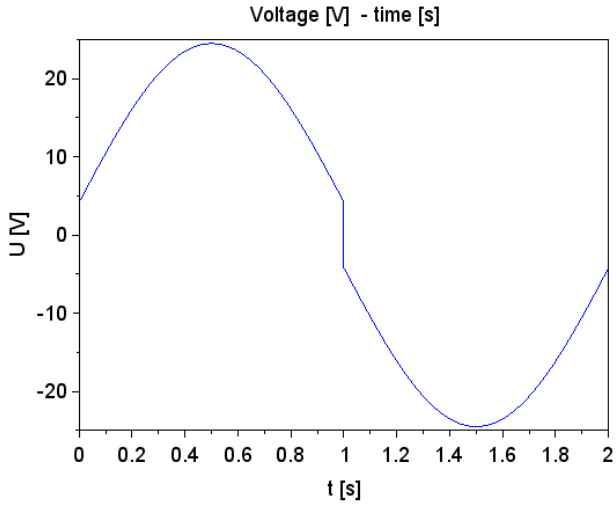
Teszt feladatként a 2. táblázatban megadott $U_N = 24V$ kapocsfeszültség értékre a (62)-(63) egyenletrendszerre alapozva készült a szimuláció. Az egyenletrendszert az áramerősségre és a sebességre megoldva: $i_M = 5.55996A, v_{input} = 1.12605\,\frac{m}{s}$.

Ebben a feladatban lassan változó sebességet írtam elő egy szinusz függvénnyel, megtartva azt a feltételezést, hogy az áramerősség és szögsebesség idő szerinti deriváltjai elhanyagolhatóan kicsi. A szimulációt a 4.24. ábrán látható szinusz generátor bemenettel végeztem el. Az időlépés 0.001s volt, és a végső szimulációs idő 2s volt. A szinusz hullámban a sebesség értéke váltakozik pozitív és negatív irányban, maximálisan a korábban kiszámított v_{input} sebességig. A szinusz hullám periódusideje a szimulációs idő, azaz 2s. Az eredményeket a 4.25. ábra mutatja.



4.24. ábra: Szinusz generátor bemenet a sebesség-feszültség átalakító szimulációjához

Megállapítható, hogy a feszültséggörbében egy ugrás van. Ennek az az oka, hogy a motor egy minimális feszültség alatt nem forog, mert a kerék és a talaj között a gördülési ellenállás nem engedi. Csak akkor forog a motor, ha a nyomatéka ezt az ellenállást leküzdötte.



4.25. ábra: "Feszültség – idő" görbe a sebesség-feszültség átalakítóból

4.4. Targonca vezérlés „DC motor dinamikai modell” része

Az AGV mozgásvezérlő rendszer **c**. modulja feszültségkimeneteket ad, amelyek bemenetként szolgálnak a **d**. modulhoz. Állandó M_t terhelési nyomaték feltételezése mellett a motor az ellenkező irányba kezd el forogni az indulásnál, mert a motor hajtónyomatéka a szimuláció szerint a kezdeti időben nulla. Valójában a M_t terhelés nem konstans.

4.4.1. A terhelőnyomaték meghatározása

Egy jármű motorjának jellemzően a motor belső súrlódását, a hajtóműben ébredő súrlódást, és a gördülésből származó ellenállást is le kell győznie, emelkedő esetén még az emelkedési ellenállás is megjelenik, az esetleges légellenállás a targonca esetén nem releváns.
Az M_t terhelőnyomaték jelen esetben két részből tevődik össze:

$$M_t = M_s + M_g \text{ [Nm]}, \tag{65}$$

ahol M_s a terhelési nyomaték fordulatszámtól függő része, amely a kinematikai lánc belső súrlódásából adódik, míg M_g a terhelési nyomaték a fordulatszámtól független része, amelyet a kerekek és a talaj közötti gördülési ellenállásból származik.

4.4.2. Összefüggések a motor belsejében lévő súrlódásból adódó terhelési nyomatékra

A motor belső súrlódásából adódó nyomaték a motorkatalógus szerint [85]:

$$M_s(\omega) = f_M \cdot \omega_M \leq 0{,}0160 \text{ Nm}. \tag{66}$$

Ha a motor szögsebessége megegyezik a névleges értékével $\omega_M = \omega_{M,n}$, akkor a súrlódási együttható az M_s névleges értékéből:

$$f_{M,n} = \frac{M_{s,n}}{\omega_{M,n}} = \frac{M_{s,n}}{\frac{n_{M,n}}{60} 2\pi} = 3{,}3215 \cdot 10^{-5}\ Nms. \tag{67}$$

ahol $n_{M,n} = 4600 \frac{1}{min}$.

4.4.3. Összefüggések hajtókerék-talaj közötti gördülési ellenállásból adódó terhelési nyomatékra

Az AGV kerekek gördülési ellenállásból adódó terhelési nyomatéka:

$$M_g = k_H \cdot r_W \cdot F_g \text{ [Nm]}, \tag{68}$$

ahol a k_H érték a 2. táblázatban a motor és a kerék közötti hajtómű hajtóviszonya, illetve az r_W kerék sugara rendelkezésre áll. A F_g értéke mérésekből határozható meg, amelyet a 4.4.4. alfejezet részletesen tárgyal.

4.4.4. Vezető nélküli targonca gördülési ellenállásának vizsgálata

Egy autonóm jármű modellezéséhez és vezérléséhez szükséges a jármű gördülési ellenállásának ismerete [86], [88], [89]. Jelen alfejezet a vezető nélküli targonca gördülési ellenállásának vizsgálatával foglalkozik. A vizsgálatok során felszabadításra került a kinematikai lánc, hogy meghatározható legyen tisztán a gördülésből származó ellenállás értéke. Így a járműre ható haladás irányú húzóerő vízszintes pályán közvetlenül szolgáltatja a gördülési ellenállást. A vizsgálatok megtörténtek a jármű ferde lejtőn való mozgatással is.

Vizsgálat esetei

Az első esetben a mozgatás vízszintes talajon történt, ahogy az a 4.26. ábra mutatja.
Ebben az esetben a húzóerő megegyezik a kerekeken ébredő gördülési ellenállásokból adódó ellenerők összegével, azaz közvetlenül szolgáltatja a gördülési ellenállást [87]:

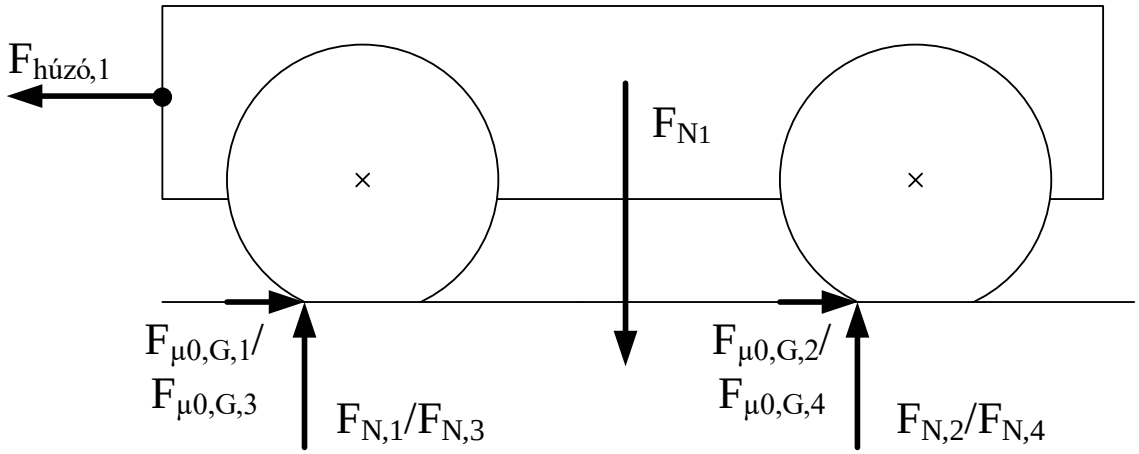
$$F_{\mu_{0,G_1}} = \sum_{W=1}^{4} F_{\mu_{0,G,W}} \cong F_{húzó,1} \text{ [N]}. \tag{69}$$

A rendszeren ébredő normálerő a következőképpen írható fel:

$$F_{N_1} = \sum_{W=1}^{4} F_{N,W} = m_{AGV} \cdot g \text{ [N]}. \tag{70}$$

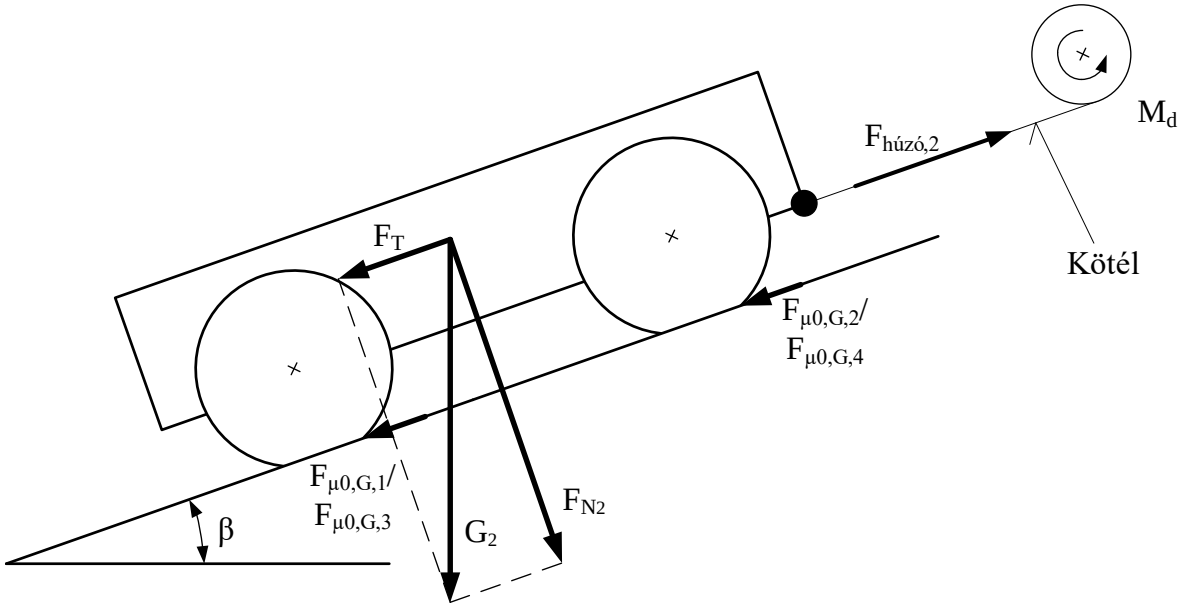
Rendszeren elinduláskor fellépő gördülési ellenállási együttható így már kiszámítható:

$$\mu_{0,G_1} = \frac{F_{\mu_{0,G_1}}}{F_{N_1}} = \frac{F_{húzó_1}}{F_{N_1}} \text{ [}-\text{]}. \tag{71}$$



4.26. ábra: Vezető nélküli targonca gördülési ellenállásának vizsgálata – 1. eset

A másik eset már egy ferde pályán történik, ahogy a 4.27. ábra részletezi. Az összefüggéseknél a lejtő szögét (β) is figyelembe kell venni. A valóságban a kerék ennyire nem nyomódik be, az ábrán a gördülésből származó ellenerő ébredési helyének szemléltetése kedvéért látható ilyen mértékű benyomódás.



4.27. ábra: Vezető nélküli targonca gördülési ellenállásának vizsgálata – 2. eset

Mozgás irányába eső tangenciális erőkomponens:

$$F_T = m_{AGV} \cdot g \cdot \sin(\beta) \text{ [N]}. \tag{72}$$

A rendszeren ébredő gördülési ellenállásból származó ellenerő:

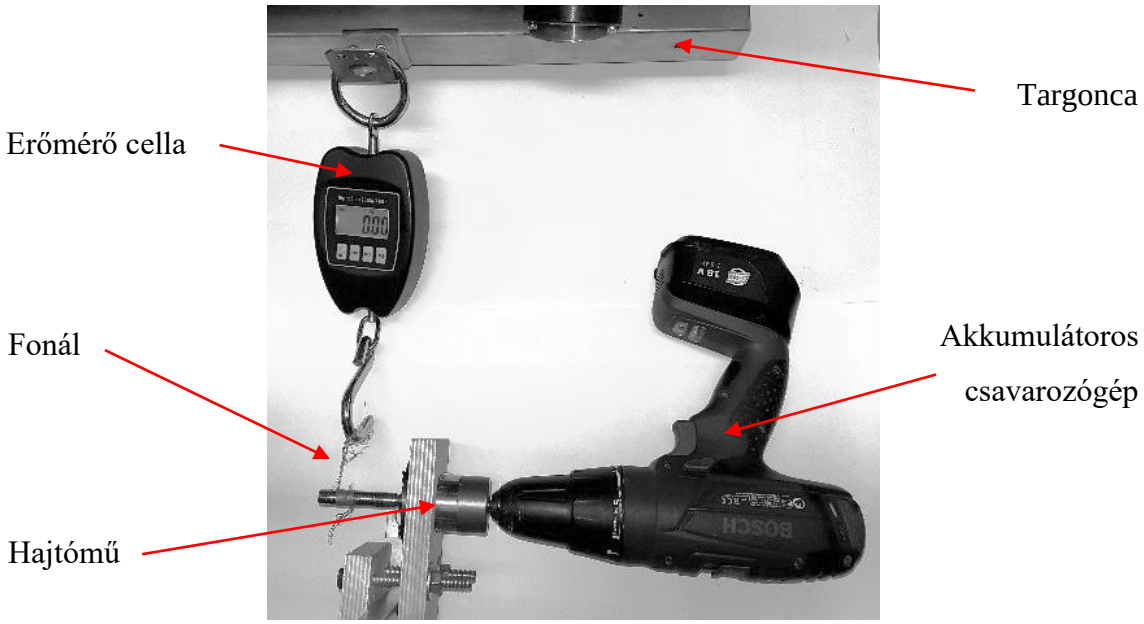
$$F_{\mu_{0,G_2}} = F_{húzó_2} - F_T \text{ [N]}. \tag{73}$$

A rendszeren elinduláskor fellépő megindulási gördülési ellenállási együttható:

$$\mu_{0,G_2} = \frac{F_{\mu_{0,G_2}}}{F_{N_2}} = \frac{F_{\mu_{0,G_2}}}{m_{AGV} \cdot g \cdot \cos(\beta)} \text{ [}-\text{]}. \tag{74}$$

Gördülési ellenállás méréséhez használt mérési elrendezés

A húzóerőt egy akkumulátoros csavarozógép által meghajtott nagy áttételű hajtómű szolgáltatja. A mozgást a hajtómű $d_{tengely} = 10mm$ átmérőjű tengelyére rögzített és felcsévélt fonál biztosítja. A ferde pályán történő mozgatás kiértékelése a (80) képlet alapján történik. Ez utóbbi módszer azért előnyös, mert a vízszintes mozgatásnál a megindításkor hirtelen leesik a húzóerő értéke és az erő oszcilláló képet mutat. A lejtőn pedig a mozgás indítási szakaszában is a húzóerő értéke relatíve stabil marad. A mérési összeállítás a 4.28. ábrán látható.



4.28. ábra: Gördülési ellenállás vizsgálatának összeállítása

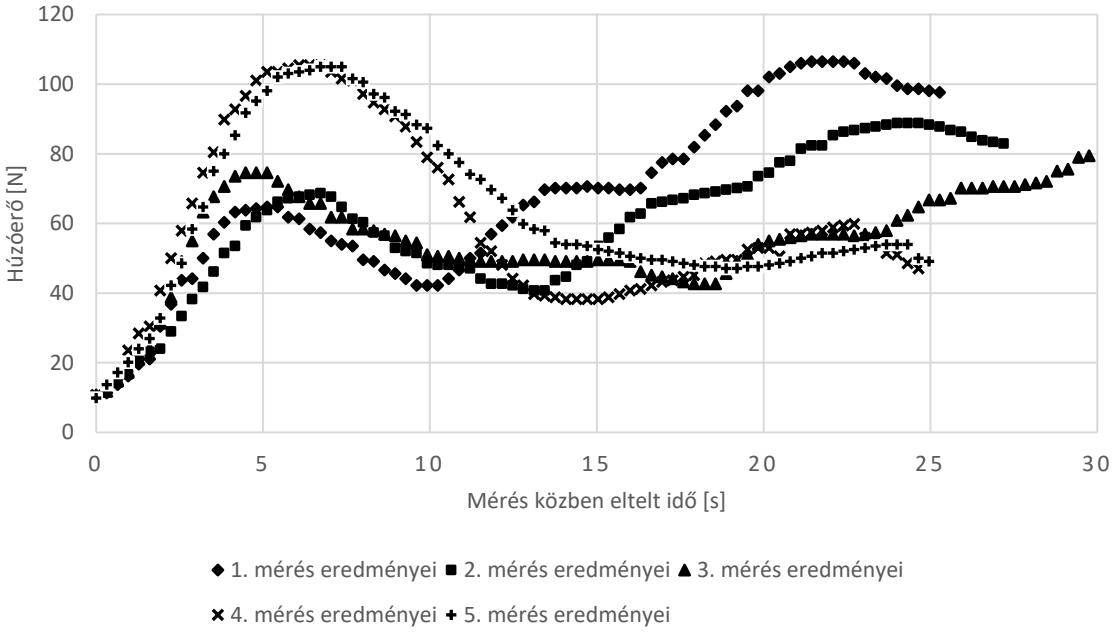
Az erőmérő cella (Adventuridge EHS-699) a fonálban ébredő erőt kilogramm súlyban méri tizedes pontossággal. Az akkumulátoros csavarozógép (Bosch PSB 18 Li-2) névleges fordulatszáma: $n_{PSB} = 440\,\mathrm{min}^{-1}$. Az alkalmazott 4 fokozatú bolygóhajtómű (Faulhaber 38/1) áttétele: $i_{Faulhaber} = 134{:}\,1$.

A vontatott targonca sebessége:

$$v_{targonca} = \frac{d_{tengely}}{2} \cdot \omega_{tengely} = \frac{d_{tengely}}{2} \cdot \frac{n_{PSB}}{60} \cdot \frac{2\pi}{i_{\mathrm{Faulhaber}}} = \frac{10mm}{2} \cdot \frac{440}{60} s^{-1} \cdot \frac{2\pi}{134} = 1{,}72\,\frac{mm}{s}. \quad (75)$$

Gördülési ellenállás vizsgálata vízszintes talajon, mérési eredmények

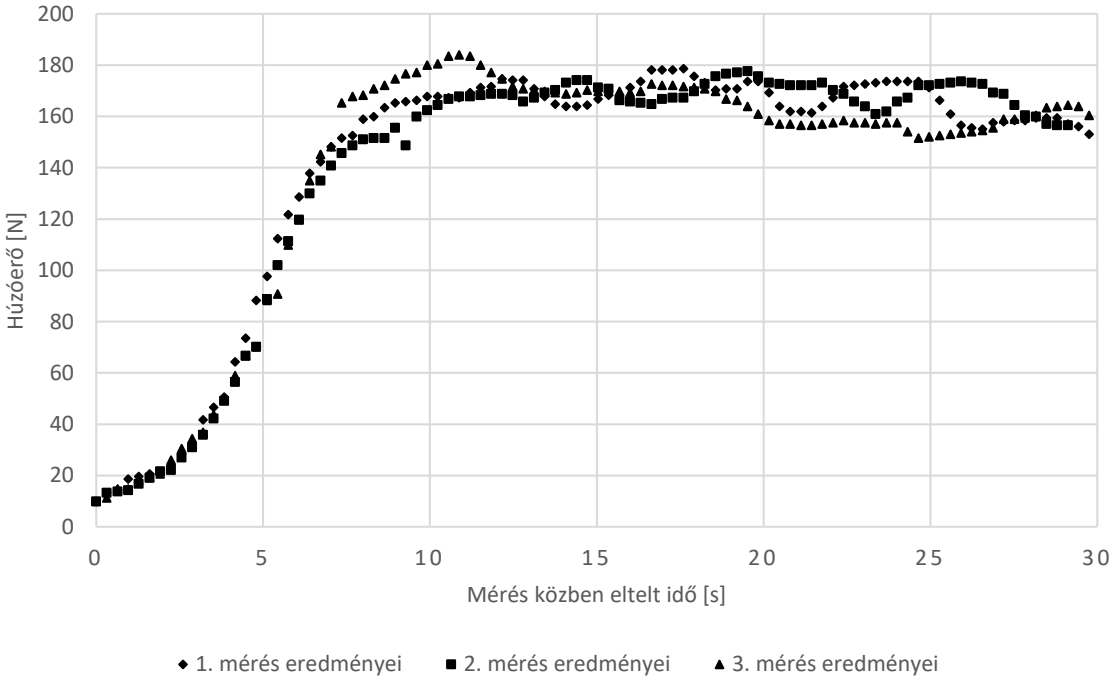
Ez a mérés a targonca vízszintes talajon való mozgatásakor fellépő gördülési ellenállását vizsgálja. A mért „húzóerő–idő” diagramot a 4.29. ábra szemlélteti. A bemutatott mérési sorozatból jól látható, hogy a kezdeti megindulás után a vonóerő lecsökkent, és csökkenő amplitúdójú lengést mutat. Ez a jelenség nehezen kiértékelhetővé teszi az állandónak feltételezett gördülési ellenállást.



4.29. ábra: Gördülési ellenállás vízszintes talajon történő vizsgálatának mérési eredményei

Gördülési ellenállás vizsgálata ferde pályán, mérési eredmények

Ennél a mérésnél a targoncát ferde pályán felfelé egyenes vonalon vontatva mérjük az AGV mozgatásához szükséges húzóerőt. Az emelkedőre való helyezés által nagyobb húzóerőre van szükség, ezt a többletet a lejtő emelkedési szögéből és az AGV súlyerejéből számítható. A mért húzóerő–idő diagramot a 4.30. ábra illusztrálja.



4.30. ábra: Gördülési ellenállás ferde pályán történő vizsgálatának mérési eredményei

A 4.30. ábrán bemutatott mérési sorozat alapján megállapítható, hogy a húzóerő ebben az esetben már relatíve egyenletes értéket mutat, a lengések amplitúdója körülbelül a felére csökkent a vízszintes pályához képest.

