

**DEBRECENI EGYETEM
GAZDÁLKODÁSTUDOMÁNYI ÉS VIDÉKFEJLESZTÉSI KAR**

TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

**MEZŐGAZDASÁGI ROBOT FEJLESZTÉSE ÉS JÖVŐBELI
BŐVÍTHETŐSÉGE**

**DEVELOPMENT AND FUTURE EXPANDABILITY OF AN
AGRICULTURAL ROBOT**

Tóth Mihály
Informatikus és szakigazgatási agrármérnöki BSc
III. évfolyam

Konzulens:
Dr. Szilágyi Róbert
egyetemi adjunktus

DEBRECEN
2014. ÁPRILIS 14

Tartalomjegyzék

Bevezetés	3
1. Témafelvetés	5
2. Szakirodalmi áttekintés	7
2.1. Lego Mindstorms.....	10
3. Anyag és módszer.....	14
4. Eredmények és azok értékelése.....	16
4.1. Meghajtás és irányítás.....	17
4.1.1. Felfüggesztés	17
4.1.2. Terepakadály kikerülési képesség (NXT-G)	19
4.1.3. Terepakadály kikerülési képesség (Robot-C).....	22
4.1.4. GPS vezérlés.....	22
4.2. Microsoft Kinect segítségével történő vezérlés	24
4.3. Felhasznált szenzorok	26
4.3.1. NXT rendszerhez mellékelt szenzorok.....	26
4.3.2. Külső gyártók által elérhető szenzorok és kiegészítők.....	28
4.4. Adatgyűjtés	28
5. Következtetések és javaslatok	31
5.1. Fejlesztési lehetőségek.....	31
5.2. Arduino rendszerek és vele kapcsolatos kiegészítők	34
5.3. Az Arduino platform előnyei és hátrányai.....	37
5.4. Felhasználási lehetőségek a jövőben.....	39
5.5. Költségekalkuláció	40
Összefoglalás	43
Irodalomjegyzék	46
Mellékletek	48

Bevezetés

Napjainkban szinte természetessé vált a számítástechnika jelenléte. Utunk során szinte mindenhol találkozunk informatikai eszközökkel, így nyugodtan kijelenthetjük, hogy az informatika világát éljük. Ebből kifolyólag az ilyen eszközök felhasználói szintű kezelése a mai generációnak nem jár különösebb kihívással. A technológiának köszönhetően munkánk során észrevehető az automatizálás által adódó könnyebb teljesíthetőség az adott folyamatok tekintetében. De az automatizálásnak emellett számos más előnye is lehet.

A munkavégzés felgyorsítása mellett e technika révén gazdaságosabban, precízebben és humán viszonylatban nézve kíméletesebben érhetjük el célkitűzésünket. Az informatika ilyen célú felhasználása esetén először legtöbbször az irodai felhasználásra, az ehhez kapcsolódó irodai programcsomagok által nyújtott lehetőségekre, különböző adatbázis kezelőkre, és internetes segédalkalmazásokra gondolnak. A sort természetesen rengeteg példával bővíthetnénk, hiszen az informatikának szinte kifogyhatatlan, ehhez hasonló lehetőségei adódnak, de véleményem szerint még kevesen gondolunk elsők között a most kifejtésre kerülő témára. Ennek oka lehet az, hogy még nem olyan régen köztudatban lévő, illetve technikailag még nem teljes mértékben kiforrott ágazatra lépünk, ami az informatika mezőgazdasági célokra történő felhasználása. Sajnos még kevesen használják ki az ezen eszközök által nyújtott pozitívumokat a termelés folyamata során annak ellenére, hogy a hagyományos gazdálkodás mellett e technika segítségével ideális esetben és megfelelő használat esetén optimálisabb eredmény érhető el.

Ha a klasszikus értelemben vett informatikai ágazatokat vizsgáljuk meg, már régebb óta találkozhatunk a piacon különböző mezőgazdasági nyilvántartó és GIS (Geographic Information Systems) rendszerekkel, melyek egyértelműen pozitívan járulnak hozzá az effektivitáshoz. De ha nem csak szoftver, hanem hardver tekintetben is foglalkozunk a lehetőségekkel, számos mezőgazdaságot segítő technológia jelenlétét figyelhetjük meg. Szinte bármelyik korszerű agrotechnikai eszközt vizsgáljuk meg, biztosak lehetünk benne, hogy az tartalmaz valamilyen, a feladatának elvégzését segítő számítástechnikai eszközt, legyen az hardver vagy szoftver szinten. Talán legegyszerűbb példa erre az egyes traktorokban található GPS vevőkészülék, melynek segítségével a vezető precízen képes a nyomvonalat tartani a változó körülmények között, mint például a talajegyenetlenségek és a rossz látási viszonyok, ezzel pedig könnyítve a további munkálatokat és a betakarítást. Erre példa a Centerline 220 GPS készülék (Forrás: I1). De számos típust említhetnénk, amely hasonló célt szolgál.

Egyes vetőgépek és műtrágyaszórók főbb paraméterei egy, a vezetőfülkébe beszerelhető beltéri egységgel módosíthatóak, tehát ezen beállítások módosításai nem igényelnek külső, mechanikus beavatkozást. A gyártó által tesztelt beállítási paraméterek mellett dolgozhatunk, így nincs szükség az egyes esetekben „analóg” beállítások bizonytalanságára hagyatkozni. Emellett a készülék folyamatos ellenőrzés alatt tartja az adott gépet, melynek segítségével időben értesülünk egy esetleges hibajelenségről, vagy éppen a tartályban lévő anyag mennyiségéről (Legyen az műtrágya, vetőmag, vagy permetszer), utántöltés szükségességéről és a kiürülés várható idejéről. A permetezőgépeknél a külső egység általi vezérelhetőség már szinte alapfelszereltség, hiszen a zárt fülkében kisebb az esélye a permetszer légutakba kerülésének, ezzel pedig minimalizálhatjuk az egészségügyi kockázatot.

A felsorolást sok hasonló eszközzel lehetne bővíteni, hiszen lényegében ez mind része a precíziós mezőgazdaság nagy halmazának, de témánk az informatika egy speciális, más irányának felvetését igényli. Ez az irány a robotika.

1. Témafelvetés

A robot egy számítógépes program és egy elektromos áramkör által irányított elektromechanikai szerkezet, amely végrehajtja a megadott feladatát. Általában előre meghatározott céllal készülnek, így a felépítésükkel és programjukkal ehhez alkalmazkodnak. Robotikai eszközök leírásával már első évezredből származó írások tanulmányozása során is találkozhatunk, sőt köztudottan Leonardo da Vinci is foglalkozott egy humanoid robot készítésének kérdésével 1495-ben. Ám az első autonóm elektronikus robot, a Machina speculatrix 1948-ban, William Gray Walter közreműködésével jött létre, Elmer és Elise néven (Forrás: I2).

Egy modern robot eszköz irányítását tekintve több kategóriába sorolható. Lehet autonóm, amelynek működési elveit teljes egészében a rajta tárolt program határozza meg. Szenzorok segítségével tájékozódik és végzi el a feladatát. Tele-vezérelt robot esetén az irányítást az ember és számítógép egyidejűleg végzi, azaz az ember által adott paramétereket képes a környezethez megfelelően alakítani. A harmadik kategória a távvezérelt robotok csoportja, ahol az adott eszköz teljes egészében az ember irányítása alatt áll. Itt gondolhatunk számítógépen keresztüli vezérlésre, rádiófrekvenciás vezérlésre egy távirányító által, vagy akár manapság érdemes megemlíteni az okostelefonok és táblagépek segítségével történő irányítást, melynek közege lehet mobilinternet (Amennyiben hasonló módon rendelkezik az eszköz aktív kapcsolattal), Wifi, illetve Bluetooth. Az alábbi közegek leginkább hatótávolságukban térnek el. Mobilinternet esetén megfelelő térrő jelenlétében gyakorlatilag bárholnan elérhetjük eszközünket. Wifi kapcsolat esetén ez IEEE szabványtól eltérően 100-5000 méter közötti maximális távolságot jelent, ideális jeltovábbítási viszonyokat feltételezve. A Bluetooth kapcsolat pedig osztálytól függően 1 és 100 méteres hatótávot enged meg. A robotokkal való kísérletezés során egyértelművé váltak a technológia előnyei. Ez először talán az ipari robotoknál mutatkozott be a produktivitás növekedése által, de idővel egyre több ágazathoz jelentek meg robot eszközök segítő tényezőként. Ilyen ágazatok például az orvoslás, haderő, bányászat, oktatás, szórakoztatóipar és nem utolsósorban a mezőgazdaság. A mezőgazdasági munkavégzést segítő robotok leginkább két szerkezeti eltérést mutató kategóriába sorolhatóak. Az első kategória a járművel vagy munkaeszkővel szorosan egybeépített robotokat takarja. Ezen eszközök közé tartozik következő sorokban említésre kerülő, traktor irányítását elősegítő dörzskormányos vagy kormányműbe épített szervomotorok, és azokat irányító rendszerek összessége, melyek akár két centiméteres pontossággal is képesek dolgozni. Ilyen rendszert képviselnek a hazai forgalmazásban is lévő

Trimble termékek (I3). A hasonló készülékek első darabjai 1999 kerültek bejelentésre, majd 2000-ben kerültek piacra az AutoFarm GPS 5000 RTK képviselésében (Forrás: I4). Az eszközök a fent említett GPS rendszerek irányításával működnek, megfelelő, specifikus szoftver közreműködésével.

A második és egyben dolgozatunkban részletesebben elemzésre kerülő típusok, a saját vázzal rendelkező robotikai eszközök.

A két típus nem választható el jelentősen egymástól, hiszen lényegüket nézve közel azonos rendszerről van szó, csupán a felhasználási területük tér el. Az előbbi inkább nagy erő igénybevételű feladatokhoz használható, míg az utóbbi precíz feladatokhoz használatos, mint például a mintagyűjtés illetve feltérképezés, vagy megfelelő méretben akár a vetés. A saját vázzal rendelkező robotok használata a mezőgazdaságban még mondhatni gyerekcipőben jár. Az első teljesen autonóm mezőgazdasági robotokkal való kísérletek ezzel kapcsolatban 2009-ben zajlottak az Fuji Heavy Industries képviselésében (Forrás: I5). Több gyártó folytat napjainkban is kutatásai a témával kapcsolatban. Ilyen gyártó például az Amazonen-Werke, a Lely és a gyártók mellett számos egyetem is foglalkozik a kérdéssel, melyek később kerülnek kifejtésre. A mezőgazdasági robotokat megfigyelve tapasztalhatjuk a már korábban említett tulajdonságot, miszerint a robotokat specifikus feladatokhoz készítjük. Nem kivétel ez alól a mezőgazdasági robotok csoportja sem, melyikek több munkavégzési cél szerinti kategóriába sorolhatóak.

2. Szakirodalmi áttekintés

A klasszikus értelemben vett mezőgazdasági robotokkal való kísérletezés és az ezzel járó automatizálás régóta foglalkoztatja az embereket, azok számos előnye miatt.

Az eszközökre való igény már az 1920-as években megmutatkozott, ugyanekkor egy ezzel kapcsolatos alapkoncepció is lefektetésre került. A technológia ekkori szintje természetesen nem engedte meg az eszköz megvalósítását, ám később, 1950-es és 1960-as években elkészült az első vezető nélküli traktor, melynek irányítását egy előre kihúzott, vezetőként szolgáló kábel biztosította. A technika a rengeteg előkészület igénye miatt inkább kísérletként szolgált. Az áttörést az 1980-as évek jelentették, mikor a Michigan State University és a Texas A&M University kutatói az akkor még újdonságnak számító CMOS képalkotó technológia segítségével elkészítették az előző technika mintájára az immáron optikai vezérlésű, autonóm útvonaltartó traktort. Nem sokkal később a University of Florida kutatói sikeresen fejlesztettek narancs betakarítására alkalmas robotot. 1997-től az agrárrobotika nagy érdeklődésnek örvendett potenciális előnyei végett, melybe beletartozik precizitás, fokozott produktivitás és a biztonság (YAGHOUBI, et al., 2013).

Egyes esetekben az eszközre a precíziós mezőgazdaság fogalma vonatkozik, amely adott felbontás alapján részekre bontva tanulmányozza a területet, de egyes esetekben érvényesül a phytotechnológia fogalma (BLACKMORE, 2013), amely egy növényt tekint egy egységnek a területi mértékegységek helyett, ezzel jelentős javulást elérve a pontosságban. Korábban említettem, hogy mint szinte minden robot, mint ahogy az ebbe a témába tartozó gépek is, specifikus céllal jönnek létre. A specializálódás folyamatát jól prezentálja mellékletben feltüntetett 1. ábra (YAEL EDAN, et al., 2011). Az ábra szerint először a szükségeket kell megállapítani, majd ezt összevetve a kívánt céllal, meghatározzuk a problémát. A probléma megoldására meghatározzuk a szükségleteket, majd ezt funkciókra bontva megoldási utakat keresünk a már létező technológiák által. Ekkor egy megoldási struktúrát kapunk, melynek segítségével összeállíthatunk egy rendszertervet. Ez a tervezet pedig a valós eszköz összeállításának az alapja.

BLACKMORE 2013-as prezentációjában tárgyalásra kerültek az agrotechnikai robot általános követelményei, melyek magába foglalják a kis önsúly, kevésbé kötött talajon történő munkavégzés, autonómítás, az időjárás függetlenség, a redundáns rendszer, a kommunikáció, a kiszámítható viselkedés és az időpont független munkavégzés fogalmait. Ugyanez a prezentáció foglalkozik az optimális munkavégzési út keresésének algoritmusával, mellyel a későbbiekben én is foglalkozom több program megírása által.

Saját vázzal, és esetekben meghajtással rendelkező mezőgazdasági gépek kapcsán beszélhetünk többek között automata fejőgépekről, szántóföldi, teljesen automatizált munkagépekről, gyümölcs- és zöldségbetakarító gépekről, vetőgépekről, permetezőgépekről, levél eltávolító gépekről, rózsametsző gépekről, robot terelőgépekről, multispektrális kamerával rendelkező quadcopter-ekről, helyszíni időjárás előrejelző gépekről és ezeken kívül még sok másról.

A klasszikus munkagépek nagy hátránya az időjárástól való függőség. Nagy nedvesség esetén szinte lehetetlen a földterületre hajtani, erős szél esetén pedig lehetetlen a permetezés. Sok eszköz célja az e tényezők fennállásakor jelenlévő korlátok kiküszöbölése, ezáltal az időjárástól független munkavégzés lehetőségének megteremtése.

Az előbb említett lehetőségeket sorra megvizsgálva szembesülhetünk a mai kor eszközeinek előnyeivel, melyeket gyakorlatilag minden ágazaton belül tapasztalhatunk.

A Gantry Robotic System egy kísérleti fázisban lévő, öntözést és gyomlálást elősegítő robot. Az Indiában fejlesztett eszköz előre megszabott pályán mozog, és munkája során adott növényre szabottan vizsgálja, majd amennyiben szükséges, pótolja a talajnedvességet. Emellett az automatikus gyomlálás gondolata tervezés alatt van a National Institute of Technology intézményén belül (KUMAR, 2007).

Az egyik legnagyobb precizitást igénylő feladat a vetés. A kísérleteket megfigyelve leginkább kisebb méretű eszközöket találtam, melynek több oka van. A kisebb méret által bár több időt tölt az eszköz munkával, a talajegyenetlenségeknek megfelelően változtatni tudja a vetési mélységet vetőmagonként, azaz a többi lehetőséggel együtt használva, mint a célzott permetezés és műtrágyázás, egyenletesebb termésátlagot eredményezne. Ezzel kapcsolatos kutatás eredményét láthattuk a 2013-as Field Robot Event-en, ahol bemutatásra került a „DoubleTrouble” nevezetű eszköz (I8). Egyéb, egyszerűbb autonóm vetőgépekkel való kísérletek is fellelhetőek, melyek röpítőtárcsás fűvetőgépként funkcionálnak. Ezek otthon is könnyedén elkészíthetőek.

A permetezés folyamata is felváltható a robotok által. A robotok által végzett mikro permetezés során számos előnnyel számolhatunk. Elsőként érdemes megemlíteni, hogy a vegyszer kiszórására szolgáló fúvókák pár centiméter magasan helyezkednek el a növénytől. Ezek a növény magasságához viszonyulva módosítják helyzetüket. A technikával kiküszöbölésre kerülhet a szél munkavégzést gátló hatása. További előnye az előzetesen, más munkavégzés során felvett „gyom térkép”, amely a permetezőgép programja által feldolgozható információt tartalmaz, így egyes vegyszerekhez tartozó fúvókákat az eszköz pontosan a célterületre érkezve nyitja, illetve zárja (SØGAARD et al., 2006). A célzott

permetezés segítségével egyes mérések szerint a permetszer felhasználása a klasszikus technológiával történő 720 g/h mennyiségről mindösszesen 1 g/h mennyiségre csökkent (PEDERSEN, 2008).

