Recent Innovations in Mechatronics (RIiM) Vol. 5. (2018). No. SI.
DOI: 10.17667/riim.2018.si/14.

Omnidirekcionális robot megvalósítása, dinamikai és mozgásszabályozási tulajdonságainak mérése

Szerző: Nagy Dániel
Mechatronikai Tanszék
Debreceni Egyetem, Műszaki Kar
Debrecen, Magyarország
nn.ddaannyy@gmail.com

Dr. Szemes Péter Tamás
Mechatronikai Tanszék
Debreceni Egyetem, Műszaki Kar
Debrecen, Magyarország
szemespeter@eng.unideb.hu

Dr. habil. Husi Géza
Mechatronikai Tanszék
Debreceni Egyetem, Műszaki Kar
Debrecen, Magyarország
husigeza@eng.unideb.hu

***Absztrakt*— Ez a kutatás a National Instruments által forgalmazott omnidirekcionális platformok teljestő képességét foglalja össze.**

I. BEVEZETŐ

Egyre több háztartásban találhatók meg például az automata robotporszívók. A 2017-ben Berlinben megrendezett IFA kiállításon a Panasonic bemutatta okoshűtőjét [1], amely hang vezérelt intelligenciával van ellátva. A megfelelő jelzésre „odamegy”, így anélkül vehetünk ki bármit belőle, hogy fel kellene állnunk a székről. Ezen találmányok hatására létrehoztunk egy olyan omnidirekcionális robot prototípusát, amely később hasonló, vagy bonyolultabb feladatok ellátására képes.

II. AZ OMNIDIREKCIONALITÁS ALAPJAI

A. Omnidirekcionális kerék

Az omnidirekcionális robot alapja egy speciális típusú kerék [1], melyet 1918-ban Joseph Grabowiecki talált fel. Az omnidirekcionáis kerék, olyan kerék, melynek a kerületén szabadon forgó hengerek vannak elhelyezve. A kialakításának köszönhetően oldalirányba is tud haladni a kerék. Ezen tulajdonságait kihasználva olyan robotokat lehet építeni, melyek képesek bármely irányba elmozdulni egy adott síkon anélkül, hogy el kellene fordulniuk.

B. Robot felépítmények

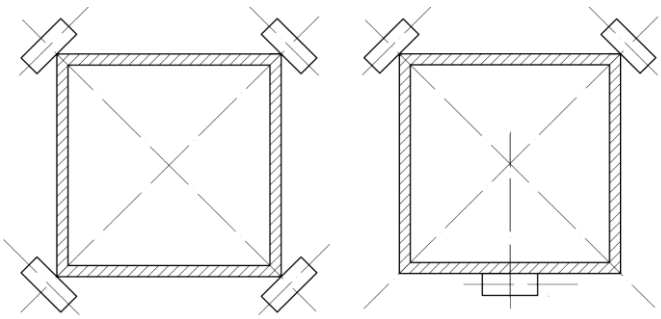
Az omnidirekcionális hajtás esetén, a felépítmények tanulmányozásakor két típust vizsgáltunk meg tüzetesebben. Az első négyzet alakú váz, melynek a csúcsaiban 45 fokkal elforgatva található 4 omnidirekcionális kerék (1. ábra bal). A második egy téglalap alakú váz, melynek az első két kereke ugyanúgy a téglalap csúcsánál találhatóak 45 fokos szögben, míg hátul egy kerék található (1. ábra jobb), a téglalap alsó oldalával párhuzamosan, az oldalfelezőn elhelyezve.

A tervezett robot elkészítéséhez mindenki által hozzáférhető, kereskedelmi forgalomban kapható, eszközöket használtunk. Ez nagyban megkönnyíti az eszköz reprodukálását, illetve sérülés esetén az egyes elemek cseréje is egyszerűen megoldható.

