



MISKOLCI EGYETEM
GÉPÉSZMÉRNÖKI ÉS INFORMATIKAI KAR



TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

Vezérlő illesztése egy 4 szabadságfokú modell- robothoz

Készítette: Rónai László

Miskolci Egyetem GÉIK mechatronikai mérnöki mester szakos hallgatója

Tankör: Gx1MRB

Neptun kód: TNDKJP

Konzulens: Dr. Szabó Tamás

egyetemi docens

Kézirat lezárva: 2014. október 30.

Tartalomjegyzék

1. Bevezetés	1
2. A modell-robot főbb paraméterei, kinematikai vizsgálata.....	2
2.1. A direkt kinematikai feladat	3
2.2. Az inverz kinematikai feladat.....	7
2.3. A robot mozgás szimulálása Scilab szoftver segítségével.....	10
3. A vezérlő ismertetése	12
3.1. A motor „shield” specifikációi	15
4. Elektronikai bekötések.....	18
4.1. Útmérők illesztése	18
4.2. A megfelelő tápellátás és megfogó vezérléséért felelős NYÁK tervezése.....	20
4.3. A megfogó „kvázi heptikussá” tétele	23
5. Az elkészült vezérlőprogram	26
6. Összefoglalás	28
Köszönetnyilvánítás	28
Irodalomjegyzék.....	29
Függelék	30

1. Bevezetés

A robot egy olyan elektromechanikai szerkezet, amely képes különböző feladatok feldolgozására, végrehajtására (pl. [6]). Alkalmazási területe igen sokrétű, az iparban szinte mindenhol használják, ahol szükséges a pontos és monoton munka. Ebben a Tudományos Diákköri Kutatásban egy olyan 4 szabadságfokú modell-robot fejlesztésének a kérdéseivel foglalkozunk, amelyet elsősorban oktatási és kutatási célra kívánunk alkalmazni.

A modell-robot szerkezete és hajtása kereskedelmi forgalomban szerezhető be, a vezérlés kifejlesztéséhez először a robot mechanizmus kinematikai leírása és az inverz kinematikai feladat megoldása szükséges [5]. A robot vezérlését egy AVR-típusú mikrokontrollerrel valósítottuk meg, amely egy Arduino gyártmányú fejlesztőplatformon található.

Maga a mikrokontrollerhez egy úgynevezett motor „shield” csatlakoztatása elengedhetetlen, ezáltal válik majd lehetővé a DC motorok vezérlése. A vezérlő C–nyelven programozható.

A Tudományos Diákköri munkám gerincét alkotó fő célkitűzésem egy olyan vezérlő illesztése a robothoz, amely szabadon programozható, így különböző feladatok modellezhetők.

A dolgozat második fejezete ismerteti a robot főbb paramétereit, valamint a kinematikai leírását [1], [2]. A harmadik fejezet tárgyalja a robot vezérléséhez használt berendezést, annak sajátosságait. A negyedik fejezetben kerülnek bemutatásra az útmérők elektronikai bekötései, valamint a megfogó vezérléséért, a tápellátásért felelős panel megtervezése, elkészítése. Az ötödik fejezet tér ki a megírt program ismertetésére, egy egyszerű intelligens rakodás példájának bemutatására.

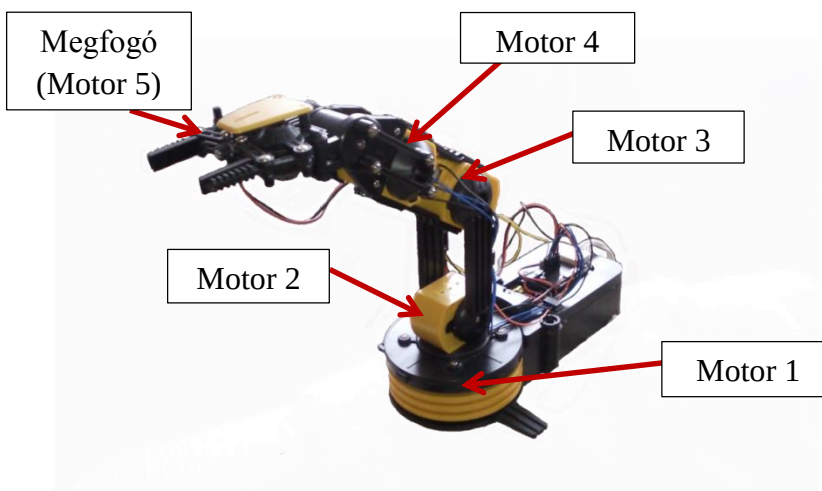
2. A modell-robot főbb paraméterei, kinematikai vizsgálata

A fejlesztésre szánt robotot a Velleman® gyártja [4]. Szereletlen állapotban csomagolják, és a fröccsöntött műanyagokat, valamint a motorokat a felhasználónak kell összeszerelnie. A szerelési folyamat kb.: 2-3 órát vesz igénybe. A robot tartalmaz egy kézi működtetésű távirányítót, amivel az 5 motort irányíthatjuk. A főbb paramétereket az 1. táblázat foglalja össze.

<u>Paraméter:</u>	<u>Értéke:</u>
Maximális kinyúlás	380 mm
Maximálisan emelhető tömeg	100 g
A robot tömege	660 g
Áramforrás	4 db LR20C góliát elem (Vezérlés során DC adapter lesz alkalmazva)
Motorok száma	5 db
Szabadságfok	4 db (RRR)

1. táblázat A főbb paraméterei a robotnak

Mint ahogy az 1. táblázat is érzékelteti, a robot nem képes komolyabb feladatokat ellátni magától ebben az állapotban. A dolgozatban bemutatásra kerül egy lehetséges kibővítése érzékelőkkel, amelyekkel már vezérelhetővé válik. Az 1. ábrán látható maga a robot. A motorok csuklókhoz való illesztése különböző, lassító- fogaskerekes áttételeken keresztül valósulnak meg. Mind az 5 db. motor tengelyén egy csigakerék található.



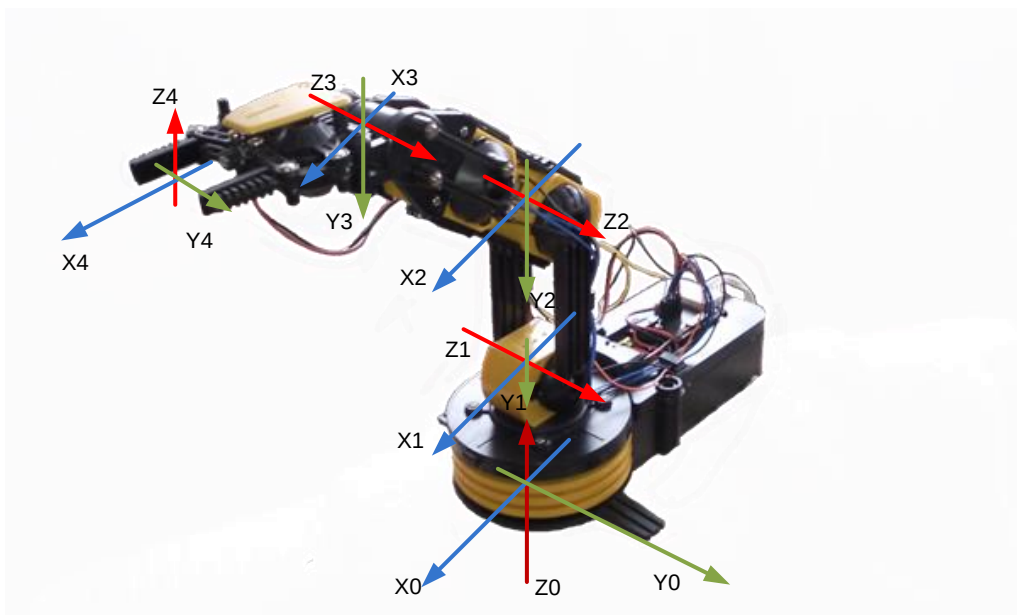
1. ábra A robot felépítése

2.1. A direkt kinematikai feladat

A robot direkt kinematikai vizsgálata a Denavit-Hartenberg-féle (DH) paraméteres leírásmód alkalmazásával történik. Direkt kinematikai leírásmódnál a robot egyes csuklóit változásának segítségével a megfogó orientációjának, pozíciójának a felírása a cél. Ahhoz, hogy a DH által definiált paraméterek meghatározásra kerüljenek, először az egyes csuklókhoz koordinátarendszereket kell rendelni. A koordinátarendszerek felvétele szabályokhoz van kötve [1]:

- Első lépés a „Z” tengely felvétele, amely az egyes csuklók forgástengelyével esik egybe.
- Második lépés az „X” tengely rögzítése, amelyet a Z_i és a Z_{i-1} közös normál-transzverzálisa (mindkét tengelyre merőleges) ad.
- Az utolsó lépésben az „Y” tengelyt kell definiálni. A koordinátarendszernek jobbsodrásúnak kell lennie, ezért az utolsó tengely felvétele egyszerű.

A 2. ábra mutatja az egyes koordinátatengelyek egy lehetséges felvételét. A következő lépés a DH paraméterek megadása, a felvett koordinátarendszerek függvényében.

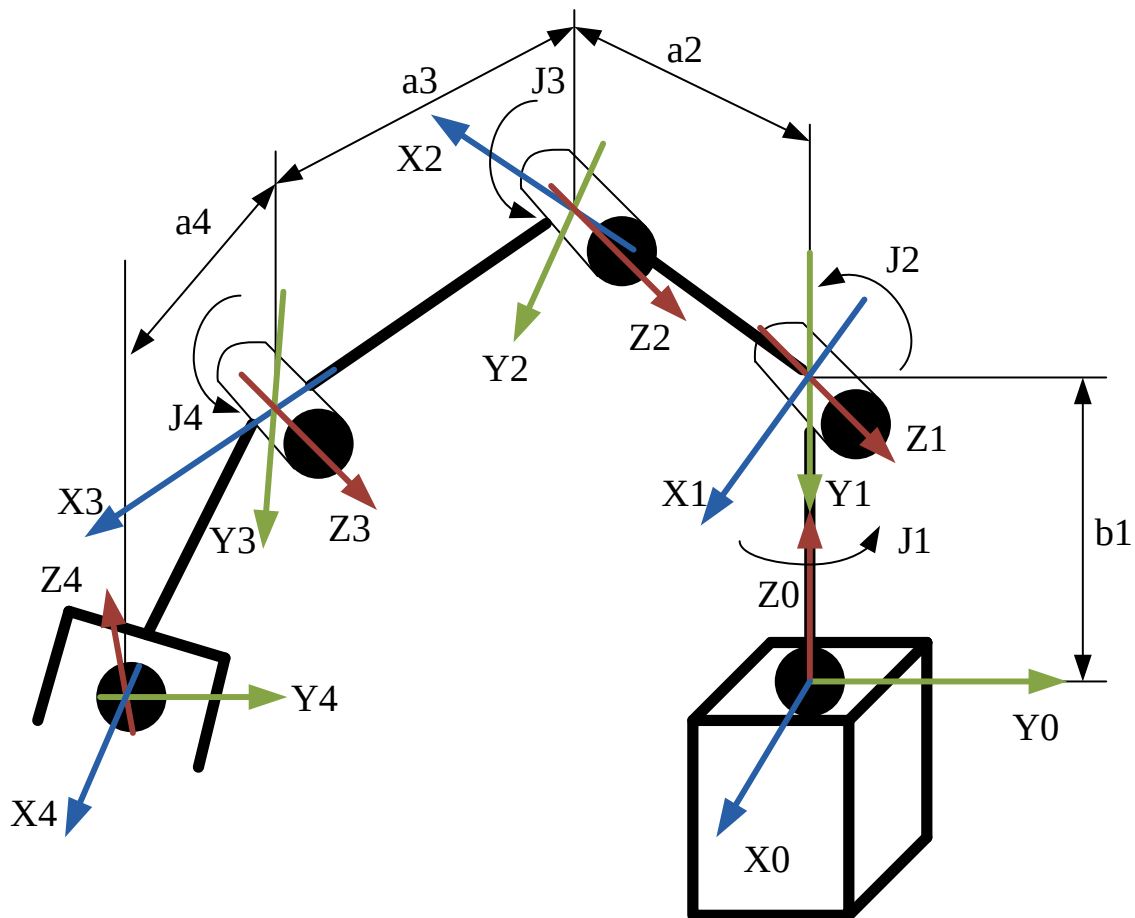


2. ábra A koordinátarendszerek rögzítése

Összesen négy paramétert kell deklarálni, ezek: a_i , b_i , β_i , α_i . A paraméterek megadásának a szabályai [1]:

- ❖ a_i a Z_{i-1} és Z_i tengelyek távolsága (a normál-transzverzális hossza, amely nem lehet negatív),
- ❖ b_i a Z_{i-1} és X_i tengelyek metszéspontjának Z_{i-1} koordinátája,
- ❖ α_i a Z_{i-1} tengely körüli forgás előjeles szöge, amely az X_{i-1} tengelyt az Z_i tengelybe viszi,
- ❖ β_i az X_i tengely körüli forgás előjeles szöge, amely a Z_{i-1} tengelyt a Z_i tengelybe viszi.

