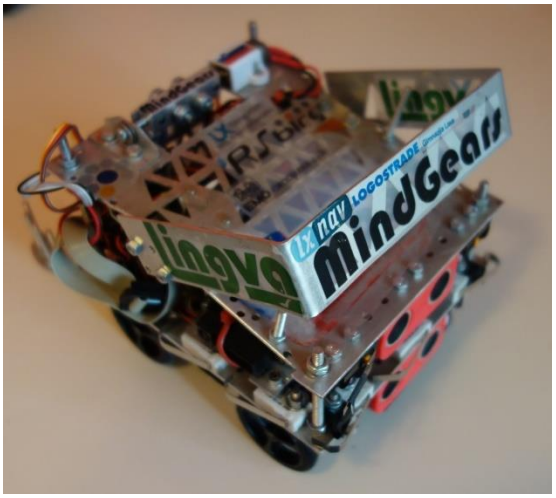
Mestna občina Celje

Komisija Mladi za Celje



Razvoj mobilnega robota na podlagi AVR mikrokontrolerjev

RAZISKOVALNA NALOGA

AVTORJI
Rok Krumpak
Jan Časl
Janez Turnšek

MENTORJA
Karmen Kotnik, univ. dipl. inž.
Tomislav Viher, univ. dipl. org.

Celje, marec 2016

Šolski center Celje

Gimnazija Lava

Razvoj mobilnega robota na podlagi AVR mikrokontrolerjev

RAZISKOVALNA NALOGA

Avtorji:

Jan ČASL, 4.e

Rok KRUMPAK, 4.e

Janez TURNŠEK, 4.a

Mentorja:

Karmen KOTNIK

Tomislav VIHER

Mestna občina Celje, Mladi za Celje

Celje, 2016

Kazalo vsebine

Povzetek **5**

1 Uvod **6**

2 Predstavitev tekmovanja ter namembnost robota **7**

2.1 Namembnost robota 8

3 Teoretični del **8**

4 Potek razvoja strojne in programske opreme **11**

4.1 Razvoj robota 11

4.1.1 Prva stopnja razvoja robota 11

4.1.2 Druga stopnja razvoja robota in prvi pravi prototip 13

4.1.3 Tretja stopnja razvoja robota in njegova končna različica 15

4.2 Razvoj programa za robota 22

4.2.1 Razdelitev programa na podprograme 22

4.2.2 Testiranje 23

4.2.3 Branje podatkov s senzorjev 23

4.2.4 Upravljanje motorjev 25

5 Rezultati in razprava **25**

6 Zaključek **27**

7 Viri **28**

Kazalo slik

Slika 1: Grafični prikaz delovanja pulzno širinske modulacije 9
Slika 2: Enostaven program za utripanje LED diode 10
Slika 3: Plošča tiskanega vezja narisana v programu EAGLE 13
Slika 4: Izrezkana plošča narejena po zgornjem načrtu 13
Slika 5: Prvi pravi prototip, s katerim smo se udeležili državnega tekmovanja 15
Slika 6: Plošča tiskanega vezja za končno različico robota narisana v programu Altium 16
Slika 7: Računalniški 3D prikaz plošče v programu Altium 16
Slika 8: Plošča tiskanega vezja narejena po naših načrtih narisanih v programu Altium 16
Slika 9: Računalniški render 3D modela robota 17
Slika 10: Računalniški render 3D model robota, ki pobira žogo 18
Slika 11: Slika robota z vidno razprto roko in zadrževalnimi vratci za žoge 18
Slika 12: Slika tiskanega vezja za barvne senzorje 19
Slika 13: Plošča vezja narisana v programu Altium 19
Slika 14: Računalniški 3D model tiskanega vezja v programu Altium 19
Slika 15: Programator z priključenim USB kablom in kablom za programiranje robota 20
Slika 16: Slika robota z vidnim imenom naše skupine MindGears 27
Slika 17: Render 3D modela robota 27

Povzetek

V nalogi smo predstavili nastajanje mobilnega robota ter programa, s katerim smo tekmovali na svetovnem prvenstvu v programiranju mobilnih robotov RoboCupJunior 2015 na Kitajskem. Opisali smo nastanek robota s tehnološkega, mehanskega in tudi elektrotehniškega in programerskega vidika, na koncu pa smo povzeli kaj smo se naučili, kaj bi lahko izboljšali in predstavili rezultate.

1 Uvod

Odkar se je na svetu pojavil misleči človek si je vedno želel olajšati življenje na tem planetu. Najprej si je iz kamna delal pestnjake, nato sekire in ostalo orodje. Večino tega, kar je človek kadarkoli ustvaril, je ustvaril za lajšanje življenja. Tako so v 20. stoletju odkrili elektriko in magnetizem, naredili prvi računalnik in še mnogo več.

Danes v rokah nosimo prenosljive naprave, ki nam pomagajo pri vsakodnevnih opravilih, doma pa imamo skoraj vsi hladilnike, ki sami vedo kakšno temperaturo morajo vzdrževati in kako to početi. Težnja po vedno večji prilagoditvi okolice našim potrebam nas je pripeljala do začetka dobe robotike.

Robot je naprava, ki je bila ustvarjena za določen namen in opravlja to funkcijo avtonomno ali pa s človekovo pomočjo. Ne glede na to ali je robot mehanska roka, ki vari avtomobilska ogrodja ali je to avtomatski sesalnik, ki sesa hišo, mora robot nekako vedeti, kako bo to počel in kaj je pravzaprav njegova naloga. Nujno torej potrebuje poleg mehanskih delov tudi nekakšne možgane, ti pa so v svetu računalništva, elektrotehike in mehanike pogosto predstavljeni s procesorji in mikrokontrolerji ali mikročipi. Vsi roboti imajo v sebi procesor, ki nadzira njihovo delovanje. Ti procesorji pa so se do danes v velikosti in hitrosti že tako izboljšali, da lahko tudi šibkejši procesorji, kot so mikropocesorji AVR, uspešno in zavidljivo dobro upravljajo manjšega robota. In prav to smo dokazovali, ko smo tekmovali z našim novim robotom na svetovnem tekmovanju v programiranju mobilnih robotov RoboCupJunior 2015 v mestu Hefei na Kitajskem. Naš robot je bil na tekmovanju odlikovan z nagrado Best Hardware Design v svoji kategoriji, kar nas je deloma tudi spodbudilo, da zdaj pišemo o raziskovalnem procesu med ustvarjanjem tega robota.

V nalogi bomo sproti postavili več delnih hipotez. S hipotezanimi in dobljenimi rezultati si bomo pomagali izbrati najbolj ustrezne komponente za sestavo našega robota in tudi najbolj primeren način pisanja določenih delov programa oziroma najbolj preprost način reševanja programskih in mehanskih ter logičnih problemov.

Glavna hipoteza v naši raziskovalni nalogi je: Z nekaj navdušenja, podporo mentorjev in veliko žrtvovanega prostega časa, lahko skupina gimnazijcev brez posebnih znanj na področju robotike, naredi in sprogramira robota, ki se lahko na svetovni ravni primerja z roboti ostalih skupin tehničnih srednjih šol.

Opisali bomo predvsem razvoj robota in predstavili različne načine reševanja problemov, na katere smo naleteli pri sestavi robota. Opisali bomo tudi, zakaj smo se odločili za končno rešitev in kako bi jo morda še lahko izboljšali.

Glavna metoda, ki smo jo uporabljali pri raziskovanju, je eksperimentalna metoda.

2 Predstavitev tekmovanja ter namembnost robota

Naš novi samogradni robot je bil ustvarjen za udeležbo na tekmovanju RoboCupJunior 2015 in sicer v kategoriji Rescue Follower Secodary. V tej kategoriji roboti niso omejeni s pravili tekmovanja temveč z nalogami, ki jih tekmovanje od robota zahteva. Tako je lahko robot velik največ 25 krat 25 centimetrov po širini in višini ter 30 centimetrov po dolžini, saj mora voziti skozi vratca teh velikosti. Je pa seveda zaželjeno, da ima robot čim manjše dimenzije, saj mu to omogoča lažje opravljanje nalog in večjo gibljivost, hrkati pa možnost manjših odstopanj od idealne lege robota, ki nastanejo zaradi nenatančnosti senzorjev in druge opreme. Naš robot je bil tako grajen z velikim poudarkom na kompaktnosti in velikosti končnega izdelka, ki pa ne sme vplivati na uspešnost robota in delovanje njegovih komponent.

Na tekmovanju prav tako ni omejeno število ali tehnologija senzorjev, v kolikor ne uporabljamo laserjev močnejših od 0,5 mW in komunikacije, s katero bi robotu lahko dajali dodatne podatke.

Na tekmovanju mora robot opravljati več nalog avtonomno, pri vsakem nenapredovanju pa se lahko predstavnik tekmovalne skupine odloči, da bo z napredovanjem poskusil ponovno in lahko robota ponovno zažene na zadnjem doseženem varnem kvadratu. Varne kvadrate ali tudi varne točke določi kapetan ekipe pred vožnjo in se po začetku teka ne smejo prestavljati. Robot ima za dokončanje vseh nalog 8 minut časa. Po tem času se robot vzame s polja in izklopi, točke pa se seštejejo.

Robot ima na svoji poti več različnih ovir, ki jih mora znati sam, ne glede na zaporedje, prečkati in nadaljevati z vožnjo po progi. Na skoraj celotni dolžini arene je narisana ali z izolacijskim trakom nalepljena črna črta debeline približno 2 centimetra, ki robota vodi po areni in povezuje razne prepreke, ki jih mora robot med sledenjem črte spremljati. Črta sama pa je lahko ravna, skrajno lomljena, pravokotno lomljena, zavita, ima pa lahko tudi prekinitve, ki se na tekmovanju štejejo kot prepreke. Nekatere druge prepreke, s katerimi mora robot obračunati, so hitrostne ovire, po areni posuti zobotrebci, ovire, ki se jim mora izogniti (opeke in težke plastenke raznih velikosti ter oblik) ter klančine do 25° naklona. Hitrostne ovire so na tla arene prilepljene bele okrogle palčke s premerom do 1 centimetra, ki so vedno pravokotne glede na črno črto, ki je pod njimi. Lahko so postavljene v skupinah po več skupaj. Robot dobi za vsako prepreko ali nalogo, ki jo uspešno opravi, vnaprej določeno število točk.

