Copyright Statement

Plymouth University

Doctoral Thesis

Biomimetic Manipulator Control Design for Bimanual Tasks in the Natural Environment

Author:
Alexander M.C. Smith

Supervisor:
Dr. Chenguang Yang

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

Centre for Robotics and Neural Systems
School of Computing, Electronics and Mathematics

August 2016

Acknowledgements

I would like to thank my supervisor Dr. Chenguang Yang, who has brought me to this point. I would also like to thank Dr. Phil Culverhouse and Prof. Angelo Cangelosi for their support over the past four years. Thanks go to Prof. Etienne Burdet at Imperial College for inspiration and technical support, also provided by Prof. Hongbin Ma at the Beijing Institute of Technology. I would also like to show my appreciation for all my friends and colleagues in the CRNS who have helped and supported me along the way (I'll miss you all): Tony M, Fede, Martin P, Martin S & Bea, Sam, Beata, Michael, Tony B, Frank, Salo, Ale, Horatio, Steve and Aash... Finally, special thanks to my partner Clare, without whom I would be much worse off.

Declaration of Authorship

I, Alexander M.C. Smith, declare that this thesis titled, 'Biomimetic Manipulator Control Design for Bimanual Tasks in the Natural Environment' and the work presented in it are my own. I confirm that:

- at no time during the registration for the research degree has the author been registered for any other University award, without prior agreement of the Graduate Sub-Committee,
- no work submitted for a research degree at Plymouth University may form part of any other degree either at the University or at another establishment,
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work,
- I have acknowledged all main sources of help.

This study was partially supported by EPSRC grants EP/L026856/1 and EP/J004561/1 (BABEL) and carried out partly in collaboration with Imperial College London.

Word Count of main body of thesis: 21,991

Signed: _____

Date: _____

Abstract

Biomimetic Manipulator Control Design for Bimanual Tasks in the Natural Environment

by Alexander McKenzie Chalmers Smith

As robots become more prolific in the human environment, it is important that safe operational procedures are introduced at the same time; typical robot control methods are often very stiff to maintain good positional tracking, but this makes contact (purposeful or accidental) with the robot dangerous. In addition, if robots are to work cooperatively with humans, natural interaction between agents will make tasks easier to perform with less effort and learning time. Stability of the robot is particularly important in this situation, especially as outside forces are likely to affect the manipulator when in a close working environment; for example, a user leaning on the arm, or task-related disturbance at the end-effector.

Recent research has discovered the mechanisms of how humans adapt the applied force and impedance during tasks. Studies have been performed to apply this adaptation to robots, with promising results showing an improvement in tracking and effort reduction over other adaptive methods. The basic algorithm is straightforward to implement, and allows the robot to be compliant most of the time and only stiff when required by the task. This allows the robot to work in an environment close to humans, but also suggests that it could create a natural work interaction with a human. In addition, no force sensor is needed, which means the algorithm can be implemented on almost any robot.

This work develops a stable control method for bimanual robot tasks, which could also be applied to robot-human interactive tasks. A dynamic model of the Baxter robot is created and verified, which is then used for controller simulations. The biomimetic control algorithm forms the basis of the controller, which is developed into a hybrid control system to improve both task-space and joint-space control when the manipulator is disturbed in the natural environment. Fuzzy systems are implemented to remove the need for repetitive and time consuming parameter tuning, and also allows the controller to actively improve performance during the task. Experimental simulations are performed, and demonstrate how the hybrid task/joint-space controller performs better than either of the component parts under the same conditions. The fuzzy tuning method is then applied to the hybrid controller, which is shown to slightly improve performance as well as automating the gain tuning process. In summary, a novel biomimetic hybrid controller is presented, with a fuzzy mechanism to avoid the gain tuning process, finalised with a demonstration of task-suitability in a bimanual-type situation.

Contents

List of Figures

List of Tables

Nomenclature

Symbol	Description	Unit
n	Number of joints	
$I^{m\times m}$	$(m \times m)$ identity matrix	
τ	Torque	N$\cdot$m
τ_r	Reference torque	N$\cdot$m
τ_u	Input torque	N$\cdot$m
P	power	W (J/s)
F	Force	N
v	Velocity	m/s
θ	Angular position	rad
ω	Angular velocity	rad/s
$\dot{\omega}$	Angular acceleration	rad/s/s
q	Joint position	rad
$\dot{q}$	Joint velocity	rad/s
$\ddot{q}$	Joint acceleration	rad/s^2
X	Cartesian position	m, rad
X_d	Desired Cartesian position	m
$\dot{X}$	Cartesian velocity	m/s, rad/s
${}^iT_j \in \Re^{4\times4}$	Homogenous transform matrix from i to j	
J	Jacobian matrix	
J^T	Transpose Jacobian matrix	
$J^\dagger$	Pseudo-inverse Jacobian matrix	
$M(q) \in \Re^{n\times n}$	Mass matrix	Kg
$C(q,\dot{q}) \in \Re^n$	Coriolis/centrifugal force vector	N
$G(q) \in \Re^n$	Gravitational force vector	N
e	Position error	m, rad
$\dot{e}$	Velocity error	m/s, rad/sec
ε	Tracking error	
K	Stiffness coefficient	N/m
D	Damping coefficient	N/m/s

γ	Forgetting factor	
α	Forgetting factor tuning gain	
$Q_{(\cdot)}$	Learning gain	
Ω	Weighting matrix	
f	Frequency	Hz
A	Amplitude	
η	Performance index	
$\mathrm{Tr}(\cdot)$	Trace of matrix	
$\otimes$	Kronecker product	
$\|\| \cdot \|\|$	Euclidean norm	

Chapter 1

Introduction

In the past, most research within robot control has been focused on accurate, repeatable and stiff movement; this can be attributed to most robots being used within industry, moving in free space to improve quality assurance and reduce manufacturing costs [2]. Tasks such as welding, spray painting and pick and place operations generally do not place any uncertain dynamics upon the robot, as well as having a set path or several set paths to work along. This allows simple control methods, e.g. PID control [3], to reach accurate joint positions, and removes the need for computationally expensive inverse kinematics.

However, as the field has expanded, modern robots will interact with fragile objects, other machines and humans [4] [5]. In addition, manipulators can be highly redundant [6] or under-actuated [7], creating new avenues of robot control problems. Interaction with the environment, such as objects or humans, requires feedback to allow the robot to maintain its objective while compensating for obstacles [8]. Further, the likely future proliferation of robots within our natural environment (particularly working with the elderly) [9] requires manipulator design to follow that of a human - it is much easier to design a machine to fit our world rather than change our whole environment to accommodate the robot. Hence, having a robot that can use two arms gives the ability to perform a much wider range of tasks than if it only had the use of one arm, such as lifting heavy objects, carrying delicate items or carrying out two tasks at once. In addition, with certain tool tasks – such as drilling or carving – the force required to perform the action can be spread across two manipulators, reducing the need for very strong, heavy and costly motors to be included in the manipulator design.

The natural environment -as opposed to the structured environment of a factory setting- is full of unpredictable disturbances which cannot be modelled before placing the robot

inside that environment. There are two ways to deal with this: using sensors to create predictive models of the environment, allowing the robot to change behaviour to compensate [10, 11], or design the control algorithm to be robust against disturbance [12–15], which can use a number of techniques for observing disturbance levels and compensating for them. Predictive models tend to be used in the mobile robot setting, where disturbance or obstacles come in the form of static objects or humans walking around, whereas the robust control methods are used for task-specific control such as pick-and-place operations.

More recently, research into impedance control has been focused on adaptation, i.e. online adjustment of stiffness and damping of the manipulator [16]. This is largely due to research which reveals the fact that the human central nervous system (CNS) adapts impedance to achieve various tasks [17–21]. Burdet et al. [22] demonstrated that humans are able to learn to stabilise unstable dynamics using energy-efficient, selective control of impedance geometry. The CNS is able change the magnitude and geometry of endpoint stiffness through selective activation of agonist and antagonist muscles, to maintain stability and minimise energy expenditure. These properties, if implemented in a robot system, would not only improve manipulator performance but could also improve the "natural feel" of interaction in scenarios where humans and robots are working together.

At the beginning of this study, the "hummingbird problem" was proposed: the theoretical problem for a robot to hold a delicate, vibrating object. A gentle grasp must be maintained so as to not cause damage, yet stiff enough so that the vibration does not break the grip. This grip must be maintained while moving the object through a trajectory and subjected to environmental disturbances, such as being bumped by humans working in close proximity. This problem is also related to sloshing control, where the object being moved contains a liquid which imparts dynamic forces at the end-effector [23, 24], and can be also be imagined as the difficult problem of carrying a tray of drinks through a crowded room. These tasks could be achieved by most humans with practice, which becomes the motivation for developing the control methods in this study using the human model of compliant control.

The **first** contribution of this thesis is the development of a hybrid control scheme, combining both joint-space and task-space control for improved rejection of internal and environmental disturbances. After developing the biomimetic controller in Chapter 3, it was noted that joint-space control appears to produce good tracking in the joint space, but small steady-state errors produce a divergent error in the task-space. Mapping of the biomimetic controller to the task-space is trivial, which is then combined via a weighting mechanism to produce a hybrid-space controller. Simulated experiments, where the manipulator is subjected to disturbances similar to what would be expected

in the natural environment, show that the hybrid controller performs better than the individual joint-space or task-space control without increasing control effort or becoming unstable.

During experimental verification of the controller it was noted that manual gain tuning was extremely time consuming, as the hybridisation meant that there was now double the number of parameters requiring attention. This became the motivation to develop the **second** contribution of this work: the implementation of fuzzy logic controllers for online estimation of control parameters. Fuzzy logic is well suited to applications where expert knowledge exists, and can be implemented in a Mamdani-type control using linguistic rules. During verification of the fuzzy tuning method is was also noted that tuning online slightly improves control performance and increases robustness to different disturbance types.

Finally, the **third** contribution is the development compliant bimanual trajectory planning system, based on a master-slave approach. The aim of this was to verify the controller is capable of carrying out tasks such as those described earlier in this chapter, specifically the "hummingbird problem": the initial inspiration for carrying out this research. The fuzzy tuning method developed in Chapter 5 is particularly useful in this context, as the bimanual system has four sets of gains to be tuned. The hybrid controller described in Chapter 4 performs well in this task, being able to maintain a contact force with the target object and remain robust to external disturbances, all without the need for force sensors within the control loop.

In addition, a full analytical representation of the dynamics of the Baxter robot is made publicly available. This was developed and verified for simulations of the described controller, so that results are representative of controller performance when applied to a real, commercially available robot. Baxter is widely used in the research community and the analytical form of the closed dynamics could prove useful to many researchers.

1.1 List of Publications

Parts of this work have been submitted as a journal paper and to several conferences during the course of study:

- Smith, A. Yang, C. Ma, H. Culverhouse, P. Cangelosi, A. Burdet, E; *Novel Hybrid Adaptive Controller for Manipulation in Complex Perturbation Environments* PloS one. **vol. 10, no 6, 2015**.

- Smith, A. Yang, C. Ma, H. Culverhouse, P. Cangelosi, A. Burdet, E; *Dual adaptive control of bimanual manipulation with online fuzzy parameter tuning* Intelligent Control (ISIC), 2014 IEEE International Symposium on, **Juan Les Pins, 10/2014, pp. 560-565**

- Smith, A. Yang, C. Ma, H. Culverhouse, P. Cangelosi, A. Burdet, E; *Biomimetic joint/task space hybrid adaptive control for bimanual robotic manipulation* 11th IEEE International Conference on Control & Automation (ICCA) **Taichung, 06/2014, pp. 1013 - 1018**

- Yang, C. Ma, H. Fu, M. Smith, A; *Optimized model reference adaptive motion control of robot arms with finite time tracking* Control Conference (CCC), 2012 31st Chinese **Hefei, 07/2012, pp. 4356 - 4360**

- Wang, B. Yang, C. Li, Z. Smith, A; *sEMG-Based control of an exoskeleton robot arm* Proceedings of the 5th international conference on Intelligent Robotics and Applications **Montreal, 10/2012, pp. 63 - 73**

- Yang, C. Li, Z. Li, J. Smith, A; *Adaptive Neural Network Control of Robot with Passive Last Joint* Proceedings of the 5th international conference on Intelligent Robotics and Applications **Montreal, 10/2012, pp. 113-122**

- Smith, A. Yang, C. Wang, X. Ma, H; *Lagrange-Euler Closed Dynamics Model of the Baxter Robot* International Conference on Mechatronics and Automation (ICMA) **Accepted May 2016**

Chapter 2

Literature Review

2.1 Methods of Robot Control

2.1.1 Classical Control

The science of control theory can be traced back to the mid 19^{th} century [25], where centrifugal governors were used to regulate the input to steam engines to produce a constant output. Maxwell's work [26] began to generate interest in the subject, particularly in the study of how feedback control affects the stability of the system and how to avoid instability due to constructive oscillation at the output. The (very) generalised diagram of a feedback system is shown in Figure 2.1; $u(t)$ is the reference input signal, i.e. the ideal output of the system, $y(t)$ is the measured output of the system, from which the error signal $e(t) = u(t) - y(t)$ can be calculated, when can be described at the variation of the actual output compared to the ideal output. The controller is some method of using this information to drive the plant dynamics such that the output $y(t)$ is aligned to the reference signal. This controller is the subject of most of the research in the subject, and can be composed of many different methods; originally being formed from rigorous mathematical models such as the Linear Quadratic Regulator [27], which typically controls the output via minimisation of the energy of the system, to more modern theoretical methods such as the Artificial Neural Network (ANN), which uses a biological model to act as a sort of "black box" system, where the dynamics of the controller are uncertain but are gradually learned over time or iteration to produce a suitable output to the plant.

In terms of robotics, control of early manipulators was aimed at minimising trajectory error, due to their primary application at the time in industrial manufacture [28]. Typical control methods at the time included resolved motion rate control (RMRC) [29],

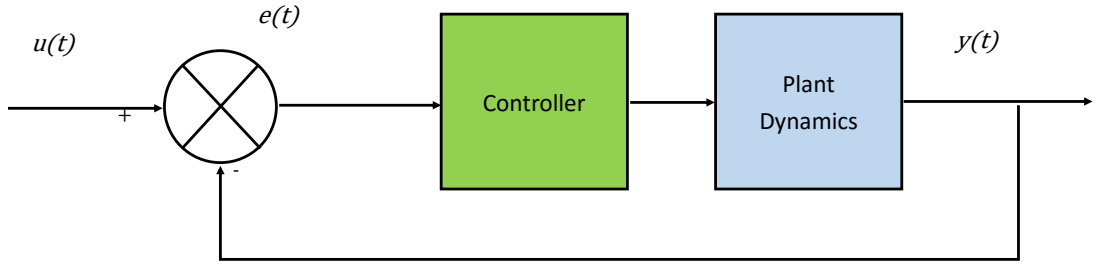

FIGURE 2.1: Generalised diagram of a simple feedback control system.

computed torque control [30], near-minimum-time control [31] and early adaptive control [32, 33]. These techniques are generally based on a classical Newtonian/Lagrangian representation of the manipulator dynamics, from which the control method is derived. This allows the control algorithm to be rigorously analysed for performance (particularly stability and trajectory error) using well documented techniques, and is easily modelled computationally so that it can be tested offline (i.e. without a physical robot), which is safer and faster than online testing.

The RMRC method is one of the simplest methods of manipulator control. It relies on solving the inverse kinematics of the manipulator through the inverse of the Jacobian $J(q)$, which relates the joint velocity to the task-space velocity

$$\delta q = J^{-1}(q)\delta X, \tag{2.1}$$

where δq is the joint space corrective command to the robot and δX is the positional and orientation error in the task space. For robots where $n = 6$ solving Equation 2.1 is relatively trivial, involving only a matrix inversion and multiplication; however, if $n \neq 6$ then $J(q)$ is non-square, and requires the use of some other inversion method, such as the Jacobian transpose, Moore-Penrose pseudo-inverse, damped least squares, etc. [34], although each of these also have their own issues. The calculation of the pseudo-inverse has become less of an issue as computational power has increased, although it is still non-trivial enough to require a powerful modern-day computer to calculate in real time. Additionally, RMRC suffers from instability issues when close to singularity (i.e. a robot configuration q_s where $J(q_s)$ is not of full rank) although research has shown methods to avoid this problem [35–39].

Computed torque control, shown in Figure 2.2, relies on a dynamic model of the manipulator to generate a desired torque from a reference trajectory, which is then fed to the robot from which position and velocity feedback is used for control. The equation for

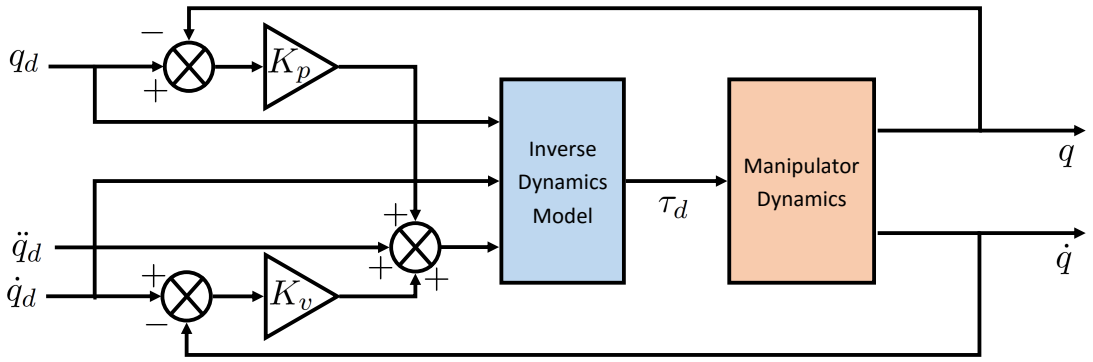

FIGURE 2.2: Block diagram of the computed torque control technique.

calculation of the command torque τ_d is given as

$$\tau_d = M(q)\big(\ddot{q}_r + K_v(\dot{q}_r - \dot{q}) + K_p(q_r - q)\big) + C(q,\dot{q})\dot{q} + G(q) \tag{2.2}$$

where the $M(q)$, $C(q,\dot{q})$ and $G(q)$ are the dynamic parameters of the manipulator (discussed further in Equation 3.1), $q_r, \dot{q}_r, \ddot{q}_r$ are the position, velocity and acceleration of the reference trajectory and $q, \dot{q}$ are the position and velocity feedback of the robot. The gains K_p and K_v can be tuned to change the stiffness and damping of the system. In essence, the dynamic model of the robot is used to generate an ideal torque for the robot (i.e. if the real dynamics matched the model exactly) and compared to the behaviour of the robot in the real world. The dynamic model is seldom truly accurate (hence the need for feedback), as other terms such as frictions and gear backlash are often present but difficult to model accurately. By keeping K_p and K_v low it is possible to create a sort of "soft" control, something akin to impedance control (see Section 2.1.2) although stability is difficult to maintain if gains are too low, or indeed too high [40]. Although it has been shown to be effective compared to other methods [41], due to the need for an accurate dynamic model it may not be suitable for some manipulators where dynamic parameters are not available, or where uncertain terms such as friction are large. However, since the original manifestation various techniques have been shown to mitigate these problems; for example, the application a fuzzy system to compensate for uncertain manipulator dynamics [42–44] or gain tuning [45], or the use of neural networks for similar improvements [46–48].

Due to it's simplicity and history of application, Proportional-Integral-Derivative (PID) control is a very common control method in many current robotics applications. As far back as the 1940's [49] research has been conducted in the tuning, stability and application of PID control and its derivatives. The essential idea behind PID control is the ability to control the gain of not only the base feedback, but also the integral and the derivative of that signal. An easy example to visualise, with relation to Figure 2.3,

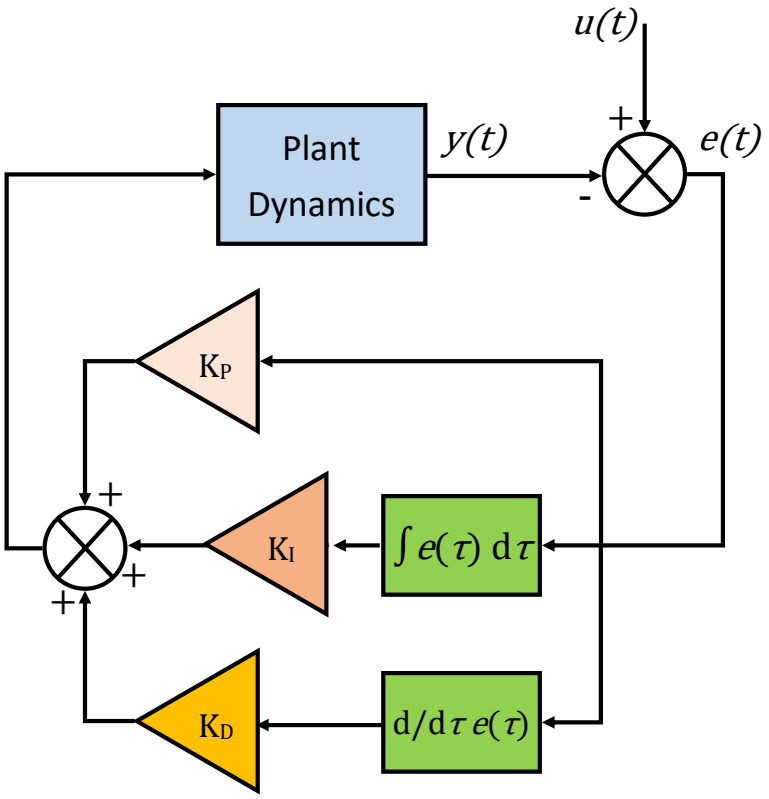

FIGURE 2.3: Simplified block diagram of PID control architecture.

would be if the input and output of the system ($u(t)$ and $y(t)$ respectively) are the velocity of the system; the velocity response of the system can be adjusted via K_P, the position response by tuning K_I and the acceleration through tuning of K_D. The careful selection of these feedback gains can be used to create a system with a very fast response with minimal tracking error, or a slowly responding system which can maintain stability in high disturbance environments; in terms of robotics, a well tuned system can compensate for the arm dynamics or end-effector load without requiring an accurate system model. However, the tuning of PID gains is mainly performed through trial and error; additionally the tuning has to be performed online so that the response can be observed. For example, the Ziegler-Nichols method [49] is a heuristic which involves taking the system to near instability by increasing K_P, then following a set of steps to tune K_I and K_D from this point. More modern methods have developed on this, such as the work of Åström and Hägglund [50], whose method is much less aggressive to the system (gains are kept lower, rather than constantly increasing them towards an instability) and also describes how high-frequency disturbances, such as the unavoidable sensor noise, can be compensated for gracefully. As with many other classical control techniques automatic tuning has been implemented using fuzzy logic, in a range of applications such as DC motor or aircraft pitch control [51–54]; fuzzy logic

lends itself particularly well to this, as the tuning heuristics can be directly implemented in a Mamdani-type system.

2.1.2 Impedance Control

It could be argued that the work of Hogan [8] describes the control method from which most modern manipulator impedance control systems are derived. It is described in this section as the method set forward is a fundamental component in modern controllers. He argued that due to the requirement of the manipulator being mechanically coupled to an object, control of a vector such as position or force is inadequate, and that the impedance must also be taken into account. Impedance control essentially is a method from which the exchange of mechanical work between the manipulator and the environment is regulated. In any system energy flow is defined by the product of two variables, an effort and a flow, for example electric power:

$$P(t) = V(t)I(t) \tag{2.3}$$

where V is the voltage (effort) and I is the current (flow) in a given system; in a mechanical system:

$$P(t) = Fv \tag{2.4}$$

where F is the applied force and v is the velocity. These two types of variable, effort and flow can be used to describe two types of system: admittances and impedances. An admittance accepts effort and gives a flow (e.g. applying a torque to a motor gives angular velocity) while an impedance takes a flow and yields effort (e.g. a damper produces an opposing force to a velocity input). Hogan argued that this distinction is fundamental in controller design. A dynamic system must consist of an admittance and impedance. The environment can sometimes be described as an impedance – motion in, force out – but this does not always hold true; however, it can always be described as an admittance, force in, motion out. Therefore the manipulator should be controlled as an impedance system. Assuming a robot that is capable of producing a joint-level torque τ, measuring joint angle θ with a homogenous transform from base to end-effector $T(\theta)$, a controller which relates force F to position X (i.e. stiffness) can be described. Given a desired equilibrium position X_d the general form of relation from F to X can be described as:

$$F = K(X_d - X) \tag{2.5}$$

where K is the stiffness gain. The Jacobian matrix relates the differential of position to joint velocities:

$$\dot{X} = J(\theta)\dot{\theta} \tag{2.6}$$

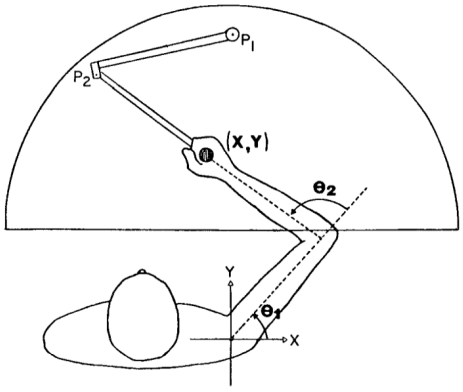

FIGURE 2.4: Diagram of manipulandum for capturing human arm movement. The forearm is typically fixed at the handle so no flexion or extension can occur at the wrist. Participants are generally asked to translate the handle at (x, y) from one point to another. Encoders placed at the joints can record the trajectories of the links, and therefore the trajectories of the human's shoulder and elbow.

From the standard relation of force and torque τ can be defined:

$$\tau = J^T(\theta)F \tag{2.7}$$

Taking Equations 2.6 and 2.7 the controller can be described in terms of a torque input to a robot:

$$\tau = J^T(\theta)K(X_d - T(\theta)) \tag{2.8}$$

As can be seen in Equation 2.8 the input command to the manipulator is a torque vector τ and the output required is joint angle θ, which is a reasonable input-output requirement for most robotic manipulators. It can also be noted that the inverse Jacobian is not required, reducing computational complexity.

The dynamic movement of a manipulator has had some differing views in the literature. Flash et al. [55] show that a minimum jerk model (jerk being the first differential of acceleration) closely matches observed human planar two-joint movements, which were carried out using a two-link manipulandum (see Figure 2.4). It was shown that the model is not just curve fitting, but also allows past predictions to be verified: that velocity increases with distance to maintain trajectory time [56], there exists temporal coupling between hand curvature and speed [57]. There also exists literature which shows that other movements to two-joint planar movements, such as handwriting and three-dimensional movements, are also observed to maintain movement scaling with time [58].

A different theory was put forward by Bullock et al. [59]. They developed a neural network model which he named vector-integration-to-endpoint (VITE). With this model arm movements emerge from continuously updated network interactions, rather than

a predefined task trajectory. Impedance is controlled through opposing interactions, regulating input to agonist and antagonist muscle groups. Bullock et al. were able to show that their VITE model could outperform Hogan's earlier work with minimum-jerk [60] in terms of matching test data, as well as pointing out that the minimum-jerk model erroneously predicts a symmetric velocity profile. In addition, it is noted that Hogan's work does not contain any representation of vector cells as in the brain, whereas the VITE model is built upon this foundation.

The VITE system has been developed in research since its inception in 1988 [61] [62] [63] [64]. For example, the work of Hersch and Billard [65] improved the practical implementation of the VITE model by using two concurrent systems - one in the task space, one in joint space. The controller in Cartesian space ensures that the manipulator reaches the target position, while the joint space controller safeguards the robot against hitting joint limits, which can cause instability [66]. This multi-referential controller was inspired by works that argue that this system is essential to human control of reaching movements [67] [68]. It must be noted that Hersch and Billard do not claim that their controller is a biological analogue, but that it is biologically inspired.

Both models, minimum-jerk and VITE, demonstrate systems which can accurately describe human arm movement. The former is arguably easier to implement, and therefore an attractive choice for robot control – real-time control is usually higher priority than the level of biomimetics. The VITE model, in particular the model developed by Hersch and Billard, sits on a better biological representation as well as producing trajectories resistant to disturbance and avoiding dangerous joint-limit situations.

2.1.3 Biomimetic Impedance Control

More recently, research into impedance control has been focused on adaptation, that is, constant adjustments made to the stiffness and damping of the manipulator. This is largely due to research that shows that this is realised in the human central nervous system (CNS) [17–21, 69]. Important work in this field was carried out by Burdet et al. [22]. They demonstrated that humans are able to learn to stabilise unstable dynamics using energy-efficient, selective control of impedance geometry. In other words, the CNS can change the shape, magnitude and orientation of the impedance ellipse through selective activation of agonist and antagonist muscles, to maintain stability and minimise energy expenditure. Muscle co-activation allows change of endpoint impedance without changing force. This mechanic is the core of minimising metabolic cost. This is highly desirable in manipulator control, to have minimum energy expenditure as well as good trajectory tracking. Mimicking this biological process of action is a strong avenue of

research, similar to biomimetic robot designs described later, in Section 2.1.4. In [70] a biomimetic controller is developed using these principles that adapts simultaneously the impedance, force and trajectory. it is proposed that overall control torque w is composed of feedforward torque u and feedback error v:

$$w = u + v \tag{2.9}$$

where

$$v^k \equiv K_S^k e^k + K_D^k \dot{e}^k, \quad e^k \equiv q_r^k - q^k \tag{2.10}$$

where $(\cdot)^k$ is the iteration or time step, $(\cdot)_r$ denotes a reference variable. Feedforward torque u is adapted as such:

$$\Delta u^k \equiv u^{k+1} - u^k \equiv \alpha \varepsilon^k - (1 - \mu) u^k \tag{2.11}$$

where $\varepsilon = \dot{e}^k + \Gamma e^k$ is the tracking error, and Γ is a positive diagonal matrix and μ is a forgetting factor to account for muscle fatigue. The stiffness is adapted using the equation:

$$\Delta K_S^k \equiv K_S^{k+1} - K_S^k \equiv \beta |\varepsilon^k| - \gamma \tag{2.12}$$

where $\beta, \gamma > 0$. The damping follows a relation to the square root of the stiffness coefficient, as described later in this section. The trajectory is also adapted, with the following law:

$$\Delta q_r^k \equiv q_r^{k+1} - q_r^k \equiv -\delta_1 \varepsilon^k + \delta_2 (q_d^k - q_r^k) \tag{2.13}$$

where δ_1 and δ_2 represent positive weights of the two attractor trajectories, $\delta_1 >> \delta_2$, and q_d represents the original desired trajectory, which q_r is initially equal to. With this controller implemented on a 1-DoF manipulator, experimental results show that the controller is robust against high frequency perturbations through an increase in stiffness. Low frequency errors are initially met with an increase in force to attempt to return to the reference position. Applied force will eventually diminish, due to the adaptation of the reference trajectory and the forgetting factor μ. The adaptation laws described as such have since been applied in a Cartesian impedance controller and, more recently, implemented in a haptic identification context [71] [72] [69].

Maeda et al. [73] investigated a method for human-robot cooperation. They proposed using the minimum-jerk model (mentioned in section Section 2.1.2) in real time, adapting a virtual compliance control. The controller was evaluated by studying the energy transfer within interactions. Results showed a reduction in unnecessary energy transfer. However, although showing a step towards improving human-robot interaction, this type of controller will only work with a robot following a fixed trajectory and may not be

suitable for uncertain tasks, such as following the movement of a human operator.

The work of Duchaine et al. [74] [75] coincides with that of Franklin, Yang, Tee and Milner. In [75] they describe how the stiffness of the human arm is increased with activation of agonist and antagonist muscles, while the damping is due to the viscous properties of muscle tissue. Studies have shown [76] [77] that the damping ratio follows a constant relation to the stiffness coefficient:

$$D = \eta\sqrt{K} \tag{2.14}$$

where D and K are the damping and stiffness coefficients respectively, and η is a constant weighting factor. Through the use of the Lyapunov theorem [78] stability frontiers are found as well as critical impedance parameters to be used as a guideline. Experimental results showed that the tools developed can provide accurate stability prediction, which is useful for future development of adaptive impedance controllers.

An alternative approach is taken by Lee et al. [79] and Wang et al. [80] where Hidden Markov Models (HMM) are used to approximate human movement intention. These works focus on the cooperation of a robot with a human, requiring some initial input to produce an output. Lee et al. use visual markers placed on the human to map joint movements, which are then virtually connected to the robot in a spring-damper model. HMM are used to estimate human intention, which is then mapped to the robot to mimic the motion. Wang et al. use the HMM to modify the reference trajectory of an admittance controlled arm to shake hands with a human. They are able to show results of the robot achieving this goal. However, it is apparent that the HMM method is limited in that it requires some human input to work, and is not suitable for both robot-human and robot-robot interaction.

Ge et al. [81] have developed a learning impedance controller for interactions between a manipulator and a non-passive environment, i.e. contact tool tasks such as grinding or drilling, based on the work of Li et al. [82]. The controller is novel in that it does not require a dynamic or inverse kinematic model. Instead, a target impedance model is set, which is then achieved through iterative learning. In addition, the torque-based impedance controller that is developed is capable of very "soft" impedances, i.e. small stiffness and damping, which is not possible with a position-based impedance control. The control system relies on measurement of the external forces acting on the manipulator; however, the effect of a noisy sensor is taken into account. The controller is tested through simulation; results show that the controller is stable with or without interaction force, the former condition reducing the controller to a trajectory tracker. In the latter condition, compliant behaviour is shown as from evident tracking error, due to

interaction force. However, the controller is not proven with a physical robot, so true performance in a realistic environment is uncertain.

2.1.4 Behaviour-based Control

The foundations of behaviour control were laid down by Rodney Brooks in 1985 [83], where an outline of a novel control method was discussed and tested on an early robot platform. Rather than sequential actions, he argued that a more flexible and robust method is to construct the controller from multiple asynchronous modules, which can all receive sensor data and influence the decisions of robot control. tasks can be programmed in a hierarchy, depending on the data available and presented from each module; for example, a wandering robot may have task priorities: 1. avoid objects, 2. explore the environment 3. map the environment 4. find a particular object, etc. From this architecture different behaviours emerge naturally, dependent on the environmental context of the robot and the local state.

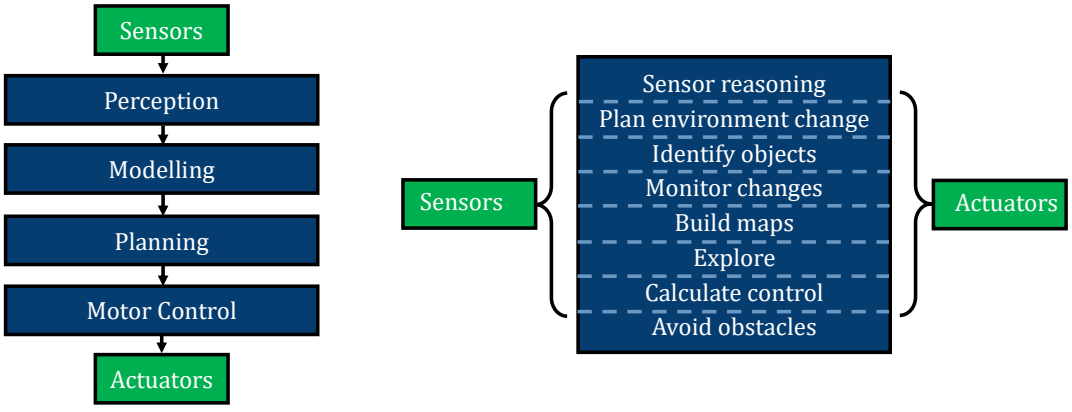

FIGURE 2.5: Decompositions of a traditional control system approach (left) and Brooks' layered control system (right).

The rationale behind this different avenue of thinking came from Brooks' realisation that the research in Artificial Intelligence (AI), based on the classical model shown in Figure 2.5 (left), had started to founder and reach a dead end. Although the classical approach was still relevant in industrial applications where the environment can be carefully controlled and modelled, the linear approach was difficult to apply to a fully unpredictable real-world environment. Brooks argues that this is due to classical AI placing foundations on the abstract manipulation of symbols to represent the environment, whereas "nouvelle" AI emphasises a grounding in the physical environment, i.e. "to build a system that is intelligent it is necessary to have its representations in the physical world . . . the world is its own best model"[84]. Biological models, such as

ANNs, were already starting to appear not only applied to classical problems such as feedback control [47] and inverse kinematics [85], but also to robot behaviour in the natural environment [86, 87] with hierarchical decision making used to generate robot behaviour.

Research has continued to look at how biological models - one of the main themes of behaviour-based control - can be applied to the robotics field. Pfeifer et al. [88] discuss how, in contrast to industrial robots which work in controlled environments with little to no uncertainty, biological organisms have evolved to operate in real-world environment which is massively uncertain. By observing and learning from the biological systems around us, it is argued that deeper developments can be made into the world of AI and robot design. For example, behaviour-based systems modelled using ANNs have been used to imitate the mechanism of switching between walking and swimming in salamanders [89] or locomotion of a climbing worm robot [90]. Increasingly robot control design is looking at a behaviour-based approach as the application moves out of the factory and into homes, where robots will be expected to perform a wide range of tasks for disabled or elderly care. The human environment has many characteristics which make these tasks particularly challenging for designers [91]: typically humans will be present while the robot is operating, which has safety implications for both the robot and users; our environment is designed to be easy for us to use, which may not necessarily be easy for a robot; the environment is dynamic, and is impossible to fully model or predict, and the robot must work in a real time frame similar that of a human in order to interact well. One programming method which seeks to solve these issues is Learning from Demonstration (LfD) [92–94], where some learning algorithm is supplied with data provided by a teacher (which can be a human or even another robot) to mimic how to move and work in the natural environment, without the need for a complex centralised controller. The downside to this, however, is that the robot must then have some generalisation mechanism when required to perform tasks outside of the teaching data.

By composing a set of basic behaviours of action, the work in [95] develops a method for coordinated motions and actions of a humanoid robot. Actions such as reach, grasp, release etc. are built up from action primitives, which are enacted using a singularity-robust inverse kinematics algorithm. However, it is noted that the simulation environment is structured, and no dynamics are considered. Biological inspiration is often explored in the behaviour-based field for dealing with the dynamics problem; for example, semi-passive control has been used in fish swimming robots [96–98] with soft bodies and impulse-driven actuators to simulate flesh and imitate muscles. This is often a much more energy efficient method of movement, as in passive walkers [99–102] where the natural dynamics of the legs provide the majority of the motion and only a small

amount of energy must be injected into the system for continuous movement. This has the drawback of being difficult to control in an exact manner, compared to something like a Zero Order Hold (ZOH) control [103], where manipulation of the dynamics allows the user to place the centre of mass of the robot in an desired position in a stable manner; additionally, the quasi-passive approach relies on operating at the edge of stability.

