SZAKDOLGOZAT

Veres Ádám

Debrecen

2009

Debreceni Egyetem

Informatikai Kar

Felderítő robotok
(Navigáció - Odometria)

Téma vezető:

Dr. Szabó István

DE Szilárdtest Fizika Tanszék

Készítette:

Veres Ádám

Mérnök Informatika

Debrecen

2009

1. Tartalomjegyzék

2. Köszönetnyilvánítás

Szeretném megköszönni a segítséget a témavezetőmnek Dr. Szabó István egyetemi docensnek a Debreceni Egyetem Szilárdest Fizika Tanszék vezetőjének. Valamint Cserháti Csabának a kölcsön kapott LEGO robotért.

3. Bevezetés

Felderítés, felfedezés, kutatás, az ismeretlen megértésének vágya egyidős az emberiséggel. Mióta civilizációnk létezik, törekszünk megismerni környezetünk és a körülöttünk lévő világ technikailag elérhető részeit. Ez a folyamat az újfajta energiatechnológiáknak köszönhetően az elmúlt 100-150 évben felgyorsult. Elkezdtük megismerni tágabb környezetünket is, ezen feladatunk megoldásában akadtak segítőtársaink is: a robotok.

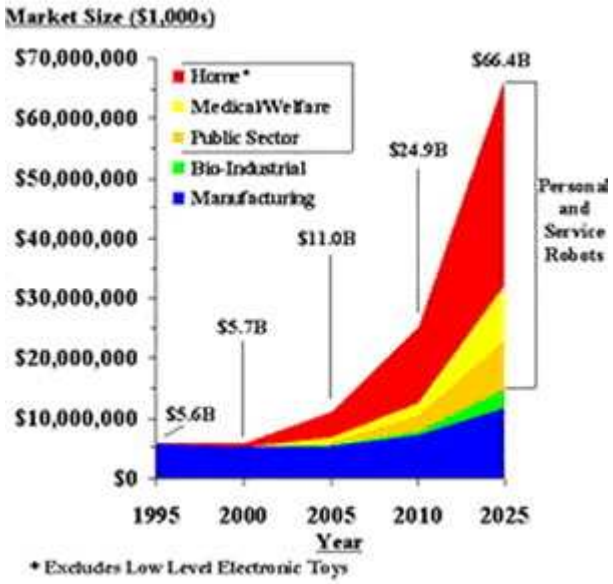
Számtalan hely létezik, ahová az embernek eljutni körülményes vagy veszélyes, ez azonban nem okozhat gondot egy robot számára. Így lehetőségünk nyílott megvizsgálni a tengerek mélyét, a világűr sötét zugait, a piramisok belsejét. Katonák százainak életét mentik meg az aknamentesítő, a földön, vízen, levegőben közlekedő felderítő robotok.

Ez mind elképzelhetetlen lenne az eszközeink, főként az informatika fejlődése nélkül. Képtelenek lennénk előállítani ezen technikai vívmányokat, modern számítógép vezérelt gyártósorok és gépeink pontos számításai nélkül. Az ember minél több munkától próbál megszabadulni, szeretné könnyebbé tenni az életét. Ez az elkövetkezendő években – évtizedekben látszik kicsúcsosodni. Mára már oda jutottunk hogy, a robot-gépeinket is robotok gyártják. Az emberiség nagy része már képtelen elképzelni életét a számítógép technológia nyújtotta kényelem nélkül. A mindennapi használati tárgyainktól kezdve, mint az egyszerű ébresztőórák, a bonyolultabb autóinkon keresztül, az űrtechnológiában használt eszközeinkig, mindent pontosan és gyorsan dolgozó gépek hada állít elő. Robotok ezrei dolgoznak mindennap helyettünk, értünk.

Azért választottam ezt a témát szakdolgozatnak, mert úgy érzem a világban még több és jobb robotra lesz szükség az elkövetkezendő időkben. Még van mit felfedezni az embernek, és engem is hajt az ismeretlen felderítése utáni vágy. Az informatikában ez komplex és sokoldalú dolog, amiből rengeteget lehet tanulni. Talán itt érezhető legjobban az alkotás, izgalmas és szórakoztató mikor, az élettelen életre kell.

A robotikán belül a navigációval és ezen belül is az odometriával fogok foglalkozni. Valamint ismertetem az általam használt rendszereket és eszközöket. Célként volt megfogalmazva, hogy rávilágítsak arra ez a módszer mennyire megbízható, mik a hibái, hogy lehet javítani a pontosságon és kalibrálni a robotot. Kicsit vissza szeretnék tekinteni a történelembe, hogyan is kezdődött mindez.

A terepen használt felderítő robotok, nagyban hagyatkoznak odométereikre, ezért azoknak nagyon pontosnak kell lenniük. A szakdolgozatban ismertetett rendszer egy kétirányú négyszögkísérletre épül. Az itt kapott eredményeket egy képfeldolgozó programmal kiértékelem, és ebből ha szükséges korrekciókat számolok, melyek a szisztematikus hibákat javítják ki. Ennek a feladatnak a megoldását, hardveres és szoftveres hátterét mutatom be.



3.1. ábra Robotokra költött összeg, az évszám függvényében
Jósolva 2025-ig, a színek a területi eloszlást mutatják

4. Robotikáról általában

4.1 Alapfogalmak

Ebben a részben szeretnék segítséget nyújtani az olvasónak, hogy elboldoguljon a robotok világában. Főleg az általam is használt és a ki nem hagyható rendszerekről, eszközökről, alapfogalmakról fogok beszélni.

4.1.1 Mi is az a robot?

Hála Hollywoodnak ma már minden emberben kialakult valamilyen kép a robotokról és ahogy születnek meg az egyre fantasztikusabb sci-fik ez a kép egyre ferdébb. A valóság általában nincs olyan izgalmas, mint az átalakulni képes, vagy embereket tökéletesen megformázó, a halált két kézzel szóró robotok története. Bár itt is találhatunk már ijesztő, aggodalomra okot adó fejlesztéseket, az érdekesség kedvéért a fejezet végén hozok majd pár példát.

A robot olyan elektronikai egység, amely programozható/ újraprogramozható és ezen programmal képes akár összetett feladatok megoldására is. Vannak emberek által távirányított robotok, és olyanok melyek működését számítógép felügyeli. Általában olyan munkákat bíznak rájuk, amelyek az ember számára nehezek, nem biztonságosak (harci felderítés), vagy monotonak, de nagy pontosságot igényelnek (autógyártás).

A robotok sok szempont szerint osztályozhatóak, minket a szakdolgozat keretén belül igazából két szempont érdekel:

- Autonómia
- Mobilitás

Az autonómia, azt határozza meg, hogy a robot mennyire önálló, mennyi külső segítséggel oldja meg a rábízott feladatot. Ez összefügg valamennyire a mobilitással is, tehát mekkora az a terület, ahol operálhat. A kisterületen mozgó, megadott feladatot ellátó és könnyen megépíthető robotkarok, jól szemléltetik az ipari

alkalmazásokat. Változatlan környezetben, akár szenzoros visszacsatolás nélkül is gyorsan és eredményesen dolgoznak, specializált feladatellátásuk miatt, könnyen programozható, összerakható és eredményesen működik. Más a helyzet a nagyobb mobilitású robotokkal melyeknek a folyamatosan változó világban kell megállni a helyüket. Különböző terep és időjárási viszonyok között változó feladatokat kell ellátniuk, ehhez egyre nagyobb autonómia, intelligencia, döntésképesség és hibatűrés kell mind szoftveresen mind hardveresen. Ezen szempontok mentén megszokás különböztetni három robotgenerációt(1).

- Első generációba tartozók nem foglalkoznak a körülvevő környezettel, az előre megírt programjuk alapján mozognak, minden helyzetben ugyanúgy. A valós világban használhatatlanok, csak statikus környezetben alkalmazhatóak (különböző gyártástechnológiákban).
- Második generációs robotok melyek már szenzorok mérési eredményeivel képesek ellenőrizni vagy javítani munkájuk menetét (például kikerülni egy akadályt).
- És végül a harmadik generáció, amely már komoly mesterséges intelligenciával van ellátva. Képes a komplex feladatmegoldásra, tanul és akár módosíthatja saját programját. Általában kutatási területeken használják, képesek alak felismerésre, hanggal vezérelhetőek és önálló döntést hoznak.

Az ipari és mobil (pl. felderítő) robotok közötti különbség jól szemléltethető ezekkel. Míg előbbiek általában első és második generációsak, addig mobil társaik okosabbak második, de főleg harmadik generációsak. Egy fontos mérőszám a szabadsági fok a térben való elmozdulás lehetőségét mutatja, ez maximálisan 6 lehet. A helyváltoztatásra képes robotoknál ez majdnem egyértelműen mindig megvan (de legalább 4), az iparban viszont annál ügyesebb, annál jobban használható egy munkagép minél több tengely mentén képes a munkadarab manipulálására.

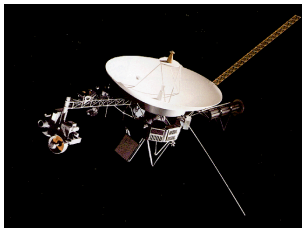
4.1. Jövőre vonatkozó jóslás a generációk fejlődéséről

A felderítő robotok tovább csoportosíthatók aszerint, hogy milyen környezeti közegben haladnak.

A **levegő**t használó robotokkal viszonylag könnyű a dolgunk, megépítésük egyszerűbb és navigációjuk könnyű, mivel a levegőben nem nagyon ütköznek kikerülendő akadályokban, ezért gyorsan terjednek és fejlődnek. Már egyetlen modern hadsereg sem engedheti meg magának, hogy ne rendelkezzen ilyen UAV (*Unmanned Aerial Vehicle*) egységekkel.

4.2. UAV

A **világűr** nagyon hasonlít a levegőhöz, csak más környezeti hatások érik az eszközeinket (pl.: időjárás ←→ kozmikus sugárzás)



4.3. Satelit (voyager2)

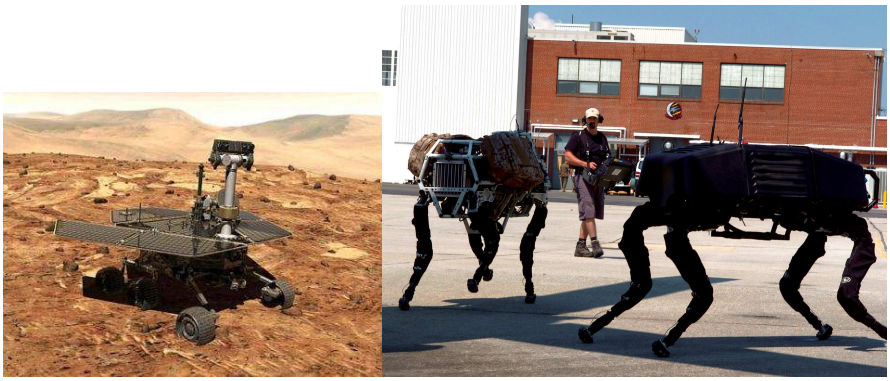
A **víz** felszínén, vagy alatta közlekedő robotok a speciális körülvevő közeg miatt érdekesek, általában tudományos és ipari célokra használják. Gyakran fizikai kialakításuk valamilyen állatot utánoz. AUV (*Automated Underwater Vehicle*)

4.4. GARBI-AUV

És talán a legbonyolultabb közeg a **szárazföld**, amin mi emberek olyan egyszerűen és ügyesen közlekedünk, kis teremtményeinknek sokszor komoly fejtörést okoz. Itt két koncepció van terjedőben a lábakon lépkedő és a valamilyen kereken, lánctalpakon mozgó járművek.

A lépegetők, leginkább a humanoid formák elég gyengék, lassúak és egy lépcsőfok is okozhat nekik akadályt. Többlábú társaik már valamivel stabilabbak, sebességük ugyan még mindig elég alacsony. Itt érdemes megemlíteni azon törekvéseket melyek egyre inkább emberszerű mintázatot utánoznak. Motorokkal, pneumatikus vagy hidraulikus úton mozgatják „végtagjaikat” vagy mimikájukat.