Ena izmed nalog, na katere mora biti robot pozoren, so tudi križišča, kjer se robot odloča po pravilu zelene barve. Križišče je pravokotno sekanje treh ali štirih črt v eni točki, na katerem je lahko nalepka zelene barve velikosti 2,5 x 2,5 cm. Nalepka je nalepljena na levi ali desni strani črte, kateri robot sledi, v smeri, iz katere naj bi pripeljal robot. Pri vsaki zeleni nalepki v križišču se mora robot odločiti za zavoj v smeri nalepke. Če je križišče brez zelene nalepke, je robot dolžan peljati naravnost.

Končna in glavna naloga tekmovanja je na vsaki areni enaka. Po prevoženem labirintu se robot pripelje do vhoda v zadnjo sobo na areni, ki je na tleh označena s širokim visoko

odbojnim srebrnim aluminijastim trakom. Soba ima dolžino 120 centimetrov in širino 90 centimetrov, vhodna vrata v sobo pa so lahko v kateremkoli kotu. V sobi se popolnoma naključno nahaja od 4 do 5 lahkih plastičnih žogic ovitih v aluminijasto folijo premera okoli 5 centimetrov. Robot v tem delu naloge simulira dogajanje na prizorišču nesreče, žogice pa predstavljajo žrtve, ki jih mora robot avtonomno rešiti in jih prenesti v varno cono v tej sobi. Varna cona je košarica v obliki prizme z osnovno ploskvijo pravokotnega trikotnika in višino 6 centimetrov, ki je lahko postavljena v poljuben kot v sobi.

2.1 Namembnost robota

Iz grobega opisa pravil vidimo, da potrebujemo robota, ki bo dovolj majhen, da se pri obhodu ovire ne bo zataknil, ki bo dovolj močan, da bo lahko preplezal hitrostne ovire, tudi ko mu bodo postavljene na klancu in robota, ki bo imel dovolj nizko težišče, da se pri vzpenjanju po klancu navzgor ne bo prevrnil. Robot bo potreboval optična tipala za sledenje črni črti, ki ga vodi skozi labirint in tudi barvne senzorje za zaznavo barve na križiščih. Robot mora biti čim hitrejši, da ostane dovolj časa še za morebitna ponavljanja delov proge na areni. Za orientacijo v prostoru bo potreboval še senzorje za merjenje razdalje, zelo dobrodošli so tudi razni pospeškomeri in giroskopi, s katerimi bomo ugotovili, kdaj se robot nahaja na klancu ter kompas za natančno vrtenje robota v pravo smer. Robot mora seveda imeti nameščenih tudi kar veliko stikal, da lahko določi, če se je v kakšno stvar na progi zataknil ali pa za zagotovitev enakomerne poravnave ob steni.

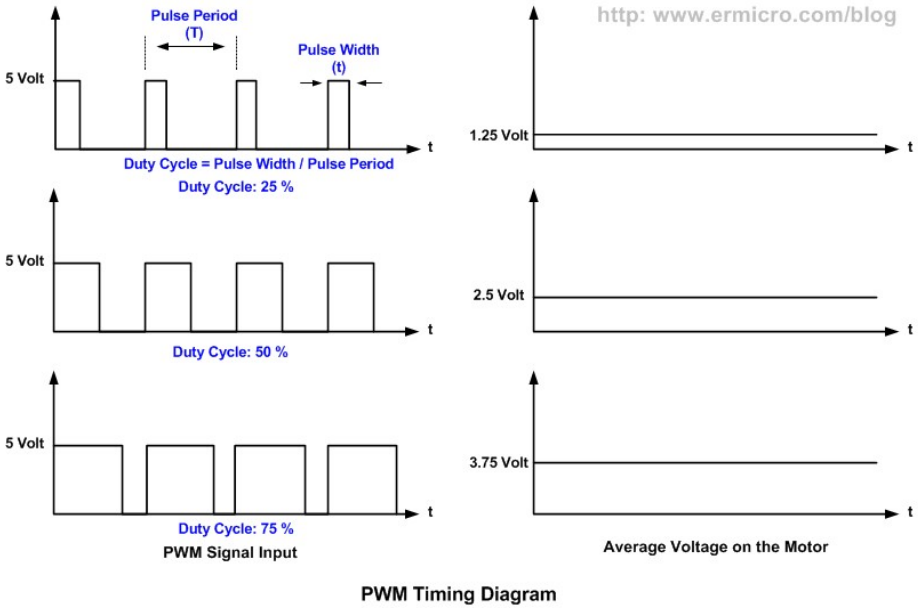
3 Teoretični del

Tehnično gledano je naš robot zelo usmerjeno razvit za potrebe tekmovanja, na katerem tekmuje. Vse posamezne komponente, ki smo jih uporabljali, razen naših tiskanih vezji, so v svetu mehanike, elektronike in elektrotehnike že dobro raziskane. Tako smo si pri načrtovanju in ustvarjanju robota lahko veliko pomagali z že opravljenimi raziskavami predvsem pa s podatkovnimi listi (ang. datasheet) posameznih kupljenih komponent, ki nam povejo veliko o načinu programiranja, komunikacije in implementacije v sistem. V njih so navedeni vsi pomembni podatki, ki jih nekdo potrebuje za uspešno uporabo. Podatkovni listi so večinoma na straneh proizvajalcev, v prilogi pa smo priložili podatkovni list procesorja ATmega2560, ki smo ga uporabili, kot glavni mirkoprocesor.

Mikropocrocesor ATmega2560 spada v družino 8-bitnih mikrokontrolerjev CMOS, ki so zasnovani na AVR izboljšani RISC arhitekturi. Navadno teče ta mikrokontroler s frekvenco 16 MHz. Ima 32 splošno uporabnih registrov, 256 kB flash pomnilnika, 4 kB EEPROM pomnilnika, 8 kB SRAM pomnilnika in 86 splošnonamenskih vhodov/izhodov. Čeprav veliko mikroprocesorjev ni optimiziranih za programski jezik C, je podjetje Atmel kupilo in izboljšalo posebno arhitekturo AVR, ki jim omogoča zelo hitro in preprosto programiranje mikroprocesorjev ATmega, ti pa dosegajo boljše rezultate in so pri tem zelo varčni.

Pri svojem delu smo veliko uporabljali tudi signal PWM, ki ga bomo za lažje razumevanje naloge v nadaljevanju razložili. V mikroprocesor je vgrajen poseben pretvornik ADC (analog to digital converter), ki oceni velikost analognega signala (napetost) in ga primerja z

ozemljitveno napetostjo (0 V). Včasih pa mora mikroprocesor oddajati tudi analogni signal. Sposoben mora biti torej virtualno prikazati katerokoli napetost od 0 V do 5 V. To je doseženo s PWM ali pulzno širinsko moduliranim signalom. Signal zelo hitro preklaplja med polno napetostjo, v našem primeru 5 V, in ozemljeno napetostjo - 0 V, tako da s tem simulira določeno napetost med mejnima vrednostma. PWM signal je ustvarjen iz zelo hitrih pulzov, ki se ponavljajo periodično na vsakih 256 ciklov notranje ure s frekvenco 16 MHz. Dolžina pulza je odvisna od številke, ki jo vpišemo kot vrednost PWM signala med programiranjem in je lahko celo število od 0 do 255. Vrednost, ki jo vpišemo kot vrednost PWM signala, se odrazi kot število ciklov polne napetosti glede na število vseh ciklov. PWM signal lahko tako simulira analogen signal, ki lahko ima vrednosti od 0 V do 5 V.



Slika 1: Grafični prikaz delovanja pulzno širinske modulacije

Pri komunikaciji s svetolobnim senzorjem smo uporabljali protokol I2C, ki deluje na principu master - slave (gospodar - suženj) na dveh vodilih. To sta vodili SCL in SDA. Vodilo SCL prenaša takt glede na katerega poteka pogovor, vodilo SDA pa je vodilo, ki prenaša podatke. Pravzaprav se podatki prenašajo z različnimi kombinacijami stanja signala na dveh vodilih. Na ti dve vodili je lahko priključenih vsega skupaj 128 ali pa 1024 naprav, saj ima vsaka naprava svoj naslov. Naslavljanje je lahko torej 7 ali pa 10-bitno.

Začetnik komunikacijo začne s startnim bitom, nato po vodilu pošlje naslov, ki bi ga rad naslovil, preden pa pošlje končni bit pošlje predzadnji bit, ki prejemniku pove ali bi rad začetnik pogovora bral podatke ali pa bi jih rad komu poslal. Če je naprava s pravkar poslanim naslovom priključena na vodilo, se odzove s posebnim potrditvenim bitom. Začetnik pogovora nato nadaljuje komunikacijo. Vsaka komunikacija pa se začne s startnim in konča s končnim bitom. Mikroprocesor ATmega2560 uporablja za naslavljanje 7-bitni naslov.

Podobno kot komunikacija I2C tudi serijska komunikacija poteka po dveh vodilih. Neposredno povezuje dva čipa z linijama Tx (ang. transceiver, slo. oddajnik) in Rx (ang.

receiver, slo. prejemnik). Vodilo, ki je pri prvem čipu vezano na izhod Tx je pri drugem čipu vezano na vhod Rx in obratno. Tako se omogoči prenos podatkov tudi če oba čipa hkrati pošljeta podatke. Pri prenosu podatkov prek te komunikacije se podatki, tudi če naslovljeni mikročip v tistem trenutku ne posluša, prenesejo v poseben pomnilnik, ki ga nato naslovljeni mikročip prebere takoj ko zazna, da se nekaj nahaja v njegovem pomnilniku. To komunikacijo smo uporabili pri povezovanju dveh mikroprocesorjev ATmega2560, saj je najbolj preprosta in je zelo hitra.

