SZAKDOLGOZAT

Kádár Bence

Pethe Szabolcs

Debrecen

2011

DEBRECENI EGYETEM
INFORMATIKAI KAR

Egyensúlyozó robot építése és modellezése

Témavezető:
Dr. Cserháti Csaba
egyetemi docens
DE TTK
Szilárdtest fizika tanszék

Készítette:
Kádár Bence
Pethe Szabolcs
Mérnök informatikus hallgatók

Debrecen
2011

TARTALOMJEGYZÉK

BEVEZETÉS

Szakdolgozatunk alapját a Yorishima Yamamoto által fejlesztett NXTway-GS képezi, ami egy LEGO Mindstorms NXT 2.0 csomag felhasználásával épített önállóan egyensúlyozó robot, valamint a hozzá Simulinkben készült szimuláció. Ezt a robotot és a szimulációját szeretnénk megismerni, a hiányosságait kívánjuk orvosolni, valamint olyan módon feldolgozni, hogy egy esetleges továbbfejlesztéshez megfelelő alapot képezzen.
A következő pontokat jelöltük ki fejlesztési célként:
- a robot egyszerű irányítása számítógépről
- a Simulinkben készített szimuláció irányíthatóvá tétele
- a Simulinkben általunk irányított mozgás naplózása, majd egy programmal történő feldolgozása, amely segítségével a szimulációban készült útvonalat a robot a valóságban meg tudja ismételni.

Ezekkel a fejlesztésekkel a robot már nem csak az egyensúlyozásra lesz képes, hanem különböző célfeladatokra tesszük alkalmassá, úgy mint felderítő robot vagy szállító platformrobot.

Napjaink műszaki felsőoktatásának fontos feladata Magyarországon az informatika- és a programozás-oktatás valamint a robotika. Kiemelten fontos feladat minden hallgató számára a megfelelő programozói ismeretek kialakítása, valamint az algoritmikus gondolkodásmód és a problémamegoldó képesség magas szintre juttatása. A programozás a robotokon keresztül gyakorlatiasabb, könnyebb a tanulók érdeklődését felkelteni és a motiváltságot fenntartani, segítségével a tanórák látványosabbá tehetők. Ha a tananyag érdekes, a tanulás szórakoztató és az eredményesség is jelentősebb, mindenki szabadon használhatja találékonyságát, és ha sikerül a kitűzött feladatot önállóan megoldani, átélhetjük a felfedezés izgalmát és diadalát. A robotok programozása lehetővé teszi a hallgatók számára, hogy a hagyományos programozás-tanulás során azonnal elvárt absztrakt gondolkodást megelőzze a készségek tapasztalati szintű használata.

A LEGO cég olyan robotkészletet fejlesztett ki, amelyik a robotika és a robotprogramozás alapjait tanulók számára az egyik legjobb segédeszközt adja [1]. A programozható LEGO RCX és NXT robotok felhasználhatók az oktatás különböző életkori- és tudásszintjén. Az eszköz használata során a hallgatók megismerkedhetnek számos programnyelvvel és programozói stratégiával, szembesülhetnek különböző gyakorlati

szituációkkal. Kiváló eszközök a programozás megszerettetése, motiváció növelésére (kezdőknek BSC szinten), de lehetőséget nyújtanak a már megszerzett elméleti ismeretek gyakorlati megvalósítására (haladóknak akár MSC szinten is) [2].

I. FEJEZET

A ROBOTOK ÉS AZ EGYENSÚLYOZÓ ROBOTOK TÖRTÉNETE

Robotok

A robot megnevezés a szláv robota, azaz munka szóból ered. Elektromechanikai szerkezet, amely előzetes programozás alapján képes különböző feladatok végrehajtására. Lehet közvetlen emberi irányítás alatt, de önállóan is végezheti a munkáját egy számítógép felügyeletére bízva. Olyan gép, amely képes megváltoztatni a környezetet, amelyben működik.

A robotokkal rendszerint olyan munkákat végeztetnek, amelyek túl veszélyesek, túl nehezek az ember számára vagy egyszerűen túl monoton, de nagy pontossággal végrehajtandó feladat, ezért egy robot nagyobb biztonsággal képes elvégezni. Robotokat hadi célokra is felhasználnak. A katonai célokra készült robotok feladata általában felderítés. A robotok fő megrendelője és kutatásfinanszírozója a hadsereg és az űrkutatás.

Egészen 1950-ig a robot szó nem volt elterjedt. Isaac Asimov tette azzá, mikor *Én, a robot* című novellásgyűjteményében lefektette a robotika három alaptörvényét annak érdekében, hogy a robotok emberek fölé kerekedését megelőzze.

1. A robot nem árthat az embernek, és nem nézheti tétlenül, ha az embert veszély fenyegeti.
2. A robot engedelmeskedni tartozik az emberek parancsainak, kivéve, ha ezek a parancsok az Első Törvénybe ütköznek.
3. A robot köteles megvédeni magát mindaddig, amíg ez nem ütközik az Első vagy a Második Törvénybe.

A robotika három törvénye Isaac Asimov kitalációja**,** az összes gyártott robot közös értékrendje. Az alaptörvények nélkül szinte lehetetlen az emberek és a robotok együttélése.
1950 táján a tudósok úgy hitték, az intelligens robotok készítésétől mindössze pár év választja el őket, ezért is kerültek előtérbe a fentebb említett problémák. A gyakorlatban azonban még a fejlettebb állati intelligencia megvalósítását napjaink technikai színvonala sem teszi lehetővé [3].

A jövő mindenképpen abba az irányba mutat, hogy az „intelligens” eszközök egyre elterjedtebbek lesznek. Először a szórakoztatóelektronikai eszközökön keresztül, később a háztartási eszközökbe építve a robotika megjelenik majd a mindennapokban. Végül a mikroelektronikai eszközök fejlődésének hatására előbb-utóbb elterjedhetnek majd az univerzális, intelligens robotok is.

A robotok általános felépítése

- Mechanikai rendszer – az akciót megvalósító részrendszer
    - Helyváltoztató berendezés – mozgás a környezetben
    - Manipulációs berendezés – környezetbeli objektumokkal való operálás
- Szenzoros rendszer – az érzékelést megvalósító rendszer
    - Belső állapot – mechanikai rendszer állapota
    - Külső állapot – környezet állapota
- Irányítórendszer – az akció és az érzékelés közti kapcsolat
    - Döntés egy adott akcióról a mechanikai rendszer és a környezet által jelentett korlátozásokat figyelembe véve [4]

Kétkerekű, egyensúlyozó robotok

A kétkerekű, egyensúlyozni képes robotok a klasszikus kocsi-rúd kísérleten alapulnak, melyben egy szabályozási rendszerrel próbálunk egyensúlyban tartani egy rudat, ami az egyik végén a kocsihoz van rögzítve, a szabad mozgása pedig egy tengelyre korlátozódik, ugyanarra, amelyen a kocsi is mozoghat. Ez a modell a fordított inga problémájaként is ismert.

A fordított inga problémája nem ismeretlen a szabályozás területén. A technológia egyedülállósága és széleskörűen alkalmazhatósága sok kutató és a robotok iránt rajongó figyelmét keltette fel. Az utóbbi évek során olyan kutatásokban jelentkezik a mozgó fordított inga modellje, mint például az autonóm kerekesszék, a humanoidok felegyenesedett járásának megvalósítása vagy a személyi szállítóeszköz rendszerek.

Amennyiben a robotot képessé tesszük arra, hogy külön, más-más sebességgel fordulhassanak a kerekei, lehetővé tesszük, hogy két tengely mentén mozogjon.
Napjainkban a kétkerekű, egyensúlyozó rendszereket általában a Segway robotként azonosítják, utalva Dean Kamen 2001-es találmányára, a Segway PT-re.

A Segway

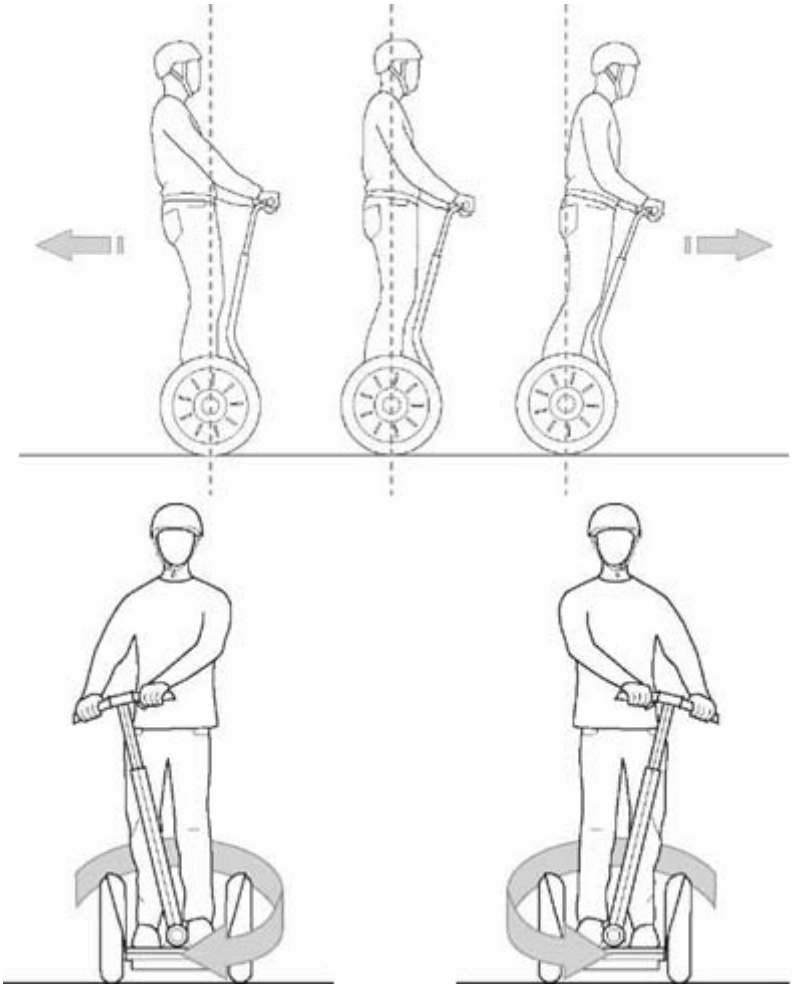
1999-ben egy híres feltaláló, Dean Kamen céget alapított Segway LCC néven, azzal a céllal, hogy létrehoznak egy olyan személyi szállításra alkalmas járművet, ami kiemelkedő hatásfokkal rendelkezik, nincs károsanyag-kibocsátása és dinamikus stabilizációt alkalmaz működése közben. 2001. december 3-án a cég bejelentette az első, önállóan egyensúlyozó szállító eszközt, a Segway Personal Transporter-t (Segway Személyi Szállítóeszköz) [5].

1. ábra: Az egyik napjainkban kapható változat, a Segway i2

Hogyan működik a Segway

A Segway az emberi test dőlését használja a gyorsításhoz és a lassításhoz, minél előrébb döntjük rajta testünket, annál gyorsabban halad előre, a maximális, körülbelül

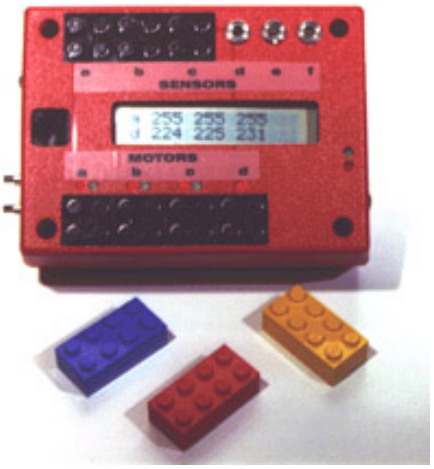
20km/h-s sebességig, a tolatást pedig a test hátra döntésével tudjuk elérni. A kormány fordításával tudunk jobbra vagy balra kanyarodni, ekkor a fordulást az okozza, hogy egyik kereke gyorsabban forog, mint a másik. A Segway PT képes egy helyben is forogni, ekkor a két kerék ellenkező irányba forog. A szabályozást 5 giroszenzorral és 2 gyorsulásmérő érzékelővel oldották meg, működése közben a test helyzetét és a terep viszonyain bekövetkező változásokat másodpercenként 100-szor vizsgálja. Az 5 giroszenzor közül csak 3 szükséges a különböző mozgások érzékelésére, a többi szenzor a termék megbízhatóságát növeli.



2. ábra: A Segway előre és hátra történő mozgatása, valamint kormányzása [6]

A LEGO Mindstorms története

A programozható LEGO elemek története az MIT-ről (Massachusetts Institute of Technology) indult. Itt kísérleteztek először azzal, hogy a hallgatók robotika oktatását Technics LEGO elemekkel tegyék gyakorlatiassá. A LEGO Technics elemei már korábban is alkalmasak voltak mechanikai szerkezetek építésére, melyet kis elektromos motorok segítségével is működésbe hozhattunk. Hiányzott azonban a vezérlő elektronika megléte. Saját vezérlő egységet építettek, ami alkalmas volt a 9V-os LEGO motorok meghajtására. Ezek a központi egységek azonban nem voltak egyszerűen megépíthetők és programozhatók, ezért a mérnökhallgatók képzésén kívül nem terjedtek el.



3. ábra: Az MIT-n kifejlesztett vezérlőegység

Ezzel párhuzamosan, némileg eltérő úton fejlődtek a LEGO cég termékei, melyek kezdetben nem voltak alkalmasak mobil robotok létrehozására, mert a program a számítógépen futott, amely egy adatátviteli kábelen keresztül kapcsolódott a robothoz. A programozható LEGO RCX tégla a két irányvonal tulajdonságainak egyesítéséből jött létre 1998-ban, ez a LEGO Mindstorms első generációja. Teljesen autonóm egység, mely számítógép nélkül is képes működni, 6 darab ceruzaelem biztosítja az érzékelők, motorok és a vezérlőegység számára az energiát.

4. ábra: Az RCX 1.0 verziója, a csatlakoztatható érzékelőkkel és beavatkozókkal

Egy robot működése során környezetéről érzékelői segítségével szerezhet információkat, az adatokat és esetleges előzetes tudását mérlegelve vezérlő, beavatkozó szervei segítségével tud környezetére hatást gyakorolni. Az RCX 3 db érzékelőt és 3 db beavatkozót tud egyszerre kezelni. Az érzékelők és beavatkozók minősége, pontossága alapvetően befolyásolja, hogy mire lesz alkalmas megépített robotunk [7].

A robot második generációja 2006-ban jelent meg LEGO Mindstorms NXT néven, mellyel a Nürnbergi játékkiállítás innovációs nagydíját is elnyerte. Ebben sok formai és technikai változtatást alkalmaztak, valamint kiküszöbölték az RCX robotok kommunikációs problémáit. Az NXT 4 db érzékelőt és 3 db beavatkozót képes egy időben kezelni. A hagyományos LEGO-val ellentétben itt nem csak modellünk alakját, hanem viselkedését is befolyásolhatjuk. A gyártó instrukciója szerint a készletből építhetünk autó vagy emberalak formát, de ezen kívül bármit, amit képzeletünk enged és össze tudunk rakni az elemekből [8]. Az NXT legfrissebb verzióját, az NXT 2.0-t 2009. augusztus 5-e óta forgalmazzák.

