Controller Design and Implementation of a New Quadrotor Manipulation System

By

Ahmed Mohammed Elsayed Khalifa

A Thesis Submitted to the
Graduate School of Innovative Design Engineering,

Egypt-Japan University of Science and Technology (E-JUST)

In Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in

Mechatronics and Robotics Engineering
September 2016
Controller Design and Implementation of a New Quadrotor Manipulation System

By
Ahmed Mohammed Elsayed Khalifa

For The Degree
of
Doctor of Philosophy
in
Mechatronics and Robotics Engineering

Supervision Committee

Name, Title                      Affiliation                     Signature, Date

Prof. Mohamed A. Fanni          Mechatronics and Robotics        ............
                                 Engineering, E-JUST

Prof. Toru Namerikawa           Systems and Control                ............
                                 Engineering, Keio University
| Name, Title                  | Affiliation                          | Approved, Date |
|-----------------------------|--------------------------------------|----------------|
| Prof. Sohair F. Rezeka      | Mechanical Engineering,              | ................ |
|                             | Alexandria University                |                |
| Prof. Khaled A. El-Metwally | Automatic Control                    | ................ |
|                             | Engineering, Cairo University        |                |
| Prof. Mohamed A. Fanni      | Mechatronics and Robotics            | ................ |
|                             | Engineering, E-JUST                  |                |
Controller Design and Implementation of a New Quadrotor Manipulation System

by Ahmed Khalifa

Copyright © 2016 Ahmed Mohammed Elsayed Khalifa. All rights reserved. This document was created with the document preparation system \LaTeX.
DECLARATION AND CERTIFICATE OF
ORIGINALITY

I certify that in the preparation of this thesis, I have observed the provisions of E-JUST Code of Ethics dated 8 September 2013. Further; I certify that this work is free of plagiarism and all materials appearing in this thesis have been properly quoted and attributed. I certify that all copyrighted material incorporated into this thesis is in compliance with the international copyright law and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and save harmless E-JUST from any and all claims that may be asserted or that may arise from any copyright violation. I hereby certify that the research work in this thesis is my original work and it does not include any copied parts without the appropriate citation.

Alexandria, Egypt,
April 19, 2019
Ahmed Mohammed Elsayed Khalifa,
The research on aerial manipulation has been increased rapidly in recent years due to its unique features that qualify it to be utilized in vital applications and to support human in daily life activities. In the previous works, a manipulator or a gripper is attached to the bottom of a quadrotor to facilitate the interaction with the environment. However, the previous introduced systems suffer from either limited end-effector Degrees Of Freedom (DOF) or small payload capacity. The systems that use a gripper suffer from the limited allowable DOF of the end-effector. The other systems have a manipulator with either two DOF but in certain topology that disables the end-effector to track arbitrary 6-DOF trajectory, or more than two DOF that decreases greatly the possible payload carried by the system. In this dissertation, a quadrotor with a 2-DOF manipulator is investigated that has a unique topology to enable the end-effector to track 6-DOF trajectory with minimum possible number of actuators/links and hence, maximize the payload and/or mission time. The proposed system is designed, modeled, and constructed. An identification process is carried out to find the system parameters. The currently developed quadrotor-based aerial manipulators depend on a high cost 3D motion capture system to estimate the system states which will restrict the functions of the system to be within limited space indoors. Therefore, an experimental setup is proposed with a 6-DOF state measurement and estimation scheme that can be utilized within arbitrary places indoors and it has a potential to work outdoors as well. The system feasibility is validated via numerical and experimental results. The proposed non-redundant system has complexity in its inverse kinematics that are required in order to transform task space trajectory to quadrotor/joint space trajectory. Such inverse kinematics require a solution of complicated algebraic-differential equations. The differential equations represent the higher-order nonholonomic constraints of the system. Therefore, an algorithm is developed to get an approximate solution of these equations. The accuracy of the solution is verified via numerical simulations. Moreover, the success of the inverse kinematics analysis proves the ability of the system to track arbitrary 6-DOF task space trajectories that lay within
the capability of the system constraints. Furthermore, the motion control of this quadrotor manipulation system is quite challenging. The system has strong nonlinearities and fast dynamics. There are complex couplings among its variables. In addition, it is naturally unstable and very susceptible to parameters variations, external disturbances (e.g., wind), noisy measurements, and actuators’ faults. Thus, a linear Disturbance Observer (DOb)-based robust control technique is utilized to address these issues. A modified DOb loop is proposed and designed to use the direct measurements from the onboard sensors in order to reduce the disturbance estimation error. For the optimization of the control signal, to save power consumption and thus to increase the mission time, as well as the consideration of the actuators’ constraints, a Model Predictive Control (MPC) is used in the external loop of the DOb to achieve these objectives. In addition to the accurate positioning of the end-effector, the manipulation tasks require estimating (applying) certain force at the end-effector as well. However, the current developed techniques have limitations because they are model-based methods, and they are based on ignoring some dynamics and/or requiring an indicator of the environment contact. Therefore, a robust sensorless force estimation and impedance control scheme is proposed to overcome these limitations based on both the robust linearization capabilities of the DOb and the Fast Tracking Recursive Least Squares (FTRLS) algorithm. The efficiency of the proposed control systems is validated via numerical results.
This thesis is dedicated to my father 1958 – 2002, and to both Prof. Ahmed Abo-Ismail 1945 – 2015 and Dr. Ahmed Ramadan 1973 – 2015 who never got to read the final draft. You are the determination in every page.
ACKNOWLEDGMENT

First of all, countless thanks to ALLAH the almighty

While my name may be alone on the front cover of this thesis, I am by no means its sole contributor. Rather, there are a number of people behind this piece of work who deserve to be both acknowledged and thanked here: My academic supervisors; Prof. Mohamed Fanni, late Prof. Ahmed Abo-Ismail, late Dr. Ahmed Ramadan, and Prof. Toru Namerikawa.

I would like to express my sincere condolences for the loss of Prof. Ahmed Abo-Ismail and Dr. Ahmed Ramadan. Prof. Ahmed Abo-Ismail is the founder of the Mechatronics and Robotics Engineering Department and he is the former Vice President of Education and Academic Affairs and one of E-JUST founders and pillars. Dr. Ahmed Ramadan made an extraordinary effort to build and establish most of the Labs in the Department of Mechatronics and Robotics Engineering at E-JUST. He was really thoroughly enjoyable, impeccably honest, straightforward, positive in his attitude and really has high morals. We pray for them to rest in peace and for their families to pass these difficult times. We ask ALLAH to meet them in the Paradise.

I am forever indebted to my supervisor, Assoc. Prof. Mohamed Fanni, for his novel ideas, enthusiasm, guidance, and unrelenting support throughout this research. He has routinely gone beyond his duties to fire fight my worries and concerns, and have worked to instill great confidence in both myself and my work. THANK YOU FOR MAKING ME MORE THAN I AM.

I would like to show my sincere appreciation and gratitude to Prof. Abdelfattah Mohamed, the former chairperson of the Department of Mechatronics and Robotics Engineering, for his support, guidance and encouragement.

I would like to convey my heartfelt thanks to my home university, Menofia University, Faculty of Electronic Engineering, and Department of Industrial Electronics and Control Engineering for their support and encouragement.
I would like to show my sincere appreciation and gratitude to Prof. Toru Namerikawa, Keio University, for accepting me in his research group and providing generous support, insightful advice and kind encouragement throughout my time at Keio University to date. The Namerikawa Lab has always been a place where people are ready to offer keen criticism and advice on any and all topics, academic or otherwise. For this I thank Mr. Okawa, Mr. Mori, Mr. Komagine, Mr. Hayashi, and Mr. Shinohara.

Without the open-source software generously released by several individuals around the world, much of the implementation and experimental part of the work in this thesis would have been considerably more difficult. I am therefore thankful to the authors of ROS, the ROS AscTec and Laser drivers, as well as those of Linux and an uncountable number of supporting packages for this wonderful operating system.

Thanks to my beloved family, my parents, my brother, and my sisters, for their self-sacrifice, consistent love, support, understanding and encouragement.

Most of all, I thank my darling wife, whose love and strength have never flagged through the many ups and downs of this journey; and my children, whose smiles upon my return from a too-long day on campus always lift my spirits.

I would like to gratefully acknowledge the financial support of the Ministry of Higher Education, Government of Egypt, for my postgraduate scholarship.

Ahmed Khalifa

September 2016
TABLE OF CONTENTS

DECLARATION AND CERTIFICATE OF ORIGINALITY ix

SUMMARY xi

ACKNOWLEDGMENT xv

TABLE OF CONTENTS xvii

LIST OF TABLES xxi

LIST OF FIGURES xxiii

1 INTRODUCTION 1
  1.1 Research Background . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
    1.1.1 Overview of the Current Developed Aerial Manipulators . . . . . . . 1
    1.1.2 Experimental Setup and Measurements . . . . . . . . . . . . . . . . . . 7
    1.1.3 Control System . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
  1.2 Objectives of Research . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
    1.2.1 Design and Modeling of the Quadrotor Manipulation System . . . . . 13
    1.2.2 Experimental Setup and State Estimation . . . . . . . . . . . . . . . . 13
    1.2.3 Inverse Kinematics . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
    1.2.4 Design of a Control Scheme . . . . . . . . . . . . . . . . . . . . . . . 14
  1.3 Thesis Contributions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
  1.4 Thesis Plan . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

2 SYSTEM DESIGN AND MODELING 17
  2.1 System Description . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
  2.2 System Design . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
    2.2.1 Quadrotor . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
    2.2.2 Manipulator . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
# TABLE OF CONTENTS

2.3 Forward Kinematics .................................................. 20
2.4 Dynamic Model ..................................................... 28

3 SYSTEM CONSTRUCTION AND EXPERIMENTS .......................... 39
3.1 Hardware ............................................................. 39
3.2 Software .............................................................. 42
3.3 Identification ......................................................... 43
3.4 6-DOF State Estimation ............................................. 45
3.5 Control Design for the Experiments ................................ 48
3.6 Simulation Results ................................................... 51
3.7 Experimental Results ................................................. 51

4 INVERSE KINEMATICS SOLUTION ....................................... 57
4.1 Novel 6-DOF Trajectory Inverse Kinematics ......................... 58
4.2 Verification ............................................................ 63

5 CONTROL SYSTEM DEVELOPMENT ..................................... 69
5.1 Control Objectives ..................................................... 70
5.2 Robust Position Control .............................................. 70
  5.2.1 DOb-based Controller ............................................ 71
  5.2.2 Simulation Study .................................................. 79
5.3 Robust Optimal Position Control .................................... 86
  5.3.1 Linearized Model .................................................. 86
  5.3.2 Continuous time MPC ............................................. 87
  5.3.3 Predicted plant response ......................................... 89
  5.3.4 Optimal control strategy .......................................... 89
  5.3.5 Constrained control ............................................... 91
  5.3.6 MPC tuning guidelines ........................................... 99
  5.3.7 Simulation Results ............................................... 101
5.4 Robust Force Estimation and Impedance Control ...................... 108
  5.4.1 Modeling of the Environment Dynamics and the Wind Effect 109
  5.4.2 Fast Tracking Recursive Least Squares ......................... 111
  5.4.3 Impedance Control ............................................... 114
  5.4.4 Simulation Results ............................................... 115
## TABLE OF CONTENTS

### 6 CONCLUSIONS
- 6.1 Conclusions ................................................................. 125
- 6.2 Future Plans ............................................................... 127

### LIST OF PUBLICATIONS ................................................. 129

### REFERENCES ............................................................... 131

### APPENDICES .............................................................. 141

A Quadrotor Platform ..................................................... 143

B Robotic Arm ................................................................. 151

C Forward Kinematics ....................................................... 161

D System Jacobian and Parametrization ............................. 163

E Laser Range Finder ......................................................... 173

F Ultrasonic Ranger .......................................................... 179

G System Internal Connections .......................................... 183

H Robot Operating System (ROS) ....................................... 185
- H.1 ROS ............................................................................. 185
- H.2 C++ Node for Extracting the Euler Angles from the Quaternion .......... 187
- H.3 ROS-based Arduino Code ............................................... 189
- H.4 Altitude Compensation Node ........................................... 193
- H.5 Joystick teleoperation Node ............................................. 194

I 6-DOF Torque/Force Sensor ............................................ 201

J Inverse Kinematics .......................................................... 205
## LIST OF TABLES

2.1 DH parameters for the manipulator .................................................. 22
3.1 System Parameters ................................................................. 46
3.2 Controller Parameters in Simulation ............................................. 52
3.3 Experimental Controller Parameters ............................................ 53
5.1 PD-DOb parameters ................................................................. 81
5.2 MPC-DOb controller parameters ................................................... 102
5.3 Comparison study: MPC-DOb vs PD-DOb ....................................... 103
5.4 FTRLS-DOB simulation parameters ............................................. 118
A.1 AscTec Pelican, technical Data [1] ............................................. 144
A.2 AscTec Atomboard technical specifications [1] .............................. 149
E.1 Laser Range Finder, URG-04LX, specifications [2,3] ....................... 175
E.2 Laser Range Finder, URG-04LX, Pins of the interface connector [2] .... 177
E.3 Laser Range Finder, URG-04LX, the USB connector ........................ 177
F.1 Ultrasonic, SRF04, technical specifications [4] ............................... 180
I.1 Pin Connections of the 6-DOF torque/force sensor [5] ....................... 202
LIST OF FIGURES

1.1 Examples of tasks that need robotics especially aerial robots ............... 2
1.2 Typical quadrotor [6] .................................................................................. 3
1.3 Quadrotor carrying a payload via a gripper [7] ........................................ 3
1.4 Quadrotor carrying a payload via a cable [8] ............................................. 4
1.5 Swooping quadrotor with claws [9] ............................................................ 5
1.6 Quadrotor carrying a payload via three 2-DOF arms [10] .......................... 5
1.7 Quadrotor carrying a payload via 6-DOF arm [11] ................................. 6
1.8 Quadrotor carrying a payload via 2-DOF robotic arm in with parallel joints’ axes [12] ................................................................. 6
1.9 Quadrotor carrying a payload via 2-DOF robotic arm that has unique topology [13] ................................................................. 7
2.1 3D CAD model of the proposed system .................................................. 18
2.2 Schematic of Quadrotor Manipulation System with relevant frames .......... 19
2.3 Different views for the quadrotor platform extracted from a SOLIDWORKS model ................................................................. 20
2.4 Manipulator’s structure analysis ............................................................... 21
2.5 Experimental Validation of the system design ......................................... 22
3.1 Aerial manipulation functional block diagram ....................................... 40
3.2 Close sight on position sensors connections ........................................... 42
3.3 Experiment to estimate the rotor coefficients ......................................... 43
3.4 Relation between the generated thrust force and the quadrotor’s squared motor speed ......................................................... 43
3.5 Relation between the generated drag moment and the quadrotor’s squared motor speed ......................................................... 45
3.6 Block diagram of the state estimation scheme ...................................... 47
LIST OF FIGURES

3.7 Block diagram of the control structure; Note that the circle block does the summation function, and the input with minus signal (-) means subtraction while the input with no signal means addition .......................... 49
3.8 Simulation results: actual position/command velocity history of the system . 54
3.9 Pictures taken during the experiment .......................... 55
3.10 Experimental results: measured position/command velocity history of the system .................................................. 56
4.1 Block diagram of the inverse kinematic algorithm .......................... 59
4.2 The actual response of the end-effector pose in the case of the slower trajectories: a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$ .......................... 65
4.3 End-effector 3D trajectory in the case of the slower trajectories .......................... 66
4.4 The actual response of the end-effector pose in the case of the faster trajectories: a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$ .......................... 67
4.5 End-effector 3D trajectory in the case of the faster trajectories .......................... 68
5.1 Functional block diagram of the proposed PD-DOb control system .......................... 71
5.2 Block diagram of the conventional DOB internal loop .......................... 72
5.3 Block diagram of the modified DOB-based controller .......................... 73
5.4 Detailed block diagram of the PD-DOb control system .......................... 78
5.5 PD-DOb results: The actual response in the quadrotor/joint space; a) x, b) $y$, c) $z$, d) $\psi$, e) $\theta_1$, and f) $\theta_2$ .......................... 82
5.6 PD-DOb results: The actual response of the end-effector pose; a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$. The inner box of each figure represents zooming on that figure from instant 0 s to instant 5 s .......................... 83
5.7 PD-DOb results: 3D trajectory of end-effector with fixed orientation (The marker represents the end-effector orientation; Green, Blue, and Red for x-, y, and z-axis) .......................... 84
5.8 PD-DOb results: The required control efforts .......................... 85
5.9 Functional block diagram of the proposed MPC-DOb control system .......................... 86
5.10 Block diagram of the detailed MPC-DOb control system for the quadrotor manipulation system .......................... 101
5.11 MPC-DOb results: The actual response in the quadrotor/joint space; a) x, b) $y$, c) $z$, d) $\psi$, e) $\theta_1$, and f) $\theta_2$ .......................... 104
5.12 MPC-DOb results: The actual response of the end-effector pose; a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$ .......................... 105
5.13 MPC-DOb results: 3D task space trajectory .......................... 106
5.14 MPC-DOb results: The required controller efforts .................................................. 107
5.15 Functional block diagram of the FTRLS-DOb control system ................................. 108
5.16 Block diagram of the detailed FTRLS-DOb control system for the quadrotor manipulation system ................................................................. 116
5.17 Profile of the wind angle ....................................................................................... 117
5.18 Profile of the wind speed ...................................................................................... 117
5.19 FTRLS-DOb results: Actual response of the estimation of the contact force .......... 120
5.20 FTRLS-DOb results: Error norm of the estimation of the contact force ................. 121
5.21 FTRLS-DOb results: The actual response of the end-effector pose: a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$ ........................................ 122
5.22 FTRLS-DOb results: 3D trajectory of the end-effector pose (The marker represents the end-effector orientation; Green, Blue, and Red for x-, y-, and z-axis, respectively) .......................................................... 123
5.23 FTRLS-DOb results: The actual trajectories of the variables in the quadrotor/joint space ........................................................................ 124

A.1 AscTec Pelican quadrotor with full options [1] .................................................... 143
A.2 Mounting the propellers [1] ................................................................................. 144
A.3 Dimensions of the Pelican quadrotor [1] ............................................................. 145
A.4 AscTec AutoPilot functional block diagram [1] .................................................... 146
A.5 AscTec AutoPilot board pinout and connections [1] ............................................. 147
A.6 AscTec Power Board pinout and connections [1] ................................................ 148
A.7 Top view of the onboard computer pinout and connections [1] ............................ 149
A.8 Bottom view of the onboard computer pinout and connections [1] ....................... 150

B.1 Lightweight gripper [14] ...................................................................................... 151
B.2 Schematic diagram of the manipulator used to select the joints’ motors .......... 152
B.3 HS-5485HB digital servo motor [14] ................................................................. 153
B.4 HS-5485HB dimensions [14] ............................................................................. 153
B.5 HS-422 dimensions [14] ................................................................................... 153
B.6 Accessories for building the manipulator: a) Aluminum Tubing, b) Servo Bracket, c) Connector Hub, and d) Long “C” Servo Bracket [14] ............... 154
B.7 Dimensions of the accessories for building the manipulator in (mm): a) Aluminum Tubing, b) Servo Bracket, c) Connector Hub, and d) Long “C” Servo Bracket .............................................................. 155
B.8 SSC-32 servo controller [14] .............................................................................. 156
B.9 SSC32 board: Shorting bar jumpers to enable serial communication with the Arduino board ................................................................. 158
B.10 Arduino Mega 2560 board 15 .......................................................... 158
B.11 12 V to 5 V 1.5 A DC-DC converter circuit diagram 16 ............... 159

E.1 URG-04LX Laser Range Finder 2 ...................................................... 173
E.2 The principle of operation of the URG-04LX 3 .............................. 174
E.3 External dimension of the Laser Range Finder URG-04LX 2 .......... 176

F.1 Devantech SRF04 Ultrasonic Range Finder 4 ................................. 179
F.2 External dimension of the SRF04 4 .................................................. 180
F.3 Terminal Connections of the SRF04 4 .......................................... 181
F.4 Beam Pattern of the SRF04 4 ......................................................... 181
F.5 Timing diagram of the SRF04 4 ...................................................... 182

G.1 Power connections of the manipulator ............................................. 183

I.1 The Torque Force sensor with the interface mount to rotor to be identified .... 201
I.2 Torque/Force sensor axis orientation from the rotor side 5 ................. 202
Chapter 1

INTRODUCTION

In this chapter, the research background on the aerial manipulation systems, which induces the motivation of this research, is presented. The objectives of this research are clearly described and the outline of the thesis is briefly introduced.

1.1 Research Background

1.1.1 Overview of the Current Developed Aerial Manipulators

Recently, the evolvement of technology is much more concentrating on improving the standard of human living. Robots and mechatronic systems are expected to support human activities in daily life, industry, entertainment, service, outer space and medicine. Such activities include; inspection, maintenance, firefighting, service robot in crowded cities to deliver light stuff such as post mails or quick meals, rescue operation, surveillance, demining, performing tasks in dangerous places (e.g., Nuclear stations), or transportation in remote places (see Fig. 1.1).

There are two main categories of the robotic systems; ground robots and aerial robots. However, the ground robots have constraints and limitations to move and access all types of the environments and terrains. Recently, the aerial vehicles are utilized to overcome these constraints and they offer possibilities of speed and access to regions that are otherwise inaccessible to ground robotic vehicles. There are several types of aerial vehicles (fixed wing, helicopters, single rotor, twin rotor, quadrotor, hexacopter, octocopter, etc). Quadro-
1. INTRODUCTION

(a) Search & rescue in a disaster area 17
(b) Demining 18
(c) Accessing difficult terrains 19
(d) Maintenance in nuclear station 20

Figure 1.1: Examples of tasks that need robotics especially aerial robots

tor system (see Fig. 1.2) is one of the Unmanned Aerial Vehicles (UAVs) which possess certain essential characteristics, such as small size and cost, Vertical Take Off and Landing (VTOL), performing slow precise movements, and impressive maneuverability, which highlight its potential for use in vital applications. Such applications include; homeland security (e.g. Border patrol and surveillance), military surveillance, service and rescue tasks, and earth sciences (to study climate change, glacier dynamics, and volcanic activity) 21. However, most research on UAVs has typically been limited to monitoring and surveillance applications where the objectives are limited to “look” and “search” but “do not touch”. Due to their superior mobility, much interest is given to utilize them for aerial manipulation. Previous research on the quadrotor-based aerial manipulation systems can be divided into three approaches.
1.1 Research Background

In the first approach, a gripper or a tool is installed at the bottom of an UAV to transport a payload or interact with existing structures, see Fig. 1.3. In [7,22,23], a quadrotor with a gripper is used for transporting blocks and to build structures. In [24], a force sensor is used to apply a normal force to a surface. Accordingly, not only the attitude of the payload/tool is restricted to the attitude of the UAV, but also the accessible range of the end-effector is confined due to the fixed configuration of the gripper/tool with respect to the UAV body and blades. Consequently, the resulting aerial system has 4 Degrees Of Freedom (DOF); three translational DOF and one rotational DOF (Yaw), i.e., the gripper/tool cannot posses pitch or roll rotation without moving horizontally.

![Figure 1.2: Typical quadrotor [6]](image)

![Figure 1.3: Quadrotor carrying a payload via a gripper [7]](image)

The second approach is to suspend a payload with cables, see Fig. 1.4. In [8], an adaptive controller is presented to avoid swing excitation of a payload. In [25,27], specific attitude and position of a payload is achieved using cables connected to one or three quadrotors. However, this approach has a drawback that the movement of the payload cannot be always
1. INTRODUCTION

regulated directly because the manipulation is achieved using a cable which cannot always drive the motion of the payload as desired. To overcome these limitations, a third approach is developed in which an aerial vehicle is equipped with a robotic manipulator that can actively interact with the environment. For example, in [28], a test bed including two 4-DOF robot arms and a crane emulating an aerial robot is proposed. By combining the mobility of the aerial vehicle with the versatility of a robotic manipulator, the utility of mobile manipulation can be maximized. When employing the robotic manipulator, the dynamics of the robotic manipulator are highly coupled with that of the aerial vehicle, which should be carefully considered in the controller design. In addition, an aerial robot needs to tolerate the reaction forces from the interactions with the object or external environment. These reaction forces may affect the stability of an aerial vehicle significantly. Very few reports exist in the literature investigate the combination of aerial vehicle with robotic manipulator. In [9], the authors propose a flying robot that can similarly swoop down and grab objects in the way that an eagle can fish with its natural claws. Their proposed system consists of a quadrotor with a 1-DOF robotic arm, see Fig. 1.5. Kinematic and dynamic models of the quadrotor combined with arbitrary multi-DOF robot arm are derived using the Euler-Lagrangian formalism in [29]. Based on that, simulation results using the Cartesian impedance control for the combined system with 3-DOF robotic arm is presented in [30]. In addition, the effects of manipulators, two 4-DOF arms, on the quadrotor are simulated based on the dynamic model which considers a quadrotor and robotic arms separately, treating the arms as the disturbance to the quadrotor control loop. In [10], a quadrotor with light-weight manipulators,
three 2-DOF arms, are tested, although the movement of the manipulators is not explicitly considered during the design of the PID controller (see Fig. 1.6). In [11, 31, 32] a manipulator with more than two links were used. However, with these systems, the final payload of the quadrotor is decreased due to the excessive manipulator weight, see Fig. 1.7. In [12], an aerial manipulation using a quadrotor with a 2-DOF robotic arm is presented but with certain topology that disable the system from making arbitrary position and orientation of the end-effector as shown in Fig. 1.8. In this system, the axes of the manipulator joints are parallel to each other and parallel to one in-plane axis of the quadrotor. Thus, the system cannot achieve orientation around the second in-plane axis of the quadrotor without moving horizontally.

From the above discussion, the systems that use a gripper suffer from the limited allowable DOFs of the end-effector. The other systems have a manipulator with either two DOFs but in certain topology that disables the end-effector to track arbitrary 6-DOF trajectory, or more than two DOFs which decreases greatly the possible payload carried by the system.
1. INTRODUCTION

![Quadrotor carrying a payload via 6-DOF arm](image1)

**Figure 1.7:** Quadrotor carrying a payload via 6-DOF arm [11]

![Quadrotor carrying a payload via 2-DOF robotic arm in with parallel joints' axes](image2)

**Figure 1.8:** Quadrotor carrying a payload via 2-DOF robotic arm in with parallel joints' axes [12]

In [13, 33], a new aerial manipulation system is proposed that consists of 2-link manipulator, with two revolute joints whose axes are perpendicular to each other and the axis of the first joint is parallel to one in-plane axis of the quadrotor, see Fig. [1.9] Thus, the end-effector is able to reach arbitrary position and orientation without moving horizontally. However, the proof that this system can perform any desired trajectory in the task space has not been achieved yet due to the resulted complications in the inverse kinematics and control. This system is similar to a serial manipulator with eight joints, two of them are passive. In this research, a novel inverse kinematics analysis is presented to prove and test the robot capability to track a 6-DOF task space trajectories. Moreover, controller design is carried out that makes use of the inverse kinematics solution to track arbitrary 6-DOF task space trajectory. Also, experimental implementation of the system is carried out to check its capabilities which is not done before.
1.1 Research Background

1.1.2 Experimental Setup and Measurements

One of the key issues in the aerial manipulation system is achieving the position holding task. To do this task, the accurate measurements and estimation of the system states must be achieved. However, all the current developed systems [9,12,34–37] use a positioning scheme that depends on a high cost 3D Motion Capture System (e.g., VICON system) which will restrict the functions of the aerial manipulation to be within limited space indoors. However, most real applications require to do tasks in arbitrary places indoors and/or outdoors. Therefore, in this work, a proposed solution to this issue is presented and tested that is based on using onboard sensors. The experiments are carried out for point-to-point control using position holding strategy in teleoperation framework rather than trajectory tracking control. Autonomous trajectory tracking control needs high accurate sensing system which is not available.

1.1.3 Control System

The control of the quadrotor manipulation system is quite challenging because it is naturally unstable, and has strong nonlinearities. In addition, it is very susceptible to parameters variations due to adding/releasing the payload and the actuators’ losses. Moreover, it is subjected to external disturbances like wind. Furthermore, its measurements are very noisy due to the thrusters that rotate at very high speed. Also, it has fast dynamics. Therefore, there is a need to design and implement an efficient and low computational cost robust control technique to address these issues.
1. INTRODUCTION

Furthermore, the actuators (thrusters and manipulator motors) have constraints, which may degrade the performance if it is not considered during controller design. Moreover, it is had to apply an optimal control effort to reduce the required power of the thrusters such that one can save power and thus increase the mission time. Therefore, a Model Predictive Control (MPC) based control technique is proposed to address this issue.

In the motion control of the aerial manipulator, achievement of the compliance control is very important because the compliance control makes possible to perform flexible motion of the manipulator according to a desired impedance. In addition, for the teleoperation purpose, the users may need to sense the force applied at the end-effector. This is very critical demand in applications such as demining and maintenance. Therefore, one have to design a sensorless force estimation to avoid the complicated design resulted from using force sensor and the extra weight to be used in impedance control system and/or teleoperation system.

Since there are high order nonholonomic constraints connecting the variables of the proposed non-redundant system, special treatment is needed to account for controlling the independent and dependent variables.

The control objectives for the proposed robot can be divided into three main targets; robustness, optimal performance, and force estimation & impedance control.

1.1.3.1 Robustness

Conventionally, robotic manipulators are controlled by using model based control methods such as feedback linearization, which is applied to the proposed system in [13]. However, the conventional design methods are very sensitive to modeling errors which may lead to the degradation of system performance and stability. Model-free control methods have been proposed by using intelligent based control methods; however, they have a problem in real time implementation due to high computational cost. In [33], Feedback Linearization, Direct Fuzzy Logic Control, and Fuzzy Model Reference Learning Control techniques have been applied to the proposed system. In the literature, several robust control methods have been proposed to improve the performance of robotic manipulators.

Disturbance Observer (DOb)-based controller is one of the most popular methods in the
1.1 Research Background

field of the robust motion control due to its simplicity and computational efficiency. The motivation for using the DOb is to provide a robustness in front of a significant class of nonlinearities/uncertainties by estimating the nonlinear terms and external disturbances. Thus, one can solve the control problem relying on a set of linearized and decoupled Single Input Single Output (SISO) systems which are not affected by nonlinear/uncertain terms. As a result, a standard linear tracking controller can be used and the resulting control scheme is characterized by a low computational load with respect to the conventional nonlinear robust solutions.

The authors in [38, 39] present the principles of DOb-based control system. In the DOb based robust motion control systems, internal and external disturbances are observed by the DOb, and the robustness is simply achieved by feeding-back the estimated disturbances in an inner-loop. Another controller is designed in an outer loop so that the performance goals are achieved without considering internal and external disturbances. In [40], the DOb-based controller is designed to realize a nominal system which can control acceleration in order to realize fast and precise servo system, even if servo system has parameter variation and suffers from disturbance. In [41–47], DOb-based control technique has been applied to robotic systems and showed an efficient performance.

However, the previous work [38, 39, 41, 42, 46–48] in the DOb requires either the acceleration measurement or the velocity estimation. In fact, this is very difficult to be achieved and has limitations due to the available sensors for flying robots. In addition, it is feasible to measure the linear accelerations and angular velocities of the quadrotor via IMU while the angular velocities of the joints can be measured by encoders. Moreover, the estimation of the disturbances based on an estimated data (e.g., velocities), as in the case of the current developed DOb technique, will make the estimation error of these disturbances to be larger. In [49], the external wrench (i.e., external forces and moments) is estimated by using a model-based method for a simple UAV. In addition, the author utilizes the IMU data for the estimation. However, this method needs to know the dynamic model and it neglects some dynamics and nonlinearities. Also, it uses a nonlinear controller and estimates only the external disturbances without estimating the system dynamics. Thus, this technique is not a good choice for the considered aerial manipulator which is a high complex dynamic
1. INTRODUCTION

and kinematic robotic system.

To cope up with these limitations, the conventional DOb is modified to be compatible and feasible with the aerial manipulation system. The conventional DOb is redesigned and reformulated to use the linear acceleration and angular velocities data, which can be measured directly via the onboard IMU and encoders, to estimate the disturbances and nonlinearities.

1.1.3.2 Optimal Performance

MPC application to robotic systems in a true industrial environment, in which disturbances affect the robotic system and the model of the robot is inevitably inaccurate, is still limited. This is due to the high nonlinearities and uncertainties of dynamics in the robotics system. In [50,51], a nonlinear MPC is utilized to solve the problem of the control of highly nonlinear systems. In [52], MPC control design with feedback linearization and sliding mode control are utilized to achieve robust trajectory tracking of robotic manipulator. In [53], nonlinear MPC with nonlinear DOb is used to control a robotic system. However, the nonlinear control methods make the MPC to be more complex, has high computational cost, and is based on the knowledge of system model.

Based on the the linearization and robustness capabilities of the DOb, one can rely on the standard linear MPC methodology to design the controller of the outer loop of the DOb-based controller such that the system performance can be adjusted to achieve an objective tracking accuracy and speed. In addition, the actuators’ constraints can be considered with optimal control effort and a low computational load.

1.1.3.3 Sensorless Robust Force Estimation and Impedance Control

Control of the interaction between a robot manipulator and the environment is crucial for successful execution of a number of practical tasks where the robot end-effector has to manipulate an object or perform some operation on a surface. Impedance control has the ability to impose, along each direction of the task space, a desired dynamic relation between the end-effector pose and the force of interaction with the environment (i.e., desired impedance). The inclusion of force information in the control of robots increases their adaptability to uncertain environments, such as those found in assembly tasks, and provides safety
1.1 Research Background

against breakage due to excessive contact forces [54–56]. Impedance control is still a rather far adopted solution in aerial robotics applications. In [30, 57–59], an impedance control is designed for aerial manipulator without the need to measure/estimate the contact force. However, in such work, the authors neglect some dynamics as well as external disturbances. In addition, the proposed algorithm is model-based and it does not have a robustness capability. In [60], better results are obtained with feeding back the measured force when implementing impedance control. On the other hand, for the tele-manipulation, the use of force feedback is essential to assist the human operator in the remote handling of objects with a slave manipulator [61]. In addition, it is very critical to get the force feedback to achieve the cooperative grasping among aerial manipulation robots. If the load is heavier than the carrying capacity of a single UAV, multiple UAVs can distribute the load among them and move the object.

However, typically, forces are measured by a force sensor connected to the robot end-effector. This situation is infeasible in aerial robotics since force sensors have a lot of inherent limitations such as narrow bandwidth, sensing noise, and high cost. In addition, it needs complicated arrangements in the aerial manipulator that increase the weight which is a critical issue. Hence, to avoid a force sensor due to their limitations, the contact force has to be estimated. The estimated interaction force will be used in the impedance control algorithm and it can be utilized for tele-manipulation as well as for cooperative grasping for high payload transportation.

Identification of the environment dynamics are more severe when the environment displays sudden changes in its dynamic parameters which cannot be tracked by the identification process. In [62], it is found that in order to faithfully convey to the operator the sense of high frequency chattering of contact between the slave and hard objects, faster identification and structurally modified methods were required. Several techniques are proposed to estimate the contact force/environment dynamics. In [63], the DOB and Recursive Least Squares (RLS) are used to estimate the environment dynamics. However, in this method, two DOBs are used besides the RLS, and the estimation of contact force is activated only during the instance of contacting, thus there is a need to detect the instant at which the contact occurs. This is not practical approach especially if one targets an autonomous system.
1. INTRODUCTION

In [64-69], several techniques are proposed to achieve force control without measuring the force. However, these techniques are based on ignoring some dynamics and external disturbances, which will produce inaccurate force estimation. In [70], a scheme is proposed which allows a quadrotor to perform tracking tasks without a precise knowledge of its dynamics and under the effect of external disturbances and unmodeled aerodynamics. In addition, this scheme can estimate the external generalized forces. However, as the authors claim, this estimator can work perfectly with constant external disturbances. Moreover, the estimated forces contain many different types such as wind, payload, environment impacts, and unmodeled dynamics. Thus, it cannot isolate the end-effector force alone from the others. The authors in [49] present a model-based method to estimate the external wrench of a flying robot. However, this method assumes that there are no modeling errors and no external disturbance. Moreover, it estimates the external force as one unit and it cannot distinguish between external disturbance and the end-effector force which is needed to calculate. In addition, it uses a model based control which needs a full knowledge of the model.