A gördülési ellenállás meghatározásának céljából le kell vonni az emelkedő szöge által megnövekedett húzóerőtöbblet. A lejtő szöge egyszerű trigonometrikus összefüggéssel kiszámítható a rámpán mért magassági és hosszúsági értékek alapján:

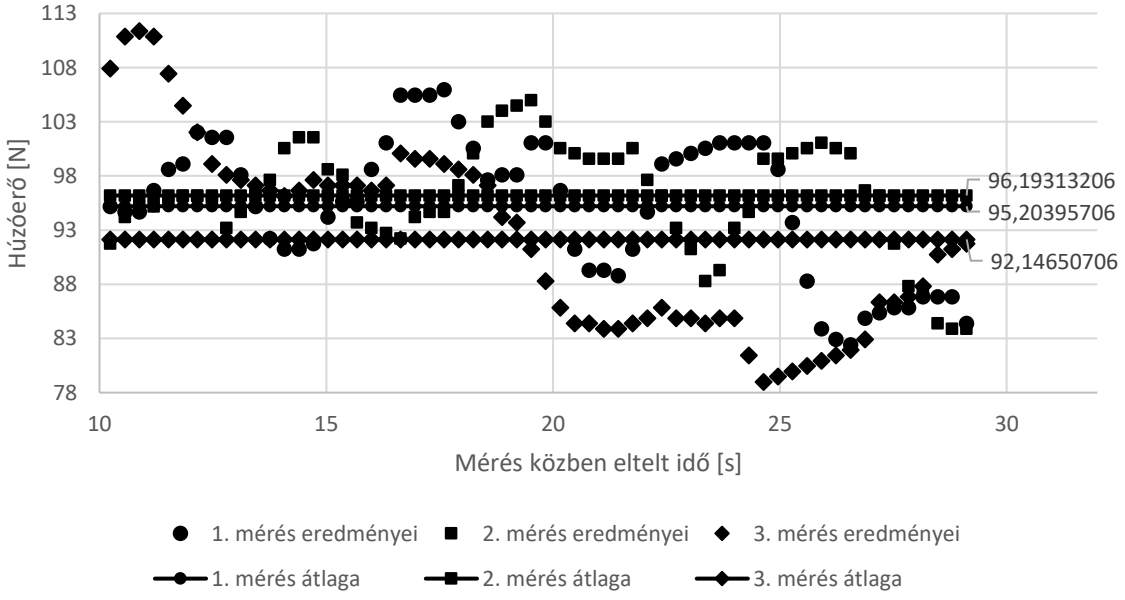
$$\beta = arctg\left(\frac{32mm}{951mm}\right) = 1{,}9283°. \tag{76}$$

A súlyerőnek a lejtő irányú vetülete, vagyis a húzóerőtöbblet kifejezhető:

$$F_T = m_{AGV} \cdot g \cdot \sin(\beta) = 220kg \cdot 9{,}81\frac{m}{s^2} \cdot \sin(1{,}9283°) = 72{,}62082N. \tag{77}$$

A többleterővel csökkentett erőértékek a 4.31. ábrán láthatók, kiemelve a felfutás utáni szakaszt, és az ábrázolt tartományon kiszámított átlag erőértékek is leolvashatók. A három mérés átlagából kiszámítható a húzóerő, azaz a gördülési ellenállási erő:

$$F_{\mu_{0,G_2}} = F_{húzó_2} - F_T = 94{,}51453\ N \tag{78}$$



4.31. ábra: Gördülési ellenállás emelkedőn történő vizsgálatának felfutás utáni eredményei, tangenciális erőt kivonva, átlagokkal

A gördülési ellenállási együttható a mérések alapján kiszámítható:

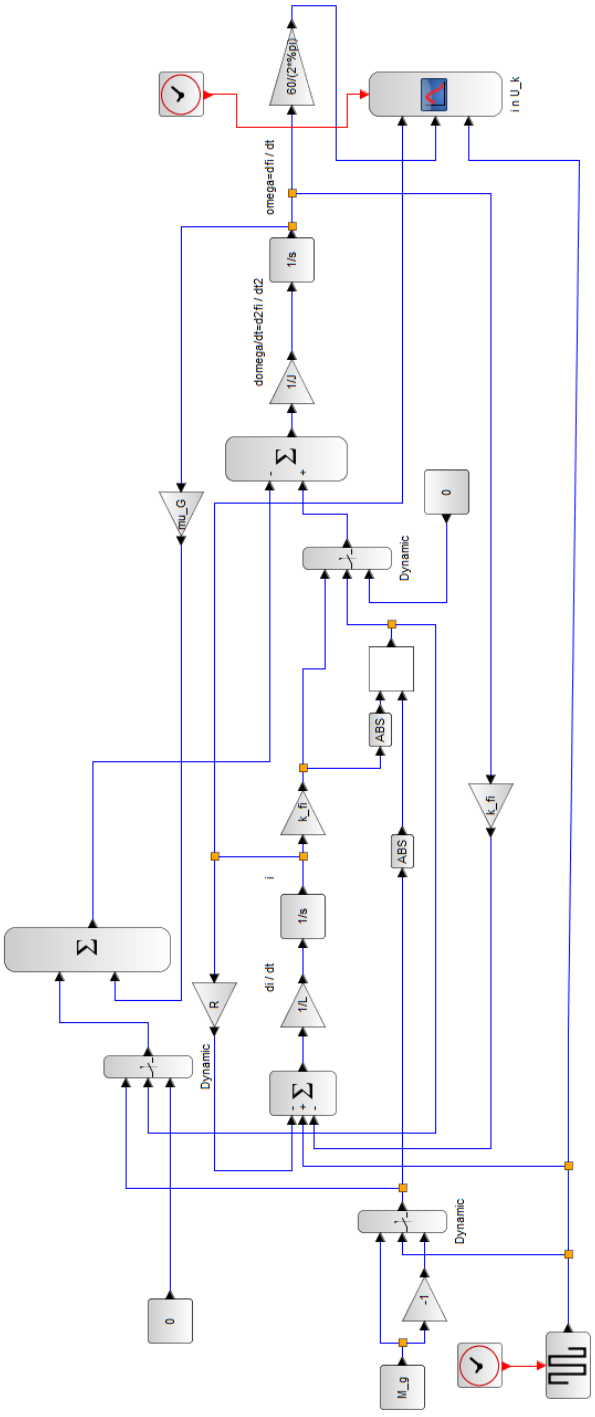
$$\mu_{0,G_2} = \frac{F_{\mu_{0,G_2}}}{F_{N_2}} = \frac{F_{\mu_{0,G_2}}}{m_{AGV} \cdot g \cdot \cos(\beta)} = \frac{94{,}51453\ N}{220kg \cdot 9{,}81\frac{m}{s^2} \cdot \cos(1{,}9283°)} = 0{,}0438. \tag{79}$$

A (78) összefüggésben meghatározott gördülési ellenerő ismeretében most már kiszámítható a motor által legyőzendő gördülési ellennyomaték:

$$M_g = k_H \cdot r_W \cdot F_{\mu_{0,G_2}} = \frac{1}{25} \cdot 0{,}07\text{m} \cdot 94{,}51453\ N = 0.26464Nm. \tag{80}$$

4.4.5. AGV elektromechanikai modellje és szimulációs modulja

Ebben a pontban a targonca mozgásáról feltételezzük, hogy a pályája egyenes, és a két hajtómotor azonos nyomatékot ad le. Ez azt jelenti, hogy elegendő a targonca fél modelljét felírni. Az AGV DC motor (58) és (59) összefüggéseibe beépítésre kerülnek a 4.4.2-4.4.4. fejezetekben kiszámított M_g és f_M paraméterek, de a fél modell miatt a gördülési ellennyomatéknak csak a fele hat.



4.32. ábra: Kiszámított terhelőnyomatékot alkalmazó elektrodinamikai modellt szimuláló Xcos program

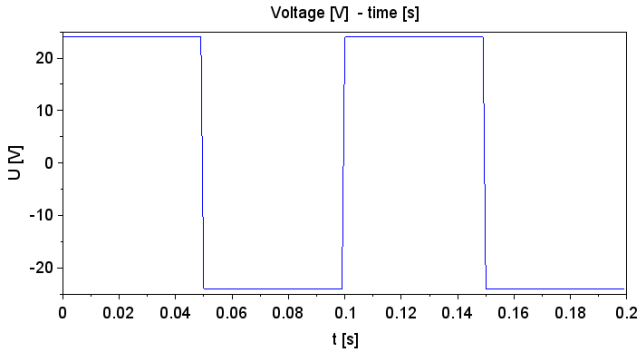
Ha a hajtónyomaték kisebb, mint a terhelőnyomaték, a motor álló marad, ezért, ha $|M_d| = |k \cdot \phi \cdot i_M| < \left|\frac{M_g}{2}\right|$ azaz $\omega_M = 0$, akkor:

$$U_k - R \cdot i_M = L \cdot \frac{di}{dt}. \tag{81}$$

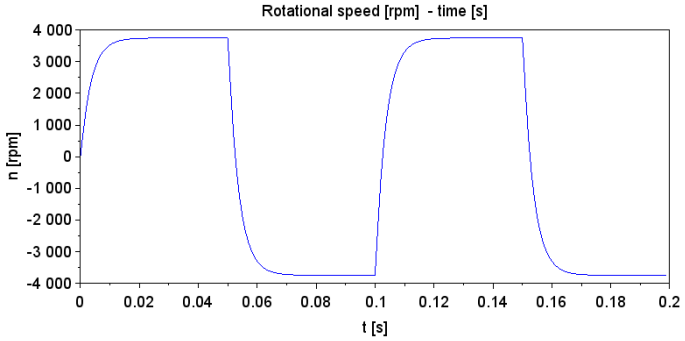
A szimuláció Xcos programját a 4.32. ábra illusztrálja. A feltételek teljesítésére kapcsolóblokkokat helyeztem el. Ezenkívül az abszolút értékre vonatkozó "ABS" blokkot és más blokkokat is beépítettem a kiszámított terhelőnyomatékot alkalmazó modell szimulálásához.

A szimulációs modul tesztelésére a feszültség bemenetet tekintve négyszögjel-generátorral került definiálásra (lásd 4.33. ábra). Az eredmények kiértékelése 0,001 másodperces időlépésenként történt. A szimuláció időtartama 0,2s. A U_M feszültség négyszögjel-generátor értéke az U_N névleges feszültség pozitív és negatív értéke között váltakozhat. A négyszögjel-generátor ciklusideje a szimulációs idő fele, azaz 0,1s. A fordulatszám és a motor áramerősségének eredményei a 4.34. és a 4.35. ábrán láthatók.

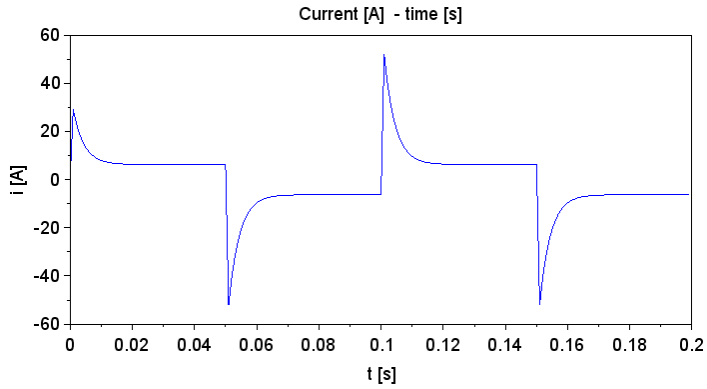
A szimuláció a modell helyes működését írja le, az alkalmazott terhelőnyomaték miatt a motor a megfelelő irányba és a megfelelő áramerősséggel kezd el forogni az állandónak feltételezett terhelőnyomatékhoz képest.



4.33. ábra: Négyszögjel bemenet az egyenáramú motor elektrodinamikai modelljének szimulációjához



4.34. ábra: "Fordulatszám – idő" diagram az egyenáramú motor elektrodinamikai modelljének szimulációja után

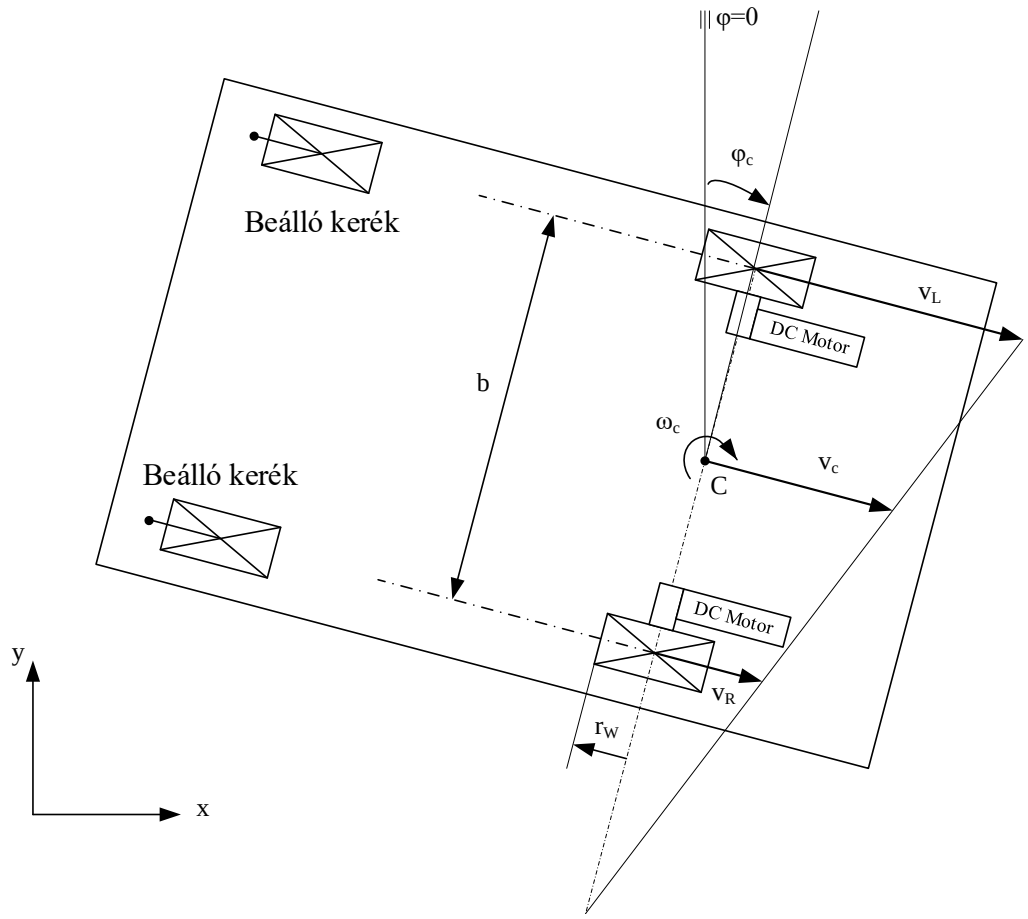


4.35. ábra: "Áramerősség – idő" diagram az egyenáramú motor elektrodinamikai modelljének szimulációja után

4.5. AGV útvonal szimulációja

A 4.4. fejezetben részletezett AGV mozgásvezérlés **d**. moduljának szögsebesség kimenetei az **e**. modul bemenetére csatlakoznak. Az egyenáramú motorok $\omega_{L_{motor}}$ és $\omega_{R_{motor}}$ szögsebességeinek felhasználásával ki lehet számítani a kerekek sebességét és az AGV pályáját is.

A vezető nélküli targonca felülnézetben ábrázolt kinematikai vázlata a 4.36. ábrán látható.



4.36. ábra: Az AGV kinematikai modellje

Az egyes kerekek szögsebességei az egyenáramú motorok szögsebességei alapján számíthatók ki a motor és kerék közötti hajtőmű hajtóviszonyának ismeretében, feltételezve, hogy a kerekek nem csúsznak meg:

$$\omega_L = \omega_{L_{motor}} \cdot k_H \left[\frac{\text{rad}}{\text{s}}\right], \qquad \omega_R = \omega_{R_{motor}} \cdot k_H \left[\frac{\text{rad}}{\text{s}}\right]. \tag{82}$$

Az AGV kerekek közötti C felezőpontjának sebessége:

$$v_c = \frac{v_{L,modul}+v_{R,modul}}{2} = \frac{r_W \cdot \omega_L + r_W \cdot \omega_R}{2} \left[\frac{\text{m}}{\text{s}}\right]. \tag{83}$$

A C felezőponton átmenő függőleges tengely körüli szögsebessége:

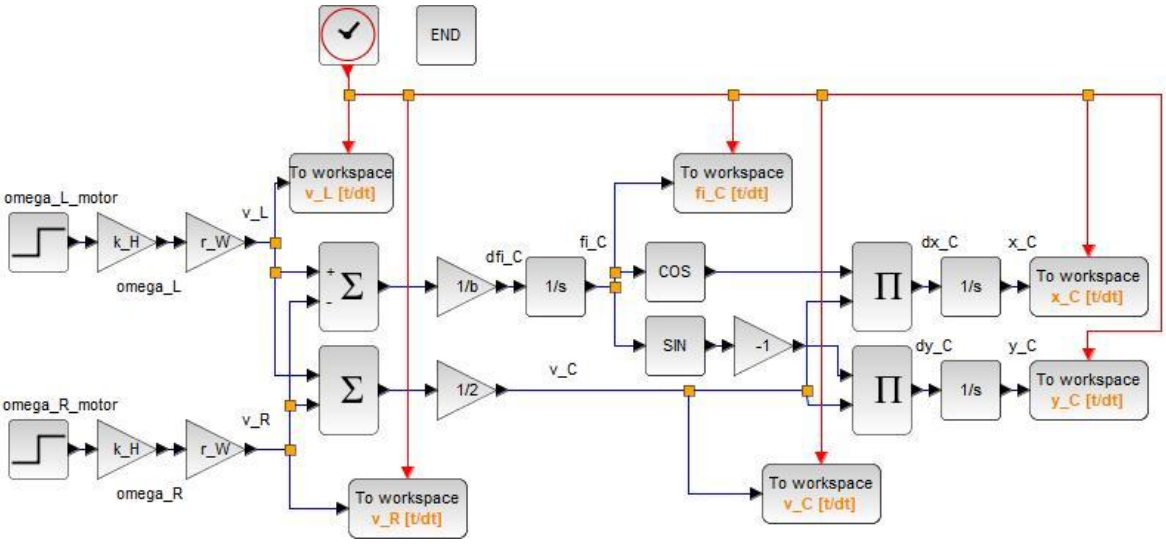
$$\omega_c = \frac{v_{L,modul}-v_{R,modul}}{b} = \frac{r_W \cdot \omega_L - r_W \cdot \omega_R}{b} \left[\frac{\text{rad}}{\text{s}}\right]. \tag{84}$$

A C pont pályájának szimulációjához használt differenciálegyenlet-rendszer:

A v_c sebesség X és Y vetületi koordinátáit és a ω_c szögsebességet az $\dot{\boldsymbol{x}}$ oszlopvektorba rendezve egy egyenletrendszert kapunk:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\varphi}_c \end{bmatrix} = \begin{bmatrix} \frac{r_W \cdot \omega_L + r_W \cdot \omega_R}{2} \cdot (cos\ \varphi_c) \\ \frac{r_W \cdot \omega_L + r_W \cdot \omega_R}{2} \cdot (-sin\ \varphi_c) \\ \frac{r_W \cdot \omega_L - r_W \cdot \omega_R}{b} \end{bmatrix}. \tag{85}$$

A (85) egyenletrendszer alapján a C pont pályájának kiszámítására készült Xcos szimulációs programját a 4.37. ábra illusztrálja. Az itt tárgyalt esetben a bemenetek egységugrás függvény jellegűek (rámpafüggvény). Ez a rámpafüggvény mindkét kerék bemeneti szögsebességét változtatja 1 másodperc elteltével egy előírt értékre módosítja.

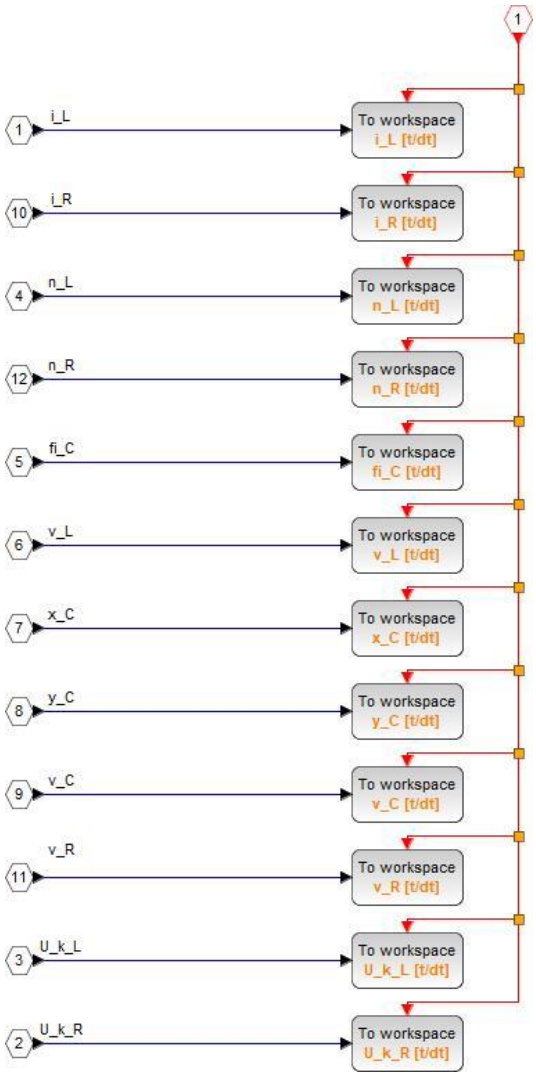


4.37. ábra: AGV pályaszimuláció Xcos szimulációs programja

4.6. Kommunikáció a szövegesen és a grafikusan programozott egységek között

Az adattovábbítás valósítható meg a Scilab alatt az Xcos grafikus- és SciNotes alatt megírt modulok között. Az **f**. modul összegyűjti az i_L, az i_R, az n_L, az n_R, az U_L és az U_R értékeket

a vezérlés **d**. moduljából, míg a φ_C, $v_{L,modul}$, $v_{R,modul}$, x_C, y_C és v_C értékeket az **e**. modulból fogadja, majd a szoftver globális változóit feltölti. A szövegesen programozott főprogram ezután már a globális változókkal végezheti a műveleteket. Az adattovábbító modul a 4.38. ábrán látható.

