Robotic Manipulation of Deformable Objects Using Robust Output Regulation

ROBOTIC MANIPULATION OF DEFORMABLE OBJECTS USING ROBUST OUTPUT REGULATION

BY

RICHARD FANSON, B.A.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2010) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Robotic Manipulation of Deformable Objects Using Robust Output Regulation
AUTHOR:	Richard Fanson B.A.Sc., (Mechatronics Engineering) University of Waterloo, Waterloo, Ontario, Canada
SUPERVISOR:	Dr. Alexandru Patriciu
NUMBER OF PAGES:	xviii, 123

I would like to dedicate this thesis to my loving family and to moja bebica.

Abstract

Traditionally, robots are used to handle rigid objects, such as in automotive and manufacturing industries. Emerging robotic fields like food industry and medical robotics has inspired research into robots capable of accurately manipulating deformable objects.

This thesis focuses on a task known as the indirect simultaneous positioning problem. The task requires accurate deformation control of a nonrigid object. The desired configuration is prescribed by control points defined within the object body. This deformation is achieved by applying forces to to manipulation points located elsewhere on the body.

This thesis approaches the problem using a linear robust output regulation framework. This framework requires the following steps:

1. construct a second-order dynamical model of the object
2. linearize about the equilibrium point
3. assess the controllability, observability, and solution to the regulator equations
4. compute the controller gains that solve the regulation problem

This control law approach is tested in both simulations using a nonlinear RKPM

model of a deformable object, and in experiments using a robot manipulator to apply the desired forces to the manipulation points of a physical, elastic, deformable object. This thesis proves the validity of this approach using homogeneous and non-homogeneous planar objects, as well as multiple control point and manipulation point configurations.

Acknowledgements

I would like to thank my supervisor, Dr Alexandru Patriciu, for all of his support, guidance and mentorship throughout my studies. Additional gratitude goes towards all the professors of the department who have made my time here a great learning experience. Special thanks to Dr Noseworthy for begin a great TA supervisor and hockey opponent, and to Dr Reilly, Dr Sirouspour, and Dr Davidson for being exceptional graduate instructors.

I would also like to thank Cheryl Gies and Alexa Huang for all of their friendly assistance during these two years, making me feel welcome in the department and ensuring a great experience here at McMaster.

Notation and abbreviations

Abbreviations:

FEM — Finite Element Methods

RKPM — Reproducing Kernel Particle Methods

SPH — Smooth Particle Hydrodynamics

PIC — Particle in Cell

DEM — Diffuse Element Methods

EFG — Element Free Galerkin

PBH — Popov, Belevitch and Hautus

MIMO — Multiple Input, Multiple Output

DOF — Degree of Freedom

DLL — Dynamic Link Library

GPU — Graphics Processing Unit

SDK — Software Development Kit

PI — Proportional, Integral

PD — Proportional, Derivative

PID — Proportional, Integral, Derivative

MSD — Mass-Spring-Damper

SVD — Singular Value Decomposition

Deformable Object Model:

k — Euclidean-space dimension

N — number of nodes

$\boldsymbol{d}(\cdot) \in \mathbb{R}^{kN}$ — deformation vector

$M(\cdot) \in \mathbb{R}^{kN \times kN}$ — mass matrix

$V(\cdot) \in \mathbb{R}^{kN \times kN}$ — damping matrix

$K(\cdot) \in \mathbb{R}^{kN \times kN}$ — stiffness matrix

$\tilde{B} \in \mathbb{R}^{kN \times m}$ — influence matrix

$\tilde{L}^T \in \mathbb{R}^{p \times kN}$ — output selector matrix

$\boldsymbol{p}_c^i$ — ith control point

$\boldsymbol{p}_m^i$ — ith manipulation point

State-space System:

n — state vector dimension

m — input vector dimension

p — output vector dimension

p_m — measured output vector dimension

$\boldsymbol{x}(\cdot) \in \mathbb{R}^n$ — state vector

$\boldsymbol{u}(\cdot) \in \mathbb{R}^m$ — input or control vector

$\boldsymbol{y}(\cdot) \in \mathbb{R}^p$ — output vector

$\boldsymbol{y}_m(\cdot) \in \mathbb{R}^{p_m}$ — measured output vector

$\boldsymbol{e}(\cdot) \in \mathbb{R}^p$ — error vector

$\boldsymbol{r}(\cdot) \in \mathbb{R}^p$ — reference trajectory

$\boldsymbol{p}(\cdot) \in \mathbb{R}^{n_p}$ — external disturbance vector

$\boldsymbol{v}(\cdot) \in \mathbb{R}^{n_p+p}$ — exogeneous signal vector (disturbances and references)

$\boldsymbol{w}(\cdot) \in \mathbb{R}^{n_w}$ — uncertainty vector

$A(\cdot) \in \mathbb{R}^{n \times n}$ — state matrix

$B(\cdot) \in \mathbb{R}^{n \times m}$ — input matrix

$C(\cdot) \in \mathbb{R}^{p \times n}$ — output matrix

$D(\cdot) \in \mathbb{R}^{p \times m}$ — feedthrough matrix

$C_m(\cdot) \in \mathbb{R}^{p_m \times n}$ — measured output matrix

$D_m(\cdot) \in \mathbb{R}^{p_m \times m}$ — measured output feedthrough matrix

$E(\cdot) \in \mathbb{R}^{n \times n_p+p}$ — matrix representing exogenous signal effect on state

$F(\cdot) \in \mathbb{R}^{p \times n_p+p}$ — matrix representing exogenous signal effect on error

$L(\cdot) \in \mathbb{R}^{n \times p}$ — estimator gain matrix

$\boldsymbol{n}_u$ — process noise

$\boldsymbol{n}_{y_m}$ — measurement noise

Robot Hybrid Control:

$\boldsymbol{\theta}_a$ — actual robot joint angles

$\boldsymbol{\theta}_e$ — robot joint angle error

$\boldsymbol{\theta}_{e_S}$ — selected subspace joint angle error

$\boldsymbol{x}_d$ — desired end-effector position/orientation

$\boldsymbol{x}_a$ — actual end-effector position/orientation

$\boldsymbol{x}_e$ — end-effector position/orientation error

$\boldsymbol{x}_{e_S}$ — selected end-effector position/orientation error

$\boldsymbol{f}_d$ — desired end-effector force/torque

$\boldsymbol{f}_a$ — actual end-effector force/torque

$\boldsymbol{f}_e$ — end-effector force/torque error

$\boldsymbol{f}_{e_S}$ — selected end-effector force/torque error

$\boldsymbol{\tau}_{e_S}$ — selected joint torque error

$\boldsymbol{\tau}_p$ — commanded joint torque from position

$\boldsymbol{\tau}_f$ — commanded joint torque from force

S — subspace selection matrix

$S^{\perp}$ — orthogonal subspace selection

Contents

List of Figures

Chapter 1

Introduction

Robotics have become increasingly popular over the last 50 years, especially in industries for manufacturing, assembly, and mass production. In most of these applications the objects handled are assumed to be rigid. More recently, the use of robotics has expanded to other fields such as exploration and medical procedures. As the areas in which robots operate develops, the tasks that they perform becomes more elaborate and complex. These tasks may require interaction with objects that cannot be assumed rigid. In fact, the task may require the grasping and maneuvering of compliant objects, controlling the specific deformation of the object, or both. Examples could include installing a rubber hose by stretching the opening over an exhaust port, tying a knot in rope or string, or folding a piece of textile fabric. In these cases, we not only need to accurately control the manipulators end effector, but to also predict the motion of the deformable object in question in order to command the robot intelligently.

Research into deformable object handling is primarily focused into two distinct fields: deformable object modeling, and robotic control strategies. The former focuses

on improving the knowledge base of predicting how an object will deform given its properties and whatever information is available about its current state. The latter is concerned with planning and commanding the robot manipulator given the knowledge of our object to precisely interact with said objects in a desirable fashion.

This thesis focuses on the control design aspect, specifically, in controlling the deformation of the non-rigid object.

1.1 Problem Statement

The task addressed in this thesis is to deform an object into a desired configuration as specified by a certain number of "control points" within the object. This is accomplished by applying forces at different locations on the object, labeled as "manipulation points." This is often referred to as the indirect simultaneous positioning problem. This problem is described here in the 2-dimensional, planar case. The planar object is an elastic material that is required to be deformed in a specific way. The final shape is defined by specifying the desired locations of points on the object that are labeled as control points, $\boldsymbol{p}_c^i$ for $i = 1, \ldots, N_c$. The deformation is accomplished by applying forces to separate points on the object, for instance at the object boundaries, that are labeled manipulation points, $\boldsymbol{p}_m^i$ for $i = 1, \ldots, N_m$. The problem then is to find the manipulation point positions, or similarly the input forces, that will cause the control point positions to converge to their desired locations. This is illustrated in Figure 1.1.

For this thesis, we assume that the forces are applied to the object using a robot manipulator equipped with a force sensor at the end-effector. Given a dynamic deformable object model that approximately predicts the motion of the material, the

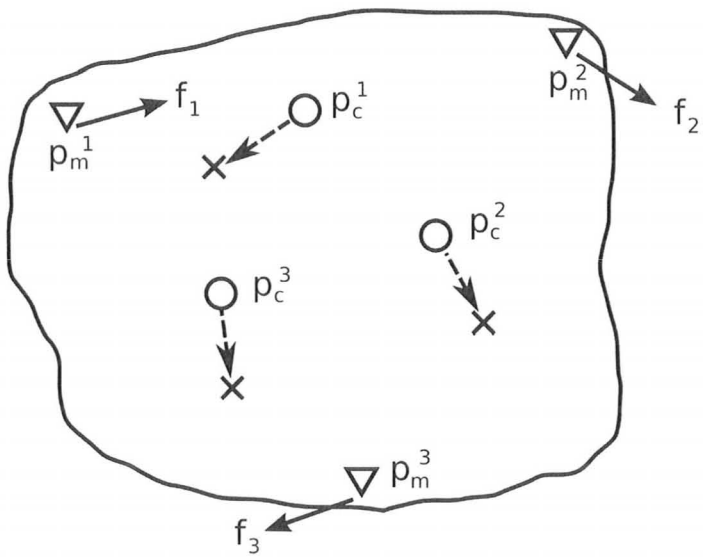


Figure 1.1: Deformable object showing manipulation points with force inputs and control points with target position outputs.

goal is to create a control scheme that determines the force inputs while monitoring the output position of the control points.

An illustrative example of this task in the medical world is during a breast biopsy; the needle is inserted into the tissue with the intention of extracting a sample from a specific target location. However, the needle can cause the tissue to deform and move this target location resulting in a poor biopsy sample. If we monitor the target location using some imaging modality, and place robotic paddles at the boundary of the tissue, then we can "stabilize" the target location to be on the needle path. Another example found in the literature is when manipulating textile fabrics. Certain locations in the fabric may have required destinations to complete a task, such as for a sewing operation. Clearly, the manipulator cannot move the the specific locations directly or else interfere with the sewing needle, motivating an indirect positioning

scheme.

1.2 Thesis Contribution

This research proposes a solution to the indirect simultaneous positioning problem based on a control technique known as robust linear output regulation. The controller will determine the forces necessary to apply to the manipulation points in order to accurately deform the object. Deformable object models based on mesh-free methods form the basis for this control scheme. The model is linearized and treated as a classic output regulation problem. Using this approach we can derive, systematically, controller gains that ensure stability and convergence of the target control point to desired set-point trajectories. Additionally, since deformable object models can become significantly complex, the use of model reduction is examined in our control scheme when determining the gains for a reduced-order controller.

The controller is designed using the following steps:

1. Model the deformable object using a suitable method such as FEM, mesh-free, mass-spring-damper, etc.

2. Linearize the model about the undeformed state to obtain a state-space representation

3. Apply model reduction to reduce the state dimension of the plant

4. Test if the regulation solution exists by performing stabilizability, detectability, and solvability tests on the model

5. Formulate the controller and state-estimator gains using linear robust output regulation theory

The controller design is validated in simulations using MATLAB's Simulink® and a nonlinear mesh-free deformable object model programmed in C++. Experiments were conducted using CRS Catalyst-5 robot manipulators to apply the manipulation point forces to a soft liquid plastic object. A stereoscopic camera monitors the output positions of the control points using infrared LEDs for measurement feedback to the controller. The results demonstrate the effectiveness of applying a robust linear output regulation based control scheme to a robot manipulator to accurately manipulate a deformable object.

Deformable object models are becoming more readily available and better understood, allowing more accurate and sophisticated control schemes to be developed. The benefits of incorporating robots to manipulate these objects can be realized, whether it be for automotive assembly tasks, textile fabric handling, or surgical procedures. Tasks that still predominantly require human workers can become automated to reduce costs, eliminate tiresome repetition, and increase precision and accuracy.

1.3 Organization of Thesis

Chapter 2 highlights the current state of research in the area of deformable object manipulation. Chapter 3 outlines the basic theory behind formulating a robust output regulation control. Following, the details of performing model reduction on the object model are given in Chapter 4. Chapter 5 formulates the control law as it applies to our models and details the simulation and experimental setup. The results of simulations

and experimental tests are presented in Chapter 6. Finally, Chapter 7 discusses the conclusions regarding the results and recommendations for further developments.

This research has resulted in the following papers:

- Fanson, R. and Patriciu, A. (2010). "Model Based Deformable Object Manipulation Using Linear Robust Output Regulation". In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* (To appear.)

- Fanson, R. and Patriciu, A. (2010). "Robotic Assisted Deformable Object Manipulation." In preparation to be submitted to the *International Journal of Robotics Research (IJRR).*

Chapter 2

Literature Review

There are numerous industries that benefit from the ongoing research of deformable object manipulation tasks such as knot tying, assembly of rubber parts, textile fabric handling, sheet metal handling, and medical robotics, to name a few. The active areas of research can be divided into four main categories: object modeling, planning and control, collaborative systems, and industrial experiences (Henrich and Worn, 2000). In this thesis, the primary focus is on the control aspect. More specifically, the goal is to accurately control the specific positions of points within a deformable object. In much of the literature, this task is referred to as indirect simultaneous positioning. The basis for the controller is a deformable object model constructed from one of several approaches available to approximate the object deformation. In this section, we highlight some previous research in the areas of deformable object modeling and robotic interactions with non-rigid materials.

2.1 Deformable Object Modeling

Deformable object modeling emerged for use in computer graphics. The applications have grown from early use in animation, to a basis for training in virtual reality simulators, and recently haptic environments and robotic deformation control. Many of the methods that have been explored throughout the years are summarized in a recent survey by Gibson and Mirtich (1997).

For animation purposes, many of the early techniques were based on non-physical models, where the designer simply defines the desired deformation manually using some control points or parameter adjustment. The object's curve or surface would be inferred from these points using some mathematical interpolation such as a cubic spline. Later models become more based on physical representations of objects rather than a manual definition of the deformation. Taking advantage of growing computational power, this shifts the workload of defining the deformation onto computer calculations rather than an animator's manual specification. Much of these physical models are based upon mass-spring-damper networks. Platt and Badler (1981) demonstrate this in early work by using tension nets to animate facial expressions. Terzopoulos and Waters (1990) used a dynamic mass-spring-damper model consisting of three separate layers of tissue to model the human face.

Mass-spring-damper models are intuitive and well-understood, however the approximation is often crude and may not truly represent the physical system. Also, it can be difficult to approximate the spring stiffness and damping coefficients of the physical objects. Current deformation analysis is generally done using models that are derived from equations of continuum mechanics, such as finite element methods (FEM). FEM models can be used to perform both static analysis to determine the

final deformation of an object given a set of external forces, or dynamical analysis where the inertial and damping (or viscous) forces are also taken into account. Examples of FEM use in deformable object modeling can be found in Gourret *et al.* (1989) who use FEM to animate the deformations that occur between a human hand and a deformable object in a grasping task. Bro-nielsen and Cotin (1996) apply FEM to predict real-time tissue deformations for use in virtual reality surgical simulators.

Although FEM maintains a dominate role in deformation analysis in many fields, large deformations or structural changes often involve recomputation and remeshing of the object. Recently, there has been significant development in mesh-free methods that can handle these large deformations without the need for remeshing computations. These methods approximate the deformation using only nodes and no elements or mesh. Several variations exist including Smooth Particle Hydrodynamics (SPH), Particle in Cell Methods (PIC), Diffuse Element Methods (DEM), Element-Free Galerkin Methods (EFG) and Reproducing Kernel Particle Methods (RKPM) (Chen *et al.*, 1996). A survey of these methods is presented in Belytschko *et al.* (1996) and more recently in Babuska *et al.* (2002). Chen *et al.* (1997) describe the details of using the RKPM method to approximate large deformations in nonlinear bodies. They then employ this method to perform large deformation analysis of hyperelastic and nearly incompressible rubber materials.

The control law in this thesis is derived independent of the modeling technique used and assumes only that the method provides a dynamic analysis of the deformable object. The dynamic model then yields a second order differential equation of the form

$$M(\boldsymbol{d})\ddot{\boldsymbol{d}} + V(\dot{\boldsymbol{d}}, \boldsymbol{d})\dot{\boldsymbol{d}} + K(\boldsymbol{d})\boldsymbol{d} = \boldsymbol{f}.$$

For the simulations a experimental results presented in this thesis, we use the RKPM method for the deformable objects model and the basis for the controller derivation.

2.2 Robot and Deformable Object Interaction

Although much of the deformable object modeling developed to this date was influenced by the needs of the computer animation world, their use in robotics is spawned by a growing need for automated deformable object interactions in industry. A recent overview provided by Henrich and Worn (2000) outlines a few of the research papers published with regard to modeling, control, and industrial applications of robotic manipulation of deformable objects. Additionally, Saadat and Nan (2002) describe much of the industrial applications that have focused on being able to manipulate deformable objects. According to Saadat, specific industrial tasks are defined and categorized based on either 1D (linear), 2D (sheet/planar), and 3D materials. Currently, the majority of research is focused on 2D applications and motivated by fabric and garment applications, but as robotics in medical and food industries become more popular, much new research may shift towards 3D applications.

Much of the linear robotic and deformable object research is focused on manipulation of wires, cables, ropes, and knot-tying tasks. Chen and Zheng use cubic spline functions in conjunction with a vision system to approximate beam deformation in Chen and Zheng (1991) and use this model to mate a flexible beam with a rigid hole in Zheng *et al.* (1991). Wakamatsu *et al.* model linear objects by minimizing the potential energy of the object subject to the geometric constraints as seen in (Wakamatsu *et al.*, 1995), (Wakamatsu *et al.*, 1996), (Wakamatsu *et al.*, 2002), and (Wakamatsu and Hirai, 2004). Nakagaki *et al.* (1997) expand on the modeling work of Wakamatsu

for inserting a flexible beam into a rigid hold. They account for plastic deformation of a wire by measuring the wire shape via a stereoscopic camera and measure the force at the tip of the wire.

For practical deformable object manipulation, the notions of image-based feedback and object modeling are indispensable. They are a common basis in most applications of deformable object manipulation where the desired task is to accurately control the deformation of a soft object. This generally involves defining certain points within the object which are manipulated inputs, and certain points within the object that are monitored output positions to be controlled, commonly referred to as indirect simultaneous positioning.

2.2.1 Indirect Simultaneous Positioning

The indirect simultaneous positioning problem was first addressed in Wada *et al.* (1998). Their work focuses primarily on textile fabrics which they model using a mass-spring-damper system and use the linearization of this model as the basis for their control law. The control law is based on a Jacobian relationship between the manipulation and control points. The convergence of the control points to their desired positions is based on iterative control method that incrementally determines the manipulation points that will result in the desired control points. Wada *et al.* (1999) then also incorporate the planning of robot finger location, and in Wada *et al.* (2001) they incorporate PID control on the feedback to ensure robustness on an approximate model, provided the deformation is not too great for the given model.

Shibata and Hirai (2006) use a similar model based PID control to control the position and deformation of a soft object. However, they use this to determine the

force inputs at certain location on the object that are required to achieve the desired positions of other points. Also, their tests only confirm the stability on a linear, 1-dimensional, mass-spring-damper models.

Smolen and Patriciu (2009a) take a slightly different approach which uses the reproducing kernel particle method (RKPM) to model the deformable object. The model is used to calculate a Jacobian transformation between the manipulation points and the control points such that the manipulation points are moved incrementally to positions that will yield the desired control point locations.

2.2.2 Needle Target Stabilization

Significant literature in deformable object manipulation is focused on medical needle insertions. In this application, typically there is a target location with a soft tissue where a task must be performed, such as taking a biopsy sample. Upon insertion of the needle, soft objects typically undergo deformation that causes the target point to move. A possible solution involves having the surgeon use steerable needles and image guidance, such as ultrasound, to try to hit the target. The work of DiMaio in DiMaio and Salcudean (2003a), DiMaio and Salcudean (2003b), DiMaio and Salcudean (2005b), and DiMaio and Salcudean (2005a), focuses on combining deformable object and flexible needle models to determine a complex needle trajectory that accounts for any deformation of the target or needle in the insertion process.

Other research shows promising results of using robot manipulators at the boundary of the tissue to instead keep the target location on the needle path. A noteworthy paper by Mallapragada *et al.* (2007) describe a PI control scheme to keep a target

biopsy location on a needle path defined in a plane. They apply forces on the boundary of a cylindrical object using two, 1-DOF robotic fingers positioned 90° to each other and a fixed boundary opposite each finger. In Torabi *et al.* (2009) the information about the target point and potential obstacles is used to plan manipulation point locations and the needle insertion site for prostate biopsies. The manipulators actuate one at a time in a phased control scheme as the needle is inserted to both avoid sensitive tissues and minimize targeting errors. The control is verified using 2D nonhomogeneous mass-spring tissue models. Smolen and Patriciu (2009b) stabilize the target to the needle path using an RKPM model and a Jacobian relationship between a paddle placed at the object boundary and the target point. Their simulations test both a static paddle and dynamic paddle placement. Their results show that a static paddle placed optimally at a boundary location can help minimize targeting error but depends greatly on the object properties. In contrast, a dynamic paddle placed at a location that gives optimal control over the target location is much more justified for nonhomogeneous objects.

Chapter 3

Output Regulation Control

The control law applied to the deformable object is based on a dynamic second-order model of a discretized object, such as finite element model, mesh-free model, or mass-spring damper system. The nonlinear dynamic system can then be linearized about the equilibrium point, and reformulated to a set of first order differential equations and represented as a state-space model. This form allows the problem to be approached as a classic output regulation control problem. Any nonlinearities, modeling uncertainties, and unknown disturbances to the system that cause a deviation from the nominal plant within some open neighbourhood, are compensated with a robust controller. The next sections outline the linearization and reformulation of the dynamic model into a state-space system and the derivation of a robust control law.