A LEGO egyensúlyozó robotok történelme

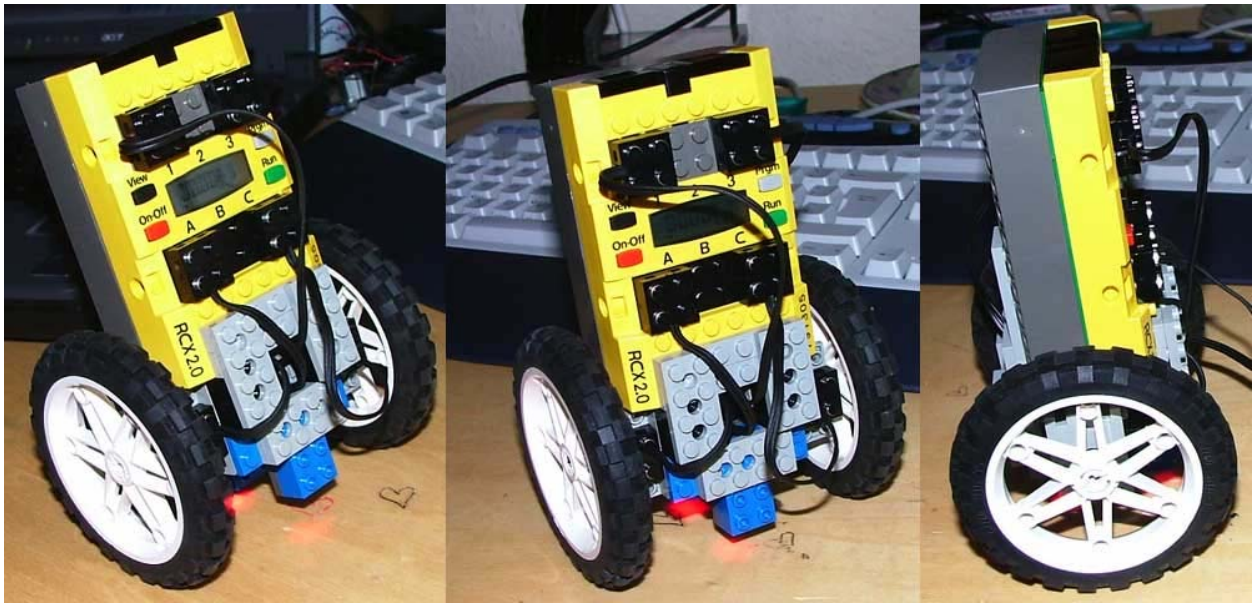
A LEGO által forgalmazott RCX és NXT robotok megfelelő szenzorjai és motorjai, valamint praktikus összeépíthetősége lehetőséget adott arra, hogy többféle egyensúlyozó robotot is kifejlesszenek segítségével.

Ezek közül a legfontosabbak:

- LegWay
- NXTway
- NXTway-G
- NXTway-GS

LegWay – RCX és elektro optikai távolság érzékelők

Steve Hassenplug az elsők között volt, aki LEGO Mindstorms RCX és két elektro-optikai távolság érzékelővel (EOPD) tudott önmagát egyensúlyozni képes robotot építeni, melyet LegWay-nek nevezett el [9]. Két keréken egyensúlyozott és képes volt egy fekete vonal követésére.



5. ábra: A Steve Hassenplug által fejlesztett LegWay

A LegWay az egyik EOPD segítségével tudott egyensúlyban maradni úgy, hogy az EOPD és a föld távolságát próbálta tartani. Úgy programozták, hogy ha csökken az előre meghatározott távolság, akkor előre mozogjon, ha pedig növekszik, akkor tolasson.

A LegWay úgy is működhet, hogy egy helyett mindkét EOPD-t használja az egyensúly megtartásához, ekkor képes a fekete vonal követésére is. Amikor az egyik érzékelő elhalad a fekete vonal felett, megállítja egy pillanatra a megfelelő kereket, így tetszőleges irányba tud fordulni, miközben a robot előre kezdi hajtani a motort, hogy egyensúlyban maradjon.

Az EOPD szenzor a HiTechnic cégtől származik. Egy belső fényforrás segítségével tudja meghatározni a szenzor és a tárgyak távolságát, ebben az esetben a föld a tárgy.

NXTway – NXT és fényérzékelő

Philippe Hurbain-t Steve Hassenplug LegWay-e ihlette, hogy építsen egy hasonló egyensúlyozó robotot, LEGO Mindstorms NXT és fényérzékelő segítségével. A robotot NXTway-nek nevezte el. Az NXTway csak rövid ideig volt képes egyensúlyban maradni, igaz Hassenplug-éhoz képest sokkal kezdetlegesebb volt a szabályozórendszer. Az alábbi képen látható, hogy a LEGO fényérzékelőt a két kerék között helyezték el [10].

6. ábra: Philippe Hurbain NXTway-e

A legmeghatározóbb különbség a LegWay és az NXTway között a távolság mérésére használt szenzorok típusában rejlik. Hassenplug úgy vélte, hogy a LEGO fényérzékelője nem biztosít megfelelő felbontást, így kevésbé alkalmas a célra, nehéz vele egyensúlyozásra képes robotot építeni.

Tekintve, hogy Hurbain NXTway-e nincs olyan jól dokumentálva, mint Hassenplug-é, nehéz a kettőt összehasonlítani. Hurbain megoldásában a robot környezetének fényereje hatással van a szenzor mért értékeire, zavart okozhat a működésben. Emiatt részesíti előnyben Hussenplug az EOPD szenzort, mivel az saját fényt bocsát ki, és ezt detektálva sokkal stabilabban működik, „kiszűrve” a környezetből jövő fényt.

NXTway-G – NXT és giroszenzor

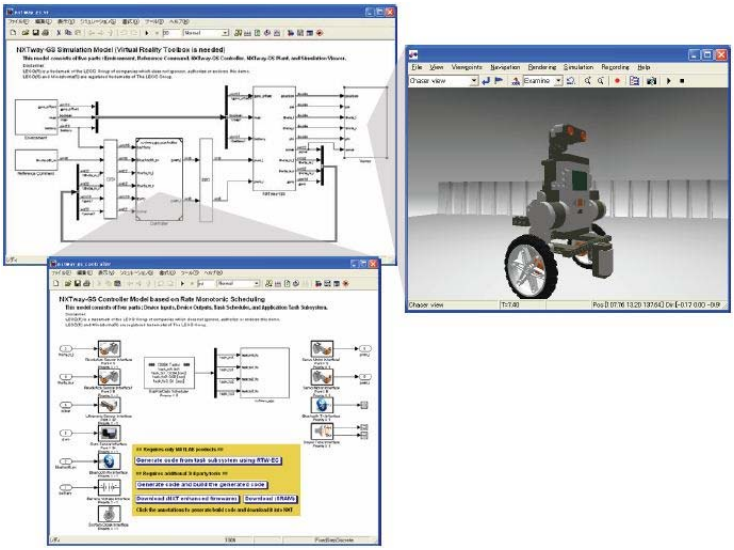
Az NXTway-G megalkotója Ryo Watanabe, a robot Steve Hassenplug LegWay-e és Philippe Hurbain NXTway-e utódjának tekinthető. A két elől egyaránt fényérzékelő szenzorokat használt a robot testdőlésének méréséhez, az NXTway-G-vel ellentétben azonban nem veszik igénybe a kerekek elfordulási szögeinek értékeit, ezáltal nem tudnak belső stabilitást elérni. Watanabe NXTway-G-je egy giroszenzorral és két elfordulás mérővel van szerelve. Az elfordulás mérők a kerekek elfordulási szögét, a giroszenzor a test szögsebességét mérik. A robot tetején egy távvezérlési interfész is helyet kapott, ennek segítségével távirányítása is lehetséges [11].

7. ábra: Az NXTway-G

NXTway-GS – NXT és HiTechnic Gyro Sensor

Az NXTway-G projektet Yorishima Yamamoto továbbfejlesztette, az egyensúlyozáshoz egy HiTechnic Gyro Sensor-t alkalmazott, a robotot képessé tette az akadályoknak ütközés elkerülését, ehhez pedig a Mindstorms készlet ultrahangos érzékelőjét használta. Két további fejlesztést is készített a robothoz. Egyrészt Gamepad-dal tudta irányítani a robot mozgását egy C++ nyelven írt program segítségével [12] Bluetooth kapcsolaton keresztül, másrészt Simulink segítségével szimulálta a robot alapvető mozgásait [11].

8. ábra: Yorishima Yamamoto egyensúlyozó robotja, az NXTway-GS



9. ábra: Egy kép az NXTway-GS szimulációjáról

Az egyensúlyozó robotok összegzése

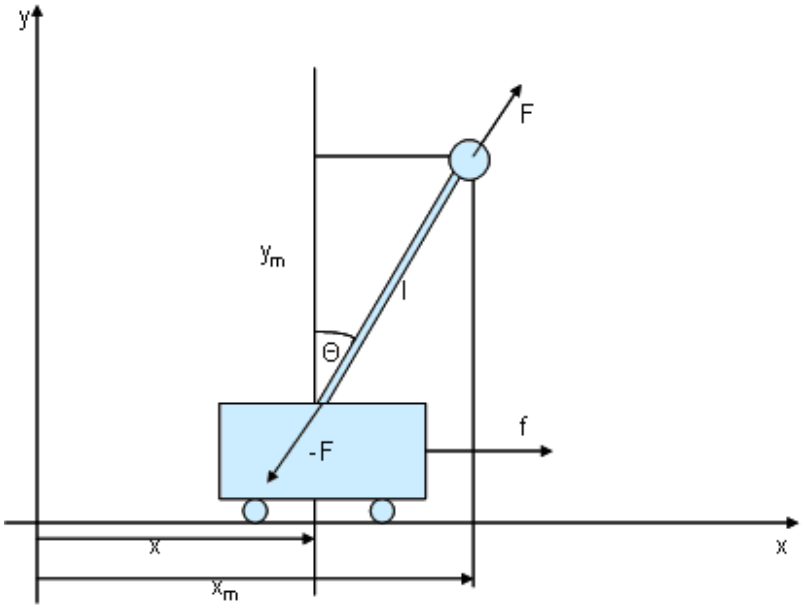
Mind a négy LEGO alapú egyensúlyozó robot projektmunkából fontos információkra tehetünk szert. Először az válik nyilvánvalóvá, hogy a LegWay-nél használt EOPD érzékelő sokkal megfelelőbb a LEGO készletben található fényérzékelőnél a földfelület detektálására. Ezt a tényt a fejlesztő az NXTway hiányosságaként is megemlíette, mivel az alap készletben található fényérzékelő rendkívül érzékeny a környezet fényerejére is. Az NXTway-G a visszavert fény helyett a test fordulási szögében bekövetkező változást észleli egy giroszenzor segítségével, ezzel az érzékelővel egy sokkal pontosabb egyensúlyozás valósult meg. Az NXTway-G fejlesztője egy távirányító interfészt használt a robot mozgásának irányítására, bár véleményünk szerint az NXT saját Bluetooth-os kommunikációs képességeit is használhatta volna. Az NXTway-GS projektben az NXTway-G robotot fejlesztették tovább, giroszenzorként a HiTechnic cég egyik érzékelőjét használva, ezen kívül egy ultrahangos érzékelő segítségével az akadályoknak ütközést is megpróbálja megelőzni a robot. Készült még egy, az egyensúlyozó robot mozgását bemutató szimuláció is, amely azonban korlátozott, mivel a készülék közvetlenül nem irányítható. Ezeket a robotokat kizárólag az egyensúlyozásra tervezték.

II. FEJEZET

A FIZIKAI PROBLÉMA ÉS A SZABÁLYOZÁSI ALAPOK

A fordított inga

A két keréken egyensúlyozó robot rendszerének szabályozása már olyan bonyolultságú, hogy a szabályzó megválasztása, valamit a szabályzó paramétereinek beállítása nem történhet próbálgatással, vagy a jól ismert Ziegler-Nichols féle empirikus beállítással. A fordított inga sok kutatóban keltett érdeklődést, a rendszer instabil természetéből kifolyólag. Az olyan robotok, amik erre épülnek, az utóbbi évek során világszerte nagy érdeklődésnek örvendenek. Mivel ezek a robotok mechanikailag ingatagok, szükségessé válik azon lehetőségek kutatása, hogy miként implementáljuk azt a szabályozási rendszert, ami megvalósítja az egyensúlyozást.



10. ábra: A fordított inga fizikai modellje

A tervezéshez a rendszer mozgásegyenleteiből indulunk ki. Vegyük a 10. ábrán látható fordított inga modelljét. Az M tömegű kocsira szerelt l hosszúságú súlytalan rúd egy csukló körül elmozdulhat. A rúd végére egy m tömegű golyót szereltek. A kocsira f erő hat. Az irányítási feladat az m tömeg felső, labilis egyensúlyi helyzetben tartása a kocsi megfelelő mozgatásával. A szabályozási algoritmust a folyamat modelljének figyelembevételével kell megtervezni.

Adjuk meg a rendszer dinamikai viselkedését leíró Newton egyenleteket a M és m tömegek mozgására [14]. A merev rúdban +F és –F kényszererők ébrednek.

$$M\ddot{x} = f - F\sin\Theta$$
$$m\ddot{x}_m = F\sin\Theta$$
$$m\ddot{y}_m = F\cos\Theta - mg$$

Adjuk össze az első két egyenletet! Szorozzuk meg a másodikat $\cos\Theta$-val, a harmadikat $\sin\Theta$-val, majd vonjuk ki őket egymásból. Így kiesik az F kényszererő és az alábbi egyenletekhez jutunk:

$$M\ddot{x} + m\ddot{x}_m = f$$
$$m\ddot{x}_m\cos\Theta - m\ddot{y}_m\sin\Theta = mg\sin\Theta$$

Számítsuk ki x_m és y_m deriváltjait!

$$x_m = x + l\sin\Theta \qquad y_m = l\cos\Theta$$
$$\dot{x}_m = \dot{x} + l\dot{\Theta}\cos\Theta \qquad \dot{y}_m = -l\dot{\Theta}\sin\Theta$$
$$\ddot{x}_m = \ddot{x} - l\dot{\Theta}^2\sin\Theta + l\ddot{\Theta}\cos\Theta \qquad \ddot{y}_m = -l\dot{\Theta}^2\cos\Theta - l\ddot{\Theta}\sin\Theta$$

Behelyettesítve $\ddot{x}_m$ és $\ddot{y}_m$ kifejezését az előző egyenletekbe, a fordított inga mechanikai viselkedését leíró nemlineáris differenciálegyenletet kapjuk.

$$(M+m)\ddot{x} - ml\dot{\Theta}^2\sin\Theta + ml\ddot{\Theta}\cos\Theta = f$$
$$m\ddot{x}\cos\Theta + ml\ddot{\Theta} = mg\sin\Theta$$

A nemlineáris differenciálegyenletek linearizálására az egyensúlyi helyzet körüli sorbafejtést alkalmazzuk. Fejezzük ki ezért mindkét egyenletből a második deriváltakat.

$$\ddot{x} = \frac{1}{M+m}(f + ml\dot{\Theta}^2\sin\Theta - ml\ddot{\Theta}\cos\Theta)$$
$$\ddot{\Theta} = \frac{g}{l}\sin\Theta - \frac{1}{l}\ddot{x}\cos\Theta$$

Linearizáljuk a rendszert a függőleges helyzet körüli kis elmozdulásokra, azaz $\Theta \approx 0$ és $\dot{\Theta} = 0, \sin\Theta \approx 0, \cos\Theta \approx 1$ közelítések figyelembevételével. Ekkor a fenti egyenletek az alábbi alakra hozhatók:

$$\ddot{x} = -\frac{mg}{M}\Theta + \frac{1}{M} f$$
$$\ddot{\Theta} = \frac{g}{l}\frac{M+m}{M}\Theta - \frac{1}{Ml} f$$

Tekintsük állapotváltozóknak $x, \dot{x}, \Theta, \dot{\Theta}$. Ezen változókkal az állapotegyenlet a következőképpen írható fel:

$$\frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \\ \Theta \\ \dot{\Theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{l}\frac{M+m}{M} & 0 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \\ \Theta \\ \dot{\Theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ -\frac{1}{Ml} \end{bmatrix} f$$

Kimenőjelnek tekinthetjük mind az x, mind a Θ változókat, a bemenőjel pedig az f erő. Az állapotegyenlet alapján felírható a rendszer átviteli függvénye mindkét változóra:

$$H_1(s) = \frac{\Theta(s)}{f(s)} = -\frac{1/Ml}{(s+\alpha)(s-\alpha)}$$
$$H_2(s) = \frac{x(s)}{f(s)} = \frac{\frac{1}{M}(s+\beta)(s-\beta)}{(s+\alpha)(s-\alpha)}$$

ahol $\alpha^2 = \frac{g}{l}\frac{M+m}{M}$, $\beta^2 = \frac{g}{l}$

Látható, hogy a linearizált rendszer átviteli függvénye tartalmaz egy labilis pólust, vagyis a rendszer instabil, ezen kívül H_2 számlálójában megjelenik egy nem minimumfázisú zérus is. A rendszer szabályzását ez a két dolog nehezíti.