A kerekeken vagy lánctalpakon mozgók, stabilabbak gyorsabbak és használhatóbbak, egyszerűbb és kisebb energiaigényű konstrukciók. Valószínűleg ennek okán jóval elterjedtebbek is.



4.5. Spirit mars rover / Big dog (DARPA)

Külön érdekességként érdemes megemlíteni a kétéltűeket melyek egy sor új, izgalmas problémát vetnek fel.

Érdekes még továbbá a robot csoportok működése, egyénileg és együtt is. Erre egy remek példa a robotfoci, de a felderítésben is komoly szerepet játszhatnak a nagy területet lefedni képes robotcsapatok.

4.6. Robotfoci csapat

A jövőben felgyorsuló fejlődés várható. Az élet minden területén egyre nagyobb hasznunkra vannak. Autonómiájuk növekedni, képességeik javulni fognak. Szenzoraink és aktuátoraink pontosabbak és gyorsabbak lesznek.

4.1.2 Szenzorok és aktuátorok

Aktuátoroknak azokat az elektromos, mechanikus, pneumatikus vagy hidraulikus elemeket nevezzük összefoglalóan, melyek a robot valamilyenfajta mozgásáért felelnek.

A szenzorok pedig valamilyen információt adnak a környezetről, ez igen széles skálán mozog. Mérhetnek távolságot, dőlést, nyomást, elmozdulást stb. Jóformán mindent, az egyszerű hőmérséklettől a képek gazdag részletességéig.

Gyakran használt eszközeink, a kamera különböző fajtái, hő, infra, normál, (szükségesek a fejlet képfeldolgozási eszközök), az ultrahangos vagy lézeres távolságmérő, nyomásérzékelők, gps.

A csoportos tevékenységet előmozdító érdemes megemlíteni a kommunikációs technológiák fejlődését, melyek előbbre mozdították a felderítés valós idejűségét. Pl.: WIFI, bluetooth, infra.

4.2 Mióta vannak köztünk

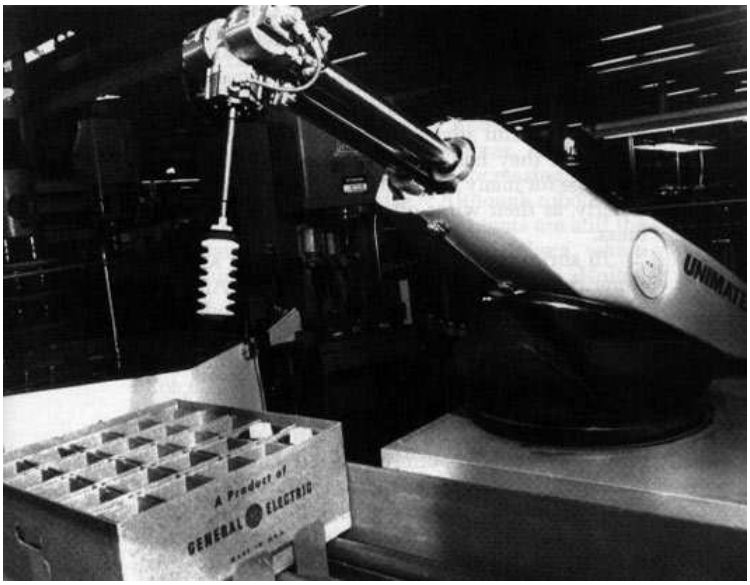
Mint általában minden új technikai vívmány a robotok is először a harctereken jelentek meg, méghozzá a II. Világháborúban. A Német hadsereg által használt távirányított kis lánctalpas egységek, a góliátok(3), képesek voltak 75-100 kg-nyi robbanótöltetet is célba jutatni, ezzel több csata megnyeréséhez is hozzájutatták a Német erőket.

4.7. Goliath

A következő évtizedekben a technológiák kellő fejletségének hiánya miatt nem nagyon használtak robotokat, a távirányításnak nem volt értelme a mesterséges intelligenciák, pedig még nagyon fejletlenek voltak, nem voltak képesek ellátni önálló harci feladatokat.

A robotika fejlesztésének fő irányvonala ekkor áttevődött az iparra, ahol automatizált gyártósorokat próbáltak meg építeni. 1956-ban George Devol és Joseph Engelberger megalapította az első ipari robotok fejlesztésére és gyártására

specializálódott céget, alig 4 évvel később pedig az MIT Servomechanisms Laboratory bemutatta a számítógépek által vezérelt gyártást. Az egyetem Automatically Programmed Tools nevet viselő projektjének keretein belül elkészítettek egy nyelvet az APT-t, amivel egy marógépet irányítottak. A gép a bemutatón minden egyes résztvevőnek készített egy hamutartót. Ezután jött az UNIMATE, az első ipari robot amely a 60-as évek elején állt munkába a General Motors-nál(4). Egy mágneses dobon rögzített utasítássornak megfelelően a 1814,5 kg súlyú kar, forró fröccsöntött fémdarabokat szortírozott. Ettől kezdve ezek a szerkezetek egyre több dolgot sajátítottak el és szép lassan meg tanultak a térben mozogni. Mint már kifejtettem, ma már elképzelhetetlen az automatizált gyártósorok nélkül, a mai modern világ igényeinek kiszolgálása.



4.8. UNIMATE

A robotika lassú lépésekben újra fejlődésnek indult a 70-es évek közepétől. Az informatikai vívmányok méretének csökkenése és kapacitásuk növekedése a 80-as évek végére a 90-es évek elejére, tette elérhetőbbé széles körben a robotika tudományának fejlesztését. A robot navigáció egyik mérföldköve a Stanford Cart ("kordé") 1979-ben bemutatott szerkezete, mely emberi közbeavatkozás nélkül át tudott haladni egy székekkel zsúfolt szobán. Köszönhető ez Hans Moravecnek, aki a gépet sztereólátással szerelte fel. A "kordé" tetején lévő sínre tv-kamerát erősített, amely különböző szögekből felvételeket készített, és ezeket továbbküldte a

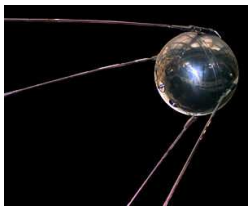
helyzetelemző számítógépnek, mely képfeldolgozási módszerekkel állapított meg magának optimális útvonalat(4).

4.9. kordé

A robotika fejlődését erősen elősegítő tudományág az űrkutatás volt, amely szorosan összekapcsolódott a katonai fejlesztésekkel, ez köszönhető annak, hogy az űrkutatás, az első műholdak megjelenése, mind a hidegháború kezdeti időszakára tehetőek. Valamint ebben az időszakban zajlott az űrverseny is, a hold meghódításáért. És természetesen az űrkutatás eszközeit lehetetlen létrehozni a számítógép vezérelt (pl.: CNC) gyártógépek nélkül(7).

4.10. CNC munkagép

Az ember természetesen nem kockáztat feleslegesen, így az űrbe is elsőnek élettelen dolgot küldött, ami nem volt más, mint a Szputnyik 1957-ban. Ez természetesen még közel sem volt robot valódi célja a hordozóeszköz tesztelése volt, és hogy elérjék az első kozmikus sebességet. Mindössze két rádióadót vitt magával egy 50 cm-es alumínium gömbbe zárva melyek 20 és 40 MHz-en sugároztak, a jeleket az egész bolygón lehetett fogni a rádióamatőröknek.

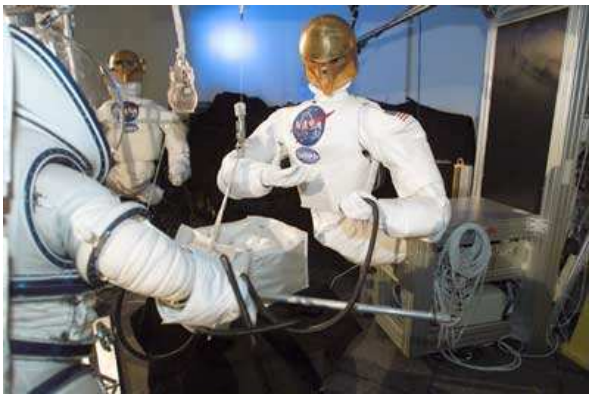


4.11. Szputnyik

A következő években számtalan más műholdat küldtek fel, valamelyik a földet derítette fel nagy részletességgel, valamelyik pedig a naprendszer egy másik égitestét. Rengeteg képpel és mérési eredménnyel szolgáltak üstökösökről, kisbolygókról, naprendszerünk tagjairól és az őket kísérő holdakról. Teljes mértékben önműködő robotok nem nagyon vannak kint, a mesterséges intelligenciák még nem elég fejlettek. A robotok nagy részét, pl. műholdak vagy a mars roverek mind távirányítottak, vagy csak részben hagyatkoznak saját programjukra és kiegészítik a parancsként kapott utasításokat.

Nagy sikernek számít a két kis Mars Rover a Spirit és az Opportunity(5). 2003-ban landoltak a Mars felszínén és mindössze pár hónapnyi munkára tervezték őket azonban a kis robotok még tavaly is bőszen szántották a Mars felszínét. Hónapról hónapra, évről évre újabb megdöbbentő felfedezéseket tettek, melyekkel közelebb jutottunk a Mars megismeréséhez. Ez érdekes és örömteli példa a felderítő robotokra. A Rovereket is távirányítják, parancssorozatokat adnak neki, hogy merre menyjen. A rover navigációs szoftvere, ha veszélyt érzékel, (stereo látása van, és 3D-s képet alkot) némileg módosíthatja a kapott útvonalat. 2006-ban az Opportunity marsjáró elakadt egy homokdűnében. Az irányítószemélyzet azt hitte, hogy haladnak előre mivel a kerekek forogtak, csak akkor jöttek rá, hogy állnak mikor körülnéztek a kamerákkal, ez egy olyan hiba az Odometriában amire még visszatérünk.

Egy másik érdekes robot az űrkutatásban, amely a közeljövőben állhat rendszerbe, ha megfelel az elvárásoknak, egy emberszabású robot mely az űrhajósoknak segíthet az űrséták alkalmával. A DARPA és a Jhonson űrközpont közös fejlesztése. A robonauta szerszámokat és alkatrészeket juttat el az űrhajósokhoz és egy próba alkalmával veszélyes vegyi anyagot takarított le egy ember társa űrruhájáról(6).

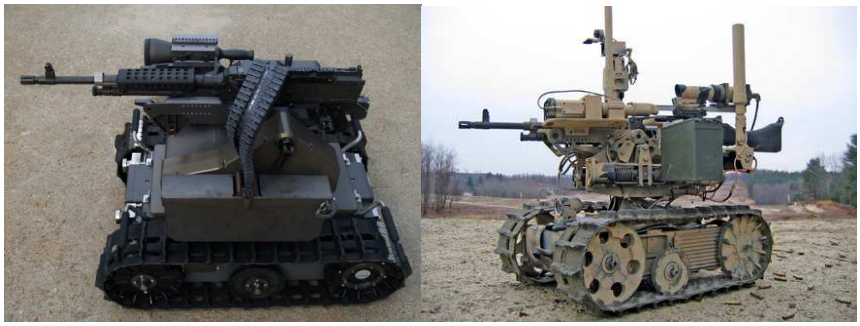


4.12. robonauta (DARPA)

Az űr sötétjének kutatása mellett saját bolygónkat is megpróbáltuk felfedezni kevésbé látványos ugyan de technikailag mégis nehéz feladat a tengerek mélyének kutatása. Ez megint nagyon szerteágazó terület a ritka fémeken, olajon, hajóroncsokon, az Antarktisz vizeinek és különleges mélytengeri állatok felkutatásán keresztül mindenfélére használják őket. Egyik nagy figyelemmel követett akció a Titanic felderítése és egy darabjának kiemelésére tett kísérlet volt. Vagy több alkalommal is felderítettek elsüllyedt tengeralattjárókat és segédkeztek a mentésben.

