Софийски университет "Св. Климент Охридски"

Факултет по математика и информатика

Моделиране и управление на антропоморфен модел на робот

ДИСЕРТАЦИЯ

на Любомира Лъчезарова Митева

за присъждане на образователна и научна степен „Доктор" в професионално направление 4.6. Информатика и компютърни науки

Докторска програма:
„Информационни системи" – Вградени и автономни системи

Научни ръководители:
проф. д-р Евгений Кръстев
доц. д-р Иван Чавдаров

гр. София, 2023 г.

Благодарности

Бих искала да изкажа специални благодарности на научните ми ръководители проф. д-р Евгений Кръстев и доц. д-р Иван Чавдаров за градивните съвети, коментари и забележки.

Изказвам благодарности и към членовете на катедри „Компютърна информатика“ и „Мехатроника, роботика и механика“ за съветите и напътствията.

Благодаря и на Институт по роботика – БАН за подкрепата и предоставената възможност за участие в научни проекти.

Съдържание

Списък на фигурите

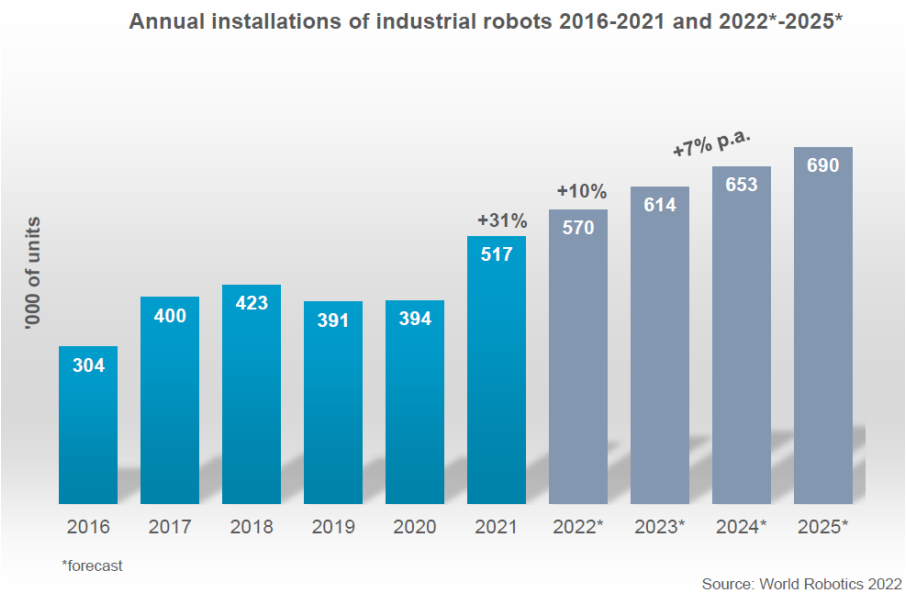
Списък на таблиците

Увод

Хората от векове се опитват да създадат машини, които биха могли да имитират тяхното поведение и да ги отменят в рутинни, повторяеми задачи. За първи път думата робот се появява в литературата през 1920 г., когато писателят Карел Чапек описва в своята пиеса „Универсални роботи на Росум“ (“Rossum’s Universal Robots”) създаването на изкуствени същества, които да заменят хората в повтарящи се дейности за индустриални задачи и са наречени роботи. Всъщност, думата робот произлиза от термина robota, означаващ работа на чешки език [1]. По-късно през 1940 година взаимодействието между роботите и хората е описано от писателя Айзък Азимов в романа му „Runaround“ [2, 3]. Роботът е представен като механична машина с човешки външен вид, но лишен от чувства. Малко по-късно в средата на 20-ти век се появява и първият робот с цифрово управление на компанията Unimate [4].

Според научната интерпретация на научно-фантастичния сценарий, роботът се разглежда като машина, която независимо от външния си вид може да променя средата, в която работи. Това се постига чрез извършване на действия, които са обусловени от определени правила на поведение, присъщи на машината, както и от данните, които роботът получава относно своето състояние и заобикалящата го среда. Роботиката се занимава с изследване и разработване на роботизирани системи, които могат да отменят човека в дейности, изискващи тежка физическа активност, извършване на манипулации в опасна среда или при рутинни и повторяеми задачи [1, 5]. Роботът е дефиниран като автономна механична компютърно управлявана машина с програмируемо от потребителя механично движение на изпълнителното устройство относно неговото позициониране и/или траектория на движение. Именно програмируемостта е една от най-важните характеристики на роботите, тъй като позволява използването на една роботизирана система за изпълнение на различни задачи.

Внедряването на роботизирани системи в работния процес увеличава продуктивността и повишава качеството на извършените задачи. Според данни на международната федерация по роботика, най-много роботи се използват за индустриални приложения, в края на 2021-ва година броят на използваните индустриални роботи е 3 477 127. Броят на продадените индустриални роботи за 2021-ва година е с 31% повече спрямо предходната година. Очакванията са до 2025-та година пазарът да се възстанови от COVID-19 пандемията и продажбите на индустриални роботи да се увеличат и да достигнат рекордните 690 000 броя за една година (Фиг. 0.1) [6].



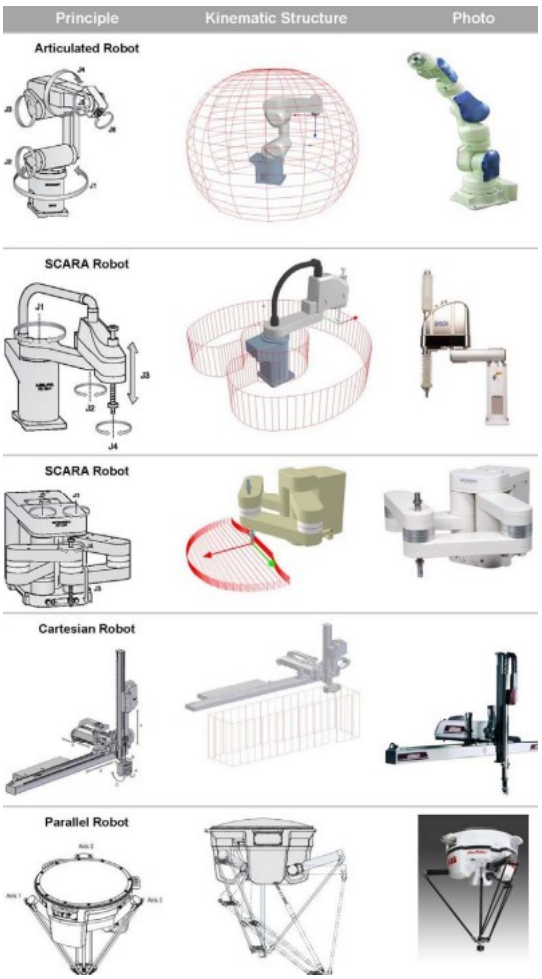
Фиг. 0.1. Продадени и внедрени индустриални роботи в периода 2016-2021 г. и очаквани продажби за 2022-2025 г.

Основните ползи от внедряването на такива системи в промишлеността са: намаляване на производствените разходи, повишаване на прецизността, качеството и производителността. Използването на индустриални роботи в производствения процес помага на компаниите да разработват по-бързо своите продукти и да бъдат конкурентноспособни. Допълнително, роботите допринасят за по-добра устойчивост за справяне с пикове на производството и справяне с непредвидени ситуации като пандемията от COVID-19, както и са способни да изпълняват все повече задачи, които могат да бъдат опасни или би било невъзможно да бъдат извършени от човек (Фиг. 0.2) [6].

Фиг. 0.2. Ползи от внедряване на индустриални роботи [6].

Индустриалните роботи са различни видове в зависимост от механичната им структура (Фиг. 0.3) [7]:

- Артикулационни роботи (Articulated robot) – имат поне три ротационни стави.
- Робот тип SCARA (Selective Compliance Assembly Robot Arm [1] или Selective Compliance Articulated Robot Arm [5]) – обикновено се характеризира с 3 ротационни стави и една транслация. Имат проста конструкция и са подходящи за работа с други роботи.
- Декартов робот (Cartesian robot) – има три основни контролни оси, които са линейни (три транслационни стави) и взаимно перпендикулярни и съответстват на движения по осите на декартова координатна система.
- Паралелни/Делта роботи (Parallel/Delta robot) – роботи с паралелни звена.
- Роботи, използващи цилиндрични координати (Cylindrical robots) – характеризират се с едно въртящо движение към основата и минимум една транслационна става.

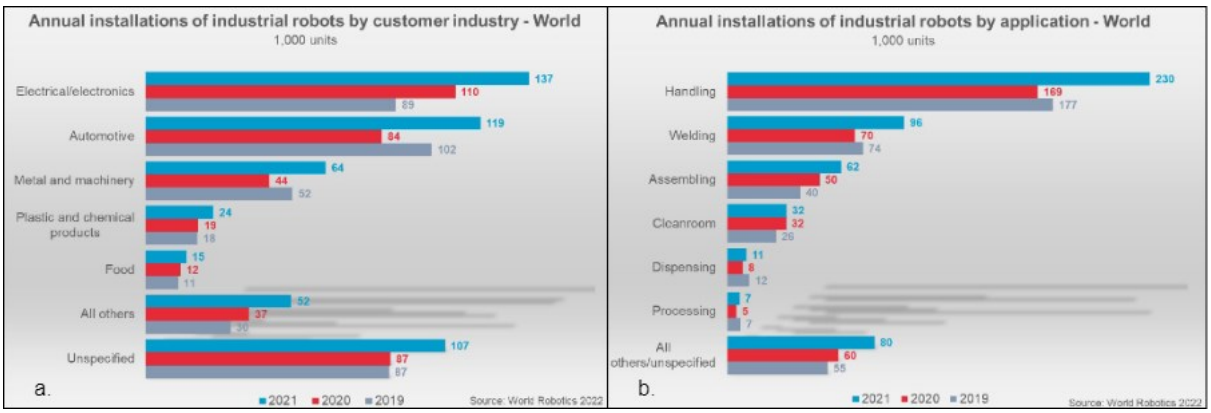


Фиг. 0.3. Индустриални типове роботи в зависимост от механичната им конструкция [7].

В този дисертационен труд обект на изследване ще бъде индустриален робот тип SCARA.

Актуалност на темата

По данни от международната федерация по роботика през 2020-та година електрониката изпреварва автомобилната индустрия по брой внедрени индустриални роботи, като тази тенденция се запазва и през 2021-ва година (Фиг. 0.4.a.). Това е исторически момент, тъй като от 1961 г., когато е продаден първият индустриален робот и използван в завода на General Motors в Ню Джърси, автомобилната индустрия винаги е била най-големият клиент на промишлени роботи. Но през 2020 г. внедрените индустриални роботи в автомобилостроенето са с 26 000 по-малко в сравнение с тези в електрониката [6]. Голяма част от индустриалните роботи използвани в електрониката са именно роботи тип SCARA. Те са сравнително лесни за конструиране с малка маса и подходящи за преместване и монтиране на фини елементи, каквито се използват в електрониката.



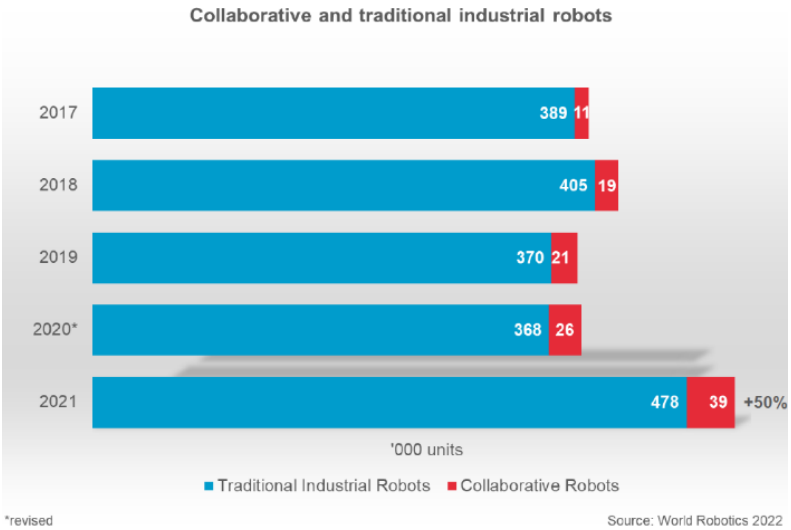
Фиг. 0.4. Използвани индустриални роботи a. в различните индустрии b. за различни индустриални дейности [6].

Индустриалните роботи обикновено изпълняват предварително дефинирани задачи, като едни от основните им дейности са обработка на метали, заваряване, сглобяване, транспортиране, боядисване и др. (Фиг. 0.4.b.). Всеки робот разполага с различен брой независими стави, които еднозначно определят позицията и ориентацията на изпълнителното му звено. Казваме, че броят на ставите на един робот определя броя на степените на свобода на този робот. От роботите се изисква да изпълнят прецизно и точно поставените им задачи, спазвайки определена траектория и съобразявайки се с налични препятствия в работното им пространство. Поради тази причина някои роботи се проектират по такъв начин, че да разполагат с повече степени на свобода, отколкото са необходими за изпълнение на желаната задача. Това ги прави по-гъвкави, по-лесно преодоляват налични препятствия в работната им зона, както и повишава производителността им. Казваме, че един манипулатор е с допълнителни степени на свобода, когато има повече степени на свобода от необходимите за изпълнение на дадена задача [8].

Мотивацията за въвеждане на допълнителни степени на свобода към механичната структура идва от целта да се увеличи надеждността на роботизираната система и да се намали

вероятността за грешка. Създаването на механична конструкция с повече степени на свобода е породено от желанието даден манипулатор да притежава гъвкавостта на човешката ръка. Както знаем, човешката ръка има три степени на свобода в рамото, една степен на свобода в лакътя и три степени на свобода на китката.

Наличието на допълнителни степени на свобода позволява движения на манипулатора, които не изместват хващача от определена позиция. Това означава, че една и съща позиция и ориентация на хващача може да бъде изпълнена с различни ставни конфигурации, което дава възможност за избягване на региони с препятствия и води до по-голяма гъвкавост на манипулатора [9]. Тази характеристика е ключова за използване на роботизирани системи в динамична и неструктурирана (недетерминирана) среда. В такава среда оперират различни индустриални и сервизни роботи и обикновено се наричат колаборативни роботи. Според данни на международната федерация по роботика, броят на използваните колаборативни роботи се е увеличил с 50% през 2021 година и се очаква да расте през следващите години (Фиг. 0.5) [6].



Фиг. 0.5. Брой използвани колаборативни и традиционни индустриални роботи [6].

Роботизираните системи с допълнителни степени на свобода се използват във важни приложения, в които се изисква висока прецизност. Едно от най-известните приложения на тези манипулатори е в Международната космическа станция, където се използва Special Purpose Dexterous Manipulator (SPDM) (известен също като Dexter или накратко Canada Arm 2) (Фиг. 0.6.a.) [10]. Друга област на приложение е медицината, където се изисква висока прецизност и вероятността за грешка трябва да бъде сведена до минимум. Една от най-използваните роботизирани системи в областта на медицината е Da Vinci Surgical System (Фиг. 0.6.b.). Той подпомага медицинските екипи в лапароскопската хирургия. Неговите предимства са прецизност, гъвкавост и добър контрол [11]. Най-много роботи с допълнителни степени на

свобода се използват за индустриални цели, като боядисване, сглобяване, заваряване и др. (Фиг. 0.6.c. [12]). Такъв тип роботи често оперират съвместно с други роботи или хора.

От друга страна, наличието на повече степени на свобода усложнява моделирането на кинематиката, както и управлението на робота [4]. Решението на обратната задача на кинематиката може да се състои от множество от ставни конфигурации. Затова и темата за моделиране и управление на роботизирани системи с допълнителни степени на свобода е една от най-изследваните в областта на роботиката.

Фиг. 0.6. Приложения на роботи с допълнителни степени на свобода: a. Special Purpose Dexterous Manipulator [10]; b. Da Vinci Surgical System [11]; c. Индустриални манипулатори [12].

Моделирането и управлението на роботи са основните етапи от разработването на роботизирани системи. Моделирането се занимава, както с изследване и изготвяне на математическия модел на робота, така и с проектиране на 3D модел с помощта на подходящ за тази цел софтуер и изготвяне на реален модел на желания робот. Моделирането и управлението зависят от вида на робота, неговото приложение и конструктивни ограничения. Целта на управлението е задвижващите устройства да получат подходящи входни параметри и роботът да изпълни успешно и ефективно желаната задача. За тази цел е необходимо да се изберат подходящи хардуерни компоненти, както и да се изследват и създадат методи за планиране на траектория в среда с налични статични и динамични препятствия.

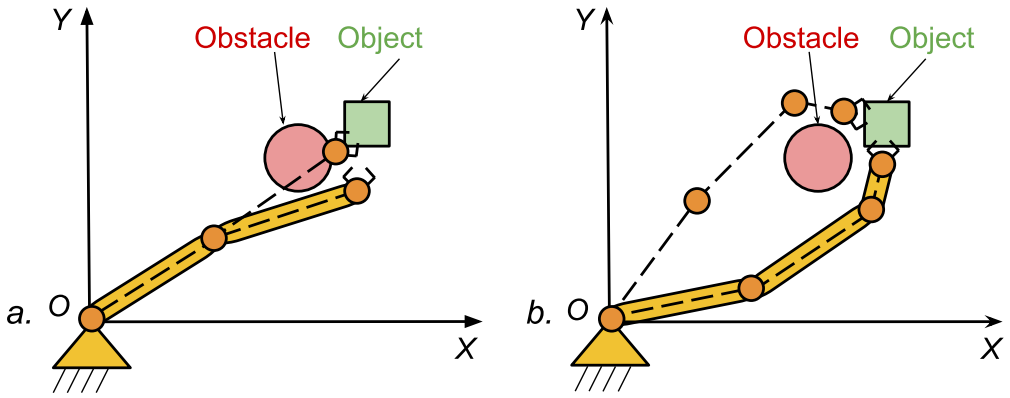
Мотивация на изследването

Тенденцията да се използват роботи във все по-сложни задачи, в които се изисква прецизност, гъвкавост и минимална вероятност за грешка, налага проектиране и създаване на по-гъвкави и прецизни роботизирани системи. Индустриалните манипулатори обикновено оперират в динамична среда, където трябва да работят съвместно с други роботи или хора и да изпълняват задачи като заваряване, сглобяване, боядисване и др. За тази цел те трябва да могат успешно да заобиколят новопоявили се препятствия в работната им зона по време на изпълнението на поставената им задача.

Ако се използват манипулатори, които имат толкова степени на свобода, колкото са необходими за изпълнението на дадена задача, то те биха имали някои ограничения. Докато

роботизираните системи изпълняват поставената им задача, се налага да съобразят своите движения със ставните си ограничения, наличието на препятствия и тяхното преодоляване [12]. Възможно е да има само една единствена ставна конфигурация за позициониране на хващача на желаната позиция или да няма нито една. Поради тази причина има вероятност целевата задача да не може да се изпълни коректно или въобще да не бъде изпълнена.

Да предположим, че манипулатор без допълнителни степени на свобода трябва да вземе даден предмет, като за целта трябва да заобиколи препятствие. Манипулаторът трябва да се съобрази с местоположението на препятствието и със ставните си ограничения. Трябва предварително да планира траекторията си и да определи през кои зони не може да премине. Спазвайки тези ограничения е възможно да няма подходящо решение или да има едно единствено решение, което да не е оптимално. На Фиг. 0.7.a. е изобразен двузвенен равнинен манипулатор, който трябва да стигне до определен предмет (зеления квадрат). Пред него обаче има препятствие (червеният кръг) и манипулаторът няма как да достигне до предмета без да се удари в препятствието. Пунктирът изобразява конфигурацията на манипулатора, с която той трябва да достигне до предмета.



Фиг. 0.7. Постановка за достигане до даден предмет при наличие на препятствие: a. робот без допълнителни степени на свобода b. робот с допълнителни степени на свобода.

Нека сега да разгледаме същата постановка на задачата, но използвайки робот с допълнителни степени на свобода. За изпълнение на произволно равнинно движение са необходими три степени на свобода. Изобразеният робот на Фиг. 0.7.b. има 4 ротационни стави. Роботът е с допълнителни степени на свобода относно изпълнението на произволно равнинно движение. Затова и възможните ставни конфигурации, с които би могъл да достигне до целевата позиция са повече от една. На фигурата са изобразени две от тях. Роботът успява успешно да заобиколи препятствието и да достигне до желания обект.

Роботите с допълнителни степени на свобода преодоляват значително по-лесно препятствията в работната им зона, докато с изпълнителното си звено изпълняват основната си задача. Въпреки всичките предимства, които този тип роботи имат пред роботите без

допълнителни степени на свобода, има и при тях някои особености. Както вече казахме, роботите с допълнителни степени на свобода имат множество от ставни конфигурации, които са решения на обратната задача на кинематиката. Те са както възможност за избор на най-подходящата ставна конфигурация, така и правят по-трудно решаването на обратната задача на кинематиката и планирането на движение. Затова при планиране на траектория е важно да се направи най-подходящият избор на ставна конфигурация. Съществуват различни метрики, които могат да се приложат за този избор. Как точно ще се избере най-подходящата ставна конфигурация зависи до голяма степен от поставената задача.

Решенията на обратната задача на кинематиката могат да се разделят на няколко типа в зависимост от стойностите на ставните ъгли. Работната зона на роботите манипулатори също може да се раздели на отделни зони в зависимост от съществуващите типове решения на обратната задача на кинематиката. При изпълнение на дадено движение, като преместване на хващача на робота от една точка до друга, роботът преминава от една ставна конфигурация в друга или извършва преход между две зони. Но това преминаване може да доведе до отклоняване от желаната траектория или увеличаване на времето за изпълнение. Затова е необходимо да се изследват различните зони в работното пространство на робота и да се предложи подход за намиране на точки, в които роботът може да промени ставната си конфигурация без отклоняване от желаната траектория. Допълнително, при наличие на препятствия, преходът между някои от зоните може да се окаже невъзможен. Ако пък поставената задача на робота е сглобяване и е необходимо в определена точка от работното му пространство да промени ориентацията на изпълнителното си звено, то е възможно при неправилно планиране на неговото движение, хващачът на робота да се отмести от желаната точка.

Възможните ограничения на роботите с допълнителни степени на свобода при изпълнение на зададено движение при наличие на препятствия мотивираха създаването на дисертацията с цел да се изследват различни методи за планиране на траектория в среда с налични статични и динамични препятствия. Планирането на движение е проблем изследван от много години и от много учени, но в литературата липсват изследвания относно класифицирането на решенията на обратната задача на кинематиката в зависимост от стойностите на ставните ъгли и използването на тази класификация при планиране на движение. За да бъдат валидирани предложените методи за планиране на траектория, ще се създаде 3D принтиран равнинен робот с допълнителни степени на свобода. Тази технология позволява бързо и ценово ориентирано създаване на прототип на робот, както и възможност за експериментиране с различни по форма и размери механични компоненти. Също така, ще се изберат подходящи хардуерни компоненти и ще се създаде софтуерна система за управлението на изследвания робот. Роботът ще е от антропоморфен вид, тъй като формата му наподобява човешка ръка.

Цел и задачи на дисертацията

Целта на тази дисертация е да се създаде математически модел и прототип на равнинен антропоморфен робот с допълнителни степени на свобода, както и да се изследват и създадат методи за управление на неговото движение. Провеждането на това изследване трябва да доведе до създаване на методи за планиране на траектория при наличие на статични или динамични препятствия в работната зона на робота. За постигане на желаната цел могат да се формулират следните задачи:

1. Класифициране на различни типове решения на обратната задача на кинематиката за равнинен робот с допълнителни степени на свобода.
2. Изследване и създаване на алгоритми за планиране на траектория за изследвания равнинен антропоморфен робот с допълнителни степени на свобода с цел преодоляване на статични и/или динамични препятствия и достигане до желана целева позиция.
3. Анализиране и подбор на подходящи хардуерни компоненти и проектиране на подходяща софтуерна система за управление на създадения равнинен робот с допълнителни степени на свобода.
4. Създаване на прототип на равнинен робот с допълнителни степени на свобода, посредством методите на 3D принтирането.
5. Верифициране на предложената хардуерна и софтуерна система за управление и на алгоритмите за планиране на траектория, чрез компютърна симулация и експеримент с проектираната роботизирана система.

Структура на дисертационния труд

Структурата на дисертационния труд се състои от Увод, 4 глави и Заключение и е с обем от 132 страници. Приложени са 75 фигури, 11 таблици, цитирани са 98 източника. В Увода се разглежда значението и актуалността на темата на дисертацията, дефинират се основните цели и задачи на дисертационния труд и се описва неговата структура.

Глава 1 представлява въведение в предметната област на изследването. В тази глава са описани различните видове роботи и техните приложения. Представен е обзор на използваните методи за решение на правата и обратната задача на кинематиката, както и различните методи за планиране на движение на роботизирани системи. Представени са ползите от 3D принтирането в роботиката и са описани някои от най-често използваните операционни системи за управление на роботи.

Глава 2 въвежда основните означения, които ще бъдат използвани в дисертацията и дефинира функционалните изисквания на моделирания модел на робот. Главата дава решение

на правата и обратната задача на кинематиката за разглеждания робот, разглежда неговите ограничения и кинематични характеристики, като коефициент на мобилност и ъгъл на сервис.

В Глава 3 се представят хардуерната и софтуерната система за управление на изследвания робот, предлагат се различни подходи за управление и планиране на траектория. Анализират се предимствата и недостатъците на разгледаните подходи.

Глава 4 описва процеса по създаване на 3D принтирания робот с допълнителни степени на свобода и верифицира описаните подходи за управление на движението на изследвания робот, посредством компютърна симулация и експерименти със създадения прототип.

Заключението обобщава получените резултати и изпълнените задачи, както и определя задачите за бъдещо развитие.

1. Обзор на предметната област

Развитието на роботите обикновено се разделя на 4 поколения. Първото поколение роботи е преди 1980 г., второто е в периода от 1980 до 1990 г., третото – от 1990 до 2000 г. и четвъртото след 2000 г. [13]. Разработването на тези роботизирани системи става възможно благодарение на развитието на различни научни области като механика, електроника, информатика, математика и др. Терминът робот се прилага за голямо разнообразие от механични устройства, като автономни автомобили, безпилотни летателни апарати и други. Роботите са системи, които изпълняват сложни контролни функции без или с минимална необходимост от човешка намеса [14]. Сред предимствата от използването на роботи са: намалени разходи за труд, по-добра прецизност и производителност, по-добра гъвкавост в сравнение със специализираните машини и замяна на човека в повтарящи се или животозастрашаващи дейности [4].

В тази глава ще бъдат разгледани различните видове роботи и техните приложения, ще се дефинират основни понятия, свързани с разглежданите роботи. Ще се анализират различни методи за планиране на движение при наличие на препятствия за роботи манипулатори, както и някои от най-често използваните операционни системи използвани за тяхното управление.

1.1 Предметна област на изследването

Роботизираните системи намират все по-голямо приложение в различни професионални области, като транспорт, медицина, контрол и наблюдение, развлечения и други [15, 16]. Всяка една от тези области и всяка една задача, в която участват изисква различни по вид и конструкция роботи. В тази глава ще бъдат разгледани различните видове роботи и техните основни приложения.

1.1.1. Видове роботи, дефиниции и означения

Роботите могат да се класифицират по различни критерии, като начин на задвижване, приложна област, начин на управление и др. В тази глава ще бъдат представени различните видове роботи според начина им на задвижване. По този критерий могат да се разделят на няколко класа: мобилни (колесни), крачещи или стационарни (манипулатори).

Мобилните роботи имат подвижна основа, която им позволява да се движат свободно. Те обикновено се състоят от едно или повече твърди тела със задвижваща система [1]. Използват се за транспортиране на различни предмети, почистване и др. Според данни на международната федерация по роботика, продажбата на мобилни роботи през последните години се е увеличила драстично. Появилата се пандемия от COVID-19 през 2020 година е предпоставка за увеличаване на броя сервизни роботи, използвани за дезинфекция в училища, университети, болници и други

[17]. Мобилните роботи асистенти се използват още в музеи и хотели за упътване и съпровождане на посетители [18].

Друг вид мобилни роботи са мобилните антропоморфни роботи (Фиг. 1.1.a.). Те имат мобилна основа и антропоморфно тяло. Използват се за преместване на предмети, сервиране, комуникация с клиенти, забавление или изследователски цели [19].

Крачещите роботи са различни видове: двукраки, четирикраки и т.н. Роботите, които по един или друг начин притежават човешки характеристики се наричат антропоморфни. Те могат да имат човекоподобни очи, лице, крака или ръка [20]. Крачещите или хуманоидните роботи са роботи, които могат да имитират човешки движения [21]. Тяхното предимство пред мобилните роботи е че могат да работят в среда с препятствия или неравен терен [22]. Но моделирането и управлението на крачещи роботи е по-сложно от това на мобилни и стационарни роботи. Затова и двукраките роботи се използват предимно за изследователски цели (Фиг. 1.1.b.).

Стационарните роботи или промишлени манипулатори имат неподвижна основа и механична ръка (Фиг. 1.1.c.). Те също се класифицират като антропоморфни роботи, тъй като имат човекоподобна ръка. Най-често се използват за сглобяване, заваряване или боядисване в различни индустриални области. Задачите, които трябва да изпълнят, изискват висока прецизност и точност. Затова манипулаторите са снабдени с различни сензори, в зависимост от дейността, която извършват (тактилни сензори, сензори за натиск, камери). Те могат да работят както в структурирана и статична среда, чийто геометрични и физични характеристики са известни предварително [1], така и в динамична среда, работейки заедно с други роботи или хора [23]. Такъв тип роботи се наричат кооперативни и се използват във все повече области като сортиране, производство, строителство [24], земеделие [25] и др.

Фиг. 1.1. Видове роботи: a. Мобилен антропоморфен робот Pepper [26]; b. Крачещ робот Asimo [27]; c.Индустриален манипулатор KUKA 500 Fortec.

Роботите могат да се класифицират спрямо типа на кинематичната им верига. Кинематичната верига е система от твърди тела, свързани помежду си чрез стави. Веригата се нарича затворена, ако тя образува затворен цикъл (Фиг. 1.2.a.). Верига, която не е затворена, се

нарича отворена (Фиг. 1.2.b.). Ако всяка връзка на една отворена верига с изключение на първата и последната връзка е свързана с две други връзки, тя се нарича последователна (серийна) верига.

Фиг. 1.2. Видове роботи спрямо кинематичната верига: a. роботи със затворена кинематична верига; b. роботи с отворена кинематична верига.

В тази дисертация обект на изследване ще бъдат манипулационните антропоморфни роботи с отворена кинематична верига. Те могат да се разделят на още два класа в зависимост от броя степени на свобода, с които разполагат.

Дефиниция 1.1

Конфигурационно пространство наричаме n-размерното пространство, съдържащо всички конфигурации на робота [16].

Конфигурацията на даден манипулатор напълно определя позицията на всяка точка на манипулатора. Тя определя взаимното разположение на звената му в даден момент. Конфигурацията на даден робот се представя, чрез точка в неговото конфигурационно пространство [16]. Една точка с дадени ставни ъгли съответства на точно една конфигурация. Затова управлението в конфигурационното пространство е по-удобно за планиране на движения.

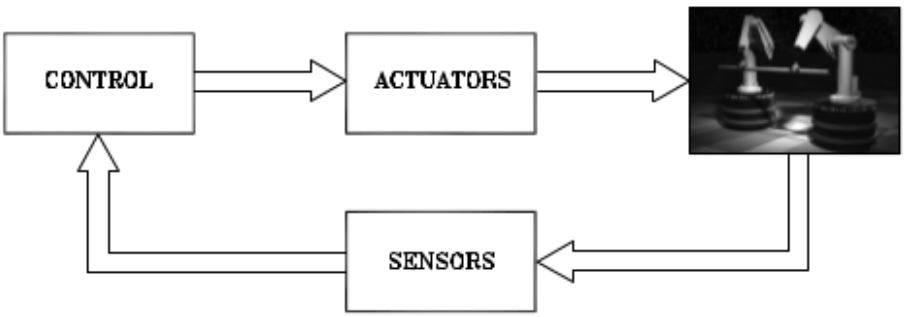
Дефиниция 1.2

Степени на свобода наричаме броя на независимите променливи *n,* необходими за еднозначно представяне на конфигурацията на манипулатор.

Това означава, че броят на степените на свобода е равен на размерността на конфигурационното пространство. При манипулаторите броят на ставите определя броя на степените на свобода. Едно твърдо тяло в тримерното пространство има 6 степени на свобода, 3 за позиция и 3 за ориентация. Ако една роботизирана система има по-малко от 6 степени на свобода, то роботът не би могъл да достигне всяка точка от работното си пространство с произволна ориентация. Някои задачи, като заобикаляне на препятствия дори изискват наличие на повече степени на свобода. Ако една роботизирана система има повече степени на свобода от

необходимите за изпълнение на дадена задача, се казва, че този робот има допълнителни степени на свобода. Използването на роботизирана система с допълнителни степени на свобода повишава производителността, точността и ефективността на тази система [28, 29].

Изпълнявайки поставените им задачи, роботите непрекъснато взаимодействат с околната среда. За тази цел те обикновено са изградени от няколко подсистеми, които обменят информация помежду си: сензори, задвижващи устройства и управляваща система (Фиг. 1.3) [18].



Фиг. 1.3. Компоненти на роботизирана система [4].

Задвижващите устройства отговарят за задвижването на механичните компоненти на робота и позволяват извършването на движение или манипулация. Обикновено са различни видове мотори и драйвери.

Сензорната система дава възможност за възприемане на информация както от външния свят (заобикалящата среда на робота), чрез сензори за натиск, близост или камери, така и за състоянието на механичните компоненти на робота като сензори за положение. Сензорната система изпраща събраните данни относно заобикалящата среда на робота или състоянието на робота към управляващата система.

Управляващата система следи получените данни от сензорите и подава команда за определено действие към задвижващите устройства. Тя управлява изпълнението на дадено действие в зависимост от зададените цели, както и от ограниченията зададени от робота и околната среда [1].

Следващата Глава 1.1.2 разглежда антропоморфни роботи с допълнителни степени на свобода.

1.1.2. Антропоморфни роботи с допълнителни степени на свобода

Ранните конструкции на манипулаторите са по-прости, за да са по-евтини за производство и по-лесни за поддръжка. Това води до появата на роботите тип SCARA, чиято основна задача е сглобяване. Но когато даден манипулатор разполага само с минималния брой степени на свобода за изпълнение на дадена задача, неговото движение може да бъде сериозно ограничено. Движението на манипулатора трябва да се планира съобразно ставните му ограничения и

наличните препятствия в работното му пространство. Затова повечето индустриални роботи манипулират в статична и детерминирана среда. Но тенденцията е все повече роботи да се използват в неструктурирана и динамична среда, където трябва да работят съвместно с други роботи или хора. Поради тази причина все повече роботи се проектират по такъв начин, че да имат повече степени на свобода отколкото са необходими за изпълнението на поставените им задачи.

Както вече казахме, един манипулатор е с допълнителни степени на свобода, когато има повече степени на свобода от необходимите за изпълнение на дадена задача. Произволно равнинно движение може да бъде изпълнено от робот с три степени на свобода. Ако се добавят допълнителни връзки (звена, стави) към конфигурацията на робота, то роботът ще има конфигурация, която е с допълнителни степени на свобода спрямо изпълнението на равнинно движение (от крайното звено). Разликата на размерността на ставното пространство и това на поставената задача определя степента на излишество (допълнителните степени на свобода) за дадена роботизирана система [30].

Роботите с допълнителни степени на свобода могат да достигнат дадена позиция и ориентация на крайното звено, използвайки различни конфигурации на механичната си структура [9]. Тези роботизирани системи се характеризират с по-добра гъвкавост и по-голяма производителност и точност при изпълнение на поставената задача [8, 28, 31]. Наличието на множество ставни конфигурации за дадена позиция и ориентация на крайното звено позволява по-лесно преодоляване на препятствия [32, 33] и колизии [34] в сравнение с останалите роботизирани системи.

Тенденцията индустриалните манипулатори да изпълняват все по-сложни и прецизни задачи и желанието да се придаде на тези манипулатори по-голяма сръчност и гъвкавост подобна на тази в човешката ръка е причината за появата на допълнителни степени на свобода към механичната структура на роботите. Кинематичното устройство на човешката ръка е възпроизведено в редица роботи, наричани манипулатори или антропоморфни роботи. Примери за антропоморфни роботи са 7-степенният DLR (Deutsches Zentrum für Luft- und Raumfahrt) (Фиг. 1.4.a.), както и роботи хуманоиди и мобилни манипулационни роботи: NASA (National Aeronautics and Space Administration) Robonaut (Фиг. 1.4.b.) и мобилният манипулатор на KUKA (Фиг. 1.4.c.) [35].

Фиг. 1.4. Антропоморфни роботи с допълни степени на свобода [35]: a. DLR (Deutsches Zentrum für Luft- und Raumfahrt); b. робот NASA Robonaut; c. мобилен манипулатор KUKA.

Планирането на движение на роботи с допълнителни степени на свобода е интересна тема, тъй като е необходим критерий за избор на подходяща ставна конфигурация в зависимост от поставената задача. В литературата са изследвани и предложени различни методи за избор на подходяща ставна конфигурация. Hollerbach предлага избор на такава ставна конфигурация, при която може да се постигне оптимизиране на въртящия момент [36], Hirakawa и Kawamura представят метод за избор на ставни конфигурации, който може да оптимизира консумираната от робота енергия [37], Chembuly и Voruganti изследват метод за преодоляване на препятствия [38] и Йошикава въвежда метриката коефициент на манипулативност за избор на ставна конфигурация [39]. Предложените методи в литературата за избор на подходяща ставна конфигурация са приложими за конкретните цели като оптимизиране на консумираната от робота енергия или прилагане на по-голяма сила. Решенията на обратната задача на такъв тип робот могат да се класифицират на няколко вида спрямо стойностите на ставните ъгли. Допълнително, на базата на различните типове решения, работната зона на робота може да се раздели на различни зони. Когато роботът променя ставната си конфигурация, може да се наложи да премине от една зона в друга. В предложените методи в литературата не се взима предвид факта, че когато роботът трябва да премине от една ставна конфигурация в друга, има вероятност да се отмести от желаната траектория. Това би довело до нарушения в движението, повече време за изпълнение на поставената задача или до некоректно изпълнение на задачата. Затова е необходимо да се изследват и намерят точките (зоните), в които роботът може да осъществи промяната в ставната конфигурация без това да доведе до отклонение от желаното движение. Целта на това изследване е да предложи метод за планиране на движение, който да се изпълни за минимално време и без отклонение от желаната траектория, вземайки предвид възможните точки за смяна на ставната конфигурация.

Следващата Глава 1.1.3 представя някои от най-популярните и използвани методи за планиране на движение.

1.1.3. Методи за планиране на движение

Движението на ставите на роботите обикновено е ограничено в определен диапазон. Хващачът на дадена роботизирана система трябва да изпълни движение от дадена стартова точка до определена целева позиция в работното пространство, заобикаляйки налични статични или динамични препятствия и съобразявайки се със ставните ограничения на робота. Планирането на движение се разглежда в конфигурационното пространство. Наличните препятствия в работната зона на робота се трансформират в конфигурационното пространство и формират, така наречената забранена зона на конфигурационното пространство. Основната цел на планирането на движение е да намери път свързващ дадена начална и крайна конфигурация и не минаващ през забранената зона [40]. Различни методи за планиране на движение са изследвани

и представени в литературата. Едни от най-популярните и използвани са Dijkstra, A*, Rapidly-exploring random tree и probabilistic roadmap методите [41]. В Таблица 1.1 са представени накратко приложенията, предимствата и недостатъците на всеки един от четирите алгоритъма.

Методът **Dijkstra** е предложен през 1959 г. от Edsger Dijkstra и се използва за намиране на най-късите пътища между възлите в даден граф [42]. Той се смята за ефективен и ефикасен метод дори при графи с голям брой възли. Възможните точки, през които роботът може да мине се представят като върхове. Алгоритъмът намира най-къс път от даден връх до друг даден връх. Алгоритъмът спира работа, след като е намерил най-краткия път между тези два върха. Методът пази информация за най-късия път намерен до момента, ако след това се намери по-къс път, информацията се актуализира [43].

Таблица 1.1. Сравнителен анализ на алгоритми: Dijkstra, A, RRT, PRM.*

Алгоритъм	Приложения	Предимства	Недостатъци
Dijkstra	Планиране на движение и преодоляване на препятствия.	Бърз, ефективен метод. Подходящ за граф с голям брой възли.	Възможно е да изисква повече време за намиране на решение.
A*	Планиране на движение и преодоляване на препятствия.	Бързо намира най-късия път, заобикаляйки препятствия.	При граф с повече възли, ефективността на алгоритъма намалява.
Rapidly exploring random tree (RRT algorithm)	Планиране на движение и преодоляване на препятствия в динамична среда.	Бързо намира подходящо решение.	Възможно е отклонение от желаното движение.
Probabilistic roadmap method (PRM)	Планиране на движение и преодоляване на препятствия в статична среда.	Лесен за имплементация метод.	Не винаги намира оптимално решение.

Алгоритъмът **A*** се основава на best-first search алгоритъма и използва евристична функция за намиране на най-краткия път чрез оценка на цената (сумарното тегло), оставаща до целевия връх [43]. Алгоритъмът започва от конкретен начален връх на даден граф и трябва да намери път до дадения целеви връх с най-малка цена (или най-малко изминато разстояние). За тази цел алгоритъмът непрекъснато търси за неизследван път в графа. Когато се достигне до целевата точка, алгоритъмът спира. Ако целта не е постигната, по последната достигната точка се определят всички съседни точки, които трябва да се изследват за оптимален път. Ако съществува решение, алгоритъмът със сигурност ще го намери. Подходящ е за планиране на движение с преодоляване на статични препятствия, но ако графът с възможните пътища е твърде голям, алгоритъмът ще загуби от ефективността си [44]. Алгоритъмът е популярен не само в роботиката, но и за намиране на път в игри [45].

Алгоритмите **Rapidly-exploring random tree** (RRT) и **Probabilistic roadmap method** (PRM) са също едни от най-популярните методи за планиране на движение [41]. RRT методът е често използван в различни комерсиални и индустриални задачи. Методът създава дърво, посредством което се изследва работното пространство на робота, чрез произволно търсене на подходящи ставни конфигурации [46]. Алгоритъмът е подходящ за динамична среда. PRM методът се използва по-често в статична среда с предварително известни препятствия [47]. Методът може да се раздели на две фази: фаза на обучение (learning phase) и фаза на заявка (query phase). Във фазата на обучение се създава вероятностна карта с възможни пътища. Пътищата се представят като граф. Във втората фаза се търси път сред възможните безопасни пътища, като се свързват началният връх от графа с целевия връх [41].

Следващата глава разглежда механичната конструкция на роботите и въвежда някои важни дефиниции, както и представя начини за решаване на правата и обратната задача на кинематиката.

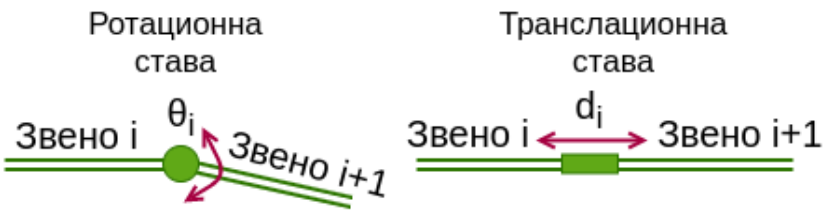
1.2 Подходи за моделиране на антропоморфни роботи

Механичната конструкция на робота се състои от множество твърди тела (звена, връзки), които са свързани помежду си. Обикновено звената, от които е изграден един робот се приемат за абсолютно твърди освен ако изрично не е уточнено друго [4].