A 3. ábra szemlélteti részletesen a robot kinematikai vázlatát. A rajz megmutatja az egyes koordinátarendszereket és a DH paraméterek értelmezését.



3. ábra A robot kinematikai vázlata

A 2. táblázat magába foglalja a csuklónkénti számszerűsített paramétereket.

Tag sorszáma:	a_i	b_i	β_i	α_i
1.	0	$b_1 = 65 \text{ mm}$	-90°	$\alpha_1 = J1$
2.	$a_2 = 90 \text{ mm}$	0	0	$\alpha_2 = J2$
3.	$a_3 = 60 \text{ mm}$	0	0	$\alpha_3 = J3$
4.	$a_4 = 100 \text{ mm}$	0	90°	$\alpha_4 = J4$

2. táblázat A tagok DH paraméterei

A DH paraméterek számszerűsítése után felírhatóvá vált a DH-féle transzformációs mátrix, definiálását az (1) kifejezés mutatja [1]:

$$\mathbf{D}_{i-1,i} = \mathbf{Trans}_Z * \mathbf{Rot}_Z * \mathbf{Trans}_X * \mathbf{Rot}_X, \quad (1)$$

$$\mathbf{Rot}_Z = \begin{bmatrix} \cos(\alpha_i) & -\sin(\alpha_i) & 0 & 0 \\ \sin(\alpha_i) & \cos(\alpha_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{Rot}_X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\beta_i) & -\sin(\beta_i) & 0 \\ 0 & \sin(\beta_i) & \cos(\beta_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (2)$$

$$\mathbf{Trans}_Z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & b_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{Trans}_X = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

ahol a $\mathbf{Trans}_X$, $\mathbf{Trans}_Z$ az eltolási (transzlációs) mátrixokat, azaz a lineáris elmozdulás mátrixokat reprezentálja, míg a $\mathbf{Rot}_X$, $\mathbf{Rot}_Z$ a rotációs, azaz a forgási mátrixot jelenti. Az (1) explicit kifejezésben elvégzett mátrix szorzás után előáll a relatív helyzetmátrix alakja (3) [1]:

$$\mathbf{D}_{i-1,i} = \begin{bmatrix} \cos(\alpha_i) & -\sin(\alpha_i) \cos(\beta_i) & \sin(\alpha_i) \sin(\beta_i) & a_i \cos(\alpha_i) \\ \sin(\alpha_i) & \cos(\alpha_i) \cos(\beta_i) & -\cos(\alpha_i) \sin(\beta_i) & a_i \sin(\alpha_i) \\ 0 & \sin(\beta_i) & \cos(\beta_i) & b_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3)$$

A vizsgálandó robot összesen négy tagból épül fel, ezért négy darab relatív helyzetmátrix felírását kell elvégezni. A (4.1-4.4) kifejezések szemléltetik a négy helyzetmátrix alakját.

$$\mathbf{D}_{0,1} = \begin{bmatrix} \cos(J1) & 0 & -\sin(J1) & 0 \\ \sin(J1) & 0 & \cos(J1) & 0 \\ 0 & -1 & 0 & b_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4.1)$$

$$\mathbf{D}_{1,2} = \begin{bmatrix} \cos(J2) & -\sin(J2) & 0 & a_2 \cos(J2) \\ \sin(J2) & \cos(J2) & 0 & a_2 \sin(J2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4.2)$$

$$\mathbf{D}_{2,3} = \begin{bmatrix} \cos(J3) & -\sin(J3) & 0 & a_3 \cos(J3) \\ \sin(J3) & \cos(J3) & 0 & a_3 \sin(J3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4.3)$$

$$\mathbf{D}_{3,4} = \begin{bmatrix} \cos(J4) & 0 & \sin(J4) & a_4 \cos(J4) \\ \sin(J4) & 0 & -\cos(J4) & a_4 \sin(J4) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4.4)$$

A mátrixok felírása után össze kell szorozni őket, hogy megkapható legyen a megfogó és az alappont közötti kapcsolat (5), azaz a helyzetmátrix:

$$\mathbf{DH}_{0,4} = \prod_{i=1}^4 \mathbf{D}_{i-1,i}. \quad (5)$$

A (6), (7) kifejezések szemléltetik a mátrix szorzás során kapott közbenső eredményeket.

$$\mathbf{D}_{0,2} = \begin{bmatrix} c(J1)c(J2) & -c(J1)s(J2) & -s(J1) & a_2 c(J2)c(J1) \\ s(J1)c(J2) & -s(J1)s(J2) & c(J1) & a_2 c(J2)s(J1) \\ -s(J2) & -c(J2) & 0 & -a_2 s(J2) + b_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (6)$$

$$\mathbf{D}_{0,3} = \begin{bmatrix} c(J1)c(J2+J3) & -c(J1)s(J2+J3) & -s(J1) \\ s(J1)c(J2+J3) & -s(J1)s(J2+J3) & c(J1) \\ -s(J2+J3) & -c(J2+J3) & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad (7)$$

$$\begin{bmatrix} a_3 c(J1)c(J2+J3) + a_2 c(J2)c(J1) \\ a_3 s(J1)c(J2+J3) + a_2 c(J2)s(J1) \\ -a_3 s(J2+J3) - a_2 s(J2) + b_1 \\ 1 \end{bmatrix}.$$

A (7) kifejezés $\mathbf{D}_{3,4}$ -el való megszorozása után áll elő a helyzetmátrix (8), amely a megfogó és az állvány közötti kapcsolatot reprezentálja. A mátrix egyszerűsített alakját lehet látni, mivel az addíciós-tételek segítségével egyszerűsítések kerültek alkalmazásra [5].

$$\mathbf{D}_{0,4} = \begin{bmatrix} c(J1)c(J2 + J3 + J4) & -s(J1) & c(J1)s(J2 + J3 + J4) \\ s(J1)c(J2 + J3 + J4) & c(J1) & s(J1)s(J2 + J3 + J4) \\ s(J4 - (J2 + J3)) & 0 & -c(J4 - (J2 + J3)) \\ 0 & 0 & 0 \end{bmatrix}, \quad (8)$$

$$\begin{bmatrix} a_4c(J1)c(J2 + J3 + J4) + a_3c(J1)c(J2 + J3) + a_2c(J1)c(J2) \\ a_4s(J1)c(J2 + J3 + J4) + a_3s(J1)c(J2 + J3) + a_2s(J1)c(J2) \\ -a_4s(J4 - (J2 + J3)) - a_3s(J2 + J3) - a_2s(J2) + b_1 \\ 1 \end{bmatrix}.$$

A helyzetmátrix általános alakja:

$$\mathbf{D} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}, \quad (9)$$

ahol $\mathbf{R}$ jelenti a forgást, $\mathbf{t}$ pedig az eltolást.

2.2. Az inverz kinematikai feladat

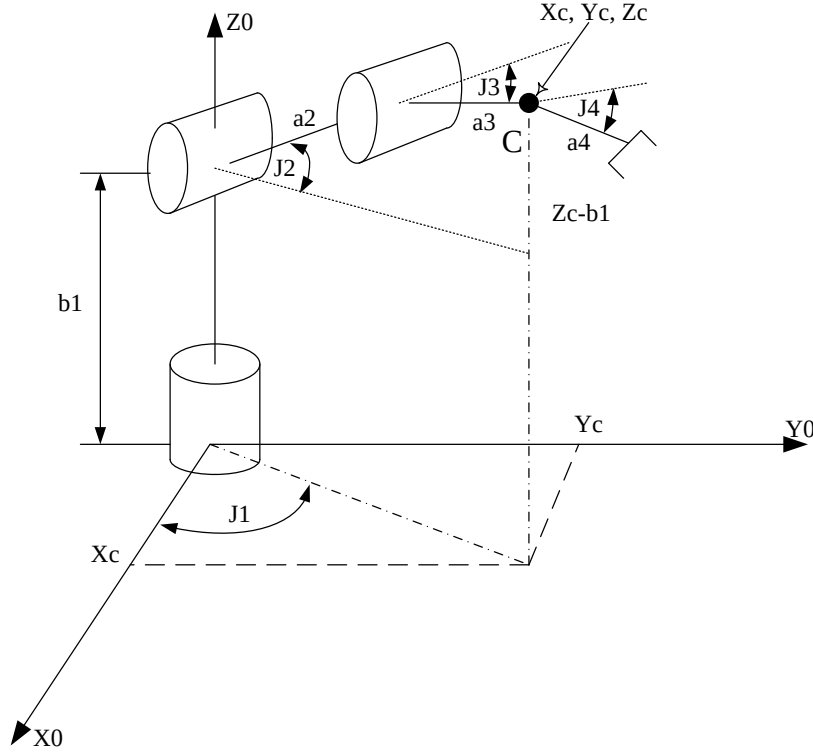
Az inverz kinematikai feladat megoldása az egyes csuklókhöz tartozó elfordulási szögeket állítja elő a megfogó pozíciója és orientációja függvényében. A feladat általánosan az alábbi alakban fogalmazható meg [1]:

$$\mathbf{q} = f^{-1}(\mathbf{s}), \quad (10)$$

ahol $\mathbf{q}$ a csuklókoordinátákat, míg $\mathbf{s}$ a világkoordinátákat jelenti. Inverz kinematikai feladatot általánosan kétféle képen lehet megoldani:

- Algebrai-, illetve numerikus-
- Geometriai úton

Egyszerűbb robot konstrukciók esetén célszerű választani a geometriai megoldásmódot, komplexebb eseteknél az algebrai alternatívát javallott választani. A 4. ábra szemlélteti a robot geometriai vázlatát.



4. ábra A robot vázlata inverz kinematikai feladat felírásához

Első lépésben a (7) relatív helyzetmátrix oszlopának első három tagja rendre a 4. ábra C pontjának X_c, Y_c, Z_c koordinátájával egyenlő:

$$\begin{aligned} X_c &= \cos J_1 (a_2 \cos J_2 + a_3 \cos(J_2 + J_3)), \\ Y_c &= \sin J_1 (a_2 \cos J_2 + a_3 \cos(J_2 + J_3)), \\ Z_c - b_1 &= -a_2 \sin J_2 - a_3 \sin(J_2 + J_3). \end{aligned} \quad (11)$$

Először a harmadik csuklóhoz tartozó elfordulási szöget lehet meghatározni, úgy hogy venni kell a három egyenlet négyzetösszegét, amely az egyszerűsítések után az alábbi alakot ölti:

$$X_c^2 + Y_c^2 + (Z_c - b_1)^2 = a_2^2 + a_3^2 + 2a_2a_3\cos J_3. \quad (12)$$

Ebből kifejezve a $\cos J_3$ -at előáll a harmadik csuklóhoz tartozó elfordulási szög koszinusza (13), és a trigonometrikus azonosságok felhasználásával megkapható a $\sin J_3$ is:

$$\cos J_3 = \frac{X_C^2 + Y_C^2 + (Z_C - b_1)^2 - a_2^2 - a_3^2}{2a_2a_3}, \sin J_3 = \pm \sqrt{1 - \cos^2 J_3}. \quad (13)$$

A J_3 csuklósög meghatározásához, fel kell használni az arkusz-tangens 2 függvényt. A függvény használatakor fontos, hogy az első helyi értékhez kell írni a J_3 szinuszt, a másodikhoz a koszinuszt (14).

$$J_3 = \text{atan2}(\sin(J_3), \cos(J_3)). \quad (14)$$

Az $\text{atan2}()$ függvény az arkusz-tangens hibáit küszöböli ki, mivel $\text{atan}(x)$ nincs értelmezve $+90^\circ$ és -90° -nál, valamint nem ad helyes információt a szög megfelelő síknegyedéről. A következőkben a többi csuklóhoz is definiálásra kerülnek az egyes szögek.

$$J_1 = \text{atan2}(Y_C, X_C). \quad (15)$$

Az X_C és Z_C egyenletek trigonometrikus azonosságokkal való átalakítása után megkapjuk az összefüggést a J_2 szög koszinuszára és szinuszára:

$$\cos J_2 = \frac{\frac{X_C}{\cos J_1}(-a_2 - a_3 \cos J_3) + (Z_C - b_1)a_3 \sin J_3}{(-a_2^2 - a_3^2 - 2a_2a_3 \cos(J_3))}, \sin J_2 = \pm \sqrt{1 - \cos^2 J_2}, \quad (16)$$

így a J_2 csuklósög felírható:

$$J_2 = \text{atan2}(\sin(J_2), \cos(J_2)). \quad (17)$$

A robot mozgása közben pl. a megfogó vízszintes állapotban tartásának feltételezésével a negyedik csukló szöge megkapható:

$$J_4 = -(J_3 + J_2). \quad (18)$$

A (11)-(18) összefüggések alapján egy szimulációs program készült Scilab szoftver alatt, amely előállítja a robot csuklónkénti elfordulási szögeit, szögsebességeit.

