



**УНИВЕРЗИТЕТ “СВ. КИРИЛ И МЕТОДИЈ” -
СКОПЈЕ**



**ФАКУЛТЕТ ЗА ЕЛЕКТРОТЕХНИКА И
ИНФОРМАЦИСКИ ТЕХНОЛОГИИ**

- ДИПЛОМСКА РАБОТА -

по предметот

**ПРОЕКТИРАЊЕ НА РОБОТИЗИРАНИ ПРОИЗВОДНИ
ЛИНИИ**

Тема

**ПРОЕКТИРАЊЕ НА УПРАВУВАЊЕ НА
СЕРИСКИ МАНИПУЛАТОР СО 5 СТЕПЕНИ
СЛОБОДА НА ДВИЖЕЊЕ**

Ментор:
Проф. д-р Елизабета Лазаревска

Изработил:
Мартин Костовски, индекс бр. 77/2008
e-mail:mkostovski08@yahoo.com

Скопје, јануари 2015

СОДРЖИНА

| | |
|---|-----------|
| АПСТРАКТ | 4 |
| 1 ВОВЕД | 5 |
| 1.1 МАНИПУЛАТОРИ, ТИПОВИ МАНИПУЛАТОРИ И НИВНИ ПРЕДНОСТИ И НЕДОСТАТОЦИ | 5 |
| 2 ПОСТАПКА ЗА ПРОЕКТИРАЊЕ НА МАНИПУЛАТОР | 6 |
| 2.1 ИДЕНТИФИКАЦИЈА НА ПОТРЕБАТА ИЛИ ПРОБЛЕМОТ..... | 8 |
| 2.2 ИСТРАЖУВАЊЕ НА ПРОБЛЕМОТ ИЛИ НА ПОТРЕБАТА | 8 |
| 2.3 РАЗВИВАЊЕ НА МОЖНИТЕ РЕШЕНИЈА | 10 |
| 2.3.1 <i>Изработка на механичката структура</i> | <i>11</i> |
| 2.3.2 <i>Пресметка на параметрите потребни за управување</i> | <i>11</i> |
| 2.3.3 <i>Управување на манипулаторот</i> | <i>18</i> |
| 2.4 ИЗБОР НА НАЈДОБРОТО МОЖНО РЕШЕНИЕ | 36 |
| 2.5 КРЕИРАЊЕ НА ПРОТОТИП | 36 |
| 2.6 ТЕСТИРАЊЕ И ПРОЦЕНКА НА РЕШЕНИЈАТА..... | 36 |
| 2.7 СПОРЕДУВАЊЕ НА РЕШЕНИЈАТА..... | 37 |
| 2.8 РЕДИЗАЈНИРАЊЕ..... | 37 |
| 3 ПРАКТИЧЕН ДЕЛ | 37 |
| 3.1 ИДЕНТИФИКАЦИЈА НА ПОТРЕБАТА ИЛИ ПРОБЛЕМОТ..... | 38 |
| 3.2 ИСТРАЖУВАЊЕ НА ПРОБЛЕМОТ ИЛИ НА ПОТРЕБАТА | 38 |
| 3.3 РАЗВИВАЊЕ НА МОЖНИТЕ РЕШЕНИЈА | 39 |
| 3.3.1 <i>Изработка на механичката структура</i> | <i>39</i> |
| 3.3.2 <i>Пресметка на параметрите потребни за управување</i> | <i>40</i> |
| 3.3.3 <i>Управување на манипулаторот</i> | <i>42</i> |
| 3.4 ИЗБОР НА НАЈДОБРОТО МОЖНО РЕШЕНИЕ | 46 |
| 3.5 КРЕИРАЊЕ НА ПРОТОТИП | 56 |
| 3.6 ТЕСТИРАЊЕ И ПРОЦЕНКА НА РЕШЕНИЈАТА..... | 57 |
| 3.7 СПОРЕДУВАЊЕ НА РЕШЕНИЈАТА..... | 57 |
| 3.8 РЕДИЗАЈНИРАЊЕ..... | 58 |
| 4 ЗАКЛУЧОК..... | 58 |

Листа на слики

| | |
|--|----|
| Слика 1 МАНИПУЛАТОР СО СЕРИСКИ КРАЦИ | 5 |
| Слика 2 ПЛАТФОРМА НА СТУАРТ (ТИПИЧЕН ПРИМЕР НА ПАРАЛЕЛЕН МАНИПУЛАТОР) | 5 |
| Слика 3 БЛОК ШЕМА НА УПРАВУВАЧ НА РОБОТСКИ МАНИПУЛАТОР СО ПРИМЕНА НА ЈАКОБИЈАН..... | 15 |
| Слика 4 РОБОТ СО ДВА КРАЦИ (ПРВ ПРОБЛЕМ ШТО МОЖЕ ДА ПРОИЗЛЕЗЕ ДОКОЛКУ ПЛАНИРАЊЕТО НА ТРАЕКТОРИЈА СЕ ВРШИ САМО ВО ДЕКАРТОВ ПРОСТОР)..... | 20 |
| Слика 5 РОБОТ СО ДВА КРАЦИ (ВТОР ПРОБЛЕМ ШТО МОЖЕ ДА ПРОИЗЛЕЗЕ ДОКОЛКУ ПЛАНИРАЊЕТО НА ТРАЕКТОРИЈА СЕ ВРШИ САМО ВО ДЕКАРТОВ ПРОСТОР)..... | 20 |
| Слика 6 РОБОТ СО ДВА КРАЦИ (ТРЕТ ПРОБЛЕМ ШТО МОЖЕ ДА ПРОИЗЛЕЗЕ ДОКОЛКУ ПЛАНИРАЊЕТО НА ТРАЕКТОРИЈА СЕ ВРШИ САМО ВО ДЕКАРТОВ ПРОСТОР)..... | 21 |
| Слика 7 ЛИНЕАРНА ИНТЕРПОЛАЦИЈА | 24 |
| Слика 8 ЛИНЕАРНА ИНТЕРПОЛАЦИЈА СО ЗАКРИВУВАЊЕ НА КРАЕВИТЕ | 25 |
| Слика 9 ТРАЕКТОРИЈА СО ТРИ МЕЃУТОЧКИ | 26 |
| Слика 10 КРУЖЕН РОБОТ ВО ОКОЛИНА СО ОБЈЕКТИ..... | 30 |
| Слика 11 ПЛАНИРАЊЕ ВО КОНФИГУРАЦИСКИ ПРОСТОР | 31 |
| Слика 12 УПРАВУВАЊЕ ВО ПРОСТОРОТ НА ЗГЛОБОВИ..... | 32 |
| Слика 13 УПРАВУВАЊЕ ВО ДЕКАРТОВ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 33 |
| Слика 14 УПРАВУВАЊЕ ВО КАРТЕЗИЈАНСКИ ПРОСТОР СО КОРИСТЕЊЕ НА ДИРЕКТНАТА КИНЕМАТИКА И ИНВЕРЗНАТА МАТРИЦА НА ЈАКОБИЈАНОТ | 34 |
| Слика 15 ИЗРАБОТКА НА ЗАПЧЕНИК СО КОРИСТЕЊЕ НА АЛАТКАТА GEARTRAX | 39 |
| Слика 16 ИЗРАБОТЕН И СОСТАВЕН РОБОТСКИ МАНИПУЛАТОР СО 5 СТЕПЕНИ НА СЛОБОДА НА ДВИЖЕЊЕ | 40 |
| Слика 17 БЛОК ШЕМА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ОД СЛИКА 16 ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 43 |
| Слика 18 БЛОК ШЕМА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ОД СЛИКА 16 ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ДИРЕКТНАТА КИНЕМАТИКА И ИНВЕРЗНИОТ ЈАКОБИЈАН | 45 |
| Слика 19 БЛОК ШЕМА НА ПИД УПРАВУВАЧ | 46 |
| Слика 20 ПОЗИЦИЈАТА НА ЗГЛОБ 1 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 47 |
| Слика 21 ПОЗИЦИЈАТА НА ЗГЛОБ 2 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 47 |
| Слика 22 ПОЗИЦИЈАТА НА ЗГЛОБ 3 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 48 |
| Слика 23 ПОЗИЦИЈАТА НА ЗГЛОБ 4 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 48 |
| Слика 24 КООРДИНАТА x НА КРАЈНИОТ ИЗВРШЕН ЕЛЕМЕНТ ВО ДЕКАРТОВИ КООРДИНАТИ ДОБИЕНА СО СИМУЛАЦИЈА НА РОБОТСКИОТ МАНИПУЛАТОР УПРАВУВАН ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 49 |
| Слика 25 КООРДИНАТА y НА КРАЈНИОТ ИЗВРШЕН ЕЛЕМЕНТ ВО ДЕКАРТОВИ КООРДИНАТИ ДОБИЕНА СО СИМУЛАЦИЈА НА РОБОТСКИОТ МАНИПУЛАТОР УПРАВУВАН ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 49 |
| Слика 26 КООРДИНАТА z НА КРАЈНИОТ ИЗВРШЕН ЕЛЕМЕНТ ВО ДЕКАРТОВИ КООРДИНАТИ ДОБИЕНА СО СИМУЛАЦИЈА НА РОБОТСКИОТ МАНИПУЛАТОР УПРАВУВАН ВО ПРОСТОРОТ НА ЗГЛОБОВИ | 49 |
| Слика 27 ПОЗИЦИЈАТА НА ЗГЛОБ 1 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 50 |
| Слика 28 ПОЗИЦИЈАТА НА ЗГЛОБ 2 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 50 |
| Слика 29 ПОЗИЦИЈАТА НА ЗГЛОБ 3 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 51 |
| Слика 30 ПОЗИЦИЈАТА НА ЗГЛОБ 4 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 51 |
| Слика 31 ПОЗИЦИЈАТА НА ЗГЛОБ 5 ВО СИМУЛАЦИЈАТА НА УПРАВУВАЊЕ НА РОБОТСКИОТ МАНИПУЛАТОР ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 51 |
| Слика 32 КООРДИНАТА x НА КРАЈНИОТ ИЗВРШЕН ЕЛЕМЕНТ ВО ДЕКАРТОВИ КООРДИНАТИ ДОБИЕНА СО СИМУЛАЦИЈА НА РОБОТСКИОТ МАНИПУЛАТОР УПРАВУВАН ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 52 |
| Слика 33 КООРДИНАТА y НА КРАЈНИОТ ИЗВРШЕН ЕЛЕМЕНТ ВО ДЕКАРТОВИ КООРДИНАТИ ДОБИЕНА СО СИМУЛАЦИЈА НА РОБОТСКИОТ МАНИПУЛАТОР УПРАВУВАН ВО ДЕКАРТОВИОТ ПРОСТОР СО КОРИСТЕЊЕ НА ИНВЕРЗНАТА КИНЕМАТИКА | 52 |

| | |
|--|----|
| Слика 34 Координата z на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на инверзната кинематика | 52 |
| Слика 35 Позицијата на зглоб 1 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 54 |
| Слика 36 Позицијата на зглоб 2 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 54 |
| Слика 37 Позицијата на зглоб 3 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 54 |
| Слика 38 Позицијата на зглоб 4 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 55 |
| Слика 39 Позицијата на зглоб 5 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 55 |
| Слика 40 Координата x на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 55 |
| Слика 41 Координата y на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 56 |
| Слика 42 Координата z на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот | 56 |

Листа на табели

| | |
|--|----|
| ТАБЕЛА 1 Општо за собирање на информации за проектот што ќе се изработува | 9 |
| ТАБЕЛА 2 Параметрите на роботски манипулатор според DENAVIT - HARTENBERG МЕТОДАТА | 12 |
| ТАБЕЛА 3 Информации за проектот PICK AND PLACE роботски манипулатор..... | 38 |
| ТАБЕЛА 4 Параметрите на роботскиот манипулатор според DENAVIT - HARTENBERG МЕТОДАТА..... | 41 |

АПСТРАКТ

Сериските манипулатори се најчесто употребувани манипулатори во индустријата. Тие претставуваат низа од краци меѓусебно поврзани со зглобови. Кај сериските манипулатори најчесто се користат ротациони зглобови, но понекогаш, во некои специјални случаи, може да се сретнат и сериски манипулатори со транслаторни зглобови. Најчесто сериските манипулатори се состојат од 6 ротациони зглобови што ги прави манипулаторите да бидат со шест степени на слобода на движење. Тоа е така, за да може извршниот елемент на роботскиот манипулатор да се позиционира во било која точка од работниот простор и да постигне било која ориентација во таа точка, за што е неопходно роботскиот манипулатор да поседува најмалку 6 степени на слобода на движење. Во овој труд е претставен нов метод на проектирање на сериските манипулатори со користење на софтверски алатки што многу го олеснуваат не само начинот на проектирање на механиката на манипулаторот туку и начинот на проектирање на соодветен управувач за манипулаторот и симулација на добиените резултати. Најпрво моделот на роботски манипулатор е проектиран во CAD софтверот SolidWorks и потоа добиениот модел со соодветна алатка е преведен во SimMechanics модел и внесен во Simulink (Matlab), каде што е проектиран управувачот и е извршена симулација на перформансите на проектираниот модел и добиените резултати се претставени во детали. Во овој труд повеќе е обрнато внимание на употребата на SolidWorks и SimMechanics за проектирање и симулација на сериски манипулатори, но SolidWorks и SimMechanics може да се применуваат и за проектирање и симулација и на паралелни манипулатори како и за некои други апликации.

1 ВОВЕД

Во воведот на овој труд ќе дадам краток опис на тоа што се манипулатори, какви манипулатори постојат и кои се нивните предности и недостатоци.

1.1 Манипулатори, типови манипулатори и нивни предности и недостатоци

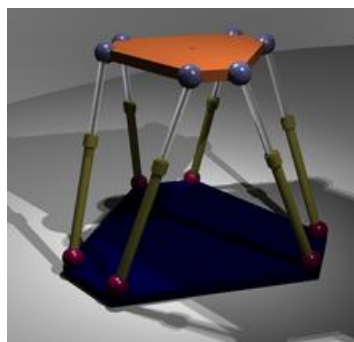
Индустриски манипулатор претставува репрограмабилен, мултифункционален уред проектиран за да може да движи предмети, делови, алатки или специјализирани уреди преку променливо програмирано движење за извршување на различни задачи.

Општо индустриските манипулатори може да се класифицираат во два типа според нивната конфигурација. Првиот тип на манипулатори се наречени манипулатори со сериски краци или накратко сериски манипулатори. Овој тип на манипулатори се состои од сериски подредени краци меѓусебно поврзани со зглобови, најчесто ротациони, а поретко транслаторни зглобови. Таков тип на манипулатор е претставен на Слика 1.



Слика 1 Манипулатор со сериски краци

Вториот тип на манипулатори се манипулатори со паралелни краци или пократко паралелни манипулатори. Тие се состојат од два дела: основна платформа или основа (неподвижна) и мобилна платформа (подвижна). Основната платформа е фиксирана за подлогата и не се движи, додека мобилната платформа се движи во однос на основата. Паралелните манипулатори имаат најмалку две основни платформи кои меѓусебно се поврзани со краци кои пак меѓусебно се поврзани со зглобови. Типичен пример на транслаторни манипулатори е платформата на Стјуарт прикажана на слика 2.



Слика 2 Платформа на Стјуарт (типичен пример на паралелен манипулатор)

Општо кажано двата типа на манипулатори имаат свои предности и недостатоци. Паралелните манипулатори се карактеризираат со поголема прецизност при позиционирањето, полесно проектирање, управување и можност за кревање на поголем товар. Дополнително кај паралелните манипулатори е возможно актуаторите да се лоцираат блиску до основата на механизмот и на тој начин се намалува инерцијата на подвижните делови на манипулаторот, а тоа дозволува користење помоќни актуатори. Од друга страна тие се помасивни, што им ја намалува брзината и ги прави енергетски понеефикасни. За подобрување на овој ефект кај паралелните манипулатори се користат лесни краци кои пак за таа сметка ја намалуваат прецизността поради појавата на вибрации и отклонувања. Затоа посебно за ваквите манипулатори од огромна важност се компјутерското проектирање и симулација.

Предност на сериските манипулатори е тоа што имаат поголем работен простор во однос на паралелните манипулатори, додека истовремено зафаќаат помалку простор. Исто така сериските манипулатори имаат предност и во пристапноста при обработка на работните предмети, бидејќи може да пристапуваат од различни страни до работната точка што може во некои случаи и да биде внатре во работниот предмет, додека паралелните манипулатори можат да пристапат до работната точка само по една од координатните оски што прави тие да не можат да заобиколуваат препреки. Потоа сериските манипулатори се енергетски поефикасни и имаат поголем опсег на различни работни задачи што може да ги извршуваат т.е. поголема флексибилност. Сериските манипулатори може да се поделат на мобилни манипулатори и немобилни манипулатори. Кај мобилните сериски манипулатори се овозможува драстично зголемување на работниот простор, подобрување на пристапноста кон работната точка, поголема флексибилност и се зголемува ефикасноста. Сепак сериските манипулатори се одликуваат со поголема сложеност во нивното моделирање, управување и симулација. За таа цел неминовна е потребата од примена на софтвер за потпомагање на процесите на проектирање на сериските манипулатори.

2 ПОСТАПКА ЗА ПРОЕКТИРАЊЕ НА МАНИПУЛАТОР

Откако веќе дадов краток вовед во тоа што се тоа манипулатори и какви типови на манипулатори постојат, сега ќе дадам краток вовед во постапката на нивно проектирање. Проектирањето на роботски манипулатори се одвива во 8 фази.

1. Идентификација на потребата или проблемот - Овде треба да се размисли која е потребата од роботски манипулатори или кој е проблемот што постои и треба да се реши со примена на роботски манипулатори.

2. Истражување на потребата или проблемот - Во оваа фаза се врши анализа на потребата од роботски манипулатор или анализата на проблемот што сакаме да го решиме со примена на роботски манипулатор. Овде треба да се прегледаат решенијата што постојат и да се види зошто тие не се соодветни решенија и за нашиот проблем или ако целта е да се креира комплетно нов проект, анализата на слични роботски манипулатори, сепак, може да биде многу корисна. Мора да се размисли за потенцијалната примена на проектот и според тоа да се определат перформансите барања што треба да ги задоволува проектот. Овде исто така треба да се постават и ограничувањата на проектот.

3. Развивање на можните решенија - За да се дојде до оптималното решение потребно е аналитичко размислување поврзано со креативност. CAD софтвер како што е на пример SolidWorks може да биде од посебна корист при овој процес бидејќи овозможува креирање на модел врз основа на идеите. Исто така ви овозможува

визуелна слика на проектот што може да води до зголемување на точноста и да води кон подобар производ.

4. Избор на најдоброто можно решение - Може да се користат неколку различни методи за оценување на некој поединечен дизајн. Може да се користи едноставен математички модел при кој се споредуваат предностите и недостатоците на решенијата. Вториот начин за оценување на дизајн е со споредба на определени параметри на дизајнот и со едноставна математичка споредба да се добие кое е оптималното решение. Овде од големо значење може да бидат софтверските алатки SolidWorks и SimMechanics бидејќи со нивна употреба би можело да се добијат квалитативни и квантитативни податоци за нашиот проект и поврзувајќи го тоа со метод за оценување на квалитет на проектот може да се определи кој е најдобриот дизајн.

5. Изработка на прототип - Понекогаш добивањето на прототип е преку користење на некоја машина со помош на која брзо може да се добие посакуваниот прототип. Вообичаено тродимензионалниот концепт лесно се претвора во дводимензионална скица. SolidWorks може автоматски да создаде дводимензионална скица на искористените елементи во моделот. Потоа може да се изработи прототипот според спецификацијата дадена во скиците. Исто така програмата за управувачот напишана во Matlab лесно може да се компајлира во извршен код кој што може да се сними во управувачот со што би го комплетираше прототипот.

6. Тестирање и проценка на решенијата - Овде се испитува дали решението (прототипот) ги задоволува поставените перформансни барања и дали се потребни некои промени во дизајнот. Ова најчесто се прави преку тестирање на прототипот во реална средина и анализа на неговото движење. Simulink овозможува поврзување на прототипот со компјутер при што покрај квалитативните, добиваме и квантитативни податоци за нашиот проект.

7. Споредување на решенијата - Во овој процес се врши презентирање на решението. Во оваа фаза, Solid Works и SimMechanics може да бидат од голема корист бидејќи овозможуваат да се изврши презентација на моделот преку симулација. Исто така SolidWorks и SimMechanics овозможуваат и полесна соработка бидејќи може лесно да се пренесуваат податоците и успешно да се комуницира помеѓу соработниците.

8. Редизајнирање - Се разгледува дали добиениот модел ги задоволува поставените барања. Ако не ги задоволува се изнаоѓа начин како може да се промени за да ги задоволи, а и во случај да ги задоволува можно е да постои начин на кој може и покрај тоа да се подобри дизајнот.

Сите овие чекори го сочинуваат процесот на проектирање и тие се повторуваат се додека не се изнајде соодветното решение.

Сите овие чекори при проектирање на индустриска машина одземаат многу време и многу често се подложни на грешки при пресметување, па затоа употребата на софтвер е од голема корист. Во овој труд јас ќе ја претставам употребата на SimMechanics софтверската алатка од Matlab за симулирање на управување на сериски роботски манипулатор со 5 степени слобода на движење. Најмногу ќе се задржам на процесот на проектирање на управувачот на тој робот, симулација на роботскиот манипулатор и приказ на добиените резултати.

2.1 Идентификација на потребата или проблемот

Овде треба да се размислува за тоа што е потребата или проблемот што сакаме да го решиме со примена на роботски манипулатор. Еве неколку примери за тоа:

1. Потребно е да се проектира и изгради роботски манипулатор што ќе се користи за бушење на дупки во препрег како еден процес во изработка на печатени електронски плочки.

2. Потребен е роботски манипулатор кој ќе се користи за фрезање на слотови, внатрешни исфрезувања или надворешна контура на печатени електронски плочки.

3. Потребно е проектирање и изградба на роботски манипулатор кој ќе може да врши асемблирање на електронски компоненти на печатена електронска плочка.

4. Потребно е проектирање и изградба на флексибилен роботски манипулатор кој ќе биде способен да врши повеќе различни операции како на пример: бушење, фрезање и асемблирање.

5. Заедницата сака да изгради робот за во зоолошка градина што ќе личи на животно што може да ја движи главата, да ја отвора устата и да произведува соодветни звуци кога ќе детектира дека некој му се приближува. Потребата е проектирање и изградба на прототип што ќе ја задоволува оваа потреба.

6. Локалната продавница на домашни миленичиња сака да продава разновиден спектар на уреди кои што автоматски ќе ги хранат животните што се чуваат во мали кафези (како што се зајаци, глувчиња итн.) кога нивните сопственици не се дома за време на викендите. Потребата е проектирање и изградба на прототип што ќе ги задоволи овие барања.

Потребно е прецизно да се дефинира што е проблемот што треба до се реши пред да се започне со проектирањето и изградбата на робот за решавање на тој проблем. Откако ќе потрошиме доволно време за изучување на повеќе различни ситуации и кога ќе се решиме која е нашата ситуација и разбереме што е точно проблемот, треба кратко да го опишеме проектот во тетратка или во електронска верзија (тоа ќе ни биде работниот документ додека работиме на роботот). Ова е краток документ што го објаснува проблемот што го решаваме.

2.2 Истражување на проблемот или на потребата

Во оваа фаза се врши анализа на потребата од роботски манипулатор или анализата на проблемот што сакаме да го решиме со примена на роботски манипулатор. Овде треба да се прегледаат решенијата што постојат и да се види зошто тие не се соодветни решенија и за нашиот проблем или ако целта е да се креира комплетно нов проект, анализата на слични роботски манипулатори, сепак, може да биде многу корисна. Мора да се размисли за потенцијалната примена на проектот и според тоа да се определат перформансните барања што треба да ги задоволува проектот. Овде исто така треба да се постават и ограничувањата на проектот.

Оваа фаза се состои од:

- * собирање на информации
- * идентификација на некои детали во проектирањето кои што мора да бидат задоволени
- * идентификација на можностите и алтернативни проектантски решенија
- * планирање и проектирање на соодветна структура која што вклучува скици

Откако ќе завршиме со краткиот опис на проблемот можеме да пристапиме кон собирање на информации кои што ќе ни помогнат во производството на успешен проект. Прво, потребно е да се одлучи кои информации ни се потребни. Тоа може да се разликува во зависност од проектот и исто така зависи од количината на информации и знаењето што веќе го имаме. Корисен чекор може да биде употребата на табела 1.

Табела 1 Општо за собирање на информации за проектот што ќе се изработува

| СОБИРАЊЕ НА ИНФОРМАЦИИ | |
|--|---|
| 1. Која е практичната функција на проектот? (Што роботот мора да биде во состојба да извршува?) | <p>Практичната функција на проектот може да вклучува:</p> <ul style="list-style-type: none"> * движење: Како роботот може да се движи во неговата околина? Доколку се постави во некоја друга средина, дали сеуште ќе биде во состојба да се движи во новодефинираниот простор? * манипулација: Како роботот ќе ги движи или ќе манипулира со другите објекти од неговата околина? * енергија: Како се напојува роботот? Може ли да поседува повеќе од еден извор на енергија? * интелигенција: Како роботот „размислува“? Што значи тоа да кажеме дека роботот „размислува“? * чувствителност: Како роботот добива информации за неговата околина? Ако го поставите роботот во друга околина дали сеуште ќе биде способен да добива информации за новата околина? |
| 2. Кој дел од неговиот изглед (облик, форма, големина, боја итн.) игра значајна улога во неговата проектантска функција? На што треба да наликува роботот? Има ли причина да изгледа онака како што изгледа? | <p>Обликот и формата се важни за естетските квалитети на проектот, ергономичноста, цврстината, стабилноста, крутоста и безбедноста</p> <p>Површинската текстура, механичките, оптичките и термичките својства, издржливоста итн.</p> |
| 3. Кои материјали се соодветни за проектот? | <p>Својствата на материјалите ќе ја определат погодноста на проектот. Во нашата работа во роботиката материјалите што најчесто се користат се алуминиум и железо. Меѓутоа, постојат повеќе типови на материјали што може да се користат во изградбата на роботите. Изборот на материјал што ќе се користи зависи од неговите својства: цврстина, тврдост,</p> |

| | |
|---|--|
| | грубост, густина, трајност, естетски квалитети како боја, површинска текстура, дезен, цената на материјалот, достапноста итн. |
| 4. Кој конструкциски метод е соодветен за проектот? | <p>Техниките на изградба влегуваат во следниве категории:</p> <ul style="list-style-type: none"> * сечење и острење * производство - составување на деловите со употреба на навратки и завртки, лепило, заварување, итн. * обликување - со примена на сила врз материјалот * лиење - употреба на калапи за добивање на потребниот облик <p>Одредени материјали може да се обработуваат со само некои постапки и ограничен број пати. Според тоа методата на обработка ќе биде определена од избраниот материјал, достапност на материјалите и производните капацитети, вештината на работната сила и трошоците на обработка.</p> |
| 5. Кои се најверојатно социјалните и еколошките ефекти на дизајнот? | <p>Производството, користењето и располагањето на секој производ ќе имаат и корисни и штетни ефекти врз животната средина. Затоа дизајнерот има голема одговорност и внимателно треба да ги разгледа потенцијалните ефекти на секој нов проект. Ова ги вклучува: здравјето, безбедност, бучавост, мирис, загадување итн.</p> |

Собирањето на информации може да вклучува читање, слушање, спроведување на интервју и набљудување.

2.3 Развивање на можните решенија

За да се дојде до оптималното решение потребно е аналитичко размислување поврзано со креативност.

Развивањето на можните решенија може да биде:

- * со нивна материјална изработка
- * со користење на некој CAD софтверски пакет (пр. Solidworks)

Развивањето на можни решенија со нивна материјална изработка подразбира материјална изградба на секое од можните решенија. Тоа одзема значително многу време и материјални трошоци. Користењето на CAD софтверски пакет овозможува

изработка на можните решенија во виртуелна реалност што значително го намалува потребното време и материјалните трошоци. Затоа современиот начин на развивање на можни решенија подразбира проектирањето на манипулаторите да биде со користење на CAD софтвер.

Развивањето на можните решенија се одвива во неколку фази:

1. Изработка на механичката структура
2. Пресметка на параметрите потребни за управување
3. Проектирање на управувањето.

Во оваа фаза пожелно е да се развијат повеќе можни решенија на дадениот проблем и потоа тие решенија да се споредат и од нив да се избере најдоброто решение.

2.3.1 Изработка на механичката структура

Оваа фаза на проектирање на роботски манипулатор одзема најмногу време. При оваа фаза најпрво треба да се изработи секој составен дел на роботскиот манипулатор поединечно и потоа составните делови се асемблираат во еден фајл. За таа цел се користи CAD софтвер како што е на пример SolidWorks.

2.3.2 Пресметка на параметрите потребни за управување

Во роботиката се среќаваат многу обемни и сложени пресметки. Тие освен што одземаат многу време за пресметување, исто така се подлежащи и на грешки при пресметувањето. Заради тоа од голема корист би било, доколку може, да се искористи компјутер за вршење на тие пресметки. За таа цел е создадена софтверската алатка Robotic toolbox која ги содржи сите потребни функции за вршење на оваа задача.

2.3.2.1 Кинематска анализа на манипулаторот

Во процесот на проектирање на управувачот најпрво е потребно да се изврши кинематска анализа на манипулаторот. Со помош на кинематската анализа на манипулаторот всушност се добива меѓусебната зависност на позицијата и ориентацијата на крајниот извршен елемент и променливите на зглобовите. Кинематската анализа на манипулаторот е поделена на два дела: директна кинематика и инверзна кинематика.

При директната кинематика ја добиваме трансформационата матрица која ја дава зависноста на позицијата и ориентацијата на крајниот извршен елемент од променливите на зглобовите. Всушност нам ни е потребно обратното, т.е. ако ни е позната позицијата и ориентацијата на крајниот извршен елемент потребно е да ги добиеме големините на аглите на зглобовите за да го позиционираме и ориентираме роботот во таа точка. Со други зборови, доколку ни се познати координатите на објектот што сакаме да го фатиме со грипелот на роботскиот манипулатор во однос на референтниот координатен систем, тогаш потребно е да се определи кои треба да бидат аглите на зглобовите на роботот за неговиот краен извршен елемент да се позиционира така што да го фати објектот. Тоа ни го дава инверзната кинематика.

