

**Óbudai Egyetem
Kandó Kálmán Villamosmérnöki Kar
Automatika Intézet**



TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

RUBIK-KOCKA KIRAKÓ ROBOT

Szerző:

Takács Bence

villamosmérnöki szak, III. évfolyam

Konzulens:

Major László

docens

Budapest, 2012

Tartalomjegyzék

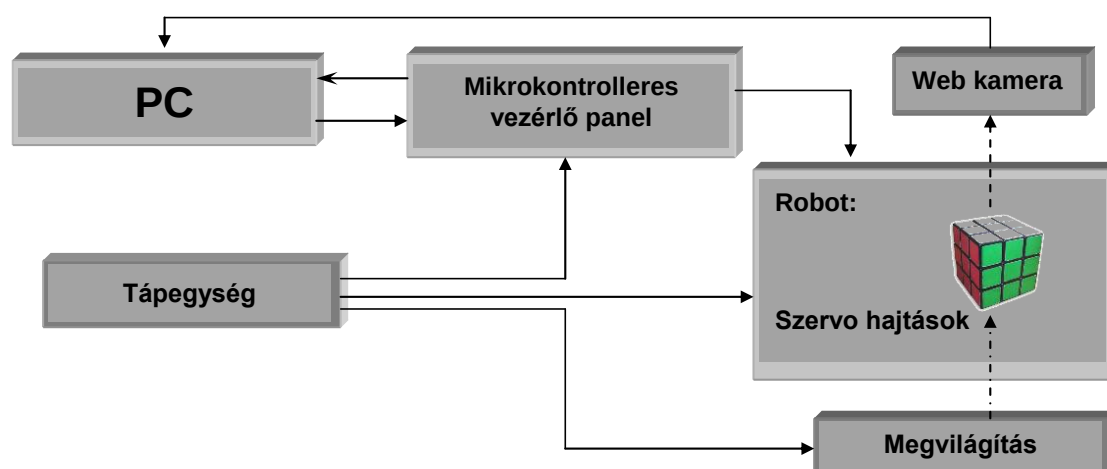
Bevezetés:	2
1. Számítógépen futtatott vezérlő program:	3
1.1 Általános program leírás:	3
1.2 Vezérlő program részletes ismertetése:	4
2. Kommunikáció:	6
3. Algoritmusok leírása:	7
3.1 Amatőr algoritmus részletesen:	8
3.2 Általam használt algoritmus leírása:	8
3.3 A „profí” algoritmusokról:	10
4. Képelemzés:	11
4.1 Jelenlegi képelemző szoftver:	11
5. Energia ellátás:	13
6. Szervo hajtások:	14
6.1 Vezérlés:	14
6.2 Belső felépítés:	15
6.3 Szervók általános paraméterei:	17
6.4 Általam használt szervo hajtás:	17
7. Mikrokontrolleres vezérlő panel:	19
7.1 Program:	19
7.2 Vezérlő panel készítése:	20
8. Mechanikai megvalósítás:	21
8.1 3D tervek elkészítése:	21
8.2 Felhasznált anyagok:	21
9. Értékelés:	23
10. További fejlesztési lehetőségek:	24
10.1 Vezérlő program fejlesztése:	24
10.2 Kommunikáció további fejlesztése:	25
10.3 Képelemzés további fejlesztései:	26
10.4 Algoritmusfejlesztés:	26
10.5 Tápellátás további fejlesztései:	26
10.6 Végleges külső kialakítás:	27
11. Köszönetnyilvánítás:	27
12. Felhasznált irodalmak:	28
12.1 Rubik-kocka kirakási algoritmusok forrásai:	28
12.2 Vezérlő kontroller programfejlesztésben felhasznált irodalom:	28
12.3 Szervo hajtások fejezetben felhasznált források:	28
12.4 Képi források:	28

Bevezetés:

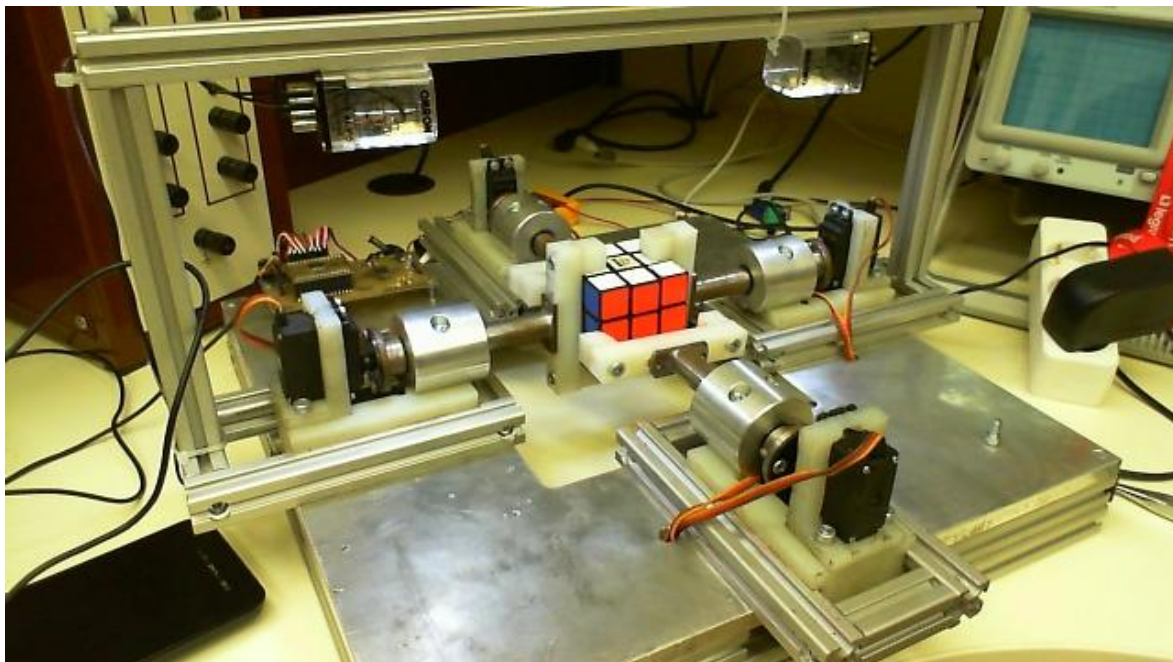
A dolgozat témájának ötlete 2012 tavaszán, a kockázás, robotok és a programozás iránti elszántság következtében született meg. A kezdeti gondolatot elhatározás követte: készítek egy célorientált robotot, mely képes autonóm módon kirakni a szabványos 3x3x3 –as Rubik-kockát.

A munka során megvalósított berendezés funkcióját tekintve az egyes automatizálási elemek használatának, azok összehangolt működésének bemutatása. A célul tűzött feladat megoldásában példát láthatunk automatizált képelemzésre, PC-s műveletirányításra, PC és mikrokontrolleres vezérlő együttműködésére, a szervo hajtások vezérlésére,

A dolgozatban megtalálható többek között Rubik-kocka kirakásához használható (és a megoldás során használt) algoritmusok ismertetése, a számítógépen futtatott program felépítése, működési elve, a képelemzés közben felmerült problémák és azok megoldásai, a PIC mikrokontrollerben futó program felépítése, a vezérlő panel felépítése, a működtető szervo hajtások ismertetése, azoknak vezérlése, és a mechanikai megvalósítások valamint a robot energiaellátása. Az 1. ábrán látható a teljes rendszer áttekintő tömbvázlata.



1. ábra A teljes rendszer áttekintő tömbvázlata.



2. ábra Az elkészült Rubik-kocka kirakó robot

1. Számítógépen futtatott vezérlő program:

1.1 Általános program leírás:

A vezérlő program fejlesztése C nyelven történt. Az felhasználói felületnek köszönhetően egyszerűen kezelhető. A program fejlesztése Microsoft Visual Stúdió fejlesztőkörnyezetben történ. Azért erre esett a választás, mert a Microsoft oldalán található sok leírás az egyes függvények használatáról, például a soros port kezelése, hiszen a kommunikációs csatolást egy FTDI USB-SOROSPORT átalakító látja el. A jelenleg futó program a V1.1-es verzió számot viseli. Az ezt megelőző, először elkészült program forgatási számban és ebből következően kirakási időben is nagy terjedelmű volt. A V1.0 verzióban csak az „amatőr” forgatási algoritmusokat tartalmazta. Így a forgatási idő átlaga 6 perc körül alakult, valamint a forgatási szám átlagosan 150 db. A fejlesztett a V1.1-es verzióba bekerült jó néhány „profi” algoritmus is, mellyel több, mint 25%-al csökkent le az idő valamint a forgatási szám. Az idő lecsökkent átlagosan 4 percre, de vannak olyan keverések melyeket közel 2 perc alatt képes kirakni, eddigi rekordja 2 perc 15 másodperc. Az átlagos forgatási szám 100, a legrövidebb kirakási időhöz 75 forgatás tartozik. Egy forgatási művelet ideje nagymértékben függ attól, hogy milyen forgatásokat használ a kirakás során, hiszen a mechanikai korlátok miatt 4 kar helyezkedik el a roboton, így a kockának csak két térbeli tengelyét képes forgatni, a harmadik tengely forgatásához 90°-al el kell fordítania a kockát. Egy olyan forgatás, amely 90°-os elforgatást tartalmaz négyszer annyi ideig tart, mint

