

Tartalomjegyzék

1. Bevezető	5
2. Bibliográfiai tanulmány	7
2.1. Robotok	7
2.2. Mobilis Robotok Helyváltoztatása	7
2.3. Kültéri mobilis robotok	7
2.4. Robot Operációs Rendszer	10
2.4.1. Uj robot integrációja ROS hoz	12
2.5. Mobilis robotok modellezése	15
2.5.1. Merőleges Nyomóerő (N)	15
2.5.2. Súlypont (X,Y) komponensenek a meghatarozása	18
2.5.3. Súlypont meghatározása mérésekkel	19
2.5.4. Kerekek Dinamikája	19
2.6. Kinematikai Modell	20
2.7. Dinamikai Modell	22
2.8. Robot Platform Sebesség Szabályzása	25
2.8.1. Eloirt nyomatekkal	25
2.8.2. Elirt kereksgzogsebessegekkel	26
3. A rendszer implementálása	27
3.1. Robot felepítése	27
3.2. Alacsony szintű modulok	27
3.2.1. Microblaze szoftvere	35
3.2.2. FPGA es UART alapú kommunikacios protokol	37
3.2.3. Paramterek FPGA modul	38
3.2.4. Kommunikáció sebessége	39
3.2.5. Biztonsagi megoldasok	40
3.3. ROS	41
3.3.1. Uzenet tipusok (.msg)	42
3.3.2. FPGA kommunikacios modul ROS oldali integracio	45
3.3.3. Előírt értékek	47
3.3.4. Vonatkoztatási Rendszerek	47
3.4. Kerekek Pid Szabalyzo hangolas	50
3.4.1. Kisebik fokozatban	51
3.5. Pályakövettesi feladatok	53
3.6. Meresek	53
3.6.1. Differenciális Forgás Vízszintes Talajon	54
3.6.2. Feloldali kerekek blokolva kavicsos talajon	54
3.6.3. Kavicsos talajon helyben forgás	57

3.6.4.	Kavicsos talajon korpalyan mozgas	59
3.6.5.	Kavicsos talajon korpalyan 7 5	59
3.6.6.	Korpalya 7 3 Kavicsos talajon	64
3.6.7.	Homokos Lejtodfdsfds	68
3.6.8.	Lepcson	71
3.6.9.	Ismeretlen terep terkepezese es robot lokalzalasa (SLAM)	76
4.	Eredmények Kiértékelése	77
4.1.	Megvalósítások	77
4.2.	Hasonló rendszerekkel való összehasonlítás	77
4.3.	További fejlesztési irányok	77

Ábrák jegyzéke

2.1.	iRobo 510 lanctalpas packbootForrás:	8
2.2.	Négy kerekű mobilis platform.Forrás:	8
2.3.	ecorobotix mezőgazdasági robotForrás:	9
2.4.	Spirit nevu Marsjáró robot.Forrás:	10
2.5.	ROS kommunicacios mechanizmus szervizhivasokraForrás:	11
2.6.	ROS kommunicacios mechanizmus adatfolyamokraForrás:	11
2.7.	Ros Control modulokForrás:	13
2.8.	Merőleges nyomóerő a talajra $4W - SSMR$ típusú robot esetében.	16
2.9.	$4W - SSMR$ típusú robot lejtőn felfele oldal nézetből.	16
2.10.	$4W - SSMR$ típusú robot lejtőn első nézetből.	17
2.13.	Kinematikai modell az $4W - SSMR$ típusú robotnak.	21
2.14.	Kinematikai modell az $SSMR$ típusú <i>MR</i> robotnak.	23
2.15.	Kinematikai modell az $SSMR$ típusú <i>MR</i> robotnak.	25
2.16.	Kinematikai modell az $SSMR$ típusú <i>MR</i> robotnak.	26
3.1.	CmodA7 FPGA-ban kialakított architektúra amely a szenzorok és motor hajtások kezelésre hivatott	28
3.2.	FPGA ban megvalositott kommunikacios protokol a 3.2.2 leírt protokol alapjan.	29
3.3.	FPGA hardver/MicroBlaze proceszor es ROS node közti komunikacio megvalistiasa UART protokol alapjan	30
3.4.	FPGA ban megvalositott szoftproceszor rendszer, legfelso negyzet.	31
3.5.	FPGA ban megvalositott szabalyzok A es B	33
3.6.	FPGA controller mag.	34
3.7.	MicroBlaze proceszoron futó szoftver diagramja	36
3.8.	FPGA komunikacios protokol altalanos csomag szerkezet	37
3.9.	Robothoz csatlakozas a Wifi-n keresztul.	41
3.10.	ROS graph	44
3.11.	ROS integrálása Uart protokolhoz.	46
3.12.	A megvalositott robot VNR-k közti reláció	49
3.13.	Nagy fokozat Hammerstein-Wiener becsult model valasza, es a mert ertekek.	51
3.14.	Kis fokozat Hammerstein-Wiener becsult model valasza, es a mert ertekek.	52
3.15.	$SSMR - 4W$ típusú robot pozicioja, X es Y tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/s$	54
3.16.	$SSMR - 4W$ típusú robot altal leírt palya, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/s$	55
3.17.	$SSMR - 4W$ típusú robot orientacioja, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/s$	55
3.18.	$SSMR - 4W$ típusú robot fordulasi szogsebessege, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/s$	56

3.19. <i>SSMR – 4W</i> típusú robot egyenesvonalú sebessegei, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	56
3.20. <i>SSMR – 4W</i> típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	57
3.21. <i>SSMR – 4W</i> típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=-50$ es a $FR=BR= 50^\circ/s$	58
3.22. <i>SSMR – 4W</i> típusú robot által leírt palya, kereksebessegek $BL=FL=-50$ es a $FR=BR= 50^\circ/s$	58
3.23. <i>SSMR – 4W</i> típusú robot orientációja, kereksebessegek $BL=FL=-50$ es a $FR=BR= 50^\circ/s$	59
3.24. <i>SSMR – 4W</i> típusú robot fordulási szögsebessege, kereksebessegek $BL=FL=-50$ es a $FR=BR= 50^\circ/s$	59
3.25. <i>SSMR – 4W</i> típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	60
3.26. <i>SSMR – 4W</i> típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	60
3.27. <i>SSMR – 4W</i> típusú robot által leírt palya, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	61
3.28. <i>SSMR – 4W</i> típusú robot orientációja, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	61
3.29. <i>SSMR – 4W</i> típusú robot fordulási szögsebessege, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	62
3.30. <i>SSMR – 4W</i> típusú robot egyenesvonalú sebessegei, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	63
3.31. <i>SSMR – 4W</i> típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	64
3.32. <i>SSMR – 4W</i> típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	65
3.33. <i>SSMR – 4W</i> típusú robot által leírt palya, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	65
3.34. <i>SSMR – 4W</i> típusú robot orientációja, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	66
3.35. <i>SSMR – 4W</i> típusú robot fordulási szögsebessege, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	66
3.36. <i>SSMR – 4W</i> típusú robot egyenesvonalú sebessegei, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$	67
3.37. Kulombozo korpalyak	68
3.38. Homokos domb 1	69
3.39. Homokos domb 1	69
3.40.	70
3.41.	71
3.42.	72
3.43.	73
3.44.	74
3.45.	75
3.46. Terkep készítése miközben tavirányitassal halad a robot.	76

fejezet 1

Bevezető

A robotokat széles körben alkalmazzák, egyre több feladatra és most már az átlag emberek életében is. Néhány nagyobb vállat mint pl: ABB, Kuka nagy területet foglalt el az iparban robot karok gyártásában. Emellett egyre több kisebb vállalat jelent meg, amelyek háztartási vagy fél katonai eszközöket gyártanak pl.: iRobo. A mezőgazdaságban is próbálnak alkalmazni autonóm robotokat pl.: echorobotics amelyek segítségével hatóanyag mentes élelmiszereket állíthatnak elő.

A dolgozat célja hogy felderítse az aktuális legmodernebb technikákat amelyeket robotokon alkalmaznak, és ezeket alkalmazza egy robot kulteri mobilis robot megepitése során. Az elozo mar egy hasonló rendszer kivitelezésékor elkeszítet modulok továbbfejlesztése és integralása az új rendszerbe. A robot mozgása negy csigaatettel és negy DC motor segítségevel valósul meg. A motrok szögsebesseget és általuk felvett aramokat merjük inkrementális szenzor és elektromagneses jelensegre alapuló aramero modul segítségevel.

A beavatkozó jelet feszültseg formájában tortenik, amelyet H-hiddal alítunk elő azáltal hogy PWM jelet generalunk.

Az inkrementális és aramero szenzorok jeleit FPGA alau fejlesztőlapokkal tortenik. A roboton helyetfoglal egy kis meretu számítógép is amely USB vezetékken keresztül csatlakozik az FPGA lapokhoz és a robotn található más szenzorokhoz pl: LIDAR, IMU, GPS..

A robot a csigaatletek miatt nagy nyomateket tud elolandani, ami a mozgási sebesseg rovasara vált. A robot képes 0.3 m/s sebessegel mozogni előre, és 20 °/s forgási sebesseget generalni súlypontja korul.

A dolgozat célja az elkeszített kulteri mobilis robottal mereseket vegezni különböző terepviszonyok között és azokat oszehasonlitani. A terepviszonyok között említhető pl: fűves, kavicsos, aszfalt, marvány stb. A mereseket szeretnék összehasonlitani a terepviszonyok fugvenyében: a robot mozgasat és az ehez szükseges energia nagyságát, a környezetben okozott behatásokat.

A robot SLAM algoritmus segítségevel dolgozza fel a LIDAR által mert adatokat és egy 2D terkepet állít elő, amelyen képes egyidőben behatrolnia pozícióját és a terkepet építeni is. Ezen meglevo programok használatával kulteri mereseket vegezni, terkepeket készíteni és majd egy előiről párban vegigmenni ezen terepeken a terkepet használva.

fejezet 2

Bibliográfiai tanulmány

2.1. Robotok

A olyan gépek amelyeket arra terveztek hogy bizonyos feladatokat automatikusan elvégezzenek gyorsabban és pontosabban mit az emberek. Manapság már sok tipusú robot létezik, ezek közül néhány repül, földön gurul vagy maszik, de léteznek már emberhez hasonlók is.

A robotok legelőször a második világháború alatt jelentek meg mint irányított bombák. A háború után W.Grey Walter neurologus fejlesztett ki egy kismeretű robotot (Elmer and Elsie) mely fényszenzorral, és nyomasszenzorral volt elátható. 1961-1963 Johns Hopkins Beast robot képes volt a fal menten végigmenni és megtalálni a toltoalomast. 1970 -ben Shakey the robot képes volt kamera segítségevel egy vonalat követni. A 1990 után a fejlesztések felgyorsultak és azóta már robot jár a naprendszerünk számos bolygojan.

Ezen dolgozat csak a **MR** típusú robotokkal fog részletesen kitterni. A **MR** típusú robotok képesek a saját pozíciójukat megvaltoztani a környezetükhez képest. Az **AMR** típusú robotok képesek saját környezetük felderítésére ismeretlen környezetben is kulombözo szenzorokat használva melyekkel képesek a környezetük paramtereinek a megmérése. Az **AMR** típusú robotokat manapság leginkább taroló helyszégeken használják ahol anyagokat palettákat vagy kulombözo dolgokat kell szállítani pl(Mobile Industrial Robots ApS valalat fejlesztései).

2.2. Mobilis Robotok Helyváltoztatása

A **MR** kerekekkel, hártyatalpakkal, vagy labakkal képesek helyüket megvaltoztatni. A leginkább használatos és robosztusnak bizonyult kulteri terepen a kerekek és a herbyatalpák. Legismertebb robot felelősek: packboot pl: iRobo510, negy kerek pl: husky, negy kormányozható kerek, hat kerek amiből negy kormányozható pl: Curiosity Mars Rover. Az utóbbi esetben a kerek fogalma is ujraertelmeződöt, a hagyományos kerekek pl: RHex Rough-Terrain amely a kerek és a lebak keveréke.

2.3. Kültéri mobilis robotok

iRobo510

A robot kulteren és belteren is egyaránt használható, felderítésékre és kisebb beavatkozásra alkalmas a manipulator kart használva. A robotot 2007-ben dobott ki a pacra. A katonaság használta bombák hatallanítására vagy felderítésre.



2.1. ábra. iRobo 510 lanctalpas packbootForrás:

Tulajdonság	Mértékegység		
Méretek	X	0.521	m
	Y	0.686	m
	Z	0.178	m
Önsúly	10.89		kg
Max Sebesség	9.3		km/h
Gyorsulásmérő és Giroszkóp	Igen		

Husky

Nagy teher birasu kulteri mobilis robot, leginkább kutatási célokra alkalmas, nagy kereknnyomateka miatt nehezterepen is jól boldogul. Nagyfelbontasú inkrementális enkoderekkel szereltek fel, alacsony mozgási sebeségére is képes. Működése 3 óra átlagos használat melett, támogatja teljes mértékben ROS-t.



2.2. ábra. Négy kerekű mobilis platform.Forrás:

Tulajdonság	Mértékegység		
Méretek	X	0.99	m
	Y	0.67	m
	Z	0.39	m
Önsúly	50 + 75		kg
Max Sebesség	3.6		km/h
Gyorsulásmérő és Giroszkóp	Igen		

Ecorobotix

Mezőgazdaság számára kifejlesztett robot, négy kerékkel rendelkezik amelyek közül kettő motort hajt, és a másik kettő ön-beálló kerék. Kamera segítségével ismeri fel a növényeket és a pozíciójukat, és ha szükséges akkor be is avatkozik.

A localizációra egy nagy pontosságú GPS RTS alkalmaznak.



2.3. ábra. ecorobotix mezőgazdasági robotForrás:

Tulajdonság	Mértékegység		
Méretek	X	2.2	m
	Y	1.70	m
	Z	1.30	m
Önsúly		130	kg
Max Sebesség		1.4	km/h
Gyorsulásmérő és Giroszkóp	Igen		

Spirit

Marsi körülményekre tervezett robot 6 kerekkel rendelkezett amelyből 4 kormányzott négy sztereó kamerával elátva, 30° lejtőt volt képes megmaszní. Működési ideje 6 év és 2 hónap volt a Marson, a végen homokba süllyedve egy sziklára akadva érte a marsi tél, amely ahhoz vezetett hogy a nap elemei nem szolgáltattak elegendő energiát és így végleg leált és kihült.



2.4. ábra. Spirit nevu Marsjáró robot. Forrás:

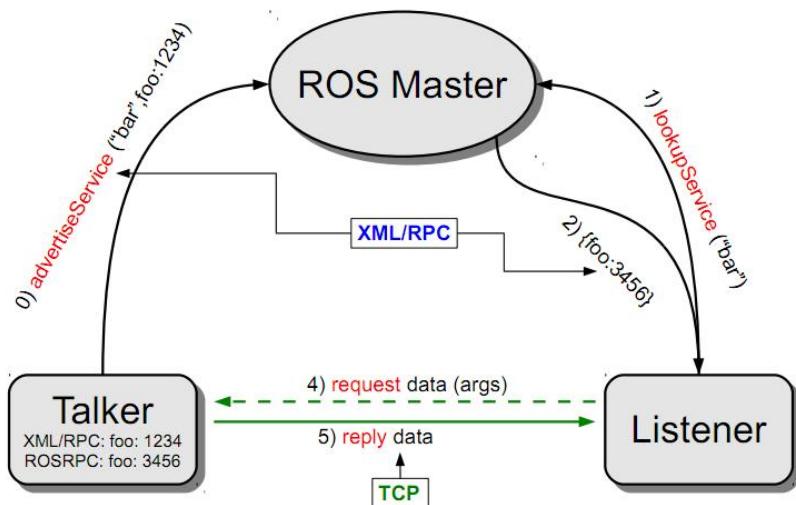
Tulajdonság		Mértékegység
Méretek	X	1.6 m
	Y	2.3 m
	Z	1.5 m
Önsuly	35 (felszereléssel 180)	kg
Max Sebesség	0.05(avg 0.01)	km/h
Gyorsulásmérő és Giroszkóp	Igen	

2.4. Robot Operációs Rendszer

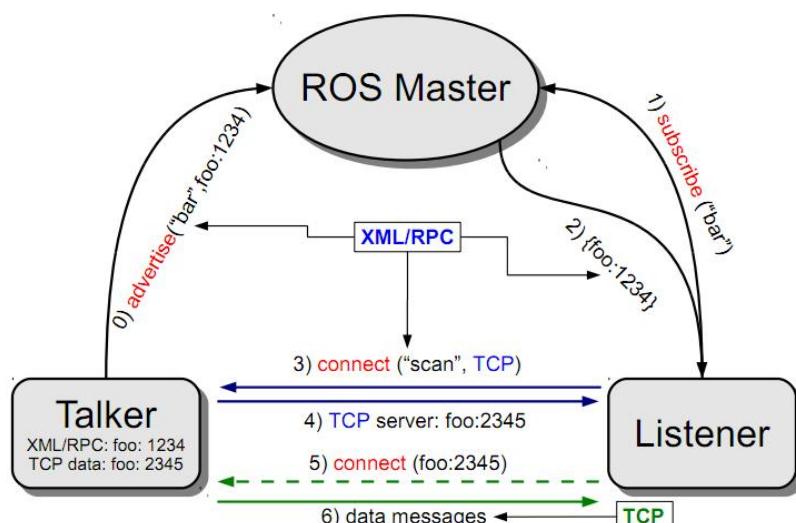
A ROS 2007-ben jelent meg, gyorsan elterjedt az egész világon, manapság szinte minden robotokkal foglalkozó cég termékeit kapcsolódnak a ROS-hoz.

A ROS mukodesehez szukseges egy gazda operacio rendszer a UNIX vagy Windows alapu amelyen a kozponti node fut (rosmaster), kezeli a tobbi node kozti kommunikaciót, parametereket, szervizhivasokat.

A kommunikacio a nodok kozolt TCP protokolra epül, amely XML/PRC teknoloiat hasznal, RPC távoli eljárashívást jelent [8]. minden node jelzi a rosmasternek milyen adatokat szeretne mgosztani ezeket az advertise fugvenyhivasok jelzik. A nodok ugyanakkor feliratkozhatnak ezeket a subscribe fugvenyekkel valosithatjuk meg pl.: [23]. A szervizhivasok 5 lepeses folyamatbol alnak, lathato a ábra 2.4.5 valamint az adatfolyamok 8 lepesbol ábra 2.4.6.



2.5. ábra. ROS kommunicacions mechanizmus szervizhivasokraForrás:



2.6. ábra. ROS kommunicacions mechanizmus adatfolyamokraForrás:

A [12] [14] segítségével megalapozhatjuk a tudásunkat. Számos előnyel rendelkezik a ROS használata egy új robot fejlesztésében mert már elkészített csomagokat használhatunk pl: SLAM ¹, AMCL, vagy előre elkészített eszközök segítenek a fejlesztésben pl: Rviz ², Gazebo ³, interfész biztosít a szenzoroknak és beavatkozóknak pl: LIDAR, IMU .

A [12] említést tesz arról hogy más hasonló keret-rendszerekhez képest a ROS stabilabb pl: ha egy modulban futás-idejű hiba lép fel az nem terjed ki a többi csomópontra. Egyeszerűbb fejlesztést lehetséges azáltal hogy kisebb modulokat fejlesztünk és nem egy nagy több szálra futó kódot. Annak ellenére hogy a forrás-kódja nyílt a keret-rendszernek nagyon jó szupportja van, rengeteg fórumon keresztül kaphatunk megoldásokat az esetleges hibákra. Több teknologiát képes összekapcsolni mint pl.: tensorflow, matlab, simulink, opencv..., V-Rep

¹ Simultaneous Localization and Mapping
egyidejű térképezés és lokalizáció

² ROS környezet vizualizációs eszköze

³ ROS környezet szimulációs eszköze

Hátrányai között említhető a Gazebo szimulátor mert a használata nem egyszerű el-lentében a V-Rep⁴ programmal. A robot modellezése nem egyszerű dolog, *URDF* fájl szükséges hozzá, csak SolidWork⁵ biztosít lehetőséget arra hogy modellt exportálhassuk.

2.4.1. Uj robot integrációja ROS hoz

Egy új robot integrációja során meg kell vizsgálni hogy milyen mérési adatok állnak rendelkezésünkre alacsony szinten, a rotációs csukló paraméterek lehetnek pl: szög pozíció, szög sebesség, kifejtett, nyomaték, ezeket a paramétereket mérhetjük, illetve referenciaiként is előírhatjuk.

Az integrációra több megoldás is lehetséges:

(a) ROS Serialon keresztül.

(b) ROS control használata.

(c) Osztott Rendszer.

Ros Serial

Egy megoldás a hardver integrációjára a ROS Serial amely UART, vagy TCP protokollra épülő, soros vagy hálózati kommunikációt használva. Korlátai miatt [7] nem képes nagy méretű üzentek használatára, valamit a nodok száma is korlátozott lásd. Szükséges a ROS csomagok használata a hardveren ami nem minden előnyös.

A [1] cikkben egy arduino típusú fejlesztő lappal valósítja meg a robot alacsony szintű szabályzását, megemlíti hogy a rosserial-t nem tudja használni a limitációk miatt. A feldolgozó oldalon elkészít egy saját szoftveres modult amely megvalositja a ROS es a hardver közti kapcsolatot. A kommunikációra soros *UART* protokollt használ.

A [6] könyv 8-ik fejezetben leírja hogyan lehet használni a rosserial-t, arduino valamint Raspberry Pi fejlesztő lapokon de viszont nem tesz említést a hátrányairól.

Ros Control

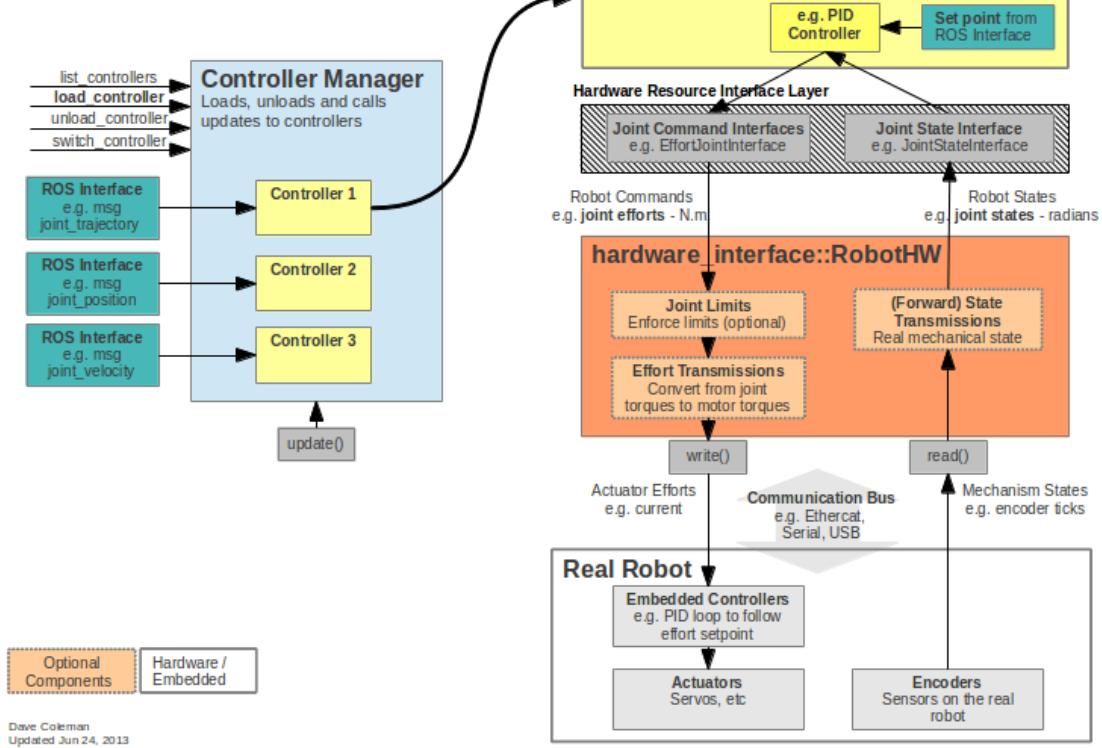
A ros controller [4] használatával összeköthetjük az alacsony szintű hardvert a ROS keretrendszerben fejlesztett modulokkal, implementálva [17] a hardware_interface::RobotHW interfészét és létrehozva minden egyes rotációs csuklónak egy hardware_interface::JointStateHandle-t. A ábra 2.4.7 látható read() és write() függvényeken keresztül kell megvalósítani a kommunikációt a hardverrel, ez történhet hálózaton vagy soros porton keresztül.

⁴ Robot szimulációs szoftver amely támogatja a ROS-t

⁵ 3D modellező szoftver

ROS Control

Data flow of controllers



2.7. ábra. Ros Control modulokForrás:

A ábra 2.4.7 abran lathato az interfeszek kapcsolatai es a fontosabb fuggvenyhivasok. A hardverel valo integraciót write() es read() fuvenyhivasokkal valosul meg, ebben a ket fugvenyen kell elkesziteni a programokat amelyek kepesek kiplvasni es beirni az eszközben a kivant jeleket. Itt hasznalhatunk tobb tipusu alacsonszintu komunikacios protokolt pl: TCP,UART..., vagy barmilyen interfeszt amit a gazga operaciorendszer elismer.

Osztott Rendszer

A [19] cikkben leír egy megoldást arra hogyan lehetne smart eszközöként tekinteni a szenzorokra és beavatkozokra. Osztott rendszert *DSC* -t alkalmaz, ahol minden szenzornak saját mestere van, ezáltal minden node független lesz a hálózaton. A hálózat konfigurációját teljes mértékben ismerni kell minden nodenak a IP címét, de ezzel a megoldással futásidőben módosíthatjuk a hálózatot, *GUID* -t használ a új maszter bejelentésre a hálózaton, valamit ezzel is oldja meg az információk áramlását is. A kommunikációt a mesterek között ROS MultiPeer Architecture (RMPA) nevezet architektúrával oldja meg. Más rendszerekhez képest kétszer jobb időkésést valósított meg. Ami a hátránya, a szenzorok mellett egy mcu is szükséges amely kepés egy operációs rendszert futtatni és egy rós masztert. Valóban robusztusabb modulárisabb lesz a rendszer ezáltal de drágább is. Kisebb rendszereknél mint inkább hátrány mint előny, de viszont komplex nagy területet lefedő rendszernél előnyös.

Egy másik hasonló megoldás ahol több ROS masztert lehet összekötni és felügyelni a

multimaster_fkie megoldja a futásidőben való új maszter szinkronizációját, a topikok és a szervizek kezelését is.

2.5. Mobilis robotok modellezése

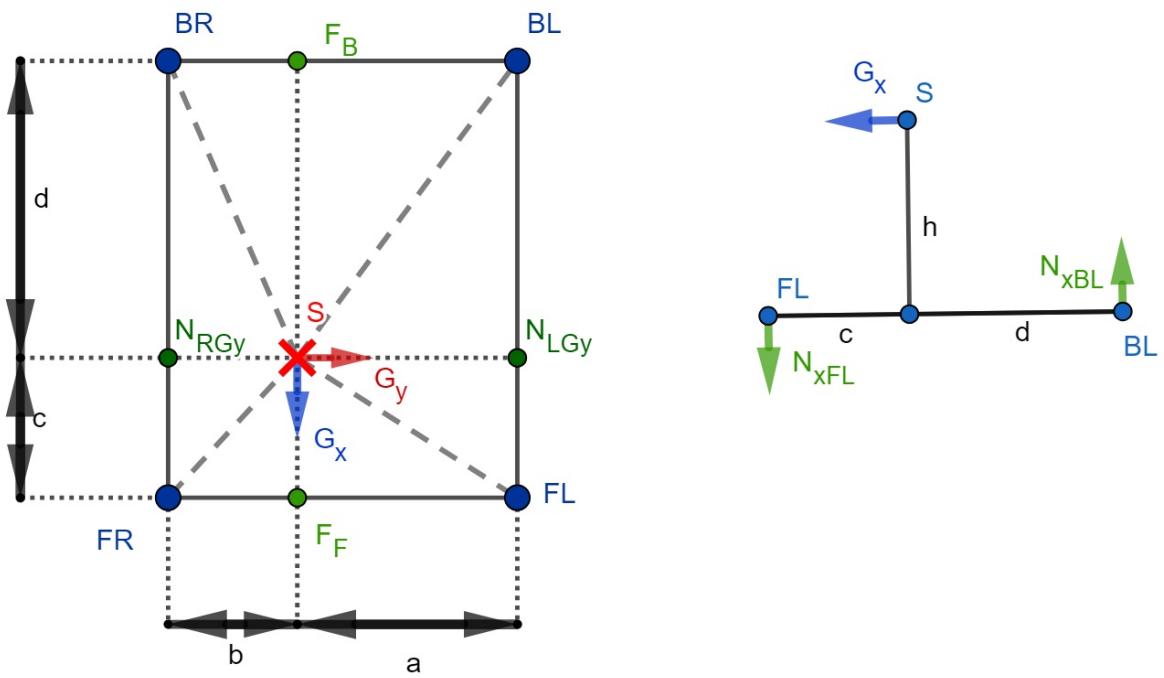
Négy kerekű modell

A mobilis robotok leginkább kerekekkel oldják meg a helyváltoztatásukat. Egy fontos probléma ezekkel a robotokkal az hogy milyen kölcsönhatások lepnek fel a kerék és a talaj között [21] [2] [3], és ezeket az erőket hogyan lehet modellezni. A [21] cikkben kidolgoz egy modellt amely segítségével képes meghatározni egy négykerekű robot pozícióját a kerekek forgási sebességéét felhasználva. A fent említett irodalmakban a *SSMR* típusú *OMP* vizsgálnak módélezés és pályakövetés szempontjából. Annak függvényében hogy a MR-t mozgató motorokat tekintve a következő variánsok lehetségesek: 4 kerék - 4 motor, 4 kerék - 2 motor, azonos oldalon levő kerekek összecsatolva fogas-szíjjal ez második megoldás egyszerűbb kevesebb szenzort és hajtó motort igényel. A [21] [3] irodalmakban a *HLC* sebesség referencia jelet ír elő a kerekeken, [2] a cikkben nyomatéket ír elő amelyet az alacsony szintű szabályozóknak. Az *ICR* meghatározásával választotta. Több dolgot is feltételez: a robot forgásközpontja a robot középpontjában van, a robot azonos oldalán levő kerekek ugyanazzal a szögsebességgel forognak, a négy kerék minden érintkezik a talajjal és méretben is megegyeznek.

A [2] kifejezetten a SSMR jól ismert a robosztus félépítése miatt, nagyon jól alkalmazható terepen. Általában *DDV* mert a fordulást azáltal oldják meg hogy a jobb és bal oldali kerekek különböző sebességgel vagy különböző irányú nyomatéket fejtenek ki a talajra. A [2] cikkben alkalmazott technológiáiak: *VFO CTM* a robot kerekeit szabályzó motoroknak nyomaték van előírva amit követniük kell.