2.3.2.1.1 Директна кинематика

Директната кинематика подразбира сложени пресметки. Вообичаено колку моделот има повеќе степени на слобода на движење, т.е. повеќе променливи на

зглобовите толку пресметувањето на директната кинематика е посложено. Освен тоа одзема и премногу време и е подложно на грешки при решавањето на сложените пресметки. Затоа во ваков случај многу подобро е користењето на софтвер.

За да ја најдеме положбата и ориентацијата на крајниот извршен елемент во зависност од положбите на аглите на зглобовите најпрво ќе го поделиме роботот на n цврсти тела. Во секое од нив ќе поставиме координатен систем цврсто врзан за цврстото тело. Доколку ни се познати положбата и ориентацијата на тој координатен систем тогаш познати ни се и положбата и ориентацијата на цврстото тело. Сега целта ќе биде да се определи позицијата и ориентацијата на подвижниот координатен систем поврзан за крајниот извршен елемент во однос на референтниот координатен систем. Тоа се прави со користење на Denavit - Hartenberg методата.

Пресметката со Denavit - Hartenberg методата се одвива на тој начин што прво се поставуваат подвижни координатни системи во секој од зглобовите на роботот, а потоа се составува табела што ги содржи параметрите на Denavit - Hartenberg.

Табела 2 Параметрите на роботски манипулатор според Denavit - Hartenberg методата

| i | d_i | θ_i | a_i | α_i |
|-----|-------|--------------|-------|------------|
| 1 | d_1 | θ_1^* | a_1 | α_1 |
| 2 | d_2 | θ_2^* | a_2 | α_2 |
| 3 | d_3 | θ_3^* | a_3 | α_3 |
| 4 | d_4 | θ_4^* | a_4 | α_4 |
| 5 | d_5 | θ_5^* | a_5 | α_5 |

Со помош на тие параметри и со користење на матрицата (1)

$${}_{i-1}T^i = \begin{bmatrix} \cos(\theta_i) & -\cos(\alpha_i) * \sin(\theta_i) & \sin(\alpha_i) * \sin(\theta_i) & a * \cos(\alpha_i) \\ \sin(\theta_i) & \cos(\alpha_i) * \cos(\theta_i) & -\sin(\alpha_i) * \cos(\theta_i) & a * \sin(\alpha_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

се добива трансформационата матрица за премин од координатниот систем поставен во $i - 1$ - виот зглоб во координатниот систем поставен во i - тиот зглоб и со множење за матриците како што е прикажано во равенката (2)

$${}_R T^n = {}_1 T^2 * {}_2 T^3 * \dots * {}_{i-1} T^i * {}_i T^{i+1} * \dots * {}_{n-1} T^n \quad (2)$$

се добива трансформационата матрица за премин од референтниот координатен систем во n - тиот координатен систем на роботот, а тоа е всушност координатниот систем поставен во крајниот извршен елемент. Добиената матрица (3)

$${}_R T^n = \begin{bmatrix} n_x & a_x & o_x & x \\ n_y & a_y & o_y & y \\ n_z & a_z & o_z & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

го претставува решението на директната кинематика на роботскиот манипулатор. Тројката (x, y, z) ги претставува координатите на позиција на крајниот извршен елемент на роботот во однос на референтниот координатен систем.

2.3.2.1.2 Инверзна кинематика

Целта на инверзната кинематика е да се определат аглите на зглобовите на роботскиот манипулатор за извршниот елемент да се позиционира во дадена точка од работниот простор на роботот.

Од резултатот од директната кинематика на роботскиот манипулатор познати ни се функциите x , y , z кои ја даваат зависноста на позицијата на крајниот извршен елемент во однос на позицијата на зглобовите на роботот. Тоа се три нелинеарни функции со n непознати. Во општ случај најдобро е ако може да се најде инверзна функција на x , y , z функциите. Но, роботот се карактеризира со сложена нелинеарност која што зависи од бројот на степени на слобода на движење на роботот. Поради таа нелинеарност наоѓањето на инверзната функција во општ случај е многу тешко, дури најчесто аналитичкото наоѓање на таа функција и не е можно. Токму поради тоа, решавање на инверзната кинематика се врши со помош на итеративна метода.

2.3.2.2 Јакобијан и анализа на сингуларитети

Првата задачи беше да се пресмета локацијата на крајниот извршен елемент во зависност од аглите на зглобовите и тоа се решава со кинематска анализа на роботскиот манипулатор. Значи со кинематската анализа се добива позицијата и ориентацијата на крајниот извршен елемент во однос на позицијата на зглобовите на манипулаторот (агол за ротациони зглобови и растојание за транслаторни зглобови). Сега ако се придвижи крајниот извршен елемент за мало растојание кое што може да се мери во текот на времето, т.е. ако почетната локација на крајниот извршен елемент T , беше определена со позиција на зглобовите q и со мало поместување на q за некое δq ќе дојде и до мало поместување на позицијата и ориентацијата на крајниот извршен елемент δT . Сега прашањето е кое е тоа δT ако се познати q и δq . Одговорот на тоа прашање е линеарна релација помеѓу δT и δq определена со матрицата на Јакобијан. Ако тоа поместување се земе да биде многу мало ќе се добие дека првиот извод од позицијата на зглобовите dq и првиот извод од поместувањето на крајниот извршен елемент dT се определени со матрицата на Јакобијан J . Доколку дополнително тоа равенство се подели со dt би се добила зависноста на брзината на движење на крајниот извршен елемент од брзината на зглобовите. Познато е дека локацијата на крајниот извршен елемент T вклучува позиција и ориентација, па во Јакобијанот ќе имаме релација што ја определува врската помеѓу линеарната брзина и агловата брзина на крајниот извршен елемент во однос на брзината на зглобовите на манипулаторот (аглова брзина за ротациони зглобови и линеарна брзина за транслаторни зглобови). Тоа математички може со релацијата (4)

$$\begin{bmatrix} v \\ w \end{bmatrix} = J * dq \quad (4)$$

каде што:

$$v = \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix}, w = \begin{bmatrix} \delta x \\ \delta y \\ \delta z \end{bmatrix} \quad (5)$$

Со разгледување на структурата на кинематската анализа на роботскиот манипулатор и нејзиното влијание врз брзината на крајниот извршен елемент се доаѓа до експлицитна форма на матрицата на Јакобијан. Со анализа на кинематиката може да се види дека секоја колона од Јакобијанот има врска со некој поединечен зглоб, на пример: ако се земе првата колона од Јакобијанот може да се забележи дека првата

колона е поврзана со влијанието на првиот зглоб врз брзината на крајниот извршен елемент, ако се земе во предвид втората колона од Јакобијанот може да се забележи дека таа е поврзана со влијанието на вториот зглоб врз брзината на движење на крајниот извршен елемент итн. Значи дека матрицата на Јакобијан ќе има толку колони колку што роботот има степени на слобода на движење (зглобови).

Оваа експлицитна форма ќе биде многу важна во основањето на модел што ги поврзува брзините на зглобовите и брзината на крајниот извршен елемент и овој модел исто така ќе биде многу значаен и во основањето на зависноста помеѓу силите. Силите или торзионите моменти што дејствуваат во зглобовите (сила за призматични зглобови или торзионен момент за ротациони зглобови) ќе предизвикаат сила или торзионен момент што дејствува во крајниот извршен елемент. Може да се покаже дека врската помеѓу силите и торзионите моменти што се појавуваат на крајниот извршен елемент доаѓаат од точно истиот модел на матрицата на Јакобијан т.е. има дуална релација помеѓу брзините и статичките сили, и тоа ќе доведе до основање на врската помеѓу вртливите моменти и силите што ги произведуваат актуаторите во зглобовите и силите и вртливите моменти што на таа сметка се појавуваат на извршниот елемент.

Доколку матрицата на Јакобијан е од ранг помал од најголемиот можен ранг се добива сингуларитет. Со други зборови сингуларитет е позицијата на роботскиот манипулатор во која ефекторот губи способност да се движи во насока на некоја од оските или да ротира во насока на некоја од оските. Таа оска се нарекува сингуларна оска.

Пресметката на Јакобијанот е од големо значење во роботиката, а тоа е затоа што Јакобијанот покрај тоа што може да се користи за пресметување на брзината на крајниот извршен елемент тој може да се користи и во управувањето на роботот. Всушност повеќето индустриски работи се управуваат токму со помош на Јакобијанот. Познато е дека Јакобијанот ја дава врската помеѓу мало поместување на крајниот извршен елемент и мало поместување во позицијата на зглобовите на манипулаторот т.е. тоа математички може да се претстави со релацијата (6).

$$\delta x = J(q)\delta q \quad (6)$$

Тоа значи дека ако ни е позната посакуваната позиција на крајниот извршен елемент x_k и моменталната позиција на извршниот елемент на манипулаторот што се добива со директна кинематика за моменталните вредности на зглобовите (можат директно да бидат мерени) на манипулаторот x_p , со едноставна разлика може да се пресмета потребното поместување на крајниот извршен елемент δx за тој да се најде во посакуваната позиција.

$$\delta x = x_k - x_p \quad (7)$$

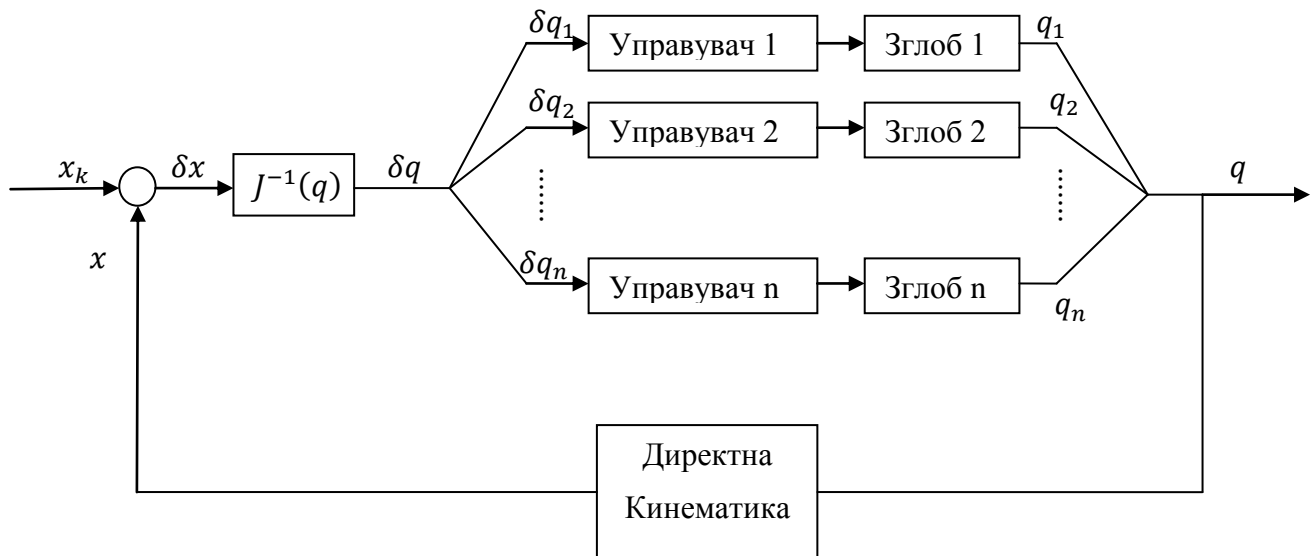
Сега со примена на матрицата на Јакобијан може да се определи потребното поместување во аглиите на зглобовите δq .

$$\delta q = J^{-1}(q)\delta x \quad (8)$$

Од равенка (8) се добива δq и бидејќи ни се познати вредностите на аглиите на зглобовите q со примена на равенката (9) се добиваат посакуваните вредности на позициите на зглобовите.

$$q_k = \delta q + q_p \quad (9)$$

Блок шемата на таквиот управувач е претставена на Слика 3.



Слика 3 Блок шема на управувач на роботски манипулатор со примена на Јакобијан

Во основа имаме тип на управувач кој што го користи Јакобијанот. Грешката во позицијата на крајниот извршен елемент се добива како разлика на посакуваната позиција на крајниот извршен елемент и моменталната позиција на крајниот извршен елемент добиена со директната кинематика и измерените вредности на аглите на зглобовите. Потоа со помош на Јакобијанот т.е. инверзната матрица на Јакобијанот се добива грешката во позицијата на аглите на зглобовите и со примена на управувач за следење на позиција во секој од зглобовите, може да се управува манипулаторот.

Ова претставува резултантно движење со степенско управување предложено од Whitney во 70 -тите. Ова е многу едноставен алгоритам. Ако не е од интерес динамиката на манипулаторот и ако не е од интерес управувањето на силите и контактите на роботот со околината тогаш ова ќе биде сосема задоволително решение. И секако неизбежно е да се работи во областа на работниот простор на манипулаторот со исклучок на сингуларните позиции и нивната оклина. Еден начин на кој може да се реши проблемот со сингуларитетите е со третирање на проблемот со секоја сингуларна позиција како посебен случај. Но, очигледно е дека ова нема многу добро да функционира во случај на роботски манипулатор што треба да следи траекторија бидејќи во тој случај би имале проблеми со забрзувањето и доколку сакаме да следиме одредено движење би наиделе на многу потешкотии, но сепак доколку имаме мали поместувања и мали брзини ова би функционирало.

Јакобијанот дополнително може да се користи и во пресметката на статичките сили. Со помош на овој Јакобијан може да се разберат силите што се добиваат на крајниот извршен елемент доколку се познати силите во трансляторните зглобови и торзионите моменти во ротационите зглобови на манипулаторот. Но исто така ќе видиме дека инверзната матрица на Јакобијанот е дел од релацијата помеѓу силите применети во крајниот извршен елемент и силите или вртливите моменти во зглобовите на манипулаторот. Тоа математички се изразува со релацијата (10).

$$\tau = J^T F \quad (10)$$

2.3.2.3 Лагранжова механика

Во овој поднаслов ќе биде прикажано како динамиката на роботската рака може да се претстави со матрична равенка од обликот (11)

$$M(q)\ddot{q} + V(q, \dot{q}) + G(q) = \tau \quad (11)$$

каде што:

q е матрица со елементи координати на зглобовите ($q = d$ е растојание за транслаторни зглобови и $q = \theta$ е агол за ротациони зглобови)

$M(q)$ е матрица на инерции на манипулаторот

$V(q, \dot{q})$ е матрица на центрифугални и Кориолисови сили

$G(q)$ е вектор на гравитациони сили

τ се збир на сите надворешни сили, вклучувајќи ги силите на триење и силите што ги произведуваат актуаторите на роботскиот манипулатор.

За да може успешно да се проектира управувач за манипулаторот од суштинско значење е изнаоѓањето на динамичкиот модел на манипулаторот. Тоа може да се направи со примена на:

1. Њутновата механика
2. Лагранжова механика

Примената на Њутновата механика во роботиката е можна, но премногу е сложена и одзема многу време. Дополнително, поради обемните пресметки многу е подложна на грешки. Заради тоа Њутновата механика не се користи многу во роботиката. Во оваа дипломска работа, при пресметувањето на динамичката равенка е користена Лагранжовата механика, која што е многу поедноставна за посложените примери. Секако и при примената на Лагранжовата механика има обемни пресметки но благодарение на големата пресметувачка моќ на компјутерите лесно се доаѓа до решението и можноста да се направи грешка при пресметките е сведена на минимум.

Пресметувањето со Лагранжова механика се заснова на пресметување на кинетичката енергија и потенцијалната енергија за секој крак од роботскиот манипулатор поодделно и потоа се пресметува Лагранжијанот L

$$L = K - P \quad (12)$$

каде што:

$K = \frac{1}{2} \dot{q}^T M(q) \dot{q}$ е вкупната кинетичка енергија во сите краци

P е вкупната потенцијална енергија во сите краци

Овде треба да се напомене дека Лагранжијан е различно од вкупна енергија на системот. Вкупната енергија на системот е збир на кинетичката и потенцијалната енергија во системот, додека Лагранжијан е разлика помеѓу кинетичката и потенцијалната енергија во системот.

Бараната динамичка равенка директно се добива од Лагранжијанот со примена на формула (13)

$$\tau = \frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}} \right) - \frac{\partial L}{\partial q} \quad (13)$$

Сега да го замениме Лагранжијанот од равенството (12) во равенството (13) и при тоа дополнително да ги искористиме и формулите за пресметување на енергиите. Тогаш ќе се добие равенството (14)

$$\begin{aligned}
\tau &= \frac{d}{dt} \left(\frac{\partial(K(q, \dot{q}) - P(q))}{\partial \dot{q}} \right) - \frac{\partial(K(q, \dot{q}) - P(q))}{\partial q} = \\
&= \frac{d}{dt} \left(\frac{\partial(K(q, \dot{q}))}{\partial \dot{q}} \right) - \frac{\partial(K(q, \dot{q}))}{\partial q} + \frac{\partial(P(q))}{\partial q} = \\
&= \frac{d}{dt} \left(\frac{\partial \left(\frac{1}{2} \dot{q}^T M(q) \dot{q} \right)}{\partial \dot{q}} \right) - \frac{\partial \left(\frac{1}{2} \dot{q}^T M(q) \dot{q} \right)}{\partial q} + \frac{\partial(P(q))}{\partial q} = \\
&= \frac{d}{dt} (M(q) \dot{q}) - \frac{1}{2} \dot{q}^T \frac{\partial M(q)}{\partial q} \dot{q} + \frac{\partial(P(q))}{\partial q} = \\
&= M(q) \ddot{q} + \dot{M}(q) \dot{q} - \frac{1}{2} \dot{q}^T \frac{\partial M(q)}{\partial q} \dot{q} + \frac{\partial(P(q))}{\partial q} \\
&= M(q) \ddot{q} + V(q, \dot{q}) + G(q)
\end{aligned} \tag{14}$$

т.е.

$$V(q, \dot{q}) = \dot{M}(q) \dot{q} - \frac{1}{2} \dot{q}^T \frac{\partial M(q)}{\partial q} \dot{q} \tag{15}$$

$$G(q) = \frac{\partial(P(q))}{\partial q} \tag{16}$$

Од равенството (14) се гледа дека при пресметувањето на кинетичката енергија се добива и матрицата на инерции на манипулаторот. Матрицата на гравитациони сили G е едноставно прв извод од потенцијалната енергија. Останува само да се пресмета матрицата на центрифугални и Кориолисови сили.

$$M(q) = \frac{1}{2} \sum_{i=1}^n (J_{v_i}^T m_i J_{v_i} + J_{w_i}^T I_{c_i} J_{w_i}) \tag{17}$$

$$b_{ijk} = \frac{1}{2} (m_{ijk} + m_{ikj} - m_{jki}) \tag{18}$$

$$V = \begin{bmatrix} b_{111} & b_{122} & \cdots & b_{1nn} \\ b_{211} & b_{222} & \cdots & b_{2nn} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n11} & b_{n22} & \cdots & b_{nnn} \end{bmatrix} \begin{bmatrix} \dot{q}_1^2 \\ \dot{q}_2^2 \\ \vdots \\ \dot{q}_n^2 \end{bmatrix} + \begin{bmatrix} 2b_{112} & 2b_{113} & \cdots & 2b_{1(n-1)n} \\ 2b_{212} & 2b_{213} & \cdots & 2b_{2(n-1)n} \\ \vdots & \vdots & \ddots & \vdots \\ 2b_{n12} & 2b_{n13} & \cdots & 2b_{n(n-1)n} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \dot{q}_2 \\ \dot{q}_1 \dot{q}_3 \\ \vdots \\ \dot{q}_{n-1} \dot{q}_n \end{bmatrix} \tag{19}$$

$$V = C(q)[\dot{q}^2] + B(q)[\dot{q}\dot{q}] \tag{20}$$

каде што C е матрица на центрифугални сили, а B е матрица на Кориолисови сили.

$$G = -(J_{v_1}^T \quad J_{v_2}^T \quad \cdots \quad J_{v_n}^T) \begin{pmatrix} m_1 g \\ m_2 g \\ \vdots \\ m_n g \end{pmatrix} \tag{21}$$

2.3.3 Управување на манипулаторот

Генерирање на траекторија претставува планирање на патеката на движење на извршниот елемент на манипулаторот во околина помеѓу други објекти. Да претпоставиме дека имаме еден манипулатор кој се наоѓа во некоја почетна позиција определена со T_A . Тој манипулатор треба да се придвижи до целна позиција определена со T_C . За да се направи ова движење поинтересно, да речеме дека ќе треба извршниот елемент на манипулаторот да помине низ неколку други меѓуточки, на пример T_B . Ова се прави затоа што доколку манипулаторот е лоциран во област во која има објекти и треба да се движи во таа област, потребно е да се насочи движењето на манипулаторот да ги избегнува објектите и во овој случај се вклучуваат меѓуточки за манипулаторот да се движи низ нив, а да не се судри со некој од објектите од околината. Ова е основниот проблем на генерирањето на траекторија.

Значи има точки од патеката:

- почетна позиција,
- целна позиција
- меѓуточки од патеката низ кои треба да помине манипулаторот.

Траекторијата во наједноставен случај е временска историја од позициите, брзините и забрзувањата за секој од степените на слобода на движење на манипулаторот. Овде ќе се планира движењето за сите степени на слобода на движење независно еден од друг и притоа ќе претпоставиме дека движењето целосно ќе може да се реализира. Бидејќи кога се тоа ќе се постави како една целина се станува многу сложено, за секој од степените на слобода на движење ќе се планира посебна траекторија.

Еве неколку примери на ограничувања што може да се сретнат:

- објекти од околината кои роботскиот манипулатор треба да ги избегнува,
- временски ограничувања - ако движењето мора да биде извршено во точно определено време (пр. роботски манипулатор треба да изврши определена операција на предмет што се движи на подвижна лента),
- рамномерност (манипулаторот треба да се движи рамномерно бидејќи на тој начин се троши помалку енергија и е полесно да се управува).

Од математичка гледна точка решавањето на тие ограничувања е многу едноставен проблем т.е. само потребно е планирање на траекторијата. Решението можеме да го бараме во неколку простори. Во пракса најчесто се користат следниве простори за генерирањето на траекторија:

- во простор на зглобовите (тоа е основниот простор на манипулаторот) и
- Декартовиот простор

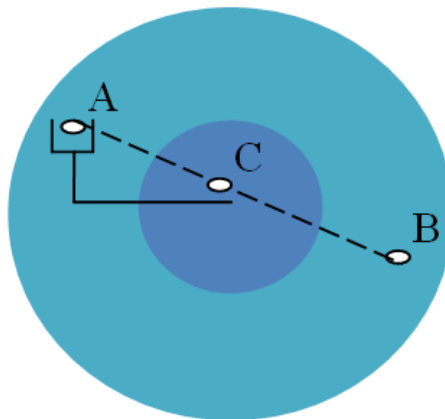
Во првиот случај едноставно е да се оди низ тие меѓуточки бидејќи точно се познати аглите на ротационите зглобови и издолжувањата на транслаторните зглобови на манипулаторот во тие меѓуточки. За да се добијат вредностите на аглите на ротационите зглобови и издолжувањата на транслаторните зглобови ќе треба да се реши задачата на инверзната кинематика во секоја од тие точки и потоа може да се планира движењето во просторот на зглобовите.

Да речеме дека се познати координатите на точките низ кои роботскиот манипулатор треба да помине и со решавање на задачата на инверзна кинематика во

тие точки се добиваат соодветните агли односно издолжувања на зглобовите и потоа се врши планирање на траекторијата во просторот на зглобовите. Тоа е прилично едноставно и има малку пресметки. Нема да има проблеми со сингуларитетите бидејќи сингуларитетите не постојат во просторот на зглобовите т.е. тоа е позицијата во која манипулаторот не може да се движи во определени насоки бидејќи физичката структура не му овозможува да го прави тоа. Во просторот на зглобовите тоа веднаш може да се забележи. Друга предност е тоа што се потребни малку пресметувања. Има потреба само да се врши пресметка на инверзната кинематика во тие точки. Проблем е што нема да може да следи права линија. Да речеме дека се пресметуваат аглите на зглобовите само во почетната и крајната положба на манипулаторот, а да заборавиме на меѓуточките. Кога се врши планирање патека во просторот на зглобовите, не се добива никаква информација за изгледот на таа патека во Декартовиот простор -дали тоа е права линија или има сосема друг облик. Исто така не може да следи точно определена траекторија. Доколку тоа не е проблем односно ако е во ред доколку линијата не е баш права линија туку само приближно се работи за права линија со некои апроксимации тогаш е во ред и нема да има многу пресметки. Но ако треба да следи точно определена траекторија, тогаш тоа да се направи во просторот на зглобовите може да биде многу сложено. Тоа е многу поедноставно да се прави во Декартовиот простор.

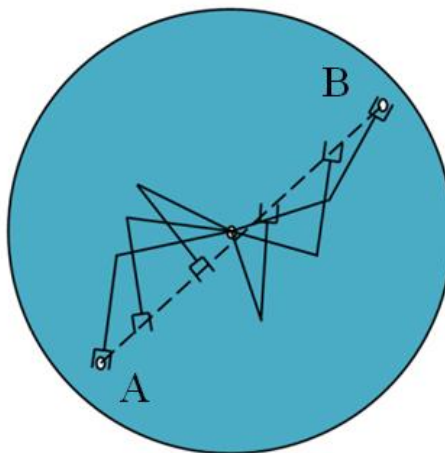
Во Декартовиот простор може да се следи било каков облик на траекторијата, бидејќи точките низ кои треба да помине извршниот елемент на манипулаторот се претставени токму во Декартовиот простор каде што манипулаторот се движи и ако му дадеме да се движи по права линија тој ќе се движи по права линија. Тоа се прави на тој начин што во основа планирањето се врши во Декартовиот простор користејќи ги Декартовите координати и потоа за да се најде ориентацијата на роботот може да се користи било кој од механизмите како што се еквивалентни оски, Ојлерови агли итн. Овој начин на следење на траекторија е многу поскап во поглед на пресметувачки ресурси. Во основа при управување на манипулатор чија траекторија е генерирана во Декартов простор, во реално време треба да се земаат примероци на точки од таа траекторија доволно често за да се насочи движењето да биде токму по таа траекторија. Ќе биде потребно пресметување на позициите на сите зглобови во секоја од тие точки и потоа тие вредности му се предаваат на управувачот кој го насочува движењето на секој од зглобовите да поминува токму низ тие точки. Што значи дека тоа е многу поскапо во поглед на пресметувачки ресурси за да се насочи движењето кон таа поединечна траекторија.

Другиот главен проблем е дисконтинуитетот. Ако планирањето се врши во Декартов простор и во тој простор траекторијата е убава права линија која роботскиот манипулатор ја следи во Декартов простор, тогаш кога некоја точка од таа права линија ќе треба да се претвори во просторот на зглобовите може да се добие дека тоа не е возможно да се направи во просторот на зглобовите што подолу ќе биде прикажано и во неколку примери. И двата простори си имаат свои предности и недостатоци. Во реалноста вообичаено се користи некој тип на хибриден пристап за да се ограничат пресметувањата, но исто така и да се осигураме дека нема манипулаторот да се судри со некој од објектите од неговата околина. Со цел да го направам воочувањето на проблемите при планирањето на патеката само со користење на Декартовиот простор поедноставно, ќе користам едноставен роботски манипулатор со само два краци.



Слика 4 Робот со два краци (прв проблем што може да произлезе доколку планирањето на траекторија се врши само во Декартов простор)

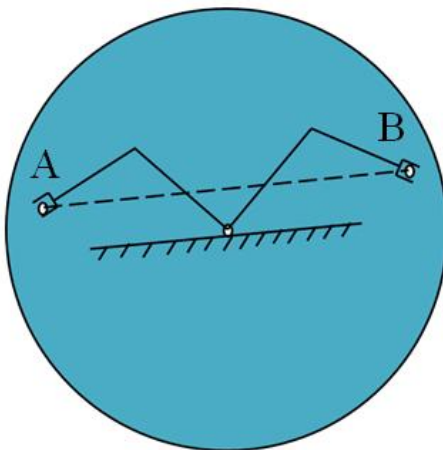
Даден е робот чиј краен извршен елемент се наоѓа во точката А. Потребно е роботот да се придвижи до крајна позиција определена со точката В. Притоа, движењето треба да се одвива по права линија. Робот е со два краци претставен на слика 4. И почетната позиција на роботот и крајната позиција се наоѓаат внатре во работниот простор на роботот. Ако се планира движење по права линија во картезијанскиот простор може да се види дека поминува низ точката С. Точката С се наоѓа надвор од работниот простор на роботот. Тоа немаше да биде познато за управувачот доколку не ги направеше пресметките по патеката, туку за овој проблем управувачот ќе дознае кога ќе најде на сингуларитетот. Тоа е првиот тип на проблем со кој може да се соочиме ако користиме само планирање во Декартов простор.



Слика 5 Робот со два краци (втор проблем што може да произлезе доколку планирањето на траекторија се врши само во Декартов простор)

На слика 5 е претставен робот со два краци и роботот треба да се придвижи од почетна позиција А до крајна позиција В по права линија. Во овој случај секоја точка по должината на траекторијата може да ја достигне манипулаторот. Сингуларитетот ќе биде точно во средината. Кога роботот се доближува до таа точка, брзината на зглоб 1 се движи кон бесконечност и очигледно е дека нема да биде во состојба да ја следи таа

права траекторија. И повторно тоа не би било познато за управувачот пред роботот да се најде во таа ситуација доколку планирањето на траекторијата се врши само во Декартовиот простор.



Слика 6 Робот со два краци (трет проблем што може да произлезе доколку планирањето на траекторија се врши само во Декартов простор)

На слика 6 е прикажана уште една ситуација во која и двете точки може да бидат достигнати од манипулаторот, сите точки од траекторијата исто така може да се достигнат и тоа без да се дојде до проблемот во кој брзината на некој од зглобовите се движи кон бесконечност. Меѓутоа, постојат две решенија на инверзната кинематика во целната точка во просторот на состојби. Па според тоа не е можно роботот да се движи од точката А до точката В по таа патека бидејќи точката А може да се дофати со левата конфигурација додека точката В е дофатлива со десната конфигурација и тие не се пресекуваат во средината.

Ова се типовите на проблеми кои може да се сретнат доколку планирањето на траекторијата на движење на роботот се врши само во картезијанскиот простор. Останува да се претстават формулите со помош на кои се врши планирањето. Ќе претпоставиме една општа променлива U . Според тоа, $u = (x, y, z)$ ако се работи за Декартов простор и $u = (\theta_1, \theta_2, \dots, \theta_n)$ ако претставувањето е во просторот на зглобови.

