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: 56/2017

Милош Д. Јанићијевић

LEGO роботика као илустрациони сценарио теоријских концепата мерења и управљања

завршни рад

Комисија за преглед и одбрану:

1. **Проф. Дипл. инг. Др. Милан Матијевић - ментор**
2. _____ Датум одбране:_____
3. _____ Оцена:_____

У оквиру овог завршног рада кандидат треба да:

Изврши надзор и управљање LEGO DC мотора у склопу са LEGO NXT Mindstorm контролером и додатним LEGO компонената помоћу којих је израђен LEGO робот. У цео поступак је укључен пролаз кроз теоријске основе и принципе мерења и управљања, појам DC мотора као једног од главних компонената склопа LEGO мотора, његов математички модел, састав и начин рада, појам LEGO технологије и могућности у сврху учења и примене у лабораторијским условима. Такође кроз рад укључују се и могућности примене програмског језика Python који се може користити у сврху програмирања LEGO NXT Mindstorm контролера као и поступак припреме радног окружења и начине као и проблематику повезивања контролера са рачунаром. Рад такође укључује и демонстрацију и експерименталну верификацију синтезе позиционог и брзинског сервомеханизма помоћу LEGO DC мотора у улози извршног органа и обраду резултата добијених овим експериментом.

Препоручена литература:

1) Милан Матијевић, Горан Јакуповић, Јелена Цар, „Рачунарски подржано мерење и управљање“, универзитетски уџбеник, Универзитет у Крагујевцу, 2008. година.

2) Ненад Бабајић, Мина Васковић, „Увод у електрична кола, сензоре и актуаторе“, скрипта, Факултет инжењерских наука, 2012. година.

3) М.Матијевић, В.Цвјетковић, В.Филиповић, Н.Јовић, „Основни концепти аутоматике и мехатронике са LEGO програмабилним роботом“, скрипта, Факултет инжењерских наука, 2014. година.

4) Р.Митровић, Пројектовање „LEGO роботске главе – Интеграција функција сензора и актуатора“, završni rad, FIN, Kragujevac, 2013. године.

5) Н.Јовић, В.Цвјетковић, М.Матијевић, „Remote control of LEGO Mindstorms NXT motors programmed in Python“, Универзитет у Крагујевцу, 2015. године.

Крагујевац, 12.03.2020.

Ментор:

Проф. Дипл. инг. Др. Милан Матијевић

Садржај рада

Резиме

У овом раду описана је LEGO Mindstorms технологија као и историјат развијања технологије од настанка па све до сада. Такође поред тога обрађено је и упознавање са хардверским делом саме технологије, компонентама од којих је сачињен један основни LEGO Mindstorms пакет опреме са DC моторима, сензорима и програмабилном коцком NXT као и додатном опремом која уз пакет долази. Описане су софтверске могућности програмирања и програмска окружења за програмирање LEGO NXT коцки као и могућности управљања и контроле истих путем интернета. Такође поред тога наведене су и могућности примене LEGO Mindstorms технологије у едукацији ученика и студената широм света као и примена у медицини. Детаљно је описан процес инсталирања NXT LEGO Mindstorms и Python IDLE програмског окружења као и осталих потребних софтвера у којима је вршен задати експеримент као и процес склапања NXT самобалансирајућег LEGO робота. На крају је извршен експеримент контроле брзине и позиције LEGO DC мотора и обрађени су добијени резултати. Након тога направљен је закључак целог рада и наведена литература коришћена у њему.

Кључне речи: LEGO NXT Mindstorms, Python, DC мотор, Хардвер, Софтвер, Сензор, Програмирање, Контрола, Управљање, Едукација...

Abstract

This paper describes LEGO Mindstorms technology as well as the history of technology development from its inception until now. In addition, we were introduced to the hardware part of the technology itself, the components of which make up a basic LEGO Mindstorms equipment package with DC motors, sensors and programmable cube NXT, as well as accessories that come with the package. The software programming possibilities and programming environments for programming LEGO NXT cubes are described, as well as the possibilities of managing and controlling them via the Internet. In addition, the possibilities of application of LEGO Mindstorms technology in the education of pupils and students around the world as well as its application in medicine are mentioned. The process of installing the NXT LEGO Mindstorms and Python IDLE programming environment as well as other necessary software in which the given experiment was performed as well as the process of assembling the NXT self-balancing LEGO robot are described in detail. Finally, an experiment to control the speed and position of the LEGO DC motor was performed and the obtained results were processed. After that, the conclusion of the whole paper was made and the literature used in it was mentioned.

Keywords: LEGO NXT Mindstorms, Python, DC motor, Hardware, Software, Sensor, Programming, Control, Management, Education…

1. Увод

Да би се на што бољи начин вршила контрола и управљање над једносмерним LEGO мотором пре свега је потребно разумети неке од битних концепата и основа из области мерења и управљања. Такође поред тога важно је познавати појмове статичких и динамичких карактеристика као и шта су то системи отворене и затворене повратне спреге, принцип рада једносмерног мотора као и начин рада PID контролера о чему се може више прочитати у књизи „Рачунарски подржано мерење и управљање“ објављеној на Универзитету у Крагујевцу. [1]

1.1 Увод у LEGO Mindstorms технологију

LEGO роботика представља скуп малих пластичних LEGO коцки од којих се могу саставити различити склопови којима се може помоћу уграђене електронике и софтвера доделити функције. LEGO програмабилни роботи другачије познати као LEGO Mindstorms раде помоћу склопа NXT микрорачунара и већ четрнаест година су доступни на глобалном тржишту као производ фирме LEGO. Између осталих активности проналазе веома битну улогу у наставном програму широм света за увођење ученика и студената у сферу роботике и програмирања као и упознавање са појмовима сензора и актуатора. На територији Сједињених америчких држава малопродајна цена се креће већ од $290 за неки основни комплет док је код нас цена мало већа. Без обзира на релативно ниску цену LEGO NXT није стандардно наставно средство на нижим нивоима образовања код нас. Са друге стране на вишим нивоима образовања се захваљујући разним страним донацијама све више уводи у програм образовања. У овом раду је описан један део LEGO Mindstorms NXT технологије као и неке од њених могућности. На Универзитету у Крагујевцу развијена је база типичних Python програма који омогућавају студентима реалтивно брз приступ експериментисању на LEGO платформама и без подршке комерцијалних софтверских алата. Такође извршени су и први експерименти контроле LEGO NXT микрорачунара путем Интернета што може знатно унапредити примену ове технологије у образовању поготово у условима где има великог броја полазника курса. Поред програмског језика Python такође постоје додатни начини за контролу LEGO мотора помоћу програмских пакета LabVIEW и MATLAB и примену NI USB 6008 AD/DA kartica и NI ELVIS лабораторијског тренажера као и Arduino микрорачунара. [2] На слици 1. може се видети изглед LEGO Mindstorms NXT робота који је један од многобројних комплета. [3]

Слика 1. LEGO Mindstorms NXT робот. [3]

1.2 Историјат развијања технологије

Када је 1930. године основана компанија LEGO у Данској, ретко ко је могао помислити да ће достићи успех који има на данашњем нивоу. У почетку су били базирани искључиво на изради дечијих играчака за децу широм света као и развијање њихових могућности и маште. Главни мото компаније како је сам оснивач Ole Kirk говорио гласи „Само најбоље је довољно добро“. [4]

LEGO компанија је током година имала неколико временских прекретница у свом пословању:

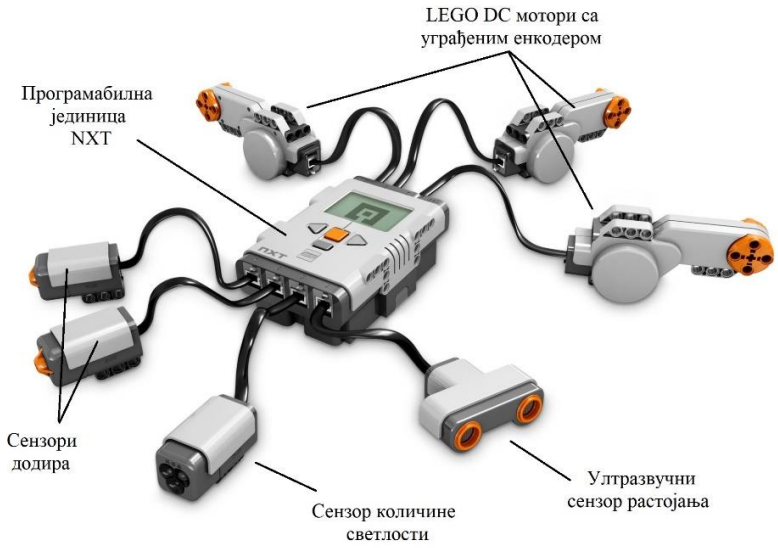
- 1980. године LEGO компанија основала је област производње производа намењених образовању.
- 1986. године се у продајној понуди појављује први производ који је могуће контролисати компијутером.
- 1988. године MIT и LEGO у сарадњи заједно раде на развоју прве паметне коцке коју је могуће програмирати.
- 1998. године LEGO покреће линију Mindstorms која се састоји из RCX (Robotic Command eXplorers) коцке програмиране помоћу језика NQC.
- 2006. године настаје нова линија паметних коцки под називом LEGO Mindstorms NXT.
- 2008. године је обележена десетогодишњица лансирања првог комплета из серије Mindstorms у продају поводом чега је LEGO објавио нову верзију паметне коцке под називом NXT 2.0 уз коју иду нови унапређени сензори са новим графичким програмским језиком као и бржом Bluetooth технологијом за комуникацију са рачунаром и уграђеним mp3 player-ом. [4]
- 2013. године излази трећа генерација LEGO Mindstorms паметне коцке под називом EV3 уз коју је додатно унапређен и олакшан софтвер за програмирање команди робота. [5]

На слици 2. може се видети прва паметна коцка развијена у сарадњи MIT и LEGO компаније. [4]

Слика 2. Прва паметна коцка развијена у сарадњи MIT и LEGO компаније. [4]

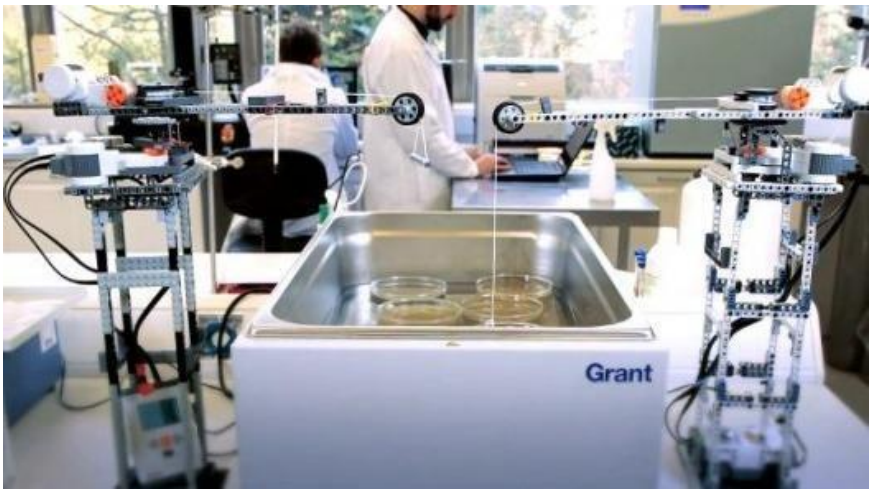
2. LEGO Mindstorms технологија

LEGO Mindstorms комплет се стандардно састоје од програмабилне јединице која садржи процесорске компоненте и контролер робота који служи за задавање функција роботу, два сензора додира, сензора количине светлости, ултразвучног сензора растојања, три LEGO DC мотора са интегрисаним енкодером. На слици 3. може се видети један LEGO Mindstorms NXT комплет. [6]



Слика 3. LEGO Mindstorms NXT комплет. [6]

Имају широке могућности у образовању и едукацији о аутоматизованим процесима који ће представљати будућност у роботици. Програмирање LEGO робота се најчешће може видети код ђака и студената у основним и средњим школама као и на факултетима. Кроз интерактивне пројеке могу се врло лако научити основе роботике. Једна од већих предности је та што се избагавају примене традиционалног учења. Поред тога LEGO роботи могу имати примену и у медицини где се могу користити за различита медицинска испитивања. Такође поред тога могу се пројектовати роботи који ће различитим звуковима опомињати људе да је време за дневну дозу лекова чиме робот може по боји гласа да препозна више корисника и тиме распореди дозу лекова за више особа истовремено. Поред тога могу наћи примену и у индустрији за разврставање предмета и производа по боји помоћу сензора количине светлости који има могућност распознавања нијансе боја као и многобројних других индустријских захтева. На слици 4. може се видети примена LEGO роботике у медицини. [6] У наставку се може прочитати нешто више о хардверском саставу LEGO DC мотора, програмабилних паметних коцки, LEGO сензора као и додатне опреме као и софтверској подршци LEGO Mindstorms технологије.

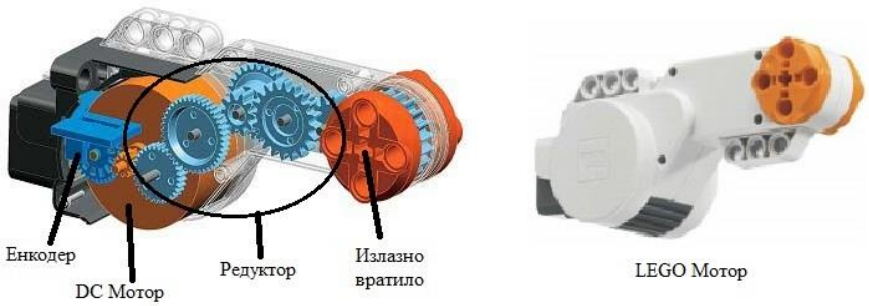


Слика 4. Примена LEGO роботике у медицини. [6]

2.1 Хардверске компоненте

2.1.1 LEGO DC мотор

LEGO DC мотор се састоји из енкодера који служи за детектовање окретања једносмерног мотора, DC мотор који је основни део и покреће цео склоп, система редуктора који прилагођавају снагу и обртни момент мотора и излазно вратило на које се повезују LEGO компоненте. Поред тога он поседује и RJ-12 прикључак за повезивање са програмабилним телом. На слици 5. може се видети изглед NXT LEGO DC мотора. [2]

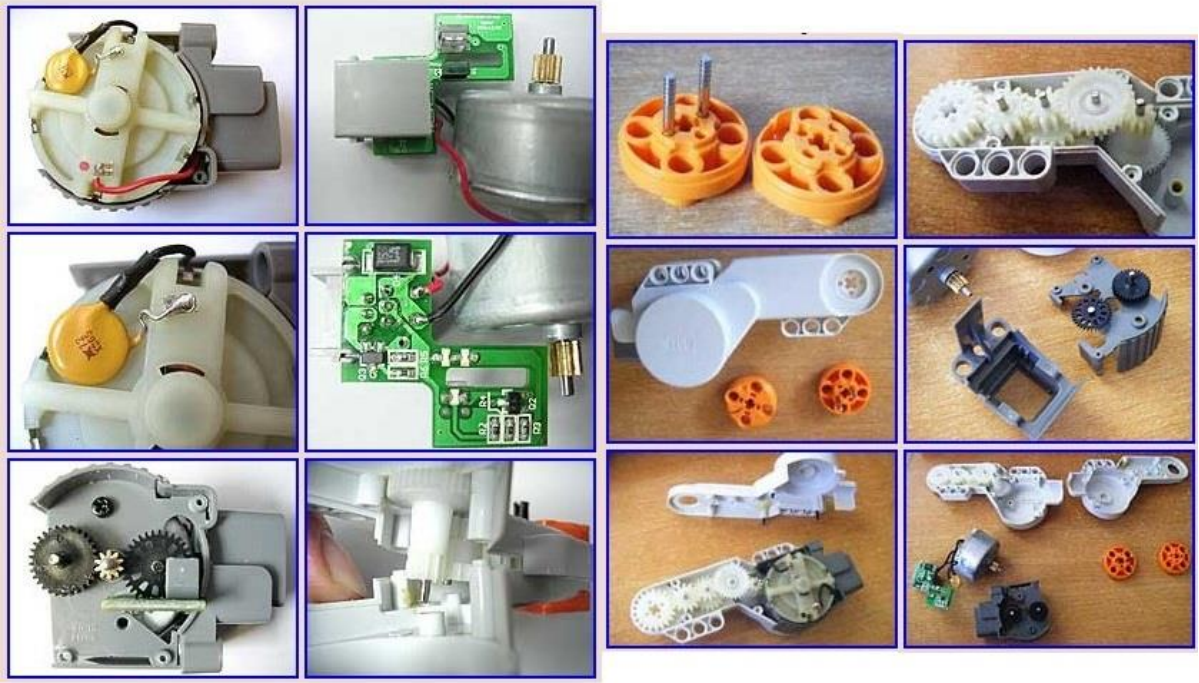


Слика 5. LEGO NXT DC мотор. [2]

Технички параметри LEGO једносмерног мотора су:

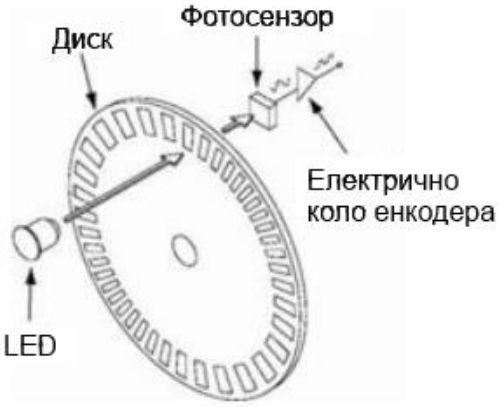
- Напајање 9V
- Угаона брзина 177 min^{-1}
- Обртни момент 16.7 Ncm
- Снага 2.03 W
- Ефикасност 41% [6]

DC мотор који се налази у склопу LEGO мотора напаја се помоћу батерије од 9V која се налази у NXT програмабилном телу. Служи за претварање електричне енергије из батерије у механичку ротацију на вратилу. Поседује сасвим довољну снагу за обављање функција кретања робота као и потребе подизања терета и померања сензора по захтеву. [6] Још један јако битан параметар код кретања LEGO DC мотора јесте такозвани импулс модулације ширине (PWM – Pulse Width Modulation) који се користи за контролу брзине сервомотора. [11] На слици 6. може се видети унутрашњи склоп NXT мотора. [7]