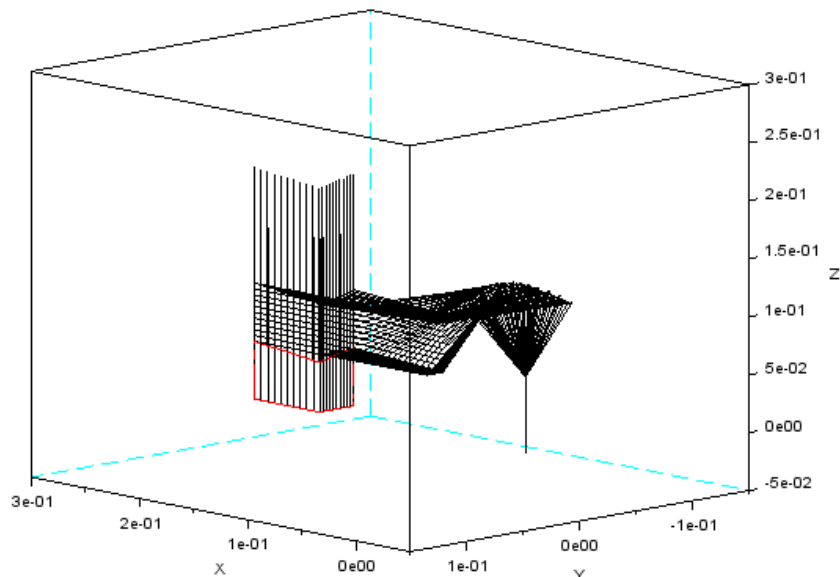
A robot inverz-kinematikai feladatának megoldása után következhet a vezérlő egység illesztése és a robot útmérővel való kibővítése.

2.3. A robot mozgás szimulálása Scilab szoftver segítségével

A 2.2. fejezet alapján elkészült egy Scilab program, amely képes a robot mozgásának szimulálására, valamint az egyes csuklók szögértékeinek és szögsebességeinek a megadására.

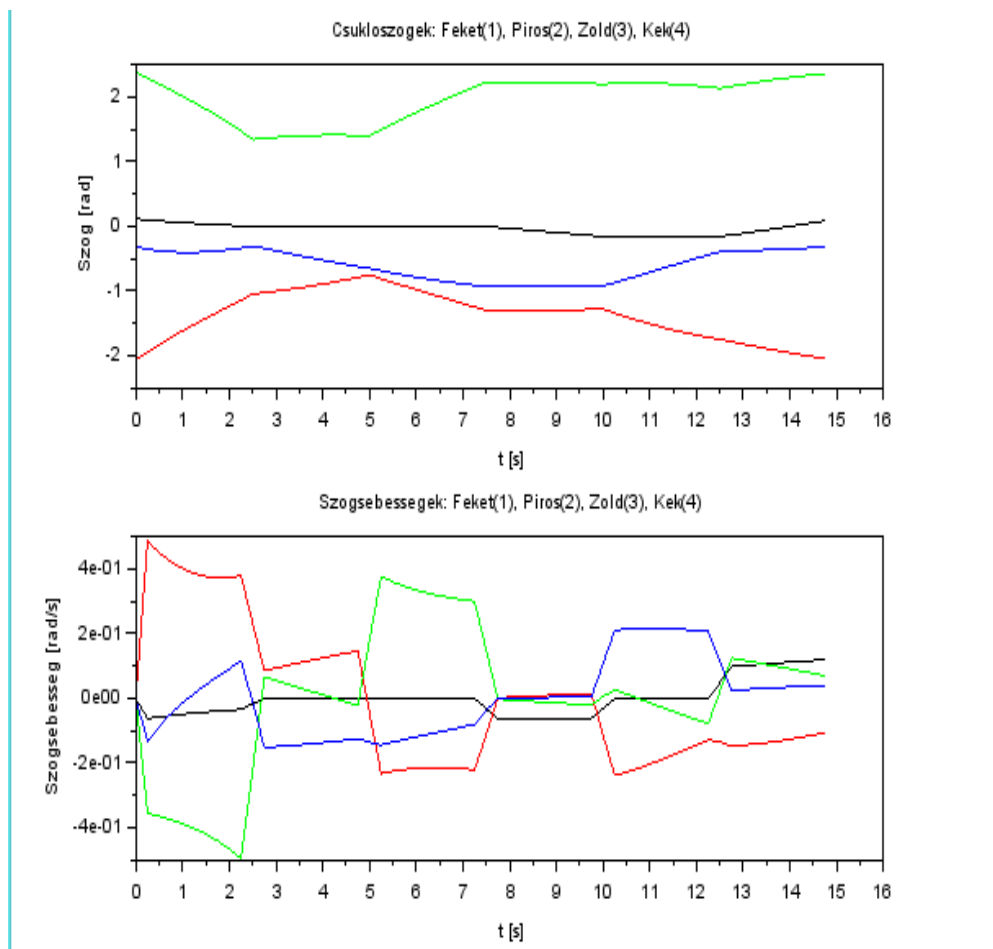
A Scilab, egy ingyenesen letölthető, Matlab-szerű szoftver.

A szimuláció során a robot egy „L” alakú pályát jár be. A pálya tehát 2 vonalból áll, amelyek egyenkénti osztásköze 10, természetesen a programban ez szabadon módosítható. A program először meghatározza DH-féle mátrixot a megfogó és a talp között, majd az inverz kinematika felhasználásával az egyes csuklókhoz a szögek megkaphatóvá váltak. Az 5. ábra mutatja a robot pályáját.



5. ábra A robot kitalált pályája

Az ábrán jól megfigyelhetők a robot mozgásterének méretei, valamint a kirajzolandó pálya is. A 6. ábrán látható görbék a szögek és szögsebességek változását mutatják az idő függvényében. Az ábrán látható egyes görbék színei a megfelelő csuklókra utalnak.

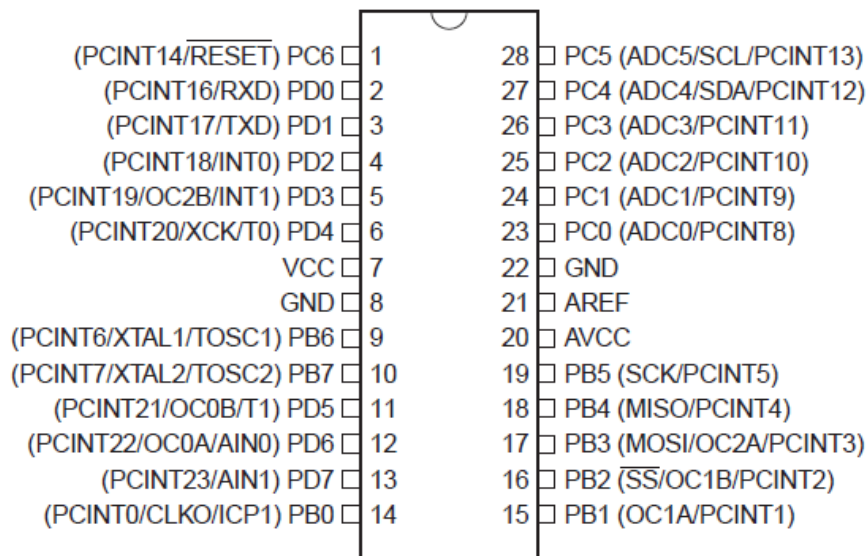


6. ábra A szimulációs program, annak értékei

A 6. ábrán szemléltetett görbék a robot vezérlésének alapjeleiül szolgálhatnak.

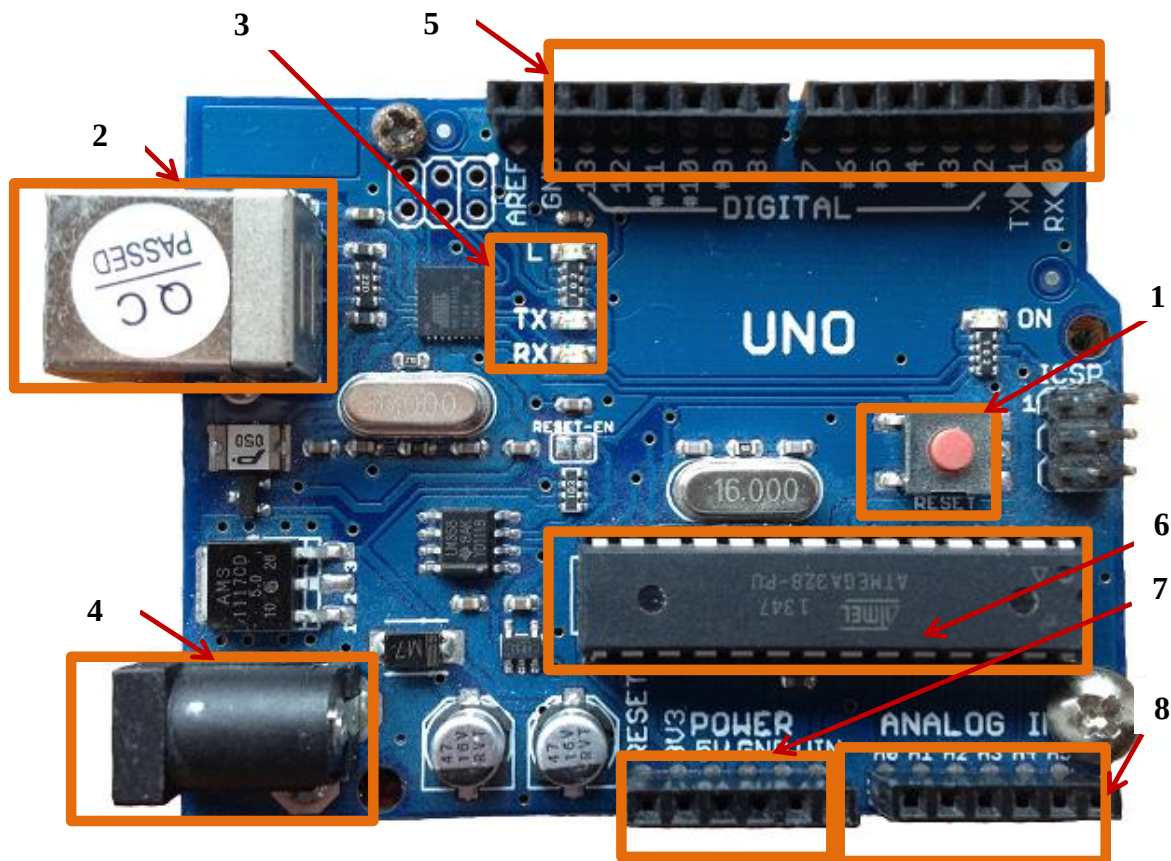
3. A vezérlő ismertetése

A TDK dolgozat elkészítése során a vezérlést egy Arduino-UNO platformon található AVR típusú mikrovezérlő oldja meg. A mikrokontrolleren található digitális ki- és bemenet, valamint analóg bemenetek is. Az alkalmazott mikrovezérlő egy ATmega328 8 bites CMOS alacsony fogyasztású mikrochip [3]. A chip tűkiosztását szemlélteti a 7. ábra.