III. HASZNÁLT ESZKÖZÖK

A. Tetrix fémépítő készlet

Az amerikai Pitsco cég által tervezett és forgalmazott Tetrix fémépítő készletből áll a robot váza. Ezek az elemek erős alumíniumból készültek. A kötőelemek acél csavarok és anyák, melyek szintén a készlet részét képezik. A váz a stabilitás miatt U-profilokból áll, ezek ugyanis kevésbé hajlamosak az elhajlásra, mint a lapos elemek.



1. ábra Robot felépítmények

B. NI myRIO [2]

A szintén amerikai székhelyű National Instruments terméke az NI myRIO. Ez az eszköz kifejezetten egyetemi projektet és kutatások megvalósításának elősegítésére készült. Az eszközben található egy FPGA chip, mely eredeti funkciója szerint a kimeneteket definiálja, de ez bármikor átprogramozható. Az FPGA chip mellett egy két magos ARM mikroprocesszor, analóg és digitális ki- és bemenetek, gyorsulásmérő, wifi kommunikáció, visszajelző ledek és egy gomb is található az eszközben.

C. DC motor, enkóder

A projektben használt DC motor és a hozzátartozó enkóder is a Tetrix készlet részei. A motor [2] egy 12 Voltos, szénkefés, egyenáramú motor. A motorra egy 1/72-es áttételű hajtómű csatlakozik, mely csökkenti a fordulatszámot, de növeli a motor által kifejtett nyomatékot. Maximális fordulatszáma 152 [RPM]. A motorhoz tartozó optikai

enkóder 1440 [PPR] felbontású és 5 [V] feszültség szükséges a működtetéséhez.

D. *Akkumulátor*

Az elektromos részek áramellátásáért egy 3 cellás, cellánként 4,2 [V] feszültségű, összesen 5100 [mAh] kapacitású Litium-Polimer akkumulátor a felelős.

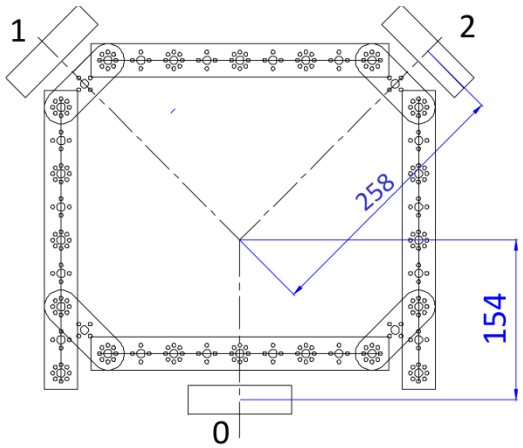
E. *LabView*

A myRIO vezérléséhez LabView alapú programkód írása szükséges. A National Instruments saját fejlesztésű grafikus felületű programja. Ahhoz, hogy a roboton futtatható legyen a kód, egy valós idejű alkalmazást kell fejleszteni és rátölteni a myRIO-ra.

IV. A VÁLASZTOTT ROBOT TÍPUS TULAJDONSÁGAI

A korábban felvázolt robot típusok közül a 3 kerekű változat megvalósítása mellett döntöttünk. Ennek a típusnak egy kicsit bonyolultabb a fizikája, mint a 4 kerekűnek, viszont ennél biztosan nem fordulhat elő, hogy az egyik kerék a levegőbe emelkedik, és nem a talajon végzi el a megadott fordulást, ami rossz irányba való elmozdulását eredményezné a robotnak. Az omnidirekcionális robotok megfelelőségének a lényege, hogy a kerekek forgásközéppontjai tengelyeinek meghosszabbításai egy pontban metsszék egymást. Ez látható a 2. ábrán.

Az 1. táblázat bemutatja a robot mozgatásához szükséges kerékelfordulások egyenleteit. A sebességek indexei a 2. ábrán látható kerék elosztásnak felelnek meg, a mennyiségek kerületi sebességet jelentenek. Az első két kerék távolsága a robot forgásközéppontjától a=258 [mm], a hátsó kerék távolsága a robot forgásközéppontjától b=154 [mm].