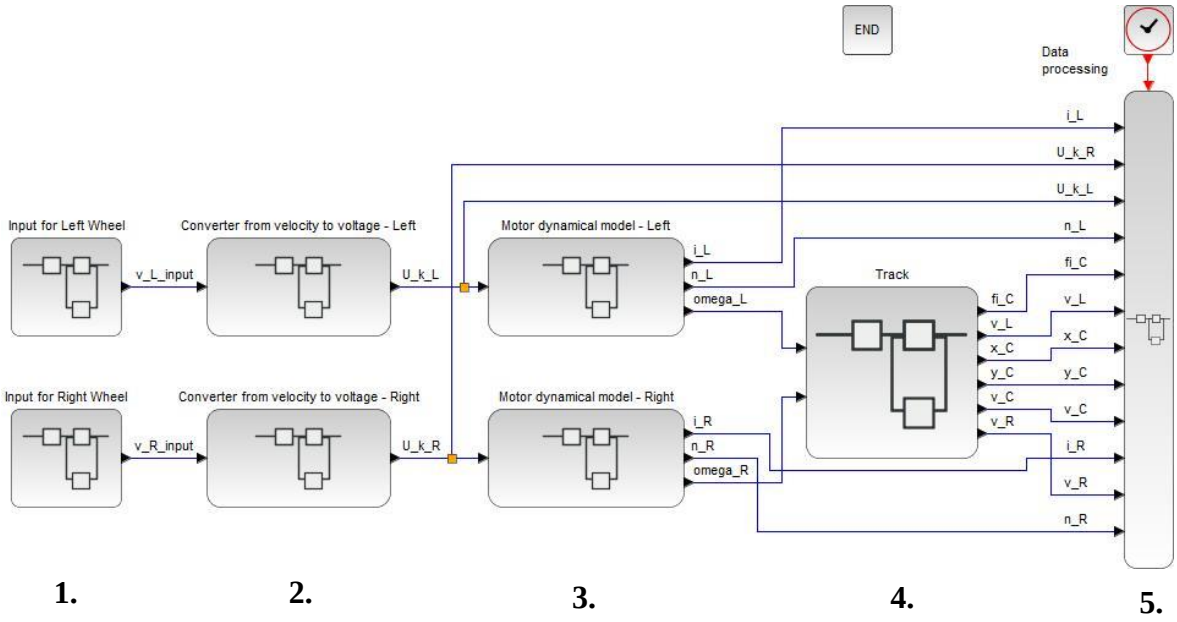


4.38. ábra: Adattovábbítás Xcos szimulációs programja

4.7. Vezérlés 4 grafikusan programozott moduljainak tesztelése három példával

Az AGV vezérlés, 4.3-4.6. alfejezetekben ismertetett **c**-**f**. moduljai grafikus úton készültek. A szövegesen megírt **a**-**b**. modul futtatása után hívja a **c**-**f**. modulláncolatot, amelyet a 4.39. ábra mutat. Balról jobbra a szimulációs blokkok a következőképpen épülnek fel:

1. Bemeneti függvények
2. Átalakító sebességről feszültségre kerekenként (**c**. modul)
3. DC motor elektrodinamikai modellje kerekenként (**d**. modul)
4. Pálya szimulációja (**e**. modul)
5. Adattovábbítás a vezérlés már részei felé (**f**. modul)



4.39. ábra: Szimulációs program a vezérlés grafikusan programozott részéből

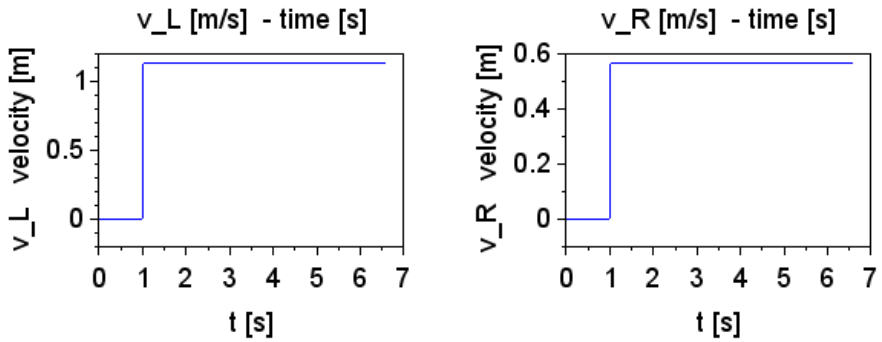
A **c**. modul bemenete normál esetben a trajektóriatervező (**b**. modul) sebesség kimenetei lenne. A tesztelés során a bemenetek három különböző esetei változnak az alábbiakban szerint.

Az első esetben az AGV egy kör mentén mozog, ezt a körmozgást az egyenáramú motorok két különböző csatlakoztatott feszültségével hajtja végre.

A 4.39. ábrán látható 1. pontjában a bemenetek jelen esetben a következők:

- $v_{L_{input}} = v_{R_{input}} = 0\,\frac{m}{s}$, **ha** $t < 1s$,
- $v_{L_{input}} = v_{input}\ \left[\frac{m}{s}\right]$ és $v_{R_{input}} = \frac{v_{input}}{2}\ \left[\frac{m}{s}\right]$, **ha** $t \geq 1s$.

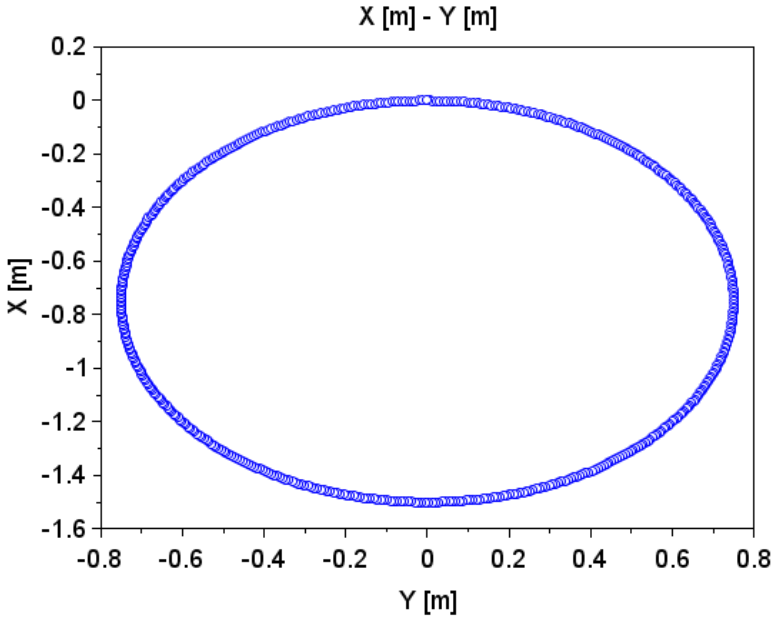
A valós v_L és v_R sebességek a 4.40. ábrán látható.



4.40. ábra: A kerekek valós sebessége az 1. esetben

A feszültségek eredményei nagyon hasonló tendenciát mutatnak a sebességhez képest, de az értékek a következőképpen különböznek: $U_L = 24.000001V$ és $U_R = 13.852480V$.

A jobb oldali kerék feszültsége nem a maximális érték fele sebesség-feszültség modulnál leírt körülmények miatt. A rajzolt pálya a 4.41. ábrán látható.

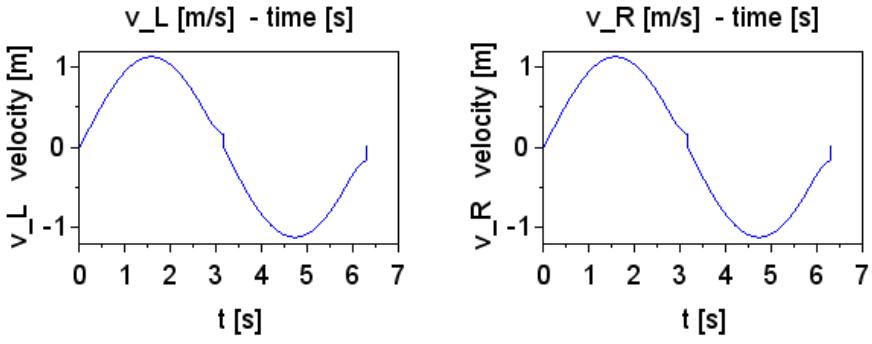


4.41. ábra: AGV szimulált pályája az 1. esetben

A második esetnél az AGV egyenes vonal mentén mozog, ugyanazzal a csatlakoztatott feszültséggel hatja végre az egyenáramú motorokkal, de a bemeneti sebességek egy szinusz hullám mentén változnak az alábbiak szerint:

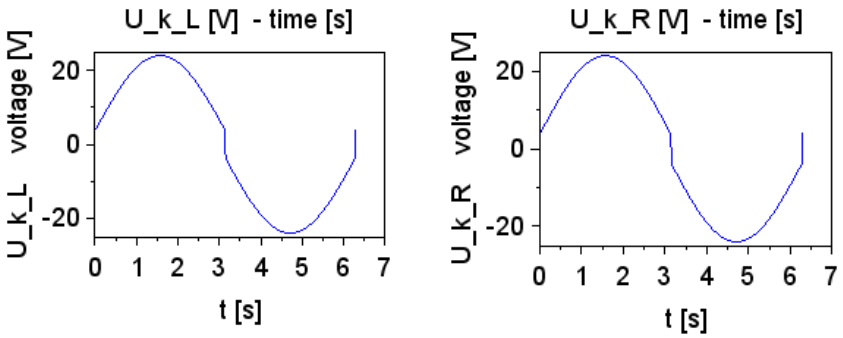
- $v_{L_{input}} = v_{R_{input}} = v_{input} \cdot \sin(t)$ $\left[\frac{m}{s}\right]$,

A valós v_L és v_R sebességeket a 4.42. ábra mutatja. Az egyenáramú motorok konvertereinek (**c**. modul) és elektrodinamikai modelljeinek (**d**. modul) köszönhetően a valós sebességek nem pontosan szinusz hullámok.



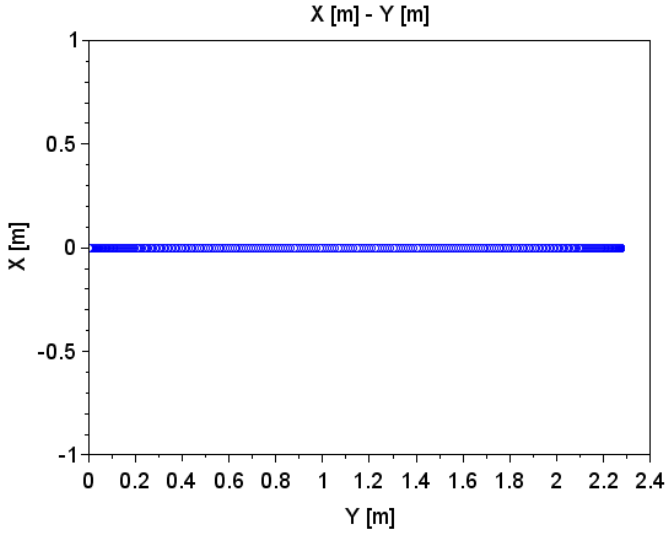
4.42. ábra: A kerekek valós sebessége a 2. esetben

A feszültségekre vonatkozó eredmények a sebességhez képest eltérő tendenciát mutatnak (lásd 4.43. ábra).



4.43. ábra: Az egyenáramú motorokhoz csatlakoztatott feszültségek a 2. esetben

A rajzolt pályát a 4.44.ábra mutatja.

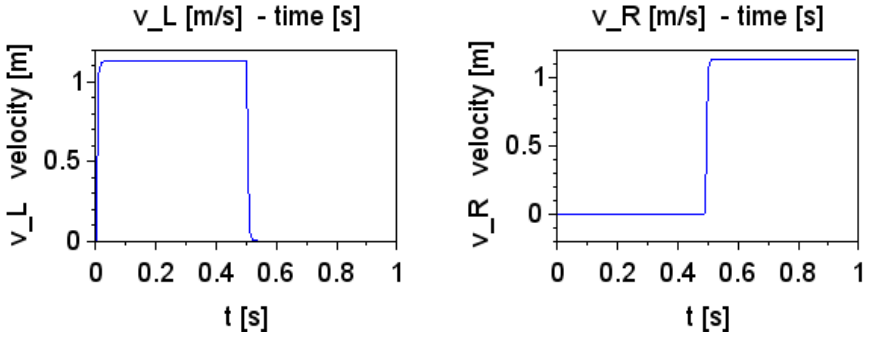


4.44. ábra: AGV szimulált pályája a 2. esetben

A harmadik esetben az AGV változó irányú ívek mentén mozog, az $t = 0s \div 1s$ intervallumok közötti sebesség-bemenetek végzik az alábbiak szerint:

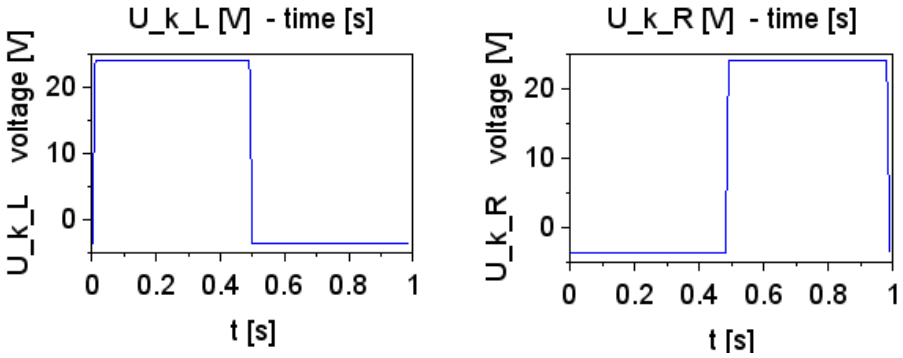
- $v_{L_{input}} = v_{input}\ \left[\frac{m}{s}\right]$, **ha** $t < 0.5s$, egyébként $v_{L_{input}} = 0\frac{m}{s}$,
- $v_{R_{input}} = v_{input}\ \left[\frac{m}{s}\right]$, **ha** $t \geq 0.5s$, egyébként $v_{R_{input}} = 0\frac{m}{s}$.

A valós v_L és v_R sebességeket a 4.45. ábra szemlélteti. A bemenetek váltakozó módon vannak csatlakoztatva.



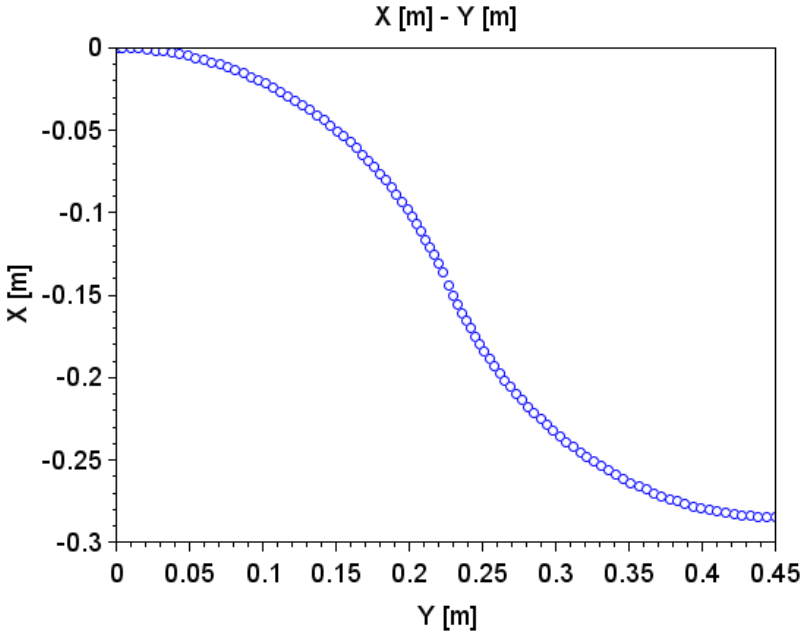
4.45. ábra: A kerekek valós sebessége a 3. esetben

A feszültségekre vonatkozó eredmények a sebességhez képest eltérő tendenciát mutatnak (lásd 4.46.ábra).



4.46. ábra: Az egyenáramú motorokhoz csatlakoztatott feszültségek a 3. esetben

A rajzolt pályát a 4.47. ábra illusztrálja.



4.47. ábra: AGV szimulált pályája a 3. esetben

3. TÉZIS: A trajektóriatervező kimeneteként adódó kerék-sebességadatok felhasználásával kialakítottam egy olyan új, moduláris rendszert, amely előállítja a pálya szimulációját és a hajtáshoz szükséges áramerősséget, valamint az adatokat (sebességek, fordulatszámok, feszültségek, áramerősségek, pozíciók és szögelfordulás) továbbít a főprogram felé. A DC motor elektrodinamikai modelljében a terhelőnyomaték értékét a targonca adottságaihoz illesztettem.

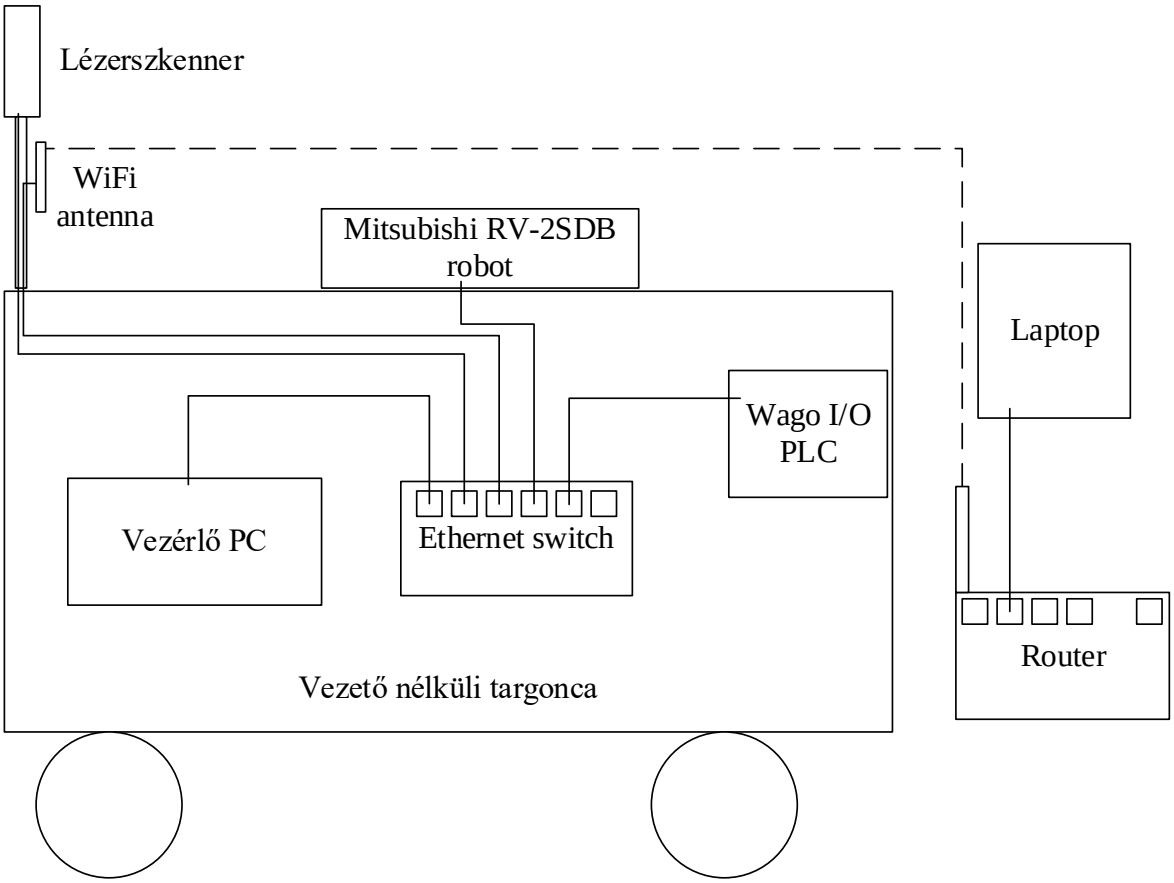
Vonatkozó publikációk: [S3], [S5], [S9]

5. AGV VEZÉRLÉSÉNEK PROGRAMFEJLESZTÉSE

Ez a fejezet a vezető nélküli targonca vezérlésének programozásával foglalkozik. A targonca hardveres komponenseit és hálózati eszközeit az 5.1. fejezet tartalmazza. Az 5.2. fejezet a targonca manuális mozgatását ismerteti, itt felhasználva a kezelőpaneljén található joystick-ot. Az 5.3. fejezet a targonca automatikus mozgás programozás lehetőségeit részletezi. Ez a biztonsági funkciókat is tartalmazza, például vészhelyzetnél a motorokra jutó áram azonnali megszakítását. Az automatikus mozgatás akadályok között egyenes vonalban alternáló módon történik az elülső-hátsó biztonsági érzékelők segítségével. Később továbbfejlesztésként az automatikus mozgatás már a navigáció adatait is felhasználja egyszerű mozgások megvalósítására. Majd ezután a 4. fejezetben ismertetett pálya- és trajektóriatervezést alkalmazó mozgásvezérlés beépítése következik.

5.1. Hálózati eszközök

A rendszerhez tartozó eszközök kapcsolódása és elhelyezkedése az 5.1. ábrán látható.



5.1. ábra: Vezető nélküli targonca hálózati topológiája

A targoncát távoli eléréssel lehet programozni, amely a VNC szoftver segítségével történik, a host a targoncában van, a kliens a külső PC-re van telepítve, a kapcsolat a router-en

keresztül valósul meg. A programozás során szükség van még a topológiában látható LIDAR szenzor és a WAGO PLC elérésére is.

A targonca mozgásának programozása a host-ra feltelepített Eclipse fejlesztőkörnyezettel valósítható meg. Ez a szoftver alapvetően Java programozási nyelv alapú, de a targonca programja C++ programozási nyelven íródott meg, és a futtatás során ezt a programot fordítja le.

