

Факултет инжењерских наука Универзитета у Крагујевцу



Назив студијског програма: Машинско инжењерство

Ниво студија: Основне академске студије

Модул: Примењена механика и аутоматско управљање

Предмет: Сензори и актуатори

Број индекса: 163/2011

Никола Д. Јовић
**Пројектовање Ардуино мобилног робота
са камером мобилног телефона у улози
сензора
завршни рад**

Комисија за преглед и одбрану:

1. Проф. Др Милан Матијевић - ментор

2. _____

3. _____

Датум

одбране: _____

Оцена: _____

У оквиру овог завршног рада кандидат треба да изврши интеграцију сензора и актуатора са Arduino микроконтролером користећи и камеру мобилног телефона као сензор, а да би интегрисани систем радио као аутономни мобилни робот. Аутономност кретања робота је заснована на развоју софтвера који треба да на интелигентан начин синхронизује рад сензора и актуатора робота при извршењу одабраних задатака за које је робот ангажован.

Препоручена литература:

- [1] <http://moodle.mfkg.rs/login/index.php> Сензори и актуатори, школска 2013/14 – моодле портал предмета
- [2] Michael Margolis, *Arduino Cookbook*, 2nd edition, O'Reilly 2012
- [3] J. Warren, J. Addams, H. Molle, *Arduino Robotics*, Springer 2012
- [4] Cory Kidd, "A Short History of Robotics" - MIT Talk "Social Robots for the Future and Now", <http://video.mit.edu/watch/social-robots-for-the-future-and-now-dr-cory-kidd-mit-club-of-northern-california-27804/>, MIT, Cambridge, USA, 2014

Крагујевац,

ментор:
Проф. Др Милан Матијевић

Резиме

Рад се бави пројектовањем робота са диференцијалним погоном који има могућност аутономне навигације помоћу камере паметног телефона. Рад садржи основе теорије за пројектовање једног таквог робота, као и практичну реализацију истог (и у хардверском и софтверском смислу). Одабрани дизајн робота поседује два електрична мотора, програм за регулацију кретања робота, програм за обраду прикупљених података, микроконтролер и пратећу електронику. Робот садржи три PID регулатора (софтверски имплементирана), затим, модул за бежичну комуникацију са рачунаром преко кога добија наредбе које треба да изврши. За навигацију, робот користи QR кодове на којима су исписане наредбе. Овај робот је базиран на Arduino развојној платформи.

Кључне речи: Робот, Навигација, Компјутерска визија, PID управљање, Arduino микроконтролер, Аутономни системи, Сензори и актуатори

Abstract

This Thesis presents the design and construction of an autonomous differential driven robot that uses smart phone camera for navigation. Thesis consists of two parts: basic general instruction how to build such a robot and the actual manufacture of it. The robot has 2 Electric Motors, differential drive algorithm, data acquisition/processing software, as well as the accessory electronics. Robot has 3 PID controllers and wireless communication module for onboard computing and navigation/sensing using computer. The navigation algorithm is done in QR code. Robot utilizes an Arduino development platform.

Key words: Robot, Navigation, Computer vision, PID controller, Arduino

Садржај

РЕЗИМЕ

ABSTRACT

1. УВОД	1
2. РОБОТ	1
2.1. ЕЛЕКТРИЧНЕ И ЕЛЕКТРОНСКЕ КОМПОНЕНТЕ РОБОТА	2
2.1.1. ARDUINO РАЗВОЈНА ПЛАТФОРМА	2
2.1.2. BLUETOOTH МОДУЛ	3
2.1.3. ЕЛЕКТРО МОТОРИ И H МОСТ	4
2.1.4. УЛТРАЗВУЧНИ СЕНЗОР РАСТОЈАЊА	6
2.1.5. ФОТОИНТЕРАПТЕРИ	8
2.1.6. ДИЈАГРАМ ПОВЕЗИВАЊА ЕЛЕКТРОНСКИХ КОМПОНЕНТИ РОБОТА	9
2.2. КОНСТРУКЦИЈА РОБОТА	10
2.3. КИНЕМАТИКА РОБОТА	13
2.3.1. РАЧУНАЊЕ ПОЗИЦИЈЕ РОБОТА У ГЛОБАЛНОМ КООРДИНАТНОМ СИСТЕМУ	14
2.3.2. ДИРЕКТНА КИНЕМАТИКА РОБОТА	15
2.3.3. ИНВЕРЗНА КИНЕМАТИКА РОБОТА	16
2.4. КАРАКТЕРИСТИКЕ СЕНЗОРА И АКТУАТОРА РОБОТА	17
2.4.1. СТАТИЧКЕ КАРАКТЕРИСТИКЕ МОТОРА	17
2.5. АЛГОРИТМИ И ПРОГРАМИ ЗА УПРАВЉАЊЕ РОБОТОМ	19
2.5.1. НАВИГАЦИЈА ПОМОЋУ QR КОДОВА	19
2.5.2. УПРАВЉАЧКИ ПРОГРАМ РОБОТА ЗА МИКРОКОНТРОЛЕР АТМЕЛ АТМЕГА 32U4	24
3. ЗАКЉУЧАК	34
ЛИТЕРАТУРА	35

1. Увод

Брзим напретком технологије, у последње време су роботи у индустријским центрима све више присутни. Они могу, за веома кратко време, да ураде широки дијапазон радњи за које би човек вишеструко више времена потрошио. Те радње су најчешће заваривање, фарбање, складиштење готових производа и полупроизвода. Да би робот обавио наведене радње, потребно је прикупљање информација из реалног света и анализа прикупљених информација у реалном времену. За ту сврху посебно су развијени програми за прикупљање и обраду визуелних информација. Скуп алгоритама који омогућавају рад оваквих програма спадају у посебну поље која се назива Компјутерска визија. Компјутерска визија је поље које укључује методе за прикупљање, анализирање и разумевање слика и података добијених из реалног света у циљу прављења нумеричких и симболичких информација [1].

Овај рад је посвећен прављењу једног робота помоћу Arduino развојне платформе који за циљ има прикупљање информација из света око њега путем камере, обраду тих информација и доношење одлука о акцијама са којима треба да одговори на стимуланс из света око њега.

Циљ рада је имплементација компјутерске визије у аутономно понашање мобилног робота. Током израде пројекта пазило се на цену готовог робота и на једноставност кода, тако да овакав тип робота може бити приступачан широким масама за улазак у свет роботике. Главни задатак робота је да детектује QR код, учита информације из њега и сходно њима да се наводи кроз полигон, више о начину препознавања кода може се наћи у [2].

У првом поглављу описане су електронске компоненте коришћене за израду робота, дате су електричне шеме повезивања и дате су карактеристике тих компоненти.

У другом поглављу је дато конструкционо решење робота, са тродимензионалним моделом робота, указано је на поступак који је коришћен приликом израде робота, и описани су поједини коришћени материјали.

У трећем поглављу је детаљно описана кинематика робота са диференцијалним погоном, дате су основне једначине које се користе за овакав тип робота, и приказан је начин на који робот може да има представу о својој позицији у свету.

У четвртом поглављу могу се наћи снимљене карактеристике мотора коришћених за израду робота.

У петом поглављу су описани алгоритми коришћени при програмирању робота, дати су листинзи програма, објашњени и приказани су системи управљања робота, дати су одзиви мотора са повратном спрегом и регулацијом.

У закључку је дато виђење аутора рада о постигнутом понашању пројектованог робота, и могућностима даљих побољшања.

2. Робот

2.1. Електричне и електронске компоненте робота

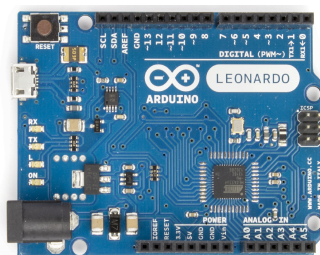
2.1.1. Arduino развојна платформа. Микроконтролер

Arduino је развојна платформа која је базирана на Atmel AVR микроконтролерима, са C++ развојним окружењем. Постоји више Arduino плоча, као што су:

- Arduino Uno
- Arduino Due
- Arduino Due Millanove
- Arduino Mega
- Arduino Leonardo
- Arduino Pro Micro
- Arduino Nano

За овај пројекат искоришћена је Arduino Leonardo развојна плоча која поседује:

- ATmega32u4 микроконтролер брзине 16 MHz
- 28KB FLASH меморије за складиштење програма
- 2.5KB RAM меморије
- 1KB EEPROM меморије за складиштење података
- 12 Аналогних улаза
- 20 Дигиталних улаза и излаза
- 7 PWM канала
- USB конекцију
- Физички тастер за поновно покретање
- LE диоду која је повезана на 13. дигитални излаз
- Заштиту од прекомерне јачине струје
- Уграђен регулатор напона на 3.3 V и 5 V

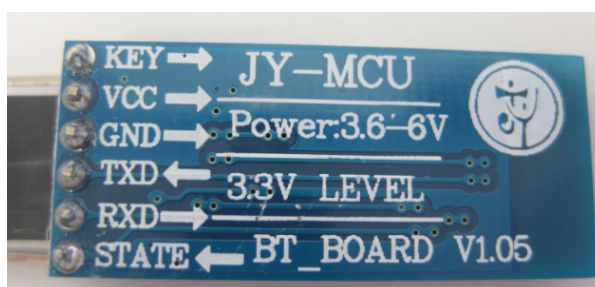


Сл. 1 Arduino Leonardo

Улазни напон је у границама од 7 до 20 V. Комуникација са рачунаром се одвија путем уграђеног USB прикључка, и рачунар га препознаје као HID уређај. USB протокол је интегрисан у микроконтролер што омогућава изостављање посебног кола за конверзију USB у TTL протокол. Максимална дозвољена јачина струје на дигиталним излазима је 40 mA, а на стабилном извору напона од 3.3 V је 50 mA. Физичке димензије развојне плоче су 68.6 mm x 53.3 mm.

2.1.2. Bluetooth модул

JY-MCU Bluetooth модул омогућава бежичну комуникацију између рачунара и робота. Базиран је на ревизији 2.0 Bluetooth стандарда. Брзина комуникације износи 9600 бита по секунди. Будући да је логичка јединица овог модула 3.3 V, а логичка јединица Arduino 5 V, потребно је користити регулатор напона који смањује 5 V на 3.3 V. Ова регулација напона је битна само између пина за слање података на Arduino и пина за примање података на Bluetooth модулу, пошто Arduino препознаје и 3.3 V као логичку јединицу. Модул користи профил серијске комуникације, тако да га рачунар види као COM порт коме је могуће приступити путем терминала, директним писањем на порт или путем различитих библиотека за програмске језике. У овом случају коришћена је PySerial библиотека за програмски језик Python 2.7. Распоред контакта модула може се видети на слици 2:



Сл. 2. JY-MCU Bluetooth модул

STATE – Контакт за читавање подешавања модула

RXD – Контакт за примање података са Arduino

TXD – Контакт за слање података на Arduino

GND – Уземљење модула

VCC – Извор напона за модул у границама од 3.6 до 6 V

KEY – Контакт за омогућавање подешавања модула путем AT команди

Извор напона овог модула је стабилних 5 V које пружа Arduino. Пинови који су искоришћени за Bluetooth модул су пинови 2 и 3, који су помоћу библиотеке SoftwareSerial за C++ Arduino платформе постављени да буду комуникациони пинови. Регулатор напона са 5 на 3.3 V је L4931 у TO92 паковању.