Therefore, in this dissertation, another scheme is proposed to overcome these limitations of the currently developed techniques. Firstly, a DOb inner loop is used to estimate both the system nonlinearities and all external forces to compensate for them, as a result, the system acts like a linear decoupled SISO systems. Secondly, a fast tracking RLS algorithm is utilized with the linearization capabilities of DOb to estimate the contact force. Thirdly, a model-free robust impedance control of the quadrotor manipulation system is implemented. The DOb is designed in the quadrotor/joint space while the impedance control is designed in the task space such that the end-effector can track the desired task space trajectories besides applying a specified environment impedance. Thus, a Jacobian based algorithm is proposed to transform the control signal from the task space to the quadrotor/joint space coordinates.

1.2 Objectives of Research

Based on the reasoning in the previous section, this research proposes the control design and implementation of a new quadrotor manipulation system which will have great impacts in the human daily activities as well as other vital and critical applications.

The objectives of this research are clearly described as follows.
1.2 Objectives of Research

1.2.1 Design and Modeling of the Quadrotor Manipulation System

It is aimed to design a quadrotor with a 2-DOF manipulator that has a unique topology to enable the end-effector to track a desired 6-DOF trajectory. It has to provide solutions to limitations found on the currently developed aerial manipulation systems by having maximum mobility with minimum weight.

Kinematic and dynamic modeling of the system are aimed to facilitate designing and simulating the control of the system.

1.2.2 Experimental Setup and State Estimation

The whole system has to be built. An identification experiment is needed to find the actuators’ coefficients. A measurement scheme is implemented to get the vehicle states (i.e., pose and velocity). Furthermore, this measurement scheme has to work in different places indoors and has potential to work outdoors as well.

Real time quadrotor/joint space control is designed, with a formulation that enables the system to work in either teleoperation or autonomous mode for pick and place operation.

An experimental test has to be carried out to demonstrate the feasibility and the effectiveness of the proposed approach with holding/releasing a payload.

1.2.3 Inverse Kinematics

The proposed aerial manipulator has complexity in its inverse kinematics. In fact, inverse kinematics of the proposed system require a solution of complicated algebraic/differential equations to prove/achieve the end-effector ability to track arbitrary 6-DOF trajectories that lay within the capability of the used actuators. Therefore, in this work, a novel inverse kinematics algorithm is proposed, that solves the complex set of nonlinear differential algebraic equations approximately. This was not done before either for redundant or non-redundant aerial manipulator in the literature. After that, its accuracy has to be verified.
1. INTRODUCTION

1.2.4 Design of a Control Scheme

This robot has several issues in point of view of the control design, which have to be addressed and solved. In this research, a linear DOb-based control technique will be utilized as the base framework for the motion control. The DOb loop is used to enforce robust linear input/output behavior of the plant by canceling disturbances, measurement noise, and plant/model mismatch. The DOb loop has to use the measurements from the onboard sensors in order to reduce the disturbance estimation error.

To optimize the control signal, to save power consumption and thus increase mission time, and to consider the actuators’ constraints, a MPC controller is designed in the external loop to achieve the required closed-loop performance.

Furthermore, the manipulation tasks require estimating (applying) a certain force at the end-effector as well as the accurate positioning of it. Thus, in this research, a robust force estimation and impedance control scheme is proposed. The robustness will be achieved based on the DOb technique. Then, a tracking linear impedance controller will be designed. Unlike the currently developed techniques for force estimation, a technique based on linearization capabilities of DOb and a Fast Tracking Recursive Least Squares (FTRLS) algorithm is proposed, which takes into consideration all the system dynamics with no need for the environment contact indication.

1.3 Thesis Contributions

The added values of this thesis can be listed as following:

- Developing a new non-redundant 6-DOF quadrotor-based aerial manipulation system that has superior features over the currently developed aerial manipulators.

- Experimental implementation of the proposed robot with the development of the measurement and estimation scheme to get the full state vector of the platform. This facilitates working within arbitrary places indoors as well as outdoors.

- Proposing a novel inverse kinematics algorithm to prove the ability of the system to track arbitrary 6-DOF task space trajectories and facilitates designing the controller
in the quadrotor/joint space.

• Development of a control system for the proposed robot as follows:
  
  – Development of the Disturbance Observer technique to solve the robustness issue of this robot.
  
  – Development of a robust optimal control based on the MPC and DOb approaches.
  
  – Development of a robust sensorless force estimation and impedance control via the FTRLS and DOb techniques.

1.4 Thesis Plan

The Thesis is organized in 6 chapters as follows:

Chapter 1 – Introduction. This chapter introduces the background and the objectives of the research. The outline of the thesis is also briefly described in this chapter.

Chapter 2 – System Design and Modeling. This chapter initially describes the proposed flying robot. After that, the design of this system is presented. Then, the kinematic model is described. Finally, the system dynamics are reviewed.

Chapter 3 – System Construction and Experiments. This chapter starts by describing the construction of the experimental system from point of view of both hardware and software. Then, it presents the identification experiment to get the rotors’ aerodynamics parameters. After that, the 6-DOF system states measurements and estimation are presented. A PID controller with gravity compensation is designed and implemented in both simulation and experimental environments. The feasibility and effectiveness of the proposed system are verified by both simulation and experimental results.

Chapter 4 – Inverse Kinematics Solution. This chapter presents a novel inverse kinematics analysis to get the reference quadrotor/joint space trajectories from desired 6-DOF task space trajectories. The accuracy of the proposed algorithm is enlightened via numerical results.

Chapter 5 – Control System Development. This chapter describes the principles of the proposed DOb-based control system with a PD control in the outer loop. After that,
the stability analysis of the closed-loop is presented. Then, it presents the application of an optimal robust control based on both DOB and MPC approaches. Moreover, presents the application of a robust sensorless force estimation and impedance control based on both DOB and FTRLS approaches. The effectiveness of the proposed methods are verified by simulation results.

Chapter 6 – Conclusions. This chapter presents the overall conclusions of this research work and describes the future developments of the research.
Chapter 2

SYSTEM DESIGN AND MODELING

This chapter initially describes the proposed flying robot. After that, the design of this system is presented. Then, the kinematic model is described. Finally, the system dynamics are reviewed.

2.1 System Description

A 3D CAD model of the proposed system is shown in Fig. 2.1. The system consists mainly of two parts; the quadrotor and the manipulator. Fig. 2.2 presents a sketch of the proposed system with the relevant frames, which indicates the unique topology that permits the end-effector to achieve arbitrary 6-DOF trajectory. The frames are assumed to satisfy the Denavit-Hartenberg (DH) convention [71]. The manipulator has two revolute joints. The axis of the first revolute joint ($z_0$), which is fixed to the quadrotor, is parallel to the body $x$-axis of the quadrotor (see Fig. 2.2). The axis of the second joint ($z_1$) is perpendicular to the axis of the first joint and will be parallel to the body $y$-axis of the quadrotor at home (extended) configuration. Thus, the pitching and rolling rotation of the end-effector is now possible independently from the horizontal motion of the quadrotor. Hence, with this new system, the capability of manipulating objects with arbitrary location and orientation is achieved. By this non-redundant system, the end-effector can achieve 6-DOF motion with
minimum number of actuators/links, which is an important factor in flight. The proposed system is distinguished from all other previous systems in the literature by having maximum mobility with minimum weight. The resulted complexity of the inverse kinematics and control are handled latter to prove the capability of the end-effector to track the reference 6-DOF trajectory.

2.2 System Design

2.2.1 Quadrotor

The quadrotor components are selected such that it can carry an additional weight equals 650 g (larger than the total arm weight and the maximum payload). AscTec pelican quadrotor form Ascending Technologies is used as the quadrotor platform. Fig. 2.3 presents the different views, which are extracted from a SOLIDWORKS model, for the quadrotor platform.

The characteristics of this quadrotor can be listed as following:

- Lightweight and customizable tower structure that lets one mount diverse payloads
2.2 System Design

Figure 2.2: Schematic of Quadrotor Manipulation System with relevant frames (e.g., sensors, avionics, and manipulator).

- Carbon fiber material with mass = 1.3 kg, so it combines between the rigidity and lightweight.

- Maximum payload = 650 g. This should include the additional sensors, avionics, manipulator and the payload to be handled.

- Maximum thrust = 36 N.

- Maximum airspeed = 16 m/s, and maximum climb rate = 8 m/s.

See Appendix A for more technical details about the AscTec quadrotor.

2.2.2 Manipulator

A lightweight manipulator that can carry a payload of 200 g and has a maximum reach of 22 cm is deigned. Its weight is 200 g. The arm components are selected, purchased and assembled.

The arm components are:
2. SYSTEM DESIGN AND MODELING

Figure 2.3: Different views for the quadrotor platform extracted from a SOLIDWORKS model

- Three DC motors; HS-422 (Max torque = 0.4 N.m) for the gripper, HS-5485HB (Max torque = 0.7 N.m) for joint 1, and HS-422 (Max torque = 0.4 N.m) for joint 2 [14].

- The manipulator structure accessories are; Aluminum Tubing - 1.50 in diameter, Aluminum Multi-Purpose Servo Bracket, Aluminum Tubing Connector Hub, and Aluminum Long 'C' Servo Bracket with Ball Bearings [14].

- Lightweight grasping mechanism [14].

See Appendix [I] for details about the manipulator’s dimensions and design.

The safety of the manipulator design, with respect to the strength and rigidity, is checked through finite element analysis using ANSYS software (see Fig. 2.4). From these figures, the maximum deflection is about 0.6 mm, which is smaller than the allowable value which equals 1 mm. In addition, the maximum stress of the structure is 113 MPa, which is smaller than the yield strength of aluminum alloy that is 270 MPa. Also, the bearings and gripper are selected to sustain the loads. Therefore, this design is safe.

The design of the proposed system is validated via experimental test as shown in Fig. 2.5.

2.3 Forward Kinematics

In this section, the position and velocity kinematic analysis are presented. Let $\Sigma_b, O_b-x_b y_b z_b$, denotes the vehicle body-fixed reference frame with its origin at the quadrotor center of mass, see Fig. 2.2. Its position with respect to the world-fixed inertial reference frame,
2.3 Forward Kinematics

\[ p_b = [x \ y \ z]^T \in \mathbb{R}^3, \]  

\[ R_b = \begin{bmatrix} 
C_\psi C_\theta & S_\psi S_\theta C_\psi - S_\psi C_\phi & S_\psi S_\phi + C_\psi S_\theta C_\phi \\
S_\psi C_\theta & C_\psi C_\phi + S_\psi S_\theta S_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\
- S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix}, \quad (2.1) \]
2. SYSTEM DESIGN AND MODELING

where $\Phi_b = [\psi \ \theta \ \phi]^T \in \mathbb{R}^3$ is the triple ZYX yaw-pitch-roll angles. Note that $C_\cdot$ and $S_\cdot$ are short notations for $\cos(\cdot)$ and $\sin(\cdot)$, respectively. Let us consider the frame $\Sigma_e$, $O_2- x_2 y_2 z_2$, attached to the end-effector of the manipulator, see Fig. 2.2. Let us define $\Theta = [\theta_1 \ \theta_2]^T \in \mathbb{R}^2$ to be the vector of joint angles of the manipulator. Table 2.1 presents the DH parameters for the two link manipulator. Transformation from $\Sigma_e$ to $\Sigma$ is given by

$$A_e = A_b A_0^b A_1^0 A_2^1. \quad (2.2)$$

| Link($i$) | $d_{mi}$ | $a_{mi}$ | $\alpha_{mi}$ | $\theta_{mi}$ |
|-----------|---------|---------|-------------|-------------|
| 0         | $-l_0$  | 0       | $-\pi/2$    | $-\pi/2$    |
| 1         | 0       | $l_1$   | $\pi/2$     | $\theta_1$  |
| 2         | 0       | $l_2$   | 0           | $\theta_2$  |
Transformation from $\Sigma_e$ to $\Sigma_b$ is given by

\[
A^b_e = A^b_2 = A^b_0 A^0_1 A^1_2 = \begin{bmatrix} R^b_e & p^b_{eb} \\ O_{1\times 3} & 1 \end{bmatrix},
\]

where $O_{1\times 3}$ represents a $1 \times 3$ zeros vector, $p^b_{eb}$ describes the position of $\Sigma_e$ with respect to $\Sigma_b$ expressed in $\Sigma_b$, and $R^b_e$ describes the orientation of $\Sigma_e$ w.r.t $\Sigma_b$. The matrices $A^b_0$, $A^0_1$, and $A^1_2$ can be obtained by

\[
A^b_0 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -l_0 \\ 0 & 0 & 0 & 1 \end{bmatrix},
\]

\[
A^0_1 = \begin{bmatrix} C_{\theta_1} & 0 & S_{\theta_1} & l_1 C_{\theta_1} \\ S_{\theta_1} & 0 & -C_{\theta_1} & l_1 S_{\theta_1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},
\]

\[
A^1_2 = \begin{bmatrix} C_{\theta_2} & -S_{\theta_2} & 0 & l_2 C_{\theta_2} \\ S_{\theta_2} & C_{\theta_2} & 0 & l_2 S_{\theta_2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\]

Thus, the position of $\Sigma_e$ with respect to (w.r.t) $\Sigma$ is given by

\[
p_e = p_b + R^b_e p^b_{eb}.
\]
The orientation of $\Sigma_e$ w.r.t $\Sigma$ can be defined by the rotation matrix

$$ R_e = R_b R_e^b, $$

(2.8)

The forward kinematics problem consists of determining the operational coordinates $\chi_e = [x_e \ y_e \ z_e \ \psi_e \ \theta_e \ \phi_e]^T \in \mathbb{R}^6$, as a function of the quadrotor/joint space coordinates $q = [x \ y \ z \ \psi \ \phi \ \theta_1 \ \theta_2]^T \in \mathbb{R}^8$. For solving the forward kinematics, the inputs are 8 variables, $q$, and the output are 6 variables, $\chi_e$, obtained from 6 algebraic equations. The end-effector position can be found from (2.7). Euler angles of the end-effector $\Phi_e$ can be computed from the rotation matrix of $R_e$, from (2.8), see Appendix C for the coding of the forward kinematics.

In the reset of this section, the velocity kinematic analysis is discussed.

The linear velocity of $\Sigma_e$ in the world-fixed frame, $\dot{p}_e$, is obtained by the differentiation of (2.7) as

$$ \dot{p}_e = \dot{p}_b - \text{Skew}(R_b p_e^b)\omega_b + R_b \dot{p}_e^b, $$

(2.9)

where $\text{Skew}(.)$ is the $(3 \times 3)$ skew-symmetric matrix operator [71], while $\omega_b$ is the angular velocity of the quadrotor expressed in $\Sigma$. The angular velocity $\omega_e$ of $\Sigma_e$ is expressed as

$$ \omega_e = \omega_b + R_b \omega_e^b, $$

(2.10)

where $\omega_e^b$ is the angular velocity of the end-effector relative to $\Sigma_b$ and is expressed in $\Sigma_b$.

The generalized velocity of the end-effector with respect to $\Sigma_b$, $v_e^b = [\dot{p}_e^b \ T \ \omega_e^b]^T \in \mathbb{R}^6$, can be expressed in terms of the joint velocities $\dot{\Theta}$ via the manipulator Jacobian, $J_e^b$, [72] as

$$ v_e^b = J_e^b \dot{\Theta}, $$

(2.11)
where \( J^b_{eb} \) is given by

\[
J^b_{eb} = \begin{bmatrix}
0 & l_2 C_{\theta_2} \\
S_{\theta_1} (l_1 + l_2 C_{\theta_2}) & l_2 C_{\theta_1} S_{\theta_2} \\
-C_{\theta_1} (l_1 + l_2 C_{\theta_2}) & l_2 S_{\theta_1} S_{\theta_2} \\
1 & 0 \\
0 & -S_{\theta_1} \\
0 & C_{\theta_1}
\end{bmatrix}.
\]

(2.12)

From (2.9) and (2.10), the generalized end-effector velocity, \( v_e = [\dot{p}_e^T \ \omega_e^T]^T \), can be expressed as

\[
v_e = J_b v_b + J_{eb} \dot{\Theta},
\]

(2.13)

where \( v_b = [\dot{p}_b^T \ \omega_b^T]^T \), \( J_b = \begin{bmatrix}
I_3 & -\text{Skew}(R_b p_b^{eb}) \\
O_3 & I_3 \\
O_3 & R_b
\end{bmatrix} \), \( J_{eb} = \begin{bmatrix}
R_b & O_3 \\
O_3 & R_b
\end{bmatrix} J^b_{eb}, \)

where \( I_m \) and \( O_m \) denote \((m \times m)\) identity and \((m \times m)\) null matrices, respectively. If the attitude of the vehicle is expressed in terms of yaw-pitch-roll angles, then (2.13) will be

\[
v_e = J_b Q_b \dot{\chi}_b + J_{eb} \dot{\Theta},
\]

(2.14)

with \( \chi_b = \begin{bmatrix}
p_b \\
\Phi_b
\end{bmatrix}, \ Q_b = \begin{bmatrix}
I_3 & O_3 \\
O_3 & T_b
\end{bmatrix}, \) where \( T_b \) describes the transformation matrix between the angular velocity \( \omega_b \) and the time derivative of Euler angles \( \dot{\Phi}_b \), and it is given as

\[
T_b(\Phi_b) = \begin{bmatrix}
0 & -S_\psi & C_\psi C_\theta \\
0 & C_\psi & S_\psi C_\theta \\
1 & 0 & -S_\theta
\end{bmatrix}.
\]

(2.15)

Thus, the system Jacobian is

\[
v_e = J \dot{q},
\]

(2.16)
where \( \dot{q} = [x_b^T \ \dot{\Theta}^T]^T \), and \( J = [J_b Q_b \ J_{eb}] \) and it is given as

\[
J = \begin{bmatrix}
1 & 0 & 0 & J_{14} & J_{15} & J_{16} & J_{17} & J_{18} \\
0 & 1 & 0 & J_{21} & J_{25} & J_{26} & J_{27} & J_{28} \\
0 & 0 & 1 & J_{34} & J_{35} & 0 & -C_{(\phi + \theta_1)}C_\theta (l_1 + l_2 C_\theta_2) & J_{38} \\
0 & 0 & 0 & C_\psi C_\theta & -S_\psi & 0 & C_\psi C_\theta & J_{48} \\
0 & 0 & 0 & C_\theta S_\psi & C_\psi & 0 & C_\theta S_\psi & J_{58} \\
0 & 0 & 0 & -S_\theta & 0 & 1 & -S_\theta & C_{(\phi + \theta_1)}C_\theta
\end{bmatrix},
\] (2.17)

with

\[
J_{14} = S_\theta ((C_\psi S_\phi - C_\phi S_\psi S_\theta)(l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) - C_\theta_1 (l_1 + l_2 C_\theta_2)(C_\psi C_\phi + S_\psi S_\phi S_\theta)
+ l_1 C_\theta_1 S_\phi (l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) + C_\phi C_\theta_1 S_\phi (l_1 + l_2 C_\theta_2)),
\] (2.18)

\[
J_{15} = -C_\psi (l_2 S_\theta_1 S_\theta_2 + C_\phi C_\theta_1 (l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) + C_\phi C_\theta_1 S_\phi (l_1 + l_2 C_\theta_2)),
\] (2.19)

\[
J_{16} = C_\theta_1 (l_1 + l_2 C_\theta_2)(C_\psi S_\phi + S_\psi S_\phi S_\theta) - (C_\psi S_\phi - C_\phi S_\psi S_\theta)(l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) - l_2 C_\theta S_\phi S_\theta_2,
\] (2.20)

\[
J_{17} = -(l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1)(C_\phi S_\psi - C_\psi S_\phi S_\theta) - (l_1 C_\theta_1 + l_2 C_\theta_1 C_\theta_2)(S_\psi S_\phi + C_\psi C_\phi S_\theta),
\] (2.21)

\[
J_{18} = l_2 C_\psi C_\phi C_\theta_2 - l_2 C_\theta_1 S_\theta_2 (C_\phi S_\psi - C_\psi S_\phi S_\theta) + l_2 S_\theta_1 S_\theta_2 (S_\psi S_\phi + C_\psi C_\phi S_\theta),
\] (2.22)
2.3 Forward Kinematics

\[ J_{24} = C_\psi C_\theta (l_2 S_\theta S_\theta_2 + C_\phi C_\theta (l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) + C_\theta C_\theta_1 S_\phi (l_1 + l_2 C_\theta_2) - S_\theta C_\theta_1 (l_1 + l_2 C_\theta_2)) - S_\theta (C_\phi S_\psi - C_\psi S_\phi S_\theta) - (S_\psi S_\phi + C_\psi C_\phi S_\theta)(l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) + l_2 C_\phi C_\theta S_\theta_2). \]  
\[(2.23)\]

\[ J_{25} = -S_\psi (l_2 S_\theta S_\theta_2 + C_\phi C_\theta (l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) + C_\theta C_\theta_1 S_\phi (l_1 + l_2 C_\theta_2)), \]  
\[(2.24)\]

\[ J_{26} = C_\theta_1 (l_1 + l_2 C_\theta_2) (C_\phi S_\psi - C_\psi S_\phi S_\theta) - (S_\psi S_\phi + C_\psi C_\phi S_\theta)(l_0 + l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1) + l_2 C_\phi C_\theta S_\theta_2, \]  
\[(2.25)\]

\[ J_{27} = (l_1 S_\theta_1 + l_2 C_\theta_2 S_\theta_1)(C_\psi C_\phi + S_\psi S_\phi S_\theta) + (l_1 C_\theta_1 + l_2 C_\theta_1 C_\theta_2)(C_\psi S_\phi - C_\phi S_\psi S_\theta), \]  
\[(2.26)\]

\[ J_{28} = l_2 C_\theta C_\theta_2 S_\psi + l_2 C_\theta_1 S_\theta_2 (C_\psi C_\phi + S_\psi S_\phi S_\theta) - l_2 S_\theta_1 S_\theta_2 (C_\psi S_\phi - C_\phi S_\psi S_\theta), \]  
\[(2.27)\]

\[ J_{34} = -C_\theta ((l_2 C_\phi + l_1 C_\phi) - l_0 S_\phi + (l_2 C_\phi + l_1 S_\theta_1 + l_2 C_\phi S_\theta))/2, \]  
\[(2.28)\]

\[ J_{35} = l_0 C_\phi S_\theta - l_2 C_\theta S_\theta_2 + l_1 C_\phi S_\theta S_\theta_1 + l_1 C_\theta_1 S_\phi S_\theta + l_2 C_\phi C_\theta_2 S_\theta + l_2 C_\theta_1 C_\theta_2 S_\phi S_\theta, \]  
\[(2.29)\]

\[ J_{38} = l_2 C_\phi C_\theta S_\theta_1 S_\theta_2 - l_2 C_\theta S_\theta + l_2 C_\phi C_\theta_1 S_\phi S_\theta_2, \]  
\[(2.30)\]
\[ J_{48} = C_{\theta_1} (C_\psi S_\phi + C_\psi C_\phi S_\theta) + S_{\theta_1} (C_\phi S_\psi - C_\psi S_\phi S_\theta), \quad (2.31) \]

\[ J_{58} = -C_{\theta_1} (C_\psi S_\phi - C_\psi S_\psi S_\theta) - S_{\theta_1} (C_\phi C_\psi + S_\psi S_\phi S_\theta), \quad (2.32) \]

See Appendix D for the programming of the system Jacobian.

Since the quadrotor is an under-actuated system, i.e., only 4 independent control inputs are available for the 6-DOF system, the position and the yaw angle are usually the controlled variables. Hence, it is worth to define the independent coordinate, \( \zeta = [x \ y \ z \ \psi \ \theta_1 \ \theta_2]^T \), as the controlled variables, and the dependent coordinates, \( \sigma_b = [\theta \ \phi]^T \), as the intermediate variables. Hence, it is worth rewriting the vector \( \chi_b \) as \( \chi_b = \begin{bmatrix} \eta_b \\ \sigma_b \end{bmatrix} \), \( \eta_b = \begin{bmatrix} p_b \\ \psi \end{bmatrix} \), \( \sigma_b = \begin{bmatrix} \theta \\ \phi \end{bmatrix} \).

Thus, the differential kinematics (2.14) will be

\[ v_e = J_\eta \dot{\eta}_b + J_\sigma \dot{\sigma}_b + J_{eb} \dot{\Theta} \]
\[ = J_\zeta \dot{\zeta} + J_\sigma \dot{\sigma}_b, \quad (2.33) \]

where \( \zeta = [\eta_b^T \ \Theta^T]^T \) is the vector of the controlled variables, \( J_\eta \) is composed by the first 4 columns of \( J_b Q_b \), \( J_\sigma \) is composed by the last 2 columns of \( J_b Q_b \), and \( J_\zeta = [J_\eta \ J_{eb}] \).

If the end-effector orientation is expressed via a triple of Euler angles, \( ZYX, \Phi_e \), the differential kinematics (2.33) can be rewritten in terms of the vector \( \dot{\chi}_e = [\dot{p}_e^T \ \dot{\Phi}_e^T]^T \) as follows

\[ \dot{\chi}_e = Q_e^{-1} (\Phi_e) v_e \]
\[ = Q_e^{-1} (\Phi_e) [J_\zeta \dot{\zeta} + J_\sigma \dot{\sigma}_b], \quad (2.34) \]

where \( Q_e \) is the same as \( Q_b \) but it is a function of \( \Phi_e \) instead of \( \Phi_b \).

### 2.4 Dynamic Model

For the manipulator dynamics, Recursive Newton Euler method is used to derive the equations of motion. Since the quadrotor is considered to be the base of the manipulator, the initial linear and angular velocities and accelerations, used in Newton Euler algorithm, are that of the quadrotor expressed in body frame. Applying the Newton Euler algorithm
to the manipulator considering that the link (with length $l_0$) that is fixed to the quadrotor is the base link, manipulator’s equations of motion can be obtained, in addition to, the forces and moments, from manipulator, that affect the quadrotor.

For the system structure, we assume:

**Assumption 2.1.** The quadrotor body is rigid and symmetric. The manipulator links are rigid.

For each link $i$, ($i = 0, 1, 2$), let us define

- $\omega_i$ is the angular velocity of frame $i$ expressed in frame $i$.
- $\dot{\omega}_i$ is the angular acceleration of frame $i$ expressed in frame $i$.
- $v_i$ is the linear velocity of the origin of frame $i$ expressed in frame $i$.
- $\dot{v}_c$ is the linear acceleration of the center of mass of link $i$ expressed in frame $i$.
- $\dot{v}_i$ is the linear acceleration of the origin of frame $i$ expressed in frame $i$.
- $r_i$ is the position vector from the origin of frame $i - 1$ to the origin of link $i$.
- $r_c$ is the position vector from the origin of frame $i - 1$ to the center of mass of link $i$.
- $g$ is the vector of gravity expressed in $\Sigma$.
- $z_i$ is a unit vector pointing along the $i^{th}$ joint axis and expressed in the $(i - 1)^{th}$ link coordinate system.
- $R_i$ is the rotation matrix from frame $i$ to frame $i - 1$.
- $I_i$ is the inertia matrix of link $i$ about its center of mass coordinate frame.
- $f_{i,i-1}$ / $n_{i,i-1}$ are the resulting force/moment exerted on link $i$ by link $i - 1$ at point $O_{i-1}$.
• $R^i$ is the rotation matrix from frame $\Sigma$ to frame $i$.

Applying a payload of mass equal to $m_p$ at the end-effector will change link 2’s parameters such as mass moments of inertia, total mass, and center of gravity as

$$I_2^2 = I_2^2 + m_2(d_{CG2} - d_{CG2})^2 + m_p(l_2 - \dot{d}_{CG2})^2,$$  \hspace{1cm} (2.35)

$$\dot{d}_{CG2} = \frac{m_2 d_{CG2} + m_p l_2}{m_2 + m_p},$$ \hspace{1cm} (2.36)

$$m_2 = m_2 + m_p,$$ \hspace{1cm} (2.37)

where $CG_2$ is the point of center of gravity of link 2, $d_{CG2} = l_2/2$, and (\_) refers to the value of the parameter after adding the payload. Changing the point of center of gravity of link 2 will make the $r_{c2}^2$ to be

$$r_{c2}^2 = [- (l_2 - \dot{d}_{CG2}) 0 0]^T.$$ \hspace{1cm} (2.38)

For the link 0, one calculates the following:

$$\omega_0^b = R_0^b \omega_b^b,$$ \hspace{1cm} (2.39)

$$v_0^b = R_0^b v_b^b,$$ \hspace{1cm} (2.40)

$$\dot{\omega}_0^b = R_0^b \dot{\omega}_b^b,$$ \hspace{1cm} (2.41)

$$\dot{v}_0^b = R_0^b \dot{v}_b^b,$$ \hspace{1cm} (2.42)

$$r_{b0}^0 = [0 l_0 0]^T,$$ \hspace{1cm} (2.43)
2.4 Dynamic Model

\[ r_{b0} = R_0 r'_{b0}, \quad (2.44) \]

\[ \omega_0^0 = \omega_b^0, \quad (2.45) \]

\[ \dot{\omega}_0^0 = \dot{\omega}_b^0, \quad (2.46) \]

\[ v_0^0 = v_b^0 + \omega_b^0 \times r_{b0}^0, \quad (2.47) \]

\[ \dot{v}_0^0 = \dot{v}_b^0 + \omega_b^0 \times r_{b0}^0 + \omega_b^0 \times (\omega_b^0 \times r_{b0}^0), \quad (2.48) \]

\[ \dot{\omega}_0 = R_0 \dot{\omega}_0^0, \quad (2.49) \]

\[ \omega_0 = R_0 \omega_0^0, \quad (2.50) \]

\[ \dot{v}_0 = \dot{v}_b + \dot{\omega}_0 \times r_{b0} + \omega_0 \times (\omega_0 \times r_{b0}). \quad (2.51) \]

For link 1, one can calculate the following:

\[ \omega_1^1 = R^1 \omega_0 + R^1_0 \dot{\theta}_1 z_0^0, \quad (2.52) \]

\[ \dot{\omega}_1^1 = R^1 \dot{\omega}_0 + R^1_0 \ddot{\theta}_1 z_0^0 + R^1 \omega_0 \times (R^1_0 \dot{\theta}_1 z_0^0), \quad (2.53) \]

\[ \dot{v}_1^1 = R^1 \dot{v}_0 + \dot{\omega}_1^1 \times r_1^1 + \omega_1^1 \times (\omega_1^1 \times r_1^1), \quad (2.54) \]

\[ \dot{v}_{c1}^1 = \dot{v}_1^1 + \dot{\omega}_1^1 \times r_{c1}^1 + \omega_1^1 \times (\omega_1^1 \times r_{c1}^1). \quad (2.55) \]
For link 2, one can calculate the following:

\[ \omega_2^2 = R_1^2(\omega_1^1 + \dot{\theta}_2 z_1^1), \]  
(2.56)

\[ \dot{\omega}_2^2 = R_1^2(\dot{\omega}_1^1 + \ddot{\theta}_2 z_1^1 + \omega_1^1 \times (\dot{\theta}_2 z_1^1)), \]  
(2.57)

\[ \dot{v}_2^2 = R_1^2(\dot{v}_1^1 + \dot{\omega}_2^2 \times r_2^2 + \omega_2^2 \times (\omega_2^2 \times r_2^2)), \]  
(2.58)

\[ \dot{v}_{c2}^2 = \dot{v}_2^2 + \dot{\omega}_2^2 \times r_{c2}^2 + \omega_2^2 \times (\omega_2^2 \times r_{c2}^2). \]  
(2.59)

The inertial forces and moments acting on link \( i \) are given by

\[ F_i^i = -m_i \dot{v}_{c_i}^i, \]  
(2.60)

\[ N_i^i = -I_i^i \omega_i^i - \omega_i^i \times I_i^i \omega_i^i. \]  
(2.61)

The total forces and moments acting on link \( i \) are given by

\[ f_{i,i-1}^i = f_{i+1,i}^i - m_i g^i - F_i^i, \]  
(2.62)

\[ n_{i,i-1}^i = n_{i+1,i}^i + (f_i^i + r_{c_i}^i) \times f_{i,i-1}^i - r_{c_i}^i \times f_{i+1,i}^i - N_i^i, \]  
(2.63)

\[ f_{i,i-1}^{i-1} = R_i^{i-1} f_{i,i-1}^i, \]  
(2.64)

\[ n_{i,i-1}^{i-1} = R_i^{i-1} n_{i,i-1}^i, \]  
(2.65)

where

\[ z_0^0 = [0 \ 0 \ 1]^T, \]  
(2.66)
2.4 Dynamic Model

\[ r_1^1 = [a_1 \ d_1 \sin(\alpha_1) \ d_1 \cos(\alpha_1)]^T, \]  
(2.67)

\[ r_{c_1}^1 = [-l_1/2 \ 0 \ 0]^T, \]  
(2.68)

\[ z_1^1 = [0 \ 0 \ 1]^T, \]  
(2.69)

\[ r_2^2 = [a_2 \ d_2 \sin(\alpha_2) \ d_2 \cos(\alpha_2)]^T. \]  
(2.70)

The gravity vector expressed in frame \( i \) is

\[ g^i = R^i g, \]  
(2.71)

where

\[ R^1 = R_0^1 R_0^0 R^b, \]  
(2.72)

\[ R^2 = R_1^2 R^1, \]  
(2.73)

and

\[ g = [0 \ 0 \ -g_r]^T, \]  
(2.74)

where \( g_r \) is the gravity acceleration.

The torques acting on joints 1 and 2 are finally given by

\[ \tau_{m_1} = (n_{1,0}^0)^T z_0^0 + b_1 \dot{\theta}_1, \]  
(2.75)

\[ \tau_{m_2} = (n_{2,1}^1)^T z_1^1 + b_2 \dot{\theta}_2, \]  
(2.76)

where \( b_1 \) and \( b_2 \) are the friction coefficients.

The interaction forces and moments of the manipulator acting on the quadrotor expressed
2. SYSTEM DESIGN AND MODELING

in \( \Sigma_b \), \( F_{m,q}^b \) and \( M_{m,q}^b \) are given as

\[
F_{m,q}^b = R_0^b(-f_1^0),
\]
\[ (2.77) \]

\[
M_{m,q}^b = R_0^b(-n_{1,0}^0 + r_0^b \times (-f_1^0)).
\]
\[ (2.78) \]

The interaction forces expressed in the inertial frame can be obtained by

\[
F_{m,q} = R_b F_{m,q}^b.
\]
\[ (2.79) \]

The equations of motion of the manipulator can be reformulated as

\[
M_1(q) \ddot{\theta}_1 = \tau_{m_1} + N_1(q, \dot{q}, \ddot{q}),
\]
\[ (2.80) \]

\[
M_2(q) \ddot{\theta}_2 = \tau_{m_2} + N_2(q, \dot{q}, \ddot{q}),
\]
\[ (2.81) \]

where \( M_1(q) \), \( M_2(q) \), \( N_1(q, \dot{q}, \ddot{q}) \), and \( N_2(q, \dot{q}, \ddot{q}) \) are nonlinear terms, and they are functions of the system states.