Naš mikroprocesor smo morali tudi sprogramirati. Programirali smo v programskem okolju Arduino IDE, ki podpira pisanje programske kode v C in C++ s posebnimi pravili za organizacijo kode. Arduino IDE je priročen, ker je v njem napisanih že ogromno knjižnic, to pa pospeši hitrost programiranja. Arduino IDE mora pred nalaganjem programa poskrbeti, da je program vsaj sintaktično pravilen, zato ga pred prevajanjem preveri. Nato ga prevede in preko USB vodila se podatki v obliki heksadecimalnih vrednosti prenesejo na programator, ki je v primeru plošče Arduino na tiskanem vezju. Programator je že vnaprej sprogramiran mikrokontroler, ki služi samo programiranju večjega in močnejšega, torej glavnega mikrokontrolerja. Ta čip nato sprogramira glavni mikroprocesor, ki se ob koncu programiranja znova zažene in požene se pravkar naloženi program.

Program napisan v Arduino IDE se imenuje »sketch« ali skica po slovensko. Tipičen program sestoji iz dveh delov. Prvi del ali setup() vsebuje kodo, ki se izvede le na začetku programa in inicializira nastavitve, ter drugi del ali loop(), ki se ponavlja dokler ima plošča napajanje. Primer enostavnega programa za utripanje LED diode je prikazan spodaj.

```
#define LED_PIN 13

void setup() {
      pinMode(LED_PIN, OUTPUT);    // Nastavi pin 13 za digitalno oddajanje
}

void loop() {
      digitalWrite(LED_PIN, HIGH);  // Prižgi LED diodo
      delay(1000);                  // Počakaj eno sekundo (1000 milisekund)
      digitalWrite(LED_PIN, LOW);   // Ugasni LED diodo
      delay(1000);                  // Počakaj eno sekundo (1000 milisekund)
}
```

Slika 2: Enostaven program za utripanje LED diode

Na vrhu vidimo v prvi vrstici definiran 13. "pin" oz. priključek, ki je vezan na LED diodo. Nato se začne inicializacijski del programa, ki se začne s stavkom void setup(). Sledi vrstica kode, ki mikroprocesorju pove, za kakšen namen bomo uporabljali ta "pin". V glavnem delu programa, ki se začne s stavkom void loop(), je nato LED dioda zaporedno vključena in nato izključena v osnovnem intervalu dveh sekund.

V zahtevnejših programih smo implementirali funkcije in metode, ki so kot nekakšni podprogrami in se izvedejo, ko jih v glavnem delu programa kličemo po imenu in jim podamo ustrezne parametre.

4 Potek razvoja strojne in programske opreme

V tem poglavju bomo opisali nastajanje robota, od samega začetka do konca. Najprej bomo predstavili razvoj strojne opreme, ali z drugimi besedami razvoj robota kot takega, nato pa si bomo ogledali nastajanje programa ter njegove komponente.

4.1 Razvoj robota

Nastajanje robota je potekalo v več stopnjah. Prva stopnja je bilo raziskovanje sveta mikroprocesorjev ter druge elektrotehniške opreme. To nam je dalo vpogled v to, kakšne možnosti imamo pri gradnji robota in kje so glavne ovire. Na tej stopnji smo začeli postopoma sestavljati vedno zahtevnejše robote, medtem ko smo se spoznavali z elektrotehniškim in programskim ozadjem delovanja komponent. Priučili smo se programskega jezika, ki ga zahteva Arduino IDE in napisali prve krmilne programe za robota.

Druga stopnja je bila potrjevanje koncepta. Elektrotehniško in programersko stopnjo zahtevnosti smo povišali na raven prvega tekmovalnega robota. Prvi pravi prototip so poganjale tri Li-ionske baterije ter naše prvo tiskano vezje, ki ga je upravljal mikroprocesor ATmega2560 na integrirani plošči Arduino Mega 2560. Ker smo uporabili ploščo Arduino, smo lahko robota programirali direktno preko USB povezave. Prvi prototip je v dokončani različici tekmoval na državnem tekmovanju maja 2015 v Mariboru, kjer pa žal, zaradi veliko napak na strojni in programski opremi, robot ni osvojil prvega mesta, kot smo upali.

Tretja stopnja je bila naša končna stopnja razvoja robota. Vse napake, ki smo jih naredili pri prejšnjih različicah robotov, smo si zapisali in jih v tej različici poskusili popraviti. Prav tako smo se odločili, da bomo namesto enega čipa vključno s celotno ploščo Arduino na naše tiskano vezje integrirali dva samostoječa mikroprocesorja ATmega2560 brez Arduino plošč, saj smo tako pridobili veliko število uporabnih izhodov in vhodov za senzorje in ostale nadzorne sisteme ne da bi izgubljali veliko prostora za nepotrebne Arduino plošče. S tem robotom smo lani poleti tekmovali na svetovnem tekmovanju na Kitajskem.

4.1.1 Prva stopnja razvoja robota

Z delom smo začeli marca 2015, saj smo se zavedali, da bo to za nas, ki na tem področju nimamo veliko predznanja, velik zalogaj. Učili smo se programirati in komunicirati z mikroprocesorjem na plošči Arduino. To je pomenilo lažjo serijsko komunikacijo ter lažje programiranje, saj ima Arduino plošča po navadi integriran programator, ki je hkrati USB komunikator in tako skrbi za programiranje robota in komunikacijo med računalnikom in robotom preko USB kabla. Že od samega začetka pa vse do končne različice smo delali z

mikrokontrolerjem ATmega2560. V prvi stopnji razvoja in tudi pri prvem pravem prototipu smo uporabili že narejeno ploščo Arduino Mega 2560, ki vsebuje zgoraj imenovani mikrokontroler.

Po izboru čipa smo se osredotočili na iskanje najbolj primernih senzorjev za sledenje črni črti. Ugotovili smo, da so za nas najbolj primerni senzorji, ki temeljijo na tehnologiji IR, ker takšna izbira senzorjev onemogoča interferenco z vidno svetlobo, prav tako pa je zanje barva podlage, v kolikor je po odbojnosti podobna beli barvi, dokaj nepomembna. V prvi in drugi stopnji razvoja robota smo uporabili senzorje QRE1113 iz spletne trgovine SparkFun. IR senzorji se v primerjavi z drugimi vrstami odbojnostnih senzorjev tudi veliko hitreje odzivajo na spremembo okolice, kar nam je omogočalo hitrejšo vožnjo brez izgube zanesljivosti in natančnosti pri sledenju.

Senzor QRE1113 na izhodu vrne analogni signal, katerega vrednost je premosorazmerna z odbojnostjo predmeta. Analogni signal nato mikroprocesor prebere in ga digitalizira v 10-bitno vrednost, program pa nato računa in se odziva na to število.

Nato smo izbrali primerne pogonske motorje za robota in gonilnike, ki bodo takšne motorje lahko poganjali. Lahko bi izbrali motorje stepper, servo motorje ali pa Micro Metal Gearmotor motorje. Za prve se nismo odločili, ker so preveliki in pretežki, servo motorje pa smo izključili zaradi cene. Izbrali smo Micro Metal Gearmotor motorje, ki smo jih kupili v internetih trgovinah Pololu, Adafruit in SparkFun. Ponujajo veliko paleto raznih prestavnih razmerji železnih prestav, ki so pritrjene neposredno na motor. So odlični za uporabo v tem projektu iz več razlogov. Med drugim so zelo majhni, dokaj močni, robustni, ne zahtevajo posebnega vzdrževanja, lahko pa izberemo motor z nam najbolj ustreznim prestavnim razmerjem.

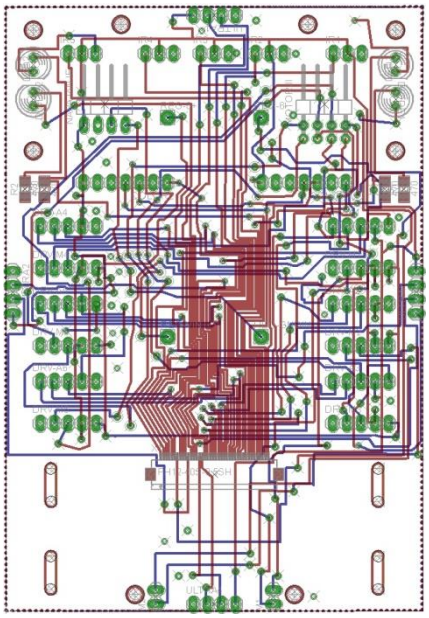
Za motorje, ki so v bistvu navadni krtačni motorji za enosmerni tok, smo izbrali gonilnike DRV8838 iz spletne trgovine Pololu. Ti gonilniki delujejo na napetosti do 11 V, pri polni obremenitvi pa lahko za kratek čas prenesejo tokove tudi do 1,8 A. Takšno konfiguracijo smo uporabili v vseh prihodnjih različicah robotov, ker je zanesljiva in jo je zelo preprosto upravljati. Gonilnike upravljamo s PWM signalom in še z nekaj drugimi digitalnimi signali, ki določajo smer vrtenja in možnost preklopa v stanje mirovanja.

Pri zadnjih verzijah robotov v tej fazi smo dodali tudi ultrazvočne senzorje za merjenje razdalje, barvnih senzorjev pa še nismo uporabljali, saj so zahtevali znanje I2C komunikacije.

