A ROBOTINO KÉPESSÉGEINEK KISZÉLESÍTÉSE LABVIEW SEGÍTSÉGÉVEL

EXTENDING ROBOTINO'S ABILITIES WITH LABVIEW

Dániel Balázs, mechatronikai mérnök B.Sc., Budapesti Műszaki és Gazdaságtudományi Egyetem, Mechatronika, Optika és Gépészeti Informatika Tanszék, email: balage.daniel@gmail.com

ÖSSZEFOGLALÁS (ABSTRACT). In this paper I explain the motives of using the mobile robot named Robotino and the reason I changed to LabVIEW instead of using the robot's original software. I give an overview of the improvements which led the Hungarian Hockey Team to come to the third place in a demonstration section of RoboCup 2009 using Robotinos.

1. BEVEZETÉS

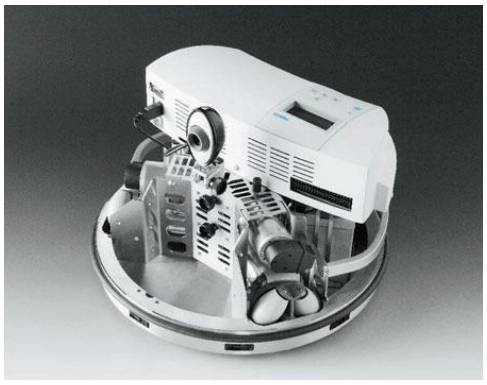
A mai ipari termelés növekedésének feltétele a folyamatok automatizálása. Már napjainkban fellelhetőek olyan ipari robotok, amelyek a hétköznapi értelemben véve inkább önjáró gépeknek tekinthetőek. Ezeknek az úgynevezett mobil robotoknak elsősorban a logisztika területén van nagy jelentőségük, segítségükkel a hosszú és többé-kevésbé egyenes gyártósorok szigorú elrendezése fellazulhat. A félkész gyártmányok megfelelő helyre juttatására a bonyolult konvejor és futószalag rendszerek helyett egy összességében olcsóbb, önjáró szállítógép alkalmazható.

Ha a mobil robotokat összehasonlítjuk a szerelő robotokkal, elmondható, hogy kinematikájuk egyszerűbb, könnyebben kezelhető, azonban önmaguk térbeli helyzetének meghatározása sokkal nagyobb kihívást jelent. Míg egy robotkar talpazata rögzített, így a munkatér és a szerszámtér is definiált, addig egy raktárban közlekedő automatizált targonca már szembenéz a pontos pozíció meghatározás nehézségeivel.

Ebben a cikkben azzal a problémával foglalkozom, hogy miként lehet egy ilyen mobil robot navigációját és akadályelkerülését kifejleszteni és egy adott feladatra alkalmazni. A kihívást a 2009-ben Graz-ban megrendezett RoboCup verseny Festo Hockey Challange Cup elnevezésű nemzetközi mezőnyt felvonultató robotprogramozási megmérettetése jelentette. A felkészülés és a verseny során gyűjtött tapasztalatokat és megoldásokat mutatom be a következőkben.

2. A ROBOTINO BEMUTATÁSA

Minden olyan feladat, amelynek középpontjában a lehetőségek kiterjesztése áll, egy már létező konstrukcióra épül. Ebben az esetben egy elsősorban oktatási célokra készült mobil robotról van szó, amellyel a jövő mérnökei már a középiskolában is ismereteket szerezhetnek programozásból.



1. ábra. Robotino [1]

2.1. A konstrukció ismertetése

A robot mozgását három, úgynevezett Omnikerék biztosítja. Ez holonomikus mozgást tesz lehetővé, amely azt jelenti, hogy két független koordináta mentén egyenes vonalban haladhat, illetve a saját tengelye körül is képes elfordulni. Természetesen ezek kombinációja is lehetséges, amely igen érdekes mozdulatokat

eredményez, és például egy szűk raktárban rendkívüli manőverező képességgel ruházza fel a robotot. A kerekeket PID szabályozással működtetett villanymotorok hajtják megfelelő áttételeken keresztül.

A robot a környezetet a kerületen elhelyezett infravörös távolságérzékelőkkel, egy ütközést érzékelő gumiperemmel, illetve web kamerával képes felderíteni. A szenzorok által szolgáltatott jeleket a fedélzeten elhelyezett ipari számítógép dolgozza fel, az elektromos részegységek áramellátásáról pedig két akkumulátor gondoskodik.