2. ábra Robot terv

Mozgás	Kerekek kerületi sebességei		
	v_0	v_1	v_2
Előre	0	$-v_r * \sin(45°)$	$v_r * \sin(45°)$
Hátra	0	$v_r * \sin(45°)$	$-v_r * \sin(45°)$
Jobbra	v_r	$-v_r * \sin(45°)$	$-v_r * \sin(45°)$

Mozgás	Kerekek kerületi sebességei		
	v_0	v_1	v_2
Balra	$-v_r$	$v_r * \sin(45°)$	$v_r * \sin(45°)$
Fordulás jobbra	v_f	$v_{fs} * \frac{258}{154}$	$v_f * \frac{258}{154}$
Fordulás balra	v_f	$v_f * \frac{258}{154}$	$v_f * \frac{258}{154}$

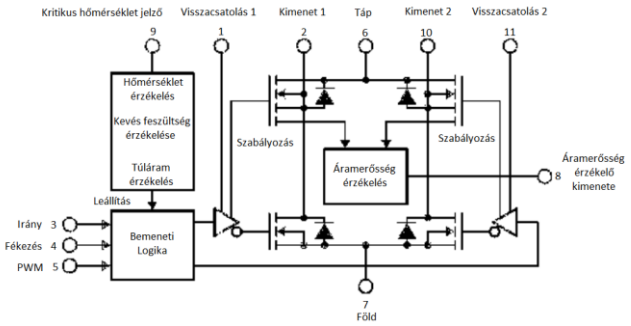
I. TÁBLÁZAT KEREKEK KERÜLETI SEBESSÉGEI

Előre, hátra, illetve jobbra, balra haladáskor az első kerekek mindig 45°-os szöget zárnak be a robot haladási irányával, ezért lesz a kerekek sebessége a robot sebessége szorozva sin(45°)-kal. Forduláskor, mivel a hátsó kerék közelebb van a robot forgásközéppontjához, ezért neki kevesebb utat kell megtennie, mint az első kerekeknek. Ezért kell az első kerekek sebességét a leírandó körök sugarának hányadosával szorozni.

V. MOTORVEZÉRLŐ ÁRAMKÖR

Az omnidirekcionális robotok pontos mozgatásának egyik sarkalatos pontja a motorok vezérlésének sebessége és pontossága. Ha az egyik motor hamarabb vagy később indul el, mint a többi, akkor a robot nem kívánt irányba fordulhat el. Ez használhatatlanná teheti a robotot az adott feladatra. Emiatt saját motorvezérlő áramkört terveztünk és építettünk a robothoz. A Texas Instruments LMD18200T [4] típusú integrált áramkörét használtam, mely kifejezetten ilyen egyenáramú motorok hajtására lett tervezve. A motorok adatlapjából kiderül, hogy maximum 1,37 [A] áramerősséget vehetnek fel. A választott integrált áramkör 3 [A] áramerősségig tudja ellátni a kimenetet. Mivel a motor által felvett áramerősség nem haladhatja meg az integrált áramkör által kiadni tudott maximális áramerősség felét, ezért az integrált áramkör megfelel, hozzáadott hűtőborda szükségességét nem indokolta.

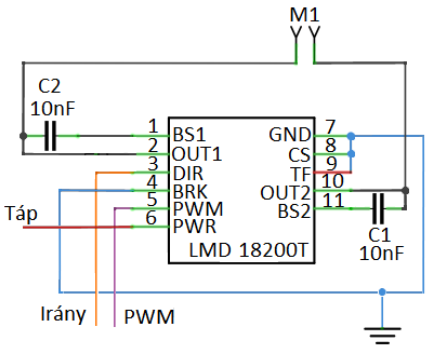
A megfelelő integrált áramkör kiválasztása után a kapcsolási rajz elkészítése következett. Egyelőre nem minden részét használtuk ki az integrált áramkörnek, melynek adatlapjából kivett rajza a 3. ábrán látható. A két kimeneti lábhoz tartozik 1-1 visszacsatolás is, melyet egy 10 [nF]-os kondenzátoron keresztül kötöttük be. Ezeknek annyi a szerepük, hogy az integrált áramkörben lévő tranzisztornak a munkapontját beállítsák. A kondenzátorra azért van szükség, hogy a motor miatt fellépő esetleges feszültségingadozást kivédje, a kapacitása az integrált áramkör adatlapja alapján lett kiválasztva.