A szabályzáshoz szükség van arra, hogy mérjük az állapotváltozókat. Mérhető például a Θ szögelfordulás és szögelfordulás sebessége, a $\dot{\Theta}$, valamint a másik két állapotváltozó is. A

megvalósítás során a robotra épített giroszenzor segítségével pont ezt a két mennyiséget mérjük. Az állapotváltozók értékét a visszacsatoló ágba rakott szabályzó kapja:

$$f = -\hat{K}\vec{x} \ ,$$

ahol $\hat{K}$ a visszacsatoló mátrix, $\vec{x}$ pedig az állapotváltozókból álló vektor, ezek időbeli ismerete szükséges ahhoz, hogy megfelelő szabályzást tudjunk megvalósítani. Redukáljuk a rendszert úgy, hogy csak két állapotváltozót tekintünk (Θ , $\dot{\Theta}$) és tegyük fel, hogy a szabályozott jellemző a gyorsulás [15]. Erre a redukált rendszerre az állapotegyenlet a következőképpen módosul:

$$\frac{d}{dt}\begin{bmatrix}\Theta \\ \dot{\Theta}\end{bmatrix} = \begin{bmatrix}0 & 1 \\ \frac{g}{l}\frac{M+m}{M} & 0\end{bmatrix} + \begin{bmatrix}0 \\ -\frac{1}{Ml}\end{bmatrix} f(t) \ .$$

Az egyenlet $\hat{A}$ mátrixa az eredeti egyenlet állapotmátrixának jobb alsó sarka, és a rendszer karakterisztikus egyenlete egyszerűen $\lambda^2 - \frac{mg}{M} = 0$ egy jobb oldali gyökkel, vagyis a rendszer instabil. A stabilizáláshoz olyan szabályzójelet generálunk, mely függvénye mind a két állapotváltozónak, eszerint:

$$f(t) = -\hat{K}\vec{x} = -\begin{bmatrix}k_1 & k_2\end{bmatrix}\begin{bmatrix}\Theta \\ \dot{\Theta}\end{bmatrix} = -k_1\Theta - k_2\dot{\Theta} \, .$$

Helyettesítsük ezt be az állapotegyenletbe:

$$\begin{bmatrix}\dot{\Theta} \\ \ddot{\Theta}\end{bmatrix} = \begin{bmatrix}0 & 1 \\ \frac{g}{l}\frac{M+m}{M} & 0\end{bmatrix}\begin{bmatrix}\Theta \\ \dot{\Theta}\end{bmatrix} + \begin{bmatrix}0 \\ -\frac{1}{Ml}(k_1\Theta + k_2\dot{\Theta})\end{bmatrix} .$$

A jobb oldalon az additív tagok összekombinálásával a következőhöz jutunk:

$$\begin{bmatrix}\dot{\Theta} \\ \ddot{\Theta}\end{bmatrix} = \begin{bmatrix}0 & 1 \\ -\frac{mg}{M} + \frac{k_1}{Ml} & \frac{k_2}{Ml}\end{bmatrix}\begin{bmatrix}\Theta \\ \dot{\Theta}\end{bmatrix} .$$

A rendszer karakterisztikus egyenlete tehát:

$$\begin{bmatrix} \lambda & -1 \\ \frac{g}{l}\frac{M+m}{M} - \frac{k_1}{Ml} & \lambda - \frac{k_2}{Ml} \end{bmatrix} = \lambda\left(\lambda - \frac{k_2}{Ml}\right) - \frac{1}{Ml}\left[g(M+m) - k_1\right] = \lambda^2 - \frac{k_2}{Ml}\lambda + \frac{1}{Ml}\left[k_1 - g(M+m)\right]$$

A redukált rendszer karakterisztikus polinomjából látható, hogy a rendszer akkor lesz stabil, ha

$$\frac{k_2}{Ml} < 0 \qquad \text{és} \qquad g < \frac{k_1}{M+m} \ ,$$

gyors válaszhoz

$$\frac{k_2}{Ml} = -16 \qquad \text{és} \qquad \frac{k_1}{M+m} = 100$$

a megfelelő választás. Egységugrás próbajelre ilyenkor a túllövés 1.5%, a beállási idő 0.5 sec körüli lesz.

A rendszer szabályzásához hagyományos PID szabályzást használtak, melyben a P (proportional) tag a hibajellel, az I (integral) tag a hibajel integráljával, a D (derivate) tag pedig a hibajel változási sebességével arányos.

Az itt leírt elmélet nem elegendő a két keréken egyensúlyozó robot szabályozásához, de mivel ennek megoldása nem volt feladatunk, ezzel csak olyan mélységben foglalkoztunk, ami a rendszer megértéséhez szükséges volt. Ettől teljesebb leírás az itt [13] letölthető csomag dokumentációjában található.

III. FEJEZET

ROBOTUNK FIZIKAI FELÉPÍTÉSE

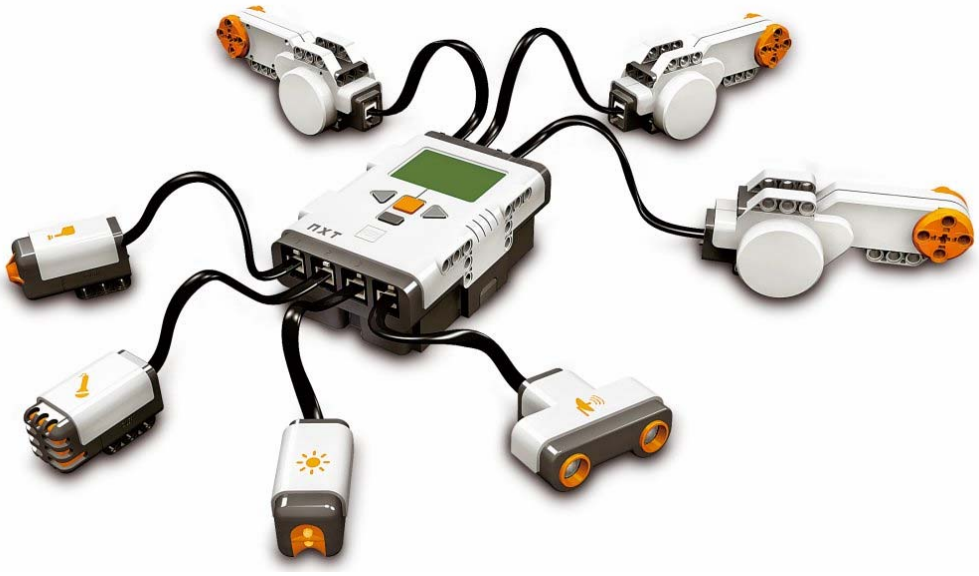
Robotunk alapja, az NXT 2.0

Robotunk alapja az NXT legfrissebb verziója, az NXT 2.0, ami a robot agyának tekinthető. Ez egy intelligens, számítógéppel programozható modul, melyhez három motor és négy szenzor csatlakoztatható egy időben. Rendelkezik grafikus kijelzővel, nyomógombokkal és hangszóróval is. A számítógépen megírt programot USB kábelen keresztül, vagy a Bluetooth csatornán keresztül lehet le- és feltölteni.

11. ábra: Az NXT 2.0

Az NXT 2.0 specifikációja:

- 32-bites AT91SAM7S256 fő mikroprocesszor (256 KB flash memória, 64 KB RAM)
- 8 bites ATmega48 mikrokontrolleres @ 4 MHz-es (4 KB flash memória, 512 bájt RAM)
- 100 × 64 képpontos LCD mátrix kijelző
- Egy USB 2.0 port
- Bluetooth (Class II) vezeték nélküli kapcsolat
- 4 bemeneti port
- 3 kimeneti port
- Digital Wire Interface [13]



12. ábra: Az NXT az alapkészlethez tartozó motorokkal és érzékelőkkel [16]

Az NXT készlethez a közismert Technic építőelemeken és a központi egységen kívül tartozik 3 szervomotor és 4 különböző szenzor.

Az NXT alapú robot szervomotorainak segítségével tud bármiféle mozgást végezni, így kerekeken gurulni, járni vagy tárgyakat emelni. A motorok sajátsága, hogy elfordulás-érzékelőt tartalmaznak, így az általuk végrehajtott elfordulás lekérdezhető, illetve viszonylag finom mozgás kivételezésre adódik lehetőség.

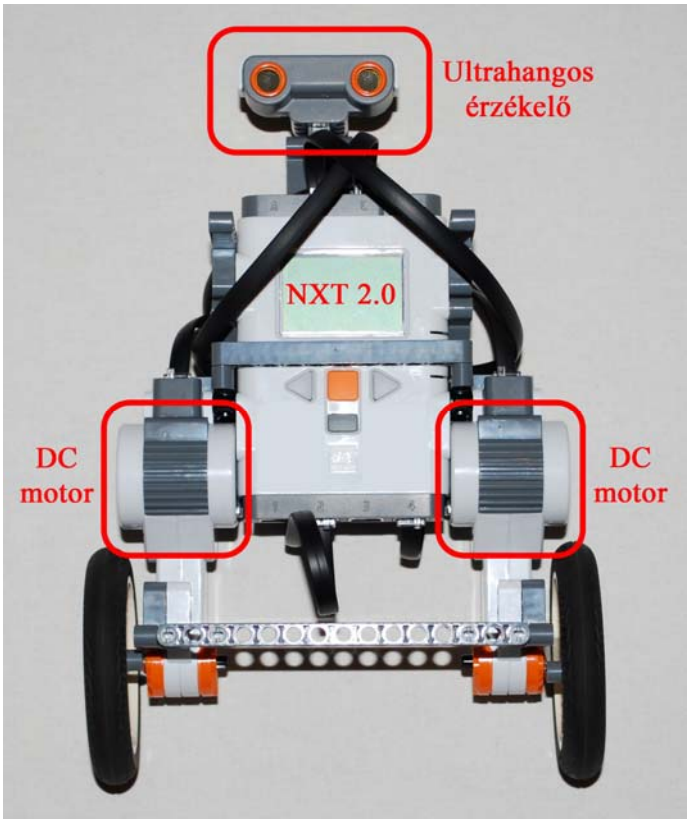
Az alap kiszerelésben a motorokban található elfordulás-érzékelők mellett négy további csatlakoztatható szenzort találunk, ezek:

- LightSensor – fényerősségeket és színeket képes megkülönböztetni egymástól.
- UltrasonicSensor – a szenzor és a céltárgy közötti távolságot képes mérni.
- SoundSensor – hangok érzékelésére szolgál.
- TouchSensor – egy nyomógombot tartalmaz, amelyik megnyomásával tetszőleges feladatra utasíthatjuk a robotot.

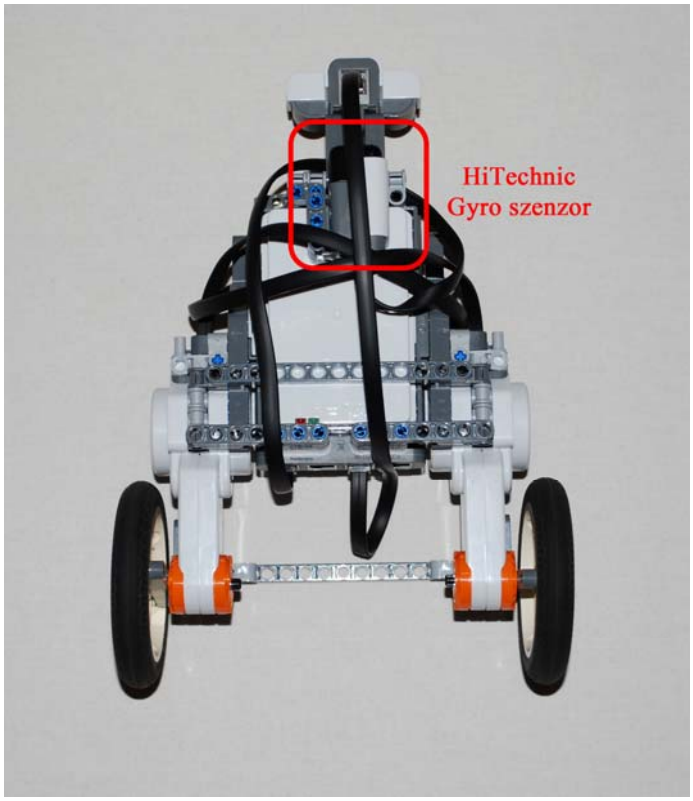
A robot fizikai felépítése

A robot legfontosabb részegységei:

- 1db Mindstorms NXT 2.0
- 1db HiTechnic NXT Gyro szenzor
- 2db DC szervo motor
- 1db ultrahangos érzékelő



13. ábra: A robot felépítése elölnézetből



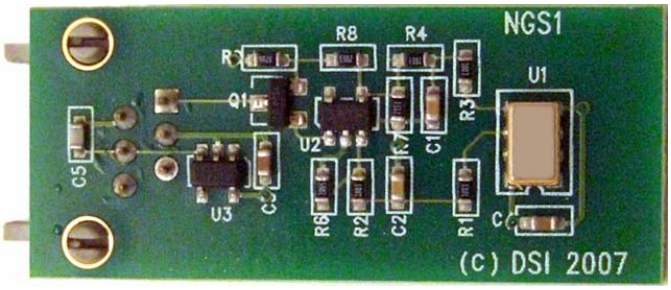
14. ábra: A robot felépítése hátulnézetből

A robot egy LEGO Mindstorms NXT 2.0 készletből és egy külön megvásárolható giroszenzorból épült. Két hajtott kerekékkel rendelkezik, az akadályoknak ütközés megelőzése érdekében ultrahangos távolságmérőt szereltünk a robotra. Ha a beállított távolságon belül akadályt érzékel a robot, tolatni kezd. Mivel a talajjal csak a két kerék érintkezik, a berendezés instabil, a roboton futó program szabályozza a kerekek elfordulását a giroszenzortól kapott elfordulás és szögsebesség adatok alapján az egyensúlyi helyzet megtartása érdekében. A robot összeszerelési útmutatója a szakdolgozathoz mellékelt CD-n található.