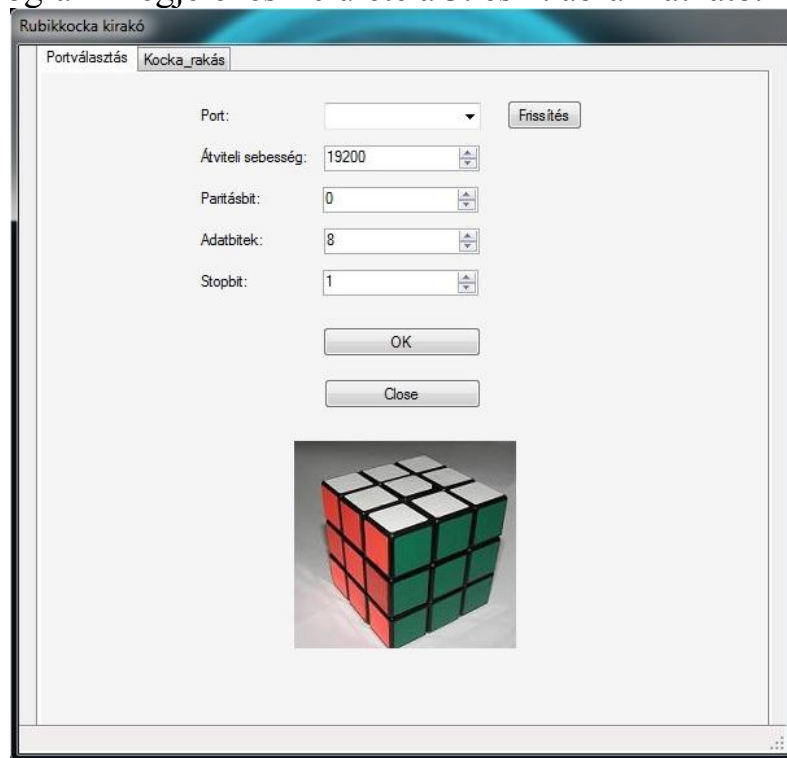
egy normál forgatás. A V1.1-es programmal átlag 1,8 másodperc egy forgatás. Ezen verzió írásakor nagy figyelmet fordítottam arra, hogy ezeket a 90°-os elforgatásokat tartalmazó lépéseket minél kevesebbet használjam, hiszen még ha két normál forgatással is tudja elvégezni ugyan azt a műveletet, akkor is fele időbe telik. A „profi” algoritmusok tovább rövidítették a forgatási számot, például a sárga oldal sarkainak kirakásakor egyes keverési kombinációknál akár 10 vagy akár 20 lépést is rövidíthet. Azonban készül a V2.0-ás verzió, amely remélhetőleg 2 perc alá szorítja az átlag kirakás idejét. Ez egy teljes „profi” algoritmust fog tartalmazni. Ebben segítséget nyújt egy ismerős, aki a kockát magas szinten kezeli, eddigi rekordja 11 másodperc körül van. Rávilágított, hogy a robot hogyan forgathatná gyorsabban a kockát, valamint ellátott forgatási algoritmus leírásokkal. Ezeknek segítségével a forgatási szám várhatóan 60 körülire fog mérséklődni, és elérhető lesz a 2 perc alatti átlag kirakás. Vannak algoritmusok, amelyek még ennél is kevesebb forgatási számmal képesek kirakni a kockát, azonban ezek programba való megírása nem egyszerű, a további fejlesztésre váró feladat.

Így a szervo hajtások sebességi korlátai miatt nem érhető el további gyorsítás.

1.2 Vezérlő program részletes ismertetése:

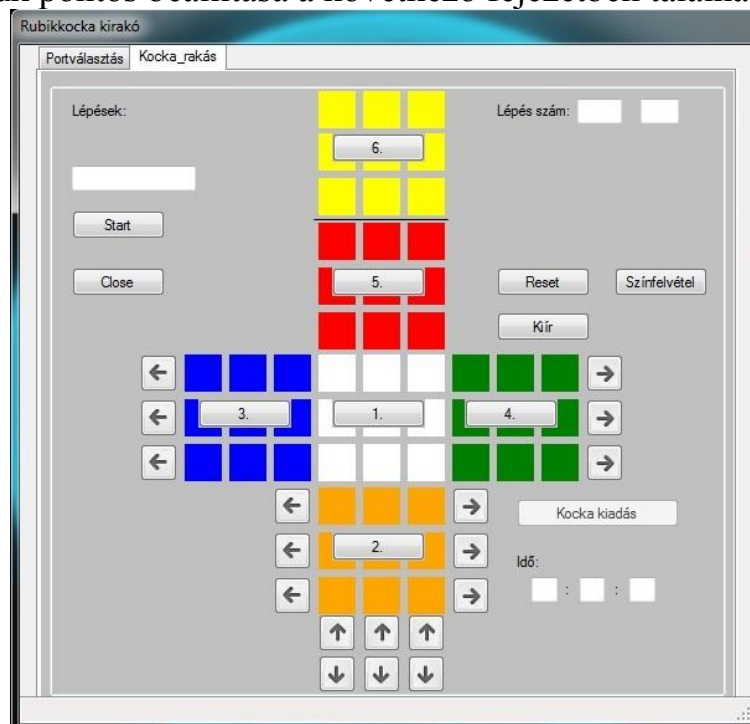
Program alkalmazói felülete:

A vezérlőprogram megjelenési felülete a 3. és 4. ábrán látható.



3. ábra Vezérlő program kommunikációs lap

A kommunikációs beállítások itt végezhetők el. A kommunikáció leírása valamint annak pontos beállítása a következő fejezetben található meg.



4. ábra Vezérlő program kocka kirakó lap

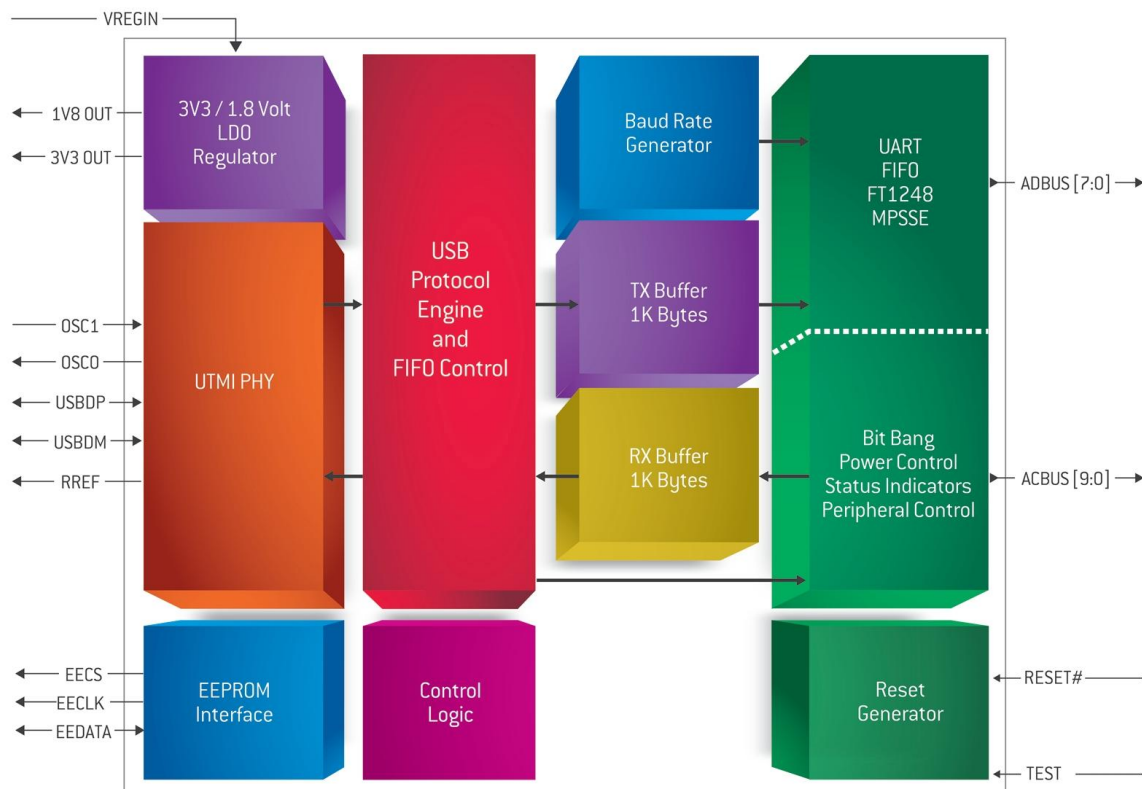
A vezérlő program második lapján található egy kiterített Rubik-kocka. Ide tölti be a program a web kamera képét elemezve az egyes kockák színeit. A Színfelvétel gombra kattintva a robot automatikusan felveszi a kocka, mind a hat oldalának színeit. Ha esetlegesen a színfelvétel nem hibamentesen történne, akkor lehetséges az egyes oldalaknak a külön-külön történő színfelvétele az 1... 6. kezelő gombokkal. Ekkor csak a kocka egyik oldalát mutatja meg a kamerának.