A növényekkel és a termőterülettel kapcsolatos folyamatos adatgyűjtés elengedhetetlen a bizonyos tényezők fennállásakor történő gyors reagálás érdekében. Egy ilyen eszköz képes a különböző környezeti tényezőkön túl egy kontraszt alapú szenzorral bizonyos gyomnövények jelenlétét feltérképezni, illetve azok koordinátáját elmenteni egy esetleges későbbi permetezés segítségére (PEDERSEN, 2008). Ilyen eszköz a University of Southern Denmark által kifejlesztett Armadillo, melynek első változatai kifejezetten adatgyűjtési célból jöttek létre. Újabb változatai ezzel szemben már több funkciót is képesek ellátni, konfigurációtól függően (I9).

Jövőbe mutató kutatások már a szelektív betakarítás gondolatát fogalmazzák meg (BLACKMORE, 2013). Az eljárás erdők tekintetében már megszokott, ám itt több, kisebb méretű fajtákra kell gondolni, melyek lehetnek például különböző zöldségek és gyümölcsök. A „The Scamp Harvester” nevezetű robot a leírások alapján már tesztelés alatt áll. Működésének lényege, hogy különböző szenzorok segítségével (Mint például a szín, illetve méret ellenőrzése és az előbb említett kontraszt alapú alakfelismerés segítségével) határozza meg, hogy az adott növény megfelel-e a meghatározott piaci igényeknek. Ilyen betakarítási kritériumok közé sorolandó még a kalászosok nedvességtartalma, amely ha a betakarítás pillanatában ideális, jelentős költségkímélést vonz magával a szárítás kapcsán. Hasonlóan hasznos lehet még a különböző növények értékét befolyásoló beltartalmi tényezők megfigyelése.

Érdekességgéppen érdemes megemlíteni a multispektrális kamerák alkalmazását. Általában ilyen felvételeket műholdak, illetve speciális repülőgépek segítségével készíthetünk. A repülőgépek által készített képek pontosak, ám költségük miatt kevesen jutnak hozzá a szolgáltatáshoz. A probléma kiküszöbölését szolgálja a Robotics Vision által Názáretben kifejlesztett „hexacopter”, amely multispektrális kamerával és teljes GPS általi vezérelhetőséggel rendelkezik. Ez nem csak lehetőséget ad a távérzékelés kedvezőbb áron történő használatára, hanem a felvételek folyamatos frissítését is biztosítja (I10). A távérzékelés fontos eljárás a földterület és a rajta lévő növények tulajdonságainak meghatározáshoz, hiszen segítségével olyan elváltozások figyelhetőek meg, melyek más módszerrel csak nehezen, vagy egyáltalán nem lehetségesek. Többek között gyors talajnedvesség ellenőrzést, jégmegfigyelést és az anyagcsere intenzitás ellenőrzését is lehetővé tesz.

Többször említésre került az adott kultúrnövények és gyomnövények kontraszt alapú felismerése. A művelet megfelelő optimalizálás mellett egy egyszerű, hétköznapi kamerával is elvégezhető a megfelelő program közbeiktatásával. A folyamat során az elkészült kép felbontását csökkentjük, majd monokrómmá konvertáljuk. Ekkor végrehajtjuk a képen a 2D Fourier-transzformációt, amely képünket lényegében binárisan meghatározott pixelhalmazra változtatja. A pixelek 1 és 0 értéket vehetnek fel, melynek hatására a képünk ténylegesen csak fekete és fehér színt fog tartalmazni. Rendszerint a fehér szín a növény sziluettjét jeleníti meg. A megadott minimum méretnél kisebb foltok eltávolítása után tiszta képet kapunk a növény formájáról, amely összevethető az eddig vizsgált egyedekkel, vagyis jó eséllyel beazonosítható annak fajtája (I11).

2.1. Lego Mindstorms

A vezérlőegység alapjaiban határozza meg az elkészült robotunk tulajdonságait, adatfeldolgozási sebességét, és talán ami még fontosabb, annak bővíthetőségét. Itt alapvetően két kategóriát különböztetünk meg. Az első a mikrokontrollert (Lego Mindstorms, Arduino, Intel Galileo), míg a másik a mikroprocesszort (Raspberry PI) tartalmazó eszközök családjá. A kettő kategória célja megegyezik. Különböző motorok, szenzorok (azaz perifériák) irányítása, adatfeldolgozás, és ezáltal a rá feltöltött program végrehajtása. A mikroprocesszor önmagában csak egy műveletvégző egység, amely nem rendelkezik memóriával, illetve más perifériával. Ezzel szemben a mikrokontroller önmagában egy kis számítógép, amely a chip-en belül rendelkezik általában kevés, de a vezérlési feladat ellátásához elegendő RAM-mal és ROM-mal.

A lényegi különbség a felhasználási területen tér el. A mikroprocesszor első sorban nagy adatmennyiségnél, grafikai alkalmazásoknál és komplex matematikai függvények megoldása esetén használható effektíven, míg a mikrokontroller a nevéből is eredően vezérlési feladatokat tud ellátni. Jellegzetessége, hogy nagyobb üzemi feszültséggel (Általában 5 volt körüli érték) kell ellátni, melynek hatására korlátozott mennyiségben önmaga is el tud látni egyes perifériákat a működéséhez szükséges üzemi feszültséggel.

A robot elkészítéséhez a választás a Lego által gyártott Mindstorms NXT rendszerre esett (1. Ábra). A gyártó hallatán feltételezhetően mindenki a játékra asszociál, hiszen a cég a harmincas évek óta játékgyártással foglalkozik, azonban itt sokkal többről van szó. A cég felismerte a kor igényeit és létrehozta a moduláris kialakítású, szabadon programozható eszközt, melynek sokszínű kialakítási lehetőségeinek csak a kreativitás szab határt. Számos

műszaki képzés alatt is a mindennapok része ezen eszköz, ha robotikáról, illetve automatizálásról van szó.



1. ábra: A Lego Mindstorms NXT rendszer, Forrás: I6

Az „intelligens téglá” 3 szervomotornak és 4 szenzornak megfelelő be és kimeneti portot tartalmaz, ami megfelelő eszközzel tovább bővíthető. Mindez 6 tűs, moduláris csatlakozón keresztül kapcsolódik a rendszerhez. Egyéb kommunikációs lehetőséggel is rendelkezik a készülék, mint az USB 2.0 adatátviteli port és a Bluetooth 2.0 támogatás. Az előbbi port a programunk feltöltését és a későbbiekben tárgyalt adatloggolást teszi lehetővé, míg az utóbbi a program feltöltése mellett lehetőséget ad több brick összekapcsolására.

Négy fő kiadását ismerjük (RCX, NXT, NXT 2.0 és EV3), melyek közül az általunk használt változat a 2.0 széria. Az eszközben egy AVR architektúrára épülő, 48 MHz órajelen dolgozó Atmel processzor, 64 Kbyte RAM és 256 Kbyte flash memória dolgozik karöltve. Mai szemmel nézve ezek nem nagy számok, ám itt céleszközzről van szó, melyet nem terhel le az egyes operációs rendszerek grafikus mivolta által gerjesztett alapkihasználtság, tehát a modellünk elkészítéséhez teljesen megfelelő sebességet nyújt.

A legelső széria RCX (Robotic Command eXplorers) néven jelent meg 1998-ban. A vezérlőt egy 8 bites Renesas processzor hajtotta és mindösszesen 32 Kbyte ROM-on osztozott a firmware és a felhasználó által írt program. Beépített infraporttal rendelkezett, emellett 3 szenzor és 3 szervomotor meghajtására alkalmas portkészlettel bírt.

A második széria, amely egyben az első NXT verzió, jelentős fejlesztésekkel került piacra, ám ez nem meglepő az első verzióhoz képest késői, 2006-os piacra kerülés által. Bejelentésére Nürnberg-ben került sor, ahol azonnal több termékdíjjal jutalmazták a fejlesztést. Megjelent a több információ átadására képes grafikus, 100x64 pixeles felbontású, folyadékkristályos kijelző, illetve a gyárilag kezelhető szenzorok száma is négyre nőtt az új mikrokontroller végett. Az általunk használt 2.0-ás verziójú vezérlőtől lényeges különbség

csupán a mellékelt kiegészítőkben rejlik és abban, hogy míg az első generációs eszköz integer műveleteket használ, a 2.0-ás verzió esetén megjelenik a lebegő pontos számítás lehetősége is. Az NXT kereskedelemben vásárolható változata mellett kapható volt oktatási célú kiadásban is, melynek lényeges különbsége az eredeti verzióhoz képest, hogy a mellékelt építőelemek számát 619-ről 1050-re növelték az egyéb, edukációs célú kísérletek kivitelezhetősége végett, mint például az áttételek vizsgálata (Forrás: I12).

A legújabb verzió az EV3 nevet viseli. A 2013. szeptember 1.-jén megjelent eszköz immáron 32 bites, ARM9 architektúrára épülő processzor segítségével üzemel, amely jelentős előrelépés számítási teljesítményt illetően az előző generációkhoz képest. Emellett 64 Mbyte RAM és 16 Mbyte flash memóriával rendelkezik az eszköz, melyből a flash memória először bővíthető micro SD port segítségével a komplexebb programok tárolása érdekében. A felhasználható portok száma eggyel növekedett a motorok tekintetében, míg a szenzorportok sebessége növekedett jelentősen. Digitális szenzorportok portok esetében 460,8-ról megközelítőleg 1200 Kbyte/sec-ra változott. Emellett biztosítja az Apple mobil platformmal való kommunikációt. Az új verzió legtöbb tekintetben előrelépés, hiszen számos újítást hozott, ám bizonyos téren, mint például a mellékelt szervomotorok teljesítménye, visszalépésnek tekinthető annak gyengítése végett. De mivel ezen alkatrészek felcserélhetők az előző generációéval, nem jelent valós problémát.

Programozása az összes verzióknak több nyelven lehetséges, a felhasználó előzetes programozási ismeretét figyelembe véve. Legismertebb talán az alapvető funkciókat működtetni képes NXT-G grafikus fejlesztőkörnyezet, amely segítségével folyamatábrás ábrázolással vagyunk képesek irányítani az eszközt. Jelenlegi legújabb kiadása az EV3 vezérlővel egy időben megjelent EV3 1.0.1 verzió, mely Windows és Mac OS operációs rendszerek alatt egyaránt képes futni. Sajnos azonban lefelé nem kompatibilis az általunk használt verzióval, amely az NXT, illetve NXT 2.0 vezérlőhöz lett fejlesztve.

A NXT mellékelt fejlesztőkörnyezetével való programozás alapeleme a blokk. A különböző blokkokat kiválasztva, paraméterezve és összekötve érhetjük el a kitűzött célt. Előnye, hogy segítségével aránylag gyorsan létre tudjuk hozni a megfelelő programot, illetve legtöbb esetben idegen szenzorok is használhatóak vele, amennyiben a gyártó mellékeli a fejlesztőkörnyezetbe importálható blokkokat. A blokkok adott funkciótól függő paraméterezhetőséggel bírnak. Amennyiben bővebb paraméterezést szeretnénk elérni, célszerű egy, a környezet által támogatott programnyelvet használni a feladathoz. Különböző fejlesztőkörnyezetek állnak rendelkezésre erre a célra. A Microsoft Robotics Developer Studio, illetve a RobotC által elérhető a C# nyelv erre a célra készült könyvtára, melyet a

későbbiekben használni is fogunk. Ezen kívül számos nyelven nyílik lehetőségünk a programozására, fejlesztőkörnyezettől függően, mint például Java, Python, Basic, Assembly, Lua, G és egyébek. Ezt a célt szolgáló segédprogram a RoboLab 2.9, NBC, NI LabView, leJOS NXJ, pbLua, LEJOS OSEK és ICON (Forrás: I13).

Napjainkban az eszköz az edukációs és szórakoztató célokon kívül számos verseny középpontjává vált. Ilyen többek között az évente megrendezésre kerülő MoonBots Challenge, melynek célja egy holdjáró modelljének elkészítése. A verseny 2012-es fordulóját magyar csapat tagjai nyerték pályamunkájukkal.

3. Anyag és módszer

Amennyiben egy, az előbbieken felsorolt kutatásokban szerelő robotok funkciót reprezentáló eszközt építünk, mindenképp tisztázni kell azok működési metódusát és fizikai felépítését. Ezt vizsgálva nyilvánvalóvá válik a robotok működését leginkább elősegítő, és több tényezőt jelentősen befolyásoló eszköz, a mikrokontroller fontossága.

Számos vezérlésre alkalmas eszköz van jelenleg a piacon. Talán legnagyobb dominanciát az Atmel Corporation élvez, de mellette jelen vannak többek között az Intel, Infineon, MIPS, Renesas, Silicon Laboratories, Toshiba és a Texas Instruments eszközei is. Ezeket az eszközöket számos gyártó felhasználja, és formába öntve árulja. Nem kivétel ez alól a korábban említett Lego Mindstorms NXT, illetve a későbbiekben felhasználásra kerülő Arduino rendszer sem. Az első tesztelésekre és kísérletezésekre az NXT rendszert használtam, mivel ennek fizikailag biztosított kényelmi funkciói, mint a moduláris csatlakozó, esélyt adnak kifejezetten a programozás kérdéseire koncentrálni a később Arduino rendszerrel tapasztalt elektronikai tervezés igényét elhagyva.

Felmértem a szakirodalmakban olvasott kutatások eredményét és célul tűztem ki az irányítási modell több változatának elkészítését, melyek modulonként tanulmányozhatóak. A választás azért az irányításra esett vizsgálandó témaként, mivel szinte minden témába vágó robotot megfigyelve ezt tekintettem az egyik legfontosabb alapfunkciónak. Emellett ez a metódus a legszélesebb körben testreszabható különböző költséges szenzorok megvásárlása nélkül.

Az irányítás autonóm és távvezérelt módon is programozásra kerül a teljes lefedettség érdekében. Fontosnak tartom a GPS vezérlés megoldása mellett a lehetséges terepakadályok felmérését és kezelését. Emellett a televezérelt mód nem megszokott módon kerül kivitelezésre. Ennek kivitelezését egy Microsoft Kinect mozgásérzékelő és elemző fogja segíteni.

A tervezési, majd kivitelezési folyamat során először célszerű a jövőben tervezett használati körülményeknek megfelelően kialakítani a vázat, illetve meghajtást. Emellett figyelembe kell venni, hogy a jelenleg épített modellünk a későbbiekben igény szerint nagyobb méretben is gond nélkül elkészíthető legyen. A feladat elvégzéséhez segítségemre lesz számos makett és terv. Ezt követően az optimális anyagok kiválasztása után el is készítem a megfelelő vázat. A kísérletezésekhez ennek ellenére a Lego Technics szettből ismerős építőelemeket használok azok teljes modulárissága miatt (3. melléklet). Jelentős idő és költségtakarékosság az ilyen módon történő modellezés, hiszen a funkcionalitásban fellépő

hiba esetén könnyedén újra lehet építeni a konstrukciót, illetve több lehetőséget tesztelhetünk a prototípus gyártás jellemzően magas költségeit elkerülve. Az itt szerzett tapasztalatokat könnyedén fel tudjuk használni a végleges váz esetleges újratervezése során. Ilyen problémák a szerkezeten túl a motorok szinkronizálásában és a méretekben fordulhatnak elő.

Az egyes irányítási módok tesztelését és finomhangolását számítógépen és fizikai környezetben egyaránt el kell végezni. Az utóbbira egy tesztkörnyezet kerül kialakításra. Számos esetben a reakciók kezdő és végpontjainak meghatározásai pontos időmérést igényelnek, amelyek eredményét egy ilyen célra használatos, hang alapján működő mérőszoftver segítette.

Az NXT platform első sorban egy működő prototípus elkészítését, illetve a különböző programok gyors tesztelését biztosítja. Ennek oka a már korábban említett moduláriság, melynek segítségével a későbbi platformokra való áttérés során nem okoz gondot. Ezzel szemben az Arduino leginkább a programozása terén tud tapasztalatokat adni, mint lehetséges későbbi alternatíva. Egy ilyen, elektronikailag komplexebb felépítésű rendszerrel a prototípus elkészítése sokkal több időt vesz igénybe, pont úgy, mint annak hiba esetén történő módosítása. Ezért mondhatjuk, hogy amennyiben az NXT rendszer segítségével egy működő konstrukciót sikerült létrehozni, annak alapjára, és az általa szerzett tapasztalatokkal az Arduino rendszerrel is időtakarékosan tudunk egy hasonló rendszert felállítani.