3.1 State-Space Representation

Assume that a deformable object is discretized and modeled using N points, or nodes, distributed throughout the body. The control points, $\boldsymbol{p}_c^i$ and manipulation points, $\boldsymbol{p}_m^i$

are assumed to be explicitly included as nodes. The object can be linear, planar, or 3-dimensional. Let the number of space dimensions be k and therefore the displacement of each node can be represented by a vector (or scalar) in $\mathbb{R}^k$. Let the displacement of all nodes be represented by the stacked vector $\boldsymbol{d}(t) \in \mathbb{R}^{kN}$. In general, the equations of motion for the object can be described by the second order system

$$M(\boldsymbol{d})\ddot{\boldsymbol{d}}(t) + V(\dot{\boldsymbol{d}}, \boldsymbol{d})\dot{\boldsymbol{d}}(t) + K(\boldsymbol{d})\boldsymbol{d}(t) = \boldsymbol{f}(t). \tag{3.1}$$

The $kN \times kN$ matrices $M(\boldsymbol{d})$, $V(\dot{\boldsymbol{d}}, \boldsymbol{d})$, and $K(\boldsymbol{d})$ represent the mass, damping, and stiffness matrices, respectively, and $\boldsymbol{f}(t) \in \mathbb{R}^{kN}$ represents the vector of external force distribution applied to each node. The mass, damping, and stiffness matrices are most likely to be non-linear. This dynamical form is shown to be obtainable from several different deformable object modeling approaches in Gibson and Mirtich (1997).

Our goal is to linearize and reformulate the nonlinear dynamical model to a familiar, continuous, time-invariant state space representation

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= A\boldsymbol{x}(t) + B\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= C\boldsymbol{x}(t) + D\boldsymbol{u}(t) \end{aligned} \tag{3.2}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^n$ is the state vector, $\boldsymbol{u}(t) \in \mathbb{R}^m$ is the input vector, $\boldsymbol{y}(t) \in \mathbb{R}^p$ is the output vector. Note that by doing so, the linear model is only a good approximation for deformations within a certain neighbourhood of the equilibrium point.

Referring to the dynamical model (3.1), assuming that the control points coincide with nodes of the discretized model, the outputs for this system are simply the positional displacements of these nodes. Accounting for output, the dynamical system is

now described by

$$M(\boldsymbol{d})\ddot{\boldsymbol{d}}(t) + V(\dot{\boldsymbol{d}}, \boldsymbol{d})\dot{\boldsymbol{d}}(t) + K(\boldsymbol{d})\boldsymbol{d}(t) = \tilde{B}\boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \tilde{L}^T\boldsymbol{d}(t). \tag{3.3}$$

The matrix $\tilde{L}^T \in \mathbb{R}^{p\times kN}$ is a *selector* matrix with entries of simply 1 or 0 which define which displacements represent the positions of our control point nodes of interest The notation $\tilde{L}^T$ was chosen to be consistent with model reduction presented in Chapter 4. Note that the framework does not change if the control points do not coincide with model nodes but rather lie somewhere in between. The output of control point displacements could then be defined as some interpolation of the displacements surrounding nodes, resulting in a more complex output matrix. The assumption merely simplifies the formulation for clarity. Additionally, the external force vector is decomposed into $\boldsymbol{f}(t) = \tilde{B}\boldsymbol{u}(t)$ such that $\boldsymbol{u}(t) \in \mathbb{R}^m$ includes only input (or control) force signals and $\tilde{B}$ in an *influence* matrix that describes how they distribute amongst the individual nodes.

First, rearrange the system in (3.3) to

$$\ddot{\boldsymbol{d}}(t) = -M(\boldsymbol{d})^{-1}K(\boldsymbol{d})\boldsymbol{d}(t) - M(\boldsymbol{d})^{-1}V(\dot{\boldsymbol{d}}, \boldsymbol{d})\dot{\boldsymbol{d}}(t) + M(\boldsymbol{d})^{-1}\tilde{B}\boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \tilde{L}^T\boldsymbol{d}(t) \tag{3.4}$$

where the mass matrix M is positive definite for a deformable object and therefore invertible (Gibson and Mirtich, 1997).

Note that this is still a nonlinear system that would be best described in state

space form as

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= f(\boldsymbol{x}(t), \boldsymbol{u}(t)) \\ \boldsymbol{y}(t) &= h(\boldsymbol{x}(t), \boldsymbol{u}(t)). \end{aligned} \tag{3.5}$$

Therefore, we linearize the system about the equilibrium point by evaluating the $M(\boldsymbol{d})$, $V(\dot{\boldsymbol{d}}, \boldsymbol{d})$, and $K(\boldsymbol{d})$ matrices of the object at the undeformed state. In other words, we extract from our model $M(0)$, $V(0,0)$, and $K(0)$, which we will abbreviate to simply M, V, and K. Then, rewrite (3.4) and including the trivial equation $\dot{\boldsymbol{d}} = \dot{\boldsymbol{d}}$ to yield

$$\begin{aligned} \dot{\boldsymbol{d}} &= \dot{\boldsymbol{d}} \\ \ddot{\boldsymbol{d}} &= -M^{-1}K\boldsymbol{d} - M^{-1}V\dot{\boldsymbol{d}} + M^{-1}\tilde{B}\boldsymbol{u} \\ \boldsymbol{y} &= \tilde{L}^T\boldsymbol{d} \end{aligned}$$

or in matrix form

$$\begin{aligned} \begin{bmatrix} \dot{\boldsymbol{d}} \\ \ddot{\boldsymbol{d}} \end{bmatrix} &= \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}V \end{bmatrix} \begin{bmatrix} \boldsymbol{d} \\ \dot{\boldsymbol{d}} \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}\tilde{B} \end{bmatrix} \boldsymbol{u} \\ \boldsymbol{y} &= \begin{bmatrix} \tilde{L}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{d} \\ \dot{\boldsymbol{d}} \end{bmatrix}. \end{aligned} \tag{3.6}$$

By choosing the state $\boldsymbol{x} = \begin{bmatrix} \boldsymbol{d}^T & \dot{\boldsymbol{d}}^T \end{bmatrix}^T$ we can write

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} \\ \boldsymbol{y} &= C\boldsymbol{x} \end{aligned} \tag{3.7}$$

where the linear state space matrices are

$$A = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}V \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ M^{-1}\tilde{B} \end{bmatrix} \quad C = \begin{bmatrix} \tilde{L}^T & 0 \end{bmatrix} \quad D = \begin{bmatrix} 0 \end{bmatrix}.$$

Note that these matrices are effectively the same as finding the partial derivatives of our nonlinear state-space equation in (3.5):

$$A = \frac{\partial f}{\partial x}(0,0), \qquad B = \frac{\partial f}{\partial u}(0,0), \qquad C = \frac{\partial h}{\partial x}(0,0), \qquad D = \frac{\partial h}{\partial u}(0,0)$$

evaluated at the origin $(\boldsymbol{x}, \boldsymbol{u}) = (0, 0)$. This linear, time-invariant state space model of our deformable object will be used for the output regulation control described in the next section.

3.2 Regulating Control Law

Here, we consider the linearized state-space model described in the previous sections and formulate a robust controller that will compensate for the modeling uncertainties and nonlinearities. Much of this formulation is based on the output regulation theory first described by Francis (1977) and further detailed in Huang (2004). For

completeness, the key concepts of the control law formulation will be highlighted in this section. This section follows the formulation presented in Huang (2004). For further details the reader is referred to his work.

3.2.1 Linear Output Regulation

We begin by taking the general state-space model of a linear time invariant system subject to disturbances, $\boldsymbol{p}$.

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} + E_p\boldsymbol{p} \\ \boldsymbol{y} &= C\boldsymbol{x} + D\boldsymbol{u} + F_p\boldsymbol{p}. \end{aligned} \tag{3.8}$$

Given a reference trajectory, $\boldsymbol{r}(t)$, the tracking error we wish to regulate is

$$\boldsymbol{e}(t) = C\boldsymbol{x} + D\boldsymbol{u} + F_p\boldsymbol{p} - \boldsymbol{r}. \tag{3.9}$$

The goal is to find the control signal, $\boldsymbol{u}(t)$ such that

$$\lim_{t\to\infty} \boldsymbol{e}(t) = 0$$

If we combine the disturbances and reference trajectories into a vector $\boldsymbol{v} = \begin{bmatrix} \boldsymbol{r}^T & \boldsymbol{p}^T \end{bmatrix}^T$ representing all exogenous signals with known dynamics to be $\dot{\boldsymbol{v}} = A_1\boldsymbol{v}$, we can rewrite

the plant and exosystem as

$$\begin{aligned}\dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} + E\boldsymbol{v} \\ \dot{\boldsymbol{v}} &= A_1\boldsymbol{v} \\ \boldsymbol{e} &= C\boldsymbol{x} + D\boldsymbol{u} + F\boldsymbol{v}.\end{aligned} \tag{3.10}$$

where $E = \begin{bmatrix} 0 & E_p \end{bmatrix}$ and $F = \begin{bmatrix} -I & F_p \end{bmatrix}$.

A solution to the regulation exists if there exists matrices X and U such that

$$\begin{aligned}XA_1 &= AX + BU + E \\ 0 &= CX + DU + F.\end{aligned} \tag{3.11}$$

These are known as the *regulator equations*. In fact, Huang (2004) shows that a solution exists if the following conditions are satisfied:

Condition 3.1. *The pair* (A, B) *is stabilizable*

Condition 3.2. *The pair* $\left(\begin{bmatrix} C & F \end{bmatrix}, \begin{bmatrix} A & E \\ A_1 & 0 \end{bmatrix} \right)$ *is detectable*

Condition 3.3. *For each eigenvalue,* λ*, of the matrix* A_1*, we have* $\operatorname{rank} \begin{bmatrix} A - \lambda I & B \\ C & D \end{bmatrix} = n + p$

To help interpret the regulator equations, consider the simplified problem where the plant is disturbance free and the exogenous system consists of only set-point

reference trajectories. Therefore $A_1 = 0$ and

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} \\ \boldsymbol{y} &= C\boldsymbol{x} + D\boldsymbol{u}. \end{aligned} \tag{3.12}$$

In order to achieve regulation, there needs to be a final equilibrium state, $\boldsymbol{x}^*$, and control signal, $\boldsymbol{u}^*$, such that the system dynamics have reached steady-state and the system output has reached the reference signal. Mathematically,

$$\begin{aligned} \lim_{t\to\infty} \dot{\boldsymbol{x}}(t) &= A\boldsymbol{x}^* + B\boldsymbol{u}^* = 0 \\ \lim_{t\to\infty} \boldsymbol{y}(t) &= C\boldsymbol{x}^* + D\boldsymbol{u}^* = \boldsymbol{r}. \end{aligned} \tag{3.13}$$

We can write this as

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^* \\ \boldsymbol{u}^* \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{r} \end{bmatrix} \tag{3.14}$$

for which a solution exists if the matrix on the left has full row rank, implying we must have at least as many control inputs, m, as outputs, p. A *unique* solution exists if the matrix is invertible which would imply the number of control inputs, m, equals the number of outputs, p, and condition 3.3 is satisfied. This gives

$$\begin{bmatrix} \boldsymbol{x}^* \\ \boldsymbol{u}^* \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \boldsymbol{r} \end{bmatrix} = \begin{bmatrix} M & X \\ N & U \end{bmatrix} \begin{bmatrix} 0 \\ \boldsymbol{r} \end{bmatrix} \tag{3.15}$$

Wada *et al.* (1999) emphasize the condition that the number of manipulation points must at least equal the number of control points. However, they do not explicitly

state that there may be boundary conditions or characteristics of the object which forbid a solution. Condition 3.3 is essential to ensure a solution exists.

Furthermore, if $m > p$ and the solution to (3.14) does exist, there are multiple solutions for $\boldsymbol{x}^*$ and $\boldsymbol{u}^*$. One possible solution is the *right pseudoinverse* represented by

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^T \left(\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix}^T \right)^{-1}. \tag{3.16}$$

In the situations where $m > p$ it may be advantageous to use the extra degrees of freedom to impose additional constraints on the system. In fact the pseudoinverse stated above results in the *min-norm* solution which minimizes the 2-norms of the final state $\boldsymbol{x}^*$ and control signal $\boldsymbol{u}^*$.

From these linear equations, when the solution does exist, we can find $\boldsymbol{x}^* = X\boldsymbol{r}$ and $\boldsymbol{u}^* = U\boldsymbol{r}$. At steady-state this gives,

$$\begin{aligned} \dot{\boldsymbol{x}} &= AX\boldsymbol{r} + BU\boldsymbol{r} \\ \boldsymbol{y} &= CX\boldsymbol{r} + DU\boldsymbol{r}. \end{aligned} \tag{3.17}$$

For this to be equivalent to equations (3.13) we need X and U to satisfy

$$\begin{aligned} 0 &= AX + BU \\ 0 &= CX + DU - I \end{aligned} \tag{3.18}$$

which are exactly the regulator equations for the set-point, disturbance-free case.

To obtain the generic regulator equations we simply reintroduce the disturbances

to the exogenous signal $\boldsymbol{v}$ with matrices $E = \begin{bmatrix} E_p & 0 \end{bmatrix}$ and $F = \begin{bmatrix} F_p & -I \end{bmatrix}$. In this case, the dynamics will not settle to zero, but rather to an invariant subspace which is a mapping of the exogenous signals $\boldsymbol{x} = X\boldsymbol{v}$ and thus $\dot{\boldsymbol{x}} = X\dot{\boldsymbol{v}} = XA_1\boldsymbol{v}$, yielding

$$\begin{aligned} \dot{\boldsymbol{x}} &= (AX + BU + E)\boldsymbol{v} = XA_1\boldsymbol{v} \\ \boldsymbol{e} &= (CX + DU + F)\boldsymbol{v} = 0. \end{aligned} \tag{3.19}$$

where the feedforward control signal $\boldsymbol{u} = U\boldsymbol{r}$ keeps the error $\boldsymbol{e}$ at zero (Isidori *et al.*, 2003). At the heart of the regulation problem is asymptotically approaching this invariant subspace for any initial conditions.

For the proof that a solution to the regulator equations still exists, the reader is referred to Huang (2004) and Isidori *et al.* (2003).

As explained in Huang (2004), assume that the full-state is available for feedback and that there is no uncertainty about the plant. The controller considered to solve the regulation problem is given by

$$\boldsymbol{u} = K_x\boldsymbol{x} + K_v\boldsymbol{v} \tag{3.20}$$

Under this control scheme, the closed loop system becomes,

$$\begin{aligned} \dot{\boldsymbol{x}} &= (A + BK_x)\boldsymbol{x} + (E + BK_v)\boldsymbol{v} \\ \dot{\boldsymbol{v}} &= A_1\boldsymbol{v} \\ \boldsymbol{e} &= (C + DK_x)\boldsymbol{x} + (DK_v + F)\boldsymbol{v} \end{aligned} \tag{3.21}$$

From before, we know that if the system can be regulated, the steady-state trajectories

will be

$$\begin{aligned}
\dot{\boldsymbol{x}}^* &= (\dot{X}\boldsymbol{v}) = X\dot{\boldsymbol{v}} = XA_1\boldsymbol{v} \\
\boldsymbol{x}^* &= X\boldsymbol{v} \\
\boldsymbol{u}^* &= U\boldsymbol{v} \\
\boldsymbol{e} &= 0.
\end{aligned} \tag{3.22}$$

The closed loop system is now

$$\begin{aligned}
XA_1\boldsymbol{v} &= (A + BK_x)X\boldsymbol{v} + (E + BK_v)\boldsymbol{v} \\
0 &= (C + DK_x)X\boldsymbol{v} + (DK_v + F)\boldsymbol{v}
\end{aligned} \tag{3.23}$$

From this, to achieve regulation we could design K_x to stabilize $(A + BK_x)$, then solve (3.23) for X and K_v to get the final controller (3.20). However, this would require resolving for X and K_v for every redesign of the matrix K_x. Noting also our controller form (3.20), on the invariant subspace we have the input signal

$$U\boldsymbol{v} = K_xX\boldsymbol{v} + K_v\boldsymbol{v} \tag{3.24}$$

Substituting (3.24) into (3.23) to eliminate K_v and K_x and dropping the common $\boldsymbol{v}$ terms results in the regulator equations (3.11). Thus we can instead solve for X and U from (3.11) independently from designing K_x and K_v

The controller that solves the regulation problem can be summarized as follows:

1. Solve the regulator equations (3.11) (if possible) for X and U

2. Design the feedback gain matrix K_x such that $(A + BK_x)$ is stable

3. Design the feedforward gain matrix K_v such that $K_v = U - K_x X$

The static state feedback controller is then given by (3.20).

3.2.2 Regulation with an Observer

The controller given in (3.20) assumes that the full-state is available for feedback. For a deformable object model, this may not be realistic. The number of discretization points, or nodes, can become quite large and it is unlikely that the position and velocity of both the x- and y-direction of each node is available for full state feedback. The solution is to employ a state estimator or observer. The control form becomes

$$\begin{aligned} u &= K_x \hat{\boldsymbol{x}} + K_v \boldsymbol{v} \\ \dot{\hat{\boldsymbol{x}}} &= A\hat{\boldsymbol{x}} + B\boldsymbol{u} + E\boldsymbol{v} + L(\boldsymbol{y_m} - C_m\hat{\boldsymbol{x}} - D_m\boldsymbol{u}) \end{aligned} \tag{3.25}$$

Note that we have used the subscript m to differentiate between measured output $\boldsymbol{y}_m$ and regulated output $\boldsymbol{y}$ for cases when we can measure more outputs to improve our estimation.

Let the closed loop system state be $\boldsymbol{x}_c = [\boldsymbol{x}^T \ \boldsymbol{e_x}^T]^T$, where $\boldsymbol{e_x} = \boldsymbol{x} - \hat{\boldsymbol{x}}$ is the state

estimation error. The closed loop system becomes

$$\begin{aligned}
\dot{\boldsymbol{x}}_c &= \begin{bmatrix} A + BK_x & -BK_x \\ 0 & A - LC_m \end{bmatrix} \boldsymbol{x}_c + \begin{bmatrix} E + BK_v \\ 0 \end{bmatrix} \boldsymbol{v} \\
\dot{\boldsymbol{v}} &= A_1 \boldsymbol{v} \\
\boldsymbol{e} &= \begin{bmatrix} (C + DK_x) & -DK_x \end{bmatrix} \boldsymbol{x}_c + \begin{bmatrix} DK_v + F \end{bmatrix} \boldsymbol{v}
\end{aligned} \tag{3.26}$$

By the well-known separation principle, the closed loop system is stable as long as the blocks $(A + BK_x)$ and $(A - LC_m)$ are each stable.

The controller is now designed from

1. Solve the regulator equations (3.11) (if possible) for X and U
2. Design the feedback gain matrix K_x such that $(A + BK_x)$ is stable
3. Design the feedforward gain matrix K_v such that $K_v = U - K_x X$
4. Design the estimator gain matrix L such that $(A - LC_m)$ is stable

and implementing the controller (3.25).

3.2.3 Linear Robust Output Regulation

Often the nominal values of the plant matrices have uncertainty included such that the plant model is described by

$$\begin{aligned}
\dot{\boldsymbol{x}} &= A_w \boldsymbol{x} + B_w \boldsymbol{u} + E_w \boldsymbol{v} \\
\boldsymbol{e} &= C_w \boldsymbol{x} + D_w \boldsymbol{u} + F_w \boldsymbol{v},
\end{aligned} \tag{3.27}$$

where the subscript w indicates the state-space matrices deviate from their nominal values such that

$$A_w = A + \Delta A, \qquad B_w = B + \Delta B, \qquad E_w = E + \Delta E$$
$$C_w = C + \Delta C, \qquad D_w = D + \Delta D, \qquad F_w = F + \Delta F.$$

The classical output regulation considers the case where exogenous signals, $\boldsymbol{v}$, are generated by $\dot{\boldsymbol{v}} = A_1\boldsymbol{v}$. However, for the task at hand we will consider set-point inputs and assume no disturbance signals, making $\boldsymbol{v}$ constant and therefore $A_1 = 0$. As before, the effect of the exogenous signals on the system is incorporated by $E = \begin{bmatrix} 0 & E_p \end{bmatrix}$, and $F = \begin{bmatrix} -I & F_p \end{bmatrix}$.

Following Huang (2004), if the full state is available, the control law can be used for robust output regulation is of the form

$$\begin{aligned} \boldsymbol{u} &= K_1\boldsymbol{x} + K_2\boldsymbol{z} \\ \dot{\boldsymbol{z}} &= G_1\boldsymbol{z} + G_2\boldsymbol{e} \end{aligned} \tag{3.28}$$

where $\boldsymbol{x}$ is the plant state vector, and $\boldsymbol{z}$ is a dynamic controller state vector to be defined later to ensure robustness. The closed loop system, with augmented state $\boldsymbol{x_c} = \begin{bmatrix} \boldsymbol{x}^T & \boldsymbol{z}^T \end{bmatrix}^T$, becomes,

$$\begin{aligned} \dot{\boldsymbol{x}}_c &= A_{cw}\boldsymbol{x}_c + B_{cw}\boldsymbol{v} \\ \boldsymbol{e} &= C_{cw}\boldsymbol{x}_c + D_{cw}\boldsymbol{v} \end{aligned} \tag{3.29}$$

where

$$A_{cw} = \begin{bmatrix} A_w + B_w K_1 & B_w K_2 \\ G_2(C_w + D_w K_1) & G_1 + G_2 D_w K_2 \end{bmatrix}, \qquad B_{cw} = \begin{bmatrix} E_w \\ G_2 F_w \end{bmatrix},$$
$$C_{cw} = \begin{bmatrix} C_w + D_w K_1 & D_w K_2 \end{bmatrix}, \qquad D_{cw} = F_w. \tag{3.30}$$

Let the nominal plant values of this closed loop system be A_{c0},B_{c0},C_{c0},D_{c0}. The output regulation problem now requires to design a controller such that

1. The matrix A_{C0} is Hurwitz

2. For uncertainties, $\boldsymbol{w}$ with some open neighbourhood, W, the error trajectory satisfies

$$\lim_{t\to\infty} \boldsymbol{e}(t) = \lim_{t\to\infty} (C_{cw}\boldsymbol{x}_c(t) + D_{cw}\boldsymbol{v}(t)) = \mathbf{0}. \tag{3.31}$$

Huang shows that this is equivalent to finding the unique matrix X_{cw} that solves the matrix equations:

$$\begin{aligned} X_{cw}A_1 &= A_{cw}X_{cw} + B_{cw} \\ 0 &= C_{cw}X_{cw} + F_{cw}. \end{aligned} \tag{3.32}$$

When the desired reference signals are constant, then $X_{cw}A_1 = 0$.

Equations (3.32) are solvable if (A_w, B_w) is stabilizable, (C_w, A_w) is detectable, and

$$\operatorname{rank} \begin{bmatrix} A_w & B_w \\ C_w & D_w \end{bmatrix} = n + p, \tag{3.33}$$

where n is the total number of states and p is the number of outputs to be regulated.