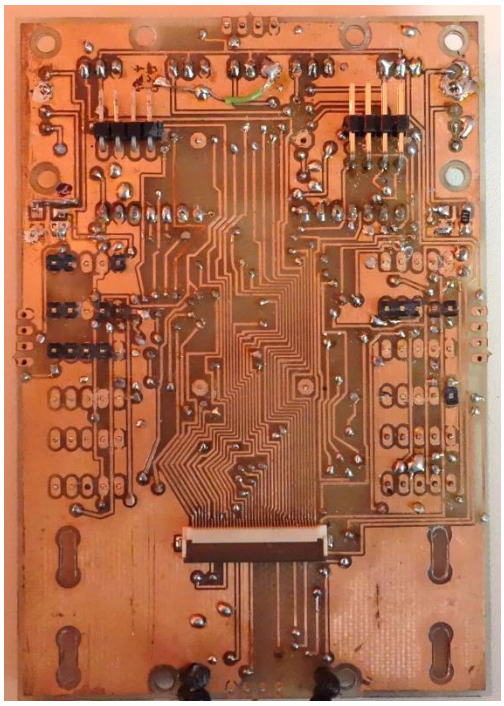
Roboti, ki so nastajali v tej fazi, so bili večinoma vsi zelo ozko funkcionalni, saj je bila njihova glavna naloga sledenje črti. Vsi so bili napajani preko tankih žic, ki so vodile do blizu postavljenega enosmerno napetostnega napajalnika. Tako smo lahko preizkusili odzivanje robota na višjo oziroma nižjo delovno napetost ter odčitali tok pri obremenitvi in tako ocenili velikost baterije, ki smo jo pri nadaljnjem delu potrebovali.

4.1.2 Druga stopnja razvoja robota in prvi pravi prototip

V tej stopnji smo zgradili prvi pravi prototip, ki smo ga uporabili na slovenskem državnem tekmovanju v Mariboru. Robota smo načrtovali in izdelali popolnoma na novo. Začeli smo z dogovarjanjem glede velikosti robota in nadaljevali z zgradbo osnovne konstrukcije. Sklenili smo narediti majhnega in okretnega robota, ki bo imel dve, eno nad drugo vzporedno postavljeni, obojestransko potiskani tiskani vezji. Spodnja plošča je služila tudi kot podvozje, saj smo nanjo privijačili motorje in pri zadnji različici celo nekaj nosilnih delov, natisnjenih s 3D tiskalnikom.



Slika 3: Plošča tiskanega vezja narisana v programu EAGLE



Slika 4: Izrezkana plošča narejena po zgornjem načrtu

Tiskana vezja za naš prvi pravi prototip robota smo načrtovali s pomočjo brezplačnega programa za risanje tiskanih vezji Eagle. Bilo je izdelano s CNC rezkarjem, kar pomeni, da smo imeli tudi veliko težav z dovoljeno debelino vodnikov na tiskanem vezju.

Spodnje tiskano vezje je skupaj z zgornjim tvorilo glavno konstrukcijo robota, zato smo za tiskano vezje izbrali dokaj debele plošče. Na sprednjem delu spodnje plošče je bilo naspajkanih 5 IR senzorjev QRE1113, ki so zaznavali položaj robota glede na črto. Na vsaki strani IR senzorjev sta bili prispajkani LED diodi s premerom 5 milimetrov ter fotodiodi, ki sta bila namenjeni zaznavanju srebrne podlage. Vendar to v praksi, zaradi premajhne razlike med odbojnostjo bele in srebrne barve merjene s to metodo, ni bilo nikoli realizirano. Na spodnji plošči je bil prispajkan tudi napetostni regulator. Ta je napetost Li-ionskih baterij spreminjal v napetost 6 V, ki smo jo potrebovali za napajanje vezja in motorjev. Na spodnjo ploščo smo prispajkali 6 gonilnikov DRV8838, ki so služili upravljanju Micro Metal 6 V DC Gearmotorjev. Na vsaki strani plošče so bili na zelo ugodni višini 2,5 centimetrov prispajkani ultrazvočni senzorji za določanje razdalj do najbližjega objekta na vsaki strani robota.

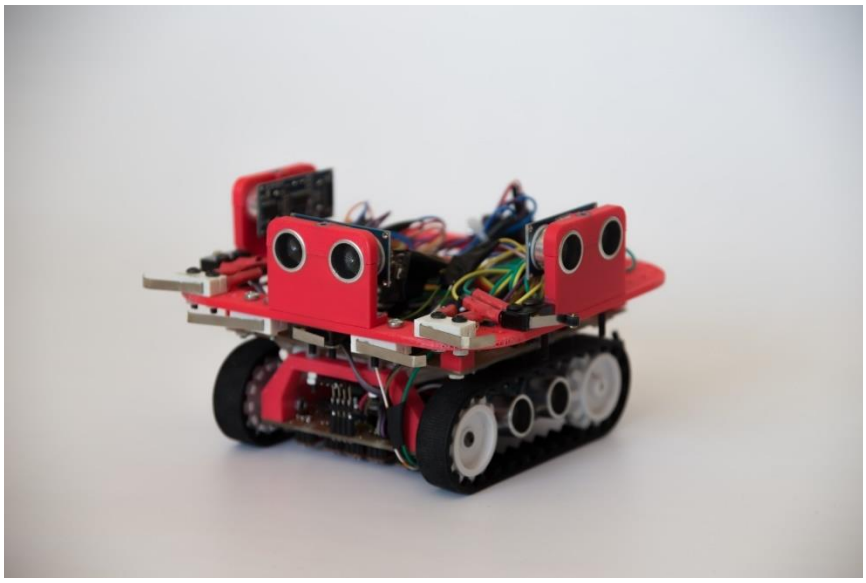
Komunikacija med zgornjo in spodnjo ploščo je potekala preko 40-žilnega flat kabla FH12-40 S-0.5 SH. S spajkanjem in dobavo tega kabla smo imeli kar nekaj težav, saj je razmik med vodniki v kablu le 0,5 milimetra. Zgornja plošča je imela na sredini izrezkano luknjo, po kateri je prišel flat kabel s spodnje plošče na zgornjo. Okoli luknje so bile po merah Arduino Mega 2560 plošče razporejene letvice, ki so omogočale, da smo na zgornjo ploščo pritrdili Arduino ploščo. Tudi na zgornji plošči so bili, tako kot na spodnji, na vsaki strani prispajkani ultrazvočni senzorji. Na zgornji plošči je bilo 12 priklopov za stikala, ki so bila razporejena po robotu, ob vsakem pa je bil še SMD (ang. surface-mount device) upor upornosti 10 kΩ, ki je potreben za vezavo stikal na Arduino ploščo. Za lažje programiranje smo na robotu uporabili tri LED diode (rdeče, rumene in zelene barve). Za zaščito pred poškodbo vezja in baterije v primeru kratkega stika smo uporabili stekleno varovalko (2 A).

Dva Micro Metal 6 V DC Gearmotorja sta poganjala gosenice. Na tej verziji robota so se gosenice obnesle odlično, a so se na končni izvedbi zaradi prevelike teže izkazale za neuporabne, saj so se pogosto "sezule". Ostali deli so bili na tej različici robota natisnjeni s 3D tiskalnikom.

Robot so poganjale tri Li-ionske baterije napetosti 3,7 V s kapaciteto 1500 mAh, ki so bile vezane zaporedno, da so skupaj dosegle napetost 11,1 V. To napetost smo nato na dveh regulatorjih napetosti uravnali na raven, ki smo jo potrebovali. Prvi regulator je bil pritrjen na spodnje tiskano vezje in je pretvarjal napetost baterij na 6 V, da smo potem z njo lahko napajali motorje. Drugi regulator pa je bil integriran na Arduino Mega 2560 plošči in je pretvarjal napetost baterij na 5 V, saj je pri tej napetosti deloval mikroprocesor na plošči Arudino in vse druge komponente.

Prav tako je robot že imel dva barvna senzorja, pritrjena na spodnji strani podvozja. Pomaknjena sta bila 5 milimetrov za IR senzorje, ki so brali odbojnost tal. Ker imajo

barvni senzorji, ki smo jih uporabljali, stalen in samo en naslov za komunikacijo I2C, na isto vodilo nismo smeli priključiti dveh enakih barvnih senzorjev. Zato smo v robota dodali Arduino Pro Mini ploščico, ki je veliko manjša plošča s procesorjem ATmega328. Služila je samo za to, da je, ob zahtevi glavnega mikrokontrolerja ATmega2560, prebrala drugi barvni senzor na svojem, od prvega procesorja neodvisnem, vodilu I2C in nato podatke po serijski komunikaciji poslala glavnemu mikrokontrolerju.



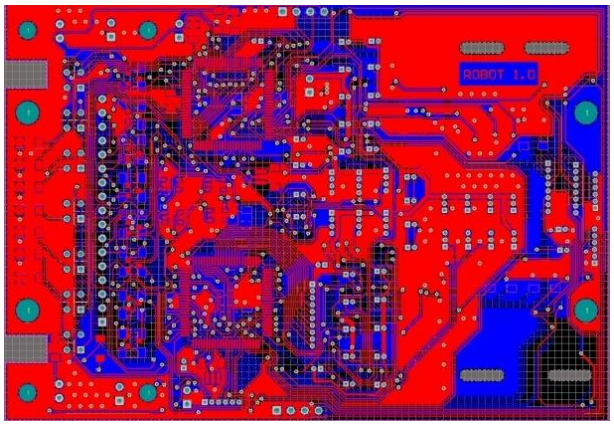
Slika 5: Prvi pravi prototip, s katerim smo se udeležili državnega tekmovanja

4.1.3 Tretja stopnja razvoja robota in njegova končna različica

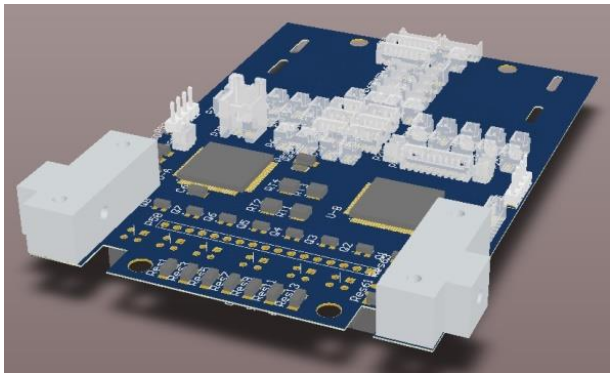
Po državnem tekmovanju smo bili z našim tedanjim robotom rahlo nezadovoljni, saj zaradi prevelikega števila strojnih in tudi programskih napak na tekmovanju ni zmagal. Odločili smo se narediti novega robota, ki bo v vseh pogledih boljši od prvega pravega prototipa, na katerem smo naredili kar nekaj napak in smo jih želeli v tej novi različici popraviti. Najbolj smo si prizadevali narediti čim bolj togega robota z velikim številom zanesljivih in natančnih senzorjev, ki bi v vsakem trenutku točno poznal svojo lokacijo in postavitev ovir ter predmetov okoli njega.