Mint azt említettem, az Arduino-val való programozás valamivel komplikáltabb, így a rendszerre számos program kerül elkészítésre annak teljes átlátása érdekében. Emellett a szenzorok, motorok és különböző elektronikai komponensek egy átmeneti kapcsolási tábla segítségével kerülnek összekapcsolásra, amely lényegesen nagyobb szabadságot nyújt, mint a klasszikus ónnal való forrasztás.

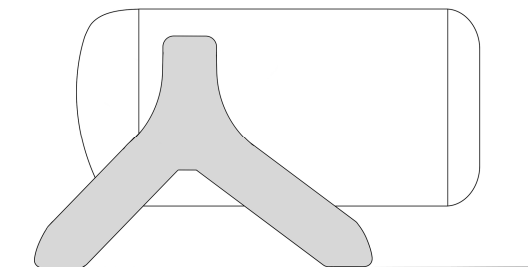
Az adatgyűjtés, mint ágazati robotoknál fontos funkció teljes elemzésre kerül. Foglalkozunk az NXT gyári fejlesztőkörnyezete által nyújtott adatlogging lehetőségekkel, illetve belső fájl létrehozásával történő adatgyűjtéssel. Az itt létrejövő értékek elemzési és prezentálási lehetőségei feltüntetésre kerülnek. Az Arduino rendszer során is megemlítjük a soros port általi, számítógép felé történő adatküldést, melyek szintén felhasználásra kerülhetnek egyes esetekben.

4. Eredmények és azok értékelése

Még a programozás és az összeállítás előtt tervezésre és megépítésre került egy, az egész rendszert keretbe foglaló váz. A váz elkészítésénél fontos volt a moduláriság és a későbbi bővíthetőségekhez való alkalmazkodás. Az első tervek az Autodesk AutoCAD tervezőszoftver segítségével készültek el. A vázat két formára vágott lemez képezi, amely 6 mm vastag fém szálakkal van összekötve négy pontban. A szál módosításával növelhetjük, vagy csökkenthetjük a készülék szélességét, a beépített szenzorok mennyiségétől és méretétől függően. Ugyan ezt a célt szolgálja a meghajtás, illetve a felfüggesztés is, amely oldalanként teljes egészében független egymástól, így azok egymástól való távolítása során nem jelentkezik mechanikai korlát. A program minimális finomhangolása után bármilyen méret mellett használható. A két lemez először 2 mm vastag alumínium lemezből lett CNC marógép segítségével kivágva, de ez a jelen rendszerünknek túl nehéznek bizonyult. Ennek megoldását a később leírt „Fejlesztési lehetőségek” fejezet tartalmazza. A hiba jelenlegi konfigurációnkhoz való kiküszöböléséhez újra legyártásra került az alkatrész, immáron 2 mm vastag akril lemez felhasználásával. A váz súlya jelentősen csökkent. Az összekötőkkel együtt, meghajtás nélkül mindössze 225 grammal számolunk, amely a felfüggesztéshez szükséges csapággyakkal együtt 251 grammal növekszik.

A felfüggesztés hasonlóképpen polikarbonát lemezek segítségével kapcsolódik a vázhoz. Itt négy fordított „Y” alakú lemeze van szó, párokba szedve, szintén az előbbieken használt szálak rövidebbre vágott példányaival összefogatva. A talajviszonyok változását követve maximum 45 fokos rádiuszban képes elfordulni a vázhoz képest.

A felfüggesztés középső pontja egy-egy 22x50x14 mm-es golyóscsapággal került a vázhoz rögzítve. A csapágy túlméretezése végett adódik egy ellenállás (Meghajtásához megközelítőleg 0,5 N*m erő szükséges), amely precízebbé és folyamatosabbá teszi a hajtás vázhoz viszonyított mozgását.



2. ábra: A váz és a felfüggesztés ábrája, Saját szerkesztés

A lap hosszanti mérete 22 cm, míg magassága 10,5 cm. A felfüggesztésből adódó hasmagasság 5 cm. A magasság az „Y” felfüggesztési elem és a hajtáslánc méretének

változtatásával a környezetnek és igényeknek megfelelően alakítható. A konstrukció elég masszív a fokozott igénybevételhez, illetve a későbbiekben tervezett arányos nagyítás során is felhasználható tervezete, mivel alkatrészei a kereskedelemben nagyobb méretben egyaránt elérhetőek bizonyos korlátig.

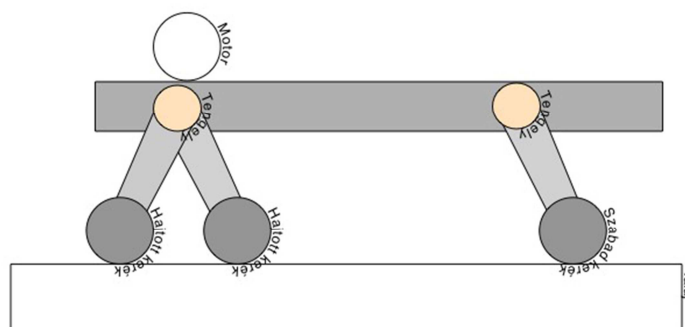
4.1. Meghajtás és irányítás

A meghajtással kapcsolatban az elsődleges szempont a terepviszonyok által okozott változó talajkörülmények kiküszöbölése, a stabil mozgás és ezáltal a precíz manőverezés elérése. Ezt elősegítve az eddigi kísérleteimhez képest radikális technikájú felfüggesztést, és egyedileg tervezett, majd legyártott alkatrészeket használtam. Emellett több irányítási modell került kidolgozásra autonóm és távvezérelt módon egyaránt.

4.1.1.Felfüggesztés

Kísérletezési céllal egy fordított „Y” alakú első felfüggesztést terveztem a terepviszonyok egyenletesebbé tételéhez. A hajtás saját gondolat alapján került modellezésre, melyet a megfelelő tesztelés és az egyedi alkatrészek legyártása után alkalmaztam. A felfüggesztés lényege, hogy az egész hajtás, a motorral együtt képes elfordulni előre és hátra meghatározott rádiuszban a saját tengelye körül, ezzel elősegítve a folyamatos, lehető legnagyobb felületű kapcsolatot a talajjal. Ténylegesen csak az első keréksor kerül meghajtásra lánc segítségével, majd ez a gumiheveder segítségével juttatja el az „Y” második, hátsó keréksorára az erőt. A hátsó tengely, amely egy középén elhelyezett, minden irányba szabadon forgó kereket tartalmaz, nincs ellátva hajtással. Inkább csak stabilizálás céljából van jelen. Szilárd talajon gyakorlatilag hiányában is képes a manőverezésre az eszköz, ám fokozott terepviszonyok mellett szükséges ez a kiegészítő (2. ábra).

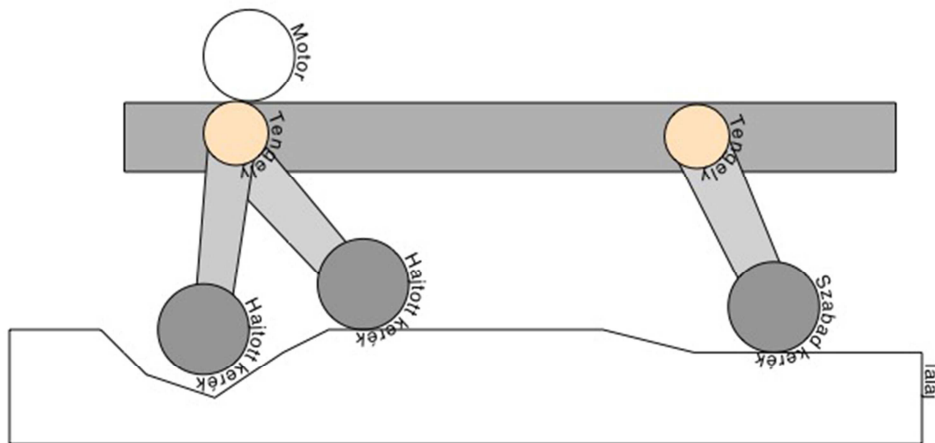
Az első tengely megközelítőleg 24 cm² területen képes tapadást biztosítani. A két oldal egyaránt egy-egy 9842-es kódszámú motorral kerül meghajtásra. Az áttétel 1:1 arányú, azaz nem történt módosítás ilyen téren. Eerre leginkább a fejlesztőkörnyezethez való optimalizálás miatt van szükség, illetve a motor sebessége és nyomatéka ideális a gyári beállítások használata mellett. A két motor által adódó felépítésnek köszönhetően a programok írása során precízebb irányítást tudunk biztosítani, mivel a két motort függetlenül tudjuk ellátni a mozgáshoz szükséges adatokkal.



2. ábra: Egyenes terepen történő haladás (sematikus ábra), Saját szerkesztés

Egy kisebb terepakadály esetén az elől lévő konstrukció 45 fokos szög mértékéig képes elmozdulni az egyenestől, így felveszi a lejtő, vagy domborulat alakját. Ezzel biztosítva a talajjal való folyamatos kontaktust (3. ábra). A megfelelő effektus elérése érdekében a motornak az előbb leírt konstrukcióhoz kell kapcsolódnia, nem pedig az alvázhoz a lehetséges csavarodás elkerülődése érdekében. Az alvázhoz nem csatlakozik jelen esetben hajtás, csupán az előbbieken említett szabadon forgó tengely, amelyet oldalnézetből vizsgálva, annak jobb és bal oldalán lévő akasztók gátolják meg a túlfordulástól. Opcionálisan a hajtott tengely alapállapotba történő könnyebb visszatérését elősegítő rugókat alkalmazhatunk, amely mindkét irányba feszesíti a tengelyt. A koncepció a talajegyenetlenségek kiegyenlítése mellett több hajtott kereket eredményez, amely a nagyobb felület miatt jobb terepjáró képességet biztosít. Klasszikus értelemben vett négykerék meghajtáshoz négy különálló motorra lenne szükség, ami jelentős energia és egy szabad port felesleges felhasználását eredményezné. Esetleges fejlesztési lehetősége a 4 hasonló felfüggesztéssel való ellátás, ám ez leginkább más platformokon elképzelhető a portok hiánya miatt. Minden méretet megtartva a talaj és a gumiheveder közötti felületi kontaktus több mint 52 cm^2 lenne ebben az esetben. Amennyiben az eszközt arányaiban nagyítanánk, megfelelő gumiheveder és önsúly esetén a tapadás mértéke felvenné a versenyt több korábban vizsgált eszközével.

Érdemes megemlíteni az Arduino platformon végzett kísérleteket, melyek egy automata sebességváltó gondolatát fogalmazzák meg. Jelentős szinkronizálást igénylő fejlesztői munkáról van szó, de eredményeképp a két platformot ötvözve a gyakorlatban használhatóbb rendszert eredményezne.



3. ábra: Terepakadály esetén történő mozgás (sematikus ábra), Saját szerkesztés

A kanyarodást a jobb és bal oldali, későbbiekben „A” és „B” elnevezésű, egyúttal hajtást is végző motor segíti. A két motor fordulatszámát és haladási irányát függetlenül változtatva idézhetünk elő íves vagy helyben történő fordulást. A kerekek elfordításával történő irányváltatáshoz több szervomotorra, és egy külön ezt elősegítő kormányműre lenne szükség, amely a független vezérléssel elérhető helyben fordulás által adott precizitást nem tudja reprodukálni. Mivel nem gyorsan haladó tárgyról van szó, az aerodinamika helyett sokkal fontosabb tényező az előbb említett terepjárási képességen kívül a maximális precizitás és az esetleges időjárás-állóság.

4.1.2. Terepakadály kikerülési képesség (NXT-G)

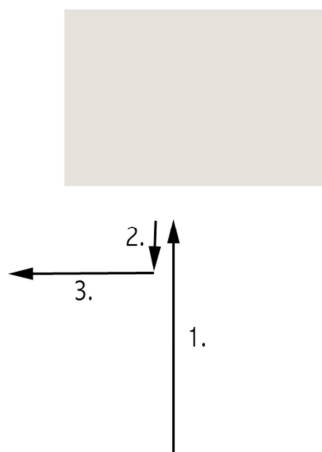
Az irányítás autonomitásának elérése érdekében több folyamatot szükséges egyidejűleg futtatni a készüléken. Az egyik ilyen szál, a mozgás során előkerülő tárgyak kikerülését szolgálja. Ezen program első verziója a mellékelt fejlesztőkörnyezet segítségével lett elkészítve. Ezzel egy időben egy folyamatábrát is kaptunk, mely segítségre lehet a későbbiekben más programkódokra való fordításnál.

A program algoritmus a magát ismétlő, hibaelkerülő mechanizmus. A robot addig megy előre (később a GPS vezérlés által megszabott útvonalon), ameddig az elől lévő, menetirányba néző ultrahangos távolságmérő 5 centiméternél kisebb távolságnak megfelelő értéket nem ad vissza a vezérlőnek. Amennyiben beteljesül ez a kritérium, a robot megáll és a biztonságos manőverezés érdekében visszalép egy teljes tengelyfordulatnyit az „A” és „B” port jelzésű szervomotorral egyaránt. A teljes megállás előtt már a tárgytól 10 centiméterre történik egy 50%-os lassítás, a szervomotorok mérsékeltebb igénybevétele miatt. Mivel a szervomotorok 1 fok pontossággal működnek, egy hirtelen megállás egy adott fokot jelentene, ami esetünkben a tárgy észlelése során fennálló értéket jelenti. A nagyobb sebesség és ehhez

társuló nagyobb önsúly fokozott mechanikai terhelést jelent a szervomotorokra egy hirtelen megállás alkalmával, melynek elkerülése végett került beépítésre a fent említett kód.

Az ultrahangos távolságmérő egy harmadik, „C” jelölésű porton lévő motor tengelyén helyezkedik el, melynek segítségével körbe tud tekinteni a megfelelő kikerülési útvonal meghatározásához. Az algoritmus szerint az előbb említett „C” ponton lévő motor tengelye először 90 fokot balra fordul, majd végez egy távolságmérést, melynek eredményét elküldi egy később összehasonlítást végző matematikai függvénynek (két változó segítségével). Ezek után ugyanez a motor 180 fokot jobbra fordul, majd a szenzor ebben az irányban is mérést végez az ultrahangos távolságmérő segítségével. A két oldal által adott eredmények közül az előbb említett matematikai függvény kiválasztja a nagyobb értéket és egy logikai, igaz-hamis függvény segítségével két szála osztja a folyamatot. Az eldöntendő feltétel, hogy a bal oldalon olvasott érték nagyobb-e a jobb oldalon olvasott értéknél. Mindkét szál a válaszhoz tartozó értéknek megfelelő fordulási irányt és az ehhez kapcsolódó későbbi műveleteket tartalmazza. Tulajdonképpen a nagyobb értékhez tartozó oldal felé fordítja a robotot az „A” és „B” motor segítségével 90 fokos, helyben történő megfordítással.

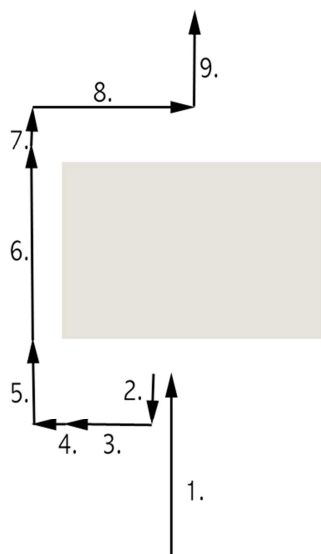
Méréseim szerint a forduláshoz szükséges időtartam 80%-os motorteljesítmény mellett 1,65 másodperc. Erre az értékre a RobotC-ben történő programozás során lesz szükségünk. A kikerülés lefutása után a program első része fut tovább, azaz egyenes mozgás indul el az „A” és „B” motor segítségével, 100%-os teljesítmény mellett (4. ábra).



4. ábra: Az első verziójú kikerülési manőver folyamatábrája, Saját szerkesztés

Ez az algoritmus jól működik az ütközés elkerülésére, de a haladási irányt megváltoztatja, így ennek kiküszöbölésére elkészítettem a második verziót az előbbi programból kiindulva. A fő változás a távolságmérés utáni részt érinti. Az akadály által elindított algoritmus elvégzi az első programban is bemutatott kétoldalú mérést, majd a

megfelelő irányba történő fordulás után a szenzort az akadályozó tényező felé forgatja, ezek után addig folytatja az előre történő mozgást mérsékelt, 80%-os teljesítmény mellett, ameddig nem észlel elégséges, jelen esetben legalább 80 centiméteres távolságnak megfelelő értéket. Ekkor a fordulás biztonságos lebonyolítása végett előre halad egy tengelyfordulatnyit, ám ez nem észrevehető, mivel nincs várakozás a két lépés közé iktatva. A lépés után elfordul a továbbhaladás irányába 90 fokkal, majd elindul, ám a szenzor továbbra is méri a terepakadály haladás irányába terjeszkedő méretét mindaddig, ameddig a szenzor nem mutat megfelelő távolságot az eredeti haladási irányhoz való visszatérésre. Jelen esetben ez a távolság is az előzőhöz hasonlóan legalább 80 centiméter. Ilyenkor ismételten szükséges egy, a forduláshoz szükséges teljes tengelyfordulatnyi előrehaladás. Ekkor a robot a megfelelő irányba történő 90 fokos irányváltoztatás után ismét előre fog haladni pontosan akkora távolságot, mint a kikerülés első lépésénél, majd egy ismételt 90 fokos irányváltoztatás után az eredeti útvonalon fog haladni (5. ábra). Ennek magyarázata, hogy az első kanyar után egy változóba számlálódik a tengelyfordulatok, illetve jelen esetben a folyamat ismétlődésének száma egészen a következő fordulóig. Azaz mérésre kerül az eredeti nyomvonaltól való kitérés mértéke, amely később felhasználásra kerül az utolsó ciklus során (6. melléklet).