Слика 6. Унутрашњи склоп NXT мотора. [7]

Једна од најбитнијих карактеристика NXT LEGO DC мотора је та што поседује интегрисан инкрементални енкодер. То је врло важно јер омогућава прецизно управљање угаоним положајем и брзином. Енкодер представља врсту сензора који даје повратну информацију о положају мотора што га чини серво мотором. Код NXT мотора овај сензор је оптички енкодер који ради помоћу инфрацрвене LED диоде, фотосензора и диска са 12 отвора који се налазе између њих. LED диода емитује инфрацрвене зраке светлости који падају на диск и у зависности од тога да ли је између диоде и фотосензора отвор или не, фотосензор ће пропуштати струју уколико прима светлост кроз отвор од стране LED диоде. На слици 7. може се видети скица оптичког енкодера NXT мотора. [8]



Слика 7. Скица оптичког енкодера NXT мотора. [8]

За један окрет излазног вратила, електромотор је потребно да се заокрене 48 пута. Пошто је претходно познато да се за 10 окрета диска енкодер електромотора окрене 32 пута можемо одредити однос између излазног вратила и енкодера. Добијамо да ће се за један окрет излазног вратила диск енкодера окренути 15 пута и фотосензор ће регистровати 180 отвора. Фотосензор мења своје стање у случају да је испред њега отвор или заклон па имамо 360 промена по једном окрету излазног вратила. На основу тога закључујемо да је осетљивост оптичког енкодера 1 степен. [8]

2.1.2 Програмабилне паметне коцке

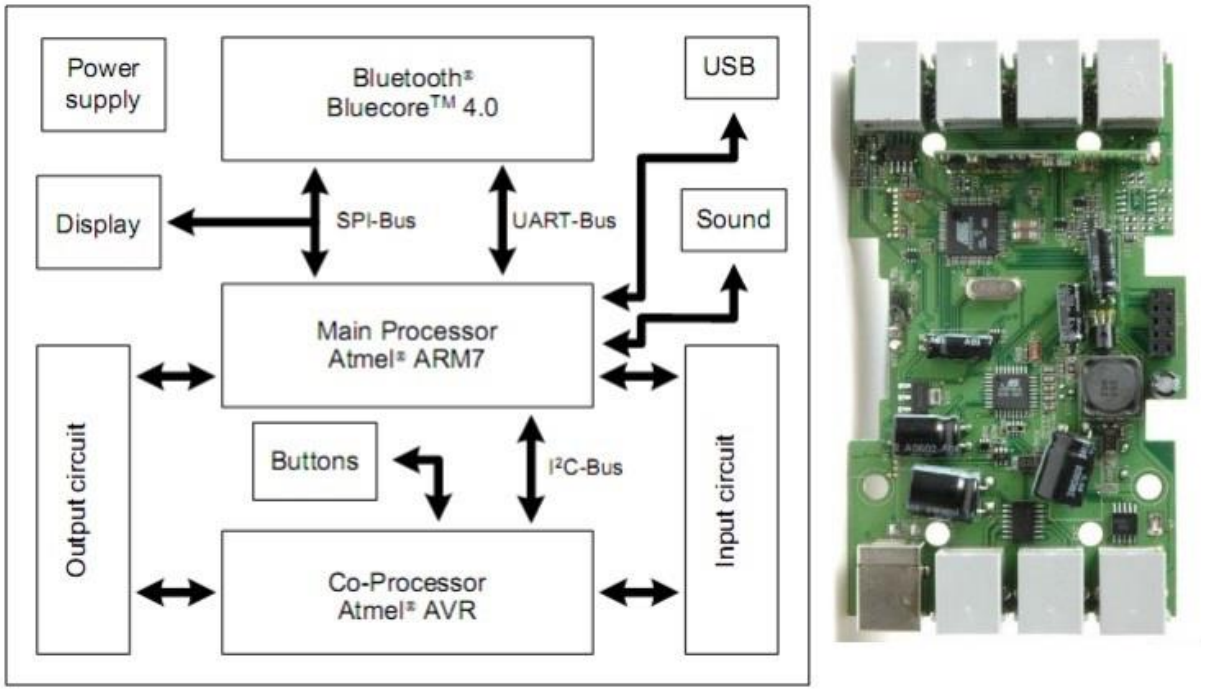
LEGO NXT програмабилна паметна коцка је микрорачунар преко ког се програмирају функције робота. Представља гравну јединицу у склопу робота преко које се повезују све потребне компоненте и врши њихово напајање. Поред LEGO NXT програмабилне коцке постоје још две верзије из породице LEGO Mindstorms од којих је један претходник а други наследник LEGO NXT-a. Један од њих се назива RCX а други EV3. На слици 8. може се видети изглед све три програмабилне јединице једна поред друге при чему се може приметити напредак у самој технологији израде. [9]

Слика 8. Породица LEGO Mindstorms програмабилних паметних коцки. [9]

LEGO NXT програмабилна паметна коцка се састоји из следећих компонената:

- Главни процесор радне фреквенције 48MHz који се састоји од 32bit-ног ARM микроконтролера Atmel AT91SAM - 7s256, Флеш меморије 256KB, RAM меморије 64KB и USB интерфејса. Служи за извршавање задатих програма.
- Помоћни процесор радне фреквенције 8MHz који се састоји од 8bit-ног Atmel AVR просесора Atmega48, Флеш меморије 4KB и 512B RAM меморије. Има улогу у контроли DC мотора.
- 4 RJ-12 аналогна улаза за сензоре.
- 3 RJ-12 PWM излаза за побуду LEGO DC мотора са каналом за читање енкодера интегрисаних у кућиште LEGO DC мотора.
- Бежични Bluetooth CSR BlueCoreTM 4 v2.0 + EDR систем који подржава серијски улаз, 47KB RAM меморије, 8MB флеш меморије, радни такт 26MHz.
- USB 2.0 за размену података максималном брзином од 12 Mbit/s.
- LCD монохромски дисплеј резолуције 100x64 pixel-a.
- Звучник 2-16KHz sa 8bit-ном резолуцијом.
- Напајање, које чине 6AA батерија од по 1.5V (укупно 9V). [6]

На слици 9. може се видети шематски приказ компонената унутар NXT програмабилне коцке. [10]



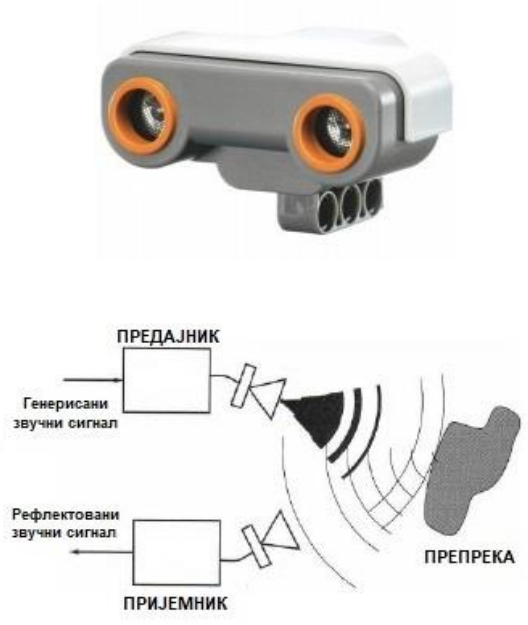
Слика 9. Шематски приказ компонената унутар NXT програмабилне коцке. [10]

2.1.3 LEGO сензори

Као што је горе наведено, NXT LEGO комплет поседује одређени број сензора који знатно подижу могућности које роботи могу да обављају. Помоћу њих робот добија информације из спољашње средине на основу којих може прецизније обављати функцију коју му корисник задаје. NXT LEGO комплет поседује ултразвучни сензор растојања који се другачије назива сонар, сензор који распознаје боју и количину светлости и два сензора додира. Поред ових сензора постоје и додатни сензори који се могу посебно купити а то су сензор звука и жироскоп сензор. [6]

- Ултразвучни сензор растојања (сонар)

Ултразвучни сензор је један од најкоришћенијих сензора који се налазе у комплету. Омогућава мерење удаљености до објекта на који је усмерен. Принцип рада ултразвучног сензора је следећи. Сензор шаље ултразвучни сигнал на фреквенцији од 40KHz који људско ухо не детектује. Сигнал путује од сензора у правцу у ком је усмерен и у тренутку када наиђе на препреку одбија се од препреке натран према сензору који прима сигнал назад и на основу његовог времена ширења израчунава удаљеност до објекта. Сензор је у стању да мери удаљеност од 0cm до 250cm са тачношћу +/- 3cm. Највећу примену има у ситуацијама када се захтева да робот избегава препреке. [12] Битно је напоменути да је тачност сензора већа уколико се ултразвучни сигнал одбија од већих објеката а мања уколико се одбија од мањих објеката и у компликованијој средини. На слици 10. може се видети изглед ултразвучног LEGO сензора и скициран принцип рада. [6]



Слика 10. Изглед ултразвучног LEGO сензора и скициран принцип рада. [6]

- Сензор боје и количине светлости

Сензор за боју може да обавља три различите функције. Може да ради као сензор за боју при чему разликује шест боја (црну, плаву, зелену, жуту, црвену, белу) а може радити и као сензор количине светлости (рефлектованог и амбијенталног). Додатно може радити и као лампа при чему може емитовати црвено, зелено или плаво светло. Плаво индикационо светло означава да робот мирује, да је комуникација са рачунаром успостављена и да је робот спреман да прихвати и изврши наредбу. Зелено светло озлачава да је робот у покрету а црвено да је робот ударио у препреку. [6] Овај сензор не препознаје боје директно као људско око већ препознаје количину светлости које поседује свака боја те на основу тога одређује боје. Још једна занимљивост је да је фототранзистор смештен у сензор много осетљивији на инфрацрвено зрачење него људско око. Његов спектар видљивости је од 400nm до 1150nm. Највећу примену има у задатцима у којима се од робота захтева праћење одређене линије. На слици 11. може се видети изглед два типа сензора боје и количине светлости. [12]

Слика 11. Два типа сензора боје и количине светлости. [12]

- Сензор додира

Сензор додира је најједноставнији сензор у NXT Mindstorms комплету. Излаз има два стања. Када је наранџасти покретни део сензора притиснут тада сензор реагује а када није тада не реагује. Овај сензор је најпогоднији за употребу као стоп тастер чиме ће наиласком на препреку заустављати робота. [12] Добра карактеристика је та што има крстасти отвор на средини што омогућава да се сензор повеже директно са осталим LEGO деловима. [6] На слици 12. може се видети изглед сензора додира. [12]

Слика 12. Сензор додира. [12]

- Сензор звука

Сензор звука NXT дизајниран је на сличан начин као и сензор светлости. Састоји се од микрофона уграђеног у пену. Јачина звука или ниво звука мере се у јединицама који се називају децибели [dB]. Децибели одређују колико је гласнији или тиши звук него било који други звук. Звук од 0dB је најнижи звук који просечна особа може да чује. Овај сензор има могућност да чује звук до јачине од 90dB. Сензор такође има могућност да се пребаци у dBA мод у ком сензор има могућност да чује звукове у спектру људског чула у опсегу од 20Hz до 20KHz. [11] Као неку референцу можемо рећи да звук јачине од 10-30% је нормални ниво разговора у близини сензора док је звук у опсегу од 30-100% представља опсег од гласаног разговора до врло гласне музике. Ове референце се могу применити на удаљености од робота на растојању од 1m или мање. [4] На слици 13. може се видети изглед сензора звука. [11]

Слика 13. Сензор звука. [11]

- Жироскоп сензор

Овај сензор омогућава читање брзине и смера ротације робота у једној оси. Вредности су дате у степенима у секунди. Сензор је у стању да мери до 360 степени у секунди а може се користити за изградњу робота који је у стању да балансира сам себе или за друге задатке где је потребно мерење брзине и ротације. Сензор користи аналогни улаз а фреквенција скенирања је приближно 300Hz. Такође овај сензор се може користити и као компас тако што мери земљино магнетно поље и израчунава одступање од северног пола у степенима. Ово омогућава прецизну навигацију робота у простору. Сензор има могућности да мери у опсегу од 0-359 степени и до 100 пута у секунди. На слици 14. може се видети жироскоп сензор. [10]

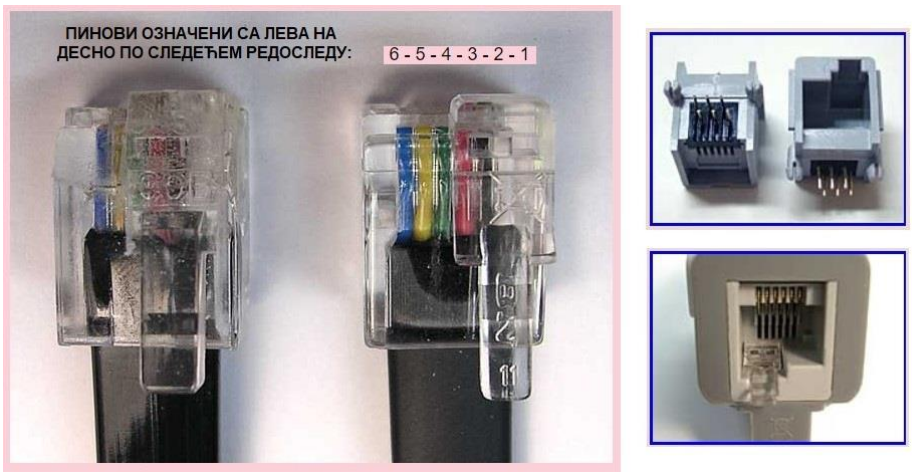
Слика 14. Жироскоп сензор. [10]

2.1.4 Додатна хардверска опрема

Приликом израде робота LEGO Mindstorms поред претходно напоменутих сензора, паметних коцки и мотора, потребно је користити и класичне LEGO коцке које пружају широк опсег могућности и креативности за израду скоро сваког замишљеног пројекта. Коцке имају главку улогу у изради конструкције робота као и физичког повезивања конструкције са сензорима, паметном коцком и моторима. Постоје у разним облицима и величинама за различите намене. На слици 15. може се видети изглед LEGO коцки од којих се састављају роботи из сета NXT. [13]

Слика 15. LEGO коцке од којих се састављају роботи из сета NXT. [13]

Такође поред физичког повезивања компоненти, битно је обратити пажњу и на комуникацију компоненти између себе. Овај процес се обавља путем LEGO NXT флет каблова са RJ-12 конекторима. Ови каблови се састоје из шест проводника различитих боја. На слици 16. може се видети изглед RJ-12 конектора на каблу и NXT паметној коцки. [14]



Слика 16. Изглед RJ-12 конектор на каблу и NXT паметној коцки. [14]

На слици 17. може се видети табела која приказује намену проводника који се налазе у флет каблу RJ-12. [14]

Редни број пина NXT кабла	Боја проводника	Функција	Сигнал / Поларитет	Облик сигнала
1	Бела	Напајање мотора	DC 9V + или -	DC
2	Црна	Напајање мотора	DC 9V - или +	DC
3	Црвена	Напајање енкодера	Уземљење (0V)	DC
4	Зелена	Напајање енкодера	+ 4.3 (до 5.0 V)	DC
5	Жута	Излаз детекције ротације 1	0 / 4.3 V супротно 3, -4.3V / 0V супротно 4	Правоугаони
6	Плава	Излаз детекције ротације 2	0 / 4.3 V супротно 3, -4.3V / 0V супротно 4	Правоугаони

Слика 17. Табела која приказује намену проводника који се налазе у флет каблу RJ-12. [14]

Као што је већ напоменуто напајање робота се врши преко шест AA батерија укупног напона од 9V. Мана NXT паметне коцке је та што нема могућност аутоматског пуњења батерија те је због тога најбоље користити пуњиве mAh батерије уз екстерни пуњач. Додатну проблематику ствара то јел се приликом сваког пуњења батерија робот мора расклапати због њиховог вађења и поново склапати након што се напуне. Поред тога уколико се NXT коцка повезује жичано на рачунар потребно је користити USB-A на USB-B кабл ради транспорта података са рачунара на коцку. Друга опција је повезивање бежично путем Bluetooth конекције. Један од многобројних пуњача са пуњивим батеријама који је доступан на тржишту и који је јако поуздан са дугим веком трајања је Sanyo Ni-MH. Има могућност пуњења до четири AA или AAA батерије истовремено и поседује индикатор пуњења у виду LED диоде. Битно је напоменути да се истовремено може пунити само паран број батерија. На слици 18. може се видети изглед Sanyo Ni-MH комплета батерија са пуњачем. [15]

Слика 18. Sanyo Ni-MH комплет батерија са пуњачем. [15]

2.2 Софтверске могућности програмирања

Са софтверског аспекта могућности програмирања NXT LEGO паметних коцки као и других микрорачунара и контролера су велике. Контролу LEGO DC мотора који се налази у склопу LEGO робота могуће је извести помоћу Arduino микрорачунара коришћењем подржаних програмских језика C и C++ а поред тога контрола и обављање програмирања могуће је и путем NI USB 6008 AD/DA картица које се користе као јефтинија замена за NI ELVIS лабораторијски тренажер. Arduino има могућност чувања кода на сопственој меморији тако да није потребно да константно буде повезан са рачунаром као код NI картица. При коришћењу напоменутих картица за програмирање користе се програмска окружења LabVIEW као и програмско окружење MATLAB/Simulink. Битно је напоменути да је у случају примене Arduino микрорачунара или NI USB 6008 AD/DA картица за контролу LEGO DC мотора потребно израдити мање електрично коло које ће прилагодити сигнале који долазе са контролера по онима који одговарају мотору. У овом раду је детаљније обрађена пажња на друге начине контроле LEGO DC мотора применом програмирања. Први начин је класичан LEGO NXT софтвер помоћу ког се NXT коцка програмира помоћу рачунара. Он је веома једноставан за употребу и интерактиван а у раду је укратко описан начин употребе и процес инсталирања радног окружења. Други начин је програмирање NXT коцке помоћу програмског језика Python у IDLE програмском окружењу са малим прилагођавањима за потребе NXT коцке који је нешто детаљније описан. Поред овога програмски језик Python је јако користан и поседује веома велику библиотеку програма који се могу користити у сврху програмирања NXT коцки. Такође NXT Python је окружење које је успешно тестирано на различитим Microsoft Windows верзијама оперативних система од XP, Vista, 7, 8, 8.1 до 10 а поред тога успешно ради и на Linux оперативним системима. Веома интересантна ствар је могућност рада на све популарнијем микрорачунару Raspberry Pi који је веома јефтин и за своју величину која је једнака величини кредитне картице, веома је јак и поседује веома корисне могућности. Он ради на прилагођеним верзијама Debian и Fedora Linux оперативних система његовом хардверу који се називају Raspian и Pidora. Raspberry Pi је пре свега развијен за потребе едукације и образовања а поред тога такође поседује и веома добру подршку за различите софтвере као и уређаје и сензоре. Ради бољег искоришћења постојећих лабораторијских LEGO NXT ресурса, на Факултету инжењерских наука покренут је развој за даљински приступ LEGO опреми која треба да омогући да студенти преко Web-а контролишу роботе као и да тестирају своја програмска решења постављених експерименталних задатака. Поред тога овакав приступ лабораторијским ресурсима помоћу софтвера на даљину погодан је из многих других разлога као на пример у случајевима избијања ванредних стања и пандемија вируса у којима је ризично окупљање великог броја студената и ученика у лабораторијским просторијама. Програмски језик Python се на различите начине може користити преко Web-а, из интернет претраживача корисника. На слици 19. може се видети основни концепт LEGO Web лабораторије која има сврху едукације студената о основним концептима роботике, сензора и актуатора, мерења и управљања као и основа програмирања и контроле на даљину. [2]

