NTNU
Norwegian University of
Science and Technology

Subsea Inspection and Intervention with Underwater Swimming Manipulators

Sara Vinje Aalbu

Master of Science in Cybernetics and Robotics
Submission date: June 2018
Supervisor: Kristin Ytterstad Pettersen, ITK
Co-supervisor: Anna Kohl, ITK
Jørgen Sverdrup-Thygeson, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Preface

This master thesis was written during the spring of 2018 as fulfillment of the Cybernetics and Robotics study programme at the Norwegian University of Science and Technology (NTNU). The problem description was developed in cooperation with Professor Kristin Ytterstad Pettersen.

I would like to thank my supervisor, Kristin Ytterstad Pettersen, for the guidance and feedback throughout the semester. I would also like to thank both my co-supervisors Anna M. Kohl and Jørgen Sverdrup-Thygeson for discussions regarding stability analyses and simulation problems. Thanks to Henrik Schmidt-Didlaukies for providing code and being available for questions. Finally, I would like to thank my family for supporting and encouraging me during the work on this thesis.

Section 1.4 in the introductory chapter of the thesis contains more details about contributions of the thesis, provided software and help received during the work on the thesis.

Sara Vinje Aalbu
Trondheim, June, 2018

Abstract

This thesis examines dynamic motion control approaches and interaction control approaches for underwater swimming manipulators (USMs). USMs are innovative underwater vehicles with potential to increase efficiency and reduce costs of subsea inspection, maintenance and repair. Dynamic motion control of USMs can be used for USM inspection tasks, while interaction control must be considered for USM interaction with the environment. To illustrate the performance of the different controllers, a USM simulator is implemented with a complete control framework. An important aspect when designing dynamic control approaches for USMs is uncertainty in model knowledge. This is taken into account by the presented dynamic motion control approaches. For dynamic motion control, three control strategies are presented: adaptive inverse dynamics control, the super-twisting algorithm with adaptive gains and non-regressor-based adaptive control. The dynamic motion controllers are implemented with the USM simulator. The stability properties of the controllers are analyzed and their expected behaviour is compared to how they behave in simulations. The simulations show that all the dynamic motion control approaches result in behaviour that is in line with the theoretical analysis. The interaction control approaches presented in the thesis are an impedance controller and a PI force controller with impedance control. The simulator is extended to include interaction forces for the task of turning a valve and the interaction controllers are implemented with the simulator. The simulation results with the different interaction controllers are presented and the control strategies are compared. Both interaction controllers perform well in the simulations.

Sammendrag

Denne oppgaven undersøker dynamisk kontroll og interaksjonskontroll for svømmende undervannsmanipulatorer (USMer). USMer er nyskapende undervannsfarkoster med potensial til å øke effektiviteten og redusere kostnadene ved undervannsinspeksjon, -vedlikehold og -reparasjon. Dynamisk kontroll kan brukes ved inspeksjon med USMer, mens interaksjonskontroll må inkluderes for USM-interaksjon med omgivelsene. For å illustrere ytelsen til de forskjellige kontrollmetodene, implementeres en USM-simulator med et komplett rammeverk for kontroll. Et viktig aspekt ved utforming av dynamiske kontrollmetoder for USMer er usikkerhet i modellen. Dette er tatt med i betraktningen av de presenterte dynamiske kontrollmetodene. For dynamisk kontroll presenteres tre kontrollstrategier: adaptiv inversdynamikkontroll, en «super-twisting» algoritme med adaptiv forsterkning og ikke-regressorbasert adaptiv kontroll. De dynamiske kontrollerne blir implementerte med USM-simulatoren. Stabilitetsegenskapene til kontrollerne analyseres og deres forventede oppførsel blir sammenlignet med hvordan de oppfører seg i simuleringer. Simuleringene viser at alle de dynamiske kontrollerne gir resultater som er i tråd med den teoretiske analysen. Interaksjonskontrollmetodene som presenteres i oppgaven er impedanskontroll og PI-kraftkontroll med impedanskontroll. Simulatoren blir utvidet til å inkludere interaksjonskreftene ved dreining av en ventil og interaksjonskontrollerne blir implementerte med simulatoren. Simuleringsresultatene, med de forskjellige strategiene for interaksjonskontroll, er presentert og kontrollstrategiene sammenlignes. Begge interaksjonskontrollerne fungerer godt i simuleringene.

Contents

List of Figures

List of Abbreviations

AIDC	Adaptive inverse dynamics control
AUV	Autonomous underwater vehicle
DOF	Degrees of freedom
DC	Dynamic motion control
EOM	Equations of motion
FC	PI force control with impedance control
I-AUV	Intervention AUV
IC	Impedance control
IK	Inverse kinematics
IMR	Inspection, maintenance and repair
NRAC	Non-regressor-based adaptive control
ROV	Remotely operated underwater vehicle
PI	Proportional Integral
STA	Super-twisting algorithm
TA	Thrust allocation
TCM	Thrust Configuration Matrix
USM	Underwater swimming manipulator
USR	Underwater snake robot
UVMS	Underwater vehicle manipulator system

List of Symbols

Symbols	Description	Vector or matrix
n	Number of joints	
m	Number of thrusters	
η_1	Position of base in inertial frame	$\eta_1 \in R^3$
η_2	Orientation of base, Euler angles	$\eta_2 \in R^3$
η_{1,b_i}	Position of link i in inertial frame	$\eta_1 \in R^3$
η_{2,b_i}	Orientation of link i, Euler angles	$\eta_2 \in R^3$
ν_1	Body-fixed linear velocity of base	$\nu_1 \in R^3$
ν_2	Body-fixed angular velocity base	$\nu_2 \in R^3$
ν_{1,b_i}	Body-fixed linear velocity of link i	$\nu_1 \in R^3$
ν_{2,b_i}	Body-fixed angular velocity of link i	$\nu_2 \in R^3$
R_I^B	Rotation matrix from frame I to frame B	$R_I^B \in R^{3\times3}$
p	Quaternion i	$p \in R^4$
η_Q	Real part of quaternion	
ϵ	Imaginary parts of the quaternion	$\epsilon \in R^3$
$\tilde{\eta_Q}$	Quaternion error, real part	
$\tilde{\epsilon}$	Quaternion error, imaginary part	$\tilde{\epsilon} \in R^3$

Symbols	Description	Vector or matrix
$J_{k,o}(\eta_2)$	Transformation matrix, Euler angles	$J_{k,o}(\eta_2) \in R^{3\times3}$
$J_{k,oq}(p)$	Transformation matrix, quaternions	$J_{k,oq}(\eta_2) \in R^{4\times3}$
q_i	Joint angle of joint i	$q \in R^n$
r	Radius of links	
l_i	Length of link i	
ζ	Configuration space velocities	$\zeta \in R^{6+n}$
ζ_d	Desired configuration space velocities	$\zeta \in R^{6+n}$
Σ_I	Earth-fixed and inertial frame	
Σ_b	Body-fixed base frame	
Σ_{b_i}	Body-fixed frame of link i	
Σ_{ee}	Body-fixed end-effector frame	
Σ_{obj}	Body-fixed object frame	
J	Jacobian	$J \in R^{6\times(6+n)}$
$J^\dagger$	Jacobian pseudoinverse	$J \in R^{(6+n)\times6}$
ω_{ee}	Angular velocity of the end-effector in inertial frame	$\omega_{ee} \in R^3$
$M(q)$	Inertia matrix	$M(q) \in R^{(6+n)\times(6+n)}$
$C(q,\zeta)$	Coriolis and centripetal matrix	$C(q,\zeta) \in R^{(6+n)\times(6+n)}$
$D(q,\zeta)$	Hydrodynamic damping matrix	$D(q,\zeta) \in R^{(6+n)\times(6+n)}$
$g(q,\eta)$	Gravity and buoyancy effects	$g(q,\eta) \in R^{(6+n)}$
τ	Generalized forces and torques	$\tau \in R^{(6+n)}$
τ_J	Generalized forces and torques from joints	$\tau_J \in R^{(6+n)}$
τ_T	Generalized forces and torques from thrusters	$\tau_T \in R^{(6+n)}$
B	Thruster Control Matrix	$B \in R^{(6+n)\times m}$
B_b	First 6 rows of B	$B_b \in R^{6\times m}$

Symbols	Description	Vector or matrix
B_J	Last n rows of B	$B_J \in R^{n \times m}$
B_{tot}	Resulting TCM	$B_{tot} \in R^{(6+n)\times(6+m)}$
u_J	Joint control input	$u_J \in R^n$
' u_T	Thruster control input	$u_T \in R^m$
τ_c	Commanded forces and torques	$\tau_c \in R^{6+n}$
$\tau_{c,T}$	Commanded forces and torques on base	$\tau_c \in R^6$
$\tau_{c,J}$	Commanded forces and torques on joints	$\tau_c \in R^n$
d_i	Drag forces on link i	$d_i \in R^6$
$d_{L,i}$	Linear drag forces on link i	$d_{L,i} \in R^6$
$d_{N,i}$	Nonlinear drag forces on link i	$d_{N,i} \in R^6$
ρ	Density of water	
$C_{d,L}$	Linear cross-sectional drag coefficient	
$C_{d,C}$	Nonlinear crossflow drag coefficient	
$C_{d,1}$	Nonlinear drag coefficient in surge	
$C_{d,4}$	Nonlinear drag coefficient in roll	
J_v	Valve inertia	
k_μ	Valve viscosity coefficient	
k	Stiffness constant	
h_{ee}	Force and moment exerted by the end-effector	$h_{ee} \in R^6$
f_{ee}	Linear force exerted by the end-effector	$f_{ee} \in R^3$
μ_{ee}	Moment exerted by the end-effector	$\mu_{ee} \in R^3$
W	Weight matrix	$W \in R^{(6+n)\times(6+n)}$
s	Error variable/sliding variable	$s \in R^{6+n}$
s'	Error variable	$s' \in R^{6+n}$
$\tilde{y}$	Error variable	$s \in R^{6+n}$

Symbols	Description	Vector or matrix
$\tilde{\zeta}$	Error variable $\zeta - \zeta_d$	$\tilde{\zeta} \in R^{6+n}$
$\hat{\theta}$	Vector of estimates	
$\tilde{\theta}$	Estimate errors	
Λ	Design parameter AIDC	$\Lambda \in R^{(6+n)\times(6+n)}$
$\phi(q, \zeta, \zeta_r)$	Regressor matrix	$R^{(6+n)\times 4}$
λ	Design parameter in STA	
ϵ	Design parameter in STA	
ω_i	Design parameter in STA	$\omega \in R^{(6+n)}$
γ_i	Design parameter in STA	$\gamma \in R^{(6+n)}$
α_i	Adaptive gain in STA	$\alpha \in R^{(6+n)}$
β_i	Adaptive gain in STA	$\beta \in R^{(6+n)}$
α_m	Bound in STA	
f_i	Design parameter in NRAC	$f \in R^5$
κ	Design parameter in NRAC	
δ_i	Bound in NRAC	
α	Unknown parameter in NRAC	
β_i	Unknown parameters in NRAC	
γ	Unknown parameter in NRAC	
σ	Design parameter NRAC	
K_x	Various gain matrices	

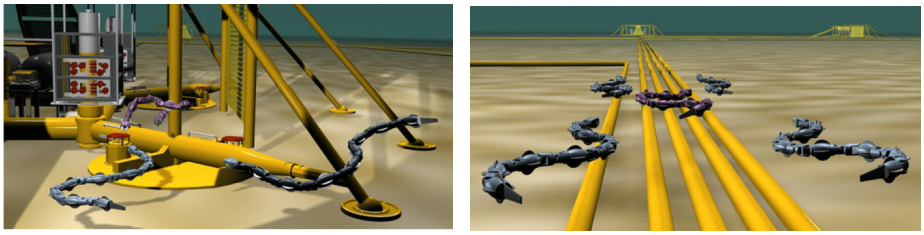
Chapter 1

Introduction

1.1 Motivation

An increasing amount of oil and gas operations are performed subsea and the existing subsea infrastructure is aging. As a result, the importance of subsea inspection, maintenance and repair (IMR) is increasing (Sverdrup-Thygeson et al.; 2016c). Most IMR operations today are performed with the support of offshore vessels and remotely operated underwater vehicle (ROV) systems which result in costly operations (Schjolberg et al.; 2016). Therefore, increasing autonomy has great potential to increase efficiency of IMR operations and thereby reduce costs (Schjolberg et al.; 2016).

The underwater swimming manipulator (USM) can replace the use of support vessels and ROVs for carrying out inspections and light intervention tasks such as cleaning and adjusting valves and chokes (Sverdrup-Thygeson et al.; 2016c). Its slender shape gives the USM the possibility to access the most narrow parts of subsea installations inaccessible to most autonomous underwater vehicles (AUVs) and ROVs due to their size (Sverdrup-Thygeson et al.; 2016b). In addition, the USM can work as a torpedo-shaped AUV for locomotion (Kelasidi et al.; 2017). Hence, the USM provides both efficient locomotion and service as a robot manipulator. As a result, the USM has the potential to significantly reduce costs related to IMR (Sverdrup-Thygeson et al.;



(a) Inspection and intervention. (b) Pipeline survey.

Figure 1.1: USRs performing inspection and intervention and pipeline survey. (Sverdrup-Thygeson et al.; 2018)

2016c).

This thesis focuses on dynamic motion control of USMs for inspection and intervention. Considerations such as uncertainties in the model knowledge must be taken into account when designing dynamic control approaches for USMs. Hydrodynamic forces on the USM are complex and highly nonlinear and therefore difficult to model. In this thesis, different dynamic motion control strategies for USMs and how they deal with uncertainties in the model are studied. For the USM to perform intervention tasks, the control force exchanged with the environment must be investigated (Antonelli; 2014). Modeling of interaction forces as well as interaction control for USM is investigated in this thesis. Figure 1.1 shows possible applications of the USM. The figure illustrates USRs, but USMs are even more appropriate for these tasks (Sverdrup-Thygeson et al.; 2018).

1.2 Background on AUVs for intervention tasks

Issues related to the use of ROVs for IMR operations are that they require a support vessel, constant supervision and are limited by the attached tether (Sverdrup-Thygeson et al.; 2018). Some of these challenges can be resolved by using AUVs and AUVs are nowadays routinely used for survey missions (Ridao et al.; 2014). Most commercially available AUVs are however not suited for subsea IMR operations (Sverdrup-Thygeson

et al.; 2018). Recently, some research projects have focused on development of interaction autonomous underwater vehicles (I-AUVs).

The first fully autonomous intervention at sea was performed by the ALIVE project (Evans et al.; 2003)(Palomeras et al.; 2014). The goal of the ALIVE project is "to develop an I-AUV capable of docking to a subsea structure which has not been specifically modified for AUV use" (Evans et al.; 2003). Once docked to a subsea panel, the manipulator works as a fixed base manipulator and operates valves on the panel (Perrier et al.; 2004). The ALIVE vehicle is shown in figure 1.2a.

The SAUVIM (Marani et al.; 2009) is an semi-autonomous intervention AUV (I-AUV) equipped with a manipulator (figure 1.2b). The vehicle is much heavier than the manipulator such that both systems behave practically decoupled. The SAUVIM is in (Marani et al.; 2009) used for recovery of a submerged target without human intervention, except for the initial decision of starting the operation, with promising results.

The GIRONA 500 light weight I-AUV, shown in figure 1.2c, is a compact size I-AUV lighter than previous I-AUVs (Ridao et al.; 2014). The I-AUV has capacity to reconfigure for different tasks. In (Ridao et al.; 2014) the GIRONA 500 I-AUV is used for docking and fixed-base manipulation, learning by demonstration for free-floating manipulation and multipurpose manipulation for object recovery. In (Palomeras et al.; 2014), the GIRONA 500 I-AUV is used for subsea panel docking followed by turning a valve and plugging/unplugging a connector.

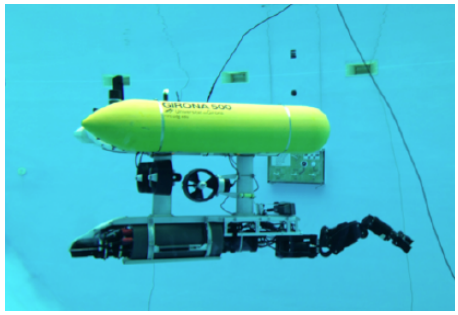
For the interested reader, (Ridao et al.; 2014) contains a summary of research projects relevant to intervention AUVs.

Underwater Vehicle Manipulator Systems (UVMSs) are underwater vehicles with one or more manipulators, i.e. a term describing both ROVs and AUVs equipped with manipulators. The books (Antonelli; 2014) and (Antonelli; 2018) contain more information on modeling and control of UVMSs.

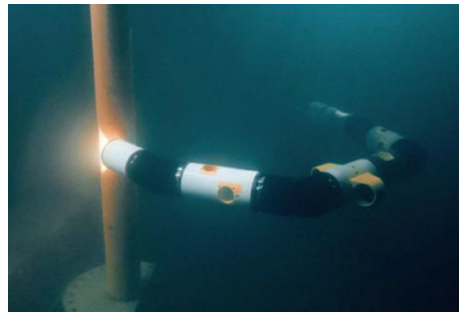
An underwater swimming manipulator (USM) is an innovative underwater vehicle that is a fusion between a conventional autonomous underwater vehicle and an underwater snake robot (USR) (Sverdrup-Thygeson et al.; 2016c). The Eelume USM is shown in figure 1.2d. USRs are biologically inspired robots that swim by mimicking

(a) The ALIVE vehicle (Perrier et al.; 2004).

(b) The SAUVIM (Marani et al.; 2009)



(c) The GIRONA I-AUV (Ridao et al.; 2014).



(d) The Eelume USM (Courtesy of Eelume).

Figure 1.2: Examples of I-AUVs.

the motion of eels (Kelasidi, Pettersen and Gravdahl; 2014). A USM is essentially a USR equipped with thrusters (Borlaug and Pettersen; 2017). USMs can also be considered as a special case of articulated I-AUVs. USMs are highly maneuverable and easy to deploy and operate (Sverdrup-Thygeson et al.; 2018). Their slender and flexible body give them the possibility to access areas difficult to access with other types of I-AUVs or UVMSs such as tight spaces. The USM has the possibility to swim like a biological eel. This is advantageous in case of thruster failure or in cases where thruster use is not recommended (Sverdrup-Thygeson et al.; 2018). This can for example be when doing inspection in an area where one wants to avoid sand whirling. "To realize operational USMs for inspection and light intervention tasks, several theoretical problems such as modeling, guidance, and control should be addressed " (Sverdrup-Thygeson et al.; 2018).

1.3 Problem description

This thesis addresses control of USMs. More specifically, it addresses control approaches for subsea inspection and intervention with underwater swimming manipulators. This is done through the following steps:

- Design and investigate dynamic control approaches for observation with a hovering robot.
- Verify the control approaches by creating a simulator for a floating-base manipulator.
- Design a controller for manipulation tasks.
- Extend the simulator to model contact forces with the environment.

1.4 Background and Contributions

This thesis presents and compares the performance of dynamic motion control strategies as well as interaction control strategies for USMs. The control strategies are tested

in simulations and the results are compared to the expected theoretical performance. Simulators have been implemented to model an underwater swimming manipulator (USM) with and without external forces based on a Matlab script with a model of the dynamic equations provided by PhD candidate Henrik Schmidt-Didlaukies. The script includes all functions necessary to implement the dynamic equations and for visualization of the USM. A summary of the main contributions of the thesis are:

- Implementation of a USM simulator in Simulink based on a Matlab script provided by Henrik Schmidt-Didlaukies (chapter 3).
- Implementation of thrust allocation (section 3.4.3).
- Implementation of differential inverse kinematics (section 3.4.2).
- Adaptation of three different dynamic motion controllers to be suitable for USMs. Stability analyses of the dynamic motion controllers have been performed and the controllers have been implemented (chapter 4).
- Modeling of interaction forces for the task of turning a valve (section 2.6).
- Extension of the simulator to include interaction forces (section 2.5/2.6).
- Design and implementation of two different interaction controllers for USMs (chapter 5).

The provided implementation of the dynamic model was intended for Matlab, so work was needed to adapt it to run in the Simulink environment. A complete control framework has been implemented with the USM simulator in Simulink including reference trajectory generation, inverse kinematics and thrust allocation in addition to dynamic motion control. When implementing the inverse kinematics and thrust allocation, functions from the provided script have been used to calculate the thrust configuration matrix and the Jacobian J_b defined in section 2.3.5. In the implementation of the dynamic motion controllers, some functions from the provided script have also been utilized. Interaction controllers are then added to the control framework with a simulator extended with interaction forces and moments. To include interaction

forces, the function calculating the Jacobian J_b from the provided script has been used. In addition to the script, PhD candidate Henrik Schmidt-Didlaukies provided a note on the theoretical background of the model, and he has been available for questions about the model and script.

The choice of controllers to investigate was done by studying literature and choosing controllers that are relevant for USMs that were not too complex to implement in the control framework. The adaptive inverse dynamics controller has been altered for ease of implementation, and the non-regressor-based adaptive controller has been altered to be relevant to USMs and for convenience in the stability analysis. The super-twisting algorithm with adaptive gains presented in the thesis is based on the super-twisting algorithm from my specialization project, but the sliding-surface is altered so that the controller is suitable for USMs. For interaction control, the controllers and modeling of interaction forces/moments are based on the article (Cataldi and Antonelli; 2015). In the article, the controllers are applied to an underwater vehicle manipulator system, while in this thesis they are applied to a USM.

Throughout the work on the thesis I have had discussions with my supervisor and co-supervisors on what direction to take, about the different control strategies and about how to best perform the stability analyses.

1.5 Outline

The thesis is divided into 6 chapters. After the introductory chapter, **chapter 2** presents the modeling background for this thesis. **Chapter 3** presents the structure and implementation of the simulator as well as the model parameters used for the simulations in the later chapters. In addition, chapter 3 presents the implemented control framework. In **chapter 4**, different dynamic motion control strategies are presented and simulation results for these controllers are presented and discussed. Stability analyses of the controllers are also included in chapter 4. Interaction control is presented in **chapter 5**. Chapter 5 also includes results of simulations with the interaction controllers and a discussion of the results. Finally, **chapter 6** contains conclusions and suggestions for further work.

Chapter 2

Modeling background

This chapter presents the modeling background for the thesis. First, the representation of position and orientation of a rigid body is presented. Then, the underwater swimming manipulator (USM) description is defined and important aspects of USM modeling is presented. Finally, modeling for interaction with the environment is introduced.

2.1 Rigid body representation

Consider a rigid body in space. The rigid body has a body-fixed frame Σ_b attached to it as shown in figure 2.1. The earth-fixed and inertial frame, Σ_I, is defined with the z-axis pointing upwards and is a right-handed coordinate system. This section contains an introduction to the position and orientation representation of rigid bodies used throughout the thesis.

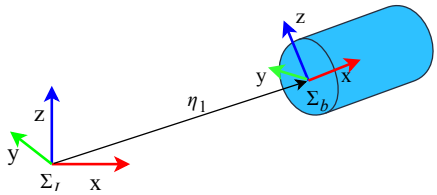


Figure 2.1: A rigid body in space with reference frames.

2.1.1 Position representation

The position of a rigid body with respect to an earth-fixed and inertial frame Σ_I, η_1 is shown in figure 2.1 and defined as

$$\eta_1 = [x \quad y \quad z]^\top, \tag{2.1}$$

where x, y, z (surge, sway, heave) are the coordinates of the origin of the body-fixed frame, Σ_b, with respect to the inertial frame in x-, y- and z-direction respectively. The linear velocity of the rigid body in the inertial frame is the time-derivative $\dot{\eta}_1$ which can be integrated to find η_1.

It is also useful to define the linear velocity of the body-fixed frame of the rigid body with respect to the inertial frame Σ_I expressed in the body-fixed frame Σ_b. Call this the body-fixed linear velocity (Antonelli; 2014):

$$\nu_1 = [u \quad v \quad w]^\top. \tag{2.2}$$

The relationship between the linear velocity of the rigid body expressed in the inertial frame and the body-fixed linear velocity is (Antonelli; 2014):

$$\nu_1 = R_I^b \dot{\eta}_1, \tag{2.3}$$

where R_I^b is the rotation matrix from the inertial frame to the body-fixed frame and will be defined in section 2.1.2.

2.1.2 Orientation representation

The orientation of a rigid body may be described by various representations (Chiaverini and Siciliano; 1999). The Euler angle representation and unit quaternion representation will be introduced in this section.

2.1.2.1 Euler angles

The Euler angle representations use three parameters to represent orientation and are therefore minimal representations (Chiaverini and Siciliano; 1999). The orientation of a rigid body can be defined by a vector of Euler angles (Antonelli; 2014) as

$$\eta_2 = [\phi \quad \theta \quad \psi]^\top. \tag{2.4}$$

where ϕ, θ and ψ (roll, pitch, yaw) are the rotations around the x-, y- and z-axis of the inertial frame respectively. There exists 12 possible different definitions of Euler angles (Chiaverini and Siciliano; 1999). The XYZ representation will from now on be used. The XYZ representation leads to the following rotation matrix R_I^b in (2.3) (Chiaverini and Siciliano; 1999):

$$R_I^b = R_x(\phi)R_y(\theta)R_z(\psi). \tag{2.5}$$

where R_x, R_y and R_z are the matrices of the elementary rotations about three independent axes of successive frames (Chiaverini and Siciliano; 1999). The rotation matrices are defined as (Fossen; 2011):

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix}, \; R_y(\theta) = \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix}, \; R_z(\psi) = \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

The abbreviations s_ϕ and c_ϕ are short for for $sin(\phi)$ and $cos(\phi)$ respectively.

The time derivative of the vector of Euler angles is $\dot{\eta}_2$. The time derivative of the

Euler angles does however not have a physical interpretation, but can be integrated to get the Euler angles η_2 (Antonelli; 2014).

As is done for the position representation, define ν_2 as the vector of body-fixed angular velocities. The body-fixed angular velocities are the angular velocities of the body-fixed frame with respect to the inertial frame expressed in the body-fixed frame:

$$\nu_2 = [p \quad q \quad r]^\top. \tag{2.7}$$

The relationship between ν_2 and $\dot{\eta}_2$ is described by (Chiaverini and Siciliano; 1999):

$$\nu_2 = J_{k,o}(\eta_2)\dot{\eta}_2 \tag{2.8}$$

where $J_{k,o}(\eta_2)$ is a proper Jacobian matrix described with the XYZ Euler angle convention as (Chiaverini and Siciliano; 1999):

$$J_{k,o}(\eta_2) = \begin{bmatrix} 1 & 0 & s_\theta \\ 0 & c_\phi & -c_\theta s_\phi \\ 0 & s_\phi & c_\theta c_\phi \end{bmatrix}, \tag{2.9}$$

where ϕ and θ are the roll and pitch angles respectively (2.4). The inverse of the matrix $J_{k,o}(\eta_2)$ is

$$J_{k,o}^{-1}(\eta_2) = \frac{1}{c_\theta}\begin{bmatrix} c_\theta & s_\theta s_\phi & -s_\theta c_\phi \\ 0 & c_\theta c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\phi \end{bmatrix}. \tag{2.10}$$

It can easily be seen that the matrix is singular for $cos\theta = 0$ and hence the representation is singular for pitch angles $\theta = \pm\frac{\pi}{2}$. These singularities are called *representation singularities* (Chiaverini and Siciliano; 1999).

2.1.2.2 Unit quaternions

While the Euler angle representations use 3 parameters to represent orientation, the unit quaternion representation consists of 4 parameters and 1 norm constraint and

is therefore non-minimal (Chiaverini and Siciliano; 1999). Using unit quaternions to represent orientation avoids the problem of representation singularities (Antonelli; 2014).

The unit quaternion vector can be defined as (Egeland and Gravdahl; 2002)

$$p = \begin{bmatrix} \eta_Q \\ \epsilon \end{bmatrix}, \tag{2.11}$$

where

$$\eta_Q = cos\Big(\frac{\beta}{2}\Big) \tag{2.12}$$

is the real part of the quaternion, and

$$\epsilon = [\epsilon_1, \epsilon_2, \epsilon_3]^\top = sin\Big(\frac{\beta}{2}\Big)\mathbf{r} \tag{2.13}$$

are the imaginary parts of the quaternion. β is the rotation angle about the axis described by $\boldsymbol{r} \in R^3$ (Chiaverini and Siciliano; 1999). Hence, every rotation is described by one angle and one axis. The angle β is defined in $\beta \in [-\pi, \pi]$ such that $\eta_Q > 0$ (Chiaverini and Siciliano; 1999). Remark: The quaternion is defined with the scalar part as the first element of the vector as in (Egeland and Gravdahl; 2002), but opposite from (Antonelli; 2014).

The unit quaternion has unit length. Hence, the unit quaternion satisfy the relation (Egeland and Gravdahl; 2002):

$$p^\top p = \eta_Q^2 + \epsilon^\top \epsilon = 1. \tag{2.14}$$

The quaternion propagation equations give the relationship between the body-fixed angular velocity v_2 and the time derivative of the quaternion. The quaternion

propagation equations are defined as (Antonelli; 2014):

$$\begin{aligned} \dot{\eta}_Q &= -\frac{1}{2}\epsilon^\top \nu_2, \\ \dot{\epsilon} &= \frac{1}{2}\eta_Q \nu_2 + \frac{1}{2}\boldsymbol{S}(\epsilon)\nu_2. \end{aligned} \tag{2.15}$$

$\boldsymbol{S(\lambda)}$ is the skew-symmetric cross-product operator defined as (Fossen; 2011)

$$\mathbf{S}(\lambda) = -\boldsymbol{S}^\top(\lambda) = \begin{bmatrix} 0 & -\lambda_3 & \lambda_2 \\ \lambda_3 & 0 & -\lambda_1 \\ -\lambda_2 & \lambda_1 & 0 \end{bmatrix}, \quad \lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix}. \tag{2.16}$$

such that $\lambda \times a = \boldsymbol{S}(\lambda)a$. The quaternion propagation equation (2.15) can be written in matrix form as:

$$\begin{bmatrix} \dot{\eta}_Q \\ \dot{\epsilon} \end{bmatrix} = J_{k,oq}(p)\nu_2 = \frac{1}{2}\begin{bmatrix} -\epsilon_1 & -\epsilon_2 & -\epsilon_3 \\ \eta_Q & -\epsilon_3 & \epsilon_2 \\ \epsilon_3 & \eta_Q & -\epsilon_1 \\ -\epsilon_2 & \epsilon_1 & \eta_Q \end{bmatrix}\nu_2. \tag{2.17}$$

It can be seen that $J_{k,oq}^\top J_{k,oq} = 0.25\mathbf{I}_3$ and equation (2.17) can therefore be rewritten on a form similar to that with Euler angles in equation (2.8) (Antonelli; 2014):

$$\nu_2 = 4J_{k,oq}^\top(p)\begin{bmatrix} \dot{\eta}_Q \\ \dot{\epsilon} \end{bmatrix}. \tag{2.18}$$

The unit quaternion vector p has unit length. Numerical integration of the time derivative of the quaternion from (2.17) may cause the resulting quaternion norm to deviate from unity (Fossen; 2011). A nonlinear feedback term may therefore be included in (2.17):

$$\begin{bmatrix} \dot{\eta}_Q \\ \dot{\epsilon} \end{bmatrix} = J_{k,oq}(p)\nu_2, + \frac{\gamma}{2}(1 - p^\top p)p \tag{2.19}$$

where $\gamma \geq 0$ (Fossen; 2011).

2.2 Underwater swimming manipulator description

An underwater swimming manipulator (USM) can be considered as a free-floating, fully submerged serial chain robot manipulator with thrusters attached to the links (Sverdrup-Thygeson et al.; 2016a). Properties of the USM will be defined in the following.

The USM consists of $n + 1$ rigid cylindrical links interconnected by n motorized joints and is equipped with m thrusters (Sverdrup-Thygeson et al.; 2016a). Figure 2.2 shows an example USM with $n + 1$ links, n joints and an end-effector mounted on the head link. The end-effector is the part of the USM that interacts with the environment. For different types of tasks, different end-effectors can be mounted on the USM.