Потребно е роботот да се движи од една точка до друга во просторот. Наједноставниот начин е движењето да се одвива по права линија. Проблемот со правата линија е неконинуалната брзина во меѓуточките од патеката бидејќи правата линија во основа дава само два параметри и нема никаква информација за брзината или за забрзувањето.

Еве еден пример. Потребно е роботот да се придвижи од точка А до точка D преку точките В и С. Наједноставната траекторија е движењето да се одвива од А до В, потоа од В до С и на крај од С до D по прави линии. Подолу тоа ќе биде прикажано со формули во повеќе детали, но во основа, ако се планира траекторијата да биде права линија, не може да се гарантира дека брзината во точката В на почетокот на сегментот ВС и брзината во истата точка на крајот од сегментот АВ ќе бидат еднакви, што ќе предизвика неконинуирано движење. Ако движењето започне од почетната точка А,

па потоа манипулаторот ќе стигне и застане во меѓуточката В, па потоа повторно тргнува од брзина 0 од точката В и се движи кон С каде што повторно ќе застане итн. тоа би било во ред, но движењето би било неkontинуирано. Вообичаено тие меѓуточки се воведени за роботот да не се судри со некој објект од неговата околина или за роботот да изврши некои едноставни задачи во средината, но како и да е, не е неопходно да се застанува во нив, всушност и не треба да се застанува во нив бидејќи на тој начин непотребно би се трошело енергија и време. Па затоа траекторијата треба да се планира на тој начин што движењето од точка А до точката D ќе се одвива континуално, притоа избегнувајќи судар со објекти од непосредната околина на тој начин што движењето ќе биде што е можно поблиску до меѓуточките. Затоа движењето ќе се одвива по прави линии со закривување во меѓуточките. Бидејќи и во почетната положба ќе биде потребно определено забрзување и тука ќе се јави мало закривување. Слично на тоа, во крајната положба потребно е роботот да застане што значи дека и тука ќе се јави закривување. Значи на почетокот ќе има мало закривување, по што ќе се одржува рамномерна брзина за определено време и потоа во близина на меѓуточката В повторно ќе има едно закривување за да се одржи континуалноста на брзината, па потоа повторно права линија помеѓу двете меѓуточки па пак закривување во близина на втората меѓуточка, па пак права линија и на крајот забавување и полека застанување во крајната точка. Значи на овој начин планирањето на траекторијата е права линија со мали закривувања на краевите.

Друг начин на планирање на траекторија е наместо користење на права линија да се искористи кубна полиномна равенка. На овој начин, наместо права линија помеѓу секоја точка од патеката се планира крива од повисок ред помеѓу тие точки од патеката и тоа би било малку посложено од гледна точка на формулата. И на крај ако постојат многу ограничувања кои треба да се задоволат, може да се избере да се користи полином од повисок ред (пр. четврт ред, петти ред ...). Во полиномот од трет ред има четири параметри па според тоа може да се задоволат само 4 ограничувања. Ако треба да се управуваат и брзините и забрзувањата ќе биде потребен полином од повисок ред, бидејќи ќе бидат потребни повеќе коефициенти за да се задоволат повеќе ограничувања. И секако колку е посложен полиномот што се користи толку ќе биде посложен и моделот што се користи.

2.3.3.1 Планирање на траекторија на движење на манипулатор со полином од трет ред

Кубната равенка како функција од времето е од обликот (22)

$$u(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (22)$$

Во кубната равенка има 4 параметри, а тоа значи дека ако планирањето на траекторијата се врши со кубна равенка тогаш е возможно да се задоволат најмногу 4 ограничувања. Типично тоа што секогаш постои како почетни услови се некоја почетна позиција и некоја крајна позиција.

почетни услови: $u(0) = u_0$; $u(t_f) = u_f$

Ова се два услови. Може да се задоволат повеќе ограничувања од досега наброените, како на пример брзината. Со диференцирање на кубната равенка по времето t се добива равенката на брзината.

$$\dot{u}(t) = a_1 + 2a_2 t + 3a_3 t^2 \quad (23)$$

Овде се вклучуваат уште два почетни услови. Во почетната точка манипулаторот започнува да се движи со брзина еднаква на нула, и кога ќе стигне во крајната точка манипулаторот треба да застане. Тоа значи дека брзината во почетната точка, како и брзината во крајната точка треба да бидат еднакви на нула. Најверојатно брзината во меѓуточките нема да биде еднаква на нула, но ова е наједноставниот случај.

$$\dot{u}(0) = 0; \dot{u}(t_f) = 0 \quad (24)$$

На овој начин веднаш се добива дека вредноста на параметарот a_1 е нула и се добива уште едно равенство за a_2 и a_3 . Сега има систем од три линеарни равенства за пресметување на параметрите a_0 , a_2 и a_3 . Овде забрзувањето не може да се управува. Тоа се добива директно од формулата. Па затоа погоре беше напоменато дека ако е потребно да се управува и забрзувањето ќе биде потребно да се искористи полином од повисок ред за да има повеќе параметри со кои може да се работи.

Доколку се пресметаат сите параметри и се заменат во кубната равенка (22) се добива равенката (25)

$$u(t) = u_0 + \frac{3}{t_f^2}(u_f - u_0)t^2 + \left(\frac{2}{t_f^3}\right)(u_f - u_0)t^3 \quad (25)$$

Како што може да се види траекторијата зависи од почетната позиција, крајната позиција и времето за кое треба да се постигне посакуваната позиција.

Претходната формула важи само во случај кога движењето е од почетна позиција до крајна позиција без меѓуточки. Доколку се планира траекторијата на движење на роботот помеѓу две меѓуточки, тогаш тоа што треба да се направи е во меѓуточките во средината да не се застанува или со други зборови нема потреба брзината во тие точки да биде еднаква на нула. Значи за континуални движења без запирања ќе бидат потребни и брзините во меѓуточките кои во овој случај ќе бидат различни од нула. Па во време 0 брзината ќе има некоја вредност $\dot{u}(0) = \dot{u}_0$ и во времето t_f ќе има некоја друга вредност $\dot{u}(t_f) = \dot{u}_f$. За сега ќе претпоставиме дека се дадени, а подоцна ќе биде опишано како се добиваат. Тие ќе бидат додадени кон почетните услови. Значи во овој случај како почетни услови ќе бидат почетната позиција, крајната позиција, брзината во почетната позиција и брзината во крајната позиција. Повторно постојат четири почетни услови и може да се пресметаат параметрите од кубната равенка користејќи ги тие четири почетни услови. Решението во овој случај ќе биде дадено со равенките (26):

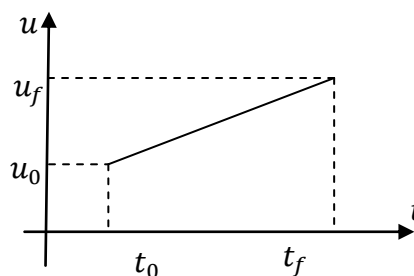
$$\begin{aligned} a_0 &= u_0 \\ a_1 &= \dot{u}_0 \\ a_2 &= \frac{3}{t_f^2}(u_f - u_0) - \frac{2}{t_f}\dot{u}_0 - \frac{1}{t_f}\dot{u}_f \\ a_3 &= -\frac{2}{t_f^3}(u_f - u_0) + \frac{1}{t_f^2}(\dot{u}_f + \dot{u}_0) \end{aligned} \quad (26)$$

Параметрите добиени со равенките (26) ја опишуваат траекторијата во Декартовиот простор, но за да се управува роботот потребно е управувачот на роботот да ги добие податоците во просторот на зглобовите. Позицијата во просторот на зглобови за секоја точка од траекторијата се добива со користење на инверзната кинематика во таа точка додека брзината може да се добие со користење на инверзната матрица на Јакобијанот.

$$\dot{u} = J^{-1} \begin{bmatrix} v \\ w \end{bmatrix} \quad (27)$$

Секој индустриски робот е направен така да се движи до одредена максимална брзина што произлегува од карактеристиките на механичката структура. Во таков случај, таа брзина може да се вклучи во условите за пресметување на патеката како посакувана брзина. Исто така наместо роботот да се движи секогаш со неговата максимална брзина би можело да се користи систем од еуристики за добивање на посакуваната брзина која што секако би била помала од максималната можна брзина на движење на роботот. Еве на пример како еуристика може да се вклучи дисипацијата на енергија. Ако роботот се движи на тој начин што прво забрзува до максималната брзина што може да ја постигне, за што се троши одредено количество на енергија, па потоа го забавува движењето за што повторно се троши енергија, тоа ќе влијае на поголема дисипација на енергија. Дисипацијата на енергија треба да биде што е можно помала. Како еуристика исто така може да се користи грешката, односно колку роботот отстапува од посакуваната позиција. Познато е дека брзината влијае на појавата на вибрации и отклонувања. Доколку брзината на движење е поголема ефектите на вибрации и отклонувања се поизразени. Секако дека и брзината не смее премногу да се намали бидејќи од тоа ќе зависи ефикасноста на роботот. Постојат и други еуристики што може дополнително да се вклучат во добивањето на посакуваната брзина на движење на роботскиот манипулатор. Од сите еуристики кои се сметаат за важни се создава систем од еуристики со помош на кој се добива посакуваната брзина. И на крај може да се додадат и некои дополнителни ограничувања, како на пример брзината во меѓуточките да биде континуална, односно токму во меѓуточките да биде еднаква во двата сегменти од траекторијата. Истото важи и за забрзувањето, како дополнително ограничување, и да се додаде тоа кон системот. Очигледно е дека ако се направи тоа мора да се исклучат некои од претходните ограничувања бидејќи би имало повеќе ограничувања од параметри. Најмногу може да се задоволат онолку ограничувања колку што има параметри. Во општ случај нема да бидат дадени брзините, а вообичаено би требало роботот да се движи од една точка кон друга итн. И времето исто така нема да биде дадено и кое исто така може да се искористи како еуристика на пример колку брзо треба да се изведе движењето или колку најбрзо може да се изведе движењето без притоа да се надмине максималната можна брзината за некој поединечен зглоб.

На Слика 7 е прикажан график на линеарна интерполација. Тоа аналитички може да се запише со равенката (28)



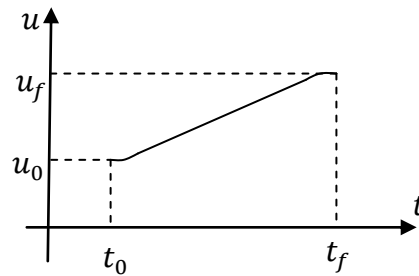
Слика 7 Линеарна интерполација

$$u(t) = a_0 + a_1 t \quad (28)$$

Постојат два услови, а тоа се почетната позиција и крајната позиција.

$$\begin{aligned} u(t_0) &= u_0 \\ u(t_f) &= u_f \end{aligned} \quad (29)$$

Ова значи неконтинуална брзина што не може физички да се задоволи, затоа ќе воведеме параболни закривување во почетната и крајната точка. Тоа е прикажано на Слика 8.



Слика 8 Линеарна интерполација со закривување на краевите

Од Слика 8 може да се забележи дека движењето започнува во моментот $t = t_0$, кога роботот се наоѓа во почетната позиција u_0 и се движи до крајна позиција u_f што ја достигнува во моментот $t = t_f$. Поради тоа што неконтинуирана брзина е физички невозможно да се оствари воведени се параболни закривувања во почетниот дел и крајниот дел од сегментот. Првото закривување започнува во моментот t_0 и завршува во моментот $t_0 + t_b$, по што следи линеарниот сегмент. Линеарниот сегмент завршува во моментот $t_f - t_b$, кога започнува второто закривување од сегментот. Второто закривување трае до завршувањето на сегментот т.е. до моментот t_f . За да се направат пресметките поедноставни ќе претпоставиме дека двете закривувања имаат еднакво времетраење. Во овој случај равенството на закривувањето кое всушност е параболно равенство, математички може да се запише со равенката (30).

$$u(t) = \frac{1}{2}at^2 \quad (30)$$

Овде има само еден параметар, а тоа е a . И за тој параметар потребно е да се вклучи некој услов, па тоа ќе додаде уште едно ограничување што може да се задоволи, што во овој случај е брзината. Брзината мора да биде континуална во текот на движењето. Па брзината тука е едноставно

$$\dot{u}(t) = at \quad (31)$$

и ако на пример се додаде условот за константно забрзување

$$\ddot{u}(t) = a \quad (32)$$

тоа ќе ја даде таа вредност a , или во тој случај во основа би се добило

$$u(t) = \frac{1}{2}\ddot{u}t^2 \quad (33)$$

Понатаму ќе биде покажано како се определува тоа забрзување за континуално движење.

Поради тоа што брзината треба да биде континуална во основа може да се пресмета времето на закривувањето од условот за континуална брзина. Од вредностите u_f , u_0 и \ddot{u} може да се пресмета времетраењето на закривувањето за тој поодделен регион што постигнува континуална брзина.

Сега да претпоставиме робот што треба да се движи поминувајќи низ повеќе однапред определени, точки по прави линии. Притоа познат е и временскиот момент во кој роботот треба да се најде во секоја од тие точки. Дополнително дадени се и магнитудите на забрзувањата. Потребно е да се генерира траекторија на движење на тој робот.

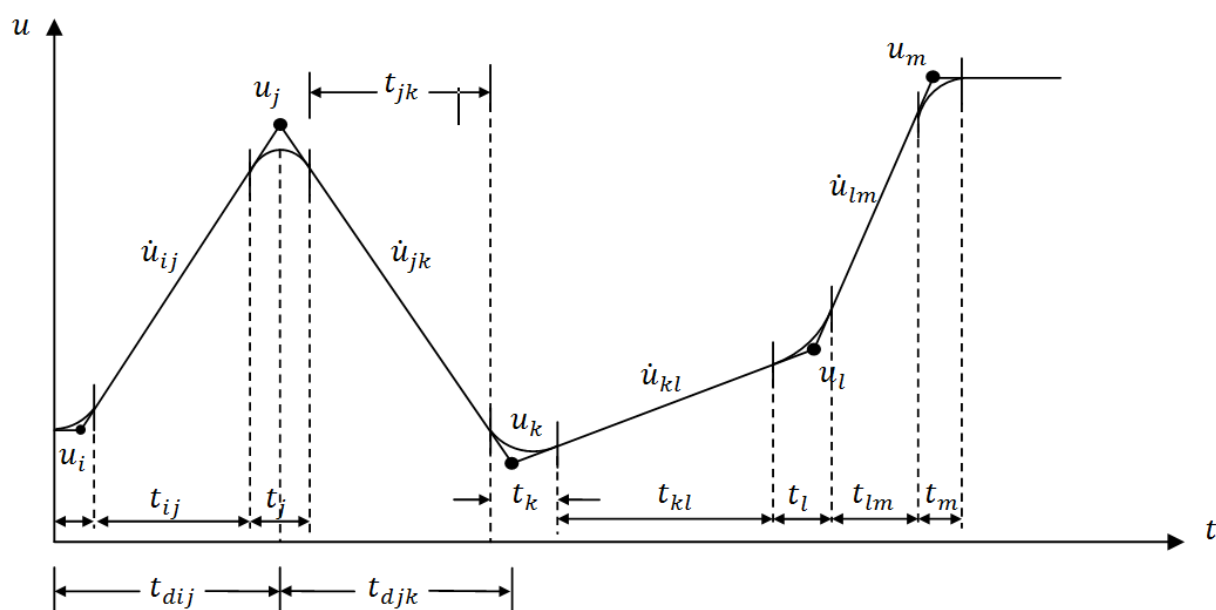
Значи овде познати се:

- * позициите u_i, u_j, u_k, u_l, u_m
- * посакуваното времетраење $t_{dij}, t_{djk}, t_{dkl}, t_{dlm}$,
- * магнитудите на забрзувањата $|\ddot{u}_i|, |\ddot{u}_j|, |\ddot{u}_k|, |\ddot{u}_l|$

Потребно е да се пресметаат

- * времињата на закривување t_i, t_j, t_k, t_l, t_m
- * времињата на правите сегменти $t_{ij}, t_{jk}, t_{kl}, t_{lm}$
- * наклоните (брзините) $\dot{u}_{ij}, \dot{u}_{jk}, \dot{u}_{kl}, \dot{u}_{lm}$
- * забрзувањата (вклучувајќи ги и знаците бидејќи веќе ги имаме магнитудите)

Тоа е прикажано на Слика 9.



Слика 9 Траекторија со три меѓуточки

Системот вообичаено ги пресметува или користи подразбирани вредности за забрзувањата што зависат од конфигурацијата на роботот, неговата механичката конструкција (пр. колку брзо би сакале да се движи, работниот простор) итн. Системот исто така може да ги пресмета посакуваните времетраења врз основа на подразбираните вредности на брзините. Очигледно е дека ќе бидат различни за различни сегменти.

Со користење на формулата (34)

$$\ddot{u}(t) = \text{sign}(u_j - u_i)|\ddot{u}_i| \quad (34)$$

може да се пресметаат забрзувањата вклучувајќи го и знакот врз основа на тоа дали се работи за забрзување или успорување. Кога тоа е веќе пресметано и бидејќи е познато времетраењето на целиот сегмент, може да се пресмета времетраењето на првото закривување t_i со помош на формулата (35).

$$t_i = t_{dij} - \sqrt{t_{dij}^2 - \frac{2(u_j - u_i)}{\ddot{u}_i}} \quad (35)$$

Потоа може да се пресмета и брзината во тој сегмент со формулата (36).

$$\dot{u}_{ij} = \frac{u_j - u_i}{t_{dij} - \frac{1}{2}t_i} \quad (36)$$

И на крајот може да се пресмета и времетраењето на линеарниот дел од тој сегмент користејќи ја формулата (37)

$$t_{ij} = t_{dij} - t_i - \frac{1}{2}t_j \quad (37)$$

На тој начин е целосно определено движењето во првиот сегмент и продолжуваме кон вториот сегмент. Вториот сегмент е помеѓу две меѓуточки. Повторно може да се најдат брзините со формулата (38)

$$\dot{u}_{kj} = \frac{u_k - u_j}{t_{djk}} \quad (38)$$

Потоа забрзувањето заедно со предзнакот користејќи ја формулата (39)

$$\ddot{u}_k = \text{sign}(\dot{u}_{kl} - \dot{u}_{jk})|\ddot{u}_k| \quad (39)$$

Исто како и претходно и овде може да се најде времетраењето на закривувањето со користење на претходно пресметаните брзини и забрзувања

$$t_k = \frac{\dot{u}_{kl} - \dot{u}_{jk}}{\ddot{u}_k} \quad (40)$$

и на крајот може да се најде времетраењето на линеарниот дел од сегментот со едноставно одземање на целото време на траење на целиот сегмент и времетраењето на закривувањата

$$t_{jk} = t_{djk} - \frac{1}{2}t_j - \frac{1}{2}t_k \quad (41)$$

Кога ќе стигнеме до последниот сегмент, слично како и во претходниот случај се користат формулите (42), (43), (44) и (45).

$$\ddot{u}_n = \text{sign}(\dot{u}_{n-1} - \dot{u}_n)|\ddot{u}_n| \quad (42)$$

$$t_n = t_{d(n-1)n} - \sqrt{t_{d(n-1)n}^2 - \frac{2(u_n - u_{n-1})}{\ddot{u}_n}} \quad (43)$$

$$\dot{u}_{(n-1)n} = \frac{u_n - u_{n-1}}{t_{d(n-1)n} - \frac{1}{2}t_n} \quad (44)$$

$$t_{(n-1)n} = t_{d(n-1)n} - t_n - \frac{1}{2}t_{n-1} \quad (45)$$

Тоа што може да се забележи тука е дека траекторијата не поминува точно низ било која од меѓуточките, туку многу близу до нив. Да се земе во предвид дека причината за која се воведуваат овие меѓуточки, т.е. основната причина за која се воведуваат овие меѓуточки е доколку се врши планирање на траекторија на движење на робот во околина во која има други објекти во кои притоа роботот не смее да удри, тогаш се вклучуваат тие меѓуточки за да се осигураме дека роботот ќе се движи околу објектите и навистина не е толку важно траекторијата на роботот да поминува точно низ тие точки. Но, доколку ситуацијата е таква што роботот навистина мора да поминува точно низ тие точки тогаш може да се направи роботот точно да поминува низ тие точки, но тоа во општ случај и не мора да биде така. Ако навистина роботот мора да поминува токму во тие точки, тоа може да се направи со користење на некој од следниве механизми.

1. Може да се користат псеудомеѓуточки. На двете страни од меѓуточката може да се додадат по една псеудомеѓуточка и да се изврши планирањето. Тие додадени псеудомеѓуточки треба да се додадени така што при планирањето траекторијата да поминува точно низ посакуваната точка или барам да биде доволно близу до неа.

2. Може да се користи доволно големо забрзување со што ќе се принуди роботот да помине низ посакуваната точка.

3. Исто така планирањето на траекторијата може да се направи и на тој начин што во секоја од меѓуточките роботот ќе застане.

Во основа тие меѓуточки се тука со една единствена намена, а тоа е да се избегнува колизија со други објекти што значи дека траекторијата не мора да поминува точно низ тие меѓуточки, туку доволно е да поминува блиску до меѓуточките.

Значи тоа беа двата начини кои може да се користат при планирање на траекторија, кубна полиномна равенка или права линија со параболични закривувања. Како што претходно беше наведено ако е потребно да се задоволат повеќе услови тогаш може да се користи полином од повисок ред. На пример ако однапред се зададени две позиции, две брзини и две забрзувања за некој поединечен сегмент тогаш се поставени шест ограничувања. Па затоа наједноставниот можен модел што би ги задоволил сите 6 ограничувања би била полиномна равенка од 5-ти степен, бидејќи во таа равенка има 6 параметри што може да се нагудуваат.

$$u(t) = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5 \quad (46)$$

Од 6-те услови произлегуваат 6 равенки со чие што пресметување се добиваат 6-те параметри.

Секако дека може да се користат и друг тип на функции, но најчесто се користат полиномни функции затоа што тие резултираат во линеарни функции што е многу

полесно да се решаваат. Досега беа претставени повеќе механизми на планирање на траекторија ако се дадени условите за траекторијата и тоа е многу теоретско т.е. само математички формули за тоа што може да се направи. А сега ќе биде претставено што всушност се прави кога се врши всушност планирањето на траекторија за роботот т.е. практичното генерирање на траекторија. Во основа треба нешто да му се предаде на управувачот т.е. да му се соопштат различните позиции, брзини и/или забрзувања за различните зглобови на роботот. Овде со θ се означени променливите на зглобовите на манипулаторот. Генераторот на патеката ја пресметува патеката во некое реално време. Нему му се дадени почетната точка, крајната точка и меѓуточките. Останатите вредности што ја опишуваат траекторијата може да се добијат на начин што беше претходно опишан. Ако планирањето се врши во просторот на зглобовите директно и ако на пример се користи кубна полиномна равенка множеството на коефициенти на траекторијата ќе се разликува за различни сегменти. На крајот на секој од сегментите може да се пресметуваат тие коефициенти и да му се соопштуваат на управувачкиот систем. Значи движењето започнува со едно множество на коефициенти кои откако ќе се пресметаат му се даваат на управувачот и роботот се движи и се доближува до следната точка. Кога роботот ќе се доближи на следната точка од траекторијата тогаш се прават нови пресметки при кои се добиваат нови коефициенти кои повторно му ги соопштуваат на управувачот и тој во точно определено време започнува да ги користи и на тој начин се добива таа кубна равенка. Ако се користи линеарна равенка со параболни закривувања, на крајот на секој сегмент треба да се провери дали моментално движењето е во линеарниот дел или во делот на параболичното закривување бидејќи се работи за различни формули во зависност од тоа дали се работи за линеарниот дел од сегментот или за закривувањето и во зависност од тоа се прават пресметките и на управувачот му се соопштуваат соодветните коефициенти. Овие пресметки всушност се одвиваат на определено време, значи постои одредена фреквенција на ажурирање на коефициентите. Значи при кубна полиномна равенка точките на ажурирање се меѓуточките и крајната точка, додека при линеарна равенка со параболни закривувања мора да се види во кој дел од формулата моментално се наоѓа роботот, што не е проблем бидејќи се познати формулите и тоа едноставно се пресметува. Секако проблемот е што не ја следи посакуваната траекторија туку некако се движиме континуално во просторот.

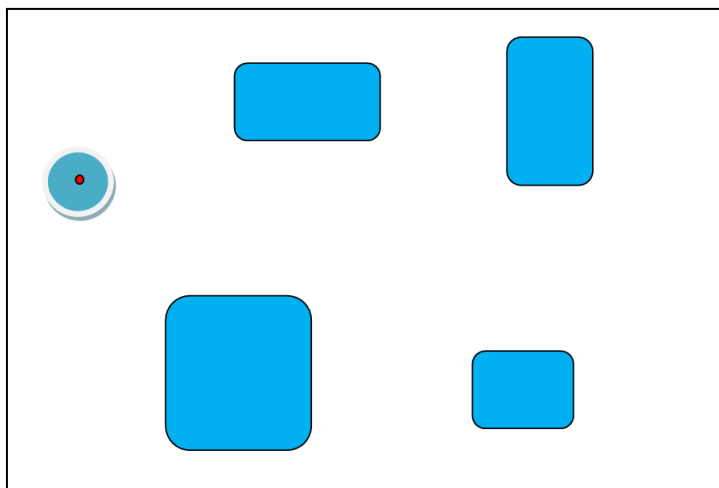
Ако планирањето се прави во Декартов простор тогаш се пресметува Декартовата позиција и ориентација во секоја точка со користење на истите формули и тогаш тие се претвораат во просторот на зглобовите со користење на инверзна кинематика или директната кинематика и инверзната матрица на Јакобијанот. Дополнително на тоа треба да се напомене дека движењето мора да биде континуално. Работите стануваат многу комплицирано кога треба да се изгради целосен систем, но сепак ова ја претставува основата.

За планирање на траекторија со пречки има неколку работи што треба да се имаат во предвид.

* локално наместо глобално планирање на траекторија - овде треба да се внимава дали планирањето се врши за целиот манипулатор т.е. дали се работи за глобално планирање на движењето во просторот или само на локалното движење како што е само каде се наоѓа крајниот извршен елемент. Вообичаено се прави некој тип на комбинирање помеѓу глобалното или локалното планирање. Глобалното планирање се користи при движењето во релативно празен простор, кога не е возможно да се случи судар на роботот со некој објект, додека кога роботот ќе се најде во просторот во кој

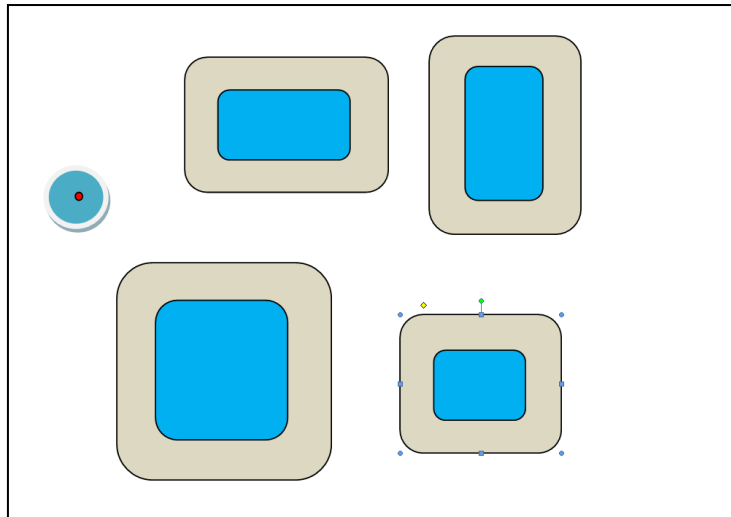
што има повеќе објекти тогаш тоа се променува во планирање на локалното движење. Тоа е многу попрецизно планирање само за крајниот извршен елемент.

* планирање со пристап кон конфигурацијата - Тоа ќе биде прикажано преку пример. На Слика 10 е прикажана околина со објекти и еден кружен робот што треба да се движи наоколу во таа околина.



Слика 10 Кружен робот во околина со објекти

Овој кружен робот е со определен радиус. Доколку сакаме да генерираме траекторија на движење на тој робот во дадениот простор во кој има други објекти и при тоа сакаме роботот при своето движење да не се судри со ниеден објект од таа околина тоа што треба да го направиме е да планираме траекторија на секоја точка од роботот така што при движењето било која точка од роботот да не допре ниедна точка од било кој од објектите. Ова претставува многу сложена математика и потребни се премногу пресметувања. Потребно е да се направи поедноставување на проблемот. Тоа поедноставување на проблемот може да се направи со примена на генерирање на траекторија во конфигурацискиот простор (К простор). Конфигурацискиот т.е. К просторот се добива на тој начин што сите објекти од околината се зголемуваат за големината на радиусот на тој робот. Тоа е прикажано на Слика 11. Во овој случај ако се погледне црвената точка и се планира патека за црвената точка така што да не се судира со тие зголемени објекти тогаш со сигурност се знае дека кругот нема да се судира со малите објекти. Па тогаш планирањето на патеката на роботот е сведено на планирање на патека на само една точка во тој простор што геометриски може да се направи на многу различни начини. Овие зголемени објекти се нарекуваат К простор - објекти од конфигурацискиот простор и ова е планирачки пристап на К просторот.



Слика 11 Планирање во конфигурациски простор

И тоа што може да се направи е да се постави мрежа во просторот и потоа се планира патеката на точката во таа мрежа така што да не се судри со објектите. Потоа кога повторно ќе се вртиме кон кружниот робот и тој нема да се судира со објектите. Ако моделот беше посложен т.е. имаше неколку степени на слобода на движење и доколку дополнително се додаде и ориентација тогаш овој конфигурациски простор со објекти нема да биде само планирачки како што е во претходниот случај туку овде ќе се работи со тродимензионални објекти и при тоа ќе треба да се мисли и на ориентацијата. Во таков случај може да се зборува за сложена математика.

Може да се додаде и графичка репрезентација на слободниот простор и да се изгради разгрането дрво за да се знае во кој дел од слободниот простор роботот се наоѓа и потоа да се планира патека и тука може да се користи некој метод како што е методот на вештачко потенцијално поле. Доколку роботот се наоѓа близу до некој објект може да се претпостави сила која дејствува на роботот за тој да се оддалечува од објектот и бидејќи е позната целта ќе има и сила која дејствува да се движи роботот и во насока на целта и врз основа на тоа може да изгради вештачко потенцијално поле со чие што следење роботот се движи кон крајната позиција.