Put another way, (3.32) requires there to be a mapping to an invariant subspace of the form

$$\mathcal{V} = \{(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{v}) : \boldsymbol{x} = X\boldsymbol{v}, \boldsymbol{z} = Z\boldsymbol{v}\}. \tag{3.34}$$

Given controller form (3.28), this subspace requires there to be matrices X_w and U_w, that solve the plant equation

$$\begin{aligned} X_w A_1 &= A_w X_w + B_w U_w + E_w \\ 0 &= C_w X_w + D_w U_w + F_w \end{aligned} \tag{3.35}$$

as well as a matrix Z_w to solve the controller equation

$$\begin{aligned} Z_w A_1 &= G_1 Z_w + G_2(C_w X_w + D_w U_w + F_w) \\ U_w &= K_1 X_w + K_2 Z_w. \end{aligned} \tag{3.36}$$

Equations (3.35) are the robust version of our regulator equations and equations (3.36) expresses that our controller contains an internal model. Combining (3.35) and (3.36) we can rearrange to

$$\begin{aligned} X_w A_1 &= (A + BK_1)X_w + BK_2 Z_w + E_w \\ Z_w A_1 &= G_2(C_w + D_w K_1)X_w + (G_1 + G_2 D_w K_2)Z_w \\ 0 &= (C + DK_1)X_w + DK_2 Z_w + F_w. \end{aligned} \tag{3.37}$$

Equations (3.37) are in fact the same as (3.32) with

$$X_{cw} = \begin{bmatrix} X_w \\ Z_w \end{bmatrix}.$$

The problem now is essentially finding a solution to the invariant subspace represented by (3.37) from the regulator equations (3.35), and the internal model (3.36) and designing a controller to asymptotically approach this subspace.

Huang (2004) shows that if the dynamic portion of the controller $\dot{\boldsymbol{z}} = G_1\boldsymbol{z} + G_2\boldsymbol{e}$ contains an internal model of the exogenous signals, and (A, B) is stabilizable, then the pair

$$\left(\begin{bmatrix} A & 0 \\ G_2C & G_1 \end{bmatrix}, \begin{bmatrix} B \\ G_2D \end{bmatrix} \right)$$

is also stabilizable. In fact, this pair corresponds to the augmented system,

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} + E\boldsymbol{v} \\ \dot{\boldsymbol{z}} &= G_1\boldsymbol{z} + G_2\boldsymbol{e} \\ \boldsymbol{e} &= C\boldsymbol{x} + D\boldsymbol{u} + F\boldsymbol{v} \end{aligned} \tag{3.38}$$

where the augmented plant matrices are

$$\tilde{A} = \begin{bmatrix} A & 0 \\ G_2C & G_1 \end{bmatrix}, \qquad \tilde{B} = \begin{bmatrix} B \\ G_2D \end{bmatrix}.$$

Then, a stabilizing controller for the augmented system leads to the solution for the linear robust output regulation problem (Huang, 2004).

In order to incorporate an internal model, the matrices G_1 and G_2 can be defined with the form

$$G_1 = \begin{bmatrix} 0 & I & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I \\ -\alpha_k I & -\alpha_{k-1} I & \cdots & -\alpha_1 I \end{bmatrix} \qquad G_2 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ I \end{bmatrix}$$

where I is the dimension of the error vector $\boldsymbol{e}$ and $\alpha_1 \ \ldots \ \alpha_{k-1}$ are the coefficients of the minimal polynomial of A_1:

$$f(\lambda) = \lambda^k + \alpha_1 \lambda^{k-1} + \ldots + \alpha_{k-1} \lambda^1 + \alpha_k.$$

For more details regarding the formulation of a controller with the internal model principle, the reader is referred to Huang (2004) and Isidori *et al.* (2003).

In addition to the dynamic controller state $\boldsymbol{z}$, as before, the positions and velocities must be estimated using a state estimator. This is accomplished by including another dynamic state in the controller, $\hat{\boldsymbol{x}}$. The new controller becomes

$$\begin{aligned} \boldsymbol{u} &= K_1 \hat{\boldsymbol{x}} + K_2 \boldsymbol{z} \\ \dot{\boldsymbol{z}} &= G_1 \boldsymbol{z} + G_2 \boldsymbol{e} \\ \dot{\hat{\boldsymbol{x}}} &= A\hat{\boldsymbol{x}} + B\boldsymbol{u} + E\boldsymbol{v} + L\left(\boldsymbol{y}_m - C_m \hat{\boldsymbol{x}} - D_m \boldsymbol{u}\right) \end{aligned} \tag{3.39}$$

where L is the observer gain matrix and $\hat{\boldsymbol{x}}$ is the estimation of our state variables. Note, again, the subscript m to differentiate between $\boldsymbol{y}_m$ and $\boldsymbol{y}$. Again, we need to

design L such that $A - LC_m$ is Hurwitz to ensure the observer error asymptotically decreases to zero.

To show that this will still solve the output regulation problem we need to show that the closed loop system is stable. Let the closed loop state be $\boldsymbol{x}_c = [\boldsymbol{x}^T \ \boldsymbol{z}^T \ \boldsymbol{e_x}^T]^T$, where we will use the error of the observer, $\boldsymbol{e_x}$, instead of the observer state, $\hat{\boldsymbol{x}}$ because it gives a more useful realization. The closed loop system matrix A_c becomes:

$$A_c = \begin{bmatrix} A + BK_1 & BK_2 & -BK_1 \\ G_2(C + D_K 1) & G_1 + G_2 D K_2 & -G_2 D K_1 \\ 0 & 0 & A - LC_m \end{bmatrix}.$$

By the separation principle, the eigenvalues of the closed loop system are the union of the eigenvalues the augmented system (3.38) and $(A - LC_m)$. Since K_1, K_2 and L are designed such that (3.38) and $(A - LC_m)$ are stable, the closed loop system is stable.

The overall process can be summarized as follows:

1. Model the object using the preferred method (e.g. FEM, mesh-free, etc.)
2. Linearize the model about the undeformed state to extract the state space model matrices A, B, C, and C_m (D and D_m are zero). For these models, we assume no disturbances, therefore $E = 0$ and $F = -I$.
3. Test that (A, B) is stabilizable
4. Test that (A, C) is detectable
5. Test that rank $\begin{bmatrix} A & B \\ C & 0 \end{bmatrix} = n + p$

6. Find G_1 and G_2 such that $\dot{\boldsymbol{z}} = G_1 z + G_2 e$ incorporates an internal model of A_1.

7. Augment the system model, as in (3.38)

8. Find fixed gain matrix $K = \begin{bmatrix} K_1 & K_2 \end{bmatrix}$ such that $\tilde{A} + \tilde{B}K$ is stable

9. Test that (A, C_m) is detectable (if using measured output $\boldsymbol{y}_m$)

10. Design observer gain matrix L such that $(A - LC_m)$ is stable

11. Implement controller (3.39).

Chapter 4

Model Reduction and Reduced Order Control

Deformable object approximations for physical systems, such as finite element methods, have become increasingly popular and quite accurate in predicting the behaviour of physical systems. However, as the level of detail and accuracy of the models increase, so too does the dimensionality. This can lead to large computational expenses and lead to simulations that no can no longer be used for real-time control. In order to overcome the dimensionality problem, a technique know as model-reduction can be used. This involves creating a system of much lower dimension that still captures the important characteristics of the original systems response. Primarily, there are two methods of achieving this reduction: SVD-based and moment-matching based methods (Antoulas *et al.*, 2001).

The model reduction problem is outlined in Antoulas *et al.* (2001) and restated here for completion. The problem involves a full dynamical system model as defined

by

$$\Sigma : \quad \begin{cases} \dot{\boldsymbol{x}}(t) & = A\boldsymbol{x}(t) + B\boldsymbol{u}(t) \\ \boldsymbol{y}(t) & = C\boldsymbol{x} + D\boldsymbol{u}(t) \end{cases} \tag{4.1}$$

which we will abbreviate to

$$\Sigma = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \in \mathbb{R}^{(n+p)\times(n+m)}.$$

The object of model reduction is to find a much smaller order system

$$\hat{\Sigma} = \left[\begin{array}{c|c} \hat{A} & \hat{B} \\ \hline \hat{C} & \hat{D} \end{array}\right] \in \mathbb{R}^{(\hat{n}+p)\times(\hat{n}+m)}$$

with $\hat{n} < n$ that approximates the original system. The approximation must ensure a significantly small error in a region of interest, and that properties such as stability and passivity are preserved.

Recall that we are concerned only with the dynamical model of a deformable object which can be represented by a second-order differential equation,

$$M(\boldsymbol{d})\ddot{\boldsymbol{d}} + V(\dot{\boldsymbol{d}}, \boldsymbol{d})\dot{\boldsymbol{d}} + K(\boldsymbol{d})\boldsymbol{d} = \tilde{B}\boldsymbol{u}$$
$$\boldsymbol{y} = \tilde{L}^T\boldsymbol{d}$$

and in state-space with matrices,

$$A = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}V \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ M^{-1}\tilde{B} \end{bmatrix} \qquad C = \begin{bmatrix} \tilde{L}^T & 0 \end{bmatrix}$$

and the state

$$\boldsymbol{x}(t) = \begin{bmatrix} \boldsymbol{d}(t) \\ \dot{\boldsymbol{d}}(t) \end{bmatrix}.$$

Because of the second order structure of our system, and the state-space matrices have a natural partitioning within them, we follow a reduction method that preserves this structure. There are several papers that focus on reduction of this type or model such as Chahlaoui *et al.* (2005), Bai and Su (2005), Li and Bai (2005), Li and Bai (2006), and Lin *et al.* (2007). The common procedure involves using a moment-matching type reduction via an Arnoldi method to project onto a Krylov subspace. In this thesis, we use the algorithm reported in Li and Bai (2006), which is summarized below.

Let the second order system be arranged in a slightly different state-space form and include the desired output, dropping parentheses for clarity:

$$\begin{bmatrix} V & M \\ -W & 0 \end{bmatrix} \dot{\boldsymbol{x}} + \begin{bmatrix} K & 0 \\ 0 & W \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} \boldsymbol{u}$$

$$\boldsymbol{y} = \begin{bmatrix} \tilde{L}^T & 0 \end{bmatrix} \boldsymbol{x}$$

where $W \in \mathbb{R}^{N \times N}$ can be any nonsingular matrix (often set to either $-M$ for symmetry or I for simplicity). Let these matrices be defined as

$$C_R = \begin{bmatrix} V & M \\ -W & 0 \end{bmatrix} \quad G_R = \begin{bmatrix} K & 0 \\ 0 & W \end{bmatrix} \quad B_R = \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} \quad L_R^T = \begin{bmatrix} \tilde{L}^T & 0 \end{bmatrix}. \tag{4.2}$$

with $C_R \in \mathbb{R}^{n \times n}$, $G_R \in \mathbb{R}^{n \times n}$, $B_R \in \mathbb{R}^{n \times m}$, and $L_R^T \in \mathbb{R}^{p \times n}$. The subscript R is used to denote their use in the reduction algorithm and avoid any confusion between

similarly named matrices presented in this thesis.

The frequency domain transfer function for this MIMO system, Σ, in this form can be expressed

$$H(s) = L_R^T(G_R + sC_R)^{-1}B_R$$

The reduced transfer function of the system $\hat{\Sigma}$ sought is

$$\hat{H}(s) = \hat{L}_R^T(\hat{G}_R + s\hat{C}_R)^{-1}\hat{B}_R$$

where $\hat{C}_R \in \mathbb{R}^{\hat{n}\times\hat{n}}$, $\hat{G}_R \in \mathbb{R}^{\hat{n}\times\hat{n}}$, $\hat{B}_R \in \mathbb{R}^{\hat{n}\times m}$, and $\hat{L}_R^T \in \mathbb{R}^{p\times\hat{n}}$ The reduction is achieved by using projection matrices $X, Y \in \mathbb{R}^{n\times\hat{n}}$ such that

$$\hat{C}_R = Y^T C_R X, \qquad \hat{G}_R = Y^T G_R X, \qquad \hat{B}_R = Y^T B_R, \qquad \hat{L_R}^T = L_R X^T$$

Because of the second order nature of the system we can notice the matrices in (4.2) have a natural partition into blocks as in

$$C_R = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_1 & n_2 \end{array} \\ \begin{bmatrix} C_{R_{11}} & C_{R_{12}} \\ C_{R_{21}} & C_{R_{22}} \end{bmatrix} \end{array}, \qquad G_R = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} n_1 & n_2 \end{array} \\ \begin{bmatrix} G_{R_{11}} & G_{R_{12}} \\ G_{R_{21}} & G_{R_{22}} \end{bmatrix} \end{array},$$

$$B_R = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{c} m \end{array} \\ \begin{bmatrix} B_{R_1} \\ B_{R_2} \end{bmatrix} \end{array}, \qquad L_R = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{c} p \end{array} \\ \begin{bmatrix} L_{R_1} \\ L_{R_2} \end{bmatrix} \end{array}.$$

where $n_1 + n_2 = n$. The projection is performed by defining X and Y as

$$X = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} \hat{n}_1 & \hat{n}_2 \end{array} \\ \begin{bmatrix} X_1 & 0 \\ 0 & X_2 \end{bmatrix} \end{array}, \qquad Y = \begin{array}{c} \\ n_1 \\ n_2 \end{array} \begin{array}{c} \begin{array}{cc} \hat{n}_1 & \hat{n}_2 \end{array} \\ \begin{bmatrix} Y_1 & 0 \\ 0 & Y_2 \end{bmatrix} \end{array}$$

where $\hat{n}_1 + \hat{n}_2 = \hat{n}$, such that the submatrices of the reduced system are in fact the reduction of the submatrices. For example:

$$\begin{aligned} \hat{C}_R &= Y^T C_R X \\ &= \begin{bmatrix} Y_1^T & 0 \\ 0 & Y_2^T \end{bmatrix} \begin{bmatrix} C_{R_{11}} & C_{R_12} \\ C_{R_21} & C_{R_22} \end{bmatrix} \begin{bmatrix} X_1 & 0 \\ 0 & X_2^T \end{bmatrix} \\ &= \begin{bmatrix} Y_1^T C_{R_{11}} X_1 & Y_1^T C_{R_{12}} X_2 \\ Y_2^T C_{R_{21}} X_1 & Y_2^T C_{R_{22}} X_2 \end{bmatrix} \end{aligned}$$

and correspondingly, for all submatrices,

$$\hat{C}_{R_{ij}} = Y_i^T C_{R_{ij}} X_j, \qquad \hat{G}_{R_{ij}} = Y_i^T G_{R_{ij}} X_j, \qquad \hat{B}_{R_i} = Y_i^T B_{R_i}, \qquad \hat{L}_{R_j}^T = L_{R_j}^T X_j.$$

The X and Y projection matrices are found using an Arnoldi implementation as stated in Li and Bai (2006). First, consider a few important concepts from Li and Bai (2006).

The power series expansion of $H(s)$ at $s = 0$ is given by

$$H(s) = \sum_{i=0}^{\infty} (-1)^i s^i M_i \tag{4.3}$$

where M_i are the moments. The reduced model has k matched moments around $s = 0$ if $M_i = \hat{M}_i$ for $0 \leq i \leq k-1$, or equivalently, $H(s) = \hat{H}(s) + \mathcal{O}(s^k)$. Additionally, define the kth Krylov subspace as

$$\mathcal{K}_k(A, Z) \stackrel{\text{def}}{=} \text{span}(Z, AZ, \ldots, A^{k-1}Z). \tag{4.4}$$

As stated in Li and Bai (2006), if

$$\mathcal{K}_k(G_R^{-1}C_R, G_R^{-1}B_R) \subseteq \text{span}(X) \tag{4.5}$$

$$\mathcal{K}_r(G_R^{-T}C_R^T, G_R^{-T}L_R) \subseteq \text{span}(Y) \tag{4.6}$$

then $H(s) = \hat{H}(s) + \mathcal{O}(s^{k+r})$. Therefore, by finding X and Y which spans the Krylov subspaces (4.5) and (4.6), then we have found a reduced system with $k + r$ matched moments. Furthermore, if we define W in (4.2) as $-M$, then C_R and G_R are symmetric. This symmetry, as stated in Li and Bai (2006), allows us to find X according to the Krylov subspace

$$\mathcal{K}_k\left(G_R^{-1}C_R, G_R^{-1}(B_R\ L_R)\right). \subseteq \text{span}(X) \tag{4.7}$$

Then by setting $X = Y$ the reduction satisfies $H(s) = \hat{H}(s) + \mathcal{O}(s^{2k})$.

Finally, as first mentioned in Su and Craig (1991), a key aspect to retaining the second order nature of the system comes from the structure of the matrix $G_R^{-1}C_R$ as seen in the MIMO transfer function in the general form

$$H(s) = L_R^T(I + sG_R^{-1}C_R)^{-1}G_R^{-1}B_R.$$

and from our desired Krylov subspace projection

$$\mathcal{K}_k(G_R^{-1}C_R, G_R^{-1}B_R).$$

Because the block structure of

$$G_R^{-1}C_R = \begin{bmatrix} K^{-1}V & K^{-1}M \\ -I & 0 \end{bmatrix}$$

contains lower left block of the form αI and lower right block of zero, it can be proven [Su and Craig (1991), Li and Bai (2005)] that

$$\text{span}(X_2) \subseteq \text{span}(X_1).$$

This allows X and Y to be defined using only X_1 as

$$X = \begin{array}{c} \\ N_1 \\ N_1 \end{array} \begin{array}{c} \begin{array}{cc} \hat{N}_1 & \hat{N}_1 \end{array} \\ \begin{bmatrix} X_1 & 0 \\ 0 & X_1 \end{bmatrix} \end{array}, \qquad Y = \begin{array}{c} \\ N_1 \\ N_1 \end{array} \begin{array}{c} \begin{array}{cc} \hat{N}_1 & \hat{N}_1 \end{array} \\ \begin{bmatrix} X_1 & 0 \\ 0 & X_1 \end{bmatrix} \end{array}$$

in order to keep the second order structure while performing the moment-matching reduction.

Using these facts, the full Arnoldi-based reduction can be outlined in Algorithm 4.1. In this implementation for second-order structure-preserving reduction, there is only control of two design parameters: s_0 – the frequency around which the moments are matched, and k – the order of moments matched.

For more detail and proofs of the Krylov subspace projection method of model

Algorithm 4.1 Arnoldi-type reduction algorithm:

Set $\hat{G}_R = G_R + s_0 C$;
Solve $\hat{G}_R \hat{Q} = B_R$ for $\hat{Q}$
for $j = 1$ to $k - 1$ **do**
  Solve $\hat{G}_R \hat{Q} = C_R Q_i$ for $\hat{Q}$
  for $i = 1$ to j **do**
    $\hat{Q} = \hat{Q} - Q_i (Q_i^T \hat{Q})$
  end for
  $Q_{j+1} = \text{orth}(\hat{Q})$
end for
Partition $\tilde{X} = (Q_1 \quad Q_2 \quad \dots \quad Q_k)$ as $\tilde{X} = \begin{bmatrix} \tilde{X}_1 \\ \tilde{X}_2 \end{bmatrix}$
$X_1 = \text{orth}(\tilde{X}_1)$
Set $X = \begin{bmatrix} X_1 & 0 \\ 0 & X_1 \end{bmatrix}$ and $Y = \begin{bmatrix} X_1 & 0 \\ 0 & X_1 \end{bmatrix}$
Compute $\hat{C}_R = Y^T C_R X$, $\hat{G}_R = Y^T G_R X$, $\hat{B}_R = Y^T B_R$, $\hat{L}_R = X^T L_R$

reduction used in this thesis see Li and Bai (2005) and Li and Bai (2006).

Chapter 5

Simulation and Experimental Setup

This chapter describes the control law formulation more specifically as it applies to our deformable object models using the theory outlined in Chapter 3 and Chapter 4. Additionally, the simulations and experimental setup are described in detail.

The use of robust output regulation control theory on deformable is validated through both simulations performed in MATLAB and experiments using robot manipulators and liquid plastic deformable objects. This section outlines the control steps used for accurate robotic manipulation of deformable objects in simulations and experiments. For all simulations and experiments presented in this thesis we use only planar deformable objects. The deformable object simulations were designed to mimic their experimental object counterpart part so that the same control law derived in the simulation would be used to drive the robot manipulator in the experiments. The methods and details of the simulation are outlined in Section 5.1 and the equipment and experimental setup in Section 5.2.

5.1 Simulation Setup

This section outlines the control design procedure as it applies to the deformable objects used in both simulations and experiments. The same control scheme is used in the experimental setup.

Deformable objects were modeled using a meshfree continuum model known as the reproducing kernel particle method (RKPM) based on the findings of Chen *et al.* (1996). These models are implemented in C++ and based on the stress-strain characteristics of a soft plastic object, planar object that will be used in the experimental tests. The characteristics are obtained from recording deformation and force characteristics using a camera and robot manipulator. Then the parameters of the model are optimized to match the deformation that occurs under load in the real object. The details of determining the object parameters are shown in Appendix A. It should be noted that the RKPM model was implemented as a quasi-static solver. Therefore in simulations the control forces may result in large initial deformations of the nonlinear model since there are no viscous or inertial forces acting as resistance. Some overshoot to appear in the simulations that may not be as evident in the experiments, where inertial and viscous effects are present.

All simulations were conducted using MATLAB's Simulink® software to synthesize the control signals and apply them to the nonlinear deformable object. The C++ object model is compiled into a Window's dynamic link library (DLL) to allow interfacing with MATLAB and Simulink. Additionally, the RKPM implementation uses a CUDA GPU to accelerate the DLL's ability to solve the RKPM model with forces input from the Simulink controller.

5.1.1 Defining the Linearized System

The first step in the controller design is to extract the useful information from our deformable object model. Recall that the dynamic object model with output is given by

$$\begin{aligned} M(\boldsymbol{d})\ddot{\boldsymbol{d}} + V(\dot{\boldsymbol{d}}, \boldsymbol{d})\dot{\boldsymbol{d}} + K(\boldsymbol{d})\boldsymbol{d} &= \boldsymbol{f} \\ \boldsymbol{y} &= \tilde{L}^T \boldsymbol{d} \end{aligned} \tag{5.1}$$

where the state-space form desired is

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} \\ \boldsymbol{y} &= C\boldsymbol{x} \end{aligned} \tag{5.2}$$

with

$$A = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}V \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ M^{-1}\tilde{B} \end{bmatrix} \qquad C = \begin{bmatrix} \tilde{L}^T & 0 \end{bmatrix} \tag{5.3}$$

and state

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{d} \\ \dot{\boldsymbol{d}} \end{bmatrix}. \tag{5.4}$$

Compute the mass, damping, and stiffness matrices from the model, linearized by evaluating them and the equilibrium point where $\boldsymbol{d}$ and $\boldsymbol{f}$ are zero.