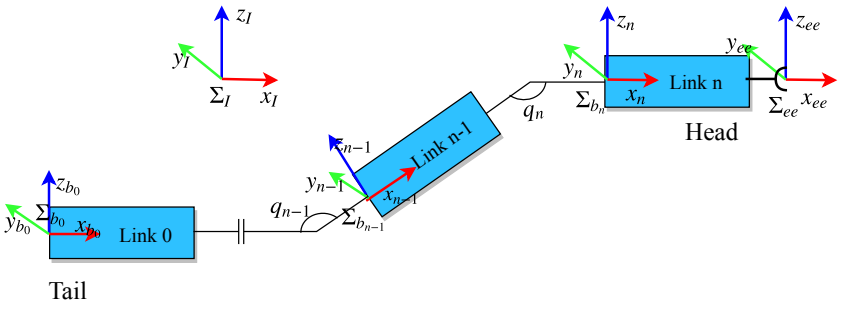


Figure 2.2: A general USM representation.

The links are numbered 0 to n where link 0 is the tail, or base, link and link n is the head link of the USM as shown in figure 2.2. The links have circular cross-sections and centroids placed on the x-axis of the link's body fixed frame (Schmidt-Didlaukies; 2018). Joints are numbered 1 to n where joint i, $i = 1...n$, interconnects link i and link $i - 1$. All joints are revolute with 1 DOF and can therefore be described by a single joint variable q_i where q_i is the joint angle of link i. The joints shown in figure 2.2 are y-revolute with examples of joint angles marked q_n and q_{n-1} in the figure.

The following reference frames will be referred to when considering the USM. All frames are shown in figure 2.2 and all frames have y-axes (green) pointing into the plane such that the coordinate systems are right-handed.

- Earth-fixed and inertial frame, Σ_I. From now on referred to as the inertial frame.
- Body-fixed frame of link i, Σ_{b_i}. All links have a body-fixed frame attached to them. The body-fixed frame of link i is fixed to the center line of the link on the side closest to the tail. The x-axis of the body-fixed frame points along the center line of the cylinder as shown in figure 2.2. Σ_{b_i} is referred to as the body-frame of link i.
- The body-fixed base frame Σ_{b_0}. From now on referred to as the base frame Σ_b.
- The end-effector frame Σ_{ee}. An additional frame is located at the tip of the end-effector as shown in figure 2.2.

The joint i coincides with the the origin of the body-frame of link i. In figure 2.2 the length of the joints are extended to show the generalized variable q_i.

Thruster j on link i has a thrust direction $\beta_{t,i,j}$ and a point of attack $r_{t,i,j}$ expressed in the body frame of link i associated with it. The position and orientation of the thrusters relative to the base of the USM is dependent on the joint angles q (Schmidt-Didlaukies; 2018).

The motion of the USM can be described by the body-fixed linear and angular velocity of the base-frame Σ_b, which from now on will be denoted by ν_1 and ν_2 respectively, as well as the joint angular velocities $\dot{q}$. These are collected in a vector ζ:

$$\zeta = \begin{bmatrix} \nu_1 \\ \nu_2 \\ \dot{q} \end{bmatrix} \in R^{6+n}. \tag{2.20}$$

The position and orientation of the base-frame Σ_b with respect to the inertial frame Σ_I is denoted by η_1 and η_2, respectively. These are collected in the vector $\eta = [\eta_1^\top, \eta_2^\top]^\top$.

The frame Σ_{ee} is attached to the tip of the end-effector. The position and orientation of the end-effector-frame Σ_{ee} with respect to the inertial frame Σ_I is denoted by $\eta_{ee} = [\eta_{1,ee}^\top, \eta_{2,ee}^\top]^\top$.

2.3 Kinematic modeling

Kinematic modeling is the description of the USMs motion without considering the forces that cause the motion (de Wit et al.; 2012). Direct kinematics can be used to describe the end-effector motion with respect to the motion of the joints and base. Inverse kinematics is the opposite, to find the base and joint motion that correspond to a desired end-effector motion (de Wit et al.; 2012). Hence, kinematics consists of the transformation between configuration space and task space or vice versa.

2.3.1 Configuration space and task space

Consider some task to be executed. The task is as usually and most conveniently described by the position and orientation of the end-effector (de Wit et al.; 2012). The position and orientation of the end-effector can be described in the *task space* as $\eta_{ee} = [\eta_{2,ee}^\top, \eta_{1,ee}^\top]^\top$. The dimension of the *task space* depends on the geometry of the task, but may not be larger than 6 (de Wit et al.; 2012), that is, three degrees of freedom (DOF) to specify position and three DOF to specify orientation.

The *configuration space* is the space in which the complete and unambiguous configuration of the USM is described. All joints have 1 DOF and the configuration of the USM can be described by n joint variables and 6 variables to specify orientation and rotation of the base. The dimension of the configuration space is therefore $6 + n$ (n number of joints). While the tasks are described in task space, a manipulator is naturally actuated in the configuration space (Siciliano and Khatib; 2007).

2.3.2 Kinematic redundancy

Kinematic redundancy is a relative term. It arises when a manipulator has more degrees of freedom (DOF) than what is required to execute a given task (Siciliano and

Khatib; 2007). Hence, when the dimension of the configuration space is larger than the dimension of the task space. If the task space has 6 DOFs, a manipulator with 7 or more DOF is kinematically redundant. However, a manipulator with only 6 DOF may also be kinematically redundant if the dimension of the task space is less than 6. A USM is therefore kinematically redundant because of the 6 DOF of the base as well as the n>0 DOF provided by the joints. The Jacobian will therefore always have full rank (Siciliano and Khatib; 2007).

2.3.3 Direct kinematics

The position and orientation of the end-effector η_{ee} can be completely described as a function of the base position and orientation η and joint variables q (Antonelli; 2014). This relationship is called the *direct kinematics*:

$$\eta_{ee} = k(\eta, q). \tag{2.21}$$

The function $k(\eta, q)$ is nonlinear may be very complex (Antonelli; 2014).

2.3.4 Differential kinematics

A task is as previously mentioned usually described by desired positions and orientations of the end-effector η_{ee}. Inverse kinematics consists of finding desired base and joint trajectories that give the desired end-effector trajectory. This essentially means to find the inverse of equation 2.21.

The function $k(\eta, q)$ in equation (2.21) is however invertible only in certain cases. Therefore, *differential kinematics* may be used to calculate the desired base and joint trajectories instead. The differential kinematics equation can be obtained by taking the time derivative of equation (2.21). This results in a *linear* mapping between configuration space velocities and the task space velocities (Siciliano and Khatib; 2007):

$$\dot{\eta}_{ee} = J(R^I_{ee}, q)\zeta. \tag{2.22}$$

where $J(R^I_{ee}, q) \in R^{6\times(6+n)}$. The linearity of this mapping makes it useful for solving

the inverse kinematics problem (de Wit et al.; 2012).

2.3.5 Jacobians

In the differential kinematics equation (2.22), the base and joint velocities are related to the time derivative of end-effector position and orientation through the Jacobian. Depending on how the time-derivatives of the end-effector position and orientation is described, different Jacobians can be defined.

First, consider the body-fixed linear and angular velocities of the end-effector, $\nu_{1,ee}$ and $\nu_{2,ee}$. The Jacobian $J_b(q)$ that gives the body fixed velocities from the base and joint velocities ζ is $J_b(q)$ (Schmidt-Didlaukies; 2018):

$$\begin{bmatrix} \nu_{1,ee} \\ \nu_{2,ee} \end{bmatrix} = J_b(q)\zeta. \tag{2.23}$$

The body-fixed linear and angular velocities of any link i, ν_{1,b_i} and ν_{2,b_i}, is related to ζ through the Jacobien $J_{b,i}$:

$$\begin{bmatrix} \nu_{1,b_i} \\ \nu_{2,b_i} \end{bmatrix} = J_{b,i}(q)\zeta. \tag{2.24}$$

These Jacobians are available in the provided Matlab script defining the USM model in this thesis.

As a task is usually described by desired positions and orientations in the inertial frame η_{ee}, it is useful to define the Jacobian $J(R^I_{ee}, q)$ that relates the time derivative of position and orientation in the inertial frame $\dot{\eta}_{ee}$ to the base and joint velocities ζ. Equation (2.3) and (2.8) describes the relations between ν_1 and $\dot{\eta}_1$ and ν_2 and $\dot{\eta}_2$ of a rigid-body respectively. These are used to define a Jacobian $J(R^I_{ee}, q)$:

$$\dot{\eta}_{ee} = \begin{bmatrix} \dot{\eta}_{1,ee} \\ \dot{\eta}_{2,ee} \end{bmatrix} = \begin{bmatrix} R^I_{ee} & 0_{3\times3} \\ 0_{3\times3} & J^{-1}_{k,o}(\eta_{2,ee}) \end{bmatrix} J_b(q)\zeta = J(R^I_{ee}, q)\zeta \tag{2.25}$$

It is also useful to define a Jacobian $J_\omega(R^I_{ee}, q)$ that relates the linear and angular

velocities of the end-effector in the inertial frame to the base and joint velocities ζ:

$$\begin{bmatrix} \dot{\eta}_{1,ee} \\ \omega_{ee} \end{bmatrix} = \begin{bmatrix} R^I_{ee} & 0_{3\times3} \\ 0_{3\times3} & R^I_{ee} \end{bmatrix} J_b(q)\zeta = J_\omega(R^I_{ee}, q)\zeta \,. \tag{2.26}$$

$\omega_{ee} = R^I_{ee} v_{2,ee}$ is the angular velocity of the end-effector frame with respect to the inertial frame expressed in the inertial frame.

2.3.6 Kinematic singularities

A kinematic singularity arises at configurations where the Jacobian is rank deficient (Siciliano and Khatib; 2007). At a kinematic singularity the mobility of a manipulator is reduced and infinite solutions to the inverse kinematics problems might exist. Close to a kinematic singularity small velocities in task space may correspond to large velocities in the configuration space (Antonelli; 2014).

2.4 Dynamic modeling

The USM is considered as free floating serial chain manipulator. The equations of motion are therefore structurally similar to those of underwater vehicle manipulator systems (UVMSs) presented in (Antonelli; 2018). The vector ζ was defined in 2.20. The model used in the modeling of a USM in this thesis is based on (Schmidt-Didlaukies; 2018). The equations of motion (EOMs) can be written as:

$$M(q)\dot{\zeta} + C(q,\zeta)\zeta + D(q,\zeta)\zeta + g(q,\eta) = \tau \tag{2.27}$$

where:

$M(q) \in R^{(6+n)\times(6+n)}$ is the inertia matrix including added mass terms,

$C(q,\zeta)\zeta \in R^{6+n}$ is the vector of Coriolis and centripetal terms,

$D(q,\zeta)\zeta \in R^{6+n}$ is the vector of hydrodynamic damping terms,

$g(q, R_B^I) \in R^{6+n}$ is the vector of gravity and buoyancy effects,

$\tau \in R^{6+n}$ is the vector of generalized forces and torques.

The derivation of the matrices may be found in (Schmidt-Didlaukies; 2018).

2.4.1 Generalized forces and torques

The generalized forces and torques τ that are applied to the USM are provided by joint motors and thrusters. The vector of generalized forces and torques τ can be divided into joint torques τ_J and generalized forces and torques from thrusters τ_T (Schmidt-Didlaukies; 2018):

$$\tau = \tau_J + \tau_T = Bu_T + \begin{bmatrix} 0_{6\times1} \\ u_J \end{bmatrix} \tag{2.28}$$

where u_T is the thruster control input and u_J is the joint control input.

2.4.2 Thrusters

The USM is equipped with thrusters. As opposed to conventional underwater robots, the position of the thrusters relative to the base is dependent on the joint angles (Sverdrup-Thygeson et al.; 2018). The thrusters provide generalized forces and moments, $\tau_T \in R^{6+n}$, that act on the USM. The thruster forces, τ_T, affect the position and orientation of the base as well as the joint angles. The control input $u_T \in R^m$ is applied to the thrusters to achieve the desired force and moments on the base, while joints angles are controlled by the joint motors. The thruster forces τ_T corresponding to a control input u_T depend on structural variables of the thrusters as well as the density of water (Antonelli; 2014). The relationship between the thruster forces τ_T and the control input u_T is as a result highly nonlinear (Antonelli; 2018). The relationship can

however be simplified to a linear relationship:

$$\tau_T = Bu_T \tag{2.29}$$

where $B \in R^{(6+n)\times m}$ with m thrusters and n joints is the Thrust Configuration Matrix (TCM) (Schmidt-Didlaukies; 2018). The TCM is assumed to be constant Schmidt-Didlaukies (2018).

The derivation of B is presented in the following. Each link has a matrix associated with it to describe how the thrusters excert forces and torques on that link. Call this matrix $B_i \in R^{6\times m_i}$ where m_i is the number of thrusters of link i. The matrix B_i is found from the thrust direction $\beta_{t,i,j}$ and the point of attack $r_{t,i,j}$ on the jth thruster of link i expressed in the frame of link i. B_i can then be written as (Schmidt-Didlaukies; 2018):

$$B_i = \begin{bmatrix} \beta_{t,i,1} & \dots & \beta_{t,i,m_i} \\ r_{t,i,1} \times \beta_{t,i,1} & \dots & r_{t,i,m_i} \times \beta_{t,i,m_i} \end{bmatrix} \tag{2.30}$$

The forces and torques on link i, $\tau_i \in R^6$, provided by the control input on the thrusters on link i, $u_i \in R^{m_i}$, is (Antonelli; 2018):

$$\tau_i = B_i u_i \tag{2.31}$$

The forces and torques on link i translates to forces and torques on the base as well as joint torques through the transpose of the body Jacobian $J_{b,i}(q)$:

$$\tau_{T,i} = J_{b,i}^{\top} \tau_i \tag{2.32}$$

$\tau_{T,i}$ are the generalized forces and torques on the USM from the thrusters on link i. The forces and torques on the base and joints from all the thrusters can therefore be expressed as:

$$\tau_T = Bu_t = \begin{bmatrix} J_{b,1}(q)^{\top} B_1 & J_{b,2}(q)^{\top} B_2 & \dots & J_{b,n}(q)^{\top} B_n \end{bmatrix} u_t \tag{2.33}$$

where B it the total TCM.

2.4.3 Hydrodynamic forces

The hydrodynamic forces induced by motion of the USM in water are complex and highly nonlinear (Kelasidi, Pettersen, Gravdahl and Liljebäck; 2014). Several assumptions are therefore made in the modeling of hydrodynamic forces. In the modeling of hydrodynamic forces according to the model in (Schmidt-Didlaukies; 2018), the following assumptions are made. The cross-flow principle is taken to be valid and the acting forces are assumed to be given by an expression resembling that of Morison (Morison; 1950)(Schmidt-Didlaukies; 2018). In addition, no current effects are considered in this thesis.

First, the drag forces experienced by a cylindrical link i are presented. Call the linear drag forces on link i $d_{L,i}$. The linear drag forces on link i are given as (Schmidt-Didlaukies; 2018):

$$d_{L,i} = D_{L,i} \begin{bmatrix} v_{1,i} \\ v_{2,i} \end{bmatrix}, \tag{2.34}$$

where

$$D_{L,i} = \rho \pi r l_i C_{d,L} v_{ref} \begin{bmatrix} \alpha_i & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \frac{1}{2} l_i \\ 0 & 0 & 1 & 0 & -\frac{1}{2} l_i & 0 \\ 0 & 0 & 0 & \gamma_i r^2 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} l_i & 0 & \frac{1}{3} l_i^2 & 0 \\ 0 & \frac{1}{2} l_i & 0 & 0 & 0 & \frac{1}{3} l_i^2 \end{bmatrix}. \tag{2.35}$$

$C_{d,L}$ is the linear drag coefficient, ρ is the density of water, r is the radius of the link and l_i is the length of the link i. v_{ref}, α_i, and γ_i are constants.

In addition to the linear drag forces, nonlinear drag forces are included in the model. The nonlinear drag forces in surge $d_{1,N,i}$ and yaw $d_{4,N,i}$ on link i are given as

$$d_{1,N,i} = -\frac{1}{2} \rho \pi r^2 C_{d,1} |v_{1,i}| v_{1,i}, \tag{2.36}$$

and

$$d_{4,N,i} = -\frac{1}{2}\rho\pi r^4 l_i C_{d,4}|v_{4,i}|v_{4,i} \tag{2.37}$$

respectively (Schmidt-Didlaukies; 2018). $C_{d,1}$ is the nonlinear drag coefficient in surge and $C_{d,4}$ is the nonlinear drag coefficient in roll. Note that current velocity is assumed to be zero in this thesis. The remaining nonlinear drag forces are calculated as in (McMillan et al.; 1995). The total drag forces on link i is the sum of the linear and nonlinear drag forces:

$$d_i = d_{L,i} + d_{N,i}. \tag{2.38}$$

In the modeling of hydrodynamic forces, the drag coefficients are chosen as in (Schmidt-Didlaukies; 2018):

$$C_{d,1} = 0.2, \quad C_{d,4} = 0.1, \quad C_{d,C} = 0.2, \quad C_{d,L} = 0.1. \tag{2.39}$$

$C_{d,L}$ is the linear cross-sectional drag coefficient, $C_{d,C}$ is the nonlinear crossflow drag coefficient and $C_{d,1}$ and $C_{d,4}$ are the nonlinear drag coefficient in surge and roll respectively.

The hydrodynamic damping matrix can be found from (Schmidt-Didlaukies; 2018)

$$D(q,\zeta) = \sum_{i=0}^{n} J_{b,i}(q)^\top D_i(q,\zeta) J_{b,i}(q), \tag{2.40}$$

where $D_i(q,\zeta)$ is the hydrodynamic damping matrix of link i and $J_{b,i}$ is defined in (2.24). The total hydrodynamic damping forces $D(q,\zeta)\zeta$ can therefore be written as:

$$\begin{aligned} D(q,\zeta)\zeta &= \sum_{i=0}^{n} J_{b,i}(q)^\top D_i(q,\zeta) J_{b,i}(q)\zeta \\ &= \sum_{i=0}^{n} J_{b,i}(q)^\top D_i(q,\zeta) \begin{bmatrix} v_{1,b_i} \\ v_{2,b_i} \end{bmatrix} = \sum_{i=0}^{n} J_{b,i}(q)^\top d_i \end{aligned} \tag{2.41}$$

where d_i are the total drag forces on link i (2.38).

2.4.4 Properties of the dynamic model

The dynamic model (2.27) has some important properties that will be used in the stability analyses of the dynamic motion controllers in chapter 4.

Property 2.1 (Antonelli; 2014, p. 56)
The inertia matrix is symmetric, positive definite. Moreover, it satisfies:

$$\lambda_{min}(M) \leq \|M\| \leq \lambda_{max}(M), \tag{2.42}$$

where $\lambda_{min}(M)$ and $\lambda_{max}(M)$ are the minimum and maximum eigenvalues of M.

Property 2.2 (Antonelli; 2014, p. 57)
The matrix $N(q,\zeta) = \dot{M}(q) - 2C(q,\zeta)$ is skew-symmetric for a particular choice of parametrization $C(q,\zeta)$ i.e.

$$z^\top N(q,\zeta)z = 0 \tag{2.43}$$

for any $(6+n)\times 1$ vector z.

Property 2.3 (Antonelli; 2014, p. 57)
The matrix $C(q,\zeta)$ satisfies

$$||C(a,b)c|| \leq c_0\|b\|\|c\| \tag{2.44}$$

for some bounded c_0.

Property 2.4 (Antonelli; 2014, p. 57)
The matrix $D(q,\zeta)$ is positive definite.

$$D(q,\dot{q}) > 0 \tag{2.45}$$

Property 2.5 (Siciliano and Khatib; 2007, p. 134)
The gravity vector $g(q)$ satisfies

$$||g(q,\eta)|| \leq g_0 \tag{2.46}$$

for some bounded constant g_0.

2.5 Contact with the environment

For a USM to successfully perform a manipulation task, it must handle the interaction between the USM and the environment (Siciliano et al.; 2008). The interaction of the USM with the environment can be described by the contact force at the end-effector. Define the forces and moments applied by the end-effector expressed in the inertial frame as:

$$h_{ee} = \begin{bmatrix} f_{ee} \\ \mu_{ee} \end{bmatrix} \tag{2.47}$$

where $f_{ee} \in R^3$ are the linear forces at the end-effector and $\mu_{ee} \in R^3$ are the moments at the end-effector. h_{ee} are the forces and torques exerted *by* the end-effector of the environment (Siciliano et al.; 2008). The forces and torques experienced by the end-effector is therefore $-h_{ee}$. The force exerted by the end-effector on the valve in x-direction $h_{ee,x}$ when the end-effector pushes on a valve is illustrated in figure 2.3.

When forces and moments are applied to the tip of the manipulator, the forces and moments propagate through the entire structure and affects the position and orientation of the base as well as the joint angles. The transpose of the Jacobian is used to describe how the contact forces affect the base and joints of the USM. The contact forces can be included in the EOM of the USM (2.27) as (Antonelli; 2014):

$$M(q)\dot{\zeta} + C(q,\zeta)\zeta + D(q,\zeta)\zeta + g(q,R^I_{ee}) = \tau - J^{\top}_{\omega}(q,R^I_{ee})h_{ee} \tag{2.48}$$

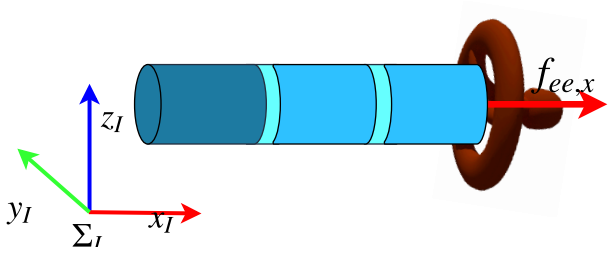


Figure 2.3: An example of force exerted by the end-effector.

where $J_\omega(q, R^I_{ee})$ is defined in (2.26).

2.6 Interaction forces and moments

Modeling of interaction forces are based on (Antonelli; 2014) and (Cataldi and Antonelli; 2015). The thesis focuses on control and quite simple models for interaction forces are considered to be sufficient. The interaction forces and moments considered are for the operation of turning a valve. Both linear interaction forces and moments are included in the modeling of interaction forces for the operation of turning a valve.

2.6.1 Interaction moment

To turn a valve, the end-effector must apply moment around the axis of rotation of the valve. The valve has an object-fixed frame Σ_{obj} attached to it as shown in figure 2.4. It is assumed that the x-axis of the object fixed frame is the axis of rotation and that it is aligned with the x-axis of the end-effector frame while turning the valve. It is also assumed that the x-axis of the object frame is aligned with the x-axis of the inertial frame. The moment applied by the end-effector causes a rotation as shown in figure 2.4.

The valve is modeled by the equation (Cataldi and Antonelli; 2015):

$$h_v = J_v \dot{\omega}_v + k_\mu \omega_v \tag{2.49}$$

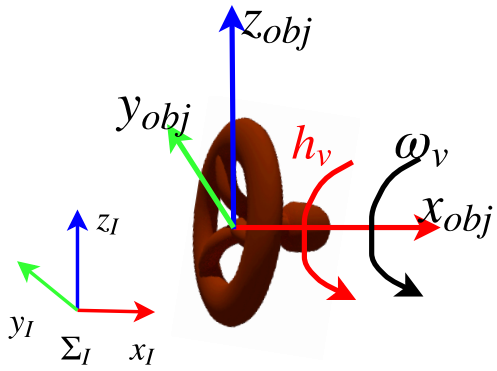


Figure 2.4: An example of moment exerted by the end-effector.

where J_v is the inertia around the axis of rotation, k_μ is the viscosity coefficient, ω_v is the angular velocity of the object fixed frame with respect to the inertial frame and $\dot{\omega}_v$ is its time derivative. The moment provided by the end-effector is h_v. The end-effector frame and the object fixed frame are coinciding throughout the task so that for a rotation about the x-axis $\omega_v = \omega_{ee,x}$. The vector of moments applied by the the end-effector when turning the valve about the x-axis expressed in the inertial frame is therefore given by $\mu_{ee} = [h_v, 0, 0]^\top$.

2.6.2 Linear interaction force

In addition to the moment applied to the USM while turning the valve, a linear force is modeled that works in positive x_{obj} direction. It is assumed that the x-axis of the end-effector frame is aligned with the x-axis of the object frame. If a valve is assumed to be mounted on a wall or a subsea panel, the end-effector will exert a force in the positive x_{obj} direction when attempting to move past the valve. The force exerted by the end-effector is visualized in figure 2.3. The yz plane in the object frame is modeled as a frictionless and elastically compliant plane for simplicity. The force exerted by the end-effector expressed in the inertial frame can then be described by (Antonelli; 2014):

$$f_{ee,x} = \begin{cases} k(x - x_v) & \text{if } x \geq x_v \\ 0 & \text{if } x < x_v \end{cases} \tag{2.50}$$

where x is the position of the end-effector expressed in the inertial frame and x_v is the x-coordinate of the valve position expressed in the inertial frame. k is a positive stiffness constant.

Chapter 3

Model and Simulator Implementation

This chapter presents the model used in the simulations in the thesis as well as background on how the simulator is implemented. Implementation of the control framework is presented and considerations that must be taken into account when designing control strategies for USMs are discussed.

3.1 Simulation model

Figure 3.1 shows a visual representation of the USM model that is used in the simulations in this thesis. The USM model is based on the Matlab script provided by PhD candidate Henrik Schmidt-Didlaukies. The USM consists of 9 rigid links and 8 revolute joints and has a total of $m = 10$ thrusters. The USM has a total length of $4.08m$ and total weight of $126.06kg$.

The USM consists of 3 different types of links shown by color coding in figure 3.1. Table 3.1 summarize the properties of the link types. In figure 3.1, only the body-fixed frame of the base link and link 4 is shown for readability, but all links have body-fixed

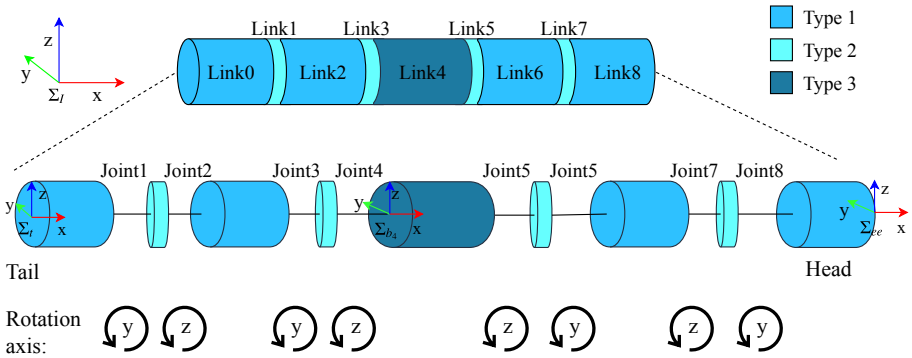


Figure 3.1: The USM model used for simulations.

frames associated with them. Note that all y-axes (in green) point into the plane to give right-handed coordinate systems. The end-effector frame is located at the end of the head link as shown in the figure. The position and orientation of the end-effector frame is what will be specified as the input to the control framework.

Joints are either y- or z-revolute about the body-fixed axis of the previous link. The rotation axis of each joint is indicated at the bottom of figure 3.1. The type 2 links are short with small mass and no drag associated with them. Type 2 links and the two joints on each side of the type 2 links model joints with 2 DOFs i.e. joints that can rotate about two axes.

Table 3.1: Link properties

	Length [m]	Mass [kg]	No. of thrusters	Thruster directions
Type 1	0.75	23.56	2	y & z
Type 2	0.02	0.10	0	-
Type 3	1.00	31.42	2	z & z

3.2 Simulation model for interaction

For USM interaction with the environment an end-effector is added to the model presented above (section 3.1).

For this thesis the task will be to rotate a valve, and the end-effector is therefore a gripper. To model this, a type 2 link is added to the head on the USM as shown in figure 3.2. This results in a quite small end-effector of length 2 cm and weight 100g. The end-effector should for the task of turning a valve be able to rotate around the body-fixed x-axis of the previous link. The joint between link 8 and link 9 is therefore implemented as an x-revolute joint.

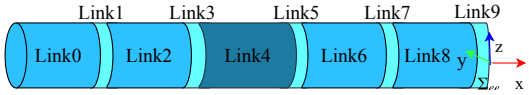


Figure 3.2: The USM with an end-effector used for interaction.

3.3 Implementation of the model

The script provided by PhD candidate Henrik Schmidt-Didlaukies is intended for Matlab. Modifications to the script have therefore been done when adapting it to Simulink. Examples of modifications that have been done are that the code has script changed to no longer include function handles in class definitions and functions no longer have a variable number of inputs or outputs.

The resulting script is implemented as a simulator in Simulink. The Simulink block that is shown in figure 3.3 contains the simulator. The available outputs from the simulator are ζ, the joint angles q, the position of the base frame η_1 and the orientation of the base frame represented by quaternions = p_{b_0}. In addition, the derivative of these values are collected in the output x_dot. The inputs to the simulator are the thrust u_T and the joint torque u_J.

Additional equations can be implemented to get additional outputs of the simulator. The differential kinematics equation (2.25) is implemented to calculate $\dot{\eta}_{1,ee}$ and $\dot{\eta}_{2,ee}$

and the results are integrated to get $\eta_{1,ee}$ and $\eta_{2,ee}$. The Euler angles $\eta_{2,ee}$ is converted to quaternions p_{ee} when needed. The angular velocity of the end-effector in the inertial frame ω_{ee} is found by implementing equation (2.26).

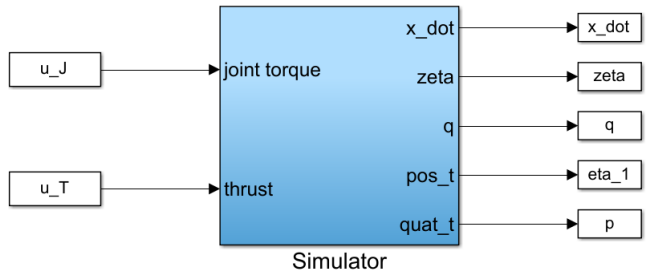


Figure 3.3: The implemented simulator in Simulink.

For simulations with interaction control, the simulator has been extended to include interaction forces (2.48). To do this, the term $-J_\omega^\top(q, R_{ee}^I)h_{ee}$ is added to the model from (Schmidt-Didlaukies; 2018). The simulator is implemented such that the term $-J_\omega^\top(q, R_{ee}^I)h_{ee}$ is only active when the end-effector is in close proximity to the valve position. The simulations in this thesis is based on the operation of turning a valve. For this operation both interaction moments and linear interaction forces are modeled as presented in section 2.6.2 and 2.6.1 respectively. A force measurement is implemented in the simulator by creating an output h_{ee}.

Two different solvers are used for the simulations in Simulink, namely ode45 and ode15s. The solver ode45 is used for simulations with the super-twisting algorithm with adaptive gains and the non-regressor-based adaptive controller. In the remaining simulations, with adaptive inverse dynamics control and both force controllers, the solver ode15s is used. The relative tolerance is set to 10^{-3} and all other parameters are set to auto.

3.4 Control framework

A control framework has been implemented to test the performance of the dynamic motion controllers and interaction controllers this thesis. An overview of the implemented control framework for the USM is shown in figure 3.4.

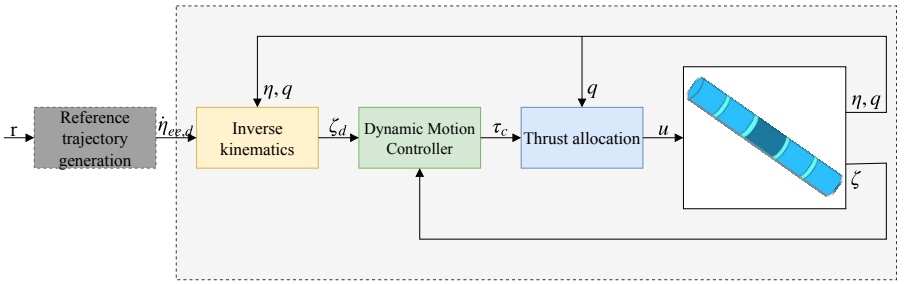


Figure 3.4: The implemented control framework for control of the USM.

The *reference trajectory generation* block is used to create a smooth reference trajectory $\eta_{ee,d}$ from the reference trajectory r. The reference values r may be for example predefined desired head-link positions or an operator specifying the desired motion of the head-link. Reference trajectory generation is elaborated on in section 3.4.1.