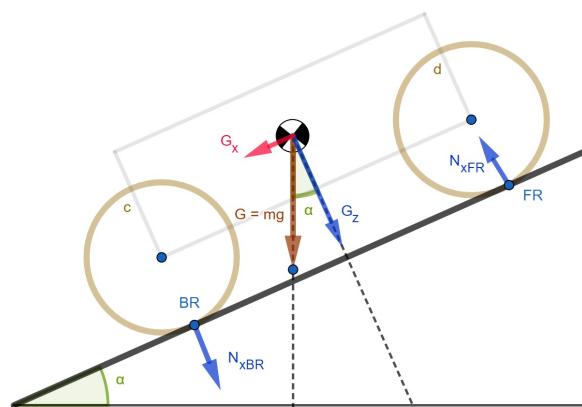
2.5.1. Merőleges Nyomóerő (N)

A kerekek és a talaj egypontban érintkezik, ezeket a pontokat jelölje a BR,BL,FL,FR a ábra 2.5.8 abran. Jelölje S a robot súlypontj, G - a súlyabol szarmazó erőt a robot alapjához rendelt kordinata rendszerben felbontva a harom tengely menten, N a merőleges nyomoero a talajra az adott pontban.

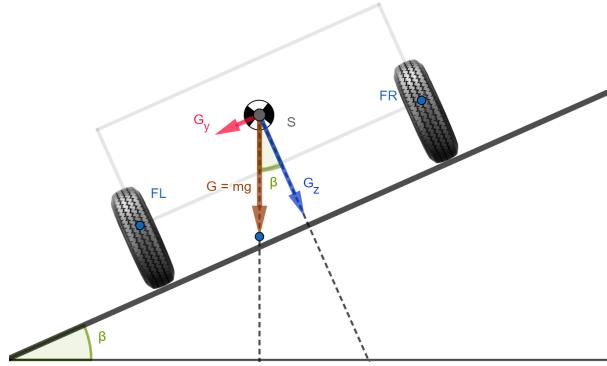


2.8. ábra. Merőleges nyomóerő a talajra 4W – SSMR típusú robot esetében.

Jelölje az α szög ha a lejton felfele halad a robotre fig:S MR4WLejtoOldalrol, β ha a robot a lejton oldara halad ábra 2.5.10. Ha ismerjük a robot súlypontjának a poziciját mindenharom tengelyen akkor kiszámolhatjuk a kerekek merőleges nyomoerejét a talajra a következő módszerrel.



2.9. ábra. 4W – SSMR típusú robot lejtőn felfele oldal nézetből.



2.10. ábra. 4W – SSMR típusú robot lejtőn első nézetből.

Egy test nyugalomban van ha a ra hato erok eredoje zero es a forgatonyomatekok eredoje, ismerve a sulypontr poziciojanak a kordinatati a robothoz kotot VNR-be akkor az 2.1 egyenlettel meghatarozuk a G_x ero altal letrehozott nyomoeroket a N_F es N_B pontokban.

$$N_{FGx} = \frac{hG_x}{c+d}, \quad N_{BGx} = -N_{FGx} \quad (2.1)$$

Meghatarozuk a G_y ero altal letrehozott nyomoeroket a N_{RGy} es N_{LGy} pontokban.

$$N_{RGy} = \frac{hG_y}{a+b}, \quad N_{LGy} = -N_{FGy} \quad (2.2)$$

Ismerve a N_{LGy} es N_{RGy} pontokban hato eroket kiszamitjuk ezek eloszlasat a robot kerekeire nezve, így megkapjuk azokat a nyomoeroket amelyet a ábra 2.5.10 abran lathato allapotban a G_y gravitaciobol szarmazo ero hoz letre.

$$N_{yBL} = \frac{dN_{LGy}}{c+d}, \quad F_{yFL} = -N_{yBL} \quad (2.3)$$

$$N_{yBR} = \frac{dN_{RGy}}{c+d}, \quad F_{yFR} = -N_{yBR} \quad (2.4)$$

Meghatarozuk a gravitacio Z komponense altal letrehozott nyomoeroekt a F_F es a F_B pontokban amelyhz hozadjuk a X komponens altal letrehozott nyomoeroekt ugyan ezekben a pontokban.

$$F_F = \frac{G_z d}{c+d} + N_{FGx}, \quad F_B = G_z - F_F + N_{BGx} \quad (2.5)$$

Ismet kiszamoljuk a kerekekre vetitett nyomoeroket ismerve az F_F es F_B eroket.

$$F_{BR} = \frac{aF_B}{a+b}, \quad F_{BL} = F_B - F_{BR} \quad (2.6)$$

$$F_{FR} = \frac{aF_F}{a+b}, \quad F_{FL} = F_F - F_{FR} \quad (2.7)$$

A meroleges nyomoero vektor az X,Y,Z gravitcios erok altal letrehozott nyomoerok osszegzesebol all.

$$N_{\perp} = [F_{FL} + N_{yFL} \quad F_{BL} + N_{yBL} \quad F_{FR} + N_{yFR} \quad F_{BR} + N_{yBR}]^T \quad (2.8)$$

2.5.2. Sulpont (X,Y) komponenseinek a meghatarozása

Ismerve a robot mereteit W jelolje a szeleseget mig a L hoszusagat, kerekkozepont kozott merve.

A robotot vizszintes helyzetbe helyezuk, es minden kerek meroleges nyomoerejet megmerve merleg segitsegevel megkapjuk a $N_{FL}, N_{FR}, N_{BL}, N_{BR}$ nyomoeroket.

$$W = a + b, \quad L = c + d \quad (2.9)$$

Meghatarozuk a a ertekeket ismerve a F_{FR} pontban a nyomoerot es kiszamolva a F_F pontban a nyomoerot a 2.11 es 2.12 egyenletek segitsegevel.

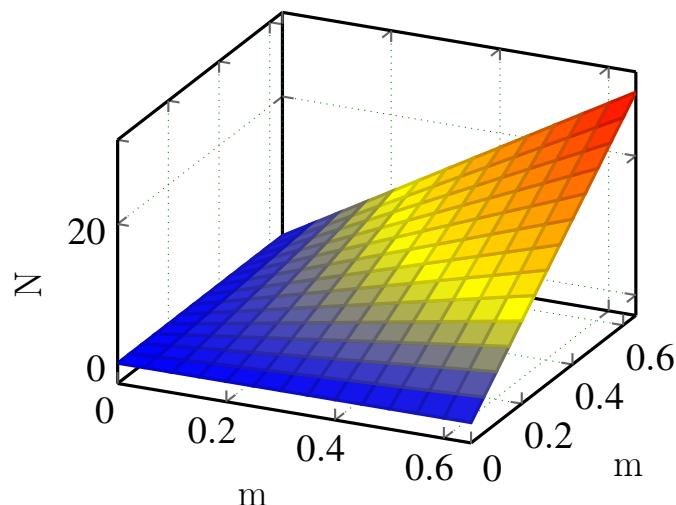
$$F_{FR} = \frac{aF_F}{W} \Rightarrow a = \frac{F_{FR}W}{F_F} \quad (2.10)$$

$$G_z = F_{FR} + F_{FL} + F_{BR} + F_{BL} \quad (2.11)$$

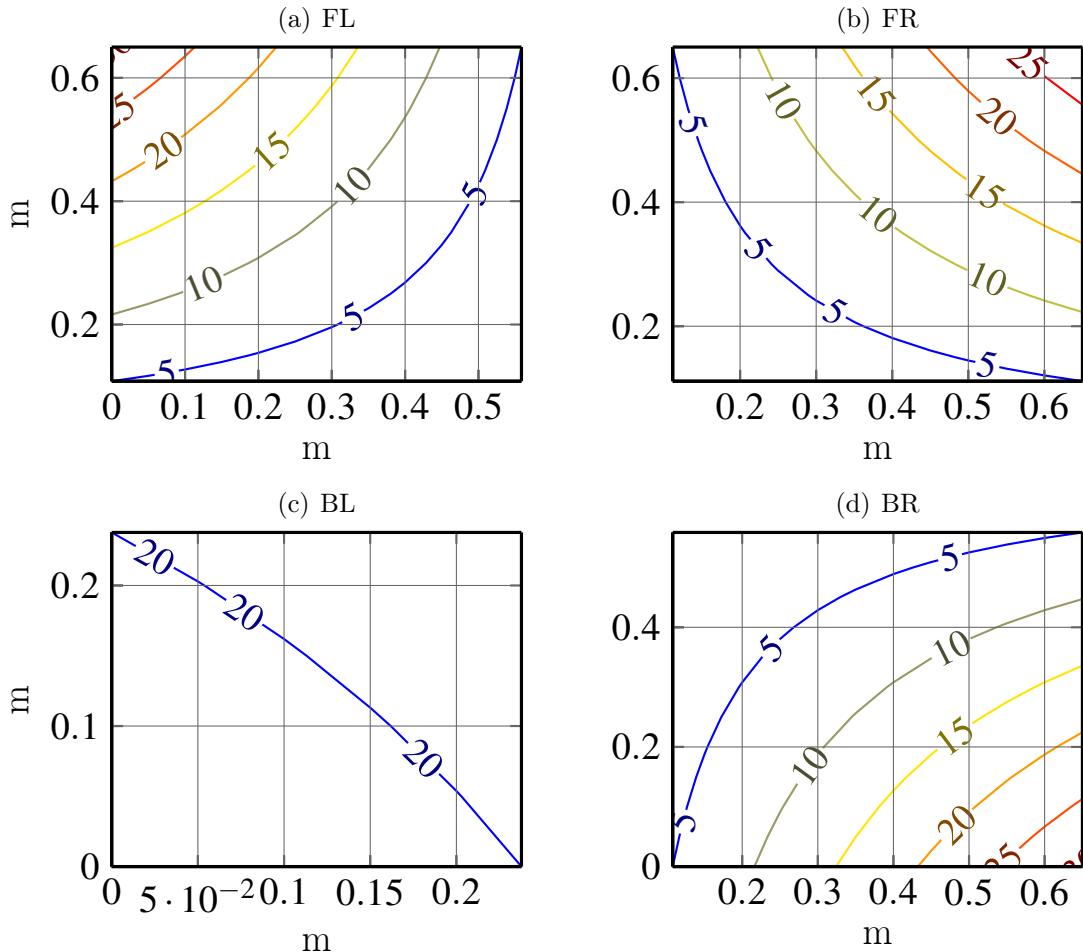
$$F_F = \frac{dG_z}{L} \Rightarrow d = \frac{F_F L}{G_z} \quad (2.12)$$

Meroleges nyomoero alakulasa a sulpont fugvenyeben

A tetelezuk fel hogy a robot sulya 28kg, a BR kerek kozebe legyen a $(0,0)$ pont, $W \in (0m, 0.6m)$ es $L \in (0m, 0.6m)$ ertekeketeket vehet fel. A ábra 2.5.11 abran lathato a FR pontban a meroleges nyomoero valtozasa a sulpont poziciojanak a fuggvenyben.



2.11. ábra. Kerek nyomoero valtozasa a sulpont fugvenyeben ha $\alpha = 0$ es $\beta = 0$



2.12. ábra. *SSMR – 4W* tipusu robot kereknyomoerok kerekenkeni változása a súlypont függvényében

A ábra 2.5.12 abran lahato a szimulacio mind a negy kerekre.

2.5.3. Súlypont meghatározása mérésekkel

A robot mélypont meghatározása egy mérleg segítségével lemérve sorra minden kerék merőleges nyomóerőjét a talajra nézve. A mért adatok vízszintes helyzetben:

Node	Nyomó erő	Mértékegység
FL	11,8	kg
FR	13,2	kg
BL	17,1	kg
BR	17,9	kg

A súlypont pozíciója: $b = 20$ es $c = 30$

2.5.4. Kerekek Dinamikája

Az $I_w \in \mathbb{R}^4$ tartalmazza a kerekek inerciáját a forgás tengelyükhez képest. $\Omega \in \mathbb{R}^4$ a kerekek szögsebessége. A $W_r \in \mathbb{R}^4$ a kerekek sugara, $\tau \in \mathbb{R}^4$ a kerekek forgatónyomatéka.

$$I_w \dot{\Omega} = \tau - W_r F_x \quad (2.13)$$

$$I_w = \begin{bmatrix} I_{FL} & 0 & 0 & 0 \\ 0 & I_{BL} & 0 & 0 \\ 0 & 0 & I_{FR} & 0 \\ 0 & 0 & 0 & I_{BR} \end{bmatrix}, \quad W_r = \begin{bmatrix} r_{FL} & 0 & 0 & 0 \\ 0 & r_{BL} & 0 & 0 \\ 0 & 0 & r_{FR} & 0 \\ 0 & 0 & 0 & r_{BR} \end{bmatrix}$$

$$\tau = [\tau_{FL} \quad \tau_{BL} \quad \tau_{FR} \quad \tau_{BR}]^T, \quad \Omega = [\omega_{FL} \quad \omega_{BL} \quad \omega_{FR} \quad \omega_{BR}]^T$$

2.6. Kinematikai Modell

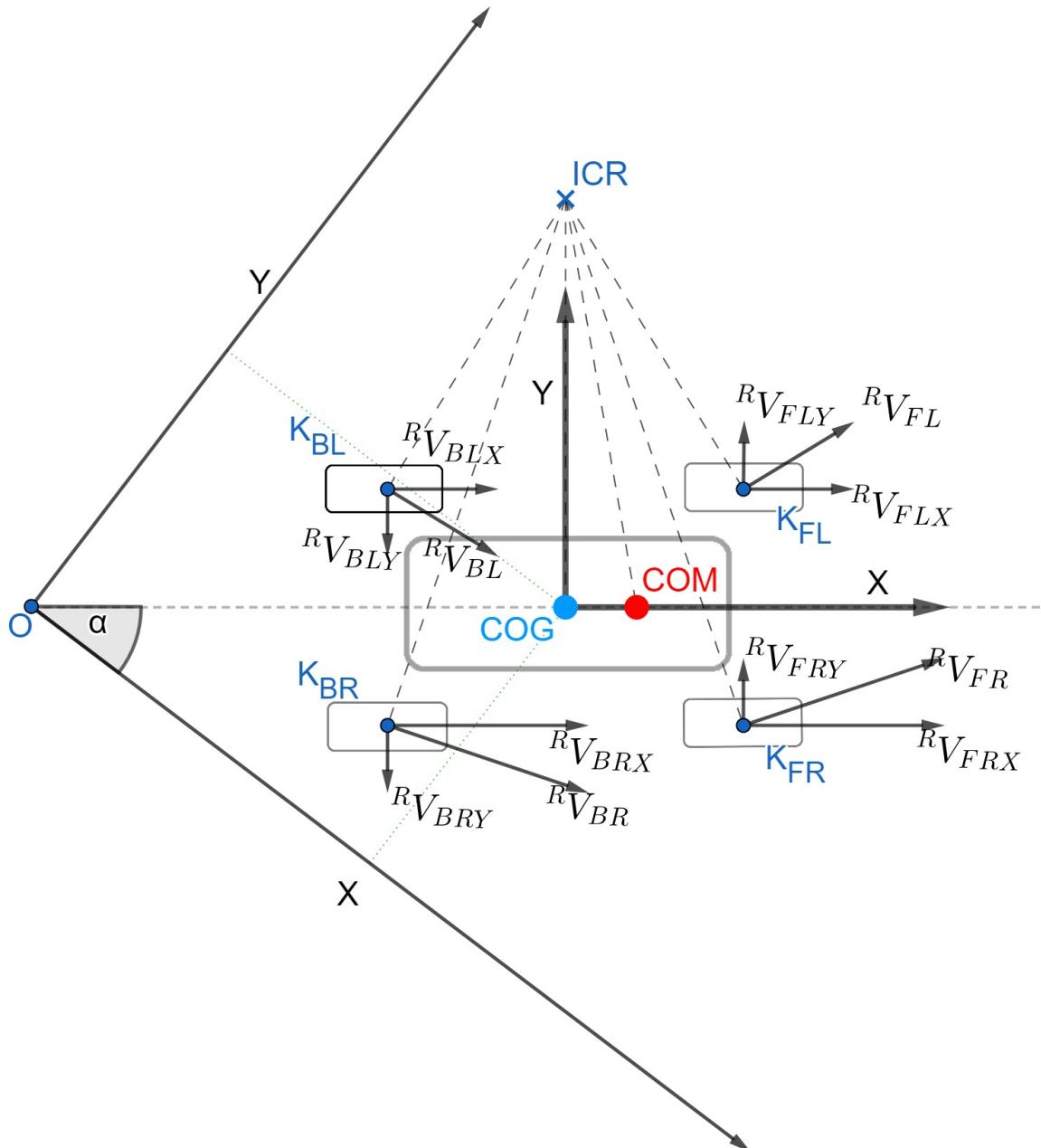
A ábra 2.6.13 látható a *4W – SSMR* kinematikai modellje. Néhány feltételezés: a robot minden kereke mindenkor érintkezik a talajjal, a kerekek nem csúsznak forgásuk közben, külön van kezelve a laterális és a longitudinális súrlódás, a robot egy tömeg központtal van jellemezve, az alacsony szintű szabályzok tökéletesen követik az előírt referenciát.

A robot a *ICR* pont körül fordul, és csak a robothoz rendelt vonatkoztatási rendszer x tengelye mentén tud elmozdulni. Az y irányú sebességeket azt okozza hogy a jobb és bal oldali kerekek forgási sebessége eltér és így létrejönne egy oldal irányú csuszás.

Jelölje a rendre a K_{ik} a kerekek a talajjal való érintkezési pontját, ${}^R V_{ik}$ a K_{ik} pontok sebességet a robothoz rendelt *VNR*-ben, ${}^R V_{ikX}$ és ${}^R V_{ikY}$ rendre a ${}^R V_{ik}$ sebesség X és Y komponense robothoz rendelt *VNR*-ben. A ${}^R V_{ikX}$ megfelel a kerekek kerületei sebességének. A ${}^R V_{ik}^{COM}$ a *COM* pont sebességet a robothoz rendelt *VNR*-ben, illetve a ${}^R V_{ikX}^{COM}$ és ${}^R V_{ikY}^{COM}$ az X és Y komponense.

A robot és a globális *VNR* x tengelye között bezárt szög θ valamint X és Y a robot pozíciója a O ponthoz viszonyítva.

Az *ICR* pont helyzete a ${}^R V_{ik}$ sebesség vektorokra merőleges egyenesek metszés pontjában található és mindenkor a robothoz rendelt *VNR* Y tengelyen helyezkedik el.



2.13. ábra. Kinematikai modell az $4W - SSMR$ típusú robotnak.

A \dot{q} a $4W - SSMR$ síkban modellezett állapot vektora a globális VNR -ben. ${}^R\omega^{COG}$ az COG pont körül szögsebesség a robothoz rendelt VNR -ben. Az η jelölje a bemeneti értékeket. A d a COG és a COM pontok közti távolság.

A COM pontban mert értékek az egyenlet (2.14) segítségével szamolhatjuk a globális VNR -be. A COM pont sebességének y komponense megadható az egyenlet (2.15) segítségével. A nemholomonikus megkötés egyenlet (2.16) biztosítja azt hogy a robot nem tud oldal irányú mozgást végezni.

$$\dot{q} = \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^R V_x^{COM} \\ {}^R V_y^{COM} \\ {}^R \omega^{COG} \end{bmatrix} \quad (2.14)$$

$${}^R V_y^{COM} = d\omega \quad (2.15)$$

$$\begin{bmatrix} -\sin \theta & -\cos \theta & -d \end{bmatrix} \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \end{bmatrix} = A(q)\dot{q} = 0 \quad (2.16)$$

$$\dot{q} = S(q)\eta \quad (2.17)$$

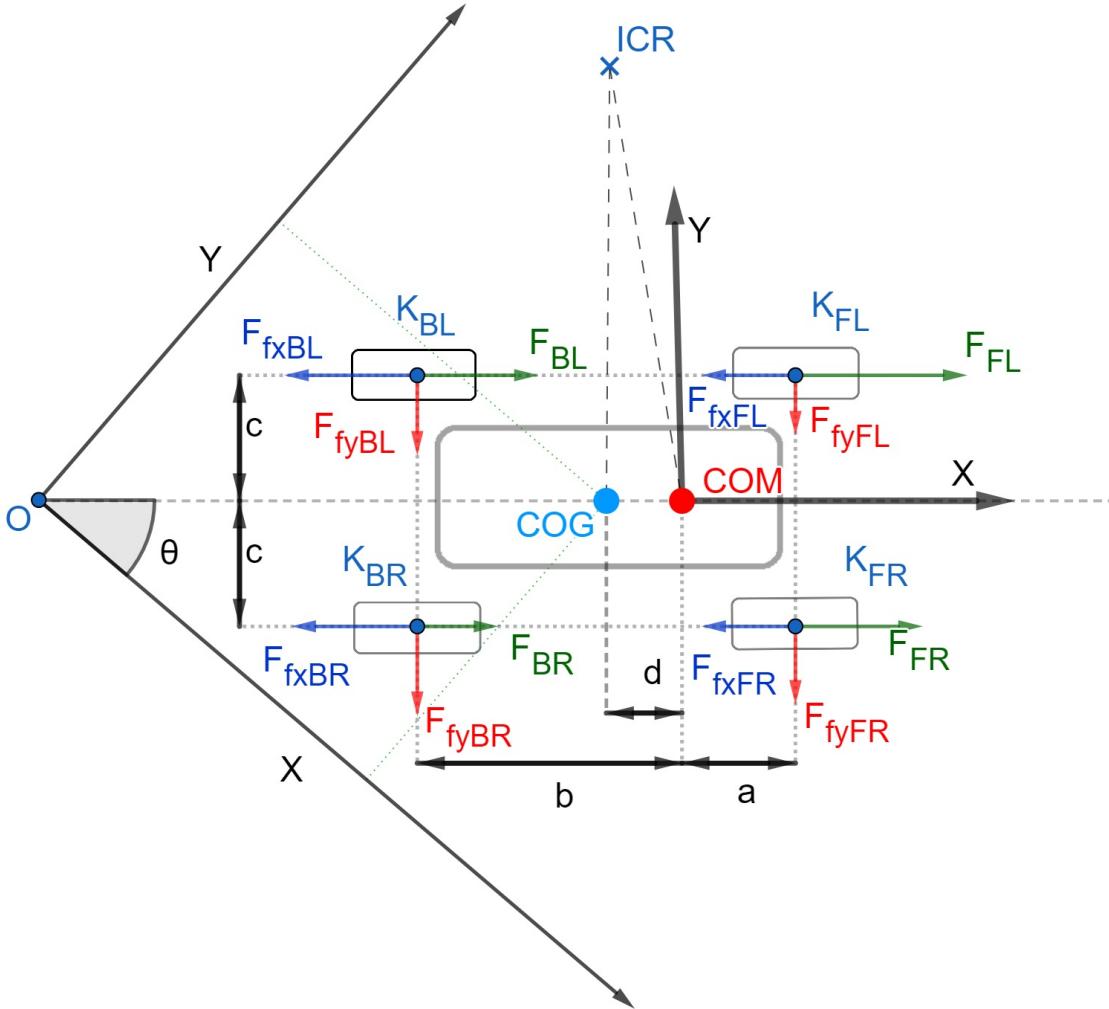
$$S(q) = \begin{bmatrix} \cos \theta & -d \sin \theta \\ \sin \theta & d \cos \theta \\ 0 & 1 \end{bmatrix}, \eta = \begin{bmatrix} {}^R V_x^{COM} \\ {}^R \omega^{COG} \end{bmatrix}$$

$$S^T A^T = 0 \quad (2.18)$$

2.7. Dinamikai Modell

A ábra 2.7.14 látható a $4W - SSMR$ -ra ható erők rendszerre. Jelölje a F_{ik} a K_{ik} pontokban a kerekek a talajra kifejtett erőt, F_{fxik} és a F_{fyik} rendre az x és y irányba ható súrlódási erőket a K_{ik} pontokban.

Az egyenletek (2.19)-(2.21) leírják a robot mozgását a globális rendszerben, felhasználva a robot VNR - ben mert erő hatásokat.



2.14. ábra. Kinematikai modell az *SSMR* típusú *MR* robotnak.

Az $F_x \in \mathbb{R}^4$ tartalmazza F_{ik} kerekek által a talajra kifejtett erőket. Az $F_{sx} \in \mathbb{R}^4$ és $F_{sy} \in \mathbb{R}^4$ súrlódási erők x és y tengely mentén a robot *VNR*-ben. A $F \in \mathbb{R}^2$ tartalmazza a jobb és bal oldali kerek által a talajra kifejtett erők összegét. Jelölje I a robot inerciáját a z tengely körül, M_a nyomatékok összege amelyeket a kerekek hoznak létre, M_r a nyomatékok összege amelyeket a súrlódások hoznak létre.

A $K_x \in \mathbb{R}^4$ és $K_y \in \mathbb{R}^4$ jelölje a súlypont pozíciója a kerek és talaj érintkezési pontokhoz viszonyítva a ábra 2.5.8 alapján.

Az $N \in \mathbb{R}^4$ tartalmazza a merőleges nyomóerőket talajra nézve, a K_{ik} pontokban.

$$m\ddot{X} = \xi F_x \cos \theta - \xi F_{sx} \cos \theta + \xi F_{sy} \sin \theta \quad (2.19)$$

$$m\ddot{Y} = \xi F_x \sin \theta - \xi F_{sx} \sin \theta - \xi F_{sy} \cos \theta \quad (2.20)$$

$$I\ddot{\theta} = M_a + M_r, \quad (2.21)$$

$$\xi = [1 \ 1 \ 1 \ 1]$$

$$F_{sx} = [F_{sxFL} \ F_{sxBL} \ F_{sxFR} \ F_{sxBR}]^T, \quad F_{sy} = [F_{syFL} \ F_{syBL} \ F_{syFR} \ F_{syBR}]^T$$

$$F_{sxik} = N_{ik} \mu_{xik} S_{xik}, \quad F_{sy} = N_{ik} \mu_{yik} S_{yik}$$

$$\mu_x = [\mu_{xFL} \ \mu_{xBL} \ \mu_{xFR} \ \mu_{xBR}]^T, \quad \mu_y = [\mu_{yFL} \ \mu_{yBL} \ \mu_{yFR} \ \mu_{yBR}]^T$$

$$S_x = [sgn(V_{xFL}) \quad sgn(V_{xBL}) \quad sgn(V_{xFR}) \quad sgn(V_{xBR})]^T$$

$$S_y = [sgn(V_{yFL}) \quad sgn(V_{yBL}) \quad sgn(V_{yFR}) \quad sgn(V_{yBR})]^T$$

$$M_r = M_{rx} + M_{ry} \quad (2.22)$$

$$M_{rx} = K_x^T F_{sx}, M_{ry} = K_y^T F_{sy}, M_a = K_x^T F_x \quad (2.23)$$

$$K_x = [a \ a \ b \ b]^T, \quad K_y = [c \ d \ c \ d]^T$$

$$F_x = [F_{FL} \ F_{BL} \ F_{FR} \ F_{BR}]^T$$

$$F = [F_{FL} + F_{BL} \ F_{FR} + F_{BR}]^T$$

Általános formában a $4W - SSMR$ dinamikai modellje a egyenlet (2.24) adható meg a [2] alapján. Jelölje $M \in \mathbb{R}^{3 \times 3}$ a tömegek és inerciák mátrixa, $R \in \mathbb{R}^3$ ellenálló nyomatékok és erők mátrixa, $B \in \mathbb{R}^{3 \times 2}$ bemeneti mátrix, A a megkötések vektora egyenlet (2.16) alapján, λ Lagrange együtthatók vektora. $F_d \in \mathbb{R}^3$ zajok vektora.

A egyenlet (2.24) az állapotok gyorsulását megkapjuk ha az egyenlet (2.17) időben deriváljuk, így az egyenlet (2.25)-t kapjuk.

Felhasználva a egyenlet (2.18) és egyenlet (2.25) és egyenlet (2.17) a egyenlet (2.24) egyenletet egyszerűbb alakra írhatjuk azáltal hogy minden tagot beszorzunk balról S^T -vel, így a egyenlet (2.26) kapjuk.

$$M(q)\ddot{q} + R(\dot{q}) + F_d = B(q)F + A^T \lambda \quad (2.24)$$

$$M = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix}, \quad B(q) = \begin{bmatrix} \cos \theta & \cos \theta \\ \sin \theta & \sin \theta \\ -a & b \end{bmatrix}, \quad D = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}^T, \quad R(\dot{q}) = \begin{bmatrix} \xi F_{sx} \cos \theta - \xi F_{sy} \sin \theta \\ \xi F_{sx} \sin \theta - \xi F_{sy} \cos \theta \\ M_r \end{bmatrix}$$

$$\ddot{q} = S(q)\dot{\eta} + \dot{S}(q)\eta \quad (2.25)$$

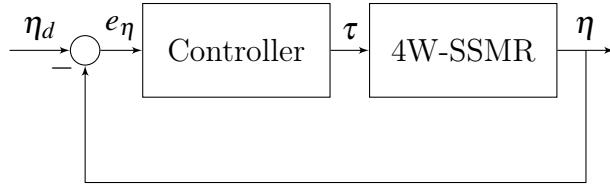
$$\bar{M}\dot{\eta} + \bar{C}\eta + \bar{R} + \bar{F}_d = \bar{B}F \quad (2.26)$$

$$\bar{M} = S^T M S, \quad \bar{C} = S^T M \dot{S}, \quad \bar{R} = S^T \dot{R}, \quad \bar{F}_d = S^T F_d, \quad \bar{B} = S^T B$$

2.8. Robot Platform Sebesség Szabályzása

2.8.1. Eloirt nyomatekkal

A kerekek előírt nyomatékát megkapjuk ha a egyenlet (2.27) -t használva. Az u szabályzó jelet kiszámíthatjuk ha az egyenlet (2.28)-t használjuk. Jelölje a K_η a szabályzó paramétereit. Csuszás szabályzó σ_η parameteri ρ_v linearis sebességért, és ρ_w szögsebességért felelős. Mindkét paraméter nagyobb kell legyen mint a zaj n megfelelő érteké.