Дефиниция 1.3

Абсолютно твърдо тяло се нарича такова тяло, при което разстоянието между всеки две точки остава постоянно във времето, независимо от силите действащи върху него [4].

Звената могат да бъдат свързани посредством различни по вид стави. Кинематичните връзки или ставите осигуряват подвижно съединение между две или повече звена на робота. Най-често се използват ротационни и транслационни стави (Фиг. 1.5).



Фиг. 1.5. Ротационна и транслационна става.

Ротационните стави позволяват относително въртене между две звена. Транслационните стави позволяват относително линейно движение между две звена. Оста на въртене при ротационна става или оста на транслация при линейна става се означава с z_i, ако ставата свързва звената i и $i+1$. Ставните координати, които изразяват относително отместване между съседни

звена се означават с θ за ротационна става и d за линейна става. Обикновено последното звено представлява инструмент за хващане или манипулиране на различни видове предмети. Друг вид стави са сферичните връзки, но те обикновено се използват като пасивни. Те са подходящи за компенсиране на пространствени грешки и намират приложение в различни конструкции, като роботи със затворена кинематична верига и др.

Дефиниция 1.4

Работно пространство се нарича множеството от конфигурации, които роботът може да достигне с изпълнителното си звено [16].

Работното пространство е ограничено от структурата на робота и от ставните ограничения. Възможно е ротационните стави да не могат да извършат пълно 360 градусово движение.

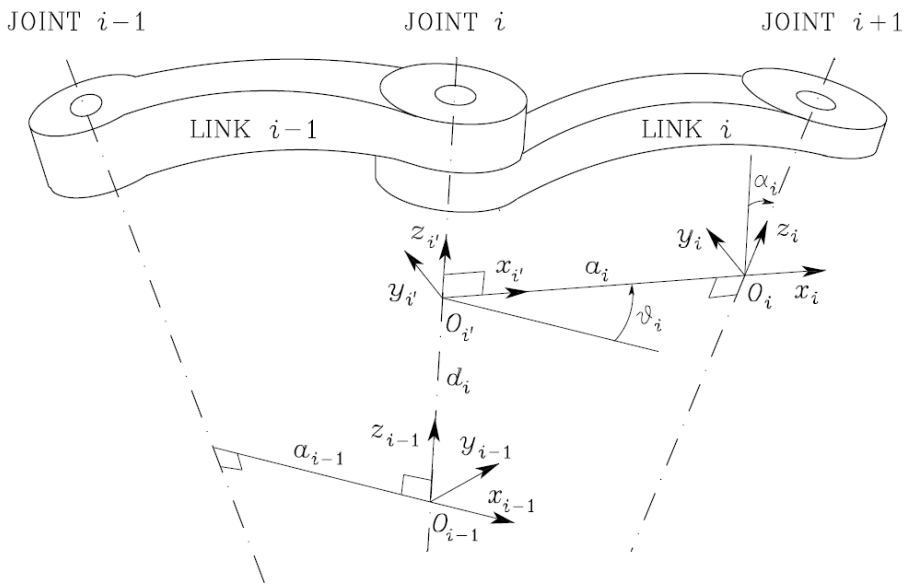
Основната задача на роботизираните системи е да извършат определено движение с цел изпълнение на поставените им задачи. Това движение не би било възможно да се осъществи без знанията за това каква позиция трябва да изпълни крайното звено на робота и какви стойности приемат останалите ставни координати при тази позиция. За тази цел ни е необходимо решение на правата и обратната задача на кинематиката.

1.2.1 Моделиране на кинематиката

Кинематиката изследва движението на една роботизирана система без да взима предвид силите и въртящите моменти, които предизвикват това движение. Кинематиката изследва позицията, скоростта и ускорението на ставните координати. Правата и обратната задача на кинематика се използват за проектиране на всяка една роботизирана система. Решенията, получени от правата и обратната задача на кинематиката определят как една система трябва да бъде проектирана, за да изпълнява ефективно своята задача [48].

Правата задача на кинематиката намира позицията и ориентацията на крайното звено на дадена роботизирана система по предварително зададени ставни (обобщени) координати. За нейното решаване е необходимо да се поставят локални координатни системи към всяко звено на роботизираната система. Най-разпространеният метод за избор на координатни системи за манипулатори с отворена кинематична верига е конвенцията на Денавит-Хартенберг (Фиг. 1.6), която дефинира релативната позиция и ориентация на две съседни звена. Необходимо е да се определят координатните системи на две съседни звена и да се намери координатната трансформация между тях. Избира се фиксирана координатна система $O_0x_0y_0z_0$ свързана с центъра на основата на манипулатора. Оста z_0 съвпада с оста на въртене и/или оста на транслация на първото звено спрямо основата, а осите x_0 и y_0 се избират, така че $O_0x_0y_0z_0$ да е дясноориентирана координатна система. Номерират се звената от основата към последното звено с $i = 1,2, \dots, N$ и към всяко звено се свързва неподвижно координатна система $O_ix_iy_iz_i$.

Конвенцията на Денавит-Хартенберг определя координатната система на i-тото звено. Оста $z_i, i = 1,2 \dots, N-1$ се избира по посока на оста на звеното. Оста $x_i, i = 1,2 \dots, N-1$ се определя от общия перпендикуляр между осите $z_i - 1$ и z_i с положителната посока от звено i към звено $i + 1$. Оста $y_i, i = 1,2 \dots, N-1$ се избира, така че да допълва дясноориентирана координатна система. Последната координатната система $O_n x_n y_n z_n$ се свързва неподвижно с крайното звено на манипулатора, като оста z_n определя ориентацията му и обикновено се явява негова ос на симетрия. Останалите оси се определят по описания начин [1].



Фиг. 1.6. Параметри на Денавит-Хартенберг [1].

След като са поставени всички локални координатни системи, позицията и ориентацията на координатната система i спрямо координатната система $i - 1$ са напълно определени от следните параметри (Фиг. 1.6): a_i е разстоянието между O_i и $O_{i'}$ по оста x_i, d_i е разстоянието между O_{i-1} и $O_{i'}$ по оста z_{i-1}, α_i е ъгълът между осите z_{i-1} и z_i спрямо оста x_i, θ_i е ъгълът между осите x_{i-1} и x_i спрямо оста z_{i-1}.

За да се изрази координатната трансформация между координатните системи $O_{i-1} x_{i-1} y_{i-1} z_{i-1}$ и $O_i x_i y_i z_i$ се изпълняват следните стъпки, състоящи се от постъпателни премествания и завъртания:

- Завъртане на ъгъл θ_i по оста z_{i-1} докато осите x_{i-1} и $x_{i'}$ станат успоредни.
- Постъпателно преместване по оста z_{i-1} на разстояние d_i до съвпадане на осите x_{i-1} и $x_{i'}$.
- Постъпателно преместване по оста $x_{i'}$ на разстояние a_i до съвпадане на координатното начало $O_{i'}$ с O_i.
- Завъртане на ъгъл α_i около оста $x_{i'}$ до съвпадане на всички координатни оси.

Трансформационната матрица между две координатни системи може да се изрази като произведение от четири трансформации:

$$\boldsymbol{T}_i^{i-1} = \boldsymbol{R}(Z_{i-1}, \theta_i)\boldsymbol{Tr}\left(Z_{i-1,} d_i\right)\boldsymbol{Tr}(X_i, a_i)\boldsymbol{R}(X_i, \alpha_i) \quad (1.1)$$

$$\boldsymbol{T}_i^{i-1} = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i) & 0 & 0 \\ sin(\theta_i) & cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha_i) & -sin(\alpha_i) & 0 \\ 0 & sin(\alpha_i) & cos(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.2)$$

След извършване на необходимите изчисления за трансформационната матрица $\boldsymbol{T}_i^{i-1}$ получаваме:

$$\boldsymbol{T}_i^{i-1} = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i)\,cos(\alpha_i) & sin(\theta_i)sin\,(\alpha_i) & a_i\,cos(\theta_i) \\ sin(\theta_i) & cos(\theta_i)\,cos(\alpha_i) & -cos(\theta_i)sin\,(\alpha_i) & a_i\,sin(\theta_i) \\ 0 & sin(\alpha_i) & cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.3)$$

Решението на правата задача на кинематиката се изразява като произведение на всички трансформационни матрици, които представят координатните трансформации между отделните звена. Конвенцията на Денавит-Хартенберг е приложима и при решаване на правата задача на кинематиката при манипулационни роботи с допълнителни степени на свобода.

При планиране на движение по зададена траектория е необходимо роботът да изпълни определени позиции и ориентации с изпълнителното си звено. За тази цел трябва да се намерят стойностите на ставните координати по зададена позиция и ориентация на хващача на манипулатора. Тази задача е известна като **обратната задача** на кинематиката [4]. Конфигурацията на робота се изразява с вектор стълб $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)^T$, който описва положението на всяка една става на роботизираната система. Казваме, че всеки манипулатор има n на брой звена и θ_i наричаме ставен ъгъл. Желаната позиция и ориентация на хващача може да се опише с хомогенна трансформационна матрица $\boldsymbol{H}$ с размерност 4x4, съгласно уравнение (1.4):

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{p} \\ 0 & 1 \end{bmatrix}, \quad (1.4)$$

където $\boldsymbol{p}$ е вектор стълб, който описва транслацията, а $\boldsymbol{R}$ е ротационна матрица с размерност 3x3. Обратната задача на кинематиката цели намирането на ставните координати $\theta_1, \ldots, \theta_n$, такива че $\boldsymbol{T}_n^0(\theta_1, \ldots, \theta_n) = \boldsymbol{H}$ [4].

Обратната задача на кинематиката е по-сложна за изчисление от правата задача на кинематиката, тъй като уравненията, които трябва да се решат са нелинейни. Възможно е да има повече от едно решение или да няма решение. Например, равнинен робот с 3 ротационни стави може да достигне всяка позиция в работното си пространство с различна ориентация [4]. Броят на решенията на обратната задача на кинематиката зависи от броя на ротационните стави на манипулатора, както и от допустимия обхват на движение на ставите (ставните ограничения) [5].

Ако изпълнителното звено на манипулатора е позиционирано и ориентирано в работната зона, то обратната задача има поне едно решение. За разлика от правата задача на кинематиката, за решението на обратната задача няма общоприет алгоритъм, по който да се реши. Обикновено се казва, че обратната задача е решена за даден манипулатор, ако ставните координати са определени по алгоритъм, който позволява намирането на всички конфигурации на ставните координати за дадена позиция и ориентация [49].

Също така, при роботите с допълнителни степени на свобода е възможно да има безкрайно много решения на обратната задача на кинематиката. Затова методите за намиране на решения на обратната задача на кинематиката на роботи без допълнителни степени на свобода не са приложими за роботи с допълнителни степени на свобода. Тъй като обект на изследване на тази дисертация са роботите с допълнителни степени на свобода, тук ще бъдат разгледани няколко метода за решаване на обратната задача на кинематиката, които са приложими за роботизирани системи с отворена кинематична верига и с допълнителни степени на свобода.

При роботите с допълнителни степени на свобода за решение на обратната задача на кинематиката най-често се използват итеративни методи, базирани на матрицата на Якоби. Те дават приближено решение и на всяка стъпка се опитват да минимизират разликата между текущата позиция и ориентация на хващача и желаната позиция и ориентация. Решенията на Якобиана представляват линейни приближения на обратната задача на кинематиката. Те моделират линейно движенията на крайното звено спрямо моментните промени в ставните ъгли. Матрицата на Якоби е функция на ставните ъгли $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_n)$ и се определя от уравнение (1.5) [50]:

$$\boldsymbol{J}_{ij}(\boldsymbol{\theta}) = \left(\frac{\partial \boldsymbol{s}_i}{\partial \theta_j}\right)_{ij}, \tag{1.5}$$

където $i = 1, \dots, n$ и $j = 1, \dots, m$. Целевата позиция се задава с вектор стълб $\boldsymbol{t} = (t_1, \dots, t_n)^T$ и текущата позиция се задава с вектор стълб $\boldsymbol{s} = (s_1, \dots, s_n)^T$. Използвайки текущите стойности $\boldsymbol{\theta}, \boldsymbol{s}$ и $\boldsymbol{t}$, матрицата на Якоби $\boldsymbol{J} = \boldsymbol{J}(\theta)$ може да бъде изчислена. Търсим нова стойност $\Delta\boldsymbol{\theta}$ за увеличаване на ставните ъгли $\boldsymbol{\theta}$ с $\Delta\boldsymbol{\theta}$: $\boldsymbol{\theta} \coloneqq \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$. Промяната в позицията на крайното звено в следствие на промяната в ставните ъгли, може да бъде изразена с помощта на уравнение (1.6):

$$\Delta\boldsymbol{s} \approx \boldsymbol{J}\Delta\boldsymbol{\theta} \tag{1.6}$$

Orin и Schrader описват в [51] как се пресмята матрицата на Якоби за различните видове стави. Матрицата на Якоби може да бъде пресметната за i-та ротационна става по следния начин: $\frac{\partial \boldsymbol{s}}{\partial \theta_i} = \boldsymbol{v}_i \times (\boldsymbol{s} - \boldsymbol{p}_i)$, където $\boldsymbol{p}_i$ е позицията на ставата, а $\boldsymbol{v}_i$ е вектор означаващ текущата ос на въртене на ставата. Ако i-та става е транслационна, то тогава трябва да намерим разстоянието, на което е преместена ставата в посока $\boldsymbol{v}_i$: $\frac{\partial \boldsymbol{s}}{\partial \theta_i} = \boldsymbol{v}_i$.

Най-лесният метод базиран на матрицата на Якоби е методът на **транспонираната матрица на Якоби**. Методът използва транспонираната матрица на Якоби вместо обратната, Използван е за първи път за решение на обратната задача на кинематиката в [52]. Промяната в ставните скорости $\Delta\boldsymbol{\theta}$ се изразява с уравнение (1.7):

$$\Delta\boldsymbol{\theta} = \alpha \boldsymbol{J}(\boldsymbol{\theta})^T \boldsymbol{e}, \tag{1.7}$$

където α е скаларна константа, която влияе върху промяната на $\boldsymbol{\theta}$ на всяка итерация [50] и $\boldsymbol{e}$ е вектор на грешката $(6x1)$, представящ отклонението на позицията и ориентацията между текущите позиция и ориентация и целевата позиция и ориентация $\boldsymbol{e} = \boldsymbol{t} - \boldsymbol{s}$. Стойността на α се избира такава че, новата стойност на вектора $\boldsymbol{e}$ да бъде минимална. Предимствата на този метод са: минималното време за изчисление и лесната имплементация [53], но недостатъкът на метода са лошите му резултати в близост до сингулярни конфигурации [54].

Обратният метод на Якоби използва обратната матрица на Якоби за изчисляване на промяната в ставните координати. Обратната матрица на Якоби съществува само при условие, че матрицата на Якоби има пълен ранг. Размерността на матрицата на Якоби расте с увеличаване на степените на свобода на робота, затова и методът ще изисква много повече време за намиране на обратната матрица при роботи с допълнителни степени на свобода [55]. В близост до сингулярна конфигурация този метод е неприложим, тъй като рангът на матрицата на Якоби не е пълен [56]. При сингулярна конфигурация роботът не би могъл да извърши движение в някоя от посоките.

Добро приблизително решение на обратната задача на кинематиката дава методът на **псевдообратната матрица на Якоби** или известна още като обратната матрица на Moore-Penrose. Тя се означава с $\boldsymbol{J}^{\dagger}$и е $n \times m$ матрица и се изразява с уравнение (1.8):

$$\Delta\boldsymbol{\theta} = \boldsymbol{J}(\boldsymbol{\theta})^{\dagger}\boldsymbol{e} \tag{1.8}$$

Уравнение (1.8) дава най-малкото квадратно решение за $\Delta\dot{\boldsymbol{\theta}}$, вектор с най-малка величина и минимизира $\|\boldsymbol{J}\Delta\boldsymbol{\theta} - \boldsymbol{e}\|$ или еквивалентно минимизира $\|\boldsymbol{J}\Delta\boldsymbol{\theta} - \boldsymbol{e}\|^2$. За роботизирани системи с допълнителни степени на свобода, псевдообратната матрица на Якоби се изразява със следното уравнение (1.9):

$$\boldsymbol{J}^{\dagger} = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1} \tag{1.9}$$

Методът на псевдообратната матрица на Якоби е сравнително бърз, но с не много добра апроксимация [46]. Методът дава като резултат много голяма скорост на крайното звено в близост до сингулярна конфигурация или когато точката, в която трябва да се позиционира крайното звено на робота е извън зоната му на достижимост.

Основният общ проблем на разгледаните до тук методи е лошата им производителност в близост до сингулярна конфигурация. С този проблем може да се справи методът на Левенберг-Маркуад – **метод на затихващите най-малки квадрати** (Damped Least Squares, DLS) с

коефициент на затихване (damping factor). Коефициентът на затихване може да намали голямата скорост в близост до сингулярност. Методът е бил използван за решаване на обратната задача на кинематиката за първи път от Wampler [57]. Решението на метода се изразява с уравнение (1.10):

$$\Delta\dot{\boldsymbol{\theta}} = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T + \lambda^2\boldsymbol{I})^{-1}\boldsymbol{e}, \tag{1.10}$$

където λ е ненулев коефициент на затихване. Колкото по-висока е стойността на коефициента на затихване, толкова е по-висока устойчивостта срещу сингулярност, но в същото време точността на метода намалява. От друга страна по-ниската стойност на коефициента на затихване дава по-лоши резултати при сингулярни конфигурации. Стойността на коефициента на затихване трябва да бъде по-голяма от най-малката собствена стойност (smallest singular value) на матрицата на Якоби [58]. Най-малката собствена стойност може да бъде намерена, чрез декомпозиция на собствени стойности (singular value decomposition) на матрицата на Якоби: $\boldsymbol{J} = \sum_{i=0}^{n} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i$, където $\boldsymbol{u}_i, \boldsymbol{v}_i$ са съответно входен и изходен сингулярен вектор и σ_i е сингулярна стойност.

Ако коефициентът на затихване е избран да бъде константа, то методът DLS не би могъл да се използва пълноценно. Затова е необходимо да се избере динамично променящ се коефициент на затихване, като се вземе предвид близостта до сингулярна конфигурация. Близостта до сингулярна конфигурация може да бъде оценена от най-малката собствена стойност на матрицата на Якоби [59].

Някои учени [53, 60] определят DLS метода за най-добрия за намиране на решение на обратната задача на кинематиката на роботи с допълнителни степени на свобода в сравнение с останалите методи, базиращи се на матрицата на Якоби. Той се справя с проблемите на псевдообратната матрица на Якоби и намалява високите скорости в близост до сингулярност. Работи значително по-добре в сравнение с метода на транспонираната матрица на Якоби. Неговият недостатък е че е по-бавен в сравнение с останалите разгледани методи, необходимо е повече изчислително време [53].

Друг итеративен метод за решаване на обратната задача на кинематиката е методът на **покоординатното спускане**. Той е лесен за имплементиране и с висока производителност. Алгоритъмът се стреми да минимизира разликата между позицията на крайното звено и желаната позиция. Извършва се по една ставна трансформация на итерация. Трансформациите се извършват от крайното звено към първото звено, като се цели трансформация на всяко звено. Алгоритъмът продължава докато не се достигне до желаната позиция. Неговият недостатък е че се извършва по една ставна трансформация на итерация и при роботи с повече степени на свобода би имал нужда от повече време за изчисление [54].

Освен моделирането на кинематиката за създаването на една роботизирана система е необходимо и правилно управление. За да се премине към управлението на един робот е

необходимо да се проектира софтуерна система. Следващата глава разглежда и коментира необходима архитектура на софтуерната система за управление на роботи, както и изборът на подходяща операционна система и междинна платформа.

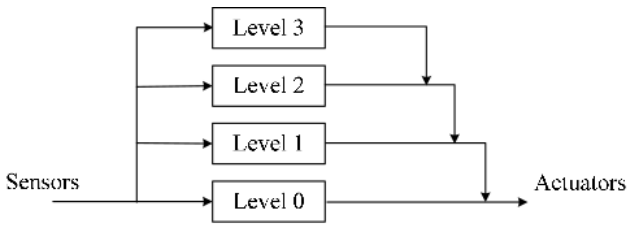
1.3 Софтуерна система за управление

Роботизираните системи изпълняват определени задачи в зависимост от предназначението им (боядисване, заваряване, сортиране, пренасяне на тежки товари и др.). За да бъдат изпълнени ефективно целевите задачи, е необходима софтуерната и хардуерна система за управление. Софтуерната система отговаря за възможността на роботите да получават и обработват информация, чрез сензорна система за заобикалящата ги среда. Също така, отговаря за способността на роботите да вземат решение за следващото движение, което трябва да изпълнят – избягване на препятствие, преместване на обекти в работното пространство и др. В основата на това стои правилната софтуерна реализация и подходящо избраната архитектура. От друга страна, хардуерната система е не по-малко важна за управлението на даден робот. Тя предоставя необходимите изчислителни ресурси за моделиране на кинематиката, обработката на различни сензорни данни, както и отговаря за задвижването на моторите.

1.3.1 Архитектура на софтуерната система за управление на роботи

Архитектурата на роботизираните системи обикновено се различава от другите софтуерни архитектури, тъй като роботите имат някои по-специфични изисквания. Например, те трябва да могат да реагират на външно въздействие в реално време и да манипулират в неструктурирана, динамично променяща се среда. Затова и архитектурата на софтуерната система за управление на роботи е необходимо да включва възможност за реагиране в реално време, да управлява задвижващи устройства и сензори, да поддържа откриване и бързо реагиране на външни въздействия [35].

През 1986 година R. A. Brooks и колегите му въвеждат така наречената "**subsumption architecture**" (Фиг. 1.7). Архитектурата се състои от различни нива, които изпълняват отделни функции. Всяка функция обработва определени данни от сензорите, след което на изпълнителните механизми се подават подходящи команди. Архитектурата е наречена реактивна, тъй като има директна връзка между сензорите и задвижващите устройства [61].



Фиг. 1.7. „Subsumption" архитектура.

Най-често за ефективна имплементация на софтуерна система за управление на индустриални роботи се използва така наречената **функционална архитектура** [1]. Функционалната архитектура се състои от няколко нива (слоя), описващи различни дейности, подредени в йерархична структура. Това означава, че долните нива на структурата отговарят за изпълнението на физическото движение на робота, докато по-високите нива са ориентирани към логическото планиране на действията. Комуникацията между нивата се осъществява, чрез обменяне на информация. Като долните нива изпращат информация към по-високите нива относно измервания или резултати от определени действия. Информацията, която получават ниските нива е свързана с команди за определени движения, които роботът трябва да изпълни.

Обикновено софтуерната система за управление се състои от няколко функционални модула, които отговарят за дейностите от всяко едно ниво. Най-често тези модули са: модул за сензорите, модул за моделиране на заобикалящата среда и модул за вземане на решения.

Модулът за сензорите отговаря за обработката на данните получени от сензорите. Този модул събира и обработва информация относно състоянието на робота и характеристиките на средата, в която е позициониран (налични препятствия, тяхната геометрия и местоположение и др.). Сензорите отчитат определени данни (разстояние, натиск, цвят и др.), на базата, на които роботът определя следващото си действие (отдалечаване от препятствие, завиване, хващане или поставяне на даден обект и др.). Затова софтуерната система трябва да разполага с подсистема, която да обработи получените от сензорите данни и да изпрати подходяща команда към роботизираната система.

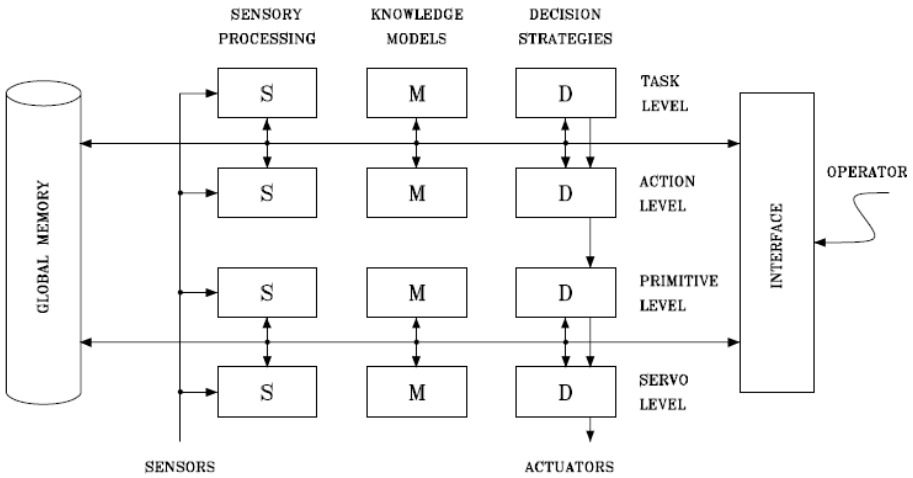
Вторият модул отговаря за изготвяне на модел на работното пространство на робота. Като се извършва и сравнение между текущата информация за заобикалящата среда и тази получена от модула на сензорите. При разминаване, модулът за моделиране на заобикалящата среда допълва своята информация.

Задачата на модула за взимане на решения е да превърне задачи от високо ниво в задачи от ниско ниво. Този модул отговаря за управлението на базови задачи, като планиране и изпълнение на определено движение.

Необходимо е всяко ниво от йерархичната структура на функционалната архитектура да разполага с интерфейс за управление. По този начин операторът би могъл да оказва контрол върху задачите, които изпълнява роботизираната система. Инструкциите, които получава модулът за взимане на решения на определено ниво, могат да бъдат получени от оператора, от по-горното ниво или като комбинация от двете. Допълнително, операторът може да получи информация за състоянието на роботизираната система.

Различните нива и модули на функционалната архитектура обменят голям поток от информация. Затова е необходима и глобална памет, която да съхранява и обновява информация за състоянието на цялата система и средата, в която манипулира роботизираната система.

На Фиг. 1.8 е показана примерна схема на функционална архитектура на софтуерна система за управление на роботизирани системи [1].



Фиг. 1.8. Примерен модел на функционална архитектура на софтуерна система за управление на роботи [1].

Архитектурата се състои от 4 нива: ниво задачи (task level), ниво действие/взимане на решения (action level), примитивно ниво (primitive level) и ниво мотори (servo level). Всяко ниво е разделено на три модула, отговарящи съответно за обработката на сензорните данни, представяне на модела на средата, в която манипулира роботът и взимането на решения. Когато операторът зададе определена задача на робота, в най-високото ниво (task level) трябва да се анализира и раздели на последователност от действия зададената задача. Последователността на действия се предава на по-ниското ниво (action level), където получените команди се трансформират в последователност от конфигурации. Тези конфигурации дефинират целевото движение. Тук модулът за взимане на решения планира движението на хващача на робота и проверява дали някоя конфигурация не предизвиква колизия с препятствия в работната зона. За тази цел се използват данните от модула за знания за средата, в която манипулира роботизираната система. Получените конфигурации от това ниво се предават на следващото по-ниско примитивно ниво, където се изчисляват възможните траектории и се планира стратегията за управление. В това планиране е важно да се вземат предвид особеностите на механичната конструкция на робота, степени на свобода, ставни ограничения. Целта на това ниво е да подаде подходящи команди за изпълнение на поставената от оператора задача към най-ниското ниво, нивото на моторите. В нивото на моторите се имплементира избраният алгоритъм за управление и се изпращат подходящи сигнали към сервомоторите на робота за изпълнение на желаната задача [1].

Всички нива имат достъп до глобалната памет и до интерфейса, който позволява на оператора да следи състоянието на роботизираната система и да изпраща команди към нея.

Друга важна задача на софтуерната система за управление е да пресмята ефективно и бързо правата и обратната задача на кинематиката. Решенията на обратната задача на кинематиката са от първостепенна важност за планиране на движение на роботизираната система. Подсистемата, която отговаря за планиране на движението трябва да знае коя е следващата целева точка, която крайното звено на робота трябва да достигне. След което е необходимо да се пресметне обратната задача на кинематиката и на база намерените решения и избрания критерий за избор на подходяща ставна конфигурация да се изпрати команда към моторите на робота за постигане на желаните ставни ъгли. Разбира се, при планирането на движение трябва да са взети предвид всички ставни ограничения на робота.

Важен компонент на софтуерната система за управление на роботизирани системи е потребителският интерфейс. Потребителският интерфейс е необходимо да бъде максимално интуитивен и лесен за работа. От него потребителят трябва да може да получи информация за текущата позиция на робота, за състоянието на моторите (включени/изключени), за наличие на препятствия в работната зона или за възникнала грешка по време на работата на робота. Също така, потребителят трябва да има възможност да управлява робота, чрез изпращане на различни команди, в зависимост от задачата, която изпълнява.

Роботите са системи, които изискват бърза обработка на данни, затова изборът на подходяща операционна система за роботи е от важно значение. Следващата глава разглежда популярни и често използвани операционни системи за роботи.

1.3.2 Операционни системи за роботи

Тъй като роботизираните системи получават информация за заобикалящата ги среда, използвайки различни сензори или камери е необходимо да могат непрекъснато да обработват получените от сензорите данни. Тази обработка трябва да се извърши в реално време, за да може роботът да планира следващото си действие (заобикаляне на препятствие, взимане на предмет, оставяне на предмет и др.). Стандартните операционни системи имат някои ограничения като ниска мощност или критично време за реакция. За да бъдат преодолени тези ограничения, роботите използват операционни системи за работа в реално време (реалновремеви операционни системи) [62]. Те могат да обработват различни задачи с различни нива на приоритет. Благодарение на това се получава предсказуем и точен резултат при изпълнение на критични задачи. Тези системи могат да бъдат с твърдо реално време, с гъвкаво реално време или с гарантирано (firm) реално време. При операционните системи с твърдо реално време има зададен максимален срок за изпълнение на всяка последователна итерация, който не може да бъде нарушен. Неспазването на този срок дори и от една от итерациите, означава, че дадената функция не се изпълнява. При гъвкаво реално време периодът за отговор се дефинира на базата на средно време за отговор, така че част от заявките се изпълняват, други отпадат. В този случай също е

дефиниран срок и колкото повече се забави дадена итерация спрямо този срок, толкова вероятността да се използва резултатът от нея намалява. При гарантирано реално време са дефинирани не един, а два периода – кратък, с характеристиките на гъвкаво реално време и по-продължителен, с характеристиките на твърдо реално време. Не се допуска закъснение на итерация след по-продължителния период [14]. Някои от най-използваните реалновремеви операционни системи за роботизирани системи са: VxWorks, Linux, Real-Time Application Interface и LynxOS. В Таблица 1.2 са представени основни характеристики на тези операционни системи.

Таблица 1.2. Сравнение на VxWorks, Linux, RATI и LynxOS операционни системи.

Операционна система	Архитектура	Вид реално време	Основни характеристики	Приложения
VxWorks	Монолитна ядро.	Твърдо реално време.	Приоритетно планиране с прекъсващи приоритети и кръгово планиране, 256 приоритетни нива.	Космонавтика, роботика, медицина. Роботи на ABB, Honda.
Linux	Монолитно ядро.	Не е реалновремева операционна система.	140 приоритетни нива, алгоритъм за напълно честно планиране (CFS).	Самолетни автопилоти, дронове, роботизирани системи.
Real-Time Application Interface (RTAI)	Използва Linux ядро.	Твърдо и гъвкаво реално време.	Бърза обработка на прекъсвания.	Приложения със строги времеви ограничения.
LynxOS	Микроядро.	Твърдо реално време.	Планиране базирано на прекъсващи приоритети, 512 приоритетни нива	Роботика, авионика.

VxWorks е една от най-често използваните реалновремеви операционни системи във вградените и роботизираните системи. Разработена е от WindRiver systems, Калифорния [63]. Разполага с мрежова поддръжка, файлова система и входно-изходно управление. Микроядрото поддържа 256 приоритетни нива, многозадачност, детерминирано превключване на контекста и планиране на семафори [64]. Друга широко разпространена операционна система е **Linux** и различните му дистрибуции. Една от характеристиките на Linux е ефективното управление както на големи системи с огромно количество памет, така и на вградени устройства с малка и съкратена йерархия на паметта. Операционната система Linux не е реалновремева операционна система, но с приоритет изпълнява реалновремеви задачи. Подходяща е за многозадачна обработка. За да се надгради Linux за работа в твърдо реално време се използва **Real-Time Application Interface** (RTAI). По този начин се постига бърза обработка на прекъсвания и изпълнение на задачи със строги времеви ограничения [65]. Друга съвместима с Linux операционна система е **LynxOS** (от LinuxWorks) с твърдо реалновремево планиране и

гарантирано максимално време за отговор. Планирането се основава на прекъсващи приоритети [18]. LynxOS е многонишкова операционна система, проектирана за сложни реалновремеви приложения. Микроядрото може да планира, управлява прекъсвания и синхронизира задачи. Също така операционната система поддържа TCP/IP потоци, файлова система, сокети и др. Поддържа 512 приоритетни нива [64].

При разработването на софтуерна система за управление на роботизирани системи могат да се използват специални софтуерни продукти, наречени междинни платформи. Тяхната задача е да улеснят разработването на софтуерната система. Следващата глава разглежда някои от популярните междинни платформи за роботи.

1.3.3 Междинни платформи за роботи

За улесняване на процеса по проектиране на софтуерната система се използват така наречените междинни платформи (мидълуер). Междинната платформа е абстракционен слой, който се намира между операционната система и софтуерните приложения. Междинната платформа се използва за опростяване на софтуерния дизайн и намаляване на разходите за разработка [66]. Разработчикът трябва само да изгради логиката или алгоритъма като компонент, след което компонентът може да се комбинира и интегрира с други съществуващи компоненти. Междинният софтуер трябва да бъде лесен за използване, надежден и лесен за поддръжка. За да бъде широко използвана междинната платформа трябва да може да работи с различни операционни системи: Windows, Linux, MacOS и др., както и да поддържа различни програмни езици: C/C++, Java, Python и др. [66]. В Таблица 1.3 са представени някои от популярните междинни платформи за роботи. Таблицата представя операционните системи, с които са съвместими, програмните езици, които поддържат, както и кратко описание на всяка една междинна платформа.

Таблица 1.3. Междинни платформи за управление на роботи.

Междинна платформа	Създатели	Операционни системи	Програмни езици	Описание
Player/Stage	Robotics Research Lab, University of Southern California (USA)	Linux, Solaris и BSD Unix.	C/C++, Tcl, Java и Python.	Платформата се базира на клиент-сървър архитектура, която използва TCP/IP сокети. Stage е плъгин към Player, който симулира популация от роботи, сензори и обекти в 2D среди. [67, 68]. Осигурява различни видове сензори и изпълнителни механизми.
Universal Robotic Body Interface (URBI)	Gostai (Aldebaran Robotics group)	Windows, Linux и MacOS.	Използва собствен език, предоставя	Клиент-сървър архитектура. Сървърната страна отговаря за управлението на робота от ниско ниво, като управление на ставите и сензорите. Клиентската страна

			библиотеки libUrbi за различни езици като Java, Matlab, Lisp и C++ [69].	се използва за управление на робота от по-високо ниво, като изпращане на действия от по-високо ниво като метод на ходене и др.
Robot Operating System (ROS)	Willow Garage в сътрудничество с Университета в Станфорд	Операционни системи, базирани на Unix (Linux, OSX и др.), Windows	C++, Python, Java, Matlab и Lisp.	Основните концепции в ROS са възлите (nodes, процеси за изпълнение на дадена задача), топици (topics) (средство за комуникация между процесите - Inter-Process Communication) и сървизи (удобни средства за двупосочна комуникация и лесно изпълнение на клиент-сървър взаимоотношения между възлите). Комуникацията между различните възли може да се изпълнява чрез TCP/IP връзка по мрежа. Всеки възел може да изпраща или получава данни от друг възел, чрез публикуване и абониране за съответния топик [70].
Open Robot Control Software (OROCOS)	European Robotics Research Network (EURON).	Linux, Windows и MacOS, поддържа ядра в реално време като RTAI и Xenomai [71]	C/C++, Tcl/Tk и други езици като Python и Simulink Orocos Scripting Languange [71].	Състои се от набор от преизползваеми компоненти, включително RTT (The Real-Time Toolkit), KDL (библиотека за кинематика и динамика), BFL (библиотека за Байесово филтриране) и драйвери за роботизирано оборудване.
Webots	Cyberbotics Ltd.	Linux, Windows, MacOS.	C, C++, Python, Java, MATLAB [72].	Софтуерът включва богата колекция от различни модели на роботи, сензори и задвижващи устройства [73]. Също така, потребителят може сам да създаде свой модел на робот или да импортира вече съществуващ 3D модел.

Представените в Таблица 1.3 междинни платформи за управление на роботи са с отворен код. **Player/Stage** поддържа много роботизирани системи като Summit RB-Rtk-01 офроуд робот на Robotnik [74], робот Koala [75], Pioneer 3-DX [76] и др. Player/Stage предлага 2D симулатор (Stage) и може да се интегрира с друг 3D симулатор (Gazebo). В такъв случай се нарича Player/Stage/Gazebo вместо Player/Stage [77]. **Universal Robotic Body Interface** (URBI) включва и Webots, който е неговият официален 3D симулатор на роботи. URBI поддържа различни роботи, като Jazz of Gostai, Aibo от Sony, HRP-2 от Kawada, Nao от Aldebaran Robotics, Pioneer

P3-DX мобилен робот и други [78]. **Robot Operating System** (ROS) осигурява колекции на хардуерна абстракция, драйвери на устройства, библиотеки, визуализатори, предаване на съобщения, управление на пакети и инструменти, които целят да опростят задачата по създаване на сложна и надеждна роботизирана система [79]. Използва rviz (2D визуализатор), Stage (2D симулатор) и Gazebo (3D симулатор). За разлика от другите платформи **Open Robot Control Software** (OROCOS) не включва среда за симулация [80]. **Webots** позволява бързо моделиране, програмиране и симулиране на различни видове роботи [81]. Използва се за създаване на 3D симулации на различни видове роботи за образователни, индустриални или изследователски цели. При моделирането на даден робот се определят както геометричните, така и физичните свойства. Софтуерът разполага с библиотека за откриване на колизии и симулация на динамиката на твърди тела.

В **клиент-сървър** (client-server) комуникационния протокол, компонентите комуникират директно с другите компоненти. Примерен модел на този протокол е Remote Procedure Call (RPC). Отдалечено извикване на процедури. Идеята е извикването на отдалечена процедура да изглежда като извикване на локална процедура (да се постигне прозрачност). CORBA (common object request broker architecture) е друг пример за клиент-сървър комуникация. Позволява на даден обект да извиква методи, които са имплементирани от друг компонент. Всички методи са дефинирани, използвайки IDL файлове (interface definition language). Предимство на client-server комуникацията е че интерфейсите са предварително ясно дефинирани и се знае кога даден интерфейс се променя. Друго предимство е, че позволява разпределен подход за комуникация без централен модул, който да разпределя данните [82]. Този комуникационен протокол се използва в различни междинни платформи като Player/Stage и URBI.

В **publish-subscribe** комуникационния протокол, даден компонент публикува информация и всеки друг компонент може да се абонира за тази информация. Обикновено централизиран процес маршрутизира информацията между компонентите, които публикуват и които се абонират. Пример за publish-subscribe протокол е Java Message Service (JMS). JMS е стандарт за изпращане на съобщения, който позволява на компонентите на приложенията, базирани на Java Platform Enterprise Edition да създават, изпращат, получават и четат съобщения. Позволява разпределена комуникация, която е надеждна и асинхронна [83]. Предимство на publish-subscribe протокола е че е лесен за използване и е особено полезен, когато броят на компонентите, които ще използват дадена информация е неизвестен. Недостатък на publish-subscribe протоколите е, че са по-трудни за отстраняване на грешки. Този комуникационен протокол се използва в ROS.

В различните междинни платформи се използват различни комуникационни протоколи. Както видяхме от разгледаните междинни платформи, най-често използваните комуникационни протоколи са client-server протокола и publish-subscribe протокола.

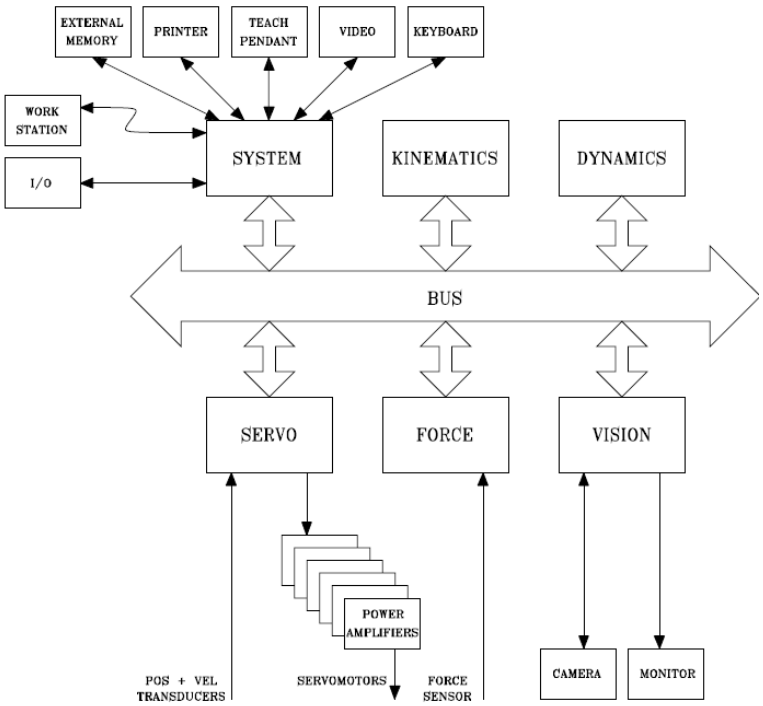
До тук разгледахме основните елементи при проектиране на софтуерна система за управление на роботи. Следващата глава представя необходимата архитектура на хардуерна система за управление, която трябва да може успешно да предостави необходими изчислителни ресурси за софтуерната система за управление на роботизирани системи.

1.3.4 Архитектура на хардуерна система за управление на роботи

Йерархичната структура на функционалната архитектура, приета като референтен модел за софтуерна система за управление на промишлен робот предполага хардуерно управление, което използва разпределени изчислителни ресурси, свързани помежду си посредством подходящи комуникационни канали [1]. Функциите, реализирани в системите за управление, се отнасят за всички нива от ниво мотори (серво управление) до ниво действие. На серво и примитивно ниво се изискват изчислителни способности с изискване за работа в реално време.

Общ модел на хардуерната архитектура за системата за управление на индустриален робот е илюстриран на Фиг. 1.9. Различните контролери с автономни изчислителни възможности са свързани към основна входно/изходна шина, която позволява поддръжка на комуникационния поток от данни; честотната лента на шината трябва да бъде достатъчно широка, за да удовлетвори изискванията, наложени от ограниченията в реално време.

Системният контролер обикновено е процесор, който разполага с локална RAM памет, серийни и паралелни портове, система за прекъсване и др. Допълнително, системният контролер трябва да има достъп до външна памет, която да служи за съхраняване на данни и приложни програми, входно/изходен интерфейс за периферни устройства и др.



Фиг. 1.9. Общ модел на хардуерна архитектура за управление на индустриален робот [1].

Другите модули, свързани към основна входно/изходна шина, могат да разполагат освен с основните компоненти на главния контролер, но и с допълнителен или алтернативен процесор за изпълнение на изискващи изчисления или специални функции. По отношение на архитектурата на Фиг. 1.9 имаме пет различни модула, отговарящи за изчислението на кинематиката, динамиката, задвижването на моторите на робота и обработката на данни от сензори за сила и камера.

Въпреки че всичките контролери са свързани към една и съща шина, честотата, с която се обменят данни, не е задължително да бъде еднаква за всички контролери. Контролерите, които отговарят за задвижването на моторите трябва да обменят информация с робота на възможно най-високата честота, напр., от 100 до 1000 Hz. От друга страна, контролерите за моделиране на кинематиката и динамиката не изискват актуализация на данните с толкова висока скорост [1].