The *Inverse Kinematics* (IK) block calculates the desired desired base and joint velocities ζ_d from the time derivative of the end-effector pose $\dot{\eta}_{ee,d}$. To do this, the block requires the base position and orientation as well as the joint angles of the USM. Inverse kinematics is done by differential kinematics inversion and is further elaborated on in section 3.4.2.

The *Dynamic motion controller* (DC) computes the commanded generalized forces and torques, or commanded torques, τ_c which is the base and joint forces/torques required to achieve the desired velocities ζ_d (Sverdrup-Thygeson et al.; 2016a). Dynamic motion control is presented in chapter 4.

The *Thrust Allocation* (TA) block computes the control input to be applied to the thrusters, u_T, that will give the desired forces and moments on the base, $\tau_{c,T}$. The joint torques computed by the dynamic motion controller are directly applied to the

USM. The desired forces and moments on the base must however be achieved with the thrusters. Thrust allocation is further elaborated on in section 3.4.3.

3.4.1 Reference trajectory generation

The reference trajectory generation block provides smooth states to the control framework. This is done do avoid problems with derivatives in the simulation. The desired states $\eta_{ee,d}$ can be expressed by a MIMO mass-damper-spring system (Fossen; 2011):

$$M_d\ddot{\eta}_{ee,d} + D_d\dot{\eta}_{ee,d} + G_d\eta_{ee,d} = G_d r \tag{3.1}$$

where M_d, D_d and G_g are positive design matrices. The output of the reference trajectory generation block is the acceleration $\ddot{\eta}_{ee,d}$ which is integrated to give the input to the IK block $\dot{\eta}_{ee,d}$.

3.4.2 Differential kinematics inversion

The goal of the IK block is to compute the desired base and joint velocities ζ_d from desired end-effector position and orientation time-derivatives $\dot{\eta}_{ee}$. This is done using differential kinematics inversion. The output of the differential kinematics inversion is the input to the dynamic controller.

Differential kinematics inversion is done using the expression for the differential kinematics in equation (2.22) with the Jacobian $J(R^I_{ee}, q)$ defined in (2.25). The Jacobian $J_b(q)$ and the rotation matrix R^I_{ee} is available from the provided code, while the $J^{-1}_{k,o}(\eta_{2,ee})$ is implemented as in equation (2.10).

A solution to the differential kinematics inversion problem is to use the pseudoinverse of the Jacobian:

$$\zeta_d = J(R^I_{ee}, q)^\dagger \dot{\eta}_{ee,d}. \tag{3.2}$$

The pseudoinverse of a Jacobian, J, is denoted $J^\dagger$. The use of a pseudoinverse for differential kinematics inversion is simple and suitable for initial simulations and concept verification (Sverdrup-Thygeson et al.; 2016a). If the Jacobian J is full rank

the pseudoinverse can be calculated as (de Wit et al.; 2012):

$$J^{\dagger} = J^{\top}\left(JJ^{\top}\right)^{-1}. \tag{3.3}$$

This corresponds finding the minimal value of $E = \frac{1}{2}\zeta_d^{\top}\zeta_d$ that satisfy equation (3.2) (Sverdrup-Thygeson et al.; 2018).

In some cases it is relevant to prioritize minimization of some degrees of freedom more than others in the minimization. This can be done using the weighted pseudoinverse (Antonelli; 2018). The weighted pseudoinverse is calculated as:

$$J_W^{\dagger} = W^{-1}J^{\top}\left(JW^{-1}J^{\top}\right)^{-1}, \tag{3.4}$$

which minimizes the value of $E = \frac{1}{2}\zeta_d^{\top}W\zeta_d$ that satisfy equation (3.2) (Antonelli; 2018). $W \in R^{(6+n)\times(6+n)}$ is a diagonal matrix that can be designed to prioritize the use of certain control inputs (Sverdrup-Thygeson et al.; 2016a).

For the interaction control part of the thesis, a weighted pseudoinverse is implemented in the inverse kinematics block, while for the dynamic motion controller a regular pseudoinverse is used. With these approaches to the inverse kinematics problem, kinematic singularities are not addressed. The focus of the thesis is on dynamic control, and methods to avoid singularities in the kinematic control layer is therefore not included.

3.4.3 Thrust allocation

The objective of the thrusters is to control the position and orientation of the base, while the motorized joints control the joint angles. The output of the dynamic controller are the desired forces and torques τ_c. τ_c contains the desired forces and torques on the base $\tau_{c,T} \in R^6$ and the desired joint torques $\tau_{c,J} \in R^n$. While the desired joint torques $\tau_{c,J} = [0_{1\times 6}, u_J^{\top}]^{\top}$ is used directly as control input, the forces and torques on the base must be achieved with use of the thrusters. Thrust allocation consists of finding the control input $u_T \in R^m$ on the thrusters that will give the desired forces and torques $\tau_{c,T}$ on the base.

Divide the matrix B into one part that describes the effect of thrusters on the base position and orientation B_b and one part that describes the effect of thrusters on joints, B_J.

$$B = \begin{bmatrix} B_b \\ B_J \end{bmatrix} \tag{3.5}$$

with $B_b \in R^{6\times m}$ the first 6 rows of B and $B_J \in R^{n\times m}$ the last n rows of B. The resulting forces and torques on the base from the control input is described through the following relationship:

$$\tau_T = B_b u_T \tag{3.6}$$

Hence, equation (3.6) gives the forces and torques on the base from a certain control input on the thrusters u_T without including the torques on the joints from the thrusters. A simple method for calculating the control input to be applied to the thrusters is a pseudoinverse (Sverdrup-Thygeson et al.; 2016a):

$$u_T = B_b^{\dagger} \tau_{c,T}. \tag{3.7}$$

where $B_b^{\dagger} = B_b^{\top}\left(B_b B_b^{\top}\right)^{-1}$. This corresponds to minimizing the value of $E = \frac{1}{2} u_T^{\top} u_T$ that satisfy (3.7). Hence, it corresponds finding the combination of the smallest absolute values of thruster control inputs that give the desired forces and torques on the base.

3.4.4 Use of Euler angles and unit quaternions

While Euler angles is a more intuitive representation of orientation, unit quaternions avoids problems of representation singularities. Both Euler angles and unit quaternions are used to represent orientation in the simulations. The input to the control framework $\eta_{ee,d}$ use Euler angles as orientation representation because this makes the desired end-effector trajectory more intuitive. The presented dynamic motion controllers use quaternions to represent orientation to avoid problems with representation singularities.

3.4.5 Control considerations

USMs are similar to underwater vehicle manipulator systems (UVMSs). In fact, USMs can be considered a special case of UVMSs. UVMSs are AUVs or ROVs equipped with a manipulator in order to perform complex underwater tasks. Quite a lot of literature exists about dynamic control of UVMS which is relevant for control of USMs. There are however some important differences between the UVMSs and USMs that must be considered when applying UVMS controllers to USMs. The UVMSs has thrusters only on the base while the USM has thrusters on the links. As opposed to UVMS, the position and orientation of the thrusters relative to the base of the USM are configuration dependent (Sverdrup-Thygeson et al.; 2016a). Also, the UVMSs consists of a heavier base and a manipulator attached to it. Manipulator motion therefore has less effect on the base of the manipulator for UVMS than for USM where motion of joints has much more effect on the base position and orientation. In some cases, the manipulator and vehicle can be considered to be decoupled for UVMS (Ridao et al.; 2014), but this cannot be done for USMs. Avoidance of representation singularities are more important for a USM because for UVMSs the pitch angle of the vehicle is usually restricted (Sverdrup-Thygeson et al.; 2016a).

The EOMs of USMs and UVMSs are structurally similar to that of ground fixed manipulators, but for underwater applications there are usually larger uncertainties in the model knowledge. Hydrodynamic forces on rigid bodies moving in water are complex and highly nonlinear (Kelasidi, Pettersen, Gravdahl and Liljebäck; 2014) and the knowledge of hydrodynamic forces is therefore usually poor. In addition, the mathematical model may be more complex for UVMSs and USMs than for regular robot manipulators and the bandwidth of the sensors' readings may be low (Antonelli; 2014). In addition, USMs and UVMSs may be difficult to control due to poor thruster performance (Antonelli; 2014). In practice there are limits to how much force/torque the thrusters and joint motors can apply and how fast they respond to a commanded input. These challenges must be taken into account when designing controllers for USMs. Saturation and time delays are however disregarded in this thesis.

Chapter 4

Dynamic motion control

In this chapter, three dynamic motion control strategies are presented. First, some general background for dynamic motion control is presented. Then, adaptive inverse dynamics control (AIDC), the super-twisting algorithm with adaptive gains (STA) and non-regressor-based adaptive control (NRAC) are presented with stability analyses. Finally, simulation results with the controllers are presented and discussed.

4.1 Dynamic motion control background

The objective of the dynamic controller is to compute the forces and torques τ_c, required to make the USM follow a desired trajectory ζ_d. The books (Antonelli; 2014) and (Antonelli; 2018) give overviews of dynamic control for UVMSs. Litterature relevant to this chapter will be presented in the following.

The adaptive inverse dynamics controller (AIDC) presented in this chapter is based on (Antonelli and Chiaverini; 1998) and (Antonelli; 2014) where the AIDC is applied to a UVMS. This controller is again based on adaptive controllers for robot manipulators presented in (Ortega and Spong; 1988) and (Slotine and Li; 1987). In (Antonelli et al.; 2001), a similar controller is applied to control only an AUV without a manipulator. Furthermore, (Antonelli et al.; 2004) presents an adaptive tracking control algorithm

that reduces computational burden and simplifies application to UVMSs with a large number of links.

The super-twisting algorithm (STA) was first introduced in (LEVANT; 1993). In (Hamerlain; 2013) the STA is applied to the PUMA 560 robot manipulator and compared to the conventional sliding mode controller. In the article it is concluded that the super-twisting algorithm gives better performance than the conventional sliding mode controller. (Lee and Choi; 2000a) use a multilayer neural network as a compensator for a conventional sliding mode controller and apply the controller to a submerged manipulator with large model uncertainties.

A STA with adaptive gains is proposed in (Shtessel et al.; 2010). The super-twisting algorithm with adaptive gains drives the sliding variable to zero with a disturbance ϕ bounded by $\phi \leq \delta|s|^{1/2}$ where δ is bounded and unknown. In (Borlaug and Pettersen; 2017) this algorithm is used for trajectory tracking of a USMs center of mass in 2D with an observer for the states. (Shtessel et al.; 2011) and (Shtessel et al.; 2012) presents a super-twisting algorithm with adaptive gains very similar to (Shtessel et al.; 2010). Both articles show that the sliding variable is driven to zero with additive and multiplicative perturbations with unknown boundaries without overestimating the gains.

In (Sarkar et al.; 1999) the non-regressor-based adaptive controller developed in (Yuh; 1996) is extended to UVMS. The NRAC is similarly applied to a manipulator on a mobile platform in (Lee and Yuh; 1999). In (Yuh and Nie; 2000) the NRAC is applied to an AUV without manipulator and (Zhao and Yuh; 2005) the NRAC with a disturbance observer is presented.

Other work on dynamic control for underwater vehicles that can be mentioned are approaches based on neural networks such as (Lee and Choi; 2000b) and (Pandian and Sakagami; 2010).

All the controllers presented in this section are adaptive controllers. The idea of adaptive control is to modify some parameters online to *adapt* the controller to unknown parameters (Antonelli; 2014). The AIDC estimates uncertain model parameters, the gains of the STA increase until they become large enough to overcome disturbances and NRAC estimates parameters define by unknown bounds on system matrices.

4.1.1 Implementation background

The dynamic motion controllers presented in this chapter use, in addition to ζ_d, the desired base position, $\eta_{1,d}$, the desired base orientation represented with quaternions, p_d, and the desired joint angles q_d to calculate τ_c. These values are calculated from ζ_d. The base position is calculated using relation (2.3), the desired base quaternions are calculated using the quaternion propagation equation (2.17) and integration of (2.19) and the desired joint angles are calculated by integration of the last n entries of ζ_d.

The control framework for the controllers presented in this section is shown in figure 3.4. A desired reference trajectory r is smoothened by the reference trajectory generation block to give $\dot{\eta}_{ee}$. The inverse kinematics block give the desired values ζ_d that is used in the dynamic motion controllers presented in this section. The time derivative of ζ_d is taken to find $\dot{\zeta}_d$ which is also used by some of the dynamic motion controllers. The output of the dynamic motion controllers is the commanded torque τ_c.

4.1.2 Model transformation

Recall the EOMs of the USM:

$$M(q)\dot{\zeta} + C(q,\zeta)\zeta + D(q,\zeta)\zeta + g(q,\eta) = \tau. \tag{4.1}$$

The equation (4.1) will be rewritten for convenience in the presentation of the controllers. The generalized forces τ in equation (4.1) can be written as

$$\tau = Bu_T + \begin{bmatrix} 0_{6\times 1} \\ u_J \end{bmatrix} \tag{4.2}$$

where u_T is the thruster control input and u_J is the joint torque control input. This can further be written as

$$\tau = \left[\begin{array}{c|c} B & \begin{matrix} 0_{6\times n} \\ I_n \end{matrix} \end{array}\right] \begin{bmatrix} u_T \\ u_J \end{bmatrix} = \begin{bmatrix} B_b & 0_{6\times n} \\ B_J & I_n \end{bmatrix} \begin{bmatrix} B_b^{\dagger}\tau_{c,T} \\ \tau_{c,J} \end{bmatrix} = \begin{bmatrix} I_6 & 0_{6\times n} \\ B_J B_b^{\dagger} & I_n \end{bmatrix} \begin{bmatrix} \tau_{c,T} \\ \tau_{c,J} \end{bmatrix} = B_{tot}\tau_c. \tag{4.3}$$

with $B_{tot} \in R^{(6+n)\times(6+n)}$ and B_b and B_J was defined in section 3.4.3.

For the stability analyses in this section it is useful to express $\dot{\zeta}$ as:

$$\dot{\zeta} = M^{-1}(q)\big(B_{tot}\tau_c - C(q,\zeta)\zeta - D(q,\zeta)\zeta - g(q,\eta)\big). \tag{4.4}$$

4.1.3 Orientation error representation

The controllers in this section use feedback of the orientation represented by quaternions to avoid representation singularities. The orientation error representation using quaternions is found as follows (Antonelli; 2018). Call the rotation matrix necessary to align the desired base orientation and the bare frame $\tilde{R}$. The quaternion error $\tilde{p}$ equivalent to $\tilde{R}$ can be computed by $\tilde{p} = p_d * p^{-1}$ which can be written as (Antonelli; 2018):

$$\begin{aligned} \tilde{\eta}_Q &= \eta_{Q,d}\eta_Q + \epsilon_d^{\top}\epsilon, \\ \tilde{\epsilon} &= \eta_Q\epsilon_d - \eta_{Q,d}\epsilon + \mathrm{S}(\epsilon)\epsilon_d. \end{aligned} \tag{4.5}$$

The quaternion propagation equation can be written in terms of the quaternion errors in matrix form similarly to (2.15):

$$\begin{bmatrix} \dot{\tilde{\eta}}_Q \\ \dot{\tilde{\epsilon}} \end{bmatrix} = \begin{bmatrix} -\tilde{\epsilon}^{\top} \\ \tilde{\eta}_Q I_3 + S(\tilde{\epsilon}) \end{bmatrix} \tilde{\nu}_2 = J_{k,oq}(z)\tilde{\nu}_2. \tag{4.6}$$

4.2 Adaptive inverse dynamics control

Inverse dynamics control is a model-based control strategy. The idea is to cancel nonlinear terms with a nonlinear state feedback and decouple the dynamics of the

USM (de Wit et al.; 2012). Nonlinearities are compensated for by adding the nonlinear terms to the control input (de Wit et al.; 2012). The following inverse dynamics controller gives a stable system:

$$\tau_c = B_{tot}^{\dagger}[K_D s' + M(q)\dot{\zeta}_r + C(q,\zeta)\zeta_r + D(q,\zeta)\zeta_r + g(q,\eta)]. \tag{4.7}$$

with K_D, s' and ζ_r defined in section 4.2.1. A stability analysis of this controller (4.7) is included in the appendix A.3. This inverse dynamics controller relies on exact cancellation of nonlinear terms. It therefore requires that the model parameters are exactly known. However, a challenge with control of USMs is uncertainty in the model knowledge and the inverse dynamics controller (4.7) is therefore not suitable for control of USMs. Instead, the controller (4.7) is modified to give an *adaptive* inverse dynamics controller. For adaptive inverse dynamics control the computational model used for inverse dynamics control is adapted online to the dynamic model of the USM (Siciliano et al.; 2008). The true control parameters are replaced with estimates of the parameters and the estimates are updated by an update law.

Adaptive inverse dynamics control requires linearity in parameters. The dynamics of the USM (4.1) can be written as (Antonelli; 2014):

$$M(q)\dot{\zeta} + C(q,\zeta)\zeta + D(q,\zeta)\zeta + g(q,\eta) = \phi(q, R_b^I, \zeta, \dot{\zeta})\theta. \tag{4.8}$$

This property holds for rigid bodies moving in space. For underwater rigid bodies it depends on the representation of the hydrodynamic terms (Antonelli; 2014). The vector $\theta \in R^{n_\theta}$ includes n_θ parameters of the dynamic system and the matrix $\phi(q, R_b^I, \zeta, \dot{\zeta}) \in R^{(6+n)\times n_\theta}$ is called the regressor matrix. For a single rigid body the number of dynamic parameters n_θ can be greater than 100 (Antonelli; 2014), but can be reduced when there are known symmetry properties. For UVMSs the vehicle and manipulator dynamics can in some cases be considered decoupled to simplify the regressor matrix (Antonelli; 2018), but this is not a valid assumption for USMs.

4.2.1 Control law and adaption

The adaptive inverse control law for UVMS is presented in (Antonelli and Chiaverini; 1998) is given as follows:

$$\tau_c = B_{tot}^{\dagger}[K_D s' + \phi(q, R_b^I, \zeta, \zeta_r, \dot{\zeta}_r)\hat{\theta}], \tag{4.9}$$

with $\dot{\hat{\theta}} = K_\theta^{-1}\phi^\top(q, R_b^I, \zeta, \zeta_r, \dot{\zeta}_r)s$. The gain matrices K_D and K_θ as well as ζ_r, s and s' are defined below.

The system in (4.8) is complex and the controller (4.9) will be simplified in the following. Hydrodynamic forces on the USM are complex and nonlinear and therefore difficult to model (Kelasidi, Pettersen, Gravdahl and Liljebäck; 2014). The mass and gravity forces are however easier to model correctly. Hydrostatics are also easier to model because the volume of the USM is known. In addition, added mass effects are less significant for slow, steady motion. Based of this, it will in the following be assumed that the parameters of $M(q)$, $C(q, \zeta)$ and $g(q, \eta)$ are known and that only parameters of $D(q, \zeta)$ are unknown. The control law (4.9) will therefore be modified to the following:

$$\tau_c = B_{tot}^{\dagger}[K_D s' + M(q)\dot{\zeta}_r + C(q, \zeta)\zeta_r + g(q, \eta) + \phi(q, \zeta, \zeta_r)\hat{\theta}] \tag{4.10}$$

where the update law for the estimates $\hat{\theta}$ is obtained from

$$\dot{\hat{\theta}} = K_\theta^{-1}\phi^\top(q, \zeta, \zeta_r)s. \tag{4.11}$$

with $K_\theta > 0$ and $K_D > 0$. With $\hat{\theta} = \theta$ the expression $\phi(q, \zeta, \zeta_r)\hat{\theta}$ equals $D(q, \zeta)\zeta_r$ such that the controller (4.10) equals the original controller (4.7). The error variable s is defined as

$$s = \begin{bmatrix} s_p \\ s_o \\ s_q \end{bmatrix} = \begin{bmatrix} \tilde{\nu}_1 \\ \tilde{\nu}_2 \\ \dot{\tilde{q}} \end{bmatrix} + \Lambda \begin{bmatrix} R_I^b \tilde{\eta}_1 \\ \tilde{\epsilon} \\ \tilde{q} \end{bmatrix} = \zeta_d - \zeta + \Lambda\tilde{y} = \tilde{\zeta} + \Lambda\tilde{y}, \tag{4.12}$$

where $\tilde{\nu}_1 = \nu_{1,d} - \nu_1$, $\tilde{\nu}_1 = \nu_{2,d} - \nu_2$, $\tilde{\eta}_1 = \eta_d - \eta$, $\tilde{q} = q_d - q$ and $\tilde{\epsilon}$ is the quaternion

error defined in (4.5). The variable s' and ζ_r are defined as:

$$s' = s + K_D^{-1} K_P \tilde{y} = \tilde{\zeta} + (\Lambda + K_D^{-1} K_P)\tilde{y}, \tag{4.13}$$

and

$$\zeta_r = \zeta_d + \Lambda \tilde{y}. \tag{4.14}$$

The remaining parameters of the controller are defined as:

$$\Lambda = \begin{bmatrix} \lambda_p I_3 & 0_{3\times 3} & 0_{3\times n} \\ 0_{3\times 3} & \lambda_o I_3 & 0_{3\times n} \\ 0_{n\times 3} & 0_{n\times 3} & \Lambda_q \end{bmatrix}, \quad \Lambda_q \in R^{n\times n}, \quad \Lambda > 0 \tag{4.15}$$

and

$$K_P = \begin{bmatrix} k_p I_3 & 0_{3\times 3} & 0_{3\times n} \\ 0_{3\times 3} & k_o I_3 & 0_{3\times n} \\ 0_{n\times 3} & 0_{n\times 3} & K_q \end{bmatrix}, \quad K_q \in R^{n\times n}, \quad K_P > 0. \tag{4.16}$$

4.2.2 The regressor matrix

The goal is to create an adaptive inverse dynamics controller (4.10) that estimates the inverse dynamics controller in (4.7) when the drag coefficients are C_{d1}, C_{d4}, C_{dC} and C_{dL} are unknown. The unknown parameters are collected in a vector θ:

$$\theta = \begin{bmatrix} C_{d1} \\ C_{d4} \\ C_{dC} \\ C_{dL} \end{bmatrix} \tag{4.17}$$

The hydrodynamic forces in the original controller (4.7), $D(q,\zeta)\zeta_r$, can then be written as $\phi(q,\zeta,\zeta_r)\theta$. The hydrodynamic damping matrix expression $D(q,\zeta)$ was stated in (2.40). Taking into account $\nu_{b_i,r} = J_{b,1}(q)\zeta_r$ (2.24), the expression for $D(q,\zeta)\zeta_r$ may be

written as:

$$\begin{aligned} D(q,\zeta)\zeta_r &= \sum_{i=0}^{n} J_{b,i}(q)^\top D_i(q,\zeta) J_{b,i}(q)\zeta_r \\ &= \sum_{i=0}^{n} J_{b,i}(q)^\top D_i(q,\zeta) \begin{bmatrix} \nu_{1,b_i,r} \\ \nu_{2,b_i,r} \end{bmatrix} = \sum_{i=0}^{n} J_{b,i}(q)^\top d_{i,r}, \end{aligned} \tag{4.18}$$

where

$$d_{i,r} = D_i(q,\zeta) \begin{bmatrix} \nu_{1,b_i,r} \\ \nu_{2,b_i,r} \end{bmatrix} \in R^6 \tag{4.19}$$

is the vector of drag forces associated with link i (2.38).

The vector of drag forces on link i, $d_{i,r}$, is calculated based on the hydrodynamic forces presented in section 2.4.3. From the equations of the drag vector, the drag coefficients may be factorized out such that

$$d_{i,r} = D_i(q,\zeta,\zeta_r)\theta, \tag{4.20}$$

which leads to an expression for the regressor matrix $\phi(q,\zeta,\zeta_r)$:

$$D(q,\zeta)\zeta_r = \sum_{i=0}^{n} J_{b,i}(q)^\top D_i(q,\zeta,\zeta_r)\theta = \left(\sum_{i=0}^{n} J_{b,i}(q)^\top D_i(q,\zeta,\zeta_r) \right)\theta = \phi(q,\zeta,\zeta_r)\theta. \tag{4.21}$$

4.2.3 Stability analysis

In this section, a stability analysis of the system with τ_c defined in (4.10) will be performed.

Consider the Lyapunov function candidate suggested in (Antonelli and Chiaverini; 1998):

$$V = \frac{1}{2} s^\top M(q) s + \frac{1}{2}\tilde{\theta}^\top K_\theta \tilde{\theta} + \frac{1}{2} k_p \tilde{\eta}_1^\top \tilde{\eta}_1 + k_o \tilde{z}^\top \tilde{z} + \frac{1}{2}\tilde{q}^\top K_q \tilde{q} \tag{4.22}$$

where $\tilde{z} = [1 - \tilde{\eta} \quad -\tilde{\epsilon}^\top]^\top$ and $\tilde{\theta} = \theta - \hat{\theta}$. V has the property $V \geq 0$ because $k_p > 0$,

$k_o > 0$ and K_θ, K_q and $M(q)$ are positive definite. The positive definiteness of $M(q)$ is provided by property 2.1. The variable s is defined in (4.12). The time derivative of (4.22) is

$$\dot{V} = \frac{1}{2}s^\top \dot{M}s + s^\top M\dot{s} + \tilde{\theta}^\top K_\theta \dot{\tilde{\theta}} + k_p \tilde{\eta}_1^\top \dot{\tilde{\eta}}_1 + 2k_o \tilde{z}^\top \dot{\tilde{z}} + \tilde{q}^\top K_q \dot{\tilde{q}}. \tag{4.23}$$

Rewrite the expression using the relation $\dot{\tilde{\eta}}_1 = R^I_{b_0}\tilde{\nu}_1$ (2.3) and that $\dot{\tilde{z}} = -\dot{z} = -J_{k,oq}(z)\tilde{\nu}_2$ (4.6).

$$\dot{V} = \frac{1}{2}s^\top \dot{M}s + s^\top M\dot{s} + \tilde{\theta}^\top K_\theta \dot{\tilde{\theta}} + k_p \tilde{\eta}_1^\top R^I_b \tilde{\nu}_1 - 2k_o \tilde{z}^\top J_{k,oq}(z)\tilde{\nu}_2 + \tilde{q}^\top K_q \dot{\tilde{q}} \tag{4.24}$$

Now, consider the expression for $\dot{s}$

$$\dot{s} = \dot{\tilde{\zeta}} + \Lambda\dot{\tilde{y}} = \dot{\zeta}_d - \dot{\zeta} + \Lambda\dot{\tilde{y}}. \tag{4.25}$$

Rewrite the expression for $\dot{s}$ using the expression for $\dot{\zeta}$ found from the dynamics equation (4.4) and the fact that $\dot{\zeta}_d = \dot{\zeta}_r - \Lambda\dot{\tilde{y}}$.

$$\dot{s} = \dot{\zeta}_r - M^{-1}(q)\big(B_{tot}\tau_c - C(q,\zeta)\zeta - D(q,\zeta)\zeta - g(q,\eta)\big). \tag{4.26}$$

Write $\dot{s}$ as a function of ζ_r. To do this, notice that $\zeta_d = \zeta_r - \Lambda\tilde{y}$ (4.14) and $s = \zeta_d - \zeta + \Lambda\tilde{y}$. Combine these to get $\zeta = \zeta_r - s$ which can be substituted into $\dot{s}$ (4.26). Substitute the resulting expression into the derivative $\dot{V}$ (4.24). Remark: In the following the arguments of the system matrices will be omitted for better readability.

$$\dot{V} = s^\top[M\dot{\zeta}_r + C\zeta_r + D\zeta_r + g - B_{tot}\tau_c] - s^\top Ds - \tilde{\theta}^\top K_\theta \dot{\hat{\theta}} + \underbrace{k_p \tilde{\eta}_1^\top R^I_b \tilde{\nu}_1 - 2k_o \tilde{z}^\top J_{k,oq}(z)\tilde{\nu}_2 + \tilde{q}^\top K_q \dot{\tilde{q}}}_{\dot{V}_1} \tag{4.27}$$

The term $\frac{1}{2}s^\top \dot{M}s - s^\top Cs$ disappears as a result of property 2.2. It is assumed that the dynamic parameters in θ are constant such that $\dot{\tilde{\theta}} = \dot{\theta} - \dot{\hat{\theta}} = -\dot{\hat{\theta}}$.

Now, the last three terms of (4.27), $\dot{V}_1$ will be considered. The variable s can be divided into a position, orientation and angle component respectively, $s = [s_p^\top, s_o^\top, s_q^\top]^\top$

(4.12). Observe from the definition of s (4.12) and the definition of Λ (4.15) that $\tilde{\nu}_1$, $\tilde{\nu}_2$ and $\dot{\tilde{q}}$ can be written as

$$\begin{aligned} \tilde{\nu}_1 &= s_p - \lambda_p R_I^b \tilde{\eta}_1, \\ \tilde{\nu}_2 &= s_o - \lambda_o \tilde{\epsilon}, \\ \dot{\tilde{q}} &= s_q - \Lambda_q \tilde{q}. \end{aligned} \tag{4.28}$$

Using this disassembly of s, the last three terms of (4.27), $\dot{V}_1$ can then be written as:

$$\dot{V}_1 = k_p \tilde{\eta}_1^\top R_b^I s_p - k_p \lambda_p \tilde{\eta}_1^\top \tilde{\eta} + k_o \tilde{\epsilon}^\top s_o - k_o \lambda_o \tilde{\epsilon}^\top \tilde{\epsilon} + \tilde{q}^\top K_q s_q - \tilde{q}^\top K_q \Lambda_q \tilde{q} \tag{4.29}$$

because $\tilde{z}^\top J_{k,oq}(z) = -\epsilon^\top$ (from equation (4.6)). The terms that include s_p, s_o and s_q can be collected in a term $s^\top K_P \tilde{y}$ with K_P defined in (4.16). The expression for $\dot{V}$ (4.27) can therefore be written as:

$$\dot{V} = s^\top [M\dot{\zeta}_r + C\zeta_r + D\zeta_r + g - B_{tot}\tau_c + K_P\tilde{y}] - s^\top Ds - \tilde{\theta}^\top K_\theta \dot{\hat{\theta}} - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta}_1 - k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} - \tilde{q}^\top K_q\Lambda_q\tilde{q} \tag{4.30}$$

Plug in the control input τ_c (4.10) and the estimate update law $\dot{\hat{\theta}}$ (4.11) and replace $D(q,\zeta)\zeta_r = \phi(q,\zeta,\zeta_r)\theta$ (4.21) and $s' = s + K_D^{-1}K_P\tilde{y}$ (4.13):

$$\begin{aligned} \dot{V} =& s^\top [M\dot{\zeta}_r + C\zeta_r + \phi(q,\zeta,\zeta_r)\theta + g - K_D(s + K_D^{-1}K_P\tilde{y}) - M\dot{\zeta}_r - C\zeta_r - g - \phi(q,\zeta,\zeta_r)\hat{\theta} \\ &+ K_P\tilde{y}] - s^\top Ds - \tilde{\theta}^\top\phi^\top(q,\zeta,\zeta_r)s - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta}_1 - k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} - \tilde{q}^\top K_q\Lambda_q\tilde{q} \\ =& s^\top[\phi(q,\zeta,\zeta_r)\theta - K_D s - \phi(q,\zeta,\zeta_r)\hat{\theta} - \phi(q,\zeta,\zeta_r)\tilde{\theta}] - s^\top Ds - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta}_1 \\ &- k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} - \tilde{q}^\top K_q\Lambda_q\tilde{q}. \end{aligned} \tag{4.31}$$

Using the fact that $\tilde{\theta} = \theta - \hat{\theta}$, this results in:

$$\dot{V} = -s^\top(K_D + D)s - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta}_1 - k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} - \tilde{q}^\top K_q\Lambda_q\tilde{q} \leq 0. \tag{4.32}$$

which is negative semi-definite over the state space $\{\tilde{y}, s, \tilde{\theta}\}$ (Antonelli and Chiaverini; 1998). Property 2.4 provides D>0. Notice that $\tilde{\theta}$ is not included in the expression (4.32). According to Barbalat's Lemma (Lemma A.1), since

- V is lower bounded,
- $\dot{V} \leq 0$ and
- $\dot{V}$ is uniformly continuous then

$\lim_{t\to\infty} \dot{V}(\tilde{y}, s, \tilde{\theta}) = 0$. Therefore, $\lim_{t\to\infty} \tilde{y} = 0$ and $\lim_{t\to\infty} s = 0$. The state $\tilde{\theta}$ is only guaranteed to be bounded (Antonelli and Chiaverini; 1998). The controller (4.10) therefore gives a stable system (definition A.1).