The Newton Euler method is used to find the equations of motion of the quadrotor after adding the forces/moments from the manipulator. They are given by (2.82-2.87).

\[
m \ddot{x} = T(C_\phi S_\theta C_\phi + S_\psi S_\phi) + F_{m,q_x}
\]
\[ (2.82) \]

\[
m \ddot{y} = T(S_\psi S_\theta C_\phi - C_\psi S_\phi) + F_{m,q_y}
\]
\[ (2.83) \]

\[
m \ddot{z} = -mg_r + TC_\theta C_\phi + F_{m,q_z}
\]
\[ (2.84) \]

\[ I_x \ddot{\phi} = \dot{\phi}(I_y - I_z) - I_r \ddot{\Omega} + \tau_a + M_{m,q_\phi}^b \]
\[ (2.85) \]
\[ I_y \ddot{\theta} = \dot{\psi}\dot{\phi}(I_z - I_x) + I_r\ddot{\phi} + \tau_{a_2} + M_{m,q\theta}^b \]  
(2.86)

\[ I_x \ddot{\psi} = \dot{\theta}\dot{\phi}(I_x - I_y) + \tau_{a_3} + M_{m,q\psi}^b \]  
(2.87)

where \( F_{m,qx}, F_{m,qy}, \) and \( F_{m,qz} \) are the interaction forces resulted from the manipulator and affected the quadrotor in \( x, y, \) and \( z \) directions expressed in the inertial frame and \( M_{m,q\phi}^b, M_{m,q\theta}^b, \) and \( M_{m,q\psi}^b \) are the interaction moments from the manipulator to the quadrotor around \( x_b, y_b, \) and \( z_b \) directions.

The variables in (2.82-2.87) are defined as follows: \( m \) is the mass of the quadrotor. Each rotor \( j \) has angular velocity \( \Omega_j \) and it produces thrust force \( F_j \) and drag moment \( M_j \) which are given by

\[ F_j = K_{f_j}\Omega_j^2, \]  
(2.88)

\[ M_j = K_{m_j}\Omega_j^2, \]  
(2.89)

where \( K_{f_j} \) and \( K_{m_j} \) are the thrust and drag coefficients, respectively.

\( T \) is the total thrust applied to the quadrotor from all four rotors, and it is given by

\[ T = \sum_{j=1}^{4} F_j. \]  
(2.90)

\( \tau_{a_1}, \tau_{a_2}, \) and \( \tau_{a_3} \) are the three input moments about the three body axes, \( x_b, y_b, z_b, \) and they are given as

\[ \tau_{a_1} = d_q(F_4 - F_2), \]  
(2.91)

\[ \tau_{a_2} = d_q(F_3 - F_1), \]  
(2.92)

\[ \tau_{a_3} = -M_1 + M_2 - M_3 + M_4. \]  
(2.93)

\( d_q \) is the distance between the quadrotor center of mass and rotor rotational axis. \( \Omega \) is given
by
\[ \Omega = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4. \] (2.94)

\( I_r \) is the rotor inertia. \( I_f \) is the inertia matrix of the vehicle around its body-frame, and it is given as
\[ I_f = I_i \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}. \] (2.95)

The dynamical model of the quadrotor-manipulator system can be rewritten in a matrix form as follows
\[
M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + \dot{d}_{ex} = \tau,
\]
\[
\tau = B_m u,
\] (2.96)

where \( M \in \mathbb{R}^{8 \times 8} \) represents the symmetric and positive definite inertia matrix of the combined system, \( C \in \mathbb{R}^{8 \times 8} \) is the matrix of Coriolis and centrifugal terms, \( G \in \mathbb{R}^8 \) is the vector of gravity terms, \( d_{ex} \in \mathbb{R}^8 \) is the vector of external disturbances, \( \tau \in \mathbb{R}^8 \) is the vector of the input generalized forces, \( u = [F_1 \ F_2 \ F_3 \ F_4 \ \tau_{m_1} \ \tau_{m_2}]^T \in \mathbb{R}^6 \) is vector of the actuator inputs, and
\[
B_m = H_m(q) \ N_m(K_{f_i}, K_{m_j}, d_q) \in \mathbb{R}^{8 \times 6}
\]
is the input matrix which is used to generate the body forces and moments from the actuator inputs. The control matrix, \( N_m \in \mathbb{R}^{8 \times 6} \), is given by
\[
N_m = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
\gamma_1 & -\gamma_2 & \gamma_3 & -\gamma_4 & 0 & 0 & 0 & 0 \\
-d_q & 0 & d_q & 0 & 0 & 0 & 0 & 0 \\
0 & -d_q & 0 & d_q & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & K_{\tau_1} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & K_{\tau_2} & 0 & 0 \\
\end{bmatrix}, \] (2.97)
where $\gamma_j = K_m j / K_f j$. $K_{\tau_1}$ and $K_{\tau_2}$ are coefficients that are used to emulate the occurring of faults in the motor’s constants of joints 1 and 2, respectively. $H_m \in \mathbb{R}^{8 \times 8}$ is a matrix that transforms the body input forces to be expressed in $\Sigma$ and is given by

$$H_m = \begin{bmatrix} R_b & O_3 & O_2 \\ O_3 & T_b^T R_b & O_2 \\ O_{2 \times 3} & O_{2 \times 3} & I_2 \end{bmatrix}.$$  \hspace{1cm} (2.98)

From the equations of the translation dynamics part of (2.82-2.84), one can extract the expressions of the second order nonholonomic constraints as

$$\sin(\phi) - \frac{\ddot{x}_f S \phi - \ddot{y}_f C \phi}{\sqrt{\ddot{x}_f^2 + \ddot{y}_f^2 + \ddot{z}_f^2}} = 0, \hspace{1cm} (2.99)$$

$$\tan(\theta) - \frac{\ddot{x}_f C \phi + \ddot{y}_f S \phi}{\ddot{z}_f} = 0, \hspace{1cm} (2.100)$$

where $\ddot{x}_f = \ddot{x} - \frac{F_{m,qx}}{m}$, $\ddot{y}_f = \ddot{y} - \frac{F_{m,qy}}{m}$, and $\ddot{z}_f = \ddot{z} + g_r - \frac{F_{m,qz}}{m}$.

It is to be noted that the force terms in the above equations are also function of the system states and their derivatives. Equations (2.99) and (2.100) can be solved for the desired trajectories of $\phi$ and $\theta$ through substituting by the desired trajectories of the other variables. These nonholonomic constrains will be utilized later to solve both the inverse kinematics and control problems.

The next step is to build the experimental system to test the system feasibility.
In this chapter, the whole system is built and its parameters are identified. Furthermore, experimental setup with a solution to the problem of the state estimation of the quadrotor manipulation system is introduced taking into consideration the position of the manipulator with respect to the sensors. This solution is based on a combination of several sensors and data fusion. Quadrotor/joint space control is designed, with a formulation which enables the system to work in either teleoperation or autonomous mode, based on PID with gravity compensation technique. After that, this controller is implemented in both simulation and real time environments. An experimental test is implemented to demonstrate the feasibility and the effectiveness of the proposed system in the presence of holding and releasing a payload.

3.1 Hardware

Fig. 3.1 shows the proposed experimental implementation of the whole connected system with the user interface and measurement system.

The quadrotor is equipped with 4 rotors with 10 inch diameter, which allow to carry an additional weight of about 650 g. Depending on the battery size and payload, flight times between 10 and 20 minutes can be achieved. In addition, the quadrotor has a Flight Control
3. SYSTEM CONSTRUCTION AND EXPERIMENTS

Figure 3.1: Aerial manipulation functional block diagram

Unit (FCU) “AscTec Autopilot” as well as a suitable structure enabling one to easily mount different payloads like computer boards, onboard sensors, and the robotic arm with its avionics. The FCU has an Inertial Measurement Unit (IMU) as well as two 32-Bit 60 MHz ARM-7 microcontrollers used for data fusion and flight control. One of these microcontrollers, the Low Level Processor (LLP), is responsible for the hardware communication, an emergency controller, and IMU sensor data fusion. An attitude and GPS-based position controller is implemented also on this processor. The other microcontroller, High Level Processor (HLP), is dedicated for custom code. All relevant and fused IMU data is provided at an update rate of 1 kHz via a high speed serial interface. In particular, this comprises body accelerations, body angular velocities, magnetic compass, height measurement by an air pressure sensor, and the estimated attitude of the vehicle. For the onboard expensive computations, a 1.6 GHz Intel Atom-based embedded computer is used. This computer is equipped with 1 GB RAM, a Micro SD card slot for the operating system, a 802.11n based miniPCI Express WiFi card. See Appendix A for more technical details.

State estimation is one of the key issue to implement quadrotor manipulation system. The available onboard sensor, IMU can only provide the quadrotor’s orientation, angular rates, and linear Accelerations. However, one can not depend on the accelerations to get linear position due to large integration drift for long term motion. Thus, there is a need to an external positioning system which may be GPS, 3D motion capture system, onboard
3.1 Hardware

camera, or range sensors (laser or sonar). However, GPS is not a reliable service (accuracy 3 – 15 m) as its availability can be limited by urban canyons and is completely unavailable in indoor environments. Onboard camera is tried but the available one (Web CAM) has low resolutions and gives bad results. As a result, finally, the range sensors are used.

For measuring the horizontal position, $x$ and $y$, an Hokuyo URG-04LX Laser Range Finder is used [2], see Fig. 3.2. It is a 2D laser range finder with a range of 4 m and a field of view of 240 degrees. It has an update rate of 30 Hz with resolution of 1 mm. It has weight of 160 g. It is connected to the onboard computer through USB connection such that the high computation processing of laser data can be carried out onboard. This sensor is placed on the top center of the quadrotor. See Appendix E for more technical details.

The vertical position, $z$, is measured using Ultrasonic Ranging module SRF04 [4], see Fig. 3.2. It is a 1D sonar range finder with a range of 3 cm to 3 m and a resolution of 3 cm. It has an update rate of 40 Hz. It is mounted downward under one of the rotors such that it is not affected by the movement of the manipulator, so one can avoid the sensor misreadings. See Appendix F for more technical details. The data from this sensor is acquired and processed by using an Arduino board [15] which is connected to the onboard computer by USB connection.

The 2-link manipulator’s components are: Three DC motors; HS-422 (Max torque = 0.4 N.m) for the gripper, HS-5485HB (Max torque = 0.7 N.m) for joint 1, and HS-422 (Max torque = 0.4 N.m) for joint 2. Motor’s Driver (SSC32) which is used as an interface between the main control unit and the motors. Wireless Play Station 2 (WPS2) R/C is used to send commands to manipulator’s motors remotely. The manipulator structure components are; Aluminum Tubing - 1.50 in diameter, Aluminum Multi-Purpose Servo Bracket, Aluminum Tubing Connector Hub, and Aluminum Long “C” Servo Bracket with Ball Bearings. The Arduino board (Mega 2560) is utilized as an interface between the low level peripherals (such as ultrasonic sensor, WPS2 receiver, and motor driver (SSC32)), and the onboard computer. See Appendix E for more technical details.

The Power management system is setup as following: A 11.1 V Li-Po battery is used to power the motors and the electronics on board. A voltage regulator circuit is used to convert this voltage to 5 V DC to power the manipulator motors and its avionics. The Hokuyo sensor,
Ultrasonic sensor, and the Arduino board are powered by the 5 V from the USB port of the onboard computer. A low voltage detection is implemented in the quadrotor making it lands once the battery voltage drops below 8 V. See Appendix G for power connections.

The ground station consists of a computer, joystick, and Futuba R/C. It allows a user to tele-operate the system such as visualization, sending commands, and emergency recovery.

In this work, a position controller and the position data fusion algorithms are implemented on the HLP, based on the Hokuyo and Ultrasonic sensors input from the onboard computer and the inertial data provided by the LLP. On the LLP, the attitude controller is used as inner loop.

3.2 Software

An Ubuntu Linux 12.04, with Robot Operating System (ROS) framework (fuerte version) [73], is installed on both the onboard computer and the ground station computer.

Since multiple different computation subsystems (PC, Onboard computer, autopilot board, and Arduino board) are used, which need to communicate among each other, the
3.3 Identification

ROS is used as a middle-ware, see Appendix H for more details about ROS. This also enable us to communicate with the ground station over the WiFi data link for monitoring and control purposes. In addition, the ROS framework has the essential drivers and software for processing the data from the Hokuyo sensor to find the horizontal position.

The algorithm on the HLP is implemented based on a Software Development Kit (SDK) that provides all the communication routines to allow us to send/receive the low level commands/measurements to/from the LLP. This algorithm is used to implement the data fusion algorithm and the position control. A C++ software (i.e., packages and nodes) is developed under ROS to manage and process data among the the computers systems.

A program is implemented on Arduino board based on a C++ Arduino IDE to acquire and process data from the sonar to get the vertical position. In addition, it receives the manipulator’s commands from the WPS2 and send the control signal to the motor drivers.

3.3 Identification

The aim of this section is to find the real parameters of the system to be used in the control design and to emulate a quite realistic setup during the simulation study. The identified parameters include the structure parameters (e.g., mass, geometrical parameters, and mass moment of inertia) and rotor parameters (i.e., $K_{f_j}$ and $K_{m_j}$). To calculate the structure parameters, a 3D CAD model is developed using SOLIDWORKS. To estimate the rotor parameters, an experimental setup is carried out, see Fig. 3.3 In this experiment, the rotor is mounted on a 6-DOF torque/force sensor (see Appendix I for more technical details) that is connected to a NI Data Acquisition Card (NI DAC). Then, the DAC is connected to a PC, running SIMULINK program as an interface, to read data from DAC. The velocity of rotor is changed gradually, and in each time, the generated thrust and drag moment is measured and recorded using SIMULINK program. By using MATLAB Curve Fitting toolbox the acquired data of thrust and moment is fitted to be in the form of (2.88 and 2.89). Thus, the thrust and moment coefficients can be obtained.

Figure 3.4 (3.5) shows the relationship between generated thrust force (drag moment) and the squared rotor speed.

To measure the goodness of data fitness, the Root Mean Squared Error (RMSE) of the
3. SYSTEM CONSTRUCTION AND EXPERIMENTS

Figure 3.3: Experiment to estimate the rotor coefficients

curve fitting is calculated, which is 0.3378 for the thrust relation and 0.003742 for the moment relation.

Figure 3.4: Relation between the generated thrust force and the quadrotor’s squared motor speed

The identified parameters are collected in Table 3.1. It is noted that any error in the estimated parameters will be compensated by using a robust control technique.
3.4 6-DOF State Estimation

Position holding is one of the most important factors to achieve the accurate aerial manipulation. The accurate measurements are crucial for implementing the position holding. In this section, the proposed state measurement/estimation scheme, which will be used to find the system states, is described. The state estimation system is presented in details in Fig. 3.6.

On the LLP, there is a data fusion algorithm to find the quadrotor orientation at rate of 1 kHz. In addition, the IMU itself can provide the angular rates directly. Thus, both the angular position and velocity of the quadrotor are available. The position measurements are obtained by using the laser and ultrasonic sensors. Since this process is slow (about 30 Hz) compared to the motion of the quadrotor (quadrotor dynamics have high bandwidth 1kHz), this information is fused with inertial sensor data (body accelerations $\ddot{p}_b$) provided by the IMU at a rate of 1 kHz. The outputs of that filter are finally fed into a position controller. The expensive computation processing of the laser data is run on the onboard computer at approximately 30 Hz, while the fusion filter and the position controller are executed on the

![Figure 3.5: Relation between the generated drag moment and the quadrotor's squared motor speed](image-url)
### Table 3.1: System Parameters

| Par. | Value   | Unit | Par. | Value   | Unit |
|------|---------|------|------|---------|------|
| $m$  | 1       | $kg$ | $l_2$| $85 \times 10^{-3}$ | $m$ |
| $d_q$| $223.5 \times 10^{-3}$ | $m$ | $m_0$| $30 \times 10^{-3}$ | $kg$ |
| $I_x$| $13.215 \times 10^{-3}$ | $N.m.s^2$ | $m_1$| $55 \times 10^{-3}$ | $kg$ |
| $I_y$| $12.522 \times 10^{-3}$ | $N.m.s^2$ | $m_2$| $112 \times 10^{-3}$ | $kg$ |
| $I_z$| $23.527 \times 10^{-3}$ | $N.m.s^2$ | $I_r$| $33.216 \times 10^{-6}$ | $N.m.s^2$ |
| $l_0$| $30 \times 10^{-3}$ | $m$ | $l_1$| $70 \times 10^{-3}$ | $m$ |
| $K_{F_1}$| $1.667 \times 10^{-5}$ | $kg.m.rad^{-2}$ | $K_{F_2}$| $1.285 \times 10^{-5}$ | $kg.m.rad^{-2}$ |
| $K_{F_3}$| $1.711 \times 10^{-5}$ | $kg.m.rad^{-2}$ | $K_{F_4}$| $1.556 \times 10^{-5}$ | $kg.m.rad^{-2}$ |
| $K_{M_1}$| $3.965 \times 10^{-7}$ | $kg.m^2.rad^{-2}$ | $K_{M_2}$| $2.847 \times 10^{-7}$ | $kg.m^2.rad^{-2}$ |
| $K_{M_3}$| $4.404 \times 10^{-7}$ | $kg.m^2.rad^{-2}$ | $K_{M_4}$| $3.170 \times 10^{-7}$ | $kg.m^2.rad^{-2}$ |

HLP at 1 kHz. Note that, this ensures the minimum possible delays, and allows us to handle the fast movements of the quadrotor.

The horizontal position from the laser range finder sensor can be obtained as following.

First, the laser scan is projected orthogonally onto the horizontal $xy$-plane by using the roll and pitch angles of the last state. The projected data is then passed to a scan matcher [74]. The scan matcher uses the yaw angle reading from the IMU as an initial guess for the yaw angle of the quadrotor. The output of the scan matcher includes $x$, $y$, and yaw angle pose components.

The vertical position from the ultrasonic sensor is obtained as following. The ultrasonic sensor is connected to the Arduino board. The sonar signal is transmitted and the echo of it is received. By measuring the time of signal traveling, one can calculate the distance ($s_z$) that is the vertical height of the sensor. However, the calculated distance must be compensated due to the roll and pitch rotations of quadrotor by

$$ z = s_z \cos(\phi) \cos(\theta). \quad (3.1) $$

Therefore, the measured horizontal position and vertical position (at rate of 30 Hz), and the angles and linear accelerations (at rate of 1 kHz) are available now. However, the 6-
DOF measurements rate are required to be high enough to enable a high update rate in the position control loop to match the quadrotor's system dynamics. Thus, these measurements must be fused to achieve this high rate. The filter is designed and decoupled for all the three axes $x$, $y$, $z$ in the world inertial frame. The body-fixed accelerations, $\ddot{p}_b$, are rotated in the global frame by the rotational matrix, $R_b$, based on the attitude angles provided by the LLP, as

$$\ddot{p}_b = R_b \ddot{p}_b^b - [0 \ 0 \ g_r]^T,$$  

where $g_r$ is the acceleration due to the gravity. The filter is based on a Luenberger observer\textsuperscript{75}. In the following, only the filter for the $x$-axis is described and thus it can be
applied for the other axes accordingly. The filter uses the position, $p_{b,x}$, speed, $\dot{p}_{b,x}$, and the acceleration sensor bias, $b_x$, as the states. The acceleration expressed in the world frame is the system input. The measurement is the position sensors’ readings, $p_{s,x}$, from the Laser (for $x$ and $y$) or Sonar (for $z$).

Let us define the state variable as $X_{obs} = [p_{b,x} \dot{p}_{b,x} b_x]^T$, $U_{obs} = \dot{p}_x$, and $Y_{obs} = p_{s,x}$. Then, the observer state equations can be written as

$$\dot{X}_{obs} = A_{obs}\hat{X}_{obs} + L_{obs}(Y - \hat{Y}) + B_{obs}U_{obs},$$

$$Y = C_{obs}\hat{X}_{obs},$$

where $A_{obs} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$, $B_{obs} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$, $C_{obs} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$, and $L_{obs} = \begin{bmatrix} L_{obs_1} \\ L_{obs_2} \\ L_{obs_3} \end{bmatrix}$.

The output of this position filter is the linear position and velocity of the vehicle at update rate of 1 kHz.

Unlike the currently developed state measurement system for aerial manipulation system, with this proposed scheme, one can measure/estimate the system states in different places indoors. In addition, one can utilize this scheme for being used outdoors by using GPS-based position holding for wide operation (quadrotor is far from the target) and using these sensors when the quadrotor is near the objects. If a high performance onboard camera is available, then the data from the laser can be fused with the camera data to obtain more accurate posing indoors and outdoors.

### 3.5 Control Design for the Experiments

In this section, the real time control system is presented. Fig. 3.7 shows the control structure. Since the quadrotor is an under-actuated system, i.e., only 4 independent control inputs are available against the 6 DOF, the position and the yaw angle are usually the controlled variables, while the pitch and roll angles are used as intermediate control inputs for horizontal position control. For position control, a cascade structure is used. As inner loop, the well tested attitude loop provided by the LLP of the FCU is used. The outer loop
3.5 Control Design for the Experiments

Figure 3.7: Block diagram of the control structure; Note that the circle block does the summation function, and the input with minus signal (-) means subtraction while the input with no signal means addition

is the position loop, and is implemented on the HLP based on the concept of PID control with gravity compensation as following

\[
\tau_p = [\tau_x \tau_y \tau_z]^T = K_p(p_{br} - p_b) + K_d(\dot{p}_{br} - \dot{p}_b) + K_i \int_0^t (p_{br} - p_b) dt + G_p, \quad (3.4)
\]

where \(G_p\) is the gravity term of the translational part in the dynamic equation of motion (2.96) and it can be obtained as \(G(1:3)\). \(K_p, K_d,\) and \(K_i\) are PID tuning parameters, and \(p_{br}\) and \(\dot{p}_{br}\) are the reference position and velocity, respectively.

Roll and pitch commands are generated from the outer loop as reference to the inner attitude controller in order to reach and maintain desired \(y\) and \(x\) position, respectively. Thrust commands are generated by the outer controller to control the height by sending desired thrust values. The desired values for the intermediate controller, \(\theta_r\) and \(\phi_r\), are obtained from the output of position controller through the following relation

\[
\begin{bmatrix}
\theta_r \\
\phi_r
\end{bmatrix} = \frac{1}{\tau_p(3)} \begin{bmatrix}
C_\psi & S_\psi \\
S_\psi & -C_\psi
\end{bmatrix} \begin{bmatrix}
\tau_p(1) \\
\tau_p(2)
\end{bmatrix}. \quad (3.5)
\]
This relation is derived from (2.1) based on the small angle approximation of the roll and pitch angles.

Since it is aimed to teleoperate the system, as a first step to autonomous mode, the manipulation task can be achieved by sending velocity commands to the controller. Then, for position holding, one will send a zero velocity command and make the current position as the desired position (at the instant of sending zero command velocity). The velocity commands can be sent using the joystick, where the quadrotor does not need to reach a target position but just follows the desired velocity, allowing a smooth flight. Moreover, this control scheme can be utilized to send position commands to work in autonomous mode.

For the manipulator joints’ control, the built-in position sensor with a PID controller is used. Since the dynamics of the manipulator is simple, one use this built-in controller as a first test. If it does not provide required performance, design another advanced controller will be designed. The simplicity in the manipulator dynamics comes from the lightweight and slow motion of the manipulator parts.

The desired position can be obtained from the command velocity and measured position as in algorithm 1.

\[ \text{Algorithm 1} \quad \text{Algorithm to generate the desired position} \]

1: Initialisation: \( k = 1 \) and \( p_{br}(0) = 0 \)

2: \textbf{while} ( \text{do} )

3: \quad \textbf{if} \ \dot{p}_{br} = 0 \ \textbf{then}

4: \quad \quad \quad p_{br}(k) = p_{br}(k - 1)

5: \quad \quad \textbf{else}

6: \quad \quad \quad p_{br}(k) = p_{b}(k)

7: \quad \quad \textbf{end if}

8: \textbf{end while}
3.6 Simulation Results

This proposed control strategy is applied in simulation MATLAB/SIMULINK program to the model of the considered aerial manipulation robot. In order to make the simulation environment to be quite realistic, a normally distributed measurement noise, with mean of $10^{-3}$ and standard deviation of $5 \times 10^{-3}$, has been added to the measured signals. In addition, one can use the parameter obtained from the experimental identification that is presented previously. The controller parameters are presented in Table 3.2. These parameters are tuned to get a satisfactory response.

The simulation results are presented in Fig. 3.8. In this figure, the target position is assumed to be in the direction of the $y$-axis of quadrotor, and the object (with weight of 50 g) is attached to the end-effector gripper before starting the operation. During the vehicle taking off, by sending a positive velocity command in the $z$ direction until it is reached at suitable height, see Fig. 3.8c, a very small drift in some coordinates occurs, see Fig. 3.8, which is then quickly compensated. At this moment, a zero velocity command is given such that the vehicle is hold in its initial position due to the position holding control. To move towards the target position, in which one have to place the carried object, a positive velocity command is sent in the $y$ direction, see Fig. 3.8b. As soon as the target position is reached, a zero $y$ velocity and a negative $z$ velocity are sent to put the end-effector near to the destination point and then a zero $z$ velocity is given to hold the vehicle at this point, and a command is sent to open the gripper to release the object. After object releasing, a negative velocity is sent in the $y$ direction to go back to the starting point again. Once it is reached at this point, a zero velocities are sent to all coordinates to finish the task. Moreover, a zero velocity command to the manipulator joints is always sent, see Figs. 3.8c and 3.8f. These results show the feasibility and efficiency of the proposed system. These results will be verified experimentally in the next section.

3.7 Experimental Results

In this section, experimental results of the proposed system are presented and discussed. The objective is to transfer an object in a teleoperation mode. For the teleoperation purpose,
it is assumed that one cannot know the accurate coordinates of the target position before operation. Thus, velocity commands will be sent to move the system to that position. The same scenario that done in the simulation study will be implemented except that in some situations a velocity command in the $x$-direction will be sent because it is difficult to determine the target place accurately along $y$-axis as assumed in the simulation test. Then, if the target position is reached, the position holding is activated to place the object in the target position. The target object, to be transfered to another place, has a weight of 50 g which will be attached to the gripper at the start of system operation. Experimental controller parameters are given in Table 3.3. Fig. 3.9 presents photos from the experiment that is divided into six phases; starts from taking off (by given velocity command in the $z$ direction) with the object attached to the gripper, moving to the destination by giving velocity command in the $y$-direction and fine adjustment in the $x$ direction, then reaching at the target position, releasing the object by achieving position holding by giving zero velocity commands, moving towards the home position, and finally, reaching the home position. The real experimental data is recorded in the ground station through the ROS framework and the WiFi connection. Fig. 3.10 presents the experimental results. In this figure, the velocity commands as well as the actual position for the coordinates $x, y, z, \psi, \theta_1, \theta_2$ are presented. The manipulator joints commands are sent using WPS2 controller assuming that the built-in controller is good enough to track accurately the sent commands. To start the mission, see Fig. 3.10c a positive $z$ velocity is sent to take off, then position holding is activated for small period by sending a zero $z$ velocity, after that the platform is commanded to go to a suitable height at lower level by sending negative $z$ velocity until the target height is reached, then one can activate the position holding again. During the vehicle taking off,
Table 3.3: Experimental Controller Parameters

| Parameter   | Value       | Parameter   | Value       |
|-------------|-------------|-------------|-------------|
| $K_{p_x,y}$ | 5           | $K_{d_{x,y}}$ | 12          |
| $K_{i_x,y}$ | 2           | $K_{p_z}$   | 15          |
| $K_{d_z}$   | 15          | $K_{i_z}$   | 4           |
| $L_{obs_{x,y}}$ | [18.01 45.18 0.45]$^T$ | $L_{obs_z}$ | [18.01 45.18 0.45]$^T$ |

A small drift occurs in the $x$, $y$, $z$, and $\psi$ directions which is quickly recovered due to the position holding at the starting point. Fig. 3.10a shows that one have to send velocity commands in $x$ direction, as it is claimed before, to fine tune the vehicle position towards the target place accurately. After taking off, the vehicle moves in $y$ direction towards the target place, see Fig. 3.10b, till reaching the target point at which the position holding is activated. At this point, a command is sent to the gripper to open and release the object. After releasing the object, the vehicle returns back to the starting point, then a negative $z$ velocity is sent until the platform becomes at height suitable to the base station. In Fig. 3.10d, as it is planed, there is no any command in $\psi$ direction, and consequently there is almost no rotation around $z$. Since, the required task is simple, and there is no need to move the manipulator joints (the manipulator needs to be stretched, i.e., $\theta_1$ and $\theta_2 = \pi/2$), zero velocity commands are sent to the manipulator joints, see Figs. 3.10c and 3.10d. However, for sophisticated tasks, one can command the joints to do the 6-DOF motion based on the specified task. These results prove the feasibility and a satisfactory efficiency of the proposed system, state estimation scheme, and the control algorithm. However, they show that the control algorithm for position control needs improvements in the future work by using robust estimation control techniques. Moreover, the sonar can be replaced by more accurate sensor.
Figure 3.8: Simulation results: actual position/command velocity history of the system
3.7 Experimental Results

(a) Taking off, with the object, from home position

(b) Moving to the target position

(c) Reaching at the target position

(d) Releasing the object

(e) Returning back to the home position

(f) Reaching at the home position

**Figure 3.9:** Pictures taken during the experiment
Figure 3.10: Experimental results: measured position/command velocity history of the system
Chapter 4

INVERSE KINEMATICS SOLUTION

All the previous research in the aerial manipulation area do not provide solution to the system inverse kinematics. However, in the motion planning of such system, the task is specified in terms of a desired trajectory for the end-effector. To achieve this task, the system inverse kinematics must be available to get the reference quadrotor/joint space trajectories from desired 6-DOF task space trajectories. Then, a controller is designed to track the generated trajectories from the inverse kinematics. Therefore, in this chapter a novel inverse kinematics analysis is presented to get the reference quadrotor/joint space trajectories from desired 6-DOF task space trajectories, in addition to prove that the end-effector of the proposed robot can achieve arbitrary 6-DOF trajectory tracking. The trajectory inverse kinematics of our proposed non-redundant system is difficult because of the existence of high order nonholonomic constraints. Eight complex algebraic differential equations are needed to be solved. However, obtaining exact solution of these equations is impractical. Thus, an algorithm is developed to get approximate efficient solution of these equations. The effectiveness of the proposed algorithm is validated and enlightened via numerical results.
4. INVERSE KINEMATICS SOLUTION

4.1 Novel 6-DOF Trajectory Inverse Kinematics

A task for the quadrotor manipulation system is usually specified in terms of a desired trajectory for the end-effector pose, \( \chi_{e,r}(t) \) (position, \( p_{e,r}(t) \), and orientation, \( \Phi_{e,r}(t) \)). In this section, six algebraic kinematic equations relating \( \chi_e \) with \( q \) are derived.

For the desired task space trajectories, one can assume:

**Assumption 4.1.** The desired trajectories for the end-effector are bounded.

To find the eight variables of \( q \) from the given six variables of \( \chi_e \), two additional equations are required. These two equations are the nonholonomic constraints (2.99 and 2.100), which are differential equations.

We propose algorithm 2 to get a solution to these eight algebraic/differential equations, see Fig. 4.1.

**Algorithm 2** Algorithm to solve the inverse kinematics problem

1: **Initialization** : Specify the desired 6-DOF trajectory in the task space, \( \chi_{e,r}(t) \). Put \( i = 1, t_i = 0, \) and \( \Delta t = \) very small positive number (1 ms). Put \( \sigma_b(t_0) = \dot{\sigma}_b(t_0) = \ddot{\sigma}_b(t_0) = [0 \ 0]^T \)

2: while \( t \leq t_f \) do

3: \( \triangleright t_f \) is the final time

3. \( \triangleright \) At time \( t_i \), obtain the values of \( \chi_{e,r}(t_i) \) and put \( \sigma_b(t_i) = \sigma_b(t_{i-1}), \dot{\sigma}_b(t_i) = \dot{\sigma}_b(t_{i-1}), \)

4: and \( \ddot{\sigma}_b(t_i) = \ddot{\sigma}_b(t_{i-1}) \).

4: Use the six inverse kinematics algebraic equations and get the unknowns \( \zeta_r(t_i), \dot{\zeta}_r(t_i), \)

5: \( \ddot{\zeta}_r(t_i) \) which with \( \sigma_b(t_i), \dot{\sigma}_b(t_i), \) and \( \ddot{\sigma}_b(t_i) \) constitute \( q(t_i), \dot{q}(t_i), \) and \( \ddot{q}(t_i) \).

5: one can substitute by \( q(t_i), \dot{q}(t_i), \ddot{q}(t_i) \) values in the right hand side of the nonholonomic constraints, and obtain the new \( \sigma_b(t_i) \) from left hand side and by numerical differentiation, one can get the new \( \dot{\sigma}_b(t_i) \) and \( \ddot{\sigma}_b(t_i) \).

6: Put \( i = i + 1, t_i = t_{i-1} + \Delta t, \) and repeat to obtain \( \zeta_r(t_i) \).

7: end while

The validation of this algorithm will be checked at the end of this section. The six algebraic inverse kinematic equations are derived next.

Define the general form for the rotation matrix \( R_e \) as a function of the end-effector
4.1 Novel 6-DOF Trajectory Inverse Kinematics

Figure 4.1: Block diagram of the inverse kinematic algorithm

variables $\chi_e$, as

$$R_e = R_b R_e^b = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix},$$

(4.1)

where the elements of $R_e$ are given as

$$R_{11} = C_{\theta_2} (C_{\theta_1} (C_{\phi} S_{\psi} - C_{\psi} S_{\phi} S_{\theta}) - S_{\theta_1} (S_{\psi} S_{\phi} + C_{\psi} C_{\phi} S_{\theta})) + C_{\psi} C_{\theta_2},$$

(4.2)

$$R_{12} = C_{\psi} C_{\theta} C_{\theta_2} - S_{\theta_2} (C_{\theta_1} (C_{\phi} S_{\psi} - C_{\psi} S_{\phi} S_{\theta}) - S_{\theta_1} (S_{\psi} S_{\phi} + C_{\psi} C_{\phi} S_{\theta})), $$

(4.3)

$$R_{13} = C_{\theta_1} (S_{\psi} S_{\phi} + C_{\psi} C_{\phi} S_{\theta}) + S_{\theta_1} (C_{\phi} S_{\psi} - C_{\psi} S_{\phi} S_{\theta}),$$

(4.4)

$$R_{21} = C_{\theta} S_{\psi} S_{\theta_2} - C_{\theta_2} (C_{\theta_1} (C_{\psi} C_{\phi} + S_{\psi} S_{\phi} S_{\theta}) - S_{\theta_1} (S_{psi} C_{psi} - C_{psi} S_{phi} S_{theta})), $$

(4.5)

$$R_{22} = S_{\theta_2} (C_{\theta_1} (C_{\psi} C_{\phi} + S_{\psi} S_{\phi} S_{\theta}) - S_{\theta_1} (C_{\psi} S_{\phi} - C_{\psi} S_{\phi} S_{\theta})) + C_{\theta} C_{\theta_2} S_{\psi}, $$

(4.6)

$$R_{23} = -C_{\theta_1} (C_{\phi} S_{\psi} - C_{\psi} S_{\phi} S_{\theta}) - S_{\theta_1} (C_{\psi} C_{\phi} + S_{\psi} S_{\phi} S_{\theta}),$$

(4.7)
4. INVERSE KINEMATICS SOLUTION

\[ R_{31} = -S_\theta S_{\theta_2} - C_{\theta_2} (C_\phi C_\theta S_{\theta_1} + C_\theta C_{\theta_1} S_\phi), \]  
\[ (4.8) \]

\[ R_{32} = S_{\theta_2} (C_\phi C_\theta S_{\theta_1} + C_\theta C_{\theta_1} S_\phi) - C_{\theta_2} S_\theta, \]
\[ (4.9) \]

\[ R_{33} = C_\phi C_\theta C_{\theta_1} - C_\theta S_\phi S_{\theta_1}. \]
\[ (4.10) \]

According to the structure of \( R_e \) from \((2.8)\), the inverse orientation is carried out first followed by inverse position.