5. ábra: A második verziójú kikerülési manőver folyamatábrája, Saját szerkesztés

Több probléma is felmerült a manőver programozása során. Az első ilyen említésre érdemes probléma a kétoldali távolság meghatározásból adódik. Amennyiben a szenzor mindkét oldalon végtelen, azaz a szenzor méréshatárán kívüli távolságnak megfelelő értékkel találkozunk, egy egyenlőségi feltétel jelenik meg a programban, így a logikai blokk nem tudja eldönteni, hogy melyik méréshez tartozik nagyobb érték. Ekkor véletlenszerű választás történik, ami nem minden esetben optimális. A programot a végtelenségig lehetne

finomhangolni, viszont ennek korlátot szab az aránylag kis rendelkezésre álló memória. Megoldás lenne a problémára az újabb verziójú hardver használata, mivel rengeteg lehetőség programozható még a modulon belül, beleértve a teljes tárgyletapogatást, melyhez két ultrahangos távolságmérő szükséges. A második távolságmérőre azért van szükség, hogy a letapogatás során (melynek során az első távolságmérő az egyik oldal irányába mér) esetlegesen a robot elé kerülő akadályt kezelni tudjuk.

4.1.3. Terepakadály kikerülési képesség (Robot-C)

A fent említett kikerülési módszer az eredeti, NXT-G környezet mellett RobotC fejlesztőkörnyezetben is programozásra került. Nehézségnek számít a programozás során, hogy egyes értékeket, melyek az NXT-G környezetben adottak, mint például a kerék adott fokban történő elforgatása, itt másodperc és teljesítmény függvényében van lehetőségünk megadni. A programozás során egyértelmű, C nyelvből ismerős ciklusokat, utasításokat és elágazásokat használunk, minimális módosításokkal és specifikálásokkal, ami nyilván szükséges a megszokottnál más felhasználási terület miatt. A kód írása előtt deklarálni kell a csatlakoztatott eszközöket, azok által felhasznált portokat és azok később használatos hivatkozási nevüket, ami bármi lehet. Ezt követően több „task elnevezés ()” blokkokat hozhatunk létre, amely a többszálú programfuttatáshoz szükséges. A fő blokk minden esetben a „task main ()” nevet viseli. Ezek után deklarálhatjuk a változókat, konstansokat, majd megadhatunk több sor terjedelmű utasítást, amely egyszer fog lefutni az indítás után. Ez lehet egy kijelzőre írt üzenet, hangjelzés, vagy bármi használatot elősegítő funkció. Amennyiben a programunk ismétlődő szakaszát tervezzük írni, célszerű azt egy while cikluson belül elvégezni. Esetünkben ez a while ciklus tartalmazza az egész, korábban NXT-G-ben is használt mechanizmust.

Klasszikus programozói gondolkodással az ilyen jellegű programozás átláthatóbb a grafikus felületnél, viszont olyan funkciók elvesztésével jár, mint a korábban említett motor teljesítmény átszámítás, illetve fontos tényező, hogy bizonyos külső gyártól származó szenzorok nem támogatják a nyelvet.

4.1.4. GPS vezérlés

Az autonóm irányítás egyik fontos tényezője a műholdak segítségével történő vezérlés. Mivel az ehhez szükséges szenzor ára elég magas, így a program megírása után elméleti szinten történik működésének prezentálása.

Talán legkézenfekvőbb GPS modulválasztás a platformhoz a Dexter Industries által gyártott dGPS elnevezésű eszköz. A csatlakozás nem igényel semmilyen jellegű átalakítást,

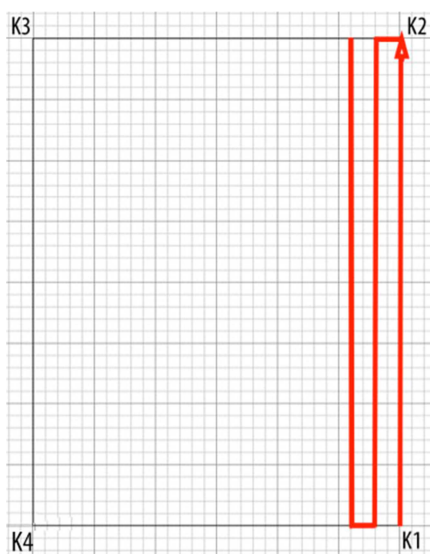
illetve a gyártó mellékli a meghajtáshoz szükséges blokkokat. Ezek a blokkok a „GPS Read”, „GPS Navigation” és „GPS Angle of Travel” nevet viselik. A GPS Read blokk képes kiolvasni a szélességi, hosszúsági és magassági koordinátákat, így ezen értékek a későbbiekben egy változóba mentve felhasználhatóvá válnak. Emellett az időt (hhmmss formátumban), sebességet (cm/s mértékegységben) és a haladási irányt (A beépített iránytű segítségével, melyek fok formátumban tárolódnak) is képesek vagyunk vele meghatározni. Ezek az adatok az irányítás mellett a későbbiekben használatos adatloggolás során is hasznosak lesznek számunkra.

A GPS Navigation blokk egy automatizált irányítási blokk, amely képes az eszközt egy előre megadott szélességi és hosszúsági koordináta segítségével adott pontra irányítani. A GPS Angle of Travel blokk a két pont, azaz az indulás és várható érkezés közötti irányt mutatja meg. Valamivel precízebb értéket ad, mint a GPS Read blokkban található haladási irány meghatározását szolgáló értékadás. Az első használat során tárolásra kerül a koordináta, majd a második használata során (rendszerint kanyar során) kapjuk meg az értéket.

A GPS vezérlés jelen esetünkben négy koordináta közötti pásztázást jelent, meghatározott felbontásban. Felbontás alatt itt az egyes számított útvonal egyenesei közötti távolságot értjük. Elsőként szükséges a területet leíró négy koordináta megadása. Ezt két módon tudjuk megtenni. A kért mód az indítás során kerül kiválasztásra. Az első lehetőség a Bluetooth kapcsolaton, ezen belül is a régebbi mobiltelefonok által használt Bluejacking technikával történő értékadás. A készülék a kijelzőn közli a felhasználónak, hogy épp melyik értékre van szüksége. Az értékek változóba kerülnek elmentésre, majd ezeket felhasználhatjuk a későbbi munkafolyamatunk során. A másik értékadási lehetőség a fizikailag történő mérés. Ebben az esetben el kell látogatnunk a négy ponthoz, majd mindegyiken egy, a 3. bemeneti portra csatlakoztatott gomb megnyomásának segítségével mérést végezni, melynek eredménye ugyan úgy mentésre kerül az adott változóba (8. melléklet). Az indulás a terület térkép szerinti délkeleti (K1) sarkából történik. A program fejlesztése során kezelhetővé tehető az összes sarokpont támogatása indítás céljából, ám ez jelentős memória igényű módosítás lenne, tehát jelen esetünkben mondhatni kivitelezhetetlen. Az indulást követően a mozgás a négy pont által bezárt terület keleti élén keresztül elindul az északi pont (K2) felé, ahol egy matematikai művelet segítségével megtörténik egy, az északnyugati és északkeleti pont (K2-K3) által bezárt egyenes mentén történő koordináta geometriai eltolás (7. melléklet). Ezzel párhuzamosan a délnyugati és délkeleti él (K1-K4) mentén is történik egy ugyan ilyen mértékű eltolás. Ezen új pontok felülírásra kerülnek a keleti koordináták (beleértve az indulás koordinátáját) változóiban, azaz újra

történik egy egyenes képzése a kijelölt területen, melyen az eszköz ismét végigmegy. Amennyiben a folyamatosan módosított keleti oldal koordinátáinak változói és a nyugati oldal koordinátáinak változói kisebb vagy egyenlő eltérést mutatnak, mint az eltolás értéke, a program befejezi a futását és a gép visszatér az induló pozícióba.

Felmerül a kérdés, hogy mi történik akkor, ha nem tökéletes négyzet áll rendelkezésünkre, így azok oldalai nem egyenlő hosszúságúak. Erre az esetre egy módosított algoritmust kell írunk, ami végül használható az előbbieken felírt helyzet esetén is. A kis programmemória végett ez a lehetőség nem került programozásra még, de várhatóan a generációváltás, vagy egy másik platform használata megoldja a problémát. A módosítás leginkább a koordinátaszámítás módját érintené. Tételezzük, hogy a 6. ábrán lévő terület K4-K2 oldala rövidebb a vele párhuzamos oldal nagyságánál. Ebben az esetben terveim szerint a program a közeledés során először elért pontot nem módosítja tovább, (amennyiben a számolt pont nagyobb vagy egyenlő, mint az előre megadott, jelen esetben K4 pont) azaz befejezettnek tekinti. A következőkben már csak a K3-K2 oldalon történik ponteltolás, míg a másik oldalon a már fix K4 pont lesz az állandó célállomás a pásztázás során.



6. ábra: Pásztázó mozgás, Saját szerkesztés

4.2. Microsoft Kinect segítségével történő vezérlés

Az előbb kifejtett autonóm vezérlési mód mellett lehetőség van egy egészen más, mondhatni jövőbe mutató távvezérlési formát is használnunk. A vezérlés alapköve a Microsoft által 2010 novemberében bemutatott Kinect mozgásérzékelésre alkalmas beviteli eszköz adja. Műszakilag az eszköz igen egyszerű. Összesen három lényegi alkatrészt tartalmaz a jel felfogására (Forrás: I14). Egy infravörös projektort, egy ezt felfogó infravörös mélységérzékelő kamerát és egy szimpla RGB kamerát. Mindkét kamera VGA, azaz 640x480

pixeles felbontásban készít felvételt, 30 Hz-es frissítési frekvencia mellett. Az adatfeldolgozást a PrimeSense PS1080 processzor végzi.

A készülék igen sok lehetőséget tartalmaz, amivel természetesen a Microsoft mellett más cég is tisztában volt. Az Adafruit Industries mintegy 3000\$ jutalmat ajánlott annak, aki elsőként ír egy PC-vel való kapcsolathoz szükséges illesztőprogramot az eszközhöz. A hullámnak és a fokozott igénynek köszönhetően nem sokkal később, 2011-ben megjelent az első hivatalos fejlesztőkörnyezet, Kinect SDK néven. A környezet számos programnyelvet támogat, beleértve a C++, CLI, C#, vagy éppen Visual Basic .NET-et. A gyári fejlesztői platform előtt kiadásra kerültek a különböző részegységekre specializálódott környezetek. Ilyen az OpenKinect által fejlesztett Libfreenect Drivers, amely a mikrofonon kívül mindenhez hozzáférést biztosít, vagy épp a PrimeSense által fejlesztett OpenNI és NITE, amely szintén egy köztes alkalmazás a felhasználó és a célalkalmazás között (MELGAR, 2012).

A gyárilag biztosított környezettel és számos más program közbeiktatásával értük el a robot kéz, vagy akár egyéb más mozdulatokkal történő vezérlését, mivel ez tartalmazza a meghajtáshoz szükséges illesztőprogramot. A fejlesztőkörnyezet letöltése után szükségünk van a FFAST (Flexible Action and Articulated Skeleton Toolkit) nevezetű programra. Ez egy MxR által kifejlesztett program, amely egy digitális csontvázat képez le és annak kulcspontjainak mozgását alakítja át valamilyen előre megadott utasítássá. A program használata során első dolgunk megadni a jövőben használni kívánt módot, amely a felhasználni kívánt területre vonatkozik. Választható csak felsőtest, illetve teljes testre vonatkozó digitális leképezés. Ezt követően a program számos egyéb paraméter módosítását támogatja a mozgás követésével kapcsolatban, mint a mozgás simítása, egyenletesebbé tétele, korrekciós faktor, előrejelzés, Jitter rádiusz (I15: Mielőtt a duplán exponált filter alkalmazásra kerül, a nyers pozíció összehasonlításra kerül az előző képkockán szereplő pozícióhoz. Ha ezek eltérése a Jitter rádiusznál nagyobb értéket mutat, a megadott rádiuszhoz kerül kerekítésre.) és maximális szögeltérés (I15: Hasonló a Jitter rádiuszhoz, ám itt a pozíció a duplán exponált filterrel azonos képkockán kerül összehasonlításra).

Kérdés lehet, hogy a technika mire használható pontosan egy ilyen eszköz irányítása kapcsán. Amennyiben egy folyamatos streaming-re képes kamerát helyezünk el a roboton, annak képe folyamatosan, élőben követhető egy, a még hatótávolságon belülre eső helyszínről. Az irányítás innen egyszerű kézmozdulatokkal történhet. A FFAST program beállításai lehetővé teszik a két, hajtást elősegítő motor szinte bármilyen kéz és lábmozdulatokkal történő irányítását. Legegyszerűbb példa erre a mellkas magasságában

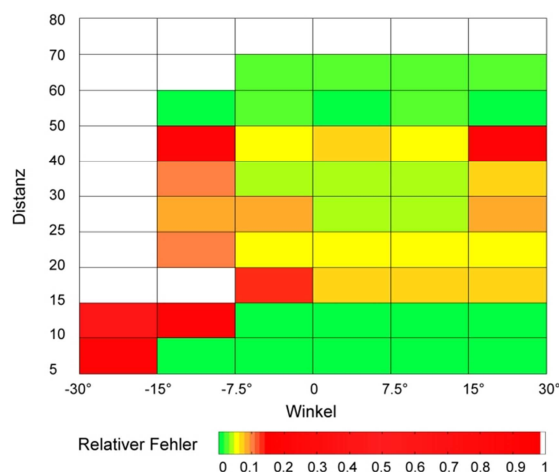
magunk előtt elhelyezett, ökölbe szorított kéz, amely a két motort reprezentálja. Előre, illetve hátra mozdítva kezünket, a motor teljesítményének értéket módosítjuk. Ezzel a felvázolt koncepcióval megvalósul a két motor független vezérlése, amely lehetővé teszi a kormányzást is.

4.3. Felhasznált szenzorok

Mivel az eszköz elkészítése során a modellezés, nem pedig a teljes funkcionalitás elérése volt az elsődleges szempont, teljesen alapvető szenzorokat használtunk, melyek kedvező áron beszerezhetők, de teljes mértékben képesek prezentálni a lehetőségeket. A szenzorokat két csoportba lehet sorolni funkciói alapján. Egyik csoport az irányítást, alkalmazkodást elősegítő szenzorok, míg a másik a produktív, felhasználó által használható adatokat közvetítő szenzorokat tartalmazza.

4.3.1. NXT rendszerhez mellékelt szenzorok

Jelen esetben az NXT rendszer végzi az irányítást, így ehhez kapcsolódnak az ezt elősegítő szenzorok. Ilyen szenzor a gyárilag is mellékelt ultrahangos távolságmérő (9856-os cikkszám), amely felismeri a készülék előtt álló terepakadályt és ezt kikerülve a szintén általa meghatározott optimális irányba halad tovább. A szenzor a készülék elején helyezkedik el, egy standard szervomotor tengelyére kötve, ezzel elősegítve a megfelelő látószög elérését. A szenzor pontosságáról számos mérés készült, amely publikálásra is került. Az eszközzel legkisebb mérhető érték 3 centiméter. Ezt célszerű a program írása során figyelembe venni. A szenzor látószöge a tárgy távolságának függvényében változik.



7. ábra: Távolság és látószögből adódó tévesztések, Forrás: I16

A szenzor mérésének precizitása és esetleges tévesztési értékei tisztán láthatóak a fent mellékelt táblázat segítségével. Az általunk használt rögzítési mód ezt a problémát kiküszöböli és három lépésben több mint 180 fokban képes felmérni a környezetét.