7. ábra A mikrovezérlő tűkiosztása [3]

A dolgozatban felhasznált mikrovezérlő modul PDIP (Plastic dual in-line package) kialakítású. A 8. ábra mutatja be a mikrovezérlőt tartalmazó alaplapon felépítését. Az alaplapon található áramkörök működéséhez elegendő az USB port által kínált feszültség is, amely 5 V. Az alaplapon tartalmaz még egy jack aljzatot is, amellyel lehetőség nyílik külső áramforrással meg táplálni az áramkört. Maga az áramkör dönti el, hogy az opcionális (USB, vagy jack aljzat) áramforrások közül melyiket válassza. Fontos megjegyezni, hogy mivel a robotot felépítő motorok működtetéséhez nagyobb áramokra van szükség, mint amelyet az USB port szolgáltat, ezért egy a későbbiekben részletezett adapter került felhasználásra.



8. ábra Az alaplap és a rajta található mikrovezérlő

Az alaplapon található főbb elemek megnevezése:

- 1) RESET gomb
- 2) USB csatlakozó
- 3) Állapotjelző LED-ek
- 4) Jack csatlakozó
- 5) Digitális ki- és bemenetek
- 6) A mikrovezérlő chip
- 7) Feszültségforrás az igénylő áramköri elemeknek
- 8) Analóg bemenetek

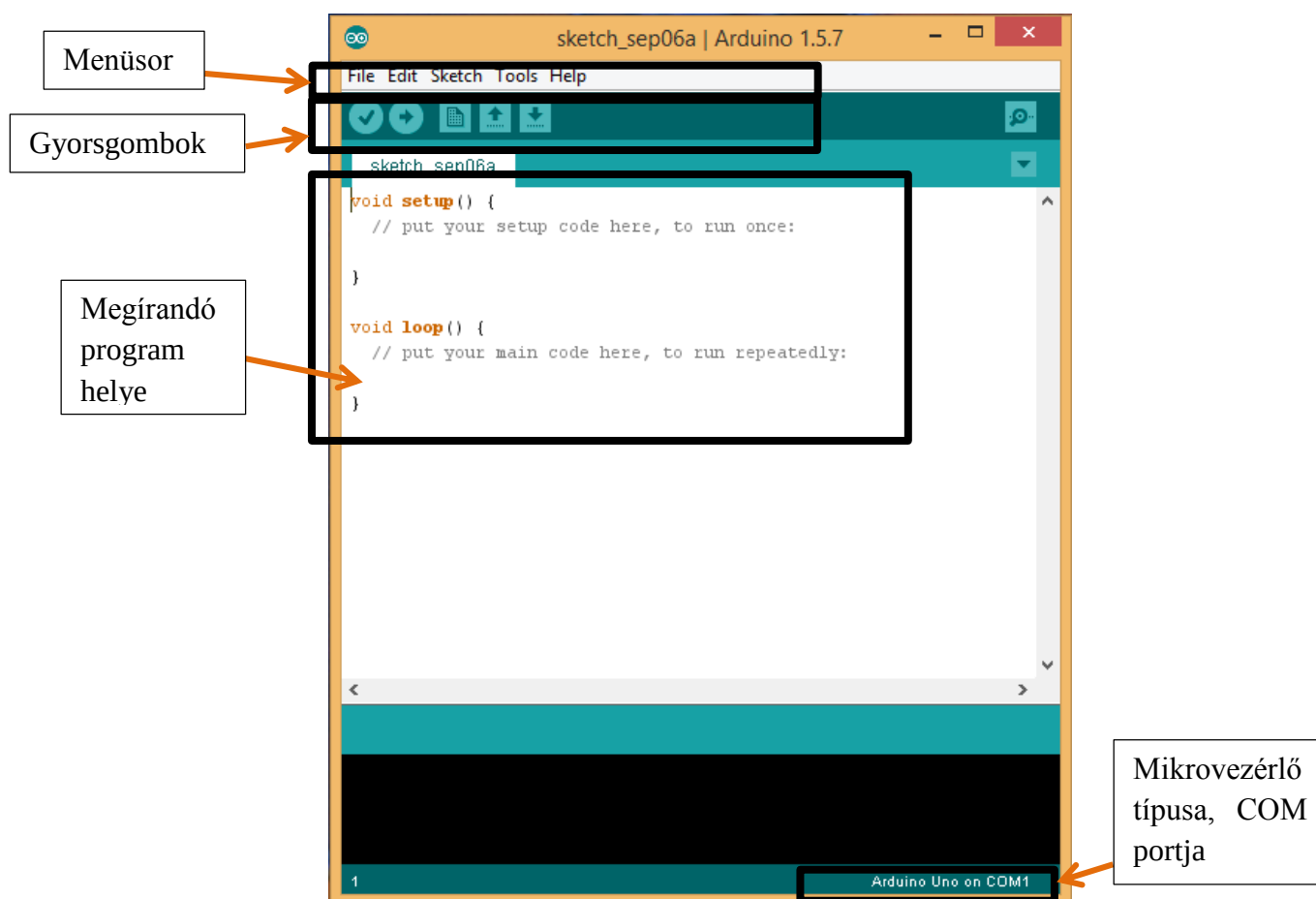
Maga az „UNO” egység főbb specifikációit a 3. táblázat ismerteti [3].

Összességében elmondható, hogy a mikrovezérlőt különböző automatizálási feladatokra lehet alkalmazni, mint például futófény, LCD kijelző programozása. Természetesen a vezérlőhöz különböző szenzorokat is lehet csatlakoztatni (potenciométer, szénmonoxid-, páratartalom-,

RGB érzékelő stb.). A robotra négy darab potenciométer került felszerelésre. A potenciométerek bekötését a 4. fejezet ismerteti részletesen.

Paraméter:	Értéke:
Mikrovezérlő	ATmel-mega328
Mikrovezérlő üzemi feszültsége	5 V
Javasolt bemeneti feszültség	7-12 V
Digitális I/O egységek száma	20 db
Analóg bemenetek száma	6 db
Órajel	16 MHz
Memória: Flash, SRAM, EEPROM	32 KB, 2 KB, 1KB

3. táblázat A mikrovezérlő panel főbb specifikációi



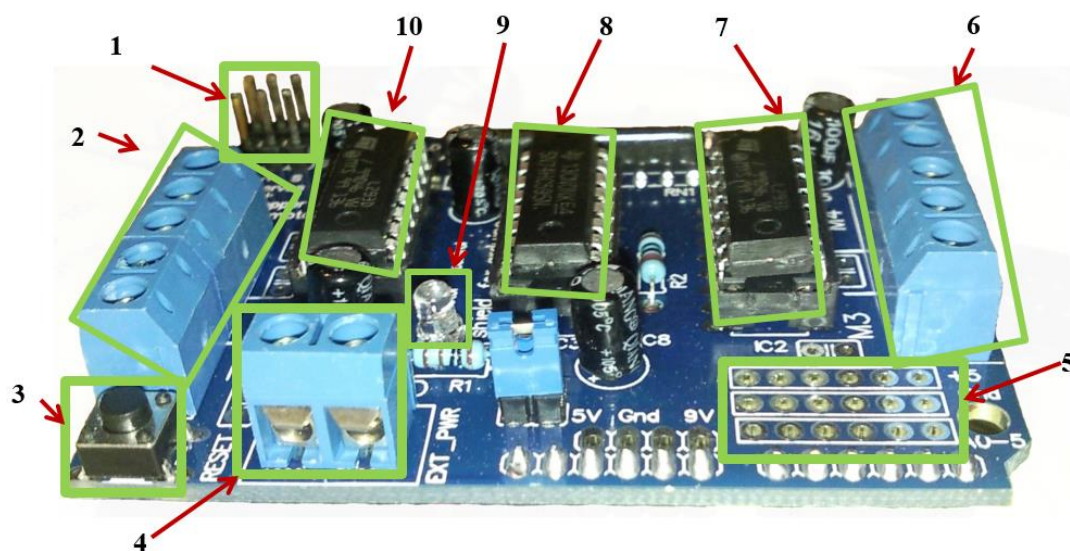
9. ábra A program kezelőfelülete

A vezérlő programozása az Arduino hivatalos oldalán található szoftver letöltése, majd a telepítése után kezdődhet el C-nyelven. A 9. ábrán látható a szoftver kezelőfelülete. A

programozáshoz az 1.5.6 verziószámú szoftver került felhasználásra. Fontos lépés minden kódolási folyamat előtt, a mikrovezérlő típusának és a kommunikációs portjának a beállítása. A szoftveren belül a *Tools* fül/ *Board* lehetőségnél állítható be a vezérlő típusa, jelen esetben az UNO. A COM port beállítása is a *Tools* fülön belül történhet a *Port* opció használatával. A *File/Settings* fülnél beállítható a szerkesztő betűméret, a rendszer nyelve stb., felhasználó specifikus beállítások végezhetők el.

3.1. A motor „shield” specifikációi

A motorvezérlő alaplapot a SaintSmart gyártotta. Segítségével lehetőség nyílt mind szervo, mind pedig egyenáramú motorok (DC) vezérlése. Összesen négy darab DC motort, valamint két darab szervo motort képes vezérelni. A „shield”-ek funkciója abban rejlik, hogy a már meglévő mikrovezérlő szerepkörét kibővítsék. Léteznek a motor „shield”-en kívül más egyéb bővítő modulok is, a teljesség igénye nélkül: GSM, GPS, Ethernet, TFT, RFID „shield”-ek stb. A 10. ábra szemlélteti a motor vezérléséhez szükséges bővítő modult. A bővítő elem a következő pin-eket használja fel az alap mikrovezérlő modulból: A0-A5 (összes analóg bemenet), 5V, GND, 9V, RESET, 3,3 V, Vin, AREF, D0-D13 (összes digitális ki- és bemenet). Az AREF pin segítségével az analóg 5 V-os feszültségszintet lehet módosítani egy felhasználó által definiált referencia feszültséggel. A Vin pin használatával lehetőség adódik külső tápforrással meg táplálni a rendszert. A GND a földelést akarja jelteni.



10. ábra A motor „shield” felépítése

A 10. ábrán található tételszámok jelentése a következő:

- 1) Szervo motor pin sor
- 2) 2 db DC motor sorkapocs
- 3) RESET mikrokapcsoló
- 4) Külső tápforrás sorkapocs
- 5) Analóg bemenetek (A0-A5)
- 6) 2 db DC motor sorkapocs
- 7) L293D Négyszeres fél H hidas IC, amely lehetővé teszi a kétirányú áram-utat. Egy darab IC-re összesen két darab DC motor csatlakoztatható. Maximum 600 mA árammal terhelhető, valamint működési feszültségtartománya: 4,5-36 V.
- 8) 8 bites shift regiszter IC, 3 állapotú kimenettel, soros be- és párhuzamos kimenettel rendelkezik [SIPO], működési feszültségtartomány: 2-6 V. Típuszáma: SN74HC595N. D-típusú tárolókból épül fel.
- 9) Állapotjelző LED (Működéskor piros)
- 10) L293D Négyszeres „Half-H” IC, lehetővé teszi a kétirányú áram-utat. Egy darab IC-re összesen két darab DC motor csatlakoztatható. Maximum 600 mA árammal terhelhető, valamint működési feszültségtartománya: 4,5-36 V.

A motor „shield”-et a mikrovezérlőre ráapplikálva válik elérhetővé a tényleges vezérlés funkciója. Mivel összesen négy darab DC motor vezérlésére képes, ezért a megfogót működtető ötödik motor irányítása egy különálló L293D IC segítségével valósítható meg. Ahogy már az előzőekben is megemlítésre került, induktív terhelésekhez nagyobb áramokra van szükség, mint amelyet az USB port szolgáltat, ezért egy AC/DC adapter került felhasználásra, melyet a 4) Külső tápforrás sorkapocsba kellett csatlakoztatni. A mikrovezérlő automatikusan észleli és kiválasztja a számára opcionálisabb feszültség bemenetet.

Az adapter egy régi telefonnak biztosította a tápellátást. Az általa kínált áram és feszültség szintek alkalmasak voltak, hogy illeszthető legyen az áramkörhöz. Az adapter főbb jellemzői leírásra kerültek a 4. táblázatban. Mind az adapter jack csatlakozója, mind pedig az ötödik motor vezérléséért felelős IC és annak állapotjelző LED-jei egy közös próba-NYÁK-ra lettek forrasztva. A motorok 6 V feszültséggel működnek, ezért erre a NYÁK-ra egy L7806 típusú feszültség-reguláló IC is beépítésre került, melyet a 13. ábra mutat be.