Let us define the end-effector orientation, \( R_{e, \Phi_e} \), as function of the Euler angles, \( \Phi_e = [\psi_e, \theta_e, \phi_e]^T \), as

\[
R_e = \begin{bmatrix}
C_{\psi_e} C_{\theta_e} & C_{\psi_e} S_{\phi_e} S_{\theta_e} - C_{\phi_e} S_{\psi_e} & S_{\psi_e} S_{\phi_e} + C_{\psi_e} C_{\phi_e} S_{\theta_e} \\
C_{\psi_e} S_{\theta_e} & C_{\psi_e} C_{\phi_e} + S_{\psi_e} S_{\phi_e} S_{\theta_e} & C_{\phi_e} S_{\psi_e} S_{\theta_e} - C_{\psi_e} C_{\phi_e} S_{\theta_e} \\
-\theta_{\phi_e} & C_{\theta_e} S_{\phi_e} & C_{\phi_e} C_{\theta_e}
\end{bmatrix} = \begin{bmatrix}
r_{11} & r_{12} & r_{13} \\
r_{21} & r_{22} & r_{23} \\
r_{31} & r_{32} & r_{33}
\end{bmatrix}, \]
\[ (4.11) \]

Equating \((4.1)\) and \((4.11)\) provides the inverse orientation as following. The inverse orientation has three cases based on the value of \( \theta_1 \) which can be calculated from the element \( r_{33} \) as

\[ C_\phi C_\theta C_{\theta_1} - S_\theta C_\phi S_{\theta_1} = r_{33}. \]
\[ (4.12) \]

By rearranging \((4.12)\) and solving the resulting equation for \( \theta_1 \), then

\[ \theta_1 = 2 \arctan2(-2b_1 \pm \sqrt{(2b_1)^2 - 4(-a_1 - r_{33})(a_1 - r_{33})}, 2(-a_1 - r_{33})) \],
\[ (4.13) \]

where \( a_1 = C_\phi C_\theta \) and \( b_1 = -S_\theta C_{\theta_1} \).

Based on the value of \( \theta_1 \), three cases for the rotational inverse kinematics solution exist.

**CASE 1:** \( \theta_1 \neq 0 \) and \( \theta_1 \neq \pi \)
4.1 Novel 6-DOF Trajectory Inverse Kinematics

Inspecting $r_{13}$ and $r_{23}$, one can find value of $\psi$ as

$$r_{13} = a_2 S_\psi + b_2 C_\psi,$$

(4.14)

$$r_{23} = c_2 S_\psi + d_2 C_\psi,$$

(4.15)

where $a_2 = C_\phi S_{\theta_1} + S_\phi C_{\theta_1}$, $b_2 = C_\phi S_\theta C_{\theta_1} - S_\phi S_\theta S_{\theta_1}$, $c_2 = b_2$, and $d_2 = -a_2$.

Solving (4.14) and (4.15), will give

$$S_\psi = \frac{r_{23} b_2 - r_{13} d_2}{-d_2 a_2 + b_2 c_2},$$

(4.16)

$$C_\psi = \frac{r_{23} a_2 - r_{13} c_2}{d_2 a_2 - b_2 c_2},$$

(4.17)

$$\psi = \text{atan2}(S_\psi, C_\psi).$$

(4.18)

By inspecting $r_{32}$ and $r_{31}$, $\theta_2$ can be found as

$$r_{32} = a_3 S_{\theta_2} + b_3 C_{\theta_2},$$

(4.19)

$$r_{31} = c_3 S_{\theta_2} + d_3 C_{\theta_2},$$

(4.20)

where $a_3 = C_\phi C_\theta S_{\theta_1} + C_\theta S_\phi C_{\theta_1}$, $b_3 = -S_\theta$, $c_3 = b_3$, and $d_3 = -a_3$.

Solving (4.19) and (4.20), then

$$S_{\theta_2} = \frac{r_{31} b_3 - r_{32} d_3}{-d_3 a_3 + b_3 c_3},$$

(4.21)

$$C_{\theta_2} = \frac{r_{31} a_3 - r_{32} c_3}{d_3 a_3 - b_3 c_3},$$

(4.22)
\[ \theta_2 = \text{atan2}(S_{\theta_2}, C_{\theta_2}). \] (4.23)

**CASE 2: \( \theta_1 = 0 \)**

If \( \theta_1 = 0 \), then the sum \( \theta_2 + \psi \) can be determined. One can assume any value for \( \psi \) and get \( \theta_2 \). Therefore, there are infinity of solutions. By putting \( \psi = 0 \), the value of \( \theta_2 \) can be determined as follows:

Inspecting \( r_{11} \) and \( r_{12} \), \( \theta_2 \) can be found as

\[ r_{11} = a_4 S_{\theta_2} + b_4 C_{\theta_2}, \] (4.24)

\[ r_{12} = c_4 S_{\theta_2} + d_4 C_{\theta_2}, \] (4.25)

where \( a_4 = C_{\theta}, \; b_4 = -S_{\phi} S_{\theta}, \; c_4 = -b_4, \; \text{and} \; d_4 = a_4. \)

Solving (4.24) and (4.25), then

\[ S_{\theta_2} = \frac{r_{12} b_4 - r_{11} d_4}{-d_4 a_4 + b_4 c_4}, \] (4.26)

\[ C_{\theta_2} = \frac{r_{12} a_4 - r_{11} c_4}{d_4 a_4 - b_4 c_4}, \] (4.27)

\[ \theta_2 = \text{atan2}(S_{\theta_2}, C_{\theta_2}). \] (4.28)

**CASE 3: \( \theta_1 = \pi \)**

Since \( S_{\theta_1} = 0 \), this is similar to case 2. However, \( \theta_2 - \psi \) can be determined and by choosing \( \psi = 0 \), an expression for \( \theta_2 \) can be determined as follows:

Inspecting \( r_{11} \) and \( r_{12} \), \( \theta_2 \) can be found as

\[ r_{11} = a_5 S_{\theta_2} + b_5 C_{\theta_2}, \] (4.29)

\[ r_{12} = c_5 S_{\theta_2} + d_5 C_{\theta_2}, \] (4.30)
where \( a_5 = a_4, b_5 = -b_4, c_5 = -b_5, \) and \( d_5 = a_5. \)

Solving (4.29) and (4.30), then

\[
S_{\theta_2} = \frac{r_{12}b_5 - r_{11}d_5}{-d_5a_5 + b_5c_5},
\]

(4.31)

\[
C_{\theta_2} = \frac{r_{12}a_5 - r_{11}c_5}{d_5a_5 - b_5c_5},
\]

(4.32)

\[
\theta_2 = \text{atan2}(S_{\theta_2}, C_{\theta_2}).
\]

(4.33)

As shown above, there are two possible solutions for the rotational inverse kinematics problem provided that one puts \( \psi = 0 \) in cases 2 and 3. When one starts applying the above described algorithm in real time, one can select the solution of \( \theta_1 \) which coincide with the given configuration of the quadrotor manipulation system. After that, one can continue with the one of the two solutions which produces this \( \theta_1 \).

Finally, the inverse position is determined from (2.7) as

\[
p_b = p_e - R_b p_{eb}^b.
\]

(4.34)

There is one solution for this inverse position problem.

### 4.2 Verification

To verify the proposed solution of the inverse kinematics, the desired task space trajectories are chosen to make a circular helix in position and quintic polynomial [71] for the orientation.

After obtaining the joint space variables from the proposed algorithm, as shown in Fig. 4.1, the forward kinematics is applied to find the actual task space trajectory.

The comparison between the actual and the desired task space trajectories is shown in Figs. 4.2 and 4.3 from which one can recognize that the actual and desired trajectories are coincided. In Figs. 4.4 and 4.5 another case study is investigated to show the capability
of the proposed algorithm to deal with higher speed desired trajectories. In this study, the
desired trajectories are chosen to be faster than that in Fig. 4.3 by ten times. Note that the
maximum speed for the trajectories are 5 m/s for the translation motion and 6 rad/s for the
rotational motion. These figures show that the actual and desired trajectories are coincided
under both slower or faster trajectories, which ensures the validity of the proposed inverse
kinematic algorithm, and proves that the proposed system has the ability to track arbitrary
6-DOF task space trajectory.

The obtained accurate inverse kinematics solution enables one to design the controller in
the quadrotor/joint space to track the desired trajectories in the task space. This control
design is the subject of the next chapter where certain control objectives are achieved for
such complex dynamic system.
Figure 4.2: The actual response of the end-effector pose in the case of the slower trajectories: a) \( x_e \), b) \( y_e \), c) \( z_e \), d) \( \phi_e \), e) \( \theta_e \), and f) \( \psi_e \).
Figure 4.3: End-effector 3D trajectory in the case of the slower trajectories
4.2 Verification

Figure 4.4: The actual response of the end-effector pose in the case of the faster trajectories: a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$. 67
Figure 4.5: End-effector 3D trajectory in the case of the faster trajectories
Chapter 5

CONTROL SYSTEM DEVELOPMENT

This chapter proposes motion control systems to address and solve the control issues of the quadrotor manipulation system. These issues include; robust position control, robust optimal position control, and robust force estimation and impedance control.

The control of the quadrotor manipulation system is quite challenging because it is naturally unstable, and has strong nonlinearities. In addition, it is very susceptible to parameters variations due to adding/releasing the payload and the actuators’ losses. Moreover, it is subjected to external disturbances like wind. Furthermore, its measurements are very noisy due to the thrusters that rotate at very high speed. Also, it has fast dynamics. Thus, there is a need to design and implement an efficient robust control technique to address these issues, which is presented in section 5.2.

Furthermore, the actuators (thrusters and manipulator motors) have constraints, which may degrade the performance if it is not considered during controller design. Moreover, we have to apply an optimal control effort to reduce the required power of the thrusters such that we can save power and thus increase the mission time and/or increase the payload. Therefore, in section 5.3, the Model Predictive Control (MPC) technique will be utilized based on the robust linearization capabilities of the DOB to address this issue.

Finally, in addition to the accurate positioning of the end-effector, the manipulation tasks require estimating (applying) certain force at the end-effector as well. Thus, in section 5.4.
5. CONTROL SYSTEM DEVELOPMENT

a robust force estimation and impedance control scheme is proposed. The robustness will be achieved based on the DOB technique. Then, a linear impedance controller will be designed in the joint space. The contact force is be identified based on robust linearization capabilities of DOB and a Fast Tracking Recursive Least Squares (FTRLS) algorithm, which takes into consideration all the system dynamics with no need for the environment contact indication.

5.1 Control Objectives

We target the design of the control system in order to satisfy the following objectives:

Control Objective 1. (Robust Stability) The considered robotic system is stable and robust in the presence of the external disturbances, uncertainties, and noises.

Control Objective 2. (Trajectory Tracking) 6-DOF task space trajectory tracking is achieved.

Control Objective 3. (Optimal Performance) Minimization of the controller efforts, and constraints them to be within the allowable limits of the system actuators (i.e., thrust forces and motor’s torques).

Control Objective 4. (Force Estimation) The end-effector contact force has to be estimated fast.

Control Objective 5. (6-DOF Impedance Control) In the presence of the contact force, desired impedance at the end-effector is achieved.

5.2 Robust Position Control

In this section, a modified version of the well-known Disturbance Observer-based robust control is proposed in order to achieve the robust 6-DOF task space trajectory tracking against the system nonlinearities, couplings, uncertainties, disturbances, and measurement noise.

To achieve control objectives 1 and 2 a control technique as shown in Fig. 5.1 is proposed. In this strategy, the system nonlinearities, uncertainties, external disturbances, $\tau^{\text{dis}}$, are treated as extended disturbances which will be estimated, $\hat{\tau}^{\text{dis}}$, and canceled by the DOB in the inner loop. The system can be now considered as linear SISO plants, and thus,
5.2 Robust Position Control

simple PD controller is used in the external loop to achieved the target performance for the system by producing \( \tau^{\text{des}} \). Another loop is presented to find the desired quadrotor/joint space trajectories from the target 6-DOF task space based on the inverse kinematic analysis that is presented in chapter \[4\]. Hence, the controller can be designed in the quadrotor/joint space.

\[5.2.1 \text{ DOb-based Controller} \]

Disturbance Observer (DOb)-based controller is one of the most popular methods in the field of robust motion control due to its simplicity and computational efficiency. In DOb-based robust motion control systems, internal and external disturbances are observed by the DOb, and the robustness is simply achieved by feeding-back the estimated disturbances in an inner-loop. Another controller is designed in an outer loop so that the performance goals are achieved without considering internal and external disturbances.

A block diagram of the conventional DOb inner loop is shown in Fig. \[5.2\]. In this figure, \( M_n \in \mathbb{R}^{8 \times 8} \) is the system nominal inertia matrix, \( \tau^{\text{des}} \) is the desired input vector, \( P = \text{diag}(g_1, \ldots, g_i, \ldots, g_8) \) with \( g_i \) is the bandwidth of the low pass filter of the \( i^{\text{th}} \) variable of \( q \), \( Q(s) = \text{diag}(\frac{g_1}{s+g_1}, \ldots, \frac{g_i}{s+g_i}, \ldots, \frac{g_8}{s+g_8}) \in \mathbb{R}^{8 \times 8} \) is the matrix of the low pass filter of DOb. The DOb requires velocity measurements. \( \tau^{\text{dis}} \) represents the system extended disturbances including the Coriolis, centrifugal and gravitational terms, and external disturbances, and \( \hat{\tau}^{\text{dis}} \) represents the system estimated disturbances. \( K \) is the external controller that is used
to achieve the required control performance.

If one can apply the concept of disturbance observer to the proposed system, then the independent coordinate control is possible without considering the coupling effect of the other coordinates. The coupling terms such as centripetal and Coriolis and gravity terms are considered as disturbance and compensated by feeding back the estimated disturbance torque.

**Figure 5.2:** Block diagram of the conventional DOb internal loop

As we noted in this structure, the DOb utilizes the estimated velocity measurement to estimate the system disturbances.

However, it is well known that the linear accelerations and angular rates of the quadrotor can be measured directly from the IMU. In addition, the angular velocities of the joints can be measured via an encoder. Therefore, a block diagram of the modified DOb-based controller is proposed as in Fig. 5.3 for our robotic system. In this modified DOb-based control, two different DOb loops are used. One is based on the measured accelerations, while the other is based on the measured velocities.

In this figure, $M_n = \begin{bmatrix} M_{na} & O_{3\times5} \\ O_{5\times3} & M_{nv} \end{bmatrix}$ is the system nominal inertia matrix with $M_{na} \in \mathbb{R}^{3\times3}$ represents the nominal inertia for the translational coordinates, $p_b$, while $M_{nv} \in \mathbb{R}^{5\times5}$ represents the nominal inertia for the rotational coordinates, $\Phi_b$ and $\Theta$. $Q_a(s) = diag(\frac{g_1}{s+g_1}, \ldots, \frac{g_3}{s+g_3})$, and $Q_v(s) = diag(\frac{g_4}{s+g_4}, \ldots, \frac{g_8}{s+g_8})$ are the matrices of the low pass filter of DOb for the translational and rotational coordinates, respectively. $P_a = diag([g_1 g_2 g_3])$
and \( P_v = \text{diag}([g_4 \ldots g_i \ldots g_8]) \) vectors of bandwidths of the DOB filter for translational and rotational coordinate, respectively. \( \tau = [\tau_a^T \tau_v^T]^T \) represents the robot’s generalized forces for the translational and rotational coordinates. \( \hat{\tau}^{\text{dis}} = [\hat{\tau}_a^{\text{dis}} \hat{\tau}_v^{\text{dis}}]^T \) represents the system estimated disturbances.

The system disturbance, \( \tau^{\text{dis}} \), can be calculated as

\[
\tau^{\text{dis}} = (M(q) - M_n)\ddot{q} + \tau^d,
\]

\[
\tau^d = C(q, \dot{q})\dot{q} + G(q) + d_{ex}.
\] (5.1)

The control input, \( \tau \), in Fig. 5.3 can be calculated as

\[
\tau = M_n\ddot{q}^{\text{des}} + \hat{\tau}^{\text{dis}},
\] (5.2)

where

\[
\hat{\tau}^{\text{dis}} = Q(\tau - M_n\ddot{q}).
\] (5.3)
Thus, the control input, $\tau$, can be calculated as

$$
\tau = M_n \ddot{q}^{des} + M_n P e_v,
$$
(5.4)

where $e_v = \dot{q}^{des} - \dot{q}$, and $\ddot{q}^{des}$ that is the output from the outer loop controller which will be designed later.

By the virtue that the DOb works ideal, then

$$
\dot{\tau}^{dis} = \tau^{dis}.
$$
(5.5)

Thus, the dynamics from the DOb loop input $\tau^{des}$ to the output of the robot manipulator are given by

$$
M_n \ddot{q}^{des} = \tau^{des}.
$$
(5.6)

Since $M_n$ is chosen to be a diagonal matrix, the system can be treated as multi-decoupled linear SISO systems as

$$
M_{ni} \ddot{q}_i^{des} = \tau_i^{des},
$$
(5.7)

or simply in the acceleration space as

$$
\ddot{q}_i = \ddot{q}_i^{des}.
$$
(5.8)

All that remains in the design of the DOb based controller is the design of the tracking controller, $K(s)$, in the DOb outer loop. A PD-based tracking controller for the system of (5.8) is chosen as

$$
\ddot{q}^{des} = \ddot{q}_r + K_P (q_r - q) + K_D (\dot{q}_r - \dot{q}),
$$
(5.9)

where $K_P$ and $K_D \in \mathbb{R}^{8\times8}$ are the proportional and derivative gains of the PD controllers, receptively. $q_r$, $\dot{q}_r$, and $\ddot{q}_r$ are the references for linear/angular positions, velocities, and accelerations, respectively. By proper selection of $K_P$ and $K_D$, one can tune the response of the the error dynamics as necessary.

To ensure that the system stability under this control can be guaranteed, we do the
following analysis.

The control law in (5.4) leads to the following error dynamics

\[ M(q)\dot{e}_v + C(q, \dot{q})e_v + K_v e_v = \delta, \]

\[ K_v = PM_n, \]  

(5.10)

and

\[ \dot{e} + K_D e + K_P \int e \, dt = e_v, \]

(5.11)

where

\[ \delta = \Delta M(q)\ddot{q}^{\text{des}} + C(q, \dot{q})\dot{q}^{\text{des}} + G(q) + d_{ex}, \]

(5.12)

and \( \Delta M(q) = M(q) - M_n. \)

Equation (5.10) represents the error dynamics of the DOb loop.

Let us assume the following Lyapunov function

\[ V = \frac{1}{2} e_v^T M(q)e_v. \]

(5.13)

The time derivative of this function is

\[ \dot{V} = e_v^T M(q)\dot{e}_v + \frac{1}{2} e_v^T M(q)e_v. \]

(5.14)

Substituting from (5.10), then (5.14) becomes

\[ \dot{V} = e_v^T \delta - e_v^T K_v e_v + \frac{1}{2} e_v^T (\dot{M}(q) - 2C(q, \dot{q}))e_v. \]

(5.15)

The dynamic equation of motion (2.96) posses several well known properties [71,76].

Property 1.

\[ \lambda_{\min} \| \nu \|^2 \leq \nu^T M(q)\nu \leq \lambda_{\max} \| \nu \|^2 \]

(5.16)

Property 2.

\[ \nu^T (\dot{M}(q) - 2C(q, \dot{q}))\nu = 0 \]

(5.17)

where \( \dot{M}(q) - 2C(q, \dot{q}) \) is a skew-symmetric matrix, \( \nu \in \mathbb{R}^8 \) represents a 8-dimensional
vector, and $\lambda_{\text{min}}$ and $\lambda_{\text{max}}$ are positive real constants that are the minimum and maximum eigen values of the matrix $M(q)$, respectively.

Substituting from (5.17) into (5.15), then (5.15) will be

$$\dot{V} = e_v^T \delta - e_v^T K_v e_v. \quad (5.18)$$

From the robot dynamic property (5.16), one can find:

$$\dot{V} \leq -\gamma V + \sqrt{\frac{2V}{\lambda_{\text{min}}}} |\delta|, \quad (5.19)$$

where $\gamma = \frac{2K_v}{\lambda_{\text{max}}}$. 

From the analysis presented in [77,78], we complete the analysis as following. Dividing (5.19) by $V^{0.5} \neq 0$ will give

$$\frac{d}{dt}(V^{0.5}) + 0.5\gamma V^{0.5} \leq \sqrt{\frac{2}{\lambda_{\text{min}}}} |\delta|, \quad (5.20)$$

By multiplying (5.20) by $e^{-0.5\gamma t}$ and integrating, then we get

$$V^{0.5} \leq e^{-0.5\gamma t} V^{0.5}(0, e_v(0)) + V_c, \quad (5.21)$$

with

$$V_c = \frac{1}{\sqrt{2\lambda_{\text{min}}}} \int_0^t e^{-0.5\gamma(t-\tau)} |\delta(\tau)| d\tau. \quad (5.22)$$

One can rewrite $V_c$ as

$$V_c = \frac{1}{\sqrt{2\lambda_{\text{min}}}} (e^{-0.5\gamma t} * |\delta(t)|). \quad (5.23)$$

Taking the $\|\|_p$ gives

$$\|V_c\|_p = \frac{1}{\sqrt{2\lambda_{\text{min}}}} \|e^{-0.5\gamma t} * \delta(t)\|_p. \quad (5.24)$$

Lemma 1. ([79]) Consider $\mathcal{Y}_1 \in L_1$, $\mathcal{Y}_2 \in L_p$, then

$$\|\mathcal{Y}_1 * \mathcal{Y}_2\|_p \leq \|\mathcal{Y}_1\|_1 \|\mathcal{Y}_2\|_p. \quad (5.25)$$
5.2 Robust Position Control

From Lemma 1, then (5.24) is
\[ \| V_c \|_p = \frac{1}{\sqrt{2 \lambda_{\min}}} \| e^{-0.5 \gamma t} \|_1 \| \delta(t) \|_p, \]  
but \( \| e^{-0.5 \gamma t} \|_1 = 0.5 \gamma \), so
\[ \| V_c \|_p = 0.5 \gamma \frac{1}{\sqrt{2 \lambda_{\min}}} \| \delta(t) \|_p. \]  
Thus, (5.21) is
\[ \| V^{0.5} \|_p \leq \| e^{-0.5 \gamma t} V^{0.5}(0, e_v(0)) \|_p + 0.5 \gamma \frac{1}{\sqrt{2 \lambda_{\min}}} \| \delta(t) \|_p. \]  
This expression (5.28) can be simplified to be
\[ \| e_v \|_p \leq \frac{1}{\gamma} + \sqrt{\frac{2}{\lambda_{\min}}} \frac{1}{p \gamma} \sqrt{V(0, e_v(0))} \| \delta \|_p. \]  
Thus, the error dynamics is \( L_p \) input/output stable with respect to the pair \( (\delta, e_v) \) for all \( p \in [1, \infty) \).

Lemma 2. (80) Let
\[ e = H_{PD}(s) e_v, \]  
where \( H_{PD}(s) \) is an strictly proper and exponentially stable transfer function. Then \( e_v \in L_p \) implies that \( e \in L_p \) and \( \dot{e} \in L_p \).

Form (5.11), the transfer function between \( e \) and \( e_v \) is proper exponentially stable as long as the parameters \( K_P \) and \( K_D \) are positive definite matrices. Therefore, Lemma 2 implies that both \( e \) and \( \dot{e} \in L_p \).

Remark 1. Equation (5.19) indicates that the convergence rate increases proportionally with \( K_v \). However, we have to take into considerations the constraints on \( g_i \) due to the sampling time.

Fig. 5.4 presents the detailed block diagram of the proposed motion control system based on the inverse kinematic analysis and on the DOB quadrotor/joint space-based control. The desired trajectories for the end-effector’s position and orientation \( \chi_{e,r}(t) \), \( p_{e,r}(t) \) and \( \Phi_{e,r}(t) \), are fed to the inverse kinematic algorithm together with \( \sigma_{b,r}(t) \) generated from the
nonholonomic constraints such that the desired trajectories in vehicle/joint space, \( \zeta_r(t) \), are obtained. After that, the controller block receives the desired trajectories and the feedback signals from the system and provides the control signal, \( \tau = Bu \).

Since the linear position, joints angles, and the yaw angle are usually the controlled variables while pitch and roll angles are used as intermediate control inputs for horizontal position control, the proposed control system consists from two DOb-based controllers; one for \( \zeta \) (with \( K_\zeta(s) \), \( M_{n_\zeta} \), \( P_{v_1} \), \( Q_{v_1} \), and \( Q_a \)) and the other for \( \sigma_b \) (with \( K_\sigma(s) \), \( M_{nv_2} \), \( P_{v_2} \), \( Q_{v_2} \)).

The desired values \( \sigma_{b,r} \) for the intermediate controller are obtained from the output of position controller, \( \tau_\zeta \), through the relation

\[
\sigma_{b,r} = \frac{1}{\tau_\zeta(3)} \begin{bmatrix} C_\psi & S_\psi \\ S_\psi & -C_\psi \end{bmatrix} \begin{bmatrix} \tau_\zeta(1) \\ \tau_\zeta(2) \end{bmatrix},
\]

which can be derived from (2.1) based on small angle approximation.

The desired value of the pitching and rolling, \( \sigma_{b,r} \), is then fed back to the inverse kinematics algorithm.
5.2 Robust Position Control

The output of the two controllers, $\tau_\zeta$ and $\tau_\sigma$, are mixed to generate the final control vector $\tau$ which is converted to the forces/torques applied to quadrotor/manipulator by

$$ u = B_6^{-1} \begin{bmatrix} \tau_\zeta(3, 4) \\ \tau_\sigma \\ \tau_\zeta(5, 6) \end{bmatrix}, $$

(5.32)

where $B_6 \in \mathbb{R}^{6 \times 6}$ is part of $B$ matrix and it is given by $B_6 = B(3 : 8, 1 : 6)$.

Remark 2. It is noted from the quadrotor dynamics [34, 81, 82] that the response of the intermediate, $\sigma$, controller must be faster than that of the position control loop. However, as stated in the above analysis, the convergence rate of errors dynamics ($e_v$ and $e$) is directly proportional to the $K_v$, $K_P$, $K_D$. Therefore, we have to tune $K_v$, $K_P$, and $K_D$, such that the response speed of $\sigma_b$ is much faster than that of $\zeta$.

5.2.2 Simulation Study

In this section, the previously proposed control strategy is simulated in MATLAB/SIMULINK program for the considered aerial manipulation system.

5.2.2.1 Simulation Environment

In order to make the simulation quite realistic, the following setup and assumptions have been made:

- The model has been identified on the basis of real data through experimental tests. The identified parameters are given in Table 3.1.
- Linear and angular position and velocity of the quadrotor are available at rate of 1 kHz.
- The position and velocity of the manipulator joints are available at rate of 1 kHz.
- A normally distributed measurement noise, with mean of $10^{-3}$ and standard deviation of $5 \times 10^{-3}$, has been added to the measured signals.
- The controller outputs are computed at a rate of 1 kHz.
5. CONTROL SYSTEM DEVELOPMENT

- In order to test the robustness to the model uncertainties, a step change is introduced, at instant 15 s, in the control matrix, \( N \), which can simulate sudden actuators’ losses, whose elements are assumed to be equal to 0.9 times their true values (i.e., 10% error).

- The end-effector has to pick a payload of value 200 g at instant 20 s and release it at 40 s.

5.2.2.2 Results and Discussion

Tables 5.1 presents the controller parameter for the proposed control technique. The reference trajectories for the end-effector are chosen such that the end-effector moves on a circular helix, while its orientation follows quintic polynomial trajectories.

The simulation results are presented in Figs. 5.6, 5.5 and 5.7. From these figures, it is possible to recognize that there are small oscillations after starting the system operation due to the time needed to estimate the nonlinearities and disturbances. This time is about 4 s. After this period, the controller can quickly recover this error and provides good tracking of the desired trajectories. Moreover, it is clear that the capability of the controller to recover, with fast response (about 0.05 s), the tracking error due to the presence of the payload and the uncertainty in system parameters. Moreover, Fig. 5.5 indicates that the quadrotor/joint space trajectories are within the allowable limits of the joints (270°). Furthermore, Fig. 5.8 indicates that the required actuators efforts, which are the required thrust force from each rotor and the torque from each manipulator’s motor, \( u \), are in the allowable limit. The maximum thrust force for each rotor is 8 N as obtained from the identification process, while the allowable input torque for the motor of joint 1 is 0.7 N.m and 0.4 N.m for that of joint 2 as stated in the motors’ data sheet. Therefore, one can contend that the control objectives 1 and 2 are achieved by using this motion control scheme.
5.2 Robust Position Control

Table 5.1: PD-DOb parameters

| Par. | Value                  | Par. | Value                  |
|------|------------------------|------|------------------------|
| $M_m$ | $\text{diag}\{1.2, 1.2, 1.2, 0.02, 0.004, 0.0009\}$ | $M_n$ | $\text{diag}\{0.02, 0.02\}$ |
| $K_{Pz}$ | $\text{diag}\{3, 3, 10, 3, 3, 3\}$ | $K_{Pz}$ | $\text{diag}\{10, 10\}$ |
| $K_{Dz}$ | $\text{diag}\{1.5, 1.5, 5, 3, 3, 3\}$ | $K_{Dz}$ | $\text{diag}\{7, 7\}$ |
| $P$ | $[10 \ 10 \ 10 \ 100 \ 100 \ 100 \ 100 \ 100]^T$ |
Figure 5.5: PD-DOb results: The actual response in the quadrotor/joint space; a) $x$, b) $y$, c) $z$, d) $\psi$, e) $\theta_1$, and f) $\theta_2$
5.2 Robust Position Control

Figure 5.6: PD-DOb results: The actual response of the end-effector pose; a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$. The inner box of each figure represents zooming on that figure from instant 0 s to instant 5 s.
Figure 5.7: PD-DOb results: 3D trajectory of end-effector with fixed orientation (The marker represents the end-effector orientation; Green, Blue, and Red for x-, y, and z-axis)
Figure 5.8: PD-DOb results: The required control efforts
5.3 Robust Optimal Position Control

In this section, based on the robust linearization capabilities of the DOb that presented in the previous section, the continuous MPC [83] technique will be used to achieve an optimal performance for the aerial manipulator, in addition to keep the actuators’s inputs within their limits.

To achieve the control objectives 1 - 3, a control technique as shown in Fig. 5.9 is proposed. In this strategy, the system nonlinearities, uncertainties, external disturbances, $\tau_{\text{dis}}$, are treated as extended disturbances which will be estimated, as $\hat{\tau}_{\text{dis}}$, and canceled by the DOb in the inner loop. The system can be now considered as linear SISO subsystems, and thus, the MPC is used in the external loop to achieve the target performance for the system by producing $\tau_{\text{des}}$. Another loop, inverse kinematics, is presented to find the desired quadrotor/joint space trajectories from the target 6-DOF task space trajectory.

5.3.1 Linearized Model

By virtue of the rejection of the nonlinearities, disturbances, uncertainties, and noise by the usage of DOb internal loop as presented in section 5.2, the MPC controller has not to consider uncertainties and can be designed based on the nominal 8 SISO decoupled models of the system with satisfying the actuators’ constraints, $u_{\text{min}} \leq u \leq u_{\text{max}}$. However, in determining $u_{\text{max}}$ and $u_{\text{min}}$, the contribution of $\hat{\tau}_{\text{dis}}$ should be considered.
5.3 Robust Optimal Position Control

The state space SISO model of the system of (5.8) is

\[
\dot{x}_p(t) = A_p x_p(t) + B_p u_{mpc}(t),
q_i(t) = C_p x_p(t),
\]

(5.33)

where \( x_p = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q_i \\ \dot{q}_i \end{bmatrix}, \ A_p = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \ B_p = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \ C_p = \begin{bmatrix} 1 & 0 \end{bmatrix}, \) and \( u_{mpc}(t) \) is the output of the external controller, MPC, which will be \( \dot{q}_i^{des} \).

5.3.2 Continuous time MPC

The MPC can be designed in continuous time or discrete time models. One of the key advantages in using continuous time predictive control instead of discrete-time predictive control is that the design model and the algorithm are robust in a fast sampling environment. The discrete-time models and predictive control algorithms could encounter problems when the system is under fast sampling condition. Another advantage is that because the design is performed in the continuous-time domain, the discretization is carried out in the implementation stage and the framework permits the control of an irregular sampled-data system. Thus, the continuous time MPC will be used in this research.

The general design philosophy of model predictive control is to compute a trajectory of a future manipulated variable, \( u_{mpc}(t) \), to optimize the future behavior of the plant output \( q_i(t) \). In other words, the MPC uses a dynamical model of the process to predict its future evolution and optimize the control signal. The optimization is performed within a limited time window starts by \( t_i \) with length \( T_p \). With a given initial time \( t_i \), the window is taken from \( t_i \) to \( t_i + T_p \). The length of the window \( T_p \) remains constant. The prediction horizon equals the length of the moving horizon window \( T_p \) and indicates how far the future will be predicted. Thus, it provides the advantage of repeated online optimization. Based on the receding horizon control principle, although the optimal trajectory of the future control signal is completely captured within the moving horizon window, the actual control input to the plant only takes the first instant of the control signal, while neglecting the rest of the trajectory. At a given time, the first value of this optimal sequence is applied to the
5. CONTROL SYSTEM DEVELOPMENT

process. When a new set of measurements is available to the controller, a re-optimization is performed, and the first value of this new optimal sequence of manipulated variables is implemented.

The continuous-time MPC design \[83\] uses the derivative of the control signal, \(\dot{u}_{mpe}(t)\), and thus it is always designed based on an augmented model of the plant which is given by

\[
\dot{x}_a(t) = A_a x_a(t) + B_a \dot{u}_{mpe}(t),
\]
\[
q_i(t) = C_a x_a(t),
\]
(5.34)

where \(x_a(t) = \begin{bmatrix} \dot{x}_p(t) \\ q_i(t) \end{bmatrix}, A_a = \begin{bmatrix} A_p & O_{2 \times 1} \\ C_p & 1 \end{bmatrix}, B_a = \begin{bmatrix} B_p \\ 0 \end{bmatrix}, C_a = \begin{bmatrix} O_{1 \times 2} & 1 \end{bmatrix}\). This implies that the signal \(\dot{u}_{mpe}(t)\) is an exponentially decay function that goes to 0.

Therefore, the control trajectory, \(\dot{u}_{mpe}(t)\), can be modeled by a set of exponentially decaying functions within the interval \(0 \leq t \leq T_p\).

One candidate function, to describe the derivative of the control signal, is the set of Laguerre polynomials \[84,85\], which is a set of orthonormal functions.

The following set of Laguerre functions is used

\[
X(t) = X(0)e^{A_u t},
\]
(5.35)

where \(A_u = \begin{bmatrix} -s_u & 0 & \ldots & 0 \\ -2s_u & -s_u & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -2s_u & \ldots & -2s_u & -s_u \end{bmatrix} \in \mathbb{R}^{n_u \times n_u}\),

\(s_u\) is the time scaling factor that determines the exponential decay rate, \(n_u\) is the number of terms of Laguerre polynomials, and the initial condition is \(X(0) = \sqrt{2s_u}[1, 1, \ldots, 1]^T \in \mathbb{R}^{n_u}\). The accuracy of the approximation increases as the number of terms increases. This approach provides a simple and efficient method to mathematically model the control trajectory with minimum prior information. Also it simplifies the solutions and enable us to
tune the predictive control system via only its two parameters (i.e., \( s_u \) and \( n_u \)). This is unlike the classic design of the MPC which is based on Taylor series expansion that provides non-parametric description for the MPC control trajectory.