3. ábra Integrált áramkör rajza, [4] alapján

A tápfeszültségen és a földelésen kívül az irány és a PWM lábát kötöttük be az integrált áramkörnek. Az irány (Irány) láb egy digitális bemenetet. Ha 0 értéket kap, a motor az óra járásával megegyező irányba forog, ha pedig 1 értéket kap, akkor a másik irányba forog. A PWM lábra értelemszerűen PWM jelet kell kötni. A myRIO FPGA kódját úgy módosítottuk, hogy az integrált áramkörök PWM bemeneteihez kapcsolt MyRIO kimenetek PWM jelet adjanak ki. Ez a myRIO-ban viszonylag egyszerűen megoldható volt. Az FPGA kódban található változónak egy értéket kell adni 0 és 19999 között, ami a térkitöltése lesz a kiadott PWM jelnek. Ha 0, akkor 0 a térkitöltése, ha pedig 19999, akkor 1 a térkitöltése a kimeneti jelnek.

A teljes kapcsolási rajzon (5. ábra) a motorvezérlő áramkör megvalósítása található. Azért van 4 integrált áramkör, hogy a későbbieknek a 4 motoros felépítmény is megvalósítható legyen anélkül, hogy új áramkört kellene tervezni és megépíteni. A jelenlegi konstrukció 3 integrált áramkört használ.



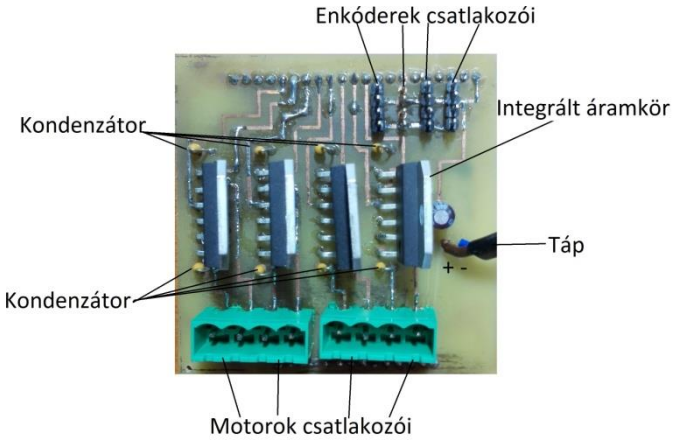
4. ábra Integrált áramkör bekötése

A kapcsolási rajz elkészítése után nyomtatott áramkört terveztünk. A jobb átláthatóság és a sok vezetősáv miatt kétoldalasra terveztük az áramkört úgy, hogy a tüskesorok illeszkedjenek a myRIO-hoz kapott próbapanel csatlakozásaihoz. Így csupán tüskesorra volt szükségünk, nem kellett a myRIO-hoz való speciális csatlakozóból beszereznem. Az elkészült áramkörről látható egy fotó a 6. ábrán.

Az áramkör megépítése után a robot vezérléséért felelős szoftver implementálása következett. Ahogy már korábban említettük, a program Labview környezetben íródott.