Williams and Hardy [104] proposed a behaviour-based architecture of a hybrid position/force controller for a robot manipulator. Rather than all modules having access to the inputs and outputs of the system, as described by Brooks and shown in Figure 2.5, in their work interfacing modules are used to perform the communication with the outside world. This is claimed to give better performance, and is also due to commercial robots often having closed controllers with limited interface. Essentially, two behaviours are created from the position and force controllers, with also a motion constraint module which inhibits unwanted motion (moving into singularity, maintains contact with a surface, etc.), which are combined in a drive behaviour before commanding the actuators. In this way the control architecture is somewhere between the classical top-down approach and Brook's distributed doctrine. The force and position controllers themselves are extremely simple, based on a comparison of target and real values and driving against the direction of error. In the work four tasks are tested; for each task the behaviour-based model is re-arranged to suit. Their results are varied, generally showing that the tasks can be completed although with significant errors, as the controller is not able to compensate for system time delays. Benefits of this behaviour-based approach are given as module re-usability, and the ability to perform in uncertain environments with different tool types.

2.1.5 Hybrid controllers

Hybrid controllers are mentioned often in the literature, normally in the form of combining position and force control [105–112], which is essentially, in its simplest form, a task-space position controller and force controller working in parallel, as described in Figure 2.6. This technique was originally developed for robots performing assembly tasks, after it was found that the positional sensors of the time (early 1980's) were not reliable enough for repeated, accurate manual dexterity [105]. Force sensors located at the wrist of the manipulator, however, could offer a low-cost method of sensing object interaction during an assembly task. By controlling the forces at the interaction point a much higher accuracy could be achieved. As such, these controllers are often designed for tasks where a constant contact force is required, such as turning a crank or following a surface. The methods which make up the position and force controllers, shown in

yellow and red in Figure 2.6, can be the same or different, and can be chosen from a wide range of control techniques.

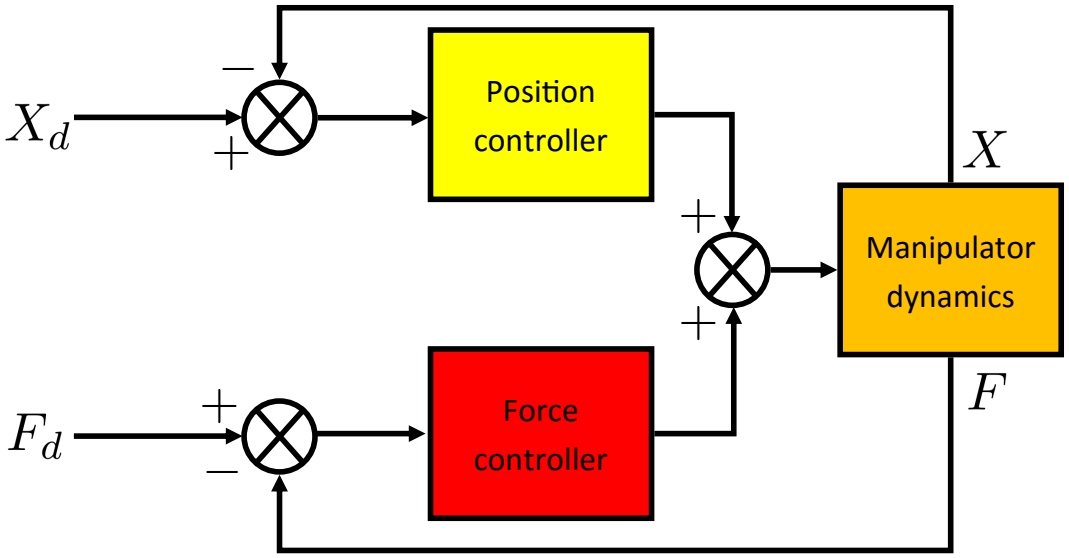

FIGURE 2.6: Conceptual diagram of a hybrid position/force controller.

For example, the work of Yoshikawa et al. [107] uses a method of linearising the manipulator dynamics with respect to the position and force on the object it is in contact with. The input vectors for their controller are given as $\ddot{r}_p$, the acceleration of the end-effector on a constrained hypersurface, and f_F, the force acting on the hypersurface. A simple PI controller is then used for position, and proportional control for the force. Gains were selected through trial and error for both controllers. By taking the arm dynamics into consideration, an improvement was seen in the position control. At the time (1988), they found the computational load of two concurrent controllers was very high, and struggled to perform calculations in real time. In addition, they had to modify the controller depending on the task.

Since then, the position/force hybrid controller has evolved somewhat, and has been used for walking robots [113], where the force with the ground is the constraint; with a neural network controller to determine manipulator stability [114]; with iterative learning of the dynamics of the constraint surface [115]; with fuzzy position control and fuzzy-tuned PI control for a rehabilitation aid [116], and for control of a quadrotor helicopter which maintains contact with the environment [117]. Note that all tasks require contact with the environment, as the force controller is used to maintain the contact while the position controller follows some task trajectory. As a result of this, position/force hybrid control has a limited set of tasks that it is applicable to.

Another application which often uses a hybrid controller is visual servoing, where a camera is used to observe the task-space pose of the robot and the target. There are

two base methods in visual servo control: Image Based Visual Servoing (IBVS) and Position/Pose Based (PBVS) [118]. The IBVS method involves calculating a pose error directly from the captured image, which is then mapped to task-space via the image Jacobian. This is a very robust and flexible method and can be used in a model-free environment, but involves a nonlinear mapping of velocities in the image to the velocities in Cartesian space [119]. The use of the image Jacobian also means that singularities may occur (making the mapping unsolvable) or local minima may be reached, giving sub-optimal solutions. In addition, as the camera trajectory is unpredictable it is not possible to achieve optimal time or energy control. The PBVS method, on the other hand, constructs a 3D model of the environment, which is then used to calculate the motion required to move the robot pose towards the target. Optimal trajectories can be obtained through the environmental model, and is easily transferrable to the robot as the coordinate frames are the same. However, the need for an environmental model makes PBVS inflexible. Sensor noise or uncertainties in the model can cause instability in the system [120], and control failure can occur if the target is lost from the field of view.

Several works have used hybrids of PBVS and IBVS to gain the benefits of each while compensating for the negatives. Tsai et al. [121] use two cameras for visual servoing: one is placed with a view of the total workspace and provides a PBVS-type control, whilst the other is mounted at the end-effector of the robot and gives a IBVS-control. Target objects are recognised using the overhead camera and matched to object models to provide grip-location information. The camera attached to the end effector is used to guide the gripper to the target in the correct orientation. The two methods are combined through a simple task-dependent switching, where the mode is switched when a task is complete and the robot is stationary. This avoids instability caused from both modes working concurrently. Through their hybrid method, experiments showed that the approach performed as well as classical IBVS methods, but without the need for multiple image features to calculate joint velocities. However, their system still suffers from sub-optimal performance normally associated with IBVS methods, and the switching method is rudimentary and arbitrary.

Concurrent IBVS and PBVS control is described in [122] instead of switching, which is slow and suffers from discontinuity when features approach the image border. In this case hybridisation is realised through a probabilistic framework and importance functions to weight control to either method. Performance indices are calculated for IBVS and PBVS: for the former, this is related to the distance of the joints from their limits, and for the latter the index is proportional to the smallest distance of a image feature to the image border. A Lyapunov function is used to prove stability of the controller. Their results show an improved performance over the individual IBVS and

PBVS controllers. Stability is maintained in experiments carried out with a robot, which is one of the main concerns when presenting hybrid visual servoing solutions. However, it is noted by the authors that their hybrid controller is still able to become stuck in local minima solutions, which reduces the performance.

A slightly different hybrid method is demonstrated by Ye et al. [119], where data from a IBVS scheme is translated to Cartesian and Polar based methods. The reasoning behind this is to improve IBVS method during large rotations, which in the classic case can cause large and unstable camera movements. The task is decomposed before operation, with the Cartesian and polar controllers being assigned to different sections of the task. Rotational movements are carried out using the polar form, with all other movements carried out with the Cartesian method. Simulations are carried out with a free-flying camera focusing on a rectangle formed of four points. They show that, by working in the two coordinate systems depending on the required motion, convergence is much quicker than the individual methods, and controller stability is improved during rotations. The task decomposition, however, requires observation of the target and calculation before execution, which would not work in a constantly changing environment.

2.2 Fuzzy Control

For a long time after the introduction of fuzzy logic set theory by Zadeh [123] it was considered inconceivable to apply the vagueness of the technique to the field of control engineering. Not until the work of Mamdani [124] was fuzzy logic successfully applied to a simple dynamic plant, followed by the first industrial application in a cement kiln a few years later [125]. In the years since it has been applied in trains, vacuum cleaners, camera focusing, washing machines and much more. Due to the nature of fuzzy logic, it is most successful in systems where:

- No models exist for the system, so cannot be mathematically approximated
- Expert operators are capable of expressing control of the system in linguistic descriptions.

Artificial Neural Networks (ANNs) are also capable of modelling complex non-linear systems but typically require long training time with large data sets before they are useful; fuzzy controllers have been shown to outperform ANNs in some tasks [126]. Genetic or evolutionary algorithms (GAs) can also be implemented for similar tasks, but suffer from poor scaling with complexity and can have a tendency to converge to local minima. There are many examples in the literature of fuzzy control being used in

conjunction with genetic algorithms, where the rule sets for the fuzzy control are evolved based on data sets. One of the earliest works combining fuzzy logic control with GAs by Thrift [127] showed that due to the discrete nature of fuzzy strategies it is possible to use GAs to learn the fuzzy rules of a system. Similarly, the work of Park et al. [128] and Herrera [129] shows that genetic algorithms can help to improve the performance of a fuzzy controller through non-subjective selection of parameters such as the membership functions.

In terms of robotics, early usage of fuzzy control was for decision making in mobile robots [130], where three subsystems were implemented: visual sensors, motor drive and a drive expert system for management. The use of inference from fuzzy engines was one of the first steps towards a more "soft" artificial intelligence rather than the previously dominating "good old fashioned artificial intelligence" (GOFAI) approach. More recently the work of Kim et al. [131] used fuzzy logic to approximate the nonlinear dynamics of a robot equipped with only joint position sensors. They were able to show that all signals in the closed loop system of the robot, observer and controller are uniformly ultimately bounded. They were able to achieve - using only position information - good tracking performance and robustness against payload uncertainty and environmental disturbance.

There are several instances of fuzzy control being used in conjunction with Sliding Mode Control (SMC) to eliminate chattering [132–135], mimic a feedback linearisation control law [136, 137], approximate unknown system functions of the SMC [138], estimate unavailable states of a MIMO nonlinear system [139–142] and compensate for uncertain dynamic time delays [143–145]. As is obvious from the literature, fuzzy logic and its use in control is used in a variety of applications and conjointly with other well defined control schemes.

2.3 Control and Coordination of Bimanual Manipulators

The subject of bimanual control is fairly young in the literature; most early research involving bimanual control was aimed at rehabilitation devices, and still is [146–149]. However, bimanual robots have started to be explored in the field of manipulation. For example, the work of Sugar and Kumar [150] describes a leader-follower type configuration, where the follower robot(s) maintain coordination with the leader through a compliant position controller, similar to the one described by Hogan [8], combined with the passive compliance inherent in spring-damper manipulators. They show that good cooperation can be obtained through this method, with several tasks demonstrated using multiple robots being performed well.

Gribovskaya and Billard's work [151] combine dynamic system movement with a Programming by Demonstration (PbD) method to create a controller to attempt to maintain spatial constraints, i.e. keeping the arms within a relation to each other, and temporal, i.e. the arms must be synchronous. The results of their tests showed that the constraint of keeping the arms within a spatial parameter severely limited the workspace of the robot, and although it was shown that the controller could manage high frequency, low amplitude perturbations, a large position error can make the robot react with jerky movements.

Edsinger and Kemp [152] developed a behaviour-based control for their robot, Domo [153]. Domo is equipped with series elastic actuators [154], which impart a passive compliance to the robot. It is apparent, however, that a behaviour-based system has limitations, in that the robot can only react to situations that have been pre-programmed, and cannot generalise to uncertain tasks.

Hwang et al. [155] have developed a motion planning method for bimanual assembly tasks. Their controller separates the assembly task into sections: approach, contact and assembly, which apply different controllers. For example, in the approach phase the velocities of each manipulator are proportional to the distance to target, increasing the accuracy of the movement and reducing force on contact. The computation is distributed across both manipulators, with a separate high level controller to ensure cooperation. Their results show that their motion planner is capable of assembly tasks, but shows limitations to only these types of tasks due to the behaviour-type control.

The work of Luo et al. [156] describes a method for bimanual object handling that controls the sum of forces acting upon an object to perform smooth, fast movements. The force applied to the object is subject to the constraints

$$\begin{aligned} max(F_{int1}(t)) < F_{MAX1} \\ max(F_{int2}(t)) < F_{MAX2} \end{aligned} \tag{2.15}$$

and

$$\begin{aligned} min(F_{int1}(t)) > F_{min1}(\mu_1) > 0 \\ min(F_{int2}(t)) > F_{min2}(\mu_2) > 0 \end{aligned} \tag{2.16}$$

where μ_1 and μ_2 are the friction coefficients at the points of contact. Movement of the object is a resultant of these two internal forces F_{int1} and F_{int2}, i.e. if $F_{int1} \neq F_{int2}$ a driving force is produced. $F_{int1} + F_{int2}$ is the total force acting on the object, and $F_{int1} = F_{int2}$ will mean that object velocity will be zero, not necessarily that the force on the object is zero. It is argued that is not possible to control the internal force with position

control, and that some force feedback is required. In addition, it is suggested that in order to consider a maximum and minimum force constraint the frequency bandwidth of each robot's position control loop should be adjusted depending on the dynamics of the held object. Their simulations show that their approach was capable of modelling the internal dynamics of the object and controlling two manipulators to move the object while maintaining force contact.

The iCub robot has been developed at IIT as part of the EU RobotCub project, and is now in use by more than twenty universities worldwide [157]. It is a child-size humanoid robot with 53-DoF, including head/eyes/neck, torso, arms/hands and legs. The proximal joints are tendon driven by harmonic drive motors, which allow low-level access to stiffness and damping control of each joint for the first five joints in each arm (see Figure 2.7). The remaining joints, including the two at the wrist, are driven with standard gears and have low torque with no impedance control available. A 6-DoF force/torque sensor is located in the upper arm, with which it is possible to calculate wrenches along the full kinematic chain, using a recursive Newton-Euler algorithm [158]. However, to detect external forces correctly it is necessary to know where the contact point is. In earlier versions of the iCub there was no way of doing this so a fixed point at the junction between wrist and arm was defined, but the development of an electronic skin [159], visible in Figure 2.7, allows the dynamics algorithm to correctly calculate external disturbances if it is applied to the skin. Despite this, from personal experience it is not straightforward to use the iCub for bimanual control experiments, or any experiments requiring high torques along the length of the arm. The inverse kinematic libraries provided by the development team incorporate minimum-jerk trajectory planners, which makes real-time control difficult. The joint limits of the arms constrict the shared workspace, as each arm cannot reach past the midpoint of the robot. It is also not possible to hold heavy or moving objects in the hands, as the wrist joints are very weak due to being tendon driven, and do not have low level impedance control available. This meant it was not possible to carry out the experiments detailed in this work using the iCub as the manipulator platform.

Developed at the German Aerospace Centre (DLR), Justin is a humanoid robot with two 7-DoF DLR-LWR-III manipulators for the arms[160]. It has been designed specifically for research in bimanual manipulation in human working environments. The manipulators include integrated joint-level torque sensors [161], which allows implementation of torque and impedance controllers. The arms also feature opposing hands, also integrated with force/torque sensors. For bimanual impedance control, they have used the Intrinsically Passive Controller (IPC) [162] which was specifically designed for use with multi-fingered robot hands. They implement the cartesian impedance using three virtual springs (see Figure 2.8); springs K_l and K_r connect the desired $H_{(\cdot),d}$ and actual

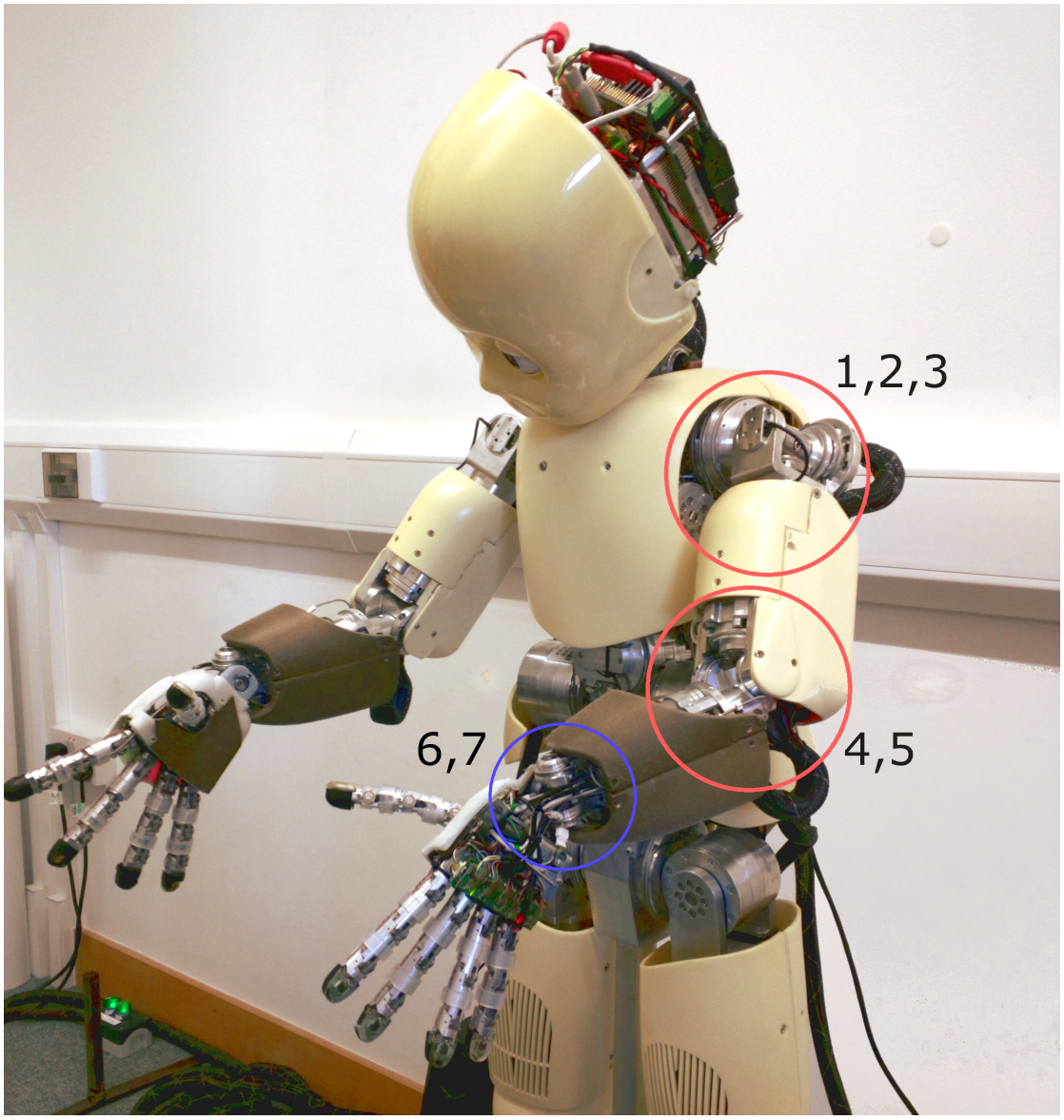

FIGURE 2.7: The iCub robot with harmonic drive joints labelled. Wrist joints 6 and 7 are circled in blue. The remaining nine joints which form the arm actuate the fingers. The electronic skin can be seen applied to the forearms.

positions $H_{(\cdot),a}$ of the hands, and the third spring K_c creates the compliant coupling. In this configuration the natural spring lengths and constants must be carefully selected to avoid interference. With this controller structure they are able to perform different behaviours including independent control when the coupling spring is set to zero. This control structure is made possible through the highly advanced manipulators with joint-level torque sensors, which give a rich data source of environmental and manipulator dynamics.

The Baxter robot from ReThink Robotics is comprised of a bimanual pair of 7-DoF manipulators equipped with Series Elastic Actuators (SEAs)[163] and end-effector cameras. The design is aimed at providing a low-cost industrial robot solution which can work

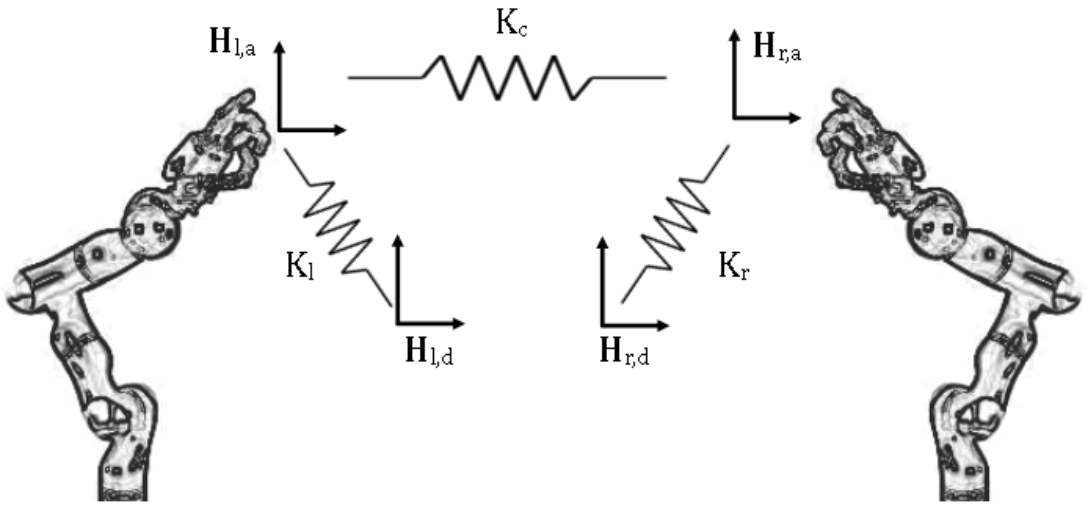

FIGURE 2.8: Bimanual Cartesian impedance control implemented on Justin robot

with humans in a low-volume production line[164, 165]. To stimulate academic interest a Software Development Kit (SDK) has also been released, resulting in several universities creating experiments using the Baxter [166–168]. Through the SEAs, a block diagram of which is seen in Figure 2.9 it is possible to perform rudimentary force sensing through spring deformation; in the simplest sense, given a joint angle q and link angle θ, with a known spring constant K the torque at the joint τ can be found

$$\tau = K(q - \theta). \tag{2.17}$$

This allows for force control schemes to be applied, which forms one of the selling points of Baxter; simple pick-and-place tasks can be easily programmed using a lead-by-nose approach. The arms have wide joint limits and good length which allows the workspace of both to overlap well in front of the robot. The software bundled with the Baxter includes a rudimentary scheme for avoiding collisions between the arms and environmental objects [169]. For obstacle avoidance, a torque observer records an impact when a sudden change in torque or when torque across a joint increases beyond a threshold. To avoid collisions with its own arms, an internal model of the robot is updated with positional information, which is then used to trigger a collision avoidance (i.e. stop the arm) when the arms are too close to each other. The design is also useful for research, as it provides two manipulators with joint-level torque sensing for less than a tenth of the price of a single comparable arm in terms of DoF and payload.

As discussed, many current approaches to bimanual cooperation involve a behaviour based approach. This is very limiting in terms of ability, as the robot is constrained

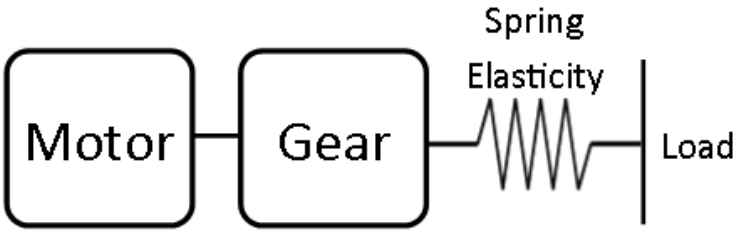

FIGURE 2.9: Block diagram of SEA actuator.

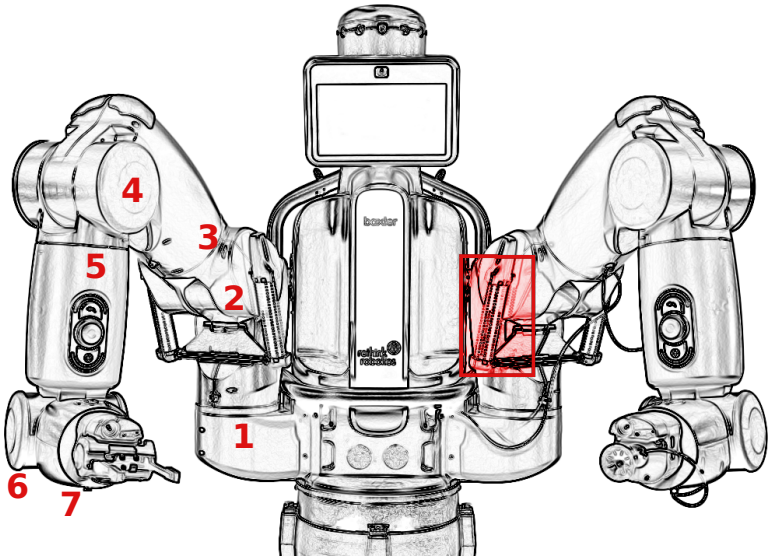

FIGURE 2.10: Diagram of the Baxter robot. Joints *1, 2, 3* comprise the shoulder, *4 and 5* the elbow, and *6, 7* the wrist. The spring highlighted in *red*, of which two are attached to each arm, provide part of the gravity compensation.

by the scope of the programming. The work of Luo et al.[156] shows promising results but needs to be explored more for stability in different situations, e.g. moving objects with internal dynamics (such as a container of liquid). Their method also requires the use of force sensors, which are expensive and often produce a noisy output; ideally the controller would not require force measurement. This is also a drawback in the work with the Justin robot, which although has impressive capabilities, it is also prohibitively expensive for most non-research applications. This problem has been negated in the Baxter robot, through the use of SEAs, although this relies on accurate spring models.

2.4 Concluding remarks

This chapter has examined a wide range of works related to this study.

In terms of robot control, there is a choice of following the classical or behaviour-based approach. Classical techniques have a long history and deep research base, meaning stability analysis is well documented for a wide range of different control structures, which are applicable for various tasks. The behaviour-based paradigm is comparatively new, although it also has been the subject of much research in the robotics community. Most of the applications where a behaviour-based approach has been used are for fully autonomous systems, where the parallel architecture means a large number of different sensors can be used to produce a wide range of behaviours. At the low level control in a behaviour-based system classical control algorithms such as PID control are commonly used for actuation of different behaviours. In the context of this work, where only one task is required, a behaviour-based architecture is not required and a classical monolithic controller is sufficient, to avoid over-complication and obfuscation.

In terms of biomimetic control, of particular interest is the work of the Burdet, Ganesh, Milner group; their attention to the biomimetic design of controllers, derived directly from experimental results in neuroscience, show a promising direction for a more natural and stable cooperation method. It is intuitive to say that the best type of control for interacting with humans would be one that mimics human control the closest; which then leads to the suggestion that this may also be the best method for human-robot and robot-robot cooperation. The cooperation control suggested by Maeda et al. show a reduction in surplus energy transfer, but it is only suitable for fixed trajectories. The HMM method can produce human-like control but requires the input of a human, which limits its application to tasks where an operator is involved. The biomimetic controller developed by Ge et al. could be a good controller for the required task problem, but at the time of writing had only been verified through basic simulations and not in a real-world setting. The Burdet/Ganesh controller has, in comparison, been applied to several robots to verify stability and the similarity to human impedance control.

Different hybrid control methods have been explored, the most common of which is the position/force hybrid control. Although this is shown to attain the benefits of each scheme while counteracting the negatives, the position/force hybrid is only applicable to tasks where a constant force is maintained between the manipulator and the environment. This makes it unsuitable for the desired task. During investigation into visual servoing, where hybrids controllers are often employed, it can be noted that the components of the hybrid can be switched between (depending on context) or combined via some weighting mechanism. Switching is slower to execute as the task must be initially

decomposed, but it is easier to implement and more stable than concurrent hybridisation. The weighting method improves stability of concurrent hybrids, by using some indices to create a probabilistic meshing of the two control commands. In the terms of this work, switching is not applicable as the task must be carried out smoothly and without delay, especially if in the context of working with a human. Additionally, a different type of hybridisation in visual servo control was noted; switching between different space representations depending on the task. In the IBVS context this was between Cartesian and polar coordinate spaces, but in the context of this work it opens the idea of working in both task-space and joint-space, as the expected disturbances are significant in both spaces individually.

In the literature surrounding fuzzy control there are many applications where it is used in conjunction with other control methods to improve performance, but rarely to estimate system parameters which would normally be set through trial and error. However, the method of application for Mamdani-type fuzzy controllers lends itself well to this task; expert knowledge, which is applicable in this case, can be applied through linguistic rules to create the control structure. In comparison ANN methods, which have also been used in this context, require a large training set to be collected and applied to allow the network to learn ideal gains. This can also suffer from local minima problems and requires various tuning for best performance, which is the problem in the first place. Additionally, ANNs can require large computational power (depending on the size of the network) which may limit real-time capability, compared to fuzzy algorithms which are composed of only a few simple addition and multiplication operations.

Compared to other platforms such as the iCub robot (not robust enough for impedance control experiments) or the Justin DLR (prohibitively expensive for common use), the Baxter robot is the most suitable for performing bimanual control experiments in this case. Good arm manoeuvrability and large workspace allows a wide range of tasks to be carried out, and the workspace overlap means it is suitable for bimanual tasks. The SEAs which drive the Baxter joints mean it can be controlled directly through torque commands, as well as providing joint level torque readings which is useful for performance analysis. In terms of bimanual control methods, a master-slave approach gives good flexibility and applicability, compared to the behavioural-types used by Edsinger and Kemp or Hwang, or the summing method described by Luo et al.

The following chapter describes the dynamic model of the Baxter robot and verification of the chosen biomimetic controller in simulation, chosen based on decisions made in this chapter. Later chapters make use of the weighting method for hybridisation of control in different space representations, based on the review of methods that have been used in visual servo control. Investigation into fuzzy systems provides the inspiration for

the automatic tuning described in Chapter 5, and review of current bimanual control methods forms the basis of the master-slave approach used in Chapter 6.

Chapter 3

Biomimetic Adaptive Controller Development

3.1 Introduction

Modern robots are frequently expected to interact with humans and the environment [170, 171]. This interaction, which is highly dynamic and often unknown, requires control methodologies which maintain stability and effectiveness within the task despite heavy disturbance. One of the first schemes proposed to control interaction with an unknown environment is impedance control [172]. The environment is modeled as an admittance and the manipulator as an impedance, so that interactive control is achieved through the exchange of energy.

Adaptive control methods can be combined with impedance control to compensate for parametric uncertainties [173–175] and improve performance. Adaptive impedance control methods such as those developed in [16, 176, 177], have improved the operational performance of a traditional impedance controller and demonstrate the stability of such controllers. In particular, the work in [16] shows how stability and successful performance can be gradually acquired in tasks where the initial instability is high, such as in tool tasks like carving or drilling [178].

Parallel studies in the biological field show that the human nervous system can adapt the mechanical impedance (i.e. the resistance to perturbations) of the arm to improve performance of tasks in stable and unstable environments [179, 180]. This is achieved through co-contraction of agonist/antagonist muscle groups, described in Figure 3.1. Motor commands are adapted by the CNS to stabilise interactions by independently

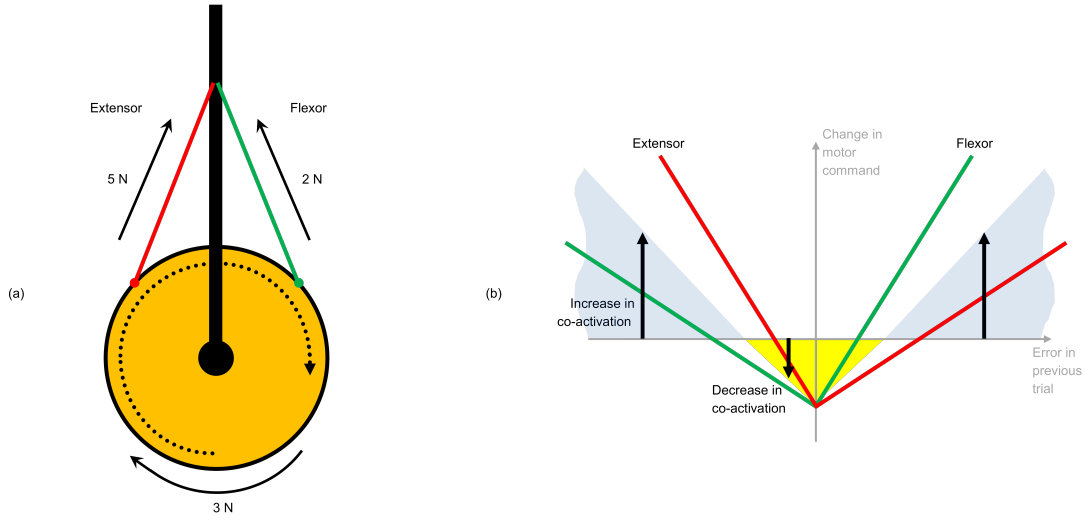

FIGURE 3.1: How co-contraction affects muscle impedance. (a): By contracting at the same time with different forces, the flexor and extensor muscles work together to maintain effector torque, but with increased impedance. (b): the "v-shape" of the adaptive law. Impedance increases irrespective of error direction, and decreases when error is below a threshold; this mechanism ensures minimisation of metabolic cost (i.e. control effort).

controlling arm impedance and force; suitable muscle activations are automatically selected to compensate for the instability and interaction force. Concurrently metabolic cost is minimised through a natural relaxation of muscle groups when the task is being sufficiently performed. A model for this, introduced in [181, 182], has given rise to a novel non-linear adaptive controller that has been successfully applied to a robot context [183]. The adaptation of impedance in this biomimetic controller follows a "v-shaped" algorithm, demonstrated in Figure 3.1(b). Conventionally, adaptive control designs focus on estimation of uncertain parameters under stable motion [30, 184–187]; in comparison, the biomimetic design is able to acquire stability in unstable dynamics as well as minimise control effort, through the adaptation of force and impedance [16]. During stable interactions compliant control, receiving much interest in recent research [112, 188–194], is also demonstrated, due to the similarity to the muscle relaxation effect within the controller design.

3.2 Baxter Dynamics and Experimental Verification

This section aims to cover the common elements of manipulator dynamics and useful methods. This forms a reference for the remaining chapters, describes the techniques used regularly during system modelling and specifically details the parameters needed to form the dynamics of the Baxter robot, used most often in this work for simulation.

This is useful in many ways: for the design of motion control systems, analysis of mechanical design, simulation of manipulator motion, etc. Many control algorithms, such as computed torque control [195], predictive control [196, 197] and sliding mode control [198–201] normally require an accurate model of the manipulator dynamics, commonly in the form:

$$M(q)\ddot{q} + C(q,\dot{q}) + G(q) = \tau + \tau_{ext} \tag{3.1}$$

where q denotes the vector of joint angles, $M(q) \in \Re^{n\times n}$ is the symmetric, bounded, positive definite inertia matrix, and n is the degree of freedom (DoF) of the robot arm; $C(q,\dot{q})\dot{q} \in \Re^n$ denotes the Coriolis and Centrifugal force; $G(q) \in \Re^n$ is the gravitational force, $\tau \in \Re^n$ is the vector of actuator torques and $\tau_{ext} \in \Re^n$ is the vector of torques resulting from external forces. In this form the kinetic energy of the manipulator is described within $M(q)\ddot{q} + C(q,\dot{q})$, and the potential energy represented in the gravity term $G(q)$. This can then be used to calculate either the forward dynamics (useful for simulation), where the manipulator motion is calculated based on a vector of applied torques, or the inverse dynamics (useful for control design) where the torques for a given set of joint parameters can be calculated.

There are two commonly used methods for formulating the dynamics in Equation 3.1: the Lagrange-Euler (L-E)[202] formulation and the Recursive Newton-Euler (RN-E) method [203]. Both are equivalent, as they both describe the dynamic behaviour of the robot motion, but are specifically useful for different purposes.

The L-E method is simple and systematic, used to calculate the kinetic and potential energies of a rigid body system. Bajeczy [204, 205] showed that the equations of dynamic motion for a robot manipulator are highly non-linear and dependent on the link physical parameters and configuration (i.e. position, angular velocity and acceleration). The L-E dynamics provide the *closed* form of the robot dynamics, and is therefore applicable to the analytical computation of robot dynamics [206], which allow a designer to create and analyse possible control strategies.

Forward and inverse dynamic calculation may be achieved with the L-E method, but this requires computation of a large number of coefficients in the inertial and coriolis terms from Equation 3.1, which can be computationally expensive depending on the number of joints. This can make the L-E method unusable when real-time control is required. Other methods, such as RN-E (described in the next section), or Lee's Generalised d'Alembert Equations (GAE) [207] produce faster derivations [203]. A recursive L-E method has also been described [208] which reduces the computational weight of the L-E formulation to a level similar to that of RN-E.

The N-E formulation is formed by examining all the forces acting on each link of the manipulator, which then forms a set of equations with a recursive solution [209]. A forward recursion propagates link positions, velocities and accelerations from the base frame to the end-effector, then a backward recursion propagates the forces and torques from the last link back along the link structure. This was developed to be a more efficient method than L-E, based on the principle of the manipulator being a kinematic chain; when a force is applied to one link, the resulting forces and motion will produce a force on the next or previous link. Due to this effect there may be repetitions of calculations [210] which can be removed if expressed recursively. This reduction can greatly reduce time needed to compute forward and backward dynamics, allowing them to be calculated in real time.