Odločili smo se za robota s tremi tiskanimi vezji. Prvo bo glavno vezje in bo nosilo dva mikrokontrolerja, nove IR senzorje in veliko priklopov za senzorje, drugo vezje bo vezje, na katerem bodo štirje samostoječi barvni senzorji, tretje pa bo vezje za gonilnike motorjev in bo zato ustvarjeno za večje tokove. Tiskana vezja smo pri tej različici robota narisali v programu Altium, ki je veliko naprednejši od programa EAGLE, ki smo ga uporabljali za risanje prejšnjega. Najprej smo vse elemente povezali na električni shemi in nato definirali velikost vezja ter začeli posamezne elemente postavljati na vezje. Ko smo dosegli želeno postavitev, smo začeli z zamudnim procesom povezovanja vseh komponent med seboj. Pri tem smo morali biti zelo iznajdljivi in previdni, saj smo morali narediti okoli 2000 povezav na plošči velikosti 76 kvadratnih centimetrov. Na spodnjih slikah

lahko vidite tiskana vezja in njihove 3D modele, narisane v programu Altium. Tiskana vezja nam je kasneje profesionalno izdelalo podjetje Lingva iz Cerknice.



Slika 6: Plošča tiskanega vezja za končno različico robota narisana v programu Altium

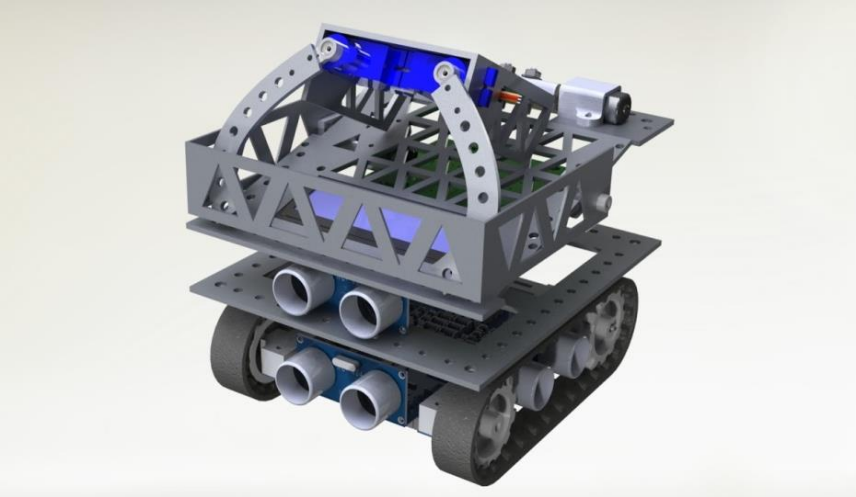


Slika 7: Računalniški 3D prikaz plošče v programu Altium

Slika 8: Plošča tiskanega vezja narejena po naših načrtih narisanih v programu Altium

Celega robota smo pred začetkom fizičnega dela narisali v programu za 3D modeliranje Solidworks. Model tiskanih vezji s prispajkanimi elementi smo izvozili iz programa Altium. Na podlagi 3D modela smo kasneje izdelovali druge elemente robota, ki smo jih združevali v dokumentu ROBOT-ASSEMBLY, dokler nismo prišli do končne različice. Slike narejene v programu Solidworks lahko vidite v nadaljevanju naloge.

Večina konstrukcijskih delov robota, ki smo jih predvideli s pomočjo 3D modela, je bila izdelana iz aluminijaste pločevine debeline 1,8 milimetrov, ki so nam jo po naših načrtih z laserjem izrezali v Emo Orodjarni Celje. Dele, ki jih je bilo potrebo upogniti, smo upognili doma s pomočjo kladiv, klešč in primeža. Najprej smo za premikanje želeli uporabiti gosenice, tako kot na vseh prejšnjih verzijah robota, a so se te zaradi prevelike teže robota "sezuvale", kar pa je pomenilo njegovo nevoznost. Zaman smo se trudili to težavo odpraviti. Poskušali smo z napenjanjem gosenic, a se je to izkazalo za neuspešno. Izdelali smo ščite, ki naj bi služili temu, da gosenice ne bi mogle zdrseti iz pogonskega kolesa, a se je tudi to izkazalo za neuspešno.

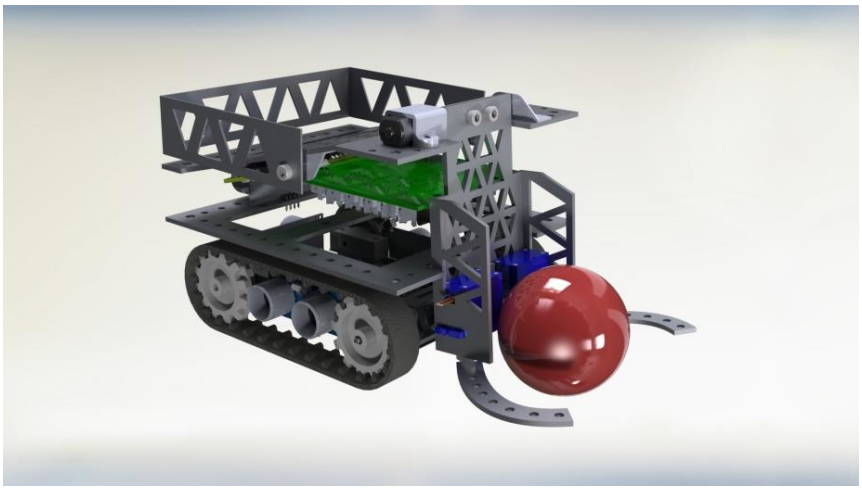


Slika 9: Računalniški render 3D modela robota

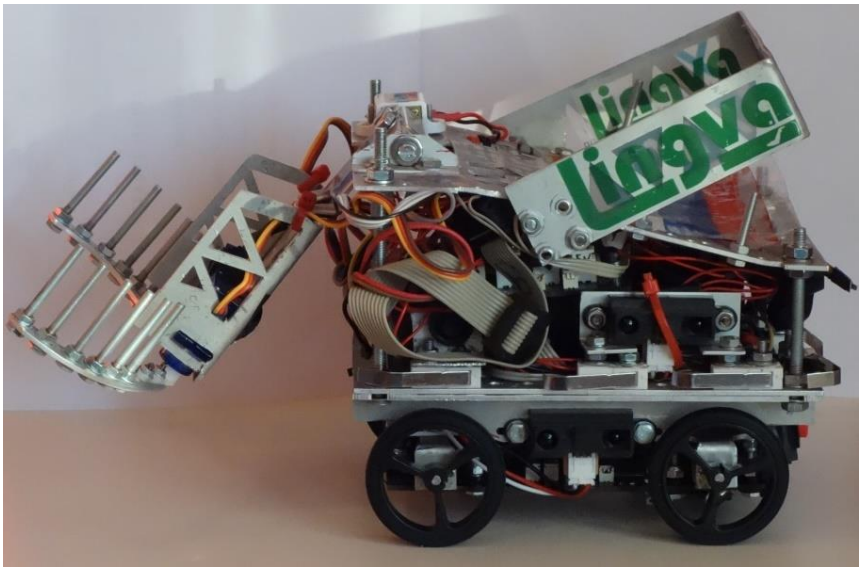
Odločili smo se za uporabo koles. Ker smo uporabljali zavijanje s pomočjo različnih hitrosti gosenic na posamezni strani robota, smo bili prisiljeni v uporabo 4 koles, ki po 2 in 2 delujeta sinhrono. Lahko bi uporabili drsljiv zadnji konec, a bi se robot zatikal na hitrostnih ovirah, ki so bile nastavljene na progi. Tako smo morali izdelati nove nosilce za dva nova motorja, ki smo jih pritrdili na sprednjem delu robota tam, kjer je bila prej nameščena le os za gosenice. Nekaj težav smo imeli z ne popolnoma sinhronim delovanjem motorjev, kar pa smo kasneje odpravili programsko.

Ker je del naloge tudi pobiranje žogic ti. "žrtev", smo izdelali za ta namen posebno roko, ki jo je dvigoval in spuščal Micro Metal 6 V DC Gearmotor motor z večjim prestavnim razmerjem. Klešče za prijem žogice sta odpirala in zapirala dva 9 gramska servo motorja. Da bi prihranili čas pri prenosu žogic na varno točko, smo si zamislili sistem, ki lahko naenkrat prevaža do maksimalno 5 žogic. Dva servo motorja jo primeta, motor dvigne celotno roko nato pa servo motorja žogico spustita. Le-ta pade na ploščad nad robotom, ki

je nagnjena za 12,5 °. Ko robota, polnega žogic, pripeljemo do varne točke, se loputa na sprednji strani nagnjene plošče odpre s pomočjo Micro Metal 6 V DC Gearmotorja in žogice se zaradi sile teže skotalijo v varno točko.