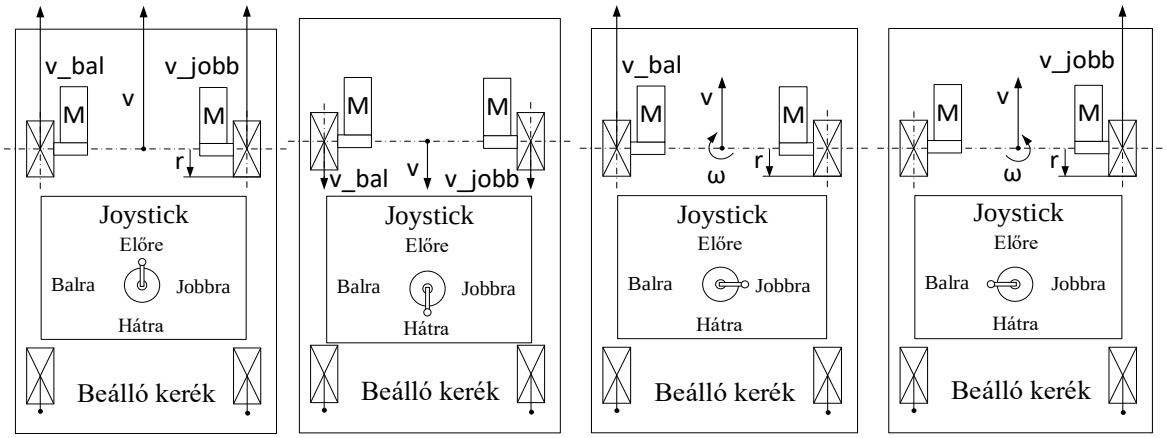
A programban rendelkezésemre állt a kommunikációhoz szükséges rész, amely kapcsolatot teremt PLC-vel és a LIDAR szenzorral. A programban van egy beépített AutoCAD-ben készített mintarajz, amely alapján az adott helyen a targonca automatikusan képes volt működni. Azonban a targonca a jelenlegi laboratóriumban ezt a programot nem alkalmazhatja a megváltozott geometriai viszonyok miatt. Az új helyszínhez illeszkedő programozáshoz az eredeti rendszert mélységében fel kellett dolgozni.

5.2. Manuális mozgatás

A targoncát manuálisan és automatikusan is lehet irányítani, jelen alfejezet az előbbit részletezi. Amint a 3. fejezetben leírásra került, a mozgatás két darab, oldalt elhelyezett egyenáramú motor által hajtott kerekekkel történik. Ezen motorok egyenkénti vagy együttes vezérlése, a forgásirány, illetve a fordulatszám határozza meg, hogy a targonca melyik irányba, egyenes vonalban, íven, vagy helyben fordulva mozog.

A manuális mozgatás a joystick-kel a következő, direkt módon vezérelhető (lásd 5.2. ábra):

- előre billentés: mindkét kerék előre hajt, ezáltal előre, egyenes vonalban mozog;
- hátra billentés: mindkét kerék hátra hajt, ezáltal hátra, egyenes vonalban mozog;
- jobbra billentés: a két kerék ellentétes irányba hajt, így egyhelyben fordul jobbra;
- balra billentés: a két kerék ellentétes irányba hajt, így egyhelyben fordul balra.

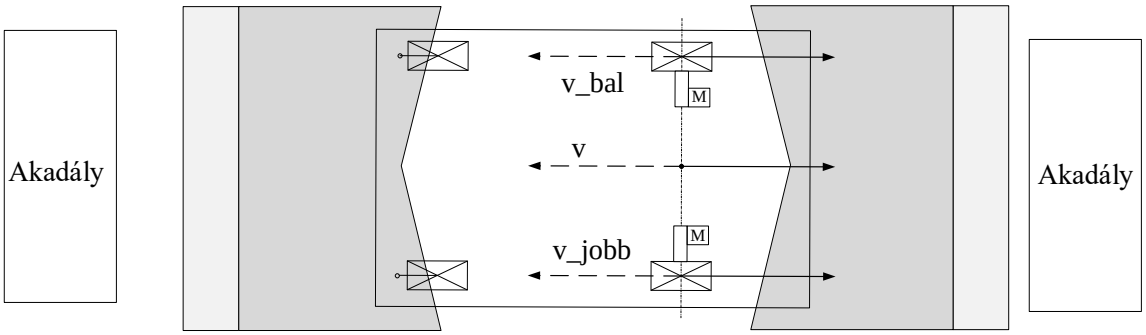


5.2. *ábra:* Vezető nélküli targonca előre, hátra, jobbra és balra mozgatása

A joystick használatához a programot ki kellett egészíteni a joystick 4 állásának érzékelésére, amely a PLC input csatornáira csatlakozik. A program az input adatokat a PLC-n keresztül kérdezi le, és továbbítja párhuzamos porton a szervomotor-vezérlőknek.

5.3. Automatikus mozgatás – két akadály között

Ez az alfejezet az automatikus mozgatás megvalósításának egyszerű példáját tárgyalja. Az automatikus mozgatás során a joystick kar már nem közvetlenül az irányokat határozza meg, hanem valamilyen folyamatot indít el. A jelen változatban a targonca egyenes vonalban mozog, és amint megközelít egy akadályt, automatikusan irányt vált. Így a mozgás egy alternáló előre-hátra mozgás lesz. A mozgást sematikusan az 5.3. ábra mutatja. A biztonsági szenzor érzékelési tartománya két zónára oszlik: a szenzortól távolabbi érzékelési tartomány csak figyelmeztetést küld a számítógépnek, míg a közelebbi érzékelési tartomány azonnal leállítja a motor villamos meghajtását, így megállítja a targoncát az ütközés elkerülése érdekében.

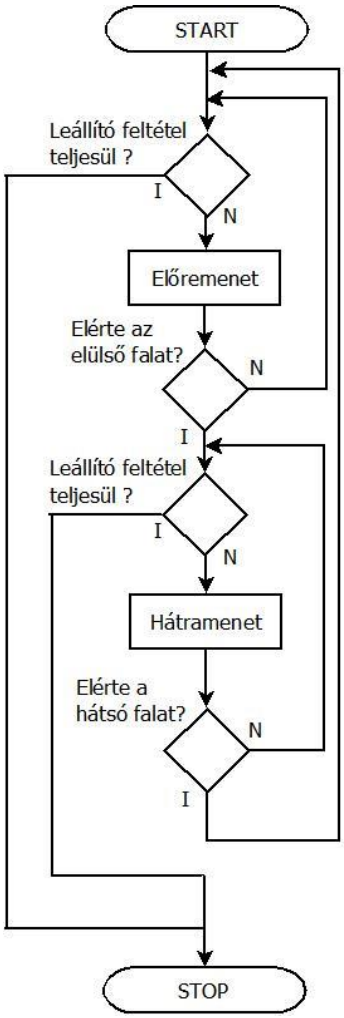


5.3. ábra: Vezető nélküli targonca két akadály közötti automatikus mozgatás sematikusan

Ezen automatikus mozgatási mód folyamatábrája az 5.4. ábrán figyelhető meg. A targonca előre mozog, amíg az elülső akadályt egy bizonyos távolságból el nem éri, majd hátrafelé mozog, amíg a hátsó akadályt meg nem közelíti.

A mozgást leállító feltételek a következők lehetnek:

- joystick karjának balra billentése;
- vészstop gomb benyomása;
- elülső vagy hátulsó biztonsági szenzor 2. zónába való kerülés.

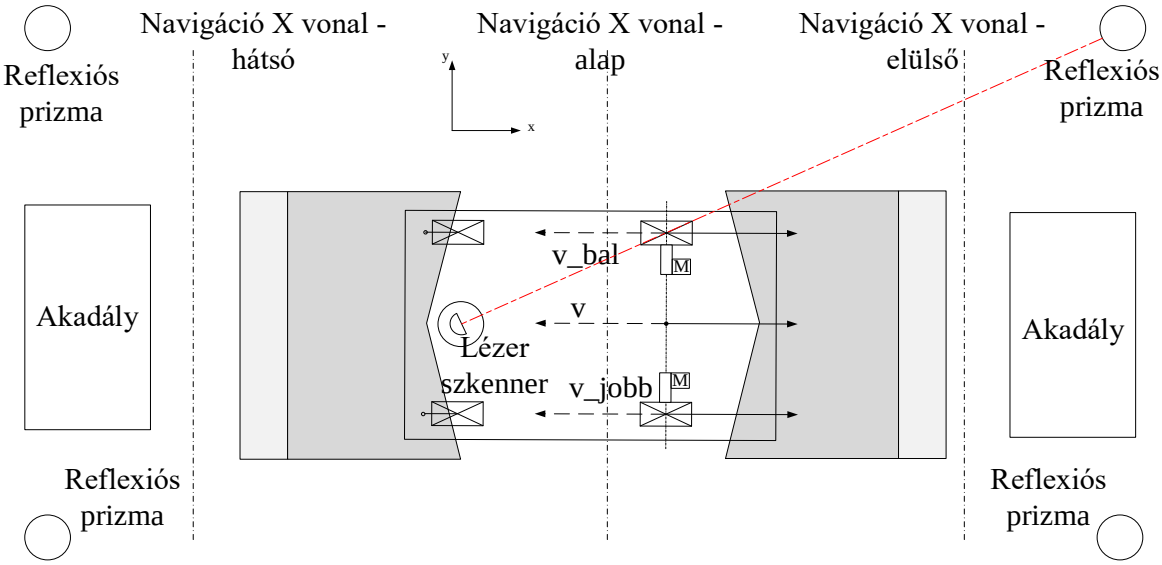


5.4. ábra: Vezető nélküli targonca automatikus mozgatás folyamatábrája

A két akadály közötti mozgatás kiegészül azzal, hogy adott navigációs pontok között hajtja az alternáló mozgást. Az automatizált mozgáshoz szenzorok szükségesek a környezet érzékeléséhez. Ebben a példában a targonca már nem csak a biztonsági szenzorokat, hanem egy Sick márkájú NAV350 típusú LIDAR szenzort (lásd 5.5. ábra) is használ a pozícióméréshez, amely a targonca tetején, körülbelül 2m magasságban található.

5.5. ábra: Vezető nélküli targoncán alkalmazott Sick márkájú NAV350 típusú LIDAR szenzor

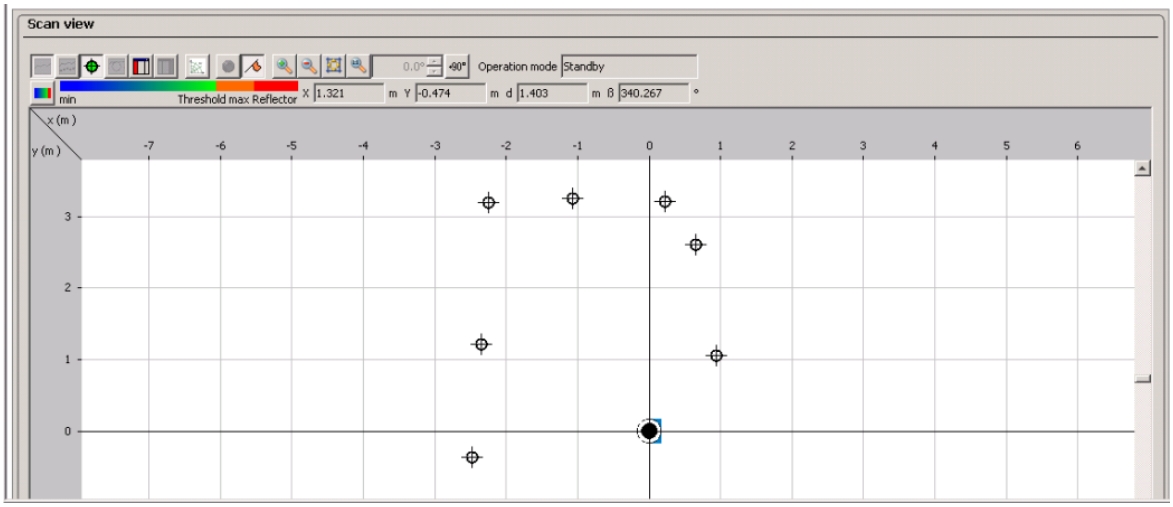
A LIDAR szenzor egy fedélzeti lézer kibocsátót és detektort tartalmaz. A LIDAR szenzor infrastruktúrájához a helység sarkaiban elhelyezett reflexiós prizmák tartoznak, ez utóbbiak képesek visszaverni a szenzor által kibocsátott lézersugarat, és ezáltal a szenzor kiszámolja az egység pozíciójának X és Y koordinátáját és orientációját, ahogy az 5.6. ábra illusztrálja. A LIDAR szenzor mérési frekvenciája $8Hz$, mérési pontossága $\pm 10 - 25mm$.



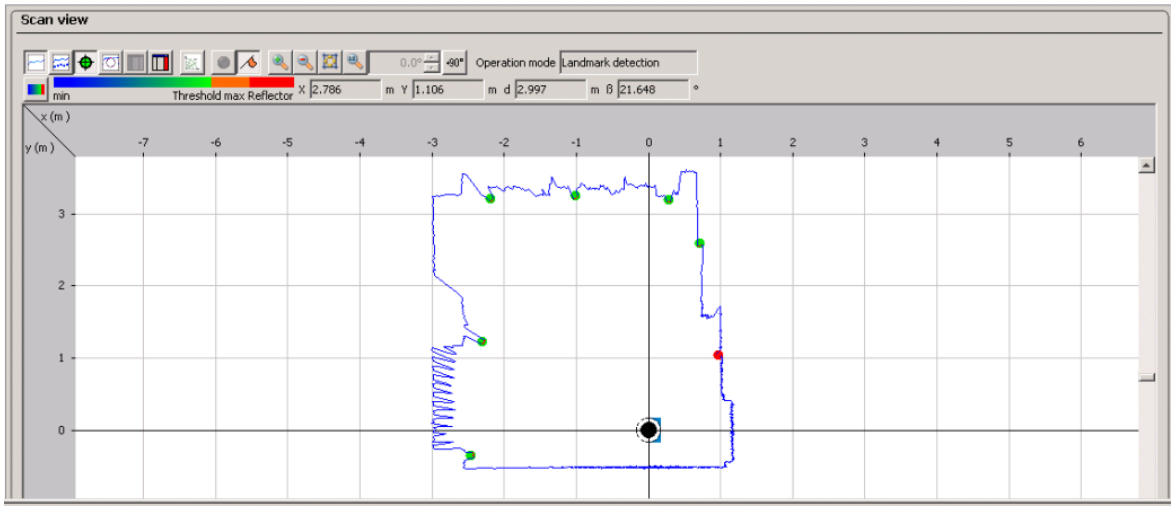
5.6. ábra: Vezető nélküli targonca automatikus mozgatásának elrendezése

A koordinátarendszer nullpontját egy szoftver segítségével kell beállítani. A navigációs egység továbbítja a PC felé a pozíciójának X és Y koordinátáját és orientációját, és ez alapján lehet pozícióalapú vezérlést kialakítani, valamint különböző pozíciómméréseket végezni.

A szoftverben első lépésként a helyiségben elhelyezett tükrök pozícióját kell beállítani, ahogy az 5.7. ábrán látható. Ezután pedig már le lehet kérdezni nem csak a tükrök helyzetét, hanem a helyiség kontúrját is, ahogy az 5.8. ábra mutatja.



5.7. ábra: A LIDAR szenzor által érzékelt tükrök a szenzorhoz biztosított szoftverben megjelenítve



5.8. ábra: A helység kontúrja a szenzorhoz biztosított szoftverben megjelenítve

A példában alkalmazandó alternáló mozgás folyamata a következőképpen valósul meg:

- A START jel után (joystick és nyomógombok kombinációja) az AGV előremeneti irányba mozog.
- Ha elérte az elülső „akadályt” VAGY az előírt navigációs vonalat, megáll, és elindul hátrameneti irányba.
- Az AGV hátrafelé mozog, amíg eléri a hátulsó „akadályt” vagy az előírt hátsó navigációs vonalat, azután előremenetben mozog.
- A STOP utasításig ezt az alternáló mozgást ismétli.

A STOP utasítás háromféleképpen állhat elő:

- Az egész folyamat során az AGV túl közel ér valamilyen akadályhoz (vészleállás).
- A vészkapcsolót benyomja valaki (vészleállás).
- A joystickot “bal” irányba dönti valaki (normál leállás).

Az 5. fejezet tartalma az [S11] és [S12] publikációban került megjelentetésre.

6. MÉRŐRENDSZER KIFEJLESZTÉSE AZ AGV ÁLLAPOTFELÜGYELETÉRE

A mozgásvezérlés részeként és annak ellenőrzése céljából egy mérőrendszer került kialakításra. A mérési adatok Microsoft Office Excel táblázatkezelőben való feldolgozhatósága érdekében egy fájlmentési rész is beépítésre került a vezérlőprogramba, amelyet a 6.1. fejezet tárgyal. A 6.2. fejezet a mérőrendszer áramerősség- és feszültség mérésre vonatkozó részét ismerteti. A 6.3. fejezet a feszültség-, áramerősség- és navigációs adatok PC-n történő fogadását és felhasználását foglalja össze.

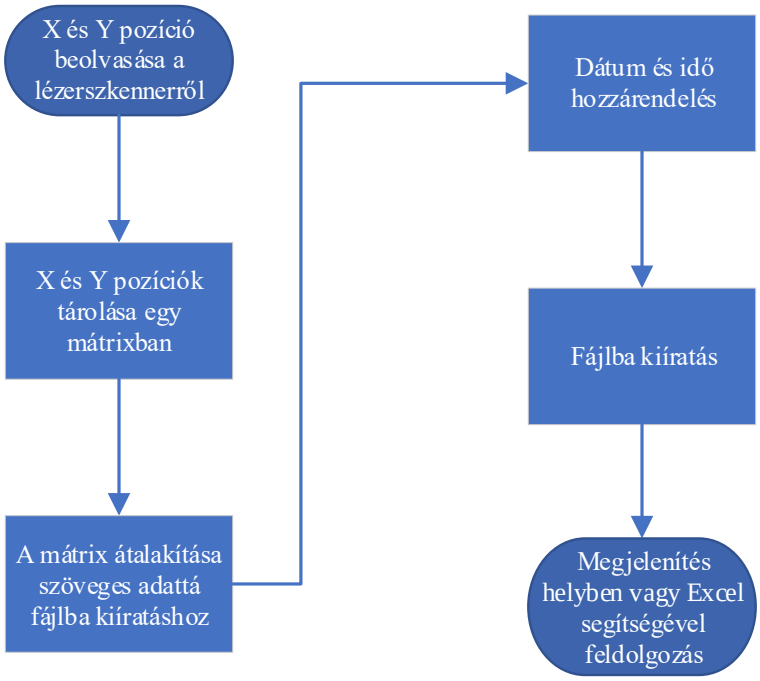
6.1. Mért adatok mentése további feldolgozáshoz

A targonca pozíciójának méréséért a LIDAR navigációs egység felelős.
A mérés elvégzéséhez és kiértékeléséhez a 6.1. ábrán szemléltetett folyamat került kialakításra, amely a következő elemeket tartalmazza:

1. X és Y pozíció beolvasása Ethernet kommunikáción keresztül a szenzorról.
2. A beolvasott pozíciók tárolása egy kétdimenziós mátrixban, amelyet a PC-n megírt program deklarált előzőleg.
3. A mérés befejezés után a mátrix átkonvertálása sztring adattípusra.
4. A mérés visszakövethetősége érdekében dátum és idő hozzárendelése egy másik sztring változóba.
5. A két sztring adat ismeretében fájlba kiíratás, ahol a fájl neve a dátumot, időt tartalmazza, a fájl tartalma pedig a mátrixból került kiírásra.
6. A fájlt tartalmát a targonca PC-jén is meg lehet jeleníteni, vagy egy másik PC-n egy Excel szoftver segítségével további feldolgozása.

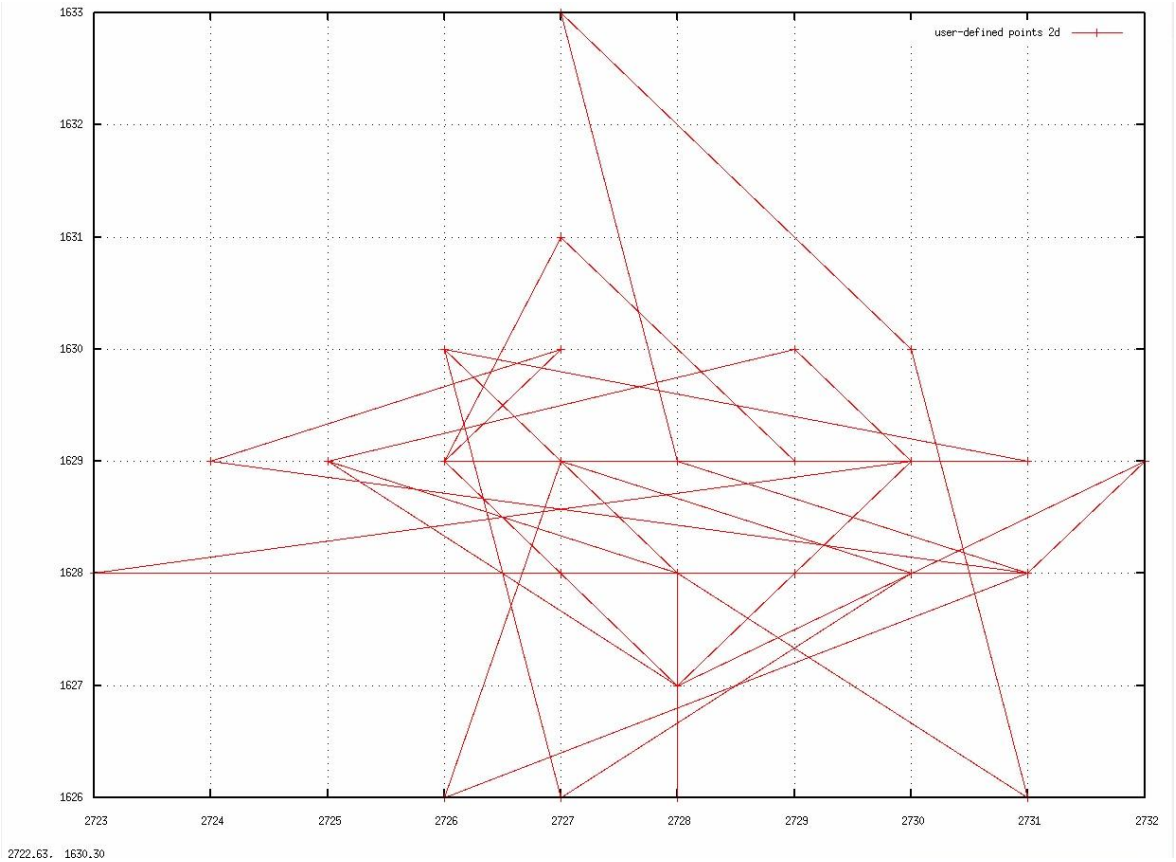
A mozgás során a helyzetek mért koordinátáinak rögzítése, mentése:

- a mérés elindítása olyan jel hatására történjen, ami nem állítja meg a főprogramot, ugyanis ekkor leállna a motorok vezérlése;
- maga a mérés sem állíthatja meg a főprogramot;
- a mérés során folyamatosan kell az X és Y pozíciót fogadni, és az értékeket megfelelően deklarált vektorokban tárolni;
- ha szükséges, akkor rajzolja ki az eddig mért pontokat a képernyőre, majd gombnyomásra fusson tovább a program;
- az értékeket egy fájlba írassa ki, további feldolgozás céljából.