4.3 Super-twisting algorithm with adaptive gains

The super-twisting algorithm is a second-order sliding mode control algorithm. In this section, it is shown how the STA with adaptive gains can be applied to the USM presented in chapter 3. Sliding mode control is a nonlinear control strategy in which the states reach a sliding surface and slide towards the origin along the surface. Sliding mode control can give stability even with disturbances. The design of a sliding mode controller consists of designing a sliding-surface and a control input such that the sliding surface is reached in finite time.

Sliding mode control is a robust and versatile control algorithm that give asymptotic stability of the tracking error even with little knowledge of the model parameters and unmodeled dynamics (de Wit et al.; 2012). Sliding mode control produces a discontinuous controller (Khalil; 2002). The main drawback of sliding mode control is that the control input will have high frequency components. The high frequency switching in control signal is called chattering. Chattering arises from imperfections in switching devices and delays (Khalil; 2002). Ideally the state should begin sliding along the trajectory as soon as it reaches the surface, but if the switching is delayed the state goes a bit past the sliding surface before the switch. This causes a zig-zag motion (oscillation) in the control input. Chattering in the control input may cause wear of the mechanical parts and reduce the lifetime (de Wit et al.; 2012). The chattering problem may be reduced by using saturation control or a higher order sliding mode controller such as the super-twisting algorithm (Shtessel et al.; 2010).

The super-twisting algorithm is one of the most powerful second-order sliding mode algorithms (Borlaug and Pettersen; 2017). It does not completely remove chattering, but attenuates it. For the super-twisting algorithm one needs to know boundaries on the disturbance gradient (Shtessel et al.; 2011) and conservative upper bounds must be used. This results in gains that are larger than necessary. Using adaptive gains in the super twisting algorithm will make the gains as small as possible while still large enough to maintain sliding (Borlaug and Pettersen; 2017).

4.3.1 Sliding surface

The choice of sliding surface is based on (Antonelli; 2018) and the following sliding surface is used for the controller.

$$s = \begin{bmatrix} R_I^b \tilde{\eta}_1 \\ \tilde{\epsilon} \\ \tilde{q} \end{bmatrix} + \lambda \begin{bmatrix} \tilde{\nu}_1 \\ \tilde{\nu}_2 \\ \dot{\tilde{q}} \end{bmatrix} = \tilde{y} + \lambda \tilde{\zeta} \tag{4.33}$$

where $\lambda > 0$. When the sliding surface $s = 0$ the equation leads to

$$\tilde{y} = -\lambda \zeta \tag{4.34}$$

which describes the motion of the states on the sliding surface. This can be written as

$$\begin{bmatrix} \tilde{\eta}_1 \\ \tilde{\epsilon} \\ \tilde{q} \end{bmatrix} = -\lambda \begin{bmatrix} \dot{\tilde{\eta}} \\ \frac{2}{\tilde{\eta}_q} \dot{\tilde{\epsilon}} \\ \dot{\tilde{q}} \end{bmatrix} \qquad \begin{bmatrix} \dot{\tilde{\eta}} \\ \dot{\tilde{\epsilon}} \\ \dot{\tilde{q}} \end{bmatrix} = -\lambda^{-1} \begin{bmatrix} \tilde{\eta}_1 \\ \frac{\tilde{\eta}_q}{2} \tilde{\epsilon} \\ \tilde{q} \end{bmatrix} \tag{4.35}$$

using the relations (2.3) and (2.17). It can easily be shown that the system *slides* towards $\tilde{y} = 0$ when the sliding surface is reached. Consider the positive definite function:

$$V = \frac{1}{2} \lambda \tilde{y}^\top \tilde{y} \tag{4.36}$$

with time derivative

$$\dot{V} = \lambda \tilde{y}^\top \dot{\tilde{y}}. \tag{4.37}$$

Substituting for $\tilde{y}$ from (4.35)

$$\dot{V} = -\tilde{\eta}^\top \tilde{\eta} - \tilde{q}^\top \tilde{q} - \frac{\tilde{\eta}_q}{2} \tilde{\epsilon}^\top \tilde{\epsilon} \leq 0 \tag{4.38}$$

$\dot{V} < 0$ for all $\tilde{y} \neq 0$ as long as $\tilde{\eta}_q \geq 0$. The quaternion error is defined in $[-\pi, \pi]$ and $\tilde{\eta}_q$ is therefore greater or equal to zero. Using theorem A.1 the system will therefore, when at the sliding surface, the *slide* asymptotically towards $\tilde{y} = 0$ (Antonelli; 2014). The rate of convergence is decided by λ.

4.3.2 Super-twisting algorithm

The super-twisting algorithm with adaptive gains proposed in (Shtessel et al.; 2010) can be written in the following form (Borlaug and Pettersen; 2017):

$$\begin{cases} u_{ST,i} = -\alpha_i |s_i|^{1/2} sgn(s_i) + v_i \\ \dot{v}_i = -\frac{\beta_i}{2} sgn(s_i) \end{cases} \tag{4.39}$$

for $i = 1, 2...(6 + n)$ where $u_{ST,i}$, s_i, v_i, α_i and β_i and are the i'th elements of the vectors u_{ST}, s, v, α and β respectively. This is a second order sliding mode control algorithm. The sign function switches the the sign of the first term of u_{ST} from positive to negative instantly at $s = 0$ as s goes from $s < 0$ to $s > 0$. Chattering is not eliminated but attenuated, as the algorithm contains a discontinuous function under the integral (Shtessel et al.; 2010). The adaptive gains α_i and β_i are given below. How these are derived is shown in the stability analysis in the section 4.3.4.

$$\dot{\alpha}_i = \begin{cases} \omega_{1,i} \sqrt{\frac{\gamma_{1,i}}{2}}, & \text{if } s_i \neq 0 \\ 0, & \text{if } s_i = 0 \end{cases} \qquad \text{and} \qquad \beta_i = 2\epsilon\alpha_i + \lambda + 4\epsilon^2 \tag{4.40}$$

for $i = 1, 2...(6 + n)$ where ω_i, γ_i and ϵ are positive constants. Note that the ϵ defined here is a design parameter of the STA, not the imaginary parts of the quaternion. With

these gains the sliding surface $s = 0$ will be reached, for any initial conditions, in finite time with super-twisting sliding mode control (Shtessel et al.; 2010). This is shown in section 4.3.4.

For implementation purposes the update law for the adaptive gain α is implemented with a small boundary on the sliding surface (Borlaug and Pettersen; 2017). The update law for the adaptive gain α_i is for implementation purposes expressed as:

$$\dot{\alpha}_i = \begin{cases} \omega_{1,i}\sqrt{\frac{\gamma_{1,i}}{2}}, & \text{if } s_i > \alpha_m \\ 0, & \text{if } |s_i| \leq \alpha_m \end{cases} \tag{4.41}$$

where α_m is a small positive constant (Borlaug and Pettersen; 2017).

4.3.3 Control design

The objective the super-twisting algorithm is to drive s and $\dot{s}$ to zero in finite time. To achieve this, one wants to choose the commanded torque τ_c such that $\dot{s} = u_{ST}$ (Borlaug and Pettersen; 2017). To find the commanded torque τ_c to achieve this, take the derivative of the sliding surface s (4.33) and substitute $\dot{\zeta}$ from the system dynamics ((4.4)).

$$\begin{aligned} \dot{s} &= \ddot{\tilde{y}} + \lambda(\dot{\zeta}_d - \dot{\zeta}) \\ &= \ddot{\tilde{y}} + \lambda(\dot{\zeta}_d - M^{-1}(q)(B_{tot}\tau_c - C(q,\zeta)\zeta - D(q,\zeta)\zeta - g(q,\eta)) = u_{ST} \end{aligned} \tag{4.42}$$

The commanded torque τ_c should cancel all the terms such that $\dot{s} = u_{ST}$. Rewrite the second line of equation (4.42) to find the required τ_c:

$$\tau_c = B_{tot}^{\dagger}\left[M(q)\lambda^{-1}(\lambda\zeta_d + \ddot{\tilde{y}} - u_{ST}) + C(q,\zeta)\zeta + D(q,\zeta)\zeta + g(q,\eta)\right] \tag{4.43}$$

If the model parameters are exactly known such that it is possible to use this expression (4.43), $\dot{s} = u_{ST}$ is achieved. However, it is not a realistic to have exact knowledge of the parameters of a USM. Assume that the term $D(q,\zeta)\zeta$ is unknown. Then it is not

possible to use the control input in (4.43). Instead, consider the control input

$$\tau_c = B_{tot}^{\dagger}\left[M(q)\lambda^{-1}(\lambda\dot{\zeta}_d + \ddot{\tilde{y}} - u_{ST}) + C(q,\zeta)\zeta + g(q,\eta)\right] \tag{4.44}$$

The resulting derivative of the sliding variable is then:

$$\dot{s} = u_{ST} + \lambda M^{-1}(q)D(q,\zeta)\zeta \tag{4.45}$$

The stability proof in section 4.3.4 shows that the super-twisting algorithm drives the sliding variable to zero, even with disturbances or unknown dynamics.

4.3.3.1 Disturbances

Assume that there are some disturbances/uncertainties that enter the state equation at the same point as the control input such that such that $\dot{s}$ is not exactly equal to u_{ST}. Call the disturbance on the ith input $\phi_i(q,\zeta)$ and assume that the initial value is 0. Assume that $|\dot{\phi}(q,\zeta)|$ is bounded by some constant L. It will be assumed that this holds for the drag forces because intuitively drag forces cannot change infinitely fast. The derivative of the sliding surface then becomes $\dot{s} = u_{ST} + \phi(q,\zeta)$. It will be shown in section 4.3.4 that the sliding surface $s = 0$ is reached in finite time even with this disturbance.

Shtessel et al. (2010) shows that the sliding surface is reached in finite time with disturbance/uncertainties $\Phi(x,t)$ when the uncertainties are bounded by $|\Phi(x,t)| \leq \delta|s|^{1/2}$ where δ is an unknown constant. This means that the disturbance vanishes when the sliding surface is reached and will not be included in the following stability proof.

4.3.4 Stability analysis

The following stability analysis is based on (Shtessel et al.; 2011) and (Shtessel et al.; 2010). It will be shown that the sliding surface $s = 0$ is reached in finite time with the choice of adaptive gains (4.40) and the choice of the commanded torque expression (4.45).

Write the system (4.39) in the following form:

$$\begin{cases} \dot{s_i} = u_{ST,i} = -\alpha_i |s_i|^{1/2} sgn(s_i) + v_i \\ \dot{v_i} = -\frac{\beta_i}{2} sgn(s_i) + \dot{\phi}_i(q, \zeta) \end{cases} \tag{4.46}$$

with the gains defined in (4.40). In the stability analysis, begin by considering only state i. The stability proof holds for all states $i = 1...(6 + n)$. Define $z = [z_1, z_2]^\top = [|s_i|^{1/2} sgn(s_i), v_i]^\top$. With this z, equation (4.46) can be rewritten as an equation in z through the following steps. First, find the derivative of z_1 and insert $\dot{s_i}$ from equation (4.46)(Shtessel et al.; 2010).

$$\begin{aligned} \dot{z}_1 =& \frac{s_i sgn(s_i) \dot{s}_i}{2|s_i|^{3/2}} \\ \dot{z}_1 =& \frac{s_i sgn(s_i) \big(- \alpha_i |s_i|^{1/2} sgn(s_i) + v_i)}{2|s_i|^{3/2}} \\ \dot{z}_1 =& (\frac{-\alpha_i sgn(s_i)}{2} + \frac{v_i}{2|s_i|^{1/2}}) \end{aligned} \tag{4.47}$$

Observe that $|z_1| = |s|^{1/2}$ and express $\dot{z}$ in terms of z_1 and z_2.

$$\begin{cases} \dot{z}_1 = \frac{1}{|z_1|} \left(\frac{-\alpha_i}{2} z_1 + \frac{1}{2} z_2) \right) \\ \dot{z}_2 = -\frac{\beta_i}{2|z_1|} z_1 + \dot{\phi}_i(q, \zeta) \end{cases} \tag{4.48}$$

Equation (4.48) can be written in matrix form as

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \frac{1}{2|z_1|} \begin{bmatrix} -\alpha_i & 1 \\ -\beta_i & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \frac{1}{2|z_1|} \begin{bmatrix} 0 \\ 2|z_1| \end{bmatrix} \dot{\phi}_i(q, \zeta) = A(z_1) z + G(z_1) \dot{\phi}_i(q, \zeta) \tag{4.49}$$

where α_i and β_i are the adaptive gains. If z_1, z_2 reach zero in finite time, then s_i and v_i reach zero in finite time and hence $\dot{s}_i$ also reaches zero in finite time.

Write $\dot{\phi}_i(q,\zeta)$ as (Shtessel et al.; 2012)

$$\dot{\phi}_i(q,\zeta) = \frac{1}{2}\rho(q,\zeta)sgn(s_i) = \frac{1}{2}\rho(q,\zeta)\frac{z_1}{|z_1|} \tag{4.50}$$

which implies that $\rho(q,\zeta) = 2\dot{\phi}_i(q,\zeta,t)sgn(s_i)$ and $|\rho(q,\zeta)| \leq 2L$ (Shtessel et al.; 2012). Equation 4.49 can then be written as

$$\begin{bmatrix}\dot{z}_1\\ \dot{z}_2\end{bmatrix} = \frac{1}{2|z_1|}\begin{bmatrix}-\alpha_i & 1\\ -(\beta_i - \rho(q,\zeta)) & 0\end{bmatrix}\begin{bmatrix}z_1\\ z_2\end{bmatrix} = \overline{A}(z_1,q,\zeta)z \tag{4.51}$$

Now, a stability analysis of the system (4.49) is performed to show that z_1, z_2 reach zero in finite time and hence that $s, \dot{s}$ reach zero in finite time. The following Lyapunov function candidate is proposed (Shtessel et al.; 2011).

$$\begin{aligned} V(z,\alpha_i,\beta_i) =& V_0(z) + \frac{1}{2\gamma_1}(\alpha_i - \alpha_i^*)^2 + \frac{1}{2\gamma_2}(\beta_i - \beta_i^*)^2\\ V_0(z) =& (\lambda + 4\epsilon^2)z_1^2 + z_2^2 - 4\epsilon z_1 z_2 = z^\top P z \end{aligned} \tag{4.52}$$

α_i^* and β_i^* are constants and P is defined as (Shtessel et al.; 2011):

$$P = \begin{bmatrix}\lambda + 4\epsilon^2 & -2\epsilon\\ -2\epsilon & 1\end{bmatrix}, \quad \lambda > 0, \quad \epsilon > 0, \quad \alpha_i^* > 0, \quad \beta_i^* > 0 \tag{4.53}$$

P is positive definite for $\lambda > 0$ and any ϵ. The time derivative of the Lyapunov function (4.52) is:

$$\dot{V}(z,\alpha_i,\beta_i) = \dot{z}^\top P z + z^\top P \dot{z} + \frac{1}{\gamma_1}(\alpha_i - \alpha_i^*)\dot{\alpha}_i + \frac{1}{\gamma_2}(\beta_i - \beta_i^*)\dot{\beta}_i \tag{4.54}$$

Consider only the derivative of V_0.

$$\begin{aligned} \dot{V}_0 =& \dot{z}^\top P z + z^\top P \dot{z} = (\overline{A}(z_1)z)^\top P z + z^\top P \overline{A}(z_1)z\\ =& z^\top \overline{A}(z_1)^\top P z + z^\top P \overline{A}(z_1)z = z^\top(\overline{A}(z_1)^\top P + P\overline{A}(z_1))z \end{aligned} \tag{4.55}$$

Substitute values for P and A to get

$$\begin{aligned}\dot{V}_0 &= -\frac{1}{2|z_1|}z^\top \begin{bmatrix} 2\lambda\alpha_i + 4\epsilon(2\epsilon\alpha_i - \beta_i) + 4\epsilon\rho(q,\zeta) & \beta_i - 2\alpha_i\epsilon - \lambda - 4\epsilon^2 - \rho(q,\zeta) \\ \beta_i - 2\alpha_i\epsilon - \lambda - 4\epsilon^2 - \rho(q,\zeta) & 4\epsilon \end{bmatrix} z \\ &\leq -\frac{1}{2|z_1|}z^\top \tilde{Q} z \end{aligned} \tag{4.56}$$

where

$$\tilde{Q} = \begin{bmatrix} 2\lambda\alpha_i + 4\epsilon(2\epsilon\alpha_i - \beta_i) - 8\epsilon L & \beta_i - 2\alpha_i\epsilon - \lambda - 4\epsilon^2 - 2L \\ \beta_i - 2\alpha_i\epsilon - \lambda - 4\epsilon^2 - 2L & 4\epsilon \end{bmatrix} \tag{4.57}$$

$\tilde{Q}$ must be positive definite. α_i and β_i are therefore designed to make $\tilde{Q}$ positive definite with a minimum eigenvalue of 2ϵ. Substitute the gain $\beta_i = 2\epsilon\alpha_i + \lambda + 4\epsilon^2$ (4.40).

$$\tilde{Q} = \begin{bmatrix} 2\lambda\alpha_i - 4\epsilon(\lambda + 4\epsilon^2) - 8\epsilon L & -2L \\ -2L & 4\epsilon \end{bmatrix} \tag{4.58}$$

An α_i that assures positive definite $\tilde{Q}$ with a minimum eigenvalue of 2ϵ is found as follows (Shtessel et al.; 2011):

$$\begin{aligned} 4\epsilon(2\lambda\alpha_i - 4\epsilon(\lambda + 4\epsilon^2) - 8\epsilon L - 2\epsilon) - 4L^2 > 0 \\ \alpha_i > \frac{L^2}{2\lambda\epsilon} + \frac{2\epsilon(\lambda + 4\epsilon^2) + 4\epsilon L + \epsilon}{\lambda} \end{aligned} \tag{4.59}$$

With these conditions on α_i and β_i it can be shown that

$$\dot{V}_0 \leq r V_0^{1/2} \quad \text{where} \quad r = \frac{\epsilon\lambda_{min}^{1/2}(P)}{\lambda_{max}(P)}. \tag{4.60}$$

The minimum eigenvalue of $\tilde{Q}$ is 2ϵ the bound on $\dot{V}_0$ can be found from

$$\dot{V}_0 \leq -\frac{1}{2|z_1|}z^\top \tilde{Q} z \leq -\frac{2\epsilon}{2|z_1|}z^\top z = -\frac{\epsilon}{|z_1|}||z||^2 \tag{4.61}$$

where $||z||^2 = z_1^2 + z_2^2 = |s_i| + z_2^2$. The bounds on $V_0 = z^\top P z$ can be expressed with the eigenvalues

$$\lambda_{min}(P)||z||^2 \leq z^\top P z \leq \lambda_{max}(P)||z||^2 \tag{4.62}$$

$$|z_1| = |s_i|^{1/2} \leq ||z|| \leq \frac{V_0^{1/2}}{\lambda_{min}^{1/2}(P)} \tag{4.63}$$

and finally the bound on $\dot{V}_0$ (4.60) is found.

Now, consider the complete derivative of the Lyapunov function (4.54). Bounds on the derivative of the Lyapunov function are found with the same method as in (Shtessel et al.; 2010):

$$\begin{aligned}\dot{V}(z,\alpha_i,\beta_i) \leq & -rV_0^{1/2} + \frac{1}{\gamma_1}(\alpha_i - \alpha_i^*)\dot{\alpha_i} + \frac{1}{\gamma_2}(\beta_i - \beta_i^*)\dot{\beta}_i \\ = & -rV_0^{1/2} - \frac{\omega_1}{\sqrt{2\gamma_1}}|\alpha_i - \alpha_i^*| - \frac{\omega_2}{\sqrt{2\gamma_2}}|\beta_i - \beta_i^*| + \frac{1}{\gamma_1}(\alpha_i - \alpha_i^*)\dot{\alpha_i} \\ & + \frac{1}{\gamma_2}(\beta_i - \beta_i^*)\dot{\beta}_i + \frac{\omega_1}{\sqrt{2\gamma_1}}|\alpha_i - \alpha_i^*| + \frac{\omega_2}{\sqrt{2\gamma_2}}|\beta_i - \beta_i^*|\end{aligned} \tag{4.64}$$

Using the fact that $(x^2 + y^2 + z^2)^{1/2} \leq |x| + |y| + |z|$ (Shtessel et al.; 2010), the following bound on the first part of the right-hand-side of (4.64) is derived

$$-rV_0^{1/2} - \frac{\omega_1}{\sqrt{2\gamma_1}}|\alpha_i - \alpha_i^*| - \frac{\omega_2}{\sqrt{2\gamma_2}}|\beta_i - \beta_i^*| \leq -\mu\sqrt{V(z,\alpha_i,\beta_i)} \tag{4.65}$$

for $\mu = min(r, \omega_1, \omega_2)$ (Shtessel et al.; 2011). This results in the following bound on the complete derivative of the Lyapunov function.

$$\dot{V}(z,\alpha_i,\beta_i) \leq -\mu\sqrt{V(z,\alpha_i,\beta_i)} + \frac{1}{\gamma_1}(\alpha_i - \alpha_i^*)\dot{\alpha_i} + \frac{1}{\gamma_2}(\beta_i - \beta_i^*)\dot{\beta}_i + \frac{\omega_1}{\sqrt{2\gamma_1}}|\alpha_i - \alpha_i^*| + \frac{\omega_2}{\sqrt{2\gamma_2}}|\beta_i - \beta_i^*| \tag{4.66}$$

Assume that α_i and β_i (4.40) are bounded such that is is always possible to find constants α_i^* and β_i^* such that $\alpha_i - \alpha_i^* < 0$ and $\beta_i - \beta_i^* < 0$. That α_i and β_i are bounded

will be shown later. Then, the last four terms of (4.66) is equal zero if

$$\dot{\alpha}_i = \omega_1 \sqrt{\frac{\gamma_1}{2}} \qquad \dot{\beta}_i = \omega_2 \sqrt{\frac{\gamma_2}{2}} \tag{4.67}$$

because $-|\alpha_i - \alpha_i^*| = (\alpha_i - \alpha_i^*)$ and $-|\beta_i - \beta_i^*| = (\beta_i - \beta_i^*)$. This $\dot{\alpha}$ (4.67) is the adaptive gain $\dot{\alpha}_i$ presented in (4.40).

Consider β_i defined in (4.40). The derivative of β_i is $\dot{\beta}_i = 2\epsilon\dot{\alpha}_i$. Therefore, ϵ must take on the following value to make the last four terms of (4.66) is equal zero.

$$\epsilon = \frac{\omega_2}{2\omega_1} \sqrt{\frac{\gamma_2}{\gamma_1}} \tag{4.68}$$

These conditions result in

$$\dot{V}(z, \alpha_i, \beta_i) \leq -\mu \sqrt{V(z, \alpha_i, \beta_i)}. \tag{4.69}$$

It then remains to show that α_i and β_i are bounded and that the sliding surface is reached in finite time. First, consider the adaptive gains. The expression for α can be written as (Shtessel et al.; 2010)

$$\alpha_i = \begin{cases} \alpha_i(0) + \omega_1 \sqrt{\frac{\gamma_1}{2}} t & 0 \leq t \leq t_r \\ \alpha_i(0) + \omega_1 \sqrt{\frac{\gamma_1}{2}} t_r & t > t_r \end{cases} \tag{4.70}$$

where t_r finite reaching time. Therefore, α_i is bounded. In addition, β_i is a function of α_i and is therefore bounded when α_i is bounded.

The condition on α_i, the inequality (4.59), is reached in finite time because α_i increases linearly with time and the right-hand side of the inequality is bounded (L is some bounded constant). When the inequality holds, the control law drives the sliding variable s_i and $\dot{s}_i$ to zero in the finite reaching time given by

$$t_r \leq \frac{2V^{1/2}(t_0)}{\mu} \tag{4.71}$$

which is found from the inequality (4.69) (Shtessel et al.; 2010). As soon as the sliding surface is reached the state $\tilde{y}_i$, $i = 1...(6 + n)$, approaches zero with exponential dynamics (Antonelli; 2014) shown in section 4.3.1.

4.4 Non-regressor-based adaptive controller

The AIDC presented in section 4.2 is based on linearity in parameters (equation 4.8) and the computational requirement increases with the number of unknown system parameters (Antonelli; 2014). While the previous AIDC presented in section 4.2 is based on the regressor matrix which requires knowledge of the structure model matrices, the non-regressor-based adaptive controller (NRAC) requires only information about number of inputs and outputs as well knowledge of existence of bounds on the system matrices (Zhao and Yuh; 2005). The NRAC presented in this section is based on the bound estimation method (Sarkar et al.; 1999). The controller estimates and updates parameters defined by unknown bounds of the model matrices. The controller adjust the gains based on the performance of the system performance rather than knowledge of the dynamic model (Sarkar et al.; 1999).

The non-regressor-based controller for UVMSs in (Sarkar et al.; 1999) is based on the assumtion that the vehicle and the manipulator are decoupled. They therefore consider two subsystems with different bandwidth. The vehicle is heavier and slower than the faster and lighter manipulator subsystem. For the USM, this assumption does not hold. Therefore, a new controller similar to that of (Sarkar et al.; 1999) is developed and applied to the USM.

4.4.1 Control design

Instead of considering two separate subsystems, a controller is developed based on $\dot{\zeta}_d$, ζ, a position error variable $\tilde{y}$ and its derivative $\dot{\tilde{y}}$. The developed controller is based on these variables to give an give advantage in the stability analysis of the controller. The

proposed control law is given as follows:

$$\tau_c = B_{tot}^{\dagger}\left[K_1\dot{\zeta}_d + K_2\zeta + K_3\kappa + K_4\dot{\tilde{y}} + K_5\tilde{y}\right] = \sum_{i=1}^{5} K_i\phi_i. \tag{4.72}$$

where $\phi_3 = \kappa > 0$ is a constant and $\phi_1 = \dot{\zeta}_d$, $\phi_2 = \zeta$, $\phi_4 = \dot{\tilde{y}}$ and $\phi_5 = \tilde{y}$. It will be shown that the control law (4.72) drives the error variable s to zero. The error variable $\tilde{y}$ is defined as:

$$\tilde{y} = \begin{bmatrix} R_I^b(\eta_{1,d} - \eta_1) \\ \eta_Q\epsilon_d - \eta_{Q,d}\epsilon + \mathrm{S}(\epsilon)\epsilon_d \\ q_d - q \end{bmatrix} = \begin{bmatrix} R_I^b\tilde{\eta}_1 \\ \tilde{\epsilon} \\ \tilde{q} \end{bmatrix}. \tag{4.73}$$

The time derivative of the error variable $\tilde{y}$ can be expressed as:

$$\dot{\tilde{y}} = \begin{bmatrix} R_I^b\dot{\tilde{\eta}} + \dot{R}_I^B\tilde{\eta} \\ \dot{\tilde{\epsilon}} \\ \dot{\tilde{q}} \end{bmatrix} = \begin{bmatrix} \tilde{v_1} + \dot{R}_I^B\tilde{\eta} \\ (\tilde{\eta}_Q I_3 + \boldsymbol{S}(\epsilon))\tilde{v_2} \\ \dot{\tilde{q}} \end{bmatrix}. \tag{4.74}$$

The rotation matrix R_I^b is included to give the position error in the body frame. The gains matrices K_i are defined as:

$$K_i = \frac{\hat{\theta}_i s\phi_i^{\top}}{\|s\|\|\phi_i\|} \tag{4.75}$$

where

$$s = \tilde{\zeta} + \sigma\tilde{y}, \quad \sigma \in R^1, \quad \sigma > 0 \tag{4.76}$$

with $\tilde{\zeta} = \zeta_d - \zeta$ and

$$\dot{\hat{\theta}}_i = f_i\|s\|\|\phi_i\|. \tag{4.77}$$

$K_3 \in R^{6+n}$ is a vector and K_1, K_2, K_4 and $K_5 \in R^{(6+n)\times(6+n)}$ are matrices. $\|x\|$ is the euclidean norm and defined as in (Khalil; 2002) with $x \in R^n$:

$$\|x\| = \sqrt{(\|x_1\|^2 + ... + \|x_n\|^2)} = \sqrt{x^\top x}. \tag{4.78}$$

It will be shown that the control input (4.72) stabilizes the error variable asymptotically to zero.

For implementation purposes, the gain matrices K_i should not be used directly as it will give large control inputs near $\|s\|\|\phi_i\| = 0$ (Sarkar et al.; 1999). The gains (4.75) are therefore implemented as:

$$K_i = \begin{cases} \frac{\hat{\theta}_i s \phi_i^\top}{\|s\|\|\phi_i\|}, & \text{if } \|s\|\|\phi_i\| > \delta_i \\ \frac{\hat{\theta}_i s \phi_i^\top}{\delta_i}, & \text{if } \|s\|\|\phi_i\| \leq \delta_i \end{cases} \tag{4.79}$$

4.4.2 Stability analysis

The NRAC is based on the bound estimation method. The system matrices will therefore be assumed to be bounded as is done in (Sarkar et al.; 1999) by the following constants:

$$\begin{aligned} &\text{(i) } \|M\| \leq \beta_1, \\ &\text{(ii) } \|C + D\| \leq \beta_2 \\ &\text{(iii) } \|g\| \leq \beta_3, \\ &\text{(iv) } \|M^{-1}\| \leq \alpha, \\ &\text{(v) } \lambda_{min}(M^{-1}) > \gamma, \end{aligned} \tag{4.80}$$

where β_i, α and γ are positive constants (Sarkar et al.; 1999). The bounds (i) and (iii) follows from property 2.1 and 2.5 respectively. Property (iv) and (v) follows from the fact that M is symmetric positive definite and the inverse of a symmetric positive definite matrix is symmetric positive definite. Finally, assumption (ii) can be justified based on property 2.3 with the idea that ζ cannot be infinitely large and that the

hydrodynamic forces cannot be infinitely large in a real system. It will in this section be presented how to estimate:

$$\theta_i = \frac{\alpha\beta_i}{\gamma} \tag{4.81}$$

where $\beta_4 = \beta_5 = \frac{\mu}{\alpha}$ with $\mu > \sigma$. α, γ, β_1, β_2 and β_3 from (4.80).