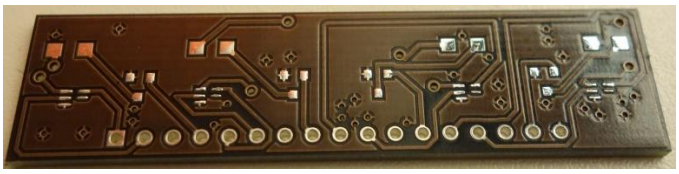
Slika 10: Računalniški render 3D model robota, ki pobira žogo



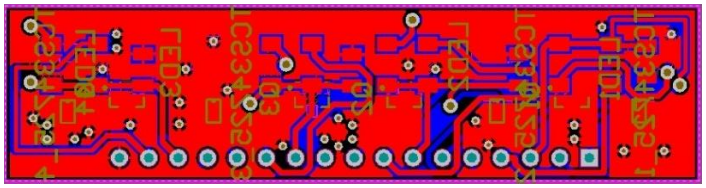
Slika 11: Slika robota z vidno razprto roko in zadrževalnimi vratci za žoge

Zaradi prevelikega števila senzorjev, ki smo jih potrebovali za natančno zaznavanje okolice, smo morali osnovni verziji dodati še en dodaten mikrokontroler. Za dva mikrokontrolerja pa smo se odločili tudi zaradi barvnih senzorjev, ki smo jih morali priklopiti na robota. Ker so imeli vsi barvni senzorji enak I2C naslov, je prihajalo do velike zmešnjave na komunikacijskem kanalu I2C. To težavo smo kasneje odpravili z razdelitvijo barvnih senzorjev na dva mikroprocesorja. Za branje barve po celotni širini robota pa nista bila dovolj le dva barvna senzorja, temveč smo uporabili štiri. Ker smo uporabljali štiri barvne senzorje TCS34725 je še vedno prihajalo do težav pri I2C kumunikaciji. To težavo smo rešili z izmeničnim izklapljanjem senzorjev v paru, tako da je bil v določenem trenutku vklopljen samo en senzor na posamezen mikrokontroler. S

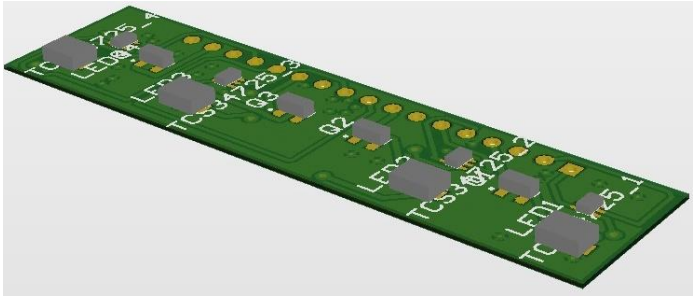
pomočjo dveh mikrokontrolerjev smo tako enkrat hitreje brali barvne senzorje, kot če bi imeli samo enega. Zaradi prostorske stiske na tiskanem vezju, ki je vsebovalo barvne senzorje, smo morali del vezja, ki je kontroliral in napajal barvni senzor, narediti na glavni plošči. Tako smo na glavno ploščo prispajkali šestnajst 10 kΩ uporov, osem 100 pF kondenzatorjev, osem BSS138LT1 in štiri RT9193. Na plošči za barvne senzorje, ki se je nahajala pod glavno ploščo in je bila na njo pritrjena s 17 pinsko letvijo, so bili poleg štirih barvnih senzorjev TCS34725 še štirje 100 pF kondenzatorji, štirje 10 kΩ upori, štirje 470 Ω upori, štirje BSS138LT1 in štiri SMD LED diode, ki so zagotavljale potrebno svetlobo za pravilno delovanje barvnih senzorjev. Poleg glavnega tiskanega vezja in tiskanega vezja za barvne senzorje smo izdelali še posebej tiskano vezje, ki je lahko zdržalo večje električne tokove, na katerem je bilo šest gonilnikov za motorje DRV8838.



Slika 12: Slika tiskanega vezja za barvne senzorje



Slika 13: Plošča vezja narisana v programu Altium



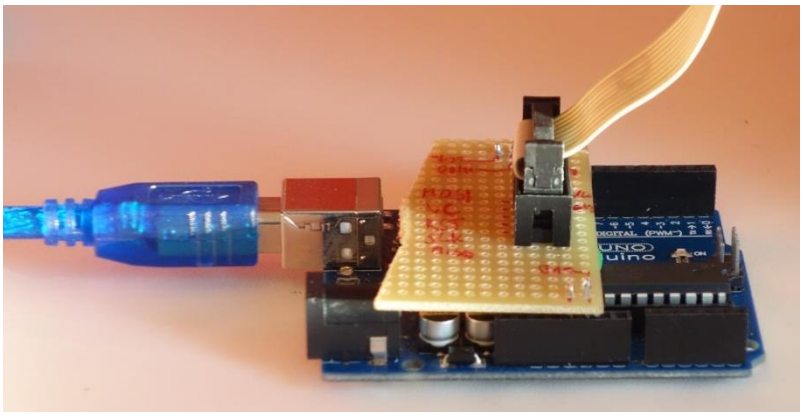
Slika 14: Računalniški 3D model tiskanega vezja v programu Altium

Tako smo sedaj imeli na glavnem tiskanem vezju dva enaka mikrokontrolerja ATmega2560, ki sta med seboj komunicirala preko serijske komunikacije. Na glavnem vezju sta jima takt določala dva resonatorja CSTCE16M0V53-R0, ki sta mikrokontrolerjem dajala takt 16 MHz. Zraven vsakega resonatorja je bil vzporedno vezan pripadajoč upor upornosti 1 MΩ, ki je skrbel za njegovo pravilno delovanje.

Mikrokontrolerju, ki smo ga na vezju poimenovali U-A, smo dodelili glavno vlogo, saj je upravljal vožnjo robota in se odločal, kako se bo le-ta odzival na različne situacije, v katerih se je lahko znašel. Skrbel je tudi za pogonske motorje, katere je kontroliral preko PWM izhodov, ki so vodili do gonilnikov za motorje DRV8838. Na glavni mikrokontroler so bili

priključeni tudi vsi novi na ploščo integrirani IR senzorji CNY70 za sledenje črti, da ne bi prihajalo do zamud in zato počasnejšega odzivanja pri zavijanju. Nove senzorje smo uporabili zaradi večje natančnosti in manjše občutljivosti na razdaljo in kot pod katerim senzorji gledajo na odbojno površino, kar je še posebej pomembno pri speljevanju na klanec in z njega. Tudi 11 stikal, ki so potrebovala takojšni odziv, smo priključili na mikrokontroler U-A. Štirje stranski ultrazvočni senzorji so bili prav tako vezani na ta mikroprocesor. Na njega je bil vezan tudi šest pinski priklop, preko katerega smo čip programirali.

Ker na vezju nismo imeli integriranega programatorja, smo morali mikrokontrolerja programirati preko zunanjega. V ta namen smo uporabili ploščo Arduino Uno, ki je imela naložen poseben program. Ta je glavni procesor na plošči spremenil v programator mikroprocesorja na robotu. Programiranje mikrokontrolerja na robotu je potekalo preko MISO, MOSI, SCK, Reset. S tem, ko smo uporabljali zunanji programator, smo na robotu prihranili veliko zelo dragocenega prostora.



Slika 15: Programator z priključenim USB kablom in kablom za programiranje robota

Za stabilizacijo napetosti in odpravljanje šuma med napajanjem smo se odločili, da za vsak mikrokontroler uporabimo štiri kondenzatorje vrednosti 100 nF. Drugi, manj pomembni mikrokontroler, smo poimenovali U-B. Ta je izvajal stranske izračune in preko serijske komunikacije pošiljal podatke senzorjev mikrokontrolerju U-A. Za izbiro enakega mikrokontrolerja kot pri U-A smo se odločili zaradi lažje izdelave. Uporabili smo ga že pri prejšnjih projektih in predhodnih verzijah našega robota, zato smo ga dobro poznali. Poleg tega ima veliko vhodnih in izhodnih linij. Na njega je bilo priklopljenih 10 stikal, ki so večinoma nadzirala delovanje roke za pobiranje žogice, več senzorjev razdalje in celo nekaj kontrolnih linij za motorje. Celotno delovanje pobiralne roke je avtonomno nadziral mikročip U-B.

Robota je v končni različici poganjala Li-poly baterija z napetostjo 11,1 V in kapaciteto 850 mAh. Baterija je zadoščala za 3 do 4 ure intenzivnega dela. Ker smo imeli do baterijskega prostora zelo preprost dostop, smo lahko baterijo ob izpraznitvi zamenjali z drugo identično baterijo, ki smo jo v vmesnem časom polnili. Tako smo lahko delali na robotu brez prestanka, saj je bila ena baterija praktično ves čas polna. Baterija je zelo majhna in je edina trocelična baterija, ki je bila dovolj majhna za uporabo v robotu.

4.2 Razvoj programa za robota

Razvoj programa je temeljil na nadgradnji in posodobitvi programa, ki smo ga uporabili za robota, ki smo ga na tovrstnem tekmovanju uporabili že leto prej. Na teh tekmovanjih namreč sodelujemo že od 1. letnika dalje. Vsako leto sestavimo novega robota ter napišemo nov program, pri katerem ohranimo vse dobre stvari in odpravimo težave, ki jih je imel prejšnji. Zaradi spremembe pravil, predvsem v zadnji sobi, smo bili primorani na novo načrtovati celoten program za zadnjo sobo.

Ker smo prvič delali v programskem okolju Arduino, smo začeli z nekaj manjšimi problemi, da smo se navadili na programsko okolje in spoznali najpogostejše težave, kot so na primer nalaganje programa na procesor v primeru napake.

4.2.1 Razdelitev programa na podprograme

Program smo začeli pisati z obsežnim spiskom definicij in inicializacij v delu setup(). Nato smo v delu loop() začeli s programom. Zaradi preglednosti in lažjega programiranja smo ga razdelili na podprograme, ki vsak posebej rešujejo vnaprej določene dele osnovnega problema. V glavno zanko v delu loop() smo nato postopoma, skozi razvoj programa, vnašali vedno nove pogoje, ki so ob izpolnitvi zagnali posamezne podprograme. Te naj glavni program kliče za izpolnitev določenega manevra, branja senzorjev ali pa za izmenjavo podatkov med mikrokontrolerjema. Tako smo dobili program s skupno dolžino preko 2500 vrstic, razdeljen na približno 16 podprogramov.