6.1. ábra: A pozíció mérés rendszer működése

A mérést elindításához két nyomógomb lenyomásának érzékelését választottam. Ez a két nyomógomb és a joystick valamelyik irányába való mozgatása elindít egy adott automata mozgatási programot, így a mozgás megkezdésével elindul a mérés.



6.2. ábra: Álló AGV mért X-Y pozíciója - Fejlesztőprogram által mutatott diagram

A program a mérés elején az adott X és Y vektort nullázza, és ezt tölti fel folyamatosan minden egyes beolvasási ciklus során. Két pozícióolvasás között eltelt idő körülbelül 0,05s, ami megközelítőleg 20 mintavételnek felel meg másodpercenként.

Az Eclipse fejlesztőprogram által megrajzolt diagramot a 6.2. ábra mutatja. Ez a fejlesztőprogram nem ábrázolja a koordinátarendszer tengelyfeliratait, és utólagos posztprocesszálást sem tesz lehetővé, ellentétben az Excel táblázatkezelőben ábrázolt diagramokkal.

A program a pozíciókat egy .txt fájlba írja ki, a fájl neve az aktuális dátum és idő lesz. A szövegfájlban tabulátor választja el az X és Y koordinátákat, és soremeléssel az egyes időpontokban mért adatokat. Az ilyen fájlt a Microsoft Excel táblázatkezelővel már kényelmesen posztprocesszálható.

6.2. Mérőrendszerbe épített Arduino fejlesztőplatform

A feszültség- és áramerősség mérésekhez egy Arduino Uno fejlesztőplatform [90] került felhasználásra. A feszültségmérés egy feszültségosztó, míg az árammérés az ACS712 20A-s méréshatárú Hall-elemes árammérő eszközzel történik.

A feszültségmérés a négy darab sorba kötött, egyenként 12V-os névleges feszültségű akkumulátoron történik. Névleges feszültségük sorba kötve $U_1 = 48V$, maximális feszültségük elérheti az 57,6V-ot, így érdemes $U_{1_{max}} = 60V$ feszültség konvertálását $U_{2_{max}} = 5V$-ra kiszámítani. A kialakítást és összefüggéseket tartalmazó cikket [S12] felhasználva a névleges szekunder feszültség:

$$U_{2\max} = \frac{R_3}{R_1+R_2+R_3} \cdot U_{1\max} = \frac{10k\Omega}{100k\Omega+10k\Omega+10k\Omega} \cdot 60V = \frac{1}{12} \cdot 60V = 5V. \quad (86)$$

Az ellenállások valóságos értékei miatt a szekunder feszültség nem lesz azonos:

$$U'_{2\max} = \frac{R'_3}{R'_1+R'_2+R'_3} \cdot U_{1\max} = \frac{10k\Omega}{100{,}1k\Omega+10{,}01k\Omega+10k\Omega} \cdot 60V = \frac{1}{12{,}011} \cdot 60V = 4{,}99542V. \quad (87)$$

A viszonyszámra (12,011) a PC-n való programozás során van szükség. Az Arduino Uno fejlesztőplatform 10 bites A/D átalakítóval rendelkezik, így az analóg feszültségértéket $2^{10} = 1024$ felbontással tudja digitális jellé alakítani. A mért feszültségérték a számítógépes programba történő átalakítási folyamata:

$$U_{Arduino} = 0 \div 5V \rightarrow value_{U,Arduino}: 0 \div 1023 \rightarrow U_{program} = 0 \div 60V. \quad (88)$$

Azaz a programban a következő számítás történik az Arduino által szolgáltatott digitális adattal:

$$U_{program} = \frac{value_{U,Arduino}}{1024} \cdot 5V \cdot 12{,}011. \quad (89)$$

Az árammérés 2db ACS712 áramérzékelővel történik, az egyik a bal oldali kereket hajtó motor, míg a másik a jobb oldali kereket hajtó motor áramerősségét méri. Mivel ezek az elektronikai eszközök is az Arduino fejlesztőplatformra vannak kötve, így a valós áramerősségből a következő átalakítási úton lesz felhasználható adat a PC-n futó program számára:

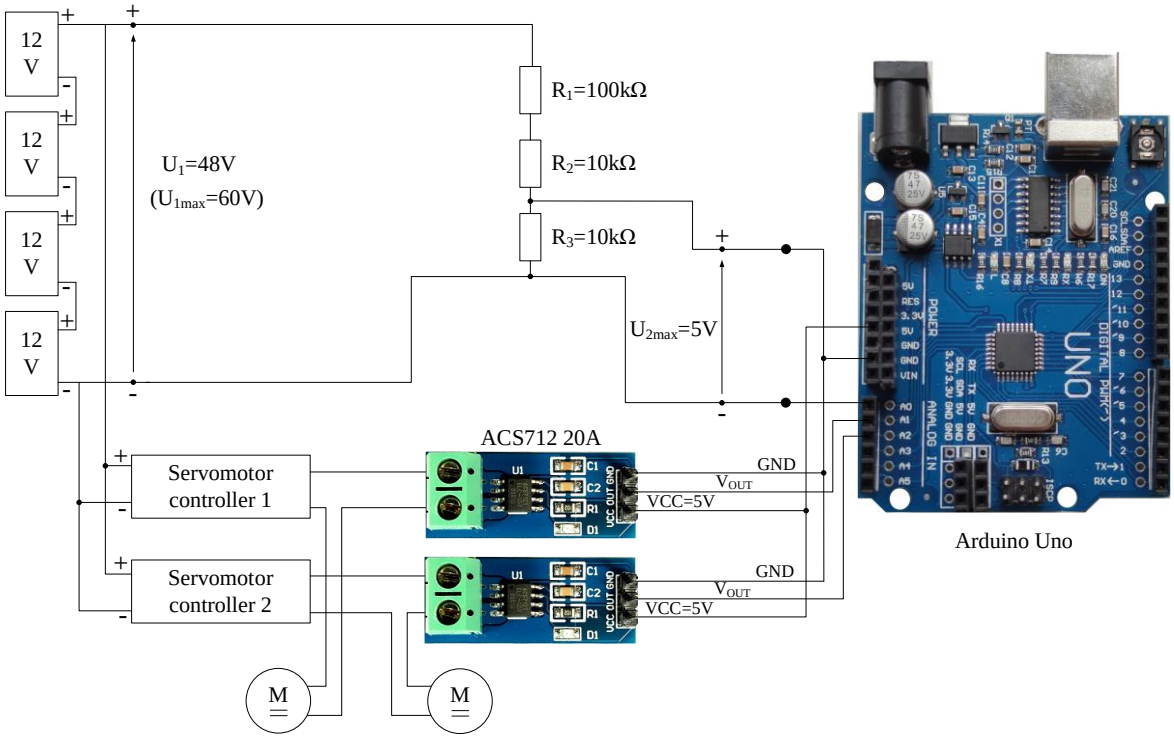
$$i_{left,Arduino} = -20A \div 20A \rightarrow value_{i_{left},Arduino}: 0 \div 1023 \rightarrow$$
$$\rightarrow i_{left,program} = 0A \div 40A \rightarrow i_{left,program} = -20A \div 20A. \quad (90)$$

A konvertálás a következőképpen történik:

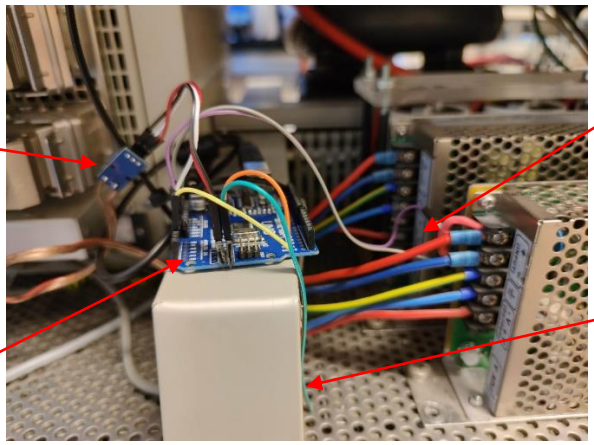
$$i_{left,program} = \frac{value_{i_{left},Arduino}}{1024} \cdot 40A - 20\text{A}. \quad (91)$$

Az (91) és (92) összefüggések fennállnak a másik árammérőre is, amely során $i_{right,Arduino}$, $value_{i_{right},Arduino}$ és $i_{right,program}$ értékek adódnak.

Az elektronikai kapcsolást a 6.3. ábra, míg a targoncába helyezett kapcsolást a 6.4. ábra mutatja.



6.3. ábra: Feszültség- és áramerősségmérő kapcsolás az Arduino Uno fejlesztőplatformmal



6.4. ábra: Feszültség- és áramerősségmérő kapcsolás beszerelve a targoncába

Az Arduino Uno platformra megírt program a feszültségosztóból érkező arányos feszültségjel, és az árammérőkből érkező feszültségjelek fogadásáért és átalakításáért felelős a PC-re történő tovább küldés érdekében.

A program első részében a változók deklarálása történik meg, mint az elektronikai elemek pin-jének hozzárendelése, és további segédváltozók deklarálása. A második részben a beállítások történnek meg, mint a soros port nyitása, az előzőleg deklarált pin-ek módjának beállítása.

A főprogram az alábbi alprogramokat hívja meg a futtatás során:

- *u_average1024():* a feszültségosztóról beérkező feszültségjel mintavételezése 100-szor, majd az ebből képzett átlag átalakítása 0-1023 tartományban levő értékké
- *i_left_1024():* az egyik árammérőről beérkező feszültségjel mintavételezése 100-szor, majd az ebből képzett átlag átalakítása 0-1023 tartományban levő értékké
- *i_right_1024():* a másik árammérőről beérkező feszültségjel mintavételezése 100-szor, majd az ebből képzett átlag átalakítása 0-1023 tartományban levő értékké

A soros porton való üzenet küldéséhez még egy *„Serial_0000”* alprogram megírására volt szükség. Önmagában az előző három alprogram egy és négy közötti helyiértékű értéket ad vissza, amely a PC oldaláról az üzenet figyelése miatt bizonytalanná válhat, így érdemes fix négy helyiértékű értéket, azaz 0000 és 1023 közötti értéket továbbítani. Ennek megoldása az adott változó értékének figyelése, és ha 10-nél, 100-nál vagy 1000-nél kisebb értékű, akkor a tényleges érték elé rendre három, kettő vagy egy db „0” -t helyez el a soros portos kiíratásnál.

A főprogram ezek után először egy „s” értéket írat ki a soros portra, ezzel jelezve, hogy az üzenet elkezdődik, majd ezután a feszültségosztóról és mindkettő árammérőről érkező feszültségjel *„Serial_0000”* alprogrammal történő soros portos kiíratás történik meg.

Megfigyelhető, hogy a legelső sorból a feszültségértékre „0755” adódik, amiből a (4) összefüggést felhasználva $U_{program} = 42{,}729V$, a kettő árammérőnél „0542” és „0514” adódik, amiből a (6) összefüggést felhasználva $i_{left,program} = 1{,}172A$ és $i_{right,program} = 0{,}078A$ értékek keletkeznek. A feszültség-, és áramérzékelők, és az Arduino fejlesztőplatform ipari eszközökhöz képesti alacsonyabb minősége miatt az értékek változhatnak.

6.3. Mérőrendszer kialakítás a PC-re

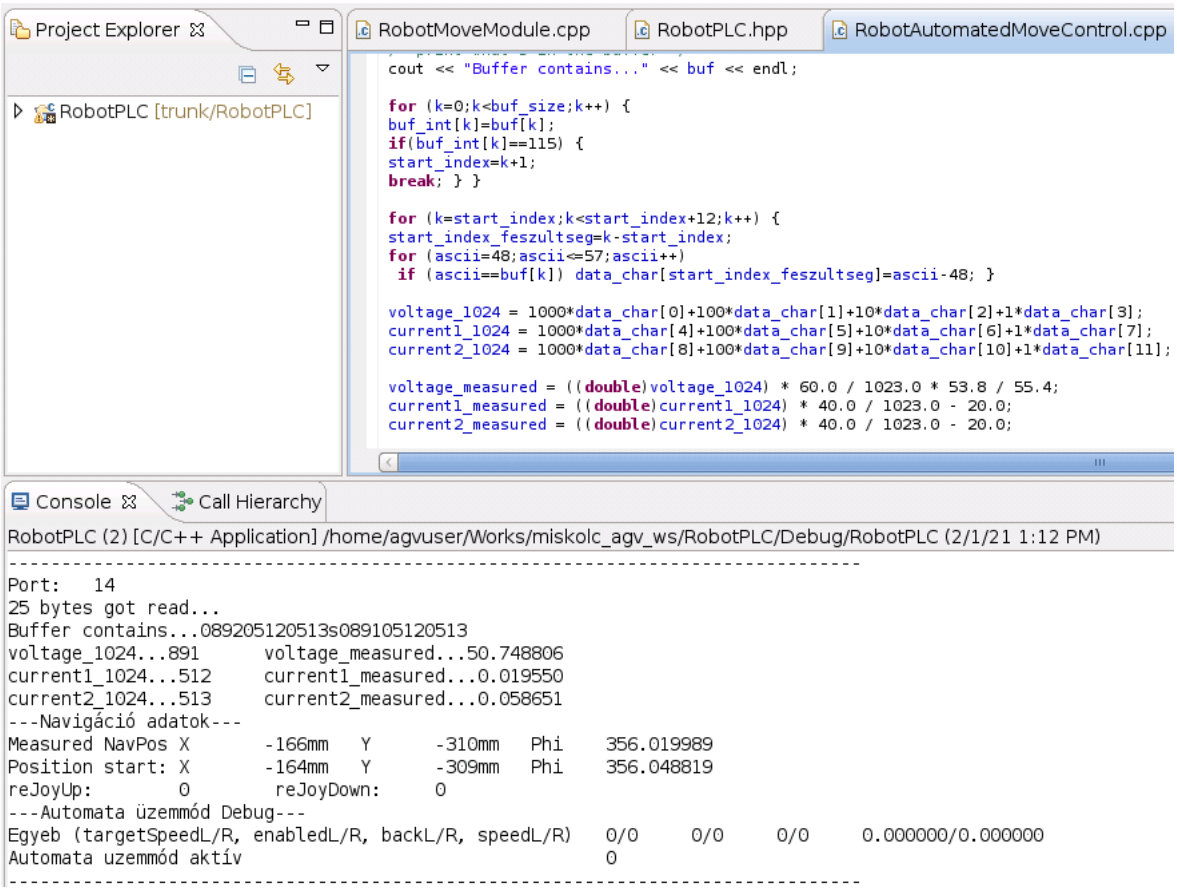
Jelen alfejezetben bemutatott részben a PC oldaláról az Arduino fejlesztőplatformról érkező üzenet, valamint a navigációs LIDAR szenzorral érkező jelek fogadása és feldolgozása történik meg.

Feszültség- és áramerősség adat fogadásához és -feldolgozásához első körben az Arduino fejlesztőplatformmal való kommunikációt kell megvalósítani. A PC-re az Arduino USB porton keresztül csatlakozik. A targoncavezérlő programban először ezt az USB portot kell megnyitni, ami Linux alapú rendszeren a „ttyUSB0” lesz.

Mivel az Arduino és PC közötti kommunikáció aszinkron jellegű, a PC-re megérkező üzenet nem biztos, hogy pontosan az „s” karakterrel kezdődik. Ennek érdekében egyszerre annyi karaktert kell beolvasni, hogy az „s” és az utána következő 12 karaktert biztosan beolvassa, még ha az „s” a 13. helyen is lenne. Ennélfogva összesen 25 karaktert kell kezelnie a PC programban. A 25 karakteres változóból a programnak karakterről karakterre léptetve meg kell keresnie az „s” karaktert, és rögzítenie a karakter sorszámát, ami 1 és 13 között lehet. Ezután tovább léptetve a karaktereken négyesével a karaktereket át kell alakítani egész számmá, amelyhez az ASCII-tábla került felhasználásra. Az egész számmá alakítás után történhet meg a (4) és (6) összefüggések szerinti átszámítás.

A 6.5. ábra felső részén a programkód egy része látható, míg az alsó részén a kiíratott eredmények. A legelső sorban lehet látni, hogy a program a 14. számú porton 25 byte-ot olvasott be, ezután a beolvasott karaktersorozatot fogadta, ahol a 13. karakterre érkezett az üzenete kezdetét jelző „s” karakter. A következő 3 sorban a feszültség- és árammérőkről érkező értékek és annak átalakított verziói olvashatók.

A targonca automatikus mozgatását érdemes nem csak egyféleképpen megtervezni, hanem úgy érdemes kialakítani, hogy több, például automata üzemmód közül lehessen választani. Ennek érdekében egy olyan figyelőrendszer került kifejlesztésre, amely végeredményben megállapítja, hogy milyen módon, melyik irányba és milyen sebességgel induljon el a targonca.



6.5. ábra: Feszültség, áramerősség, valamint navigációs adatok feldolgozás a PC-n

A figyelőrendszer részét képezi a targonca kezelőpaneljén található 1db fekete és 1db zöld nyomógomb és középen egy 4 irányú joystick, ahogy a 6.6. ábrán látható.

6.6. ábra: Vezető nélküli targonca kezelőpanelje

Valamelyik automata üzemmód indításához 5 másodpercen belül a két nyomógombot egyszerre kell megnyomni, és a joystick-ot előírt számban és megfelelő irányba billenteni, ahogy a 3. táblázat részletezi. A joystick billentésének irányát és számát a programban egy 5s időintervallumon belül, a joystick irányainak felfutó él jeleinek vizsgálatával történik, amint a „reJoyUp” és „reJoyDown” értékek láthatók a 6.5. ábra alsó felén.

3. táblázat: Automata üzemmódok

Üzemmód	**Jellemző**	**Indítás feltétele**
0	Nem automata	-
1	Előre-hátra mozgás két akadály között	Joystick 1x FEL
2	Adott ponthoz, előre mozgás alacsonyabb sebességgel	Joystick 2x FEL
3	Adott ponthoz, előre mozgás magasabb sebességgel	Joystick 3x FEL
4	Adott ponthoz, hátra mozgás	Joystick 1x FEL

A targonca automatizált mozgatáshoz szükség van a pozíció és orientáció meghatározásához, azaz navigációs adatok feldolgozásához. Ha $1mm$ pontossággal kellene kezelni az adatokat, akkor nagy bizonytalansággal működne a targonca, ahogy a 7.3. ábra szemlélteti. Ezért az a következtetés tehető, hogy a navigációs adatok feldolgozása minél kevesebbszer és minél ritkábban kerüljön alkalmazásra. A programozás során arra törekedtem, hogy csak az automatizált mozgatás előtt történjen egy mérés, menet közben már ne. Ennek érdekében viszont relatíve pontos pálya- és trajektóriatervezésre van szükség, ahogy azt a 4. fejezet tárgyalja.

Az automata üzemmód indítás előtti mintavételezés nem csak egy mérési adaton, hanem megközelítőleg 5 másodpercen keresztüli folyamatos vizsgálaton alapul. A program az így kapott 40 adatot átlagolja, ezzel csökkentve a mérési bizonytalanságból adódó kilengések mértékét. Ahogy a 6.5. ábra alsó felében a „Navigációs adatok” részben is látható, az aktuális és a mintavételezett navigációs adatok a program futása során folyamatosan követhető.

4. TÉZIS: A targonca vezérléséhez szükséges paraméterek meghatározásához egy olyan új mérőrendszert fejlesztettem ki, amely fogadja a targonca navigációs adatait, méri a motorok áramerősségeit és a sorba kötött akkumulátorok feszültségét, és transzformálja a vezérlőrendszer számára. A mérőrendszer felhasználásával egy olyan vezérlővezérlőrendszert alakítottam ki, amely a kisebb áramfelvétellel járó utat választva vezérli a targonca mozgását.

Vonatkozó publikációk: [S2], [S4], [S6]

7. A TARGONCÁN ELVÉGZETT KÍSÉRLETEK ÉS MÉRÉSEK

A 7.1. fejezet különböző technikai feltételek figyelembevételével elvégzett mérési sorozatot ismerteti, amely során vizsgálatra kerül a motor fordulatszámának a hatásoktól való függése. A 7.2. alfejezet a LIDAR szenzorral mért különböző példákat és eredményeit mutatja be. A 7.3. fejezet a 6. fejezetben részletezett mérőrendszer felhasználásával a targonca előre- és hátrameneti automata mozgása során mért eredményeket dolgozza fel.

7.1. AGV kerék fordulatszámának vizsgálata különböző esetekben

Korábban nem állt rendelkezésre olyan adat, amely biztosan jelezte volna, hogy a hajtómotor fordulatszáma változik-e az alábbi körülmények között:

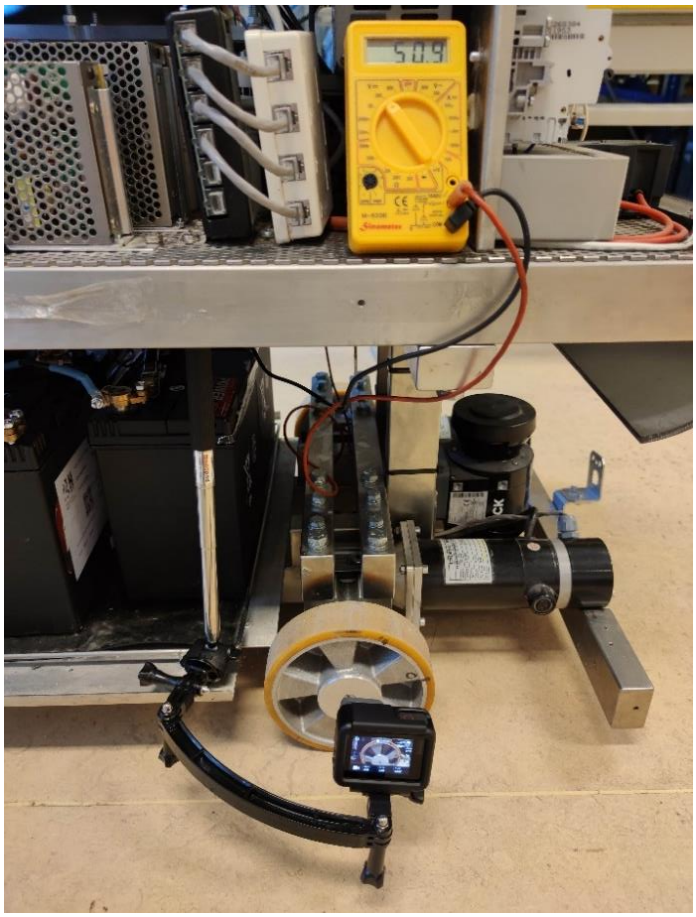
- 4db sorba kötött akkumulátor feszültségének változása,
- előre vagy hátra történő mozgás,
- terhelés nélküli eset, azaz a targonca felemelt helyzetében a kerék szabadon foroghat, vagy terheléses eset, azaz a targonca normál helyzetben, a kerekeivel mozog,
- PC-n írt programban a kimenő érték állításával ténylegesen lineárisan változik-e a fordulatszám.