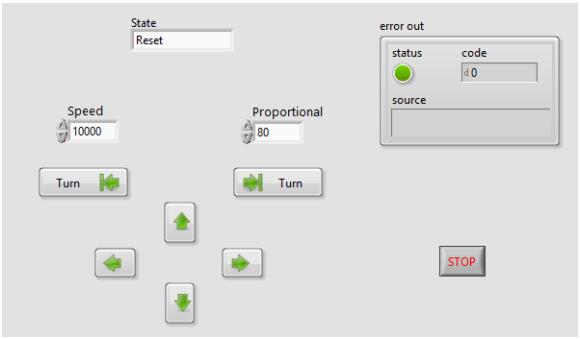
A. *Front Panel*

A Front Panel részen kaptak helyet a vezérléshez használatos gombok. 4 nyíl a 4 ortogonális irányú mozgatáshoz, kettő pedig az óramutató járásával megegyező, illetve az óramutató járásával ellentétes irányú forduláshoz. A vezérlő gombokon kívül itt található a sebesség megadásához szükséges bemenet, az arányos tag bemenete és a program megállításáért felelős Stop gomb is. Ahogy a 6. ábrán is látható egy hiba kimenet is van a programban, ami a futás után kijelzi, ha esetleges hiba vagy figyelmeztetés történt.



5. ábra Kész motorvezérlő

VI. BEÁGYAZOTT VEZÉRLŐ SZOFTVER



6. ábra Front Panel

A. *Block Diagram*

A program inicializálással kezdődik. Itt nullázzuk a változókat, Start-ra állítjuk a kezdő állapotot, inicializáljuk a myRIO-t és betöltjük a konfigurált FPGA kódhoz tartozó bit fájlt. A szoftver váza egy állapotgép, mely ideális az ilyenfajta kódoknál. Az alábbi 7. ábrán jól látható a program működésének logikája. A kezdeti állapotban, vagyis a Start állapotban van a program addig, míg valamelyik gombra rá nem kattintunk a Front Panelen. Ekkor egy másik állapotba ugrik a szoftver, attól függően, hogy melyik gombot nyomtuk meg. Ha a Fel vagy Le gombok valamelyikét, akkor az

Előre/hátra mozgás állapotba kerülünk. Itt a korábban megadott arányos tag, illetve sebesség változók értékei függvényében értéket adunk az FPGA kód PWM kimeneteinek, mely hatására a robot elindul. A sebesség szabályozásához használt képlet a következő:

$$\left(\frac{|C_1|+|C_2|}{2} - |C_x|\right) \cdot P + S \qquad (1.)$$

A C_1 és C_2 értékeket az enkóderek adják. Az adott kerék egy teljes körbefordulása esetén ez az érték 1440. Abszolút értéket veszünk, mert a forgás irányától függően negatív számot kaphatunk, nekünk viszont a mért érték 0-tól való távolsága számít. A kivonással megkapjuk az adott motor jelenlegi helyzetének eltérését az átlagtól. Ezt az eredményt szorozzuk egy erősítő tényezővel, majd hozzáadjuk az elérni kívánt sebességet. Így ha az egyik kerék kevesebbet fordult, mint a másik, akkor növeli a sebességét, és fordítva. Ezzel elérhető, hogy szépen haladjon a robot. A C_x indexe lehet 1, vagy 2, attól függően, hogy melyik motor képletét vizsgáljuk.

Az értékadás után a program visszatér a Start állapotba, ahol ismét döntés következik, hogy mi a következő állapot. Ha kizárólag a Bal vagy Jobb gombok valamelyikére kattintunk, a Balra/Jobbra mozgás állapotba kerül a program. Ez nagyban hasonlít az előbb taglalt állapothoz, csupán a PWM jelek értékeinek kiszámítási módszere más. Az itt használt képletek a következőképpen néznek ki:

$$\left(\frac{|C_0 \cdot \sin(45°)| + |C_1| + |C_2|}{3} - |C_x|\right) \cdot P + S \qquad (2.)$$

$$\frac{\left(\frac{|C_0 \cdot \sin(45°)| + |C_1| + |C_2|}{3} - |C_0 \cdot \sin(45°)|\right) \cdot P + S}{\sin(45°)} \qquad (3.)$$