Paraméter:	Értéke:
Bemenet	AC 230 V, 50 Hz
Kimenet	DC 9 V, 200 mA
Adapter védettsége	IP20, tehát víz elleni védettsége nincsen

4. táblázat Az adapter jellemzői

4. Elektronikai bekötések

Ez a fejezet ismerteti azokat a hardveres átalakításokat, bővítéseket, amelyek elengedhetetlenek a robot automatizálásához. A 4.1. alfejezet részletesen leírja az útmérők csatlakoztatásának menetét. A 4.2. alfejezet tér ki egy olyan panel megtervezéséhez, mely a robot megfogójának vezérléséért, és a rendszer tápfeszültségének szabályozásáért felelős. Az utána következő alfejezet demonstrálja a megfogónak a felszerelését egy mikrokapcsolóval, amely jelzi majd a rendszernek, ha a munkadarabot a robot már megfelelő erővel megfogta.

4.1. Útmérők illesztése

A robotohoz összesen négy darab abszolút forgó útmérő lett illesztve. Az útmérők szerepét egy darab lineáris karakterisztikájú forgó potenciométer és 3 db lineáris karakterisztikájú trimmer fogja betölteni, amelyeket a 11. ábrán lehet megfigyelni.



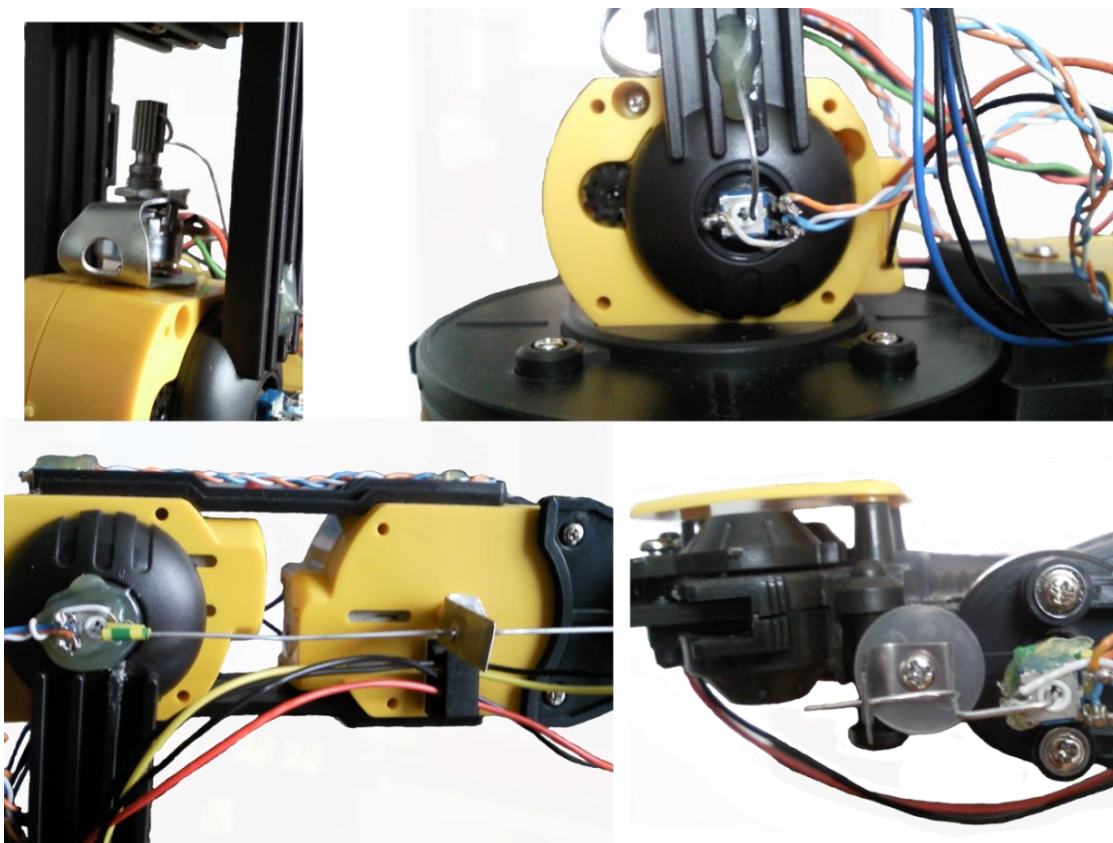
11. ábra Az alkalmazott útmérők

Első lépés a potenciométer elhelyezése az első motor forgástengelyébe. A legalkalmasabb hely a második motoros hajtás tetején levő két darab furathoz való applikálása volt. A potenciométernek három kivezetése van, a középső lesz az Arduino analóg inputjába, a két

szélső kivezetés pedig a GND és az 5 V-ba kerül bekötésre. Itt mindegy, hogy a két szélső érintkező közül melyik kapja az 5 V-ot.

A felszerelése 2 darab csavar és egy hilti-szalag segítségével valósult meg, a 12. ábra-montázs bal felső sarkában látható felszerelt állapotában. Az alapkoncepció szerint a potenciométernek a teste fog forogni, és a forgó része lesz stabilan rögzítve, ez a rögzítés egy megfelelően alakított gémkapocs segítségével került megoldásra. A potenciométer három kivezetéséhez egy-eres vezeték közvetlenül forrasztáson keresztül kapcsolódik. A vezetékek végére krimplhető NSR érintkezők kerültek felfogatásra, amelyek ezután NSR-03-as házba vannak beépítve. Összességében elmondható, hogy a trimmereknél is ez a huzalozási megoldás került kivitelezésre, mivel így egyszerűen csatlakoztathatók az Arduino-n található motor „shield”-hez, melynek analóg bemenetire érintkező tűskéket kellett forrasztani, hogy az útmérők csatlakoztathatók legyenek.

A következő lépés a második motor forgástengelyébe egy trimmer felszerelése volt. A trimmerek kis beszerelési méretük miatt kerültek kiválasztásra. Egy ipari ragasztópisztoly segítségével valósult meg a tengelyre való felfogatása. Ezt a 12. ábra-montázs jobb felső képe mutatja. Ebben az esetben a forgórész fog forogni, amelynek forgatásáról szintén egy megfelelően alakított gémkapocs gondoskodik, amely rögzítése a robot karjához történt. A harmadik motor esetében is trimmer alkalmazása volt a legmegfelelőbb az útmérésre. A gémkapocs itt egy speciálisan kialakított transzformátorlemezhez kapcsolódik, amelyet a 12. ábra-montázs bal alsó képén láthatunk. Ahogy forog a harmadik motor, úgy fogja a gémkapocs forgatni a trimmert. Az utolsó lépés a negyedik motor útmérésre alkalmassá tétele volt. A trimmert egy hajlított, lyukasztott transzformátor lemez tartja, a gémkapocs szintén transzformátorlemezre került rögzítésre, így biztosítva a diszkrét helyzetérzékelést. A 12. ábra-montázs jobb alsó sarkában figyelhető meg a negyedik trimmer felszerelése.

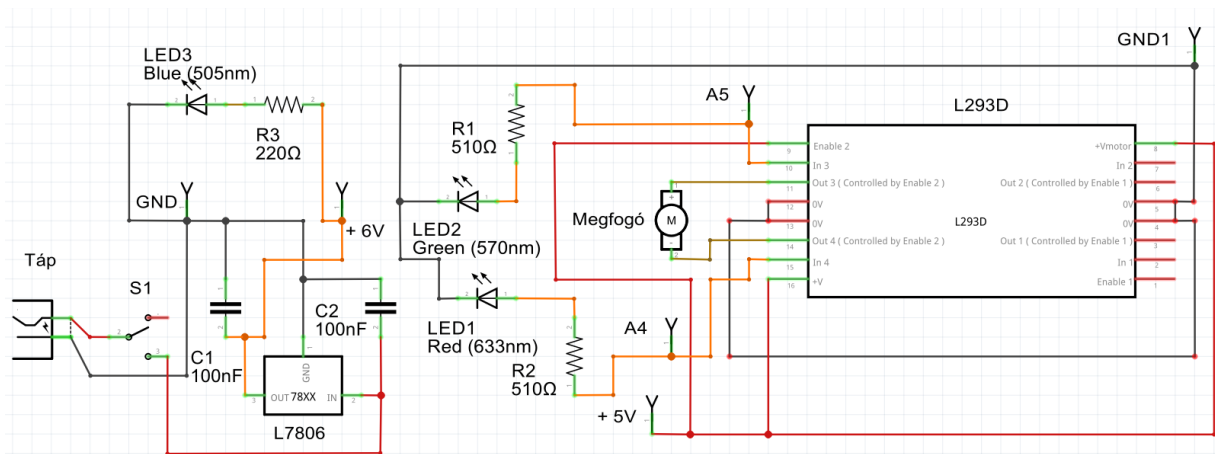


12. ábra Az útmérők felszerelése

4.2. A megfelelő tápellátás és megfogó vezérléséért felelős NYÁK tervezése

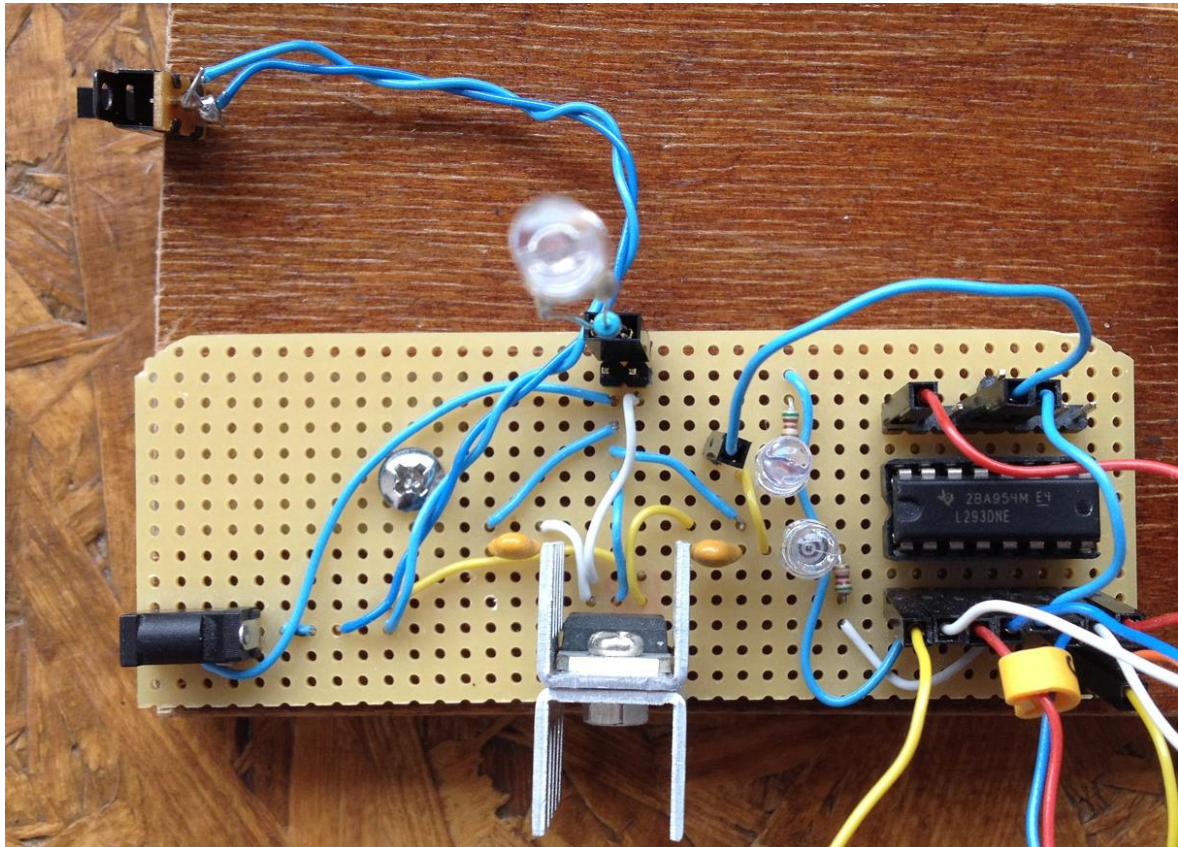
Az adapter 9 V értékű feszültségszintjét illeszteni kell a DC motorokhoz, ezért azt le kell transzformálni 6 V-ra, amely egy L7806 típusjelzésű IC-vel történik. Az elektronikai zajok elkerülése miatt két darab 100 nF értékű kerámia kondenzátor is bekötésre került. Egy hagyományos tolókapcsoló fog gondoskodni a könnyű ki- és bekapcsolásról.