2.15. ábra. Kinematikai modell az *SSMR* típusú *MR* robotnak.

$$\tau = W_r D \bar{B} [\bar{M} u + \bar{C} \eta + \bar{R}] + I_w \dot{\Omega} \quad (2.27)$$

$$u = \dot{\eta}_d + K_\eta e_\eta + \sigma_\eta \quad (2.28)$$

$$e_\eta = \eta_d - \eta \quad (2.29)$$

$$e_\eta = [e_v \ e_w]^T \in \mathbb{R}^2, \quad \sigma_\eta = [\sigma_v \ \sigma_w]^T \in \mathbb{R}^2, \quad \sigma_v = \rho_v sgn(e_v), \quad \sigma_w = \rho_w sgn(e_w) \quad (2.30)$$

Mesterséges Erő Módszere

A [5] cikk alapján egy másik megközelítést használva modellezzi a robot. A q állapotokat meg kiegészíti a jobb és bal oldali kereke szögsebességével. Feltételezi hogy a kerekek sugara r mind a négy kereknél egyenlő, és a *COM* pont a robot szimmetriatengely helyezkedik el. Jelölje F a ellenálló erők és nyomatékok vektora. Hasonlóképpen az egyenlet (2.24) -hez a Lagrange egyenletet ír fel a dinamikai modellre. Az η tartalmaz az előírt sebességek vektora, az η_3 a sebességek vektora, a 3-dik eleme tartalmazza a generált sebességet amelyet úgy kell előírnunk hogy a hozzá tartozó előírt kerek sebesseg nulla legyen. Az $u \in \mathbb{R}^3$ a jobb és bal oldali kerekek előírt erőleadása a talajra.

$$A(q)\dot{q} = \begin{bmatrix} \cos \theta & \sin \theta & -c & -r & 0 \\ \cos \theta & \sin \theta & c & 0 & -r \end{bmatrix} \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \\ \Omega_L \\ \Omega_R \end{bmatrix} \quad (2.31)$$

$$M(q) = \begin{bmatrix} m & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & I_{FL} + I_{BL} & 0 \\ 0 & 0 & 0 & 0 & I_{FR} + I_{BR} \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$F(q, \dot{q}) = [F_x \cos \theta - F_y \sin \theta \quad F_x \sin \theta - F_y \cos \theta \quad M_r \quad 0 \quad 0]^T \quad (2.32)$$

$$\dot{q} = G_e(q) \eta = \begin{bmatrix} \cos \theta & \cos \theta & -\sin \theta \\ \sin \theta & \sin \theta & \cos \theta \\ \frac{1}{c} & -\frac{1}{c} & 0 \\ 0 & \frac{2}{r} & 0 \\ \frac{2}{r} & 0 & 0 \end{bmatrix} \begin{pmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{pmatrix} \quad (2.33)$$

$$\underbrace{G_e^T M G_e}_{M^*} \dot{\eta} + \underbrace{G_e^T (M \dot{G}_e + C G_e)}_{C^*} \dot{\eta} + \underbrace{G_e^T F}_{F^*} = \underbrace{G_e^T B u}_{B^*} \quad (2.34)$$

$$M^*(q) = \begin{bmatrix} m + \frac{I}{c} + 4 \frac{I_k}{r^2} & m - \frac{I}{c^2} & 0 \\ m - \frac{I}{c^2} & m + \frac{I}{c^2} + 4 \frac{I_k}{r^2} & 0 \\ 0 & 0 & m \end{bmatrix}, \quad F^*(q, \dot{q}) = [-F_x - \frac{M_r}{c} \quad -F_x + \frac{M_r}{c} \quad -F_y]^T$$

$$C^*(q) = \begin{bmatrix} 0 & 0 & -m\dot{\theta} \\ 0 & 0 & -m\dot{\theta} \\ m\dot{\theta} & m\dot{\theta} & 0 \end{bmatrix}$$

$$B^* = \begin{bmatrix} \frac{2}{r} & 0 & 0 \\ 0 & \frac{2}{r} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (B^*)^{-1} = \begin{bmatrix} \frac{r}{2} & 0 & 0 \\ 0 & \frac{r}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \eta_r = \begin{pmatrix} \eta_{r_1} \\ \eta_{r_2} \\ \eta_{r_3} \end{pmatrix}$$

$$u = (B^*)^{-1} \{ M^* \dot{\eta}_r + C^* \eta_r + F^* - K_d e \} \quad (2.35)$$

$$\tau = W_r D [u_1, u_2]^T \quad (2.36)$$

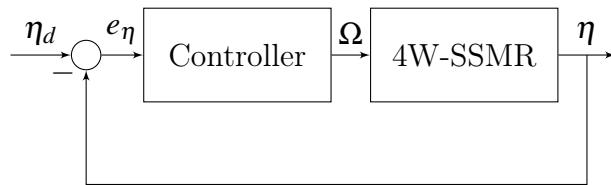
$$e_{\eta_i} = \eta_i - \eta_{r_i} \quad (2.37)$$

$$\dot{\eta}_{r_3} = \frac{m\dot{\theta}(\eta_{r_1} + \eta_{r_2}) - F_y - K_d(\eta_3 - \eta_{r_3})}{m} \quad (2.38)$$

$$\eta_{r_1} = \frac{{}^R V_x^{COM} - {}^R \omega_r^{COG} c}{2}, \quad \eta_{r_2} = \frac{{}^R V_x^{COM} + {}^R \omega_r^{COG} c}{2}$$

2.8.2. Elirt kerekszogsebessegekkel

A [3] cikkben a kerekek sebességét szabályozza, A jobb és bal oldali kerekek modelljét ARX becsléssel meghatározza a matematikai modellt és pólusáthelyezéses módszerrel a kívánt modellt állítja elő.



2.16. ábra. Kinematikai modell az *SSMR* típusú *MR* robotnak.

fejezet 3

A rendszer implementálása

3.1. Robot felepitese

A robot egy négykerekű kültéri mobilis robot *4W – SSMR* amelynek négy különálló csiga áttétel, és egy DC motor mozgat.

Paraméter	érték	Mértékegység
Szélesség	80	[cm]
Hosszúság	80	[cm]
Magasság	40	[cm]
Önsúly	54	[kg]
Max sebessége	25	[cm/s]
Max fordulási sebesség	25	[°/s]
Kerék-átmérő	40	[cm]
Maximális kerék forgatónyomaték	1000	[N/m]

A robton egy performáns számítógép felelős a ROSmaster és egyéb kiszolgáló nodok fűtatasáért, a rendszerhez, USB ábra 3.2.1 csatlakozóval kapcsolódó modulok:

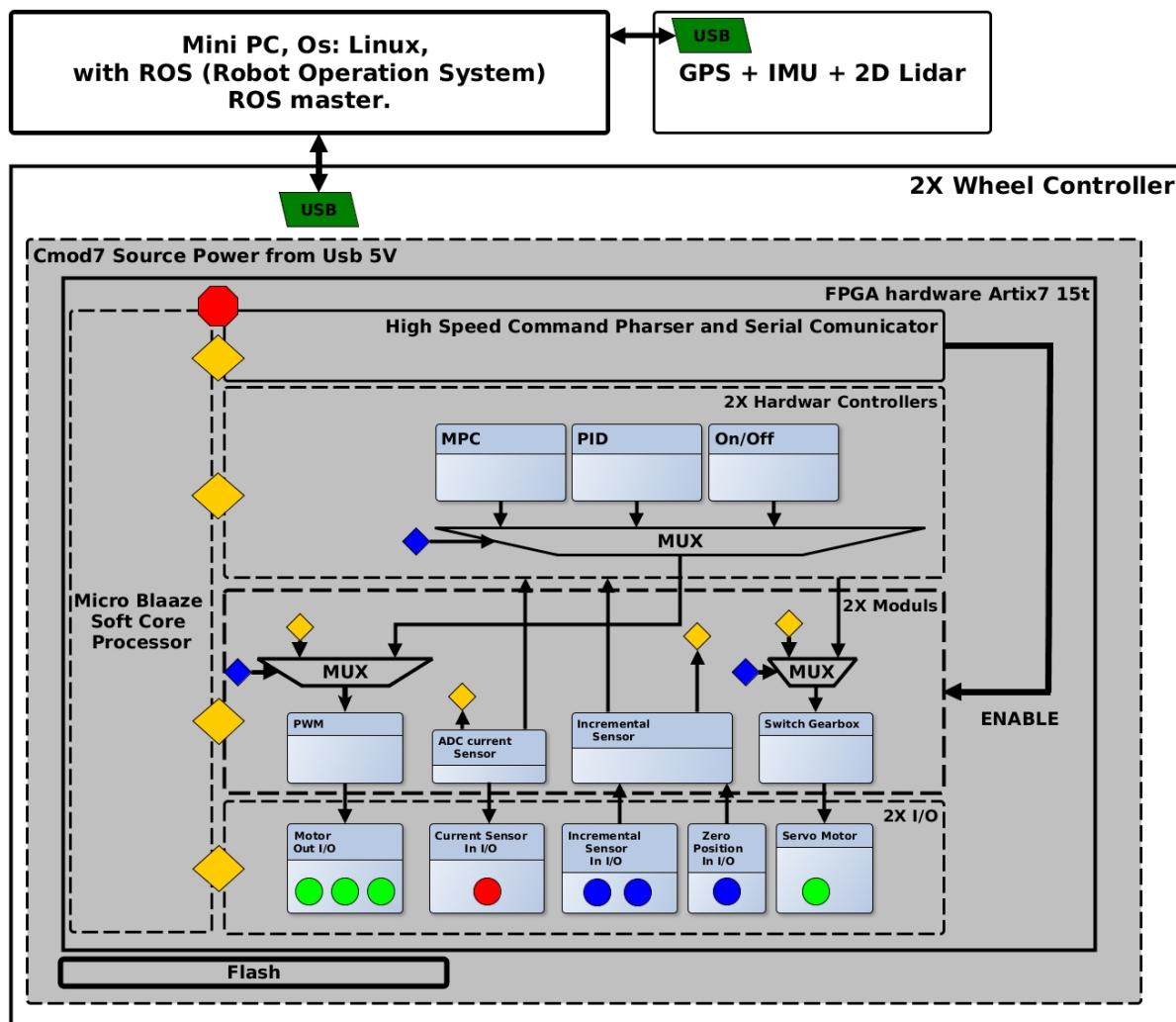
Modulnév	Leírás
2xFPGA	CmodA7 <i>https://reference.digilentinc.com/reference/programmable-logic/cmod-a7/reference-manual</i>
IMU	ESP8266 arduino alapú rendszer, SPI protokollon keresztül olvassa az IMU mert értékeit, átalakítás után string formában uart protokollon küldődik a kiszolgáló ROS nodnak
GPS	Uart porton NMEA protokoll alapján ROS node fogadja az értékeket.
LIDAR	Leírás: <i>https://www.robotshop.com/media/files/pdf2/rpk-02-datasheet.pdf</i> , gyártó által biztosított ROS <i>http://wiki.ros.org/rplidar</i> integráció.

3.2. Alacsony szintű modulok

A robton alacsonyszintű feladatait amelyek a motor hajtásokhoz kapcsolódó szenzorokat és vezérlő jelek elő allitasara hivatott a kétdaram CmodA7 20T FPGA fejlesztőlap. Négy

tengely szögsebesseget kel szabályozni, egy FPGA két hajtást valósít meg. A ábra 3.2.1 alapján egy hajtáshoz tartozó I/O-k:

Név	Darab	Típus
Inkrementális enkoder	2	Digitális Input
Árammérő szenzor	1	Analóg Input
Motor vezérlő	3	Digitális Output
Végálas kapcsoló	1	Digitális Input



3.1. ábra. CmodA7 FPGA-ban kialakított architektúra amely a szenzorok és motor hajtások kezelésre hivatott

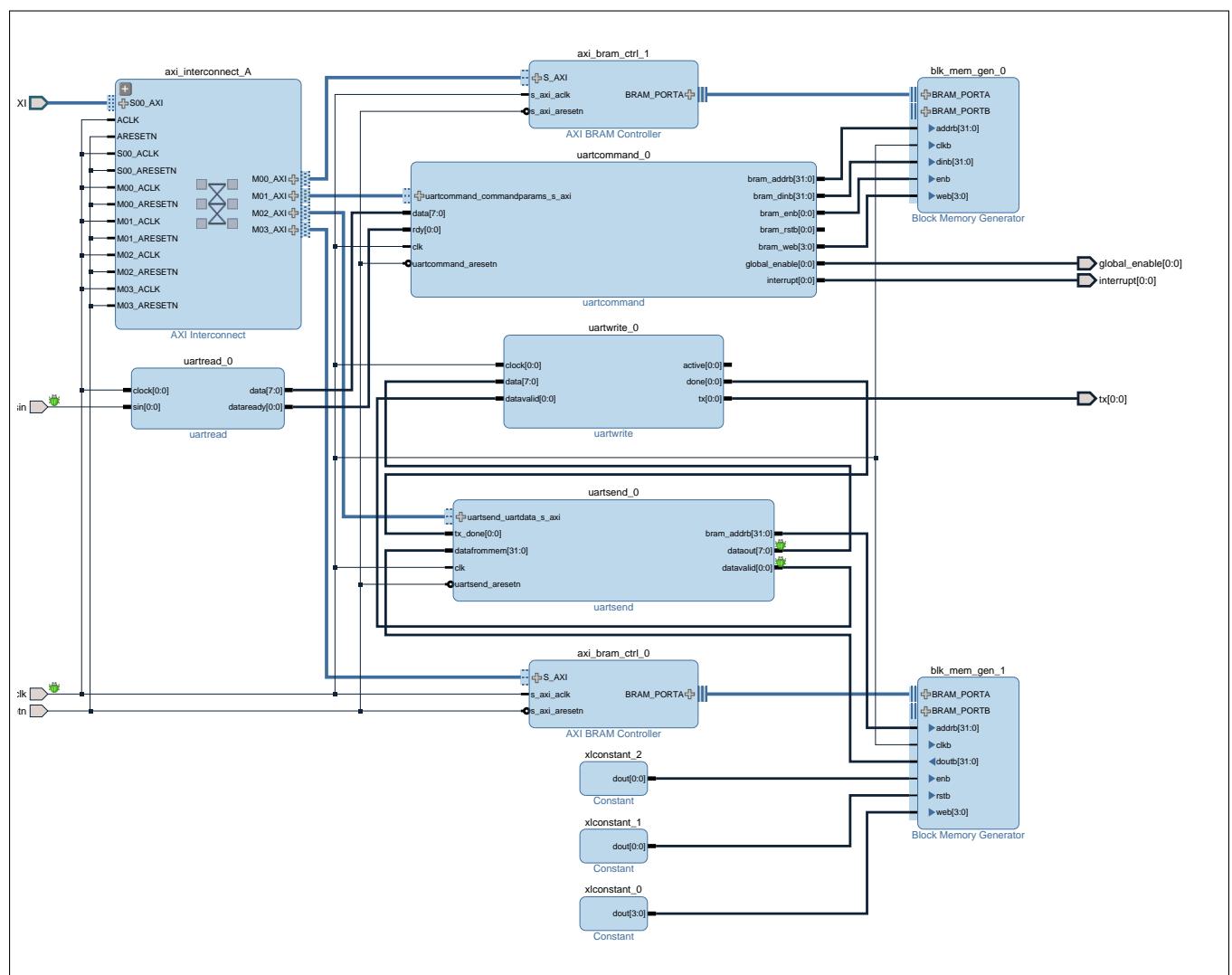
A ábra 3.2.4 a Vivado tervezoprogramban FPGA ban megvalositott proceszor architektura lathato, tartalmaz egy szoft core proceszort (microBlaze) a Xilinx. Az utasitas es az adatmemoria az *microblaze_0_local_memory* ban talalhato. A memoria merete 24Kbyte 32bit savszeleseggel, amely az FPGA block memoriajabol van osszerakva. A programkod maga egy kulso memoriaban van tarolva ami QSPI alapu interfacevel kapcsolodik a FPGA hoz, az *ExternalMemorys* kapcsolja be az AXI busz rendszeren keresztul a proceszor

memoriacimataartomanyaba. A kulso memoria merete 30MByte. A proceszor mukodesehez szukseges a *ClockAndReset* modul eloalitja az orajelet ami 100MHz, ezen a frekvencian mukodik az proceszor es a memoria is. Az *mdm_1* modul a rendszerben torteno jelek megfigyelesere es a proceszorn futo kod debugolasara hivatott.

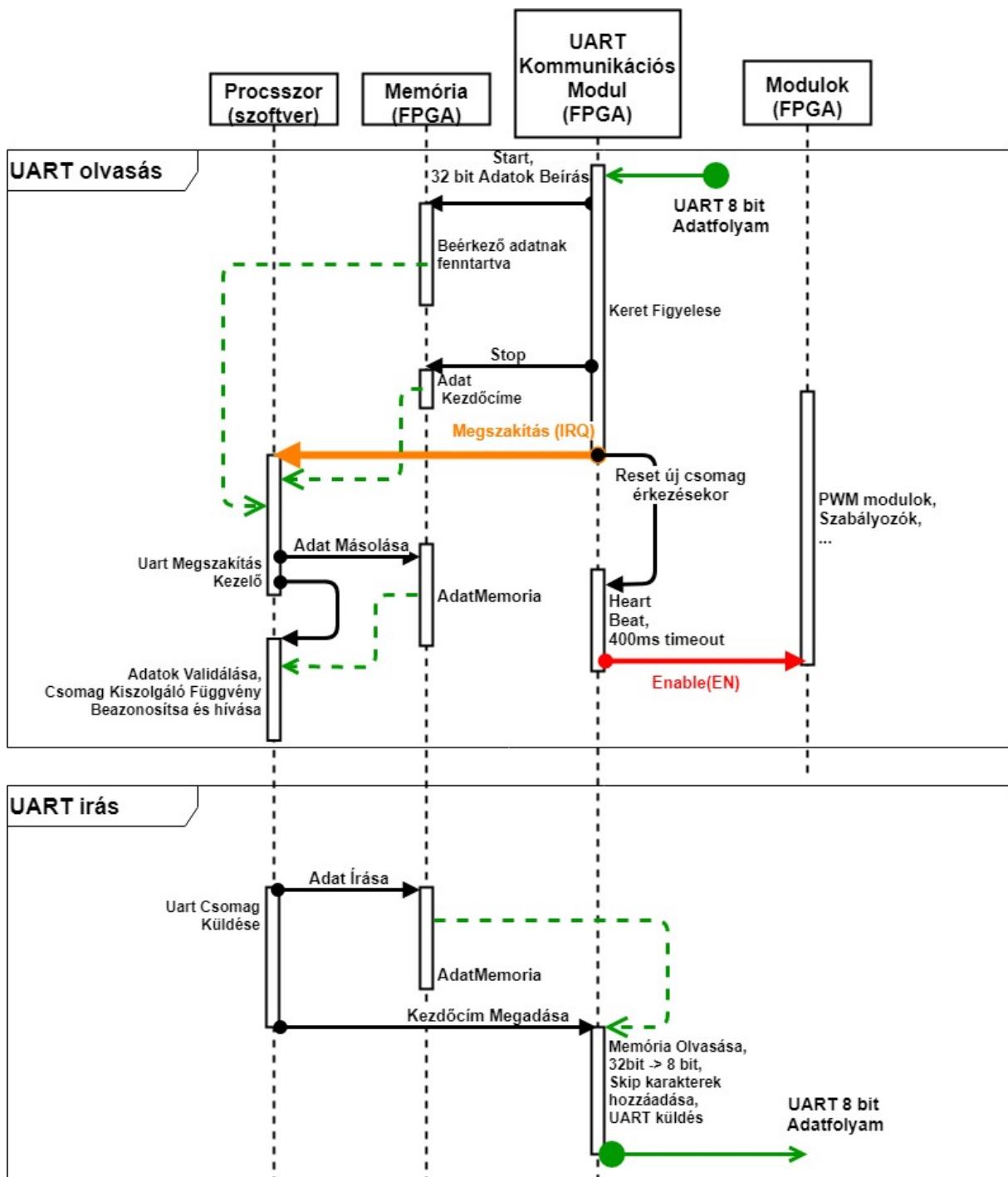
A ábra 3.2.4 abran lathato *UartCom* modul belso szerkezete lathato az ábra 3.2.2 abran. A modula *uartcommand_0* szerepe a komunikacos protokol implmentalasa hardveres szinten, a modul fogadja az UART protokolen 8bit csomagokat. A protokol altal meghatarozott keretezesi bytokat figyelve (Start,Stop,Skip). Ezek paremeterkent megadhatok a microBlaze proceszoren keresztul. A procszor indulaskor bealitja ezeket a kovetkezo ertekekre: Start='S', Stop='P', es Skip='

'. Start keret byte erkezesetol kezdve az oszess byte negyeseval bekerul a 32bit, 400 byteos, *blk_mem_gen_0* memoriaba sorfojtonosan. Abban az esetben ha az uzenet megfaladja a 200 byte-t akkor a modul ugytekinti hogy hibas adatok erkeztek es viszater a kiidnulo allapotba. Abban az esetben ha binaris adatokat kuldunk tartalmazhat szpecialis protokol keretezesben resztvevi karaktereket ily ezeket elotti karakter Skip kell hogy legyen. A skip karakterek nem kerulnek bele a memoriaba ezeket a hardver kiveszi.

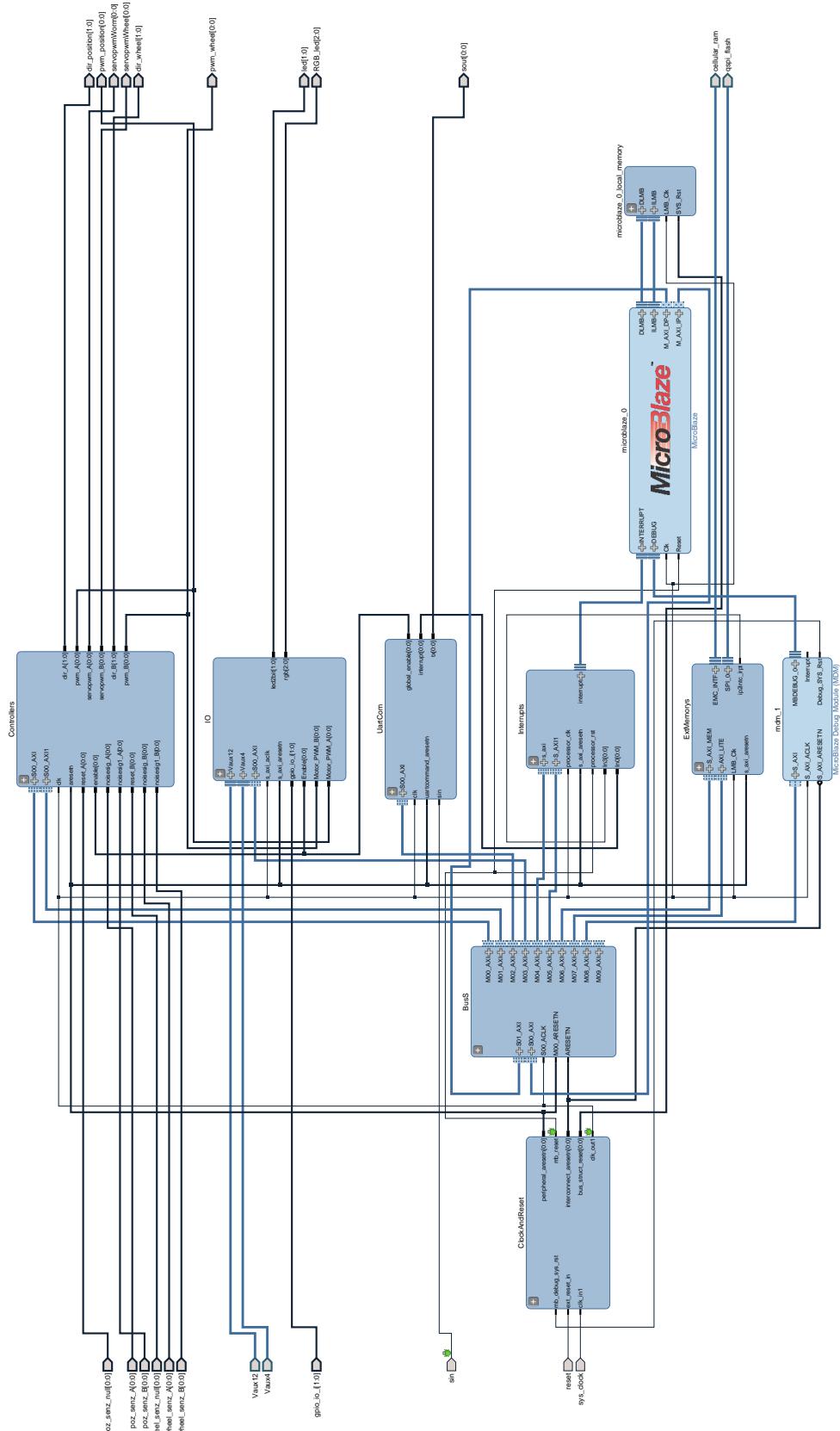
A kommunikacio folyamatat a ábra 3.2.3 abra is vegigkovethetjuk.



3.2. ábra. FPGA ban megvalositott komunikacios protokol a 3.2.2 leirt protokol alapjan.



3.3. ábra. FPGA hardver/MicroBlaze proceszor es ROS node kozti kommunikacio megvalistiasa UART protokol alapjan



3.4. ábra. FPGA ban megvalositott szoftproceszor rendszer, legfelső negyzet.

A keret veget jelzo byte erkezesekor a modul general egy megszakitast a microBlaze proceszornak az *interrupt*[0 : 0] kivezetesen keresztul. A megszakitas kiszolgolasakor a proceszor kiolvasa az *uartcommand*_0 modulbol az uj uzenet kezdocimet. A proceszor az adatokat az *axi_bram_ctrl_1* modulon keresztul kepes kiolvasni, az uj uzenetre mutato pointer ertekeket megkapjuk, ha az *uartcommand*_0 modul proceszor memoriajaba illesztett kezdocimet es a modul altal jelzet kezdocimet osszeadjuk. Abban az esetben, ha az adatok feldolgozasa elott uj csomag erkezik a hardver komunikacios modulhoz, elkezdodik annak beirasa a memoriaba, az elozo adatok felulnemirasaval.

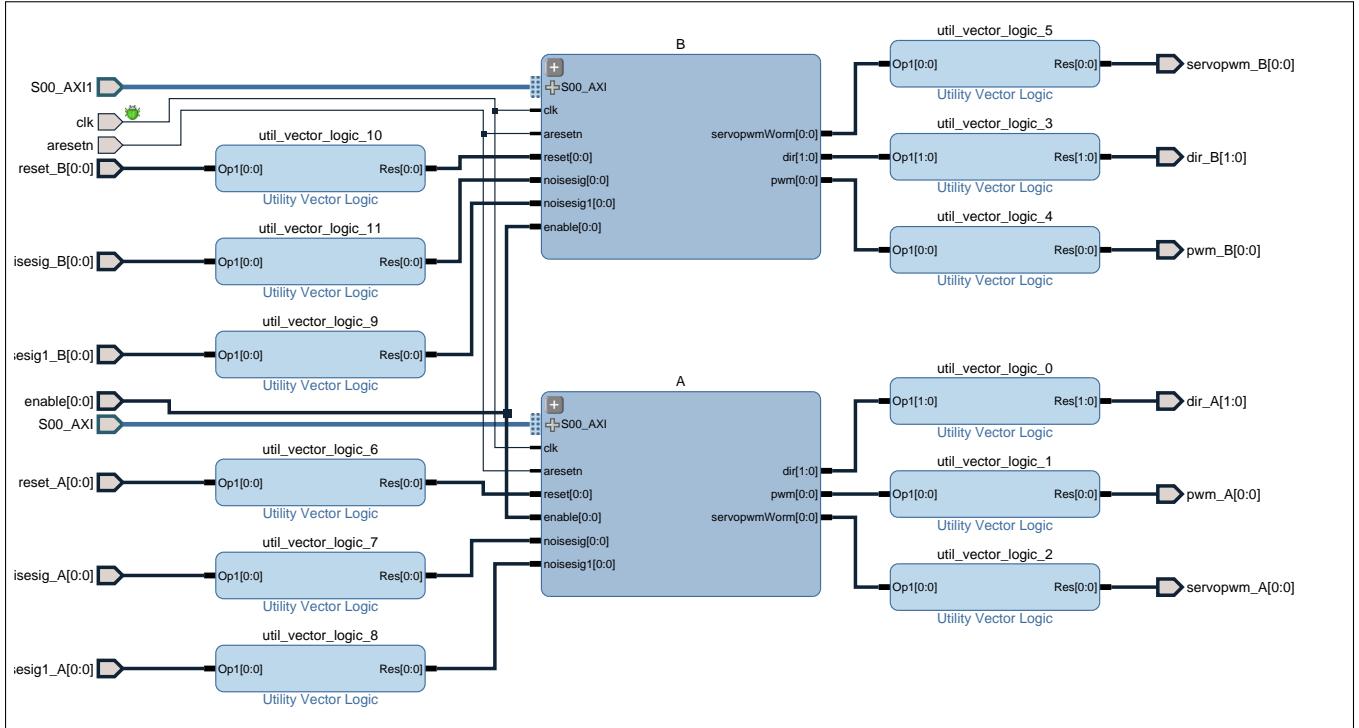
A *uartcommand*_0 modul kimenete *global_enable*[0 : 0] engedelyezo jel, abban az esetben ha nemkap a hardver *SREP* uzenetet legalabb 300ms kent akkor az engedelyezo jelet alacsonyra alitja vagy ha *SRDP* uzenet erkezett.

Az *uartsend_0* modul hasonlokeppen mukodik csak az adatok kuldesere hivatott. A microBlaze proceszor az *axi_bram_ctrl_0* modulon keresztul beleirodnak a *blk_mem_gen_1* [24] 32bites es 400byte meretu meoriaba. Az iras vegevel a proceszor bealitja a kuldes felget, amely egy parameter a hardvernek. A modul elkezdi kikuldeni az adatokat. Abban az esetben ha a csomag tartalamz szpecialis protokolt leiro byte-t akkor a modul automataikusan elkuldi elote a Skip karaktert.

Az *uartread_0* az UART protokol implemetalja FPGA hardverebe, a *sin* az RX bemenet, a *data*[7 : 0] a 8bite szeles adatvezetek ami tartalmaza az utolso berkezet byte uzenetet. A *dataready*[0 : 0] felfuto el jelzi az uj adat erkezeset. Az *uartwrite_0* modul a *data*[7 : 0] 8bites sinen erkezo adatot kului ki UART protokol alapjan a *tx*[0 : 0] kimenetere abban az esetben ha a *datavalid*[0 : 0] bemeneten egy felfuto el erkezik.