A Start gombbal indítható a kirakás. Helyet kapott a felületen egy óra, amely méri a kirakás idejét, a színfelvétel ideje nem számít bele az időbe. A kocka a kirakás után alaphelyzetbe kerül (fehér oldal van felül), akkor a Kocka kiadás gomb megnyomásával a robot kiadja a kockát. A kockát manuálisan is lehetséges forgatni, ez azonban csak abban az esetben lehetséges, ha a annak kirakási művelete nem lett elindítva. Ekkor a megfelelő nyílra kattintva, a végezhető el a kívánt forgatás.

A vezérlőprogram futtatása Windows operációs rendszer alatt lehetséges.

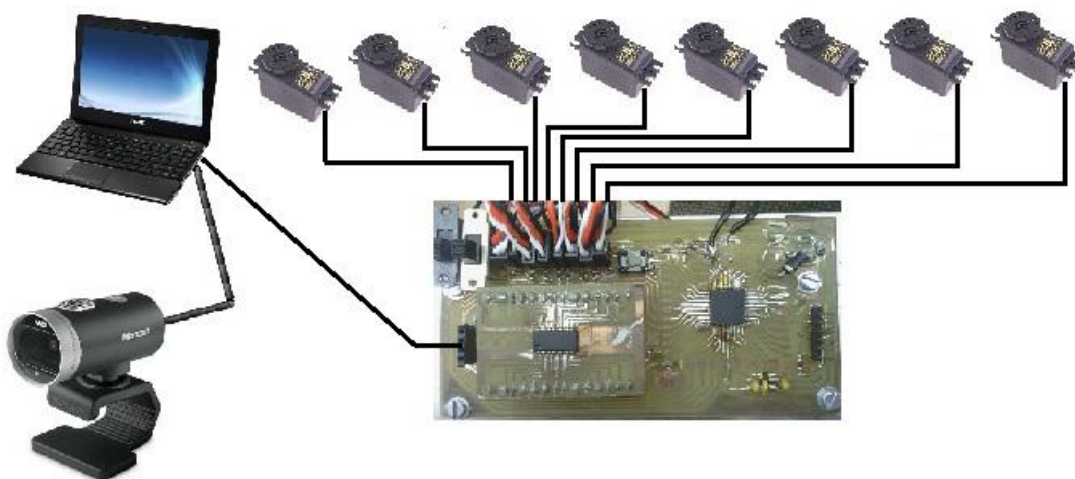
2. Kommunikáció:

A kommunikációt egy USB-SOROSPORT átalakító chipvalósítja meg, amelyet az FTDI (Future Tehnology Devices International Ltd) cég gyárt. A pontos típusa FT232RL, blokkvázlata az 5. ábrán látható.



5. ábra TF232RL belső blokk vázlata [1]

Az átalakító a PC felé USB HID-al (Human Interface Device) kapcsolódik és több kimenettel is rendelkezik, mint például párhuzamos port, szinkron soros port, a robot aszinkron soros port-kimenetet használja. A PC ezen keresztül kommunikál a vezérlő controllerrel, PC oldalról sorosportként érzékelhető. A PIC mikrokontroller EUSART (ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER) perifériával csatlakozik az átalakítóhoz. Az aszinkron kommunikáció sebessége 19200 baud, amely 10 bites kommunikáció esetén 1920 kbit/s-nak felel meg. Az adatforgalom mindössze 1 bájt mind a PC felől, mind a controller felől. A PC elküldi a forgatásnak megfelelő számértéket, amelyből a controller eldönti milyen forgatás kell elvégeznie, ha megtörtént a forgatás a kocka és a karok alaphelyzetben vannak, akkor küld egy nyugtázó bájtot a PC felé, hogy küldheti a következő adatot. A megvalósítás egyes elemeinek kapcsolatát a 6. ábra mutatja be.



6. ábra Egyes elemek közti kapcsolat

3. Algoritmusok leírása:

A Rubik-kocka kirakásának teljes algoritmus leírása hosszadalmas jelentős terjedelem idényű, így dolgozatomban csak a feladat megoldása szempontjából lényeges algoritmus részeket emeltem ki.

A robot által jelenleg használt algoritmus alap keretét tekintve „amatőr”, viszont a korábban említett néhány „profí” algoritmus beiktatásával gyorsabb lett.

Forgatások betűs rövidítései:

- L: Bal oszlop forgatása (Left)
- R: Jobb oszlop forgatása (Right)
- B: Túló (hátsó) lap forgatása (Back)
- F: Szemközti (elülső) lap forgatása (Front)
- U: Felső sor forgatása (Up)
- D: Alsó sor forgatása (Down)
- x: Teljes kocka fordítása a nézés síkjára vízszintes tengelyen
- y: Teljes kocka fordítása a nézés síkjára függőleges tengelyen
- z: Teljes kocka fordítása a nézés síkjára párhuzamosan tengelyen
- M: Középső oszlop forgatása (Middle)
- E: Középső sor forgatása
- S: Középső lap forgatása

A F, S, B forgatásokhoz kell elfordítania a robotnak a kockát x, y vagy z szerint.

3.1 Amatőr algoritmus részletesen:

A kezdők által használt kirakási módszer 6 fő lépésből épül fel:

1. fehér kereszt (élek) kirakása,
2. fehér sarkok,
3. középső sor éleinek rendezése,
4. sárga kereszt (élek) kirakása,
5. sárga sarkok orientálása,
6. sárga sarkok permutálása.

A fehér kereszt, sarkok valamint a középső sor éleinek rendezése a kocka kirakás első lépése. Ennek megoldására számtalan lehetőség kínálkozik, a kezdő kockaforgatók is önállóan egyéni megoldásokkal jutnak el az eredményhez.

A sárga kereszt kirakása két lépésből tevődik össze:

- 4.1 sárga élek orientálása,
majd a sárga sarkok orientálása után
- 4.2 sárga élek permutálása.

Az élek orientálása közben a kereszt elemek sárga oldalszíne számít (többi szín közömbös), így ha a kereszt elkészült, az élek még nem a végleges helyükre kerülnek, a permutálás lépés alatt a megfelelő élek cseréjével kerülnek végleges pozíciójukba.

A sárga sarkok orientálása közben csak az egyes elemek sárga oldalszíne számít (többi szín közömbös), nem fontos foglalkozni az elemen elhelyezkedő további színekkel. Ezt követően a sárga sarkok permutálása lépésben a sárga sarkok tényleges helyükre kerülnek. Így elkészül a kocka.

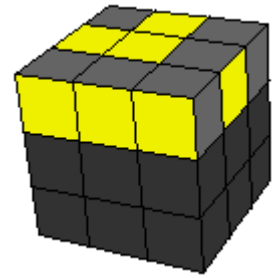
A kirakásról további részletek találhatóak a www.rubikkocka.hu oldalon.

3.2 Általam használt algoritmus leírása:

A kezdő algoritmushoz hasonlóan az én programom is 6 fő lépésre épül. Az eltérés mindössze annyi, hogy a fehér sarkok helyrerakásánál programom 1 sarkot kihagy, tovább lép a középső élek kirakására majd behelyezi a kimaradó sarkot a helyére. Erre ezért volt szükség, mert így a középső élek behelyezése kevesebb forgatással és rövidebb idő alatt elvégezhető, valamint a robotnál fontos szempont hogy nem használja azokat a forgatásokat, amelyek 90°-os átbillentést (F, S, B) igényelnek. Ezzel sikerült az első 3 lépés idejét átlag 3 percről átlag 2 percre csökkenteni.

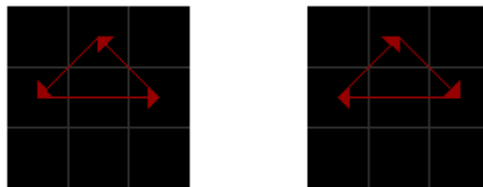
A kezdő algoritmusról továbblépve a 4. 5. 6. lépésekben használat algoritmusok közé bekerültek „profí” algoritmusok:

4.1 lépésben az amatőr kirakási algoritmus szerint csak a következő esetre van egy lépés sorozat, amelyet megfelelő ideig ismételve a sárga kereszt elkészül. Az algoritmus a következő: $F R' F' R U R U' R'$



Így ha a sárga oldalon a kis L, I vagy csak egy pont helyezkedik el, akkor ezzel az algoritmussal bekerül, ha egyszer, kétszer vagy háromszor elvégezzük, ez attól függ melyik eset áll fenn. Ebből következően nem lesz állandó a forgatások száma. Azonban az én programom tartalmaz minden eshetőségre egy-egy algoritmust így a forgatások száma, valamint elvégzésének ideje közel azonos.

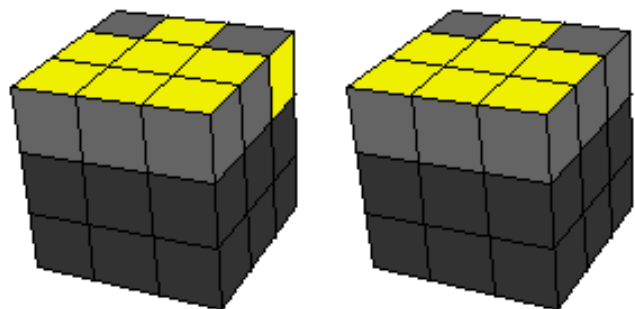
4.2 lépésben az „amatőr” algoritmushoz hozzá illesztettem még további 2 algoritmust, amelyek a következők:



7. ábra A 4.2 lépésbe beiktatott plusz algoritmusok képi magyarázata [2]

Ezek olyan algoritmusok, amelyek a 7. ábrán látható elemeket a nyíllal megegyező sorrendben felcserélik, azonban a kocka további elemét nem forgatják el helyeikről.