Модул поседује LE диоду путем које се може видети тренутно стање модула (Константно светли – модул је умрежен са другим уређајем, LED трепери – модул није умрежен). Модул има уграђену антену која му пружа домет до 10 метара од уређаја са којим је упарен.

2.1.3. Електро мотори и Н мост

Као погонски елементи, искоришћена су два DC мотора са четкицама и намонтираним редуктором са преносним односом од 120:1. Мотори раде у опсегу напона од 3 V до 12 V. У овом случају, мотори су повезани на четири AA батерије укупног напона 6 V. Мотори имају следеће карактеристике:

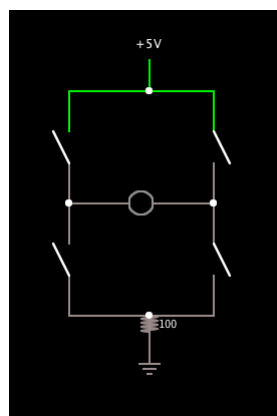
- Брзина без оптерећења 100 min^{-1} при напону од 4.5 V
- Јачина струје без оптерећења 200 mA при напону од 4.5 V
- Обртни момент при кочењу 2700 g*cm
- Јачина струје при кочењу 1250 mA

На вратило мотора прикачена су четири крака која заједно са оптичким интерраптером дају брзински енкодер који шаље четири импулса по пуном обртају.



Сл. 3. Изглед DC мотора са редуктором

Да би се заштитио Arduino од високе јачине струје, као и да би се обезбедила ротација мотора у оба правца, потребно је привезати мотор на Н мост [3]. Н мост је коло које се састоји од четири прекидача и мотора у средини распоређених тако да изгледом подсећају на латинично слово Н (Сл. 4). Уместо прекидача, за моторе мале снаге, се користе биполарни транзистори.

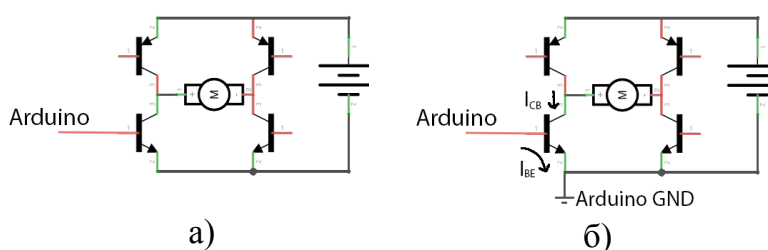


Сл. 4. Н мост

Прекидачи су:

- Високи леви
- Високи десни
- Ниски леви
- Ниски десни

На крајевима кола се налазе напонски изводи, са горње стране извор напона, а са доње стране уземљење. Будући да се за напајање мотора користи посебан извор напона, а не Arduino, потребно је спојити уземљење тог извора напона на уземљење (GND пин) Arduino и тиме добити заједничко уземљење. То је битно да би постојала затворена контура између сигнала са Arduino и Н моста, и тиме омогућио проток струје I_{BE} и I_{CB} кроз транзисторе Н моста (Сл. 6).



Сл. 6. а) Неправилно повезивање Н моста
б) Правилно повезивање Н моста

Принцип рада Н моста је веома једноставан: Сигнали се доводе на дијагоналне транзисторе, и тако се мења смер кретања електро мотора. У случају пропуштања струје кроз високи леви и ниски десни транзистор, мотор је директно поларисан и креће се у смеру казаљке на сату. У случају пропуштања струје кроз ниски леви и високи десни транзистор, мотор је инверзно поларисан и креће се у смеру супротном смеру казаљке на сату. У Таб. 1. исписана је таблица истинитости Н моста.

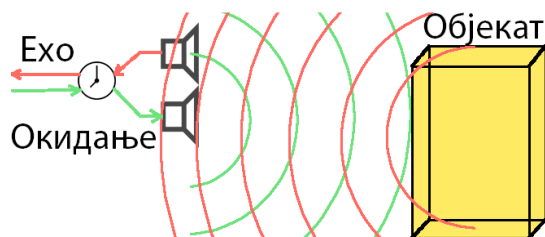
Таб. 1. Таблица истинитости Н моста

ВЛ	НЛ	ВД	НД	Мотор
1	0	0	1	Негативни мат. смер
0	1	1	0	Позитивни мат. смер
0	0	0	0	Слободан ход
0	1	0	1	Кочење
1	0	1	0	Кочење
1	1	0	0	Кратак спој
0	0	1	1	Кратак спој
1	1	1	1	Кратак спој

Коришћени Н мост у роботу је интегрисано коло ознаке L9110s.

2.1.4. Ултразвучни сензор растојања

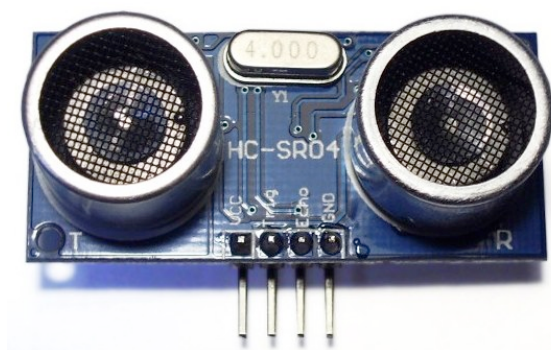
Удаљеност од објекта који се прати или од препреке робот одређује помоћу ултразвучног сензора растојања. Сензор ради по принципу сонара(Сл. 7): одашиље звук високе учестаности, мери време до повратка звука и, пошто је позната брзина звука, на основу времена које је било потребно да се звук врати одређује удаљеност сензора од објекта у који је уперен. Будући да се ради о звуку, најбољи резултати се добијају ако се мери удаљеност од предмета који се не понаша као звучни изолатор.



Сл. 7. Принцип рада сензора

Искоришћени сензор носи ознаку HC-SR04 (Сл. 8) и има следеће карактеристике:

- Мерни опсег од 2 cm до 4 m
- Прецизност 3 mm
- Угао мерења 15°
- Учестаност 40 kHz
- Димензије 40 mm* 20 mm * 15 mm

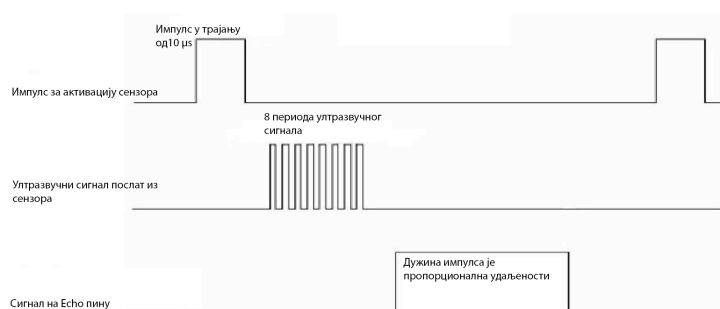


Сл. 8 HC-SR04 Ултразвучни сензор

Овај сензор поседује два пиезо трансдјусера, један да шаље, а други да прима сигнал. Од извода поседује:

VCC – Извор напона
TRIG – Улаз за окидање сензора
ECHO – Излаз
GND - Уземљење

Побуда сензора се врши тако што се на Trig пин пошаље импулс од 5 V у трајању од 10 μ s, после чега сензор емитује осам периода ултразвучног сигнала учестаности 40 kHz. По повратку сигнала, сензор шаље импулс на Echo пин чија је дужина једнака времену које је било потребно да се послат сигнал врати назад до пријемника (Сл. 9) која је пропорционална пређеном путу звучног таласа (растојању) који се може изразити помоћу формуле (1).



Сл. 9. Дијаграм побуде и одзива ултразвучног модула

Даљина се рачуна по формули:

$$l = \frac{t_{imp} * v_{zv}}{2} \quad (1)$$

Где је:

l – Дужина [cm]

t_{imp} – Време трајања повратног импулса [s]

v_{zv} – Брзина звука [$340 \frac{m}{s}$]

Звук прелази дупли пут при мерењу, до објекта према ком се мери даљина и назад до сензора, па самим тим је потребно поделити једначину за рачунање пређеног пута са два. Пауза између узастопног окидања сензора не сме бити мања од 60 ms, да не би дошло до интерференце између одлазног и долазног сигнала.

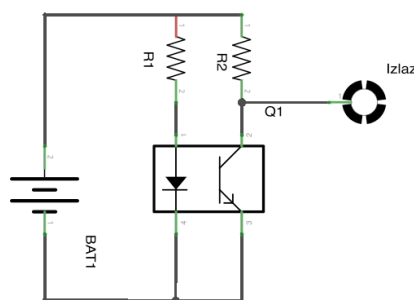
Да би се уштедела меморија на Arduino, не користе се специјалне библиотеке направљене за окидање и читавање вредности са сензора, већ се директно шаље логичка јединица у трајању од 10 μ s на сензор, и потом се помоћу команде PulseIn читава дужина трајања повратног импулса.

2.1.5. Фотоинтераптери

Фотоинтераптери су компоненте које у себи садрже извор светлости и фотоелемент који региструје светлост. Фотоелемент може бити фото-отпорник, фотодиода или фототранзистор. Због спорог реаговања и нелинеарне карактеристике, фото отпорници се готово не користе. Фотоинтераптери имају задатак да прекину или пропусте струју при наиласку светлости одређене таласне дужине која одговара извору светлости са фотоинтераптера. Као извор светлости се најчешће користе инфрацрвене LE диоде. Фотоинтераптери се постављају нормално на рефлектујућу површину, и мењају стање ако се нађе препрека између рефлектујуће површине и фотоинтераптера. Користе се у градњи енкодера тако што се ставе нормално на кружну површину на којој се налазе црне и беле траке. Пошто црна трака апсорбује светлост, при наиласку на исту, фотоинтераптер мења стање и шаље импулс. У овом раду искоришћен је ELITR 9904 фотоинтераптер са ИЦ LE диодом и фото транзистором, који заједно са четири крака међусобно распоређених на угаоном растојању од 90° на вратилу електро мотора чини оптички енкодер који даје четири импулса по ротацији електро мотора. Карактеристике овог фотоинтераптера су:

- Пробојни напон диоде 5 V
- Максимални напон колектор-емитер 30 V
- Снага дисипације 100 mW
- Време успона 15 μ s

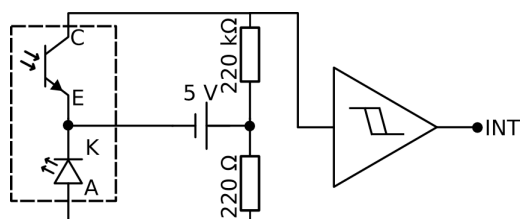
Колектор фототранзистора је прикључен на улазни пин Arduino, тако да фотоинтераптер шаље логичку јединицу када светлосни сигнал не пада на фототранзистор. Због електромагнетне индукције у водовима, потребно је искористити Pull-up отпорник. Вредност Pull-up отпорника у овом случају је 220 k Ω . Сигурносни отпорник на LE диоди је 100 Ω . Шема повезивања може се видети на слици 10.



