Model based motion planning for manipulation with heterogeneous robotic systems under constraints

Ivanović, Antun

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Antun Ivanović

MODEL BASED MOTION PLANNING FOR MANIPULATION WITH HETEROGENEOUS ROBOTIC SYSTEMS UNDER CONSTRAINTS

DOCTORAL THESIS

Zagreb, 2023

University of Zagreb

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Antun Ivanović

MODEL BASED MOTION PLANNING FOR MANIPULATION WITH HETEROGENEOUS ROBOTIC SYSTEMS UNDER CONSTRAINTS

DOCTORAL THESIS

Supervisor: Associate Professor Matko Orsag, PhD

Zagreb, 2023

Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Antun Ivanović

NA MODELU ZASNOVANO PLANIRANJE GIBANJA HETEROGENIH ROBOTSKIH SUSTAVA ZA MANIPULIRANJE UZ OGRANIČENJA

DOKTORSKI RAD

Mentor: izv. prof. dr. sc. Matko Orsag

Zagreb, 2023.

Doctoral thesis is written at the University of Zagreb, Faculty of Electrical Engineering and Computing, Department of Control and Computer Engineering.

Supervisor: Associate Professor Matko Orsag, PhD

Doctoral thesis has 161 pages

Dissertation No.: _____

About the Supervisor

Matko Orsag is an associate professor at the University of Zagreb Faculty of Electrical Engineering and Computing (UNIZG-FER). He has been involved as a researcher in various projects financed by the government and industry. In 2011/2012, he worked as a visiting researcher at the Drexel University, Philadelphia, USA as a recipient of the Fulbright exchange grant.

As a researcher, he participated in several national and international research projects in the field of robotics, control, and automation. Currently, he is working as a researcher in several EU projects: AeRoTwin, ENCORE, RoboCom++. He is the principal investigator of project SPECULARIA, funded by the Croatian Science Foundation, and project ENDORSE, funded through the H2020 framework. He authored and coauthored over 30 scientific and professional papers, including journal and conference papers, as well as a monograph and a book chapter in the field of unmanned aerial systems and robotics.

He is serving as an associate editor of AUTOMATIKA – Journal for Control, Measurement, Electronics, Computing, and Communications. He is a member of IEEE and euRobotics Aerial Robotics Topic Group. He is currently a co-chair of Croatian section of IEEE RAS.

He is involved as a reviewer in journals and at conferences, and as an editor and guest editor in several scientific journals (Automatika, JINT, etc.). He is a member of the IEEE, currently serving as the chairman of the IEEE Robotics and Automation Society, Section IEEE Croatia, and a member of topical group for aerial robotics. He is a member of Scientific and Expert Council of the Nikola Tesla Innovation Center (ICENT). In 2019, he received the young scientist award of the Croatian Technical Academy "Vera Johanides".

O mentoru

Matko Orsag je izvanredni profesor na Sveučilištu u Zagrebu, Fakultet Elektrotehnike i Računarstva (UNIZG-FER). Kao istraživač, bio je uključen u brojne projekte financirane državnim sredstvima te industrijske projekte. U razdoblju 2011/2012, kao dobitnik sredstava za razmjenu u sklopu programa Fulbright, radio je kao gostujući istraživač na sveučilištu Drexel, Philadelphia, Sjedinjene Američke Države.

Kao istraživač sudjelovao je u nekoliko nacionalnih i internacionalnih istraživačkih projekata u polju robotike, upravljačkih algoritama te automatizacije. Trenutačno je zaposlen kao istraživač na nekoliko EU projekata: AeRoTwin, ENCORE, RoboCom++. Glavni je istraživač na projektu SPECULARIA financiranom od strane Hrvatske Zaklade za Znanost te na projektu ENDORSE financiranom kroz program Obzor 2020. Kao autor i koautor objavio je preko 30 znanstvenih i profesionalnih članaka, uključujući članke u časopisima i konferencijama, kao i monografiju te poglavlje u knjizi na temu bespilotnih letjelica i robotike.

Trenutno je urednik časopisa AUTOMATIKA - Journal for Control, Measurement, Electronics, Computing, and Communications. Član je IEEE te euRobotics Aerial Robotics Topic grupe kao i supredsjedatelj hrvatskog ogranka IEEE RAS.

Angažiran je kao recenzent u časopisima i na konferencijama, te kao urednik i gostujući urednik nekoliko časopisa (Automatika, JINT i dr.). Član je IEEE-a, trenutno služi kao predsjedavajući ogranka Društva IEEE Robotics and Automation Society Sekcije IEEE Hrvatska i član tematske grupe za zračnu robotiku. Član je Znanstvenog i stručnog vijeća Inovacijskog centra Nikola Tesla (ICENT). Godine 2019. dobio je nagradu za mladog znanstvenika Hrvatske tehničke akademije „Vera Johanides“.

Acknowledgements

I would like to express my sincere gratitude to all those who have supported me throughout my journey in completing this thesis. I would like to extend my heartfelt thanks to my advisor associate professor Matko Orsag, for their guidance, support, and encouragement throughout this project. Their expertise and invaluable advice have been instrumental in shaping the direction of my research.

Furthermore, I would like thank my colleagues at LARICS (Laboratory for Robotics and Intelligent Control Systems) for their support throughout my research. Their expertise, resources, and facilities have been essential to the completion of this work.

I would like to extend my gratitude to my family and friends for their love, support, and encouragement throughout my academic journey. Their unwavering belief in me and my abilities has been a source of strength and inspiration.

Thank you all for your support and belief in me.

Abstract

The main focus of this thesis is motion planning for unmanned aerial manipulators. Such vehicles consist of an unmanned aerial vehicle, equipped with a manipulator. In this thesis, the target sub-class of unmanned aerial vehicles are multirotors, which feature multiple propellers producing thrust. Most popular type of such vehicles are co-planar multirotors, where the thrust produced by all rotors is directed along the vehicle body z axis. Due to the multirotors' limited payload, the manipulator design is required to be lightweight. This requirement is most commonly resolved by limiting the number of manipulator joints to reduce weight, which calls for different manipulator design depending on the task at hand.

The core problem this thesis tackles are the effects produced by the underactuated nature of co-planar multirotors. To change their position, such vehicles first need to execute angular motion. This in turn displaces the manipulator end-effector, which needs to be addressed to successfully manipulate an object and complete the given task. By using the manipulator null space, it is possible to achieve the desired end-effector configuration, even though the multirotor attains some tilt angle. One way to correct the end-effector configuration is online, with an inverse kinematics solver in the loop. However, the inverse kinematics solution might not exist for a given end-effector configuration. Therefore, this thesis incorporates the aerial manipulator dynamical model in the motion planning procedure. First, a trajectory is planned without considering the roll and pitch angles of the multirotor. It is then executed in a simulation that consists of the aerial manipulator mathematical model and the controller structure. While executing the trajectory, the previously unknown roll and pitch angles are recorded. Based on the obtained full state trajectory, the end-effector configuration is corrected by employing the manipulator null space. The final, corrected trajectory is then executed by the system. This method is extensively tested in both simulation environment and real world experiments, employing a classical control, as well as the impedance force control.

The formerly described method is extended to heterogeneous multi-robot systems manipulating a common object. In such a case, the dynamical model of each robot is observed separately, and is then coupled through the common object being manipulated. To evaluate this method, a set of simulation trials with various heterogeneous aerial manipulator combinations is conducted. The goal is to determine which aerial manipulators, or combinations of aerial manipulators, is suitable in different situations. For a fair comparison, a set of benchmark trajectories is planned for the manipulated object configuration. The evaluation metric focuses on the object itself, determining the relative error between the end-effector configurations.

Apart from unmanned aerial manipulators, flying hands are also a popular configuration. Instead of having a complex robotic arm, such vehicles feature only an end-effector attached to the multirotor body. The main idea is employing a flying hand in a fire extinguishing scenario,

where a payload filled with a fire extinguishing agent is launched in the fire. Based on the target provided in the world coordinate system, a set of suitable launch configurations is obtained for releasing the fire extinguishing agent. A dynamically challenging trajectory is planned towards the launch point, considering the vehicle safety both before and after launching the payload. The method is extensively tested in both simulation and real world experimental setup.

There are three main contributions of this thesis:

- *Unmanned aerial manipulator end-effector motion planning method based on the dynamical model of the system* (Chapter 5)
- *A heterogeneous multi-robot system motion planning method based on the coupled dynamical model for cooperative manipulation* (Chapter 6)
- *Unmanned aerial manipulator motion planning method with dynamical constraints at the release point for precision ballistic airdrop* (Chapter 7)

Keywords: Unmanned Aerial Vehicle, Aerial Manipulation, Motion Planning, Multi-Robot Systems

Prošireni sažetak

Glavni fokus ove doktorske disertacije je planiranje gibanja bespilotnih zračnih manipulatora. Takvi sustavi sastoje se od bespilotne letjelice opremljene robotskom rukom. U sklopu ove disertacije, ciljana podskupina bespilotnih letjelica su višerotorske letjelice koje proizvode potisak pomoću više propelera. Najpopularniji tip takvih sustava su koplanarne višerotorske letjelice kod kojih svi propeleri proizvode potisak u smjeru z osi tijela letjelice. Zbog ograničene nosivosti, potrebno je osigurati izradu manipulatora od lakih materijala. Da bi se smanjila masa manipulatora, najčešće se ograničava broj stupnjeva slobode, što u konačnici zahtjeva dizajn manipulatora prilagođen zadatku koji je potrebno izvesti.

Srž problema kojih se ova disertacija dotiče su posljedice svojstva podaktuiranosti koplanarnih višerotorskih letjelica. Kako bi se takve letjelice mogle kretati u prostoru, potrebno je nagnuti tijelo letjelice. Samim naginjanjem tijela se pomiče i alat manipulatora, što treba uzeti u obzir kako bi se uspješno odradio zamišljeni zadatak. Korištenjem nul prostora manipulatora moguće je postići željenu konfiguraciju alata u prostoru, čak i dok je tijelo letjelice nagnuto za neki kut. Jedan od načina ispravljanja konfiguracije alata je korištenjem inverzne kinematike u upravljačkoj petlji za vrijeme izvođenja planiranog gibanja. Glavni nedostatak takvog pristupa je da rješenje za željenu konfiguraciju alata ne postoji. Kako bi se riješio navedeni problem, ova disertacija uključuje dinamički model bespilotnog zračnog manipulatora u proceduru planiranja gibanja. Prvi korak je planiranje željene trajektorije vrha alata manipulatora uz pretpostavku da su kutevi valjanja i poniranja letjelice jednaki nuli. Inicijalna trajektorija se izvodi pomoću matematičkog modela sustava, uz upravljačku strukturu koja odgovara konačnom sustavu. Za vrijeme izvođenja trajektorije se spremaju prethodno nepoznate veličine kuteva valjanja i poniranja, na temelju kojih se konstruira vektor stanja sustava pomoću kojega se korigira konfiguracija vrha alata kroz nul prostor manipulatora. Konačna trajektorija dobivena na ovaj način se nakon toga šalje na sustav. Opisana metoda je ispitana u simulacijskom okruženju te kroz eksperimente u stvarnom svijetu, uz korištenje standardnog upravljanja višerotorskom letjelicom i impedantnog upravljanja silom.

Nadalje, prethodno opisana metoda je proširena za heterogene višerobotske sustave koji manipuliraju istim objektom. U tom slučaju dinamički model svakog robota se uzima u obzir zasebno te se združeni model promatra kroz manipulirani objekt. Za evaluaciju proširene metode upravljanja izvedeni su simulacijski testovi s različitim kombinacijama zračnih manipulatora. Cilj evaluacije je određivanje koji od navedenih manipulatora su pogodni za različite situacije. Kako bi usporedba bila objektivna, osmišljen je skup trajektorija koji se planira za centar mase prenošenog objekta. Na taj način se gledaju rotacijske i translacijske greške u slijeđenju trajektorije kroz relativnu pogrešku između konfiguracija alata više robota.

Izuzev bespilotnih zračnih manipulatora, zračne ruke su također česta konfiguracija. Um-

jesto da se na letjelicu ugrađuje kompleksan manipulator s više stupnjeva slobode, u tom slučaju se ugrađuje samo alat. Navedena konfiguracija je ispitana kroz scenarij gašenja požara, gdje je potrebno dostaviti teret punjen sredstvom za gađenje u vatru. Ulaz u algoritam je pozicija mete, odnosno lokacija požara, na temelju koje se proračunava skup konfiguracija za izbacivanje tereta. Nakon toga se planira trajektorija prema točki izbacivanja, s dinamički zahtjevnim parametrima, uzimajući u obzir sigurnost sustava prije i poslije izbacivanja tereta. Navedena metoda je u širokom opsegu ispitana kroz simulaciju i eksperimente.
Disertacija je podijeljena u osam poglavlja.

U prvom poglavlju dan je uvod u područje bespilotnih letjelica i zračnih manipulatora. Prikazane su osnove bespilotnih letjelica, od konstrukcije i dizajna do upravljanja niže i više razine. Bespilotne letjelice su idealan izbor za mnoštvo pasivnih zadataka poput snimanja, slikanja, skupljanja podataka o okolini itd. Ugrađivanjem manipulatora na takve letjelice postiže se pomak prema aktivnim zadacima. Umjesto da samo promatraju i skupljaju podatke o okolini, zračni manipulatori mogu djelovati na okolinu i mijenjati istu.

U drugom poglavlju je prikazan pregled područja od interesa za ovu disertaciju. Upravljanje je razloženo na bespilotne letjelice i zračne manipulatore, s dodatnim pregledom upravljanja silom kroz impedanciju. Za planiranje gibanja pregledano je područje planiranja putanje i trajektorije, ponovo za bespilotne letjelice i zračne manipulatore. Poglavlje je završeno pregledom područja za izbacivanje tereta.

U trećem poglavlju predstavljeni su matematički model i upravljanje zračnim manipulatorima. Matematički model razdvojen je na kinematiku i dinamiku. Dinamički model manipulatora izveden je kroz Newton-Eulerov formalizam, gdje je prvi korak proračun brzina i akceleracija članaka manipulatora, od prvog prema zadnjem članku. Drugi korak je proračun sila i momenata zglobova, u suprotnom smjeru od prvog koraka. Matematički model letjelice izveden je na temelju suma sila i momenata koji djeluju na tijelo letjelice. U drugom dijelu poglavlja prezentirano je upravljanje zračnim manipulatorom. Upravljanje letjelicom temelji se na kaskadnom upravljanju kutevima i kutnim brzinama (niska razina) te kaskadnom upravljanju pozicijom i brzinom (visoka razina). Upravljanje manipulatorima nije razmatrano u detalje te su navedeni istraživački materijali s detaljnim opisima upravljanja manipulatorima.

U četvrtom poglavlju prikazano je planiranje gibanja zračnog manipulatora. Prvi korak u planiranju gibanja je planiranje putanje. U ovom slučaju koristi se Rapidly-exploring Random Trees (RRT*) algoritam, koji na temelju početne i konačne konfiguracije robota planira putanju izbjegavajući prepreke. Valja napomenuti kako RRT* algoritam efikasno pretražuje visokodimenzionalne prostore, poput onih koji opisuju stanje zračnog manipulatora. Kako bi to bilo moguće, potrebno je prikazati prostor u formatu pogodnom za računala. Ovdje se koristi OctoMap reprezentacija prostora koja nudi efikasno pretraživanje uz ograničeno korištenje radne memorije računala. Sljedeći korak je planiranje trajektorije, odnosno vremenska interpolacija

putanje. U ovoj disertaciji koristi se Time Optimal Path Parameterization (TOPP) algoritam, koji je objašnjen na jednostavnom primjeru.

U petom poglavlju predstavljena je metoda planiranja trajektorije bespilotnog zračnog manipulatora temeljena na dinamičkom modelu sustava. Prvi korak je planiranje putanje na temelju početne i konačne konfiguracije zračnog manipulatora. Putanja se planira u visokodimenzionalnom vektorskom prostoru koji obuhvaća stupnjeve slobode letjelice i manipulatora. Zatim se na temelju putanje planira trajektorija pomoću TOPP-RA algoritma te se izvodi kroz dinamički model i strukturu upravljanja. Tijekom izvođenja spremaju se nepoznati kutevi valjanja i poniranja kako bi se dobilo cjelokupno stanje zračnog manipulatora kroz čitavu trajektoriju. Koristeći inverznu kinematiku i nul prostora manipulatora, konfiguracija vrha alata se korigira koristeći cjelokupno stanje sustava. Time se proračunavaju novi kutevi zakreta svakog zgloba manipulatora te se korigirana trajektorija izvodi na zamišljenom sustavu.

Opisana metoda planiranja prvo je testirana u simulacijskom okruženju. S obzirom da na pogrešku praćenja trajektorije vrha alata utječe kretanje tijela letjelice, u prvom dijelu analize postavljene su razne kombinacije ograničenja brzine i akceleracije letjelice. Potom su planirane i izvršene trajektorije za svaku kombinaciju dinamičkih ograničenja, čime su dobiveni rezultati odstupanja praćenja vrha alata u ovisnosti o brzini i akceleraciji. Iz tih rezultata može se zaključiti da povećavanje dinamičkih ograničenja također povećava i odstupanje vrha alata.

Na temelju dobivenih odstupanja izabrana su ograničenja brzine i akceleracije za provedbu simulacija u kompleksnom okruženju. Provedene su simulacije umetanja šipke u cijev na primjeru jednostavnog modela tvornice gdje su tri cijevi od interesa: vodoravna cijev polumjera $r = 14cm$, cijev postavljena pod kutem od 60° i promjera $r = 21cm$, te cijev promjera $r = 4cm$. Navedene cijevi određivale su konačnu konfiguraciju zračnog manipulatora, dok je početna konfiguracija izabrana tako da se putanja i trajektorija planiraju prolazeći kroz tvornicu. Eksperimentalna analiza provedena je na AscTec NEO letjelici na koju je ugrađen manipulator s tri stupnja slobode. Eksperimenti su provedeni u laboratorijskom okruženju koristeći Optitrack sustav za povratnu vezu letjelice po poziciji. Cilj je također umetnuti šipku u cijev, koja je u ovom slučaju imala polumjer $r = 3.5cm$.

Nakon toga je fokus na planiranju trajektorije s ciljem dodirivanja zida te održavanja referentne kontaktne sile. Ovdje je zadržana kaskadna upravljačka struktura za kuteve i poziciju, na koju je dodan adaptivni impedancijski regulator za upravljanje silom. Time su proširene početna i krajnja konfguracija, koje trebaju sadržavati informacije o željenoj sili koju sustav treba postići. Princip postizanja neke kontaktne sile na zid je zadavanje reference pozicije "u zid", što je izlaz adaptivnog impedancijskog regulatora. S obzirom da je zid nepomičan, konačna pozicija letjelice ostaje nepromijenjena uz postizanje željene kontaktne sile. Obzirom da i letjelica i manipulator mogu biti odgovorni za postizanje sile, uvedena je distribucija impedancije na oba podsustava. Odziv kontaktne sile je potom analiziran za različite omjere impedancije

letjelice i manipulatora u simulacijskom okruženju, te je odabran najbolji distribucijski faktor. Simulacije su zatim izvedene na ravnom i kosom zidu.

U šestom poglavlju je metoda iz prethodnog poglavlja proširena na heterogene višerobotske sustave. Pri tome svaki sustav zasebno korigira konfiguraciju vrha alata kroz vlastiti nul prostor. Metoda planiranja putanje i trajektorije proširena je na visokodimenzionalni konfiguracijski prostor višerobotskog sustava, pri čemu su svi stupnjevi slobode uključeni u planiranje gibanja. Korištena su tri sustava s istim tijelom letjelice pri čemu su ugrađeni različiti manipulatori: manipulator s pet stupnjeva slobode, s četiri stupnja slobode, te s tri stupnja slobode. S obzirom da je zamišljeni scenarij prenošenje tereta pomoću višerobotskog sustava, prvi korak metode je planiranje putanje prenošenog tereta. Iz homogene transformacije tereta u inercijalnom koordinatnom sustavu određuju se prihvatne transformacije za svaki alat. Potom se primjenjuje heuristička određivanja optimalne konfiguracije vrha alata na temelju konfiguracije prihvatne točke, što je potrebno kako bi se odredila pozicija i orijentacija letjelice u svakoj točki putanje. Potom se planira visokodimenzionalna putanja i trajektorija u konfiguracijskom prostoru višerobotskog sustava, uz korekciju vrha alata svakog robota.

Kako bi se opisana metoda ispitala u simulacijskom okruženju, korištene su kombinacije koje se sastoje dva zračna manipulatora. Imajući na umu scenarij prenošenja tereta, metrika za evaluaciju je odstupanje relativne transformacije između dva vrha alata u izvršenoj trajektoriji u odnosu na planiranu trajektoriju. Pritom se zasebno gleda pozicijsko i orijentacijsko odstupanje uz napomenu da oba odstupanja moraju biti relativno malena kako bi teret u konačnici bio uspješno prenesen. Metoda je analizirana na četiri tipa trajektorije: ravna trajektorija gdje se sustav giba naprijed i nazad; trajektorija kvadrata bez mijenjanja kuta zakreta oko z osi; kružna trajektorija uz mijenjanje orijentacije na način da predmet uvijek gleda prema centru kružnice; te trajektorija planirana u simuliranom skladištu. Dobiveni rezultati uspoređeni su zasebno za odstupanje izvedene trajektorije vrha alata od planirane, sa i bez primjene korekcija po dinamičkom modelu.

U sedmom poglavlju je predstavljena metoda planiranja gibanja za izbacivanje tereta, imajući u vidu scenarij gašenja požara gdje letjelica izbacuje sredstvo za gašenje na vatru. Nakon izbacivanja, na teret djeluju sila gravitacije i otpor zraka. S obzirom da se radi o malenim brzinama projektila, pokazano je da se otpor zraka može zanemariti, što u konačnici rezultira paraboličnim izgledom trajektorije. Time se jednadžbe parabole u trodimenzionalnom prostoru koriste kako bi se na temelju mete odredio skup prihvatljivih konfiguracija izbacivanja koje dovode teret do te mete. Planiranje trajektorije se odvija u tri koraka. Prvi je planiranje trajektorije pomoću TOPP-RA algoritma uz pretpostavku stajanja u početnoj i krajnjoj točki. Nakon toga se pomoću polinoma petog stupnja planira trajektorija za izbacivanje tereta. Peti stupanj polinoma odabran je zato što ima šest slobodnih koeficijenata, a to je dovoljno za specificiranje pozicije, brzine i akceleracije na početku i kraju polinoma. Nakon toga se planira zaustavna

trajektorija jer će letjelica imati neku brzinu prilikom izbacivanja tereta te je nužno provjeriti hoće li navedeno gibanje imati sudare s okolinom. Konačna trajektorija je spoj tri prethodno navedene trajektorije.

Navedena metoda je ispitana u simulacijskom okruženju, počevši s testovima u prostoru bez prepreka. U tom slučaju su ispitane razne konfiguracije točke izbacivanja, prateći odstupanje tereta od zadane mete u prostoru. Nakon toga je planer testiran u relativno velikom okruženju simuliranog grada, gdje je potrebno isplanirati dugačku trajektoriju kako bi se sredstvo za gašenje požara ubacilo kroz prozor zgrade. Planer je nakon toga testiran i na primjeru uredskog prostora, što rezultira kraćom trajektorijom, ali je potrebno pružiti više pažnje izbjegavanju prepreka. Nakon ispitivanja i ugađanja parametara metode u simulaciji, slijedi eksperimentalna provjera. Da bi se izbacio teret, bilo je potrebno dizajnirati prihvat za teret. U ovom slučaju prihvat koristi elektromagnete koji su vrlo pogodni za izbacivanje jer se elektromagneti mogu elektronički paliti i gasiti. Prvi eksperimenti odrađeni su u laboratorijskom okruženju uz Optitrack sustav za praćenje pozicije letjelice. Pritom su odrađeni eksperimenti u praznom prostoru te eksperimenti uz izbjegavanje prepreke. Na kraju je metoda ispitana u vanjskom okruženju, uz Cartographer algoritam simultane lokalizacije i mapiranja kao povratna veza pozicije i brzine. Dio ispitivanja odrađen je u prostoru bez prepreka, dok je za izbjegavanje prepreke bilo potrebno izgraditi kartu prostora pomoću Cartographer algoritma.

U osmom poglavlju dan je zaključak ove disertacije.

Ova disertacija sadrži tri glavna znanstvena doprinosa:

- *Metoda planiranja gibanja vrha alata bespilotnog zračnog manipulatora temeljena na dinamičkom modelu sustava* (Poglavlje 5)
- *Metoda planiranja gibanja heterogenog višerobotskog sustava temeljena na združenom dinamičkom modelu za kooperativnu manipulaciju* (Poglavlje 6)
- *Metoda planiranja gibanja bespilotnog zračnog manipulatora s dinamičkim ograničenjima u točki izbačaja za balističku trajektoriju terete* (Poglavlje 7)

Ključne riječi: bespilotna zračna letjelica, zračna manipulacija, planiranje gibanja, višerobotski sustavi

Contents

CHAPTER 1

Introduction

Unmanned Aerial Vehicles (UAVs) are aircraft systems without a human pilot aboard. The classification of these vehicles by the design type can be divided in three categories: fixed-wing aircraft, single rotor blade aircraft like helicopters and monocopters, and multirotors, as introduced in [1]. Representative examples for each UAV type are shown on Fig. 1.1. Fixed-wing aircraft are composed of a fuselage and wings, with propellers producing thrust in the direction of motion. This makes fixed-wings a very efficient cruising aircraft capable of covering large distances. On the other hand, they require some kind of specialized infrastructure to get airborne, such as a launch ramp or runway. The second category, helicopters, produce thrust with a single large blade giving them the ability of Vertical Take-Off and Landing (VTOL), requiring much less space to get airborne. The rotor blade can change the angle of attack and thrust direction, which makes the helicopter mechanical design relatively complex and imposes a high maintenance cost. Multirotors are composed of multiple propulsion units rigidly attached to a frame at some distance from the center. Each rotor produces thrust independently, and the multirotor is controlled by varying the rotor angular velocity. This makes their mechanical design relatively simple, with little maintenance required to keep it operational, at the expense of limited payload capabilities.

(a) Skywalker X8 fixed wing UAV. (b) Vapor 55 helicopter. (c) DJI Mavic 3 quadrotor.

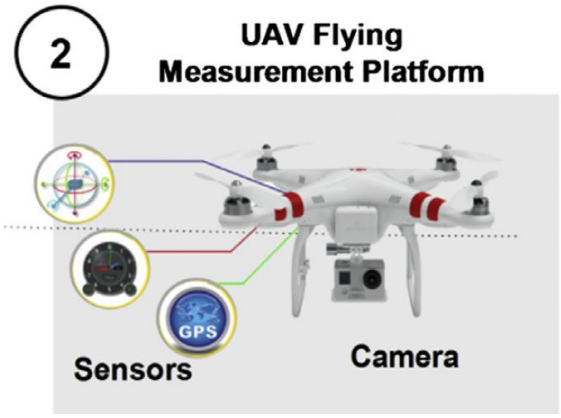
Figure 1.1: Representative examples of three main categories of unmanned aerial vehicles.

The number of multirotors is steadily growing. Their simple design, reliability and ease of

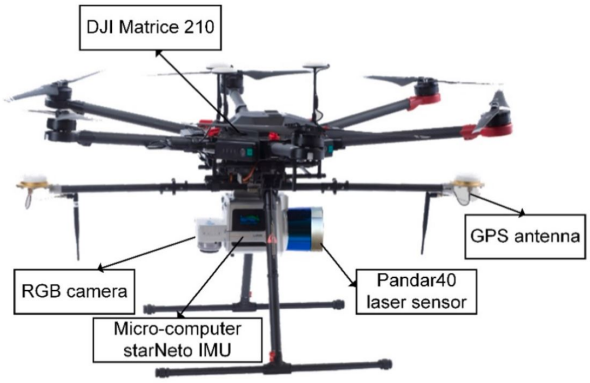
use makes them attractive and available to researchers, business owners, enthusiasts, and general public. Nowadays, it is common to find an off-the-shelf ready-to-fly multirotor in shops, making them accessible to virtually anybody. Most commonly found vehicles are quadcopter with four propulsion units, however, hexacopters and octocpters with six and eight propulsion units are also among frequently found configurations. Depending on the sensors, brand and design quality, the price range of these vehicles is rather wide, pushing the availability argument even further. The ease of use for multirotors stems from their hovering ability and intuitive control inputs, controlling the vehicle by moving sticks on a radio transmitter unit. There are three predominantly used flight modes: *stabilize* mode, where the pilot controls all vehicle angles and thrust; *altitude hold*, where the vehicle holds a certain height, alleviating some stress from the pilot; and *loiter* mode that maintains the specified location and heading of the vehicle. The latter is the first step towards the high level mission planning, which is today a standard for most popular off-the-shelf platforms.

All the aforementioned research and development makes multirotors readily available and accessible. The versatility of these vehicles has sparked their usage in a wide range of applications. Most commonly, multirotors are equipped with a camera that allows taking pictures and videos of the environment. In most cases, a camera equipped multirotor is used for recreational or hobby purposes, with professionals aiming to record various events such as weddings, concerts, etc. Furthermore, these vehicles are less frequently used for more complex professional applications, since they often require an experienced user with specific expertise about such applications. Several examples of such professional applications are shown on Fig. 1.2. For instance, these include photogrammetric surveys [2], terrain mapping [3], archaeological surveys [4], infrastructure inspection such as bridges [5, 6], wind turbines [7], oil and gas pipelines [8], and railroad inspection and monitoring [9]. Apart from the camera, a Light Detection And Ranging (LiDAR) sensor is also frequently used. The LiDAR works by emitting beams of infrared light to obtain distance measurements, where a 3D version of the sensor usually implies scanning the full 360° degrees around the spinning axis, with a limited vertical scan angle. Some typical use cases are terrain mapping [10], powerline inspection [11], and bridge inspection [12]. Some multirotors can even have a limited interaction with the environment. For example, precision agriculture requires spraying crops, where the goal is to lower the amount of pesticides by applying it to the most affected areas. The well known multirotor manufacturer *DJI* has recently introduced the *DJI Agras* platform, that features tanks for spraying crops, and it can even spread seeds. Another application of a similar platform can be found in firefighting scenarios, where the multirotor is tasked to extinguish a fire, as envisioned in [13, 14]. All the aforementioned applications and the appearance of cheap personal multirotors have vastly expanded customer reach while keeping the prices in an affordable range.

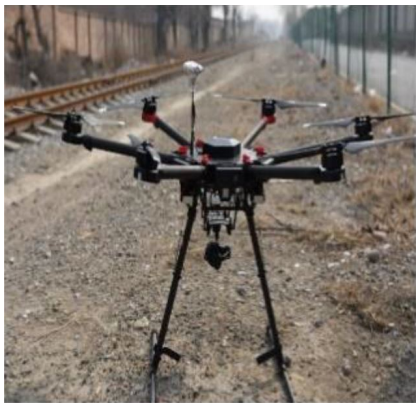
As the scientific community pushes the boundaries of multirotor UAVs, they are becoming



(a) Aerial archaeology multirotor. Used with permission of Elsevier Science & Technology Journals, from [4]; permission conveyed through Copyright Clearance Center, Inc.



(b) Power line inspection multirotor. Used with permission of Elsevier Science & Technology Journals, from [11]; permission conveyed through Copyright Clearance Center, Inc.



(c) Railroad inspection multirotor. Copyright [9] CC BY 4.0.

(d) Wind turbine inspection multirotor. Copyright [7] CC BY 4.0.

Figure 1.2: Examples of multirotors employed for various applications.

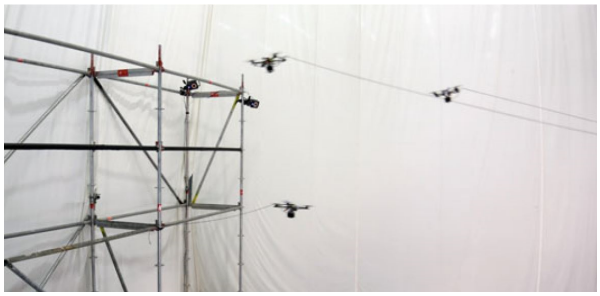
more and more present in the everyday life. Researchers are still deriving novel control laws, trying to make multirotors more agile, aggressive and reactive to unknown disturbances. Attaching sensors such as Red-Green-Blue-Depth (RGB-D) cameras, 2D and 3D LiDARs, Inertial Measurement Units (IMU), and others, together with the increase in computing power, allows these vehicles to run localization and mapping algorithms. Having a map of the environment further allows for motion planning procedures to search for paths and trajectories that avoid collision with the environment. Although such algorithms are computationally expensive, it gives the multirotor the ability to safely navigate the environments, opening doors to new applications. To stimulate new research and real world application scenarios, more and more robotic competitions including multirotors emerge, i.e. [15, 16, 17, 18, 19]. The emphasis in these challenges is on exploring an unknown environment to perform a predefined mission goal, such as providing locations of trapped or injured personnel, identifying and catching an intruder UAV, disaster relief, and similar situations. Still, the aforementioned research and challenge scenarios are mostly relying on gathering data about the environment, and in some cases delivering

mission specific goods, rather than interacting and changing the environment.

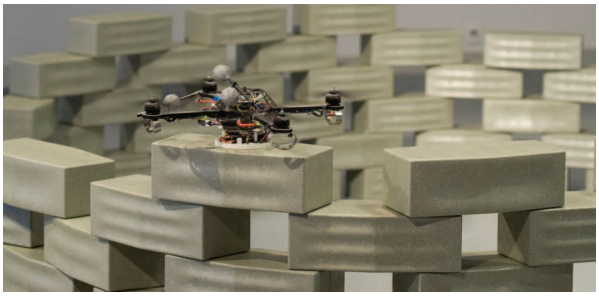
A sort of a paradigm shift in the scientific community happened when researchers started augmenting multirotors with robotic manipulators. Rather than just observe and gather data (passive tasks), the newly conceived aerial manipulators have the ability to interact and change the environment (active tasks). This unleashed the potential for countless research directions and applications. Naturally, researchers had to devote special attention to constructing manipulators fit for relatively small multirotors. Due to the limited multirotor payload capabilities, it is required that such manipulators are lightweight. Furthermore, from the control perspective, the manipulator introduces some disturbance which has to be taken into account when designing controllers to maximize the stability region. Therefore, lightweight materials and small servo motors are usually used as construction blocks for aerial manipulators. According to [20], there are two main types of aerial manipulators: Flying Hands (FH) and Unmanned Aerial Manipulators (UAM). As far as aerial manipulators go, flying hands are relatively simple in the mechanical design since the operation mode is grasping an object. Although the manipulation workspace is limited for such vehicles, and subject to the multirotor dynamical abilities, a lot of tasks can be accomplished with flying hands. Pick and place tasks are a good example of the effectiveness of the relatively simple mechanical design. As already stated, flying hands can have a gripper attached to their body. Some examples include grippers actuated with servo motors and magnetic grippers. Transporting a load with cables or tether mechanisms is also considered as flying hands. Unmanned aerial manipulators provide a complete kinematic chain, allowing them to accurately track the end-effector configuration even in case of unknown disturbances.

To perform even simple tasks, flying hands are first required to achieve and maintain a stable contact with the environment [21]. Although such a requirement seems straightforward at a first glance, it is essential to address stability conditions before, during, and after the contact. This opens the way towards more complex tasks, yet still simple in the eye of the human observer. Such tasks include pushing [22], door opening [23], and aerial writing [24]. Furthermore, flying hands can be grouped in teams to achieve even more impressive feats. Collaborative tasks require precise coordination and teamwork to achieve the desired goal. In [25] a team of flying hands cooperates to build a rope bridge. The final structure can withstand a human crossing it, which unlocks the potential to apply such technology in real world to cross rivers or crevasses. Another example of a successful cooperation is constructing a complex brick structure with flying hands [26].

The more elaborate mechanical design of unmanned aerial manipulators, coupled with multiple degrees of freedom, increases the dexterity of these vehicles allowing them to perform more complex manipulation tasks. One of the greatest advantages of unmanned aerial manipulators is maintaining the desired end-effector configuration regardless of various disturbances present during the flight. For example, to compensate wind gusts, the multirotor body must tilt



(a) Flying hands cooperatively building a rope bridge. Used with permission of Springer, from [25]; permission conveyed through Copyright Clearance Center, Inc.



(b) Flying hand performing a construction task by building a complex structure. ©2014 IEEE. Reprinted with permission from [26].

Figure 1.3: Examples of flying hands building a rope bridge and performing construction tasks.

to counteract the wind force. The attached manipulator can adjust its joint positions to keep the end-effector at the desired configuration, not compromising the manipulation task. There are three representative tasks that researchers started with: peg-in-hole insertion [27], pick-and-place [28], and valve turning [29]. These tasks might seem effortless to human audience, which is somewhat expected due to our fine tuned motor skills, supreme vision and touch sensing abilities. Indeed, humans have a very intuitive force sensing that manifests through touch. Throughout our whole lives we learn how to interpret these inputs which results in very precise movements. The scientific community also realized the importance of force and torque sensing, which is frequently applied to aerial manipulators.

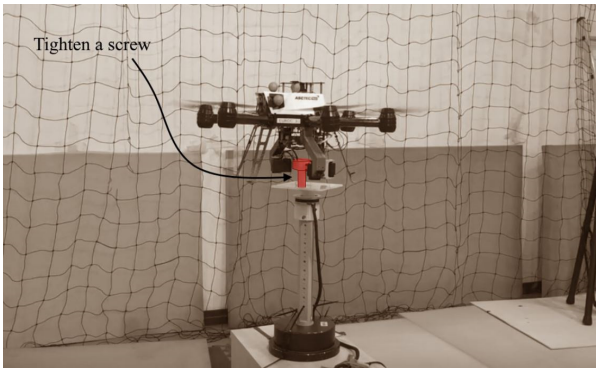
Although this field is still present only in the research community, the envisioned and performed tasks are getting ever more close to commercial products. Current application scenarios are mostly oriented towards infrastructure inspection and maintenance. Such applications include oil and gas industrial inspections [30], contact based inspection for curved tanks [31], bridge inspection and sensor mounting [32, 33], wind turbine inspection and repair [34], and tightening a screw [35]. The experimental analysis performed by researchers is currently validated in both laboratory and relevant environments, which indicates a strong tendency for commercialization in the following years.

1.1 Contributions

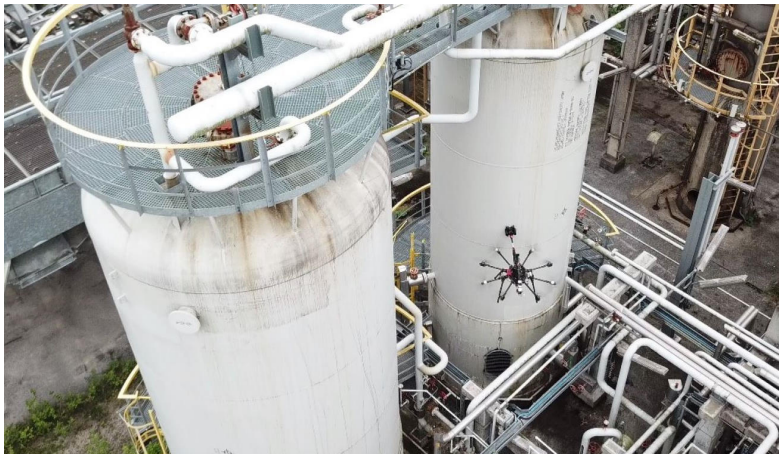
There are three main contributions of this thesis.

- *Unmanned aerial manipulator end-effector motion planning method based on the dynamical model of the system* (Chapter 5)
- *A heterogeneous multi-robot system motion planning method based on the coupled dynamical model for cooperative manipulation* (Chapter 6)
- *Unmanned aerial manipulator motion planning method with dynamical constraints at the release point for precision ballistic airdrop* (Chapter 7)

(a) Valve turning experiment. ©2014 IEEE. Reprinted with permission from [29].



(b) Tightening a bolt. ©2018 IEEE. Reprinted with permission from [35].



(c) Gas and oil plant inspection. Copyright [30] CC BY 4.0.

(d) Bridge inspection. ©2017 IEEE. Reprinted with permission from [32].

Figure 1.4: Unmanned aerial manipulators performing experiments in laboratory and relevant environments.

1.2 Thesis Outline

This thesis is organized in seven chapters, as follows:

Chapter 2: The problem description is given within this chapter, as well as the state-of-the-art overview and how do contributions proposed by this thesis expand the scientific body of knowledge.

Chapter 3: Within this chapter the Newton-Euler mathematical model of the aerial manipulator is overviewed. This is followed by description of the used control laws.

Chapter 4: This chapter introduces the basic building blocks for the motion planning. Some planning prerequisites are given together with the path and trajectory planning for high dimensional aerial manipulator configurations.

Chapter 5: The first contribution of this thesis is introduced within this chapter. It revolves around the motion planner for aerial manipulators based on the dynamical model of the system. The planner is extensively tested in a realistic simulation and laboratory environment. Furthermore, the planner is augmented with a force reference to perform a controlled wall contact in the simulation environment.

Chapter 6: The second contribution is covered within this chapter. Similar to the first contribution, a motion planner is introduced for a heterogeneous multi-robot system, with the dynamical model embedded in the planning procedure. A simulation analysis is performed on various scenarios to validate the proposed motion planning method.

Chapter 7: The final chapter proposes the third contribution of this thesis. The motion planner is proposed for delivering a payload to a specified location in the environment. The planner is then extensively tested in the simulation, indoor laboratory environment, and outdoor environment.

Chapter 8: Finally, the thesis is finished with a conclusion of the presented work.

CHAPTER 2

Problem Description and Related Work

As the title of this thesis suggests, the problem tackled within this work is planning a feasible motion while respecting spatial and dynamical constraints, and including the dynamical model in the planning procedure. To explain in more detail, one can imagine a multirotor vehicle with a manipulator attached to its body. The goal is tracking the specified end-effector configuration in the world coordinate system, which is planned in accordance with the mission requirements. The multirotors employed in this thesis are considered to produce thrust of each rotor along the multirotor body z axis. The direct consequence of such a configuration is the underactuated nature of such vehicles. Namely, the vehicle is required to tilt for producing a horizontal motion, which compromises the initial concept of specifying the end-effector configuration as the roll and pitch angles are unknown at that stage. One way to account for the underactuated nature of the multirotors is to include the dynamical model of the aerial manipulator in the motion planning procedure. Although this inevitably complicates the motion planning, the end-effector configuration can be corrected by obtaining the previously unknown roll and pitch angles through the dynamical model. Simulating the aerial manipulator dynamics within the motion planning procedure yields the full vehicle state, which is required to correct the end-effector configuration through the manipulator inverse kinematics. This principle can be also applied online, while the trajectory is being executed, however, there is no guarantee the motion will be feasible.

The second major problem encountered in this work is planning dynamically challenging maneuvers in cluttered environments. The third contribution proposes a motion planner for delivering a payload to a specified location. When the payload is released, only the gravitational

force acts on it, which gives it the characteristic parabolic trajectory shape. The dynamical conditions at the release point are specified in terms of velocity and acceleration. To achieve the release configuration, the multirotor must execute rather agile and aggressive maneuver, while taking care not to collide with the environment. The motion after the release instance also needs to be taken into account to avoid collisions and safely return to the hovering conditions.

Multirotor Control

The first consideration for both model based and agile motion planning is the aerial manipulator control. In most cases, the multirotor control problem is separated in the attitude and position control. The attitude controller points the multirotor in the desired direction, controlling the angular velocities and angles themselves. The position controller generates the input for the attitude controller, while reaching the desired position and velocity references. Tuning such a cascade controller structure is by no means trivial. With the help of the singular perturbation theory, sometimes referred to as the time-scale separation principle, it is possible to observe each controller in the cascade separately [36]. The inner (attitude) loop is considered to have higher bandwidth and these dynamics can be ignored by the outer (position) loop. Over the past two decades, researchers proposed numerous control structures. In [37], the attitude of a tilt rotor hexacopter is controlled through a geometric controller, exploiting the configuration to achieve a fully actuated system. The researchers in [38] compare a linear with optimal control methods for attitude, with experimental analysis. In [39], the attitude is controlled through a Proportional-Integral-Derivative (PID) controller, which is augmented with a bi-linear quadrotor model. The work presented in [40] proposes a sliding mode controller for multirotors, while [41] introduces an adaptive controller. Former methods are concentrating on the attitude controller, while the linear Model Predictive Control (MPC) for multirotor position is researched in [42]. The controller is augmented with obstacle avoidance in the cost function, and the trajectory tracking is validated through an experimental setup. A non-linear MPC (NMPC) is introduced in [43], where researchers compare it to a linear MPC with an extensive experimental analysis. A backstepping position controller is employed in [44] where a stability proof based on Lyapunov theory is provided. A relatively simple, yet highly efficient and effective, PID position controller is used by researchers in [45]. The common denominator of the previously described methods is separating the low level attitude control from the high level position control. In [46], researchers propose a combined position and attitude NMPC and test it on a hexacopter with failed rotors. In [47], a geometric controller capable of tracking both attitude and position reference is developed. Naturally, much more research has been conducted for the multirotor control problems, and an interested reader is referred to a detailed multirotor control survey [48].

Aerial Manipulator Control

When it comes to controlling the aerial manipulators, flying hands are mostly covered in the former paragraph. The payload a flying hand is transporting adds an extra mass to the system and changes its overall center of mass. The work in [49] observes systems with varying center of mass as hybrid systems. The results can also be applied to flying hands rigidly manipulating some object. If the system is stable both with and without transporting the object, than the overall hybrid system is stable as long as there is enough time between switches to allow the system to stabilize. This is indeed the case with flying hands observed within this thesis. This does not extend to a slung load transportation which requires additional considerations on stability and control [50, 51].

As unmanned aerial manipulators have a moving manipulator that continuously changes the system center of mass, the control for such vehicles is studied in more detail. There are two dominant approaches: deriving a controller for the full aerial manipulator, taking into account the full kinematic and dynamic model in the process; and observing the multirotor and the manipulator as two independent systems that are considered to be external disturbances on each other. The work presented in [52] proposes a backstepping controller for an aerial manipulator with 7 Degree-of-Freedom (DoF) arm. It identifies the three main effects when the attached manipulator moves: shift in the system's center of mass; changes in the moment of inertia; and constantly changing forces and torques produced on the multirotor body. Researchers in [53] introduce an adaptive sliding mode controller for a multirotor with two DoF arm attached. In [54] researchers exploit the manipulator dynamics directly in the controller structure, improving the end-effector tracking performance. This is the first step towards attaching heavier manipulators to the multirotor, as the manipulator movement can be used to produce moments on the multirotor body. In our previous work [55, 56], we employed a cartesian manipulator on a large scale multirotor with slow rotor dynamics. Roll and pitch angles are controlled through both the manipulator and slow dynamics propulsion units, improving the angle tracking performance. Furthermore, the system is tested with a position controller in an experimental setup.

In the latter case, the multirotor and attached manipulator are controlled independently. The manipulator control has been studied extensively in the past [57]. Nowadays, the lightweight manipulators for aerial manipulation are mostly constructed with servo motors with angular or torque inputs. The control of such motors is resolved internally, and therefore, it is not in focus of this work. Researchers in [58] minimize the manipulator effect on the multirotor by proposing a moving battery that counteracts the statics of the robotic arm. The work in [59] proposes a variable parameter integral backstepping controller, which is compared to a standard PID controller. An experimental analysis is performed with a multi link aerial manipulator. Researchers in [60] use a redundant 7-DoF robotic arm attached to a helicopter, with a controller design counteracting the arm center of mass horizontal movement. Aerial manipulators have

been employed in our previous work, observing them as a disturbance on the multirotor body. In [61, 62], the focus is on delivering packages with a heterogeneous aerial-ground team. The aerial manipulator is tasked to pick up a package and place it on the ground robot. Furthermore, in [63] a multi robot cooperative valve turning is performed, with two aerial robots serving as the operator visual feedback, while the employed aerial manipulator turns the valve.

Aerial Manipulator Impedance Control

The aerial manipulator control examples mentioned so far introduced some interaction with the environment, mainly in manipulating objects, as well as pick and place tasks. Some applications require achieving a contact with the environment and maintaining it, while controlling the contact force. Having the ability to exert some force on the environment increases the number of potential applications. However, it also complicates the mechanical design due to incorporating a force sensor on the manipulator, as well as the controller structure that takes the force feedback into account. The contact can be achieved without force measurements, by setting the position reference of the end-effector beyond the obstacle and relying on the multirotor body to push against the wall. Researchers in [64] use a quadrotor with a delta manipulator to achieve a wall contact. They perform an experimental analysis without including the force estimate in the controller structure. In [65] a Model Predictive Control (MPC) strategy is used to achieve and maintain a wall contact. This allowed for performing aerial writing and an uneven pipe inspection experiments in a laboratory environment.

Attaching a force sensor on the manipulator allows for a more precise force tracking and control. In [66] researchers propose an impedance filter with a trajectory generator to achieve compliant behavior, which is experimentally demonstrated by pulling a rope and semi-flexible bar. In [67], a fully actuated aerial manipulator is employed to compare the controller design with and without the force feedback. The work presented in [68] controls the force through an impedance filter to achieve a desired force reference. The approach is experimentally validated by pressing an emergency button, and maintaining the force reference afterwards. In [69] a lightweight aerial manipulator employs a force controller to achieve compliance, which is demonstrated on an indoor and outdoor sensor installation experiments. In our previous work, an adaptive impedance controller with force feedback is employed on a dual arm manipulator to tighten a bolt in both simulation [70], and real world experiments [35]. Furthermore, a similar approach is employed on a single arm manipulator for attaching sensors for performing bridge inspections [33].

Path Planning

Designing the control system for multirotors and unmanned aerial manipulators is the first step towards the motion planning of such vehicles. The problem of motion planning can be

divided in two separate steps: path, and trajectory planning. A path can be defined as a set of waypoints in the environment. The objective of a path planner is to find an obstacle free path based on the environment map. Typical environment decomposition consists of voxels, a fixed size cubes that are either free or occupied. One such environment representation is called OctoMap [71], which is a hierarchical tree structure that stores occupancy probabilities. Each voxel of the OctoMap can be divided in eight equal parts, which increases the efficiency of the OctoMap representation. In recent years, a new type of environment representation based on euclidean signed distance fields has become more and more used for unmanned aerial vehicles [72].

According to [73], path planning problems for multirotors can be divided in sampling based techniques and artificial intelligence techniques. In this work, sampling based techniques are considered and employed in the motion planning procedure, as they perform well on high dimensional robot configuration spaces. One such approach is based on cell decomposition, where the configuration space is divided into free and occupied cells where it is computationally inexpensive to find an obstacle-free path. An example is provided in [74], where researchers modify the original idea by transforming the configuration space into a graph, and the search for a path is then performed on the graph. Another approach is representing the environment as potential fields, where a path can be found by following the minimum field values [75]. One subgroup of the sampling based techniques with most attention over the past two decades is the roadmap method. Such techniques perform two steps until a feasible path is found: construction step, where the connection between nodes is computed and ensured to be obstacle free; and query step, where the concatenated connections are checked for start and goal states. Probabilistic Roadmaps (PRM) consists of randomly sampled nodes connected with obstacle-free straight lines [76]. A different method, dubbed A*, operates by decomposing the environment into a grid and guarantees optimal solution on that grid [77]. However, the most popular method for path planning is Rapidly-exploring Random Trees (RRT). The method constructs a tree beginning from the start node and searches for a connection towards the goal node. The advantage of such a planner is that it does not need to precompute a grid or perform environment decomposition, as it is an exploration planner [78]. Throughout the years, researchers produced many versions of the RRT planner, such as RRT-Connect that can account for multiple multirotors collision avoidance [79]. One downside of the RRT planner is that it searches in all directions of the configuration space. In some cases, this can significantly increase the planning time, which is not a desired behavior. To alleviate this problem, researchers in [80] propose an Informed RRT path planner that bounds the search space within an ellipse spanned between the start and goal states. In this work, we opted for the RRT* planner, which is an extension of the RRT that optimizes the path within the allotted time, rather than returning the first obstacle-free path found. Furthermore, the RRT* planning time scales well with increasing dimension of

the configuration space, which is also a requirement as the aerial manipulator can have a lot of degrees of freedom. To find out more about path planning for multirotors, the reader is referred to a comprehensive review presented in [73].

Trajectory Planning for Multirotors

As stated earlier, the second step in the motion planning procedure is the trajectory planning. Recalling, the path is a set of waypoints in the robot configuration space, while the trajectory is obtained when the path is interpolated with a velocity profile. This is a minimal definition of the trajectory and it can be expanded to include higher order derivatives like acceleration, jerk or snap. Since flying hands do not continuously change the overall system dynamics, it is sufficient to plan a trajectory only considering the multirotor. A widely accepted trajectory generation method introduced in [81] uses a convex optimization to minimize the integral of 4^{th} position derivative, often referred to as snap. The full optimization criterion consists of both snap and time, which yields feasible and aggressive trajectories respecting the dynamical constraints of the multirotor. This approach has been frequently utilized, for instance, researchers in [82] experimentally verifies the planner by passing through narrow vertical gaps, exploiting limits of the multirotor. In [83], a similar approach is employed with a closed form solution for computing the unknown derivative conditions at the supplied waypoints. This line of research is further explored in [84], where the authors improve the numerical stability of the original approach. The work presented in [85] plans a polynomial spline trajectory to intercept a projectile. This approach allows for specifying the endpoint derivative conditions, which is demonstrated by attaching a tennis racket on the multirotor to return a thrown ball. A similar approach is presented in [86] by generating a large number of spline primitives, and choosing one to intercept the projectile with proper multirotor attitude. Relying on the differential flatness property, researchers in [87] plan spline trajectories to pass through narrow gaps by specifying the dynamic conditions at the gap point and concatenating adjacent splines. To ensure the planned trajectory is collision free, the approach from [88] models the multirotor as an ellipsoid and generates a trajectory as a sequence of motion primitives. Apart from quadratic programming approaches, planners based on sequential convex programming [89] and mixed integer programming [90] are also in the research focus. Some planners offer online replanning [91], where the first step is to plan the initial, global trajectory, followed by local replanning to avoid obstacles.

In our previous work [61, 62], we developed a planner that operated on 5^{th} order splines, with 6^{th} order splines placed at the beginning and end of the trajectory. Although this planner worked reliably on small number of waypoints, the planning time increased significantly with the number of waypoints. We experienced a similar behavior in minimum snap convex optimization planner from [81]. A family of planners based on the numerical integration can be employed to alleviate the problem of a large number of waypoints as the input for the trajectory

generation. The input path is parameterized with a scalar function, leaving only one variable to optimize. The time optimal trajectory is searched in path-velocity $(s, \dot{s})$ plane with the "bang-bang" principle, integrating with maximum and minimum accelerations. The initial approach was conceived four decades ago in [92, 93], with the main advantage of generating the optimal trajectory with low computational power demand. Over the years the planner has been refined, however, researchers struggled to address various singularities, ultimately leading to few specialized approaches with robotic manipulators. Finally, in [94] the researchers provide a general implementation of the Time Optimal Path Parameterization (TOPP) planner. This planner has been further extended to TOPP by Reachability Analysis (TOPP-RA) [95], offering a different way of planning the time optimal path. The planner has a widely accepted open source practical implementation. Furthermore, the TOPP-RA is a very reliable and fast planner, yielding high dimensional trajectories in milliseconds. In our previous work, we employed the planner for wind turbine inspections [7], civil infrastructure inspection by a team of multirotors [96], and while performing the environment exploration [97, 98]. The common denominator of these works is using the TOPP-RA planner for the multirotor only.

Trajectory Planning for Unmanned Aerial Manipulators

Trajectory planning for aerial manipulator has to account for additional degrees of freedom, provided by the manipulator itself. This naturally complicates the problem and typically increases planning time, which can become essential for online planners. To manipulate an object using an aerial manipulator, researchers in [99] impose task space constraints to demonstrate opening a drawer or carrying a bucket without tilting it. In [100], task constraints are imposed on an underactuated robot balancing an inverted pendulum. Task constraints are often employed in aerial manipulation, the difference being the floating base of the manipulator. The work presented in [101] proposes a 6D end-effector trajectory tracking in task space for an aerial manipulator, relying on the differential flatness principle. In [102], researchers decoupled the multirotor and manipulator planners, switching between different planners depending on the task at hand. They also compare planners from two widely accepted planning libraries: Open Motion Planning Library (OMPL) [103]; and MoveIt library [104]. Researchers in [105] observe a quadrotor with an arbitrary manipulator, generating dynamically feasible trajectories in task space. The system they use is differentially flat, however, the flat outputs do not correspond to the end-effector motion. A family of trajectories that produces exact task trajectory is searched and the final trajectory is chosen by minimizing energy consumption in execution. Researchers in [106] also optimize energy, as well as time, for a two arms long reach manipulator. The devised planner is based on RRT* approach, taking both manipulator and multirotor into account while planning. A planner based on Dynamic Motion Primitives (DMP) is developed within [107] and verified experimentally with two aerial manipulators cooperatively transport-

ing an object. A similar learning method employing parametric DMPs is presented in [108] and tested with two aerial manipulators cooperative transportation. In [109], researchers propose a cooperative motion planner based on potential fields and demonstrate its effectiveness by transporting a long object in a cluttered environment. In [110], a locally optimal polynomial trajectory is planned using sequential quadratic programming approach. Using this method, an aerial manipulator is tasked to pick an object from a box, carefully considering spatial constraints.

Multirotors with co-planar propulsion units are underactuated systems since they need to roll or pitch to produce lateral motion. Having a manipulator attached to such a vehicle intuitively suggests that the end-effector motion is coupled with the underactuated property of the multirotor. To follow the desired end-effector trajectory it is possible to employ inverse kinematics solver and correct the end-effector configuration through manipulator joints. Doing so online can potentially result in infeasible manipulator configurations, which ultimately compromises the end-effector trajectory tracking. One way to resolve this problem is by including the aerial manipulator dynamical model in the planning procedure, and applying corrections before executing the trajectory. In [111], the control strategy is tightly coupled with the motion planning and the aerial manipulator dynamical model is included in the planning procedure. The planner is based on the RRT algorithm, where the neighbor nodes are connected with trajectory primitives, respecting constraints imposed on all degrees of freedom of the aerial manipulator. In our previous work, a planner based on the dynamical model is proposed [112]. The unwanted motion caused by the underactuated property of the multirotor is corrected by obtaining the full aerial manipulator state through the dynamical model. This approach is further employed with an impedance force control [33], achieving and maintaining a wall contact with the aerial manipulator. In both of these works, the aforementioned TOPP-RA planner is utilized to generate trajectories for the aerial manipulator. This further validates the TOPP-RA planner as it scales well with high dimensional problems for aerial manipulators.

Precision Airdrop

The task of precision airdrop is delivering a payload to a specified target in the environment. These tasks are typically performed with fixed-wing vehicles from significant heights, often employing an active payload guidance system. Within the third contribution, the inspiration is drawn from this task, however, it is applied to flying hands. Unlike fixed-wing aircraft, multirotors work best in cluttered environments, shifting the focus on delivering a payload in obstacle-populated environments which yield much shorter payload trajectories.

A lot of work has already been done in the field of precision airdrop. Work of researchers in [113, 114] considers the package deployment using parachutes while estimating wind conditions to improve the precision. Researchers in [115] provide a similar approach, but include

the aerodynamic properties of the package in the release point calculation. The work in [116] presents the Joint Precision Airdrop System (JPAS) for precise military resupply, and [117] analyzes the optimal dispersion altitude of multiple packages and solves a traveling salesman problem with minimizing the risk for ground troops package pickup. Researchers in [118] take it a step further and propose a guidance system capable of steering a parafoil in an environment with sparse obstacles, while [119] uses a wind shear model to improve the landing precision of the parafoil.

Parafoils are a good solution in an obstacle-free environment and are not suitable for dense environments such as forests, cities, etc. In some cases, a ballistic free fall can be used to deploy packages. Researchers in [120] are carefully calculating the approach for the ballistic airdrop, while considering the wind speed. Later on, they extended their approach to an autonomous ballistic airdrop relying on the wind field and air resistance models in [121]. The work in [122] revolves around the precise ballistic airdrop based on the Global Positioning System (GPS).

The common denominator of the methods presented in above mentioned papers are i) using the fixed-wing aircraft for the delivery, and ii) the payload is usually released from a significant height. Although the energy efficiency of the fixed-wing aircraft enables flying to distant areas, they are not suitable for flying in cluttered environments. On the other hand, the agility of small-scale multirotor vehicles is perfect for parabolic airdrop applications in cluttered environments, which is the focus of the third contribution. Situations such as deploying a fire extinguishing agent through a window of a building in a city environment are investigated. In such a scenario, the system is required to execute aggressive and precise maneuvers to deliver the agent, while avoiding collision with the environment. Approaches related to this are mentioned earlier in this section.

CHAPTER 3

Mathematical Model and Control of Unmanned Aerial Manipulator

Within this chapter, a mathematical model of an unmanned aerial manipulator with an arbitrary number of degrees of freedom (DoF) is first derived. The UAV propulsion consists of an arbitrary number of rotors that are constrained in a single plane, and are pointing in the body z axis direction. Typical arrangements of such vehicles are quadcopters, hexacopters and octocopters. The manipulator with arbitrary number of DoFs is attached to the body of the UAV. In this thesis, the manipulator joints are rotational. However, in a general case, the joints can also be prismatic. All these properties are captured in the Denavit-Hartenberg (DH) parameters, which define the serial manipulator chain. Together with the UAV body state, these parameters are essential in the direct and inverse kinematics, providing the end-effector position and orientation. This is crucial when addressing the motion planning problem with end-effector correction, since the motion planner relies on both the kinematics and the dynamics of the system.

The dynamic model is observed separately for the multirotor body and the manipulator. Since there is a single rigid attachment between the multirotor and the manipulator, the cross interactions between two dynamical system are also taken into account. Namely, the manipulator acts as a known disturbance on the multirotor, with forces and torques at the attachment point. It influences the multirotor body dynamics together with the forces and torques produced from the propellers, as well as gravity. The manipulator dynamics are modelled through the well known Newton-Euler model, with the initial conditions supplied by the accelerations and angular velocities at the attachment point.

The second crucial element of the motion planning based on the dynamic model is the control strategy, since it also defines the overall motion of the system. Typically, the control

problem is divided in two parts: low level attitude control, and high level position control. Both control loops considered in this section are based on the cascade PID structure. The attitude loop controls the angular velocities and angles, achieving the desired multirotor body angle. The high level position controller is a cascade controlling velocities and positions, supplying angular references to the attitude control, and forwarding the thrust reference from the height controller.

3.1 Mathematical Model

The mathematical model is separated in two parts: the kinematic model in Section 3.1.1 and the dynamic model in Section 3.1.2. The kinematic model expresses the multirotor and the end-effector position and orientation in the desired coordinate system. The dynamic model takes the influence of forces and torques produced by propellers and joints, as well as the external influence on the system.