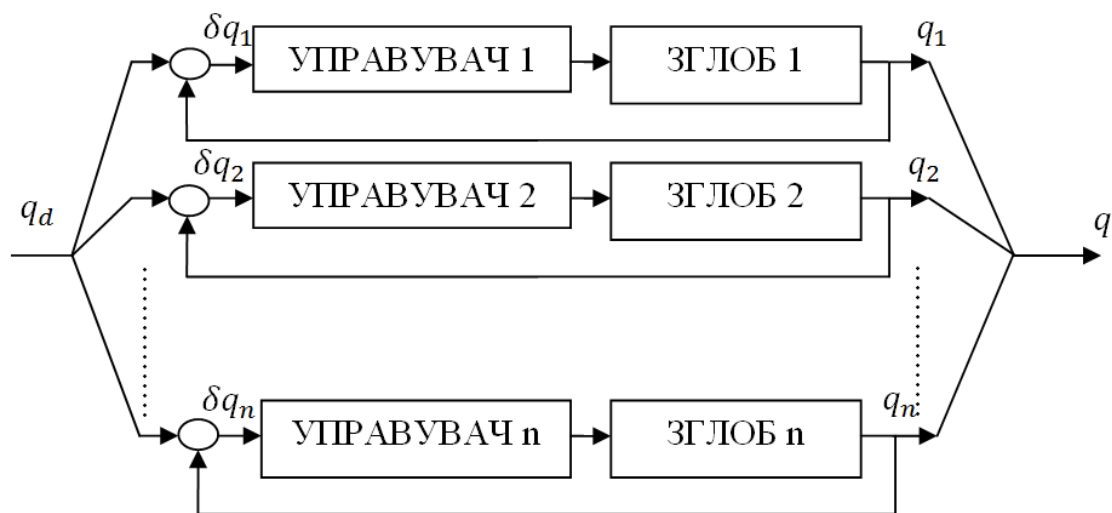
Дополнително на тоа може да се планира и траекторија на повеќе работи што се движат во иста околина. Тоа дополнително ќе го направи проблемот уште посложен.

2.3.3.2 Управување на роботски манипулатор

Во овој поднаслов се зборува за основите на управувањето. Овде роботот ќе биде управуван така да следи траекторија што е дадена во просторот на зглобовите или во картезијанскиот простор. Најпрво ќе биде објаснет наједноставниот начин на кој може да се управува роботскиот манипулатор, а тоа е управување на манипулаторот во просторот на зглобови. Потоа ќе биде објаснето што дополнително треба да се додаде на тој управувач за управувањето да биде во картезијанскиот простор. Притоа, ќе бидат прикажани два начини за управување на манипулаторот во картезијанскиот простор. Дополнително овде ќе биде објаснето како за управување на роботот може да се искористи ПИД (П-пропорционален, И-интегрирачки и Д-диференцијален) управувач. Потоа ќе биде објаснета и друга управувачка техника со која што може да се управува роботски манипулатор директно во однос на задачата на начинот на кој тоа го прави

човекот. Тоа е преку директно применување на сила, а не преку инверзната кинематика, туку директно управување на брзината и забрзувањето на крајниот извршен елемент. На крајот ќе се види дека управувањето на движењето не е единствената работа што треба да се направи кога се управува робот, туку исто така треба да се влијае во околината. Со цел да се влијае во околината ќе треба да се управуваат допирните сили, на пример ако има лизгање по некоја површина потребно е движење по таа површина, но и истовремено применување на сила на притискање и тоа ќе биде критична техника со цел да се дејствува во таа околина. Значи влијанието во околината може да опфаќа составување, движење на објекти и соработување со други роботи. Управувањето на роботот треба да се одвива на тој начин што сервомоторите од роботот ќе треба да произведат соодветен торзионен момент. Колкав ќе биде генерираниот торзионен момент ќе зависи од управувачот.

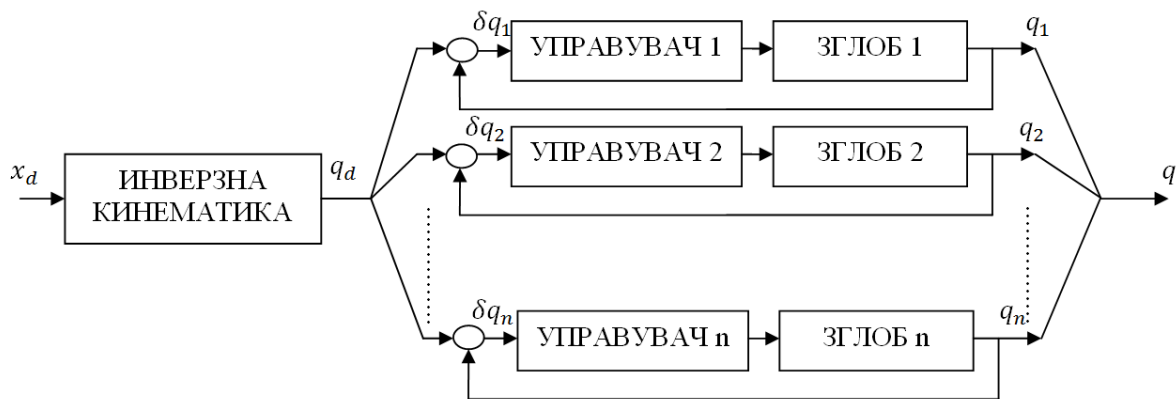
На слика 12 е прикажан модел на управување со роботски манипулатор ако траекторијата што роботот треба да ја следи е дадена во просторот на зглобови. Тоа е наједноставниот модел на управување со манипулатор. Во овој модел влезот е вектор од посакуваните позиции на зглобовите. За секој од зглобовите на манипулаторот е поставен посебен сервоуправувач чија што задача е да го управува соодветниот зглоб на тој начин што ќе ја следи посакуваната позиција добиена на влез во системот. Тој управувач може да биде од типот на ПИД управувач за кој подолу ќе зборувам.



Слика 12 Управување во просторот на зглобови

Управувањето на робот во картезијанскиот простор е малку посложено. За управувањето на роботот да биде во картезијанскиот простор на блок шемата од слика 12 треба дополнително да се додаде уште еден или повеќе блокови за да може од влезот во моделот кој е во картезијански координати да се добијат посакуваните позиции на зглобови. Тоа може да се направи со користење на блок што ја врши инверзната кинематика. На слика 13 е претставена токму таква блок шема на управувач. Овде задачата е да се управува роботот во картезијански простор. Посакуваната позиција во картезијанскиот простор на која треба да се позиционира роботот е зададена со влезот во системот x_d . Односно дадени се координатите x , y , и z ,

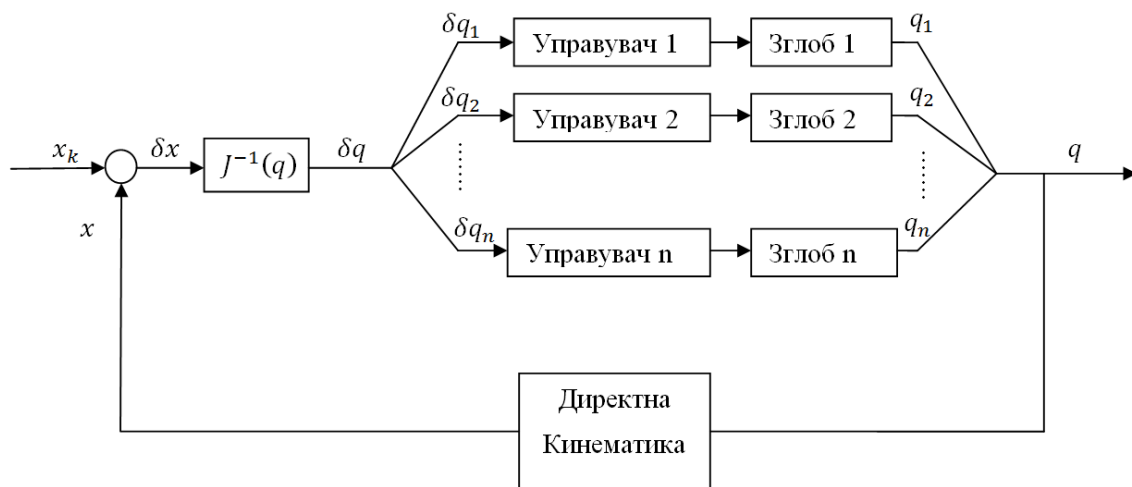
како и ориентацијата α , β и γ . Со помош на инверзната кинематика може да се определат посакуваните позиции на секој од зглобовите. Моменталната позиција на зглобовите се добива со читање на енкодерите поставени во зглобовите на роботот. Разликата на посакуваната позиција на зглобот и моменталната позиција на зглобот ја дава грешката која е поставена на влез на управувачот кој што е проектиран на начин на кој зглобот ќе го придвижува во насока на намалување на грешката. За секој од зглобовите поставен е посебен серво управувач. Овде проблемот е тоа што во блок шемата се јавува инверзната кинематика.



Слика 13 Управување во Декартов простор со користење на инверзната кинематика

За да се реши проблемот со инверзната кинематика добиен е нов начин на управување на манипулаторот. Тој начин на управување се појавил во 1972 и претставува управување како резултат на брзината на движење (result motion rate control). Идејата е наместо да се користи инверзната кинематика овде се користи директната кинематика која што ја заменува инверзната. Најпрво треба да се најде изместување δq што одговара на посакуваното изместување δx . Јакобијанот прецизно ја дава врската помеѓу δx и δq . Ако се користи инверзниот Јакобијан треба да сме сигурни дека позицијата не е сингуларитет или ако е сингуларитет, би требало да се користи специјален третман на конфигурацијата. Ако не се работи за сингуларитет и ако роботот е со 6 степени на слобода на движење може да се најде инверзна матрица на Јакобијанот, но ако роботот има повеќе или помалку од 6 степени на слобода на движење мора решавањето да се изврши со примена на општа инверзна матрица или псеудо инверзна матрица. Значи во секој случај може да се пресмета δq . За мало δx постои мало δq . Ако е позната моменталната позиција на секој од зглобовите следната конфигурација во која ќе се најде зглобот ќе биде $q + \delta q$. Значи се започнува од тековната позиција во која што се наоѓа роботот, потоа се пресметува грешката што всушност е разлика помеѓу посакуваната позиција на извршниот елемент и моменталната позиција на извршниот елемент и потоа се пресметува изместувањето δq . И се продолжува да се управува роботот до следната позиција на q^+ што всушност е каде тековно се наоѓа роботот плус малото изместување. Внатре во моделот сега го има инверзниот Јакобијан и наместо инверзната кинематика овде се среќава директната кинематика која што е многу поедноставно да се пресмета. Значи се користи директна кинематика и инверзниот Јакобијан кој за мали работи може да се добие и во аналитичка форма. Во основа се пресметува δq и им се соопштува неговата

вредност на секој од зглобовите и повторно постои засебен сервоуправувач за секој од зглобовите.



Слика 14 Управување во картезијански простор со користење на директната кинематика и инверзната матрица на Јакобијанот

Претходно беа претставени неколку блок шеми со кои се управуваат манипулаторите. При тоа во секоја од овие блок шеми може да се забележи дека постојат блоковите Управувач 1, Управува2, ... Управувач n. Потребно е да се разјасни што всушност се случува во овие блокови. На влез во секој од овие блокови е посакуваното поместување на соодветниот зглоб. Целта на блокот е врз основа на тоа поместување да се произведе управувачки сигнал кој ќе влијае на движењето на зглобот на тој начин што зглобот ќе се помести токму за тоа поместување. Овој блок е најчесто од типот на ПИД управувач. Тој може да се опише со преносната функција (47).

$$G = \frac{K_i + K_p s + K_d s^2}{s} \quad (47)$$

Од преносната функција (47) може да се забележи дека постојат три параметри K_p , K_i и K_d кои треба да се пресметаат. Пресметувањето на првиот параметар K_p се одвива со користење на потенцијалната енергија на системот. За нас од интерес е системот да има точно една рамнотежна состојба. Претходно беше покажано дека динамичката равенка на системот е од обликот (48).

$$\tau = M(q)\ddot{q} + V(q, \dot{q}) + G(q) \quad (48)$$

Рамнотежната состојба настанува во моментот кога брзината и забрзувањето се еднакви на нула од што произлегува равенството (49).

$$\tau = G(q) \quad (49)$$

Со користење на равенство се определува која е минималната вредност на K_p за системот да има точно едно решение односно да постои точно една рамнотежна состојба. Од изборот на коефициентот K_d ќе зависи преодниот режим додека параметарот K_i е воведен со цел намалување на стационарната грешка во позиционирањето.

Претходно опишав како може да се врши управувањето со роботскиот манипулатор и при тоа во првиот случај роботот следи траекторија зададена во просторот на зглобови, во вториот случај управувањето е во Декартовиот простор и при тоа се користи инверзната кинематика додека во третиот директната кинематика и инверзната матрица на Јакобијанот. При движењето на човековата рака, човекот не прави пресметка ниту на инверзната кинематика ниту пак на директната кинематика и Јакобијанот. Тој едноставно ја движи раката кон целта без вршење на било која сложена пресметка. Истиот принцип на управување што го користи човекот може да се примени и во роботиката. Да ја погледнеме динамичката равенка (50) на манипулаторот.

$$M(q)\ddot{q} + V(q, \dot{q}) + G(q) = \tau \quad (50)$$

Кога во системот подолго време не дејствуваат надворешни сили тогаш системот ќе биде во мирување што значи брзината и забрзувањето ќе бидат еднакви на нула т.е. динамичката равенка на системот во тој случај ќе биде равенката (51).

$$G(q) = 0 \quad (51)$$

Значи позицијата во која во овој случај ќе се најде роботот ќе зависи само од обликот на функцијата $G(q)$. Доколку во системот дејствуваат константни надворешни сили, позицијата во која ќе се најде роботот ќе зависи и од големината на тие сили.

$$G(q) = \tau \quad (52)$$

Овде може да избереме надворешните сили да бидат функција од обликот (53).

$$\tau = \tilde{G}(q) - G_n(q) \quad (53)$$

каде што $\tilde{G}(q)$ претставува апроксимација на функцијата $G(q)$. Претходно беше опишано аналитичкото добивање на таа функцијата и токму таа функција може да се искористи за функцијата $\tilde{G}(q)$. Функцијата $G_n(q)$ треба да се избере така што да важи (54).

$$G_n(q_p) = 0 \quad (54)$$

каде што q_p е посакуваната позиција во која сакаме да се најде роботот. Овде многу едноставно се управува роботот, но управувањето на роботот е само за тој да може да се позиционира во дадена точка. Со овој начин на управување, роботот не може да следи траекторија.

Друг начин на управување на робот за кој единствена цел е да се позиционира во дадена точка без притоа да се интересираме за обликот на траекторија туку едноставно сакаме роботот да не се судри со објектите од неговата околина, е да се користи методата на виртуелно потенцијално поле. При оваа метода целиот простор во кој се наоѓа роботот го замислуваме како простор во кој има наелектризирани честички. Самиот робот го заменуваме со негативно наелектризирана честичка што слободно се движи во просторот. Сите објекти од околината кои сакаме роботот да ги избегнува ги заменуваме со исто така негативно наелектризирана честичка. Целта ја заменуваме со позитивно наелектризирана честичка. Бидејќи од електротехника знеаме дека истоимено наелектризираните честички меѓусебно си дејствуваат со одбивни сили тоа значи дека меѓу роботот кој што е негативно наелектризиран и објектите кои исто така се негативно наелектризирани ќе се појават одбивни сили. Со други зборови роботот ќе ги избегнува објектите. Од друга страна пак, бидејќи целта е позитивно наелектризирана меѓу целта и роботот ќе се појават привлечни сили. Со други зборови овде на роботот ќе му дејствува сила која ќе го придвижува кон целта. Вкупната сила

може да се пресмета како сума од сите поединечни сили што дејствуваат на роботот. Кога веќе ни е позната силата што му дејствува на роботот со примена на инверзната матрица на Јакобијанот може да се пресметаат силите и торзионите моменти во зглобовите на роботот и ако тие сили т.е. торциони моменти ги предадеме на сервомоторчињата во зглобовите на роботот, роботот ќе се движи кон целната позиција притоа избегнувајќи ги објектите од непосредната околина. И на овој начин на управување на робот не е можно следење на однапред определена траекторија.

2.4 Избор на најдоброто можно решение

Во овој чекор се определува кои од понудените решенија ги задоволуваат поставените барања. Често повеќе од едно од добиените решенија ги задоволуваат поставените барања. Ние треба да избереме некое од тие решенија. Може да се користат неколку различни методи за оценување на некој поединечен дизајн. Може да се користи едноставен математички модел при кој се споредуваат предностите и недостатоците на решенијата. Вториот начин за оценување на дизајн е со споредба на определени параметри на дизајнот и со едноставна математичка споредба да се добие кое е оптималното решение. Овде од голема значење може да бидат софтверските алатки Solid Works и SimMechanics бидејќи со нивна употреба би можеле да добиеме квалитативни и квантитативни податоци за нашиот проект и поврзувајќи го тоа со метод за оценување на квалитет на проектот може да се определи кој е најдобриот дизајн. Со цел да го избереме најдоброто од понудените решенија потребно е да се постави некој критериум за оценка на понудените решенија. Таков критериум би можело да биде поврзан со прецизноста на позиционирањето, брзината на движење, енергетската ефикасност, сложеноста во изработката итн. Најчесто секој од наведените критериуми е важен, но не еднакво важен. Во таков случај за секој критериум што е важен во изборот на можно решение се поставува коефициент на важност. Колку наведениот критериум е важен толку коефициентот на важност на тој критериум е поголем. Потоа може да се стартува симулацијата и врз основа на добиените податоци од симулацијата за тие критериуми на важност може да се изврши изборот на можното решение.

2.5 Креирање на прототип

Кога веќе ги имаме резултатите од симулацијата на проектот и тие резултати ни ги задоволуваат поставените барања, останува само уште да се изработи прототип и доколку во практика се покаже дека прототипот доволно добро ја извршува својата задача може проектот да се достави за сериско производство и пласман на пазарот. Вообичаено приманата на CAD софтверски пакети значително ја поедноставува оваа фаза на проектирање и изработка на роботски манипулатори.

2.6 Тестирање и проценка на решенијата

Ова е фазата во која се врши тестирање на прототипот на манипулаторот во реалната средина. Во ваков случај од голема корист би било, доколку покрај квалитативни податоци, може да се добијат и квантитативни податоци за однесувањето на роботот во реалната средина и дополнително добиените податоци да може да се зачуваат на некоја надворешна меморија. Ова исто така е овозможено со Simulink од Matlab. Од добиените квантитативни податоци може да се види дали прототипот на манипулатор ги задоволува поставените барања и доколку не ги задоволува да се види што може да се направи за да се подобри проектот и да ги задоволи барањата. Во ваков случај, идеално би било прво да се размислува за најмалку три различни начини за решавање на проблемот пред да се сконцентрираме на некој поединечно. Откако ќе се

реши проблемот повторно се враќаме на чекорот тестирање и проценка на решенијата за да се види дали сега проектот ги задоволува барањата. Доколку манипулаторот ги задоволува поставените барања многу често и покрај тоа што проектот ги задоволува поставените барања може да се изнајде начин како проектот дополнително да се подобри.

2.7 Споредување на решенијата

Од сите модели на манипулатори кои што ги задоволуваат поставените барања потребно е да се избере само еден. Сакаме да го избереме најдоброто можно решение. Веќе во претходниот чекор ги имавме добиено квалитативните и квантитативните податоци за секој од прототиповите на манипулатор. Потребно е да ги искористиме овие податоци за да го избереме најдобриот проект. Во ваков случај треба да дефинираме критериум за проценка на манипулаторите. Во тој критериум се вклучуваат определени својства на манипулаторот. Такви својства би можело да бидат на пример:

- брзината на движење на манипулаторот
- дисипацијата на енергија
- тежината на предметот што манипулаторот е во состојба да го крене
- обликот на работниот простор на манипулаторот
- прецизноста во позиционирањето
- сложеноста на конструкцијата итн.

За секој од овие својства се додава тежински фактор (реален број) кој определува колку е значајно тоа својство во целокупниот критериум. На пример ако брзината на движење на манипулаторот е со тежински фактор 1, прецизноста во позиционирањето е значајна со тежински фактор 5 и колку блиску до меѓуточката поминува манипулаторот со тежински фактор 2 се добива следниот критериум:

$$K = v - 5 * e - 2 * e_m \quad (55)$$

Оваа формула се користи за секој прототип и оној прототип кој има најголема вредност за К ќе биде избран за најдобро можно решение.

2.8 Редизајнирање

Овде се разгледуваат резултатите од тестирањето на крајниот производ. Се разгледува дали добиениот модел ги задоволува поставените барања. Ако не ги задоволува се изнаоѓа начин како може да се промени за да ги задоволи, а и во случај да ги задоволува можно е да постои начин на кој може и покрај тоа да се подобри дизајнот. Доколку моделот ги задоволува поставените барања и нема потреба од понатамошно подобрување на проектот, проектот е подготвен за сериско производство и пласман на пазарот.

3 ПРАКТИЧЕН ДЕЛ

Во претходниот дел од оваа дипломска работа е даден општ опис на тоа како се изработува роботски манипулатор. Во овој дел е опишана практичната изведба на моделот и симулацијата на роботски манипулатор со 5 степени слобода на движење.

3.1 Идентификација на потребата или проблемот

Задачата на оваа дипломска работа е проектирање на сериски роботски манипулатор со 5 степени слобода на движење. Роботот треба да биде во состојба да се позиционира во која било точка од неговиот работен простор, додека ориентацијата на извршниот елемент во таа точка не е од интерес. Потребно е роботот да може да земе предмет што се наоѓа на некоја позиција од неговиот работен простор и да го премести на друга однапред дефинирана позиција.

3.2 Истражување на проблемот или на потребата

Информациите за проектот изработен во овој труд се дадени во Табела 3.

Табела 3 Информации за проектот pick and place роботски манипулатор

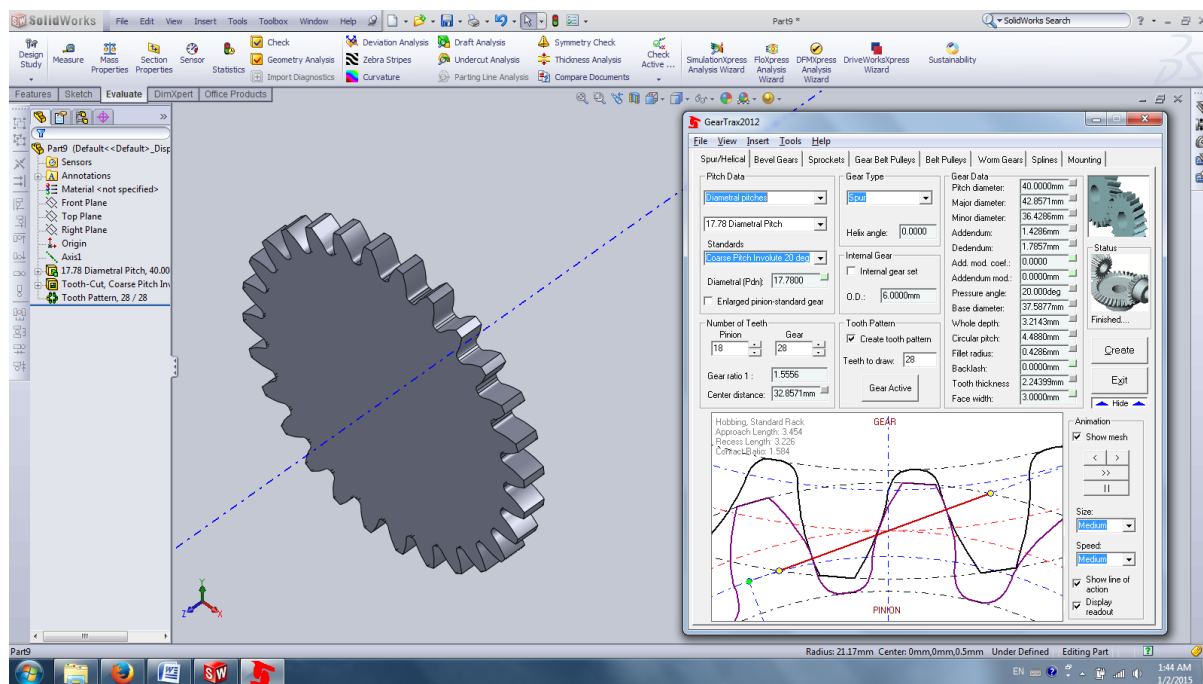
| СОБИРАЊЕ НА ИНФОРМАЦИИ | |
|--|--|
| 1. Која е практичната функција на проектот? Што роботот мора да биде во состојба да извршува? | Потребно е роботот да може да се постави во одредена позиција зададена преку координати, да може да го земе објектот што се наоѓа на таа позиција и да го пренесе на друга позиција зададена повторно со координати. |
| 2. Како роботот добива информации за позицијата на неговите зглобови, позицијата на објектот што треба да го фати, позицијата на која треба да го постави како и позициите на објектите од неговата околина со кои не треба да се судри? | Роботот во секој од зглобовите има вградено енкодер со помош на кој се добива дигитална информација за позицијата на зглобот. Роботот не е во состојба да ги детектира објектите од неговата околина. Нивната позиција треба да му биде соопштена на роботот од некој надреден уред. |
| 3. Кои материјали се соодветни за проектот? | Материјалот што може да се употреби за изградбата на нашиот манипулатор е алуминиум. Тој е цврст, лесен, отпорен е на корозија и е лесно достапен. Лесно се сече, обликува, буши и витка. Алуминиумот не е поцврст од челикот, но се карактеризира со поголем однос цврстина тежина што го прави покорисен во нашиот случај. Исто така, иако килограм челик е поефтин од килограм алуминиум, алуминиумот е многу полесен од челикот и не е потребно толку алуминиум колку би ни требало челик. |
| 4. Кој конструкциски метод е соодветен за проектот? | Методот на обработка кој ќе се користи зависи од дебелината на алуминиумот што се обработува. Во роботиката вообичаено се работи со мали дебелини на алуминиум, па затоа сечењето и бушењето на алуминиум се многу едноставни. |

3.3 Развивање на можните решенија

Развивањето на можните решенија е извршено со користење на CAD софтверскиот пакет SolidWorks. Во SolidWorks ќе биде изработена целата механичка структура на роботскиот манипулатор. При тоа е употребена и софтверската алатка GearTrax за изработка на запченици. По завршувањето на изработката на механичкиот модел на роботски манипулатор, со помош на програмата solidworks2simmechanics translator добиениот модел е преведен од .sldasm формат во .xml формат и потоа импортиран во Simulink на Matlab. Ова е неопходно за да може да се изврши проектирање на управувањето на роботскиот манипулатор во SimMechanics како дел од Simulink на Matlab. При проектирањето на управувањето се употребени функциите од библиотеката на Robotic toolbox - софтверска алатка изработена за примена во Matlab.

3.3.1 Изработка на механичката структура

Оваа фаза на проектирање на роботски манипулатор одзема најмногу време. Во оваа фаза потребно е многу иновативност и креативност. Најпрво треба да се изработи секој составен дел на роботскиот манипулатор поединечно и потоа составните делови се асемблираат во еден фајл, наречен Assem.sldasm. Составните делови од кои е составен манипулаторот се дадени во прилог А. Од деловите претставени во прилогот А и од целосниот роботски манипулатор претставен на Слика 16 може да се забележи дека во извршниот елемент се употребени два запченици. Запчениците се карактеризираат со сложена геометрија и токму поради тоа нивното изработување во CAD софтвер како што е Solidworks не е едноставно. За таа цел е искористена софтверската алатка GearTrax во која се внесуваат параметрите на запченикот што сакаме да го изработиме во SolidWorks. Алатката GearTrax автоматски го изработува моделот на запченикот во SolidWorks врз основа на внесените параметри. Потоа тој запченик може да се користи во составувањето на моделот на роботскиот манипулатор. Сите податоци за запченикот што се внесени во алатката GearTrax се доволни за било која компанија за изработка на запченици да го изработи нашиот запченик онака како што ние го имаме замислено.



Слика 15 Изработка на запченик со користење на алатката GearTrax

Изработениот и составен роботски манипулатор е претставен на Слика 16.



Слика 16 Изработен и составен роботски манипулатор со 5 степени на слобода на движење

3.3.2 Пресметка на параметрите потребни за управување

За пресметување на параметрите што се потребни при проектирањето на управувачот е употребена софтверската алатка Robotic toolbox.

3.3.2.1 Кинематска анализа на манипулаторот

Користењето на Robotic toolbox може многу да го олесни вршењето на кинематската анализа на манипулаторот. Со помош на функциите од таа софтверска алатка се вршат сложените пресметки што се среќаваат во кинематската анализа. На овој начин значително се намалува времето потребно да се изврши кинематската анализа на манипулаторот и истовремено се елиминира можноста за грешки при пресметувањето.

3.3.2.1.1 Директна кинематика

Добрата страна на SimMechanics при пресметка на директната кинематика е тоа што со самото импортирање на моделот се подразбира директната кинематика и нема потреба од нејзино пресметување. Со тоа се заштедува време и се елиминира можноста за грешка при пресметувањето на директната кинематика.

Во SimMechanics, директната кинематика е претставена преку блокови. SimMechanics се разликува од Simulink на Matlab. Основната разлика помеѓу SimMechanics и Simulink е тоа што блоковите во Simulink при компајлирањето се преведуваат во математички функции кои потоа се искористуваат како математички функции и резултатот се добива токму со извршувањето на тие математички функции, додека во SimMechanics при компајлирањето не настанува преведување на

SimMechanics блоковите во математички функции. Доколку ни се потребни математичките функции што ја претставуваат директната кинематика на роботскиот манипулатор ќе мора пресметувањето на директната кинематика да се изврши рачно. Тоа овде е направено преку Denavit - Hartenberg методата за пресметка на директната кинематика и при тоа се користени функциите од Robotic toolbox.

Параметрите на Denavit - Hartenberg за роботскиот манипулатор од Слика 16 се дадени во Табела 4.