Слика 19. Концепт рада LEGO Web лабораторије на Факултету инжењерских наука. [2]

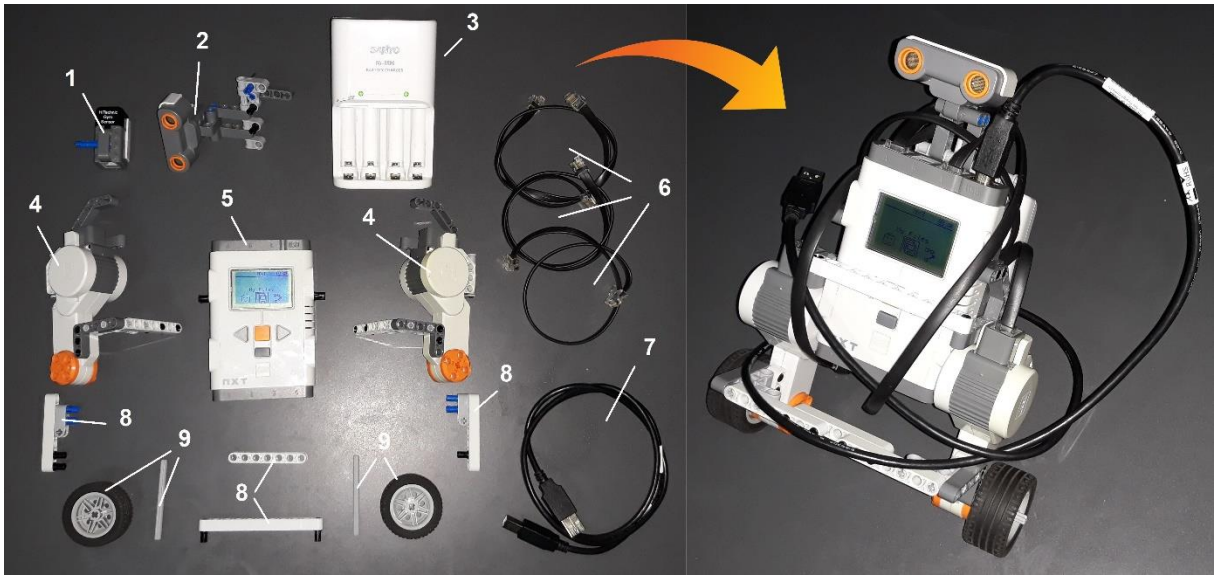
Овако подешена Web лабораторија функционише на следећи начин:

Удаљени корисник – студент преко мреже користећи било који савремени Web претраживач пријављује се администраторском систему Web сервера где су дефинисана његова права приступа и коришћења LEGO Web робота. Python програми који контролишу роботе се извршавају на Web серверу који се налази у лабораторији и на који су прикључени LEGO роботи преко USB кабла или Bluetooth конекције. Пријављени студент може да Upload-ује свој фајл са Python програмом који контролише LEGO робота и да на тај начин тестира свој програм. Сесија удаљеног корисника је имплементирана преко AJAX-а који је облик Web сервера за складиштење података. Помоћу њега се омогућава слање података од корисника до лабораторијског сервера. Резултат рада програма се може пратити преко података које Python програм генерише и враћа преко Web сервера. У систем је такође имплементована Web камера која омогућава визуелно праћење померања робота у колико је то потребно. Осим слања резултата рада у виду нумеричких вредности и посматрања кретања преко Web камере, на Web страници се може адекватном анимацијом коришћењем програмских језика HTML, CSS и JavaScript представити ефекат управљања роботом или само његовим моторима без потребе за инсталацијом додатних Plug-in-ова за приказ у коришћеном Web претраживачу. Претходно приказан концепт рада на слици 19. је независан од оперативног система као и хардверске платформе. Може се користити подједнако како на калсичном рачунару тако и на знатно скромнијој хардверској платформи Raspberry Pi уз примену различитих Web сервера. [2]

3. Припрема опреме и радног окружења

3.1 Склапање робота

Склапање LEGO робота је веома лак и једноставан процес. Конкретно за извођење експеримента за тестирање брзине и позиције LEGO мотора као и добијање карактеристика није толико важна намена и изглед самог робота тако да се може изабрати модел по жељи. У овом случају изабран је самобалансирајући модел LEGO робота који на себи поседује жироскоп и ултразвучни сензор. Такође на његову конструкцију се могу имплементовати и други сензори по жељи корисника и по потреби за извршавање датог задатка. На слици 20. може се видети изглед самобалансирајућег LEGO робота и делови од којих је састављен.



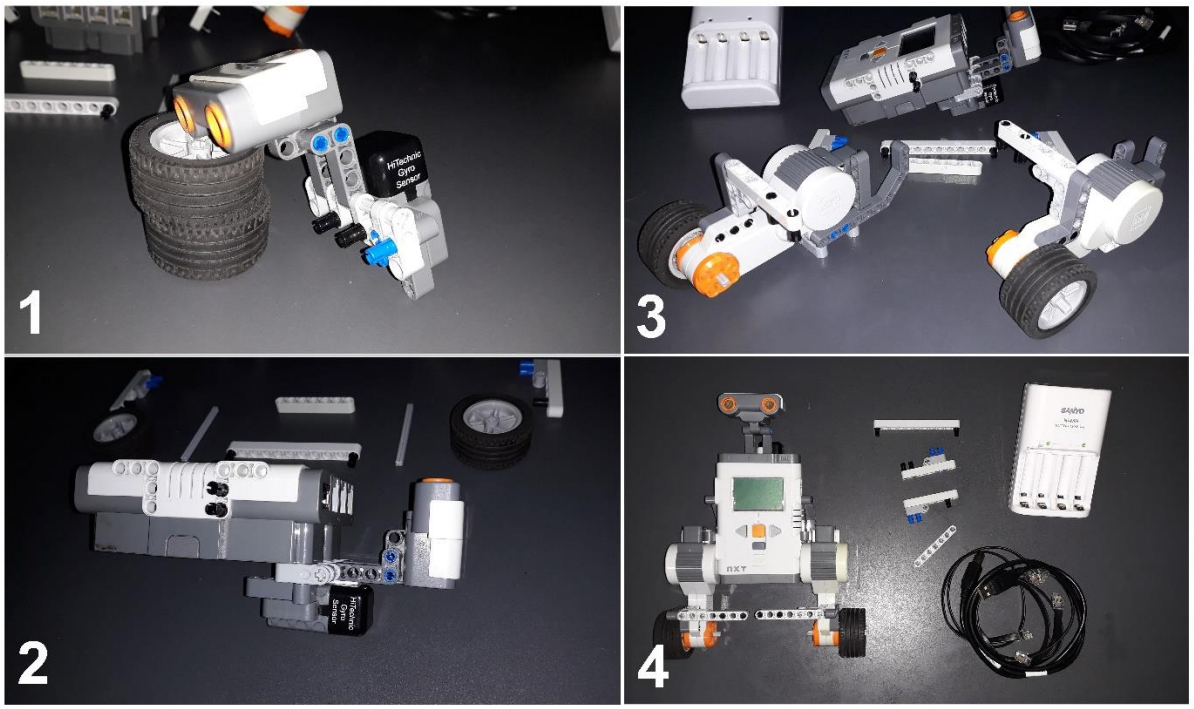
Слика 20. Изглед самобалансирајућег LEGO робота и делова од којих је састављен.

Делови са слике 20. су означени са бројним ознакама од 1 до 9 који представљају:

1 – Жироскоп сензор,
2 – Ултразвучни сензор (сонар),
3 – Sanyo Ni-MH пуњач батерија,
4 – LEGO DC мотори са конструкцијом за спајање са NXT коцком,
5 – NXT паметна програмабилна коцка,
6 – RJ-12 каблови за пренос сигнала од мотора и сензора до NXT коцке,
7 – USB кабл за комуникацију са рачунаром,
8 – Делови за појачање чврстоће конструкције,
9 – Точкови са осовинама преко којих се спајају са мотором.

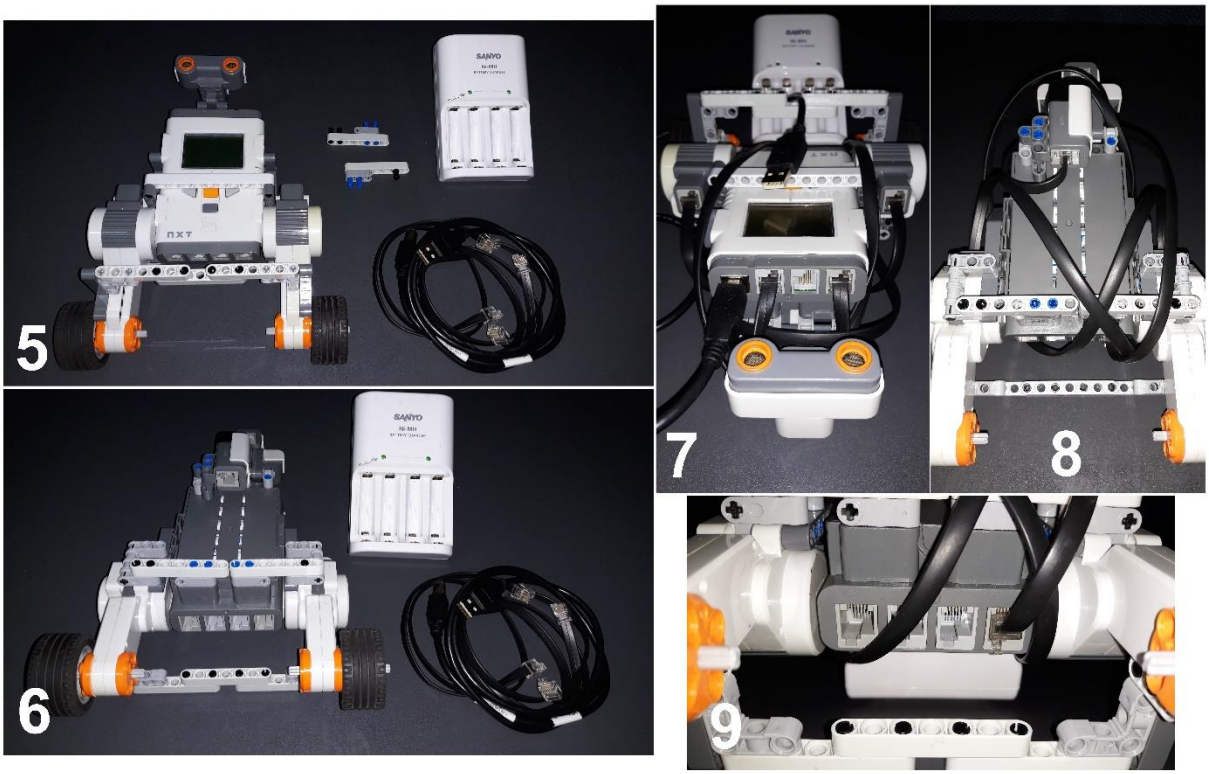
Процес склапања робота је следећи:

- Први корак – спајање жироскоп сензора и ултразвучног сензора заједно. Они се помоћу конструкције израђене помоћу LEGO коцки спајају заједно при чему та конструкција има могућност конектовања на задњу страну NXT коцке тако да је ултразвучни сензор окренут у смеру кретања робота а жироскоп сензор одмах испод њега. На слици 21. може се видети склоп жироскоп и ултразвучног сензора означен са бројем 1 као и њихово спајање са NXT коцком означен са бројем 2.



Слика 21. Процес склапања робота (први део).

- Други корак – на слици 21. се такође може видети и склоп мотора који су повезани са точковима преко осовине као и конструкција која спаја моторе са NXT коцком означена са бројем 3. Са бројем 4 је означен изглед робота након спајања склопа мотора и точкова са NXT коцком на коју је претходно спојен склоп сензора. Овим се практично добија приближно финалан изглед робота који је замишљен за израду али је битно напоменути пар ствари. Робот који је овако конструисан нема баш толико чврсту конструкцију тако да не може најефикасније да објављa задатке. Да би се отклонио овај проблем користе се додатне LEGO коцке које спајају конструкције оба мотора заједно са предње и задње стране LEGO робота чиме се добија веома робот са веома чврстом конструкцијом. На слици 22. може се видети робот са учвршћеном конструкцијом са предње и задње стране робота означено са бројевима 5 и 6.



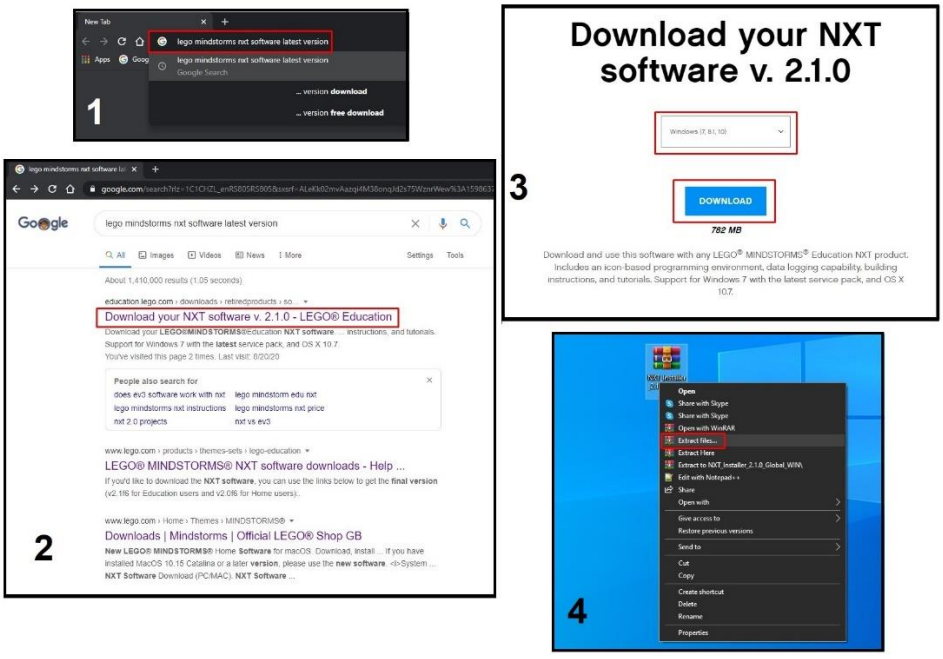
Слика 22. Процес склапања робота (други део).

- Трећи корак – на слици 22. се још може видети и процес повезивања сензора и мотора са NXT коцком помоћу RJ-12 каблова као и повезан USB кабл са NXT коцком означени са бројевима 7, 8 и 9. У овом случају жироскоп сензор је повезан на порт 1 на NXT коцки, портови 2 и 3 су празни док се на порт 4 повезује ултразвучни сензор. Гледано са предње стране LEGO робота, леви мотор је повезан на порт A на NXT коцки док је десни мотор повезан на порт C. Порт B се не користи у овом случају мада битно је напоменути да уколико се врши измена портова програм се мора изменити по направљеним изменама на роботу. Овим се добија финални изглед робота који се може користити у сврху обављања жељеног експеримента. На слици 23. може се видети финални изглед LEGO NXT самобалансирајућег робота.

Слика 23. LEGO NXT самобалансирајући робот.