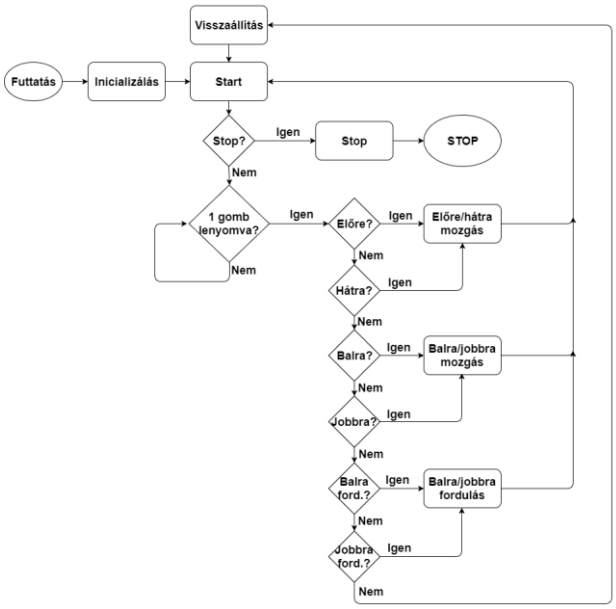
A C_1, C_2, C_x, P és S változók a korábban leírtaknak felelnek meg. A C_0 a harmadik, vagyis a hátsó motorhoz tartozó enkóder értéke. Az előzőekben erről azért nem volt szó, mert a robot kialakítása miatt előre és hátra menetben ez a kerék nem forog, tehát az értéke ilyen esetekben mindig 0. Balra, jobbra mozgáskor viszont szükség van rá, ezért itt már a képletben is szerepel. A felső képlet az első két motorhoz tartozó PWM jelek értékének meghatározásához szükséges, míg az alsó a hátsó kerékhez tartozó motor PWM értékét adja meg. Korábban a Választott robot típus tulajdonságai rész alatt leírtak miatt található a képletben a sin (45°)-os szorzás, illetve osztás. A különbség annyi, hogy ott a két első kerék kimeneti értékét szoroztuk sin (45°)-kal, itt pedig a hátsó kerék értékét osztjuk ugyanezzel az értékkel. Az eredményen ez nem változtat, csupán egyenletrendezésről van szó.

Miután megtörtént az oldalirányú mozgás a program újból visszatér a Start állapotba. Innen egy következő állapot lehet a Fordulás állapota, mely a fordulás gombok valamelyikének lenyomásakor érhető el. Ez az állapot az előzőhöz hasonlóan mind a 3 motort hajtja, és itt is eltér az első kerekek PWM jel értékének kiszámítása a hátsó kerék értékének kiszámításától. Az itt használt hányados szintén a Választott robot típus tulajdonságai fejezetben taglaltak szerint van. Alább láthatók a használt képletek. Előbb az első kerekekhez, majd a hátsó kerékhez tartozó számítások.

$$\left(\frac{\left|C_0 \cdot \frac{258}{154}\right| + |C_1| + |C_2|}{3} - |C_x|\right) \cdot P + S \qquad (4.)$$

$$\frac{\left(\frac{\left|C_0 \cdot \frac{258}{154}\right| + |C_1| + |C_2|}{3} - \left|C_0 \cdot \frac{258}{154}\right|\right) \cdot P + S}{\frac{258}{154}} \qquad (5.)$$

Az elmozdulás után innen is a Start állapotba térünk vissza. Ahogy az a korábban említett folyamatábrán is jól látható, két másik lehetséges állapotba kerülhet még a program a korábban taglaltakon kívül. Az egyik állapot akkor következik be, ha nem egy gombot nyomunk meg egyszerre, vagy egy gombot sem tartunk lenyomva. Ilyenkor a Visszaállítás állapotba kerül a szoftver, ahol megállítjuk a motorokat és nullázzuk az enkóderek értékeit.