Consider the following Lyapunov function candidate proposed in (Sarkar et al.; 1999):

$$V = \frac{1}{2}s^\top s + \frac{1}{2}\sum_{i=1}^{5}\frac{1}{f_i}\gamma(\theta_i - \hat{\theta}_i)^2. \tag{4.82}$$

The time derivative of V is:

$$\dot{V} = s^\top\dot{\tilde{\zeta}} + \sigma s^\top\dot{\tilde{y}} - \sum_{i=1}^{5}\frac{1}{f_i}\gamma(\theta_i - \hat{\theta}_i)\dot{\hat{\theta}}_1. \tag{4.83}$$

In order to find an expression for $\dot{\tilde{\zeta}}$, the expression for $\dot{\zeta}$ from the dynamics (4.4) is used. The resulting expression is written as a sum for convenience:

$$\begin{aligned}
\dot{\tilde{\zeta}} &= \dot{\zeta}_d - \dot{\zeta} = \dot{\zeta}_d - M^{-1}\Big(B_{tot}\tau_c - C\zeta - D\zeta - g\Big) \\
&= \dot{\zeta}_d - M^{-1}\Big(K_1\dot{\zeta}_d + K_2\zeta + K_3\kappa + K_4\dot{\tilde{y}} + K_5\tilde{y} - C\zeta - D\zeta - g\Big) \\
&= M^{-1}(M - K_1)\dot{\zeta}_d + M^{-1}(C + D - K_2)\zeta + M^{-1}(\frac{1}{\kappa}g - K_3)\kappa - M^{-1}K_4\dot{\tilde{y}} - M^{-1}K_5\tilde{y} \\
&= M^{-1}\sum_{i=1}^{5}(P_i - K_i)\phi_i
\end{aligned} \tag{4.84}$$

where $P_1 = M$, $P_2 = C + D$, $P_3 = \frac{1}{\kappa}g$ and $P_4 = P_5 = 0$. The complete derivative of the

Lyapunov function candidate (4.82) can then be written as:

$$\begin{aligned}\dot{V} &= s^\top M^{-1}\sum_{i=1}^{5}(P_i - K_i)\phi_i + \sigma s^\top \dot{\tilde{y}} - \sum_{i=1}^{5}\frac{1}{f_i}\gamma(\theta_i - \hat{\theta}_i)\dot{\hat{\theta}}_i \\ &= \underbrace{\left(s^\top M^{-1}\sum_{i=1}^{5}P_i\phi_i - \sum_{i=1}^{5}\frac{1}{f_i}\gamma\theta_i\dot{\hat{\theta}}_i + \sigma s^\top\dot{\tilde{y}}\right)}_{\dot{V}_1} + \underbrace{\left(\sum_{i=1}^{5}\frac{1}{f_i}\gamma\hat{\theta}_i\dot{\hat{\theta}}_i - s^\top M^{-1}\sum_{i=1}^{5}K_i\phi_i\right)}_{\dot{V}_2}\end{aligned} \tag{4.85}$$

The expressions in the brackets will be considered separately and it will be shown that both $\dot{V}_1 \leq 0$ and $\dot{V}_2 \leq 0$. First, consider the expression in the first bracket, $\dot{V}_1$. Substitute θ_i from (4.81) and $\dot{\hat{\theta}}_i$ from (4.77). $\dot{V}_1$ can then be written as:

$$\dot{V}_1 = s^\top M^{-1}\sum_{i=1}^{3}P_i\phi_i - \sum_{i=1}^{3}\alpha\beta\|s\|\|\phi_i\| - \mu\|s\|\|\tilde{y}\| - \mu\|s\|\|\dot{\tilde{y}}\| + \sigma s^\top\tilde{y} \tag{4.86}$$

Due the fact that $s^\top\dot{\tilde{y}} \leq \|s\|\|\dot{\tilde{y}}\|$ and the previously stated bounds on the matrices (4.80), it can be shown that $\dot{V}_1 \leq 0$ when $\mu > \sigma$:

$$\dot{V}_1 \leq \sum_{i=1}^{3}(\|M^{-1}\|\|P_i\| - \alpha\beta_i)\|s\|\|\phi_i\| + (\sigma - \mu)\|s\|\|\dot{\tilde{y}}\| - \mu\|s\|\|\tilde{y}\| \leq 0. \tag{4.87}$$

Consider the expression in the second bracket. Using the expression for the euclidean norm (4.78) and the bound on M^{-1} (4.80) it can be shown that $\dot{V}_2 \leq 0$:

$$\begin{aligned}\dot{V}_2 &= \sum_{i=1}^{5}\gamma\hat{\theta}_i\|s\|\|\phi_i\| - s^\top M^{-1}\sum_{i=1}^{5}\frac{\hat{\theta}_i s\phi_i^\top\phi_i}{\|s\|\|\phi_i\|} = \sum_{i=1}^{5}\left(\gamma - \frac{s^\top M^{-1}s}{s^\top s}\right)\|s\|\|\phi_i\|\hat{\theta}_i \\ &\leq \sum_{i=1}^{5}\left(\gamma - \lambda_{min}(M^{-1})\right)\|s\|\|\phi_i\|\hat{\theta}_i \leq 0.\end{aligned} \tag{4.88}$$

This results in $\dot{V} = \dot{V}_1 + \dot{V}_2 < 0$ for all $s \neq 0$ with the unknown bounds on the matrices (4.80). To elaborate this, consider the case when $s \neq 0$. In this case, either $\tilde{\zeta}$ or $\tilde{y}$ or both must be not equal to zero from the definition of s. Consider the last two terms of

(4.87) with $\mu > \sigma$:

$$\dot{V}_{1,1} = (\sigma - \mu)\|s\|\|\dot{\tilde{y}}\| - \mu\|s\|\|\tilde{y}\| \tag{4.89}$$

If $\tilde{y} = 0$, then $\tilde{\zeta} \neq 0$. From the expression of $\dot{\tilde{y}}$ (4.74) it can be seen that $\dot{\tilde{y}} \neq 0$ when $\tilde{\zeta} \neq 0$. Therefore, $\dot{V}_{1,1} < 0$ when $\tilde{\zeta} \neq 0$. If $\tilde{y} \neq 0$ then $\dot{V}_{1,1}$ is also strictly less than zero. Therefore $\dot{V} < 0$ for all $s \neq 0$. As $V < 0$ for all $s \neq 0$ it can be concluded based on theorem A.1 that with the control law τ_c (4.72), the error s will asymptotically converge to zero (Sarkar et al.; 1999). Unmodeled dynamics can easily be included in the stability proof as is done in (Sarkar et al.; 1999).

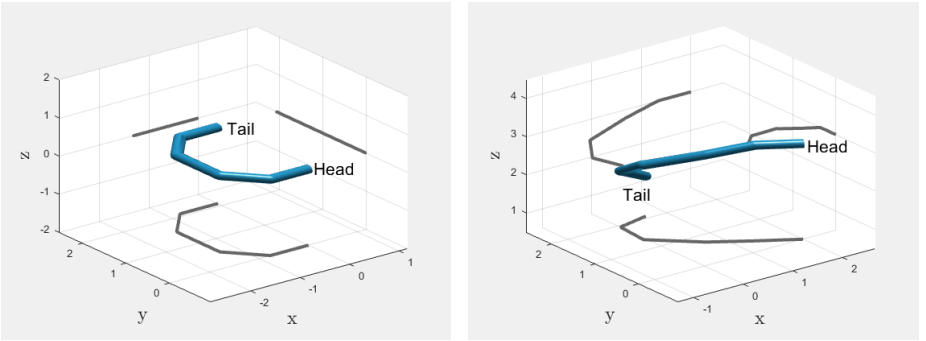
For the non-regressor based adaptive controller the condition $\tilde{\zeta} = -\sigma\tilde{y}$ is reached asymptotically. This is different than for the STA where the condition $\tilde{\zeta} = -\sigma\tilde{y}$ is reached in finite time.

4.5 Simulations and Discussion

Simulations with the controllers presented above have been done to demonstrate the performance of the controllers. The simulation results are presented in this section. The control framework is implemented as in figure 3.4 with the reference trajectory generation block, inverse kinematics block and thrust allocation block implemented as explained in section 3.4.1, 3.4.2 and 3.4.3 respectively. The difference between the simulations is only the dynamic controllers.

In the simulations, the input is a trajectory of desired end-effector positions and orientations $\eta_{ee,d}$ for the end-effector of the USM to follow. This can be related to a situation in which the USM has a camera mounted in its head link and is to make the camera follow a certain trajectory. The initial and final configuration of the USM is shown in figure 4.1a and 4.1b respectively. The trajectory begins in the origin of the inertial frame with zero rotation and ends in the position $\eta_{1,ee} = [2, 0, 3]^\top$ with rotation $\eta_{2,ee} = [0, 0, -0.5]^\top$. In the initial configuration, the joint angles of the z-revolute joints are $\frac{\pi}{4}$ and the joint angles of the y-revolute joints are 0.

It is assumed in all simulations that the joint motors and thrusters are able to apply exactly the calculated control input u_J and u_T.



(a) Initial configuration.

(b) Final configuration.

Figure 4.1: Visualization of the initial and final configuration of the USM from simulations. Initial end-effector position and orientation is $\eta_{ee} = [0, 0, 0, 0, 0, 0]^\top$ and final end-effector position and orientation is $\eta_{ee} = [2, 0, 3, 0, 0, -0.5]^\top$.

4.5.1 Adaptive inverse dynamics control

In section 4.2.3 it was shown that the system with the adaptive inverse dynamics controller (AIDC) gives states $\tilde{y}$ and s that go to zero. Asymptotic stability was not shown as the state $\tilde{\theta} = \theta - \hat{\theta}$ is only guaranteed to be bounded. The computational requirement for the AIDC is quite large as it requires computation of system matrices and the regressor matrix.

The solver used for simulations with the AIDC is ode15s with relative tolerance 10^{-3}. The gains matrices used in the simulation are chosen to give good simulation results as

$$\begin{aligned} K_D &= 500 I_{14} \\ K_P &= blockdiag\{I_3, I_3, 30 I_8\}, \\ K_\theta &= 10^{-3} \times diag\{1, 0.01, 0.1, 100\} \text{ and} \\ \Lambda &= blockdiag\{40 I_3, 40 I_3, 40 I_8\}. \end{aligned} \tag{4.90}$$

These matrices satisfy the required conditions on the gains presented in 4.2.1.

The resulting end-effector positions and orientations η_{ee} are shown in figure 4.2. The figure shows that the AIDC gives great tracking of the desired end-effector

traejctory in the simulation. It was assumed in the development of the controller as well as in the simulations that the model is completely known except for the hydrodynamic damping terms $D(q,\zeta)\zeta$. When this is the case, the AIDC performs very well. For applications with a real USM, it is not very likely that one has exact knowledge of the model except hydrodynamic damping terms and this will affect the performance of the controllers. Added mass effects are, although less significant for slow steady motion, for example difficult to model accurately.

Figure 4.3 shows the time evolution of the error variable $s = \tilde{\zeta} + \Lambda\tilde{y}$. All error variables s_i, $i = 1...14$, converge to zero as expected. The errors before convergence in the simulations are small, in order of magnitude 10^{-4} to 10^{-5}. The initial oscillations in the error originate from the oscillations in the estimate $\hat{\theta}$ evident in figure 4.5.

Figure 4.4 shows the time evolution of the error variable $\tilde{y}$. It is evident that the $\tilde{y}$ approaches zeros as expected from the stability analysis.

While it was shown that the values of s and $\tilde{y}$ converge to zero, it was shown that the estimate error $\tilde{\theta}$ will only be bounded. With the presented AIDC, estimate vector $\hat{\theta}$ contains the estimated drag coefficients. The estimated drag coefficients with their actual values are shown in figure 4.5. The figure contains both the estimates when the actual drag coefficients are the original drag coefficients from the model, $\theta = [0.2, 0.1, 0.8, 0.1]^\top$, and the estimates when drag coefficients in the model are changed to $\theta = [0.5, 0.5, 0.5, 0.5]^\top$. The figure shows that the drag coefficients do not converge completely to the actual values, but the errors $\tilde{\theta}$ are bounded as expected. Initially, the estimated drag coefficients in $\hat{\theta}$ are zero such that the term $\phi(q,\zeta,\zeta_r)\hat{\theta}$ is zero. The estimate $\phi(q,\zeta,\zeta_r)\hat{\theta}$ is supposed to model the hydrodynamic damping term $D(q,\zeta)\zeta_r$. When the estimate is zero, no hydrodynamic damping is included in the controller. As soon as the USM begins to move, the estimates reach $\hat{\theta} \neq 0$ which introduces hydrodynamic damping terms in the controller. Changing the gain matrix K_θ changes the rate of change of the estimates $\hat{\theta}$.

It is also interesting to look at how the estimate $\phi(q,\zeta,\zeta_r)\hat{\theta}$ evolves over time. Figure 4.6 shows the estimate $\phi(q,\zeta,\zeta_r)\hat{\theta}$ with the value of $D(q,\zeta)\zeta_r$. Even though the drag coefficients do not reach their actual values, the estimate of $D(q,\zeta)\zeta_r$ is very similar to the actual values of $D(q,\zeta)\zeta_r$ as seen from the figure. The blue lines in the

figure shows the values of $\phi(q, \zeta, \zeta_r)\hat{\theta}$ when the actual value of the estimates θ are $[0.2, 0.1, 0.8, 0.1]^\top$. The green line is when the actual values are $[0.5, 0.5, 0.5, 0.5]^\top$. The case where the actual values are 0.5 are included to show that the controller gives good estimates of $D(q, \zeta)\zeta_r$ with different values of the drag coefficients. If $\phi(q, \zeta, \zeta_r)\hat{\theta} = D(q, \zeta)\zeta_r$ the controller is stable as is shown in the stability analysis in appendix A.3. As seen from the figure, the estimated values follow the actual values which shows intuitively why the controller works. There are some small oscillations in the values of $\phi(q, \zeta, \zeta_r)\hat{\theta}$ initially resulting from the oscillations on $\hat{\theta}$, but these disappear fast.

To summarize, the AIDC shows expected behaviour in simulation and seem to be suitable for dynamic motion control for USMs. Practical considerations such as limited knowledge of the system matrices M, C and g and the computational complexity of the AIDC has however not been taken into account in the simulations.

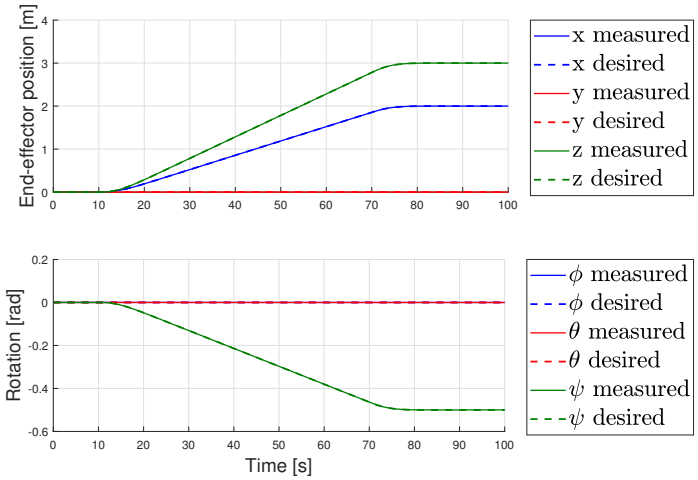


Figure 4.2: AIDC - The time evolution of η_{ee} with $\eta_{ee,d}$. The x,y and z positions of the end-effector, $\eta_{1,ee}$, are collected in the first subplot. The roll ϕ, pitch θ and yaw ψ, $\eta_{2,ee}$, are collected in the second subplot.

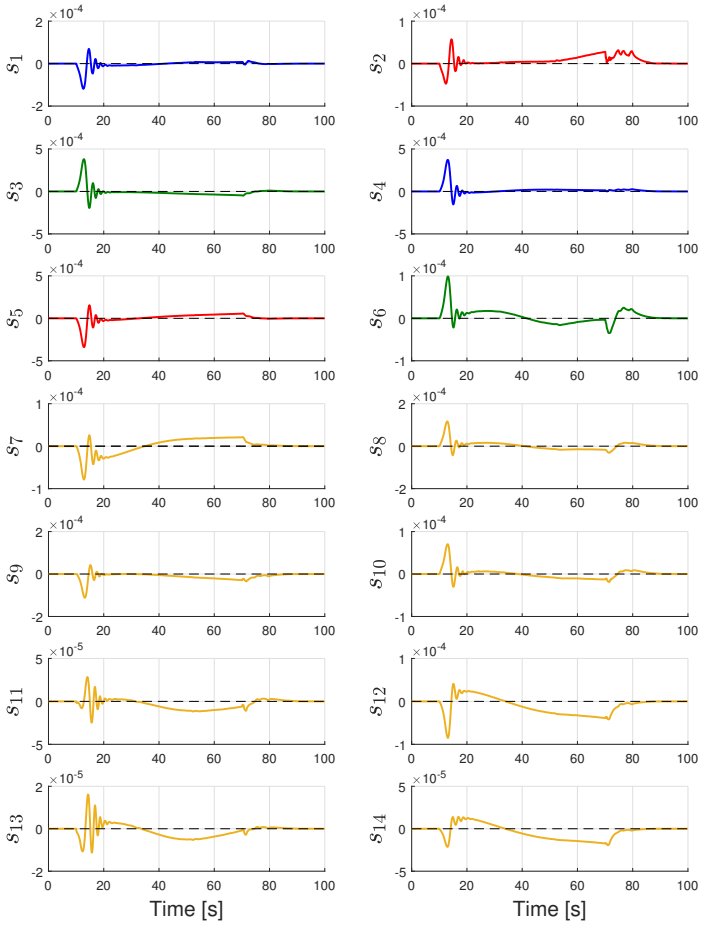


Figure 4.3: AIDC - The time evolution of the error variable $s = \tilde{\zeta} + \Lambda\tilde{y}$. s_i, i=1...14, is the i'th variable of the vector s. The plots in yellow are the errors associated with the joints while the blue, red and green are the errors associated with the x,y and z position/orientation respectively.

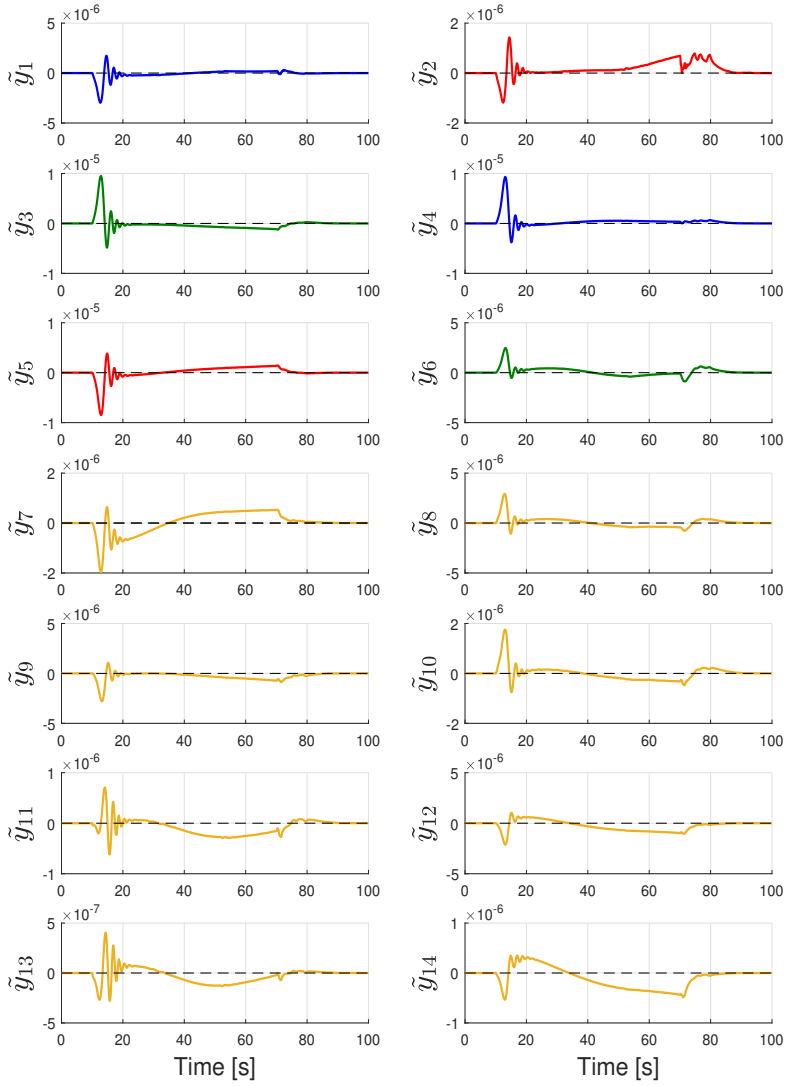


Figure 4.4: AIDC - The time evolution of the error variable $\tilde{y}$. $\tilde{y}_i$, i=1...14, is the i'th variable of the vector $\tilde{y}$. The plots in yellow are the errors associated with the joints while the blue, red and green are the errors associated with the x,y and z position/orientation respectively.

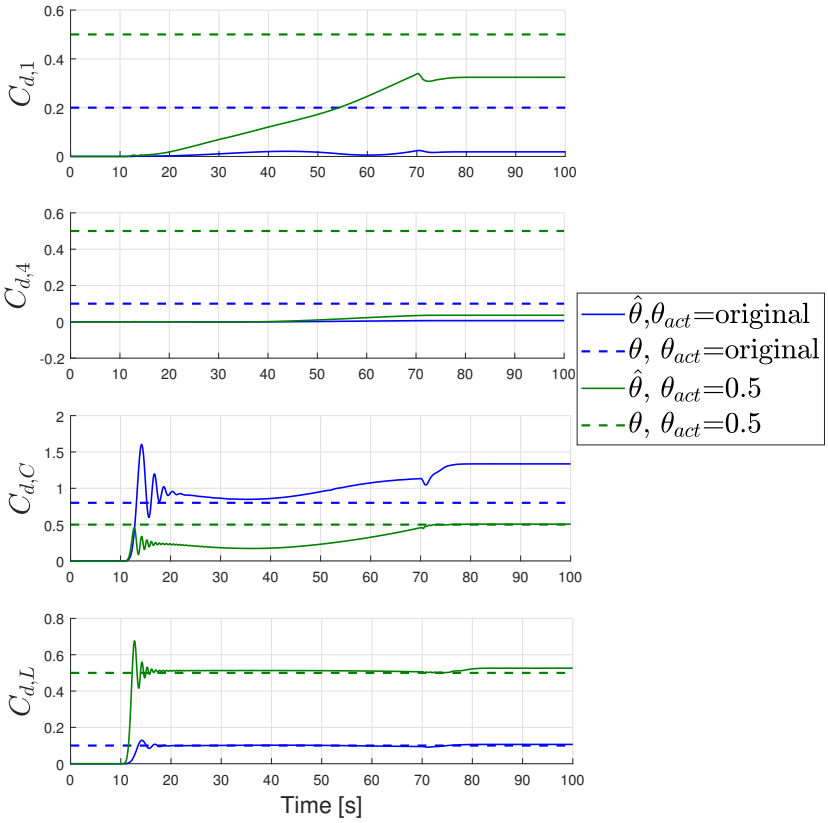


Figure 4.5: AIDC - The time evolution of estimated drag coefficients $\hat{\theta}$ with the actual values of the drag coefficients θ. The blue lines show the actual and estimated drag coefficients when $\theta = [0.2, 0.1, 0.8, 0.1]^\top$ and the green lines show the actual end estimated drag coefficients when $\theta = [0.5, 0.5, 0.5, 0.5]^\top$.

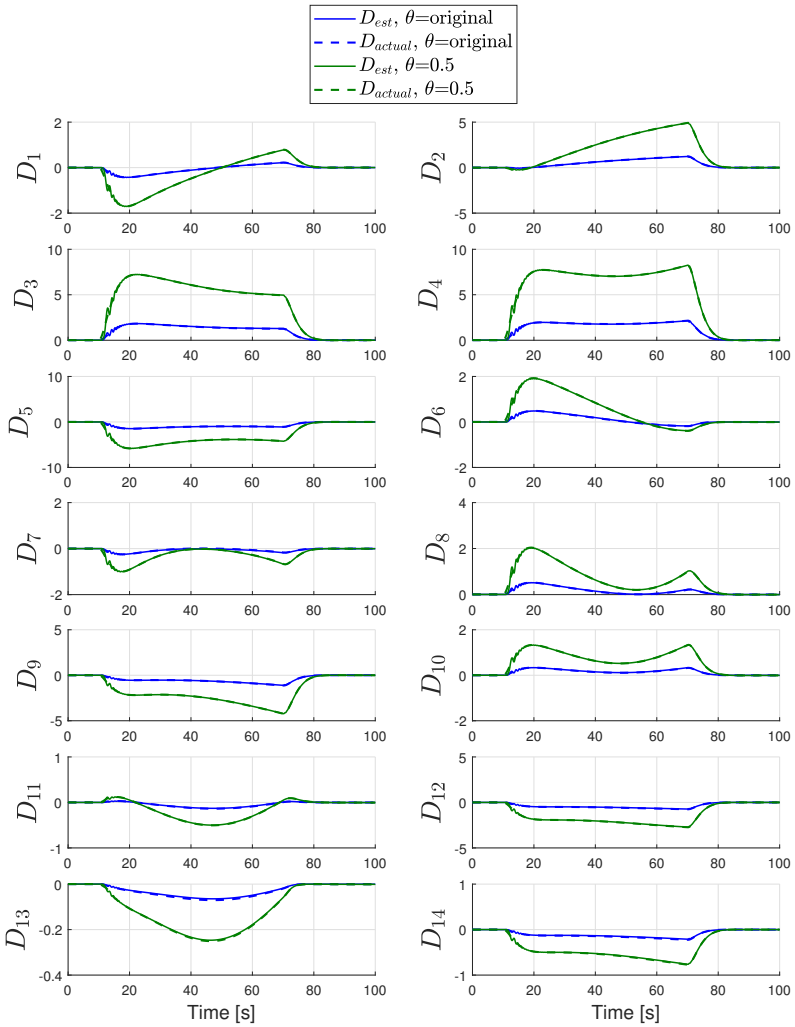


Figure 4.6: AIDC - The time evolution values of $\hat{D}(q,\zeta)\zeta_r = \phi(q,\zeta,\zeta_r)\hat{\theta}$ with the values of $D(q,\zeta)\zeta_r = \phi(q,\zeta,\zeta_r)\theta$. D_i, $i = 1...14$, are the i'th elements of the vectors. The blue lines show the actual values and estimates when $\theta = [0.2, 0.1, 0.8, 0.1]^\top$ and the green lines show the actual values end estimates when $\theta = [0.5, 0.5, 0.5, 0.5]^\top$.

4.5.2 Super-twisting algorithm with adaptive gains

In chapter 4 it was shown that the sliding surface s is reached in finite time with the super-twisting algorithm with adaptive gains with unknown hydrodynamic forces. When the sliding surface is reached, the error variable $\tilde{y}$ converges to zero.

The solver used for simulations with the STA is ode45 with relative tolerance 10^{-3}. The tuneable parameters of the STA are λ, γ, ϵ and α_m. These are chosen as follows to give satisfactory results:

$$\lambda = 15, \quad \gamma = 1, \quad \omega_i = 0.5, \quad \epsilon = 1, \text{ and } \alpha_m = 0.02. \tag{4.91}$$

Figure 4.7 shows the end-effector positions and orientations, η_{ee}, from the simulation with the STA with adaptive gains. The figure shows good tracking performance, but some initial oscillations are evident. The oscillations does however disappear quite quickly. In plot of Euler angles at the bottom of figure 4.7, the initial oscillations disappear after about 15 seconds and for the positions in the first plot of figure 4.7 the oscillations disappear even sooner.

Figure 4.8 shows the behaviour of the sliding surface $s = \tilde{y} + \lambda\tilde{\zeta}$ over time. As soon as the desired $\eta_{ee,d} \neq 0$, the error variables s_i become quite large and oscillating. The error does however become smaller quickly. After about 15 seconds, all s_i are within the bound $|s|_i < \alpha_m$ and stays within the bound throughout the simulation. The bound α_m is shown by the black dashed line in the figure. The sliding surface $|s|_i < \alpha_m$ is therefore reached in finite time as expected from the stability analysis. Quite a lot of chattering can be observed in the error signal resulting from the discontinuity of the controller.

Figure 4.9 shows the error variable $\tilde{y}$. It can be seen from the figure that all $\tilde{y}_i$ converges to zero as expected from the stability analysis. The expected *sliding* along the sliding surface is especially evident in for example the plot of $\tilde{y}_9$. When the sliding surface is reached after about 15 seconds, the larger oscillations stop and the variable *slides* towards zero.

Consider the control input on joints u_J and thrusters u_T shown in figure 4.10 and figure 4.11 respectively. The control input from the adaptive inverse dynamics

controller (AIDC) is plotted together with the control input for the STA. The control input for the STA is dominated by chattering, but its trend is similar to that of the AIDC. In the simulations it is assumed that the calculated control input is exactly what is applied to the USM. The control input from the STA is characterized by chattering and large values. It is not realistic that the joint motors and the thrusters will be able to provide this desired input. A saturation block was added to the control input to avoid too large thrust in joint torques in the model. If too large control inputs are applied to real joints and thrusters this may break the joints and thrusters. In addition, the chattering in the input signal may in a real system cause high heat loss, low control accuracy and cause wear on moving mechanical parts (Khalil; 2002). In that sense, the control input from the AIDC is better than the control input from the STA. One of the traits of the STA is that it attenuates chattering, but the attenuation of chattering is limited in this simulation. Smoothing the input signals could be done, but that would cause some, if not all, the relevant information in the to be lost. Attempts were made to change the gains to give smoother control inputs, but better performance was not achieved.

Figure 4.12 shows the adaptive gains α_1 and β_1. All α_i and β_i show similar behavior and only the gains for $i = 1$ is shown in the figure for readability. The gains stop increasing when the sliding surface $|s_i| < \alpha_m$ is reached. By comparing figure 4.12 and 4.8 it can be seen that when $|s|_i < \alpha_m$, the gains are constant. The gains are at this point large enough to overcome the unknown disturbances.

In summary, the STA with adaptive gains give good tracking of the end-effector trajectory with $\tilde{y} \rightarrow 0$, but the resulting control input is characterized by chattering that may damage the USM.

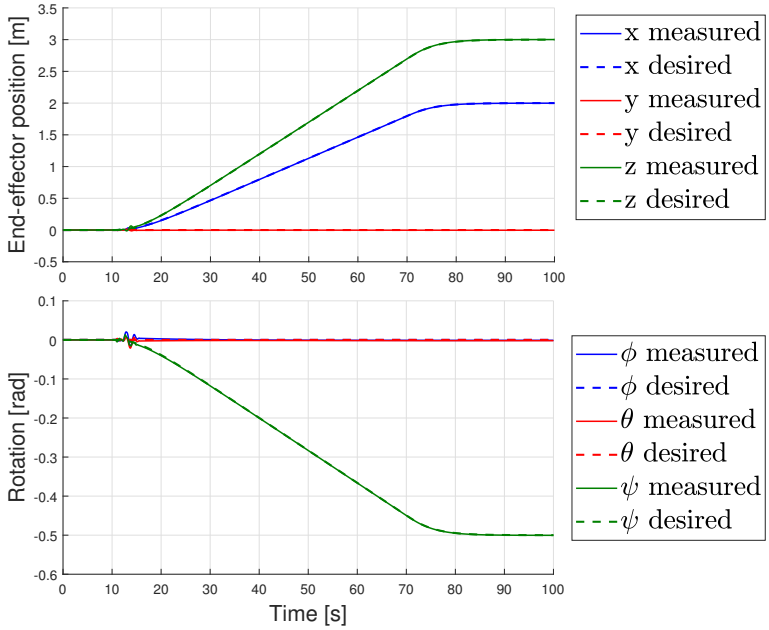


Figure 4.7: STA - The time evolution of η_{ee} with $\eta_{ee,d}$. The x,y and z positions of the end-effector, $\eta_{1,ee}$, are collected in the first subplot. The roll ϕ, pitch θ and yaw ψ, $\eta_{2,ee}$, are collected in the second subplot.

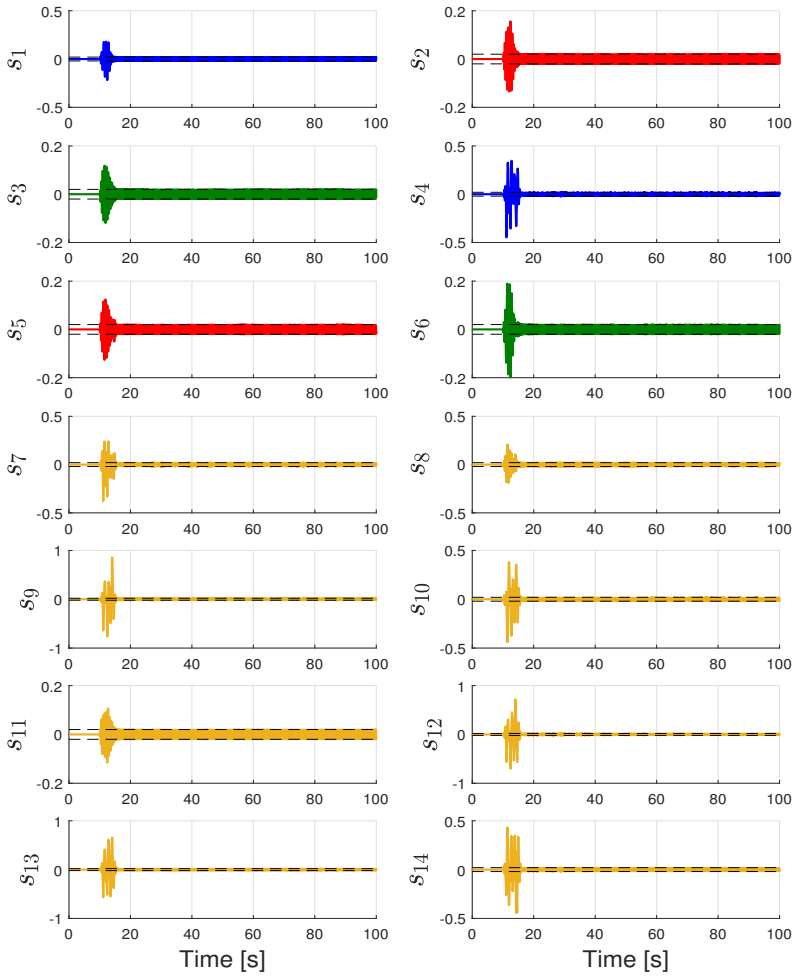


Figure 4.8: STA - The time evolution of the sliding surface $s = \tilde{\zeta} + \lambda\tilde{y}$. s_i, i=1...14, is the i'th variable of the vector s. The plots in yellow are the errors associated with the joints while the blue, red and green are the errors associated with the x,y and z position/orientation respectively.