Considering the moving time window from \( t_i \) to \( t_i + T_p \), and the time variable within this window is \( \iota \), the derivative of the control signal, \( \dot{u}_{mpc}(\iota) \), can be described by this set of functions as

\[
\dot{u}_{mpc}(\iota) = X(\iota)^T \eta,
\]

where \( \eta = [c_{u1}, c_{u1}, ..., c_{u_{n_u}}]^T \) is a coefficient vector, which will be designed later.

### 5.3.3 Predicted plant response

The predicted future state vector at time \( \iota \) is given as

\[
x_a(t_i + \iota | t_i) = e^{A_a \iota} x_a(t_i) + \Phi_{mpc}(\iota)^T \eta,
\]

where \( \Phi_{mpc}(\iota)^T = \int_0^\iota e^{A_a(\iota - \gamma)} BX(\gamma)^T d\gamma \). From the prediction of the state variable, the predicted output at time \( \iota \) is

\[
q_i(t_i + \iota | t_i) = C_a e^{A_a \iota} x_a(t_i) + C_a \Phi_{mpc}(\iota)^T \eta.
\]

Computation of \( \Phi_{mpc}(\iota) \) can be obtained in a recursive solution in terms of \( \iota = 0, h, 2h, \ldots, T_p \), by

\[
\Phi_{mpc}(kh)^T = e^{A_a h} \Phi_{mpc}((k - 1)h)^T + \Phi_{mpc}(h)^T e^{(k-1)A_a h}.
\]

### 5.3.4 Optimal control strategy

The MPC design is based on the solution of the so-called Finite Horizon Optimal Control Problem which consists of minimizing a suitably defined cost function with respect to the control sequence. The cost function at time \( t_i \) is

\[
J_{mpc} = \int_0^{T_p} (q_i(t_i) - q(t_i + \iota | t_i))^T Q_{mpc}(q_i(t_i) - q(t_i + \iota | t_i)) + \dot{u}_{mpc}(\iota)^T R_{mpc} \dot{u}_{mpc}(\iota) dt,
\]

89
where $Q_{mpc}$ and $R_{mpc}$ are weighting matrices that must be positive semi-definite and positive definite, respectively, to ensure the stability of the outer loop \[86\]. The optimal control $\dot{u}_{mpc}(t)$ can be calculated by minimizing this cost function, $J_{mpc}$, which is the optimal control within the moving window.

By substituting the prediction of the state variable $x_a(t_i + \iota|t_i)$ from (5.37) into (5.40), the cost function becomes function of $\eta$ as

$$J_{mpc} = (\eta + \Omega_{mpc}^{-1}\Psi_{mpc}x_a(t_i))^T\Omega_{mpc}(\eta + \Omega_{mpc}^{-1}\Psi_{mpc}x_a(t_i))$$

$$+ x_a(t_i)^T \int_0^{T_p} e^{A\iota}Q_{mpc}e^{A\iota} - x_a(t_i)^T\Psi_{mpc}^T\Omega_{mpc}^{-1}\Psi_{mpc}x_a(t_i).$$  

Then, by minimizing this cost function in terms of $\eta$, the optimal $\eta$ is

$$\eta = -\Omega_{mpc}^{-1}\Psi_{mpc}x_a(t_i),$$

where $\Omega_{mpc} = \int_0^{T_p} \Phi_{mpc}(t)Q_{mpc}\Phi_{mpc}(t)^T dt + R_{mpc}$, and $\Psi_{mpc} = \int_0^{T_p} \Phi_{mpc}(t)Q_{mpc}e^{A\iota}dt$. Thus, the optimal control signal can be calculated, from (5.36), in the window from $0 \leq \iota \leq T_p$.

It is difficult to obtain the analytical solutions for these integral expressions, $\Omega_{mpc}$ and $\Psi_{mpc}$. However, these matrices are computed over a given prediction horizon $T_p$. Thus, the integral expressions can be evaluated off-line using a numerical approximation as

$$\Omega_{mpc} = \sum_{k=0}^{T_p} \Phi_{mpc}(kh)Q_{mpc}\Phi_{mpc}(kh)^T h + R_{mpc},$$

$$\Psi_{mpc} = \sum_{k=0}^{T_p} \Phi_{mpc}(kh)Q_{mpc}e^{A_k h}.$$  

By applying the principle of receding horizon control (i.e., the control action will use only the derivative of the future control signal at $\iota = 0$), one can get the derivative of the optimal control as following:

$$\dot{u}_{mpc}(t) = -X(0)^T\Omega_{mpc}^{-1}\Psi_{mpc}x_a(t),$$
5.3 Robust Optimal Position Control

5.3.5 Constrained control

The next step is to formulate the constraints as part of the design requirements, and relate them to the original model predictive control problem. Since the predictive control problem is formulated and solved in the framework of receding horizon control, the constraints are taken into consideration frame-by-frame for each moving horizon window. This gives us the means to solve the constrained control problem numerically.

The considered constraints, in this work, are that of the robot actuators ($u_{\text{min}} \leq u \leq u_{\text{max}}$). Thus, firstly, we have to relate them to the optimized control trajectory of the MPC (i.e., $u_{\text{mpc min}} \leq u_{\text{mpc}} \leq u_{\text{mpc max}}$) as following:

The input torque from the controller and applied to the robot, $\tau$, can be calculated from

$$\tau = \tau_{\text{mpc}} + \dot{\tau}_{\text{dis}}$$

(5.47)

where $\tau_{\text{mpc}}$ is the torque generated from the MPC controller and it can be calculated from

$$\tau_{\text{mpc}} = M_n \ddot{q}_{\text{des}} = M_n u_{\text{mpc}}.$$

(5.48)

Thus, one can build the following constraints on $\tau_{\text{mpc}}$ as

$$\tau_{\text{min}} - \dot{\tau}_{\text{dis}} \leq \tau_{\text{mpc}} \leq \tau_{\text{max}} - \dot{\tau}_{\text{dis}}.$$

(5.49)

The generalized input forces, $\tau$, can be related to the actuators’ inputs via (2.90 - 2.93)
5. CONTROL SYSTEM DEVELOPMENT

and (5.31) as

\[
\tau = \begin{bmatrix}
\tau_x \\
\tau_y \\
\tau_z \\
\tau_\psi \\
\tau_\theta \\
\tau_\phi \\
\tau_{m1} \\
\tau_{m2}
\end{bmatrix} = \begin{bmatrix}
\tau_z(\theta C_\psi + \phi S_\psi) \\
\tau_z(\theta S_\psi - \phi C_\psi) \\
F_1 + F_2 + F_3 + F_4 \\
-\gamma_1 F_1 + \gamma_2 F_2 - \gamma_3 F_3 + \gamma_4 F_4 \\
d_q(F_3 - F_1) \\
d_q(F_4 - F_2) \\
\tau_{m1} \\
\tau_{m2}
\end{bmatrix}.
\] (5.50)

By reformulating the 3rd to the 6th element in (5.50), one can get the thrusters’ inputs as following

\[
\begin{bmatrix}
F_1 \\
F_2 \\
F_3 \\
F_4
\end{bmatrix} = \begin{bmatrix}
\frac{\tau_x}{4} + \frac{\tau_\psi}{4 \gamma_1} - \frac{\tau_\theta}{2 d_q} \\
\frac{\tau_y}{4} - \frac{\tau_\psi}{4 \gamma_2} + \frac{\tau_\phi}{2 d_q} \\
\frac{\tau_z}{4} + \frac{\tau_\psi}{4 \gamma_3} + \frac{\tau_\theta}{2 d_q} \\
\frac{\tau_\psi}{4} - \frac{\tau_\phi}{4 \gamma_4} - \frac{\tau_\theta}{2 d_q}
\end{bmatrix}.
\] (5.51)

To constrain the thrusters’ input to be \( F_{min} \leq F_i \leq F_{max} \), relation (5.51) can be utilized to get the following constraints on the robot’s inputs \( \tau_z, \tau_\psi, \tau_\theta, \) and \( \tau_\phi \).

**Constraints for \( \tau_z \):**

To calculate the required constraints on \( \tau_z \), each row in (5.51) is solved to satisfy the constrain \( F_i = [F_{min}, F_{max}] \). Thus, there are four constraints on \( \tau_z \). The minimum values of the constraints are given by

\[
\begin{bmatrix}
\tau_{z, min}^1 \\
\tau_{z, min}^2 \\
\tau_{z, min}^3 \\
\tau_{z, min}^4
\end{bmatrix} = \begin{bmatrix}
4F_{min} - \frac{\tau_\psi}{\gamma_1} + \frac{2 \tau_\theta}{d_q} \\
4F_{min} + \frac{\tau_\psi}{\gamma_2} - \frac{2 \tau_\phi}{d_q} \\
4F_{min} - \frac{\tau_\psi}{\gamma_3} - \frac{2 \tau_\theta}{d_q} \\
4F_{min} + \frac{\tau_\psi}{\gamma_4} + \frac{2 \tau_\phi}{d_q}
\end{bmatrix},
\] (5.52)
5.3 Robust Optimal Position Control

where $\tau_{z}^{min}$ is the lower limit of the $\tau_z$ constraint obtained from the constraint of $F_i$.

The maximum values of the constraints are given by

\[
\begin{bmatrix}
\tau_{z}^{\text{max}1} \\
\tau_{z}^{\text{max}2} \\
\tau_{z}^{\text{max}3} \\
\tau_{z}^{\text{max}4}
\end{bmatrix}
= \begin{bmatrix}
4F_{\text{max}} - \frac{\tau_\psi}{\gamma_1} + \frac{2\tau_\theta}{d_q} \\
4F_{\text{max}} + \frac{\tau_\psi}{\gamma_2} - \frac{2\tau_\theta}{d_q} \\
4F_{\text{max}} - \frac{\tau_\psi}{\gamma_3} - \frac{2\tau_\theta}{d_q} \\
4F_{\text{max}} + \frac{\tau_\psi}{\gamma_4} + \frac{2\tau_\theta}{d_q}
\end{bmatrix}. \tag{5.53}
\]

While $\tau_{z}^{\text{min}}$ can be calculated from

\[
\tau_{z}^{\text{min}} = \max(\tau_{z}^{\text{min}1}, \tau_{z}^{\text{min}2}, \tau_{z}^{\text{min}3}, \tau_{z}^{\text{min}4}), \tag{5.54}
\]

and $\tau_{z}^{\text{max}}$ can be calculated from

\[
\tau_{z}^{\text{max}} = \min(\tau_{z}^{\text{max}1}, \tau_{z}^{\text{max}2}, \tau_{z}^{\text{max}3}, \tau_{z}^{\text{max}4}). \tag{5.55}
\]

The torques, $\tau_\psi$, $\tau_\theta$ and $\tau_\phi$, in the right side of [5.52 and 5.53] are substituted by their values from last time step (i.e., $\tau_\psi(t - \delta t)$, $\tau_\theta(t - \delta t)$ and $\tau_\phi(t - \delta t)$) or by their predicted values (i.e., $\tau_\psi(t - \delta t) + \Delta\tau_\psi(t - \delta t)$, $\tau_\theta(t - \delta t) + \Delta\tau_\theta(t - \delta t)$, and $\tau_\phi(t - \delta t) + \Delta\tau_\phi(t - \delta t)$). Where $\Delta\tau(t - \delta t) = \tau(t - \delta t) - \tau(t - 2\delta t)$. Both of these two cases are tested.

Similarly, the constraints for $\tau_\psi$, $\tau_\theta$, and $\tau_\phi$ are calculated as:

**Constraints for $\tau_\psi$:**

\[
\begin{bmatrix}
\tau_{\psi}^{\text{min}1} \\
\tau_{\psi}^{\text{min}2} \\
\tau_{\psi}^{\text{min}3} \\
\tau_{\psi}^{\text{min}4}
\end{bmatrix}
= \begin{bmatrix}
\gamma_1(4F_{\text{min}} - \tau_z + \frac{2\tau_\theta}{d_q}) \\
\gamma_2(-4F_{\text{max}} + \tau_z + \frac{2\tau_\theta}{d_q}) \\
\gamma_3(4F_{\text{min}} - \tau_z - \frac{2\tau_\theta}{d_q}) \\
\gamma_4(-4F_{\text{max}} + \tau_z - \frac{2\tau_\theta}{d_q})
\end{bmatrix}. \tag{5.56}
\]
5. CONTROL SYSTEM DEVELOPMENT

The maximum values of the constraints are given by

\[
\begin{bmatrix}
\tau_{\psi}^{\max 1} \\
\tau_{\psi}^{\max 2} \\
\tau_{\psi}^{\max 3} \\
\tau_{\psi}^{\max 4}
\end{bmatrix} =
\begin{bmatrix}
\gamma_1 (4F_{\max} - \tau_z + \frac{2\tau_\theta}{d_q}) \\
\gamma_2 (-4F_{\min} + \tau_z + \frac{2\tau_\phi}{d_q}) \\
\gamma_3 (4F_{\max} - \tau_z - \frac{2\tau_\theta}{d_q}) \\
\gamma_4 (-4F_{\min} + \tau_z - \frac{2\tau_\phi}{d_q})
\end{bmatrix},
\]  
\text{(5.57)}

While \(\tau_{\psi}^{\min}\) can be calculated from

\[
\tau_{\psi}^{\min} = \max(\tau_{\psi}^{\min 1}, \tau_{\psi}^{\min 2}, \tau_{\psi}^{\min 3}, \tau_{\psi}^{\min 4}),
\]  
\text{(5.58)}

and \(\tau_{\psi}^{\max}\) can be calculated from

\[
\tau_{\psi}^{\max} = \min(\tau_{\psi}^{\max 1}, \tau_{\psi}^{\max 2}, \tau_{\psi}^{\max 3}, \tau_{\psi}^{\max 4}).
\]  
\text{(5.59)}

Constraints for \(\tau_{\theta}\):

\[
\begin{bmatrix}
\tau_{\theta}^{\min 1} \\
\tau_{\theta}^{\min 3}
\end{bmatrix} =
\begin{bmatrix}
2d_q (-F_{\max} + \frac{\tau_z}{4} + \frac{\tau_\psi}{4\gamma_1}) \\
2d_q (F_{\min} - \frac{\tau_z}{4} - \frac{\tau_\psi}{4\gamma_3})
\end{bmatrix},
\]  
\text{(5.60)}

The maximum values of the constraints are given by

\[
\begin{bmatrix}
\tau_{\theta}^{\max 1} \\
\tau_{\theta}^{\max 3}
\end{bmatrix} =
\begin{bmatrix}
2d_q (-F_{\min} + \frac{\tau_z}{4} + \frac{\tau_\psi}{4\gamma_1}) \\
2d_q (F_{\max} - \frac{\tau_z}{4} - \frac{\tau_\psi}{4\gamma_3})
\end{bmatrix},
\]  
\text{(5.61)}

While \(\tau_{\theta}^{\min}\) can be calculated from

\[
\tau_{\theta}^{\min} = \max(\tau_{\theta}^{\min 1}, \tau_{\theta}^{\min 3}),
\]  
\text{(5.62)}

and \(\tau_{\theta}^{\max}\) can be calculated from

\[
\tau_{\theta}^{\max} = \min(\tau_{\theta}^{\max 1}, \tau_{\theta}^{\max 3}).
\]  
\text{(5.63)}
5.3 Robust Optimal Position Control

Constraints for $\tau_\phi$:

$$\begin{bmatrix} \tau_{\phi}^{\min^2} \\ \tau_{\phi}^{\min^4} \end{bmatrix} = \begin{bmatrix} 2d_q(F_{\min} - \frac{\tau_z}{4} + \frac{\tau_\psi}{4\gamma_2}) \\ 2d_q(-F_{\max} + \frac{\tau_z}{4} - \frac{\tau_\psi}{4\gamma_4}) \end{bmatrix}. \quad (5.64)$$

The maximum values of the constraints are given by

$$\begin{bmatrix} \tau_{\phi}^{\max^1} \\ \tau_{\phi}^{\max^3} \end{bmatrix} = \begin{bmatrix} 2d_q(F_{\max} - \frac{\tau_z}{4} + \frac{\tau_\psi}{4\gamma_2}) \\ 2d_q(-F_{\min} + \frac{\tau_z}{4} - \frac{\tau_\psi}{4\gamma_4}) \end{bmatrix}. \quad (5.65)$$

While $\tau_{\phi}^{\min}$ can be calculated from

$$\tau_{\phi}^{\min} = \max(\tau_{\phi}^{\min^2}, \tau_{\phi}^{\min^4}), \quad (5.66)$$

and $\tau_{\phi}^{\max}$ can be calculated from

$$\tau_{\phi}^{\max} = \min(\tau_{\phi}^{\max^2}, \tau_{\phi}^{\max^4}). \quad (5.67)$$

Constraints for $\tau_x$ and $\tau_y$:

From the 1st and 2nd elements in (5.50), one can get the constraints for $\tau_x$ and $\tau_y$ as:

If $(\theta C_\psi + \phi S_\psi) > 0$, then

$$\tau_{x}^{\min} = (\theta C_\psi + \phi S_\psi) \tau_{z}^{\min}, \quad (5.68)$$

$$\tau_{x}^{\max} = (\theta C_\psi + \phi S_\psi) \tau_{z}^{\max}. \quad (5.69)$$

If $(\theta C_\psi + \phi S_\psi) < 0$, then

$$\tau_{x}^{\min} = (\theta C_\psi + \phi S_\psi) \tau_{z}^{\max}, \quad (5.70)$$

$$\tau_{x}^{\max} = (\theta C_\psi + \phi S_\psi) \tau_{z}^{\min}. \quad (5.71)$$
5. CONTROL SYSTEM DEVELOPMENT

If \((\theta S_\psi - \phi C_\psi) > 0\), then

\[
\tau_{y_{\text{min}}} = (\theta S_\psi - \phi C_\psi) \tau_z^{\text{min}},
\]

\[
\tau_{y_{\text{max}}} = (\theta S_\psi - \phi C_\psi) \tau_z^{\text{max}}.
\]

If \((\theta S_\psi - \phi C_\psi) < 0\), then

\[
\tau_{y_{\text{min}}} = (\theta S_\psi - \phi C_\psi) \tau_z^{\text{max}},
\]

\[
\tau_{y_{\text{max}}} = (\theta S_\psi - \phi C_\psi) \tau_z^{\text{min}}.
\]

In the case that the coefficients, \((\theta C_\psi + \phi S_\psi)\) and \((\theta S_\psi - \phi C_\psi)\), are very small, the constraints on the \(\tau_x\) and \(\tau_y\) are released to keep a minimum distance between the upper and lower limits of \(\tau_x\) and \(\tau_y\) as following:

If \(\tau_{x_{\text{max}}} \leq \tau_x^s\), then \(\tau_x^{\text{max}} = \tau_x^s\). If \(\tau_{x_{\text{min}}} \leq -\tau_x^s\), then \(\tau_x^{\text{min}} = -\tau_x^s\). If \(\tau_{y_{\text{max}}} \leq \tau_y^s\), then \(\tau_y^{\text{max}} = \tau_y^s\). If \(\tau_{y_{\text{min}}} \leq -\tau_y^s\), then \(\tau_y^{\text{min}} = -\tau_y^s\). Where \(\tau_x^s\) and \(\tau_y^s\) are constants.

**Constraints for \(\tau_{m1}\) and \(\tau_{m2}\)**: These constraints can be obtained directly from the joints motors’ data sheet which they are the limits on the joints motors’ torques.

After calculating the limits for the constraints of the input torques, the next step is to calculate the limits for the \(u_{\text{mpc}}\) as

\[
\frac{\tau_{\text{min}} - \hat{\tau}_{\text{dis}}}{M_n} \leq u_{\text{mpc}} \leq \frac{\tau_{\text{max}} - \hat{\tau}_{\text{dis}}}{M_n},
\]

This means that

\[
u_{\text{mpc}}^{\text{min}} = \frac{\tau_{\text{min}} - \hat{\tau}_{\text{dis}}}{M_n},
\]

\[
u_{\text{mpc}}^{\text{max}} = \frac{\tau_{\text{max}} - \hat{\tau}_{\text{dis}}}{M_n}.
\]

Since these constraints are time varying, they will be calculated at the start of each time window \((T_p)\) then they are hold (i.e., zero order hold) for the reset period of this window.

Now we will illustrate how the MPC will constrain the \(u_{\text{mpc}}\).

Let us assume that \(\delta t\) is the sampling interval for implementation, the first sample of the
control signal at the optimization window, is calculated as

\[ u_{mpc}(t_i) = u_{mpc}(t_i - \delta t) + X(0)^T \eta \delta t, \]  

\[ (5.79) \]

where \( u_{mpc}(t_i - \delta t) \) is the previous information of the control signal which is shifted backward by \( \delta t \).

Thus, at the first sample time of the window, the control signal constraints are

\[ u_{mpc}^{\text{min}} - u_{mpc}(t_i - \delta t) \leq X(0)^T \eta \delta t \leq u_{mpc}^{\text{max}} - u_{mpc}(t_i - \delta t). \]

\[ (5.80) \]

However, at the arbitrary time \( \iota_i \), the control signal is

\[ u_{mpc}(\iota_i) = u_{mpc}(t_i) + \int_0^{\iota_i} \dot{u}_{mpc}(\gamma) d\gamma \]

\[ = u_{mpc}(t_i) + (X(\iota_i) - X(0))^T A_u^{-T} \eta. \]

With the information of \( u(t_i - \delta t) \), the future control signal at time \( \iota_i \) is presented as

\[ u_{mpc}(\iota_i) = u_{mpc}(t_i - \delta t) + (X(0)^T \delta t + X(\iota_i)^T A_u^{-T} - X(0)^T A_u^{-T}) \eta. \]

\[ (5.82) \]

Thus, the inequality constraints of the control signal are expressed as

\[ u_{mpc}^{\text{min}} - u_{mpc}(t_i - \delta t) \leq C_u \eta \leq u_{mpc}^{\text{max}} - u_{mpc}(t_i - \delta t), \]

\[ (5.83) \]

where \( C_u = X(0)^T \delta t + X(\iota_i)^T A_u^{-T} - X(0)^T A_u^{-T} \).

These constraints can be reformulated as linear inequalities as following

\[ M_{mpc} \eta \leq \gamma_{mpc}, \]

\[ (5.84) \]

where

\[ M_{mpc} = \begin{bmatrix} X(0)^T \delta t + X(\iota_i)^T A_u^{-T} - X(0)^T A_u^{-T} \\ -(X(0)^T \delta t + X(\iota_i)^T A_u^{-T} - X(0)^T A_u^{-T}) \end{bmatrix}, \]

\[ (5.85) \]
5. CONTROL SYSTEM DEVELOPMENT

and

\[
\gamma_{mpc} = \begin{bmatrix}
u_{mpc}^{max} - u_{mpc}(t_i - \delta t) \\
u_{mpc}^{min} + u_{mpc}(t_i - \delta t)
\end{bmatrix}.
\]  

(5.86)

Since the objective function (5.42) is quadratic, and the constraints are linear inequalities, the problem of finding an optimal predictive control becomes one of finding an optimal solution to a standard quadratic programming problem. These inequality constraints consist of active constraints and inactive constraints. An inequality \( M_{mpc} \eta \leq \gamma_{mpc} \) is said to be active if \( M_{mpc} \eta = \gamma_{mpc} \) and inactive if \( M_{mpc} \eta < \gamma_{mpc} \).

A simple and efficient algorithm, called Hildreth’s quadratic programming procedure \[87\], is proposed for solving this problem. In this method, to reduce the computational load, the constraints that are not active is identified systematically, then they are eliminated in the solution. The active constraints are equality constraints, so one can rely on the Lagrange multipliers, \( \Lambda \), to solve this optimization problem (Quadratic objective function with equality constraints). This method will lead to very simple programming procedures for finding optimal solutions of constrained minimization problems. This algorithm is based on an element-by-element search, therefore, it does not require any matrix inversion.

In this method, the \( i^{th} \) Lagrange multiplier at iteration \( k + 1 \) is calculated from

\[
\Lambda_i^{k+1} = \max(0, W_i^{k+1}),
\]

(5.87)

where

\[
W_i^{k+1} = \frac{1}{h_{ii}}(e_i + \sum_{j=1}^{i-1} h_{ij}\Lambda_j^{k+1} + \sum_{j=i+1}^{n} h_{ij}\Lambda_j^{k}),
\]

(5.88)

\( h_{ij} \) represents the \( ij^{th} \) element of the matrix \( H_{mpc} = M_{mpc} \Omega_{mpc}^{-1} M_{mpc}^T \), and \( e_i \) is the \( i^{th} \) element of the vector \( E_{mpc} = \gamma_{mpc} + M_{mpc} \Omega_{mpc}^{-1} \Psi_{mpc} x_a \).

The iteration will be terminated when the iterative counter reaches its maximum value, \( m_{max} \), at which the \( \Lambda \) converges to its optimal value, \( \Lambda^* \), which contains zeros for inactive constraints and the positive components corresponding to the active constraints. Then, the optimal \( \eta \) can be calculated from

\[
\eta^* = -\Omega_{mpc}^{-1}(\Psi_{mpc} + M_{mpc}^T \Lambda^*),
\]

(5.89)
5.3 Robust Optimal Position Control

and thus the optimal control signal can be obtained from

\[ u_{mpc}(t) = \int_0^t \dot{u}_{mpc}(\tau) d\tau. \] (5.90)

with

\[ \dot{u}_{mpc}(\tau) = X(\tau)^T \eta^*. \] (5.91)

Summary of the MPC technique is presented as algorithm 3.

**Algorithm 3** MPC algorithm

1: time \( t = t_i \)

2: while \((t \leq t_f)\) do

3: \hspace{1em} for \( t = t_i : \delta t : T_p \) do

4: \hspace{2em} Get the current state \( x_a(t) \)

5: \hspace{2em} Solve the Optimal Control Problem numerically by minimizing the cost function \( J_{mpc} \) subjected to the constraints (5.84)

6: \hspace{2em} Find the optimal control sequence \( u^*_{mpc}(t_i) \ldots u^*_{mpc}(t) \ldots u^*_{mpc}(T_p) \)

7: \hspace{2em} Apply only \( u^*_{mpc}(t_i) \)

8: \hspace{1em} end for\>

9: end while \>

▷ Repeat the procedure in the next prediction window of length \( T_p \)

5.3.6 MPC tuning guidelines

The design variables in the continuous time model predictive control algorithm presented above are [84]:

- Weighting matrices, \( Q_{mpc} \) and \( R_{mpc} \): The smaller elements in \( R_{mpc} \) and larger elements of \( Q_{mpc} \) corresponds to faster closed-loop response speed.

- Pole location of the Laguerre model for the future control signal, \( s_u \): is a tuning parameter for the closed-loop response speed. Larger \( p \) results in larger changes in the future control signal in the initial period, hence faster closed-loop response speed.
5. CONTROL SYSTEM DEVELOPMENT

- Number of terms used in the Laguerre model to capture the control signal, \( n_u \): As \( n_u \) increases, the degree of freedom in describing the control trajectory increases. The control signal tends to be more aggressive when \( n_u \) increases thus faster closed-loop response speed.

- Prediction horizon \( T_p \): For stable system, the prediction horizon is recommended to be chosen approximately equal to the open loop process settling time. For unstable and integrating processes, which as in our case, \( T_p \) is recommended to be chosen as a parameter of \( s_u \) approximately equal to \( \frac{10}{s_u} \).

Fig. 5.10 shows the complete block diagram of the proposed control system. As it is stated before, the position, the manipulator joints’ angles, and the yaw angle are usually the controlled variables, while pitch and roll angles are used as intermediate control inputs for horizontal positions control. Therefore, the proposed control system consists from two DOB-based controllers by dividing \( q \) into two parts; one for \( \zeta = [x \ y \ z \ \psi \ \theta_1 \ \theta_2]^T \) (with \( MPC_\zeta, M_{n_u}, M_{nv_1}, P_{v_1}, Q_{v_1}, \text{and} \ Q_a \)) and the other for \( \sigma_b = [\theta \ \phi]^T \) (with \( MPC_\sigma, M_{nv_2}, P_{v_2}, \text{and} \ Q_{v_2} \)). The desired trajectories for the end-effector’s position and orientation \( (p_{e,r} \text{ and } \Phi_{e,r}) \) are fed to the inverse kinematics algorithm together with \( \sigma_{b,r}(t) \) such that the desired quadrotor/joint space trajectories \( \zeta_r(t) \) are obtained. After that, the controller block receives the desired trajectories and the feedback signals from the system and then provides the control signal, \( \tau \). The desired values \( \sigma_{b,r} \) for the intermediate controller are obtained from the output of position controller, \( \tau_\zeta \), through (5.31)

The output of the two controllers, \( \tau_\zeta \text{ and } \tau_\sigma \), are mixed to generate the final control vector \( \tau \) which is then converted to the forces/torques applied to quadrotor/manipulator via the relation (5.32)

Remark 3. It is better to make the response of the \( \sigma \) controller faster than that of the quadrotor position controller such that it can track the changes in the position controller \([81,82]\). This can be achieved by the tuning parameters of both DOB and MPC.
5.3 Robust Optimal Position Control

### 5.3.7 Simulation Results

In this section the previously proposed control strategy is applied in simulation MATLAB/SIMULINK program to the model of the considered aerial manipulation robot. The simulation environments similar to that in section 5.2.2.1 is used here. Parameters of the MPC controller are given in Table 5.2.

The system response in the quadrotor/joint space is presented in Fig. 5.11. Figs. 5.12 and 5.13 show responses of the proposed scheme in the task space (the actual end-effector position and orientation can be found from the forward kinematics). Note that the marker represents the end-effector orientation; Green, Blue, and Red for x-, y-, and z-axis. These results show that the proposed motion control scheme able to track the desired trajectories, and so the control objectives 1 - 3 are achieved. Moreover, the MPC reduces the oscillations that appears at the start of motion by using the PD-DOb method.

Fig. 5.14 shows the control effort, the required thrust forces and manipulator torques which are in the allowable limits. It is noted also from this figure that the control effort becomes lower and has no sharpness than that in the case of using the PD-DOb technique.
Table 5.2: MPC-DOb controller parameters

| Parameter | Value |
|-----------|-------|
| $Q_{mpc}$ | $C^TC$ |
| $R_{mpc}$ | $I_6$ |
| $Q_{mpc_a}$ | $2C^TC$ |
| $R_{mpc_a}$ | $0.2I_2$ |
| $u^{max}$ | $[6 6 6 6 0.7 0.4]^T$ |
| $u^{min}$ | $[0 0 0 0 -0.7 -0.4]^T$ |
| $s_{u\zeta}$ | 2 |
| $s_{u\sigma}$ | 10 |
| $n_{u\zeta}$ | 3 |
| $n_{u\sigma}$ | 6 |
| $h$ | 0.01 |
| $k_{max}$ | 38 |
| $\tau_{x,s}^s$, $\tau_{y,s}^s$ | 0.5 |

Table 5.3 presents a comparison study between PD-DOb and MPC-DOb controllers. The first part in this table compares the Root Mean Square Error (RMSE) in the tracking of the end-effector position and orientation. The RMSE of the MPC is slightly lower than that of the PD-DOb. The second part presents the summation of quadrotor thrusters efforts as well as the manipulator torques. This table enlightens the superior performance of the MPC-DOb over the PD-DOb. The using of the MPC leads to a 15% reduction in the required thrust effort.
### 5.3 Robust Optimal Position Control

| RMSE   | PD-DOb | MPC-DOb |
|--------|--------|---------|
| $x_e$ [m] | 0.0390 | 0.0377  |
| $y_e$ [m] | 0.0370 | 0.0306  |
| $z_e$ [m] | 0.0227 | 0.0031  |
| $\psi_e$ [deg] | 0.4469 | 0.0250  |
| $\theta_e$ [deg] | 1.0199 | 0.0309  |
| $\phi_e$ [deg] | 0.2876 | 0.8113  |

Control Effort

| Sum $T$ [N] | PD-DOb | MPC-DOb |
|-------------|--------|---------|
| $\sum T$ [N] | 748.3817 | 645.4135 |
| $\sum \tau_{m1}$ [N.m] | 11.6674 | 11.6460 |
| $\sum \tau_{m2}$ [N.m] | 4.6629  | 4.6355  |
Figure 5.11: MPC-DOb results: The actual response in the quadrotor/joint space; a) $x$, b) $y$, c) $z$, d) $\psi$, e) $\theta_1$, and f) $\theta_2$.
Figure 5.12: MPC-DOb results: The actual response of the end-effector pose: a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$
5. CONTROL SYSTEM DEVELOPMENT

Figure 5.13: MPC-DOb results: 3D task space trajectory
Figure 5.14: MPC-DOb results: The required controller efforts
5.4 Robust Force Estimation and Impedance Control

In this section, a robust sensorless force estimation and impedance control system is presented to achieve the objectives 1, 2, 4, and 5.

The proposed methodology to satisfy these control objectives is illustrated in Fig. 5.15. In this control strategy, the system nonlinearities, external disturbance (wind), $\tau_w$, and contact force, $\tau_l$, are treated as extended disturbances, $\tau^{\text{dis}}$, which will be estimated, as $\hat{\tau}^{\text{dis}}$, and canceled by the DOb in the inner loop. The system can now be considered as a group of linear SISO subsystems. The output of DOb with system measurements of quadrotor/joint space variables and the derived task space variables are used as the inputs to the Fast Tracking Recursive Least Squares (FTRLS) algorithm to obtain the end-effector contact force $\hat{F}_e$. The task space impedance control is used in the external loop of DOb and its output is transformed to the quadrotor/joint space through a coordination transformation algorithm.

Figure 5.15: Functional block diagram of the FTRLS-DOb control system
5.4 Robust Force Estimation and Impedance Control

5.4.1 Modeling of the Environment Dynamics and the Wind Effect

The dynamical model of the quadrotor-manipulator system (2.96) can be rewritten as

\[
M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + \tau_w + \tau_l = \tau,
\]

(5.92)

where \(\tau_w \in \mathbb{R}^8\) is the vector of the external wind effect, and \(\tau_l \in \mathbb{R}^8\) is the vector of the contact force effect.

The environment dynamics, contact force, \(\tau_l\), can be modeled as following.

\[
\tau_l = J^T F_e,
\]

(5.93)

where the task space contact force, \(F_e\), can be modeled from

\[
F_e = S_c (\chi_e - \chi_{e,o}) + D_c \dot{\chi}_e,
\]

(5.94)

where the quantity \(\chi_{e,o}\) is taken equal to the previous state until the contact occurs. At the contact, \(\chi_{e,o}\) is equal to the state at the start of the interaction with the environment. \(S_c = diag\{S_{c1}, S_{c2}, S_{c3}, S_{c4}, S_{c5}, S_{c6}\}\) and \(D_c = diag\{D_{c1}, D_{c2}, D_{c3}, D_{c4}, D_{c5}, D_{c6}\}\) represent the environment stiffness and the environment damping, respectively. It is to be noted that the algorithm for contact force estimation is always run. If any component of the estimated force exceeds the threshold (0.1 N / 0.01 N.m), contact with the environment is indicated and its consequences are followed. Similarly, if the all components of the estimated force are below the threshold, non-contact condition with environment is indicated and its consequences are followed.

The wind dynamics, \(\tau_w\), can be modeled as following \[88-90]\:

The wind velocity is determined by

\[
V_{wz} = V_{wz_0} \frac{z}{z_0},
\]

(5.95)
where $V_{wz}$ is the wind velocity at altitude $z$, $V_{wz_0}$ is the specified wind velocity at altitude $z_0$.

To simulate wind disturbances, it is worth calculating the wind generalized force, $\tau_w$, which influences the platform. This force can be determined by

$$\tau_w = 0.61 * A_e V_{wz}^2,$$  \hspace{1cm} (5.96)

where 0.61 is the rate of wind velocity ($\text{m/s}^2$) conversion to pressure ($\text{N/m}^2$). $A_e$ is the effective area that depends on the quadrotor structure and its orientation which will be presented latter.