3.1.1 Kinematics

The coordinate systems and nomenclature convention used throughout this thesis is laid out alongside the kinematic model of the aerial manipulator. Note that all considered coordinate systems are right-handed. Fig. 3.1 depicts an aerial manipulator coordinate systems and relevant vectors. The inertial (world) coordinate system is denoted L_{W}. The gravity vector $\mathbf{g} = \begin{bmatrix} 0 & 0 & -g \end{bmatrix}^T$ is directed along the negative z_{W} axis. The moving coordinate system L_{B} is attached to the UAV body center of gravity. The relative relationship between the world and body frames is encompassed in the position and orientation:

$$\mathbf{p}_{\mathrm{W}}^{\mathrm{B}} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \mathbf{\Theta}_{\mathrm{W}}^{\mathrm{B}} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \tag{3.1}$$

where $\mathbf{p}_{\mathrm{W}}^{\mathrm{B}}$ denotes the body frame position in the world frame, and $\mathbf{\Theta}_{\mathrm{W}}^{\mathrm{B}}$ denotes the orientation. The standard Euler angles representation is used for the vehicle attitude. The chosen rotation order is the following: first, rotate around the z axis for yaw angle ψ; next, rotate around the y axis for pitch angle θ; and finally, rotate around x axis for roll angle ϕ. Combining these three

rotations yields the rotation matrix from the body to world frame:

$$\mathbf{R}_{\mathrm{W}}^{\mathrm{B}} = \mathbf{R}_z(\psi) \cdot \mathbf{R}_y(\theta) \cdot \mathbf{R}_x(\phi)$$

$$\mathbf{R}_{\mathrm{W}}^{\mathrm{B}} = \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\theta S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix}, \tag{3.2}$$

where S_α and C_α represent the sine and cosine function of the indexed angle. Cobining the rotation matrix from equation (3.2) and translation from equation (3.1), a homogeneous transform matrix can be formed:

$$\mathbf{T}_{\mathrm{W}}^{\mathrm{B}} = \begin{bmatrix} \mathbf{R}_{\mathrm{W}}^{\mathrm{B}} & \mathbf{p}_{\mathrm{W}}^{\mathrm{B}} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}. \tag{3.3}$$

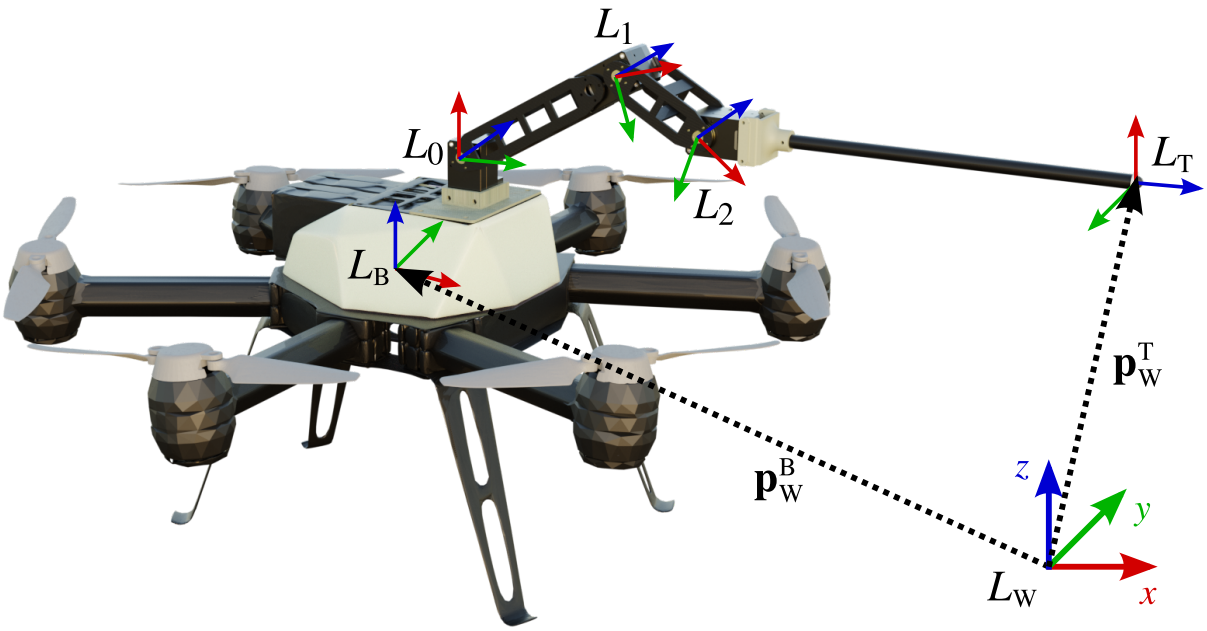


Figure 3.1: Coordinate systems of a 3-DoF aerial manipulator with position vectors of the multirotor body and end-effector.

The above notation is applicable to any two coordinate systems, and multiplying successive homogeneous transformations yields the kinematic chain of an aerial manipulator. Coordinate systems of the manipulator are explored next. The L_0 denotes the coordinate system of the manipulator attachment point to the UAV body. It is also the coordinate system of the first joint. The following joint coordinate systems are labeled as $L_1 \ldots L_n$, where n is the number manipulator DoFs. The end-effector is mounted at the manipulator end and its coordinate system is labeled L_{T}. Therefore, the manipulator consists of n joints connected with $n+1$ links. The joints' z axes are aligned with their respective operation direction. Since only revolute joints are considered in this thesis, this refers to rotation direction. In general case a manipulator can

have a prismatic joint, however, this is not considered here. The end-effector coordinate system is usually defined by three vectors: z_{T} is the approach vector, y_{T} is sliding vector, and x_{T} completes the right-handed coordinate system. Multiplying the homogeneous transforms between all frames yields the manipulator's kinematic chain, or in other words, direct kinematics:

$$\mathbf{T}_0^{\mathrm{T}} = \mathbf{T}_0^1 \cdot \mathbf{T}_1^2 \cdots \mathbf{T}_{n-1}^{\mathrm{T}}. \tag{3.4}$$

In robotics, the manipulator kinematic chain is typically described with Denavit-Hartenberg (DH) parameters [123]. This method offers relatively simple and concise way of obtaining the homogeneous transforms between successive coordinate systems of the manipulator. An interested reader can find details and applications of this method in [124, 125], and the most important information about the DH method is outlined here. Normally, when describing the transformation between two coordinate systems, one would expect to require six parameters: three translations and three rotations. In the DH method, only four parameters are required. The transformation between two successive joints can be written as:

$$\mathbf{T}_{k-1}^{k}(\theta_k, d_k, a_k, \alpha_k) = \begin{bmatrix} C_{\theta_k} & -S_{\theta_k}C_{\alpha_k} & S_{\theta_k}S_{\alpha_k} & a_k C_{\theta_k} \\ S_{\theta_k} & C_{\theta_k}C_{\alpha_k} & -C_{\theta_k}S_{\alpha_k} & a_k S_{\theta_k} \\ 0 & S_{\alpha_k} & C_{\alpha_k} & d_k \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.5}$$

where θ_k and d_k are the joint related variables, while α_k and a_k are the link related variables.

In most of manipulation cases, it is necessary to obtain the end-effector configuration in the world frame. In aerial manipulation, it is also necessary to take the manipulator attachment point on the UAV body into account, as well as the UAV location in the world frame. This defines the aerial manipulator kinematic chain:

$$\mathbf{T}_{\mathrm{W}}^{\mathrm{T}} = \mathbf{T}_{\mathrm{W}}^{\mathrm{B}} \cdot \mathbf{T}_{\mathrm{B}}^{0} \cdot \mathbf{T}_{0}^{\mathrm{T}}, \tag{3.6}$$

where $\mathbf{T}_{\mathrm{W}}^{\mathrm{B}}$ is the measured state of the UAV, $\mathbf{T}_{\mathrm{B}}^{0}$ represents a fixed transform and depends on the manipulator attachment point on the UAV body, and $\mathbf{T}_{0}^{\mathrm{T}}$ can be obtained through the DH method.

Obtaining the aerial manipulator configuration $\mathbf{q}$ from the specified end-effector transform $\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$ is usually referred to as inverse kinematics. It is a very well studied problem within the robotics community. If only the attached manipulator configuration is subject to the inverse kinematics problem, an analytical approach is widely used. The inverse kinematics equations can be derived for each robot separately, however, this leads to a separate function later in the implementation. In [126], researchers developed the *ikfast* algorithm that generates an analytical solution based on the manipulator configuration This approach often gives multiple solu-

tions, with some of them discarded due to violating the manipulator constraints or divisions by zero, and ultimately finds the closest one to the current manipulator configuration. Extending this approach to aerial manipulators is not a trivial task. A solution based on the manipulator configuration decomposition and workspace analysis is explained in detail in Section 6.2.2. The underlying principle of this solution is to find an optimal manipulator configuration based on the required end-effector orientation. The multirotor is observed as a floating base which can then reach the required position calculated through the obtained manipulator configuration.

3.1.2 Dynamics

The dynamical model introduces the notion about how different parts of the system affect each other. Namely, forces and torques produced at each joint and link are required to control the end-effector motion properly. In this thesis, the aerial manipulator mathematical model is derived in two stages. First, the manipulator model is presented using the well known Newton-Euler approach. This is followed by the multirotor mathematical model. In both stages, the dynamical effects between two parts of the system are taken into account.

3.1.2.1 Manipulator Model

When modelling the manipulator dynamics, two methods are predominantly used: Lagrange-Euler where the motion is described based on the energy, and Newton-Euler which describes the motion in terms of forces and torques. Therefore, the Newton-Euler accounts for dynamical effects experienced by each link and joint of the manipulator. Furthermore, the motion, forces, and torques of each link and joint are derived successively, which ultimately allows for a relatively simple implementation. As stated before, the manipulator consists only of rotational joints with n_{M} degrees of freedom.

There are two parts in the Newton-Euler method: forward equations where velocities and accelerations are derived, and backward equations where forces and torques are considered. In both cases, vectors are represented in the base coordinate system L_0. The forward dynamics is presented on a single link depicted on Fig. 3.2.

The angular velocity of frame i can be expressed as:

$$\boldsymbol{\omega}_0^i = \boldsymbol{\omega}_0^{i-1} + \dot{q}_i \mathbf{z}_0^{i-1}, \tag{3.7}$$

where $\boldsymbol{\omega}_0^i$ is the angular velocity of L_i relative to the base L_0, $\dot{q}_i$ is the angular velocity produced by the joint i, and $\mathbf{z}_0^{i-1}$ is the z axis of coordinate system L_{i-1} expressed in L_0 as well as the

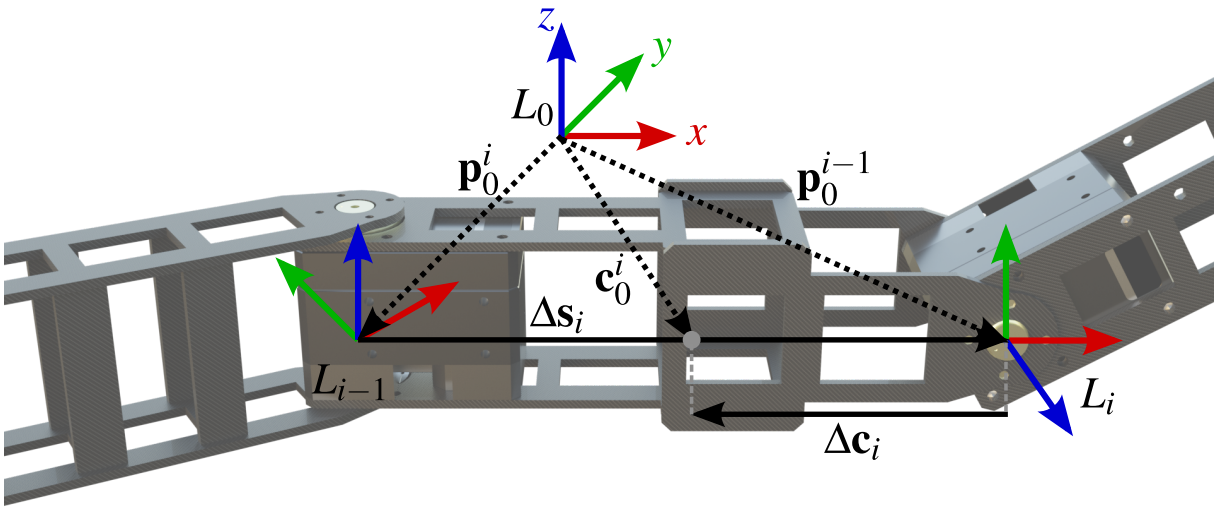


Figure 3.2: Representation of the i-th manipulator link with two successive joints. All relevant coordinate systems required by the Newton-Euler method are shown. The center of mass is placed at link's center.

rotation axis of joint i. The angular acceleration is a time derivative of the angular velocity:

$$\begin{aligned}\boldsymbol{\alpha}_0^i &= \frac{d}{dt}\left(\boldsymbol{\omega}_0^{i-1} + \dot{q}_i \mathbf{z}_0^{i-1}\right) \\ &= \boldsymbol{\alpha}_0^{i-1} + \ddot{q}_i \mathbf{z}_0^{i-1} + \dot{q}_i \frac{d}{dt}\mathbf{z}_0^{i-1} \\ &= \boldsymbol{\alpha}_0^{i-1} + \ddot{q}_i \mathbf{z}_0^{i-1} + \boldsymbol{\omega}_0^{i-1} \times \left(\dot{q}_i \mathbf{z}_0^{i-1}\right),\end{aligned} \tag{3.8}$$

where $\boldsymbol{\alpha}_0^i$ denotes the angular acceleration of L_i, and $\ddot{q}_i$ is the acceleration produced by joint i. The linear velocity has two components. One is simply the linear velocity of the previous joint, and the other is the tangential velocity produced by the link rotation. So first, the vector between two successive coordinate systems is defined:

$$\Delta\mathbf{s}_0^i = \mathbf{p}_0^i - \mathbf{p}_0^{i-1}. \tag{3.9}$$

The linear velocity can be written as:

$$\begin{aligned}\mathbf{v}_0^i &= \mathbf{v}_0^{i-1} + \frac{d}{dt}\Delta\mathbf{s}_0^i \\ &= \mathbf{v}_0^{i-1} + \boldsymbol{\omega}_0^{i-1} \times \Delta\mathbf{s}_0^i.\end{aligned} \tag{3.10}$$

The linear acceleration is a time derivative of the linear velocity:

$$\begin{aligned}\mathbf{a}_0^i &= \mathbf{a}_0^{i-1} + \frac{d}{dt}\left(\boldsymbol{\omega}_0^i \times \boldsymbol{\Delta}\mathbf{s}_0^i\right) \\ &= \mathbf{a}_0^{i-1} + \boldsymbol{\alpha}_0^i \times \boldsymbol{\Delta}\mathbf{s}_0^i + \boldsymbol{\omega}_0^i \times \left(\boldsymbol{\omega}_0^i \times \boldsymbol{\Delta}\mathbf{s}_0^i\right).\end{aligned} \tag{3.11}$$

The forward equations propagate velocities and accelerations starting in the system L_0, all the way to the end-effector. When $i = 1$, the initial conditions have to be defined. With a fixed base, the initial conditions would be null-vectors. This is not true for the floating base, such as a multirotor. As defined in equation (3.6), the manipulator is attached at an arbitrary point on the UAV body. This offset from the body center of mass produces additional members in the initial conditions:

$$\begin{aligned}\boldsymbol{\omega}_0 &= \mathbf{R}_\mathrm{B}^0 \boldsymbol{\omega}_\mathrm{B} \\ \boldsymbol{\alpha}_0 &= \mathbf{R}_\mathrm{B}^0 \boldsymbol{\alpha}_\mathrm{B} \\ \mathbf{v}_0 &= \mathbf{R}_\mathrm{B}^0 \left(\mathbf{v}_\mathrm{B} + \boldsymbol{\omega}_\mathrm{B} \times \mathbf{p}_\mathrm{B}^0\right) \\ \mathbf{a}_0 &= \mathbf{R}_\mathrm{B}^0 \left(\mathbf{a}_\mathrm{B} + \boldsymbol{\alpha}_\mathrm{B} \times \mathbf{p}_\mathrm{B}^0 + \boldsymbol{\omega}_\mathrm{B} \times \left(\boldsymbol{\omega}_\mathrm{B} \times \mathbf{p}_\mathrm{B}^0\right)\right).\end{aligned} \tag{3.12}$$

Note here that $\mathbf{R}_0^\mathrm{B}$ rotates all vectors to be expressed in the L_0 coordinate system which is the base for the Newton-Euler procedure. It is also assumed that the multirotor body related vectors $\boldsymbol{\omega}_\mathrm{B}$, $\boldsymbol{\alpha}_\mathrm{B}$, $\mathbf{v}_\mathrm{B}$, and $\mathbf{a}_\mathrm{B}$ are expressed in the body coordinate system L_B. To recap the forward dynamics, all equations are summarized here:

$$\begin{aligned}\boldsymbol{\omega}_0^i &= \boldsymbol{\omega}_0^{i-1} + \dot{q}_i \mathbf{z}_0^{i-1} \\ \boldsymbol{\alpha}_0^i &= \boldsymbol{\alpha}_0^{i-1} + \ddot{q}_i \mathbf{z}_0^{i-1} + \boldsymbol{\omega}_0^{i-1} \times \left(\dot{q}_i \mathbf{z}_0^{i-1}\right) \\ \mathbf{v}_0^i &= \mathbf{v}_0^{i-1} + \boldsymbol{\omega}_0^{i-1} \times \boldsymbol{\Delta}\mathbf{s}_0^i \\ \mathbf{a}_0^i &= \mathbf{a}_0^{i-1} + \boldsymbol{\alpha}_0^i \times \boldsymbol{\Delta}\mathbf{s}_0^i + \boldsymbol{\omega}_0^i \times \left(\boldsymbol{\omega}_0^i \times \boldsymbol{\Delta}\mathbf{s}_0^i\right).\end{aligned} \tag{3.13}$$

In the backward dynamics, the goal is to derive forces and torques on each manipulator link. As these act on the link center of mass, it is necessary to modify equation (3.13) to account for the offset between L_i and the link center of mass $\mathbf{c}_0^i$. Therefore, a vector $\boldsymbol{\Delta}\mathbf{c}_0^i = \mathbf{c}_0^i - \mathbf{p}_0^i$ is introduced as shown on Fig. 3.2. Since only rotational joints are considered, and the link is a rigid body, the angular velocity and acceleration of i-th link's center of mass is the same as L_i.

The linear velocity and acceleration need to be modified as follows:

$$\begin{aligned}
\mathbf{c}_0^i &= \boldsymbol{\Delta}\mathbf{s}_0^i + \boldsymbol{\Delta}\mathbf{c}_0^i \\
\dot{\mathbf{c}}_0^i &= \mathbf{v}_0^{i-1} + \boldsymbol{\omega}_0^{i-1} \times \left(\boldsymbol{\Delta}\mathbf{s}_0^i + \boldsymbol{\Delta}\mathbf{c}_0^i\right) \\
\ddot{\mathbf{c}}_0^i &= \mathbf{a}_0^{i-1} + \boldsymbol{\alpha}_0^i \times \left(\boldsymbol{\Delta}\mathbf{s}_0^i + \boldsymbol{\Delta}\mathbf{c}_0^i\right) + \boldsymbol{\omega}_0^i \times \left(\boldsymbol{\omega}_0^i \times \left(\boldsymbol{\Delta}\mathbf{s}_0^i + \boldsymbol{\Delta}\mathbf{c}_0^i\right)\right),
\end{aligned} \tag{3.14}$$

where $\dot{\mathbf{c}}_0^i$ represents the linear velocity, and $\ddot{\mathbf{c}}_0^i$ represents the linear acceleration of the link center of mass. The backward dynamics propagate forces and torques from the end-effector towards the base. Each link's motion is influenced by all forces and torques acting on its center of mass:

$$\begin{aligned}
\sum_{j=1}^{n} \mathbf{F}_i^j &= \frac{d}{dt} m_i \dot{\mathbf{c}}_0^i \\
\sum_{j=1}^{n} \boldsymbol{\tau}_i^j &= \frac{d}{dt} \mathbf{D}_i \boldsymbol{\omega}_0^i,
\end{aligned} \tag{3.15}$$

where $\mathbf{F}_i^j \in \mathbb{R}^{3\times 1}$ represents the sum of all forces acting on the link, $\boldsymbol{\tau}_i^j \in \mathbb{R}^{3\times 1}$ represents the sum of all torques acting on link, m_i denotes the link mass, and $\mathbf{D}_i \in \mathbb{R}^{3\times 3}$ is the inertia matrix with respect to the link center of mass. There are three forces acting acting on the link: gravity, and forces exerted by joints i and $i+1$:

$$\mathbf{f}_0^i = \mathbf{f}_0^{i+1} + m_i \ddot{\mathbf{c}}_0^i + m_i \mathbf{R}_{\mathrm{w}}^0 \mathbf{g}, \tag{3.16}$$

where $\mathbf{f}_0^i$ is the force that joint i exerts onto link i, and the same force in opposite direction is exerted on link $i-1$. The force $\mathbf{f}_0^{i+1}$ is exerted by joint $i+1$ on link i. The acceleration $\ddot{\mathbf{c}}_0^i$ is taken from equation (3.14), $\mathbf{g}$ denotes gravity vector in the inertial frame rotated by $\mathbf{R}_{\mathrm{w}}^0$ to express it in L_0, and m_i is the i-th link mass.

Apart from the force, the torque on each link also needs to be addressed. Similar as in the force equation (3.16), torques acting on the link's center of mass come from joint torques and angular acceleration. However, since torque is also produced by a force acting at some distance, force components from equation (3.16) are also present. The following can be written:

$$\begin{aligned}
\mathbf{D}_0^{c_i} \boldsymbol{\alpha}_0^i + \boldsymbol{\omega}_0^i \times \left(\mathbf{D}_0^{c_i} \boldsymbol{\omega}_0^i\right) &= \boldsymbol{\tau}_0^i - \boldsymbol{\tau}_0^{i+1} + \left(-\boldsymbol{\Delta}\mathbf{s}_0^i - \boldsymbol{\Delta}\mathbf{c}_0^i\right) \times \mathbf{f}_0^i - \boldsymbol{\Delta}\mathbf{c}_0^i \times \mathbf{f}_0^{i+1} \\
\boldsymbol{\tau}_0^i &= \boldsymbol{\tau}_0^{i+1} + \left(\boldsymbol{\Delta}\mathbf{s}_0^i + \boldsymbol{\Delta}\mathbf{c}_0^i\right) \times \mathbf{f}_0^i + \boldsymbol{\Delta}\mathbf{c}_0^i \times \mathbf{f}_0^{i+1} + \mathbf{D}_0^{c_i} \boldsymbol{\alpha}_0^i + \boldsymbol{\omega}_0^i \times \left(\mathbf{D}_0^{c_i} \boldsymbol{\omega}_0^i\right).
\end{aligned} \tag{3.17}$$

Here, $\boldsymbol{\tau}_0^i$ and $\boldsymbol{\tau}_0^{i+1}$ denote torques produced by their respective joints. Forces $\mathbf{f}_0^i$ and $\mathbf{f}_0^{i+1}$ also

act in coordinate systems of their respective joints. Therefore, the offset between the link's center of mass and the joint coordinate systems is producing an additional torque on the link. The inertia matrix $\mathbf{D}_0^{c_i} = \mathbf{R}_0^{c_i}\mathbf{D}_i\left(\mathbf{R}_0^{c_i}\right)^T$ is expressed in the L_0 coordinate system, while $\mathbf{D}_i$ is the inertia matrix expressed in the link's center of mass.

In the forward dynamics, the initial conditions produced by the UAV had to be taken into account. Here, the initial conditions are the external force and torque applied to the end-effector. For example, these can be produced by picking up and carrying an object, by contact with the environment, or can simply be zero. The external influences can be written as:

$$\left\{\mathbf{f}_0^{i+1} = \mathbf{f}_{ext}, \boldsymbol{\tau}_0^{i+1} = \boldsymbol{\tau}_{ext} \mid i = n_{\mathrm{M}}\right\}. \tag{3.18}$$

Finally, it is also useful to express the manipulator center of mass with respect to the L_0 coordinate system. This is important information in the control structure, as the manipulator displaces the overall system center of mass when attached to a multirotor. Each link is assumed to have a different mass denoted with m_i. The manipulator center of mass is simply a weighted sum of all link contributions:

$$\mathbf{p}_0^{\mathrm{M}} = \frac{\sum_{i=1}^{n_{\mathrm{M}}} m_i \mathbf{c}_\mathbf{0}^\mathbf{i}}{m_{\mathrm{M}}}, \tag{3.19}$$

where m_{M} is the total manipulator mass.

3.1.2.2 Multirotor UAV Model

As already mentioned in the previous section, the floating base for the modelled manipulator is a multirotor UAV. This kind of vehicle allows the end-effector to move freely in space, reaching far more configurations than with a fixed base. A typical multirotor UAV is composed of a central body with multiple rotors placed at the end of each arm. In this thesis, an arbitrary number of rotors n_r is considered. However, rotors are constrained to produce thrust only along the body z_{B} axis, and are placed in a single plane. Generally speaking, multirotor vehicles can be fully actuated [127], however, due to the aforementioned constraints, this is not the case in this thesis. The observed multirotors are indeed underactuated due to the thrust being produced only along a single axis. This property is important while addressing the control problem, which is described within Section 3.2.

As shown on Fig. 3.3, the multirotor center of gravity coordinate system is denoted as L_{B}. There are n_r rotors placed in the same plane with $\mathbf{p}_{\mathrm{B}}^{r_i}$ offset from the base. Each rotor produces

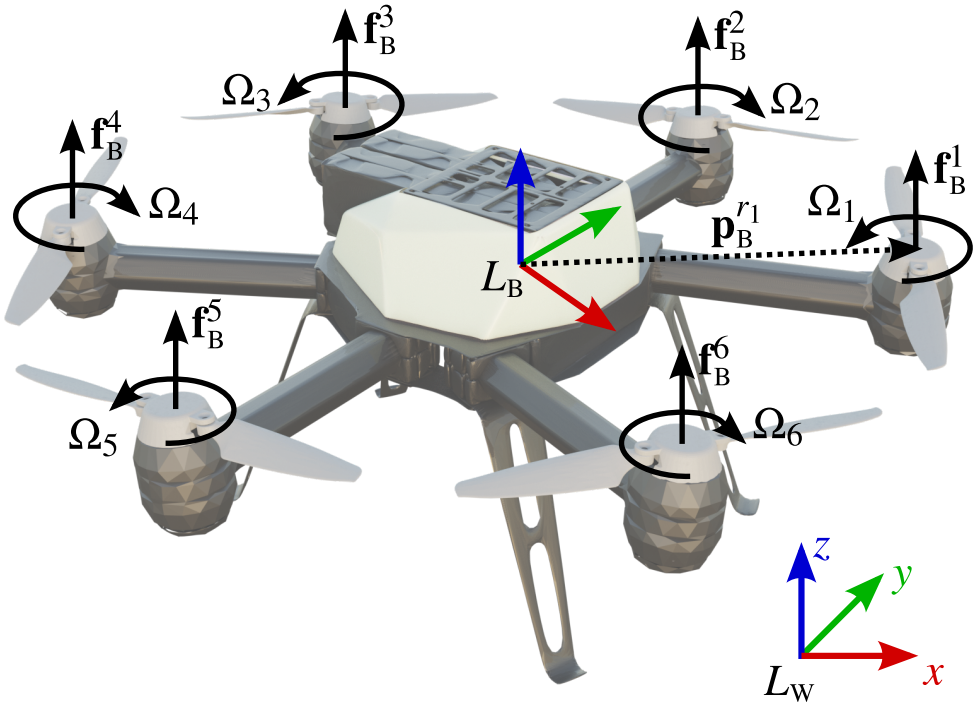


Figure 3.3: A hexacopter multirotor with indicated forces and rotational velocities of each rotor. Vector $\mathbf{p}_\mathrm{B}^{r_1}$ denotes the position of the first rotor in frame L_B, and other position vectors are omitted for clarity. Reaction torque produced by each propeller acts in opposite direction from its rotation.

thrust and drag proportional to the squared rotational velocity:

$$\begin{aligned} F_{r_i} &= c_\mathrm{T}\Omega_{r_i}^2 \\ \tau_{r_i} &= \xi_i c_\mathrm{D}\Omega_{r_i}^2, \end{aligned} \tag{3.20}$$

where c_T and c_D are the thrust and drag coefficients of rotors, respectively. The rotational velocity of each rotor is denoted Ω_{r_i}, and $\xi_i \in \{-1, 1\}$ defines the rotational direction with $\xi_i = -1$ for counter-clockwise rotation and $\xi_i = 1$ for clockwise rotation. Note that the thrust and drag coefficients can differ between propellers. However, due to the fact that most multirotor UAVs are built out of the same motor-propeller pairs, it is assumed that the thrust and drag coefficients do not change between propellers. Since the UAV body and rotor $\mathbf{z}$ axes are always aligned, forces and torques of each propeller can be written in vector form:

$$\begin{aligned} \mathbf{f}_\mathrm{B}^{r_i} &= c_\mathrm{T}\Omega_{r_i}^2\mathbf{z}_\mathrm{B} \\ \boldsymbol{\tau}_\mathrm{B}^{r_i} &= \xi_i c_\mathrm{D}\Omega_{r_i}^2\mathbf{z}_\mathrm{B}. \end{aligned} \tag{3.21}$$

Apart from the rotors' force, there are two additional forces acting on the UAV body: gravity

and force induced by the manipulator:

$$m_{\mathrm{B}}\ddot{\mathbf{p}}_{\mathrm{W}}^{\mathrm{B}} = -m_{\mathrm{B}}\mathbf{g} + \sum_{i=1}^{n_r} \mathbf{R}_{\mathrm{B}}^{\mathrm{W}}\mathbf{f}_{\mathrm{B}}^{r_i} - \mathbf{R}_{0}^{\mathrm{W}}\mathbf{f}_0, \tag{3.22}$$

where m_{B} denotes the UAV body mass, $\ddot{\mathbf{p}}_{\mathrm{W}}^{\mathrm{B}}$ is the UAV linear acceleration in world frame, the sum member takes into account the contributions of rotor forces. The force $\mathbf{f}_0$ is defined through equation (3.16) and represents the force the UAV body experiences from the attached manipulator. Since Newton-Euler model accounts for the gravity, the manipulator mass is already included in this equation through the force $\mathbf{f}_0$.

To account for the moment, the angular velocity has to be defined. As stated in equation (3.1), the $\mathbf{\Theta}_{\mathrm{W}}^{\mathrm{B}}$ defines the orientation of the UAV in the world frame. Taking the time derivative, the angular velocity yields:

$$\boldsymbol{\omega}_{\mathrm{W}}^{\mathrm{B}} = \dot{\mathbf{\Theta}}_{\mathrm{W}}^{\mathrm{B}} = \begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T. \tag{3.23}$$

Usually, the angular velocity is measured in the body frame as $\boldsymbol{\omega}_{\mathrm{B}} = \begin{bmatrix} p & q & r \end{bmatrix}^T$. To express the angular velocity in the world frame, the following transformation is applied:

$$\boldsymbol{\omega}_{\mathrm{W}}^{\mathrm{B}} = \mathbf{W}_{\omega}\boldsymbol{\omega}_{\mathrm{B}} = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi / C_\theta & C_\phi / C_\theta \end{bmatrix} \boldsymbol{\omega}_{\mathrm{B}}, \tag{3.24}$$

where T_α denotes the tangent function, and $\mathbf{W}_\omega$ is a mapping matrix between world and body coordinate systems. This yields the moment equation for the UAV:

$$\mathbf{I}_{\mathrm{B}}\dot{\boldsymbol{\omega}}_{\mathrm{B}} + \boldsymbol{\omega}_{\mathrm{B}} \times (\mathbf{I}_{\mathrm{B}}\boldsymbol{\omega}_{\mathrm{B}}) = \sum_{i=1}^{n_r} (\mathbf{p}_{\mathrm{B}}^{r_i} \times \mathbf{f}_{\mathrm{B}}^{r_i} + \boldsymbol{\tau}_{\mathrm{B}}^{r_i}) + \mathbf{p}_{\mathrm{B}}^{0} \times (\mathbf{R}_{0}^{\mathrm{B}} \cdot (-\mathbf{f}_0)) - \mathbf{R}_{0}^{\mathrm{B}}\boldsymbol{\tau}_0. \tag{3.25}$$

The $\mathbf{I}_{\mathrm{B}} \in \mathbb{R}^{3\times 3}$ is the inertia matrix of the UAV. The sum represents moments acting on the UAV body produced by rotors. The first member $\mathbf{p}_{\mathrm{B}}^{r_i}$ represents the moment produced by the force acting at some distance from the UAV body, and the second member $\boldsymbol{\tau}_{\mathrm{B}}^{r_i}$ is the moment produced by the propeller as stated in equation (3.21). The last two members are exerted by the manipulator onto the system L_0. Again, there is some force $\mathbf{f}_0$ acting at some distance from the body, and moment $\boldsymbol{\tau}_0$, which are derived in equations (3.16) and (3.17).

Having the mathematical model is the first step towards the model-based planning. The overall system motion is also influenced by the employed control strategy. The following section describes the control problem and discusses different aerial manipulator control strategies.

3.2 Control

To control a multirotor vehicle, a pilot can send the rotational velocity reference for each motor using a radio transmitter. It is immediately obvious that such a control is highly impractical and takes a lot of skill and practice to achieve even a hovering flight. It is much more natural to control a multirotor by sending inputs as desired angles and thrust. Achieving such a control requires remapping the control inputs to rotate around the body axes. Applying the angular feedback yields a stable hovering flight which is usually referred to as the attitude control. Having some external disturbances, i.e. wind, can move the multirotor freely in space, even though it keeps a stable orientation. This leads towards the position control, where the problem definition is to keep the desired position in the world frame. Therefore, the multirotor control can be divided in four main parts:

1) *Motor control*. The objective is to reach the desired thrust by varying the rotor angular velocity Ω.
2) *Control allocation.* Based on the desired force and moment, allocate the angular velocity of each rotor.
3) *Attitude control.* Calculates the required moment to reach the desired input angles.
4) *Position control.* Based on the desired position and orientation around world z axis, calculates the required attitude angles and thrust force.

The hierarchical controller structure is depicted on Fig. 3.4. From the perspective of the motion planning, this kind of structure is very practical since it allows directly supplying the desired position and yaw angle, while other control values are determined internally by their respective blocks. Careful reader can notice that not all six degrees of freedom are controlled independently as the roll and pitch angles are computed internally. This is the direct consequence of the multirotor underactuated nature due to the planar rotor configuration, as has been discussed in Section 3.1.2. This is relevant for the following sections describing the model-based planning, because the roll and pitch angles contribute to the end-effector motion. From the perspective of the model-based planning, this motion must be taken into account to accurately track the end-effector, and the control law with its parameters determines the magnitude of this motion.

3.2.1 Control Prerequisites

As shown on Fig. 3.4, the control allocation is required for the attitude and position control to work. The controllers' output is force and moment, which then have to be mapped to the motor velocities through equation (3.21). The control allocation is essentially a matrix depending on the configuration of a multirotor. As discussed earlier, the multirotors considered in this thesis

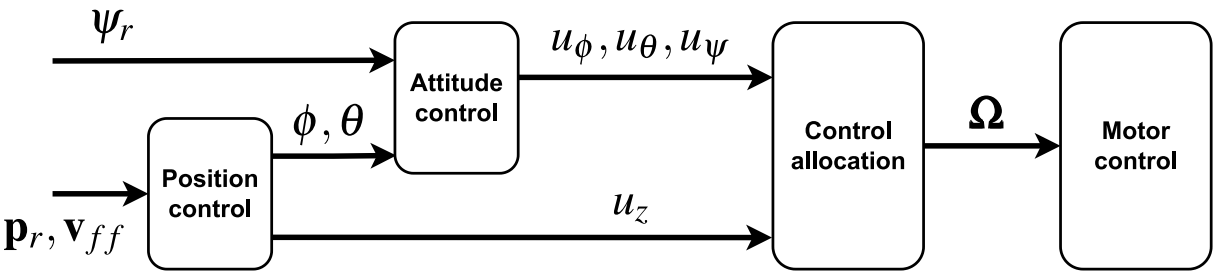


Figure 3.4: The hierarchical visualization of the flight controller. A user, or a motion planning algorithm, supplies the position $\mathbf{p}_r$ and yaw angle ψ_r references, with optional feed forward for the velocity. Attitude and position control internally determine the control inputs, which are afterwards allocated to the motor control.

are underactuated. Therefore, the controller outputs are:

$$\mathbf{u} = \begin{bmatrix} u_\phi & u_\theta & u_\psi & u_z \end{bmatrix}^T, \tag{3.26}$$

where u_ϕ, u_θ, u_ψ are the attitude controller outputs in terms of moments around their respective axes, and u_z is the height controller output in terms of force along the z_W axis. The rotor angular velocities vector can be defined as:

$$\boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 & \Omega_2 & \dots & \Omega_{n_r} \end{bmatrix}^T. \tag{3.27}$$

The relation between controller outputs and rotor velocities is following:

$$\mathbf{u} = \boldsymbol{\Gamma}\boldsymbol{\Omega}_{sq}, \tag{3.28}$$

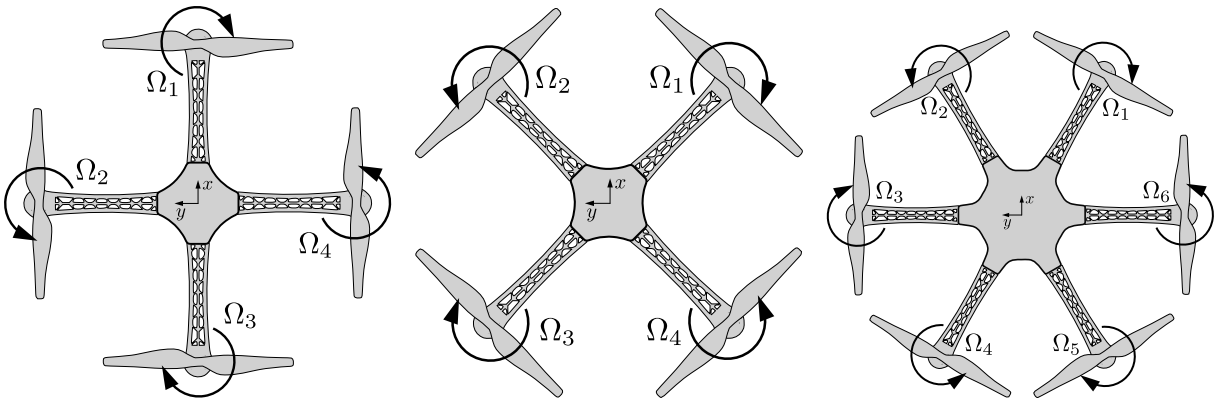
where $\boldsymbol{\Gamma} \in \mathbb{R}^{4\times n_r}$ is the control allocation matrix that depends on the number of motors, and $\boldsymbol{\Omega}_{sq} = \begin{bmatrix} \Omega_1^2 & \dots & \Omega_{nr}^2 \end{bmatrix}^T$ contains the squared velocities of rotors. Inverting this relation yields the rotor angular velocities required to achieve the required thrust and moments.

One of the most popular multirotor configurations is a quadcopter. It offers a minimal number of rotors placed symmetrically around the body center, to achieve a stable flight with four controllable degrees of freedom. Other frequent configurations include hexacopters and octocopters with six and eight rotors, respectively. These configurations are redundant, which is a good feature in recovery due to a broken propeller or arm. However, this goes beyond the scope of this work and it is assumed that the rotors are always in place. Two popular multirotor types are used in this work: quadrotor and hexarotor, shown on Fig. 3.5. Typically, quadcopters are arranged in plus (+) or "x" (×) configuration. In the + configuration, the body x axis is aligned with one of the quadrotor's arms, while in the × configuration it points between two rotors. As an example of control allocation, matrices for both aforementioned quadrotor and hexarotor configurations are given, assuming all rotors are at the same distance l from the body

center:

$$\boldsymbol{\Gamma}_{+} = \begin{bmatrix} 0 & lc_{\mathrm{T}} & 0 & -lc_{\mathrm{T}} \\ -lc_{\mathrm{T}} & 0 & lc_{\mathrm{T}} & 0 \\ c_{\mathrm{D}} & -c_{\mathrm{D}} & c_{\mathrm{D}} & -c_{\mathrm{D}} \\ c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} \end{bmatrix}, \quad \boldsymbol{\Gamma}_{\times} = \begin{bmatrix} -\frac{\sqrt{2}}{2}lc_{\mathrm{T}} & \frac{\sqrt{2}}{2}lc_{\mathrm{T}} & \frac{\sqrt{2}}{2}lc_{\mathrm{T}} & -\frac{\sqrt{2}}{2}lc_{\mathrm{T}} \\ -\frac{\sqrt{2}}{2}lc_{\mathrm{T}} & -\frac{\sqrt{2}}{2}lc_{\mathrm{T}} & \frac{\sqrt{2}}{2}lc_{\mathrm{T}} & \frac{\sqrt{2}}{2}lc_{\mathrm{T}} \\ c_{\mathrm{D}} & -c_{\mathrm{D}} & c_{\mathrm{D}} & -c_{\mathrm{D}} \\ c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} \end{bmatrix} \tag{3.29}$$

$$\boldsymbol{\Gamma}_{hex} = \begin{bmatrix} -\frac{1}{2}lc_{\mathrm{T}} & \frac{1}{2}lc_{\mathrm{T}} & lc_{\mathrm{T}} & \frac{1}{2}lc_{\mathrm{T}} & -\frac{1}{2}lc_{\mathrm{T}} & lc_{\mathrm{T}} \\ -\frac{\sqrt{3}}{2}lc_{\mathrm{T}} & -\frac{\sqrt{3}}{2}lc_{\mathrm{T}} & 0 & \frac{\sqrt{3}}{2}lc_{\mathrm{T}} & \frac{\sqrt{3}}{2}lc_{\mathrm{T}} & 0 \\ c_{\mathrm{D}} & -c_{\mathrm{D}} & c_{\mathrm{D}} & -c_{\mathrm{D}} & c_{\mathrm{D}} & -c_{\mathrm{D}} \\ c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} & c_{\mathrm{T}} \end{bmatrix}$$



(a) Plus configuration: $\boldsymbol{\Gamma}_{+}$ (b) $\times$ configuration: $\boldsymbol{\Gamma}_{\times}$ (c) Hexacopter configuration: $\boldsymbol{\Gamma}_{hex}$

Figure 3.5: Three distinct multirotor configurations used throughout this thesis.

One of the greatest advantages of the control allocation matrix is that the attitude and position control strategies do not need to change across different configurations. Naturally, the control parameters need to be tuned, but the functional schematics stays the same. It can be observed that the matrices from equation (3.29) assume all the rotors have identical parameters. In an arbitrary case, a multirotor vehicle can have different rotors and the parameters would be captured in the control allocation matrix. However, such vehicles are not a common occurrence due to more complicated mechanical design.

The second prerequisite for the attitude control is the motor control itself. In vast majority of cases, multirotors are equipped with brushless Direct Current (DC) motors and Electronic Speed Controllers (ESC) with Pulse Width Modulation (PWM) as the input. This type of motors feature a rapid response to the input, which is important for high performance attitude and position control. To produce thrust, a propeller is fixed on the motor which gives it the same angular velocity as the motor. From the dynamics standpoint, these motors can be approximated with a first order transfer function. The specifics of the motor control are out of the scope of

this thesis, for more details please refer to [1, 124].

3.2.1.1 Linearization

Oftentimes, to synthesize the controller it is necessary to linearize the model. The multirotor mathematical model is therefore linearized around the hovering condition. This implies the angles and angular velocities are close to zero, and the produced thrust counteracts the gravitational force. The control allocation matrices, from equation (3.28), assume a symmetrical vehicle, with the center of mass coinciding with the body coordinate system. In case of attaching a manipulator, the center of mass shifts, which results in different moments produced since propellers are no longer placed symmetrically around the multirotor body. To take the overall center of mass into account, a new coordinate system L_{CM} is introduced and coincides with the aerial manipulator center of mass. The underlying assumption for the linearization is that the manipulator remains static around its home position. Naturally, as manipulator moves, the center of mass also moves with it. This motion is ignored in the linearization process, and the controllers are synthesized around the home position. The center of mass shift during the manipulator motion is bounded within some finite region around the home position, and it is regarded as an unknown disturbance from the controller perspective.

First, the aerial manipulator center of mass needs to be determined. In equation (3.19), the manipulator center of mass is given in the L_0 coordinate system. The overall system center of mass position in the L_0 coordinate system is:

$$\mathbf{p}_0^{\mathrm{CM}} = \frac{m_{\mathrm{B}}\mathbf{p}_0^{\mathrm{B}} + m_{\mathrm{M}}\mathbf{p}_0^{\mathrm{M}}}{m_{\mathrm{B}} + m_{\mathrm{M}}}. \tag{3.30}$$

In the body coordinate system this becomes:

$$\mathbf{p}_{\mathrm{B}}^{\mathrm{CM}} = \mathbf{p}_{\mathrm{B}}^{0} + \mathbf{p}_0^{\mathrm{CM}}. \tag{3.31}$$

As mentioned earlier, the multirotor body is considered to be symmetrical around its center of gravity. This is not the case when observed from L_{CM}, thus, not all rotors produce the same moment since the distance from the center of mass is different. To account for that, it is necessary to define vector from L_{CM} to each rotor:

$$\mathbf{p}_{\mathrm{CM}}^{r_i} = \mathbf{p}_{\mathrm{B}}^{r_i} - \mathbf{p}_{\mathrm{B}}^{\mathrm{CM}}, \tag{3.32}$$

where $\mathbf{p}_{\mathrm{B}}^{\mathrm{CM}}$ is defined in equation (3.31), and $\mathbf{p}_{\mathrm{B}}^{r_i}$ can be obtained by measuring the multirotor or from the Computer Aided Design (CAD) model. Depending on the manipulator contact point, the vector $\mathbf{p}_{\mathrm{CM}}^{r_i} = [x_{\mathrm{CM}}^{r_i}\ y_{\mathrm{CM}}^{r_i}\ z_{\mathrm{CM}}^{r_i}]^T$ has different components for each rotor.

The forces and moments produced by each rotor are defined in equation (3.21). Since the coordinate systems L_{B} and L_{CM} have the same rotation, this equation also applies to the z_{CM}

axis. Total force and moment produced by rotors can be written as:

$$\begin{aligned}\mathbf{f}_{\mathrm{CM}} &= \sum_{i=1}^{n_r} \mathbf{f}_{\mathrm{CM}}^{r_i} = \sum_{i=1}^{n_r} \begin{bmatrix} 0 & 0 & c_{\mathrm{T}}\Omega_{r_i}^2 \end{bmatrix}^T \\ \boldsymbol{\tau}_{\mathrm{CM}} &= \sum_{i=1}^{n_r} \left(\mathbf{p}_{\mathrm{CM}}^{r_i} \times \mathbf{f}_{\mathrm{CM}}^{r_i} + \boldsymbol{\tau}_{\mathrm{CM}}^{r_i}\right) = \sum_{i=1}^{n_r} \begin{bmatrix} c_{\mathrm{T}}\Omega_{r_i}^2 y_{\mathrm{CM}}^{r_i} \\ -c_{\mathrm{T}}\Omega_{r_i}^2 x_{\mathrm{CM}}^{r_i} \\ \xi_{r_i} c_{\mathrm{D}}\Omega_{r_i}^2 \end{bmatrix},\end{aligned} \tag{3.33}$$

where $x_{\mathrm{CM}}^{r_i}$ and $y_{\mathrm{CM}}^{r_i}$ are distances to rotors along the L_{CM} axes, as given by equation (3.32). Due to the displaced center of mass, the control allocation matrix also needs to be adjusted. The matrices derived in equation (3.29) are assuming the L_{B} as the center of mass. Note that these matrices are still accurate representation in case where the center of mass is only displaced along the z_{B} axis. The control allocation matrix can be derived in general form:

$$\boldsymbol{\Gamma} = \begin{bmatrix} c_{\mathrm{T}} y_{\mathrm{CM}}^{r_1} & c_{\mathrm{T}} y_{\mathrm{CM}}^{r_2} & \dots & c_{\mathrm{T}} y_{\mathrm{CM}}^{r_n} \\ -c_{\mathrm{T}} x_{\mathrm{CM}}^{r_1} & -c_{\mathrm{T}} x_{\mathrm{CM}}^{r_2} & \dots & -c_{\mathrm{T}} x_{\mathrm{CM}}^{r_n} \\ \xi_{r_1} c_{\mathrm{D}} & \xi_{r_2} c_{\mathrm{D}} & \dots & \xi_{r_n} c_{\mathrm{D}} \\ c_{\mathrm{T}} & c_{\mathrm{T}} & \dots & c_{\mathrm{T}} \end{bmatrix}. \tag{3.34}$$

The control allocation matrices from equation (3.29) can be observed as a special case of the general matrix $\boldsymbol{\Gamma} \in \mathbb{R}^{4\times n_r}$. The generalized matrix requires a more careful approach when determining its members. The distances from the mass center $x_{\mathrm{CM}}^{r_i}$ and $y_{\mathrm{CM}}^{r_i}$ are calculated through equation (3.32), while ξ_i determines the rotational direction covered in Section 3.1.2.2. It is worth mentioning that the generalized matrix in this form assumes all the propellers have the same parameters. This is not true in an arbitrary case, however, it is easily incorporated in the matrix since in that case each column has unique thrust and moment coefficients.

Recall equations (3.22) and (3.25) describing the multirotor linear and angular motion. Having defined the control allocation matrix, as well as controller inputs, the linear motion equation can be written as:

$$\begin{aligned} m\ddot{\mathbf{p}}_{\mathrm{W}}^{\mathrm{CM}} &= -m\mathbf{g} + \mathbf{R}_{\mathrm{CM}}^{\mathrm{W}}\mathbf{u}_1 \\ \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} &= -g \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{u_z}{m} \begin{bmatrix} C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\phi S_\theta S_\psi - S_\phi C_\psi \\ C_\phi C_\theta \end{bmatrix}, \end{aligned} \tag{3.35}$$

where $\ddot{\mathbf{p}}_{\mathrm{W}}^{\mathrm{CM}}$ is the center of mass linear acceleration expressed in L_{W}, $\mathbf{u}_1 = \begin{bmatrix} 0 & 0 & u_z \end{bmatrix}^T$ is the controller input that can be mapped to the rotors' angular velocities through equation (3.28), and

$m = m_{\mathrm{B}} + m_{\mathrm{M}}$ is the overall system mass. Note that $\mathbf{R}_{\mathrm{W}}^{\mathrm{CM}} = \mathbf{R}_{\mathrm{W}}^{\mathrm{B}}$ since these two coordinate systems have the same orientation. To derive the angular equation, it is required to determine the overall system inertia. The first part is the multirotor body inertia, which is denoted with $\mathbf{I}_{\mathrm{B}}$. The second part is the contribution from each manipulator link, with $\mathbf{D}_i$ representing the inertia tensor in the i-th link center of mass. Both multirotor and each link inertia tensors are considered to be expressed in the principal axes of their respective bodies, which means all of these matrices are diagonal. To transform the inertia tensors to the center of mass coordinate system, the Parallel axis theorem is used:

$$\begin{aligned}\mathbf{I}_{\mathrm{S}} = \mathbf{I}_{\mathrm{B}} + m_{\mathrm{B}} \left(\left(\mathbf{p}_{\mathrm{CM}}^{\mathrm{B}}\right)^T \mathbf{p}_{\mathrm{CM}}^{\mathrm{B}} \mathbf{E}_3 - \mathbf{p}_{\mathrm{CM}}^{\mathrm{B}} \left(\mathbf{p}_{\mathrm{CM}}^{\mathrm{B}}\right)^T \right) \\ + \sum_{i=1}^{n_r} \left[\mathbf{R}_{\mathrm{CM}}^{0} \mathbf{D}_i \left(\mathbf{R}_{\mathrm{CM}}^{0}\right)^T + m_i \left(\left(\mathbf{p}_{\mathrm{CM}}^{c_i}\right)^T \mathbf{p}_{\mathrm{CM}}^{c_i} \mathbf{E}_3 - \mathbf{p}_{\mathrm{CM}}^{c_i} \left(\mathbf{p}_{\mathrm{CM}}^{c_i}\right)^T \right) \right],\end{aligned} \tag{3.36}$$

where $\mathbf{E}_3 \in \mathbb{R}^{3\times 3}$ represents the identity matrix. The Parallel axis theorem observes the inertia tensor around its principal axes and than translates it. The inertia tensor of each link $\mathbf{D}_i$ needs to be rotated first since its principal axes do not match L_{CM} orientation, hence the first member of the sum. Note that the total inertia tensor $\mathbf{I}_{\mathrm{S}}$ obtained this way is not a diagonal matrix. The vector from center of mass to each link breaks down to:

$$\mathbf{p}_{\mathrm{CM}}^{c_i} = \mathbf{p}_{\mathrm{CM}}^{\mathrm{B}} + \mathbf{p}_{\mathrm{B}}^{0} + \mathbf{p}_{0}^{c_i}, \tag{3.37}$$

where $\mathbf{p}_{\mathrm{CM}}^{\mathrm{B}}$ is the inverse vector from the one defined in (3.31), $\mathbf{p}_{\mathrm{B}}^{0}$ is a fixed transform depending on the manipulator attachment point, and $\mathbf{p}_{0}^{c_i}$ is obtained from (3.14) as a part of Newton-Euler forward dynamics. The angular acceleration equation is:

$$\mathbf{I}_{\mathrm{S}} \dot{\boldsymbol{\omega}}_{\mathrm{CM}} = -\boldsymbol{\omega}_{\mathrm{CM}} \times (\mathbf{I}_{\mathrm{S}} \boldsymbol{\omega}_{\mathrm{CM}}) + \mathbf{u}_2, \tag{3.38}$$

where $\boldsymbol{\omega}_{\mathrm{CM}} = \boldsymbol{\omega}_{\mathrm{B}} = \begin{bmatrix} p & q & r \end{bmatrix}^T$ represents the angular velocity in the L_{CM} coordiante system, and $\mathbf{u}_2 = \begin{bmatrix} u_\phi & u_\theta & u_\psi \end{bmatrix}^T = \boldsymbol{\Gamma}\boldsymbol{\Omega}_{sq}$ contains the moment inputs. As mentioned earlier, $\mathbf{I}_{\mathrm{S}}$ contains the off-diagonal elements often called products of inertia. The diagonal elements are far greater in magnitude than the off-diagonal ones. Therefore, in the linearization process it is safe to assume the inertia tensor is a diagonal matrix, while ignoring the products of inertia. Taking these assumptions into account, the previous equation expands to:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{Iyy - Izz}{Ixx} qr \\ \frac{Izz - Ixx}{Iyy} pr \\ \frac{Ixx - Iyy}{Izz} pq \end{bmatrix} + \begin{bmatrix} \frac{u_\phi}{Ixx} \\ \frac{u_\theta}{Iyy} \\ \frac{u_\psi}{Izz} \end{bmatrix}, \tag{3.39}$$

with assumption $\mathbf{I}_{\mathrm{S}} = diag(I_{xx}, I_{yy}, I_{zz})$. This equation is expressed in the center of mass frame,

and transforming the obtained angular accelerations to the world frame can be done with:

$$\begin{aligned}\dot{\boldsymbol{\omega}}_{\mathrm{W}}^{\mathrm{CM}} &= \frac{d}{dt}\left(\mathbf{W}_{\omega}\cdot\boldsymbol{\omega}_{\mathrm{CM}}\right) = \frac{d}{dt}\left(\mathbf{W}_{\omega}\right)\boldsymbol{\omega}_{\mathrm{CM}} + \mathbf{W}_{\omega}\boldsymbol{\alpha}_{\mathrm{CM}} \\ &= \begin{bmatrix} 0 & \dot{\phi}C_{\phi}T_{\theta} + \dot{\theta}S_{\phi}C_{\theta}^{2} & -\dot{\phi}S_{\phi}T_{\theta} + \dot{\theta}C_{\phi}C_{\theta}^{2} \\ 0 & -\dot{\phi}S_{\phi} & -\dot{\phi}C_{\phi} \\ 0 & \dot{\phi}C_{\phi}/C_{\theta} + \dot{\theta}S_{\phi}T_{\theta}/C_{\theta} & -\dot{\phi}S_{\phi}/C_{\theta} + \dot{\theta}C_{\phi}T_{\theta}/C_{\theta} \end{bmatrix}\boldsymbol{\omega}_{\mathrm{CM}} + \mathbf{W}_{\omega}\boldsymbol{\alpha}_{\mathrm{CM}},\end{aligned} \tag{3.40}$$

where $\mathbf{W}_{\omega}$ is the angular velocity mapping matrix defined in equation (3.24), and $\boldsymbol{\alpha}_{\mathrm{CM}}$ is the center of mass angular acceleration expressed in L_{CM}. To synthesize the controller, hovering conditions are assumed. This implies small angles, yielding the cosine function $C_{\alpha} \approx 1$ and the sine function $S_{\alpha} \approx \alpha$. Furthermore, the angular velocities are also small, which yields product of multiple angular velocity components as zero. A simplified linearized model based on equations (3.35) and (3.39) can be written as:

$$\begin{aligned} \ddot{x} &= g\cdot\theta, \quad \ddot{y} = -g\cdot\phi, \quad \ddot{z} = \frac{u_z}{m} \\ \dot{p} &= \frac{u_{\phi}}{I_{xx}}, \qquad \dot{q} = \frac{u_{\theta}}{I_{yy}}, \qquad \dot{r} = \frac{u_{\psi}}{I_{zz}} \end{aligned} \tag{3.41}$$

The linearized model contains the contributions of the manipulator in the form of the overall mass and inertia of the system. The manipulator is considered to be in a home position for the linearization purposes, which can vary depending on the use case. Recall equations (3.22) and (3.25) describing the multirotor linear and angular motion. These equations also contain the forces and torques produced by the manipulator motion, acting on the multirotor body. Throughout the linearization procedure, these contributions are considered as disturbances and are ignored. By measuring the manipulator state, it is possible to obtain these values in the real world. Thus, these forces and moments can be included in the control structure as feed forward terms, with the intention to be compensated by the multirotor control strategy. However, the manipulator motion dynamics is usually faster than the multirotor dynamics. This could produce a rapidly changing signal as the input to the manipulator control, which is not a desired behavior and can potentially lead to actuator saturation or even damage, which is the main reason these contributions are ignored in the control strategy. Note that the home configuration described here can also be extended to a certain task configuration around which the task is executed. The manipulator is assumed to deviate within some boundary region around the task configuration, yielding a bounded change of inertia. This assumption is taken into account in later sections describing the control algorithm.

3.2.2 Attitude and Position Control

A typical approach when designing the multirotor controller is separating the attitude and position control. This allows for both controllers to be tuned separately, with the position controller generating references for the attitude controller. Furthermore, different control principles can be employed for both loops. In this section, a standard cascade Proportional-Integral-Derivative (PID) control law is applied for both low level attitude and high level position control.

3.2.2.1 Attitude Control

The attitude control refers to setting the desired angles and achieving them by applying some control law. In practice, the PID control laws are mostly applied to the multirotor control due to their implementation and tuning simplicity. Within this section, a cascade attitude controller is derived with the inner loop controlling the angular velocities and outer loop controlling the angles.

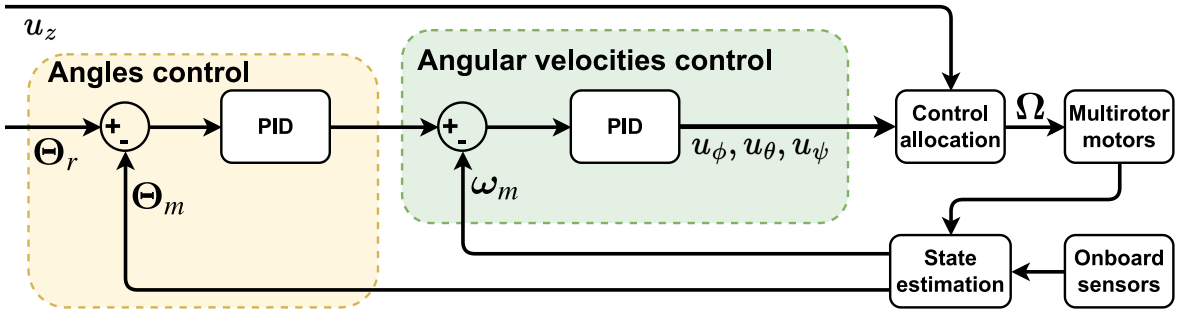


Figure 3.6: Attitude control schematic with inner loop controlling the angular velocity and outer loop controlling angles. The u_z control input is considered to be supplied by the position controller.

The cascade attitude control schematic can be observed on Fig. 3.6. The input to the attitude control is a vector of desired angles $\boldsymbol{\Theta}_r = [\phi_r \ \theta_r \ \psi_r]^T$. This can be supplied either directly by the pilot, or as an output from the position controller. The backbone of the cascade is the PID controller that can be written as:

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de}{dt}, \tag{3.42}$$

where $e(t)$ is the error signal between the reference and measurement, while K_p denotes the proportional gain, K_i integral gain, and K_d derivative gain. Note that the PID parameters are different for each controlled angle and angular velocity. In cascade attitude control, the outer loop that controls the angle generates the reference for the inner loop, usually referred to as *rates*. The inner loop controls the angular velocities in the multirotor body frame.

As stated in Section 3.2.1.1, the multirotor model is linearized for hovering conditions and for the manipulator nominal configuration depending on the task at hand. Naturally, the manip-

ulator movement and configuration affects the overall system stability. An example of a stability analysis is performed in the remainder of this section. In the analysis, three static manipulator configurations are taken into account by varying the overall system inertia. These correspond to the manipulator nominal configuration $I_{xx,n}$, fully extended configuration maximizing the inertia $I_{xx,max}$, and fully contracted configuration minimizing the overall inertia $I_{xx,min}$. This analysis is performed for the pitch angle controller, and an analogous stability analysis can be performed for both roll and yaw control loops. The attitude controller building blocks from Fig. 3.6 can be written in the Laplace form as:

$$G_p(s) = \frac{1}{I_{xx}s}, \quad G_m(s) = \frac{K_m}{1+T_m s}, \quad G_{c,\omega_y} = K_d, \quad G_{c,\theta} = K_p + \frac{K_i}{s}, \tag{3.43}$$

where $G_p(s)$ is the simplified multirotor model with integral behavior, $G_m(s)$ is the transfer function of motors, approximated with first order dynamics with the gain K_m and time constant T_m, G_{c,ω_y} is the inner loop controller with proportional behavior, and $G_{c,\theta}$ is the outer loop PI controller. This controller structure is known as PI-D form in literature. The overall closed loop of the system is:

$$G_{cl,\theta}(s) = \frac{\theta(s)}{\theta_r(s)} = \frac{K_m K_d (K_i + K_p s)}{I_{xx}T_m s^4 + I_{xx}s^3 + K_m K_d s^2 + K_m K_p K_d s + K_m K_d K_i}. \tag{3.44}$$

The stability is analyzed with the well known Routh-Hurwitz criterion, taking into account the characteristic polynomial of the closed system. Two necessary conditions for the system stability are obtained:

$$\begin{aligned} K_d K_m (1 - K_p T_m) > 0 \Rightarrow K_p < \frac{1}{T_m} \\ I_{xx} K_i < K_p K_m K_d (1 - K_p T_m). \end{aligned} \tag{3.45}$$

To guarantee the overall system stability, these two conditions need to be satisfied by varying the three controller parameters K_p, K_i and K_d. To visualize the stability region, some parameters need to be fixed. The motor parameters are chosen as a typical DC motor with $K_m = 75Vs/rad$ and $T_m = 0.05s$. The inertia is also approximated with a typical multirotor at $I_{xx,n} = 0.1kg \cdot m^2$, with maximum inertia at $I_{xx,max} = 0.15kg \cdot m^2$ and minimum at $I_{xx,min} = 0.05kg \cdot m^2$. The stability region is visualized on Fig. 3.7, with boundaries for nominal, maximum and minimum inertia.

