A ROBOT EVOLUTION ROBOTMODELLEZÕ ÉS MECHANO-ELEKTRONIKAI ÉPÍTÕRENDSZER OKTATÁSI ALKALMAZHATÓSÁGA

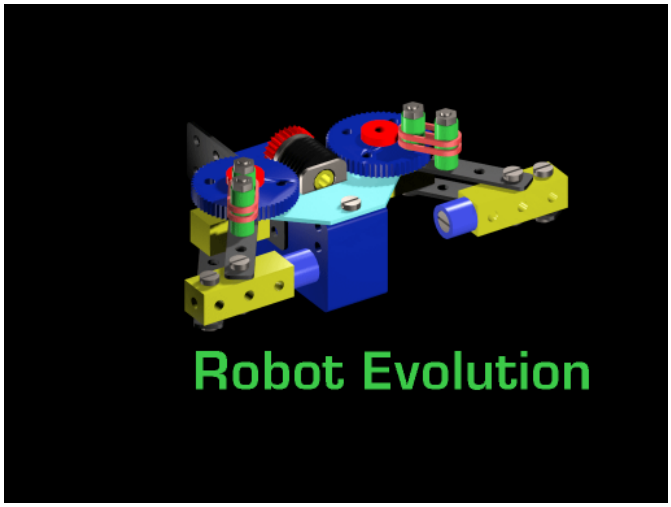
Szánthó Dezsõ, *dezsi@gemini.ektf.hu*
Eszterházy Károly Tanárképzõ Fõiskola Oktatástechnológiai és Informatikai Tanszék

Instructional application of Robot Evolution robot making and mechano-electronic building system

According to the modern pedagogy, the student's activity gets the centre, and his active participation becomes more important. The most important things of the Robot Evolution system are robot making, simulating,which has the secret of the self-discovering method.
Studying of the robots gives you wide-ranging informations. The philosophy of the building set, like an education system, is based on these informations. While the students put the robots together and program them, they can get informations from the most different places. The result-orientation, the possibility of the independent work and the new theme keep the motivating in a high level, so informations are secured effectually. You can build not only robots, but another gadgets from the building system, whatever is possible from the set.

A rendszer pedagógiai elemzése:

A pedagógia korszerû felfogása szerint a hallgató tevékenysége kerül a középpontba, és elõtérbe kerül az ismeretszerzésben való aktív részvétele. Ehhez adhat segítséget az ismertetésre kerülõ Robot Evolution robotmodellezõ és mechano-elektronikai építõrendszer. Az önálló problémamegoldás megtanítása, megtanulása igen fontos helyen áll a tanítás-tanulás folyamatában . Ha erre mind tantárgy mind esetlegesen valamely eszköz oldaláról segítõ lehetõségünk adódik, azt feltétlenül ki kell használnunk. Tipikusan ezt biztosíthatja ez a rendszer, úgy hogy ez a tanulás és tanítás segédeszköze, a tanulás irányítása pedig továbbra is a tanár kezében m arad.



A Robot Evolution természetébõl adódóan a szimuláció és az ehhez kapcsolódó szemléltetés, demonstráció kerül elõtérbe. A jelenségek térben és idõben transzformálhatók. A valóságban csak egyedi, vagy nagy költséggel létrehozható folyamatok laboratóriumi körülmények között elõállíthatók és vizsgálhatók. A különbözõ paraméterek változtatásával a jelenségek kísérleti jelleggel tanulmányozhatók, akárhányszor megismételhetõk, rendszerszemléletû gondolkodásmód kerülhet elõté rbe.

Ez az önfelfedezõ módszer a különbözõ összefüggések felismerésének módszereivel olyan módon ismerteti meg a hallgatót, hogy annak maga is részese lesz, így a megszerzett ismeret rendkívül szilárd. Ugyanakkor a RE rendszer lehetõséget ad arra, hogy mindezt játékos körülmények között tehesse meg.
Pedagógiai szempontból a játék definícióját tekintve kétféle megközelítésrõl beszélhetünk, úgymint a játék és a játszma. A játék, a tanulással kapcsolatban elsõsorban motivációs szempontból jut jelentõs szerephez, a játszma a problémamegoldás, a gondolkodási stratégiák kialakítása szempontj ából kap jelentõséget.

A Robot Evolution rendszerben a lényeg tehát a háttérben meghúzódó szimuláció, amelynek kivitelezése játékos önfelfedezõ módszerben rejlik. Ez a módszer olyan célkitûzéseket is szolgál, mint a döntéshozatal készségének fejlesztése, az interperszonális kommunikáció erõsítése és olyan attitûdök kialakítása mint a mások véleményének figyelembevétele, konzultációs készség, az adott probléma többféle megoldásának vizsgálata, értékelése.

A Robot Evolution rendszer általános elméleti alapjai:

Az olyan berendezéseket, amelyek bemenõ adatként valamilyen információt kapnak az õket körülvevõ környezettõl, kimenõ adatként pedig új információt adnak ennek a rendszernek, információs gépeknek nevezhetjük. Ezen kritériumoknak a vizsgált rendszer (RE) megfelel. Az információs gépek általános , absztrakt matematikai elmélete pedig nem más mint az automaták elmélete. Itt azonnal felvetõdik a kérdés hogy milyen automatákat lehet ilyen absztrakt módszerekkel elõállítani, milyen formális nyelvek kapcsolhatók hozzájuk.

Lényeges a formális nyelvek kérdését megvizsgálni, ugyanis ez az információ-feldolgozásban azért fontos, mert ha az egyes adathalmazokra úgy tekintünk mint véges sok jelbõl alkotott szavakra, akkor a feldolgozásuk során alkalmazhatjuk a formális nyelvek kezelésére kidolgozott automatikus módszereket.

Az automata, mint a vizsgált Robot Evolution, bizonyos bemeneti adatok alapján új adatokat szolgáltat kimenetként, és ezt úgy valósítja meg, hogy összesen csak véges sok különbözõ belsõ állapotba kerülhet, azaz véges a memóriája. Így a formális nyelvek kezeléséhez megadható módszerektõl megkövetelhetõ, hogy az adott nyelvet véges eszközökkel írják le.

A Robot Evolution az automaták alapvetõ típusai közül teljesíti az ún. *felismerõ automata* kritériumait. Az automata bemenõ adatai egy adott véges halmazból, az ún. bemenõ ábécébõl alkotott véges jelsorozatok (szavak), kimenõ adata pedig egy bináris szimbólum. Bekapcsoláskor a felismerõ automata egy meghatározott kezdõállapotban van, s mûködése determinisztikus, azaz ha ugyanazt a szót adjuk be bemenetként különbözõ pillanatokban a gép ugyanúgy fog válaszolni.
Ugyanakkor a RE teljesíti az *átalakító automata* ismérveit is, azaz egy adott bemeneti nyelv szavait alakítja át a kimeneti ábécé szavaira, így az új szavak egy új kimeneti nyelvet is meghatároznak. Mûködése szintén determinisztikus.