В моделирането и управлението на роботизирани системи освен моделирането на кинематиката, изборът на подходящи методи за управление и проектирането на софтуерна система се включва и реализирането на реален модел на робот. 3D принтирането прави възможно бързото създаване на прототип. Но за да се стигне до етапа на принтиране на робота е необходимо преди това да се изготви негов 3D модел и да се избере подходящ материал за печат. 3D принтирането позволява да се експериментира с различни размери и форми и да се изберат най-подходящите. Основните етапи и приложения при 3D принтирането са разгледани в следващата глава.

1.4 Проектиране на роботи, чрез 3D принтиране

През последните години 3D печатът се развива изключително бързо, с прилагане на различни принципи на работа, използване на различни материали и различни области на приложение. Предимството на 3D принтирането е че позволява лесно реализиране на сложни форми с висока точност [84]. С помощта на 3D принтирането се намаляват значително времето и цената за създаване на прототипи. Концепцията на модела, на който е направен 3D печата, може да бъде изпробвана, а деформации, дефекти или неточности могат лесно и бързо могат да бъдат идентифицирани и по-късно променени в софтуерния модел на продукта. Обикновено закупуването на даден манипулационен робот изисква доста финансови средства. 3D принтирането позволява пресъздаване на различни по вид роботи на ниска цена. Това дава възможност за използване на 3D принтирани роботи в различни научни и образователни институции [84, 85].

В тази глава се разглеждат някои от най-използваните програми за моделиране на 3D обекти, основни изисквания при моделирането на манипулатори, както и материали за 3D печат.

1.4.1 CAD програми за моделиране и симулация

3D принтирането позволява бързо и лесно изготвяне на прототип в дадена сфера. За тази цел обаче е необходимо да се създаде 3D модел на обекта, който искаме да създадем. За това се използват различни софтуерни продукти за компютърно подпомогнато проектиране (CAD, Computer Aided Design). Целта при създаването на модела е максимално точно да представя функционалните качества на моделирания обект. Най-често при проектирането се използват множество от характеристични точки (вектори), които описват формата и разположението на обектите. Едни от най-разпространените софтуерни инструменти за моделиране на 3D обекти са представени в Таблица 1.4:

Таблица 1.4. Сравнение на OpenSCAD, AutoCAD и Blender софтуери за 3D моделиране.

CAD Софтуер	Предназначение	Операционни системи	Особености
AutoCAD	Софтуер за създаване на 2D/3D обекти. Използва се от широк кръг специалисти като инженери, архитекти, интериорни и графични дизайнери и др. Поддържа създаването на анимации.	Windows и MacOS.	Има богат набор от инструменти, поради тази причина спада към по-сложните инструменти за 3D моделиране [86].
OpenSCAD	Софтуер за създаване на 3D обекти. Възможна е анимация със скорост от няколко изображения в секунда за прости модели.	Linux, Windows, MacOS.	OpenSCAD използва собствен програмен език за описание на параметрите на желания обект в скриптов файл [87].
Blender	Софтуер за създаване на 3D модели, анимационни филми, визуални ефекти, компютърни игри и др.	Linux, Windows, MacOS, Android.	Предлага голямо разнообразие от инструменти, затова е по-сложен за разучаване и използване [88].

И трите софтуерни програми **AutoCAD**, **OpenSCAD** и **Blender** за моделиране са широко използвани в различни области. OpenSCAD и Blender са безплатни за използвани. С помощта на Blender се създават различни анимирани герои и анимационни филми. OpenSCAD се използва масово сред любителите на 3D принтирането, които имат познания в програмирането. AutoCAD е безплатен само за научни и образователни цели, но е подходящ за изучаване на основните концепции в 3D моделирането. Освен познаването и използването на определен софтуерен продукт за създаването на даден прототип е необходимо да се съобразят и някои особености в конструкцията.

Следващата глава разглежда особеностите при моделиране на манипулационни роботи с цел 3D принтиране.

1.4.2 Моделиране на манипулационни роботи с цел 3D принтиране

При проектиране на конструкцията на даден манипулационен робот е необходимо да се отчетат изискванията към звената му. Те зависят от множество условия, които може да са посочени в заданието или зависят от него като следствие. Важни са условията на работа, средата в която работи робота и задачите, които трябва да изпълнява. Необходимо е да удовлетворява с размерите и формата си необходимата метрика и възможност за реализиране на ставните ограничения. Звената трябва да имат необходимата здравина, както и минимална маса. Също така е необходимо всяко едно звено да може да се монтира и демонтира лесно в конструкцията. Това позволява реализирането на експерименти с различни по дължина или форма на звената, както и бърза и лесна промяна на хващача на робота. Към звената обикновено се закрепват различни по вид датчици и сензори. Затова трябва да се предвидят канали за кабели и закрепващи елементи [84, 89].

След като са определени и спазени всички изисквания към конструкцията на моделирания манипулационен робот и е създаден негов тримерен модел може да се премине към неговото 3D принтиране. За тази цел се използват различни по вид материали. Следващата секция разглежда най-използваните материали за 3D принтиране.

1.4.3 Технологии и материали за 3D принтери

Обикновено 3D принтерите създават обектите с добавяне на материал, като изграждат обектите слой по слой. Принтерите използват различни видове материали (стомана, титан, злато, сребро, керамика, каучук) в различни цветове и прозрачност. Най-често използваната технология за 3D печат представлява отлагане на разтопен материал (fused deposition modelling – FDM). При тази технология се създават обекти от разтопена пластмаса, която се екструдира през дюзата на 3D принтера. Пластмасовата нишка се навива на макара и се подава към дюзата за екструдиране. Едновременно с това дюзата или обектът (или и двете) се преместват по три взаимно перпендикулярни оси от компютърно контролиран механизъм. Материалът се втвърдява веднага след екструзия [84].

Различни видове материали се използват от различните принтери за 3D печат, например, метали – стомана, титан, злато, сребро, керамика, каучук и др. Най-разпространени са два основни вида пластмаса. Те се използват масово за печат, тъй като цената им е ниска. Това са: PLA и ABS.

PLA е най-разпространеният материал на основата на полимлечна киселина (Polylactide Acid, PLA). Той е биоразградим и биоактивен термопластичен алифатен полиестер, извлечен от

възобновяеми ресурси. Сред тези суровини попадат царевично нишесте, захарни суровини и др. Температурата му на топене е в диапазона 180–230° C. Обектите, отпечатани от PLA, са стабилни, но крехки. Затова не са подходящи за използване при висока температура. Над 60 градуса може да се деформира [84].

Друг широко разпространен материал е ABS. Той е направен на основата на акрилонитрил бутадиен стирол. Това е пластмаса на петролна основа. Изпаренията на този материал се считат за опасни за здравето. Затова при по-дълга работа с ABS е необходимо да се използва допълнителна вентилация за извличане на дима. Температурата на топене на ABS е 210–260° C. Предимството на ABS над PLA е, че получените обекти са по-стабилни, по-малко крехки и могат да устоят на по-високи температури [84].

1.5 Заключение

В тази глава са разгледани различните видове роботи и техните приложения. Представени са няколко класификации на роботизираните системи в зависимост от тяхното задвижване, в зависимост от кинематичната им схема и в зависимост от броя на степените им на свобода. Направен е обзор на различни методи за решение на правата и обратната задача на кинематиката, както и на методи за управление и планиране на движение на роботизирани системи. В главата са обсъдени и някои важни компоненти при проектирането на софтуерни приложения за роботи. Разгледани са популярни архитектури на софтуерна и хардуерна система за управление на роботи, както и различни операционни системи и междинни платформи. Обсъдени са и техниките за 3D печат и приложенията на 3D принтирането в роботиката.

Следващата глава дефинира функционалните изисквания и желаните характеристики, на които проектираният робот в тази дисертация трябва да отговаря и ще бъдат разгледани различни кинематични характеристики на робота.

2. Моделиране на равнинен антропоморфен робот

Моделирането на роботи е основна стъпка от разработването на роботизирани системи. Неговата цел е да изследва и създаде кинематичен модел на робота.

За целите на дисертацията ще се моделира и проектира равнинен робот с допълнителни степени на свобода. Казваме, че роботът е антропоморфен, тъй като формата му наподобява човешка ръка. В тази глава ще се представят функционалните изисквания за изследвания робот и ще се въведат основните означения, които ще бъдат използвани и в следващите глави на дисертационния труд. Ще се опишат използваните методи за решаване на правата и обратната задача на кинематиката за разглеждания антропоморфен робот, ще се изследват някои кинематични характеристики, като коефициент на манипулативност и ъгъл на сервис.

2.1 Функционални изисквания

Основната задача на роботите е да изпълняват зададено движение от точка до точка или по предварително зададена траектория. За тази цел е необходимо да се вземат предвид конструктивните ограничения на робота, както и ограниченията породени от средата, в която той манипулира. Обикновено тези ограничения са препятствия в работната зона на робота. Те могат да бъдат статични или динамични. Предварителна информация, като местоположение и геометрична форма е налична за статичните препятствия, докато в случай на динамични препятствия, роботът трябва в процеса на работа (в реално време) да събира информация за тяхната позиция в пространството. И в двата случая целта на роботизираната система е да изпълни зададената задача максимално прецизно, точно и безопасно, преодолявайки наличните препятствия. Антропоморфният модел на робот, който ще бъде моделиран в този дисертационен труд е необходимо да може да изпълни поставена задача и в статична и в динамична среда. За тази цел той трябва да отговаря на следните изисквания:

- Роботът ще изпълнява движения в O_{xy} равнината. В предходната глава разгледахме предимствата на роботите с допълнителни степени на свобода при преодоляване на препятствия. Тези предимства мотивираха избора роботът да бъде с допълнителни степени на свобода. Това означава, че роботът трябва да има поне три звена и 4 ротационни стави за изпълнение на движения в равнината.
- Дължините на звената и ставните ограничения трябва да са съобразени с параметрите на задвижващите механизми, които ще се използват. Необходимо е да се покрива максимално работно пространство. Теглата на звената трябва да се минимизират, което ще подобри динамичните параметри на робота. Звената трябва да имат достатъчна здравина, за да не се деформират при взаимодействие с обекти или препятствия.

- Ставите на робота трябва да оказват минимално съпротивление при движение. Необходимо е да могат да се реализират пространствени траекторни движения.
- Необходимо е моделираният робот да може да изпълнява дадена траектория прецизно, без отклонение от зададения път и за минимално време.
- Роботът трябва да разполага с подходяща хардуерна и софтуерна система с помощта, на която да може да локализира и преодолява наличните препятствия в работната си зона. Това би позволило на робота да работи в динамична среда съвместно с други роботи или хора. В такъв вид среда е възможно във всеки един момент да се появи непредвидено препятствие. Роботът трябва да може да реагира по подходящ начин в подобна ситуация.

Следващата глава описва робота, който ще се моделира, проектира и изследва в тази дисертация и въвежда някои основни означения за него.

2.2 Описание на антропоморфен робот

Както вече казахме, роботите с допълнителни степени на свобода могат посредством множество от различни ставни конфигурации да достигнат с изпълнителното си звено до една и съща точка от работното им пространство. Ако един робот има повече степени на свобода, отколкото са му необходими за изпълнение на дадена задача, то той е с допълнителни степени на свобода. Нека с m да означим броя на независимите параметри, описващи изпълнението на дадено задание в работното пространство на робота, а с n – броя на степените на свобода на робота. За да се класифицира един робот, като робот с допълнителни степени на свобода при изпълнение на движение в равнината O_{xyz}, трябва n да бъде по-голямо от m. Положението на хващача на даден манипулатор се определя от позиция и ориентация. За изпълнение на дадена позиция и ориентация на хващача в равнината O_{xyz} са необходими общо 6 степени на свобода, три степени на свобода за транслационни движения по осите x, y и z и три степени на свобода за ротация около трите оси. За изпълнение на произволно равнинно движение са необходими 3 степени на свобода (две за транслация и една за ротация), т.е. $m = 3$. В дисертацията ще се разглеждат равнинни движения, затова е необходимо роботът да има повече от 3 степени на свобода.

Нека с $l_i, i = 1, \dots, 4$ да означим дължините на звената на робота. Моделираният робот ще има 4 ротационни стави (Фиг. 2.1). Нека с $\theta_i, i = 1, \dots, 4$ да означим ставните координати на робота. Крайното звено на робота ще наричаме изпълнително звено или хващач. Позицията на хващача в равнината O_{xyz} ще означаваме с (x, y, z). Роботът ще изпълнява задания, при които оста z се приема за константа ($z = const$) и един от векторите на ориентация остава успореден на O_z. Ротационните стави ще имат ставни ограничения:

$$\theta_i^{min} \leq \theta_i \leq \theta_i^{max}, \text{за } i = 1, \dots, 4 \quad (2.1)$$

При изпълнение на предварително дефинирано движение е необходимо роботът да може да преодолее наличните препятствия в работната си зона и да изпълни зададеното движение по безопасен и точен начин. За тази цел роботът трябва да използва подходяща ставна конфигурация, която позволява изпълнение на движението без опасност от сблъсък с някое от наличните препятствия.

Първата стъпка от моделирането на даден робот, без която не би могло да бъде възможно по-нататъшното му управление е решаването на правата и обратната задача на кинематиката. Основната задача на кинематиката е да изследва позицията, скоростта и ускорението на ставните координати. Моделирането на кинематиката на даден робот обикновено се състои от две части: решаване на права и обратна задача. Правата задача на кинематиката намира позицията и ориентацията на хващача при дадени ставни координати. Обратната задача на кинематиката намира стойностите на ставните координати при дадени позиция и ориентация на хващача. Решаването на обратната задача на кинематиката е важна част от планирането на движение и изпълнението на зададена траектория. Възможно е да няма решение или да има безкрайно много решения. Правата задача при роботи с отворена кинематична верига има едно единствено решение и като резултат се получава една конфигурация на робота.

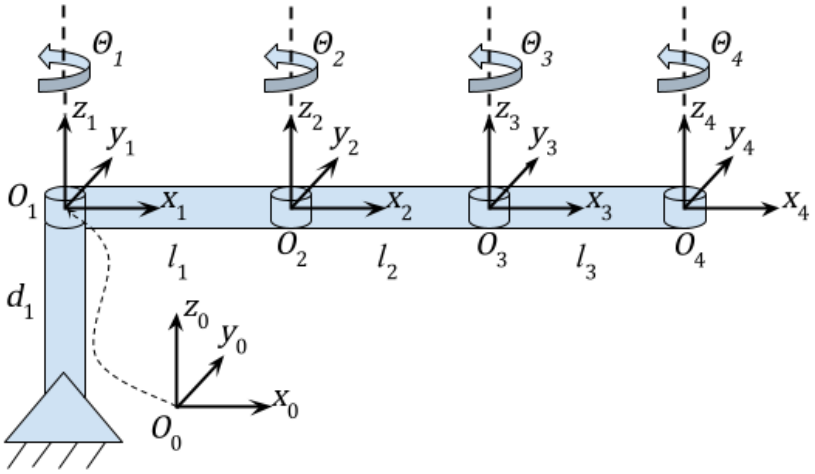
2.3 Права задача на кинематиката

При роботизирани системи с отворена кинематична верига ставните координати определят еднозначно позицията и ориентацията на хващача. Целта на правата задача на кинематиката е да се намери позицията и ориентацията на хващача [16].

Изследваният робот (Фиг. 2.1) с допълнителни степени на свобода има 4 ротационни стави и съответно ставни координати - $\theta_1, \theta_2, \theta_3$ и θ_4. Когато четвъртото звено е с нулева дължина, то четвъртата ставна координата θ_4 променя само ориентацията на хващача и не оказва влияние върху неговата позиция. В такъв случай, ставните координати $\theta_1, \theta_2, \theta_3$ определят позицията на хващача в O_{xy} равнината. Интересуваме се само от позицията на хващача, тъй като ориентацията се получава от сумата на четирите ставни координати. Тя се променя, чрез задаване на ставната координата θ_4. На Фиг. 2.1 е представена кинематичната схема на моделирания равнинен робот с допълнителни степени на свобода. Нека да означим позицията на хващача в работното пространство с (x, y) и ставните координати в обобщеното пространство с $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4)$. За решаването на правата задача на кинематиката ще използваме конвенцията на Денавит-Хартенберг, която е подробно представена в предната глава.

Координатна система $O_i x_i y_i z_i$ е поставена на всяка става i, $i = 0, 1, 2, \dots, n$, като координатната система $O_0 x_0 y_0 z_0$ е поставена към основата на робота. Оста z_i съвпада с оста на

въртене на става $i+1$ спрямо става i. За удобство ос z_0 е избрана да съвпада с ос z_1 и $O_0 \equiv O_1$. Ос z_n е избрана, така че да представя ос на ротация на хващача. Ос x_i е перпендикулярна на двете оси z_{i-1} и z_i. Ос y_i допълва $O_i x_i y_i z_i$ към дясно ориентирана координатна система.



Фиг. 2.1. Кинематична схема на равнинен робот с допълнителни степени на свобода.

След като към всяка става е поставена неподвижна локална координатна система $O_i x_i y_i z_i$, може да се определи позицията и ориентацията на координатна система $O_i x_i y_i z_i$ спрямо координатна система $O_{i-1} x_{i-1} y_{i-1} z_{i-1}$. За тази цел се използват следните параметри. Разстоянието между O_{i-1} и O_i по оста x_i се означава с a_{i-1}, разстоянието между две съседни координатни системи по оста z_i се представя с d_i, α_{i-1} е ъгълът между осите z_{i-1} и z_i спрямо оста x_i. Обобщените координати θ_i представят ротация спрямо оста z_i, като ставата i е съответно ротационна става [90].

Стойностите на параметрите на Денавит-Хартенберг за изследвания робот с допълнителни степени на свобода са представени в Таблица 2.1.

Таблица 2.1. Параметри на Денавит-Хартенберг.

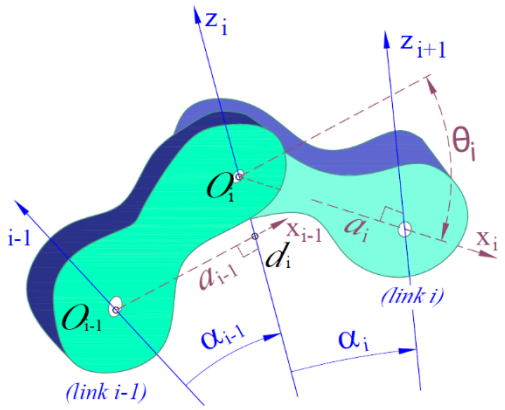
i	α_{i-1} [rad]	a_{i-1} [mm]	d_i [mm]
1	0	0	d_1
2	0	l_1	0
3	0	l_2	0
4	0	l_3	0

Конвенцията на Денавит-Хартенберг ни позволява да изразим всяка трансформационна матрица между две координатни системи (Фиг. 2.2), като се извършат следните постъпателни премествания и завъртания.

Завъртане на ъгъл θ_i около оста z_{i-1}, докато осите x_{i-1} и x_i станат успоредни. Постъпателно преместване по оста z_{i-1} на разстояние d_i до съвпадане на осите x_{i-1} и x_i. Постъпателно

преместване по оста x_i на разстояние a_i до съвпадане на координатното начало O_{i-1} с O_i. Завъртане на ъгъл α_i около оста x_i до съвпадане на всички координатни оси. Тези трансформации се представят като произведение на четири матрици (1.1).

След умножение на дадените матрици за трансформационната матрица $\boldsymbol{T}_i^{i-1}$ получаваме матрицата от (1.3).



Фиг. 2.2. Кинематична схема на две съседни звена.

За да решим правата задача на кинематиката е необходимо да намерим трансформационна матрица $\boldsymbol{T}_4^0$, която се изразява като произведение на всички трансформационни матрици:

$$\boldsymbol{T}_4^0 = \boldsymbol{T}_1^0\boldsymbol{T}_2^1\boldsymbol{T}_3^2\boldsymbol{T}_4^3 \tag{2.2}$$

След изразяване и заместване в (1.3) и (2.2) за трансформационната матрица $\boldsymbol{T}_4^0$ получаваме следния резултат (2.3):

$$\boldsymbol{T}_4^0 = \begin{bmatrix} c_{1234} & -s_{1234} & 0 & a_1c_1 + a_2c_{12} + a_3c_{123} \\ s_{1234} & c_{1234} & 0 & a_1s_1 + a_2s_{12} + a_3s_{123} \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

Където за $1 \leq i, j, k, l \leq 4$:

- $s_i = \sin(\theta_i), c_i = \cos(\theta_i)$,
- $s_{ij} = \sin(\theta_i + \theta_j), c_{ij} = \cos(\theta_i + \theta_j)$,
- $s_{ijk} = \sin(\theta_i + \theta_j + \theta_k), c_{ijk} = \cos(\theta_i + \theta_j + \theta_k)$,
- $s_{ijkl} = \sin(\theta_i + \theta_j + \theta_k + \theta_l), c_{ijkl} = \cos(\theta_i + \theta_j + \theta_k + \theta_l)$.

Така за x, y и z координатите, определящи позицията на крайното звено на робота, получаваме (2.4).

$$x = a_1c_1 + a_2c_{12} + a_3c_{123}, y = a_1s_1 + a_2s_{12} + a_3s_{123}, z = d_1 \tag{2.4}$$

В случаите, когато роботът трябва да изпълни зададена траектория е необходимо да знаем множеството от ставни конфигурации, които определят определена позиция на хващача на

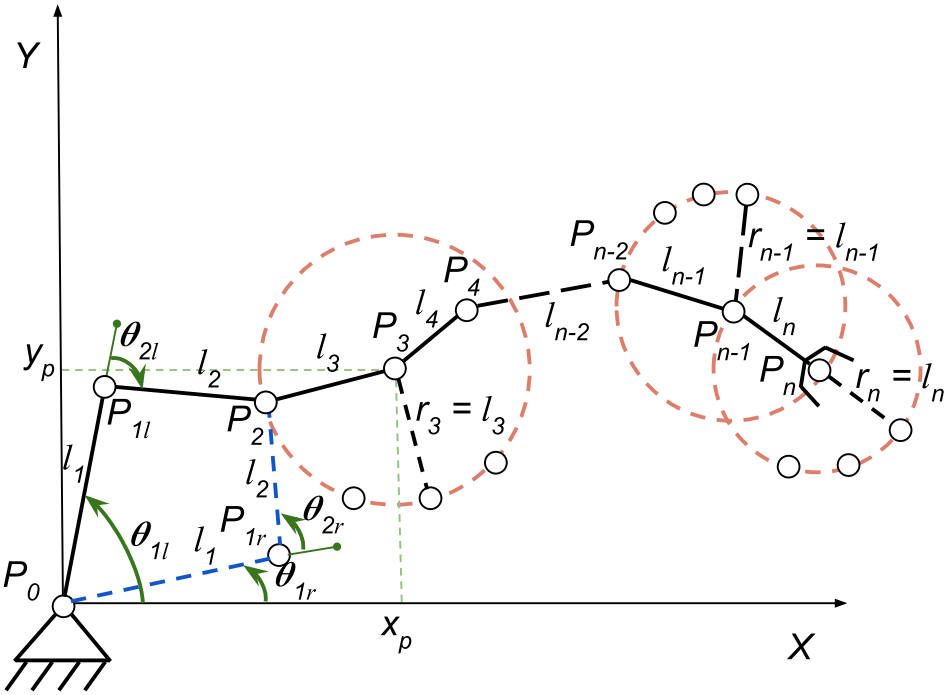
робота. За да намерим това множество е необходимо да решим обратната задача на кинематиката.

2.4 Обратна задача на кинематиката и класифициране на намерените решения

Целта на обратната задача на кинематиката е да се намерят необходимите ставни координати при дадена позиция и ориентация на хващача на робота. Решението на обратната задача на кинематиката при роботи с отворена кинематична структура най-често се използва за определяне на множество от точки, с които може да се достигне до определена точка в конфигурационното пространство при планиране на движение. Тъй като роботите с допълнителни степени на свобода могат да достигнат дадена позиция в работната им зона с множество ставни конфигурации, то и решенията на обратната задача на кинематиката са безкрайно много.

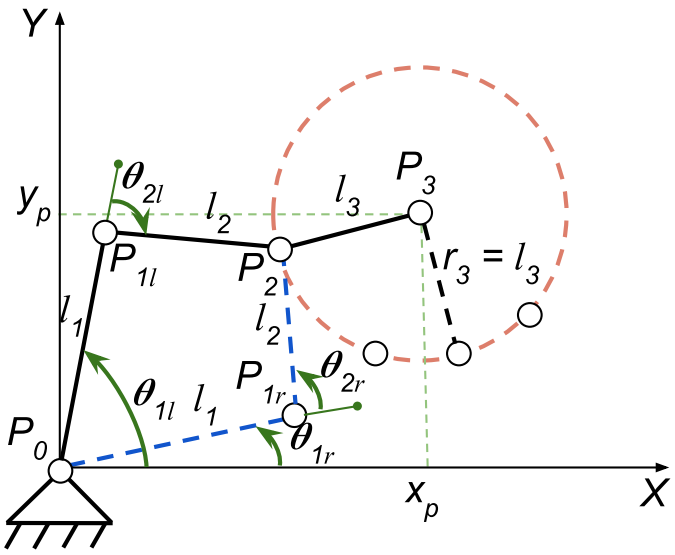
В Глава 1 разгледахме някои от популярните методи базирани на матрицата на Якоби за решаване на обратната задача на кинематиката за роботи с допълнителни степени на свобода. Повечето от разгледаните алгоритми не се справят добре, когато роботът се намира в ставна конфигурация близка до сингулярност, изискват повече време за изчисление или не успяват да намерят всички решения на обратната задача на кинематиката. Затова за целите на този дисертационен труд ще бъде използван геометричен подход за решаване на обратната задача на кинематиката, който преодолява тези ограничения [49]. На Фиг. 2.3 е показан равнинен манипулатор с n звена, роботът е с допълнителни степени на свобода. С $l_1, \dots, l_n$, където $i = 1, \dots, n$, са означени дължините на звената. Хващачът на робота трябва да се позиционира в точка P_n. За тази цел е необходимо да се намерят стойностите на ставните координати, с помощта на които хващача достига целевата точка. Подходът за намиране на решение на обратната задача на кинематиката започва от хващача (от последната ставна координата) и търси възможно решение за всяка една ставна координата докато се стигне до първата ставна координата. Генерира се множество $\boldsymbol{S}_n$ от точки равномерно разпределени върху окръжност с център P_n и радиус $r_n = l_n$. За всяка точка P_{n-1} от множеството $\boldsymbol{S}_n$ се проверява дали съществува конфигурация, с която да позиционира края на звено $n-1$ в точка P_{n-1}. За тази цел по аналогичен начин се генерира множество $\boldsymbol{S}_{n-1}$ от точки равномерно разпределени върху окръжност с център P_{n-1} и радиус $r_{n-1} = l_{n-1}$. Тези действия се повтарят за всяка една ставна координата от последното звено към основата на робота. Когато се достигне до множество $\boldsymbol{S}_3$ от точки равномерно разпределени върху окръжност с център P_3 и радиус $r_3 = l_3$. В този случай се търси решение на обратната задача на кинематиката на двузвенен механизъм. Този проблем е

известен и изследван. Необходимо е за всяка точка P_2 от множеството $\boldsymbol{S}_3$, да се провери дали съществува валидна конфигурация за двузвенния механизъм.



Фиг. 2.3. Решение на обратната задача на кинематиката на равнинен робот с n звена.

Нека разгледаме подробно Фиг. 2.4. Роботът трябва да позиционира третото си звено в точка P_3. Решение съществува, ако точка P_3 е на разстояние $R \leq (l_1 + l_2 + l_3)$ от центъра $O \equiv P_0$. Целта е да се намери решение на обратната задача на кинематиката за всяка точка P_2 от множеството $\boldsymbol{S}_3$ на двузвенен механизъм, състоящ се от звена l_1 и l_2 .



Фиг. 2.4. Решение на обратната задача на кинематиката на равнинен робот с допълнителни степени на свобода.

Поради конструкцията на робота в общия случай са възможни две решения. В частни случаи е възможно да няма решение, когато точката не е достижима или има само едно решение, когато $\theta_2 = \theta_3 = 0$. Поради геометричната конструкция на робота лесно могат да бъдат определени решенията на описаните случаи:

$$\text{Ако } |l_1 - l_2| < \sqrt{P_{2x}^2 + P_{2y}^2} < l_1 + l_2\,, \tag{2.5}$$

то съществуват две решения, когато:

$$\alpha = artg\left(\frac{P_{2y}}{P_{2x}}\right), \beta = \arccos\left(\frac{P_{2x}^2+P_{2y}^2+l_1^1-l_2^2}{2l_1\sqrt{P_{2x}^2+P_{2y}^2}}\right), \gamma = \arccos\left(\frac{l_1^1+l_2^2-P_{2x}^2-P_{2y}^2}{2l_1 l_2}\right). \tag{2.6}$$

Решение 1 (ще го наричаме решение от тип лява ръка) се дефинира по следния начин:

$$\theta_{1l} = \alpha - \beta, \theta_{2l} = \pi - \gamma. \tag{2.7}$$

Решение 2 (ще го наричаме решение от тип дясна ръка):

$$\theta_{1r} = \alpha + \beta, \theta_{2r} = \gamma - \pi. \tag{2.8}$$

Помощните ъгли α и β са дефинирани в (2.6) , а всички останали геометрични параметри са съгласно Фиг. 2.3.

$$\text{Ако } |l_1 - l_2| > \sqrt{P_{2x}^2 + P_{2y}^2} > l_1 + l_2\,, \tag{2.9}$$

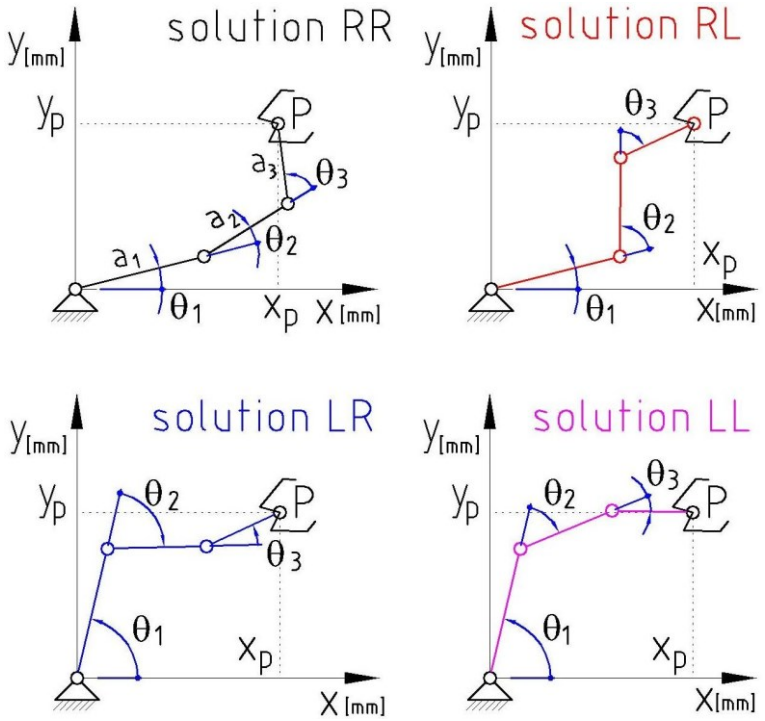
то тогава няма решение. В този случай точка P_2 е извън зоната на достижимост на двете звена.

Ако съществува поне едно решение, се проверява дали намерените ставни координати отговарят на ставните ограничения. Ставните ъгли, които удовлетворяват ставните ограничения определят ставните конфигурации, които могат да бъдат изпълнени от робота. Тогава определените от точки $P_0, P_{1r}, \dots, P_n$ и $P_0, P_{1l}, \dots, P_n$ конфигурации се запазват като решение на обратната задача.

Допълнително, решенията на обратната задача на кинематиката могат да се класифицират на няколко типа в зависимост от стойностите на ставните координати. Следващата секция описва тази класификация.

2.4.1 Класификация на решенията на обратната задача на кинематиката

Нека да разгледаме равнинен манипулатор с три звена с ненулеви дължини. В зависимост от знака на втората и третата ставна координата могат да се дефинират четири различни типа решения на обратната задача на кинематиката: Решение RR, когато $\theta_2 > 0$ и $\theta_3 > 0$. Решение RL, когато $\theta_2 > 0$ и $\theta_3 < 0$. Решение LR, когато $\theta_2 < 0$ и $\theta_3 > 0$. Решение LL, когато $\theta_2 < 0$ и $\theta_3 < 0$. Различните типове решения са показани на Фиг. 2.5.



Фиг. 2.5. Четири типа решения на обратната задача на кинематиката.

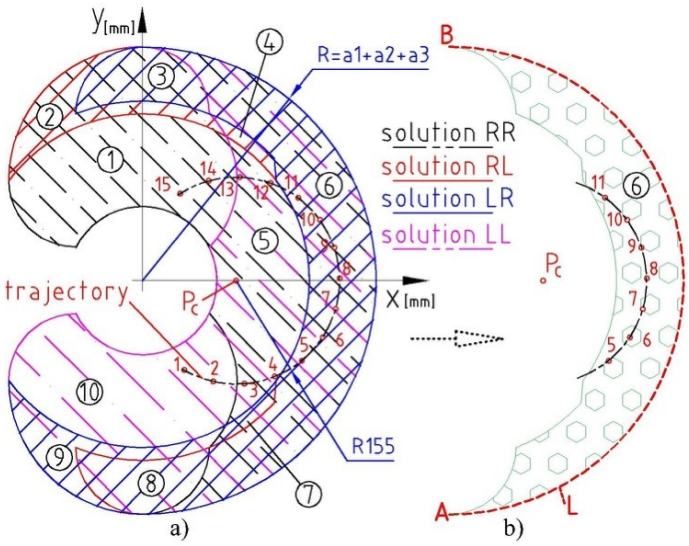
Четирите типа решения не са възможни за всяка точка от работното пространство. Преминаването от един тип решение към друг е от важно значение при планиране на движение на робота.

Изискванията към дължините на звената на изследвания робот са да покриват максимално работно пространство с цел по-лесно преодоляване на налични препятствия. Дължините на звената трябва да бъдат съобразени и с параметрите на задвижващите устройства. Затова при избора на дължините на звената трябва да се вземе предвид дължината на повечето сервомотори, тъй като във всяко звено трябва да може да се събере един сервомотор. Взимайки предвид описаните изисквания, нека с $l_i, i = 1, \dots, 4$ да означим дължините на звената на робота и да приемем, че изследваният робот има следните дължини на звената (2.10) и ставни ограничения (2.11):

$$l_1 = 150\ mm, l_2 = 100\ mm, l_3 = 100\ mm, l_4 = 0\ mm, \tag{2.10}$$

$$-\frac{\pi}{2} \leq \theta_i \leq \frac{\pi}{2}, \text{за } i = 1, \dots, 4. \tag{2.11}$$

Работното пространство на робота може да бъде разделено на различни зони, в зависимост от различните видове решения на обратната задача на кинематиката и ставните му ограничения. За разглеждания робот са налични десет различни зони (Фиг. 2.6).



Фиг. 2.6. Различните зони в работното пространство на робота а) обединението на всички зони b) сечението на всички зони.

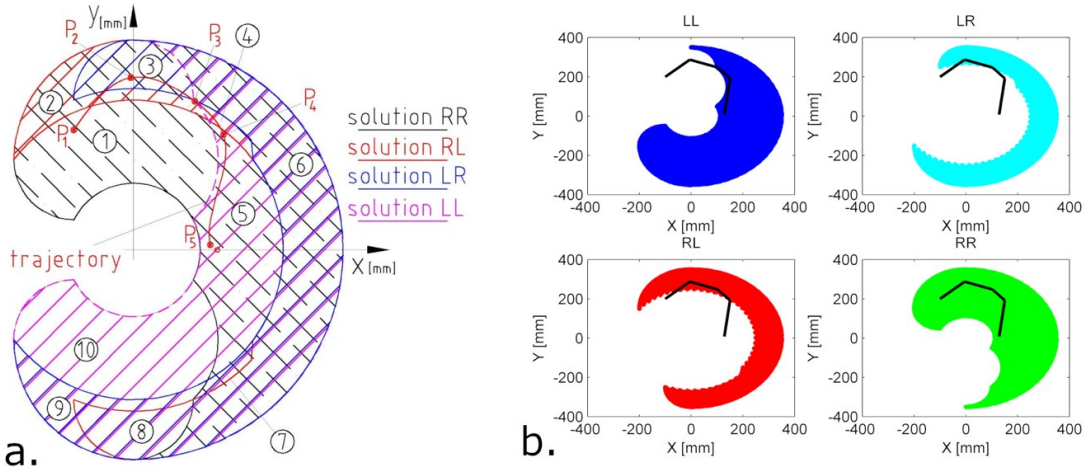
Обединението на всички зони представлява цялото работно пространство на разглеждания манипулатор (Фиг. 2.6.a). Сечението на всички зони представлява област от работното пространство, в която се намират и четирите различни вида ставни конфигурации (Фиг. 2.6.b). Цветовете на различните видове решения на обратната задача на кинематиката от Фиг. 2.5 съответстват на цветовете на зоните, в които се срещат от Фиг. 2.6. В Таблица 2.2 са представени типовете решения във всяка една зона от работното пространство на робота.

Таблица 2.2. Различните зони и типовете решения в тях.

Зона	Тип решение	Зона	Тип решение
1	RR	**6**	RR, RL, LR, LL
2	RR, RL	**7**	RR, LR, LL
3	RR, RL, LR	**8**	LL, LR, RL
4	RR, LL, RL	**9**	LL, LR
5	LL, RR	**10**	LL

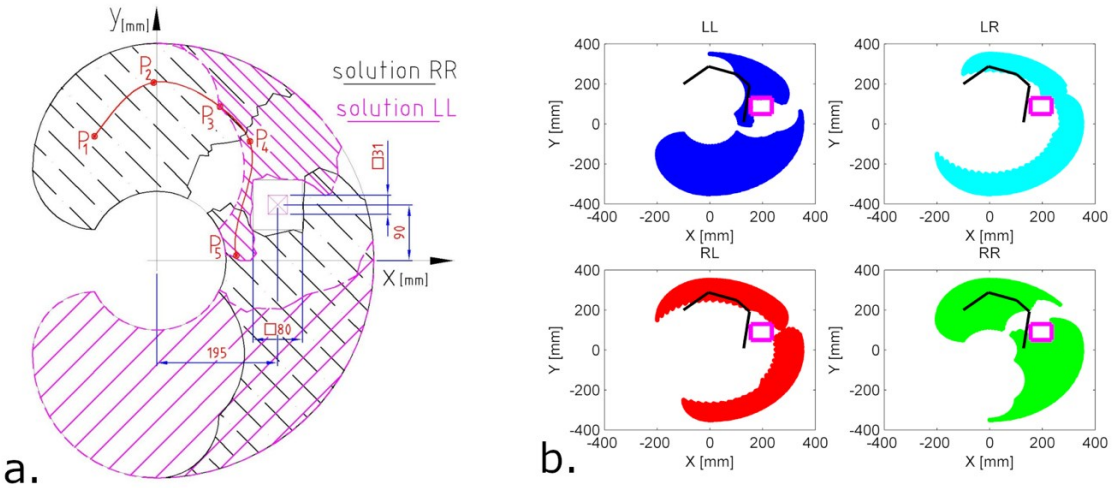
В работното пространство на робота съществуват траектории, за чието изпълнение се изисква смяна на вида решение на обратната задача на кинематиката. На Фиг. 2.6 е показана такава траектория. Роботът не може да извърши движение от точка 1 до точка 15 без да премине от един вид ставна конфигурация в друг. Това е така, тъй като точка 1 се намира в зона 10, а точка 15 в зона 1 и зоните 1 и 10 не се пресичат. Разбира се, ако желаното движение е изцяло в зона 1 или зона 10, то движението ще бъде изпълнено без смяна на типа решение, тъй като в тези зони съществува само един тип ставна конфигурация (Таблица 2.2).

Допълнително, ако в работното пространство има препятствия, те биха могли да накъсат тези зони. На Фиг. 2.7 е показано работното пространство на робота и различните типове решения на обратната задача на кинематиката, както и желана траектория от точка P_1 до точка P_5. Представена е позицията на желаното движение спрямо различните типове решения. От фигурата се вижда, че желаното движение се намира изцяло в зона с тип решение RR и може да бъде изпълнено без смяна на този тип.



Фиг. 2.7. Работно пространство на робота и желана траектория: a. работно пространство b. зоните на четирите типа решения и желана траектория.

Но освен ставните ограничения, дефинирани в (2.11), които определят работното пространство на манипулатора, в работното пространство е възможно да има препятствия. В зависимост от разположението и геометрията им, работната зона на манипулатора се променя. Нека да предположим, че имаме препятствие с квадратна форма и координати (195, 90). Дължина на страната 31 mm и допълнителни 49 mm за ширината на ставите, за да се избегне колизия с препятствието (Фиг. 2.8.a.).



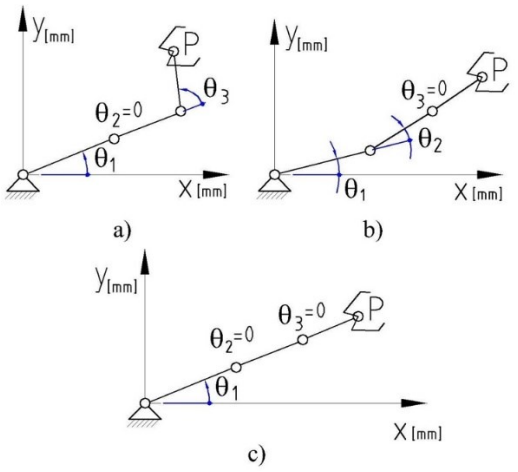
Фиг. 2.8. Работно пространство на робота при наличие на препятствие: a. позиция на препятствието в работното пространство; b. промяна в четирите типа решение.

Както се вижда от фигурата, зоните съдържащи тип решение RR и LL (Таблица 2.2) са разделени на две отделни части. Препятствието променя зоните на четирите типа решения на обратната задача на кинематиката (Фиг. 2.8.b.). Ако роботът трябва да изпълни движение с начална позиция от зона номер 1 и крайна точка в зона номер 5 (Фиг. 2.6), то това движение не би било възможно без смяна на типа решение. Ако смяната на типа решение се извърши в произволна ставна конфигурация, има вероятност хващача на робота да се отклони от желаното движение и да изпълни некоректно поставената задача. Затова е необходимо да се анализират преходните конфигурации и да се определи кои от тях са подходящи за промяна на типа решение на обратната задача на кинематиката.

2.4.2 Изследване на преходни конфигурации

За изследвания робот с допълнителни степени на свобода класифицирахме решенията на обратната задача на кинематиката на 4 типа, в зависимост от стойностите на ставните координати. Когато роботът трябва да изпълни зададено движение е възможно да се наложи да смени типа решение на обратната задача на кинематиката. При смяна на типа решение е възможно хващачът на робота да се отклони от желаната траектория. Това пък може да доведе до увеличение на времето за изпълнение на поставената задача или неточно изпълнение. Затова е необходимо да изследваме преходните моменти за разглеждания робот.

Взимайки предвид ставните ограничения (2.11) на разглеждания равнинен робот с допълнителни степени на свобода, може да се каже, че съществуват три типа конфигурации, при които се извършва смяна на типа решение на обратната задача на кинематиката (Фиг. 2.9). Такава преходна конфигурация имаме, ако $\theta_2 = 0$ (Фиг. 2.9.a.), ако $\theta_3 = 0$ (Фиг. 2.9.b.), или ако $\theta_2 = 0$ и $\theta_3 = 0$ (Фиг. 2.9.c.). При последната конфигурация съществува само едно единствено решение на обратната задача на кинематиката и позицията на хващача се намира на границата на работната зона на робота. Външната граница на работната зона е представена на Фиг. 2.6.b. с червена пунктирана дъга L от точка A до точка B.