The influence matrix $\tilde{B}$ matrix needs to be created from knowledge of how external particles will distribute over the object nodes. This will vary depending on

the modeling approach. As a simple illustration, consider the mass-spring-damper model with four nodes as shown in Figure 5.1. For this simple planar model we have node number $N = 4$ and space-dimension $k = 2$. Define the displacement vector $\boldsymbol{d} \in \mathbb{R}^8$ such that the entries represent the x- and y-displacements of each node starting with the top left and reading left to right. Two planar (two dimensional) forces act on the two rightmost nodes which correspond to elements $3, 4$ and $7, 8$ of our $\boldsymbol{d}$ vector. The external force vector would be $\boldsymbol{f} = \begin{bmatrix} 0 & 0 & F_{x1} & F_{y1} & 0 & 0 & F_{x2} & F_{y2} \end{bmatrix}^T$. Decomposing this simple into an influence matrix and input vector would give unity entries:

$$\begin{bmatrix} 0 \\ 0 \\ F_{1x} \\ F_{1y} \\ 0 \\ 0 \\ F_{2x} \\ F_{2y} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} F_{1x} \\ F_{1y} \\ F_{2x} \\ F_{2y} \end{bmatrix}. \tag{5.5}$$

Note again, that more complex influence matrices can be used to apply more distributed forces, depending on the model.

The selector matrix $\tilde{L}^T$ is defined in a similar fashion by selecting which nodal displacements represent the positions of nodes which we wish to regulate. In our simple example, if we wish to use the two forces to control the position of the lower left node, the positions would be entries 5, and 6 in $\boldsymbol{d}$. The selector matrix would

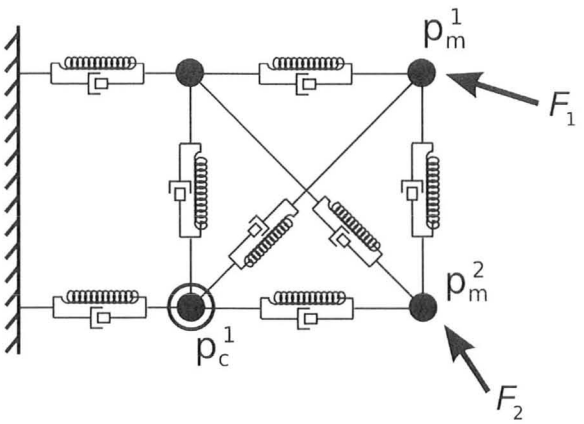


Figure 5.1: Simple 4 node MSD deformable object for illustration

simply be

$$\tilde{L}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0. \end{bmatrix}.$$

For the linear state-space model, simply substitute M, V, K, $\tilde{B}$, and $\tilde{L}^T$ into (5.3).

Recall, for these models, that the D feedthrough matrix is zero. Additionally, as explained in Chapter 3, we may have access to more states than the ones to be regulated. In this case we also have the measurable output vector $\boldsymbol{y}_m = C_m \boldsymbol{x}$. The matrix $C_m \in \mathbb{R}^{p_m \times n}$ is constructed in the same manner as C, using a corresponding selector matrix $\tilde{L}_m^T$ to indicate all the nodal deformations for which we have measurement access. The feedthrough matrix in this case, D_m, is also zero.

To remain consistent with the formulations of Chapter 3, we assume that the disturbances on the plant do not occur from any know exogenous signals and therefore $E = 0$. Also, are reference signals, $\boldsymbol{r}(t)$ are what we want the output $\boldsymbol{y}$ to exactly match, therefore $F = -I$. Finally, the exogenous reference signals are assumed to be set-point signals with no dynamical model, giving $A_1 = 0$, as well.

These matrices fully defined our linearized state space system with regulated outputs, and measured outputs, and error signal:

$$\begin{aligned} \dot{\boldsymbol{x}} &= A\boldsymbol{x} + B\boldsymbol{u} \\ \boldsymbol{y} &= C\boldsymbol{x} \\ \boldsymbol{y}_m &= C_m\boldsymbol{x} \\ \boldsymbol{e} &= C\boldsymbol{x} + F\boldsymbol{r}. \end{aligned} \tag{5.6}$$

5.1.2 Reducing the Simulation Model

For each deformable model, the control law was applied to both the original full state model, and the reduced state model. For cases when the model is reduced, the procedure as outlined in Chapter 4 is followed, summarized briefly below for our model.

Using the same mass, spring, and damping matrices, we form the a slightly varied version of the state space equations with the matrices

$$\begin{aligned} C_R\dot{\boldsymbol{x}} + G_R\boldsymbol{x} &= B_R\boldsymbol{u} \\ \boldsymbol{y} &= {L_R}^T\boldsymbol{x} \\ \boldsymbol{y}_m &= {L_{mR}}^T\boldsymbol{x} \end{aligned}$$

with

$$C_R = \begin{bmatrix} V & M \\ -W & 0 \end{bmatrix} \quad G_R = \begin{bmatrix} K & 0 \\ 0 & W \end{bmatrix} \quad B_R = \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix} \quad L_R^T = C \quad L_{mR}^T = C_m$$

We use the subscript R to differentiate these matrices from any other similarly named matrices already defined. Following Algorithm 4.1, we find the projection matrices X and Y. For our problem, we reduced around the expansion point $s_0 = 0$ in the algorithm because we wish to moment match around the origin of $H(s)$. That is, we desire to keep the low frequency moments of the MIMO transfer function. The reduced state space matrices are

$$\hat{C}_R = Y^T C_R X, \quad \hat{G}_R = Y^T G_R X, \quad \hat{B}_R = Y^T B_R, \quad \hat{L}_R^T = L_R^T X^T \quad \hat{L}_{mR}^T = L_{mR}^T X^T$$

The reduced state space form is recovered by

$$\begin{aligned} A &= -\hat{C}_R^{-1}\hat{G}_R & B &= \hat{C}_R^{-1}\hat{B}_R \\ C &= \hat{L}_R^T & D &= 0 \\ C_m &= \hat{L}_{mR}^T & D_m &= 0 \end{aligned}$$

For the different deformable objects tested, the reduced system is compared to the original system to illustrate that the low frequency properties were retained.

5.1.3 Control Design

Once the state space matrices have been obtained, we need to ensure the object model in combination with the manipulation points, control points, and any boundary conditions are defined such that a robust regulating controller exists. This involves ensuring the conditions outlined in section 3.2.1 are met. First, we ensure our matrix pair (A, B) is stabilizable. Without this condition, the exponential stability of the closed-loop system cannot be guaranteed by simple state feedback. For the models and

configurations presented in the results of this thesis, the system is in fact controllable, allowing more flexibility in the design of the stabilizing matrix gains. Next, we need to be sure our system is detectable. For this we check the matrix pair (C_m, A) since we are relying on the feedback of our measured output $\boldsymbol{y}_m$ and not just the regulated output $\boldsymbol{y}$. However, since we assume measurement feedback, but have a controller based on error, we need to make sure that the error is *readable* from $\boldsymbol{y}_m$. That is, there exists a matrix T such that $C = TC_m$. Our formulations guarantee this since the regulated outputs $\boldsymbol{y}$ are in fact a subset of the measured outputs $\boldsymbol{y}_m$. Together with the detectability of (C_m, A), this essentially ensures the detectability of (C, A).

For our simulations, these conditions are checked using the well known PBH tests for stabilizability defined as

$$\text{rank} \begin{bmatrix} A - \lambda I & B \end{bmatrix} = n$$

for all eigenvalues λ of A, and for detectability defined as

$$\text{rank} \begin{bmatrix} C_m \\ A - \lambda I \end{bmatrix} = n$$

for all eigenvalues λ of A, where n is the dimension of the state vector $\boldsymbol{x}$. Finally, we ensure that a solution to the regulator equations exists with the condition

$$\text{rank} \begin{bmatrix} A & B \\ C & 0 \end{bmatrix} = n + p.$$

For the simulations presented in this thesis, all criteria to ensure a solution are

met. The next step is to augment the plant system with an internal model variable $\boldsymbol{z}$. Defining the internal model for the exogenous signals is simple in the case of set-point trajectories. It can be achieved by defining

$$\dot{\boldsymbol{z}} = G_1\boldsymbol{z} + G_2\boldsymbol{e} \tag{5.7}$$

with $G_1 = 0_{p\times p}$ and $G_2 = I_{p\times p}$. This is effectively incorporating an integral control scheme for set-point reference signals. The augmented system is then given by

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{z}} \end{bmatrix} = \begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{z} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \boldsymbol{u} + \begin{bmatrix} 0 \\ -I \end{bmatrix} \boldsymbol{r}$$

with augmented state

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{z} \end{bmatrix} \tag{5.8}$$

The problem now remains twofold: determine the gain K_1 nd K_2 such that $\boldsymbol{u} = K_1\boldsymbol{x} + K_2\boldsymbol{z}$ stabilizes the augmented system, and design the estimator system for $\hat{\boldsymbol{x}}$ so that our actual controller law is $\boldsymbol{u} = K_1\hat{\boldsymbol{x}} + K_2\boldsymbol{z}$. We approach these problems with the well-known linear-quadratic gaussian (LQG) control philosophy.

First, given the augmented system, we find the optimal stabilizing gain matrix K from LQR design. That is, we wish to find the controller that stabilizes the system and minimizes the cost function

$$J = \int_0^\infty (\tilde{\boldsymbol{x}}^T Q\tilde{\boldsymbol{x}} + \boldsymbol{u}^T R\boldsymbol{u} + 2\tilde{\boldsymbol{x}}^T N\boldsymbol{u})dt \tag{5.9}$$

under the state feedback form

$$\boldsymbol{u} = K\tilde{\boldsymbol{x}} = \begin{bmatrix} K_1 & K_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{z} \end{bmatrix}. \tag{5.10}$$

Q can be interpreted as a penalty on the size of the states $\tilde{x}$, R a penalty on control signal size, and N penalizing a combination of the two.

However, since we have a desired set-point output $\boldsymbol{y}$ it makes more sense to penalize the desired output from deviating from its desired value and the control signal from deviating from its steady-state value. Recall from Section 3.2.1 that these steady state values are given from the solution to

$$\begin{bmatrix} A & B \\ C & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^* \\ \boldsymbol{u}^* \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{r} \end{bmatrix} \tag{5.11}$$

and from the solution to our regulator equations, we have $\boldsymbol{x}^* = X\boldsymbol{r}$ and $\boldsymbol{u}^* = U\boldsymbol{r}$. If the system has equal number of inputs and outputs (i.e. $m = p$), and condition 3.3 is met, then we can solve

$$\begin{bmatrix} A & B \\ C & 0 \end{bmatrix}^{-1}. \tag{5.12}$$

The matrix X corresponds to the upper-rightmost $n \times p$ submatrix, and U the lower-rightmost $m \times p$ submatrix. Alternatively, we could solve for these matrices from block matrix inversion formulas which gives

$$X = A^{-1}BS_A^{-1} \tag{5.13}$$

$$U = S_A^{-1} \tag{5.14}$$

where $S_A = D - CA^{-1}B$ is the Schur complement of A.

If there are more inputs than outputs ($m > p$), then we do not have a unique solution. X and U are best solved from the Moore-Penrose pseudoinverse of the linear system. In MATLAB, this is achieved using the `pinv` function. Note that this results in the minimum 2-norm solution for $\boldsymbol{x}^*$ and $\boldsymbol{u}^*$ to the linear system.

Mathematically, the cost function we want to minimize is

$$J = \int_0^\infty \{\alpha||\boldsymbol{y} - \boldsymbol{r}||_2 + \beta||\boldsymbol{z}||_2 + \gamma||\boldsymbol{u} - \boldsymbol{u}^*||_2\}\, dt. \tag{5.15}$$

Modifying the cost function to instead minimize the square of the 2-norm of these errors (which does not change the solution of the cost function), we can manipulate to attain a proper LQR cost function as follows:

$$\begin{aligned}
J &= \int_0^\infty \left\{\alpha||\boldsymbol{y} - \boldsymbol{r}||_2^2 + \beta||\boldsymbol{z}||_2^2 + \gamma||\boldsymbol{u} - \boldsymbol{u}^*||_2^2\right\} dt \\
&= \int_0^\infty \left\{\alpha(\boldsymbol{y} - \boldsymbol{r})^T(\boldsymbol{y} - \boldsymbol{r}) + \beta\boldsymbol{z}^T\boldsymbol{z} + \gamma(\boldsymbol{u} - \boldsymbol{u}^*)^T(\boldsymbol{u} - \boldsymbol{u}^*)\right\} dt \\
&= \int_0^\infty \left\{\alpha(\boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{y}^T\boldsymbol{r} + \boldsymbol{r}^T\boldsymbol{r}) + \beta\boldsymbol{z}^T\boldsymbol{z} + \gamma(\boldsymbol{u} - \boldsymbol{u}^*)^T(\boldsymbol{u} - \boldsymbol{u}^*)\right\} dt \\
&= \int_0^\infty \left\{\alpha(\boldsymbol{x}^T C^T C\boldsymbol{x} - 2\boldsymbol{x}^T C^T C\boldsymbol{x}^* + \boldsymbol{x}^{*T} C^T C\boldsymbol{x}^*) + \beta\boldsymbol{z}^T\boldsymbol{z} + \gamma(\boldsymbol{u} - \boldsymbol{u}^*)^T(\boldsymbol{u} - \boldsymbol{u}^*)\right\} dt \\
&= \int_0^\infty \left\{(\boldsymbol{x} - \boldsymbol{x}^*)^T\alpha C^T C(\boldsymbol{x} - \boldsymbol{x}^*) + \boldsymbol{z}^T\beta I\boldsymbol{z} + (\boldsymbol{u} - \boldsymbol{u}^*)^T\gamma I(\boldsymbol{u} - \boldsymbol{u}^*)\right\} dt \\
&= \int_0^\infty \left\{ \begin{bmatrix} (\boldsymbol{x} - \boldsymbol{x}^*)^T & \boldsymbol{z}^T \end{bmatrix} \begin{bmatrix} \alpha C^T C & 0 \\ 0 & \beta I \end{bmatrix} \begin{bmatrix} (\boldsymbol{x} - \boldsymbol{x}^*) \\ \boldsymbol{z} \end{bmatrix} + (\boldsymbol{u} - \boldsymbol{u}^*)^T\gamma I(\boldsymbol{u} - \boldsymbol{u}^*) \right\} dt \\
&= \int_0^\infty \left\{ (\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}^*)^T \begin{bmatrix} \alpha C^T C & 0 \\ 0 & \beta I \end{bmatrix} (\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}^*) + (\boldsymbol{u} - \boldsymbol{u}^*)^T\gamma I(\boldsymbol{u} - \boldsymbol{u}^*) \right\} dt
\end{aligned}$$

which corresponds to an LQR problem with the weighting matrices

$$Q = \begin{bmatrix} \alpha C^T C & 0 \\ 0 & \beta I \end{bmatrix} \qquad R = \gamma I \qquad N = 0.$$

The α, β, and γ, parameters provide a convenient way to tweak the resulting gain matrix and alter the transient response characteristics of the output while still maintaining stability and robustness. Note that although they presented as scalars here, α, β and γ could be defined as weighting matrices to affect the difference components of $\boldsymbol{y}$, $\boldsymbol{z}$ and $\boldsymbol{u}$ individually.

Note also that the vectors in our cost function are $\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}^*$ and $\boldsymbol{u} - \boldsymbol{u}^*$, as opposed to simply $\boldsymbol{x}$ and $\boldsymbol{u}$. This means that our control signal over which we minimize must be of the form

$$\boldsymbol{u} - \boldsymbol{u}^* = K(\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}^*) \tag{5.16}$$

or more properly

$$\begin{aligned} \boldsymbol{u} &= K(\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}^*) + \boldsymbol{u}^* \\ \boldsymbol{u} &= K_1(\boldsymbol{x} - \boldsymbol{x}^*) + K_2\boldsymbol{z} + \boldsymbol{u}^* \\ \boldsymbol{u} &= K_1(\boldsymbol{x} - X\boldsymbol{r}) + K_2\boldsymbol{z} + U\boldsymbol{r} \\ \boldsymbol{u} &= K_1\boldsymbol{x} + K_2\boldsymbol{z} + (U - K_1X)\boldsymbol{r}. \end{aligned} \tag{5.17}$$

Let the third gain matrix in (5.17) be $K_3 = U - K_1X$. Notice that this is exactly the feedforward mapping of the reference signal $\boldsymbol{r}$ described in Chapter 3.

Upon substitution into are state space formulation, we see that using $\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}^*$ and $\boldsymbol{u} - \boldsymbol{u}^*$ with the optimal gain matrix K will still stabilize the augmented system just

as (5.10) would:

$$\begin{aligned}\begin{bmatrix}\dot{\boldsymbol{x}}\\ \dot{\boldsymbol{z}}\end{bmatrix} &= \begin{bmatrix}A & 0\\ C & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{z}\end{bmatrix} + \begin{bmatrix}B\\ 0\end{bmatrix}\left(\begin{bmatrix}K1 & K2\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{z}\end{bmatrix} + (U - K_1X)\boldsymbol{r}\right)\\ &= \begin{bmatrix}A + BK_1 & BK_2\\ C & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{z}\end{bmatrix} + \begin{bmatrix}B(U - K_1X)\\ 0\end{bmatrix}\boldsymbol{r}\\ &= \tilde{A}_c\tilde{\boldsymbol{x}} + \tilde{B}_c\boldsymbol{r}.\end{aligned}$$

The closed loop state matrix, $\tilde{A}_c$ is exactly the same as it would be under control law (5.10). Thus we can conclude that since, by definition, the LQR optimal gain K stabilizes the augmented system under (5.10), the same K will stabilize the system under (5.17).

The next part of LQG controller design is creating the state estimation $\hat{\boldsymbol{x}}$. Given our state space system, we construct the estimation from

$$\dot{\hat{\boldsymbol{x}}} = (A - LC_m)\hat{\boldsymbol{x}} + B\boldsymbol{u} + L\boldsymbol{y_m} \tag{5.18}$$

where the corrective gain matrix L is chosen such that $A - LC_m$ is stable. Often, a good approach to designing L is to approximate the process and measurement white noise characteristics that affect our states and output, respectively, and minimize their effect on the state-estimate. The optimal gain matrix L in the presence of this noise results in the well-known *Kalman-Bucy* estimator.

For our simulations and experiments we assume the process noise $\boldsymbol{n}_u$ affects the system control input, with a variance of $\mathrm{E}(\boldsymbol{n}_u\boldsymbol{n}_u^T)$. Similarly, the measurement noise, $\boldsymbol{n}_{y_m}$ is added to the output $\boldsymbol{y}_m$ with a variance $\mathrm{E}(\boldsymbol{n}_{y_m}\boldsymbol{n}_{y_m}^T)$. The optimal estimator

gain L is obtained using MATLAB's `kalman` command.

From this LQG approach, we have matrices K and L matrices that stabilize $\tilde{A} + \tilde{B}K$ and $A - LC_m$, respectively. The entire control can now be described as

$$\begin{aligned} \dot{\hat{\boldsymbol{x}}} &= (A - LC_m)\hat{\boldsymbol{x}} + B\boldsymbol{u} + L\boldsymbol{y}_m \\ \dot{\boldsymbol{z}} &= \boldsymbol{y} - \boldsymbol{r} \\ \boldsymbol{u} &= K_1\hat{\boldsymbol{x}} + K_2\boldsymbol{z} + K_3\boldsymbol{r}. \end{aligned} \tag{5.19}$$

The topography of this control scheme depicted in Figure 5.2.

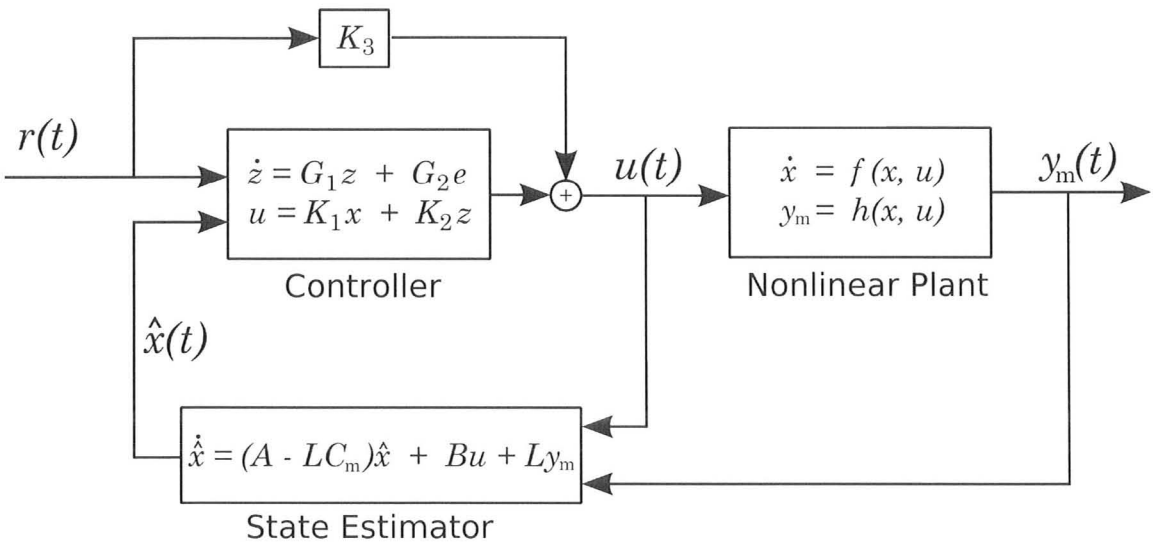


Figure 5.2: Block Diagram representing simulation control loop.

5.1.4 The Feedforward gain, K_3

It is worth some discussion on the feedforward gain K_3. This term is not necessary to ensure robust regulation for the special case of set-point regulation. This is essentially because the constant nature of the reference $\boldsymbol{r}$ means it can always be treated as zero

under an appropriate change of coordinates. Since this term maps the reference signal to our invariant subspace, a linear mapping of zero will still be zero. However, our initial conditions, under the new coordinates, imply that we are not on the invariant subspace, i.e. $\boldsymbol{u}(0) \neq U\boldsymbol{r}(0)$ and the integral term will force our control signal to the mapping such that of $\lim_{t\to\infty} \boldsymbol{u}(t) = U\boldsymbol{r}(t)$. Since for set-point regulation we do, in fact, know the initial conditions of $\boldsymbol{u}(0)$ and $\boldsymbol{r}(0)$, as well as the entire trajectory of $\boldsymbol{r}(t)$, using the feedforward matrix K_3 will allow a faster convergence to the steady state values of $\boldsymbol{u}$ and $\boldsymbol{x}$ with the integration term only accounting for uncertainties and disturbances of the system.

Additionally, for the case when the number of inputs equals the number of outputs, $(m = p)$, the use of K_3 will not affect the final values of $\boldsymbol{u}$. Since there is only one, unique solution, the input must converge to the single solution to achieve regulation. However, in the case where there are more inputs than outputs the results may vary depending on settings of the LQR parameters and the solution obtained from (5.11). For example, if we find the min-norm solution from the pseudoinverse, we find the values of $\boldsymbol{u}^*$ and $\boldsymbol{x}^*$ with the minimum 2-norms which solve the *linear* system. Since the actual system is not linear and there is likely uncertainty present, these values will not be the actual steady-state values, and the integral term will cause convergence to the final values by minimizing the deviation of $\boldsymbol{y}$ and $\boldsymbol{u}$ from $C\boldsymbol{x}^*$ and $\boldsymbol{u}^*$, respectively, as weighted by the LQR cost function parameters. If the feedforward term K_3 is omitted, then the final input values achieved are the ones that minimizing the the deviation of $\boldsymbol{y}$ and $\boldsymbol{u}$ from *zero*, according to the LQR weighting parameters.