Сл. 10. Шема везивања фото интераптера

Овакав начин повезивања на излазу даје аналогни сигнал чије максималне вредности нису увек једнаке, јер зависе од количине рефлектованог светла које долази до фототранзистора, а она зависи од врсте материјала

од које се светлост одбија, удаљености материјала, угла постављања и друго. Да би се тај проблем превазишао, и да би се на излазу добио чист дигитални сигнал искоришћено је инвертујуће Шмитово окидно коло које носи ознаку 40106. Ово коло поседује хистерезис од 0.8 V за напон улаза од 5 V, где је логичка нула све преко 3 V, а логичка јединица све испод 2.2 V. Укупно кашњење сигнала у колу износи 60 ns. Ово интегрисано коло поседује шест Шмитових окидних кола, од чега су искоришћена два, по једно за сваки опто-интераптер. Шема повезивања кола је приказана на слици 11.

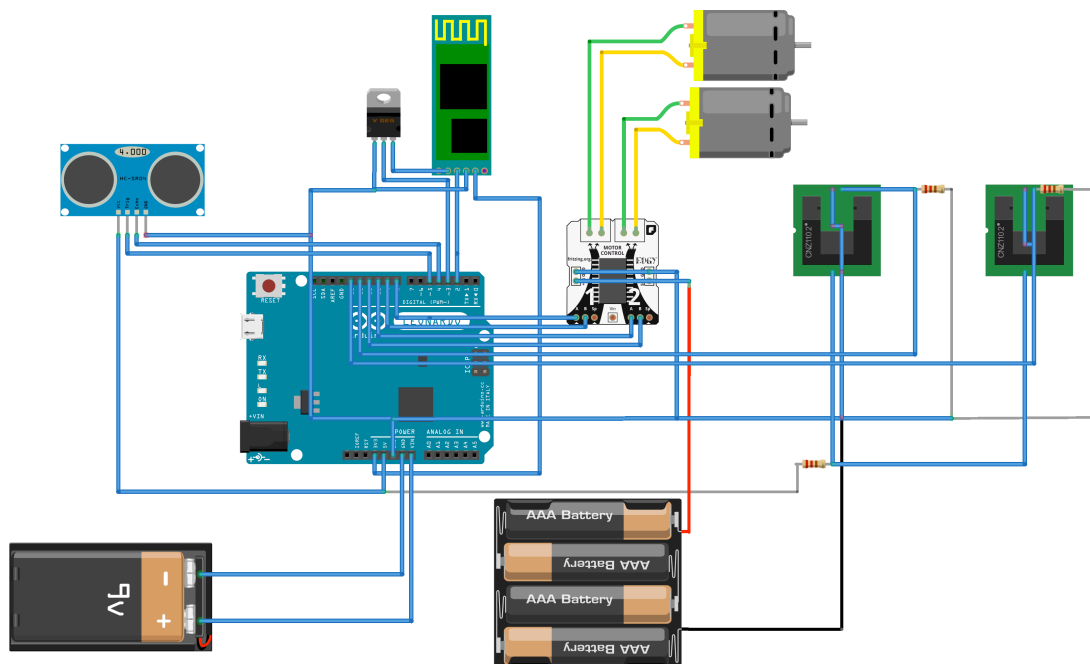


Сл. 11. Повезивање Шмитовог окидача

Будући да фотоинтераптер шаље четири импулса по једној ротацији електромотора, а преносни однос редуктора је 120:1, добија се да је могуће регистровати угаони померај точка од 0.75° , што је задовољавајућа прецизност за овакву намену.

2.1.6. Дијаграм повезивања електронских компоненти робота

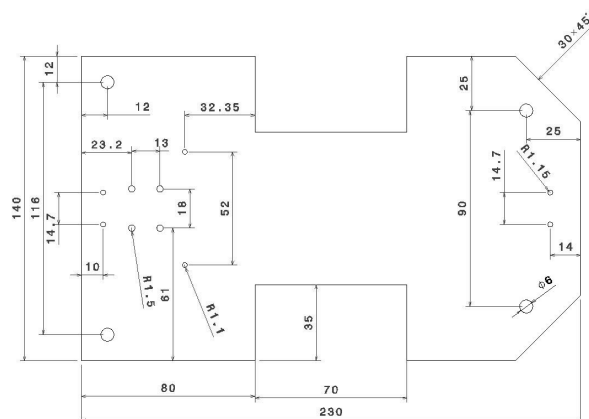
Горе наведене компоненте су повезане без лемљења, путем игличастих краткоспојника. Извори напона су 4 АА батерије укупног напона од 6 V, које су задужене за напајање Н моста и електро мотора, а једна батерија од 9 V је задужена да напаја Arduino и, преко Arduino, остале периферије. Bluetooth модул и ултразвучни сензор се напајају путем стабилних 5 V са Arduino, LE диода помоћу 3.3 V, а фототранзистор и Шмитово окидно коло са 5 волти. Свака компонента је прикачена на заједничко уземљење које се налази на Arduino плочи. Дијаграм повезивања може се видети на слици 12.



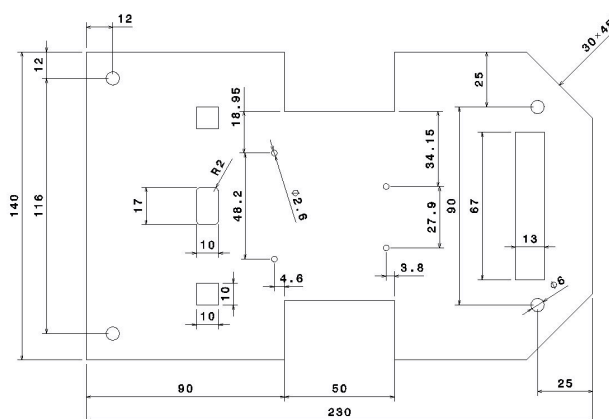
Сл. 12. Дијаграм повезивања електронских компоненти робота

2.2. Конструкција робота

Робот је састављен од шасије која има два нивоа. Доњи ниво (сл. 13) садржи моторе, опто интераптере, Н мост и батерије за напајање електромотора. Горњи ниво (сл. 14) носи Arduino развојну плочу, батерију за напајање Arduino, прото плочу на којој се налазе напонске магистрале, Шмитово окидно коло, Bluetooth модул и регулатор напона Bluetooth модула. Такође, горњи ниво поседује отвор за постављање мобилног телефона. Погонски точкови су смештени на средини доње плоче, и имају пречник 66 mm. На предњем и задњем крају робота налазе се Ball caster-и који представљају сфере које се могу ротирати по све три осе. Горњи и доњи ниво су спојени са четири навоја Ф6. Мотори се налазе у сендвичу између два слоја плексигласа на које су причвршћени лепком и навојем Ф6.



Сл. 13. Доњи ниво шасије

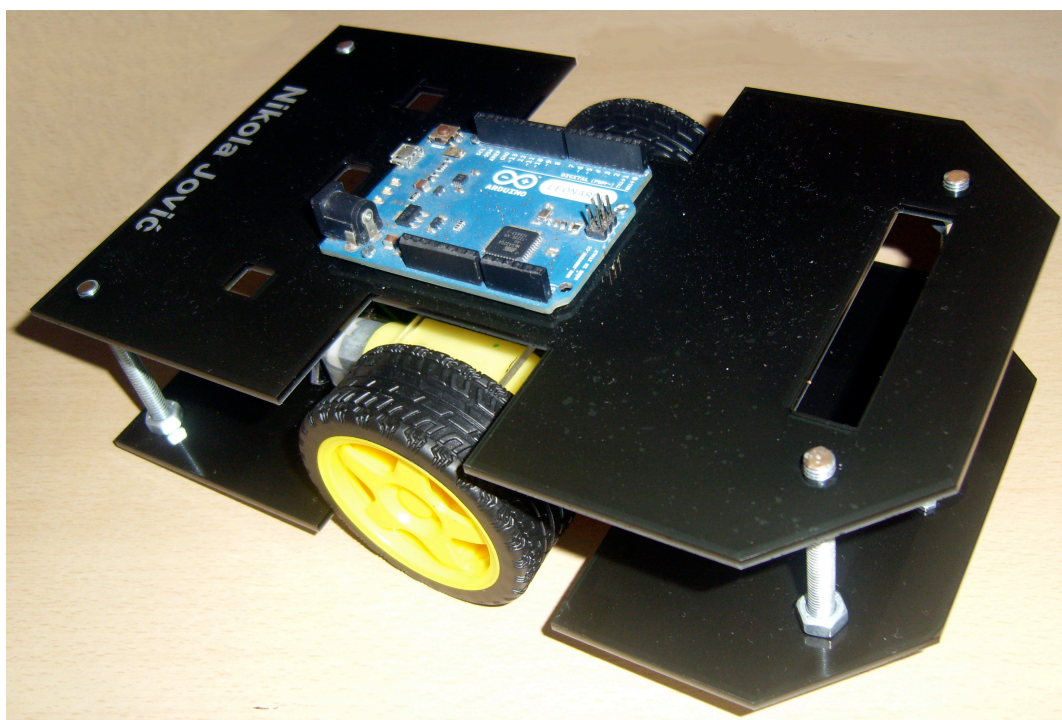


Сл. 14. Горњи ниво шасије

На слици 15 приказан је тродимензиони модел робота са постављеним мобилним телефоном, на слици 16 се виде делови и готов производ.



Сл. 15. 3Д модел робота

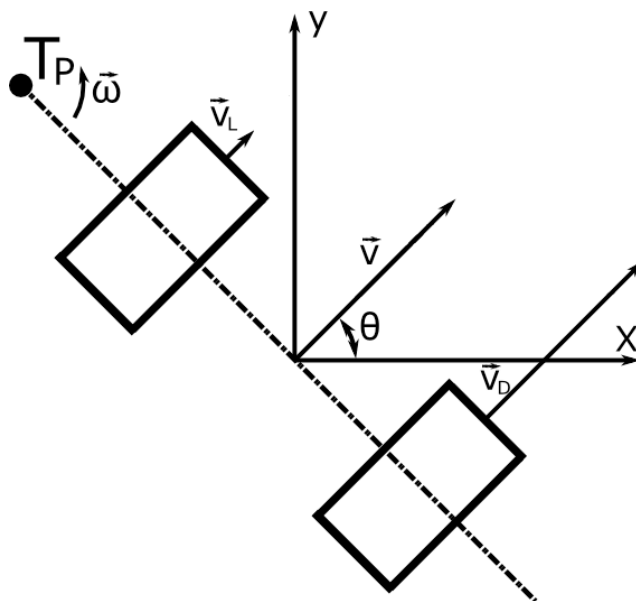


Сл. 16. Састављен робот

Кроз централни отвор на горњем нивоу иду каблови за контролу мотора, каблови за напајање мотора и каблови за напајање опто интерактера. За сигналне каблове опто интерактера предвиђена су два отвора поред централног да не би долазило до интерференце између каблова за напајање мотора и сигнала који шаљу опто интерактери. Целокупна шасија је направљена од плексигласа на CNC ласерској глодалици.