Фиг. 2.9. Преходни конфигурации за равнинен робот с допълнителни степени на свобода.

Да се върнем отново на Фиг. 2.6.а. и да разгледаме зона 5. В тази зона имаме два вида възможни решения: тип LL и тип RR. Ако желаната траектория на хващача се намира изцяло в зона 5, то роботът няма да може да премине през преходна конфигурация и да промени типа на решение от LL на RR или обратно. В случай, че целевата траектория се намира в зона 2 или зона 9, манипулаторът може да премине през конфигурациия, при която $\theta_3 = 0$.

Ако пък хващачът изпълнява движение в зони 3, 4, 6, 7 или 8, но не на външната граница (на дъгата L), роботът може да премине през конфигурации, при които $\theta_2 = 0$ или $\theta_3 = 0$. При изпълнение на траектория, намираща се по дъгата L, роботът ще премине през конфигурация, при която $\theta_2 = 0$ и $\theta_3 = 0$.

Всяко преминаване през такъв тип конфигурации увеличава времето за изпълнение на задачата. Роботът е с допълнителни степени на свобода и това предоставя много възможности за избор на конкретен момент, в който да се промени текущия тип решение на обратната задача на кинематиката и да се премине през такъв тип особена конфигурация.

В някои от разгледаните зони изпълнението на определена ориентация на хващача в дадена точка от работното пространство може да бъде постигната само с един тип ставна конфигурация. Но има зони, в които ориентацията на хващача може да се изпълни с различни по тип ставни конфигурации. Допълнително, ако се налага промяна в ориентацията на хващача в същата точка, ще е необходимо роботът да смени типа решение. При тази смяна може да се наложи да се отклони от желаната точка. Затова е необходимо да се изследват множеството ориентации на хващача, които могат да бъдат реализирани, когато той е позиционирано в дадена точка от работното пространство. Параметърът, който описва тези ориентации се наричана ъгъл на сервис.

2.4.3 Ъгъл на сервис

Терминът ъгъл на сервис е въведен от Виноградов през 1971г.

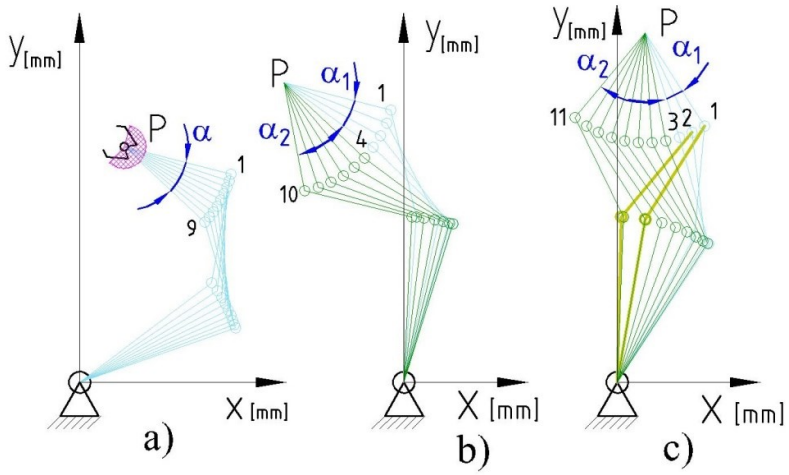
Дефиниция 2.1

Ъгъл на сервис на манипулатор в точка от неговото работно пространство наричаме ъгъла, който съдържа в себе си всички възможни ориентации на хващача на манипулатора, чрез които може да достигне до точката [91].

Четирите типа решения разделят работното пространство на робота на десет зони (Фиг. 2.6). В някои от зоните, като първа и десета съществува само един тип решение на обратната задача на кинематиката. Затова възможните ориентации на четвъртата става могат да бъдат изпълнени само от един тип решение. В останалите зони съществуват точки, които роботът може да достигне с изпълнителното си звено с една и съща ориентация, но с различен тип решение на обратната задача на кинематиката. В тези точки, ако хващачът на робота трябва да промени

ориентацията си, то е възможно да се отмести от точката, в която е позиционирано, при преминаване от един тип решение в друг. Поради тази причина е необходимо да разгледаме ъгъла на сервис в различните зони. Заради симетричността на работното пространство на робота е достатъчно да се изследва ъгълът на сервис за първите шест зони.

На Фиг. 2.10 са показани всичките възможни ориентации за позициониране на хващачът в точка P от зона 1. Третото звено може да се завърти на ъгъл α от позиция 1 до позиция 9. Четвъртото звено има дължина равна на нула и има ставни ограничения дадени в (2.11). Фиг. 2.10.a представя ъгъла на сервис на четвъртата става. Точка P може да бъде достигната само с решения от тип RR.



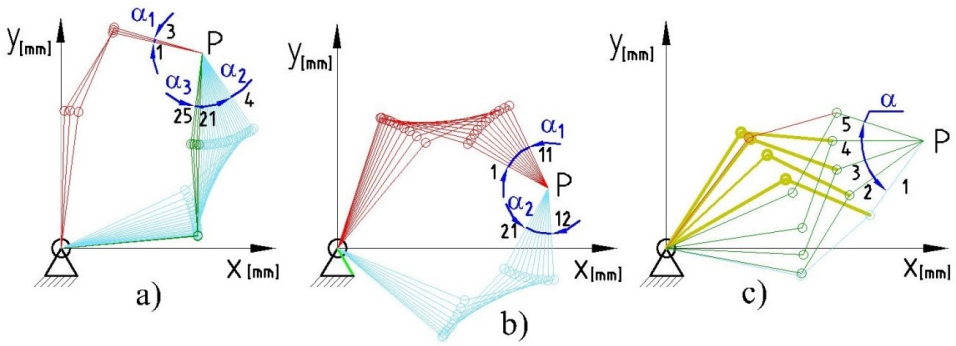
Фиг. 2.10. Различни ставни конфигурации, с които хващача на робота може да достигне до точка P в: a. зона 1; b. зона 2; c. зона 3.

На Фиг. 2.10.b. са изобразени възможните конфигурации за достигане на точка P в зона 2. В тази зона съществуват два типа решения на обратната задача на кинематиката RR и RL. Преходът между двата типа решения настъпва около конфигурация 4. В тази конфигурация звената l_2 и l_3 застават в една линия и роботът преминава през преходна конфигурация при промяна на типа решение. Ъгълът на сервис може да се представи като сума на ъглите α_1 и α_2. В тази зона хващачът на робота може да промени своята ориентация без да е необходимо отместване от точка P.

Ориентацията в зона 3 е показана на Фиг. 2.10.c. В тази зона съществуват три типа решения: RR, RL и LR. Хващачът на робота може да промени своята ориентация без да е необходимо да се отмести от точка P.

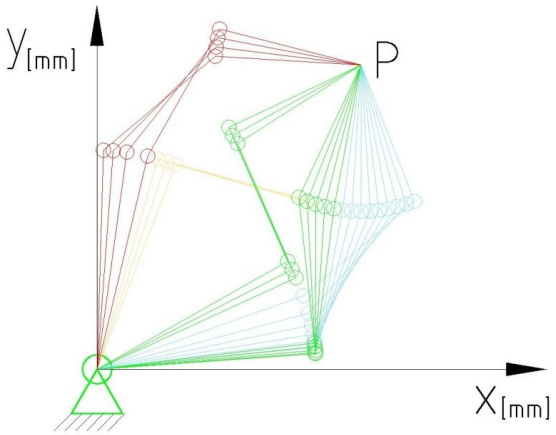
Ако точка P е в зона 4 са възможни 3 типа решения на обратната задача на кинематиката: RR, LL и RL. Както се вижда от Фиг. 2.11.a., различните ъгли на хващача са групирани в две зони. Тук, за да се премине от една зона в друга е необходимо да се напусне точка P. Това е

важно условие, което трябва да се има предвид при планиране на движения. В зона 5 (Фиг. 2.11.b.) ситуацията е аналогична на зона 4. Тук съществуват два типа решения: LL и RR.



Фиг. 2.11. Различни ставни конфигурации, с които хващача на робота може да достигне до точка P в: a. зона 4; b. зона 5; c. зона 6.

В зона 6 съществуват всички типове решения (Фиг. 2.11.c). В зависимост от позицията на точка *P* в зона 6 е възможно, всички възможни конфигурации да формират непрекъснат интервал. По този начин би била възможна промяна в ориентацията на хващача без отместване от точка P. Също така, в тази зона има точки, за които решенията на обратната задача на кинематиката не образуват непрекъснат интервал. В този случай, хващачът на робота трябва да се отмести от точка *P*, за да премине от един тип решение в друг (Фиг. 2.12).

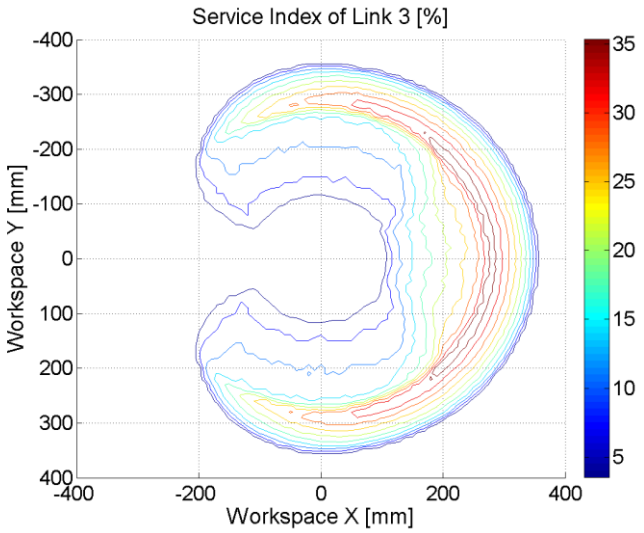


Фиг. 2.12. Различни ставни конфигурации, с които хващачът на робота може да достигне до точка P в зона 6.

Ако хващачът на робота е позициониран в определена позиция и трябва да промени само своята ориентация, то трябва да се вземат предвид направените разсъждения и да се избере такава ставна конфигурация, при която се променя ориентацията и се минимизира отместването от целевата точка. Ако целта е роботът да приложи повече сила или по-голяма скорост, то трябва да се разгледа коефициентът на манипулативност. Следващата секция изследва коефициента на манипулативност в работното пространство на робота.

2.5 Коефициент на манипулативност

Може да се каже, че ако обхватът на движение на четвъртата ротационна става е равен или по-голям от ъгъла на сервис изчислен за първите три стави, то за всяка желана позиция и ориентация може да бъде избрана конфигурация с най-голям коефициент на манипулативност. Може да се приеме, че максималният обхват на ъгъла на сервис е 2π. Под коефициент на сервис за определена точка от работното пространство на даден манипулатор се разбира отношението на ъгъла на сервис в тази точка към максималния обхват на ъгъла на сервис от 2π. На Фиг. 2.13 са представени стойностите на коефициента на ъгъла на сервис за първите три ротационни стави (с координати $\theta_1, \theta_2, \theta_3$) за всяка точка от работното пространство на разглеждания робот. Коефициентът на ъгъла на сервис е представен в проценти. От Фиг. 2.13 се вижда, че зоните, които са близо до границата на работното пространство имат по-нисък коефициент на ъгъл на сервис.



Фиг. 2.13. Ъгъл на сервис на равнинен робот с допълнителни степени на свобода.

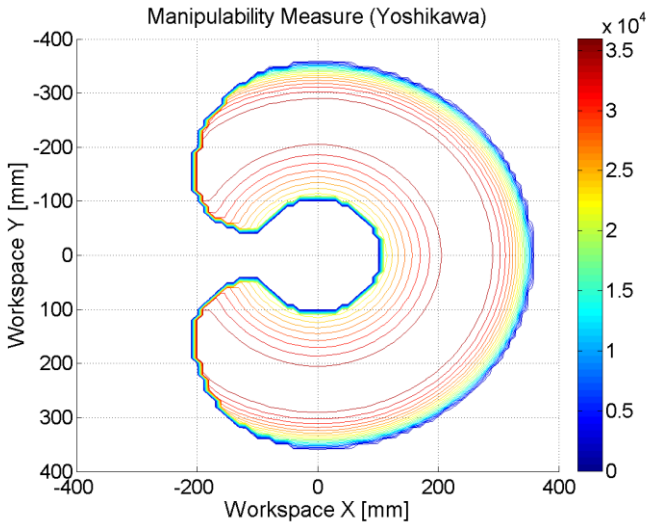
Зоната с най-голям ъгъл на сервис има стойност 36%. Обхватът на θ_4 е π радиана. Това означава, че за всяка позиция от работното пространство на робота и за всяка ориентация, която е в обхвата на ъгъла на сервис, на хващача на робота може да се избере ставна конфигурация с най-голям коефициент на манипулативност.

Понятието манипулативност (manipulability) е въведено от Йошикава през 1985 година [39]. Метриката коефициент на манипулативност позволява да се избере такава ставна конфигурация, при която роботът би могъл да извърши желаната задача с най-голяма ставна скорост. Изчислението на коефициента на манипулативност μ се базира на матрицата на Якоби $\boldsymbol{J}(\boldsymbol{\theta})$. Матрицата на Якоби е от важно значение при анализа на роботизирани системи. Описва връзката между ставните скорости и съответните линейни и ъглови скорости на изпълнителното звено и е функция на ставните координати.

Ако роботът е с допълнителни степени на свобода, то коефициентът на манипулативност се изразява като корен квадратен от произведението на матрицата на Якоби по нейната транспонирана матрица $\mu = \sqrt{det\, \boldsymbol{J}(\boldsymbol{\theta})\boldsymbol{J}^{\boldsymbol{T}}(\boldsymbol{\theta})}$.

Ако роботът не е с допълнителни степени на свобода, то матрицата на Якоби е квадратна матрица и коефициентът на манипулативност μ е равен на абсолютната стойност на детерминантата на матрицата на Якоби: $\mu = |det\, \boldsymbol{J}(\boldsymbol{\theta})|$ [1].

Да разгледаме коефициентът на манипулативност в работното пространство на изследвания равнинен манипулационен робот с допълнителни степени на свобода. Роботът има ставни ограничения, дефинирани в (2.11) и размери на звената (2.10). Отново поради факта, че четвъртата ротационна става влияе единствено на ротацията на хващача, няма да бъде взета предвид в изследването. На Фиг. 2.14 са показани зоните с различни стойности за всяка позиция от работното пространство на равнинния робот на коефициента на мобилност. Тъй като роботът е с допълнителни степени на свобода относно произволно равнинно движение, то за всяка една позиция от работното пространство, съществува множество от ставни конфигурации. За всяка една позиция от работното пространство е визуализирана максималната възможна стойност за коефициента на манипулативност.

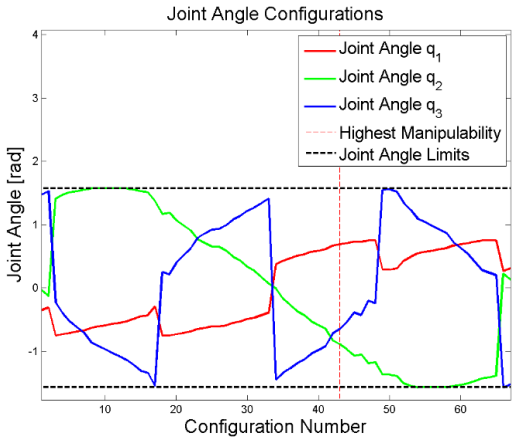


Фиг. 2.14. Максимална стойност на коефициента на манипулативност за всяка точка от работното пространство на равнинен робот с допълнителни степени на свобода.

Както се вижда от фигурата, роботът има по-голям коефициент на манипулативност в центъра на неговото работно пространство и по-ниска по границата на работното пространство. Това означава, че във вътрешните зони от работното пространство, роботът може да развие по-високи ставни скорости отколкото по границите на своето работно пространство.

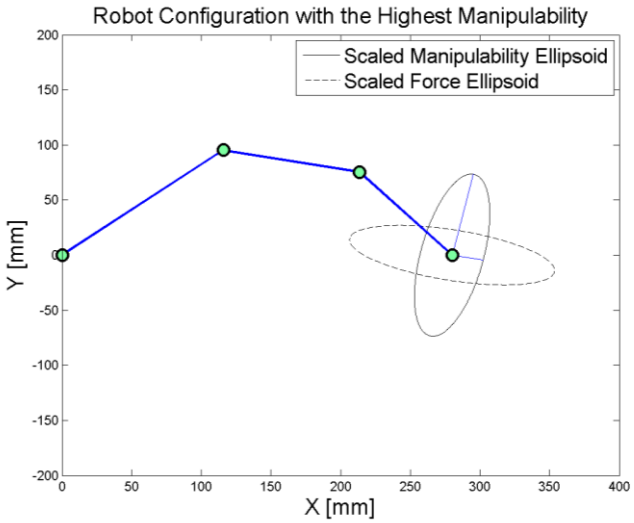
Нека да разгледаме точка с координати (280 mm, 0 mm). Тази точка е близка до центъра на работното пространство на робота. Затова има голям ъгъл на сервис и висок коефициент на

манипулативност. На Фиг. 2.15 са изобразени различните ставни конфигурации на робота за позициониране на желаните координати. Всяка конфигурация има различни стойности на ставните ъгли $\theta_1, \theta_2, \theta_3$. Червеният пунктир показва конфигурацията с най-висок коефициент на манипулативност. Както се вижда от Фиг. 2.15, за да се достигне тази позиция е необходимо ставните координати θ_2 и θ_3 да имат стойности, близки до своите ставни ограничения. Това означава, че при изчислението на коефициента на манипулативност взима предвид целия обхват на движение на тези две стави.



Фиг. 2.15. Стойности на ставните ъгли при различните конфигурации за позиция с координати (280, 0).

На Фиг. 2.16 е показана конфигурацията с максимална стойност на коефициента на манипулативност.



Фиг. 2.16. Ставна конфигурация с най-висок коефициент на манипулативност за позиция с координати (280, 0).

Изобразени са също и два елипсоида за дадената позиция. Първият показва посоките, в които манипулаторът може да развие по-голяма скорост (scaled manipulability ellipsoid). Вторият елипсоид обозначава посоките, в които роботът би могъл да приложи по-голяма сила (force

ellipsoid). Както се вижда от фигурата, зоните, в които роботът може да приложи повече сила, не може да развие голяма скорост и обратно. Симетричната конфигурация на манипулатора относно оста X ще има същите характеристики поради конструкцията на робота.

2.6 Приноси към Глава 2

Разработен е подход за класифициране по тип на решенията на обратната задача на кинематиката. Този подход открива 4 вида решения на обратната задача на кинематиката в случай на равнинен манипулатор с разглежданата структура (Фиг. 2.1). Получават се 4 не пресичащи се множества в конфигурационното пространство на робота. С правата задача на кинематиката тези множества се трансформират в реалното пространство, където разделят работното пространство на робота на пресичащи се множества (Фиг. 2.6). Броят и видът на тези множества зависи от много фактори: размери, ставни ограничения, препятствия и др. Множествата са не пресичащи се в смисъл на това, че липсва гладко движение на манипулатора, при което хващачът запазва положението си в работното пространство и същевременно манипулаторът променя конфигурацията, принадлежаща на една група решения в конфигурация, принадлежаща на друга група решения. В литературата липсва информация относно класифициране на типовете решения на обратната задача на кинематиката за роботи с допълнителни степени на свобода. Те са важни при планиране на траектория, тъй като роботът минава през преходна конфигурация, когато извършва смяна на типа решение. Това може да доведе до отклонение от желаната траектория, увеличаване на времето за изпълнение или друг проблем свързан с управлението на робота. Затова е от съществено значение знанието за това в кои области от работното пространство на робота кои типове решения се срещат. Тъй като роботът е с допълнителни степени на свобода има голям избор от ставни конфигурации, сред които може да се избере най-подходящата за смяна на типа решение.

Според наличните типове решения на обратната задача на кинематиката, работното пространство може да се раздели на различни зони. При планиране на движение от точка до точка или изпълнение на зададена траектория трябва да се имат предвид не само ставните ограничения на робота, но и ограниченията породени от налични препятствия в работната зона. Препятствията биха могли да накъсат по такъв начин работната зона на робота, че изпълнението на зададеното движение да стане по-трудно или дори невъзможно. Затова е важно да знаем как се променят зоните в работното пространство на робота при наличие на препятствия.

Изследванията представени в тази Глава 2 са публикувани в публикации [П1], [П2], [П4] и докладвани на [Д1], [Д3].

2.7 Заключение

Правилното моделиране на даден робот е от важно значение при разработването на роботизирани системи и е необходимо за управлението на дадения робот. Тази глава представи кинематичен анализ на равнинен робот с допълнителни степени на свобода. Правата задача на кинематиката е решена, посредством конвенцията на Денавит-Хартенберг, а за обратната задача на кинематиката е използван геометричен метод за решение. Тъй като, роботът е с допълнителни степени на свобода, обратната задача на кинематиката има множество решения. Затова при планиране на движение е необходимо да се избере най-подходящата ставна конфигурация в зависимост от поставената задача. Разгледаната метрика на Йошикава, коефициент на манипулативност, показва при кои ставни конфигурации роботът би могъл да развие по-голяма скорост. Допълнително, решенията на обратната задача на кинематиката могат да се разделят на няколко типа в зависимост от стойността на ставните ъгли. Когато роботът трябва да премине от един тип решение в друг, може да е необходимо да се отклони от желания път. Затова планирането на моментите, в които роботът може да промени типа решение е от важно значение. Също така, при наличие на препятствие в работната зона на разглеждания робот, зоните дефинирани от различните типове решения на обратната задача на кинематиката се накъсват и това би могло да доведе до невъзможност за изпълнение на дадена траектория. В тази глава са представени резултатите от изпълнението на първата задача от задачите по дисертационния труд.

Следващата глава представя алгоритми за управление на движението на равнинен робот с допълнителни степени на свобода при наличие на статични или динамични препятствия в работната му зона.

3. Управление на равнинен антропоморфен робот

Основната цел на управлението е да се намерят и зададат подходящи команди (позиции) към задвижващите устройства. Задачата на задвижващите устройства е да позиционират ставните координати по такъв начин, че хващачът на робота да достигне желаната позиция и ориентация. За тази цел е необходима хардуерна и софтуерна система, която да извършва необходимите изчисления и да задвижва робота, както и подходящ алгоритъм за планиране на движение. При планирането на движение трябва да се вземат предвид както ограниченията на робота, така и наличните в работното пространство статични или динамични препятствия. Алгоритъмът за планиране на движение ще даде като резултат позициите, които хващачът на робота трябва да достигне. След това, ще се извършат необходимите изчисления, за да се намерят ставните координати на робота, които удовлетворяват тези позиции. Накрая, ще се изпратят необходимите сигнали към задвижващите устройства за изпълнение на желаното движение. Тази глава представя необходимия хардуерен и софтуерен дизайн за управление на антропоморфен модел на робот с допълнителни степени на свобода, както и предлага методи за планиране на движение при наличие на статични и динамични препятствия.

3.1 Хардуер за управление на проектирания робот

Производителността и бързодействието на една роботизирана система са силно зависими от нейната хардуерна система. Хардуерната система предоставя необходимите изчислителни ресурси на софтуерната система. Отговаря за задвижването на робота и би могла както да му даде повече възможности, така и да го ограничи. Допълнително, за някои изчисления, като решаване на правата задача на кинематиката или обработката на определени данни, би било от значение дали се извършват в реално време. Затова изборът на хардуерни компоненти и комуникацията между тях до голяма степен зависи от задачите, които роботът ще изпълнява и хардуерните изисквания към тях. Тази глава разглежда хардуерните изисквания към управляващата електроника и задвижващите устройства на разглеждания робот, както и представя избраните хардуерни компоненти, комуникацията между тях и проектираната хардуерна архитектура.

3.1.1 Хардуерни изисквания

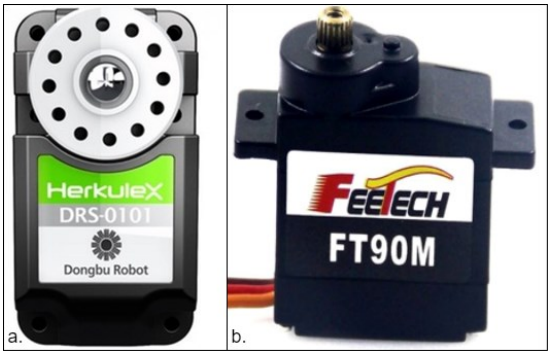
Необходимо е роботът да може да изпълнява предварително зададено движение или последователност от движения. За тази цел роботът се нуждае от хардуерна система, която да управлява задвижването на робота. Изискванията към хардуерните компоненти са следните:

- Управляващият контролер трябва да разполага с необходимите изчислителни ресурси за пресмятане на правата и обратната задача на кинематиката. За решаване на правата и обратната задача на кинематиката ще се използват описаните методи в Глава 2.
- Управляващата електроника е необходимо да разполага и с достатъчно памет, за да позволява зареждане на последователност от позиции, които роботът трябва да изпълни с изпълнителното си звено.
- Управляващата електроника е необходимо да е базирана на платформи с отворен код и удобна среда за разработка.
- Необходимо е роботът да позволява управление както през USB порт, така и по Wi-Fi връзка. За тази цел ще е необходим модул, който да предоставя възможност за безжично свързване с робота.
- Управляващият контролер трябва да разполага с поне 3 серийни UART порта. Един за комуникация със задвижващите устройства, един за модула, който ще осигурява безжичната комуникация с робота и един за UART комуникацията с компютъра на оператора.
- Управляващата електроника трябва да разполага със светлинни индикатори за обозначаване на състоянието на моторите (включени/изключени), дали управляващият контролер има захранване и дали е осъществена безжична връзка с робота.
- Задвижващите устройства трябва да позволяват 180 градусово движение на ротационните стави и управление по позиция.
- Тъй като равнинният робот с допълнителни степени на свобода ще се използва за научноизследователски и учебни цели, проектираната хардуерна система за неговото управление трябва да позволява разширяване и подобряване. Това ще позволи изследване и сравняване на различни хардуерни компоненти за управление на робота.

Следващата глава представя избраните хардуерни компоненти и проектираната хардуерна система, която отговаря на описаните изисквания.

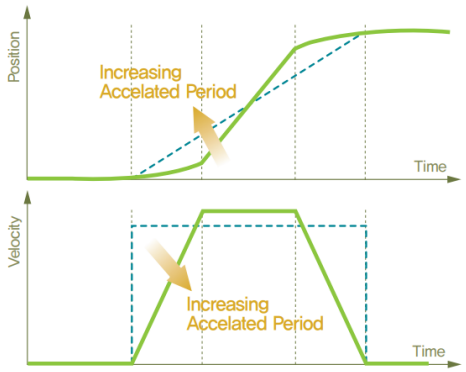
3.1.2 Хардуерни компоненти

Избраната електроника за управление на робота се базира на платформата **Arduino** и **Wi-Fi модула WeMos D1 ESP8266**. Необходимо е ротационните стави на робота да се управляват по позиция, затова за тяхното управление са избрани **умни сервомотори HerkuleX DRS-0101** (Фиг. 3.1.a.). За управлението на хващача и транслационния механизъм се използват **сервомоторите FeeTech FT90M** (Фиг. 3.1.b.).



Фиг. 3.1. Задвижващи устройства: a. Умен сервомотор HerkuleX DRS-0101 b. Сервомотор FeeTech FT90M.

Четирите ротационни стави се задвижват от умни сервомотори: **HerkuleX DRS-0101**. Те се управляват, чрез серийна комуникация и могат да върнат информация за текущата им позиция и скорост. Позволяват плавен контрол на позицията им. Когато моторът HerkuleX DRS-0101 получи команда за дадено движение, автоматично създава трапецовиден профил на скоростта при управление по позиция (Фиг. 3.2). Моторите поддържат и други алгоритми за управление, като PID и управление с обратна връзка, но по подразбиране се използва трапецовиден профил на скоростта. Това позволява прецизно изпълнение на движението.



Фиг. 3.2. Трапецовиден профил за управление на скоростта [92].

Използването на UART комуникация позволява лесна промяна на скоростта, позицията, цвета на LED индикатора, управление едновременно до 254 сервомотора. Умните сервомотори поддържат и до седем различни типа грешки. За наличието на някоя от тези грешки се съобщава, чрез LED индикатора. Моторите са подходящи за управление на роботи, механични ръце и др. Тяхното работно напрежение е 7.4V и имат ъгъл на завъртане 320 градуса [92]. Ротационните стави на разглеждания робот могат да се завъртат на 180 градуса. Затова е необходима редукция на предавката от 1:0.5625, за да може да се използва пълният обхват на движение на моторите. Допълнително, редукцията гарантира, че ограниченията на ставните координати няма да бъдат нарушени. В Таблица 3.1 са представени основните характеристики на сервомотора HerkuleX DRS-0101.

Таблица 3.1. Характеристики на сервомотор HerkuleX DRS-0101.

Работно напрежение	7.4 V
Максимална скорост	0.166 s/60° (7.4 V)
Въртящ момент (Stall torque)	12 kg.cm (7.4 V)
Ъгъл на завъртане	320°
Обратна връзка	Позиция, скорост, температура, напрежение.
Алгоритми за управление	Трапецовиден профил на скоростта, PID, управление с обратна връзка и др.
Вид комуникация	Серийна асинхронна комуникация

Транслационният механизъм и хващача на робота се задвижват от **мини сервомотори Feetech FT90M**. Те нямат обратна връзка, както и контрол на скоростта и се управляват от широчинно-импулсни сигнали (ШИМ, PWM). ШИМ е техника за управление на аналогови и цифрови схеми. ШИМ използва правоъгълна импулсна вълна, чиято ширина на импулса е модулирана, което води до промяна на средната стойност на формата на вълната. Работното им напрежение е 4.8-6 V [93]. В Таблица 3.2 са представени основните характеристики на сервомотор FeeTech FT90M.

Таблица 3.2. Характеристики на сервомотор FeeTech FT90M.

Работно напрежение	4.8-6 V
Максимална скорост	0.07 s/60° (6 V)
Въртящ момент (Stall torque)	2.15 kg.cm (6 V)
Ъгъл на завъртане	180°
Обратна връзка	Няма
Вид комуникация	Широчинно-импулсна модулация (PWM)

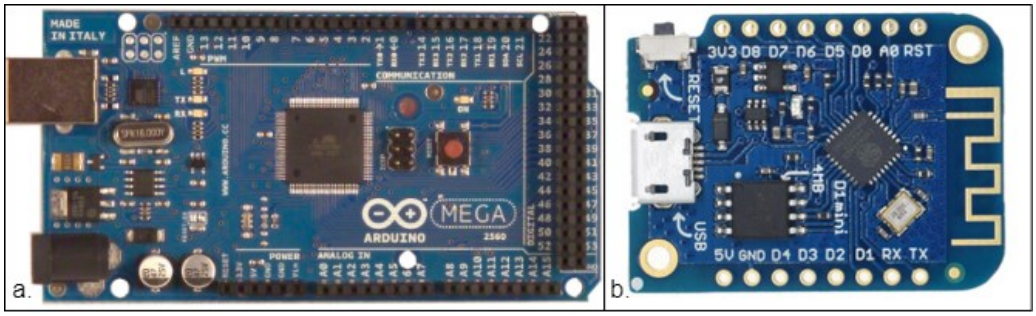
За управлението на робота са избрани микроконтролера Arduino Mega 2560 и Wi-Fi модула WeMos D1 mini. **Arduino Mega 2560** (Фиг. 3.3.а.) разполага с 256 KB памет, 8 KB SRAM. Той е микроконтролер, базиран на микроконтролера ATmega2560. Има 54 цифрови входни/изходни пина, 16 аналогови входа, позволява USB свързване и има бутон за рестартиране. Arduino Mega може да комуникира с компютър, друго Arduino или друг микроконтролер. Разполага с 4 хардуерни серийни UART порта. Програмирането му се извършва, чрез безплатната среда на Arduino посредством USB свързване, без да са необходими допълнителни хардуерни устройства [94]. Този микроконтролер е подходящ за управление на равнинен робот с допълнителни степени на свобода, тъй като за неговото задвижване са необходими 3 серийни UART порта. Един за комуникация с умните сервомотори, един за Wi-Fi модула и един за UART комуникацията с компютъра на оператора. Допълнително, микроконтролерът може да генерира PWM сигнали за

управление на сервомоторите на хващача и транслационния механизъм. Микроконтролерът има всички необходими изчислителни ресурси за пресмятане на правата и обратната задача на кинематиката, които са необходими за изпълнение на дадено движение и правилно позициониране на хващача. Arduino Mega 2560 е достатъчно, ако се подават команди на робота от компютър. Таблица 3.3 представя основните характеристики на избрания контролер.

Таблица 3.3. Характеристики на микроконтролер Arduino Mega 2560.

Микроконтролер	ATmega2560
Цифрови пина за вход/изход	54
Аналогови входа	16
Хардуерни серийни порта (UARTs)	4
EEPROM	4 KB
SRAM	8 KB
Флаш памет	256 KB, от които 8 KB са заети от bootloader
Решаване на обратната задача на кинематиката	Само офлайн

Wi-Fi модулът WeMos D1 mini (Фиг. 3.3.b.) позволява на робота да се подават команди през WebSocket, както и предоставя удобен за програмиране и демонстрация уеб базиран графичен потребителски интерфейс. Wi-Fi модулът позволява на робота да получава команди, чрез WebSocket комуникация и може да бъде използван като HTTP сървър за разработка на уеб базиран потребителски интерфейс. WeMos D1 Mini е малко Wi-Fi устройство, базирано на ESP8266EX чип. Разполага с 4 MB памет, работното напрежение е 3.3 V и входно напрежение 7-12 V [95]. Може да се препрограмира през Wi-Fi връзка.



Фиг. 3.3. Управляваща електроника: а. микроконтролер Arduino Mega 2560 b. Wi-Fi модул WeMos D1 mini.

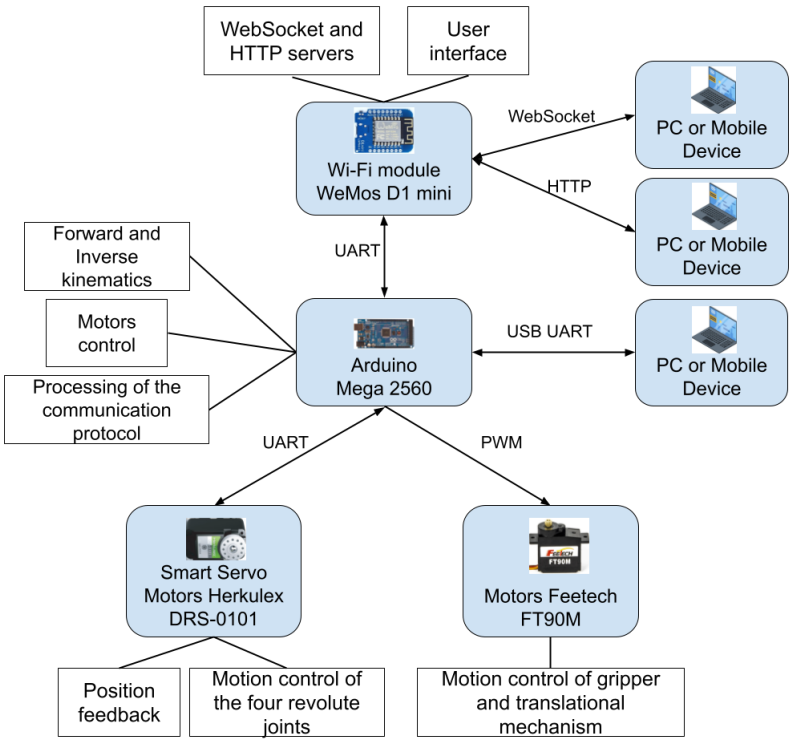
Сервомоторите на робота изискват захранване на 7.5 волта. Захранването може да се свърже директно към 2.1 mm букса на Arduino Mega 2560. Ако платката се захранва само през USB порта, то четирите умни мотора няма да работят.

Управляващата електроника разполага с три основни светлинни индикатора:

- червен: свети постоянно, когато платката има захранване;
- зелен: свети постоянно, когато моторите са включени. Когато е изгасен, те могат да се движат свободно. При изгасен индикатор роботът няма да изпълнява команди за движение;
- син: намира се на Wi-Fi модула. Когато този индикатор свети постоянно, то има отворен WebSocket към робота. В режим на безопасност при затваряне на последната отворена връзка към WebSocket-а, роботът автоматично спира моторите. Възможно е това да се изключи, в такъв случай индикаторът ще мига на половин секунда, когато няма активна връзка към робота.

3.1.3 Хардуерна архитектура и комуникация

Начинът на комуникация на описаните хардуерни компоненти е показан на Фиг. 3.4.



Фиг. 3.4. Хардуерна архитектура.

Както вече споменахме, три от хардуерните UART порта на микроконтролера Arduino се използват за комуникация със сервомоторите DRS-0101, с Wi-Fi модула и с компютъра на оператора. UART интерфейсите позволяват двупосочна комуникация. Умните сервомотори са свързани на същия комуникационен канал. Те отговарят за задвижването на четирите ротационни стави и връщат информация за текущата позиция. Малките сервомотори се управляват, чрез генерираните ШИМ сигнали от микроконтролера Arduino. Този тип на свързване е еднопосочен, моторите не връщат информация за тяхната позиция.

Микроконтролерът Arduino Mega отговаря за пресмятането на правата и обратната задача на кинематиката, за управлението на моторите и за обработка на комуникационния протокол.

Wi-Fi модулът осигурява както WebSocket, така и HTTP сървър. WebSocket действа като прокси и препраща заявки и отговори от и към платката на Arduino. HTTP сървърът обслужва разработения графичен уеб-базиран интерфейс. Този интерфейс използва връзката WebSocket за управление на робота. Също така, роботът може да бъде управляван директно, чрез USB свързване между компютър и микроконтролера Arduino.

Протоколът за комуникация се състои от текстови команди. Командите към Arduino Mega 2560 трябва да започват с главна латинска буква C и да завършват със символ (;) (Фиг. 3.5). Този вид команди могат да бъдат подавани през USB или през WebSocket. Командите, които се отнасят към Wi-Fi модула започват с главна латинска буква D и завършват на (;). Този вид команди не могат да се подават през USB, а само през WebSocket. Върнатият отговор на командите започва с главна латинска буква R и завършва на (;).

```
 *  CTN; -> Torque On
 *  CTF; -> Torque Off
 *  CM[<ID><POSITION_4_BYTES>]+T<TIME_4_BYTES>; -> Set axis <ID> to <POSITION>;
position is in encoder ticks
 *  CP;  -> Get motor Positions
RP1<POSITION_4_BYTES>2<POSITION_4_BYTES>3<POSITION_4_BYTES>4<POSITION_4_BYTES>;
 *  CS;  -> Get Status: 1 -> Torque ON; 2 -> Torque Off
RS1; or RS2;
 *  CE;  -> Get End effector
RE<POSITION_3_BYTES><POSITION_3_BYTES>;
 *  CG<TRANSLATION_3_BYTES><GRIPPER_3_BYTES>;  -> Grab: set translation and set
gripper
 *  CC[<ID><CALIBRATION_OFFSET]+; -> Set calibration offsets
```

Фиг. 3.5. Примерни команди към Arduino Mega 2560.

Отговорите на командите се връщат както през USB, така и през WebSocket. Отговорът се изпраща до всички клиенти през WebSocket-а, както и през USB.

След като са избрани хардуерните компоненти, удовлетворяващи хардуерните изисквания и е създадена хардуерната система за управление на робота е необходимо да се проектира подходяща софтуерна система. Следващата глава разглежда софтуерните изисквания към тази система и нейното реализиране.

3.2 Софтуерен дизайн за управление на равнинен робот

Както проектирането на хардуерната система, така и проектирането на софтуерната система зависи от задачите, които роботът трябва да изпълни и изискванията към него. Софтуерната система отговаря за обработката на получените данни от задвижващи устройства или сензори, както и за начина, по който ще се решат правата и обратната задача на кинематиката. Предоставя

потребителски интерфейс, чрез който операторът (потребителят) би могъл да управлява робота или да следи текущото състояние на ставните координати и моторите. Тази глава представя изискванията към софтуерната система, както и проектираната софтуерна система за управление на робота.

3.2.1 Софтуерни изисквания

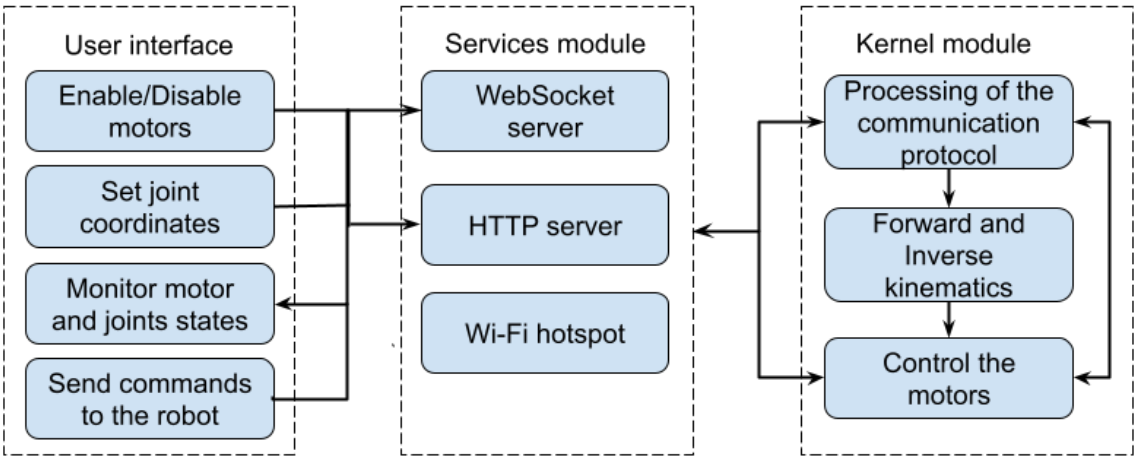
Основната цел на робота е да може да изпълни предварително зададено движение или множество от движения. За тази цел трябва да се разработи софтуер, който отговаря на следните изисквания:

- За да може роботът да изпълнява дадено движение и да позиционира правилно изпълнителното си звено, то софтуерът трябва да може да решава правата и обратната задача на кинематиката коректно и бързо.
- Позволява на потребителя да управлява робота, да включва или изключва моторите, да задава желана позиция в декартови или обобщени координати, да задава последователност от точки за определено движение.
- Софтуерната система трябва да предоставя комуникационен протокол за комуникация с робота и възможност за следене на върнатите отговори от робота.
- Софтуерната система трябва да позволява на потребителя да следи параметрите на робота – текущата позиция на ставните координати, хващача и транслационния механизъм.
- Софтуерната система трябва да предоставя и потребителски интерфейс, който може да бъде както графичен, така и текстови.
- Проектираната софтуерна система трябва да позволява добавяне на нови компоненти и функционалности. Това би позволило разработената роботизирана система да се използва за различни образователни, изследователски и индустриални задачи.

Следващата глава представя проектираната софтуерна система за управление на изследвания робот.

3.2.2 Софтуерна система за управление

Разработена е софтуерна система, която отговаря на описаните по-горе изисквания. Софтуерната система се състои от три основни модула: модул ядро (kernel module), обслужващ модул (services module) и графичен потребителски интерфейс (user interface). Първият модул се изпълнява върху микроконтролера Arduino, а останалите два върху Wi-Fi модула WeMos D1 mini. На Фиг. 3.6 е представена архитектурата на проектираната софтуерна система.



Фиг. 3.6. Архитектура на софтуерната система за управление на равнинен робот.

Модулът на ядрото (kernel module) отговаря за обработката на подадените команди, чрез комуникационния протокол, както и решаването на правата и обратната задача на кинематиката. Модулът отговаря и за подаването на необходимите команди към моторите. Софтуерът за модула на ядрото е реализиран, посредством програмния език C++ и средата за програмиране Arduino IDE. Командите към моторите на робота могат да се изпратят, чрез USB свързване на компютъра и микроконтролерът или чрез WebSocket комуникация, за която отговаря обслужващият модул.