5.2 Experimental Setup

In addition to applying an the control law on a nonlinear computer model in simulations, experimental test were done using robot manipulators and a physical deformable object. The object is constructed from soft liquid plastic solution. The solution is mixed with either plastic softeners or hardeners to achieve varying stiffness characteristics.

The control law to determine the forces applied to the object is the same as described in the simulation setup of Section 5.1. However, the overview of system changes slightly in the sense that we are no longer applying a force control signal directly to the manipulation points of the deformable object as in the simulation and no longer calculating the measured output directly from an object model. Instead, the control signal is passed to the robot controller and the robot manipulator applies the forces to the object. For the output signals, we monitor the manipulation point displacement from the coordinates of the robot end-effector. Additionally, the control point displacement is measured using a stereoscopic camera and an infrared LED marker placed at the desired location on the object. The general overview of the system is shown in Figure 5.3. The physical setup is seen in Figure 5.4 showing a robot manipulator, stereoscopic camera, and planar object phantom with a single control point and single manipulation point.

Although we monitor the position of the control point here directly using a stereo vision camera, the method of determining the output position can vary. More generally, we need the error to be readable through whatever measured output, $\boldsymbol{y}_m$, is available. In a medical setting, this may entail monitoring a target in tissue by some imaging modality like ultrasound guidance. The position of the manipulation points

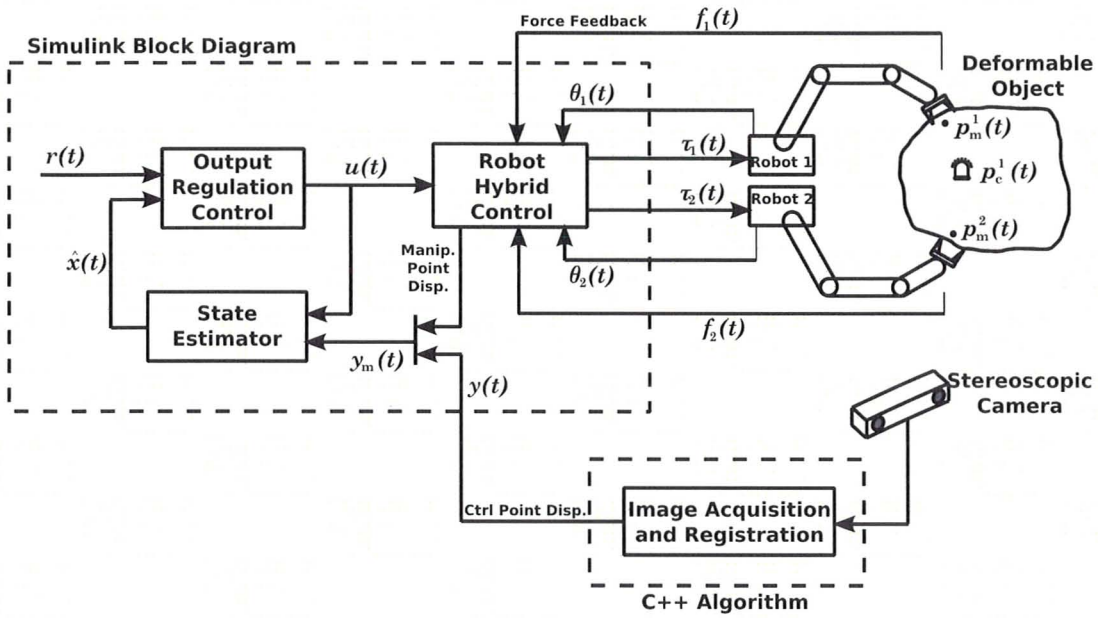


Figure 5.3: System topography representing the experimental setup.

Figure 5.4: Physical experimental setup with robot, camera and planar phantom.

are another natural addition for plant output since most robotic manipulators have an accurate feedback of the joint angles and thus the end-effector location via forward kinematics. Since the controller is based on the feedback of an estimation of the states of the plant model, any additional output fed into the estimator can help improve the accuracy of the model.

The setup comprises two CRS Catalyst-5 five degree of freedom (DOF) robots from Thermo Scientific. They are controlled using MATLAB's Simulink interfaced to the robot controllers using QuaRC® by Quanser. The robots are also equipped with ATI's Gamma® sensors at the end-effectors to provide force and torque values in six dimensions. The stereoscopic camera used is the Bumblebee2® from Point Grey Research. The 3D position of the control point is calculated from the stereo images using a C++ program and sent to Simulink real-time using UDP protocol.

The use of this equipment in the experimental setup requires two important aspects to be integrated with the output regulation controller:

- camera to robot registration for control point feedback
- hybrid position/force control to apply the forces in object plane

5.2.1 Camera Registration and Control Point Feedback

Image registration is required because the 3D position of the control point is captured in the cameras frame of reference and needs to be converted to a point in the robot's coordinate system. We accomplish this with a fairly simple three point registration algorithm. Three points, represented with infrared LED markers, are placed in a known location in the robots workspace. These points are captured in the initialization of the camera and thus known in the camera coordinate system as

well. Using these three points we can construct the rotation matrix that converts the camera coordinate frame the the robot coordinate frame. Any displacement of the control point recorded in the camera coordinate system is converted with this rotation to a displacement in the robot coordinate system. For these experiments, the object is placed in the robot workspace such that the x- and y-directions agree with the robots coordinate system, thus a further registration between object and robot is not necessary. Further details of the image registration algorithm are given in Appendix B.

5.2.2 Robot Hybrid Position/Force Control

The hybrid position/force control is required so that we can command the robot with desired position for the z-coordinate as well as pitch and roll angles, while issuing force commands in the x,y-plane. The general control scheme is illustrated in Figure 5.5. The idea is to break the tasks of position and force control into two orthogonal subspaces, determine the joint level torques required to perform each, and sum them to create the total manipulator input.

From this diagram, we see that the joint angles of the robot are given by an $n \times 1$ vector $\boldsymbol{\theta}_a$ where n is the number of joints. The torques controlling the joints of the robot are given in the $n \times 1$ vector $\boldsymbol{\tau}$. The joint angles of the robot can be solved, via forward kinematics, for the 6×1 vector $\boldsymbol{x}_a$ which represents the full Cartesian position and orientation of the end-effector. Similarly, a sensor at the end-effector gives a 6×1 vector measuring the Cartesian force and torque acting on the end-effector, $\boldsymbol{f}_a$.

The error describing the difference between desired and actual Cartesian position/orientation is $\boldsymbol{x}_e = \boldsymbol{x}_d - \boldsymbol{x}_a$. The error describing the error between the desired

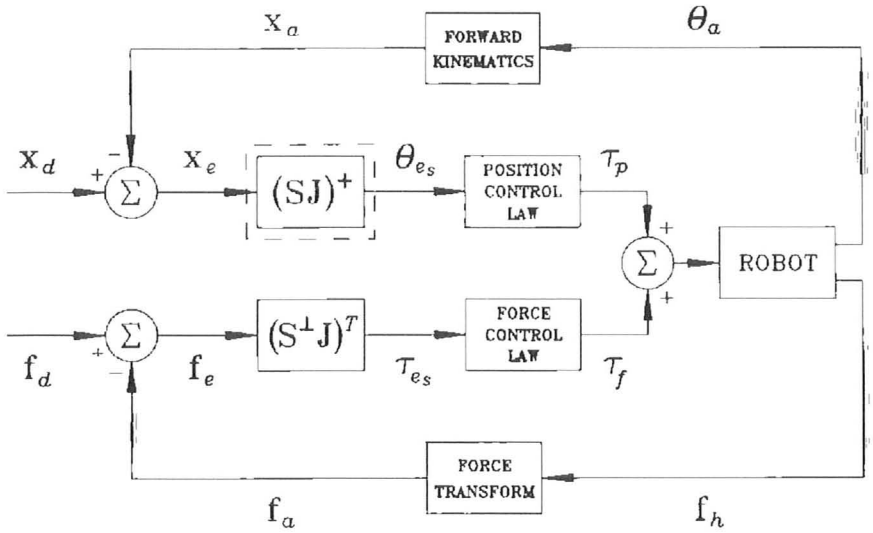


Figure 5.5: Block diagram illustrating the control law used for hybrid position/force control of the robot manipulator (Fisher and Mujtaba, 1992).

and actual Cartesian force/torque is $\boldsymbol{f}_e = \boldsymbol{f}_d - \boldsymbol{f}_a$.

In order to achieve hybrid control, we simply select the subspace in which we want to achieve a desired force/torque, and the orthogonal subspace remains in position control. This is accomplished with the projection matrices S and $S^\perp$. For our experimental setup, we use 5-DOF robots with the desired subspace for force control being the robot's x,y-plane. Therefore, the projection matrices we use are simple 5×6 selector matrices given by

$$S = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad S^\perp = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

5.3 Position Control Law

The selected Cartesian position/orientation error is given by

$$\boldsymbol{x}_{e_S} = S\boldsymbol{x}_e \tag{5.20}$$

The relationship between small Cartesian errors and joint-level errors is given by

$$\boldsymbol{x}_e = J\boldsymbol{\theta}_e \tag{5.21}$$

where the Jacobian matrix J represents a first order approximation of the relationship between differential joint-level motions and Cartesian space motions. Therefore, our selected subspace errors are related to the joint-level errors by

$$S\boldsymbol{x}_e = \boldsymbol{x}_{e_S} = SJ\boldsymbol{\theta}_e \tag{5.22}$$

As shown in Fisher and Mujtaba (1992), the matrix SJ is not invertible but rather we solve for one of the infinite number of solutions, $\boldsymbol{\theta}_{e_S}$, from

$$\boldsymbol{\theta}_{e_S} = (SJ)^+ \boldsymbol{x}_{e_S}. \tag{5.23}$$

Here, $(SJ)^+$ is the pseudoinverse of (SJ) given by

$$(SJ)^+ = (SJ)^T \left[(SJ)(SJ)^T\right]^{-1} \tag{5.24}$$

and is often called the right pseudoinverse. This corresponds to finding the minimum two-norm solution of $\boldsymbol{\theta}_e$ that satisfies (5.22). It should be noted that there are

other solutions for the joint level error that can be found from adding vectors in the orthogonal complement space to $(SJ)^+$ as given by the general solution

$$\boldsymbol{\theta}_{e_G} = (SJ)^+ \boldsymbol{x}_{e_S} + \left[I - (SJ)^+(SJ)\right] \boldsymbol{z} \tag{5.25}$$

However, there are conditions that must be satisfied in order to remain kinematically stable, and the minimum norm solution is the simplest that meets these constraints. For more details the reader is referred to Fisher and Mujtaba (1992).

Once having solved for the joint level error $\boldsymbol{\theta}_{e_S}$, we can apply a position control law to determine the robot joint torques, $\boldsymbol{\tau}_P$. In our experiments, we use a simple joint level PD control law.

5.4 Force Control Law

Similar to the position control, the Cartesian force/torque subspace of interest is given by

$$\boldsymbol{f}_{e_S} = S^{\perp} \boldsymbol{f}_e. \tag{5.26}$$

The relationship between Cartesian force/torque measurements and joint level torques is given by

$$\boldsymbol{\tau} = J^T \boldsymbol{f}. \tag{5.27}$$

Therefore, the joint level torques errors are given by

$$\begin{aligned} \boldsymbol{\tau}_{e_S} &= J^T S^{\perp} \boldsymbol{f}_e \\ &= (S^{\perp} J)^T \end{aligned} \tag{5.28}$$

With the joint-level torque errors, we use a simple PI Control Law to calculate the joint control torques $\boldsymbol{\tau}_F$ from the error, $\boldsymbol{\tau}_{e_S}$.

The control torques determined from the position and force control laws can be summed, as depicted in Figure 5.5, and applied to the robot to achieve the final hybrid control scheme.

The experimental Simulink implementation of this hybrid control law and integration with the output regulation controller and camera feedback system can be found in Appendix C.

Chapter 6

Simulation and Experimental Results

This section highlights some of the simulation and experimental results that were obtained using the control design simulation and experimental setups described in Chapter 5. First, the control is applied to a homogeneous object using one manipulation point, and one control point (Section 6.1). Next, a nonhomogeneous object is manipulated with one control point and two manipulation points (Section 6.2). Finally, we implement the controller scheme to a nonhomogeneous object using two control points and two manipulation points (Section 6.3). Each object and configuration is tested first in simulation using the a nonlinear RKPM model, then in experiments with an actual planar deformable object. Each simulation and experiment tests both the full state linearization (if possible) and the reduced version.

6.1 Homogeneous Object Set Point Regulation

The first object model used is a planar, square, homogeneous object broken up into a 9×9 grid of nodes as seen in Figure 6.1. The object is modeled using an RKPM mesh-free model measuring 140 mm in both height and width. This model is constructed to resemble the deformable object used in the experiments.

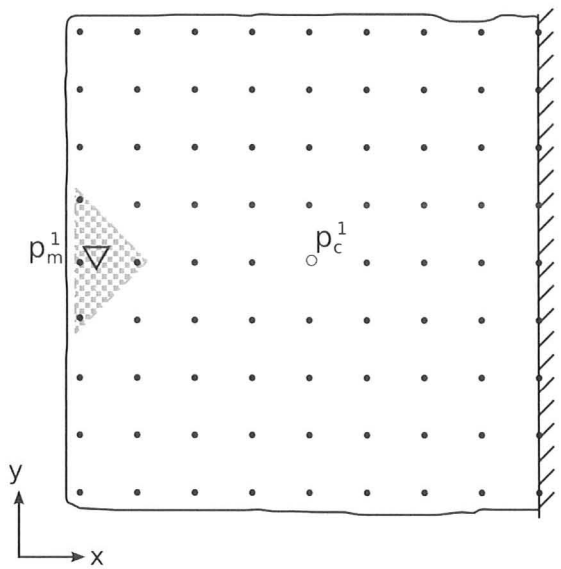


Figure 6.1: Planar deformable object discretized into a 9×9 grid.

The right edge of these nodes is fixed as an essential boundary, restricting the right-most 9 nodes to be immobile. This leaves 72 free nodes with both x and y displacements considered in the deformation model. The manipulator applies a force at the center of the left edge. In the experimental setup, the force is applied using a rod inserted into the object. To reflect this, we assume that that the force will be distributed amongst the four manipulation point nodes shown in Figure 6.1. It is important to note that these are not four independent manipulation points. A single control point is used for the output located at the center node of the object.

6.1.1 Full State

Reformulating the second order dynamics into a state space for a 72 node model results in 288 states. The state vector, input vector, output vector, and measured output vector are respectively

$$\boldsymbol{x} \in \mathbb{R}^{288}, \qquad \boldsymbol{u} \in \mathbb{R}^{2}, \qquad \boldsymbol{y} \in \mathbb{R}^{2}, \qquad \boldsymbol{y}_m \in \mathbb{R}^{10},$$

with corresponding state matrix, input matrix output matrix and measured output matrix

$$A \in \mathbb{R}^{288\times 288}, \qquad B \in \mathbb{R}^{288\times 2}, \qquad C \in \mathbb{R}^{2\times 288}, \qquad C_m \in \mathbb{R}^{10\times 288},$$

with D and D_m set to zero.

6.1.1.1 Simulations

For this model, we first run a simulation using the full 288 states for feedback control. The measured output we given noise characteristics of $0.01\,\text{mm}^2$ to simulated the noise observed in camera measurements. The control signal was also given additive noise of $1 \times 10^{-5}\,\text{N}^2$ to replicate the filtered force measurements observed in the experimental setup. The gain matrices are calculated from LQR optimal control and broken up into $K_1 \in \mathbb{R}^{2\times 288}$, $K_2 \in \mathbb{R}^{2\times 2}$, and $K_3 \in \mathbb{R}^{2\times 2}$ as described in Section 5.1.3. The LQR parameters used were $\alpha = 10$, $\beta = 1000$, $\gamma = 0.5$.

The set-point reference signal used for this simulation is $-3\,\text{mm}$ in the x-direction and $2\,\text{mm}$ in the y-direction. The output position, $\boldsymbol{y}(t)$, of the control point location is calculated from the RKPM nonlinear model with the applied input force control

signal. The output signal is shown in Figure 6.2. The dark blue and green solid lines represents the displacement of the control point in the x-direction and the y-direction, respectively. The lighter dashed blue and green lines represent the corresponding desired set-point reference signal.

The control signal $\boldsymbol{u}(t)$ applied to the model is shown in Figure 6.7. The blue represents the force applied in the x-direction and the green the force in the y-direction. Note that the input forces almost directly reach their steady-state value. This implies that, at this amount of deformation, the linearization is a good approximate of the actual nonlinear model used in simulation. Thus, the component of the force from the feedforward gain matrix nearly achieves regulation. Some correction from the internal model states of the controller, $K_2\boldsymbol{z}$, can be seen compensating for the error shown immediately after the step input.

For a visualization, the deformation of the object before and after the set-point reference step is shown in Figure 6.4. The original undeformed object nodes are shown as blue dots with the final object deformation as red circles.

6.1.1.2 Experiments

A homogeneous object is made from a single composition of super soft liquid plastic. The same control law and gains are applied to the experimental setup. This object is shown in the experimental setup in Figure 6.5.

For this experiment, the desired control point position was set to $-3\,\text{mm}$ in x, followed by $3\,\text{mm}$ in y. The control point trajectory captured by the camera, shown in Figure 6.6, clearly shows that the force applied by the robot manipulator are able to achieve sub-millimeter positioning of the control point. Comparing the input control

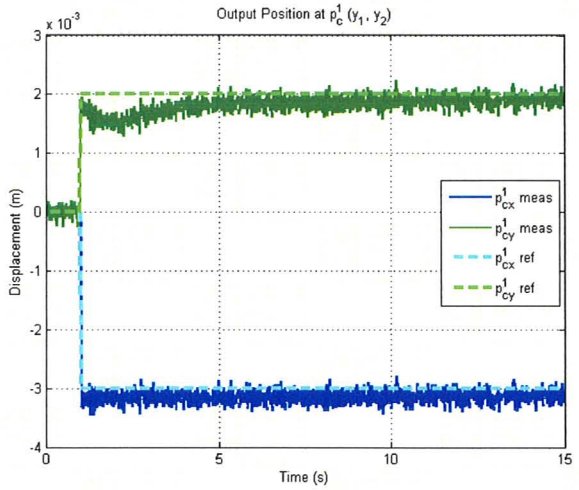


Figure 6.2: Simulated control point position output for full state estimation of the homogeneous object.

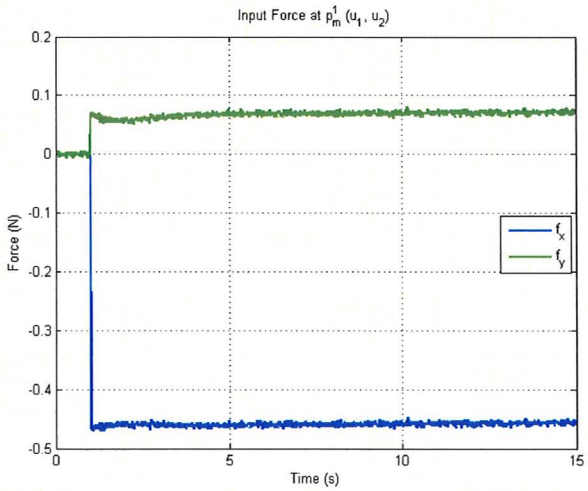


Figure 6.3: Simulated manipulation point input force for full state estimation of the homogeneous object.

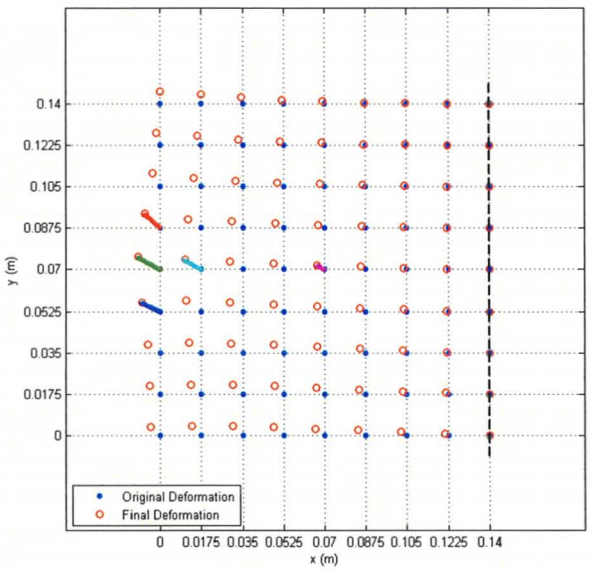


Figure 6.4: Simulated deformation of the homogeneous object after control point regulation with full state estimation.

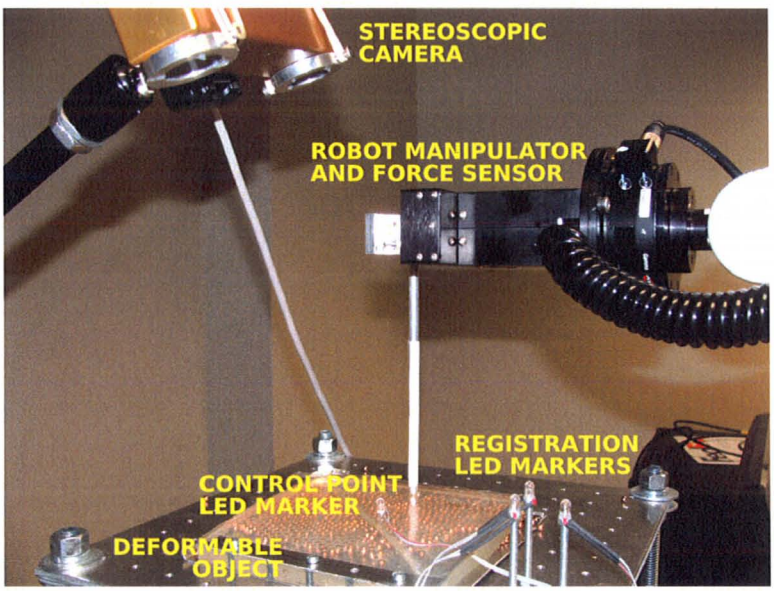


Figure 6.5: Experimental setup for the homogeneous planar object.

force in Figure 6.7, alongside the position output, we can see significant differences from the simulation. When the x-direction reference is commanded, there is slight deviation in the y-position, as well. This could be attributed to an imperfection in the camera to robot registration exact direction the robot applies the force on the object as well as to modeling errors representing the interactions between the input force in the x-direction and the output position in the y-direction. Another noteworthy observation is decrease in the force command even in areas where the position is relatively steady. This appears to be an example of relaxation in the object — a nonlinear property not taken into account in the RKPM model. However, the robust control law compensates for the discrepancy quite well.