3.2.2.2 Position Control

Similar to the attitude controller, the position control is also based on a cascade PID structure, depicted on Fig. 3.8. The controller expects the position reference $\mathbf{p}_r$ as the input and computes the velocity reference. The inner loop generates references for roll and pitch angles, as well

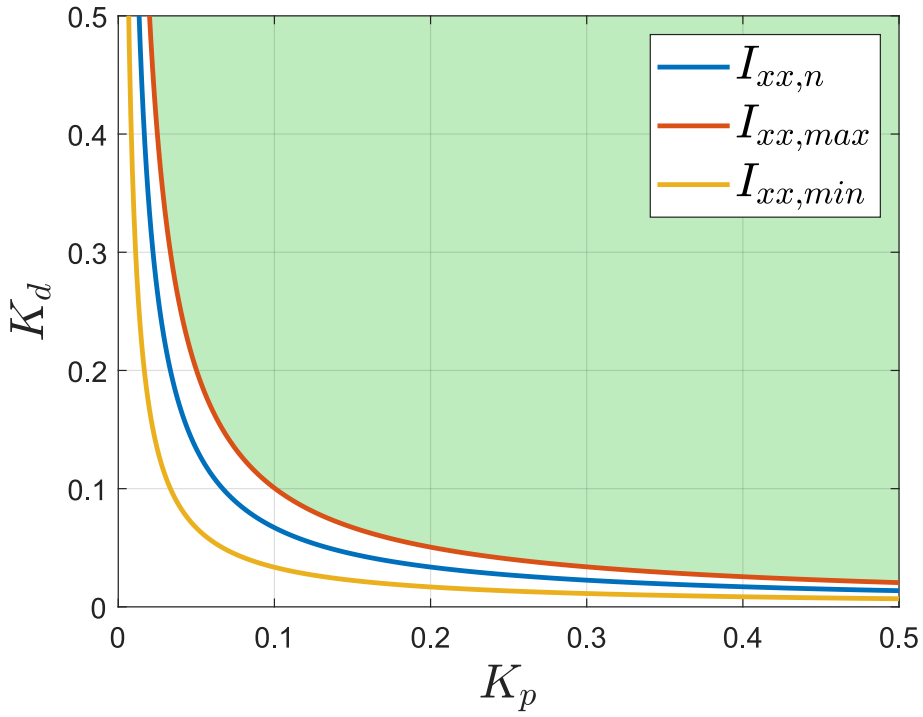


Figure 3.7: The attitude controller stability analysis example. With the fixed integral gain K_i, the stability region (shaded green) depends on K_p and K_d. The stability region also depends on the aerial manipulator inertia, with the three boundaries indicated on the plot.

as thrust which is forwarded directly to the control allocation block. The yaw reference ψ_r is also an expected input and is directly forwarded to the attitude controller. Furthermore, the controller can optionally receive the velocity feed forward v_{ff}, which is most commonly determined through the trajectory planner.

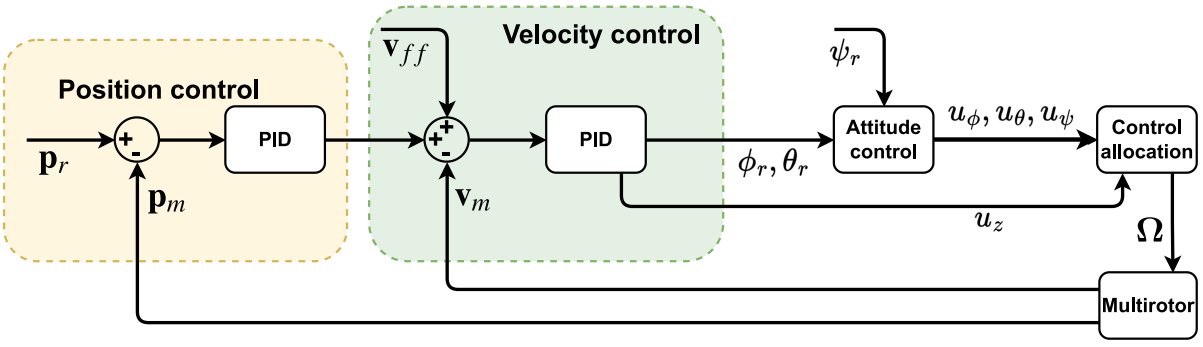


Figure 3.8: The position controller schematic with inner the loop controlling velocity and outer loop controlling position. The position controller outputs roll and pitch references, as well as thrust. The yaw reference is supplied directly to the attitude controller. The velocity controller also features feed forward term.

The described cascade controller structure is widely used for multirotors, and is therefore chosen as an example for this thesis. It is simple to implement, not computationally expensive, and relatively simple to tune. In the real world, the attitude controller is most commonly implemented on a flight controller unit. As the attitude controller, the position control can also

be implemented on a flight controller, while in some cases it is implemented on a companion computer. Although this introduces more complexity in terms of communication and design, it offers implementation of more computationally expensive controllers such as Model Predictive Control, optimal control approaches, impedance control, etc. The comprehensive review on the multirotor control approaches can be found in [48].

3.2.2.3 Manipulator Control

A short note on the manipulator control is provided here. The manipulator is considered to be constructed of multiple rotational joints connected in series. Each joint has its own independent control that consists of a DC motor, gear reductor and electronics. Therefore, the joint angle control loop is closed directly on the low level electronics, and in vast majority of cases it is some form of the PID control law. Thus, each manipulator joint receives an independent angular reference.

3.2.2.4 Differential Flatness

Differentially flat systems have a unique property that all states and inputs can be expressed in terms of outputs and outputs' derivatives. If so, the outputs are commonly referred to as flat outputs. This property is very useful when the system has to follow a trajectory. If states and inputs can be written in terms of the generated trajectory, it is possible to calculate inputs required to track that trajectory and check if the trajectory satisfies feasibility conditions. In case of a linear system differential flatness is equivalent to controllability as stated in [128]. This is extremely useful as it is much easier to check if a system is controllable than to find a set of equations that prove the differential flatness for that system.

In an arbitrary case of a non-linear system, the differential flatness property may not be evident. Although the system in this thesis is linearized for the control purposes, in reality it is still a non-linear system. It is shown in [81], that a non-linear quadcopter model is a differentially flat system, and that property was tested on aggressive trajectories. In fact, this also stands for other popular configurations with co-planar rotor setup, like hexacopters or octocopters. However, in this thesis the emphasis is on aerial manipulators, where the dynamic model includes the robotic arm and its motion. This problem is addressed in [129], where researchers observe an arbitrary number of manipulators with their respective first joint attached to the multirotor center of mass, which they call *protocentric aerial manipulator*. The proof of differential flatness is provided for such a class of systems. Although the mathematical model presented in Section 3.1.2.2 considers an arbitrary attachment point on the multirotor body, the aforementioned proof can still be employed under some assumptions. When designing an aerial manipulator, the first joint is placed as close as possible to the body center of mass. Furthermore, aerial manipulation tasks considered in this thesis do not require aggressive maneuvers or fast motions. Quite on

the contrary, the aerial manipulator motion is observed in near hover conditions, especially in tasks such as wall contact. This allows for exploiting the differential flatness property even if the aerial manipulator is not protocentric, because the control inputs are well below saturation.

CHAPTER 4

Motion Planning

This chapter presents the motion planning problem in aerial robotics and gives the necessary prerequisites for the following chapters. Generally speaking, the objective of motion planning is to find a feasible path and trajectory based on the robot initial and goal state. In this thesis, the first step towards a feasible motion is path planning. A path can be defined as a set of waypoints, either in the task or configuration space, that a robot has to follow. Usually, the objective of the path planning is to find a collision-free path based on the environment map. This is followed by the trajectory planning, where the velocity profile is generated respecting the system's constraints. Such a pipeline enables the robot to navigate the environment without collisions, respecting the imposed spatial and dynamical constraints. It also enables a wide variety of tasks that can be accomplished, making aerial manipulators a versatile option.

4.1 Planning Prerequisites

Within this section, some prerequisites and definitions are given since the path planning algorithm relies on this information. The planning problem can be represented in both high dimensional aerial manipulator configuration space, or in the 6-DoF end-effector configuration. The main advantage of the latter is a fixed degree of freedom in planning. However, transforming the end-effector velocities and accelerations later in the trajectory planning can violate the dynamic constraints. Therefore, it is safer to plan in the configuration space of the robot because the constraints can be supplied for each joint independently. The planner relies on the differential flatness property of the system, which has been addressed in Section 3.2.2.4. Although the aerial manipulator systems used in this thesis are not strictly differentially flat, the general principle for dynamical constraints can still be employed to some extent. Namely, the velocity

and acceleration constraints are set more conservatively than for the differentially flat system, to ensure the control inputs remain within the feasible bounds. The main disadvantage of the configuration space planning is solving the high-dimensional planning problem. To overcome this, the path and trajectory planners are carefully chosen to successfully solve high-dimensional problems in relatively short amount of time.

To start, the configuration space of the aerial manipulator needs to be defined. The multirotor by itself is a 6-DoF system, described with position and orientation:

$$\mathbf{q}_{\mathrm{B}} = \begin{bmatrix} x & y & z & \phi & \theta & \psi \end{bmatrix}^T, \tag{4.1}$$

where $\mathbf{q}_{\mathrm{B}}$ represents the multirotor configuration. In Section 3 it has been stated that the multirotors UAVs with co-planar rotors are underactuated systems. Since it is not possible to control all six DoFs independently, only four are chosen as the position and yaw angle. The body configuration is still considered to be 6-DOF, however, the planner needs to be constrained to assume zero roll and pitch angles. As stated in Section 3.1.2.1, the manipulator is considered to have n_{M} DoFs where the configuration vector can be written as:

$$\mathbf{q}_{\mathrm{M}} = \begin{bmatrix} q_1 & q_2 & \dots & q_{n_{\mathrm{M}}} \end{bmatrix}^T. \tag{4.2}$$

Combining the multirotor and manipulator degrees of freedom yields:

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_{\mathrm{B}}^T & \mathbf{q}_{\mathrm{M}}^T \end{bmatrix}^T, \tag{4.3}$$

where $\mathbf{q} \in \mathbb{R}^{6+n_{\mathrm{M}}}$ is the aerial manipulator generalized coordinates vector.

4.2 Path Planning

A typical input to the path planning algorithm is a set of waypoints a robot needs to visit. Each waypoint is defined as a specific aerial manipulator configuration, generally speaking waypoint $\mathbf{w} = \mathbf{q}$. However, if the problem is planning for multiple robots simultaneously, each waypoint can be extended to:

$$\mathbf{w} = \begin{bmatrix} \mathbf{q}_1^T & \mathbf{q}_2^T & \dots & \mathbf{q}_{n_r}^T \end{bmatrix}^T, \tag{4.4}$$

where n_r defines the number of robots. This definition is given for a general case with multiple robots. Also, note that waypoints can contain some other parameters, such as force if the mission objective is the wall contact, which is considered in following sections. In this section, the path planning is observed from the perspective of a single aerial manipulator. The objective of the path planner is to find an obstacle-free path in the environment between the mission

specific start $\mathbf{w}_s$ and goal $\mathbf{w}_g$. The output path is a set of piecewise-straight line connected waypoints:

$$\mathscr{P} = \left\{ \mathbf{w}_i \mid i \in (1, 2, \ldots, n_p) \right\}, \tag{4.5}$$

where n_p denotes the number of waypoints. The dimension of each waypoint depends on the robot configuration, which is relatively high dimensional in aerial manipulation cases. Therefore, it is necessary to select a path planner capable of producing piecewise-straight obstacle-free path while dealing with high dimensional configurations.

To deal with the aforementioned requirements, the Rapidly-exploring Random Tree (RRT) path planner is chosen [78]. This is a sampling based planner capable of quickly exploring large environments, while handling high dimensional robot configurations. It is well established within the robotic community and often used with static and mobile manipulators. The RRT algorithm starts from the start node $\mathbf{w}_s$ and performs a random sampling in multiple directions. Valid states from the random sample are added to the tree and the same search is performed again for the newly added nodes. This way, the tree grows in all directions trying to reach the goal state $\mathbf{w}_g$. The output is the first path that connects the start and goal nodes. In this thesis, a variant of the RRT algorithm called RRT* is employed. The main difference between the two algorithms is that the RRT* refines the first feasible path within a specified time limit. Therefore, the final path obtained in this manner is shorter than the initial solution. An example of a path generated by the RRT* algorithm is shown on Fig. 4.1. The RRT* implementation used in this thesis is from the widely accepted Open Motion Planning Library [103].

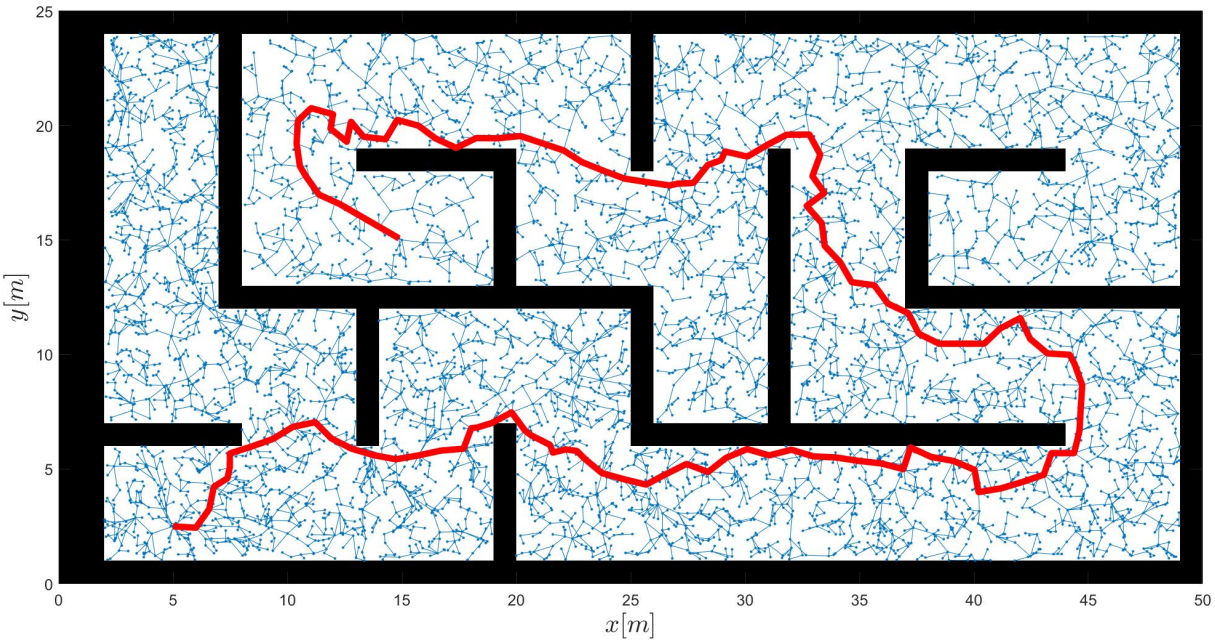
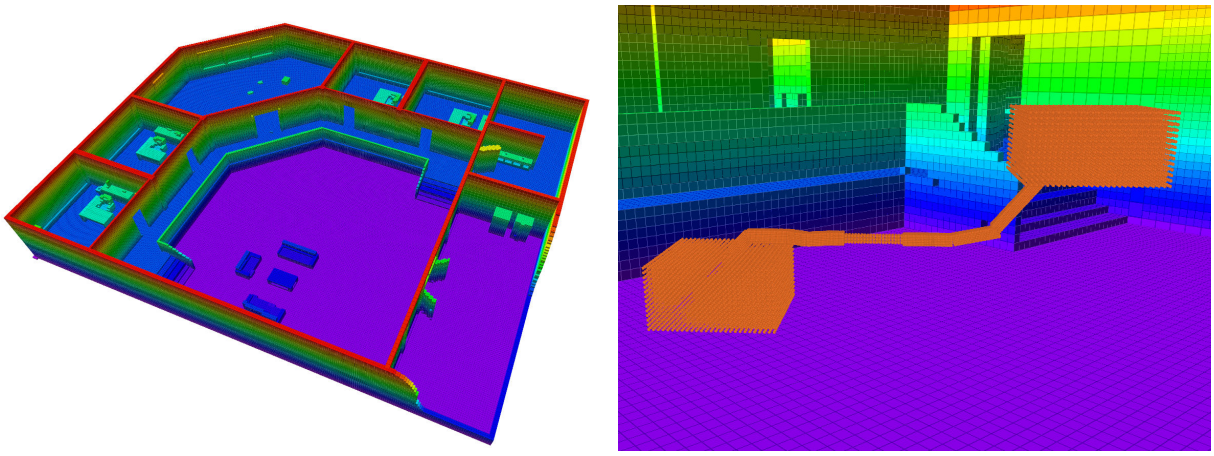


Figure 4.1: An illustrative example of RRT* path planning algorithm in 2D space. The planner produces an obstacle-free path between the start and goal configurations.

4.2.1 Environment Representation

In order to validate a certain configuration $\mathbf{w}$ sampled by the path planner, the environment needs to be represented, which can be done in the form of an occupancy grid. If a 3D environment is taken into account, it can be sampled into cubes of certain size, commonly called voxels. Each voxel can be thought of as occupied or free, holding the necessary information for the state validity checker. A simple implementation of such a space decomposition is a 3-dimensional array corresponding to three spatial dimensions. For large environments or high resolution, this approach can lead to high memory consumption which limits the computer performance.

To alleviate this problem, an efficient algorithm for 3D space decomposition called *OctoMap* has been proposed in [71]. It is a hierarchical volumetric representation of the environment with 16 distinct resolution depths. This allows for efficient querying and memory usage since voxels from a higher depth can be grouped together. An example of the OctoMap is given on Fig. 4.2a. The OctoMap representation is well suited for collision checking, however, the aerial manipulator itself also needs to be represented as a 3D object. Since the OctoMap querry requires a 3D point, the aerial manipulator is sampled with a specified resolution, as depicted on Fig. 4.2b. The multirotor body is considered to be a rectangular prism with the corresponding dimensions, sampled with a specified resolution. Naturally, there is a trade-off between the number of sampled points and collision checking time. In this work, the spatial resolution for all axes of the body is kept at $5cm$, which is chosen based on empirical results. It offers very detailed body representation with limited number of points needed to be checked in the path planning. The manipulator links are also considered to be rectangular prisms, however, the sampling resolution is lowered to $3cm$ due to their smaller dimensions. To properly check for collisions, it is necessary to transform the links' locations to the world frame. This is covered in Sections 3.1, particularly in equations (3.5) and (3.14).



(a) OctoMap representation of an environment.

(b) State representation of two aerial manipulators.

Figure 4.2: The OctoMap environment representation and a state of a complex system consisting of two aerial manipulators transporting an object. Each state point is checked for collision during path planning.

4.3 Trajectory Planning

As stated in the previous section, the path planner output is a series of waypoints ensuring an obstacle-free path. If the aerial manipulator was to move in a point-to-point manner, it would need to stop at each waypoint. This is not a desired behavior and it can lead to significant increase in execution time, depending on the number of waypoints. A typical trajectory planner needs to be supplied with dynamical constraints in terms of the generalized velocity and acceleration of each joint. The task of the trajectory planning is to generate a smooth motion based on the underlying path, while respecting dynamical constraints. The differential flatness property, discussed in Section 3.2.2.4, allows supplying constraints for each joint independently.

There are numerous trajectory planners that are able to produce a smooth trajectory. Researchers in [81] use convex optimization to plan a spline trajectory minimizing the 4^{th} position derivative, often called snap. In our previous work [61, 62], we extended the Ho-Cook method described in [125]. Namely, the order of splines in [125] is 4-3-4, which allows for specifying position and velocity at the start and end of each spline, with additional acceleration at the beginning of the first and the end of the last spline. This has been extended to orders 6-5-6 to include acceleration constraints at each waypoint. Both methods are developed for four DoF of a multirotor vehicle. Even though the number of DoFs is limited, both methods planning time rapidly increased with number of waypoints. A thirty waypoint problem was typically solved in several minutes, which is a significant amount of time if the multirotor is airborne. Extending these methods to arbitrary number of DoFs would render them suitable for aerial manipulator trajectory planning, however, it would only increase planning time.

To resolve this problem, a different class of the trajectory planner is mainly employed in this thesis. The Time Optimal Path Parameterization (TOPP) [94] is a numerical integration approach that operates on the bang-bang principle of the actuator control value. The algorithm supports high dimensional problems which is required for aerial manipulation. Furthermore, by employing the bang-bang principle it ensures at least one is in the velocity or acceleration limit, yielding a time optimal trajectory. The input to the TOPP is a path as described in equation (4.5). If each waypoint within path is of dimension $\mathbf{w} \in R^{n_w}$, the velocity and acceleration constraints can be written as:

$$\begin{aligned} \mathbf{v}^{max} &= \begin{bmatrix} v_1^{max} & v_2^{max} & \dots & v_{n_w}^{max} \end{bmatrix}^T \\ \mathbf{a}^{max} &= \begin{bmatrix} a_1^{max} & a_2^{max} & \dots & a_{n_w}^{max} \end{bmatrix}^T, \end{aligned} \tag{4.6}$$

where v_i^{max} and a_i^{max} are the generalized velocity and acceleration constraints of each joint.

Based on the planned path and dynamical constraints, the output trajectory can be written as:

$$\mathscr{T}_{\mathrm{C}} = \left\{ \mathbf{x}(t) \mid \mathbf{x}(t) \in \mathbb{R}^{3 \cdot n_w}, t \in (0, t_{end}) \right\}, \tag{4.7}$$

where $\mathbf{x} = \begin{bmatrix} \mathbf{w}^T & \dot{\mathbf{w}}^T & \ddot{\mathbf{w}}^T \end{bmatrix}^T$ is one trajectory point with position, velocity and acceleration in the generalized coordinates of the aerial manipulator, t is time, and t_{end} the trajectory duration. In practical implementation, the system is always discretized with some time period T_s. Therefore, the trajectory also needs to be discretized with the same time period to be properly used as reference:

$$\mathscr{T}_{\mathrm{D}} = \left\{ \mathbf{x}(kT_s) \mid \mathbf{x}(kT_s) \in \mathbb{R}^{3 \cdot n_w}, k \in (0, \ldots, n_t) \right\}, \tag{4.8}$$

where n_t is the number of discretized time steps based on the trajectory duration t_{end}.

4.3.1 Time Optimal Path Parameterization Example

To give the reader an idea how the TOPP algorithm from [94] works, a trajectory is planned on a simple two dimensional example. The TOPP derivation in this section follows the work from [94] and [130]. The underlying path is a semicircle in $x-y$ plane, which can be parameterized as:

$$\mathbf{q}(t) = \mathbf{h}(s) = \begin{bmatrix} r \cdot cos(s) \\ r \cdot sin(s) \end{bmatrix}, \tag{4.9}$$

where $\mathbf{q} = \begin{bmatrix} x & y \end{bmatrix}^T$ is the simple 2D robot state, s is the parameterization variable, r is the circle radius, and $\mathbf{h}(s)$ is the parameterization function. To generate a semicircle from this parameterization, it is sufficient to set the parameter $s \in (0, \pi)$. Differentiating $\mathbf{q}$ over time yields velocities and accelerations in the robot configuration space:

$$\begin{aligned} \dot{\mathbf{q}}(t) &= \frac{d}{dt}\mathbf{h}(\mathbf{s}) = \frac{d}{ds}\mathbf{h}(s)\frac{d}{dt} = \mathbf{q}_s \dot{s} = \begin{bmatrix} -r \cdot sin(s)\dot{s} \\ r \cdot cos(s)\dot{s} \end{bmatrix} \\ \ddot{\mathbf{q}}(t) &= \mathbf{q}_s \ddot{s} + \mathbf{q}_{ss}\dot{s}^2 = \begin{bmatrix} -r \cdot sin(s)\ddot{s} - r \cdot cos(s)\dot{s}^2 \\ r \cdot cos(s)\ddot{s} - r \cdot sin(s)\dot{s}^2 \end{bmatrix}. \end{aligned} \tag{4.10}$$

The above parameterization is given as a guideline to the reader on how to geometrically approach such a problem. In general, with multiple degrees of freedom, it may not be evident how to parameterize an arbitrary path given the input waypoints using a geometrical approach. A general way to approach the parameterization is using a polynomial interpolation between the provided waypoints. The user may then provide an arbitrary number of waypoints n_p. In

this example, each segment between two waypoints is interpolated using a cubic spline, with the initial time guess based on the distance between waypoints, assuming a constant velocity:

$$\Delta t_i = |q_i - q_{i-1}|, i \in (1 \dots n_p), \tag{4.11}$$

where Δt_i is the spline duration between two subsequent waypoints. The waypoints, underlying piecewise-straight path, and parameterized path using the cubic splines is shown on Fig. 4.3. The cubic spline parameterization still follows the principles set by the equation (4.10), however, the path differentials q_s and q_{ss} are determined through cubic spline derivatives.

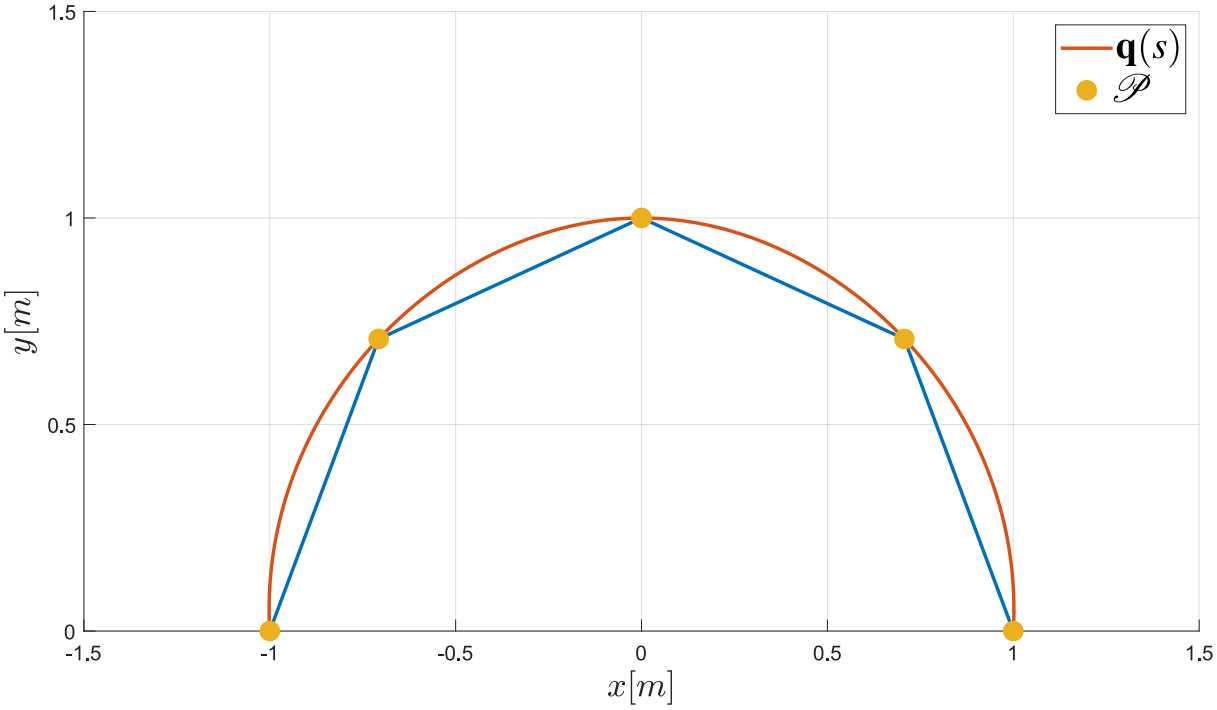


Figure 4.3: Based on the input path $\mathscr{P}$, the parameterized path $\mathbf{q}(s)$ is generated using the cubic splines. The underlying piecewise-straight path is shown in blue.

This parameterization can be interpreted as a reduction to a single dimension, through which the time optimal trajectory can be found. The constraints in a high dimensional configuration space from equation (4.6) are converted to the scalar path velocity $\dot{s}(s)$ and acceleration $\ddot{s}(s)$. Each degree of freedom is considered to have separate velocity and acceleration constraint:

$$\begin{aligned} -\dot{q}_i^{max} \leq \dot{q}_i \leq \dot{q}_i^{max} \\ -\ddot{q}_i^{max} \leq \ddot{q}_i \leq \ddot{q}_i^{max}. \end{aligned} \tag{4.12}$$

The acceleration constraints are first taken into account. Using equation (4.10), the above ex-

pression can be expanded to:

$$\begin{aligned} -\ddot{q}_i^{max} \leq q_{s,i}\ddot{s} + q_{ss,i}\dot{s}^2 \leq \ddot{q}_i^{max} \\ -c_i(s) \leq a_i(s)\ddot{s} + b_i(s)\dot{s}^2 \leq c_i(s), \end{aligned} \tag{4.13}$$

where $a_i(s) = q_{s,i}$, $b_i(s) = q_{ss,i}$, and $c_i(s) = \ddot{q}_i^{max}$. Although this substitution seems unnecessary at a first glance, it is important for including the torque constraints, as explained in Section 4.3.1.1. This substitution allows the same notation in the reminder of this derivation for both acceleration and torque constraints. The goal is to determine the parameterized path acceleration $\ddot{s}$, which is the basis for the numerical integration as both path velocity $\dot{s}$ and path s can be determined by integrating the acceleration. There are three cases that arise in determining the acceleration:

$$\begin{aligned} a_i(s) > 0 &\Rightarrow \frac{-c_i(s) - b_i(s)\dot{s}^2}{a_i(s)} \leq \ddot{s} \leq \frac{c_i(s) - b_i(s)\dot{s}^2}{a_i(s)} \\ a_i(s) < 0 &\Rightarrow \frac{-c_i(s) - b_i(s)\dot{s}^2}{a_i(s)} \geq \ddot{s} \geq \frac{c_i(s) - b_i(s)\dot{s}^2}{a_i(s)} \\ a_i(s) = 0 &\Rightarrow \dot{s} \leq \sqrt{\frac{c_i}{|b_i(s)|}}. \end{aligned} \tag{4.14}$$

Note that in the last case, the path acceleration cannot be directly determined.

The path acceleration is then bounded between the minimum and maximum accelerations:

$$\alpha(s,\dot{s}) \leq \ddot{s} \leq \beta(s,\dot{s}), \tag{4.15}$$

where $\alpha(s,\dot{s})$ is the minimum acceleration profile, and $\beta(s,\dot{s})$ is the maximum acceleration profile. These profiles are determined across all degrees of freedom:

$$\begin{aligned} \alpha(s,\dot{s}) &= \max_{i \in (1 \ldots n_w)} \left(\frac{-c_i(s)}{|a_i(s)|} - \frac{b_i(s)\dot{s}^2}{a_i(s)} \right) \\ \beta(s,\dot{s}) &= \min_{i \in (1 \ldots n_w)} \left(\frac{c_i(s)}{|a_i(s)|} - \frac{b_i(s)\dot{s}^2}{a_i(s)} \right), \end{aligned} \tag{4.16}$$

where n_w is the robot configuration dimension. Naturally, the minimum acceleration alpha profiles must always remain below the maximum acceleration beta profiles $\alpha(s,\dot{s}) \leq \beta(s,\dot{s})$. This must also be true for all combinations of degrees of freedom, which ultimately yields the

following inequality:

$$-\left|\frac{b_i(s)}{a_i(s)}-\frac{b_j(s)}{a_j(s)}\right|\dot{s}^2+\left(\frac{c_i(s)}{|a_i(s)|}+\frac{c_j(s)}{|a_j(s)|}\right)\geq 0. \tag{4.17}$$

This can be thought of as a set of downward facing parabolas around zero path velocity. Equalizing the above expression with zero provides a set of possible path velocities, from which the minimum is taken as the bounding velocity. This gives the Maximum Velocity Curve from the acceleration constraints:

$$MVC_{acc}(s)=min\left\{\min_{\substack{i\in(1\ldots n_w)\\ j\in(i+1\ldots n_w)}}\sqrt{\frac{\frac{c_i}{|a_i(s)|}+\frac{c_j}{|a_j(s)|}}{\left|\frac{b_i(s)}{a_i(s)}-\frac{b_j(s)}{a_j(s)}\right|}},\min_{i\in(1\ldots n_w)}\sqrt{\frac{c_i(s)}{|a_j(s)|}}\right\}. \tag{4.18}$$

Now, the velocity constraints can be taken into account to determine the overall maximum velocity curve. From equation (4.12), the velocity condition can be written as:

$$-c_i(s)\leq a_i(s)\dot{s}\leq c_i(s)\Rightarrow MVC_{vel}(s)=\min_{i\in(1\ldots n_w)}\frac{c_i(s)}{|a_i(s)|}. \tag{4.19}$$

The overall maximum velocity curve is the minimum among the bounds obtained from both acceleration and velocity constraints is shown for the 2D example on Fig. 4.4.

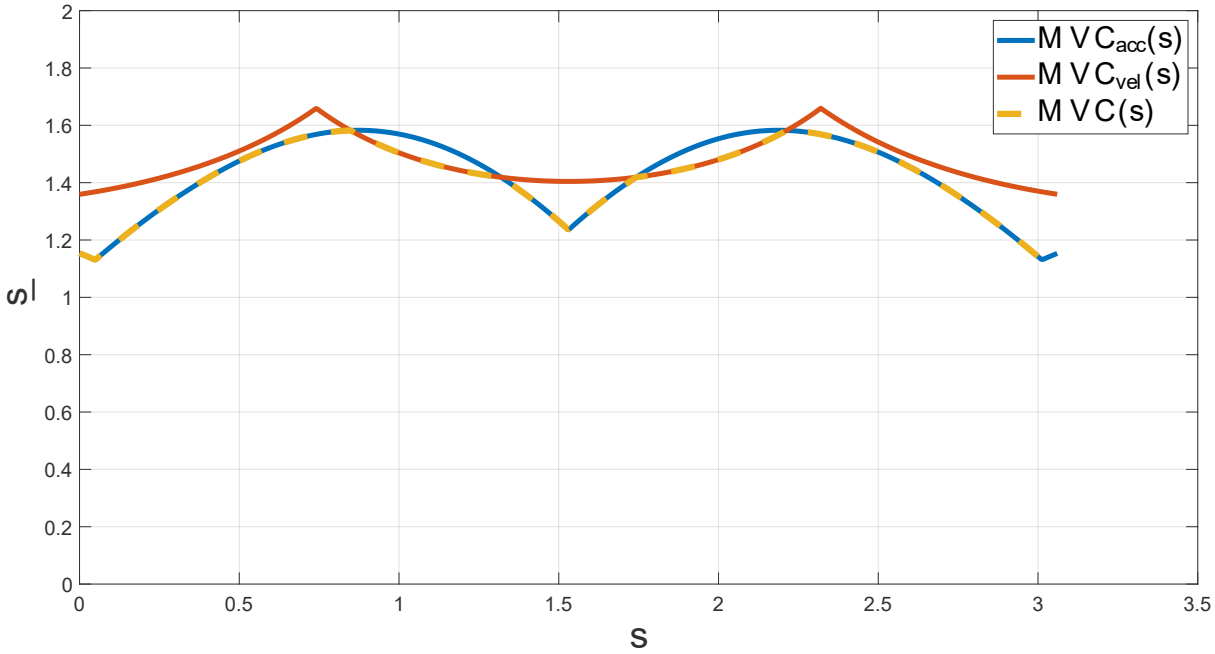


Figure 4.4: Maximum velocity curves obtained for the 2D example. The combined MVC is the minimum of both acceleration and velocity bounds. Each feasible trajectory must lie below the MVC. Violating this constraint renders the trajectory non feasible.

Computing the maximum velocity curve is the first step of the TOPP algorithm. The next step is obtaining the time optimal velocity profile, which is done through a bang-bang control. Namely, at least one degree of freedom is at its velocity or acceleration constraint at any given instance. This can be achieved by following the α and β profiles from equation (4.16), keeping in mind that a feasible trajectory lies below the maximum velocity curve and above $\dot{s} > 0$. Any trajectory violating these rules requires a motion that is not achievable by the actuators. The numerical integration procedure is quite extensive, with multiple singularity conditions that need to be addressed, which goes beyond the scope of this thesis. An interested reader is referred to [94, 130] for details about the numerical integration approach. However, to give the reader an idea about the inner workings of the TOPP algorithm, a simpler example is provided. A conservative trajectory can be achieved by restricting the path velocity to $\dot{s}_{max} < min(MVC(s))$. First, follow the maximum acceleration β profile until the $\dot{s}_{max}$ is reached. Then, simply keep the path velocity constant at $\dot{s}_{max}$ until an appropriate point is reached to follow the maximum deceleration α profile to reach the final configuration $s = s_{max}, \dot{s} = 0$. An example of such a trajectory is provided on Fig. 4.5. Equation (4.10) can be used to obtain the position, velocity and acceleration profiles in the time domain. Although relatively simple, such an approach can be effectively used in tasks that require a constant velocity motion, for example in agricultural surveying or wind turbine inspection.

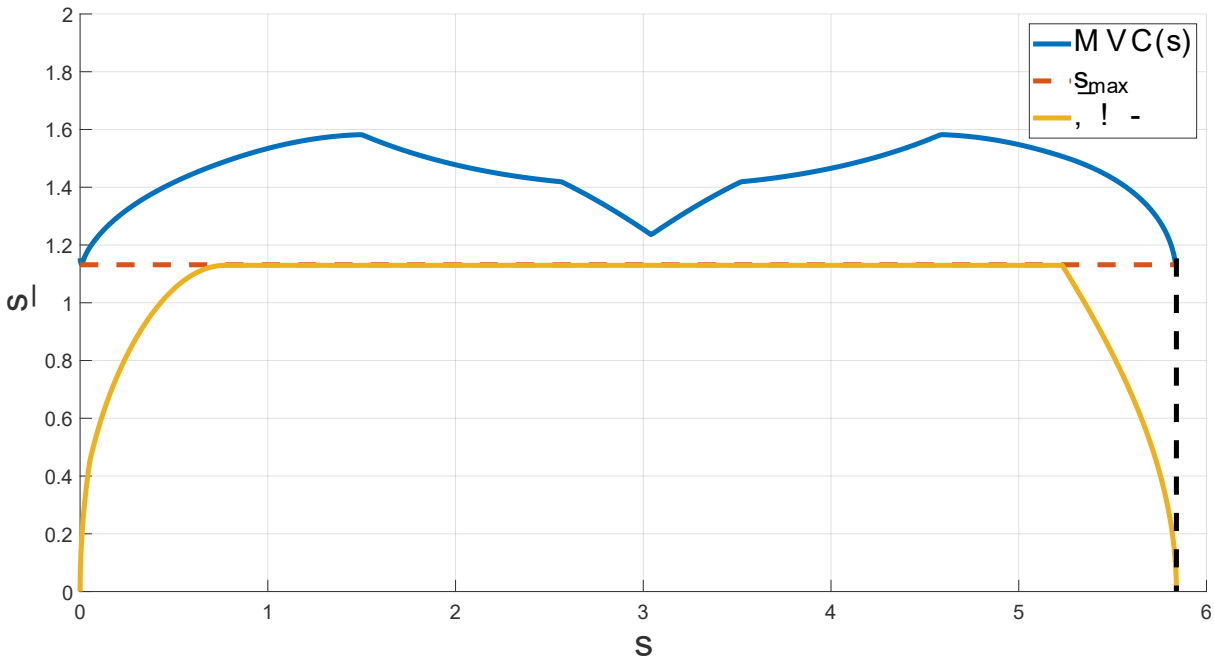


Figure 4.5: Time parameterized trajectory following the constant velocity. First, accelerate along the β profile until the constant velocity $\dot{s}$ is reached. Follow the line with $\ddot{s} = 0$ until an appropriate point for maximum deceleration along α profile is reached. Concatenate the three profiles into the final trajectory.

4.3.1.1 Note on Torque Constraints

The TOPP algorithm derivation in the previous section considered only acceleration and velocity constraints. In an arbitrary robotic use case, the torque constraints can be also considered. Therefore, a dynamical model of the system is required. Each degree of freedom can be treated as a second order system:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{q}}^T \mathbf{C}(\mathbf{q})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}, \tag{4.20}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n_w \times n_w}$ is a positive definite inertia matrix, $\mathbf{C}(\mathbf{q}) \in \mathbb{R}^{n_w \times n_w}$ captures centrifugal and Coriolis forces, $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^{n_w \times 1}$ is the gravity vector, and $\boldsymbol{\tau} \in \mathbb{R}^{n_w \times 1}$ is the actuator torque vector. Similar to the equation (4.12), each degree of freedom is considered to have a torque limit:

$$-\tau_i^{max} \leq \tau_i(t) \leq \tau_i^{max}. \tag{4.21}$$

Substituting the path velocity and acceleration from equation (4.10) into (4.20) yields:

$$-\boldsymbol{\tau}^{max} \leq \ddot{s}\mathbf{M}(\mathbf{q})\mathbf{q}_s + \dot{s}^2\mathbf{M}(\mathbf{q})\mathbf{q}_{ss} + \dot{s}^2\mathbf{q}_s^T\mathbf{C}(\mathbf{q})\mathbf{q}_s + \mathbf{g}(\mathbf{q}) \leq \boldsymbol{\tau}^{max}. \tag{4.22}$$

Recalling the expression (4.13), a similar substitution can take place:

$$\begin{aligned} \mathbf{a}(s) &= \mathbf{M}(\mathbf{q}(s))\mathbf{q}_s(s) \\ \mathbf{b}(s) &= \mathbf{M}(\mathbf{q}(s))\mathbf{q}_{ss}(s) + \mathbf{q}_s^T(s)\mathbf{C}(\mathbf{q}(s))\mathbf{q}_s(s) \\ \mathbf{c}(s) &= \mathbf{g}(\mathbf{q}(s)) - \boldsymbol{\tau}^{max}. \end{aligned} \tag{4.23}$$

Each row of the matrices above represents one inequality equation, and the TOPP procedure can be followed from equation (4.13) onwards, with bounds on path velocity and acceleration defined through the robot torques and dynamical properties. Note that the TOPP performed in the previous section is adequate for differentially flat systems, as it is possible to guarantee the planned trajectory feasibility based only on the velocity and acceleration constraints. The procedure drawn here considers more complex systems, where the connection between the torque limits and acceleration heavily depends on the current robot configuration.

CHAPTER 5

Unmanned Aerial Manipulator Model-based Motion Planning

As stated earlier, attaching a manipulator to a multirotor complicates the mathematical modeling and the overall system. The trade-off is making an aerial manipulator capable of interacting and possibly changing the environment. Naturally, this depends on the tool and task in place. One of the canonical tasks in robotics is peg-in-hole. This can have a multitude of applications, such as inspection, inserting parts during assembly, etc. If this task is extended to an aerial manipulator, the operation scope can be extended to places unreachable by a robotic arm. However, due to the underactuated nature of co-planar multirotors, the planned and executed end-effector motion can differ significantly, making this task hard to perform depending on the tolerated margin of error.

One option is to execute quasi-static maneuvers to keep the system close to hover conditions. This can increase execution times, putting at stake the mission objective. One way to cope with it is applying the end-effector corrections online. This requires having inverse kinematics solver in the loop and relying on the manipulator dynamics to suffice the requested motions. However, some manipulator configurations might not be reachable, or the motion cannot be executed in the required time. To alleviate these problems, this section proposes exploiting the dynamical model of the system in the motion planning procedure. After the trajectory is planned, it is executed by the model and full model state is recorded. Namely, since the multirotor is underactuated, the roll and pitch angles are omitted in the initial trajectory planning. When the trajectory is executed by the model, these angles are recorded, which is followed by employing the manipulator inverse kinematics to reach the desired end-effector configuration. After the corrections are applied, the collision checks can be performed, as well as checking the feasi-

bility in terms of dynamical requirements, determining whether the trajectory is feasible before executing it.

5.1 Model-based Motion Planning

The model-based motion planning framework consists of multiple steps. Given a start and end configuration, a series of waypoints is obtained with the RRT* path planner, as described in Chapter 4. The functional block diagram is depicted on Fig. 5.1. In this case, a waypoint consists of a 6-DoF multirotor configuration and a manipulator configuration with n_{M} DoF. A single high dimensional waypoint can be written as:

$$\mathbf{w} = \begin{bmatrix} x & y & z & \phi & \theta & \psi & q_1 & q_2 & \dots & q_{n_{\mathrm{M}}} \end{bmatrix}^T \tag{5.1}$$

where $\mathbf{w} \in \mathbb{R}^{6+n_{\mathrm{M}}}$. The RRT* requires limits for each DoF to bound the search space. The spatial constraints for the allowed multirotor positions are extracted from the environment map. The roll and pitch angles are constrained to empty space, which instructs the planner to exclude these states from the search. However, ϕ and θ are going to be obtained during the model motion. The yaw angle ψ is represented as a $SO(2)$ state, which instructs the planner to perform the search on a rotation matrix. Indeed, there are no bounds for the yaw angle and the multirotor is allowed to turn around a full circle. Finally, the bounds for the manipulator DoFs are set to its physical limits. This way of constraining the planner is particularly useful since joint limits are not violated in the planned path.

The input to the path planner is a set of $n_w > 2$ waypoints. The simplest example is providing only two waypoints as the start and goal. Depending on the specific task, multiple points of interest can be optionally added. The path planner produces an obstacle-free piecewise straight path by adding intermediate waypoints, as described in equation (4.5). Note that the piecewise straight refers to high dimensional lines connecting the waypoints, and the RRT* guarantees no collision along these lines. It can be observed that if there are no obstacles between the provided waypoints, the path planning algorithm simply returns the provided waypoints.

Based on the planned path, the TOPP-RA trajectory is generated according to Section 4.3, and can be written as:

$$\mathscr{T}_{\mathrm{D}} = \left\{ \mathbf{x}(kT_s) \mid \mathbf{x}(kT_s) \in \mathbb{R}^{3\cdot(6+n_{\mathrm{M}})}, k \in (0, \dots, n_t) \right\}, \tag{5.2}$$

where each trajectory point consists of position, velocity and acceleration of each joint. Since the roll and pitch angles are ignored in path planning, the trajectory interpolation is also ignoring them at this stage. The trajectory planned in this way is suitable for executing through the mathematical model to obtain the full aerial manipulator state. The sampling time T_s corresponds to

the discretized model sampling time. As shown on Fig. 5.1, the path and trajectory planning is terminated after n_p trials. This approach is required since the planned trajectory deviates from the underlying path, and it is possible for the trajectory to fail the collision checking. After the user specified number of attempts, the planning is considered infeasible.

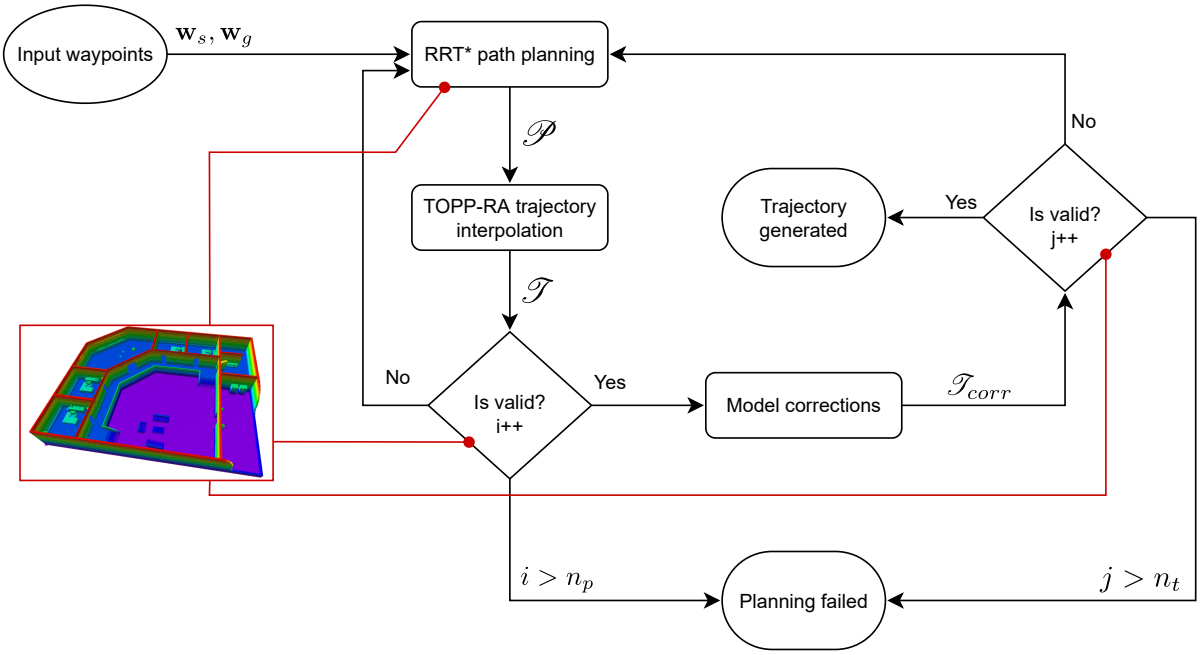


Figure 5.1: A functional block diagram describing the model-based trajectory planning. The input is considered to be the start point $\mathbf{w}_s$, which is usually the aerial manipulator current configuration, and the goal point $\mathbf{w}_g$. First part is the path and trajectory planning, and ensuring the initial trajectory is not in collision with the environment. This is followed by employing model corrections to the end-effector configuration which produces the final trajectory that is also checked for collisions. In case the initial trajectory is not valid for n_p times, the planning procedure is terminated. The same goes for the final trajectory with n_t trials.

5.1.1 Model-in-the-loop

The first step towards applying the model corrections is simulating the trajectory planned in equation (5.2). This requires implementing the mathematical model and control described in Chapter 3. In this thesis, the model is implemented in *Gazebo* [131] simulator within the Robotic Operating System (ROS) [132] middleware.

5.1.1.1 Aerial Manipulator Gazebo Model

The aforementioned Gazebo simulator offers a real-time physics and rendering engine suitable for a robotic simulation. It is modular since it supports custom sensor and actuator integration through plugins. Within this section, the model is explained using a simple example consisting of a quadcopter with a lightweight manipulator attached. The quadcopter is modelled as a single rigid body with four actuators placed at the end of each arm. Each actuator is considered to be

a motor with a propeller. To simulate the rotor dynamics, the RotorS [133] propeller plugin is employed. Based on the motor and propeller parameters, it calculates the force and torque produced by the actuator, which makes it ideal for the quadcopter modelling. The attitude and position cascade PID control is implemented within the ROS middleware to control the vehicle. Naturally, any control requires feedback, and in this case it is supplied through sensor plugins. For angles and angular rates, the Hector quadrotor package [134] is employed. Among a multitude of sensors, it offers an Inertial Measurement Unit (IMU) capable of providing the required feedback for the attitude controller. The IMU sensor in Gazebo is attached to the multirotor center of mass. For the position and velocity feedback, an odometry plugin from the RotorS package is used. It supports the noise and bias parameters of the sensor, yielding realistic feedback signals.

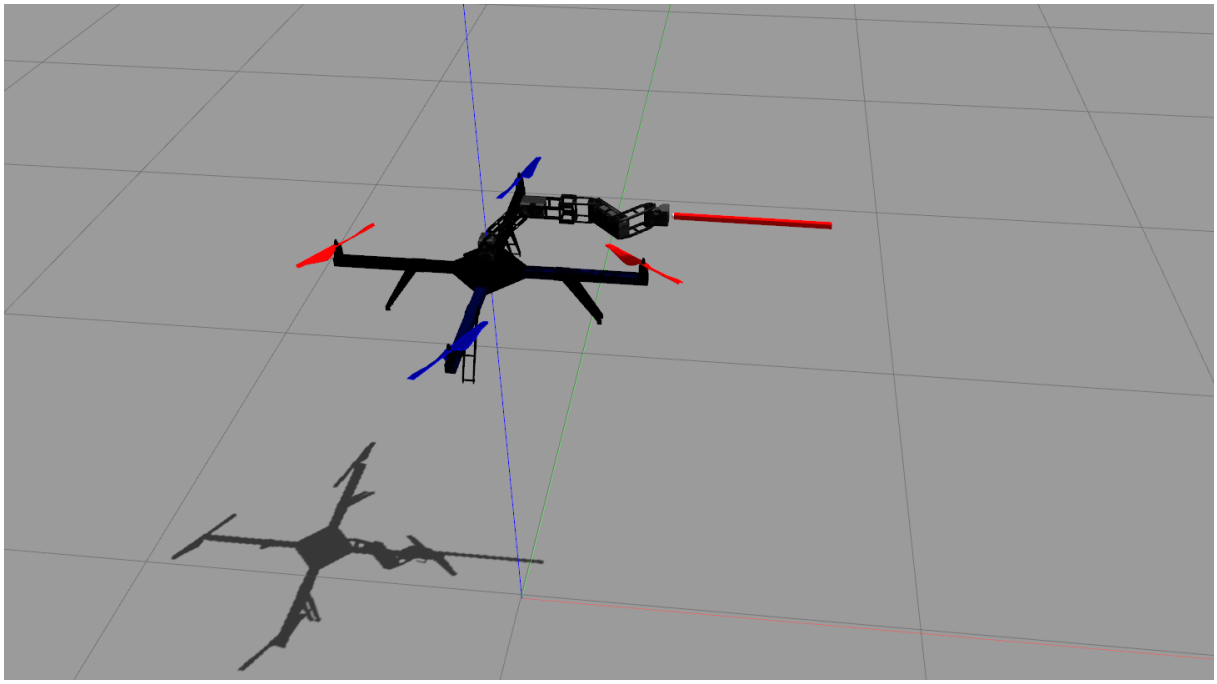


Figure 5.2: A snapshot from the Gazebo simulator showing an aerial manipulator in flight. It consists of a quadrotor body endowed with a 5-DoF serial chain manipulator and a rod-like tool.

The manipulator is then attached to the multirotor body with some transformation $\mathbf{T}_{\mathrm{B}}^{0}$. The Gazebo simulation environment supports both rotational and prismatic joints. In this case, the manipulator is built of successive joint-link pairs, where only rotational joints are used, with links' dimensions set the same as their real-world counterparts. The tool is added as the final link, which yields a standard serial manipulator chain. Depending on the mission, different sensors can be attached at the end-effector, i.e. a force sensor or a camera. The Gazebo environment also features a fixed joint that has no capability of actuation, used to rigidly attach sensors to the end-effector. Each joint has separate cascade PID controller structure with inner loop controlling the angular velocity and outer loop controlling the angle. These controllers are based on *JointVelocityController* and *JointPositionController* integrated in ROS through the

ros_control package [135].

The described controller structure successfully separates multirotor and manipulator control, allowing the references to be set independently. This is in accordance with the motion planning, as a high dimensional space is considered during the trajectory planning.

5.1.1.2 Applying Model Corrections

Having the model implemented, the next step is to apply the corrections derived from the model. To do so, the initially planned trajectory from equation (5.2) needs to be executed. In a quasi-static motion, where the dynamic constraints are small, the planned and executed trajectory may not differ a lot. Depending on the task, this can be a satisfactory behavior. However, it inevitably leads to conservative plans that extend the mission duration. Since the multirotor is an underactuated system, the roll and pitch angles are omitted in the path and trajectory planning. Therefore, while executing the trajectory through the model, these DoFs are recorded to obtain the full state of the aerial manipulator. This yields the full state executed trajectory:

$$\mathscr{T}_{\mathrm{E}} = \left\{ \mathbf{x}(kT_s) \mid \mathbf{x}(kT_s) \in \mathbb{R}^{3\cdot(6+n_{\mathrm{M}})}, k \in (0,\ldots,n_t) \right\}. \tag{5.3}$$

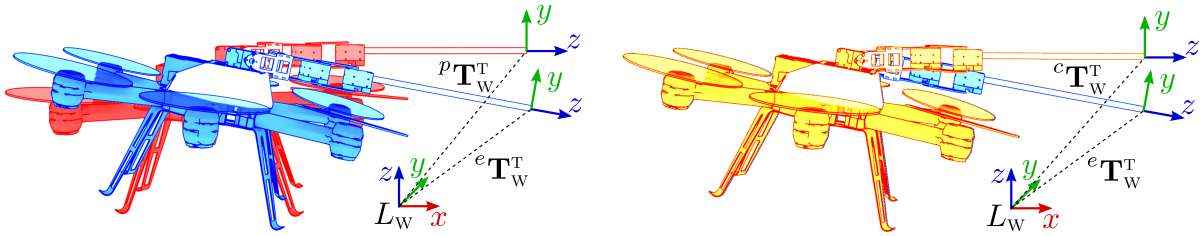
Note that the number of discretized points is the same in the initial and executed trajectories.

The planned end-effector configuration can be denoted as ${}^{p}\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$, which represents a homogeneous transform of the end-effector in the world frame. In terms of the planning state vector $\mathbf{x}$, this transform can be calculated through direct kinematics of the manipulator, derived in Section 3.1.1. After executing the trajectory, the end-effector configuration ${}^{e}\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$ deviates from the planned one, shown on Fig. 5.3a. Utilizing the aerial manipulator null space, this deviation can be eliminated. Recall equation (3.4) that describes the aerial manipulator kinematic chain $\mathbf{T}_{\mathrm{W}}^{\mathrm{T}} = \mathbf{T}_{\mathrm{W}}^{\mathrm{B}} \cdot \mathbf{T}_{\mathrm{B}}^{0} \cdot \mathbf{T}_{0}^{\mathrm{T}}$. This can be rearranged as:

$${}^{c}\mathbf{T}_{0}^{\mathrm{T}} = (\mathbf{T}_{\mathrm{B}}^{0})^{-1} \cdot ({}^{e}\mathbf{T}_{\mathrm{W}}^{\mathrm{B}})^{-1} \cdot {}^{p}\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}, \tag{5.4}$$

where ${}^{c}\mathbf{T}_{0}^{\mathrm{T}}$ is the end-effector transform expressed in the manipulator base coordinate system, in other words, the correction transform. This step is depicted on Fig. 5.3b where the manipulator null space is used to correct the end-effector configuration. Note that $\mathbf{T}_{\mathrm{W}}^{\mathrm{B}}$ can be also obtained in the path planning step, however, since the roll and pitch angles are constrained to zero space this does not yield the full multirotor state. Therefore, the transform ${}^{e}\mathbf{T}_{\mathrm{W}}^{\mathrm{B}}$ is constructed using the executed trajectory.

Finally, obtaining the correction transform ${}^{c}\mathbf{T}_{0}^{\mathrm{T}}$ leads to using the inverse kinematics of the manipulator to compute new joint values, ones that correct the end-effector to the planned configuration. In this thesis, the *MoveIt!* [104] inverse kinematics solver is employed. It is based on the numerical inverse kinematics solver *IKFast* developed within the *OpenRAVE* planning



(a) Difference between the planned (red) and executed (blue) end-effector configuration, without performing dynamic model corrections. ©2020 IEEE. Reprinted with permission from [112].

(b) Difference between states in trajectory with (yellow) and without (blue) corrections. The manipulator null space is exploited for applying end-effector corrections. ©2020 IEEE. Reprinted with permission from [112].

Figure 5.3: An example of the manipulator end-effector configuration difference between planned and executed trajectory. The figure illustrates a snapshot in time while executing a trajectory with significant pitch angle. The underlying idea is correcting the error between the planned end-effector configuration ${}^{p}\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$ and the executed end-effector configuration ${}^{e}\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$ using the null space of the manipulator. Therefore, the planned ${}^{p}\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$ and corrected ${}^{c}\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$ configurations are identical.

tool [126]. The solver offers a simple interface and supports both exact and approximate inverse kinematics solutions. Depending on the manipulator configuration, sometimes the exact solution does not exist. In these cases, an approximate solution is used. Corrected joint values are computed for each discrete point within the executed trajectory from equation (5.3), yielding the corrected trajectory:

$$\mathscr{T}_{\mathrm{C}} = \left\{ \mathbf{x}(kT_s) \mid \mathbf{x}(kT_s) \in \mathbb{R}^{3\cdot(6+n_{\mathrm{M}})}, k \in (0,\ldots,n_t) \right\}. \tag{5.5}$$

Finally, the trajectory in this form can be sent to the system to be executed. This kind of planned can be employed in tasks such as wall contact or insertion. The main benefit of using the model based planning framework is keeping a steady end-effector orientation during the approach.

5.2 Results

Within this section, the model-based motion planning method is tested in both simulation and laboratory environment. Multiple tests are performed in the simulation to obtain the relationship between the dynamical constraints and the end-effector deviation. This forms a solid ground to choose the dynamical constraints for simulation in a factory environment, performing a pipe insertion. The experimental analysis is performed in a laboratory environment, inserting a rod into a transparent tube.

5.2.1 Simulation

The simulation is conducted in the Gazebo environment with an aerial manipulator modelled according to Section 5.1.1.1. A lightweight manipulator with 5 DoFs is attached to the multirotor

body, above the center of mass:

$$\mathbf{T}_{\mathrm{B}}^{0} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0.075 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.6}$$

The manipulator consists only of rotational joints connected with links. The end-effector is a long rod suitable for insertion tasks, and also exaggerates the deviation from the planned trajectory, making it more straight forward to benchmark the planning method. The manipulator DH parameters are given in Table 5.1.

Table 5.1: DH parameters of the 5-DoF manipulator attached to the multirotor in the simulation. Note that last joint 5* is virtual and is required to properly align the end-effector coordinate system.

	θ_k	d_k	α_k	a_k
1	0	0	0	$0.1225m$
2	0	0	$-\pi/2$	$0.1365m$
3	0	0	0	$0.0755m$
4	0	0	0	$0.0725m$
5	$\pi/2$	0	$\pi/2$	0
5*	0	$0.35m$	0	0

5.2.1.1 End-effector Deviation

The goal of the first set of simulations is twofold: determining the difference between the planned and executed end-effector trajectories, and getting insight how the dynamical constraints influence corrections. This is done by planning a straight line trajectory along the x axis, exerting only the multirotor pitch during motion. In such a setup, the end-effector has the most significant deviation in the $x-z$ plane, allowing for a relatively simple performance measurement. The trajectory length along the x_{W} axis is kept at $5m$ for all trials, and a typical trial is shown on Fig. 5.4. One can observe the forward motion of the end-effector on top portion of the figure. The end-effector z reference is constant, however, the response deviates from the planned value. The deviations are the greatest at the start and end. This is expected since the acceleration is proportional to the pitch angle, and the maximum acceleration is achieved during the start and stop. The middle of the response is relatively still, without significant deviation. Furthermore, one can observe the reduced deviation for the corrected trajectory $\mathscr{T}_{\mathrm{C}}$, which is the intended behavior of the model based planning method.