4.2.1.1 Komunikacija med čipoma

Ker ima robot implementirana dva čipa hkrati, morata ta dva za pravilno odzivanje robota na okolico delovati v paru in sicer z medsebojno serijsko komunikacijo, ki poteka po načelu master-slave. Čip A je master, čip B pa slave. Čip B čaka dokler mu A ne pošlje sporočila, B prebere sporočilo, naredi zahtevano in konča svoje delo z dogovorjenim signalom, ki čipu A sporoči, da lahko nadaljuje s svojim delom, predvsem pa, da je čip B končal s svojim delom in mu tako lahko pošlje naslednjo skupino navodil.

4.2.1.2 Sledenje črti

Robot sledi črti s pomočjo palete svetlobnih senzorjev na spodnji sprednji strani tiskanega vezja. S senzorjem, ki se nahaja na sredini robota, poskuša ves čas ostati nad črto. Ko pride do ovinka, določi kako oster je ovinek in temu primerno prilagodi moč motorjev. Pri prekinitvah najprej preveri, če je res na koncu črte in če je, se poravna, nanjo zapelje naravnost za dolžino prekinitve in začne iskati črto z novega mesta. Če je to iskanje neuspešno, se vrne nazaj na zadnje mesto, kjer je še zaznal črto in ponovi iskanje. Zeleno barvo pri križiščih zazna z barvnima senzorjema na vsaki strani, nato se odloči na kateri izvoz mora zaviti.

4.2.1.3 Ovire na poti

Ko robot doseže oviro, s pomočjo ultrazvočnih senzorjev razdalje pregleda levo in desno, da ugotovi na kateri strani je več prostora. S to informacijo se, glede na druge dodatne

parametre, obrne v najbolj primerno smer in obvozi oviro. Ko pride na drugo stran ovire, začne ponovno iskati črto, s pomočjo kompasa pa ohrani pravilno orientacijo.

4.2.1.4 Iskanje žrtve

Robot zazna srebrno črto s pomočjo svetlobnih diod. S tem ve, da je prišel v tako imenovano "zadnjo sobo". V njej se najprej poravna glede na stene vzdolž daljše stranice. Začne slediti steni do konca sobe, kjer se obrne, pomakne za eno širino in nadaljuje nazaj do druge strani sobe. To ponavlja dokler ne prevozi celotne površine sobe. Ko pred sabo zazna žogico, se obrne in jo pobere. Ko ima pobrane 4 žogice ali pa, ko je pregledal celotno sobo, se poravna na steno in obišče vse kote, dokler ne najde prostora kamor mora odložiti žogice. Tam se poravna na steno, se obrne in jih odloži. Če še ni pregledal celotne sobe, začne ponovni pregled od začetka. Žogice se namreč ob stiku z robotom lahko začnejo kotaliti po prostoru.

4.2.1.5 Zamenjava gosenic s kolesi

Po zamenjavi gosenic za kolesa smo morali popraviti večino kode, ker smo pri gosenicah uporabljali le dva motorja za vožnjo in obračanje, pri kolesih pa štiri. Zamenjava je posledično pomenila dodatno kodo za zavijanje. Zato smo ustvarili dve metodi, ki urejata hitrost levih in desnih koles.

4.2.1.6 Nenatančnost kompasa in težave pri prehodu čez 360 stopinj

Pri meritvah kompasa po daljšem premikanju po polju lahko pride do manjše napake zaradi razlik v magnetnem polju na različnih delih arene. Zato imamo poseben podprogam za preverjanje ovinkov, ki bi se naj končali v bližini 10 stopinj magnetnega severa. Ta pri prevelikem zasuku robota poskrbi za njegov zasuk nazaj na mesto, ki ga je preskočil.

4.2.2 Testiranje

Robota smo testirali na našem šolskem poligonu po vsaki posodobitvi programske opreme. Posebej pa smo testirali vse dele proge za katere smo menili, da bi nam lahko povzročali težave. Seveda smo se zavedali, da s testiranjem ne moremo dokazati odsotnosti napak, le njihovo prisotnost. Zato je robota vsak konec tedna stestirala oseba, ki je bila neodvisna od razvoja in ji nismo povedali, kaj smo na njem popravljali. V osnovi smo testirali predvsem po modulih, ob zaključku dneva pa smo naredili še vsaj eno testiranje celotnega sistema, kjer smo vključili vse nove module in testirali celoto.

4.2.3 Branje podatkov s senzorjev

4.2.3.1 Stikala

Pri branju stikal pričakujemo le vrednost 1 ali 0, kar pomeni ali je stikalo pritisnjeno ali pa ni. Stikalo je na eni strani vezano na 5 V, na drugi strani pa je upor, za njim pa vhod na procesorju. Ko je stikalo pritisnjeno, pride napetost 5 V na stran vhoda in vhod to prebere kot vrednost 1. Zato za branje stikal uporabimo kar funkcijo digitalRead(). V odgovor dobimo true oziroma 1, če je stikalo pritisnjeno ali false oziroma 0, če ni.

4.2.3.2 Ultrazvočni senzorji

Za branje ultrazvočnih senzorjev za merjenje razdalje smo uporabili že napisano knjižnico NewPing, ki skrbi za branje podatkov z več senzorjev brez prekrivanj. Vsaka knjižnica za branje teh vrst senzorjev najprej pošlje kratek zvok visoke frekvence senzorju. Na senzorju sta dva majhna zvočnika. Ob prejetem signalu s procesorja se eden od njiju zelo hitro zatrese in proizvede nam neslišni ultrazvok. Ko se ta odbije od bližnjega objekta, pripotuje nazaj do drugega zvočnika in ta prejme tresljaje, ki se nato ojačajo in pošljejo nazaj procesorju. Čas, ki ga signal porabi da pride nazaj, je tako premosorazmeren z dolžino, ki jo je zvok prepotoval. Po branju dobimo v obdelavo razdaljo do najbližje ovire, ki jo pri velikih razdaljah priredimo na 100 centimetrov, ker nič na tej razdalji ne vpliva več na robota in bi ga samo zmedlo.

4.2.3.3 Barvni senzorji

Z barvnimi senzorji beremo barve zelo preprosto. Z LED diodo, ki oddaja kar najbolj belo barvo, svetimo na površino, ta pa določene valovne dolžine bele barve vpije, druge odbije. Odbita svetloba gre nato skozi tri različne filtre (rdečega, zelenega in modrega) za njimi pa so fotodiode, ki definirajo količino svetlobe, ki jo filtri spustijo skozi za vsako valovno dolžino posebej. Barvni senzorji nam tako vračajo RGB komponente barve, ki jo zaznajo. Mi samo pregledamo, če je ta barva zelena (RGB komponente zelene barve smo izvedeli od organizatorjev tekmovanja). Pri tem upoštevamo možne napake pri meritvi zaradi svetlobe in nepokrivanja celotne površine. Paziti moramo tudi na prižiganje LED diod za bolj pravilno branje. Ker imajo vsi barvni senzorji enak I2C naslov, smo morali paziti na njihovo prižiganje in ugašanje, da nismo brali z večih naenkrat in s tem popačili signala.

4.2.3.4 IR senzorji za razdaljo

IR senzor za razdaljo deluje po principu toplotnega sevanja. Senzor privzame, da so objekti, do katerih meri razdaljo, približno enako dovzetni za odbijanje sevanja. Tako meri intenziteto prejetega sevanja in jo primerja z oddanim sevanjem. S senzorja beremo analogno razdaljo, ki jo algoritem obdela v robotu razumljivo razdaljo - centimetre.

4.2.3.5 Fotodiode

Na začetku vsakega teka robota najprej postavimo nad srebrno črto in pritisnemo na stikalo, da prebere odbojnost srebrnega traka s pomočjo fotodiod, ki so nameščene pod robotom. Med branjem mikročip bere analogno vrednost, ki jo vrača senzor. Ta rezultat uporabljamo v primerjavi z rezultatom, ki ga dobimo, ko z isto metodo beremo odbojnost bele barve. S tem ugotovimo ali smo trenutno nad srebrnim trakom. Zaradi zelo majhnih razlik v velikosti vrednosti odbojnosti bele in srebrne barve je ta metoda za določanje srebrne barve lahko problematična.

4.2.3.6 Kompas senzor

Uporabljamo knjižnico compass. Najprej s compass.read() posodobimo vse meritve, nato pa s compass.heading() od kompasa dobimo kot, za katerega je robot obrnjen glede na sever. Paziti smo morali, da smo kompas na začetku vsake vožnje kalibrirali, zaradi spreminjajočega se magnetnega polja zaradi napeljave, orodij in okolice.

4.2.3.7 IR senzorji za odbojnost

Iz IR senzorjev za odbojnost dobimo analogno vrednost tako, da beremo napetost, ki jo ustvari IR fotodioda v senzorju, ko na njo pade del od tal odbite svetlobe, ki jo je proizvedla IR LED dioda. Pred vožnjo naredimo kalibracijo črne črte in bele podlage in s tem določimo meji, ki sta nato uporabljeni v kodi. Podatkov ne obdelujemo, ker je za naše zahteve dovolj primerjava analognih branj s senzorja.

4.2.4 Upravljanje motorjev

Navadnim motorjem ukazujemo z dvema podatkoma in sicer s hitrostjo podano kot vrednost PWM signala, ki je lahko katerokoli celo število med vrednostma 0 in 255. To vrednost signalu na določenem pinu določimo z ukazom *analogWrite ("pin").* Motorjem pa lahko spreminjamo tudi smer, kar naredimo z digitalnim pinom vezanim na določen digitalni vhod na gonilniku za motor, kateremu spreminjamo smer. Digitalni pin ima lahko vrednosti 0 ali 1, kar določa ali se bo motor vrtel v smeri urinega kazalca ali pa obratno.

Za servo motorje uporabljamo knjižnico servo. Najprej definiramo servo motorje, potem pa jih zavrtimo za želeni kot. Izdelati smo morali blokado v kodi, da ga nismo zavrteli za več kot 180°, saj bi ga drugače lahko uničili.