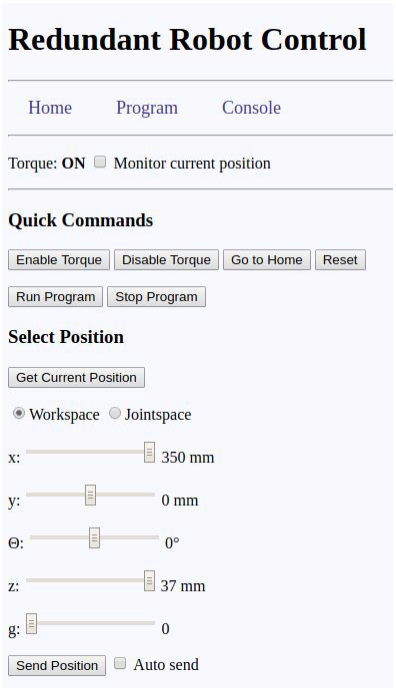
Обслужващият модул (services module) се изпълнява независимо от модула на ядрото и предоставя възможност за WebSocket и HTTP комуникация, както и Wi-Fi точка за достъп на робота. Този модул отговаря за осъществяване на връзката между потребителския интерфейс и модулът на ядрото.

Микроконтролерът и Wi-Fi модулът имат ограничена памет и изчислителна мощ. Намирането на възможните решения на обратната задача на кинематиката се случва на микроконтролера. Но решението на обратната задача изисква повече изчислителни ресурси, а използваният метод за решение на обратната задача на кинематиката е итеративен. Затова решението на обратната задача на кинематиката не се изпълнява за константно време. Това може да доведе до непредвидено време за изчисление. Ако е изпратена команда за една позиция към робота в декартови координати, то микроконтролерът пресмята решението в реално време. Ако е изпратена команда, съдържаща последователност от позиции, които роботът трябва да изпълни, микроконтролерът първо трябва да пресметне обратната задача на кинематиката за всяка точка от траекторията и след това роботът може да изпълни намерените позиции в реално време. Микроконтролерът има ограничена памет и само 256 позиции могат да бъдат препрограмирани. Времето за изпълнение на движение от една позиция до друга може да бъде зададено като параметър. Това позволява различно максимално време за изпълнение на препрограмираните траектории.

Както вече казахме, Wi-Fi модулът поддържа HTTP сървър, който позволява разработването на **графичен потребителски интерфейс (user interface)**. Потребителският интерфейс може да се достъпва от всеки уеб браузер, след като е осъществена връзка с робота посредством Wi-Fi точка за достъп. За реализацията на потребителския интерфейс са използвани само JavaScript и HTML/CSS.

Потребителският интерфейс се състои от няколко секции: навигационна, информационна, основни команди, програмиране и конзола. През **навигационната секция**, потребителят може да достъпва следните секции: основни команди, програмиране и конзола (Фиг. 3.7).

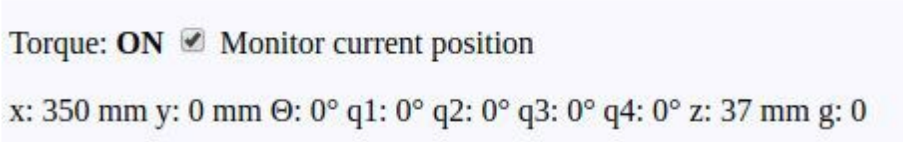
Секцията **основни команди** (Фиг. 3.7) е разделена на две подсекции: “Quick Commands” и “Select Position”. Подсекцията “Quick Commands” включва команди за включване и изключване на моторите, стартиране и спиране на зададена последователност от позиции към робота, изпращане на робота в начална позиция (350, 0). Подсекцията “Select Position” включва команди за получаване на информация за текущата позиция на робота. Налични са плъзгачи, които показват текущата позиция на робота и позволяват на потребителя да го управлява. Командите към робота могат да бъдат в ставни или декартови координати. Ако е активирана опцията “Auto send”, потребителят автоматично изпраща команди към робота, задавайки ги от плъзгачите.



Фиг. 3.7. Потребителски интерфейс: навигационна секция, информационна секция, основни команди и позиции.

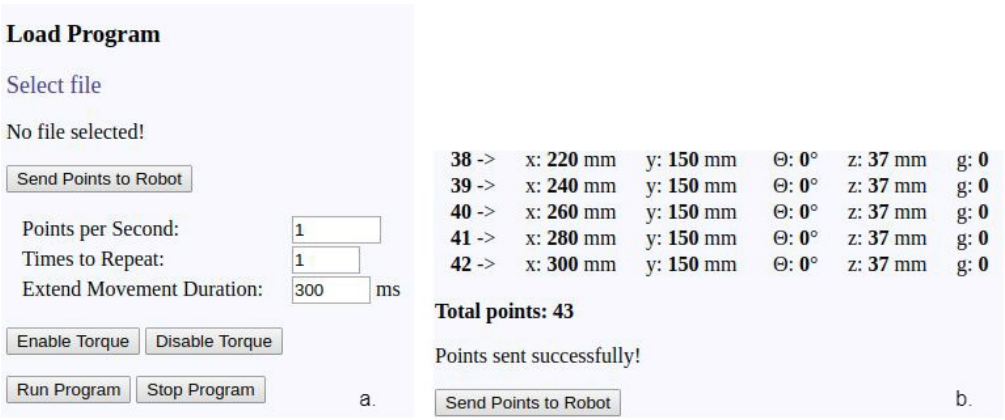
Информационната секция (Фиг. 3.8) дава информация за състоянието на моторите, дали са включени или изключени. Ако има отметка в полето на “Monitor current position”, потребителят може да следи текущата позиция на хващача. Допълнително, роботът връща информация за четирите ротационни стави, хващача и транслационния механизъм. Показаната

информация за x и y координатите се пресмята локално от модула на потребителския интерфейс, чрез решаване на правата задача на кинематиката.



Фиг. 3.8. Информационна секция.

Секцията **програмиране** позволява да се зареждат в паметта на робота до 256 предварително известни позиции (Фиг. 3.9.a.). Последователността от позиции се зарежда от текстов файл. Позициите могат да бъдат или в ставни, или в декартови координати. Ако позицията е зададена в ставни координати, то тя трябва да има 6 елемента: координати в градуси за четирите ротационни стави (от -90 до 90 градуса), позиция на транслацията (от 0 до 37 mm), затвореност на хващача число от 0 до 9. Ако е в декартови координати, то тя трябва да има 5 елемента: координати x и y в работното пространство (от -350 до 350 mm), ориентация на хващача в работната област (от -90 до 90 градуса), позиция на транслацията (от 0 до 37 mm), затвореност на хващача число от 0 до 9. Всяка позиция трябва да е на нов ред. Елементите на всяка позиция трябва да са разделени със запетая (,). Реалните числа трябва да са с десетична точка (.). В един файл може да има позиции и от двата вида. Позициите, зададени в декартови координати е възможно да попаднат извън работната област на робота. В такъв случай те няма да бъдат изпълнени от робота. От тази секция потребителят може да определя и колко позиции да бъдат изпълнени за една секунда и колко пъти да се изпълнят. Потребителят може и да удължава времето за движение. Това позволява изпълнението на плавно движение. След зареждане на текстов файл, намерените позиции се визуализират таблично (Фиг. 3.9.b.).

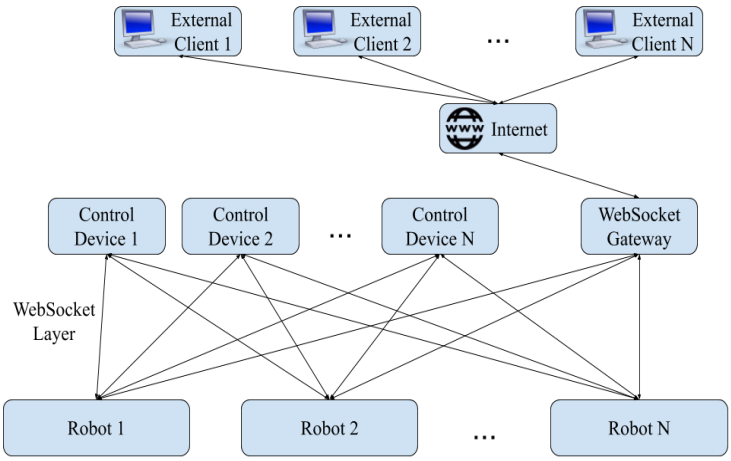


Фиг. 3.9. Секция програмиране: a. зареждане на текстов файл с позиции; b. визуализиране на заредени позиции.

Секцията **конзола** (Фиг. 3.10) позволява директно изпращане на команди към робота. Показват се и върнатите отговори, ако има такива. В полето “Command” се въвежда командата към робота, в голямото текстово поле се показват изпратените команди и получените отговори.

Фиг. 3.10. Секция конзола.

Допълнително, WebSocket [96] интерфейсът позволява създаване на множество от кооперирващи роботи. Както се вижда от Фиг. 3.11 множество роботи могат да бъдат управлявани от едно управляващо устройство. Освен това може да има и множество управляващи устройства за един робот.



Фиг. 3.11. Множество от кооперирващи роботи.

Също така, може да се добави допълнителен WebSocket възел, който би позволил на множество от външни устройства да управляват и следят множество от роботи през интернет.

Освен проектирането на хардуерна и софтуерна система, друг важен компонент от управлението на робота е планирането на неговото движение и коректното изпълнение на зададена траектория. Следващата глава представя метод за планиране на траектория, използвайки теорията на графите.

3.3 Приложение на теорията на графите за планиране на траектория

Обикновено задачата на индустриалните манипулатори е да изпълнят предварително дефинирано движение без отклонение от желаната траектория и за минимално време. Затова и времето за изпълнение на дадено движение или зададена траектория е важна част от управлението на роботизираните системи. Колкото по-бързо и точно роботът може да изпълни зададената му траектория, толкова по-голяма ще бъде и неговата продуктивност.

Поради ограниченото ставно пространство на разглеждания робот, дефинирано в (2.11), броят на ставните конфигурации, с които може да достигне дадена позиция с изпълнителното си звено е ограничен. Освен това, както видяхме в предходната глава, в зависимост от стойностите на ставните координати, могат да бъдат класифицирани четири типа решения на обратната задача на кинематиката. При изпълнение на зададена траектория е възможно да се наложи роботът да промени типа решение. В зависимост от това в кой момент се извършва тази смяна, хващачът на робота би могъл да се отклони от желаното движение. Отклонението от предварително дефинираната траектория пък би довело до увеличаване на времето за изпълнение на поставената задача.

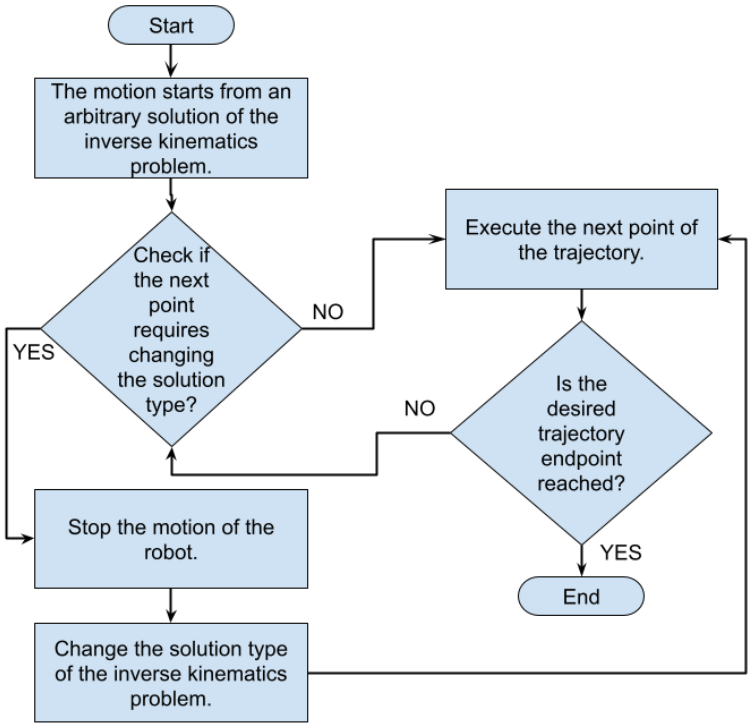
При роботите с допълнителни степени на свобода, какъвто е разглежданият в тази дисертация робот, съществува множество от ставни конфигурации за дадена позиция на хващача. Това позволява да се планира движението на робота по такъв начин, че да се избере подходяща ставна конфигурация, в която да се извърши промяната на типа решение. По-този начин би могло да се избегне нежелано отклонение от целевата траектория. Затова тази глава има за цел да предложи алгоритъм за планиране на оптимална по време траектория. Това е такава траектория, която може да бъде изпълнена с минимално отклонение от желания път и минимално увеличение на времето за изпълнение.

3.3.1 Алгоритъм тип „Алчен“ (Greedy approach)

Изследваният робот в тази дисертация има 4 ротационни стави със ставни ограничения (2.11) и дължини на звената (2.10). Тъй като четвъртата ротационна става променя единствено ориентацията на хващача, а ние се интересуваме само от неговата позиция, ще бъдат разглеждани само първите три ротационни стави.

За да бъде изпълнено дадено движение, то може да бъде планирано по следния начин. Движението започва от произволно решение на обратната задача на кинематиката. Типът на това решение се запазва и се използва за следващите точки от желаната траектория, докато не се наложи смяна на типа решение на обратната задача на кинематиката. Ако се налага смяна на типа решение, то движението трябва да спре и да се смени типа на ставната конфигурация на

робота. След което, изпълнението на зададеното движение може да продължи. Този метод е от тип алчен алгоритъм (greedy approach). На Фиг. 3.12 е представена схема на алгоритъма.



Фиг. 3.12. Схематично представяне на алчния алгоритъм.

Необходимо е да се отбележи, че при прилагане на такъв тип метод за робот с допълнителни степени на свобода и ограничено ставно пространство, при смяна на типа решение на обратната задача на кинематиката, роботът ще се отклони от желаната траектория. От друга страна, ако изпълнението на зададената траектория е възможно при спазване на ограниченията на ставното пространство, винаги ще има възможност за промяна на вида решение по време на движение по желаната траектория. Затова е необходимо да се създаде алгоритъм за планиране на траектория с минимално отклонение от желания път. Следващата Глава 3.3.2 представя алгоритъм за планиране на траектория с минимално отклонение от желания път.

3.3.2 Планиране на траектория с минимално отклонение от желания път

Вече отбелязахме, че е от важно значение една роботизирана система да може да изпълни поставената задача максимално прецизно и за минимално време. Тъй като повечето роботи изпълняват предварително определена траектория, то това означава, че по време на своето движение не трябва да се отклоняват от тази траектория. Използвайки алгоритми като алчния алгоритъм, Rapidly exploring random tree или Probabilistic roadmap method са възможни отклонения от желаната траектория, затова е необходимо да се създаде алгоритъм за планиране на движение с минимално отклонение от желаната траектория.

Алгоритъмът трябва да определи видовете решения на обратната задача на кинематиката и възможните преходни конфигурации. Желаната траектория трябва да се раздели на определен брой точки, за да може да се използва дискретна структура като насочен граф за планиране на траектория. Подходът трябва да намери възможните решения на обратната задача на кинематиката за всяка точка и да ги класифицира според типа им, след което да построи граф с тегла и да намери път с минимална цена за желаната траектория. Така описан алгоритъмът за планиране на траектория може да се представи в няколко фази:

1. **Кинематичен анализ на равнинен робот с допълнителни степени на свобода**

Първата фаза от алгоритъма се състои в извършване на кинематичен анализ на равнинен робот, взимайки се предвид ставните му ограничения. Кинематичният анализ ще даде информация относно видовете решения на обратната задача на кинематиката и видовете преходни конфигурации, в които роботът може да попадне.

Две решения на обратната задача на кинематиката се приемат, че са от различен тип, ако при изпълнение на движение от точка до точка между тези две решения, роботът преминава през преходна конфигурация. Кинематичният анализ на разглеждания в тази дисертация робот е представен в предходната Глава 2.

2. **Анализ на желаната траектория**

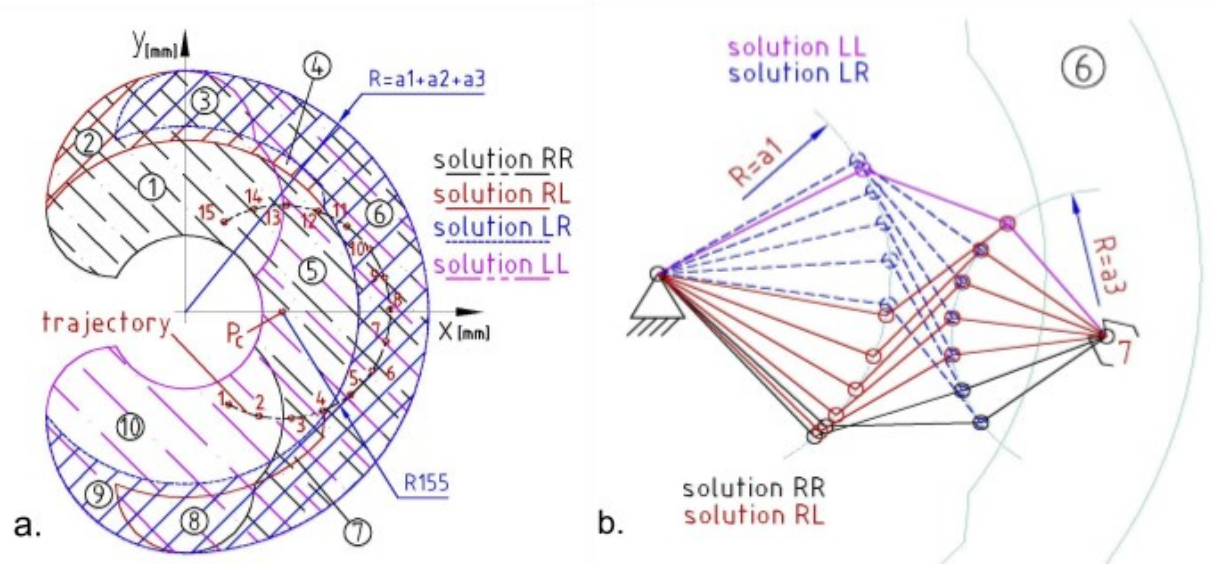
Втората фаза се състои в извършване на анализ на желаната траектория, за която трябва да се планира движението на робота. Тази траектория трябва да бъде разделена на множество от точки, намиращи се на равно разстояние една от друга. Това ще позволи планирането на траектория да бъде реализирано, чрез дискретна структура, като например претеглен насочен граф.

3. **Генериране на решения**

В третата фаза е необходимо да се изчислят различните решения на обратната задача на кинематиката за всяка от избраните точки от траекторията от предходната стъпка. Тъй като разглежданият равнинен робот е с допълнителни степени на свобода, то съществува множество от решения на обратната задача на кинематиката за всяка една точка. Затова решенията трябва да бъдат ограничени и да бъдат на равни отстояния едно от друго.

4. **Класификация на генерираните решения**

В тази фаза всички генерирани решения трябва да бъдат класифицирани според намерените типове решения на обратната задача на кинематиката в кинематичния анализ от първата фаза. Намерените и класифицирани решения на обратната задача на кинематиката за избрана точка (точка 7 от траекторията на Фиг. 3.13.a.) са представени на Фиг. 3.13.b.



Фиг. 3.13. Различни типове решения на обратната задача на кинематиката: a. различни типове решения; b. класификация на решенията на обратната задача на кинематиката за точка 7.

5. Конструиране на насочен граф с тегла

Следващата фаза се състои в конструирането на насочен граф с тегла. Всеки един от върховете на графа съответства на генерирано решение от стъпка 3. Между два върха v и u ще има ребро (v, u), ако v и u съответстват на две съседни точки p_v и p_u от желаната траектория и p_v е преди p_u. Теглото $\omega(v, u)$ за всяко ребро (v, u) може да се дефинира по следния начин:

$$\omega(v,u) = \sum_{i=1}^{k} |\theta_i^v - \theta_i^u| \qquad (3.1)$$

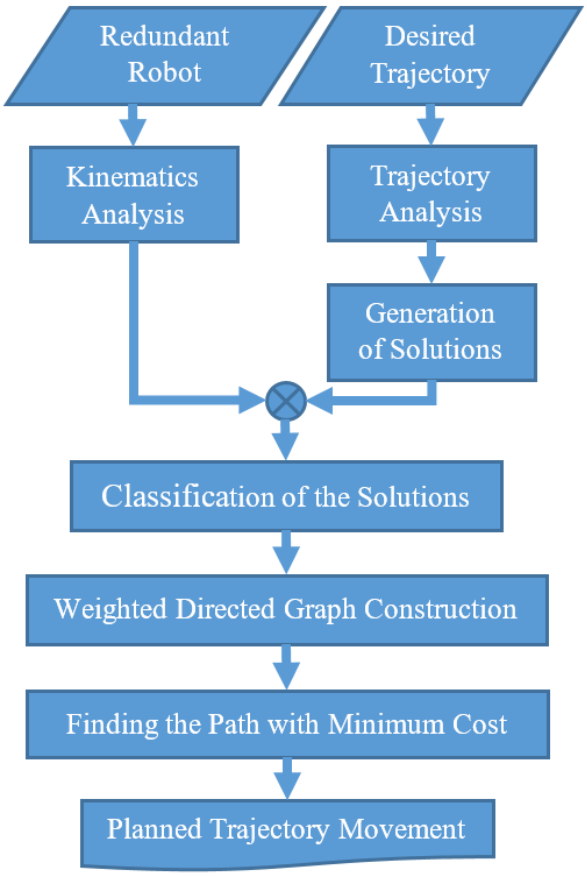
Където k е броят на ставните координати и $(\theta_1^v, \theta_2^v, \dots \theta_k^v)$ и $(\theta_1^u, \theta_2^u, \dots \theta_k^u)$ са решенията, съответстващи на върховете v и u. За разглеждания робот $k = 3$. Уравнение (3.1) не взима предвид времето, необходимо за ускорение или спиране на движението. Тези времена могат да бъдат игнорирани, тъй като се счита, че планираната траектория има ускорение само в началото и намаляване на скоростта в края на движението. Уравнението (3.1) отчита разликата в координатите на ставите на всички k върха. Необходимо е да се отбележи, че преминаването през преходна конфигурация може да изисква по-дълго време за движение.

6. Намиране на път с минимална цена за желаната траектория

Последната фаза от алгоритъма е да се намери оптимален път от всеки връх, който представлява решение на стартовата позиция до всеки връх, който представлява решение на целева позиция от желаната траектория. Два от най-използваните метода в теорията на графите са Floyd-Warshall и Dijkstra. Floyd-Warshall алгоритъмът има висока изчислителна сложност и е подходящ за графи с малък брой върхове. Друг популярен алгоритъм е евристичният метод A*. При прилагане на алгоритмите Dijkstra и A* за обхождане и търсене в граф има само един начален

и краен връх. В разглеждания случай за планиране на траектория за равнинен робот с допълнителни степени на свобода има повече от един начален и целеви връх, в зависимост от броя генерирани решения. Всички намерени начални върхове трябва да бъдат разгледани. Затова за планиране на траектория за робот с допълнителни степени на свобода, базирано на теория на графите, ще се използва алгоритъмът Floyd-Warshall.

Алтернативно, желаната траектория може да бъде разделена на няколко по-къси траектории, за които генерираният граф ще има по-малък брой върхове. На Фиг. 3.14 е представена схема, включваща отделните стъпки от алгоритъма за планиране на оптимална по време траектория.



Фиг. 3.14. Схематично представяне на алгоритъма за планиране на траектория.

Крайният резултат на алгоритъма ще бъде път от начален до целеви връх с минимално тегло. Това ще бъдат решенията на обратната задача на кинематиката за последователни точки от желаната траектория. Между всеки две последователни точки представящи различен тип решение, трябва да се планира спиране на движението с цел преминаване през преходна конфигурация и промяна на текущия тип решение. По този начин движението по траекторията може да бъде планирано с минимално време за изпълнение. За верифициране на предложения алгоритъм са направени няколко експеримента. Експериментите сравняват алчния алгоритъм и предложения алгоритъм за планиране на траектория и са представени в следващата глава.

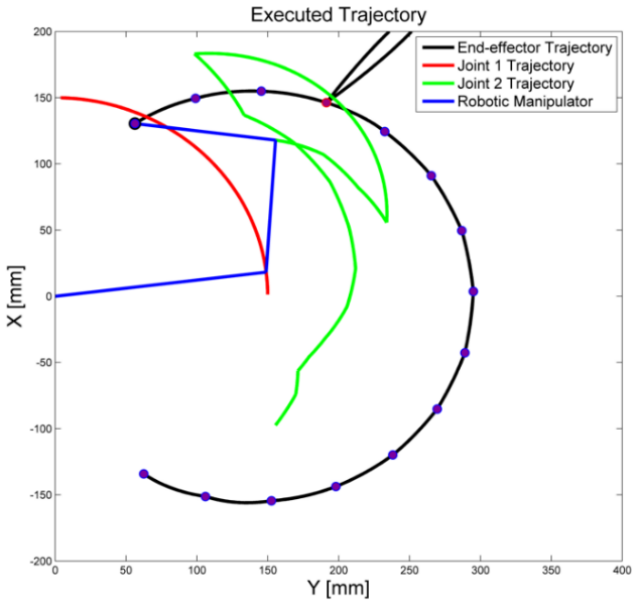
3.3.3 Сравнение на алгоритмите за планиране на траектория

За да сравним представените алгоритми за планиране на траектория ще използваме траекторията от Фиг. 3.13.а. и разглеждания в тази дисертация равнинен робот. Отново ще припомним, че роботът има дължини на звената дефинирани в (2.10) и ставите му имат ограничения, които са дефинирани в (2.11).

Траекторията, която роботът трябва да изпълни представлява движение по дъга. Дъгата е част от окръжност, чийто център е с координати (140, 0) и радиус 155 mm. Желаната траектория не може да бъде изпълнена от конкретния робот, използвайки само един тип решение на обратната задача на кинематиката. Затова ще бъде необходима поне една смяна на типа решение.

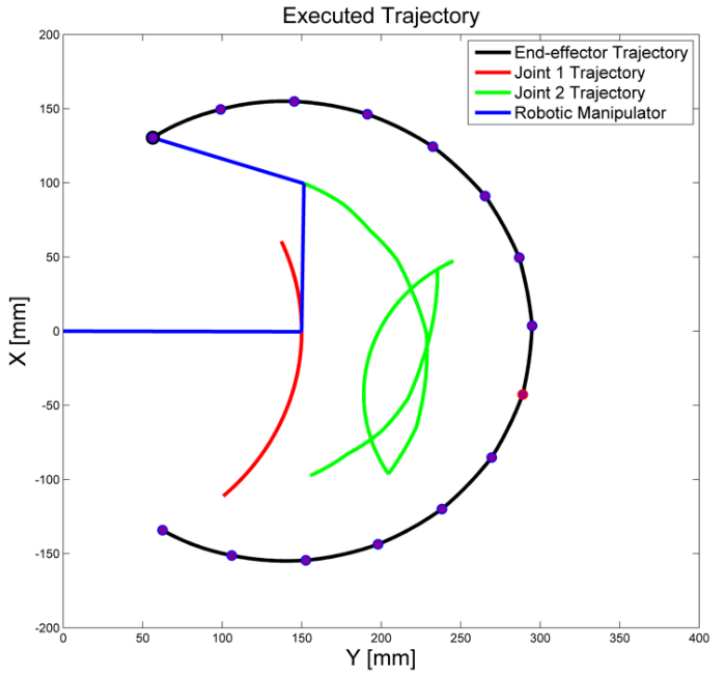
Желаната траектория е разделена на 15 точки, намиращи се на равно разстояние една от друга. За всяка точка се генерират до 100 решения на обратната задача на кинематиката. Тези решения се категоризират според 4-те възможни типа решения на обратната задача на кинематиката. Тъй като разглежданият робот има ограничено ставно пространство, съществуват точки с малък брой решения за избраната траектория. Общият брой намерени решения е 408.

Ако планираме движението с алчния алгоритъм (greedy approach) може да се изпълни с една единствена смяна на типа решение. Тази смяна ще се извърши в точка 12. Движението е показано на Фиг. 3.15. Но ограниченото ставно пространство ще изисква отклонение от желаната траектория при изпълнение на тази смяна.



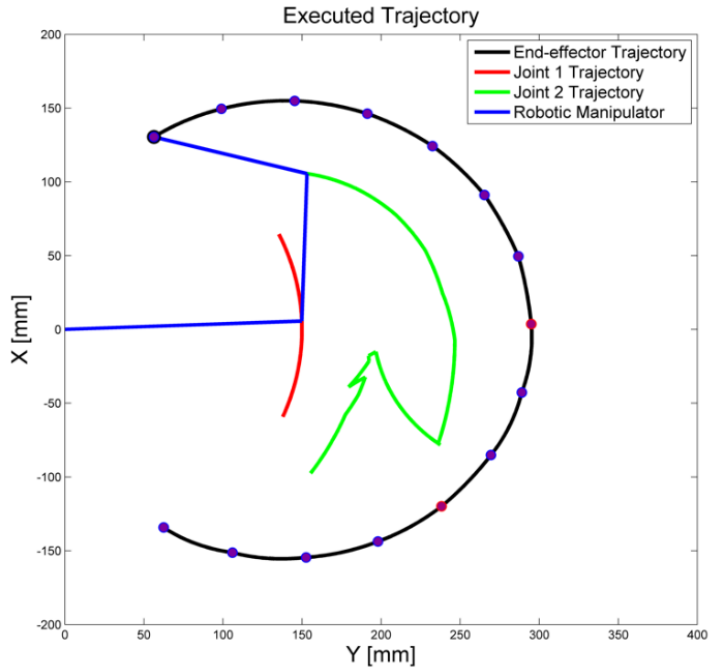
Фиг. 3.15. Изпълнение на траектория от равнинен робот, посредством алгоритъм тип „алчен“.

Движението може да бъде планирано без отклонение от желаната траектория, ако смяната на типа решение се извърши в по-ранна позиция. На Фиг. 3.16 е показана изпълнената траектория, ако смяната се планира за точка 7.



Фиг. 3.16. Изпълнение на траектория от равнинен робот, извършвайки една смяна на типа решение на обратната задача на кинематиката.

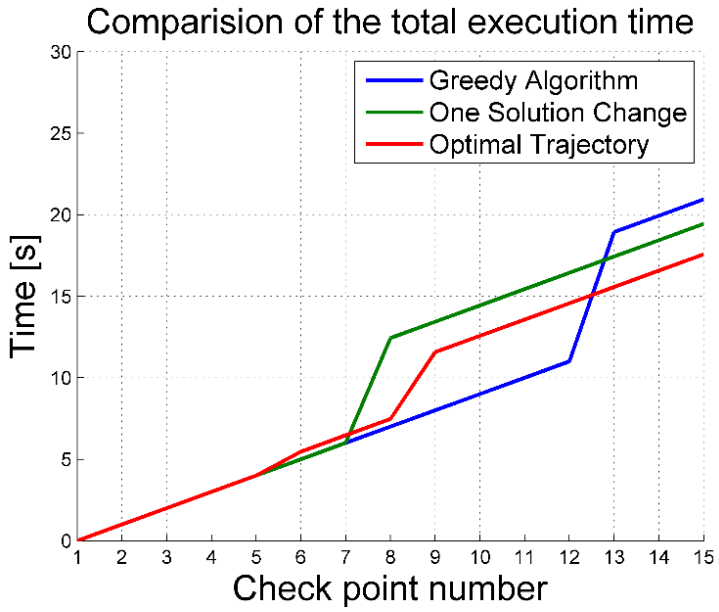
Нека сега да разгледаме предложения алгоритъм за планиране на траектория с минимално време за изпълнение (Фиг. 3.17).



Фиг. 3.17. Изпълнение на траектория от равнинен робот, посредством предложения алгоритъм.

Общият брой на намерените решения на обратната задача на кинематиката е 408. Това означава, че конструираният граф има 408 върха. Затова Floyd-Warshall алгоритъмът може да бъде директно приложен, тъй като броят на върховете на графа е по-малък от 1000. Планираният

оптимален път се състои от траектория, за чието изпълнение са необходими две смени на типа решение на обратната задача на кинематиката. Първата смяна е необходимо да се направи в точка 5, а втората в точка 8. На Фиг. 3.17 са изобразени и траекториите, които различните стави изпълняват. Сравнение на времето за изпълнение на желаното движение, използвайки трите подхода: greedy approach, движение с една смяна и предложеният алгоритъм е показано на Фиг. 3.18.



Фиг. 3.18. Сравнение на времето за изпълнение на желаната траектория на трите разгледани подхода.

Както се вижда от фигурата, алчният алгоритъм е най-бърз до момента на смяната на типа решение, в точка 12. Тук е необходимо повече време за извършване на промяна в типа решение на обратната задача на кинематиката, тъй като смяната в точка 12 предизвиква отклонение от желаната траектория. Планираното движение с една смяна на типа решение в точка 7 е с по-малко време за изпълнение, тъй като смяната е извършена в позиция, която изисква по-малко отместване. Както виждаме, предложеният алгоритъм има най-малко време за изпълнение. Извършени са две смени на типа решение и след втората смяна алгоритъмът взима преднина в сравнение с другите два алгоритъма.

В тази глава е предложен подход за планиране на движение с минимално отклонение от желаната траектория за изследвания равнинен робот с допълнителни степени на свобода. Но често в работното пространство на робота има налични препятствия, които роботът трябва да преодолее, избягвайки възможен сблъсък с тях и изпълнявайки желаното движение. Тези препятствия могат да бъдат както статични, така и динамични. Следващата глава разглежда проблемите при планиране на движение при наличие на статични и динамични препятствия за робот с допълнителни степени на свобода.

3.4 Управление на движение при наличие на препятствия

Роботите изпълняват все по-сложни задачи, като понякога в средата, в която манипулират има и други обекти като роботи или хора. Затова планирането на движение при наличие на препятствия е важна част от управлението на роботизирани системи. Препятствията могат да бъдат статични или динамични. Когато роботът манипулира в среда със статични препятствия, управляващата система има информация за формата и размерите на препятствията, както и тяхната позиция в работното пространство. В този случай трябва да се планира движение, което осигурява безопасно изпълнение на поставената задача, заобикаляйки наличните статични препятствия. В среда с динамични препятствия, задачата на управляващата система се усложнява. В този случай управляващата система трябва да следи в реално време позицията на препятствията. При появило се ново препятствие, което възпрепятства изпълнението на желаното движение е необходимо да се планира нова траектория, която заобикаля препятствието.

Тази глава представя алгоритми за планиране на движение при наличие на статични и динамични препятствия за робот с допълнителни степени на свобода.

3.4.1 Планиране на движение при наличие на статични препятствия

Както видяхме различните типове решения на обратната задача на кинематиката (Фиг. 3.13.а.) разделят работното пространство на робота на различни зони. Движението, което роботът трябва да изпълни е възможно да изисква преминаване през няколко от тези зони. Това преминаване изисква от робота да промени типа на текущата ставна конфигурация. При наличие на препятствия в работното пространство на робота, зоните се променят и е възможно някои от тях да бъдат разделени на няколко части от препятствията. Затова е необходимо промяната в типа решение на обратната задача на кинематиката да се извърши в подходящ момент. В противен случай е възможно отклонение от желаното движение и увеличаване на времето за изпълнение.

Множеството от точки, в които роботът може да промени типа решение на обратната задача на кинематиката ще наричаме преходни точки. За разглеждания робот са възможни следните промени в типа решение: от LL към LR, от RR към RL, от LL към RL и от LR към RR, възможен е преход от един тип в друг в обратен ред. Както казахме, разглежданият робот е с допълнителни степени на свобода, затова съществува повече от 1 решение за дадена позиция на обратната задача на кинематиката. Когато се разглежда движение от точка до точка ставното пространство може да бъде дискретизирано, което ще позволи предварително изчисляване на всички възможни конфигурации на ставите и съответната позиция в работното пространство на хващача на робота. След това позициите в работното пространство, за които съществуват множество

типове решения, могат да бъдат маркирани като подходящи преходни точки. Необходимо е да се отбележи, че различните типове решения се определят от стойностите на ставните координати θ_2 и θ_3. Преходните точки, които ни интересуват са тези, за които има два различни типа решение и една от ставните координати θ_2 и θ_3 е близка до нула. Например, преходните точки от LL към LR са тези, за които съществува LL и LR тип решение, $\theta_2 < 0$ и стойността на θ_3 е близка до нула.

Когато даден манипулатор се намира в среда с препятствия, за които се знаят геометричните особености и позицията им остава постоянна по време на движението на робота, казваме че роботът се намира в среда със статични препятствия. В този случай трябва да се планира движение, което може да се изпълни без опасност от сблъсък с налично препятствие. За тази цел ще се създаде алгоритъм за планиране на движение с избягване на препятствия от точка до точка за равнинен робот, като се вземат предвид кинематичните характеристики на робота, както и подходящите преходни точки. Тъй като ще се разглежда равнинно движение роботът е с допълнителни степени на свобода. Ставните му ограничения са дефинирани в (2.11).

Работното пространство на робота може да бъде разделено на краен брой квадратни сегменти с дължина на страната a и краен брой точки. Допълнително, можем да разгледаме крайно множество $\boldsymbol{A}$ от валидни ставни координати, за които няма сблъсък с препятствия. Решението на правата задача на кинематиката за разглежданото множество ще бъде резултат от крайно множество $\boldsymbol{B}$ от координати в работното пространство на робота. След което, може да се създаде неориентиран непретеглен граф $\boldsymbol{G}$ по следния начин.

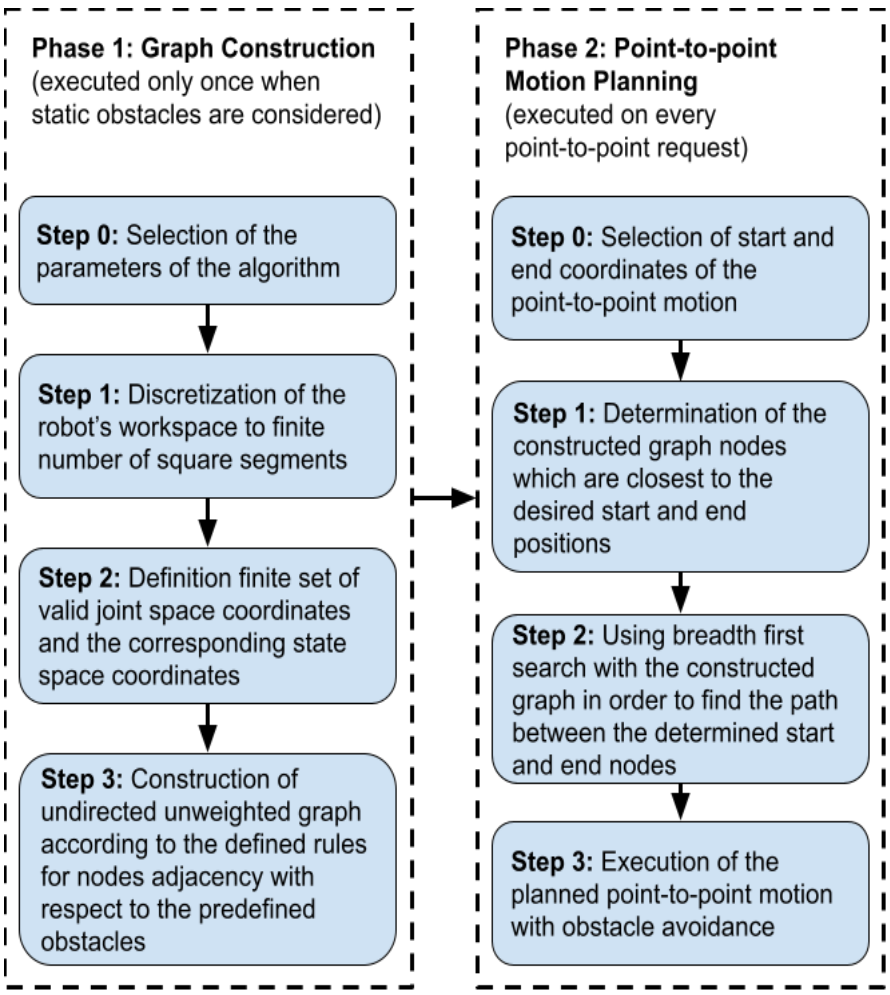
Върховете на графа съответстват на всяка една точка от множеството $\boldsymbol{B}$. За всеки два различни върха u, v съществуват ребра (u, v) и (v, u), ако:

- нормата на векторната разлика на съответстващите на u и v координати в работното пространство е по-малка от предварително дефинирана стойност α;
- нормата на векторната разлика на съответните ставни координати на u и v е по-малка от предварително определена стойност β;
- няма опасност от сблъсък с налично препятствие при движение на робота от u до v;
- съответните координати в работното пространство се намират в рамките на един и същи сегмент от дискретизираното работно пространство (това позволява на робота да промени конфигурацията си от един тип решение на обратната задача на кинематиката на друг) или ако двете конфигурации имат един и същ тип решение на обратната задача на кинематиката. Това позволява на робота да извърши движение до съседна позиция без промяна на текущия тип решение.

Алгоритъмът за търсене в ширина (breadth first search, BFS) може да бъде използван за намиране на път между два върха в графа $\boldsymbol{G}$. За построяването на графа $\boldsymbol{G}$ и изпълнението на BFS

е достатъчно да се реши само правата задача на кинематиката. Това е изчислително ефективна задача за манипулатори с отворена кинематична верига. Ако желаната стартова или целева позиция на движението от точка до точка не съответства на никой от върховете в графа $\boldsymbol{G}$, тогава алгоритъмът за планиране на движение трябва да намери най-близкия съответстващ връх на стартовата и/или целевата позиция. Тогава планирането на движение между дадена стартова и целева позиция се свежда до намиране на път между съответстващия най-близък стартов и целеви връх. След което, пътят между двата върха се планира, посредством BFS алгоритъма.

Необходимо е да се отбележи как се изпълнява движението между две позиции. Такова движение се планира като едновременно движение на всички ставни координати с еднаква продължителност. По време на едно такова движение, изпълнителното звено на робота също трябва да извършва движение, в противен случай ставите на робота могат да се сблъскат с налично препятствие. Този възможен сблъсък се взима предвид при конструирането на графа $\boldsymbol{G}$. При планиране на движение от точка до точка, чрез използване на граф $\boldsymbol{G}$ и BFS се гарантира, че няма опасност от сблъсък с налично препятствие от работната зона на робота. Последователността на алгоритъма за планиране на движение е показана на Фиг. 3.19.



Фиг. 3.19. Алгоритъм за планиране на движение при наличие на статични препятствия.

Алгоритъмът може да бъде разделен на две фази: построяване на граф $\boldsymbol{G}$ и планиране на движение от точка до точка. Фазата за създаване на $\boldsymbol{G}$ се изпълнява само веднъж преди планирането на движението от точка до точка и се взимат предвид само известни статични препятствия в работната зона на робота. В тази фаза се избират подходящи параметри на алгоритъма и се разделя работното пространство на робота на краен брой квадратни сегменти. Дефинира се крайно множество от ставни координати и съответстващите им координати в работното пространство, както и се конструира неориентиран непретеглен граф. При създаването на графа се взимат предвид дефинираните правила за съседство на два върха и информацията за налични препятствия.

Втората фаза се състои в планиране на движение от точка до точка и се изпълнява при всяка заявка за движение до определена точка. В тази фаза се избират стартова и целева позиция за изпълнение на движение от точка до точка. Определят се върховете от графа, които са най-близо до желаните стартова и целева позиция. Използва се BFS алгоритъма върху построения граф в първата фаза за намиране на път между стартовия и целевия връх. Фазата завършва с изпълнение на планираното движение от точка до точка с преодоляване на статични препятствия.