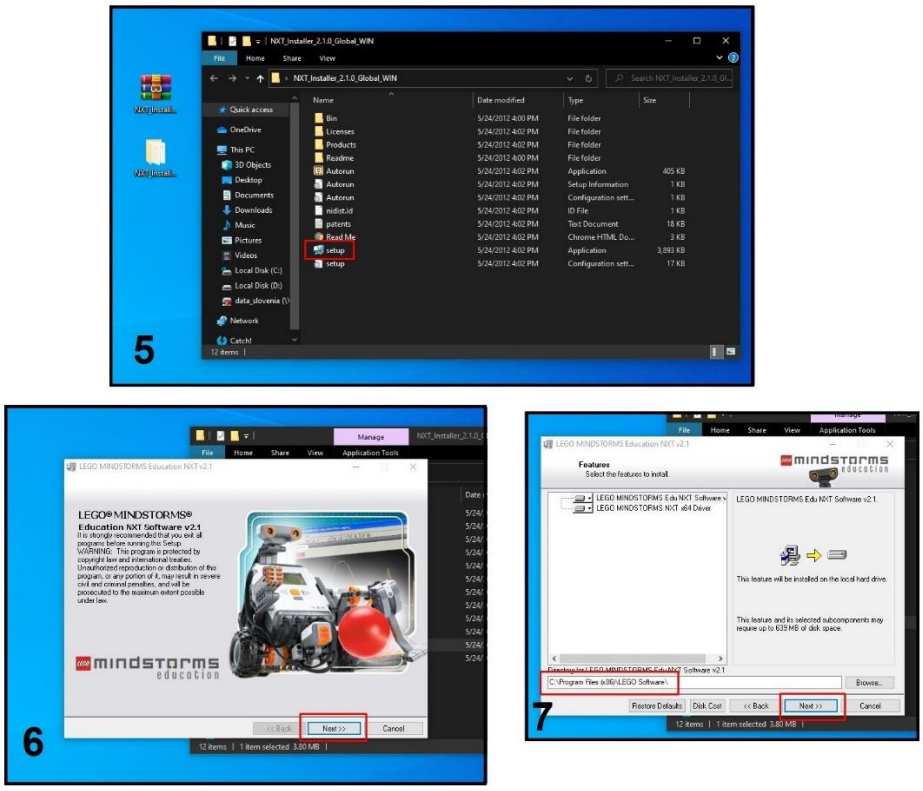
3.2 Процес инсталирања LEGO NXT програмског окружења

За почетак да би се инсталирало LEGO Mindstorms NXT програмско окружење потребно је отворити Web претраживач и на сајту Google укуцати „lego mindstorms nxt software latest version“ а затим притиснути тастер „Enter“ као што је то приказано на позицији 1 након чега ће Google пронаћи следеће опције са позиције 2. Потребно је изабрати опцију селектовану црвеним правоугаоником која је официјални сајт са ког се може преузети легална верзија LEGO Mindstorms софтвера. Након тога треба одабрати оперативни систем који се користи на рачунару са ког се преузима програм и кликнути на опцију „Download“ као на позицији број 3. Када се преузимање заврши добија се документ у .rar архиви коју треба распаковати десним кликом и бирањем опције „Extract files…“ као што је то приказано на позицији 4. Када се распаковање заврши добија се фолдер са истим називом као .rar архива који треба отворити и притиском на фајл „Setup“ покренути инсталирање софтвера као на позицији 5. Када се покрене инсталација у дијалогу на позицији 6 је потребно кликнути на „Next“ а затим на следећем изабрати локацију инсталирања софтвера и кликнути на „Next“ као на позицији 7. Након овога на позицији 8 имамо опцију да прочитамо сагласност са лиценцом софтвера коју морамо прихватити уколико желимо да инсталирамо софтвер што се може учитини чекирањем кружића „I accept the above 2 License Agreement“ а затим наставити притиском на опцију „Next“. Након тога на позицији 9 је финална потврда пред инсталацију на коју је потребно кликнути „Next“ након чега се започиње инсталација софтвера на позицији 10. Након завршетка инсталације потребно је кликнути на опцију „Finish“ као на позицији 11. Програм се стартује кликом на иконицу „NXT 2.1 Programming“ која се појавила на Desktop-у након инсталације као на позицији 12 након чега ће програм учитавати пар секунди (позиција 13). На позицији 14 се може видети стартован програм са почетним менијем. На слици 24. може се видети први део процеса инсталације NXT програмског окружења. [19]



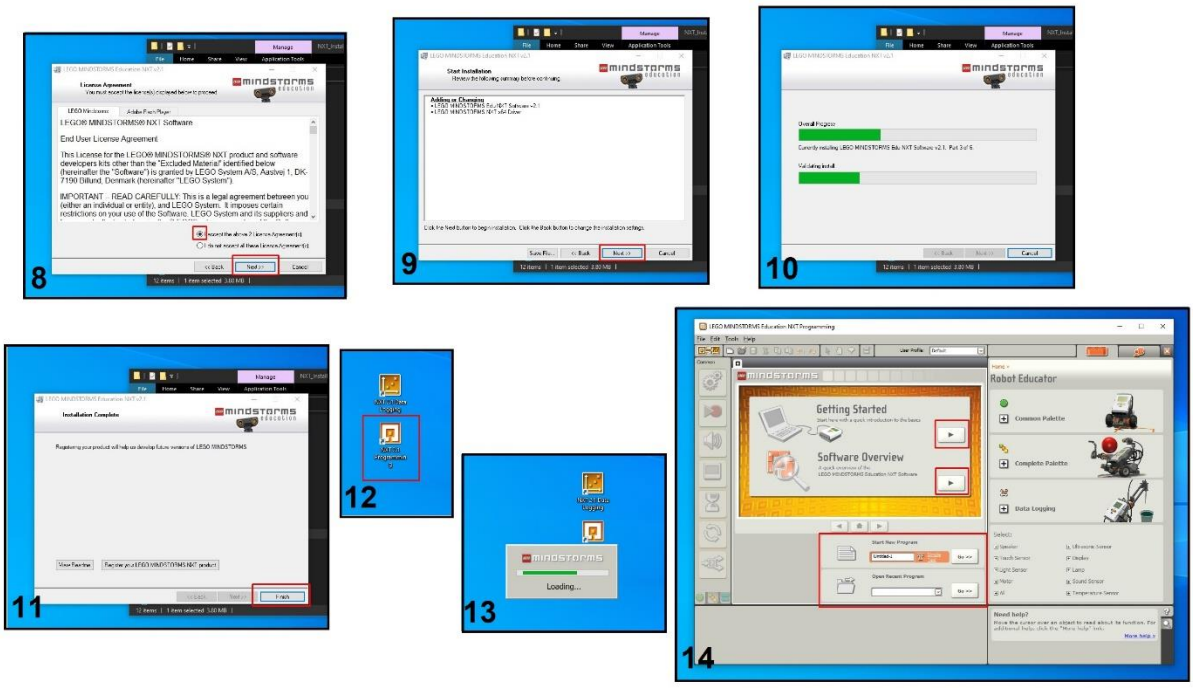
Слика 24. Процес инсталације NXT програмског окружења (први део). [19]

На слици 25. може се видети други део процеса инсталације NXT програмског окружења.



Слика 25. Процес инсталације NXT програмског окружења (други део).

На слици 26. може се видети трећи део процеса инсталације NXT програмског окружења.



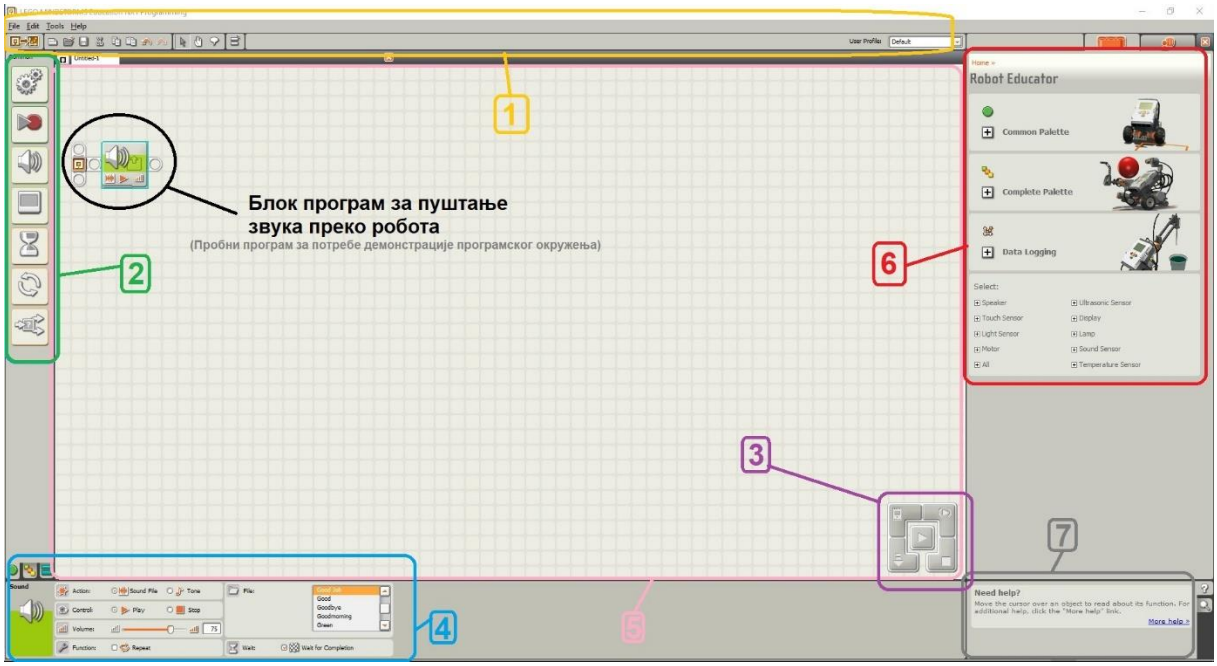
Слика 26. Процес инсталације NXT програмског окружења (трећи део).

3.2.1 Процес повезивања са рачунаром

Након исталације NXT LEGO Mindstorms 2.1 Софтвера са официјалног сајта LEGO, повезивање са рачунаром је прилично једноставно било да је у питању USB или Bluetooth конекција. Инсталацијом софтвера инсталирају се и сви потребни драјвери за препознавање NXT паметне коцке чиме је довољно само прикључити USB кабл на рачунар и NXT коцку и притиснути наранџасто дугме за покретање NXT коцке. Након пар секунди чује се звук на рачунару који означава да је NXT коцка конектована са рачунаром након чега је потребно сачекати пар минута да Windows оперативни систем конфигурише драјвере у поспуности. Када се овај процес заврши у доњем десном углу ће се појавити порука да је USB уређај успешно конфигурисан и спреман за употребу. Након тога NXT коцка се може програмирати преко NXT LEGO Mindstorms 2.1 софтвера. Повезивање преко Bluetooth-а је подједнако једноставно стим што је потребно пре повезивања на NXT коцки уђи у подешавања а затим у опцију Bluetooth и онда притиском на опцију „on“ омогућити повезивање. Након тога постоје две опције тражење рачунара преко коцке или тражење коцке преко рачунара од којих је друга опција знатно једноставнија. Приликом повезивања рачунара и NXT коцке преко Bluetooth конекције уколико је рачунар непознат коцки тачније уколико се први пут извршава веза, коцка ће тражити уношење индентификационог броја на рачунару који се приликом повезивања појављује на екрану коцке. Након уношења овог броја коцка препознаје рачунар као поуздан уређај чиме се веза може остварити. Након тога процес је готово исти као и помоћу USB конекције. Све што преостаје након тога је израдити програм и послати га на коцку. Детаљније о повезивању NXT коцке са рачунаром може се видети на [16] где је сликовно представљен цео процес у корацима. За потребе експеримента коришћена је USB веза као поузданија од обе могућности.

3.2.2 Увод у LEGO NXT програмско окружење

Приликом стартовања LEGO NXT Mindstorms програмског окружења појавиће се прозор са пар битних опција. Опције „Getting Started“ и „Software Overview“ нам пружају могућности брзог увода у основе коришћења софтвера и прегледа главних опција. Поред тога постоје опције „Start New Program“ која омогућава покретање новог пројекта за израду новог програма и „Open Recent Program“ која омогућава отварање већ постојећег програма за сврхе коришћења или дораде истог. Ове поције се могу видети на слици 26. позиција 14. Када покренемо нови пројекат програм нас пребацује у радно окружење за израду програма помоћу функцијских блокова. Израда програма је прилично једноставна. Све што треба урадити је изабрати жељени командни блок из палете функција са леве стране и превући је на мрежу. Након тога све што треба урадити је повезати блокове уколико их има више и конфигурисати по жељи и намени за функцију за коју су предвиђени након чега је програм спреман за слање на робота и његово тестирање. На слици 27. може се видети изглед радног окружења LEGO NXT програма.

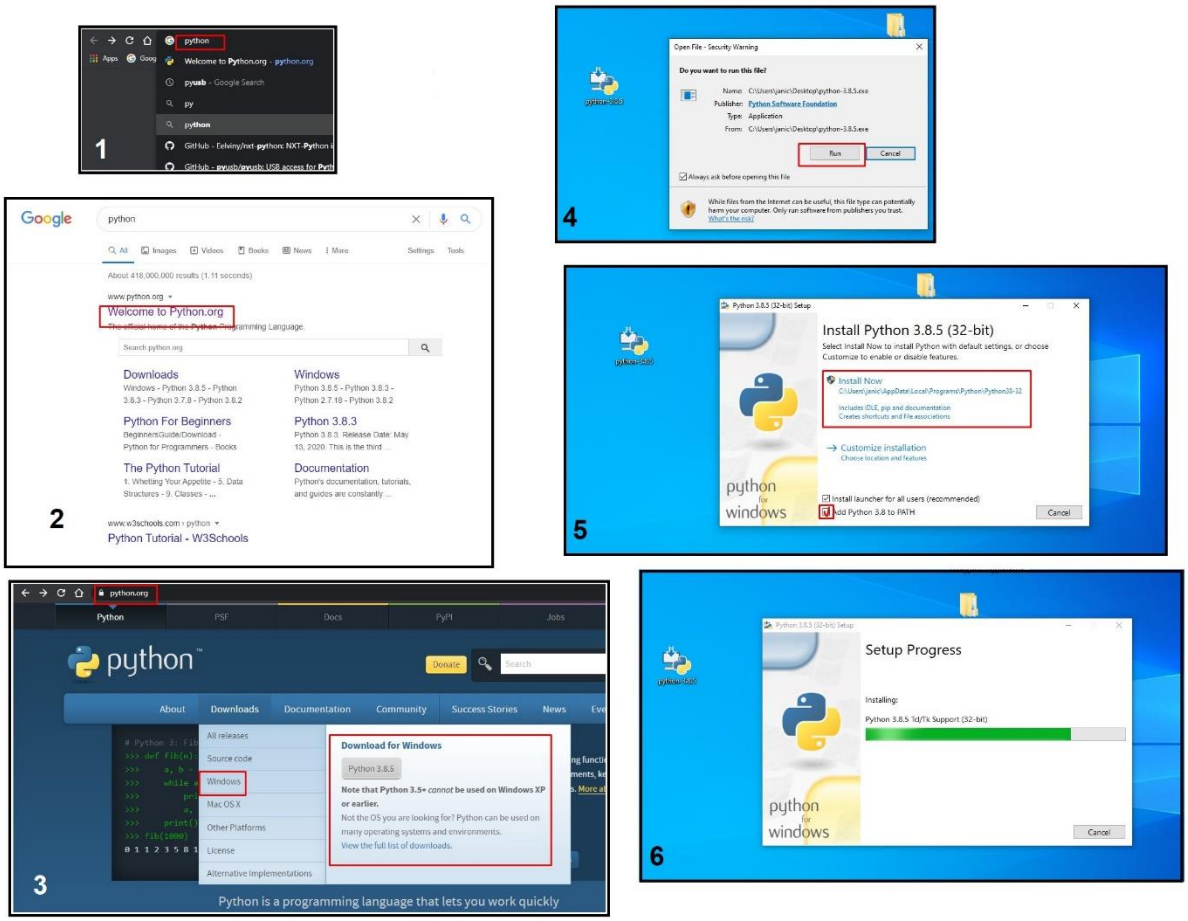


Слика 27. Изглед радног окружења LEGO NXT програма.

На слици 27. бројем 1 је означен мени са алатима у ком имамо опције да сачувамо програм, отворимо постојећи или нови програм, копирамо и додајемо компоненте из других програма, вршимо враћање корака напред и назад... Бројем 2 је означена палета функција програма из које узимамо блок функције од којих састављамо програме. Бројем 3 је означен контролер којим имамо функцију да шаљемо програм на NXT коцку, покрећемо и стопирамо програм. Бројем 4 означен је прозор за конфигурисање блок функција. Бројем 5 означена је мрежа на којој се склапа програм (главни радни простор). Бројем 6 означен је прозор који нам даје идеје за пројекте и првенствено је намењен за људе који почињу да раде са овим софтвером. Бројем 7 означен је прозор за помоћ којим можемо да нађемо решење за проблеме са радом програма. Нешто више о LEGO NXT програмском окружењу може се прочитати на [16] где је детаљније описано радно окружење програма. За потребе експеримента овај софтвер је коришћен искључиво са тестирање исправности NXT коцке, сензора и мотора због своје лакоће коришћења. Такође битно је напоменути да без обзира на могућности релативно лаког програмирања и коришћења LEGO робота, овај софтвер има веома широке могућности у извршавању већине замишљених функција.