Az 5. lépésben csak két algoritmus található az „amatőr” kirakás szerint ezek a következők:



A két variáció algoritmus törzse ugyan az, csak van néhány előzetes lépés, amellyel a kocka előkészül.

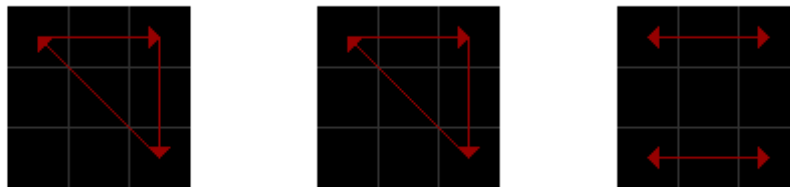
Algoritmus:

Az első esetben a kockának oly módon kell elhelyezkednie, hogy a sárga oldal van felül és jobb oldalt található a két rosszul álló sarok e esetben. A második esetben a kocka oly módon helyezkedik el, hogy a hiányzó két sárga lap felfelé álljon

A forgatási algoritmus mind két esetben a következő:

L U L' U L 2U L' R' U' R U' R' 2U' R

A 6. lépésben, az amatőr algoritmusban az L permutációt használják, azonban én bele tettem még további három algoritmust, amelyek a következők:



8. ábra 6. lépésbe beiktatott plusz algoritmusok képi magyarázata [3]

3.3 A „profí” algoritmusokról:

Az interneten található jó néhány profik által megalkotott metódus eljárás. Ezek közül néhány:

- Fridrich metódus
- Gilles Roux's metódus
- Lars Petrus metódus
- Zborowski-Bruchmen (ZB) metódus

Az általam tervezett tovább fejlesztésekben teljes egészen Fridrich metódus szerepel. Ezzel az algoritmussal maximum 56 lépésben kirakható a kocka, ez a robotom esetében némileg megnövekszik amiatt, hogy kiküszöböljem a 90°-os elfordítást, igénylő lépéseket. Elméletben ezzel az algoritmussal 17 másodperc alatt kirakható a kocka, viszont a megépített robot mechanikai korlátai miatt ez körülbelül 2 perces átlag kirakást eredményezne.

Fridrich metódus lépései:

1. Fehér kereszt (ugyan az mint az amatőr kirakásban)
2. F2L ebben a lépésben a fehér sarok valamint második sor élei egy lépésben bekerülnek a helyükre
3. OLL sárga oldalra a sárga lapok felfordítása további lapok figyelembe vétele nélkül.
4. PLL előző lépésben elhanyagolt lapok helyre tétele.

Fridrich metódus 120 algoritmust tartalmaz az általam betáplált programmal szemben, ami 20.

A teljes „profí” algoritmusokat tartalmazó program azért nem készült el, mert a csaknem 6-szor annyi algoritmust igen nehézkel leprogramozni, hiszen a programban fel kell állítani az össze előfordulási variációt.

4. Képelemzés:

4.1 Jelenlegi képelemző szoftver:

A képelemzést feltétele egy Microsoft HD web kamera, így kevésbé torzulnak a színek különböző fényviszonyok között. Kezdetben egy normál olcsó web kamerával próbálkoztunk azonban bármilyen apró fény hatására nagy különbségek voltak a színek között. A színfelismerésben új algoritmusokat kellett létrehozni, mert az új web kamerával sem sikerült megoldani azokat az eseteket, amikor a színek egybe olvadtak és ez által a program nem volt képes megkülönböztetni az egymástól eltérőeket, mivel az a működés közben az egyes alapszínek (RGB; Piros, Zöld, Kék) intenzitását figyeli.

A régi képelemző eljárás problémája az volt, hogy a különböző megvilágítások hatására az egyes színek egyes komponenseinek intenzitása mindig máshogy változott, nem volt egyenletes növekedés csökkenés mindhárom komponensnél, ezért nem lehetett felállítani olyan tartományokat, amelyekbe belefértek volna a színek komponensei anélkül, hogy az egyes tartományok ne értek volna egymásba.

Az új algoritmusok segítségével sikerült elérni, hogy akár kinti napfényben is lehetséges legyen a színek elemzése, természetesen az előzetes kalibrációk után. Az algoritmusokkal előállítunk három változót, amelyeknek kombinációjával már biztosabb lesz a színfelvétel. Az egyes változók kiszámítására az alábbi képleteket használok:

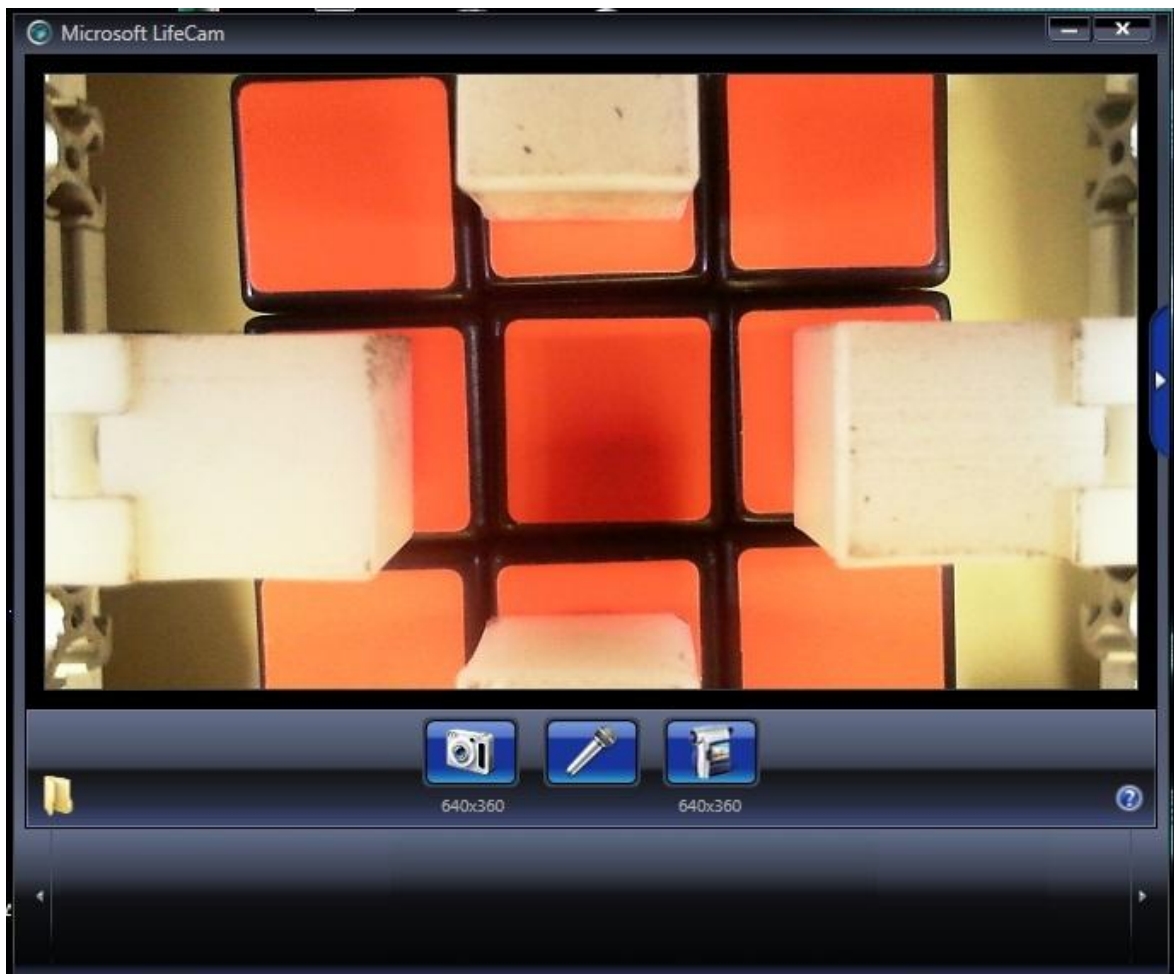
- 1. változó: $R \cdot G / B$
- 2. változó: $G \cdot B / R$
- 3. változó: $R \cdot B / G$

Ezen változók kombinációja olyan egyedi és bizonyos határokon belül egyenesen arányosak egymással, hogy különböző megvilágításoknál is közel azonos értékek között mozognak.

Ezeknek, a képleteknek köszönhetően a színek intenzitása már a kamerán át is egyenesen arányosan változott, függetlenül a megvilágítástól, így lehetővé tette a pontos szín és képelemzést.

Az általam írt program, amely egyben a vezérlő program is, a web kamera saját szoftverének képét figyeli, így a gyári programmal állítható a színek mélysége, kontrasztja, fényereje illetve a fókusza. Ezek segítségével a különböző fényviszonyokhoz kalibrálhatjuk a kamera képét. Az 9. ábrán látható a web kamera szoftvere. A program neve „Microsoft LifeCam” amely ingyenesen letölthető a Microsoft oldaláról:

<http://www.microsoft.com/hardware/en-us/p/lifecam-cinema/H5D-00001#support> .