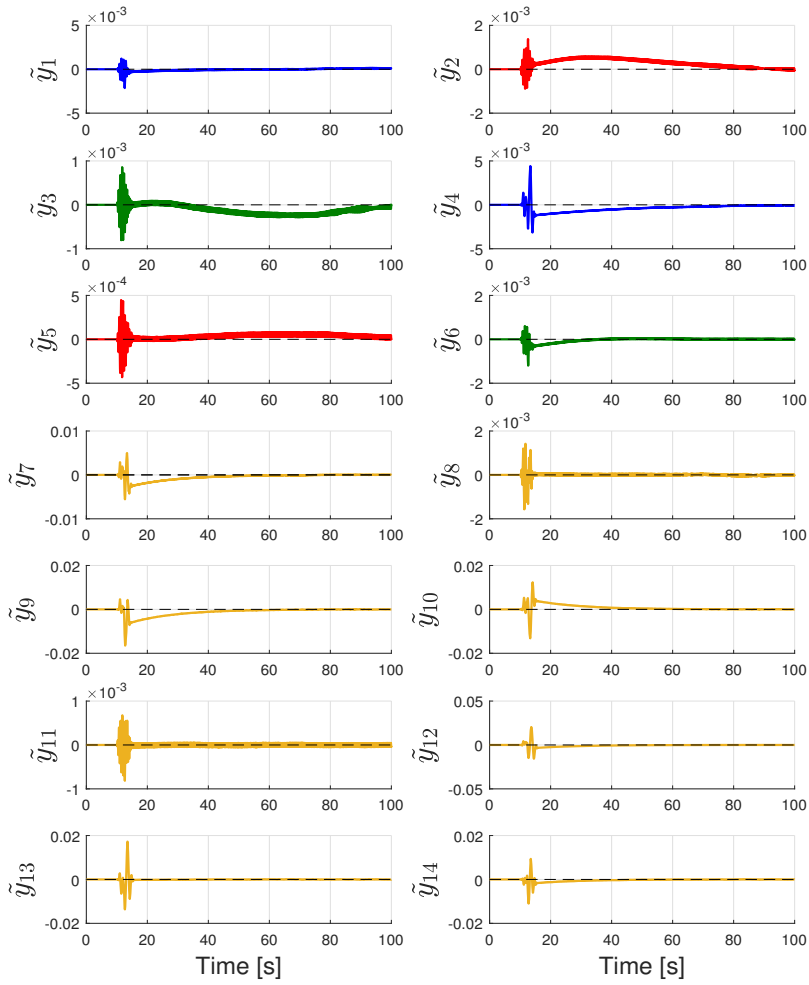


Figure 4.9: STA - The time evolution of the error variable $\tilde{y}$. $\tilde{y}_i$, i=1...14, is the i'th variable of the vector $\tilde{y}$. The plots in yellow are the errors associated with the joints while the blue, red and green are the errors associated with the x,y and z position/orientation respectively.

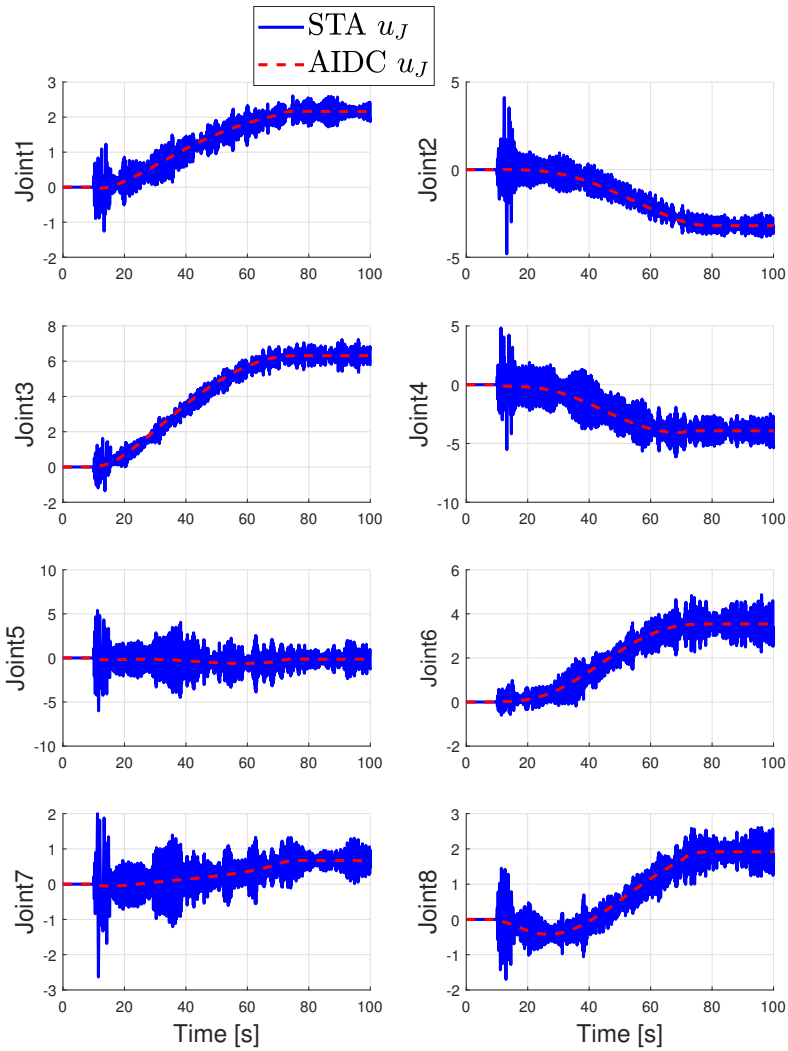


Figure 4.10: STA - The time evolution of applied joint torque u_J from the STA with applied joint torque u_J from the AIDC.

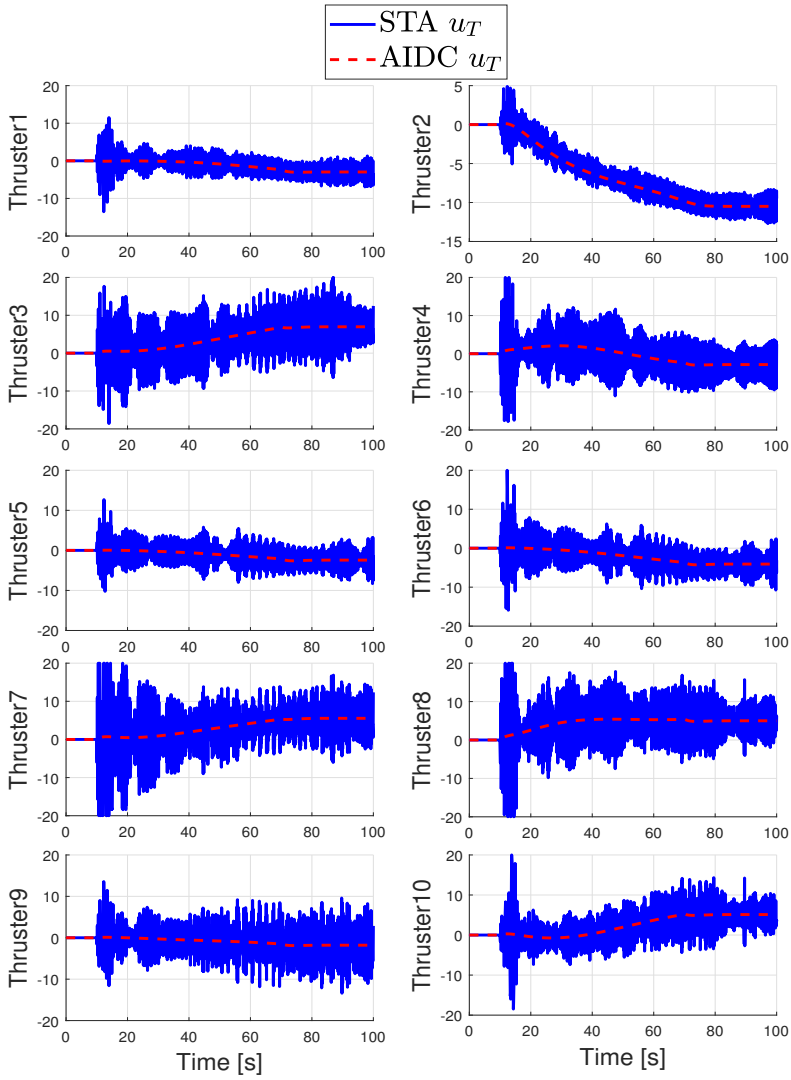


Figure 4.11: STA - The time evolution of thruster control input u_T from the STA with thruster control input u_T from the AIDC.

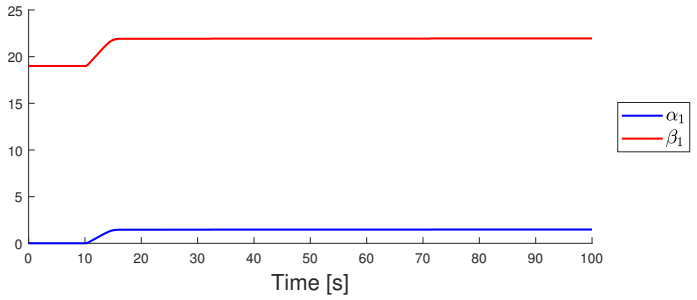


Figure 4.12: STA - The time evolution of the adaptive gains α_1 and β_1. All the adaptive gains α_i and β_i, $i = 1...14$ show similar behavior and only α_1 and β_i is included for readability.

4.5.3 Non-regressor-based adaptive control

It was shown in chapter 4 that the non-regressor-based adaptive controller (NRAC) drives the error variable s asymptotically to zero.

The solver used for simulations with the NRAC is ode45 with relative tolerance 10^{-3}. The parameters that must be specified for the controllers are σ, f and κ. The values of these are chosen to give good simulation results as:

$$\begin{aligned} \sigma &= 30, \\ f &= [100, 50, 50, 200, 100]^\top, \\ \kappa &= 0.15. \end{aligned} \tag{4.92}$$

The end-effector position and orientation is shown in figure 4.13. It can be seen from the plots that the there are some oscillations in the end-effector position and orientation initially, but after about 20 seconds these become unnoticeable for all positions and orientations in the plot. After the initial oscillations, the end-effector follows its desired trajectory well.

The time evolution of error variable $s = \tilde{\zeta} + \sigma\tilde{y}$ is shown in figure 4.14. The error oscillates initially, but approaches to zero after about 22 seconds. The initial values of the estimates are zero. Hence, the gains K_i are initially zero which leads to zero

control input. As soon as the error variables $s, \phi_i \neq 0$, the estimates begin to increase until they are large enough to model the unknown bounds of the system matrices. The behaviour of the error variable s is therefore as expected - larger error initially, but approach zero when the estimates are large enough. The time evolution of the estimates $\hat{\theta}$ is show in figure 4.15. All the estimates increase rapidly as soon as $s \neq 0$ and are steady or slowly increasing after about 22 seconds.

The error variable s will as shown in the stability analysis in section 4.4 approach zero, but the stability analysis does not show that $\tilde{y}$ goes to zero. Figure 4.16 shows the time evolution of $\tilde{y}$. The figure shows that $\tilde{y}$ approaches zero in the simulations. When both $\tilde{y} = 0$ and $s = 0$, $\tilde{\zeta}$ must be zero from the definition of the error variable s. In section 4.3.1 it was shown that in the system $\dot{\tilde{\zeta}} = -\sigma\tilde{y}$, the states converge to zero. Hence, when the variable $s = 0$, the states $\tilde{\zeta}$ and $\tilde{y}$ converges to zero. The difference between the stability proof with the STA and the NRAC is that for the STA it was shown that the sliding surface σ is reached in finite time, while for the NRAC it was shown that the error variable s approach zero asymptotically. In the simulations, the $s \approx 0$ after about 22 seconds. Therefore, the $\tilde{y}$ and $\tilde{\zeta}$ also approaches zero.

Figure 4.17 and 4.18 shows the control input on thrusters, u_T, and control input on joints, u_J, respectively. The figures also include the control input on joints and thrusters from the AIDC for comparison. From the figures, it can be seen that the control input has oscillations initially but stabilizes after about 25 seconds in the simulation. At this point, the error variable $\tilde{y}$ also stops oscillation. After this, the control input is similar to that from the AIDC. Initially, the estimates $\hat{\theta}$ are too small which cause the initial oscillation. When the values of $\hat{\theta}$ are large enough, the errors and hence the control inputs are stabilized. Consider a case when the initial value of the estimates are not equal to zero. Figures 4.17 and 4.18 also includes an example where the initial values of the estimates are equal or slightly less than the final estimated values in figure 4.15, $\hat{\theta} = [1, 2, 50, 2, 2]^\top$. In this case, the resulting control inputs are smooth. The end-effector also shows great tracking of the desired trajectory in this case. Therefore, if one has an idea of the bounds of the system matrices, a possibility is to use these as the initial values of the estimates in the NRAC.

To summarize, the NRAC gives good tracking of the end-effector trajectory and

the error variable s converges to zero. The NRAC requires a limited amount of prior knowledge of the system and is therefore attractive for its simplicity and computational efficiency (Sarkar et al.; 1999), but give undesirable initial oscillations of the control input when the initial estimate is $\hat{\theta} = [0, 0, 0, 0, 0]$. In general, it was observed in the simulations that the performance of the NRAC is sensitive to small changes of the tuneable parameters and therefore harder to work with than the two previous controllers.

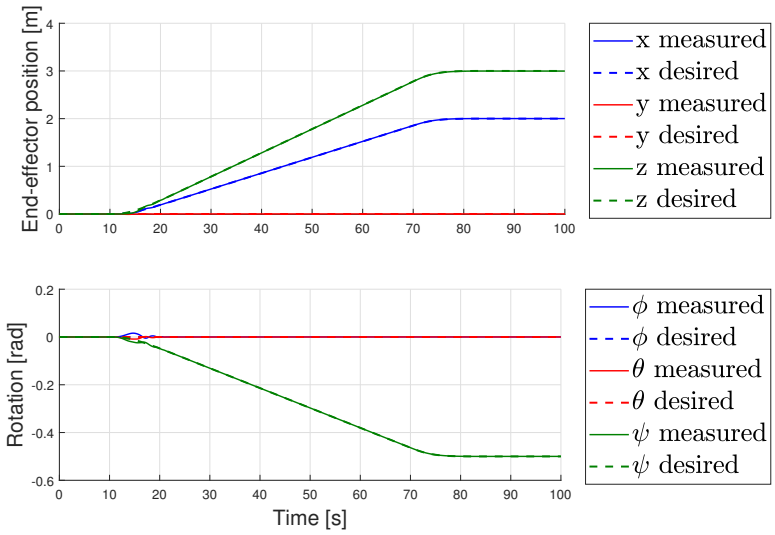


Figure 4.13: NRAC - The time evolution of η_{ee} with $\eta_{ee,d}$. The x,y and z positions of the end-effector, $\eta_{1,ee}$, are collected in the first subplot. The roll ϕ, pitch θ and yaw ψ, $\eta_{2,ee}$, are collected in the second subplot.

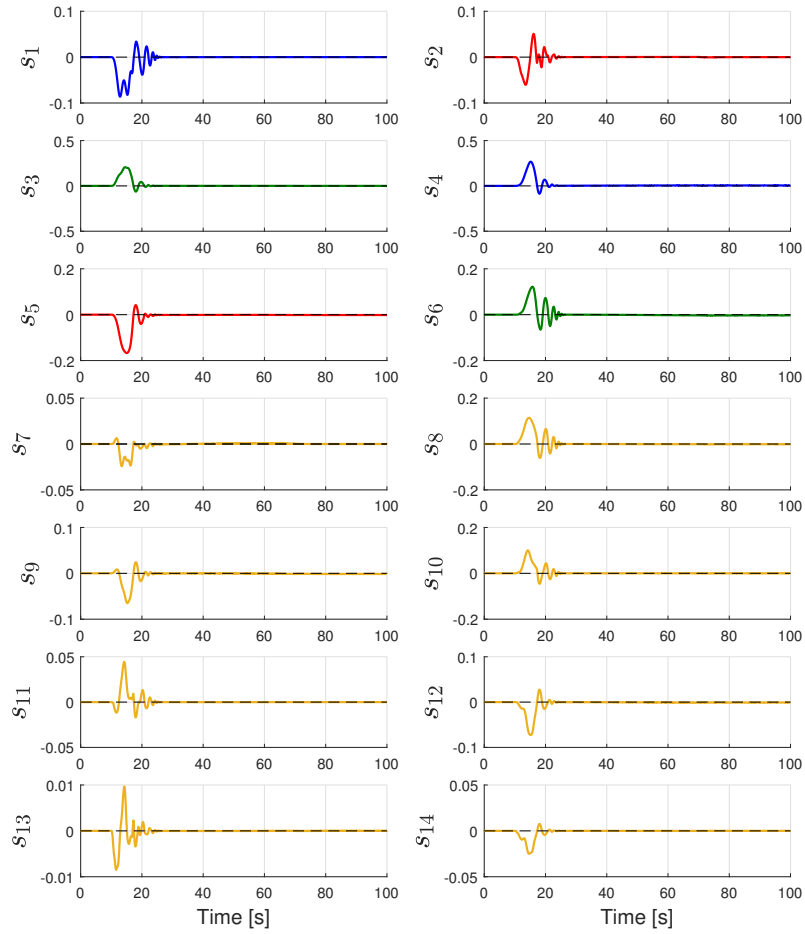


Figure 4.14: NRAC - The time evolution of the error variable $s = \tilde{\zeta} + \Lambda\tilde{y}$. s_i, i=1...14, is the i'th variable of the vector s. The plots in yellow are the errors associated with the joints while the blue, red and green are the errors associated with the x,y and z position/orientation respectively.

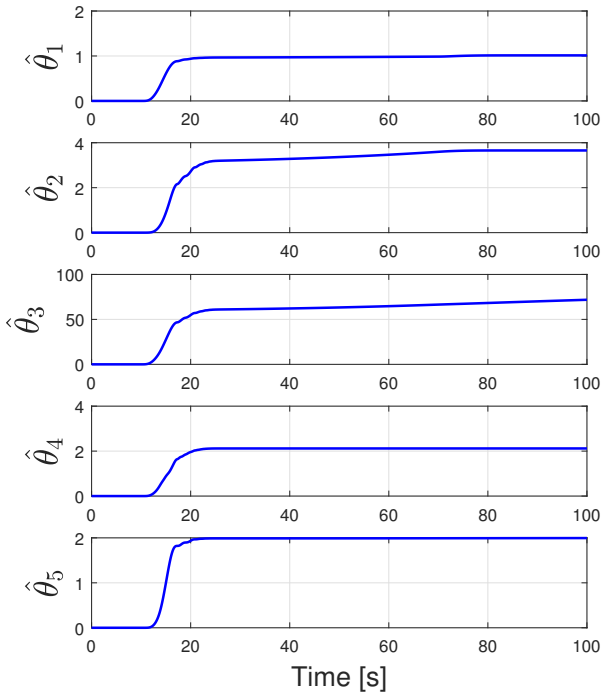


Figure 4.15: NRAC - The time evolution of the estimates $\hat{\theta}_i$, $i = 1...5$.

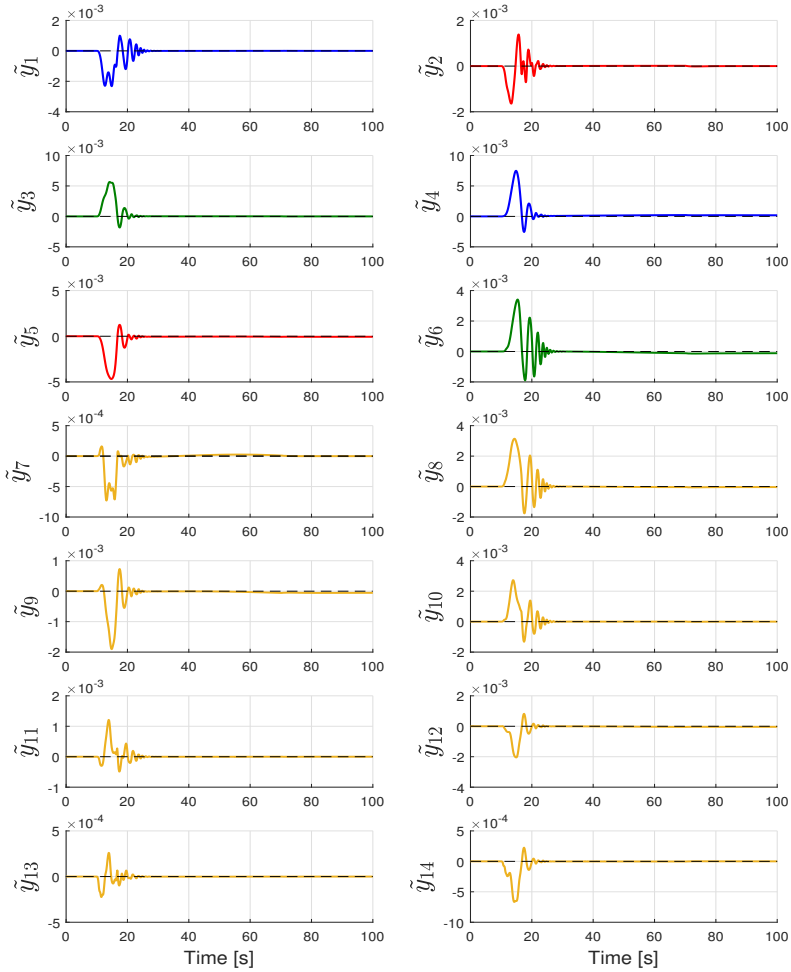


Figure 4.16: NRAC - The time evolution of the error variable $\tilde{y}$. $\tilde{y}_i$, i=1...14, is the i'th variable of the vector $\tilde{y}$. The plots in yellow are the errors associated with the joints while the blue, red and green are the errors associated with the x,y and z position/orientation respectively.

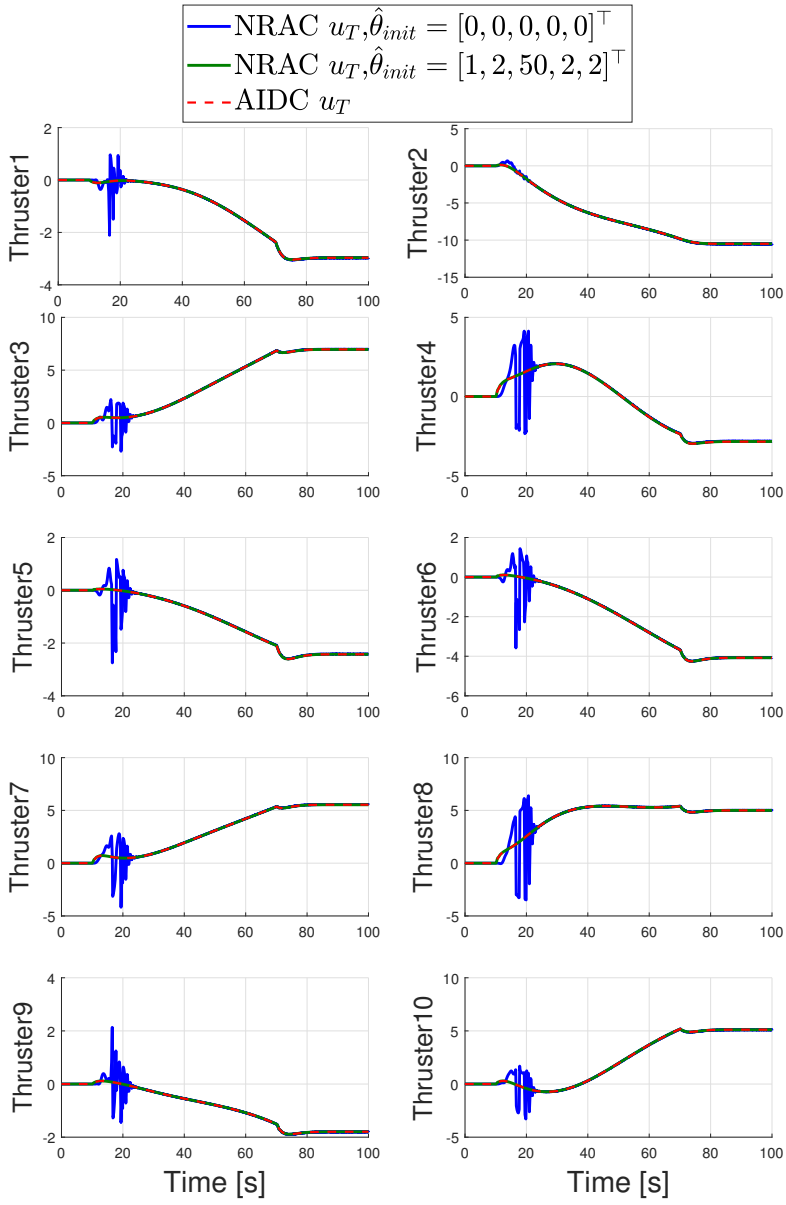


Figure 4.17: NRAC - The time evolution of applied joint torque u_J from the NRAC with initial values of the estimates $\hat{\theta} = [0,0,0,0,0]^\top$ and $\hat{\theta} = [1,2,50,2,2]^\top$. Applied joint torque u_J from the AIDC is included in the plot.

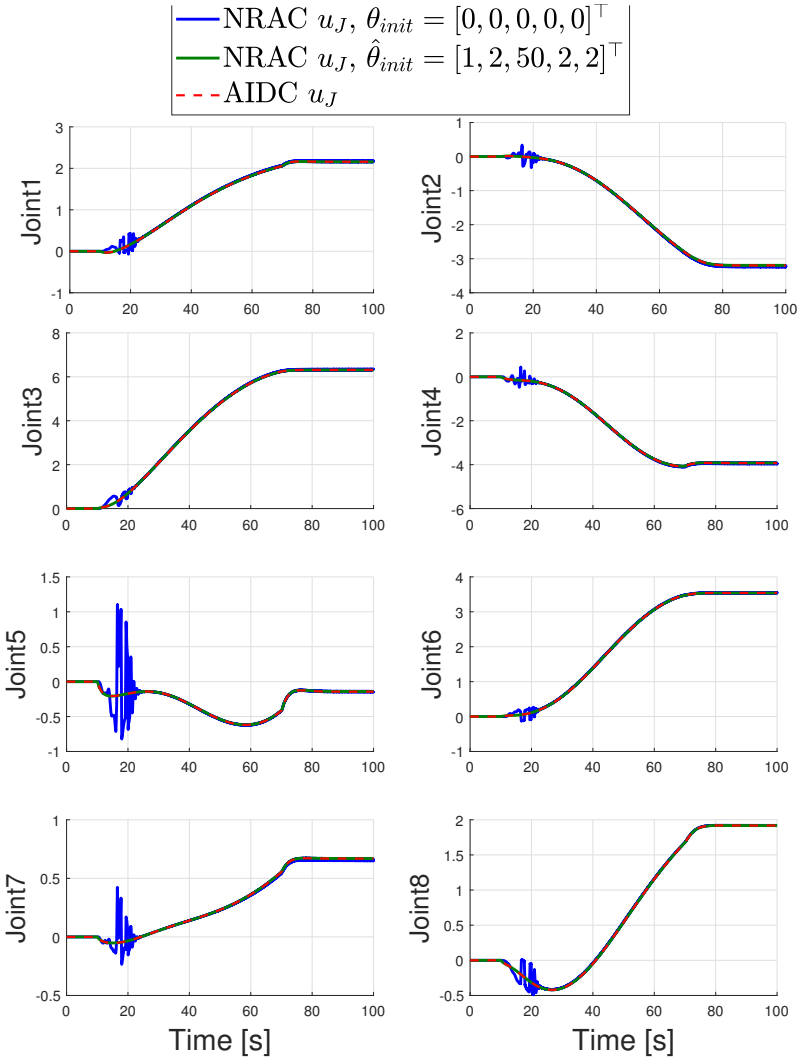


Figure 4.18: NRAC - The time evolution of thruster control input u_T from the NRAC with initial values of the estimates $\hat{\theta} = [0,0,0,0,0]^\top$ and $\hat{\theta} = [1,2,50,2,2]^\top$. Applied joint torque u_T from the AIDC is included in the plot.

Chapter 5

Interaction control

In this chapter, interaction control for USMs is presented. After an introduction to interaction control, the task that will be considered is presented. Then, two interaction control schemes, namely impedance control (IC) and PI force control with impedance control (FC) is presented. Finally, simulation results with the interaction controllers are presented and discussed.

5.1 Interaction control background

For a USM to perform a completely autonomous mission, the force exchanged between the USM and the environment should be considered (Antonelli; 2014). The interaction can be described by the contact force at the end-effector of the USM (Siciliano et al.; 2008). High values of contact forces are in general undesirable as it can damage the USM and/or the manipulated object. Examples of relevant tasks for the USM are turning valves on subsea panels or plugging/unplugging connectors (Palomeras et al.; 2014). The contact force is most conveniently described in task space. In interaction with the environment, the paths that can be followed by the USM are constrained by the environment. Motion controllers only give successful execution of interaction tasks if the task is accurately planned (Siciliano et al.; 2008). There are usually limitations of

knowledge of the USM model and environment. This leads to planning errors which cause contact forces. These contact forces leads to deviations from the end-effector trajectory and potentially build up of interaction force (Siciliano et al.; 2008).

The choice of interaction controller depends on the task to be performed, the required control accuracy and availability of force measurements (Siciliano et al.; 2008). If force measurements are available, these can be used in the controller. Force/moment sensor readings are however usually corrupted by noise (Antonelli; 2018). Interaction control strategies can be grouped into two categories, namely *indirect force control* and *direct force control* (Siciliano et al.; 2008). *Indirect force control* achieves force control through motion control without an explicit closure of a force feedback loop (Siciliano et al.; 2008) while *direct force control* contains a force feedback loop so that the contact force can be controlled to a desired value (Siciliano et al.; 2008). Some interaction control schemes are introduced in the following.

Stiffness control is an indirect force control strategy which consists of assigning a desired stiffness at the end-effector, and the apparent stiffness at the end effector is controlled (Salisbury; 1980). It is obtained by using a suitable position control scheme when the end-effector is in contact with the environment (Antonelli; 2014). Impedance control is also an indirect force control approach in which the goal is to achieve a desired mechanical impedance at the end-effector (Hogan; 1984; Siciliano et al.; 2008). Hybrid force/position control allows for position and force constraints to be satisfied simultaneously (Raibert and Craig; 1981). With hybrid force/position control the end-effector motion and contact forces are split up into two decoupled subproblems (Siciliano and Khatib; 2007). While these control schemes requires detailed knowledge of the geometric features of the environment, a parallel force control force/position controller overcomes this limitation (Antonelli; 2014).

The controllers presented in depth in this section are based on the work in (Cataldi and Antonelli; 2015). The article presents two control schemes for UVMS interaction with the environment. In the article, an impedance controller is applied when the task is to a turn a valve task and a PI force controller with impedance control is applied to a push a button task. In the article the controllers are applied to a UVMS. This chapter presents how both controllers can be applied to a USM performing a turn a valve task.

The impedance controller presented in this section is an *indirect force control* scheme, while the PI force controller presented in this section is a *direct force control* scheme.