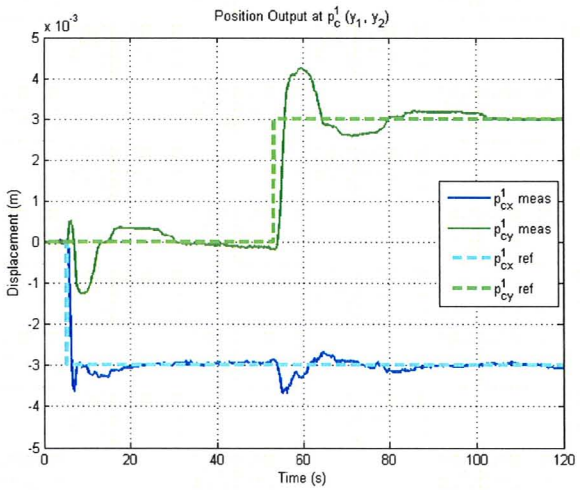


Figure 6.6: Experimental control point position output for full state estimation of the homogeneous object.

6.1.2 Model Reduction

In this section we apply model reduction to our homogeneous linear model to reduce the number of states in the state-space model while still capturing the dynamic

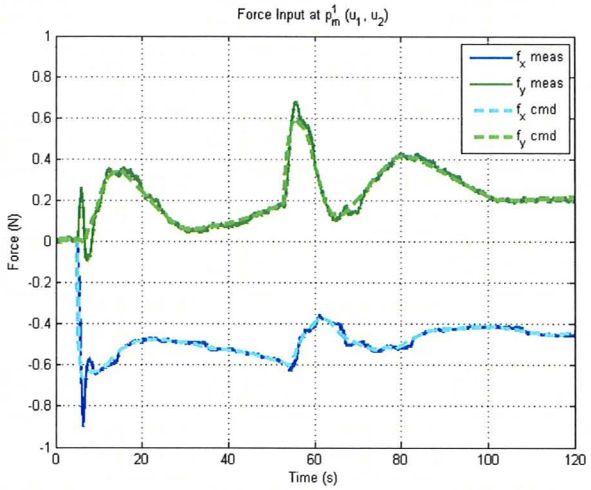


Figure 6.7: Experimental manipulation point input force for full state estimation of the homogeneous object.

behaviour of the object. Algorithm 4.1 described in Chapter 4 is used. For these simulations and experiments we reduce the model from 288 states down to just 12. The reduced model is compared to the old model in Figure 6.8 which shows the bode plots of the full state and reduced state MIMO systems. Using only 12 states the reduced system mimics the original almost identically from DC–10 Hz and only deviates slightly afterwards.

6.1.2.1 Simulations

The same simulation setup and parameters as the full state model are used. Also, we apply the control force onto the exact same nonlinear model. Applying a reference of -5 mm in x and -3 mm in y this time, we see the output position converge toward the target in Figure 6.9. The reduced model behaves quite similarly to the full-state counterpart, as results are quite similar. The larger step seems to have caused the linearization to deviate more drastically from the actual, indicated by the

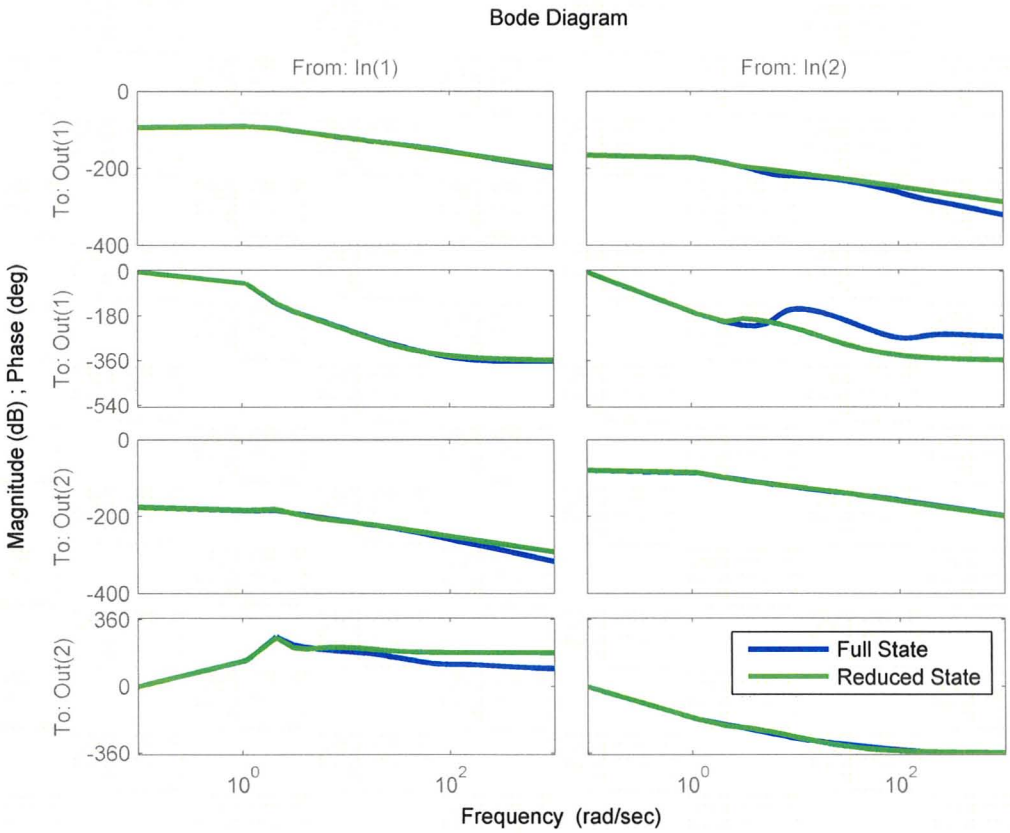


Figure 6.8: Bode plots comparing the full state (blue) model to the reduced state (green) homogeneous object model.

error in the y-position immediately after the reference. Changes to α, β, and γ can be made when determining the gain matrices from the LQR design to achieve faster convergence at the expense of larger control signals, potential overshoot, etc. As is evident in Figure 6.10, the force in the y-direction grows slowly after the initial feed-forward component. The parameters can be altered to achieve the desired transient response at the simulation stage, before applying to the practical experiment. The final deformation is overlaid with the original object again in Figure 6.11.

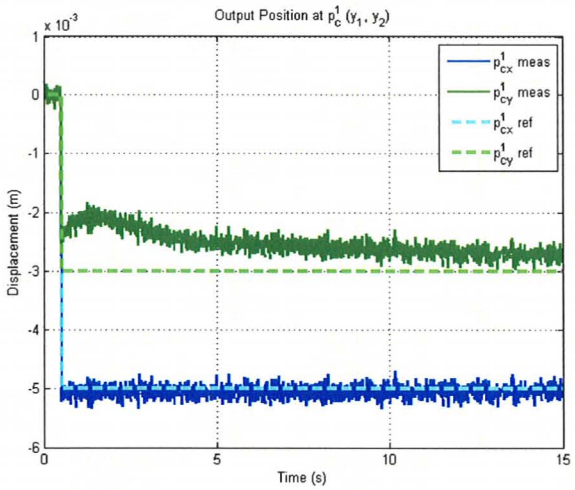


Figure 6.9: Simulated control point position output for reduced state estimation of the homogeneous object.

6.1.2.2 Experiments

The controller parameters determined from the reduced model simulation are used in the experimental setup on the homogeneous phantom. The control point trajectory is shown to converge to within sub-millimeter error of the reference again in Figure 6.12. The force applied is shown in Figure 6.13. The experimental results when using the reduced model are quite similar to the those of full-state controller. The reduced

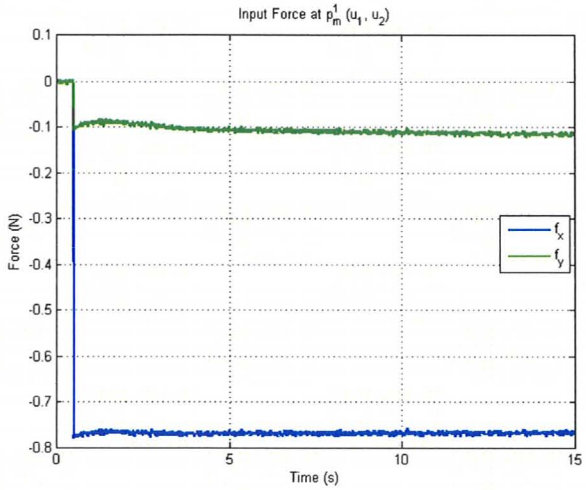


Figure 6.10: Simulated manipulation point input force for reduced state estimation of the homogeneous object.

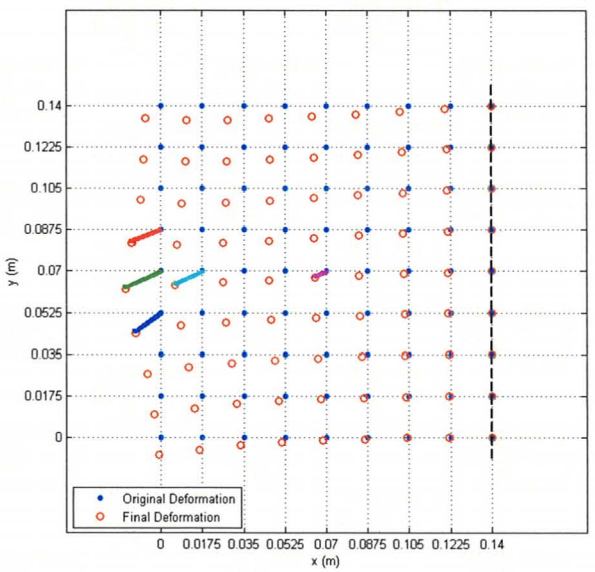


Figure 6.11: Simulated deformation of the homogeneous object after control point regulation with reduced state estimation.

state linear model clearly shows the capability of regulating set-point trajectories for our homogenous object.

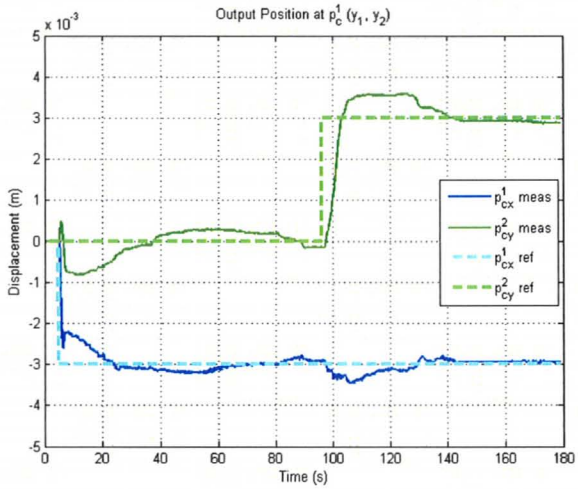


Figure 6.12: Experimental control point position output for reduced state estimation of the homogeneous object.

6.2 Nonhomogeneous Object Set Point Regulation

In this section we use the same control formulation but apply it to a more complex model with inhomogeneities and two manipulation points. The object this time is a circular object, approximately 140 mm in diameter, discretized into 205 nodes as shown in Figure 6.14. The bottommost 3 nodes are defined as an essential boundary in the model and thus fixed and not included in the states of the linearized model. The object is composed of material with two different consistencies. The majority of the object is relatively soft, with a stiffer rectangular region in center, defined by the dashed box. The nonlinear RKPM model for this object was defined with all positions in mm, and all forces in N.

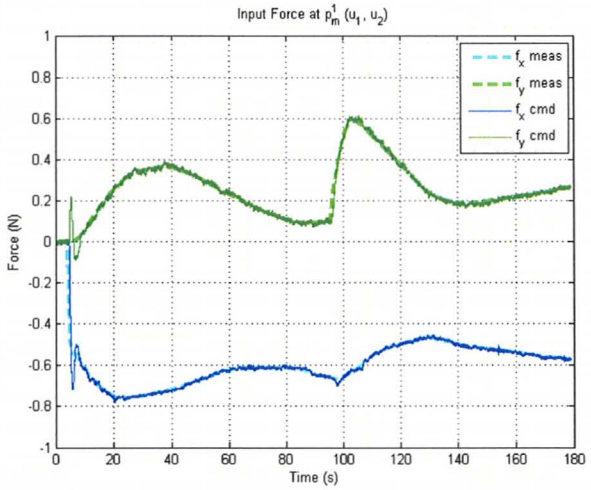


Figure 6.13: Experimental manipulation point input force for reduced state estimation of the homogeneous object.

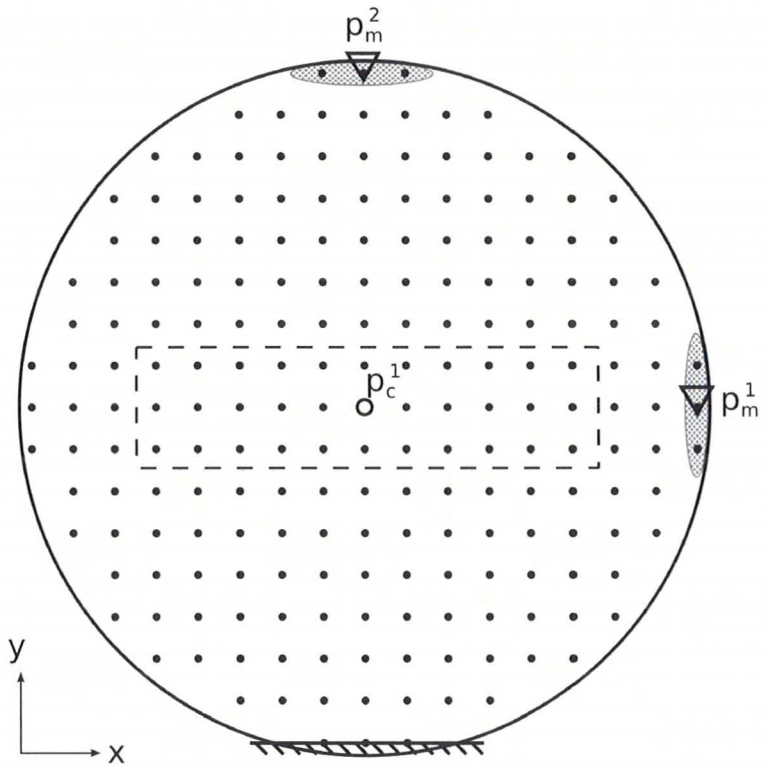


Figure 6.14: Nonhomogeneous planar deformable object discretized into a 205 nodes.

6.2.1 Full State

For 202 free nodes, the linear state space model will result in 808 states. With our configuration, we have two manipulation points, with forces distributed over 3 nodes each, and one control point which gives state-space vector and matrix dimensions

$$\begin{array}{llll} \boldsymbol{x} \in \mathbb{R}^{808}, & \boldsymbol{u} \in \mathbb{R}^{4}, & \boldsymbol{y} \in \mathbb{R}^{2}, & \boldsymbol{y}_m \in \mathbb{R}^{14}, \\ A \in \mathbb{R}^{808\times 808}, & B \in \mathbb{R}^{808\times 4}, & C \in \mathbb{R}^{2\times 808}, & C_m \in \mathbb{R}^{14\times 288}. \end{array}$$

6.2.1.1 Simulations

For the full state simulation, we design the controller with all LQR cost function parameters, α, β, γ, set to unity. The reference signal given is 1 mm in x and 3 mm in y. The simulation output from the nonlinear model is shown in Figure 6.15. The x-coordinate of the control point is shown in blue, the y-coordinate in green. The large spike in the x-coordinate immediately after the step is a result of the nonlinear model focusing on quasistatic representation of the object. The linearized model contained significant damping and inertial dynamic components not represented in the RKPM model. The results is a larger control force to overcome these dynamics that causes a larger than expected deformation in the nonlinear model. However, even with such a large discrepency between the linear and nonlinear model, the controller compensates effectively to reach the desired trajectory.

The control signal for manipulation point $\boldsymbol{p}_m^1$ is shown in Figure 6.16 a) and for manipulation point $\boldsymbol{p}_m^2$ in Figure 6.16 b). The initial and final deformation of the nonhomogeneous object is shown in Figure 6.17 along with the measured control and manipulation point trajectories, $\boldsymbol{y}_m$.

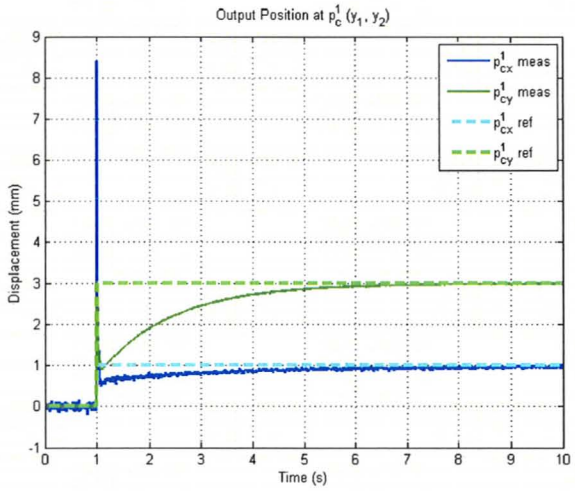
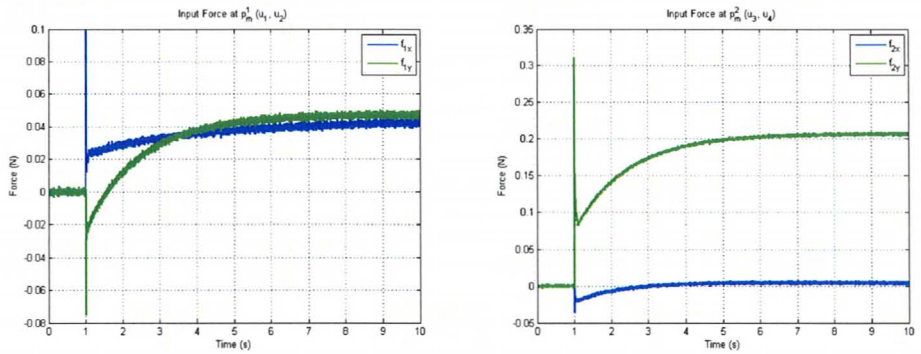


Figure 6.15: Simulated control point position output for full state estimation of the nonhomogeneous object.



(a) Manipulation Force 1 (b) Manipulation Force 2

Figure 6.16: Simulated manipulation point input force for full state estimation of the nonhomogeneous object.

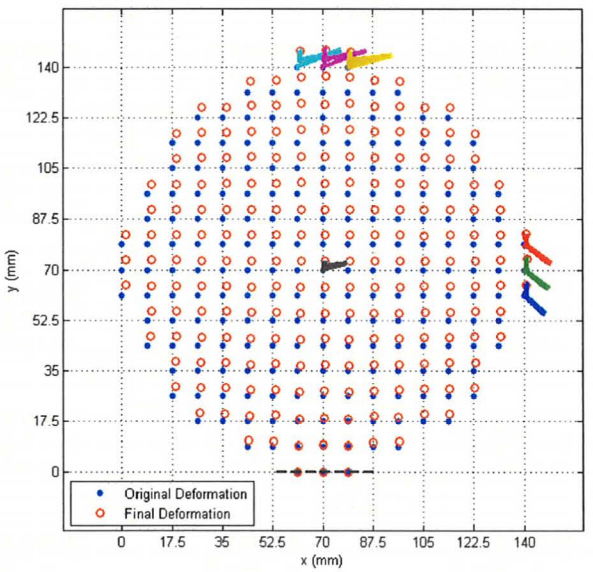


Figure 6.17: Simulated deformation of the nonhomogeneous object after control point regulation with full state estimation.

6.2.1.2 Experiments

The controller could not be implemented on the experimental setup for the full state model. The amount of memory required to estimate the full state model proved too much for the Simulink real-time workshop. This reinforces the motivation for accurate control laws formulated from reduced models.

6.2.2 Model Reduction

For the nonhomogeneous object, we apply model reduction as before, resulting in a linear state space system with 40 states. The bode plot comparing the reduced model to the full state nonhomogeneous model is shown in Figure 6.18.

6.2.2.1 Simulations

The same unity parameters were used for the α and β LQR parameters however $\gamma = 1000$. This is to ensure that the control signal cannot not grow too much larger than the approximated steady state values ($\boldsymbol{u}^*$) and therefore limiting the overshoot that will show up in the quasi-static nonlinear RKPM model.

Again using reference 1 mm in x and 3 mm in y, the results remain quite similar to the full state simulation. The output position is shown in Figure 6.19 with manipulation point forces for $\boldsymbol{p}_m^1$ and $\boldsymbol{p}_m^2$ shown in Figure 6.20(a) and Figure 6.20(b), respectively.

6.2.2.2 Experiments

The nonhomogeneous object was positioned in the experimental setup as shown in Figure 6.22. Again, the same reduced-state regulating controller was implemented

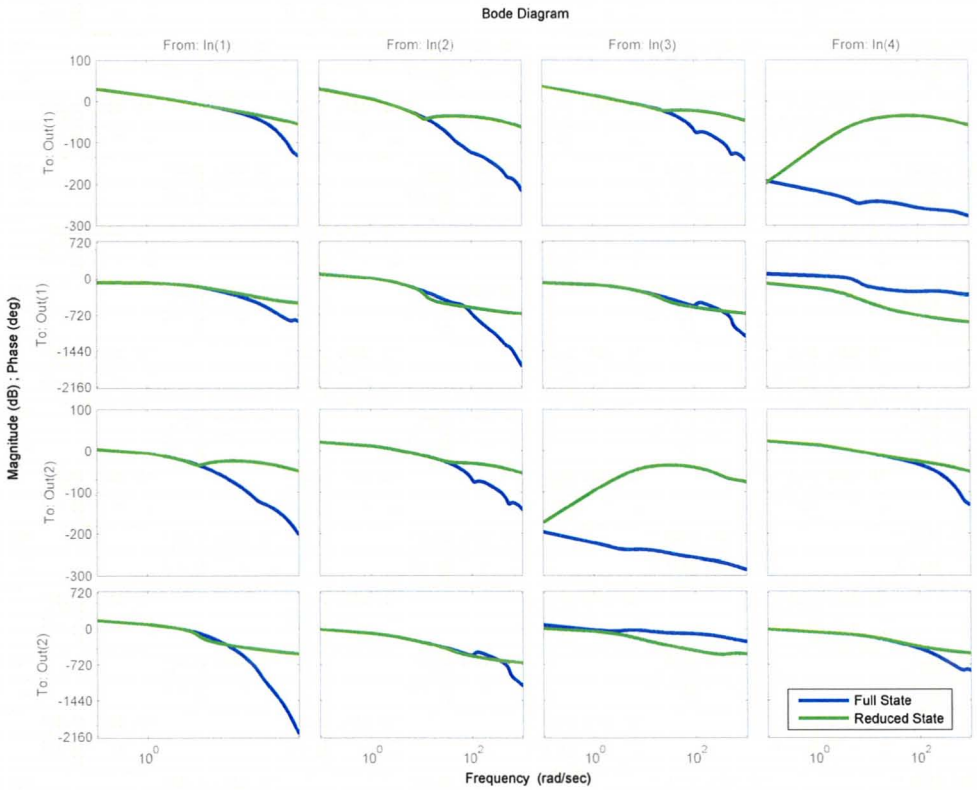


Figure 6.18: Bode plots comparing the full state (blue) to the reduced state (green) nonhomogeneous object models.