A HiTechnic NXT Gyro szenzor

Az HiTechnic NXT Gyro szenzorral a forgási szög változását pontosan tudjuk mérni [17]. A szenzor a forgási iránynak és mértékének megfelelő szögperc értékeket adja vissza, +/- irányban, másodpercenként maximum 360 foknyi eltérést értelmezve. Működési elve megegyezik a Segway-ben használt giroszenzoréval.

15. ábra: A HiTechnic Gyro Sensor

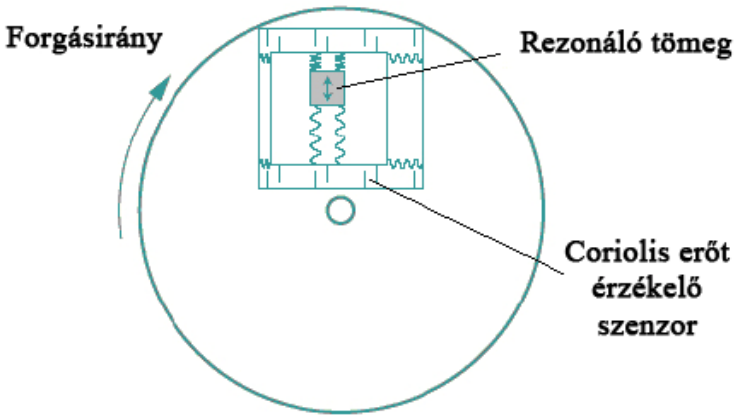


16. ábra: A HiTechnic Gyro Sensor PCB lapkája [18]

A giroszenzor működési elve

A Segway-ben MEMS (Mikro-Elektromechanikai Rendszer) technológiára épülő vibrációs giroszenzorokat alkalmaznak. Ezek a lézeres megoldásoknál sokkal pontatlanabbak, ám nagyságrendekkel kisebbek és olcsóbbak. A mikrogépészet és a Si-mikroelektronika együttes lehetőségeinek kiaknázásával a MEMS technológia meghatározó alaptechnológiává vált. Az érzékelés és vezérlés, beavatkozás egyetlen rendszeren belüli megvalósításával, az IC-k számítási képességét ötvözi a mikroérzékelés és -beavatkozás lehetőségeivel.

A giroszenzor egy rugalmasan rögzített tehetetlen tömegből és a tömeg elmozdulását érzékelni képes kapacitásérzékelőből áll [19]. Működési elve azon alapszik, hogy egy forgó koordináta rendszerben mozgó testre a szögsebességgel arányos Coriolis-erő hat, mely a vonatkoztatási rendszerhez képest forgó és egyben gyorsuló testre ható tehetetlenségi erő. A mozgó testet egy a szilíciumhordozón kialakított rezgő tömeg jelenti, melyre a köré épített test elfordulása esetén a szögsebességgel arányos, ám a rezgés irányára merőleges erő hat. Az erő hatására a lapkához rögzített tömeg a rezgés irányára merőlegesen mozdul el, mely kitérés kapacitív módon könnyen mérhető.



17. ábra: A MEMS giroszenzor működési elve

A DC szervomotor

A motorok beépített elfordulás érzékelővel rendelkeznek, ami lehetővé teszi a motorok precíz irányítását. Fokokban vagy teljes fordulatokban mér, pozitív és negatív irányban. Az egyes motoroknak különböző fordulatszámot is adhatunk, így a robot képes haladási irányt változtatni.

18. ábra: Az NXT készlethez tartozó motor

Az ultrahangos érzékelő

Az ultrahangos érzékelő segítségével a robot észlelni tudja az előtte lévő tárgyakat, így képes arra, hogy elkerülje az akadályoknak ütközést.
Működésének elve: ultrahang-hullámokat bocsát ki, ami az előtte lévő akadályról visszaverődik a szenzor felé. A kibocsátás és visszaérkezés közötti időtartamot figyelve megállapítja a közte és az akadály közötti távolságot.

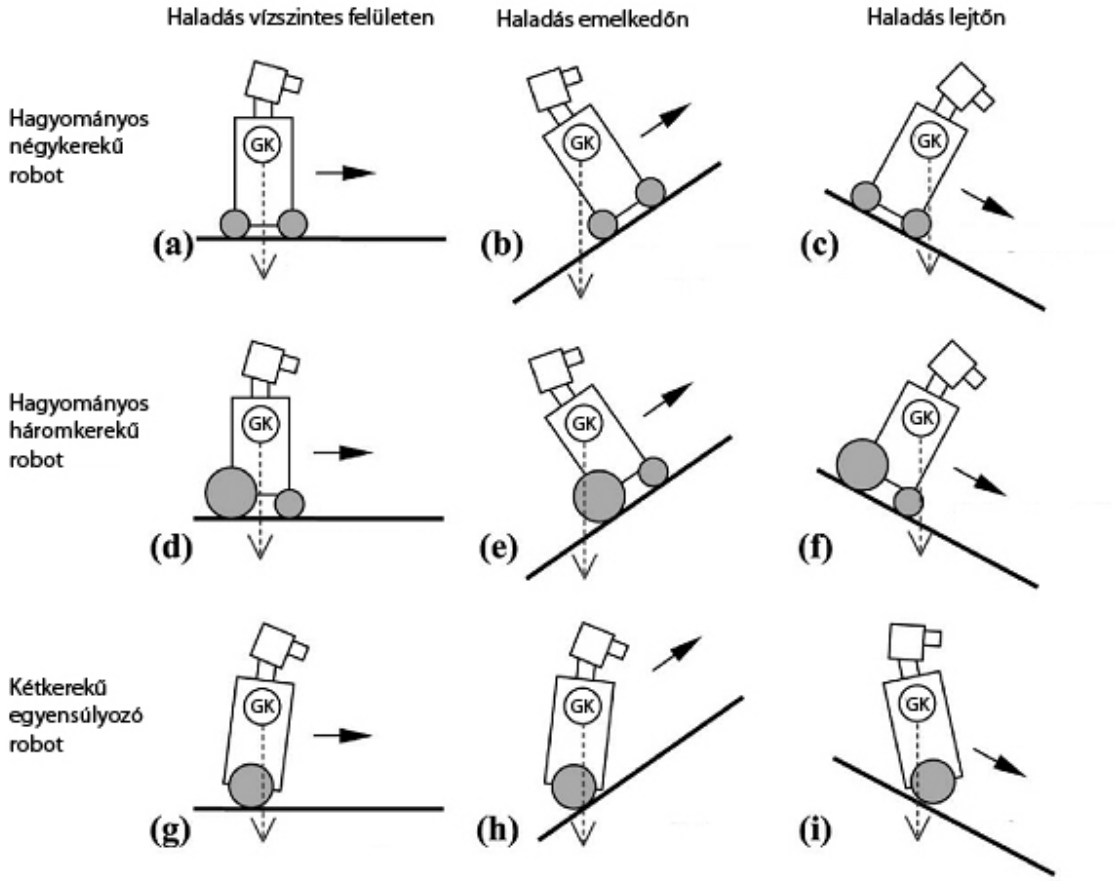
19. ábra: Az NXT készlethez tartozó ultrahangos érzékelő

A robot fizikai felépítéséből származó előnyök

A robot alakja hasonlítható az emberéhez, így szociálisan könnyebben elfogadható az emberek számára. Ez akkor lehet fontos, ha később emberek között fog működni. Kis méretéből és nyomtávjából adódóan szűk helyeken is könnyedén navigálható.
Két hajtott kereke van, ezáltal mozgékonyabb, mintha több kerékkel rendelkezne. A két kerék ellentétes irányba történő forgatásával helyben képes megfordulni.

20. ábra: A robot felépítéséből származó előnyök

Az egyik legfontosabb előny a több kerekű robotokkal szemben az, hogy folyamatosan változtatható a gravitációs középpont helyzete a kerekek felett. Ez lehetővé teszi a robot számára, hogy az emelkedőket és lejtőket is könnyen teljesíthesse, miközben biztosítja a szenzorok számára, hogy megközelítőleg ugyanolyan pozícióban legyenek.



A kétkerekű egyensúlyozó robot stabilabban teljesíti az emelkedőket és a lejtőket a hagyományos többkerekű robotokhoz képest, mert a gravitációs középpontjának helyzetét szabadon tudja váltogatni a tengelye fölött.

21. ábra: A kétkerekű robotok különböző dőlésszögű felületeken mutatkozó előnye a többkerekű robotokkal szemben [20]

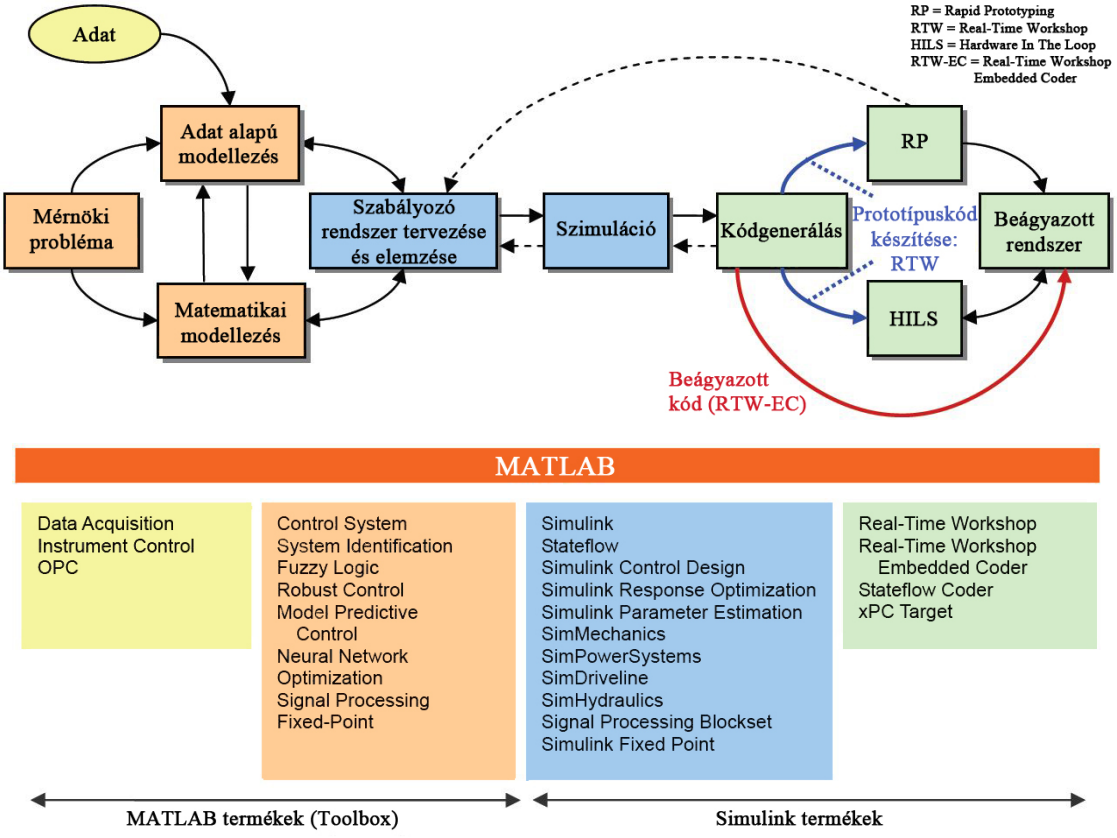
IV. FEJEZET

A ROBOT SZIMULÁCIÓJA ÉS ANNAK FEJLESZTÉSE

Modell alapú programtervezés

Napjainkban egyre több új programtervezési módszertan lát napvilágot. Ezeknek a módszereknek a legfontosabb célja, hogy csökkentsék az egyre komplexebbé váló szoftverek előállításának költségeit, valamint a korábban befektetett munkát újrafelhasználhatóvá tegyék. Az egyik legdinamikusabban fejlődő, és legtöbbet kutatott módszertan a modell alapú programtervezés.

Néhány program lehetővé teszi számunkra, hogy egy szabályozási modellt elkészítve automatikusan C/C++ kódot generáljunk vele a beágyazott rendszerünkhöz. A MATLAB környezetben is adott ez a lehetőség, az RTW-EC végzi a kódgenerálást. Az alábbi képen a MATLAB termékek segítségével készíthető szabályozási rendszerek modell alapú általános tervezési elvét láthatjuk.



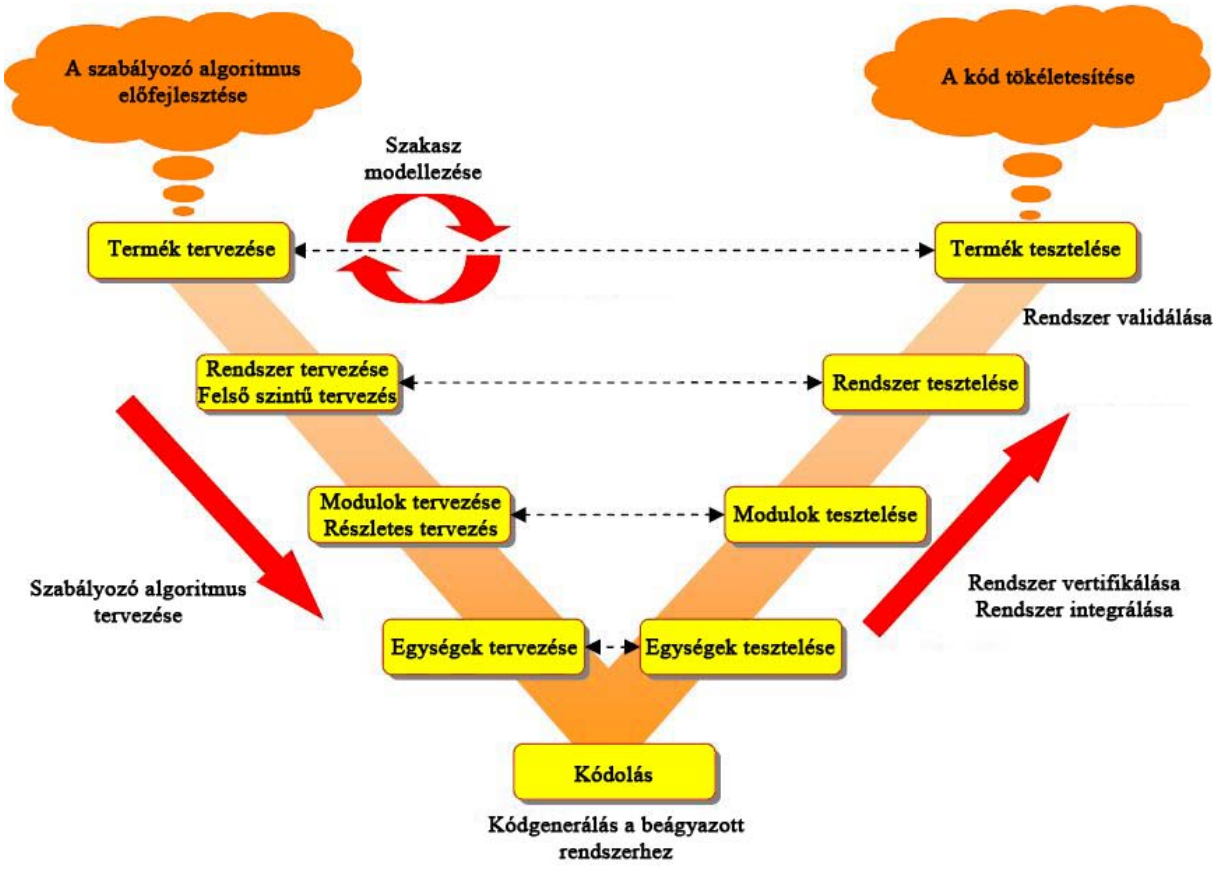
22. ábra: Szabályozási rendszerek modell alapú tervezése a MATLAB termékcsalád segítségével [13]

V-modell

Szabályozási rendszerek és szimulációs modellek modell alapú programozásakor a V-modellt használják. A modell jól kifejezi a tervezési folyamat fentről lefele történő haladását (top-down megközelítés), míg a tesztelési folyamat lentről felfelé halad (buttom-up megközelítés). A gyakorlatban a különböző fázisok nem feltétlenül a megadott sorrendben hajtódnak végre.