За всяко последователно планиране на движение се използва само алгоритъма BFS. Това прави планирането изчислително ефективно и в зависимост от големината на графа, може да бъде изпълнено в реално време. Размерността на графа $\boldsymbol{G}$ се определя от размерността на множеството $\boldsymbol{A}$. Представеният алгоритъм има следните параметри: дължина на страната на сегментите на работното пространство a, крайно множество $\boldsymbol{A}$ от валидни ставни координати, стойност на нормата на работното пространство α, стойност на нормата на ставното пространство β. Ефективността на алгоритъма е валидирана, чрез реален експеримент с 3D принтиран прототип на изследвания робот. Експериментът ще бъде представен в Глава 4.

Възможно е да има ситуации, в които роботът трябва да изпълнява поставена задача в среда с динамични препятствия. Тогава управляващата система трябва да извърши планирането на движение в реално време, съобразявайки се с новопоявили се препятствие. Следващата секция разглежда управлението на равнинен робот с допълнителни степени на свобода при наличие на динамични препятствия.

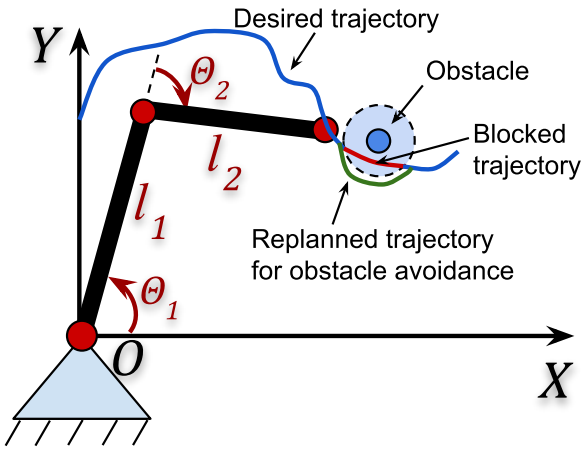
3.4.2 Управление на движение при наличие на динамични препятствия

Когато даден манипулатор работи съвместно с други роботи или хора, казваме че се намира в среда с динамични препятствия. Движението на робота не може да бъде разглеждано като движение от точка до точка, тъй като разглеждаме случай, в който роботът се намира в динамична среда. Във всеки един момент робот, човек или друг обект може да попадне в неговото работно пространство. За тази цел управляващата система на робота трябва динамично да планира неговата траектория в зависимост от наличните препятствия и успешно да ги

преодолее. Управляващата система на робота трябва да планира новата траектория по такъв начин, че отклонението от предварително определената траектория да е възможно най-малко.

Нека да разгледаме роботизирана система с n звена, която трябва да изпълни предварително дефинирана равнинна траектория $\boldsymbol{\theta}_d(t)$, където $t = [0, T]$ е времевият интервал. Траекторията $\boldsymbol{\theta}_d$ е планирана по такъв начин, че роботът ще преодолее всяко едно предварително известно статично препятствие. Управляващата система трябва да открие всяко едно новопоявило се препятствие и да промени движението на робота, така че препятствието да бъде заобиколено без опасност от сблъсък с него. Откриването на препятствията и планирането на траектория при наличие на препятствие трябва да се изпълнят в реално време. Хващачът на робота е необходимо да се позиционира на желаната целева позиция, преодолявайки наличните препятствия в работното пространство на робота.

Препятствията се представят в декартови координати (workspace coordinates). Когато се появи ново препятствие, управляващата система на робота трябва да открие частта от траекторията, която няма да може да бъде изпълнена безопасно и да я блокира. След което, управляващата система е необходимо да планира отново траекторията по такъв начин, че роботът да избегне препятствието (Фиг. 3.20). Алгоритъмът за планиране на траектория при наличие на динамични препятствия е базиран на теория на графите и използва алгоритъм за търсене в ширина. По този начин се извършва изчислително ефективно промяната на вече планираното движение. Това дава възможност за прилагане на алгоритъма в реално време.



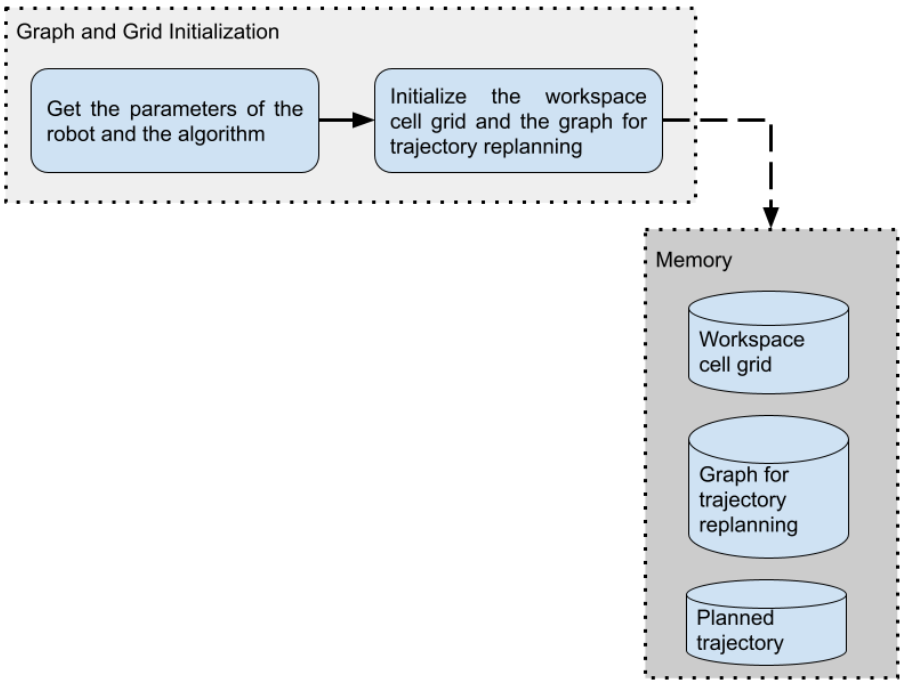
Фиг. 3.20. Планиране на траектория на равнинен робот при наличие на препятствия в работната му зона.

Целта е в края на движението роботът да позиционира изпълнителното си звено в предварително зададена крайна позиция. Нека звената на робота да бъдат означени с i, а дължините на звената с l_i. При робот с две звена са възможни максимум две различни множества от ставни координати за една и съща позиция на хващача на робота. Когато разглеждаме робот с три звена, какъвто е изследваният в тази дисертация робот, броят на различните ставни

конфигурации за една и съща позиция на хващача на робота може да бъде безкраен. Това е необходимо да се вземе предвид от управляващата система при планиране на траекторията.

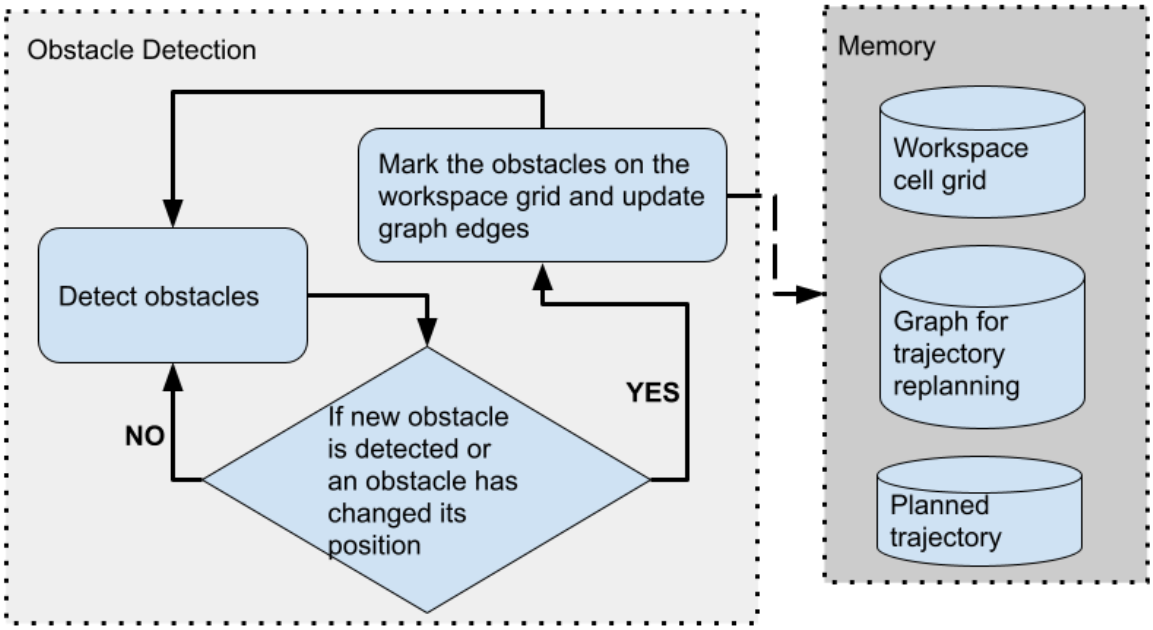
Препятствията се откриват, когато се появят в работното пространство на робота. Откритите препятствия могат да се оградят от допълнителна гранична зона, която да взима предвид ширината на ставите на робота. Това ни позволява да представим робота в работното пространство чрез отсечки с ширина една единица. За планиране на траектория в реално време работното пространство на робота може да се представи като граф с върхове, съответстващи на всяко едно възможно множество от ставни координати, отговарящи на позицията на хващача на робота в работното пространство. Ребрата на графа свързват два върха в зависимост от тяхната съответстваща позиция в работното пространство. Два върха от графа са свързани, ако техните съответстващи позиции са еднакви или съседни и ако необходимото движение е под предварително зададен прагов параметър (*threshold).* След това, работното пространство на робота може да бъде разделено на малък брой позиции, за да се намали броят на върховете и ребрата в графа. Това разделяне се определя от параметъра за разделителна способност на алгоритъма (*resolution)*. По време на изпълнението на траекторията се следи за препятствия и ако се появи препятствие, някои ребра от графа се маркират като блокирани. Когато роботът попадне в ситуация близка до сблъсък с налично препятствие, управляващата система намира частта от траекторията, която трябва да бъде повторно планирана и прилага алгоритъм за търсене в широчина на граф (breath first search, BFS) за планиране на траектория в съответствие със създадения граф. Алгоритъмът има три основни стъпки: **създаване на граф**, **проследяване на препятствия** и **изпълнение на траектория и повторно планиране на траектория**. В обща памет се пази информация за създадения граф, планираната траектория и клетките, на които е разделено работното пространство. По време на всяка една от стъпките се следи за промяна в графа и планираната траектория и съответната информация в общата памет се обновява. На Фиг. 3.21 е показана първата стъпка от алгоритъма.

Създаването на графа (Фиг. 3.21) зависи от следните параметри на алгоритъма: броят на звената n, дължината на звената a_i, където $1 \leq i \leq n$, *threshold* стойността, която определя кога две различни ставни конфигурации се считат за съседни и *resolution* параметърът, който разделя работното пространство на робота на определен брой позиции. Работното пространство на робота се представя като мрежа от клетки (*workspace_grid*). Всяка клетка представлява позиция от работното пространство на робота. След което, се създава граф, като всеки връх съответства на решение на обратната задача на кинематиката, което позиционира хващача на робота в центъра на клетката. За всеки връх от графа се намира съответната клетка от мрежата, нейната съседна клетка и съответните върхове от графа. Алгоритъмът добавя ребро в графа, свързвайки онези върхове, за които съответната клетка на мрежата е една и съща или съседна и необходимото движение отговаря на *threshold* параметъра.



Фиг. 3.21. Схематично представяне на алгоритъма по създаване на граф.

След като е създаден графът и роботът започне да изпълнява траекторията, управляващата система трябва да следи за препятствия (Фиг. 3.22).

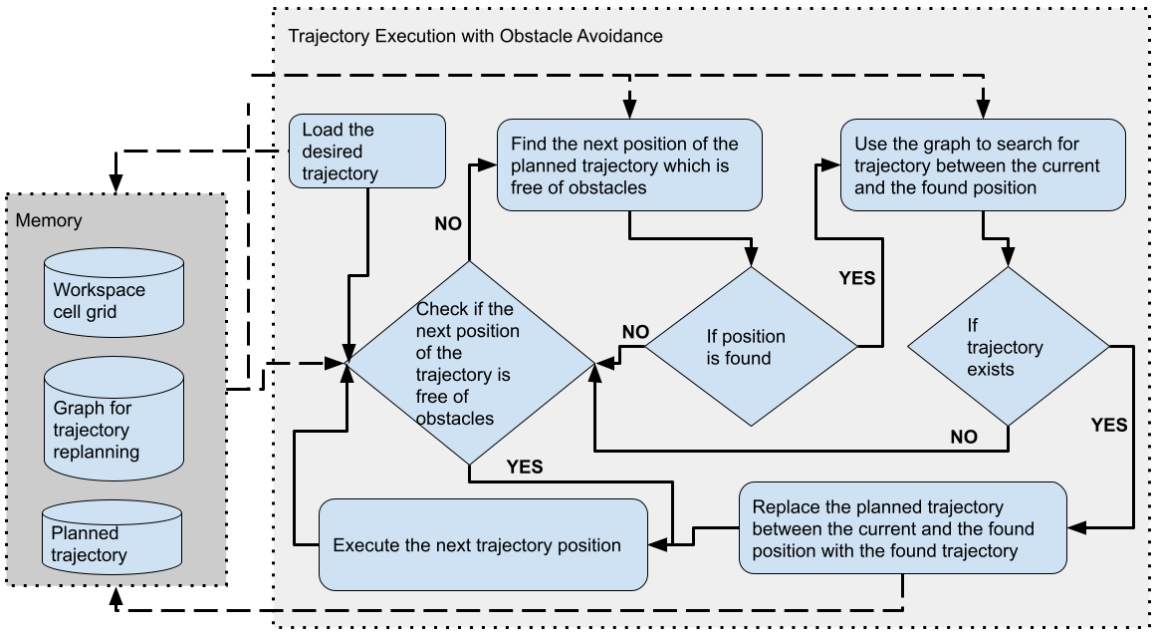


Фиг. 3.22. Схематично представяне на алгоритъма за проследяване на препятствия.

Проследяването на препятствия зависи от следните параметри: *obstacles* - описание на намерените препятствия, *workspace_grid* - мрежа на работното пространство, създадена по време на фазата за създаване на графа, *desired_trajectory* - първоначално желаната и планирана траектория, *current_time* – текущо време и *reaction_time* - предварително дефинирана константа,

която отчита възможностите на робота да спре движението си, което определя колко близо до препятствието може да бъде планирана новата траектория. Когато се открие дадено препятствие се отбелязва в мрежата от клетки (*workspace_grid*) и ръбовете от графа, които свързват върховете, съответстващи на клетките се маркират като неизползваеми. Алгоритъмът проверява и дали препятствието е ново или е сменило своята позиция. Откриването на препятствие се извършва в реално време.

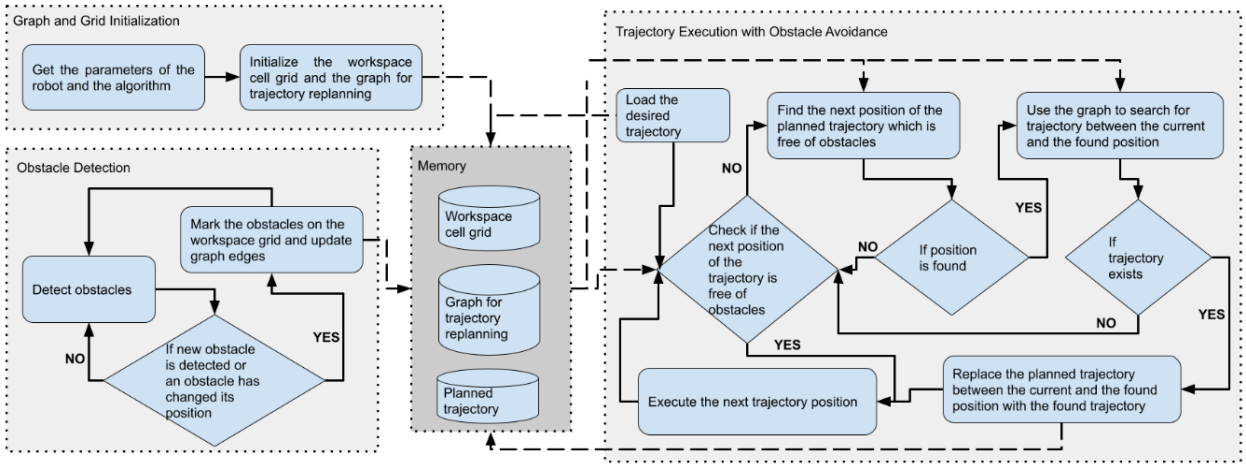
По време на проследяването на изпълнението на траекторията и движението на препятствията, е възможно управляващата система да планира отново движението, поради новопоявили се препятствия (Фиг. 3.23).



Фиг. 3.23. Схематично представяне на алгоритъма за изпълнение на траектория с преодоляване на препятствия.

Повторното планиране на траекторията може да се извърши, посредством генерирания граф и стандартния BFS алгоритъм и зависи от следните параметри: *workspace_grid* – създадена в първата фаза мрежа на работното пространство от генерирания граф, *graph* – също генериран в първата фаза, *start_node* и *end_node* – начален и финален връх, намерен по време на фазата за следене (monitoring phase). Тези върхове съответстват на началото и края на блокираната част от зададената траектория. Алгоритъмът проверява дали следващата позиция от планираната траектория е безопасна за изпълнение (няма опасност от сблъсък с препятствие). Ако няма опасност от сблъсък с препятствие, роботът изпълнява следващата позиция от траекторията. В противен случай, алгоритъмът намира позиция от планираната траектория, която не е блокирана от препятствие. С помощта на създадения граф и BFS алгоритъма се намира път между текущата позиция на хващача на робота и следващата безопасна за изпълнение позиция. Ако такава

траектория не може да бъде намерена се търси следваща точка от траекторията. След като е намерена, новата траектория заменя частта от планираната траектория между текущата позиция и новата намерена позиция и роботът продължава своето движение с изпълнение на следващата точка от траекторията. Планирането на новата траектория се извършва в реално време. На Фиг. 3.24 са показани всички стъпки на алгоритъма за планиране на траектория и преодоляване на динамични препятствия.



Фиг. 3.24. Алгоритъм за планиране на траектория и преодоляване на динамични препятствия.

Предложеният алгоритъм за препланиране на траекторията планира нова траектория с цел избягване на текущите налични препятствия. Предложеният подход се опитва да върне робота възможно най-бързо към желаната траектория, след като препятствията са преодолени. Това позволява да се генерира граф и решетка на работното пространство със среден брой върхове и ребра. Това води до намаляване на обработваните данни и позволява проследяване на препятствията в реално време и избягване на динамични препятствия. Предложеният алгоритъм е валидиран, чрез компютърен и реален експеримент, който е представен в Глава 4.

3.5 Приноси към Глава 3

Една от основните задачи на робота е да изпълнява движение между две точки или последователност от точки. Затова при създаването на подходящо управление е важно да се обърне внимание на начина на планиране на траектория. Разглежданият робот е с допълнителни степени на свобода, когато разглеждаме равнинни движения, което означава, че за една позиция и ориентация на хващача са възможни множества от ставни конфигурации. Във втора глава разгледахме и зоните, на които се разделя работното пространство на робота на базата на типа решение на обратната задача на кинематиката. При изпълнение на дадено движение роботът трябва да смени типа решение и това може да доведе до отклонение от желаната траектория. Затова е предложен алгоритъм за планиране на траектория, който взима предвид описаните и

разгледани проблеми. Алгоритъмът разделя желаната траектория на определен брой позиции (точки) и построява насочен граф с тегла от възможните решения на обратната задача на кинематика за избраните точки от траекторията. След което намира път в графа с минимална цена. Намереният път позволява желаната траектория да се изпълни за минимално време, което увеличава продуктивността на робота. От направеното сравнение с „алчния“ алгоритъм се вижда, че при прилагане на предложения подход, хващачът на робота не се отклонява от желаната траектория при смяна на типа решение и изпълнява зададено движение за по-кратко време.

Планирането на траектория е сложна задача от управлението на робота. Роботите не винаги се намират в безопасна среда без препятствия. Много често им се налага да се справят с различни статични и динамични препятствия в работната им зона. Тези препятствия биха могли да накъсат работното пространство на робота по такъв начин, че изпълнението на желаното движение да бъде невъзможно. Затова важна част от управлението на робота е планирането на траектория при наличие на статични или динамични препятствия в работното пространство на робота.

При планиране на траектория при наличие на статични препятствия, роботът има предварителна информация за геометричните особености и местоположението на препятствията. Едно такова движение може да се разглежда като движение от точка до точка. Предложеният подход за планиране на траектория при наличие на статични препятствия взима предвид особеностите на робота и извършва анализ на кинематиката на робота. В предложените методи в литературата за планиране на траектория при наличие на препятствия [40, 41, 43] не се взимат предвид типовете решения на обратната задача на кинематиката и съответно подходящите преходни точки. Предложеният алгоритъм в тази дисертация взима предвид подходящите преходни точки за смяна на типа решение на обратната задача на кинематиката. Това е от ключово значение, тъй като в противен случай, хващачът на робота би могъл да извърши твърде голямо отклонение от желаната траектория. Това би довело до некоректно изпълнение на поставената задача или би увеличило много времето за изпълнение на поставената задача. Алгоритъмът създава граф с възможните безопасни позиции на робота и намира път в графа за изпълнение на движение от точка до точка. Алгоритъмът прилага BFS върху граф за планиране на траектория между стартов и целеви връх. Използването на BFS алгоритъма за намиране на път в граф прави задачата по планиране на траектория изчислително ефективна и позволява използването на алгоритъма в реално време.

В среда с динамични препятствия управлението на робота става по-трудно, тъй като нямаме предварителна информация за тях. Във всеки един момент може да се появи ново препятствие и роботът трябва да може да предотврати сблъсък с него. Затова едно такова движение не може да се разглежда като движение от точка до точка и планирането на траекторията трябва да се извършва в реално време. За планиране на траектория при наличие на динамични препятствия

отново е използван граф с възможните ставни координати на робота. Работното пространство на робота се представя като решетка. Местоположението на всяко едно препятствие се следи от управляващата система в реално време. Когато управляващата система открие вероятност за сблъсък с дадено препятствие, позицията на препятствието се маркира и съответните ставни координати в графа се блокират. В този момент управляващата система трябва да планира нова траектория, която може да се изпълни без опасност от колизия с препятствие. Алгоритъмът се стреми да върне хващачът на робота възможно най-бързо към първоначалната целева траектория. Алгоритъмът за планиране на движение при наличие на динамични препятствия е разширение на алгоритъма за планиране на движение при наличие на статични препятствия в работното пространство на робота.

Проектирана е хардуерна и софтуерна система за управление на робота. Управляващата електроника се състои от контролера Arduino Mega 2560 в комбинация с Wi-Fi модула WeMos D1 mini. Тези хардуерни компоненти са избрани, тъй като са удобни за програмиране и управление, както и са икономически изгодни. Избраните задвижващи механизми и предавателно отношение ограничават движението на всяка става в рамките на нейните ставни ограничения, което предотвратява възможността за нежелани повреди по робота породени от неправилно изпратени команди при управление. Разработеният софтуер решава правата и обратната задача на кинематиката и позволява на потребителя да управлява робота, като следи неговото състояние в реално време.

Проектираната хардуерна и софтуерна система, както и предложените подходи за планиране на траектория при наличие на статични и динамични препятствия и направените анализи са резултат от работата по публикации [П1], [П3], [П4], [П5] и са докладвани на [Д3], [Д2], [Д4].

3.6 Заключение

За да може един робот да изпълни максимално точно и коректно поставените му задачи е необходимо да се реализира подходящо управление. За създаване на подходящо управление е необходима информация за кинематичните характеристики на робота, както и решения на правата и обратната задача на кинематиката. Затова резултатите от направения кинематичен анализ във втора глава са използвани при създаването на управлението на равнинния робот с допълнителни степени на свобода.

Проектирането на подходяща хардуерна и софтуерна система е от особено значение за управлението на една роботизирана система. В тази глава е представена хардуерната и софтуерната система за управление на разглеждания в тази дисертация равнинен робот с допълнителни степени на свобода. Избраните хардуерни компоненти позволяват бързо и лесно програмиране, както и предоставят необходимите изчислителни ресурси за решаване на правата

и обратната задача на кинематиката. Комуникационният интерфейс пък позволява лесно и удобно разширение и подобрение на хардуерната и софтуерната система за различни изследователски и научни цели.

Друга важна част от управлението на робота е правилното планиране на движение. При изпълнение на дадена задача е важно роботът да изпълни максимално точно и бързо зададената му траектория. Всяко едно отклонение от желаната траектория би увеличило времето за изпълнение на поставената задача. Допълнително, наличните статични и динамични препятствия в работното пространство на робота затрудняват изпълнението на поставената задача. В тази глава са разгледани и предложени алгоритми за планиране на движение при наличие на статични и динамични препятствия. Целта на предложените подходи е да осигурят на робота безопасно изпълнение на зададеното движение. При наличие на динамични препятствия управляващата система в реално време следи за препятствия и коригира траекторията, по която трябва да се движи роботът. Тази глава представя резултатите от изпълнението на задачи 2 и 3 от задачите на дисертационния труд. Следващата Глава 4 верифицира предложените алгоритми, чрез компютърни и реални експерименти със създадения и изследвания равнинен робот.

4. Експериментална верификация, чрез 3D принтиран робот

В предходната глава разгледахме два алгоритъма за планиране на движение на равнинен робот с допълнителни степени на свобода при наличие на статични и динамични препятствия в работната зона на робота. Представени са и избраните хардуерни компоненти за управление на робота, както и проектираната софтуерна система. Целта на тази глава е да верифицира приложимостта на предложените алгоритми, чрез компютърна симулация и реален експеримент с 3D принтиран прототип на изследвания робот. Експериментите с 3D принтирания робот ще покажат и дали избраните хардуерни компоненти и проектираната софтуерна система са подходящи за управление на такъв тип робот.

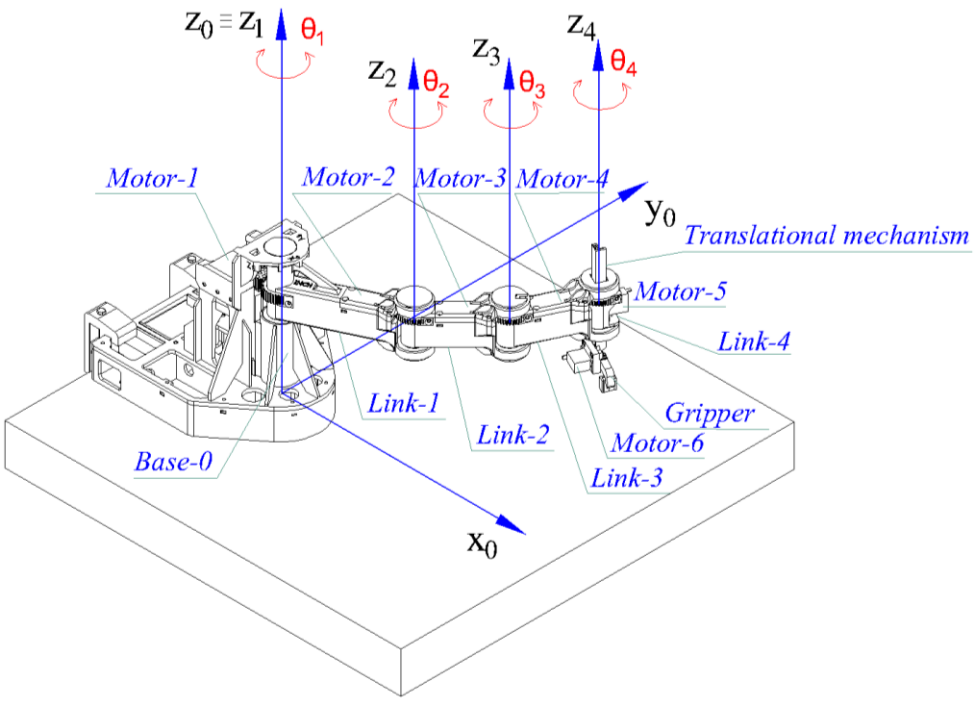
4.1 Проектиране на 3D принтиран прототип на робот

Използвана е технологията на 3D печата за създаване на прототип на изследвания робот с допълнителни степени на свобода. За изготвянето на 3D принтирания прототип са необходими: файлът с 3D модела на робота и 3D принтер. За проектирането на 3D модела е необходимо да се използва някоя от програмите за 3D моделиране: AutoCAD, OpenSCAD или др. В Глава 1 са разгледани някои от тях и са представени техните предимства и недостатъци. За създаването на 3D модела на желания в тази дисертация робот (Фиг. 4.1) е използвана софтуерната програма AutoCAD. Тя поддържа множество от инструменти за създаване и редактиране на 3D обекти, което позволява сравнително бързо да се начертае желания обект.

Едно от изискванията към робота е да бъде с допълнителни степени на свобода, когато изпълнява равнинни движения. Както вече казахме, за тази цел той трябва да има минимум 4 степени на свобода. Затова проектираният робот ще има 4 ротационни успоредни оси. Необходимо е също, роботът да може да изпълнява задачи свързани с преместване на различни обекти, както и да изпълнява пространствени движения. Хващането на тези обекти ще се осъществява с помощта на хващач, който е прикрепен към транслационен механизъм.

Роботът се състои от Base-0 – основа, Link-1 – първо звено, Link-2 – второ звено, Link-3 – трето звено, Link- 4 – четвърто звено, Translational mechanism - транслационен механизъм, Gripper – хващач. В основата на робота е поставена управляващата електроника, която се базира на Arduino Mega 2560 и W-Fi модул WeMos D1 mini. Първият мотор Motor-1 е монтиран в основата на робота и задвижва първото звено. За да може транслационният механизъм да извършва движение по оста z, първото звено е закрепено към основата на височина от 150 mm по z. Вторият мотор е позициониран в края на звено 1 и задвижва второто звено. По аналогичен начин е монтиран и третият мотор в края на второто звено и задвижва третото звено. Четвъртият

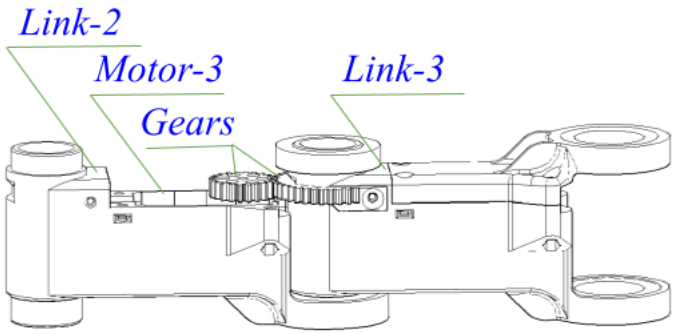
мотор се намира в края на третото звено и променя ориентацията на хващача. В следствие на създадената конструкция, четирите ротационни стави на робота имат ставни ограничения и дължини на звената дефинирани в (2.11) и (2.10).



Фиг. 4.1. 3D модел на равнинен робот с допълнителни степени на свобода.

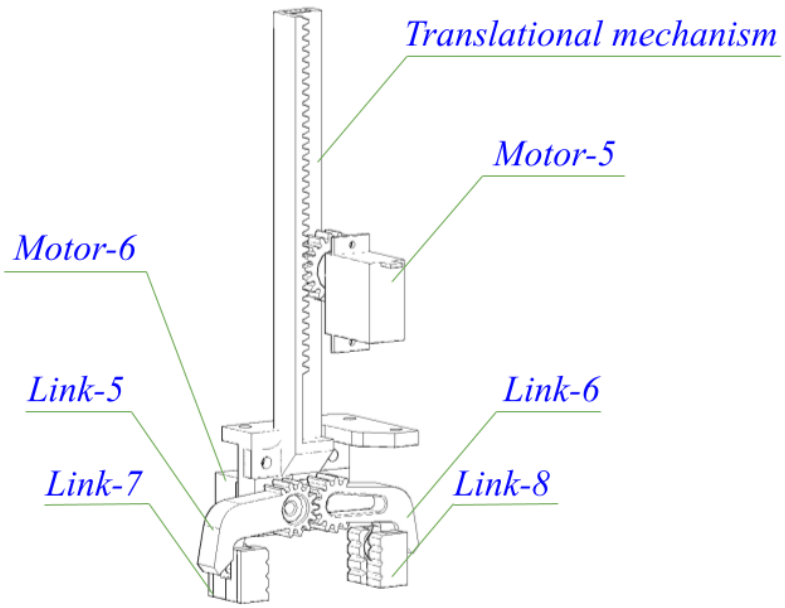
Петият мотор Motor-5 отговаря за задвижването на транслационния механизъм, а шестият мотор Motor-6 за отварянето и затварянето на пръстите на хващача.

Четирите ротационни стави се задвижват от умни сервомотори HerkuleX DRS-0101. Транслационният механизъм и хващача се задвижват от малки сервомотори FeeTech FT90M. Умните сервомотори HerkuleX DRS-0101 имат ъгъл на завъртане от 320 градуса. Четирите ротационни стави на робота могат да се завъртат на 180 градуса. Това означава, че е необходима редукция на предавката от 1:0.5625. По този начин може да се използва пълният обхват на движение на моторите и ограниченията на ротационните стави няма да бъдат нарушени. За тази цел е проектиран предавателен механизъм, който се монтира към осите на сервомоторите. Предавателният механизъм се състои от две зъбни колела с по 22 и 39 зъба, които осигуряват необходимата редукция. На Фиг. 4.2 е показана връзката между звено 2 (Link-2) и звено 3 (Link-3). Както вече казахме третият мотор (Motor-3), който задвижва третото звено се намира в края на второто звено. Към оста на третия сервомотор е монтирано зъбно колело с 22 зъба и завърта третото звено, чрез зъбна предавка с 39 зъба. По същият начин са монтирани и останалите три умни сервомотора съответно за първата, втората и четвъртата ротационна става.



Фиг. 4.2. Свързване на две звена.

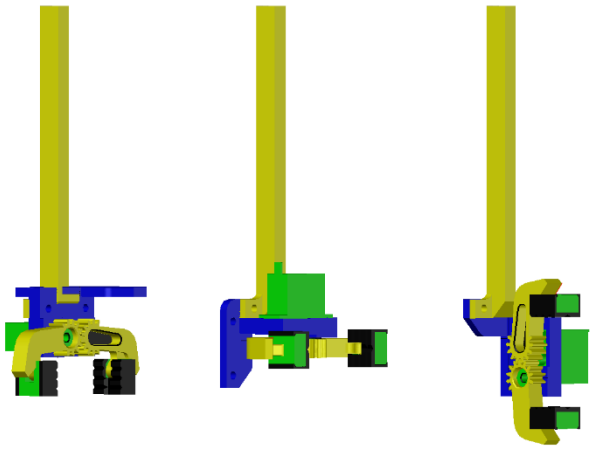
При монтирането на моторите за транслационния механизъм и хващача също са използвани предавателни механизми към осите на сервомоторите (Фиг. 4.3). Към оста на петия сервомотор е прикрепено зъбно колело, което задвижва транслационния механизъм. Транслационният механизъм изпълнява линейно движение по оста z и може да се отмести до 37 mm. Оста на шестия мотор задвижва петото звено от изпълнителния механизъм. Към това звено има зъбно колело, което задвижва шестото звено. Останалите две звена (Link-7 и Link-8) представляват пръстите на хващача. В зависимост от задачата на робота те могат да бъдат изработени от гъвкав материал, който би им осигурил по-добро захващане.



Фиг. 4.3. Устройство на транслационен механизъм и хващача.

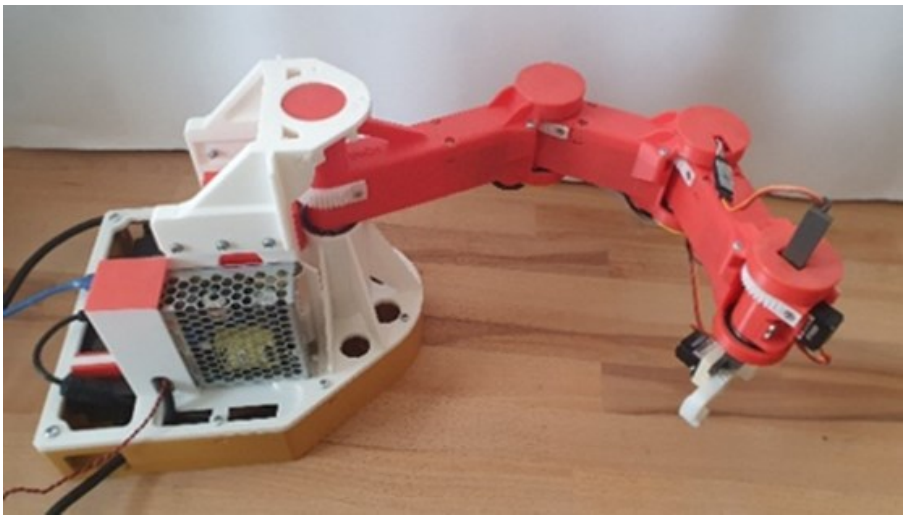
Връзките на робота са конструирани по подходящ начин в съответствие с технологията за 3D печат. Предимството на 3D печата е, че позволява създаване на звена с кухини. Кухините олекотяват конструкцията и се използват за преминаване на кабели за захранване на сервомоторите [84]. 3D принтирането позволява изработката на различни видове изпълнителни

звена в зависимост от задачата, която роботът трябва да изпълни. Хващачът на робота е проектиран по такъв начин, че да позволява различни позиции (Фиг. 4.4). Хващачът може да бъде разположен отгоре или отдолу на рейката на транслационния механизъм, както и може да бъде ориентиран вертикално или хоризонтално.



Фиг. 4.4. Различни позиции на хващача на робота.

След като 3D моделът на желания от нас робот е готов е необходимо да се обработи със софтуер за 3D печат и след това да се разпечата с помощта на 3D принтер. Материалът, който е избран за изготвяне на модела е PLA. Той е най-често използваният материал на основата на полимлечна киселина (Polylactide, PLA) и е ценово ориентиран. Както отбелязахме в предходната глава, обектите, отпечатани от PLA не са подходящи за използване при висока температура, тъй като са крехки и не са устойчиви при температура над 60 градуса. Готовият 3D принтиран робот е показан на Фиг. 4.5.



Фиг. 4.5. 3D принтиран прототип на робот.

След като вече знаем конкретните стойности на дължините на звената на робота, както и ставните ограничения, може да дефинираме явно параметрите на Денавит-Хартенберг:

Таблица 4.1. Стойности на параметрите на Денавит-Хартенберг.

i	α_{i-1} [rad]	a_{i-1} [mm]	d_i [mm]
1	0.00	0	150
2	0.00	150	0
3	0.00	100	0
4	0.00	100	0

Избраната управляваща електроника и задвижващи устройства са подробно описани в предходната Глава 3.

Следващата Глава 4.2 представя проведените компютърни експерименти с цел верификация на предложените алгоритми за управление на движението на изследвания робот.

4.2 Резултати от компютърни експерименти

Компютърните експерименти или симулациите ни позволяват да приложим разработените алгоритми за планиране на движение върху модел на робот с подобни или същите характеристики като реалния. Целта на симулациите е да провери коректността и ефективността на алгоритмите преди да бъдат приложени върху реален физически робот. По този начин би могло да се предотвратят евентуални повреди по робота при неправилно управление. Допълнително, процесът по отстраняване на евентуални грешки или подобряване на алгоритмите за управление е по-бърз и удобен, ако се извършва в симулационна среда. Направени са два компютърни експеримента. За първия експеримент е използван софтуерът MATLAB, а вторият експеримент е направен в симулационната среда Webots.

4.2.1 Изпълнение на траектория при наличие на динамични препятствия

За валидиране на предложения алгоритъм в предходната Глава 3.4 за планиране на траектория при наличие на динамични препятствия са направени два компютърни експеримента с равнинни робота с три звена. Роботите са с допълнителни степени на свобода, тъй като изпълняват равнинно движение. Целта на първия експеримент е да приложи алгоритъма върху модел на робот с подобни характеристики като изследвания в тази дисертация робот. За първия експеримент е използван робот със следните дължини на звената: $l_1 = 100\,mm, l_2 = 80\,mm$ и $l_3 = 80\,mm$. Роботът трябва да изпълни предварително дефинирана траектория, докато преодолява динамични препятствия в работната си зона. Роботът има ставни ограничения, дефинирани в (4.1).

$$-\frac{\pi}{2} \leq \theta_i \leq \frac{\pi}{2}, i = 1, 2, 3 \qquad (4.1)$$

Алгоритъмът представя работното пространство като мрежа от клетки, параметърът *resolution* е избран да бъде 10 mm. Желаната траектория е дадена в (4.2).

$$\theta_1(t) = -0.5 + \frac{1.6t}{5}, \theta_2(t) = -1.57 + \frac{0.87t}{5}, \theta_3(t) = -0.4 + \frac{0.8t}{5}, t = [0,5] \quad (4.2)$$

Роботът е с допълнителни степени на свобода, когато изпълнява движение в равнината O_{xy}. Това означава, че съществува повече от 1 решение на обратната задача на кинематиката за всяка една достижима позиция в работното пространство на робота. Предложеният алгоритъм за управление на робота при наличие на динамични препятствия може да се раздели на няколко стъпки: създаване на граф, проследяване на препятствия и изпълнение на траектория и повторно планиране на траектория.

При откриване на препятствие, траекторията на робота може да бъде отново планирана, като се фиксира третата ставна координата на робота и се планира нова траектория за първите две звена. За тази цел разглеждаме първите две звена и се създава мрежа (grid) и граф за тях.

Алгоритъмът за създаване на графа зависи от броя на звената n, дължината на звената a_i, *threshold* стойността и *resolution* параметъра и може да бъде имплементиран по следния начин:

Алгоритъм 1. Създаване на граф

```
workspace_grid = init zeros array with dimensions (2 * sum(a_1 + ... + a_n) / resolution + 1) by (2 *
sum(a_1 + ... + a_n) / resolution + 1)
mark all cells in workspace_grid which are attainable by the end-effector of the robot
graph = create an empty graph
for each cell in workspace_grid which is marked as attainable:
  add nodes to graph for each solution of inverse kinematics which position the end-effector to the center
  of the grid cell
for each node in graph:
  find the corresponding cell, its adjacent cell and corresponding graph nodes:
    add edge to the graph connecting those for which their corresponding grid cell is either the same or
    adjacent and if the required motion is below a threshold parameter
```

Така получаваме малка мрежа с 53 на 53 клетки и граф с 499 върха. Втората стъпка от алгоритъма е проследяването на препятствия и зависи от *obstacles, workspace_grid*, *desired_trajectory*, *current_time* и *reaction_time* . При откриване на препятствие се отбелязва в мрежата от клетки (*workspace_grid*). Ръбовете от графа, които свързват върховете, съответстващи на клетките се маркират като неизползваеми. Алгоритъмът може да бъде имплементиран по следния начин:

Алгоритъм 2. Откриване на препятствия и планиране на траектория

```
for each obstacle in obstacles:
  workspace_grid.mark_cells_as_containing_obstacles(cells_corresponding_to_obstacle(obstacle))
start_time = None; end_time = None
for t from current_time to desired_trajectory.end_time:
  if workspace_grid.contains_obstacle(cells_corresponding_to(desired_trajectory[t])):
    if not start_time: start_time = t – 1; desired_trajectory [t].mark_as_collision()
    else:
      if start and not end_time: end_time = t
```

```
    desired_trajectory[t].mark_as_collision_free()
if start_time and not end_time: end_time = desired_trajectory.end_time - 1
if start_time and start_time - current_time < reaction_time and end_time:
 start_node = graph.find_node_closest_to(desired_trajectory[start_time])
 end_node = graph.find_node_closest_to(desired_trajectory[end_time])
 obstacle_avoidance_trajectory = find_obstacle_free_trajectory_connecting(start_node, end_node)
 if not obstacle_avoidance_trajectory: stop_robot_motion_and_recheck()
 else:
  desired_trajectory = desired_trajectory.replace_trajectory_with(start_time, end_time,
   obstacle_avoidance_trajectory)
```

След като се извърши повторно планиране на траекторията за второто звено, трябва да се добави допълнително движение по траекторията за коригиране на ставната позиция на третото звено. Допълнително времето за повторно планиране на траекторията може да се намали, като се планира траектория директно до желаната позиция на хващача.