2.3. Кинематика робота

Робот је конструисан по принципу диференцијалног погона, што значи да поседује два независна точка постављена на средини робота кроз које пролази заједничка оса. Овакав вид кретања је најједноставнији за реализацију. Задавањем одређених брзина моторима може се управљати трајекторијом робота. У сваком временском тренутку, постоји тачка око које се робот ротира и постоји угаона брзина ротације робота. Тачка око које се робот ротира назива се тренутни пол (Сл. 17).



Сл. 17. Раванско кретање робота (није у размери)

Из једначина:

$$\omega \left(R + \frac{l}{2} \right) = v_d \quad (2) \text{ и}$$

$$\omega \left(R - \frac{l}{2} \right) = v_l \quad (3)$$

где су:

$\frac{l}{2}$ – дужина полуосовине точка

R – растојање центра од тренутног пола

ω – угаона брзина робота

могу да се израчунају тренутне брзине левог и десног точка. Ако су брзине левог и десног точка познате, путем једначина (4) и (5) могу се израчунати растојање тренутног пола од центра робота и угаона брзина робота, респективно.

$$R = \frac{l (v_l + v_d)}{2 (v_d - v_l)} \quad (4)$$

$$\omega = \frac{v_d - v_l}{l} \quad (5)$$

Из једначина (4) и (5) могу да се закључе два интересантна случаја. Ако је $v_d = v_l$, тренутни пол се налази на бесконачном растојању од центра робота, а угаона брзина је једнака нули, што значи да робот врши праволинијско кретање. Други случај представља $v_d = -v_l$, где је растојање тренутног пола од центра једнако нули, и робот се ротира у месту. За све остале случајеве робот прати криву чији се центар налази у тренутном полу, мењајући притом и позицију и ротацију робота. Овакав вид управљања роботом има један недостатак, а то је да ни једном комбинацијом брзина точкова робот не може да се креће по заједничкој оси точкова. Возило са диференцијалним погоном је веома осетљиво на разлику брзина точкова, тако да и мала разлика може проузроковати неправилно кретање робота, што ограничава употребу оваквих робота.

2.3.1. Рачунање позиције робота у глобалном координатном систему

Пошто су пречници точкова познати, бројањем обртаја точкова може се одредити брзина робота. Апсолутна позиција робота може се одредити интегралњем добијених брзина робота. Процес одређивања локације у простору без знања о спољашњем свету се назива *Dead reckoning*. Да би се израчунала, на пример, позиција по x оси у идеалној ситуацији помоћу вектора брзине $\frac{d\vec{x}}{dt}$ користи се формула:

$$x = \int_{t_0}^{t_k} \frac{dx}{dt} dt \quad (6)$$

где је t_0 тренутак почетка кретања, а t_k тренутак краја кретања. Ако су познати виши изводи позиције по времену (нпр. убрзање), потребно је интегралити више пута да би се добила позиција. Што је извод већег реда, грешка при интегралњењу расте. За дискретне системе, као што је робот, за добијање позиције користи се формула 7.

$$x = \sum_{i=0}^n \delta_i \quad (7)$$

Где је δ_i вектор разлике положаја. Овај метод има задовољавајућу прецизност ако су кораци довољно мали, и ако се робот налази на одговарајућој подлози на којој неће долазити до проклизавања.

2.3.2. Директна кинематика робота

Са директном кинематиком може да се одреди крајња тачка у просору где ће робот доћи са постављеним параметрима v_d и v_l . Пошто су v_d и v_l , као и ω и R функције од времена, могуће је показати да ако робот заузима неки положај (x, y, θ) у тренутку t , и ако су познате величине v_d и v_l током периода $t \rightarrow t + \delta t$ да је тренутни пол одређен са формулом:

$$T_p = (x - R \sin(\theta), y + R \cos(\theta)) \quad (8)$$

и у тренутку $t + \delta t$ положај робота је одређен са (9).

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega \delta t) & -\sin(\omega \delta t) & 0 \\ \sin(\omega \delta t) & \cos(\omega \delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - T_{px} \\ y - T_{py} \\ \theta \end{bmatrix} + \begin{bmatrix} T_{px} \\ T_{py} \\ \omega \delta t \end{bmatrix} \quad (9)$$

Ова једначина описује ротацију робота око тренутног пола на растојању R угаоном брзином ω . Интеграљењем ове једначине са почетним условима (x_0, y_0, θ_0) могуће је одредити положај робота у сваком тренутку узимајући у обзир брзине тачкова. Положај робота који се креће под углом $\theta(t)$ брзином $V(t)$ одређен је једначинама:

$$\begin{aligned} x(t) &= \int_0^t V(t) \cos(\theta(t)) dt \\ y(t) &= \int_0^t V(t) \sin(\theta(t)) dt \\ \theta(t) &= \int_0^t \omega(t) dt \end{aligned} \quad (10)$$

За случајеве где је $v_d \neq v_l$:

$$\begin{aligned} x(t) &= \frac{1}{2} \int_0^t (v_r(t) + v_l(t)) \cos(\theta(t)) dt \\ y(t) &= \frac{1}{2} \int_0^t (v_r(t) + v_l(t)) \sin(\theta(t)) dt \\ \theta(t) &= \frac{1}{l} \int_0^t (v_r(t) - v_l(t)) dt \end{aligned} \quad (11)$$

2.3.3. Инверзна кинематика робота

У већини случајева погодније је задати крајњи положај робота, а онда израчунати брзине да би робот дотле дошао. Овакав приступ се назива инверзна кинематика. Приступ се састоји у томе да се једначине (11) преиначе тако да за се за дату позицију (x, y, θ) добију брзине v_d и v_l . Добијене једначине имају решење само у неким случајевима као, на пример, када су v_d и v_l константне и различите. У том случају, добијају се једначине (12).

$$\begin{aligned}x(t) &= \frac{l}{2} \frac{v_d + v_l}{v_d - v_l} \sin\left(\frac{t}{l}(v_d - v_l)\right) \\y(t) &= -\frac{l}{2} \frac{v_d + v_l}{v_d - v_l} \cos\left(\frac{t}{l}(v_d - v_l)\right) \\ \theta(t) &= \frac{t}{l}(v_d - v_l)\end{aligned}\quad (12)$$

где је $(x, y, \theta)_{t=0} = (0,0,0)$. Када се задају параметри x, y, t , врло лако је могуће израчунати v_d и v_l , али не постоји независна контрола над параметром θ . Уместо да се једначине (11) инверују, погодније решење је проблем поделити у неколико делова. Посматрајмо случај када робот врши праволинијско кретање ($v_d = v_l$). У том случају, једначине кретања робота се упрошћавају на:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + v \cos(\theta) \delta t \\ y + v \sin(\theta) \delta t \\ \theta \end{bmatrix} \quad (13)$$

Други случај је да робот врши ротацију око своје осе ($v_d = -v_l = v$). Тада се једначине кретања робота редукују на:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta + 2v \frac{\delta t}{l} \end{bmatrix} \quad (14)$$

Да би робот заузео неку позицију (x, y, θ) у неком тренутку t , могуће је прво ротирати робота у месту тако да гледа на тачку (x, y) , затим праволинијским кретањем треба да дође на ту тачку и, на крају, треба да се ротира у месту тако да заузме положај (x, y, θ) . [4]

2.4. Карактеристике сензора и актуатора робота

2.4.1. Статичке карактеристике мотора

Карактеристике мотора су измерене тако што су се моторима задавали различити ефективни напони путем модулације ширине импулса у границама од 0 V до 6 V. Тајмер задужен за модулацију ширине импулса је осмобитни, што значи да постоји укупно 256 прагова (0 – 255). Фреквенција износи 490 Hz. Свака промена напона је била за пет прагова већа од предходне, односно за пет прагова мања од претходне за добијање силазне карактеристике. Извршено је скалирање тако да x оса показује проценте по формули (15), док y оса показује број обртаја у минути по формули (16).

$$Snaga[\%] = \frac{n}{255} * 100 \quad (15)$$

$$\omega[min^{-1}] = \frac{\Delta n_{enk}}{4 * \Delta t} * 0.5 \quad (16)$$

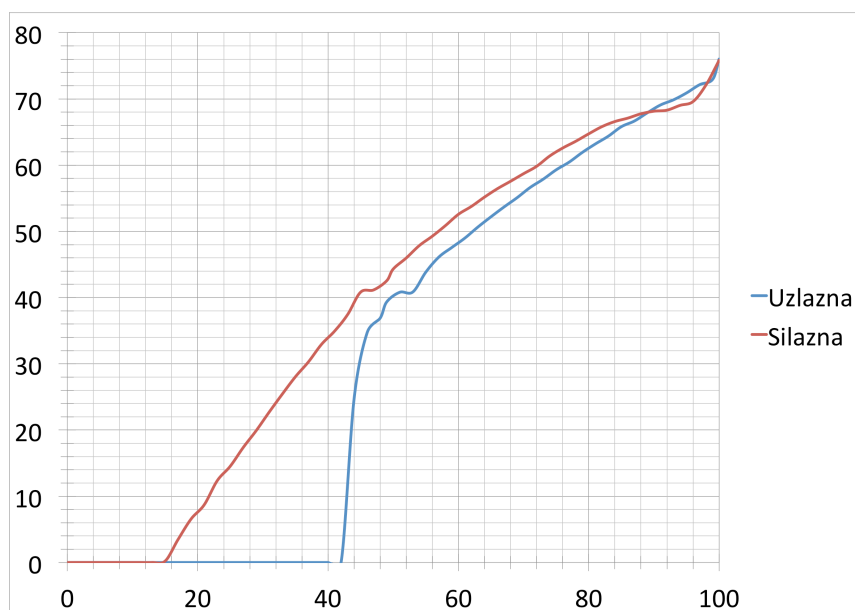
Где је:

n – n -ти праг

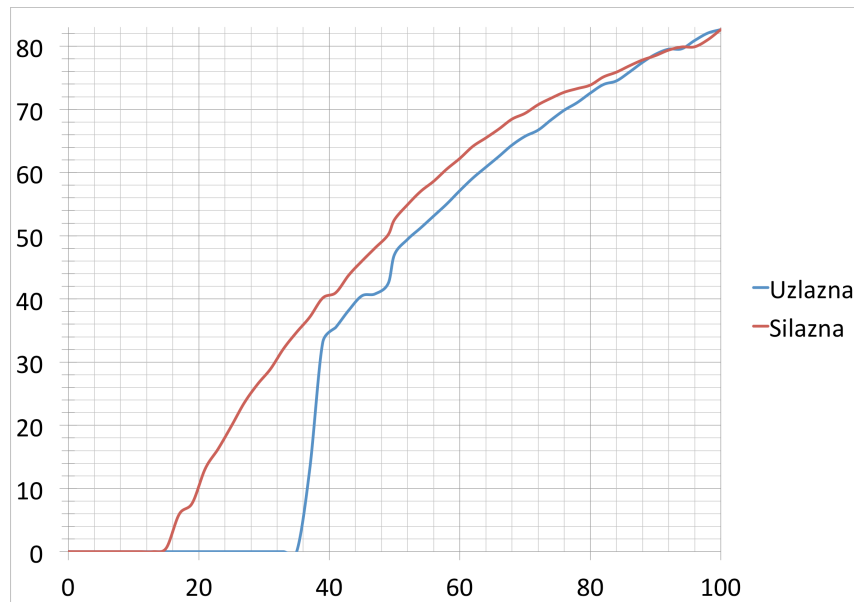
Δn_{enk} – број импулса са енкодера током временског периода Δt