The article (Chiaverini et al.; 1999) and the book (Siciliano and Khatib; 2007) contains overviews of different interaction control schemes for robot manipulators. The book (Antonelli; 2014) gives an overview of interaction control of UVMSs. The authors in (Ferretti et al.; 1997) present an explicit force control strategy with an outer force control loop. The idea is to apply force without affecting the trajectory tracking. In (Antonelli; 2014) two different versions of the scheme presented in (Ferretti et al.; 1997) is presented. A hybrid impedance control approach that achieves force/position control of redundant manipulators is presented in (Oh et al.; 1998). In (Cui and Yuh; 2003) a force control scheme for UVMS based on a non-regressor-based adaptive controller similar to that in section 4.4 is presented. The controller in (Cui and Yuh; 2003) combines adaptive impedance control with hybrid force/position control with fuzzy switching.

Finally, recall the implemented model for the USM in contact with the environment:

$$M(q)\dot{\zeta} + C(q,\zeta)\zeta + D(q,\zeta)\zeta + g(q,\eta) = \tau - J_{\omega}^{\top}(q, R_B^I)h_{ee} \tag{5.1}$$

5.2 Task description - turn a valve

The objective of the USM is to turn a valve by 45 degrees. Figure 5.1 shows a visual example of a manipulator and a valve. A gripper is mounted on the head-link of the USM. It is assumed that the end-effector has already gripped the valve and that the end-effector frame Σ_{ee} is perfectly aligned with the object frame Σ_{obj} at the beginning of the task. It is also assumed that the end-effector frame Σ_{ee} and the object frame Σ_{obj} has no rotation with respect to the inertial frame initially, i.e $\eta_{2,ee} = [0,0,0]^{\top}$ and $\eta_{2,obj} = [0,0,0]^{\top}$.

The valve rotates about the axis x_{obj} as shown in figure 5.1 which is aligned with the end-effector x-axis x_{ee}. The rotation of the valve expressed in world frame is therefore equal to the rotation of the end-effector frame expressed in the inertial frame. Hence, the task is completed when the end-effector has reached a rotation of

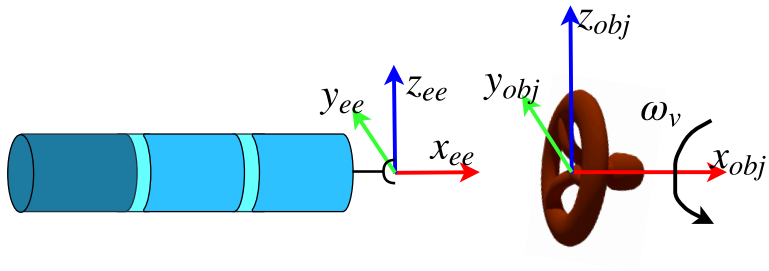


Figure 5.1: Visualization of the task of turning a valve. The object frame Σ_{obj} and the end-effector frame Σ_{ee} is included in the figure.

45 degrees.

An important aspect of interaction control is to avoid too much contact force and hence avoid stress on the USM as well as the valve (Siciliano et al.; 2008). In a real situation it might occur that external disturbances such as current cause the USM to lose contact with the valve (Antonelli; 2018). In the following, loss of contact is not taken account and it is assumed that the USM has gripped the valve throughout the trajectory.

5.3 Impedance control

The goal of impedance control is to achieve a desired dynamic behaviour for the end-effector (Siciliano and Khatib; 2007). The impedance control scheme implemented for the USM to turn a valve is based on (Cataldi and Antonelli; 2015). The objective of the controller is to obtain a desired end-effector impedance (Cataldi and Antonelli; 2015). It is an *indirect force control* scheme as the interaction force is not directly regulated.

The control structure for impedance control is shown in figure 5.2. The control structure consists of an internal and an external loop as shown in figure 5.2. The internal loop is the same as the loop marked in gray in figure 3.4. The external loop controls the interaction force with the environment while the internal loop contains the inverse kinamatics, dynamic control and thrust allocation.

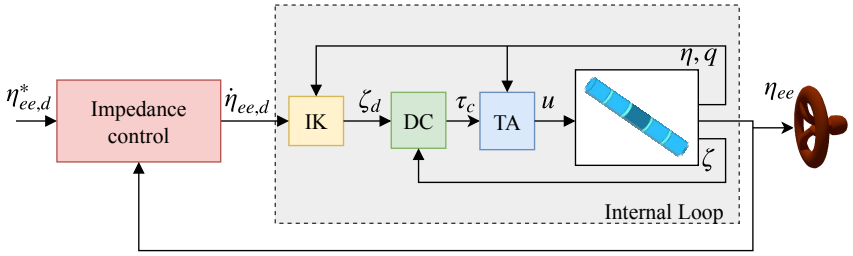


Figure 5.2: The control framework for impedance control. The internal loop is the same as the area marked in gray in figure 3.4.

5.3.1 Control design

The control law used for impedance control is (Cataldi and Antonelli; 2015):

$$\eta_{ee,d} = K_S \tilde{\eta}_{ee} + K_D \tilde{v}_{ee} + K_I \int_{t_0}^{t} \tilde{\eta}_{ee}(\sigma) d\sigma, \tag{5.2}$$

which is differentiated with respect to time to give the input to the internal loop $\dot{\eta}_{ee,d}$. The error variables $\tilde{\eta}_{ee}$ is defined as:

$$\tilde{\eta}_{ee} = \begin{bmatrix} \eta^*_{ee,1,d} - \eta_{ee,1} \\ \tilde{\epsilon} \end{bmatrix}, \tag{5.3}$$

where $\tilde{\epsilon}$ was defined in (4.5) with p_d found from converting the Euler angles of $\eta^*_{ee,d}$ to quaternions. The error variable $\tilde{v}_{ee}$ is defined as:

$$\tilde{v}_{ee} = \begin{bmatrix} \dot{\eta}^*_{ee,1,d} - \dot{\eta}_{ee,1} \\ \omega_{ee,d} - \omega_{ee} \end{bmatrix}, \tag{5.4}$$

where ω_{ee} is the angular velocity of the end-effector expressed in the inertial frame and $\dot{\eta}^*_{ee,1,d}$ is the time derivative of the first three entries of the input to the impedance controller as shown in figure 5.2.

K_S, K_D and $K_I \in R^{6\times6}$ are the stiffness, damping and integral gain matrices re-

spectively. The integral term in the control law (5.2) is added to remove steady state error. In addition, consider a controller without the integral term with the desired objective to turn a valve by a certain amount at a constant velocity. The valve will begin to turn when $\eta_{ee,d}$ becomes large enough so that the applied force overcomes the friction required to begin to turn the valve. If K_S and K_D are too small, the valve may never turn. When the valve does not turn, the error variables $\tilde{\eta}_{ee}$ and $\tilde{v}_{ee}$ reach a constant value after some time. If these values of the error variables does not make the applied force large enough to overcome the friction, the valve will never turn. When the integral term is added, the applied force will increase and make the applied force large enough, and overcome the problem. If the valve is stuck, this will however cause a problem with a large build up of force that can damage the USM, the valve or both.

5.4 PI force control with impedance control

Consider the case in which a force sensor is mounted on the end-effector in order to measure contact forces and moments. Assume measurements of the forces and moments applied by the end-effector in the inertial frame h_{ee} are available. While desired end-effector positions and orientations, $\eta^*_{ee,d}$, is the input to the impedance controller, the input to the PI force controller is the desired force to be exerted by the end-effector $h_{ee,d}$ (Cataldi and Antonelli; 2015).

5.4.1 Control design

The control structure is shown in figure 3.4. The control structure for the PI force controller also consists of an internal loop with inverse kinematics, dynamic motion control and thrust allocation and an external loop charged with controlling the interaction force with the environment as shown in figure 5.3. The Proportional Integral (PI) action is used to stabilize the force error (Cataldi and Antonelli (2015)):

$$\eta^*_{ee,d} = K_P \tilde{h}_{ee} + K_I \int_{t_0}^{t} \tilde{h}_{ee}(\sigma) d\sigma \tag{5.5}$$

where $\tilde{h}_{ee} = h_{ee,d} - h_{ee}$. $K_P \in R^{6\times6}$ and $K_I \in R^{6\times6}$ are the proportional and integral gain matrices respectively.

h_{ee} are the forces and moments measured by the force sensor and is usually characterized by strong noise (Cataldi and Antonelli; 2015). Its derivative $\dot{h}_{ee}$ is therefore usually useless and the derivative is therefore not included in the force controller. Damping is instead provided by the impedance controller presented in the previous section.

The loop containing the PI force controller is added outside the impedance controller from section 5.3 as shown in figure 5.3

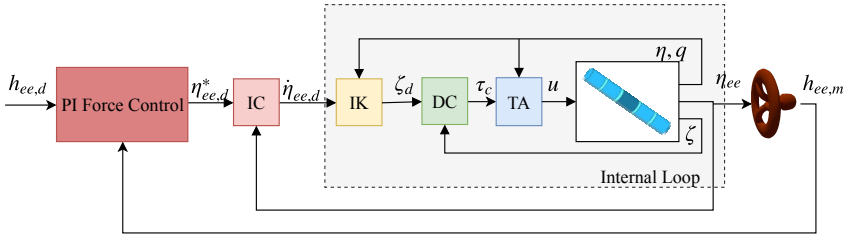


Figure 5.3: The control framework for PI force control with impedance control. The internal loop is the same as the area marked in gray in figure 3.4 and the IC block is the same as the impedance control block in figure 5.2.

Consider the situation where the valve is stuck and not possible to turn. With only the impedance controller, the input is end-effector position and orientation. Therefore, the longer the valve is not turning, the more force will be applied by the end-effector because of the integral term and the increasing $\tilde{\eta}_{ee}$. Eventually, with too much force, either the valve, the USM or both might break. The input to the force controller is however the desired exerted force. As the PI force controller controls the force applied by the end-effector and not the position and orientation of the end-effector. The force will therefore not build up to a value large enough to damage the USM or the valve when the valve is stuck if the input $h_{ee,d}$ is chosen appropriately.

5.5 Simulations and Discussion

Simulations have been performed to test the performance of the interaction controllers introduced above. The simulation results are presented in this section.

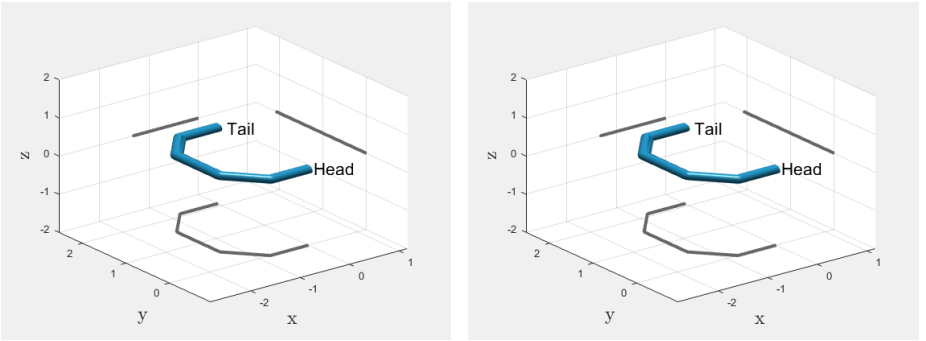
The interaction forces' effect on the USM were modeled by equation (5.1). The task considered in this section is to turn a valve *at least* 45 degrees. One can think of this as a closed valve that is to be opened by the USM. When the valve has rotated 45 degrees, the valve is opened, but may still rotate further. It is assumed that the end-effector has already gripped the valve at the beginning of the simulation so that the end-effector frame Σ_{ee} and the object frame Σ_{obj} is coinciding.

The interaction moment exerted by the end-effector from the valve is modeled by (2.49). The inertia about the x-axis is chosen as $J_v = 3$ and the viscosity coefficient is chosen as k_μ for the simulations. It is assumed that the angular velocity of the valve about the x-axis of the inertial frame equals the angular velocity of the of the end-effector about the x-axis of the inertial frame, $\omega_{v,x} = \omega_{ee,x}$ because the end-effector has gripped the valve. The linear interaction forces exerted by the end-effector is modeled by equation (2.50) with k=20. The position of the valve is in the origin of the inertial frame $[0, 0, 0]^\top$.

The objective is to achieve rotation of the valve, and hence the end-effector frame, about the x-axis of the inertial frame. The initial configuration and final configuration of the USM when turning a valve is shown in figure 5.4. Joint 9 of the USM is x-revolute and the the valve rotation can therefore be achieved by rotating only the last joint. The final configuration (figure 5.4b) therefore looks similar to the initial configuration (figure 5.4a), but with rotation of the end-effector. For the simulations, a weighted pseudoinverse (3.4) is implemented with the weight matrix W:

$$W = blockdiag\{I_3, I_3, I_8, 0.1\}. \tag{5.6}$$

This is done to prioritize rotation of the last link (end-effector). When minimizing the value of $E = \frac{1}{2}\zeta_d W \zeta_d$, this W prioritizes velocity of the last joint. The USM therefore remains almost unmoving except for the last link. In this case, the thrusters have to compensate for the force applied by the end-effector to keep the USM in position.



(a) Initial configuration.

(b) Final configuration.

Figure 5.4: Initial and final configuration of the USM in space for turning a valve. Initial end-effector position and orientation is $\eta = [0, 0, 0, 0, 0, 0]^\top$ and final end-effector position and orientation is $\eta = [0, 0, 0, \frac{\pi}{4}, 0, 0]^\top$ for turning a valve $\frac{\pi}{4}$ radians.

Figure 5.5 shows an example of the interaction forces and moments experienced by the end-effector during the task of turning the valve with impedance control. The force to the end-effector in x-direction work in the negative x-direction which is expected. The moment experienced by the end-effector around the x-axis work in the opposite direction of the motion which is also as expected. The forces and moments are smoothed by a low pass filter in the simulations to avoid too sudden changes in the forces/moments and hence avoid simulation problems. The force in the x-direction is a function of how much the end-effector moves beyond the position of the valve (2.50). In the simulations, the end-effector only moves past the valve by a tiny amount and the force is therefore very small. The moment experienced by the end-effector has a maximum absolute value of $6Nm$ with the values chosen for J_v and k_μ. Comparing this to the plots of joint torques in the previous chapter (figure 4.18 and 4.10), if can be seen that $6Nm$ is about what is applied to joint 3 in the simulations in the previous chapter. This is quite small for turning a valve, but will be considered sufficient for testing of the controllers.

With an external moment working in the opposite direction of the rotation, intuitively more force from the thrusters and more joint torque is required than without

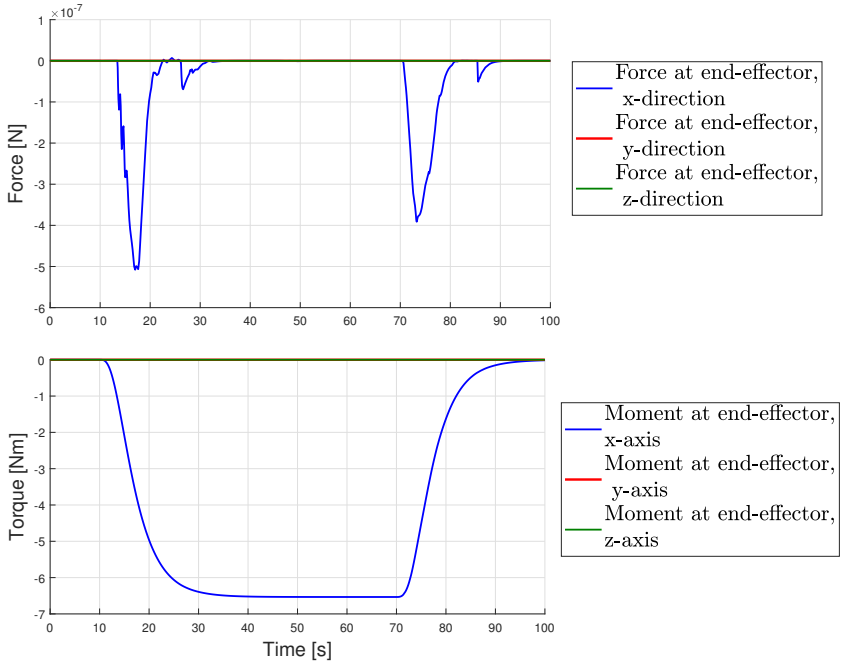


Figure 5.5: The time evolution of the forces and torques applied to the end-effector by the valve expressed in the inertial frame when turning the valve. The figure shows the force from simulations with impedance control, but the results are similar for impedance and force control.

the external torque. Figure 5.6 shows applied thrust to the USM when turning a valve and when only rotating the end-effector. Figure 5.7 shows the applied joint torque in the same two cases. It can be seen from these figures that the absolute value of thrust and joint torque are larger for turning the valve than for only turning the end-effector which is expected. These plots show the thrust and joint torque using the impedance controller, but the results are similar for both controllers presented in this section.

The adaptive inverse dynamics controller gives great results for motion control as was shown in section 4.5.1. The controller is tested for the turn a valve operation. The resulting end-effector position and orientation is shown in figure 5.8. It is evident that using only the adaptive inverse dynamics controller is not sufficient for the turn a valve operation.

The interaction controllers presented in this chapter are based on an internal loop with inverse kinematics, dynamic control and thrust allocation. The adaptive controller is used as the motion controller for both the following simulations.

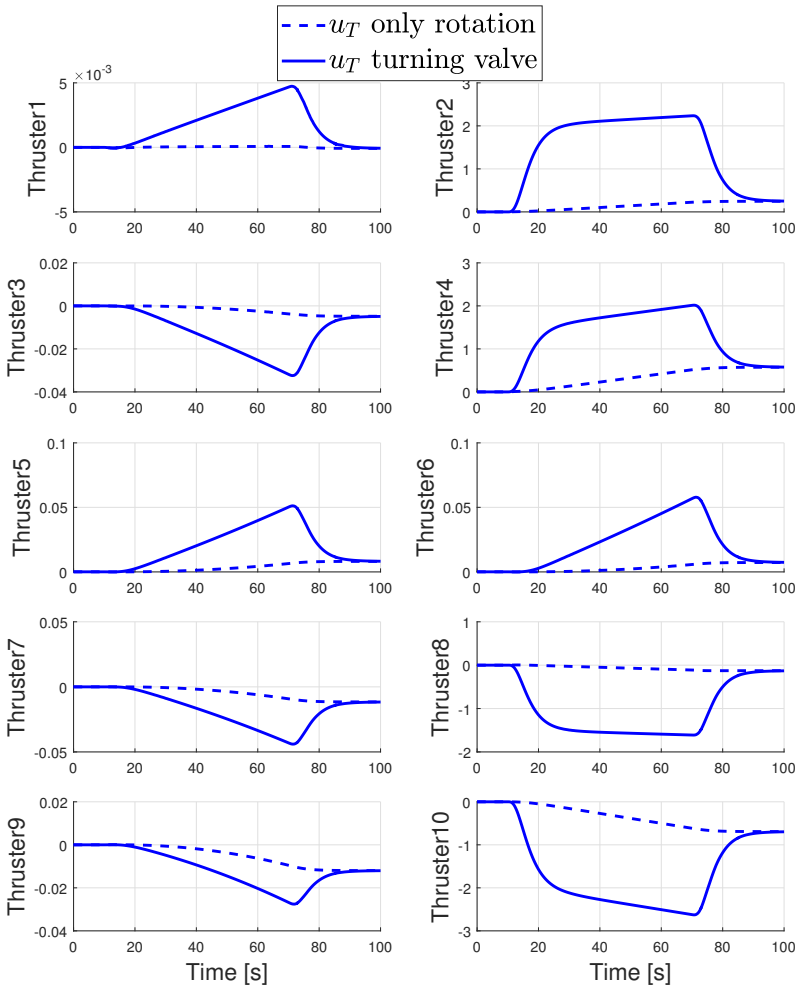


Figure 5.6: The time evolution of thruster control input u_T on each thruster with and without external force when rotation the end-effector.

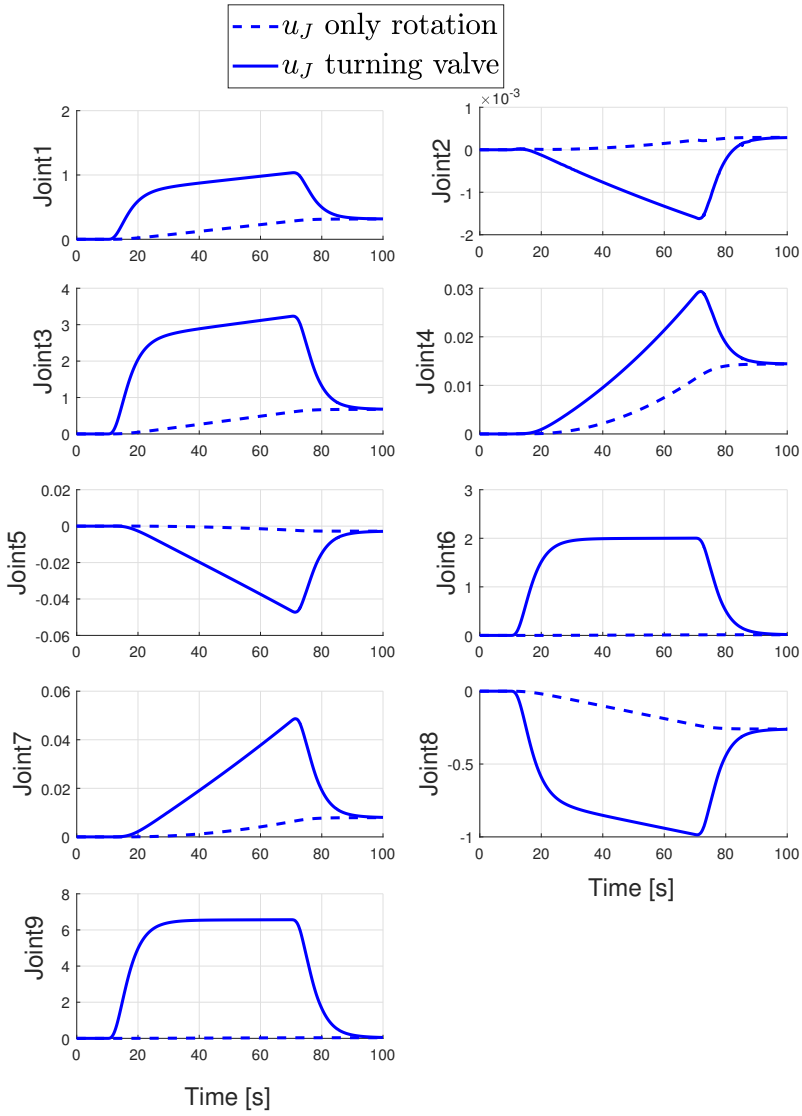


Figure 5.7: The time evolution of applied joint torque u_J on each joint with and without external force when rotating the end-effector.

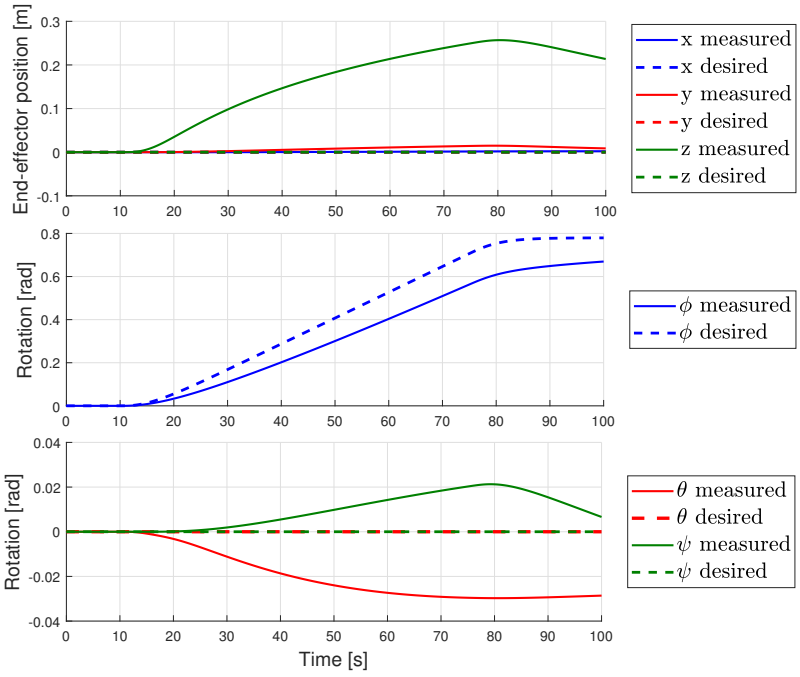


Figure 5.8: The time evolution of η_{ee} with $\eta_{ee,d}$ when only the AIDC is used to turn a valve. The x,y and z positions of the end-effector, $\eta_{1,ee}$, are collected in the first subplot. The roll ϕ in the second subplot and pitch θ and yaw ψ are collected in the third subplot.

5.5.1 Impedance control

The input to the impedance control law are the desired end-effector positions and orientations. The desired the end-effector trajectory is a rotation of $\frac{\pi}{4}$ rad about the x-axis of the inertial frame. The input is therefore a trajectory starting in $[0, 0, 0, 0, 0, 0]^\top$ and ending in $[0, 0, 0, \frac{\pi}{4}, 0, 0]^\top$ such that the valve will be rotated $\frac{\pi}{4}$ radians. The trajectory is generated through the reference trajectory generation block presented in section 3.4.

The solver used for simulations is ode15s with relative tolerance 10^{-3}. The stiffness, damping and integral gain matrices are chosen to give good simulation results as:

$$\begin{aligned} K_S &= diag\{30, 30, 30, 6, 6, 6\}, \\ K_D &= diag\{1, 1, 1, 0.2, 0.2, 0.2\}, \\ K_I &= diag\{4, 4, 4, 3, 3, 3\}. \end{aligned} \tag{5.7}$$

The resulting time evolution of valve angles is shown in figure 5.9. The figure shows that the impedance controller achieves the objective of rotating the valve by $\frac{\pi}{4}$ radians. The forces and moments experiences by the end-effector when turning the valve is shown in figure (5.5). The forces and moments exerted by the end-effector are the equal, but oposite forces of those in figure 5.5.

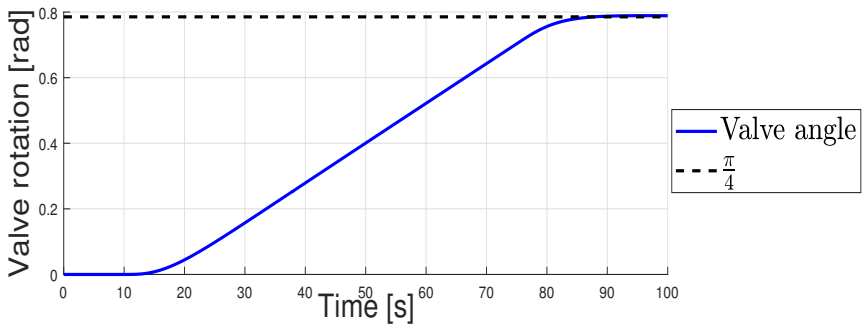


Figure 5.9: IC - The time evolution of the valve angle with a line at $\frac{\pi}{4}$ radians for reference.

The desired end-effector positions and orientations $\eta^*_{ee,d}$ are closely traced by the end-effector as seen in figure 5.10. The x-rotation is presented in a separate subplot so that the errors in the rotations about the y and z-axis can be noticed in the third plot. The roll of the end-effector follows its desired trajectory closely. It can easily be seen by comparing figure 5.8 and 5.10 that adding an external impedance control loop to the internal loop gives much better results than with only the AIDC. In the simulations with the impedance controller there is some error in order of magnitude 10^{-5} (compared to 10^{-1} for AIDC) in the z-position as seen from the first subplot in figure 5.10. The error does however seem to converge to zero from the plot. After about 72 seconds, the force is no longer constant and the absolute value of the force decreases after this point. This causes the changes in z-direction and pitch and yaw at about 72 seconds. The errors in the rotation about the y- and z-axis are in the order of magnitude 10^{-5} (compared to 10^{-2} for AIDC) and the error in rotation also seems to converge to zero in the simulations.

For the impedance controller the desired force is not specified, only the desired end-effector positions and rotations. The goal is to achieve a desired impedance at the end-effector. It is not necessary to measure force/moment at the end-effector, and the impedance controller may be a good option if measurements of force/moment at the end-effector are not available. As the integrator term is included, the force at the end-effector will eventually become large enough to rotate the valve. If however the valve is stuck, for example rusted, the force applied to the valve may become too large and potentially brake either the valve or the USM/end-effector or both.

The impedance controller gives satisfactory results in these simulations, but is a simple control algorithm. As the control setup with the impedance controller consists of an external loop and an internal loop that is equal to the motion control loop (figure 5.2). More sophisticated interaction control strategies may be easily adopted (Cataldi and Antonelli; 2015).

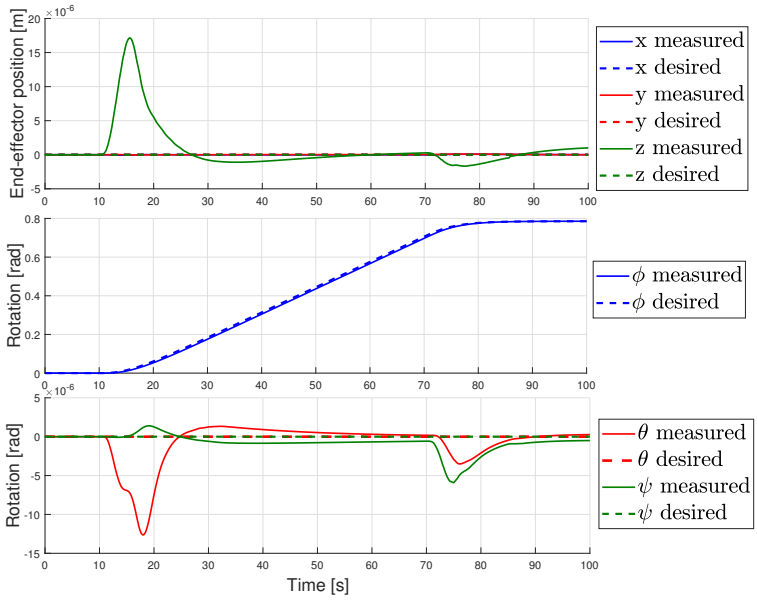


Figure 5.10: IC - The time evolution of η_{ee} with $\eta^*_{ee,d}$ when turning a valve. The x,y and z positions of the end-effector, $\eta_{1,ee}$, are collected in the first subplot. The roll ϕ in the second subplot and pitch θ and yaw ψ are collected in the third subplot.

5.5.2 PI force control with impedance control

In the simulations with the PI force controller with impedance control it is assumed that measurements of the interaction forces and moments at the end-effector in the inertial frame are available. The PI force controller with impedance control presented in section 5.4 is implemented. In practice, this means that a PI force control block is added in front of the impedance control block from the previous section. Figure 5.3 shows an overview of the control framework. The solver used for simulations is ode15s with relative tolerance 10^{-3}. The impedance control gains are chosen as in (5.7). The gains in the PI force controller are chosen to give satisfactory results as:

$$\begin{aligned} K_P &= diag\{0.77, 0.77, 0.77, 0.15, 0.15, 0.15\}, \text{ and} \\ K_I &= diag\{0.48, 0.48, 0.48, 0.29, 0.29, 0.29\}. \end{aligned} \tag{5.8}$$

The force/moment errors $\tilde{h}_{ee}$ in y and z-direction and about the y- and z-axis are always zero because the input trajectory is zero in these directions and no external force is modeled in these directions. The entries of the gain matrices corresponding to these states may therefore be any value.

The input to the controller is the desired forces and torques $h_{ee,d}$ to be applied by the end-effector. The desired forces and torques are generated to give a similar trajectory of that of only the impedance controller. The desired forces and torques are therefore generated from the desired end-effector position $\eta^*_{ee,d}$ from the previous section. This gives a smooth force trajectory that avoids problems in the simulations.

For the simulations with PI force control with impedance control, the objective is defined as to rotate the valve *at least* 45 degrees. It is assumed that the valve is opened when it has reached 45 degrees, but can still be rotated by some amount. The desired force trajectory is therefore such that it increases smoothly until it reaches a constant force. Then, when the end-effector has reached the desired rotation, the desired force goes smoothly down to zero. Therefore, it keeps rotating a while after $\frac{\pi}{4}$ radians.