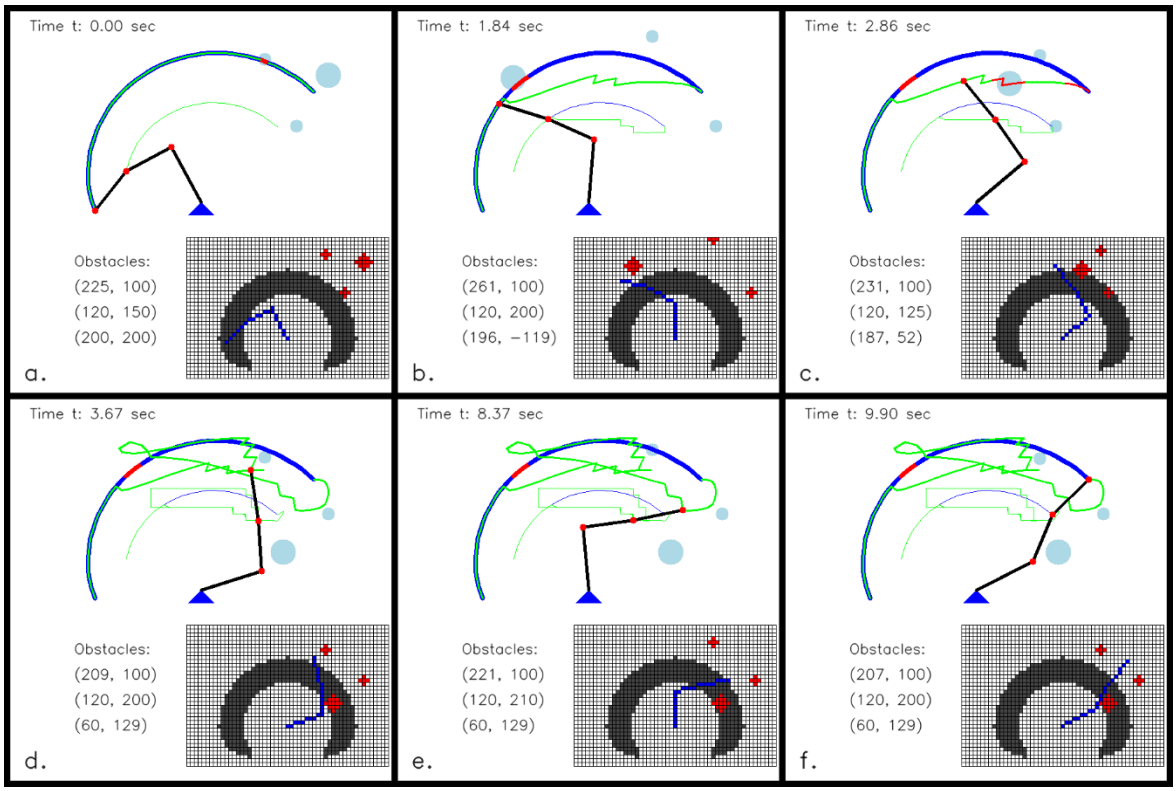
Повторното планиране на траекторията зависи от параметрите *workspace_grid*, *graph*, *start_node* и *end_node* и използва генерирания граф и стандартния BFS алгоритъм за планиране на нова траектория. Алгоритъмът може да бъде имплементиран по следния начин:

Алгоритъм 3. Препланиране на траектория

```
prev = {}; visited = {}; q = Queue(); q.put(start_node); visited[start_node] = True; found = False
while not q.empty():
 current_node = q.get()
 if current_node == end_node: found = True; break
 for node in graph[current_node].neighbors:
 if not node in visited:
  prev[node] = current_node
  if not workspace_grid.contains_obstacle(grid_cells_corresponding_to_node(node)): q.put(node)
  visited[node] = True
current_node = end_node; path = []
if found:
 while current_node != start_node: path.append(current_node);  current_node = prev[current_node]
 path.append(start_node)
 path.reverse()
 return path
return None
```

Изпълнението на траекторията и препланирането в реално време е показано на Фиг. 4.6. Мрежата на работното пространство е показана в долния десен ъгъл на фигурата. Тъмносивите клетки на мрежата на работното пространство маркират клетката, която се използва от алгоритъма за препланиране. Те не представят пълното работно пространство на робота. Те представят работното пространство само на първите две звена на робота. Препланираме само траекторията на първите две звена. Третото звено е фиксирано по време на препланирането на траекторията. Отчита се само при откриване на сблъсък с препятствие. Препятствията са показани със светлосини кръгове. Желаната траектория е показана в синьо. Блокираните части

от траекторията са показани в червено. Траекторията е препланирана успешно и роботът успява да достигне желаната крайна позиция със зададената конфигурация на хващача. Изпълнената траектория е показана в зелено.



Фиг. 4.6. Изпълнение на траектория с избягване на препятствия за робот с три звена.

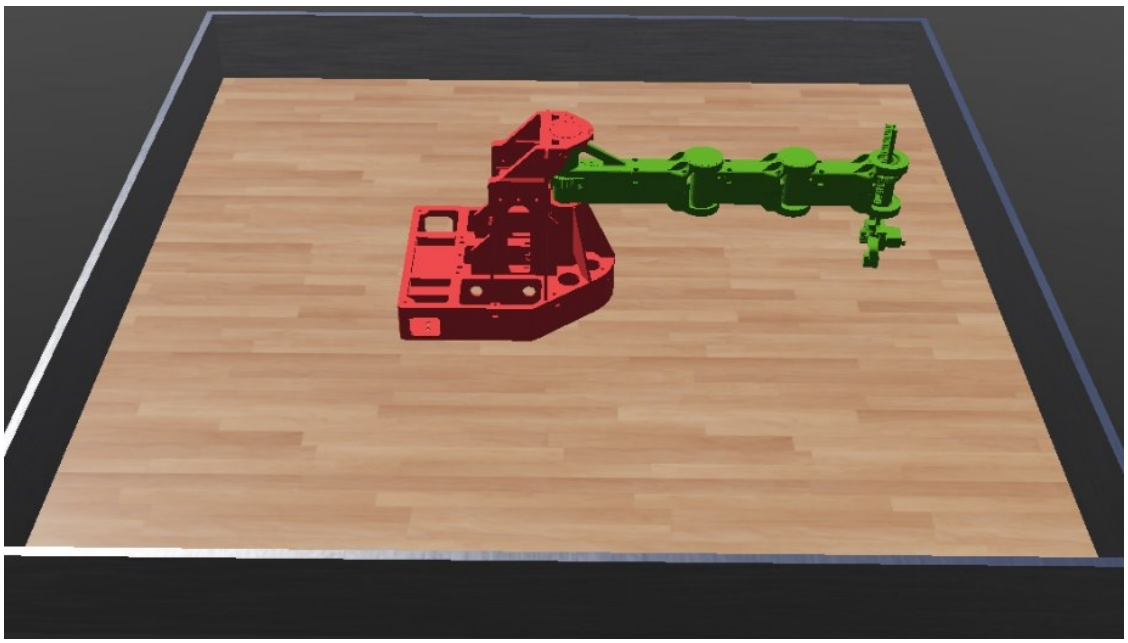
В този експеримент във фазата на планиране на новото движение се фиксира третото звено и новата траектория се планира само за първите две звена, като след това се добавя траектория за движението на третото звено. След проведеният експеримент се установи, че по този начин има ситуации, в които роботът би могъл да преодолее препятствието (има възможност за изпълнение на траектория, заобикаляйки препятствието), но управляващата система не може да планира тази траектория, тъй като движението на третото звено е блокирано. Поради тази причина роботът спира своето движение. Затова в алгоритъма за втория експеримент има промяна в планирането на новата траектория. При появило се препятствие третото звено не се фиксира, а започва да се сгъва. Целта на движението на третото звено е да намали разстоянието между него и основата на робота. За всяка една нова позиция, която изпълнява третата ротационна става се проверява дали може да се създаде безопасна траектория и ако има такава, роботът продължава своето движение.

4.2.2 Експеримент в симулационна среда Webots

След направения обзор в първа глава на някои от най-популярните симулационни софтуерни продукта е избран софтуерът **Webots.** Неговите предимства са че е безплатен софтуер

с отворен код, работи с Windows и Linux и поддържа голям брой симулационни роботи. Предлага и възможност за създаване на собствен модел на робот, който може да се създаде в симулационния софтуер или да се импортира от 3D CAD софтуер. Потребителят може да зададе форма, размери, позиция, ориентация, цветове и текстура на обектите в симулационната среда. Допълнително, софтуерът позволява да се определят някои физични характеристики като маса, триене и др. Използва се в индустриални задачи, за образователни и изследователски цели. Софтуерът предлага богата колекция от сензори и задвижващи устройства. Webots използва ODE (Open Dynamics Engine) за откриване на сблъсъци (колизии) и симулиране на динамиката на твърди тела. Библиотеката ODE позволява прецизно симулиране на физически свойства на обекти като скорост, инерция и триене. Webots поддържа различни програмни езици като C/C++, Python, Java и MATLAB. Webots предлага възможност за заснемане и запис на симулации. Създадената симулационна среда в Webots се съхранява в *.wbt файлове, чийто формат е базиран на езика VRML. Потребителите могат да взаимодействат с работещата симулация, като движат роботите и другите обекти с мишката [72].

За целите на втория експеримент е създаден 3D модел на изследвания робот в симулационната среда Webots. За тази цел е използван 3D моделът на робота, създаден в AutoCAD. Моделът на всяко едно звено на робота се записва като .stl файл. След това се създава .proto файл в Webots, където се импортират файловете на робота и се определят ставните ограничения и геометрията му (Фиг. 4.7). В симулационната среда могат да се добавят и различни сензори, камери, както и препятствия или други необходими обекти.

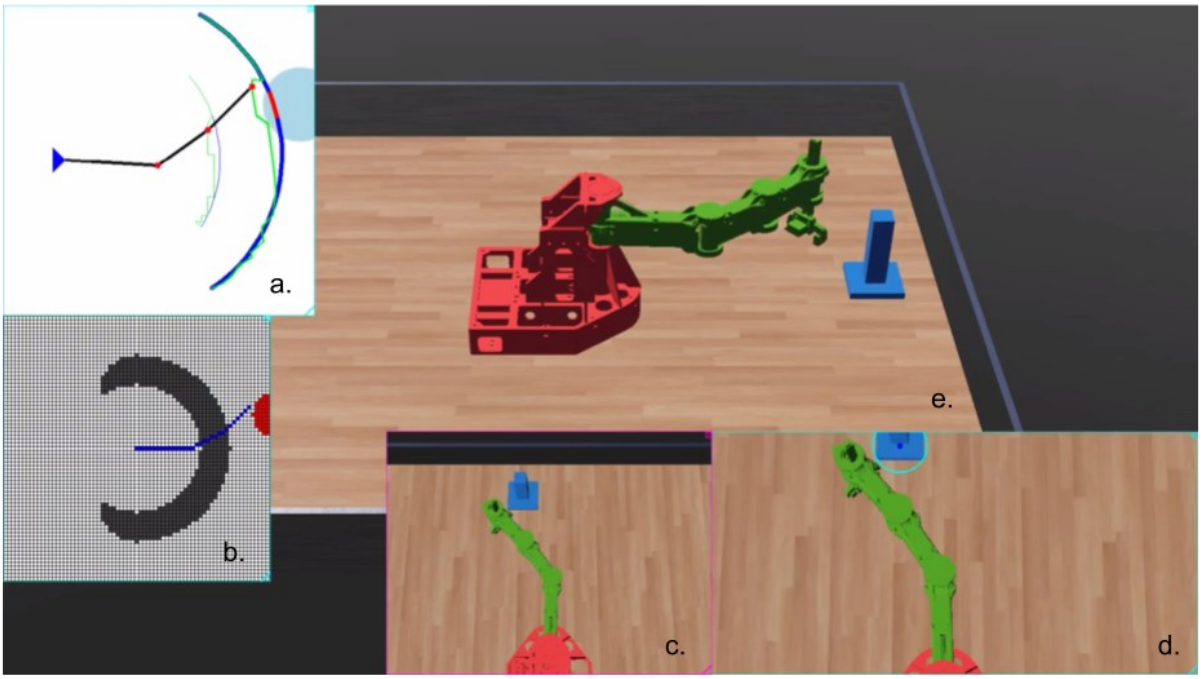


Фиг. 4.7. Модел на изследвания робот в Webots.

Симулационната среда Webots позволява добавяне на различни видове сензори и обекти. За целите на този експеримент са добавени камера и препятствие с дължина на страната 31 mm и височина 120 mm. Целта на експеримента е да докаже, че предложеният алгоритъм за планиране на движение с преодоляване на динамични препятствия е ефективен и подходящ за такъв тип робот. Роботът отново трябва да достигне крайната позиция от зададената траектория с изпълнителното си звено, преодолявайки наличните динамични препятствия в работната му зона. Роботът има дължини на звената и ставни ограничения, дефинирани съответно в (2.10) и (2.11). Алгоритъмът представя работното пространство като мрежа от клетки, параметърът *resolution* е избран да бъде отново 10 mm. Желаната траектория е дадена в (4.3):

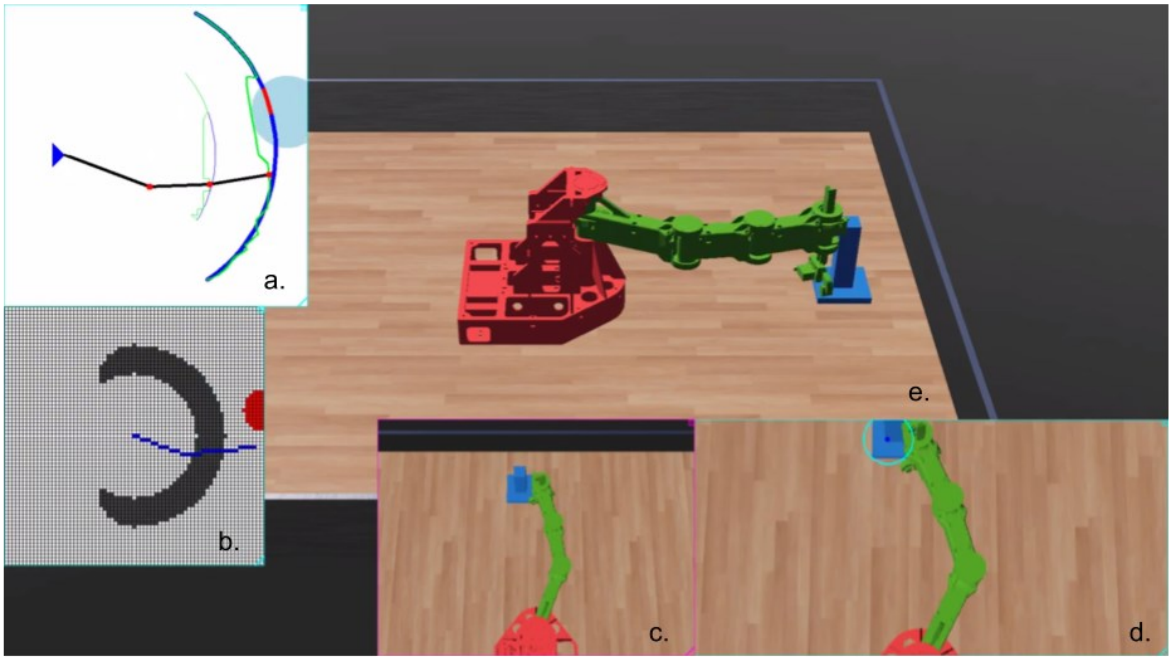
$$\theta_1(t) = 0.4 - \frac{0.7t}{5}, \theta_2(t) = 0.5 - \frac{0.9t}{5}, \theta_3(t) = 0.5 - \frac{0.12t}{5}, t = [0,5] \qquad (4.3)$$

По време на движението на робота препятствието ще променя своята позиция и роботът трябва успешно да го заобиколи. Когато симулационната камера разпознае препятствието, позициите в графа, които са блокирани от препятствието се отбелязват като недостъпни. Частта от траекторията, която е блокирана от препятствието е маркирана в червено (Фиг. 4.8.а.).



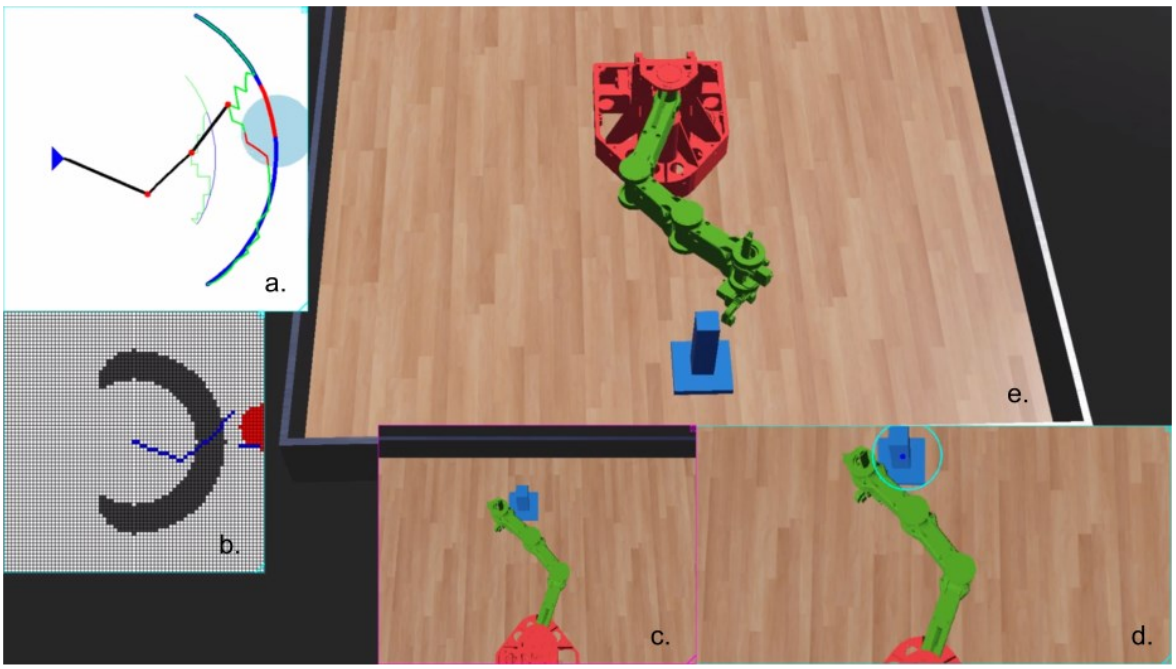
Фиг. 4.8. Блокиране на част от траекторията от динамично препятствие.

Управляващата система трябва да планира в реално време нова траектория, която успешно да заобиколи движещото се препятствие. Допълнително, новата траектория трябва да е планирана по такъв начин, че максимално бързо да върне хващача на робота към целевата траектория (Фиг. 4.9). Зелената линия представлява новата планирана траектория (Фиг. 4.9.а).



Фиг. 4.9. Планиране на нова траектория с цел избягване на динамично препятствие.

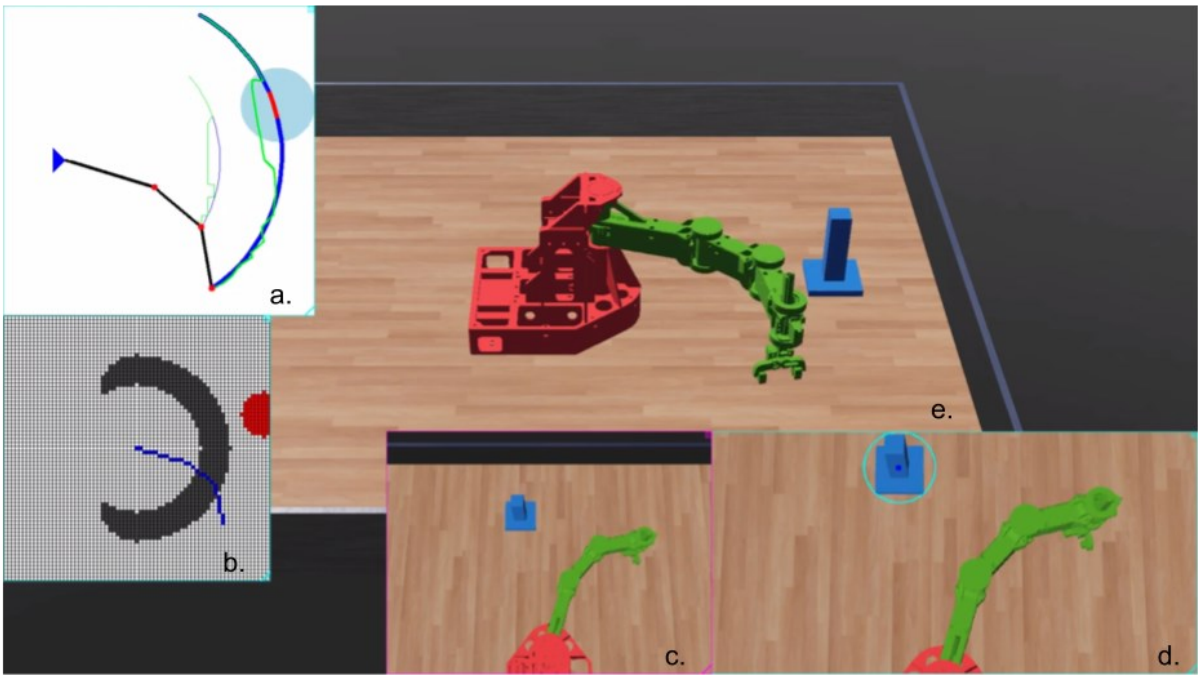
По време на повторното планиране на траекторията е възможно препятствието отново да се премести по такъв начин, че отново да застъпва желаната траектория. Затова управляващата система трябва да следи постоянно по време на движението на робота за промяна на позицията на вече откритите препятствия, както и за новопоявили се такива (Фиг. 4.10).



Фиг. 4.10. Повторно изместване на позицията на препятствието и блокиране на част от препланираната траектория.

Отново е показана мрежата на работното пространство на първите две звена на робота. Тъмносивите клетки на решетката на работното пространство маркират клетката, която се използва от алгоритъма за препланиране (Фиг. 4.10.b.). Тъмносинята дъга представя траекторията, която хващачът на робота трябва да изпълни. Когато препятствие блокира част от нея, тази част се оцветява в червено. Другата тънка синя дъга показва желаната траектория на втората ротационна става. Зеленатa линия изобразява новото планирано движение с цел преодоляване на препятствие (Фиг. 4.10.a.). Когато препятствието се появи в обхвата на камерата и управляващата система го отчете, в симулацията препятствието се маркира със синя окръжност (Фиг. 4.10.d.).

Траекторията е препланирана успешно и роботът успява да достигне желаната крайна позиция с желаната ставна конфигурация (Фиг. 4.11.e).



Фиг. 4.11. Достигане на хващача на робота до крайната точка от планираната траектория.

След като компютърните експерименти са проведени успешно и са доказали, че алгоритъмът е подходящ за управление на антропоморфен робот при наличие на динамични препятствия може да се премине към провеждане на реални експерименти с изследвания робот. Следващата Глава 4.3 представя проведените реални експерименти с 3D принтирания робот с допълнителни степени на свобода с цел верификация на хардуерната и софтуерната система за управление на робота, както и предложените подходи за планиране на траектория при наличие на статични и динамични препятствия в работното пространство на робота.

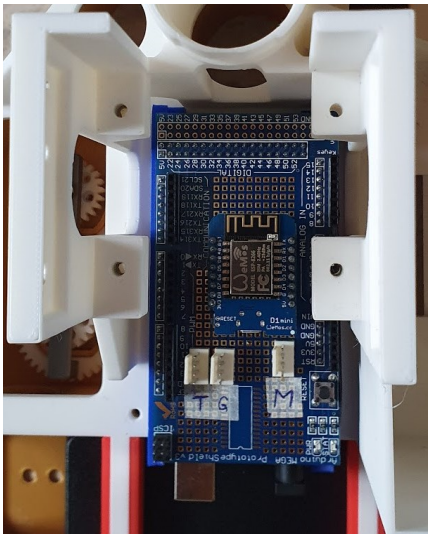
4.3 Експериментална верификация с прототип на равнинен робот

Създадените и описаните симулации за изпълнение на предварително дефинирано равнинно движение от хващача на равнинен робот с допълнителни степени на свобода при наличие на динамични препятствия в работната зона на робота доказаха, че предложеният алгоритъм за управление е подходящ и може да се изпълни в реално време. Следващата стъпка е да се валидира алгоритъма, чрез провеждане на експеримент с 3D принтирания прототип на робота.

Проведени са три експеримента с реалния робот. Преди да се зададе на робота да изпълнява движение в среда с налични препятствия е необходимо да се провери дали проектираната хардуерна и софтуерна система функционира коректно. Това би могло да предотврати евентуални повреди по робота. Затова целта на първия експеримент е да валидира функционалността на хардуерната и софтуерната система на робота. След като се докаже, че роботът получава и изпълнява коректно зададените му команди може да се премине към следващите два експеримента. Вторият и третият експеримент валидират приложимостта и ефективността на предложените алгоритми за изпълнение на зададено движение при наличие съответно на статични и динамични препятствия в работната зона на робота.

4.3.1 Експериментална верификация на хардуерната и софтуерната система

В предходната трета глава подробно разгледахме избраните хардуерни компоненти и проектираната софтуерна система за управление на равнинния робот. Управляващата електроника на робота се състои от микроконтролер Arduino Mega 2560 и Wi-Fi модул WeMos D1 mini. Използваните задвижващи устройства са умните сервомотори HerkuleX DRS-0101, които задвижват четирите ротационни стави и малките сервомотори FeeTech FT90M, които задвижват хващача и транслационния механизъм. На Фиг. 4.12 е показана сглобената управляваща електроника, която се намира в основата на робота. Тази глава ще разгледа проведеният експеримент с описания 3D принтиран робот и създадената система за управление. Целта на експеримента е да се установи дали избраните хардуерни компоненти, комуникационната схема и разработената софтуерна система са подходящи за управление на равнинния робот. Роботът ще изпълнява равнинни движения, затова е с допълнителни степени на свобода. Направеният експеримент е базов, тоест неговата цел е да провери как се справя хардуерната и софтуерната система при изпълнение на движение по зададена последователност от позиции в работното пространство на робота.



Фиг. 4.12. Сглобената управляваща електроника на робота.

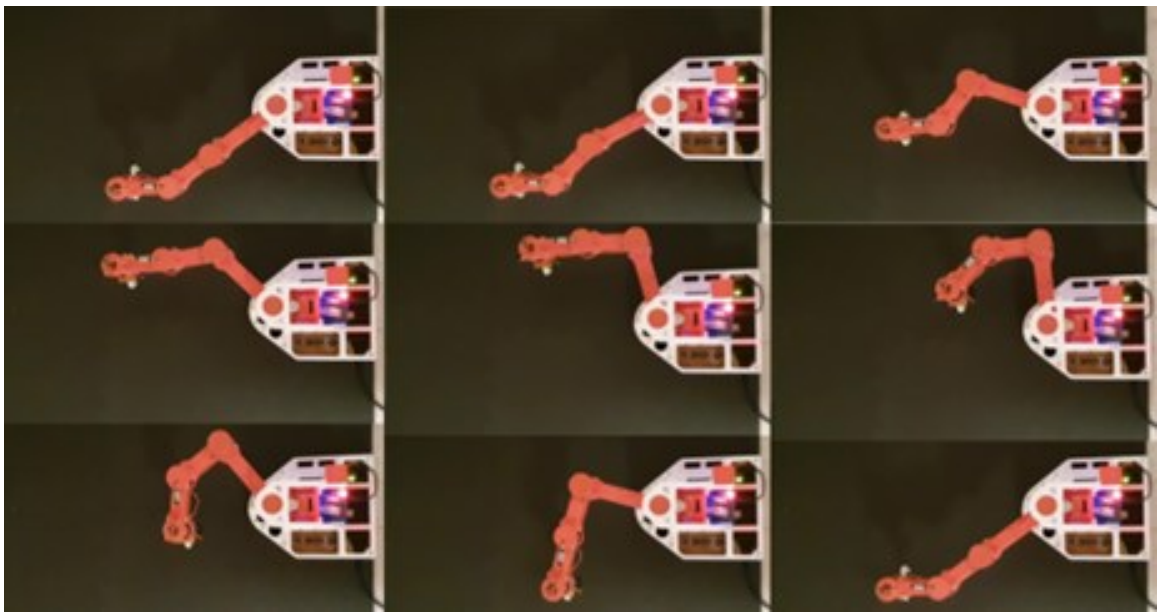
Желаните позиции са 43. Някои от тях са показани в Таблица 4.2. Позициите са записани в обикновен текстов файл. Позициите са в декартови координати, затова на всеки ред от файла трябва да има 5 елемента: координати x и y в работното пространство в mm (от -350 до 350 mm), ориентация на хващача в работната област в градуси (от -90 до 90 градуса), позиция на транслацията z в mm (от 0 до 37 mm), затвореност на хващача g - число от 0 до 9. Всяка позиция трябва да е на нов ред, а нейните елементи да са разделени със запетая (,). Ако някоя от зададените позиции е извън работната област на робота, няма да бъде изпълнена. Файлът с дефинираните позиции се зарежда в потребителския интерфейс и се изпраща към модула ядро на софтуерната система (kernel module), който се изпълнява на контролера Arduino. След това модулът извършва необходимите изчисления и подава съответните команди към моторите на робота. В случая позициите са зададени в декартови координати, затова е необходимо да се реши обратната задача на кинематика, за всяка една от зададените позиции, за да се намерят желаните ставни координати. Обратната задача на кинематиката се решава, използвайки описания алгоритъм във втора глава. Желаната траектория представлява правоъгълник в работната зона на робота.

Таблица 4.2. Зададени позиции за изпълнение на експериментално движение.

№	***x***	***y***	***z***	***g***
1	130	300	37	0
2	110	300	37	0
…	…	300	37	0
16	-150	300	37	0
17	-150	280	37	0
…	-150	…	37	0

22	-150	180	37	0
23	-130	180	37	0
…	…	180	37	0
37	150	180	37	0
38	150	200	37	0
…	150	…	37	0
43	150	300	37	0

Обратната задача на кинематиката е решена правилно и роботът достига успешно зададените позиции. Потребителят може да наблюдава в реално време текущата позиция на робота, чрез уеб-базирания потребителски интерфейс. На Фиг. 4.13 са показани някои от изпълнените ставни конфигурации по време на изпълнение на зададеното движение.



Фиг. 4.13. Изпълнение на зададена траектория.

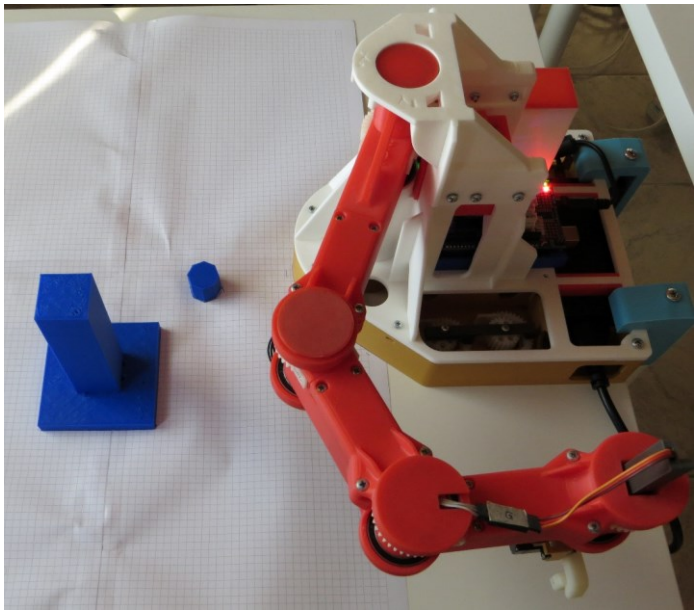
Проведеният експеримент потвърждава, че софтуерната и хардуерната система е подходяща за управление на изследвания робот. Вътрешната памет на микроконтролера Arduino е ограничена, затова би било полезно да се добави външна памет (SD карта) към хардуерната система.

Следващите експерименти имат за цел да покажат приложимостта на предложените алгоритми за управление на траектория при наличие на статични и динамични препятствия в работното пространство на робота.

4.3.2 Заобикаляне на препятствие

За експерименталната верификация на алгоритмите за управление и планиране на траектория се използва създадения прототип на изследвания робот. За валидиране на

предложения алгоритъм в Глава 3.4 за планиране на движение при наличие на статични препятствия е поставена следната задача. Роботът трябва да хване и пренесе определен предмет, докато изпълнява движение от точка до точка и преодолява налични препятствия в работното му пространство. Експерименталната постановка е показана на Фиг. 4.14.



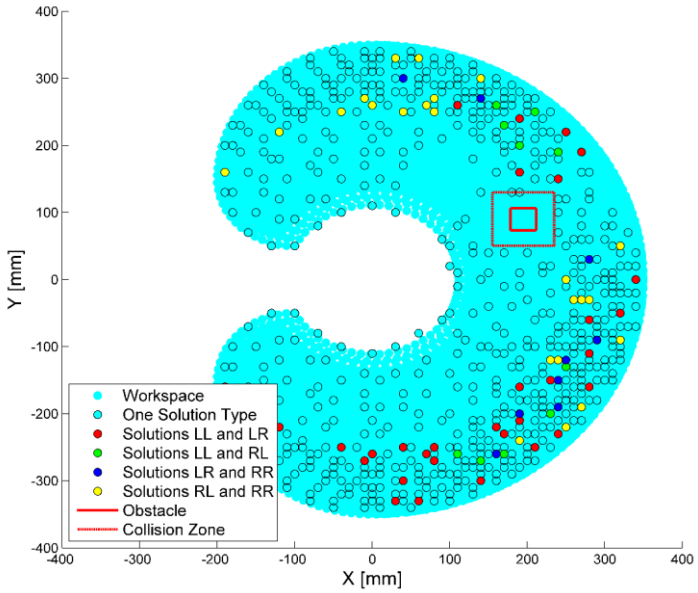
Фиг. 4.14. Експериментална постановка за преодоляване на статични препятствия с 3D принтиран прототип на равнинен робот с допълнителни степени на свобода.

Препятствието е поставено на позиция с координати (195, 90) в равнината O_{xy}. Препятствието има дължина на страната 31 mm и е високо 120 mm. Равнинният робот започва движението си от позиция с координати (-100, 210) в равнината O_{xy} и трябва да достигне предмет, поставен на позиция с координати (120, 10) в равнината O_{xy}. Роботът има ставни ограничения, дефинирани в (2.11).

Първата стъпка от предложения подход е да се създаде крайно множество $\boldsymbol{A}$ от валидни ставни координати. Възможният диапазон от валидни ставни координати се дискретизира линейно и се разглеждат 10 стойности за всяка координата. Това ще даде резултат от максимум 1000 точки за множеството $\boldsymbol{A}$. След проверка на всички възможни конфигурации за възможен удар с препятствие, получаваме множествата $\boldsymbol{A}$ и $\boldsymbol{B}$, всяко с 811 точки. Дължината на страната на сегментите на работното пространство α е избрана да бъде 10 mm. На Фиг. 4.15 са показани различните типове сегменти, означени с точките от множеството $\boldsymbol{B}$ със съответния им тип решение на обратната задача на кинематиката.

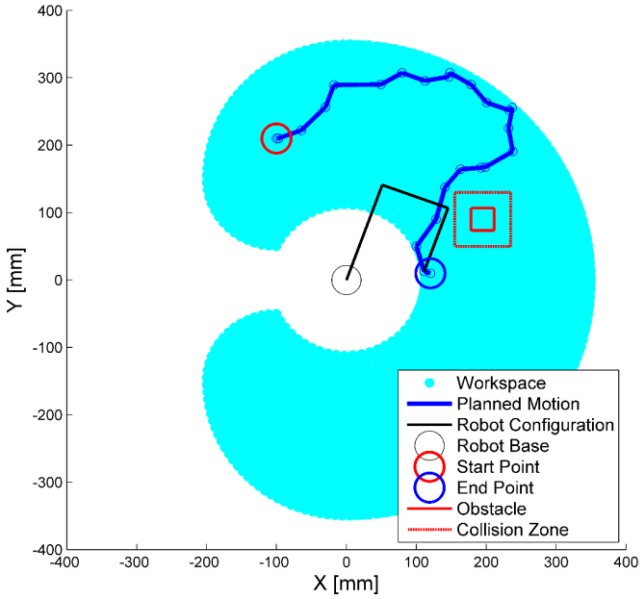
За построяването на графа $\boldsymbol{G}$ нормата на работното пространство α е зададена на 50 (координатите в работното пространство са в mm), а нормата на ставните координати β е 0.8 (ставните координати са в радиани). С тези параметри създаденият неориентиран граф $\boldsymbol{G}$ се състои от 811 върха и 5104 ребра. Алгоритъмът за планиране на движение определи позицията

(-97.18, 210.19) като най-близка до желаната стартова позиция и позицията (111.07, 12.78) като най-близка до желаната крайна позиция. За намиране на път между двете позиции се използва BFS алгоритъма. Той намира път с 22 върха от началната позиция до крайната.



Фиг. 4.15. Точките от работното пространство, които са в графа.

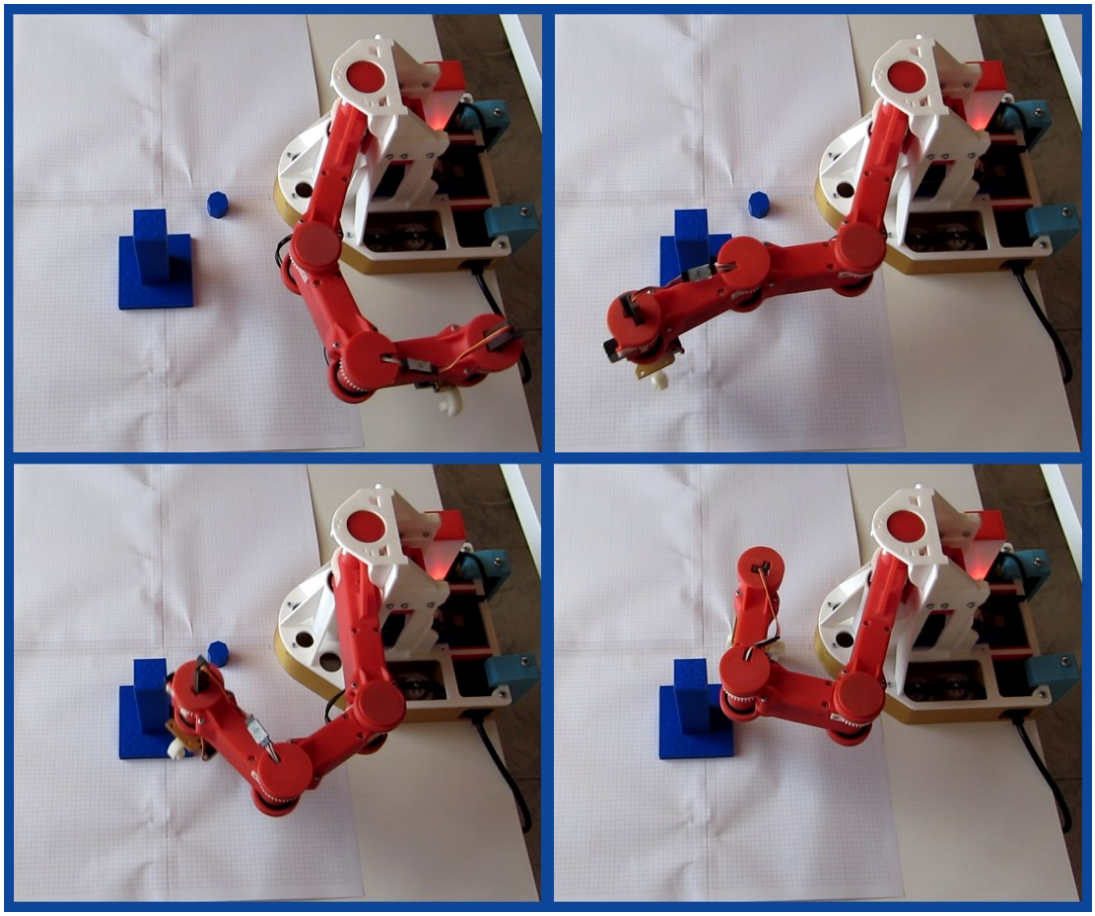
На Фиг. 4.16 е показано планираното движение. Препятствието има удължена дължина на страната от 80 mm, за да се вземат предвид физическите размери на звената на робота.



Фиг. 4.16. Планирано движение от точка до точка.

На Фиг. 4.17 са показани няколко ставни конфигурации, които са близки до препятствието. Алгоритъмът е изпълнен на една нишка на Intel® Core™ i7-4710HQ CPU. Графът е създаден за 19.8 секунди. Бяха необходими 0.019 секунди на BFS алгоритъма, за да планира движението на

робота. Алгоритъмът е тестван и с недостижима точка (100, -300) за планиране на движение. На BFS алгоритъма му бяха необходими 0.24 секунди, за да установи, че не може да планира движение. Това е и максималното необходимо време за планиране на движение от точка до точка.



Фиг. 4.17. Избрани ставни конфигурации от изпълнението на движението.

Резултатите от проведения експеримент потвърдиха, че предложеният алгоритъм може да бъде използван за планиране на движение в реално време от точка до точка при наличие на статични препятствия в работното пространство на робота. От най-много време за изчисления се нуждае процесът за създаване на графа. Но създаването на графа се извършва само веднъж, преди първото планиране на движение.

Следващата глава разглежда проведените експерименти с изследвания равнинен робот с цел валидиране на предложения алгоритъм за управление на робота при наличие на динамични препятствия в работното му пространство.

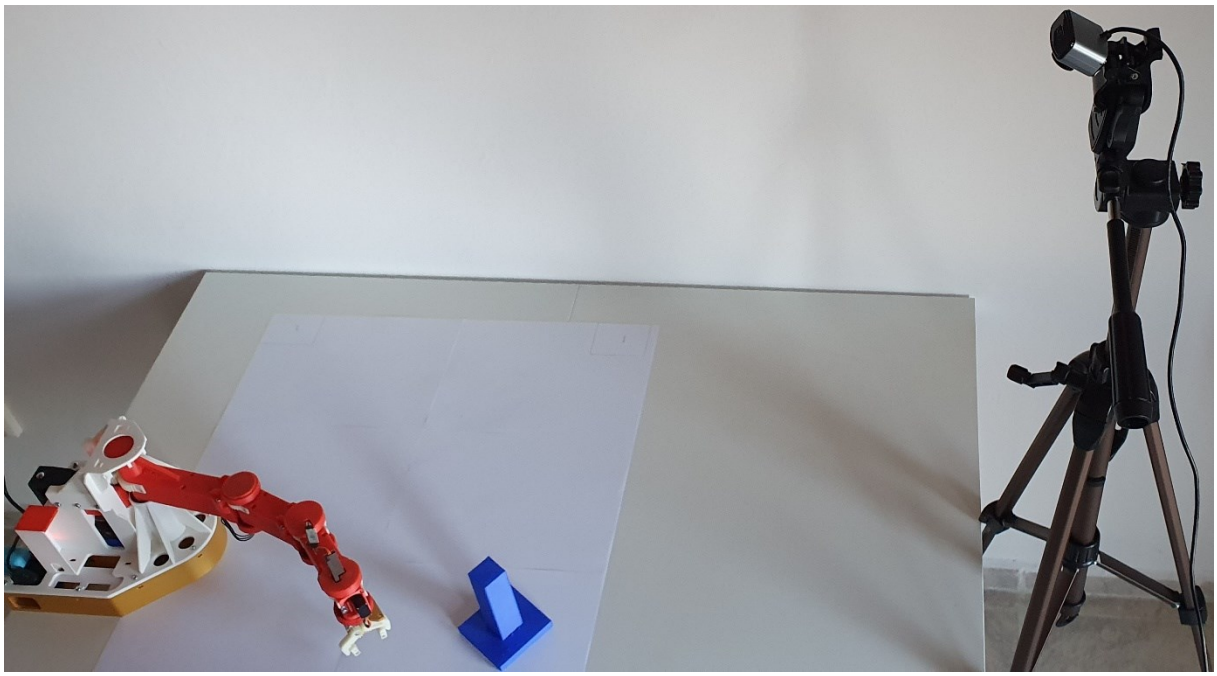
4.3.3 Заобикаляне на препятствия в реално време

След валидиране на предложения алгоритъм за изпълнение на движение при наличие на статични препятствия в работната зона на робота може да се премине към провеждане на

експеримент за изпълнение на предварително зададена траектория при наличие на динамично движещи се препятствия в работното пространство на робота. Алгоритъмът за заобикаляне на динамични препятствия е разширение на алгоритъма за преодоляване на статични препятствия и е представен в трета глава и компютърно валидиран в симулационна среда Webots в Глава 4.1.