Једначина (16) се дели са 4 зато што енкодер има четири крака, да би се добила пуна ротација мотора, а множи се са 0.5, јер је преносни однос редуктора 1:120, тако да $\frac{\Delta n_{enk}}{120 * 4 * \Delta t}$ представља број обртаја по секунди, а све то помножено са 60 представља број обртаја по минути.

На слици 18 се може видети статичка карактеристика левог мотора, док на слици 19 се може видети статичка карактеристика десног мотора.

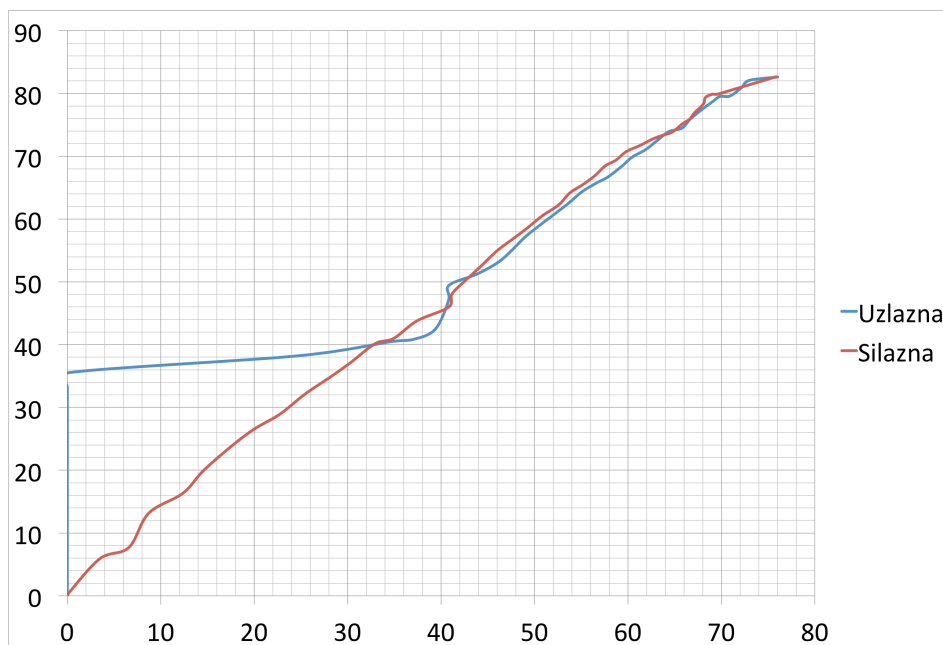


Сл. 18. Статичка карактеристика левог мотора



Сл. 19. Статичка карактеристика десног мотора

Оба мотора имају мртву зону до 40% укупног PWM опсега, мали хистерезис при високим брзинама који се повећава како се брзине снижавају. На вишим бројевима обртаја, поготово од 50 до 70 обртаја по минути, мотори имају релативно линеарну карактеристику, тако да ће се радна тачка управо налазити између тих брзина. На слици 20 је приказана разлика у брзинама мотора при истим вредностима улаза.



Сл. 20. Разлика брзина мотора при истим улазним вредностима

Експеримент мерења брзине је рађен тако што се после сваке промене улазних вредности чекало да мотор уђе у стационарно стање, и потом се бележила тренутна брзина мотора.

2.5. Алгоритми и програми за управљање роботом

2.5.1. Навигација помоћу QR кодова

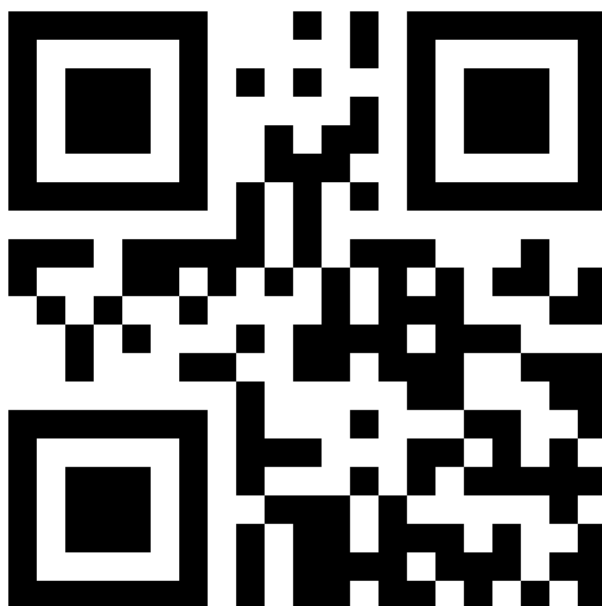
QR код представља скраћеницу за Quick Response (*енгл. брзи одзив*) код. Првенствено је коришћен у Toyota за праћење делова у производњи возила, док се сада користи у много ширем контексту. QR кодови имају следећи капацитет података:

- Само нумерички подаци:	Максимално 7089 карактера
- Алфанумерички подаци:	Максимално 4296 карактера
- Бинарни подаци:	Максимално 2953 бајтова
- Канџи/Кана:	Максимално 1817 карактера

QR кодови користе Рид-Соломонов код за исправљање грешака. По капацитету исправке грешака QR кодови се деле у четири нивоа[5]:

- Ниво L:	Исправља до 7% грешака
- Ниво M:	Исправља до 15% грешака
- Ниво Q:	Исправља до 25% грешака
- Ниво H:	Исправља до 30% грешака

За контролу робота користи се QR код који у себи има укупно две речи. Прва реч представља смер ротације робота за 90 °, а друга реч представља растојање од QR кода на којем робот треба да се заустави и обави ротацију. На слици 21 је приказан пример QR кода који шаље роботу наредбу да се заустави на 20 cm од QR кода и да се ротира у негативном математичком смеру (DESNO 20).



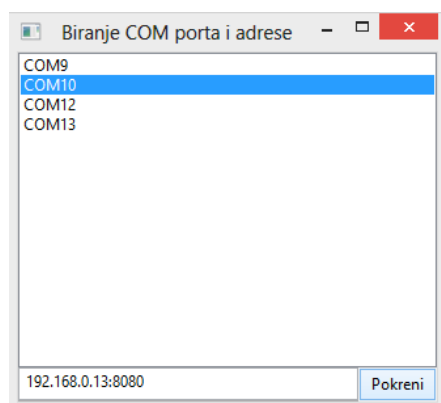
Сл. 21. Пример QR кода

Читљивост QR кода је највише условљена резолуцијом камере. Мобилни телефон шаље mJpeg видео стрим резолуције 320 x 240 пиксела због што бржег преноса, што може представљати проблем приликом читања QR кодова. Из тог разлога, потребно је имати што мање података на QR коду да би ћелије на QR коду биле ређе. Такође, због могуће удаљености робота од QR кода, потребно је да он буде довољно крупан да би се уочио са растојања од једног метра.

Апликација за препознавање QR кодова и за слање одређених команди роботу је написана у програмском језику Python 2.7. Коришћене библиотеке су:

- SimpleCV Библиотека задужена за компјутерску визију
- PySerial Библиотека задужена за серијску комуникацију
- wxPython Библиотека задужена за графичи интерфејс

На слици 22 је приказан дијалог за одабир одговарајућег серијског порта и за уношење адресе ИП камере, док је на слици 23 приказан главни прозор програма за навигацију путем QR кодова.



Сл. 22. Дијалог за одабир COM порта и ИП адресе



Сл. 23. Главни прозор апликације

Главни прозор приказује више информација као што су тренутна слика са ИП камере, позицију QR кода, као четири реда текстуалних информација. У првом реду се налази садржај QR кода који је зелене боје ако је QR код тренутно видљив, жуте боје ако QR код није тренутно видљив (садржај задњег виђеног QR кода), или обавештење црвене боје ако још није детектован ни један QR код. У другом реду се налази адреса ИП камере, у трећем серијски порт Bluetooth модула, а у четвртном се налази информација о активним битовима задњег пакета који је послат. Сваки пакет је величине једног бајта, где пет битова са најмањом тежином представљају растојање на којем робот треба да се заустави, док прва три бита носе информацију о окретању робота (бит са највећом тежином – окрет на десно, други бит – окрет на лево, трећи бит – потпуно заустављање). Изворни код апликације је дат испод.

```

import wx
import serial.tools.list_ports
# begin wxGlade: dependencies
import gettext
# end wxGlade
import SimpleCV
from SimpleCV import Barcode, Color, Camera
# begin wxGlade: extracode
# end wxGlade

class MyFrame1(wx.Frame):
    def __init__(self, *args, **kwargs):
        # begin wxGlade: MyFrame1.__init__
        kwargs["style"] = wx.DEFAULT_FRAME_STYLE
        wx.Frame.__init__(self, *args, **kwargs)
        self.list_box_1 = wx.ListBox(self, wx.ID_ANY,
choices=[])
        self.text_ctrl_1 = wx.TextCtrl(self, wx.ID_ANY,
_("192.168.0.13:8080"))
        self.button_1 = wx.Button(self, wx.ID_ANY,
_("Pokreni"))
        self.button_1.Bind(wx.EVT_LEFT_UP, self.pokreni)
        self.__set_properties()
        self.__do_layout()
        # end wxGlade

    def __set_properties(self):
        # begin wxGlade: MyFrame1.__set_properties
        self.SetTitle(_("Biranje COM porta i adrese"))
        self.list_box_1.SetMinSize((320, 240))
        self.text_ctrl_1.SetMinSize((260, 25))
        self.button_1.SetMinSize((60, -1))
        # end wxGlade

    def __do_layout(self):
        # begin wxGlade: MyFrame1.__do_layout
        sizer_3 = wx.FlexGridSizer(2, 1, 0, 0)
        sizer_4 = wx.FlexGridSizer(1, 2, 0, 0)
        sizer_3.Add(self.list_box_1, 0, 0, 0)
        for port in serial.tools.list_ports.comports():
            self.list_box_1.Append(port[0])
        sizer_4.Add(self.text_ctrl_1, 0, 0, 0)
        sizer_4.Add(self.button_1, 0, 0, 0)
        sizer_3.Add(sizer_4, 1, wx.EXPAND, 0)
        self.SetSizer(sizer_3)
        sizer_3.Fit(self)
        self.Layout()
        # end wxGlade

    def pokreni(self, event):
        QR(self.list_box_1.GetStringSelection(),
self.text_ctrl_1.GetValue())
# end of class MyFrame1