Az elmúlt 15-20 évben ismét a katonai alkalmazás vált meghatározóvá, az informatika fejlődésének hála(2). Számottevően az Amerikai hadseregben van rájuk a legnagyobb igény, csak Irakban több mint 10000 robot teljesít szolgálatot. Nagyrészük légi vagy szárazföldi felderítő illetve akna és bombamentesítő feladatokat lát el táv irányítva persze. Vannak a hajók automata védelmi rendszerét képző rakéták elleni pajzsok melyek nagyteljesítményű géppuskákkal lövik ki biztos távolságban a rakétákat. És vannak olyan egységek is melyek géppuskákkal,

gránátvetőkkel vannak felszerelve és az ellenség felkutatása és meg semmisítése a céljuk, remélhetőleg emberi kontroll mellett.



4.13. SWORD

5. Odometria

Ez az egyik leggyakrabban alkalmazott helymeghatározási módszer. Az odometria vagy relatív helymeghatározás, egy olyan navigációs módszert takar, amikor nem a körülvevő környezetet pásztázzuk szenzorokkal, annak érdekében, hogy tudjuk hol vagyunk, hanem egy kezdeti állapot ismeretéből, és jelen esetben a mozgást megvalósító motorok szögelfordulásának mértékéből, kikövetkeztetjük, hogy a mobil egység hova jutott el a térben a kezdő állapotához képest. A pozíciónkat két dimenzióban a középpont koordinátáival és a robot irányszögével írhatjuk le. Ezen kívül ismernünk kell az egységet meghatározó geometriai paramétereket is (kerekek átmérője, tengelytávolság), ha ezeket a paramétereket nem ismerjük kellő pontossággal, akkor a helymeghatározás szisztematikus hibákkal terhelt lesz. Ezen hibák mértéke kalibrálással csökkenthető.

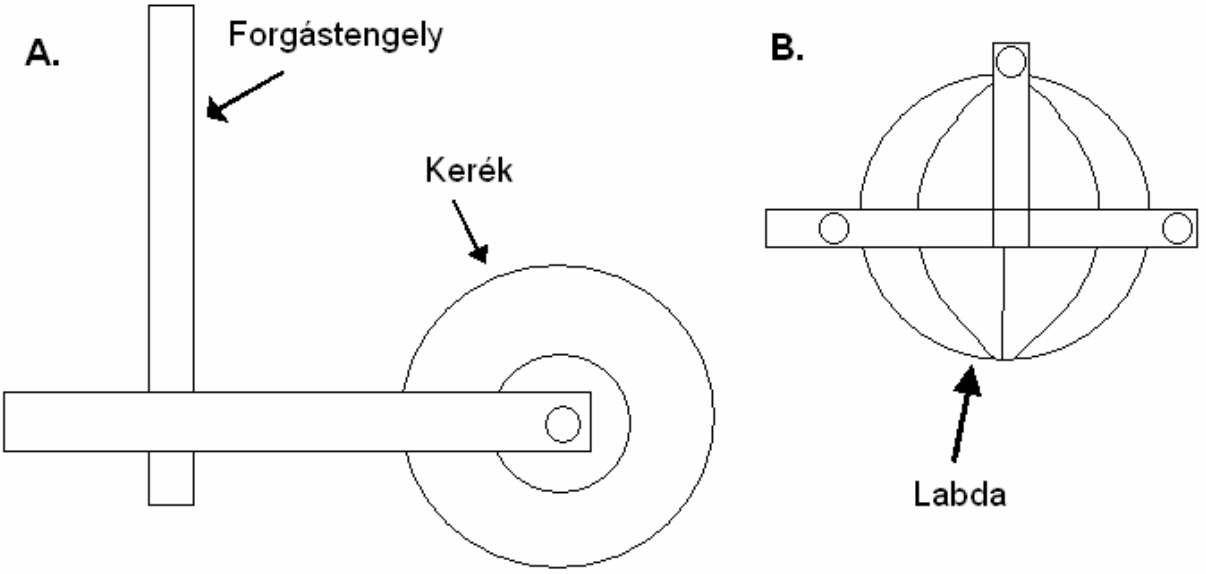
Az odometria önmagában természetesen nem alkalmas komplex navigációra, csak valamilyen globális rendszerrel, vagy a külvilágot pásztázó szenzorokkal együttműködve. Viszont nagy előnye hogy olcsó és rövidtávon jó pontosságot biztosít, viszont a hibák idővel felgyűrűznek, és összeadódnak különösképpen fordulásnál. Két abszolút mérés között alkalmazható, nagyobb pontosságú relatív helymeghatározás mellett kevesebb költséges szenzoros mérést kell végezni és ennek következményeként nincs szükség annyi adatfeldolgozásra.

5.1 A konstrukció

Kísérletezés képen a LEGO cég által kifejlesztet Mindstorm egységet használtam, könnyű szerelhetősége és kész elektronikája miatt. Később mutatkoztak meg a hátrányai is.

5.1. LEGO Mindstorm

Egy differenciál hajtásos modellt raktam össze, ami úgy nézet ki, hogy két egymástól függetlenül hajtott kereke volt és egy harmadik, nem hajtott kereken támaszkodott, ami követte a robot mozgását. A támaszkodó kerék elfordulás előtti helyzete nagyban befolyásolta, hogy a robot a neki megadott fokot elfordulja, néha be-be akadt, ezért később ezt egy labdára cseréltem. 5.2. ábra.



5.2. Az A. konstrukció sokszor beragadt, míg a B. jól működött

Ennek a konstrukciónak a sima egyenes terep az épületek az utcák fekszenek a terepjáró képessége nagyon gyenge. Felépítése roppant egyszerű, ezért vezérlése könnyű, a kerekek ellentétes irányba forgatásával képes egy helyben megfordulni. A kerekek elfordulásának mérésével könnyű meghatározni hogy merre mozdult el a robot.

A LEGO készletből adódó további problémák, hogy a robot önsúlyát is alig bírta el, a kerekek tengelyei meghajoltak. Nem nagyon találkozni két egyformára gyártott motorral, tehát azonos paraméterek mellett az egyik motor kisebb távot tesz meg, mint a másik. Később próbálkoztam szenzorokkal is, mondanom sem kell, hogy két készletből származó, azonos szerepkörű szenzor más-más értékeket mér. Sajnos magamnak is köszönhetek néhány problémát, ezeket a bajokat tetéztem azzal, hogy ragaszkodtam a C# fejlesztői rendszerhez, és bluetoothon kommunikáltam a robottal. A laptopon számoltam, míg a tégla csak ''haza'' küldte a szenzorok adatait és fogadta a motormozgási parancsokat. A bluetooth puffere 5 adatot tud tárolni ami küldésre vár, ha több adat jön a legrégebbi elvész, ez adatvesztéssel járt, ezért a pontosság nagyon romlott, így arra a döntésre jutottam hogy az adatfeldolgozást meggyorsítandó Java rendszerre váltok és telepítek egy JVM-et (Java Virtual Machine) a LEGO irányító egységére. Ez felgyorsította kicsit a munkafolyamatot, sokkal kevesebb lett a hibalehetőség.

Technical specifications LEGO Mindstorms NXT Brick:

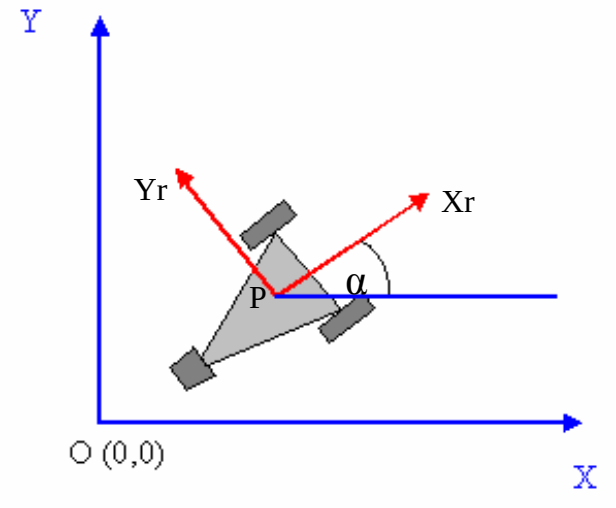
- 32-bit ARM7 microcontroller
- 256 Kbytes FLASH, 64 Kbytes RAM
- 8-bit AVR microcontroller
- 4 Kbytes FLASH, 512 Byte RAM
- Bluetooth wireless communication (Bluetooth Class II V2.0 compliant)
- USB full speed port (12 Mbit/s)
- 4 input ports, 6-wire cable digital platform (One port includes a IEC 61158 Type 4/EN 50 170 compliant expansion port for future use)
- 3 output ports, 6-wire cable digital platform
- 100 x 64 pixel LCD graphical display
- Loudspeaker - 8 kHz sound quality. Sound channel with 8-bit resolution and 2-16 KHz sample rate.
- Power source: 6 AA batteries

5.2 Koordináta rendszerek

A robotunk, mozgását mindig egy referencia ponthoz viszonyítva végzi. A terepen, a robot mozgás terében, ki kell jelöljünk egy ilyen pontot, ez lesz a képzeletbeli globális koordináta rendszerünk origója azaz a (0,0) koordinátájú pont. A robot pozícióját ehhez képest kell leírjuk az alábbi módon:

$$\xi_i = \begin{pmatrix} x \\ y \\ \alpha \end{pmatrix}$$

A robot pozíciója (x,y) koordináták és α szögelfordulás. Szükségünk van még egy koordináta rendszerre melynek origóját, a roboton kell elhelyeznünk, ezt nevezzük P bázis pontnak. A roboton azért szükséges egy irány kijelölése, hogy megadjuk a robot önmagához képesti pozícióját. A referencia koordináta sík: (Xr,Yr).



5.3. koordináta rendszerek

A két koordináta rendszer között szükséges kapcsolatot teremtenünk, ha le akarjuk írni a robot pontos mozgását. Ezt az ortogonális rotációs mátrix segítségével tehetjük meg.

$$R(\alpha) = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

A robot viselkedése a globális koordináta rendszerben a következő képen írható le:

$$\dot{\xi}_i = R(\alpha)^{-1} * \xi_r$$

A mozgást leíró egyenletet a geometria kialakítás ismerete mellett, speciálisan arra tudjuk felírni. A mi esetünkben, differenciál hajtás (két külön mozgatható kerék) és egy támaszkodó kerék mellett, ami elvileg tökéletesen követi a robotot:

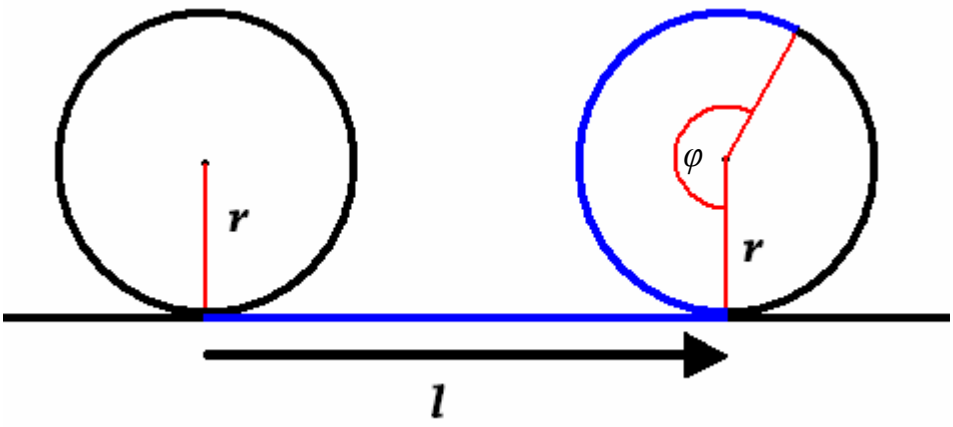
$$\dot{\xi}_i = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\alpha} \end{pmatrix} = f(l, r, \alpha, \theta_1, \theta_2)$$

A mozgás leírása a kerekek közötti távolság (l), a kerekek sugara (r), az egység iránya α, és a kerekek sebessége alapján (θ_1, θ_2) lehetséges. Az egyenlet felírásához tehát, először a robot saját koordinátarendszeréhez képesti mozgását kell deklarálni, majd ezt a mozgást kell a globális koordináta rendszerhez képest megállapítani. A fenti példánál maradva, ha a robot két kereke azonos sebességgel azonos irányban forog, a robot az Xr tengely mentén egyenes vonalú mozgást végez. Ha a két kerék ellentétes irányban forog, a robot saját középpontja körül forog. Bonyolultabb eset, ha a kerekek eltérő sebességgel forognak, mivel ekkor a robot egy referencia pont körüli körpályán történő mozgást végez.