5 Rezultati in razprava

Robot je v veliko pogledih presegel pričakovanja in naše domneve v hipotezah. Senzorji IR so se v areni odzivali zelo hitro in tudi s presenetljivo natančnostjo. Algoritem za sledenje črti smo s tako velikim številom senzorjev lahko zelo izpopolnili in tako dobili robota, ki je imel skoraj nične možnosti izgube črne črte.

Motorji Micro Metal 6 V DC Gearmotor so se kot pogonski motorji obnesli nadpovprečno dobro, saj so dajali dovolj navora za prekoračitev katerekoli ovire in hkrati bili dovolj hitri, da smo lahko uspešno opravili nalogo v približno 4 minutah, če med izvajanjem ni bilo napak.

Prestavitev pogonskega sredstva z gosenic na kolesa je bila na žalost nujna rešitev velikega problema, ki je za nas imela tudi nekaj negativnih posledic. Na tekmovanju se je zaradi tega robot žal včasih na hitrostnih ovirah zataknil in obstal, kar se pri vožnji z gosenicami ne bi zgodilo. Predvidevamo, da bi se ob namestitvi boljših gosenic, ki se ne snamejo tako preprosto, robot odrezal veliko bolje, a smo zadovoljni z rezultatom kakršnega smo dosegli. Pozitivne posledice prestavitve so se odražale v hitrejšem obračanju robota in hitrejši vožnji nasploh, kar je pripomoglo h krajšemu času in torej večji možnosti za uvrstitev na visoko 13. mesto.

Komunikacija med obema mikrokontrolerjema je večino časa potekala odlično in tudi zelo hitro, tako da je branje barvnih senzorjev človeškemu očesu izgledalo popolnoma sinhronizirano.

Branje barve z barvnih senzorjev je bilo zelo preprosto, vrednosti RGB pa konsistenčne in ponovljive. V času celotnega tekmovanja je robot zeleno barvo spregledal samo enkrat.

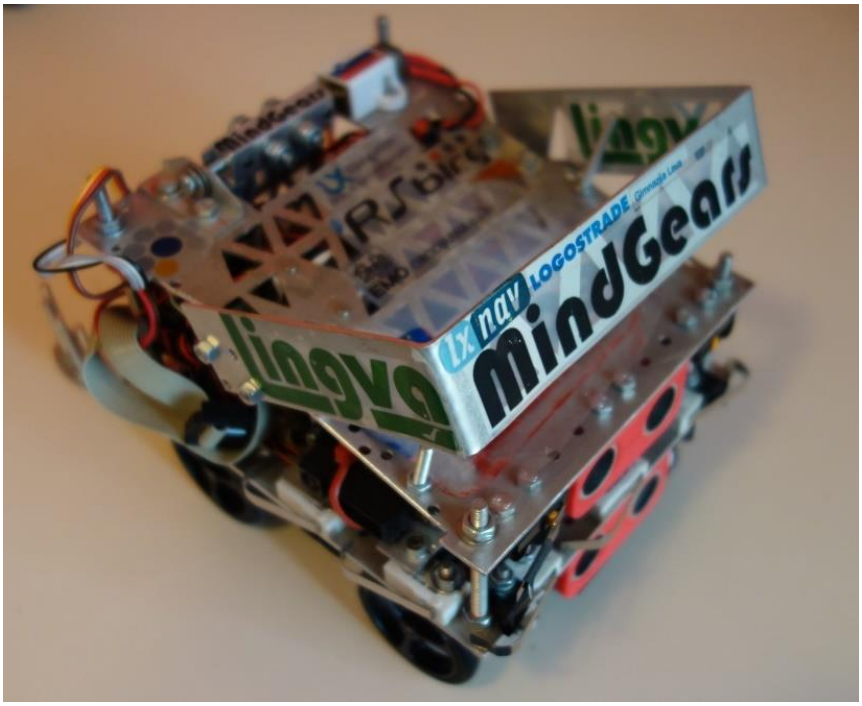
Razdalje določene z IR senzorji za razdaljo in z ultrazvočnimi senzorji so bile na naše veliko presenečenje dokaj natančne in so zaznavale tudi dosti majhne objekte. Pri nalogi določanja razdalj do bližnjih objektov bi se dalo robotski vid izboljšati s kamero ali pa morda z boljšimi ultrazvočnimi senzorji, saj smo opazili nekonsistenčnost meritev na različnih senzorjih in enakih razdaljah. Predvidevamo, da so bili za to krivi poceni senzorji, ki smo jih kupili za nekaj centov.

Končno verzijo našega robota poganja baterija, ki ima ravno prav veliko kapaciteto za tako majhnega robota. S seboj smo vedno imeli tudi nadomestno baterijo, ki se je med uporabo prve polnila in je bila tako primerna za uporabo takoj, ko smo prvo baterijo izpraznili. Baterije smo menjali brez problema, saj je bil dostop čisto zgoraj, pod nalepko slovenske zastave, skozi katero je sijal varnostni voltmeter, ki nas je obveščal o stanju baterije. Sistem je deloval brezhibno.

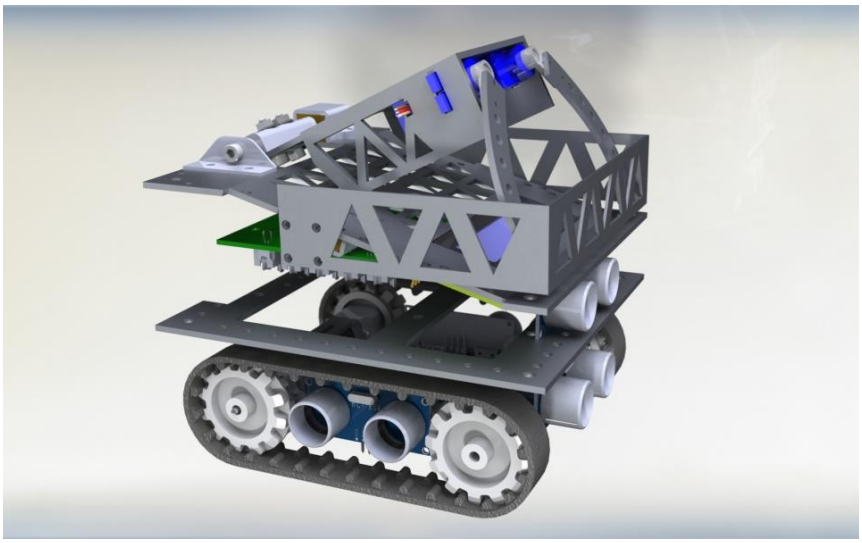
Podatki so potrdili našo glavno hipotezo, saj nam je v tekmovalnem letu uspelo narediti in sprogramirati odličnega, delujočega in funkcionalnega samogradnjega robota, ki je primerljiv tudi z najboljšimi roboti drugih ekip.

6 Zaključek

Med raziskovalnim procesom smo pridobili ogromno praktičnega in tudi teoretičnega znanja programiranja, 3D konstruiranja in načrtovanja vezji ter mehanske izvedbe robotskih komponent. Najbolj nam je primanjkovalo časa, s katerim smo se borili od samega začetka projekta. Z robotom smo seveda nadpovprečno zadovoljni in smo na svoje delo zelo ponosni, še posebej ker je bil robot nagrajen z nagrado Best Hardware Design za najboljšega robota v svoji kategoriji. Na žalost pa nismo imeli dovolj čas za programiranje, da bi v celoti lahko izkoristili potencial tega res zelo posebnega in nadvse zmogljivega, a zapletenega robota.



Slika 16: Slika robota z vidnim imenom naše skupine MindGears



Slika 17: Render 3D modela robota

7 Viri

Arduino (online). Arduino. (Zadnja sprememba 2016). (citirano 10. 3. 2016). Dostopno na naslovu: https://www.arduino.cc/.

Atmel ATmega60/V-1280/V-!”(!/V-2560/V-2561/V datasheet (online). Atmel. (Zadnja sprememba 18. 7. 2015). (citirano 10. 3. 2016). Dostopno na naslovu: http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf.

Atmel AVR (online). Wikipedia: The Free Encyclopedia. (Zadnja sprememba 14. 3. 2016 04:47). (citirano 10. 3. 2016). Dostopno na naslovu: https://en.wikipedia.org/wiki/Atmel_AVR.

CMOS (online). Wikipedia: The Free Encyclopedia. (Zadnja sprememba 2. 2. 2016 11:01). (citirano 10. 3. 2016). Dostopno na naslovu: https://en.wikipedia.org/wiki/CMOS.

DRV8838 Single Brushed DC Motor Driver Carrier (online). Pololu. (Zadnja sprememba 2014). (citirano 10. 3. 2016). Dostopno na naslovu: https://www.pololu.com/product/2990.

Lego Mindstorms EV3 (online). Wikipedia: The Free Encyclopedia. (Zadnja sprememba 14. 3. 2016 14:49). (citirano 10. 3. 2016). Dostopno na naslovu: https://en.wikipedia.org/wiki/Lego_Mindstorms_EV3.

Reduced instruction set computing (online). Wikipedia: The Free Encyclopedia. (Zadnja sprememba 21. 2. 2016 04:02). (citirano 10. 3. 2016). Dostopno na naslovu: https://en.wikipedia.org/wiki/Reduced_instruction_set_computing.

Rizman Žalik, Krista. C, C++ in C za okolje oken s primeri. 1. natis. Ljubljana: Zavod Republike Slovenije za šolstvo, 1995 (Ljubljana: Paco), 1995. ISBN 86-7759-335-7.

SparkFun Line Sensor Breakout - QRE1113 (Analog) (online). SparkFun. (Zadnja sprememba 21. 1. 2015). (citirano 10. 3. 2016). Dostopno na naslovu: https://www.sparkfun.com/products/9453.

Žumer, Viljem in Brest, Janez. Uvod v programiranje in programski jezik C++. 2. dopolnjena in razširjena izdaja. Maribor: Fakulteta za elektrotehniko, računalništvo in informatiko, Inštitut za računalništvo, 2004 (Maribor: Založniška tiskarska dejavnost tehniških fakultet), 2004. ISBN 86-435-0636-2.