3.2.1 Lagrange-Euler Formulation

The Lagrange-Euler equations of motion for a conservative system [203] are given by

$$\begin{aligned} L &= K - P, \\ \tau &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} \end{aligned} \tag{3.2}$$

where K and P are the total kinetic and potential energies of the system respectively, $q \in \Re^n$ is the generalised robot coordinates, and τ is the generalised torque at the robot joints [203]. The kinematic and potential energies are given by:

$$\begin{aligned} K &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=1}^{i}\left[\mathrm{Tr}\left(U_{ij}\ J_i\ U_{ik}^T\right)\dot{q}_j\dot{q}_k\right] \\ P &= \sum_{i=1}^{n} -m_i\ \mathbf{g}\left({}^0T_i\ \bar{r}_i\right) \end{aligned} \tag{3.3}$$

which, when substituted into Equation 3.2, gives the expression:

$$\begin{aligned} \tau_i &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} \\ &= \sum_{j=i}^{n}\sum_{k=1}^{j}\mathrm{Tr}(U_{jk}J_jU_{ji}^T)\ddot{q}_k \\ &+ \sum_{j=i}^{n}\sum_{k=1}^{j}\sum_{m=1}^{j}\mathrm{Tr}(U_{jkm}J_jU_{ji}^T)\dot{q}_k\dot{q}_m \\ &- \sum_{j=i}^{n} m_j\mathbf{g}U_{ji}\bar{r}_j. \end{aligned} \tag{3.4}$$

This can be expressed more simply in the form given in Equation 3.1, as a sum of the inertia, Coriolis/centrifugal and gravity terms. The elements of the symmetric matrix $M(q)$ are given by

$$M_{i,k} = \sum_{j=\max(i,k)}^{n} \mathrm{Tr}(U_{jk}\, J_j\, U_{ji}^T) \quad i,k = 1,2,\ldots n, \tag{3.5}$$

the Coriolis/centrifugal force vector $C(q,\dot{q})$

$$\begin{aligned} C_i &= \sum_{k=1}^{n}\sum_{m=1}^{n} h_{ikm}\dot{q}_k\dot{q}_m \\ h_{ikm} &= \sum_{j=\max(i,k,m)}^{n} \mathrm{Tr}(U_{jkm}\, J_j\, U_{ji}^T) \end{aligned} \tag{3.6}$$

and the gravity vector $G(q)$

$$G_i = \sum_{j=i}^{n}(-m_j \mathbf{g} U_{ij} \bar{r}_j) \tag{3.7}$$

where $\mathbf{g} = [0,\ 0,\ -9.81,\ 0]$ is the gravity row vector. The matrix U_{ij} is the rate of change of points on link i relative to the base as the joint position q_j changes

$$U_{ij} \equiv \frac{\partial T_i^0}{\partial q_j} = \begin{cases} {}^0T_{j-1}\, Q_j\, {}^{j-1}T_i & j \le i \\ 0 & j > i \end{cases} \tag{3.8}$$

which allows derivation of the interaction effects between joints, U_{ijk}

$$\begin{aligned} U_{ijk} &\equiv \frac{\partial U_{ij}}{\partial q_k} \\ &= \begin{cases} {}^0T_{j-1}\, Q_j\, {}^{j-1}T_{k-1}\, Q_k\, {}^{k-1}T_i & i \ge k \ge j \\ {}^0T_{k-1}\, Q_k\, {}^{k-1}T_{j-1}\, Q_j\, {}^{j-1}T_i & i \ge j \ge k \\ 0 & i < j \text{ or } i < k \end{cases} \end{aligned} \tag{3.9}$$

where, for revolute joints,

$$Q_j = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.10}$$

The J_i matrices are independent of link position or motion, and therefore only needs to be calculated *once* from the inertia tensors, link masses and link centre of mass:

$$J_i = \begin{bmatrix} \frac{-I_{xxi}+I_{yyi}+I_{zzi}}{2} & I_{xyi} & I_{xzi} & m_i\bar{x}_i \\ I_{xyi} & \frac{I_{xxi}-I_{yyi}+I_{zzi}}{2} & I_{yzi} & m_i\bar{y}_i \\ I_{xzi} & I_{yzi} & \frac{I_{xxi}+I_{yyi}-I_{zzi}}{2} & m_i\bar{z}_i \\ m_i\bar{x}_i & m_i\bar{y}_i & m_i\bar{z}_i & m_i \end{bmatrix} \tag{3.11}$$

This concludes the calculations required to form the L-E dynamics.

3.2.2 Recursive Newton-Euler Formulation

As mentioned in Section 3.2, the N-E formulation of manipulator dynamics is derived from the motion of each joint within its reference frame, which are then propagated along the kinematic chain. Part of the computational speed-up comes from only requiring the rotation matrix $R \in \Re^{3\times3}$, which also has the property of ${}^{i}R_{i+1}^{-1} = {}^{i+1}R_i = {}^{i}R_{i+1}^{T}$, i.e. invertible via the transpose. The mass m, moments of inertia I, location of the centre of mass $\mathbf{c}$, and orientation of the body are related to the motion of the load and the forces and torques exerted on it through the Newton-Euler equations. The net force ${}_q\mathbf{f}$ and net torque ${}_q\mathbf{n}$ acting on the load at the centre of mass are:

$$\begin{aligned} {}_q\mathbf{f} &= \mathbf{f} + m\mathbf{g} = m\ddot{\mathbf{r}} \\ {}_q\mathbf{n} &= \mathbf{n} - \mathbf{c}\times\mathbf{f} = {}_q\mathbf{I}\dot{\omega} + \omega \times ({}_q\mathbf{I}\omega) \end{aligned} \tag{3.12}$$

where $\mathbf{f} \in \Re^{3\times1}$ is the force at the end effector, $\mathbf{g} \in \Re^{3\times1}$ is the gravity vector, $\ddot{\mathbf{r}} \in \Re^{3\times1}$ is the acceleration of the centre of mass of the end effector, $\mathbf{n} \in \Re^{3\times1}$ the torque at the end effector, $\omega \in \Re^{3\times1}$ the angular velocity vector and $\dot{\omega} \in \Re^{\mathbf{3\times1}}$ angular acceleration vector.

The dynamics calculations are then split into forwards and backwards recursions. First, the angular velocities/accelerations and linear accelerations are calculated in the forward recursions as in Equation 3.13. Starting from a static system, the initial conditions are

given as $\omega_0 = \dot{\omega}_0 = [0,\ 0,\ 0]^T$ and $\dot{v}_0 = {}^0R_1\ \mathbf{g}$.

$$\begin{aligned}
\omega_i &= {}^iR_{i-1}(\omega_{i-1} + z\ \dot{q}_i) \\
\dot{\omega}_i &= {}^iR_{i-1}\left(\dot{\omega}_{i-1} + \omega_{i-1} \times (z\ \dot{q}_i) + z\ \ddot{q}_i\right) \\
\dot{v}_i &= \dot{\omega}_i \times p_i^* + \omega_i \times (\omega_i \times p_i^*) + {}^iR_{i-1}\ \dot{v}_{i-1} \\
\hat{v}_i &= \dot{\omega}_i \times r_i + \omega_i \times (\omega_i \times r_i) + \dot{v}_i \\
F_i &= m_i\ \hat{v}_i \\
N_i &= I_i\ \dot{\omega}_i + \omega_i \times (I_i\ \omega_i).
\end{aligned} \tag{3.13}$$

The backward recursions in Equation 3.14 start at the end effector from link n, where the initial conditions of f_i and n_i are set depending on the load applied on the end effector; this can be set to zero for no load. In addition, when $i = n$, the rotation matrix ${}^iR_{i+1}$ is the identity matrix.

$$\begin{aligned}
f_i &= {}^iR_{i+1}\ f_{i+1} + F_i \\
n_i &= {}^iR_{i+1}\left(n_{i+1} + \left({}^{i+1}R_i\ p_i^*\right) \times f_{i+1}\right) + (p_i^* + r_i) \times F_i + N_i.
\end{aligned} \tag{3.14}$$

Finally, the torque at the joints τ can be calculated

$$\tau_i = n_i^T\ ({}^iR_{i-1}\ z) \tag{3.15}$$

where $z = [0,\ 0,\ 1]^T$ and $p_i^* = [a_i,\ d_i \sin(\alpha_i),\ d_i \cos(\alpha_i)]^T$.

3.2.3 The Baxter Robot

As the Baxter Robot, described in Section 2.3, appears to be suitable for the proposed tasks it will be necessary to obtain a computational model for the design and evaluation of control algorithms. This section describes the parameters required for L-E or RN-E calculation, which are then verified experimentally using data collected from the Baxter robot.

The Denavit-Hartenberg (D-H) parameters and link masses of the Baxter manipulator are given in Table 3.1 and are taken from the Universal Robot Descriptor File (URDF) [211], set by ReThink robotics. These parameters describe the configuration of the links, and form the basis of the L-E formulation. The standard homogenous link transform

matrices are formed from the D-H parameters as such:

$$ {}^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.16} $$

and the transform from the manipulator base to link i can be calculated through ${}^0T_i = {}^0T_1\ {}^1T_2 \ldots {}^{i-1}T_i$.

TABLE 3.1: D-H parameters of the Baxter robot.

Link	θ	d (m)	a (m)	α (rad)	m (kg)
1	θ_1	0.2703	0.069	$-\pi/2$	5.70044
2	θ_2	0	0	$\pi/2$	3.22698
3	θ_3	0.3644	0.069	$-\pi/2$	4.31272
4	θ_4	0	0	$\pi/2$	2.07206
5	θ_5	0.3743	0.01	$-\pi/2$	2.24665
6	θ_6	0	0	$\pi/2$	1.60979
7	θ_7	0.2295	0	0	0.54218

The inertia tensors of each joint are given in Table 3.2, represented by the inertias working in each axis I_{xx}, I_{yy}, I_{zz} and cross-talk inertia between axes I_{xy}, I_{yz}, I_{xz}. Here it is represented as a row vector, but is also commonly found in $I^{3\times3}$ symmetric matrix form.

TABLE 3.2: Link inertia tensors (all units kg $\cdot$ m^2)

Link	I_{xx}	I_{yy}	I_{zz}
1	0.0470910226	0.035959884	0.0376697645
2	0.027885975	0.020787492	0.0117520941
3	0.0266173355	0.012480083	0.0284435520
4	0.0131822787	0.009268520	0.0071158268
5	0.0166774282	0.003746311	0.0167545726
6	0.0070053791	0.005527552	0.0038760715
7	0.0008162135	0.0008735012	0.0005494148
Link	I_{xy}	I_{yz}	I_{xz}
1	-0.0061487003	-0.0007808689	0.0001278755
2	-0.0001882199	0.0020767576	-0.00030096397
3	-0.0039218988	-0.001083893	0.0002927063
4	-0.0001966341	0.000745949	0.0003603617
5	-0.0001865762	0.0006473235	0.0001840370
6	0.0001534806	-0.0002111503	-0.0004438478
7	0.000128440	0.0001057726	0.00018969891

The centre of mass (CoM) for each link is given in Table 3.3, which forms the homogenous column vector $\bar{r}_i = [\bar{x}_i\ \bar{y}_i\ \bar{z}_i\ 1]^T$. These are the complete parameters required for

TABLE 3.3: Centre of mass (all units in m)

Link	$\bar{x}$	$\bar{y}$	$\bar{z}$
1	-0.05117	0.07908	0.00086
2	0.00269	-0.00529	0.06845
3	-0.07176	0.08149	0.00132
4	0.00159	-0.01117	0.02618
5	-0.01168	0.13111	0.0046
6	0.00697	0.006	0.06048
7	0.005137	0.0009572	-0.06682

calculation of the dynamics of Baxter, and can be used for verification of the L-E and RN-E methods.

3.2.4 Experimental Verification

To collect reference data from the Baxter robot, a PID position controller is employed in a double loop configuration, as shown in Figure 3.2. A reference position q_r generates the outer loop:

$$e = q_r - q, \quad \dot{e} = \frac{\mathrm{d}}{dt}e$$
$$\dot{q}_r = K_p e + K_d \dot{e} \tag{3.17}$$

which is then used to generate the inner loop:

$$\dot{\epsilon} = \dot{q}_r - q_r, \quad \epsilon = \int \dot{\epsilon}\, \mathrm{d}t$$
$$\tau_r = K_p \dot{\epsilon} + K_i \epsilon. \tag{3.18}$$

The trajectories were created in two ways: generated using sinusoidal patterns or using

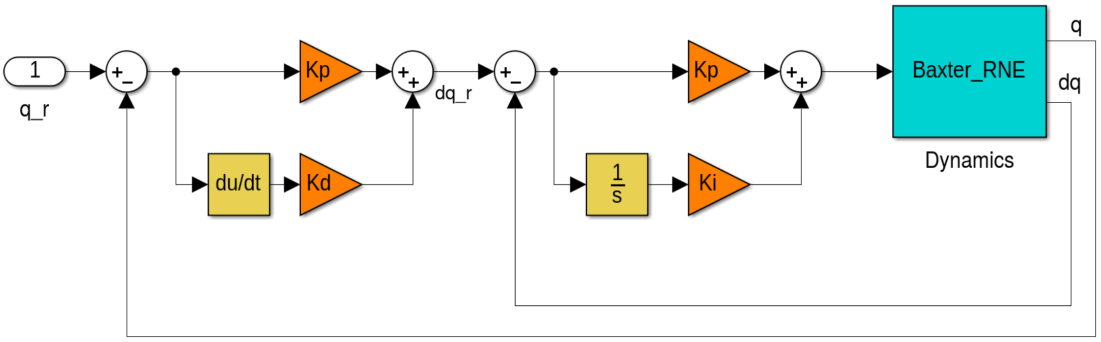

FIGURE 3.2: Block diagram of torque control system.

a touchpad with human-controlled input, both in Cartesian space. Inverse kinematics

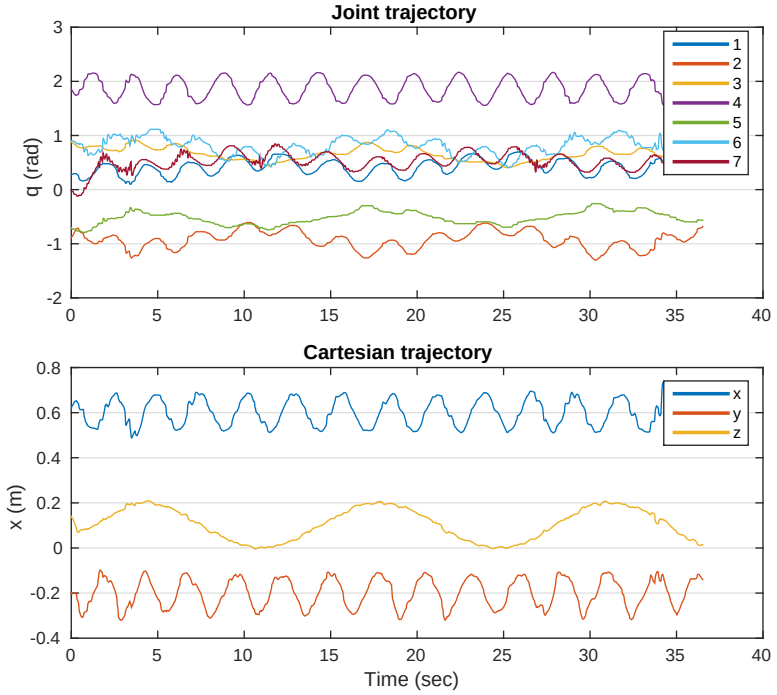

FIGURE 3.3: Test trajectory 1, all dimensions following a sinusoidal pattern with periods of 2.5 seconds in x and y axes, and 15 seconds in the z-axis.

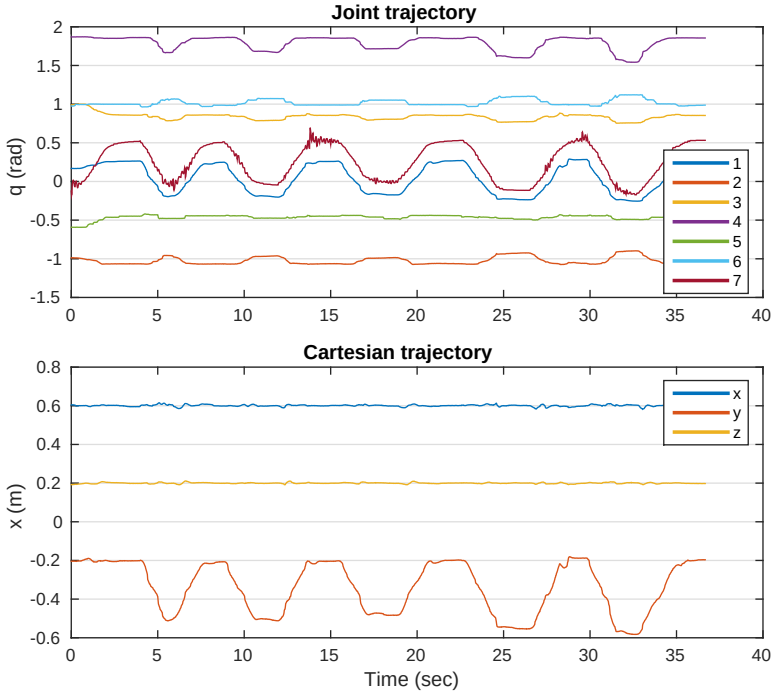

FIGURE 3.4: Test trajectory 2, movement only in the y-axis.

are performed using the inverse Jacobian method, that is

$$\dot{q}_r = J^{\dagger}(q)\dot{x}_r \tag{3.19}$$

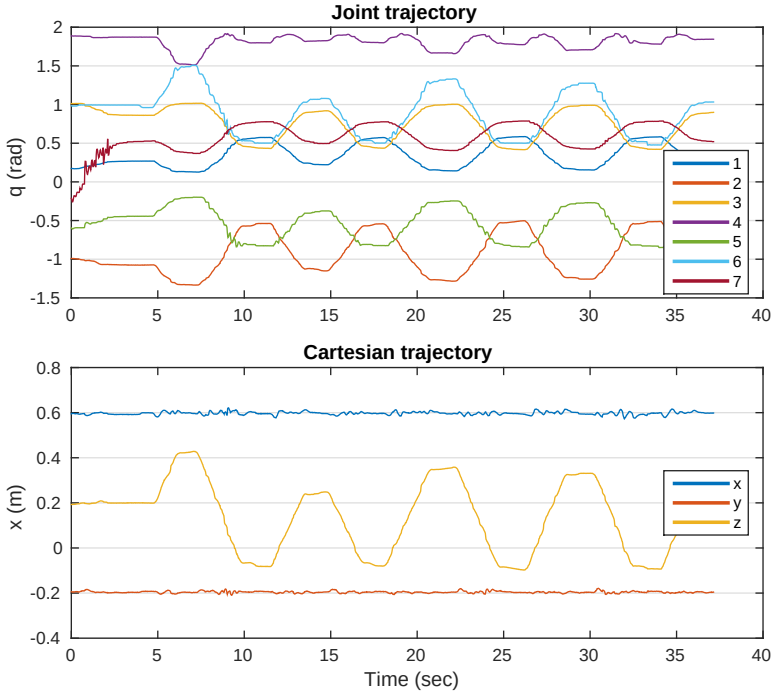

FIGURE 3.5: Test trajectory 3, moving in the z-axis.

where $\dot{x}_r$ is the reference Cartesian velocity and $J^{\dagger}$ is the pseudo-inverse of the Jacobian matrix. The test trajectories in Figures 3.3, 3.4 and 3.5 show the actual test trajectories in Cartesian space $X \equiv [x\ y\ z]^T$ which are calculated from $X = \Lambda(q)$, where $\Lambda(q)$ is the forward kinematics of the robot. The sinusoidal trajectory in Figure 3.3 was chosen as the first and second derivatives (velocity and acceleration respectively) will be excited, ensuring that all dynamics are present. The trajectories in the y (Figure 3.4) and z (Figure 3.5) axes are designed to verify dynamics in typical pick-and-place operations, with fast switching to excite the second derivative. The right-side manipulator of the Baxter was driven through these three trajectories and data collected at 50Hz, including joint positions and velocities $q, \dot{q}$, Cartesian position X and the torques applied to the motors τ. These are calculated on-board Baxter from the SEA spring deflection (and large external springs at joint 2) summed with the gravity compensation model. The joint accelerations $\ddot{q}$ were estimated through first order numerical differentiation of $\dot{q}$, which produces a noisy output. The MATLAB code used can be found in Appendix A for reference.

3.2.5 Results

Joint positions and velocities were recorded from the Baxter moving in three different trajectories as described above. Joint accelerations were estimated through numerical

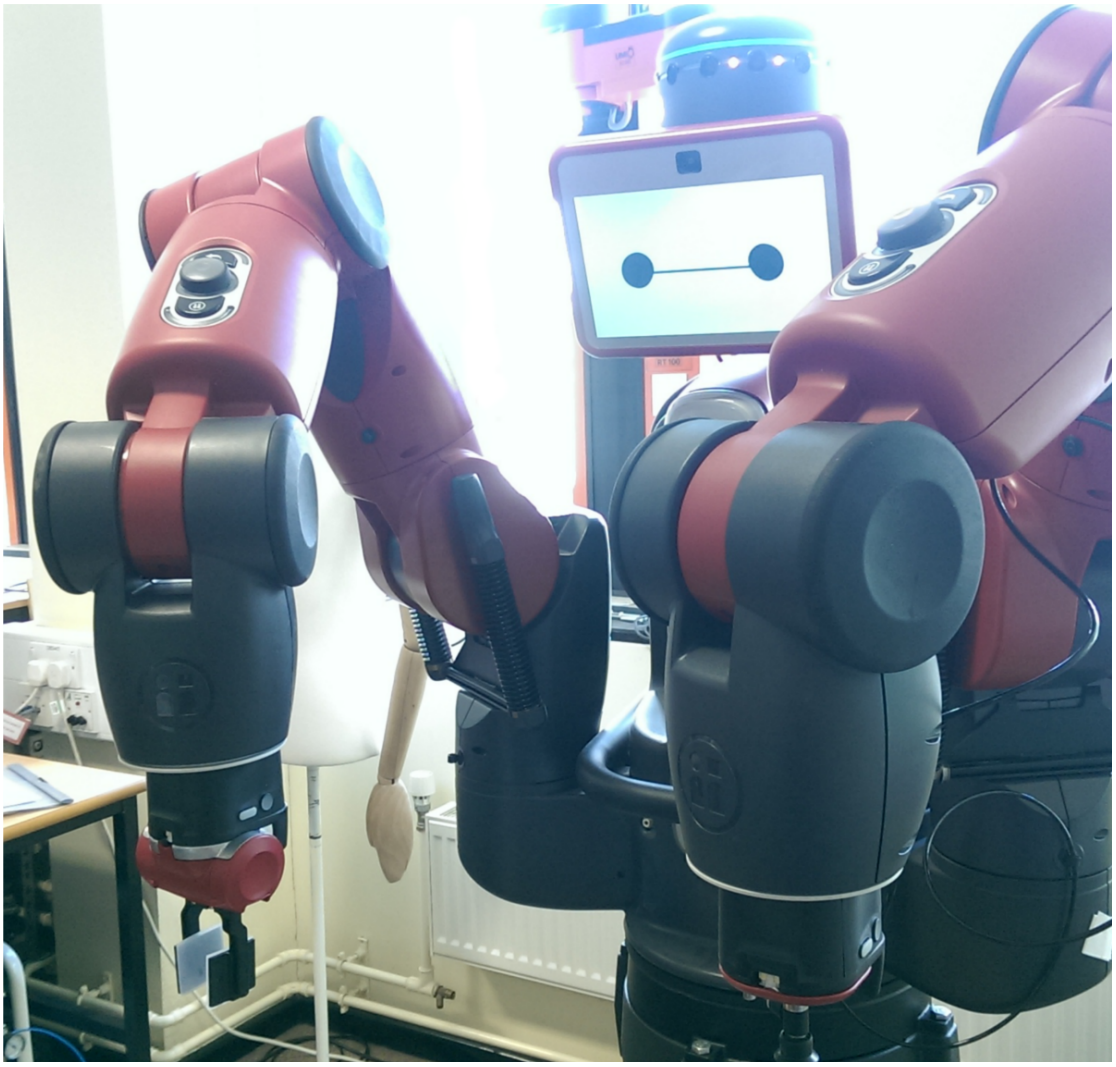

FIGURE 3.6: Baxter robot in the test position.

differentiation, i.e.

$$\ddot{q}_i = \frac{\dot{q}_i - \dot{q}_{i-1}}{dt}, \tag{3.20}$$

where $dt = 0.02$ is the sampling period of the trajectory recorder (i.e. 50 Hz) and i is the sample number. The results from the L-E and RN-E methods are compared against trajectories recorded from the Baxter, and analysed for accuracy in the dynamic model. Both methods should produce similar results.

Due to the way Baxter is controlled, the recorded torques are a sum of the actuator torques measured via internal spring deflection and two torque vectors acting on joint 2 (hysteresis and crosstalk) to compensate for the large external springs, mentioned previously. All results shown are calculated and collected for the right-hand manipulator, with the end effector aligned with the z-axis as in Figure 3.6. No external forces were applied to the arm during testing.

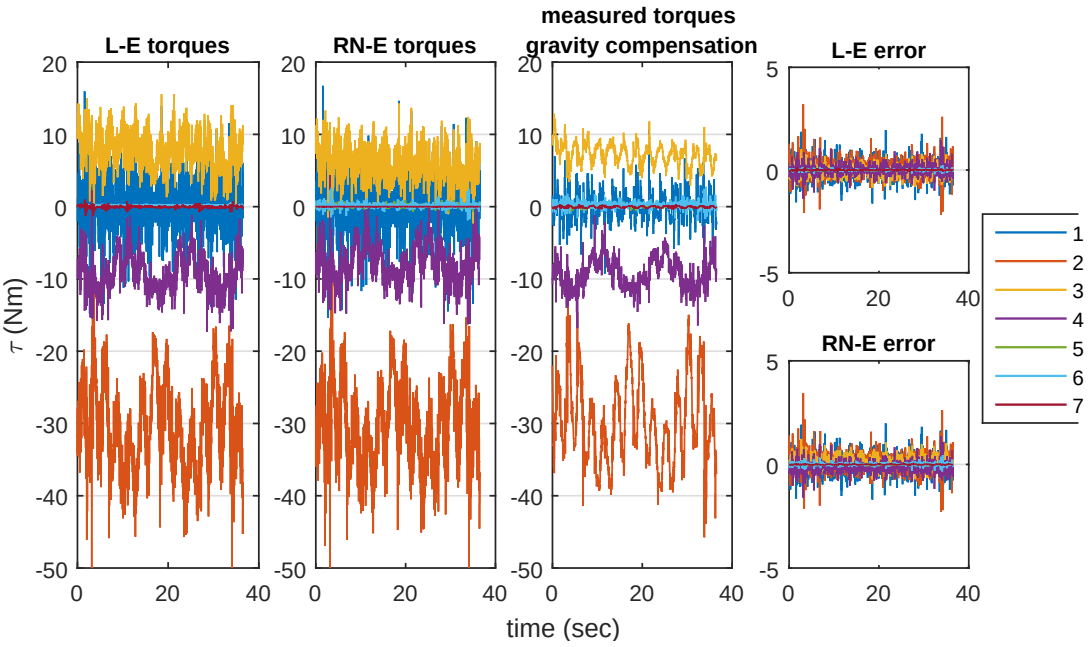

FIGURE 3.7: Comparing torque generated through L-E and RN-E methods with torques recorded from the Baxter robot during the trajectory from Figure 3.3. The trajectory for this sample was moving the end-effector in a circular trajectory in the x, y planes and in a cosine pattern in the z-axis, where $x = 0.6 + 0.1\sin(t)$, $y = -0.2 + 0.1\cos(t)$ and $z = 0.1 + 0.1\cos(0.2t)$. The errors (*far right*) are the modeled torques subtracted from the recorded torques.

In Figure 3.7 the arm was moving in all three planes. It is noticeable that the torques calculated from L-E and RN-E are much noisier; this is due to the numerical differentiation of the joint accelerations $\ddot{q}$ as described in Equation 3.20, which could be reduced through the use of a low pass filter. However, it is clear the trajectory shapes are similar. The first joint and distal joints 5-7 show only small torque input as they are mostly unaffected by gravity. By examining the L-E error plot in Figure 3.7, we can see the noise dominates the largest errors but are centred around zero for all joints, i.e. no bias errors are present on any joint. The RN-E error plot shows a similar range of error, but it is noticeable that the error for joint 3 has some positive bias error.

In Figure 3.8 we have similar results, with an even smaller error result. In this case the arm was moved in a way to generate higher accelerations by quickly switching the target position in the y-axis only. This movement is mostly achieved using joint 2 at the shoulder, noticeable in the plots. The low error result in this case confirms a good match for the kinetic part of the dynamic model. Again, looking at the RN-E there is an obvious positive bias error in the torques calculated for joint 3.

A slower trajectory was applied to the robot for the results in Figure 3.9, moving primarily in the z-axis, which is evident by the large changes occurring in joint 4. Noise is reduced due to minimal acceleration in the trajectory. The error results again show

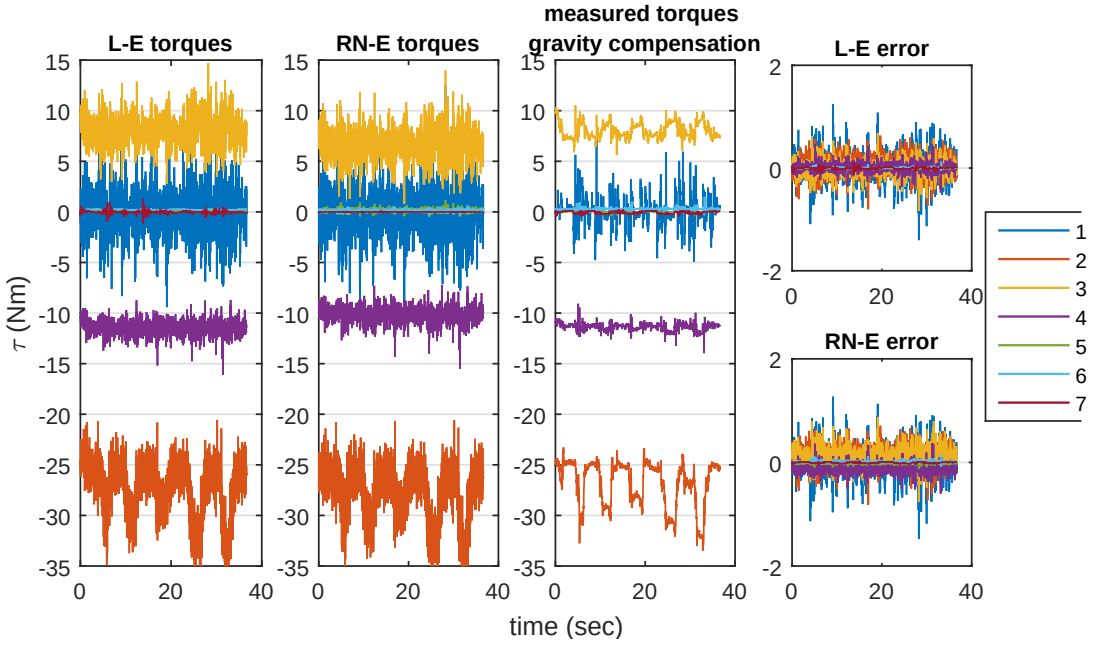

FIGURE 3.8: Second comparison of torque trajectories from Figure 3.4; for this sample the end-effector is fixed in the x, z plane and is switched quickly between two positions in the y-axis.

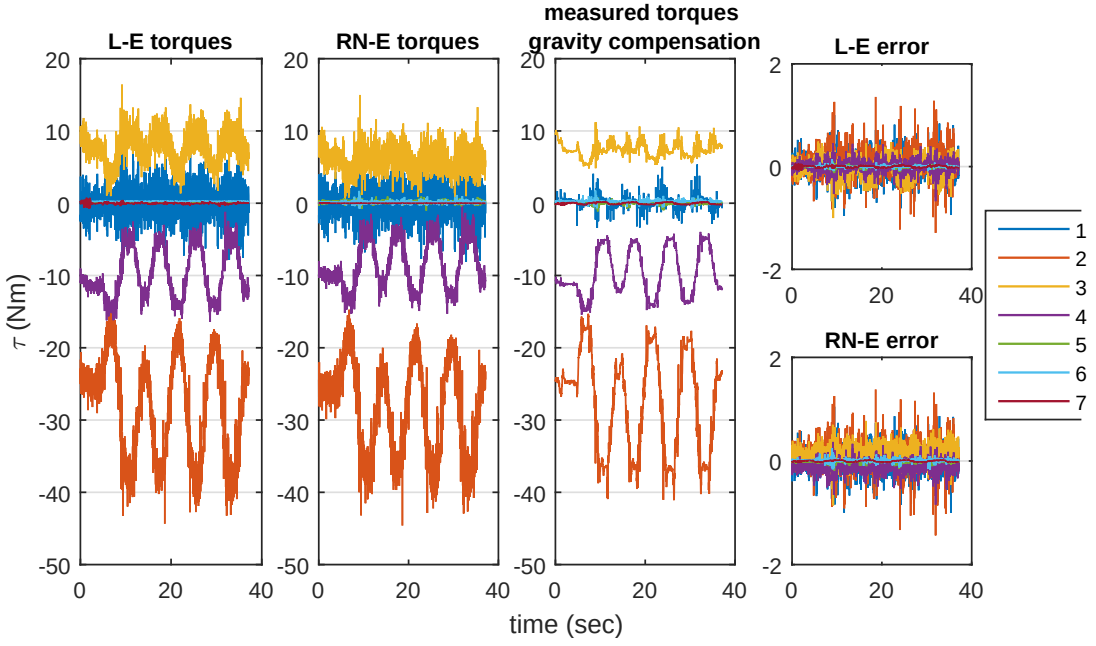

FIGURE 3.9: Final comparison of torque trajectories from Figure 3.5 input. The end-effector was moved up and down in the z-axis, and held constant in the x, y plane.

no bias errors, and within a good tolerance of around ± 1.5 Nm which mostly can be accounted for by noise from the acceleration trajectory derivation.

A clearer comparison of the results is shown in Table 3.4, where the average and sum total torque errors of both methods are shown side by side. These are calculated from

TABLE 3.4: Averages of calculation errors for L-E and RN-E methods.

	Set					
	1		2		3	
Joint	L-E	RN-E	L-E	RN-E	L-E	RN-E
$\lvert\bar{e}_1\rvert$	0.0105	0.0105	0.0148	0.0149	0.0061	0.0058
$\lvert\bar{e}_2\rvert$	0.1002	0.0665	0.0464	0.0931	0.0703	0.0872
$\lvert\bar{e}_3\rvert$	0.0475	0.1382	0.0083	0.1355	0.0367	0.1358
$\lvert\bar{e}_4\rvert$	0.0151	0.1231	0.0079	0.1210	0.0064	0.1148
$\lvert\bar{e}_5\rvert$	0.0068	0.0120	0.0099	0.0145	0.0079	0.0140
$\lvert\bar{e}_6\rvert$	0.0006	0.0103	0.0047	0.0136	0.0003	0.0113
$\lvert\bar{e}_7\rvert$	0.0012	0.0036	0.0003	0.0055	0.0013	0.0024
$\sum_{i=1}^{n} \bar{e}_i$	0.0710	0.0869	0.0527	0.1161	0.0366	0.1089

each trajectory set, i.e. set 1 in Table 3.4 corresponds to the first trajectory results in Figure 3.7. Looking through the table, it can be seen that the average error for each joint is comparable between the methods, apart from in joints 3-4 which have significantly larger errors in every set. All average error values are also within reasonable limits.

3.3 Controller Design

The human-like adaptive law for tuning the feed-forward and feedback components of the control torque $\tau_u \in \Re^{n\times 1}$ (i.e. applied to all joints) from [16] can be applied in either joint or task space. It is described here in terms of continuous movement, rather than trial-on-trial, so that tracking error and effort are continuously minimised. Let us define

$$\tau_u(t) = \tau_r(t) - \tau_j(t) - L(t)\varepsilon(t) \tag{3.21}$$

with

$$\begin{aligned} e &= q - q_r \\ \dot{e} &= \dot{q} - \dot{q}_r \\ \varepsilon &= \dot{e} + \kappa e, \quad \kappa > 0 \end{aligned} \tag{3.22}$$

and

$$\tau_j = \tau + K(t)e(t) + D(t)\dot{e}(t) \tag{3.23}$$

where $-\tau(t)$ is the learned *feed-forward* torque, and $-K(t)e(t)$ and $-D(t)\dot{e}(t)$ are *feed-back* torque terms due to stiffness and damping, respectively, and ε is the tracking error used frequently in manipulator control [212]. The term $-L\varepsilon(t)$ describes a stability margin, which in the human arm is produced by the passive compliance of the muscles and

tendons [213]. A reference torque τ_r compensates for manipulator dynamics, described by:

$$\tau_r = M\ddot{q}_r + C\dot{q}_r + G \tag{3.24}$$

where M, C and G are the manipulator dynamics from Equation 3.1 and $\ddot{q}_r$, $\dot{q}_r$ are reference trajectory acceleration and velocity respectively. The adaptive laws introduced in [16] for a trajectory of period T are given as:

$$\begin{aligned}
\delta\tau(t) &\equiv \tau(t) - \tau(t-T) = Q_\tau\big(\varepsilon(t) - \gamma(t)\tau(t)\big),\\
\delta K(t) &\equiv K(t) - K(t-T) = Q_K\big(\varepsilon(t)e^T(t) - \gamma(t)K(t)\big),\\
\delta D(t) &\equiv D(t) - D(t-T) = Q_D\big(\varepsilon(t)\dot{e}^T(t) - \gamma(t)D(t)\big).
\end{aligned} \tag{3.25}$$

The design is changed here to decouple the forgetting factor $\gamma(t)$ from the gain matrices $Q_{(\cdot)}$ to avoid high frequency oscillation which occurs when both γ and $Q_{(\cdot)}$ are set to high values. The effect of γ is to reduce the amount feedback being applied when the tracking error is low, similar to human muscle relaxation [214], e.g. if there is no error then γ will be maximal and subtract from the applied torques.

The adaptation is also considered in continuous time, rather than by iteration over consecutive trials, yielding the new adaptation laws:

$$\begin{aligned}
\delta\tau(t) &\equiv \tau(t) - \tau(t-\delta t) = Q_\tau\ \varepsilon(t) - \gamma(t)\ \tau(t)\,,\\
\delta K(t) &\equiv K(t) - K(t-\delta t) = Q_K\ \varepsilon(t)e^T(t) - \gamma(t)\ K(t)\,,\\
\delta D(t) &\equiv D(t) - D(t-\delta t) = Q_D\ \varepsilon(t)\dot{e}^T(t) - \gamma(t)\ D(t)
\end{aligned} \tag{3.26}$$

where δt is the sampling period, $K(0) = 0_{[n\times n]}$ and $D(0) = 0_{[n\times n]}$. $Q_\tau, Q_K, Q_D \in \Re^{n\times n}$ are diagonal positive-definite gain matrices and determine the adaptation rate of each part.

Another change is made to the controller from [16]; $\gamma(t) \in \Re^{n\times n}$ is a diagonal matrix previously defined as

$$\gamma_{ii}(t) = \frac{a}{1 + b\|\varepsilon_i(t)\|^2} \tag{3.27}$$

which requires two tuning parameters, a and b. To simplify parameter selection, γ is redefined in this work as

$$\gamma_{ii}(t) = \alpha_{j_i}\ \exp\left(-\frac{\varepsilon_{j_i}^2(t)}{0.1\,\alpha_{j_i}^2}\right), \quad 0 < \alpha_j \leq 1 \tag{3.28}$$

which requires only one variable, α_j, to describe the shape shown in Figure 3.10, but maintaining the same functionality.