A rendszer lehetséges alkalmazási területei, néhány egyszerû példán keresztül:

A robotok a legbonyolultabb és legfejlettebb gépek. Megtestesítik a mûszaki haladás szinte valamennyi eredményét a kohászattól az informatikáig. A robotok mûködésének megértéséhez széleskörû ismeretek szükségesek, de ez fordítva is így van: a robotok beható tanulmányozása széleskörû ismereteket nyújt. Erre épül az építõkészlet, mint oktatóeszköz filozófiája. A robotok összeállítása és programozása során a tanulók a legkülönbözõbb területekrõl szereznek ismerteket. Az eredményorientáltság, az önálló alkotás lehetõsége és a téma újszerûsége a motiváltságot mindvégig magas szinten tartja, így az ismertek hatásosan rögzülnek. Az építõrendszerbõl, nemcsak robotokat, hanem másféle szerkezeteket is lehet készíteni, gyakorlatilag bármit, amit az elemkészlet lehetõvé tesz. Nyilván e kérdésben a hallgatók ismeretszintje és a megvalósítani kívánt pedagógiai cél dönt a fe lhasználás módjáról.

Természetébõl adódóan az általános iskola felsõ tagozatán és a középiskolákban különbözõ tantárgyakhoz köthetõ, ezek közül a technika, számítástechnika és a fizika a kiemelkedõ. A technika tantárgyában a gépelemek, közlõmûvek témakörétõl kezdve az egyszerû vezérlési és szabályozási problémák megfogalmazásáig és megoldásáig alkalmazható. Számítástechnika terén elsõsorban a programozás kapcsán, igen hasznos szerepet tölt be az algoritmikus gondolkodásmód fejlesztésében . A fizika tantárgyban a sokoldalú modellezési lehetõség miatt számíthat sikerre. Alkalmazható szakközépiskolákban és szakmunkásképzõkben minden olyan tantárgynál ahol bizonyos géptani, jármûszerkezettani, villamos jellegû, kommunikációs irányú vezérlési, szabályozási kérdések, vagy egyszerû modellezési igények lépnek fel. Egyszóval minden olyan képzési területen jól alkalmazható, ahol a szimuláción alapuló rendszerszemléletû gondolkodásmód fejlesztése szükséges.

A rendszer ismertetése:

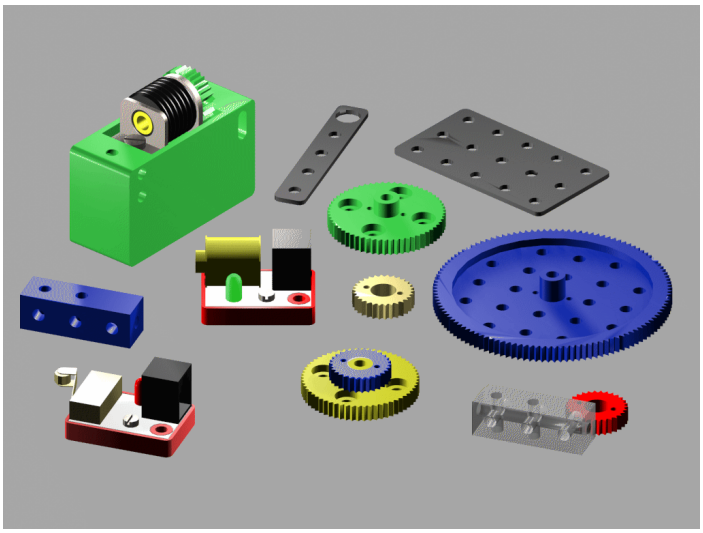
A Robot Evolution egyszerû robotok megépítését, valamint azok mûködését igyekszik bemutatni úgy, hogy azok akár számítógéppel is vezérelhetõek legyenek.

Az építõkészlet legfontosabb eleme a vezérlõberendezés. Ez a mikroprocesszoros egység irányítja a hajtómûveket, kapcsolatot tart az input és output elemekkel. Kétirányú kapcsolat építhetõ ki a számítógép és a vezérlõ között. Ez az egység önmagában is , vagy számítógéppel összekapcsolva is használható az adott feladat megoldásához. Érdekes lehetõség , hogy a vezérlõk egymással is összekapcsolhatók, így ennek segítségével mintegy megsokszorozhatók annak képességei. A legfontosabb ismérve azonban az, hogy képes egy adott program betanulására, tárolására és futtatására.

Alapvetõen a következõ üzemmódjai lehetségesek:

- *Kézivezérlõ mód* , ilyenkor a nyomógombokról kapott közvetlen parancsokat teljesíti
- *Programozó üzemmód* , tanító típusú programozás, azaz a mozdulatokat valóban végre kell hajtani, mint a kézivezérlõ módban, de azok most tárolhatók (max. 127 lépés).
- *Végrehajtó üzemmód* , a tárolt program futtatása, amely történhet a mozdulatsor elejérõl vagy a végérõl kezdõ irányba, továbbá lehetõség adódik a program lépésenkénti, valamint ciklikus, azaz ismétlõdõ végrehajtására
- *PC interface üzemmód* , ekkor a vezérlõ a számítógép printer portjával összeköttetésbe kerül, továbbá a gépen elindítható a RE programja, amely felépítését tekintve azokat a szolgáltatásokat biztosítja mint a vezérlõegység önmagában, de ebben az üzemmódban lehetséges elõre megírt programok felhasználása, a tanult vagy beírt program szerkesztése, lemezre mentése és vissza töltése is.

A vezérlõberendezésen kívül az alapkészlet 4db komplett hajtómûvet tartalmaz, amelyek egyenáramú motorok, jeladóval és különbözõ áttételi elemekkel kiegészítve.
A rendszer részét képezik a közlõmûvek, amelyek itt különbözõ fokszámú fogaskerekeket és menetes orsókat jelentenek, valamint az input és output elemek, továbbá vázelemek és kötõelemek.



Rendkívül elõnyös a termék nyílt architektúrájú felépítése, lehetõség van az elemkészlet igény szerinti változtatására és bõvítésére. Jelenleg is forgalmazás alatt áll egy kisebb teljesítményû vezérlõberendezés és egyszóval a fejlesztõ kft. adaptálni szeretné a legújabb mûszaki eredményeket, új rendszerelemek kifejlesztésével be kívá nja mutatni a fejlõdés új állomásait.

Fejlesztés alatt áll a beszédszintetizátor, a távvezérlõ és telemetriai egység, valamint egy újabb nagyobb teljesítményû vezérlõegység. A meglévõ PC-s szoftver is folyamatos átalakuláson megy keresztül, mindez oly módon, hogy a felhasználónak is joga és lehetõsége van azt továbbfejleszteni, ehhez a gyártó minden támogatást megad.