To determine how the deviation depends on the dynamical constraints, multiple simulations were conducted. Since the multirotor body motion is planned only along the x axis, the follow-

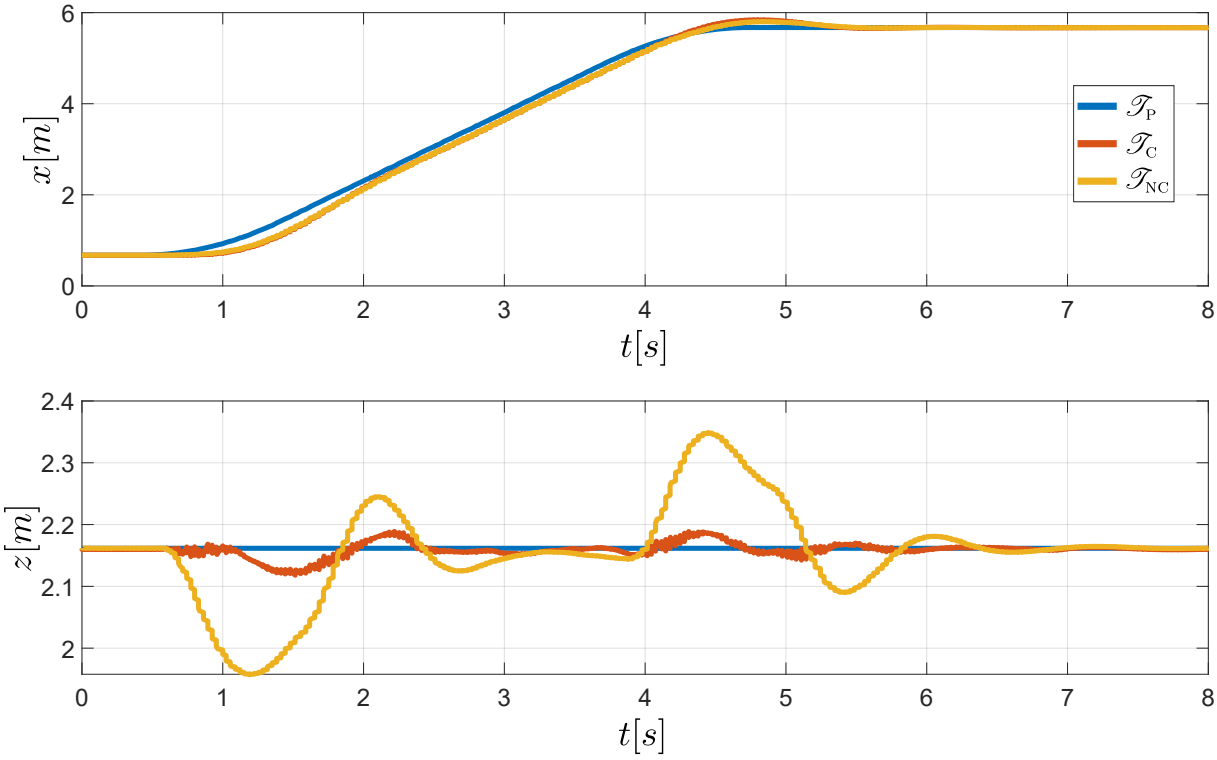


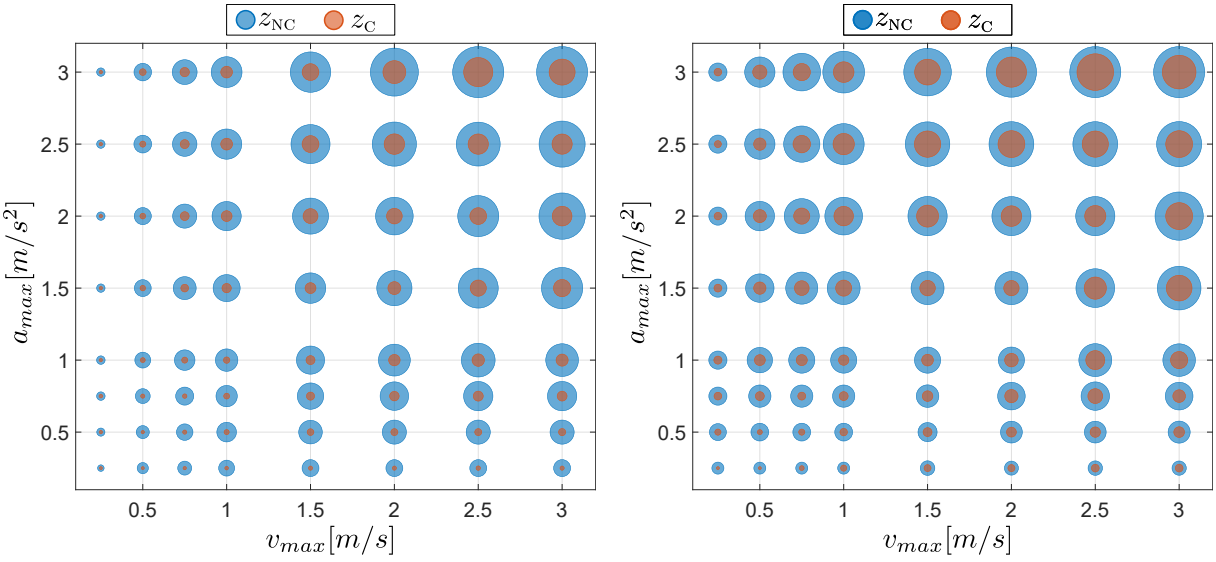
Figure 5.4: A representative response of simulation trials conducted with the straight line planned trajectory. The planned end-effector trajectory is denoted with $\mathscr{T}_\mathrm{P}$. The executed trajectory with the corrections is $\mathscr{T}_\mathrm{C}$. Trajectory $\mathscr{T}_\mathrm{NC}$ shows the end-effector motion without corrections. In the depicted trial, the velocity and acceleration constraints for the x axis are set to $v_{max} = 1.5m/s$ and $a_{max} = 1.5m/s^2$.

ing constraints are used for the velocity and acceleration:

$$\begin{aligned} v_{max} &\in \{0.25, 0.5, 0.75, 1.0, 1.5, 2.0, 2.5, 3.0\}\, m/s \\ a_{max} &\in \{0.25, 0.5, 0.75, 1.0, 1.5, 2.0, 2.5, 3.0\}\, m/s^2. \end{aligned} \tag{5.7}$$

During all trials, the maximum angular velocity of all joints is set to $\omega_{max} = 1.2rad/s$, and the maximum angular acceleration is set to $\alpha_{max} = 1.2rad/s^2$. A total of 10 simulation trials are conducted for each velocity-acceleration constraint pair. The first measured value is the mean end-effector deviation along the z axis, while the second is the maximum deviation. In both cases, the end-effector deviation is compared with and without corrections, with the results shown on Fig. 5.5. It can be observed that increasing any dynamical constraint results in a larger end-effector deviation. This is the consequence of a larger pitch angle required to follow dynamically challenging trajectories. In all trials, the corrected trajectory deviation is lower than the non-corrected, proving the model based planning method effectiveness. Note that above the maximum acceleration $a_{max} > 2.0m/s^2$ the multirotor body motion dominates the error. Such a high acceleration constraint allows for significant pitch angles, which cause a drop in multirotor z axis position. This is further reflected on the end-effector position, yielding higher mean and

maximum errors.



(a) Mean end-effector deviation along z axis.

(b) Maximum end-effector deviation along z axis.

Figure 5.5: The end-effector mean and maximum deviation along the z axis. Blue circles represent the executed trajectory without corrections $\mathscr{T}_{\text{NC}}$, while the red circles represent the executed trajectory with applied correction $\mathscr{T}_{\text{C}}$. The circle radii are the mean and maximum deviation of the end-effector z axis expressed in meters $[m]$.

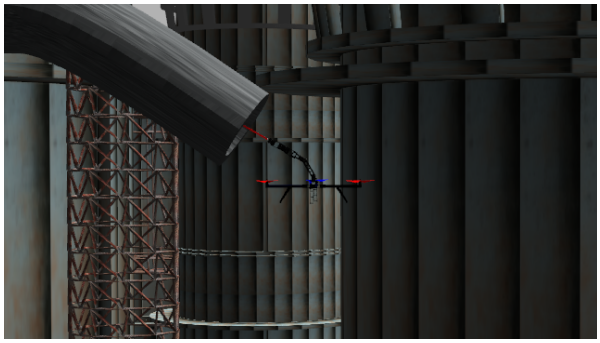
5.2.1.2 Complex Environment Planning

The second set of simulations focuses on the RRT* path planning in complex environments, followed by the model-based trajectory planning. The aerial manipulator is tasked to perform pipe insertion, which is one version of the canonical peg-in-hole task. To that end, a simple factory environment with three pipes of interest is used. The first pipe has radius $r_{p1} = 0.21m$ and is inclined at an angle $\alpha_{p1} = 60^\circ$ relative to the $x-y$ plane. The second pipe with radius $r_{p2} = 0.14m$ is placed horizontally, while the third pipe poses the most significant challenge with radius $r_{p3} = 0.04m$ and is also placed horizontally. The environment, together with insertion for each pipe, is depicted on Fig. 5.6.

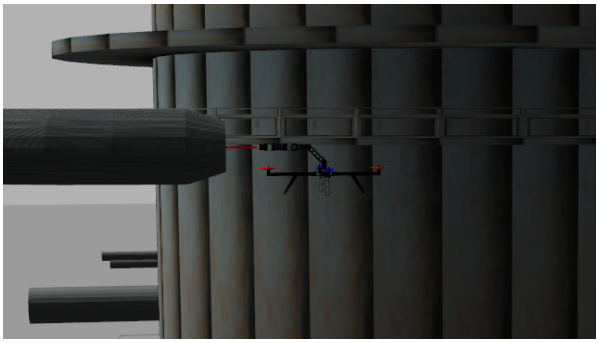
Taking into account the results obtained in Section 5.2.1.1, the chosen dynamical constraints are given within Table 5.2. Although setting lower velocity and acceleration constraints ensures smaller end-effector deviations, the constraints are relatively high to conform to the environment

Table 5.2: Trajectory planning constraints set for each DoF in the experimental verification. The units for linear axes x, y and z are expressed in m/s for velocity and m/s^2 for acceleration. The angular DoFs' units are expressed in rad/s for velocity and rad/s^2 for acceleration.

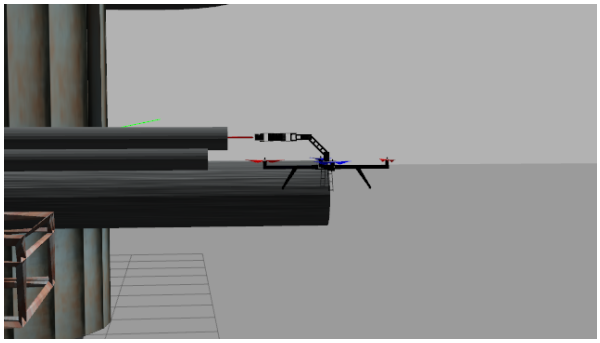
	x	y	z	ψ	q_1	q_2	q_3	q_4	q_5
v_{max}	1.5	1.5	0.5	0.5	1.2	1.2	1.2	1.2	1.2
a_{max}	1.0	1.0	0.5	0.5	1.2	1.2	1.2	1.2	1.2



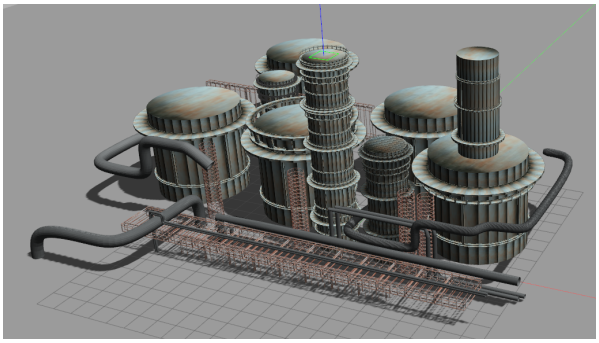
(a) The first pipe inclined insertion.



(b) The second pipe horizontal insertion.



(c) The third pipe horizontal insertion with challenging pipe radius.



(d) The factory environment spawned in the Gazebo simulator.

Figure 5.6: The aerial manipulator performing the pipe insertion task, together with the full perspective of the factory environment used in the Gazebo simulator.

size. This way, the total trajectory time is reduced, while ensuring a successful insertion even in the most challenging case. To use the RRT* path planning, the environment is first decomposed into an OctoMap. Several planned trajectories for each pipe are shown on Fig. 5.7, together with the OctoMap environment. For each pipe there is a single starting point $\mathbf{w}_s$ and goal point $\mathbf{w}_g$. Planned path and trajectories differ between trials because of the random sampling performed by the RRT* algorithm.

Five trials have been performed for each pipe, yielding a different trajectory each time. The aerial manipulator has performed a successful insertion in each of these trials. Granted, the first and second pipes have a rather large radius. Nevertheless, trials on these pipes produced valuable insights and helped in tuning the planner parameters. One observed phenomenon is the occasional increase in error of the corrected trajectory, surpassing the non-corrected values. This occurs when the planned joint values drive the end-effector towards the workspace limit. In such cases, the inverse kinematics cannot find a solution that effectively minimizes the end-effector error.

5.2.2 Experimental Verification

For the experimental verification, the AscTec NEO hexacopter is employed. It is equipped with a compact Intel NUC onboard computer running Linux Ubuntu 14.04 and ROS Indigo

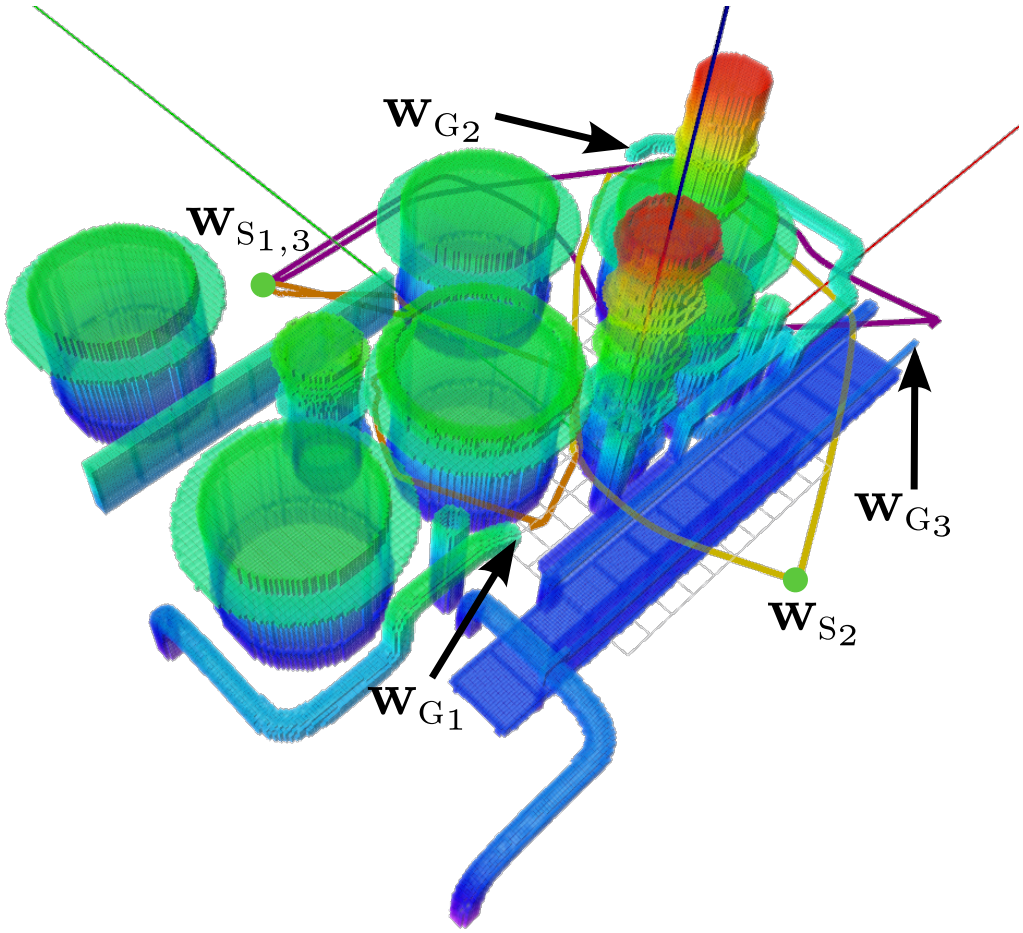
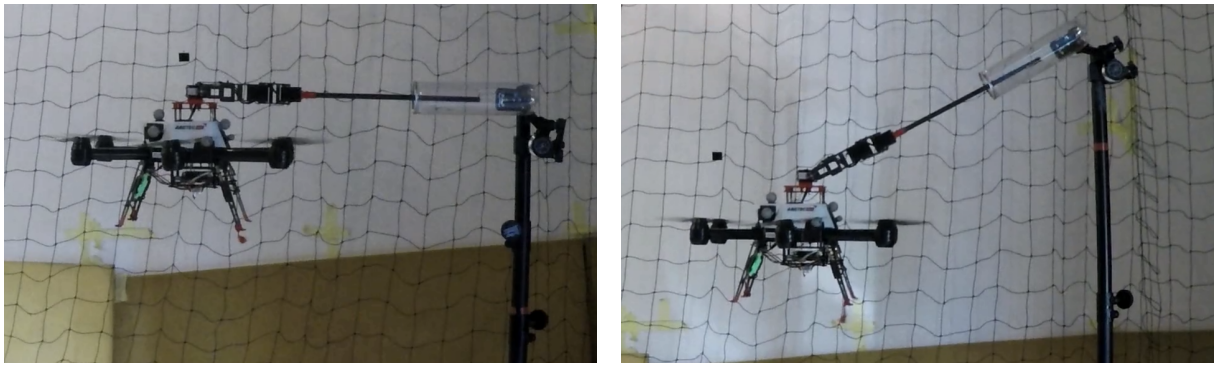


Figure 5.7: OctoMap visualization of the factory environment used for planning. The start points $\mathbf{w}_S$ are chosen to force the RRT* algorithm planning through the environment. The goal points $\mathbf{w}_G$ correspond to each pipe. Note that the depicted trajectories show the multirotor body position.

middleware. The low-level attitude controller is tripple redundant AscTec Trinity, connected to the onboard computer with serial communication. The maximum payload of the UAV is $2kg$ with up to $26min$ flight time, which is enough to execute various missions while safely carrying a lightweight manipulator. A high-level model predictive controller is employed for position control, implemented within the ROS middleware [43]. The high-level controller inputs are positions in x, y, and z axes, as well as the yaw angle ψ, with support for the velocity feed forward. The output consists of roll, pitch, yaw rate, and thrust and is sent to the low-level controller. The overall system performing experiments is depicted on Fig. 5.8.

A 3-DoF serial chain manipulator is attached to the top plate of the UAV. It consists of custom built carbon fiber links to reduce weight, and a carbon fiber rod tool that extends beyond the UAV body. The joints are actuated with Dynamixel XM430-W350R servo motors, connected to the onboard computer with a dedicated Universal Serial Bus (USB) interface. The DH parameters of the manipulator are given in Table 5.3.

The high-level controller feedback is provided through the Optitrack motion capture system. The system consists of seven Prime 13 operating at 100 frames per second. It offers



(a) Horizontal tube insertion.

(b) Inclined tube insertion.

Figure 5.8: The AscTec NEO hexacopter endowed with a 3-DoF manipulator performing tube insertion experiments. The chosen tube is transparent to easily determine the insertion success.

Table 5.3: DH parameters of the 3-DoF manipulator attached to the AscTec NEO UAV. Note that there is a virtual joint 3* required to properly align the end-effector coordinate system.

	θ_k	d_k	α_k	a_k
1	$\pi/2$	0	$3\pi/2$	$0.1365m$
2	0	0	0	$0.0725m$
3	$3\pi/2$	0	$3\pi/2$	0
3*	0	$0.4m$	0	0

high precision position and orientation feedback, however, it does not provide the velocity measurements. To do so, an Extended Kalman Filter (EKF) is employed as described in [136], by fusing together the IMU and Optitrack provided feedback. Both position and velocity feedback are required by the high-level model predictive controller.

To test the model-based motion planning method, the goal of experiments was to insert the rod end-effector into a tube. The tube itself is transparent in order to see if the insertion has been performed successfully. The rod length is $40cm$, which reflects to the last joint parameters in DH Table 5.3. The choice of such a relatively long rod is twofold. First, this allows the end-effector to go beyond the body of the UAV and safely perform its task without propellers colliding with the environment. Second, the errors in joint positioning are exaggerated at the end-effector. Imposing a task constraint in terms of the tube $7cm$ diameter presents the overall effectiveness of the model-based motion planning even further. Two sets of experiments were conducted. In the first set, an insertion repeatability has been tested with the tube placed horizontally (Fig. 5.8a), and with the tube inclined for $\alpha_t = 30^\circ$ (Fig. 5.8b). In the second set, an obstacle is introduced in the environment requiring the aerial manipulator to circumnavigate it and perform the insertion task. The common planning parameters for each task are the maximum velocity and acceleration of each DoF, as given in Table 5.4. Although the NEO hexarotor actual maximum dynamical constraints are much higher, these had to be decreased due to the experimental arena limited space.

Table 5.4: Trajectory planning constraints set for each DoF in the experimental verification. The units for linear axes x, y and z are expressed in m/s for velocity and m/s^2 for acceleration. The angular DoFs' units are expressed in rad/s for velocity and rad/s^2 for acceleration.

	x	y	z	ψ	q_1	q_2	q_3
v_{max}	0.8	0.8	0.4	0.5	1.2	1.2	1.2
a_{max}	0.8	0.8	0.4	0.5	1.2	1.2	1.2

5.2.2.1 Insertion Repeatability

In the first set of experiments, the system is tasked to perform a peg-in-hole experiment. Each experiment consists of the approach and insertion phase. A typical response of the horizontal insertion task is depicted on Fig. 5.9. As formerly mentioned, the insertion tube diameter is $7cm$, which is indicated with two black lines in the z axis response. Due to the forward motion of the multirotor, the end-effector deviation is most pronounced when the pitch angle is at its largest. This happens at the start of the approach as the system accelerates towards the tube, as well as at the end of the trajectory when the insertion is performed. Therefore, it is crucial to correct the end-effector deviation to successfully perform the insertion. To test the overall repeatability of the system, the peg-in-hole insertion task is performed over multiple trials. For

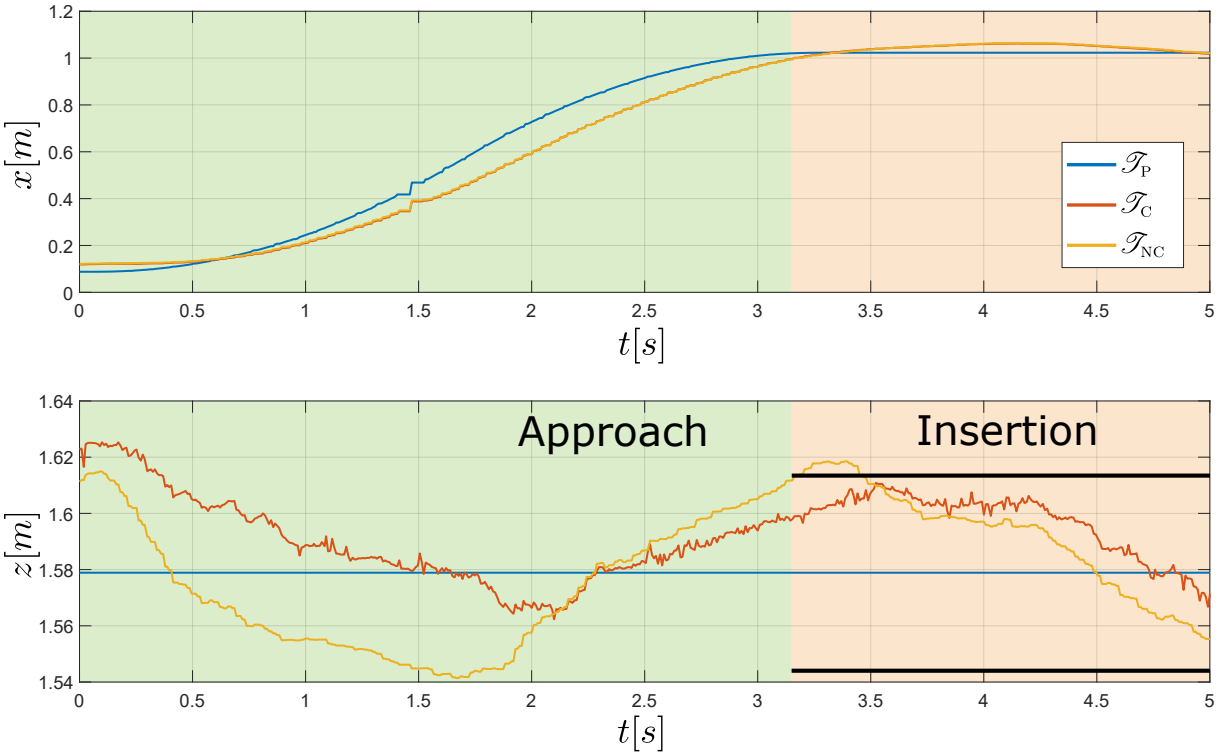


Figure 5.9: A representative response of experiments conducted with the straight approach. The insertion tube is shown for z axis in black lines. The planned end-effector trajectory is denoted with $\mathcal{T}_P$. The executed trajectory with the corrections is $\mathcal{T}_C$. Trajectory $\mathcal{T}_{NC}$ combines the multirotor's executed trajectory with static manipulator joint values, approximating the end-effector motion without applied corrections.

the horizontal insertion, the tube has been placed in front of the aerial manipulator, at a known position. Going from a start point, the aerial manipulator managed to perform a successful insertion in $11/15$ trials.

Apart from the horizontal insertion, the system was also tasked with an inclined tube insertion. In this experiment, the tube angle is set to $\alpha_t = 30^\circ$ measured around the world y axis. To achieve a straight line approach, an intermediate point is added along the line defined by the inclination angle and tube center. A representative response is depicted in $x-z$ plane on Fig. 5.10. As the end-effector is nearing the tube, the planned trajectory becomes a straight line, steering the end-effector towards the tube. Overall, the aerial manipulator performed a successful insertion in $8/10$ trials.

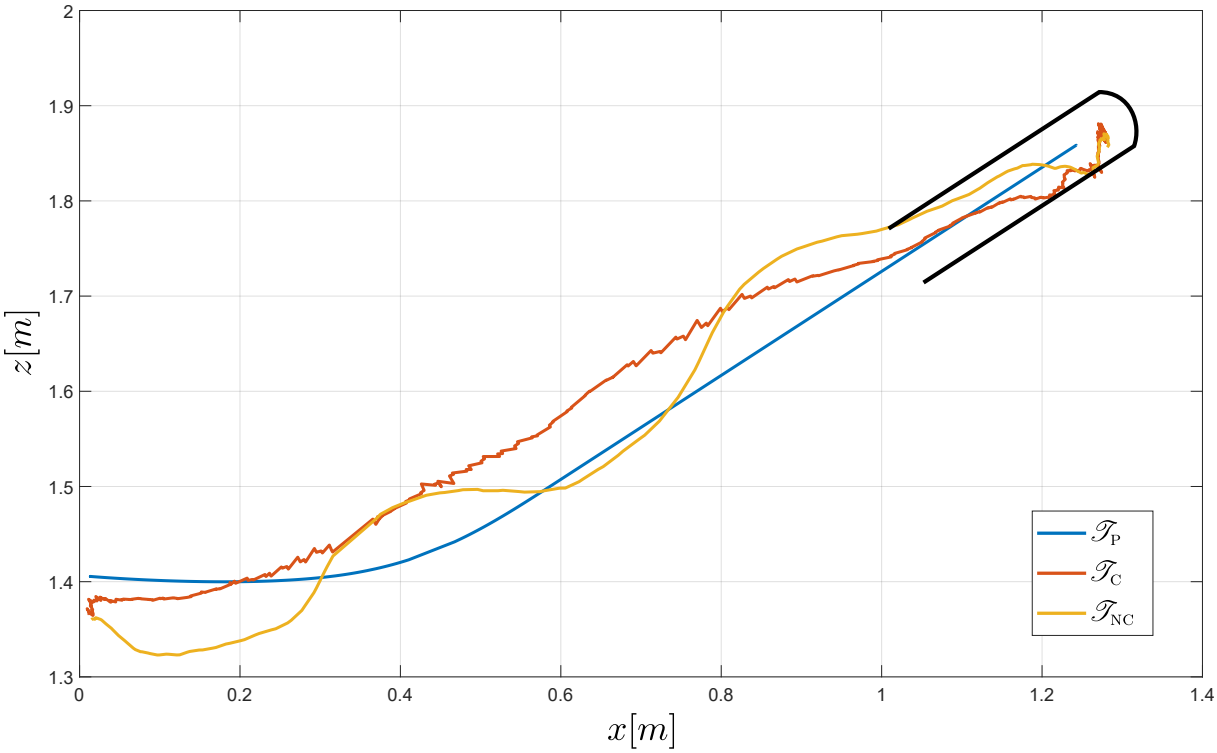


Figure 5.10: A representative response in $x-z$ plane of experiments conducted with the tube inclined for the angle $\alpha_t = 30^\circ$. The tube is depicted as a black outline. The planned end-effector trajectory is denoted with $\mathscr{T}_P$. The executed trajectory with the corrections is $\mathscr{T}_C$, and $\mathscr{T}_{NC}$ represents the end-effector trajectory without corrections.

5.2.2.2 Obstacle Avoidance

The former experiments tested the aerial manipulator insertion repeatability in an obstacle-free environment. In this set of experiments, an obstacle is placed in the middle of the arena, and the aerial manipulator is tasked to perform peg-in-hole insertion. The end-effector trajectory for obstacle avoidance is shown on Fig. 5.11. It can be observed that the aerial manipulator moves around the obstacle and successfully performs the insertion task. The largest deviations between

the planned and executed trajectories occur at the start and the end of the trajectory, which corresponds to largest acceleration. This is clearly visible on the figure, where the insertion task would not be successful without applying corrections.

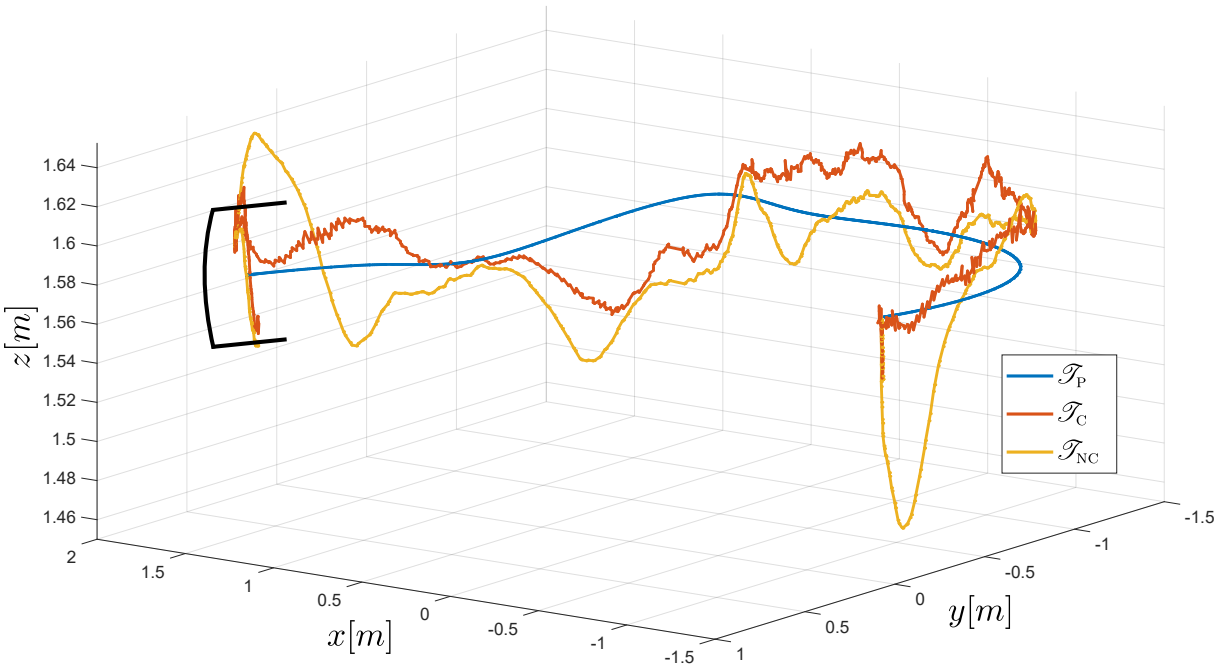


Figure 5.11: A 3D plot of the end-effector planned trajectory $\mathscr{T}_{\mathrm{P}}$, corrected trajectory $\mathscr{T}_{\mathrm{C}}$, and trajectory without corrections $\mathscr{T}_{\mathrm{NC}}$. The tube is is outlined in black. Note that the tube shape is distorted due to the z axis scale.

5.3 Wall Contact Planning

As discussed in previous sections, one of the possible applications for the model-based planning are insertion tasks. However, aerial manipulators are suitable for even wider range of application. One example of such tasks is related to contact with the environment. This includes tasks such as contact based inspection, gluing on a vertical surface, drilling, etc. Employing an aerial manipulator enables performing these tasks in hard to reach places, instead of using some kind of specialized machinery or sending humans after carefully considering safety. A stable contact with the environment can be achieved and maintained through an adaptive impedance control. It augments the classical cascade position controller described in Section 3.2 with force measurements. This inevitably complicates the system, as a force sensor or force estimation is required for properly employing the controller. In turn, it opens up a multitude of possible applications, rendering this trade-off acceptable.

Within this section, the impedance control is first derived. As it is not in the focus of this thesis, only a relatively simple overview is given to familiarize the reader with the concept. In short, the impedance is a pre-filter on the system reference that follows the desired second order

system dynamics, effectively forming new system dynamics. After introducing the impedance control, a set of simulations is conducted by performing a wall contact. The underlying model-based planning method is not conceptually changed. Rather, as the aerial manipulator control is augmented, the impedance controller is taken into account while executing the model trajectory.

5.3.1 Position-based Impedance

The main idea of the position-based impedance is simultaneous control of both position and force, only by commanding the end-effector position. During a contact with a stiff surface, the measured end-effector position remains static. The concept lies in modifying the reference to "penetrate" the stiff surface in order to exert the desired force. The underlying assumption is that an aerial manipulator is equipped with a force sensor providing accurate force measurements. Furthermore, the Cartesian impedance is used where all force references and measurements are expressed in a common coordinate system, namely, in the world frame L_{W}. Setting a force reference in the world frame is relatively straightforward, assuming a known contact point. On the other hand, force measurements are obtained in the local force sensor coordinate system L_{FS}. Since the force sensor is typically mounted at the end-effector or manipulator base, the direct kinematics described in Section 3.1.1 are employed. This yields the measured force in the world frame as:

$$\mathbf{f}_{\mathrm{W}} = \mathbf{R}_{\mathrm{W}}^{\mathrm{B}} \cdot \mathbf{R}_{\mathrm{B}}^{0} \cdot \mathbf{R}_{0}^{\mathrm{FS}} \cdot \mathbf{f}_{\mathrm{FS}} \tag{5.8}$$

where $\mathbf{f}_{\mathrm{FS}}$ is the force measured in the sensor local coordinate system, and $\mathbf{R}_{0}^{\mathrm{FS}}$ is the rotation matrix between the manipulator base and force sensor, which can be obtained through the manipulator direct kinematics. The full system schematic with adaptive impedance can be observed on Fig. 5.12.

Throughout this derivation, a single degree of freedom is considered for simplicity, and the equations are the same for other degrees of freedom. First, the impedance filter itself is derived, followed by the adaptation laws. The interaction with the environment is modelled as a linear spring:

$$f_m(t) = k_e \cdot (x_m(t) - x_e(t)), \tag{5.9}$$

where $f_m(t)$ is the measured force, $x_m(t)$ is the measured position, $x_e(t)$ is the environment position, and k_e is the spring stiffness. The environment position $x_e(t)$ is considered to be either a-priori known or obtained through some form of visual or point cloud detection.

The main idea of the impedance filter is to perturb the end-effector position reference to produce the desired contact force. The impedance filter command is considered to have a second

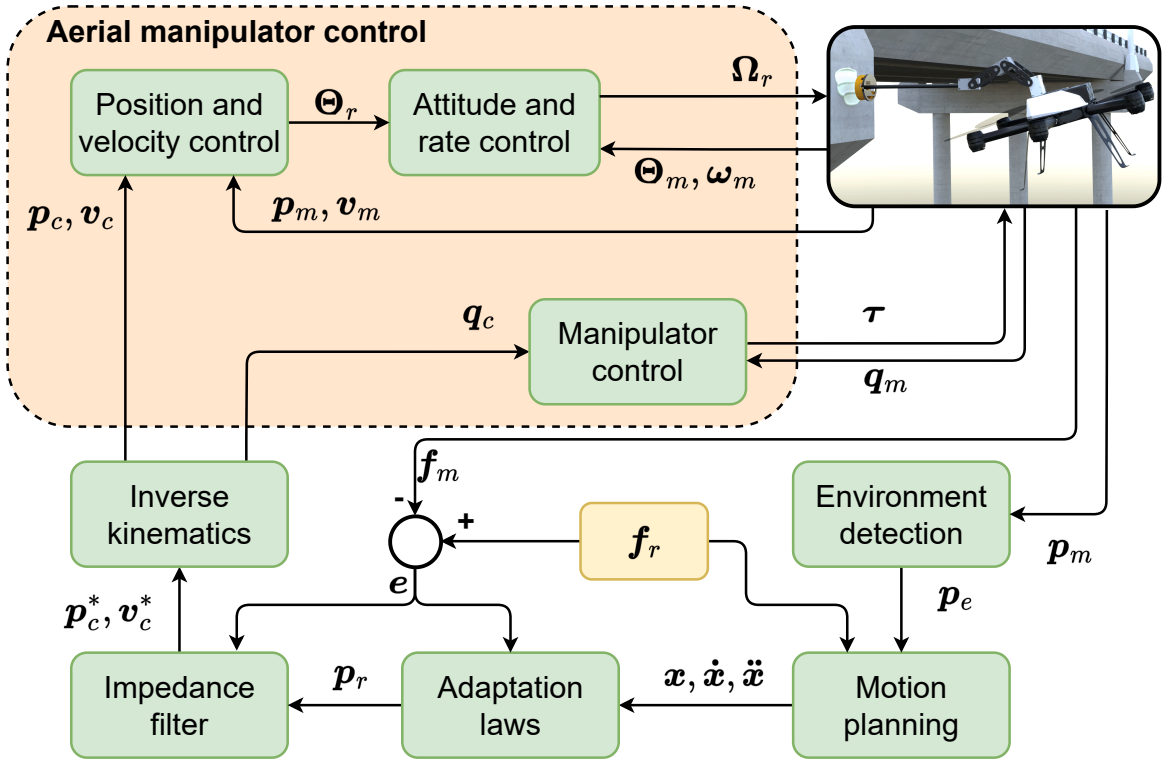


Figure 5.12: The overall system schematic with the aerial manipulator control described in Section 3.2, augmented with the adaptive impedance approach to include force tracking. The general workflow begins with the environment detection which provides a contact point. This is followed by the motion planning which plans a trajectory towards the contact point, including the force reference $\mathbf{f}_r$ provided by the user. The trajectory is then sent to the adaptive position-based impedance which generates suitable references for the multirotor position controller and manipulator joints. Note that all position vectors consist of three components, $\mathbf{p}_* = \begin{bmatrix} x_* & y_* & z_* \end{bmatrix}^T$.

order dynamics:

$$f_r(t) - f_m(t) = m \cdot (\ddot{x}_c(t) - \ddot{x}_r(t)) + b \cdot (\dot{x}_c(t) - \dot{x}_r(t)) + k \cdot (x_c(t) - x_r(t)), \tag{5.10}$$

where $f_r(t)$ is the force reference supplied by the user, $x_r(t)$ is the position input to the impedance filter, and $x_c(t)$ is the perturbed reference required to achieve the desired force reference. The filter has three parameters: m is the mass, b is the damping factor, and k is the stiffness. These parameters are controlling the impedance filter dynamics and are typically set by the user to follow the desired dynamics. Since these dynamics will most likely differ from the aerial manipulator dynamics, they must be taken into account in the motion planning. Namely, dynamical constraints have to be set to conform with the impedance filter dynamics. Note that during the free space motion, the measured force is going to be $f_m(t) = 0$, and the user set referent force also needs to be set to zero. This way, the $x_c(t)$ follows the filter dynamics since the force error is also zero.

Taking the linear spring force model from equation (5.9), the force error can be written as:

$$e(t) = f_r(t) - f_m(t) = f_r(t) - k_e\left(x_m(t) - x_e(t)\right). \tag{5.11}$$

Assuming a constant force reference $f_r(t) = F_r$, and rearranging the equation above yields the measured position and its derivatives:

$$\begin{aligned} x_m(t) &= \frac{F_r - e(t)}{k_e} + x_e(t) \\ \dot{x}_m(t) &= -\frac{\dot{e}(t)}{k_e} + \dot{x}_e(t) \\ \ddot{x}_m(t) &= -\frac{\ddot{e}(t)}{k_e} + \ddot{x}_e(t). \end{aligned} \tag{5.12}$$

Recall the perturbed reference $x_c(t)$ (commanded position) introduced in equation (5.10). This is the impedance filter output, which also serves as an input to the aerial manipulator controllers. The underlying assumption is that the aerial manipulator can follow the commanded position with negligible error. The overall system dynamics follows the impedance filter dynamics in such a case. This assumption can be resolved by setting the filter parameters m, b and k to be slower than the aerial manipulator dynamics. Indeed, in such a case the measured position closely follows the commanded position $x_c(t) \approx x_m(t)$. Plugging this assumption in equations (5.12), (5.11) and expanding equation (5.10) yields the force error dynamics:

$$\begin{aligned} m\ddot{e}(t) + b\dot{e}(t) + (k + k_e)e(t) = {} & mk_e\left(\ddot{x}_e(t) - \ddot{x}_r(t)\right) + bk_e\left(\dot{x}_e(t) - \dot{x}_r(t)\right) \\ & + k\left(F_r + k_e x_e(t) - k_e x_r(t)\right). \end{aligned} \tag{5.13}$$

The above equation shows the evolution of error with respect to the user provided trajectory and environment position. In steady state, all derivatives are considered to be zero, which yields the error:

$$e_{ss} = \frac{k}{k + k_e}\left(F_r + k_e x_e - k_r x_r\right). \tag{5.14}$$

The position reference required to eliminate the steady state force error is:

$$x_r = \frac{F_r}{k_e} + x_e. \tag{5.15}$$

Considering the equation above, setting $x_r = x_e$ drives the end-effector to the environment contact point, i.e. wall contact. Setting some force F_r moves the referent position "into" the wall for the amount F_r/k_e, which is consistent with the linear spring model. Knowing the environ-

ment stiffness k_e and position x_e perfectly, allows for exact calculation of the position reference required to achieve that force at the contact point. In some cases, where the exact contact force is not relevant, this may be a satisfactory behavior. However, in general case it is not recommended to assume the same stiffness k_e since it can significantly change for each contact point.

5.3.1.1 Adaptive Impedance

To alleviate the aforementioned problem, an adaptive approach to estimate the unknown stiffness can be employed. Although both k_e and $x_e(t)$ are unknown, the latter can be obtained through a visual or point cloud detection. Therefore, only the environment stiffness is left unknown and can be subjected to an adaptation law. An adaptive parameter $\kappa(t)$ is introduced to account for environment stiffness:

$$\begin{aligned} x_r(t) &= \kappa(t)F_r + x_e(t) \\ \dot{x}_r(t) &= \dot{\kappa}(t)F_r + \dot{x}_e(t) \\ \ddot{x}_r(t) &= \ddot{\kappa}(t)F_r + \ddot{x}_e(t), \end{aligned} \tag{5.16}$$

where $x_r(t)$ is the input to the impedance filter. Plugging the above into equation (5.13) yields:

$$\ddot{e}(t) + \frac{b}{m}\dot{e}(t) + \frac{k+k_e}{m}e(t) = F_r\left(\frac{k\left(1-k_e\kappa(t)\right) - bk_e\dot{\kappa}(t)}{m} - k_e\ddot{\kappa}(t)\right). \tag{5.17}$$

The term $\kappa(t)$ replaces the environment stiffness parameter with an unknown adaptive value. In the steady state, when all derivatives are zero together with the error, the adaptive parameter becomes an inverse of the environment stiffness $\kappa_{ss} = 1/k_e$. Using the Routh-Hurwitz stability criterion when designing the impedance filter, together with Lyapunov stability analysis for the adaptive parameter dynamics, yields a stable system response:

$$\begin{aligned} m\dddot{\kappa}(t) + b\ddot{\kappa}(t) + k\dot{\kappa}(t) &= \gamma\sigma(t) + \gamma_d\dot{\sigma}(t) \\ \sigma(t) &= p_2 e(t) + p_3\dot{e}(t), \end{aligned} \tag{5.18}$$

where parameters p_2 and p_3 are chosen according to the stability region obtained with the Routh-Hurwitz criterion, and intervals of parameters γ and γ_d are obtained through the Lyapunov analysis. The aforementioned parameters also have to yield the adaptive parameter $\kappa(t)$ dynamics that are faster than the system dynamics. Otherwise, the adaptive laws will not converge. The stability proofs for equation (5.18) go beyond the scope of this thesis, but an interested reader is encouraged to read more in [33, 137, 138].

5.3.2 Bridge Sensor Mounting

An aerial manipulator equipped with a suitable sensory apparatus can be employed for a bridge inspection. There are three types of bridge inspections: periodic, special and damage inspections. Periodic inspections are usually performed in time intervals defined by national standards of each country. Special inspections are required when a deficient bride element needs frequent monitoring, and damage inspections are performed after a destructive event, i.e. environmental catastrophe. There are numerous inspection methods that can be performed to detect defects such as corrosion, voids, cracks, weak connections, and concrete delamination [33]. Some of these methods include mounting a specialized small sensor to monitor the bridge state, i.e. accelerometers, tilt meters, pressure sensors, and strain gauges. Typically, mounting these sensors requires specialized vehicles with cranes and platforms, designed to reach underneath the bridge. Sensors are then mounted by human operators, which poses a safety risk.

The aforementioned sensors can be mounted with an aerial manipulator capable of achieving and maintaining a wall contact. Since bridges have a lot of hard to reach places, the inspection sensors can be mounted by aerial manipulators. One way to mount a sensor to a wall or ceiling is by gluing it with a two-component adhesives, which form a solid bond after being combined. In such a scenario, the first aerial manipulator sprays a "resin" component on a desired surface for sensor mounting. The second aerial manipulator carries the sensor with pre-applied "hardener" component, and performs a wall contact using the adaptive impedance approach. The envisioned team of two aerial manipulators required to perform the sensor mounting task is shown on Fig. 5.13. The curing time of the two components varies from minutes to days, depending on the glue type and manufacturer. However, a certain amount of pressure needs to be applied to properly mix the two components while maintaining the wall contact, which is achieved through the adaptive impedance approach. In this case, two vehicles are chosen to perform the task in order to simplify the end-effector design. A single vehicle can also be employed, however, it would complicate the overall approach and the end-effector design.

5.3.2.1 Augmenting End-effector Kinematics

As derived in Section 3.1.1, both multirotor body and manipulator joint positions influence the end-effector configuration. On the other hand, the impedance filter only accounts for the end-effector position. Since there is coupling present between the body and manipulator motion, a $\beta \in [0,1]$ weighing parameter is introduced to distribute the end-effector motion. Recalling the equation (5.15) and rewriting it in vector form yields:

$$\mathbf{p}_r = \mathbf{F}_r \cdot diag\left(\frac{1}{k_{e,x}}, \frac{1}{k_{e,y}}, \frac{1}{k_{e,z}}\right) + \mathbf{p}_e = \Delta\mathbf{p} + \mathbf{p}_e, \tag{5.19}$$

where $\mathbf{p}_r$ is the reference to the impedance filter, $\mathbf{p}_e$ is the environment position, $\mathbf{F}_r$ is the force

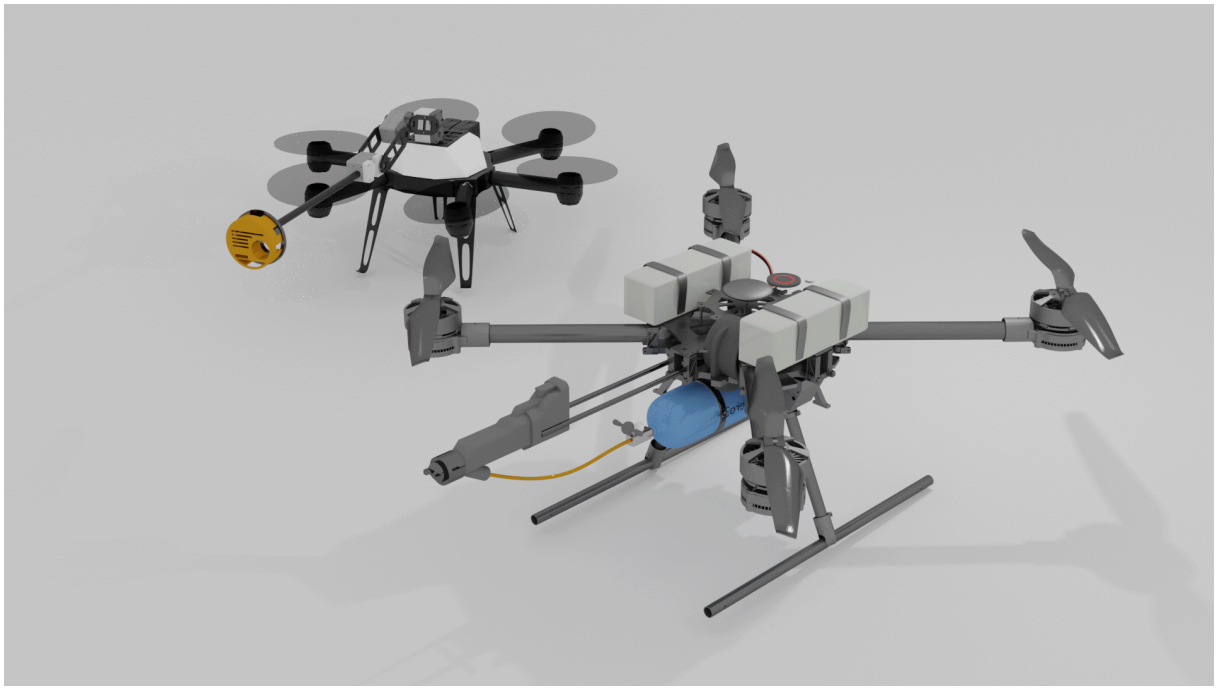


Figure 5.13: The envisioned team consisting of two aerial manipulators for bridge sensor mounting. The right aerial manipulator carries a pressurized tank with the resin component, and applies it on a bridge surface. The left aerial manipulator carries the sensor with the pre-applied hardener component, and glues the sensor on the bridge surface. Copyright [33] CC BY 4.0.

reference, and the diagonal matrix accounts for the environment stiffness in the three spatial dimensions. The linear spring member can be substituted by equivalent end-effector displacement $\Delta\mathbf{p}$, which is required to achieve the desired force $\mathbf{F}_r$. The end-effector displacement can be distributed between the multirotor position control and joint commands:

$$\begin{aligned} \Delta\mathbf{p}_{\text{UAV}} &= \beta \cdot \Delta\mathbf{p} \\ \Delta\mathbf{p}_{arm} &= (1-\beta) \cdot \Delta\mathbf{p}. \end{aligned} \tag{5.20}$$

Here, the $\Delta\mathbf{p}_{\text{UAV}}$ changes the body reference directly, while $\Delta\mathbf{p}_{arm}$ displaces the end-effector through the manipulator motion so the new joint values need to be computed with the inverse kinematics. Combining the two displacements gives the total end-effector displacement:

$$\begin{aligned} \Delta\mathbf{p} &= \Delta\mathbf{p}_{\text{UAV}} + \Delta\mathbf{p}_{arm} \\ &= \beta \cdot \Delta\mathbf{p} + (1-\beta) \cdot \Delta\mathbf{p}. \end{aligned} \tag{5.21}$$

Both displacements regarded above are expressed in the world coordinate system L_{W}, and can be expressed in any other coordinate system using the aerial manipulator kinematic chain from equation (3.6). If $\beta = 1$ is set, the multirotor body motion is used to control the position of

the end-effector and to achieve the required contact force. Setting $\beta = 0$ reverses the situation, where only the manipulator motion is controlling impedance. Any value in between distributes the impedance motion between the manipulator and multirotor control. In some situations, such an end-effector motion distribution is beneficial because of the limited manipulator reach. In that case, the multirotor body can move making it possible to follow the desired force reference. Therefore, the following section provides the attached manipulator workspace analysis to determine the optimal manipulator configuration, maximizing its end-effector displacement $\Delta\mathbf{p}_{arm}$.

5.3.2.2 Manipulator Workspace Analysis

To test the adaptive position-based impedance control, a 3 DoF manipulator is mounted on the multirotor body, with the base attached above the center of mass. The attached end-effector in the simulation environment is a rod. However, in real world it would be swapped with a sensor that has to be mounted on a bridge. The DH parameters of the manipulator are given in Table 5.5.

Table 5.5: DH parameters of the 3-DoF manipulator attached to the aerial manipulator in simulation. Note that there is a virtual joint 3* required to properly align the end-effector coordinate system. Careful reader will notice that first two links are, by design, the same length.

	θ_k	d_k	α_k	a_k
1	0	0	0	$0.0755m$
2	0	0	0	$0.0755m$
3	$\pi/2$	0	$\pi/2$	0
3*	0	$0.3450m$	0	0

Using the direct kinematics and DH parameters of the manipulator, the end-effector configuration can be expressed as a function of joint variables q_1, q_2 and q_3. For simplicity, the full homogeneous transform matrix $\mathbf{T}_0^{\mathrm{T}}$ is omitted. The position and approach vector are written as:

$$\mathbf{p}_0^{\mathrm{T}} = \begin{bmatrix} a_1(C_1 + C_{12}) + d_3 C_{123} \\ a_1(S_1 + S_{12}) + d_3 S_{123} \\ 0 \end{bmatrix}, \quad \mathbf{z}_0^{\mathrm{T}} = \begin{bmatrix} C_{123} \\ S_{123} \\ 0 \end{bmatrix}, \tag{5.22}$$

where a_1, a_2 and d_3 are defined in DH parameters Table 5.5, and the trigonometric functions are abbreviated, i.e. $C_{123} = cos(q_1 + q_2 + q_3)$. In practice, a contact point will also be defined by its position and approach vector. Therefore, the multirotor is mostly responsible to steer the system to reach the contact point, while the manipulator ensures the required approach angle. The approach vector can be written as $\begin{bmatrix} cos(\delta) & sin(\delta) & 0 \end{bmatrix}^T$, where δ is the desired inclination in

$x-z$ plane. A straightforward relation between joint angles can be defined as:

$$q_3 = \delta - q_1 - q_2. \tag{5.23}$$

The equation above constrains the q_3 to a single solution based on other two joint values. It also simplifies the workspace analysis since now it depends only on q_1 and q_2. The analysis in the remainder of this section is performed to obtain the optimal approach configuration $\mathbf{q}_{\mathrm{M}}^{*} = \begin{bmatrix} q_1^* & q_2^* & q_3^* \end{bmatrix}^T$. To obtain the optimal joint values, three measures are taken into account: dexterity $\mathfrak{D}(q_1, q_2)$, reach $\mathfrak{R}(q_1, q_2)$, and physical limit $\mathfrak{L}(q_1, q_2)$. Therefore, the optimal configuration ensures the manipulator is as far as possible from joint limits, while having the maximum reach of the end-effector.

With equation (5.23), the manipulator approach angle is constrained, and the joint q_3 is defined as a function of other two joints. The dexterity index determines how far is the current configuration from the manipulator null space [139]. It can be obtained from the Jacobian matrix:

$$\mathfrak{D}(q_1, q_2) = |\mathbf{J}^T \cdot \mathbf{J}|, \tag{5.24}$$

where the Jacobian matrix can be observed in a reduced form:

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial \mathbf{p}_0^{\mathrm{T}}}{\partial q_1} & \dfrac{\partial \mathbf{p}_0^{\mathrm{T}}}{\partial q_2} \end{bmatrix} \in \mathbb{R}^{3\times 2}. \tag{5.25}$$

Note that the approach vector $\mathbf{z}_0^{\mathrm{T}}$ remains constant, and q_3 is defined as function of q_1 and q_2. This renders the reduced Jacobian matrix sufficient for the dexterity index calculation.

The second measure is the manipulator reach. In the sensor mounting use case, the goal is to keep the manipulator end-effector as far as possible from the multirotor body. Such an approach improves the overall safety of the system, reducing the possibility of crashing into a wall. The reach is defined with:

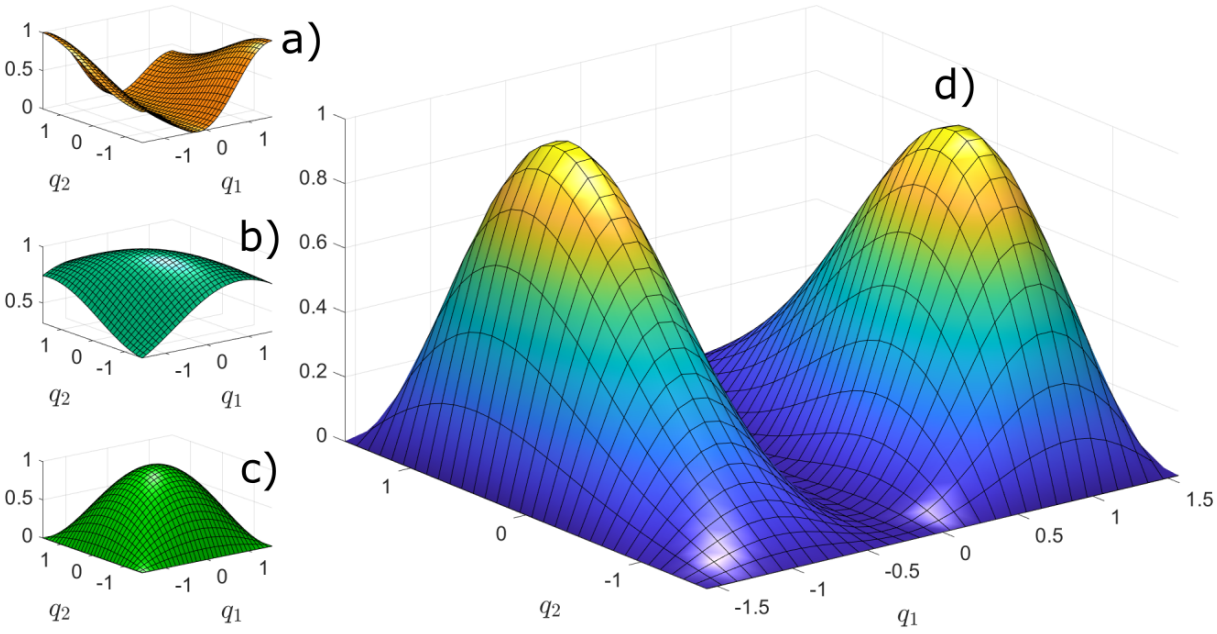
$$\mathfrak{R}(q_1, q_2) = \left(\mathbf{p}_0^{\mathrm{T}}\right)^T \cdot \mathbf{p}_0^{\mathrm{T}}, \tag{5.26}$$

which is the squared distance from the manipulator attachment point.

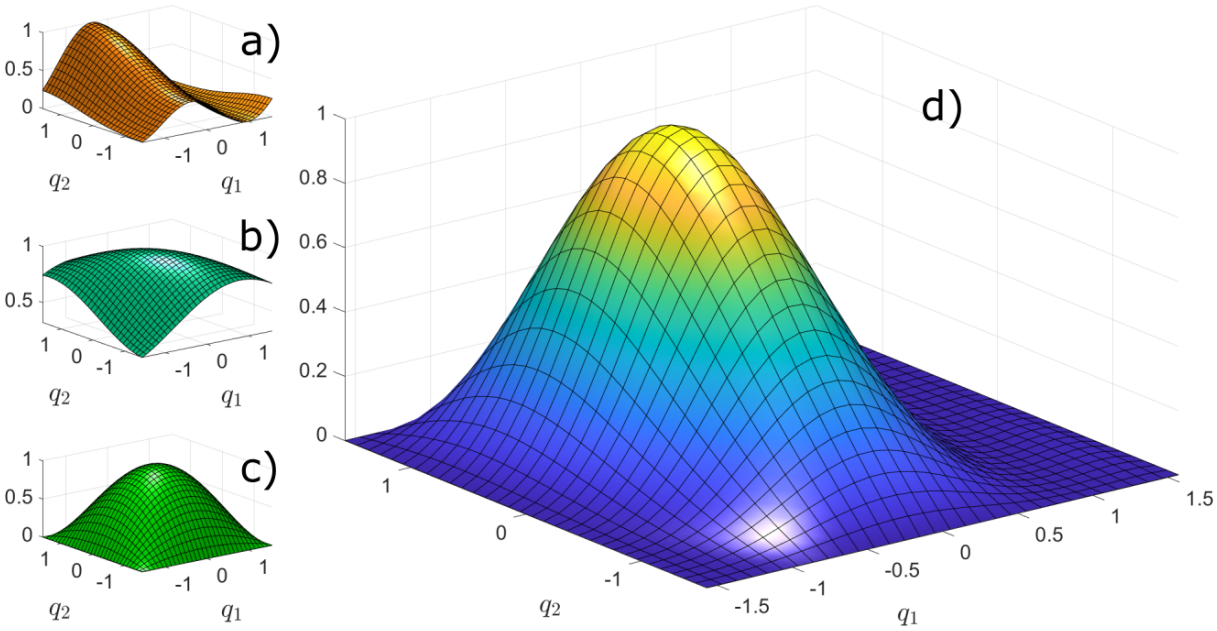
The third measure, limit, in this analysis determines how far are joint values from their respective limits. More favorable configurations in this sense are far from joint limits. The limit measure can be written as:

$$\mathfrak{L}(q_1, q_2) = \frac{(q_1^2 - q_{1,max}^2)(q_2^2 - q_{2,max}^2)}{q_{1,max} \cdot q_{2,max}}, \tag{5.27}$$

where $q_{1,max}$ and $q_{2,max}$ are the respective joint limits.



(a) Workspace analysis for $\delta = 0°$.



(b) Workspace analysis for $\delta = 68°$.

Figure 5.14: A visual decomposition of the performed workspace analysis for the horizontal contact with $\delta = 0°$ and inclined contact with $\delta = 68°$. *a)* Dexterity surface $\mathfrak{D}$ shows how far is some configuration from the manipulator null space. *b)* Reach surface $\mathfrak{R}$ defines how much can the end-effector move for some configuration. *c)* Limit surface $\mathfrak{L}$ shows how far is the currenc configuration from manipulator's physical limit. *d)* Combined manifold surface $\mathfrak{M}$. High values offer a better trade-off between dexterity, limit and reach, defining the optimal manipulator configuration $\mathbf{q}^*_{\mathrm{M}}$.

The three described measures are subsequently combined in a single manifold:

$$\mathfrak{M}(q_1, q_2) = \mathfrak{D}(q_1, q_2) \cdot \mathfrak{L}(q_1, q_2) \cdot \mathfrak{R}(q_1, q_2). \tag{5.28}$$

The visual representation of the manifold and each separate surface is shown on Fig. 5.14. The

analysis is performed for two representative cases, with $\delta = 0^\circ$ for a horizontal contact and $\delta = 68^\circ$ for an inclined contact, with detail explanation provided in Section 5.3.2.4. In case of $\delta = 0$, two peaks can be observed. Since all three manipulator joints have parallel rotation axes, these are considered as redundant configurations. The configuration $\mathbf{q}_{\mathrm{M}}^{*}(\delta = 0^\circ) = \begin{bmatrix} -0.861 & 0.557 & 0.304 \end{bmatrix}^T$ is chosen since the redundant configuration is impossible as the manipulator would collide with the multirotor body. In the second case with $\delta = 68^\circ$, only one maximum exists for configuration $\mathbf{q}_{\mathrm{M}}^{*}(\delta = 68^\circ) = \begin{bmatrix} -0.253 & 0.355 & 1.085 \end{bmatrix}^T$.

5.3.2.3 Trajectory Planning

As already discussed, the envisioned team consists of two multirotors with specialized tools for applying the resin and hardener component. The first multirotor applies the resin component, and it is no controlled through the adaptive impedance. Therefore, trajectory planning is rather straightforward in this case. On the other hand, the second multirotor is required to achieve wall contact and maintain the preset pressure to ensure a good bond between two components. Apart from position, orientation and manipulator joint values, the planning algorithm needs to be augmented with the force reference and weighing parameter β.

To do so, the waypoints supplied to the system need to be augmented with the force reference and weighing parameter:

$$\mathbf{w} = \begin{bmatrix} \mathbf{q}_{\mathrm{B}}^T & \mathbf{q}_{\mathrm{M}}^T & \mathbf{f}_r^T & \beta \end{bmatrix}^T \in \mathbb{R}^{13}, \tag{5.29}$$

where $\mathbf{q}_{\mathrm{B}} \in \mathbb{R}^6$ and $\mathbf{q}_{\mathrm{M}} \in \mathbb{R}^3$ are the multirotor and manipulator configurations defined in Section 5.1. The force reference $\mathbf{f}_r = \begin{bmatrix} f_x & f_y & f_z \end{bmatrix}^T \in \mathbb{R}^3$ is supplied in the world coordinate system, and β is a scalar parameter. The contact waypoint configuration relies on the position and orientation of the wall contact point. It is assumed here that the contact point position and plane normal are supplied by user or some detection algorithm. First, the angle δ is determined from the plane normal:

$$\begin{aligned} \mathbf{z}_{\mathrm{W}}^{\mathrm{C}} &= \begin{bmatrix} x_c & y_c & z_c \end{bmatrix}^T \\ \delta &= atan\left(\frac{z_c}{x_c}\right), \end{aligned} \tag{5.30}$$

where $\mathbf{z}_{\mathrm{W}}^{\mathrm{C}}$ is the plane normal expressed in the world frame. Based on the manipulator workspace analysis from Section 5.3.2.2, the optimal configuration $\mathbf{q}_{\mathrm{M}}^{*}$ is determined and set to the contact waypoint. Assuming hovering conditions at the contact point, the multirotor configuration $\mathbf{q}_{\mathrm{B}}$ is determined from the aerial manipulator kinematic chain, discussed in Section 3.1.1. The force reference $\mathbf{f}_r$ and weighing parameter β are supplied by the user.