A táblázatból kiolvasható még, hogy a szenzor pontosabb mérést tesz lehetővé a jobb oldali távolságok mérésére. Ennek oka, hogy az ultrahangos távolságmérő működése egy hangszóró, és a hangszóró által kibocsátott 20,000 Hz feletti tartományba eső hangot felfogó mikrofonon alapul. A hangszóró által kibocsátott hang visszatérésének ideje kalibrálható össze adott távolsági értékekkel. Jelen esetben a mikrofon a szenzor jobb oldalán található, így a hang itt érkezik be feldolgozásra. Kisebb tárgytávolságnál a be és kimenet interferálásából adódóan téves értéket kapunk -15 és -30 fokban látószögnél (bal oldalt), mivel ez távol van a jobb oldalt lévő mikrofontól. Ez a tévesztés szoftveresen korrigálható, ám mérlegelendő, hogy a tévesztés nem sokban tér el a szenzor effektív pontosságától. Így ha ennél pontosabb eredmény szükséges az adott feladat végrehajtáshoz, célszerű ezt pontosabb műszerrel helyettesíteni. Az ultrahangos távolságmérő egy nagy hátránya a sarokpontok mérése. Képzeljünk el egy szituációt, melyben a szenzor pont egy kockatest egyik élével áll szemben. Amennyiben az él megközelítőleg a mikrofon és a hangszóró közé ékelődik, közeli tárgytávolság mellett hamis értéket ad vissza. Elég szélsőséges szituáció, ám érdemes megemlíteni.

Két más módszert is használhatnánk, amely az infravörös hullám és lézerrel történő távolságmérés, de e két technika rettentő szűk effektív látószöggel rendelkezik (szinte csak pár fok), ami alkalmatlanná teszi őket a jelen feladathoz, illetve beszerzésük is körülményesebb, hiszen nem mellékelt kiegészítőről van szó.

A szervomotor maga is tartalmaz egy beépített szenzort, amely a tengely által megtett fordulatot méri. Az általa mért érték később visszaolvasható a programunk által, így felhasználható visszatérő műveletek elvégzésére.

A mellékelt kiegészítők közé tartozik még két nyomógomb és egy szín felismeréséhez használható RGB szenzor. Az utóbbi közvetlen kontaktust igényel a vizsgálandó tárggyal. A mérés során a sárga, zöld, piros, fehér, fekete és kék színhez történő egyezőséget méri, melyek átlagolva kerülnek tárolásra eredményként egy adott színt megnevezve (Forrás: I17).

A hőmérsékletmérő szenzor egy olyan alapvető mérőeszköz, amely már a felhasználó által is felhasználható adatot tartalmaz. Régebbi modellek esetében a szett tartalmazta, ám sajnos az újabb generációknál nem mellékelik az eszközt. Ám elkészítése igen egyszerű egy ellenállás segítségével, illetve könnyen kalibrálható egy egyszerű hőmérő és folyadék segítségével.

4.3.2. Külső gyártók által elérhető szenzorok és kiegészítők

Mivel az NXT egy meglehetősen zárt rendszer, aránylag kevés cég foglalkozik kompatibilis szenzorok gyártásával. Ennek ellenére, még ha kevés gyártótól is, számos célt szolgáló eszköz található a piacon. Az általunk később bemutatásra kerülő Vernier Software & Technology szinte laboratóriumi pontosságú műszereket kínál, melynek egyes példányai használhatóak a mezőgazdasági termelés során is.

Emellett a mindsensors.com weboldalon találhatunk olyan kiegészítőket, mint a kamera, numerikus pad, infravörös távolságmérő, feszültség és áramerősség mérő, különböző kontroller interfészek, és ami még fontosabb, port bővítő szettek. Az utóbbira mindenképp szükség van egy komplex rendszer építésénél. Motorok terén a gyári szervomotorokat használva maximum 5 motort, míg standard szervomotorok esetén 8 pontot biztosít, megfelelő külső tápellátás mellett (Forrás: I18).

A HiTechnic Products a gyári szenzorokhoz hasonló kiserelésben készít különböző célú szenzorokat. Az általuk forgalmazott eszközök között találunk többek között egy fokos pontossággal működő forgás érzékelőt, gyorsulásmérőt, barométert, EOPD távolságmérőt, erőmérőt, giroszkópot, infraszenzort és iránytűt (Forrás: I19).

Másik fontos kiegészítőkkal foglalkozó cég a Dexter Industries. Kínálatukban találunk a későbbi kísérletünkhöz szükséges GPS modult, illetve más kiegészítőket is, mint a dSolar nevezetű napelemes áramellátás, illetve a dFlex nevezetű hajlítást érzékelő szenzor. Az alábbi eszközök természetesen közvetlenül csatlakoztathatóak NXT rendszerekre, és a programozását szolgáló blokkok szabadon hozzáférhetőek (Forrás: I20).

Az előbb felsorolt kiegészítők mellett a webáruházból elérhető a Raspberry PI integrációt elősegítő vezérlő, digitális iránytű, Wifi egység, mozgásérzékelő, infraszenzor, nyomásérzékelő és hőmérsékletmérő szenzor. A kiegészítők mutatják, hogy a platform több irányba bővíthető és fejleszthető. A felsorolásból kimaradt Vernier által forgalmazott szenzorok később kerülnek kifejtésre, mivel legnagyobb mértékben ezek kapcsolódnak a jelenlegi témánkhoz.

4.4. Adatgyűjtés

Mivel egy mezőgazdasági robotról van szó, nem mehetünk el szó nélkül az adat logging-gal kapcsolatos lehetőségek előtt. Jelen kísérletezésünk az irányítás megoldása mellett e funkció fejlesztése, illetve az eddigi lehetőségek használatának bemutatása. Szerencsére az általunk használt Lego Mindstorms rendszer már gyárilag rendelkezik több ehhez használatos rendszerrel.

Az NXT rendszer adatgyűjtési módszerei

Az adatgyűjtési módszerek két nagyobb csoportra oszthatóak. Egyik módszer a saját memóriába, azon belül is egy *.txt fájlba történő adatgyűjtés. Mivel a jelen NXT verzió nem támogat külső adattárat, így a beépített 64 Kbyte memóriából kell feláldoznunk erre a célra. Az újabb verzió esetén, mint az EV3, nincs ilyen probléma a beépített SD kártyaolvasó miatt. Működtetéséhez az NXT-G fejlesztőkörnyezet „File access” blokkját kell használnunk, melyben meg kell adni a létrehozandó, vagy már meglévő fájl nevét, ahová az adatokat szeretnénk gyűjteni. Emellett hozzá kell kapcsolni az adott szenzor kimeneti jelét a megfelelő „text”, vagy „number” ághoz. Az itt bemenő adatok továbbíthatóak is, amennyiben szükség van rá, például a kijelző irányába. A létrehozott fájlt könnyű feldolgozni, mivel az értékeket egymás alá helyezi el a program. Amennyiben több szenzor adatát gyűjtsük, célszerű azokat külön fájlba menteni a későbbi könnyebb válogatás és feldolgozás érdekében.

A beépített adatgyűjtés által létrehozott *.txt fájlt elemezhetjük Microsoft Excel táblázatkezelőbe való importálással, illetve bizonyos adatok elemezhetőek már egy kísérleti fázisban lévő saját készítésű program segítségével, amely a megfelelő fájl kiválasztása után importálja az adatokat egy programon belüli táblázatba, amely képes egy vonaldiagrammal szemléltetni az adatok változását. A program használatához a chart könyvtár használata végett mindenképp szükséges a .NET Framework legújabb változata (5. melléklet).

Az előbb említett adatgyűjtési módszer tartogat még lehetőségeket magában. Amennyiben nem egy átlagos szenzor olvasott adatait nézzük, hanem mondjuk a GPS vevő által folyamatosan olvasott szélességi és hosszúsági koordináta mellé jegyezzük a magassági koordinátákat is, a létrejövő nyers adatokat felhasználhatjuk domborzati térképek elkészítésénél. Egy ilyen térkép készítéséhez szükséges valamilyen GIS program, illetve magam is elkezdtem foglalkozni egy hasonló program elkészítésével. Amennyiben az eszköz rendeltetésszerűen van használva, a pásztázás során gyakorlatilag az egész földterület lefedésre kerülne, meghatározott felbontás mellett. Már létező GIS programokat vizsgálva ilyen térképépítő funkcióra használható a 3D Field, Contour Maps szoftvere (Forrás: I21).

Az adatgyűjtés egy másik változata, amikor az az adatok mintavételi frekvenciának megfelelően, folyamatosan érkeznek a célgépre, tehát a művelet közben látjuk az értékeke változását. Ilyenkor valamilyen csatornán keresztül az adatok közvetlenül a számítógépre kerülnek küldésre. Ilyen közvetítő csatorna lehet az egyszerű USB adatkábel, vagy a Bluetooth kapcsolat. Mindkettőnek megvan a maga korlátja, főleg hatótávolság terén, így ez a módszer leginkább tesztelesek esetén használatos. A funkció használatához az NXT fejlesztőkörnyezetet át kell váltani az „NXT Data logging” környezetre, amely a menü azonos

nevű pontjával lehetséges. Ekkor lehetőségünk van a kísérletet elnevezni, majd a várható mérési időtartamot megadni. Emellett meg kell adni a mintavételezés frekvenciáját, illetve a portokhoz kapcsolt szenzorokat. A program futtatása során a folyamat közben tekinthetjük meg egy vonaldiagram segítségével a szenzor, vagy szenzorok által mért értéket. Az adatgyűjtés végeztével kijelölhetünk időintervallumokat, majd a program meghatározza az adott tartományhoz tartozó maximum, minimum és átlag értéket. Az egész méréshez, vagy adott részéhez leírást tűzhetünk, képernyőmentést készíthetünk róla, vagy exportálhatjuk az adatokkal együtt.

LabView használata adatgyűjtésre

A LabView (Laboratory Virtual Instrument Engineering Workbench) a National Instruments által kifejlesztett vizuális programozási fejlesztőkörnyezet, melynek első kiadása 1986-ban jelent meg Macintosh rendszerekre (Forrás: I22). Az „NI LabView for Lego Mindstorms” kiegészítővel az NXT rendszerünk is programozhatóvá válik vele, ám mivel ez struktúrájában nagy hasonlóságot mutat az NXT-G nyelvvel, inkább az adatgyűjtési funkcióira helyezem a hangsúlyt. Az rendszer előnye a mérhetetlen testreszabhatóság. Csak pár egyszerű példát említve a mintavételezés előtt, amennyiben menteni szeretnénk a mért adatokat (melyet jelezni kell egy blokk közbeiktatásával), olyan paramétereket állíthatunk be, mint például a fejléc megjelenése, időbélyeg használata, adatok elválasztásának módja, bináris fájlformátum használata, illetve különböző leírások hozzáadása. Emellett a LabView támogatja az egyedi dashboard-ok létrehozását, melyek lényegesen elősegítik a munkát, hiszen a felhasználó által fontosnak ítélt információkat közli egy egyénileg elrendezett környezetben. Hasonlóképpen kifelé is közvetíthetünk parancsokat, szintén előre beállított módon. Ezek a parancsok lehetnek mozgással, vagy bármilyen előre beállított művelettel kapcsolatban.

5. Következtetések és javaslatok

5.1. Fejlesztési lehetőségek

NXT kompatibilis motorok és szenzorok

A bővítési lehetőségeket, mivel jelen esetben a vezérlőegység fix, első körben a szenzorok teszik lehetővé. Különböző leírások segítségével magunk is készíthetünk a készülék által kezelhető 6 tús, moduláris csatlakozóval ellátott szenzorokat, de rengeteg előre legyártott lehetőség áll rendelkezésünkre. Amennyiben szett formájában vásároljuk meg a vezérlőegységet, az általunk használt 2.0-ás verzióhoz 4 szenzort és 3 szervomotort mellékelnek. A mellékelt szenzorok kísérletezés céljából tökéletesek, de amennyiben valamilyen más célra szeretnénk használni a vezérlőegységet, célszerű egyéb gyári, vagy harmadik fél által gyártott szenzorok körében nézelődni. A gyártó webshop-jában is adódnak lehetőségek, ám ezek leginkább általános célt szolgálnak. Ilyen szenzor például a gyorsulásmérő, iránytű, infravörös szenzor, giroszkóp és a gyárilag is mellékelt eszközök.

Amennyiben specifikusabb céllal tervezzük a bővítést, célszerű más gyártó termékeit is vizsgálni, mint például a korábban említett Vernier, amely talán témánkhoz legszorosabban fűződő szenzorokat forgalmazza és készíti. Jelenleg forgalmazott érzékelőinek száma meghaladja a 60-at és a legkülönbözőbb felhasználási területeken van jelen.

Összegyűjtöttem a kínálatukban lévő, ágazatban is használható szenzorokat (Forrás: I23).

Az ANM-BTA kódnevet viselő anemometer a levegő mozgási sebességét méri. A mérési intervalluma 0,5-30 m/s, míg felbontása $1,2 \times 10^{-2}$ m/s. A szántóföldi növények esetén a talajszárazság egyik meghatározó tényezője a szél és a napsugárzás. A talajnedvesség tudatában annak csökkenése előre jelezhető.

A BAR-BTA nevű barométer az atmoszférikus nyomást képes mérni 0,8-1,05 atm között. A készülék felbontása 1.0026×10^{-4} atm, míg válaszideje 100 mikroszekundum. Csökkenő légnyomás esetén nő a várható esőzés valószínűsége.

A CO2-BTA széndioxid mérő beállításától függően 0-10 000, illetve 0-100 000 ppm mérési intervallumban képes dolgozni, melynek pontossága az előzetes beállítástól függően +-100 ppm eltérést, vagy a valós értéktől való 20%-os eltérést enged meg. A magasabb széndioxid koncentráció elősegíti a növények növekedését. 1270 ppm koncentráció jelenlétében egy tesztelt növény, a tehénborsó 280%-kkal több gázt kötött meg a 450 ppm CO₂ koncentrációjú környezetben lévő társához képest, ami a fejlettségén jelentősen megmutatkozik (Forrás: I24).

A Davis Vantage komplex időjárás elemző eszköz nem feltétlen mozgó járműre készült, ám kompatibilitása révén hasznos információkkal szolgálhat különböző műveletekre vonatkozó döntés végrehajtása kapcsán. Lényegében egy csomagban tartalmazza a szélerősség mérőt, hőmérséklet mérőt, páratartalom mérőt, légnyomásmérőt és csapadékmennyiség mérőt.

ETH-BTA nevezetű etanol mérő szenzor mérési intervalluma 0-3% közé esik, míg pontossága 0,02-0,001%-os hibahatárt enged meg. Az etanol jelenléte csökkenti a növény növekedésének ütemét, bizonyos határ felett pedig a növény pusztulásához vezet. A természetben sosem fordul elő tisztán, csak híg vizes oldatban. A humusz többek között tartalmazhatja kis mennyiségben az etanolt.

Bizonyos esetekben hasznos lehet a FLO-BTA nevű átfolyási sebesség mérő. Mérési intervalluma 0-4 m/s közé tehető, míg pontossága $\pm 1\%$ a készülék adatlapja szerint. Ez használható például öntözés során felhasznált folyadék mennyiségének mérésére, amennyiben ismert a felhasznált cső átmérője.

Az LS-BTA fénymérő egész napos használatával meg tudjuk határozni az adott területre vetülő fény mennyiségét megfelelő időintervallumokra bontva. Három módban használható, melyek együttes, teljes mérési intervalluma 0-150 000 lux. Pontossága a harmadik mód (a maximális érték használata esetén) 50 lux tévedést enged meg. A fény a fotoszintézis kapcsán egyértelműen nagy szerepet tölt be a növények fejlődése során, így annak mérése különböző statisztikák felállításához szükséges.

A GC2-MINI egy komplex, a felhasználási közegben jelenlévő gázok levegőhöz viszonyított százalékos mennyiségének kifejezésére alkalmas eszköz. Mintegy 14 anyag jelenlétét képes meghatározni.

Az O2-BTA eszköz az oxigén százalékos mennyiségét képes meghatározni az adott közegben. Mérési tartománya 0-27% közé esik, míg pontossága 760 higany mm mellett $\pm 1\%$ tévedést enged meg.

A Vernier által forgalmazott pH mérő eszköz mérési tartománya pH 0-14 közé tehető, pontossága a mérési körülményekhez képest változik. A talaj savanyodása köztudottan káros hatással van a növények fejlődésére.

Az UVB-BTA kódú UV szenzor 290-320 nm közötti hullámhossz jelenlétét méri, amely beleesik az UV-B sugárzás intervallumába. Az UV-B fény jelenlegi kutatások alapján jótékony hatással van a növényekre tápérték és egészségesség szempontjából, ám külső fejettségüket csökkenti (Forrás: I25).

Az RH-BTA relatív páratartalom mérő 0-95%-os mérési intervallummal rendelkezik, melynek pontossága $\pm 5\%$. Erre leginkább üvegházak automatizálásánál lehet szükség.

Az SAL-BTA a közegben lévő só tartalmát képes mérni. Mérési intervalluma 0-50 ppt, míg pontossága $\pm 1\%$.