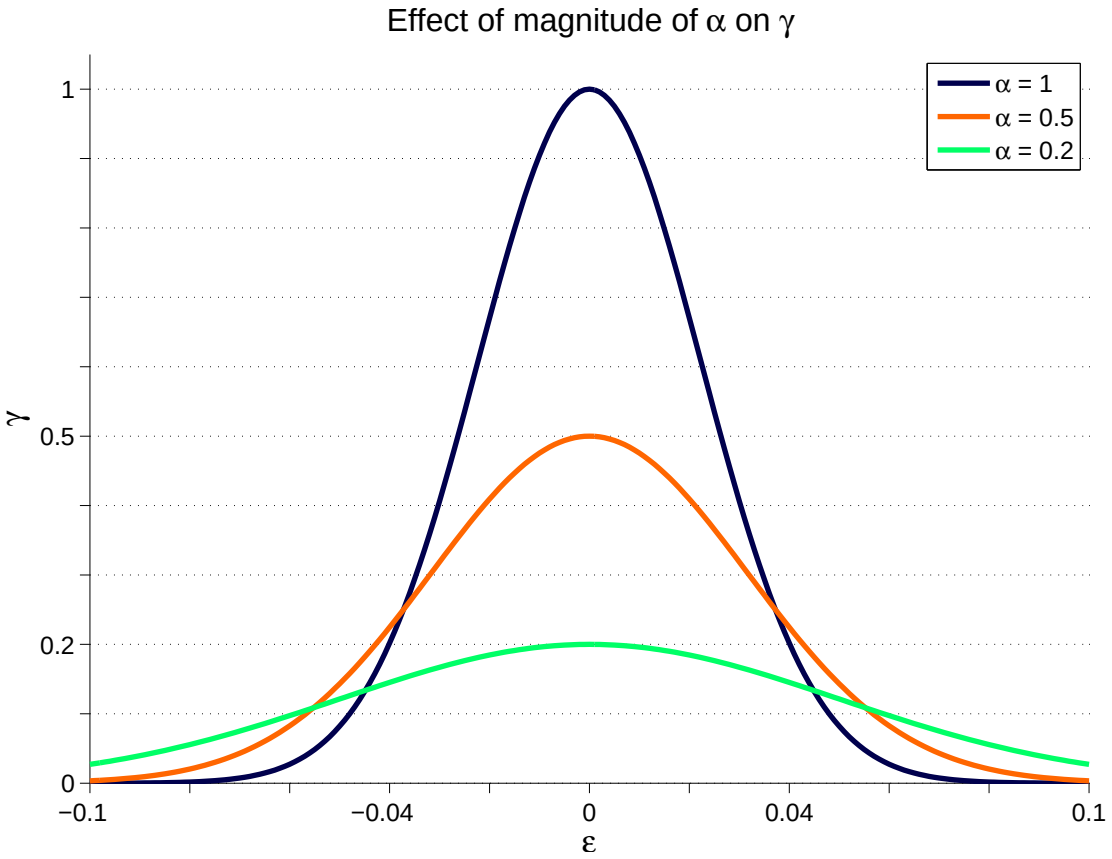

FIGURE 3.10: How the magnitude of α affects the forgetting factor γ. Higher values of α have a high gain and narrow shape, so that when tracking performance is good the control effort is greatly reduced. When tracking performance is poor, the forgetting factor is small and control effort is not "relaxed".

3.3.1 Experimental Verification

To verify that the altered controller can still achieve the results of the former, a simulation task with a model of the Baxter arm was designed. First, the arm is tested holding a initial position whilst under the influence of disturbance force fields. Then a second test is carried out with the arm moving in a minimum-jerk trajectory, under the same divergent conditions.

Three different force fields are applied to the end-effector of the manipulator during testing. These consist of a constant force

$$F_c = \begin{bmatrix} 0 \\ 10 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{N}; \tag{3.29}$$

a position dependent force, relative to the initial position:

$$F_p(t) = \begin{bmatrix} k\,\big(x(t) - x(0)\big) \\ k\,\big(y(t) - y(0)\big) \\ k\,\big(z(t) - z(0)\big) \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad k = 100 \text{ N/m}; \tag{3.30}$$

and a velocity dependent force field:

$$F_v(t) = \begin{bmatrix} d\,\dot{x}(t) \\ d\,\dot{y}(t) \\ d\,\dot{z}(t) \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad d = 10 \text{ Ns/m}; \tag{3.31}$$

The adaptive terms τ, K and D from Equation 3.21 all start from zero, i.e. the initial feedback is minimal and only provided by the $L\,\varepsilon(t)$ term. The force fields are applied in four equal phases:

- Phase I: $F_D = F_c$,
- Phase II: $F_D = F_c + F_p$,
- Phase III: $F_D = F_c + F_p + F_v$,
- Phase IV: $F_D = 0$, i.e. no interaction force.

For the following tests, the controller parameters were chosen as such:
$Q_\tau = \text{diag}(30\ 20\ 20\ 30\ 10\ 20\ 0.1)$, $Q_K = \text{diag}(80\ 80\ 80\ 80\ 80\ 80\ 0.1)$, $Q_D = \text{diag}(20\ 20\ 20\ 20\ 20\ 20\ 0.1)$, $L = \mathbf{I}^{7\times7}$, $\alpha = 0.5$ and $\kappa = 7$. Simulations were all set to run with a 1 kHz update rate. The initial position of the end-effector, of the left arm, was set to $X(0) = [0.522\ 0.778\ 0.161\ -180\ 0\ 178.2]^T$, which is a position comfortably within the workspace in front of the robot, where most pick-and-place tasks are likely to take place.

3.3.1.1 Static reference position

The first simulation consists of a zero velocity reference trajectory, i.e. $X_r(t) = X(0)$, lasting for ten seconds, with the force fields being applied for 2.5 seconds each. The

purpose of this is to demonstrate how the feed-forward and feed-back terms respond to the applied disturbance and the relaxing effect of the forgetting factor γ when no disturbance is applied.

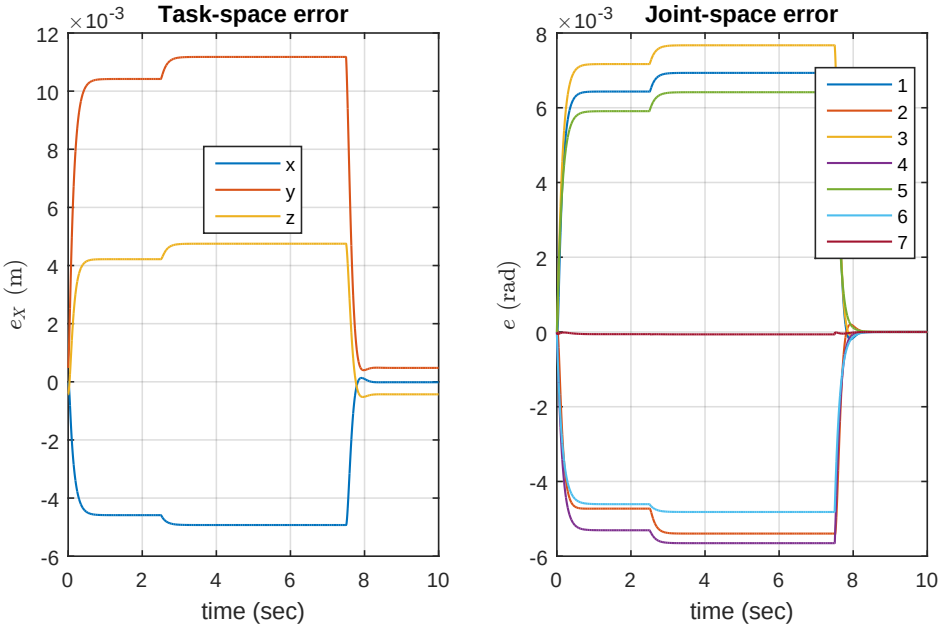

FIGURE 3.11: Task-space error (left) and joint-space error (right) for ten second simulation with static reference position.

The results of position error in task-space and joint-space shown in Figure 3.11. Phase changes occur at 2.5, 5 and 7.5 seconds. Looking at the period between 2.5 seconds to 7.5 we can see that the introduction of the velocity dependent force field F_v does not affect the error, as is expected. We can also see that after 7.5 seconds, when the force fields are switched off, that the joint-space error appears to converge to zero. However, by comparing this to the task-space error that in the final phase there is some position offset error even without disturbances forces, which tells us that there are some very small joint errors occurring in this phase. The feed-forward and feedback terms from Equation 3.21 are mapped into x, y task-space for analysis via the identities:

$$\begin{aligned} F &= J^{-T}\,\tau \\ K_C &= J^{-T}\,K\,J^{-1} \\ D_C &= J^{-T}\,D\,J^{-1} \end{aligned} \tag{3.32}$$

which are presented in Figure 3.12. Looking at the damping terms, we can see that for the majority of the test period no feedback is added, apart from at the phase boundaries where a small movement can be seen at the end effector. The majority of disturbance compensation is attributed to the feed-forward force, which increases to approximately

10 N in the y-axis to compensate for F_c defined in Equation 3.29, and then levels off as a acceptable tracking error has been achieved, the level of which is set by tuning the parameter α. A similar effect can be seen looking at the evolution of the stiffness matrix,

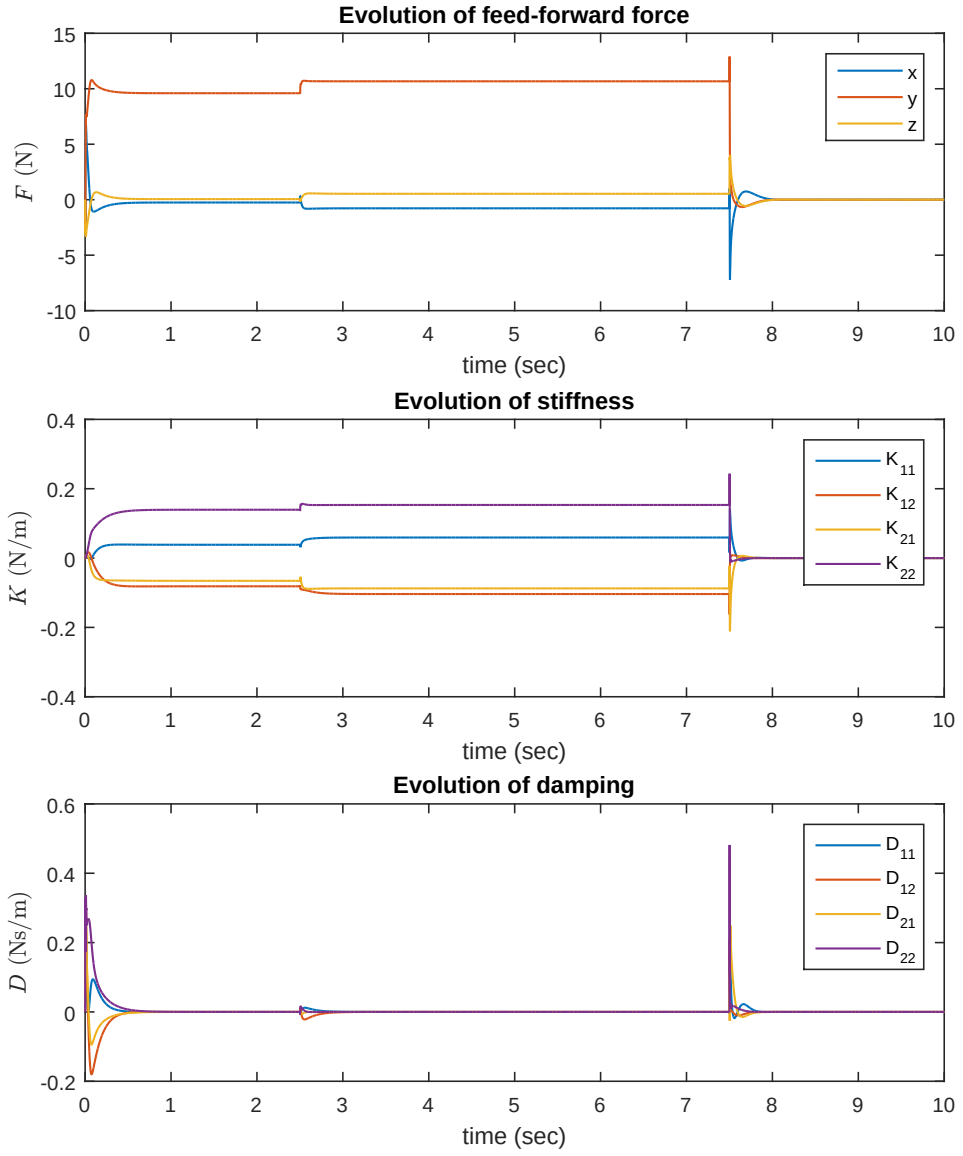

FIGURE 3.12: Evolution of feed-forward force, stiffness and damping (in task-space), static reference position.

where in the first phase an increase in elements relating to the y-axis. As phase II is applied another small increase is seen to compensate for F_p in all three axes. No change appears as expected in phase III, but when all disturbances are removed in phase IV we can see that all terms converge to zero due to the effect of γ. At this point the torques being applied to the modelled Baxter consist only of the dynamics compensation τ_r and

the stability margin $L\,\varepsilon(t)$, which causes the small steady-state task-space error seen in Figure 3.11. This could be reduced even further by reducing the value of α or increasing L, but this would remove the natural compliance from the arm which is part of the design specifications.

3.3.1.2 Moving trajectory

The second task consists of tracking a smooth minimal-jerk trajectory along the x, y coordinates defined as:

$$\begin{aligned} X(t) &= X(0) + \big(X(T) - X(0)\big)(10\bar{t}^3 - 15\bar{t}^4 + 6\bar{t}^5), \quad \bar{t} \equiv \frac{2t}{T} \\ x_r(t) &= -X(t), \\ y_r(t) &= X(t), \end{aligned} \tag{3.33}$$

where T is the trajectory period. Joint-space angular velocity is computed using the pseudo inverse $J^\dagger(q) \equiv J^T(JJ^T)^{-1}$ of the Jacobian, through

$$\dot{q}_r(t) = J^\dagger(q)\ [x_r(t),\ y_r(t),\ z(0),\ \phi(0),\ \varphi(0),\ \psi(0)]^T, \tag{3.34}$$

where ϕ, φ and ψ are the roll, pitch and yaw angles respectively, from which the joint positions and accelerations can be found through integration and derivation:

$$q_r(t) \equiv \int_0^t \dot{q}_r(t)\ \mathrm{d}t\,, \quad \ddot{q}_r \equiv \frac{d}{dt}\left(\dot{q}_r(t)\right). \tag{3.35}$$

The simulation is run for longer (40 seconds) so that each disturbance phase is applied for 10 seconds.

The resulting Cartesian trajectory is shown in Figure 3.13 along with the reference trajectory X_r, coloured depending on the phase. The initial position $X(0)$ is located towards the left-most end of the black trace. As can be seen from the graph, the end-effector immediately takes on a position bias in positive z in phase I, which remains throughout the test including phase IV when no disturbances are applied. The effect of F_p can be seen as the trajectory approaches the point furthest away from $X(0)$ towards the right of the graph in phases II and III, with the most diverse trajectory being in phase III as would be expected due to all three force fields being in effect. The task-space and joint-space errors are compared in Figure 3.14. It is clear that there is a position drift occurring in task-space, despite errors in joint-space appearing to be bounded, including in phase IV where no disturbance is applied. Left unchecked this would result in the end-effector drifting far from the reference trajectory. Examining the evolution

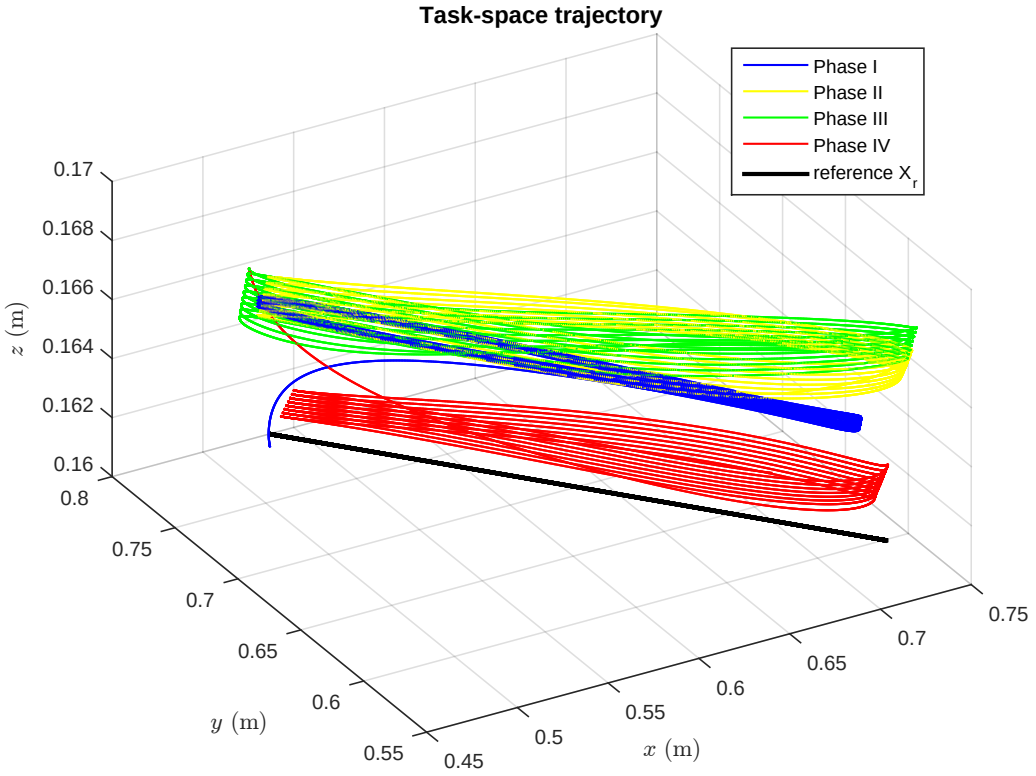

FIGURE 3.13: Task-space trajectory of the Baxter end-effector after 40 second simulation through minimum-jerk trajectory.

of feed-forward force for this run in Figure 3.15 we can see how this term in phase I fluctuates around 10 N in the y-axis to compensate for F_c, with some compensation coming from stiffness feedback and very little from the damping. In phases II and III an increase in F and K is seen in both x, y axes to compensate for F_p, while the damping remains low. In phase III, as the velocity dependent F_v is introduced we can see that the damping term also increases to help compensate against this. Finally in phase IV, starting at 30 seconds, we can see that all three terms drop to much lower levels, reducing the control effort applied to the manipulator. We can also look at the stiffness geometry by plotting the ellipse in Figure 3.16. We can see that the ellipses in phases I-III are orientated mainly in the direction of the trajectory, and slightly more angled in the y-axis where more disturbance is applied. As F_D is removed in phase IV (dashed) the ellipse becomes much smaller and more equally distributed in the x and y axes to compensate only for the trajectory.

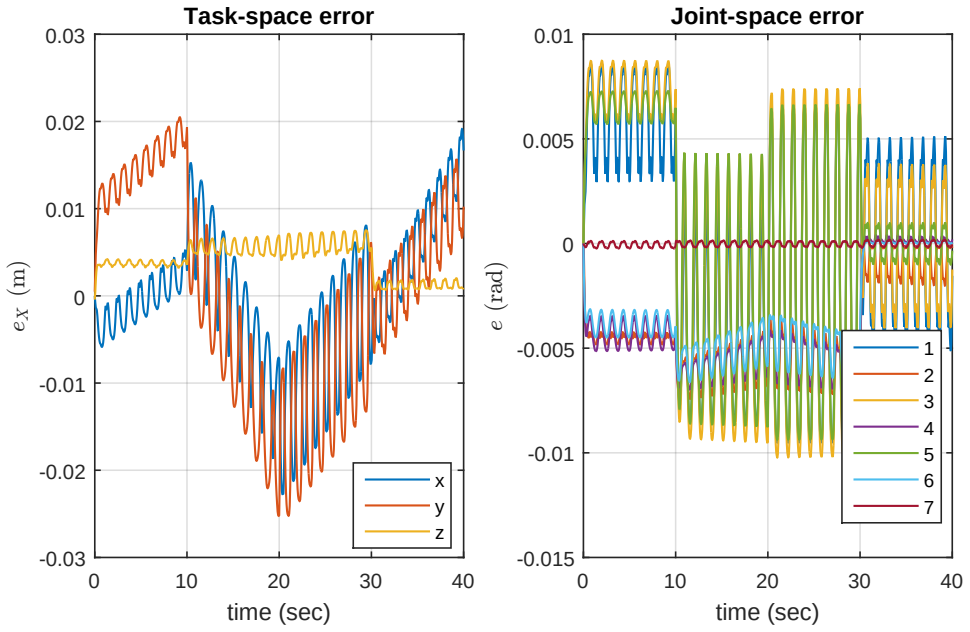

FIGURE 3.14: Task-space error on the left, joint-space on the right, 40 second simulation with minimum-jerk trajectory.

3.4 Concluding Remarks

This chapter describes the method for derivation of the Lagrange-Euler and recursive Newton-Euler dynamics, followed by experimental verification using data collected from the Baxter robot; this is the foundation for simulated experiments carried out in this work. The biomimetic controller is then adapted from [16] and discussed. The controller was then verified through simulated experimentation using the model discussed in the first part of the chapter.

The results show that the derived model is a good match to the real dynamics, with low errors in three different end-effector trajectories shown in Figures 3.7, 3.8 and 3.9. The majority of the error can be attributed the noise caused by numerical derivation of joint accelerations. Without this noise, a very low torque error would be achievable and show that a close computational model of the Baxter dynamics is possible. However, the test trajectories that were used may not fully exploit all possible dynamics. For example, the trajectory in Figure 3.3 is sinusoidal, but only in Cartesian space, which may mean that there is low or no excitation of mass in some joints (joint 5 in Figure 3.3 for example). In addition, the test trajectories were all located close to the robot body, where the gravitational forces at proximal joints is kept low. The accuracy of the dynamic model may not be so good when the arms are at extension and close the limits of the workspace, where the weight of the distal joints will increase the effect of gravity on the proximal

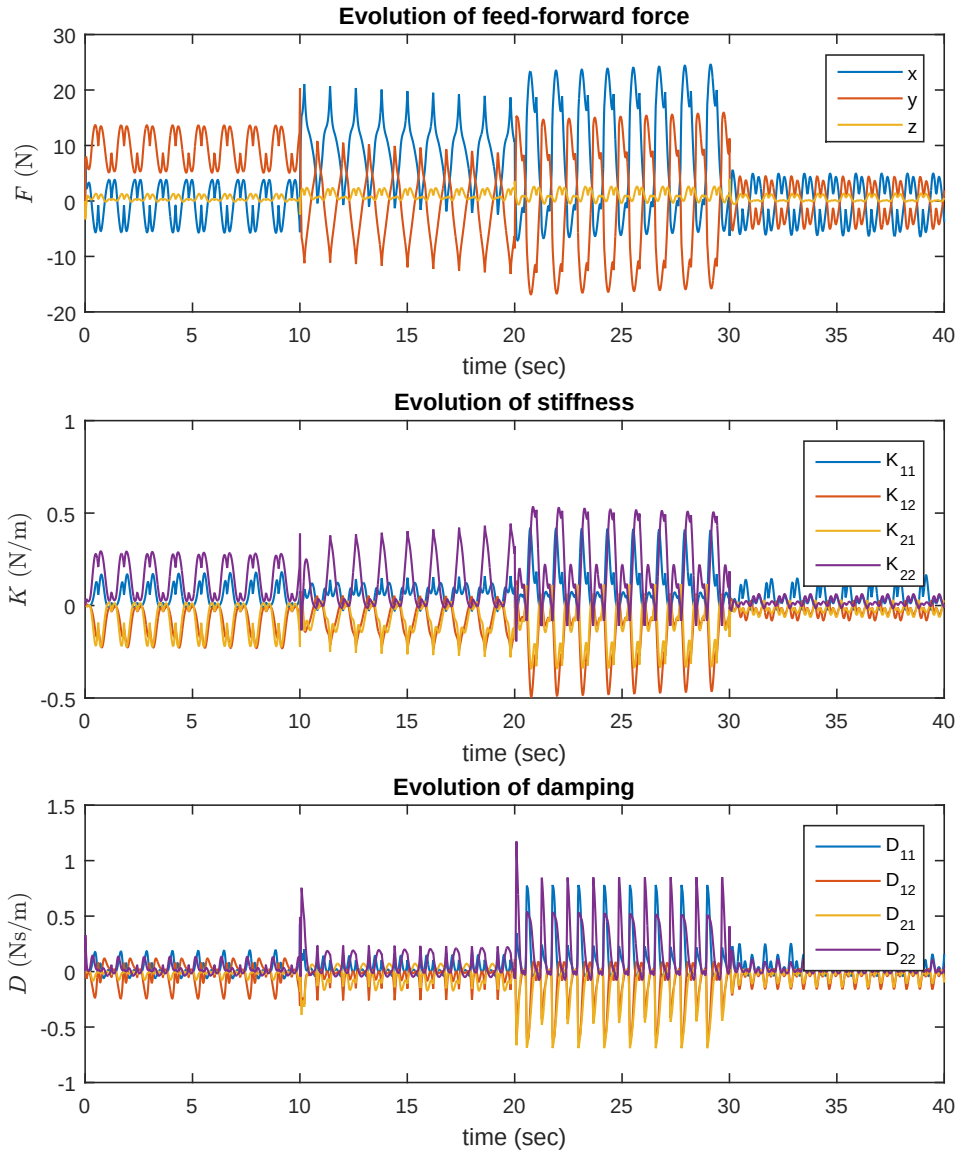

FIGURE 3.15: Evolution of the terms described in Equation 3.32 over the four phases, minimum-jerk trajectory.

joints. Therefore, more trajectories should be tested, such as those described in [215], to fully explore the dynamics in the workspace.

To find the explicit closed form of Baxter manipulator dynamics, MATLAB's symbolic toolbox was utilised. In their raw state, the symbolic representations of the elements of $D(q), C(q, \dot{q})$ and $G(q)$ have many coefficients (over half a million), so cannot be printed here but are available online in MATLAB workspace format for analytical use. This is a useful tool for control system design, provided in a commonly used format, so could prove useful to many researchers.

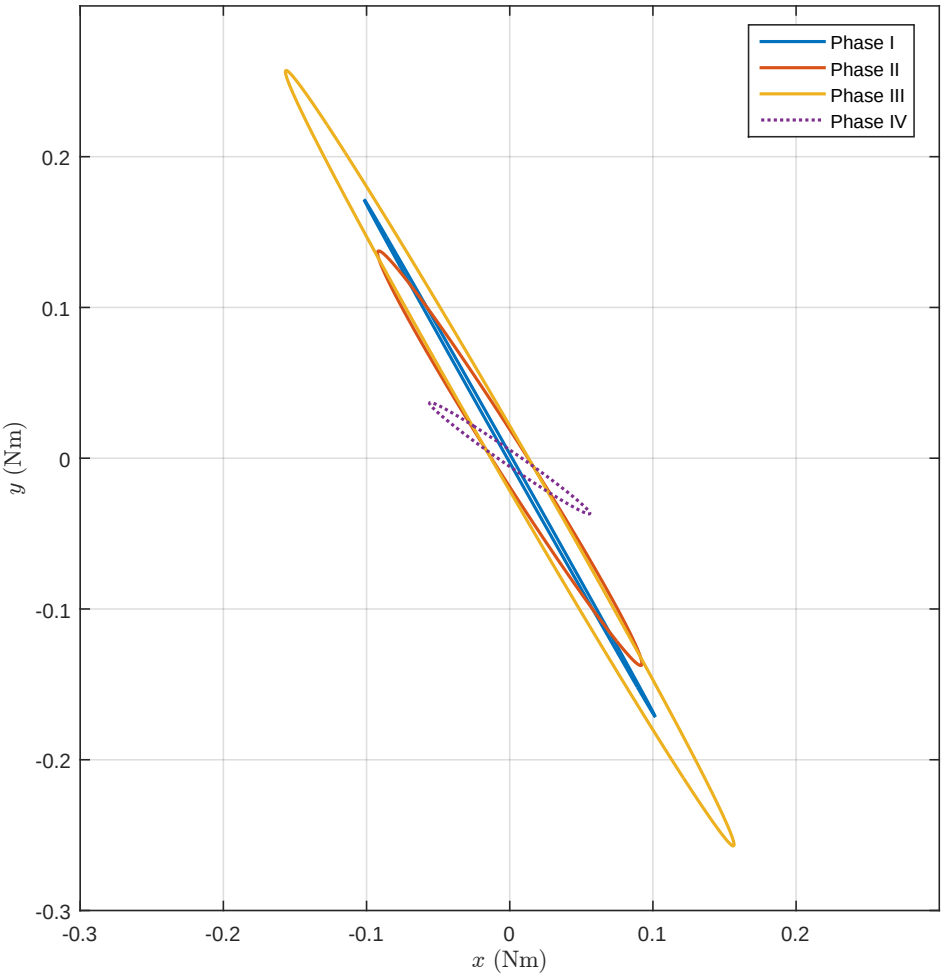

FIGURE 3.16: Stiffness geometry of each phase taken from the average of K across the phase.

The results from the controller simulations show that it is able to work in continuous time and with the change made to γ to reduce the number of control parameters, on a 7 DoF robot both in a static pose and while moving through a trajectory. We have also seen how the feed-forward and feedback terms of the controller only increase to compensate for disturbance interactions when required, then are reduced automatically when tracking error is low. This is similar to what is observed in humans with the minimisation of metabolic cost [179].

An unwanted effect is also observed due to the control structure operating in joint-space: despite having only small joint position errors, the Cartesian trajectory is subject to drifting which would eventually lead to sudden instability if left unchecked. This is due to the nature of the forgetting factor γ, which will always allow some small error to be

present, and without which would result in a very stiff control scheme, undesirable for our purposes. This is what leads us to the next chapter: a hybrid controller working in both joint-space and task-space, so that the task-space drift is removed whilst maintaining the compliant nature of the controller.

Chapter 4

Joint-space and Task-space Hybrid Control of the Biomimetic Controller

4.1 Introduction

As discussed in the previous chapter, the biomimetic controller shows promising performance but also demonstrates an unwanted position drift in task-space when the controller is implemented in joint space. This lead to the novel idea of a hybrid controller working concurrently in both joint and task-space, to counteract the drift and improve the ability of the controller. Conventionally, a manipulator controller is implemented in either task-space or joint-space, depending on the application. There are benefits and drawbacks to both, for example:

- A Cartesian controller is easier to visualise, as it is understandable in terms of direction and magnitude. As humans we can estimate how long half a metre is, and what twenty Newton of force feels like. It is more intuitive to design a task trajectory in Cartesian space rather than in joint-space, which is nearly impossible in most applications.
- A manipulator will normally require the control input to be in joint-space, e.g. motor torques rather than forces and wrenches. Therefore a joint-space controller will be less computationally heavy due to not requiring calculation of the inverse kinematics, which may be complicated for some manipulator designs, such as underactuated or redundant robots where the Jacobian matrix is non-square.

- As observed in the previous chapter, small errors in joint-space may result in large task-space errors dependent on the kinematics and configuration of the manipulator.
- In a joint-space controller, control gains can be tuned to suit the motors, e.g. if the proximal joints are driven by larger, more powerful motors then more torque can be applied than the distal motors which are likely to be smaller. However, a Cartesian controller allows the user to tune gains dependent on the task if disturbances are expected from a certain direction.

We consider a task here where disturbances could be applied at any point along the arm, at the end-point, or both; this can be visualised as moving through a crowd while holding a tray of drinks [216]. Correspondingly, we would like to combine

- joint control which can provide robustness against disturbance applied at any point on the arm, and
- Cartesian-space control, which is particularly sensitive to perturbations at the end-effector.

The remaining of this chapter is formed from a description of the hybrid controller with justification, experimental verification similar to that in Chapter 3 and discussion of the results.

4.2 Controller Design

The hybrid controller is composed of the same controller as previously, but with two working in joint-space and task-space concurrently. A weighting matrix is employed to balance the feedback between the two controllers. Because of the nature of the biomimetic control, that is, the constant minimisation of control effort, the two controllers can work together without providing excess feedback torque.

The *task-space* controller is designed in a similar manner to joint space. First, we define the error term in Cartesian space:

$$\begin{aligned} e_x(t) &= X(t) - X_r(t) \\ \dot{e}_x(t) &= \dot{X}(t) - \dot{X}_r(t) \\ \varepsilon_x(t) &= \dot{e}_x(t) + \kappa e_x(t). \end{aligned} \tag{4.1}$$

Correspondingly, the task-space feed-forward and feedback terms similar to Equation 3.26 are defined as

$$\begin{aligned}\delta F_x(t) &= Q_F\ \varepsilon_x(t) - \gamma_x(t)\ F_x(t)\\ \delta K_x(t) &= Q_{K_x}\ \varepsilon_x(t) e_x^T(t) - \gamma_x(t)\ K_x(t)\\ \delta D_x(t) &= Q_{D_x}\ \varepsilon_x(t) \dot{e}_x^T(t) - \gamma_x(t)\ D_x(t)\end{aligned} \tag{4.2}$$

so that, similar to Equation 3.23, the Cartesian feedback torque is defined,

$$\tau_x(t) = J^T(q)(F_x(t) + K_x(t)e_x(t) + D_x(t)\dot{e}_x(t)), \tag{4.3}$$

and the corresponding forgetting factor is defined similarly to Equation 3.28, below:

$$\gamma_x(t) = \alpha_x\ \exp\left(-\frac{\varepsilon_x^2(t)}{0.1\,\alpha_x^2}\right), \quad 0 < \alpha_x \leq 1\,. \tag{4.4}$$

The reference torque Equation 3.24 and stability margin $L\ \varepsilon(t)$, the joint-space controller of Equation 3.23 and the task-space controller of Equation 4.3 yields the *hybrid controller*, and therefore the revised input torque τ_u

$$\tau_u(t) = \tau_r(t) - L\ \varepsilon(t) + \tau_x(t) + \Omega\tau_j(t) \tag{4.5}$$

where $\Omega \in \Re^{n\times n}$ is a weighting matrix, designed such that the joint torque feedback is limited to certain joints, dependent on the required task. Assuming an accurate dynamic model of the robot is available, the torques due to disturbance τ_{dist} are estimated as

$$\tau_{dist} = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) - \tau_u \tag{4.6}$$

i.e. the manipulator dynamics *minus* the input torque. This torque vector τ_{dist} is normalised to the maximum element, and the weighting matrix Ω can be formed:

$$\Omega_{ii} = \frac{\tau_{dist_i}}{\max\limits_{1\leq i\leq n}(\tau_{dist})}, \quad \Omega_{ij} \equiv 0\,(i \neq j) \tag{4.7}$$

which is then applied in Equation 4.5, so that joint-space feedback is primarily applied to areas of the arm under the influence of large disturbance forces, and less to those which are not; this again limits the control effort being applied unnecessarily, reducing the overall control effort that would otherwise be applied.

4.3 Experimental Verification

Similar to the experiment in the previous chapter, the task consists of one Baxter arm moving in a minimum-jerk trajectory. The disturbance forces are made more complex to demonstrate the benefits of the hybrid controller and resemble the crowded room task mentioned in Section 4.1.

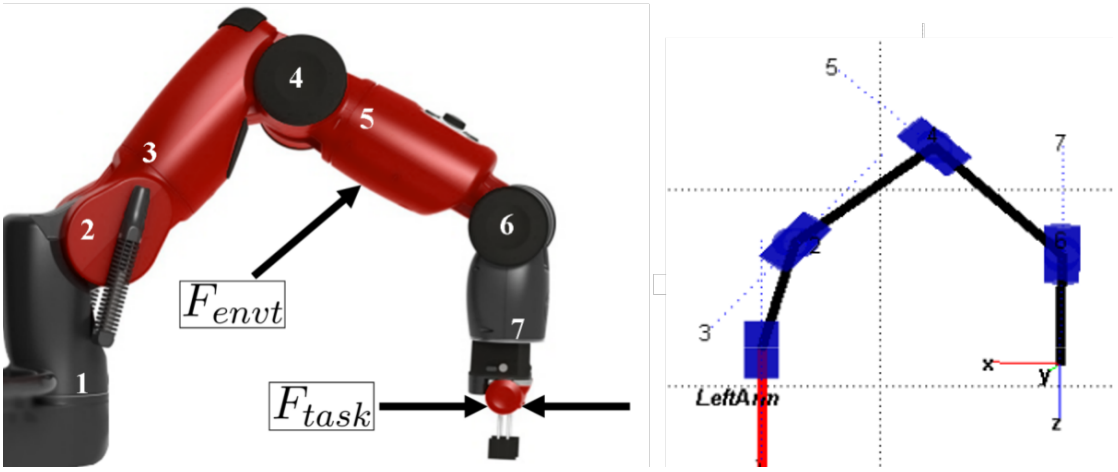

FIGURE 4.1: Diagram demonstrating how the simulated task disturbance F_{task}, and environmental disturbances F_{envt}, are applied to the modeled Baxter arm.

We model the disturbance torque τ_{dist} as two components, to simulate both a force acting at the end-effector, described here as F_{task}, and an environmental disturbance F_{envt} applied on the arm, as shown in Figure 4.1:

$$F_{task} \equiv [p\ 0\ 0\ 0\ 0\ 0]^T, \quad p = A_p \sin(2\pi f_p t) \tag{4.8}$$

is applied on the endpoint, where $0 < A_p \leq 20$N is the amplitude and $100 < f_p \leq 1000$ the frequency of oscillation in Hertz. In joint space, the torque applied is then

$$\tau_{task} = J^T(q)\, F_{task}. \tag{4.9}$$

The environmental disturbance is given by

$$F_{envt} \equiv [r\ 0\ 0\ 0\ 0\ 0]^T, \quad r = A_r \sin(2\pi f_r t) \tag{4.10}$$

where $20N < A_r \leq 100$N is the perturbation amplitude, similar to average limits of human push/pull strength [217], and $0.1 < f_r \leq 1$ the frequency in Hertz, which provides a slowly changing disturbance like being pushed upon. To simulate the environmental force F_{envt} being applied at a point on the arm, e.g. at the elbow, the Jacobian matrix

J is reduced by a matrix Z, defined as

$$Z \equiv \begin{bmatrix} \mathrm{I}_{[z\times z]} \\ 0_{[(n-z)\times z]} \end{bmatrix} \tag{4.11}$$

where z is the number of joints from the base to the contact point and can be chosen for the simulation, e.g. if the force is applied on the elbow, similar to what is depicted in Figure 4.1, $z = 4$. The torque can then be found as

$$\tau_{envt} = (J(q)\ Z)^T\ F_{envt}. \tag{4.12}$$

The simulated disturbance torque τ_{dist} is then comprised of a combination of terms in Equations 4.9 and 4.12, and again applied to the manipulator during testing in four phases:

- Phase I: No disturbance,
- Phase II: F_{task} only,
- Phase III: F_{envt} only,
- Phase IV: F_{envt} and F_{task}.

This should demonstrate that the task-space controller will provide feedback when F_{task} is applied, joint-space control will compensate for F_{envt} and both controllers will minimise control effort when tracking performance is reasonable. Measurement noise is simulated through the injection of white noise into the calculation of τ_{dist} from Equation 4.6. Noise levels are bounded to levels similar to those described for the Baxter robot in [218], and as such does not cause noticeable variation between simulation runs; therefore no statistical analysis is required.

For evaluation purposes a performance index η was calculated from the integral of the product of input force F_u and task-space tracking error ε_x:

$$\eta = \int_{t_s}^{t_f} F_u(t)^T \Upsilon F_u(t) + \varepsilon_x^T(t) \Psi\ \varepsilon_x(t)\ \mathrm{d}t \tag{4.13}$$

where $\Upsilon, \Psi \in \Re^{6\times 6}$ are positive diagonal scaling matrices, and t_s and t_f were set to obtain η for each phase of the simulation. A low performance index η corresponds to low tracking error and control effort, thus indicating specifications are met.