Based on the final waypoint, a path is planned in the high dimensional space. However, due to the step change in the force and weighing parameter, the planned waypoints need to be addressed carefully. Namely, during the free space motion, the weighing parameter is set to $\beta = 1$. This ensures only the multirotor is affecting the end-effector impedance. The force reference is set to zero during the free space motion. Therefore, the waypoints planned with RRT* algorithm yield:

$$\begin{aligned} \mathscr{P} &= \left\{ \mathbf{w}_i \mid \mathbf{w}_i \in \mathbb{R}^{13}, i \in (1,2,\ldots,n_p) \right\} \\ \mathbf{w}_i &= \begin{bmatrix} \mathbf{q}_{\mathrm{B},i} & \mathbf{q}_{\mathrm{M},i} & \mathbf{0}_{3\times 1} & 1 \end{bmatrix}^T \\ \mathbf{w}_{n_p} &= \begin{bmatrix} \mathbf{q}^*_{\mathrm{B}} & \mathbf{q}^*_{\mathrm{M}} & \mathbf{f}_r & \beta_r \end{bmatrix}^T \end{aligned} \tag{5.31}$$

where $\mathbf{q}^*_{\mathrm{M}}$ is the optimal manipulator configuration based on the dexterity analysis, $\mathbf{q}^*_{\mathrm{B}}$ is the multirotor configuration that ensures reaching contact point with the optimal manipulator values.

The next step is the trajectory planning, via the TOPP-RA planner introduced in Section 4.3. Since the adaptive impedance controller expects a step change in the force and weighing parameter, the final waypoint has been added to ensure it. As for the TOPP-RA planner, large velocity and acceleration constraints are allowed on the force reference and the weighing parameter. However, due to the inner workings of the TOPP-RA planner, the end result yields a smooth profile, regardless of the dynamical constraints. Furthermore, as shown on Fig. 5.15, the TOPP-RA profiles have overshoots and undershoots which violate hard constraints for the weighing parameter $\beta \in [0,1]$. To alleviate this problem, a simple constant velocity interpolation is employed for the selected DoFs. By setting a large velocity constraint, it produces a step change expected by the adaptive impedance controller. The final trajectory is planned according to Section 5.1.1, with the adaptive impedance controller in the loop.

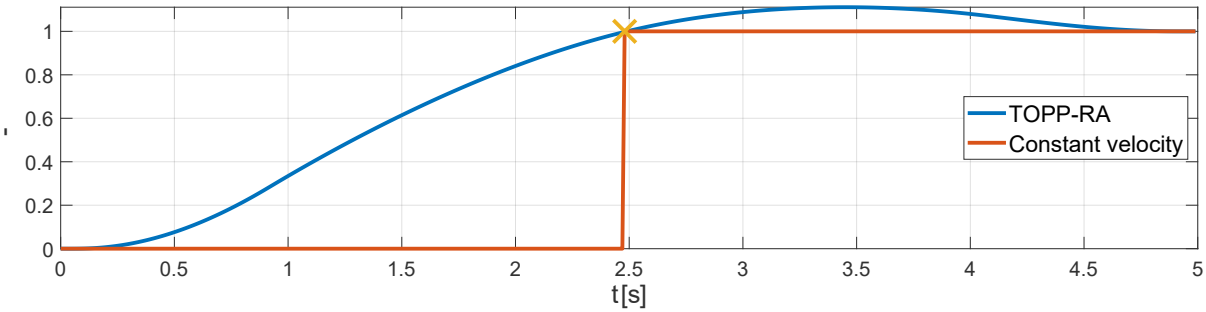


Figure 5.15: A comparison between the TOPP-RA and constant velocity interpolation with same input waypoints. Although the dynamical constraints are very large in both cases, the inner workings of the TOPP-RA take into account other DOFs. The produced trajectory overshoots the hard constraint imposed on the weighing parameter β, which is not suitable for the adaptive impedance control. Copyright [33] CC BY 4.0.

5.3.2.4 Simulation Results

The simulation environment used in this case is Gazebo with ROS middleware, as described in Section 5.1.1.1. To enable the force feedback required by the impedance filter and adaptation laws, a force sensor from the Gazebo library is mounted on the rod type end-effector. Similar to the real world force sensors, it provides high frequency feedback at $1kHz$, which is often noisy. To provide a smooth and stable feedback, a moving average filter is employed together with an exponential filter. The envisioned bridge sensor mounting mission is depicted on Fig. 5.16.

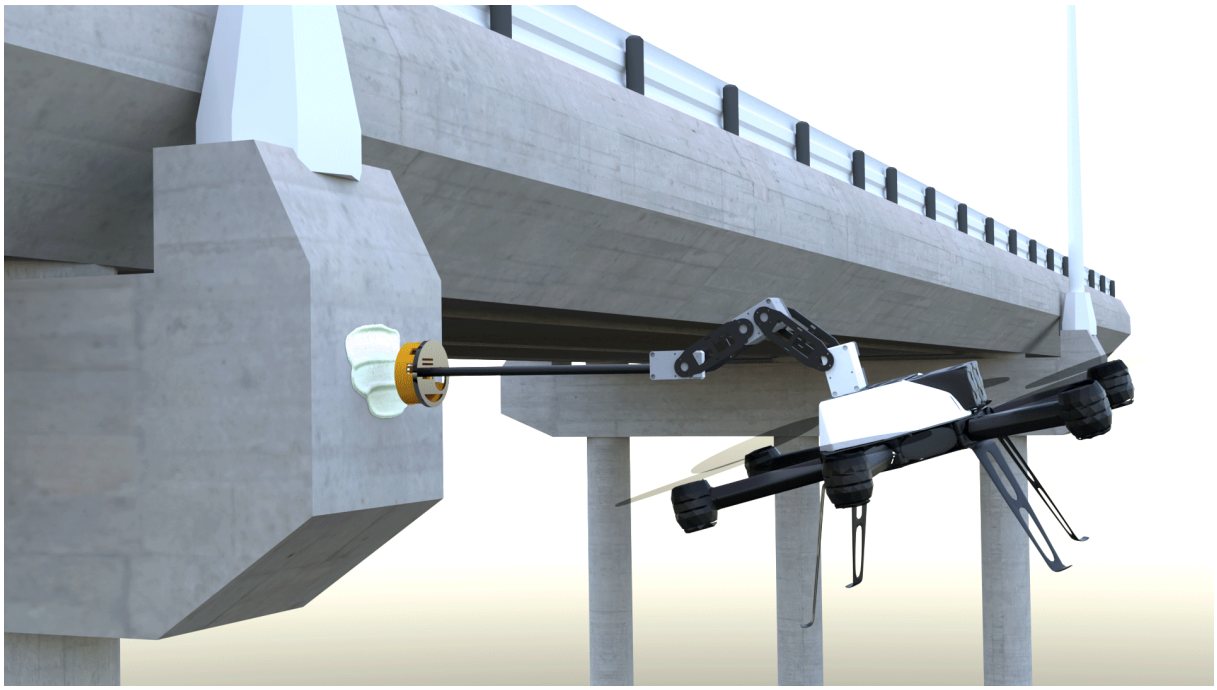


Figure 5.16: A snapshot of the aerial manipulator performing the bridge contact. Copyright [33] CC BY 4.0.

End-effector motion distribution analysis

In Section 5.3.2.1, the end effector motion distribution is introduced, depending on the weighing parameter β. To recap, the weighing parameter defines the ratio of how much are the manipulator joints and multirotor position contributing to achieve the desired end-effector configuration. For $\beta = 1$ only multirotor position is changed, moving it towards the desired configuration. For $\beta = 0$ the manipulator inverse kinematics are used.

An analysis with different values of the weighing parameter have been performed to determine its influence on the overall system. The desired contact point is chosen on the vertical bridge side, requiring a contact only along the global x axis. Note that upon contact there is some amount of force measured in other axes due to the end-effector slippage, however, that amount is relatively small and it is quickly eliminated. The results for the contact force with various weighing parameter values is depicted on Fig. 5.17. As can be observed, the force re-

sponse for various weighing parameter produced no significant difference between trials. Each response is characterized with starting oscillations which eventually settle and reach the desired value. Therefore, based only on the provided responses, there is no obvious conclusion how to choose the weighing parameter β. However, several factors are considered from a kinematic standpoint. Changing only the multirotor position to achieve some contact force inevitably tilts the multirotor body which increases the chance of the end-effector slipping. On the other hand, if the manipulator is responsible for changing the end-effector configuration, the desired configuration might not be feasible due to the multirotor motion. This is especially true during the approach, where the multirotor exhibits the largest tilt angles. The finally chosen weighing parameter is $\beta = 0.5$. This allows both parts of the system to influence the end-effector configuration while keeping the manipulator within its limits and effectively using its workspace.

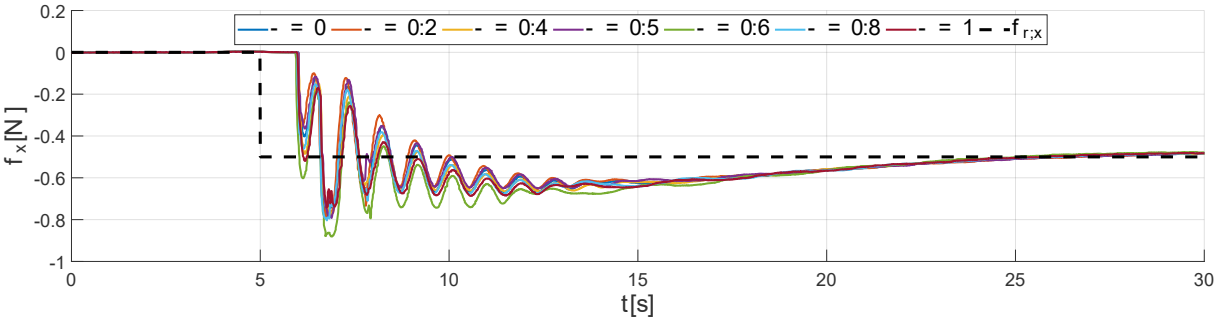


Figure 5.17: The force response for various values of the weighing parameter β. Trials are performed with force reference set only along the global x axis, $f_{r,x} = -0.5N$. Copyright [33] CC BY 4.0.

Horizontal and inclined sensor mounting

Since the main idea of the simulation is to test mounting sensors on a bridge, the trials are tailored in that direction. Namely, two representative surfaces are chosen: a vertical surface with $\delta = 0°$, and an inclined surface with $\delta = 68°$, with details provided in Section 5.3.2.2. Note that due to the manipulator attachment point on top of the multirotor, mounting on surfaces with $\delta < 0°$ is not recommended due to potential end-effector collision with the multirotor propellers.

The first set of simulation trials is conducted by achieving and maintaining a contact force along the world x axis, with a representative example shown on Fig. 5.18. A time delay between the reference and achieving the contact point can be observed. This is present due to the impedance filter which slows the overall system dynamics. After the initial contact, some oscillations are present while the adaptation laws estimate the environment stiffness k_e, after which the system settles and reaches the desired force reference.

The second set of simulation trials is conducted with an inclined contact. The surface is located underneath the bridge and corresponds to $\delta = 68°$. This requires the system to approach the contact point from below and achieve contact in both world x and z axes, as shown on Fig.

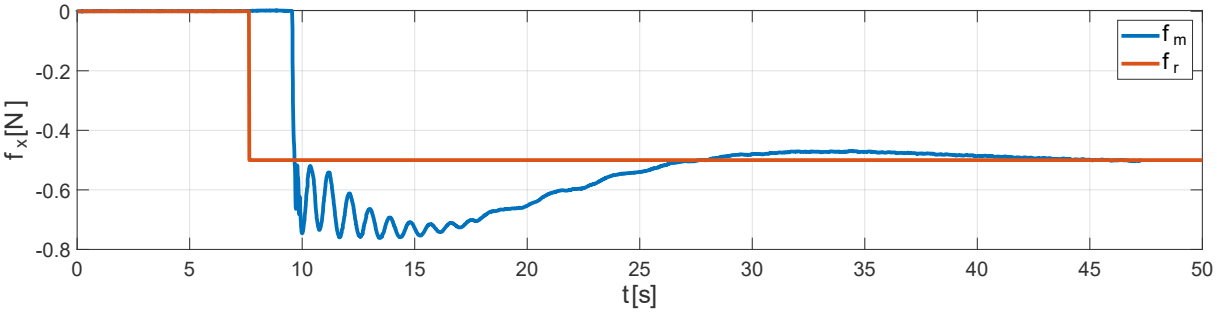


Figure 5.18: An example of force reference f_r and measured force f_m for horizontal contact with $\delta = 0°$. For simplicity, the force only acts along the world x axis. Copyright [33] CC BY 4.0.

5.19. Upon contact, there are some oscillations and the force eventually settles at the referent value, after the environment stiffness k_e has converged.

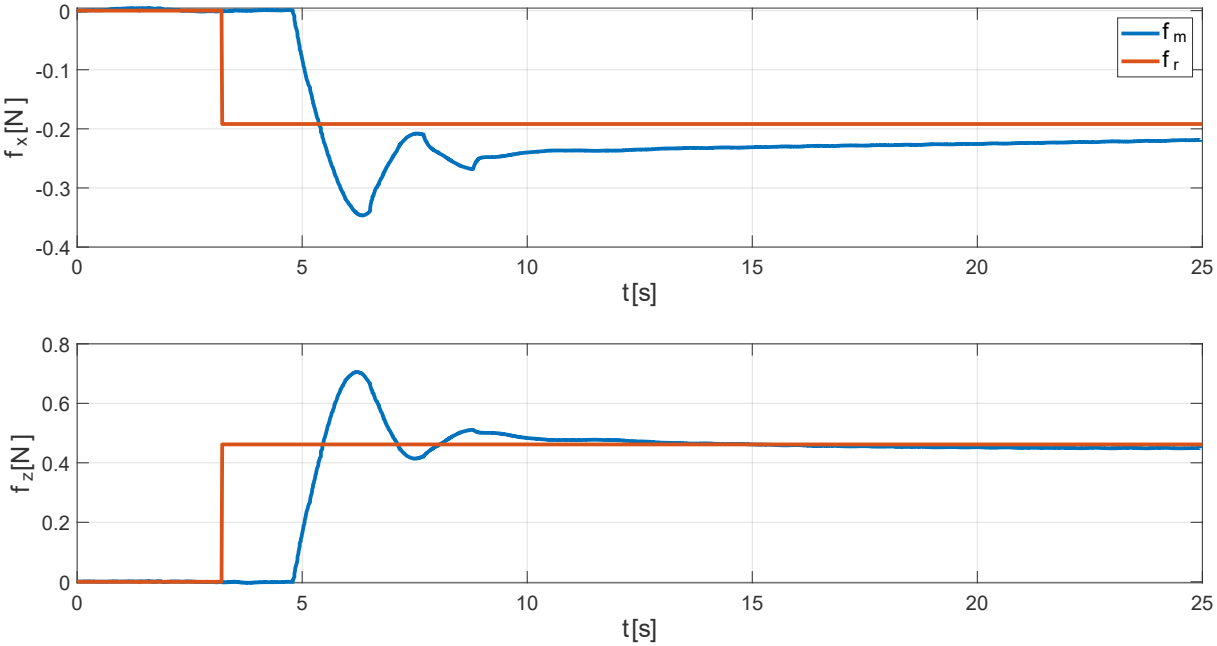


Figure 5.19: An example of force reference f_r and measured force f_m for inclined contact with $\delta = 68°$. In this case, the force is set along both world x and z axes. Copyright [33] CC BY 4.0.

To evaluate the system performance in the simulation environment, $n = 10$ trials were performed for both wall inclinations. The first evaluation metric is the end-effector approach vector deviation. If the target normal is denoted with $\mathbf{z}_{\mathrm{W}}^{\mathrm{TG}}$ and the end-effector approach vector with $\mathbf{z}_{\mathrm{W}}^{\mathrm{T}}$, the metric is defined as the dot product of these two vectors. The dot product value of $\mathbf{z}_{\mathrm{W}}^{\mathrm{TG}} \cdot \mathbf{z}_{\mathrm{W}}^{\mathrm{T}} = 1$ means the two vectors are parallel, which is the intended behavior. The second metric is the distance from the target, where $\mathbf{p}_{\mathrm{W}}^{\mathrm{TG}}$ is the target position and $\mathbf{z}_{\mathrm{W}}^{\mathrm{T}}$ is the end-effector position in the world frame. The distance is measured after the force response has settled, using the norm $d = |\mathbf{p}_{\mathrm{W}}^{\mathrm{TG}} - \mathbf{p}_{\mathrm{W}}^{\mathrm{T}}|$. The results of the repeatability analysis are summarized on Fig. 5.20. The left portion of the figure shows the orientation deviation, which indicates the end-effector is parallel with the contact plane normal. Although a slight error is present, it can be neglected

since the dot product values are very close to 1. The distance from the target is below $0.1m$ for all trials, indicating a relatively small position error. For the inclined contact, the errors are higher than in the horizontal case. This is attributed to the observed increased slippage while performing the inclined contact trials. It is worth mentioning that the distance error is useful for the motion planning of the resin spraying multirotor. Since sensors do not have to be mounted on the bridge with high precision, it indicates how large area around target should be sprayed to guarantee successful attachment.

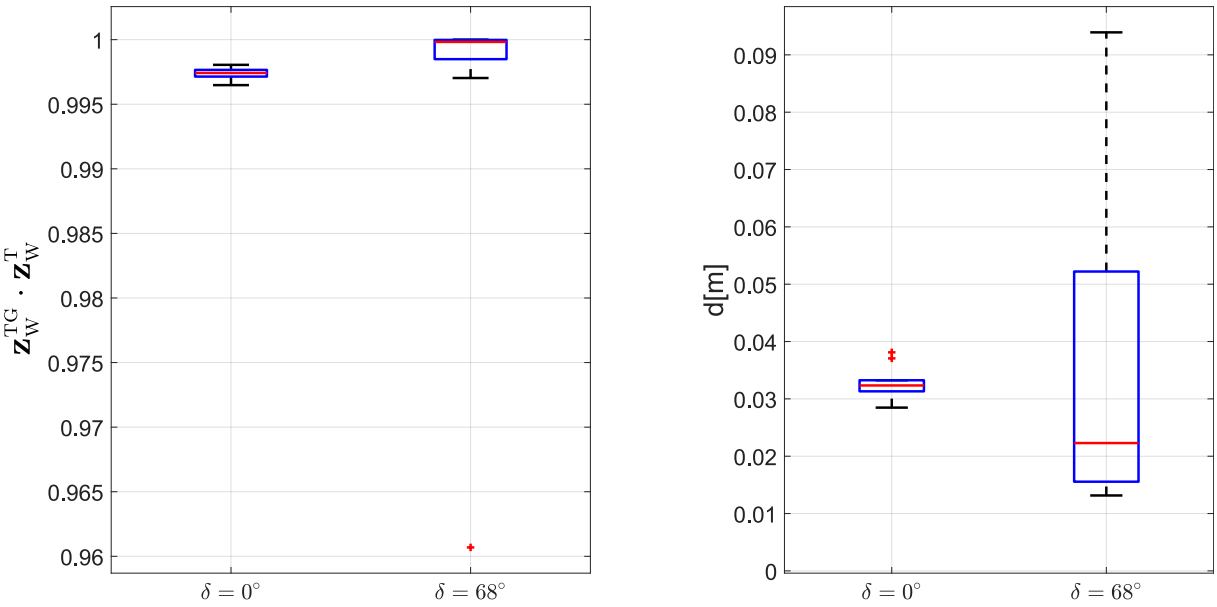


Figure 5.20: *Left:* box and whiskers plot of the dot product between the target plane normal and the end-effector approach vector. *Right:* box and whiskers plot of the distance from the target after achieving contact. Copyright [33] CC BY 4.0.

CHAPTER 6

Heterogeneous Multi-robot System Motion Planning

The previous chapter introduced a motion planner for aerial manipulators based on the system dynamical model. The main issue the previous chapter tackled is the underactuated nature of the multirotor vehicles, with the emphasis on planning for the specified end-effector motion. The goal of this chapter is to extend the single-vehicle model planning to multiple vehicles manipulating the same object. This increases both planning and execution complexity, since the manipulators' end-effectors influence the motion of each other, due to their coupling through the transported object. Depending on the task at hand, the error tolerance while manipulating the common object can be severely limited, requiring a careful approach in the motion planning procedure. On the other hand, such an approach increases the extent of potential applications, i.e. transporting objects in cluttered environments, lifting objects too heavy for a single robot, etc.

With main focus on the underactuated system, the goal of this method is to correct the end-effector configuration by executing the planned trajectory in a simulation environment. With multiple agents in the team, each robot is considered to correct its respective end-effector configuration independently. Naturally, the effectiveness of these corrections heavily depends on the manipulator null space, which is directly tied to the manipulator construction. The motion planning method starts by supplying the desired configuration of the transported object in the world frame. This is followed by a heuristic approach to determine each manipulator configuration for grasping the object. Based on the end-effector configuration in the world frame, as well as the manipulator configuration, the multirotor body configuration is determined. This allows to obtain the full aerial manipulator configuration for each sampled object configuration, which

is the basis for trajectory planning and corrections. The method is evaluated on several aerial manipulator combinations, determining the error of the planned trajectory against trajectories executed by the model and finally executed corrected trajectory.

6.1 Multi-robot System Model-based Motion Planning

Having multiple robots manipulating a single object requires specifying each end-effector attachment point on that object. Although the motion method presented in the reminder of this section supports an arbitrary number of robots, the examples and simulation trials are conducted on heterogeneous two robot systems. Namely, such systems comprise of two aerial manipulators with different robotic arms attached to their bodies. The cooperative team is considered to transport a rod, which simplifies specifying the grasping points for the end-effectors. An illustrative example of the transportation task is shown on Fig. 6.1.

The coordinate system of the manipulated object (payload) is denoted with L_{P}, and it is assumed it corresponds to the object's center of mass. In a general case, n_r robots are considered to manipulate the payload. Throughout this chapter, the subscript i will denote the i^{th} aerial manipulator. An appropriate grasping point for each end-effector can be defined with respect to the L_{P} coordinate system, as ${}_i\mathbf{T}_{\mathrm{P}}^{\mathrm{T}}$, where i denotes the i^{th} end-effector. This allows obtaining each end-effector transform in the world frame as:

$$ {}_i\mathbf{T}_{\mathrm{W}}^{\mathrm{T}} = \mathbf{T}_{\mathrm{W}}^{\mathrm{P}} \cdot {}_i\mathbf{T}_{\mathrm{P}}^{\mathrm{T}}, i \in (1 \dots n_r), \tag{6.1} $$

where $\mathbf{T}_{\mathrm{W}}^{\mathrm{P}}$ is the payload transform in the world frame, and ${}_i\mathbf{T}_{\mathrm{W}}^{\mathrm{T}}$ is the i^{th} end-effector transform in the world frame. The above equation is particularly useful if the desired object transform is specified in the world frame, as it allows to calculate the desired end-effector configuration of each aerial manipulator. Note that the transforms ${}_i\mathbf{T}_{\mathrm{P}}^{\mathrm{T}}$ are determined only once by the user, and a careful approach is recommended to properly align the end-effector with the attachment point on the payload.

6.1.1 Waypoint Configuration

Having the transform of each end-effector in the world frame is the first step towards determining the full aerial manipulator state to achieve the desired configuration. For determining the waypoint configuration, and for the path planning, this is followed by two assumptions: i) the multirotor base is at hovering state; ii) there exists a manipulator configuration satisfying the desired end-effector configuration. The former assumption has already been introduced in Chapter 5, where the planning method resides on planning with zero roll and pitch angles and later applies the end-effector corrections based on the system dynamical model. This approach

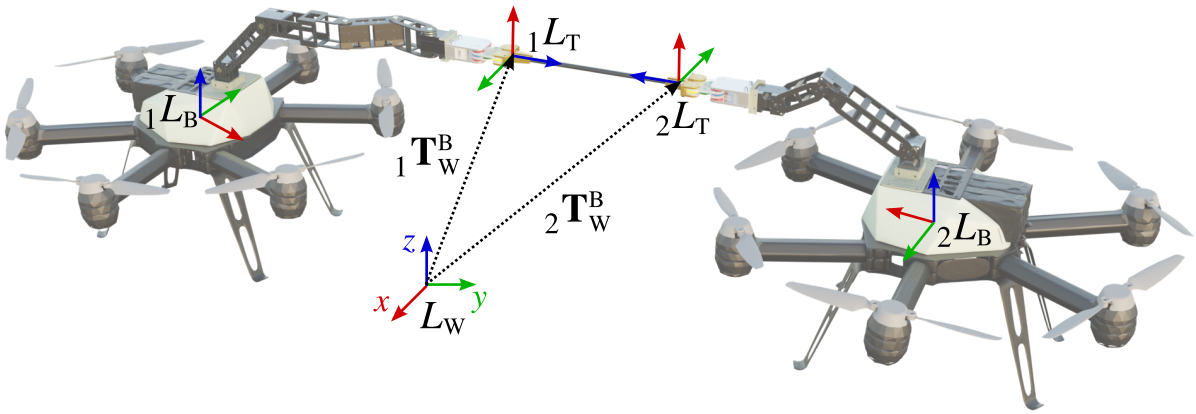


Figure 6.1: Coordinate systems of two aerial manipulators transporting a common object. In this case, the two aerial manipulators are coupled through a rod type payload.

can be further extended to any type of the manipulator base, fixed or floating, while this thesis focuses on the co-planar multirotors. The second assumption considers the transformation between the manipulator attachment point and the tool ${}_i\mathbf{T}^T_0$ exists and respects the hovering condition. Since the end-effector transform can be anywhere in the world frame, it is required to determine the multirotor base transform $\mathbf{T}^B_W$. This renders the inverse kinematics solution impractical, as the multirotor base transform is a-priori unknown.

Alternative way to calculate the manipulator configuration is through a workspace analysis, similar to the one performed in Section 5.3.2.2. The basis of such a method is analyzing the manipulator dexterity, limit and reach for some end-effector configuration. The outcome is the best-fitting manipulator configuration $\mathbf{q}_M$, that is the farthest away from joint limits, while being able to utilize the manipulator null space to keep the desired end-effector configuration during motion. In the example of the safe payload transportation, the most important concern is keeping the relative desired position and orientation. Having the aerial manipulator use case in mind, finding the optimal manipulator configuration can be done by first computing the orientations from the planned payload trajectory. The workspace analysis can be then performed for each orientation to obtain the optimal manipulator configuration $\mathbf{q}_M$, while satisfying the hovering assumption of the multirotor body. This is a relatively general description of the concept since the attached manipulators can greatly differ depending on the task. However, it is applicable to most manipulators by analyzing the kinematics and workspace. An example and guideline for determining the manipulator configuration $\mathbf{q}_M$ is given in the following Section 6.1.1.1.

Recalling the aerial manipulator generalized coordinates vector from equation 4.3, the multirotor position and orientation vector ${}_i\mathbf{q}_B$ remains to be determined. Taking advantage of the determined manipulator configuration vector, the desired multirotor body transform in the world

frame can be written as:

$$\begin{aligned} {}_i\mathbf{T}_{\mathrm{W}}^{\mathrm{B}} &= {}_i\mathbf{T}_{\mathrm{W}}^{\mathrm{T}} \cdot ({}_i\mathbf{T}_{0}^{\mathrm{T}})^{-1} \cdot \left({}_i\mathbf{T}_{\mathrm{B}}^{0}\right)^{-1} \\ &= \mathbf{T}_{\mathrm{W}}^{\mathrm{P}} \cdot {}_i\mathbf{T}_{\mathrm{P}}^{\mathrm{T}} \cdot ({}_i\mathbf{T}_{0}^{\mathrm{T}})^{-1} \cdot \left({}_i\mathbf{T}_{\mathrm{B}}^{0}\right)^{-1}, \end{aligned} \tag{6.2}$$

where ${}_i\mathbf{T}_{\mathrm{B}}^{0}$ is the fixed transform depending on the manipulator attachment point on the base, and ${}_i\mathbf{T}_{\mathrm{W}}^{\mathrm{B}}$ is the multirotor body transform required to reach the desired end-effector configuration. Recalling the first assumption, the multirotor is considered to be in the hovering state at this planning stage, yielding zero roll and pitch angles. This is in line with the previous chapter, as the end-effector configuration is later corrected through the dynamical model. The above transformation allows obtaining the multirotor configuration vector ${}_i\mathbf{q}_{\mathrm{B}}$, as defined through equation 4.1. Having both manipulator and multirotor configurations, the full aerial manipulator generalized coordinates vector can be written as:

$$\mathbf{q}_i = \begin{bmatrix} {}_i\mathbf{q}_{\mathrm{B}}^T & {}_i\mathbf{q}_{\mathrm{M}}^T \end{bmatrix}^T. \tag{6.3}$$

Note that this definition can be extended to any manipulator base, where a fixed base would be constrained to a single position within the path planner.

Finally, each waypoint consists of multiple manipulators and can be written in a general form:

$$\mathbf{w} = \begin{bmatrix} \mathbf{q}_1^T & \mathbf{q}_2^T & \dots & \mathbf{q}_{n_r}^T \end{bmatrix}^T. \tag{6.4}$$

At a first glance, this type of waypoint notation seems very tightly packed as it carries the information about the overall multi-robot system. However, defining a high dimensional waypoint like so allows time synchronizing trajectories through the TOPP-RA planner, as discussed in Section 4.3.

6.1.1.1 Determining Manipulator Configuration Example

To give the reader an idea how to determine the manipulator configuration $\mathbf{q}_{\mathrm{M}}$, an example is provided on a 5-DoF manipulator. This manipulator has already been used in the previous chapter, with details provided in Section 5.2.1. This example is used as a guideline, note that each manipulator requires a slightly different approach based on its design. The example manipulator is depicted on Fig. 6.2. Observed from the multirotor base frame, the first two joints of the manipulator determine the end-effector pitch, while the following three joints determine the yaw angle.

Since the manipulator home position is straight up, to achieve a zero end-effector pitch

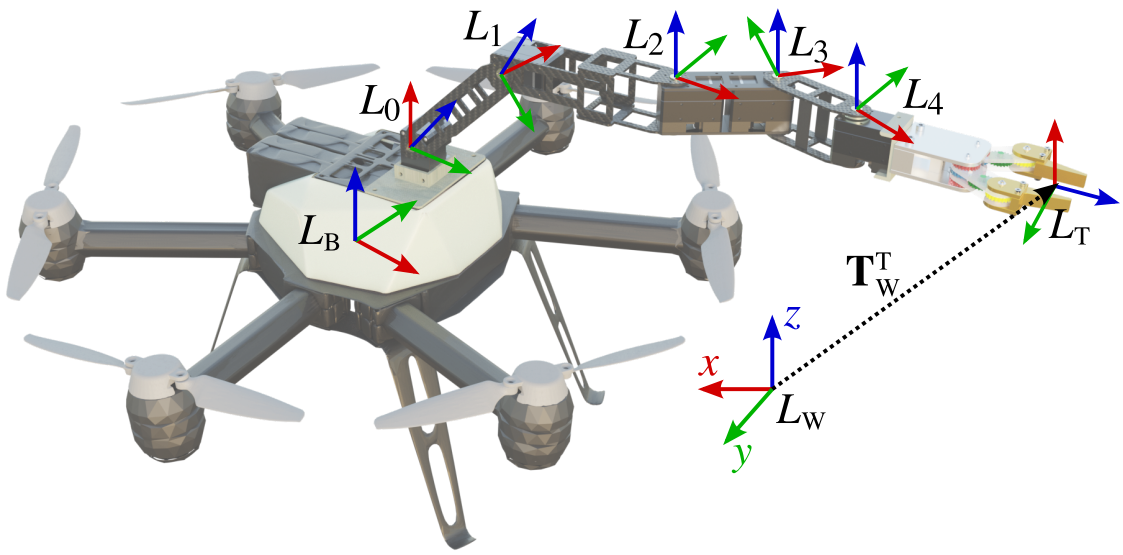


Figure 6.2: The 5-DoF manipulator serving as an example for determining the configuration $\mathbf{q}_M$, with indicated coordinate systems.

angle $\theta_T = 0$ the manipulator needs to rotate for $\pi/2$. Since two joints operate to change this angle, the reference is distributed equally yielding $q_1 = \pi/4$ and $q_2 = \pi/4$. In case the required angle is different from zero, it is also equally distributed between the first two joints yielding $q_1 = \pi/4 + \theta_T/2$ and $q_2 = \pi/4 + \theta_T/2$. Note that the lower bound for θ_T is defined by the physical limits of the multirotor body and propellers. This definition for the first two joints keeps the manipulator far away from its joint limits and potential collisions with the multirotor body, while leaving enough workspace to correct errors induced by the multirotor underactuated nature.

The second concern is about the end-effector yaw angle ψ_T. There are three joints determining this orientation. Since the multirotor body can freely rotate around its z_B axis, the first constraint on these three joints is $q_3 + q_4 + q_5 = 0$. This ensures the manipulator is always pointing along the x_B multirotor body axis. Setting all three joints to zero will satisfy this constraint, however, the manipulator is going to be far away from the null space since it is fully extended in such a configuration. In section 5.3.2.2, a workspace analysis has been conducted for a 3-DoF manipulator with same configuration as the last three joints of this manipulator. In a general case, a similar analysis can be performed, but for this particular manipulator the joint values are already determined by the workspace analysis, yielding $q_3 = 0.557$, $q_4 = -0.861$, and $q_5 = 0.304$. This configuration keeps the manipulator at zero yaw, at the same time being far from the joint limits which leaves enough room for applying the dynamical model corrections. The final manipulator configuration can be written as:

$$\mathbf{q}_M(\theta_T) = \begin{bmatrix} \pi/4 + \theta_T/2 & \pi/4 + \theta_T/2 & 0.557 & -0.861 & 0.304 \end{bmatrix}^T. \tag{6.5}$$

In this example, the manipulator workspace is decoupled for two manipulator orientation

angles θ_T and ψ_T. Since the manipulator design does not allow changing the roll angle, it is not taken into account when determining the manipulator configuration. Decoupling the manipulator workspace simplifies determining the configuration $\mathbf{q}_M$, while leaving enough joint movement for applying the model corrections. Note that this analysis is heuristic in nature, relying on the user experience and imposed task constraints.

6.1.2 Path and Trajectory Planning

The path planning problems presented so far in this thesis are developed for a single aerial manipulator. The input to the path planner is a set of two or more waypoints, and the RRT* algorithm samples the high dimensional space yielding a piecewise straight obstacle-free path. Such an approach was possible due to relatively relaxed task constraints imposed on the planner. Namely, the final aerial manipulator configuration is basically the only task constraint the planner has to satisfy, and there were no requirements on how to reach it. Note that this approach can be extended to multiple vehicles that are not cooperating or are physically linked.

The fundamental difference in case of the multiple aerial manipulators transportation task is the physical link between aerial manipulator, which is the manipulated object itself. As discussed in the previous section, the end-effector and full vehicle configurations are determined from the payload position and orientation. This creates a manifold within the search space that contains valid samples, complying with the imposed task space constraint. Indeed, this is a hard task space constraint imposed on the planner since all random samples must strictly follow the determined configuration of each robot. The main issue is sampling configurations lying on the desired manifold in the high dimensional space. Namely, by performing a random sampling in the high dimensional space, it is impossible for the sampled point to lie on the desired manifold. The work presented in [99] proposes a solution to a similar problem by orthogonally projecting the sampled configuration on the constraint manifold. However, determining the required manifold is not trivial, and it can be computationally intensive to find the closest configuration on the highly nonlinear constraint manifold.

Therefore, a different approach is proposed within this section. Instead of planning for the high dimensional multi-robot system configuration, the path is planned for the payload instead. Based on the planned payload path, the full system state can be calculated as described in the previous section. An illustrative example of such a path planner is depicted on Fig. 6.3. The payload center of mass configuration vector is identical to the waypoint configuration, defined as:

$$\mathbf{w}_P = \mathbf{q}_P = \begin{bmatrix} x_P & y_P & z_P & \phi_P & \theta_P & \psi_P \end{bmatrix}^T, \tag{6.6}$$

which is a full 6-DoF configuration. The planner spatial constraints are defined by the envi-

ronment, as aerial manipulators can typically reach any point except obstacles. The roll ϕ_{P} and pitch θ_{P} constraints are subject to the maximum angles the manipulator can achieve without colliding with itself or the multirotor body, while the yaw ψ_{P} angle is sampled freely as $SO(2)$ group, because it can be achieved by the multirotor body. Based on the start and goal payload configuration, the RRT* planner produces a path:

$$\mathscr{P}_{\mathrm{P}} = \left\{ \mathbf{w}_{\mathrm{P},j} \mid j \in (1,2,\ldots,n_{p,\mathrm{P}}), \mathbf{w}_{\mathrm{P},j} \in \mathbb{R}^6 \right\}. \tag{6.7}$$

This path is expressed in terms of the payload configuration at this point. Before it is transformed to the full system configuration, two concerns need to be considered.

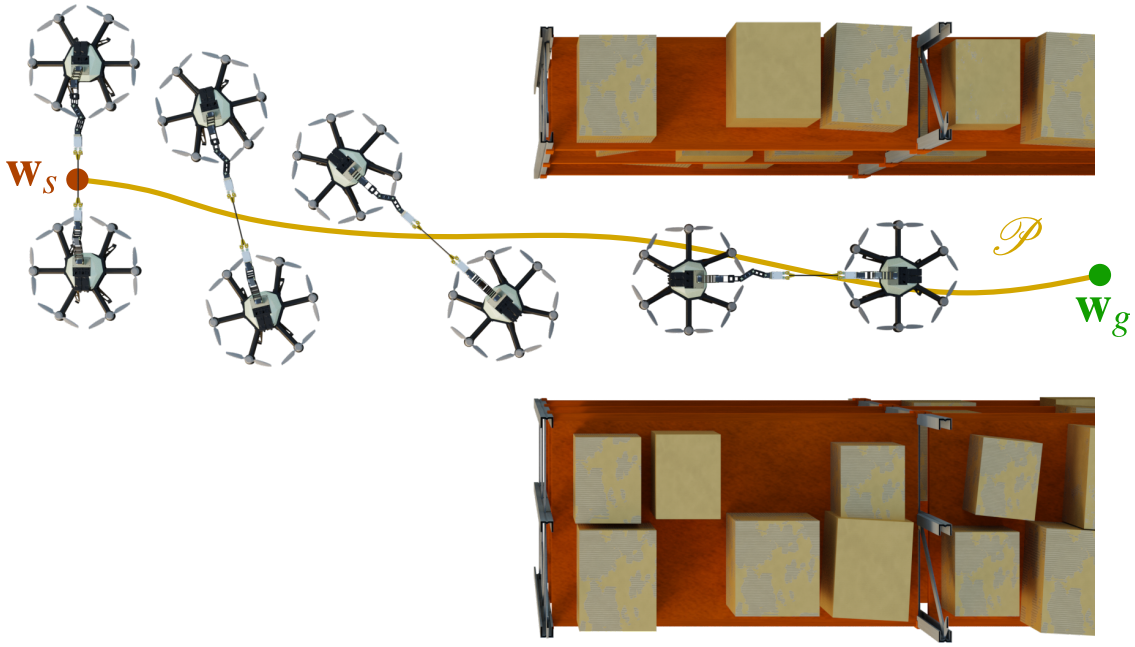


Figure 6.3: A path planned and smoothed with Bezier curves for a two robot system transporting a rod payload. The path $\mathscr{P}$ is planned between the start configuration $\mathbf{w}_s$ and goal configuration $\mathbf{w}_g$.

The first concern for transporting a payload is the possibility of having a long distance between the payload center of mass and the multirotor body. The system turning around the z axis can possibly require a large tangential velocity of the multirotor body, which can violate the dynamical velocity constraints. Therefore, the payload states need to be carefully sampled as two consecutive yaw angles need to be reasonably close. This is natively supported in the Open Motion Planning Library RRT* implementation [103] by supplying weights for different planner subspaces. Therefore, the yaw angle subspace is constructed with lower weight.

The second concern is related to the distance between the sampled payload configurations. Imagine an example with two payload waypoints, significantly separated spatially and rotated by a significant angle. A trajectory between those two waypoints is a straight line regarding the payload. However, this does not guarantee the task constraint where multiple aerial manipula-

tors need to achieve specific end-effector configurations to successfully transport the payload along the full trajectory. Therefore, the initial payload path $\mathscr{P}_{\mathrm{P}}$ is interpolated with Bezier curves to smooth it out and obtain a set of waypoints close to each other, noted with $\mathscr{P}_{\mathrm{P},s}$ with a total of $n_{p,s}$ waypoints. Note that the Open Motion Planning Library offers this option within its implementation.

Finally, the smooth payload path is transformed to the full system configuration. Each waypoint of the smooth path $\mathscr{P}_{\mathrm{P},s}$ is expressed in the world coordinate system. Based on that, each manipulator configuration ${}_i\mathbf{q}_{\mathrm{M}}$ is obtained through method explained in previous sections. The direct kinematics of the manipulator are used to determine the transformation between the manipulator base and end-effector ${}_i\mathbf{T}_0^{\mathrm{T}}$. Equation 6.2 is used to determine each multirotor body configuration ${}_i\mathbf{q}_{\mathrm{B}}$, required to construct the full state waypoint $\mathbf{w}$, as defined with equation 6.4. This allows writing the full state path as:

$$\mathscr{P} = \left\{ \mathbf{w}_j \mid j \in (1,2,\ldots,n_p), \mathbf{w}_j \in \mathbb{R}^{6\cdot n_r + \sum_{i=1}^{n_r} n_{\mathrm{M},i}} \right\}. \tag{6.8}$$

Each full state waypoint consists of the 6-DoF base, which can be fixed or floating, and a manipulator with $n_{\mathrm{M},i}$ DoFs. Therefore, the overall dimension of each waypoint n_w is the sum of all involved degrees of freedom.

Trajectory Planning

Having a path as a set of consecutive high dimensional waypoints is a prerequisite for the trajectory planning. As stated earlier, the trajectory is planned using the TOPP-RA approach. Apart from the planned path, the algorithm requires dynamical constraints in terms of the velocity and acceleration for each degree of freedom, as described in Section 4.3. Recall, the output is a time discretized trajectory as defined by equation 4.8:

$$\mathscr{T}_{\mathrm{D}} = \left\{ \mathbf{x}(kT_s) \mid \mathbf{x}(kT_s) \in \mathbb{R}^{3\cdot n_w}, k \in (0,\ldots,n_t) \right\}, \tag{6.9}$$

where n_t is the number of discretized points. Each trajectory point consists of position, velocity and acceleration, which increase its dimension when compared to path waypoints. Since this trajectory is sampled with the time period T_s, it can be executed by the vehicle in simulation or real world environment. Note since each trajectory point $\mathbf{x}$ contains multi-robot system degrees of freedom, it needs to be properly distributed to each agent in the implementation. This trajectory is then executed by the dynamical model to obtain the full state of each aerial manipulator, and the executed trajectory is used to obtain the corrected trajectory:

$$\mathscr{T}_{\mathrm{C}} = \left\{ \mathbf{x}(kT_s) \mid \mathbf{x}(kT_s) \in \mathbb{R}^{3\cdot n_w}, k \in (0,\ldots,n_t) \right\}. \tag{6.10}$$

It can be observed that the number of trajectory points is the same for the initially planned

trajectory $\mathscr{T}_{\mathrm{D}}$ and the corrected trajectory above. This is due to applying the manipulator corrections at each trajectory point. Recall that applying the manipulator corrections resolves the deviations in end-effector trajectory induced by the underactuated nature of co-planar multirotors. The same method as in Section 5.1.1.2 is employed for each aerial manipulator separately, yielding the corrected full state trajectory of the overall system.

6.2 Simulation

All simulation trials within this chapter are conducted in the Gazebo environment, which is described in Section 5.1.1.1. The simulations are conducted on different aerial manipulators, sharing a common multirotor body with various serial chain manipulators. The objective of the simulations is to transport a rod payload with two aerial manipulators, which is conducted on both homogeneous and heterogeneous systems. The employed aerial manipulators are described later within this section. To consistently evaluate different aerial manipulator pairs, the same set of trajectories is executed and the performance analysis is conducted afterwards.

6.2.1 Performance Indicators

When transporting a payload with two aerial manipulators, it is essential to always keep the relative end-effector configurations close to the planned trajectory. This is important to properly secure the payload during transportation. Naturally, some deviations in both position and orientation are allowed, however, too large deviations have the possibility to result in dropping the carried object. Instead on focusing on how well does each multirotor and end-effector track the desired trajectory, their relative transformation is observed. Furthermore, to assess the performance of the planned and executed trajectory, only two end-effectors are considered in this analysis.

Detailed illustration of the transforms required to obtain the difference between the two end-effectors is depicted on Fig. 6.4. The relative transform between two end-effectors can be expressed as:

$$\mathbf{T}_{\mathrm{T}_1}^{\mathrm{T}_2} = \left(\mathbf{T}_{\mathrm{W}}^{\mathrm{T}_1}\right)^{-1} \cdot \mathbf{T}_{\mathrm{W}}^{\mathrm{T}_2}, \tag{6.11}$$

where $\mathbf{T}_{\mathrm{W}}^{\mathrm{T}_1}$ and $\mathbf{T}_{\mathrm{W}}^{\mathrm{T}_2}$ are the first and second end-effector configurations expressed in the world frame. The relative transform can be obtained for both planned and executed trajectories. Concretely, there are three trajectories of interest: planned trajectory $\mathscr{T}_p$; the trajectory executed by the model in simulation $\mathscr{T}_{me}$; and the trajectory executed after applying the end-effector corrections $\mathscr{T}_e$. The corresponding relative transforms of the end-effectors are denoted with same subscripts as trajectories: ${}^{p}\mathbf{T}_{\mathrm{T}_1}^{\mathrm{T}_2}$, ${}^{me}\mathbf{T}_{\mathrm{T}_1}^{\mathrm{T}_2}$ and ${}^{e}\mathbf{T}_{\mathrm{T}_1}^{\mathrm{T}_2}$, respectively. To compare different payload tra-

jectories, these relative transforms are computed for each sample of the planned and executed trajectories.

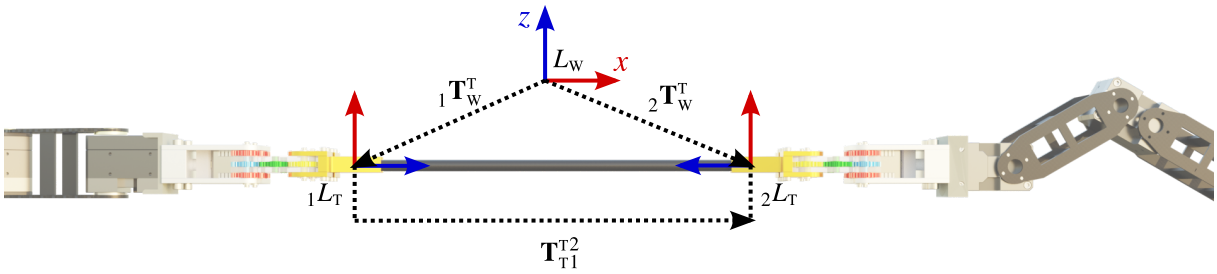


Figure 6.4: Illustration of obtaining the transform between two end-effectors.

For comparing two homogeneous transform matrices, the following formula can be utilized:

$$\mathbf{T}_{a,b} = \mathbf{T}_a \cdot (\mathbf{T}_b)^{-1}, \tag{6.12}$$

where $\mathbf{T}_a$, $\mathbf{T}_b$ and $\mathbf{T}_{a,b}$ are generic homogeneous transform matrices. Intuitively, if $\mathbf{T}_a \equiv \mathbf{T}_b$ holds, the resultant matrix $\mathbf{T}_{a,b}$ is the identity matrix. If these two matrices differ, both translation and rotation can be separately observed. The difference in translation can be directly extracted from the resultant matrix:

$$\Delta\mathbf{p}_{a,b} = \mathbf{p}_a - \mathbf{p}_b = \begin{bmatrix} \Delta x & \Delta y & \Delta z \end{bmatrix}^T, \tag{6.13}$$

where Δx, Δy and Δz capture the deviation in the respective axes. To obtain the orientation error, the following relation is used:

$$\Delta\lambda = \arccos \frac{\mathrm{tr}\left(\mathbf{R}_{a,b}\right) - 1}{2}, \tag{6.14}$$

where $\mathrm{tr}\left(\mathbf{R}_{a,b}\right)$ is the trace of matrix, and $\Delta\lambda$ is the difference angle between the two rotation matrices. This representation of the difference between two rotations is particularly useful since it is a scalar and can be easily compared with other rotation errors.

The objective of this analysis is to compare the initial trajectory executed by the model and the corrected trajectory executed by the aerial manipulator against the planned trajectory. This comparison reveals the influence of corrections for two aerial manipulators transporting a payload. Namely, the following errors are observed:

$$\begin{aligned} \Delta\mathbf{T}_{p,me} &= {}^{p}\mathbf{T}^{T2}_{T1} \cdot \left({}^{me}\mathbf{T}^{T2}_{T1}\right)^{-1} \\ \Delta\mathbf{T}_{p,e} &= {}^{p}\mathbf{T}^{T2}_{T1} \cdot \left({}^{e}\mathbf{T}^{T2}_{T1}\right)^{-1}, \end{aligned} \tag{6.15}$$

where $\Delta\mathbf{T}_{p,me}$ is the payload deviation for the model executed trajectory, and $\Delta\mathbf{T}_{p,e}$ captures the

deviation for trajectory executed after applying corrections. This approach allows to directly compare the position and orientation differences for trajectories executed with and without applying corrections based on the dynamical model. The compared indicators are the norm of the position deviation $|\Delta\mathbf{p}|$, and the angle difference $\Delta\lambda$. Since these errors are determined for each point of both executed trajectories, the mean and maximum deviations are reported for each simulation trial.

Note that within this analysis, the tracking performance of the multirotor and manipulator is not observed. Only errors in the difference between the two end-effectors are observed, because these ultimately determine whether the payload can be transported or not. However, errors in the overall system trajectory tracking are embedded in the aforementioned performance indicators. Furthermore, to successfully transport the payload, both the position deviation $|\Delta\mathbf{p}|$ and the angle difference $\Delta\lambda$ errors need to be small. Intuitively, if the position deviation is high, the end-effectors are too far apart to transport the payload. On the other hand, if the orientation error is high, the transported payload is not properly secured by the end-effectors which can cause the object to slip.

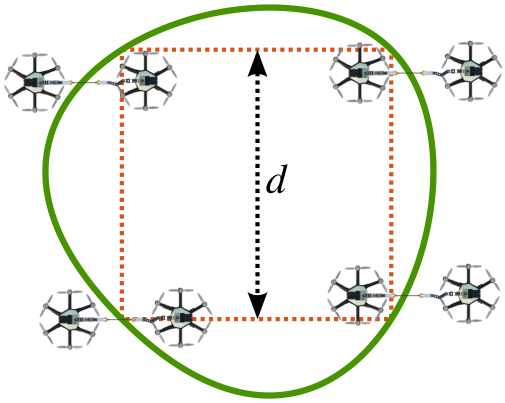
6.2.1.1 Benchmark Trajectories

As there are multiple combinations of different aerial manipulators performing the payload transportation task, four different trajectory types are executed by each system. Each trajectory type is executed for $n = 20$ times, providing the mean and maximum end-effector position and orientation deviations. This allows to directly compare different systems and determine which are better suited for different trajectories. The planned trajectory types are depicted on Fig. 6.5.

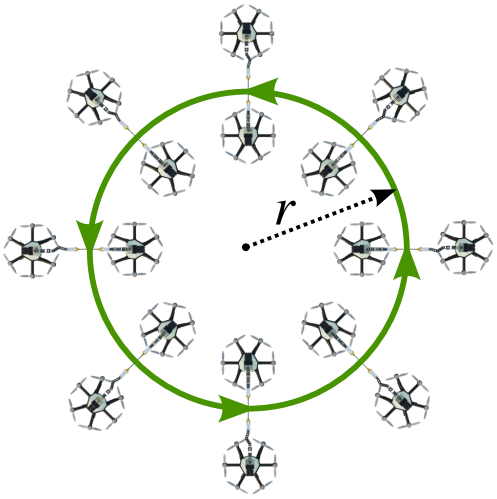
Straight line trajectory. The first planned trajectory is a simple straight line along the x_{W} world axis. The distance between the waypoints is $d = 2m$, and a single trial is performed by planning a trajectory moving forward and back. This trajectory mostly induces rotation around the y_{B} axis, pitching the multirotor and dominantly displacing the end-effector along the z_{W} axis. The orientation of the payload is constant for this trajectory.

Square trajectory. The square trajectory consists of five waypoints, returning the system to the initial set point. Similar to the straight trajectory, the square side has the length of $d = 2m$. Since the TOPP-RA algorithm generates time optimal trajectories, the final trajectory passes through square vertices while deviating from the sides. The payload orientation is kept constant for this trajectory.

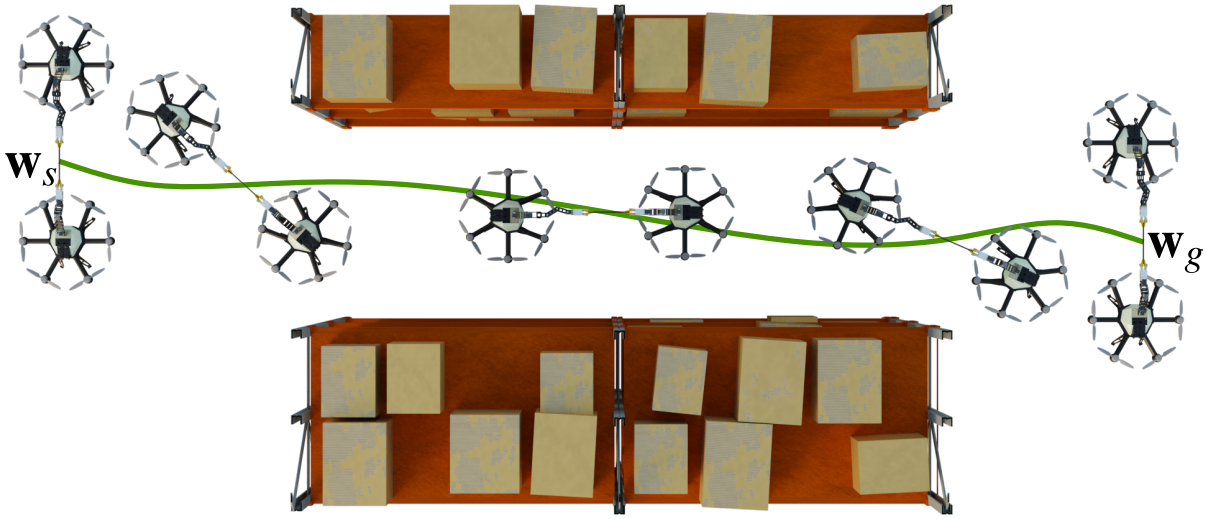
Circular trajectory. The third trajectory is a circle, with the payload orientation changing to point towards the circle center at all times. The circle radius is set to $r = 2m$. This trajectory



(a) Square trajectory with side length $d = 2m$.



(b) Circular trajectory with radius $r = 2m$.



(c) Trajectory the warehouse environment featuring a narrow corridor that forces the planner to rotate the system.

Figure 6.5: Benchmark trajectories performed by heterogeneous aerial manipulator teams.

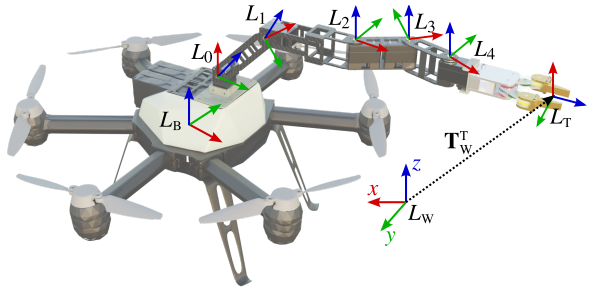
tests both position and orientation deviation from the planned payload configuration.

Warehouse trajectory. The first three trajectories are planned in an empty space. The final trajectory is planned in the environment with a narrow corridor. The objective of the planner is to find a path and trajectory that changes the orientation, allowing the system to pass through the narrow corridor. The initial and final configurations are perpendicular to the corridor, forcing the planner to rotate the payload.

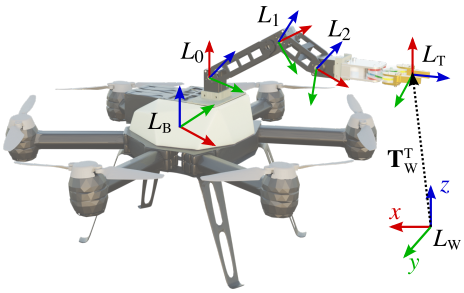
6.2.2 Aerial Manipulator Types

To thoroughly test the motion planning procedure for multiple manipulators, several distinct aerial manipulators are used and described below. Various combinations of these manipulators are chosen to perform the benchmark trajectories and compare them against each other. The

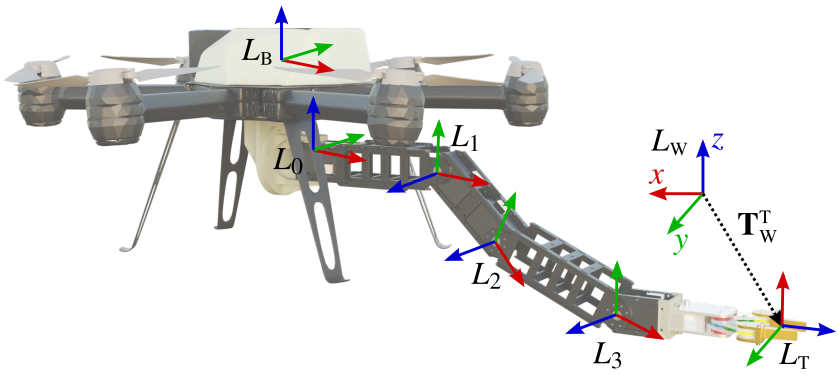
aerial manipulators used in simulation are depicted in Fig. 6.6.



(a) Multirotor with 5-DoF manipulator.



(b) Multirotor with 3-DoF manipulator.



(c) Multirotor with 4-DoF manipulator.

Figure 6.6: Aerial manipulators used in the simulation with indicated joint rotation axes.

6.2.2.1 5-DoF Aerial Manipulator

The first aerial manipulator has already been used within Section 5.2.1. It features a lightweight 5-DoF manipulator attached above the quadrotor body center of mass. Recall the transform between the multirotor body and the manipulator:

$$\mathbf{T}_B^0 = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0.075 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.16}$$

Furthermore, the manipulator DH parameters are given in Table 6.1, and the full aerial manipulator is depicted in Fig. 6.6a. The d_5 parameter differs from the one used in the previous chapter, as the manipulator end-effector is considered to be at the attached tool. In the previous chapter, this tool was a long rod that increased the d_5 parameter according to its length.

Determining the manipulator configuration from the specified payload transform is provided as an example in Section 6.1.1.1. The determined configuration is provided in equation (6.5). The dynamical constraints imposed on the aerial manipulator are summarized in table 6.2. This choice is based on the analysis performed in Section 5.2.1.1, where numerous multirotor body

Table 6.1: DH parameters of the 5-DoF manipulator attached to the multirotor in the simulation. Note that last joint 5* is virtual and is required to properly align the end-effector coordinate system.

	θ_k	d_k	α_k	a_k
1	0	0	0	$0.1225m$
2	0	0	$-\pi/2$	$0.1365m$
3	0	0	0	$0.0755m$
4	0	0	0	$0.0725m$
5	$\pi/2$	0	$\pi/2$	0
5*	0	$0.048m$	0	0

velocity-acceleration pairs are evaluated against the overall end-effector deviation. The chosen parameters are out of the quasi-static domain, while retaining a low end-effector deviation.

Table 6.2: Trajectory planning constraints set for the multirotor with 5-DoF aerial manipulator. The units for linear axes x, y and z are expressed in m/s for velocity and m/s^2 for acceleration. The angular DoFs' units are expressed in rad/s for velocity and rad/s^2 for acceleration.

	x	y	z	ψ	q_1	q_2	q_3	q_4	q_5
v_{max}	0.75	0.75	0.5	0.75	1.2	1.2	1.2	1.2	1.2
a_{max}	0.5	0.5	0.5	0.5	1.2	1.2	1.2	1.2	1.2

6.2.2.2 4-DoF Aerial Manipulator

The 4-DoF manipulator used in this analysis has the first joint rotating around the body z_B axis, while the other joints rotate around y_B, as shown on Fig. 6.6c. Similar to the previous manipulator, it is constructed of lightweight links with slightly different servo motor models. The manipulator is attached below the body center of mass, with the following transform:

$$\mathbf{T}_B^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.17}$$

To obtain the transform between the manipulator base and the end-effector, the DH parameters are determined in table 6.3.
Furthermore, the dynamical constraints are chosen on the same principle as in the previous section, outlined in table 6.4.

To determine the required end-effector transform in the world frame, based on the payload transform in the world, equation 6.1 is employed. The first step is to decompose this transform into euler angles, which can be directly used to determine the manipulator configuration $\mathbf{q}_M$. As the first degree of freedom q_1 rotates around the multirotor body z_B axis, it is considered to be zero at all times $q_1 = 0$. Since the remaining DoFs operate around a common axis, the overall

Table 6.3: DH parameters of the 4-DoF manipulator attached to the multirotor in the simulation. Note that last joint 4* is virtual and is required to properly align the end-effector coordinate system.

	θ_k	d_k	α_k	a_k
1	0	0	$\pi/2$	$0.1796m$
2	0	0	0	$0.1015m$
3	0	0	0	$0.1510m$
4	$\pi/2$	0	$\pi/2$	0
4*	0	$0.048m$	0	0

Table 6.4: Trajectory planning constraints set for the multirotor with 4-DoF aerial manipulator. The units for linear axes x, y and z are expressed in m/s for velocity and m/s^2 for acceleration. The angular DoFs' units are expressed in rad/s for velocity and rad/s^2 for acceleration.

	x	y	z	ψ	q_1	q_2	q_3	q_4
v_{max}	0.75	0.75	0.5	0.75	1.2	1.2	1.2	1.2
a_{max}	0.5	0.5	0.5	0.5	1.2	1.2	1.2	1.2

end-effector yaw angle is achieved by the multirotor body. Note that when applying corrections, the first DoF is still used to minimize the end-effector configuration error.