Ezen hatások vizsgálatára egy mérés-sorozat került megtervezésre és végrehajtásra. A mérés 4db különböző, a targoncába beépített PC által előállított kimenő értékre történt, ezeket az első 3 felsorolt eset esetén megismételve, összesen 32db mérés került elvégzésre. A mérés a kerék mellé rögzített kamerával történt, amely videófelvételeket készített. A videók elemzésével történt meg az egy fordulathoz szükséges idő megállapítása.

A 7.1. ábra alsó részén látható a mérés fő eszköze egy GoPro márkájú Hero 5 típusú kamera, amely a targoncára került rögzítésre. A mérési videó készítése során 240fps (frame-per-second) és 848x480-as felbontási beállítást alkalmaztam, mert a magas felbontás nem volt lényeges. Ebben az esetben két képkocka közötti periódusidő kiszámolható:

$$T_{camera} = \frac{1}{240fps} = 0{,}004167s \tag{92}$$

A mérések egyik esete a targoncát hajtó 4db sorba kötött akkumulátoron mért feszültség volt, amely egy multiméterrel történik, ahogy a 7.1. ábra felső része mutatja.



7.1. ábra: Kerék fordulatszámának mérésének összeállítása

A kiértékelés a videók alapján történt. A 7.2. ábra illusztrálja a kamera szemszögéből látható képet. Amint az ábrán látható, a keréken felső részén egy függőleges vonal került megjelölésre. A kerék forgása során ez a vonal is forog, azonban a videóban mindig lesz két olyan időpillanat, amikor ez a vonal ismét a felső helyzetében lesz. A kielemzés során ezen két helyzet között eltelt idő került meghatározásra.

7.2. ábra: Kerék fordulatszámának mérése a kamera szemszögéből

A mérési videók kielemzése és a feszültségek, irányok, terhelési esetek, program által beállított sebességértékek meghatározása után foglaltam össze a 32db mérés eredményét az

4. táblázat szerint. A mért időből ki lehet számolni a kerék percenként megtett fordulatszámát:

$$n_{W,mérés} = \frac{1}{t_{W,mérés}} \cdot 60\ [rpm]\,. \tag{93}$$

4. táblázat: Kerék fordulatszámának mérési eredményei

Mérés ID	Program sebesség-érték [-]	Targonca helyzete	Forgás iránya	Mért feszültség [V]	Mért idő [s]	Kerék percenkénti fordulatszám [rpm]	Egységre vetített fordulatszám [rpm]
1	1000	Felemelt	Előre	46,4	0,8	75	75
2	1000	Felemelt	Hátra	46,4	0,797	75,28230866	75,28230866
3	500	Felemelt	Előre	46,7	1,602	37,45318352	74,90636704
4	500	Felemelt	Hátra	46,7	1,595	37,61755486	75,23510972
5	300	Felemelt	Előre	46,8	2,664	22,52252252	75,07507508
6	300	Felemelt	Hátra	46,8	2,66	22,55639098	75,18796992
7	100	Felemelt	Előre	46,9	7,999	7,500937617	75,00937617
8	100	Felemelt	Hátra	46,9	7,999	7,500937617	75,00937617
9	1000	Leengedett	Előre	46,1	0,802	74,81296758	74,81296758
10	1000	Leengedett	Hátra	46,1	0,797	75,28230866	75,28230866
11	500	Leengedett	Előre	46,4	1,599	37,52345216	75,04690432
12	500	Leengedett	Hátra	46,4	1,605	37,38317757	74,76635514
13	300	Leengedett	Előre	46,6	2,668	22,48875562	74,96251874
14	300	Leengedett	Hátra	46,6	2,664	22,52252252	75,07507508
15	100	Leengedett	Előre	46,7	8	7,5	75
16	100	Leengedett	Hátra	46,7	7,999	7,500937617	75,00937617
17	1000	Felemelt	Előre	50,5	0,801	74,90636704	74,90636704
18	1000	Felemelt	Hátra	50,5	0,8	75	75
19	500	Felemelt	Előre	50,6	1,601	37,47657714	74,95315428
20	500	Felemelt	Hátra	50,6	1,6	37,5	75
21	300	Felemelt	Előre	50,7	2,664	22,52252252	75,07507508
22	300	Felemelt	Hátra	50,7	2,668	22,48875562	74,96251874
23	100	Felemelt	Előre	50,8	7,997	7,502813555	75,02813555
24	100	Felemelt	Hátra	50,8	7,997	7,502813555	75,02813555
25	1000	Leengedett	Előre	50,2	0,794	75,56675063	75,56675063
26	1000	Leengedett	Hátra	50,2	0,799	75,09386733	75,09386733
27	500	Leengedett	Előre	50,3	1,596	37,59398496	75,18796992
28	500	Leengedett	Hátra	50,3	1,598	37,54693367	75,09386733
29	300	Leengedett	Előre	50,5	2,666	22,50562641	75,01875469
30	300	Leengedett	Hátra	50,5	2,665	22,51407129	75,04690432
31	100	Leengedett	Előre	50,6	8,017	7,484096295	74,84096295
32	100	Leengedett	Hátra	50,6	7,999	7,500937617	75,00937617

A különböző sebességértékek esetén különböző fordulatszámok keletkeznek, ezek linearitását egységre vetített fordulatszámként vizsgálom az alábbi összefüggéssel:

$$n_{W,mérés,1000} = \frac{n_{W,mérés}}{n_{M,setup}} \cdot 1000\ [rpm]. \tag{94}$$

Az egységre vetített fordulatszámokból megállapítható, hogy a $n_{W,mérés,1000} = 75$ értéktől még a legnagyobb eltérés maximum 0.4%. Ennélfogva a kerék fordulatszáma a különböző hatásoktól független, a beállított sebességértékkel lineáris marad. Ez azzal magyarázható, hogy a szervomotor az enkóder segítségével fordulatszámát pontosan szabályozza és az előírt értéken tartja.

A program a megfelelő sebesség eléréséhez egy adott szám alapján (például 100, 300, 500 vagy 1000) állítja elő a motorvezérlő számára szükséges vezérlőjelet. Azonban ennek tényleges megvalósulása nem volt ismert a targonca kerék fordulatszámára vetítve. Így szükség van egy viszonyszámra, amely a trajektóriatervező által előállított v_R és v_L kerék sebességértékeket átkonvertálja a motorvezérlő számára szükséges értékké. Az alábbiakban egy kerékre történik a számítás.

A mérésből meg lehetett állapítani, hogy a $n_{M,setup} = 1000$ vezérlőprogram beállított értékhez $n_W = 75rpm$ kerék fordulatszám tartozik, ez alapján a $n_{M,setup}$ a kerék fordulatszáma függvényében:

$$n_{M,setup} = \frac{n_W}{75} \cdot 1000 = \frac{n_M \cdot i_H}{75} \cdot 1000 = \frac{n_M}{1875} \cdot 1000\ [-]. \tag{95}$$

A kerék fordulatszáma a kerék szögsebessége és sebessége segítségével felírva:

$$n_W = \frac{\omega_W}{2 \cdot \pi} = \frac{v_W}{2 \cdot r_W \cdot \pi} \cdot 60\ [rpm]. \tag{96}$$

A motor fordulatszáma a kerék fordulatszámából átszámítva:

$$n_M = n_W \cdot 25 = \frac{25 \cdot v_W}{2 \cdot r_W \cdot \pi} \cdot 60\ [rpm]. \tag{97}$$

A vezérlőprogram beállított értéke így már kiszámítható a kerék sebesség ismeretében:

$$n_{M,setup} = \frac{n_M}{1875} \cdot 1000 = \frac{25 \cdot 60 \cdot 1000}{2 \cdot 1875 \cdot r_W \cdot \pi} \cdot v_W = \frac{400}{r_W \cdot \pi} \cdot v_W\ [-]. \tag{98}$$

Erre a viszonyszámra van szükség a vezérlőprogramban, hogy a helyes kerék sebesség álljon elő.

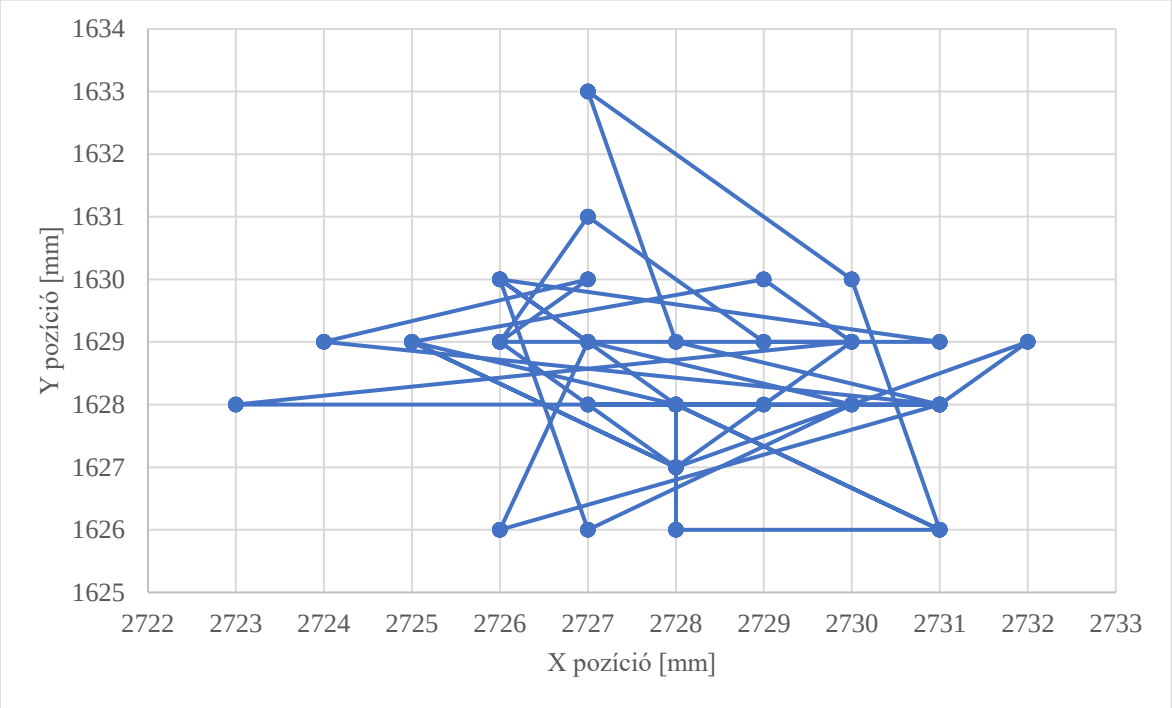
7.2. Targonca LIDAR szenzorával mért pozícióértékek

A jelen alfejezet a targonca LIDAR szenzorával mért 3 különböző mérést ismertet, a mérőrendszer működőképességének tesztelésére:

- A targonca egyhelyben áll.
- A targonca 1 db egyenes vonalú, oda-vissza mozgása X=0-1000mm között.
- A targonca 5 db egyenes vonalú, oda-vissza mozgása X=0-1000mm között.

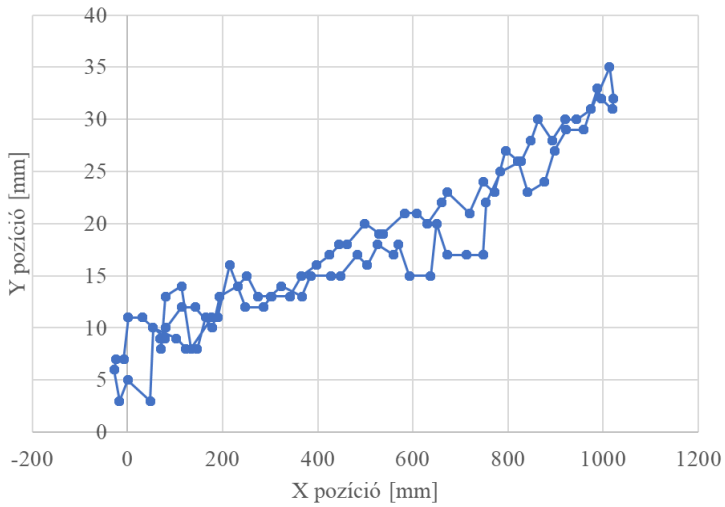
A helyben álló targonca pozíciójának meghatározásakor az X és Y koordinátáknak nem kellene változniuk. A kapott mérési eredmények a 7.3. ábrán látható, hogy a mért értékek X

irányban maximum 9mm-t, Y irányban pedig 7mm-t változnak. A helymeghatározás hibával terhelt.



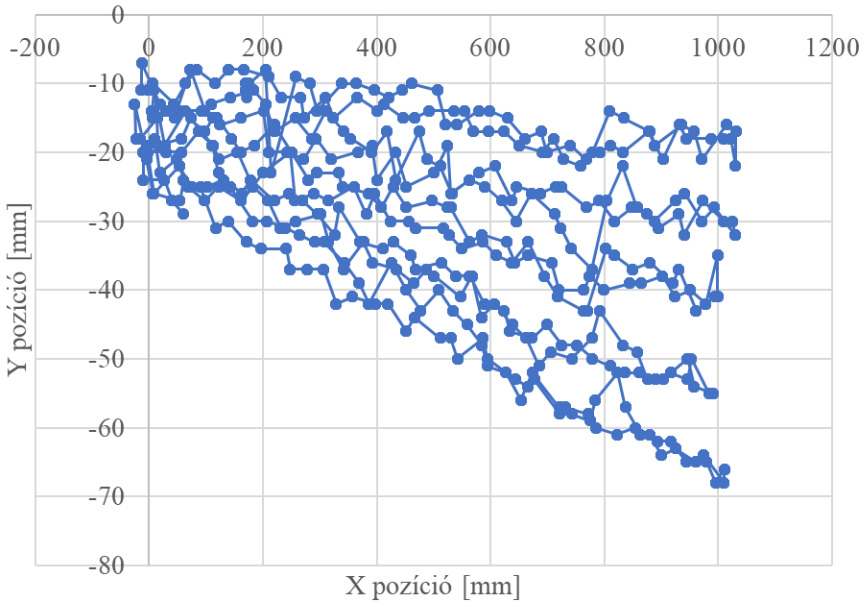
7.3. ábra: Álló AGV mért pozíció X és Y koordinátái

A második mérés során a targonca egy egyenes vonalú pályán egyszer oda-vissza mozgást végez X=0-1000mm között, így elméletileg két egyenesnek kellene adódnia. A 7.4. ábra alapján azonban megállapítható, hogy a targonca mért pályája nem tökéletesen egyenes vonalú, és az oda-vissza pályák nem esnek egybe. Ez részben a helymeghatározás hibájával, a targonca irányváltása során a beálló kerék elfordulásával, valamint a kerék és talaj közötti felületi egyenetlenségekkel magyarázható. A maximális hiba $Y = 35mm$.



7.4. ábra: Mért pozíció X és Y koordinátái – egyszeri egyenes vonalú, oda-vissza mozgás X=0-1000mm között

A harmadik mérés során a targonca ötszöri egyenes vonalú oda-vissza mozgást végez X=0-1000mm úthosszon. Elméletileg tíz egymást fedő egyenest kellene kapni. A 7.5. ábrán látható mérési adatok alapján megállapítható, hogy az ismételt pályák párosával fokozatosan távolodnak egymástól Y irányban. Az ötödik kísérletre a maximális hiba $Y = -68mm$.



7.5. ábra: Mért pozíció X és Y koordinátái – ötszöri egyenes vonalú, oda-vissza mozgás X=0-1000mm között

7.3. Mérési eredmények a targonca automatikus pályavezérlésénél

Jelen alfejezet a targonca egy adott ponthoz való mozgása közben mért feszültség, áramerősség és navigációs adatait mutatja be diagramok formájában.

A targonca LIDAR szenzorának minden egyes mérésekor vett adatát át kell konvertálni a kerekek közötti felezőpontra, ezáltal összehasonlíthatóvá téve a pályatervező által generált pályát és a ténylegesen megvalósított pályát a következő módon:

$$X_{actual,center} = X_{actual,LIDAR} - s_{LIDAR-W} \cdot \sin(\varphi_{actual,LIDAR}), \quad (99)$$

$$Y_{actual,center} = Y_{actual,LIDAR} + s_{LIDAR-W} \cdot \cos(\varphi_{actual,LIDAR}), \quad (100)$$

$$\varphi_{actual,center} = \varphi_{actual,LIDAR}, \quad (101)$$

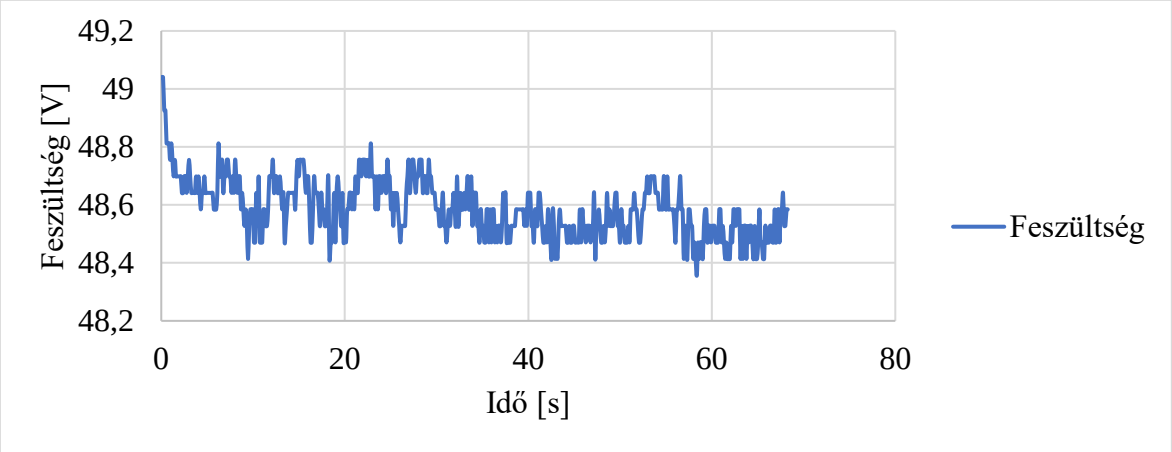
ahol $X_{actual,LIDAR}$, $Y_{actual,LIDAR}$ és $\varphi_{actual,LIDAR}$ a LIDAR szenzor által mért értékek, $X_{actual,center}$, $Y_{actual,center}$ és $\varphi_{actual,center}$ a kerekek közötti felezőpontra transzformált értékek.

Az előremeneti mozgás megtervezése során a targonca pálya- és trajektóriatervezése 2 szegmenset vesz alapul:

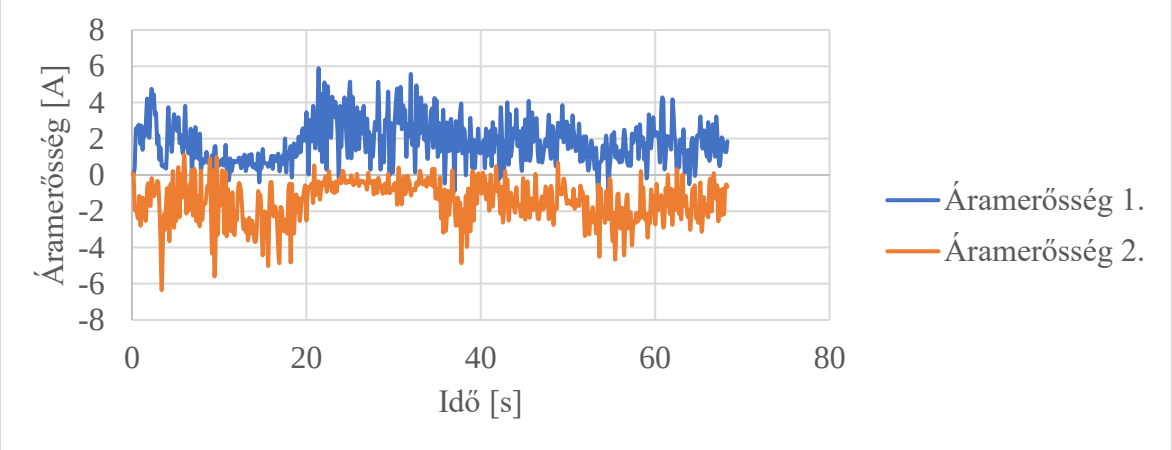
1. szegmens: A mért $\boldsymbol{P}^1_{start} = [X^1_{start}; Y^1_{start}]$ pozícióból indult a $\boldsymbol{P}^1_{end} = [X^1_{end}; Y^1_{end}] = [1.5m; 0m]$ pozícióba és $\varphi^1_{end} = 0°$ orientációba,
2. szegmens: A 1. szegmens végéről a $\boldsymbol{P}^2_{end} = [X^2_{end}; Y^2_{end}] = [3m; 0m]$ pozícióba és $\varphi^2_{end} = 0°$ orientációba folytatta útját.

A mérés kezdetén a targonca a $\boldsymbol{P}^1_{start} = [X^1_{start}; Y^1_{start}] = [0.150m; 0.468m]$ pozícióból és $\varphi^1_{start} = 351.604°$ orientációval indult el.

A hajtóakkumulátorok feszültségét az idő függvényében a 7.6. ábra, míg a két szervomotor által felvett áramerősség értékeket az idő függvényében a 7.7. ábra, a targonca tervezett és megvalósított útját X-Y koordinátarendszerben a 7.8. ábra illusztrálja. A feszültség függvénye oszcilláló képet mutat, tendenciájában kissé csökkent, amely a töltöttség csökkenésére utal.