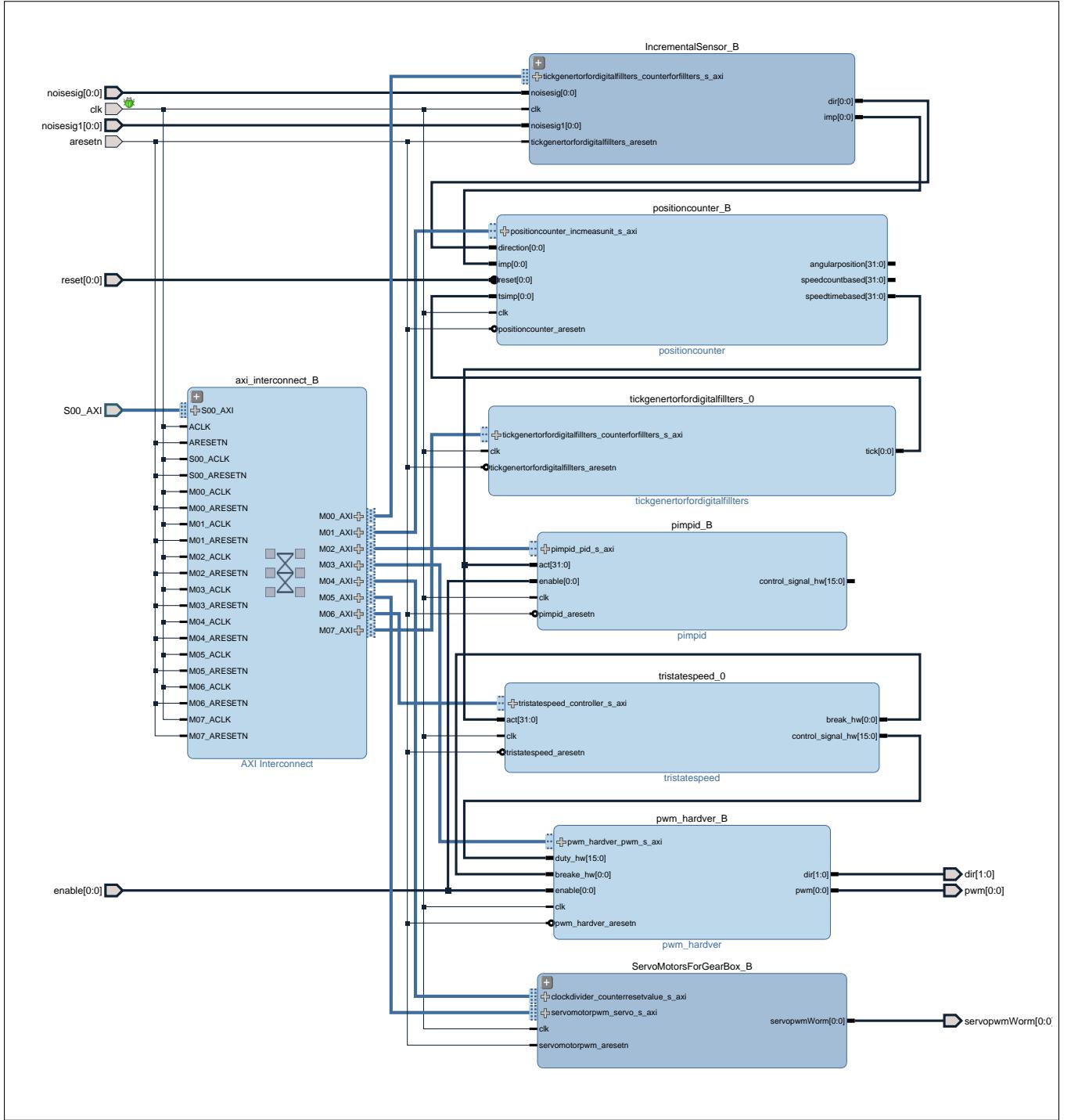
Az ábra 3.2.5 hathato *A* es *B* modul feladatuk egy-egy DC motor szabalyzasat lassak el. Mindket modulban ugyan azon alegysegek talalhatok meg az *A* modul az robot else, mig a *B* modul a robot hatso kereket szabalyozza.

A ábra 3.2.6 abran lathato a modulok belso szerkezete. A *pwm_hardver_B* PWM jelet allit a *pwm*[0 : 0] kimeneten, es egy iranyjelet *dir*[0 : 0] a bemeneti 16bites elojeles szambol. A PWM kitoltesi tenyezoje -32000 tol 32000 ig terjed ki az elojele meghatarozza az iranyt mig az abszolut erteke a pwm kitoltesi ertekeket. A *enable*[0 : 0] jel engedelyezi a kimenetet, ha erteke alacsony akkor a pwm kimenet is alacsony.



3.5. ábra. FPGA ban megvalositott szabalyzok A es B

Az *InkrementalisSensor_B* axi sinen keresztul kapcsolódik a proceszorhoz, feladata az inkrementális szenzortól erkező A és B jeleknek a feldolgozása. A modul két jelet küld ki: $dir[0 : 0]$ amely a forgás irányát jelzi, $imp[0 : 0]$ egy orajelperiodusig tartó felfutojelet küld ki az inkrementális szenzor minden egyes elmozdulsára.



3.6. ábra. FPGA controller mag.

A *positioncounter_B* modul meri a szögsebeseget és a szogpoziciót idő és impulzus számolási módszereket használva. A proceszor a saját mintaavetelézési frekvenciával olvassa ki a modulból a mert értékeket és alakítja át a kivánt mértékégszégekbe.

Az ábra 3.2.6 abban latható többi modul jelen pilanatban továbbfejlesztési lehetőségekkel jeleni meg. A *pimpid_B* modul FPGA alapú PID szabalyzó modul, *ServoMotorGearBox_B* a DC motrokon levo fogaskerek kapcsolási lehetőséget hivatott majd alitan egy szervomotor segítségével.

A szabalyzók referencia értékeit a *RefValues* funkcionális biztosítja, a megfelelő üzemetek érkezésekor a memoriába beállítodnak a kivánt referencia értékek, és majd a szoftve-

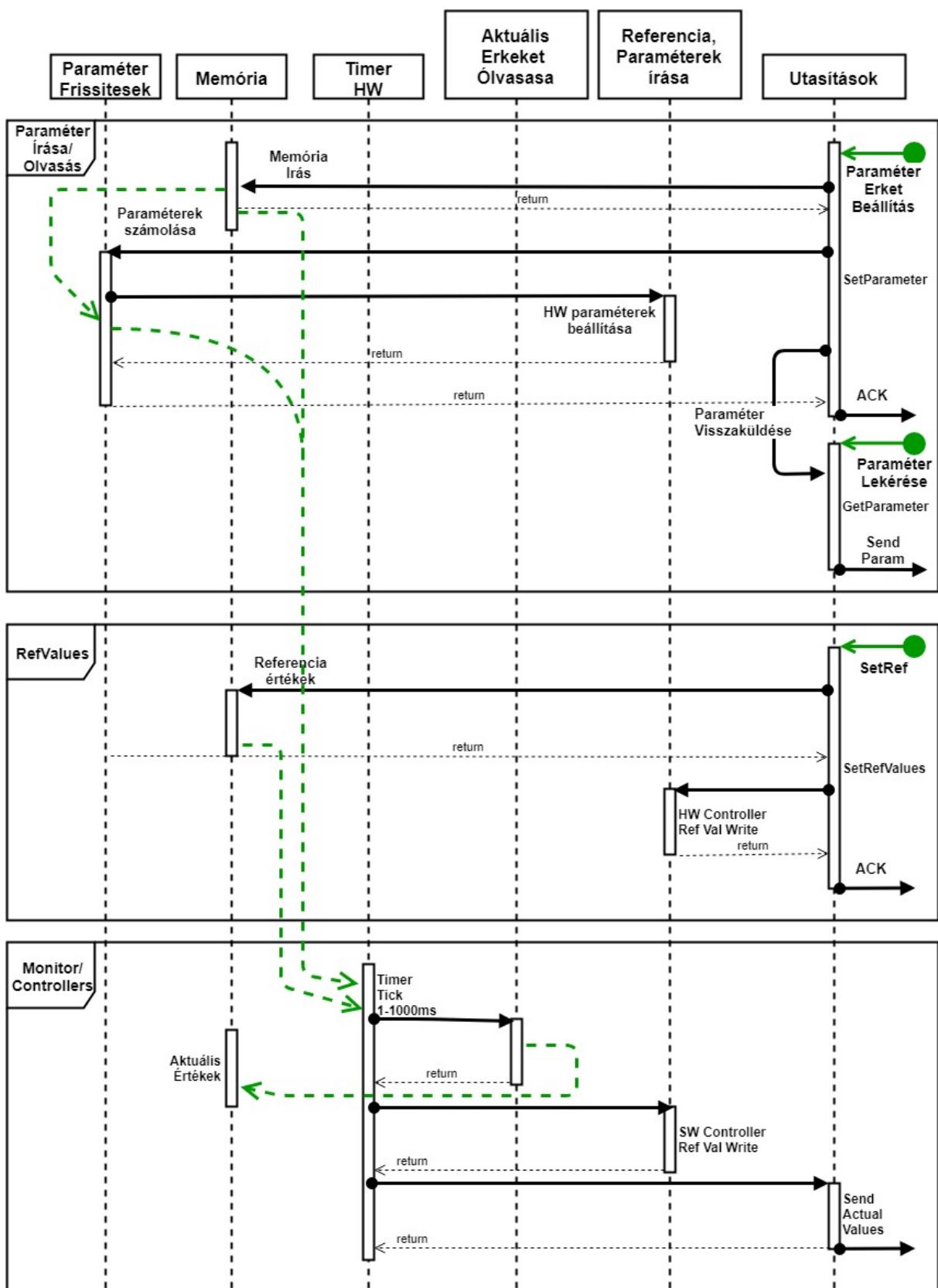
res szabalyzok innet olvasak ki. A hardveres szabalyzoknak pedig a beerkezes pillanatban lekuldodik a hardveres moduloknak.

A *Monitors/Controllers* funkcionalitas egy parameterbol allithato periodusu idozito amely megszakitasokat general a proceszornak. A proceszor ezekre a megszakitasokra szamolja ki a beavatkozi jelek uj ertekeit, valamint a elkului a mert,es a kapott referencia ertekeket UART on keresztul.

3.2.1. Microblaze szoftvere

A rendszer tervezesenel a fo koncepcio az volt hogy a rendszer arhitekturaja dinamikus legyen a fejleszhetoseget tekitve, ily a ábra 3.2.1 levo arhitekurat kaptuk.

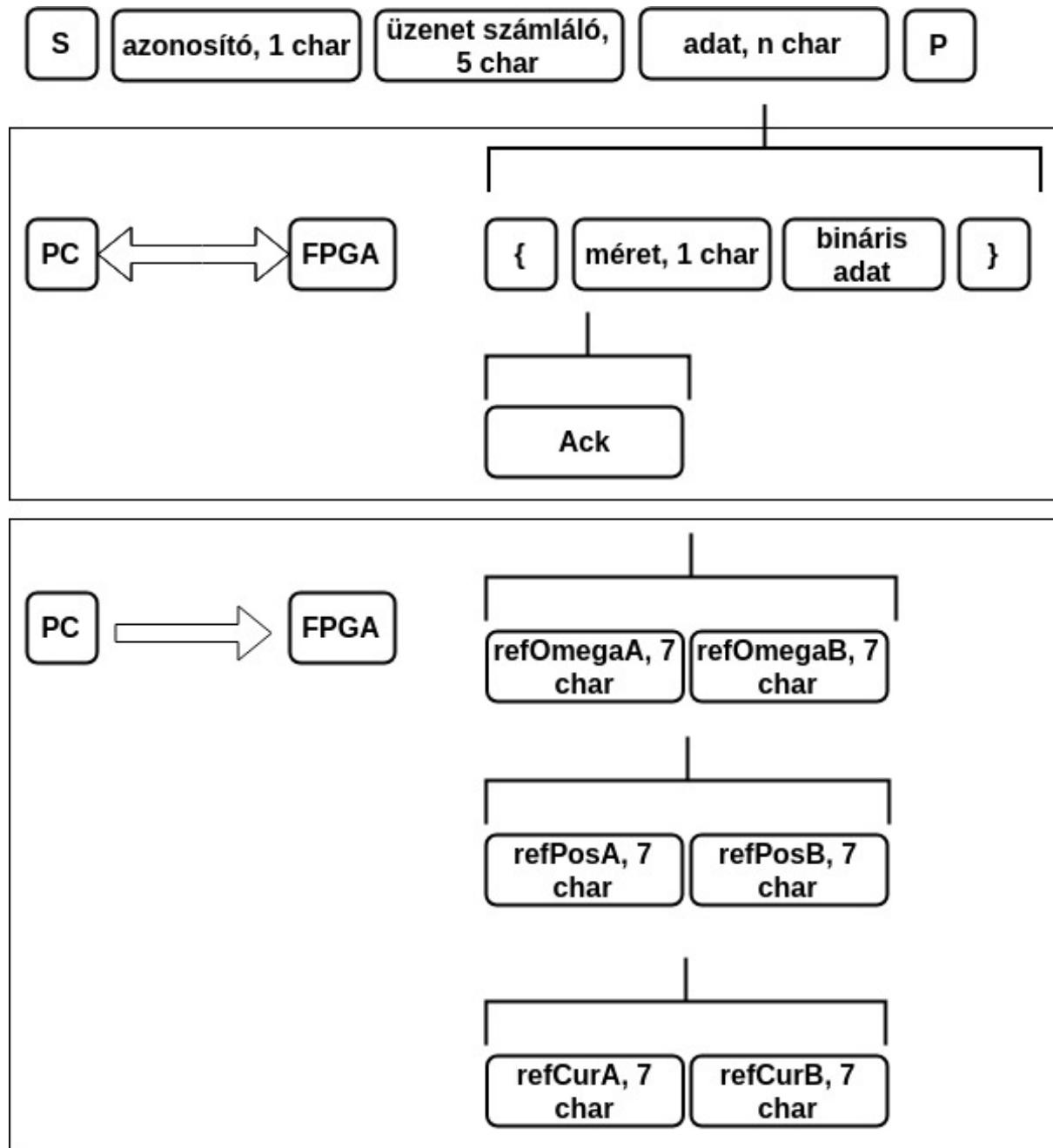
A microblaze proceszor feladatai hardveres modulok kezdeti allapotainak a beallitasa, szoftveresen furt szabalyzot mukdtetese, az beerkezett uzenetek feldolgozasa. A proceszoron futo szoftvert az ábra 3.2.7 UML diagram szemleteti. A peremeterek irasa es olvasasa funkcionalisban, az ujonan erkezet parameter ertkenek a bealitasi mente a kovetkezo: az uzenet beleirodik a memoriaba, megszakitas generalodik a proceszor iranyaban, a procesyor feldolgozza az uj uzenetet, eszleli hogy parameter ertkenek a bealitasa tipusu uzenet erkezett, meghivodik az eszt kiszolgalo fuggveny. A tovabiakkban az osszes parameter ujraszamolodik, es ujrakuldodik a hardveres modulok iranyaba. Abban az esetben ha minden sikeresen zajlot a proceszor pozitiv viszajelzst(ACK) uzenetet kuld es viszakulde a bealitott parameter ertekeket, acelbol hogy a kuldo is megbizonyosodhadjon a parameter helyes bealitasarol.



3.7. ábra. MicroBlaze proceszoron futó szoftver diagramja

3.2.2. FPGA es UART alapu komunikacios protokol

Megvalositva a komunikaciót a kiszolgalo ROS noddal, amely UART protokolra épített saját uzenetekbol all ábra 3.2.8.



3.8. ábra. FPGA komunikacios protokol altalanos csomag szerkezet

A protokol keretek koze foglalt adatok sorrendisegere epul. Szpecialis karakterek¹ jelzik az uzenet kezdetet es veget jelen esetben az *S* csomag kezdetet, mig a *P* a csomag veget

¹ specialis karakterek amelyek jelzik az ertelmezeseben nem kell vegrehajtani.

mezo szamara hogy olyan karakter kovetke-

jelentik. Az csomag tartalmazhat binaris formatumo adatot is ezeket minden esetben a { es } szpecialis karakterek koze kel kerulniuk, mert a koztuk levo adat nem fog ertelemzesre kerulni a feldolgozo altal. Binaris reszt pl.: struktura tipusu adatra hasznalhatjuk. Mindig a { utani elso karakter megadja a binaris adat hosszat ily tudja az ertelmezo hogy hol kell varnia a lezaro karaktert. A binaris szekco hossza nemlehet nagyobb mint 254 byte hosszu, de amenyiben szukseg es kiterjeszheto.

Minden uzenet tartalmaz egy egyedi azonostot ami meghatarozza az uzenet tipusat ez alapjan tudja eldonteni majd a fogad fel hogy melyik kiszolgalo rutint kell felhivnia. Ezt koveti egy uzenet szamlalo 5 char hoszu mezo amely string formaban tartalmez egy egesz szamot, amely minden kikuldott uzenet utan novelni kell. Az uzenetszamlalo alol kivetelt kepeznek a direkt hardveres uzenetek pl.: *SREP* es a *SRDP*. A hasznos adat további fele-piteset midontjuk el annak fugvenyeben hogy mit szeretnek. Jelen esetben a uzenetipus orientalt parancsokat szerkesztunk amelyekkel a referencia ertekeket, parametereket allit-juk be, valamint binaris szekciot is tartalmazo strukturat kul az FPGA a PC nek amely a szabeszest es monitorizalast szolgalyja.

3.2.3. Paramterek FPGA modul

A parameterek segitsegevet tudjuk bealitani a mintavetelesi periodusokat, az inkrementalis szenzorok felbontasat, a szabalyozok bealitasait is ezaltal oldhatjuk meg. Configuracios parameterek.: valaszthatunk szabalyzotipusokat. Az alábbi tablazatban egy leirjuk a ROS-banhasznalt parametereket. A paremeterek nevenek vegen levo X jelolje A vagy B, attol fuggoen hogy melyik DC motorhajtashoz tartozik. Az ábra 3.2.5abran lathato modulokat tudjuk configuralni.

Id	Nev (X lehet A vagy B)	Ertekek		Tipus	Leiras
		Min	Max		
1	TsTimerPeriod	1	1000	int16	Mintavetelezesi periodus [ms] ban.
2	GetDataPeriodical	0	1	int16	Kapcsolo ha 0 akkor nem kuld az FPGA mert ertekeket, kulomben a TsTimerPeriod periodusi mintavetellel kuld.
3	TorqueCoefX			float16	Motor aram es nyomatek kozti egyuthato.
4	ActiveControllerX	0	65535	int16	Valaszhato szabalyzo tipusok hajtasonkent 0=Szoftvare PID szogsebesseg, 1=Hardver PID szogsebesseg, 2=Szoftver PID aram, 3=Hardver PID aram
5	MaxControlSiggnaX	0	32760	sint16	A beavatkozo PWM jel maximalis kitoltesi tenyezoje, linearisan 0->0% -tol 32760->100% -ig.
6	IncSenzResX	0	65535	int16	Inkrementalis szenzor altal generalt impulzusok szama egy teljes kerekfordulatra.FPGA oldalon ez a szam 10-el szorzodik.
7	IncSenzCountDirectionX	-1	1	sint16	inkrementalis szenzor jeleit feldogozo mudul szamolas iranyanak valtoztatasara szolgalo egyuthato.
8	Kp_Whell_PidX	0		float16	szogsebeseg szabalyzo, PID erositesi parametere.
9	Ti_Whell_PidX			float16	szogsebeseg szabalyzo, PID integralasi ido.
10	Td_Whell_PidX			float16	szogsebeseg szabalyzo, PID derivalasi ido.

3.2.4. Kommunikáció sebessége

Az uart sebesege 1MBd ² ami megfelel 131072 byte/s adatforgalomnak. A valosagban a komunikacio hibatlanul mukodik 1ms periodussal kuldott 100byte hoszu uzeneteket. Osszehasonlitva *AXI_UART_Lite* [25] modullal elert eredményekkel 50ms periodust tudtunk csak elerni ammi annak tudhato be hogy a protoko ertelmezeset a szoftver vegezte es nem a hardver. A FIFO hasznalata elenere sem skerult nagyobb mintavetelezesi sebeseget elerni, hogy nem minden karakter utan generalt proceszor megszakitast csak minden 16-dik karakter utan.

$$frekvencia = \frac{131072}{SizeOfPachage} = \frac{131072[byte/s]}{100[byte]} = 1310,072[Hz] \quad (3.1)$$

² megabaud

3.2.5. Biztonsagi megoldasok

Abban az esetben ha kikultuk az elirt ertekeket a modulnak es ezutan a komunikacio megszakadt a modullal akkor a szabalyzok probaljak tartani az elirt ertekeket annak elenere is hogy az mar lehet hogy ez mar nem akktualis. Erre a celra bepiresre kerult egy uzenet es egy logika (HeartBeat) amely csak a komunikacios modulhoz erkezik meg, es jelzi hogy a ROS jolmukodik, es forditva is. Abban az esetben ha a komunikacio megszakad lealitja a motrokat es a szabalyzokat, ábra 3.2.3 alapjan az Enable(EN) jel.

Irany	Uzenet	Periodis
FPGA->ROS	SEP	300 ms kotelezo
ROS->FPGA	mintavetelezett ertekek kuldese	dinamikusan modosithato

3.3. ROS

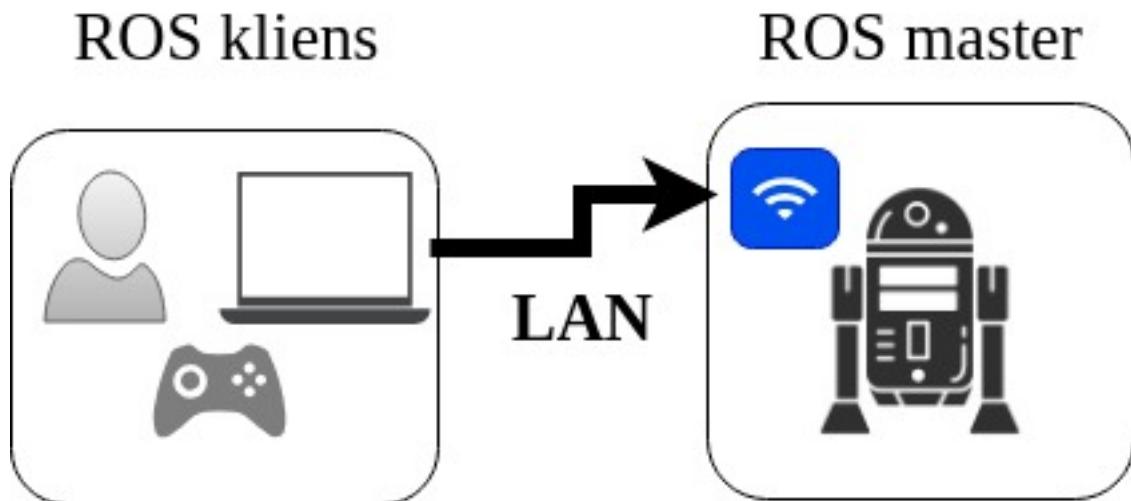
A roboton levő számítógépen Ubuntu 16.4 linux operációs rendszer fut, és ROS lunar keret rendszerben fejlesztet programok megoldják: -FPGA ROS kommunikációt, -távirányítás, -heartbeat biztonsági funkcionálitás,-IMU szenzor adatainak a beategyüttese,-terképezés és lokalizálás LIDAR alapján.

A robot irányítását megtehetjük azáltal hogy csatlakozunk a roboton wifin keresztül a robothoz. Az irányítashoz szükséges egy számítógép amelyre kell legyen telepítve a ROS. A kilens gépen fognak futni a következő felhasználói eszközök: -a rendszerben futó programok által loggolt üzeneteket szurhatjuk és tekinthetjük meg valósidobban. -parametereket beállíthatjuk egy grafikus felhasználói felület segítségével (*dynamically_reconfigure*).-rviz amelyel megjeleníti a robot 3D modeljét a terépen a robot aktualis pozíciójában és irányában a környezetéhez viszonyítva. -*JuglerPlot* segítségele online ábrázolhatuk mert értékeket mint pl.: a szabályzókorokban mert értékeket.

Jelenleg ket működési mod kozul választhatunk:

-tavirányitorol irányítva.A tavirányiton található egy joystick amely segítségevel a robotnak referencia sebesegeket irhatunk elő. A hozzárendelt referencia rendszerhez képest az X tengelyen m/s sebeseget adunk meg, mig a Z tengelyre koruk °/s forgási sebeseget adunk meg. A tavirányiton get gomolyamatosan lenyomva tartásával a deadman kapcsolat valósítatjuk meg. Abban az esetben ha minden gomb nincsen lenyomva a szabályzó működése és a pwm generatorok kimenete letiltva marad.

-automata mod, itt rviz program segítségevel előiránytunk egy referencia pozíciót és irányt ahova szeretnénk hogy a robot eljusson. Ezalatt a tavirányiton a deadman kapcsolat lenyomva kell tartanunk. A referencia értékeket megadhatjuk rviz használata nélkül azáltal hogy a megfelelő típusú és nevű üzenetet beküldjük a ROS rendszerbe.



3.9. ábra. Robothoz csatlakozás a Wifi-n keresztül.

Az ábra 3.3.10 láthatóak a főbb nodok, topikok és a közöttük levő relációk. A nodok és

a topikok leírását a ??

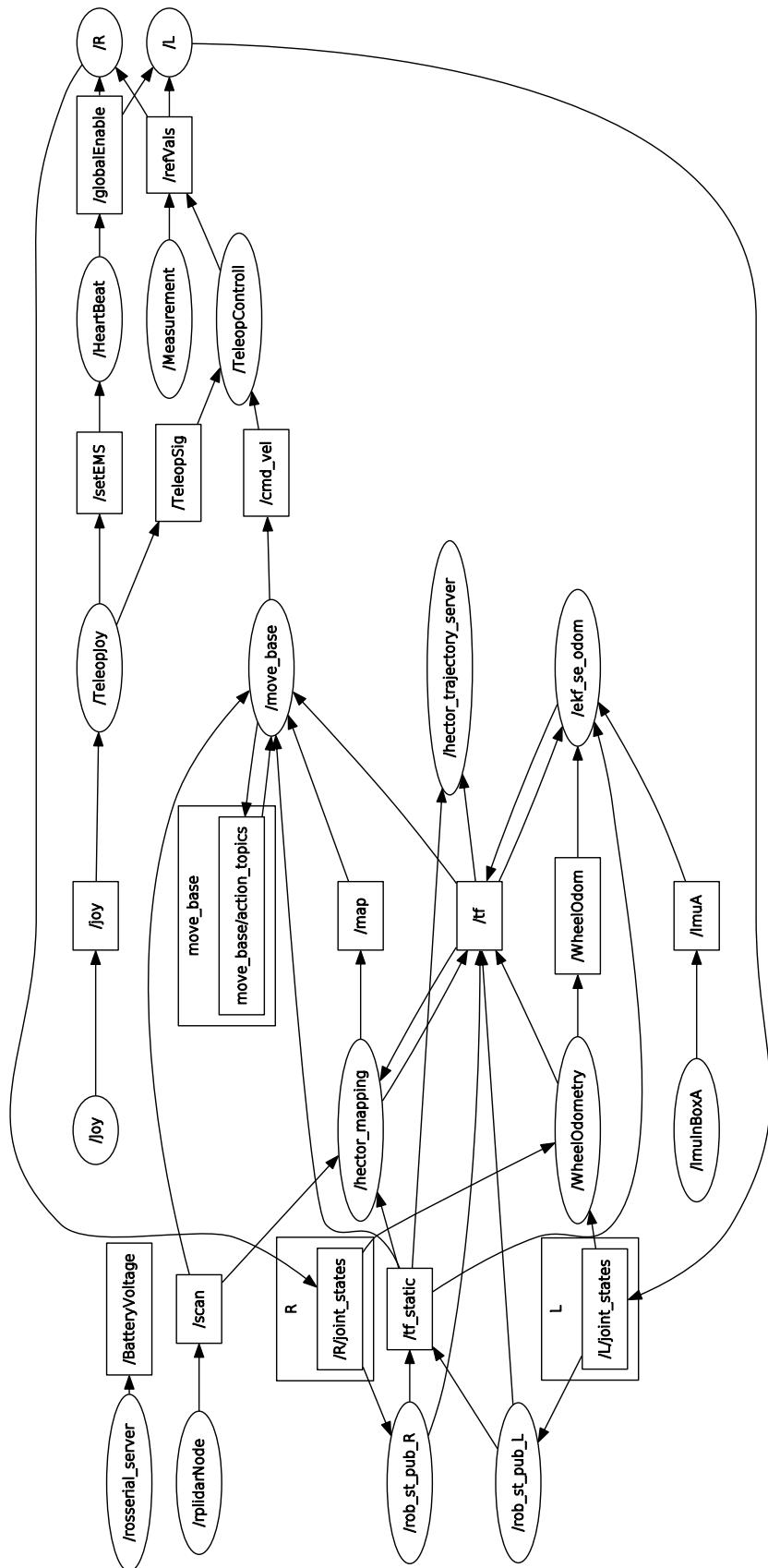
Node Nev	Tipus	Leírás
/R es /L	FPGA Communication Modul	Az FPGA-val való kommunikációért felelős. Adatokat fogad/küld a ?? fejezetben leírt protokoll alapján, amelyeket továbbít a ROS operációs rendszerben működő nodoknak.
/ImuInBoxA	Imuxxxx	Feladata szöveges formában érkező adatok feldolgozása és a ROS keretrendszerbe integrálását oldja meg. Mert fizikai mennyiségek:
/WheelOdometry		Kerekek mozgásából számolt robot elméleti pozíciója a térben
/TeleopJot		Feladata a joystick-tol érkező parancsok fogadása. Megvalósítja a <i>DeadManSwitch</i> gomb kezelését létrehoz egy globálisan engedélyező jelet a /setEMS-t és a /TeleopSigt amely előírja a robot linearis mozgási sebességét és a forgási sebességét a Z tengely körül.
/rplidarNode		A lidar merési adatait olvassa ki és továbbítja a /scan topikban.
/rosserial_server		A megvalósítja a kommunikációt egy esp8266 fejlesztőlap és a ROS között amely az akkumulátorok feszültségeinek a merését végzi [7]. Joystick eszközök integrációját valósítja meg [18].
/Joy		Lidar merései alapján 2D térképet készít a környezetről, miközben lokalizálja a robotot ezen a térképen [13]
/hector_mapping		merésék elvégzésére szolgáló node, amely egy előre beállított intervallumokban az megadat referenciaértékeket küld ki az FPGA ban levő szabályzóknak.
/Measurement		Átalakítja a robot sebesség állapotait és kiszámolja nyílt hurokban a szabályzok előírt értékeit. Abban az esetben ha a /movebase nodot használjuk, ez megoldja a robot pozíció szabályzását a térképen, így a /cmd,el csomagot csak átalakítja /refVals csomaggá azáltal hogy azonos oldalon levő kereke ugyanazt a referencia értéket kell követniük.
/TelepoController		

3.3.1. Uzenet tipusok (.msg)

Az alábbi táblázatban láthatjuk a ROS operációs rendszer által szolgáltatott .msg üzenetek kiterjesztését amelyek lehetővé teszik az FPGA integrációját a ROS környezethez.