A szabályozó rendszerekhez a tervező egy szakasz egységet és egy szabályozót, vagy azoknak egy részét modellezi, és a szabályozó algoritmust számítógép- vagy valós időn alapuló szimulációkkal teszteli. A valós idejű szimuláció lehetővé teszi számunkra, hogy közvetlenül ellenőrizzük és jóváhagyjuk az algoritmust.

A V-modellt a 23. ábra mutatja, képet adva a szabályozó rendszerek modell alapú tervezésének folyamatáról. A V-modell magába foglalja a tervezés, kódolás és tesztelés állapotokat. Minden egyes teszt állomás összhangban van a megfelelő tervezés állomással.



23. ábra: A V-modell [13]

A modell-alapú tervezés érdemei:

- Nyomon követhetők a jellemző hibák a fejlesztés korai szakaszaiban
- Prototípus hardver számának csökkentése, és hibamentességének megerősítése valós idejű szimulációval
- Hatékony tesztelés modell alapú ellenőrzéssel
- Kódolási idő és hiba csökkentése automatikus kódgenerálással

Szimuláció

Az NXTway-GS projekt keretein belül Yorishima Yamamoto a kétkerekű, HiTechnic Gyro Sensor alkalmazásával önállóan egyensúlyozó robotjához egy Simulinkben futtatható szimulációt is készített a robothoz az Embedded Coder Robot NXT Demo [21] segítségével, amit Takashi Chikamasa fejlesztett 2006-ban.

A Demo három példarobotot is tartalmaz, ezen kívül lehetőséget ad arra, hogy a 3D megjelenítőben bmp kiterjesztésű képfájlként bármilyen pályát rajzolhassunk a robotunkhoz.

Az NXTway-GS-hez készült szimuláció [22] kiválóan bemutatja a robot egyensúlyozó képességét.

Az NXTway-GS szimuláció telepítése

Az ECRobot egy Simulink alapú modellező és kódgeneráló környezet a LEGO Mindstorms NXT-hez. Emellett szükséges még néhány program a működéshez:

- Cygwin – Unix-szerű környezet Windows alá
- GNU ARM Compiler – C fordító az ARM processzorhoz az NXT-ben
- LEGO Mindstorms driver – Meghajtó program az NXT-vel való kommunikációhoz USB kábelen keresztül
- NeXTTool – PC-ről az NXT-re való fájlátvitelhez szükséges
- ECRobot – Az „Embedded Coder Robot”
- nxtOSEK – Valós idejű operációs rendszer a LEGO Mindstorms NXT-hez

Rendszer követelmények:

- 32-bites vagy 64-bites Windows XP, Vista vagy Windows 7
- 32-bites MATLAB R2007a vagy későbbi

Ezen programok telepítésére készített egy automatizált, részletes angol nyelvű telepítőt Gautam Vallabha, amely elérhető a Mathworks honlapján [23].
A telepítő alapvetően három MATLAB script állományból áll:

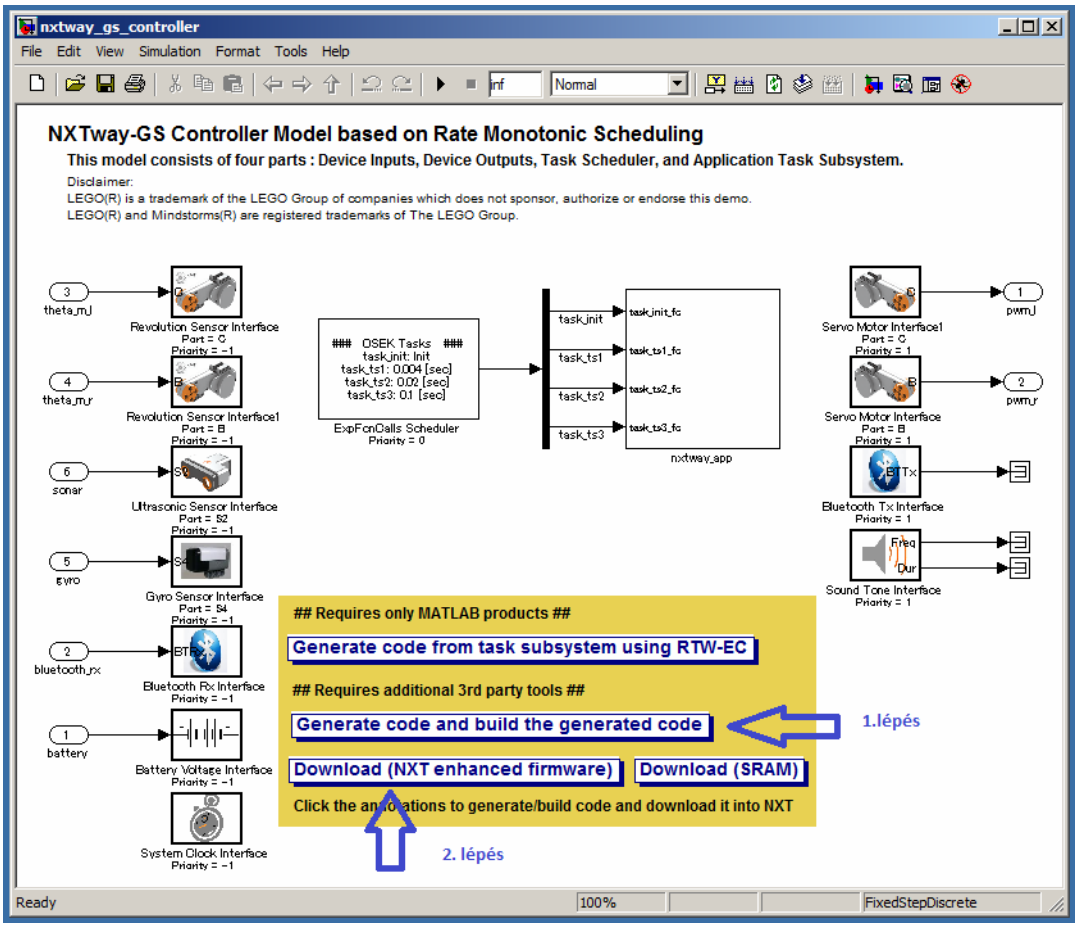
- download_ecrobot_tools.m – a komponensek letöltésére szolgál
- install_ecrobot_tools.m – a komponensek telepítését és konfigurálását végzi
- update_nxt_firmware.m – egy grafikus felület segítségével segíti az NXT gyári firmware cseréjét

Ez a telepítő annyi kiegészítésre szorul, hogy az automatizált telepítés végén le kell cserélni (telepítés után felülírni) a cygwin-t az általunk mellékeltre, valamint a libusb0.dll-t kell a „C:\Windows\system32” mappába helyeznünk a hibátlan működés érdekében. A telepítés maximum 10-15 percet vesz igénybe az internet kapcsolat sebességétől függően. A telepítés hibátlan végbemenetelét és a rendszer újraindítását követően számítógépünk teljesen alkalmassá válik a szimuláció futtatására és a LEGO Mindstorms NXT kezelésére.

NXTway-GS feltöltése a robotra

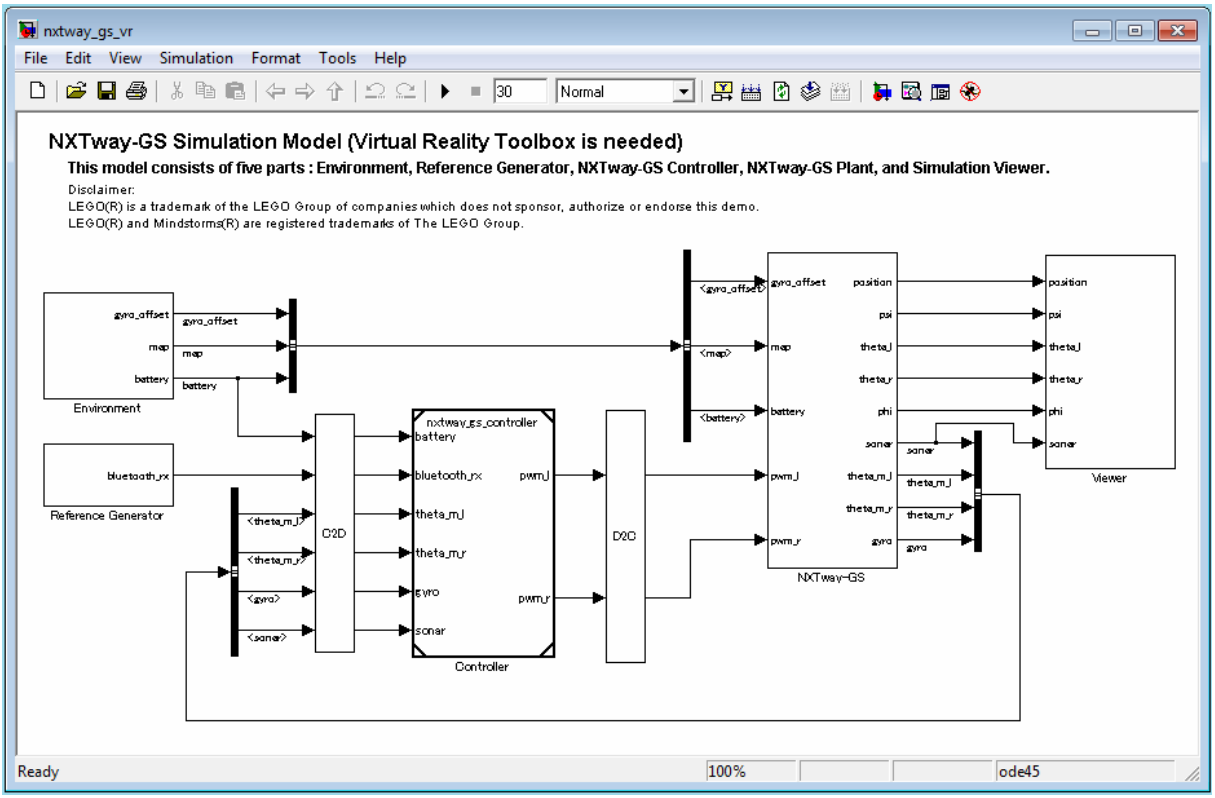
Nyissuk meg (MATLAB-ból) az nxtway_gs_controller.mdl állományunkat, majd a 24. ábra alapján végezzük el a következő lépéseket.

1. Generáljuk és fordítsuk a kódot a „Generate code and build the generated code” gomb megnyomásával.
2. Kapcsoljuk össze USB kábellel a PC-t az NXT-vel, majd nyomjuk meg a „Download (NXT enhanced firmware)” gombot.



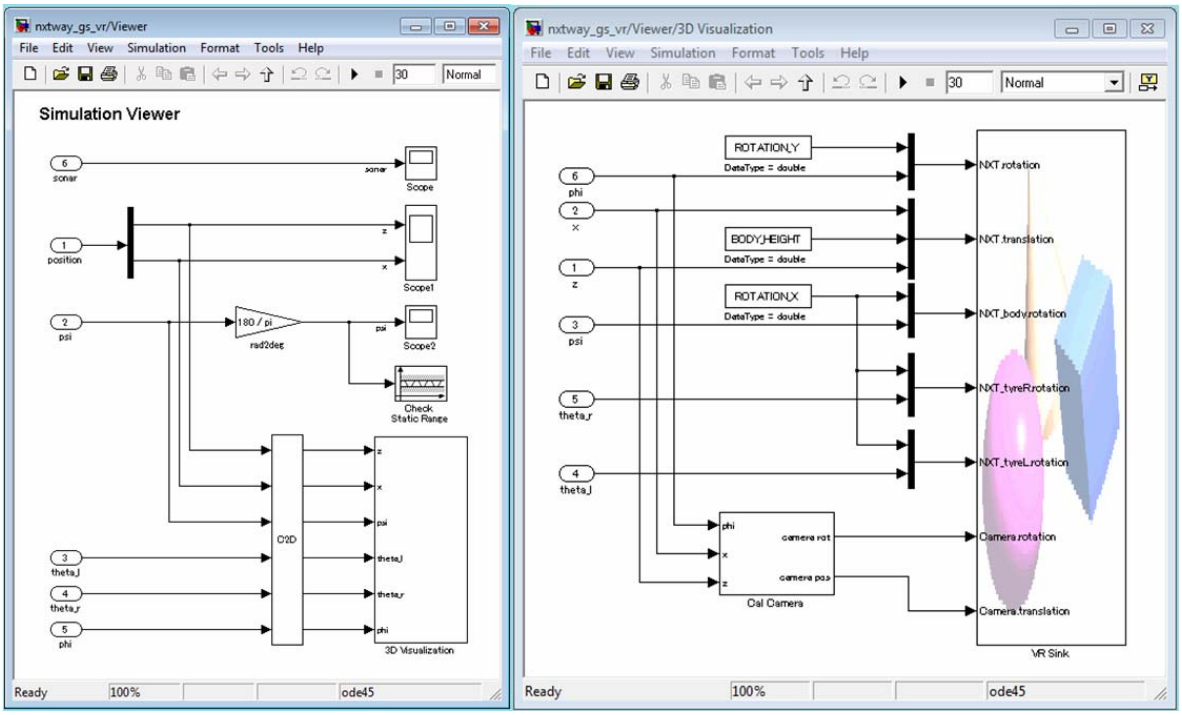
24. ábra: A szoftver NXT-re való feltöltéséhez szükséges lépések

Ezt követően az NXTway-GS szimuláció modelljét kell indítanunk az NXTway-GS fájlcsomagjából a nxtway_gs/models/nxtway_gs_vr.mdl fájlt kiválasztva. A Simulinkes szimuláció modelljét a 25. ábra mutatja.