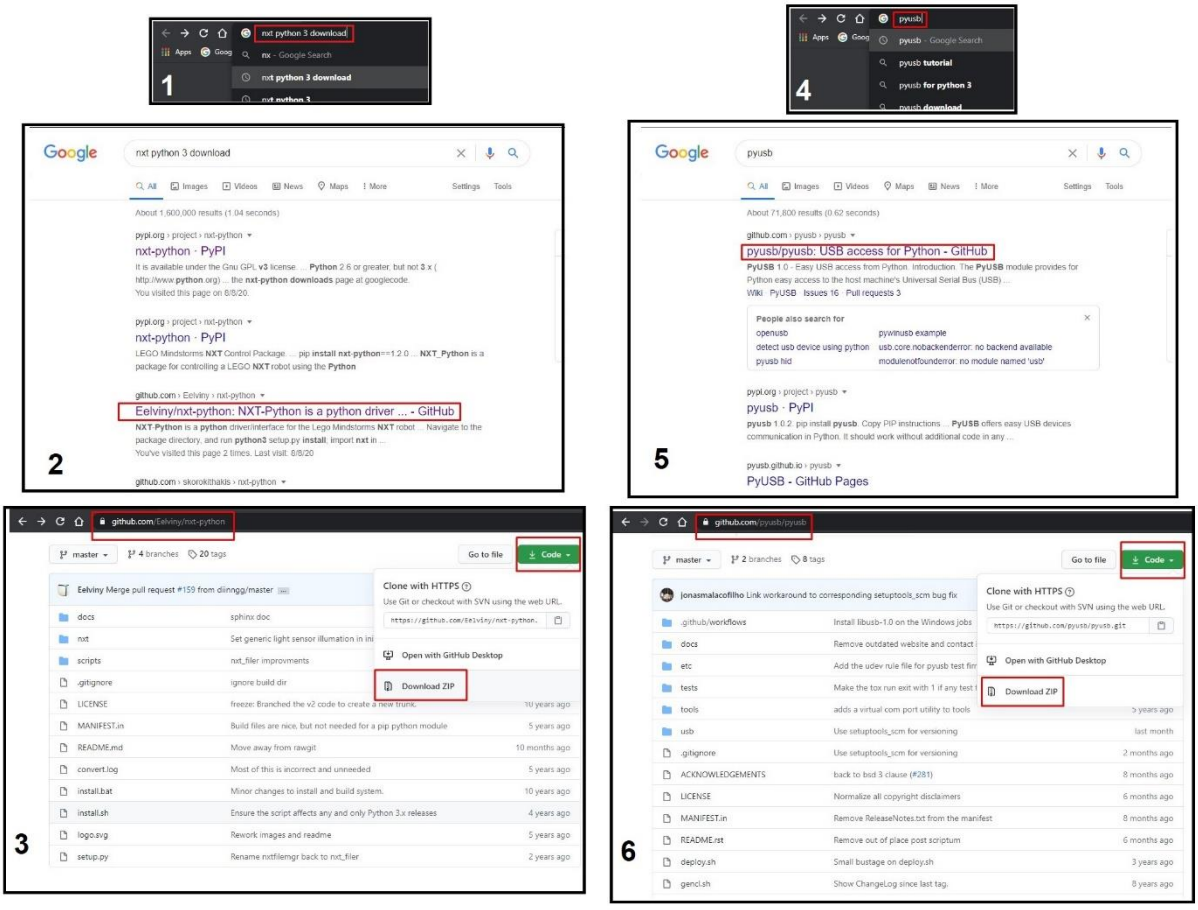
3.3 Процес инсталирања Python NXT програмског окружења

Да би се LEGO NXT коцка програмирала помоћу програмског језика Python пре свега је потребно инсталирати Python IDLE програмско окружење за писање кода. То се може урадити отварањем Web претраживача и на сајту Google укуцати „python“ као на позицији 1 са слике 28. и притиснути „Enter“. Након тога потребно је отворити „Welcome to Python.org“ сајт са позиције 2. Када се сајт отвори потребно је изабрати падајући мени „Download“ а затим оперативни систем који се користи на уређају са ког ће се програм користити након чега је потребно кликнути на опцију „Python 3.8.5“ или новију верзију у будућности што је приказано на позицији 3. Када се програм преузме потребно је покренути инсталацију и изабрати опцију „Run“ на прозору који се отвори (позиција 4). Након притиска на ову опцију појавиће се нови прозор у коме је битно чекирати квадратић поред ког пише „Add Python 3.8 to PATH“ а затим изабрати опцију „Install Now“ што се може видети на позицији 5. Након овога програм ће започети инсталацију након чега је програм спреман за употребу што се може видети на позицији 6. На слици 28. може се видети процес инсталирања Python IDLE програмског окружења. [20]



Слика 28. Процес инсталирања Python IDLE програмског окружења. [20]

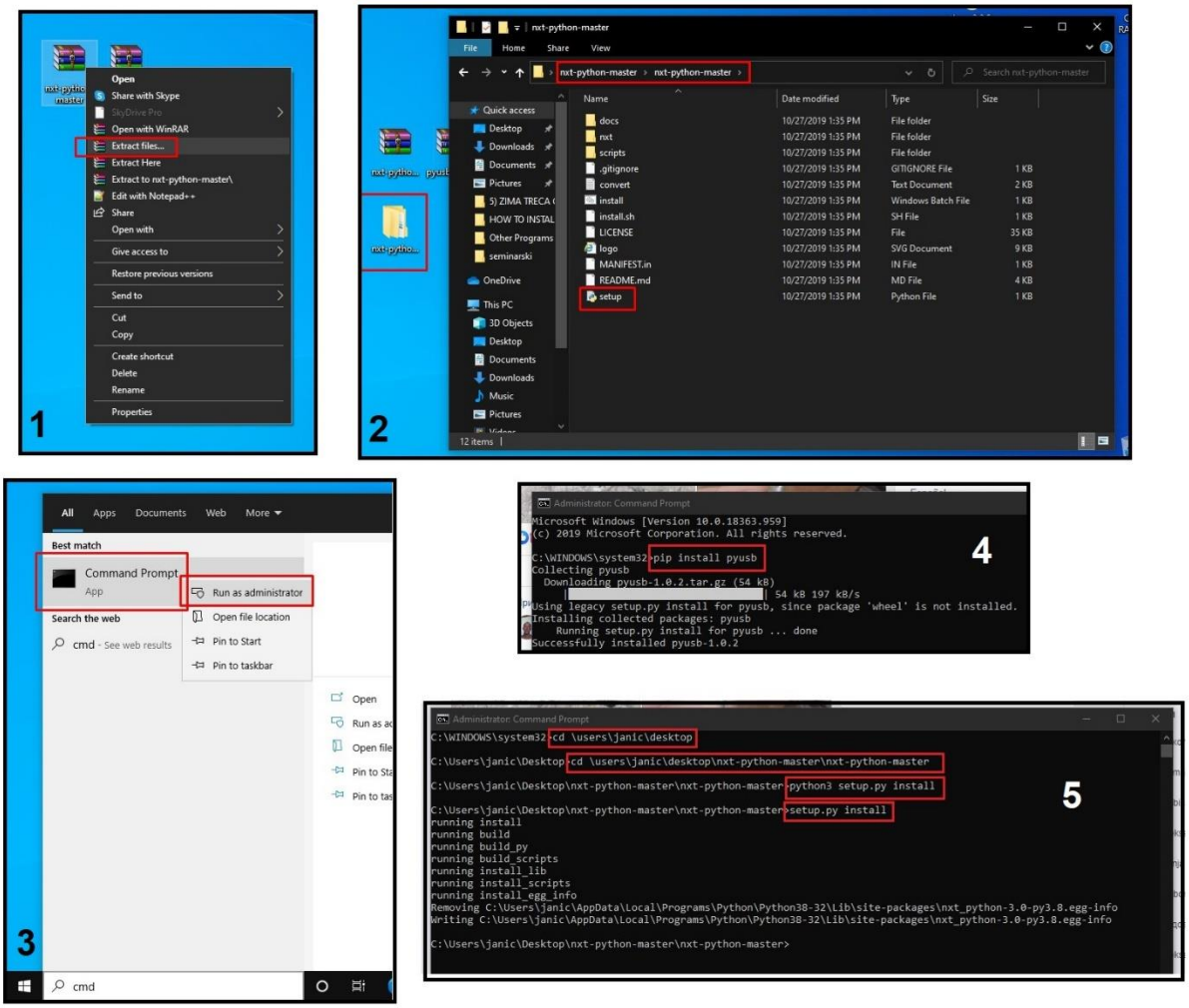
Поред Python IDLE програмског окружења потребно је инсталирати још пар софтверских пакета да би се омогућила могућност повезивања робота са програмом. Први софтверски пакет који је потребно инсталирати је NXT Python 3 што се може учинити отварањем Web претраживача и на сајту Google укуцати „nxt python 3 download“ што се може видети на позицији 1 на слици 29. Након извршене претраге потребно је изабрати опцију са GitHub.com сајта која је представљена на позицији 2. Када се сајт отвори у горњем десном углу је потребно изабрати опцију „Code“ а затим из падајућег менија одабрати „Download ZIP“ што се може видети на позицији 3. Други софтверски пакет који је потребно инсталирати је Py USB што се може учитини на исти начин као и за први програм стим што је у претрази потребно укуцати „pyusb“ и такође изабрати GitHub.com сајт након чега је процес потпуно исти што се може видети на позицијама 4, 5 и 6. На слици 29. може се видети процес преузивања NXT Python 3 и Py USB софтверских пакета. [21][22]



Слика 29. Процес преузимања NXT Python 3 и Py USB софтверских пакета. [21][22]

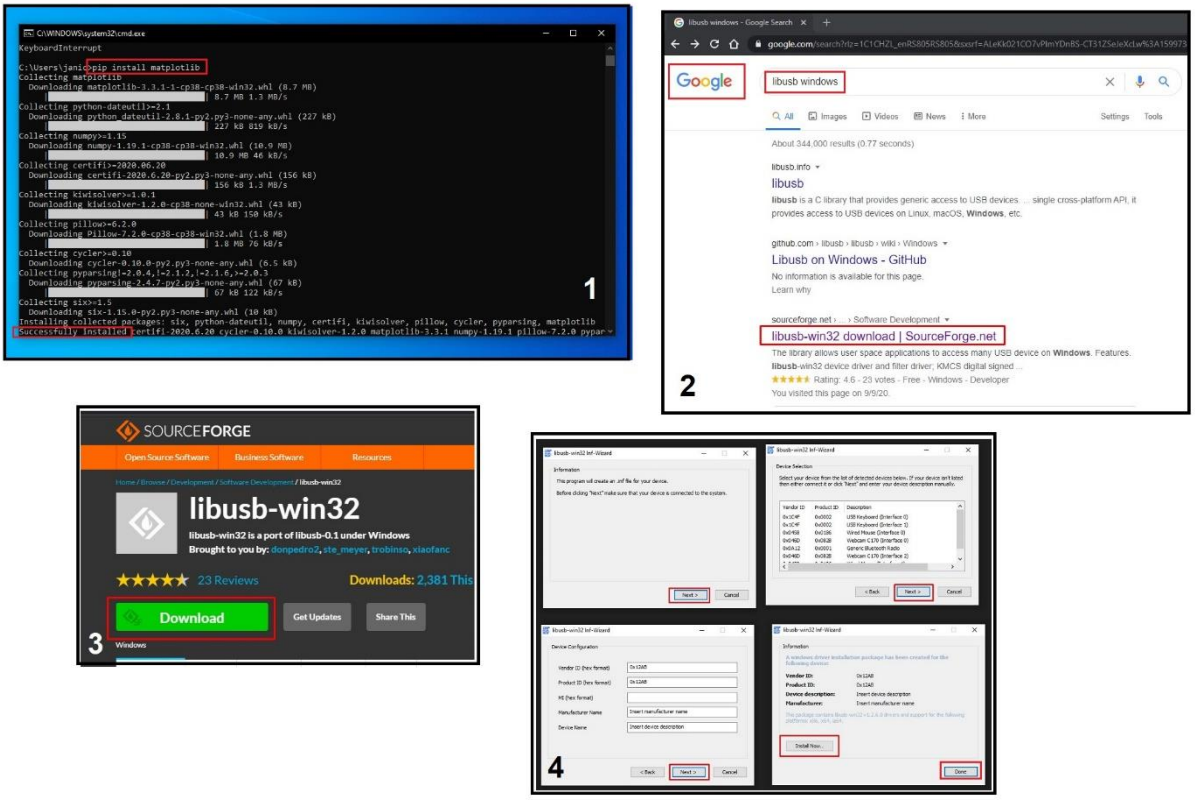
Након преузимања ова два софтверска пакета, потребно је распаковати .rar фајл у ком се налази инсталација за NXT Python 3. То се може учинити десним кликом на .rar фајл а затим изабрати опцију „Extract files…“ што је приказано на позицији 1 са слике 30. Након тога потребно је отворити фолдер који се појавио у коме се налази инсталација и погледати његову тачну локацију као што је то приказано на позицији 2. Након провере

локације инсталационих фајлова, да би се инсталирали софтверски пакети прво је потребно инсталирати Py USB а затим NXT Python 3. Они се инсталирају на следећи начин: Прво је потребно у пољу за претрагу у старт менију укуцати „CMD“ и покренути Command Prompt десним кликом и одабиром опције „Run as administrator“ што је приказано на позицији 3. Након стартовања командне конзоле потребно је укуцати „pip install pyusb“ и притиснути „Enter“ након чега ће започети инсталација Py USB софтверског пакета. Када се инсталација заврши у командној конзоли ће се појавити порука „Successfully installed pyusb-1.0.2“ као на позицији 4. Након овога потребно је рестартовати конзолу а затим када се поново стартује преко команде лоцирати локацију фолдера у ком се налази инсталација за NXT Python 3. У овом случају као што је то приказано на позицији 5 то је команда „cd \users\janic\desktop\nxt-python-master\nxt-python-master“. Након лоцирања инсталације у фолдеру све што је потребно урадити јесте унети команду „setup.py install“ што покреће инсталацију софтверског пакета NXT Python 3 и након приказа текста са позиције 5 софтверски пакет је инсталиран и NXT коцка је спремна за програмирање коришћењем програмског језика Python. На слици 30. може се видети процес инсталирања NXT Python 3 и Py USB софтверских пакета.



Слика 30. Процес инсталирања NXT Python 3 и Py USB софтверских пакета.

За представљање добијених вредности путем графика потребно је инсталирати Matplotlib који се инсталира на исти начин као и Py USB командом „pip install matplotlib“ у командној конзоли што се може видети на позицији 1 на слици 31. На одређеним рачунарима уколико дође до проблема са USB конекцијом потребно је инсталирати и Libusb који се може преузети са SourceForge.net сајта или сајта GitHub.com. Поступак инсталирања је следећи: Прво је потребно на сајту Google укуцати „libusb windows“ и од понуђених опција изабрати опцију са сајта SourceForge.net. На сајту је потребно изабарти опцију „Download“ и преузети инсталацију што се може видети на позицијама 2 и 3. Након тога потребно је распаковати .rar архиву као у случају са NXT Python 3 инсталацијом и покренути фајл „inf-wizard.exe“ из фолдера „bin“ а затим понављањем опције „Next“ и „Install“ инсталирати програм пошто не постоје никакве конфигурације током процеса инсталације што се може видети на позицији 4. Додатно око процеса инсталирања може се прочитати на [17] где је визуално описан процес инсталирања у Windows 7 оперативном систему. На слици 31. може се видети процес инсталирања Matplotlib и Libusb програма. [23]



Слика 31. Процес инсталирања Matplotlib и Libusb програма. [23]

4. Експеримент и резултати

Што се тиче експерименталног дела, задатак рада је да се постигне контрола брзине и позиције LEGO DC мотора и представе карактеристике мотора путем дијаграма помоћу програмског језика Python. Експеримент је вршен помоћу претходно напоменутог LEGO NXT самобалансирајућег робота са два LEGO DC мотора и сваки од корака експеримената је поновљен више пута како би се испитала тачност резултата и избегле евентуалне грешке са тог аспетка управљања. Програм се покреће притиском на тастер F5 у програмском окружењу IDLE након повезивања робота на USB. У наставку се може видети процес вршења експеримента као и добијени резултати из истог.

4.1 Статичка карактеристика за оба LEGO DC мотора

Да би се добила статичка карактеристика LEGO DC мотора потребно је написати програм који ће контролисати брзину мотора у зависности од снаге. Да би се добила таква карактеристика мотору се постепено повећава снага у процентима од 0% до 100% а затим се од 100% снага спушта на -100%. при чему за негативне вредности мотор мења смер окретања. Када мотор достигне вредност од -100% тада се снага смањује поново на 0% и мотор се зауставља. Промене снаге се врше у интервалима од по 10% на сваких 5 секунди. На слици 32. и 33. може се видети Python код програма коришћен за добијање статичке карактеристике мотора. [18]

Staticka_karakteristika.py - C:\Users\janic\Desktop\ZAVRSNI\FINAL\Staticka_karakteristika.py (3.8.5)

File Edit Format Run Options Window Help

```
__author__ = 'Nikola'
from matplotlib import pyplot as plt
import nxt
from nxt import *
import time
import kontrola
robot = nxt.locator.find_one_brick()
time.sleep(2)
a = kontrola.motor(robot)
m1 = Motor(robot, PORT_A)
m2 = Motor(robot, PORT_B)
teta1 = []
teta2 = []
omega1 = [0.0]
omega2 = [0.0]
snaga = [0]
snaga2 = 0
vreme = [0]
T = 1
tpocetno = time.process_time()
ttrenutno = 0
i = 0
a.iskljuci()
teta1.append(m1.get_tacho().block_tacho_count)
teta2.append(m2.get_tacho().block_tacho_count)
gsu = [teta1[0] - teta2[0]]
gsb = [0.0]
k = 0
while True:
    while ttrenutno + T > time.process_time() - tpocetno:
        pass
    ttrenutno += T
    if ttrenutno % 5 == 0:
        if ttrenutno <= 50:
            i += 1
            snaga2 = i * 10
#            snaga.append(i * 10)
#            vreme.append(ttrenutno)
        elif ttrenutno == 55:
            i = 1
            snaga2 = 90
#            vreme.append(ttrenutno)
#            snaga.append(90)
```

Слика 32. Python код програма за добијање статичке карактеристике мотора (први део). [18]

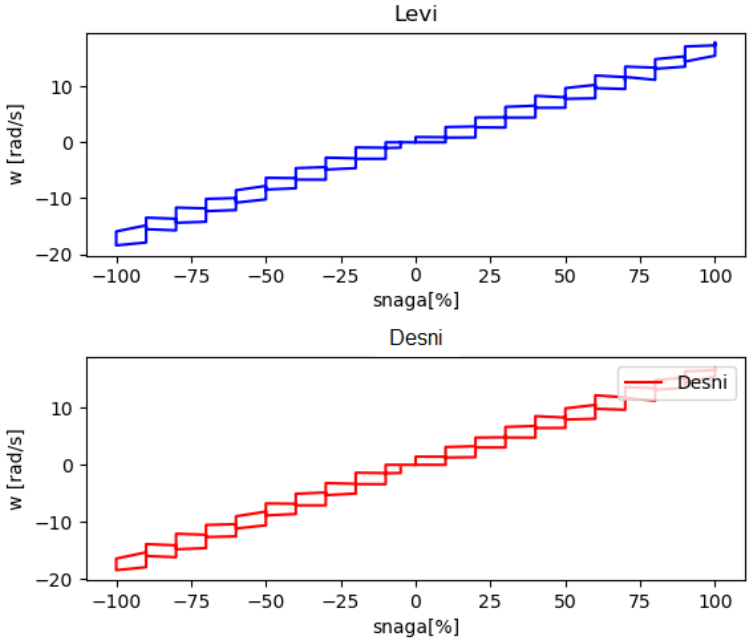
Staticka_karakteristika.py - C:\Users\janic\Desktop\ZAVRSNI\FINAL\Staticka_karakteristika.py (3.8.5)

File Edit Format Run Options Window Help

```
        elif ttrenutno <= 150:
            i += 1
            snaga2 = 100 - i * 10
#            vreme.append(ttrenutno)
#            snaga.append(100 - i * 10)
        elif ttrenutno == 155:
#            vreme.append(ttrenutno)
#            snaga.append(-90)
            i = 1
            snaga2 = -90
        elif ttrenutno <= 195:
#            vreme.append(ttrenutno)
            i += 1
            snaga2 = -100 + i * 10
#            snaga.append(-100 + i * 10)
        elif ttrenutno <= 200:
#            vreme.append(ttrenutno)
#            snaga.append(-5)
            snaga2 = -5
        else:
            break
    k += 1
    snaga.append(snaga2)
    vreme.append(k*T)
    a.brzina('A',snaga[k])
    a.brzina('B',snaga[k])
    teta1.append(m1.get_tacho().block_tacho_count * 3.1415926 / 180)
    teta2.append(m2.get_tacho().block_tacho_count * 3.1415926 / 180)
    omega1.append((teta1[k] - teta1[k-1])/T)
    omega2.append((teta2[k] - teta2[k-1])/T)
    gsu.append(teta1[k] - teta2[k])
    gsb.append(omega1[k] - omega2[k])
    print(omega1)
#    print ttrenutno
#    print snaga
#    print vreme
a.iskljuci()
print(snaga)
print(vreme)
print(len(snaga))
print(len(vreme))
snaga2 = []
f = open("podaci.txt", 'w')
for i in range(len(snaga)):
    f.write(str(vreme[i]) + " " + str(snaga[i]) + " " + str(teta1[i]) + " " + str(teta2[i]) + " " + str(omega1[i])
            + " " + str(omega2[i]) + " " + str(gsu[i]) + " " + str(gsb[i]) + '\n')
# Sastav datoteke:
# vreme, upravljacka promenljiva, ugao levi, ugao desni, brzina levi, brzina desni, razlika uglova, razlika brzina
f.close()
fig = plt.figure()
plt.subplot(211)
plt.title("Levi")
plt.xlabel("snaga[%]")
plt.ylabel("w [rad/s]")
plt.plot(snaga, omega1, 'b-')
plt.subplot(212)
plt.title("Desni")
plt.xlabel("snaga[%]")
plt.ylabel("w [rad/s]")
plt.plot(snaga, omega2, 'r-', label="Desni")
plt.legend(loc='upper right')
plt.show()
```

Слика 33. Python код програма за добијање статичке карактеристике мотора (други део). [18]

Након покретања програма и обављања његове функције, податци се чувају у .txt фајлу а график зависности се аутоматски појављује на екрану. Податци се чувају по колонама по следећем редоследу: Време, Снага, Угао првог мотора, Угао другог мотора, Брзина првог мотора, Брзина другог мотора, Грешка синхронизације по углу, Грешка синхронизације по брзини. При вршењу експеримента може се уочити да се точкови на роботу са повећањем снаге брже окрећу а са смањењем снаге спорије. На основу тога може се закључити да карактеристика брзине мотора у зависности од задате снаге треба да буде линеарна што се потврђује добијеним графиком. На графику брзина мотора означена је са W у радијанима по секунди [rad/s] на вертикалној оси а снага у процентима [%] на хоризонталној. На слици 34. може се видети график карактеристика брзине мотора у зависности од снаге.