9. ábra Microsoft LifeCam szoftver

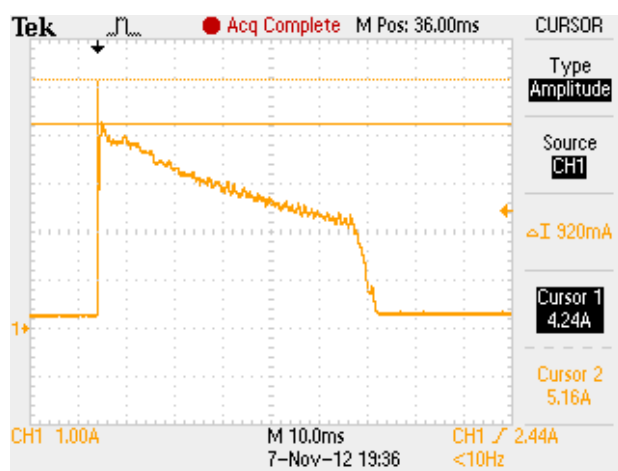
Továbbra is problémát jelent az erős, felülről érkező fény, bár a robotra szerelt erős LED fényforrás a legtöbb zavaró fényt elnyomja. Azonban ha például egy fénycső, világít a mennyezeten, akkor az a kocka fényes felületéről visszatükröződő fény rontja a képelemzés minőségét. Kísérletezések folynak polár szűrő alkalmazásával, remélhetőleg így a tükröződések elkerülhetőek lesznek. További lehetőség lett volna, hogy a kocka egyes matricáit lecseréljük matt (nem tükröződő) matricára így a további zavaró hatások is eltűntek volna, így azonban a robot mobilitása megszűnne, hiszen nem lett volna képes a megszokott, standard, gyári 3x3x3-as kocka kirakására.

Jelen esetben a robot képes bármilyen szabványos színekkel ellátott kocka kirakására.

5. Energia ellátás:

A robot tápjának üzemelésére szükség van hálózati feszültségre (50Hz, 220V). Egy 5V/3A kapcsoló üzemű tápegység előállítja a világításhoz, motorokhoz, valamint a vezérlő kontrollerhez szükséges tápfeszültséget.

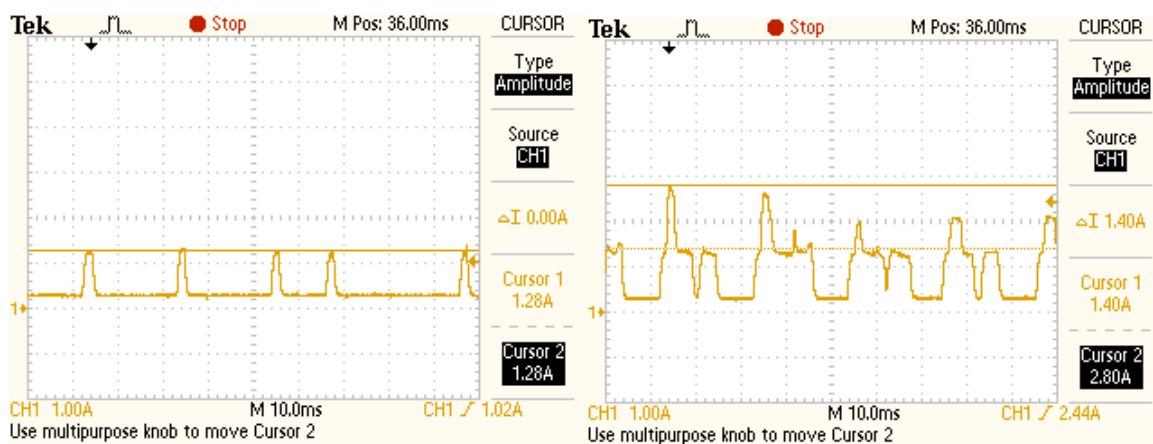
A motorok tápfeszültségre kapcsolásakor a vezérlő jel hiánya miatt hirtelen áramfelvétel tapasztalható, ezért az én megvalósításban a kontrollert, valamint a motorokat külön kapcsolóra kötöttem, de későbbiekben más megoldásra is gondoltam, például, a kontroller kapcsolja a tápfeszültséget a motorokra. A motorok nagy induló áramának hatására a kontrolleren a feszültség leesett így az újra indult. Az USB-SOROSPORT átalakító PC USB csatlakozójából kapja a tápfeszültséget, hiszen a PC és az átalakító között a kapcsolat megszakadt a motorok indulásának pillanatában. A motorok tápfeszültség alá helyezésekor történő hirtelen áramfelvétel mérési eredménye a 10. ábrán látható. Méréseket TDS 2002C oszcilloszkóp és Fluke i30 lakatfogó segítségével végeztem el.



10. ábra Motorok indulási áramfelvétele

Jól látható hogy a motorok indulásakor 5,16A a maximális áram felvétel. Ebből kifolyólag a mikrokontroller a változtatások elvégzése előtt újra indult.

További méréseket végeztünk, hogy meghatározzuk a kirakás közbeni áramfelvételt, amely nem állandó értékű, hiszen az áramfelvétel attól függ, hogy hány motor üzemel éppen. A 11. ábrán látható egy normál illetve szorítási működés közben végzett áramfelvétel mérés eredménye.



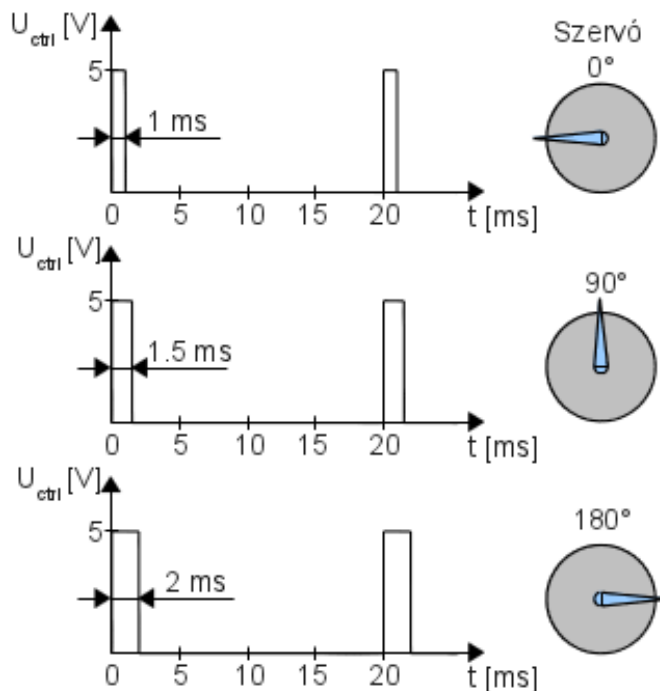
11. ábra Motor normál üzemi áramfelvétele: normál helyzetben (bal) illetve szorítás esetén (jobb)

6. Szervo hajtások:

A kocka forgatásáért felelős mechanikai mozgatót szervóhajtások biztosítják. Azért erre esett a választás, mert ehhez a feladathoz a pontosságuk is megfelel valamint könnyű a vezérlésük. Bekötésük egyszerű, hiszen mindössze három vezetékkel csatlakozik a vezérlő panelhez, így a vezetékezés sokkal egyszerűbb.

6.1 Vezérlés:

A vezérlést nagyban leegyszerűsíti, hogy nem kell a visszacsatolással, valamint a motor meghajtással foglalkozni, mert a választott szervó-hajtás magában foglalja ezeket. A szervó meghajtó vezérléshez szükséges jelalakok a 12. ábrán láthatóak.



12. ábra Szervo hajtások vezérlése [4]

Látható, hogy a vezérlés valójában impulzus szélesség moduláción (PWM) alapul. Melynek periódus ideje 20ms a kitöltési tényezője pedig 5% - 10% között változik.

6.2 Belső felépítés:

A servók két csoportra oszthatóak:

- Digitális servók
- Analóg servók

Külső vezérlést tekintve a két típus teljesen azonos, mindössze a belső felépítésben térnek el egymástól.

Digitális servók:

Digitális servók visszacsatolása is digitális jel, ebből következően a bejövő digitális jelet nem igényel további átalakítást a feldolgozás előtt. Ezen servók pontossága sokkal nagyobb, mint az analóg visszacsatolással rendelkező servók esetében.

Analóg servók:

A servóban található egy digitál/analóg átalakító, amely a bementre érkező PWM jelet átalakítja, feszültséggé majd ezt összehasonlítva a potenciométerből érkező visszacsatolt jellel szabályozza a motor meghajtásáért felelős hídkapcsolást. A 13. ábra bemutatja az analóg servo sematikus belső felépítését.