7. ábra Szoftver folyamatábra

Ezekre azért van szükség, hogy bármilyen elmozdulás után bármilyen következhessen anélkül, hogy a robot rossz értékeket kapna. Ezt az állapotot is a Start állapot követi, ahonnan az utolsó állapot ahova kerülhetünk a Stop állapot. Ha a Front Panelen található Stop gombot megnyomjuk, a Start állapotból rögtön a Stop állapotba kerülünk. Itt a Visszaállítás állapothoz hasonló műveletek végzünk el: leállítjuk a motorokat és nullázzuk az enkóderek értékeit. Az egyetlen különbség, hogy ez az állapot leállítja a program fő részében található ciklust. A ciklus leállítása után a program a lezárás részbe kerül, ahol visszaállítja a myRIO-t az indulási állapotba, majd az egész szoftver leáll.

VII. SZABÁLYOZÁS, PÁLYAKÖVETÉS

Az előzőekben leírt program megvalósítása után teszteltük a robot, majd átalakítottuk és tovább fejlesztettük a motor vezérlő kódot. Az első feladat a robot mozgásának kiterjesztése volt. Mindenképpen szerettünk volna olyan kódot írni, mely lehetővé tesz a robot adott síkon belüli bármely irányba való mozgását. Ehhez a korábbiakban használt képletek alakítottuk át úgy, hogy magukban foglalják a megadott irányt. A Front Panelen helyet kapott egy Irány nevű változó, melyben 0 és 360 közötti egész számmal lehet megadni, hogy milyen irányba mozduljon el a robot. A 0 jelenti az előre haladást, a 90 a balra haladást és így tovább. Az átalakított képletek a következők:

$$PWM_x = |\sin(-\alpha)| \cdot S + \{dS_x \cdot P + (dS_x + dS_x SUM) \cdot I + (dS_x - dS_x PREV) \cdot D\} \quad (6.)$$

A képletben található „x” indexek az adott motor sorszámát jelentik. Így a képletben használt dSx értékek a következők:

A képletekben található α értéke az irány értékével egyenlő, amerre a robotnak el kellene mozdulnia.

A képletek implementálása után a PID szabályozás megfelelő tagjainak a megtalálása következett. Ezt kísérleti úton tettük meg. A tagokat egyesével változtattuk meg, majd vizsgáltuk a változtatás hatását a robot viselkedésére. Hosszas kísérletezés után megtaláltuk az ideálisnak mondható értékeket. A robot tesztelésekor ezeket az értékeket használtuk.

$$dS_0 = \frac{\frac{ENC_0 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(-\alpha)|} + \frac{ENC_1 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha - 45)|} + \frac{ENC_2 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha + 45)|}}{3} - \frac{ENC_0 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(-\alpha)|} \quad (7.)$$

$$dS_1 = \frac{\frac{ENC_0 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(-\alpha)|} + \frac{ENC_1 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha - 45)|} + \frac{ENC_2 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha + 45)|}}{3} - \frac{ENC_1 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha - 45)|} \quad (8.)$$

$$dS_2 = \frac{\frac{ENC_0 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(-\alpha)|} + \frac{ENC_1 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha - 45)|} + \frac{ENC_2 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha + 45)|}}{3} - \frac{ENC_2 \cdot \frac{100 \cdot \pi}{1440}}{|\sin(\alpha + 45)|} \quad (9.)$$

A szabályozás megvalósítása után egy világkoordináta rendszert definiáltunk. Ez a teszteléskor használt helyszín egy kinevezett része. A programot kiegészítettük, hogy a világkoordináta rendszerben lehessen megadni, hogy hova menjen a robot. A Front Panelen 3 változóban meg kell adni a robot kiindulási helyzetét. Két koordinátával és egy szöggel. A két koordináta a robot forgásközéppontja a világkoordináta rendszerben, míg a szög a robot saját koordináta rendszerének a világkoordináta rendszerrel bezárt szöge. Ezután ha megadunk egy X és egy Y értéket a világkoordináta rendszeren belül, a robot odamegy.

VIII. TESZTELÉS

A program megvalósítása után a kész rendszer tesztelése következett. A korábban implementált programot is teszteltük, majd a szabályozás utáni eredményekkel összehasonlítottam.