Слика 34. График карактеристика брзине мотора у зависности од снаге.

4.2 Хармонијски одзив мотора

Што се тиче хармонијског одзива мотора, моторе је потребно покренути сигналом облика $B + Asin(kT\frac{2\pi}{\tau})$ у коме τ представља периоду синусног сигнала, B вертикални померај сигнала, A амплитуду, T периоду одабирања. Хармонијски одзив представља одзив система на различите побуде фреквенције. Вежба служи за одређивање пропусног опсега мотора NXT робота. Програм се покреће тастером F5 након чега се уносе вредности A, B, T и τ. На крају експеримента програм чува податке мерења у .txt фајл и показује график одзива мотора у радијанима по секунди [rad/s] у зависности од времена. Податци се по колонама чувају по следећем редоследу: Време, Снага, Угао првог мотора, Угао другог мотора, Брзина првог мотора, Брзина другог мотора, Грешка синхронизације по углу, Грешка синхронизације по брзини. На слици 35. и 36. може се видети Python код програма за добијање хармонијског одзива мотора. [18]

Frekvencijski_odziv.py - C:\Users\janic\Desktop\ZAVRSNI LAP\FINAL\Frekvencijski_odziv.py (3.8.5)
File Edit Format Run Options Window Help

```
__author__ = 'Nikola'
from matplotlib import pyplot as plt
import time
import math
import nxt
from nxt import *
import kontrola
omega1, omega2 = [0.0], [0.0]
teta1, teta2 = [], []
a, b, tau, T = float(input("Koeficijent a: ")), \
    float(input("Koeficijent b: ")),\
    float(input("Tau: ")), float(input("T: "))
snaga = []
print(int(tau/T))
robot = nxt.locator.find_one_brick()
x = kontrola.motor(robot)
time.sleep(2)
m1 = Motor(robot, PORT_A)
m2 = Motor(robot, PORT_B)
for i in range(int(tau / T) * 10):
    snaga.append(b + a*math.sin(i * T * 2 * math.pi / tau))
print(snaga)
```

Слика 35. Python код програма за добијање хармонијског одзива мотора (први део). [18]

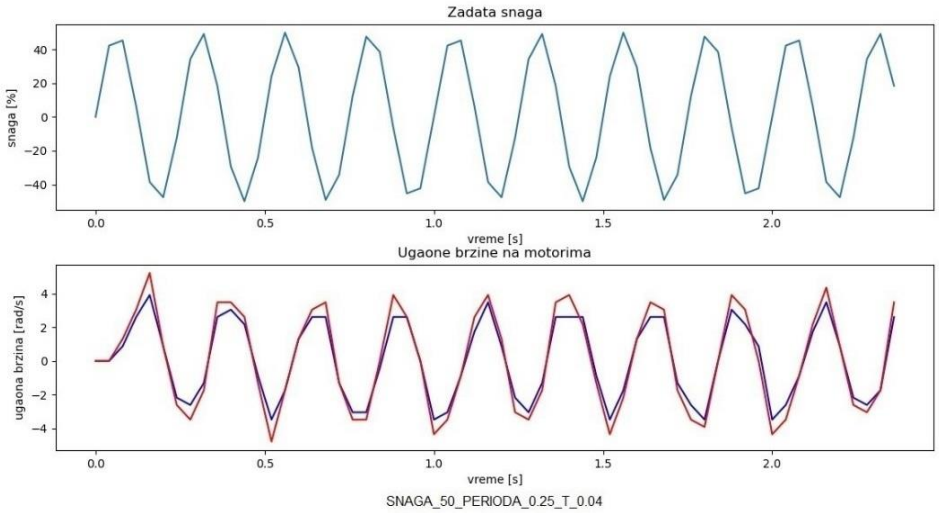
Frekvencijski_odziv.py - C:\Users\janic\Desktop\ZAVRSNI LAP\FINAL\Frekvencijski_odziv.py (3.8.5)

File Edit Format Run Options Window Help

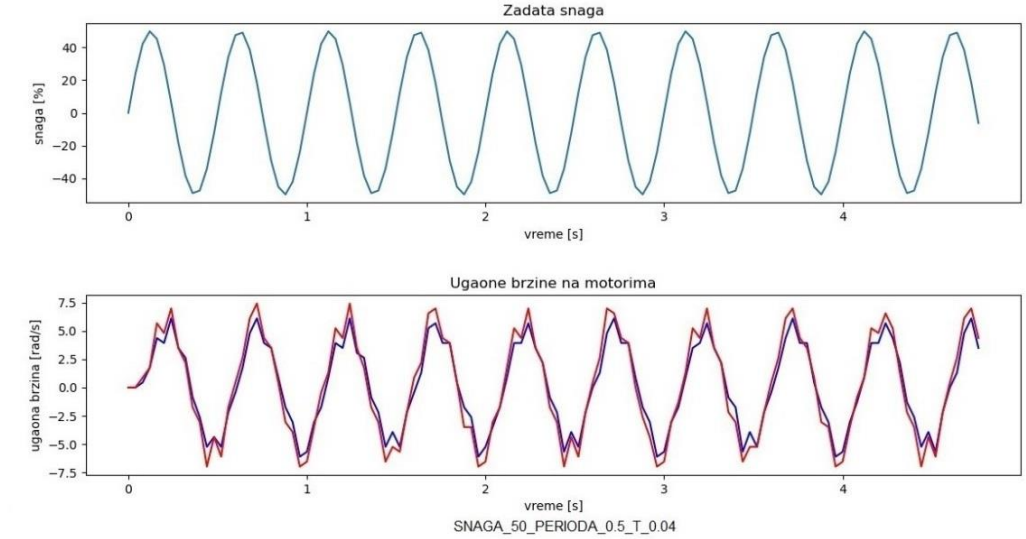
```
vreme = [0.0]
tpocetno = time.process_time()
ttrenutno = 0
i = 0
x.iskljuci()
time.sleep(T)
teta1.append(m1.get_tacho().block_tacho_count * 3.1415926 / 180)
teta2.append(m2.get_tacho().block_tacho_count * 3.1415926 / 180)
gsu = [teta1[0] - teta2[0]]
gsb = [0.0]
while True:
    while ttrenutno + T > time.process_time() - tpocetno:
        pass
    ttrenutno += T
    i += 1
    teta1.append(m1.get_tacho().block_tacho_count * 3.1415926 / 180)
    teta2.append(m2.get_tacho().block_tacho_count * 3.1415926 / 180)
    omega1.append((teta1[i] - teta1[i-1])/T)
    omega2.append((teta2[i] - teta2[i-1])/T)
    x.brzina('A',int(snaga[i]))
    x.brzina('B',int(snaga[i]))
    vreme.append(ttrenutno)
    gsu.append(teta1[i] - teta2[i])
    gsb.append(omega1[i] - omega2[i])
    if i == len(snaga) - 1:
        break
x.iskljuci()
f = open("SNAGA " + str(a) + " PERIODA " + str(tau) + ".txt", 'w')
for i in range(len(snaga)):
    f.write(str(vreme[i]) + " " + str(snaga[i]) + " " + str(teta1[i]) + " " + str(teta2[i]) + " " + str(omega1[i]) + " " +
            str(omega2[i]) + " " + str(gsu[i]) + " " + str(gsb[i]) + '\n')
f.close()
fig = plt.figure()
plt.subplot(211)
plt.xlabel("vreme [s]")
plt.ylabel("snaga [%]")
plt.title("Zadata snaga")
plt.plot(vreme, snaga)
plt.subplot(212)
plt.xlabel("vreme [s]")
plt.ylabel("ugaona brzina [rad/s]")
plt.title("Ugaone brzine na motorima")
plt.plot(vreme, omega1, 'b-', label='Levi motor')
plt.plot(vreme, omega2, 'r-', label='Desni motor')
plt.show()
```

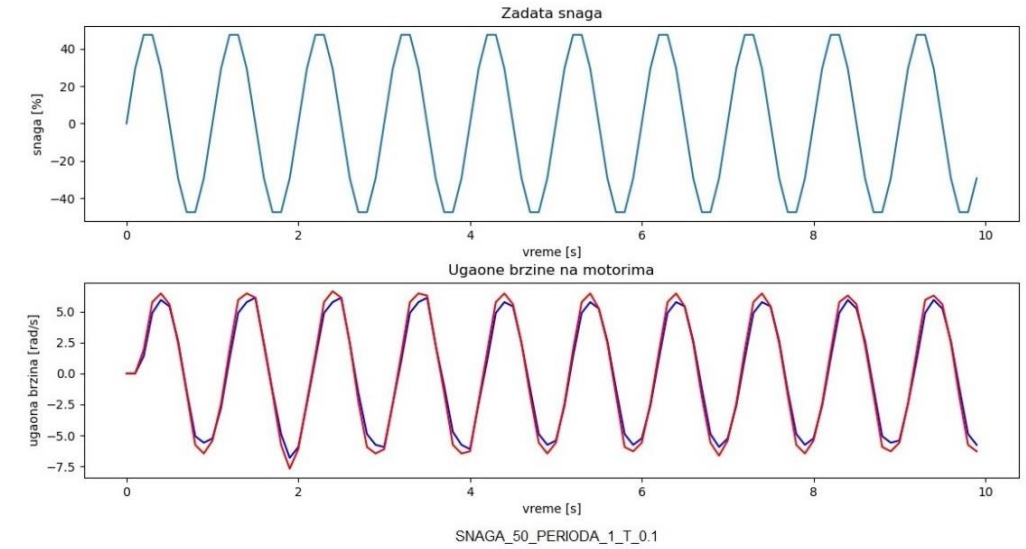
Слика 36. Python код програма за добијање хармонијског одзива мотора (други део). [18]

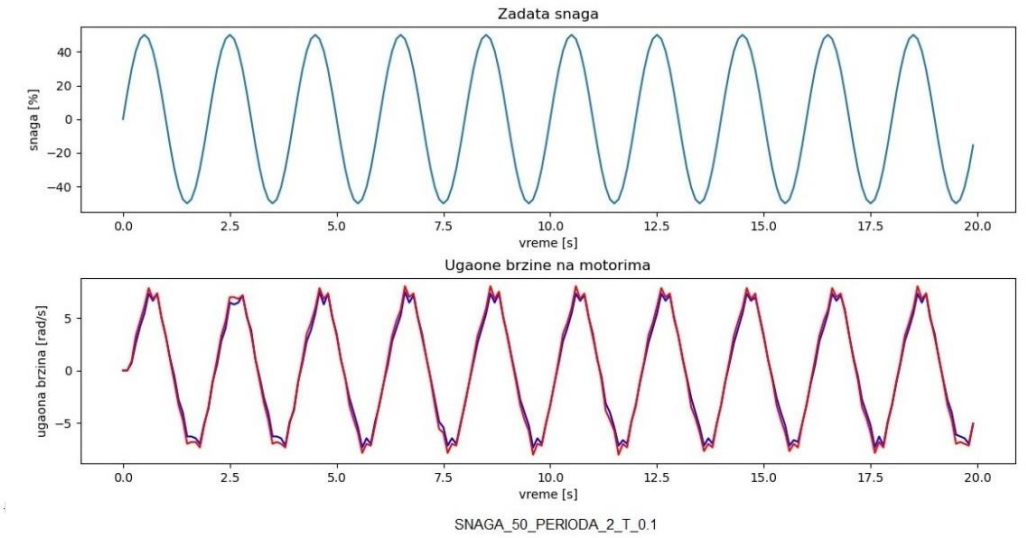
На слици 37. и 38. као и 39. може се видети хармонијски одзив мотора за различите задате параметре фреквенције (снаге, периоде синуса и периоде одабирања). На графицима означеним светло плавом бојом може се видети задата снага у процентима [%] која је за све вредности иста. Графици са плавом и црвеном бојом представљају угаоне брзине W [rad/s] за леви и десни мотор. На основу ових графика може се видети разлика између задатог синусног сигнала и стварног сигнала који се очитава помоћу уграђеног енкодера на мотору. Приликом извођења експеримента мотори су наизменично мењали смер окретања што се може видети и на основу добијених графика. Информације са ових графика могу помоћи у изради система са повратном спрегом помоћу PID контролера.



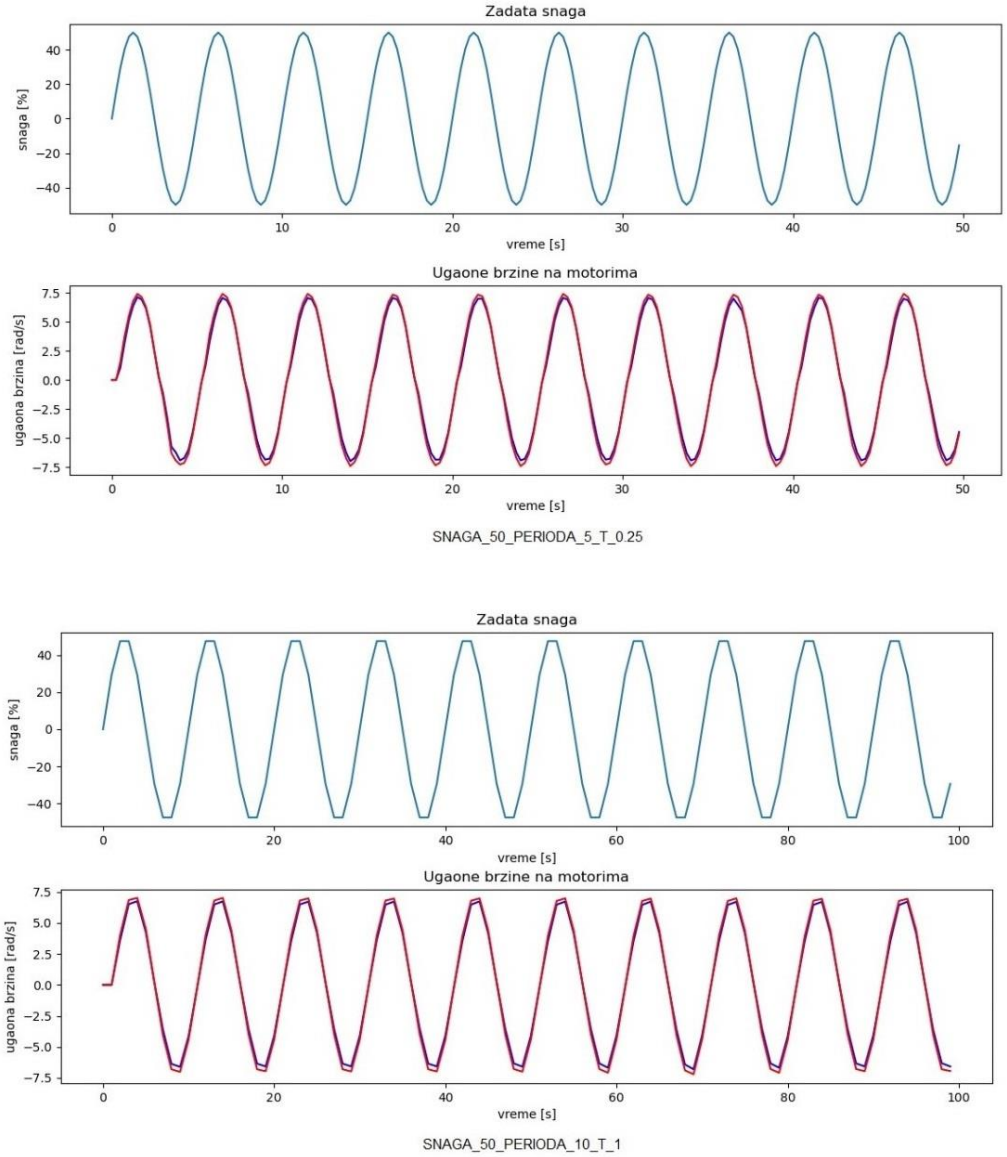
Слика 37. Хармонијски одзив мотора за задате параметре фреквенције испод графика.