Az L293D típusú H híd felelős a megfogó szorításáért, illetve elengedéséért. A megfogást egy piros, nagy fényerejű LED, illetve az elengedés pillanatát egy zöld szintén nagy fényerejű LED jelzi, melyekhez előtét ellenállás bekötése szükséges. A számolt előtét ellenállás helyett a biztonságos működés érdekében egy nagyobb ellenállás került beépítésre, számszerűen $R = 510 \Omega$. A LED-ek katódja közösítésre került. A 13. ábrán látható a kapcsolási rajza a megtervezett panelnak. A 14. ábra pedig a már elkészült próba-NYÁK-ot szemlélteti. A felső ábra a Fritzing nevű szoftver használatával készült el.



13. ábra A megtervezett kapcsolás

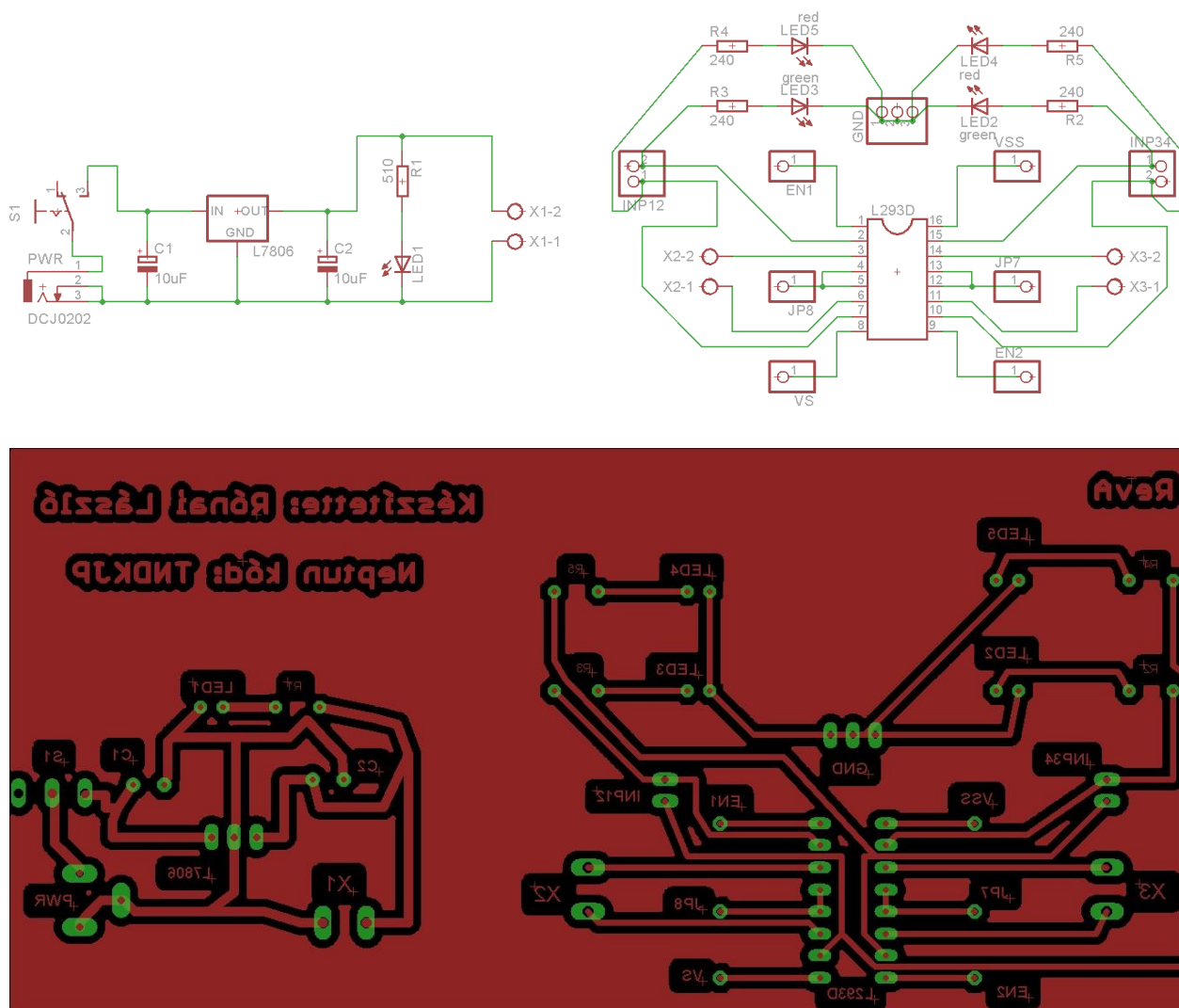
Az így előállított feszültségszint már megfelelő a motorok számára, és beköthető a motor „shield” tápcsatlakozójába. A megfogó működtetéséhez további kimenetekre van szükség, mivel a motor „shield” ezek nagy részét elhasználta, ezért az, A4 és A5-ök lesznek digitális kimenetként definiálva. A közvetlen 5 V-os csatlakozás az IC EN lábára adva fogja engedélyezni a működését, míg A4 és A5 digitálisként funkcionáló kimenetek fogják küldeni a vezérlő jelet, hogy mikor, melyik irányba működjön a megfogó.



14. ábra Az elkészült panel

Jól megfigyelhető, hogy az L7806-os IC-n két alumíniumból készült hűtőborda is található, hiszen a motorok működése közben melegedésre hajlamos. A panel, illetve maga a mikrovezérlő egy fa alaplapra került felfogatásra. Az L293D négyszeres fél H hidas IC-nek csak az egyik oldalára volt szükség, hiszen csak a megfogóban található motor vezérlését végzi.

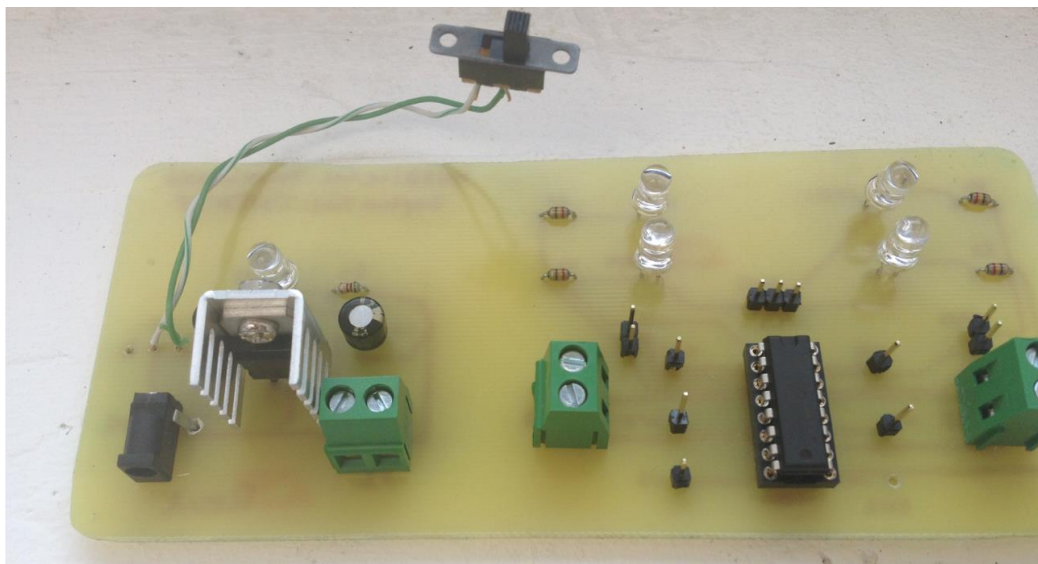
A saját összeállítású próba NYÁK elkészülése után, látva annak eredményes próba működését, továbbfejlesztésként egy kifinomultabb panel megtervezése vált szükségessé. Első lépésben a kapcsolási rajzot kellett elkészíteni, amely a NYÁK tervvel egyetemben az Eagle nevezetű tervező szoftverrel valósult meg. A 15. ábra mutatja a véglegesen elkészült kapcsolási rajzot, és az abból létrehozott NYÁK-tervet.



15. ábra A panel tervezés ábrái

A tervek elkészülte után a NYÁK-terv lézernyomtatóval egy etikett papírra kinyomtatásra került. Ezután a terv átvitele egy egyoldalas NYÁK lapkára házi vasalóval történt meg. Körülbelül 200 °C-on olvadt meg a nyomtatófesték, így adva lehetőséget a panelra való átfolyásának. Lehűlés után egy pár korrigáló beavatkozást kellett végezni alkoholos filctollal, hogy a kapcsolás még összefüggőbb, homogénebb legyen a maratás elvégzése után.

A Robert Bosch Mechatronikai Intézeti Tanszéken lehetőség nyílt kimaradni a NYÁK-tervet vas-kloridos fürdő segítségével. Körülbelül 2 óra kellett a felesleges réz lemaratásához. Miután elkészült a panel, Dremel fűrő segítségével a megfelelő helyeken elkészültek a megfelelő átmérőjű furatok. A forrasztási procedúra előtt a panel hátoldalát forrasztást elősegítő lakkal be kellett vonni. A 16. ábra mutatja az elkészült panelt. A maratási, fűrési folyamatokat a függelék tartalmazza.



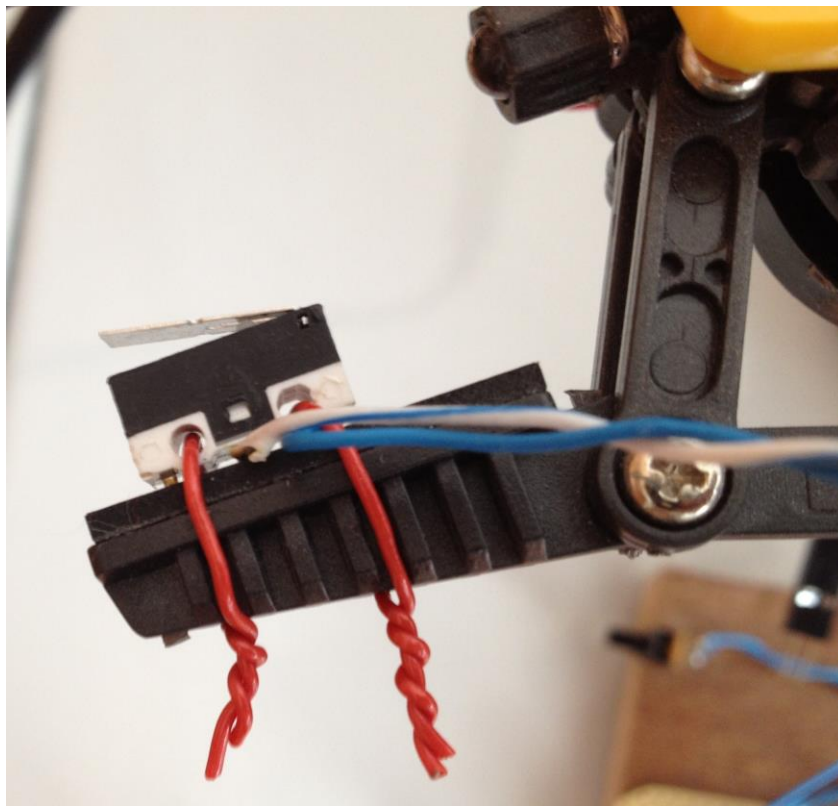
16. ábra Az elkészült vezérlő panel

Az elektronikai alkatrészek beültetése és ellenőrzése után használhatóvá vált a panel.

4.3. A megfogó „kvázi heptikussá” tétele

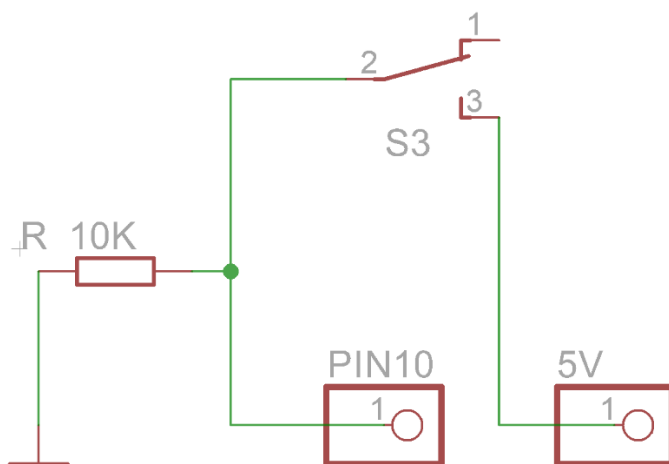
A robot működése során az egyik fontos szempont a munkadarabok manipulálása, mozgatása. A megfogó, amikor megfogja a munkadarabot, akkor kell valamilyen visszacsatolás, hogy elég erősen tartja ahhoz, hogy az ne essen ki a szorításból. A visszacsatolás egy mikrokapcsoló segítségével került megvalósításra. A 17. ábra mutatja a robot megfogójára applikált mikrokapcsolót. A motor „shield”-en található összes szabad digitális be- és kimenet használatban van már, ezért a panelen található két darab szervó pin alkalmazása volt a

megoldás. Mivel a modulra nem lett szervo motor csatlakoztatva, ezért a pin 9 és 10 digitális I/O-k szabadon konfigurálhatóvá váltak.