Az SMS-BTA nevű talajnedvesség mérő, amely nevéhez híven a talaj nedvességtartalmát képes mérni. Mérési intervalluma 0-45%, míg pontossága $\pm 4\%$. A szélerősség és hőmérsékletmérő szenzorral kombinálva hasznos eredményt adhat a talaj várható nedvességtartalmával kapcsolatban.

Talán legegyszerűbb szenzornak számít az előbb említett hőmérsékletmérő, amely -40 és 140 °C közötti értéket képes mérni, míg pontossága $0,2-0,5$ °C közé tevődik.

Az előbb felsorolt szenzorok csatlakoztatásához szükséges a BTA-NXT szenzor adapter, amely biztosítja a fizikai kapcsolatot. Programozás terén pedig szabadon letölthető blokkok állnak rendelkezésünkre a gyártó honlapján.

A szenzorok csatlakozási felületét vizsgálva észrevehetjük, hogy a moduláris csatlakozó kiosztásában szerepel a földelés és az 4,3 voltos áramellátás mellett két digitális csatlakozási port. Ez arra enged következtetni, hogy megfelelő átalakító építésével portonként kettő szenzor is köthető az eszközhöz egyes esetekben. Kérdés ennek a szoftveres leképezése, azonban a RobotC fejlesztőkörnyezet bővítményével és egy firmware módosítással az eszköz klasszikus mikrokontrollerként használható, azaz hozzáférést nyerünk a processzor által biztosított lábakhoz.

Érdemes áttérni a szenzorok vitatása után a robot meghatásához szükséges összetevőkhöz. A gyárilag mellékelt szervomotorok (9842-as típuszámú), terhelés nélkül 170 percenkénti fordulatszám mellett, 9 volt áramfelvétel esetén 2,03W mechanikai teljesítményre képesek (41%-os hatékonyság mellett), amely a legtöbb feladat elvégzésére elég, ám vannak lehetőségek az eltérő modellek által a teljesítmény növelésére, ezzel elősegítve a bővítmények által okozott nagyobb súly mozgatását, illetve a hajtómű fejlesztésével együtt a jobb terepjáróképiséget (Forrás: I26). A gyári motorok választéka meghaladja a 16-ot, ám ezen szervomotorok teljesítménye nem éri el azt a szintet, amely megbízhatóan mozgatná a nagyobb tömeggel rendelkező eszközt, ráadásul nehezített terepviszonyok mellett. Ennek kiküszöbölésére megvizsgáltam a motorok csatlakozására szolgáló port kiosztását. Szerkezetileg azonos a szenzorok esetében használt portokkal, ám adatátviteli kiosztása más. Bemenetei sorra az M+, M-, GND, +5V, SCL és SDA (I²C) nevet viselik. Ez szembeesik a ténnyel, hogy standard szervomotorok, melyek +5V, GND és DATA ágat használnak, csak átalakítóval használhatóak.

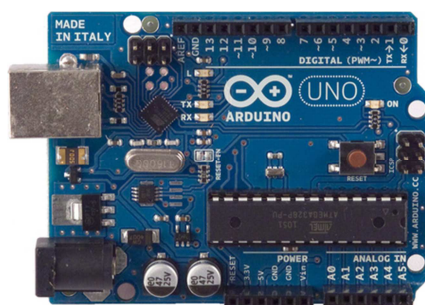
Emellett a tápellátás túlterhelésének lehetőségére gondolva érdemes egy harmadik utat járni, ami a Lego által gyártott Tetrax szettet jelenti. A szetthez csomagolt motorok fém

fogaskerekkel és tengelyekkel rendelkeznek, melyek nagyobb igénybevétel mellett is képesek rendeltetésszerűen működni. A motor teljesítménye annak adatlapja szerint 10,7W, forgatónyomatéka 11,72 kg*cm, 12 voltos üzemi feszültség esetén (Forrás: I15). Mivel gyárilag a kimenet maximum 9 voltos áramellátást tud biztosítani, szükségünk van egy motorvezérlőre, melyet a Tetrix szett biztosít a csomagban.

5.2. Arduino rendszerek és vele kapcsolatos kiegészítők

A dolgozat írásának pillanatában is foglalkozok már az Arduino mikrokontroller adta lehetőségekkel. Míg az NXT egy zárt platform, drága, korlátozott kiegészítőkkel és paraméterezhetőséggel, az Arduino nyílt szabványú rendszerre épül, melyhez alacsony áron széles körben vásárolhatunk szenzorokat, illetve motorokat vagy akár saját magunk is elkészíthetjük őket megfelelő tervrajz segítségével. A még meglehetősen fiatal, 2005-ben alapított cég termékei meglehetősen nagy sikernek örvendenek a mikrokontroller piacon. A platform lelke az Atmel AVR 8 bites mikrokontroller család. Az Arduino lényegében ezeket a kontrollereket úgymond formába öntve árulja. A kontroller minden, a felhasználó által használható lába ki van vezetve egy foglalatba, melyre szabadon csatlakoztathatjuk a kívánt eszközöket. Ezen felül tartalmaz egy soros port konvertert, amely lehetővé teszi az USB-n történő programozást és adatloggolást. Különböző kiadások érhetőek el a piacon, mint a Nano, Mini Pro, LillyPad, Mini, Mega, Tre, Uno, Leonardo és a Due. Ezek leginkább mikrokontrollerük típusában, ezzel együtt ki és bemenetük mennyiségében, illetve a NYÁK alakjában térnek el.

A kísérletezésünk során az Arduino Uno disztribúcióját (R3-as verzió) használjuk.



8. ábra: Arduino Uno, Forrás: I27

A mikrokontroller jelen esetben az Atmel ATmega328 nevet viseli, amely biztosít 14 digitális ki vagy bemenetet (ebből 6 PWM), illetve 6 analóg ki és bemenetet. A NYÁK emellett biztosít 5V és 3,2V feszültségű kimenetet, és két földelést. Az eszköz által használt órajel 16 Mhz (16 MIPS), mely más kristály alkalmazásával az adatlap szerint 20 Mhz-ig növelhető. Memóriája 32 kbyte, ám ebből 0,5 kbyte az első használat során beégetendő

bootloader. Ezen bootloader segítségével használhatjuk a mikroprocesszor programozásához az Arduino által biztosított ingyenes licenzű fejlesztőkörnyezetet.

A teszteléshez a vezérlő mellett egy 4,3g-s szervomotort, DHT11-es hőmérséklet és páratartalom mérő szenzort, HC-SR04-es ultrahangos távolságmérőt, illetve I²C interfész közbeiktatásával egy 16x2 karakteres kijelzőt használtunk. Mindez egy átmeneti kapcsolótábla segítségével került összekapcsolásra.

A megfelelő könyvtárak importálása után három tesztprogram megírása során ismerkedtem meg a rendszerrel. A fejlesztőkörnyezet az Arduino SDK nevet viseli, amely nem rendelkezik annyi kényelmi funkcióval, mint például a korábban használt RobotC vagy az NXT-G. Jelen esetben szövegrész kiegészítés sem áll rendelkezésünkre (mint az mondjuk egy Visual Studio, vagy sok más környezet használata során megszokott), így célszerű egy ezzel kapcsolatos útmutató elolvasásával, vagy más, erre a platformra íródott programok tanulmányozásával kezdeni az ismerkedést, amely leírja az alapfunkciókat. Az importált könyvtárak egyedi funkcióit azok szimpla szövegszerkesztővel való megnyitásával tudjuk tanulmányozni. Mivel még nem rendelkezem elég szervomotorral egy, az előzőleg megépített rendszer elkészítéséhez, olyan alkalmazást hozok létre, amely csak a rendelkezésre álló eszközök kihasználását és a programnyelv megismerését biztosítja. Ez a lépés feltétlen szükséges a későbbi programok létrehozásához.

A rendszer programozása

A program alapja a C++ programnyelv, ám a különböző könyvtársak és a speciális használati kör miatt mégis lényegesen eltér az általános C nyelvektől. A program megírásának első lépéseként meg kell adni a használandó könyvtárak nevét, majd deklarálni kell az egyes portok tulajdonságait. Az ezt követő void Setup() rész egyszer lefutásra kerülő programkódot tartalmaz. Jelen esetünkben itt indítjuk el a soros kimenetet a számítógépen való adatelemzéshez. Inicializáljuk a kijelzőt, majd megadjuk annak tulajdonságait, elérhetőségét. Mivel a kijelző I²C protokollon keresztül csatlakozik eszközünkhöz, így ennek címét egy külön program lefuttatásával tudjuk meghatározni. A setup rész folytatásaként megnyitjuk a szervomotor által használt portot, amivel együtt csatlakoztatjuk a rendszerhez.

Az ezt követő szakasz a void Loop() nevet viseli, ami nevéből adódóan folyamatosan ismétlődő programkódot tartalmaz. Esetünkben itt történik a szenzorok kiolvasása, az ezzel kapcsolatos változók deklarálása, értékadása, ezek soros porton keresztüli küldése, illetve más formában a kijelzőre való kiírása. Mivel a kijelző mérete elég szűkös, így ezt egy if elágazással küszöböljük ki. Alapesetben kiírásra kerül a hőmérséklet és a páratartalom, ám ha az ultrahangos távolságmérő 50 centiméternél kisebb távolságnak megfelelő értéket észlel, a

kijelző ennek értékét fogja kiírni az első sorba, alá pedig a szervomotor tengelyének pillanatnyi pozícióját fokokban kifejezve. Erre azért van szükség, mivel pozíciója a távolság értékéből adódik. A 0-50 közötti távolságmérő szenzor által mért értékét megfeleltetjük egy map függvény segítségével a szervomotor által használható 0-180 fokos értékintervallummal. Azaz közelítve a szenzorhoz egyre nagyobb értéket küldünk a szervomotor felé, mely ennek hatására egyre jobban tér ki a 0 foktól. Minden értékváltoztatás után fontos a szervomotor programkódban történő lecsatlakoztatása, mivel a művelet hiányában folyamatosan azonos értéket küldenénk a motor felé, melynek hatására úgynevezett „servo shake” hatás jön létre. Illetve az ellenőrzés érdekében a loop folyamat késleltetve van 100 milliszekundummal. Összegezve a program lényege az, hogy minden csatlakoztatott szenzortól adatot kérjen be és ezzel műveletet végezzen.

A második program a felhasználó általi adatbevitelt teszteli. Mivel nem rendelkezünk még külső adatbeviteli eszközzel, így soros porton van lehetőségünk adatot bevinni egyes változókba és ezt felhasználni programunk során. Kísérletezésem során létrehoztam egy kisebb programot, amely egy terminál funkcióval rendelkezik. A szenzorok egyes értékeit egy soros porton történő rövid parancs bevitelével érhetjük el. Ezek a parancsok mérhetetlenül egyszerűek, mint a „temp”, „hum”, „dist_cm” és „clr”. A parancsok megadása után a soros monitoron folyamatosan kiírásra kerülnek a kívánt értékek, melyek a „clr” parancs, vagy az előzetesen megadott parancstól eltérő kulcsszó hatására megszakíthatóak. Ismeretlen parancs esetén figyelmeztet, illetve a program visszatér alapállapotába.

A harmadik program a több mikrokontroller közötti kommunikációt hozza létre. Egy Arduino eszközzel már rendelkezem, de ahelyett, hogy több előre elkészített lapot használnánk a teszteléshez, inkább csak több, az Arduino Uno-ban használatos Atmel Mega IC-t szereztem be. Ezek működtetéséhez mindenképp szükség van egy 16 Mhz órajelű kristályra, két darab 22 pF-es kondenzátorra és egy 220k ohm-os ellenállásra. Ez az alap rendszer, amely nem tartalmaz tápellátást, illetve egyéb kényelmi funkciókat, mint a soros port konverter és reset kapcsoló. Az eszköz egy ideiglenes kapcsolási táblán került összeszerelésre. A mikrokontroller adatlapján (Forrás: I28) részletesen leírásra kerül többek között a lábak kiosztása, melyre szükségünk van az összeszereléshez. Az első lábat egy előbb említett ellenállással földeljük, mely gátolja a folyamatos újraindulást. A 9. és 10. lábra kerül a rezonátor, amely az előbb említett kondenzátorral kerül földelésre. Mivel a feszültséggel az Arduino fogja ellátni, gyakorlatilag kész van a működő eszköz. Ám mielőtt programot írhatnánk rá, be kell égetni az Arduino által használt bootloader-t. Ehhez ismételten a meglévő Arduino alaplapot használtam ISP (in-system programmer) rendszerként. A művelet

lényegében abból áll, hogy a meglévő Arduino-ra töltünk egy olyan programot, amely az eszköz programozási portjain befutó jelsorozatot előre meghatározott lábakon küldi tovább. Ezek a lábak pedig össze vannak kötve az új mikrokontrollerünk előre meghatározott lábaival (4. melléklet).

A bootloader beégetése alatt az eszközünket gátolni kell a soros port megnyitásával járó újraindulástól, tehát a reset lábat egy 110 uF értékű kondenzátorral összekötjük az 5V-os kimenettel. A bootloader feltöltése után feltölthetjük a programot a mikrokontrollerre az előzőhöz hasonló módszerrel. A programozás során meg kell határozni, hogy melyik kontroller lesz a master, illetve a slave beállítású. Lényegében egy hierarchiát állítunk fel az eszközök között. Jelen esetben master eszköznek a gyári konfiguráció, míg slave eszköznek az új eszközünk kerül beállításra. A két eszköz Az Analog 4 és Analog 5 porton került összekapcsolásra, melyből a 4-es port az SLC, azaz az órajelért, míg az 5-ös port az SDA, ami az adatátvitelért felelős. A művelet végrehajtásához szükségünk van a „Wire.h” könyvtárra. A void setup() blokkban csatlakozni kell az összes eszközzel az I²C buszhoz a „Wire.begin();” sorral. A slave eszköznél szükséges a zárójelen belül megadni annak sorszámát, melyre később hivatkozni tudunk. Amennyiben a master eszköztől adatot szeretnénk küldeni a további eszközök valamelyikére, a „Wire.beginTransmission(4);” kóddal kezddhetjük meg a műveletet, melyből a 4-es szám a 4-esként deklarált eszközt jelenti címeztként. Ezek után egy „Wire.write();” sorral továbbíthatjuk a változót vagy konstanszt a meghatározott irányba. A fogadó fél egy új blokkban, a „void receiveEvent()” blokkban fogadja az üzenetet.

Kísérletemben a master eszköz egy változót küld a slave eszköznek, melynek hatására az az érték a soros portra kerül kilistázásra, majd egy hozzárendelt ütemben hang szólal meg a slave eszköz 12. lábára csatlakoztatott hangszórón keresztül. Az értéknek megfelelően bármilyen más műveletet elvégezhet a program. A programozás során nem a program komplexitása volt a mérvadó, hanem a technikával való megismerkedés.

5.3. Az Arduino platform előnyei és hátrányai

Az eszköz használata során felmerült első probléma, illetve hátrány a lényegesen kevesebb memória. Ezt más eszközökkel, mint például az Arduino Mega-val ki lehet küszöbölni, vagy akár több eszköz párhuzamos használatával is teljesíthetjük az igényeket. A másik, hasonlóan említést érdemlő probléma a többszálú alkalmazásfuttatás. A kontroller nem támogatja natívan ezt a lehetőséget. Van egy programsor, melybe saját kódunkat ágyazva elérhetünk hasonló hatást, ám ez teljesítmény tekintetében nem tökéletes. Az árat szem előtt

tartva talán legegyszerűbb és legjobb megoldás ismételten csak a több eszköz egyidejű használata. Ezek közötti kommunikáció az előbbiekben említett két analóg porton keresztül közvetlenül, vagy akár egy rádiófrekvenciás xBee adó-vevő segítségével is megoldható. Amennyiben fizikailag, kábelek segítségével csatlakoztatjuk az eszközöket egymáshoz, létre tudunk hozni az I²C protokoll segítségével felé és alárendeltségi viszonyt a kontrollerek között.

Eltérést mutat a szervomotorok használhatósága. Az NXT által biztosított motorok egyidejűleg általános DC motorként is üzemelhetnek, míg az Arduino-nál használtak sok esetben 180 vagy 360 fokos fordulatot képesek megtenni szerkezetük miatt.

Az Arduino által kínált testreszabhatóságához és költségkíméletességéhez az is hozzájárul, hogy építése és programozása a célfeladat komplikáltságától függő elektronikai ismereteket igényel. Viszont pozitívumként szolgál többek között, hogy az előbb említett hátrányok szinte kivétel nélkül kiküszöbölhetőek, így valamivel több munkával egy teljesen testreszabható rendszert kapunk.