Bekötés:

- 1- Vezérlő jel (fehér)
- 2- Pozitív tápfeszültség (piros)
- 3- GND (fekete)

6.3 Szervók általános paramétereit:

- Tömeg: g
- Méret: mm
- Feszültség: V
- Sebesség sec/degrees
- Nyomaték: kg/cm
- Használati hőmérséklet

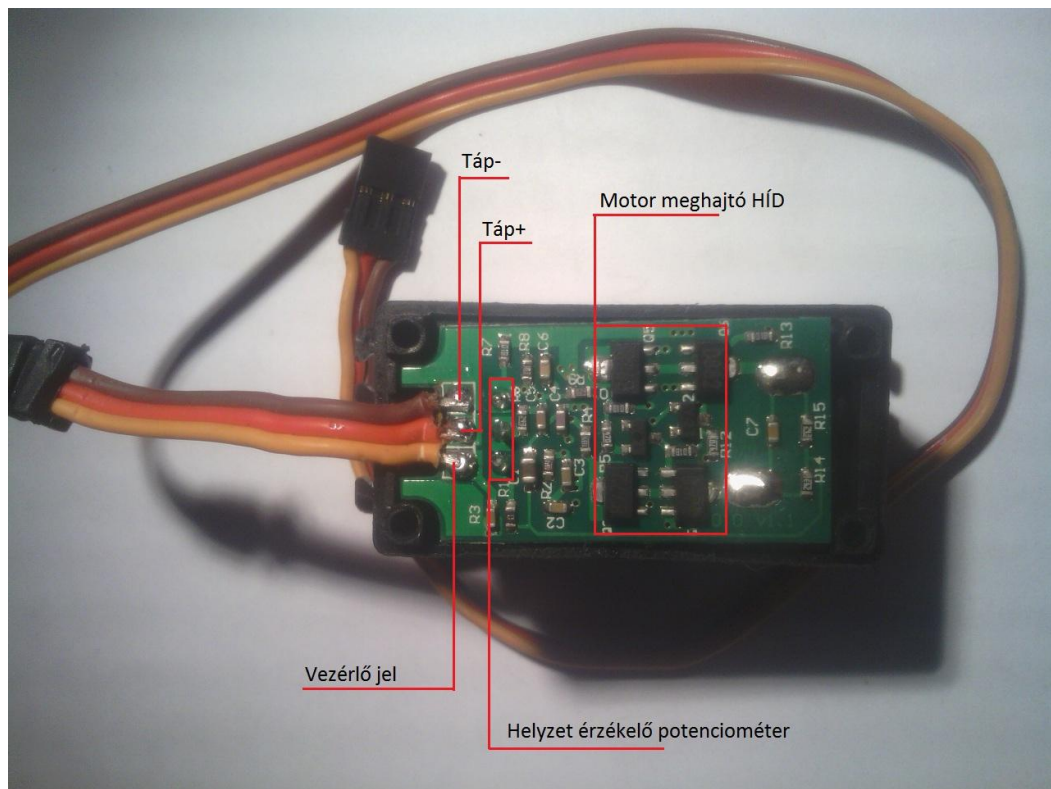
6.4 Általam használt szervó hajtás:

A roboton TowerPro cég által gyártott szervót találhatók melynek típusa: SG5010. Ezen szervó ár érték aránya elég jó. Ennek a típusnak továbbá előnye, hogy a kihajtásnál a fogaskerék mindkét oldalt golyóscsapággal van ellátva, így sokrétűen használható, hiszen képes tengelyén terhet tartani ellentétben a csapággal, nem rendelkezőkkel. További előnye, hogy a súrlódás miatt veszteség is sokkal kisebb.



15. ábra Általam használt szervó mechanikai felépítése (áttétel) [7]

Az általam használt szervóról nem találtam kapcsolási rajzot és a kapcsolás visszafejtése nehézkes, mert a nyomtatott áramköri lemez többrétegű. Levélben kértem a gyártó cégtől a kapcsolást, levelemre azonban még nem válaszolta.



16. ábra Szervo meghajtó panel

A meghajtó panelon nem látszik a vezérlő IC.

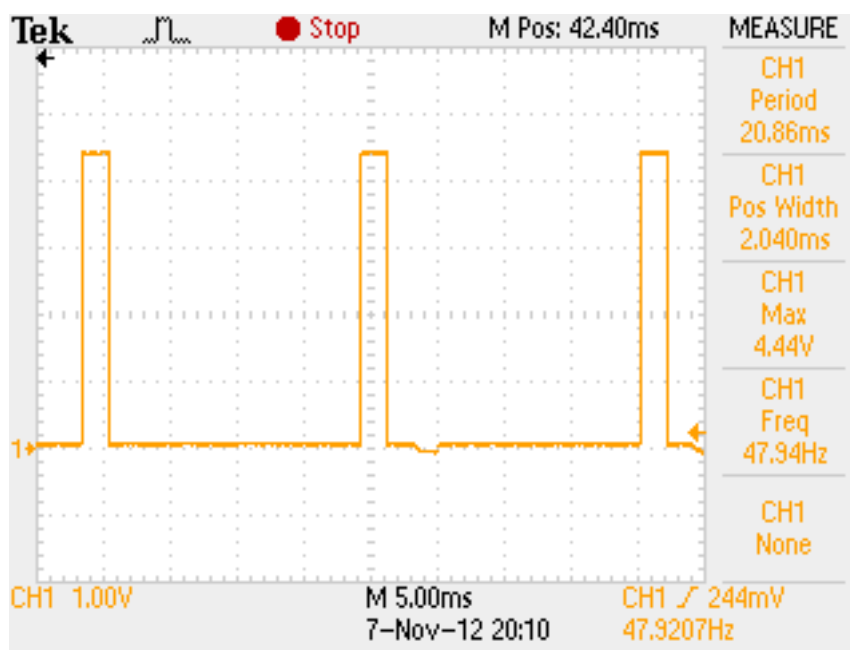
Általam használt szervo hajtás paramétereit:

- Tömeg: 42 gramm
- Méret: 40mmx20mmx36,5mm
- Feszültség: 4. 8-6.0V
- Sebesség: (4.8V): 0.20sec/60 degrees
- (6.0V): 0.16sec/60 degrees
- Nyomaték: (4.8V):(5.2 kg/cm)
- (6.0V):(6.5 kg/cm)
- Használati hőmérséklet: -20-tól 60 fokig.

7. Mikrokontrolleres vezérlő panel:

7.1 Program:

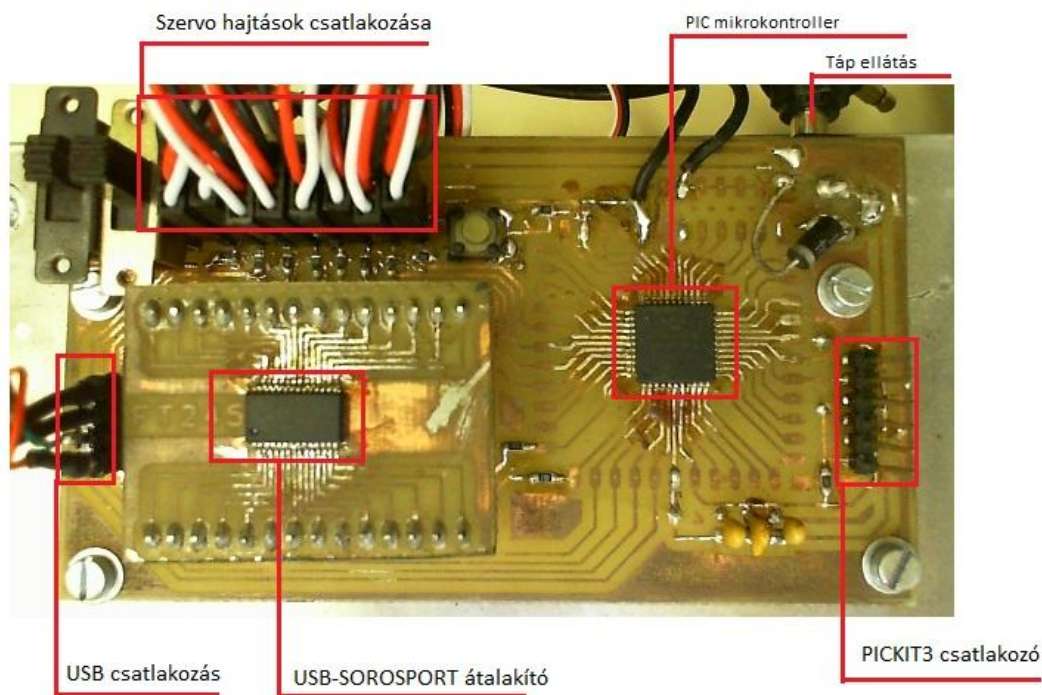
A mikrokontrolleres vezérlő panelon egy a Microchip által gyártott PIC18F45k20 található, amely az FT232rl átalakítón keresztül kommunikál a PC-vel, erre az EUSART perifériát használtam fel mivel így a szoftveres soros kommunikációval szemben leveszi a terhet a kontrollerről. Az órajelet belső oszcillátorról állítja elő (16MHz) majd ezt egy PLL áramkör négyszeresére szorozza, így a controller 64MHz-es órajelet frekvenciával működik. Erre azért volt szükség, mert a szervo hajtások vezérléséhez szükséges PWM jelet nem a beépített perifériával állítja elő, hanem Timer2 időzítő áramkör segítségével egy szoftveres impulzus szélesség modulációt végez. Így a felbontó képesség megnövekedett, így a szervo pontosabb vezérlése vált lehetővé (további pontosítást érhetek el precíziós kvarckristály használatával). A PIC program fejlesztése a Microchip által ingyenesen elérhető MPLAB 8.86 fejlesztő környezettel történt. A 17. ábrán példaként látható egy, a controller által előállított PWM jel.