This force can be projected on the axes of frame $\Sigma$ as

$$\tau_{wx} = f_{wx_1} z^2 \sin(\theta) + f_{wx_2} z^2 \cos(\theta),$$

$$\tau_{wy} = f_{wy_1} z^2 \sin(\phi) + f_{wy_2} z^2 \cos(\phi),$$  \hspace{1cm} (5.97)

where

$$f_{wx_1} = 0.61 * A_{e_1} \left( \frac{V_{wz_0}}{z_0} \right)^2 \cos(\psi_w),$$

$$f_{wx_2} = 0.61 * A_{e_2} \left( \frac{V_{wz_0}}{z_0} \right)^2 \cos(\psi_w),$$

$$f_{wy_1} = 0.61 * A_{e_1} \left( \frac{V_{wz_0}}{z_0} \right)^2 \sin(\psi_w),$$

$$f_{wy_2} = 0.61 * A_{e_2} \left( \frac{V_{wz_0}}{z_0} \right)^2 \sin(\psi_w),$$

$\psi_w$ represents the angle of wind direction. Both $A_{e_1}$ and $A_{e_2}$ depend on the quadrotor design parameters and they are given as following:

Let us represent the quadrotor as a cylinder with height, $h_q$, and radius, $r_q$. Thus,

$$A_{e_1} = \beta_q \pi r_q^2,$$  \hspace{1cm} (5.102)
5.4 Robust Force Estimation and Impedance Control

\[ A_{e_2} = \alpha_q \pi r_q h_q, \quad (5.103) \]

where \( \beta_q \) and \( \alpha_q \) are the fill factors of the cylinder areas.

5.4.2 Fast Tracking Recursive Least Squares

In this part, a technique is developed which utilizes a Fast Tracking Recursive Least Squares (FTRLS) to estimate the contact force with the aid of the DOB linearization capabilities. The FTRLS algorithm is one of the fast online least squares-based identification methods used for the identification of environments with varying dynamic parameters \cite{91,92}. To apply FTRLS, the dynamic equations (5.92-5.97) have to be parametrized (i.e., to be product of measurement data regressor and dynamic parameters) as following.

The system dynamics part, \( \tau_{int} = M(q)\ddot{q} + C(q, \dot{q}) \dot{q} + G(q) \), can be rewritten as the product of data regressor, \( Y_i(q, \dot{q}, \ddot{q}) \), and platform parameters, \( h_i \).

\[
h_i = [m_1 m_2 m I_1 I_2 I_x l_0 m_1 l_0 m_2 l_1 m_1 l_1 m_2 l_2 m_1 l_2 m_2 l_0^2 m_1 l_0^2 m_2 l_1^2]
\]

(5.104)

The environment dynamics, \( \tau_e \), can be reformulated as \( Y_i(q, \dot{q}, \ddot{q}, \chi_e, \dot{\chi}_e) h_l \), where, \( Y_i = J^T Y_e \), \( Y_e \) is a function of the end effector states, \( (\chi_e, \dot{\chi}_e) \), which can be calculated as

\[
Y_e = \begin{bmatrix}
\chi_{e,e}(1) & \dot{\chi}_{e,e}(1) & O_{1 \times 1} & O_{1 \times 9} \\
O_{1 \times 2} & \chi_{e,e}(2) & \dot{\chi}_{e,e}(2) & O_{1 \times 8} \\
O_{1 \times 4} & \chi_{e,e}(3) & \dot{\chi}_{e,e}(3) & O_{1 \times 6} \\
O_{1 \times 6} & \chi_{e,e}(4) & \dot{\chi}_{e,e}(4) & O_{1 \times 4} \\
O_{1 \times 8} & \chi_{e,e}(5) & \dot{\chi}_{e,e}(5) & O_{1 \times 2} \\
O_{1 \times 9} & \chi_{e,e}(6) & \dot{\chi}_{e,e}(6) & O_{1 \times 1}
\end{bmatrix}
\]

(5.105)

\( h_l \) is the vector of the environment parameters \( S_c \) and \( D_c \) and it is given as

\[
h_l = [S_{c_1} D_{c_1} S_{c_2} D_{c_2} S_{c_3} D_{c_3} S_{c_4} D_{c_4} S_{c_5} D_{c_5} S_{c_6} D_{c_6}]^T. \quad (5.106)\]
Finally, the wind effect is formulated as $Y_w(z, \theta, \phi) h_w$. $Y_w$ can be calculated as

\[
Y_w = \begin{bmatrix}
  z^2 S_\theta & z^2 C_\theta & 0 & 0 \\
  0 & 0 & z^2 S_\phi & z^2 C_\phi \\
  0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0
\end{bmatrix}.
\] (5.107)

$h_w$ is the vector of the wind parameters and it is given as

\[
h_w = [f_{wx_1}, f_{wx_2}, f_{wy_1}, f_{wy_2}]^T.
\] (5.108)

Therefore, the total dynamics can be reformulated as

\[
\tau = Y_s h_s,
\]

\[
Y_s = [Y_i, Y_l, Y_w],
\] (5.109)

\[
h_s = [h_i, h_l, h_w]^T,
\]

where $Y_s \in \mathbb{R}^{8 \times 40}$ and $h_s \in \mathbb{R}^{40}$ are the data regressor and parameters vector of (5.92), respectively. The details of $Y_i$ and $Y_l$ are given in Appendix D.

The parameter estimation error is

\[
\tilde{h}_s(t) = h_s - \hat{h}_s(t),
\] (5.110)

while the estimation error is

\[
\tilde{\tau}(t) = \tau(t) - Y_s(t)\hat{h}_s(t) = Y_s(t)\tilde{h}_s(t).
\] (5.111)
5.4 Robust Force Estimation and Impedance Control

By minimizing a cost function with respect to the parameter estimation error, one can find the time derivative of the estimated parameters vector, \( \hat{h}_s \), as following

\[
\frac{d}{dt} \hat{h}_s(t) = R(t)Y^T_s(t)\tilde{\tau}(t), \tag{5.112}
\]

where \( R(t) \) is the parameters’ covariance matrix, and it can be calculated from

\[
\frac{d}{dt} R^{-1}(t) = -\eta_h(t)R^{-1}(t) + Y^T_s(t)Y_s(t), \tag{5.113}
\]

where \( \eta_h \) is the forgetting factor, and it is given as

\[
\eta_h(t) = \eta_h^{\text{min}} + (1 - \eta_h^{\text{min}})2^{(-NINT(\gamma_g\|\tilde{\tau}(t)\|^2))}, \tag{5.114}
\]

where \( \eta_h^{\text{min}} \) is a constant representing the minimum forgetting factor, \( NINT(.) \) is the round-off operator, and \( \gamma_g \) is a design constant. This adaptive formulation of the forgetting factor enables the RLS to track the non-stationary parameters to be estimated.

Finally, the estimated contact generalized force in the quadrotor/joint space and task space can be calculated from

\[
\hat{\tau}_l = Y_l\hat{h}_l, \quad \hat{F}_e = Y_e\hat{h}_l. \tag{5.115}
\]

Therefore, unlike the current developed schemes, this technique enables the isolation and the estimation of the end-effector contact force apart from the whole estimated forces in the systems. This estimated force will be utilized in the next section to design an impedance control for the proposed system. Furthermore, for the tele-manipulation, the use of force feedback is essential to assist the human operator in the remote handling of objects. Moreover, it is very critical to get the force feedback to achieve the cooperative grasping among aerial manipulation robots. If the load for the UAVs is heavier than the carrying capacity of a single UAV, multiple UAVs can distribute the load among them and move the object.
5. CONTROL SYSTEM DEVELOPMENT

5.4.3 Impedance Control

The objective of the impedance control is to impose, along each direction of the task space, a desired dynamic relation between the end-effector pose and the force of interaction with the environment (i.e., desired impedance). The linear impedance control is designed in the task space. This is based on the linearization effect of the designed DOb in the quadrotor/joint space. The desired acceleration in the task space, $\ddot{\chi}_{e,\text{des}}$, can be calculated from

$$
\ddot{\chi}_{e,\text{des}} = \ddot{\chi}_{e,r} + M_{c,d}^{-1} \left( S_{c,d}(\chi_{e,r} - \chi_{e}) + D_{c,d}(\dot{\chi}_{e,r} - \dot{\chi}_{e}) - \hat{F}_{e} \right),
$$

(5.116)

where $M_{c,d}$, $S_{c,d}$, and $D_{c,d}$ are constant diagonal positive definite matrices representing the desired inertia, stiffness, and damping, respectively, which determine the desired impedance that the end-effector will apply to the environment.

The control law (5.116) is active during the contact between the end-effector and the environment. Another control is used during the non-contact which is a tracking PD controller and it is given as

$$
\ddot{\chi}_{e,\text{des}} = \ddot{\chi}_{e,r} + K_{P,e}(\chi_{e,r} - \chi_{e}) + K_{D,e}(\dot{\chi}_{e,r} - \dot{\chi}_{e}),
$$

(5.117)

where $K_{P,e}$ and $K_{D,e}$ are constant diagonal positive definite matrices representing the proportional and derivative gains of the task space PD controller. Thereby, if the contact occurs the task space desired accelerations are computed from (5.116) while they are calculated from (5.117) if there is no contact.

Fig. 5.16 shows the complete block diagram of the proposed control system during contact. The quadrotor position, the manipulator joints’ angles and the yaw angle are usually the controlled variables, while pitch and roll angles are used as intermediate control inputs for horizontal positions control. Therefore, the proposed control system consists from two DOb-based controllers by dividing $q$ into two parts; one for $\zeta = [x \ y \ z \ \psi \ \theta_{1} \ \theta_{2}]^{T}$ (with $K_{\text{imp}}$, $Q_{a}$, $M_{na}$, $M_{nm1}$, $P_{v1}$, $Q_{v1}$) and the other for $\sigma_{b} = [\theta \ \phi]^{T}$ (with $K_{\sigma}$, $M_{nv2}$, $P_{v2}$, and $Q_{v2}$). The desired trajectories for the end-effector’s position and orientation, $\chi_{e,r}$, their actual values (by using the forward kinematics), and the estimated end-effector force, are fed to the impedance
control law, $K_{imp}$ that is described in (5.116). Then, a coordination transformation from task space to quadrotor/joint space is done by using (5.118) to obtain $\ddot{\zeta}^{des}$. The desired acceleration in the quadrotor/joint space, $\ddot{\zeta}^{des}$, can be calculated by differentiating (2.34) with respect to time as

$$
\ddot{\zeta}^{des} = J^{-1}_\zeta \left( Q_e \ddot{x}^{des}_e + \dot{Q}_e \dot{x}_e - J_b \ddot{\theta}_b - J_\sigma \ddot{\sigma}_b \right),
$$

(5.118)

The desired acceleration in quadrotor/joint space, $\ddot{\zeta}^{des}$, is then applied to the DOb of the independent coordinates, $\zeta$, to produce $\tau_\zeta$. The desired values for the intermediate DOb controller, $\sigma_{b,r}$, are obtained from the output of position controller, $\tau_\zeta$, through (5.31).

The external controller of the DOb for $\sigma$, $\tau_\sigma$, is used as a proportional controller with velocity feedback, $K_\sigma$, as following:

$$
\ddot{\sigma}^{des}_b = K_{p\sigma}(\sigma_{b,r} - \sigma_b) - K_{d\sigma} \dot{\sigma}_b.
$$

(5.119)

After that, $\ddot{\sigma}^{des}_b$ is applied to the DOb for $\sigma$ to generate $\tau_\sigma$.

However, the response of the $\sigma$ controller must be much faster than that of the quadrotor position controller such that it can track the changes in the position controller. This can be achieved by the tuning parameters of both the DOb and the proportional controller with velocity feedback of $\sigma_b$.

The output of the two controllers, $\tau_\zeta$ and $\tau_\sigma$, are combined together to generate the final control vector $\tau$ which is then converted to the forces/torques applied to quadrotor/manipulator through (5.32).

### 5.4.4 Simulation Results

In this section, the previously proposed control strategy is applied in MATLAB/SIMULINK to the model of the considered aerial manipulation robot with the same setup of section 5.2.2.1.

Parameters of the proposed algorithm are given in Table 5.4. The profile of the wind angle, $\psi_w$, is simulated as in Fig. 5.17. To emulate a real scenario, the value $V_{w_{z0}}$ is chosen to has two parts; the first one is constant and the second part is random variable to simulate
5. CONTROL SYSTEM DEVELOPMENT

Figure 5.16: Block diagram of the detailed FTRLS-DOb control system for the quadrotor manipulation system

The gust effect (sudden and random changes in the wind speed), see Fig. 5.18.

The aim of considered case study is to perform tracking of the desired 6-DOF end-effector trajectories (i.e., circular helix for the position and Quintic trajectories for the orientation) while an external force of 1.6 N acts along the z-axis of the end-effector, $z_e$, for $z_e \leq 10$ cm (i.e., the contact occurs during period from 50 s to 60 s). This scenario simulates, for instance, the case of the manipulator in contact with a soft environment. Thereby, the manipulator end-effector is chosen to be rigid. Hence the stiffness matrix, $S_{c,d}$, has been tuned to $S_{c,d} = diag\{20, 20, 20, 50, 50, 50\}$, while the damping matrix value is $D_{c,d} = diag\{15, 15, 15, 100, 100, 100\}$.

5.4.4.1 Estimation of the End-effector Contact Force

Figs. 5.19 shows the actual end-effector contact force and its estimation. 5.20 illustrates the norm of the contact force estimation error. From these figures, it is possible to recognize that before instant 50 s there is no contact with the environment and the contact starts at instant 50 s. In addition, it is observed that the norm error of the end-effector generalized force in all directions have values around zero during the non-contact period. At instant 50
5.4 Robust Force Estimation and Impedance Control

![Figure 5.17: Profile of the wind angle](image1.png)

![Figure 5.18: Profile of the wind speed](image2.png)

Figure 5.17: Profile of the wind angle

Figure 5.18: Profile of the wind speed
5. CONTROL SYSTEM DEVELOPMENT

Table 5.4: FTRLS-DOB simulation parameters

| Parameter     | Value          |
|---------------|----------------|
| $A_{e_1}$     | 0.16           |
| $S_{c,d}$     | $diag\{20,20,20,50,50\}$ |
| $D_c$         | $diag\{0,0,10,0,0,0\}$ |
| $M_{c,d}$     | $2I_6$         |
| $D_{c,d}$     | $diag\{15,15,15,100,100,100\}$ |
| $A_{e_2}$     | 0.032          |
| $\eta_{h_{min}}$ | 0.8            |
| $\gamma_g$   | 5              |
| $S_c$         | $diag\{0,0,20,0,0,0\}$ |
| $z_0$         | 1              |
| $K_{p_e}, K_{d_a}$ | $20I_2$       |
| $K_{p,c}$     | $diag\{25,25,150,200,200,500\}$ |
| $K_{D,e}$     | $diag\{15,15,100,100,500,100\}$ |

$s$, at which the contact occurs, there is an increase in the estimation error norm. The norm of estimation error in both $x$ and $y$ directions have maximum value of 0.016 N and 0.013 N, respectively. While it has value of 0.03 N in the $z$ direction, which is equivalent to 1.8 % with respect to the actual value. In $\phi$, $\psi$ and $\theta$ directions, the norms reach value of 0.0001 N.m. Thus, it is possible to appreciate the estimation performance of the end-effector generalized forces. Consequently, one can contend that the control objective 4 is achieved.

5.4.4.2 Impedance Control

Fig. 5.21 shows the response of the system in the task space (the actual end-effector position and orientation can be found from the forward kinematics). From this figure, it is possible to recognize that the controller has good tracking of the desired trajectories of the end-effector due to the using of the tracking controller during the period of non-contact. It also possible to recognize that at the time of the contact the required impedance is applied and this produces small error in the position tracking to satisfy the impedance requirements. Fig. 5.22 shows the motion of the end-effector in the 3D dimension (The markers represent the orientation). These results show that the proposed impedance motion control scheme provides a robust
5.4 Robust Force Estimation and Impedance Control

performance to track the desired end-effector trajectories as well as to achieve the desired impedance effect on the environment taken into consideration the external disturbances and noises. As a result, one can claim that the control objectives 1, 2, and 5 are achieved. Finally, Fig. 5.23 shows the actual trajectories of the system in the quadrotor/joint space from which it is noted that the generated trajectories in the quadrotor/joint space are in the allowable limits of the manipulator joints.
Figure 5.19: FTRLS-DOb results: Actual response of the estimation of the contact force
5.4 Robust Force Estimation and Impedance Control

![Graphs showing error norms for different coordinates](image)

**Figure 5.20:** FTRLS-DOb results: Error norm of the estimation of the contact force

- (a) $x_e$
- (b) $y_e$
- (c) $z_e$
- (d) $\phi_e$
- (e) $\theta_e$
- (f) $\psi_e$
Figure 5.21: FTRLS-DOb results: The actual response of the end-effector pose: a) $x_e$, b) $y_e$, c) $z_e$, d) $\phi_e$, e) $\theta_e$, and f) $\psi_e$
Figure 5.22: FTRLS-DOb results: 3D trajectory of the end-effector pose (The marker represents the end-effector orientation; Green, Blue, and Red for x-, y-, and z-axis, respectively)
Figure 5.23: FTRLS-DOb results: The actual trajectories of the variables in the quadrotor/joint space
Chapter 6

CONCLUSIONS

6.1 Conclusions

This thesis presents the development of control design and practical implementation of a new non-redundant quadrotor manipulation system that has superior features over the currently developed systems and has vital applications. The following conclusions are obtained:

1. Successful experimental verification is done for the design of a new quadrotor manipulation system that its 2-DOF manipulator has unique topology to enable the end-effector to track arbitrary 6-DOF trajectory. This system provide solutions to limitations found in the currently developed aerial manipulation systems by having maximum mobility with minimum weight.

2. The system parameters are identified to be utilized in both controller implementation and simulation of the developed dynamic model of the system.

3. A measurement and estimation scheme is proposed and implemented to get the six states representing the position and orientation of the system by using the onboard sensors. In contrast to previous works, this scheme enables utilizing the system within arbitrary place indoors and provides potential to work outdoors as well. This makes the system capable to do real practical applications.
6. CONCLUSIONS

4. The proposed system feasibility and efficiency are proved in both simulation and experiment with a scenario consisting of taking off with an object, moving, and releasing it at a desired position. Both simulation and experimental results demonstrate a satisfactory performance.

5. A novel inverse kinematics analysis is carried out to transform the task space trajectory to quadrotor/joint space trajectory and hence enables controlling the system in the quadrotor/joint space. Such analysis has not been done before either for redundant or non-redundant aerial manipulator.

6. An algorithm is developed to get approximate solution to the complex differential algebraic equations resulted from the inverse kinematics analysis that incorporate the high order nonholonomic constraints of the system.

7. Numerical results validate the high accuracy of the inverse kinematics solution.

8. The success of the inverse kinematics proves the ability of the system to track arbitrary 6-DOF task space trajectories that lay within the capability of the system’s actuators.

9. A quite realistic simulation setup is built to test the developed control systems.

10. Numerical results show robust performance and satisfactory tracking of a desired 6-DOF task space trajectory using the developed DOB-based controller. A modification of the DOB-based controller is carried out to use the direct measurements from the onboard sensors in order to reduce the disturbance estimation error.

11. Optimum performance while keeping the control signal within the allowable limits is achieved by developing a robust optimal control based on the DOB and MPC approaches. Numerical results show that the control efforts are reduced and thus the mission time is increased.

12. A robust sensorless force estimation and impedance control scheme is developed based on the FTRLs and DOB techniques, which takes into consideration all the system dynamics with no need for the environment contact indication. The efficiency of the proposed control systems is validated via numerical results.
6.2 Future Plans

Following are some of the problems for future research work:

- Increasing the accuracy of the measurement of the position in $z$-axis. This can be done by replacing the sonar with a downward laser beams.

- Repeat the experiment with more sophisticated tasks indoors and in the field outdoors.

- In addition to the laser ranger, vision based localization and navigation scheme can be built to improve the motion accuracy both indoors and specially outdoors.

- Robust state estimation to increase the accuracy of the state estimation.

- Experimental implementation of the inverse kinematics, PD-DOb, MPC-DOb, and FTRLS-DOb algorithms to verify their efficiency in real time.

- Autonomous motion for picking and placing an object or implementing specific task.
6. CONCLUSIONS
LIST OF PUBLICATIONS

Conference Papers

1. **Ahmed Khalifa**, Mohamed Fanni, Ahmed Ramadan, and Ahmed Abo-Ismail, “Controller Design of a New Quadrotor Manipulation System Based on Robust Internal-loop Compensator”, in *the IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, Vila Real, Portugal, April 8-10, 2015, pp. 97-102.

2. **Ahmed Khalifa**, Mohamed Fanni, and Toru Namerikawa, “MPC and DOb-based Robust Optimal Control of a New Quadrotor Manipulation System”, in *the European Control Conference (ECC)*, Aalborg, Denmark, June 29 - July 1, pp. 483-488.

3. **Ahmed Khalifa**, Mohamed Fanni, and Toru Namerikawa, “On the Fault Tolerant Control of a Quadrotor Manipulation System via MPC and DOb Approaches”, in *the 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, Tsukuba, Japan, September 20-23, 2016.

4. **Ahmed Khalifa**, Mohamed Fanni, and Toru Namerikawa, “Hybrid Acceleration/Velocity-based Disturbance Observer for a Quadrotor Manipulation System”, in *the 2016 IEEE Multi-Conference on Systems and Control (MSC)*, *International Conference on Control Applications (CCA)*, Buenos Aires, Argentina, September 19-22, 2016.

Journal Papers

1. **Ahmed Khalifa** and Mohamed Fanni, “Position Inverse Kinematics and Robust Internal-loop Compensator-based Control of a New Quadrotor Manipulation System”, in *the International Journal of Imaging and Robotics*, 16.1 (2016): 94-113.

2. **Ahmed Khalifa** and Mohamed Fanni, “POSITION ANALYSIS AND CONTROL OF A NEW QUADROTOR MANIPULATION SYSTEM”, in *the International Journal of Robotics and Automation*, (Accepted for publication).
3. Ahmed Khalifa and Mohamed Fanni, “A New Quadrotor Manipulation System: Modeling and Point-to-Point Task Space Control”, in *the International Journal of Control, Automation and Systems*, (Accepted for publication).
REFERENCES

[1] Asctec Pelican Quadrotor, Oct. 2014. Available at http://www.asctec.de/en/uav-uas-drone-products/asctec-pelican/.

[2] Hokuyo URG-04LX Laser Range Finder, Oct. 2014. Available at https://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx.html/.

[3] Y. Okubo, C. Ye, and J. Borenstein, “Characterization of the hokuyo urg-04lx laser rangefinder for mobile robot obstacle negotiation,” in SPIE Defense, Security, and Sensing, pp. 733212–733212, International Society for Optics and Photonics, 2009.

[4] Devantech SRF04 Ultrasonic Range Finder, Oct. 2014. Available at https://nodna.de/Devantech-SRF04-Ultrasonic-Range-Finder

[5] Multi-axis load cell, Oct. 2014. Available at http://www.jr3.com/.

[6] Typical Quadrotor, Oct. 2014. Available at http://www.life-prog.ru/2_88206_sistema-magnitnoy-levitatsii.html

[7] D. Mellinger, Q. Lindsey, M. Shomin, and V. Kumar, “Design, modeling, estimation and control for aerial grasping and manipulation,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2668–2673, IEEE, 2011.

[8] M. Bisgaard, A. la Cour-Harbo, and J. Dimon Bendtsen, “Adaptive control system for autonomous helicopter sling load operations,” Control Engineering Practice, vol. 18, no. 7, pp. 800–811, 2010.
[9] J. Thomas, J. Polin, K. Sreenath, and V. Kumar, “Avian-inspired grasping for quadrotor micro uavs,” in *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. V06AT07A014–V06AT07A014, American Society of Mechanical Engineers, 2013.

[10] M. Orsag, C. Korpela, and P. Oh, “Modeling and control of mm-uav: Mobile manipulating unmanned aerial vehicle,” *Journal of Intelligent & Robotic Systems*, vol. 69, no. 1-4, pp. 227–240, 2013.

[11] G. Heredia, A. Jimenez-Cano, I. Sanchez, D. Llorente, V. Vega, J. Braga, J. Acosta, and A. Ollero, “Control of a multirotor outdoor aerial manipulator,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, pp. 3417–3422, IEEE, 2014.

[12] S. Kim, S. Choi, and H. J. Kim, “Aerial manipulation using a quadrotor with a two dof robotic arm,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4990–4995, IEEE, 2013.

[13] A. Khalifa, M. Fanni, A. Ramadan, and A. Abo-Ismail, “Modeling and control of a new quadrotor manipulation system,” in *Proceedings of the IEEE/RAS International Conference on Innovative Engineering Systems*, pp. 109–114, IEEE, 2012.

[14] *LYNXMOTION*, Oct. 2014. Available at [http://www.lynxmotion.com/default.aspx](http://www.lynxmotion.com/default.aspx).

[15] *Arduino Board*, Oct. 2014. Available at [http://store.arduino.cc/product/A000067](http://store.arduino.cc/product/A000067).

[16] *12 V to 5 V 1.5 A DC-DC converter*, Oct. 2014. Available at [http://www.simple-electronics.com/2009/11/12v-to-5v-dc-converter-using-7805.html](http://www.simple-electronics.com/2009/11/12v-to-5v-dc-converter-using-7805.html).

[17] S. M. Weiss, *Vision based navigation for micro helicopters*. PhD thesis, Citeseer, 2012.
[18] Global Gemining Efforts, Oct. 2014. Available at https://www.army.mil/article/83702/Researchers__officials_collaborate_for_global_demining_efforts

[19] Difficult Terrains, Oct. 2014. Available at https://www.jalamteas.com/pages/pulang-old-tree-tea

[20] Robots at Work and Play, Oct. 2014. Available at http://pixtale.net/2011/04/robots-at-work-and-play/

[21] S. Gupte, P. I. T. Mohandas, and J. M. Conrad, “A survey of quadrotor unmanned aerial vehicles,” in Southeastcon, 2012 Proceedings of IEEE, pp. 1–6, IEEE, 2012.

[22] Q. Lindsey, D. Mellinger, and V. Kumar, “Construction with quadrotor teams,” Autonomous Robots, vol. 33, no. 3, pp. 323–336, 2012.

[23] J. Willmann, F. Augugliaro, T. Cadalbert, R. D’Andrea, F. Gramazio, and M. Kohler, “Aerial robotic construction towards a new field of architectural research,” International journal of architectural computing, vol. 10, no. 3, pp. 439–460, 2012.

[24] A. Albers, S. Trautmann, T. Howard, T. A. Nguyen, M. Frietsch, and C. Sauter, “Semi-autonomous flying robot for physical interaction with environment,” in 2010 IEEE Conference on Robotics Automation and Mechatronics (RAM), pp. 441–446, IEEE, 2010.

[25] N. Michael, J. Fink, and V. Kumar, “Cooperative manipulation and transportation with aerial robots,” Autonomous Robots, vol. 30, no. 1, pp. 73–86, 2011.

[26] F. A. Goodarzi, D. Lee, and T. Lee, “Geometric control of a quadrotor uav transporting a payload connected via flexible cable,” International Journal of Control, Automation and Systems, vol. 13, no. 6, pp. 1486–1498, 2015.

[27] F. A. Goodarzi and T. Lee, “Dynamics and control of quadrotor uavs transporting a rigid body connected via flexible cables,” in American Control Conference (ACC), 2015, pp. 4677–4682, IEEE, 2015.
REFERENCES

[28] C. M. Korpela, T. W. Danko, and P. Y. Oh, “Mm-uav: Mobile manipulating unmanned aerial vehicle,” Journal of Intelligent & Robotic Systems, vol. 65, no. 1-4, pp. 93–101, 2012.

[29] V. Lippiello and F. Ruggiero, “Cartesian impedance control of a uav with a robotic arm,” in 10th International IFAC Symposium on Robot Control, pp. 704–709, 2012.

[30] V. Lippiello and F. Ruggiero, “Exploiting redundancy in cartesian impedance control of uavs equipped with a robotic arm,” in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3768–3773, IEEE, 2012.

[31] F. Ruggiero, M. Trujillo, R. Cano, H. Ascorbe, A. Viguria, C. Peréz, V. Lippiello, A. Ollero, and B. Siciliano, “A multilayer control for multirotor uavs equipped with a servo robot arm,”

[32] R. Cano, C. Pérez, F. Pruaño, A. Ollero, and G. Heredia, “Mechanical design of a 6-dof aerial manipulator for assembling bar structures using uavs,” RED-UAS, 2013.

[33] A. Khalifa, M. Fanni, A. Ramadan, and A. Abo-Ismail, “Adaptive intelligent controller design for a new quadrotor manipulation system,” in 2013 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 1666–1671, IEEE, 2013.

[34] Y.-C. Choi and H.-S. Ahn, “Nonlinear control of quadrotor for point tracking: Actual implementation and experimental tests,” IEEE/ASME Transactions on Mechatronics, 2015.

[35] V. Ghadiok, J. Goldin, and W. Ren, “On the design and development of attitude stabilization, vision-based navigation, and aerial gripping for a low-cost quadrotor,” Autonomous Robots, vol. 33, no. 1-2, pp. 41–68, 2012.

[36] F. Ruggiero, M. Trujillo, R. Cano, H. Ascorbe, A. Viguria, C. Peréz, V. Lippiello, A. Ollero, and B. Siciliano, “A multilayer control for multirotor uavs equipped with a servo robot arm,” in 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 4014–4020, IEEE, 2015.
REFERENCES

[37] K. Baizid, G. Giglio, F. Pierri, M. Trujillo, G. Antonelli, F. Caccavale, A. Viguria, S. Chiaverini, and A. Ollero, “Experiments on behavioral coordinated control of an unmanned aerial vehicle manipulator system,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 4680–4685, IEEE, 2015.

[38] B. K. Kim and W. K. Chung, “Performance tuning of robust motion controllers for high-accuracy positioning systems,” *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 4, pp. 500–514, 2002.

[39] S. Li, J. Yang, W.-h. Chen, and X. Chen, *Disturbance observer-based control: methods and applications*. CRC Press, 2014.

[40] K. Yamada, S. Komada, M. Ishida, and T. Hori, “Analysis of servo system realized by disturbance observer,” in *Proceedings., 1996 4th International Workshop on Advanced Motion Control, 1996. AMC’96-MIE.*, vol. 1, pp. 338–343, IEEE, 1996.

[41] H.-T. Choi, S. Kim, J. Choi, Y. Lee, T.-J. Kim, and J.-W. Lee, “A simplified model based disturbance rejection control for highly accurate positioning of an underwater robot,” in *Oceans-St. John’s, 2014*, pp. 1–5, IEEE, 2014.

[42] T. Hsia, T. Lasky, and Z. Guo, “Robust independent robot joint control: design and experimentation,” in *Proceedings., 1988 IEEE International Conference on Robotics and Automation, 1988.*, pp. 1329–1334, IEEE, 1988.

[43] T. Murakami, F. Yu, and K. Ohnishi, “Torque sensorless control in multidegree-of-freedom manipulator,” *IEEE Transactions on Industrial Electronics*, vol. 40, no. 2, pp. 259–265, 1993.

[44] X.-y. Wang and Y.-j. Pi, “Trajectory tracking control of a hydraulic parallel robot manipulator with lumped disturbance observer,” *International Journal of Robotics and Automation*, vol. 28, no. 2, 2013.

[45] M. Chen, B. Jiang, and R.-X. Cui, “Robust control for rigid robotic manipulators using nonlinear disturbance observer,” *International Journal of Robotics and Automation*, vol. 29, no. 3, 2014.
REFERENCES

[46] K. S. Eom, I. H. Suh, and W. K. Chung, “Disturbance observer based path tracking control of robot manipulator considering torque saturation,” Mechatronics, vol. 11, no. 3, pp. 325–343, 2001.

[47] E. Sariyildiz, H. Yu, K. Yu, and K. Ohnishi, “A nonlinear stability analysis for the robust position control problem of robot manipulators via disturbance observer,” in Mechatronics (ICM), 2015 IEEE International Conference on, pp. 28–33, IEEE, 2015.

[48] E. Sariyildiz and K. Ohnishi, “Stability and robustness of disturbance-observer-based motion control systems,” IEEE Transactions on Industrial Electronics, vol. 62, no. 1, pp. 414–422, 2015.

[49] T. Tomic and S. Haddadin, “A unified framework for external wrench estimation, interaction control and collision reflexes for flying robots,” in Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on, pp. 4197–4204, IEEE, 2014.

[50] T. Hatanaka, T. Yamada, M. Fujita, S. Morimoto, and M. Okamoto, “Nonlinear model predictive control towards new challenging applications,” Nonlinear Model Predictive Control Towards New Challenging Applications, vol. 384, pp. 561–569, 2009.

[51] G. Pin, D. M. Raimondo, L. Magni, and T. Parisini, “Robust model predictive control of nonlinear systems with bounded and state-dependent uncertainties,” Automatic Control, IEEE Transactions on, vol. 54, no. 7, pp. 1681–1687, 2009.

[52] A. Ferrara, G. P. Incremona, and L. Magni, “A robust mpc/ism hierarchical multi-loop control scheme for robot manipulators,” in Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on, pp. 3560–3565, IEEE, 2013.

[53] C. Liu, W. Chen, and J. Andrews, “Trajectory tracking of small helicopters using explicit nonlinear mpc and dobc,” in 18th IFAC World Congress, IFAC WC, Milano, Italy, vol. 18, pp. 1498–1503, 2011.

[54] H. Seraji and R. Colbaugh, “Force tracking in impedance control,” The International Journal of Robotics Research, vol. 16, no. 1, pp. 97–117, 1997.
[55] L. J. Love and W. J. Book, “Environment estimation for enhanced impedance control,” in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 2, pp. 1854–1859, IEEE, 1995.

[56] S. K. Singh and D. O. Popa, “An analysis of some fundamental problems in adaptive control of force and impedance behavior: Theory and experiments,” *Robotics and Automation, IEEE Transactions on*, vol. 11, no. 6, pp. 912–921, 1995.

[57] F. Forte, R. Naldi, A. Macchelli, and L. Marconi, “Impedance control of an aerial manipulator,” in *American Control Conference (ACC), 2012*, pp. 3839–3844, IEEE, 2012.

[58] V. Lipiello and F. Ruggiero, “Cartesian impedance control of a uav with a robotic arm,” in *10th IFAC Symposium on Robot Control*, 2012.

[59] F. Caccavale, P. Chiacchio, A. Marino, and L. Villani, “Six-dof impedance control of dual-arm cooperative manipulators,” *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 5, pp. 576–586, 2008.

[60] N. Hogan, “Stable execution of contact tasks using impedance control,” in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4, pp. 1047–1054, IEEE, 1987.

[61] S. Misra and A. M. Okamura, “Environment parameter estimation during bilateral telemanipulation,” in *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2006 14th Symposium on*, pp. 301–307, IEEE, 2006.

[62] K. Hashtrudi-Zaad and S. E. Salcudean, “Adaptive transparent impedance reflecting teleoperation,” in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2, pp. 1369–1374, IEEE, 1996.

[63] E. Sariyildiz and K. Ohnishi, “A guide to design disturbance observer based motion control systems,” in *2014 International Power Electronics Conference (IPEC-Hiroshima 2014-ECCE-ASIA)*, pp. 2483–2488, IEEE, 2014.
REFERENCES

[64] T. Murakami, R. Nakamura, F. Yu, and K. Ohnishi, “Force sensorless impedance control by disturbance observer,” in *Power Conversion Conference, 1993. Yokohama 1993.*, *Conference Record of the*, pp. 352–357, IEEE, 1993.

[65] K. S. Eom, I. H. Suh, W. K. Chung, and S.-R. Oh, “Disturbance observer based force control of robot manipulator without force sensor,” in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 4, pp. 3012–3017, IEEE, 1998.