2.2. Programozás

A robot működtetésére többféle üzemmód áll rendelkezésre. Lehetőség van arra, hogy az elkészített programot a fedélzeti számítógép tárolja és futtassa, így teljesen autonóm működés is megvalósítható.

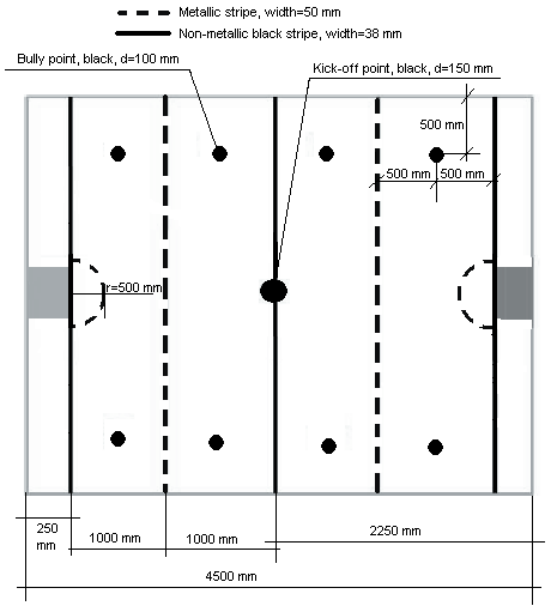
Az általam és csapatom használt megoldás során a robot egy személyi számítógéphez vezeték nélküli hálózaton keresztül csatlakoztatott perifériának felel meg. A robot átküldi a szenzorok jeleit egy nagyobb teljesítményű számítógépnek, az elvégzi a szükséges számításokat, majd utasítást ad a motorok fordulatszám változtatására. Így lehetőség van az egyszerű programok mellett bonyolultabb képfeldolgozási algoritmusok lefuttatására is. A bevezetőben már említett verseny szabályzata ezt az üzemmódot írta elő.

A Robotino fejlesztői lehetőséget adtak arra, hogy a programozók a saját preferenciáiknak megfelelő programnyelvet alkalmazzák, ugyanis mindenki számára lehetővé tették a Robotino API (Application Programming Interface) elérését. Ez a Robotino-t működtető rendszer függvényeinek, eljárásainak és szolgáltatásainak olyan elérési módja, amellyel programozási nyelvtől függetlenül és a belső működés ismerete nélkül is lehetővé válik a robot irányítása [2].

A versenyre való felkészülés során felmerült a kérdés, hogy milyen fejlesztői környezetet használjunk. Végül egy grafikus programozású, adatfolyam elvű program-nyelvre esett a választás, amely a National Instruments LabVIEW programja. Ezzel közeli rokonságot mutat a robot eredeti programozási környezete, a Robotino View is, azonban a LabVIEW által nyújtott lehetőségek sokkal tovább mutatnak mind a programok szervezhetősége, mind pedig a kész algoritmusok tekintetében.

3. A KIHÍVÁS

A robot programozási lehetőségeinek kibővítése akkor vált szükségessé, amikor a 2009-es, első alkalommal megrendezett Robotino Olimpia győztes csapatának tagjaként én is lehetőséget kaptam arra, hogy részt vegyek a Robotino egyik nemzetközi megmérettetésén, a Festo Hockey Challange Cup-on. Itt a feladat olyan robotcsapat programozása volt, amely egyszerűsített szabályokkal megrendezett jégkorong mérkőzést képes lejátszani. A magyar színeket képviselő csapat (Hungarian Hockey Team, röviden HHT) kapitányaként, nekem jutott az a megtisztelő feladat, hogy a felkészülést, illetve a versenyt koordináljam. Természetesen a feladataim között szerepelt a robotok programozása is.