Целта на експеримента е да покаже, че за проектирания робот този алгоритъм може да работи в реално време. За разлика от предишния експеримент, в който позицията на препятствието беше предварително дефинирана, в този експеримент нямаме предварителна информация за позицията на препятствието. За да работи алгоритъмът в реално време е необходимо във всеки един момент да може да се получи информация за позицията на препятствието.

За целите на валидация на алгоритъма, без ограничение на общността може да се разгледа препятствие, което е едноцветно и което ще се движи само в равнината O_{xy}. Неговата позиция по z няма да се променя, а височината му ще е фиксирана и константна. Такова препятствие може лесно да бъде открито в реално време, чрез обикновена цветна камера и алгоритми за обработка на изображения. Камерата не е необходимо да е на точно определена позиция, защото можем да направим първоначална калибрация. Важното е да не променя своята позиция по време на управлението и да остане статична. За целите на експеримента е използвана цветна камера Canyon. Камерата се свързва посредством USB към компютър, където изпраща отчетените данни и се извършва тяхната обработка. Експерименталната постановка е показана на Фиг. 4.18. Тя се състои от 3D принтирания робот със ставни ограничения, дефинирани в (2.11) и дължини на звената (2.10), препятствие и камера за разпознаване на динамично движещия се обект.



Фиг. 4.18. Експериментална постановка.

Тъй като експериментът изисква динамично движещи се обекти е необходимо периодично препятствието да променя своята позиция. За тази цел, то ще бъде ръчно преместване.

Задачите, които трябва да се решат, за да може да се изчисли позицията на обекта са следните: **откриване на обекта**, **начин на калибрация** и **намиране на координатите на пространствените координати** в калибрирано изображение. За имплементирането на тези задачи е използван програмният език Python и библиотеката за компютърно зрение OpenCV [97].

Първата задача е откриване на обекта и определяне на координатите на неговия център в изображението. Това може да стане, чрез филтриране по цвета на обекта в HSV пространството, бинаризация и намиране на центъра на най-големия регион в бинаризираното изображение.

След като могат да се определят координатите в кадъра, то може да се премине към калибрацията. Тя се състои в определяне на координатите на обекта в кадъра, когато той е поставен на четири предварително избрани позиции в пространството. Позициите в m са следните: $(0.5, 0.4, 0)$, $(0, 0.4, 0)$, $(0, -0.4, 0)$ и $(0.5, -0.4, 0)$. Те са избрани такива, защото общата дължина на звената на робота е 0.35 m. Когато обектът е по-далече, няма как да има сблъсък и не се налага препланиране на траекторията. След като са определени съответствията на позициите вече може да се премине към същинската работа на алгоритъма.

Когато имаме съответствията на позициите може на всяка стъпка да се вземе само частта от кадъра, която попада в четириъгълника образуван от координатите, които са съответни на калибриращите позиции. В зависимост от разположението на камерата (което не е предварително определено) ще се получи четириъгълник, който не е правоъгълник. Може чрез операция за промяна на перспективата, да коригираме изображението и да го сведем до правоъгълник с правилни пропорции. Целта е да се преобразува от произволен четириъгълник до правоъгълник с размери 500 на 800 пиксела. Това може да стане с функция warpPerspective в OpenCV [98]. Разгледаните четири позиции в пространството, използвани при калибрацията, имат следните координати като съответствия в нетрансформираното изображение: (580,219), (521,51), (180,45), (126,208) и съответните им координати в трансформираното изображение са: (0,0), (499,0), (499,799), (0,799).

След като имаме кадър 500 на 800 пиксела може лесно да се намерят координатите на препятствието в него. Пак филтрираме по цвят и търсим центъра на най-големия регион. Този център ще има координати, които вече лесно могат да се съпоставят на пространствени. Центърът (0, 0) в O_{xy} равнината на робота ще има координати (0, 400) в изображението. Координатите на препятствието в изображението могат да се преобразуват към координати в работното пространство на робота с помощта на уравнения (4.4) и (4.5).

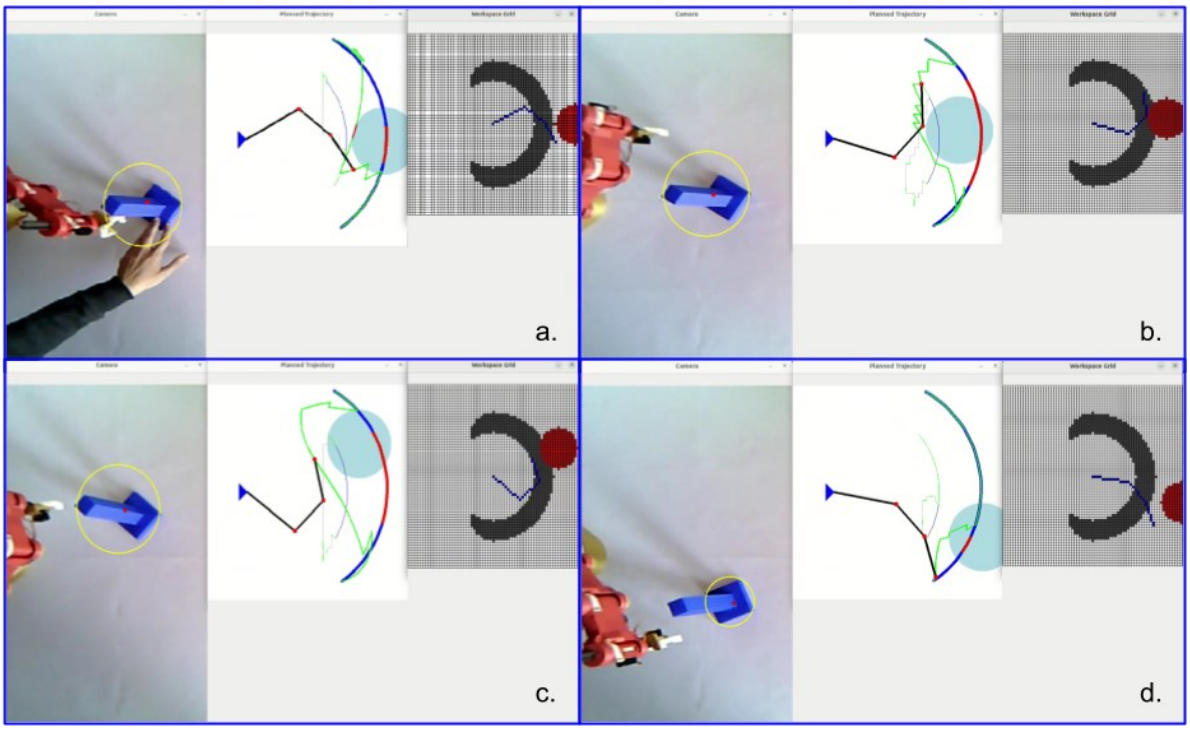
$$obstacle_x = \frac{(500.0 \; - \; object_x)}{500} \; * \; 0.5, \tag{4.4}$$

$$obstacle_y = \frac{(400.0 - object_y)}{400} * 0.4, \quad (4.5)$$

където $obsacle_x$ и $obstacle_y$ са съответно търсените координати x и y в O_{xy} равнината, $object_x$ и $object_y$ са координатите x и y на препятствието в изображението.

На Фиг. 4.19 са показани 4 позиции на робота по време на изпълнението на експеримента. Позицията на препятствието в равнината и съответната позиция в изображението за показаната извадка е: В първата позиция (Фиг. 4.19.a.) координатите на препятствието в ранината са (0.34, 0) и координатите в изображението ще бъдат (157, 403). Във втората позиция (Фиг. 4.19.b.) координатите на препятствието в работното пространство на робота са (0.29, 0.02), а в изображението: (203, 380). Съответно в третата (Фиг. 4.19.c.) и четвъртата (Фиг. 4.19.d.) позиция координатите в равнината са: (0.27, 0.11) и (0.35, -0.1), които се съпоставят на координати: (221, 288) и (148, 505) в изображението.

Всяка една от изобразените позиции се състои от три полета. В първото поле е дадено изображение от камерата, която се използва за разпознаване на препятствието (камерата на управляващата система). Жълтата окръжност върху препятствието означава, че то е открито от управляващата система и са извършени успешно всички описани стъпки по неговото разпознаване. Изпълнението на зададената траектория и препланирането на нова траектория с цел избягване на препятствие се осъществява по описания начин в трета глава и извършения симулационен експеримент в Глава 4.1.

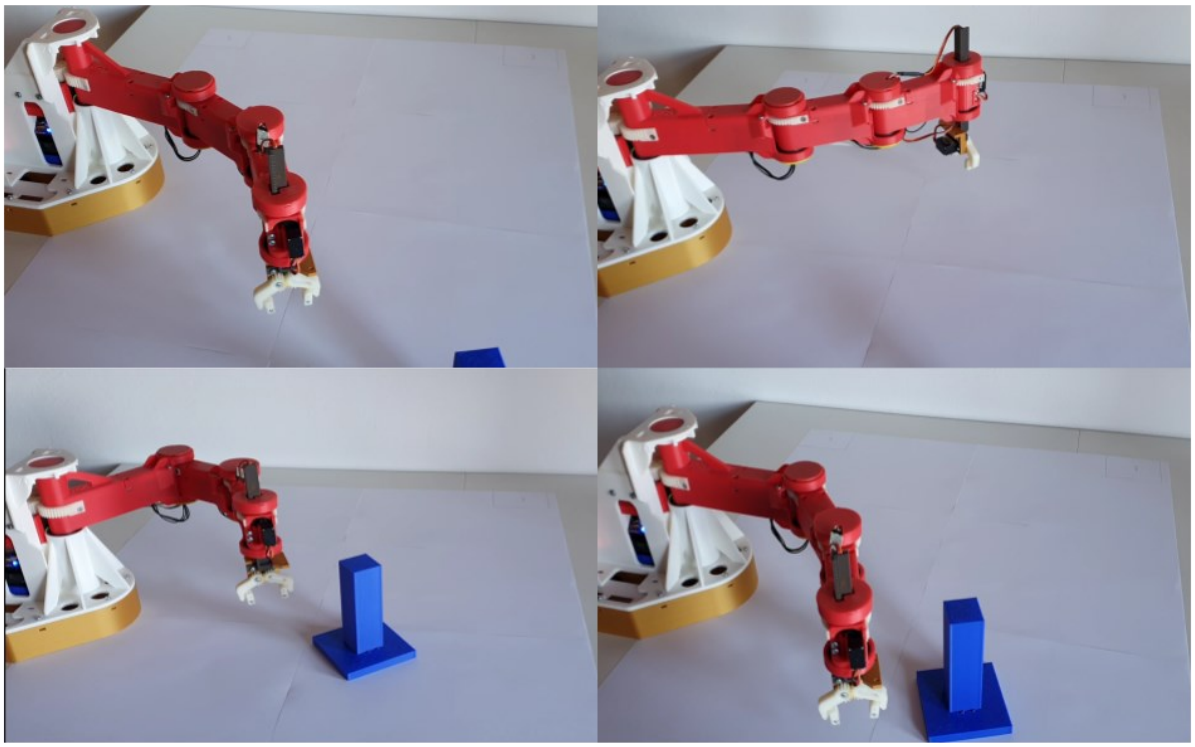


Фиг. 4.19. Избрани позиции от проведения експеримент.

Второто поле от всяка една изобразена позиция илюстрира целевата траектория (синята дъга), препланирания път (зелената линия), частта от желаната траектория, която е блокирана от препятствието (обозначена с червен цвят) и позицията на препятствието (синият кръг).

Както при компютърния експеримент и тук е изобразена мрежата на работното пространство на първите две звена на робота (третото поле от Фиг. 4.19.d.). Клетките, които се използват от алгоритъма за препланиране са оцветени в тъмносиво.

Допълнително е използвана още една камера. Нейната цел е да заснеме експеримента отстрани. На Фиг. 4.20 са показани още четири различни конфигурации на равнинния робот по време на експеримента. В първите две конфигурации препятствието се намира извън работната зона на робота. Затова в този случай роботът изпълнява желаната траектория с изпълнителното си звено. Във вторите две конфигурации препятствието се е появило в работната зона на робота и възпрепятства изпълнението на траекторията. Управляващата система го разпознава и в реално време планира нова траектория, която роботът успешно изпълнява.



Фиг. 4.20. Изпълнение на траектория при налични динамични препятствия в работната зона на робота.

Проведеният експеримент е изпълнен успешно. Управляващата система успешно разпознава появило се препятствие в работната зона на робота и планира нова траектория с цел избягване на сблъсък с него и възможно най-бързо връщане на хващача на робота към целевата траектория. По време на планирането е възможно препятствието да смени своята позиция, управляващата система следи за това и ако е необходимо планира нова траектория. За

планирането на траекторията се използва алгоритъм за търсене в широчина в граф, което прави задачата изчислително ефективна и успешна за изпълнение в реално време.

4.4 Приноси към Глава 4

Тенденцията е все повече индустриални роботи да изпълняват поставените им задачи заедно с други роботи или хора. Това означава, че от роботите се изисква не само да изпълнят коректно и ефективно поставената им задача, но и да заобиколят успешно наличните препятствия в работното им пространство.

Препятствията могат да бъдат както статични, така и постоянно движещи се. При работа в среда със статични препятствия е предварително известна тяхната позиция и тя не се променя във времето. В трета глава беше предложен подход за планиране на движение при наличие на статични препятствия в работното пространство на робота, а в тази глава алгоритъмът е успешно валидиран. За проведения експеримент се използва 3D принтиран робот.

При работа в среда с динамични препятствия, каквито могат да бъдат други роботи или хора няма предварителна информация за техните позиции. Появилите се обекти в работното пространство на робота трябва да бъдат разпознати от камерата на робота и управляващата система на робота трябва да планира нова траектория, ако някое препятствие възпрепятства изпълнението на целевото движение. Алгоритъмът за управление на движението на равнинен робот при наличие на динамични препятствия е разширение на алгоритъма за изпълнение на движение при наличие на статични препятствия. Алгоритъмът създава граф с малък брой върхове и планира траектория, взимайки предвид наличните препятствия. При появило се препятствие, върховете от графа, които представят частта от блокираната траектория се маркират като недостъпни. Управляващата система в реално време препланира нова траектория. За препланирането се използва алгоритъм за търсене в широчина в граф, което прави задачата изчислително ефективна и подходяща за изпълнение в реално време. Алгоритъмът е валидиран, чрез компютърна симулация и реален експеримент с изследвания робот. За компютърната симулация е използван симулационният софтуер Webots. Той позволява създаването на симулационен модел на реалния робот, задаването на ставни ограничения, както и симулиране на различни сензори, в конкретния случай на камера. Провеждането на симулационни експерименти позволява своевременно откриване на грешки или неточности в алгоритъма и тяхното отстраняване. Това би предотвратило евентуални повреди по физическия робот, ако алгоритъмът се тества директно върху него. Всички проведени експерименти показаха, че предложеният подход е подходящ за управление на такъв тип роботи в среда с налични динамични препятствия.

Предложените алгоритми за планиране на движение при наличие на статични и динамични препятствия и проведените експерименти са резултат от работата по публикации [П3], [П4], [П5], [П6] и са докладвани на [Д2], [Д4].

4.5 Заключение

След проведените експерименти с 3D принтирания робот може да се заключи, че избраните хардуерни компоненти са подходящи за управление на робота. Роботът получава изпратените към него команди веднага и може да върне отговор веднага. Може да изпълни желана позиция и ориентация с хващача и да работи с команди в ставни или декартови координати. Потребителят може да следи състоянието на робота и да изпраща команди към него посредством потребителския интерфейс на софтуерната система.

Проведен е експеримент за валидиране на предложения алгоритъм за планиране на движение при наличие на статични препятствия. Изследваният робот успешно достига зададената му целева позиция и пренася желания обект, преодолявайки статично препятствие в работната му зона. Експериментът показа, че първата част от алгоритъма – построяването на графа изисква най-много време за изчисление, но тя се извършва веднъж преди началото на движението. Частта свързана с планиране на движение от точка до точка е изчислително ефективна и планирането може да бъде извършено в реално време.

Предложеният алгоритъм за управление на робота при наличие на динамични препятствия в работната му зона също е валидиран, чрез проведен компютърен и реален експеримент. Експериментът показа, че алгоритъмът е подходящ за управление на роботи с допълнителни степени на свобода. Алгоритъмът създава граф с малък брой върхове и планира траектория, съобразявайки се с наличните препятствия. Предложеният подход е изчислително ефективен и може да бъде приложен при управление в реално време. Тази глава представи резултатите от изпълнението на задачи 4 и 5 от задачите на дисертационния труд.

Заключение

В дисертационния труд са анализирани предимствата и недостатъците на роботите с допълнителни степени на свобода. Мотивиран е изборът за използване на робот с допълнителни степени на свобода, при задачи свързани с преодоляване на препятствия в работната зона на робота. Изготвен е математически модел на робот с допълнителни степени на свобода. Правата задача на кинематиката е решена с помощта на конвенцията на Денавит-Хартенберг, а обратната задача, посредством геометричен метод за решение. Решенията на обратната задача на кинематиката на такъв тип робот могат да се класифицират на няколко типа, в зависимост от знака на ставните координати. В дисертацията са разгледани тези типове решения и как те разделят работното пространство на робота на различни зони. Преходът от една зона в друга може да изисква смяна на типа решение, което може да доведе до отклонение от желаното движение. Затова са изследвани преходни точки, в които смяната на типа решение може да се извърши без отклонение от желания път.

Проектиран и реализиран е прототип на робот с 4 ротационни стави, посредством методите на 3D принтирането. За неговото проектиране е използван CAD софтуерът AutoCAD. Създадена е хардуерна и софтуерна система за управление на изследвания равнинен робот, която се базира на платформите с отворен код Arduino Mega 2560 и Wi-Fi модул ESP8266. Създаден е потребителски интерфейс, през който операторът може да следи състоянието на робота и да изпраща команди към него. Разработен е алгоритъм за планиране на движение. Алгоритъмът взима предвид преходните точки и се базира на теорията на графите.

В дисертацията са предложени и създадени алгоритми за планиране на траектория при наличие на статични и динамични препятствия. Алгоритмите са валидирани, чрез създаване на компютърни експерименти и провеждане на реални експерименти с 3D принтирания прототип на изследвания робот. При изпълнение на дадена задача при наличие на статични препятствия, управляващата система на робота има предварителна информация за позицията на препятствията. Задачата става по-сложна, когато роботът манипулира в среда с динамични препятствия. В този случай управляващата система трябва в реално време да следи за позицията на препятствията и когато дадено препятствие блокира планираното движение е необходимо препланиране на движението на робота и максимално бързо връщане към целевата траектория. Проведените и описани експерименти доказаха, че разработените алгоритми за планиране на траектория при наличие на статични и динамични препятствия са подходящи за управление на такъв тип робот, изчислително ефективни са и това позволява да бъдат приложени в задачи, изискващи работа в реално време.

Всички задачи, които бяха поставени в началото на дисертационния труд са изпълнени. Целта за създаване на математически модел и прототип на равнинен антропоморфен робот с

допълнителни степени на свобода и създаване на алгоритми за управление на неговото движение е постигната.

Перспективи за развитие

Изследванията върху алгоритмите за планиране на траектория при наличие на статични или динамични препятствия ще продължат и за в бъдеще. Може да се изследва по какъв начин изборът на параметрите на алгоритъма за планиране на траектория при наличие на статични препятствия влияе на изпълнението на алгоритъма. Също така, как фазата по построяване на графа може да се реализира с по-висока изчислителна ефективност или дори да се извърши в реално време при наличие на динамични препятствия.

Проведените експерименти показаха, че проектираната хардуерна система е подходяща за управление на такъв тип робот. Но вътрешната памет на контролера Arduino е ограничена. Затова би било добре, за в бъдеще, да се добави външна памет към хардуерната система на робота.

Литература

[1] Br. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, Robotics: Modelling, Planning and Control (1st. ed.), Springer Publishing Company, Incorporated., 2009.

[2] Br. Siciliano and O. Khatib., "Humanoid Robots: Historical Perspective, Overview and Scope," in *Humanoid Robotics*, 2018, pp. 1-6.

[3] A. Goswami and P. Vadakkepat, "Humanoid robotics," in *Springer Netherlands*, 2019.

[4] M. W. Spong, S. Hutchinson and M. Vidyasagar, Robot Modeling and Control, 2020.

[5] J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ред., 2004.

[6] "International Federation of Robotics, Executive Summary World Robotics 2022 Industrial Robots," 2022. [Online]. Available: https://ifr.org/img/worldrobotics/Executive_Summary_WR_Industrial_Robots_2022.pdf. [Accessed 18. 10. 2022].

[7] "Industrial robots - definition and classification," [Online]. Available: https://ifr.org/img/office/Industrial_Robots_2016_Chapter_1_2.pdf. [Accessed 01. 10. 2022].

[8] V. Potkonjak, M. Popović, M. Lazarević and J. Sinanović, "Redundancy problem in writing: From human to anthropomorphic robot arm," in *Systems and Cybernetics, Part B: Cybernetics, IEEE Transactions*, 1999.

[9] T. Petric, A. Gams, N. Likar and L. Zlajpah, „Obstacle avoidance with industrial robots," в *Springer International Publishing*, 2015.

[10] F. Fahimi, Autonomous Robots: Modeling, Path Planning, and Control, 2018.

[11] G. Ballantyne, F. Moll, „The da Vinci telerobotic surgical system: the virtual operative field," 2003.

[12] St. Chiaverini, G. Oriolo and A. Maciejewski, „Redundant Robots," 2016.

[13] A. Dobra, "General classification of robots," in *23rd International Conference on Robotics in Alpe-Adria-Danube Region, IEEE RAAD 2014 - Conference Proceedings. 2015*, 2014.

[14] В. Георгиев, М. Нишева, Б. Бончев, Вградени и автономни системи, 2014.

[15] T. Boiadjiev, G. Boiadjiev, K. Delchev, I. Chavdarov and R. Kastelov, "Orthopedic Bone Drilling Robot ODRO: Basic Characteristics and Areas of Applications," in *Medical Robotics*, 2021.

[16] A. Mueller, "Modern Robotics: Mechanics, Planning, and Control," *IEEE Control Systems Magazine,* vol. 39, no. 6, pp. 100-102, 2019.

[17] "International Federation of Robotics, Executive Summary World Robotics 2020 Service Robots," 2021. [Online]. Available: https://ifr.org/img/worldrobotics/Executive_Summary_WR_Service_Robots_2021.pdf. [Accessed 25. 01. 2022].

[18] В. Георгиев, Патиас, Хр. Христов, Автономните системи: платформи, приложения, перспективи, 2020.

[19] Pandey, A. Kumar and G. Rodolphe, „A Mass-Produced Sociable Humanoid Robot: Pepper: The First Machine of Its Kind," *IEEE Robotics & Automation Magazine,* 2018.

[20] E. Phillips, X. Zhao, D. Ullman and B. Malle, „What is Human-like? : Decomposing Robots' Human-like Appearance Using the Anthropomorphic robot (ABOT) Database," в *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, 2018.

[21] P. Raju, S. Sikka, A. Garg and M. Pandey, "A Brief Review of Recent Advancement in Humanoid Robotics Research," *Mukt Shabd Journal,* vol. IX, no. VI, 2018.

[22] M. Hirose and K. Ogawa, "Honda humanoid robots development. Philosophical transactions," in *Series A, Mathematical, physical, and engineering sciences*, 2007.

[23] H. Bozma and M. Kalalıo, „Multirobot coordination in pick-and-place tasks on a moving conveyor," *Robotics and Computer-Integrated Manufacturing,* pp. 530-538, 2012.

[24] S. Liu, E. Kurniawan, P. Tan, P. Zhang, S. Sun and S. Ye, "Dynamic scheduling for heterogeneous resources with time windows and precedence relation," in *TENCON2017-2017 IEEE Region 10 Conference*, 2017.

[25] S. Saravanan, K. Ramanathan, Ramya and M. Janardhanan, „Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems," *Industrial Robot,* 2020.

[26] Sh. Kajita, H. Hirukawa, K. Harada and K. Yokoi, Kazuhito, Introduction to Humanoid Robotics, 2014.

[27] S. Shigemi, „ASIMO and Humanoid Robot Research at Honda,“ 2017.

[28] Ch.Klein and Br. Blaho, "Dexterity Measures for the Design and Control of Kinematically Redundant Manipulators," *The International Journal of Robotics Research,* vol. 6, pp. 72-83, 1987.

[29] E. Krustev, „Sliding Mode Control of a Redundant Robot Arm in Motion Planning Subject to Joint Space Constraints,“ 2017.

[30] G. Oriolo, M. Cefalo and M. Vendittelli, „Repeatable motion planning for redundant robots over cyclic tasks,“ *IEEE Transactions on Robotics,* pp. 1-14, 2017.

[31] V. Chembuly and H. Voruganti, "Trajectory planning of redundant manipulators moving along constrained path and avoiding obstacles," in *Procedia Computer Science*, 2018.

[32] A. Almarkhi and A. Maciejewski, „Singularity analysis for redundant manipulators of arbitrary kinematic structure,“ в *Proceedings of the 16th International Conference on Informatics in Control, Automation and Robotics*, 2019.

[33] K. Chang and O. Khatib, "Manipulator control at kinematic singularities: A dynamically consistent strategy," in *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems, Human Robot Interaction and Cooperative Robots*, 1995.

[34] J. Ahuactzin and K. Gupta, "The kinematic roadmap: A motion planning based global approach for inverse kinematics of redundant robots," in *Robotics and Automation, IEEE Transactions on*, 1999.

[35] D. Husbands, Phil and St. Nolfi, Handbook of Robotics, Springer, 2008.

[36] J. Hollerbach and Ki Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE Journal on Robotics and Automation,* vol. 3, pp. 308-316, 1987.

[37] A. R. Hirakawa and A. Kawamura, "Trajectory planning of redundant manipulators for minimum energy consumption without matrix inversion," in *Proceedings of International Conference on Robotics and Automation*, Albuquerque, 1997.

[38] H. Voruganti and V. Chembuly, "An optimization based inverse kinematics of redundant robots avoiding obstacles and singularities," in *Advances in robotics Conference*, 2017.

[39] T. Yoshikawa, "Manipulability of robotic mechanisms," in *IEEE International Conference on Robotics and Automation*, 1985.

[40] R. Geraerts and M. Overmars, "A comparative study of probabilistic roadmap planners," *Algorithmic Foundations of Robotics,* vol. V, 2004.

[41] Ch. Zhou, B. Huang and P. Fränti, "A review of motion planning algorithms for intelligent robots," *Journal of Intelligent Manufacturing,* vol. 33, pp. 387-424, 2022.

[42] R. Venkat, "Path Finding - Dijkstra's Algorithm," 2014.

[43] S. Skiena, „Dijkstra’s algorithm,“ в *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, Addison-Wesley, Boston, Mass, USA, 1990, pp. 225-227.

[44] Z. Du, G. Ouyang, J. Xue and Y. Yao, "A Review on Kinematic, Workspace, Trajectory Planning and Path Planning of Hyper-Redundant manipulators," in *10th Institute of Electrical and Electronics Engineers International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 2020.

[45] Fr. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico and L. Jurišica, "Path Planning with Modified a Star Algorithm for a Mobile Robot," in *Procedia Engineering*, 2014.

[46] S. LaValle and J. Kuffner, „Randomized kinodynamic planning,“ *The International Journal of Robotics Research,* 1999.

[47] L. Kavraki, P. Svestka, J. Latombe and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," in *IEEE Transactions on Robotics and Automation*, 2002.

[48] A. Gupta, S. Agrawal, A. Deshmukh, Pr. Bhargava, "A Geometric Approach to Inverse Kinematics of a 3 DOF Robotic Arm," 2018.

[49] I. Chavdarov, V. Nikolov, B. Naydenov and G. Boiadjiev, "Design and Control of an Educational Redundant 3D Printed Robot," in *International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, Croatia, 2019.

[50] Buss, R. Samule , „Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods,“ 2009.

[51] D. Orin and W. Schrader, "Efficient computation of the Jacobian for robot manipulators," *International Journal of Robotics Research,* pp. 66-75, 1984.

[52] J. Baillieul, „Kinematic programming alternatives for redundant manipulators,“ в *IEEE International Conference on Robotics and Automation*, 1985.

[53] M. Kalasariya, V. Patel, A. Thakkar, „Comparative Study of Iterative Inverse Kinematics Methods for Serial Manipulators,“ *International Journal of Engineering and Technical Research,* 2018.

[54] L. Barinka, R. Berka, „Inverse Kinematics - Basic Methods“.

[55] M. R. Gier, „Control of a robotic arm: Application to on-surface 3Dprinting,“ 2015.

[56] E. Krustev, "Passing Through Jacobian Singularities in Motion Path Control of Redundant Robot Arms," in *Proceedings of the 27th International Conference on Robotics in Alpe-Adria Danube Region (RAAD 2018)*, 2018.

[57] Wampler and W. Charles, "Manipulator Inverse Kinematic Solutions Based on Vector Formulations and Damped Least-Squares Methods," in *IEEE Transactions On Systems, Man, And Cybernetics*, 1986.

[58] Br. Siciliano and St. Chiaverini, "Review of the Damped Least-Squares Inverse Kinematics with Experiments on an Industrial Robot Manipulator," in *IEEE Transactions On Control Systems Technology*, 1994.

[59] A. Maciejewski, Ch. Klein, "Numerical Filtering for the Operation of Robotic Manipulators through Kinematically Singular Configuration," *Journal of Robotic Systems, John Wiley & Sons,* vol. V, pp. 527-552, 1998.

[60] I. Duleba, M. Opalka, "A Comparison Of Jacobian–Based Methods Of Inverse Kinematics For Serial Robot Manipulators," *International Journal of Applied Mathematics in Computer science,* vol. 23, pp. 373-382, 2013.

[61] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation,* vol. II, pp. 14-23, 1986.

[62] A. Shukla, R. Sharma and Pr. Muhuri, "A Review of the Scopes and Challenges of the Modern Real-Time Operating Systems," *International Journal of Embedded and Real-Time Communication Systems,* pp. 66-82, 2018.

[63] Benjamin, „Performance Analysis of VxWorks and RTLinux,“ 2001.

[64] S. Baskiyar, "A Survey of Contemporary Real-time Operating Systems," *Informatica,* pp. 233-240, 2005.

[65] P. Hambarde, R. Varma and S. Jha, "The Survey of Real Time Operating System: RTOS," in *2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies*, Nagpur, 2014.

[66] E. Colon, Robotics Frameworks. Introduction to Modern Robotics II, iConcept Press, 2014.

[67] D. Michal, L. Etzkorn, "Comparison of Player/Stage/Gazebo and Microsoft Robotics Developer Studio," in *The 49th ACM Southeast Conference*, 2011.

[68] J. Kerr and K. Nickels, "Robot Operating Systems: Bridging the Gap between Human and Robot," in *The 44th South-Eastern Symposium on System Theory*, 2012.

[69] Ch. Baillie, A. Demaille, Qu. Hocquet, M. Nottale, S. Tardieu, „The Urbi Universal Platform for Robotics,“ в *The International Conference on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR 2008)*, Venice, Italy, 2008.

[70] A. Koubâa, Robot Operating System (ROS): The Complete Reference, vol. III, Springer, 2018.

[71] P. Iigo-Blasco, F. Diaz-del-Rio, C. Romero-Ternero, D. Cagigas-Muiz and S. Vicente-Diaz, "Robotics software frameworks for multi-agent robotic systems development," *Robotics and Autonomous Systems,* pp. 803-821, 2012.

[72] "Webots," [Online]. Available: https://cyberbotics.com/doc/guide/index. [Accessed 25. 03. 2022].

[73] L. Hohl, R. Tellez, O. Michel and A. Ijspeert, "Aibo and Webots: Simulation, Wireless Remote Control and Controller Transfer," *Robotics and Autonomous Systems,* pp. 472-485, 2006.

[74] R. B. Rusu, A. Maldonado, M. Beetz and B. Gerkey, "Extending Player/Stage/Gazebo towards Cognitive Robots Acting in Ubiquitous Sensor-equipped Environments," in *The International Conference on Robotics and Automation (ICRA2007), Workshop for Networked Robot Systems*, Roma, Italy, 2007.

[75] M. Anderson, L. Thaete and N. Wiegand, "Player/Stage: A Unifying Paradigm to Improve Robotics Education Delivery," in *Workshop on research in robots for education at robotics: science and systems conference*, 2007.

[76] F. Espinosa, M. Salazar, F. Valds and A. Bocos, "Communication architecture based on Player/Stage and sockets for cooperative guidance of robotic units," in *The 16th Mediterranean Conference on Control and Automation*, 2008.

[77] A. Hentout, A. Maoudj and Br. Bouzouia, "A survey of development frameworks for robotics," 2016.

[78] Ch. Baillie, A. Demaille, G. Duceux, D. Filliat, Qu. Hocquet and M. Nottale, "Software architecture for an exploration robot based on Urbi," in *The 6th National Conference on Control Architectures of Robots*, Grenoble, France, 2011.

[79] "Robot Operating System," [Online]. Available: http://www.ros.org/. [Accessed 04. 05. 2022].

[80] "Orocos," [Online]. Available: http://www.orocos.org/. [Accessed 04. 05. 2022].

[81] O. Michel, "Cyberbotics ltd. webots professional mobile robot simulation," *International Journal of Advanced Robotics Systems,* vol. I, pp. 39-42, 2004.

[82] D. Kortenkamp, R. Simmons and D. Brugali, „Robotic Systems Architectures and Programming," 2016.

[83] M. Hapner, R. Burridge, R. Sharma, J. Fialli and K. Stout, „Java Message Service," 2013.

[84] И. Чавдаров, Моделиране на роботи с 3D принтер, "Св. Климент Охридски".

[85] C. Ventola, „Medical Applications for 3D Printing: Current and Projected Uses," 2014.

[86] "AutoCAD," [Online]. Available: https://www.autodesk.com/prod. [Accessed 04. 05. 2022].

[87] "OpenScad," [Online]. Available: https://openscad.org/. [Accessed 04. 05. 2022].

[88] "Blender," [Online]. Available: https://www.blender.org/. [Accessed 04. 05. 2022].

[89] И. Чавдаров, В. Павлов, А. Вацкичев, В. Николов, Ръководство за проектиране на роботи, Издателство на ТУ.

[90] G. Boiadjiev, E. Krastev, I. Chavdarov I, L. Miteva , "A Novel, Oriented to Graphs Model of Robot Arm Dynamics," *Robotics,* vol. 10, no. 128, 2021.

[91] Vinogradov, A. Kobrinski, Y. Stephenko, L. Tives, "Details of Kinematics of Manipulators with the Method of Volumes," *Mekanika Mashin,* vol. I, no. 5, pp. 5-16, 1971.

[92] "Smart Servo Motors HerkuleX DRS-0101," [Online]. Available: https://www.robotshop.com/en/herkulex-drs-0101-robot-servo.html. [Accessed 01. 11. 2022].

[93] "Homepage FeeTech," [Online]. Available: http://www.feetechrc.com/. [Accessed 01 11 2022].

[94] "Arduino Mega 2560," [Online]. Available: https://store.arduino.cc/arduino-mega-2560-rev3. [Accessed 01. 11. 2022].

[95] "Wemos D1 mini," [Online]. Available: https://docs.zerynth.com/latest/official/board.zerynth.wemos_d1_mini/docs/index.html. [Accessed 01. 11. 2022].

[96] "WebSockets," [Online]. Available: https://www.fullstackpython.com/websockets.html. [Accessed 01. 11. 2022].

[97] "OpenCV," [Online]. Available: https://opencv.org/. [Accessed 20. 11. 2022].

[98] "OpenCV getPerspective Transform Example," [Online]. Available: https://pyimagesearch.com/2014/08/25/4-point-opencv-getperspective-transform-example/. [Accessed 20. 11. 2022].

Приноси на дисертацията

Взимайки предвид работата по дисертацията и резултатите, получени от проведените изследвания и представени в дисертационния труд, могат да се формулират следните приноси:

Научноприложни приноси

- Създаден е подход за класифициране по тип на решенията на обратната задача на кинематиката за равнинен робот с допълнителни степени на свобода. (Глава 2, [П1], [Д3]).
- Извършен е анализ на работното пространство на равнинен робот с допълнителни степени на свобода в зависимост от наличните препятствия (Глава 2, [П4], [Д4]).
- Изследван е ъгълът на сервис в работното пространство на равнинен робот с допълнителни степени на свобода (Глава 2, [П2]).
- Създаден е алгоритъм за планиране на траектория на равнинен робот с допълнителни степени на свобода и ограничено ставно пространство, базиран на теорията на графите (Глава 3, [П1], [Д3]).
- Създаден е подход за планиране на движение при наличие на статични препятствия за равнинен робот с допълнителни степени на свобода (Глава 3, [П4], [Д4]).
- Създаден е алгоритъм за избягване в реално време на динамични препятствия в работното пространство на равнинен робот с допълнителни степени на свобода (Глава 3, [П5]).

Приложни приноси

- Проектирана е хардуерна и софтуерна система за управление на равнинен робот с допълнителни степени на свобода (Глава 3, [П3], [Д2]).
- Създаден е компютърен експеримент на разработените методи за планиране на траектория с помощта на симулационен софтуер Webots (Глава 4, [П6]).
- Проведени са реални експерименти с 3D принтиран прототип на равнинен робот с допълнителни степени на свобода с цел верификация на алгоритмите за планиране на траектория при наличие на статични или динамични препятствия в работното пространство на робота (Глава 4, [П4], [П5], [Д4]).

Декларация за оригиналност

Декларирам, че представената във връзка с провеждането на процедура за придобиване на образователна и научна степен „доктор“ в Софийски университет „Св. Климент Охридски“ дисертация на тема: „Моделиране и управление на антропоморфен модел на робот“ е мой труд.

Цитиранията на всички източници на информация, текст, илюстрации, таблици, изображения и други са обозначени според стандартите.

Резултатите и приносите на проведеното дисертационно изследване са оригинални и не са заимствани от изследвания и публикации, в които нямам участие.

Подпис: ..……

/Любомира Митева/

Публикации към дисертацията

П1. **Lyubomira Miteva**, Ivan Chavdarov, Kaloyan Yovchev, *Trajectory Planning for Redundant Robotic Manipulators with Constrained Joint Space*, 2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2020, DOI: 10.23919/SoftCOM50211.2020.9238296, ISBN: 978-953-290-099-6, ISSN: 1847-358X, Ref Scopus, Ref IEEE Xplore, https://ieeexplore.ieee.org/document/9238296.

П2. **Lyubomira Miteva**, Galya Pavlova, Roumen Trifonov, Kaloyan Yovchev, *Manipulability Analysis of Redundant Robotic Manipulator*, ACM International Conference Proceeding Series, 21st International Conference on Computer Systems and Technologies, CompSysTech 2020, Pages: 135-140, DOI: 10.1145/3407982.3407987, ISBN 978-145037768-3, Ref Scopus, SJR (0.182 – 2020), https://dl.acm.org/doi/10.1145/3407982.3407987 (Наградена за най-добър доклад в секцията).

П3. **Lyubomira Miteva**, Kaloyan Yovchev, *Hardware and Software Design for Redundant Robotic Manipulators*, CEUR Workshop Proceedings, 2020, Volume: 2656, Pages: 167-180, ISSN: 1613-0073, Ref Scopus, SCOPUS, SJR (0.177 – 2020), https://ceur-ws.org/Vol-2656/paper17.pdf.

П4. **Lyubomira Miteva**, Kaloyan Yovchev, Ivan Chavdarov, *Point-to-point Motion Planning with Obstacle Avoidance for Hard Constrained Redundant Robotic Manipulators*, 2021 XXX International Scientific Conference Electronics (ET), 2021, DOI: 10.1109/ET52713.2021.9579820, ISBN: 978-166544518-4, Ref Scopus, Ref IEEE Xplore, https://ieeexplore.ieee.org/document/9579820.

П5. Kaloyan Yovchev, **Lyubomira Miteva**, *Real-time Trajectory Replanning for Dynamic Obstacles Avoidance for Robotics Manipulators*, ACM International Conference Proceeding Series, 23rd International Conference on Computer Systems and Technologies, 2022, Pages: 45–50, DOI: 10.1145/3546118.3546135, ISBN 978-145039644-8, Ref Scopus, SJR (0.232 – 2021), https://dl.acm.org/doi/abs/10.1145/3546118.3546135 (Наградена за най-добър доклад в секцията).

П6. **Lyubomira Miteva**, Kaloyan Yovchev, Denis Chikurtev, *Software and Hardware Infrastructure for Research and Development of Intelligent Control for Robotic Manipulators*, 2022 XXXI International Scientific Conference Electronics (ET), 2022, DOI: 10.1109/ET55967.2022.9920270, ISBN: 978-166549878-4, Ref Scopus, Ref IEEE Xplore, https://ieeexplore.ieee.org/document/9920270.

Доклади

Д1. **Manipulability Analysis of Redundant Robotic Manipulator**, международна конференция CompSysTech 2020. http://www.compsystech.org/_cst20

Д2. **Hardware and Software Design for Redundant Robotic Manipulators**, международна конференция ISGT 2020. https://isgt.fmi.uni-sofia.bg

Д3. **Trajectory Planning for Redundant Robotic Manipulators with Constrained Joint Space**, международна конференция SoftCom 2020. http://softcom2020.fesb.unist.hr

Д4. **Point-to-point motion planning with obstacle avoidance for hard constrained redundant robotic manipulator**, международна конференция Electronics – ET 2021, гр. Созопол. https://e-university.tu-sofia.bg/e-conf/?konf=24

Участие в проекти, свързани с темата на дисертацията

- **Изследване и развитие на алгоритми със самообучение за взаимодействие между индустриални роботи и обекти в работната зона**, № КП-06-М47/5 от 27.11.2020 г., Фонд научни изследвания, МОН.
- **Изследване и моделиране на нови роботи чрез нетрадиционни технологии и материали**, ФНИ № 17/10 от 12.12.2017 г., Фонд научни изследвания, МОН.
- **Разработка на методи и алгоритми за управление на манипулационни роботи**, ФНИ на СУ № 80-10-23/18.03.2020 г.
- **Разработка на методи и алгоритми за управление на манипулационни роботи**, ФНИ на СУ № 80-10-89/2021.
- **Разработка на методи и алгоритми за управление на манипулационни роботи**, ФНИ на СУ № 80-10-70/10.05.2022