Az Arduino rendszer egyéb kiegészítői

Mivel egy egyszerű mikrokontrollerről van szó, szinte bármilyen eszközt képesek vagyunk csatlakoztatni. Gondoljunk csak egy belső számítógép hangszóróra, vagy egy egyszerű LED fényforrásra. Mind olyan elektronikai komponensek, amelyek csak egy meghatározott feszültség értéket igényelnek a működésükhöz. Tehát ezek mindenféle könyvtár és egyéb módosítás nélkül használhatóak pár sor program megírása és a megfelelő lábhoz való csatlakoztatás után. Vagy gondoljunk egy 7 szegmenssel rendelkező LED kijelzőre. Eredetileg ez, mivel gyakorlatilag 7 LED diódáról van szó, 7 kimenetet venne igénybe. De megfelelő meghajtó félvezető segítségével ezt lecsökkenthetjük kettőre. Ez is mutatja, hogy szinte mindenre van megoldás és rengeteg lehetőség van az eszközben.

De amennyiben ennyire nem megyünk bele az elektronikai háttérbe az eszközöknek, lehetőségünk nyílik előre gyártott NYÁK-ra épített, vezérlővel ellátott eszközöket vásárolni, megfizethető áron. Rengeteg internetes webáruház rendelkezik nagy kínálattal ilyen téren. Alapvetően megkülönböztetünk ilyen téren úgynevezett „Shield”-eket, szenzorokat, motorokat és egyéb kiegészítőket. A shield a nevében is mutatja, hogy pajzsként van jelen az eszköz tetején. Gyakorlatilag egy Arduino alaplap méretű nyomtatott áramkör, melynek alsó részén tűk találhatóak. Ezek a tűk kiosztásban és elhelyezkedésben alkalmazkodnak a rendszerhez, tehát csatlakoztatása egyszerű. Ilyen eszköz lehet például a szenzorportok bővítését szolgáló EC-194535 pajzs, a 16x2 karakteres kijelzőt és 6 nyomógombot tartalmazó

EDC-262470 pajzs, az RFID képes EDC-279573 pajzs, a játékvezérlőként szolgáló EDC-269822 pajzs, az Ethernet hálózatkezelő W5100, a motorvezérlő L293D, és a 2,4"-os TFT LCD kijelzővel rendelkező EDC-270550.

Szenzorok terén sokkal nagyobb a választék. Az általunk használt eszközökön kívül rendelkezésre áll többek között az MLX90614 infravörös hőmérsékletérzékelő, folyadékérzékelő, gyorsulásmérő. Lángérzékelő, amely a 760-1100 nm-es hullámhossz közötti tartományt érzékeli. Emellett emberi érintést észlelő szenzor, TCS230 színérzékelő szenzor, UV detektor, etanol gázérzékelő, füstérzékelő, talajnedvesség mérő, esőérzékelő és légtömegmérő. Kiegészítők terén lehetőségünk van még például nagyobb feszültségű eszközök meghajtására szolgáló reléket, IC-eket, SD kártyaolvasót, Bluetooth adó-vevő készüléket, különböző GPS helymeghatározó modulokat, napelemmel rendelkező panelt, kamerákat, illetve minden olyan egyszerű elektronika eszközt, melyeket a mindennapokban használunk, beleértve nyomógombokat és LED diódákat. Kiegészítőkhöz tartoznak még a rendkívül széles teljesítmény és fordulatszám tartományban rendelkezésre álló DC, szervó és léptetőmotorok.

5.4. Felhasználási lehetőségek a jövőben

Az általam elkészített eszközök egyértelműen modellként szolgálnak. Egy kis része csak a lehetőségek széles halmazának. Hasonló méretben alkalmazható az eszköz a már említett adatgyűjtési célokra, amennyiben megfelelő szenzorokkal vértessük fel. Ezzel gazdaságilag hasznot tudunk előállítani, de gondolhatunk a további lehetőségekre is. A cél ehhez hasonló eszközök nagyobb méretben történő elkészítése lenne. Az elektromos motorok már nagyon jó technológiai kiforrottsággal rendelkeznek. A hatásfokuk kimagasló, akár 90% feletti értéket is mutathat. Nyomatékuk és teljesítményük szintén magas értéket képvisel. Erre egy példa a Palo Altóban székelő Tesla Motors gépjárművei. Jelenlegi modelljei 225,2, 269,9 és mintegy 310,2 Kw/h teljesítményű változatban rendelkeznek. A nyomaték, ami akár 600 N*m is lehet, gyakorlatilag álló helyzetből is elérhető, teljesen a kezdeti fordulatszámtól. Hatótávolsága legjobb esetben 480 kilométer, 85 Kw*h-ás akkumulátort feltételezve. Tehát a teljesítmény adott lenne megfelelő vázba való építés esetén nagyobb igénybevételt megkövetelő talajmunkákra is.

Egyedüli sarkalatos pont a működéshez szükséges energia tárolása, hiszen azt feltételezve, hogy a motorok folyamatosan teljes teljesítmény mellett működnek (munkavégzés terhe, illetve a tapadáshoz szükséges súly miatt), valószínűleg csökken annak hatótávolsága. De amennyiben olyan műveletekre használjuk fel a gépet, ami inkább

precizitást igényel, mint teljesítményt, tökéletesen használható technológiáról lenne szó. Ilyen művelet például a vetés, gyomirtás, illetve műtrágyázás. A töltéshez szükséges energia egy részét pedig az egyre csak javuló hatásfokkal rendelkező napelemek is adhatják. Ezt az energiát folyamatosan egy puffer céllal létesített akkumulátorba gyűjtve bármikor felhasználhatnánk.

A műveletek elvégzésére akár teljes autonóm irányítást is alkalmazhatnánk, vagy éppen élőképpel segített távvezérlést. Ezzel a gazda gyakorlatilag otthonról, vagy más pontról irányíthatná a történéseket, függetlenül az időjárástól és egyéb szélsőséges hatástól, mint például a vegyszerezés során keletkező permet.

Számos előnye van a technológiának. Első sorban megemlítendő a sokszor említett precizitáson túl a kis önsúlyból adódó mérsékelt talajtömörítés, illetve a fosszilis energiaforrásokat használó munkagépekhez viszonyított alacsony fenntartási költség.

De egyelőre elegendő lenne jelen projektünket folytatni, amely jelenleg is fejlesztés alatt van. Egyre több szenzort tudok beszerezni, amely egyre több lehetőséget ad a fejlesztésre. A folytatás, amennyiben nem történik semmi ezt megakadályozó tényező, Atmel alapokra fog épülni, ám nem kifejezetten Arduino disztribúció segítségével, hanem saját tervezésű nyomtatott áramköri lapok felhasználásával. Így egy lapon lehetőségünk nyílik több mikrokontroller együttes és specifikus használatára.

De amennyiben nem számolunk ilyen komplex rendszer fejlesztésével, a technika kínál egyszerűbb, de annál praktikusabb és hasznosabb eszközöket is. Gondolok itt például automatikus öntözőrendszerre, amely a talaj nedvességtartalmának és a hőmérsékletnek függvényében indítja el az öntözést. Üvegházak automatizálásához is alkalmas lehet egy ilyen eszköz, mint azt már sok kísérletnél tapasztaltam.

5.5. Költségkalkuláció

A kísérletezés során érdemes a két platformot külön említeni, mivel lényeges árbeli különbségek tapasztalhatóak, illetve ezzel a két platformot ilyen módon is összehasonlíthatjuk.

A Lego Mindstorms rendszer két kiadása került a dolgozatban használható alternatívaként említésre. A régebbi, NXT 2.0 rendszer teljes kiegészítőkkel, ami magába foglalja a korábban említett szenzorokat, illetve 400-600 Lego Technics-ből ismert építőelemet, 84.060 Ft költséget jelent (Forrás: 29). Az újabb, EV3-as verzió hasonlóan teljes kiegészítő csomaggal, amely tartalmazza az új generációs szenzorokat, motorokat és építőelemeket, beszerzési helytől függően 92.600 Ft és 119.000 Ft közötti áron mozog a piacon (Forrás: 30).

Mivel jelen esetben az NXT rendszerhez nem használtunk külső gyártótól származó szenzort, azok nem jelentenek költséget. Viszont érdemes megemlíteni, hogy az ágazat hasznát szolgáló specifikus szenzorok ár igen magas. A gyakran említett dGPS egység ára közvetlenül a gyártótól rendelve 90\$ költséget jelent szállítási díjat nem számolva, amely mai áron átszámítva 22.176 Ft.

A korábban említett Vernier által forgalmazott szenzorok használatához szükségünk van a gyártó eredeti átalakítójára, melynek darabja 50\$ költséget jelent. Ennek segítségével használhatjuk az általunk kívánt szenzort. Jelen esetben csak az ágazatnál használatos szenzorok kerülnek felsorolásra. Többet között a szélerősséget mérő anemometer 122\$, légnyomást mérő barométer 97\$, szén-dioxid mérő készülék 318\$, időjárás elemző központ 538\$, oldott oxigén mérő 530\$, etanol mérő szenzor 149\$, fényerősség mérő 75\$, különböző gázok mérését szolgáló eszköz 2.875\$, oxigénkoncentráció mérő 240\$, pH mérő szenzor 101\$, relatív páratartalom mérő 94\$, sótartalom mérő 118\$, talajnedvesség mérő 130\$, hőmérséklet mérő 36\$, illetve UV fény mérő 137\$ költséget jelent a beszerzés során. Az előbbi árakat vizsgálva hozzá kell tenni, hogy az eszközök precizitása egyes esetekben már-már laboratóriumi pontosságú, illetve minden szenzor előre készített műanyag házban érkezik, mely a szabadkézi használatot is lehetővé teszi.

Ezzel szemben az Arduino platform eszközei lényegesen kedvezőbbek. Egy gyári alaplap (Uno) hazai ára 12.000 Ft körül mozog, kereskedőtől függően. A saját kézzel elkészített példányom megközelítőleg 3000 FT összeget jelentett. Amennyiben az NXT-hez hasonló alapfunkciókat szeretnénk elérni (Kijelző, hangszóró, Bluetooth kapcsolat), ezen eszközöket be kell szerezni. Egy általunk is használt 16x2 karakteres kijelző, I²C interfésszel 3000 Ft, egy hangszóró minimális, pár száz FT-os tétel, míg a Bluetooth modul 4.565 Ft összeget jelent. Az általam használt más szenzorok, mint a hőmérsékletmérő, relatív páratartalom mérő, ultrahangos távolságmérő és 4,3g-s szervomotor, ha az átmeneti kapcsolási táblát és a hozzá szükséges kábeleket is beleszámoljuk együttesen jelentettek megközelítőleg 10.000 FT költséget jelentettek.

Azt feltételezve, hogy minden eszközt készen vásárolunk, illetve mindkét rendszerhez azonos célú kiegészítőket vásárolunk, elkészíthető a két platform teljes költségbeli összehasonlítása. Az NXT rendszer esetében a kereskedelemben kapható kit, amely tartalmazza az alapvető építőelemeket és szenzorokat, mint az ultrahangos távolságmérő, nyomógomb, színszenzor és három darab szervomotor, nem igényli ezek külön beszerzését. Egy csomagon belül hozzájutunk a felsorolt alkatrészekhez. Viszont ha a jelenleg működő Arduino szettemmel tervezem az összehasonlítását, célszerű beszerezni egy, a korábban

említett Vernier által forgalmazott hőmérséklet, relatív páratartalom és talajnedvesség mérő eszközt. A szenzorok költsége így sorra 36\$, 94\$, illetve 130\$, mely mai áron számítva összesen 58.240 Ft költséget jelent. Az NXT rendszer költségével együtt ez 142.300 Ft, melyben az összeszerelés és programozás számos napja nincs kalkulálva.

Az Arduino rendszer esetében a vezérlő maga, mint azt említettem, nem tartalmaz semmit, csak magát a mikrokontrollert. Korábban kiszámításra került, hogy amennyiben egy NXT vezérlő használati fokát szeretnénk elérni, a teljes költség 19.865 Ft. Mivel esetünkben nem áll rendelkezésre építőelem, amely a vázat alkothatná, annak költsége (A korábban leírt egyéni vázat számolva) megközelítőleg 5.000 Ft. A szenzorok költségének számítása során a hőmérsékletmérő, relatív páratartalom mérő és talajnedvesség mérő mellett számolnunk kell ultrahangos távolságmérő költségével is, mivel az nem része a készletnek. A négy szenzor együttes költsége hazai árakat feltételezve 4.633 Ft költséget jelent. A szenzorok mellett szükségünk van szervomotorokra, melynek az NXT rendszerben használatos eszközhez közelítő teljesítményű alternatívája 3.200 Ft, azaz három darabot számolva 9.600 Ft költséget jelent. Összegezve a teljes rendszer megépítése 39 098 Ft költséget jelent, az előbbihez hasonlóan a rá szánt időt nem számolva. Ezen rendszer építése valamivel több időt vesz igénybe annak komplexitása miatt.

A két rendszert összehasonlítva nyilvánvaló, hogy a kedvezőbb árat az Arduino rendszerrel lehet elérni. Viszont mérlegelendő a későbbi használat célja. Amennyiben kísérletezünk az elkészített rendszerrel, mindenképp jobb választás az Lego által kínált platform, hiszen az itt kínált kiegészítők és vezérlők minden tagja saját készülékházzal rendelkezik és a kiegészítők csatlakoztatása is csak egy kattintást vesz igénybe. Viszont ha ténylegesen egy teljes értékű gépet tervezünk építeni, melyet több irányba szeretnénk bővíteni, érdemes az Arduino platform felé tendálni.

Összefoglalás

A dolgozat első oldalaiban betekintést nyerhettük a téma informatikai vonatkozásaiba, illetve a robotika alapelveibe. A robotika témáját tovább bontva megismertük annak különböző alapelveit, kategóriáit. Felsorolásra kerültek ezzel kapcsolatos kutatások, melyek rávilágítottak a különböző eredményekre és célokra. Összehasonlítottam a különböző platformok jellemzőit, programozási környezeteket, illetve különböző kiegészítőkkel (Szenzorok, motorok) való ellátottságukat.

A dolgozat fő témájaként elkészítettem egy több irányítási módot támogató, NXT alapokra épülő robotikai eszközt, amely számos célra használható. Az irányítási módok között autonóm és távvezérelt módban egyaránt kísérleteztem. Az autonóm irányítási mód több programozási környezetben került elkészítésre. Ezáltal tesztelésre került a grafikus (NXT-G) és a szöveges (RobotC) programozási környezet, annak minden előnyével és hátrányával. Az autonóm módon belül GPS általi vezérlés mellett létrehozásra került több verzióban egy terepakadály kikerülő mechanizmus is, amely ultrahangos távolságmérő segítségével térképezi fel a területet. Az első verzió csak kitér az akadály elől, míg a második már ki is kerüli azt a megfelelő szenzorok segítségével.

Az GPS koordinátákkal alátámasztott autonóm irányítás algoritmusaként felvázoltam a vizsgálandó talaj pásztázására alkalmas programot, amely több adatbeviteli módszert támogat, beleértve a közvetlen adatmegadást és a manuális mérést.

A kísérletezésekhez a Lego szettben lévő építőelemeket használtam, ám felmértem a lehetséges igénybevételi körülményeket, és ennek megfelelően terveztem, majd megépítettem egy lehetséges vázat, előbb alumíniumból, majd a súlycsökkentés végett akril felhasználásával. Az egyedi felfüggesztés egyszerre biztosít nagy felületű kontaktust a talajjal és talajegyenetlenségek kiegyenlítését szolgáló mechanizmust.

Az adatgyűjtési módszereket vizsgálva felmértem a belső, illetve élő adatgyűjtés előnyeit, és az ez által gyűjtött információk felhasználási módjait. Az adatelemzési lehetőségeket a gyári fejlesztőkörnyezetben és az NI LabView szoftverben is tanulmányoztam. Felvettem a GPS vevő által mért hosszúsági és magassági koordináták adatai mellé csatolt magassági koordinátákból készíthető domborzati térképének készítésének lehetőségét. Emellett egy saját, ezzel kapcsolatos alapfunkciókat ellátó program gondolta is megfogalmazódott. Az program első tesztverziója is megírásra került C# programozási nyelv alkalmazásával

Bemutattam az NXT rendszerhez használható szenzorokat és kiegészítőket, különböző webáruházak kínálatain keresztül. Itt külön kiemelésre kerültek a Vernier ágazatspecifikus szenzorai és azok alapvető tulajdonságai.