Слика 38. Хармонијски одзив мотора за задате параметре фреквенције испод графика.



Слика 39. Хармонијски одзив мотора за задате параметре фреквенције испод графика.

4.3 Одзив на одскочну (Step) функцију

Одскочни одзив система нам служи за добијање динамичке карактеристике система управљања. На основу њих можемо добити велику количину битних информација о параметрима система управљања на основу којих се карактеристике система могу побољшати коришћењем различитих контролера. У експерименту задаје се снага мотора у виду одскочне функције са 0% на 60% и прати прелазни процес система благи прескок и осцилација мотора око жељене вредности са циљем доласка у стационарно стање. За потребе овог експеримента потребно је користити два различита Python кода. Први код нам служи за задавање снаге мотору у виду одскочног сигнала и снимање параметара док други код служи за добијање параметара система из снимљених података методом најмањих квадрата. Овај систем се може представити у облику функције преноса система

првог реда G(s) = $\frac{K}{Ts+1}$, а параметре K и Ts добијамо из снимљених података помоћу другог Python кода. На слици 40. може се видети Python код за задавање одскочног сигнама и снимање параметара мотора. [18]

Odskocni_odziv.py - C:\Users\janic\Desktop\ZAVRSNI LAP\FINAL\Odskocni_odziv.py (3.8.5)

File Edit Format Run Options Window Help

```
__author__ = 'Nikola'
import time
import nxt
from nxt import *
robot = nxt.locator.find_one_brick()
m1, m2 = Motor(robot, PORT_A), Motor(robot, PORT_B)
teta1, teta2, omega1, omega2 = [], [], [], []
T = 0.11
a, b = int(input("A:")), int(input("B:"))
tpocetno = time.process_time()
ttrenutno = 0
teta1.append(m1.get_tacho().tacho_count)
teta2.append(m2.get_tacho().tacho_count)
omega1.append(0)
omega2.append(0)
k = 0
m1.run(a)
m2.run(a)
uk = [a]
while True:
    while ttrenutno + T > time.process_time() - tpocetno:
        pass
    k += 1
    # ttrenutno += k*T
    ttrenutno = k*T
    if k < 101:
        u = a
    else:
        u = b
    teta1.append(m1.get_tacho().tacho_count * 3.1415926 / 180)
    teta2.append(m2.get_tacho().tacho_count * 3.1415926 / 180)
    omega1.append((teta1[k] - teta1[k-1])/T)
    omega2.append((teta2[k] - teta2[k-1])/T)
    uk.append(u)
    m1.run(u)
    m2.run(u)
    if k == 300:
        break
m1.brake()
m2.brake()
f = open("IdentAB.m", 'w')
for i in range(len(omega1)):
    f.write(str(uk[i]) + ", " + str(omega1[i]) + ", " + str(omega2[i]) + ", " + str(i*T) + "\n")
f.close()
```

Слика 40. Python код за задавање одскочног сигнала и снимање параметара мотора. [18]

На слици 41. и 42. може се видети Python код за добијање параметара система из снимљених података методом најмањих квадрата. [18]

Identifikacija_sistema.py - C:\Users\janic\Desktop\ZAVRSNI LAP\FINAL\Identifikacija_sistema.py (3.8.5)

File Edit Format Run Options Window Help

```
__author__ = 'Nikola'
import numpy
from numpy import *
from matplotlib import pyplot as plt
a = numpy.loadtxt("identAB.m", delimiter=',', usecols=(0, 2))
n1 = len(a)
ul = a[:, 0]
izl = a[:, 1]
izlnom = izl[30:99].mean()
ulnom = ul[30:99].mean()
print(izlnom)
print(ulnom)
ulprir = ul - ulnom
izlprir = izl - izlnom
print(ulprir)
print(izlprir)
N = n1-100
Y = numpy.zeros(N)
U = numpy.zeros(N)
YN = numpy.zeros(N)
tt = numpy.zeros(N)
'''Y = []
U = []
YN = []
tt = []'''
#FIN = numpy.zeros((2, 600))
for i in range(0, N):
    Y[i] = (izlprir[i+100-1])
    U[i] = (ulprir[i+100-1])
    YN[i] = (izlprir[i+100])
    tt[i] = (0.11*i)
FIN = numpy.matrix((Y.conj().T, U.conj().T))
FIN = FIN.H
print(numpy.shape(numpy.linalg.inv(numpy.dot(FIN.conj().T, FIN))))
```

Слика 41. Python код за добијање параметара система из снимљених података (први део). [18]

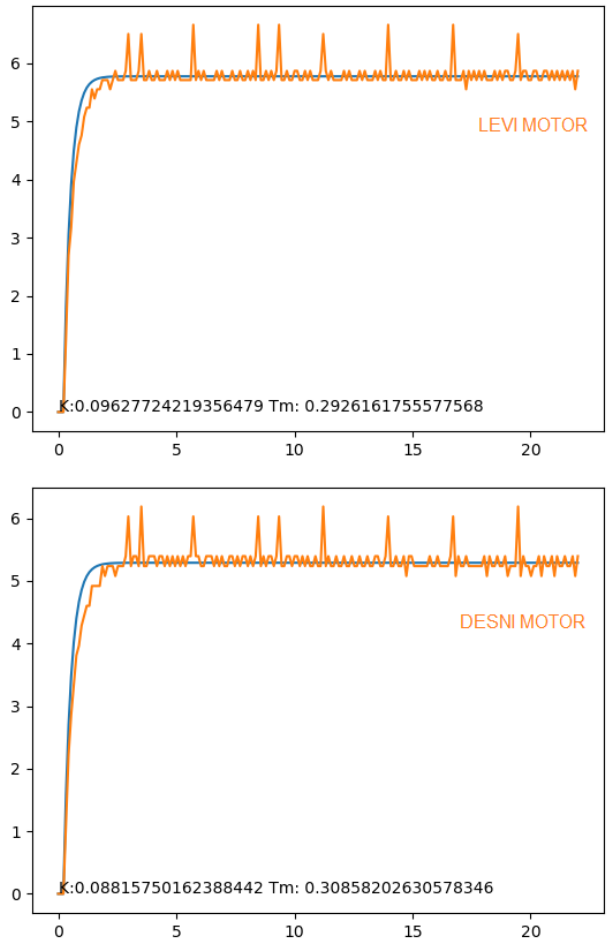
Identifikacija_sistema.py - C:\Users\janic\Desktop\ZAVRSNI LAP\FINAL\Identifikacija_sistema.py (3.8.5)

File Edit Format Run Options Window Help

```
pk = numpy.linalg.inv(numpy.dot(FIN.conj().T, FIN))
pk1 = pk * FIN.conj().T
TetaE = numpy.dot(pk1, YN)
aE = TetaE[0, 0]
KE = TetaE[0, 1] / (1 - aE)
Tm = -0.11 / math.log(aE)
y = []
for i in range(0, N):
    if i == 0:
        y.append(Y[i])
    else:
        y.append(y[i-1] * math.exp(-0.11 / Tm) + U[i-1] * KE * (1 - aE))
plt.plot(tt, y)
plt.plot(tt, Y)
plt.text(0,0,('K:' + str(KE) + ' Tm: ' + str(Tm)))
plt.draw()
plt.show()
```

Слика 42. Python код за добијање параметара система из снимљених података (други део). [18]

На слици 43. може се видети одзив на одскочну функцију за леви и десни мотор. Такође у самом дну графика могу се видети добијени параметри K и Ts (Tm). Наранџастом бојом је на графику означен стварни одзив мотора а плавом бојом жељена вредност задата одскочним сигналом. Вредност K за леви мотор износи K = 0,096 док Ts (Tm) износи Ts = 0,29. За десни мотор ове вредности износе K = 0,088 а Ts = 0,30



Слика 43. Одскочна функција за леви и десни мотор као и добијени параметри K и Ts (Tm).

4.4 Цртање Bode-овог дијаграма

Помоћу Bode-овог дијаграма можемо представити амплитудно фреквенцијску карактеристику за оба мотора. Такође помоћу њих могу се очитати резултати пропусног опсега који су добијени то јест учестаност на ком амплитудно фреквенцијска карактеристика пада на 70,7% своје вредности на учестаности ω = 0 што представља вредност од 3dB на Bode-овом дијаграму. При извођењу експеримента за добијање амплутидно фреквенцијске карактеристике мотору се задаје снага од 60% а циљ је исцртати Bode-ов дијаграм за систем првог реда облика G(s) = $\frac{K}{Ts+1}$. На слици 44. може се видети Python код коришћен за исцртавање Bode-овог дијаграма за оба мотора. [18]

Crtanje_Bode.py - C:\Users\janic\Desktop\ZAVRSNI LAP\FINAL\Crtanje_Bode.py (3.8.5)

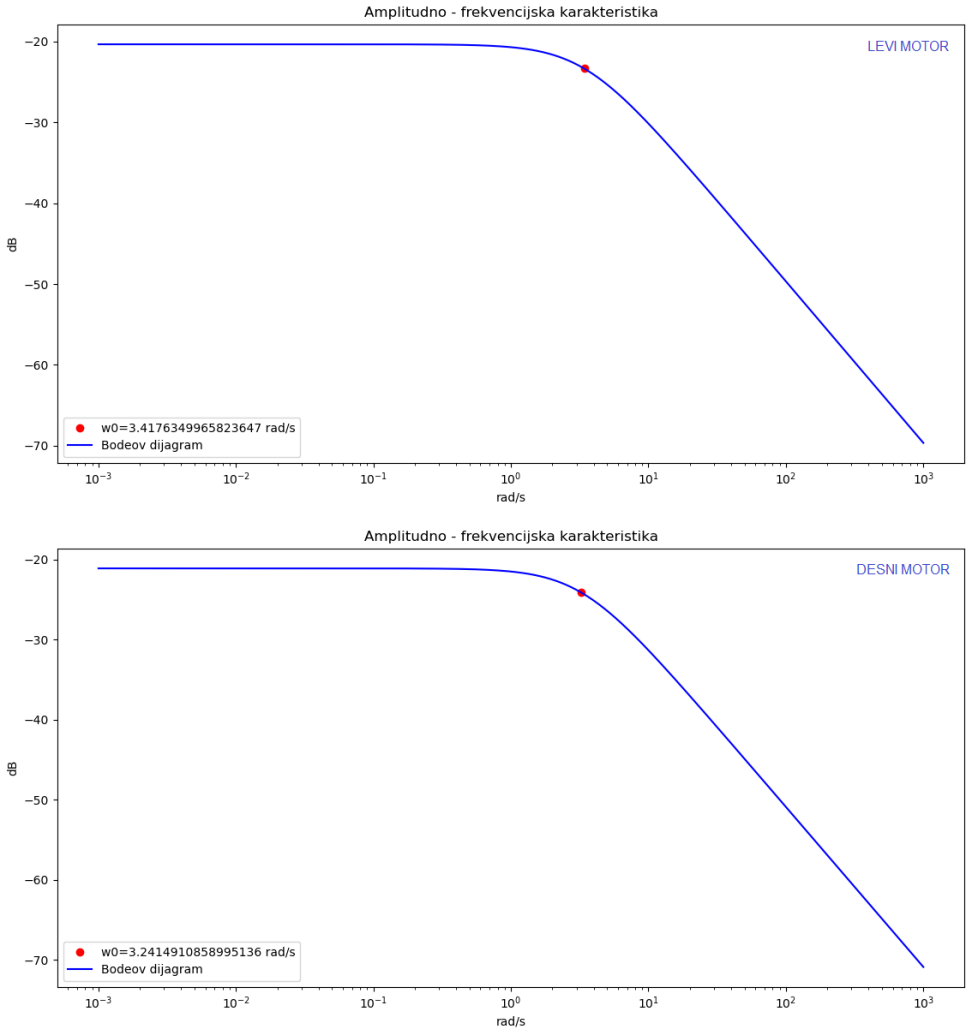
File Edit Format Run Options Window Help

```
__author__ = "Nikola"
import math
import numpy
from matplotlib import pyplot as plt
print ("+--------------------------+")
print ("|  Iscrtavanje bodeovog    |")
print ("|  dijagrama za sisteme    |")
print ("|   prvog reda oblika      |")
print ("|                          |")
print ("|                  K       |")
print ("|     G(s)=  --------      |")
print ("|             Ts + 1       |")
print ("+--------------------------+")
k = float(input("Pojacanje K: "))
ts = float(input("Vremenska konstanta Tm: "))
# Sistem je oblika K/(Ts+1) = K/(s/w0 + 1) => w0 = 1/T
# Pretvara se u oblik K/(jw0t+1)
# Odakle je arg(1/jwt) priblizno (1/wt)
# Ovde trazimo w0, da bi se nasao raspon x ose
w0 = 1.0 / ts
y = []
# x osa ide od w0/1000 do trenutka kada je pocetni signal smanjen za 60db (w0 + 1000*w0)
# 200 podeoka po dekadi
x = numpy.logspace(int(math.log10(w0))-3, int(math.log10(w0)+3), (3 + abs(int(math.log(w0)+3)))*200)
x = x.tolist()
for i in range(len(x)):
    y.append(20 * math.log10(k) - 10 * math.log10((x[i] * ts) ** 2 + 1))
plt.figure(211)
plt.title("Amplitudno - frekvencijska karakteristika")
plt.xlabel("rad/s")
plt.ylabel("dB")
p1, = plt.semilogx(w0, 20 * math.log10(k) - 10 * math.log10(2), 'ro', )
print("NF asimptota: " + str(y[0]))
print("Magnituda pri w0: " + str(20 * math.log10(k) - 10 * math.log10((w0 * ts) ** 2 + 1)))
p2, = plt.plot(x, y, 'b-')
plt.legend([p1, p2], [("w0=" + str(w0)) + " rad/s", "Bodeov dijagram"], loc=3, numpoints=1)
plt.draw()
plt.show()
```

Слика 44. Python код коришћен за исцртавање Bode-овог дијаграма за оба мотора. [18]

Овај код поред исцртавања Bode-овог дијаграма за оба мотора такође и рачуна и приказује учестаност на којој креће пад на амплутидно фреквенцијској карактеристици. Ова учестаност је у коду означена са w0 а вредност се приказује у доњем левом углу графика у радијанима по секунди (rad/s). За леви мотор добијена вредност учестаности на којој креће пад на амплитудно фреквенцијској карактеристици износи w0 = 3,41 rad/s а за десни мотор ова учестаност износи w0 = 3,24 rad/s. На основу претходно добијених вредности за Ts можемо израчунати пропусни опсег за оба мотора преко обрасца τ = 1/Ts на основу ког добијамо вредност за леви мотор која износи 3,448 rad/s као и десни мотор која износи 3,33 rad/s што је приближна вредност као вредност добијена Bode-овим дијаграмом и доказује тачност добијених дијаграма и представља одређени вид провере

рада и добијених резултата. На слици 45. може се видети добијен Bode-ов дијаграм за леви и десни мотор.



Слика 45. Bode-ов дијаграм за леви и десни мотор.

4.5 Контрола мотора помоћу PID контролера

Да би се побољшао одзив мотора на задати сигнал приликом управљања и добила што ближа вредност добијене карактеристике брзине и позиције жељеној задатој вредности, у овај систем је могуће имплементирати PID (Пропорционално, Интегрално, Диференцијални) контролер. Конкретно за управљање LEGO DC мотором довољно је користити P и PI регулацију да би се добили жељени параметри управљања а систем имао приближно идеалан одзив на задати сигнал. Конкретно у овом експерименту брзински параметри P и PI одређени су Ziegler-Nichols-овом методом а параметри позиције су одређени експериментално пошто су параметри позиције увек стабилни. За одређивање параметара брзине методом Ziegler-Nichols потребно је експериментално пронаћи критично појачање помоћу истог кода коришћеног у хармонијском одзиву а у

овом случају добијено је да то појачање износи Kkrit = 10 и Tkrit = 0.63. На основу њих се помоћу Python кода врло лако могу одредити параметри брзине мотора. На слици 46. може се видети Python код коришћен за добијање брзинских параметара P и PI регулатора. [18]

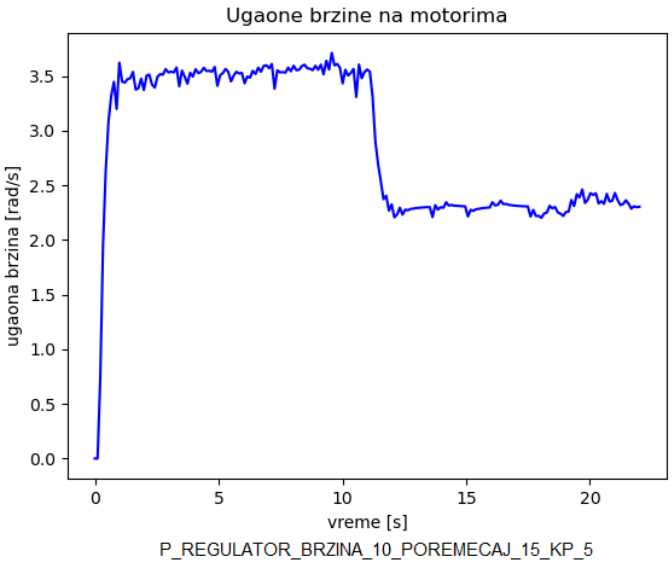
pi_regulator.py - C:\Users\janic\Desktop\ZAVRSNI LAPTOP\FINAL\pi_regulator.py (3.8.5)