Igen komoly alkotói tevékenység eredményeképpen az APC Stúdió és a Robot Evolution Kft. az APC LOGO 3.2 verziójába integrálta a vezérlõberendezés LOGO nyelven történõ programozási lehetõségét, ami igen komoly elõrelépés abban a tekintetben, hogy a rendszert egyszerû programlépéseken keresztül tudjuk üzemeltetni, illetve a kitûzött feladatokat megoldani.

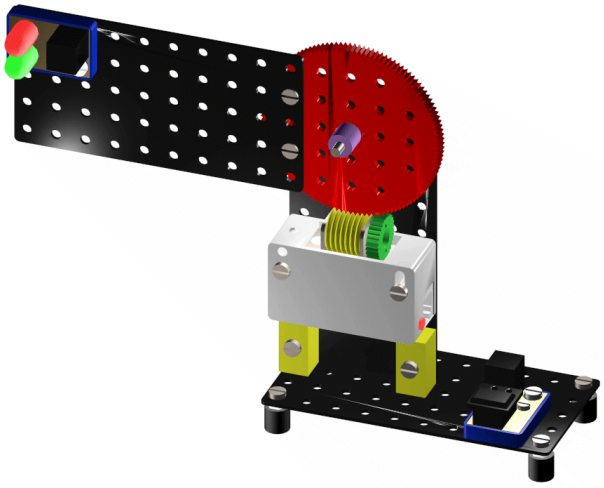
Egy példa a Robot Evolution rendszer ezirányú alkalmazási lehetõségérõl:

```
2,":ol :n ism :n
elore :ol
jobbra 360/:n
ismvege
","NSZOG"
1,":ol ism 4
elore :ol
jobbra 90
ismvege","NEGYZET"
0,"szivacs
kozep
toll
orr
fejszog 0","INIT"
1,":ol ism 3
elore :ol
jobbra 120
ismvege","HAROMSZOG"
1,":ol ha :ol>0
 kor 20 ;
kor 10
negyzet 40
vege
","KOROK"
0,"init
helyez 160 190
ism 200
balra 10
ha :veletlen > 5
  jobbra 20 ;
elore 1
ismvege","VELETLEN"
2,":ol :n ism :n
nszog :ol 5
jobbra 360/:n
ismvege","SOK"
0,"legyen :lpos :m5
orr
kozep
hurok
ha :m5 > :lpos
elore :m5-:lpos
legyen :lpos :m5
;
amig :s5 = 1

","KIIR"
0,"hang 523 1
vege","DO"
0,"hang 589 1
vege
","RE"
0,"hang 658 1
vege","MI"
0,"hang 699 1
vege","FA"
0,"hang 782 1
vege
","SZO"
0,"hang 880 1
vege","LA"
0,"hang 990 1
vege
","TI"
0,"hang 20000 1
vege","SZUN"
0,"ism 2
do re mi re mi re do szun
do re mi re do szun do szun
ismvege
szo fa mi re mi re do szun
do re mi fa szo szun szo szun
szo fa mi re mi re do szun
do re mi re do szun do

","SZUNYOG"
0,"ism 2
ism 2
hang 523 2 hang 658 2
ismvege
hang 782 4 hang 782 4
ismvege
hang 1046 2 hang 990 2
hang 880 2 hang 782 2
hang 699 4 hang 880 4
hang 782 2 hang 699 2
hang 658 2 hang 589 2
hang 523 4 hang 523 4
vege



","BOCI"
0,"hang 1046 1
vege","DOO"
0,"do re mi fa szo la ti doo
szun
doo ti la szo fa mi re do","SKALA"
0,"hurok
ird :m1
amig :s1 = 1
","MVAR"
0,"init
ird .1-es,2-es .gomb .forgat .a .0-as
.vonalat .huz
hurok
legyen :b :bill
ha :b = 48 elore 5 ;
ha :b = 49 balra 45 ;
ha :b = 50 jobbra 45 ;
amig :b > 32
","RAJZOLO"
0,"szivacs
toll
varj 1
helyez 0 100
legyen :a 5
legyen :b 180
ism 10
iv :a :b
legyen :a :a + 3
legyen :b :b * -1
ismvege
vege
","HULLAM"
1,":b motor ? :b
hurok
ird :m?
amig :s? = 1
ird :m?
","MOTIR"
0,"do re mi fa szo la ti doo szun
","SKALA1"
0,"doo ti la szo fa mi re
do","SKALA2"
0,"ird .Ha .valaminek .nekimegyek,
.megallok.
ism 5 do szun szun ismvege
ird .MEGYEK!
perm 1 10
hurok
mi
amig :i2 = 0
motor 1 -1
hang 1000 10
ird .Itt .az .akadaly!","PERMAK"
0,"toll
kor 3
tollne
vege","AKAD2"
0,"legyen :m1 0
orr
helyez 10 100
fejszog 90
akad2
vege","AKAD1"
0,"amotor 1
legyen :a :i1
ird .Motor :amotor
hurok
legyen :k :bill
amotvalt
ha :k = 333 mot 1 ird :m? ;
ha :k = 331 mot -1 ird :m? ;
ha :k = 372 motir 10 ;
ha :k = 371 motir -10 ;
amig :k > 13
","KEZI"
0,"motnul
mo 1 60 mo 2 25 mo 1 -15
mo 2 -25 mo 1 90 mo 2 25
mo 1 20 mo 2 -25 skala1
mo 2 25 mo 1 -20 mo 2 -25
mo 1 -90 mo 2 25 mo 1 15
mo 2 -25 mo 1 -60 skala2



","KETMOTROB"
0,"ism 1000
hurok perm 1 1000 ird .zold amig :i1
= 1
hurok amig :i1 = 0
hurok perm 1 -1000 ird .piros amig :i1
= 1
hurok amig :i1 = 0
ismvege
```