As already mentioned, the remaining degrees of freedom rotate around parallel axes, and are suitable for ensuring the proper pitch angle of the end-effector θ_{T}. To determine the manipulator configuration for the specified end-effector pitch angle, a workspace analysis is performed, based on the analysis from Section 5.3.2.2. To match the required end-effector angle, a following relationship can be written:

$$q_4 = \theta_{\mathrm{T}} - q_2 - q_3. \tag{6.18}$$

This dependency performing the workspace analysis only with joints q_2 and q_3. Three values are then constructed, as in the previous analysis, consisting of dexterity as the reduced Jacobian product determinant $\mathfrak{D}(q_2,q_3) = \left|\mathbf{J}^T \cdot \mathbf{J}\right|$ defined in equation (5.24), reach $\mathfrak{R}(q_2,q_3) = (\mathbf{p}_1^{\mathrm{T}})^T \cdot \mathbf{p}_1^{\mathrm{T}}$ defined in equation (5.26), and limit $\mathfrak{L}(q_2,q_3)$ defined in equation (5.27). Since the first joint rotation is determined in the former paragraph, the analysis does not include it, hence the end-effector position is observed from the second manipulator joint coordinate system L_1. The Jacobian matrix is also slightly different:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathbf{p}_1^{\mathrm{T}}}{\partial q_2} & \frac{\partial \mathbf{p}_1^{\mathrm{T}}}{\partial q_3} \end{bmatrix}. \tag{6.19}$$

Combining the dexterity, reach and limit yields the complete manifold $\mathfrak{M}(q_2,q_3) = \mathfrak{D}(q_2,q_3) \cdot \mathfrak{R}(q_2,q_3) \cdot \mathfrak{L}(q_2,q_3)$, shown of Fig. 6.7 for $\theta_{\mathrm{T}} = 0$. A similar line of reasoning behind this analysis is employed here. To apply the corrections in the planning phase, it is beneficial to have the manipulator as far away as possible from its limits, while maintaining maximum reach to correct larger errors, and to remain in the null space as determined by dexterity. The maximum

of this manifold determines the optimal manipulator configuration for a given end-effector pitch angle θ_T. The analysis from Section 5.3.2.2 is extended to the full manipulator workspace with $\theta_T \in [-\pi/2, \pi/2]$. The obtained results are shown on Fig. 6.8. The figure shows the analysis results alongside a line approximation for each joint. Although the curves obtained from the analysis are not lines, the deviation from the line approximation is very small, making it appropriate to use in the implementation. Furthermore, this approximation offers a good trade off for the computing time, as determining the optimal manipulator state in the correction phase would greatly increase the planning time.

The second concern that needs to be addressed is the discontinuity around $\theta_T = 0$. This discontinuity is occurs due to the redundant configuration of the manipulator. The obtained manifold for $\theta_T = 0$ has two equivalent maximum values corresponding to the redundant configuration, as shown on Fig. 6.7. Note that this surface is very similar to the 3-DoF manipulator workspace analysis depicted on Fig. 5.14a due to the similar manipulator construction. The main issue with the discontinuity is the rapid switch from one to the other redundant configuration. While executing this switch, there is no guarantee that the desired end-effector orientation will be preserved. Therefore, a sub-optimal approach is devised for the two possible manipulator attachment points. If the manipulator is attached below the center of mass ($z_B^0 < 0$), there is going to be a maximum end-effector pitch it can achieve without colliding with propellers. Conversely, if it is attached above the center of mass ($z_B^0 > 0$), a minimum end-effector pitch angle defines the limit. In such a case, a sub-optimal configuration corresponding to the lower peak of the manifold $\mathfrak{M}$ is chosen. This way, the initial constraint from equation (6.18) is conserved, while the manipulator configuration is still far from its limit with enough reach and

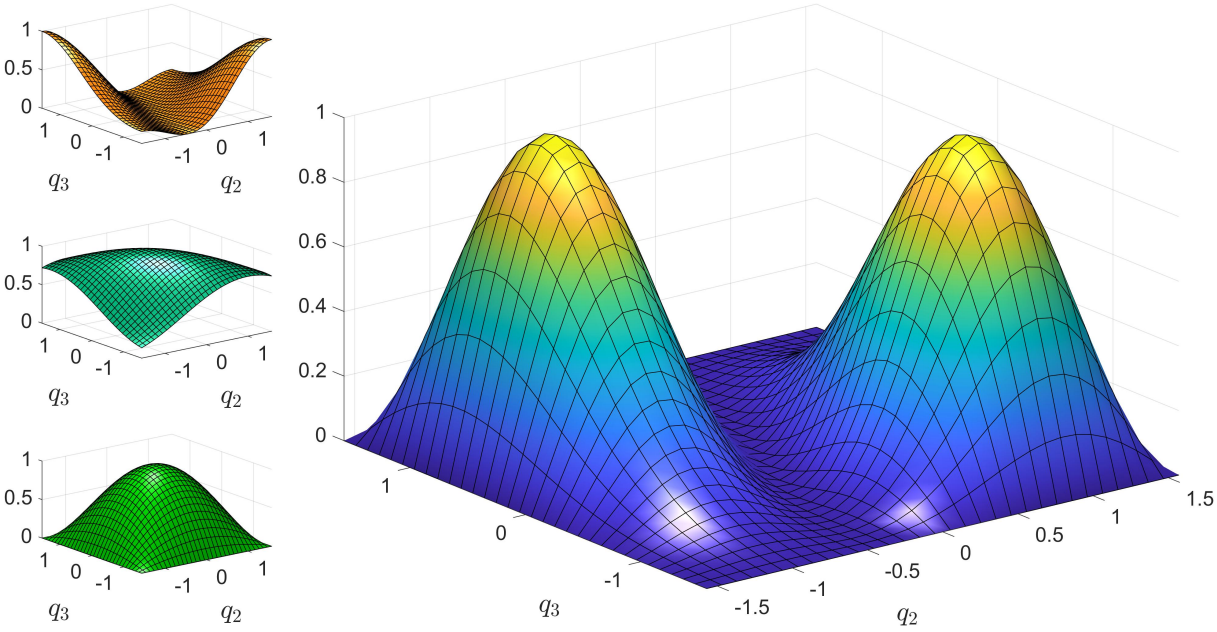


Figure 6.7: The 4-DoF manipulator workspace analysis for $\theta_T = 0°$. The combined manifold $\mathfrak{M}$ is shown on the right, while the left column depicts dexterity $\mathfrak{D}$, reach $\mathfrak{R}$ and limit $\mathfrak{L}$, respectively.

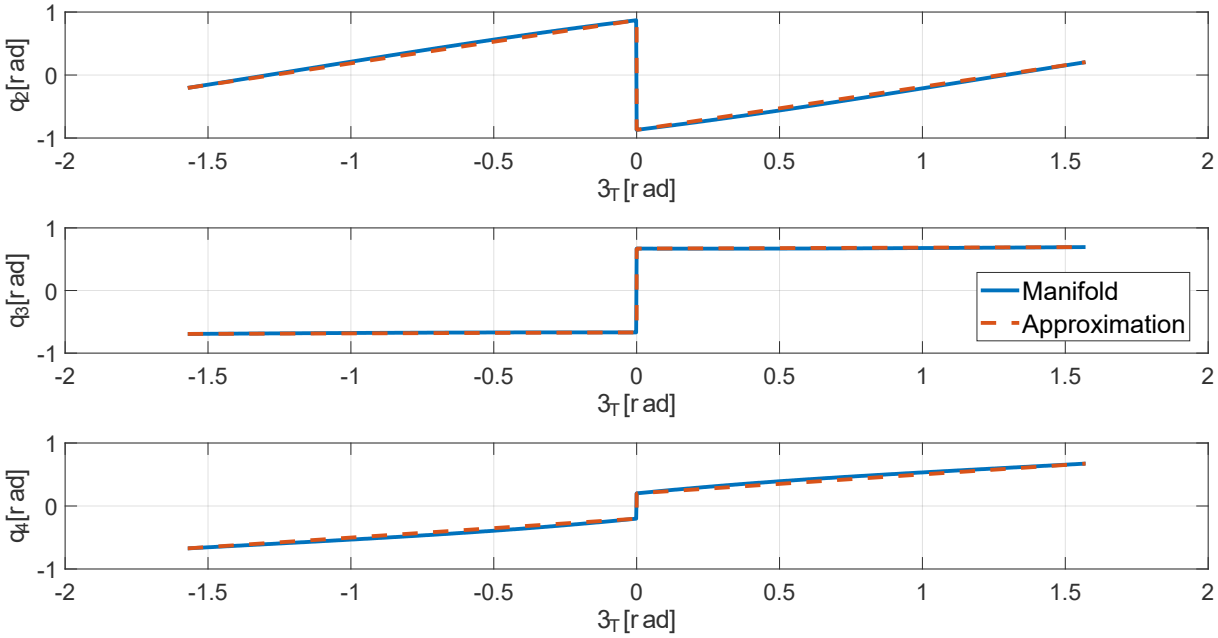


Figure 6.8: The optimal manipulator configuration with respect to the required end-effector pitch θ_T. The discontinuity around the center is caused by the redundant configurations of the manipulator.

dexterity to correct the end-effector motion properly. An example of such a configuration with $\theta_{T,max} = -\theta_{T,min} = 0.5rad$ is depicted on Fig. 6.9.

The determined manipulator state based on the required end-effector pitch is:

$$\mathbf{q}_M(\theta_T) = \begin{bmatrix} 0 & q_2(\theta_T) & q_3(\theta_T) & q_4(\theta_T) \end{bmatrix}^T, \tag{6.20}$$

where functions $q_2(\theta_T)$, $q_3(\theta_T)$ and $q_4(\theta_T)$ are determined from the line equations depicted on Fig. 6.9.

6.2.2.3 3-DoF Aerial Manipulator

The next manipulator used in simulation has 3 degrees of freedom all rotating around the body y_B axis, as shown on Fig. 6.6b. The manipulator is constructed of lightweight links with servo joints, and it is attached above the vehicle center of mass with the following transform:

$$\mathbf{T}_B^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6.21}$$

The DH parameters required for the direct kinematics are summarized in table 6.5. Furthermore, the dynamical constraints used in this simulation are shown in table 6.6.

The workspace analysis for this particular manipulator is performed within Section 5.3.2.2. However, there it is limited to only two end-effector pitch angles θ_T. This analysis is expanded

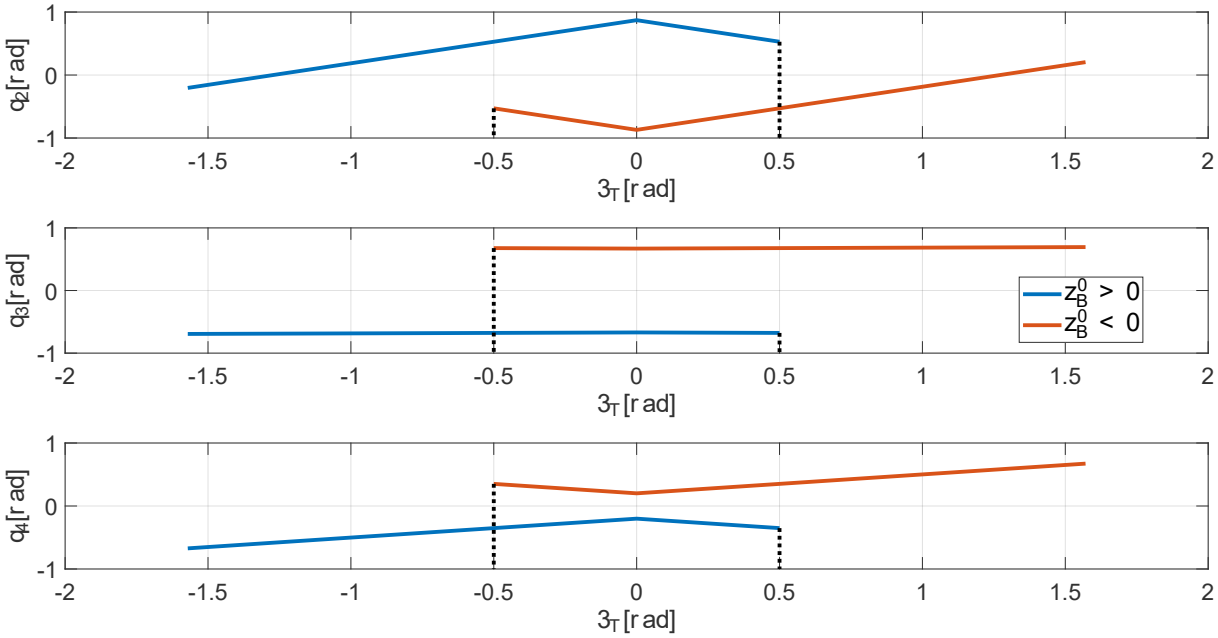


Figure 6.9: A sub-optimal manipulator configuration eliminating the discontinuity at the $\theta_{\mathrm{T}} = 0$. The blue depicts the manipulator mounted above the body center of gravity with maximum end-effector pitch $\theta_{\mathrm{T},max} = 0.5rad$, while the red shows the configuration for mounting the manipulator below the body center of mass with minimum pitch $\theta_{\mathrm{T},max} = -0.5rad$. The maximum and minimum end-effector pitch are determined by the aerial manipulator construction.

Table 6.5: DH parameters of the 3-DoF manipulator attached to the aerial manipulator in simulation. Note that there is a virtual joint 3* required to properly align the end-effector coordinate system. Careful reader will notice that first two links are, by design, the same length.

	θ_k	d_k	α_k	a_k
1	0	0	0	$0.0755m$
2	0	0	0	$0.0755m$
3	$\pi/2$	0	$\pi/2$	0
3*	0	0.048	0	0

according to the procedure outlined for the 4-DoF manipulator, performed in the previous section. Recalling the results obtained there, the main issue is the discontinuity around the $\theta_{\mathrm{T}} = 0$, which corresponds to the redundant manipulator configurations that is a direct consequence of three joints rotating around parallel axes. As the procedure to obtain the sub-optimal manipulator configuration is the same as in the previous section, the details are omitted here. The final sub-optimal manipulator configuration that depends on the manipulator mounting point is shown on Fig. 6.10, and the manipulator joint values are calculated according to the following relation:

$$\mathbf{q}_{\mathrm{M}}(\theta_{\mathrm{T}}) = \begin{bmatrix} q_1(\theta_{\mathrm{T}}) & q_2(\theta_{\mathrm{T}}) & q_3(\theta_{\mathrm{T}}) \end{bmatrix}^T. \tag{6.22}$$

Table 6.6: Trajectory planning constraints set for the multirotor with 3-DoF aerial manipulator. The units for linear axes x, y and z are expressed in m/s for velocity and m/s^2 for acceleration. The angular DoFs' units are expressed in rad/s for velocity and rad/s^2 for acceleration.

	x	y	z	ψ	q_1	q_2	q_3
v_{max}	0.75	0.75	0.5	0.75	1.2	1.2	1.2
a_{max}	0.5	0.5	0.5	0.5	1.2	1.2	1.2

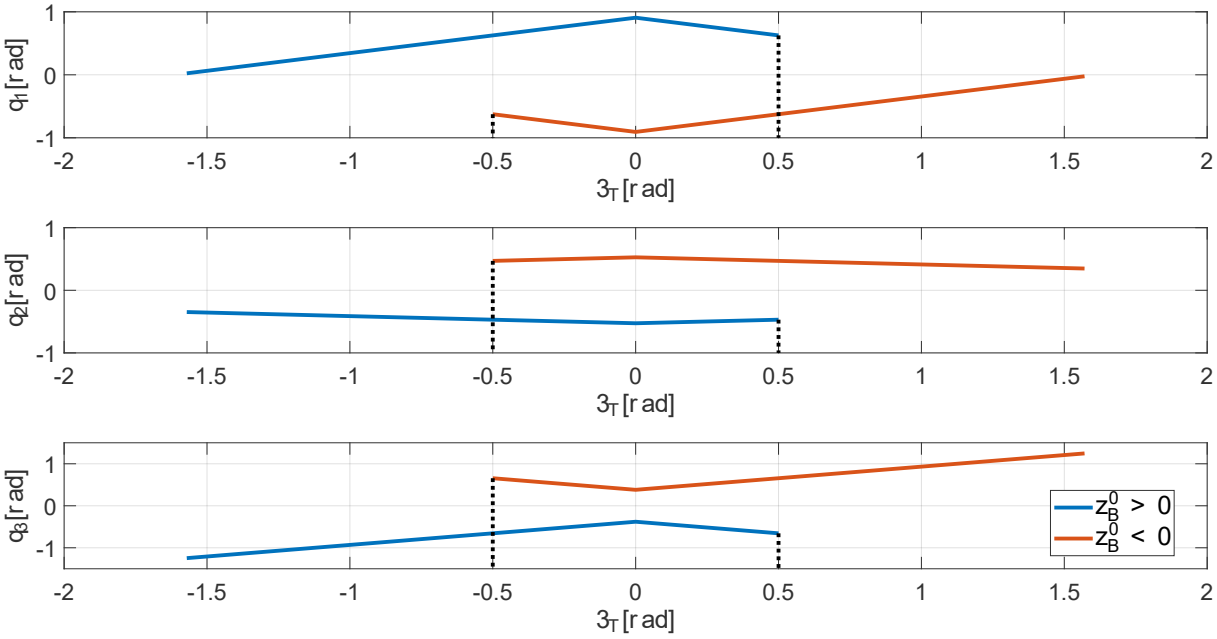


Figure 6.10: A sub-optimal 3-DoF manipulator configuration eliminating the discontinuity at the $\theta_T = 0$. The blue depicts the manipulator mounted above the body center of gravity with maximum end-effector pitch $\theta_{T,max} = 0.5rad$, while the red shows the configuration for mounting the manipulator below the body center of mass with minimum pitch $\theta_{T,max} = -0.5rad$. The maximum and minimum end-effector pitch are determined by the aerial manipulator construction.

6.2.3 Simulation Results

To recapitulate, the simulation trials are performed over four benchmark trajectories: straight, square, circle and warehouse, as described in Section 6.2.1.1. Each trajectory is executed for $n = 20$ times to obtain the average deviation from the reference. The former three trajectories are always planned exactly the same, in an empty environment. The warehouse trajectory is subject to the RRT* path planner, generating a different path for each trial. The transported payload is considered to be a rod of length $l = 0.3m$. Within this section, the payload is not coupled to the end-effectors, rather, the relative transform between the end-effectors is observed to determine whether it is possible to transport the payload after applying corrections to the trajectory.

To evaluate the model correction method, the planner is tested with all combinations of the previously described aerial manipulators. This gives an insight how different manipulator configurations work together and which motions covered within benchmark trajectories are suitable for performing corrections. Namely, four combinations are evaluated within this section:

- **5-DoF - 5-DoF** This combination features two 5-DoF manipulators both attached above the multirotor center of mass.
- **5-DoF - 4-DoF** The 5-DoF manipulator is attached above the center of mass, while the 4-DoF manipulator is below it.
- **5-DoF - 3-DoF** In this combination, both manipulators are attached above the multirotor center of mass.
- **3-DoF - 4-DoF** The final combination with 3-DoF manipulator attached above and the 4-DoF manipulator attached below the multirotor center of mass.

Furthermore, the objective of the simulation analysis is to determine the deviation of the relative end-effector transform from the planned one. The procedure is explained in detail within Section 6.2.1. In other words, the deviation can be thought of as observed from the coordinate system of the transported payload. Since the dynamical model is an integrated part of the planning method, two tracking errors are observed as stated in equation (6.15). Namely, $\Delta\mathbf{T}_{p,me}$ captures the deviation of the end-effector trajectory executed by the model, and $\Delta\mathbf{T}_{p,e}$ is the error after applying corrections. The position and orientation deviations are observed separately, with the position taken as the norm, while the orientation is computed according to equation (6.14). An example of a straight trajectory response for 5-DoF - 5-DoF system is given in Fig. 6.11, to give the reader an idea of the relative end-effector deviation. The main difference occurs in z axis, which is expected due to executing a straight trajectory. The errors in x and pitch are relatively small, so corrections do not play a vital role in these cases. Nevertheless, the decreased deviation in the z axis after applying correction suggests that the payload can be transported by applying corrections.

The previous figure shows a detailed example of comparison between the model executed and final trajectory for the straight trajectory example. Since there are four benchmark trajectories as introduced by Section 6.2.1.1, and four aerial manipulator combinations, the results are summarized in two figures. Fig. 6.12 shows the end-effector deviation in terms of the position vector norm from equation (6.13), and the orientation from equation (6.14). The errors on this figure are shown for the trajectory executed by the model, before applying corrections. All combinations of the aerial manipulators are shown for each benchmark trajectory. The perfect tracking is achieved if both errors are zero, and the objective of later corrections is to minimize this error. The position error $\left|\mathbf{p}_{p,me}\right|$ is similar for all four benchmark trajectories. The warehouse trajectory stands out due to its long narrow corridor where the system slowly navigates forward, yielding a small position error between $t = 7s$ and $t = 17s$. Due to the rotation around z axis, the error is increased before and after the narrow corridor. The orientation error for both straight and square trajectories is similar for all combinations. When it comes to the circular trajectory, the orientation error increases due to constant rotation around the z axis, which also laterally displaces the end-effectors. The orientation error in the warehouse is subject to the

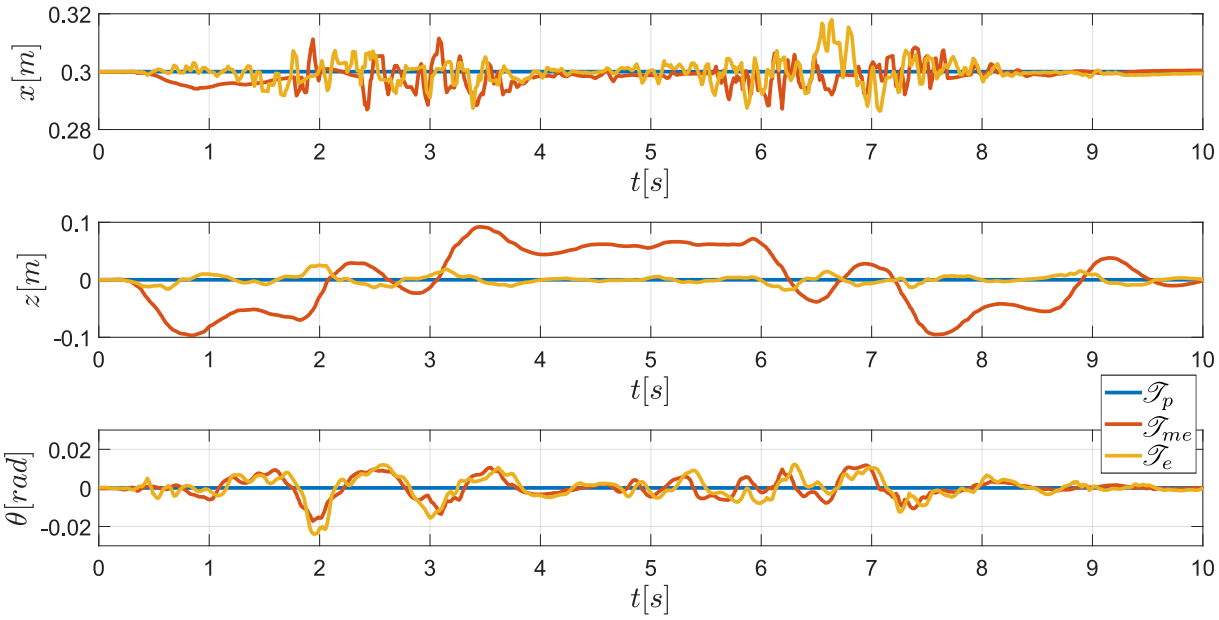


Figure 6.11: A relative end-effector deviation for a straight trajectory performed by the 5DoF-5DoF system. Since a straight trajectory is performed, the error in y axis, as well as in the roll and yaw angles are negligible and therefore omitted. The planned transform between the two end-effectors ${}^{p}\mathbf{T}_{\mathrm{T}_1}^{\mathrm{T}_2}$ corresponds to the transported payload size which is $0.3m$ along the x axis.

random sampling performed by the RRT* algorithm, and it peaks the highest at the beginning of the trajectory.

The set of trajectories executed by the model shows very similar tracking errors in both position and orientation. Applying corrections, however, is a different story. Each manipulator has a different correction capabilities that are defined by its null space. Intuitively, the lower number of DoFs can be expected to have a more limited null space, making it harder to minimize the error by performing corrections. The results are shown on Fig. 6.13. In general, it can be observed that the position and orientation errors are reduced to some extent. Namely, the trends for the 5DoF-5DoF and 5DoF-4DoF are roughly the same. Both position and orientation errors are significantly reduced across all benchmark trajectories, when compared to the model executed counterparts. This is the expected outcome of applying the model corrections to the trajectory. On the other hand, 5DoF-3DoF and 3DoF-4DoF configurations show a different results. For straight and square trajectories, the overall position tracking is reduced. The orientation error peaks at some instances. This happens when the inverse kinematics of the manipulator tries to find a manipulator configuration that minimizes both position and orientation error, which is impossible to achieve due to the manipulator configuration. The result is a peak in orientation error with reduced position error. The position error in the circular trajectory is reduced at the expense of the increased orientation error. This trajectory requires a lateral movement while rotating around the body z axis. The high orientation error arises due to the 3-DoF manipulator construction with all joints rotating around the common axis. Therefore, the error in yaw angle cannot be reduced by this manipulator. A similar effect happens in the beginning and end of the

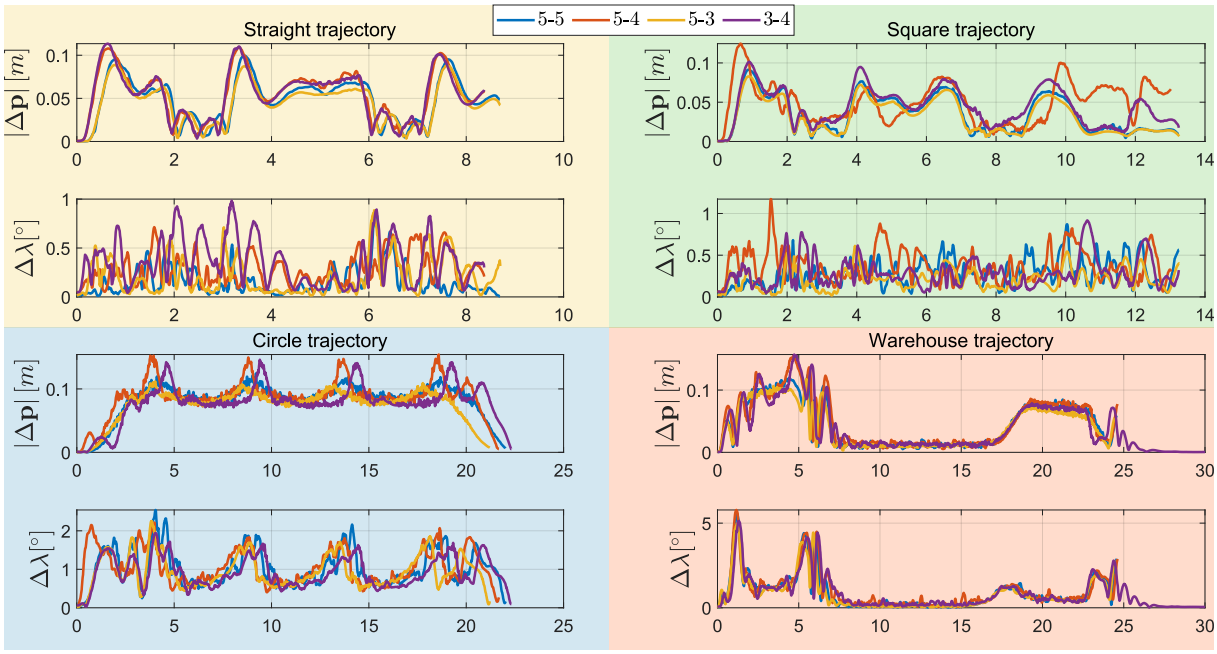


Figure 6.12: Position and orientation errors of the four benchmark trajectories executed by models of four aerial manipulator combinations. Each benchmark trajectory is planned in the payload configuration, making it possible to achieve the same payload waypoints across different configurations. This allows for directly observing and comparing the errors. The perfect trajectory tracking is achieved when both position and orientation errors are zero.

warehouse trajectory, where the system rotates to enter and exit the narrow corridor. Although having less degrees of freedom, the 3-DoF manipulator is not suitable for minimizing errors induced by multirotor yaw rotations. One way to alleviate this problem is to reduce the maximum velocity and acceleration in the trajectory planning. Although this has the possibility to reduce the orientation error, it inevitably increases the trajectory duration, which might compromise the mission due to a flight time limitation.

As stated earlier, all four benchmark trajectories are executed for $n = 20$ trials for each aerial manipulator combination. The overall results are summarized on Fig. 6.14. The figure compares position and orientation errors between the initially planned trajectory executed by the model and the executed trajectory after applying corrections. The errors are shown in terms of mean and maximum deviation from the planned trajectory. The position errors, Fig. 6.14a, is consistently reduced across all aerial manipulator combinations and planned trajectory types. Similar results are achieved for straight and square trajectory, which do not change the transported payload orientation. The position errors for circular trajectory are a bit higher, which is somewhat expected due to the lateral system movement and rotation around the z axis to keep the payload facing the circle center. The path for the warehouse trajectories is planned using the RRT* algorithm, and consequently has the highest maximum error. In this case, each trajectory has a different underlying path. Note that the maximum error of the model executed trajectory depends on the manipulator reach, which is defined by the manipulator link sizes and config-

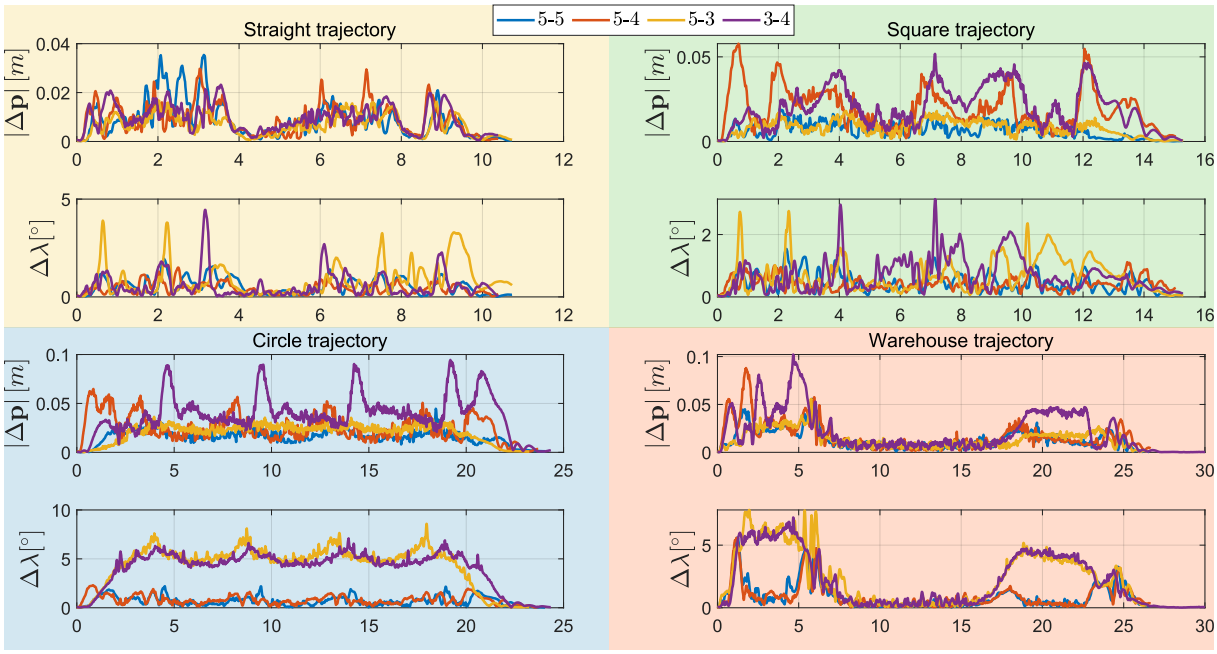
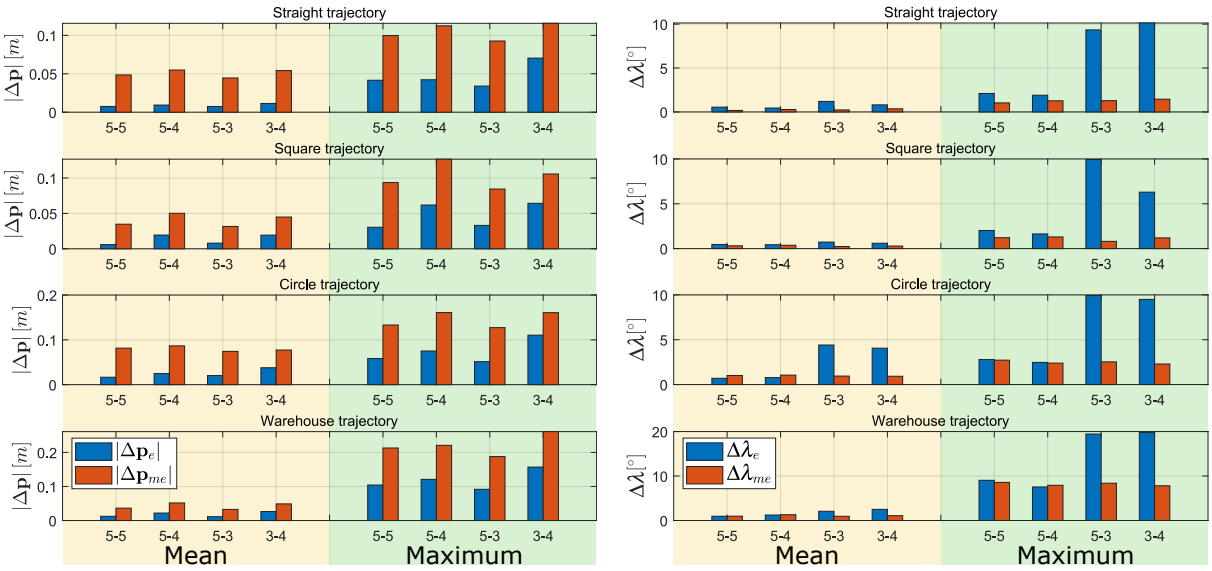


Figure 6.13: Position and orientation errors of the four benchmark trajectories, corrected and executed by four aerial manipulator combinations. Each benchmark trajectory is planned in the payload configuration, making it possible to achieve the same payload waypoints across different configurations. This allows for directly observing and comparing the errors. The perfect trajectory tracking is achieved when both position and orientation errors are zero.

uration $\mathbf{q}_{\mathrm{M}}$. This yields a slightly lower maximum error for the combinations with the 3-DoF manipulator.

The orientation error, Fig. 6.14b, for the straight and square trajectories shows very similar results. However, a counter intuitive effect takes place here. The overall error after applying corrections is actually higher than for the initial trajectory. The low orientation error for the model executed trajectory happens because both multirotors pitch in the same direction. On the straight trajectory example, the end-effectors are facing one another in this case, although, they are displaced along the z axis which requires corrections. The effect is most pronounced for maximum deviations in combinations with the 3-DoF manipulator. As mentioned earlier, all joints of this manipulator are rotating around parallel axes, and sometimes the inverse kinematics struggles to find a solution that satisfies both the position and orientation of the end-effector. An approximate solution trades off these errors, yielding peaks in orientation. Note that this happens for all four benchmark trajectories, yielding a high maximum error when the 3-DoF aerial manipulator is employed. The circle trajectory shows an increased mean error for the combinations with the 3-DoF manipulator, which is expected because this manipulator cannot correct the orientation errors around the z axis due to the system executing yaw motion. The overall mean errors in orientation are relatively low, except for the circle trajectory with the 3-DoF manipulator, which proves the effectiveness of the correction method. However, the high peaks sometimes occurring with the 3-DoF manipulator suggest that this configuration is not



(a) Position error bar graph.

(b) Orientation error bar graph.

Figure 6.14: Position and orientation errors for all four aerial manipulator combinations across all benchmark trajectories. The red bars show the error of the model executed trajectory, while the blue bars capture the executed trajectory error after applying corrections. The left portion of each chart shows the mean value across $n = 20$ trajectories, while the right portion shows the maximum value.

suitable to use in a transportation task using a heterogeneous multi robot system.

To successfully transport an object, both position and orientation errors need to be low. In case of a high position error, the end-effector displacement can exert forces pulling on both manipulators, assuming a secure connection of the transported payload. This can result in the manipulator, or the object itself, colliding with propellers yielding a crash and compromising the transportation task. Similar can happen with a large orientation error, when the end-effectors are not facing each other. The main objective of this analysis is to determine the differences between various aerial manipulator combinations. The combinations with the 3-DoF manipulator experience higher peaks in the orientation error, reaching up to 10°. This relatively high error can dislodge the transported payload, rendering the 3-DoF manipulator less suitable for transportation tasks. The other two combinations, 5DoF-4DoF and 5DoF-5DoF, show promising reduction in both the position and orientation errors, making them good candidates for object transportation tasks. The complete results obtained by this analysis, in terms of mean, maximum and median errors, are summarized in Table 6.7.

Table 6.7: This table summarizes the results obtained by performing the benchmark trajectories with various combinations of aerial manipulators.

	Ind	Mean				Maximum				Median			
		5-5	5-4	5-3	3-4	5-5	5-4	5-3	3-4	5-5	5-4	5-3	3-4
Straight	$\lvert\mathbf{p}_{p,me}\rvert$	0.0485	0.0549	0.0445	0.0541	0.0997	0.1124	0.0926	0.1157	0.0537	0.0593	0.0492	0.0570
	$\lvert\mathbf{p}_{p,e}\rvert$	0.0075	0.0093	0.0074	0.0114	0.0416	0.0422	0.0340	0.0704	0.0063	0.0081	0.0065	0.0096
	$\Delta\lambda_{p,me}$	0.1731	0.2783	0.2293	0.3568	1.0284	1.2690	1.2844	1.4516	0.1231	0.2455	0.1540	0.3143
	$\Delta\lambda_{p,e}$	0.5457	0.4477	1.1992	0.8066	2.1012	1.9041	9.3440	10.1283	0.4930	0.3935	0.8584	0.4721
Square	$\lvert\mathbf{p}_{p,me}\rvert$	0.0347	0.0502	0.0316	0.0449	0.0936	0.1266	0.0844	0.1057	0.0265	0.0516	0.0238	0.0422
	$\lvert\mathbf{p}_{p,e}\rvert$	0.0059	0.0194	0.0080	0.0193	0.0304	0.0618	0.0331	0.0643	0.0046	0.0178	0.0074	0.0175
	$\Delta\lambda_{p,me}$	0.3134	0.3731	0.2299	0.2858	1.2170	1.3014	0.8094	1.1958	0.2934	0.3487	0.2142	0.2454
	$\Delta\lambda_{p,e}$	0.4643	0.4322	0.7307	0.5993	2.0217	1.6365	9.9402	6.2963	0.4133	0.3945	0.5870	0.4015
Circle	$\lvert\mathbf{p}_{p,me}\rvert$	0.0815	0.0865	0.0743	0.0773	0.1331	0.1609	0.1272	0.1607	0.0885	0.0876	0.0815	0.0798
	$\lvert\mathbf{p}_{p,e}\rvert$	0.0166	0.0247	0.0203	0.0377	0.0583	0.0751	0.0514	0.1104	0.0161	0.0221	0.0219	0.0364
	$\Delta\lambda_{p,me}$	1.0095	1.0528	0.9463	0.9270	2.7228	2.3916	2.5314	2.2907	0.9005	0.9320	0.8325	0.8210
	$\Delta\lambda_{p,e}$	0.6982	0.7750	4.4114	4.0576	2.7857	2.4726	9.9236	9.4989	0.5790	0.6955	5.0541	4.6105
Warehouse	$\lvert\mathbf{p}_{p,me}\rvert$	0.0364	0.0516	0.0327	0.0489	0.2130	0.2208	0.1878	0.2611	0.0163	0.0400	0.0167	0.0387
	$\lvert\mathbf{p}_{p,e}\rvert$	0.0124	0.0220	0.0112	0.0264	0.1043	0.1210	0.0917	0.1570	0.0079	0.0139	0.0074	0.0195
	$\Delta\lambda_{p,me}$	0.9784	1.3042	0.9463	1.0861	8.5725	7.9166	8.3830	7.8098	0.3763	0.5201	0.2955	0.4472
	$\Delta\lambda_{p,e}$	0.9547	1.2325	2.0735	2.5138	9.0384	7.5443	19.4421	19.7684	0.3987	0.5788	0.9083	1.8562

CHAPTER 7

Parabolic Airdrop Motion Planning

Within this chapter, a motion planning method for parabolic airdrop using a multirotor UAV is presented. Since any kind of airdrop implies detaching a payload, additional considerations on the mathematical model with payload are regarded first. The main requirements on the parabolic airdrop are reaching a launch point with a certain velocity that guides the payload towards the specified target point, shown on Fig. 7.1. Therefore, a motion planning method is derived to steer the system towards the launch point. The developed motion planning method is extensively tested in simulation environment, as well as in real world laboratory and outdoor experiments.

The inspiration for developing a parabolic airdrop motion planner for multirotor UAVs stems from the precision airdrop task, where the goal is to deliver a payload to a target in an environment. In literature, this task is almost exclusively performed by fixed wing aircraft, deploying the payload from a significant height. This is somewhat expected since the focus is mostly on resupplying tasks in remote areas. In such missions, the fixed wing aircraft outperform their multirotor counterparts due to the energy efficient flight performance. On the other hand, they need a specialized infrastructure or launch ramps to get airborne. The family of multirotor UAVs takes advantage of the Vertical Takeoff and Landing capability, making them capable to get airborne in obstacle-rich environments.

The parabolic airdrop considered in this chapter is performed in cluttered environments using multirotor UAVs. The agility and maneuverability of such vehicles makes them ideal for delivering a payload at close ranges. The envisioned task concentrates on indoor firefighting in hard to reach places, i.e. high rise buildings. Extinguishing a fully developed building fire requires a huge amount of water. Due to the limited payload capabilities of the multirotor aircraft, delivering water is inefficient and the fire is not likely to be extinguished. To alleviate

Figure 7.1: An example of the multirotor performing the parabolic airdrop in an outdoor environment. The multirotor steers towards the launch point and releases the payload, after which it executes a stopping motion. After the release instance, the free-fall parabolic trajectory can be observed. Copyright [14] CC BY 4.0.

this problem, the focus is drawn to deploying a fire extinguishing agent which is a more potent solution than delivering water. A fire extinguishing ball is a type of extinguishing agent that explodes several seconds after coming in contact with a fire, and disperses a fire retardant powder. However, in large fires one fire extinguishing ball is far from sufficient. Therefore, a team of multirotors continuously delivering such a payload is envisioned as a practical use case. Coordination and planning for a team of multirotors goes beyond the scope of this work, and the focus is on developing a viable motion planning method for a single multirotor.

7.1 Mathematical Model

The mathematical model of the multirotor UAV is presented in Section 3.1.2.2. For parabolic airdrop tasks, the system is considered to transport a payload and the mathematical model is altered to account for the payload, which is considered to be rigidly attached to the multirotor. The coordinate system convention is shown on Fig. 7.2.

The kinematics of an aerial manipulator are introduced in Section 3.1.1, with equation (3.6). In this case, the kinematic chain does not contain a manipulator, which yields a simpler equation:

$$\mathbf{T}_{\mathrm{W}}^{\mathrm{P}} = \mathbf{T}_{\mathrm{W}}^{\mathrm{B}} \cdot \mathbf{T}_{\mathrm{B}}^{\mathrm{P}}, \tag{7.1}$$

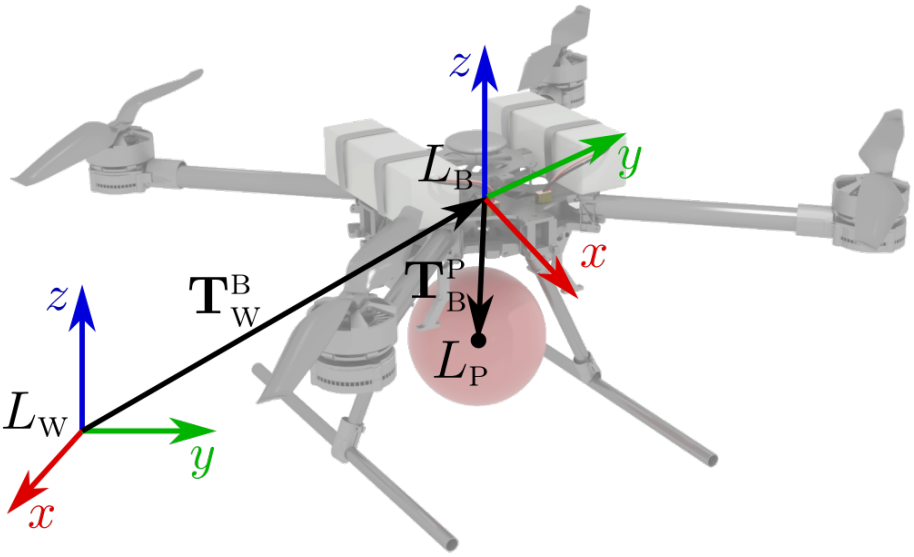


Figure 7.2: Coordinate system and transformations of a multirotor with an attached payload. L_W denotes the inertial coordinate system, L_B is attached to the multirotor body, and L_P is the payload center of mass. The transform between the world and body is denoted with $\mathbf{T}^B_W$, and the transform between the body and payload is denoted with $\mathbf{T}^P_B$. Copyright [14] CC BY 4.0.

where $\mathbf{T}^B_W$ denotes the homogeneous transform between the world and body frame, and $\mathbf{T}^P_B$ is a fixed transform from the body to the payload center of mass. Rearranging the kinematics equation yields:

$$\mathbf{T}^B_W = \mathbf{T}^P_W \cdot (\mathbf{T}^P_B)^{-1}. \tag{7.2}$$

This particular relation is especially useful when specifying the payload release point in the world frame. The parabolic airdrop motion planning, discussed in Section 7.2, considers the motion planning for the multirotor body. The above relation allows to calculate the body position in the world frame, based on the determined release point.

The dynamics of the multirotor are derived in Section 3.1.2. The sum of all forces, given in equation (3.22), can be rewritten as:

$$m_S \ddot{\mathbf{p}}^B_W = -m_S \mathbf{g} + \sum_{i=1}^{n_r} \mathbf{R}^B_W \mathbf{f}^{r_i}_B, \tag{7.3}$$

where m_S denotes the total mass of the system. If the multirotor is transporting a payload, the total mass can be written as a sum of the multirotor body and payload mass $m_S = m_B + m_P$. After the payload is released, the total mass of the system reduces to the multirotor mass. The total system inertia also changes depending on the payload. Since the payload is not at the body

center of mass, its inertia tensor can be obtained using the parallel axis theorem:

$$\mathbf{I}_{\mathrm{P}}^{\mathrm{B}} = \mathbf{I}_{\mathrm{P}} + m_{\mathrm{P}} \left((\mathbf{p}_{\mathrm{B}}^{\mathrm{P}})^{T} \cdot \mathbf{p}_{\mathrm{B}}^{\mathrm{P}} \cdot \mathbf{E}_{3\times 3} - \mathbf{p}_{\mathrm{B}}^{\mathrm{P}} \cdot (\mathbf{p}_{\mathrm{B}}^{\mathrm{P}})^{T} \right), \tag{7.4}$$

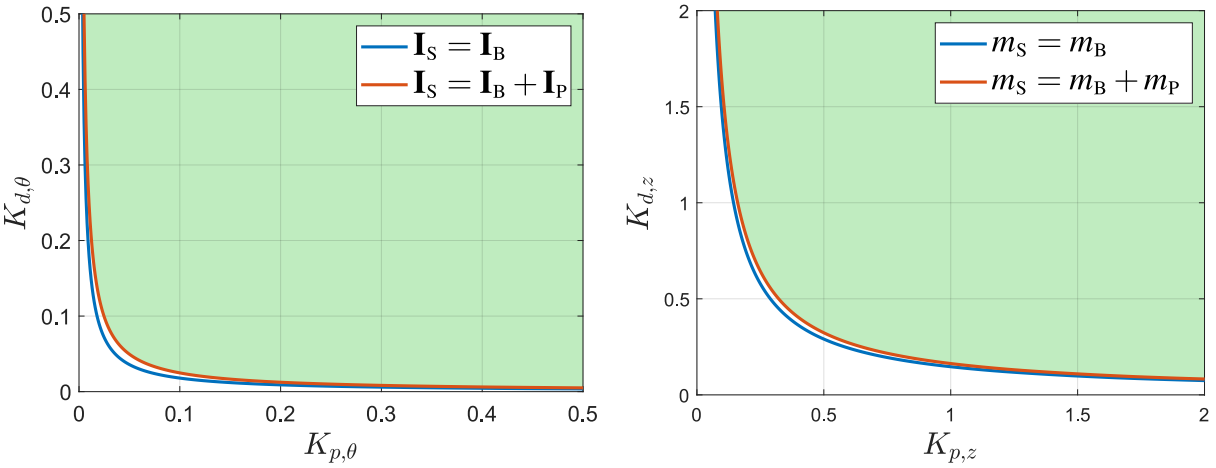
where $\mathbf{I}_{\mathrm{P}}$ is the payload inertia measured at its center of mass, $\mathbf{p}_{\mathrm{B}}^{\mathrm{P}}$ is the position vector from the body to the payload, and $\mathbf{E}_{3\times 3}$ is the identity matrix in $\mathbb{R}^{3\times 3}$ space. The total inertia of the system, expressed in the L_{B} coordinate system, is $\mathbf{I}_{\mathrm{S}} = \mathbf{I}_{\mathrm{B}} + \mathbf{I}_{\mathrm{P}}^{\mathrm{B}}$ when the multirotor is transporting the payload, and $\mathbf{I}_{\mathrm{S}} = \mathbf{I}_{\mathrm{B}}$ after the payload is released. The moment equation of the multirotor with a manipulator is given in equation (3.25). Removing the influence of the manipulator and including the total system inertia yields:

$$\mathbf{I}_{\mathrm{S}} \dot{\boldsymbol{\omega}}_{\mathrm{B}} + \boldsymbol{\omega}_{\mathrm{B}} \times (\mathbf{I}_{\mathrm{S}} \boldsymbol{\omega}_{\mathrm{B}}) = \sum_{i=1}^{n_r} (\mathbf{p}_{\mathrm{B}}^{r_i} \times \mathbf{f}_{\mathrm{B}}^{r_i} + \boldsymbol{\tau}_{\mathrm{B}}^{r_i}). \tag{7.5}$$

The controller structure employed is the standard cascade attitude and position control, described in Section 3.2. The identical procedure for the stability analysis of the angular control loops can be employed here. The payload changes the total inertia of the system, which is analyzed through equations (3.44) and (3.45), utilizing the Routh-Hurwitz criterion. Two stability boundaries are obtained, for system with and without the payload. The identical approach can be employed for the height controller, as it depends on the overall system mass. The stability regions obtained in this manner are shown on Fig. 7.3. This stability analysis is performed for the real world AscTec NEO hexacopter carrying a payload. The parameters are following: $T_m = 0.0182s$, $K_m = 92.5Vs/rad$, $I_{xx,\mathrm{B}} = 0.0331kg \cdot m^2$, $m_{\mathrm{B}} = 2.662kg$, $I_{\mathrm{P}} = 0.000675kg \cdot m^2$ measured around its center of mass, $m_p = 0.3kg$, and $r_{\mathrm{B}}^{\mathrm{P}} = [0\ 0\ -0.2]^T$. It can be observed that the payload slightly shifts the stability region. When the payload is detached, it acts as a step disturbance on the system. This can be observed as a hybrid system with two states, with and without the payload. According to the hybrid systems theory, the hybrid system is stable if all its states are stable, and the system is given enough time to stabilize after a switch occurs [49]. Since there is only one switch that corresponds to detaching the payload, and both states are stable, the overall hybrid system is considered to be stable as well.

7.2 Airdrop Trajectory Planning

In this section, the parabolic airdrop motion planner for multirotors is derived and discussed. As stated earlier, the payload is considered to be rigidly attached to the multirotor body, and it is assumed that it can be detached instantaneously upon request. The detached payload follows a parabolic trajectory, which is the direct consequence of its initial velocity and force of gravity. The motion planning method goes through three stages: based on the provided target in the world frame, a set of parabolic trajectory candidates is determined that satisfy the payload



(a) Stability region for the pitch controller.

(b) Stability region for the height controller.

Figure 7.3: Stability region of the system with and without payload.

travelling towards the target; next, an obstacle free path is planned towards the launch point; and finally, a collision-free trajectory is planned based on the underlying path from the previous step. The trajectory is then sent to the system to be executed.

7.2.1 Parabolic Free-fall Trajectory

When a certain projectile motion is observed in a free-fall, it can be described as a parabolic trajectory. One intuitive example of such a trajectory is when a basketball player throws a ball towards a basket. To define the trajectory in a more precise manner, it should be referred to as a ballistic trajectory due to the air resistance force acting on the projectile during motion. However, the distances considered for performing the parabolic airdrop in this thesis are relatively small, and the projectile is launched with a low speed. Therefore, the air resistance can be safely neglected, which is described in detail in Section 7.2.6.

Note: As the parabolic trajectory is introduced in this section, the notation referring to it has a lower-case p in subscript. The upper-case P denotes the payload. Furthermore, to simplify the notation, all vectors are expressed in the world coordinate frame.

At the release instance, the payload has some position and initial velocity in the world frame. Although the overall system can exhibit some acceleration just before the launch, after the release only the gravity force acts on the payload, giving it the characteristic parabolic trajectory shape. The initial conditions allow for obtaining the parabolic payload trajectory

with respect to time:

$$\begin{aligned} x_p(t) &= x_{\mathrm{L}} + v_{0,p} \cdot cos(\theta_p) cos(\psi_p) \cdot t \\ y_p(t) &= y_{\mathrm{L}} + v_{0,p} \cdot cos(\theta_p) sin(\psi_p) \cdot t \\ z_p(t) &= z_{\mathrm{L}} + v_{0,p} \cdot sin(\theta_p) \cdot t - \frac{g \cdot t^2}{2}, \end{aligned} \tag{7.6}$$

where $\mathbf{p}_{\mathrm{L}} = \begin{bmatrix} x_{\mathrm{L}} & y_{\mathrm{L}} & z_{\mathrm{L}} \end{bmatrix}^T$ is the launch position in the world frame, $v_{0,p}$ is the payload initial speed, θ_p is the payload launch angle, ψ_p is the payload direction angle, g is the gravity magnitude along the negative z_{W} axis, and t is time. The projection of the parabola on a 2D plane is depicted on Fig. 7.4.

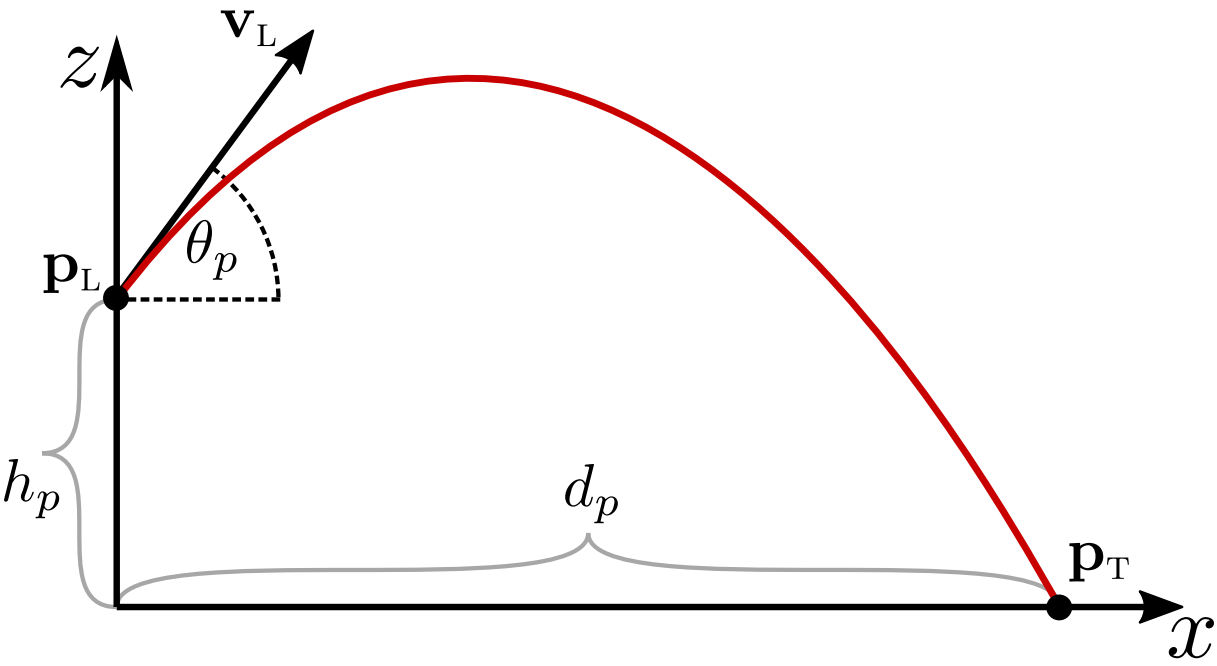


Figure 7.4: A parabolic trajectory in $x-z$ plane. The trajectory is defined with the start point $\mathbf{p}_{\mathrm{L}}$ and velocity $\mathbf{v}_{\mathrm{L}}$ which drive the payload towards the target point $\mathbf{p}_{\mathrm{T}}$. Distance between the launch and target point is denoted with d_p, height difference with h_p, and θ_p the vertical launch angle.

Since one of the inputs for the parabolic airdrop planning is the target position in the world frame, the problem becomes how to obtain a valid launch position and velocity. Furthermore, if the payload parabolic trajectory is observed in a 3D space, there are infinitely many solutions that satisfy the target position. Therefore, it is necessary to define the launch position and velocity as functions of parabola parameters and the target point. An illustration of the multiple parabolic trajectory solutions is depicted on Fig. 7.5. Taking advantage of the fact that the parabolic trajectory can be projected on a 2D plane, the configuration vector can be defined as:

$$\mathbf{C}_p = \begin{bmatrix} d_p & h_p & v_{0,p} & \theta_p & \psi_p \end{bmatrix}^T, \tag{7.7}$$

where the new parameters d_p and h_p are horizontal and vertical displacements between the launch and target points. Based on the parabola configuration vector and targat point, the launch point can be defined as:

$$\mathbf{p}_{\mathrm{L}} = \begin{bmatrix} x_{\mathrm{T}} - d_p \cdot cos(\theta_p) cos(\psi_p) \\ y_{\mathrm{T}} - d_p \cdot cos(\theta_p) sin(\psi_p) \\ z_{\mathrm{T}} + h_p \end{bmatrix}, \tag{7.8}$$

and the velocity vector as:

$$\mathbf{v}_{\mathrm{L}} = \begin{bmatrix} v_{0,p} \cdot cos(\theta_p) cos(\psi_p) \\ v_{0,p} \cdot cos(\theta_p) sin(\psi_p) \\ v_{0,p} \cdot sin(\theta_p) \end{bmatrix}. \tag{7.9}$$

This notation allows constructing a parabolic trajectory based on a set of configuration parameters. However, the parabola configuration parameters are tightly coupled based on equation (7.6). In other words, it is not possible to choose an arbitrary combination of these parameters and expect the payload to reach the target point. In fact, the only parameter that can be chosen arbitrarily is the angle ψ_p, which only determines the direction of the parabola.

To determine a feasible parabolic trajectory, some parameters need to be set by the user, while others are calculated based on them. Intuitively, the initial velocity $v_{0,p}$ is bounded with the maximum velocity of the multirotor. The launch angle θ_p can be chosen arbitrarily, however, steep angles require either high initial speed or close proximity to the target point. If the horizontal displacement d_p is chosen together with the initial speed and launch angle, the vertical displacement can be uniquely determined using equation (7.6):

$$\begin{aligned} T_p &= \frac{d_p}{v_{0,p} \cdot cos(\theta_p)} \\ h_p &= v_{0,p} \cdot sin(\theta_p) \cdot T_p - \frac{g T_p^2}{2}, \end{aligned} \tag{7.10}$$

where T_p is the time of impact. This completes the parabola configuration vector $\mathbf{C}_p$, which is required to obtain the launch point $\mathbf{p}_{\mathrm{L}}$.

7.2.1.1 Search Space Construction

As mentioned earlier, there is an infinite set of launch points that satisfy the target point $\mathbf{p}_{\mathrm{T}}$. As convenient as that is, a single launch point needs to be chosen among this set. To do so, the relevant variables in the search set are discretized yielding a finite search space. For simplicity

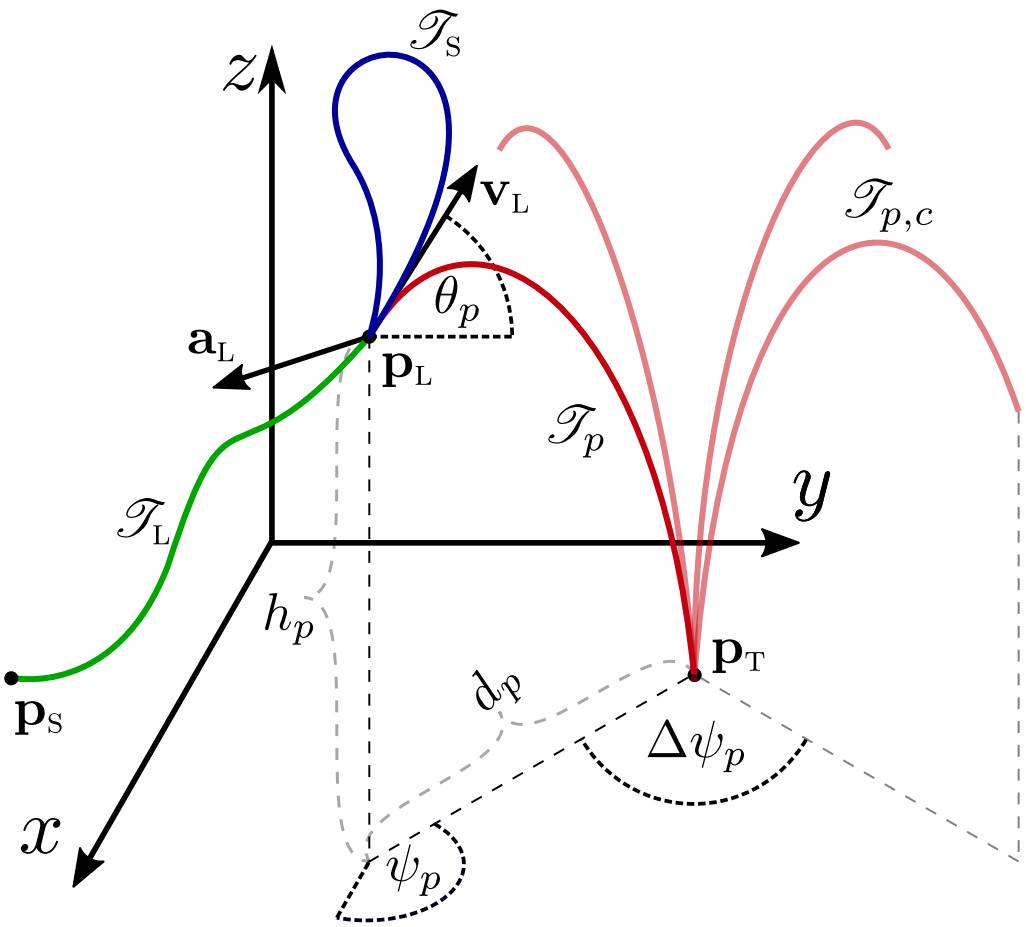


Figure 7.5: An example of a parabolic trajectory in 3D space. A single payload trajectory $\mathscr{T}_p$ is chosen among multiple candidates $\mathscr{T}_{p,c}$. The multirotor executes the launch trajectory $\mathscr{T}_\mathrm{L}$, as well as the stopping trajectory $\mathscr{T}_\mathrm{S}$ after the release. Copyright [14] CC BY 4.0.

and brevity, based on equations (7.8) and (7.9), a launch configuration can be defined as:

$$\mathbf{C}_\mathrm{L} = \begin{bmatrix} \mathbf{p}_\mathrm{L}^T & \mathbf{v}_\mathrm{L}^T & \mathbf{a}_\mathrm{L}^T \end{bmatrix}^T \tag{7.11}$$

where a new parameter $\mathbf{a}_\mathrm{L} \in \mathbb{R}^3$ is introduced, representing the acceleration at the release instance. Specifying an acceleration at the launch point might seem redundant since after the release instance only gravity accelerates the payload. However, the multirotor transporting the payload is still required to reach the specified acceleration, which can be used to perform more aggressive maneuvers when the airdrop is performed in close proximity to an obstacle.