17. ábra PWM vezérlő jel

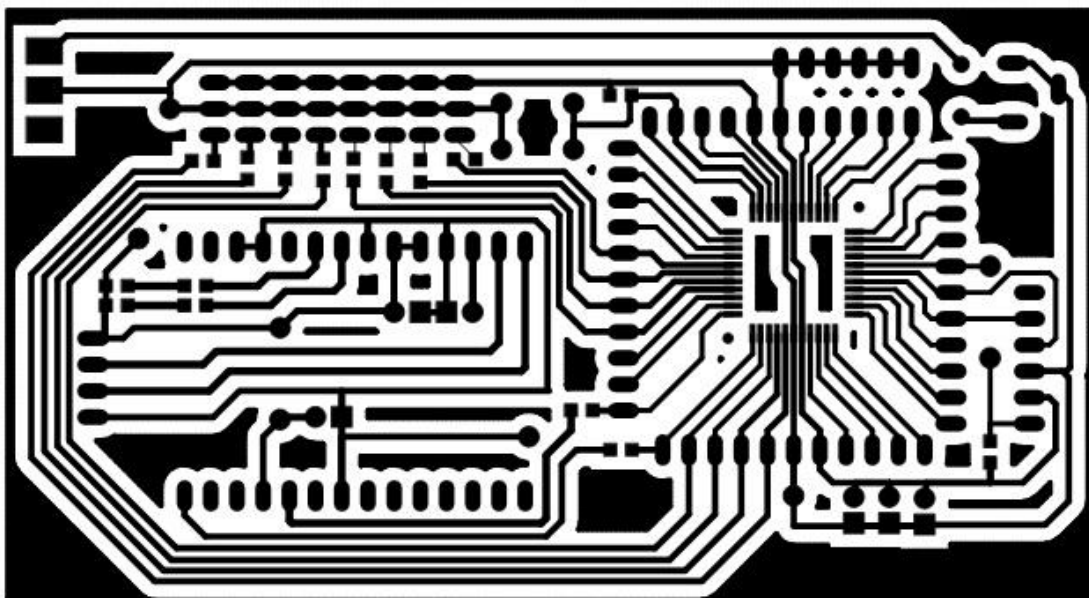
7.2 Vezérlő panel készítése:

A 18. ábrán látható az általam tervezett és készített panel, amely tartalmazza a vezérlő kontrollert valamint az USB-SOROSPORT átalakítót.



18. ábra Vezérlő panel

A panelt előzetesen Orcad Capture programmal megterveztem majd, Orcad Layout nyáktervező programmal elvégeztem a le huzalozást. A 19. ábrán látható az elkészült nyákterv.



19. ábra Orcad Layout programmal készített nyákterv

A panel tervezésekor és megvalósításakor SMD tokozású alkatrészeket használtam. Ennek oka az volt, hogy a panel méretét nagymértékben

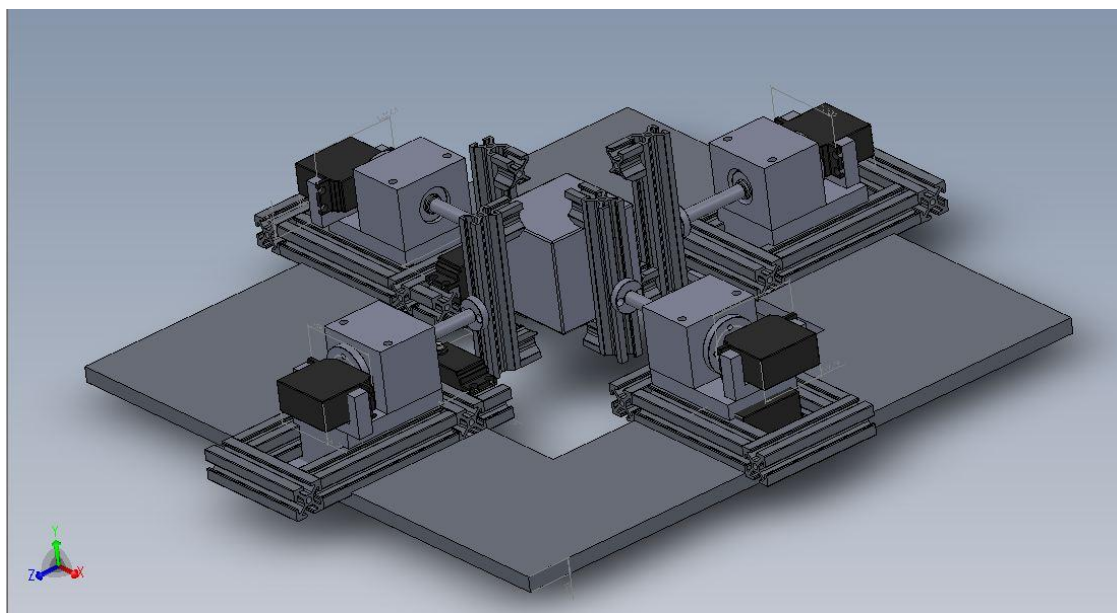
csökkenteni tudjam. A PIC mikrokontroller felületszerelt (TQFP) tokozásban olcsóbb, mint a furatszerelt (DIP) tokozásban, valamint az FT232RL USB-SOROSPORT átalakító chip nem kapható már csak SSOP28 tokozásban. A 18. ábrán látható a vezérlő panel elrendezése.

A nyomtatott áramkor készítését „fotózás” eljárással végeztem.

8. Mechanikai megvalósítás:

8.1 3D tervek elkészítése:

A terveket Work tervező programmal készítettem. Az egyes építő elemek megrajzoltam, majd ezek segítségével megépítettem a robotot. A tervezés elsődleges célja a robot alkatrészeinek megfelelő méretezése továbbá a szervo hajtások legoptimálisabb elhelyezésének megválasztása. A megvalósításnál sajnos nem tudtam a 3D-s terveket teljes mértékben követni, a könnyebb mechanikai megmunkálás miatt.



20. ábra Solid Work programmal készített 3D terv

8.2 Felhasznált anyagok:

A robot alapkeretét, valamint a csúszó pófák keretét továbbá a web kamera és a hozzá tartozó világítást az Item vállalat által gyártott szabványos 20X20-as alumínium idomokból készítettem el. Ezekhez beszerezhető szabvány összeépítő elemek, amelyek nagyban megkönnyítik az összeszerelést.

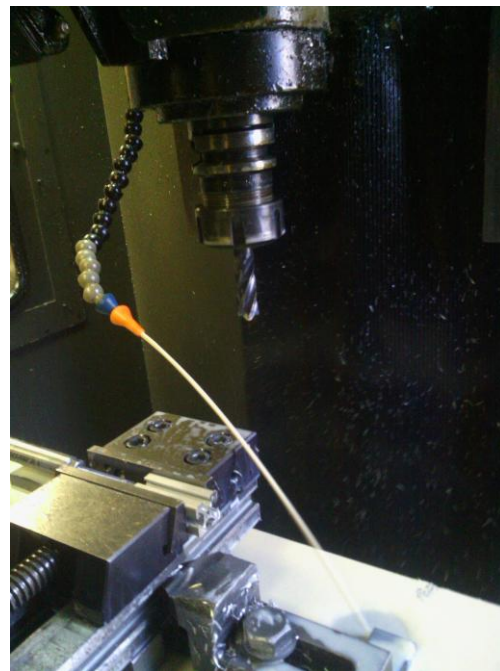


21. ábra Item 20X20 profil [8]

Az alap keretre került egy 10mm vastag alumínium alaplap, mely precíz megmunkálásnak köszönhetően rendkívül pontos illesztés valósít meg a csúszó keretek számára. Professzionális felsőmaró gép alkalmazásával az egyes karok közötti szögek pontosak, így minimális a köztük levő excentricitás.



22. ábra Csapágyház valamint csúszó lapok megmunkálása



23. ábra Alaplap valamint csúszó keret távtartó megmunkálása

A csúszó keret távtartóinak megmunkálása Brother TC229 típusú függőleges kismegmunkálón történt.

Az alaplap, csapágyház valamint a csúszó lapok megmunkálása 5 tengelyes Oerlikon Taurus függőleges CNC megmunkáló központon végeztük el.

9. Értékelés:

A robot tervezésénél kompromisszumot kellett kötni, hiszen az általam kiválasztott szervó-hajtások nem képesek teljes (360 °-os) forgatásokra, ezért a kezdetektől fogva tisztában voltam azzal, hogy a robot nem fog időben és a forgatások számában világrekordot dönteni.

Az általam készített robot képes egy tetszőleges megkevert szabványos 3x3x3-as Rubik-kocka kirakására. A kirakási idő átlagosan 4 perc amelyhez átlagosan 100 forgatás tartozik . Azonban a kirakási idő valamint a forgatások száma nagyban eltérhet egyes keverési kombinációknál, hiszen legjobb ideje 2 perc 15 másodperc és 75 forgatás.