Üzenet típus	Értékek	Erkek típusa	Leírás
header	-	std_msgs/Header	Minden üzenet tartalmaz egy fejlécét amely információkat tartalmaz az üzenetről.
	seq	uint32	minden üzenetet egyedileg beazonosító szám
	stamp	time	idő-bélyeg amely a küldés időpillanatát tárolja.
	frame_id	string	
/GlobalEnable	systemIsOk	int16	=0 - a HLC működik. <>0 - a HLC nem működik
/refVals	names	string[]	
	ref_position	float[]	előírt szög pozíció
	ref_velocity	float[]	előírt szög sebesség
	ref_effort	float[]	előírt forgatónyomaték
/setEMS	value	int16	=0 - Vészleállító aktív. <>0 - Vészleállító nem aktív
/joyControll	vx	float64	A robot X tengely mentén előírt sebessége m/s-ban.
	omega	float64	előírt szög pozíció A robot Z tengely körüli forgása °/s-ban.
	ControlMode	int64	Választhatunk a HLC szabályzok típusa vagy a manuális irányítás közül =0 move_base szabályzó, =1 manuális irányítás joystick segítségével.

A ?? abban lathatjuk a nodok es az uzenetek kozti kapcsolatot.



3.10. ábra. ROS graph

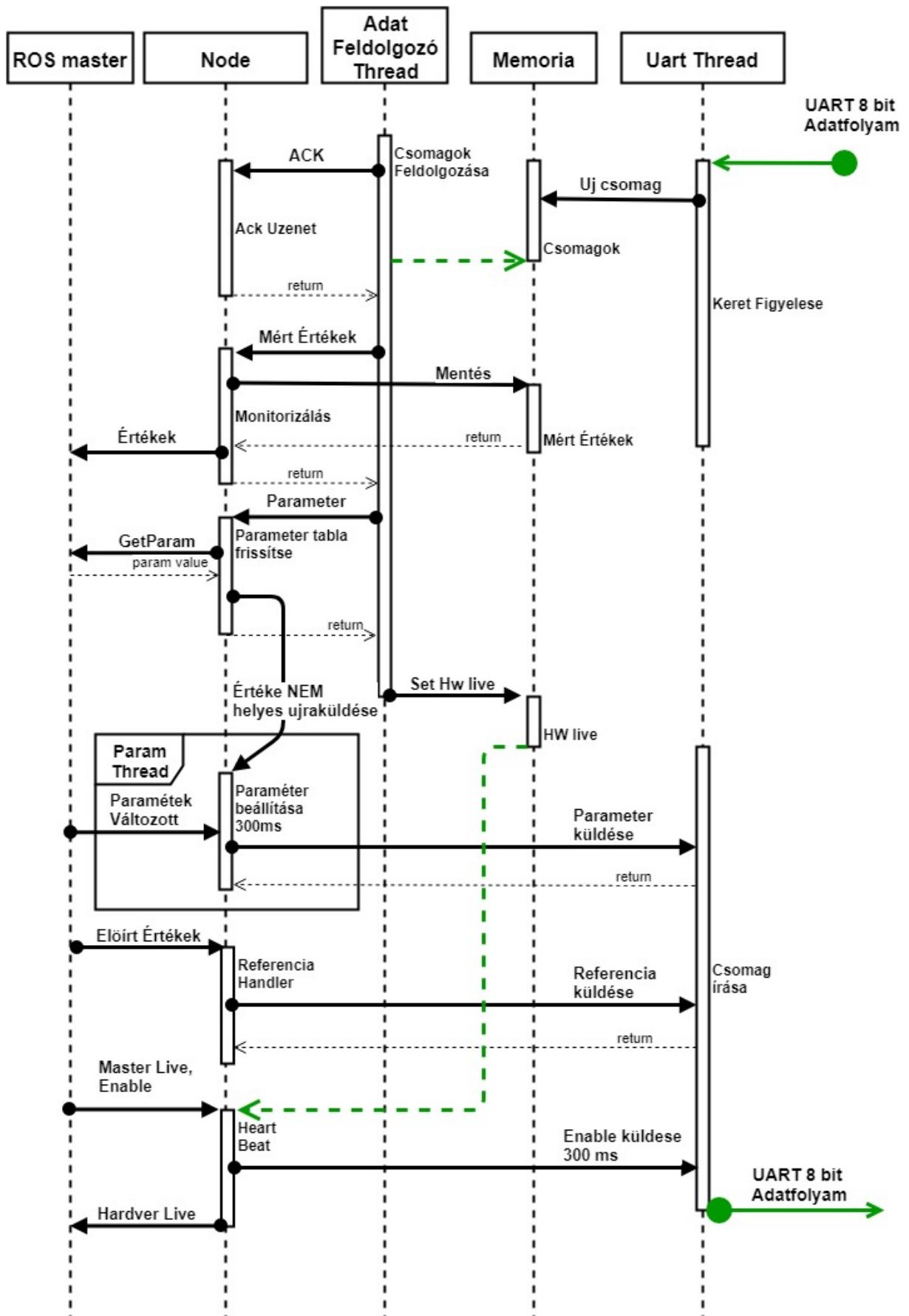
3.3.2. FPGA komunikacios modul ROS oldali integracio

A ROS biztosít a fejlesztőknek egy megoldást amelyek kepések újonnan létrehozót robot integrálását ROS környezetben [7]. Előnye hogy gyorsan látványos eredményt érhetünk el, de a működés sebessége és az üzenetek méretében is korlátozott. Ezen hátrányokból kifolyólag sajátos integráció szükséges amely integrálta az FPGA UART kommunikáció protokollt a ROS keretrendszerben működő más modulokhoz.

A ábra 3.3.11 diagram a kommunikáció node technikai megvalósítását lathatjuk. Különálló szál gondoskodik az UART adatok olvasásáról és írásáról. Az üzenetek értelemezését egy külön szál végzi és hívja fel a kiszolgáló fövenyeket. A paraméterek helyes beállításáról a ParamThread szál gondoskodik, paraméterek helyes beállításáról FPGA oldalon. Abban az esetben ha a hardver kap egy új paramétert a ábra 3.2.7 alapján az FPGA visszaküldi a kapott paramétert, a visszajelzésből a eldönthető hogy a paraméter a hardverben helyesen beállítódott-e. Abban az esetben ha nem megfelelő újraküldődik mindaddig amíg nem sikeres a beállítás.

A paraméterek kezelésére a ROS paraméter szerver a felelős [20], abban az esetben ha egy paraméter megváltozott amely az illető nodehoz köthető akkor a ábra 3.3.11 ábrán látható ParameterValtozott esemény előidézi a megváltozott paraméter ertekenek az elküldését FPGAnak irányába.

A globális engedélyező jel a ábra 3.3.11 MasterLive Enable, /globalEnable típusú üzenettel engedélyezhetjük a szabályzok működését, a folyamatos működéshez 500ms periódussal kell érkeznie. Abban az esetben ha a központi számítógép valami okból leállna akkor a hardveres szabályzok is leállnak. A node 300ms periódussal küldi tovább az engedélyező jelet az FPGA modulnak. A HardverLive jel információt szolgáltat a többi ROS környezetben futó és a működés szempontjából kritikus nodenak hogy az adott modul megfelelően működik e. Ezen információ birtokában a HeartBeat node leállítja a rendszert ha egyik FPGA modul nem válaszol.



3.11. ábra. ROS integrálása Uart protokolhoz.

3.3.3. Előirt értékek

A /refVals tipusu üzenetben megadjuk minden egyes motor előírt érteket annak fövenyében °/s ha sebesség alapján szabályzunk vagy N/m előírt nyomaték alapján szabályzunk.

3.3.4. Vonatkoztatási Rendszerek

A vonatkoztatási rendszereket szükségesek mert a szenzorok és beavatkozó eszközök egymáshoz viszonyított helyzete és pozicioja is változhat. Sok esetben szükséges ismernünk egy adott eszköznek a múltbeli helyzetét, vagy egy másik vonatkoztatási rendszerhez képest a pozíóját vagy irányát. A ROS biztosít egy tf [22] nevű csomagot amely megvalósítja a szükséges transzformációkat a VNR-k kozott. A ábra 3.3.12 látható a kialakított vonatkoztatási rendszerek a roboton amely hűen modellez a fizikai robot kialakítását. A vonatkoztatási rendszereket két csoportba oszthatok:

- (a) rögzített pozíció és szögek, szabadságok 0: Szenzorok laser, BODY_link, wheel_odom, ImuALink VNR je a base_link a globális robot VNR höz:
- (b) rögzített pozíció csak szögek változnak, szabadságok 1: Kerekek VNR je: FL_link, BL_link, FR_link, BR_link a BODY_link hez képest csak Y korú foroghat.
- (c) pozíció és szög is változik, szabadságok 6: A robot base_link az helymeghatározás odom, és az odometria a térképhez map viszonyítva.

A robot modellt ROS környezetben URDF robot leíró, xml alapú fájjal lehetjük meg, [10] [11] [9]. Az <origin> tag az xzy paraméter alatt megadhatjuk a csukló pozíóját mindenkor tengelyen méterben kifejezve a <parent> tagban szereplő linkhez képest. Az rpy paraméter alatt az elfordulásokat rendre x, y, z tengelyek mentén radiánban kifejezve. Az <axis> tagban beállíthatjuk a kényszereket, jelen esetben csak az y tengely körüli forgás engedélyezett a kerekek esetében. Az <link> tagban robot elemeket hozhatunk létre. Az alábbi XML-ben látható a robot fizikai leírása amely megfelel a valós szerkezetnek.

```
<robot name="mobile_robot_platform_4Wheel">
    <link name="base_link" > </link>
    <link name="FL_link" > </link>
    <link name="BR_link" > </link>
    <link name="BL_link" > </link>
    <link name="BODY_link"> </link>
    <link name="ImuALink"> </link>
    <link name="laser"> </link>

    <joint name="FL" type="continuous">
        <parent link="BODY_link"/>
        <child link="FL_link"/>
        <origin xyz="0.29 -0.33 0" rpy="0 0 0" />
        <axis xyz="0 1 0" />
    </joint>

    <joint name="FR" type="continuous">
        <parent link="BODY_link"/>
```

```

        <child link="FR_link"/>
        <origin xyz="0.29 0.330 0" rpy="0 0 0" />
        <axis xyz="0 1 0" />
    </joint>

    <joint name="BL" type="continuous">
        <parent link="BODY_link"/>
        <child link="BL_link"/>
        <origin xyz="-0.29 -0.330 0" rpy="0 0 0" />
        <axis xyz="0 1 0" />
    </joint>

    <joint name="BR" type="continuous">
        <parent link="BODY_link"/>
        <child link="BR_link"/>
        <origin xyz="-0.29 0.330 0" rpy="0 0 0" />
        <axis xyz="0 1 0"/>
    </joint>

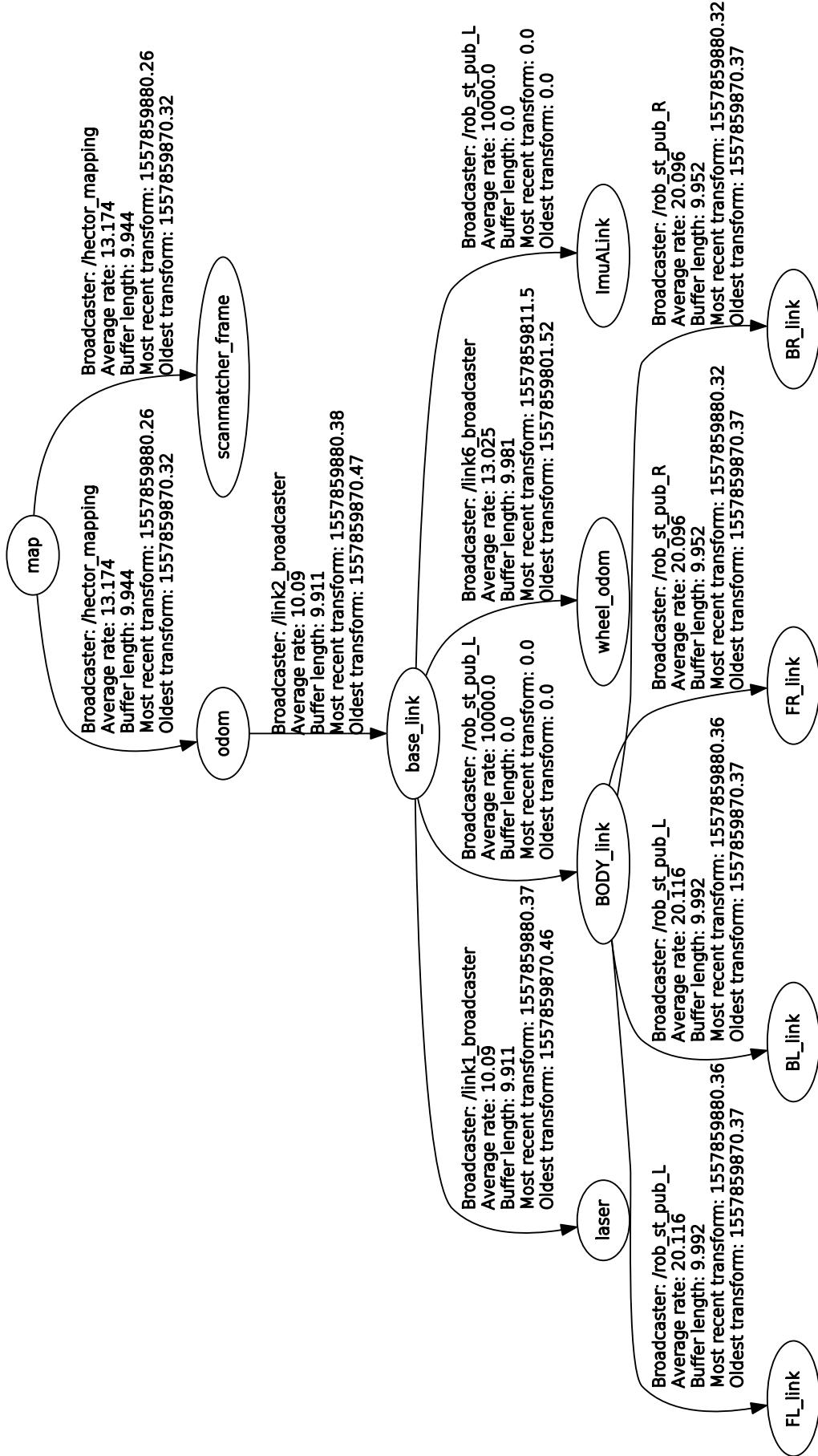
    <joint name="imuAandGPS" type="fixed">
        <parent link="base_link"/>
        <child link="ImuALink"/>
        <origin xyz="0.125 0.03 0.11" rpy="0 0 0" />
    </joint>

    <joint name="laserAJoin" type="fixed">
        <parent link="base_link"/>
        <child link="laser"/>
        <origin xyz="0.39 -0.02 0.23" rpy="0 0 3.14" />
    </joint>

    <joint name="contact" type="fixed">
        <parent link="base_link"/>
        <child link="BODY_link"/>
    </joint>
</robot>
```

A ábra 3.3.12 lathatjuk hogy a robot torzset a *BODY_link* alkotja, amelyhez kapcsolodnak a kerekek: *BL_link*, *FL_link*, *BR_link*, *FR_link*. A *base_link* es a *BODY_link* egybesnek. A szenzorok a *laser* amely a lidarnak fele meg, *ImuALink* IMU szenzor ezek a *base_link*-hez kapcsolodnak. A *map* VNR a terkepnek amelyen meghatarozuk a robot poziciojat *odom*.

Recorded at time: 1557859811.5

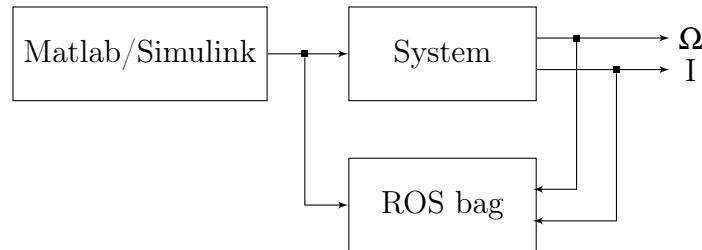


3.12. ábra. A megvalositott robot VNR-k közti reláció

3.4. Kerekek Pid Szabalyzo hangolas

A pid a legelterjedtebb szabályozó egyszerű feladatok elvégzésére, esetünkben is elegendő a kerekek szögsebesség szabályzására kerekenkénti egy PID szabályzóval. A PID szoftveresen fut a uBlaze processzoron. Bemenete egy előírt forgási sebesség °/s ban és kimenete egy -32000 és 32000 egész típusú érték. A kimenti érteke a PWM kitöltési tényezőt jeleni, az előjel pedig a beavatkozás irányát. Matlab/Simulink környezetben használva a Robotix Toolbox segítségével direktben pwm beavatkozó referencia ertekeket írtam elő a motroknak. A beavatkozó jel előállítása és elküldési a fizikai eszköznek 0-100%-ig 10% lépcsőkben amelyek 0% kitoltesékkal vanak megszakítva. A mert adatokat rosbag csomagba mentve majd a System Identification Toolbox használatával identifikáljuk a rendszer modellt. A rendszer bemenete egy beavatkozó jel ami fizikailag feszültségnek fele meg 0V és 12V között. A kimenetek a forgási sebesseg. A mert adatokat Matlab/System Identification használataval megbecsüljük a rendszer modeleket. Nemlinearis modelt becslunk Hammerstein-Wiener model [16] használva, 1 kimenet és 1 bemenet, a linearis átviteli fugveny fokszama: zérusok nb = 2, polusok nf = 3, keses a bemenet és a kimenet között nk = 1. A becsült adatok 94%ban megfelelnek a mert rendszernek.

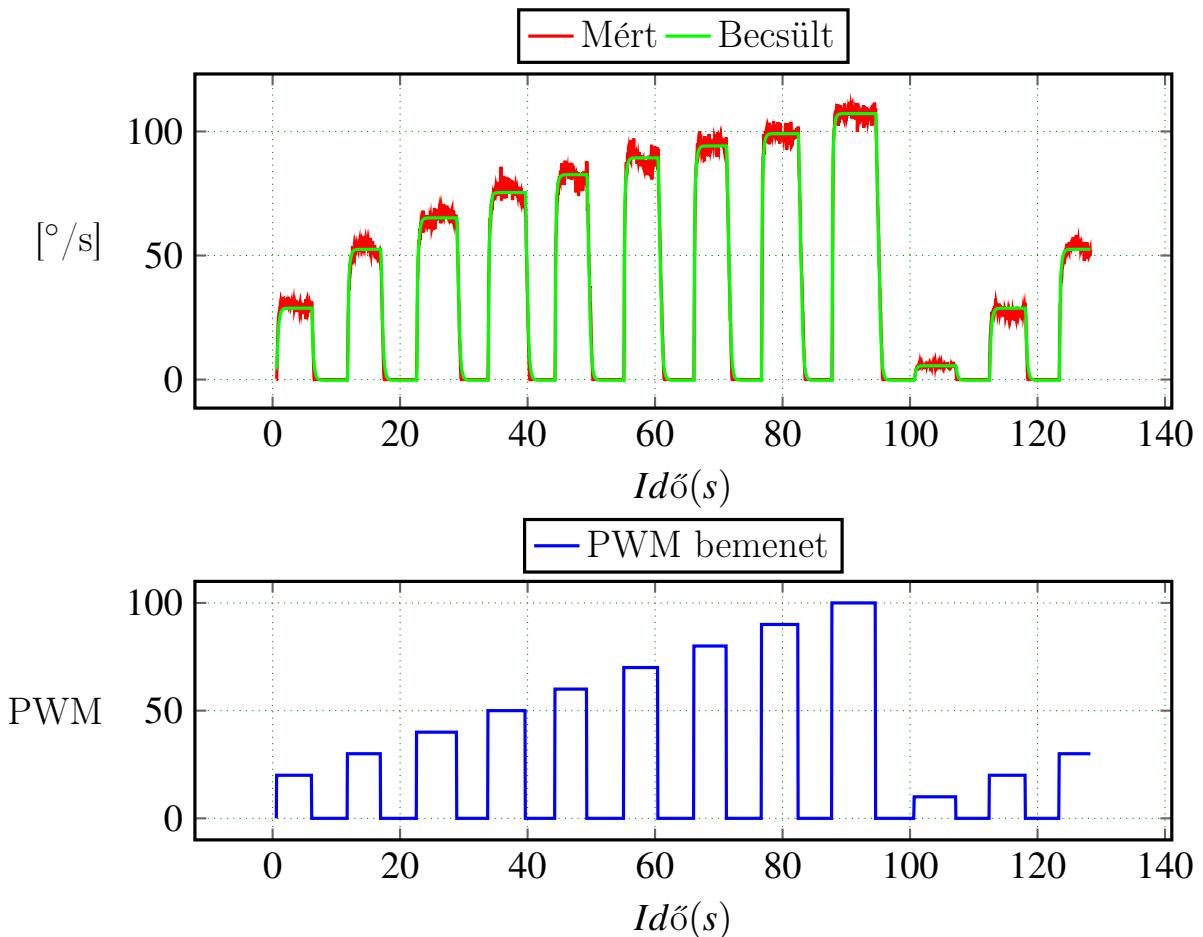
A mereseket a robot kerekei és a talaj erintekzese nélkül vegeztem. A becsült modelt a bemenet 50% korul linearizáljuk és a linearizált modelból átviteli fugvenyt készítünk. $tf = tf(linearize(model, 16000))$; utasutast használva Matlab környezetben. A linearizált modelt Matlab/PidTuning eszközt használva behangolunk kiszámítjuk a megfelelő PID szabályzó paramtereit.



A becsült rendszer átviteli függvénye $H_s(z)$, mintavetelzési periódus $T_s = 0.05s$.

Nagyobik fokozatban

A becsült modelt oszehasonlitva a mert értékekkel a ábra 3.4.13, a model nemlinearis becsült model megfelel a mert értékeknek.



3.13. ábra. Nagy fokozat Hammerstein-Wiener becsült model valasza, es a mert ertekkek.

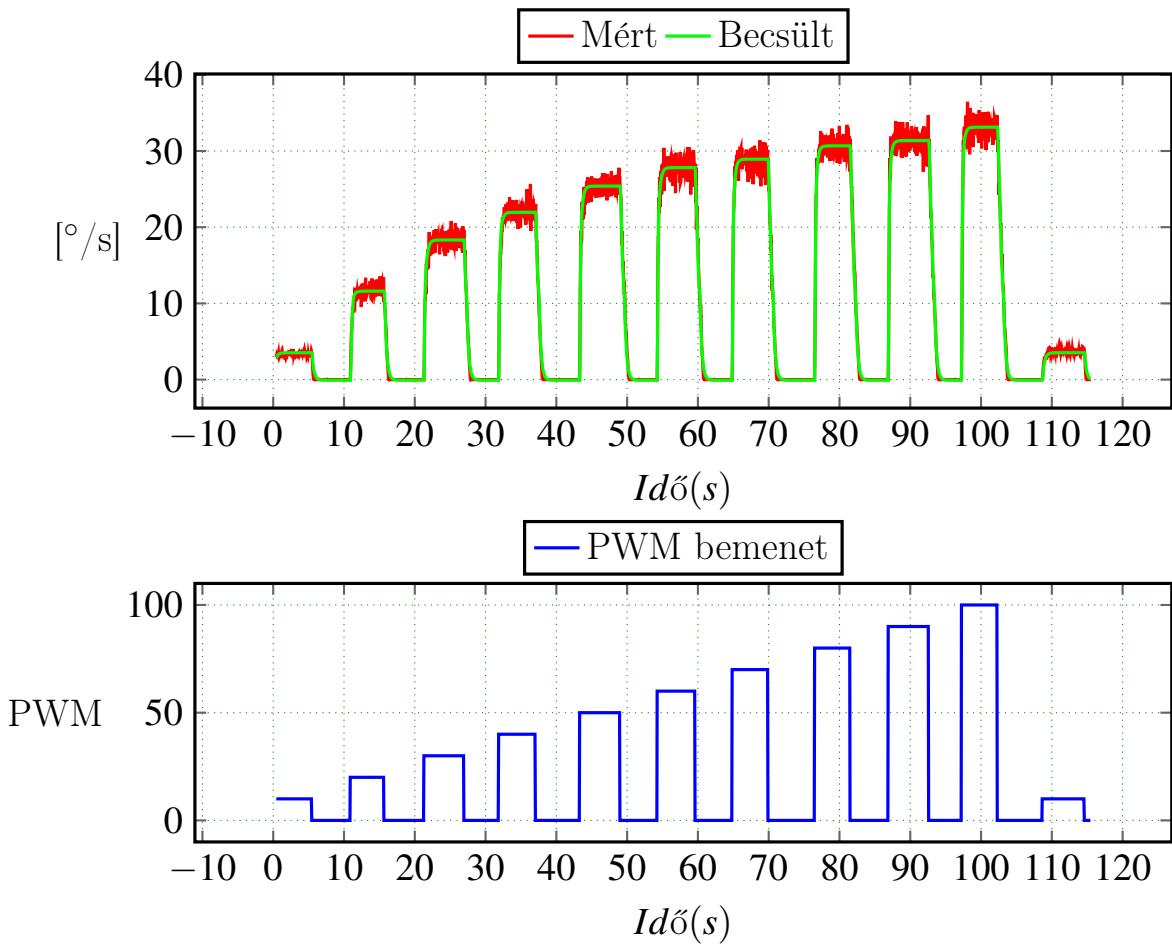
Az atviteli fuggveny a bemenet 50/% korul linearizalva.

$$H_s(z) = \frac{-0.07017z^{-2} - 0.053z^{-1}}{-0.2117^{-3} + 0.7321z^{-2} - 1.393z^{-1} + 1} \quad (3.2)$$

A tervezett PID szabályozó paramétere Kp: 7.11 , Ti: 23.66 , Td: 0.43

3.4.1. Kisebik fokozatban

A becsült modelt oszehasonlitva a mert ertekkel a ábra 3.4.14, a model nemlinearis becsult model megfelel a mert erteknek.



3.14. ábra. Kis fokozat Hammerstein-Wiener becsült model valasza, es a mert ertekek.

Az atviteli fuggveny a bemenet 50/% korul linearizalva.

$$H_s(z) = \frac{-0.0291z^{-2} - 0.009263z^{-1}}{-0.198z^{-3} + 0.7058z^{-2} - 1.394z^{-1} + 1} \quad (3.3)$$

A tervezett PID szabályozó paraméterek: Kp: 15.96 , Ti:51.51 , Td:1.237

3.5. Pályakövettes feladatok

Algorithm 1 Pályakövetes Algoritmusa

```

1: function GetNextControl( $X_a, Y_a, \alpha_a, X_t, Y_t, \alpha_t, Tr, Tr_\alpha, \Omega_{max}, V_{max}$ )
2:    $e_x = X_a - X_t$ 
3:    $e_y = Y_a - Y_t$ 
4:    $d = \sqrt{e_x^2 + e_y^2}$ 
5:    $\alpha_i = dir(X_a, Y_a, X_t, Y_t)$            ▷ Két ponton átmenő egyenes iránytényezője
6:    $e_\alpha = \alpha_i - \alpha_a$ 
7:   if  $d > Tr$  then
8:     if  $e_\alpha > Tr_\alpha$  then                  ▷ Fordulj a cél fele
9:        $\Omega = Rotate(e_\alpha, \Omega_{max})$ 
10:       $V_x = 0$ 
11:    else                                     ▷ Haladj a cél fele és korrigáld az élfordulást
12:       $\Omega = Rotate(e_\alpha, \Omega_{max})$ 
13:       $V_x = Vx(d, V_{max})$ 
14:    end if
15:  else                                     ▷ Kitűzött pozícióba érkezett
16:     $e_\alpha = \alpha_t - \alpha_a$ 
17:    if  $e_\alpha > Tr_\alpha$  then                  ▷ Fordulj a kitűzött irányba
18:       $\Omega = Rotate(e_\alpha, \Omega_{max})$ 
19:       $V_x = 0$ 
20:    else                                     ▷ Pozíció és irány rendben
21:       $\Omega = 0$ 
22:       $V_x = 0$ 
23:    end if
24:  end if
25: end function

```

3.6. Meresek

Ebben a fejezetben tanulmányozásra kerül a robot viselkedése abban az esetekben ha valamely kerék meghibásodik, és ezáltal nem megfelelően működik. Hasonló eset történt a Marson a Spinit mars roverrel (2006, Március, 13) [15] amikor az első jobb kereke meghibásodott. A megoldás az volt hogy a robot mozgása optimálisabb lesz energia felhasználás szempontjából ha háttal megy. Az energia ellátása is véges volt, kiszolgáltatott volt a napsütésnek, a nap elemekre rakodót por miatt csökkentek a ayok hatásfoka így sokkal alaposabb mozgás pálya tervezésre volt szükség. Problémák adódtak a homokos talajjal is, a Spirit mars járonak, kerei a homokba süllyedtek és beragadtak, a földi irányító csoport egy másolat segítségével próbálta kimozdítani a csapdaból. A hasonló eseteket elkerülhetők lennének, ha ismerve a robot korlátait olyan mozgás pályát határoznak meg amellyel elkerülhetjük ezen akadajokat, vagy időben detektálhatjuk ezen problémákat pl: homokba süllyedés érzékelése.