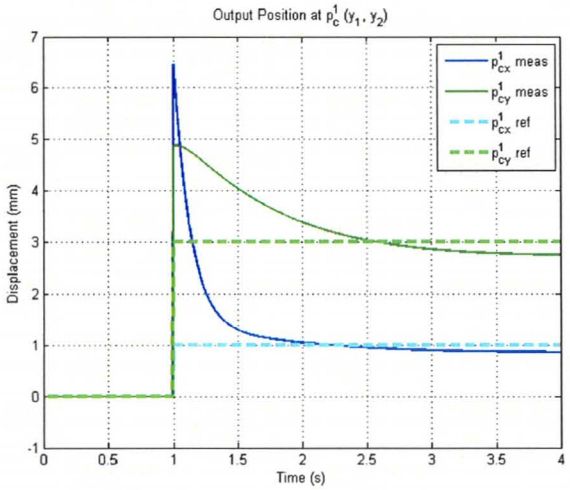
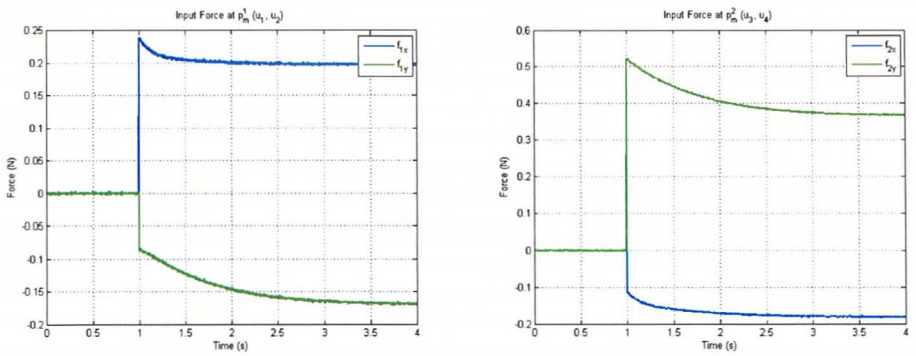


Figure 6.19: Simulated control point position output for reduced state estimation of the nonhomogeneous object.



(a) Manipulation Force 1 (b) Manipulation Force 2

Figure 6.20: Simulated manipulation point input force for reduced state estimation of the nonhomogeneous object.

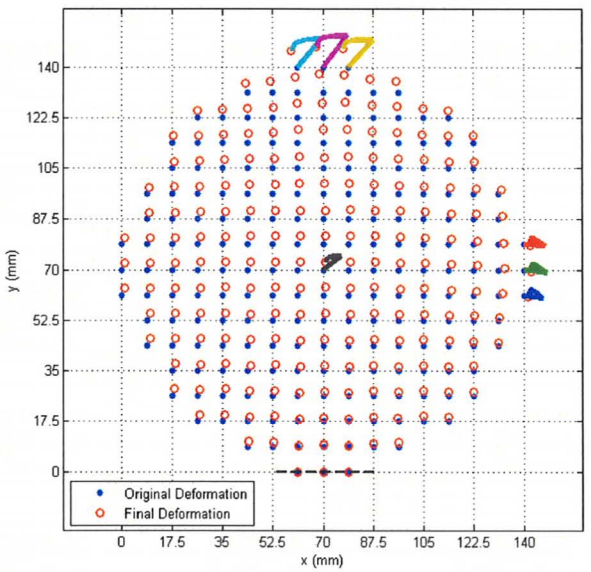


Figure 6.21: Simulated deformation of the nonhomogeneous object after control point regulation with reduced state estimation.

as in simulations and set-point reference steps applied. The recorded control point position is shown in Figure 6.23 with the corresponding forces for manipulation points 1 and 2 shown in Figure 6.24(a) and Figure 6.24(b), respectively. The reference signal in this experiment consists of several steps: first 1 mm in y, then 1 mm in x, then up to 2 mm before both returning to 0. The recorded control point position is noticeably noisier in this experiment then previously because of the position of the camera. Since two robotic manipulators were used an overhead view of the object was more obstructed. The camera had to be moved farther away and at the side which caused the y-direction in the object-space to be primarily corresponding to depth in the camera which has the poorest resolution. The resulting affect is a step-like chattering of the recorded control point position. However, it is evident from the experimental results that the nonhomogeneous object, with two manipulation points and one control point is still successfully regulated with set-point inputs.

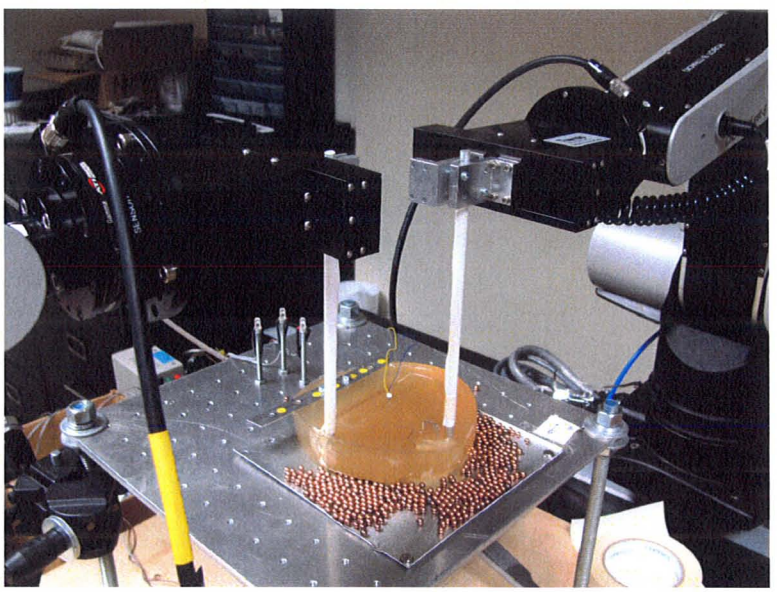


Figure 6.22: Experimental setup with nonhomogeneous object phantom.

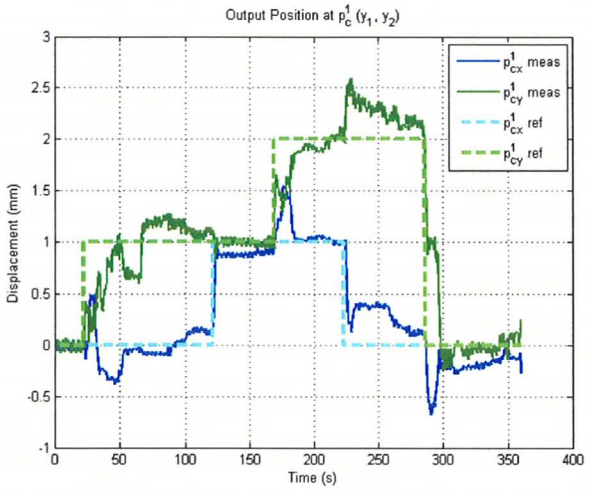
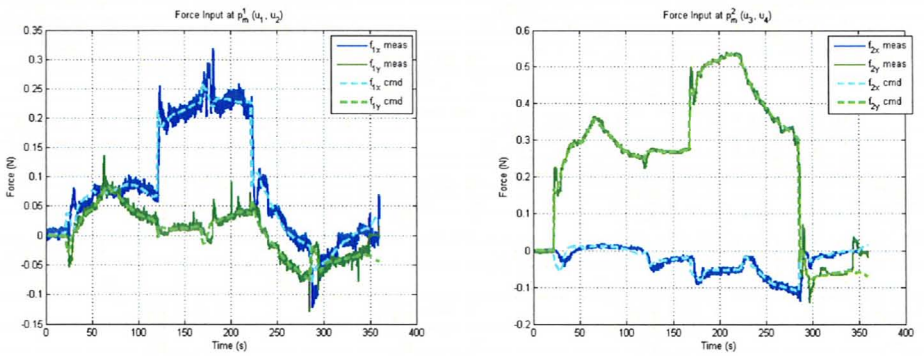


Figure 6.23: Experimental control point position output for reduced state estimation of the nonhomogeneous object.



(a) Manipulation Force 1 (b) Manipulation Force 2

Figure 6.24: Experimental manipulation point input forces for reduced state estimation of the nonhomogeneous object.

6.3 Two Control Point Nonhomogeneous Object Set-point Regulation

In this section, we use the same nonhomogeneous object as before, but require regulation of an additional control point position. The object configuration with two control point locations and two manipulation points is shown in Figure 6.25. The actual object fitted with LEDs is shown in Figure 6.26.

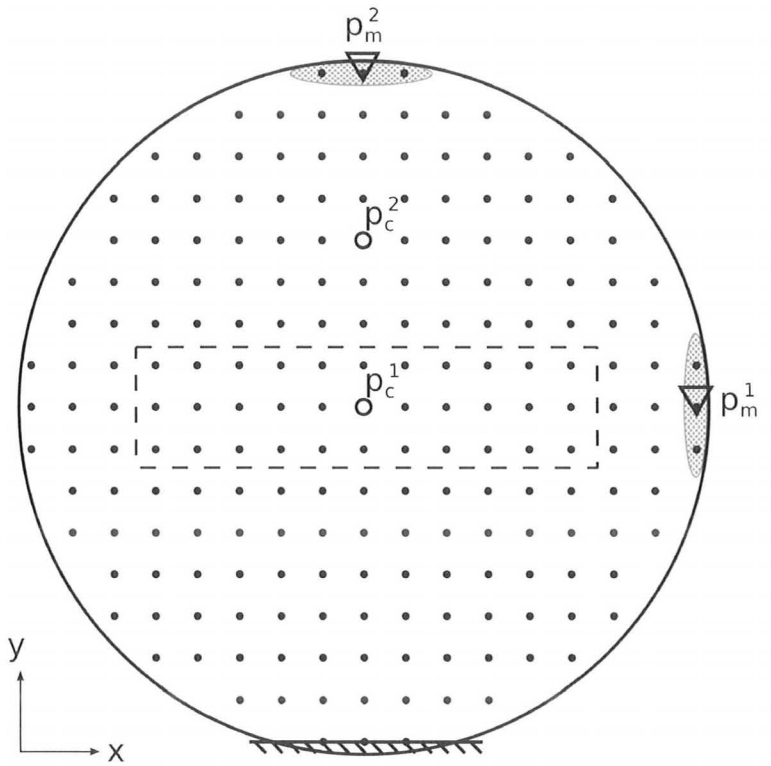


Figure 6.25: Nonhomogeneous, planar deformable object with two control points.

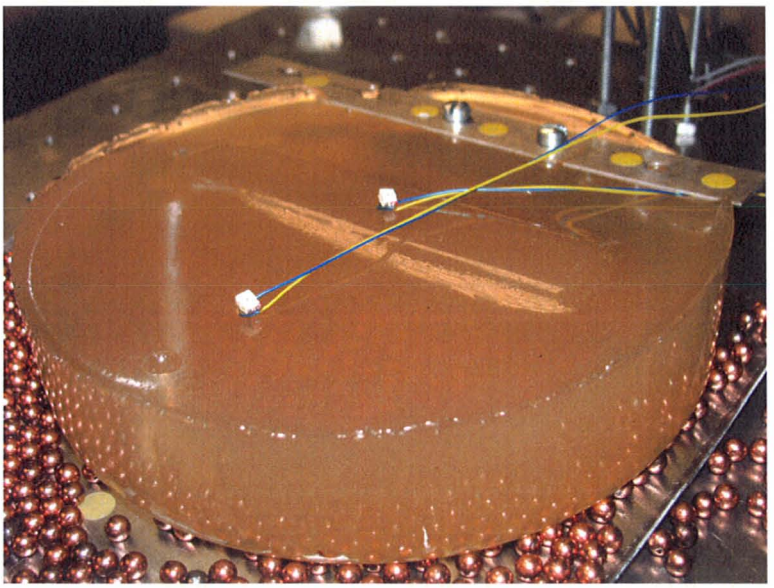


Figure 6.26: Actual nonhomogeneous deformable object with two control points.

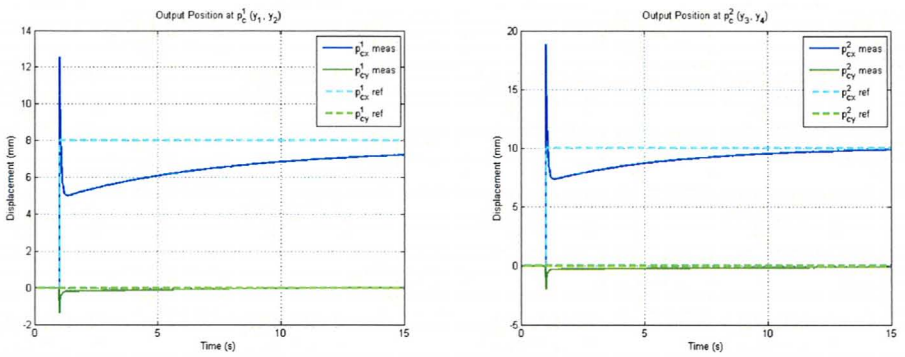
6.3.1 Full State

The linearized system remains the same as the previous model using one control point. The only change occurs in the output matrices, C and C_m, which are now of dimensions 4×808 and 16×808, respectively.

6.3.1.1 Simulations

For these simulations the object is subjected to a larger deformation inthe x-direction while remaining fixed in y. The set-point reference is 8 mm and 0 mm in x and y for control point $\boldsymbol{p}_c^1$ and 10 mm and 0 mm for control point $\boldsymbol{p}_c^2$. We tune the controller using LQR parameters to $\alpha = 0.01$, $\beta = 0.01$, $\gamma = 100$ to give a slightly slower integral action and still keep the control from growing too large from the steady-state value. The resulting simulation control point positions and input forces are shown in Figure 6.27, and Figure 6.28, respectively. The object deformation is shown in

Figure 6.29.



(a) Control Point 1 Position (b) Control Point 2 Position

Figure 6.27: Simulated outputs for full state estimation of the nonhomogeneous object with two control points.

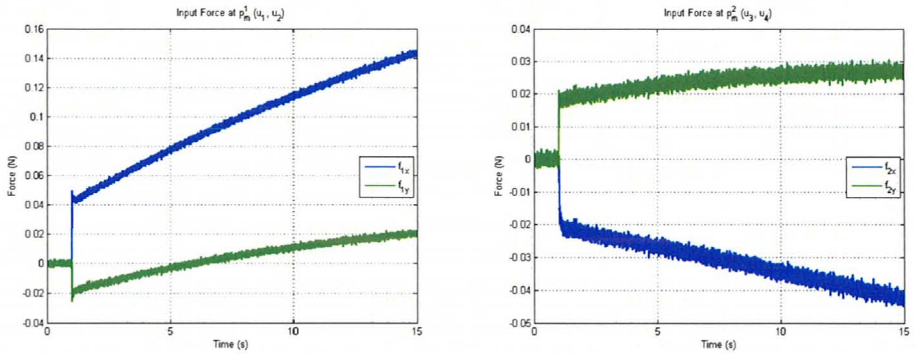
Again, we see significant overshoot stemming from the lack of proper inertial and viscous characteristics in our nonlinear model implementation. However the system remains stabilized and converges to the desired set-point postitions even under extreme deformation with multiple control and manipulation points.

6.3.1.2 Experiments

As before, the state dimension for the full model is too large to perform an experimental study with the robotic manipulators. The feasibility of the regulation control in experiments is demonstrated using the reduced model in the next sections.

6.3.2 Reduced State

As in the single control point case, the reduced-state model is represented again by 40 states. The comparison of full-state linear model to reduced-state system is shown



(a) Manipulation Force 1 (b) Manipulation Force 2

Figure 6.28: Simulated manipulation point input forces for full state estimation of the nonhomogeneous object with two control points.

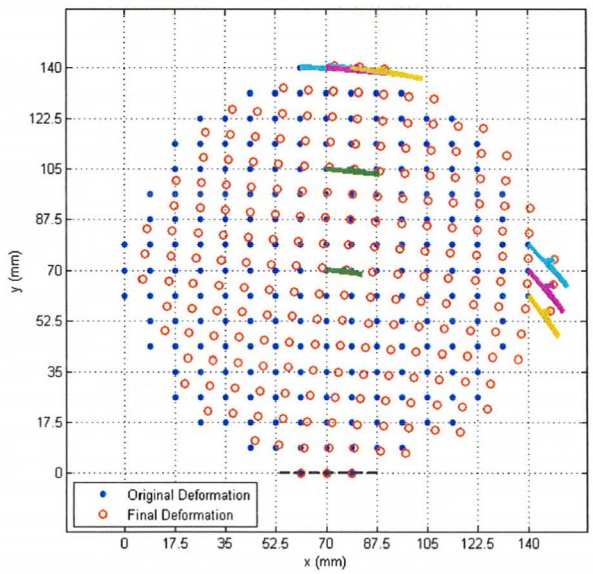


Figure 6.29: Simulated deformation of the nonhomogeneous object after control point regulation with full state estimation.

the bode plots of the MIMO system in Figure 6.30 with 4-inputs and 4-output signals.

6.3.2.1 Simulations

The simulation results for the reduced model are almost identical to the force inputs and position outputs to the full-state linear model. This implies the 40 state representation for this model is just as suitable as the 808-state model for regulating two control points. The control point positions in Figure 6.31, and input forces in Figure 6.32, as well as object deformation in Figure 6.33 are nearly identical to their full-state counterparts.

6.3.2.2 Experiments

The reduced state control gains of the simulation were implemented on the robotic experimental setup. For these experiments the same, large-deformation reference signals are applied of $(x, y) = (8, 0)$ for control point $\boldsymbol{p}_c^1$ and $(x, y) = (10, 0)$ for $\boldsymbol{p}_c^2$. The output position results are shown in Figure 6.34 and control signals in Figure 6.35. The results are promising. Immediately after the step, the feedforward contribution brings the output towards the desired targets. Due to nonlinearities and modeling inaccuracies, the initial feedforward compensation deviates significantly from the set-point values. However, the internal model of the regulating controller simultaneously brings both the control points to the desired values.

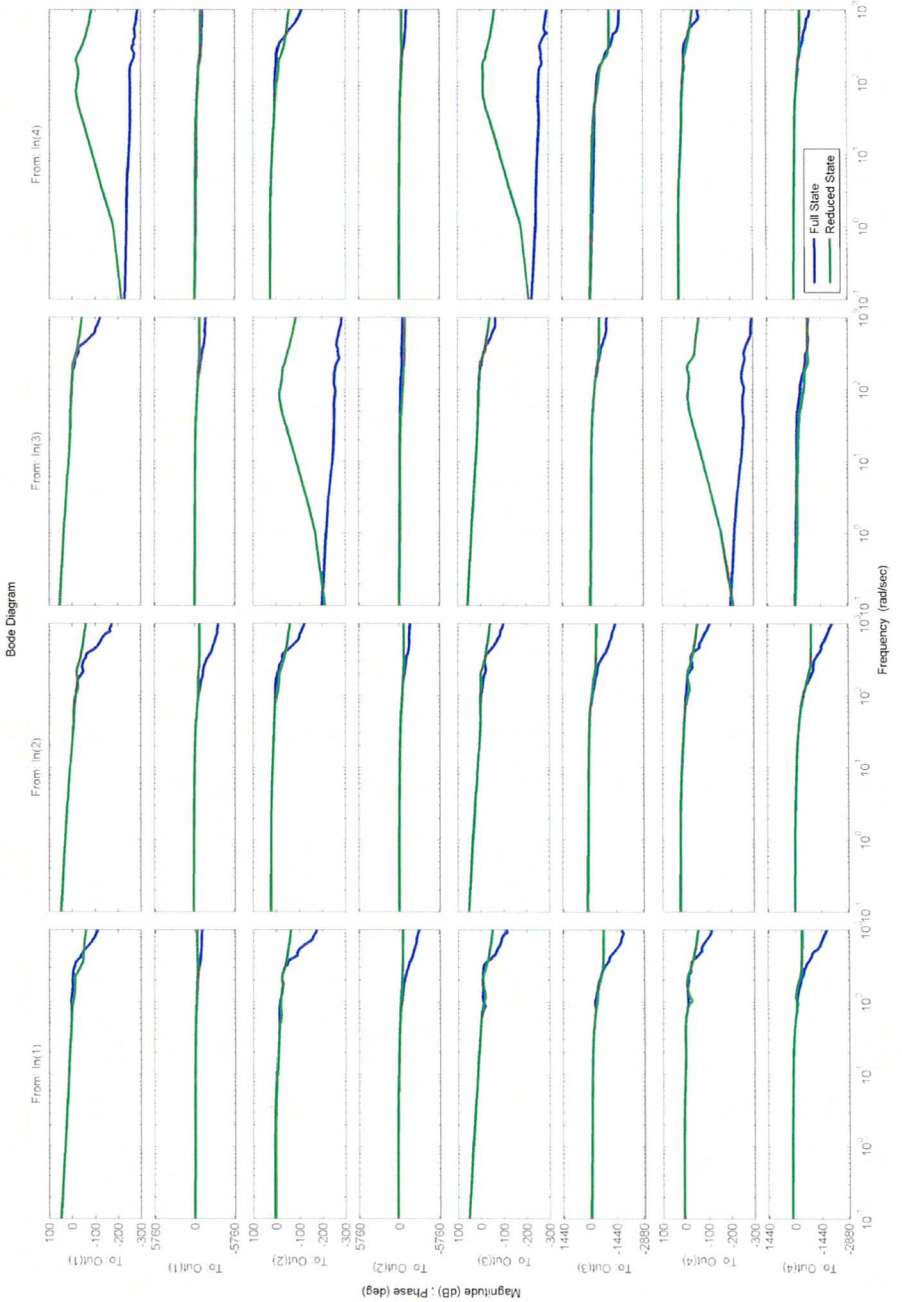
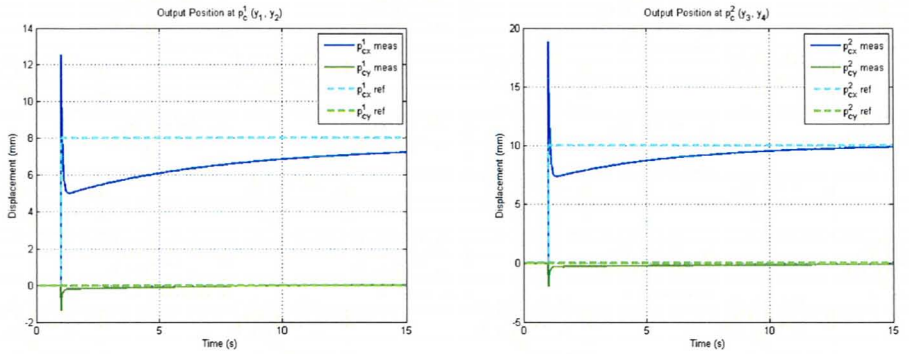
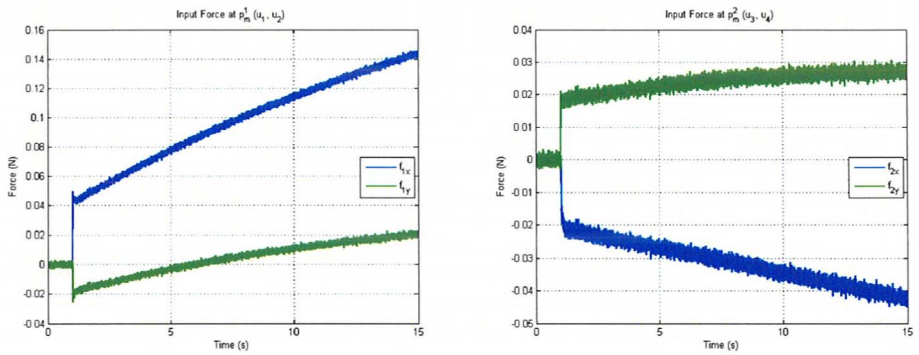


Figure 6.30: Bode plots comparing the full state (blue) to the reduced state (green) nonhomogeneous object models with two control points.