2. ábra. A jégkorongpálya felülnézeti képe

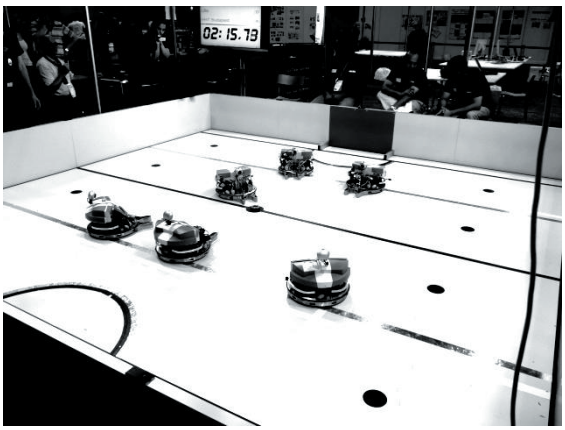
A verseny szabályainak értelmében [3] három robotot kellett irányítani úgy, hogy a játékrészek kezdésekor az irányító

számítógépek egy-egy gombját megnyomva autonóm programot kellett futtatni. A meccs folyamán minden rögzített játékhelyzetben lehetőséget adtak a gomb ismételt megnyomására.

A szabálykönyv rögzítette azt is, hogy a korongot az ellenfél kapujába csak lövéssel lehet bejuttatni. A korong kapu felé tereléséhez, illetve a lövéshez a verseny szervezői készítettek a robot számára egy villát, amelybe a korong becsúszott, így ennek oldalával lehetőség volt az ütésre is. A pályát felülnézetből a 2. ábra mutatja.

A bal oldali kaput védő csapat szempontjából a két szélső gólvonal között négy részre osztott játékrész első negyede a védekező zóna, ahol csak egy robot tartózkodhat egyszerre, a többi három pedig támadó zóna. A kapuk előtt lévő félkörben csak büntetőrúgáskor tartózkodhat védekező robot, míg a támadó játékosnak a félkörön kívülről kell gól szereznie.

A versenyen Németország, Franciaország, Svájc, Egyiptom és Tunézia csapataival kellet megvívnunk először csoportmérkőzéseken, majd az első négy helyért egyenes kieséses rendszerben.



3. ábra. A közelebb eső robotok a Hungarian Hockey Team játékosai, az ellenfél Lille (Franciaország) csapata. Az állás 4-0 Magyarország javára.

Csoportelsőként jutottunk az elődöntőbe, ahol Németország csapata 2-1 arányában győzött ellenünk három mérkőzésen. Lendületes játékunkat passzív, de hatásos védekezéssel törték meg, így gól csak a hosszabbításban, büntetőkből esett. A döntőt Egyiptom ellen játszották a németek, ahol újból sikert arattak, a mi csapatunk a bronzmérkőzésen Franciaországot győzte le.

4. A FEJLESZTÉS ÁLLOMÁSAI

Tekintettel arra, hogy egy teljesen új programozási környezetet akartunk alkalmazni az első feladat a kommunikáció biztosítása volt. Miután ez a probléma megoldódott a részfeladatok közül a kapu megtalálását, megközelítését, majd pedig a lövés indítását kellett kifejlesztenem.

4.1. A kommunikáció átalakítása

Az előzetes tapasztalatok azt mutatták, hogy a vezérlő számítógép, illetve a robot közötti vezeték nélküli hálózat stabilitása nem megfelelő. Az eredeti programfejlesztők az adatátvitel hibáját úgy kezelték, hogy a program futását megszakították és a további működéshez kézi beavatkozásra volt szükség. A verseny szabályzata miatt ezt a problémát át kellett hidalni. Ehhez a Robotino API-t felhasználva új DLL-t írtam, amelyet a LabVIEW is fel tudott használni a kommunikációra.

Ahogy a robot észlelte, hogy a hálózaton hibás adatot kapott, automatikusan megszakította a kapcsolatot és megállt. A LabVIEW vezérlőprogramja ezt érzékelte, majd újra létrehozta a hálózati kapcsolatot, így külső beavatkozásra nem volt szükség.