Controller parameters were chosen similar to the previous test as:
$Q_\tau = \mathrm{diag}(7,\ 7,\ 7,\ 7,\ 7,\ 7,\ 0.1)$, $Q_K = \mathrm{diag}(50,\ 50,\ 50,\ 50,\ 50,\ 50,\ 0.1)$, $Q_D = \mathrm{diag}(1,\ 1,\ 1,\ 1,\ 1,\ 1,\ 0.1)$; for the task-space controller, $Q_F = \mathrm{diag}(7,\ 7,\ 7,\ 7,\ 7,\ 7)$,

$Q_{K_x} = \text{diag}(50,\ 50,\ 50,\ 50,\ 50,\ 50)$, $Q_{D_x} = \text{diag}(1,\ 1,\ 1,\ 1,\ 1,\ 1)$. The stability margin was set as $L = \mathbf{I}^{7\times7}$, $\alpha = 0.5$ and $\kappa = 10$. Simulations were all set to run with a 1 kHz update rate. The initial position of the end-effector, of the left arm, was set to $X(0) = [0.6,\ 0.2,\ 0.4,\ 90,\ 0,\ 171.9]^T$, with a final target position of $X_r(t_f) = [0.6,\ 0,\ 0.4,\ 90,\ 0,\ 171.9]^T$ moving in a minimum-jerk trajectory as previously described in Equation 3.33.

4.3.1 Results

The performance of the hybrid controller $\tau_u(t) = \tau_r(t) - L\ \varepsilon(t) + \tau_x(t) + \Omega\tau_j(t)$ was compared against the controller in joint-space only, when $\tau_u(t) = \tau_r(t) - L\ \varepsilon(t) + \tau_j(t)$, and in task-space only, where $\tau_u(t) - L\ \varepsilon(t) = \tau_r(t) + \tau_x(t)$. Disturbance parameters remain the same in each case; for $F_{task}(t)$ defined in Equation 4.8, $p = 20\sin(2\pi\,50\,t)$, and for $F_{envt}(t)$ from Equation 4.10 the parameters are $r = 100\sin(2\pi\,0.1042\,t)$. The trajectory period and travel distance were set to 4.8s and 0.2m respectively. Each simulation phase corresponds to one completion of the trajectory.

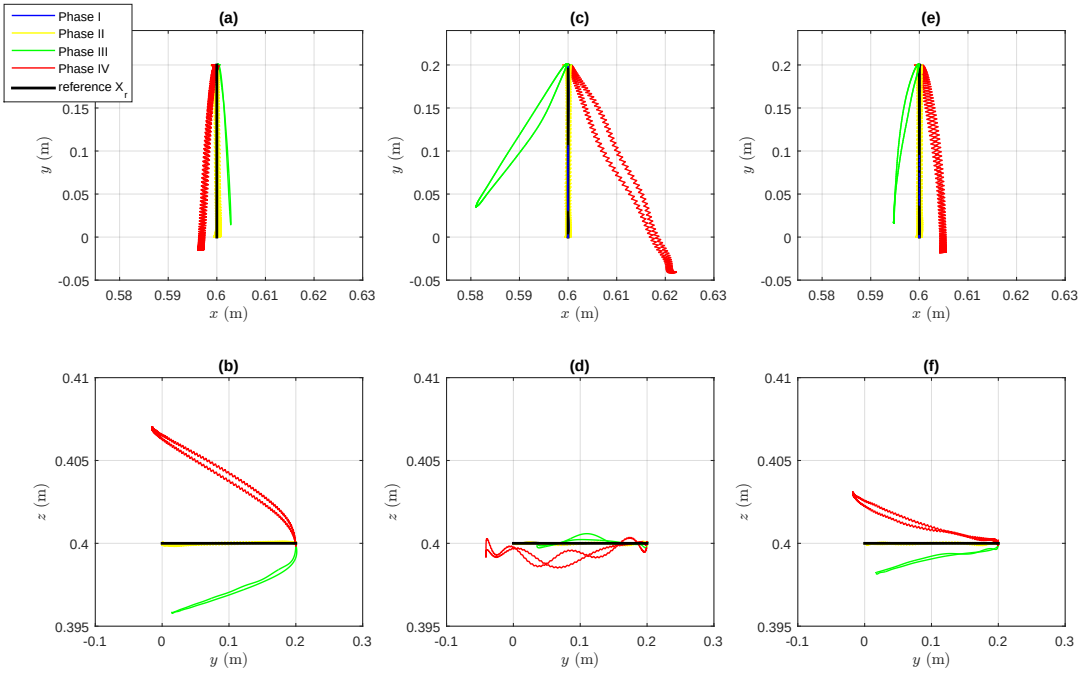

FIGURE 4.2: Comparing the cartesian trajectories of the three control schemes. The top row shows trajectories in the x, y plane, bottom row in the y, z plane. Graphs (a) and (b) correspond to joint-space control, (c) and (d) to task-space, (e) and (f) to the hybrid scheme.

The trajectory path for each controller is first analysed, and shown in Figure 4.2, with both the x, y and y, z planes shown separately for clarity. Looking at the joint space

controller in (a) and (b), we can see that the trajectory is followed well in the x, y plane, but deviates massively in the z-axis in phases III and IV. Comparing this to task-space control shown in (c),(d), we can see that the position in the x, y plane is pushed to either side in phases III and IV due to application of F_{envt} but has a better performance in the z-axis. We can also see that the task-space controller is perturbed less by F_{task} in phase II, and is able to follow the desired trajectory closer. What we can gather from comparing these two is that the joint-space controller can compensate for F_{envt} better than task-space, but suffers from poor performance in the axis which has no disturbance applied. By now examining the hybrid controller shown in (e) and (f), we can see it has taken the good points of both controllers; it is more robust against the large environmental task, similar to the joint-space, but also limits deviation in all three axes similar to the task-space controller.

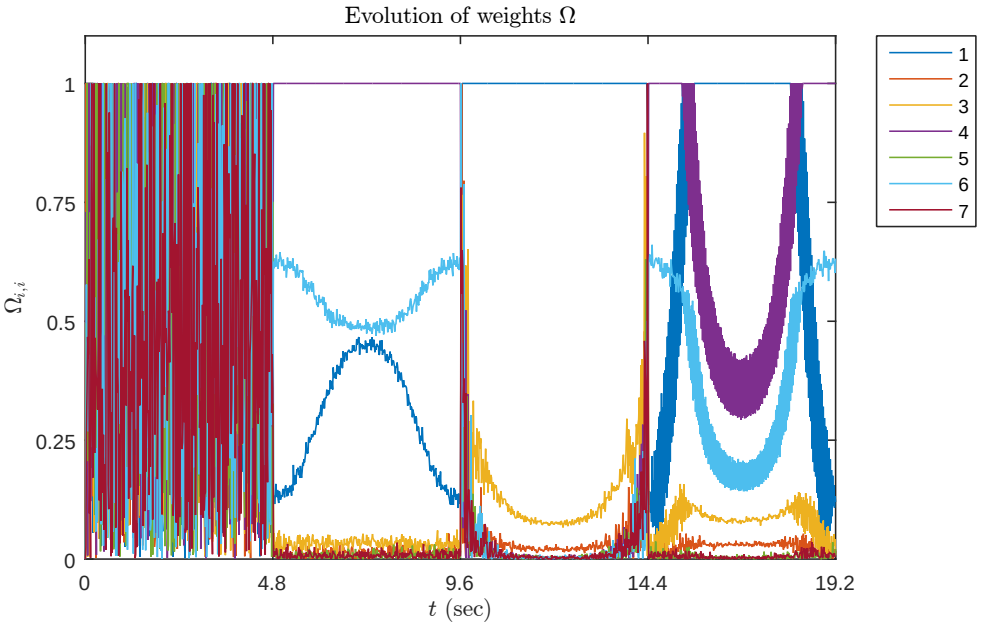

FIGURE 4.3: Plot of how the weighting matrix Ω changes over the simulation period.

The elements of the weighting matrix Ω are plotted in Figure 4.3, which, when compared in conjunction with Figure 4.1, indicates if τ_j is being applied as expected dependent on the type of disturbance. From the definitions of F_{task} and F_{envt} it can be expected that

- in phase II the vibration at the end-effector in the x-axis will cause higher errors (and therefore higher weights) on joints along the whole length of the arm, mostly in joints 6, 4 and 1 which have axes perpendicular to the direction of disturbance force,
- phase III disturbance is only applied at the elbow (joint 4), so weighting will be higher at joints 1,2 and 3,

- a combination of the above in phase IV where both are applied.

By examining Figure 4.3 we can see that Ω behaves as expected. The switch point between phases is very clear. In phase I the weights for all joints oscillate, as all errors will be at similar low values. In phase II the joints with highest weights are 4, 6 and 1 respectively, as predicted. Weights in phase III also match expectations, with highest values assigned to joints 1, 3 and 2 respectively. Finally in phase IV Ω_{max} is assigned to joints 1 and 4, with some smaller weighting assigned to joints 6, 3 and 2. Through all three disturbance phases the weights are close to zero for joints 5 and 7, as desired in the design.

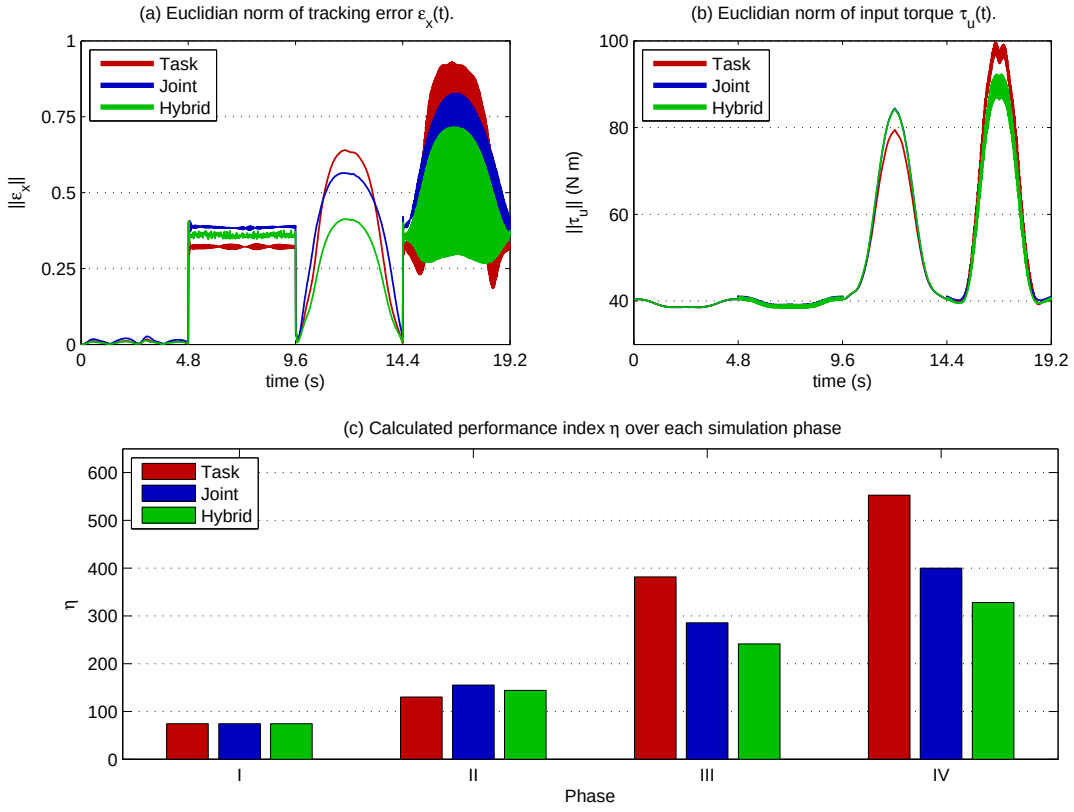

FIGURE 4.4: Comparison of joint-space, task-space and hybrid controller performance, (a): Comparing tracking error, (b): Input torque, and therefore control effort, and (c): performance indices, a combination of both.

The Cartesian tracking error ε_x in Figure 4.4(a), for all three controllers, shows how task-space performs better when a tool-type disturbance is applied, but suffers when a large disturbance is applied away from the end-effector; in this case, joint-space control was able to track more effectively. When combined in the hybrid controller, tracking error was reduced further. In the first phase, $(0 < t < 4.8)$ little difference can be observed in tracking error for the three controllers. In phase II task-space has the lowest error and joint-space the highest, with the hybrid control in between, as expected due to the

perturbation being located at the end-effector. In the next two phases ($9.6 < t < 19.2$) task-space control produces the highest error, while the hybrid controller shows a much lower tracking error than its component parts.

From Figure 4.4(b) it can be noted that there was little difference in the overall amount of control effort being applied between the three methods. The measures of tracking error and control effort were combined to form the performance index η for each phase, shown in Figure 4.4(c). A clear difference could be seen in the performances of the task-space and joint-space controllers between phases II and III, where the disturbance type was switched from F_{task} to F_{envt}; task-space control was better at handling the former, and joint-space the latter. The hybrid controller showed a slight improvement over joint-space in phase II but exhibited an improvement over its component parts in phases III and IV. Considering $||\tau_u||$ was similar for all three, as seen in Figure 4.4(b), this suggests that the hybrid control was applying control in a more targeted fashion, i.e. only applying additional feedback to the joints which require it.

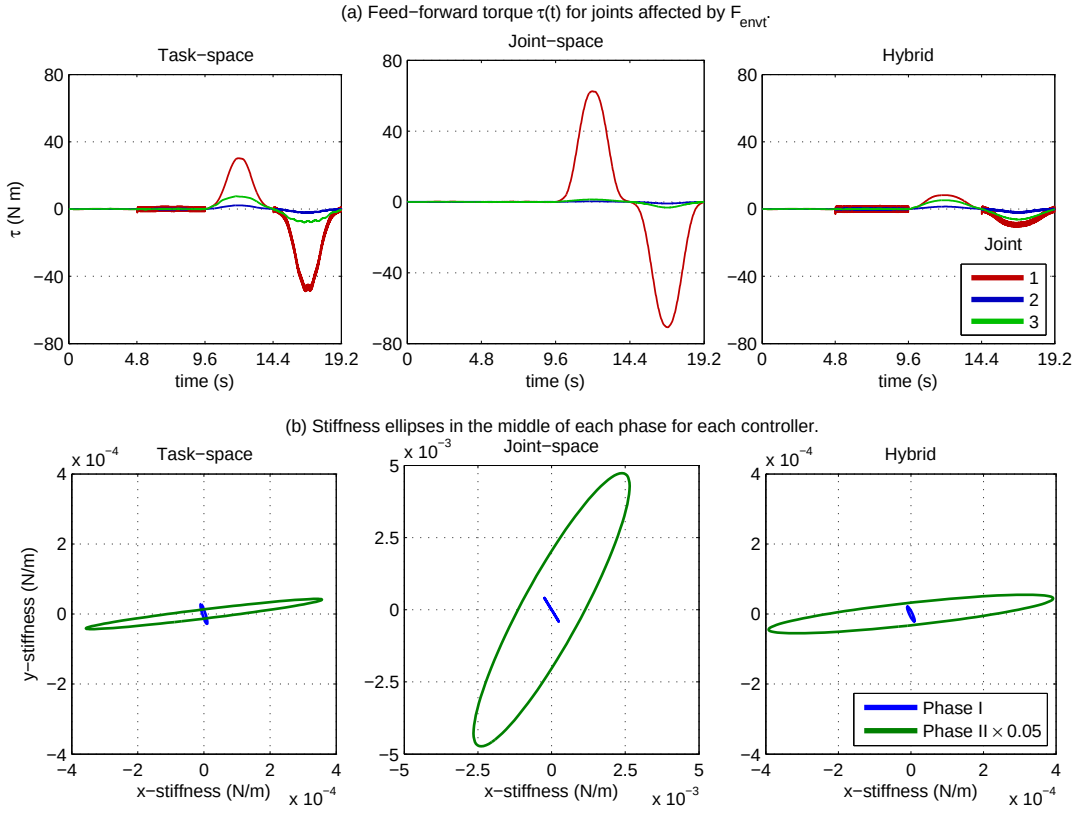

FIGURE 4.5: Learned feed-forward torque and stiffness. (a): Evolution of the feed-forward torques for the first three joints. (b): Stiffness ellipses in the x and y planes, of midpoint of phases I - II.

By examining the evolution of feed-forward torque in Figure 4.5(a) we see how in phases III and IV large increases were made in the last two phases to compensate for the low

frequency F_{envt} disturbance, predominantly in the first joint (the rotation of which is aligned with the x, y plane). Comparing the magnitude of feed-forward torque between controllers it is clear that joint-space control generated much higher torques, while hybrid control torques were much lower and less weighted towards joint 1.

Cartesian stiffness ellipses can be used to examine the geometry of feedback, and are shown in Figure 4.5(b). In task-space and hybrid control, it can be observed how the stiffness changed from a slight orientation in the y-direction (due to the trajectory moving along this axis) to a much larger ellipse predominantly in the x-axis: aligned with the direction of disturbance. Joint-space control, however, produced ellipses less-aligned with the direction of disturbance. This shows that feedback torque is being applied inefficiently in this case.

4.4 Concluding Remarks

This chapter has discussed the design of a hybrid task/joint-space controller and its implementation. A simulation experiment was carried out and results collected to demonstrate the effectiveness.

We can see clearly from the resulting task-space trajectories in Figure 4.2 how the hybrid controller follows a path which is somewhere in between the joint-space and task-space controllers, with the benefits of both; that is, the joint space controller improves robustness against the environmental disturbance force, and the task-space controller improves trajectory tracking and robustness to end-effector disturbance.

Examination of how the weighting matrix changes in 4.3 shows that the weighting matrix works as expected in reducing joint-space torque in joints where low disturbance is detected. It should be noted that the weighting is necessary for stability, as preliminary experiments showed that simple summation or averaging of the two control torques resulted in excessive applied torque. As the weighting matrix only reduces the torque from the joint-space controller, it can be said that the majority of control is provided by the task-space control but with some "bootstrapping" provided in joint-space.

In addition to improved performance, we can see in Figure 4.4 that the overall input torque, equivalent to metabolic cost, is the same or less compared to the component controllers; this tells us that the available torque is being applied more "correctly" to improve the overall controller performance. This is confirmed when examining the reference geometry in Figure 4.5 where we can see that the stiffness ellipse is better aligned with the disturbance direction than in joint-space.

As these simulations were carried out, it was noted that setting and tuning of the controller parameters (for example, Q_τ, α, etc.) was both time consuming and tedious; the parameter was set, the simulation run, then the output analysed for tuning. This is what leads us to the next chapter: the automation of this trial-and-error process.

Chapter 5

Application of Fuzzy Controller for Parameter Estimation

5.1 Introduction

Controller parameters, such as PID gains (Proportional, Integral and Derivative), are typically tuned by the user during primary testing to get the desired performance from the system. Tuning can be aimed at, for example, reducing tracking error, increasing the learning rate of an adaptive algorithm or increasing the compliance of an impedance controller. Tuning the parameters for a robotic manipulator is no easy task, as complex dynamics and cross-talk interactions may be affected unexpectedly by small changes of an initial gain. In addition, a robot manipulator may have interaction with the environment, which can be impossible to predict and model.

Some research has shown how using neural networks system uncertainties can be estimated [219, 220] to avoid some of these problems. Fuzzy logic, however, has often been used in aspects where human operator expertise can be transferred into a mathematical representation, to automate rational decision making in the face of imprecise data [125, 221, 222]. For many years fuzzy control has been used in systems to improve performance, as discussed in Chapter 2, and more recently applied to non-linear systems [223] and robot manipulator control [224].

In this chapter, fuzzy control is used in a novel application to self-tune the hybrid controller discussed in Figure 4. In particular, the adaptation gains are auto-tuned to improve the reaction to sudden perturbations and increase the rate at which control effort is reduced when tracking performance is good. The hybrid controller has uncertain and complex dynamics which are problematic to apply traditional tuning methods to,

and linguistic rules exist for the desired controller performance, e.g. as the user I can decide whether the tracking error is "good" or "worse"; this is a good match for a fuzzy system.

5.2 Preliminaries of Fuzzy Control

Before continuing with the fuzzy system design, it is important to know the fundamentals. What is described here is focused on the Mamdani, rather than Sugeno [225], type of fuzzy system, due to its being more intuitive and well suited to human input.

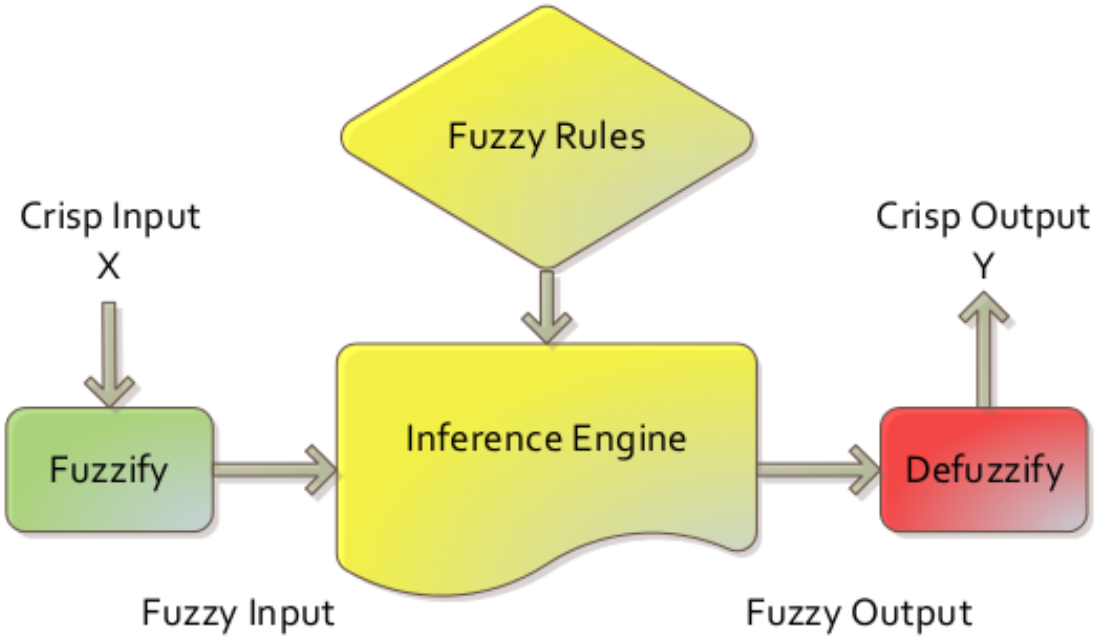

FIGURE 5.1: Simple flow diagram of fuzzy system. The input X is fuzzified through membership functions. The output sets are generated by the inference engine based on the fuzzy rules. Y is the "real world", or crisp, output which is retrieved through the defuzzification of the output fuzzy set [1].

As shown in Figure 5.1, there are three main steps for get a crisp output from a crisp input or inputs. First, fuzzification maps the inputs (for example, in an air conditioning system these could be temperature and humidity) into fuzzy space through the use of membership functions. Membership functions can take a number of shapes such as trapezoidal, gaussian, etc. and choice is usually subjective, but commonly the triangular shape is used for simplicity and low sensitivity [226]. If we let X be a space of points with elements $x \in X$, a fuzzy set A in X can be described by the membership function $\mu_A(x)$ [123]. This associates a grade of membership $\mu_A(x_i)$ in the interval $[0, 1]$ to any point x in set A.

Several technical definitions are required to describe the method of inference. Corresponding to the descriptive connection OR, a *union* of two sets A and B is a fuzzy set C

$$C = A \cup B; \quad \mu_C(x) = \max\big[(\mu_A(x), \mu_B(x)\big],\ x \in X. \tag{5.1}$$

An intersection is used to describe the connective AND, which can be similarly described:

$$C = A \cap B; \quad \mu_C(x) = \min\big[(\mu_A(x), \mu_B(x)\big],\ x \in X. \tag{5.2}$$

A relation between two or more fuzzy sets can be described by the Cartesian product. Given A, a set in universe X, and B, a set in universe Y, [227] the Cartesian product of A and B will result in a relation R

$$A \times B = R \subset X \times Y \tag{5.3}$$

where the fuzzy relation R has a membership function

$$\mu_R(x, y) = \mu_{A \times B}(x, y) = \min\big[\mu_A(x), \mu_A(y)\big]. \tag{5.4}$$

This is used in the Mamdani min-implication to relate an input set to an output set, i.e.

$$\text{IF } x \text{ is } A \quad \text{THEN } y \text{ is } B. \tag{5.5}$$

Through this method a series of rules can be described linguistically, for example, if *temperature* is *high* and *humidity* is *high* then *AC output* is *high*. The rule set implicates the system fuzzy output from their aggregated output. Aggregation is the process of combining the fuzzy set output of each rule into a single fuzzy set, commonly by taking the maximum value from all rules across the output set. This results in a mapping of the output in fuzzy space.

Several methods exist for defuzzification, but the centroid method is commonly used for accuracy [228]. The defuzzified value (or crisp output) y^* is given through calculation of the centre of mass of the aggregated output fuzzy set shown in Equation 5.6, and relates the value μ back to a real-world value (such as fan speed, for the running example).

$$y^* = \frac{\int \mu_B(y)\, y \, \mathrm{d}x}{\int \mu_B(y) \, \mathrm{d}x}. \tag{5.6}$$

5.3 Application to Biomimetic Controller

As discussed in the previous chapter, an apparent issue the requirement for arbitrary learning parameters to be set by the user. This problem is addressed here through a novel application of a fuzzy system to take the linguistic rules of a control engineer to infer these parameters online, based on the performance criteria of tracking error and control effort. The system gains to be inferred are set to some low value initially, to ensure a stable simulation run. Data is collected during this initial run, which then forms a performance baseline. The next run of the simulation then compares the current performance to the previous, but this time tunes the gains according to the fuzzy system using a set of linguistic rules. In this way performance is hoped to improve run-on-run, as the fuzzy engine seeks to continuously improve upon the previous iteration. The gains to be tuned are related to the learning rate, $Q_{(\cdot)}$ and $\alpha_{(\cdot)}$, and are chosen for relation to improving tracking error and control effort respectively. Metrics of these two criteria are calculated online to provide inputs to the fuzzy system.

For the hybrid controller a separate fuzzy system is created for both the joint-space and task-space components, and although they follow the same rules the output membership functions are defined differently. The basis of the inputs for fuzzification are the joint-space tracking error ε_j defined in Equation 3.22 and τ_j from Equation 3.23, and correspondingly in task-space ε_x and F_u from Equations 4.1,4.3. Before fuzzification the inputs are normalised so that the same fuzzy system can be applied regardless of input magnitude. This is possible due to how inference is made comparing the current input to the metrics from the previous iteration, which we will call the baseline. This baseline is formed from an average of tracking errors $\hat{\varepsilon}_j \in \Re^n$, $\hat{\varepsilon}_x \in \Re^6$, input torque $\hat{\tau}_u \in \Re^n$ and input force $\hat{F}_u \in \Re^6$ for each degree of freedom, over the total simulation time per time step ${}^{t_f}/_{\delta t}$, i.e.

$$\hat{\varepsilon}_{j_i} = \frac{\sum |\varepsilon_{j_i}(t)|}{{}^{t_f}/_{\delta t}}, \quad \hat{\varepsilon}_{x_i} = \frac{\sum |\varepsilon_{x_i}(t)|}{{}^{t_f}/_{\delta t}}, \quad \hat{\tau}_{u_i} = \frac{\sum |\tau_{u_i}(t)|}{{}^{t_f}/_{\delta t}}, \quad \hat{F}_{u_i} = \frac{\sum |F_{u_i}(t)|}{{}^{t_f}/_{\delta t}} \tag{5.7}$$

which are then used to normalise the corresponding values from the current iteration, as such:

$$\bar{\varepsilon}_{j_i}(t) = \frac{\sigma\ |\varepsilon_{j_i}(t)|}{\hat{\varepsilon}_{j_i}}, \quad \bar{\varepsilon}_{x_i}(t) = \frac{\sigma\ |\varepsilon_{x_i}(t)|}{\hat{\varepsilon}_{x_i}}, \quad \bar{\tau}_{u_i}(t) = \frac{\sigma\ |\tau_{u_i}(t)|}{\hat{\tau}_{u_i}}, \quad \bar{F}_{u_i}(t = \frac{\sigma\ |F_{u_i}(t)|}{\hat{F}_{u_i}} \tag{5.8}$$

where σ is set to 0.5 for our purposes. The result of this is the simplification of the input membership functions; for example, if the current tracking error $\varepsilon_{j_i}(t)$ is better (i.e. lower) than the baseline, then $\bar{\varepsilon}_{j_i}(t)$ will be in the region $0 \rightarrow 0.5$, and correspondingly if performance is worse than the previous iteration then the normalised input will be

in the region $0.5 \rightarrow 1$ (assuming it is not more than double the baseline). Therefore, for simplicity only three input classifications are needed: "lower", "same" and "higher" which are spaced equally across the input range of $0 \rightarrow 1$. As there is no upper bound to the normalisation the membership functions are defined so that any crisp input greater than 1 is assigned a membership value of 1 in the "higher" classification.

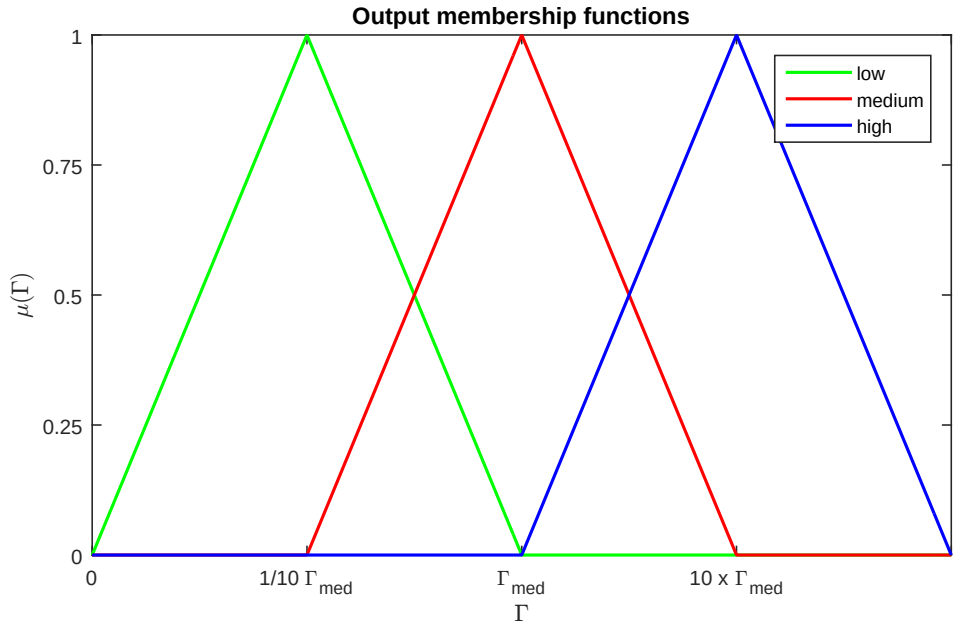

FIGURE 5.2: Graph showing how output membership functions are set, where Γ can be replaced with Q_τ etc. and Γ_{med} is the selected "medium" value.

We then select our gains to be inferred to be $Q_\tau \equiv Q_\tau(\bar{\varepsilon}_j, \bar{\tau}_j)$, $Q_K \equiv Q_K(\bar{\varepsilon}_j, \bar{\tau}_j)$, $Q_D \equiv Q_D(\bar{\varepsilon}, \bar{\tau})$, $\alpha \equiv \alpha(\bar{\varepsilon}_j, \bar{\tau}_j)$ from Equations 3.26 and 3.28 for the joint-space controller, the corresponding task-space gains from Equations 4.2 and 4.4. The membership functions for these are selected based on expert knowledge of the system; an average value is selected as a "medium" value, with "high" and "low" values set as half and double the set value, respectively. The triangular membership functions are then set as shown in Figure 5.2, which simplifies the process. Note that this also demonstrates the bounding of the output gains, which will be important later when discussing stability of the system.

The rules are described using expert knowledge of the system, but are fairly general, for example: IF control effort is too high THEN gain is set low; IF tracking error is poor THEN gain is set high, as shown in Table 5.1 for $Q_{(\cdot)}$. Note also that if the control effort is high then the $Q_{(\cdot)}$ gains are set to a medium level. The truth table for the forgetting factor gain (Table Table 5.2) is slightly different, in that α is required to be larger when tracking error is improving. This can also be visualised in Figure 5.3, which also demonstrates how the rule sets differ for $Q_{(\cdot)}$ gains in (a) from those commanding α in (b).

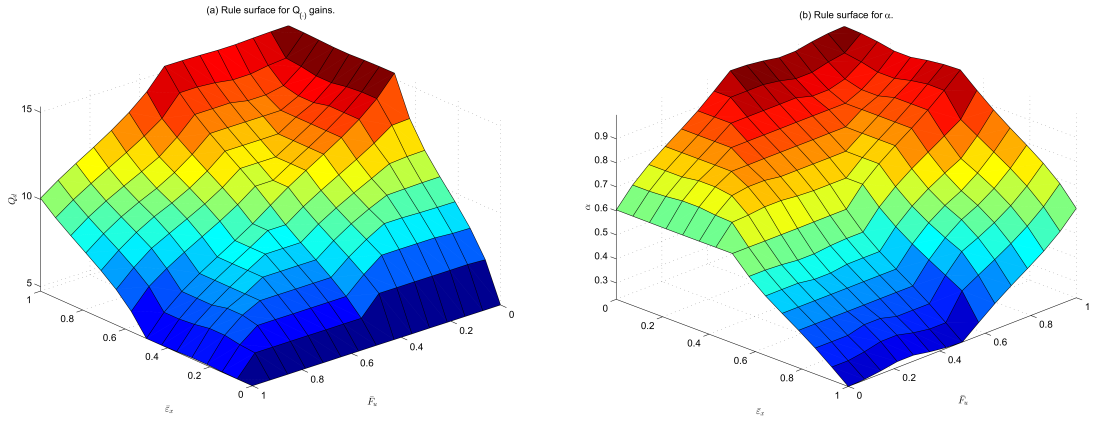

FIGURE 5.3: Gradient maps demonstrating rule surfaces. (a): adaptation gain Q_{Dx}, and (b): value of α_x, based on inputs $\bar{\varepsilon}_x$ and $\bar{F}_u$ described in Figure 5.8. Joint-space gains are characterised by similar surfaces.

TABLE 5.1: Truth table for inference of output $Q_{(.)}$ based on fuzzy memberships of $\bar{\varepsilon}_{j_i}, \bar{\varepsilon}_{x_i}, \bar{\tau}_{u_i}, \bar{F}_{u_i}$.

		Input				Output
R_1:	**IF**	$\bar{\varepsilon}_j, \bar{\varepsilon}_x < \sigma$			**THEN**	$Q_{(.)}$ *low*
R_2:	**IF**	$\bar{\varepsilon}_j, \bar{\varepsilon}_x \approx \sigma$	and	$\bar{\tau}_u, \bar{F}_u < \sigma$	**THEN**	$Q_{(.)}$ *low*
R_3:	**IF**		and	$\bar{\tau}_u, \bar{F}_u \geq \sigma$	**THEN**	$Q_{(.)}$ *medium*
R_4:	**IF**	$\bar{\varepsilon}_j, \bar{\varepsilon}_x > \sigma$	and	$\bar{\tau}_u, \bar{F}_u \leq \sigma$	**THEN**	$Q_{(.)}$ *high*
R_5:	**IF**		and	$\bar{\tau}_u, \bar{F}_u > \sigma$	**THEN**	$Q_{(.)}$ *medium*

TABLE 5.2: Truth tables for inference of output $\alpha_{(.)}$ based on fuzzy memberships of $\bar{\varepsilon}_{j_i}, \bar{\varepsilon}_{x_i}, \bar{\tau}_{u_i}, \bar{F}_{u_i}$.

		Input				Output
R_1:	**IF**	$\bar{\varepsilon}_j, \bar{\varepsilon}_x < \sigma$	and	$\bar{\tau}_u, \bar{F}_u < \sigma$	**THEN**	α *medium*
R_2:	**IF**		and	$\bar{\tau}_u, \bar{F}_u \geq \sigma$	**THEN**	α *high*
R_3:	**IF**	$\bar{\varepsilon}_j, \bar{\varepsilon}_x \approx \sigma$	and	$\bar{\tau}_u, \bar{F}_u \leq \sigma$	**THEN**	α *medium*
R_4:	**IF**		and	$\bar{\tau}_u, \bar{F}_u > \sigma$	**THEN**	α *high*
R_5:	**IF**	$\bar{\varepsilon}_j, \bar{\varepsilon}_x > \sigma$	and	$\bar{\tau}_u, \bar{F}_u \leq \sigma$	**THEN**	α *low*
R_6:	**IF**		and	$\bar{\tau}_u, \bar{F}_u > \sigma$	**THEN**	α *medium*

5.4 Stability Analysis

Stability of the biomimetic controller described in Chapter 3 and convergence to a small bounded set were shown in [16], with the task-space controller having a similar proof; the small changes made (such as decoupling of $Q_.$ and γ) do not affect the same proof. However, with the application of the fuzzy system the adaptation gain matrices $Q_{(.)}$ are now time varying and must be accounted for in the stability proof.