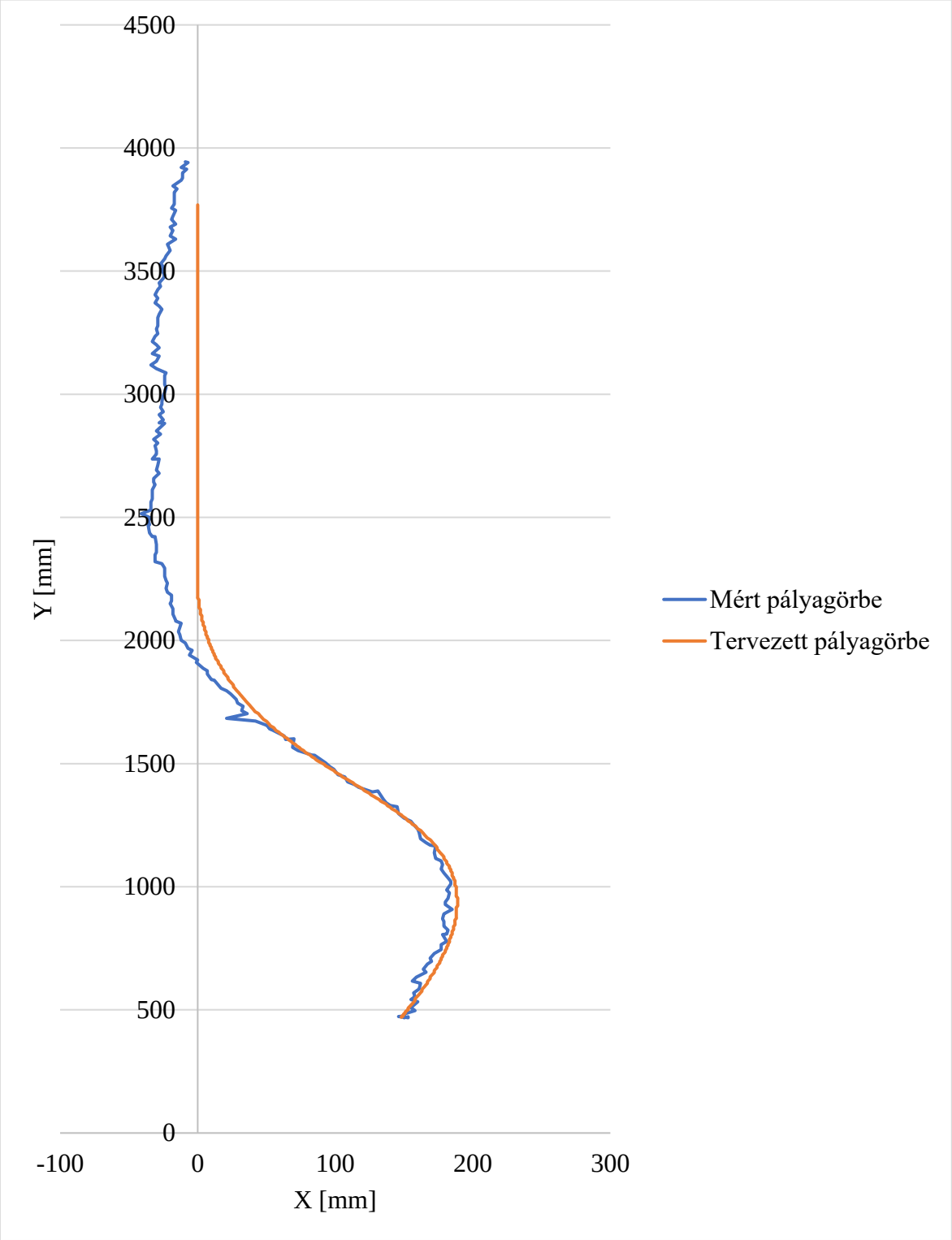
7.6. ábra: Mért feszültség – targonca automata előremeneti mozgás



7.7. ábra: Mért áramerősségek – targonca automata előremeneti mozgás

A 7.7. ábrán a két motor áramértéket ellenkező előjellel ábrázoltam az összehasonlítás kedvéért. A két függvény lefutása nem mondható szimmetrikusnak, hiszen a targonca nagyrészt íves pályán mozog, ezért az áramfelvétel ezt tükrözi. A targonca pályakövetése az

út első harmadában láthatóan pontos, de az idő előrehaladtával eltérés átmenetileg növekszik, majd kissé csökken.



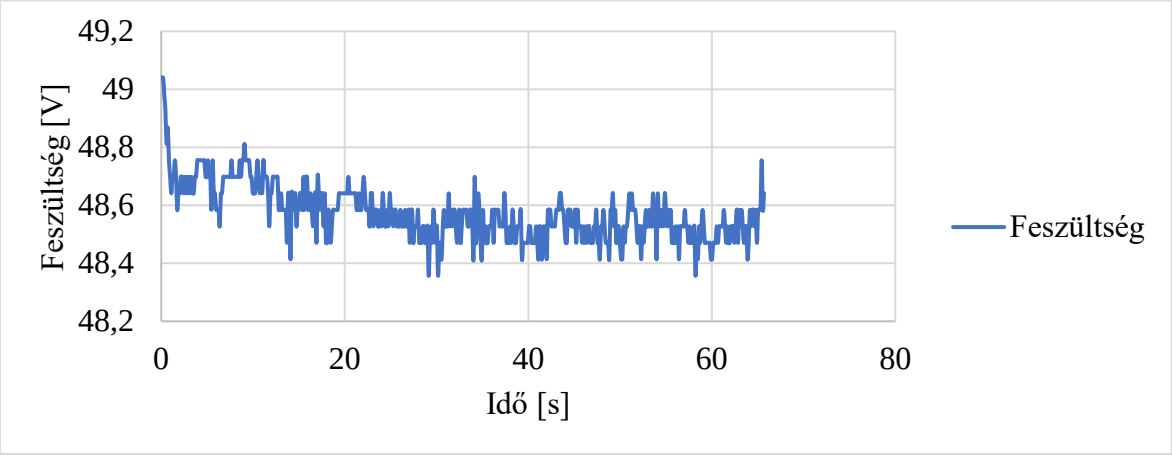
7.8. ábra: Mért pozíció X és Y koordinátái – targonca automata előremeneti mozgás

A hátrameneti mozgás megtervezése során a targonca pálya- és trajektóriatervezése 2 szegmensre épül:

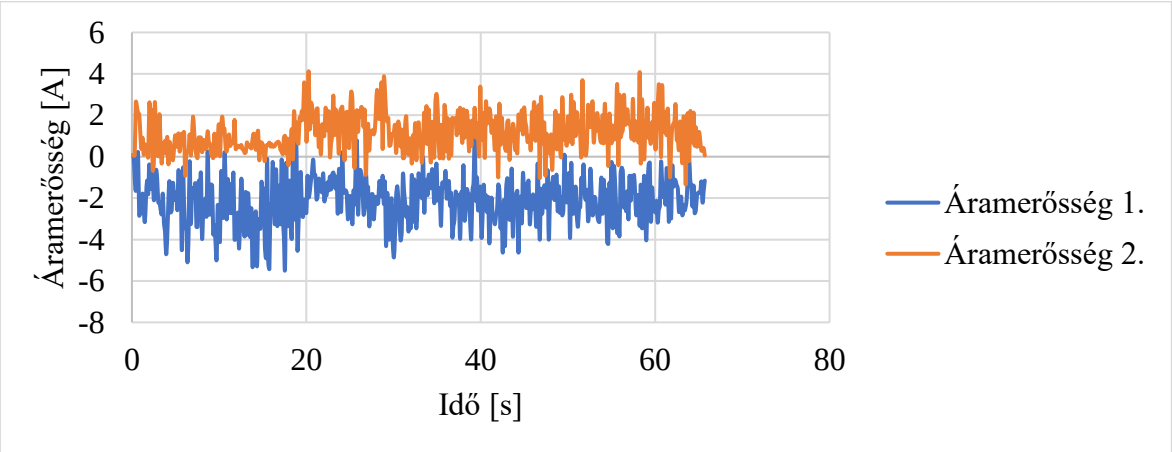
1. szegmens: A mért $\boldsymbol{P}^1_{start} = [X^1_{start}; Y^1_{start}]$ pozícióból indult a $\boldsymbol{P}^1_{end} = [X^1_{end}; Y^1_{end}] = [1.5m; 0m]$ pozícióba és $\varphi^1_{end} = 0°$ orientációba,
2. szegmens: A 1. szegmens végéről a $\boldsymbol{P}^2_{end} = [X^2_{end}; Y^2_{end}] = [0m; 0m]$ pozícióba és $\varphi^2_{end} = 0°$ orientációba folytatta útját.

A mérés kezdetén a targonca a $\boldsymbol{P}^1_{start} = [X^1_{start}; Y^1_{start}] = [-0.007m; 3.958m]$ pozícióból és $\varphi^1_{start} = 353.628°$ orientációval indult el.

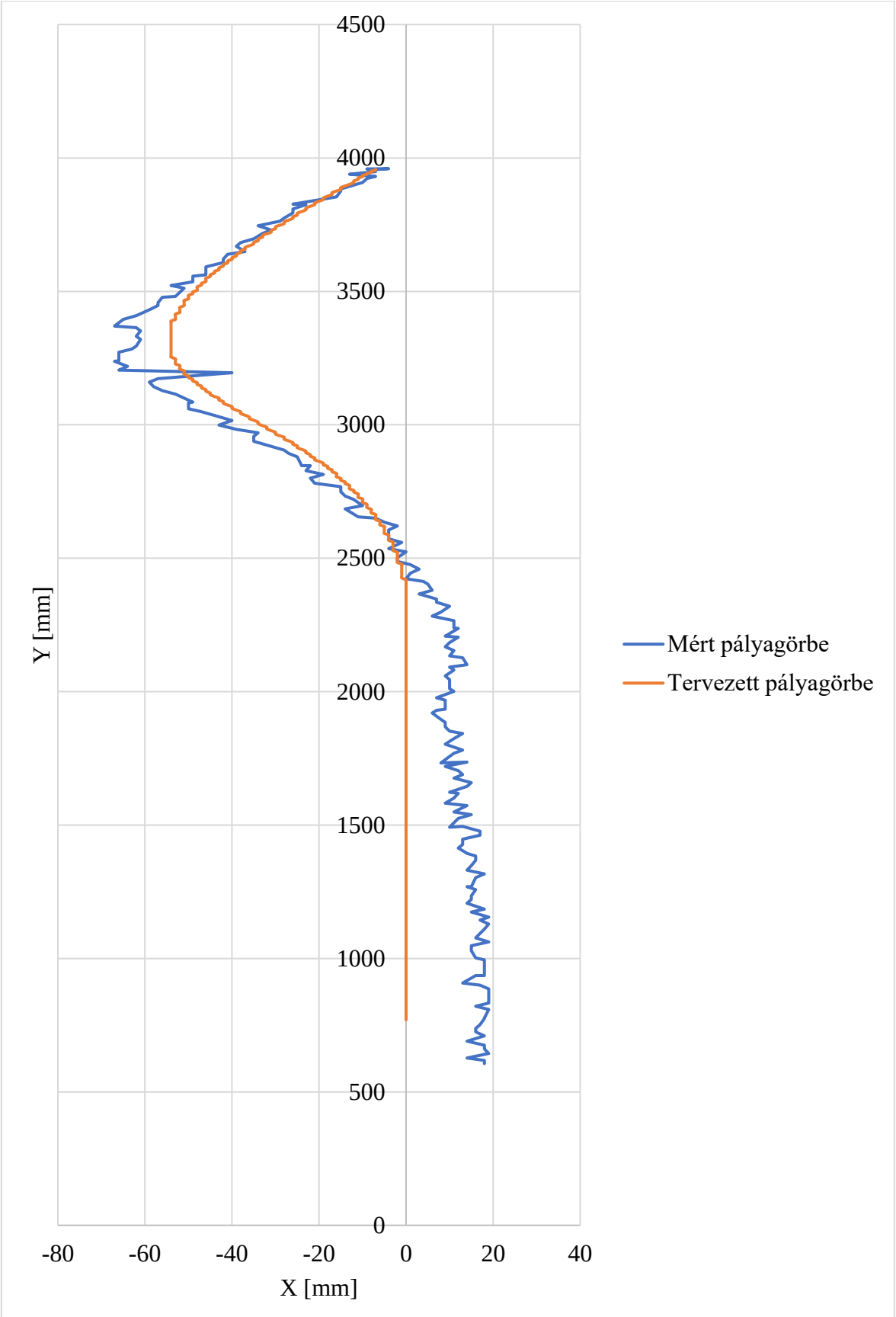
A hajtóakkumulátorok feszültségét az idő függvényében a 7.9. ábra, míg a két szervomotor által felvett áramerősség értékeket az idő függvényében a 7.10. ábra, az AGV tervezett és megvalósított útját X-Y koordinátarendszerben a 7.11. ábra szemlélteti. A feszültség függvénye ismét oszcilláló képet mutat, tendenciájában kissé csökkent. A 7.10. ábrán a két függvény lefutása a pálya utolsó kétharmadában közel szimmetrikusnak mondható. A targonca pályakövetése átmeneti hibanövekedés után csökken, majd enyhén növekvő tendenciát mutat.



7.9. ábra: Mért feszültség – targonca automata hátrameneti mozgás



7.10. ábra: Mért áramerősségek – targonca automata hátrameneti mozgás



7.11. ábra: Mért pozíció X és Y koordinátái – targonca automata hátrameneti mozgás

8. TÉZISEK – ÚJ TUDOMÁNYOS EREDMÉNYEK

Az alábbiakban kerülnek megfogalmazásra azok a tézis értékű új tudományos eredmények, amelyek kidolgozása a doktori kutatómunka része volt:

T1. Kidolgoztam egy olyan új pályatervező megoldást, amely egyszerre két módszerrel (Hermite-görbe és Bezier-görbe) határozza meg egy két szállítószalagos vezető nélküli szállítótargonca mozgásának vezérléséhez szükséges pályapontokat a kezdeti- és cél pozíciók és -orientációk felhasználásával.
Vonatkozó publikáció: [S1]

T2. Kidolgoztam egy olyan új trajektóriatervező megoldást, amely egyszerre két módszerrel is (Hermite-görbe és Bezier-görbe) meghatározza egy két szállítószalaggal rendelkező vezető nélküli szállítótargonca hajtott kerekeinek sebességét a pályatervező pályapontok geometriai és idő adatai alapján.
Vonatkozó publikáció: [S1]

T3. A trajektóriatervező kimeneteként adódó kerék-sebességadatok felhasználásával kialakítottam egy olyan új, moduláris rendszert, amely előállítja a pálya szimulációját és a hajtáshoz szükséges áramerősséget, valamint az adatokat (sebességek, fordulatszámok, feszültségek, áramerősségek, pozíciók és szögelfordulás) továbbít a főprogram felé. A DC motor elektrodinamikai modelljében a terhelőnyomaték értékét a targonca adottságaihoz illesztettem.
Vonatkozó publikációk: [S3], [S5], [S9]

T4. A targonca vezérléséhez szükséges paraméterek meghatározásához egy olyan új mérőrendszert fejlesztettem ki, amely fogadja a targonca navigációs adatait, méri a motorok áramerősségeit és a sorba kötött akkumulátorok feszültségét, és transzformálja a vezérlőrendszer számára. A mérőrendszer felhasználásával egy olyan vezérlővezérlőrendszert alakítottam ki, amely a kisebb áramfelvétellel járó utat választva vezérli a targonca mozgását.
Vonatkozó publikációk: [S2], [S4], [S6]

9. ÖSSZEFOGLALÁS

A doktori disszertáció egy vezető nélküli targonca mozgástervezésével, -vezérlésével és szimulációjával foglalkozott. Az értekezés áttekintette a kutatáshoz kapcsolódó szakirodalmat, kitérve a pálya- és trajektóriatervező módszerekre példákon keresztül. Továbbá megvizsgáltam az interpolációs és approximációs módszereket, és ezek közül bemutattam két az AGV vezérlésére jól alkalmazható módszert.

A doktori kutatás egyik célkitűzése volt egy olyan új pálya- és trajektóriatervező módszer megvalósítása, amely a legkisebb áramfelvételre törekszik a két alkalmas tervezési módszerre alapozva.

Az áramfelvétel meghatározásához a DC motor elektrodinamikai modelljét alkalmaztam. A modellben szereplő terhelőnyomatékot két részre osztottam fel és részleteztem ezek meghatározását. A gördülési ellenállásból adódó ellennyomaték meghatározásához méréseket végeztem, amely kiértékelésével meghatároztam a gördülési tényezőt.

Az elektrodinamikai modell kimeneteként adódó szögsebesség adatokat pályaszimulációra használtam fel és ismertettem az adatok továbbítására szolgáló kommunikációs modult.

A disszertációm következő része a targonca vezérléséhez szükséges programfejlesztéseket tárgyalta, amely a manuális és automatikus mozgatást valósítja meg.

Az értekezésem további részében az AGV állapotfelügyeletére mérési rendszert fejlesztettem ki. A mérési rendszer tartalmazza a navigációs adatok fogadását, a feszültség- és áramerősség mérését, majd ezen adatok transzformálását a PC-n futó vezérlési program számára. A mérések során kitértem a motor fordulatszámát és AGV pozícióját érintő megállapításokra, valamint a pályavezérlést is használó automata mozgás eredményeire.

A továbbfejlesztési terveim között szerepel a targonca vezérlésének ipari igényeket is kielégítő feladatainak szimulációja és megvalósítása. A további fejlesztéseknél fontosnak tartom az Ipar 4.0 technológia elvárásainak való megfelelést.

9.1. Summary

The doctoral dissertation dealt with the motion planning, control, and simulation of a driverless carrier vehicle. The dissertation reviewed the literature related to the research, covering path and trajectory planning methods through examples. Furthermore, I examined the interpolation and approximation methods and presented two of these methods that are well applicable to AGV control.

One of the objectives of the doctoral research was to implement a new path and trajectory planning method that strives for the lowest current consumption based on the two suitable planning methods.

To determine the current consumption, I used the electrodynamic model of the DC motor. I divided the load torque in the model into two parts and detailed their definition. To determine the counter-torque resulting from the rolling resistance, I performed measurements and by evaluating I determined the rolling factor.

I used the angular velocity data resulting from the output of the dynamic model for path simulation and described the communication module for data transmission.

The next part of my dissertation discussed the software developments needed to control the AGV, which implements manual and automatic movement.

In the rest of my dissertation, I developed a measurement system for AGV condition monitoring. The measurement system includes receiving the navigation data, measuring the voltage and current, and then transforming this data for the control program running on the PC. During the measurements, I covered the findings concerning the motor rotational speed and AGV position, as well as the results of the automatic movement using the track control.

My development plans include the simulation and implementation of AGV control tasks that also meet industrial needs. For further developments, I consider it important to meet the requirements of Industry 4.0 technology.

KÖSZÖNETNYILVÁNÍTÁS

Elsőként szeretnék köszönetet mondani témavezetőmnek, Dr. Szabó Tamás ny. egyetemi docensnek, volt intézeti tanszékvezetőnek, aki sok építőjellegű megjegyzést, segítséget nyújtott a BSc, MSc és 4 éves doktori képzésem során, és akivel számos közös publikációt készíthettem. A mechatronika iránti szeretetemet neki köszönhetem, hiszen már alapszakos hallgatóként lehetőség nyílt bekapcsolódni a tudományos élet világába.

Köszönet illeti a Robert Bosch Mechatronikai Intézeti Tanszék munkatársait, különösképpen Dr. Rónai László intézeti tanszékvezető egyetemi adjunktust és Lénárt József tanársegédet, a szakmai és erkölcsi támogatásukért.

Köszönet illeti még a Logisztikai Intézet munkatársait is, amely Intézethez tudományos segédmunkatársként volt lehetőségem csatlakozni 2019 decemberében. Itt kiemelném Dr. habil. Tamás Péter intézetigazgató egyetemi docenst, aki munkahelyi vezetésével és támogatásával hozzájárult a doktori munkám sikeréhez. Köszönetet szeretnék még kifejezni Dr. habil. Bányai Tamás és Dr. Telek Péter egyetemi docenseknek, akik irodatársukként folyamatosan segítettek a mindennapi munkámban és a doktori munkához is adtak értékes tanácsokat.

Ezúton szeretném megköszönni az egykori Fráter György Katolikus Gimnázium és Kollégium tanárainak, akik szilárd alapot adtak a kezembe az egyetemi tanulmányaim megkezdéséhez. És végül, de nem utolsó sorban szeretnék köszönetet mondani Szüleimnek, akik mindvégig, minden tekintetben támogattak és kitartottak a doktori pályám során.

Értekezésemet Nagyapámnak, Haraszti Rezsőnek ajánlom, aki a Miskolci Egyetem jogelődjének, a Nehézipari Műszaki Egyetem Ábrázoló Geometriai Tanszékének oktatója volt egyetemista korától nyugdíjazásáig. Az ő emléke örökké bennem él, bizonyára büszke lenne, ha tudná, hogy az oktatói pályát követtem én is ugyanazon az egyetemen.