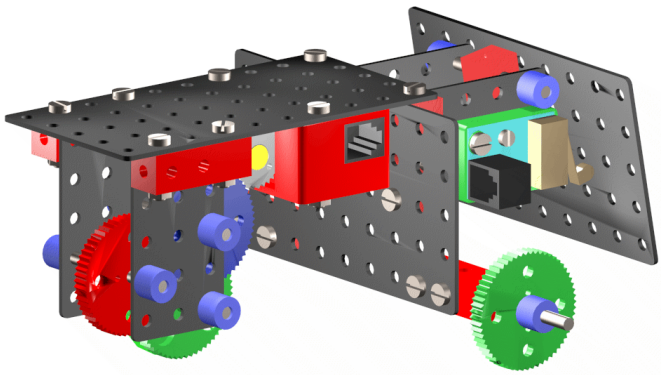
```
","KAPCSVALT"
0,"ird .Naplo: .a .sorompo .zarva
.tartasanak .idotartama 1000 -
:idozito","SORIR"
0,"motnul
robot .status","TOTNUL"
0,"ird .sorompo .leengedes .ideje
.%d%I","SORIR1"
0,"ird .sorompo .felnyitas .ideje
.%d%i
ird . ird . ird .","SORIR2"
0,"legyen :q 0
akad1
hurok
mo 1 1
elore 10
ird .Az .ut :m1 * 5
ha :m1 > 30
ird .nincs .akadaly
mo 1 -:m1
legyen :q :q + 1
akad1
varj 5 ;
ha :q = 3 vege ;
amig :i2 = 0
akad2
do
ird .Az .akadaly .tavolsaga .mm-ben
:m1 * 5
mo 1 -:m1

","AKADALY"
0,"skala
mo 10 10
mo 9 45
mo 10 -23
ism 2
perm 11 99 ird .magnes .fog
skala1
mo 10 40
mo 9 90
mo 10 -30
perm 11 0 ird .magnes .enged
mo 10 30
mo 9 -90
mo 10 -40
ismvege
perm 11 -1
ird .eleg
mo 10 40
perm 11 0
mo 9 -45
mo 10 -27



","MAGNESKEZ"
0,"legyen :x :amotor
ha :k = 328 legyen :x :x + 1 ;
ha :k = 336 legyen :x :x - 1 ;
ha :x = 0 legyen :x 1 mi ;
ha :x > 12 legyen :x 12 mi ;
amotor :x ird .Motor :amotor
legyen :q :i?

","AMOTVALT"
2,":c :b amotor :c
legyen :x :m?
motor ? :b
hurok ha :s? = -1 robot .stop vege ;
amig :s? = 1
hanem :m? - :x = :b ird .A .kovetkezo
.szamu .motor .nem .mukodik :amotor
;
","MO"
3,":v :x :y perm :v 100
varj :x :y
perm :v -100
varj :x :y","KETVILL"
3,":v :x :y perm :v -100
varj :x :y
perm :v 0
varj :x :y","PIROSVILL"
3,":v :x :y perm :v 100
varj :x :y
perm :v 0
varj :x :y","ZOLDVILL"
0,"hurok amig :i? = 0
hurok amig :i? = 1
","INPNEZ"
0,"perm 6 0
hurok
amotor 2
inpnez
perm 6 1000 inpnez
perm 6 -1000
inpnez
perm 6 0
amig 1 = 1
","BEMUTVILL"
0,"output 7 20
perm  7 3200
hurok
hurok amig :i6 = 0
legyen :idozito 1000
sorir1
perm 6 -200 do
perm 7 -200
mo 5 -90
hurok
pirosvill 7 0 2 mi
amig :i6 = 0
perm 7 -100
perm 6 200
mo 5 90
perm 7 200
sorir
sorir2
amig 0 = 0
","SOROMPO"
0,"legyen :a 1
ism 12
amotor :a
legyen :m? 0
legyen :x :i?
legyen :a :a + 1
ismvege
","MOTNUL"
0,"hurok
legyen :b :i1 legyen :c :i2
ha :b > :c mo 4 -2 ;
ha :b < :c mo 4 2 ;
amig 1 = 1
","KETKAPCS"
0,"ird .Kerem .az .impulzusokat!
legyen :m2 0
hurok
legyen :idozito 5
input 2 1 5
ird .impulzus .szam :m2
amig :idozito > 0
skala1
ism :m2
mo 1 1
varj 0 1
ismvege
skala2
mo 1 -:m2
","IMPVAR"
0,"ism 100
mo 1 10
ism 10
ism 50
ketvill 7 0 0
ismvege
ism 30
ketvill 7 0 1
ismvege
ism 10
ketvill 7 0 3
ismvege
mo 5 20
output 7 -20
mo 5 -20
output 7 20
mo 1 -10
skala
ismvege","KIALLIT"
```

SOROMPO eljárás

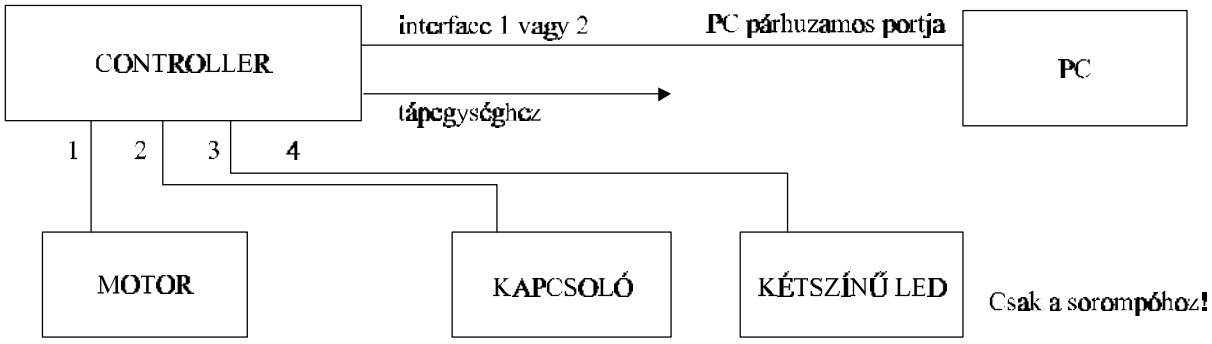
A SOROMPO eljárás a megépített sorompo modellhez készült ezért az eljárás tanulmányozása elõtt a modellt tanácsos megépíteni.



Ez egyébként a legegyszerûbb modell, melyet az Építési tanácsadó füzet javasol.
Az AKADALY a PERMAK és a IMPVAR eljárások az egymotoros kocsihoz készültek, ezért javasoljuk, hogy az eljárások tanulmányozása elõtt építsék meg az egymotoros kocsit.



A három eljárás egymás utáni kipróbálása jól mutatja, hogy ugyanaz a mechanikus szerkezet, az egymotoros kocsi mennyire más és más lesz, pusztán a mûködtetõ program cseréjével. Megjegyezzük még, hogy a kocsi a kis- és a nagy RobotEvolution készletbõl egyaránt megépíthetõ. A kocsi vezérléséhez a kétféle CONTROLLER bármelyike felhasználható. A kocsi motorját a CONTROLLER 1-es, a kapcsolót a 2-es csatornához kell csatlakoztatni. Ne felejtsük el a CONTROLLERT tápfeszültséggel ellátni és a PC párhuzamos portjához csatlakoztatni. Ha mindez megtörtént, csak akkor indíthatjuk a LOGO nyelvet az LG ROBOT utasítással. Ennek hatására a ROBOT.LOG eljárás csomag automatikusan betöltõdik.



Az egymotoros kocsi, a CONTROLLER és a PC összekapcsolása. A Sorompo bekötése ugyanez, de ott még egy output elem van a két színû lámpa(LED). Ezt a 3-as csatornához kö ssük.