From [16] Appendix C, the difference in energy of the system $\delta V(k) = \delta V_p(t) + \delta V_c(t)$ is shown to converge to zero. No change to the derivation of the first part $\delta V_p(t)$ is required (apart from being in continuous time rather than iterative), so that section of

the proof still holds, and is given as:

$$\begin{aligned}\delta V_p &= V_p(t) - V_p(t-T)\\ &\leq \int_{t-\delta t}^{t} -\varepsilon^T(\sigma)L(\sigma)\varepsilon(\sigma) - \varepsilon^T\tilde{K}(\sigma)e(\sigma) - \varepsilon^T(\sigma)\tilde{D}(\sigma)\dot{e}(\sigma) - \varepsilon^T(\sigma)\tilde{\tau}(\sigma)\\ &- \varepsilon^T(\sigma)K_E e - \varepsilon^T D_E\dot{e} - \varepsilon^T(\sigma)\tau_E + \varepsilon^T\tau_I \,\mathrm{d}\sigma\\ &\leq \int_{t-\delta t}^{t} -\varepsilon^T(\sigma)L(\sigma)\varepsilon(\sigma) - \varepsilon^T\tilde{K}(\sigma)e(\sigma) - \varepsilon^T(\sigma)\tilde{D}(\sigma)\dot{e}(\sigma) - \varepsilon^T(\sigma)\tilde{\tau}(\sigma)\,\mathrm{d}\sigma. \end{aligned} \tag{5.9}$$

The present controller differs in [16] Equations (39 - 41) where $Q_{(\cdot)}^{-1}$ is replaced with $Q_{(\cdot)}^{-1}(\sigma)$. The original inequality is given as:

$$\begin{aligned}\delta V_c(t) = \frac{1}{2}\int_{t-\delta t}^{t} \Big\{ &\mathrm{tr}\left(\tilde{K}^T(\sigma)Q_K^{-1}\tilde{K}(\sigma) - \tilde{K}^T(\sigma-\delta t)Q_K^{-1}\tilde{K}(\sigma-\delta t)\right)\\ &+\mathrm{tr}\left(\tilde{D}^T(\sigma)Q_D^{-1}\tilde{D}(\sigma) - \tilde{D}^T(\sigma-\delta t)Q_D^{-1}\tilde{D}(\sigma-\delta t)\right)\\ &+\tilde{\tau}^T(\sigma)Q_\tau^{-1}\tilde{\tau}(\sigma) - \tilde{\tau}^T(\sigma-\delta t)Q_\tau^{-1}\tilde{\tau}(\sigma-\delta t)\Big\}\,\mathrm{d}\sigma. \end{aligned} \tag{5.10}$$

Due to the time-varying nature of $Q_{(\cdot)}^{-1}$ in this work, the inequality is redefined as:

$$\begin{aligned}\delta V_c(t) = \frac{1}{2}\int_{t-\delta t}^{t} \Big\{ &\mathrm{tr}\left(\tilde{K}^T(\sigma)Q_K^{-1}(\sigma)\tilde{K}(\sigma) - \tilde{K}^T(\sigma-\delta t)Q_K^{-1}(\sigma-\delta t)\tilde{K}(\sigma-\delta t)\right)\\ &+\mathrm{tr}\left(\tilde{D}^T(\sigma)Q_D^{-1}(\sigma)\tilde{D}(\sigma) - \tilde{D}^T(\sigma-\delta t)Q_D^{-1}(\sigma-\delta t)\tilde{D}(\sigma-\delta t)\right)\\ &+\tilde{\tau}^T(\sigma)Q_\tau^{-1}(\sigma)\tilde{\tau}(\sigma) - \tilde{\tau}^T(\sigma-\delta t)Q_\tau^{-1}(\sigma-\delta t)\tilde{\tau}(\sigma-\delta t)\Big\}\,\mathrm{d}\sigma. \end{aligned} \tag{5.11}$$

Defining a new variable $\delta Q \equiv \mathrm{diag}\left[\mathrm{I}\otimes\delta Q_K, \mathrm{I}\otimes\delta Q_D, \mathrm{I}\otimes\delta Q_\tau\right]$ (where $\otimes$ is the Kronecker product) allows us to add another term to the end of [16](44), producing

$$\begin{aligned}\delta V_c(t) = &-\frac{1}{2}\int_{t-\delta t}^{t}\delta\tilde{\Phi}^T(\sigma)Q^{-1}(\sigma)\delta\tilde{\Phi}(\sigma)\,\mathrm{d}\sigma - \int_{t-\delta t}^{t}\gamma(\sigma)Q^{-1}(\sigma)\tilde{\Phi}^T(\sigma)\Phi(\sigma)\,\mathrm{d}\sigma\\ &+\int_{t-\delta t}^{t}\varepsilon^T(\sigma)\tilde{K}(\sigma)e(\sigma) + \varepsilon^T(\sigma)\tilde{D}(\sigma)\dot{e}(\sigma) + \varepsilon^T(\sigma)\tilde{\tau}(\sigma)\,\mathrm{d}\sigma\\ &+\int_{t-\delta t}^{t}\tilde{\Phi}^T(\sigma)\delta Q^{-1}(\sigma)\tilde{\Phi}(\sigma)\,\mathrm{d}\sigma. \end{aligned} \tag{5.12}$$

The term inside the last integrand can be described by $\varepsilon_Q\tilde{\Phi}^T\tilde{\Phi}$ where

$$\varepsilon_Q\tilde{\Phi}^T\tilde{\Phi} \geq \mathrm{tr}\left(\varepsilon_K\tilde{K}^T\tilde{K} + \varepsilon_D\tilde{D}^T\tilde{D} + \varepsilon_\tau\tilde{\tau}^T\tilde{\tau}\right), \quad \varepsilon_Q = \max\left(\varepsilon_K, \varepsilon_D, \varepsilon_\tau\right) \tag{5.13}$$

given that $\tilde{K}^T\Phi_K^{-1}\tilde{K} \leq \varepsilon_K\tilde{K}^T\tilde{K}$, $\tilde{D}^T\Phi_D^{-1}\tilde{D} \leq \varepsilon_D\tilde{D}^T\tilde{D}$ and $\tilde{\tau}^T\Phi_\tau^{-1}\tilde{\tau} \leq \varepsilon_\tau\tilde{\tau}^T\tilde{\tau}$, where $\varepsilon_{K,D,\tau}$ are defined as the minimum eigenvalues of $\Phi_{K,D,\tau}^{-1}$. This can then be added to the condition in [16](46) which gives the inequalities

$$\delta V \geq \lambda_L\|\varepsilon\|^2 + \bar{\gamma}_{max}\|\tilde{\Phi}\|^2 - \gamma'\|\tilde{\Phi}\|\|\Phi^*\| \geq \lambda_L\|\varepsilon\|^2 + \bar{\gamma}\|\tilde{\Phi}\|^2 - \gamma'\|\tilde{\Phi}\|\|\Phi^*\| \geq 0 \quad (5.14)$$

where $\gamma' = Q^{-1}\gamma$, and $\bar{\gamma} = \gamma' + \varepsilon_Q$ and $\lambda_{min}\big(L(t)\big) \leq \lambda_L > 0$. Following LaSalle's theorem [229] the terms $\|\varepsilon\|^2$ and $\|\tilde{\Phi}\|$ will converge to an invariant set $\Upsilon_s \subseteq \Upsilon$ in which $\delta V(t) = 0$, and Υ is the bounding set defined by:

$$\Upsilon \equiv \left\{ (\|\varepsilon\|^2, \|\tilde{\Phi}\|), \frac{(\lambda_L\|\varepsilon\|^2) + \alpha(\|\tilde{\Phi}\| - \|\Phi^*\|)}{(\lambda_L\|\varepsilon\|^2)^\alpha} \leq 1 \right\}. \quad (5.15)$$

Given that $Q(t)$ is bounded by the output of fuzzy inference as shown in Figure 5.2 and that γ is always a positive value asymptotic to zero, as shown in Figure 3.10, then convergence will always be attainable and the value of γ will only affect convergence speed and the size of the set.

5.5 Experimental Verification

To test the effectiveness of the fuzzy tuning of parameters the same simulation was run as seen in the previous chapter. No other changes were made apart from the addition of online parameter tuning the existing hybrid controller. The only parameters that needed to be set were the medium values of the inferred gains (see Γ_{med} in Figure 5.2), which were taken as the set values from the previous experiment described in Section 4.3. In the case of the parameter α, it is bounded as described in Equation 3.28 so the output membership function is distributed equally across the range. The data collected from that experiment was also used to form the performance baselines described in Equation 5.7. The same performance indices were calculated for the fuzzy tuned controller, to allow for direct comparison.

5.5.1 Results

The resulting task-space trajectory in Figure 5.4 is compared to the same controller with fixed gains. In phase II, shown in yellow, the fuzzy tuning has made a clear improvement. Interestingly the trajectory in the y, z plane in (d) was improved the most, with only small divergence seen in the z-axis and a smaller overshoot in the direction of movement (y-axis). However, comparing Figure 5.4(a) and (c) a higher divergence is noted in the

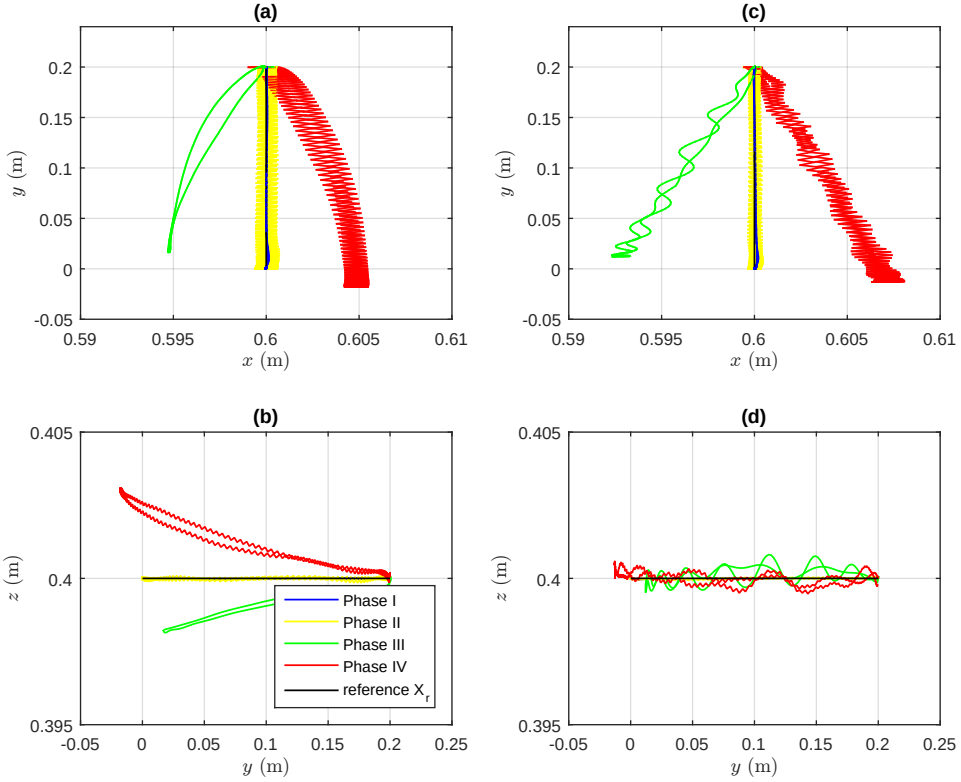

FIGURE 5.4: Comparison of task-space trajectories of the hybrid controller: with static gains in (a)(b), fuzzy tuned gains in (c) and (d).

x-axis for the fuzzy tuned controller with some oscillation seen particularly in phase III, most likely due to the adaptation gains competing to reduce the control effort and tracking error at the same time.

Figure 5.5 shows how the tuned gains change over time in joint-space and task-space, with clear differences between phases. For all $Q_{(\cdot)}$ gains in phase I the values are much lower than the midpoint value, with α increasing to reduce control effort. In phase II we can see some increases made to certain joints in (a), (b), (c), and in all three axes x, y, z in (e),(f) and (g), particularly in the x-axis aligned with disturbance. In the last two phases when disturbance forces are greatest the $Q_{(\cdot)}$ gains show large increases over the midpoint values, while α is reduced to allow the amount of feedback to accumulate faster, therefore hopefully compensating better against F_{dist}.

We can examine the tracking error and control effort more clearly in Figure 5.6. In (a) there is clearly a reduction in error in phase II but little change in all other phases. However, looking at the control effort in (b) we can see almost no change in phases I and II, but significant reduction in phases III and IV. This demonstrates that online tuning is minimising the tracking error when control effort is low, but is also able to

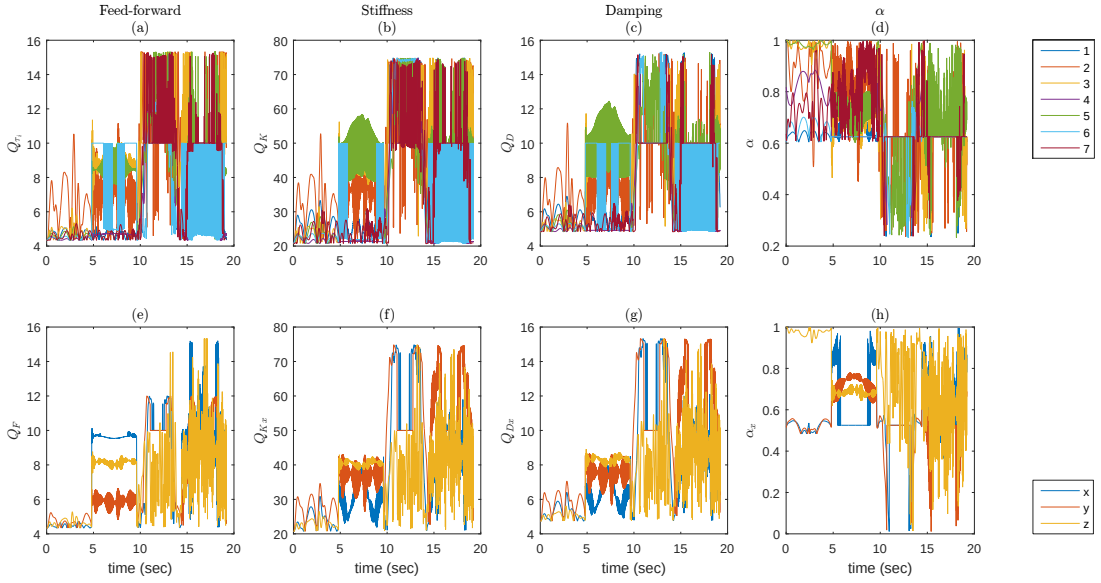

FIGURE 5.5: Evolution of fuzzy tuned gains over simulation period, joint-space above and task-space below. Feed-forward terms in (a),(e); stiffness in (b),(f); damping in (c),(g) and alpha gains in (d),(h).

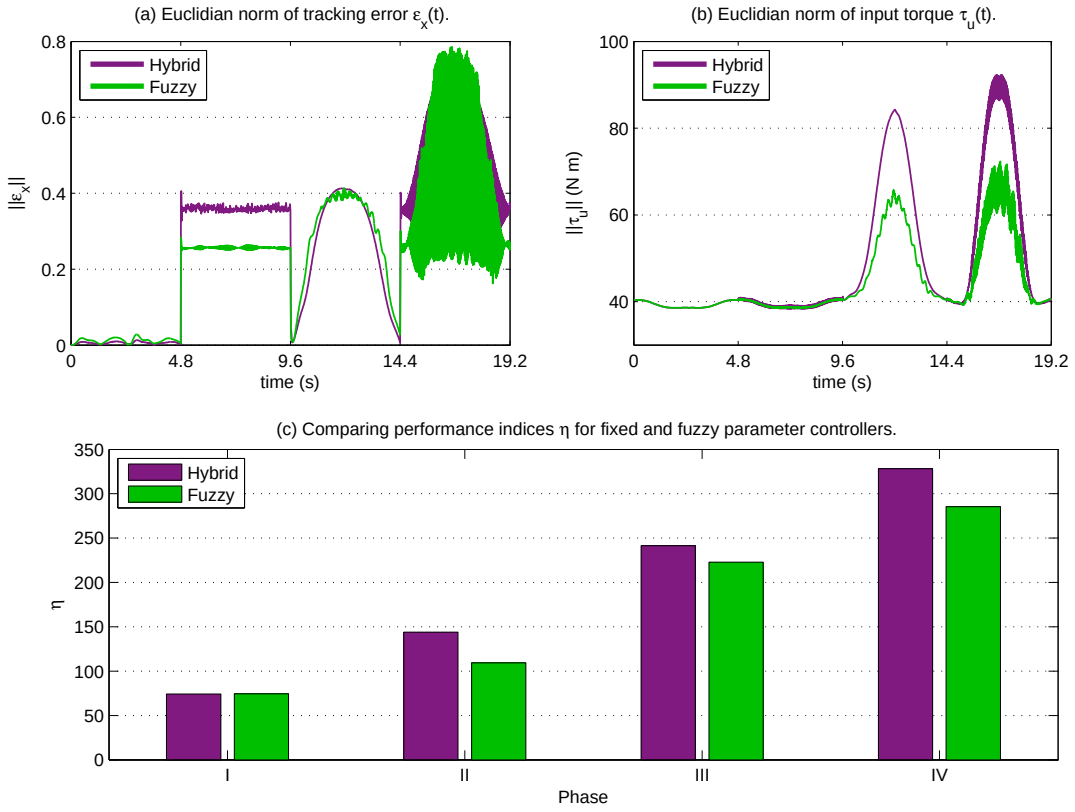

FIGURE 5.6: Performance of controller with online parameter tuning compared to fixed parameters. (a): Euclidean error over simulation time, (b): Euclidian of input torque and (c): performance indices η for each phase.

reduce the amount of control effort while maintaining the same level of tracking error. This is reflected in the resulting performance indices in (c): in phase I performance is similar as both tracking error and control effort cannot be reduced much further, in all other phases performance is improved in either possible way when the gains are tuned online.

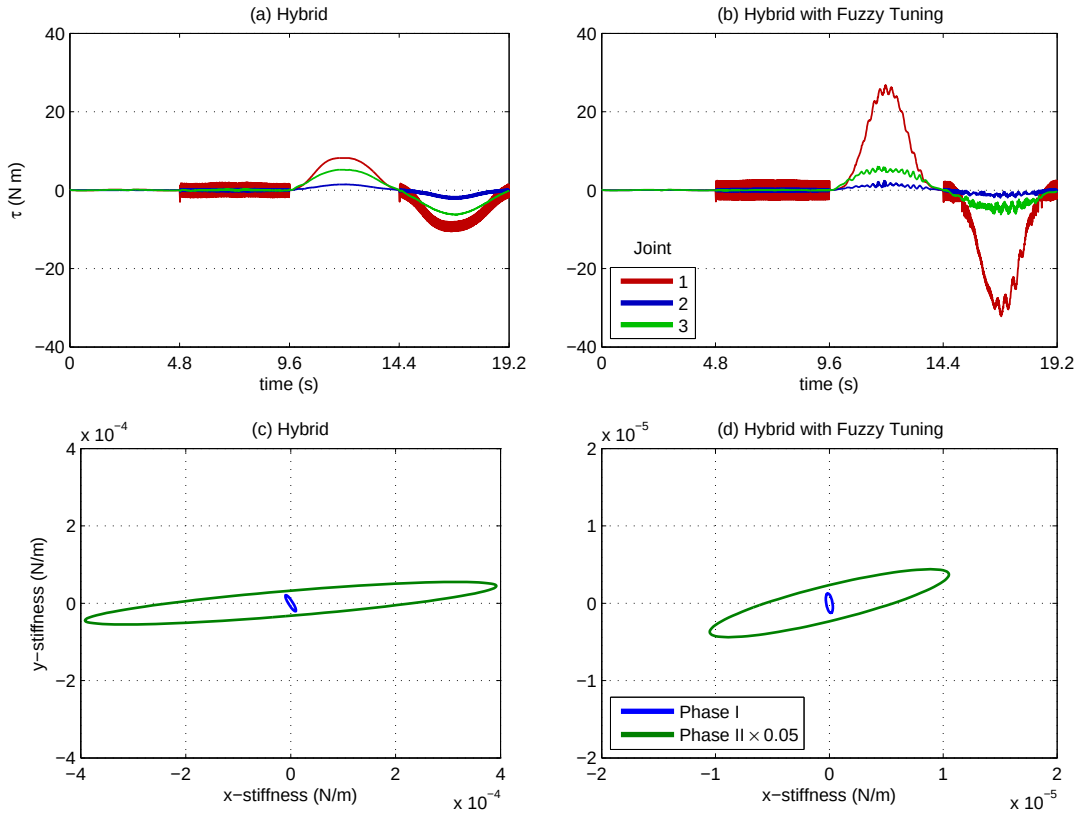

FIGURE 5.7: Comparing the evolution of feed-forward force and stiffness of fixed gain in (a) and (c), and fuzzy tuned controller in (b) and (d).

The feed-forward torques of the proximal joints are compared in Figure 5.7(a) and (b). When gains are tuned online we can see that the torque in the first joint is increased dramatically when F_{envt} is applied in phases III and IV, and a slight reduction in joints 2,3. Some oscillation occurs as the fuzzy rules attempt to minimise the performance criteria simultaneously. Examining the change in stiffness geometry between phases I and II (note scaling for second phase, so that the shape in phase I can be seen) it can be observed that stiffness is still aligned similarly between fixed gain and tuned gain controllers, but in the tuned case the magnitude is more than twenty times smaller. From this we can see that although feed-forward torque is increased dramatically in one joint (in (b)), applied torque is reduced in other areas (stiffness feedback), i.e. application of torque is being targeted in required areas more efficiently.

5.6 Concluding Remarks

This chapter has investigated how fuzzy control techniques have been applied in the past and what types of application are suited to fuzzy control over other techniques. The prerequisite mathematics of a Mamdani-type fuzzy controller are given. A description of how the fuzzy system can be applied to the hybrid biomimetic controller from Chapter 4, with a verification of stability is shown, followed by an experimental verification of online parameter tuning using the same experimental platform as before for direct comparison. The new controller is analysed for performance criteria and how the online tuning affects the gains corresponding to learning and forgetting rates.

From the results we can see that although some performance improvements are made in terms of tracking error and control effort, by creating two sets of rules for tuning $Q_{(\cdot)}$ gains and α gains there is some interaction where one set is trying to increase gain to reduce error, while the other is trying to minimise gain and therefore control effort. However despite these competing affects we can see from the performance indices in Figure 5.6 that improvements are still being made when disturbance forces are applied. Overall, control effort was reduced by 24% and performance increased by 15% when the controller parameters are tuned online. This is an added bonus to the original specification, which was to avoid the time-consuming task of tuning parameters offline from repeated simulation runs.

This concludes the description of the manipulator controller, so that in Chapter 6 it is applied to a simulated bimanual simulated task in the face of complex disturbance forces and analysed for suitability and performance.

Chapter 6

Application of Controller to a Bimanual Task

6.1 Introduction

Bimanual, that is, with two manipulators, robots have become more common in the world, as manufacturers look towards a more fully humanoid design. Well known robots like Honda's Asimo [230] and the KAIST HUBO [231] are being designed to fit in with our world and work with us in everyday environments, but are mainly focused on bipedal walking development. Others, such as the iCub and Justin described in Section 2.3 are being developed for research in cognition and Human-Robot Interaction (HRI) experiments. The Baxter robot, also previously described, has been designed with two arms to improve its capabilities in the workplace, but is also useable in a research scenario.

The problems inherent to bimanual manipulator control are various and several solutions appear in the literature, described in Section 2.3. In the scope of this work a particular bimanual task is focused on, where both hands are required to use a tool with dynamic properties (as of a running motor), whilst under the influence of environmental disturbance such as coming into contact with a human operator. Minimal tracking error must be maintained to avoid dropping the tool and performing the task, but the system must remain compliant for effective contact tooling [178, 232] and operator safety [191, 233, 234].

The previous chapters have described the development of a controller suited to this task. The basis of the controller, the biomimetic design, allows force and impedance of the manipulator to adapt independently, ideal for the described task. The minimisation of

control effort ensures natural compliance, but also means that impedance is minimised in directions not aligned with that of disturbances.

This chapter describes a method of bimanual control to perform the aforementioned task, and experimental results to confirm suitability. The developed controller is implemented to demonstrate its effectiveness in this situation.

6.2 Object dynamics

The object, or tool, is modelled as a mass-spring-damper system, which additionally exerts a high frequency, low amplitude disturbance force normal to the point of contact with the hand. For simplicity but without loss of generality the object is assumed to be a homogenous sphere, as shown in Figure 6.1.

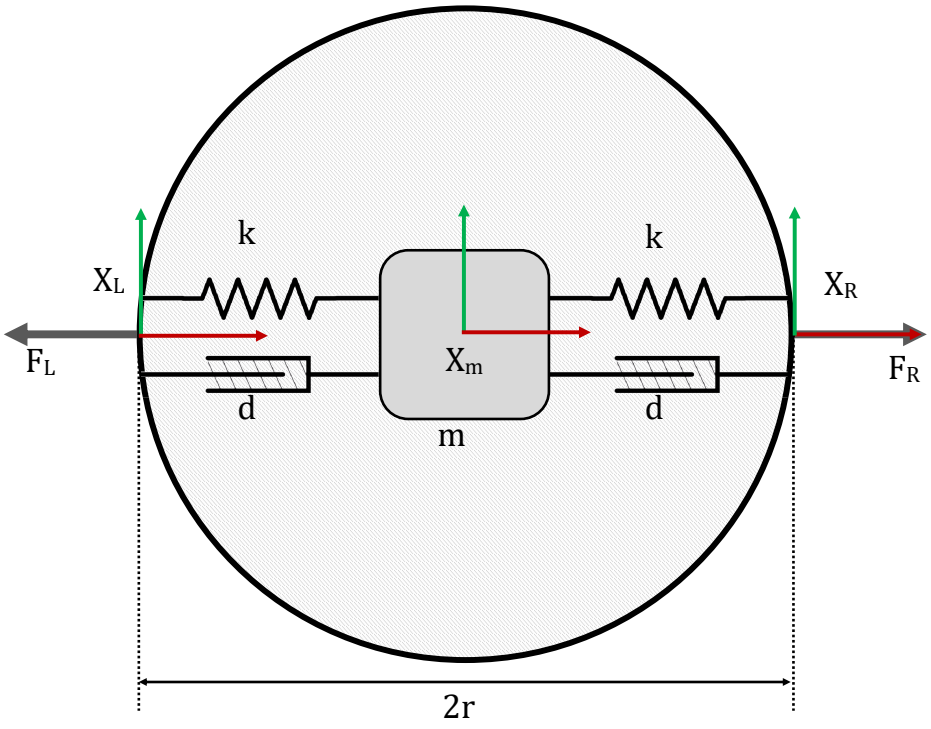

FIGURE 6.1: Diagram of dynamic object, showing structure of mass-spring-damper system.

The interaction forces which result from contact with the object are described with reference to the Cartesian positions and velocities of the central mass $(X_m, \dot{X}_m)$ and the

contact points of the left $(X_L, \dot{X}_L)$ and right $(X_R, \dot{X}_R)$ end-effectors [235], as

$$\begin{aligned} F_R &= -k\big(r - (X_m - X_R)\big) - d(\dot{X}_R - \dot{X}_m) \\ F_L &= k\big(r - (X_L - X_m)\big) - d(\dot{X}_m - \dot{X}_L) \end{aligned} \tag{6.1}$$

where k, d and r are the stiffness and damping coefficients and the natural radius of the sphere respectively. The forces F_L and F_R are the sum of forces arising from interaction of the object mass m and the end-effectors against the spring-damper system distributed equally across the surface; from Newton's Second Law of motion we have:

$$-\, m\, \ddot{X}_m = F_L + F_R \tag{6.2}$$

which, when combined with Equation 6.1, allows us to combine the terms to:

$$-\, m\, \ddot{X}_m = 2d\, \dot{X}_m + 2k\, X_m - d(\dot{X}_L + \dot{X}_R) - k(X_L + X_R). \tag{6.3}$$

A state space representation of the dynamics is now constructed for clarity and easier modelling. The mass position and velocity become the states with the end-effector positions and velocities as inputs, as so:

$$\begin{bmatrix} \dot{X}_m \\ \ddot{X}_m \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{-2k}{m} & \frac{-2d}{m} \end{bmatrix} \begin{bmatrix} X_m \\ \dot{X}_m \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{k}{m} & \frac{k}{m} & \frac{d}{m} & \frac{d}{m} \end{bmatrix} \begin{bmatrix} X_L \\ X_R \\ \dot{X}_L \\ \dot{X}_R \end{bmatrix}, \tag{6.4}$$

and the outputs:

$$\begin{bmatrix} F_L \\ F_R \end{bmatrix} = \begin{bmatrix} k & d \\ k & d \end{bmatrix} \begin{bmatrix} X_m \\ \dot{X}_m \end{bmatrix} + \begin{bmatrix} -k & 0 & -d & 0 \\ 0 & -k & 0 & -d \end{bmatrix} \begin{bmatrix} X_L \\ X_R \\ \dot{X}_L \\ \dot{X}_R \end{bmatrix} + \begin{bmatrix} k\, r \\ -k\, r \end{bmatrix}. \tag{6.5}$$

In addition, a simple friction model is employed, where the force normal to the surface F_n must keep the condition:

$$F_n \geq \frac{m\, \mathbf{g}}{\mu_f} \tag{6.6}$$

where μ_f is the coefficient of friction and $\mathbf{g}$ is the acceleration due to gravity.

In addition to the forces resulting from contact with the object, two disturbance forces are also presented to the manipulators as per the task described in Section 6.1; these are the same as F_{dist} as described in Equation 4.8 and Equation 4.10, except that in

the case of F_{task} the direction is normal and opposite the contact surface, and F_{envt} is normal to the direction of travel, and only applied to the leading arm.

6.3 Bimanual Trajectory Design

Examining previous works, it was decided that the work of Sugar and Kumar [150] and other similar literature [236–238] shows a promising starting point for the design of the bimanual trajectory system. A leader-follower methodology means that a constant distance can be maintained between the hands despite perturbations being applied to the leading arm. With respect to the controller design here, it also means that the complete system can remain compliant while the following arm is stiff, to compensate for disturbance.

One manipulator is designated the leader, the other the slave; for the remaining descriptions, the role of *master* is given to the left manipulator and *slave* to the right; where the subscripts $(\cdot)_L$ and $(\cdot)_R$ denote any variable which corresponds to that arm. Therefore, the trajectories of both arms can be defined

$$\begin{aligned} X_{L_d}(t) &= \left[x_{L_d}(t),\ y_{L_d}(t),\ z_{L_d}(t),\ \vartheta_{L_d}(0),\ \phi_{L_d}(0),\ \psi_{L_d}(0)\right]^T \\ X_{R_d}(t) &= \left[x_L(t) + p_{x_d},\ y_L(t) + p_{y_d},\ z_L(t) + p_{z_d},\ \vartheta_R(0),\ \phi_R(0),\ \psi_R(0)\right]^T \\ \dot{X}_{R_d}(t) &= \dot{X}_L(t) \end{aligned} \tag{6.7}$$

where $X_{L_d}(t)$ and $X_{R_d}(t)$ denote the *desired* Cartesian positions of the left and right manipulator respectively, $X_L(t)$ and $X_R(t)$ the corresponding *actual* positions, and $p_d = [p_{x_d},\ p_{y_d},\ p_{z_d}]$ are the positional offsets between the hands, which are set so that $||p_d|| \leq 2r$ from Equation 6.1 but also fulfilling the inequality in Equation 6.6, dependent on the stiffness and mass of the object, assuming a coefficient of friction $\mu_c = 0.9$, approximately the same as the rubber often found on the handles of small tools [239].

6.4 Experimental Verification

To test the controller in the bimanual task described above a modelled simulation is carried out with two Baxter manipulators, modelled to match the actual robot in terms of relative position and orientation. The adaptive controller is applied to both left and right unchanged from that described in Chapter 5, with initial parameters set the same on both sides. The disturbance forces are introduced in the same order as before, with the small difference described in Section 6.2.

To compare performance the same simulation is run using a computed torque control method, where the input torque is defined as

$$\tau_u = M(q)\big(\ddot{q}_d + K(q_d - q) + D(\dot{q}_d - \dot{q})\big) + C(q, \dot{q})\dot{q} + G(q), \tag{6.8}$$

where $K = 50$ and $D = 5$. This is a simple compliant design, only working in joint-space and with no adaptive control.

The desired trajectory of the leading arm, X_{L_d}, also remains the same as previously, with the desired trajectory of the following arm corresponding to Equation 6.7 and $p_d = [0,\ -0.4,\ 0]$. The parameters of the object are set to $r = 0.204$m, $m = 0.2$kg, $k = 10$N/m, $d = 5$Ns/m.

6.4.1 Results

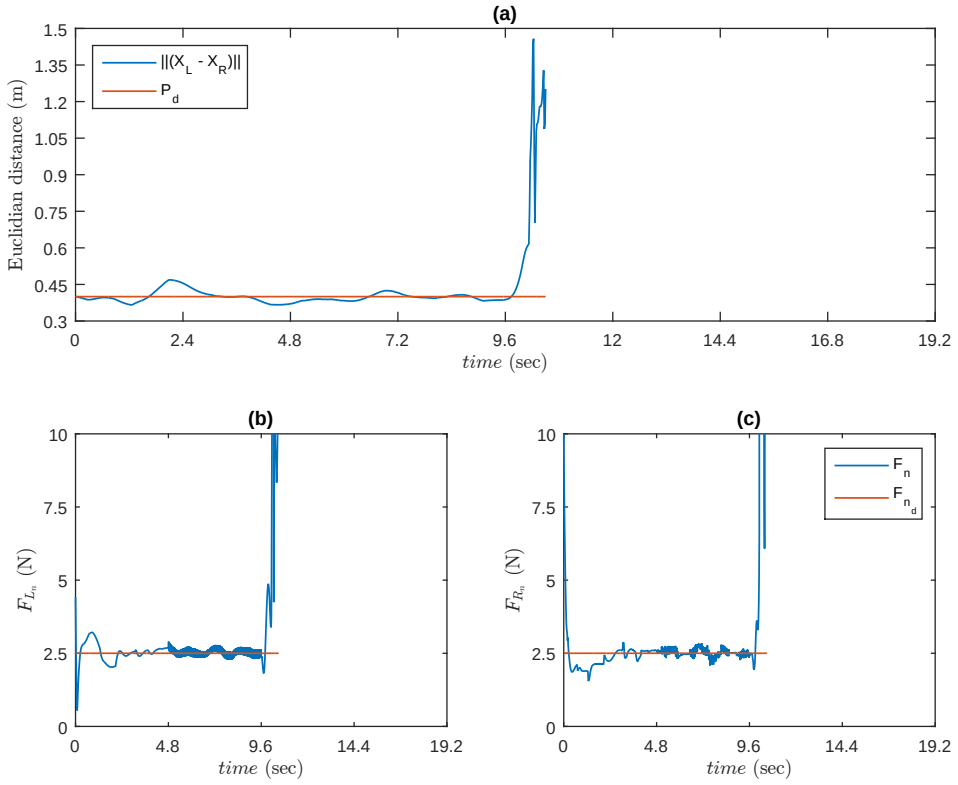

FIGURE 6.2: Results from simulated task with computed torque control input. (a): Euclidian distance between end-effectors, (b): normal force at left end-effector, (c): normal force at right end-effector.

The computed torque control method is tested first, with performance shown in Figure 6.2. The controller was not able to compensate sufficiently when the disturbance force F_{envt} at $t = 9.6$, and becomes unstable. In the first two phases $0 < t < 9.6$ we can

see in (a) that the distance between the end-effectors is maintained poorly, fluctuating both above and below the desired distance p_d. Note that the object would be dropped if this distance is greater than 0.408 (natural diameter of the object), which occurs around $1.7 < t < 3$ and again around $6.6 < t < 7.3$ before the system becomes unstable. We can also examine the contact forces for left and right arms in Figure 6.2(a) and (b) respectively, where the applied force must be at least F_{n_d} to provide enough friction to counteract gravity. We can observe that this condition is often not fulfilled by both arms, even when no disturbance forces are applied at $0 < t < 4.8$.

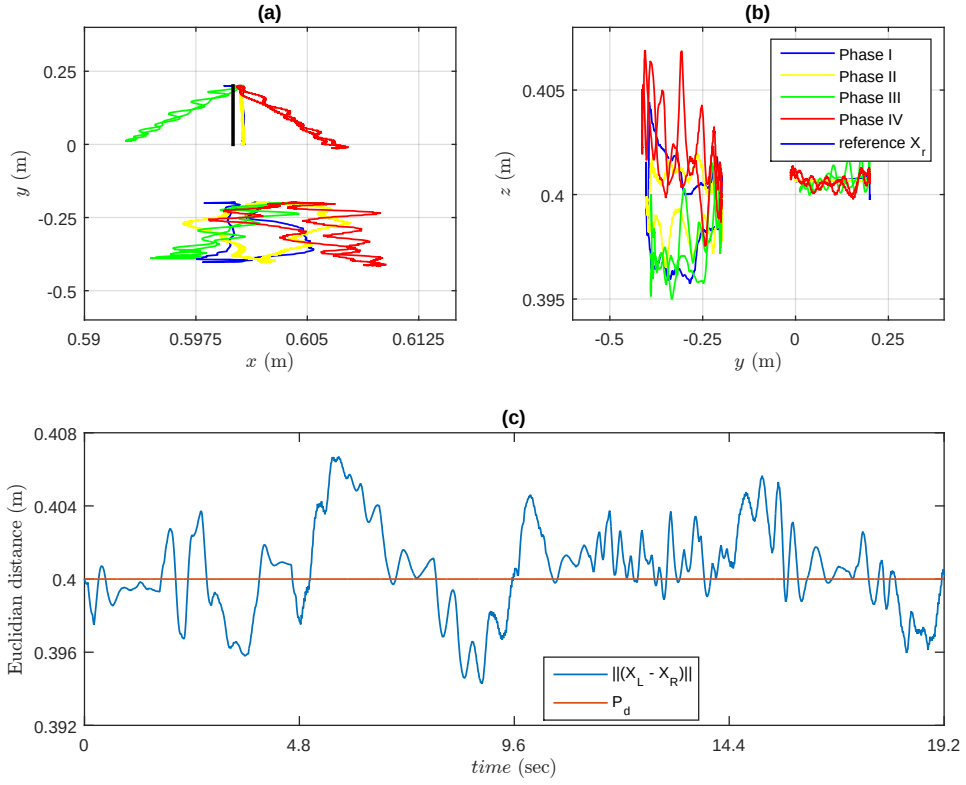

FIGURE 6.3: Resulting task-space trajectories showing both left and right arms; x, y plane in (a), y, z plane in (b), and Euclidian distance between manipulators in (c).

The task-space trajectories of both manipulators are plotted together in Figure 6.3(a) and (b), where the right arm deviates much more from the desired trajectory than the left arm. However, we can also see that the range of deviation in both x and z planes is approximately 0.01m. The important distance for successfully carrying the object without dropping it is shown in Figure 6.3 i.e. the Euclidian distance between end-effectors. The natural diameter of the object is equal to the top limit of the graph y-axis, with the target distance shown in red. As shown, the distance between end-effectors never becomes too large to drop the object for the entirety of the simulation time.

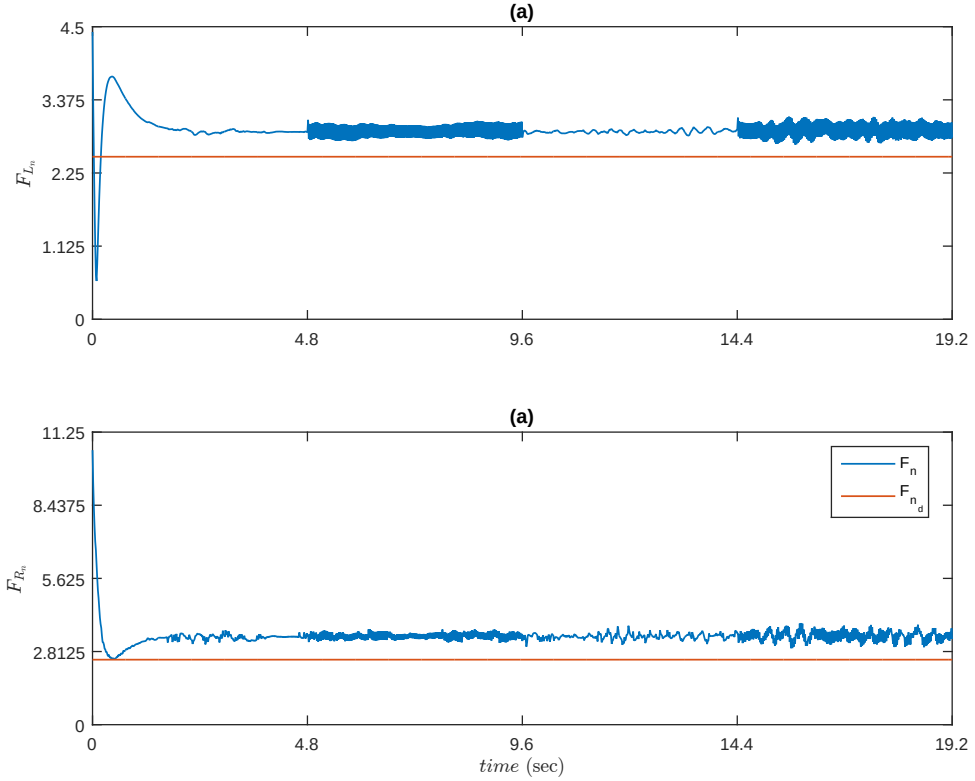

FIGURE 6.4: Normal forces acting on surface from left arm (a) and right (b), with desired normal force for sufficient friction shown in red.