[66] M. Van Damme, P. Beyl, B. Vanderborght, V. Grosu, R. Van Ham, I. Vanderniepen, A. Matthys, and D. Lefeber, “Estimating robot end-effector force from noisy actuator torque measurements,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 1108–1113, IEEE, 2011.

[67] L. D. Phong, J. Choi, and S. Kang, “External force estimation using joint torque sensors for a robot manipulator,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 4507–4512, IEEE, 2012.

[68] A. Colomé, D. Pardo, G. Alenya, and C. Torras, “External force estimation during compliant robot manipulation,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 3535–3540, IEEE, 2013.

[69] A. Alcocera, A. Robertsson, A. Valerac, and R. Johanssona, “Force estimation and control in robot manipulators,” in *Robot Control 2003 (SYROCO’03): A Proceedings Volume from the 7th IFAC Symposium, Wroclaw, Poland, 1-3 September 2003*, vol. 1, p. 55, International Federation of Automatic Control, 2004.

[70] F. Ruggiero, J. Cacace, H. Sadeghian, and V. Lippiello, “Impedance control of vtol uavs with a momentum-based external generalized forces estimator,” in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 2093–2099, IEEE, 2014.

[71] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*, vol. 3. Wiley New York, 2006.
[72] L.-W. Tsai, *Robot analysis: the mechanics of serial and parallel manipulators*. Wiley-Interscience, 1999.

[73] ROS, “Robot Operation System.” [http://wiki.ros.org/](http://wiki.ros.org/), 2014. [Online; accessed 19-Oct-2014].

[74] I. Dryanovski, R. G. Valenti, and J. Xiao, “An open-source navigation system for micro aerial vehicles,” *Autonomous Robots*, vol. 34, no. 3, pp. 177–188, 2013.

[75] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart, “Onboard imu and monocular vision based control for mavs in unknown in-and outdoor environments,” in *2011 IEEE international conference on Robotics and automation (ICRA)*, pp. 3056–3063, IEEE, 2011.

[76] P. J. From, J. T. Gravdahl, and K. Y. Pettersen, *Vehicle-Manipulator Systems*. Springer, 2014.

[77] N. Sadegh and R. Horowitz, “Stability and robustness analysis of a class of adaptive controllers for robotic manipulators,” *The International Journal of Robotics Research*, vol. 9, no. 3, pp. 74–92, 1990.

[78] R. Bickel and M. Tomizuka, “Passivity-based versus disturbance observer based robot control: equivalence and stability,” *Journal of dynamic systems, measurement, and control*, vol. 121, no. 1, pp. 41–47, 1999.

[79] C. A. Desoer and M. Vidyasagar, *Feedback systems: input-output properties*, vol. 55. Siam, 2009.

[80] P. A. Ioannou and J. Sun, *Robust adaptive control*. Courier Corporation, 2012.

[81] M.-D. Hua, T. Hamel, P. Morin, and C. Samson, “Introduction to feedback control of underactuated vtol vehicles: A review of basic control design ideas and principles,” *Control Systems, IEEE*, vol. 33, no. 1, pp. 61–75, 2013.

[82] A. Isidori, L. Marconi, and A. Serrani, *Robust autonomous guidance: an internal model approach*. Springer Science & Business Media, 2012.
[83] L. Wang, *Model predictive control system design and implementation using MATLAB®*. Springer Science & Business Media, 2009.

[84] L. Wang, “Continuous time model predictive control design using orthonormal functions,” *International Journal of Control*, vol. 74, no. 16, pp. 1588–1600, 2001.

[85] T. Barry and L. Wang, “A model-free predictive controller with laguerre polynomials,” in *Control Conference, 2004. 5th Asian*, vol. 1, pp. 177–184, IEEE, 2004.

[86] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, “Constrained model predictive control: Stability and optimality,” *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[87] D. A. Wismer and R. Chattergy, *Introduction to nonlinear optimization: a problem solving approach*. North-Holland New York, 1978.

[88] Z. Y. A. S. I. O. Solovyev Viktor V., Finaev Valery I. and B. D. A., “Simulation of wind effect on a quadrotor flight,” *ARPN Journal of Engineering and Applied Sciences*, vol. 10, pp. 1535–1538, 2015.

[89] S. Hsu, “Verifying wind profile equations under hurricane conditions,” *Open Ocean Engineering Journal*, vol. 4, pp. 60–64, 2011.

[90] P. L. Andrews et al., “Modeling wind adjustment factor and midflame wind speed for rothermel’s surface fire spread model,” 2012.

[91] H. Hu and R. Ding, “Least squares based iterative identification algorithms for input nonlinear controlled autoregressive systems based on the auxiliary model,” *Nonlinear Dynamics*, vol. 76, no. 1, pp. 777–784, 2014.

[92] Y. Wang and F. Ding, “Recursive least squares algorithm and gradient algorithm for hammerstein–wiener systems using the data filtering,” *Nonlinear Dynamics*, pp. 1–9, 2015.

[93] *Robotic Gripper for Robotic Arm*, Oct. 2014. Available at [http://robokits.co.in](http://robokits.co.in).
Appendix A

Quadrotor Platform

We use a quadrotor from Ascending Technologies (AscTec) of type Pelican that is shown in Fig. A.1.

Figure A.1: AscTec Pelican quadrotor with full options
A. QUADROTOR PLATFORM

The propellers are mounted as shown in Fig. A.2

![Image of quadrotor with propellers](image)

**Figure A.2:** Mounting the propellers

Table A.1 presents the technical specification for Pelican quadrotor from AscTec.

**Table A.1: AscTec Pelican, technical Data**

| Dimensions     | 65.1 × 65.1 × 18.8 cm |
|----------------|-----------------------|
| Propeller size | 10 inch               |
| Motors         | 4 × 160 W electrical, brushless (sensorless) |
| Max. thrust    | 36 N                  |
| Max. payload   | 650 g                 |
| Max. total weight | 1650 g              |
| Max. airspeed  | 16 m/s                |
| Max. flight time | 30 mins (without payload) |
| Range          | 1 km                  |
| Battery        | 6250 mAh (LiPo)       |

The quadrotor dimensions are given in Fig. A.3

Fig. A.4 illustrates functional block diagram of the FCU. This flight control unit contains all necessary sensors to function as an IMU, it also has two onboard ARM7 microprocessors
and various communication interfaces.

It can directly interpret the R/C data sent by the pilot (R/C) and the serial data sent from the ground PC. For automatic missions, you can program and upload our control algorithms directly onto the High Level processor of the UAV.

The LLP handles sensor data processing, data fusion as well as our fast and stable attitude control algorithm with an update rate of 1000 Hz. It also processes the position control algorithm, using the onboard magnetometer and GPS module as additional sensor inputs. Sensor data can be retrieved from the LLP via a serial interface using a predefined serial protocol. Furthermore, the serial interface can be used to send attitude commands (pitch angle, roll angle, yaw rate and thrust) or even waypoint commands to the vehicle depending on the flight mode. The data structures from and to the LLP are
A. QUADROTOR PLATFORM

Figure A.4: AscTec AutoPilot functional block diagram

- LLP STATUS: Status information of the low level processor.
- IMU RAWDATA: Sensor raw-values.
- IMU CALCDATA: Calibrated sensor outputs and data fusion results.
- CTRL OUT: Controller outputs.
- RC Data: Data received by the R/C receiver.
- GPS Data: GPS data.
- GPS Data Advanced: GPS data + position and speed estimates after data fusion.

The HLP can be flashed with a custom algorithm and let it control the flight system. To recover from critical flight situations during the test flights, the pilot can always switch back to the well proven control algorithms on the LLP as a safety backup.
The two microprocessors communicate with an extremely fast rate of 1000 Hz, hence all sensor data is available to the HLP as well. Control commands can be sent back to the LLP at the same frequency, for example the: rotational speed commands for each individual motor, pitch/roll/yaw/thrust commands, attitude commands or waypoint commands. The HLP offers UART (Universal Asynchronous Receiver Transmitter), SPI (Serial Peripheral Interface) and I2C (Inter-Integrated Circuit) interfaces as well as simple port I/Os to connect other devices like additional sensors, servo motors or extension boards. Also custom payloads, can be powered by a 5 V or 12 V supply of the AscTec AutoPilot.

Fig. A.5 shows the pinout and connections of the AscTec AutoPilot board.

![AscTec AutoPilot board pinout and connections](image)

**Figure A.5:** AscTec AutoPilot board pinout and connections [1]

Fig. A.6 shows the pinout and connections of the AscTec Power Board.
The AscTec Atomboard is the perfect choice for an x86-technology computer packed into a minimum form factor. By communicating with the AscTec Autopilot, IMU data can be included into your algorithms and high level commands can be sent to the flight system. Table A.2 presents the technical data of this onboard computer.

Fig. A.7 shows the top view of pinout and connections of the AscTec Atomboard (onboard computer).

Fig. A.8 shows the bottom view of pinout and connections of the AscTec Atomboard (onboard computer).
Table A.2: AscTec Atomboard technical specifications

| Specification    | Value                        |
|------------------|------------------------------|
| CPU              | Intel Atom Processor Z530    |
| CPU L2 Cache     | 512k                         |
| Number of cores  | 1                            |
| Clock Speed      | 1.6GHz                       |
| RAM              | 1GB DDR2                     |
| GPU              | Intel GMA500                 |
| Hard disk        | 8GB SD Card                  |
| Display connector| LVDS                        |
| USB2.0           | 7x mini-USB A/B              |
| UART             | 2x                           |

Figure A.7: Top view of the onboard computer pinout and connections
Figure A.8: Bottom view of the onboard computer pinout and connections [1]
Appendix B

Robotic Arm

Starting with design of the gripper mechanism, a little gripper (Fig. B.1), from “LYNXMOTION”, is used. It is made from injection molded ABS. This gripper can open to 3.3 cm. It has a driving motor ($mtr_2$), see Fig. B.2, of type servo motor HS-422. This gripper can carry a payload of 200 g [93].

![Lightweight gripper](image)

**Figure B.1:** Lightweight gripper [14]

The total length of the arm is chosen such that it can provide enough work space. If the length is increased, then a motor with larger output torque is required, and thus larger weight and more consumed power are resulted.

Based on this design, the required specifications of the selected joints motors are set from Fig. B.2.

The torques, $T_{mtr_0}$ and $T_{mtr_1}$, of motors, $mtr_0$ and $mtr_1$, respectively, see Fig. B.2, can
where \( W_e \) is the weight from the end effector and the payload that it has mass of \( m_p \).

The average weight of the available motors in the light categories is 50 g (including the arm accessories that will be described next).

Therefore, from the values of lengths and masses shown in Fig. B.2, the value of \( T_{mtr_0} \) is 0.45 N.m and \( T_{mtr_1} \) is 0.26 N.m (@ payload \( m_p = 200 \) g). Multiplying this value by factor of 1.2 for safety, the selected motors should have output torque at least of 0.55 N.m for \( mtr_0 \) and 0.31 N.m for \( mtr_1 \).

A digital standard servo (HS-5485HB), see Fig. B.3, is used to provide the rotational motion (Revolute Joint) for joint 1. It has a rotational range of 180°, weight of 45 g, speed of 350°/s, and operating voltage of 6 V with maximum running current of 1 A. It is capable of producing a torque of 0.63 N.m. Another digital standard servo (HS-422) is used for both joint 2 and the gripper. It has a rotational range of 180°, weight of 45 g, speed of 350°/s, operating voltage of 6 V with maximum running current of 0.51 A, and torque of 0.32 N.m [14]. Figs. B.4 and B.5 present the dimensions of motors 1 and 2 respectively.

Fig. B.6 presents the required accessories to build the the two arms of the manipulator. Also, the dimensions of these accessories are given in Fig. B.7. Serial servo controller (SSC-
Figure B.3: HS-5485HB digital servo motor

Figure B.4: HS-5485HB dimensions

Figure B.5: HS-422 dimensions

32) from LYNXMOTION, see Fig. B.8, is a small preassembled servo controller with some big features. It has high resolution (1 µs) for accurate positioning, and extremely smooth
B. ROBOTIC ARM

Figure B.6: Accessories for building the manipulator: a) Aluminum Tubing, b) Servo Bracket, c) Connector Hub, and d) Long “C” Servo Bracket

moves. The range is 0.50 ms to 2.50 ms for a range of about 180°. This board contains a MCU of Atmel ATMEGA168-20PU as well as driver interface between the controller unit and the motors. This driver unit operates at voltage of 12 V. This unit will take its commands from the Arduino MEGA 2560 board. The pin configuration of this board is as following.

1. The Low Dropout regulator will provide 5vdc out with as little as 5.5vdc coming in. This is important when operating your robot from a battery. It can accept a maximum of 9vdc in. The regulator is rated for 500mA, but we are de-rating it to 250mA to
Figure B.7: Dimensions of the accessories for building the manipulator in (mm): a) Aluminum Tubing, b) Servo Bracket, c) Connector Hub, and d) Long “C” Servo Bracket.

prevent the regulator from getting too hot.

2. This terminal connects power to servo channels 16 through 31. Apply 4.8vdc to 6.0vdc for most analog or digital servos. This can be directly from a 5-cell NiMH battery pack. 7.2vdc - 7.4vdc can be applied to HSR-5980 or HSR-5990 servos. This can be directly from a 6-cell NiMH battery pack or a 2-cell LiPo battery pack.

3. These jumpers are used to connect VS1 to VS2. Use this option when you are powering
all servos from the same battery. Use both jumpers. Alternately, if you want to use two separate battery packs, one on each side, then remove both of these jumpers.

4. This is the Logic Voltage, or VL. This input is normally used with a 9vdc battery connector to provide power to the ICs and anything connected to the 5vdc lines on the board. The valid range for this terminal is 6vdc - 9vdc. This input is used to isolate the logic from the Servo Power Input. It is necessary to remove the VS1=VL jumper when powering the servos separately from the logic VL. The SSC-32 should draw 35mA with nothing connected to the 5vdc output.

5. This jumper allows powering the microcontroller and support circuitry from the servo power supply. This requires at least 6vdc to operate correctly. If the microcontroller resets when too many servos are moving at the same time, it may be necessary to power the microcontroller separately using the VL input. A 9vdc works nicely for this. This jumper must be removed when powering the microcontroller separately.

6. This terminal connects power to servo channels 16 through 31. Apply 4.8vdc to 6.0vdc for most analog or digital servos. This can be directly from a 5-cell NiMH battery pack. 7.2vdc - 7.4vdc can be applied to HSR-5980 or HSR- 5990 servos. This can be directly from a 6-cell NiMH battery pack or a 2-cell LiPo battery pack.

7. This is where you connect the servos or other output devices. Use caution and remove power when connecting anything to the I/O bus.
8. This is where the Atmel IC chip goes. Be careful to insert it with Pin 1 in the upper right corner as pictured. Take care to not bend the pins.

9. The two BAUD inputs allow configuring the baud rate.

10. The ABCD inputs have both static and latching support. The inputs have internal weak (50k) pullups that are used when a Read Digital Input command is used. A normally open switch connected from the input to ground will work fine.

11. This is the Processor Good LED. It will light steady when power is applied and will remain lit until the processor has received a valid serial command. It will then go out and will blink whenever it is receiving serial data.

12. Simply plug a straight-through M/F DB9 cable from this plug to a free 9-pin serial port on your PC for receiving servo positioning data. Alternately a USB-to-serial adapter will work well. Note, many USB-to-serial adapters require a separate power supply to work well.

13. This is an 8-pin EEPROM socket.

14. This is the TTL serial port or DB9 serial port enable. Install two jumpers as illustrated below to enable the DB9 port. Install wire connectors to utilize TTL serial communication from a host microcontroller.

Fig. B.9 presents the way to mount the jumpers of the SSC32 board to enable the communication between this board and the Arduino board such that one can send the motor commands from the Arduino to the SSC32 board. One microcontroller that has gotten special attention from the robotics community world-wide is the Arduino. This microcontroller platform is quite inexpensive and has a C-based language development environment that is very intuitive to use. From the different versions of the Arduino, the selected one for this project was the Arduino Mega 2560 (see Fig. B.10). It has 54 digital input/output pins of which 15 can be used as PWM outputs, 16 analog inputs, 4 UARTs (hardware serial ports), 16 MHz crystal oscillator, USB connection, power jack, and 256 KB flash memory for storing code [15]. This board has a open source ROS library.
B. ROBOTIC ARM

Figure B.9: SSC32 board: Shorting bar jumpers to enable serial communication with the Arduino board

Figure B.10: Arduino Mega 2560 board

Fig. B.11 shows the circuit diagram for the 12 V to 5V 1.5 A DC-DC converter that is used to power the manipulator’s motors. This circuit uses the 7805 IC to converts the 12 VDC from the quadrotor battery to the required 5 VDC at 1.5 A.
Figure B.11: 12 V to 5 V 1.5 A DC-DC converter circuit diagram [16]
B. ROBOTIC ARM
Appendix C

Forward Kinematics

function Xe = For_Kin(zt, sb)
L0 = 30e-03;
L1 = 70e-03;
L2 = 85e-03;
X = zt(1);
Y = zt(2);
Z = zt(3);
ep = zt(4);
th1 = zt(5);
th2 = zt(6);
ph = sb(2);
th = sb(1);

%% =================================
A0_B = Transformation matrix from frame 0 to frame B
A0_B = [0 0 1 0; -1 0 0 0; 0 -1 0 -L0; 0 0 0 1];
A1_0 = [cos(th1) 0 sin(th1) L1*cos(th1); sin(th1) 0 -cos(th1) L1*sin(th1);
0 1 0 0; 0 0 0 1];
A2_1 = [cos(th2) -sin(th2) 0 L2*cos(th2); sin(th2) cos(th2) 0 L2*sin(th2);
0 0 1 0; 0 0 0 1];
AB_I = [cos(ep)*cos(th), cos(ep)*sin(ph)*sin(th)-cos(ph)*sin(ep),
sin(ep)*sin(ph)+cos(ep)*cos(ph)*sin(th), X; 
cos(th)*sin(ep), cos(ep)*cos(ph)+sin(ep)*sin(ph)*sin(th),]
C. FORWARD KINEMATICS

\[
\begin{align*}
\cos(\phi) \cdot \sin(\epsilon) \cdot \sin(\theta) - \cos(\epsilon) \cdot \sin(\phi), Y; \\
-\sin(\theta), \cos(\theta) \cdot \sin(\phi), \cos(\phi) \cdot \cos(\theta), Z; \\
0, 0, 0, 1];
\end{align*}
\]

\[
T_2 = A_{B-I} \cdot A_{0-B} \times A_{1-0} \times A_{2-1};
\]

\[
\% T_2 = \text{eval}(T_2_I);
\]

\[
X_e = T_2_I(1,4); \\
y_e = T_2_I(2,4); \\
z_e = T_2_I(3,4);
\]

\[
\% \text{ Computing Euler angles from a rotation matrix}
\]

\[
R_{31} = T_2_I(3,1); \\
R_{32} = T_2_I(3,2); \\
R_{33} = T_2_I(3,3); \\
R_{21} = T_2_I(2,1); \\
R_{11} = T_2_I(1,1); \\
R_{12} = T_2_I(1,2); \\
R_{13} = T_2_I(1,3); \\
\]

\[
\text{if } (R_{31} == +1 \ \&\ \& \ R_{31} == -1)
\]

\[
\text{the} = -\text{asin}(R_{31});
\]

\[
\% \text{ or}
\]

\[
\text{the} = \pi - (-\text{asin}(R_{31}))
\]

\[
\text{phe} = \text{atan2}(R_{32}/\cos(\text{the}),R_{33}/\cos(\text{the}));
\]

\[
\text{epe} = \text{atan2}(R_{21}/\cos(\text{the}),R_{11}/\cos(\text{the}));
\]

\[
\text{else}
\]

\[
\text{epe} = 0; \ % \text{anything; can set to 0}
\]

\[
\text{if } (R_{31} == -1)
\]

\[
\text{the} = \pi/2;
\]

\[
\text{phe} = \text{epe} + \text{atan2}(R_{12}, R_{13});
\]

\[
\text{else}
\]

\[
\text{the} = -\pi/2;
\]

\[
\text{phe} = -\text{epe} + \text{atan2}(-R_{12}, -R_{13});
\]

\[
\text{end}
\]

\[
\text{end}
\]

\[
X_e = [x_e; y_e; z_e; epe; the; phe];
\]

\[
\text{end}
\]
Appendix D

System Jacobian and Parametrization

The coding of the system Jacobian is given as following.

```matlab
function [Jz,Js,TAe] = jacob(Xdde_ref,zt,ztd(sb,sbd,Xe,sbdd,Jzd,Jsd,TAed,Xde)
L0 = 30e-03 ;
L1 = 70e-03;
L2 = 85e-03;
X=zt(1);
Y=zt(2);
Z=zt(3);
ep=zt(4);
th1=zt(5);
th2=zt(6);
ph=sb(2);
th=sb(1);
phd=sbd(2);
thd=sbd(1);
xe=Xe(1);
xe=Xe(2);
ze=Xe(3);
```
epe = Xe(4);
the = Xe(5);
phe = Xe(6);

%% ===================
I3 = eye(3); O3 = zeros(3);

A0 =\[
\begin{bmatrix}
0 & 0 & 1 & 0; \\
-1 & 0 & 0 & 0; \\
0 & -1 & 0 & L0; \\
0 & 0 & 0 & 1;
\end{bmatrix}
\]

A1 =\[
\begin{bmatrix}
\cos(th1) & 0 & \sin(th1) & L1 \cos(th1); \\
\sin(th1) & 0 & -\cos(th1) & L1 \sin(th1); \\
0 & 1 & 0 & 0; \\
0 & 0 & 0 & 1;
\end{bmatrix}
\]

A2 =\[
\begin{bmatrix}
\cos(th2) & -\sin(th2) & 0 & L2 \cos(th2); \\
\sin(th2) & \cos(th2) & 0 & L2 \sin(th2); \\
0 & 0 & 1 & 0; \\
0 & 0 & 0 & 1;
\end{bmatrix}
\]

Rb_I =\[
\begin{bmatrix}
\cos(ep) \cos(th), & \cos(ep) \sin(ph) \sin(th) - \cos(ph) \sin(ep), \\
\sin(ep) \sin(ph) + \cos(ep) \cos(ph) \sin(th), \\
\cos(th) \sin(ep), & \cos(ep) \cos(ph) + \sin(ep) \sin(ph) \sin(th), \\
\cos(ph) \sin(ep) \sin(th) - \cos(ep) \sin(ph); \\
\end{bmatrix}
\]

R0_B = A0_B(1:3,1:3);
R1_0 = A1_0(1:3,1:3);
Rb_I = Rb_I;

a1 = L1; d1 = 0;
a2 = L2; d2 = 0;
z0 = eye(3) * [0; 0; 1]; z1 = R1_0 * [0; 0; 1];
r1_0 = [a1 * cos(th1); a1 * sin(th1); d1];
r2_1 = [a2 * cos(th2); a2 * sin(th2); d2];
p2_2 = [0; 0; 0];
p2_1 = R1_0 * r2_1 + p2_2;
p2_0 = eye(3) * r1_0 + p2_1;
alpha1 = cross(z0, p2_0);
beta1 = z0;
alpha2 = cross(z1, p2_1);
beta2 = z1;
A2_B = A0_B * A1_0 * A2_1;
peb_b = A2_B(1:3, 4);
Jt = [R0_B * alpha1, R0_B * alpha2];
Jr=[R0_B*beta1,R0_B*beta2];
Jeb_b=[Jt;Jr];
Rb=RB_I;
Tb=[0,-sin(ep),cos(ep)*cos(th);
 0,cos(ep),sin(ep)*cos(th);
1,0,-sin(th)];
TAb=[I3,O3,O3,Tb];
Jb=[I3,-skew(Rb*peb_b);O3,I3];
Jeb=[Rb,O3,O3,Rb]*Jeb_b;
JbA=Jb*TAb;
Jet=JbA(:,1:4);
Jz=[Jet,Jeb];
Js=JbA(:,5:6);
Te=[0,-sin(epe),cos(epe)*cos(the);
 0,cos(epe),sin(epe)*cos(the);
1,0,-sin(the)];
TAe=[I3,O3,O3,Te];
end

The details and coding of the regressor matrices, \( Y_w, Y_e, Y_t, Y_i, \) and \( Y_s \), are given as following.

```matlab
% x=sym('x','real'); y=sym('y','real'); z=sym('z','real');
ph=sym('ph','real'); th=sym('th','real'); epd=sym('ep','real');
th1=sym('th1','real'); th2=sym('th2','real');
L0=sym('L0','real'); L1=sym('L1','real'); L2=sym('L2','real');
mb=sym('mb','real'); ml=sym('ml','real'); m2=sym('m2','real');
I1=sym('I1','real'); I2=sym('I2','real'); Ix=sym('Ix','real');
g=sym('g','real');
```
D. SYSTEM JACOBIAN AND PARAMETRIZATION

\[
\begin{align*}
\text{xd} &= \text{sym('xd','real'); yd} = \text{sym('yd','real'); zd} = \text{sym('zd','real');} \\
\text{phd} &= \text{sym('phd','real'); thd} = \text{sym('thd','real'); epd} = \text{sym('epd','real');} \\
\text{thd1} &= \text{sym('thd1','real'); thd2} = \text{sym('thd2','real');} \\
\text{xdd} &= \text{sym('xdd','real'); ydd} = \text{sym('ydd','real'); zdd} = \text{sym('zdd','real');} \\
\text{phdd} &= \text{sym('phdd','real'); thdd} = \text{sym('thdd','real'); epdd} = \text{sym('epdd','real');} \\
\text{thdd1} &= \text{sym('thdd1','real'); thdd2} = \text{sym('thdd2','real');} \\
\end{align*}
\]

\[
\text{TH} = [m1;m2;mb;I1;I2;Ix;L0*m1;L0*m2;L1*m1;L1*m2;L2*m1;L2*m2;L0^2*m1;L0^2*m2;L1^2*m1;L1^2*m2;L2^2*m1;L2^2*m2;L0*L1*m1;L0*L1*m2;L1*L2*m1;L1*L2*m2;L0*L2*m1;L0*L2*m2];
\]

\[
\begin{align*}
\text{Y1_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(1)}; \\
\text{Y2_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(2)}; \\
\text{Y3_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(3)}; \\
\text{Y4_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(4)}; \\
\text{Y5_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(5)}; \\
\text{Y6_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(6)}; \\
\text{Y7_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(7)}; \\
\text{Y8_TH} &= \text{zzz*ones(size(TH'))*TH + Mqdd_Cqd_G(8)}; \\
\end{align*}
\]

\[
\begin{align*}
\text{Y1_TH} &= \text{subs(Y1_TH, [L0*L1*m1,L0*L1*m2,L1*L2*m1,L1*L2*m2,L0*L2*m1,L0*L2*m2,L0^2*m1,L0^2*m2,L1^2*m1,L1^2*m2,L2^2*m1,L2^2*m2,L0*m1,L0*m2,L1*m1,L1*m2,L2*m1,L2*m2]) ...} \\
\text{, [L0L1m1L0L1m2L1L2m1L0L2m1L0L2m2L02m1L02m2L12m1L12m2L22m1L0m1L0m2L1m1L1m2L2m1L2m2]);} \\
\text{[cy1,ty1] = coeffs(Y1_TH, [m1,m2,mb,I1,I2,Ix,L0m1,L0m2,L1m1,L1m2,L2m1,L2m2, L02m1,L02m2,L12m1,L12m2,L22m1,L22m2,L0L1m1,L0L1m2,L1L2m1,L1L2m2,L0L2m1,L0L2m2]);} \\
\end{align*}
\]

\[
\begin{align*}
\text{Y2_TH} &= \text{subs(Y2_TH, [L0*L1*m1,L0*L1*m2,L1*L2*m1,L1*L2*m2,L0*L2*m1,L0*L2*m2,L0^2*m1,L0^2*m2,L1^2*m1,L1^2*m2,L2^2*m1,L2^2*m2,L0*m1,L0*m2,L1*m1,L1*m2,L2*m1,L2*m2]) ...} \\
\text{, [L0L1m1L0L1m2L1L2m1L0L2m1L0L2m2L02m1L02m2L12m1L12m2L22m1L0L1m1L0L1m2L1L2m1,L1L2m2,L0L2m1,L0L2m2]);}
\end{align*}
\]
[cy2, ty2] = coeffs(Y2.TH, [m1, m2, mb, I1, I2, Ix, L0m1, L0m2, L1m1, L1m2, L2m1, L2m2, L02m1, L02m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);

%===================================
Y3.TH = subs(Y3.TH, [L0*L1*m1, L0*L1*m2, L1*L2*m1, L1*L2*m2, L0*L2*m1, L0*L2*m2, L0^2*m1, L0^2*m2, L1^2*m1, L1^2*m2, L2^2*m1, L2^2*m2, L0*m1, L0*m2, L1*m1, L1*m2, L2*m1, L2*m2] ... , [L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);
[cy3, ty3] = coeffs(Y3.TH, [m1, m2, mb, I1, I2, Ix, L0m1, L0m2, L1m1, L1m2, L2m1, L2m2, L02m1, L02m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);

%===================================
Y4.TH = subs(Y4.TH, [L0*L1*m1, L0*L1*m2, L1*L2*m1, L1*L2*m2, L0*L2*m1, L0*L2*m2, L0^2*m1, L0^2*m2, L1^2*m1, L1^2*m2, L2^2*m1, L2^2*m2, L0*m1, L0*m2, L1*m1, L1*m2, L2*m1, L2*m2] ... , [L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);
[cy4, ty4] = coeffs(Y4.TH, [m1, m2, mb, I1, I2, Ix, L0m1, L0m2, L1m1, L1m2, L2m1, L2m2, L02m1, L02m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);

%===================================
Y5.TH = subs(Y5.TH, [L0*L1*m1, L0*L1*m2, L1*L2*m1, L1*L2*m2, L0*L2*m1, L0*L2*m2, L0^2*m1, L0^2*m2, L1^2*m1, L1^2*m2, L2^2*m1, L2^2*m2, L0*m1, L0*m2, L1*m1, L1*m2, L2*m1, L2*m2] ... , [L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);
[cy5, ty5] = coeffs(Y5.TH, [m1, m2, mb, I1, I2, Ix, L0m1, L0m2, L1m1, L1m2, L2m1, L2m2, L02m1, L02m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);

%===================================
Y6.TH = subs(Y6.TH, [L0*L1*m1, L0*L1*m2, L1*L2*m1, L1*L2*m2, L0*L2*m1, L0*L2*m2, L0^2*m1, L0^2*m2, L1^2*m1, L1^2*m2, L2^2*m1, L2^2*m2, L0*m1, L0*m2, L1*m1, L1*m2, L2*m1, L2*m2] ... , [L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2, L12m1, L12m2, L22m1, L22m2, L0L1m1, L0L1m2, L1L2m1, L1L2m2, L0L2m1, L0L2m2]);
D. SYSTEM JACOBIAN AND PARAMETRIZATION

\[ [cy_6, ty_6] = \text{coeffs}(Y_6, \{m_1, m_2, mb, I_1, I_2, Ix, L_{0m1}, L_{0m2}, L_{1m1}, L_{1m2}, L_{2m1}, L_{2m2}, L_{02m1}, L_{02m2}, L_{12m1}, L_{12m2}, L_{22m1}, L_{22m2}, L_{0L1m1}, L_{0L1m2}, L_{1L2m1}, L_{1L2m2}, L_{0L2m1}, L_{0L2m2}\}); \]
\[ \text{coeffs}(Y_7, \{m_1, m_2, mb, I_1, I_2, Ix, L_{0m1}, L_{0m2}, L_{1m1}, L_{1m2}, L_{2m1}, L_{2m2}, L_{02m1}, L_{02m2}, L_{12m1}, L_{12m2}, L_{0L1m1}, L_{0L1m2}, L_{1L2m1}, L_{1L2m2}, L_{0L2m1}, L_{0L2m2}\}); \]
\[ \text{coeffs}(Y_8, \{m_1, m_2, mb, I_1, I_2, Ix, L_{0m1}, L_{0m2}, L_{1m1}, L_{1m2}, L_{2m1}, L_{2m2}, L_{02m1}, L_{02m2}, L_{12m1}, L_{12m2}, L_{0L1m1}, L_{0L1m2}, L_{1L2m1}, L_{1L2m2}, L_{0L2m1}, L_{0L2m2}\}); \]

\[ Yi = [cy_1; cy_2; cy_3; cy_4; cy_5; cy_6; cy_7; cy_8]; \]

\[ \text{syms } xe \ yd e \ ze \ phe \ the \ epe \]
\[ \text{syms } xde \ yde \ zde \ phde \ thde \ epde \]
\[ \text{syms } Kc_1 \ Mc_1 \ Kc_2 \ Mc_2 \ Kc_3 \ Mc_3 \ Kc_4 \ Mc_4 \ Kc_5 \ Mc_5 \ Kc_6 \ Mc_6 \]

\[ Xe=[xe; ye; ze; epe; the; phe]; \]
\[ Xde=[xdde; yde; zde; epde; thde; phde]; \]

\[ \text{Fe}._{\text{reg}} = [\text{Xe}(1), \text{Xde}(1), \text{zeros}(1, 10)]; \]
zeros(1,2), Xe(2), Xde(2), zeros(1,8);
zeros(1,4), Xe(3), Xde(3), zeros(1,6);
zeros(1,6), Xe(4), Xde(4), zeros(1,4);
zeros(1,8), Xe(5), Xde(5), zeros(1,2);
zeros(1,10), Xe(6), Xde(6)];

85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101

Y_l = JT*Fe_reg;

%% ========= Yw ============

D_ex_reg1 = [z^2*sin(th), z^2*cos(th), 0, 0];
D_ex_reg2 = [0, 0, z^2*sin(ph), z^2*cos(ph)];
D_ex_reg3 = zeros(1,4);
D_ex_reg4 = zeros(1,4);
D_ex_reg5 = zeros(1,4);
D_ex_reg6 = zeros(1,4);
D_ex_reg7 = zeros(1,4);
D_ex_reg8 = zeros(1,4);