File Edit Format Run Options Window Help

```
import time
import nxt
from nxt import *
from matplotlib import pyplot as plt
import kontrola
omega = 0
robot = nxt.locator.find_one_brick()
motor = Motor(robot, PORT_A)
theta = [0]
omega = [0,0]
T = 0.11
ct = kontrola.motor(robot)
a,b,kp,ki=float(input("Brzina:")),float(input("Poremecaj:")),float(input("Kp:")),float(input("Ki:"))
tpocetno = time.process_time()
ttrenutno = 0
theta.append(motor.get_tacho().block_tacho_count * 3.1415926/180)
k = 0
ct.brzina('A', 0)
uk = [a]
vreme=[0,0]
inte=0
try:
    while True:
        while ttrenutno + T > time.process_time() - tpocetno:
            pass
        tp = time.process_time() - tpocetno - ttrenutno
        uk.append(a)
        k += 1
        vreme.append(k*T)
        theta.append(float(motor.get_tacho().block_tacho_count) * 3.1415926/180.0)
        omega.append(0.33 * omega[-2] + 0.33 * omega[-1] + 0.33*(float(theta[-1] - theta[-2])/tp))
        e = uk[-1] - omega[-1]
        inte+=e*tp
        print("E:"+str(inte))
        if k < 101:
            u = kp*e + ki*inte
        else:
            u = kp*e + ki*inte - b
        print("U:"+str(u))
        ttrenutno = k*tp
        if u>100:u=100
        if u<-100:u=-100
        print(omega[-1])
        ct.brzina('A', int(u))
        #motor.run(int(u))
        if k==200:
            break
except e:
    print(e)
ct.brzina('A',0)
plt.xlabel("vreme [s]")
plt.ylabel("ugaona brzina [rad/s]")
plt.title("Ugaone brzine na motorima")
plt.plot(vreme, omega, 'b-', label='Levi motor')
plt.show()
```

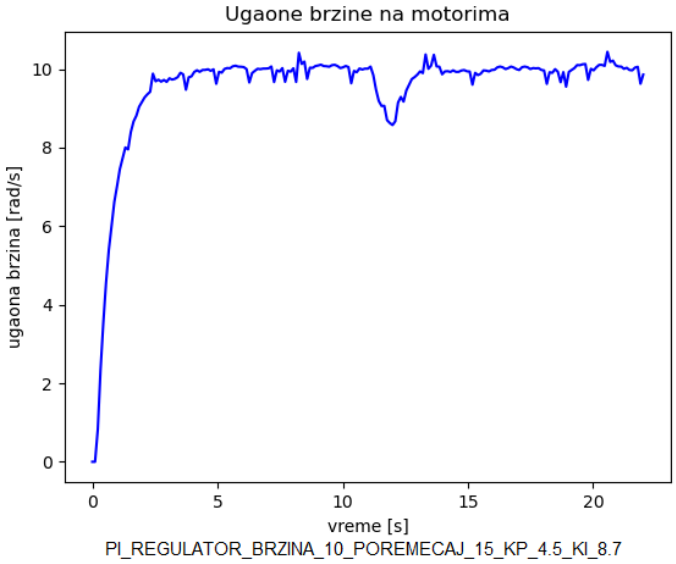
Слика 46. Python код коришћен за добијање брзинских параметара P и PI регулатора. [18]

Након покретања овог кода потребно је унети параметре Брзине, Поремећаја и P регулације за P контролу. У нашем случају задата угаона брзина мотора је 10 rad/s, поремећај је 15 а пропорционално појачање Kp = 5. Овим експериментом ће мотор помоћу P контроле покушати да достигне брзину од 10 rad/s а затим добити поремећај који може бити неки утицај спољашње средите у стварним условима коришћења што се у овом случају симулира. Са добијеног дијаграма може се видети да P регулација не успева да достигне жељену брзину мотора али без обзира на то након увођења дејства поремећаја успева да дође у стабилно стање са малим осцилацијама. На слици 47. може се видети дијаграм брзинске P регулације.



Слика 47. Дијаграм брзинске P регулације.

Такође за добијање дијаграма брзинске PI регулације потребно је унети параметре Брзине која износи 10 rad/s, поремећаја који износи 15 и у овом случају параметре P појачања које износи Kp = 4.5 и I појачања које износи Ki = 8.7. За разлику P регулатора код PI регулатора систем успева да достигне жељену угаону брзину мотора уз благе осцилације које се код обе регулације предпостављају да су последица деградације компонената током времена. Након увођења симулираног поремећаја, на графику се може видети благи пад брзине који PI регулација успева да отклони за релативно кратак временски период при чему се брзина поново враћа на жељену вредност. На слици 48. може се видети дијаграм брзинске PI регулације.



Слика 48. Дијаграм брзинске PI регулације.

Код експерименталне методе параметри P и PI регулације су добијени извођењем експеримента више пута на основу чега је закључено који параметри највише одговарају мотору. На слици 49. могу се видети Python код коришћен за добијање дијаграма позиционе P и PI регулације. [18]

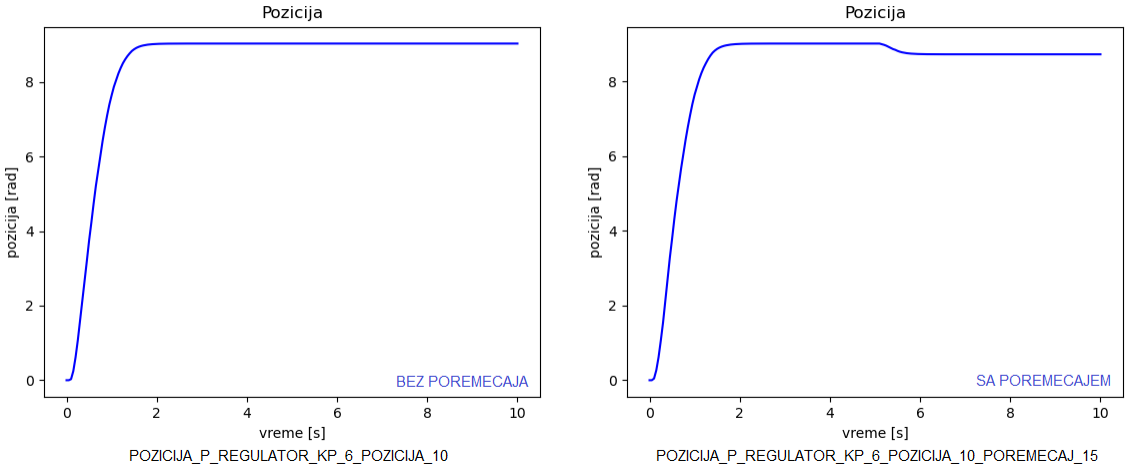
pi_pozicija.py - C:\Users\janic\Desktop\ZAVRSNI LAPTOP\FINAL\pi_pozicija.py (3.8.5)

File Edit Format Run Options Window Help

```
import time
import nxt
from nxt import *
from matplotlib import pyplot as plt
import kontrola
omega = 0
robot = nxt.locator.find_one_brick()
motor = Motor(robot, PORT_A)
motor.reset_position(True)
theta = [0]
omega = [0,0]
T = 0.05
ct = kontrola.motor(robot)
a,b,kp,ki=float(input("Pozicija:")),float(input("Poremecaj:")),float(input("Kp:")),float(input("Ki:"))
tpocetno = time.process_time()
ttrenutno = 0
theta.append(motor.get_tacho().block_tacho_count * 3.1415926/180)
k = 0
ct.brzina('A', 0)
uk = [a]
vreme=[0,0]
inte=0
try:
    while True:
        while ttrenutno + T > time.process_time() - tpocetno:
            pass
        tp = time.process_time() - tpocetno - ttrenutno
        uk.append(a)
        k += 1
        vreme.append(k*T)
        theta.append(0.33 * theta[-2] + 0.33 * theta[-1] + 0.33 * float(motor.get_tacho().block_tacho_count) * 3.1415926/180.0)
        e = uk[-1] - theta[-1]
        inte+=e*tp
        print("E:"+str(inte))
        if k < 101:
            u = kp*e + ki*inte
        else:
            u = kp*e + ki*inte - b
        print("U:"+str(u))
        ttrenutno = k*tp
        if u>100:u=100
        if u<-100:u=-100
        print(omega[-1])
        ct.brzina('A', int(u))
        #motor.run(int(u))
        if k==200:
            break
except e:
    pass
ct.brzina('A',0)
plt.xlabel("vreme [s]")
plt.ylabel("pozicija [rad]")
plt.title("Pozicija")
plt.plot(vreme, theta, 'b-', label='Levi motor')
plt.show()
```

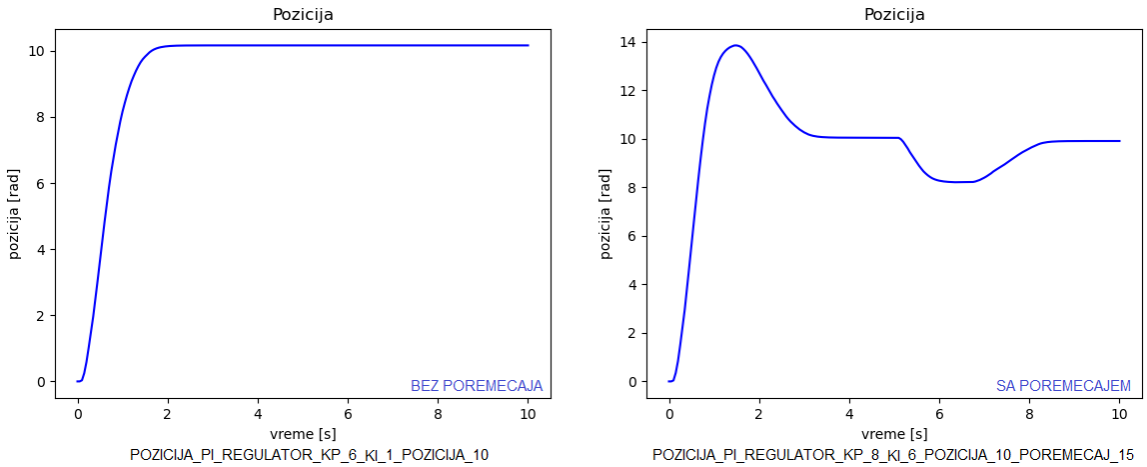
Слика 49. Python код коришћен за добијање дијаграма позиционе P и PI регулације. [18]

Након покретања кода потребно је унети параметре који се предпостављају да су добри и тиме понављати процес све док се не добију добри параметри P и PI регулатора. Овај процес добијања параметара се не препоручује код система управљања који има критична појачања јер може доћи до нестабилности система. Пошто у овом случају за позициону регулацију не постоје критична појачања систем не може доћи у стање нестабилности те се тиме може применити овај метод. За P регулацију позиције задати су параметри 10 rad са P појачањем Kp = 6 и поремећајем 15. Као код брзинске карактеристике може се приметити да мотор не може да постигне задате параметре са P регулацијом. Након увођења поремећаја може се видети да се позиција благо смањује али као што је напоменуто и даље не долази до нестабилности мотора или осцилација. На слици 50. може се видети дијаграм позиционе P регулације са и без поремећаја.



Слика 50. Дијаграм позиционе P регулације са и без поремећаја.

Што се тиче PI регулације задати су параметри позиције 10 rad, P појачања Kp = 6 и I појачања Ki = 1. У овом случају мотор одмах успева да достигне жељене параметре позиције без икаквих осцилација при чему је добијена једна веома идеална карактеристика. Као додатан део експеримента тестирано је шта би се десило уколико би променили параметре PI регулатора и додатно увели поремећај. У овом случају задати су параметри Kp = 8, Ki = 6, поремећај 15 док је жељена позиција остала иста. Са дијаграма можемо видети да се јавља прескок од 40% преко жељене вредности позиције након чега долази до постизања жељене вредности. Две секунде након што је жељена вредност постигнута уводи се поремећај који благо смањује позицију након чега се она веома брзо враћа на жељену вредност. Овим се на неки начин доказује стабилност мотора да испуни параметре позиције. Дијаграм са прескоком има много брже исправљање поремећаја од дијаграма без па је погоднији у случајевима где прескок може да се толерише. У случајевима где је битно да нема прескока и осцилација као код роботске руке, вредности са првог дијаграма су погоднije. На слици 51. може се видети дијаграм позиционе PI регулације са и без поремећаја.



Слика 51. Дијаграм позиционе PI регулације са и без поремећаја.

5. Закључак

На основу претходно обрађеног истраживања и експеримената можемо са сигурношћу закључити да LEGO Mindstorms технологија има веома велике потенцијале како у областима едукације тако и у областима истраживања на неким основнијим и мање захтевнијим нивоима. Широке могућности израде робота различитих облика, димензија и функција омогућавају примену у скоро свим замишљеним операцијама. Такође цена опреме је релативно ниска за разлику од неких других могућности на тржишту а технологија је веома приступачна и доступна широм света. Сама технологија је прилично једноставна за употребу тако да се и корисници који никад нису имали сличних искуства са сфером роботике контроле и управљања могу лако прилагодити и савладати исту. Такође можемо закључити да је технологија доста напредовала од својих почетака па све до сада, а тренутно поседује широку подршку како са хардверскох тако и са софтверског аспекта. LEGO сензори, мотори као и програмабилне коцке и додатна опрема који долазе у једном Mindstorms комплету су веома високог квалитета израде са прилично добрим хардверским карактеристикама. Поред тога софтвер Mindstorms има веома велике могућности уз веома лаку употребу а и додатни софтвери као што су LabVIEW, MATLAB као и програмски језици C, C++, Python се могу лако користити уз неке од додатних хардверских могућности као што су NI картице, Arduino контролери и NI ELVIS тренажери. Windows и Linux платформе су такође подржане као и контрола путем интернета уз израде специјалних Web интерфејса и система сервера помоћу AJAX-a и програмских језика Javascript, HTML, CSS и Python помоћу PC рачунара као и Raspberry Pi микрорачунара. Такође поред тога инсталирање потребних софтвера и припрема радног окружења као и израда самих LEGO робота је веома једноставна као што је приказано. Такође због веома прецизнор енкодера који се налази у DC мотору прецизно се могу одредити параметри LEGO једносмерног мотора помоћу Python кодова као што су статична карактеристика, хармонијски одзив мотора, одзив на одскочну функцију као и нацртати Bode-ов дијаграм помоћу ког се могу проверити неки од претходно добијених резултата. Поред тога на основу добијених параметара може се применити P и PI регулација система помоћу PID контролера за добијање што бољег одзива мотора за задате параметре. Све у свему закључак целог истраживања је да цела технологија има широке могућности и да у наредним годинама можемо очекивати само побољшања исте са изласком нових модела и потенцијалним падом цене као и повећаном применом у области едукације.

6. Литература

[1] М. Матијевић, Г. Јакуповић, Ј. Цар, „Рачунарски подржано мерење и управљање“, Универзитет у Крагујевцу, 2008. године.

[2] М. Матијевић, В. Цвјетковић, В. Филиповић, Н. Јовић, „Основни концепти аутоматике и мехатронике са LEGO програмабилним роботом“, Универзитет у Крагујевцу, 2014. године.

[3] Интернет страница „Roman’s Robots“, Линк адреса: http://www.romansrobots.com/product/robot-gladiators-lego-mindstorms-nxt-2-0-robotics-grades-3-6/, Приступљено: 19.08.2020. године.

[4] P. Trojánek, „Using the LEGO MINDSTORMS robot while learning“, Дипломски рад, Faculty of Electrical Engeneering Czech Prague, 2009. године.

[5] Интернет страница „Wikipedia“, Линк адреса: https://en.wikipedia.org/wiki/Lego_Mindstorms#:~:text=The%20Lego%20Mindstorms%20EV3%20is,elements%20and%20a%20remote%20control., Приступљено: 19.08.2020. године.

[6] Ж. Крстић, „LEGO роботика“, Мастер рад, Факултет инжењерских наука Крагујевац, 2017. године.

[7] Интернет страница „Philohome“ Линк адреса: https://www.philohome.com/nxtmotor/nxtmotor.htm, Приступљено: 21.08.2020. године.

[8] Р. Митровић, „Пројектовање управљачког кола роботске главе помоћу програма CMax“, Завршни рад, Факултет инжењерских наука Крагујевац, 2013. године.

[9] Интернет страница „Robotsquare“ Линк адреса: http://robotsquare.com/2013/07/16/ev3-nxt-compatibility/, Приступљено: 21.08.2020. године.

[10] T. Bělík, „Usage of the Lego Minstorms Robots Design and realization of the special tasks“, Дипломски рад, Faculty of Electrical Engeneering Czech Prague, 2010. године.

[11] M. Jaroslav, „Use of the LEGO MINDSTORMS robot – preparation robotic seminar for high school“, Дипломски рад, Faculty of Electrical Engeneering Czech Prague, 2010. године.

[12] D. Martinec, „Use of the LEGO MINDSTORMS robot during A3B99RO Robots“, Дипломски рад, Faculty of Electrical Engeneering Czech Prague, 2010. године.

[13] Интернет страница „Prorobot“ Линк адреса: https://www.prorobot.ru/lego/nxt_9797.php, Приступљено: 23.08.2020. године.

[14] Интернет страница „Travox Tripod“ Линк адреса: http://trivox.tripod.com/lego-nxt-motor-input-output.html, Приступљено: 23.08.2020. године.

[15] Интернет страница „Eneloop101“ Линк адреса: https://eneloop101.com/charge/eneloop-chargers/, Приступљено: 23.08.2020. године.

[16] Интернет страница „Generation Robots“ Линк адреса: https://www.generationrobots.com/media/Lego-Mindstorms-NXT-Education-Kit.pdf, Приступљено: 28.08.2020. године.

[17] М. Васковић, „Визуално упутство за инсталацију NXT Python-а“, Факултет инжењерских наука Moodle портал, Линк адреса: http://moodle.mfkg.rs/pluginfile.php/75782/mod_resource/content/0/Vizuelno_uputstvo_za_instalaciju_NXT_Pythona.pdf, 2014. године.

[18] Н Јовић, Сензори и актуатори Тема 5 и 6 Python кодови, Факултет инжењерских наука Moodle портал, Линк адреса: http://moodle.mfkg.rs/course/view.php?id=139, Приступљено: 04.09.2020. године.

[19] Интернет страница „Education LEGO“, Линк адреса: https://education.lego.com/en-us/downloads/retiredproducts/nxt/software, Приступљено: 28.08.2020. године.

[20] Интернет страница „Python“, Линк адреса: https://www.python.org/, Приступљено: 29.08.2020. године.

[21] Интернет страница „GitHub“, Линк адреса: https://github.com/Eelviny/nxt-python, Приступљено: 29.08.2020. године.

[22] Интернет страница „GitHub“, Линк адреса: https://github.com/pyusb/pyusb, Приступљено: 29.08.2020. године.

[23] Интернет страница „Source Forge“, Линк адреса: https://sourceforge.net/projects/libusb-win32/, Приступљено: 29.08.2020. године.