Figure 5.11 shows that the objective of rotating the valve at least $\frac{\pi}{4}$ radians is achieved. It can be seen that once the valve has rotated $\frac{\pi}{4}$ radians, the rotation rate slows down and reach a constant value of about 0.85 radians.

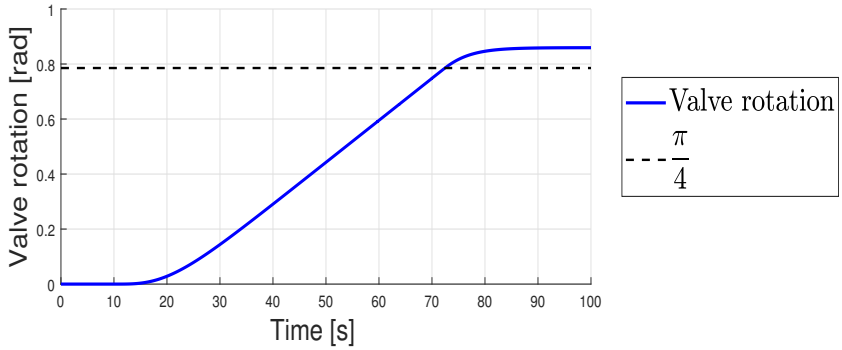


Figure 5.11: FC - The time evolution of the valve angle with a line at $\frac{\pi}{4}$ radians for reference.

The force in x-direction of the inertial frame applied by the end-effector $h_{ee,m}(1)$ together with the desired force in x-direction of the inertial frame at the end effector $h_{ee,d}(1)$ is shown in the first plot of figure 5.12. It can be seen that the error in the force in the x-direction is very small, of magnitude $10^{-5}N$. This is because the end-effector barely moves past the position of the valve in x-direction. The second plot of figure 5.12 shows the moment exerted by the end effector about the x-axis of the inertial frame $h_{ee,m}(4)$ with the desired torque about the x-axis of the inertial frame $h_{ee,d}(4)$. The applied torque follows its desired value closely throughout the trajectory.

Figure 5.13 shows the end-effector positions and orientations η_{ee}. The end-effector position and orientation is not defined by the user, but the desired values plotted in the figure are the inputs to the IC block, $\eta^*_{ee,d}$ (see fig 5.3). The errors in positions and orientations are as can be seen from figure 5.13 small and the desired rotation about the x-axis, ϕ, is followed closely. $\eta^*_{ee,d}$ is the output of the PI-force controller and it is therefore interesting to look at the desired values. The small shift of desired x-positions comes from the small errors in force in x-direction (5.12). The integral term in the controller (5.12) gives this negative desired x-position.

The PI force controller with impedance control gives satisfactory results in simulations. The advantage of the PI force controller with impedance control over the IC is

that it does not cause build up of force as can happen for the impedance controller when the valve is stuck.

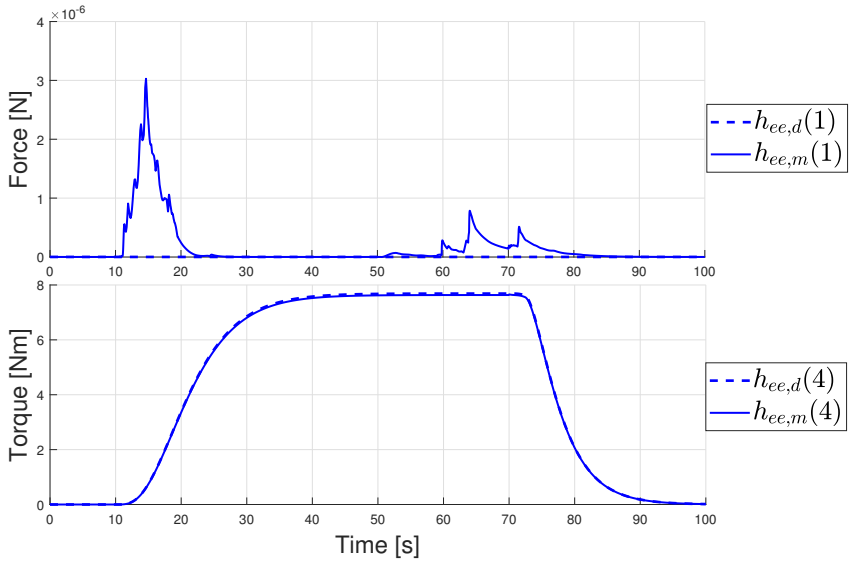


Figure 5.12: FC - The time evolution of the force in x-direction exerted by the end-effector $h_{ee,m}(1)$ with the desired value $h_{ee,d}(1)$ and the torque about the x-axis exerted by the end-effector $h_{ee,m}(4)$ with the desired value $h_{ee,d}(4)$.

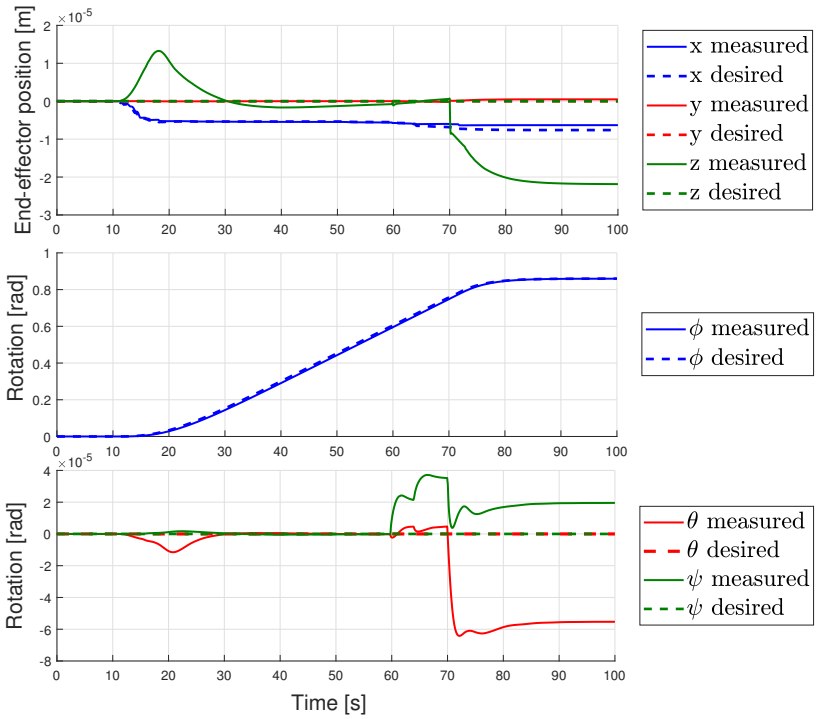


Figure 5.13: FC - The time evolution of η_{ee} with $\eta^*_{ee,d}$ when turning a valve. The x,y and z positions of the end-effector, $\eta_{1,ee}$, are collected in the first subplot. The roll ϕ in the second subplot and pitch θ and yaw ψ are collected in the third subplot.

Chapter 6

Conclusions and future work

6.1 Conclusions

This thesis has presented dynamic motion control approaches as well as interaction control approaches for underwater swimming manipulators (USMs). The dynamic motion control approaches presented in the thesis are adaptive inverse dynamics control (AIDC), the super-twisting algorithm with adaptive gains (STA) and non-regressor-based adaptive control (NRAC). The dynamic motion control algorithms have been implemented with a USM simulator. The stability properties of the controllers have been analyzed and their expected behaviour has been compared to how they behave in simulations. The interaction control approaches presented in the thesis are an impedance controller (IC) and a PI force controller with impedance control (FC). The interaction controllers have been implemented in a simulator that includes interaction force and the controllers have been compared.

The AIDC requires knowledge of the model structure. It does not guarantee asymptotic stability, but guarantees bounded states. The super-twisting algorithm is a discontinuous controller and gives chattering in the control input. The implemented STA requires some model knowledge, but gives finite time convergence of the sliding variable s even with disturbances. The NRAC requires only a limited amount of

knowledge of the system and is attractive for its simplicity and computational efficiency. Asymptotic convergence of the error variable s has been shown for the NRAC.

All the implemented dynamic motion controllers give good tracking of a desired end-effector trajectory in simulations. The differences between the controllers are most evident in the control input. While the AIDC gives a smooth control input, the control input from the STA is characterized by chattering that may damage the USM. The NRAC gives rapid oscillations initially in control input, but the oscillations can be reduced by choosing nonzero initial values of the estimates $\hat{\theta}$.

Both the interaction controllers perform well in the simulations. The impedance controller gives good tracking of the desired end-effector position and orientation. With the FC, the force exerted by the end-effector follows the desired force trajectory well. The IC can be used when no force measurements are present, while the FC requires a force sensor. Both the interaction controllers are added as an external loop and more sophisticated interaction control schemes may therefore easily be adopted.

6.2 Recommendations for further work

A relevant continuation of the work in this thesis is to test the controllers on real USMs. Before applying the controllers to a real USM, the controllers can be tested in a simulator that includes modeling of thruster and joint dynamics and modeling of saturation and delays to give more realistic simulations.

The pseudoinverse used for differential inverse kinematics used in the control framework is sufficient for initial simulations and concept verification. A recommendation for further work is therefore to implement more advanced kinematic control algorithms that provides for example singularity avoidance. More advanced thrust allocation algorithms could be implemented such as algorithms in (Fossen et al.; 2009). It is also interesting to look at the power consumption of the USM for different control approaches.

In the thesis it was shown that the AIDC where only the drag forces are unknown gives good results in simulations. A complete AIDC where all unknown parameters of the model is estimated should be implemented as is done for UVMSs in (Antonelli and

Chiaverini; 1998). Other motion controllers can also be implemented for USMs and compared to the results in this thesis. Examples of suggested control strategies are strategies based on neural networks and controllers that are not divided into separate kinematic and dynamic control parts.

Interaction control should be investigated further. More complex control strategies should be implemented and compared to the simple interaction control strategies in this thesis. To make the simulations with the interaction controllers more realistic, noise can be added to the force measurement. In addition, the simulator can be extended with more realistic modeling of interaction forces and constraints. Loss of contact while performing the task should also be taken into account.

Appendix A

Stability and definitions

The stability of the controllers presented in this project is analyzed. The definitions and theorems needed for the stability analyses are presented here.

A.1 Positive definiteness

The following is taken from (Khalil; 2002). A function $V(x)$ is said to be positive definite if $V(x)$ satisfies $V(0) = 0$ and $V(x) > 0$ for all $x \neq 0$. If the function satisfies $V(x) \geq 0$ for all $x \neq 0$ it is said to be positive semidefinite. The function $V(x)$ is negative definite or positive semidefinite if $-V(x)$ is positive definite or negative definite respectively. If $V(x) = x^\top Px$ is positive definite (or positive semidefinite), then the matrix P is positive definite (or positive semidefinite). Equally the matrix P is positive definite (or positive semidefinite) if all the eigenvalues of P are positive (or nonnegative).

A.2 Stability

Consider the system

$$\dot{x} = f(x) \tag{A.1}$$

where $f : D \to R^n$ is a locally Lipschitz map from a domain $D \subset R^n$ into R^n (Khalil (2002)). First, define stability and asymptotic stability.

Definition A.1 (Definition 4.1, Khalil (2002)) *The equilibrium point $x = 0$ of (A.1) is*

- *stable if, for each $\epsilon > 0$, there exists a $\delta = \delta(\epsilon) > 0$ such that*

$$||x(0)|| < \delta \quad \Rightarrow ||x(t)|| < \epsilon, \quad \forall t \geq 0$$

- *unstable if it is not stable*

- *asymptotically stable if it is stable and δ can be chosen such that*

$$||x(0)|| < \delta \Rightarrow \lim_{t \to \infty} x(t) = 0$$

Lyapunov's direct method is a method for establishing stability and asymptotic stability. Lyapunov's direct method is given in the following theorem.

Theorem A.1 (Lyapunov's Direct method, Theorem A.1 Fossen (2011)) *Let x_e be the equilibrium point of $\dot{x} = f(x), x(0) = x_0$ and assume that $f(x)$ is locally Lipschitz in x. Let $V : R^n \to R_+$ be a continuous differentiable function V(x) satisfying:*

$$\textit{(i)}\ V(x) > 0 \ \textit{(positive definite) and}\ V(0) = 0 \tag{A.2}$$

$$\textit{(ii)}\ \dot{V}(x) = \frac{\delta V(x)}{\delta x} f(x) \leq -W(x) \leq 0 \tag{A.3}$$

$$\textit{(iii)}\ V(x) \to \infty \ \textit{as}\ ||x|| \to \infty \ \textit{(radially unbounded)} \tag{A.4}$$

Then the equilibrium of point x_e is globally stable is $W(x) \geq 0$ (positive semi-definite) and globally asymptotically stable is $W(x) > 0$ (positive definite) for all $x \neq 0$.

Lemma A.1 (Barbalat's Lemma, Lemma A.1 Fossen (2011)) *Let $\phi : R_+ \to R$ be a*

uniformly continuous function and suppose that $\lim_{t\to\infty}\int_0^t \phi(\tau)d\tau$ *exists and is finite; then*

$$\lim_{t\to\infty}\phi(t) = 0 \tag{A.5}$$

Barbalat's lemma only guarantees global convergence. If there exists a uniform continuous function $V : R^n \times R_+ \to R$ satisfying:

$$\text{(i) } V(x,t) \geq 0 \tag{A.6}$$

$$\text{(ii) } \dot{V}(x,t) \leq 0 \tag{A.7}$$

$$\text{(iii) } \dot{V}(x,t) \text{ is uniformly continuous} \tag{A.8}$$

Then, according to Barbalat's lemma $\lim_{t\to\infty}\dot{V}(x,t) = 0$.

A.3 Inverse dynamics control stability analysis

This section contains the stability analysis of the controller:

$$\tau_c = B_{tot}^{\dagger}[K_D s' + M(q)\dot{\zeta}_r + C(q,\zeta)\zeta_r + C(q,\zeta)\zeta_r + g(q,\eta)]. \tag{A.9}$$

with $K_D > 0$,

$$s' = \tilde{\zeta} + (\Lambda + K_D^{-1}K_P)\tilde{y}, \tag{A.10}$$

$$\zeta_r = \zeta_d + \Lambda\tilde{y}, \tag{A.11}$$

$$\Lambda = \begin{bmatrix} \lambda_p I_3 & 0_{3\times 3} & 0_{3\times n} \\ 0_{3\times 3} & \lambda_o I_3 & 0_{3\times n} \\ 0_{n\times 3} & 0_{n\times 3} & \Lambda_q \end{bmatrix}, \quad \Lambda_q \in R^{n\times n}, \quad \Lambda > 0, \tag{A.12}$$

and

$$K_P = \begin{bmatrix} k_p I_3 & 0_{3\times 3} & 0_{3\times n} \\ 0_{3\times 3} & k_o I_3 & 0_{3\times n} \\ 0_{n\times 3} & 0_{n\times 3} & K_q \end{bmatrix}, \quad K_q \in R^{n\times n}, \quad K_P > 0. \tag{A.13}$$

The stability analysis presented here is similar to that of the adaptive inverse dynamics controller presented in section 4.2, but without the estimates $\hat{\theta}$.

Define the error variable s:

$$s = \begin{bmatrix} s_p \\ s_o \\ s_q \end{bmatrix} = \begin{bmatrix} \tilde{\nu}_1 \\ \tilde{\nu}_2 \\ \dot{\tilde{q}} \end{bmatrix} + \Lambda \begin{bmatrix} R_I^B \tilde{\eta}_1 \\ \tilde{\epsilon} \\ \tilde{q} \end{bmatrix} = \zeta_d - \zeta + \Lambda\tilde{y} = \dot{\tilde{\zeta}} + \Lambda\tilde{y}. \tag{A.14}$$

Consider the Lyapunov function candidate suggested in (Antonelli; 2014):

$$V = \frac{1}{2} s^\top M(q) s + \frac{1}{2} k_p \tilde{\eta}_1^\top \tilde{\eta}_1 + k_o \tilde{z}^\top \tilde{z} + \frac{1}{2} \tilde{q}^\top K_q \tilde{q} \tag{A.15}$$

where $\tilde{z} = [1 - \tilde{\eta} \quad -\tilde{\epsilon}^\top]^\top$. V has the property $V \geq 0$ because $k_p > 0$, $k_o > 0$ and K_q and $M(q)$ are positive definite. The positive definiteness of $M(q)$ is provided by property 2.1. The time derivative of (A.15) is

$$\dot{V} = \frac{1}{2} s^\top \dot{M} s + s^\top M \dot{s} + k_p \tilde{\eta}_1^\top \dot{\tilde{\eta}}_1 + 2k_o \tilde{z}^\top \dot{\tilde{z}} + \tilde{q}^\top K_q \dot{\tilde{q}}. \tag{A.16}$$

Rewrite the expression using the relation $\dot{\tilde{\eta}} = R_B^I \tilde{\nu}_1$ and that $\dot{\tilde{z}} = -\dot{z} = -J_{k,oq}(z)\tilde{\nu}_2$:

$$\dot{V} = \frac{1}{2} s^\top \dot{M} s + s^\top M \dot{s} + k_p \tilde{\eta}_1^\top R_B^I \tilde{\nu}_1 - 2k_o \tilde{z}^\top J_{k,oq}(z) \tilde{\nu}_2 + \tilde{q}^\top K_q \dot{\tilde{q}}. \tag{A.17}$$

Now, consider the expression for $\dot{s}$

$$\dot{s} = \ddot{\tilde{\zeta}} + \Lambda\dot{\tilde{y}} = \dot{\zeta}_d - \dot{\zeta} + \Lambda\dot{\tilde{y}}. \tag{A.18}$$

Rewrite the expression for $\dot{s}$ using the expression for $\dot{\zeta}$ found from the dynamics

equation (4.4) and the fact that $\dot{\zeta}_d = \dot{\zeta}_r - \Lambda\dot{\tilde{y}}$.

$$\dot{s} = \dot{\zeta}_r - M^{-1}(q)\big(B_{tot}\tau_c - C(q,\zeta)\zeta - D(q,\zeta)\zeta - g(q,\eta)\big). \tag{A.19}$$

Write $\dot{s}$ as a function of ζ_r. To do this, notice that $\zeta_d = \zeta_r - \Lambda\tilde{y}$ (A.11) and $s = \zeta_d - \zeta + \Lambda\tilde{y}$. Combine these to get $\zeta = \zeta_r - s$ which can be substituted into $\dot{s}$ (A.19). Substitute the resulting expression into the derivative $\dot{V}$ (A.17). Remark: In the following the arguments of the system matrices will be omitted for better readability.

$$\dot{V} = s^\top[M\dot{\zeta}_r + C\zeta_r + D\zeta_r + g - B_{tot}\tau_c] - s^\top Ds + \underbrace{k_p\tilde{\eta}_1^\top R_B^I\tilde{\nu}_1 - 2k_o\tilde{z}^\top J_{k,oq}(z)\tilde{\nu}_2 + \tilde{q}^\top K_q\dot{\tilde{q}}}_{\dot{V}_1} \tag{A.20}$$

The term $\frac{1}{2}s^\top\dot{M}s - s^\top Cs$ disappears because of property 2.2. It has been shown in section 4.2 that the last part of the derivative of the Lyapunov function A.20 can be written as. Using this disassembly of s, the last three terms of (4.27), $\dot{V}_1$ can then be written as:

$$\dot{V}_1 = k_p\tilde{\eta}_1^\top R_B^I s_p - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta} + k_o\tilde{\epsilon}^\top s_o - k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} + \tilde{q}^\top K_q s_q - \tilde{q}^\top K_q\Lambda_q\tilde{q} \tag{A.21}$$

The terms that include s_p, s_o and s_q can be collected in a term $s^\top K_P\tilde{y}$ with K_P defined in (A.13). The expression for $\dot{V}$ (4.27) can therefore be written as:

$$\dot{V} = s^\top[M\dot{\zeta}_r + C\zeta_r + D\zeta_r + g - B_{tot}\tau_c + K_P\tilde{y}] - s^\top Ds - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta}_1 - k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} - \tilde{q}^\top K_q\Lambda_q\tilde{q} \tag{A.22}$$

Plug in the control input τ_c (4.10) and replace $s' = s + K_D^{-1}K_P\tilde{y}$.

$$\begin{aligned}\dot{V} =& s^\top[M\dot{\zeta}_r + C\zeta_r + D\zeta_r + g - K_D(s + K_D^{-1}K_P\tilde{y}) - M\dot{\zeta}_r - C\zeta_r - g - D\zeta_r \\ &+ K_P\tilde{y}] - s^\top Ds - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta}_1 - k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} - \tilde{q}^\top K_q\Lambda_q\tilde{q} \\ =& - s^\top K_D s - s^\top Ds - k_p\lambda_p\tilde{\eta}_1^\top\tilde{\eta}_1 - k_o\lambda_o\tilde{\epsilon}^\top\tilde{\epsilon} - \tilde{q}^\top K_q\Lambda_q\tilde{q}.\end{aligned} \tag{A.23}$$

This results in:

$$\dot{V} = -s^\top (K_D + D)s - k_p \lambda_p \tilde{\eta}_1^\top \tilde{\eta}_1 - k_o \lambda_o \tilde{\epsilon}^\top \tilde{\epsilon} - \tilde{q}^\top K_q \Lambda_q \tilde{q} \leq 0. \tag{A.24}$$

From theorem A.1 it can be concluded that the controller gives an asymptotically stable system.

References

Antonelli, G. (2014). *Underwater Robots*, Springer International Publishing.

Antonelli, G. (2018). *Underwater Robots*, Springer International Publishing.

Antonelli, G., Caccavale, F. and Chiaverini, S. (2004). Adaptive tracking control of underwater vehicle-manipulator systems based on the virtual decomposition approach, *IEEE Transactions on Robotics and Automation* **20**(3): 594–602.

Antonelli, G. and Chiaverini, S. (1998). Adaptive tracking control of underwater vehicle-manipulator systems, *Control Applications, 1998. Proceedings of the 1998 IEEE International Conference on*, Vol. 2, IEEE, pp. 1089–1093.

Antonelli, G., Chiaverini, S., Sarkar, N. and West, M. (2001). Adaptive control of an autonomous underwater vehicle: experimental results on odin, *IEEE Transactions on Control Systems Technology* **9**(5): 756–765.

Borlaug, I.-L. G., G. J. S.-T. J. and Pettersen, K. Y. (2017). Trajectory tracking for underwater swimming manipulators using a super twisting algorithm, *SWARM 2017: The 2nd International Symposium on Swarm Behavior and Bio-Inspired Robotics*.

Cataldi, E. and Antonelli, G. (2015). Basic interaction operations for an underwater vehicle-manipulator system, *Advanced Robotics (ICAR), 2015 International Conference on*, IEEE, pp. 524–529.

Chiaverini, S. and Siciliano, B. (1999). The unit quaternion: A useful tool for inverse kinematics of robot manipulators, *Systems Analysis Modelling Simulation* **35**(1): 45–60.

Chiaverini, S., Siciliano, B. and Villani, L. (1999). A survey of robot interaction control schemes with experimental comparison, *IEEE/ASME Transactions on Mechatronics* **4**(3): 273–285.

Cui, Y. and Yuh, J. (2003). A unified adaptive force control of underwater vehicle-manipulator systems (uvms), *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, Vol. 1, IEEE, pp. 553–558.

de Wit, C. C., Siciliano, B. and Bastin, G. (2012). *Theory of robot control*, Springer Science & Business Media.

Egeland, O. and Gravdahl, J. T. (2002). *Modeling and simulation for automatic control*, Vol. 76, Marine Cybernetics Trondheim, Norway.

Evans, J., Redmond, P., Plakas, C., Hamilton, K. and Lane, D. (2003). Autonomous docking for intervention-auvs using sonar and video-based real-time 3d pose estimation, *Oceans 2003. Celebrating the Past ... Teaming Toward the Future (IEEE Cat. No.03CH37492)*, Vol. 4, pp. 2201–2210 Vol.4.

Ferretti, G., Magnani, G. and Rocco, P. (1997). Toward the implementation of hybrid position/force control in industrial robots, *IEEE Transactions on robotics and automation* **13**(6): 838–845.

Fossen, T. I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*, John Wiley & Sons, Ltd.

Fossen, T. I., Johansen, T. A. and Perez, T. (2009). *A survey of control allocation methods for underwater vehicles*, INTECH Open Access Publisher.

Hamerlain, A. G. A. B. M. (2013). Higher order sliding mode control of robot manipulator, *The Ninth International Conference on Autonomic and Autonomous Systems* .

Hogan, N. (1984). Impedance control: An approach to manipulation, *1984 American Control Conference*, pp. 304–313.

Kelasidi, E., Pettersen, K. Y. and Gravdahl, J. T. (2014). Stability analysis of underwater snake robot locomotion based on averaging theory, *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, pp. 574–581.

Kelasidi, E., Pettersen, K. Y., Gravdahl, J. T. and Liljebäck, P. (2014). Modeling of underwater snake robots, *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4540–4547.

Kelasidi, E., Pettersen, K. Y., Gravdahl, J. T., Strømsøyen, S. and Sørensen, A. J. (2017). Modeling and propulsion methods of underwater snake robots, *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 819–826.

Khalil, H. (2002). *Nonlinear Systems*, Pearson Education, Prentice Hall.

Lee, M. and Choi, H.-S. (2000a). A robust neural controller for underwater robot manipulators, *IEEE Transactions on Neural Networks* **11**(6): 1465–1470.

Lee, M. and Choi, H.-S. (2000b). A robust neural controller for underwater robot manipulators, *IEEE Transactions on Neural Networks* **11**(6): 1465–1470.

Lee, P.-M. and Yuh, J. (1999). Application of non-regressor based adaptive control to an underwater mobile platform-mounted manipulator, *Proceedings of the 1999 IEEE International Conference on Control Applications (Cat. No.99CH36328)*, Vol. 2, pp. 1135–1140 vol. 2.

LEVANT, A. (1993). Sliding order and sliding accuracy in sliding mode control, *International Journal of Control* **58**(6): 1247–1263.

Marani, G., Choi, S. K. and Yuh, J. (2009). Underwater autonomous manipulation for intervention missions auvs, *Ocean Engineering* **36**(1): 15 – 23. Autonomous Underwater Vehicles.
URL: *http://www.sciencedirect.com/science/article/pii/S002980180800173X*

McMillan, S., Orin, D. E. and McGhee, R. B. (1995). Efficient dynamic simulation of an underwater vehicle with a robotic manipulator, *IEEE Transactions on Systems, Man, and Cybernetics* **25**(8): 1194–1206.

Morison, J. R., J. J. W. . S. S. A. (1950). The force exerted by surface waves on piles, *Journal of Petroleum Technology* **2**(5): 149 – 154.

Oh, Y., Chung, W. K., Youm, Y. and Suh, I. H. (1998). Motion/force decomposition of redundant manipulator and its application to hybrid impedance control, *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, Vol. 2, pp. 1441–1446 vol.2.

Ortega, R. and Spong, M. W. (1988). Adaptive motion control of rigid robots: a tutorial, *Proceedings of the 27th IEEE Conference on Decision and Control*, pp. 1575–1584 vol.2.

Palomeras, N., Penalver, A., Massot-Campos, M., Vallicrosa, G., Negre, P. L., Fernández, J. J., Ridao, P., Sanz, P. J., Oliver-Codina, G. and Palomer, A. (2014). I-auv docking and intervention in a subsea panel, *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, IEEE, pp. 2279–2285.

Pandian, S. R. and Sakagami, N. (2010). A neuro-fuzzy controller for underwater robot manipulators, *2010 11th International Conference on Control Automation Robotics Vision*, pp. 2135–2140.

Perrier, M., Brignone, L. et al. (2004). Optical stabilization for the alive intervention auv, *The Fourteenth International Offshore and Polar Engineering Conference*, International Society of Offshore and Polar Engineers.

Raibert, M. H. and Craig, J. J. (1981). Hybrid position/force control of manipulators, *Journal of Dynamic Systems, Measurement, and Control* **103**(2): 126–133.

Ridao, P., Carreras, M., Ribas, D., Sanz, P. J. and Oliver, G. (2014). Intervention auvs: the next challenge, *IFAC Proceedings Volumes* **47**(3): 12146–12159.

Salisbury, J. K. (1980). Active stiffness control of a manipulator in cartesian coordinates, *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pp. 95–100.

Sarkar, N., Yuh, J. and Podder, T. K. (1999). Adaptive control of underwater vehicle-manipulator systems subject to joint limits, *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No.99CH36289)*, Vol. 1, pp. 142–147 vol.1.

Schjolberg, I., Gjersvik, T. B., Transeth, A. A. and Utne, I. B. (2016). Next generation subsea inspection, maintenance and repair operations, *IFAC-PapersOnLine* **49**(23): 434 – 439. 10th IFAC Conference on Control Applications in Marine SystemsCAMS 2016.

Schmidt-Didlaukies, H. (2018). The quick snake robot model, *Technical report*, Norwegian University of Science and Technology.

Shtessel, Y. B., Moreno, J. A., Plestan, F., Fridman, L. M. and Poznyak, A. S. (2010). Super-twisting adaptive sliding mode control: A lyapunov design, *49th IEEE Conference on Decision and Control (CDC)*, pp. 5109–5113.

Shtessel, Y., Plestan, F. and Taleb, M. (2011). Lyapunov design of adaptive super-twisting controller applied to a pneumatic actuator, *IFAC Proceedings Volumes* **44**(1): 3051 – 3056. 18th IFAC World Congress.

Shtessel, Y., Taleb, M. and Plestan, F. (2012). A novel adaptive-gain supertwisting sliding mode controller: Methodology and application, *Automatica* **48**(5): 759 – 769. **URL:** *http://www.sciencedirect.com/science/article/pii/S0005109812000751*

Siciliano, B. and Khatib, O. (2007). *Springer Handbook of Robotics*, Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Siciliano, B., Sciavicco, L., Villani, L. and Oriolo, G. (2008). *Robotics: Modelling, Planning and Control*, 1st edn, Springer Publishing Company, Incorporated.

Slotine, J.-J. E. and Li, W. (1987). On the adaptive control of robot manipulators, *The international journal of robotics research* **6**(3): 49–59.

Sverdrup-Thygeson, J., Kelasidi, E., Pettersen, K. and Gravdahl, J. (2016a). A control framework for biologically inspired underwater swimming manipulators equipped with thrusters, *IFAC-PapersOnLine* **49**(23): 89 – 96. 10th IFAC Conference on Control Applications in Marine SystemsCAMS 2016.
URL: *http://www.sciencedirect.com/science/article/pii/S2405896316319139*

Sverdrup-Thygeson, J., Kelasidi, E., Pettersen, K. and Gravdahl, J. (2016b). Modeling of underwater swimming manipulators, *IFAC-PapersOnLine* **49**(23): 81 – 88. 10th IFAC Conference on Control Applications in Marine SystemsCAMS 2016.
URL: *http://www.sciencedirect.com/science/article/pii/S2405896316319127*

Sverdrup-Thygeson, J., Kelasidi, E., Pettersen, K. Y. and Gravdahl, J. T. (2016c). The underwater swimming manipulator - a bio-inspired auv, *2016 IEEE/OES Autonomous Underwater Vehicles (AUV)*, pp. 387–395.

Sverdrup-Thygeson, J., Kelasidi, E., Pettersen, K. Y. and Gravdahl, J. T. (2018). The underwater swimming manipulator—a bioinspired solution for subsea operations, *IEEE Journal of Oceanic Engineering* **43**(2): 402–417.

Yuh, J. (1996). An adaptive and learning control system for underwater robots, *IFAC Proceedings Volumes* **29**(1): 145 – 150. 13th World Congress of IFAC, 1996, San Francisco USA, 30 June - 5 July.
URL: *http://www.sciencedirect.com/science/article/pii/S1474667017576533*

Yuh, J. and Nie, J. (2000). Application of non-regressor-based adaptive control to underwater robots: experiment, *Computers & Electrical Engineering* **26**(2): 169 – 179.
URL: *http://www.sciencedirect.com/science/article/pii/S0045790699000397*

Zhao, S. and Yuh, J. (2005). Experimental study on advanced underwater robot control, *IEEE Transactions on Robotics* **21**(4): 695–703.