Imagine an example where the multirotor is tasked to launch the payload through a window of a building. At the launch point, the multirotor is going to have some velocity pointing towards the building wall. Depending on the velocity magnitude, it is possible that multirotor is not able to stop before hitting the wall. Obviously, this is not the desired behavior. To alleviate this problem, a horizontal acceleration a_h in opposite horizontal direction from the launch velocity

is introduced. Using this heuristic, the launch acceleration is a function of a single variable a_h:

$$\mathbf{a}_{\mathrm{L}} = \begin{bmatrix} -a_h \cdot cos(\psi_p) & -a_h \cdot sin(\psi_p) & 0 \end{bmatrix}^T. \tag{7.12}$$

This makes the launch configuration vector a function of several parameters:

$$\mathbf{C}_{\mathrm{L}} = f\left(d_p, v_{0,p}, \theta_p, \psi_p, a_h\right) \in \mathbb{R}^9, \tag{7.13}$$

where the function arguments will become the search space variables. Note that the height displacement h_p is also a parameter defining the parabolic trajectory, however, it is not required in the search space as it is uniquely determined through equation (7.10).

To construct a finite search space, parameters from equation (7.13) can be bounded and discretized. The bounds are imposed by user, based on the task and environment. The discretization steps are also set by the user, where the lower step yields a more dense and larger search space. Therefore, the search space can be defined as a set of all possible combinations of search variables:

$$\mathscr{L}_p = \left\{ \mathbf{C}_{\mathrm{L}} \;\middle|\; \begin{array}{c} d_p \in \left\{d_p^{min}, d_p^{min} + \Delta d_p, \ldots, d_p^{max}\right\} \\ v_{0,p} \in \left\{v_{0,p}^{min}, v_{0,p}^{min} + \Delta v_{0,p}, \ldots, v_{0,p}^{max}\right\} \\ \theta_p \in \left\{\theta_p^{min}, \theta_p^{min} + \Delta \theta_p, \ldots, \theta_p^{max}\right\} \\ \psi_p \in \left\{0, \Delta\psi_p, \ldots, 2\pi - \Delta\psi_p\right\} \\ a_h \in \left\{a_h^{min}, a_h^{min} + \Delta a_h, \ldots, a_h^{max}\right\} \end{array} \right\} \tag{7.14}$$

where Δd_p, $\Delta v_{0,p}$, $\Delta\theta_p$, $\Delta\psi_p$ and Δa_h are discrete steps for the search space variables. Each combination from the above set yields a valid launch point configuration $\mathbf{C}_{\mathrm{L}}$, which defines a finite number of parabolic trajectory candidates $\mathscr{T}_{p,c}$. Naturally, not all candidates are suitable for performing the airdrop. In subsequent sections, some candidates are ruled out due to collision with the environment or infeasible multirotor trajectories for the corresponding launch point configuration.

7.2.2 Path Planning

A general description of the path planning approach is outlined in Section 4.2. In case of the parabolic airdrop, only a single multirotor is considered. Therefore, a single waypoint consists of the position and yaw angle of the multirotor:

$$\mathbf{w} = \begin{bmatrix} x & y & z & \psi \end{bmatrix}^T. \tag{7.15}$$

The RRT* path planner is employed to obtain an obstacle-free path, where the three spatial positions are represented as a vector, and the rotational degree of freedom ψ is represented as a

$SO(2)$ state, instructing the planner to perform search on a rotation matrix. The bounds for the planner are determined based on the environment size, and the environment itself is represented as OctoMap, discussed in Section 4.2.1.

The start waypoint $\mathbf{w}_S$ is the multirotor current configuration in the world frame. The end point for the path planning is the launch position and horizontal orientation $\mathbf{w}_L = [\mathbf{p}_L^T \ \psi_p]^T$, determined in the previous section. The output of the path planner is an obstacle-free piecewise straight line path:

$$\mathscr{P} = \left\{ \mathbf{w}_i \mid i \in (1,2,\ldots,n_p), \mathbf{w}_i \in \mathbb{R}^4 \right\}. \tag{7.16}$$

The launch configuration $\mathbf{C}_L$ requires a known position, velocity and acceleration at the release instance. However, the RRT* does not account for dynamics and can only ensure the launch position is reached. To satisfy the dynamical conditions, a trajectory is planned based on the path $\mathscr{P}$, which is discussed in subsequent sections.

7.2.3 TOPP-RA Trajectory Interpolation

The TOPP-RA planner has been introduced in Section 4.3, and is used to plan dynamically feasible trajectories. To recap, the planner requires a set of waypoints as an input, which is provided in the form of a path $\mathscr{P}$. Furthermore, it requires velocity and acceleration constraints of each degree of freedom. Based on the inputs, the TOPP-RA generates a trajectory:

$$\mathscr{T}_T = \left\{ \mathbf{x}(kT_s) \mid \mathbf{x}(kT_s) \in \mathbb{R}^{3\cdot 4}, k \in (0,\ldots,n_t \right\}, \tag{7.17}$$

where $\mathbf{x}$ is a single trajectory point comprised of position, velocity and acceleration, T_s is the trajectory sampling time, and n_t is the number of points in trajectory.

The trajectory $\mathscr{T}_T$ is a stop-to-stop trajectory at this point. However, a requirement imposed in Section 7.2.1 contains the velocity and acceleration at the launch point. The TOPP-RA algorithm allows for specifying non-zero velocities at the beginning and the end of the trajectory. However, the widely accepted practical implementation struggles to find a feasible solution in such cases. Through an empirical validation, the TOPP-RA managed to find a solution if the provided velocities are close to zero, otherwise it failed to find a solution in a vast majority of cases. It has also been discovered that the run time of the algorithm increases in such a scenario. Furthermore, the algorithm does not support specifying a non-zero acceleration at any point, which is in this thesis a part of the launch configuration vector $\mathbf{C}_L$.

Although the TOPP-RA fails when arbitrary non-zero velocity and acceleration are set at the start or at the end, it is still a reliable planner. TOPP-RA plans time optimal trajectories while respecting the given constraints, and does this in a very short time period for both small and large environments. Throughout the extensive testing in simulation and experimental en-

vironments, the planner never failed to produce a feasible stop-to-stop trajectory. Furthermore, in our analysis of the planner so far, it never failed on high-dimensional problems, described in Section 5.3.2.3. In such cases, the planning time only slightly increased when compared to the multirotor-only planning with four DoF. One disadvantage of the TOPP-RA when compared to the convex optimimzation methods is the non-continuity of the high-order derivatives, such as jerk or snap. Methods from [62, 81] work well with a small number of points in the path ($n < 15$) and with a limited number of DoF. As large environments are used in following sections, the path planning produces a large number of waypoints which imposes a significant planning time for the aforementioned methods. This kind of reliability and a fast planning time for a large number of waypoints prompts us to still use the TOPP-RA for the initial trajectory planning. Achieving a non-zero velocity and acceleration at the launch point is described in the following section.

7.2.4 Cubic Spline Interpolation

The TOPP-RA planner discussed in the previous section is both reliable and fast planner. To introduce velocity and acceleration for the launch point, a 5^{th} order spline can be used. This spline has six free coefficients, which allow specifying a position, velocity and acceleration at both ends of the spline. A single point of the initial trajectory consists of a position, velocity and acceleration, which allows for 5^{th} order spline replanning of any segment of the initial trajectory. A general equation of the 5^{th} order spline can be written as:

$$p(t) = \sum_{i=0}^{5} a_i t^i, \tag{7.18}$$

where $p(t)$ is the position polynomial and a_i are its coefficients. The first derivative of the position polynomial is velocity $v(t) = \dot{p}(t)$ and the second derivative is acceleration $a(t) = \ddot{p}(t)$. If the duration of the polynomial is denoted as T, the start and end conditions are defined as:

$$\begin{aligned} p(t=0) &= p_s, \quad v(t=0) = v_s, \quad v(t=0) = v_s \\ p(t=T) &= p_e, \quad v(t=T) = v_e, \quad v(t=T) = v_e, \end{aligned} \tag{7.19}$$

where subscript s denotes the start configuration, and subscript e denotes the end configuration. The start and end configurations can be either supplied by the user, or taken from the initial trajectory if it needs to be partially replaced. In any case, they are considered to be known values at this point.

With known start and end configuration, the spline coefficients can be uniquely determined:

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 & 0 \\ -\frac{10}{T^3} & \frac{10}{T^3} & -\frac{6}{T^2} & -\frac{4}{T^2} & -\frac{3}{2T} & \frac{1}{2T} \\ \frac{15}{T^4} & -\frac{15}{T^4} & \frac{8}{T^3} & \frac{7}{T^3} & \frac{3}{2T^2} & -\frac{1}{T^2} \\ -\frac{6}{T^4} & \frac{6}{T^4} & -\frac{3}{T^3} & -\frac{3}{T^3} & -\frac{1}{2T^2} & \frac{1}{T^2} \end{bmatrix} \cdot \begin{bmatrix} p_s \\ v_s \\ a_s \\ p_e \\ v_e \\ a_e \end{bmatrix}. \tag{7.20}$$

A careful reader will notice that the spline duration T is also a free variable and needs to be determined, as it directly influences the value of free coefficients. To determine the spline duration, a simple subgradient method can be employed. Focusing on a single degree of freedom, dynamical constraints in the form of maximum velocity v_{max} and a_{max} are applied. The initial guess for the spline duration is set to $T_0 = |p_e - p_0| / v_{max}$. A spline is generated based on the initial duration guess, however, the maximum velocity $v_{max,s}$ and acceleration $a_{max,s}$ achieved on such a spline can violate the imposed dynamical constraints. The subgradient search is performed to find an adequate spline duration:

$$\begin{aligned} s &= \max\left\{ \frac{v_{max,s}}{v_{max}}, \sqrt{\frac{a_{max,s}}{a_{max}}} \right\} \\ T_{i+1} &= T_i \cdot (1 + \alpha \cdot sgn(s-1)), \end{aligned} \tag{7.21}$$

where α is the convergence factor, i is the iteration counter starting from $i = 0$, s is the derivative derivative ratio factor, and sgn denotes the signum function. The goal of this search is to meet either velocity or acceleration constraint along the spline, which happens when $s = 1$. The subgradient search is considered finished if the derivative ratio factor is within some user defined bounds $-\varepsilon < s - 1 < \varepsilon$.

In case of multiple degrees of freedom, two approaches can be employed. In the first approach, each degree of freedom is observed separately, where the spline duration for each DoF is going to be different. This is acceptable for planning the stopping trajectory, which will be discussed in Section 7.2.4.2. However, for planning the launch trajectory discussed in Section 7.2.4.1, each DoF needs to reach the launch waypoint at the same time instance. To account for all DoFs at once, the maximum value of all derivative ratio factors and duration times is selected:

$$s = \max\{s_1, s_2, \ldots, s_n\} \tag{7.22}$$

This extension produces an n-dimensional, time synchronized spline. Note that only one DoF meets its dynamical constraint using this approach.

7.2.4.1 Launch Trajectory

Recall that the initial trajectory planned in Section 7.2.3 is a stop-to-stop motion, due to the inner workings of the TOPP-RA algorithm. However, to achieve a successful parabolic airdrop, the multirotor has to reach some launch velocity $\mathbf{v}_{\mathrm{L}}$, as well as acceleration $\mathbf{v}_{\mathrm{L}}$. To do so, a part of the initial trajectory $\mathscr{T}_{\mathrm{T}}$ is replaced with a 5^{th} order spline to achieve the required motion at the launch instance. A visual illustration of the launch trajectory planning is depicted on Fig. 7.6.

A set of launch trajectory candidate is computed by going from the very end of the initial trajectory, and replacing a part of it. Each candidate is created with a distance step Δl_{T}, by moving backwards along the initial trajectory. To calculate the arc length of the initial trajectory, following sum formula can be used:

$$l_{\mathrm{T}} = \sum_{k=k_i}^{n_t-1} \sqrt{[x(kT_s) - x((k+1)T_s)]^2 + [y(kT_s) - y((k+1)T_s)]^2 + [z(kT_s) - z((k+1)T_s)]^2}, \tag{7.23}$$

where n_t is the total number of trajectory points, T_s is the trajectory sampling time, and k_i is the instance the measurement started. Every Δl_{T} meters, a candidate trajectory point $\mathbf{x}_{c,i}$ is taken from the initially planned trajectory. Since the trajectory is discretized, the above formula sums the distances between each two subsequent discretization points, yielding the arc length. It is saved in a set of candidates, and the procedure is repeated until the measured trajectory distance surpasses some predefined value $l_{max,\mathrm{T}}$. As a result, a set of n_c trajectory point candidates $\mathbf{x}_{c,i}$ is obtained.

A spline trajectory is planned afterwards, between each candidate point $\mathbf{x}_{c,i}$ and the launch point $\mathbf{x}_{\mathrm{L}} = \begin{bmatrix} \mathbf{w}_{\mathrm{L}}^T & \dot{\mathbf{w}}_{\mathrm{L}}^T & \ddot{\mathbf{w}}_{\mathrm{L}}^T \end{bmatrix}^T$. The planned spline trajectory becomes a part of the launch candidates set:

$$\mathscr{L}_c = \{ \mathscr{T}_{c,i} \mid i \in (1, \ldots, n_c) \}, \tag{7.24}$$

where $\mathscr{T}_{c,i}$ is a single spline trajectory candidate. The criterion to choose a single launch trajectory is twofold. First, a candidate launch trajectory is checked for collisions with the environment. The candidate launch trajectory is discarded if it collides with the environment. The employed selection criterion for the collision-free candidates is the length ratio between the candidate itself and the part of the initial trajectory it replaces:

$$r_{l,i} = \frac{l_{c,i}}{l_{\mathrm{T},i}}, \tag{7.25}$$

where $l_{c,i}$ is the candidate spline trajectory length, and $l_{\mathrm{T},i}$ is length of the initial trajectory part.

The candidate with the ratio closest to one, ideally $r_l = 1$, is chosen and becomes the launch spline $\mathcal{T}_\mathrm{L}$. Although this criterion is relatively simple, it has proven to be very effective in the empirical analysis. Other criteria has also been tested, such as the velocity and acceleration deviation from the initial trajectory, and weighted sum of multiple criteria. In the end, the length ratio criterion produced trajectories with the least amount of bumps and curves along the spline. A careful reader should note that the ratio $r_{l,i}$ is not monotonically decreasing or increasing, therefore, it is necessary to evaluate all candidates to select the most suitable one.

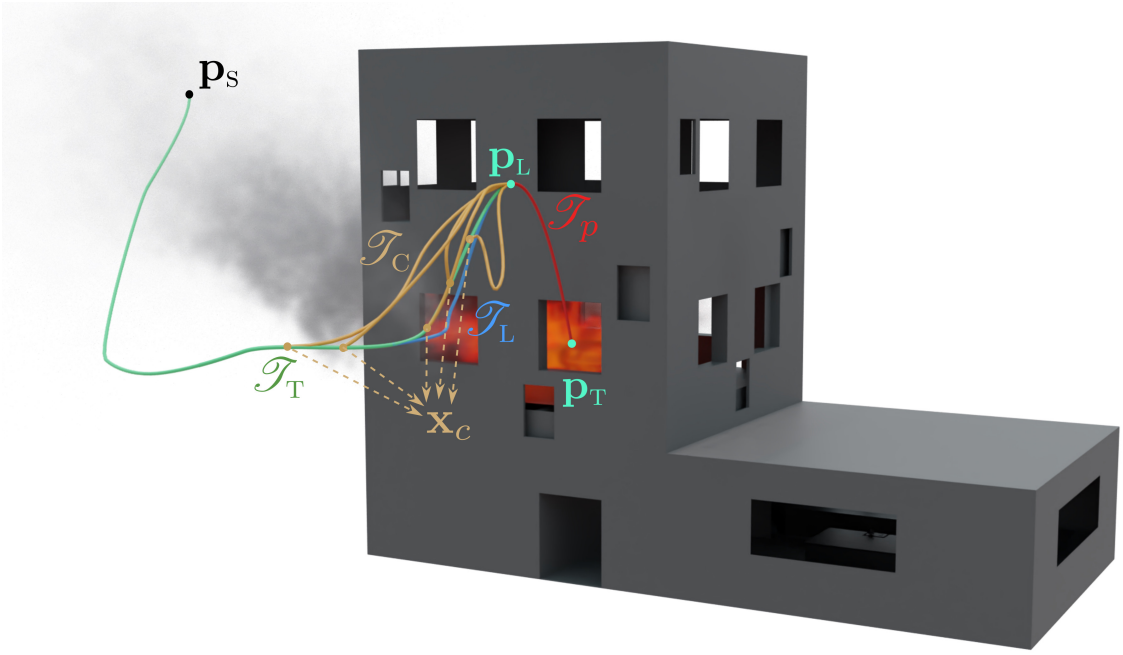


Figure 7.6: An illustrative example of the launch trajectory planning. A part of the initial trajectory $\mathcal{T}_\mathrm{T}$ is replaced by the launch trajectory $\mathcal{T}_\mathrm{L}$, which is chosen among multiple candidates $\mathcal{T}_c$. Points along the initial trajectory denoted with $\mathbf{x}_c$ represent the candidate for the launch spline start. The launch point $\mathbf{p}_\mathrm{L}$ is chosen from the set of launch point candidates, and is determined based on the supplied target point $\mathbf{p}_\mathrm{T}$. The initial multirotor position is denoted with $\mathbf{p}_\mathrm{S}$.

7.2.4.2 Stopping Trajectory

After the payload launch, the multirotor has some non-zero velocity. Therefore, it will execute some kind of a stopping motion, and by doing so it could potentially hit an obstacle. To account for the stopping motion, a stopping spline trajectory $\mathcal{T}_\mathrm{S}$ is planned. The start point of the spline is the launch configuration $\mathbf{x}_\mathrm{L}$, introduced in the previous section. The stopping trajectory end point is the same as the launch point, except for the velocity and acceleration which are set to zero:

$$\mathbf{x}_\mathrm{R} = \begin{bmatrix} \mathbf{w}_\mathrm{L}^T & \mathbf{0}_{1\times 4} & \mathbf{0}_{1\times 4} \end{bmatrix}^T. \tag{7.26}$$

Since the previous section ensured the launch configuration to be valid, the resting point point will certainly be valid. The spline planned between $\mathbf{x}_{\mathrm{L}}$ and $\mathbf{x}_{\mathrm{R}}$ has to be checked for collisions because the stopping motion can get in contact with the environment. If this happens, the launch configuration is discarded and the algorithm continues to search the next launch configuration.

As already mentioned in Section 7.2.4, there are two ways to plan a spline trajectory. For the stopping trajectory, each axis is parameterized separately. Consequently, each degree of freedom has a different execution time, and the resting point is not reached simultaneously. In turn, such an approach yields a more aggressive trajectory since all degrees of freedom will reach their respective dynamical constraint.

7.2.5 Airdrop Planning Overview

To summarize, the parabolic airdrop trajectory planning, derived within previous sections, is briefly overviewed. The user provides the desired target $\mathbf{p}_{\mathrm{T}}$ in the world frame. Based on the target, a finite set of launch point candidates $\mathscr{L}_c$ is obtained, described in Section 7.2.1. Iterating through the set, the parabolic trajectory candidates are checked for collision and a collision-free path $\mathscr{P}$ is planned between the multirotor start point and the collision-free launch candidate, described in Section 7.2.2. Next, an initial stop-to-stop trajectory $\mathscr{T}_{\mathrm{T}}$ is planned using the TOPP-RA algorithm, as described in Section 7.2.3. Since the multirotor must achieve the planned velocity and acceleration at the launch point, a part of the initial trajectory is replaced with a 5^{th} order spline, denoted as launch trajectory $\mathscr{T}_{\mathrm{L}}$ and described in Section 7.2.4.1. After reaching the launch point, the multirotor will have some velocity which necessitates the stopping motion. Therefore, a stopping trajectory $\mathscr{T}_{\mathrm{S}}$ is planned according to Section 7.2.4.2. The TOPP-RA, launch and stopping trajectories are then concatenated into the final parabolic airdrop trajectory $\mathscr{T}_{\mathrm{P}}$. Throughout the whole procedure it is ensured that all multirotor and payload trajectories are collision-free. Iterating through the launch candidate set $\mathscr{L}_p$ is done sequentially. As soon as a collision-free trajectory $\mathscr{T}_{\mathrm{P}}$ is found the search is considered finished and subsequent candidates are not evaluated. The full planning procedure is captured within Algorithm 1.

Dynamical constraints note. All three trajectories planned in the previous sections require dynamical constraints in the form of maximum velocity and acceleration for each degree of freedom. These constraints can be set independently for each stage of the trajectory planning. While searching for a suitable launch trajectory start candidate $\mathbf{x}_c$, the velocity and acceleration magnitude is not a-priori known. In worst case, the candidate point features maximum velocity or acceleration. If the constraints are set the same or less for the launch spline planning, there is a possibility that the procedure will never end. Therefore, it is recommended to set higher dynamical constraints for the launch trajectory compared to the initial trajectory. This

PlanAirdropTrajectory $(\mathbf{p}_\mathrm{T}, \mathbf{w}_\mathrm{S})$**:**
  inputs : Target point in the world frame $\mathbf{p}_\mathrm{T}$,
    Multirotor current configuration in the world frame $\mathbf{w}_\mathrm{S}$
  output: Trajectory $\mathscr{T}_\mathrm{P}$
  `/* Generate set of candidate launch points, Section 7.2.1 */`
  $\mathscr{L}_p$ = generateLaunchCandidates($\mathbf{p}_\mathrm{T}$)
  forall $\boldsymbol{C}_\mathrm{L} \in \mathscr{L}_p$ **do**
    `/* Plan path, Section 7.2.2 */`
    $\mathbf{x}_\mathrm{L}$ = generateLaunchPoint($\mathbf{C}_\mathrm{L}$);
    $\mathscr{P}$ = planPath($\mathbf{w}_\mathrm{S}$, $\mathbf{x}_\mathrm{L}$);
    `/* Plan initial trajectory, Section 7.2.3 */`
    $\mathscr{T}_\mathrm{T}$ = planToppraTrajectory($\mathscr{P}$);
    `/* Plan launch spline, Section 7.2.4.1 */`
    $\mathscr{T}_\mathrm{L}$ = planLaunchTrajectory($\mathscr{T}_\mathrm{T}$, $\mathbf{x}_\mathrm{L}$);
    `/* Plan stopping spline, Section 7.2.4.2 */`
    $\mathscr{T}_\mathrm{S}$ = planStoppingTrajectory($\mathbf{x}_\mathrm{L}$);
    `/* Concatenate trajectories */`
    $\mathscr{T}_\mathrm{P}$ = concatenateTrajectories($\mathscr{T}_\mathrm{T}$,$\mathscr{T}_\mathrm{L}$, $\mathscr{T}_\mathrm{S}$);
    if *isCollisionFree*($\mathscr{T}_\mathrm{P}$) **then**
      return $\mathscr{T}_\mathrm{P}$;
    else
      continue searching;
    end
  end
end

Algorithm 1: Parabolic airdrop trajectory planning overview.

is also going to reduce the subgradient search time, rendering the method less time consuming. Similar holds for the stopping trajectory, where the start conditions are the launch velocity and acceleration. Setting the constraints higher than the launch configuration yields a faster search.

7.2.6 Neglecting Air Resistance

The whole parabolic airdrop trajectory planning procedure has an underlying assumption that after the launch instance only the force of gravity acts on the payload. In reality, this is not true because the air resistance plays a role in the payload trajectory. This is captured in the term ballistic trajectory, where air resistance is taken into account while predicting the projectile target. Such an approach is perfectly sensible for payloads travelling from significant heights or payloads launched with high initial velocity, since the air resistance depends on the velocity magnitude. However, in this thesis, the payload is considered to travel a relatively short distance with limited speed. The purpose of this section is to briefly address the air resistance influence on the payload reaching the specified target.

To that end, the payload is considered to a ball. The air resistance force of a spherical object can be calculated with:

$$F_d = \frac{1}{2}\rho_{air} \cdot v^2 \cdot C_d\left(Re\left(v,r\right)\right) \cdot A, \tag{7.27}$$

where $\rho_{air} = 1.1839 kg/m^3$ is the air density, v is the projectile speed, C_d is the drag coefficient of the sphere that depends on the Reynolds number Re, r is the sphere radius, and $A = r^2\pi$ is the sphere projection area perpendicular to the velocity direction.

The Reynolds number itself depends on the speed and the sphere radius, and can be expressed as:

$$Re(v,r) = \frac{\rho_{air} v r}{3\mu_{air}}, \tag{7.28}$$

where $\mu_{air} = 1.837 \cdot 10^{-5} Ns/m^2$ is the dynamic air viscosity. To obtain the drag coefficient of a sphere the work from [140] is consulted, where authors analyze the drag coefficient for various object speeds and radii. A suitable equation for the expected payload speed is repeated here:

$$C_d = \frac{777\left(\frac{669806}{875} + \frac{114976}{1155}Re + \frac{707}{1380}Re^2\right)}{646Re\left(\frac{32869}{952} + \frac{924}{643}Re + \frac{1}{385718}Re^2\right)}. \tag{7.29}$$

With the equation above, the air resistance force F_d can be determined.

Based on the system and task specific limitations, following constraints are considered: the initial horizontal velocity magnitude of the released object is $v_h < 5m/s$; the maximum height between the release point and the target is $\Delta z < 10m$; the radius of the payload is within

interval $0.04m < r < 0.25m$; and the mass of the payload is within the interval $0.3kg < m_p < 3kg$. Furthermore, if the payload is released horizontally, the maximum speed is achieved at the target. The maximum speed can be obtained with $v_{max} = \sqrt{v_h^2 + 2g\Delta z}$, where g denotes gravitational acceleration. Plugging all the above assumptions in equations (7.27), the obtained interval for the Reynolds number of $1278 < Re < 7987$, and finally for the maximum drag force along the payload trajectory $0.0007N < F_d < 0.0147N$.

As the drag force of the sphere depends on the speed, it increases as the object accelerates. To be on the safe side, the worst case scenario is assumed, in which the force along the whole path is constant and equals the maximum drag force at the impact. Taking the mass of the payload into account and relying on the fact that the path traversed under constant acceleration equals $s = 0.5 \cdot \frac{F_d}{m_p} t^2$, the obtained maximum deviation of the object with respect to the target is $\Delta s_{max} = 0.005m$ measured along the full path s. The obtained maximum deviation is negligible, especially considering the fact that the assumptions that led to the maximum deviation are favoring the worst case scenario. Therefore, the air resistance can be safely neglected when accounting for the ballistic free fall trajectory.

7.3 Simulation Results

The simulation environment used to verify the parabolic airdrop motion planning is Gazebo with ROS middleware, as described in Section 5.1.1.1. To attach a payload to the multirotor, the *storm_gazebo_ros_magnet* Gazebo plugin simulating a magnet is used. The plugin is developed as part of work in [141], with the emphasis on using a permanent dipole magnet. This approach is sufficient for picking up and transporting a payload, however, to release it the plugin is augmented with a simple gain turning it on and off, effectively simulating an electromagnet [142]. The payload is modelled as a ball, with mass and radius as parameters. The ball is augmented with a permanent magnet to use the Gazebo magnet plugin. Throughout the simulation, the ball radius is set to $r_b = 0.1m$ and the mass is kept at $m_b = 0.3kg$. To measure the executed ball trajectory, an odometry sensor is attached to the ball center of gravity.

Several tests are performed in the simulation to evaluate the planner and the overall system performance. First, the planner is tested in an obstacle-free environment to concentrate on the trajectory planning and executed payload trajectory. Next, the full planner is tested in a large outdoor environment, as well as in a dense office environment. Common performance indicators are the distance between the target and payload impact point, as well as Root Mean Square Error (RMSE) for trajectories. To clarify, the RMSE for all tests is measured as a Hausdorff distance from each point on the planned trajectory to the executed trajectory. Throughout the analysis, several performance indicators are measured:

- **Success rate** - The ratio of successful parabolic airdrops performed to total number of

trials. What constitutes a success is determined separately for each simulation and experimental environment.

- $RMSE_{\mathrm{T}}$ - The deviation of the executed part of the trajectory from the TOPP-RA planned trajectory.
- $RMSE_{\mathrm{L}}$ - The deviation of the executed part of the trajectory from the launch spline planned trajectory.
- $RMSE_{\mathrm{S}}$ - The deviation of the executed part of the trajectory from the stopping spline planned trajectory.
- $RMSE_p$ - The deviation of the executed payload trajectory from the planned parabolic trajectory. Higher numbers indicate a higher error in the executed launch configuration, which suggests such launch configurations are dynamically more challenging and harder to be achieved by the multirotor.
- d_{avg} - The average distance of the payload impact point from the target.
- d_{max} - The maximum distance of the payload impact point from the target.
- d_{med} - The median distance of the payload impact point from the target.

7.3.1 Parabolic Trajectory Analysis

The first part of the simulation analysis is testing the parabolic airdrop motion planning method. Multiple trajectories are planned in obstacle-free space to focus on executing planned trajectories. Since the ball radius is set to $r_b = 0.1$, the target position is kept at $\mathbf{p}_{\mathrm{T}} = \begin{bmatrix} 0 & 0 & 0.1 \end{bmatrix}^T$. When the ball impacts the ground, its center of gravity is always going to be at a height equal to the ball radius. Therefore, to properly measure the ball deviation from target, a height offset is applied to the target.

To analyze the deviation of the planned and executed parabolic airdrop trajectories, $n = 150$ different parabola configuration vectors $\mathbf{C}_p$ are prepared. These configurations are obtained through all combinations of the height displacement, launch velocity and launch angle:

$$\begin{aligned} h_p &\in \{1.0, 1.5, \ldots, 4.5\}\, m \\ v_{0,p} &\in \{1.5, 2.0, \ldots, 3.5\}\, m/s \\ \theta_p &\in \{0, 7, 14, 21\}^\circ, \end{aligned} \tag{7.30}$$

while the horizontal displacement d_p is calculated as discussed in Section 7.2.1. For simplicity, the direction angle is set to $\psi_p = 0$ and the horizontal acceleration to $a_h = 0$. Note that some configurations from the above equation are impossible to achieve. A representative example is setting $h_p = 1.0$, $v_{0,p} = 3.5$ and $\theta_p = 21^\circ$, which yields a trajectory that goes below the ground. Such configurations are not taken into account within this section.

Each trajectory configuration is executed in 10 trials, while measuring the Root Mean Square Error (RMSE) of trajectories and distance from the target. A representative set of

examples is shown on Fig. 7.7. To better visualize and compare trajectories, the planner is constrained to $x-z$ plane. It is hard to evaluate success rate without having some target with certain dimensions. In this case, a shallow bucket is placed at the target with radius $r_{\mathrm{T}} = 0.375m$. This number might seem arbitrary, however, in subsequent section this is the radius of a target bucket. All trials that end up within the designated radius from the target point are considered to be successful.

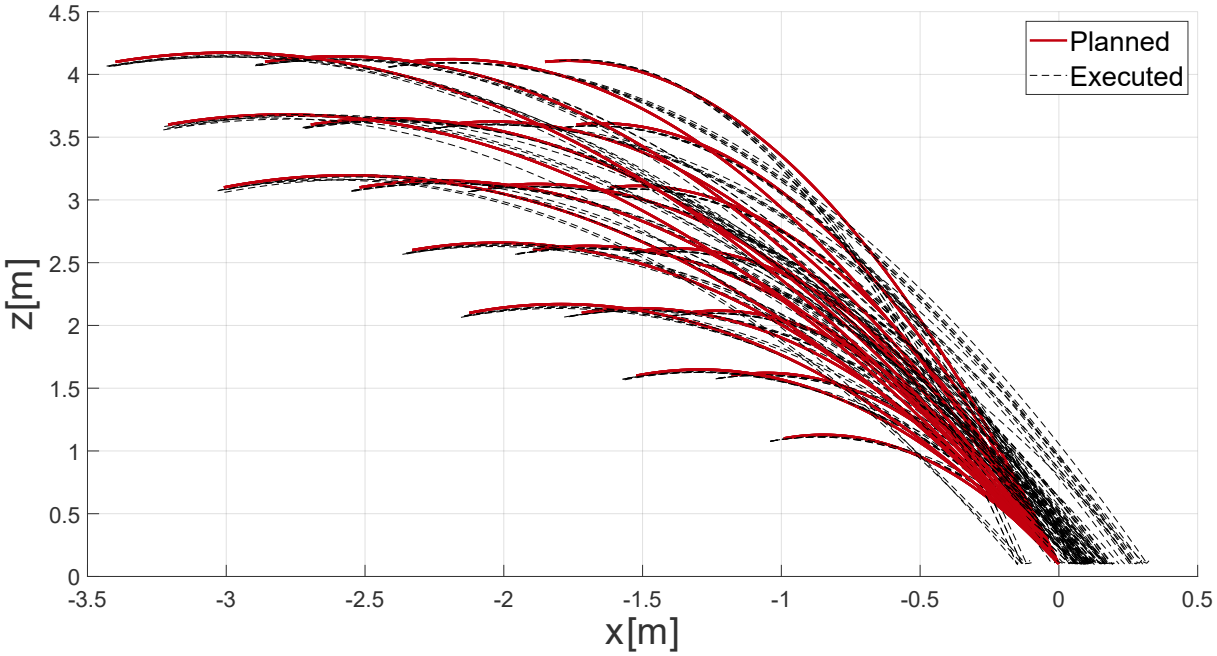


Figure 7.7: An example set of planned parabolic trajectories $\mathscr{T}_p$ and executed $\mathscr{T}_{p,e}$.

While performing parabolic airdrop trials, the multirotor dynamical constraints are set according to Table 7.1. As already discussed, the constraints of each subsequent trajectory planning stage requires higher constraints than the previous one. Note that these constraints are kept constant throughout all simulation environments. Since the environment is obstacle-free, the RRT* path planning algorithm simply returns the start and launch point. For all trials, the start point is set at $\mathbf{p}_{\mathrm{S}} = [-9.0\ 0.0\ 4.5]^T$, while the launch point varies depending on the parabola configuration vector $\mathbf{C}_p$.

Table 7.1: Velocity and acceleration constraints for all three segments of the planned trajectory. $\mathscr{T}_{\mathrm{T}}$ denotes the approach trajectory, planned through TOPP-RA. $\mathscr{T}_{\mathrm{L}}$ denotes the launch spline and $\mathscr{T}_{\mathrm{S}}$ denotes the stopping spline. All values are in standard SI units: $v[m/s]$, $a[m/s^2]$, $\omega[rad/s]$ and $\dot{\omega}[rad/s^2]$. Copyright [14] CC BY 4.0.

	v_x	v_y	v_z	ω_z	a_x	a_y	a_z	$\dot{\omega}_z$
$\mathscr{T}_{\mathrm{T}}$	2.0	2.0	1.5	1.0	1.2	1.2	0.8	1.0
$\mathscr{T}_{\mathrm{L}}$	5.0	5.0	3.0	1.0	2.5	2.5	1.0	1.0
$\mathscr{T}_{\mathrm{S}}$	8.0	8.0	3.0	1.0	3.0	3.0	1.5	1.0

The total number of executed trajectories in the obstacle-free environment is 1500, with

1104 successfully reaching the target. The performance indicators are shown in Table 7.2. The set of launch point configurations is relatively large, performing the airdrop from respectable heights and with relatively high launch velocities. Naturally, some of the launch configurations put high requirements on the system, rendering higher errors in the distance from target. This is especially true for a high velocity launch, where the multirotor velocity tracking error is higher than at low velocities. Nevertheless, the average and median distance from target are lower than the success distance d_t, which makes most of the executed payload trajectories successful. The trajectory tracking RMSE changes between the different stages of the trajectory planning. This occurs because of the differences in planning approaches between stages. A representative trajectory response in $x-z$ plane is shown on Fig. 7.8.

Table 7.2: Performance indicators of the executed trajectories in the obstacle-free environment. Copyright [14] CC BY 4.0.

Success [%]	$RMSE_{\mathrm{T}}$	$RMSE_{\mathrm{L}}$	$RMSE_{\mathrm{S}}$	$RMSE_{\mathrm{P}}$	d_{avg}	d_{max}	d_{med}
73.6	0.1299	0.1123	0.0795	0.2046	0.2717	0.9065	0.2481

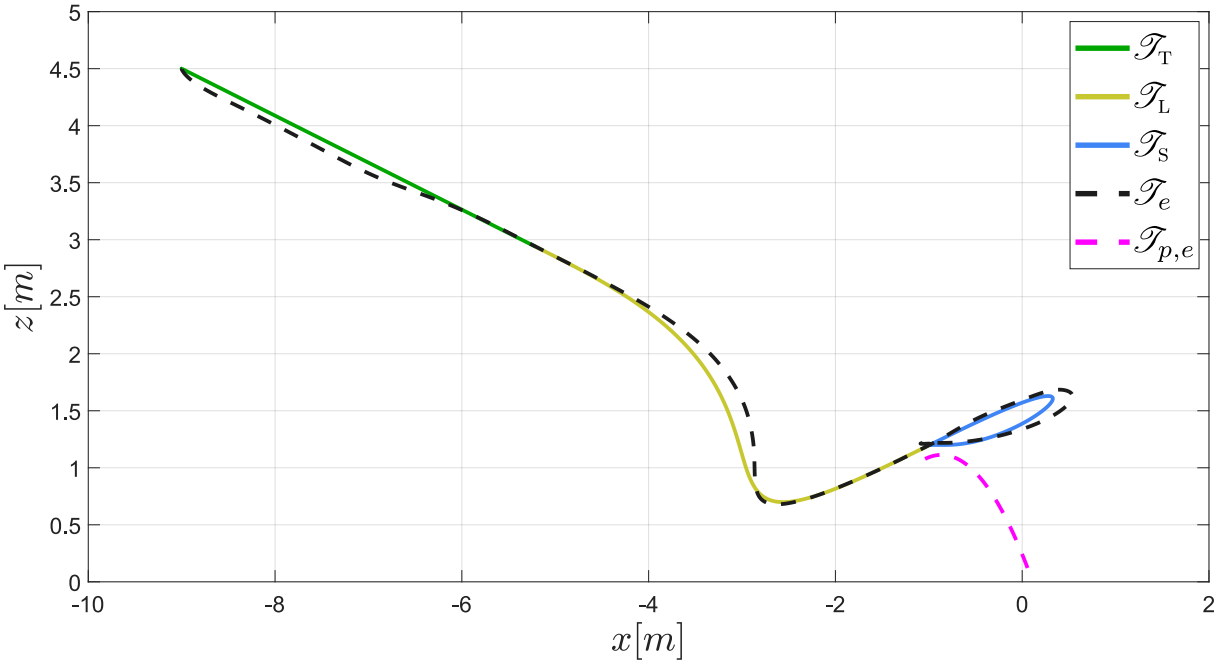


Figure 7.8: Planned and executed trajectories of the multirotor for a parabolic airdrop task. The planned trajectory is differently colored, depending on the planning stage: $\mathscr{T}_{\mathrm{T}}$ is the initial TOPP-RA trajectory, $\mathscr{T}_{\mathrm{L}}$ is the launch trajectory, and $\mathscr{T}_{\mathrm{S}}$ is the stopping trajectory. The executed trajectory is denoted with $\mathscr{T}_e$. The projectile trajectory $\mathscr{T}_{p,e}$, given in magenta, is depicted for clarity. Copyright [14] CC BY 4.0.

7.3.2 Large-scale Environment

To test how the airdrop trajectory planning performs together with a path planned through the RRT* algorithm, a large scale city environment is chosen. The envisioned practical scenario

is deploying a fire extinguishing ball to a building of known location in the city. To plan a collision-free path, it is assumed that an occupancy map of the city is known. This is a bold assumption, nevertheless, a rudimentary map can be constructed through cadastral urban maps or built with mapping algorithms based on LiDAR technology. However, building such a map goes well beyond the scope of this work, as it requires a significant effort to accurately map large environments. The focus of this section is to test how the planner performs with the RRT* path planning, OctoMap environment representation, and parabolic airdrop trajectory generation in the pipeline.

The city environment alongside several executed trajectories is shown on Fig. 7.9. The simulated urban environment is scaled down from its real size to improve clarity. This does not affect the planner since the environment still retains the same level of complexity as the full scale city. The scaled down environment is roughly the size $100m \times 100m \times 40m$. The start point is set to the far side of the city for testing the planners ability to find a feasible path and trajectory in a large environment. A total of $n = 40$ trials are performed, aiming to deliver the ball into the building on fire. For consistency, the same window is chosen in each trial. Due to the environment size, the average planned trajectory length is $l_{\mathscr{T}} = 167.79m$. The results are summarized in Table 7.3.

Table 7.3: Performance indicators of the executed trajectories in the city environment. Copyright [14] CC BY 4.0.

Success [%]	$RMSE_{\mathrm{T}}$	$RMSE_{\mathrm{L}}$	$RMSE_{\mathrm{S}}$	$RMSE_{\mathrm{P}}$	d_{avg}	d_{max}	d_{med}
85.0	0.1086	0.1402	0.0614	0.1557	0.2896	0.3275	0.2917

The results show a similar average RMSE for trajectory tracking as in Section 7.3.1. This is to be expected, as the trajectory constraints are not changed compared to the previous section. The system delivered the ball through the window in $34/40$ instances. This yields a slightly higher success rate which is attributed to the window size. However, the executed parabolic trajectory has a slightly poorer performance, but still comparable to the previous section. This discrepancy in the performance is attributed to the path planning algorithm. The RRT* generates the waypoints randomly, therefore, for each trial the trajectory has a slightly different approach. Nevertheless, the satisfactory performance is evident from the high rate of successful ball deliveries.

7.3.3 Dense Indoor Environment

To further test the parabolic airdrop motion planning method, a dense indoor environment is employed. Namely, a generic office space layout with several rooms and a common place is used. The office space with representative executed trajectories for each room is shown on Fig. 7.10. The envisioned mission is again the fire extinguishing, where the ball needs to be delivered

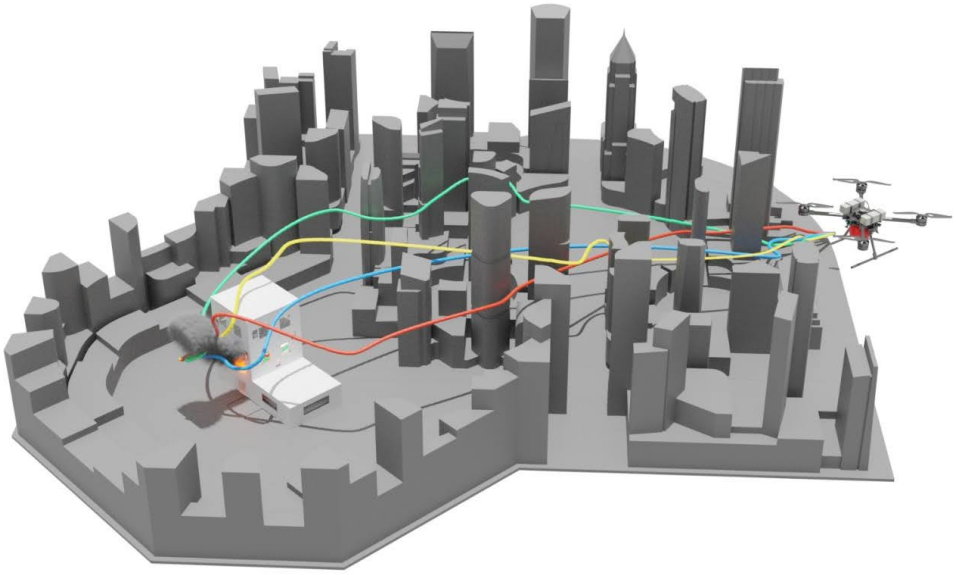


Figure 7.9: The city environment simulated in Gazebo. The multirotor is tasked to deploy a fire extinguishing ball to a building on the far side of the city. Several trajectories are shown, leading the multirotor towards the launch point. Note that the multirotor size is exaggerated for illustration purposes. Copyright [14] CC BY 4.0.

to each room of the office space, by launching it through the door. From a planning perspective, such an environment is rather challenging since it has narrow passages and limited height. Therefore, the multirotor needs to be capable to abruptly stop after performing a parabolic airdrop near an obstacle.

Naturally, the planner's task is to find a feasible parabolic airdrop trajectory with a launch configuration that does not drive the multirotor into an obstacle. As this is quite challenging environment, the feasible solutions included a non-zero horizontal trajectory a_h, discussed in Section 7.2.1. To recall, this parameter instructs the multirotor to start decelerating before it releases the ball. The resulting stopping trajectory is therefore shorter, without colliding with the environment.

The first test performed is delivering the payload to each room of the office environment. A total of 120 trials are performed, with 20 trials for each room of the office. The average trajectory length measures at $l_{\mathscr{T}} = 38.35m$. The results are captured within Table 7.4. The trajectory tracking error is comparable to the large scale environment. A slight increase in the launch trajectory deviation can be attributed to the non-zero horizontal acceleration a_h at the launch configuration. This requires planning a more aggressive trajectory that is tracked with a higher error. On the other hand, there is a significant decrease in precision when it comes to the deviation of the payload impact point from the target. This is again attributed to the horizontal acceleration which poses high dynamical requirements at the launch instance. Having the me-

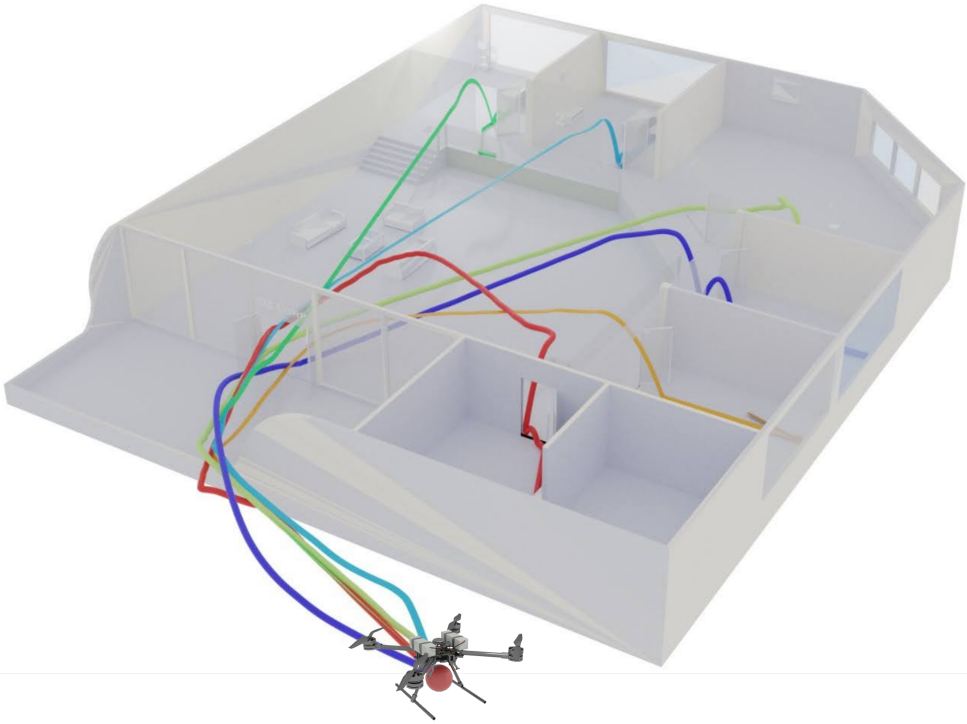


Figure 7.10: Representative examples of trajectories executed to each room of the office environment. The most challenging part of the trajectory is passing through the narrow entrance before the office common space. Copyright [14] CC BY 4.0.

dian deviation less than the mean deviation suggests that the system performs well in most of cases, however, when it misses the target the deviation is rather large. Nevertheless, the system managed to deploy the ball into a room in $87/120$ trials.

A second set of tests has been performed aiming at a bucked placed in one room of the office, totalling in 40 more trials. The radius of the placed bucket is $r_{bucket} = 0.375m$ and the ball radius is $r_b = 0.1m$. With such parameters the system has been able to perform a successful parabolic airdrop in $26/40$ trials. The results are captured within the second row of Table 7.4.

Table 7.4: Performance indicators of the executed trajectories in the office environment. The second row denotes the trials for delivering the ball to a bucket placed in the largest room of the environment. Copyright [14] CC BY 4.0.

Type	Success [%]	$RMSE_{\mathrm{T}}$	$RMSE_{\mathrm{L}}$	$RMSE_{\mathrm{S}}$	$RMSE_{\mathrm{P}}$	d_{avg}	d_{max}	d_{med}
Rooms	72.5	0.0894	0.1522	0.0728	0.4167	0.4723	0.9936	0.4389
Bucket	65.0	0.0872	0.1405	0.0646	0.2643	0.3076	0.7914	0.2837

7.3.4 Algorithm Runtime

As the parabolic airdrop motion planning algorithm has been tested in the simulation, the planning times of each stage are given in Table 7.5. The RRT* planning step does not occur for the obstacle-free space, hence the zero planning time. The path planning times in the office

Table 7.5: Average execution time of different planning procedure steps, expressed in seconds. Note that RRT* step does not occur in an empty environment, thus, the planning time is zero. Copyright [14] CC BY 4.0.

Step \ Env	Empty	City	Office
RRT*	0	7.944	33.165
TOPP-RA	0.022	0.127	0.054
Launch	0.749	2.611	1.969
Stopping	0.066	0.123	0.091
Total	0.837	10.805	35.729

environment are greater than the city environment. Although that might seem counter intuitive due to the difference in size, it is expected since the city does not have a narrow passage where the RRT* struggles to pass through. On the contrary, the planning time of TOPP-RA is lower for the office environment than the city. This is the direct consequence of the longer path length in the city environments, which increases the number of discretization points when planning the TOPP-RA trajectory. The launch spline planning time mostly depends on how many launch configurations are feasible. In an empty environment, all configurations can be achieved and the launch spline planning time is the lowest. In both city and office environment, the launch spline is longer since the appropriate start configuration along the initial TOPP-RA trajectory must be found. This configuration can pose rather challenging initial conditions on the launch spline planning. Lastly, the stopping spline planning time only slightly varies between the environments. These slight variations are induced by the initial velocity and acceleration, which are defining the subgradient search time.

7.4 Experimental Verification

After analyzing the trajectory planning and overall system performance in the simulation environment, an extensive experimental verification is conducted. First, experiments were performed in an indoor laboratory environment using the AscTec NEO hexacopter, introduced in Section 5.2.2. This is followed by a set of outdoor experiments using a custom built quadcopter developed by company Kopterworx. The goal in both cases was to deploy a ball into a box. The performed experiments consist of repeatability analysis in an obstacle-free environment and obstacle avoidance.

As the custom built Kopterworx quadcopter is used in this section, it is briefly introduced here. The multirotor features four T-motor P60 KV170 motors, equipped with $22in$ folding propellers. The vehicle dimensions are $1.2m \times 1.2m \times 0.45m$ with total mass $m = 9kg$ including all electronics and batteries. The larger vehicle scale enables a larger payload capacity, including the possibility of mounting more sensors and higher battery capacity that allows for $30min$ of

non-interrupted flight time. The onboard flight controller is the ProfiCNC Pixhawk 2.1 running the ArduPilot flight stack, together with the HEX-Kore high current power board capable of adequately supplying the propulsion system. Furthermore, an Intel NUC onboard computer is mounted on the multirotor, which allows for running the computationally expensive high level applications, such as planning and mapping. The onboard computer is running on Linux Ubuntu 18.04 LTS with the ROS Melodic middleware. The vehicle also features a Velodyne Puck LITE 16-channel LiDAR and an external LPMS-CU2 Inertial Measurement Unit (IMU), which are uitilized together for simulataneous localization and mapping. Both multirotors used for the experimental verifications are shown on Fig. 7.11.

(a) AscTec NEO hexacopter.

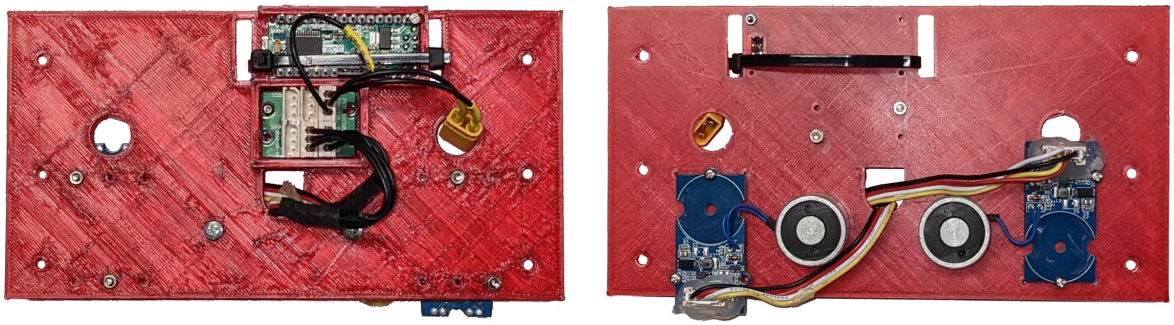
(b) Custom built Kopterworx quadcopter.

Figure 7.11: Multirotors with their sensory apparatus used for performing the parabolic airdrop in real world experiments.

7.4.1 Magnetic Gripper Design

To release the ball, a magnetic gripper is designed as shown on Fig. 7.12. The gripper consists of two electromagnetic elements developed by Grove Electromagnet. Each unit has the peak electromagnetic force of $10N$, and operates on $U = 5V$ voltage and $I = 400mA$ peak current while active. The amount of the force produced by each electromagnet is more than enough to hold the payload, which has the mass of $m_p = 0.3kg$ and radius $r_b = 0.05m$. However, due to potential sharp turns and aggressive maneuvers, the gripper force is purposely oversized. As the ball used for the payload is made of a plastic material, a ferromagnetic washer is connected to it as a contact point for the gripper attachment. The gripper size of $16cm \times 7.5cm$ allows for easy mounting on both multirotors used in performing the parabolic airdrop experiments.

Each electromagnet has a signal pin to trigger it on and off, which is connected to a digital pin of an Arduino Nano board. This allows for an external electromagnet triggering, offering a near-instantaneous release of the payload upon request. The advantage of using an Arduino board is the ability to utilize the *rosserial_arduino* library. The Arduino board is connected to the onboard computer running the ROS middleware, which allows triggering the electromagnet through a ROS topic.



(a) Top plate of the magnetic gripper with the Arduino Nano controller and power distribution board.

(b) Bottom plate of the magnetic gripper with Grove Electromagnet controller boards.

Figure 7.12: The electromagnetic gripper design featuring two Grove Electromagnet elements, and an Arduino Nano board for communication with the onboard computer.

7.4.2 Indoor Laboratory Environment

To test the method on a real world systems, first experiments are performed indoor, in a laboratory environment. The AscTec NEO hexacopter is endowed with the magnetic magnetic gripper developed within the previous section to perform the indoor experiments. The objective is to perform a parabolic airdrop with the target in form of a box of dimensions $0.33m \times 0.28m \times 0.23m$, placed on the ground. To provide the feedback for the multirotor controllers, the Optitrack motion capture system is used. Since the real world multirotor and the laboratory environment differ from the simulation, a set of dynamical constraints imposed on the system is given in Table 7.6. These constraints are a bit more conservative compared to the simulation, mainly due to the limited size of the flight arena.

Table 7.6: Velocity and acceleration constraints for all three segments of the planned trajectory. $\mathscr{T}_\mathrm{T}$ denotes the approach trajectory, planned through TOPP-RA. $\mathscr{T}_\mathrm{L}$ denotes the launch spline and $\mathscr{T}_\mathrm{S}$ denotes the stopping spline. All values are in standard SI units: $v[m/s]$, $a[m/s^2]$, $\omega[rad/s]$ and $\dot{\omega}[rad/s^2]$. Copyright [14] CC BY 4.0.

	v_x	v_y	v_z	ω_z	a_x	a_y	a_z	$\dot{\omega}_z$
$\mathscr{T}_\mathrm{T}$	2.0	2.0	1.2	1.0	0.8	0.8	0.5	0.8
$\mathscr{T}_\mathrm{L}$	4.0	4.0	3.0	2.0	1.0	1.0	1.0	1.0
$\mathscr{T}_\mathrm{S}$	5.0	5.0	3.0	2.0	2.0	2.0	1.5	1.0

Two different sets of experiments are performed in the laboratory environment: repeatability analysis and obstacle avoidance. In both cases, a static map of the environment, shown on Fig. 7.13 is created for ensuring collision-free trajectories. For purposes of testing the system repeatability, the obstacle in the middle of the environment is removed from the arena and the OctoMap. A total of 20 straight line trajectories with various parabola configuration vectors $\mathbf{C}_p$ have been tested. The results are summarized in Table 7.7. The system managed to perform a successful parabolic airdrop in 16/20 trials. The RMSE of different parts of the trajectory is

larger than the one reported in simulation. This is rather expected since the real world system is susceptible to various uncertainties, and is still the subject of quite challenging dynamical constraints.

Unlike in the simulation environment, directly measuring the payload trajectory is fairly complex. Nevertheless, it is possible to reconstruct the parabolic trajectory using the payload position and velocity at the launch instance, by employing the equation (7.6). The only unknown parameter is the impact time, which can be approximately determined when the payload height is equal to the target height. Note that this reconstruction depends on the multirotor estimated state, therefore it is prone to errors. However, the success rate obtained with the reconstructed parabolic payload trajectory is consistent with the observed outcome (hit or miss). The impact distance from the target obtained with the described reconstruction method shows relatively small deviation from the target. This is mainly due to the AscTec NEO flight performance, as well as the consistent millimeter precision of the Optitrack motion capture system.

After the repeatability analysis, the planner is tested for the obstacle avoidance task. An obstacle is placed in the middle of arena to navigate the system around it. The target position remains unchanged for this task. The system performed 12 trajectories with the identical starting point and various parabola configurations $\mathbf{C}_p$. Several examples are shown on Fig. 7.13, where the system is deliberately instructed to navigate around the left side of the obstacle because the operator location is on the right. The system managed to perform a successful airdrop in $11/12$ instances, with trajectory tracking errors shown in Table 7.7. Using the parabola reconstruction method, the distance from the target and the parabolic trajectory deviation are obtained. The results are very similar to the repeatability analysis trials, indicating that the system is able to consistently perform the parabolic airdrop task.

Table 7.7: Performance indicators of the executed trajectories for the indoor environment. The table compares the repeatability analysis and obstacle avoidance trials. Copyright [14] CC BY 4.0.

Type	Success [%]	$RMSE_{\mathrm{T}}$	$RMSE_{\mathrm{L}}$	$RMSE_{\mathrm{S}}$	$RMSE_{\mathrm{P}}$	d_{avg}	d_{max}	d_{med}
Rep	80.0	0.1720	0.2909	0.1465	0.1494	0.1885	0.3357	0.1880
Obs	91.7	0.0751	0.1975	0.1632	0.2080	0.2273	0.3286	0.2410

7.4.3 Outdoor Environment

The final step in testing the parabolic airdrop trajectory planning method are experiments in an outdoor environment. To perform these experiments, a large scale multirotor described in Section 7.4 is used. Unlike the indoor environment, localizing the multirotor is a more complex task in the outdoor environment. One common way of localization is the Global Positioning System (GPS). However, most GPS receivers do not provide precise positioning, which is usually within a few meters margin of error. A base station can improve those measurements, yet

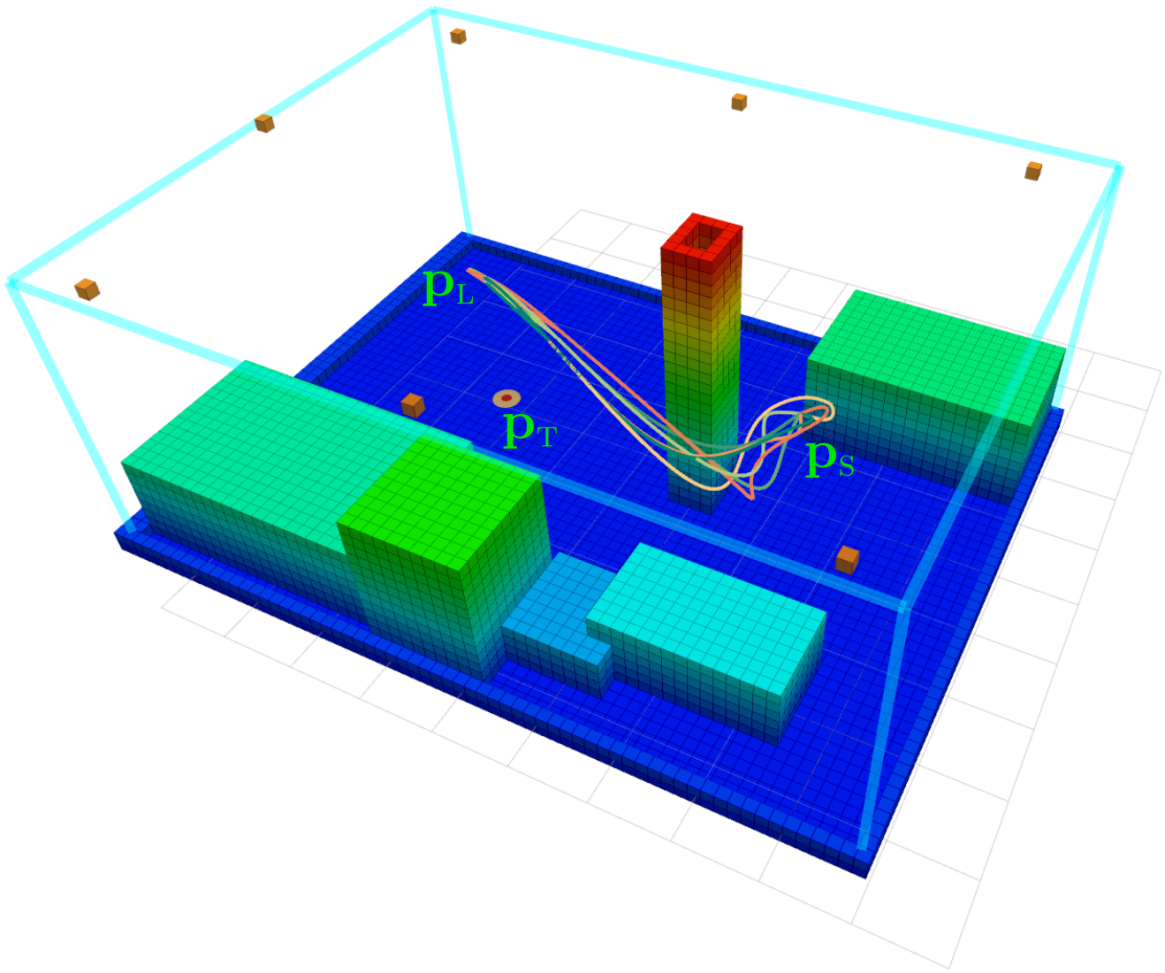


Figure 7.13: Several obstacle avoidance trajectories performed in the indoor environment, together with the arena OctoMap. Several prismatic obstacles are present at the side of the arena, and in the middle is the manually placed obstacle to test the planner avoidance. Orange blocks are the Optitrack motion capture system cameras mounted in the arena. Copyright [14] CC BY 4.0.

it complicates the system and setup.

To provide the position, orientation, and velocity feedback for the multirotor, an onboard Simultaneous Localization and Mapping (SLAM) algorithm is used. Namely, the Cartographer SLAM is employed, which requires a LiDAR generated point cloud and IMU measurements. As already mentioned, the Kopterworx quadcopter is equipped with the Velodyne VLP-16 LiDAR and LPMS CU2 IMU, sufficient for the Cartographer algorithm. The work presented in [143] compares different SLAM algorithms, including the Cartographer, while the researchers in [144] further adapt the Cartographer algorithm for implementation on the multirotor vehicles. Using the findings from the two aforementioned papers, the Cartographer is effectively employed for providing feedback and mapping the environment. To build the OctoMap of the environment, the internal submaps of the Cartographer are employed and joined together, shown on Fig. 7.14. By excluding the GPS measurement, this approach can be used in GPS-denied environments.