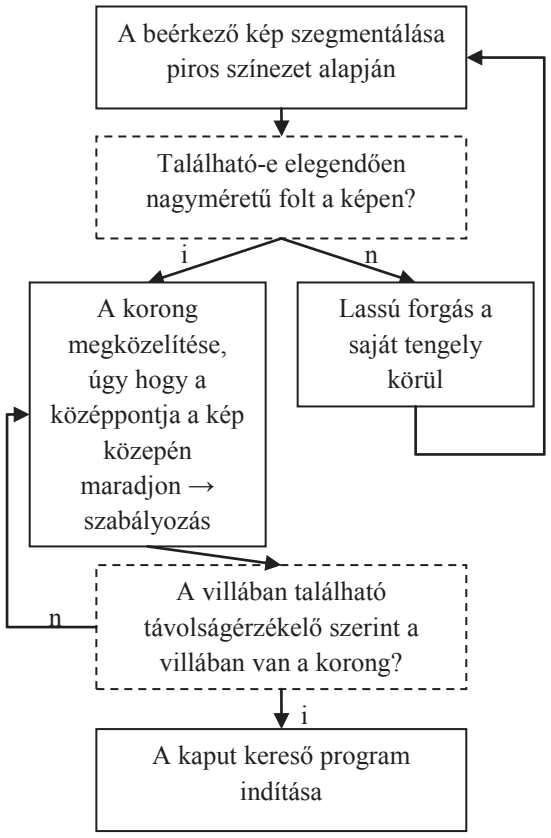
4.2. Képfeldolgozási feladatok

Mivel a robotok számára a pálya viszonylag tiszta, „jól látható” környezetet biztosított, lehetővé vált a web kamera QVGA (320x240 pixel @ 24 FPS) felbontású képének feldolgozása is. A nagyobb felbontás alkalmazása két akadályba ütközött: a kamera alacsonyabb feldolgozási sebessége, illetve a hálózat sávszélessége. Elsődleges szempont a megbízható működés volt, így a kisebb felbontású, de nagyobb frissítési frekvencián működő beállítást választottuk.

A feladat kiírása azt sugallta, hogy színezet alapú képszegmentálást érdemes használni, ugyanis a korong piros színű volt, a két célterület (kapu) pedig kék, illetve zöld színnel volt a pálya oldalán jelölve. Egy támadás tehát a piros színű kisméretű folt megkereséséből, majd a megfelelő színű nagy területű folt megközelítéséből állt. A következőekben ezen feladatok finomságait szeretném bemutatni.

4.3. A korong megkeresése

Mivel a játék célja a gólszerzés, a korong megszerzése és birtoklása gyorsaságot és pontosságot igényelt. A korongkereső algoritmus a következő felépítésű volt:



4. ábra. A korong keresésének blokk diagramja

Ennek a megvalósítása az eredeti Robotino programkörnyezetben is lehetséges lett volna, azonban a szegmentálás bemenő paramétereit (színezet, telítettség, fényerő minimum és maximum értékei) egy saját fejlesztésű, a LabVIEW Vision Toolbox modul felhasználásával írt szubrutin pontosabban beállíthatóvá tette. Az első nagy lépés a környezeti fényviszonyoktól való függőség csökkentése volt, amelyre a saját fejlesztésű színszegmentáló alprogram jó megoldást nyújtott.

4.4. A kapu megkeresése és a lövés indítása

A mérkőzések során a csapatok kapujának színét sorsolással döntötték el, így a programban lehetőséget kellett biztosítani a gyors térfélcserére. A két kapu színének mintáját elmentettük és az indításkor kellett beállítani, hogy éppen melyik kapura kell támadni.

A kapu megközelítésének folyamatában számítani kellett arra, hogy az ellenfél robotjai is „meglátják” a korongot és rátámadnak a mi robotunkra. Ennek elkerülésére nem csak képfeldolgozási algoritmusokat, hanem egy egyszerű elkerülési manővert is alkalmaztunk. A robot kerülete mentén elhelyezett a távolságérzékelők jelezték, amikor oldalról, vagy elölről akadály került az útba. Ekkor az akadály irányával ellentétes kitérő mozdulatot tett a robot. Ez a „kód” részlet a fallal való ütközés kockázatát is csökkentette, hiszen nem tett különbséget robot és fal között.

A következő lépés az ellenfél robotjainak felismerése volt a kamera által szolgáltatott képen. A robotok a saját kapujukkal nagyjából megegyező színű borítást kaptak, azonban az előzőekben említett saját fejlesztésű színszegmentáló program képes volt a megkülönböztetésükre. Ebben az esetben egy cselező mozdulatsor futott le. Azonban szem előtt kellett tartani azt is, hogy ha eközben a robot elveszíti a korongot, akkor meg kell szakítani a „kapu megközelítése” szubrutin futását és vissza kell térni a korong megkereséséhez.