A robottal a következő méréseket fogjuk elvégezni:

(a) BL kerék blokált

(b) BL és BR kerék blokált

(c) BL kerék maximálisan pörög

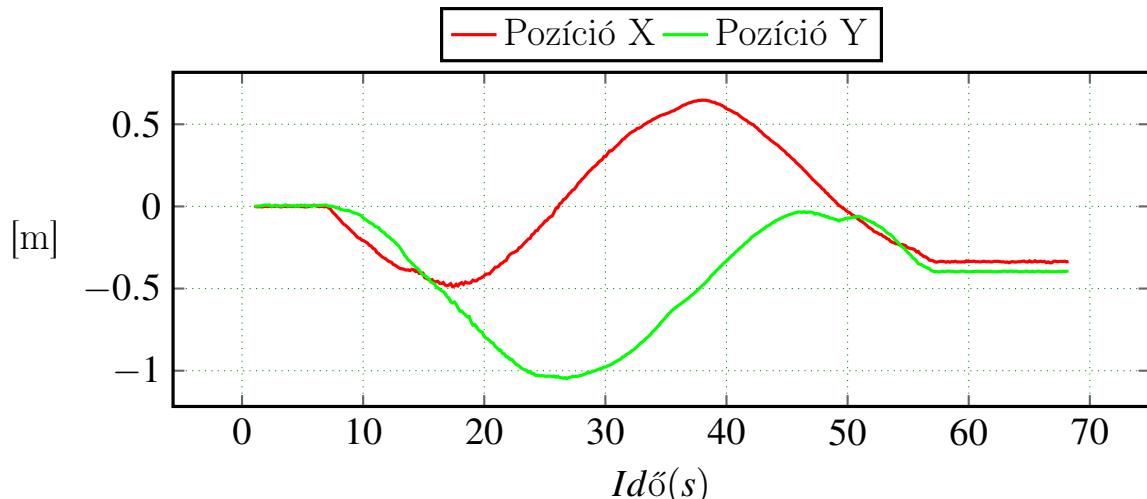
(d) BL és BR kerék maximálisan pörög

3.6.1. Differenciális Forgás Vízszintes Talajon

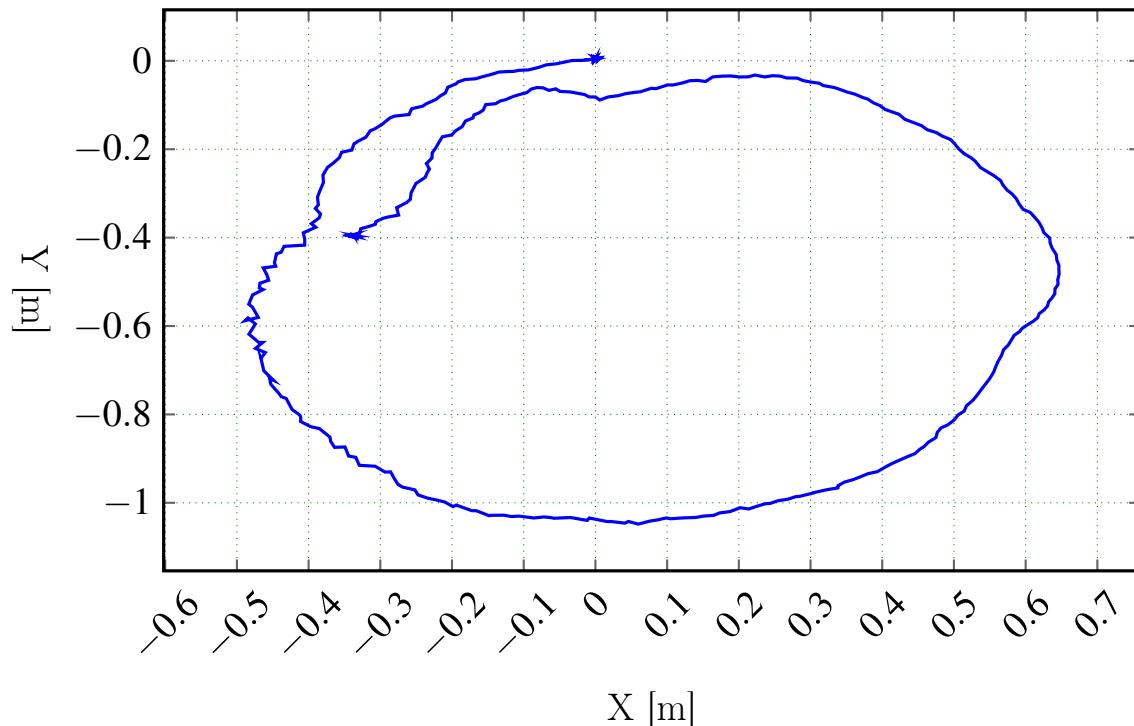
Diferenciális forgásnak nevezük azt amikor a jobb és bal oldali kerekek sebesege megegyező, csak irányukban ellenkező, így a robot belső területén belül jön létre a ICR pont a COG közelében kelene legyen ábra 2.6.13.

3.6.2. Feloldali kerekek blokolva kavicsos talajon

A robot baloldali kerekei leblokkolva és a jobboldali kerekei $50^\circ/\text{s}$ szögsebeséggel forognak. Az eredmények alapjan a ábra 3.6.16 latható a robot által leírt pálya. A mozgás során több mint 360° ot fordult és mondhatni korpalyát írt le. A talajjal való súrolások miatt a robot nem tökéletesen fordul ez latható abban is hogy a másodszori fordulás mar az előző pályatól eltérő.



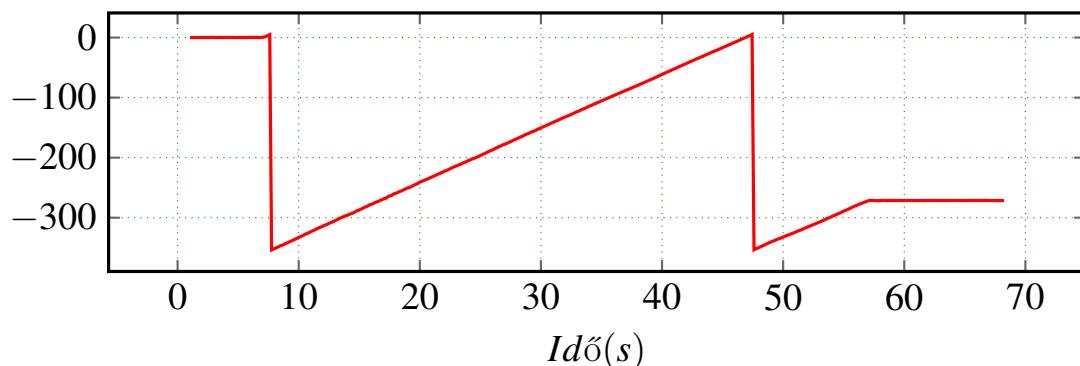
3.15. ábra. $SSMR-4W$ típusú robot pozíciója, X és Y tengelyekre bontva, kereksebessegek $BL=FL=0$ és a $FR=BR=50^\circ/\text{s}$



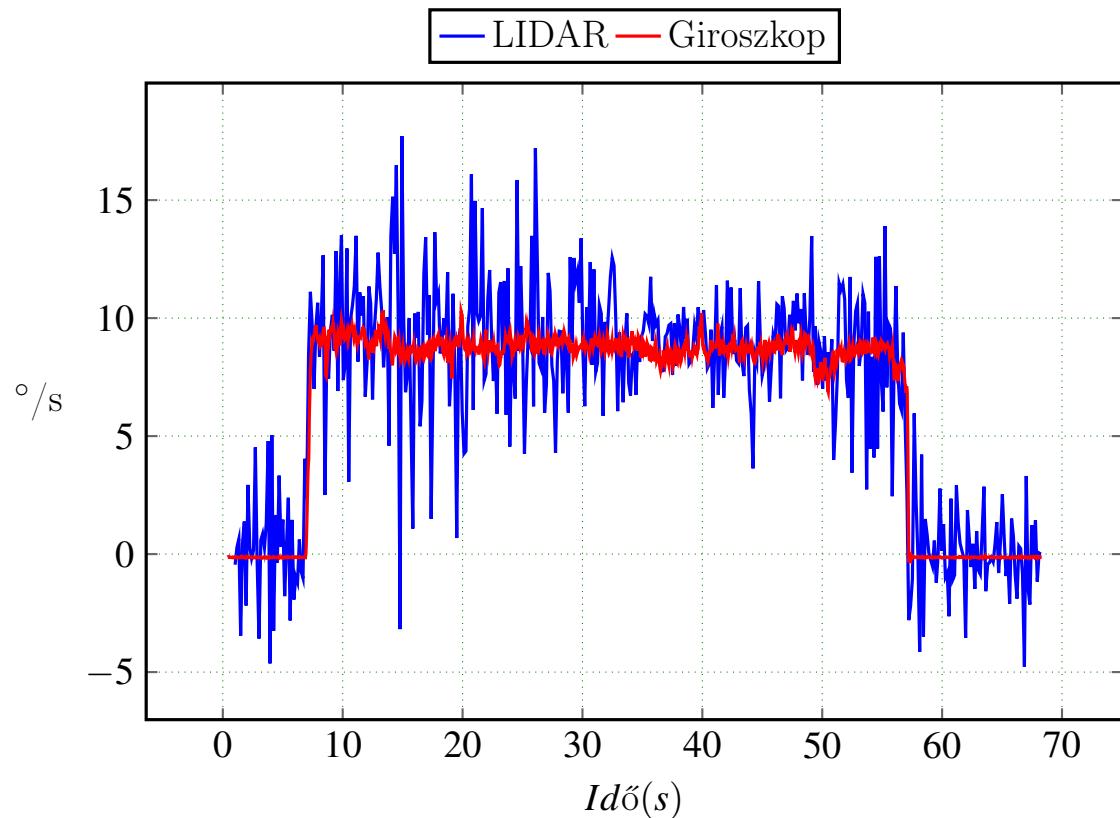
3.16. ábra. *SSMR – 4W* típusú robot altal leírt palya, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$

A meres soran a fordulási szögsebesseg $9^\circ/s$ latható a ábra 3.6.17 abran. A LIDAR es HectorMap segítségével mert abszolut szögsebesseg zajosabb mint a giroszkop altal mert. A lidar altár mert szögsebesseg elonyosebb mert a zajokat nem kell megintegralni ahoz hogy megkapjuk a szögsebesseget a gyroszkóppal elentetben.

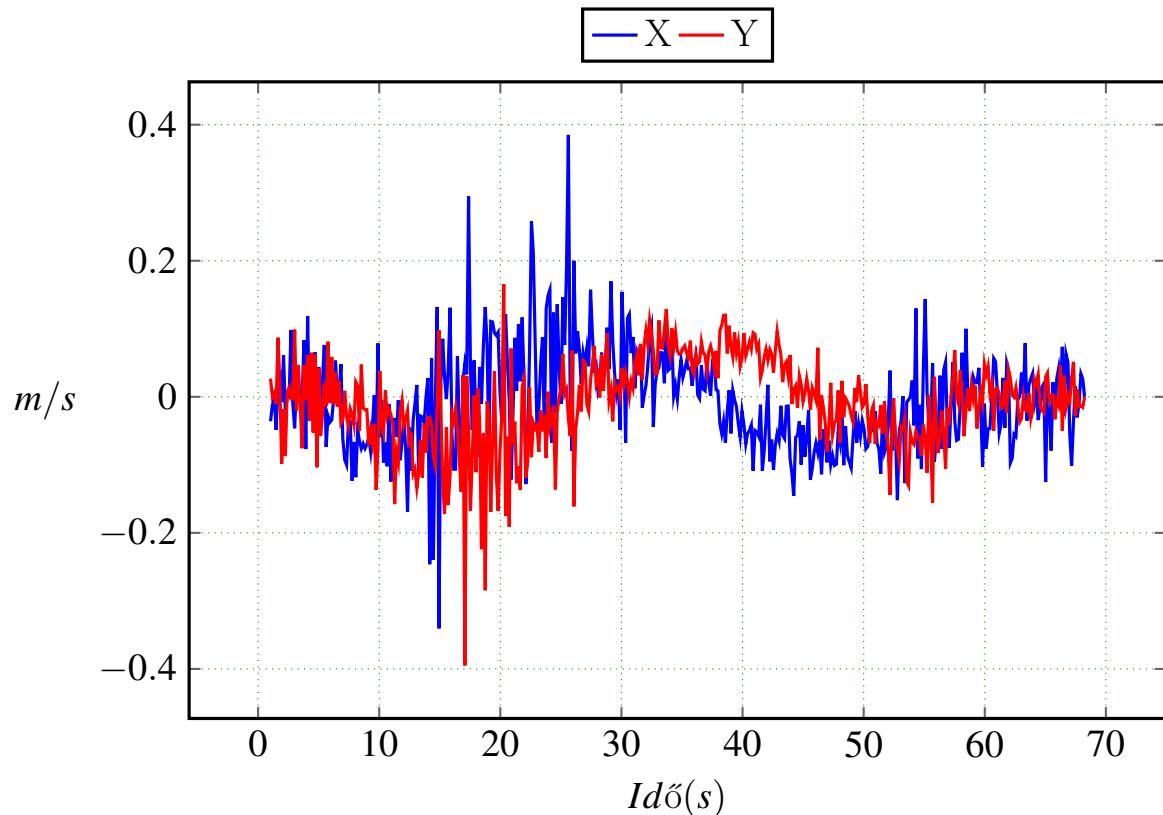
A linearis sebessegeket tekintve ábra 3.6.18 szinuszosan változnak, az X es Y kovot megfigyelhető egy 90° eltolódás. A kerületi sebesseg 0.1 m/s korúli.



3.17. ábra. *SSMR – 4W* típusú robot orientacioja, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$



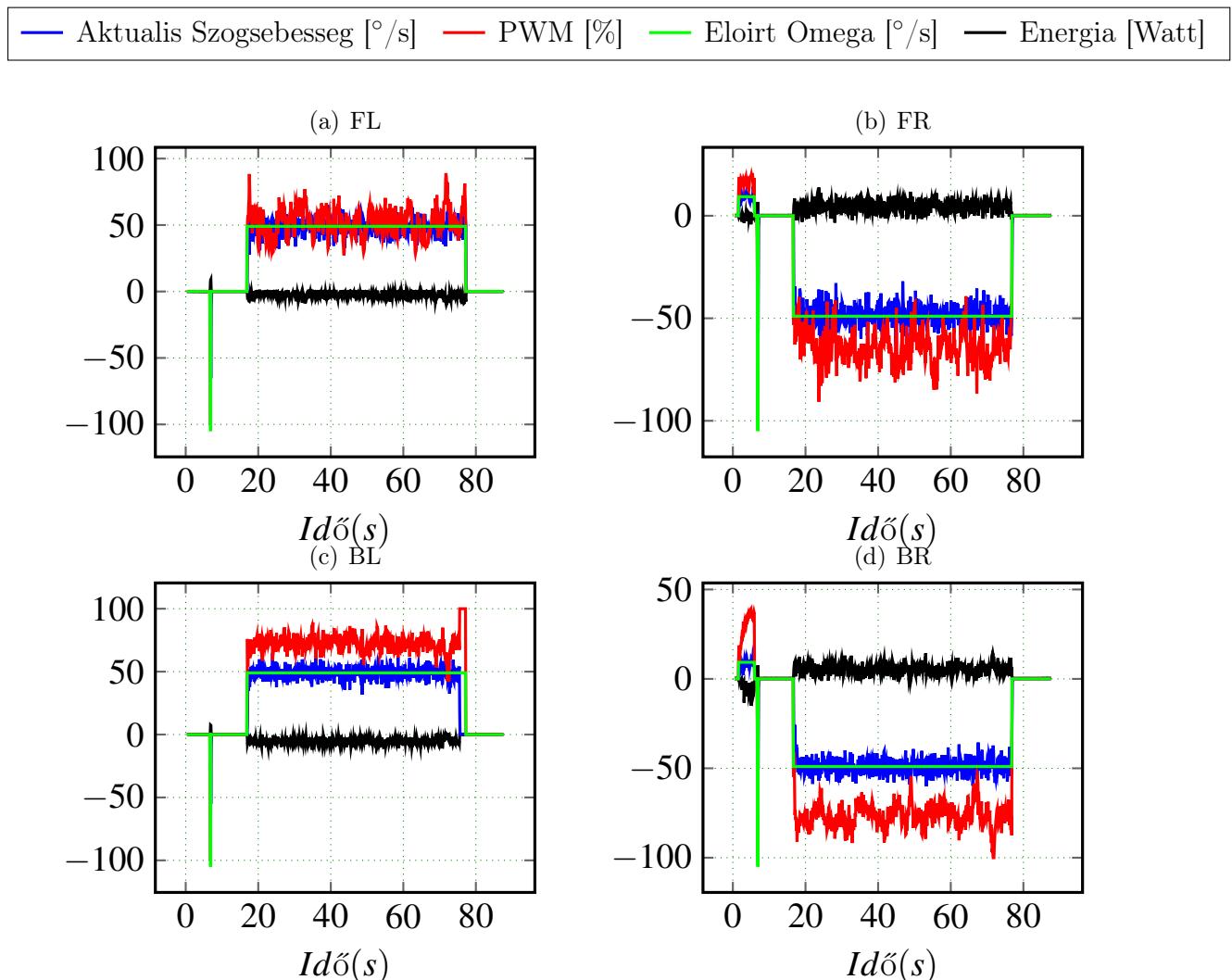
3.18. ábra. *SSMR-4W* típusú robot fordulási szögsebessege, kereksebessegek BL=FL=0 es a FR=BR= $50^{\circ}/\text{s}$



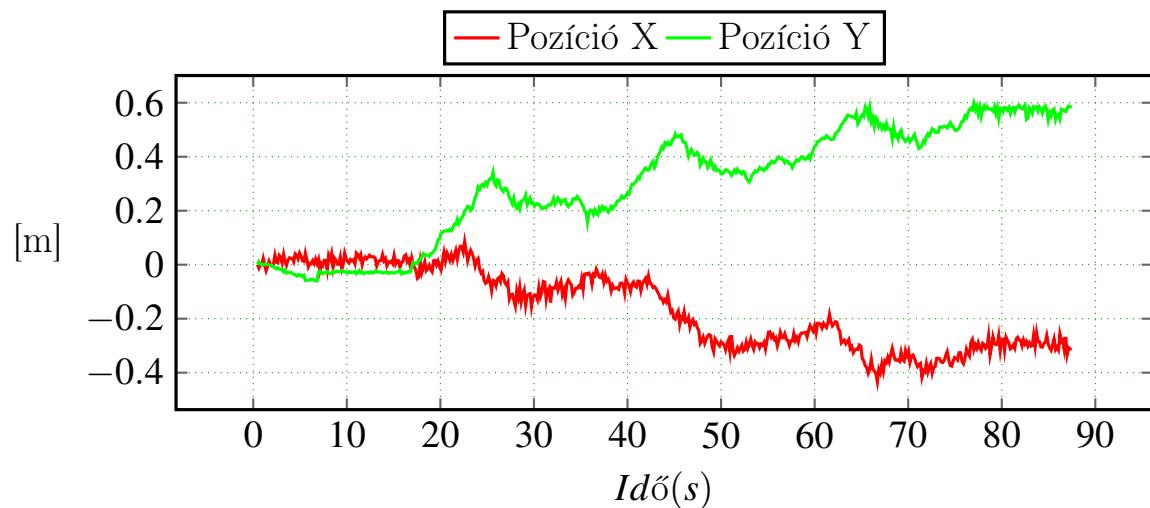
3.19. ábra. *SSMR-4W* típusú robot egyenesvonalú sebessegei, kereksebessegek BL=FL=0 es a FR=BR= $50^{\circ}/\text{s}$

3.6.3. Kavicsos talajon helyben forgás

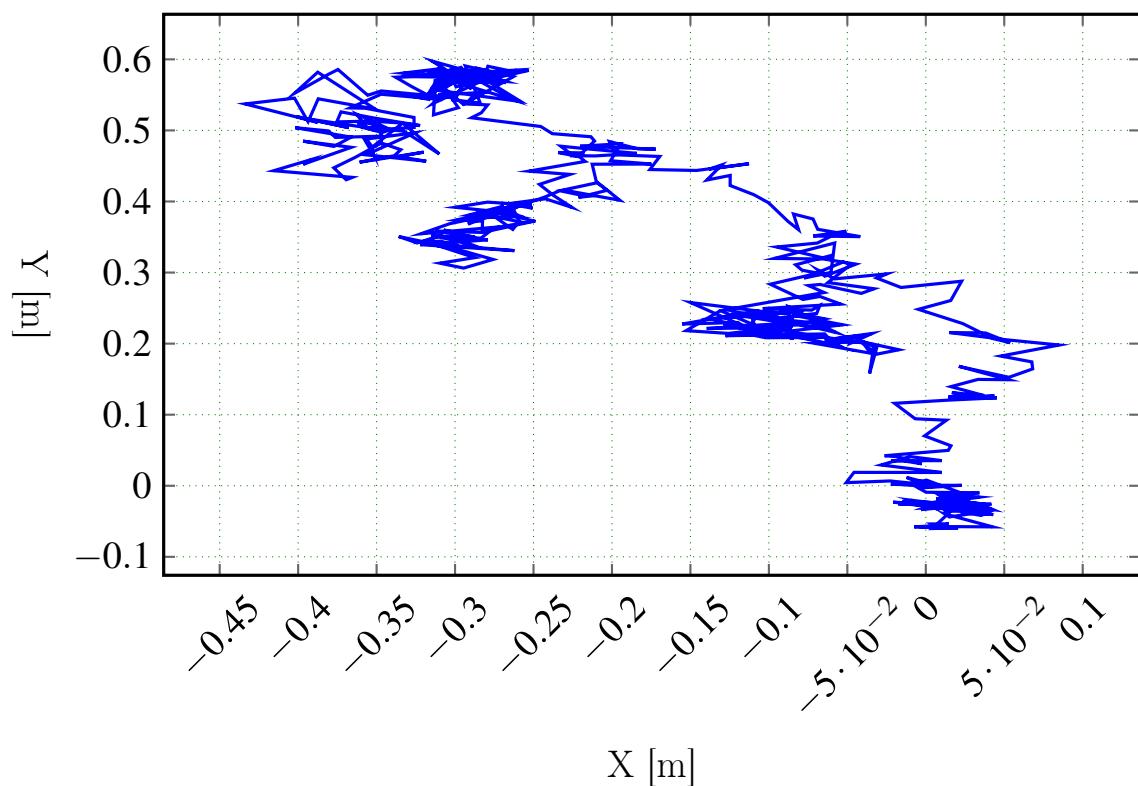
A ábra 3.6.21 megfigyelhető amint a robot kavicsos talajon differenciálisan fordul 60 másodpercen keresztül, ezalatt háromszor teljen korbefordul. A palyat tekintve letrejön egy oldalirányú mozgás is így az X tengelyen 0.22m és a Y tengelyen 0.6m mozdul el. Az oldalirányú mozgas a nem egyenlo surlodasi erok miatt jön létre. A fordulás kozben a kerekek követik az eloirt referencia szogsebesgeket, amint az ábra 3.6.20 abran is lathato. A frdulasi szogsebeseg 20 °/s, az X es Y tengelyen valo sebesseg elhanyagolható nagysagu de jelen van mivel a robot forgasi kozepontja elmozdul ábra 3.6.22.



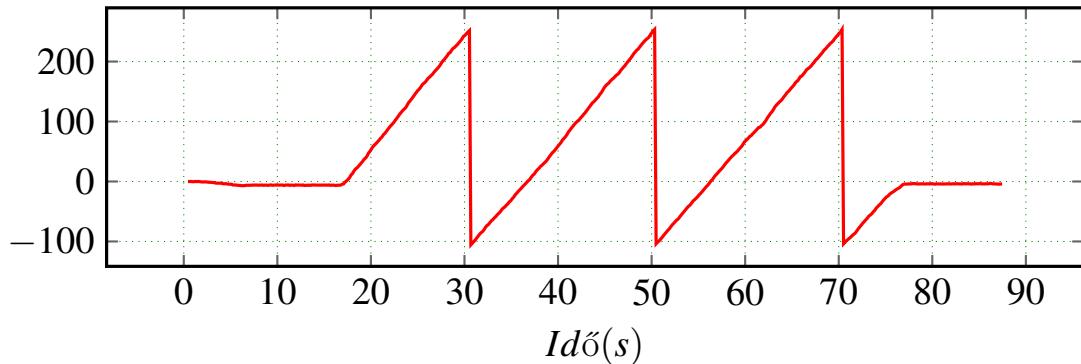
3.20. ábra. *SSMR – 4W* típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^{\circ}/s$



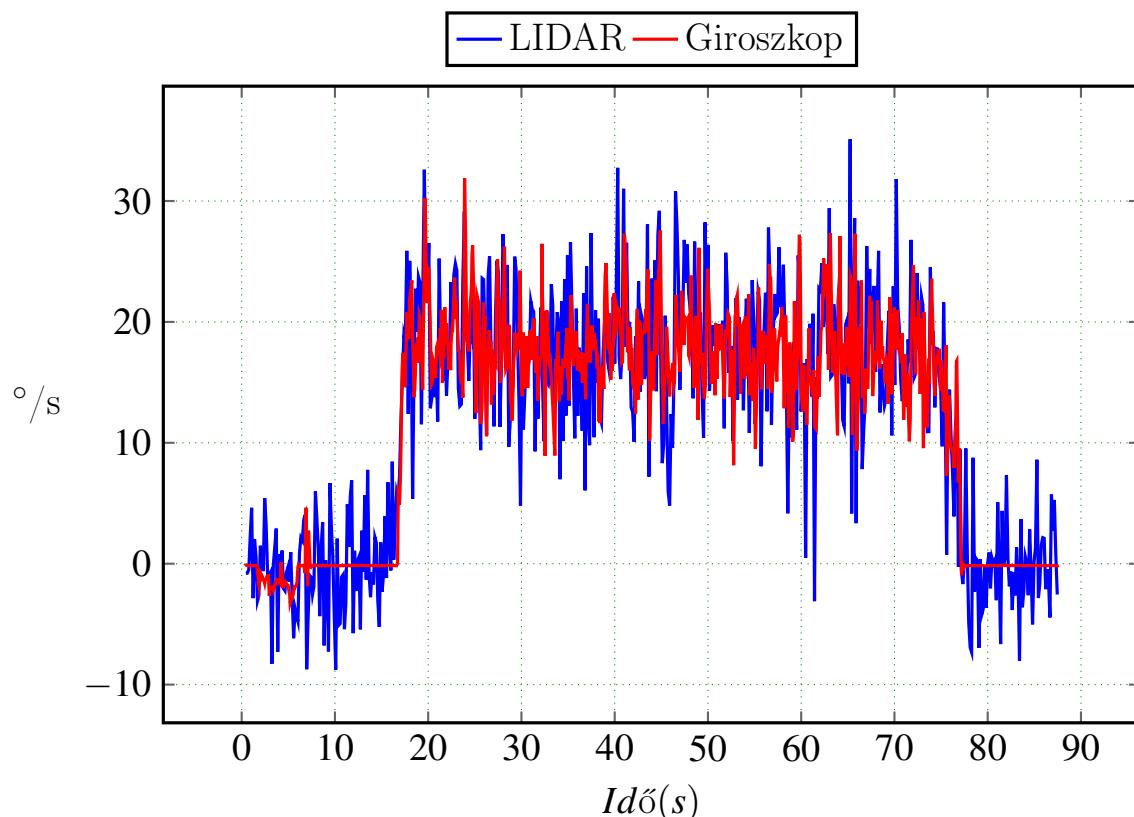
3.21. ábra. *SSMR – 4W* típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=-50$ es a $FR=BR=50^\circ/s$



3.22. ábra. *SSMR – 4W* típusú robot altal leírt palya, kereksebessegek $BL=FL=-50$ es a $FR=BR=50^\circ/s$



3.23. ábra. *SSMR – 4W* típusú robot orientacioja, kereksebessegek $BL=FL=-50$ es a $FR=BR= 50^\circ/s$

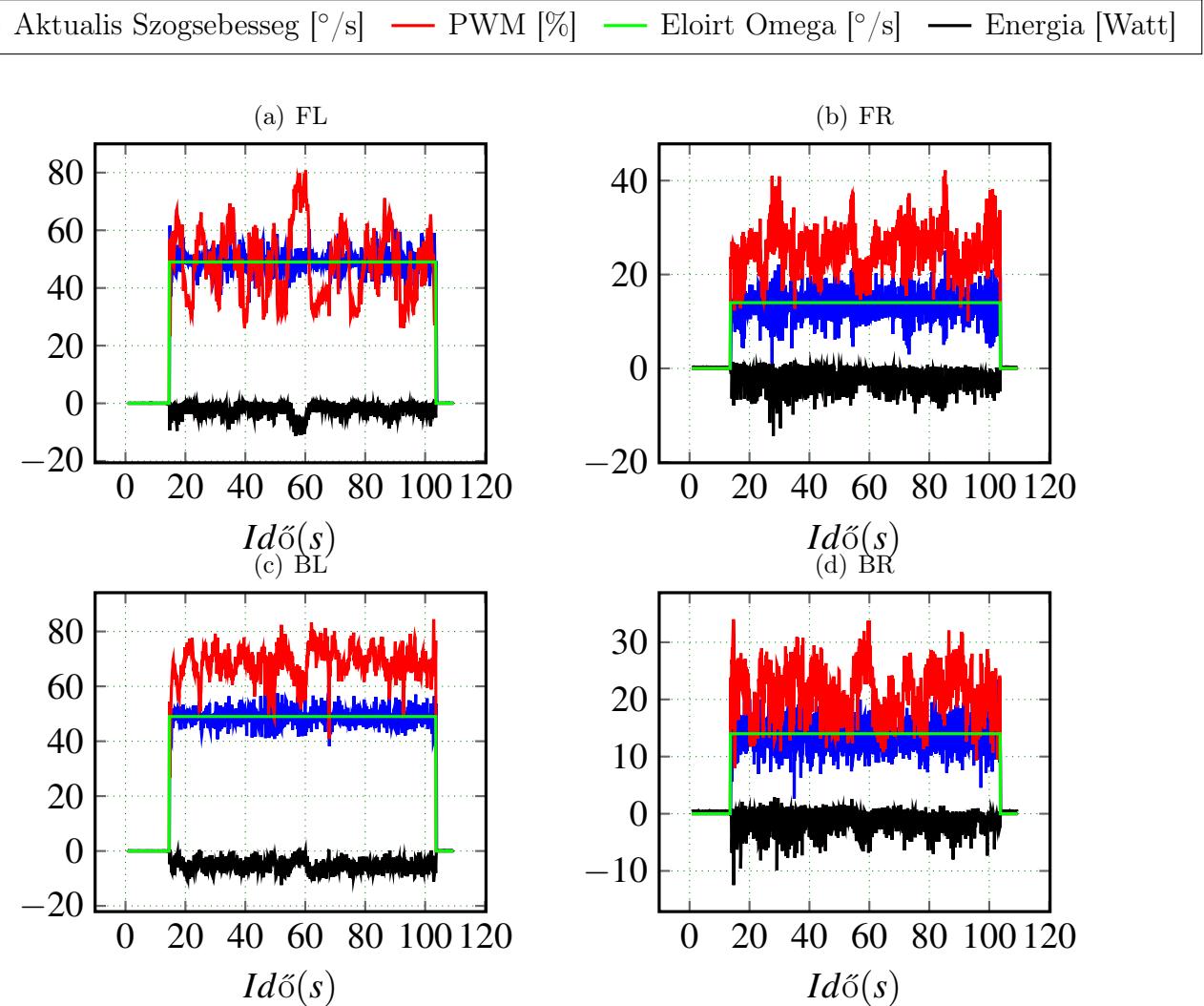


3.24. ábra. *SSMR – 4W* típusú robot fordulasi szogsebessege, kereksebessegek $BL=FL=-50$ es a $FR=BR= 50^\circ/s$

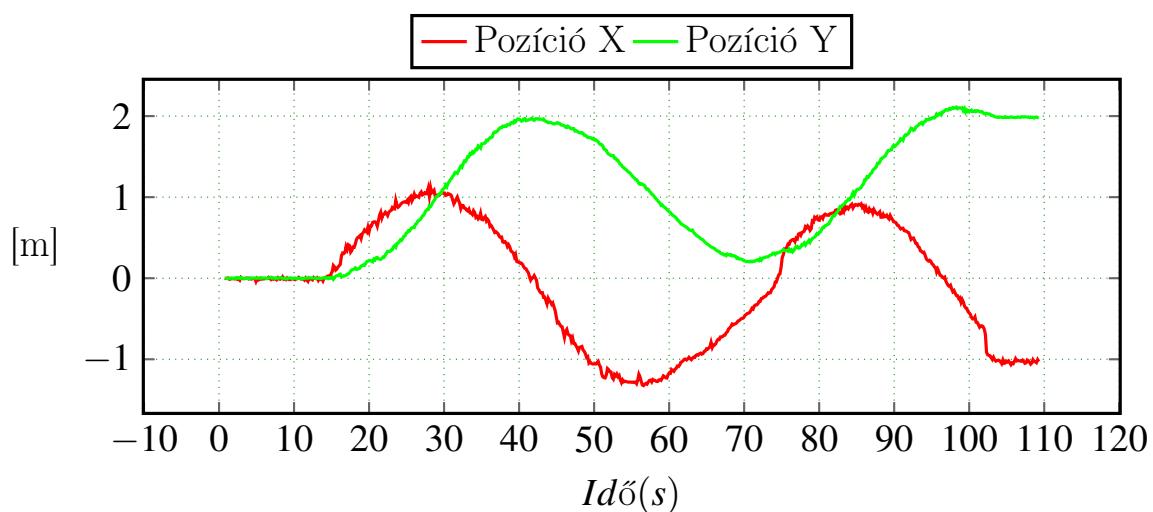
3.6.4. Kavicsos talajon korpalyan mozgas

3.6.5. Kavicsos talajon korpalyan 7 5

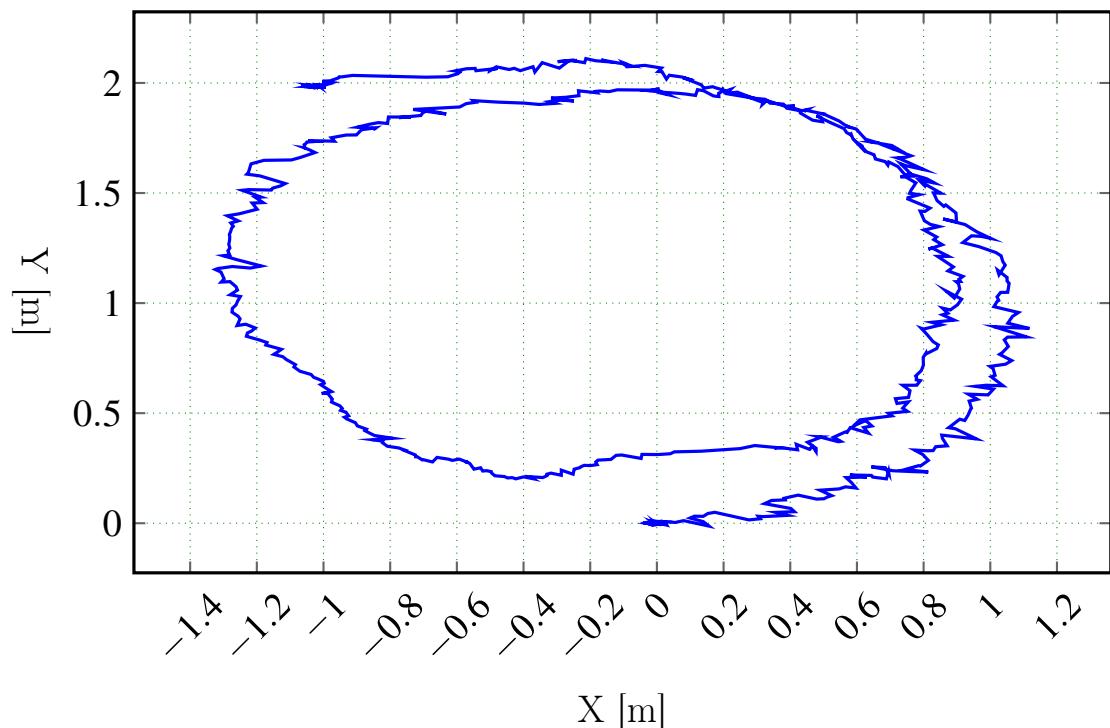
A ábra 3.6.26 megfigyelhető amint a robot kavicsos talajon differenciálisan fordul 60 másodpercen keresztül, ezalatt háromszor teljen korbefordul. A palyat tekintve letrejön egy oldaliranyú mozgás is így 0.4m kerül odaebb. Az oldaliranyú mozgás a nem egyenlo surlodások es eroeloszlasok miatt jön létre.