In addition to the distance between the end-effectors the force normal to the ball surface, in Figure 6.4, must also be sufficient to provide enough friction to stop the object from slipping through the grasp. For the left manipulator we can see that initially the force is too low but quickly reaches and settles at a level where applied force is at maximum 0.35N above required, enough to hold without dropping. For the right side, the normal force is initially too high and drops close to the dropping threshold, but settles quickly, with a maximum of 0.91N. The maximum sum of forces acting on the object at any one point is 6.91N.

6.5 Concluding Remarks

This chapter has shown how the controller developed in previous chapters is able to successfully perform a bimanual task, where two manipulators are required to carry a compliant object through a trajectory, whilst under the influence of disturbances at the end-effector and the manipulator length. The dynamics of a compliant object are described, along with a state space representation for use in modelling. A friction model is also described to ascertain if contact forces are enough to realistically overcome the

weight of the object. A simple bimanual trajectory is then designed based on a master-slave system, with justification given.

A simulated experiment was carried out to test the capabilities of the controller designed in previous chapters. Results were first collected from a computed torque control method, which was not only unable to maintain a acceptable distance between end-effectors or contact forces, but also became unstable when environmental disturbances were applied. In comparison, the controller from Chapter 5 was not only able to maintain an acceptable distance between the hands in Figure 6.3 but also maintain stability and sufficient but not excessive contact force in Figure 6.4 to overcome the weight of the object.

Chapter 7

Conclusions and Future Aims

7.1 Summary of Thesis and Contributions

At the start of this work it was stated that the aim was to develop a controller based on a biomimetic design, to carry out the theoretical "hummingbird problem", which can also be analogously described as having to move a vibrating tool with two arms through a trajectory, whilst under the influence of unknown environmental disturbances.

Chapter 2 presented a review of works related to this field, including the development of bimanual robots and biomimetic adaptive controllers, based on human muscle impedance models. This led to the Baxter robot being chosen as the manipulator for further experiments, and the choice of a biomimetic controller as a starting point due to the properties of concurrent force and impedance adaptation, as well as the ability to reduce metabolic cost similar to relaxation of muscles.

Chapter 3 gives a comprehensive description of the calculations required to perform either L-E or RN-E inverse dynamics, and forms the first contribution of this work: a closed-form analytical model of the Baxter manipulator, available publicly and thought to be useful to the growing community of researchers using the Baxter robot as an experimental platform. Modelled dynamics are compared to readings collected from the physical robot, and show a close match, allowing simulations to be run with confidence in manipulator behaviour. A description of the biomimetic controller is then given, and describes some modifications to improve usability and performance. A simulated experiment was carried out to determine suitability for the original task, which was successful. This proves the suitability of the chosen biomimetic controller for pick-and-place tasks under natural environment-like disturbances. In addition, the results highlighted a drawback which led to the development of the next main contribution.

Chapter 4 forms the main contribution of this work, due to its novelty and possible applications in other control methods. Due to the nature of the biomimetic controller it was possible to combine task-space and joint-space controllers into a single hybrid via a weighting matrix, which improves overall performance when the manipulator is subjected to forces at the end-effector and also along the length of the arm. Results show that tracking error is improved without substantial increase in control effort. The designed weighting matrix is shown to successfully limit joint-space feedback to only the required joints. Implications of this are the possibility of applying the joint/task-space hybrid technique to other controllers as well as the biomimetic control, which may provide improved performance in other applications such as position or force control, although the particular nature of the biomimetic controller lends itself well to this type of hybridisation.

Chapter 5 describes another contribution of this work, that is, the novel implementation of fuzzy control systems to tune controller parameters online. This not only removes the need for lengthy tuning of the controller, but also shows slight improvement of controller performance under disturbance forces. Stability analysis is performed and deemed satisfactory. Results show that fuzzy tuning is able to reduce tracking error without increasing overall control effort, and also reduce control effort without incrementing tracking error, dependent on the type of disturbance applied. It is thought that methods similar to this could be applied to other controllers where setup includes the requirement of parameter tuning, but only in applications where it is possible for an expert operator to provide some sort of linguistic rule set.

finally in Chapter 6 the hybrid biomimetic controller with online parameter tuning is applied to a simulated experiment matching the original specification. This is compared to the same task carried out with a computed torque control method. A simple master-slave trajectory is designed for bimanual control. For the final contribution, results show that the controller is able to perform the task under all disturbance conditions while maintaining safe contact forces, which is shown to be not possible using a computed torque control method. However, it was noted during experimentation that the master-slave approach means that the slave arm is particularly sensitive to environmental forces, which at high levels can cause instability. More development is required to avoid this, and investigation of using different methods for bimanual coordination using the hybrid biomimetic controller. However, despite this the proposed approach is easy to implement and provides a degree of full system compliance, which may be applicable for certain tasks where high environmental disturbances can be avoided.

7.2 Proposals for Future Work

The largest and most regrettable limitation to the research here is that it has not been carried out in a real scenario, using a physical robot. This was due to unsuitability of the iCub robot, which was the only available platform at the start of this work. The Baxter robot only became available towards the end, which allowed for verification of the dynamic model seen in Chapter 3. It has been noted since experiments were carried out that the trajectories used for verification of the dynamics are limited to a small part of the workspace, so the dynamic model cannot be relied on in cases when the robot is operating outside of the tested area. Trajectories specifically designed for testing a dynamic model are well documented [240, 241], and should be used to truly verify the dynamic model. The controller developed in this work needs physical verification, to ensure it is capable of dealing with actual environmental disturbances and that the control loop is able to be computed at a good update rate. This would strengthen the conclusions presented in this work.

It is also suggested that the fuzzy tuning method described in Chapter 5 could be developed into an iterative tuner to improve control performance; as it relies on a baseline calculated from a previous iteration, the baseline could be re-calculated iteratively and iteratively attempt to improve performance. This could be an interesting avenue of research, and could be developed into a new form of fuzzy iterative learning.

Appendix A

MATLAB codes for inverse dynamics using the L-E and RN-E methods

This appendix contains details of the hardware, software and codes used for simulation studies.

A.1 Hardware and Software Versions

Table A.1: Hardware

Model	custom build PC
processor	2 x Intel Xeon 2GHz, 8 cores each
RAM	16GB

Table A.2: Software

Operating system	Ubuntu 14.04 LTE and Windows 7
MATLAB version	R2015a
Robotics Toolbox	9.10

A.2 Codes for Dynamic calculations in MATLAB

A.2.1 Lagrange-Euler closed-form of Baxter Manipulator

```
%% Form LE dynamics of Baxter arm (left arm)
% Uses symbolic method with Baxter kinematic and dynamic information to
% generate symbolic matrices representing closed form dynamics of the
% Baxter manipulator.

% Requires Peter Corke's robotics toolbox, at least version 9.10, and the
% function roundSymbolic() to be in the local folder or path. This is very
% memory intensive due to the large number of symbolic coefficients
% generated in the matrices. To aid in this, variables are saved to a .mat
% file after generation, and then cleared from RAM.

%% Preamble
clearvars
close all
clc

%% Load model and get primary variables:

Baxter;
% Number of joints:
n = Baxter_l.n;

q = sym('q',[1 n]); assume(q, 'real');
% velocity
qd = sym('qd',[1 n]); assume(qd, 'real');

% Account for any offsets
for i = 1:n
    q(i) = q(i) + Baxter_l.links(i).offset;
end

% All joints are revolute
a = zeros(1,n);
d = zeros(1,n);
alpha = zeros(1,n);
for i = 1:n
    a(i) = Baxter_l.links(i).a;
    d(i) = Baxter_l.links(i).d;
    alpha(i) = Baxter_l.links(i).alpha;
end

% Inertia
I = zeros(6,n);
for i = 1:n
  I(:,i) = [diag(Baxter_l.links(i).I); Baxter_l.links(i).I(1,2); Baxter_l.links(i).I(2,3); Baxter_l.links(i).I(1,3)];
end

% Mass
M = zeros(1,n);
for i = 1:n
    M(i) = Baxter_l.links(i).m;
end

% Centre of Mass
CoM = ones(4,n);
for i = 1:n
   CoM(1:3,i) = Baxter_l.links(i).r;
end

%% Inertia tensor (J)
J = zeros(4,4,n);
for i = 1:n
    J(:,:,i) = [0.5*(-I(1,i)+I(2,i)+I(3,i))                    I(4,i)                    I(6,i)  M(i)*CoM(1,i)
                                    I(4,i)  0.5*(I(1,i)-I(2,i)+I(3,i))                    I(5,i)  M(i)*CoM(2,i)
                                    I(6,i)                    I(5,i)  0.5*(I(1,i)+I(2,i)-I(3,i))  M(i)*CoM(3,i)
                             M(i)*CoM(1,i)              M(i)*CoM(2,i)              M(i)*CoM(3,i)           M(i)];
end
```

```
%% Constants
Q = zeros(4,4);
Q(1,2) = -1; Q(2,1) = 1;

%% Link transforms
T01 = [cos(q(1)),  -cos(alpha(1))*sin(q(1)),  sin(alpha(1))*sin(q(1)), a(1)*cos(q(1));
       sin(q(1)),   cos(alpha(1))*cos(q(1)), -sin(alpha(1))*cos(q(1)), a(1)*sin(q(1));
       0,           sin(alpha(1)),            cos(alpha(1)),           d(1);
       0,           0,                        0,                       1];

T12 = [cos(q(2)),  -cos(alpha(2))*sin(q(2)),  sin(alpha(2))*sin(q(2)), a(2)*cos(q(2));
       sin(q(2)),   cos(alpha(2))*cos(q(2)), -sin(alpha(2))*cos(q(2)), a(2)*sin(q(2));
       0,           sin(alpha(2)),            cos(alpha(2)),           d(2);
       0,           0,                        0,                       1];

T23 = [cos(q(3)),  -cos(alpha(3))*sin(q(3)),  sin(alpha(3))*sin(q(3)), a(3)*cos(q(3));
       sin(q(3)),   cos(alpha(3))*cos(q(3)), -sin(alpha(3))*cos(q(3)), a(3)*sin(q(3));
       0,           sin(alpha(3)),            cos(alpha(3)),           d(3);
       0,           0,                        0,                       1];

T34 = [cos(q(4)),  -cos(alpha(4))*sin(q(4)),  sin(alpha(4))*sin(q(4)), a(4)*cos(q(4));
       sin(q(4)),   cos(alpha(4))*cos(q(4)), -sin(alpha(4))*cos(q(4)), a(4)*sin(q(4));
       0,           sin(alpha(4)),            cos(alpha(4)),           d(4);
       0,           0,                        0,                       1];

T45 = [cos(q(5)),  -cos(alpha(5))*sin(q(5)),  sin(alpha(5))*sin(q(5)), a(5)*cos(q(5));
       sin(q(5)),   cos(alpha(5))*cos(q(5)), -sin(alpha(5))*cos(q(5)), a(5)*sin(q(5));
       0,           sin(alpha(5)),            cos(alpha(5)),           d(5);
       0,           0,                        0,                       1];

T56 = [cos(q(6)),  -cos(alpha(6))*sin(q(6)),  sin(alpha(6))*sin(q(6)), a(6)*cos(q(6));
       sin(q(6)),   cos(alpha(6))*cos(q(6)), -sin(alpha(6))*cos(q(6)), a(6)*sin(q(6));
       0,           sin(alpha(6)),            cos(alpha(6)),           d(6);
       0,           0,                        0,                       1];

T67 = [cos(q(7)),  -cos(alpha(7))*sin(q(7)),  sin(alpha(7))*sin(q(7)), a(7)*cos(q(7));
       sin(q(7)),   cos(alpha(7))*cos(q(7)), -sin(alpha(7))*cos(q(7)), a(7)*sin(q(7));
       0,           sin(alpha(7)),            cos(alpha(7)),           d(7);
       0,           0,                        0,                       1];

T00 = eye(4);
T11 = eye(4);
T22 = eye(4);
T33 = eye(4);
T44 = eye(4);
T55 = eye(4);
T66 = eye(4);
T77 = eye(4);
disp('Starting transform simplification... ')
tic
T02 = simplify(T01*T12);
T03 = simplify(T02*T23);
T04 = simplify(T03*T34);
T05 = simplify(T04*T45);
T06 = simplify(T05*T56);
T07 = simplify(T06*T67);

T13 = simplify(T12*T23);
T14 = simplify(T13*T34);
T15 = simplify(T14*T45);
T16 = simplify(T15*T56);
T17 = simplify(T16*T67);

T24 = simplify(T23*T34);
T25 = simplify(T24*T45);
T26 = simplify(T25*T56);
T27 = simplify(T26*T67);

T35 = simplify(T34*T45);
T36 = simplify(T35*T56);
T37 = simplify(T36*T67);

T46 = simplify(T45*T56);
T47 = simplify(T46*T67);
```

```
T57 = simplify(T56*T67);
toc


%% U matrix
disp('Starting U matrix calculation... ')
tic
for i = 1:n
    for j = 1:n
        if j <= i
            U(:,:,i,j) = eval(strcat('T',num2str(0),num2str(j-1))) * Q * eval(strcat('T',num2str(j-1),num2str(i)));
        else
            U(:,:,i,j) = zeros(4);
        end
    end
end
toc
disp('Starting U matrix simplification... ')
tic
U = simplify(U);
toc

%% 3D U matrices
disp('Starting U3 matrix calculation... ')
tic
for i = 1:n
    for j = 1:n
        for k = 1:n
            if (i >= k) && (k >= j)
                U3(:,:,i,j,k) = eval(strcat('T',num2str(0),num2str(j-1))) * Q *  ...
                eval(strcat('T',num2str(j-1),num2str(k-1))) * Q * eval(strcat('T',num2str(k-1),num2str(i)));
            elseif (i >= j) && (j >= k)
                U3(:,:,i,j,k) = eval(strcat('T',num2str(0),num2str(k-1))) * Q * ...
                eval(strcat('T',num2str(k-1),num2str(j-1))) * Q * eval(strcat('T',num2str(j-1),num2str(i)));
            else
                U3(:,:,i,j,k) = zeros(4);
            end

        end
    end
end
toc
disp('Starting U3 matrix simplification... ')
tic
U3 = simplify(U3);
toc
save('baseSymbols.mat', 'U', 'U3', 'J', 'n', 'qd', 'CoM', 'M')
clearvars

%% D matrix (inertia)

load('baseSymbols.mat', 'n', 'U', 'J')

disp('Starting D matrix calculation... ')
D = sym('D',[n n]);
tic
for i = 1:n
    for k = 1:n
        sum = 0;
        for j = max(i,k):n
            sum = sum + trace(U(:,:,j,k)*J(:,:,j)*U(:,:,j,i)');
        end
        D(i,k) = sum;%simplify(sum);
        fprintf('Rounding row %d, column %d', i,k)
        tic
        D(i,k) = roundSymbolic(D(i,k),6);
        toc
    end
end
fprintf('Total time: ')
toc
clearvars -except D
save inertiaD.mat
clearvars
disp('Starting D matrix simplification... ')
```

```
tic
D = simplify(D);
toc

%% h vector (coriolis)
disp('Starting H vector calculation... ')

load('baseSymbols.mat', 'n', 'U', 'U3', 'J', 'qd')

tic
% h3 = sym('h3',[n n n]);
for i = 1:n
    for k = 1:n
        for m_idx = 1:n
            sum = 0;
            for j = max([i, k, m_idx]):n
                sum = sum + trace(U3(:,:,j,k,m_idx) * J(:,:,j) * U(:,:,j,i)');
            end
            fprintf('h3: i= %d, k = %d, m = %d \n', i,k,m_idx)
            h3(i,k,m_idx) = sum;
        end
    end
end
clearvars -except h3 n qd

h = sym('h',[n 1]);
for i = 1:n
    sum_k = 0;
    for k = 1:n
        sum_m = 0;
        for m_idx = 1:n
            fprintf('h: i= %d, k = %d, m = %d \n', i,k,m_idx)
            sum_m = sum_m + h3(i,k,m_idx) * qd(k) * qd(m_idx);
        end
        sum_k = sum_k + sum_m;
    end
%     fprintf('Rounding row %d, column %d', i,k)
%     tic
%     h(i,1) = roundSymbolic(sum_k,6);
    h(i,1) = sum_k;
%     toc
end
fprintf('Total time: %d', toc)
clearvars -except h
save coriolisH.mat
clearvars
disp('Starting H vector simplification... ')
tic
% h = simplify(h);
toc

%% c vector (gravity)
load('baseSymbols.mat', 'n', 'U', 'qd')
g = 9.81;
g_vec = [0 0 -g 0];
disp('Starting c vector calculation... ')
tic
c = sym('c',[n 1]);
for i = 1:n
    sum = 0;
    for j = i:n
        sum = sum + (-M(j) * g_vec * U(:,:,j,i) * CoM(:,1));
    end
    fprintf('Rounding row %d', i)
    tic
    c(i,1) = roundSymbolic(sum,6);
    toc
end
fprintf('Total time: ')
toc
disp('Starting c vector simplification... ')
tic
c = simplify(c);
toc
clearvars -except c
```

```
save gravityC.mat
clearvars
```

A.2.2 Function to round large symbolic matrices

```
function [ simpEle ] = roundSymbolic(mEle, deg)
% roundSymbolic Round coefficients in symbolic expression
%   Takes a symbolic expression, breaks it down into coefficients and
%   terms, rounds to zero if less than specified degree.
[cof, trm] = coeffs(mEle);
num = length(cof);
fprintf('There are %d coeffs to round: \n', num)
parfor i = 1:length(cof)
    disp(i, ' of ', num)
    cof(i) = round(cof(i)*10^deg)/10^deg;
end
simpEle = dot(cof,trm);
end
```

A.2.3 Lagrange-Euler Computational Form of Baxter Dynamics

Note that this code could be easily adapted for any robot defined using the Robotics Toolbox.

```
function [M, C, G] = LEBaxterNumeric(q,qd,robot)
%% Form LE dynamics of robot arm
% Uses symbolic method with Baxter kinematic and dynamic information to
% generate symbolic matrices representing closed form dynamics of the
% Baxter manipulator.

%% Load model and get primary variables:

Baxter;
% Number of joints:
n = robot.n;

% Account for any offsets
for i = 1:n
    q(i) = q(i) + robot.links(i).offset;
end

% All joints are revolute
a = zeros(1,n);
d = zeros(1,n);
alpha = zeros(1,n);
for i = 1:n
    a(i) = robot.links(i).a;
    d(i) = robot.links(i).d;
    alpha(i) = robot.links(i).alpha;
end

% Inertia
I = zeros(6,n);
for i = 1:n
  I(:,i) = [diag(robot.links(i).I); robot.links(i).I(1,2); ...
  robot.links(i).I(2,3); robot.links(i).I(1,3)];
end

% Mass
M = zeros(1,n);
for i = 1:n
    M(i) = robot.links(i).m;
end

% Centre of Mass
```

```
CoM = ones(4,n);
for i = 1:n
   CoM(1:3,i) = robot.links(i).r;
end

%% Inertia tensor (J)
for i = 1:n
    J(:,:,i) = [0.5*(-I(1,i)+I(2,i)+I(3,i))                    I(4,i)                    I(6,i)  M(i)*CoM(1,i)
                                    I(4,i)  0.5*(I(1,i)-I(2,i)+I(3,i))                    I(5,i)  M(i)*CoM(2,i)
                                    I(6,i)                    I(5,i)  0.5*(I(1,i)+I(2,i)-I(3,i))  M(i)*CoM(3,i)
                             M(i)*CoM(1,i)              M(i)*CoM(2,i)              M(i)*CoM(3,i)           M(i)];
end

%% Constants
Q = zeros(4,4);
Q(1,2) = -1; Q(2,1) = 1;

%% Link transforms
% %% Transforms
T01 = [cos(q(1)),  -cos(alpha(1))*sin(q(1)),  sin(alpha(1))*sin(q(1)), a(1)*cos(q(1));
       sin(q(1)),   cos(alpha(1))*cos(q(1)), -sin(alpha(1))*cos(q(1)), a(1)*sin(q(1));
       0,           sin(alpha(1)),            cos(alpha(1)),           d(1);
       0,           0,                        0,                       1];

T12 = [cos(q(2)),  -cos(alpha(2))*sin(q(2)),  sin(alpha(2))*sin(q(2)), a(2)*cos(q(2));
       sin(q(2)),   cos(alpha(2))*cos(q(2)), -sin(alpha(2))*cos(q(2)), a(2)*sin(q(2));
       0,           sin(alpha(2)),            cos(alpha(2)),           d(2);
       0,           0,                        0,                       1];

T23 = [cos(q(3)),  -cos(alpha(3))*sin(q(3)),  sin(alpha(3))*sin(q(3)), a(3)*cos(q(3));
       sin(q(3)),   cos(alpha(3))*cos(q(3)), -sin(alpha(3))*cos(q(3)), a(3)*sin(q(3));
       0,           sin(alpha(3)),            cos(alpha(3)),           d(3);
       0,           0,                        0,                       1];

T34 = [cos(q(4)),  -cos(alpha(4))*sin(q(4)),  sin(alpha(4))*sin(q(4)), a(4)*cos(q(4));
       sin(q(4)),   cos(alpha(4))*cos(q(4)), -sin(alpha(4))*cos(q(4)), a(4)*sin(q(4));
       0,           sin(alpha(4)),            cos(alpha(4)),           d(4);
       0,           0,                        0,                       1];

T45 = [cos(q(5)),  -cos(alpha(5))*sin(q(5)),  sin(alpha(5))*sin(q(5)), a(5)*cos(q(5));
       sin(q(5)),   cos(alpha(5))*cos(q(5)), -sin(alpha(5))*cos(q(5)), a(5)*sin(q(5));
       0,           sin(alpha(5)),            cos(alpha(5)),           d(5);
       0,           0,                        0,                       1];

T56 = [cos(q(6)),  -cos(alpha(6))*sin(q(6)),  sin(alpha(6))*sin(q(6)), a(6)*cos(q(6));
       sin(q(6)),   cos(alpha(6))*cos(q(6)), -sin(alpha(6))*cos(q(6)), a(6)*sin(q(6));
       0,           sin(alpha(6)),            cos(alpha(6)),           d(6);
       0,           0,                        0,                       1];
if n > 6
T67 = [cos(q(7)),  -cos(alpha(7))*sin(q(7)),  sin(alpha(7))*sin(q(7)), a(7)*cos(q(7));
       sin(q(7)),   cos(alpha(7))*cos(q(7)), -sin(alpha(7))*cos(q(7)), a(7)*sin(q(7));
       0,           sin(alpha(7)),            cos(alpha(7)),           d(7);
       0,           0,                        0,                       1];
end
T00 = eye(4);
T11 = eye(4);
T22 = eye(4);
T33 = eye(4);
T44 = eye(4);
T55 = eye(4);
T66 = eye(4);
T77 = eye(4);


%% U matrix
disp('Starting U matrix calculation... ')
tic
for i = 1:n
    for j = 1:n
        if j <= i
            U(:,:,i,j) = eval(strcat('T',num2str(0),num2str(j-1))) * Q * eval(strcat('T',num2str(j-1),num2str(i)));
        else
            U(:,:,i,j) = zeros(4);
        end
    end
```

```
end
toc
disp('Starting U matrix simplification... ')
tic
U = simplify(U);
toc

%% 3D U matrices
disp('Starting U3 matrix calculation... ')
tic
for i = 1:n
    for j = 1:n
        for k = 1:n
            if (i >= k) && (k >= j)
                U3(:,:,i,j,k) = eval(strcat('T',num2str(0),num2str(j-1))) * Q * ...
                 eval(strcat('T',num2str(j-1),num2str(k-1))) * Q * eval(strcat('T',num2str(k-1),num2str(i)));
            elseif (i >= j) && (j >= k)
                U3(:,:,i,j,k) = eval(strcat('T',num2str(0),num2str(k-1))) * Q *  ...
                eval(strcat('T',num2str(k-1),num2str(j-1))) * Q * eval(strcat('T',num2str(j-1),num2str(i)));
            else
                U3(:,:,i,j,k) = zeros(4);
            end

        end
    end
end
toc
disp('Starting U3 matrix simplification... ')
tic
U3 = simplify(U3);
toc

%% D matrix (inertia)
disp('Starting D matrix calculation... ')
D = sym('D',[n n]);
tic
for i = 1:n
    for k = 1:n
        sum = 0;
        for j = max(i,k):n
            sum = sum + trace(U(:,:,j,k)*J(:,:,j)*U(:,:,j,i)');
        end
        D(i,k) = sum;%simplify(sum);
    end
end
toc
disp('Starting D matrix simplification... ')
tic
% D = simplify(D);
toc

%% h vector (coriolis)
disp('Starting H vector calculation... ')
tic
h = sym('h',[n 1]);
for i = 1:n
    for k = 1:n
        for m_idx = 1:n
            sum = 0;
            for j = max([i, k, m_idx]):n
                sum = sum + trace(U3(:,:,j,k,m_idx) * J(:,:,j) * U(:,:,j,i)');
            end
            h3(i,k,m_idx) = sum;
        end
    end
end

for i = 1:n
    sum_k = 0;
    for k = 1:n
        sum_m = 0;
        for m_idx = 1:n
            sum_m = sum_m + h3(i,k,m_idx) * qd(k) * qd(m_idx);
        end
        sum_k = sum_k + sum_m;
```

```
    end
    h(i,1) = sum_k;
end
toc
disp('Starting H vector simplification... ')
tic
% h = simplify(h);
toc

%% c vector (gravity)
g = max(robot.gravity);
g_vec = [0 -g 0 0];
disp('Starting c vector calculation... ')
tic
c = sym('c',[n 1]);
for i = 1:n
    sum = 0;
    for j = i:n
        sum = sum + (-m(j) * g_vec * U(:,:,j,i) * CoM(:,1));
    end
    c(i,1) = sum;
end
toc
disp('Starting c vector simplification... ')
tic
c = simplify(c);
toc
```

A.2.4 Function for Inverse Dynamics using Recursive Newton-Euler method

```
function [ tau ] = RNE_dyn( robot, q, dq, ddq )
%   RNE Perform recursive Newton-euler calculation
%   Pass robot, joint position, velocity and acceleration, then perform the
%   recursive Newton-Euler calculation through a forward pass (i_1++ ... i_n)
%   then a backward pass (i_n-- ... i_1). Assumes no load at end effector.

%% Initialise variables

n = robot.n;
g = robot.gravity;
z = [0 0 1]';

% omega = zeros(3,n);
% domega = zeros(3,n);
% dv = zeros(3,n);
% f = zeros(3,n+1);
% nn = zeros(3,n+1);

% m = zeros(1,n);
% r = zeros(3,n);
% I = zeros(3,3,n);
% for j = 1:n
%     link(j) = robot.links(j);
%     a(j) = link(j).a;
%     d(j) = link(j).d;
%     alpha(j) = link(j).alpha;
%     m(j) = link(j).m;
%     r(:,j) = link(1).r';
%     I(:,:,j) = link(j).I;
% end
% tau = zeros(n,1);

%% Calculate torque
Rb = t2r(robot.base)';
omega = Rb*zeros(3,1);
domega = Rb*zeros(3,1);
dv = Rb*g;
% omega(:,1,1) = Rb*zeros(3,1);
% domega(:,1,1) = Rb*zeros(3,1);
```

```
% dv(:,1) = Rb*g;

% display(Rb)
% display(g)


%% Forward recursion
Fm = [];
Nm = [];
for j = 1:n
%         disp('forward pass')
    link = robot.links(j);
    R = t2r(link.A(q(j)))'; % transpose last
    p = [link.a; link.d*sin(link.alpha); link.d*cos(link.alpha)];
    r = link.r;

    domega = R * (domega + z*ddq(j) + cross(omega, z*dq(j)));
    omega = R * (omega + z*dq(j));
    dv = cross(domega, p) + cross(omega,cross(omega, p)) + R*dv;
    vhat = cross(domega,r') + cross(omega,cross(omega,r')) + dv;
    fu = link.m * vhat;
    nu = link.I*domega + cross(omega, link.I*omega);
    Fm = [Fm fu];
    Nm = [Nm nu];
end

%% Backward recursion
fext = zeros(6, 1); % external forces acting on the arm (optional)
f = fext(1:3);      % force/moments on end of arm
nn = fext(4:6);
for j = n:-1:1
%         disp('backward pass')
    link = robot.links(j);
    p = [link.a; link.d*sin(link.alpha); link.d*cos(link.alpha)];
    r = link.r;
    if j == n       % If last link, R is identity
        R = eye(3);
    else
        R = t2r(robot.links(j+1).A(q(j+1)));
    end

    nn = R*(nn + cross(R'*p, f)) + cross(p+r',Fm(:,j)) + Nm(:,j);
    f = R*f + Fm(:,j);

    R = t2r(link.A(q(j)));

    tau(j) = nn'*(R'*z); %+ link.G^2 * link.Jm*ddq(j) - link.friction(dq(j));
end

end%Function end
```

Bibliography

[1] L-X Wang. Stable adaptive fuzzy controllers with application to inverted pendulum tracking. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 26(5):677–691, 1996.

[2] Mark Doms, Timothy Dunne, and Mark J Roberts. The role of technology use in the survival and growth of manufacturing plants. *International Journal of Industrial Organization*, 13(4):523–542, 1995.

[3] Vicente Parra-Vega, Suguru Arimoto, Yun-Hui Liu, Gerd Hirzinger, and Prasad Akella. Dynamic sliding pid control for tracking of robot manipulators: theory and experiments. *Robotics and Automation, IEEE Transactions on*, 19(6):967–976, 2003.

[4] Michael A Peshkin, J Edward Colgate, Wit Wannasuphoprasit, Carl A Moore, R Brent Gillespie, and Prasad Akella. Cobot architecture. *Robotics and Automation, IEEE Transactions on*, 17(4):377–390, 2001.

[5] Olivier Lambercy, Ludovic Dovat, Roger Gassert, Etienne Burdet, Chee Leong Teo, and Theodore Milner. A haptic knob for rehabilitation of hand function. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 15(3):356–366, 2007.

[6] Jeffrey Scott Pettinato and Harry E Stephanou. Manipulability and stability of a tentacle based robot manipulator. In *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pages 458–463. IEEE, 1989.

[7] Giuseppe Oriolo and Yoshihiko Nakamura. Control of mechanical systems with second-order nonholonomic constraints: Underactuated manipulators. In *Decision and Control, 1991., Proceedings of the 30th IEEE Conference on*, pages 2398–2403. IEEE, 1991.

[8] Neville Hogan. Impedance control: An approach to manipulation. In *American Control Conference, 1984*, pages 304–313. IEEE, 1984.

[9] Tony J Prescott, Tracy Epton, Vanessa Evers, Kevin McKee, Mark Hawley, Thomas Webb, David Benyon, Sebastian Conran, Roger Strand, Madeleine de Cock Buning, et al. Robot companions for citizens: roadmapping the potential for future robots in empowering older people. 2012.

[10] Sujeong Kim, Stephen J Guy, Wenxi Liu, David Wilkie, Rynson WH Lau, Ming C Lin, and Dinesh Manocha. Predicting pedestrian trajectories for robot navigation. In *Proc. of the workshop Crossing the Reality Gap: Control, Human Interaction and Cloud Technology for Multi-and Many-Robot Systems*. Citeseer, 2014.

[11] Kostas J Kyriakopoulos and George N Saridis. An integrated collision prediction and avoidance scheme for mobile robots in non-stationary environments. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 194–199. IEEE, 1992.

[12] Jaywant P Kolhe, Md Shaheed, TS Chandar, and SE Talole. Robust control of robot manipulators based on uncertainty and disturbance estimation. *International Journal of Robust and Nonlinear Control*, 23(1):104–122, 2013.

[13] Zi-Jiang Yang, Youichirou Fukushima, and Pan Qin. Decentralized adaptive robust control of robot manipulators using disturbance observers. *IEEE Transactions on Control Systems Technology*, 20(5):1357–1365, 2012.

[14] Farzin Piltan, MohammadHossain Yarmahmoudi, Mina Mirzaie, Sara Emamzadeh, and Zahra Hivand. Design novel fuzzy robust feedback linearization control with application to robot manipulator. *International Journal of Intelligent Systems and Applications*, 5(5):1, 2013.

[15] Emre Sariyildiz and Kouhei Ohnishi. Stability and robustness of disturbance-observer-based motion control systems. *IEEE Transactions on Industrial Electronics*, 62(1):414–422, 2015.

[16] C. Yang, G. Ganesh, S. Haddadin, S. Parusel, A. Albu-Schäeffer, and E. Burdet. Human-Like Adaptation of Force and Impedance in Stable and Unstable Interactions. *IEEE Transactions on Robotics*, 27(5):918–930, 2011.

[17] David W Franklin, Rieko Osu, Etienne Burdet, Mitsuo Kawato, and Theodore E Milner. Adaptation to stable and unstable dynamics achieved by combined impedance control and inverse dynamics model. *Journal of neurophysiology*, 90 (5):3270–3282, 2003.

[18] Keng Peng Tee, David W Franklin, Mitsuo Kawato, Theodore E Milner, and Etienne Burdet. Concurrent adaptation of force and impedance in the redundant muscle system. *Biological cybernetics*, 102(1):31–44, 2010.

[19] David W Franklin, Etienne Burdet, Keng Peng Tee, Rieko Osu, Chee-Meng Chew, Theodore E Milner, and Mitsuo Kawato. Cns learns stable, accurate, and efficient movements using a simple algorithm. *The Journal of Neuroscience*, 28(44):11165–11173, 2008.

[20] Keng Peng Tee, Etienne Burdet, Chee-Meng Chew, and Theodore E Milner. A model of force and impedance in human arm movements. *Biological cybernetics*, 90(5):368–375, 2004.

[21] Theodore E Milner and David W Franklin. Impedance control and internal model use during the initial stage of adaptation to novel dynamics in humans. *The Journal of physiology*, 567(2):651–664, 2005.

[22] Etienne Burdet, Rieko Osu, David W Franklin, Theodore E Milner, and Mitsuo Kawato. The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature*, 414(6862):446–449, 2001.

[23] Satoru Sakai and Masakazu Sato. Visual systems & control on polynomial space and its application to sloshing problems. *IEEE Transactions on Control Systems Technology*, 22(6):2176–2187, 2014.

[24] Nobuyuki Kobayashi, Tomohiro Sato, and Ayako Torisaka. Passive control of liquid sloshing in floating roof tank with multi dynamic absorber. In *ASME 2013 Pressure Vessels and Piping Conference*, pages V008T08A031–V008T08A031. American Society of Mechanical Engineers, 2013.

[25] Jerome Wheelook. Centrifugal governor, July 3 1860. US Patent 29,025.

[26] J Clerk Maxwell. On governors. *Proceedings of the Royal Society of London*, 16: 270–283, 1867.

[27] Rudolf Kalman. On the general theory of control systems. *IRE Transactions on Automatic Control*, 4(3):110–110, 1959.

[28] CS George Lee. Robot arm kinematics, dynamics, and control. *Computer*, 15(12): 62–80, 1982.

[29] Daniel E Whitney. Resolved motion rate control of manipulators and human prostheses. *IEEE Transactions on man-machine systems*, 1969.

[30] RH Middleton and GC Goodwin. Adaptive computed torque control for rigid link manipulations. *Systems & Control Letters*, 10(1):9–16, 1988.

[31] Michael Edwin Kahn and B Roth. The near-minimum-time control of open-loop articulated kinematic chains. *Journal of Dynamic Systems, Measurement, and Control*, 93(3):164–172, 1971.

[32] Steven Dubowsky and DT DesForges. The application of model-referenced adaptive control to robotic manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 101(3):193–200, 1979.

[33] Roberto Horowitz and Masayoshi Tomizuka. An adaptive control scheme for mechanical manipulators—compensation of nonlinearity and decoupling control. *Journal of dynamic systems, measurement, and control*, 108(2):127–135, 1986.

[34] Samuel R Buss. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation*, 17, 2004.

[35] Giacomo Marani, Jinhyun Kim, Junku Yuh, and Wan Kyun Chung. A real-time approach for singularity avoidance in resolved motion rate control of robotic manipulators. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1973–1978. IEEE, 2002.

[36] Clément Gosselin and Louis-Thomas Schreiber. Kinematically redundant spatial parallel mechanisms for singularity avoidance and large orientational workspace. *IEEE Transactions on Robotics*, 32(2):286–300, 2016.

[37] Yaghoub Azizi and Nima Azhdar Zadeh. An adaptive control algorithm to improve singularity avoidance in 7-dof redundant manipulators. *Majlesi Journal of Mechatronic Systems*, 3(1), 2014.

[38] Abhishek Agarwal, Chaman Nasa, and Sandipan Bandyopadhyay. Dynamic singularity avoidance for parallel manipulators using a task-priority based control scheme. *Mechanism and Machine Theory*, 96:107–126, 2016.

[39] Peiyao Shen, Yongchun Fang, and Xuebo Zhang. Trajectory planning of omnidirectional mobile robots with active casters: Multi-motor coordination and singularity avoidance. In *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2015 IEEE International Conference on*, pages 151–156. IEEE, 2015.

[40] BR Markiewicz. Analysis of the computed torque drive method and comparison with conventional position servo for a computer-controlled manipulator. 1973.

[41] Chae H An, Christopher G Atkeson, John D Griffiths, and John M Hollerbach. Experimental evaluation of feedforward and computed torque control. *Robotics and Automation, IEEE Transactions on*, 5(3):368–373, 1989.

[42] Zuoshi Song, Jianqiang Yi, Dongbin Zhao, and Xinchun Li. A computed torque controller for uncertain robotic manipulator systems: Fuzzy approach. *Fuzzy Sets and Systems*, 154(2):208–226, 2005.

[43] Arman Jahed, Farzin Piltan, Hossein Rezaie, and Bamdad Boroomand. Design computed torque controller with parallel fuzzy inference system compensator to control of robot manipulator. *International Journal of Information Engineering & Electronic Business*, 5(3), 2013.

[44] Yuan Chen, Guangying Ma, Shuxia Lin, and Jun Gao. Adaptive fuzzy computed-torque control for robot manipulator with uncertain dynamics. *International Journal of Advanced Robotic Systems*, 9, 2012.

[45] Miguel A Llama, Rafael Kelly, and Victor Santibańez. Stable computed-torque control of robot manipulators via fuzzy self-tuning. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 30(1):143–150, 2000.