Van robot, amely Technic legoból építve NXT vezérlővel felszerelve, valamint egy okos telefon segítségével, képes 5,35 másodperc alatt kirakni a kockát. Ez megtekinthető a legtöbb videó megosztó oldalon. Található sokféle megoldású robot, a legtöbb robot lego építő elemekből készül, amellyel nem érhető el akkora pontosság valamint a legtöbb kevés fogó karral rendelkezik így a forgatások elég, nehézkesek, lassúak. A világhálót böngészve azt hiszem, elmondhatom, hogy a robot benne van az első 10 leggyorsabb között a világon, azonban azok a robotok mind valamilyen rendkívül kevés forgatási számú (20 fogatás) programokat használnak. Hazai kitekintés során

információim szerint 1 db robot készült még az országban, ami viszont elég lassú 10 perc alatt képes kirakni a kockát. Ez a robot megoldás mechanikai korlátokkal rendelkezik. A készítője elmondása szerint a cél a precíz megfogás volt, amelyet egy satu megfogásához hasonló megoldással ér el, azonban ennek alkalmazásával az egyes forgatások sokáig tartanak, mert e megfogásért felelős mozgató egy léptetőmotor szolgáltatja, amely egy menetes szálal hajtva húzza össze pofákat. Így a szorítás rendkívül masszív. A kirakás ideje 10-12 perc között alakul.

Bizonyított, hogy elméletileg bármely keverés után legfeljebb 20 tekerésből kirakható, összetettsége miatt összesen 43 252 003 274 489 856 000-féle azaz kb. $4,3 \cdot 10^{19}$ –féle eltérő állás hozható létre.

http://www.technet.hu/hir/20100811/megfejtettek_a_rubik_kocka_rejtelyet/

10. További fejlesztési lehetőségek:

10.1 Vezérlő program fejlesztése:

További fejlesztések folynak egy Raspberry Pi nevezetű (bank kártya méretű számítógép) fejlesztő panellel, amelyet egy nonprofit cég gyárt és a következő linken érhető el <http://www.raspberrypi.org>. Ez egy ARM processzort tartalmaz, ezzel a számítógép szerepe kiváltható lenne, hiszen a 700MHz órajellel futó processzor képes Linux alapú operációs rendszer futtatására. Egyes segéd programok használatával futtatható lenne rajta a vezérlő program, így egy kisméretű képernyővel összekapcsolva, mindez elhelyezhető lenne a robot platformon, így kompakt lenne, nem kellene külön számítógép a használatához. A következő képen (24. ábra) látható módon az elképzelt felépítés. A processzort tartalmazó központi egység már beszerzésre került, alkalmazására megkezdődtek a kísérletek.

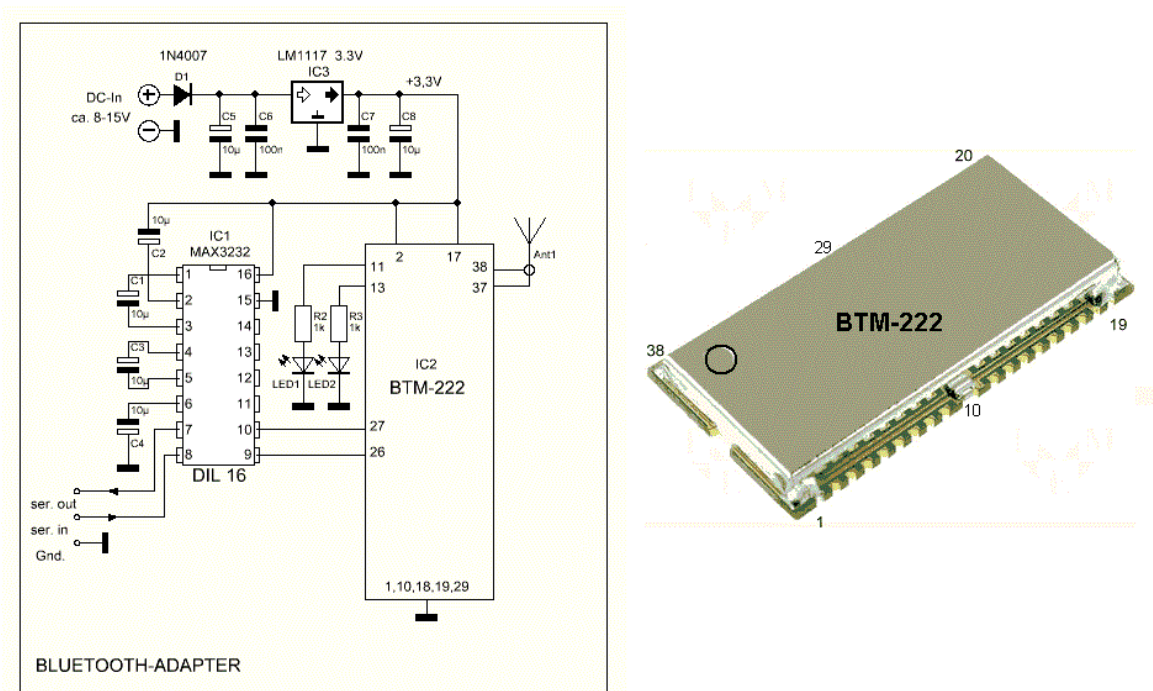


24. ábra Raspberry Pi fejlesztő panel [9]

A panelon található egy kisméretű VGA monitor, amely elegendő lenne a program futtatásához, valamint egy vezeték nélküli billentyűzetet valamint egeret, amely lehetővé tennék az irányítást.

10.2 Kommunikáció további fejlesztése:

Az USB-SEROS átalakító helyére továbbá beszerelhető egy BLUETOOTH-SEROS modul, amellyel vezeték nélküli kapcsolat alakítható ki a PC és ROBOT között. Ezen átalakító a BTM222. Távoli irányítás hatótávolsága így illesztett antennával akár 100m-re is növekedhet.



25. ábra BTM222 modul és lehetséges alkalmazási kapcsolása[10]

10.3 Képelemzés további fejlesztései:

Terveim között szerepel egy olyan program készítése, amely elindítás után bekalibrálja magának a megfelelő értékekre a web kamera szoftverét, ezzel minden egyes kirakás előtt, ha változott a megvilágítás vagy bármilyen zavaró hatás lépett fel, megpróbálja kikompenzálni. Így a kalibrálási lépés elkerülhető lenne, és nem léphetne fel olyan hiba, hogy valamely szín tulajdonságai nem felelnek meg a képelemzésben ledeklarált tulajdonságokkal.

További próbálkozások polár szűrő alkalmazásával történhetnek, így kiküszöbölhető lenne a fentről érkező erős fény tükröződése a kocka fényes felületén.

10.4 Algoritmusfejlesztés:

Általam készített vezérlő szoftver fejlesztése között tervben vannak a „profí” algoritmusok. Amelyek a Fridrich metódus teljes betáplálásával teljessé válnak.

10.5 Tápellátás további fejlesztései:

További fejlesztésként a megvilágítás intenzitását szeretnénk növelni, valamint külön tápegységgel ellátni, mert a motorok nem egyenletes teljesítmény felvétele miatt a megvilágítás fényereje változik a motorok

üzemeltetése közben.

10.6 Végleges külső kialakítás:

A fejlesztési munka végének fontos feladata a robot megjelenésének végleges kialakítás, valamint a szoftverbe beépülő 3D-s kockanézet létrehozása.

11. Köszönetnyilvánítás:

Elméleti alapokban valamint a szerkezethez szükséges egyes elemek beszerzésében konzulensem Major László docens sokat segített. A mechanikai megvalósításban édesapám Takács Károly CNC forgácsoló segédkezett, aki professzionális megmunkáló gépek segítségével precízen ki tudta alakítani a robot egyes elemeit, valamint jelentős szerepet vállalt az egyes alapanyagok beszerzésében.

12. Felhasznált irodalmak:

12.1 Rubik-kocka kirakási algoritmusok forrásai:

- <http://www.rubikkocka.hu>
- <http://boca.bee.pl>

12.2 Vezérlő kontroller programfejlesztésben felhasznált irodalom:

- **Kónya László- Kopják József: PIC Mikrovezérlők alkalmazástechnikája, PIC programozás C nyelven** (harmadik, bővített kiadás) ChipCad Kft. Budapest 2009

12.3 Szervo hajtások fejezetben felhasznált források:

- <http://www.hooked-on-rc-airplanes.com/servo-tutorial.html>
- http://kecskemetircmodell.blog.hu/2011/03/22/hogyan_mukodnek_a_szervok
- http://www.rcmodell.com/shop/product_info.php/cPath/67_29/products_id/122

12.4 Képi források:

- [1]: <http://www.ftdichip.com/Corporate/FTPress.htm>
- [2]: <http://boca.bee.pl/cat.php?l=en&cat=th&m=fridrich&ch=pll>
- [3], [4]: <http://www.ivanov.eu/?q=node/1399>
- [5]: <http://www.digitalnemesi.com/info/docs/rcservo/>
- [6]: <http://www.seattlerobotics.org/encoder/200009/S3003C.html>
- [7]: <http://www.hobbypartz.com/kahaoubmo14.html>
- [8]: http://www.hellopro.de/ITEM_INDUSTRIETECHNIK_UND_MASCHINENBAU_GMBH_-4566-noprofil-2011241-38198-0-1-1-fr-societe.html
- [9]: <http://www.geek.com/articles/chips/raspberry-pi-turned-into-a-portable-workstation-20120719/>
- [10]: <http://www.kh-gps.de/btm222.htm>
- A 3.2 fejezetben használt képi illusztrációk:
<http://boca.bee.pl/cat.php?l=en&cat=th&m=fridrich&ch=oll>