A disszertációban ismertetett kutató munka az EFOP-3.6.1-16-2016-00011 jelű „Fiatalodó és Megújuló Egyetem – Innovatív Tudásváros – a Miskolci Egyetem intelligens szakosodást szolgáló intézményi fejlesztése” projekt részeként – a Széchenyi 2020 keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

IRODALOMJEGYZÉK

[1] *Matyi, H., Veres, P., Banyai, T., Demin, V.; Tamas, P.:* **Digitalization in Industry 4.0: The Role of Mobile Devices**, Journal of Production Engineering, Vol. 23, No. 1, pp. 75-78. (2020)

[2] *He, W.:* **Production allocation technologies for industrial product assembly lines based on the Internet of Things**, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp. 158-163. (2020)

[3] *Liu, W.:* **Production scheduling and equipment matching of flexible workshops based on multi-objective and multi-process hybrid optimization algorithm**, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp. 151-157. (2020)

[4] *Sujová, E., Vysloužilová, D., Čierna, H., Bambura, R.:* **Simulation Models of Production Plants as a Tool for Implementation of the Digital Twin Concept into Production**, Manufacturing Technology, Vol. 20, No. 4, pp. 527-533. (2020)

[5] *Sujová, E., Střihavková, E., Čierna, H.:* **An Analysis of the Assembly Line Modernization by Using Simulation Software**, Manufacturing Technology, Vol. 18, No. 5, pp. 839-845. (2018)

[6] *Tamás, P.:* **Examining the Possibilities for Efficiency Improvement of SMED Method Using Simulation Modelling**, Manufacturing Technology, Vol. 17, No. 4, pp. 592-597. (2017)

[7] *Ulewicz, R., Mazur, M.:* **Economic Aspects of Robotization of Production Processes by Example of a Car Semi-trailers Manufacturer,** Manufacturing Technology, Vol. 19, No. 6, pp. 1054-1059. (2019)

[8] *Benotsmane, R., Dudás, L, Kovács, Gy.:* **Simulation and trajectory optimization of collaborating robots by application of Solidworks and Matlab software in Industry 4.0**, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp. 191-197. (2020)

[9] *Wang, X., Gao, J.*: **An AGV scheduling algorithm for smart workshops with limited logistics capacity**, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp. 23-27. (2020)

[10] *Rónai, L., Szabó, T.:* **Snap-fit Assembly Process with Industrial Robot Including Force Feedback**, Robotica, Vol. 38, No. 2, pp. 317-336. (2020)

[11] *Edwards, M.:* **The Difference Between AGVs and Mobile Robots**, Cross Company (2016), https://www.crossco.com/blog/difference-between-agvs-and-mobile-robots

[12] *Müller, T.:* **Automated Guided Vehicles**, IFS (Publications) Ltd, UK, Springer-Verlag, Berlin, Heidelberg, New York, ISBN 3-540-12629-5 (1983)

[13] *Hammond, G.:* **AGVs at work, Automated Guided Vehicle Systems**, IFS (Publications) Ltd, UK, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, ISBN 3-540-16677-7 (1986)

[14] *Gamma Digital Kft.:* **Vezető nélküli targonca** (Driverless Carrier), Budapest (2011)

[15] *Papp, Á., Szilassy, L., & Sárosi, J.:* **Navigation of differential drive mobile robot on predefined, software designed path**. Recent Innovations in Mechatronics (RIiM), Vol. 3, No. 1-2, pp. 1–5. (2016)

[16] *Skapinyecz, R., Illés, B.:* **Introduction of the High-Tech Logistics Laboratory installed at the Institute of Logistics of the University of Miskolc**, COMEC, Kuba (2019)

[17] *Péter Tamás; Tamás Bányai; Béla Illés; Sándor Tollár; Péter; Ákos Cservenák; Ibolya Hardai; Róbert Skapinyecz:* **Development Possibilities of the High-tech Logistics Laboratory Established at the Institute of Logistics of the University of Miskolc,** JOURNAL OF ENGINEERING RESEARCH AND REPORTS, Vol. 13, No. 3, pp. 60-68. (2020)

[18] *Vale, A., Ventura, R., Lopes, P., Ribeiro, I.:* **Assessment of navigation technologies for automated guided vehicle in nuclear fusion facilities**, Robotics and Autonomous Systems (2017)

[19] *Petriu, E.M., Basran, J.S., Groen, F.C.A.:* **Automated guided vehicle position recovery**, IEEE Transactions on Instrumentation and Measurement, Vol. 39, No. 1, pp. 254–258. (1990)

[20] *Petriu, E.M.:* **Automated guided vehicle with absolute encoded guide-path**, IEEE Transactions on Robotics and Automation, Vol. 7, No. 4, pp. 562 – 565. (1991)

[21] *Cox, I.J.:* **Blanche-an experiment in guidance and navigation of an autonomous robot vehicle**, IEEE Transactions on Robotics and Automation Vol. 7, No. 2, pp. 193-204. (1991)

[22] *Francis, S. L. X., Anavatti, S. G., Garratt, M., Shim, H.:* **A ToF-camera as a 3D Vision Sensor for Autonomous Mobile Robotics,** International Journal of Advanced Robotic Systems, Vol. 12, No. 11, pp. 1-15. (2015)

[23] *Kelly, A., Nagy, B., Stager, D., Unnikrishnan, R.:* **An Infrastructure-Free Automated Guided Vehicle Based on Computer Vision**, IEEE Robotics & Automation Magazine, pp. 24-34. (2007)

[24] *Bačík, J., Ďurovský, F., Biroš, M., Kyslan, K., Perduková, D., Sanjeevikumar, P.:* **Pathfinder – Development of Automated Guided Vehicle for Hospital Logistics**, IEEE Access, Vol. 5, pp. 26892-26900. (2017)

[25] *Jia, S.:* **Implementation of intelligent robot control algorithm based on visual servo control**, Academic Journal of Manufacturing Engineering, Vol. 18, No. 3, pp. 131-140. (2020)

[26] *Mou, H.:* **Research on the formation method of omnidirectional mobile robot based on dynamic sliding control**, Academic Journal of Manufacturing Engineering, Vol. 18, No. 2, pp. 148-154. (2020)

[27] *Gasparetto, A., Boscariol, P., Lanzutti, A., & Vidoni, R.:* **Motion and Operation Planning of Robotic Systems**. Springer (2015)

[28] *Ghafil, H. M., Jármai, K.:* **Optimization for Robot Modelling with MATLAB**, Springer Nature Switzerland AG (2020)

[29] *Raja, P., Pugazhenthi, S.:* **Optimal path planning of mobile robots: A review**, International Journal of Physical Sciences Vol. 7, No. 9, pp. 1314-1320. (2012)

[30] *Lozano-Perez, T.:* **Spatial planning: A configuration space approach**. IEEE Transactions on Computers, Vol. 100, No. 2, pp. 108-120. (1983)

[31] *Takahashi, O., Schilling, R. J:* **Motion planning in a plane using generalized Voronoi diagrams**. IEEE Transactions on Robotics and Automation, Vol. 5, No. 2, pp. 143-150. (1989)

[32] *Janchiv, A., Batsaikhan, D., Kim, B., Lee, W.G., Lee, S.-G:* **Time-efficient and complete coverage path planning based on flow networks for multi-robots**, International Journal of Control, Automation and Systems, Vol. 11, No. 2, pp. 369-376. (2013)

[33] *Zhu, D., Latombe, J.-C.:* **New Heuristic Algorithms for Efficient Hierarchical Path Planning**, IEEE Transactions on Robotics and Automation, Vol. 7, No. 1, pp. 9-20. (1991)

[34] *Min, H., Lin, Y., Wang, S., Wu, F., Shen, X.:* **Path planning of mobile robot by mixing experience with modified artificial potential field method**, Advances in Mechanical Engineering, Vol. 7, No. 11, pp. 1-17. (2015)

[35] *Fedele, G., D'Alfonso, L., Chiaravalloti, F., D'Aquila, G.:* **Obstacles Avoidance Based on Switching Potential Functions**, Journal of Intelligent and Robotic Systems: Theory and Applications, Vol. 90, No. 3-4, pp. 387-405. (2018)

[36] *Li, C., Wang, F., Zhao, L., Li, Y., Song, Y.:* **An improved chaotic motion path planner for autonomous mobile robots based on a logistic map regular paper**, International Journal of Advanced Robotic Systems, Vol. 10, No. 273, pp. 1-7. (2013)

[37] *Bohlin, R., Kavraki, L.E.:* **Path planning using Lazy PRM**, Proceedings-IEEE International Conference on Robotics and Automation, Vol. 1, pp. 521-528. (2000)

[38] *Gasparetto, A., Zanotto, V.:* **A new method for smooth trajectory planning of robot manipulators, Mechanism and Machine Theory**, Vol. 42, pp. 455–471. (2007)

[39] *Bobrow, J.E., Dubowsky, S., Gibson, J.S.:* **Time-optimal control of robotic manipulators along specified paths**, International Journal of Robotics Research, Vol. 4, No. 3, pp. 554–561. (1985)

[40] *Shin, K.G., McKay, N.D.:* **Minimum-time control of robotic manipulators with geometric path constraints**, IEEE Transactions on Automatic Control, Vol. 30, No. 6, pp. 531–541. (1985)

[41] *Lee, J.:* **A Dynamic Programming Approach to Near Minimum-Time Trajectory Planning for Two Robots**, IEEE Transactions on Robotics and Automation, Vol. 11, No. 1, pp. 160-164. (1995)

[42] *Constantinescu, D.:* **Smooth time optimal trajectory planning for industrial manipulators**, PhD disszertáció, The University of British Columbia, (1998)

[43] *Shiller, Z.:* **Time-energy optimal control of articulated systems with geometric path constraints**, Journal of Dynamic Systems, Measurement, and Control, Vol. 118, pp. 139-143. (1996)

[44] *Joonyoung, K., Sung-Rak, K., Soo-Jong, K., Dong-Hyeok, K.:* **A practical approach for minimum-time trajectory planning for industrial robots**, Industrial Robots: An International Journal, Vol. 37, No. 1, pp. 51-61. (2010)

[45] *Rubio, F., Valero, F., Sunyer, J., Cuadrado, J.:* **Optimal time trajectories for industrial robots with torque, power, jerk and energy consumed constraints**, Industrial Robot: An International Journal, Vol. 39, No. 1, pp. 92 – 100. (2012)

[46] *Wang, C.H., Horng, J.G.:* **Constrained minimum-time path planning for robot manipulators via virtual knots of the cubic B-Spline functions**, IEEE Transactions on Automatic Control, Vol. 35, No. 5, pp. 573-577. (1990)

[47] *Lin, C.S., Chang, P.R., Luh, J.Y.S.:* **Formulation and optimization of cubic polynomial joint trajectories for industrial robots**, IEEE Transactions on Automatic Control, Vol. 28, No. 12, pp. 1066-1073. (1983)

[48] *Wei, Y., Kim, D.:* **Reliable and energy-efficient routing protocol for underwater acoustic sensor networks**, 2014 International Conference on Information and Communication Technology Convergence (ICTC), Busan, pp. 738-743. (2014)

[49] *Kim, H., Kim, B.K.:* **Online minimum-energy trajectory planning and control on a straight-line path for three-wheeled omnidirectional mobile robots**, IEEE Transactions on Industrial Electronics, Vol. 61, No. 9, pp. 4771-4779. (2014)

[50] *Saramago S.F.P., Steffen Jr. V.:* **Dynamic Optimization for the Trajectory Planning of Robot Manipulators in the Presence of Obstacles,** Jorunal of the Brazilian Society of Mechanical Sciences, Vol. 21, No. 3, pp. 1-12. (1999)

[51] *Saramago, S.F.P., Jr. Steffen. V.:* **Optimal trajectory planning of robot manipulators in the presence of moving obstacles**, Mechanism and Machine Theory, Vol. 35, No. 8, pp. 1079-1094. (2000)

[52] Kim, H., Kim, B. K.: **Minimum-Energy Trajectory Planning and Control on a Straight Line with Rotation for Three-Wheeled Omni-Directional Mobile Robots**, IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Algarve, Portugália (2012)

[53] Kim, H., Kim, B. K.: **Minimum-energy Cornering Trajectory Planning with Self-rotation for Three-wheeled Omni-directional Mobile Robots**, International Journal of Control, Automation and Systems, Vol. 15, pp. 1-10. (2017)

[54] *Peidró, A., Gallego, J., Payá, L., Marín, J. M., Reinoso, Ó.:* **Trajectory Analysis for the MASAR: A New Modular and Single-Actuator Robot, Robotics**, MDPI, Vol. 8, No. 78, pp. 1-22. (2019)

[55] *Shruthi, C. M., Sudheer, A. P., Joy, M. L.:* **Optimal crossing and control of mobile dual-arm robot through tension towers by using fuzzy and Newton barrier method**, Journal of the Brazilian Society of Mechanical Sciences and Engineering, Vol. 41, No. 245, pp. 1-25. (2019)

[56] *Nazemizadeh, M., Rahimi, H.N. Amini Khoiy, K.:* **Trajectory planning of mobile robots using indirect solution of optimal control method in generalized point-to-point task**, Frontiers of Mechanical Engineering. Vol. 7, No. 1, pp. 23–28. (2012)

[57] *Gracia, L., Tornero, J.:* **Optimal Trajectory Planning for Wheeled Mobile Robots Based on Kinematics Singularity**. Journal of Intelligent Robot Systems, Vol. 53, pp. 145–168. (2008)

[58] *Suh, J., Gong, J., Oh, S.:* **Fast Sampling-Based Cost-Aware Path Planning With Nonmyopic Extensions Using Cross Entropy**, IEEE Transactions on Robotics, Vol. 33, No. 6, pp. 1313-1326. (2017)

[59] *Zhu, Y., Jin., B., Li, S:* **Optimal design of hexapod walking robot leg structure based on energy consumption and workspace**, Transactions of the Canadian Society for Mechanical Engineering, Vol. 38, No. 3, pp. 305-317. (2014)

[60] *Louste C., Liégeois, A.:* **Path planning for non-holonomic vehicles: a potential viscous fluid field method**, Robotica, Vol. 20, pp. 291-298. (2002)

[61] *Yan, Y., Mostofi, Y.:* **To Go or Not to Go: On Energy-Aware and Communication-Aware Robotic Operation**, IEEE Transactions on Control of Network Systems, Vol. 1, No. 3, pp. 218-231. (2014)

[62] *Capi, G., Nasu, Y., Barolli, L., Mitobe, K., Takeda, K.:* **Application of Genetic Algorithms for biped robot gait synthesis optimization during walking and going up-stairs**, Advanced Robotics, Vol. 15, No. 6, pp. 675-694. (2001)

[63] *Tokekar, P., Karnad, N., Isler, V.:* **Energy-optimal trajectory planning for car-like robots**, Auton Robot, Vol. 37, pp. 279–300. (2014)

[64] *Liu, S., Sun, D.:* **Minimizing Energy Consumption of Wheeled Mobile Robots via Optimal Motion Planning**, IEEE/ASME Transactions on Mechatronics, Vol. 19, No. 2, pp. 401-411. (2014)

[65] *Bhattacharya, S., Agrawal, S. K.:* **Spherical rolling robot: a design and motion planning studies**, IEEE Transactions on Robotics and Automation, Vol. 16, No. 6, pp. 835-839. (2000)

[66] *Piazzi, A., Visioli, A.:* **Global minimum-jerk trajectory planning of robot manipulators**, IEEE Transactions on Industrial Electronics, Vol. 47, No. 1, pp. 140-149. (2000)

[67] *Chen, C.*: **Numerical Methods, 6. Interpolation and Approximation**, tananyag, Department of Computer Science, National Tsing Hua University (2009)

[68] *Daniela Velichová:* **Constructive Geometry B – Lectures, 6. Approximation and Interpolation Curves**, tananyag, Slovak Technical University, Bratislava (2012)

[69] *Deuflhard, P., Hohmann, A.:* **Numerical Analysis in Modern Scientific Computing**, Springer-Verlag New York (2003)

[70] *Salomon, D.:* **Curves and Surfaces for Computer Graphics,** Springer-Verlag, Berlin, Heidelberg (2005)

[71] *Heath, M. T, Munson, E. M.:* **Scientific Computing: An Introductory Survey**, McGraw-Hill Higher Education (1996)

[72] *Lyche, T., Mørken, K.:* **Spline Methods, Draft**, Department of Informatics, Centre of Mathematics for Applications, University of Oslo (2008)

[73] *Jaklic, G., Kozak, J., Krajnc, M., Vitrih, V., Zagar, E.:* **Hermite geometric interpolation by rational Bézier spatial curves**, Society for Industrial and Applied Mathematics Journal on Numerical Analysis, Vol. 50, No. 5, pp. 2695–2715. (2012)

[74] *Mahaffy, J. M.:* **Math 541 - Numerical Analysis Interpolation and Polynomial Approximation — Piecewise Polynomial Approximation; Cubic Splines**, prezentációs tananyag, Department of Mathematics and Statistics Dynamical Systems Group Computational Sciences Research Center San Diego State University (2018)

[75] *Barbic, J.:* **CSCI 420 Computer Graphics Lecture 9, Splines**, prezentációs tananyag, University of Southern California (2019)

[76] *Hashemi-Dehkordi, S., Valentini, P.P.:* **Comparison between Bezier and Hermite cubic interpolants in elastic spline formulations**. Acta Mech, Vol. 225, pp. 1809–1821. (2014)

[77] *Mészáros, J.:* **Numerical Methods, II. Approximations of functions**, tananyag, Alkalmazott Matematikai Tanszék, Miskolci Egyetem (2011)

[78] *Li, H.:* **CSCI 420: Computer Graphics, 4.2 Splines,** prezentációs tananyag, University of Southern California (2014)

[79] *Lensch, H.:* **Computer Graphics, Splines**, prezentációs tananyag, Universität Tübingen (2007)

[80] *Forsyth, D. A.:* **Interpolating Curves**, prezentációs tananyag, Grainger College of Engineering Computer Science, University of Illinois (2013)

[81] *Eren, H., Fung, C., Evans, J.:* **Implementation of the spline method for mobile robot path control**, Conference Record - IEEE Instrumentation and Measurement Technology Conference 2., Vol. 2, pp. 739 – 744. (1999)

[82] *Abut T.:* **Modeling and Optimal Control of a DC Motor**, International Journal of Engineering Trends and Technology (IJETT), Vol. 32, No. 3, pp. 146-150. (2016)

[83] *Deb, P. B., Saha, O., Saha, S., Paul, S.:* **Dynamic Model Analysis of a DC Motor in MATLAB**, International Journal of Scientific & Engineering Research, Vol. 8, No. 3, pp. 57-60. (2017)

[84] *Okubanjo, A., Oyetola, O., Olaluwoye, O., Alao, O., Olateju, A., Abatan, T.:* **Modeling and Simulation of DC Motor Using Simelectronics and Simulink**, Gazi Mühendislik Bilimleri Dergisi, Vol. 5, No. 1, pp. 91-100. (2019)

[85] *Regulace – Automatizace Bor*: **Servo Motor HSM 150**, katalógus, Novy Bor, Csehország (2018)

[86] *Schramm, D., Hiller, M., Bardini, R.*: **Vehicle Dynamics, Modeling and Simulation**, Springer, ISBN 3-5403- 6044-1 (2014)

[87] *Égert, J. – Pere, B.*: **Mechanika, Statika, tankönyv**, Széchenyi István Egyetem, Győr (2006)

[88] *Wang, J., Liu, X. & Guo, W.:* **Analysis of mechanical parameters for asymmetrical strip rolling by slab method**, International Journal of Advanced Manufacturing Technology, Vol. 98, pp. 2297–2309. (2018)

[89] *El-Sayegh, Z., El-Gindy, M., Johansson, I., Öijer, F.:* **Improved tire-soil interaction model using FEA-SPH simulation**, Journal of Terramechanics, Vol. 78, pp. 53-62., ISSN 0022-4898 (2018)

[90] *ATMEL*:**8-bit AVR Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash**, datasheet, Rev. 8025I–AVR–02/09 (2009)

SAJÁT PUBLIKÁCIÓK DISSZERTÁCIÓ TÉMÁJÁBAN

[S1] *Ákos Cservenák:* **Path and Trajectory Planning for an Automated Carrier Vehicle Equipped with two Conveyor Belts used in Manufacturing Supply,** Manufacturing Technology, Engineering Science and Research Journal, Institute of Technology and Production Management University of J.E. Purkyne (2021), **Q2 minősítés, Scopus által indexált, elfogadva, megjelenés alatt**

[S2] *Ákos Cservenák:* **Creating voltage, current and navigation measurement system on an AGV for motion controlling,** ACADEMIC JOURNAL OF MANUFACTURING ENGINEERING, Editura Politehnica (2021 június), **Q3 minősítés, elfogadva, megjelenés alatt**

[S3] *Ákos Cservenák:* **Simulation of a mobile robot's motion,** ACADEMIC JOURNAL OF MANUFACTURING ENGINEERING, Editura Politehnica (2021 március), **Q3 minősítés, elfogadva, megjelenés alatt,** Hivatkozások száma: 1 | Független hivatkozás: 0 | Függő hivatkozás: 1

[S4] *Cservenák Ákos:* **Mérőrendszer, kommunikáció és adatfeldolgozás kialakítása vezető nélküli targoncán,** MULTIDISZCIPLINÁRIS TUDOMÁNYOK: A MISKOLCI EGYETEM KÖZLEMÉNYE, Vol. 11, No. 4, pp. 53-59. (2021)

[S5] *Ákos Cservenák:* **Simulation and modeling of a DC motor used in a mobile robot,** ACADEMIC JOURNAL OF MANUFACTURING ENGINEERING, Editura Politehnica, Vol. 18, No. 4, pp. 183-190. (2020 december), **Q3 minősítés**, Hivatkozások száma: 1 | Független hivatkozás: 0 | Függő hivatkozás: 1

[S6] *Cservenák Ákos:* **Vezető nélküli targonca mérési rendszerének fejlesztése,** Doktoranduszok Fóruma: Gépészmérnöki és Informatikai Kar Szekciókiadványa, Miskolc, Hungary, 2019.11.21

[S7] *Cservenák Ákos:* **Mobil robot mozgásának vezérlése,** MULTIDISZCIPLINÁRIS TUDOMÁNYOK: A MISKOLCI EGYETEM KÖZLEMÉNYE, Vol. 9, No. 4, pp. 438-443. (2019) Hivatkozások száma: 2 | Független hivatkozás: 1 | Függő hivatkozás: 1

[S8] *Cservenák Ákos:* **Vezető nélküli targonca gördülési ellenállásának vizsgálata,** Doktoranduszok fóruma 2018: Gépészmérnöki és Informatikai Kar szekciókiadványa, Miskolc, Hungary 2018.11.22, pp. 14-19. (2019)

[S9] *Cservenák Ákos; Szabó Tamás:* **Jármű gördülési ellenállásának meghatározása méréssel,** XXVII. Nemzetközi Gépészeti Konferencia OGÉT 2019 Oradea, Romania 2019.04.25. - 2019.04.28, pp. 79-82. (2019)

[S10] *Ákos Cservenák:* **Motion planning for Automated Guided Vehicle,** ACTA TECHNICA CORVINIENSIS – BULLETIN OF ENGINEERING, Vol. XI, No. 4, pp. 33-38. (2018), Hivatkozások száma: 1 | Független hivatkozás: 1 | Függő hivatkozás: 0

[S11] *Cservenák Ákos:* **Vezető nélküli targonca programozása,** Doktoranduszok Fóruma 2017: Gépészmérnöki és Informatikai Kar szekciókiadványa, Miskolc, Hungary 2017.11.16, pp. 13-17. (2018)

[S12] *Ákos Cservenák:* **Further development of an AGV control system,** LECTURE NOTES IN MECHANICAL ENGINEERING, pp. 376-384. (2018), **Q4 minősítés, Scopus által indexált**, Hivatkozások száma: 5 | Független hivatkozás: 4 | Függő hivatkozás: 1

Publikációkra való hivatkozások utoljára ellenőrizve: 2021. 03. 12.

ÁBRAJEGYZÉK

TÁBLÁZATJEGYZÉK