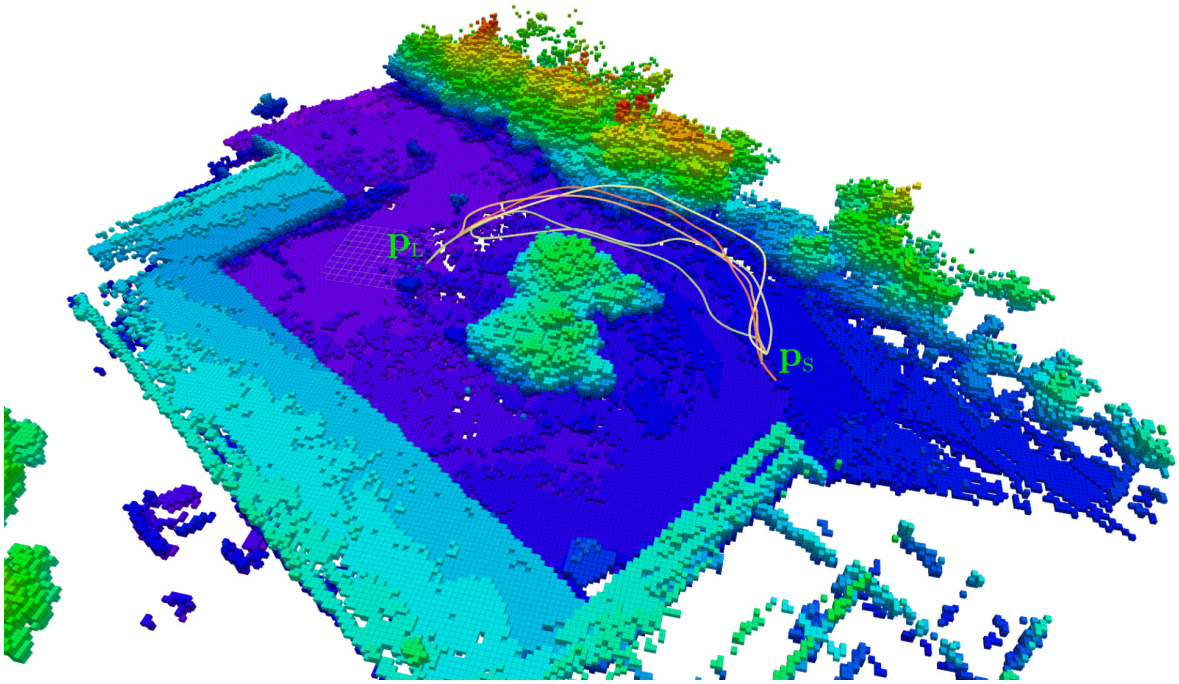


Figure 7.14: Several obstacle avoidance trajectories performed in the outdoor environment, together with the OctoMap built by the Cartographer SLAM algorithm. All trajectories navigate around the central obstacle. Note that the starting point $\mathbf{p}_S$ slightly varies between trials. Copyright [14] CC BY 4.0.

The outdoor environment is roughly the size of $50m \times 80m \times 15m$, shown on Fig. 7.14. With the maximum height of $15m$, the system would easily navigate above the central obstacle. To force the planner to steer around the obstacle, the maximum height is limited to $8m$. As the planner relies on the OctoMap for collision checking, the environment is mapped prior to trials by performing a manual flight. An environment exploration method, such as the one presented in [97], can be employed to autonomously obtain the environment map. However, manual flight has been chosen as employing such a method goes beyond the scope of this work. The target is a wooden crate of size $1.3m \times 1.0m \times 0.7m$ placed on the ground. This is a larger target than the one used in the indoor environment. However, the larger size is chosen due to the unexpected effects of the outdoor environment, i.e. wind gusts, and the fact that onboard positioning system is used which also introduces some uncertainties. As the Kopterworx multirotor differs from the AscTec NEO used indoors, a different set of constraints is imposed, outlined within Table 7.8.

Similar to the indoor environment, the repeatability analysis and obstacle avoidance tests are performed in the outdoor environment. The repeatability analysis is performed over 56 straight line trajectories with various parabola configuration vectors $\mathbf{C}_p$. As the map is built each time before performing one or more experiments, the target and starting positions are slightly varied over multiple trials. The results are outlined in Table 7.9. The system managed to perform a successful parabolic airdrop in $39/56$ instances. The trajectory tracking error is higher than the one obtained for the indoor environment. This is expected as the outdoor environment is

Table 7.8: Velocity and acceleration constraints for all three segments of the planned trajectory. $\mathscr{T}_\mathrm{T}$ denotes the approach trajectory, planned through TOPP-RA. $\mathscr{T}_\mathrm{L}$ denotes the launch spline and $\mathscr{T}_\mathrm{S}$ denotes the stopping spline. All values are in standard SI units: $v[m/s]$, $a[m/s^2]$, $\omega[rad/s]$ and $\dot{\omega}[rad/s^2]$.

	v_x	v_y	v_z	ω_z	a_x	a_y	a_z	$\dot{\omega}_z$
$\mathscr{T}_\mathrm{T}$	1.5	1.5	0.75	1.0	0.5	0.5	0.5	0.8
$\mathscr{T}_\mathrm{L}$	4.0	4.0	3.0	2.0	0.8	0.8	0.6	1.0
$\mathscr{T}_\mathrm{S}$	4.0	4.0	4.0	2.0	0.8	0.8	0.8	1.0

more challenging and features unpredictable disturbances such as wind gusts. The distance from target error is obtained by reconstructing the parabolic trajectory, described in Section 7.4.2. A rather high error in distance from target is obtained, which is expected in the outdoor environment. Note that the higher success rate is the consequence of having a larger target.

After the repeatability analysis, the obstacle avoidance is performed in an outdoor environment. The OctoMap of the environment together with several executed obstacle avoidance trajectories is shown on Fig. 7.14. A total of 6 trajectories are executed for the obstacle avoidance with the success in $3/6$ instances. The results are outlined in the second row of Table 7.9. Compared to the simulation and real world experiments, the least precise results are obtained while avoiding obstacles in the outdoor environment. However, this is by far the most challenging experiment performed for testing the parabolic airdrop motion planner. It features obstacle avoidance with the Cartographer SLAM feedback. The long trajectories planned are affecting the Cartographer with loop-closures and drifts, which can introduce a shift of the map and therefore a shift of the target. Nevertheless, the system managed to successfully deliver the payload in multiple occasions.

Table 7.9: Performance indicators of the executed trajectories for the outdoor environment. The table compares the repeatability analysis and obstacle avoidance trials.

Type	Success [%]	$RMSE_\mathrm{T}$	$RMSE_\mathrm{L}$	$RMSE_\mathrm{S}$	$RMSE_\mathrm{P}$	d_{avg}	d_{max}	d_{med}
Rep	69.6	0.1901	0.2595	0.1531	0.5124	0.6191	1.8303	0.5256
Obs	50.0	0.3642	0.4650	0.3561	0.8844	1.0962	1.6499	1.0833

7.5 Discussion

In both simulation and experimental verification, the same tests were performed to analyze the system's performance. The combined results can be observed in Table 7.10, and the statistical box-and-whiskers plot is shown on Fig. 7.15. The presented data shows a lot of similarities with the indoor experiments, confirming the realistic character of the simulation environment. The only part where the simulation underperforms is the success rate of delivering the ball to the bucket in the office environment. This is the most challenging simulation task because the

RRT* algorithm is required to plan a path through the environment and into the room where the bucket is located. The resulting path occasionally featured some sharp turns near the end of the trajectory which directly impacts the launch trajectory.

Furthermore, the difference in the performance between the simulation and the real world can also be attributed to different types of vehicles and feedback used in certain tasks. The target position was in both cases measured by a human operator, and this measurement error is embedded in the final result. The Optitrack system provides a very precise feedback opposed to the Cartographer SLAM feedback. These factors mainly influence the difference in the performance between the indoor and outdoor real-world environments. The AscTec NEO was used in the indoor environment. The vehicle itself can perform quite agile and aggressive maneuvers through the on-board attitude controller and the position MPC. The trajectory tracking RMSE for the indoor scenario is similar to ones reported in the literature. Researchers in [145] report RMSE of 17.53cm for a circular trajectory and 11.27cm for a lemniscate trajectory. In [146] an RMSE of 16.8cm is reported for a trajectory without jerk and snap tracking. In both cases the experiments were conducted indoors with the Optitrack motion capture system as feedback. The indoor setup performance from this thesis is in line with the tracking errors reported in the former research paper, which is the main factor for high success rate.

The outdoor experiments have the lowest success rate even though the target was the largest among all performed tests. Fig. 7.15 shows how the outdoor system has higher uncertainty than all others. However, there are several factors one has to take into account. This is the most challenging environment of all because the localization is done only with the onboard sensors. Furthermore, the environment map is a-priori unknown and is built by flying manually before executing the task. This map is then used in the planning procedure, as well as for determining the position of the target. All these factors contribute to the uncertainty and success rate. One can observe a slightly higher drop in success rate for the outdoor obstacle avoidance task. This behavior is somewhat expected since this is the most challenging task imposed on the system. The main difference when compared to the outdoor repeatability analysis is the possibility of a loop closure along the relatively longer trajectory, which the Cartographer SLAM performs during the flight. These loop closures are shifting the map of the environment, and therefore the target box, which decreases the success rate of the airdrops. One can argue that the system can take advantage of hovering when performing the drop, especially in case of outdoor experiments where no obstacles were around the target. However, the aim of these experiments is to obtain real-world data to assess planner and overall system performance, keeping in mind the initial intention to deliver the fire extinguishing agent through window or door. Nevertheless, the system managed to perform the parabolic airdrop in most of the experiments.

Table 7.10: This table contains data from all conducted simulation and real-world experiments in one place. The success rate is expressed in percentages for easier comparison. Apart from the success rate, the measurement unit for all other fields is meter *m*. Copyright [14] CC BY 4.0.

	Environment	Success	$RMSE_{\mathrm{T}}$	$RMSE_{\mathrm{L}}$	$RMSE_{\mathrm{S}}$	$RMSE_{\mathrm{P}}$	d_{avg}	d_{max}	d_{med}
Simulation	Repeatability	73.6	0.1299	0.1123	0.0795	0.2046	0.2717	0.9065	0.2481
	City	85.0	0.1086	0.1402	0.0614	0.1557	0.2896	0.3275	0.2917
	Office	72.5	0.0894	0.1522	0.0728	0.4167	0.4723	0.9936	0.4389
	Office bucket	65.0	0.0872	0.1405	0.0646	0.2643	0.3076	0.7914	0.2837
Real world	Indoor	80.0	0.1720	0.2909	0.1465	0.1494	0.1885	0.3357	0.1880
	Indoor obs	91.7	0.0751	0.1975	0.1632	0.2080	0.2273	0.3286	0.2410
	Outdoor	69.6	0.1901	0.2595	0.1531	0.5124	0.6191	1.8303	0.5256
	Outdoor obs	50.0	0.3642	0.4650	0.3561	0.8844	1.0962	1.6499	1.0833

7.5.1 Sources of uncertainty

In this section it is pointed out that there are several significant sources of uncertainty that affect the executed parabola in both simulation and experiments. First, there is feedback noise which directly affects the position and the velocity at the launch point. Second, higher velocity and acceleration specified at the launch point will result in a more challenging trajectory with a slightly higher error when reaching the desired launch configuration. Third, the planner assumes zero pitch angle at the launch point. However, this will not be the case while executing the trajectory. Due to the payload displacement from the UAV center of mass $\mathbf{T}_{\mathrm{B}}^{\mathrm{P}}$, the non-zero pitch angle while executing the trajectory is introducing the error in the payload position. Fourth, the planner also assumes zero angular rates at the launch point which will not be the case due to disturbances and localization errors. Angular rates result in some tangential velocity, thus, increasing the uncertainty of the payload velocity. Fifth(simulation only), due to the coupling between the ball and the UAV induced by the magnetic force of the simulated magnet, the release position and the release velocity of the ball differ from the planned ones. Small errors at the release point consequently result in errors at the impact point which then differs from the target point.

All of the aforementioned reasons result in some error in the launch point configuration because they affect initial conditions of the release, and ultimately decrease precision towards the target point. Through the extensive simulation and experimental analyses, the observed cumulative effect of these uncertainties through presented performance indicators. Even though uncertainties have some effect on the overall results, it is still possible to achieve a successful parabolic airdrop even in outdoor environments.

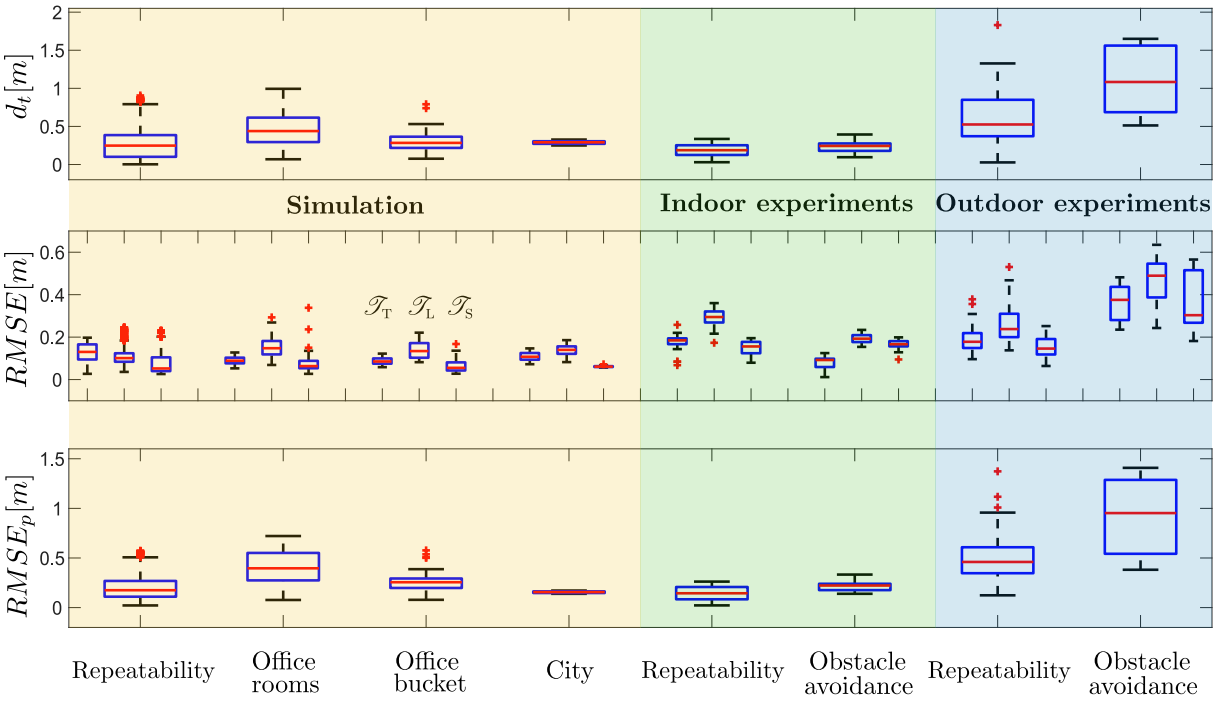


Figure 7.15: Box-plot of the data obtained through the simulation and the experimental analysis. The top plot shows the distance from the target. The middle plot shows the RMSE of trajectories for all the conducted tests. Note that the three stages of the trajectory planning are shown. For each test, the left bar is the TOPP-RA trajectory, the middle bar is the launch spline and the right bar is the stopping trajectory. The bottom plot shows the RMSE between the planned and the executed parabolic trajectories. Copyright [14] CC BY 4.0.

CHAPTER 8

Conclusion

The problem in focus of this thesis is planning a feasible motion for aerial manipulators while respecting both spatial and dynamical constraints. The common steps in the motion planning procedure are path planning based on the input waypoints, and trajectory planning based on the underlying path. The spatial constraints are provided as an environment map, allowing the path planner to avoid obstacles, while the dynamical constraints are provided in terms of velocity and acceleration for each degree of freedom. The principle behind this procedure can be applied to virtually any class of robotic systems, while in this thesis aerial manipulators are used. The three main scientific contributions are laid out in the reminder of this section.

Unmanned aerial manipulator end-effector motion planning method based on the dynamical model of the system

The first contribution provides a motion planning procedure based on the dynamical model of an aerial manipulator. Typically, the desired end-effector trajectory is specified by the user or determined by the path planning algorithm. When a co-planar multirotor executes any trajectory, it is inevitably tilting due to its underactuated nature. This motion is not captured in the path planning step, which leads to the end-effector deviation from the desired configuration while executing a trajectory. Using an aerial manipulator dynamical model in the motion planning procedure, the full state of the system is obtained. The end-effector configuration can be corrected through the manipulator null space, achieving the desired configuration.

Although the end-effector corrections can be applied online, it is possible that the desired configuration is not within the manipulator workspace, or that it is far from the current manipulator joint configuration. This can lead to unwanted end-effector motion or even overloading actuators, which can compromise the task at hand. These effects can be regarded within the motion planning, by checking the trajectory after applying corrections. The final trajectory

obtained in this way is not sent to the system if the motion is not feasible. The main limitation of this method is the underlying assumption that the manipulator dynamics operate on higher bandwidth than the multirotor. Applying the corrections with a low bandwidth manipulator dynamics and aggressive multirotor maneuvers inevitably leads to unfeasible manipulator motion, and such a plan is then discarded by the motion planner. In any case, it is better to detect infeasible motion during a planning step rather than while executing the trajectory. The second limitation is planning time, which includes the Gazebo simulation in the planning procedure. Although Gazebo encompasses the system dynamics realistically, it is not suitable for fast simulations. Therefore, deploying this method on a companion computer mounted on a multirotor requires simulating dynamics through the devised aerial manipulator mathematical model, preferably implemented in C++ programming language to reduce the computing time.

Nevertheless, this method has a great potential in real world applications. The envisioned scenario of pipe insertion is an attractive option for inspection of hard-to-reach pipes with aerial manipulators. Augmenting the aerial manipulator with force sensing ability widens the scope of real world applications. Employing the impedance force control allows aerial manipulators to achieve and maintain wall contact. One potential application is bridge inspection, where a specialized machinery is used to attach sensors for monitoring the bridge state, often requiring road closure. Aerial manipulators can be used to apply adhesive and glue the required sensors on the bridge structure. A magnetic localization principle developed in [147] can be employed to find the sensor on the bridge and download the logged data. Force sensing further allows a number of applications, for example, tightening a bolt, ultrasonic wall or hull inspection, power line infrastructure inspection, etc. The developed motion planning method is helpful in such scenarios as it corrects the end-effector configuration for a straight approach.

A heterogeneous multi-robot system motion planning method based on the coupled dynamical model for cooperative manipulation

The aerial manipulator model based planning method is then extended to a multi-robot system, where multiple end-effectors manipulate the same object. In this case, the end-effector corrections are applied for each robot separately. Even though this method is tested on two aerial manipulators transporting a common object, both mathematical model and implementation are composed in a general way. The mathematical model is comprised of 6-DoF base and a manipulator, where the base is considered to be floating for aerial manipulator. Should a static manipulator be used, the base can simply be considered fixed, ignoring its degrees of freedom. This principle has been kept in mind while implementing the algorithm, making it straightforward to include a fixed base manipulator.

The envisioned scenario and the greatest potential of this method is transporting a common object. Although it is much more complicated to transport an object with multiple aerial manip-

ulators compared to a single vehicle, it is necessary in some situations. The transported object can be too heavy to be lifted by a single vehicle, or the environment configuration requires changing the object tilt for navigating narrow passages. In any case, realizing this potential is a complex task comprised of uncertainties in modelling, control, localization, and coupling forces, making it hard to apply in an arbitrary scenario. Nevertheless, first research scenarios with reliable feedback are well under way, which is the foundation of future applications. The method developed within this contribution aims to aid the transition to an arbitrary scenario by carefully applying model based corrections for a transportation task.

Unmanned aerial manipulator motion planning method with dynamical constraints at the release point for precision ballistic airdrop

The third contribution differs from the former two by employing a flying hand instead of an aerial manipulator. A simpler mechanical design of such a vehicle makes it ideal for less complex scenarios. The envisioned task in this thesis is deploying a fire extinguishing agent, while regarding the vehicle safety. Based on the target position in the world, the motion planner generates a set of suitable launch configurations and plans both path and trajectory towards the first feasible configuration. The target is considered to be known since this thesis focuses on the motion planning, however, various vision or LiDAR based detection methods can be employed to obtain the target position. A simple magnetic gripper is designed and employed for transporting the fire extinguishing agent, releasing it upon request generated by the motion planner.

Although simplest in terms of aerial manipulation, the employed flying hand has a great potential in application scenarios. In a fire extinguishing scenario, one vehicle deploying a fire extinguishing agent is hardly enough to stop the fire. Especially considering the limited payload of such vehicles. Having a team of such vehicles gives the opportunity to deploy the agent continuously, which greatly increases chances of successfully extinguishing or containing fire until the first responders arrive.

Bibliography

[1] Quan, Q., Introduction to multicopter design and control. Springer Singapore, 2017, available at: https://doi.org/10.1007/978-981-10-3382-7

[2] Tahar, K. N., Ahmad, A., Akib, W. A. A. W. M., "Uav-based stereo vision for photogrammetric survey in aerial terrain mapping", in 2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE), 2011, pp. 443-447.

[3] Budiharto, W., Irwansyah, E., Suroso, J. S., Chowanda, A., Ngarianto, H., Gunawan, A. A. S., "Mapping and 3d modelling using quadrotor drone and GIS software", Journal of Big Data, Vol. 8, No. 1, Mar. 2021, available at: https://doi.org/10.1186/s40537-021-00436-8

[4] Daponte, P., De Vito, L., Mazzilli, G., Picariello, F., Rapuano, S., "A height measurement uncertainty model for archaeological surveys by aerial photogrammetry", Measurement, Vol. 98, 2017, pp. 192-198, available at: https://www.sciencedirect.com/science/article/pii/S0263224116306789

[5] Ereiz, S., Bartolac, M., Goricanec, J., Orsag, M., "Application of UAVs for assessment of bridge infrastructure", Journal of the Croatian Association of Civil Engineers, Vol. 73, No. 11, Dec. 2021, pp. 1095–1106, available at: https://doi.org/10.14256/jce.3254.2021

[6] Lei, B., Wang, N., Xu, P., Song, G., "New crack detection method for bridge inspection using UAV incorporating image processing", Journal of Aerospace Engineering, Vol. 31, No. 5, Sep. 2018, available at: https://doi.org/10.1061/(asce)as.1943-5525.0000879

[7] Car, M., Markovic, L., Ivanovic, A., Orsag, M., Bogdan, S., "Autonomous wind-turbine blade inspection using lidar-equipped unmanned aerial vehicle", IEEE Access, Vol. 8, 2020, pp. 131 380-131 387.

[8] Shukla, A., Xiaoqian, H., Karki, H., "Autonomous tracking and navigation controller for an unmanned aerial vehicle based on visual data for inspection of oil and gas pipelines",

in 2016 16th International Conference on Control, Automation and Systems (ICCAS), 2016, pp. 194-200.

[9] Wu, Y., Qin, Y., Wang, Z., Jia, L., "A uav-based visual inspection method for rail surface defects", Applied Sciences, Vol. 8, No. 7, 2018, available at: https://www.mdpi.com/2076-3417/8/7/1028

[10] Jozkow, G., Toth, C., Grejner-Brzezinska, D., "UAS TOPOGRAPHIC MAPPING WITH VELODYNE LiDAR SENSOR", ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. III-1, Jun. 2016, pp. 201–208, available at: https://doi.org/10.5194/isprs-annals-iii-1-201-2016

[11] Guan, H., Sun, X., Su, Y., Hu, T., Wang, H., Wang, H., Peng, C., Guo, Q., "Uav-lidar aids automatic intelligent powerline inspection", International Journal of Electrical Power & Energy Systems, Vol. 130, 2021, pp. 106987, available at: https://www.sciencedirect.com/science/article/pii/S0142061521002271

[12] Bolourian, N., Soltani, M. M., Albahri, A., Hammad, A., "High level framework for bridge inspection using lidar-equipped uav", in Proceedings of the 34th International Symposium on Automation and Robotics in Construction (ISARC), Cheng, M.-Y. N. T. U. o. S., Technology), Chen, H.-M. N. T. U. o. S., Technology), Chiu, K. C. N. T. U. o. S., Technology), (ur.). Taipei, Taiwan: Tribun EU, s.r.o., Brno, July 2017, pp. 683-688.

[13] Imdoukh, A., Shaker, A., Al-Toukhy, A., Kablaoui, D., El-Abd, M., "Semi-autonomous indoor firefighting uav", in 2017 18th International Conference on Advanced Robotics (ICAR), 2017, pp. 310-315.

[14] Ivanovic, A., Orsag, M., "Parabolic airdrop trajectory planning for multirotor unmanned aerial vehicles", IEEE Access, Vol. 10, 2022, pp. 36 907-36 923.

[15] Siciliano, B., Caccavale, F., Zwicker, E., Achtelik, M., Mansard, N., Borst, C., Achtelik, M., Jepsen, N. O., Awad, R., Bischoff, R., "Euroc - the challenge initiative for european robotics", in ISR/Robotik 2014; 41st International Symposium on Robotics, 2014, pp. 1-7.

[16] "DARPA subterranean challenge", https://www.subtchallenge.com/, accessed: 2022-11-30.

[17] "ERL - european robotics league", https://eu-robotics.net/eurobotics/activities/european-robotics-league/, accessed: 2022-11-30.

[18] "MBZIRC", https://www.mbzirc.com/, accessed: 2022-11-30.

[19] "Aerial robotics control and perception challenge", https://www.med-control.org/med2018/?page_id=602, accessed: 2022-11-30.

[20] Ruggiero, F., Lippiello, V., Ollero, A., "Aerial manipulation: A literature review", IEEE Robotics and Automation Letters, Vol. 3, No. 3, 2018, pp. 1957-1964.

[21] Mersha, A. Y., Stramigioli, S., Carloni, R., "Variable impedance control for aerial interaction", in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 3435-3440.

[22] Papachristos, C., Alexis, K., Tzes, A., "Efficient force exertion for aerial robotic manipulation: Exploiting the thrust-vectoring authority of a tri-tiltrotor uav", in 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 4500-4505.

[23] Tsukagoshi, H., Watanabe, M., Hamada, T., Ashlih, D., Iizuka, R., "Aerial manipulator with perching and door-opening capability", in 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 4663-4668.

[24] Darivianakis, G., Alexis, K., Burri, M., Siegwart, R., "Hybrid predictive control for aerial robotic physical interaction towards inspection operations", in 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 53-58.

[25] Mirjan, A., Augugliaro, F., D'Andrea, R., Gramazio, F., Kohler, M., "Building a bridge with flying robots", in Robotic Fabrication in Architecture, Art and Design 2016. Springer International Publishing, 2016, pp. 34–47, available at: https://doi.org/10.1007/978-3-319-26378-6_3

[26] Augugliaro, F., Lupashin, S., Hamer, M., Male, C., Hehn, M., Mueller, M. W., Willmann, J. S., Gramazio, F., Kohler, M., D'Andrea, R., "The flight assembled architecture installation: Cooperative construction with flying machines", IEEE Control Systems Magazine, Vol. 34, No. 4, 2014, pp. 46-64.

[27] Korpela, C., Orsag, M., Danko, T., Oh, P., "Insertion tasks using an aerial manipulator", in 2014 IEEE International Conference on Technologies for Practical Robot Applications (TePRA), 2014, pp. 1-6.

[28] Kim, H., Seo, H., Kim, J., Kim, H. J., "Sampling-based motion planning for aerial pick-and-place", in 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 7402-7408.

[29] Orsag, M., Korpela, C., Bogdan, S., Oh, P., "Valve turning using a dual-arm aerial manipulator", in 2014 International Conference on Unmanned Aircraft Systems (ICUAS), 2014, pp. 836-841.

[30] Trujillo, M. A., Martínez-de Dios, J. R., Martín, C., Viguria, A., Ollero, A., "Novel aerial manipulator for accurate and robust industrial ndt contact inspection: A new tool for the oil and gas inspection industry", Sensors, Vol. 19, No. 6, 2019, available at: https://www.mdpi.com/1424-8220/19/6/1305

[31] Tognon, M., Chávez, H. A. T., Gasparin, E., Sablé, Q., Bicego, D., Mallet, A., Lany, M., Santi, G., Revaz, B., Cortés, J., Franchi, A., "A truly-redundant aerial manipulator system with application to push-and-slide inspection in industrial plants", IEEE Robotics and Automation Letters, Vol. 4, No. 2, 2019, pp. 1846-1851.

[32] Jimenez-Cano, A., Heredia, G., Ollero, A., "Aerial manipulator with a compliant arm for bridge inspection", in 2017 International Conference on Unmanned Aircraft Systems (ICUAS), 2017, pp. 1217-1222.

[33] Ivanovic, A., Markovic, L., Car, M., Duvnjak, I., Orsag, M., "Towards autonomous bridge inspection: Sensor mounting using aerial manipulators", Applied Sciences, Vol. 11, No. 18, Sep. 2021, pp. 8279, available at: https://doi.org/10.3390/app11188279

[34] Kocer, B. B., Orr, L., Stephens, B., Kaya, Y. F., Buzykina, T., Khan, A., Kovac, M., "An intelligent aerial manipulator for wind turbine inspection and repair", in 2022 UKACC 13th International Conference on Control (CONTROL), 2022, pp. 226-227.

[35] Car, M., Ivanovic, A., Orsag, M., Bogdan, S., "Impedance based force control for aerial robot peg-in-hole insertion tasks", in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 6734-6739.

[36] Zagaris, A., Kaper, H. G., Kaper, T. J., "Fast and slow dynamics for the computational singular perturbation method", available at: https://arxiv.org/abs/math/0401206 2004.

[37] Ryll, M., Bicego, D., Franchi, A., "Modeling and control of fast-hex: A fully-actuated by synchronized-tilting hexarotor", in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 1689-1694.

[38] Prado, I. A. A., de Freitas Virgílio Pereira, M., de Castro, D. F., dos Santos, D. A., Balthazar, J. M., "Experimental evaluation of hjb optimal controllers for the attitude dynamics of a multirotor aerial vehicle", ISA Transactions, Vol. 77, 2018, pp. 188-200, available at: https://www.sciencedirect.com/science/article/pii/S0019057818301496

[39] Seah, C. H., Inyang, I. J., Whidborne, J. F., "Bilinear modelling and attitude control of a quadrotor", IFAC-PapersOnLine, Vol. 50, No. 2, 2017, pp. 193-198, control Conference Africa CCA 2017, available at: https://www.sciencedirect.com/science/article/pii/S2405896317335711

[40] Derafa, L., Benallegue, A., Fridman, L., "Super twisting control algorithm for the attitude tracking of a four rotors uav", Journal of the Franklin Institute, Vol. 349, No. 2, 2012, pp. 685-699, advances in Guidance and Control of Aerospace Vehicles using Sliding Mode Control and Observation Techniques, available at: https://www.sciencedirect.com/science/article/pii/S0016003211002821

[41] Chen, F., Wu, Q., Jiang, B., Tao, G., "A reconfiguration scheme for quadrotor helicopter via simple adaptive control and quantum logic", IEEE Transactions on Industrial Electronics, Vol. 62, No. 7, 2015, pp. 4328-4335.

[42] Marzat, J., Bertrand, S., Eudes, A., Sanfourche, M., Moras, J., "Reactive mpc for autonomous mav navigation in indoor cluttered environments: Flight experiments", IFAC-PapersOnLine, Vol. 50, No. 1, 2017, pp. 15 996-16 002, 20th IFAC World Congress, available at: https://www.sciencedirect.com/science/article/pii/S2405896317325375

[43] Kamel, M., Stastny, T., Alexis, K., Siegwart, R., Model Predictive Control for Trajectory Tracking of Unmanned Aerial Vehicles Using Robot Operating System. Cham: Springer International Publishing, 2017, pp. 3–39, available at: https://doi.org/10.1007/978-3-319-54927-9_1

[44] Rosaldo-Serrano, M., Aranda-Bricaire, E., "Trajectory tracking for a commercial quadrotor via time-varying backstepping", IFAC-PapersOnLine, Vol. 51, No. 13, 2018, pp. 532-536, 2nd IFAC Conference on Modelling, Identification and Control of Nonlinear Systems MICNON 2018, available at: https://www.sciencedirect.com/science/article/pii/S2405896318310899

[45] Liu, Z., Zhang, Y., Yuan, C., Ciarletta, L., Theilliol, D., "Collision avoidance and path following control of unmanned aerial vehicle in hazardous environment", Journal of Intelligent & Robotic Systems, Vol. 95, No. 1, Sep. 2018, pp. 193–210, available at: https://doi.org/10.1007/s10846-018-0929-y

[46] Aoki, Y., Asano, Y., Honda, A., Motooka, N., Ohtsuka, T., "Nonlinear model predictive control of position and attitude in a hexacopter with three failed rotors∗∗this work was partly supported by jsps kakenhi grant number 15h02257", IFAC-PapersOnLine, Vol. 51, No. 20, 2018, pp. 228-233, 6th IFAC Conference on Nonlinear Model Predictive Control NMPC 2018, available at: https://www.sciencedirect.com/science/article/pii/S2405896318326727

[47] Goodarzi, F., Lee, D., Lee, T., "Geometric nonlinear pid control of a quadrotor uav on se(3)", in 2013 European Control Conference (ECC), 2013, pp. 3845-3850.

[48] Nascimento, T. P., Saska, M., "Position and attitude control of multi-rotor aerial vehicles: A survey", Annual Reviews in Control, Vol. 48, 2019, pp. 129–146, available at: https://doi.org/10.1016/j.arcontrol.2019.08.004

[49] Orsag, M., Korpela, C., Bogdan, S., Oh, P., "Dexterous aerial robots—mobile manipulation using unmanned aerial systems", IEEE Transactions on Robotics, Vol. 33, No. 6, 2017, pp. 1453-1466.

[50] Palunko, I., Cruz, P., Fierro, R., "Agile load transportation : Safe and efficient load manipulation with aerial robots", IEEE Robotics & Automation Magazine, Vol. 19, No. 3, 2012, pp. 69-79.

[51] Ivanovic, A., Polic, M., Salah, O., Orsag, M., Bogdan, S., "Compliant net for auv retrieval using a uav", IFAC-PapersOnLine, Vol. 51, No. 29, 2018, pp. 431-437, 11th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2018, available at: https://www.sciencedirect.com/science/article/pii/S2405896318321384

[52] Heredia, G., Jimenez-Cano, A., Sanchez, I., Llorente, D., Vega, V., Braga, J., Acosta, J., Ollero, A., "Control of a multirotor outdoor aerial manipulator", in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 3417-3422.

[53] Kim, S., Choi, S., Kim, H. J., "Aerial manipulation using a quadrotor with a two dof robotic arm", in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 4990-4995.

[54] Mersha, A. Y., Stramigioli, S., Carloni, R., "Exploiting the dynamics of a robotic manipulator for control of uavs", in 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 1741-1746.

[55] Haus, T., Ivanovic, A., Car, M., Orsag, M., Bogdan, S., "Mid-ranging control concept for a multirotor uav with moving masses", in 2018 26th Mediterranean Conference on Control and Automation (MED), 2018, pp. 339-344.

[56] Ivanovic, A., Car, M., Orsag, M., Bogdan, S., "Centroid vectoring control using aerial manipulator: Experimental results", in 2019 International Conference on Unmanned Aircraft Systems (ICUAS), 2019, pp. 372-377.

[57] Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G., Robotics: Modelling and Control. Springer London, 2009, available at: https://doi.org/10.1007/978-1-84628-642-1

[58] Ruggiero, F., Trujillo, M., Cano, R., Ascorbe, H., Viguria, A., Peréz, C., Lippiello, V., Ollero, A., Siciliano, B., "A multilayer control for multirotor uavs equipped with a servo robot arm", in 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 4014-4020.

[59] Jimenez-Cano, A., Martin, J., Heredia, G., Ollero, A., Cano, R., "Control of an aerial robot with multi-link arm for assembly tasks", in 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 4916-4921.

[60] Huber, F., Kondak, K., Krieger, K., Sommer, D., Schwarzbach, M., Laiacker, M., Kossyk, I., Parusel, S., Haddadin, S., Albu-Schäffer, A., "First analysis and experiments in aerial manipulation using fully actuated redundant robot arm", in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 3452-3457.

[61] Arbanas, B., Ivanovic, A., Car, M., Haus, T., Orsag, M., Petrovic, T., Bogdan, S., "Aerial-ground robotic system for autonomous delivery tasks", in 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 5463-5468.

[62] Arbanas, B., Ivanovic, A., Car, M., Orsag, M., Petrovic, T., Bogdan, S., "Decentralized planning and control for UAV–UGV cooperative teams", Autonomous Robots, Vol. 42, No. 8, Feb. 2018, pp. 1601–1618, available at: https://doi.org/10.1007/s10514-018-9712-y

[63] Orsag, M., Haus, T., Tolić, D., Ivanovic, A., Car, M., Palunko, I., Bogdan, S., "Human-in-the-loop control of multi-agent aerial systems", in 2016 European Control Conference (ECC), 2016, pp. 2139-2145.

[64] Fumagalli, M., Naldi, R., Macchelli, A., Carloni, R., Stramigioli, S., Marconi, L., "Modeling and control of a flying robot for contact inspection", in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 3532-3537.

[65] Alexis, K., Darivianakis, G., Burri, M., Siegwart, R., "Aerial robotic contact-based inspection: Planning and control", Auton. Robots, Vol. 40, No. 4, April 2016, pp. 631–655, available at: http://dx.doi.org/10.1007/s10514-015-9485-5

[66] Cataldi, E., Muscio, G., Trujillo, M. A., Rodriguez, Y., Pierri, F., Antonelli, G., Caccavale, F., Viguria, A., Chiaverini, S., Ollero, A., "Impedance control of an aerial-manipulator: Preliminary results", in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 3848-3853.

[67] Nava, G., Sablé, Q., Tognon, M., Pucci, D., Franchi, A., "Direct force feedback control and online multi-task optimization for aerial manipulators", IEEE Robotics and Automation Letters, Vol. 5, No. 2, 2020, pp. 331-338.

[68] Meng, X., He, Y., Li, Q., Gu, F., Yang, L., Yan, T., Han, J., "Contact force control of an aerial manipulator in pressing an emergency switch process", in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 2107-2113.

[69] Hamaza, S., Georgilas, I., Fernandez, M., Sanchez, P., Richardson, T., Heredia, G., Ollero, A., "Sensor installation and retrieval operations using an unmanned aerial manipulator", IEEE Robotics and Automation Letters, Vol. 4, No. 3, 2019, pp. 2793-2800.

[70] Car, M., Ivanovic, A., Orsag, M., Bogdan, S., "Position-based adaptive impedance control for a uav", in 2018 International Conference on Unmanned Aircraft Systems (ICUAS), 2018, pp. 957-963.

[71] Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., Burgard, W., "OctoMap: An efficient probabilistic 3D mapping framework based on octrees", Autonomous Robots, 2013, software available at https://octomap.github.io, available at: https://octomap.github.io

[72] Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., Nieto, J., "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning", in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 1366-1373.

[73] Aggarwal, S., Kumar, N., "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges", Computer Communications, Vol. 149, Jan. 2020, pp. 270–299, available at: https://doi.org/10.1016/j.comcom.2019.10.014

[74] Samaniego, F., Sanchis, J., García-Nieto, S., Simarro, R., "Uav motion planning and obstacle avoidance based on adaptive 3d cell decomposition: Continuous space vs discrete space", in 2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM), 2017, pp. 1-6.

[75] Bai, W., Wu, X., Xie, Y., Wang, Y., Zhao, H., Chen, K., Li, Y., Hao, Y., "A cooperative route planning method for multi-uavs based-on the fusion of artificial potential field and b-spline interpolation", in 2018 37th Chinese Control Conference (CCC), 2018, pp. 6733-6738.

[76] Kavraki, L., Svestka, P., Latombe, J.-C., Overmars, M., "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", IEEE Transactions on Robotics and Automation, Vol. 12, No. 4, 1996, pp. 566-580.

[77] Chen, T., Zhang, G., Hu, X., Xiao, J., "Unmanned aerial vehicle route planning method based on a star algorithm", in 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2018, pp. 1510-1514.

[78] LaValle, S. M., Kuffner Jr, J. J., "Randomized kinodynamic planning", The international journal of robotics research, Vol. 20, No. 5, 2001, pp. 378–400.

[79] Kuffner, J., LaValle, S., "Rrt-connect: An efficient approach to single-query path planning", in Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), Vol. 2, 2000, pp. 995-1001 vol.2.

[80] Gammell, J. D., Srinivasa, S. S., Barfoot, T. D., "Informed rrt: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic", in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 2997-3004.

[81] Mellinger, D., Kumar, V., "Minimum snap trajectory generation and control for quadrotors", in 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 2520-2525.

[82] Mellinger, D., Michael, N., Kumar, V., "Trajectory generation and control for precise aggressive maneuvers with quadrotors", The International Journal of Robotics Research, Vol. 31, No. 5, 2012, pp. 664-674.

[83] Richter, C., Bry, A., Roy, N., "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments", in Springer Tracts in Advanced Robotics. Springer International Publishing, 2016, pp. 649–666.

[84] de Almeida, M. M., Akella, M., "New numerically stable solutions for minimum-snap quadcopter aggressive maneuvers", in 2017 American Control Conference (ACC), 2017, pp. 1322-1327.

[85] Müller, M., Lupashin, S., D'Andrea, R., "Quadrocopter ball juggling", in 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011, pp. 5113-5120.

[86] Mueller, M. W., Hehn, M., D'Andrea, R., "A computationally efficient motion primitive for quadrocopter trajectory generation", IEEE Transactions on Robotics, Vol. 31, No. 6, 2015, pp. 1294-1310.

[87] Boeuf, A., Cortés, J., Alami, R., Siméon, T., "Planning agile motions for quadrotors in constrained environments", in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 218-223.

[88] Liu, S., Mohta, K., Atanasov, N., Kumar, V., "Search-based motion planning for aggressive flight in se(3)", IEEE Robotics and Automation Letters, Vol. 3, No. 3, 2018, pp. 2439-2446.

[89] Augugliaro, F., Schoellig, A. P., D'Andrea, R., "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach", in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 1917-1922.

[90] Landry, B., Deits, R., Florence, P. R., Tedrake, R., "Aggressive quadrotor flight through cluttered environments using mixed integer programming", in 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 1469-1475.

[91] Oleynikova, H., Burri, M., Taylor, Z., Nieto, J., Siegwart, R., Galceran, E., "Continuous-time trajectory optimization for online uav replanning", in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 5332-5339.

[92] Bobrow, J., Dubowsky, S., Gibson, J., "Time-optimal control of robotic manipulators along specified paths", The International Journal of Robotics Research, Vol. 4, No. 3, Sep. 1985, pp. 3–17, available at: https://doi.org/10.1177/027836498500400301

[93] Shin, K., McKay, N., "Minimum-time control of robotic manipulators with geometric path constraints", IEEE Transactions on Automatic Control, Vol. 30, No. 6, 1985, pp. 531-541.

[94] Pham, Q., "A general, fast, and robust implementation of the time-optimal path parameterization algorithm", IEEE Transactions on Robotics, Vol. 30, No. 6, Dec 2014, pp. 1533-1540.

[95] Pham, H., Pham, Q., "A new approach to time-optimal path parameterization based on reachability analysis", IEEE Transactions on Robotics, Vol. 34, No. 3, June 2018, pp. 645-659.

[96] Goričanec, J., Ivanović, A., Marković, L., Pavlak, I., Bogdan, S., "Civil infrastructure data acquisition in urban environments based on multi-uav mission", in 2021 International Conference on Unmanned Aircraft Systems (ICUAS), 2021, pp. 542-551.

[97] Batinovic, A., Petrovic, T., Ivanovic, A., Petric, F., Bogdan, S., "A multi-resolution frontier-based planner for autonomous 3d exploration", IEEE Robotics and Automation Letters, Vol. 6, No. 3, 2021, pp. 4528-4535.

[98] Batinovic, A., Ivanovic, A., Petrovic, T., Bogdan, S., "A shadowcasting-based next-best-view planner for autonomous 3d exploration", IEEE Robotics and Automation Letters, Vol. 7, No. 2, 2022, pp. 2969-2976.

[99] Stilman, M., "Global manipulation planning in robot joint space with task constraints", IEEE Transactions on Robotics, Vol. 26, No. 3, June 2010, pp. 576-584.

[100] Cefalo, M., Oriolo, G., "Task-constrained motion planning for underactuated robots", in 2015 IEEE International Conference on Robotics and Automation (ICRA), May 2015, pp. 2965-2970.

[101] Yu, Y., Lippiello, V., "6d pose task trajectory tracking for a class of 3d aerial manipulator from differential flatness", IEEE Access, Vol. 7, 2019, pp. 52 257-52 265.

[102] Ragel, R., Maza, I., Caballero, F., Ollero, A., "Comparison of motion planning techniques for a multi-rotor uas equipped with a multi-joint manipulator arm", in 2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS), 2015, pp. 133-141.

[103] Şucan, I. A., Moll, M., Kavraki, L. E., "The Open Motion Planning Library", IEEE Robotics & Automation Magazine, Vol. 19, No. 4, December 2012, pp. 72–82, https://ompl.kavrakilab.org.

[104] Coleman, D., Sucan, I. A., Chitta, S., Correll, N., "Reducing the barrier to entry of complex robotic software: a moveit! case study", CoRR, Vol. abs/1404.3785, 2014, available at: http://arxiv.org/abs/1404.3785

[105] Welde, J., Paulos, J., Kumar, V., "Dynamically feasible task space planning for underactuated aerial manipulators", IEEE Robotics and Automation Letters, Vol. 6, No. 2, 2021, pp. 3232-3239.

[106] Caballero, A., Suarez, A., Real, F., Vega, V. M., Bejar, M., Rodriguez-Castaño, A., Ollero, A., "First experimental results on motion planning for transportation in aerial long-reach manipulators with two arms", in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 8471-8477.

[107] Lee, H., Kim, H., Kim, H. J., "Planning and control for collision-free cooperative aerial transportation", IEEE Transactions on Automation Science and Engineering, Vol. 15, No. 1, 2018, pp. 189-201.

[108] Kim, H., Seo, H., Son, C. Y., Lee, H., Kim, S., Kim, H. J., "Cooperation in the air: A learning-based approach for the efficient motion planning of aerial manipulators", IEEE Robotics & Automation Magazine, Vol. 25, No. 4, 2018, pp. 76-85.

[109] Lee, H., Son, C. Y., Kim, H. J., "Collision-free path planning for cooperative aerial manipulators under velocity and curvature constraints", IEEE Access, Vol. 7, 2019, pp. 171 153-171 162.

[110] Seo, H., Kim, S., Kim, H. J., "Locally optimal trajectory planning for aerial manipulation in constrained environments", in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 1719-1724.

[111] Tognon, M., Cataldi, E., Chavez, H. A. T., Antonelli, G., Cortés, J., Franchi, A., "Control-aware motion planning for task-constrained aerial manipulation", IEEE Robotics and Automation Letters, Vol. 3, No. 3, July 2018, pp. 2478-2484.

[112] Ivanovic, A., Car, M., Orsag, M., Bogdan, S., "Exploiting null space in aerial manipulation through model-in-the-loop motion planning", in 2020 International Conference on Unmanned Aircraft Systems (ICUAS), 2020, pp. 686-693.

[113] Ward, M., Costello, M., "Adaptive glide slope control for autonomous airdrop systems", in 2012 American Control Conference (ACC), 2012, pp. 2557-2562.

[114] Cacan, M. R., Scheuermann, E., Ward, M., Costello, M., Slegers, N., "Autonomous airdrop systems employing ground wind measurements for improved landing accuracy", IEEE/ASME Transactions on Mechatronics, Vol. 20, No. 6, 2015, pp. 3060-3070.

[115] VanderMey, J. T., Doman, D. B., Gerlach, A. R., "Release point determination and dispersion reduction for ballistic airdrops", Journal of Guidance, Control, and Dynamics, Vol. 38, No. 11, Nov. 2015, pp. 2227–2235.

[116] Joshua, M., Eaton, A. N., "Point of impact: Delivering mission essential supplies to the warfighter through the joint precision airdrop system", in 2013 IEEE International Systems Conference (SysCon), 2013, pp. 783-790.

[117] Gerlach, A. R., Manyam, S. G., Doman, D. B., "Precision airdrop transition altitude optimization via the one-in-a-set traveling salesman problem", in 2016 American Control Conference (ACC), 2016, pp. 3498-3502.

[118] Li, B., Han, J., Xiao, J., "Real-time trajectory planning for autonomous parafoil in obstacle-rich environment", in 2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), 2018, pp. 457-462.

[119] Rakesh, R., Harikumar, R., "Autonomous airdrop system using small-scale parafoil", in 2019 International Conference on Computer Communication and Informatics (ICCCI), 2019, pp. 1-6.

[120] Mathisen, S. H., Grindheim, V., Johansen, T. A., "Approach methods for autonomous precision aerial drop from a small unmanned aerial vehicle", in 2017 20th IFAC World Congress, Vol. 50, 2017, pp. 3566-3573.

[121] Mathisen, S. G., Leira, F. S., Helgesen, H. H., Gryte, K., Johansen, T. A., "Autonomous ballistic airdrop of objects from a small fixed-wing unmanned aerial vehicle", Autonomous Robots, Vol. 44, No. 5, Jan. 2020, pp. 859–875.

[122] Mardiyanto, R., Pujiantara, M., Suryoatmojo, H., Dikairono, R., Irfansyah, A. N., "Development of unmanned aerial vehicle (uav) for dropping object accurately based on global positioning system", in 2019 International Seminar on Intelligent Technology and Its Applications (ISITIA), 2019, pp. 86-90.

[123] Denavit, J., Hartenberg, R. S., "A kinematic notation for lower-pair mechanisms based on matrices", Journal of Applied Mechanics, Vol. 22, No. 2, Jun. 1955, pp. 215–221, available at: https://doi.org/10.1115/1.4011045

[124] Orsag, M., Korpela, C., Oh, P., Bogdan, S., Aerial Manipulation. Springer International Publishing, 2018, available at: https://doi.org/10.1007/978-3-319-61022-1

[125] Kovačić, Z., Bogdan, S., Krajči, V., Fundamentals of robotics. Graphis, 2002.

[126] Diankov, R., Kuffner, J., "Openrave: A planning architecture for autonomous robotics", Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34, Vol. 79, 2008.

[127] Brescianini, D., D'Andrea, R., "An omni-directional multirotor vehicle", Mechatronics, Vol. 55, Nov. 2018, pp. 76–93, available at: https://doi.org/10.1016/j.mechatronics.2018.08.005

[128] Rigatos, G. G., Nonlinear Control and Filtering Using Differential Flatness Approaches. Springer International Publishing, 2015, available at: https://doi.org/10.1007/978-3-319-16420-5

[129] Yüksel, B., Buondonno, G., Franchi, A., "Differential flatness and control of protocentric aerial manipulators with any number of arms and mixed rigid-/elastic-joints", in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 561-566.

[130] Kunz, T., Stilman, M., "Time-optimal trajectory generation for path following with bounded acceleration and velocity", Proc. Robot.: Sci. Syst. Conf., Vol. 8, 2012, pp. 09-13.

[131] Koenig, N., Howard, A., "Design and use paradigms for gazebo, an open-source multi-robot simulator", in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Vol. 3, 2004, pp. 2149-2154 vol.3.

[132] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A. Y. *et al.*, "Ros: an open-source robot operating system", in ICRA workshop on open source software, Vol. 3, No. 3.2. Kobe, Japan, 2009, pp. 5.

[133] Furrer, F., Burri, M., Achtelik, M., Siegwart, R., Robot Operating System (ROS): The Complete Reference (Volume 1). Cham: Springer International Publishing, 2016, ch. RotorS—A Modular Gazebo MAV Simulator Framework, pp. 595–625, available at: http://dx.doi.org/10.1007/978-3-319-26054-9_23

[134] Meyer, J., Sendobry, A., Kohlbrecher, S., Klingauf, U., von Stryk, O., "Comprehensive simulation of quadrotor uavs using ros and gazebo", in Simulation, Modeling, and Programming for Autonomous Robots, Noda, I., Ando, N., Brugali, D., Kuffner, J. J., (ur.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 400–411.

[135] Chitta, S., Marder-Eppstein, E., Meeussen, W., Pradeep, V., Rodríguez Tsouroukdissian, A., Bohren, J., Coleman, D., Magyar, B., Raiola, G., Lüdtke, M., Fernández Perdomo, E., "ros_control: A generic and simple control framework for ros", The Journal of Open Source Software, 2017, available at: http://www.theoj.org/joss-papers/joss.00456/10.21105.joss.00456.pdf

[136] Lynen, S., Achtelik, M. W., Weiss, S., Chli, M., Siegwart, R., "A robust and modular multi-sensor fusion approach applied to mav navigation", in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 3923-3929.

[137] Marković, L., Car, M., Orsag, M., Bogdan, S., "Adaptive stiffness estimation impedance control for achieving sustained contact in aerial manipulation", in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 117-123.

[138] Seraji, H., Colbaugh, R., "Force tracking in impedance control", in [1993] Proceedings IEEE International Conference on Robotics and Automation, 1993, pp. 499-506 vol.2.

[139] Zargarbashi, S., Khan, W., Angeles, J., "The jacobian condition number as a dexterity index in 6r machining robots", Robotics and Computer-Integrated Manufacturing, Vol. 28, No. 6, Dec. 2012, pp. 694–699, available at: https://doi.org/10.1016/j.rcim.2012.04.004

[140] Mikhailov, M., Freire, A. S., "The drag coefficient of a sphere: An approximation using shanks transform", Powder Technology, Vol. 237, Mar. 2013, pp. 432–435.

[141] Taddese, A. Z., Slawinski, P. R., Obstein, K. L., Valdastri, P., "Closed loop control of a tethered magnetic capsule endoscope", in Robotics: Science and Systems XII. Robotics: Science and Systems Foundation, 2016.

[142] "Storm ros gazebo magnet plugin", https://github.com/larics/storm_gazebo_ros_magnet/tree/melodic_electromagnet_dev, accessed: 10-01-2023.

[143] Milijas, R., Markovic, L., Ivanovic, A., Petric, F., Bogdan, S., "A comparison of lidar-based slam systems for control of unmanned aerial vehicles", in 2021 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, 2021, pp. 1148–1154.

[144] Oršulić, J., Milijas, R., Batinovic, A., Markovic, L., Ivanovic, A., Bogdan, S., "Flying with cartographer: Adapting the cartographer 3d graph slam stack for uav navigation", in 2021 Aerial Robotic Systems Physically Interacting with the Environment (AIRPHARO), 2021, pp. 1-7.

[145] Faessler, M., Franchi, A., Scaramuzza, D., "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories", IEEE Robotics and Automation Letters, Vol. 3, No. 2, 2018, pp. 620-626.

[146] Tal, E., Karaman, S., "Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness", IEEE Transactions on Control Systems Technology, Vol. 29, No. 3, 2021, pp. 1203-1218.

[147] Martinović, D., Vuletić, J., Stuhne, D., Orsag, M., Kovačić, Z., "Object localization by construction of an asymmetric isobody of the magnetic gradient tensor contraction using two identical permanent magnets", IEEE Transactions on Magnetics, 2023, pp. 1-1.

Biography

Antun Ivanović is a Ph.D. student and researcher at the Laboratory for Robotics and Intelligent Control Systems within the University of Zagreb Faculty of Electrical Engineering and Computing (UNIZG-FER). He graduated and received his master's degree in 2015 at this same institution and after that joined LARICS later that year. During his undergraduate studies, he received a Rector's award for the work entitled "Augmented human machine interface for aerial manipulators".

His research interests are robotics, unmanned aerial vehicles, aerial manipulation and motion planning. As a Ph.D. student, he participated in EuRoC competition (European Robotics Challenge), MBZIRC2020 (Mohamed Bin Zayed International Robotics Challenge), MORUS project (Unmanned system for maritime security and environmental monitoring), Encore (ENergy aware BIM Cloud Platform in a COst-effective Building REnovation Context), and AeroWind (Autonomous UAV inspection of wind turbine blades). He is currently working on several EU or nationally funded projects including Specularia, VirtualUAV, and AeroStream. In 2018, he was a visiting researcher at United States Military Academy West Point, USA, where he collaborated work related to aerial-ground cooperative manipulation. So far, he has authored and co-authored over 20 research papers, and two book chapter.

List of publications

Book chapters

1. Tolić, D., Palunko, I., Ivanović, A., Car, M., and Bogdan, S. (2016). Decentralized Cooperative Control in Degraded Communication Environments. Control of Complex Systems: Theory and Applications, 373.
2. Arbanas, B., Petric, F., Batinović, A., Polić, M., Vatavuk, I., Marković, L., Car, M., Hrabar, I., Ivanović, A., and Bogdan, S. (2021). From ERL to MBZIRC: Development of An Aerial-Ground Robotic Team for Search and Rescue. In Automation and Control

[Working Title]. IntechOpen.

Journal papers

1. Arbanas, B., Ivanović, A., Car, M., Orsag, M., Petrović, T., and Bogdan, S. (2018). Decentralized planning and control for UAV-UGV cooperative teams. Autonomous Robots, 42(8), 1601–1618.
2. Car, M., Marković, L., Ivanović, A., Orsag, M., and Bogdan, S. (2020). Autonomous Wind-Turbine Blade Inspection Using LiDAR-Equipped Unmanned Aerial Vehicle. IEEE Access, 8, 131380–131387.
3. Batinović, A., Petrović, T., Ivanović, A., Petric, F., and Bogdan, S. (2021). A Multi-Resolution Frontier-Based Planner for Autonomous 3D Exploration. IEEE Robotics and Automation Letters, 6(3), 4528–4535.
4. Ivanović, A., Marković, L., Car, M., Duvnjak, I., and Orsag, M. (2021). Towards Autonomous Bridge Inspection: Sensor Mounting Using Aerial Manipulators. Applied Sciences, 11(18).
5. Batinović, A., Ivanović, A., Petrović, T., and Bogdan, S. (2022). A Shadowcasting-Based Next-Best-View Planner for Autonomous 3D Exploration. IEEE Robotics and Automation Letters, 7(2), 2969–2976.
6. Ivanović, A., and Orsag, M. (2022). Parabolic Airdrop Trajectory Planning for Multirotor Unmanned Aerial Vehicles. IEEE Access, 1–1.

Conference papers

1. Tolić, D., Palunko, I., Ivanović, A., Car, M., and Bogdan, S. (2015). Multi-agent control in degraded communication environments.2015 European Control Conference (ECC), 404–409.
2. Arbanas, B., Ivanović, A., Car, M., Haus, T., Orsag, M., Petrović, T., and Bogdan, S. (2016). Aerial-ground robotic system for autonomous delivery tasks. 2016 IEEE International Conference On Robotics and Automation, 5463–5468.
3. Orsag, M., Haus, T., Tolić, D., Ivanović, A., Car, M., Palunko, I., and Bogdan, S. (2016). Human-in-the-loop control of multi-agent aerial systems. 2016 European Control Conference (ECC), 2139–2145.
4. Car, M., Ivanovic, A., Orsag, M., and Bogdan, S. (2018). Impedance based force control for aerial robot peg-in-hole insertion tasks. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 6734–6739.
5. Car, M., Ivanović, A., Orsag, M., and Bogdan, S. (2018). Position-based adaptive impedance control for a UAV. 2018 International Conference on Unmanned Aircraft Systems (ICUAS),

957–963.

6. Haus, T., Ivanović, A., Car, M., Orsag, M., and Bogdan, S. (2018). Mid-Ranging Control Concept for a Multirotor UAV with Moving Masses. 2018 26th Mediterranean Conference on Control and Automation (MED), 339–344.
7. Ivanović, A., Polić, M., Salah, O., Orsag, M., and Bogdan, S. (2018). Compliant net for AUV retrieval using a UAV. IFAC-PapersOnLine, 51(29), 431–437.
8. Ivanović, A., Car, M., Orsag, M., and Bogdan, S. (2019). Centroid vectoring control using aerial manipulator: Experimental results. 2019 International Conference on Unmanned Aircraft Systems (ICUAS), 372–377.
9. Marković, L., Ivanović, A., Car, M., Orsag, M., and Bogdan, S. (2019). Geometric Tracking Control of Aerial Robots Based on Centroid Vectoring. 2019 18th European Control Conference (ECC), 2701–2706.
10. Ivanović, A., Car, M., Orsag, M., and Bogdan, S. (2020). Exploiting Null Space in Aerial Manipulation through Model-In-The-Loop Motion Planning. 2020 International Conference on Unmanned Aircraft Systems (ICUAS), 686–693.
11. Goričanec, J., Ivanović, A., Marković, L., Pavlak, I., and Bogdan, S. (2021). Civil infrastructure data acquisition in urban environments based on multi-UAV mission. 2021 International Conference on Unmanned Aircraft Systems (ICUAS), 542–551.
12. Milijaš, R., Marković, L., Ivanović, A., Petric, F., and Bogdan, S. (2021). A Comparison of LiDAR-based SLAM Systems for Control of Unmanned Aerial Vehicles. 2021 International Conference on Unmanned Aircraft Systems (ICUAS), 1148–1154.
13. Oršulić, J., Milijaš, R., Batinović, A., Marković, L., Ivanović, A., and Bogdan, S. (2021). Flying with Cartographer: Adapting the Cartographer 3D Graph SLAM Stack for UAV Navigation. 2021 Aerial Robotic Systems Physically Interacting with the Environment (AIRPHARO), 1–7.
14. Polić, M., Ivanović, A., Marić, B., Arbanas, B., Tabak, J., and Orsag, M. (2021). Structured Ecological Cultivation with Autonomous Robots in Indoor Agriculture. 2021 16th International Conference on Telecommunications (ConTEL), 189–195.
15. Ivanovic, A., Polic, M., Tabak, J., and Orsag, M. (2022). Render-in-the-loop Aerial Robotics Simulator: Case Study on Yield Estimation in Indoor Agriculture. 2022 International Conference on Unmanned Aircraft Systems (ICUAS), 787–793.

Životopis

Antun Ivanović je doktorand i istraživač u Laboratoriju za Robotiku i Inteligentne Sustave Upravljanja u sklopu Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu (UNIZG-FER). Diplomirao je 2015. godine na istom fakultetu i nakon toga se pridružio LARICS-u. Tijekom svog studija, dobio je rektorovu nagradu za rad pod naslovom " Prošireno korisničko sučelje za upravljanje robotskim manipulatorom u letu".

Njegove interesne sfere su roboti, bespilotne letjelice, zračna manipulacija i planiranje gibanja. Kao doktorand, sudjelovao je u natjecanju EuRoC (European Robotics Challenge), MBZIRC2020 (Mohamed Bin Zayed International Robotics Challenge), projektu MORUS (Unmanned system for maritime security and environmental monitoring), Encore (ENergy aware BIM Cloud Platform in a COst-effective Building REnovation Context) i AeroWind (Autonomous UAV inspection of wind turbine blades). Trenutno radi na nekoliko EU ili nacionalno financiranih projekata, uključujući Specularia, VirtualUAV i AeroStream. Godine 2018. boravio je kao istraživač u Sjedinjenim Američkim Država u akademiji West Point, gdje je surađivao na radu vezanom za kooperativnu manipulaciju između bespilotne letjelice i zemaljskog vozila. Dosad je autor ili koautor više od 20 znanstvenih radova i dva poglavlja knjige.