```

```

class MyApp(wx.App):
    def OnInit(self):
        wx.InitAllImageHandlers()
        frame_2 = MyFrame1(None, wx.ID_ANY, "")
        self.SetTopWindow(frame_2)
        frame_2.Show()
        return 1
# end of class MyApp
def QR(com, adresa):
    disp = SimpleCV.Display(title="QR Navigacija")
    cam = SimpleCV.JpegStreamCamera(adresa+"/video")
    zadnjiqr = None #Cuva vrednost zadnjeg prepoznatog
QR koda
    vidi = False #Ako je QR kod trenutno vidljiv,
ispisuje sadrzaj QR-a zelenom bojom, u suprotno zutom
    ser = None #Port za bluetooth
    byte = 0 #Ovde se cuva vrednost sta se salje
    Arduinu
    p = []
    try:
        ser = serial.Serial(com)
    except:
        pass
    try:
        while disp.isNotDone:
            i = cam.getImage()
            bc = i.findBarcode()
            if bc:
                if str(bc[-1].data) != zadnjiqr :
                    zadnjiqr = str(bc[-1].data)
                    parse = zadnjiqr.split(" ")
                    if parse[0] == 'LEVO':
                        byte = 64
                    elif parse[0] == 'DESN0':
                        byte = 128
                    else:
                        byte = 32
                    byte = byte | int(parse[1])
                    if ser:
                        ser.write(chr(byte))
                        print bin(byte)
                    bc.draw(width=2, )
                    vidi = SimpleCV.Color.GREEN
            else:
                vidi = SimpleCV.Color.YELLOW
            if zadnjiqr is not None:
                i.drawText(zadnjiqr, 0, 0, vidi)
            else:

```

```

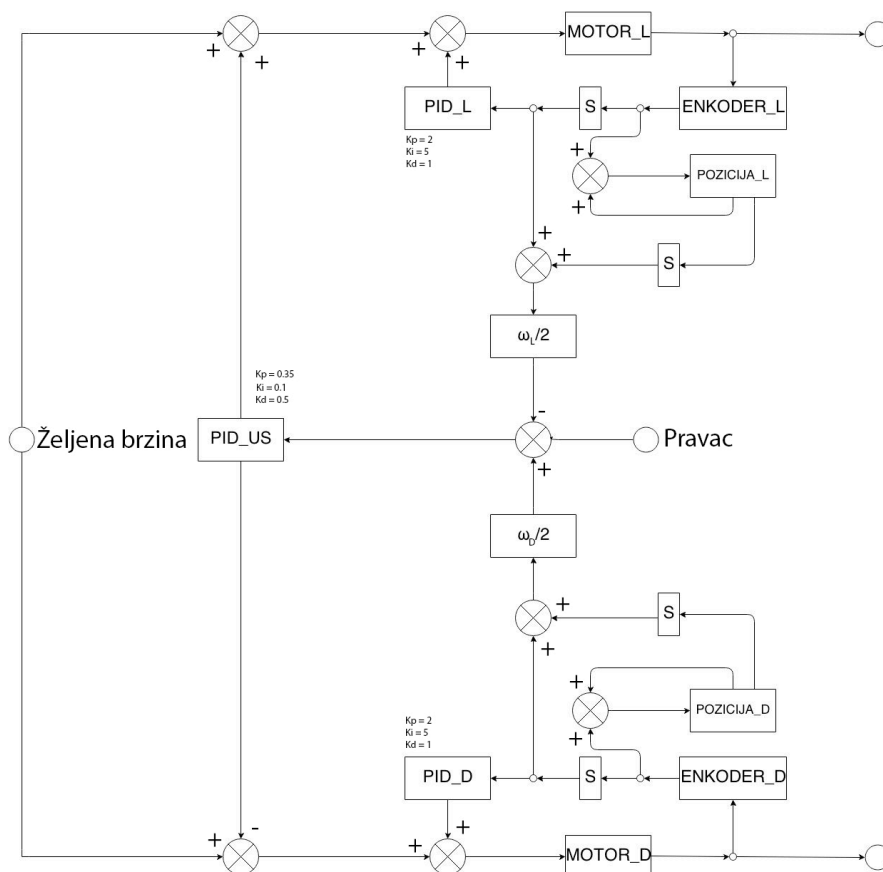
        i.drawText("QR kod nije pronadjen", 0,
0, SimpleCV.Color.RED)
        i.drawText(text=("WiFi adresa: " + adresa),
x=0, y=20)
        i.drawText(text="Bluetooth adresa: " + com,
x=0, y=40)
        for (k, j) in zip(reversed([2**n for n in
range(8)]), str(bin(byte)[2:].zfill(8))):
            p.append(str(int(k) * int(j)))
            p.append(" ")
        s = ''.join(p)
        i.drawText(text=(s), x=0, y=60)
        p = []
        i.save(dis)
        if disp.mouseLeft:
            break
    except KeyboardInterrupt:
        pass
if __name__ == "__main__":
    gettext.install("app")
    app = MyApp(0)
    app.MainLoop()

```

Класе MyFrame1 и MyApp су направљене аутоматски од стране програма wxGlade, који је коришћен за израду графичког интерфејса. Функција `rokreni` у класи MyFrame1 је задужена за покретање потпрограма задуженог за препознавање QR кодова и за комуникацију са роботом. Параметри који се прослеђују тој функцији су ИП адреса камере и КОМ порт на коме се налази Bluetooth модул. Комуникација са Bluetooth модулом је серијска брзине 9600 бауда. Као што је речено, сваки пакет је ширине једног бајта. Максимално растојање на коме робот може да се заустави од препреке износи 31 cm. Пакет за слање се формира тако што се чита други податак у QR коду који представља растојање, претвара се у целобројни тип података. Затим се, у зависности од првог податка у QR коду, врши логичка операција ИЛИ са променљивом где је складиштено растојање, који јој додаје 128, 64 или 32. Тако формиран бајт се шаље преко серијског порта. Пре слања, податак мора да се претвори у карактер. Овакав метод слања је знатно бржи од слања целобројног типа преко серијског порта, и може знатно да уштеди процесорско време на Arduino приликом декодирања примљеног пакета. Пакет се шаље само једном када се QR код детектује, а не узастопно, да би се умањио промет података, као и да не би дошло до забуне приликом ротације робота када може доћи до случаја да се виде два QR кода.

2.5.2. Управљачки програм робота за микроконтролер Atmel AtMega 32u4

Управљачки програм робота садржи у себи рутине за праволинијско кретање и ротацију робота, аквизицију података са ултразвучног сензора растојања серијски интерфејс за комуникацију помоћу Bluetooth модула за управљање и USB порта за дијагностику. Програм је писан у програмском језику C++ намењеном Ателовим микроконтролерима. Коришћен компјалер је GCC, а развојно окружење Arduino IDE. Контрола мотора за праволинијско кретање је направљена помоћу два одвојена PID контролера, где се као повратна спрега користе енкодери направљени од фотоинтераптера постављених испред вратила мотора. Да би се осигурало праволинијско кретање робота, обављено је унакрсно спрезање са још једним PID контролером, чији је улаз разлика позиције између два мотора [6]. На слици 24 може се видети блок дијаграм система за праволинијско кретање.



Сл. 24. Блок дијаграм система за праволинијско кретање

Улазне величине су жељена брзина и правац. Изабрана жељена брзина је 30 обртаја у минути. О правцу ће бити речи касније. Излазне величине су угаони померај мотора. PID контролер за синхронизацију мотора мора да ради у обрнутом режиму (што је већа грешка, излаз контролера је већи), јер се грешка синхронизације одузима од правца.

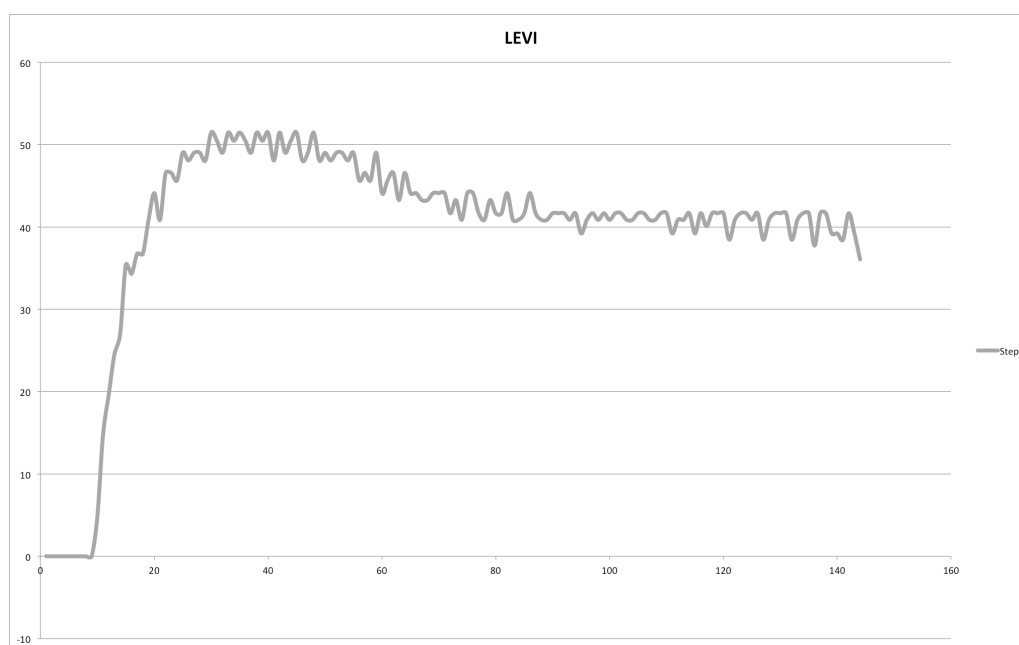
Параметри PID контролера су подешени по следећој процедури [7]:

- Одабрати радну тачку и повећавати K_P док систем не почне да осцилује
- Поделити K_P са 2
- Спуштати и подизати радну тачку за 5% и притом повећавати K_D док се не дође до пригушења осцилација
- Полако повећавати K_I док систем не почне да осцилује
- Поделити K_I са два или три
- Проверити перформансе контролера, ако нису задовољавајуће, вратити се на корак 1