MOT eljárás

Amikor a PC a LOGO program motor x y utasítását hajtja végre, az adott motorra vonatkozó parancsot küld a CONTROLLER egység felé. A CONTROLLER hozzákezd a parancs végrehajtásához, bekapcsolja a kiválasztott motort és figyeli hogy megtörtént e a kívánt számú motor fordulat. Ha igen, a forgás leáll. A PC a forgás alatt is lekérdezheti a CONTROLLERT a mozgás állapotáról. A kapott választ az :s változóba kerül. Ha a mozgás még tart, az :s értéke 1, ha befejezõdött :s értéke 0. Amíg az egyik csatornára kötött motor még forog, a PC illetve a LOGO ugyanannak a CONTROLLER-nek másik csatornájára kötött motorját is bekapcsolhatja. Mivel a CONTROLLER csak egyszerre csak egy csatornát kezelhet, így az elsõként bekapcsolt csatorna motorja megáll, és a második utasítás végrehajtása kezdõdik el. Azaz forogni kezd a másik motor. Ennek akkor van értelme, ha több CONTROLLER van a PC-hez kötve és amíg az egyik javában dolgozik mozgási parancs végrehajtásán, a másik kaphat neki szóló mozgásutasítást a PC-tõl. Tulajdonképpen így lehetséges két motor egyidejû mûködtetése. Azért, hogy a PC ne tudja újabb mozgásutasítással megszakítani a korábbi mozgási parancs végrehajtását, egy eljárást kell alkalmazni. A MOT eljárás az aktuális motort (lásd amotor utasítás) a :b paraméterben megadott értékkel forgatja. Ezután hurkot alkalmaz, és addig nem kerülhet ki belõle, amíg a forgás tart (:s?=1). Ha a forgás végetért, (:s?=0) kilép a hurokból. MOT tartalmaz még egy vizsgálatot is, hogy a forgás elérte-e a célját, azaz a kívánt elmozdulás azonos-e a ténylegessel. Ha nem, hibaüzenetet küld.

MO eljárás

A **MO** eljárás két paramétert vár. Az elsõvel beállítja az amotor értéket, (:c) azaz, hogy melyik motort kívánjuk vezérelni, a második pedig a kívánt elmozdulásértéket tárolja(:b).

amotor :c beállítja az aktuális motort
legyen :x :m? az akt. motor számlálójának értékét az *:x* változóba másolja.
motor ? :b az akt. motort *:b* értékkel forgatni kezdi

hurok ha :s? = -1 robot .stop vege ; amíg :s? = 1
hurokban marad amíg a motor forog (*:s?=1*) közben feltételvizsgálat, ha hibás a motor parancs végrehajtás (*:s=-1)* kiszáll
hanem :m? - :x = :b ird .A .kovetkezo .szamu .motor .nem .mukodik :amotor ;
amikor a forgás lezajlott, vagy valami miatt megszakadt, egy feltételviz sgálat következik Az eljárás elején az *:x*-be tett érték és a mostani *:m?* érték különbségének *:b* értéket kell adnia. Ha nem, h ibaüzenettel száll ki.

példa a MO eljárás használatára:

.
.

mo 2 23	meghívja a mo eljárást 2 23 paraméter értékek kel. Ennek hatására a 2-es motor 23-at forog pozitív irányban.

példa vége

AKADALY eljárás

Az **AKADALY** eljárás az egymotoros kocsihoz készült. A motort az 1-es csatornához, a ka pcsolót a 2-eshez kell csatlakoztatni!
Az AKADALY eljárással kocsink automatikus felderítõ robottá válik, mely az utjába esõ akadályok távolságát méri fel és ezekrõl tájékoztatja a központot, a PC-t. Az eljárás végrehajtásakor a kocsi szakaszos mozgással addig megy elõre, amíg akadályba nem ütközik. A kocsi mozgása a grafikus képernyõn is követhetõ, a megtett távolság leolvasható. Ha akadályhoz ér, a kapcsoló megnyomódik, a kocsi leáll. A képernyõn kirajzolódik az akadály helye és kiíródik a távolsága. Ezután a kocsi folyamatos mozgással visszatér a kiinduló helyzetbe. Az eljárás újabb futtatásakor a régi akadály helye is a képernyõn marad. Ha a kocsi egy bizonyos távolságon belül nem ütközött akadályba, visszamegy a helyére és kis idõ múlva újrakezdi a keresést. Ezt háromszor ismétli meg majd hib aüzenettel kiszáll az eljárásból.

Az AKADALY eljárásban két segédeljárást használunk, AKAD1 és AKAD2 néven. Ezek azért kellenek, mert egy eljárás hossza maximum 250 karakter lehet. Az AKADALY eljárás, ha minden szükséges utasítást beleírnánk, hosszabb lenne 250 karakternél .Ezért néhány mûveletet külön eljárásban irtunk meg és az AKADALY nevû eljárásban a nevükkel hívjuk be ezeket.

AKAD1 segédeljárás.	Beállítja az alaphelyzetet.
legyen :m1 0	Az *:m1* változóba *0*-t ír, lenullázza az *1*-es motorcsatorna számlál óját.
orr	Láthatóvá teszi a teknõs orrát.
helyez 10 100	Elhelyezi képernyõ fele magasságában, a bal széltõl 10 egységre.
fejszog 90	A teknõc orra jobbra néz.
akad2	Meghívja az AKA D2 segédeljárást

AKAD2 segédeljárás	Ez az eljárás egy három egység sugarú kört rajzol oda, ahol épp a teknõs van.
toll	Leteszi a tollat.
kor 3	Megrajzolja a kört.
tollne	Felveszi a tollat.

AKADALY eljárás

legyen :q 0	*:q* változót lenullázza. Ebbe kerül majd a sikertelen akadálykeresések sz áma
akad1	Hívja AKAD1-et
hurok	Hurok, melybõl az akadály megtalálásakor léphet ki.
mo 1 1	Az 1-es motor 1-et lép elõre. A fogaskerék áttételek miatt egy motorlépésre a kocsi 5 millimétert halad elõre. Minden hurok lefutás 1 lépést jelent, 1-gyel növelve :m1 értékét. Lásd "MO" eljárás!
elore 5	A képernyõn a teknõc orra 5 egységet megy elõre. (Irányát az AKAD1 állította be.)
*ird .az .ut :m1 * 5*	Az *:m1* változóból kiolvasott adatot, amely a motor1 elmozdulásértéke, azaz az egyes lépések száma, megszorozza 5-tel, majd "az ut" szöveg után kiíratja a képernyõ aljára.