Табела 4 Параметрите на роботскиот манипулатор според Denavit - Hartenberg методата

| i | d_i | θ_{i1} | a_i | α_{i1} |
|---|-------|-----------------|-------|---------------|
| 1 | d_1 | θ_{11}^* | 0 | 90 |
| 2 | 0 | θ_{21}^* | a_2 | 0 |
| 3 | 0 | θ_{31}^* | a_3 | 0 |
| 4 | 0 | θ_{41}^* | 0 | -90 |
| 5 | d_5 | θ_{51}^* | 0 | 0 |

Потоа со користење на таа табела и со функциите од Robotic toolbox се добива трансформационата матрица

$${}_R T^n = \begin{bmatrix} n_x & a_x & o_x & x \\ n_y & a_y & o_y & y \\ n_z & a_z & o_z & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (56)$$

која го претставува решението на директната кинематика на роботскиот манипулатор. Тројката (x, y, z) ги претставува координатите на позиција на крајниот извршен елемент на роботот во однос на референтниот координатен систем.

3.3.2.1.2 Инверзна кинематика

Од резултатот од директната кинематика на роботскиот манипулатор познати ни се функциите x, y, z кои ја даваат зависноста на позицијата на крајниот извршен елемент во однос на позицијата на зглобовите на роботот. Тоа се три нелинеарни функции со 5 непознати. Во општ случај најдобро е ако може да се најде инверзна функција на x, y, z функциите. Но, роботот се карактеризира со сложена нелинеарност и аналитичкото наоѓање на инверзните функции на x, y и z функциите не е можно. Токму поради тоа, решавањето на инверзната кинематика е направено со помош на итеративна метода. Бидејќи треба да се реши систем од 3 равенки со 5 непознати решението на инверзната кинематика во нашиот случај ќе има бесконечно многу решенија. Нас ќе ни биде потребно да се најде едно од тие решенија. За таа цел може да се користи функцијата ikine() од софтверската алатка Robotic toolbox. Оваа функција ја определува инверзната кинематика со повеќе итерации. Секоја од овие итерации зафаќа определено време што го прави извршувањето на функцијата споро. Со цел пресметувањето на инверзната кинематика да се одвива значително побрзо, во рамките на оваа дипломска работа е испрограмирана нова функција која што ја нареков invkine() која што коректно ја пресметува инверзната кинематика за многу пократко време.

3.3.2.2 Јакобијан и анализа на сингуларитети

И при пресметувањето на Јакобијанот и при анализата на сингуларитетите, Robotic toolbox алатката може да биде од голема корист. Може да се најде матрицата на Јакобијанот со едноставно повикување на наредбата `jacob0`. Аналитичкото наоѓање на сингуларитетите се одвива со пресметување на вредностите на аглите на зглобовите на роботот за кои матрицата на Јакобијанот претставува сингуларна матрица. При тоа се добива дека сингуларитети постојат само на границите на работниот простор на роботот.

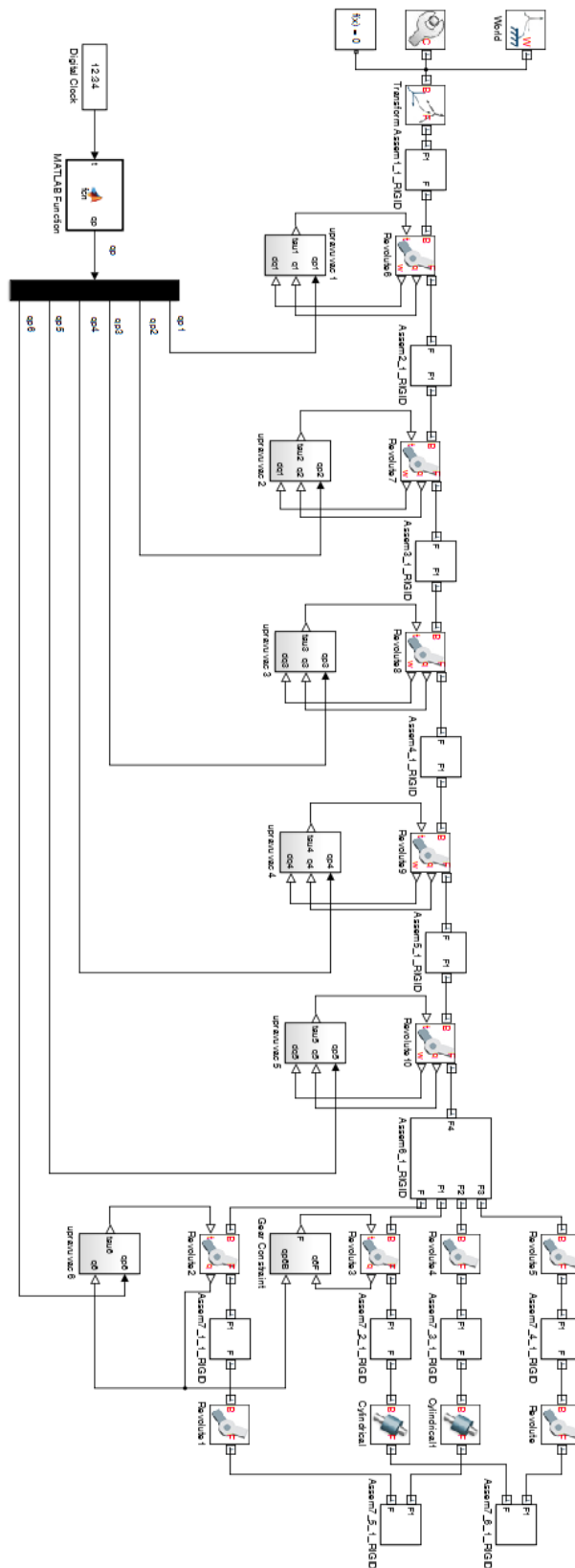
3.3.2.3 Лагранжова механика

Овде се врши анализа на динамиката на манипулаторот. Потребно е да се пресметаат динамичките равенки од кои подоцна ќе зависат параметрите на управувањето. Овде има најмногу пресметувања, па токму од оваа причина неизбежно е употребата на компјутер за вршење на тие пресметки. Со користење на функциите од Matlab за вршење на пресметките значително се намалува времето потребно да се изврши динамичката анализа на манипулаторот и се елиминира можноста за грешки при пресметувањето.

3.3.3 Управување на манипулаторот

Во општиот дел опишав неколку различни начини на кои може да се врши управувањето на роботскиот манипулатор. Во симулацијата беа симулирани однесувањето на манипулаторот управуван на следниве три начини: управување на манипулаторот во просторот на зглобови, управување на манипулаторот во Декартовиот простор со користење на инверзната кинематика и управување на манипулаторот во Декартовиот простор со користење на директната кинематика и Јакобијанот.

Блок шемата во SimMechanics на управување на манипулаторот во просторот на зглобовите е дадена на Слика 17. При тоа, во блокот Matlab Function block се исчитуваат почетната позиција на манипулаторот, меѓуточките и крајната позиција на манипулаторот сите претставени во просторот на зглобовите. Врз основа на тие податоци се врши генерирање на траекторија и потоа следење на таа траекторија.



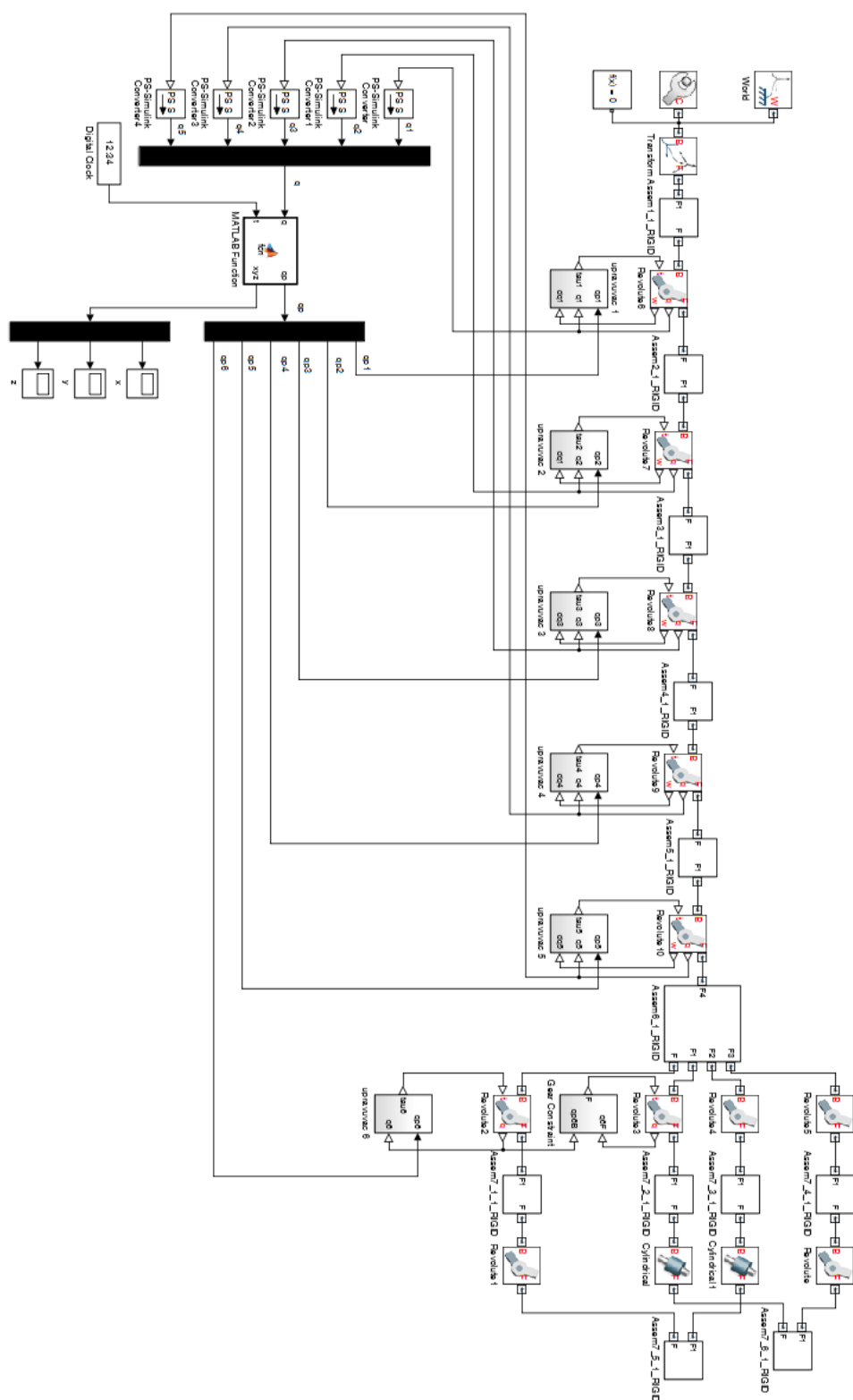
Слика 17 Блок шема на управување на роботскиот манипулатор од слика 16 во просторот на зглобови

Вториот начин на управување на манипулаторот е во Декартовиот простор со користење на инверзната кинематика. Блок шемата на овој начин на управување во

SimMechanics е идентична со блок шемата на управување во просторот на зглобовите. Единствена разлика е во кодот сместен во Matlab Function блокот. Во овој случај во Matlab Function блокот почетната позиција на манипулаторот, позицијата на меѓуточките и крајната позиција на манипулаторот се дадени во Декартови координати. Најпрво пред да се започне со движењето на манипулаторот потребно е да се провери дали секоја од дадените позиции навистина се наоѓа во работниот простор на роботот, т.е. дали движењето е изводливо. Доколку се добие дека движењето не може да се изврши ќе се јави грешка и роботот нема ни да започне со движење. Доколку се е во ред со избраните позиции, тогаш се врши генерирањето на траекторијата во Декартовиот простор. Овде се генерира траекторија на линеарни сегменти со параболични закривувања на краевите. При секое извршување на Matlab Function блокот се пресметува посакуваната позиција од траекторијата во моментот t во Декартови координати. Потоа се врши инверзната кинематика и од неа се добиваат позициите на зглобовите на роботот за тој да се најде во таа позиција, дадена во Декартови координати. Тие податоци се предаваат на управувачот на секој од зглобовите и на тој начин роботот ја следи посакуваната траекторија.

Со овој начин на управување на манипулаторот можно е следење на било која траекторија. Со други зборови, наместо да ни бидат дадени почетната позиција, позицијата на меѓуточките и крајната позиција на манипулаторот, врз основа на кои треба да се креира траекторија, траекторијата што треба да ја следи манипулаторот може да биде дадени и експлицитно. Ако експлицитно ни беше дадена траекторијата што треба да се следи тогаш нема да има потреба од генерирање на траекторија и следењето на таа траекторија ќе се одвива така што во секој временски момент ќе се пресметува посакувата позиција од траекторијата и потоа со користење на инверзната кинематика може да се пресмета позицијата на зглобовите. Тие податоци се предаваат на управувачите на зглобовите.

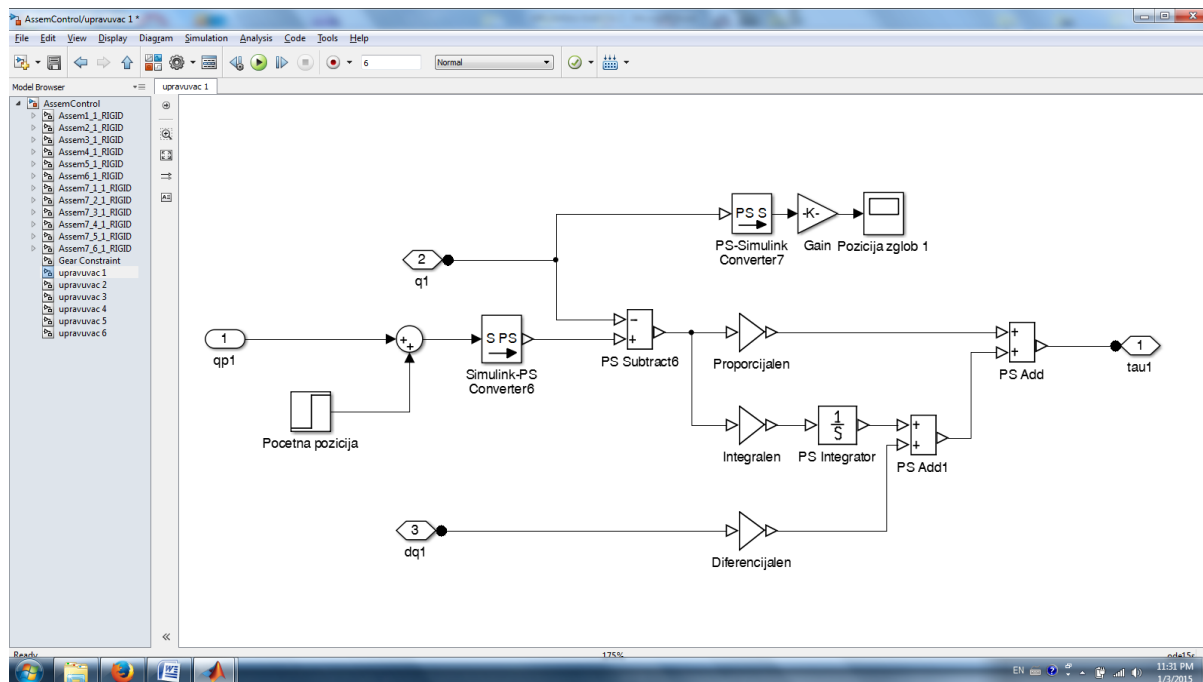
При овој начин на управување на манипулаторот се забележува присуството на инверзната кинематика во секое извршување на Matlab Function блокот. Инверзната кинематика се пресметува со повеќе итерации чие што пресметување зафаќа одредено време. Тоа придонесува за помала фреквенција на ажурирање на податоците на излезот од Matlab Function блокот. Од таа причина се намалува и брзината со која се движи манипулаторот и неговата прецизност во позиционирањето. Многу корисно би било доколку инверзната кинематика може да се замени со нешто што ќе може да се извршува значително побрзо. Тоа е направено во третиот начин на управување на манипулаторот. Овде наместо инверзната кинематика се користи директната кинематика и инверзната матрица на Јакобијанот. Овде мора да се напомене дека овој начин на управување не може да се користи во случај на сингуларитет бидејќи во тој случај не може да се пресмета инверзната матрица на Јакобијанот. Токму поради тоа случаите на сингуларитет ќе бидат третирани како специјални случаи. Блок шемата на овој тип на управување на манипулаторот е претставена на Слика 18.



Слика 18 Блок шема на управување на роботскиот манипулатор од Слика 16 во Декартовиот простор со користење на директната кинематика и инверзниот Јакобијан

Од блок шемата се забележува дека покрај времето како влез во Matlab function блокот се поставени и тековните вредности на аглите на зглобовите.

Во секој од претходно опишаните начини на управување на манипулаторот се среќава блокот управувач за секој од зглобовите по еден. Овој тип на управувач е од типот на ПИД управувач. Блок шемата на ПИД управувачот на зглобот 1 е претставена на следната слика.



Слика 19 Блок шема на ПИД управувач

За ПИД управувачот потребно е да се пресметаат параметрите K_p , K_d и K_i . За пресметување на овие параметри може да се користат повеќе различни методи. Во дипломскиот труд е применет методот на пресметка на параметрите на управувачот опишан во делот 2.3. При тоа пресметаните вредности за параметрите се:

$$K_p = [1878.4 \quad 1957 \quad 1895.2 \quad 1981.3 \quad 1960.2 \quad 2000] \quad (57)$$

$$K_d = [-87.84 \quad -95.7 \quad -89.52 \quad -98.13 \quad -96.02 \quad -100] \quad (58)$$

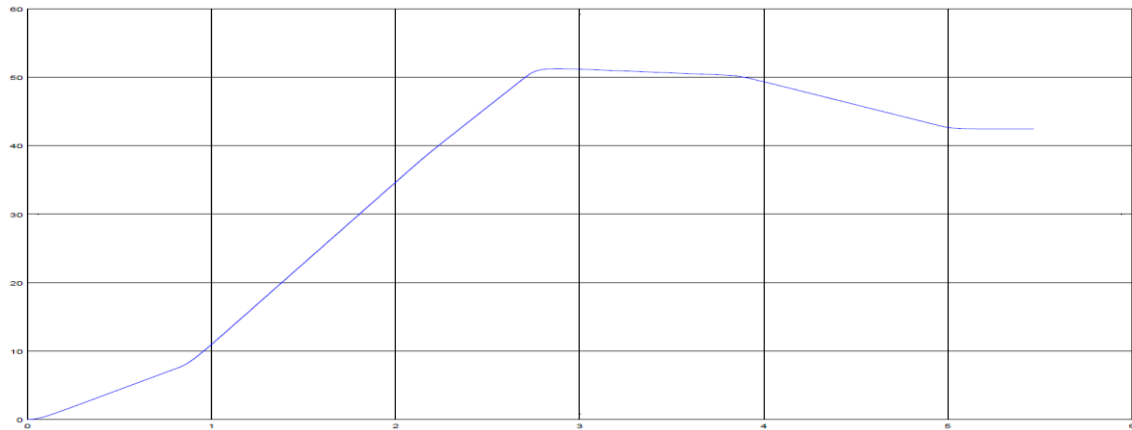
$$K_i = [-12.16 \quad 4.3 \quad 10.48 \quad 1.87 \quad 3.98 \quad 0] \quad (59)$$

3.4 Избор на најдоброто можно решение

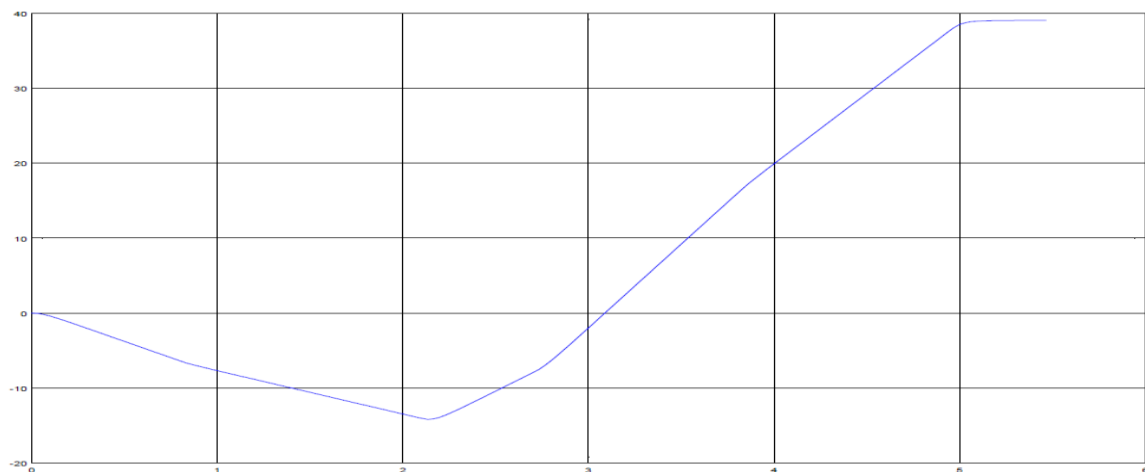
Во оваа фаза се врши симулацијата и се прегледуваат резултатите. Најпрво е симулирано однесувањето на манипулаторот кога тој е управуван во просторот на зглобовите. При тоа почетната позиција на роботот, меѓуточките и крајната позиција на роботот се дадени во следната матрица

$$q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 8.1301 & -7.0018 & -17.4214 & 50.5380 & 0 \\ 38.6598 & -14.4134 & -9.6907 & 58.7247 & 0 \\ 51.3402 & -7.0297 & -6.3489 & 49.2004 & 0 \\ 50.1944 & 17.4999 & -18.6145 & 49.2004 & 0 \\ 42.4362 & 39.0145 & -36.2960 & 29.8306 & 0 \end{bmatrix} \quad (60)$$

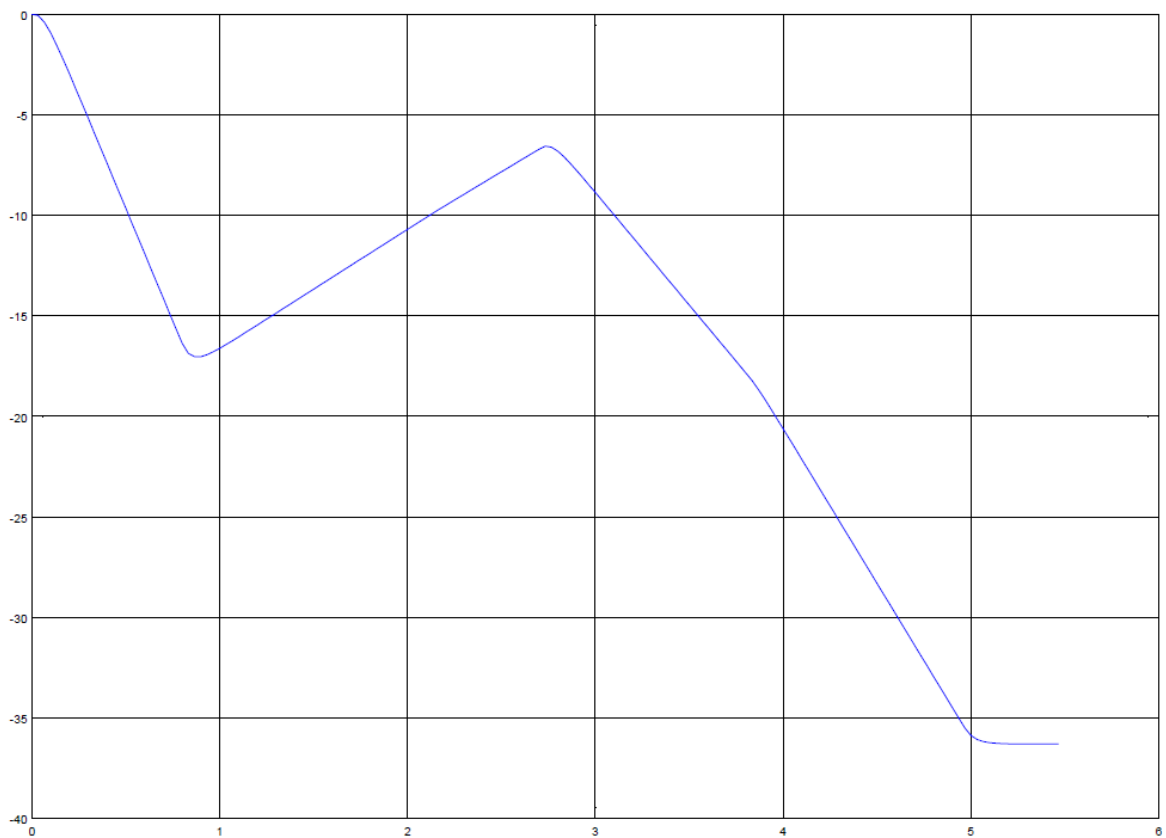
При ова управување мерени се вредностите на аглите на зглобовите на манипулаторот и x, y и z координатата на позицијата на извршниот елемент. Резултатите од симулацијата се претставени на сликите 20, 21, ... 26.



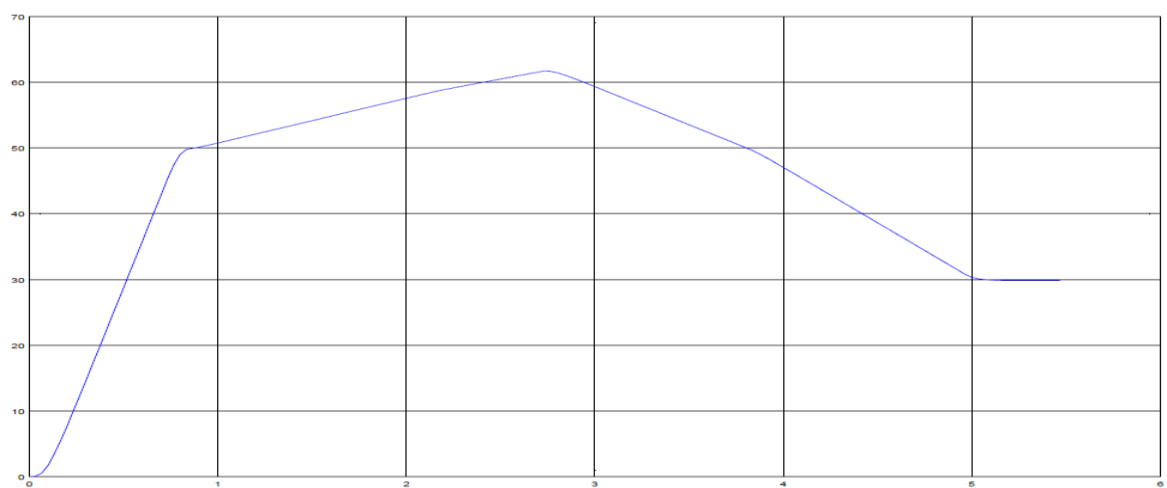
Слика 20 Позицијата на зглоб 1 во симулацијата на управување на роботскиот манипулатор во просторот на зглобови



Слика 21 Позицијата на зглоб 2 во симулацијата на управување на роботскиот манипулатор во просторот на зглобови

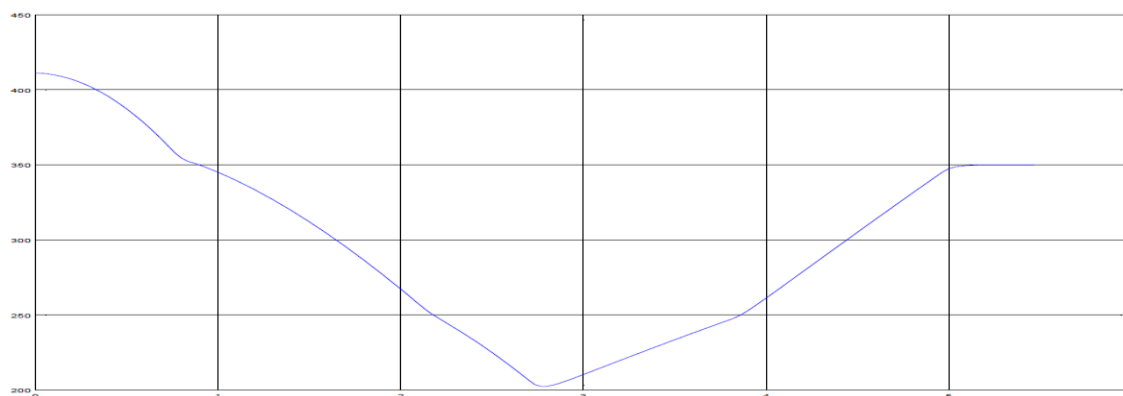


Слика 22 Позицијата на зглоб 3 во симулацијата на управување на роботскиот манипулатор во просторот на зглобови

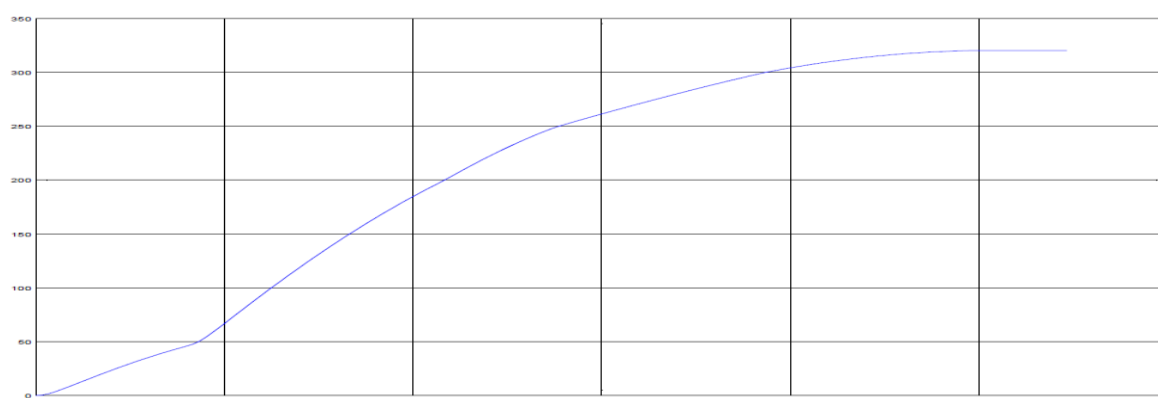


Слика 23 Позицијата на зглоб 4 во симулацијата на управување на роботскиот манипулатор во просторот на зглобови

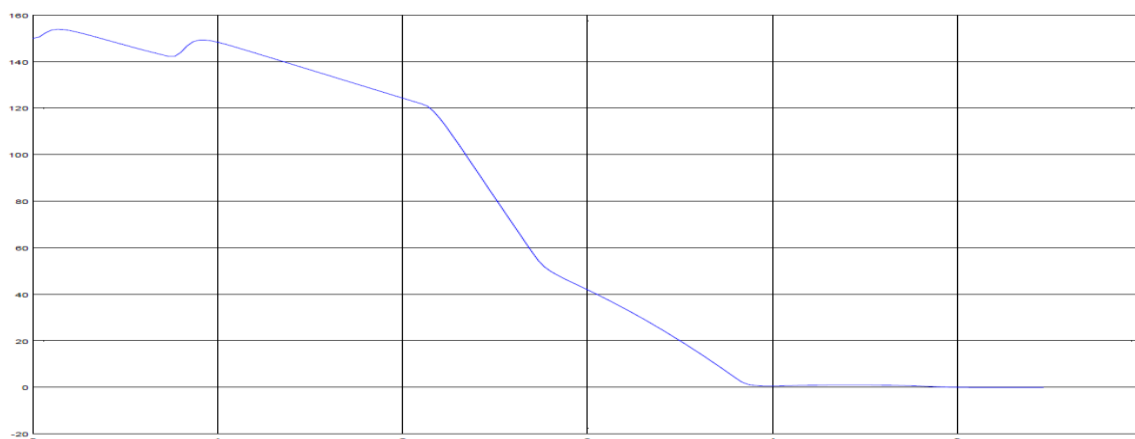
При тоа во Декартовиот координатен систем се добиваат граfiците од Слика 24, 25 и 26.