25. ábra: Yorishima Yamamoto szimulációjának modellje

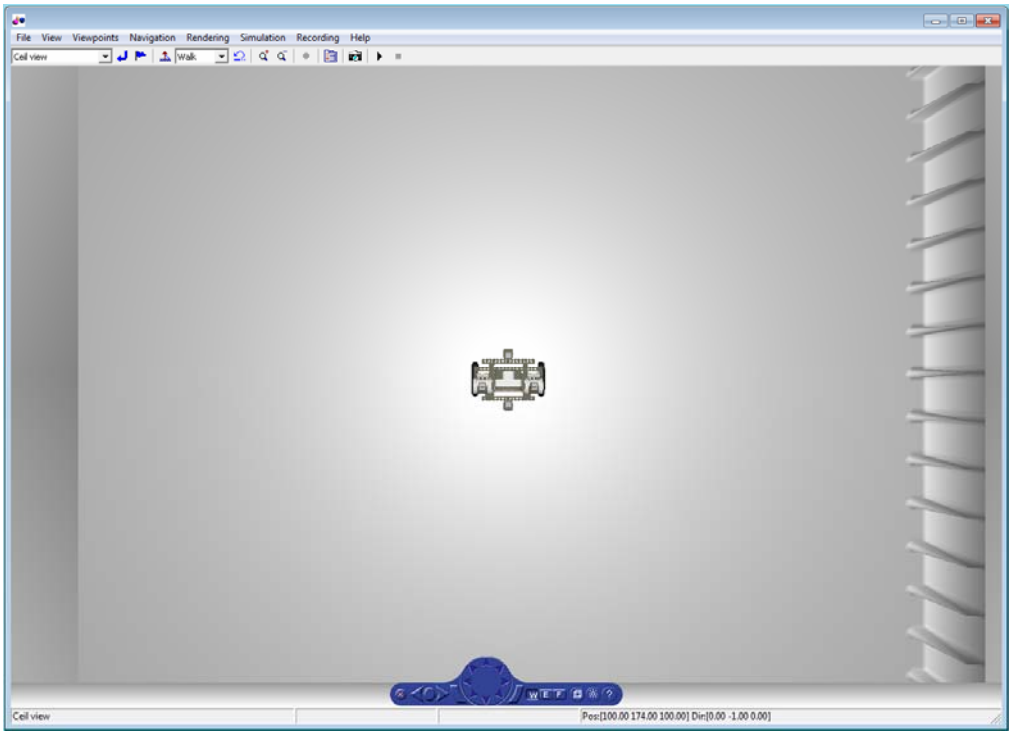
A szimuláció elindításához a Viewer blokkon belül a 3D Visualisation, majd a VR Sink blokkot kell megnyitni:



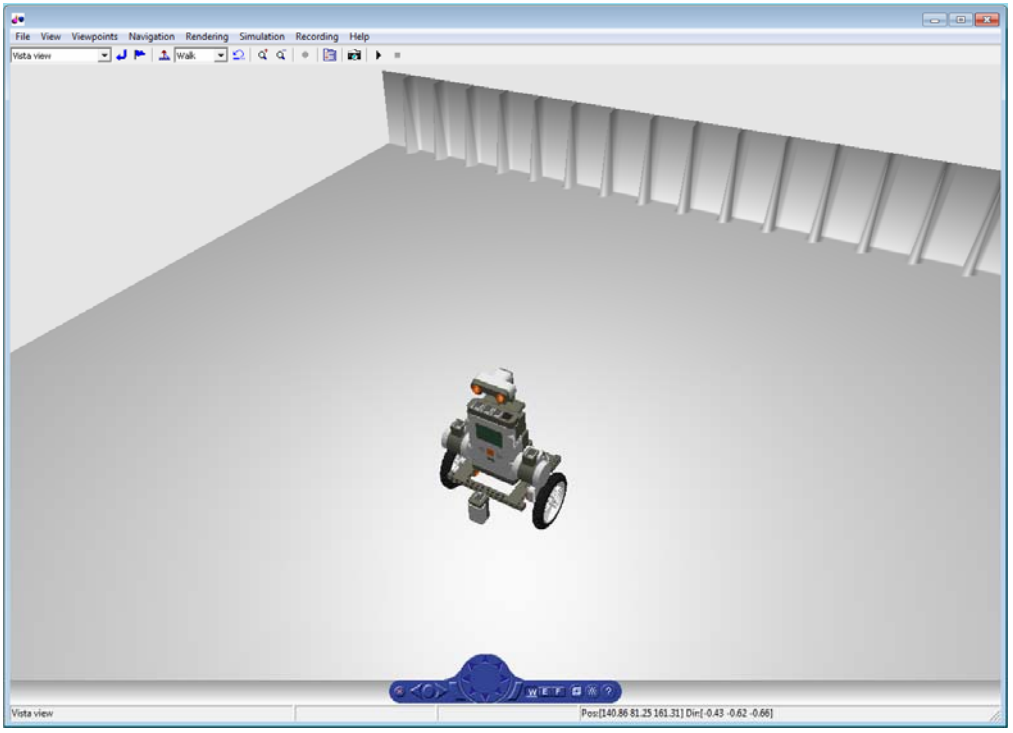
26. ábra: A szimuláció indításához a „VR Sink” blokkot kell megnyitni

Így láthatjuk a robot szimulációját, és el is indíthatjuk azt.

A szimulációt elindítva különböző nézetek közül választhatunk, közelíthetünk, távolíthatunk vagy oldal irányban elforgathatjuk a nézetet. Néhány kép a futó szimulációról:

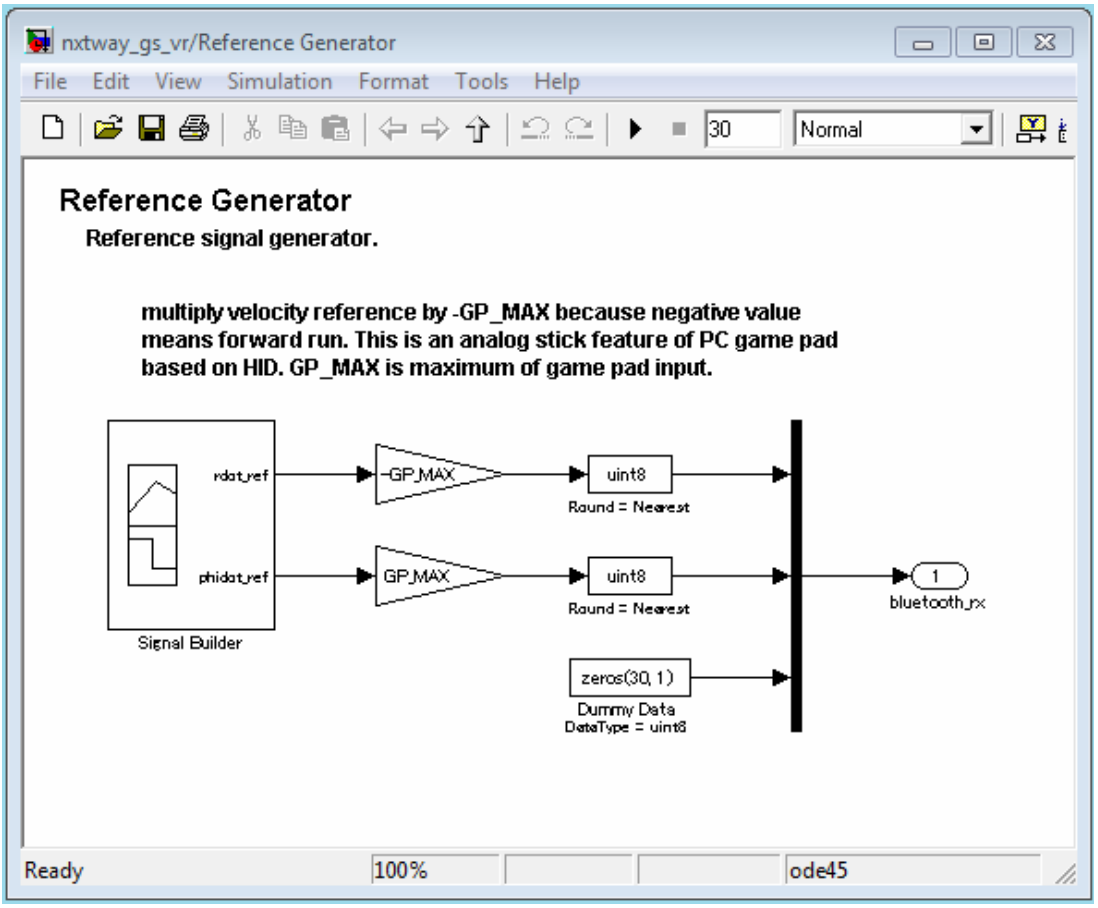


27. ábra: Kép a futó szimulációról 1



28. ábra: Kép a futó szimulációról 2

A szimuláció Reference Generator blokkjában lehetőségünk nyílik a robot néhány előre meghatározott mozgásának indítására.



29. ábra: A Reference Generator blokk

A Signak Builder blokkon belül a szimulált robotnak négy referenciajel közül adhatunk egyet egy időben, majd a szimulációt lefuttatva ez a jel fogja a robot mozgását meghatározni.

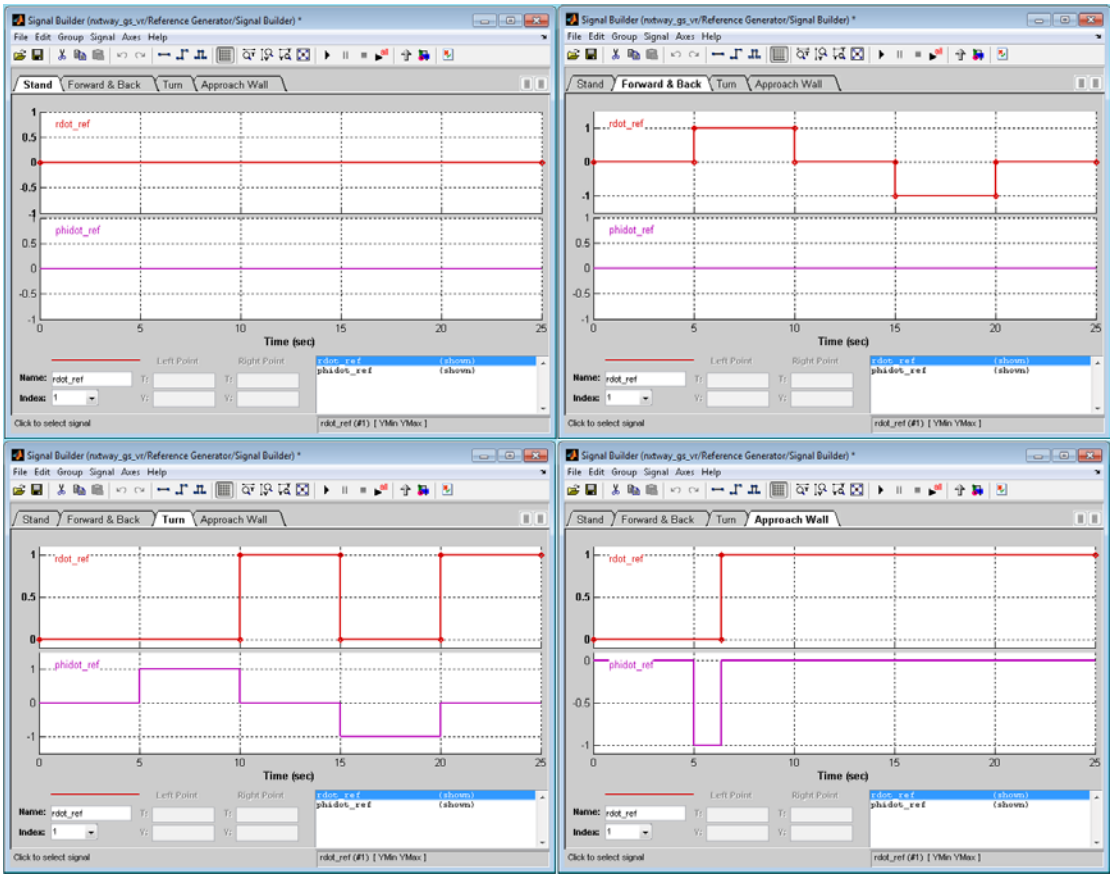
Ezek sorra a következők:

Stand – a robot egy helyben egyensúlyozik.

Forward & Back – a robot előre elindul, majd megáll, végül tolat.

Turn – a robot először egyik irányba, majd a másik irányba forog.

Approach Wall – a robot megközelíti a falat.



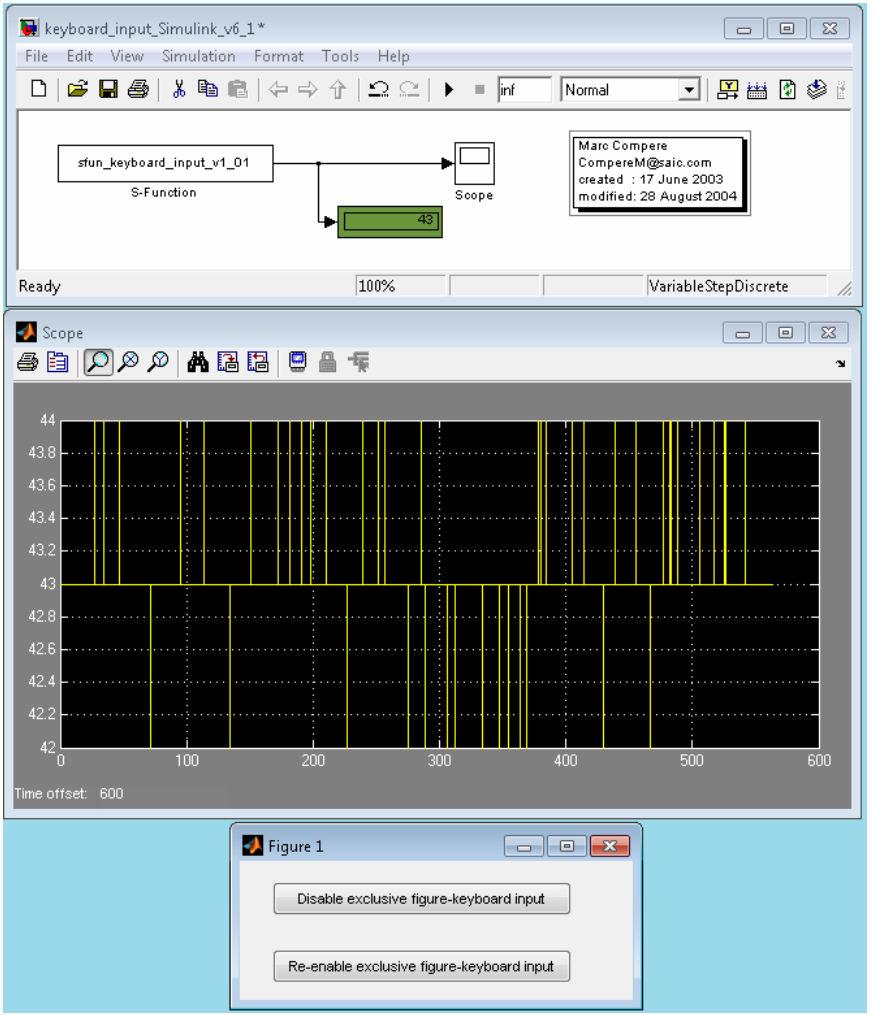
30. ábra: Az eredeti szimulációnak adható referenciajelek

Egy referenciajel futtatásakor kétfajta jel továbbítódik, az egyik az rdot_ref, a másik a phidot_ref. Ezeknek az értékei -1,0,1 lehet, amelyek a valódi robotot irányító Gamepad analóg karjainak szélsőértékeit reprezentálják.