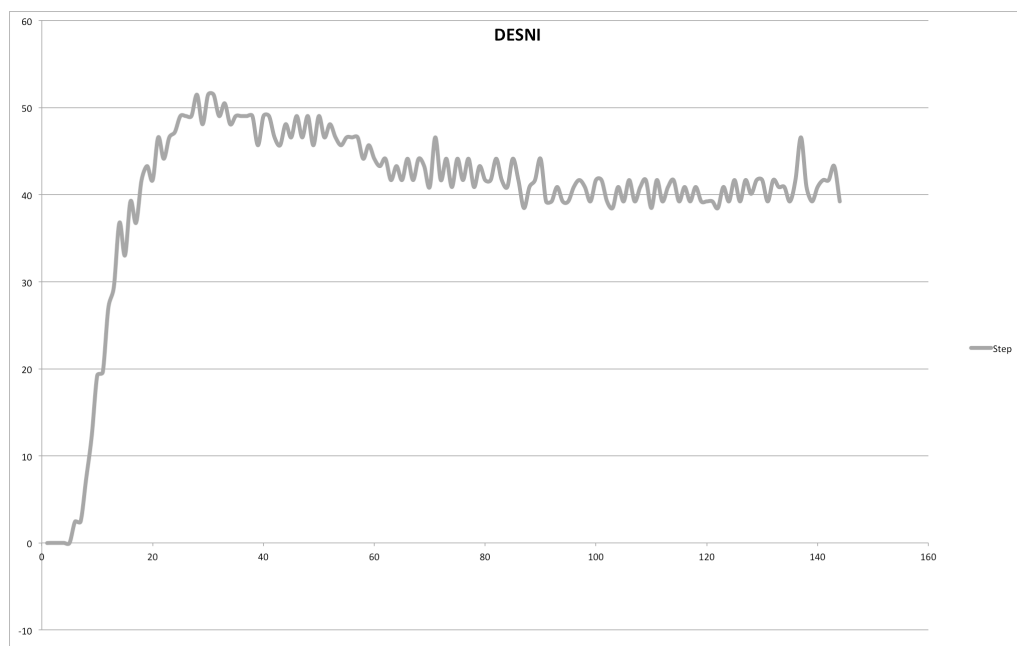
Оваквом методом су добијени следећи параметри:

- PID за моторе: $K_P = 0.4$, $K_I = 0.2$, $K_D = 0.1$
- PID за унакрсно спрежање: $K_P = 0.6$, $K_I = 0.3$, $K_D = 0.15$

Пошто понашање система зависи од напона батерија, при полу празним батеријама робот више не иде право, већ заноси у једну страну. Тај проблем се решава уз помоћ променљиве за правац, путем које робот фаворизује један мотор наспрам другог. Овакав начин спрежања мотора елиминише грешке синхронизације настале трењем у редуктору мотора и кретањем мотора у супротном правцу од предвиђеног, али има мали утицај на грешке настале проклизавањем точка због лоше подлоге. Мерењем је утврђено да грешка синхронизације у идеалним условима не прелази 1.5° (пола обртаја мотора). На слици 25 и 26 се могу видети одзиви мотора са PID контролерима на одскочни сигнал.



Сл. 25. Одзив левог серво мотора (Задата брзина 40 min^{-1})



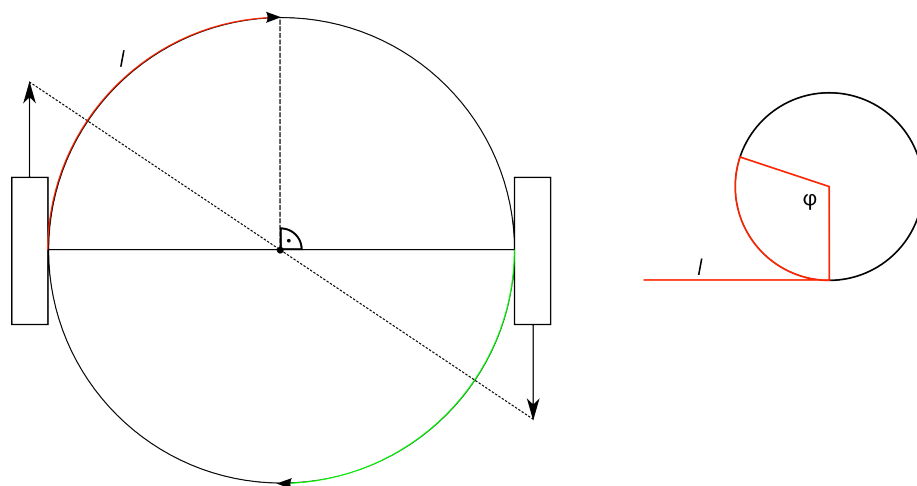
Сл. 26. Одзив десног серво мотора (Задата брзина 40 min^{-1})

За ротацију од 90° се користи алгоритам налик контролерима у малим DC сервомоторима. По доласку до жељеног угла, мотор се искључује. Угао заокретања точка се може наћи ако је позната дужина осовине између два точка и радијус точкава. Прво је потребно наћи дужину кружног исечка од 90° чији је полупречник полупречник осовине (17), затим је потребно наћи угао кружног исечка точка исте дужине (сл. 27). На крају, добијен угао треба поделити са 0.75 да би се добио број импулса које је потребно остварити да би се дошло до тог угла (18).

$$l = \frac{\alpha}{360} * 2\pi r \quad (17)$$

$$\varphi = \frac{l}{2r\pi * 0.75} * 360 \quad (18)$$

За ротацију од 90° , пречником осовине 8 cm, и пречником точка од 6.6 cm потребно је 145 импулса [8].



Сл. 27. Дијаграм окретања робота око нормалне осе

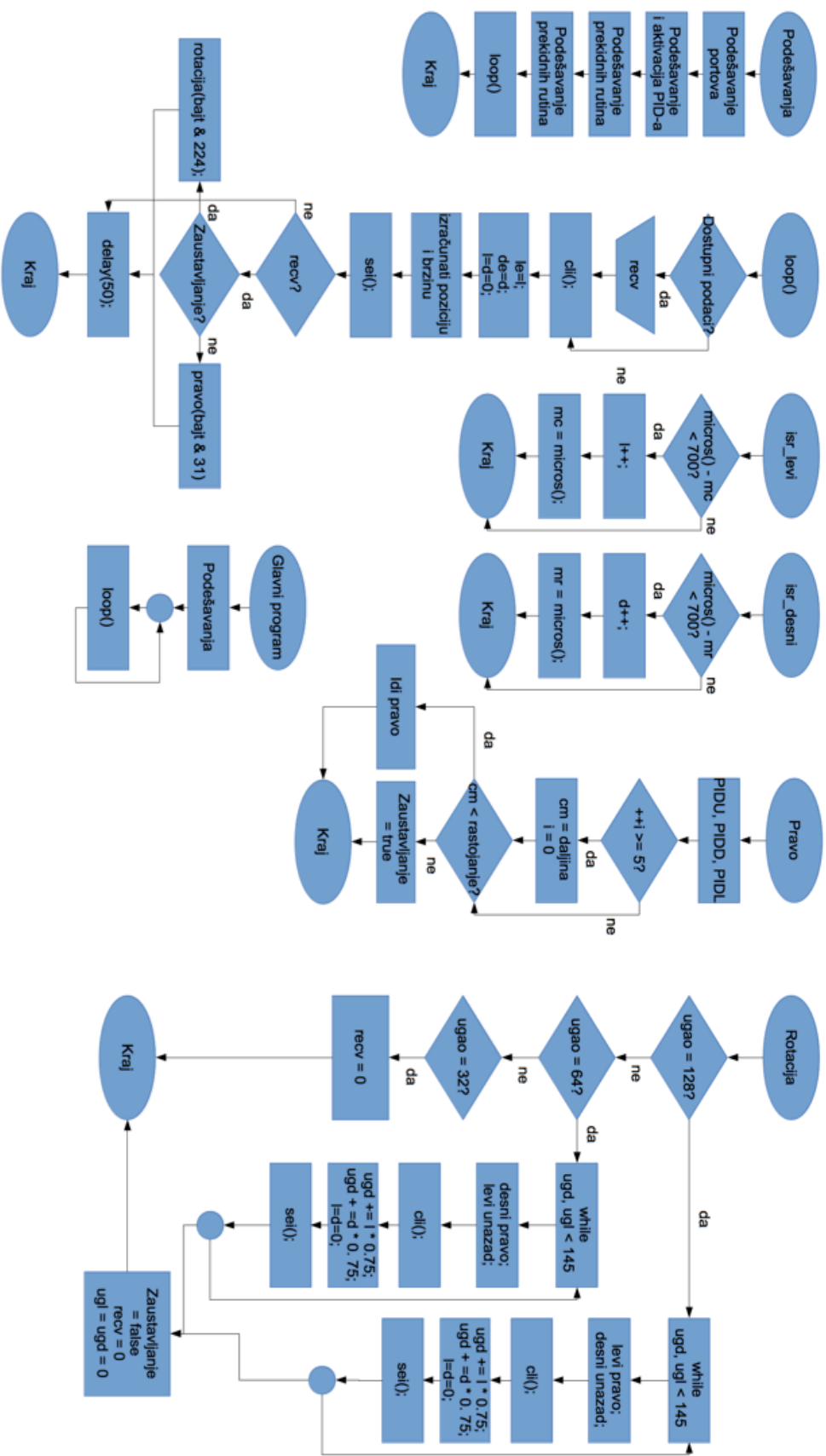
Комуникација са рачунаром се остварује помоћу Bluetooth модула, где је ширина једног пакета 8 бита (један бајт). Нижих 5 бита носе информацију о растојању на којем се робот зауставља, а највиша три бита носе информацију о ротацији. Подаци се из пакета ваде помоћу логичке операције “И” на следећи начин:

- растојање = пакет & 00011111
- ротација = пакет & 11100000

Постоје три случаја ротације:

- За вредност 128 робот се ротира на десно
- За вредност 64 робот се ротира на лево
- За вредност 32 робот стаје

Ротација се обавља пошто робот дође на одређено растојање од QR кода. Потпрограм за ротацију је блокирајући, што значи да неће да се врати у главну петљу све док се ротација не изврши. Потпрограм за кретање у напред није блокирајући. За праћење стања кретања, уведена је заставица *Zaustavljanje* која даје до знања да ли је робот стигао до циља и да ли сме да започне ротацију. Да не би долазило до грешке, растојање се проверава једном у пет циклуса, где један циклус траје 50 милисекунди. Путем USB комуникације могуће је пратити стање било ког система робота. Најинтересантније је пратити грешку синхронизације и тренутни пакет, тако да је то остављено у главном програму. У случају да је тренутни пакет једнак нули, робот се зауставља. Ово је искоришћено за регулацију кретања после ротације. После ротације, заставица *Zaustavljanje* се спушта, па робот наставља праволинијско кретање, што не би требало да се дешава. Због тога, по обављеној ротацији, тренутни пакет се изједначава са нулом и робот поново креће тек када детектује следећи QR код. Ради дијагностике, ЛЕ диода која се налази на Arduino плочи се укључује када се започиње ротација, а искључује при праволинијском кретању. Библиотеке коришћене при изради овог робота су PID_V1 за PID контролере и SoftwareSerial за комуникацију путем Bluetooth модула. У потпрограму за подешавање микроконтролера налазе се информације о портovima (улази и излази), иницијализира се серијска USB и Bluetooth комуникација, додељују се прекидне рутине за одређене портове и активирају се PID контролери. Искоришћене су две прекидне рутине, које служе за читање позиције мотора. Најмање временско растојање између два узастопна окидања прекидне рутине је 700 микросекунди, да не би долазило до лажног окидања [9]. Да не би долазило до блокирања програма при читању ширине импулса са ултразвучног сензора, постављено је да највећа пауза сме да буде 75 милисекунди. PID контролери избацују вредности од -40 до 40. Време узорковања PID контролера је 50 милисекунди. Алгоритам за управљање роботом се налази на слици 28.