Слика 24 Координата x на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во просторот на зглобови



Слика 25 Координата y на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во просторот на зглобови

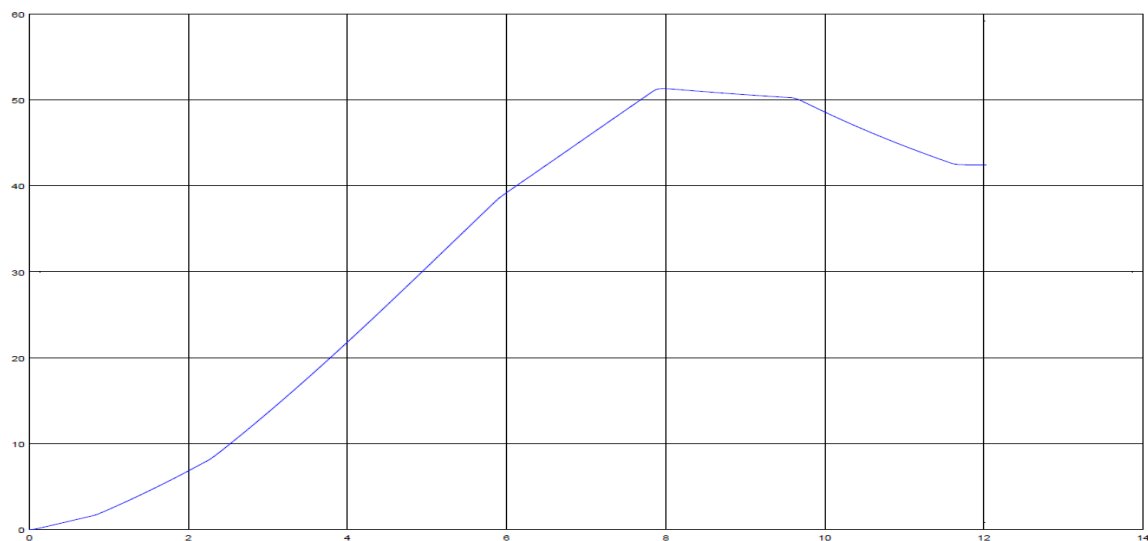


Слика 26 Координата z на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во просторот на зглобови

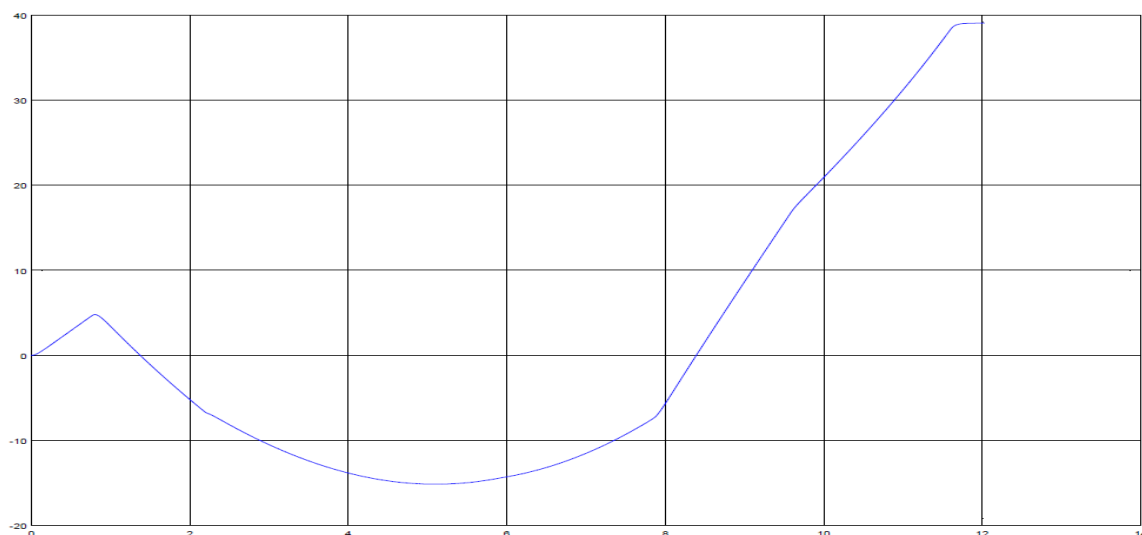
Од овие резултати од симулацијата може да се забележи дека манипулаторот успешно ја следи траекторијата генерирана во просторот на зглобови. Може да се забележи дека манипулаторот поминува низ дефинираните меѓуточки и на крајот се позиционира во крајната позиција. Бидејќи генерирањето на траекторијата се одвива во просторот на зглобови изгледот на таа траекторија во просторот на зглобови е онаков

каков што ние сме избрале да биде додека во Декартовиот простор се добива траекторијата. Тоа значи дека во Декартовиот простор траекторијата не мора да наликува на непрекршена права линија со мали закривувања во почетната точка, меѓуточките и крајната точка. Тоа може да се забележи од претставените резултати.

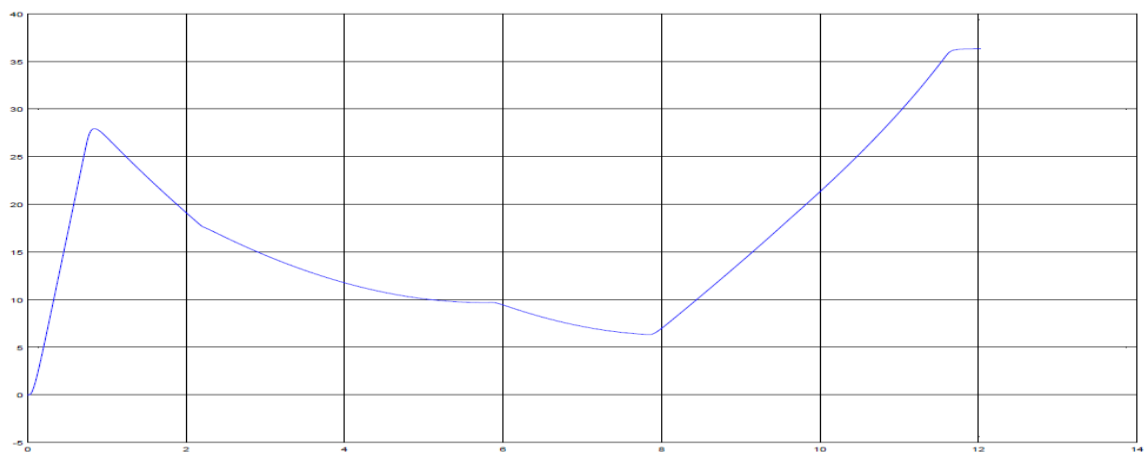
Вториот начин на управување на роботскиот манипулатор за кој е извршена симулација е управување на манипулаторот со користење на инверзната кинематика и при тоа се мерени следниве сигнали: вредностите на аглите на зглобовите на манипулаторот и x , y и z координатата на позицијата на извршниот елемент. Резултатите од симулацијата се претставени на следниве слики.



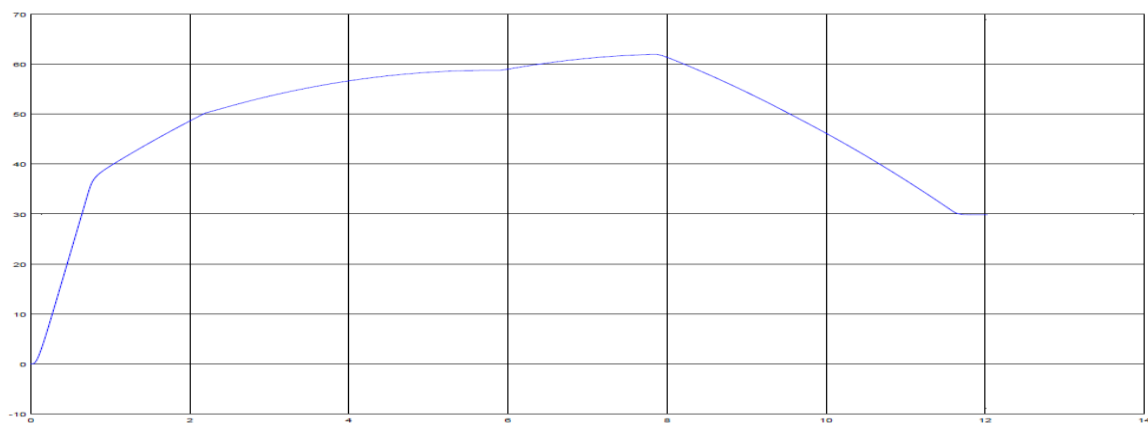
Слика 27 Позицијата на зглоб 1 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на инверзната кинематика



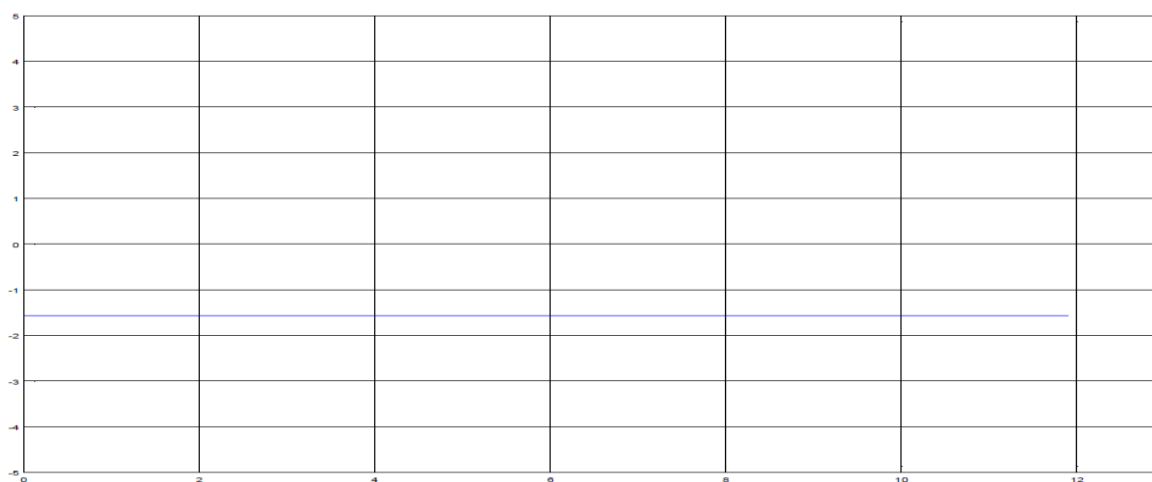
Слика 28 Позицијата на зглоб 2 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на инверзната кинематика



Слика 29 Позицијата на зглоб 3 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на инверзната кинематика

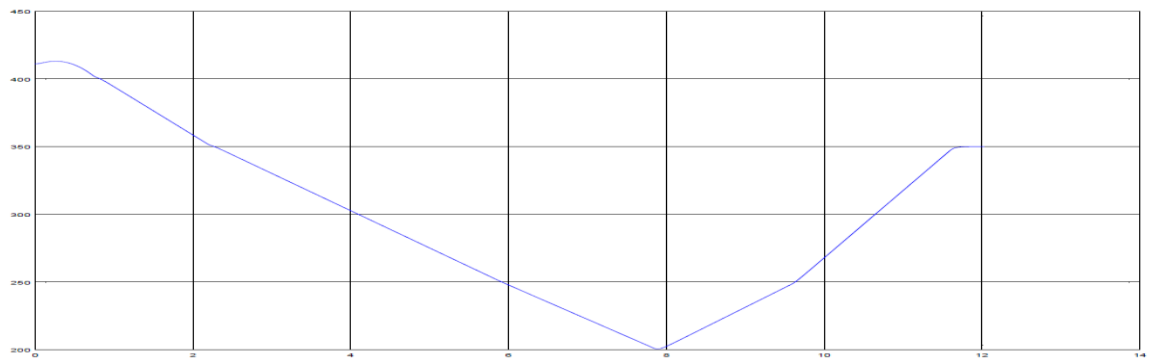


Слика 30 Позицијата на зглоб 4 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на инверзната кинематика

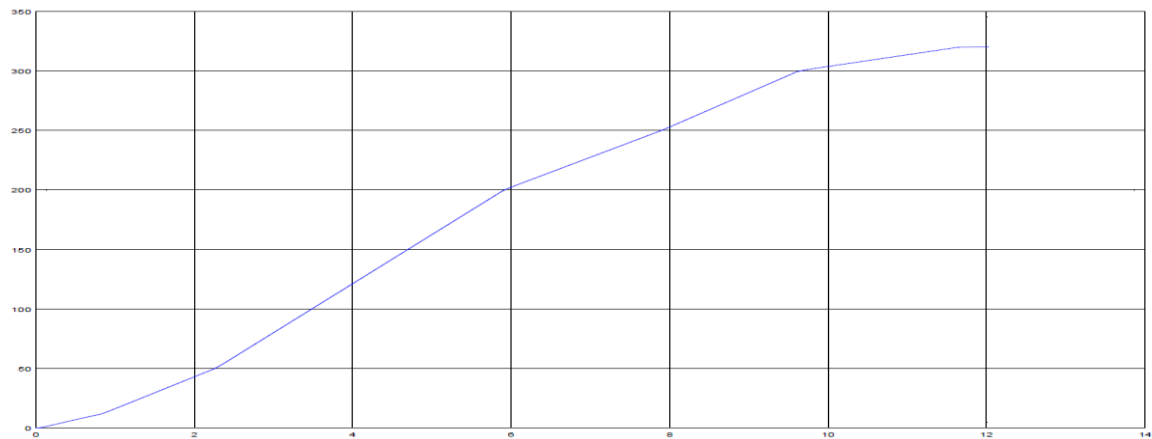


Слика 31 Позицијата на зглоб 5 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на инверзната кинематика

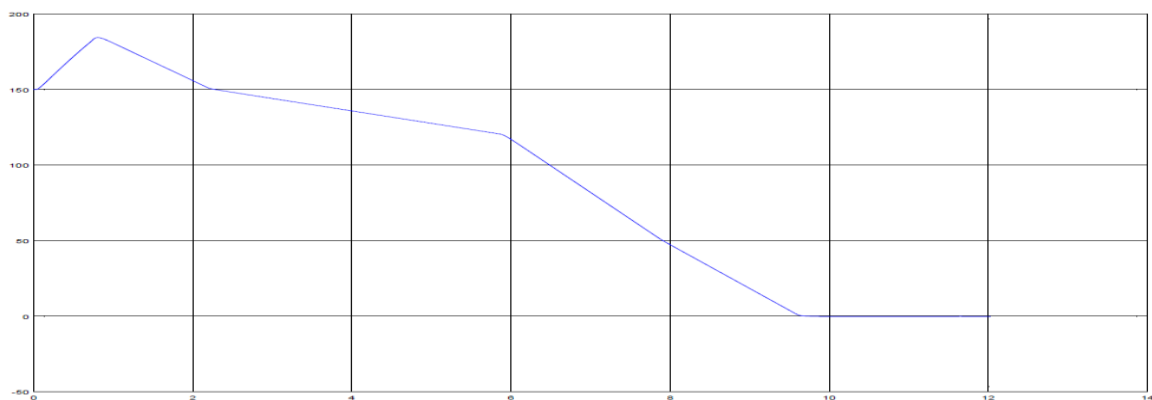
Координатите на позицијата на извршниот елемент при управување на роботскиот манипулатор во картезијанскиот простор се дадени на сликите 32, 33 и 34.



Слика 32 Координата x на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на инверзната кинематика



Слика 33 Координата y на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на инверзната кинематика



Слика 34 Координата z на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на инверзната кинематика

Од резултатите од симулацијата може да се забележи дека роботот се придвижува од почетна позиција определена со позиција во Декартовиот простор $xyz = (400, 11.5, 185)$ до крајна позиција определена со $xyz = (350, 320, 0)$. При тоа роботот треба да избегнува судири со објектите од неговата непосредна околина поради што потребно е роботот да се движи по определена патека (траекторија). Поради постоење на објекти по права патека од почетната позиција на роботот до посакуваната позиција, во траекторијата на роботот има вметнато 4 меѓуточки, па целото движење на роботот е поделено на пет сегменти и тоа:

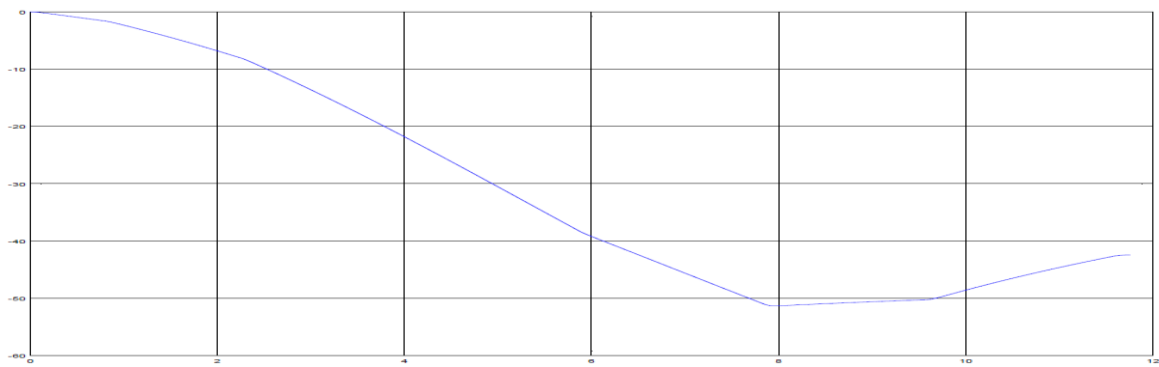
1. Прв сегмент што започнува од почетната позиција на роботот и завршува во првата меѓуточка $xyz = [350 \ 50 \ 150]$
2. Втор сегмент од прва меѓуточка до втора меѓуточка $xyz = [250 \ 200 \ 120]$
3. Трет сегмент од втората меѓуточка до третата меѓуточка $xyz = [200 \ 250 \ 50]$
4. Четврт сегмент што започнува од четвртата меѓуточка и завршува во петтата меѓуточка $xyz = [250 \ 300 \ 0]$
5. Петти сегмент од петтата меѓуточка до крајна позиција на роботот.

Секој од овие сегменти претставува линеарен сегмент со мали закривувања на краевите. При тоа од резултатите може да се забележи дека роботот многу добро ја следи определената траекторија која што поминува точно низ поставените меѓуточки што значи дека роботот успешно ги избегнува објектите од неговата непосредна околина. Потоа може да се забележи дека роботот во точно определениот временски момент ја постигнува посакуваната позиција. Движењето на роботот е глатко, значи нема непотребно застанување во меѓуточките и непотребно трошење на енергија. Со други зборови роботот ги задоволува поставените барања.

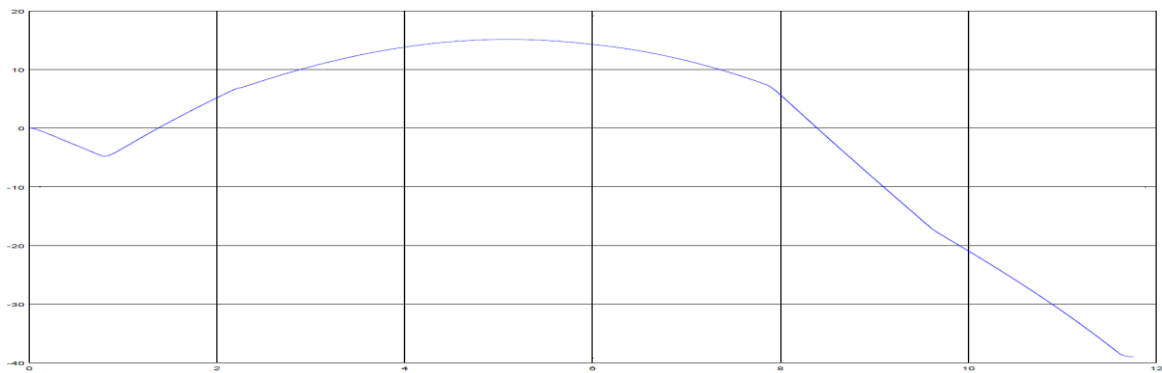
Во нашиот случај на управување на роботскиот манипулатор беа дадени почетната позиција, меѓуточките и крајната позиција и беше генерирана траекторија по која се движеше манипулаторот. Освен овој начин на управување на манипулаторот со користење на инверзната кинематика можно е и управување на манипулаторот така што траекторија што треба да ја следи ќе биде дадена експлицитно од некој нареден уред.

Третиот начин на управување на роботскиот манипулатор за кој е извршена симулација е управување на манипулаторот со користење на директната кинематика и инверзната матрица на Јакобијанот и при тоа се мерени следниве сигнали: вредностите на аглите на зглобовите на манипулаторот и x , y и z координатата на позицијата на извршниот елемент. Резултатите од симулацијата се претставени на Слика 35, 36, ..., 42. Во (61) се дадени вредности на почетната позиција, меѓуточките и крајната позиција на извршниот елемент на манипулаторот за кои е извршена симулацијата.

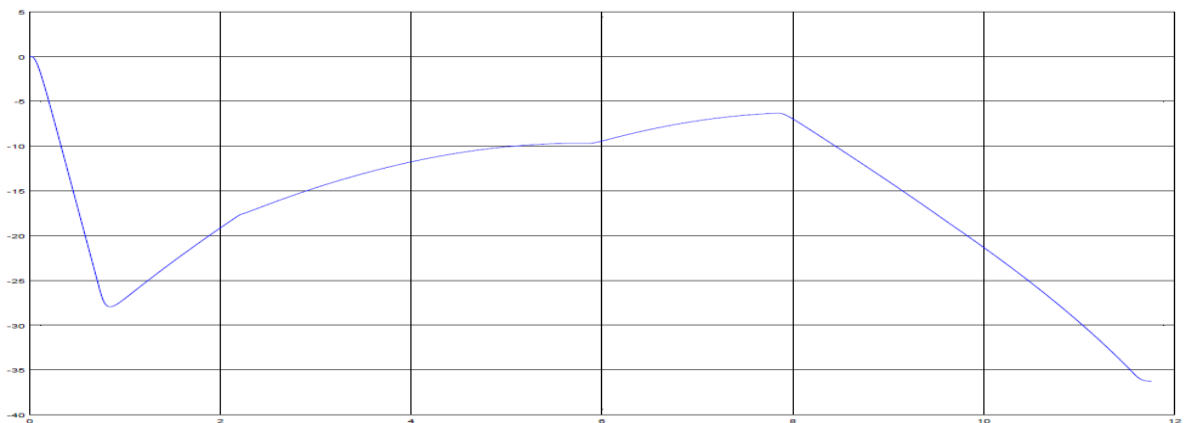
$$xyz = \begin{bmatrix} 400 & 11.5 & 185 \\ 350 & 50 & 150 \\ 250 & 200 & 120 \\ 200 & 250 & 50 \\ 250 & 300 & 0 \end{bmatrix} \quad (61)$$



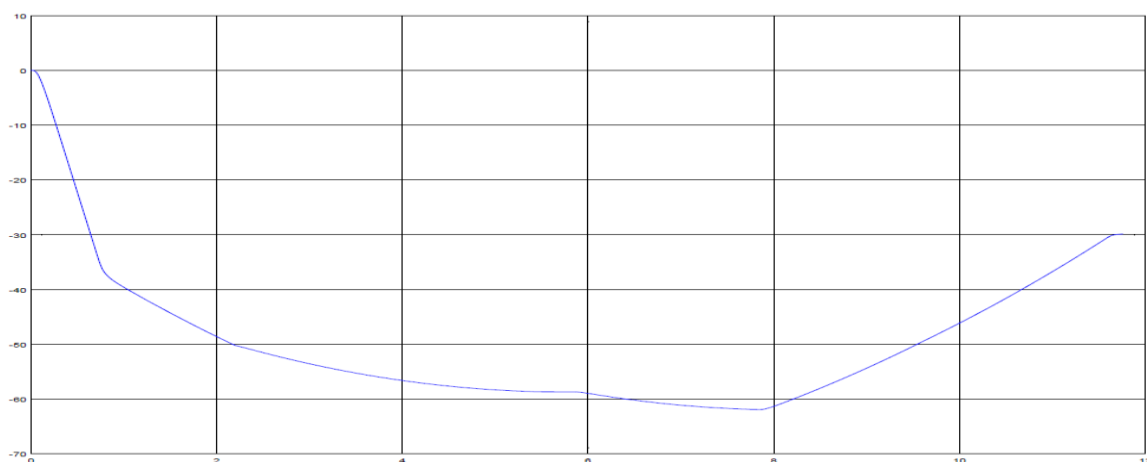
Слика 35 Позицијата на зглоб 1 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот



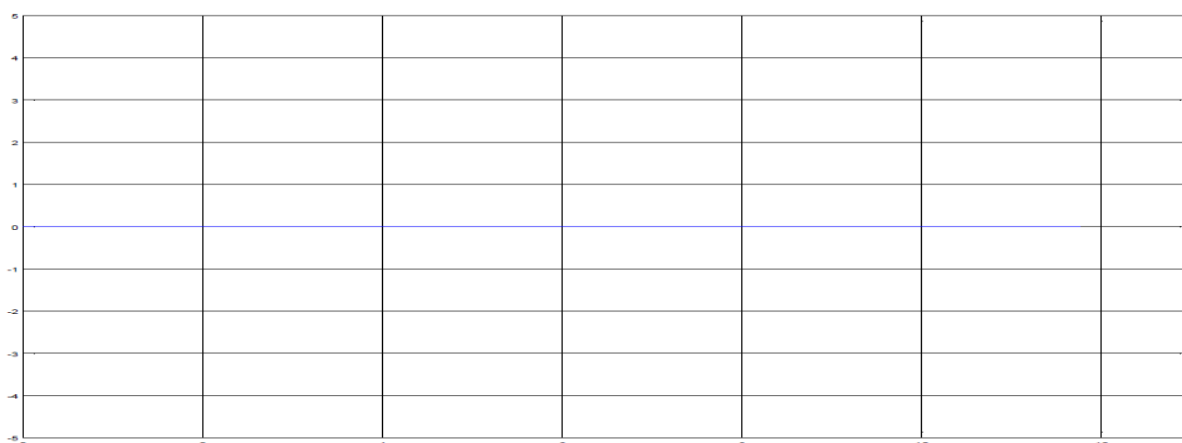
Слика 36 Позицијата на зглоб 2 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот



Слика 37 Позицијата на зглоб 3 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот



Слика 38 Позицијата на зглоб 4 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот

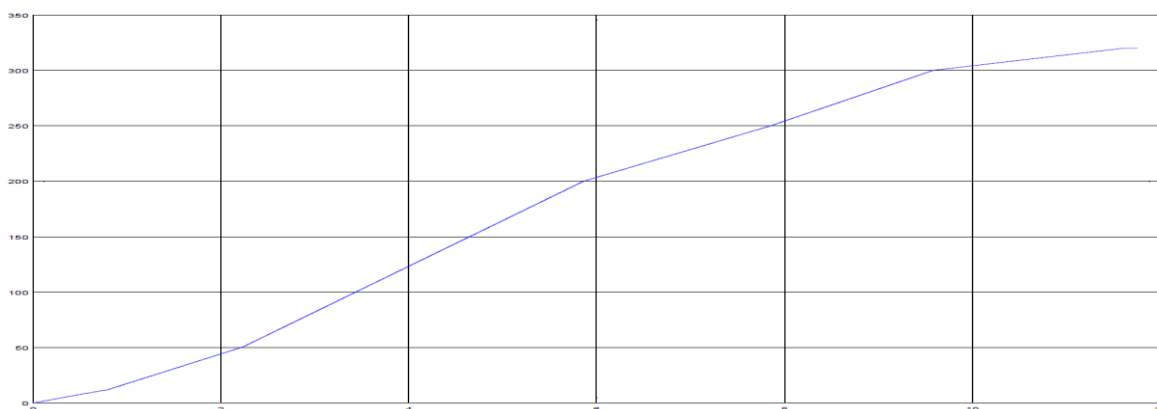


Слика 39 Позицијата на зглоб 5 во симулацијата на управување на роботскиот манипулатор во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот

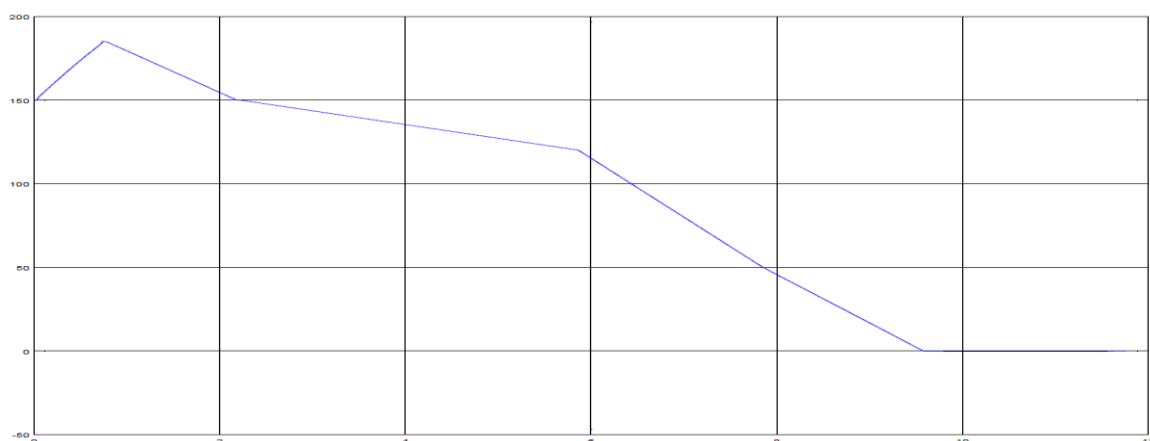
Изгледот на траекторијата во Декартови координати е прикажан на сликите 40, 41 и 42.