5.3 Elmozdulások számítása

Az egyszerűség kedvéért nem használtam áttétet a motorok és a kerek között így amennyit a motor fordult annyit fordult a kerék is. A motor fordulatából és a hozzátartozó kerék átmérőjéből egy egyszerű képlettel megadható a robot előre illetve hátra mozdulása(8).

Az l hosszúságú előre mozdulás:



5.4. Előre mozdulás

Radiánban:

$$l = \varphi * r$$

Én fokban jobban szeretem:

$$l = \varphi * r * \frac{\pi}{180}$$

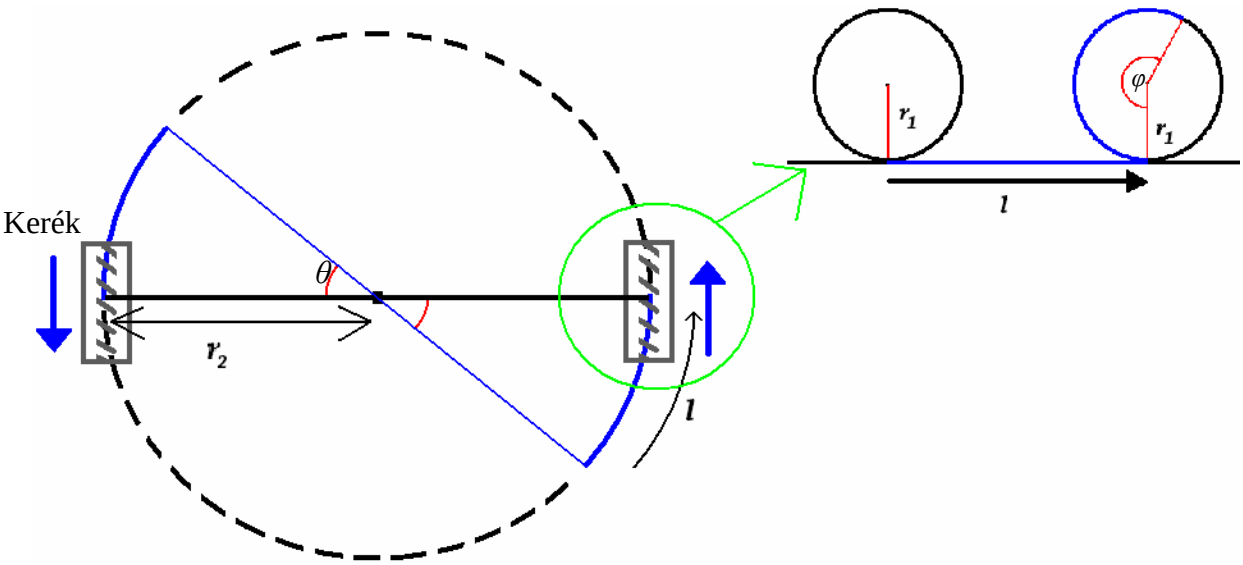
Mivel én adom meg, hogy a robotnak mennyit kellene mozdulni, az l-et ismerem. Amire szükségem van az pedig a φ.

$$\frac{l}{\varphi} = \frac{r * \pi}{180} \rightarrow \varphi = \frac{180}{r * \pi} * l$$

A két kerék távolságának meghatározása után már a fordulások is megadhatók.

A θ szögű elfordulás:

A kerekek egy körpályán mozognak körbe, ellentétes irányú elmozdulással. A kör középpontja, a robot kerekeit összekötő tengely középpontja(8).



5.5. Fordulás

Fokban:

$$(l =)\theta * r_2 * \frac{\pi}{180} = \varphi * r_1 * \frac{\pi}{180}$$

Ebből θ egyenlő:

$$\theta = \frac{\varphi * r_1 * \pi}{180} * \frac{180}{r_2 \pi} = \varphi * \frac{r_1}{r_2}$$

Tehát:

$$\theta = \varphi * \frac{r_1}{r_2}$$

A Javaban megírt próba kód:

```
import lejos.nxt.*;

public class proba
{
	public static void main (String[ ] args) throws Exception
	{
		Motor aMotor = Motor.A;		//motor a példányosítása
		Motor cMotor = Motor.C;
		double szog = 90;		//fordulás kívánt mértéke szögben
		double tavolsag = 1;		//megtett távolság méterben
		double kerek_atmero = 0.043;	//méter
		double kerektav = 0.12;
		double elore_szamitas = 0;
		double fordulas_szamitas = 0;
		int fsz = 0, fsz2 = 0;

		elore_szamitas = (180/((kerek_atmero/2)*Math.PI))*tavolsag;
		fsz = (int)Math.rint(elore_szamitas);
		fordulas_szamitas = (kerektav/kerek_atmero)*szog;
		fsz2 = (int)Math.rint(fordulas_szamitas);

		aMotor.rotate(fsz, true);		//motor mozgatási parancs
		cMotor.rotate(fsz, false);
		aMotor.rotate(-fsz2,true);
		cMotor.rotate(fsz2,false);

		Thread.sleep(500);
		aMotor.stop();			//motor állj parancs
		cMotor.stop();
	}
}
```

5.4 Hibák

Sajnos itt még nem vagyunk kész, teljesen idealizált környezetben idealizált robottal ez minden gond nélkül működik és roppant hosszú parancs kódok végrehajtása során sem történik katasztrófa. A baj, mint tudjuk a világ nem ideális. Számtalan hiba lehetőséget rejt magában ez a módszer, egy részük viszonylag könnyen kijavítható (pl.: a különböző kerékátmérőkből eredő hiba, a pontosabb gyártástechnológiával, vagy kalibrációval orvosolható), vannak azonban olyan hibák amelyeket sosem tudunk kiszűrni. A hibák csoportosításának legjobb módja, ha úgy osztjuk őket ketté, mint állandó (rendszeres, szisztematikus) hibák és esetleges (nem állandóan fennálló, rendszertelen) hibák(8).

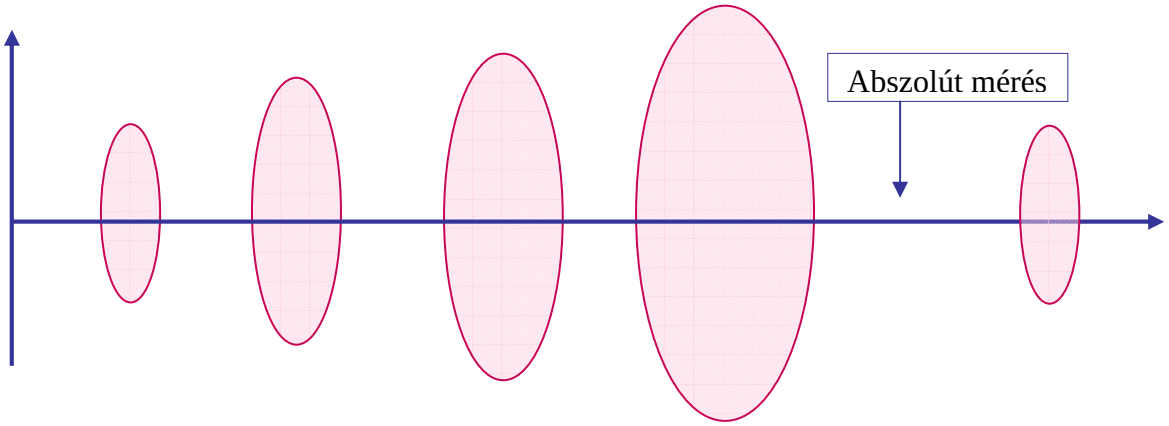
Rendszeres hibák:

- A kerekek átmérője különböző
- A kerekek tényleges átmérőjének átlaga különbözik a kerekek névleges átmérőjének átlagától
- A kerekek valós távolsága különbözik a névlegestől
- A kerekek nem egy vonalban helyezkednek el
- A motorok valós teljesítmény adatai eltérnek a névlegestől
- Véges jeladó felbontás/mintavételezési ráta

Rendszertelen hibák:

- Csúszós út miatt a kerekek ugyan állnak de a robot mozog, vagy a kerekek mozognak és a robot ál (kipörgés, vagy beássa magát), nagy sebességnél a robot továbbcsúszik, fordulásnál sodródik, hathatnak rá a környezetéből tárgyak valaminek nekimegy, vagy belső kényszerek (pl.: egy beragadó támasztó kerék)
- Pontszerű kapcsolat a talajjal
- Egyenetlen úttest
- Váratlan akadályon való áthajtás

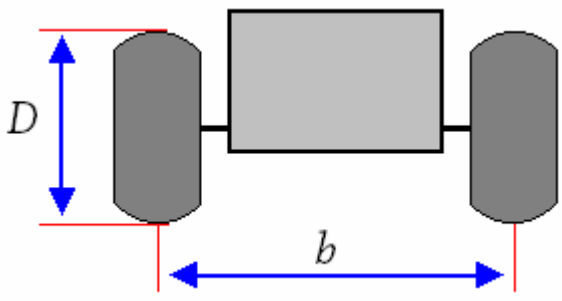
Mint arról már esett némi szó, az odometriai hibák összeadódnak, a lenti ábrán a hibaellipszis mutatja ezt, a megtett úttal nő a hibaellipszis nagysága. Ez csökkenthető abszolút pozícióméréssel(8).



5.6. hiba ellipszis az x tengely a megtett utat mutatja az y a hiba mértékét

5.5 Hibajavítás

A hibák hatásának figyelembevételére célszerű bevezetni a tényleges és valódi mennyiségek hányadosait, mivel sok hiba adódik abból, hogy a névleges érték eltér a valóditól. Ezeket a szisztematikus hibákat kívánjuk csökkenteni a lenti kalibrációs módszer segítségével. Legyen $D_{névl}$ a névleges átlagos kerékátmérő, és $D_{ténył}$ a kerekek átlagos átmérőjének valódi értéke. A két mennyiség hányadosa $E_s=D_{névl}/D_{ténył}$. A két kerék tényleges mérete is különbözhet, tehát a két kerék nem egyforma.A tényleges átmérők hányadosa legyen $E_d=D_R/D_L$. További hibát okoz a tényleges és névleges tengelytávolság eltérése, ezt jellemezze $Eb=b_{ténył}/b_{névl}$. Ahol *D* a kerekek átmérője, *b* pedig a kerekek távolsága(8).