Сл. 28. Управљачки алгоритам робота

Изворни код робота:

```
#include <PID_v1.h>
// #include <NewPing.h>
#include <SoftwareSerial.h>
#define puls A0
#define eho A1
volatile int d = 0;
volatile int l = 0;
float pozl = 0;
float pozd = 0;
float brzd = 0;
float brzl = 0;
float uprl, uprd = 0, upru = 0;
int k = 0, i = 4;
volatile unsigned long mc = 0;
volatile unsigned long mr = 0;
unsigned long vreme = 0;
unsigned long svreme = 0;
float le, de;
double InputL = 0, SetpointL = 30, SetpointD = 30,
OutputL = 0, InputD = 0, OutputD = 0, InputU = 0,
OutputU = 0, SetpointU = -0.0;
double Kp_1 = 0.3, Ki_1 = 0.26, Kd_1 = 0.005;
boolean Zaustavljanje = false;
unsigned int cm = 0;
byte recv;
float ugl, ugd = 0;
PID PIDL(&InputL, &OutputL, &SetpointL, 0.4, .2, 0.1,
DIRECT);
PID PIDD(&InputD, &OutputD, &SetpointD, 0.4, .2, 0.1,
DIRECT);
PID PIDU(&InputU, &OutputU, &SetpointU, 0.6, 0.3, 0.15,
DIRECT);

SoftwareSerial bt(11, 12);

void pcl(){
    if (micros() - mc > 700){
        l++;
        mc = micros();
    }
}

void pcd(){
    if (micros() - mr > 700){
        d++;
        mr = micros();
    }
}
```

```

void setup(){
    Serial.begin(9600);
    bt.begin(9600);
    pinMode(13, OUTPUT);
    pinMode(2, INPUT_PULLUP);
    pinMode(3, INPUT_PULLUP);
    pinMode(5, OUTPUT);
    pinMode(6, OUTPUT);
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(puls, OUTPUT);
    pinMode(eho, INPUT);
    digitalWrite(5, LOW);
    analogWrite(6, 0);
    digitalWrite(9, LOW);
    analogWrite(10,0);
    attachInterrupt(1, pcl, FALLING);
    attachInterrupt(0, pcd, FALLING);
    PIDL.SetMode(AUTOMATIC);
    PIDD.SetMode(AUTOMATIC);
    PIDU.SetMode(AUTOMATIC);
    PIDU.SetOutputLimits(-40, 40);
    PIDL.SetOutputLimits(-40, 40);
    PIDD.SetOutputLimits(-40, 40);
    PIDL.SetSampleTime(50);
    PIDD.SetSampleTime(50);
    PIDU.SetSampleTime(50);
    delay(1000);
}
unsigned int daljina(){
    digitalWrite(puls, LOW);
    delayMicroseconds(2);
    digitalWrite(puls, HIGH);
    delayMicroseconds(10);
    unsigned long vreme = pulseIn(eho, HIGH, 7500);
    return (vreme / 2) / 29.1;
}
void pravo(byte rastojanje){
    digitalWrite (13,LOW);
    InputU = (double) (de - le);
    // InputU = (double) ((brzd - brzl) + (float)(pozd -
    pozl)/(4.0 * ((float)vreme / 1000.0)) * 0.5) ;
    PIDU.Compute();
    SetpointL -= OutputU;
    SetpointD += OutputU;
    SetpointL = SetpointL < 0 ? 0 : SetpointL > 60 ? 60 :
SetpointL;
    SetpointD = SetpointD < 0 ? 0 : SetpointD > 60 ? 60 :
SetpointD;
    InputL = (double) brzl;
    PIDL.Compute();
    InputD = (double) brzd;
    PIDD.Compute();
}

```

```

uprl += OutputL;
uprd += OutputD;
uprl = uprl < 0 ? 0 : uprl > 255 ? 255 : uprl;
uprd = uprd < 0 ? 0 : uprd > 255 ? 255 : uprd;
if (++i >= 5){
    cm = daljina();
    i = 0;
}
if ((cm < (int) растоjanje) && (cm > 0)){
    analogWrite(10, 0);
    analogWrite(6, 0);
    SetpointL = 0;
    SetpointD = 0;
    Zaustavljanje = true;
    i = 4;
} else {
    /* if (Zaustavljanje == true){
        SetpointL = 30;
        SetpointD = 30;
        Zaustavljanje = false;
    }*/
    analogWrite(10,(int) uprl);
    analogWrite(6, (int) uprd);
}
}
void rotacija(byte ugao){
    digitalWrite(13, HIGH);
    switch ((int) ugao){
        case B10000000:
            while (true){
                analogWrite(6, LOW);
                analogWrite(5, ugd >= 130? 0 : 180);
                analogWrite(10, ugl >= 130? 0 : 180);
                analogWrite(9, LOW);
                noInterrupts();
                le = (float)l;
                de = (float)d;
                l = 0;
                d = 0;
                interrupts();
                ugl += le * 0.75;
                ugd += de * 0.75;
                if ((ugd >= 130) && (ugl >= 130)){
                    ugd = ugl = 0;
                    analogWrite(5, LOW);
                    analogWrite(6, LOW);
                    analogWrite(9, LOW);
                    analogWrite(10, LOW);
                    break;
                }
            }
        break;
    }
}

```

```

    case B01000000:
        while (true){
            analogWrite(5, LOW);
            analogWrite(6, ugd >= 130? 0 : 180);
            analogWrite(9, ugl >= 130? 0 : 180);
            analogWrite(10, LOW);
            noInterrupts();
            le = (float)l;
            de = (float)d;
            l = 0;
            d = 0;
            interrupts();
            ugl += le * 0.75;
            ugd += de * 0.75;
            if ((ugd >= 130) && (ugl >= 130)){
                ugd = ugl = 0;
                analogWrite(5, LOW);
                analogWrite(6, LOW);
                analogWrite(9, LOW);
                analogWrite(10, LOW);
                break;
            }
        }
        break;
    }
    recv = 0;
    le = de = l = d = 0;
    Zaustavljanje = false;
    SetpointL = 30;
    SetpointD = 30;
    OutputU = 0;
    OutputL = 0;
    OutputD = 0;
    cm = 0;
    Serial.println(Zaustavljanje);
    delay(150);
}

void loop(){
    if (bt.available()){
        recv = bt.read();
    }
    // vreme = millis() - svreme;
    // svreme = millis();
    noInterrupts();
    le = (float)l;
    de = (float)d;
    d = 0;
    l = 0;
    brzl = (float)le / (4.0 * ((float)vreme / 1000.0)) *
0.5;
    brzd = (float)de / (4.0 * ((float)vreme / 1000.0)) *
0.5;

```



```

    pozl += 360.0 * (float)le / 480.0;
    pozd += 360.0 * (float)de / 480.0;
    interrupts();

    if ((Zaustavljanje == false) && (recv > 0)){
        pravo(recv & B00011111);
    }
    if ((Zaustavljanje) && (recv > 0)){
        rotacija(recv & B11100000);
    }
    Serial.print((int) (recv));
    Serial.print(", ");
    //Serial.println(0.4 * brzl + 0.6 * (upru / k));
    Serial.println(pozd - pozl);
    delay(50);
}

```

3. Закључак

Рад концизно описује пројектовање робота у свим релевантним хардверским и софтверским аспектима. Описани робот је релативно лак за изградњу и програмирање, и има задовољавајуће перформансе. Робот успешно обавља задатак навигације кроз полигон помоћу QR кодова.

Због трења подлоге, при скретању робот за тренутно имплементирани алгоритам управљања не може да тачно направи задат угао. Овај проблем може да се реши додавањем жироскопа или компаса путем којих би вршио корекцију путање. Жироскоп или компас би такође решили проблем праволинијског кретања независно од подлоге, јер би робот имао увид у угао који прави током кретања. PID контролери немају исте перформансе при различитим степенима напуњености батерије. Тај проблем би се могао решити дц-дц конвертером напона после чега би ишао регулатор напона ради стабилизације.

Роботу се може проширити примена на разне начине. Могуће је направити forth интерпретер којим би се робот боље контролисао. Овај језик је изабран због једноставности имплементације интерпретера као и због његове употребе речи, чиме би програми успели да стану у QR код. Једноставно је направити и потпрограма за ротацију робота под произвољним углом, у том случају пакет који се шаље требало би бити величине два бајта.

Даљи развој подразумева развој и имплементацију алгоритама управљања, и евентуално додавање моћније хардверске подршке (бољи сензори и актуатори).

Резултате рада могуће је видети на [10].

Литература

- [1] http://en.wikipedia.org/wiki/Computer_vision#cite_note-Shapiro-Stockman-2001-2, *Компјутерска визија* (приступано 25.08.2014)
- [2] Kurt Demaagd, Anthony Oliver, Nathan Oostendorp, and Katherine Scott, *Practical Computer Vision with SimpleCV*, O'Reilly 2012
- [3] http://library.utm.my/index2.php?option=com_docman&task=doc_view&gid=4532&Itemid=208 (приступано 25.08.2014)
- [4] Gregory Dudek & Michael Jenkin, *Computational Principles of Mobile Robotics second edition*, Cambridge University Press, 2010
- [5] http://sr.wikipedia.org/sr/QR_kod (приступано 25.08.2014)
- [6] <http://moodle.mfkg.rs/login/index.php> *Сензори и актуатори, школска 2013/14* – моодле портал предмета
- [7] Thomas Bräunl, *Embedded Robotics Mobile Robot Design and Applications with Embedded Systems second edition*, Springer, 2006
- [8] J. Warren, J. Addams, H. Molle, *Arduino Robotics*, Springer 2012
- [9] Michael Margolis, *Arduino Cookbook 2nd edition*, O'Reilly 2012
- [10] Никола Јовић, "Никола Јовић – Завршни Рад", <https://www.youtube.com/watch?v=usNO3xgcOIg>, , 2014