Слика 40 Координата x на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот



Слика 41 Координата y на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот



Слика 42 Координата z на крајниот извршен елемент во Декартови координати добиена со симулација на роботскиот манипулатор управуван во Декартовиот простор со користење на директната кинематика и инверзната матрица на Јакобијанот

Од овие резултати може да се забележи дека манипулаторот успешно ја следи траекторијата генерирана во Декартовиот простор. Исто така може да се забележи дека манипулаторот поминува низ точно дефинираните меѓуточки и на крајот движењето го завршува во посакуваната позиција. При овој начин на управување на манипулатор покрај тоа што може да се генерира траекторија и потоа манипулаторот да ја следи таа генерирана траекторија, може да се модифицира проблемот така што манипулаторот да следи произволна крива траекторија.

3.5 Креирање на прототип

Во оваа фаза на проектирање и изградба на роботски манипулатор се врши материјалната изработка на прототип на манипулатор.

Доколку ја имаме користено Solid Works како CAD софтверска алатка за проектирање на механичкиот дел од проектот, може многу едноставно да се изработи дводимензионална скица на секој од искористените делови во механиката на проектот. Потоа таа дводимензионална скица може да се внесе во машина и таа машина да го изработи делот според добиената спецификација. Изработката на дводимензионалната

скица во Solid Works може да се одвива автоматски. Добиените скици се дадени во прилог А.

Откако ќе се изработат деловите и ќе се состави манипулаторот останува само уште програмата на управувачот. При изработката на симулацијата во Simulink, ја изработивме и програмата на управувачот. Таа програма може лесно да се искомпајлира и добиената извршна програма да се внесе во управувачот на релниот роботски манипулатор. Програмата во Matlab код може да се види во прилог Б.

Откако и програмата ќе биде внесена во управувачот на манипулаторот, прототипот е целосно изработен и може да се пристапи кон негово тестирање во реална средина.

3.6 Тестирање и проценка на решенијата

Ова е фазата во која се врши тестирање на прототипот на манипулаторот во реална средина. Simulink од Matlab ни овозможува поврзување на прототипот со компјутер. На тој начин се овозможува добивање на квантитативни податоци за однесувањето на прототипот во релната средина. Дополнително тие податоци може да се зачуваат на хард диск или некоја друга надворешна меморија. Од добиените квантитативни податоци може да се види дали прототипот ги задоволува поставените барања. Ако не ги задоволува потребно е да се направат одредени модификации за да ги задоволи. Во некои случаи може роботот да ги задоволи поставените барања, но и покрај тоа што ги задоволува поставените барања може да се изнајде начин на кој може дополнително да се подобри проектот.

3.7 Споредување на решенијата

Од сите модели на манипулатори кои што ги задоволуваат поставените барања потребно е да се избере само еден. Сакаме да го избереме најдоброто можно решение. Веќе во претходниот чекор ги имавме добиено квалитативните и квантитативните податоци за секој од прототиповите на манипулатор. Потребно е да ги искористиме овие податоци за да го избереме најдобриот проект. Во ваков случај треба да дефинираме критериум за проценка на манипулаторите. Во тој критериум се вклучуваат определени својства на манипулаторот. Такви својства би можело да бидат на пример:

- брзината на движење на манипулаторот
- дисипацијата на енергија
- тежината на предметот што манипулаторот е во состојба да го крене
- обликот на работниот простор на манипулаторот
- прецизноста во позиционирањето
- сложеноста на конструкцијата итн.

За секој од овие својства се додава тежински фактор (реален број) кој определува колку е значајно тоа својство во целокупниот критериум. На пример ако брзината на движење на манипулаторот е со тежински фактор 1, прецизноста во позиционирањето е значајна со тежински фактор 5 и колку блиску до меѓуточката поминува манипулаторот со тежински фактор 2 се добива следниот критериум:

$$K = v - 5 * e - 2 * e_m$$

Оваа формула се користи за секој прототип и оној прототип кој има најголема вредност за K ќе биде избран за најдобро можно решение.

3.8 Редизајнирање

Овде се разгледуваат резултатите од тестирањето на крајниот производ. Се разгледува дали добиениот модел ги задоволува поставените барања. Ако не ги задоволува се изнаоѓа начин како може да се промени за да ги задоволи, а и во случај да ги задоволува можно е да постои начин на кој може и покрај тоа да се подобри дизајнот. Доколку моделот ги задоволува поставените барања и нема потреба од понатамошно подобрување на проектот, проектот е подготвен за сериско производство и пласман на пазарот.

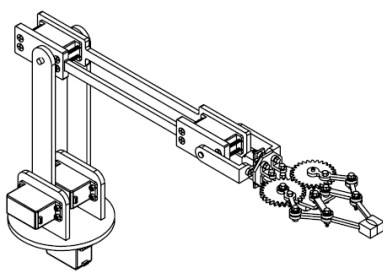
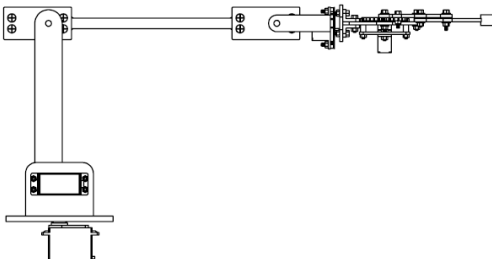
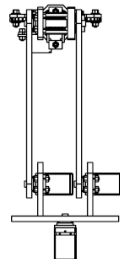
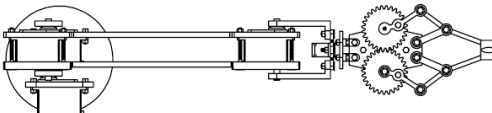
4 ЗАКЛУЧОК

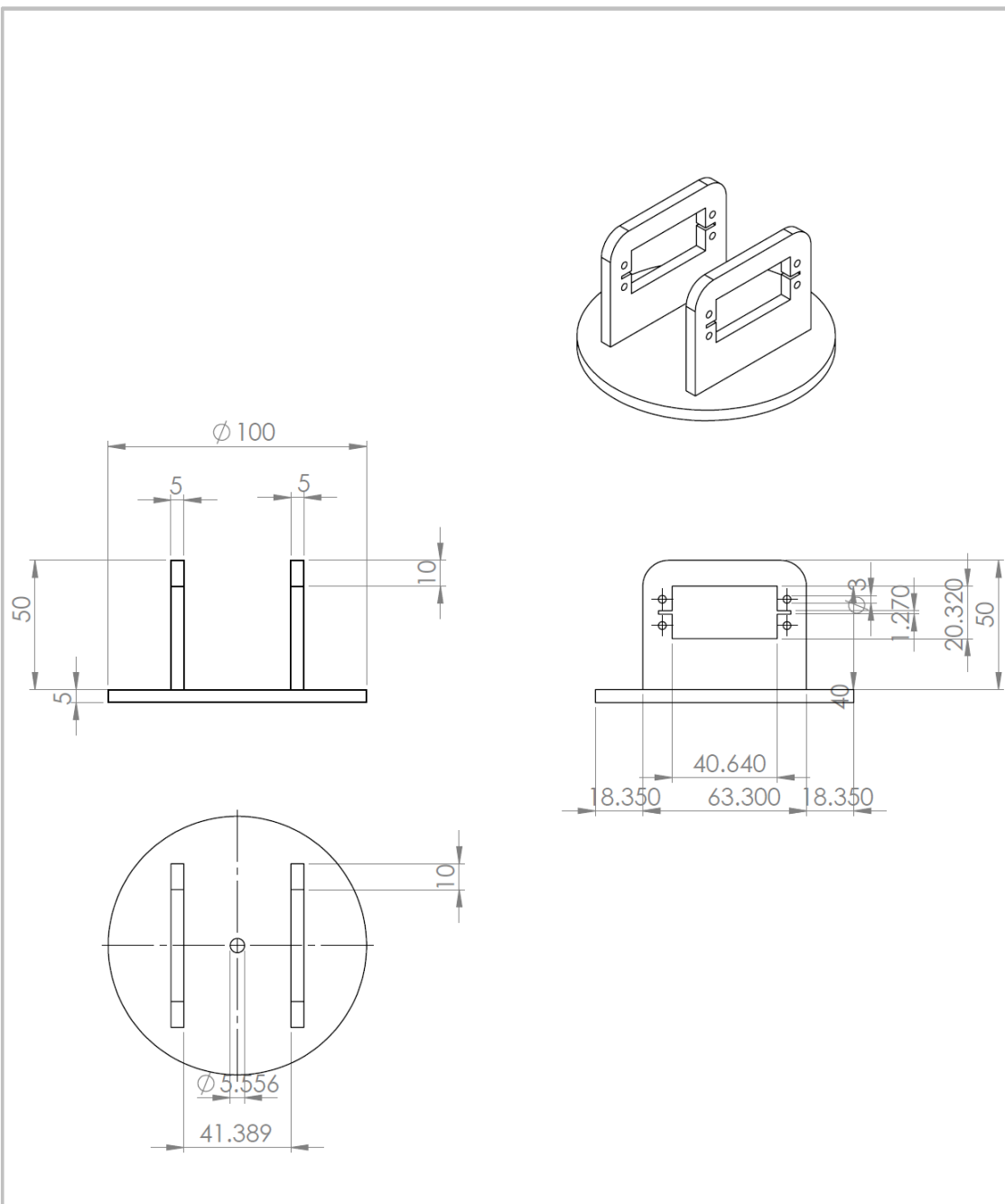
Може да се забележи дека проектирањето на роботски манипулатор претставува сложен процес составен од повеќе фази. За квалитетна изработка на еден проект на роботски манипулатор, неопходна е потребата од многу обемни и сложени пресметки. Тие обемни и сложени пресметки не само што зафаќаат многу време за нивно решавање, туку се и подлежни на грешки при пресметувањето. Еден начин на решавање на овој проблем е користењето на CAD софтверски алатки во процесот на проектирање.

Во овој труд беше претставено проектирањето на едноставен роботски манипулатор со користење на еден таков CAD компјутерски софтвер за проектирање и симулација на роботски манипулатори. Користена е софтверската алатка SolidWorks за изработка на механичката структура на манипулаторот и потоа е искористен SimMechanics алатката од Simulink на Matlab за проектирање на управувач на манипулаторот и за негова симулација. На овој начин значително се поедноставува процесот на проектирање на манипулатор. Освен тоа многу од пресметките компјутерот ги извршува автоматски со што многу се заштедува на време и се елиминира можноста за грешки во пресметувањето.

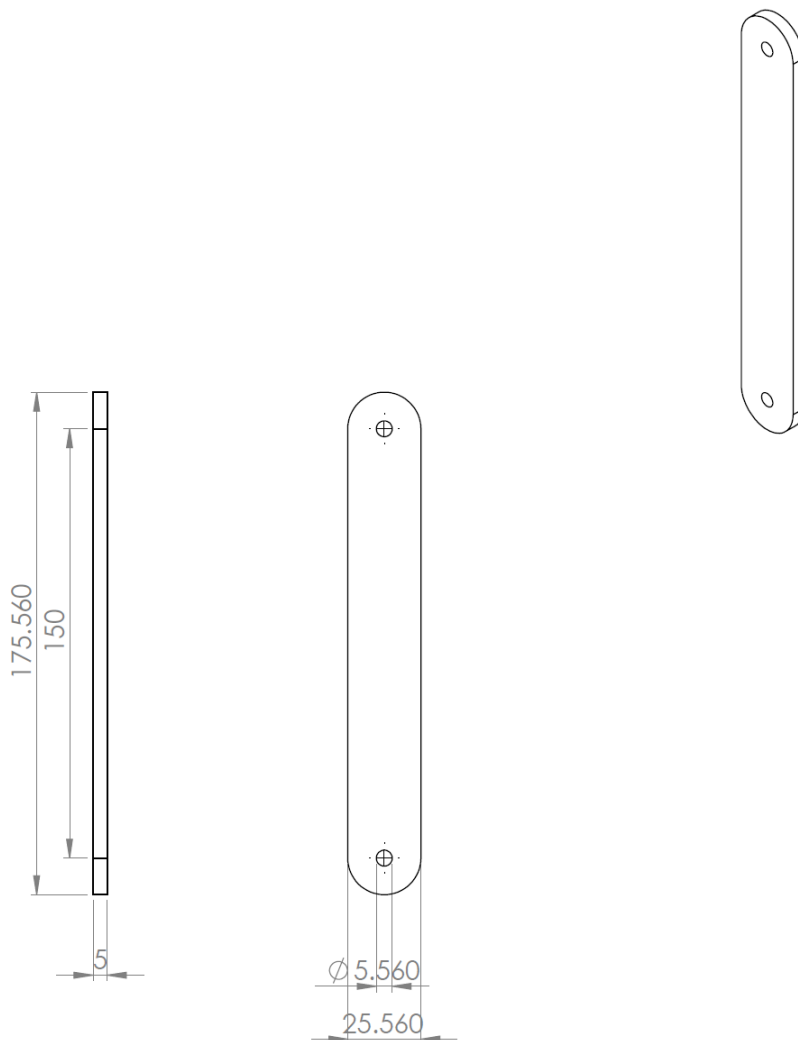
ПРИЛОГ А

Во овој прилог се дадени сите информации за составните делови на манипулаторот. Секоја компанија за изработка на делови би требало да биде во состојба да ги изработи составните делови врз основа информациите претставени на овие скици.

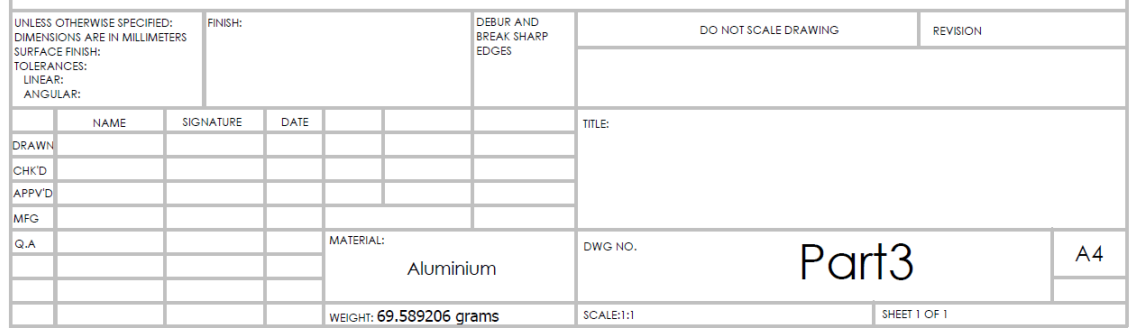
| | | | |
|---|------|----------------------------|-----------------------------------|
|  | | | |
|  | | | |
|  | | | |
|  | | | |
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR: | | FINISH: | DEBUR AND BREAK SHARP EDGES |
| | | DO NOT SCALE DRAWING | REVISION |
| DRAWN | NAME | SIGNATURE | DATE |
| CHK'D | | | |
| APP'VD | | | |
| MFG | | | |
| Q.A. | | | |
| | | MATERIAL: Aluminium | DWG NO. AssemSM2G |
| | | WEIGHT: 964.0023 grams | SCALE:1:1 |
| | | SHEET 1 OF 1 | |
| | | A4 | |

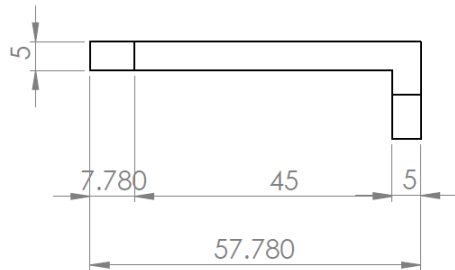
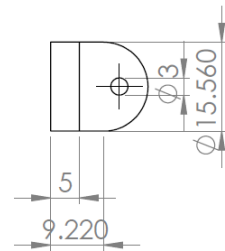
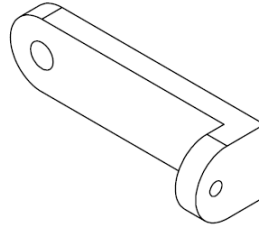
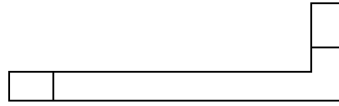


| | | | | | | | | | | | |
|--|--|-----------|--|---------|--|-----------------------------------|--|----------------------|--|--------------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| SURFACE FINISH: | | | | | | | | | | | |
| TOLERANCES: | | | | | | | | | | | |
| LINEAR: | | | | | | | | | | | |
| ANGULAR: | | | | | | | | | | | |
| NAME | | SIGNATURE | | DATE | | | | TITLE: | | | |
| DRAWN | | | | | | | | | | | |
| CHK'D | | | | | | | | | | | |
| APPV'D | | | | | | | | | | | |
| MFG | | | | | | | | | | | |
| Q.A | | | | | | | | | | | |
| | | | | | | MATERIAL: | | DWG NO. | | Part1 | |
| | | | | | | Aluminium | | | | A4 | |
| | | | | | | WEIGHT: 166.574 grams | | SCALE: 1:1 | | SHEET 1 OF 1 | |

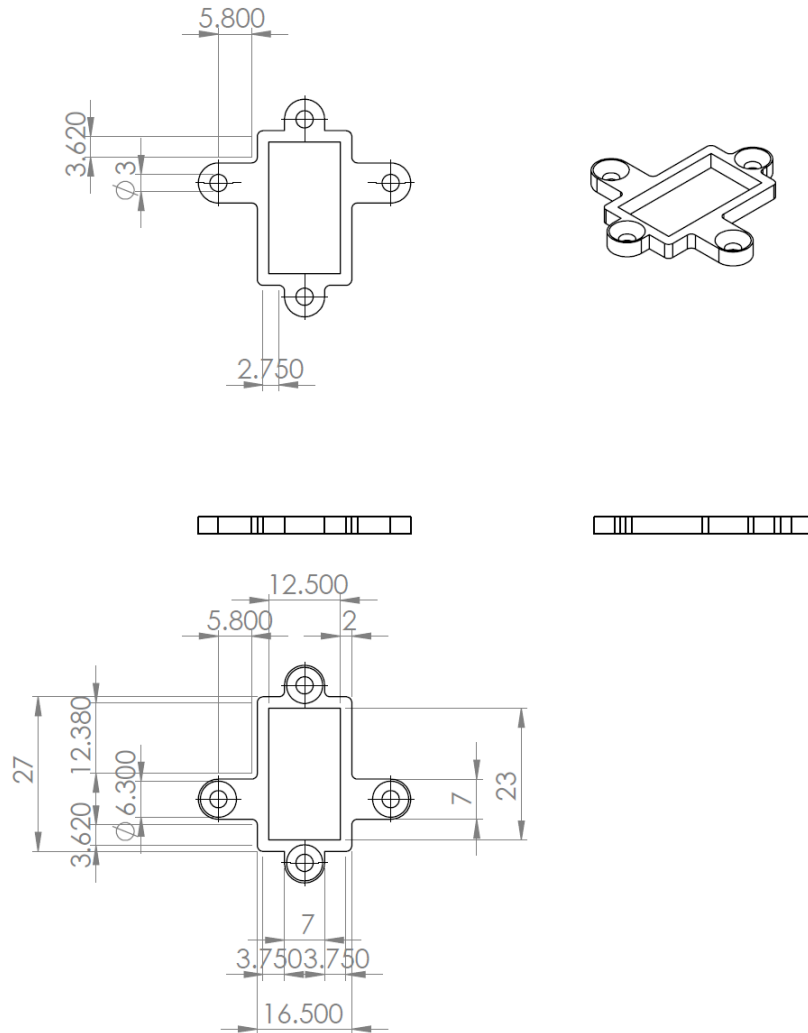


| | | | | | | | | | | | |
|--|--|-----------|--|---------|--|-----------------------------------|--|----------------------|--|--------------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| TOLERANCES: LINEAR: ANGULAR: | | | | | | | | | | | |
| NAME | | SIGNATURE | | DATE | | | | TITLE: | | | |
| DRAWN | | | | | | | | | | | |
| CHK'D | | | | | | | | | | | |
| APPV'D | | | | | | | | | | | |
| MFG | | | | | | | | | | | |
| Q.A | | | | | | MATERIAL: | | DWG NO. | | A4 | |
| | | | | | | Aluminium | | Part2 | | | |
| | | | | | | WEIGHT: 58.030452 grams | | SCALE:1:2 | | SHEET 1 OF 1 | |

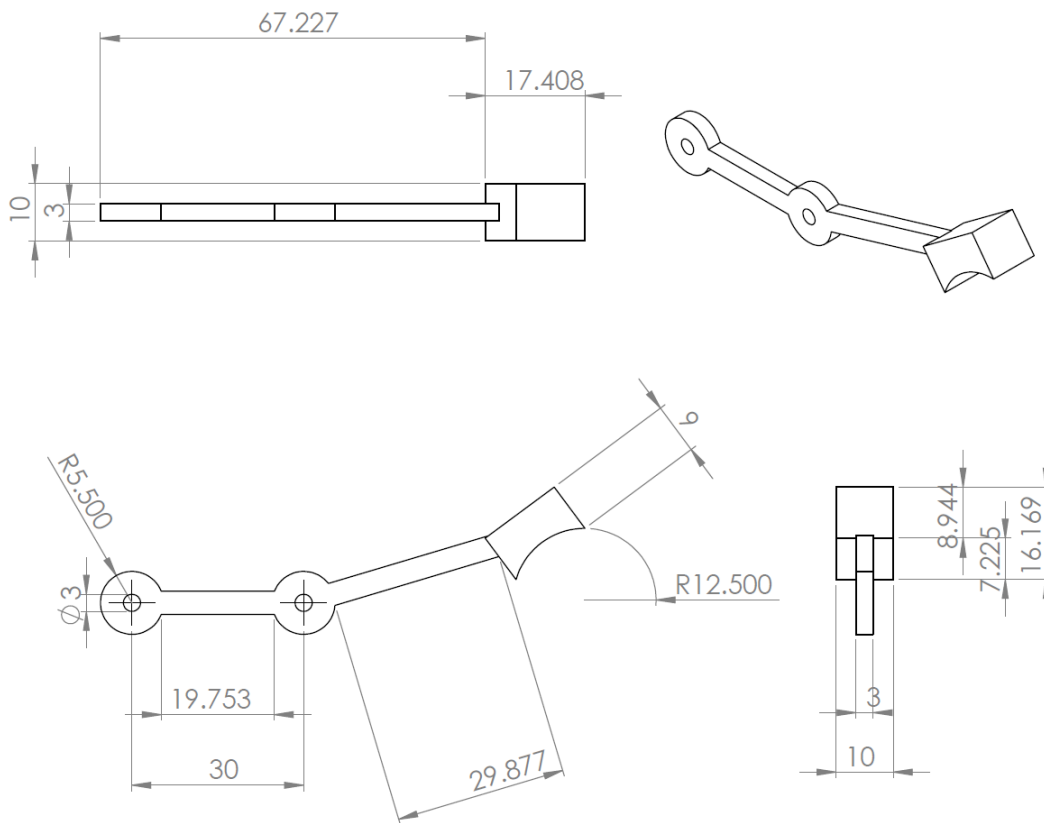




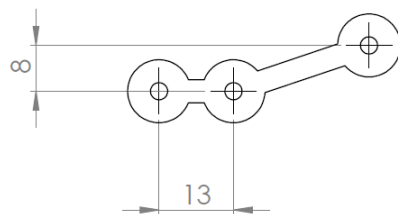
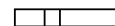
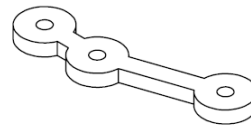
| | | | | | | | | | | | |
|--|--|-----------|--|-------------------------|--|-----------------------------------|--|----------------------|--|----------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| SURFACE FINISH: | | | | | | | | | | | |
| TOLERANCES: | | | | | | | | | | | |
| LINEAR: | | | | | | | | | | | |
| ANGULAR: | | | | | | | | | | | |
| NAME | | SIGNATURE | | DATE | | | | TITLE: | | | |
| DRAWN | | | | | | | | | | | |
| CHK'D | | | | | | | | | | | |
| APPVD | | | | | | | | | | | |
| MFG | | | | | | | | | | | |
| Q.A | | | | | | | | | | | |
| | | | | MATERIAL: | | | | DWG NO. | | | |
| | | | | Aluminium | | | | part5 | | | |
| | | | | WEIGHT: 13.533345 grams | | | | SCALE: 1:1 | | | |
| | | | | | | | | SHEET 1 OF 1 | | | |
| | | | | | | | | A4 | | | |

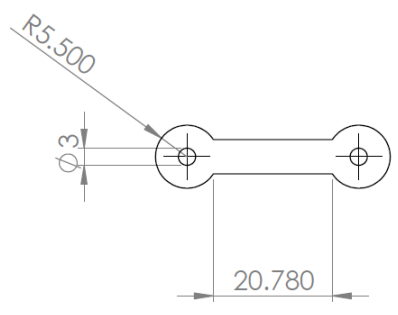
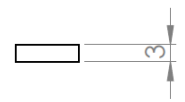
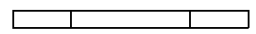
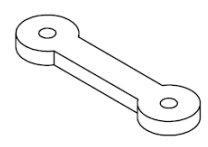
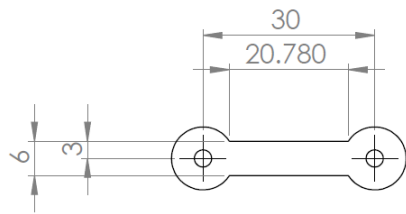


| | | | | | | | | | | | |
|--|--|-----------|--|---------|--|-----------------------------------|--|----------------------|--|--------------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| TOLERANCES: LINEAR: ANGULAR: | | | | | | | | | | | |
| NAME | | SIGNATURE | | DATE | | | | TITLE: | | | |
| DRAWN | | | | | | | | | | | |
| CHK'D | | | | | | | | | | | |
| APPV'D | | | | | | | | | | | |
| MFG | | | | | | | | | | | |
| Q.A | | | | | | MATERIAL: | | DWG NO. | | A4 | |
| | | | | | | Aluminium | | part6 | | | |
| | | | | | | WEIGHT: 2.486268 grams | | SCALE:1:1 | | SHEET 1 OF 1 | |

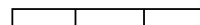
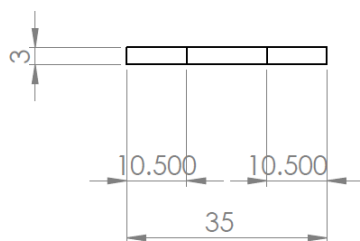
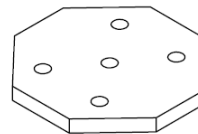
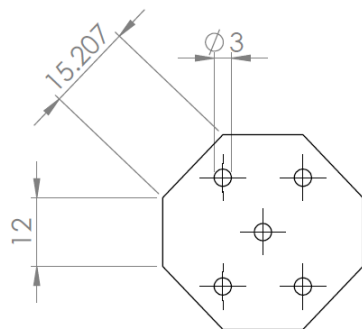


| | | | | | | | | | | | |
|--|--|-----------|--|---------|--|-----------------------------------|--|----------------------|--|--------------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| TOLERANCES: LINEAR: ANGULAR: | | | | | | | | | | | |
| NAME | | SIGNATURE | | DATE | | | | TITLE: | | | |
| DRAWN | | | | | | | | | | | |
| CHK'D | | | | | | | | | | | |
| APPVD | | | | | | | | | | | |
| MFG | | | | | | | | | | | |
| Q.A | | | | | | MATERIAL: | | DWG NO. | | A4 | |
| | | | | | | Aluminium | | Part8 | | | |
| | | | | | | WEIGHT: 5.940637 grams | | SCALE:1:1 | | SHEET 1 OF 1 | |

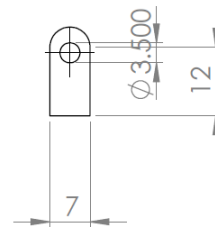
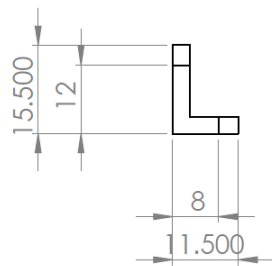
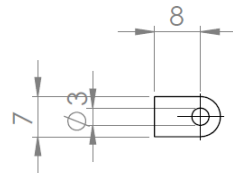




| | | | | | | | | | | | |
|---|--|-----------|--|-----------------------|--|-----------------------------------|--|----------------------|--|----------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR: | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| | | | | | | | | | | | |
| NAME | | SIGNATURE | | DATE | | | | TITLE: | | | |
| DRAWN | | | | | | | | | | | |
| CHK'D | | | | | | | | | | | |
| APPV'D | | | | | | | | | | | |
| MFG | | | | | | | | | | | |
| Q.A | | | | | | | | | | | |
| | | | | MATERIAL: | | | | DWG NO. | | | |
| | | | | Aluminium | | | | Part10 | | | |
| | | | | WEIGHT: 2.37627 grams | | | | SCALE:1:1 | | | |
| | | | | | | | | SHEET 1 OF 1 | | | |
| | | | | | | | | A4 | | | |



| | | | | | | | | | | | |
|---|--|--|--|---------|--|-----------------------------------|--|----------------------|--|----------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR: | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| | | | | | | | | | | | |
| | | | | | | | | TITLE: | | | |
| | | | | | | | | DWG NO. | | | |
| | | | | | | | | Part11 | | | |
| | | | | | | | | A4 | | | |
| | | | | | | | | SCALE:1:1 | | | |
| | | | | | | | | SHEET 1 OF 1 | | | |



| | | | | | | | | | | | |
|---|--|------|-----------|---------|--|-----------------------------------|--|-----------------------|--|--------------|--|
| UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR: | | | | FINISH: | | DEBUR AND BREAK SHARP EDGES | | DO NOT SCALE DRAWING | | REVISION | |
| | | | | | | | | | | | |
| DRAWN | | NAME | SIGNATURE | DATE | | | | TITLE: | | | |
| CHK'D | | | | | | | | | | | |
| APPV'D | | | | | | | | | | | |
| MFG | | | | | | | | | | | |
| Q.A | | | | | | | | | | | |
| | | | | | | MATERIAL: | | DWG NO. Part12 | | | |
| | | | | | | Aluminium | | | | | |
| | | | | | | WEIGHT: 1.140426 grams | | SCALE:1:1 | | SHEET 1 OF 1 | |
| | | | | | | | | | | A4 | |

ПРИЛОГ Б

Во овој прилог се дадени функциите од моделот на роботски манипулатор.