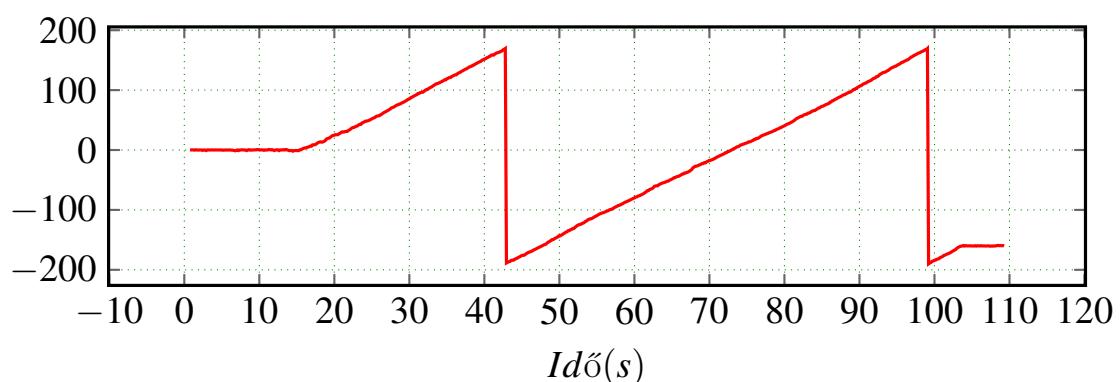
3.25. ábra. *SSMR – 4W* típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/\text{s}$



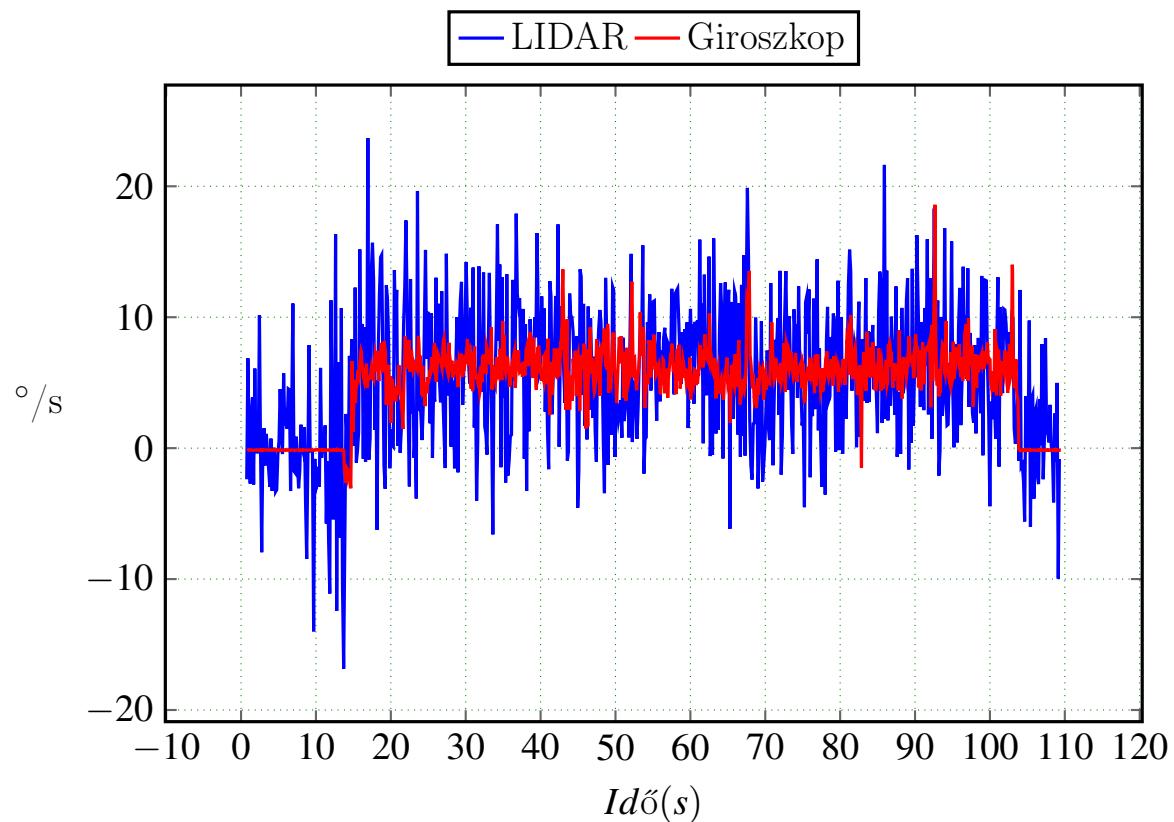
3.26. ábra. *SSMR – 4W* típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/\text{s}$



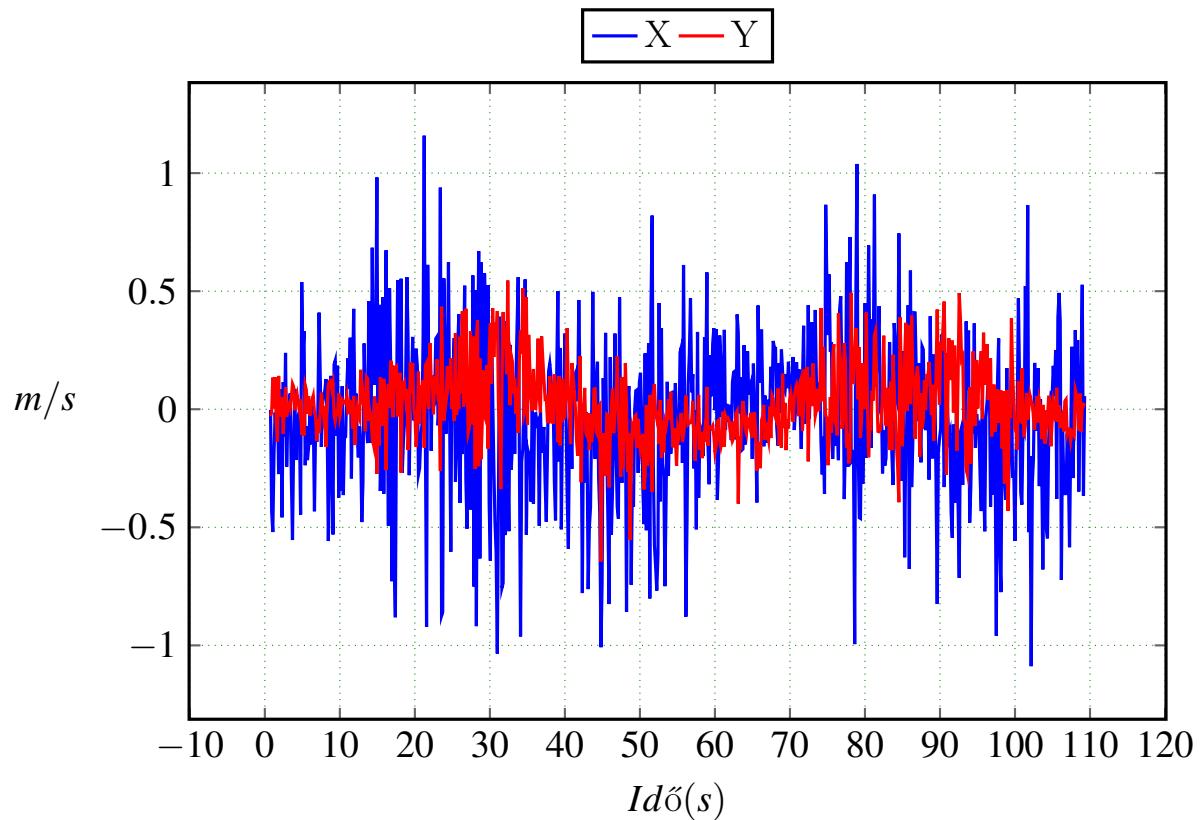
3.27. ábra. $SSMR - 4W$ típusú robot által leírt palya, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$



3.28. ábra. $SSMR - 4W$ típusú robot orientacioja, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/s$



3.29. ábra. *SSMR – 4W* típusú robot fordulási szögsebessege, kereksebessegek BL=FL=0 es a FR=BR= $50^{\circ}/s$

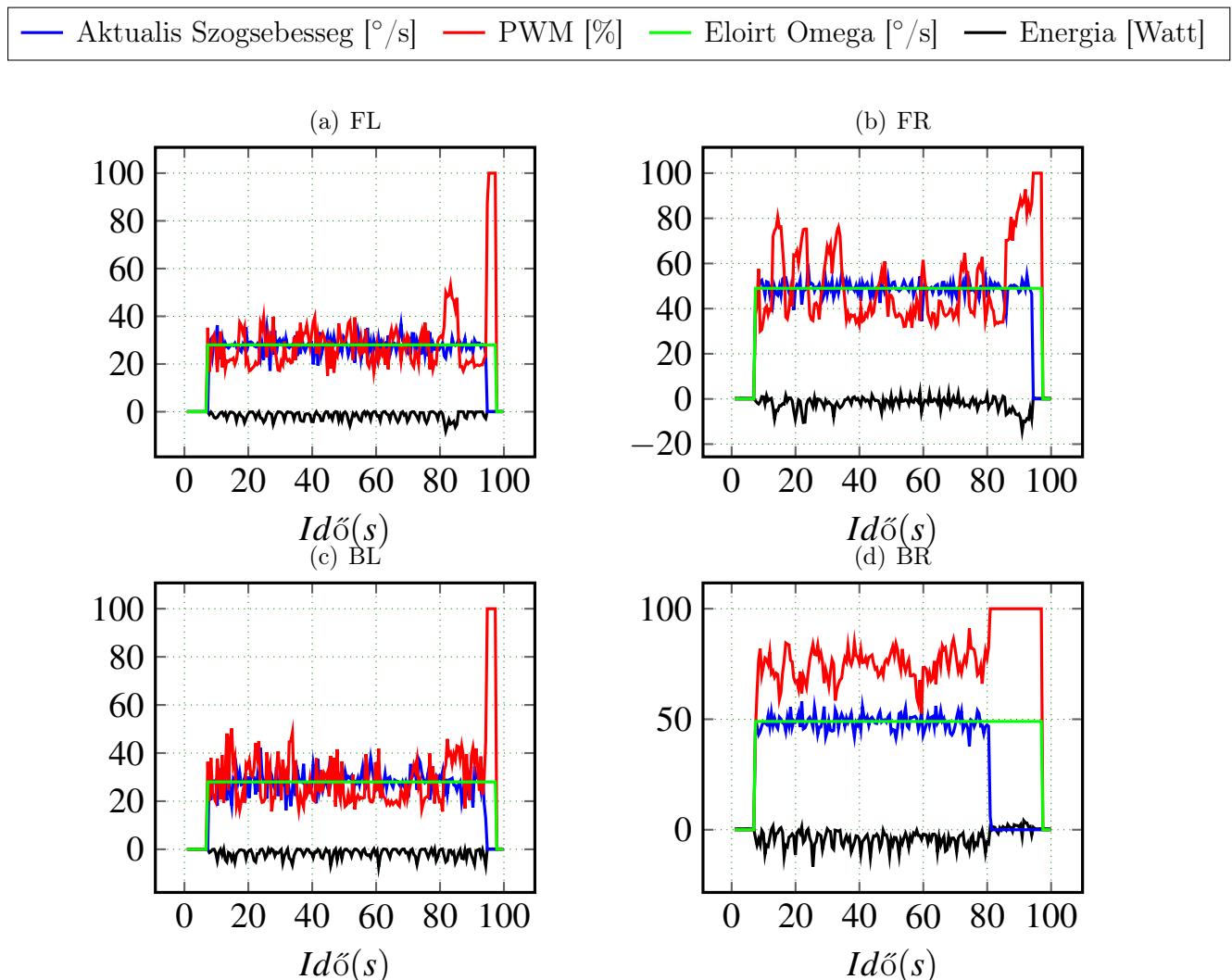


3.30. ábra. *SSMR-4W* típusú robot egyenesvonalú sebessegei, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/\text{s}$

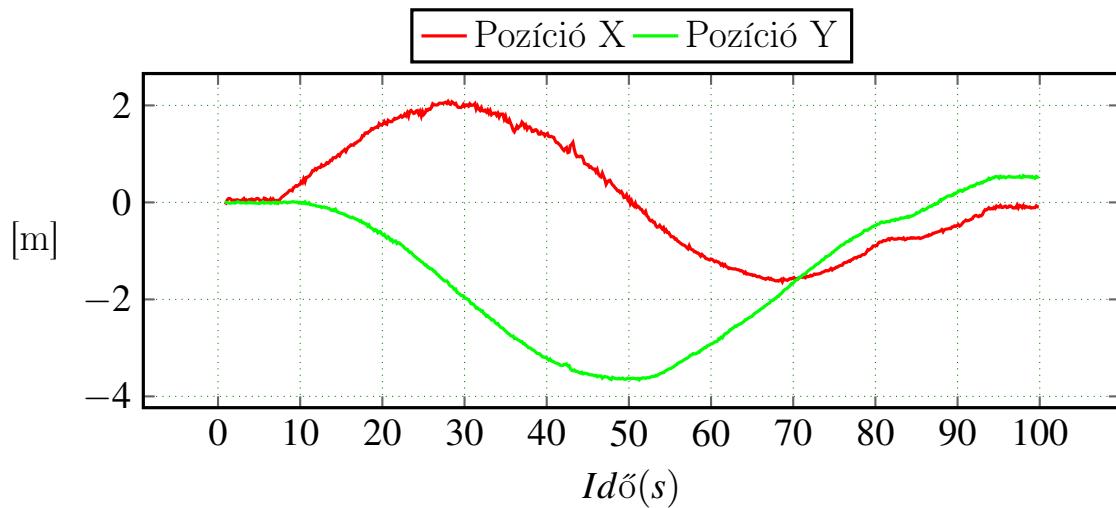
A mozgas nyilt hurokban tortenik nincsen a pozicio vagy a rogsebeseg szabalyozva. A robot ICR pontja a K_{BL} es K_{FL} pontokat oszekoto tengelyen helyezkedik el ábra 2.6.13.

;

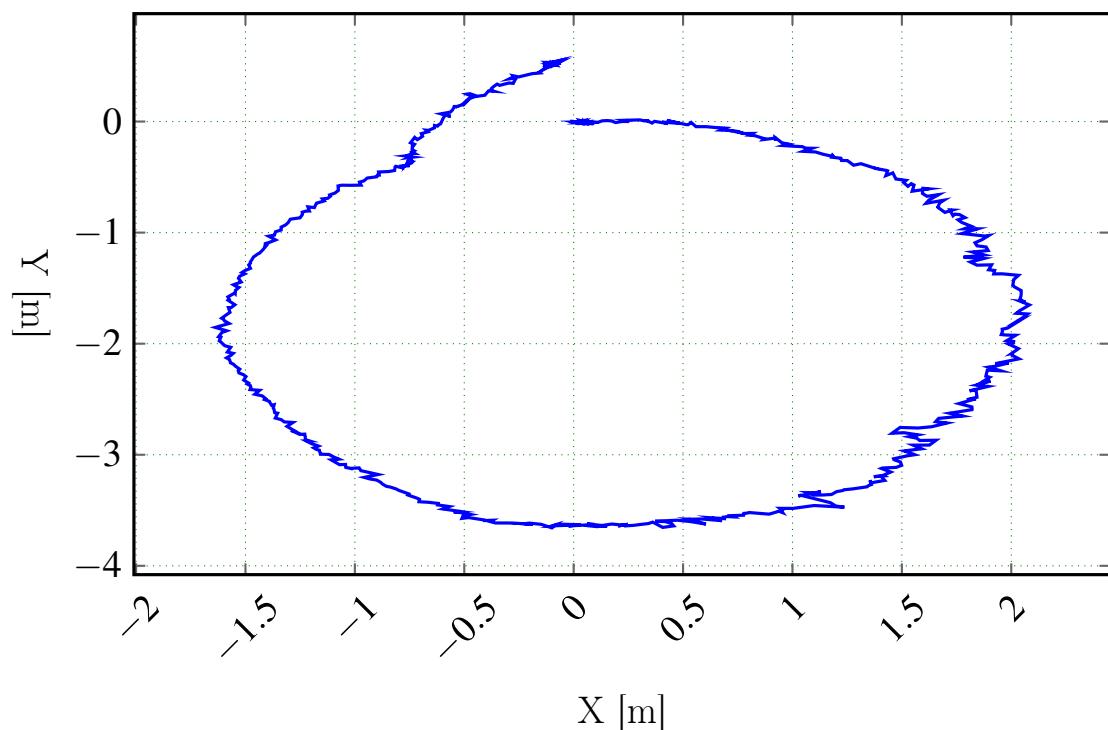
3.6.6. Korpalya 7 3 Kavicsos talajon



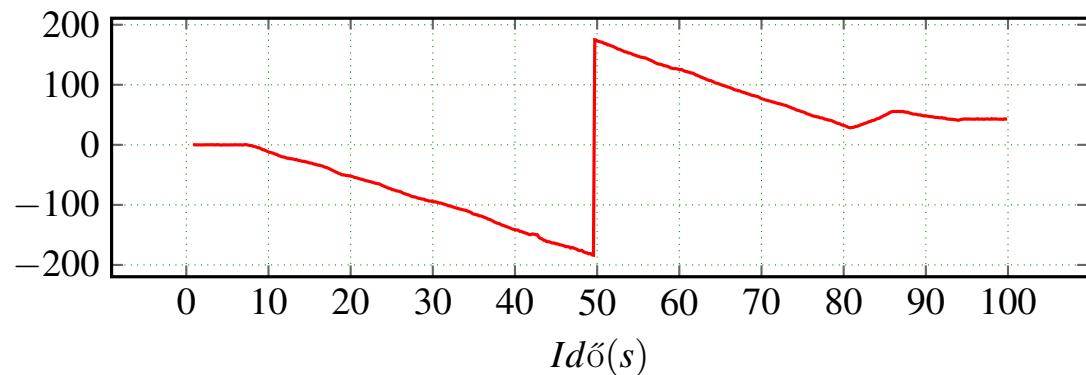
3.31. ábra. *SSMR – 4W* típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^{\circ}/s$



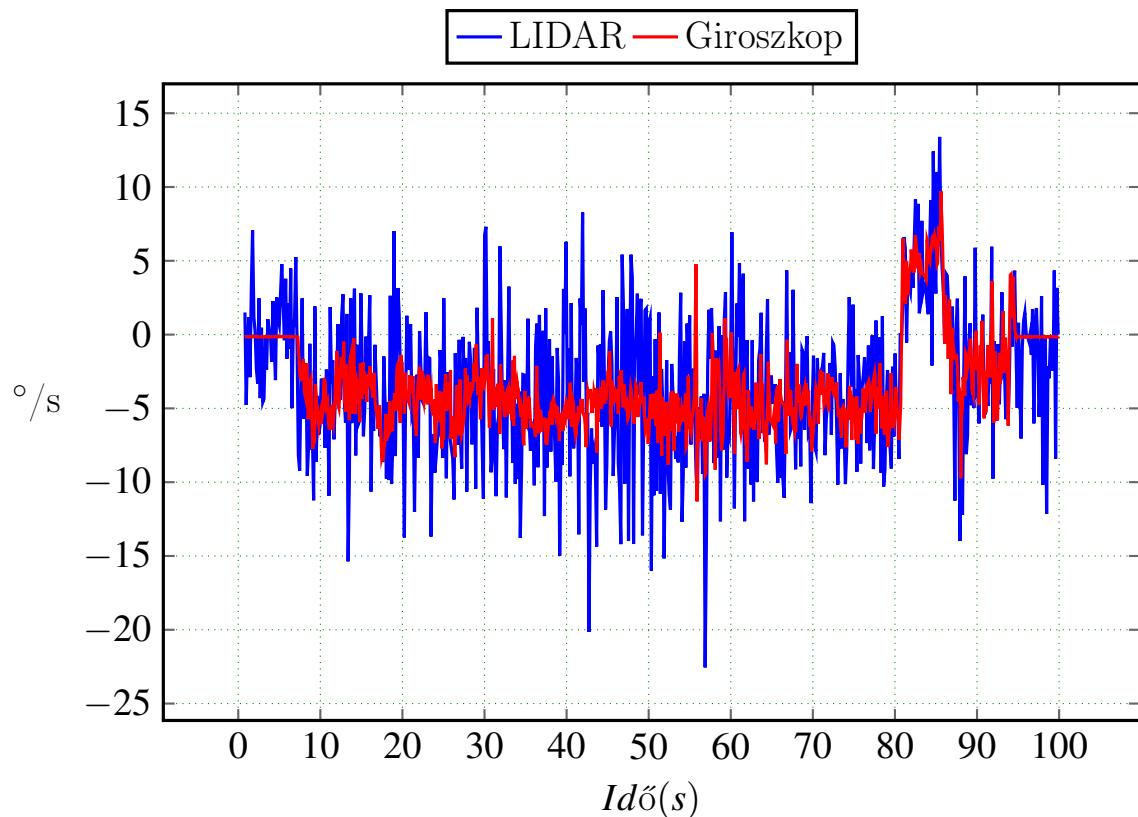
3.32. ábra. *SSMR – 4W* típusú robot mozgása, tengelyekre bontva, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/s$



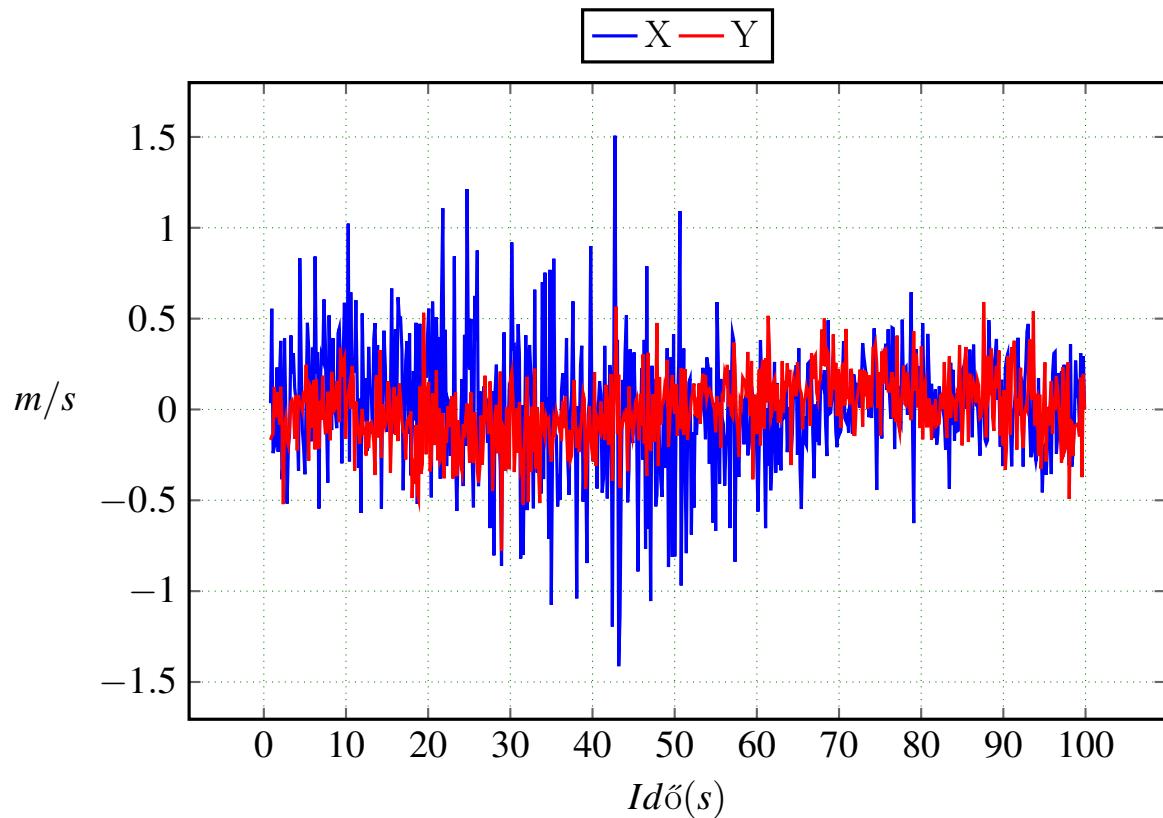
3.33. ábra. *SSMR – 4W* típusú robot aljal leírt palya, kereksebessegek $BL=FL=0$ es a $FR=BR=50^\circ/s$



3.34. ábra. *SSMR – 4W* típusú robot orientacioja, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/\text{s}$



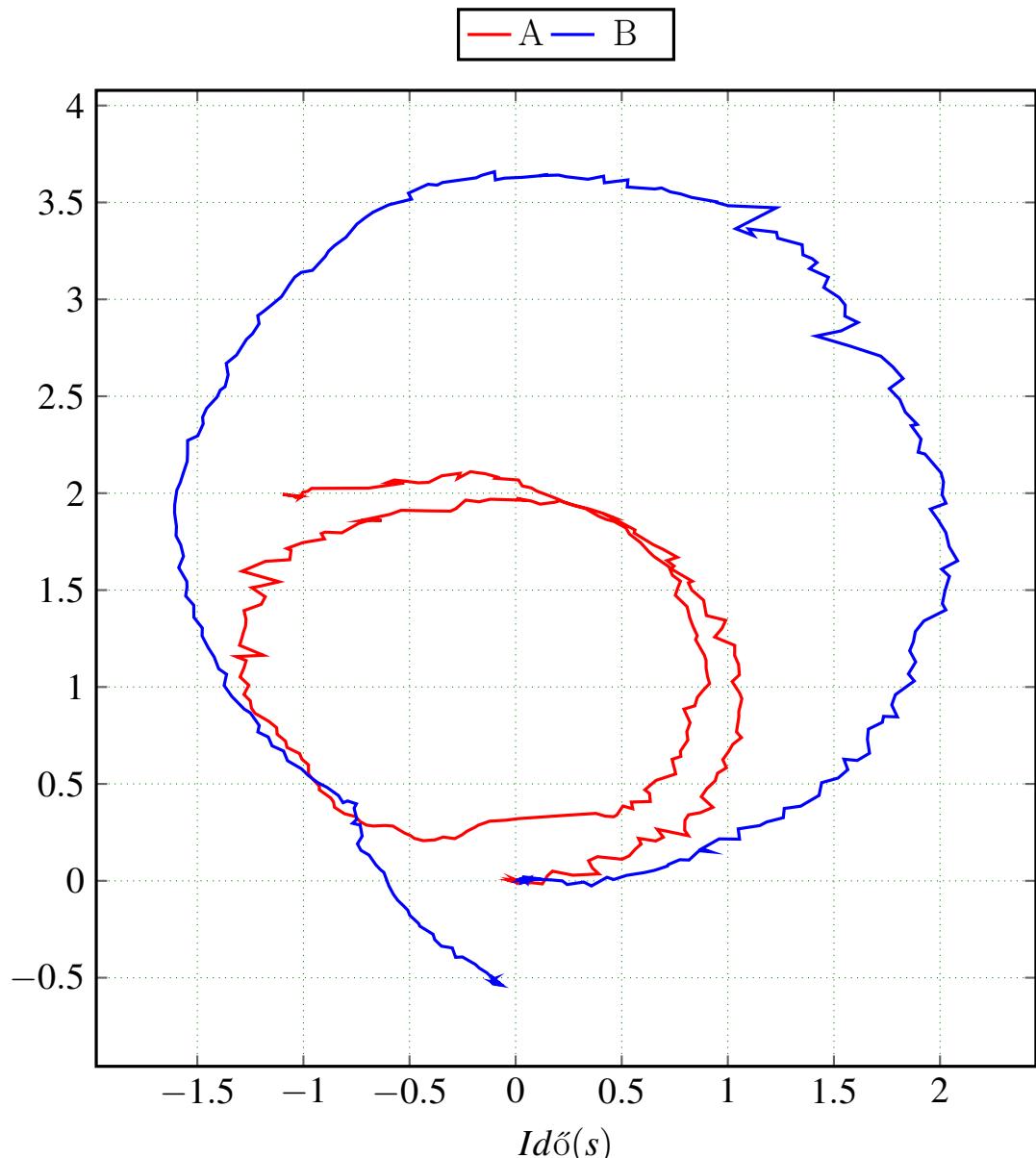
3.35. ábra. *SSMR – 4W* típusú robot fordulasi szogsebessege, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/\text{s}$



3.36. ábra. *SSMR-4W* típusú robot egyenesvonalú sebessegei, kereksebessegek $BL=FL=0$ es a $FR=BR= 50^\circ/\text{s}$

A mozgas nyilt hurokban tortenik nincsen a pozicio vagy a rogsabalyozva. A robot ICR pontja a K_{BL} es K_{FL} pontokat oszekoto tengelyen helyezkedik el ábra 2.6.13.