Feltételvizsgálat:

ha :m1 > 60	Ha *:m1* értéke már nagyobb mint 60, azt jelen ti, hogy a hurok már 61-szer lefutott, a kocsi 61x5=305 milliméterre távolodott el a kiindulás helyétõl, anélkül, hogy akadályba ütközött volna.

ird .nincs .akadaly Kiírja a "nincs akadaly" szöveget, majd
mo 1 -:m1 *:m1* értékének -1 szeresével hátratolat a kiindulási helyre.
legyen :q :q + 1 *:q* változót egyel növeli, ez jelzi a sikertelen akadálykeresések számát.

akad1 Meghívja AKAD1-et, beáll az alaphelyzet.
varj 5 ; Beállított ideig így várakozik. Ha az idõ letelt továbblép az amíg utasításra, mely visszaküldi és újabb 60 lépéses akadálykeresés következhet.

Az amíg elõtt még egy feltételvizsgálat.
ha :q = 3 vege ; Ha 3 sikertelen akadálykeresés volt, kiszáll az eljárásból. Hibaüzenettel.

amíg :i2 = 0 A hurok végén lévõ amíg utasítás azt vizsgálja, hogy a kettes csatorna bemenete, input vonala milyen állapotú? Ha a kapcsoló nincs megnyomva, a 2-es input 0 állapotú, a program a hurokban folytatódik. Ha azonban a kapcsoló nyomva van, mert az akadály aminek a kocsi ütközött, megnyomta,
akkor :i2 = 1, erre a program kilép a hurokból.

akad2 A teknõc orra eddig együtt mozgott a kocsival és most kijelöli az AKAD2 számára azt a helyet, ahová a 3-as sugarú kört kell rajzolni.

do Megszólal a do hang. Lásd "DO" eljárást!

Kiíratás következik.
*ird .A .tavolsag .mm-ben :m1 * 5*
Az *:m1* változóban van a motor által elvégzett lépések száma. Ezt 5-tel szorozva kapjuk a megtett utat milliméterben. Ezt írja ki ez az utasítás.

mo 1 -:m1 A kocsi folyamatos mozgással a helyére megy.

Feltûnhet, hogy a kocsi elõremenet szakaszosan, 1-es lépésekkel haladt, visszafelé pedig folyamatosan megy. Ennek oka, hogy a ROBOT CONTROLLER 4 egyszerre csak egy csatornával foglalkozhat. Vagy a motort vezérli amelyik az 1-es csatornához van csatlakoztatva, vagy a kapcsolót kérdezi le, amelyik a 2-es csatornához csatlakozik. Ezért a program úgy mûködik, hogy elõre küldi a kocsit (motor1) majd megáll és lekérdezi a 2-es csatorna inputját (:i2). Ezután kezdõdik újra: mozgás, kérdezés, mozgás, kérdezés stb. Végrehajtáskor ezért villog felváltva a két csatornajelzõ zöld LED. Visszafelé már nem kell a 2-es csatornával törõdni, ezért lehet folyamatos mozgással hátraküldeni a kocsit. Felmerülhet az, hogy a kocsit permanens azaz állandóan bekapcsolt csatorna utasítással küldjük elõre. A perm utasítással bekapcsolt csatornák mellett más csatornákkal is képes foglalkozni a vezérlõ. Pl. amíg a kocsi a perm 1 20 utasítás hatására 20 másodpercig folyamatosan halad elõre, ezalatt a vezérlõ figyelheti a 2-es csatornát és a rákötött kapcsoló állapotát. Ezt szemlélteti a PERMAK nevû eljárás. Van azonban egy bökkenõ. A permanens (állandóan) bekapcsolt csatornához kötött motor forgását nem figyeli a vezérlõ ezért a mozgás adatok sem tárolódnak el az :m változókban. Emiatt nem tudjuk, hogy a motor mennyit is forgott. Ezért nem is tudjuk robotunkat visszavezérelni a kiinduláshelyére.

PERMAK eljárás

A **PERMAK** eljárás az egymotoros kocsihoz készült. A motort az 1-es csatornához, a kapcsolót a 2-eshez kell csatlakoztatni!

Szöveg kiíratása
ird .Ha .valaminek .nekimegyek, .megallok.

ism 5 do szun szun ismvege Ötször megszólal a dó hang. Lásd "DÓ" eljárást!

ird .MEGYEK !	Indulási figyelmeztetés.
perm 1 10	A kocsi elindul, és ha nem találkozik akadállyal akkor is megáll 10 máso dperc múlva.
hurok	Hurok utasítás, melybõl a kapcsoló lenyomott állapota (:i2=1) léptetheti ki a programot.
mi	Menet közben csipogunk. Lásd a "MI" eljárást!
amíg :i2 = 0	
motor 1 -1	Ha nekiment az akadálynak (:i2=1) és kiléptünk a hurokból, most a motor 1-et visszalép, hogy lemásszon az akadályról.
hang 1000 10	1000 Hz-es hang szól 1 másodpercig.
ird .Itt .az .akadaly!	Kiírjuk, hogy elértük az akadályt.

IMPVAR eljárás

Az **IMPVAR** eljárás az egymotoros kocsihoz készült. A motort az 1-es csatornához, a ka pcsolót a 2-eshez kell csatlakoztatni!
Az eljárás mûködése: Ahányszor megnyomjuk a kapcsolót mûködtetõ lemezt, kis várakozás után a kocsi annyiszor 5 mm-t halad szakaszosan elõre, majd folyamatosan vissza.

legyen :m2 0	Lenullázza az :m2 változót. A kapcsoló megnyomásainak száma az :m2 be kerül majd. Minden kapcsoló nyomás 1-el növeli majd :m2 értékét.
hurok	Hurok szervezés. A végrehajtás mindaddig a hurokban marad, míg az :idõzítõ nevû változó nagyobb nullánál, azaz le nem jár a beállított idõ.
legyen :idozito 5	Ez az utasítás az :idõzítõ-t 5 másodpercre állítja be. Ha a hurokban nem lenne olyan utasítás, amely várakozna valamire, a hurok sohasem érne véget, hiszen ennek feltétele, hogy az :idõzítõ étéke nulla legyen, márpedig ez az utasítás mindig újra beállítja 5-re.
input 2 1 5	Ez az utasítás vá rakoztatja a végrehajtást. Ebbõl csak úgy lehet kikerülni, ha a kérdezett csatornán (2) beérkezett a beállított számú(1) impulzus, vagy lejárt a beállított várakozási idõ (5sec) Most a 2-es csatornát kérdezzük. 1 megnyomást várunk, 5 másodperc ideig. Ha 5 másodpercen belül megnyomják a kapcsolót, a várakozás megszûnik, a program továbbléphet. Természetesen a várakozás alatt az elõbb beállított :idõzítõ 5 másodperce is telik. Ha 5 másodpercig nem nyomták meg a kapcsolót, úgy lép ki a várakozásból, hogy közben az :idõzítõ 5 másodperce is letelt, így hurok végén lévõ feltétel vizsgálat üres, azaz 0 értékû :idõzítõ-t talál és kilép a hurokból. Ha volt impulzus, azaz az 5 másodperc letelte elõtt kilépett a várakozásból, az :idõzítõ-nek sem volt ideje lenullázódni, így a hurok végi feltételvizsgálat :idõzítõ értékét nagyobbnak fogja találni 0-nál, ezért a hurok újra kezdõdik
ird .impulzus .szam :m2	A vizsgálat elõtt kiírja :m2 értékét, azaz a beérkezett impulzusok számát.
amíg :idozito > 0	A hurok végi feltételvizsgálat (lásd feljebb!)
skala1	Nem jött új impulzus, letelt az :idõzítõ 5 másodperce, megszólal a zene. Lásd "SKALA1" eljárást!
ism :m2	Most az motor annyit lép, ahányszor a kapcsolót megnyomtuk. Egyes lép éseket fog tenni annyiszor, amennyi az :m2 értéke
mo 1 1	Ez az egyes lépés. 1-es motor 1-et lép. Lásd még a "MO" eljárást! Itt csak azért lépegettünk egyesével, hogy meg tudjuk számolni a lépéseket. Mehettünk volna folyamatosan is. mo 1 :m2