31. ábra: A szimulált Gamepad parancsok szemléltetése

A szimuláció irányítása

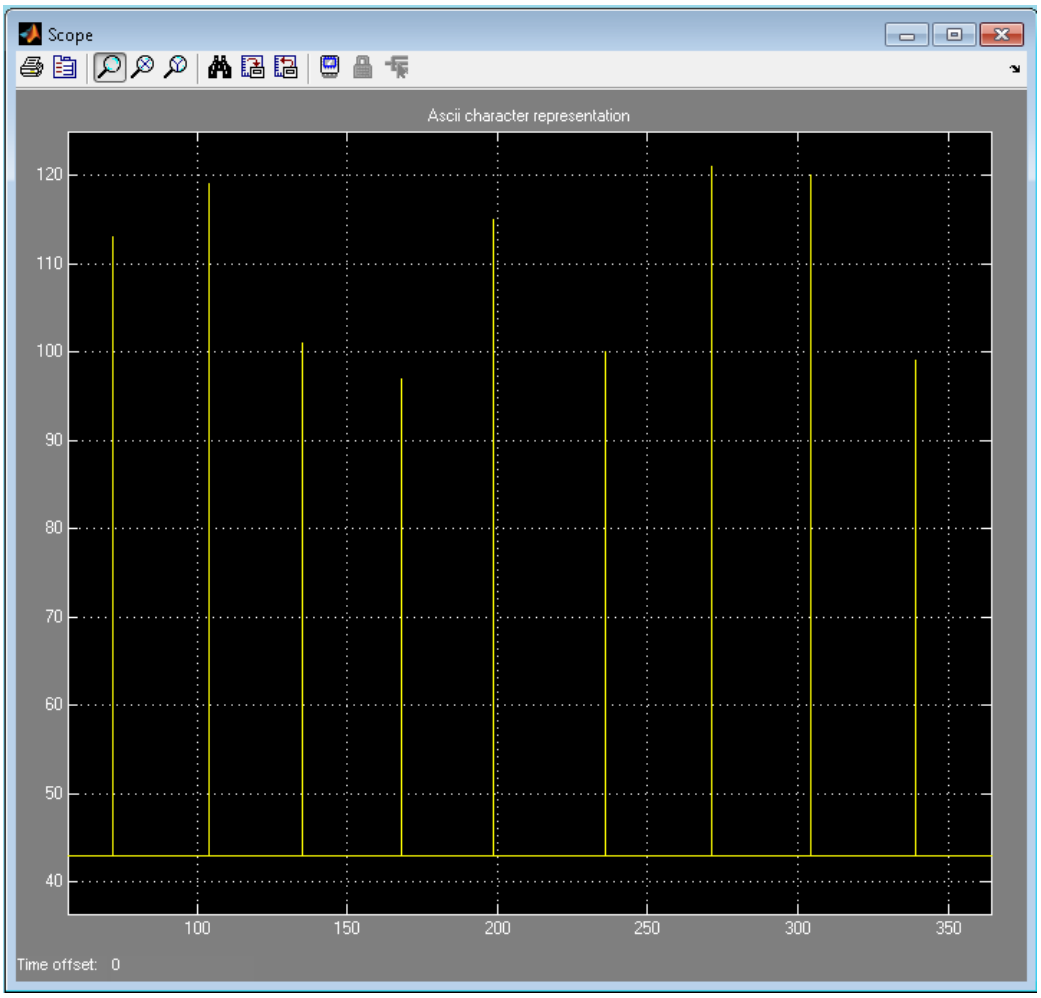
A szakdolgozat egyik kitűzött célja a robot-szimuláció irányításának megvalósítása. Alapvetően korlátozott a szimulált robot mozgása, egy időben a négy alapmozgás közül csak egyet választhatunk az eredeti szimulációban, ezeknek az elindítása nehézkes és nem kombinálható. A szimuláció számítógép billentyűzetével történő irányításának legegyszerűbb módja, ha az egyes billentyűk ASCII kódját használjuk fel. A Mathworks honlapján található egy billentyűleütéseket ASCII kódokká konvertáló program Simulink Keyboard Input [24] megnevezéssel, melyet Marc Compere készített 2004-ben. Ez az S-Function arra készült, hogy billentyűparancsokat tudjon fogadni egy futó Simulink szimuláció közben. Mivel a Windows operációs rendszerekben mindig csak az aktív ablak kapja a billentyűparancsokat, ezért egy segédablak jelenik meg a program indításakor, ez lesz az az aktív ablak, amelyik segítségével a program érzékeli a billentyűparancsokat.



32. ábra: A billentyűleütéseket értelmező program, működés közben

Alapértelmezett értékként a 43-as értéket továbbítja (a + jel ASCII kódja). A program a zöld kijelzőben megjeleníti az aktuálisan leütött billentyűk ASCII kódját, ez azonban a mai korszerű számítógépeken rendkívül gyorsan, szinte láthatatlanul történik, ezért a program rendelkezik egy Scope grafikonnal is, melyen a program indításától kezdődően minden leütött billentyűt láthatunk. A program működését néhány billentyűt véletlenszerűen leütve a 32. ábra illusztrálja.

A szimuláció irányítására a q,w,e,a,s,d,y,x,c billentyűket jelöltük ki. Ezen billentyűknek az ASCII kódját a fenti programmal ellenőriztük le, a program által adott kódokat használtuk fel a továbbiakban. A billentyűkhöz tartozó ASCII kódokhoz a következő grafikont kaptuk:

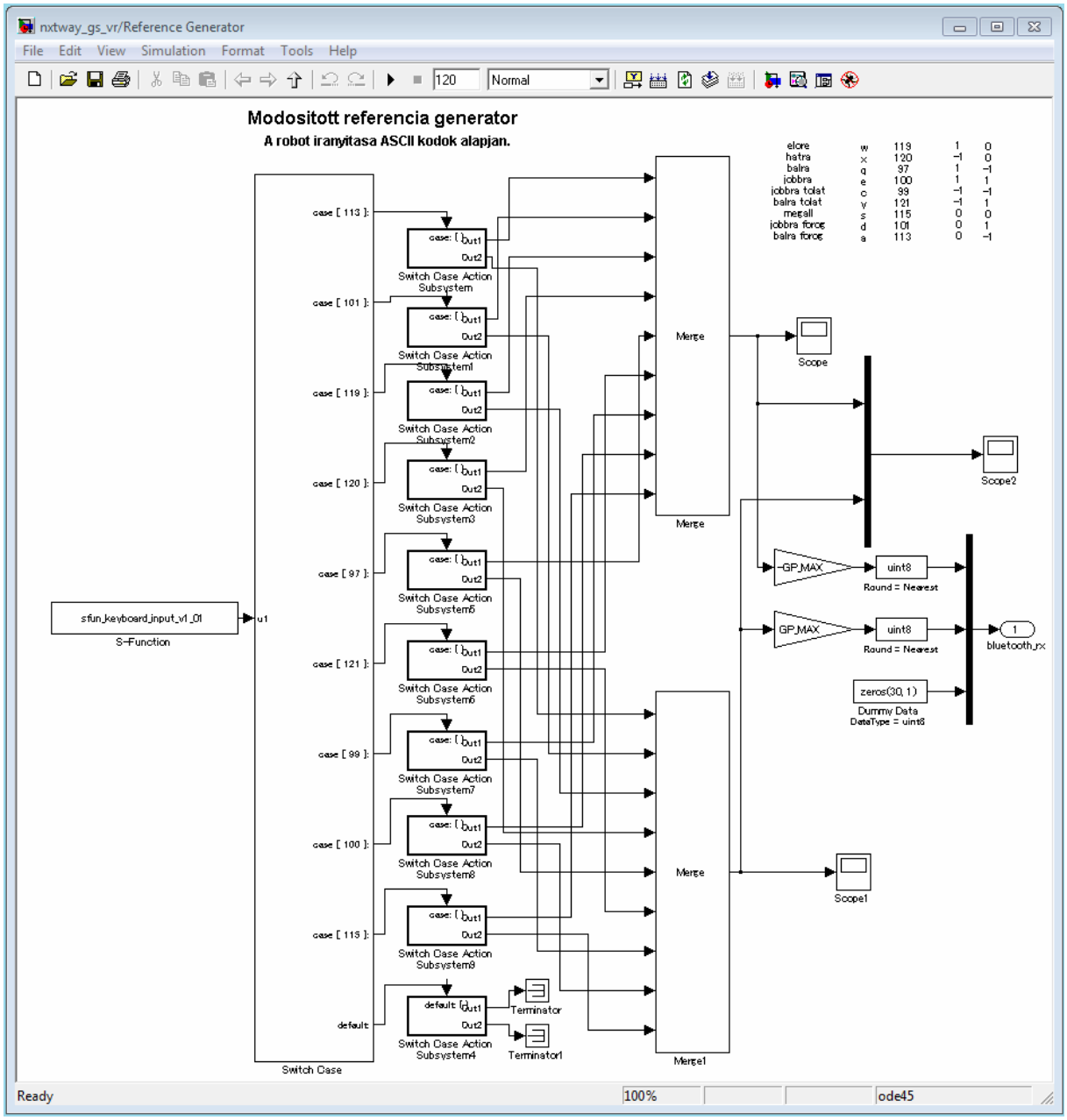


33. ábra: A grafikon a következő billentyűk ASCII kódjait tartalmazza: q,w,e,a,s,d,y,x,c

A billentyűparancsokhoz rendelt értékek és funkciók a következő táblázatban láthatók:

Billentyű	ASCII kód	-GP_MAX	GP_MAX	Szimuláció irányításában betöltött feladat
w	119	1	0	Előre haladás.
x	120	-1	0	Hátra haladás.
q	97	1	-1	Balra fordulás.
e	100	1	1	Jobbra fordulás.
a	113	0	-1	Balra forgás egy helyben.
d	101	0	1	Jobbra forgás egy helyben.
y	121	-1	1	Balra tolatás.
c	99	-1	-1	Jobbra tolatás.
s	115	0	0	Megáll.

Mivel a Signal Builder block határozza meg a robot mozgását, ezért az irányításhoz a Reference Generator blokkban ezt kell felcserélni a billentyű leütéseket értelmező blokkal, és az ASCII kódokat megfelelő –GP_MAX és GP_MAX jelekké alakító blokkokal. A módosított Reference Generator így a következőképpen néz ki:



34. ábra: A módosított Reference Generator

Ezt a módosítást elvégezve a szimulált robotot szabadon irányíthatjuk.

V. FEJEZET

AZ NXT-N HASZNÁLHATÓ PROGRAMOZÁSI NYELVEK ÉS A ROBOT MOZGÁSÁT VEZÉRLŐ PROGRAM

A robot programozása

A robot programozása két irányból közelíthető meg:

- a robot NXT egységébe tölthető le és futtatható egy számítógépen megírt program
- egy számítógépen futó program USB vagy Bluetooth kapcsolaton keresztül közvetlenül irányíthatja a robotot

Az első megközelítés előnye, hogy a robot teljesen önállóan működhet, ugyanakkor csak kisebb méretű és egyszerű programok írhatók meg így. A második megoldás mellett az szól, hogy a kis számítási és memóriakapacitású NXT eszköz helyett a számításokat a nagyobb hardverkörnyezet végzi el. Ekkor viszont a kommunikációs kapcsolatnak kellően gyorsnak és megbízhatónak kell lennie.

Programozási nyelvek

Az NXT programozására számos programozási nyelv alkalmazható [25]. Fontos megjegyezni, hogy a Mindstorms készlet programozása világszerte sokak fantáziáját megmozgatja, ezért a lehetőségek szinte napról napra bővülnek, mindig érdemes újabb, a kitalált feladathoz jól illeszkedő szoftvereket keresni, a használt környezetek legújabb változatát alkalmazni.

Az NXT-n működő programnyelvek

A LEGO Mindstorms NXT Software, az NXT-G

Az első csoportba tartozó legkézenfekvőbb programozási mód a LEGO készlethez adott CD-n található LEGO Mindstorms NXT Software használata. Ez a programozási környezet elsősorban olyan egyéneknek készült, akik még nem programoztak, így a grafikus építőkockákból a robot építéséhez hasonlóan viszonylag gyorsan állíthatják össze és konfigurálhatják az irányító programot. Gyakorlatilag kódírás nélkül tudunk programot

készíteni, így gyerekeknek is ajánlják. A program hátránya, hogy hagyományos programozási nyelvekhez szokott fejlesztőknek eléggé idegen, komolyabb programok már nem igazán áttekinthetők benne. A készülő kód a többi megoldáshoz képest lassan fut és sok memóriát igényel.

NI Labview Toolkit

Az NI Labview adatfolyam programozási nyelvhez kiegészítésként szolgáló grafikus nyelv. Magasabb szintű az NXT-G-nél, több lehetőséget kínál az alapprogram eszközeivel együtt. A Labview fejlesztői környezet „VI" palettáját NXT-hez kapcsolódó eszközökkel bővíti ki, mint például motor, szenzorok és Bluetooth mailbox.

Az NBC/NXC

A következő lehetőség az ingyenesen letölthető NBC/NXC nyelvpáros használata, melyhez szintén ingyenesen a Bricx Command Center biztosít programozási környezetet. A két nyelv közül az NXC a magasabb szintű, C programozási nyelvhez való hasonlóságát és különbözőségét neve is mutatja (Not eXactly C). Az NBC (Next Byte Codes) az NXT egység bájtkódjának felel meg, az NXC programok is erre a nyelvre fordulnak le. Ez a két programozási nyelv gyakorlott programozók számára készült, mélyebb szintű betekintést enged az NXT világába. Az eredményül kapott kód gyorsabb, kisebb méretű, mint az NXT-G programok.
Ugyanakkor a nyelv kifejezőereje még eléggé korlátos, azonban készült olyan változata, amely tömbök esetében a változóval címzést is engedélyezi, ami egy ciklus megírásához fontos. Ezen kívül csak az egész számokat ismeri, így racionális számok kezelésére kerülő megoldásként például a százszoros érték alkalmazását használhatjuk.
A programnak szinte havonta jön ki új változata, ezért középtávon ezek a problémák is valószínűleg meg fognak oldódni.

A RobotC

Érdekes lehet még a robotikában komoly hírnévnek örvendő Carnegie Mellon egyetem robotakadémiája által fejlesztett RobotC nyelv, mely többek között a Mindstorms készlettel is használható. Bár ez a környezet nem ingyenes, létezik egyhónapos próbaváltozata is.

A RobotC firmware-e eltér a gyáritól, így használata előtt firmware frissítés szükséges. A nyelv C-re épül, de az NXC-vel ellentétben a C teljes funkcionalitását biztosítja és ehhez ad ipari szabványokban megszokott C támogatást.