Amennyiben sikerült eljutni az ellenfél kapujának közelébe a következő feladat a megfelelő távolságból elindított lövés megvalósítása volt. A távolság meghatározásához a kapu szegmentált képét alapul véve a képterület nagyságát használtam fel. Nagy távolságból nincs igazán jelentősége robot irányának, azonban a lövéshez olyan

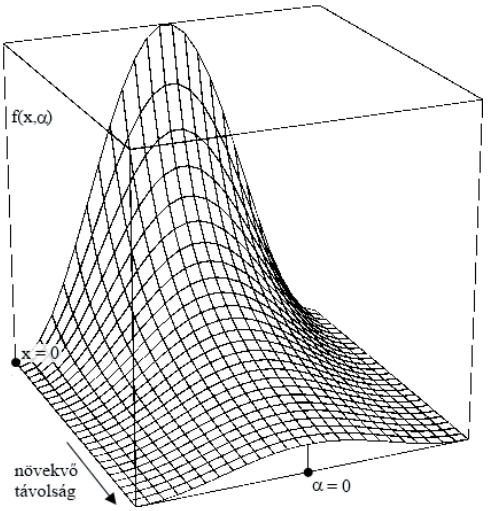
közel kellett menni, hogy a pálya közepétől való távolság már jelentősen befolyásolta a látott terület méretét. Ez egy kétváltozós függvénnyel érzékeltethető, ahol a két független változó a távolság (x), illetve a betekintési szög (α), a függvény értéke pedig a kapu területét adja meg képpontokban. A függvény a következő alakban írható fel:

$$f(x,\alpha) = \frac{1}{\left(x - OF + \frac{1}{\sqrt{R}}\right)^2} \cdot \cos^2 \alpha \quad (1)$$

$$x \in \mathbf{N} \cup \{0\}, OF > 0, R > 0,$$

$$\alpha \in \left[-\frac{\pi}{2}; \frac{\pi}{2}\right] \subset \mathbf{R}.$$

OF jelöli azt a távolságot, amikor a kapu kitöltötte a kamera látómezejét valamely irányban, R pedig az összes képpont száma. A fenti függvény alakját az 5. ábra mutatja. Az ábrázoláson $x \in \mathbf{R}$ és $OF = 0$.



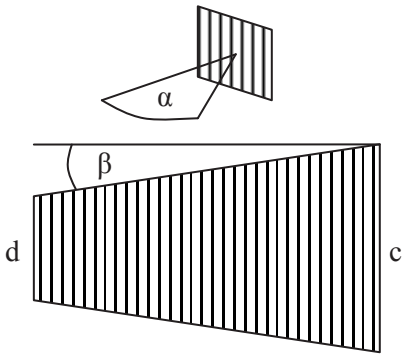
5. ábra. A kapu területének alakulása [4]

Az ábrán látható az a jelenség, ahogy a pálya széléről közelítve a robot közelebbről fog a kapura lőni, mivel a rálátási szög növekedésével a látott terület mérete csökken.

A megoldás nem kívánt olyan pontosságot, amely a függvény teljes megoldását igényelte volna, a gyakorlatban egy korrekciós tényezőt iktattam be a kapu területének számításába. Így kilépési feltételnek elegendő volt beállítani azt az értéket, amikor a robot pontosan szemben volt a kapuval, amennyiben szöget zár be vele, már automatikusan kiszámította a szükséges távolságot.

Ehhez meg kellett határozni a kapuval bezárt szöget. A feladat megoldására először egyszerűbb eszközökhöz nyúltam. A távolságérzékelők jele gyorsan és könnyen feldolgozható volt, első programom erre épült. Amikor a kapuhoz közeli helyről kellett a lövést elvégezni, nem okozott problémát ez a változat, azonban a tesztelések folyamán az optimális lövési távolság meghatározása után már nem volt elegendő a szenzorok által szolgáltatott jelszint.

A megoldást a kapu képének torzulása szolgáltatta. Egy téglalap trapézzá alakul, ha nem a síkjára merőleges irányból tekintünk rá. Jelen esetben a kapu két szélének torzulását használtam fel.



6. ábra. A rálátási szög (α) meghatározása

A 6. ábrán látható jelölésekkel alkossuk meg a következő összefüggést:

$$g(\alpha) = \left| \frac{c(\alpha)}{d(\alpha)} - 1 \right| \quad (2)$$

A (2)-es függvényt mérések alapján lineáris függvénnyel helyettesítettem. Természetesen ez csak közelítés, azonban a tesztek azt bizonyították, hogy az ebből meghatározott,

$$\mu = 1 - g'(\alpha) \quad (3)$$

korrekciós tényező a várt eredményt hozta. A tesztelések során a Robotino a kapu előtti

félkörtől minden helyzetben azonos távolságra állt meg a kapu közepe felé nézve.