ismvege Ha megtörtént az :m2 változóban tárolt számú
skala2 lépés, megszólal a zene. Lásd "SKALA2" eljárást!

mo 1 -:m2 A kocsi folyamatos mozgással visszamegy eredeti helyére. Az elmozdulás nagyságát tartalmazó :m2 változó értékének -1 szeresével mozdul el az 1-es motor, azaz a kocsi hátra felé megy. Lásd még "MO" eljárást!

KEZI eljárás

A **KEZI** eljárással lehetõvé válik a kiválasztott motor tetszõleges irányú léptetése a "cursor jobbra" és "cursor balra nyilak" nyomkodásával. Ha a nyilak valamelyikét lenyomjuk a kiválasztott motor egyet lép, ha a nyilakat a CTRL gombbal együtt nyomjuk, a motor 10-et lép. A fel ill. a le nyilakkal lehetõség van motorváltásra is. A KEZI nevû eljárás csak egy CONTROLLER 4 kiszolgálására képes, de könnyen átírható több vezérlõre is.

A KEZI eljárás két segédeljárást használ a MOTIR-t és az AMOTVALT-ot.

MOTIR eljárás Teljesen úgy mûködik, mint a MOT eljárás, csak
motor ? :b itt az aktuális motor forgása közben
hurok kiírja a motor fordulatainak számát is, mely
ird :m? az :m? változóban található.
amíg :s? = 1
ird :m? A forgás befejezése után ismét kiírja :m?-et.

A **MOTVALT** eljárás

legyen :x :amotor Az aktuális motor sorszámát tartalmazó :amotor változó értékét az :x változóba teszi. A :k változóban az utoljára nyomott billentyû kódja van. A hívó KEZI eljárás tette bele. Ha :k -ban lévõ kód nem 328 ill. 336 az AMOTVALT mindent változatlanul hagy.

ha :k = 328 legyen :x :x + 1 ; Ha fel nyíl volt :x értéke 1-el nõ
ha :k = 336 legyen :x :x - 1 ; Ha le nyíl volt :x értéke 1-el csökken
ha :x < 1 legyen :x 1 mi ; x értéke 1 és 4 közt lehet, egy vezérlõ
ha :x > 4 legyen :x 4 mi ; esetén. Kettõnél 1 és 8 stb. Ha ezeket túllépjük, :x-be beíródik a határérték, majd "MI" hangjelzést kapunk. Csak az ellenkezõ irányú nyíllal válthatunk ilyenkor csatornát!

amotor :x ird .Motor :amotor Az aktuális motor az :x értékét veszi fel és ez ki is íródik.
legyen :q :i? Ez az utasítás lekérdezi a kiválasztott csatorna input vonalát, de kapott értékre valójában nincs szükség. Az egész azért kell, hogy a PC a CONTROLLER-hez forduljon és kapcsolja be az aktuális csatorna zöld LED-jét. Így amikor csatornát váltogatunk, a LED-ek sorban kigyulladnak és megmutatva az aktuális csatornát.

KEZI eljárás

amotor 1 Az aktuális csatorna az 1-es.
legyen :q :i1 Kigyullad a csatornajelzõ LED. Lásd elõbb!
ird .Motor :amotor Kiírja az aktuális motor (csatorna) számát.
hurok Belépünk a hurokba, melybõl az ENTER gombbal léphetünk ki.
legyen :k :bill Billentyû lekérdezés. A kapott kód a :k változóba kerül.
amotvalt Motorváltás? Az AMOTVALT segédeljárás hívása.
Feltételvizsgálatok
ha :k = 333 mot 1 ird :m? ; Ha jobbra nyíl volt, az aktuális motor egyet elõre lép. Kiírja a :m? értékét
ha :k = 331 mot -1 ird :m? ; Ha balra nyíl volt, motor egyet vissza, kiír.
ha :k = 372 motir 10 ; Ha CTRL+jobbra nyíl volt, motor 10-et elõre lép a MOTIR segédeljárás hívásával. Motor forgás alatt folyamatos :m? kiíratás van.

ha :k = 371 motir -10 ; Ha CTRL+balra nyíl volt, motor 10-et vissza lép MOTIR-ral. Folyamatos :m? kiíratás van.
amig :k > 13 ENTER-rel kiléphetünk.

RAJZOLO eljárás

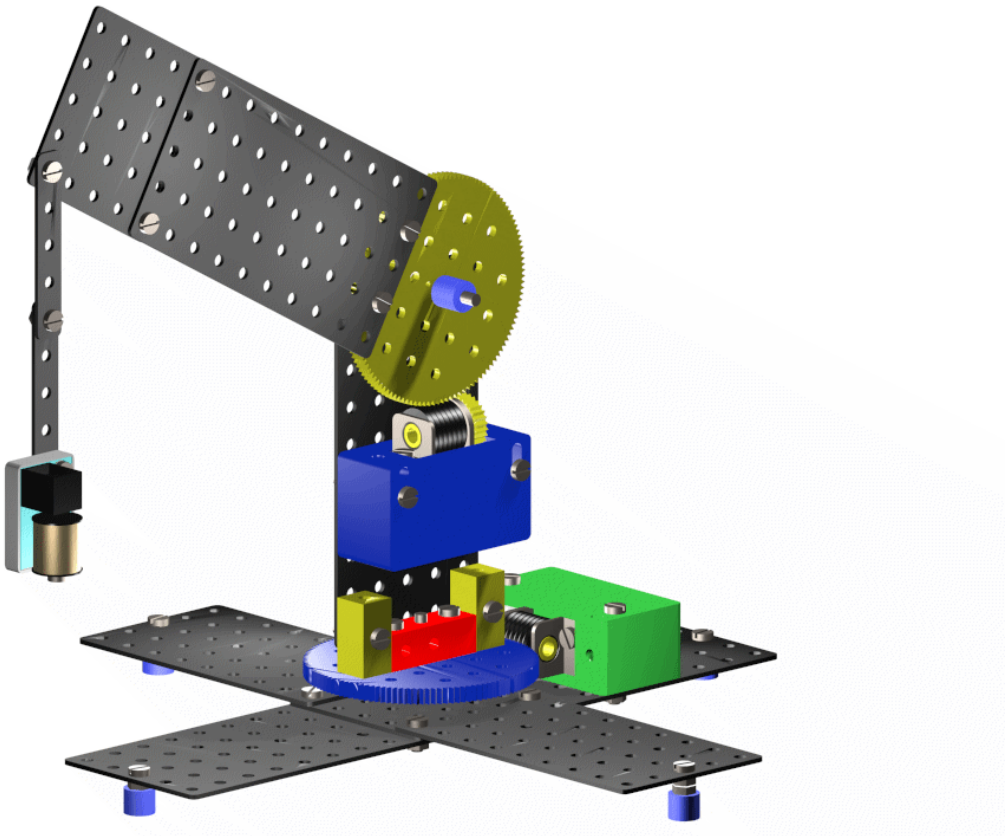
A **RAJZOLO** eljárással különféle alakzatokat rajzolhatunk a grafikus képernyõre. Egyenes és n * 45 fokos vonalakat húzhatunk. Az eljárás bemutatja :bill változó használatát és a "ha" feltételes végrehajtást és a szöveg kiíratást.