Megismerkedtem lehetséges alternatívaként az Arduino platformmal, melynek segítségével ismertté vált a rendszerek elektronikai háttere, melynek segítségével kijelenthetjük, hogy számos új eszközt képesek vagyunk elkészíteni. Az AVR mikrokontrollerek programozását elősegítő Arduino SDK-ban három program került létrehozásra, melyen célja a programozási környezet megismerése és a konklúziók levonása. Az első program minden meglévő szenzort és kiegészítőt kihasznál, majd az ezek által gyűjtött értékek segítségével interakciót hoz létre. Jelen esetben folyamatosan mérésre kerül a hőmérséklet és páratartalom, melynek értékei láthatóak a hozzá csatlakoztatott kijelzőn és a soros monitoron egyaránt, ám az ultrahangos távolságmérő által mért 50 cm-nek, vagy annál kisebbnek megfelelő érték esetén az adatgyűjtés a távolság értékére összpontosul. Emellett a csatlakoztatott szervomotor is a távolság értékének megfelelően vesz fel 0 és 180 fok közötti értéket.

A második program kapcsán létrehoztam egy terminálalkalmazást, amely a soros porton történő parancsbevitelre válaszol az egyes szenzorok által mért értékkel a soros kijelzőre és a csatlakoztatott kijelzőre egyaránt.

A harmadik program megírására egy jövőbe tekintő terv miatt volt szükség. Az Arduino platform hátrányaként került felsorolásra a memória mérete, illetve a művelet végrehajtás sebessége. Ezt kiküszöbölve több eszköz párhuzamos működését tűztem ki célul. A második eszköz jelen esetben immáron nem az elkészült Arduino alaplap, hanem egy saját készítésű, Atmel mikrokontroller által működtetett vezérlő. Ennek elkészítése megismertette az Arduino alaplap teljes működését. Az előbb említett második egység, mivel nem rendelkezik közvetlen számítógépes csatlakozással, megismertette az Arduino ISP-ként, azaz programozó modulként való használatát. Az előző három program megírása által szerzett ismeret elegendő a korábban NXT-re írt program reprodukálására, és még sok másra.

Felmérésre kerültek a lehetőségek, miszerint a készülék megfelelő szenzor és kiegészítő ellátottsággal rendelkezik egy komplex rendszer előállításához. A kis egységek modellezésével egy ilyen rendszer alapjait képező kísérleteket végezhettem. Az NXT programozása által megismert navigációs és adatgyűjtési technikák, illetve az Arduino programozása során megismert számos programozási metódus segítségével elkészíthető egy, a két rendszer tapasztalatait ötvöző eszköz.

A költségkalkuláció világossá tette számunkra a két platform közötti árkülönbséget, amely leginkább az Arduino platform felé billenti a mérleget ebben a tekintetben. Amennyiben a gyakorlatban is beválnak a tervezett módosítások, amelyek a rendszer hátrányainak kiküszöbölését próbálják elérni, a folytatást mindenképp ebbe az irányba érdemes folytatni.

Ezektől eltekintve a lényegét maga az alap elkészítése, a programozás majd összeszerelés által adódó tapasztalat jelentette, melynek segítségével gyakorlatilag bármilyen irányba képesek vagyunk fejlődni. Az Arduino-ban található mikrokontrollerek sikeresen irányítanak a kísérletek szerint levegőben mozgó adatgyűjtő eszközöket is, ráadásul ezek szerkezete is még a megvalósítható kategóriába tartozik. Ezáltal a klasszikus robotika mellett ezzel az iránnyal is kísérletezhetünk későbbiekben. Az irányítás kezdeti programja, mint azt említettem, sikeresen elkészítésre került. Az adatgyűjtés algoritmus is szintén ismert és működőképes. Már csak egy megfelelő célt kell kiválasztanunk, annak elérését megtervezni, majd az ehhez szükséges szenzorokat beszerezni a rendszer létrehozásához.

Irodalomjegyzék

Enrique Ramos Melgar, Ciriaco Castro Diez (2012): Arduino and Kinect Projects. Apress, 416. ISBN: 978-1-4302-4167-6

Michael Gasperi, Philippe “Philo” Hurbain (2009): Extreme NXT: Extending the LEGO MINDSTORMS NXT to the Next Level. Apress, 360. ISBN: 978-1-4302-2453-2

Kevin. W. Bradley, Bill Johnson, Reid Smeda, Chris Boerboom (2009): Practical Weed Science for the Field Scout Corn and Soybean. University of Missouri Extension

Ancha Srinivasan (2006): Handbook of Precision Agriculture: Principles and Applications. CRC Press. ISBN: 978-1560229551

Josh Baichtal, Matthew Beckler, Adam Wolf (2012): Make: LEGO and Arduino Projects: Projects for extending MINDSTORMS NXT with open-source electronics. Maker Media, Inc. 328. ISBN: 978-1449321062

Pedersen, S. M. – Fountas, S. – Blackmore, S. (2008): Agricultural Robots – Applications and Economic Perspectives. InTech. 370-382.

YAGHOUBI, S. – AKBARZADEH, N. A. – BAZARGANI, S. S. – BAZARGANI, S. S. – BAMIZAN, M. – ASL, M. I. (2013) Autonomous Robots for Agricultural Tasks and Farm Assignment and Future Trends in Agro Robots, International Journal of Mechanical & Mechatronics Engineering (Vol: 13 No:03.) 1-6.

SATISH KUMAR K. N. - SUDEEP C. S. (2007) Robots for Precision Agriculture, 13th National Conference on Mechanisms and Machines

SISTLER, F. (1986) Robotics and intelligent machines in agriculture, Robotics and Automation, IEEE Journal (Volume: 3, Issue: 1) 3-6.

SØGAARD, H. T. AND I. LUND. 2005. Investigation of the accuracy of a machine vision based robotic micro spray system. 5th European conference, Precision Agriculture, 8-11.

Prezentációk

Bart van Tuijl - Eldert van Henten - Yael Edan, *Agricultural Robotic*, (2009)

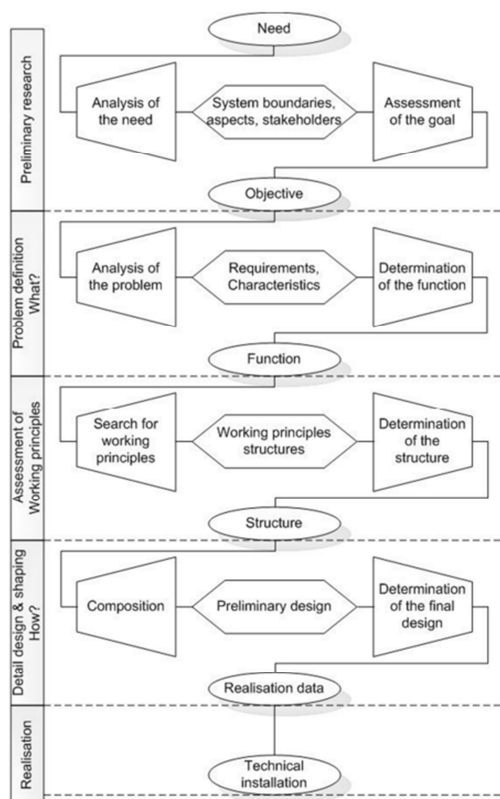
Prof Simon Blackmore, *Robotic Agriculture; Designing systems for the farm of tomorrow*, Harper Adams University, (2013)

Internet

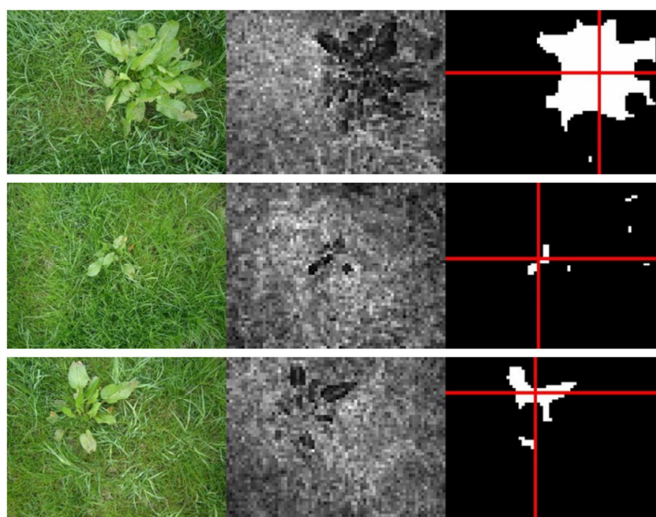
- I1: <http://www.teejet.com/hungarian/home/products/precision-farming-products/gps-guidance/centerline-220.aspx> (2014.01.05.)
- I2: http://www.beam-wiki.org/wiki/Machina_Speculatrix (2014.01.09.)
- I3: <http://www.agromatic.hu/trimble-term> (2014.01.09.)
- I4: <http://www.gpsfarm.com/About/HistoryMilestones/tabid/135/Default.aspx> (2014.01.15.)
- I5: <http://techcrunch.com/2009/10/27/first-agricultural-robot-that-can-autonomously-work-on-fields/> (2014.01.20.)
- I6: <https://sites.psu.edu/almugren/2013/09/19/nxt-robot/> (2014.01.29.)
- I7: <http://botbench.com/blog/2013/01/08/comparing-the-nxt-and-ev3-bricks/> (2014.02.05.)
- I8: <http://autsys.aalto.fi/en/FieldRobot2013> (2014.02.05.)
- I9: http://www.frobomind.org/index.php/FroboMind_Robot:Armadillo (2014.02.06.)
- I10: <http://www.nnu.edu/blogs/robotics-vision/2013/03/06/what-is-robotics-vision-at-nnu/> (2014.02.06.)
- I11: <http://edepot.wur.nl/28203> (2014.02.07.)
- I12: http://static.robotclub.ab.ca/pages/nxt/InventoryComparison/nxt_1_vs_2_vs_edu.html (2014.02.09.)
- I13: <http://www.teamhassenplug.org/NXT/NXTSoftware.html> (2014.02.11.)
- I14: <http://msdn.microsoft.com/en-us/library/jj131033.aspx> (2014.02.25.)
- I15: <http://social.msdn.microsoft.com/Forums/en-US/982d37df-7735-49a3-9821-e52204b7b76e/what-is-the-role-of-jitterradius-transformsmooth?forum=kinectsdknuiapi> (2014.03.15.)
- I16: http://www.tik.ee.ethz.ch/mindstorms/sa_nxt/index.php?page=tests_us (2014.03.15.)
- I17: <http://www.philohome.com/colcomp/cc.htm> (2014.03.10.)
- I18: http://www.mindsensors.com/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=116&MMN_position=40:40 (2014.03.11.)
- I19: <http://www.hitechnic.com/sensors> (2014.03.15.)
- I20: <http://dexterindustries.com/products.html> (2014.02.15.)
- I21: <http://3dfmaps.com/> (2014.03.15.)
- I22: <http://www.ni.com/academic/mindstorms/> (2014.03.15.)
- I23: <http://www.vernier.com/products/> (2015.03.05.)
- I24: <http://joannenova.com.au/2010/04/co2-is-the-magic-gas-that-makes-plant-grow/> (2014.03.09.)
- I25: http://mta.hu/biologia_hirek/ultraibolya-sugarzas-a-novekedesert-126969 (2014.03.15.)
- I26: <http://www.philohome.com/nxtmotor/nxtmotor.htm> (2014.03.17.)
- I27: <http://arduino.cc/en/Main/arduinoBoardUno> (2014.04.01.)
- I28: <http://www.atmel.com/Images/doc8161.pdf> (2014.03.19.)
- I29: http://www.argep.hu/product_1084571.html (2014.03.22.)
- I30: http://www.argep.hu/product_2305722.html (2014.03.09.)

Mellékletek

1. melléklet: Mezőgazdasági robot tervezése, Forrás: Yael EDAN, et al., 2011



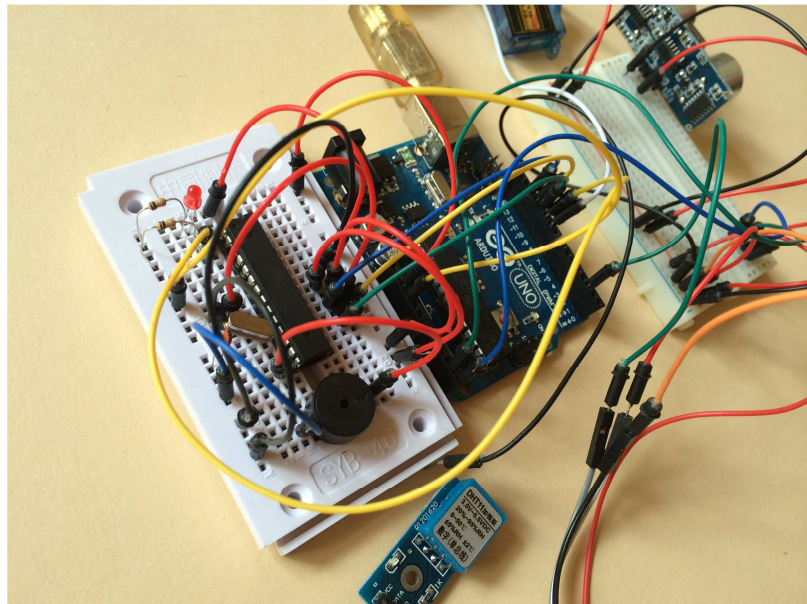
2. melléklet: A 2D Fourier transzformáció, Forrás: I11



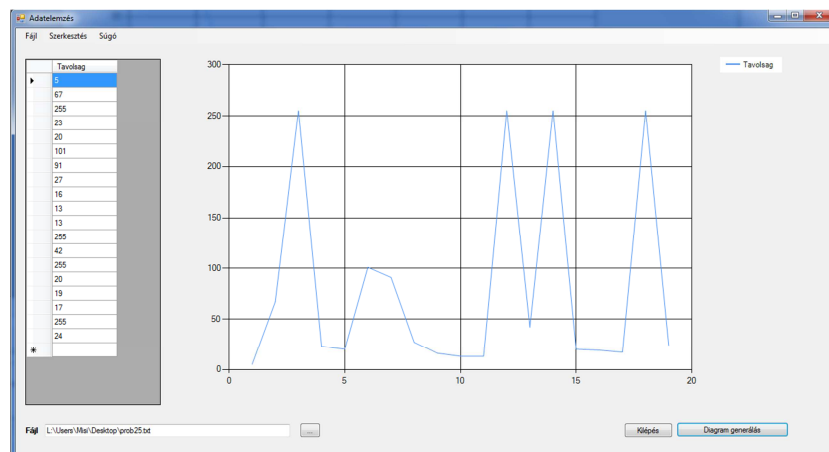
3. melléklet: A Lego Mindstorms NXT rendszer, Saját felvétel



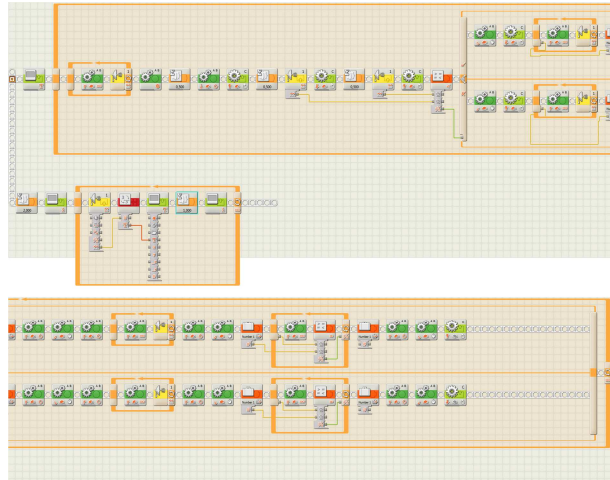
4. melléklet: Az Arduino ISP-ként való használata, Saját felvétel



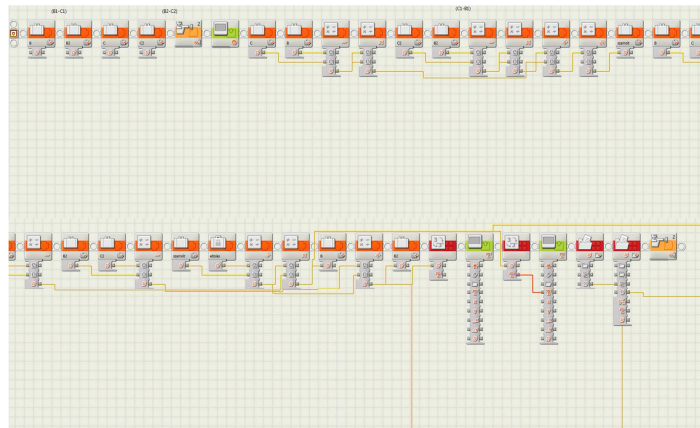
5. melléklet: Saját készítésű adatelemző program kezdeti fázisa, Saját felvétel



6. melléklet: NXT-G-ben készült második kikerülési mechanizmus, adatgyűjtéssel kiegészítve, Saját felvétel



7. melléklet: A pásztázáshoz szükséges számítás folyamatábrája NXT-G-ben, Saját felvétel



8. melléklet: A GPS vezérléshez szükséges adatbevitel programozása NXT-G felületen, Saját felvétel