%% ========= Yl ============

JT_Fe_reg1 = [ xe, xde, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ];
JT_Fe_reg2 = [ 0, 0, ye, yde, 0, 0, 0, 0, 0, 0, 0, 0 ];
JT_Fe_reg3 = [ 0, 0, 0, 0, ze, zde, 0, 0, 0, 0, 0, 0 ];
JT_Fe_reg4 = [ xe*(cos(ep)*sin(ph) - cos(ph)*sin(ep)*sin(th))*L0 + L1*sin(th1) + L2*cos(th2)*sin(th1) - cos(th1)*(L1 + L2*cos(th2))*cos(ph) + sin(ep)*sin(ph)*sin(th) + L2*cos(th)*sin(ep)*sin(th2)), -xde*((cos(ep)*sin(ph) - cos(ph)*sin(ep)*sin(th))*(L0 + L1*sin(th1) + L2*cos(th2)*sin(th1)) - cos(th1)*(L1 + L2*cos(th2))*cos(ph) + sin(ep)*sin(ph)*sin(th) + L2*cos(th)*sin(ep)*sin(th2)), ye*(cos(th1)*(L1 + L2*cos(th2))*cos(ph)*sin(th) - (sin(ep)*sin(ph) + cos(ep)*cos(ph)*sin(th))*L0 + L1*sin(th1) + L2*cos(th2)*sin(th1)) + L2*cos(ep)*cos(th)*sin(th2)), yde*(cos(th1)*(L1 + L2*cos(th2))*cos(ep)*cos(ph)*sin(th) - (sin(ep)*sin(ph) + cos(ep)*cos(ph)*sin(th))*L0 + L1*sin(th1) + L2*cos(th2)*sin(th1)) + L2*cos(ep)*cos(th)*sin(th2)), 0, 0, 0, 0, 0, 0, phe, phde];
D. SYSTEM JACOBIAN AND PARAMETRIZATION

\[
\begin{aligned}
L_2 \cdot \cos(\theta_2) \cdot \sin(\theta_1)) + \cos(\theta) \cdot \cos(\theta_1) \cdot \sin(\phi) \cdot (L_1 + L_2 \cdot \cos(\theta_2)) \), -ye * \sin(\phi) * (L_1 + L_2 \cdot \cos(\theta_2)))
\end{aligned}
\]

\[
\begin{aligned}
\text{JT}_F\text{e}_{\text{reg}6} = \{ xe^{*}(\sin(\theta_1) \cdot ((\cos(\phi) \cdot \sin(\phi) - \cos(\phi) \cdot \sin(\theta_1) \cdot \sin(\phi) \cdot \sin(\theta_1)) - \cos(\phi) \cdot \sin(\theta_1) \cdot \sin(\phi) \cdot \sin(\theta_1)))} \\
\text{JT}_F\text{e}_{\text{reg}7} = \{ -xe^{*}(L_1 \cdot \sin(\theta_1) + L_2 \cdot \cos(\theta_2) \cdot \sin(\theta_1) \cdot \cos(\phi) \cdot \sin(\phi) - (L_1 + L_2 \cdot \cos(\theta_2))) \cdot \cos(\phi) - \sin(\phi) \cdot \sin(\phi) \cdot \sin(\theta_1) - \cos(\phi) \cdot \sin(\phi) \cdot \sin(\theta_1)] \}
\end{aligned}
\]
(ep)•sin(ph)•sin(th)) + (L1•cos(th1) + L2•cos(th1)•cos(th2))•(sin(ep)•sin(ph) + cos(ep)•cos(ph)•sin(th) + cos(ep)•cos(ph) + sin(ep)•sin(ph)•sin(th)) + (L1•cos(th1) + L2•cos(th1)•cos(th2))•(sin(ep)•sin(ph) + cos(ep)•cos(ph)•sin(th))), -xde•((L1•sin(th1) + L2•cos(th2)•sin(th1) + L2•cos(th2)•sin(th1))•(cos(ep)•cos(ph) + sin(ep)•sin(ph)•sin(th)) + (L1•cos(th1) + L2•cos(th1)•cos(th2))•(cos(ep)•sin(ph) - cos(ph)•sin(ep)•sin(th)))), ye•((L1•sin(th1) + L2•cos(th2)•sin(th1) + L2•cos(th2)•sin(th1))•(cos(ep)•cos(ph) + sin(ep)•sin(ph)•sin(th))), -ze•cos(ph + th1)•cos(th)•(L1 + L2•cos(th2)), -zde•cos(ph + th1)•cos(th)•(L1 + L2•cos(th2)), epe•cos(ep)•cos(th), epde•cos(ep)•cos(th), the•cos(ph) • sin(th), thde•cos(ph)•sin(th), -phe•sin(th), -phde•sin(th));

JT_\text{Fe\_reg8}=[ xe•(L2•cos(ep)•cos(th)•cos(th2) - L2•cos(th1)•sin(th2) • (cos(ph) • sin(ep) - cos(ep) • sin(ph) • sin(th))) + L2•sin(th1)•sin(th2) • (sin(ep)•sin(ph) + cos(ep)•cos(ph)•sin(th))), xde•(L2•cos(ep)•cos(th)•cos(th2) - L2•cos(th1)•sin(th2)•(cos(ph) • sin(ep) - cos(ep) • sin(ph) • sin(th))) + L2•sin(th1)•sin(th2) • (sin(ep)•sin(ph) + cos(ep)•cos(ph)•sin(th))), ye•(L2•cos(th)•cos(th2) • sin(ph) • sin(th)) + L2•cos(th1)•sin(th2) • (cos(ep)•cos(ph) + sin(ep)•sin(th)) + (L1•cos(th1) + L2•cos(th1) • cos(th2))•(cos(ep)•sin(ph) - cos(ph)•sin(ep)•sin(th))), -ze•cos(ph + th1)•cos(th)•(L1 + L2•cos(th2)), -zde•cos(ph + th1)•cos(th)•(L1 + L2•cos(th2)), epe•cos(ep)•cos(th), epde•cos(ep)•cos(th), -the•cos(ph) • sin(th), -thde•cos(ph)•sin(th), phe•cos(ph + th1)•cos(th), phde•cos(ph + th1)•cos(th)];

Ys=[Ya(1,:), JT_\text{Fe\_reg1,D\_ex\_reg1};
Ys=[Ya(2,:), JT_\text{Fe\_reg2,D\_ex\_reg2};
D. SYSTEM JACOBIAN AND PARAMETRIZATION

\begin{verbatim}
109  Ya(3,:),JT_Fe_reg3,D_ex.reg3;
110  Ya(4,:),JT_Fe_reg4,D_ex.reg4;
111  Ya(5,:),JT_Fe_reg5,D_ex.reg5;
112  Ya(6,:),JT_Fe_reg6,D_ex.reg6;
113  Ya(7,:),JT_Fe_reg7,D_ex.reg7;
114  Ya(8,:),JT_Fe_reg8,D_ex.reg8);
115
116  \% ================ hs ================
117  hs = [m1;m2;mb;I1;I2;Ix;L0*m1;L0*m2;L1*m1;L1*m2;L2*m1;L2*m2;L0^2*m1;L0^2*m2;
         L1^2*m1;L1^2*m2;L2^2*m]
\end{verbatim}
Appendix E

Laser Range Finder

URG-04LX is a laser sensor for area scanning, see Fig. E.1. The light source of the sensor is infrared laser of wavelength 785nm with laser class 1 safety. Scan area is 240° semicircle with maximum radius 4000mm. Principle of distance measurement is based on calculation of the phase difference, due to which it is possible to obtain stable measurement with minimum influence from objects color and reflectance.

Figure E.1: URG-04LX Laser Range Finder

It has the following features:

- High accuracy, high resolution and wide angle which provide the best solution for autonomous robots moving in the unknown environment.
- Compact size allows more designing freedom. Light weight and low power consumption.
contribute to the long time operations.

- No influence by how bright the environment will be. Excellent performance in darkness. Recognizing human body’s size and position without hurting their privacy.

Fig. E.2 illustrates the principle of operation of the URG-04LX. The laser emits an infrared beam, and a rotating mirror changes the beams direction. Then the laser hits the surface of an object and is reflected. The direction of reflected light is changed again by a rotating mirror, and captured by the photo diode. The phases of the emitted and received light are compared and the distance between the sensor and the object is calculated. A rotating mirror sweeps the laser beam horizontally over a range of 240°, with an angular resolution of 0.36°. As the mirror rotates at about 600 rpm, the scan rate is about 100 msec.

![Diagram of URG-04LX](image)

**Figure E.2:** The principle of operation of the URG-04LX

Technical specifications of this device is given in Table E.1.

Fig. E.3 illustrates schematic diagrams with the relevant dimensions.

There are two connectors RS-connector (Table E.2) and USB connector (Table E.3).
Table E.1: Laser Range Finder, URG-04LX, specifications

| Model No.                | URG-04LX                                                                 |
|--------------------------|--------------------------------------------------------------------------|
| Power source             | 5VDC±5%. Sensor will not operate with USB bus power. Prepare power source separately. |
| Current consumption      | 500mA or less (800mA when start-up)                                      |
| Measuring area           | 60 to 4095mm (white paper with 70mm) 240°                               |
| Accuracy                 | is ±10 mm for distances of less than 1 m. For greater distances, the error is quoted as ±2% |
| Repeatability            | 60 to 1,000mm : ±10mm                                                    |
| Angular resolution       | Step angle : approx. 0.36 (360/1,024 steps)                              |
| Light source             | Semiconductor laser diode (wave length ≈785nm)                          |
| Scanning time            | 100ms/scan                                                               |
| Noise                    | 25dB or less                                                             |
| Interface                | USB, RS-232C (19.2k, 57.6k, 115.2k, 250k, 500k, 750kbps)                |
| Ambient temperature/humidity | -10 to +50 degrees C, 85% or less (Not condensing, not icing)          |
| Vibration resistance     | 10 to 55Hz, double amplitude 1.5mm Each 2 hour in X, Y and Z directions |
| Impact resistance        | 196m/s², Each 10 time in X, Y and Z directions                          |
| Weight                   | Approx. 160g                                                             |
| Accessory                | Cable for power communication/input-output (1.5m), D-sub connector with 9 pins |
Figure E.3: External dimension of the Laser Range Finder URG-04LX
**Table E.2:** Laser Range Finder, URG-04LX, Pins of the interface connector

| Pin No. | Signals                        | Colors |
|---------|--------------------------------|--------|
| 1       | N.C.                           | Red    |
| 2       | N.C.                           | White  |
| 3       | OUTPUT(Synchronous output)     | Black  |
| 4       | GND(5th pin of 9-pin, D-sub connector) | Purple |
| 5       | RxD(3rd pin of 9-pin, D-sub connector) | Yellow |
| 6       | TxD(2nd pin of 9-pin, D-sub connector) | Green |
| 7       | 0V                             | Blue   |
| 8       | DC5V                           | Brown  |

**Table E.3:** Laser Range Finder, URG-04LX, the USB connector

| Connector Type | USB-miniB (5pin) |
|----------------|------------------|
Appendix F

Ultrasonic Ranger

The Devantech SRF04 ultrasonic range finder (see Fig. F.1) provides precise, non-contact distance measurements from about 3 cm to 3 m. The SRF04 works by transmitting an ultrasonic (well above human hearing range) pulse and measuring the time it takes to “hear” the pulse echo. Output from the SRF04 is in the form of a variable-width pulse that corresponds to the distance to the target. Technical specifications of this device is given in Table F.1.

![Devantech SRF04 Ultrasonic Range Finder](image)

**Figure F.1:** Devantech SRF04 Ultrasonic Range Finder

Fig. F.2 illustrates schematic diagrams of the sonar with the relevant dimensions.

Fig. F.3 shows the terminal connections of the sonar.

The SRF04 beam pattern is given in Fig. F.4. Its Timing diagram is shown in Fig. F.5. We only need to supply a short 10 µs pulse to the trigger input to start the ranging. The SRF04 will send out an 8 cycle burst of ultrasound at 40khz and raise its echo line high. It then listens for an echo, and as soon as it detects one it lowers the echo line again. The echo
Table F.1: Ultrasonic, SRF04, technical specifications

| Sensor type       | Reflective Ultrasonic          |
|-------------------|---------------------------------|
| Sender            | N1076, Receiver N1081           |
| Voltage           | 5v only required                |
| Current           | 30mA Typical, 50mA Max.        |
| Frequency         | 40kHz                           |
| Range             | 3cm - 3m                        |
| Sensitivity       | Detect 3cm diameter             |
| Input Trigger     | 10µS Min. TTL (Transistor Transistor Logic) level pulse |
| Echo Pulse        | Positive TTL level signal, width proportional to range |
| I/O required      | two digital lines, 1 output, 1 input |
| Small Size        | 43mm x 20mm x 17mm height      |
| Weight            | 11.33 gm                        |

Figure F.2: External dimension of the SRF04

line is therefore a pulse whose width is proportional to the distance to the object. By timing the pulse it is possible to calculate the range in inches/centimeters or anything else. If the width of the pulse is measured in µs, then dividing by 58 will give you the distance in cm.
Figure F.3: Terminal Connections of the SRF04

Figure F.4: Beam Pattern of the SRF04
F. ULTRASONIC RANGER

Figure F.5: Timing diagram of the SRF04 [4]
Appendix G

System Internal Connections

Internal connections (i.e., power and data connections) of the proposed robot are given in Fig. G.1.

Figure G.1: Power connections of the manipulator
Appendix H

Robot Operating System (ROS)

H.1 ROS

ROS is an open-source, meta-operating system for your robot. It provides the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management. It also provides tools and libraries for obtaining, building, writing, and running code across multiple computers.

Why do we use ROS?

There are specific issues in the development of software for robots that ROS can help to resolve, which are:

- Distributed computation: Many modern robot systems rely on software that spans many different processes and runs across several different computers. For example: Some robots carry multiple computers, each of which controls a subset of the robots sensors or actuators. Even within a single computer, its often a good idea to divide the robots software into small, stand-alone parts that cooperate to achieve the overall goal. This approach is sometimes called “complexity via composition”.

- Software reuse: The rapid progress of robotics research has resulted in a growing collection of good algorithms for common tasks such as navigation, motion planning, mapping, and many others. Of course, the existence of these algorithms is only truly
H. ROBOT OPERATING SYSTEM (ROS)

useful if there is a way to apply them in new contexts, without the need to reimplement each algorithm for each new system. As a result, developers that use ROS can expect, of course, climbing ROSs initial learning curve to focus more time on experimenting with new ideas, and less time reinventing wheels.

• Rapid testing: One of the reasons that software development for robots is often more challenging than other kinds of development is that testing can be time consuming and error-prone. Physical robots may not always be available to work with, and when they are, the process is sometimes slow and finicky. Working with ROS provides two effective workarounds to this problem.

  – Well-designed ROS systems separate the low-level direct control of the hardware and high-level processing and decision making into separate programs. Because of this separation, we can temporarily replace those low-level programs (and their corresponding hardware) with a simulator, to test the behavior of the high-level part of the system.

  – ROS also provides a simple way to record and play back sensor data and other kinds of messages.

Therefore, because the real robot, the simulator, and the bag playback mechanism can all provide identical (or at least very similar) interfaces, your software does not need to be modified to operate in these distinct scenarios, and indeed need not even know whether it is talking to a real robot or to something else.

All ROS software is organized into packages. A ROS package is a coherent collection of files, generally including both executables (nodes) and supporting files, that serves a specific purpose. A robot control system comprises many nodes. For example, one node controls a laser range-finder, one Node controls the robot’s wheel motors, one node performs localization.

The primary mechanism that ROS nodes use to communicate is to send messages. Messages in ROS are organized into named topics. The idea is that a node that wants to share information will publish messages on the appropriate topic; a node that wants to receive
information will subscribe to the topic that it is interested in. The ROS master takes care of ensuring that publishers and subscribers can find each other; the messages themselves are sent directly from publisher to subscriber.

H.2 C++ Node for Extracting the Euler Angles from the Quaternion

The ROS AscTec driver gives the quadrotor orientation in the form of quaternion. However, I need the roll and pitch angles to compensate the measured height from the sonar. Thus, I developed a ROS node to convert the quaternion to Euler angles.

```cpp
/* Conversion from a quaternion to roll, pitch and yaw. */

#include "ros/ros.h"
#include "geometry_msgs/Vector3.h"
#include "geometry_msgs/Quaternion.h"
#include "tf/transform_datatypes.h"
#include "LinearMath/btMatrix3x3.h"
#include <sensor_msgs/Imu.h>

// Here I use global publisher and subscriber, since I want to access the
// publisher in the function MsgCallback:
ros::Publisher rpy_publisher;
ros::Subscriber quat_subscriber;

// Function for conversion of quaternion to roll pitch and yaw. The angles
```
// are published here too.

void MsgCallback(const sensor_msgs::ImuPtr& msg)
{
    // the incoming geometry_msgs::Quaternion is transformed to a tf::Quaternion
    tf::Quaternion quat;
    tf::quaternionMsgToTF(msg->orientation, quat);

    // the tf::Quaternion has a method to access roll pitch and yaw
    double roll, pitch, yaw;
    tf::Matrix3x3(quat).getRPY(roll, pitch, yaw);

    // the found angles are written in a geometry_msgs::Vector3
    geometry_msgs::Vector3 rpy;
    rpy.x = roll;
    rpy.y = pitch;
    rpy.z = yaw;

    // this Vector is then published:
    rpy_publisher.publish(rpy);
    ROS_INFO("published rpy angles: roll=%f pitch=%f yaw=%f", rpy.x, rpy.y, rpy.z);
}

int main(int argc, char **argv)
{
    ros::init(argc, argv, "talker");
    ros::NodeHandle n;
    rpy_publisher = n.advertise<geometry_msgs::Vector3>("rpy_angles", 1000);
    quat_subscriber = n.subscribe("rotation_quaternion", 1000, MsgCallback);

    // check for incoming quaternions until ctrl+c is pressed
    ROS_INFO("waiting for quaternion");
    ros::spin();
    return 0;
}
H.3 ROS-based Arduino Code

In the Arduino board, I write a program based on the rosserial library which enables me to communicate with ROS. Firstly, this C++ code reads the command from the WPS2 receiver, then process this command and then it to the SSC32 board via serial communications. Thirdly, it acquires and processes data from the sonar. Finally, it sends the height to the ROS.

```cpp
#include <DistanceSRF04.h>
#include <Wire.h>
#include <ros.h>
#include <geometry_msgs/Vector3.h>
#include <ros/time.h>
#include <std_msgs/Float32.h>

//=============== manp ==================
#include <PS2X_lib.h>
#define PS2_DAT 5 //14
#define PS2_CMD 4 //15
#define PS2_SEL 3 //16
#define PS2_CLK 2 //17
//#define pressures true
#define pressures false
//#define rumble true
#define rumble false

PS2X ps2x;
int error = 0;
byte type = 0;
byte vibrate = 0;
int angle0 = 90;
int angle1 = 90;
int grasp = 90;
int angle0_pwm;
int angle1_pwm;
int grasp_pwm;
```
int joint0_ch=0;
int joint1_ch=4;
int grasp_ch=8;
int anglestep =1;
int time = 1;

//=======================================

std_msgs::Float32 sonar_msg;
ros::Publisher pub_sonar("sonar", &sonar_msg);
ros::NodeHandle nh;

DistanceSRF04 Dist;

void setup()
{
    Dist.begin(8,9);
    nh.initNode();
    nh.advertise(pub_sonar);

    // ================= Manp ================
    Serial1.begin(115200);
    delay(500);
    error = ps2x.config_gamepad(PS2_CLK, PS2_CMD, PS2_SEL, PS2_DAT,
    pressures, rumble);
    type = ps2x.readType();
    // ================= Manp ================
}

long publisher_timer;

void loop()
{
    if (millis() > publisher_timer) {
        sonar_msg.data = Dist.getDistancemeter();
        pub_sonar.publish(&sonar_msg);
        publisher_timer = millis() + 100; //publish once a second
manp();

nh.spinOnce();

// ======================== Manp Functions =================
void manp()
{
    if(error == 1) //skip loop if no controller found
        return;

    if(type == 2)
        //Guitar Hero Controller
        ps2x.read_gamepad(); //read controller
    else
        //DualShock Controller
        ps2x.read_gamepad(false, vibrate); //read controller and set large
        motor to spin at 'vibrate' speed

    if(ps2x.Button(PSB_PAD_UP)) //will be TRUE as long as button is
        pressed
        angle0 +=anglestep;
    //

    if(ps2x.Button(PSB_PAD_DOWN))
        angle0 -=anglestep;
    //

    if(ps2x.Button(PSB_PAD_RIGHT))
        angle1 +=anglestep;
    //

    if(ps2x.Button(PSB_PAD_LEFT)){
angle1 -= anglestep;
Serial1.print("pressures = ");
}
if (angle0 < 0) angle0 = 0;
if (angle0 > 180) angle0 = 180;
if (angle1 < 0) angle1 = 0;
if (angle1 > 180) angle1 = 180;

angle0_pwm = map(angle0, 0, 180, 500, 2500);
move(joint0_ch, angle0_pwm, time);
angle1_pwm = map(angle1, 0, 180, 500, 2500);
move(joint1_ch, angle1_pwm, time);

vibrate = ps2x.Analog(PSAB_CROSS); //this will set the large motor
vibrate speed based on how hard you press the blue (X) button

if (ps2x.Button(PSB_L1) || ps2x.Button(PSB_R1)) {
    if (ps2x.Button(PSB_L1)) {
        grasp -= anglestep;
    }
    if (ps2x.Button(PSB_R1)) {
        grasp += anglestep;
    }
}
if (grasp < 0) grasp = 0;
if (grasp > 180) grasp = 180;
grasp_pwm = map(grasp, 0, 180, 500, 2500);
move(grasp_ch, grasp_pwm, time);
} delay(50);

void move(int servo, int position, int time) {

Serial1.print("#");
Serial1.print(servo);
Serial1.print(" P");
Serial1.print(position);
Serial1.print(" T");
Serial1.println(time);
delay(time);
}

H.4 Altitude Compensation Node

This node receives the height from the Arduino and then it compensates the altitude based on the measured roll and pitch angles.

#include "ros/rosh"
#include <ros/time.h>
#include "geometry_msgs/Vector3.h"
#include "mav_msgs/Height.h"
#include "std_msgs/Float32.h"
#include "var.h"

ros::Publisher alt_pub;
ros::Subscriber sonar_sub;
ros::Subscriber rpy_sub;
float sonar_alt, ph, th;
char frameid[] = "";

void MsgCallback(const std_msgs::Float32& sonar_msg)
{
  sonar_alt = sonar_msg.data;
  return;
}

void MsgCallbackrpy(const geometry_msgs::Vector3& rpy_msg)
{

H. ROBOT OPERATING SYSTEM (ROS)

```c
22 ph=rpy_msg.x;
23 th=rpy_msg.y;
24 return;
25 }
26 int main(int argc, char **argv)
27 {
28 ros::init(argc, argv, "altitude");
29 ros::NodeHandle n;
30 sonar_sub = n.subscribe("sonar", 5, MsgCallback);
31 rpy_sub = n.subscribe("rpy_angles", 5, MsgCallbackrpy);
32 alt_pub = n.advertise<mav_msgs::Height>("alt", 5);
33 ros::Rate loop_rate(10);
34 while (ros::ok())
35 {
36 mav_msgs::Height alt_msg;
37 alt_msg.height = sonar_alt*cos(ph)*cos(th);
38 alt_msg.header.stamp = ros::Time::now();
39 alt_msg.header.frame_id = frameid;
40 alt_msg.height_variance = 0;
41 alt_msg.climb= 0.0;
42 alt_msg.climb_variance= 0;
43 alt_pub.publish(alt_msg);
44 ros::spinOnce();
45 loop_rate.sleep();
46 }
47 ros::spin();
48 return 0;
49 }
```

H.5 Joystick teleoperation Node

This node reads the joystick buttons and converts them to position and velocity commands.

```c
#include <ros/rosh>
```

194
```cpp
#include <sensor_msgs/Joy.h>
#include <std_msgs/Bool.h>
#include <std_msgs/Float32.h>
#include <stdlib.h>

ros::Subscriber joy_sub;
ros::Publisher x_pub;
ros::Publisher y_pub;
ros::Publisher z_pub;
ros::Publisher ep_pub;
ros::Publisher reset_pub;
ros::Publisher zvel_pub;

//last_joy_event_ = ros::Time::now();

float x_step_size=0.01;
float x_old=0;
float x_delta=0;
float y_step_size=0.01;
float y_old=0;
float y_delta=0;
float z_step_size=0.01;
float z_old=0;
float z_delta=0;
float z_min=0;
float ep_step_size=0.01;
float ep_old=0;
float ep_delta=0;
float zvel_step_size=0.01;
float zvel_old=0;
float zvel_delta=0;
float zvel_min=0;
bool sw=0;
bool b0,b1,b2,b3,b5,b7,b8,b10,b11,b4,b6,b9;

// ****
void joyCallback(const sensor_msgs::JoyPtr& joy_msg)
{
    // start of callback
    b0=joy_msg->buttons[0];

    // rest of callback

    // end of callback
}
```
H. ROBOT OPERATING SYSTEM (ROS)

```c
b1=joy_msg->buttons[1];
b2=joy_msg->buttons[2];
b3=joy_msg->buttons[3];
b5=joy_msg->buttons[5];
b7=joy_msg->buttons[7];
b8=joy_msg->buttons[8];
b10=joy_msg->buttons[10];
b11=joy_msg->buttons[11];
b4=joy_msg->buttons[4];
b6=joy_msg->buttons[6];
b9=joy_msg->buttons[9];

//===============================================
std_msgs::Float32 x_msg;
std_msgs::Float32 y_msg;
std_msgs::Float32 z_msg;
std_msgs::Float32 ep_msg;
std_msgs::Bool reset_msg;
std_msgs::Float32 zvel_msg;

// ================== x-axis ================================
if (b0 ==1 || b2 == 1)
{
    if (b0 == 1)
    {
        x_delta= x_step_size ;
    }
    else if (b2 == 1)
    {
        x_delta= -x_step_size ;
    }

    x_msg.data = x_old + x_delta;
    x_old=x_msg.data;
}
else {x_msg.data = x_old;}

// ==================================================================
```
// ================== y-axis ============================== 
if (b3 == 1 || b1 == 1) 
{
  if (b3 == 1) 
  {
    y_delta = y_step_size;
  }
  else if (b1 == 1) 
  {
    y_delta = -y_step_size;
  }
  y_msg.data = y_old + y_delta;
  y_old = y_msg.data;
} 
else {y_msg.data = y_old;}
// ============================================================== 
// ================== z-axis ============================== 
if (b5 == 1 || b7 == 1) 
{
  if (b5 == 1) 
  {
    z_delta = z_step_size;
  }
  else if (b7 == 1) 
  {
    z_delta = -z_step_size;
  }
  z_msg.data = z_old + z_delta;
  z_msg.data = std::max(z_msg.data, z_min);
  z_old = z_msg.data;
} 
else {z_msg.data = z_old;}

// =======================================================
// ================== ep-axis ==============================  
if (b11 ==1 || b10 == 1)  
{
  if (b11 == 1)  
  {
    ep_delta= ep_step_size ;
  }
 else if (b10 == 1)  
  {
    ep_delta= -ep_step_size ;
  }

  ep_msg.data = ep_old + ep_delta;
 ep_old=ep_msg.data;
  }

else {ep_msg.data = ep_old;}

// =======================================================
// ================== reset ============================== 

if (b9 ==1 || sw==1 && b8==0) {reset_msg.data=1; sw=1;}
else {reset_msg.data=0; sw=0;}

// =======================================================
// ================== zvel-axis ==============================  
if (b4 ==1 || b6 == 1)  
{
  if (b4 == 1)  
  {
    zvel_delta= zvel_step_size ;
  }
 else if (b6 == 1)  

{ 
  zvel_delta = -zvel_step_size;
}

zvel_msg.data = zvel_old + zvel_delta;
zvel_old = zvel_msg.data;
}

else {zvel_msg.data = zvel_old;}

// =======================================================

x_pub.publish(x_msg);
y_pub.publish(y_msg);
z_pub.publish(z_msg);
ep_pub.publish(ep_msg);
reset_pub.publish(reset_msg);
zvel_pub.publish(zvel_msg);

//==================================
return;
}

} // end of call back

int main(int argc, char ** argv)
{
  ros::init(argc, argv, "quad_joystick_teleop");
  ros::NodeHandle nh;
  x_pub = nh.advertise<std_msgs::Float32>("x_des", 5);
  y_pub = nh.advertise<std_msgs::Float32>("y_des", 5);
  z_pub = nh.advertise<std_msgs::Float32>("z_des", 5);
  ep_pub = nh.advertise<std_msgs::Float32>("ep_des", 5);
  reset_pub = nh.advertise<std_msgs::Bool>("reset_des", 5);
  zvel_pub = nh.advertise<std_msgs::Float32>("zvel_des", 5);
  joy_sub = nh.subscribe("joy", 5, joyCallback);
  ros::spin();
  return 0;
}
Appendix I

6-DOF Torque/Force Sensor

We use a torque force sensor of model, IFS-67M25A25-I40ANA, see Fig. [I.1]. In this sensor, foil strain gages sense the loads imposed on the sensor. The strain gage signals are amplified and combined to become analog representations of the force loads on the three axes and the moments or torques about the three axes.

![Image of the torque force sensor](image)

**Figure I.1:** The Torque Force sensor with the interface mount to rotor to be identified

Fig. [I.2] presents the sensor axis orientation from the rotor side.

It utilizes a DE-9P connector to produce the analogue output. The pin assignment is given in Table [I.1].
I. 6-DOF TORQUE/FORCE SENSOR

Figure I.2: Torque/Force sensor axis orientation from the rotor side

Table I.1: Pin Connections of the 6-DOF torque/force sensor

| PIN | SIGNAL                  |
|-----|-------------------------|
| 1   | \( F_x \)               |
| 2   | \( F_y \)               |
| 3   | \( F_z \)               |
| 4   | \( M_x \)               |
| 5   | \( M_y \)               |
| 6   | \( M_z \)               |
| 7   | + power, Sensor excitation: ±5.0V |
| 8   | - power                 |
| 9   | power and signal common |

Data from analog sensors must be processed using the Calibration Matrix provided with the sensor.

The six by six calibration matrix is multiplied by the six element voltage (column) vector. The result is the calibrated force and moment data in the units of N and Nm. Multiply the calibration matrix and the sensor voltage vector to determine the loads (N and Nm).
\[
\begin{bmatrix}
F_x \\
F_y \\
F_z \\
M_x \\
M_y \\
M_z \\
\end{bmatrix}
= 
\begin{bmatrix}
49.487 & -1.552 & -0.671 & -4.053 & -1.636 & -0.043 \\
1.852 & 49.666 & -0.483 & 1.450 & -2.442 & -1.495 \\
-1.695 & 2.721 & 86.221 & 1.107 & -0.006 & 0.081 \\
0.000 & 0.121 & 0.026 & 2.904 & 0.140 & 0.032 \\
-0.135 & 0.001 & 0.018 & -0.058 & 0.024 & 0.064 \\
-0.128 & 0.012 & 2.800 & 0.033 & -0.098 & 3.042 \\
\end{bmatrix}
\times
\begin{bmatrix}
F_{x\text{voltage}} \\
F_{y\text{voltage}} \\
F_{z\text{voltage}} \\
M_{x\text{voltage}} \\
M_{y\text{voltage}} \\
M_{z\text{voltage}} \\
\end{bmatrix}
\]
I. 6-DOF TORQUE/FORCE SENSOR
Appendix J

Inverse Kinematics

function zt_des = Inverse_Kin(Xe_des, sb)
L0 = 30e-03;
L1 = 70e-03;
L2 = 85e-03;

xe=Xe_des(1);
ye=Xe_des(2);
ze=Xe_des(3);
epe=Xe_des(4);
the=Xe_des(5);
phe=Xe_des(6);

ph = sb(2);
th = sb(1);

T_end = [ cos(epe)*cos(the), cos(epe)*sin(phe)*sin(the) - cos(phe)*sin(epe),
          sin(epe)*sin(phe) + cos(epe)*cos(phe)*sin(the),
          cos(the)*sin(epe), cos(epe)*cos(phe) + sin(epe)*sin(phe)*sin(the),
          cos(phe)*sin(epe)*sin(the) - cos(epe)*sin(phe),
          -sin(the), cos(the)*sin(phe), cos(phe)*cos(the)];

r11 = T_end(1,1);
r12 = T_end(1,2);
r13 = T_end(1,3);
r23 = T_end(2,3);
r31 = T_end(3,1);
r32 = T_end(3,2);
r33 = T_end(3,3);

a1 = \cos(\theta) \cos(\phi);
b1 = \cos(\phi) \sin(\theta);

if((-2*b1)^2 - 4*(-a1-r33)*(a1-r33))<0
  t2=0;
else
  t2=(-(-2*b1) -sqrt((-2*b1)^2 - 4*(-a1-r33)*(a1-r33)))/(2*(-a1-r33));
end
th1=2*atan(t2);

% Case#2 =====> Sin(th1) = 0
if(th1<=0.0001 & th1>=-0.0001)
  th1=0;
  ep=0;
a6 = \cos(\theta) \cos(\phi);
b6 = \cos(\phi) \sin(\phi) \sin(\theta) \cos(\phi) - \sin(\phi) \sin(\theta) \cos(\phi);
a7 = \sin(\phi) \sin(\theta) \cos(\phi) - \cos(\phi) \sin(\phi);
b7 = \cos(\theta) \cos(\phi);
sth2 = (b7*r11 + b6*r12)/(a6*b7+a7*b6);
th2 = atan2(sth2,cth2);

% Case#3 =====> Sin(th1) = pi
elseif (th1==pi)
  th1=pi;
  ep=0;
a6 = \cos(\theta) \cos(\phi);
b6 = -\cos(\phi) \sin(ep) + \sin(\phi) \sin(th) \cos(ep);
a7 = -\sin(\phi) \sin(th) \cos(ep) + \cos(\phi) \sin(ep);
b7 = cos(th) * cos(ep);
sth2 = (b7 * r11 + b6 * r12) / (a6 * b7 + a7 * b6);
cth2 = (a7 * r11 + a6 * r12) / (a6 * b7 + a7 * b6);

th2 = atan2(sth2, cth2);

%% Case#1 ======>> Sin(th1) != 0 ======>> r13 = 0 && r23 = 0
else

a2 = cos(ph) * sin(th1) + sin(ph) * cos(th1);
b2 = cos(ph) * sin(th) * cos(th1) - sin(ph) * sin(th) * sin(th1);

a3 = cos(ph) * sin(th) * cos(th1) - sin(ph) * sin(th) * sin(th1);
b3 = -sin(ph) * cos(th1) - cos(ph) * sin(th1);

a4 = cos(ph) * cos(th) * sin(th1) + cos(th) * sin(ph) * cos(th1);
b4 = -sin(th);

a5 = -sin(th);
b5 = -cos(ph) * cos(th) * sin(th1) - cos(th) * sin(ph) * cos(th1);

sep = (b3 * r13 + b2 * r23) / (a2 * b3 + a3 * b2);
cep = (a3 * r13 + a2 * r23) / (a2 * b3 + a3 * b2);
ep = atan2(sep, cep);

sth2 = (b5 * r32 + b4 * r31) / (a4 * b5 + a5 * b4);
cth2 = (a5 * r32 + a4 * r31) / (a4 * b5 + a5 * b4);

th2 = atan2(sth2, cth2);
end

%% Translational Part

X = xe - (- L0 * (sin(ph) * sin(ep) + cos(ph) * sin(th) * cos(ep)) + L2 * cos(th2) * (cos(th1) * (cos(ph) * sin(ep) - sin(ph) * sin(th) * cos(ep)) - sin(th1) * (sin(ph) * sin(ep) + cos(ph) * sin(th) * cos(ep))) + L1 * cos(th1) * (cos(ph) * sin(ep) - sin(ph) * sin(th) * cos(ep)) - L1 * sin(th1) * (sin(ph) * sin(ep) + cos(ph) * sin(th) * cos(ep)) + L2 * cos(th) * cos(ep) * sin(th2));

Y = ye - (+ L0 * (sin(ph) * cos(ep) - cos(ph) * sin(th) * sin(ep)) - L2 * cos(th2) * (cos(th1) * (cos(ph) * cos(ep) + sin(ph) * sin(th) * sin(ep)) - sin(th1) * (sin(ph) * cos(ep) + cos(ph) * sin(th) * sin(ep))) - L1 * cos(th1) * (cos(ph) * cos(ep) + sin(ph) * sin(th) * sin(ep)) + L1 * cos(th) * cos(ep) * sin(th2));
\[
\begin{align*}
L_1 \cdot \sin(\theta_1) \cdot (\sin(\phi) \cdot \cos(\epsilon) - \cos(\phi) \cdot \sin(\theta) \cdot \sin(\epsilon)) + \\
L_2 \cdot \cos(\theta) \cdot \sin(\epsilon) \cdot \sin(\theta_2)); \\
Z &= -L_2 \cdot \sin(\theta) \cdot \sin(\theta_2) - L_0 \cdot \cos(\phi) \cdot \cos(\theta) - \\
&- L_2 \cdot \cos(\theta_2) \cdot (\cos(\phi) \cdot \cos(\theta) \cdot \sin(\theta_1) + \cos(\theta) \cdot \sin(\phi) \cdot \cos(\theta_1)) - \\
&- L_1 \cdot \cos(\phi) \cdot \cos(\theta) \cdot \sin(\theta_1) - L_1 \cdot \cos(\theta) \cdot \sin(\phi) \cdot \cos(\theta_1)); \\
\text{zt}_\text{des} &= [X; Y; Z; \epsilon; \theta_1; \theta_2];
\end{align*}
\]