17. ábra A megfogóra szerelt mikrokapcsoló

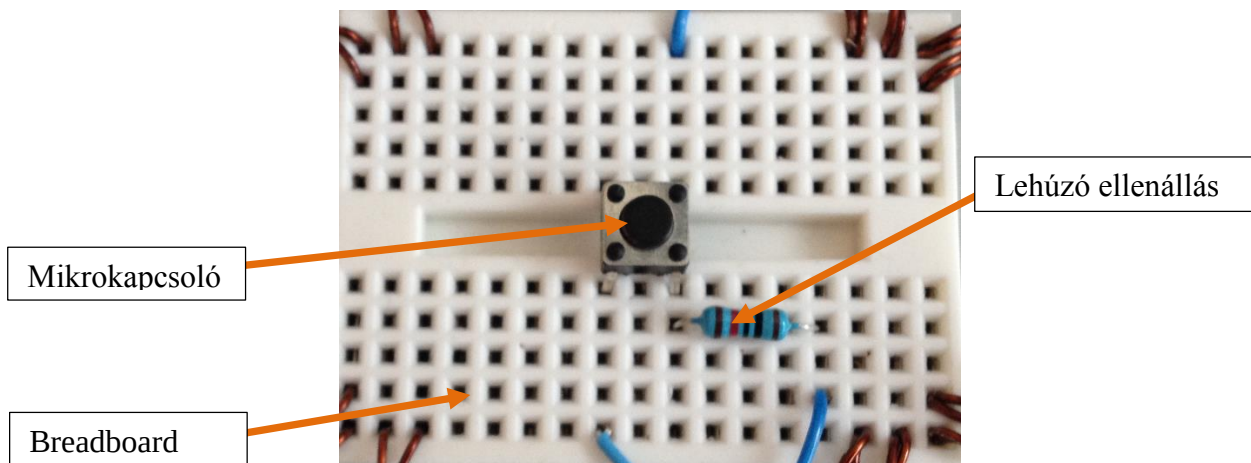
Az ábrán jól látható a kapcsoló kialakítása, annak rögzítése. A mikrovezérlő digitális bemenetéhez csatlakoztatásához még egy lehúzó ellenállás is bekötésre került, amelynek értéke 10 k Ω . A 18. ábra mutatja a kapcsoló illesztését a controllerhez.



18. ábra A lehúzó ellenállással bekötött kapcsoló

Mindezek után a robot megfogójának működése „kvázi heptikusnak” tekinthető.

A 20. ábrán található Tároló 1 telítettségét is hasonlóképpen jelzi a rendszer. A tároló alatt található egy nyomógomb, amely szintén egy lehúzó ellenállás segítségével csatlakozik a földponthoz. Amikor a nyomógomb benyomódik, azzal egy jelet küld a mikrokontroller Pin 2-es bementének, hogy a másik tárolóba kell pakolnia a robotnak, mert az elsődleges megtelt. A 19. ábra mutatja a Tároló 1 alatt kialakított részt.

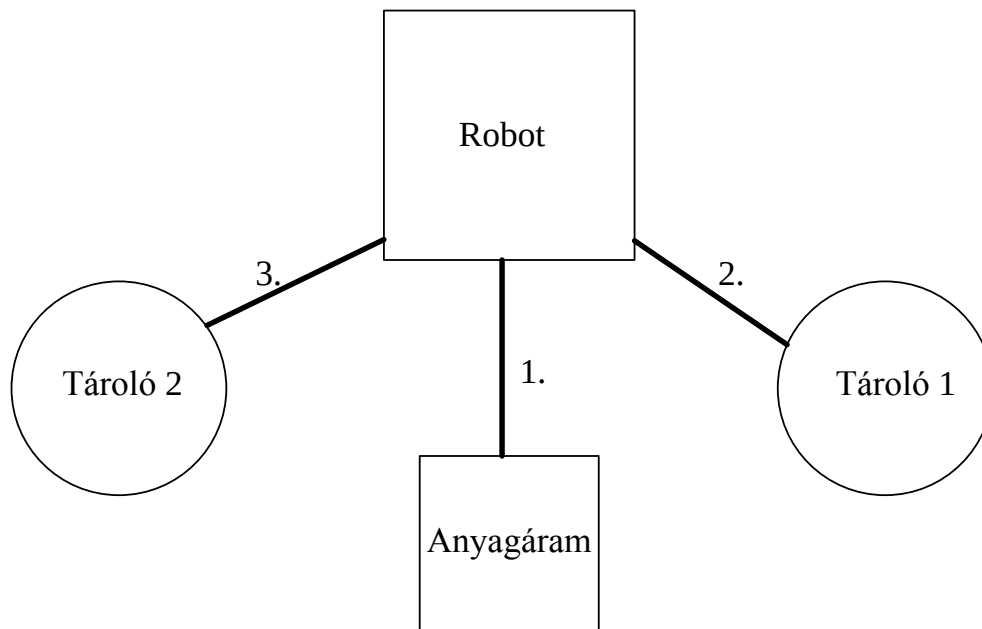


19. ábra A Tároló 1 alatt kialakított próbapanel (breadboard)

5. Az elkészült vezérlőprogram

A vezérlőprogram C nyelv alapú fejlesztőkörnyezetben valósult meg. Az első lépés a motor „shield-hez” használt könyvtár (library) meghívása volt. A következő pont a csuklókhoz tartozó motorok definiálása volt, amelyet a változók, segédváltozók és a portok deklarálása követett. A `void setup()`-on belüli rész a modul indulása után csak egyszer fut le, ezért itt kell a beállítási műveleteket elvégezni, mint például a motorok sebességének konfigurálását, amelyet PWM segítségével old meg a rendszer. Többek között itt kell megadni, hogy az egyes pinek kimenetek, vagy bemenetek legyenek. Lehetőség nyílik az analóg bemenetek digitális kimenetként való alkalmazására, amelyre szükség is volt, a megfogó működtetéséhez. A `void loop()` végtelenszer futó szegmense a programnak. Ide kerültek az elágazások, amelyek `if()` és `if() else` típusúak. Fontos megjegyezni, hogy a potenciométerek analóg feszültségértékét a modul 0-1023 intervallumon belül található egész számokra képezi le (10 bites felbontás jellemzi), amely a feladat elvégzéséhez elegendőnek bizonyult.

A modell-robot vezérlésének demonstrálásához készített program egy logisztikai problémát hivatott megoldani. A 20. ábra szemlélteti részletesen.

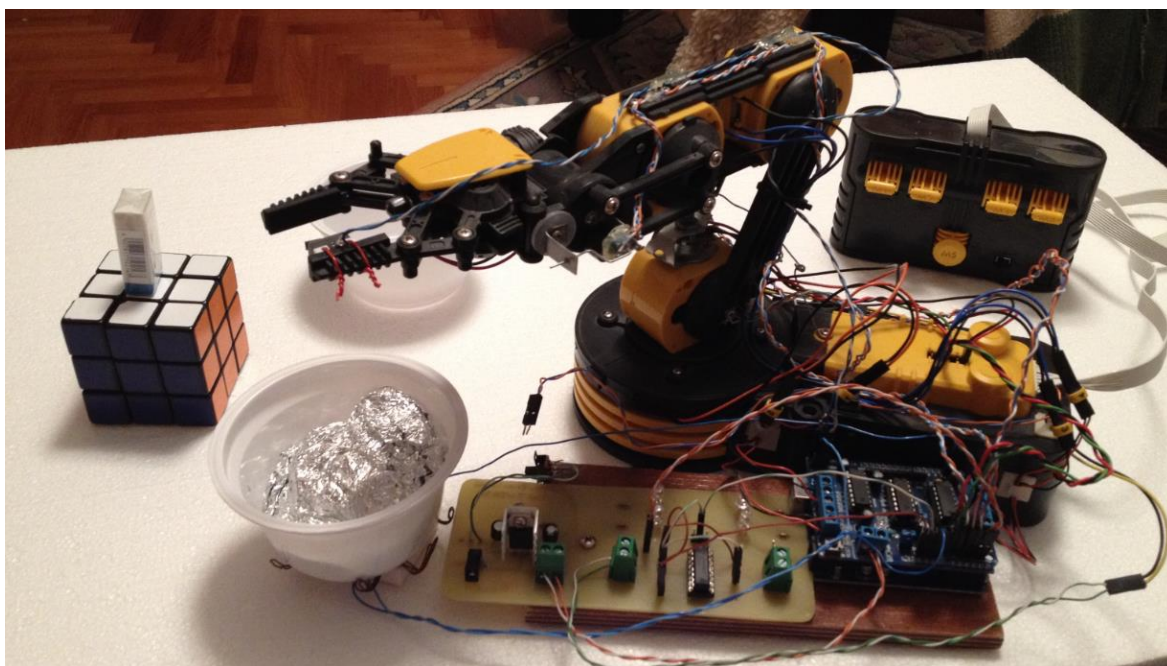


20. ábra A robot munkatere

A folyamat kezdetekor a robot 2., vagy a 3. pozícióban tartózkodik, a program eldönti, hogy melyikben is van ténylegesen, majd az 1. pozícióba mozog, hogy felvegye az onnan érkező anyagot. A megfogás után a 2. állásba indul, hogy a Tároló 1-be rakhassa az érkező anyagot. Összesen a folyamat kétszer ismétlődik meg, a ciklusszám természetesen szabadon módosítható. A „shield”-en található nyomógommbal a folyamat újraindítható. A folyamat során, ha a Tároló 1 megtelt, akkor a robot automatikusan az anyag felvétele után már a Tároló 2-be fogja pakolni az anyagot. A tároló telítettségét egy nyomógomb fogja jelezni, amely a mikrovezérlő kettes digitális bementére küldi majd a jelet.

A modell robot a jelenlegi állapotában a mozgási műveleteket egymás után végrehajtott csuklómozgásokkal végzi el.

A 21. ábrán látható az elkészült rendszer teljes egésze. Jól megfigyelhető a vezérlés, az útmérők elhelyezkedése.



21. ábra Az elkészült robot-vezérlés

6. Összefoglalás

A Tudományos Diákköri Kutatómunkám során sikerült egy kereskedelmi forgalomban kapható modell-robotot automatizálni és egy egyszerű intelligens rakodási feladatra beprogramozni.

Az áttekintett irodalmak alapján kidolgoztam az adott modell direkt-, valamint az inverz kinematikai feladatát és szimulációs programját. A szimulációval kapott görbék a robot vezérlésének alapjeleiül szolgálhatnak. A robot vezérelhetővé tételéhez hardveres továbbfejlesztéseket is kellett végezni, melyek többek között az útmérők, mikrokapcsolók, megfogó vezérlő- és feszültség szabályozó panel felszerelése, megtervezése volt. A nyomtatott áramköri lapot az Eagle nevezetű NYÁK tervező programban készítettem el. A robothoz egy Arduino UNO fejlesztőplatformon található AVR mikrovezérlőt kellett illeszteni, amelyre kiegészítésként egy motor „shield” is került. A mikrokontroller analóg bemeneteire bekötöttem az útmérőket, valamint a digitális bemeneteire a megfelelő mikrokapcsolókat, amelyek a tároló telítettségét, illetve a megfogó „kvázi heptikusságát” hivatottak reprezentálni. A rakodási feladat megoldására C nyelvű programot írtam.

A jövőbeni tervek között szerepel a robot demonstrációs feladatokra való alkalmazása. A jelenlegi sorrendi vezérlést szeretnénk továbbfejleszteni, úgy, hogy a robot képes legyen egyszerre mozgatni az összes csuklóját.