5.7. D a kerékátmérő, b a kerekek távolsága

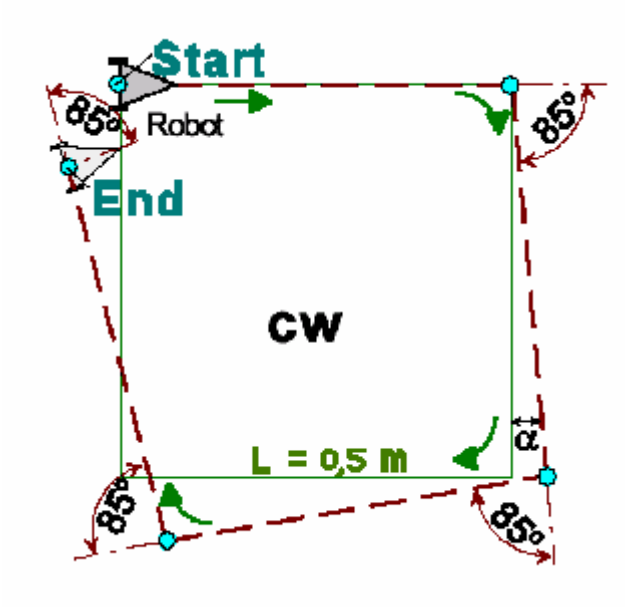
5.5.1 A hibakalibrálás elmélete

A hibák meghatározására egy négyzetet járunk be a robottal mind a két irányba. A bejárás során ideálisan a robot pont 0,5 métert halad előre és 90°-ot fordul egyhelyben. A négyzetet többször járjuk be mind a két irányba, és meghatározzuk a pozíció és szög eltéréseket a kezdeti és végállapotok között.

A fellépő hibák két csoportba oszthatók. A tengelytávolság pontatlan ismerete esetén a robot többet vagy kevesebbet fordul, így mindkét irányban kisebb vagy nagyobb lesz a teljes elfordulás (A típusú hiba). A kerékméretek eltérése esetén az egyenes szakaszok ívesek lesznek, ami az egyik irányban növeli, a másik irányban körbejárva csökkenti a teljes elfordulást (B típusú hiba).

Az A típusú hiba által okozott eltérés mindkét körüljárásnál azonos, a B típusú hiba a kétféle körüljárásra ellentétes eltérést eredményez.

A továbbiakban feltételezzük, hogy a szisztematikus eltérések kicsik, azaz elsőrendű közelítő formulákat lehet alkalmazni, továbbá a kétféle hibát egymástól függetlenül kezelhetjük. Az A típusú hiba túl kicsi, vagy túl nagy elfordulásokat eredményez a sarkoknál. Legyen ez a szögeltérés alpha. Ekkor a négyszög oldalainak bejárása során fellépő eltérések(8):



5.8. Óramutató járásával megegyező irányú bejárás

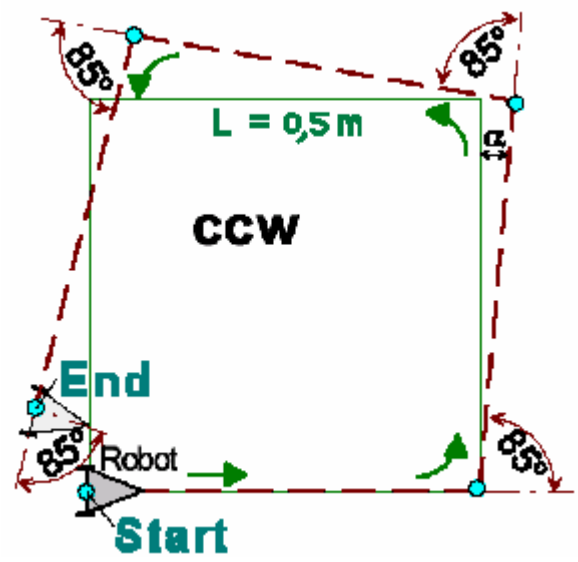
$$x_1 = x_0 + L$$
$$y_1 = y_0$$

$$x_2 = x_1 + L * \sin(\alpha) \approx L - L * \alpha$$
$$y_2 = y_1 + L * \cos(\alpha) \approx -L$$

$$x_3 = x_2 - L * \cos(2 * \alpha) \approx L * \alpha$$
$$y_3 = y_2 - L * \sin(2 * \alpha) \approx -L - 2 * L * \alpha$$

$$x_4 = x_3 - L * \sin(3 * \alpha) \approx -2 * L * \alpha$$
$$y_4 = y_3 + L * \cos(3 * \alpha) \approx -2 * L * \alpha$$

Az ellentétes irányú bejárás során:



5.9. Óramutató járásával ellentétes irányú bejárás

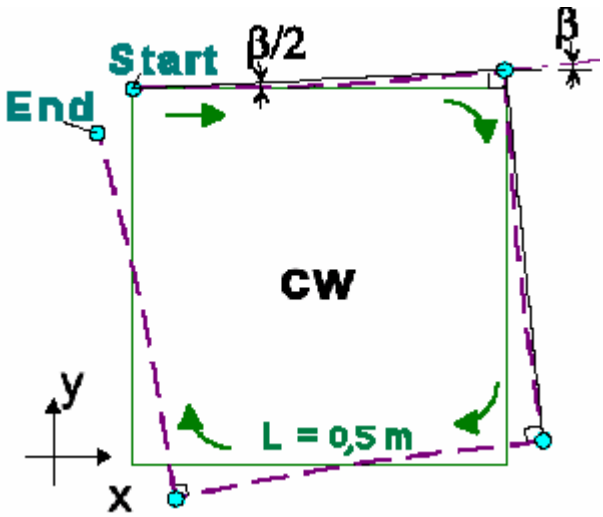
$$x_1 = x_0 + L$$
$$y_1 = y_0$$

$$x_2 = x_1 + L * \sin(\alpha) \approx L + L * \alpha$$
$$y_2 = y_1 + L * \cos(\alpha) \approx +L$$

$$x_3 = x_2 - L * \cos(2 * \alpha) \approx L * \alpha$$
$$y_3 = y_2 + L * \sin(2 * \alpha) \approx L + 2 * L * \alpha$$

$$x_4 = x_3 - L * \sin(3 * \alpha) \approx -2 * L * \alpha$$
$$y_4 = y_3 - L * \cos(3 * \alpha) \approx 2 * L * \alpha$$

A B típusú hibát a kerékátmérők eltérése okozza, ami ív menti mozgást eredményez és minden szakasz végén egy béta szögelforduláshoz vezet. A négyszög oldalainak bejárása során fellépő eltérések, ekkor:



5.10. Óramutató járásával megegyező irányú bejárás

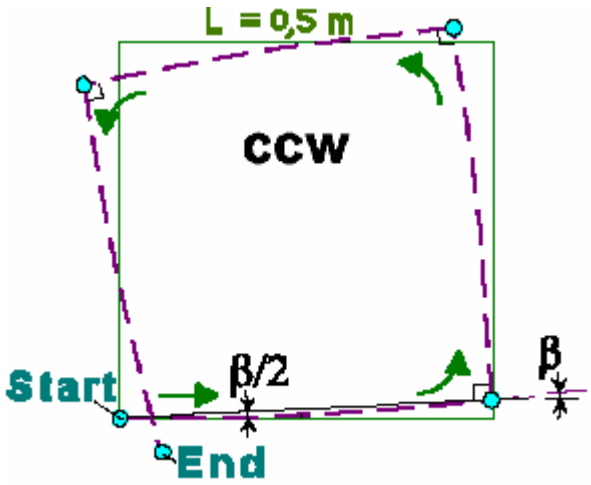
$$x_1 = x_0 + L * \cos(\beta/2) \approx L$$
$$y_1 = y_0 + L * \sin(\beta/2) \approx L * \beta/2$$

$$x_2 = x_1 + L * \sin(3 * \beta/2) \approx L + 3 * L * \beta/2$$
$$y_2 = y_1 - L * \cos(3 * \beta/2) \approx L * \beta/2 - L$$

$$x_3 = x_2 - L * \cos(5 * \beta/2) \approx 3 * L * \beta/2$$
$$y_3 = y_2 - L * \sin(5 * \beta/2) \approx -L * (2 * \beta + 1)$$

$$x_4 = x_3 - L * \sin(7 * \beta/2) \approx -2 * L * \beta$$
$$y_4 = y_3 + L * \cos(7 * \beta/2) \approx -2 * L * \beta$$

Az óramutató járásával ellentétes irányú körbejárás után felhalmozódó hibák:



5.11. Óramutató járásával megegyező irányú bejárás

$$x_1 = x_0 + L * \cos(\beta / 2) \approx L$$
$$y_1 = y_0 + L * \sin(\beta / 2) \approx L * \beta / 2$$

$$x_2 = x_1 + L * \sin(3 * \beta / 2) \approx L - 3 * L * \beta / 2$$
$$y_2 = y_1 + L * \cos(3 * \beta / 2) \approx L * \beta / 2 + L$$

$$x_3 = x_2 - L * \cos(5 * \beta / 2) \approx -3 * L * \beta / 2$$
$$y_3 = y_2 - L * \sin(5 * \beta / 2) \approx -2 * L * \beta + L$$

$$x_4 = x_3 + L * \sin(7 * \beta / 2) \approx 2 * L * \beta$$
$$y_4 = y_3 - L * \cos(7 * \beta / 2) \approx -2 * L * \beta$$

A kétféle hiba együttes fellépése esetén, az x irányú eltérésekre kapható egyenletek:

$$x_{cw} : -2 * L * \alpha - 2 * L * \beta = -2 * L * (\alpha + \beta) = x_{a,cw}$$
$$x_{ccw} : -2 * L * \alpha + 2 * L * \beta = -2 * L * (\alpha - \beta) = x_{a,ccw}$$

Ha ezt a két egyenletet összeadjuk, illetve kivonjuk egymásból, megkapjuk az eltérést jellemző két szöget:

$$\beta = \frac{x_{a,cw} - x_{a,ccw}}{-4 * L} * \frac{180}{\pi}$$

illetve

$$\alpha = \frac{x_{a,cw} + x_{a,ccw}}{-4 * L} * \frac{180}{\pi}$$

Továbbá mind a két mennyiség kifejezhető az y irányú eltérések összehasonlításával is:

$$\beta = \frac{y_{a,cw} + y_{a,ccw}}{-4 * L} * \frac{180}{\pi}$$

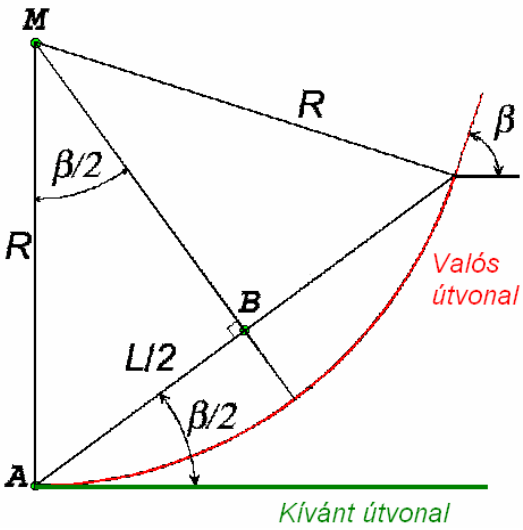
illetve

$$\alpha = \frac{y_{a,cw} - y_{a,ccw}}{-4 * L} * \frac{180}{\pi}$$

A fenti számolás révén a távolságok segítségével határozhattuk meg a fellépő szögeltéréseket. Mivel a hibák kicsik, ez jóval pontosabban meghatározható mintha az eredményt közvetlenül a kezdeti és végső orientáció eltéréséből próbálnánk meg kiértékelni.