A 8. ábrán látható oszlopok a robot által megtett távolságok eltérését ábrázolja a megadott értékekhez képest. A fentebbi ábra a szabályozás nélküli, a lenti pedig a szabályozás megvalósítása utáni eredmények.

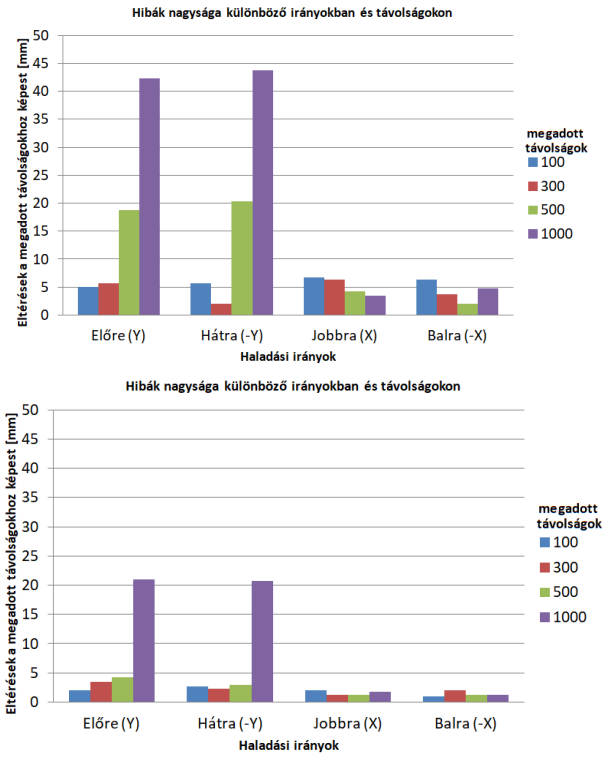
IX. EREDMÉNYEK

A kutatás megvalósítási ideje alatt sikerült megvalósítani a kitűzött feladatokat. A robot alakjának megtervezése, mechanikai megépítése, az áramkör tervezése, megépítése és a szoftver implementációja is megvalósult. A kezdeti szoftvert kiegészítettük szabályozással és a világkoordináta rendszer megvalósításának köszönhetően pontosan vezérelhető a robot mozgása. Az alábbi 9. ábrán látható két fotó az elkészült robotról.

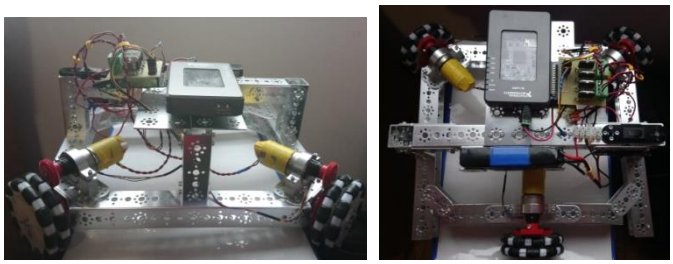
KÖSZÖNETNYILVÁNÍTÁS

A publikáció elkészítését az EFOP-3.6.1-16-2016-00022 számú projekt támogatta. A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

Köszönet az NI Hungary Kft. munkatársainak, hogy rendelkezésemre bocsájtották a dolgozatban leírt robot megépítéséhez szükséges hardvereket és szoftvereket. Külön köszönet Kökényesi Illésnek, aki erős szakmai tudásával segítette a munkámat.



8. ábra Robot pontosságának tesztelése



9. ábra Az elkészült robot

HIVATKOZÁSOK

[1] Discover Magazine. [Online]. Available: http://discovermagazine.com/1997/jul/1997discoverawar1170

[2] NI myRIO. [Online]. Available: http://www.ni.com/pdf/manuals/376047c.pdf.

[3] Pitsco. [Online]. Available: https://www.tetrixrobotics.com/TETRIX-DC-Gear-Motor.

[4] Texas Instruments. [Online]. Available: http://www.ti.com/lit/ds/symlink/lmd18200.pdf.