Matlab function block

```
function qp = fcn(t)
%#Glavnata programa
    coder.extrinsic('GlavnaPrograma');
    qp = zeros(1, 6);
    qp = GlavnaPrograma( t );
end

function [dxyz, ddxyz] = GenerirajTraektorija_PrviSegment( xyz1, xyz2, xyz3,
td1, td2 )
% Ovde se presmetuvaat parametrite na traektorijata samo za prviot
% segment ako ni se dadeni pocetnata pozicija, prvata megjutocka,
% vtorata megjutocka, posakuvanoto vremetraenje na prviot segment i
% posakuvanoto vremetraenje vtoriot segment.
a = 1000;

[ ddxyz1 ] = Presmetaj_ddxyz1( xyz1, xyz2, a );
[ t1 ] = Presmetaj_t1( xyz1, xyz2, td1, a );
[ dxyz1 ] = Presmetaj_dxyz1( xyz1, xyz2, td1, t1 );
[ dxyz2 ] = Presmetaj_dxyz2( xyz2, xyz3, td2 );
[ ddxyz2 ] = Presmetaj_ddxyz1( dxyz1, dxyz2, a );

ddxyz = [ddxyz1; ddxyz2];
dxyz_p = [0 0 0];
dxyz_k = (dxyz2 - dxyz1)/2;
dxyz_sr = Novo_dxyz(xyz2 - xyz1, dxyz_p, dxyz_k, ddxyz1, ddxyz2, td1);
dxyz = [dxyz_p; dxyz_sr; dxyz_k];
end

function [ ddxyz ] = Presmetaj_ddxyz1( xyz1, xyz2, a )
% Ovde se presmetuva zabrzuvanjeto ddxyz zaedno so negoviot predznak vo
% prvoto zakrivuvanje na prviot segment pri pocetna brzina ednakva na 0
% amplitudata na zabrzuvanjeto e poznata i e ednakva na a. Istate
% funkcija se koristi i pri presmetkata na zabrzuvanjeto vo vtoroto
% zakrivuvanje od segmentot samo sto namesto poziciite vo vlezot na
% funkcijata figuriraat brzini.
a1 = a; a2 = a; a3 = a;
ddxyz1 = sign(xyz2(1)-xyz1(1))*a1;
ddxyz2 = sign(xyz2(2)-xyz1(2))*a2;
ddxyz3 = sign(xyz2(3)-xyz1(3))*a3;
ddxyz = [ddxyz1 ddxyz2 ddxyz3];
end

function [ t1 ] = Presmetaj_t1( xyz1, xyz2, td1, a )
% Odve se presmetuva vremetraenjeteto na prvoto zakrivuvanje t na prviot
% segment sto zapocnuva so pocetna brzina 0.
% zabrzuvanjeto e konstantno i ednakvo na a
t1_1 = td1 - sqrt(td1*td1 - abs(2*(xyz2(1)-xyz1(1))/a));
t1_2 = td1 - sqrt(td1*td1 - abs(2*(xyz2(2)-xyz1(2))/a));
t1_3 = td1 - sqrt(td1*td1 - abs(2*(xyz2(3)-xyz1(3))/a));
t1 = [t1_1 t1_2 t1_3];
end
```



```

function [ dxyz1 ] = Presmetaj_dxyz1( xyz1, xyz2, td12, t1 )
%   Ovde se presmetuva brzinata dxyz1 na linearniot del od prvot segment.
dxyz1_1 = (xyz2(1)-xyz1(1))/(td12-(1/2)*t1(1));
dxyz1_2 = (xyz2(2)-xyz1(2))/(td12-(1/2)*t1(2));
dxyz1_3 = (xyz2(3)-xyz1(3))/(td12-(1/2)*t1(3));
dxyz1 = [dxyz1_1 dxyz1_2 dxyz1_3];
end

```

```

function [ dxyz ] = Presmetaj_dxyz2( xyz2, xyz3, td23 )
%   Ovde se presmetuva vrednosta na brzinata vo linearniot del na
%   segment pomegju dve megjutocki.
dxyz1 = (xyz3(1) - xyz2(1))/td23;
dxyz2 = (xyz3(2) - xyz2(2))/td23;
dxyz3 = (xyz3(3) - xyz2(3))/td23;
dxyz = [dxyz1 dxyz2 dxyz3];
end

```

```

function [dxyz, ddxyz] = GenerirajTraektorija_VtorSegment( xyz1, xyz2,
xyz3, dxyz_p, td1, td2 )
%   Ovde se presmetuvaat parametrite na traektorijata za segmentot pomegju
%   dve megjutocki, osven pretposledniot segment.
%   Vleznite parametri na finkcijata se:
%   xyz1 = [x1, y1, z1] pocetna pozicija na segmentot
%   xyz2 = [x2, y2, z2] krajna pozicija na segmentot
%   xyz3 = [x3, y3, z3] e krajna pozicija na sledniot segment
%   dxyz1 = [dx1, dy1, dz1] e pocetnata brzina na pocetokot na prvotot
%   zakrivuvanje
%   td1 e vremetraenjeto na celiot segment
%   td2 e vremetraenjeto na sledniot segment
a = 1000;

dxyz_ls = Presmetaj_dxyz2( xyz1, xyz2, td1 );
dxyz_ls2 = Presmetaj_dxyz2( xyz2, xyz3, td2 );
dxyz_k = (dxyz_ls + dxyz_ls2)/2;

ddxyz1 = Presmetaj_ddxyz1( dxyz_p, dxyz_ls, a );
ddxyz2 = Presmetaj_ddxyz1( dxyz_ls, dxyz_k, a );

ddxyz = [ddxyz1; ddxyz2];
dxyz_sr = Novo_dxyz((xyz2-xyz1), dxyz_p, dxyz_k, ddxyz1, ddxyz2, td1);
dxyz = [dxyz_p; dxyz_sr; dxyz_k];

end

```

```

function [ dxyz ] = Presmetaj_dxyz2( xyz2, xyz3, td23 )
%   Ovde se presmetuva vrednosta na brzinata vo linearniot del na
%   segment pomegju dve megjutocki.
dxyz1 = (xyz3(1) - xyz2(1))/td23;
dxyz2 = (xyz3(2) - xyz2(2))/td23;
dxyz3 = (xyz3(3) - xyz2(3))/td23;
dxyz = [dxyz1 dxyz2 dxyz3];
end

```

```

function [ dxyz ] = Presmetaj_ddxyz1( dxyz1, dxyz2, a )
% Ovde se presmetuva zabrzuvanje dxyz zaedno so negoviot predznak vo
% zakrivuvanje na segmentot pri pocetna brzina ednakva na dxyz1 i
% krajna brzina ednakva na dxyz2. Amplitudata na zabrzuvanje e poznata
% i e ednakva na a.
a1 = a; a2 = a; a3 = a;
ddxyz1 = sign(dxyz2(1)-dxyz1(1))*a1;
ddxyz2 = sign(dxyz2(2)-dxyz1(2))*a2;
ddxyz3 = sign(dxyz2(3)-dxyz1(3))*a3;
ddxyz = [ddxyz1 ddxyz2 ddxyz3];
end

```

```

function [dxyz, ddxyz] = GenerirajTraektorija_PredPosledenSegment( xyz1,
xyz2, xyz3, dxyz_p, td1, td2 )
% Ovde se presmetuvaat parametrite na traektorijata za pretposledniot
% segment od traektorijata.
% Vleznite parametri na finkcijata se:
% xyz1 = [x1, y1, z1] pocetna pozicija na segmentot
% xyz2 = [x2, y2, z2] krajna pozicija na segmentot
% xyz3 = [x3, y3, z3] e krajna pozicija na sledniot segment
% dxyz1 = [dx1, dy1, dz1] e pocetnata brzina na pocetokot na prvotot
% zakrivuvanje
% td1 e vremetraenje na celiot (pretposledniot) segment
% td2 e vremetraenje na sledniot (posledniot) segment
a = 1000;

t3 = Presmetaj_t3( xyz2, xyz3, td2, a );
dxyz_ls2 = Presmetaj_dxyz3( xyz2, xyz3, td2, t3 );
dxyz_ls1 = Presmetaj_dxyz2( xyz1, xyz2, td1 );
dxyz_k = (dxyz_ls1 + dxyz_ls2)/2;

ddxyz1 = Presmetaj_ddxyz1( dxyz_p, dxyz_ls1, a );
ddxyz2 = Presmetaj_ddxyz1( dxyz_ls1, dxyz_ls2, a );

dxyz_sr = Novo_dxyz ((xyz2-xyz1), dxyz_p, dxyz_k, ddxyz1, ddxyz2, td1);

dxyz = [dxyz_p; dxyz_sr; dxyz_k];
ddxyz = [ddxyz1; ddxyz2];

end

```

```

function [ t ] = Presmetaj_t3( xyz3, xyz4, td23, a )
% Odve se presmetuva vremetraenje na vtoroto zakrivuvanje t na
% posledniot segment sto zavrshva so krajna brzina 0
% zabrzuvanje e konstantno i ednakvo na a
t1 = td23 - sqrt(td23*td23-(2*(xyz4(1) - xyz3(1)))/a);
t2 = td23 - sqrt(td23*td23-(2*(xyz4(2) - xyz3(2)))/a);
t3 = td23 - sqrt(td23*td23-(2*(xyz4(3) - xyz3(3)))/a);
t = [t1 t2 t3];
end

```

```

function [ dxyz ] = Presmetaj_dxyz3( xyz1, xyz2, td, t3 )
% Ovde se presmetuva brzinata vo linearniot del od posledniot segment.
% Posledniot segment sekogas zavrshva so brzina ednakva na nula.
dxyz1 = (xyz2(1) - xyz1(1))/(td - (1/2)*t3(1));

```

```

    dxyz2 = (xyz2(2) - xyz1(2))/(td - (1/2)*t3(2));
    dxyz3 = (xyz2(3) - xyz1(3))/(td - (1/2)*t3(3));
    dxyz = [dxyz1 dxyz2 dxyz3];
end

function [ dxyz ] = Presmetaj_dxyz2( xyz2, xyz3, td23 )
% Ovde se presmetuva vrednosta na brzinata vo linearniot del na
% segment pomegju dve megjutocki.
    dxyz1 = (xyz3(1) - xyz2(1))/td23;
    dxyz2 = (xyz3(2) - xyz2(2))/td23;
    dxyz3 = (xyz3(3) - xyz2(3))/td23;
    dxyz = [dxyz1 dxyz2 dxyz3];
end

function [ dxyz ] = Presmetaj_ddxyz1( xyz1, xyz2, a )
% Ovde se presmetuva zabrzuvanje dxyz zaedno so negoviot predznak vo
% vtoroto zakrivuvanje na posledniot segment pri krajna brzina ednakva na
% 0. Amplitudata na zabrzuvanje e poznata i e ednakva na a
    a1 = a; a2 = a; a3 = a;
    dxyz1 = sign(xyz2(1)-xyz1(1))*a1;
    dxyz2 = sign(xyz2(2)-xyz1(2))*a2;
    dxyz3 = sign(xyz2(3)-xyz1(3))*a3;
    dxyz = [dxyz1 dxyz2 dxyz3];
end

function [dxyz, dxyz] = GenerirajTraektorija_PosledenSegment( xyz1, xyz2,
dxyz_p, td1 )
% Ovde se presmetuvaat parametrite na traektorijata za segmentot pomegju
% poslednata megjutocka i krajnata tocka od traektorijata.
% Vleznite parametri na finkcijata se:
% xyz1 = [x1, y1, z1] pocetna pozicija na segmentot
% xyz2 = [x2, y2, z2] krajna pozicija na segmentot
% dxyz1 = [dx1, dy1, dz1] e pocetnata brzina na pocetokot na prvotot
% zakrivuvanje
% td1 e vremetraenieto na celiot (posledniot) segment
    a = 1000;

    t3 = Presmetaj_t3( xyz1, xyz2, td1, a );
    dxyz_ls1 = Presmetaj_dxyz3( xyz1, xyz2, td1, t3 );
    dxyz_k = [0 0 0];

    dxyz1 = Presmetaj_ddxyz1( dxyz_p, dxyz_ls1, a );
    dxyz2 = Presmetaj_ddxyz1( dxyz_ls1, dxyz_k, a );

    dxyz_sr = Novo_dxyz ((xyz2-xyz1), dxyz_p, dxyz_k, dxyz1, dxyz2, td1);

    dxyz = [dxyz_p; dxyz_sr; dxyz_k];
    dxyz = [dxyz1; dxyz2];
end

function [ t ] = Presmetaj_t3( xyz3, xyz4, td23, a )
% Odve se presmetuva vremetraenieto na vtoroto zakrivuvanje t na
% posledniot segment sto zavrsva so krajna brzina 0
% zabrzuvanje e konstantno i ednakvo na a
    t1 = td23 - sqrt(td23*td23-(2*(xyz4(1) - xyz3(1)))/a);
    t2 = td23 - sqrt(td23*td23-(2*(xyz4(2) - xyz3(2)))/a);

```

```

t3 = td23 - sqrt(td23*td23-(2*(xyz4(3) - xyz3(3)))/a);
t = [t1 t2 t3];
end

function [ dxyz ] = Presmetaj_dxyz3( xyz1, xyz2, td, t3 )
% Ovde se presmetuva brzinata vo linearniot del na posledniot segment.
% Posledniot segment zavrzuva so brzina ednakva na 0.
dxyz1 = (xyz2(1) - xyz1(1))/(td - (1/2)*t3(1));
dxyz2 = (xyz2(2) - xyz1(2))/(td - (1/2)*t3(2));
dxyz3 = (xyz2(3) - xyz1(3))/(td - (1/2)*t3(3));
dxyz = [dxyz1 dxyz2 dxyz3];
end

function [ ddxz ] = Presmetaj_ddxz1( xyz1, xyz2, a )
% Ovde se presmetuva zabrzuvanje ddxz zaedno so negoviot predznak vo
% vtoroto zakrivuvanje na posledniot segment pri krajna brzina ednakva na
% 0. Amplitudata na zabrzuvanje e poznata i e ednakva na a
a1 = a; a2 = a; a3 = a;
ddxz1 = sign(xyz2(1)-xyz1(1))*a1;
ddxz2 = sign(xyz2(2)-xyz1(2))*a2;
ddxz3 = sign(xyz2(3)-xyz1(3))*a3;
ddxz = [ddxz1 ddxz2 ddxz3];
end

function [ out ] = Forward( xyz, dxyz, ddxz, td, t )
%Ovde se presmetuva pozicijata po site tri koordinati.
% Vleznite promenlivi vo funkciojata se:
% xyz=[x;y;z] - pocetna pozicija na segmentot
% dxyz=[dx; dy; dz], kade sto dx=[dx1, dx2, dx3];
% - dx1 e brzinata na pocetotkot na prvoto zakrivuvanje,
% - dx2 e brzinata vo linearniot del na segmentot,
% - dx3 brzinata na krajot na vtoroto zaktivavanje
% ddxz=[ddxz1; ddxz2], kade sto ddxz1=[ddx1; ddx2]
% - ddx1 zabrzuvanje vo prvoto zakrivuvanje
% - ddx2 e zabrzuvanje vo vtoroto zakrivuvanje
% td e vkupnoto vremetraenje na segmentot
% t e vremenskiot moment za koj se presmetuva pozicijata.
out1 = ForwardX( xyz(1), dxyz(:, 1), ddxz(:, 1), td, t );
out2 = ForwardX( xyz(2), dxyz(:, 2), ddxz(:, 2), td, t );
out3 = ForwardX( xyz(3), dxyz(:, 3), ddxz(:, 3), td, t );
out = [out1 out2 out3];
end

function [ out ] = ForwardX( x, dx, ddx, td, t )
% Ovde se presmetuva polozbata vo momentot t po samo 1 koordinata
% Vleznite promenlivi vo funkciojata se:
% x - pocetna pozicija na segmentot
% dx=[dx1; dx2; dx3]
% - dx1 e brzinata na pocetotkot na prvoto zakrivuvanje,
% - dx2 e brzinata vo linearniot del na segmentot,
% - dx3 brzinata na krajot na vtoroto zaktivavanje
% ddx=[ddx1; ddx2]
% - ddx1 zabrzuvanje vo prvoto zakrivuvanje
% - ddx2 e zabrzuvanje vo vtoroto zakrivuvanje
% td e vkupnoto vremetraenje na segmentot
% t e vremenskiot moment za koj se presmetuva pozicijata.

```

```

if ddx(1) == 0
    if ddx(2) == 0
        out = x(1) + dx(2)*t;
    else
        t2 = (dx(3)-dx(2))/ddx(2);
        t3 = td - t2;
        if t < t3
            out = x(1) + dx(2)*t;
        else
            out1 = x(1) + dx(2)*t3;
            out = out1 + dx(2)*(t-t3) + ddx(2)*(t-t3)*(t-t3)/2;
        end
    end
elseif ddx(2) == 0
    t1 = (dx(2)-dx(1))/ddx(1);
    if t < t1
        out = x(1) + dx(1)*t + ddx(1)*t*t/2;
    else
        out1 = x(1) + dx(1)*t1 + ddx(1)*t1*t1/2;
        out = out1 + dx(2)*(t-t1);
    end
else
    t1 = (dx(2)-dx(1))/ddx(1);
% t1 e vremetraenje na prvoto zakrivuvanje od segmentot
    t2 = (dx(3)-dx(2))/ddx(2);
% t2 e vremetraenje na vtoroto zakrivuvanje od segmentot
    if t < t1
        out = x(1) + dx(1)*t + ddx(1)*t*t/2;
    elseif t < td - t2
        out1 = x(1) + dx(1)*t1 + ddx(1)*t1*t1/2;
        out = out1 + dx(2)*(t-t1);
    else
        out1 = x(1) + dx(1)*t1 + ddx(1)*t1*t1/2;
        out2 = out1 + dx(2)*(td-t1-t2);
        t3 = t1 + (td - t1 - t2);
        out = out2 + dx(2)*(t - t3) + ddx(2)*(t-t3)*(t-t3)/2;
    end
end
end
end

% Ova e idealnata funkcija za presmetka na inverznata kinematika
% rabotniot prostor e svera so radiusi 220 i 510. nadvor od tie radiusi
% robotot ne e vo sostojba da dofati i resenieto na inverznata kinematika
% ke bide pogresno.

function angles = invkine(xyz)
% Ova e funkcijata so koja se presmetuva inverznata kinematika na
% robotskata raka.
% theta230 = [theta20 theta30]
% xyz = [x y z]
% b = [x z theta1 theta4]
theta230=[0 0];
% xyz = [20 20 440];
theta4 = (pi/2) * (1 -
((xyz(1)*xyz(1)+xyz(2)*xyz(2)+xyz(3)*xyz(3))/(580*580)));

if xyz(1)==0
    if xyz(2) > 0

```

```

        theta1 = pi/2;
    elseif xyz(2) < 0
        theta1 = -pi/2;
    else
        % disp("nezavisno od prvata koordinata")
        theta1 = 0;
    end
else
    theta1 = atan(xyz(2)/xyz(1));
end

b = [xyz(1) xyz(3) theta1 theta4];

options = optimoptions('fsolve','Display','none');
f = @(theta230) myfun_new(theta230, b);
jointangles = fsolve(f, theta230, options);

angles = [theta1 jointangles theta4 0]*180/pi;
angles = poednostavi(angles);
end

function F = myfun(theta230, b)
% Ova e funkcijata so koja se presmetuvaat aglitate na vtoriot i tretiot
% zglob na robotot vo zavisnost od pozicijata na krajniot izvršen element.
% Pri toa se koristi iterativna postapka pri koja theta0 = [theta(2)
theta(3)]
% dodeka vlezot vo funkcijata e b = [x, z, theta(1) theta(4)]
theta = [b(3) theta230(1) theta230(2) b(4)];
x = b(1);
z = b(2);

F = [(3555*cos(theta(2)+theta(3)) - 3297*sin(theta(2)+theta(3)+theta(4)) +
2500*cos(theta(2)) - 50*x/(3*cos(theta(1)))));
(9891*cos(theta(2)+theta(3)+theta(4)) + 10665*sin(theta(2)+theta(3)) +
7500*sin(theta(2)) - 50*z + 1492)];

end

function xyz = pom1(theta)
% Ova e pomosna funkcija so koja se dobivat funkcijate za presmetka na x,
% y i z koordinatite na krajniot izvršen element vo zavisnost od
% vrednostite na aglitate na zglobovite na robotot.
d1 = 29.84; a2 = 150; a3 = 213.3; d5 = 197.82;

L1 = Link([theta(1) d1 0 pi/2]);
L2 = Link([theta(2) 0 a2 0]);
L3 = Link([theta(3) 0 a3 0]);
L4 = Link([theta(4) 0 0 -pi/2]);
L5 = Link([theta(5) d5 0 0]);
R = SerialLink([L1 L2 L3 L4 L5], 'name', 'Robot');

T = R.fkine(theta);

x = T(1, 4);
y = T(2, 4);
z = T(3, 4);
xyz = [x, y, z];

```

```
end
```

```
function theta_iz = poednostavi(theta_vl)
% Ova funkcija vrši poednostavuvanje na matricata od agli. So drugi
% zborovi gi normalizira vrednostite na aglite da bidat vo granicite od
% (-180,180)
```

```
s = size(theta_vl);
n = s(2);
for i=1:n
    if theta_vl(i) > 180
        theta_iz(i) = theta_vl(i) - 360;
    elseif theta_vl(i) < -180
        theta_iz(i) = theta_vl(i) + 360;
    else
        theta_iz(i) = theta_vl(i);
    end
end
end
```

```
function F = myfun_new(theta230, b)
% Ova e funkcijata so koja se presmetuvaat aglite na vtoriot i tretiot
% zglob na robotot vo zavisnost od pozicijata na krajniot izvršen element.
% Pri toa se koristi iterativna postapka pri koja theta230 = [theta(2)
theta(3)]
% dodeka vlezot vo funkcijata e b = [x, z, theta(1) theta(4)]
theta = [b(3) theta230(1) theta230(2) b(4)];
x = b(1)/cos(theta(1));
z = b(2);
```

```
F = [(150*sin(theta(2)) + 213.3*sin(-theta(2)-theta(3)+(pi/2)) +
197.82*sin(-theta(2)-theta(3)-theta(4)+(pi/2)) - x);
(150*cos(theta(2)) - 213.3*cos(-theta(2)-theta(3)+(pi/2)) -
197.82*cos(-theta(2)-theta(3)-theta(4)+(pi/2)) - z)];
```

```
end
```

```
function [ dxyz ] = Novo_dxyz( delta_xyz, dxyz1, dxyz2, ddxyz1, ddxyz2, td
)
% Ovde se presmetuva posakuvanata vrednost na brzinata vo linearniot del
% na segmentot taka sto toj segmentot zapocnuva so točno opredeleni
% brzina dxyz1 i zabrzuvanje ddxyz1 i završuva so točno opredeleni brzina
% dxyz2 i zabrzuvanje ddxyz2. Vremetraenijeto na segmentot e td, a
% pominatoto rastojanje e delta_xyz.
tmp1 = Presmetaj_dxyz_sr( delta_xyz(1), dxyz1(1), dxyz2(1), ddxyz1(1),
ddxyz2(1), td );
tmp2 = Presmetaj_dxyz_sr( delta_xyz(2), dxyz1(2), dxyz2(2), ddxyz1(2),
ddxyz2(2), td );
tmp3 = Presmetaj_dxyz_sr( delta_xyz(3), dxyz1(3), dxyz2(3), ddxyz1(3),
ddxyz2(3), td );
dxyz = [tmp1, tmp2, tmp3];
end
```

```
function [ dxyz_sr ] = Presmetaj_dxyz_sr( delta_xyz, dxyz1, dxyz2, ddxyz1,
ddxyz2, td )
% Ovde se presmetuva posakuvanata vrednost na brzinata vo linearniot del
```

```

% na segmentot taka sto toj segmentot zapocnuva so točno opredeleni
% brzina dxyz1 i zabrzuvanje ddxyz1 i završuva so točno opredeleni brzina
% dxyz2 i zabrzuvanje ddxyz2. Vremetraenje na segmentot e td, a
% pominatoto rastojanje e delta_xyz.
% OVDE MORA DA SE NAPOMENE DEKA FUNKCIJATA E SAMO ZA EDNA KOORDINATA I
% DOKOLKU SAKAME DA RABOTIME SO POVEKJE OD EDNA KOORDINATA FUNKCIJATA
% MORA DA SE POVIKUVA POODDELNO ZA SEKOJA KOORDINATA.
if ddxyz1 == 0
    dxyz_sr = dxyz1;
elseif ddxyz2 == 0
    dxyz_sr = dxyz2;

elseif ddxyz1 == ddxyz2
%    disp('A = 0')
    B = td + (dxyz1/ddxyz1) - (dxyz2/ddxyz2);
    C = ((dxyz2*dxyz2)/(2*ddxyz2)) - ((dxyz1*dxyz1)/(2*ddxyz1)) -
delta_xyz;
    dxyz_sr = -C/B;
else
    A = ((1/ddxyz2) - (1/ddxyz1))/2;
    B = td + (dxyz1/ddxyz1) - (dxyz2/ddxyz2);
    C = ((dxyz2*dxyz2)/(2*ddxyz2)) - ((dxyz1*dxyz1)/(2*ddxyz1)) -
delta_xyz;
    if B == 0;
        dxyz_sr = sqrt(-C/A);
    elseif B > 0
%        disp('B > 0');
        dxyz_sr = (-B + sqrt(B*B-4*A*C))/(2*A);
    else
%        disp('B < 0');
        dxyz_sr = (-B - sqrt(B*B-4*A*C))/(2*A);
    end
end
end

function [ td ] = PosakuvaniVremetraenje( xyz )
% Ovde se presmetuva posakuvano vremetraenje td na daden segment kako
% rastojanje pomegu tockite pomnozeno so odredena konstanta. Taa
% konstanta ja definira brzinata na dvizenje na robotot.

koef = 1/100;
s = size(xyz);
td = [];
for i = 1:(s(1)-1);
    td = [td PosakuvanoVremetraenje(xyz(i, :), xyz(i+1, :))];
end
td = td*koef;
end

function [ td ] = PosakuvanoVremetraenje( xyzi, xyzj )
% Ovde se presmetuva rastojanieto pomegu tockite xyzi i xyzj
td = sqrt(((xyzi(1)-xyzj(1))^2) + ((xyzi(2)-xyzj(2))^2) + ((xyzi(3)-
xyzj(3))^2));
end

function [ out ] = Presmetaj_Pozicija( xyz, td, t )
% Ovde se generira traektorija. Taa traektorija pretstavuva isprekrsena

```



```

% prava linija so parabolicni zakrivuvanja na kraevite i na mestata na
% prekršuvanje.
% Vlezni parametri se:
% xyz = [x1, y1, z1; x2, y2, z2; ... ... ..; xn, yn, zn] kade sto x1,
% y1, z1 se koordinati na pocetokot na traektorijata. xi, yi, zi za i=0:n
% se koordinatite na prekršuvanje na traektorijata i xn, yn, zn se
% koordinatite na zavrsetotkot na traektorijata. td se vremetraenjata na
% dvizenjeto po traektorijata za sekoj od segmentite na traektorijata. t
% momentalno vreme. Izlezot na funkcijata se koordinatite x, y i z koi
% ja opredeluvaaat polozbata od traektorijata vo koja se naogjame vo
% mmomentot t.
persistent Seg;
persistent t_PocSeg;
persistent br_Seg;
persistent dxyz;
persistent ddxyz;

    if isempty(Seg)
        Seg = 1;
        [dxyz, ddxyz] = GenerirajTraektorija_PrviSegment( xyz(1, :), xyz(2,
:), xyz(3, :), td(1), td(2) );
        t_PocSeg = 0;
        s = size(xyz);
        br_Seg = s(1)-1;
    elseif Seg == br_Seg+1
        dxyz = [0 0 0; 0 0 0; 0 0 0];
        ddxyz = [0 0 0; 0 0 0];
    elseif t > (t_PocSeg + td(Seg))
        Seg = Seg + 1;
        t_PocSeg = t_PocSeg + td(Seg-1);
        if Seg == 2
            [dxyz, ddxyz] = GenerirajTraektorija_VtorSegment( xyz(2, :),
xyz(3, :), xyz(4, :), dxyz(3, :), td(2), td(3) );
        elseif Seg == (br_Seg - 1)
            [dxyz, ddxyz] = GenerirajTraektorija_PredPosledenSegment(
xyz(Seg, :), xyz(Seg+1, :), xyz(Seg+2, :), dxyz(3, :), td(Seg), td(Seg+1)
);
        elseif Seg == br_Seg
            [dxyz, ddxyz] = GenerirajTraektorija_PosledenSegment( xyz(Seg,
:), xyz(Seg+1, :), dxyz(3, :), td(Seg) );
        elseif and(Seg>2, Seg<(br_Seg-1))
            [dxyz, ddxyz] = GenerirajTraektorija_VtorSegment( xyz(Seg, :),
xyz(Seg+1, :), xyz(Seg+2, :), dxyz(3, :), td(Seg), td(Seg+1) );
        end
    end

    if Seg < br_Seg+1
        out = Forward( xyz(Seg, :), dxyz, ddxyz, td(Seg), t-t_PocSeg );
    else
        out = xyz(br_Seg+1, :);
    end

end

```

КОРИСТЕНА ЛИТЕРАТУРА:

1. <http://www.galileo.org/robotics/design.html>
2. <http://see.stanford.edu/see/courseinfo.aspx?coll=86cc8662-f6e4-43c3-a1be-b30d1d179743>
3. Introduction to Robotics - Analysis, Systems, Applications Saeed B. Niku
4. Control of Robot Manipulators in Joint Space - R. Kelly, V. Santibanez and A. Loria
5. Robotics, Vision & Control: (c) Peter Corke 1992-2011