[46] Zhiyong Yang, Jiang Wu, and Jiangping Mei. Motor-mechanism dynamic model based neural network optimized computed torque control of a high speed parallel manipulator. *Mechatronics*, 17(7):381–390, 2007.

[47] W Thomas Miller III, Robert P Hewes, Filson H Glanz, and L Gordon Kraft III. Real-time dynamic control of an industrial manipulator using a neural network-based learning controller. *Robotics and Automation, IEEE Transactions on*, 6(1): 1–9, 1990.

[48] Young H Kim and Frank L Lewis. Optimal design of cmac neural-network controller for robot manipulators. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 30(1):22–31, 2000.

[49] John G Ziegler and Nathaniel B Nichols. Optimum settings for automatic controllers. *trans. ASME*, 64(11), 1942.

[50] KJ Åström and T Hägglund. Revisiting the ziegler–nichols step response method for pid control. *Journal of process control*, 14(6):635–650, 2004.

[51] Xiao-Gang Duan, Hua Deng, and Han-Xiong Li. A saturation-based tuning method for fuzzy pid controller. *Industrial Electronics, IEEE Transactions on*, 60(11):5177–5185, 2013.

[52] Jose Luis Meza, Víctor Santibáñez, Rogelio Soto, and Miguel A Llama. Fuzzy self-tuning pid semiglobal regulator for robot manipulators. *Industrial Electronics, IEEE Transactions on*, 59(6):2709–2717, 2012.

[53] Nurbaiti Wahid and Nurhaffizah Hassan. Self-tuning fuzzy pid controller design for aircraft pitch control. In *Intelligent Systems, Modelling and Simulation (ISMS), 2012 Third International Conference on*, pages 19–24. IEEE, 2012.

[54] PM Meshram and Rohit G Kanojiya. Tuning of pid controller using ziegler-nichols method for speed control of dc motor. In *Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on*, pages 117–122. IEEE, 2012.

[55] Tamar Flash and Neville Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *The journal of Neuroscience*, 5(7): 1688–1703, 1985.

[56] P Viviani and C Terzuolo. Trajectory determines movement dynamics. *Neuroscience*, 7(2):431–437, 1982.

[57] W Abend, E Bizzi, and P Morasso. Human arm trajectory formation. *Brain: a journal of neurology*, 105(Pt 2):331–348, 1982.

[58] Richard A Schmidt. 8 on the theoretical status of time in motor program representations. *Advances in Psychology*, 1:145–166, 1980.

[59] Daniel Bullock and Stephen Grossberg. Neural dynamics of planned arm movements: emergent invariants and speed-accuracy properties during trajectory formation. *Psychological review*, 95(1):49, 1988.

[60] Neville Hogan. An organizing principle for a class of voluntary movements. *The Journal of Neuroscience*, 4(11):2745–2754, 1984.

[61] Daniel Bullock, Raoul M Bongers, Marnix Lankhorst, and Peter J Beek. A vector-integration-to-endpoint model for performance of viapoint movements. *Neural Networks*, 12(1):1–29, 1999.

[62] Paolo Gaudiano and Stephen Grossberg. Adaptive vector integration to endpoint: Self-organizing neural circuits for control of planned movement trajectories. *Human Movement Science*, 11(1):141–155, 1992.

[63] Daniel Bullock, Stephen Grossberg, and Christian Mannes. A neural network model for cursive script production. *Biological Cybernetics*, 70(1):15–28, 1993.

[64] Daniel Bullock. Adaptive neural models of queuing and timing in fluent action. *Trends in cognitive sciences*, 8(9):426–433, 2004.

[65] Micha Hersch and Aude G Billard. A biologically-inspired controller for reaching movements. In *Biomedical Robotics and Biomechatronics, 2006. BioRob 2006. The First IEEE/RAS-EMBS International Conference on*, pages 1067–1072. IEEE, 2006.

[66] JOHNM Hollerbach and Ki Suh. Redundancy resolution of manipulators through torque optimization. *Robotics and Automation, IEEE Journal of*, 3(4):308–316, 1987.

[67] Jacques Paillard. Motor and representational framing of space. *Brain and space*, pages 163–182, 1991.

[68] JA Kelso. *Dynamic Patterns: The Self Organization of Brain and Behaviour*. The MIT Press, 1995.

[69] Chenguang Yang, Gowrishankar Ganesh, Sami Haddadin, Sven Parusel, Alin Albu-Schaeffer, and Etienne Burdet. Human-like adaptation of force and impedance in stable and unstable interactions. *Robotics, IEEE Transactions on*, 27(5):918–930, 2011.

[70] Gowrishankar Ganesh, A Albu-Schaffer, Masahiko Haruno, Mitsuo Kawato, and Etienne Burdet. Biomimetic motor behavior for simultaneous adaptation of force, impedance and trajectory in interaction tasks. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2705–2711. IEEE, 2010.

[71] Gowrishankar Ganesh, Nathanael Jarrassé, Sami Haddadin, Alin Albu-Schaeffer, and Etienne Burdet. A versatile biomimetic controller for contact tooling and haptic exploration. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3329–3334. IEEE, 2012.

[72] Chenguang Yang, Zhijun Li, and Etienne Burdet. Human like learning algorithm for simultaneous force control and haptic identification. In *International Conference on Intelligent Robots and Systems, 2013*. IEEE, 2013.

[73] Yusuke Maeda, Takayuki Hara, and Tamio Arai. Human-robot cooperative manipulation with motion estimation. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 4, pages 2240–2245. IEEE, 2001.

[74] Vincent Duchaine and Clement M Gosselin. General model of human-robot cooperation using a novel velocity based variable impedance control. In *EuroHaptics Conference, 2007 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2007. Second Joint*, pages 446–451. IEEE, 2007.

[75] Vincent Duchaine and Clement M Gosselin. Investigation of human-robot interaction stability using lyapunov theory. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2189–2194. IEEE, 2008.

[76] F Lacquaniti, F Licata, and JF Soechting. The mechanical behavior of the human forearm in response to transient perturbations. *Biological Cybernetics*, 44(1):35–46, 1982.

[77] Robert E Kearney and Ian W Hunter. System identification of human joint dynamics. *Critical reviews in biomedical engineering*, 18(1):55–87, 1989.

[78] Aleksandr Mikhailovich Lyapunov. The general problem of the stability of motion. *International Journal of Control*, 55(3):531–534, 1992.

[79] Dongheui Lee, Christian Ott, and Yoshihiko Nakamura. Mimetic communication model with compliant physical contact in human-humanoid interaction. *The International Journal of Robotics Research*, 29(13):1684–1704, 2010.

[80] Zheng Wang, Angelika Peer, and Martin Buss. An hmm approach to realistic haptic human-robot interaction. In *EuroHaptics conference, 2009 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2009. Third Joint*, pages 374–379. IEEE, 2009.

[81] Yanan Li, Shuzhi Sam Ge, and Chenguang Yang. Learning impedance control for physical robot–environment interaction. *International Journal of Control*, 85(2): 182–193, 2012.

[82] Yanan Li, Chenguang Yang, and Shuzhi Sam Ge. Learning compliance control of robot manipulators in contact with the unknown environment. In *Automation Science and Engineering (CASE), 2010 IEEE Conference on*, pages 644–649. IEEE, 2010.

[83] Rodney A Brooks. A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14–23, 1986.

[84] Rodney A Brooks. Elephants don't play chess. *Robotics and autonomous systems*, 6(1):3–15, 1990.

[85] Allon Guez and Ziauddin Ahmad. Solution to the inverse kinematics problem in robotics by neural networks. In *Neural Networks, 1988., IEEE International Conference on*, pages 617–624. IEEE, 1988.

[86] Mitsuo Kawato, Yoji Uno, Michiaki Isobe, and Ryoji Suzuki. Hierarchical neural network model for voluntary movement with application to robotics. *Control Systems Magazine, IEEE*, 8(2):8–15, 1988.

[87] J Barhen, WB Dress, and CC Jorgensen. Applications of concurrent neuromorphic algorithms for autonomous robots. In *Neural Computers*, pages 321–333. Springer, 1989.

[88] Rolf Pfeifer, Max Lungarella, and Fumiya Iida. Self-organization, embodiment, and biologically inspired robotics. *science*, 318(5853):1088–1093, 2007.

[89] Auke Jan Ijspeert, Alessandro Crespi, Dimitri Ryczko, and Jean-Marie Cabelguen. From swimming to walking with a salamander robot driven by a spinal cord model. *science*, 315(5817):1416–1420, 2007.

[90] Wei Wang, Yingying Wang, Jinghao Qi, Houxiang Zhang, and Jianwei Zhang. The cpg control algorithm for a climbing worm robot. In *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*, pages 675–679. IEEE, 2008.

[91] Charles C Kemp, Aaron Edsinger, and Eduardo Torres-Jara. Challenges for robot manipulation in human environments. *IEEE Robotics and Automation Magazine*, 14(1):20, 2007.

[92] Stefano Michieletto, Nicola Chessa, and Emanuele Menegatti. Learning how to approach industrial robot tasks from natural demonstrations. In *Advanced Robotics and its Social Impacts (ARSO), 2013 IEEE Workshop on*, pages 255–260. IEEE, 2013.

[93] Scott Niekum, Sachin Chitta, Andrew G Barto, Bhaskara Marthi, and Sarah Osentoski. Incremental semantically grounded learning from demonstration. In *Robotics: Science and Systems*, volume 9, 2013.

[94] Shu Huang, Erwin Aertbeliën, Herman Bruyninckx, and Hendrik Van Brussel. Behavior-based task learning by demonstration on mobile manipulation. *International Journal of Automation and Smart Technology*, 3(1):19–28, 2013.

[95] S Chiaverini and A Meddahi. A null-space based behavioural control approach to coordinated motion of a humanoid robot. In *Information and Automation, 2015 IEEE International Conference on*, pages 20–25. IEEE, 2015.

[96] C Stefanini, S Orofino, L Manfredi, S Mintchev, S Marrazza, T Assaf, L Capantini, E Sinibaldi, S Grillner, P Wallén, et al. A novel autonomous, bioinspired swimming robot developed by neuroscientists and bioengineers. *Bioinspiration & biomimetics*, 7(2):025001, 2012.

[97] Kyoo Jae Shin. Development of autonomous bio-mimetic ornamental aquarium fish robotic. *KIPS Transactions on Software and Data Engineering*, 4(5):219–224, 2015.

[98] Hyung-Jung Kim, Sung-Hyuk Song, and Sung-Hoon Ahn. A turtle-like swimming robot using a smart soft composite (ssc) structure. *Smart Materials and Structures*, 22(1):014007, 2012.

[99] Jae Hoon Lee, Shingo Okamoto, Hisashi Koike, and Keiya Tani. Development and motion control of a biped walking robot based on passive walking theory. *Artificial Life and Robotics*, 19(1):68–75, 2014.

[100] Ying Cao, Soichiro Suzuki, and Yohei Hoshino. Turn control of a three-dimensional quasi-passive walking robot by utilizing a mechanical oscillator. *Engineering*, 2014, 2014.

[101] Hiroki Iba, Shingo Okamoto, and Jae Hoon Lee. Computer simulation of semi-passive walking robot with four legs and verification of it's validity using developed experimental robot. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 2, 2016.

[102] Yan Huang, Bram Vanderborght, Ronald Van Ham, Qining Wang, Michael Van Damme, Guangming Xie, and Dirk Lefeber. Step length and velocity control of a dynamic bipedal walking robot with adaptable compliant joints. *Mechatronics, IEEE/ASME Transactions on*, 18(2):598–611, 2013.

[103] Philippe Sardain and Guy Bessonnet. Forces acting on a biped robot. center of pressure-zero moment point. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 34(5):630–637, 2004.

[104] Tomos G Williams and Nigel W Hardy. Behavioural modules in force control of robotic manipulators. In *Proceedings of the 13th World Congress: International Federation of Automatic Control, San Francisco, USA, 30th June-5th July 1996*, page 177. Pergamon, 1997.

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

[106] Tsuneo Yoshikawa. Dynamic hybrid position/force control of robot manipulators–description of hand constraints and calculation of joint driving force. *Robotics and Automation, IEEE Journal of*, 3(5):386–392, 1987.

[107] Tsuneo Yoshikawa, Toshiharu Sugie, and Masaki Tanaka. Dynamic hybrid position/force control of robot manipulators-controller design and experiment. *Robotics and Automation, IEEE Journal of*, 4(6):699–705, 1988.

[108] Samad Hayati. Hybrid position/force control of multi-arm cooperating robots. In *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, volume 3, pages 82–89. IEEE, 1986.

[109] Tsuneo Yoshikawa and Xin-Zhi Zheng. Coordinated dynamic hybrid position/force control for multiple robot manipulators handling one constrained object. *The International Journal of Robotics Research*, 12(3):219–230, 1993.

[110] Masaru Uchiyama and Pierre Dauchez. A symmetric hybrid position/force control scheme for the coordination of two robots. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 350–356. IEEE, 1988.

[111] John J Craig and Marc H Raibert. A systematic method of hybrid position/force control of a manipulator. In *Computer Software and Applications Conference, 1979. Proceedings. COMPSAC 79. The IEEE Computer Society's Third International*, pages 446–451. IEEE, 1979.

[112] Di Xiao, Bijoy K Ghosh, Ning Xi, and Tzyh Jong Tarn. Sensor-based hybrid position/force control of a robot manipulator in an uncalibrated environment. *Control Systems Technology, IEEE Transactions on*, 8(4):635–645, 2000.

[113] Thomas Buschmann, Sebastian Lohmeier, and Heinz Ulbrich. Biped walking control based on hybrid position/force control. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 3019–3024. IEEE, 2009.

[114] Naveen Kumar, Vikas Panwar, Nagarajan Sukavanam, Shri Prakash Sharma, and Jin-Hwan Borm. Neural network based hybrid force/position control for robot manipulators. *International Journal of Precision Engineering and Manufacturing*, 12(3):419–426, 2011.

[115] Abílio Azenha. Iterative learning in variable structure position/force hybrid control of manipulators. *Robotica*, 18(02):213–217, 2000.

[116] Ming-Shaung Ju, Chou-Ching K Lin, Dong-Huang Lin, Ing-Shiou Hwang, and Shu-Min Chen. A rehabilitation robot with force-position hybrid fuzzy controller: hybrid fuzzy control of rehabilitation robot. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 13(3):349–358, 2005.

[117] Steven Bellens, Joris De Schutter, and Herman Bruyninckx. A hybrid pose/wrench control framework for quadrotor helicopters. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 2269–2274. IEEE, 2012.

[118] Seth Hutchinson, Gregory D Hager, and Peter I Corke. A tutorial on visual servo control. *Robotics and Automation, IEEE Transactions on*, 12(5):651–670, 1996.

[119] Guoqiang Ye, Weiguang Li, Hao Wan, and Huidong Lou. Novel two-stage hybrid ibvs controller combining cartesian and polar based methods. In *Mechatronics and Automation (ICMA), 2015 IEEE International Conference on*, pages 397–402. IEEE, 2015.

[120] Francois Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In *The confluence of vision and control*, pages 66–78. Springer, 1998.

[121] Chi-Yi Tsai, Ching-Chang Wong, Chia-Jun Yu, Chih-Cheng Liu, and Tsung-Yen Liu. A hybrid switched reactive-based visual servo control of 5-dof robot manipulators for pick-and-place tasks. *Systems Journal, IEEE*, 9(1):119–130, 2015.

[122] AH Abdul Hafez, Enric Cervera, CV Jawahar, and IIIT CVIT. Stable hybrid visual servo control by a weighted combination of image-based and position-based algorithms. *International Journal of Control and Automation*, 6(3):149–164, 2013.

[123] Lotfi A Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[124] Ebrahim H Mamdani. Application of fuzzy algorithms for control of simple dynamic plant. In *Proceedings of the Institution of Electrical Engineers*, volume 121, pages 1585–1588. IET, 1974.

[125] Kazuo Tanaka. *An introduction to fuzzy logic for practical applications*. Springer, 1997.

[126] S-G Kong and Bart Kosko. Comparison of fuzzy and neural truck backer-upper control systems. In *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*, pages 349–358. IEEE, 1990.

[127] Philip R Thrift. Fuzzy logic synthesis with genetic algorithms. In *ICGA*, pages 509–513, 1991.

[128] Daihee Park, Abraham Kandel, and Gideon Langholz. Genetic-based new fuzzy reasoning models with application to fuzzy control. *Systems, Man and Cybernetics, IEEE Transactions on*, 24(1):39–47, 1994.

[129] Francisco Herrera, Manuel Lozano, and Jose L Verdegay. Tuning fuzzy logic controllers by genetic algorithms. *International Journal of Approximate Reasoning*, 12(3):299–315, 1995.

[130] Mikio Maeda, Yasushi Maeda, and Shuta Murakami. Fuzzy drive control of an autonomous mobile robot. *Fuzzy sets and systems*, 39(2):195–204, 1991.

[131] Euntai Kim. Output feedback tracking control of robot manipulators with model uncertainty via adaptive fuzzy logic. *Fuzzy Systems, IEEE Transactions on*, 12 (3):368–378, 2004.

[132] Rong-Jong Wai. Fuzzy sliding-mode control using adaptive tuning technique. *Industrial Electronics, IEEE Transactions on*, 54(1):586–594, 2007.

[133] J Wang, Ahmad B Rad, and PT Chan. Indirect adaptive fuzzy sliding mode control: Part i: fuzzy switching. *Fuzzy sets and Systems*, 122(1):21–30, 2001.

[134] Her-Terng Yau and Chieh-Li Chen. Chattering-free fuzzy sliding-mode control strategy for uncertain chaotic systems. *Chaos, Solitons & Fractals*, 30(3):709–718, 2006.

[135] HF Ho, Yiu-Kwong Wong, and Ahmad B Rad. Adaptive fuzzy sliding mode control with chattering elimination for nonlinear siso systems. *Simulation Modelling Practice and Theory*, 17(7):1199–1210, 2009.

[136] Rong-Jong Wai, Chih-Min Lin, and Chun-Fei Hsu. Adaptive fuzzy sliding-mode control for electrical servo drive. *Fuzzy sets and systems*, 143(2):295–310, 2004.

[137] Farzin Piltan, Mohammad Keshavarz, Ali Badri, and Arash Zargari. Design novel nonlinear controller applied to robotmanipulator: Design new feedback linearization fuzzy controller with minimum rule base tuning method. *International Journal of Robotics and Automation*, 3(1):1–12, 2012.

[138] Woonchul Ham. Adaptive fuzzy sliding mode control of nonlinear system. *Fuzzy Systems, IEEE Transactions on*, 6(2):315–321, 1998.

[139] Shaocheng Tong and Han-Xiong Li. Fuzzy adaptive sliding-mode control for mimo nonlinear systems. *Fuzzy Systems, IEEE Transactions on*, 11(3):354–360, 2003.

[140] M Roopaei and M Zolghadri Jahromi. Chattering-free fuzzy sliding mode control in mimo uncertain systems. *Nonlinear Analysis: Theory, Methods & Applications*, 71(10):4430–4437, 2009.

[141] Vahab Nekoukar and Abbas Erfanian. Adaptive fuzzy terminal sliding mode control for a class of mimo uncertain nonlinear systems. *Fuzzy Sets and Systems*, 179 (1):34–49, 2011.

[142] W-S Lin and C-S Chen. Robust adaptive sliding mode control using fuzzy modelling for a class of uncertain mimo nonlinear systems. *IEE Proceedings-Control Theory and Applications*, 149(3):193–202, 2002.

[143] Fang-Ming Yu, Hung-Yuan Chung, and Shi-Yuan Chen. Fuzzy sliding mode controller design for uncertain time-delayed systems with nonlinear input. *Fuzzy Sets and Systems*, 140(2):359–374, 2003.

[144] Chih-Lyang Hwang, Li-Jui Chang, and Yuan-Sheng Yu. Network-based fuzzy decentralized sliding-mode control for car-like mobile robots. *Industrial Electronics, IEEE Transactions on*, 54(1):574–585, 2007.

[145] Yuanqing Xia, Mengyin Fu, Hongjiu Yang, and Guo-Ping Liu. Robust sliding-mode control for uncertain time-delay systems based on delta operator. *Industrial Electronics, IEEE Transactions on*, 56(9):3646–3655, 2009.

[146] Charles G Burgar, Peter S Lum, Peggy C Shor, and HF Machiel Van der Loos. Development of robots for rehabilitation therapy: the palo alto va/stanford experience. *Journal of rehabilitation research and development*, 37(6):663–674, 2000.

[147] Peter S Lum, Charles G Burgar, Peggy C Shor, Matra Majmundar, and Machiel Van der Loos. Robot-assisted movement training compared with conventional therapy techniques for the rehabilitation of upper-limb motor function after stroke. *Archives of physical medicine and rehabilitation*, 83(7):952–959, 2002.

[148] BT Volpe, HI Krebs, N Hogan, L Edelstein, C Diels, and M Aisen. A novel approach to stroke rehabilitation robot-aided sensorimotor stimulation. *Neurology*, 54(10):1938–1944, 2000.

[149] Robert Riener, Tobias Nef, and Gery Colombo. Robot-aided neurorehabilitation of the upper extremities. *Medical and Biological Engineering and Computing*, 43 (1):2–10, 2005.

[150] Thomas G Sugar and Vijay Kumar. Control of cooperating mobile manipulators. *Robotics and Automation, IEEE Transactions on*, 18(1):94–103, 2002.

[151] Elena Gribovskaya and Aude Billard. Combining dynamical systems control and programming by demonstration for teaching discrete bimanual coordination tasks to a humanoid robot. In *Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on*, pages 33–40. IEEE, 2008.

[152] Aaron Edsinger and Charles C Kemp. Manipulation in human environments. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 102–109. IEEE, 2006.

[153] Aaron Edsinger-Gonzales and Jeff Weber. Domo: a force sensing humanoid robot for manipulation research. In *Humanoid Robots, 2004 4th IEEE/RAS International Conference on*, volume 1, pages 273–291. IEEE, 2004.

[154] Gill A Pratt and Matthew M Williamson. Series elastic actuators. In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 1, pages 399–406. IEEE, 1995.

[155] Myun Joong Hwang, Doo Yong Lee, and Seong Youb Chung. Motion planning of bimanual robot for assembly. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 240–245. IEEE, 2007.

[156] Z-W Luo, Koji Ito, and Masami Ito. Multiple robot manipulators' cooperative compliant manipulation on dynamical environments. In *Intelligent Robots and Systems' 93, IROS'93. Proceedings of the 1993 IEEE/RSJ International Conference on*, volume 3, pages 1927–1934. IEEE, 1993.

[157] iCub home. http://www.icub.org/. Accessed: 2015-10-10.

[158] S. Ivaldi, M. Fumagalli, M. Randazzo, F. Nori, G. Metta, and G. Sandini. Computing robot internal/external wrenches by means of inertial, tactile and f/t sensors: theory and implementation on the icub. In *Proc. of the 11th IEEE-RAS International Conference on Humanoid Robots, Bled, Slovenia*, 2011.

[159] Giorgio Cannata, Marco Maggiali, Giorgio Metta, and Giulio Sandini. An embedded artificial skin for humanoid robots. In *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, pages 434–438. IEEE, 2008.

[160] Ch Ott, Oliver Eiberger, Werner Friedl, Berthold Bäuml, Ulrich Hillenbrand, Ch Borst, Alin Albu-Schaffer, Bernhard Brunner, Heiko Hirschmüller, Simon Kielhöfer, et al. A humanoid two-arm system for dexterous manipulation. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 276–283. IEEE, 2006.

[161] Gerd Hirzinger, Norbert Sporer, A Albu-Schaffer, M Hahnle, R Krenn, A Pascucci, and Markus Schedl. Dlr's torque-controlled light weight robot iii-are we reaching the technological limits now? In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1710–1716. IEEE, 2002.

[162] Stefano Stramigioli. Intrinsically passive control. *Modeling and IPC control of interactive mechanical systems—A coordinate-free approach*, pages 125–145, 2001.

[163] Matthew M Williamson. Series elastic actuators. 1995.

[164] Rethink robotics. http://www.rethinkrobotics.com/, 2015. [Online; accessed 6 June 2015].

[165] C Fitzgerald. Developing baxter. In *Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference on*, pages 1–6. IEEE, 2013.

[166] Zhangfeng Ju, Chenguang Yang, and Hongbin Ma. Kinematics modeling and experimental verification of baxter robot. In *Control Conference (CCC), 2014 33rd Chinese*, pages 8518–8523. IEEE, 2014.

[167] Ashesh Jain, Brian Wojcik, Thorsten Joachims, and Ashutosh Saxena. Learning trajectory preferences for manipulators via iterative improvement. In *Advances in Neural Information Processing Systems*, pages 575–583, 2013.

[168] Florian T Pokorny, Majd Hawasly, and Subramanian Ramamoorthy. Multi-scale topological trajectory classification with persistent homology. In *The 2014 Robotics: Science and Systems Conference, July 12-16, 2014 Berkeley, USA*, 2014.

[169] ReThink Robotics collision avoidance and detection. http://sdk.rethinkrobotics.com/wiki/Collision_Avoidance_and_Detection. Accessed: 2015-11-15.

[170] Michael A Peshkin, J Edward Colgate, Wit Wannasuphoprasit, Carl A Moore, R Brent Gillespie, and Prasad Akella. Cobot architecture. *Robotics and Automation, IEEE Transactions on*, 17(4):377–390, 2001.

[171] Olivier Lambercy, Ludovic Dovat, Roger Gassert, Etienne Burdet, Chee Leong Teo, and Theodore Milner. A haptic knob for rehabilitation of hand function. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 15(3):356–366, 2007.

[172] Neville Hogan. Impedance control: An approach to manipulation. In *American Control Conference, 1984*, pages 304–313. IEEE, 1984.

[173] Long Cheng, Yingzi Lin, Zeng-Guang Hou, Min Tan, Jian Huang, and WJ Zhang. Adaptive tracking control of hybrid machines: a closed-chain five-bar mechanism case. *Mechatronics, IEEE/ASME Transactions on*, 16(6):1155–1163, 2011.

[174] Zhijun Li, Chenguang Yang, Nan Ding, Stjepan Bogdan, and Tong Ge. Robust adaptive motion control for underwater remotely operated vehicles with velocity constraints. *International Journal of Control, Automation and Systems*, 10(2): 421–429, 2012.

[175] Zhijun Li, Chenguang Yang, and Yong Tang. Decentralised adaptive fuzzy control of coordinated multiple mobile manipulators interacting with non-rigid environments. *IET Control Theory & Applications*, 7(3):397–410, 2013.

[176] RAFAEL Kelly, RICARDO Carelli, Mauricio Amestegui, and Romeo Ortega. On adaptive impedance control of robot manipulators. In *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pages 572–577. IEEE, 1989.

[177] Richard Colbaugh, Homayoun Seraji, and Kristin Glass. Direct adaptive impedance control of robot manipulators. *Journal of Robotic Systems*, 10(2):217–248, 1993.

[178] Gowrishankar Ganesh, Nathanael Jarrassé, Sami Haddadin, Alin Albu-Schaeffer, and Etienne Burdet. A versatile biomimetic controller for contact tooling and haptic exploration. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3329–3334. IEEE, 2012.

[179] E. Burdet, R. Osu, D.W. Franklin, T.E. Milner, and M. Kawato. The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature*, 414(6862):446–449, 2001.

[180] David W Franklin, Rieko Osu, Etienne Burdet, Mitsuo Kawato, and Theodore E Milner. Adaptation to stable and unstable dynamics achieved by combined impedance control and inverse dynamics model. *Journal of Neurophysiology*, 90 (5):3270–3282, 2003.

[181] David W Franklin, Etienne Burdet, Keng Peng Tee, Rieko Osu, Chee-Meng Chew, Theodore E Milner, and Mitsuo Kawato. CNS learns stable, accurate, and efficient movements using a simple algorithm. *The Journal of Neuroscience*, 28(44):11165–11173, 2008.

[182] Keng Peng Tee, David W Franklin, Mitsuo Kawato, Theodore E Milner, and Etienne Burdet. Concurrent adaptation of force and impedance in the redundant muscle system. *Biological cybernetics*, 102(1):31–44, 2010.

[183] G. Ganesh, A. Albu-Schaffer, M. Haruno, M. Kawato, and E. Burdet. Biomimetic motor behavior for simultaneous adaptation of force, impedance and trajectory in interaction tasks. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2705–2711. IEEE, 2010.

[184] TC Hsia. Adaptive control of robot manipulators-a review. In *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, volume 3, pages 183–189. IEEE, 1986.

[185] John J Craig, Ping Hsu, and S Shankar Sastry. Adaptive control of mechanical manipulators. *The International Journal of Robotics Research*, 6(2):16–28, 1987.

[186] Jean-Jacques E Slotine and Weiping Li. On the adaptive control of robot manipulators. *The international journal of robotics research*, 6(3):49–59, 1987.

[187] Muhammad Nasiruddin Mahyuddin, Said Ghani Khan, and Guido Herrmann. A novel robust adaptive control algorithm with finite-time online parameter estimation of a humanoid robot arm. *Robotics and Autonomous Systems*, 62(3):294–305, 2014.

[188] Julio Gonzalo Garcia, Anders Robertsson, Juan Gómez Ortega, and Rolf Johansson. Sensor fusion for compliant robot motion control. *Robotics, IEEE Transactions on*, 24(2):430–441, 2008.

[189] Gürsel Alici and Bijan Shirinzadeh. Enhanced stiffness modeling, identification and characterization for robot manipulators. *Robotics, IEEE Transactions on*, 21 (4):554–564, 2005.

[190] Alessandro De Luca, Alin Albu-Schaffer, Sami Haddadin, and Gerd Hirzinger. Collision detection and safe reaction with the dlr-iii lightweight manipulator arm. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 1623–1630. IEEE, 2006.

[191] Antonio Bicchi, Stefano Lodi Rizzini, and Giovanni Tonietti. Compliant design for intrinsic safety: General issues and preliminary design. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 4, pages 1864–1869. IEEE, 2001.

[192] Said G Khan, Guido Herrmann, Alexander Lenz, Mubarak Al Grafi, Tony Pipe, and Chris Melhuish. Compliance control and human–robot interaction: Part ii—experimental examples. *International Journal of Humanoid Robotics*, 11(03), 2014.

[193] Wei Wang, Robert NK Loh, and Edward Y Gu. Passive compliance versus active compliance in robot-based automated assembly systems. *Industrial Robot: An International Journal*, 25(1):48–57, 1998.

[194] Michael Zinn, Bernard Roth, Oussama Khatib, and J Kenneth Salisbury. A new actuation approach for human friendly robot design. *The international journal of robotics research*, 23(4-5):379–398, 2004.

[195] Mark Uebel, Ioannis Minis, and Kevin Cleary. Improved computed torque control for industrial robots. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 528–533. IEEE, 1992.

[196] David W Clarke. Application of generalized predictive control to industrial processes. *Control Systems Magazine, IEEE*, 8(2):49–55, 1988.

[197] Ph Poignet and M Gautier. Nonlinear model predictive control of a robot manipulator. In *Advanced Motion Control, 2000. Proceedings. 6th International Workshop on*, pages 401–406. IEEE, 2000.

[198] Katsuhisa Furuta. Sliding mode control of a discrete system. *Systems & Control Letters*, 14(2):145–152, 1990.

[199] Man Zhihong, AP Paplinski, and HR Wu. A robust mimo terminal sliding mode control scheme for rigid robotic manipulators. *Automatic Control, IEEE Transactions on*, 39(12):2464–2469, 1994.

[200] Jurgen Guldner, Vadim Utkin, et al. Sliding mode control for gradient tracking and robot navigation using artificial potential fields. *Robotics and Automation, IEEE Transactions on*, 11(2):247–254, 1995.

[201] Yong Feng, Xinghuo Yu, and Zhihong Man. Non-singular terminal sliding mode control of rigid manipulators. *Automatica*, 38(12):2159–2167, 2002.

[202] Roy Featherstone and David Orin. Robot dynamics: equations and algorithms. In *ICRA*, pages 826–834, 2000.

[203] KS Fu and CSG RC Gonzalez. Robotics control, sensing, vision, and. 1987.

[204] Antal K Bejczy. Robot arm dynamics and control. *Jet Propulsion Laboratory Technical Memo*, pages 33–669, 1974.

[205] AK Bejczy and RP Paul. Simplified robot arm dynamics for control. In *Decision and Control including the Symposium on Adaptive Processes, 1981 20th IEEE Conference on*, pages 261–262. IEEE, 1981.

[206] SaFid M Megahed. *Principles of robot modelling and simulation*. John Wiley & Sons, Inc., 1993.

[207] CSG Lee, BH Lee, and R Nigam. Development of the generalized d'alembert equations of motion for mechanical manipulators. In *Decision and Control, 1983. The 22nd IEEE Conference on*, pages 1205–1210. IEEE, 1983.

[208] John M Hollerbach. A recursive lagrangian formulation of maniputator dynamics and a comparative study of dynamics formulation complexity. *Systems, Man and Cybernetics, IEEE Transactions on*, 10(11):730–736, 1980.

[209] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2009.

[210] Philip Mckerrow. *Introduction to robotics*. Addison-Wesley Longman Publishing Co., Inc., 1991.

[211] Zhangfeng Ju, Chenguang Yang, and Hongbin Ma. Kinematic modeling and experimental verification of Baxter robot. In *Proceedings of the 33rd Chinese Control Conference Nanjing, China, 28-30 Jul, 2014*, pages 8518–8523. CCC, 2014.

[212] J. E. Slotine and W. Li. *Applied Nonlinear Control*. Englewood Cliff, NJ: Prentice-Hall, Inc., 1991.

[213] E. J. Perreault, R. F. Kirsch, and P.E. Crago. Multijoint dynamics and postural stability of the human arm. *Experimental Brain Research*, 157(4):507–517, 2004.

[214] G. Ganesh, M. Haruno, and E. Burdet. Transitions between reciprical activation and co-contraction during posture control. *Poster at Neural Control of Movement*, NCM'08, 2008.

[215] Chae H An, Christopher G Atkeson, and John M Hollerbach. Estimation of inertial parameters of rigid body links of manipulators. In *Decision and Control, 1985 24th IEEE Conference on*, pages 990–995. IEEE, 1985.

[216] G. Ganesh, M. Haruno, M. Kawato, and E. Burdet. Motor memory and local minimization of error and effort, not global optimization, determine motor behavior. *Journal of Neurophysiology*, 104(1):382–390, 2010.

[217] Wesley E Woodson, Barry Tillman, and Peggy Tillman. *Human factors design handbook: information and guidelines for the design of systems, facilities, equipment, and products for human use*. McGraw-Hill, 1992.

[218] Conor Fitzgerald. Developing baxter. In *Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference on*, pages 1–6. IEEE, 2013.

[219] Long Cheng, Zeng-Guang Hou, and Min Tan. Adaptive neural network tracking control for manipulators with uncertain kinematics, dynamics and actuator model. *Automatica*, 45(10):2312–2318, 2009.

[220] Long Cheng, Zeng-Guang Hou, Min Tan, and Wen-Jun Zhang. Tracking control of a closed-chain five-bar robot with two degrees of freedom by integration of an approximation-based approach and mechanical design. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 42(5):1470–1479, 2012.

[221] Lofti A Zadeh. Fuzzy logic. *Computer*, 21(4):83–93, 1988.

[222] Ebrahim H Mamdani and Sedrak Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International journal of man-machine studies*, 7(1): 1–13, 1975.

[223] Hongyi Li, Jinyong Yu, Chris Hilton, and Honghai Liu. Adaptive sliding-mode control for nonlinear active suspension vehicle systems using t–s fuzzy approach. *Industrial Electronics, IEEE Transactions on*, 60(8):3328–3338, 2013.

[224] Jiacheng Tan, Zhaojie Ju, Steve Hand, and Honghai Liu. Robot navigation and manipulation control based-on fuzzy spatial relation analysis. *International Journal of Fuzzy Systems*, 13(4):292–301, 2011.

[225] Michio Sugeno. An introductory survey of fuzzy control. *Information sciences*, 36 (1):59–83, 1985.

[226] Bernadette Bouchon-Meunier, Mariagrazia Dotoli, and Bruno Maione. On the choice of membership functions in a mamdani-type fuzzy controller. In *Proceedings of the First Online Workshop on Soft Computing, Nagoya, Japan*. Citeseer, 1996.

[227] Timothy J Ross. *Fuzzy logic with engineering applications*. John Wiley & Sons, 2009.

[228] Tomohiro Takagi and Michio Sugeno. Fuzzy identification of systems and its applications to modeling and control. *Systems, Man and Cybernetics, IEEE Transactions on*, 15(1):116–132, 1985.

[229] Joseph P LaSalle. Stability of nonautonomous systems. *Nonlinear Analysis: Theory, Methods & Applications*, 1(1):83–90, 1976.

[230] Yoshiaki Sakagami, Ryujin Watanabe, Chiaki Aoyama, Shinichi Matsunaga, Nobuo Higaki, and Kikuo Fujimura. The intelligent asimo: System overview and integration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2478–2483. IEEE, 2002.

[231] Ill-Woo Park, Jung-Yup Kim, Jungho Lee, and Jun-Ho Oh. Mechanical design of humanoid robot platform khr-3 (kaist humanoid robot 3: Hubo). In *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*, pages 321–326. IEEE, 2005.

[232] H Kazerooni, JJ Bausch, and BM Kramer. An approach to automated deburring by robot manipulators. *Journal of dynamic systems, measurement, and control*, 108(4):354–359, 1986.

[233] Jochen Heinzmann and Alexander Zelinsky. Quantitative safety guarantees for physical human-robot interaction. *The International Journal of Robotics Research*, 22(7-8):479–504, 2003.

[234] Antonio Bicchi, Michael A Peshkin, and J Edward Colgate. Safety for physical human–robot interaction. In *Springer handbook of robotics*, pages 1335–1348. Springer, 2008.

[235] Alex Smith, Chenguang Yang, Hongbin Ma, Phil Culverhouse, Angelo Cangelosi, and Etienne Burdet. Biomimetic joint/task space hybrid adaptive control for bimanual robotic manipulation. In *Control & Automation (ICCA), 11th IEEE International Conference on*, pages 1013–1018. IEEE, 2014.

[236] Herbert G Tanner, Kostas J Kyriakopoulos, and NJ Krikelis. Modeling of multiple mobile manipulators handling a common deformable object. *Journal of Robotic Systems*, 15(11):599–623, 1998.

[237] ZhiDong Wang, Eiji Nakano, and Takayuki Takahashi. Solving function distribution and behavior design problem for cooperative object handling by multiple mobile robots. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 33(5):537–549, 2003.

[238] Kazuhiro Kosuge and Tomohiro Oosumi. Decentralized control of multiple robots handling an object. In *Intelligent Robots and Systems' 96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, volume 1, pages 318–323. IEEE, 1996.

[239] Mark R Cutkosky, John M Jourdain, and Paul K Wright. Skin materials for robotic fingers. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 1649–1654. IEEE, 1987.

[240] Lorenzo Sciavicco and Bruno Siciliano. *Modeling and control of robot manipulators*, volume 8. McGraw-Hill New York, 1996.

[241] Brian Armstrong, Oussama Khatib, and Joel Burdick. The explicit dynamic model and inertial parameters of the puma 560 arm. In *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, volume 3, pages 510–518. IEEE, 1986.