Amennyiben a (2)-es egyenlet abszolút értéken belüli kifejezésének előjelét megvizsgáljuk lehetőségünk nyílik annak megállapítására, hogy a pálya melyik oldaláról támadunk. Ezt az információt a lövést végrehajtó programban is felhasználtuk. Abból a feltételezésből indultunk ki, hogy az ellenfél valószínűleg a kapu közepe felé helyezkedik, mivel így tudja a legnagyobb területet védeni. Ezért a mi robotunk mindig a pálya közelebbi széle felé eltérve kísérelte meg a kapuba juttatni a korongot.

A fenti gondolatok szükségessé tették, hogy a szegmentált képből egyéb adatokat is meghatározzunk, mint az alakzat súlypontját. Erre az eredeti fejlesztői környezet már nem nyújt lehetőséget, ennek megoldásához elengedhetetlen volt egy fejlett programozói környezet, amelyet számunkra a LabVIEW nyújtott.

4.5. További részletek, indokok

A versenyen alkalmazott programról nem lehet ezen a cikk keretein belül részleteiben beszámolni, azonban a komplexitását a következőekben vázolom.

A kommunikációt a robotok és az irányító számítógépek között tizenkét programmodul használatával valósítottuk meg. Az adatokat 0,02 másodpercenként kíséreltük meg kiolvasni a robotból a kamera képének kivételével, amely a kamera saját sebességének ütemében történt.

A támadási illetve védekezési szekvenciákat szétbontottunk, minden egyes játékhelyzetre legalább három-négy különböző reakciót megvalósító programot írtunk. A robot bizonyos körülmények között képes volt ezek között a szekvenciák között váltani. Például abban az esetben, ha az ellenfél támadott és a mi játékosunk a kapu előtt védekezett, de a korong bekerült a villájába, átváltott támadó állapotra és megkísérelt pontot szerezni. Az ehhez szükséges logikai hálózat felépítésére kifejezetten könnyen átlátható szerkezetet nyújtott a LabVIEW grafikus, adatfolyam elvű programozhatósága.

5. ÖSSZEFOGLALÁS

A fejlesztés több mint egy hónapnyi, napi 10-12 órás elfoglaltságot jelentett. A kezdeti nehézségek elsősorban a tapasztalatlanságnak voltak köszönhetőek, azonban a végeredmény biztató a jövőre nézve. A verseny első akciógólját a magyar csapat szerezte, illetve a csoportmérkőzések során több mint húsz gólt (találatot) értünk el. A verseny szervezéséért felelős vezetőbíró pedig kijelentette, hogy látszik, hogy a magyar Robotino-k *céltudatosan* mozognak a pályán.

6. KÖSZÖNETNYILVÁNÍTÁS

Köszönöm a Festo Didactic Magyarország hozzájárulását ahhoz, hogy cikkemben a Robotino, illetve a Festo nevét felhasználhattam. Itt szeretném megköszönni a HHT csapat többi tagjának, Raj Leventének, Bolla Dánielnek és a felkészülést segítőknek, Cmerk Andrásnak, Szabó Tibornak, Szabó Norbertnek és Németh Attilának azt az elképzelhetetlen mennyiségű munkát, amely ezt a fejlesztést és eredményt lehetővé tette.

A munka szakmai tartalma kapcsolódik a "Minőségorientált, összehangolt oktatási és K+F+I stratégia, valamint működési modell kidolgozása a Műegyetemen" c. projekt szakmai célkitűzéseinek megvalósításához. A projekt megvalósítását az ÚMFT TÁMOP-4.2.1/B-09/1/KMR-2010-0002 programja támogatja.

7. HIVATKOZÁSOK

[1] Kép forrása: http://www.worlddidacaward.org/award2006/pics/robotino. jpg

[2] Wikipedia: http://hu.wikipedia.org/wiki/Alkalmazásprogramozási_felület

[3] Angol nyelvű szabálykönyv: http://www.robocup2009.org/286-0-Rules

[4] Kép forrása: Online 3D grapher http://www.livephysics.com/ptools/online-3d-function-grapher.php