A kapott szögeltérések alapján hátra van még a két hibát jellemző arányszám meghatározása. A béta szögelfordulás és az L távolság segítségével kiszámítható az íves pálya görbületi sugara fokban:

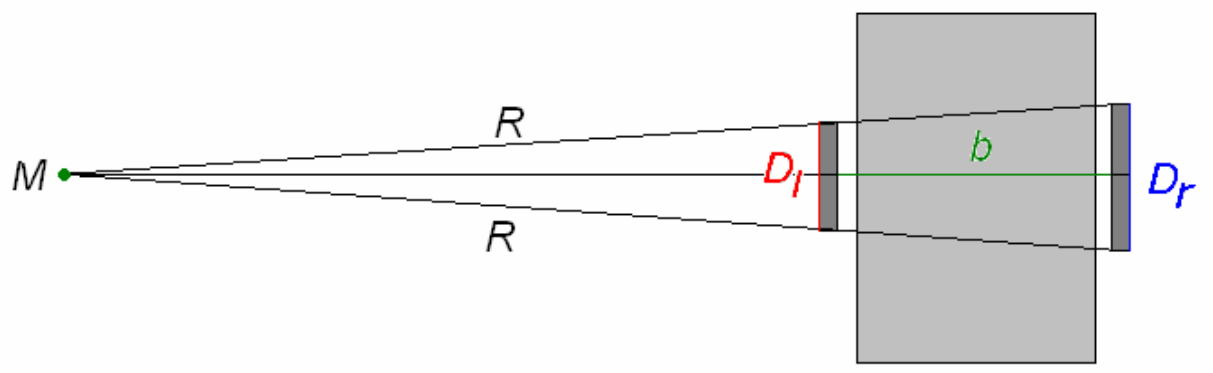
$$R = \frac{L/2}{\sin(\beta/2) * (\pi/180)}$$



5.12. A különböző kerékátmérők, vagy a különböző erősségű motorok miatt a robot nem tud egyenes pályán mozogni.

A görbületi sugár segítségével könnyű megadni a kerékátmérők arányát:

$$E_d = \frac{D_R}{D_L} = \frac{R + b/2}{R - b/2}$$



5.13. Különböző kerékátmérő

A névleges és aktuális keréktávolságok arányosak a megfelelő szögelfordulásokkal:

$$E_b = b_{actual} / b_{do\,\min al}$$

azaz

$$E_b = 90/(90 - \alpha)$$

5.5.2 Legó robot kalibrálása a képfeldolgozó programmal

A kísérlet a fent leírtakhoz igazodik. Az 5.3-as fejezetben leírt képletekkel és az ismertetett Java kód segítségével. Egy 0,5 méteres oldalhosszú négyzeten küldtem körbe a LEGO robotot, mind a két irányba. A robot geometriai felépítése tükrözi a már 5.1-es fejezetben leírt differenciálhajtásos módszert.

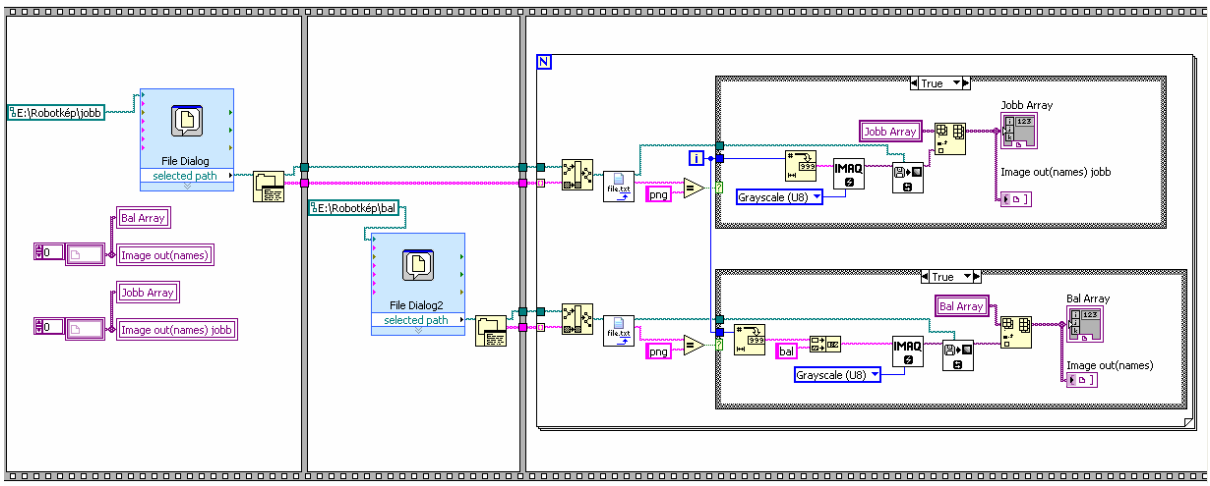
Meghatároztam egy kezdeti pozíciót amelyet vehetünk origónak, a robotot minden kör után ide helyeztem vissza. Majd készítettem egy képet az eredeti helyzetéről. Mind a két irányba 5-ször ment körbe az egység. Miután megérkezett mindig képet készítettem az adott pozíciójáról, összesen tehát 11 kép készült, egy referencia és 5-5 a végállapotokról. Ezután jött a tényleges munka, írtam egy képfeldolgozó programot, ami egy mintaillesztéses módszerrel megadta a pozíció eltéréseket és a szögelfordulásokat, majd ezekből korrekciós számításokat végzett. A korrekciókkal javított, Java kóddal újra elvégeztem ezt a procedúrát, az eredmény az 5.5.2.2-es fejezetben olvasható.

5.5.2.1 A képfeldolgozó program

A program a National Instruments által gyártott, Labview Imaq Visionben készült. A Vision széles tárházát kínálja a képfeldolgozási eszközöknek, csak meg kell találni a megfelelő VI-okat.

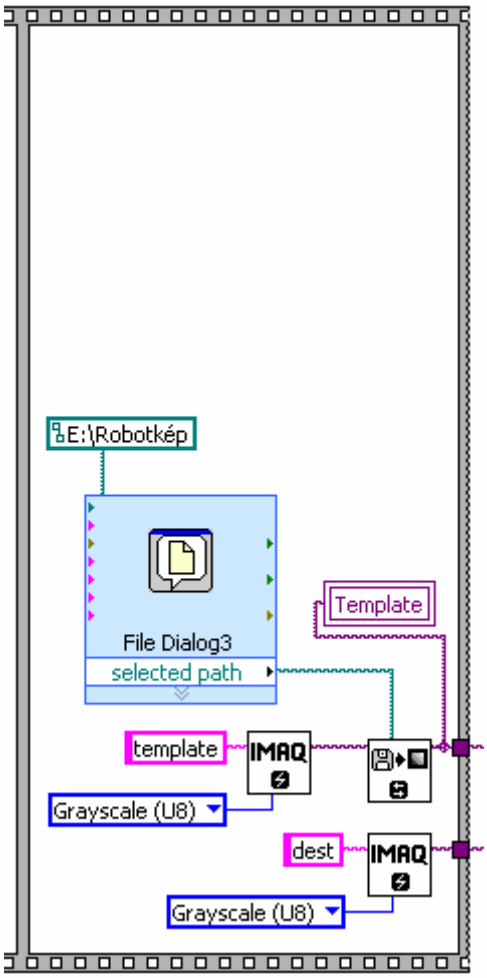
Pár szó a Labviewról (10):

A LabVIEW egy grafikus fejlesztő eszköz, amely egy adatfolyam nyelvre épül, melynek a neve G. A LabVIEW külön érdekessége, hogy benne nem szövegalapú forráskódot írunk, hanem egy úgynevezett blokk diagramot rajzolunk. Ez lesz a „forráskód”. Ezt az eszközt kifejezetten mérnökök (méghozzá villamosmérnökök) számára mérési, automatizálási és folyamatirányítási célokra fejlesztették ki. Segítségével úgynevezett virtuális műszereket (Virtual Instruments röviden: VI) hozhatunk létre. A virtuális műszer egy olyan program, amely egy fizikai műszer, külső megjelenését és működését modellezi.



5.14. Flat Sequence

A program egy Flat Sequence-el kezdődik. Az első ''filmkocka'' tartalmazza két képeket tartalmazó tömb inicializálását, valamint egy File Dialogot, ami úgy van beállítva, hogy egy teljes könyvtárat beolvasson, a második rész szintén egy könyvtárat olvas be. A harmadik részében, az elején kinullázott tömböt tölti fel a File Dialog által megnyitott könyvtárakból, ezek lesznek azok a képek amik a robot elmozdulás utáni végállapotait tartalmazza. A könyvtárban lévő fájlokat egyenként megvizsgálja, egy for ciklus segítségével, és ha a kiterjesztésük .png, akkor a ciklusváltozóból generált névvel a képeket eltárolja a tömbbe, miközben lefoglal nekik egy 8 bites szürkeárnyalatos képnek tárterületet.

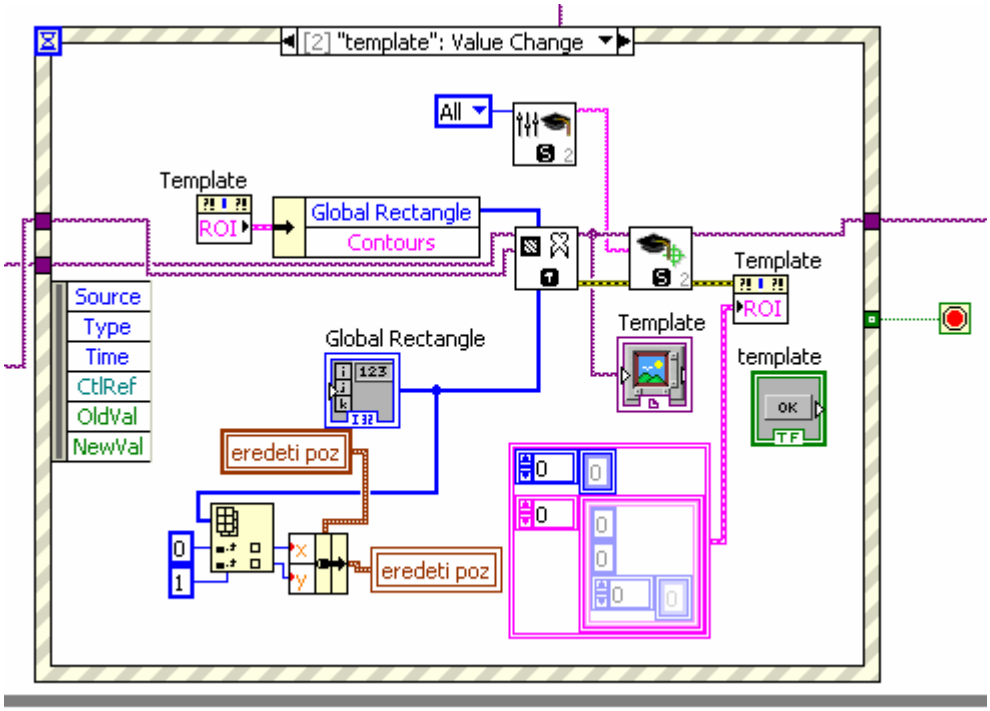


5.15. Flat Sequence vége

Ez ennek a Flat Sequencnek az utolsó eleme, Itt a File Dialog úgy van beállítva, hogy csak egyetlen egy fájlt nyisson meg, itt is van két tárterület foglalás, az egyik a Template képnek, ami a mintaillesztés alapjául fog szolgálni, egy pedig egy számoló terület (dest).

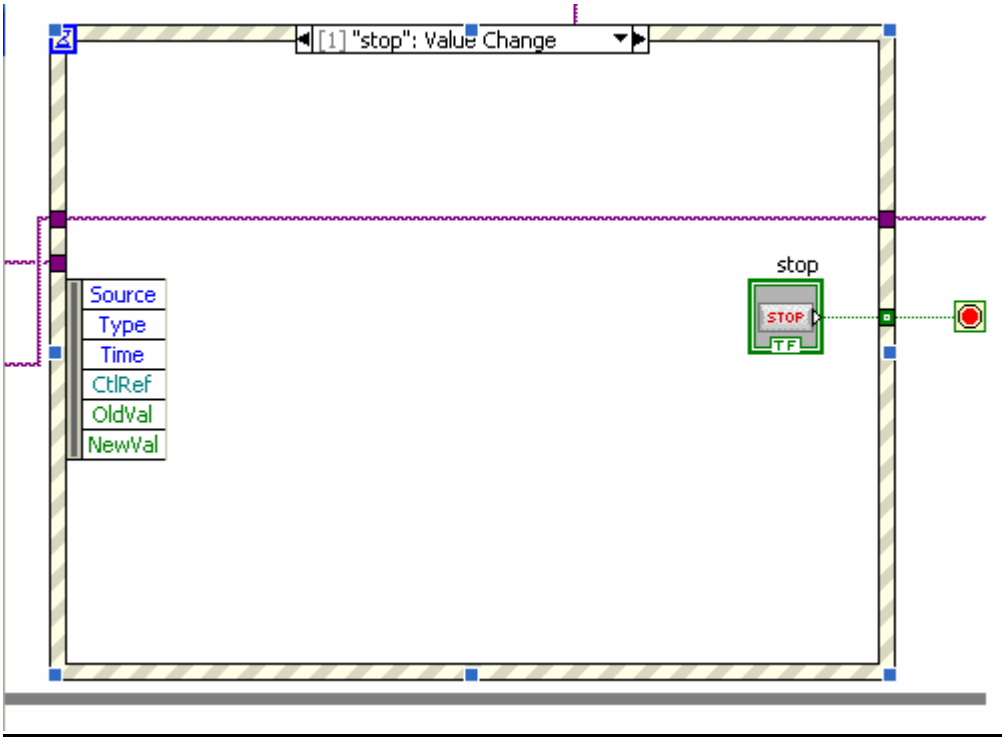
A Flat Sequence jelentősége az, hogy így tudom szabályozni mi, mi után fusson le, ha nem lenne benne nem tudom milyen sorrendben, indulnának el a képbetöltések.