;



3.37. ábra. Kulombozo korpalyak

3.6.7. Homokos Lejtodfdfsdfs

A lejto meredeksege 45° , a szerkezete homokos, nagyobb meretu szilard darakobbal mae lyek elerik a 4cm atmerot is. A lejtot az FL es FR kerekekkel kozelitjuk meg, így a BL es BR kerekekre nagyobb nyomoero jut. A ábra 3.6.45 lathato a PWM beavatkozo jelek a BL es BR kerekeken 20% nagyobb peavatkozo jelet szuksegges ugyanannak a referencia jelnek a kovetesere.

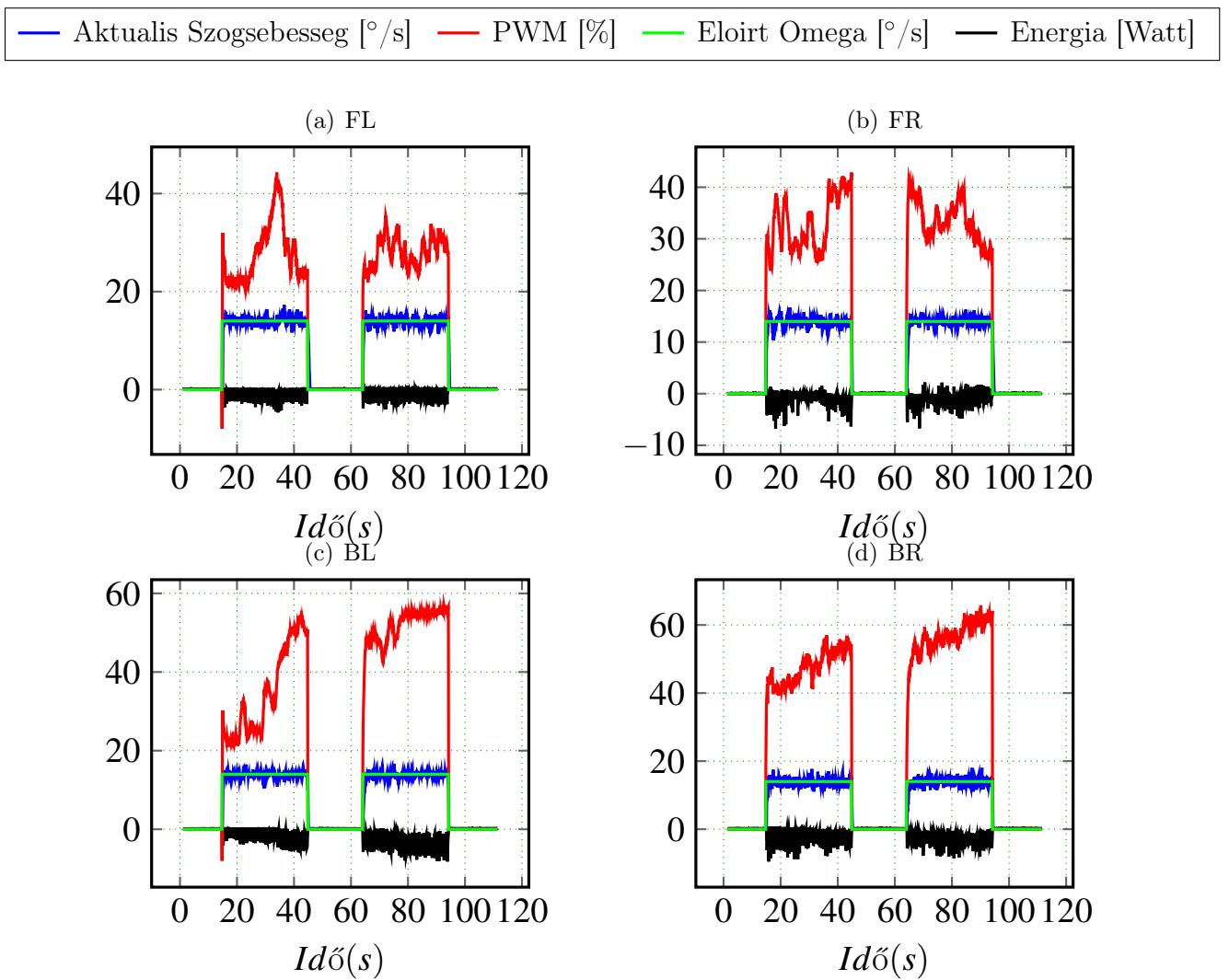
A kereke eloirt forasi sebessege $15^\circ/\text{s}$, torekedtunk hogy a kereke neassak be magukat a homokba de 0.5m ábra 3.6.39 megtetele utan a BL es a BR kereke 10 cm melyen a homokba sulyetek es a robot elakadt ábra 3.6.38. Egy masik eszrevetel hogy a FL es FR kereke egyaltalan nem astak be a talajba magukat forgas kozben ami a kisebb meroleges nyomoeronek tulajdonithato.



3.38. ábra. Homokos domb 1

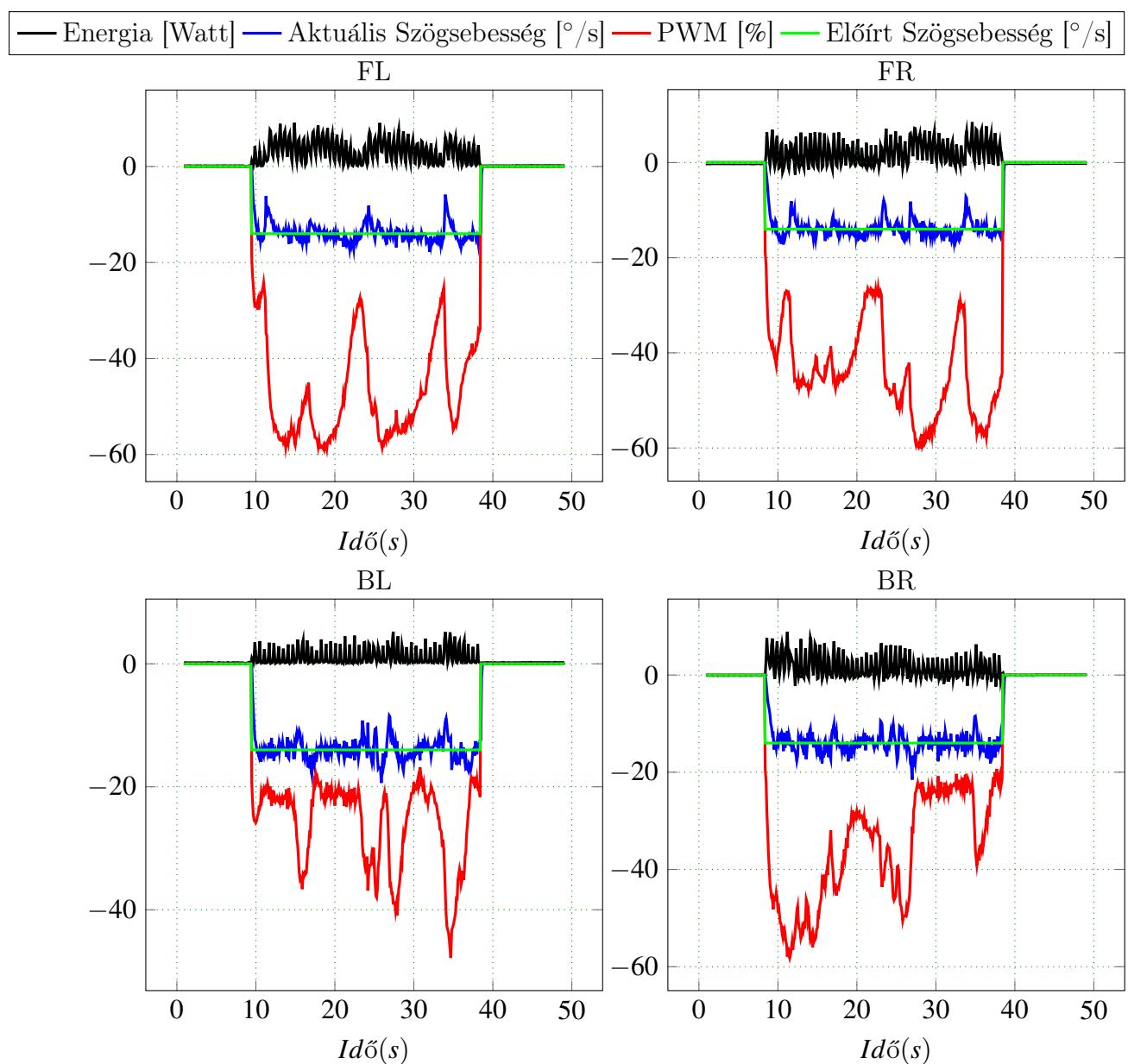


3.39. ábra. Homokos domb 1

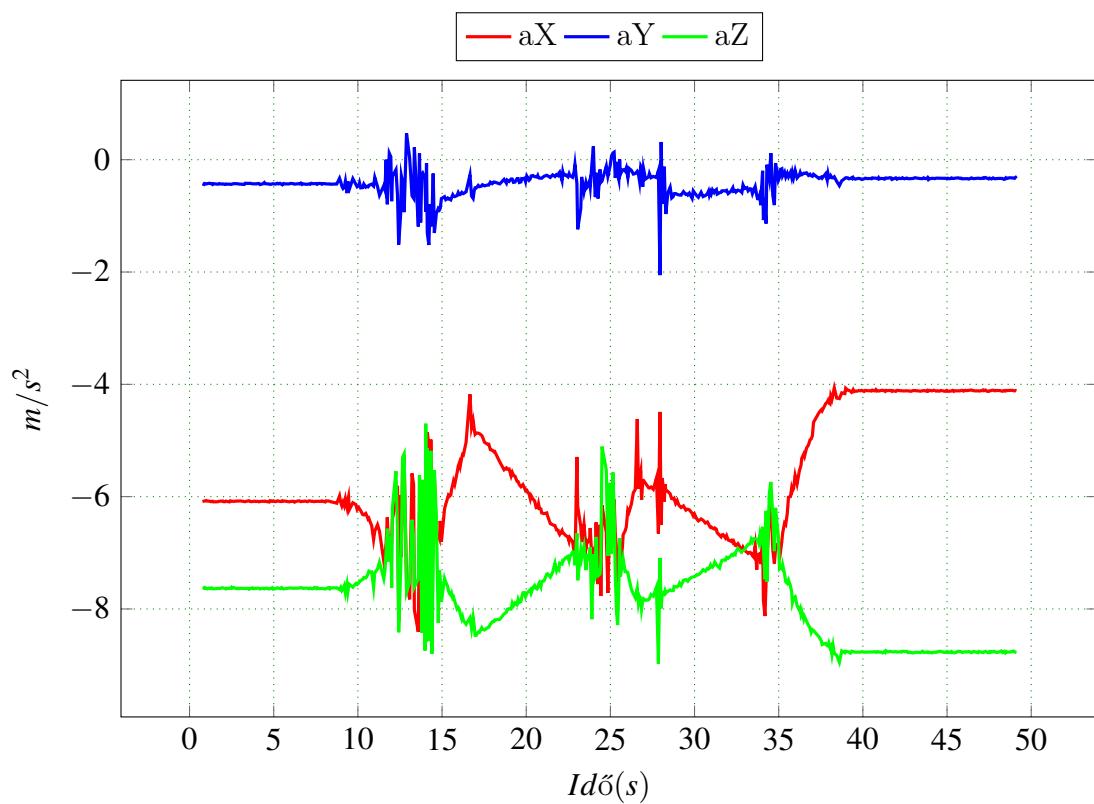


3.40. ábra

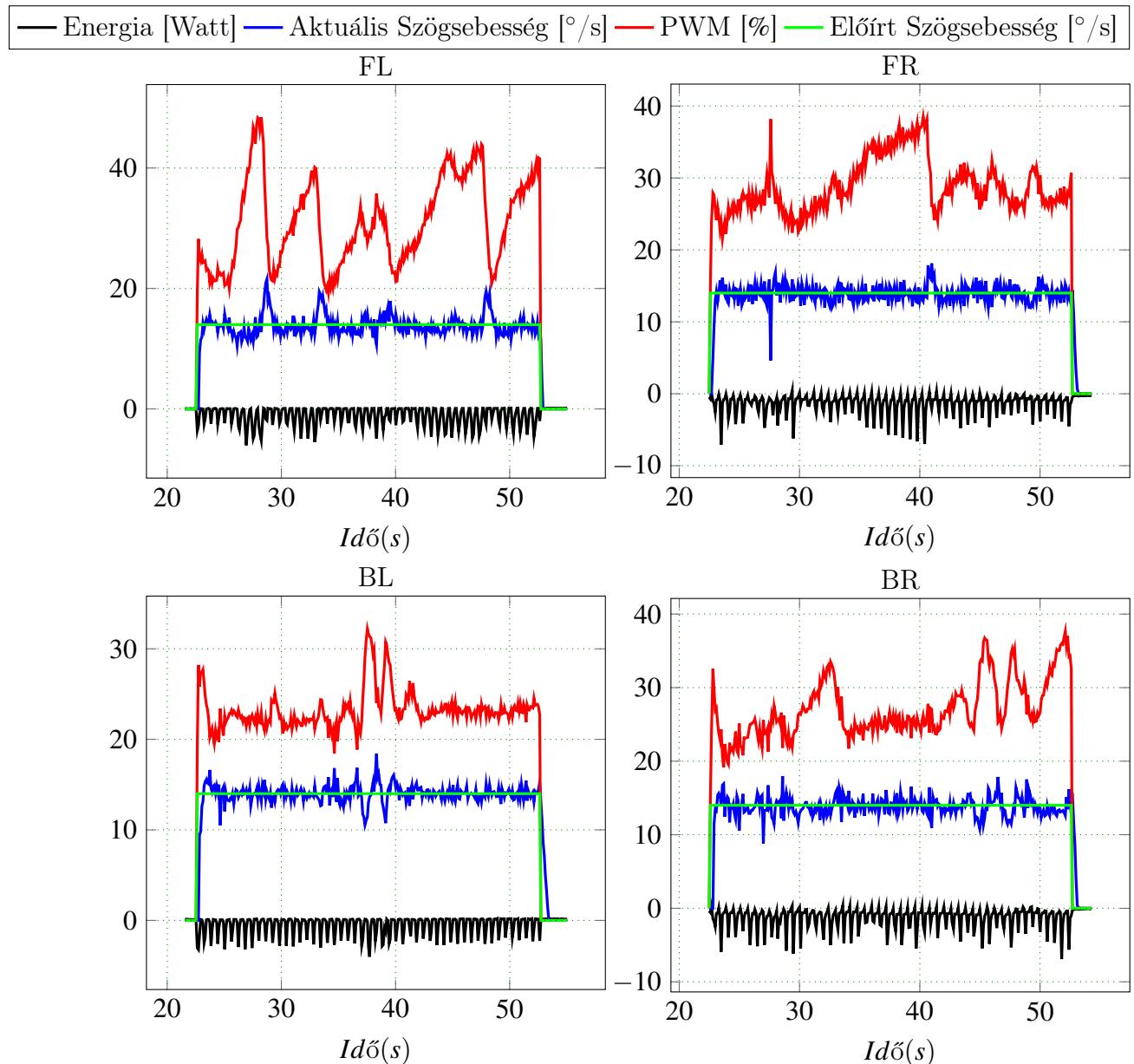
3.6.8. Lepcsón



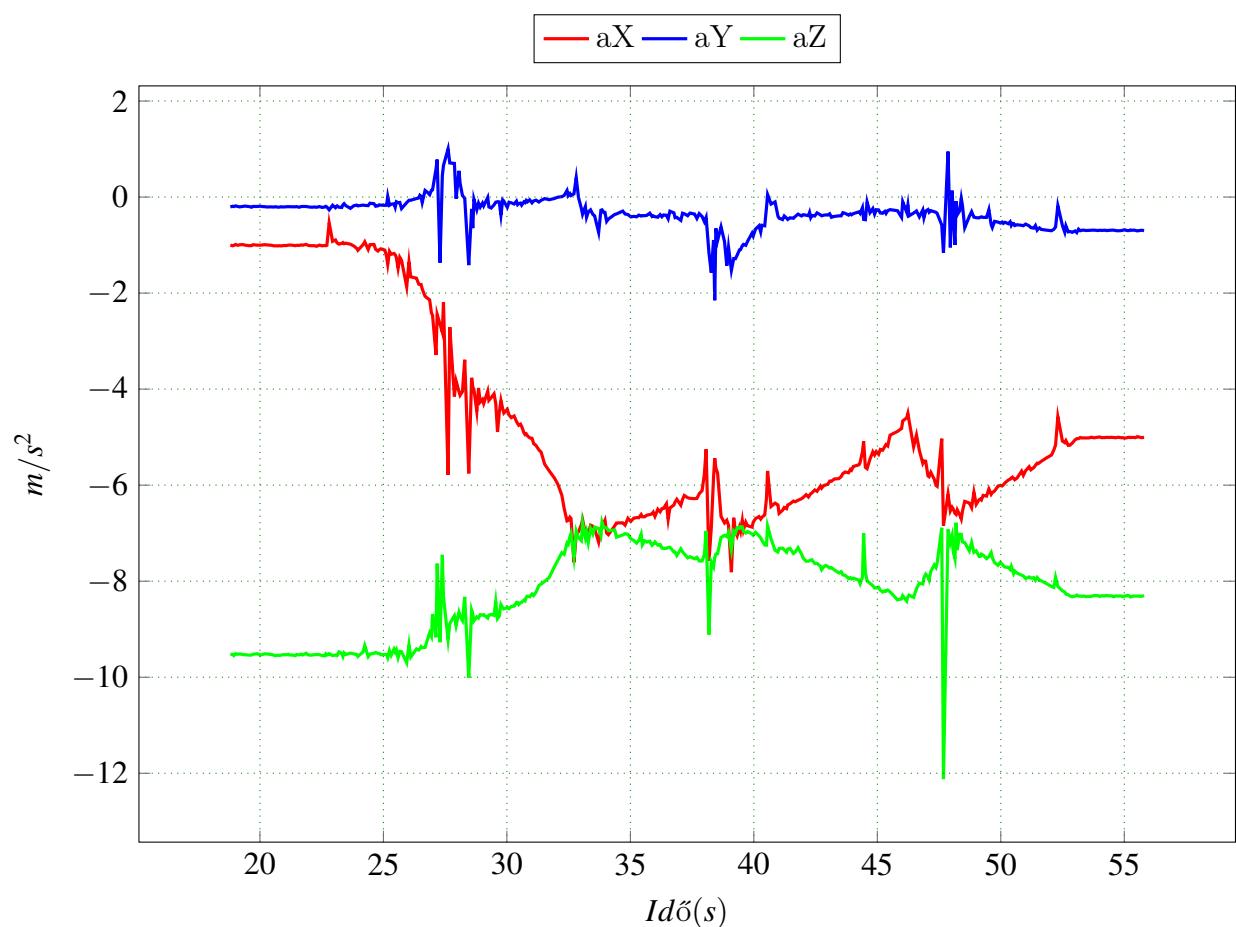
3.41. ábra



3.42. ábra

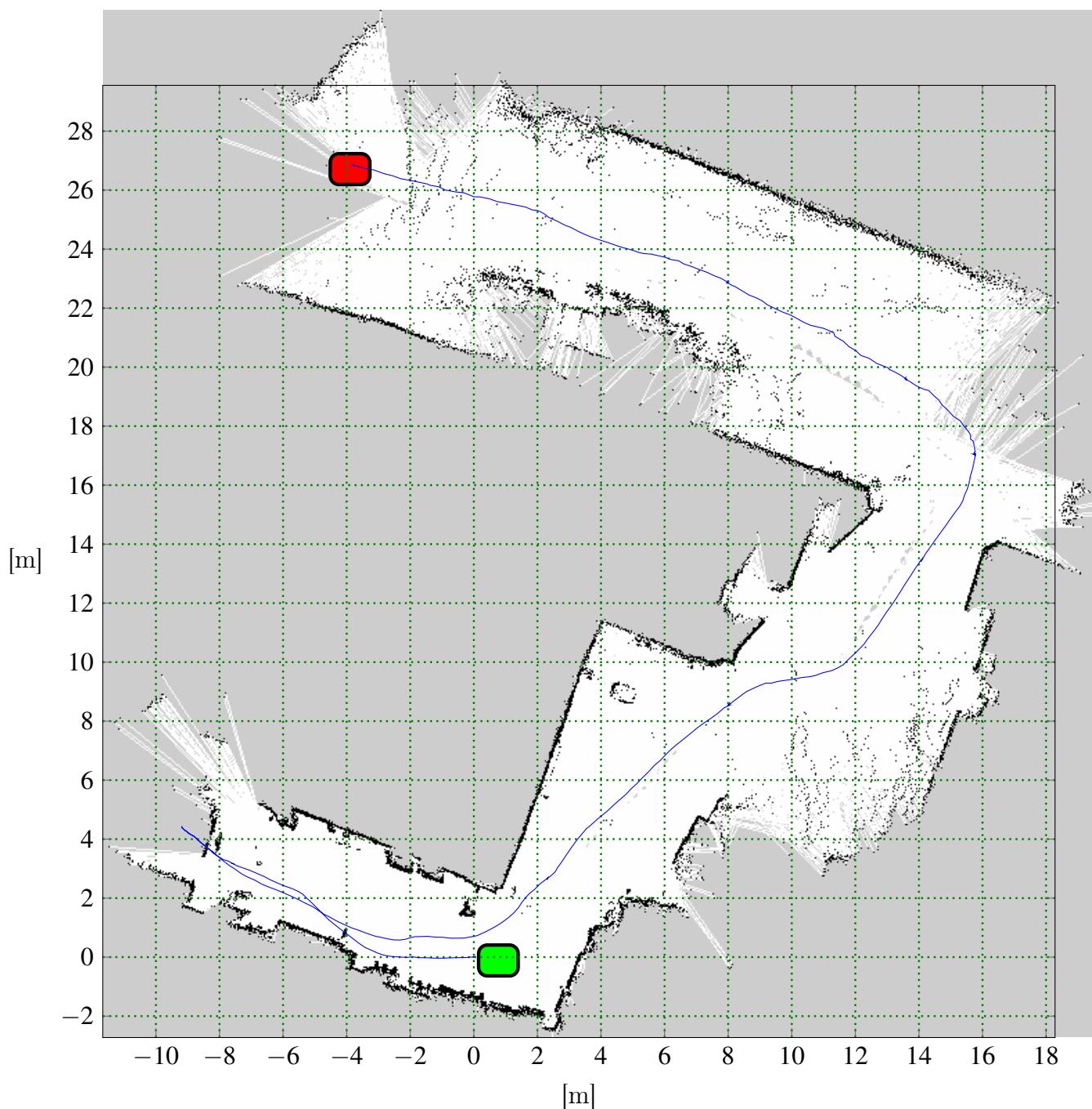


3.43. ábra



3.44. ábra

3.6.9. Ismeretlen terep terkepezese es robot lokalzalasa (SLAM)



3.45. ábra. Terkep készítése miközben tavidanyitassal halad a robot.

fejezet 4

Eredmények Kiértékelése

4.1. Megvalósítások

Ezzel a dolgozattal négy év folyamatos munkájának egy szakaszát szeretném lezárni, a következő eredményeket sikerült felmutatni, kronológiai sorrendben: A mechanikai szerkezet fejlesztése, a 2011 be fejlesztett verzióhoz képest amely sokkal robusztusabb, kültére alkalmasabb mint az előző, és egyszerűbb is. Ugyan megmaradt az a tendencia hogy csiga-áttételeket alkalmazzak a kerekek meghajtására. Az áttételeket magam terveztem és gyártattam le, a kezdeti alacsony költségvetés miatt az 3D nyomtatával elkészített inkrementális szenzort nem lehetett alkalmazni robusztusan, ami nem az jeleni hogy nem is lehetséges csak túl sok időt igényelt volna, a csiga tengely konyogása miatt. Az alkatrészeket 3D tervező programban elkészítettem, és 3D nyomtatával elkészítettem, a tapasztalatom ezekkel az alkatrészekkel: nagy mechanikai terhelés elviselésére nem alkalmasak hosszútávon, ezért történt meg hogy a csiga tengely csápagyháza terhelés alatt széttört. Vivado környezetet használtam az FPGA fejlesztésére, megvalósítottam egy uB-laze processzorrendszer kialakítását és több hardveres modult is amelyek a következők: PWM modul, UART protokoll csomag értelmező amely támogatja a nagy sebességű kommunikációt, globális engedélyező jel, ezeket a modulokat System Generator-ban valósítottam meg és IP mag készült ezekből. Sikeresen elsajátítottam a ROS alapjait, és megterveztem egy sajátos kommunikációt FPGA alapú rendszer és a ROS között. Az integráció a robot és a ROS között jól működik, minden egyes szenzor mért adata bekerül a ROS környezetbe.

A hectormap segítségével az ismeretlen terep terkepezése és a roboto lokalizálása sikeresen bekonfiguráltam a movebase nevu eszközt amely segítségével a roboto egy adott pozicioba és irányba tudjuk elvinni. A move base megoldja az akadalyok kikerülését is.

4.2. Hasonló rendszerekkel való összehasonlítás

4.3. További fejlesztési irányok

Irodalomjegyzék

- [1] David Couceiro Micael Rocha Rui Araújo, André Portugal. Integrating arduino-based educational mobile robots in ros, 2013.
- [2] S. Arslan and H. Temeltaş. Robust motion control of a four wheel drive skid-steered mobile robot. In 2011 7th International Conference on Electrical and Electronics Engineering (ELECO), pages II–415–II–419, Dec 2011.
- [3] Marissa G. Campa, J.L. Gordillo, and Rogelio Soto. Speed and point-to-point control for trajectory tracking of a skid-steered mobile robot. In 11th IEEE International Conference on Control & Automation (ICCA). IEEE, jun 2014.
- [4] Sachin Chitta, Eitan Marder-Eppstein, Wim Meeussen, Vijay Pradeep, Adolfo Rodríguez Tsouroukdissian, Jonathan Bohren, David Coleman, Bence Magyar, Genaro Raiola, Mathias Lüdtke, and Enrique Fernández Perdomo. ros_control: A generic and simple control framework for ros. The Journal of Open Source Software, 2017.
- [5] Mateusz Cholewiński and Alicja Mazur. Influence of choosing the extending column in trajectory tracking control of ssmp platform using artificial force method. Advances in Intelligent Systems and Computing, 323:129–140, 01 2015.
- [6] Carol Fairchild and Dr. Thomas L. Harman. ROS Robotics By Example - Second Edition: Learning to control wheeled, limbed, and flying robots using ROS Kinetic Kame. Packt Publishing, 2017.
- [7] Michael Ferguson. rosserial @ONLINE, November 2018.
- [8] Scripting News Inc. Xml-rpc, 2018.
- [9] Jackie Kay Ioan Sucan. link, 2018.
- [10] Jackie Kay Ioan Sucan. Urdf, 2018.
- [11] Jackie Kay Ioan Sucan. Urdf, 2018.
- [12] Lentin Joseph. Mastering ROS for Robotics Programming. Packt Publishing, 12 2015.
- [13] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf. A flexible and scalable slam system with full 3d motion estimation. In Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR). IEEE, November 2011.
- [14] Anis Koubaa, editor. Robot Operating System (ROS). Springer International Publishing, 2016.

- [15] Jet Propulsion Laboratory. Spirit struggles to survive the martian winter, 2006.
- [16] Matlab. link, 2018.
- [17] Wim Meeussen. Create your own hardware interface @ONLINE, November 2018.
- [18] Kevin Watts Blaise Gassend Morgan Quigley, Brian Gerkey. joy, 2018.
- [19] Eduardo Munera, Jose-Luis Poza-Lujan, Juan-Luis Posadas-Yague, Jose Simo, and J. Francisco Blanes Noguera. Distributed real-time control architecture for ros-based modular robots. IFAC-PapersOnLine, 50(1):11233 – 11238, 2017. 20th IFAC World Congress.
- [20] ROS.org. Parameter server, 2018.
- [21] Maciej Trojnacki. Dynamics Model of a Four-Wheeled Mobile Robot for Control Applications – A Three-Case Study, volume 323. 01 2014.
- [22] Wim Meeussen Tully Foote, Eitan Marder-Eppstein. tf, 2018.
- [23] wiki.ros.org. Writing a simple publisher and subscriber (c++), 2018.
- [24] Xilinx. Block memory generator v8.2, 2015.
- [25] Xilinx. Axi uart lite v2.0, 2017.