init Alaphelyzet beállítás. Lásd "INIT" eljárást! A következõ utasítás kiírja, hogy melyik gombokkal lehet rajzolni. írd .1-es,2-es .gomb .forgat .a .0-as .vonalat .húz

hurok Hurok utasítás. A billentyû figyelését az E NTER (kódja 32) lenyomásáig folyatja.

legyen :b :bill Billentyû lenyomásra vár. A lenyomott billentyû kódja a :bill változóba kerül. A :b felveszi :bill értékét, azaz billentyû kód :b-be is bekerül.
ha :b = 48 elore 5 ; Ha 0-ás billentyût nyomják (kódja 48) a teknõc elõre megy 5-öt.
ha :b = 49 balra 45 ; Ha az 1-est nyomják (kód 49) a teknõc 45 fokot fordul balra.
ha :b = 50 jobbra 45 ; Ha a 2-est nyomják (kód 50) a teknõc 45 fokot fordul jobbra. Finomíthatjuk is rajzolóprogramunkat, ha a feltételvizsgálatokhoz kisebb elmozdulás és szögelfordulás értékeket rendelünk!
amig :b > 32 Ha ENTER-t nyomnak kilép a hurokból.

KETMOTROB eljárás



Az eljárás a kétmotoros csigakerekes robothoz készült. A vezérlõ csatornakiosztása: 1.csat: platform motor 2.csat: karemelõ motor.

Az eljárás egyszerû motorvezérlõ utasításokból áll. Lényeges, hogy a indításkor a robot karja vízszintesen álljon. Normál programlefutáskor a robot a kiindulási helyére tér vissza. Kilépés bármikor lehetséges ESC billentyûvel. Ezután el kell indítani a KEZI eljárást, aminek segítségével a robotot vissza lehet vezetni a kiinduló helyzetbe(A kurzormozgató nyilakkal). Itt kilépünk a KEZI eljárásból (ENTER) Programfutás során a robot felhorgássza a terhet, majd leteszi.

MÁGNESKÉZ eljárás

Az eljárás a kétmotoros csigakerekes robothoz készült. A vezérlõ csatornakiosztása: 1.csat: platform motor 2.csat: karemelõ motor 3.csat: elektromágnes. Az eljárás egyszerû motorvezérlõ utasításokból áll. A mágnest a perm utasítások mûködtetik, hiszen ennek bekapcsolva kell maradni amíg a robot a terhet emeli. A program lefutása során a robot vas csavarokat rakodik egyik dobozból a másikba.

Az EKTF Informatika Tanszéke által indított kísérleti programról:

A most induló tanévtõl kezdõdõen, elõzetes megbeszélések és megállapodások alapján az egri 4.sz. Gyakorló Általános Iskola és Gimnáziumban fogjuk a tanítási óra konkrét folyamatában is kipróbálni a terméket. A gyakorló iskola pedagógusaival szoros és folyamatos konzultációt tartva, közösen dolgozunk ki alkalmazási lehetõségeket az általános és a középiskolában konkrét példákon keresztül, elsõsorban a technika és a számítástechnika tantá rgyakban.

Ezzel a munkával párhuzamosan a fõiskolai hallgatók számára jelenleg a II. évf. technika szakosak kaptak robottechnikai ismereteket az információtechnika tantárgy keretén belül. Elmondhatom, hogy igen kedvezõen fogadták ezt az újdonságnak számító tanegységrészt és ezen felbuzdulva, szeretnénk a második félévtõl kezdõdõen egy robottechnikai specializációt indítani. (A gyártó ígérete szerint már egy nagyobb kiépítettségû, több elemet tartalmazó rendszer segítség évél.)

Az már azonban most is azonnal látszik, hogy a rendszerrel történõ eredményes munka csak úgy valósítható meg, ha a képzõ intézmény minimum 10-15 darabos, azaz egy fél tanulócsoportra elegendõ készlettel rendelkezik, ugyanis a termék jellege, véleményem szerint feltétlenül megköveteli az önálló problémamegoldást, illetve annak fejlesztése csak így lehetséges.

Tanszékünk a gyártó és forgalmazó kft.-vel megállapodva, pedagógiai szempontból és természetesen ehhez a munkához kötõdõ szakmai feladatokat megoldandó, a bázisintézményi feladatokat elvállalta. Programunk igazán eredményes persze csak akkor lehetne, ha minél több fõiskola, egyetem megpróbálná képzési tervébe iktatni a rendszert, továbbá minél több általános és középiskola is bekapcsolódna ebbe a munkába. Várjuk tehát az érdeklõdõk jelentkezését, akiknek szívesen állunk rendelkezésére képzési anyagok átadásával, konzultációs lehetõsé ggel.

Irodalomjegyzék

[1] Horváth Péter - Nagy Lajos : Intelligens gépek, robotok 1. 2. 3. 4. 5.
LSI Budapest, 1989
[2] dr. Siegler András : Robot irányítási modellek
LSI Budapest, 1987
[3] dr. Arz Gusztáv, dr. Lipóth András, dr. Merksz István : Robotmanipulátorok
LSI Budapest, 1988
[4] Isaac Asimov - Karen A. Frenkel : Robotok az emberformájú gépek
Akadémiai Kiadó Budapest, 1992
[5] Várhelyi István : Robot Evolution építési és szerelési útmutató
Robot Evolution Kft. Eger, 1995
[6] Glózik Zoltán - Kurczina István : LOGO 3.02 interpreter
APC Studió Gyula, 1996