Ezután az egész program egy while cikluson belül helyezkedik el, ami tartalmaz egy másik Flat Sequencet, meg egy Event Structuret-t. Az Event Structure 3 ''case'' ágból épül fel.



5.16. Event Case

A legfontosabb eleme ez az ág, itt a harmadik File Dialog által megnyitott képen kijelölt templatet, az Image Display ROI-ján keresztül, ''megtanítatjuk'' a programnak. A ROI-ból leválogatjuk a Global Rectangle elemet, ami a kijelölés koordinátáját tartalmazza. Közben az ''eredeti poz'' Clusterben eltárolom a kezdő koordinátákat, amit majd a Globális koordináta rendszerem origójaként fogok használni. Ezek a folyamatok a Font Panelen lévő ''Template'' gomb lenyomásával indulnak meg, ez a szerkezet szolgál a Labviewban az eseménykezelésre. Az IMAQ Extract VI, a templatemet beleteszi a dest lefoglalt tárhelybe, és ezt adja tovább ''megtanulni'' az IMAQ Learn Pattern 2-nek. Ha nem lenne kijelölve template, tehát a ROI-ben a Global Rectangle üres lenne akkor hiba fut végig a VI között a sárga-fekete vezetéken, és a végén a ROI-t kinullázza, hiba nem történik csak nem ad vissza eredményeket a program.

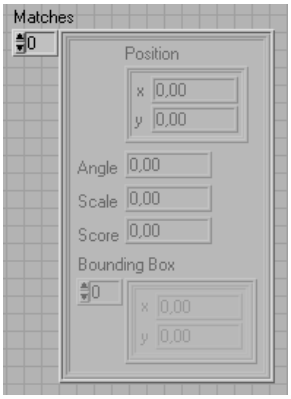


5.17. stop ág

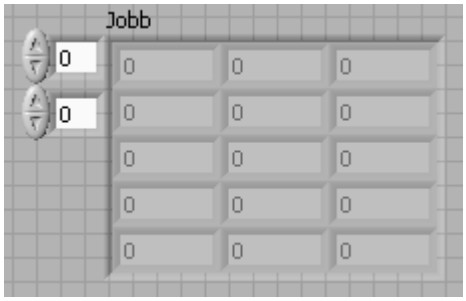
A másik két case ágban, nem sok minden található, a timeout ágat illik is tisztán hagyni, mert az mindig fut, amikor nem történik esemény. A stop gombot, a front panelen ezzel állítható le a program, pedig azért tettem be egy case ágba (5.17. kép) mert így gyorsabban érzékeli a lenyomását a program és jobban működik. Itt is és a timeout ágban is a ''dest'' tárhely van átkötve a template kimenethez, hogy az a kiválasztás után ne változzon.

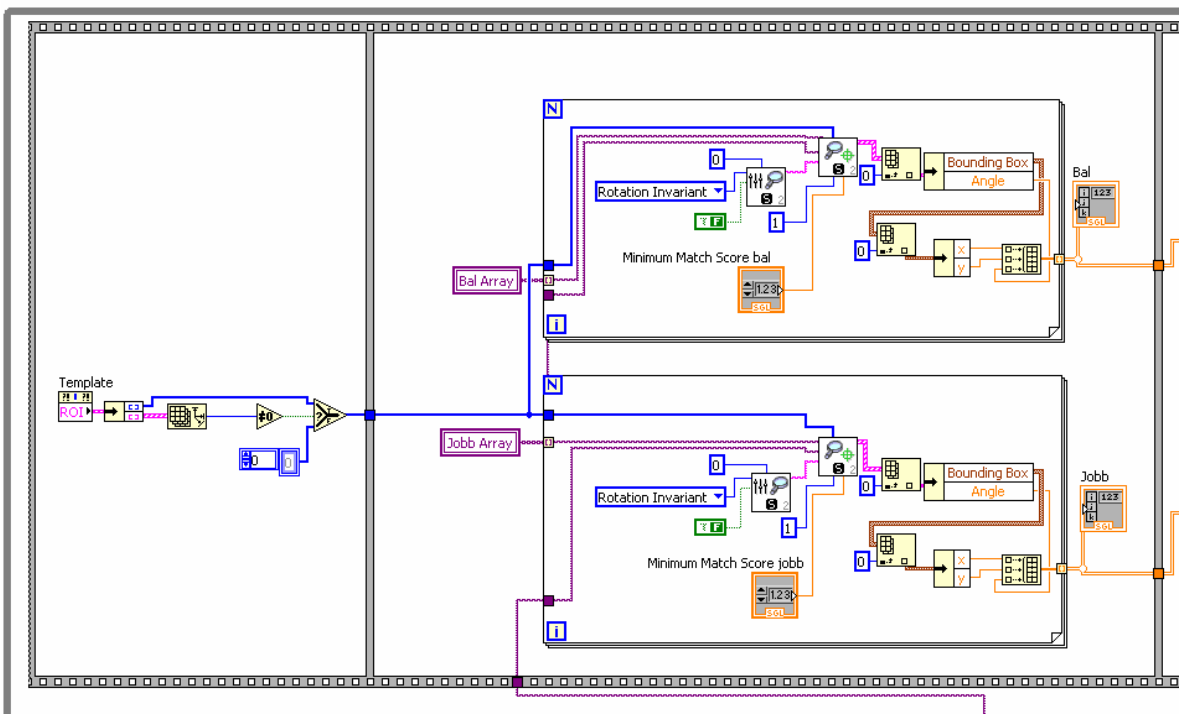
A cikluson belül található még egy Flat Sequence, aminek első két kockája a mintaillesztését végzi el, és a maches cluster tömbből csinál nekem egy egyszerű két dimenziós tömböt, aminek első két oszlopában a pozíció valamint a harmadik oszlopban a szögeltérés szerepel. Itt található az IMAQ Match Pattern 2 ami egy for ciklusba ágyazva a template -> ROI -> Global Rectangel-éből, a képek tömbön végighaladva a template kép segítségével megkeresi az egyezéseket. Ehhez szüksége van még az IMAQ Setup Match Pattern 2-re, ami beállítja milyen módon, milyen pontosan keressen, itt elforgatva és eltolva keres. Valamint szükséges a Match Patternnek megadni, hogy mennyire tökéletesen egyezzen a minta és hány egyezést keressen.

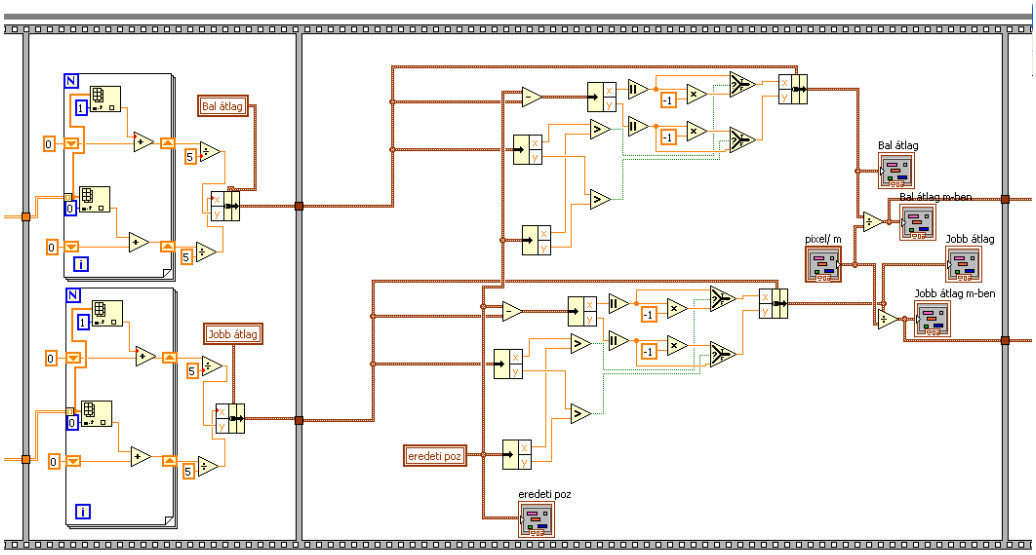
5.18. Matches cluster



5.19. Két dimenziós tömb

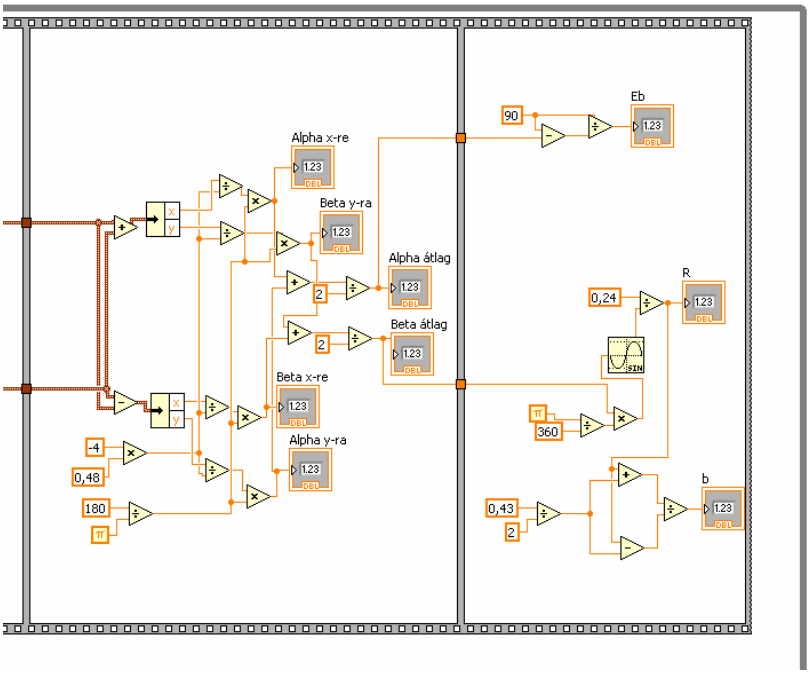






5.20. átlagok

A Flat Sequence következő kockájában, az 5-5 mérés átlagát számolom ki, majd clasterbe rendezem őket. Mivel a pixelek (0,0) pontja vagyis az origo a bal felső sarok, a következő kockában kénytelen vagyok az origót a robotra (template) helyezni és átszámolni ennek megfelelően az eredményeket és előjeleket, majd egy arányszámmal a pixelt méterré átváltani.



5.21. korrekciók

A következő két kockában, pedig a fenti számolásokat írtam be a programba, így az automatikusan kidobja a korrekció értékeket.