Köszönetnyilvánítás

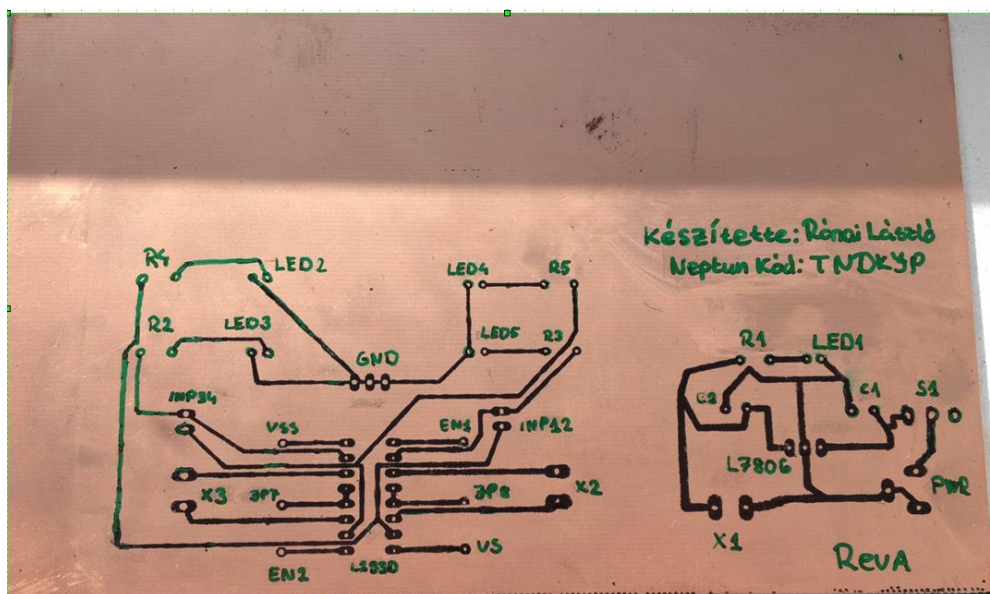
A kutatómunka a Miskolci Egyetem stratégiai kutatás területén működő Mechatronikai és Logisztikai Kiválósági Központ keretében valósult meg.

Irodalomjegyzék

- [1] Király B.: Ipari robotok kinematikai és dinamikai elemzése, oktatási segédlet, Miskolc, 1995.
- [2] Spong, W. M. – Hutchinson, S. – Vidyasagar, M.: Robot Modelling and Control, Wiley, 2006
- [3] Arduino platformok: <http://arduino.cc/>
- [4] Velleman® KSR10 Robotic Arm – User Manual
- [5] Rónai L., Szabó T.: Egy 4 szabadságfokú modell-robot inverz kinematikai feladata, OGÉT konferencia, Nagyszeben, 2014
- [6] Rónai L., Cservenák Á.: Robot kiszolgáló rendszer kinematikai szimulációja, Házi TDK Konferencia, 2012

Függelék

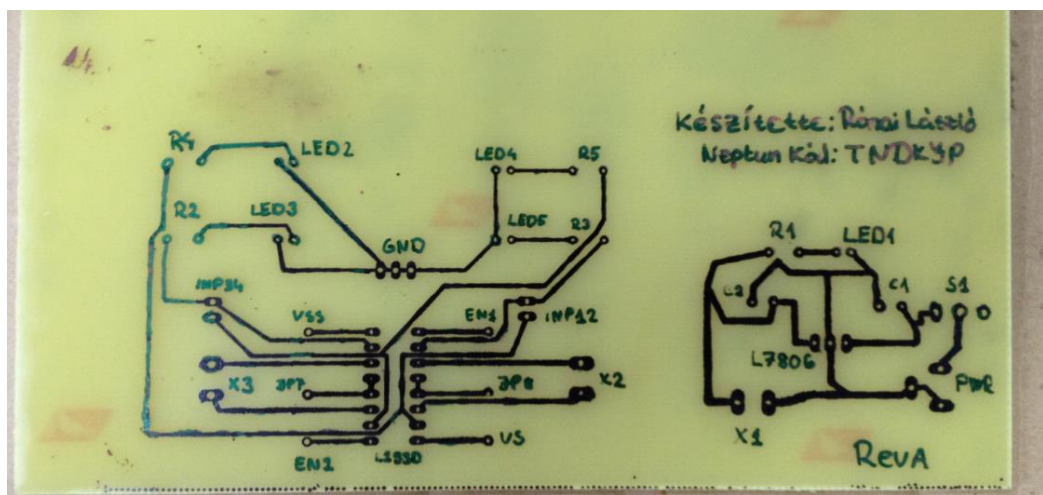
1. A NYÁK gyártási folyamata



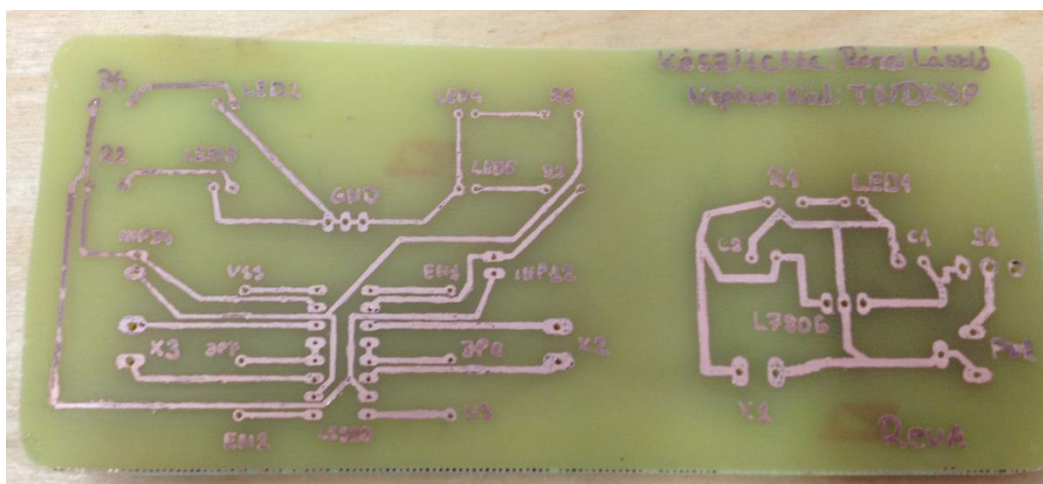
Függelék 1. ábra A nyomtatófesték felvitele



Függelék 2. ábra A maratási folyamat



Függelék 3. ábra A kimaratott panel



Függelék 4. ábra A készre fűrt, tisztított panel

2. Az Arduinora megírt vezérlő program forráskódja

```
#include <AFMotor.h>           // Motor library (könyvtár) meghívása.

AF_DCMotor motor1(1);          // Az egyes csuklók motorainak deklarálása.
AF_DCMotor motor2(2);
AF_DCMotor motor3(3);
AF_DCMotor motor4(4);

int sensPin1 = A0;              // A potméterek analóg értékeinek deklarálása, helyének
                                // megadása.

int sensPin2 = A1;
int sensPin3 = A2;
int sensPin4 = A3;
int gomb = 2;
int megfogo = 10;
int allapot = 0;
int csat1 = A4;                 // A megfogóhoz szükséges kimenetek, itt az analóg kimenet
                                // digitálisként funkcionál.

int csat2 = A5;

int sensValue1 = 0;             // Segédváltozó kinullázása, ahová az analóg jelszintnek
                                // megfelelő érték fog letárolódni.
```

```
int sensValue2 = 0;
```

```
int sensValue3 = 0;
```

```
int sensValue4 = 0;
```

```
int valt = 0;           // Segédváltozók az if feltételeihez.
```

```
int valt2 = 0;
```

```
int i =0;
```

```
void setup()
```

```
{
```

```
    pinMode(gomb, INPUT);           // A tároló telítettségének ellenőrzésére szolgáló gomb.
```

```
    pinMode(csat1, OUTPUT);         // Kimenetre állítása A4 és A5-nek.
```

```
    pinMode(csat2, OUTPUT);
```

```
    motor1.setSpeed(150);           // Motorok sebességének beállítása.
```

```
    motor2.setSpeed(220);
```

```
    motor3.setSpeed(150);
```

```
    motor4.setSpeed(150);
```

```
}
```

```
void loop()
```

```
{
```

```
allapot = digitalRead(gomb);      // Az allapot nevű változó feltöltése a digitális bemenet  
                                   aktuális értékével.
```

```
sensValue1 = analogRead(sensPin1); // A segédváltozók feltöltése a potméterek aktuális  
                                   értékével.
```

```
sensValue2 = analogRead(sensPin2);
```

```
sensValue3 = analogRead(sensPin3);
```

```
sensValue4 = analogRead(sensPin4);
```

```
if (i < 2)                        // Ismétlési szám megadása.
```

```
{
```

```
if(allapot == LOW)                // A tároló 1 amennyiben nincs megtelve, akkor oda pakol
```

```
{
```

```
if (valt == 0)                    // Feltételek sokasága, amelyekkel vezérelhető a robot.
```

```
{
```

```
if (sensValue3 > 660 && sensValue3 < 750)
```

```
{
```

```
    motor3.run(BACKWARD);
```

```
}
```

```
if (sensValue3 == 558)
```

```
{
```

```
    motor3.run(RELEASE);
```

```
    motor4.run(FORWARD);
```

```

}

if(sensValue4 == 730)

{

    if(sensValue1 < 500)

    {

        motor4.run(RELEASE);

        motor1.run(BACKWARD);

    }

    if(sensValue1 > 600)

    {

        motor4.run(RELEASE);

        motor1.run(FORWARD);

    }

}

if(sensValue1 == 520)

{

    motor1.run(RELEASE);

    motor2.run(BACKWARD);

}

if(sensValue2 == 320)

{

    motor2.run(RELEASE);

    delay(500);

```

```

digitalWrite(csat1, HIGH);

digitalWrite(csat2, LOW);

delay(500);

while (digitalRead(megfogo) == 0) {}

digitalWrite(csat1, LOW);

digitalWrite(csat2, LOW);

delay(2000);

valt++;

}

}

if (valt == 1)

{

    if (sensValue2 > 200 && sensValue2 < 350)

    {

        motor2.run(FORWARD);

    }

    if(sensValue2 == 530)

    {

        motor2.run(RELEASE);

        motor3.run(FORWARD);

    }

    if(sensValue3 == 690)

    {

```

```

    motor3.run(RELEASE);

    motor4.run(BACKWARD);

}

if(sensValue4 == 655)

{

    motor4.run(RELEASE);

    motor1.run(BACKWARD);

}

if (sensValue1 == 610)

{

    motor1.run(RELEASE);

    delay(2000);

    digitalWrite(csat1, LOW);

    digitalWrite(csat2, HIGH);

    delay(500);

    digitalWrite(csat1, LOW);

    digitalWrite(csat2, LOW);

    delay(2000);

    valt--;

    i++;

}

}

}

```

```

else

{

if(valt2 == 0)

{

if (sensValue3 > 660 && sensValue3 < 750)

{

motor3.run(BACKWARD);

}

if (sensValue3 == 558)

{

motor3.run(RELEASE);

motor4.run(FORWARD);

}

if(sensValue4 == 730)

{

if(sensValue1 < 500)

{

motor4.run(RELEASE);

motor1.run(BACKWARD);

}

if(sensValue1 > 600)

{

motor4.run(RELEASE);

```

```

        motor1.run(FORWARD);

    }

}

if(sensValue1 == 520)

{

    motor1.run(RELEASE);

    motor2.run(BACKWARD);

}

if(sensValue2 == 320)

{

    motor2.run(RELEASE);

    delay(500);

    digitalWrite(csat1, HIGH);

    digitalWrite(csat2, LOW);

    while (digitalRead(megfogo) == 0) {}

    digitalWrite(csat1, LOW);

    digitalWrite(csat2, LOW);

    delay(2000);

    valt2++;

}

}

if (valt2 == 1)

{

```

```
if (sensValue2 > 200 && sensValue2 < 350)
```

```
{
```

```
    motor2.run(FORWARD);
```

```
}
```

```
if(sensValue2 == 530)
```

```
{
```

```
    motor2.run(RELEASE);
```

```
    motor3.run(FORWARD);
```

```
}
```

```
if(sensValue3 == 690)
```

```
{
```

```
    motor3.run(RELEASE);
```

```
    motor4.run(BACKWARD);
```

```
}
```

```
if(sensValue4 == 655)
```

```
{
```

```
    motor4.run(RELEASE);
```

```
    motor1.run(FORWARD);
```

```
}
```

```
if (sensValue1 == 440)
```

```
{
```

```
    motor1.run(RELEASE);
```

```
    delay(2000);
```

```
digitalWrite(cs1, LOW);  
digitalWrite(cs2, HIGH);  
delay(500);  
digitalWrite(cs1, LOW);  
digitalWrite(cs2, LOW);  
delay(2000);  
val2--;  
  
i++;  
  
}  
  
}  
  
}  
  
}  
  
}
```