A LeJOS

A LeJOS programozási környezet az NXT operációs rendszerét is lecseréli. Ennek hatására egy kicsi Java virtuális gép jelenik meg a roboton. Ez NXJ API-nak megfelelő egyszerűsített, ugyanakkor NXT-vel kapcsolatos lehetőségekkel bővített Java nyelvű programok futtatását teszi lehetővé. Ez a megoldás sem tekinthető még igazán kiforrottnak, de néhány havonta új verziója jelenik meg.

Az NXT-t távolról irányító megoldások

Az NXT kockát nem csak a saját beépített processzorán futó programok irányíthatják, hanem az USB vagy a Bluetooth csatlakozás túloldalán lévő eszköz, leginkább számítógép vagy mobiltelefon is vezérelheti. Ennek egyszerűbb formája, amikor a vezérlő eszköz távirányítóként működik, vagyis az emberi utasításokat a kapcsolat sebességével viszi át a robotra.
Erre egy mobiltelefonra töltött Javás alkalmazás, a BrickxCC vagy a RobotC is képes. Ennél érdekesebb lehetőség, amikor egy PC-n futó program távirányítja a robotot a Bluetooth kapcsolaton keresztül. Ebből a célból számos ismert programozási nyelvhez készült kiegészítés, így Java-hoz, C#-hoz, Perlhez, Pythonhoz és Rubyhoz is [25].

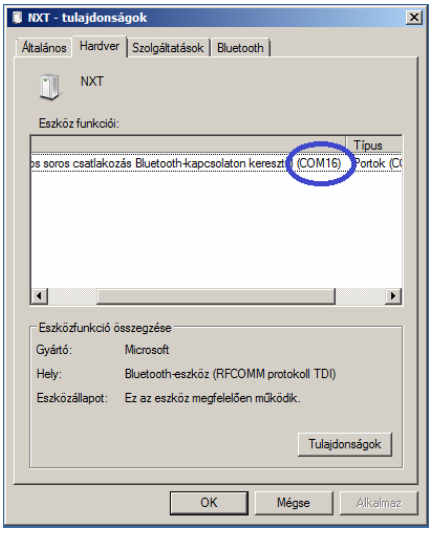
PC – NXT összekapcsolása

Mindenekelőtt párosítsuk az eszközöket (a párosítást úgy végezzük el, hogy a PC kezdeményezze a párosítás folyamatát).

1. Indítsuk el a programot az NXT-n, ahol a következő képet kell látnunk

35. ábra: Az NXT a program indítását követően

2. Keressük ki a párosított Bluetooth eszközök közül az NXT tulajdonságok lapjáról, hogy mely portot (pl. COM16) kapta meg kommunikációra



36. ábra: A kommunikációs Port

3. Az nxtBTC program elindítását követően válasszuk ki a megfelelő portot, majd nyomjunk a „Connect” gombra. Amennyiben sikeresen felépült a kapcsolat ezt mindkét program visszajelzi számunkra.

37. ábra: Sikeres kapcsolódás

4. Ahhoz, hogy egyensúlyozni tudjon a robot, szükségünk van egy „0” értékű pontra, melyet a tervezett egyensúlyi állapotban kell rögzítenünk. Ez lesz a mozgás mérésekor a viszonyítási pont. Ehhez a robotot egyensúlyi helyzetben kell tartani és a RUN gomb megnyomásával el kell indítani az egyensúlyozást, amelynek az indulását sípolással jelzi a robot. Ekkor elengedhetjük.

5. Ezt követően lehetőségünk van a robot szabad irányítására vagy egy korábbi, MATLAB-ban futtatott szimuláció útvonalának megismételtetésére.

Az nxtBTC program bemutatása

Kitűzött céljaink között szerepelt a robot számítógéppel történő egyszerű irányítása valamint, hogy a szimulációban bejárt útvonalat vissza tudjuk játszani az általunk épített robottal. Ezekre a problémákra nyújt megoldást az nxtBTC (nxt Bluetooth Controller) nevű általunk fejlesztett program amit a következőkben szeretnénk bemutatni.

A fejlesztés hardveres és szoftveres környezete:

Hardver:

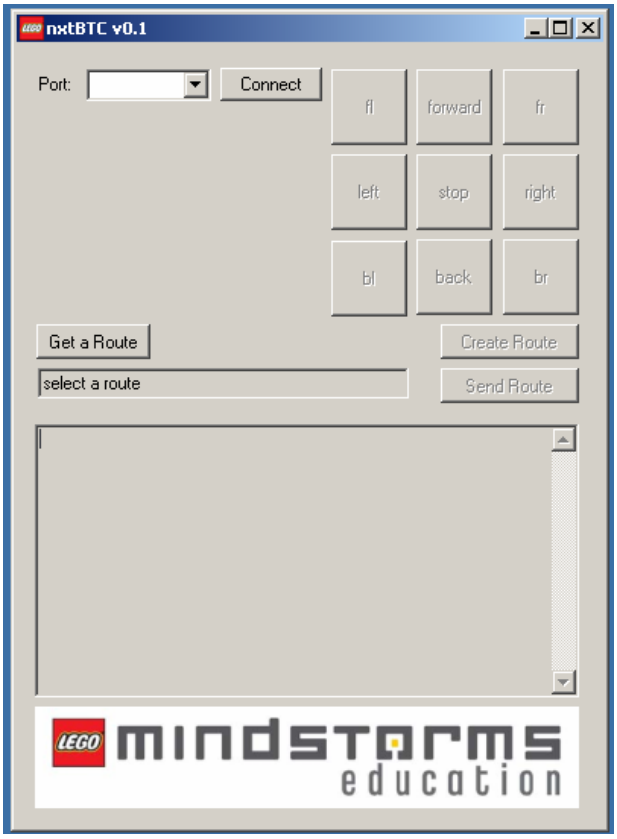
- Processzor: Intel T9300
- Memória: 3 GB
- Winchester: 250 GB

Szoftver:

- Operációs rendszer: Windows 7
- Fejlesztő eszköz: Visual Studio 2010 Professional
- Programozási nyelv: C#

A program funkciói:

- kapcsolattarás az NXT-vel (Bluetooth-on keresztül)
- a robot távirányítása
- a MATLAB-os szimulációtól érkezett útvonal állomány (teszt.xls) feldolgozása és átjátszása az NXT-re
- folyamatos információközlés az eseményekről



38. ábra: Az nxtBTC kezelőfelülete

Kapcsolattartás

Az NXT támogatja a Bluetooth kommunikációt a roboton futó program és egy másik Bluetooth eszköz között. A másik eszköz jelen esetben a számítógép de lehetne akár egy mobiltelefon vagy egy másik NXT is. A fejlesztés során megpróbáltuk kihasználni az NXT által kínált lehetőségeket. Az SPP (Serial Port Profile) segítségével sikerült C# alatt úgy létrehoznunk a Bluetooth kapcsolatot mintha az egy soros port lenne jelentősen leegyszerűsítve ezzel a munkánkat.

```
//csatlakozás
SerialPort BluetoothConnection= new SerialPort();
BluetoothConnection.PortName = „COM16”;
BluetoothConnection.Open();

// kapcsolat bontása
BluetoothConnection.Close();
```

A program egy legördülő menüben jeleníti meg a lehetséges portoknak a nevét, megkönnyítve a választást.

```
//részlet a menü készítéséből
string[] ports = SerialPort.GetPortNames();
foreach (string port in ports)
{
  if(port.Length>5)
    comboBox1.Items.Add(port.Substring(0,5));
  else
    comboBox1.Items.Add(port);
}
```

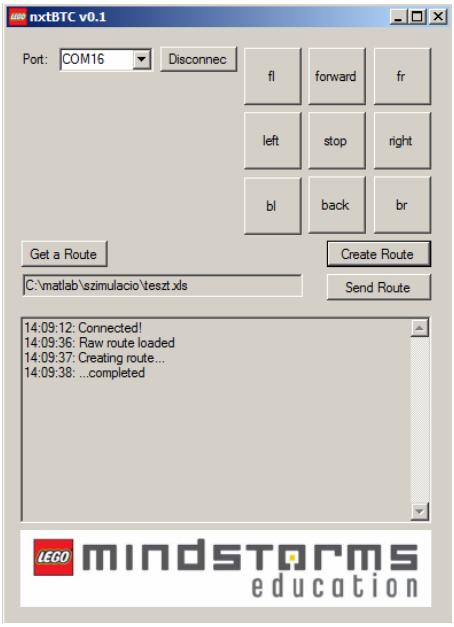
Távirányítás

A kommunikáció során az NXT egy bájt tömböt fogad a PC-től amely tartalmazza az irányítással kapcsolatos adatokat.

```
//adatok küldése visszaolvasás nélkül
 private void NXTSendCommand(byte[] Command)
{
        byte[] MessageLength = { 0x00, 0x00 };
        MessageLength[0] = (byte)Command.Length;
        BluetoothConnection.Write(MessageLength, 0, MessageLength.Length);
        BluetoothConnection.Write(Command, 0, Command.Length);
}
```

Útvonal készítése

A szimuláció folyamatos naplózást végez századmásodperces pontossággal a kiadott mozgatási parancsokról, amelyet a szimuláció befejezésekor automatikusan elment egy „teszt.xls” állományba. Az nxtBTC „Get a Route” gombjának lenyomására előkereshetjük (csak és kizárólag .xls kiterjesztésű állományok látszanak a dialógus során) ezt az állományt. Amennyiben a kiválasztott fájl ténylegesen egy útvonal információit tárolja, aktiválódik a „Create Route” nyomógomb, amellyel előállíthatunk egy útvonaltervet, amit majd a „Send Route” gomb megnyomásával útnak is indíthatunk az NXT felé.



39. ábra: A program működés közben

ÖSSZEGZETT KÖVETKEZTETÉSEK

Szakdolgozatunk kitűzött céljai közül sikeresen megoldottuk a szimulált robot billentyűzettel, valamint a valódi robot vezeték nélküli, számítógéppel történő irányítását. Munkánk során több nehézséggel szembesültünk. Egyrészt a Yorishima Yamamoto által kidolgozott projektmunkát kellett működésre bírnunk. Megnehezítette a dolgunkat a hiányos dokumentáció és az addig csak kezdetleges állapotban létező telepítési útmutató. Miután kiismertük a szimuláció működését, megkezdhettük a szimuláció fejlesztését, irányíthatóvá tételét. Másrészt meg kellett ismernünk az NXT Bluetooth-os kommunikációs képességeit. Ismereteink bővítésében segítségünkre volt a „Távközlő hálózatok" megnevezésű kurzus teljesítése is, ahol megismerkedtünk a Bluetooth működésével és a Microsoft Visual Studio 2010 használatával, egyben a C# programnyelv alkalmazásával.

Miután a fejlesztett programmal képesek lettünk a robot irányítására, valamint a robot szimulációjának fejlesztése is befejeződött, a szimulált robot útvonalának a valódi robottal történő megismétlésére koncentrálhattunk. Kutatásaink során arra a következtetésre jutottunk, hogy a szimulált útvonal pontos megismétléséhez a robot hardveres fejlesztése elengedhetetlen, jelenlegi állapotában csak megközelítőleg tudja teljesíteni azt. Ahhoz, hogy az ismétlés közel hibamentesen történjen, visszacsatolás lenne szükséges, melyhez példaként használható lenne a HiTechnic cég Compass Sensor terméke. Ezt a szenzort iránytűként használva a robot mindig pontosan tudná tartani a kívánt fordulási irányt.

Általános tapasztalatok a LEGO Mindstorms NXT használatával kapcsolatosan

Az NXT a LEGO készletek közül az egyik legdrágább, a valódi robotot tartalmazó programozási környezetek közül azonban az egyik legolcsóbb. Az alapcsomag nagyjából 80 ezer Ft-ért kapható. Nagy előnye a szabad átépítéseknek köszönhető flexibilitás. Rendkívül összetett formájú robot építhető LEGO Technic elemek felhasználásával. A hardver gyengepontja a kábelezés. A motorok és szenzorok bekötését úgy kell megoldani, hogy a kábelek ne akadályozzák egymást és a robot mozgását sem.

Mindent egybevéve a készlet lehetőséget ad arra, hogy megismerjük a robotika alapjait, elsajátítsuk a robotokat vezérlő programok írásának lépéseit és akár komplex feladatok végrehajtását is az NXT-vel végeztessünk el. Szerencsére a Debreceni Egyetem mérnökoktatása is felismerte a LEGO készletekben rejlő lehetőségeket.

IRODALOMJEGYZÉK

1. http://mindstorms.lego.com/en-us/Default.aspx
2. http://www.agr.unideb.hu/if2008/kiadvany/papers/G36.pdf
3. http://www.vilagtudomany.hu/index.php?data%5Bmid%5D=7&data%5Bid%5D=898&a-robot-9-eve
4. http://www.aut.vein.hu/oktatok/MagyarA/Rob1.pdf
5. http://en.wikipedia.org/wiki/Segway_PT
6. http://www.segway.com/downloads/pdfs/Getting_Started_Manual.pdf
7. http://www.sulinet.hu/inform/balazscikk/legorobot/legorobot.html
8. http://www.nxtprograms.com/index2.html
9. http://www.generation5.org/content/2004/legway.asp
10. http://www.philohome.com/nxtway/nxtway.htm
11. http://web.mac.com/ryo_watanabe/iWeb/Ryo's%20Holiday/NXTway-G.html
12. http://lejos-osek.sourceforge.net/nxtway_gs.htm
13. http://www.mathworks.com/matlabcentral/fileexchange/19147
14. Keviczky L., Bars R., Hetthéssy J., Barta A., Bányász Cs: Szabályozástechnika; Műegyetemi Kiadó 2007
15. R.C. Dorf, R.H. Bishop: Modern Control Systems;Pearson Educational International, 10th edition
16. http://legolab.daimi.au.dk/DigitalControl.dir/NXT/Sensors.html
17. http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NGY1044
18. http://mightor.wordpress.com/2009/11/17/you-spin-me-right-round-baby-right-round/
19. http://www.analog.com/library/analogDialogue/archives/37-03/gyro.html#return1
20. Albert Ko, H. Y. K. Lau, T. L. Lau: General Suppression Control Framework: Application in Self-balancing Robots, 2005
21. http://www.mathworks.com/matlabcentral/fileexchange/13399
22. http://www.mathworks.com/matlabcentral/fileexchange/19147-nxtway-gs-self-balancing-two-wheeled-robot-controller-design
23. http://www.mathworks.com/matlabcentral/fileexchange/25207
24. http://www.mathworks.com/matlabcentral/fileexchange/3630-simulink-keyboard-input
25. http://www.jataka.hu/rics/lego/index.html

KÖSZÖNETNYILVÁNÍTÁS

Köszönetet mondok témavezetőnknek, Dr. Cserháti Csabának, akinek segítőkészsége és tanácsai nélkül ez a szakdolgozat nem valósulhatott volna meg, valamint Pethe Szabolcsnak, akivel egy csapatban dolgoztunk szakdolgozatunk elkészítésében.

Köszönetet mondok továbbá a Szilárdtest Fizika Tanszéknek, amiért rendelkezésünkre bocsátotta a Lego Mindstorms NXT egy példányát.