5.5.2.2 A módszer eredményessége

A lentebb látható táblázatokon és diagrammokon szeretném megmutatni a módszer hatékonyságát. Az első (5.1.) táblázat tartalmazza az első 10 mérés (5-5 minden irányban), végállapotainak koordinátáját és szögeltérését. Minden méterben illetve fokban van. A cw jelentése óramutatóval megegyező irányú a ccw óramutatóval ellentétes irányú. A 300 körüli szögeltérések igazából 300-360 fokok, de a program óramutató járásának megfelelően forgatta körbe a templatet, így egyértelműbb, ezért így hagytam. A legkisebb pozíció eltérés 7,3 cm a legnagyobb 15,4 cm egy tengelyen. A legkisebb szögeltérés 13 fok a legnagyobb 23,1 fok.

Korrekció előtt			
kép	x (m)	y (m)	szög (fok)
ccw1	0,094525	0,107011	18,0907
ccw2	0,094482	0,103045	17,0176
ccw3	0,080392	0,09574	15,9338
ccw4	0,073029	0,092943	13,0305
ccw5	0,07496	0,091607	16,4171
cw1	0,123965	-0,12046	339,705
cw2	0,152286	-0,11806	337,54
cw3	0,147047	-0,11494	336,938
cw4	0,153587	-0,11636	336,869
cw5	0,134867	-0,10682	338,048

5.1. Táblázat

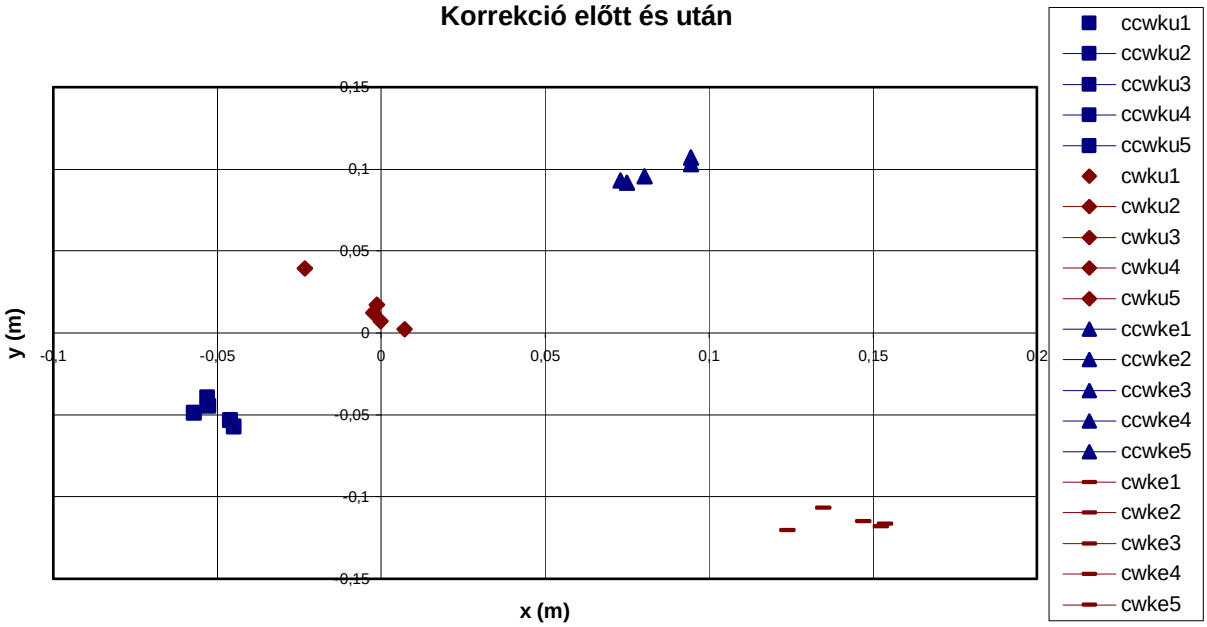
A diagramon jól látszik a szórás. Az eltérések 0,1 tized méter környékiek, tehát olyan 10 centiméter.

A korrekció után egy kicsit változatosabb a kép, nem olyan egyértelműen a koordináta rendszer egyes negyedébe esnek az azonos irányú végállapotok és a szögeltérések sem olyan agyhangúak. Itt több olyan van, ahol csak milliméteres, vagy még kisebb az eltérés. A legkisebb pozíció eltérés 0,009 cm a legnagyobb 5,73 cm. A legkisebb szögeltérés 0,32 fok a legnagyobb 9,9 fok.

Korrekció után			
kép	x	y	szög
ccw1	-0,05268	-0,04471	352,519
ccw2	-0,04488	-0,05727	350,16
ccw3	-0,05294	-0,03947	351,748
ccw4	-0,04601	-0,05335	350,286
ccw5	-0,05706	-0,04875	350,095
cw1	-0,00122	0,01739	2,2525
cw2	-0,00244	0,012521	1,78211
cw3	0,007214	0,002199	359,676
cw4	-0,000091	0,007181	2,3916
cw5	-0,02332	0,039342	3,2554

5.2. Táblázat

Az itt kapott eredmények sokkal jobbak, tehát a korrekciók sikerültek.

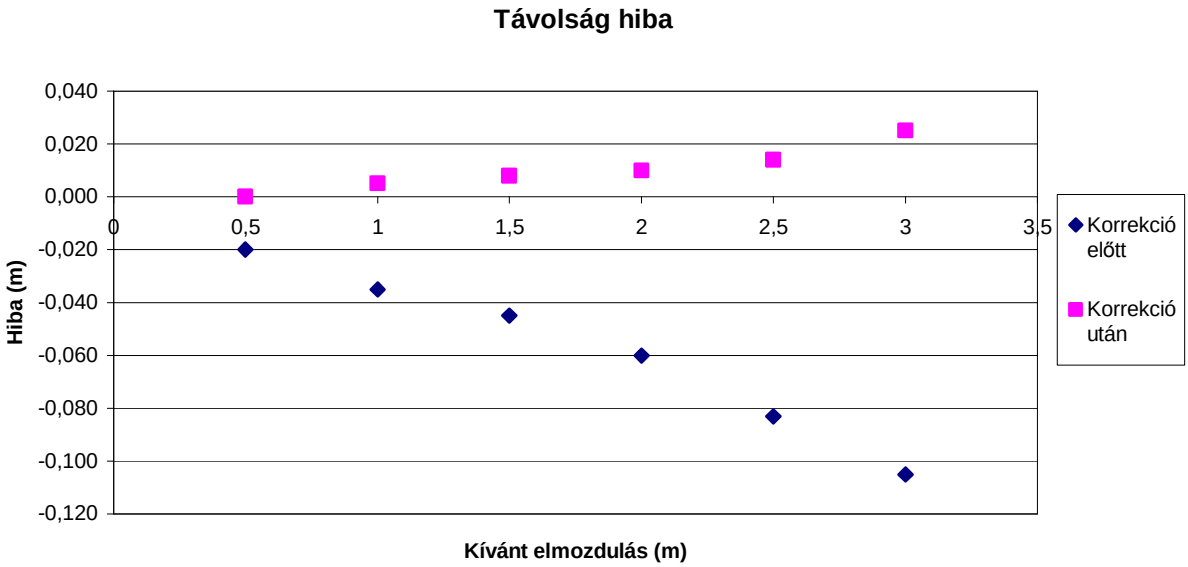


5.1. Diagram

Van egy további hiba amit sajnos csak későn vettem észre, ez pedig az hogy robotom a megadott ”l” távolságnál az előremozdulás számításához használt képletben kapott értékhez képest kevesebbet megy. Alul láthatóak a hibaadatok, a hiba lineárisan nő a távolság függvényében. A kísérletben persze a valós adatokat használtam nem pedig az idealizáltat.

Korrekció előtt						
m	0,5	1	1,5	2	2,5	3
hiba (m)	-0,020	-0,035	-0,045	-0,060	-0,083	-0,105
Korrekció után						
m	0,5	1	1,5	2	2,5	3
hiba (m)	0,000	0,005	0,008	0,010	0,014	0,025

5.3. Táblázat



5.3. Diagram

A korrekciós képlet a következő levezetéssel együtt. Először az eredeti képlet amivel a távolságot szögelfordulásra váltjuk, itt a névleges sugarat használjuk:

$$\varphi = \frac{180}{r_n * \pi} * l$$

A tényleges sugárral visszaszámolható az elmozdulás valódi értéke:

$$l' = \frac{r_t * \pi}{180} * \varphi$$

A második egyenletbe behelyettesítjük az elsőt:

$$\frac{180}{r_n * \pi} * l * \frac{r_t * \pi}{180} = l'$$

Az egyszerűsítés után:

$$\frac{r_t}{r_n} = \frac{l'}{l}$$

r_t-t kifejezve:

$$r_t = r_n * \frac{l'}{l}$$

6. Összefoglalás

A négyszögkísérletek elvégzése nagyon sikeresnek bizonyult. Főleg hogy automata rendszert sikerült összeállítani, mivel a kiértékeléseket egy képfeldolgozó program végzi, nekem a korrekció értékeket csak be kellett írni a Java kódba és az elmozdulások egyből pontosabbak lettek.

Összességében az elmúlt egy év alatt míg a szakdolgozatomat írtam, sokat tanultam a robotok navigációs módszereiről. Az itt leírt kalibrációs módszerrel és a készített programmal, bármilyen a mostanihoz hasonló, geometriai felépítésű robot odometriai kalibrálása elvégezhető. Ha kivesszük a képletből a Lego-t és valami komolyabb platformmal foglalkozunk, akkor a hibajavítás még sikeresebb is lehet. Valamint apróbb változtatásokkal más felépítésű robotokra is alkalmazható (pl.: 3 vagy 4 kerekűekre).

A szakdolgozatban kitűzött célt sikerült elérni és az elméletben működő módszert a gyakorlatba átültetni. Önmagában ez kevés egy felderítő robothoz, de jó alapot képez és várom, hogy az életben is kipróbálhassam ezt a tudást valamilyen formában.

7. Irodalomjegyzék:

1. Kömlődi Ferenc – Autonóm Mobil Robotok
   http://www.google.hu/url?sa=t&source=web&ct=res&cd=3&ved=0CBAQFjAC&url=http%3A%2F%2Fwww.nhit.hu%2Fdata%2F101419%2Fmobilrobotika3.2.doc&ei=lpwCS9zqEKaqmwO4kr16&usg=AFQjCNFwDjqkyrzM1n4QBXTo8G5I7d36aQ&sig2=L8yzOaX-mymFeK24Oda66A
2. Molnár András – Kutató robotok
   http://www.zmne.hu/tanszekek/ehc/konferencia/april2001/molnar.html
3. Magyarósi Csaba – Vegetáriánus robot a jövő hadseregében (Index)
   http://index.hu/tech/hardver/2009/07/19/vegetarianus_robot_a_jovo_hadseregeben/
4. Tornyi Molnár Zoltán – Robotok
   http://www.google.hu/url?sa=t&source=web&ct=res&cd=1&ved=0CAkQFjAA&url=http%3A%2F%2Fszamtechklub.extra.hu%2Fdownload%2Frobotok.pps&ei=zZsCS6qDIYzWmgOt7ulx&usg=AFQjCNFZ2UNDAoBn50NlX7ICIsR8jUp1rg&sig2=M6aicZ6yaCSwRLv3VPZF4A
5. http://marsrovers.nasa.gov
6. SG.hu – Humanoid robotok segíthetik az űrsétákat
   http://www.sg.hu/cikkek/28204/humanoid_robotok_segithetik_az_ursetakat
7. Husi Géza – Ipari robotok.ppt (oktatási segédanyag)
8. Szatmári István – Robotika.ppt
9. Wikipedia – online enciklopédia
   http://hu.wikipedia.org/
10. Disznós Imre - VIRTUÁLIS MŰSZEREK ÉS A LABVIEW
    http://docs.google.com/gview?a=v&q=cache:CBiaj3tW4eEJ:www.muszeroldal.hu/measurenotes/virtualis_muszerek.pdf+Diszn%C3%B3s+Imre+-+VIRTU%C3%81LIS+M%C5%B0SZEREK+%C3%89S+A+LABVIEW&hl=hu&gl=hu&sig=AFQjCNFuP4lCdONK4hXfzCKzpWO_skDIMA