(a) Control Point 1 Position (b) Control Point 2 Position

Figure 6.31: Simulated outputs for reduced state estimation of the nonhomogeneous object with two control points.



(a) Manipulation Force 1 (b) Manipulation Force 2

Figure 6.32: Simulated manipulation point input forces for reduced state estimation of the nonhomogeneous object with two control points.

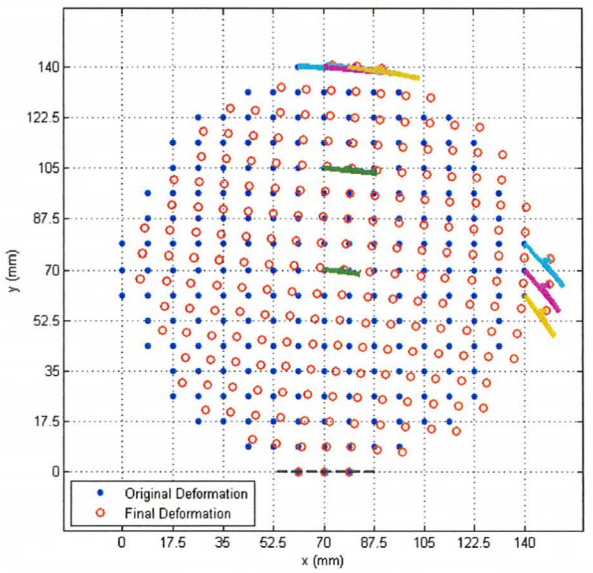
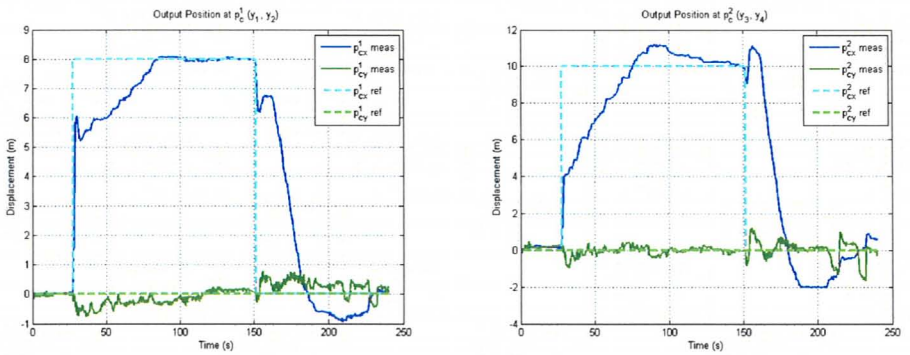
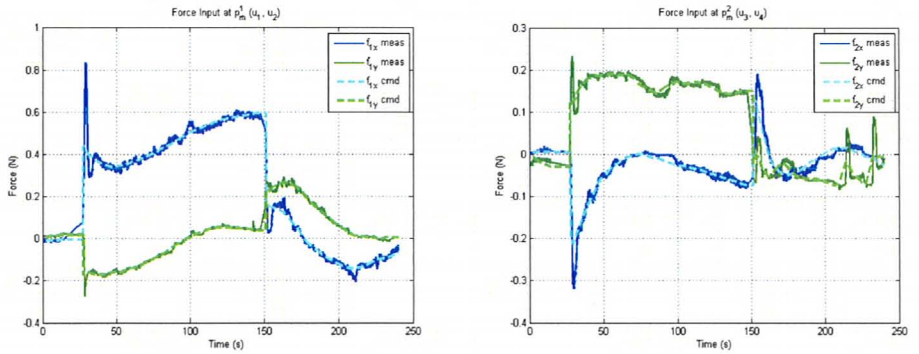


Figure 6.33: Simulated deformation of the 2 control point nonhomogeneous object with reduced state estimation.



(a) Control Point 1 Position (b) Control Point 2 Position

Figure 6.34: Experimental outputs for reduced state estimation of the nonhomogeneous object with two control points.



(a) Manipulation Force 1 (b) Manipulation Force 2

Figure 6.35: Experimental manipulation point input forces for reduced state estimation of the nonhomogeneous object with two control points.

6.4 Summary

These simulation and experimental results represent various configurations of objects together with various controller performance parameters. The objects and configurations tested include:

- homogeneous planar objects
- nonhomogeneous planar objects
- one-control point, one-manipulation point configuration
- one-control point, two-manipulation point configuration
- two-control point, two-manipulation point configuration

The results show the potential for stable and robust convergence with all configurations under both small and large deformations while showing significant freedom for a designer to set controller performance.

Chapter 7

Conclusions and Discussion

This thesis proposes a new approach to designing the control law that solves the indirect simultaneous positioning problem. This approach is based upon the linearization of an accurate object model representing the second-order dynamics of the physical object. From the linearization, a state-space model of the object is obtained which is then controlled using robust output regulation theory. This controller forces convergence of the output variables to their desired references in the presence of uncertainty and deviation of the plant model parameters from their nominal values. Due to the inherent state-feedback form of this controller, and the often large number of nodes involved in discretization of accurate object models, the state vector can become significantly large and unrealistic to implement in real-time experimental procedures. Therefore, model reduction techniques are investigated in this thesis to retain accurate representations of the object dynamics while reducing the size of the state vector needed for feedback. The control framework is verified in both simulations using MATLAB's Simulink®, and experiments using robot manipulators.

7.1 Discussion

Some previous reported results use control schemes that rely on a linearization between the differential motions of the manipulation and control points, namely the Jacobian. A benefit of this formulation is that it allows the input variable to be the position of the manipulation points. This may eliminate the need for additional sensors such as the force/torque sensors in the experimental setup of this thesis, and reduce overall costs. However, these controllers neglect information about the inertial and viscous effects of the object's motion. The benefits of using a dynamic model for the basis of the control law include more flexibility in designing for desired transient responses.

A controller that is based on second order dynamics allows the freedom of changing controller parameters to obtain desired transient response characteristics without compromising the stability of the system. Additionally, using state-feedback for the stabilizing portion of the control allows for a more systematic approach dealing with objects that may have complicated compositions and behaviour. Designs focused on simply PID control gains on error signals may not be able to stabilize objects with certain configurations and boundary conditions.

An important note to the reader is that the control scheme presented in this thesis is robust in a classical output regulation definition, such as defined in the work of Francis (1977) and Huang (2004). The controller is robust by construction to uncertainty and nominal plant value deviation within some neighbourhood of parameter variation. The control remains stabilizing and regulating so long as the controller gains defined for the nominal plant still stabilize the closed loop system, and that the rank condition (3.33) remains valid, given the plant variation. Controllers exist that

use *a priori* knowledge of the entire range of plant parameter variations to guarantee robustness within that range, but require a different approach, such as H_∞ control, and are beyond the scope of this thesis.

The experimental procedures presented in this thesis demonstrate the feasibility of this control architecture in practice. Depending on the setup, there may be other circumstances to consider. For instance, the position feedback of the control point used in this thesis was based on a stereoscopic camera and infrared filters and LEDs. Using other imaging modalities, such as ultrasound or MRI, may require more complicated procedures to properly complete the coordinate registration and image segmentation to determine the position of any measured output points. Also, since the control scheme is based on the dynamic characteristics of the object, there is an inherent lower bound on the sampling rate of which the output deformation is fed-back in order to facilitate the controller in real-time.

7.2 Further Development

Further research could include extending this control approach beyond planar objects to the 3D case. Additionally, the branch of work could be expanded to include even more complicated objects with holes and more inhomogeneities, as well as objects without a fixed essential boundary. In the future, there would also be a more thorough analysis of the amount of deformation and uncertainty allowed in the model while still ensuring stability and robustness to set-point reference trajectories.

Additionally, we could explore different tasks such as needle insertion. This problem would be quite similar in formulation but involve the convergence of the control point to a line, namely the needle path, rather than to a fixed point. This would

likely also involve incorporating a more complicated object model that accounts the change in object characteristics due to the needle's insertion.

The area of model reduction provided a means for reducing the state vector required in the controller synthesis while still capturing the characteristics of the object. However, there is room for a more thorough investigation into the optimal order of the reduced model for a given object. A more systematic choice the the number of matched moments in the reduced model would be beneficial in the controller design process. Also, there would be value in an comparing a controller that first reduces the model for the basis of a "full-state" controller, as presented in this thesis, and one that determines optimal projection equations for reduced order compensation. The key difference being when during controller design a reduction procedure occurs.

Another area of focus could involve coupling the now separate control aspects of the object controller and the robot hybrid controller. This would involve including the robot dynamics into the interaction with the object and treating the inputs to the system as the manipulator joint torques, rather than the forces on the object manipulation points.

Finally, the results show successful manipulation and and object with more inputs than outputs, which implies there is no unique solution to the required input forces that achieve the desired output positions. However, there was no exploration into the benefits of the additional degrees of freedom in the system and how to design the system to converge to one solution over another. Future work could involve similar experimental setups that use this extra degree of freedom to incorporate more complex path planning or to impose some constraints on the manipulation points.

Appendix A

Model Parameters Optimization

The properties of the hyper-elastic plastic materials were determined by experimentally deforming the object with a robot manipulator. Both the deformation of the object was recorded as well as the interacting forces applied by the manipulator. Figure A.1 shows a sample of material being deformed with certain point locations being recorded by a stereo vision camera. Simulations were then performed while tuning object parameters until the simulated deformation matched the experimental deformation. For these objects, the second Piola-Kirchhoff stress is defined as

$$S(E) = \lambda \operatorname{tr}(E) I + 2\mu E \tag{A.1}$$

where E is the Lagrangian Green strain and λ and μ are the first and second Lamé parameters. This is derived from the partial derivative of the strain-energy density function

$$W(E) = \frac{\lambda}{2} [\operatorname{tr}(E)]^2 + \mu \operatorname{tr}(E^2). \tag{A.2}$$

The parameter identification comprised two steps. In the first step, a grid search

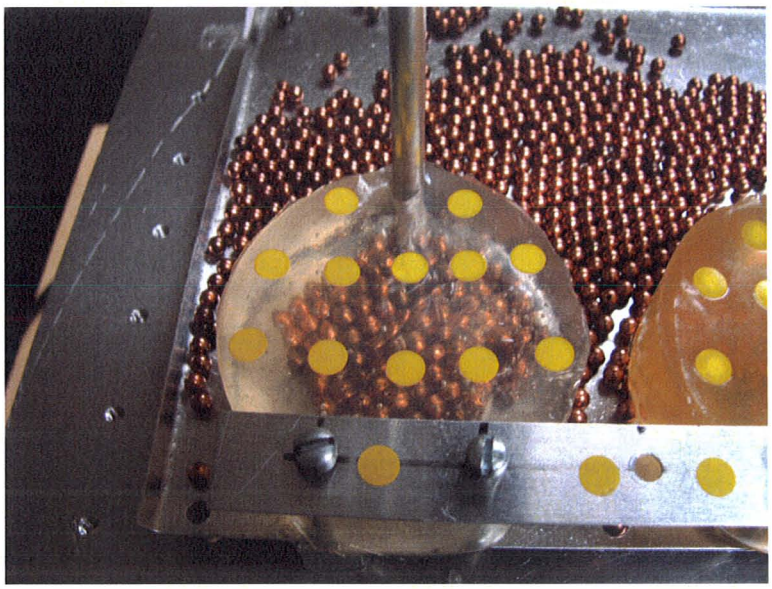


Figure A.1: Deformed object sample with markers to track deformation.

was used to identify a set of parameters that matches approximately the object. Then, an optimization routine was started from that point.

The cost function that is used for identifying λ and μ is defined based on the difference between the recorded position of the control points and the positions of them in simulation. This cost function should be minimized such that the best values for λ and μ that match the deformation of the simulated object and the deformation recorded from the experiment are found. Therefore this cost function is defined as

$$f(\lambda, \mu) = \sum_{i=1}^{n} \|p_i - q_i\|_2^2 \tag{A.3}$$

where p_i's are the positions of the control points on the object, q_i's are the positions of the control points in simulation, and n is the number of control points. The surface plot of the values of this cost function as a function of λ and μ is shown in Figure A.2.

Using the above mentioned algorithm for identifying the object's parameters leads

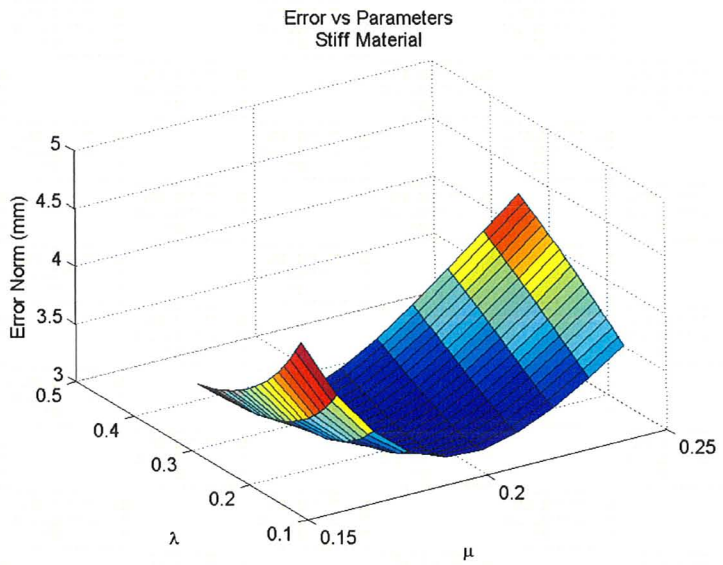


Figure A.2: Surface plot of the error cost as a function of the parametres μ and λ.

to values of 0.189151 and 0.147688 for μ and λ, respectively. Figure A.3 shows the simulation of the deformation with both the simulated control points positions and the recorded control points positions for the object with these parameters. Error histogram for the simulation is also shown in Figure A.4. The mean error of the simulation was 0.80 mm with standard deviation of 0.46 mm.

For the nonhomogeneous objects, another solution using a plastic softening additive was used to create another sample with distinct material characteristics. The second, softer material deformation was recorded experimentally and parameter-matched using the same process. The cost function for the Lamé parameters of this material is shown in Figure A.5. The optimal parameters are found to be $\mu = 0.039$ and $\lambda = 0.043$. The simulated marker point deformation is compared to the experimental recordings in Figure A.6 with the error histogram shown in Figure A.7.

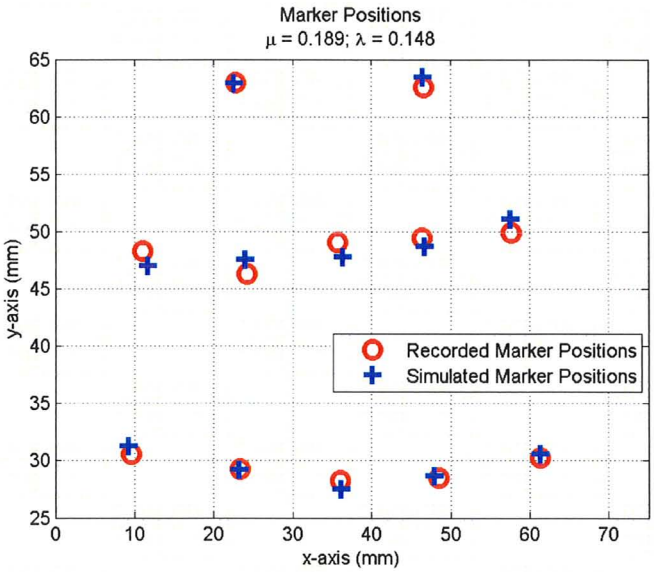


Figure A.3: Object deformation showing both the simulated control point positions and the recorded control point positions.

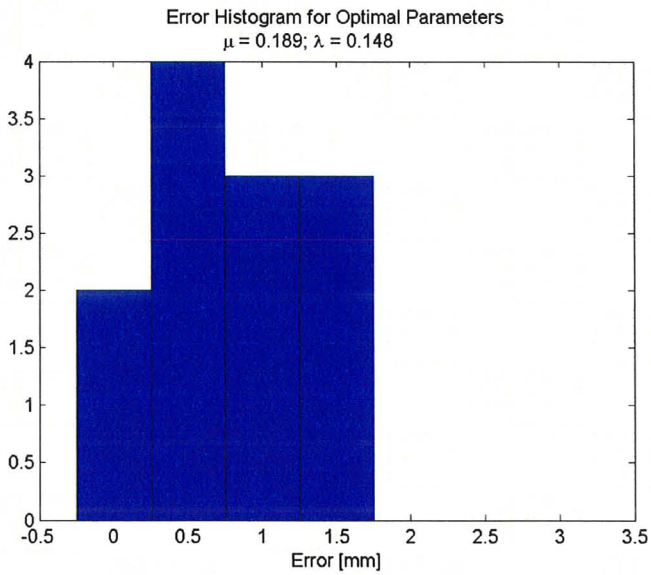


Figure A.4: Error histogram of the simulated vs recorded marker point deformation.

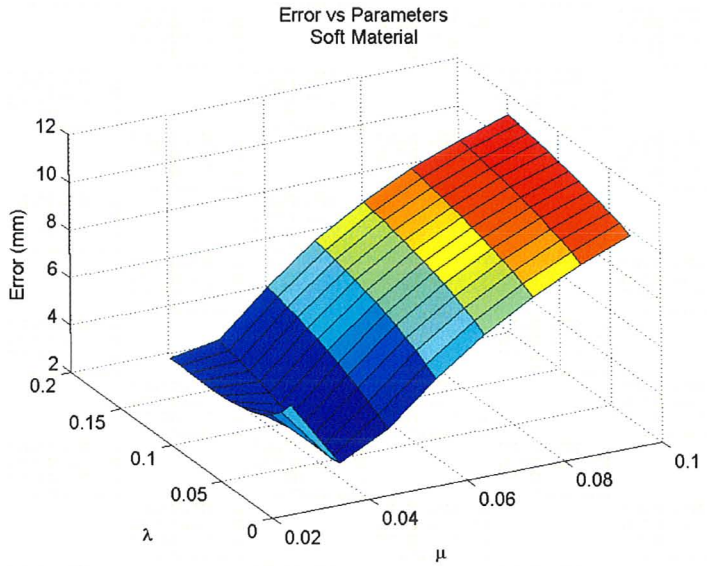


Figure A.5: Surface plot of the softer material error cost function.

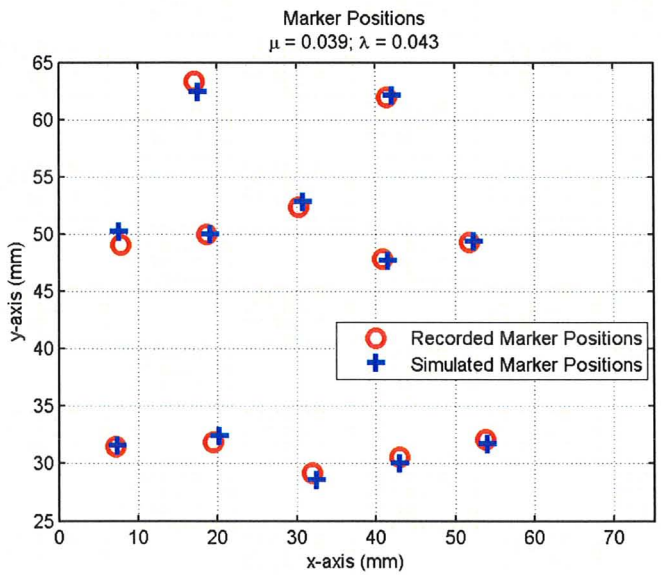


Figure A.6: Object deformation showing both the simulated vs recorded control point positions of the softer material.

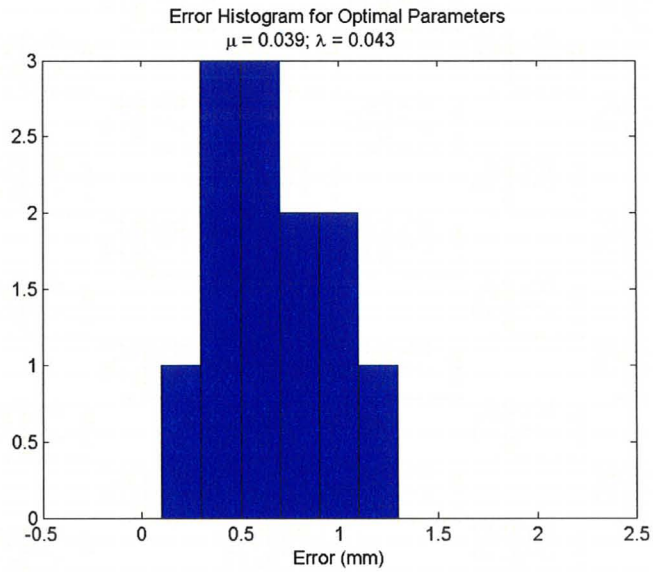


Figure A.7: Error histogram of the softer material marker point deformation.

Appendix B

Image Acquisition and Camera-Robot Registration

For all experimental tests, the position of the control point is tracked using the Bumblebee®2 Stereo Vision CCD camera from Point Grey Research. The camera captures two images at a resolution of 800×600 and transferred to a computer via IEEE-1394 (Firewire). We add to the camera two infrared filter lenses from Newport Corporation with a cutoff frequency of 780 nm (part no. FSR-RG780), as seen in Figure B.1. Using the filters and infrared LEDs as the markers allows easier segmentation of the points of interest in the cameras field of view.

Custom software was developed using the Triclops SDK from Point Grey Research to process the images acquired by the camera and determine the 3D coordinates of the control point on the object. The general outline of the software algorithm to capture the control point deformation is given in Algorithm B.1.

The camera is initialized so that, given the ambient setting of the experiment, only the infrared LEDs, are visible in the camera image. For our experiments, this is

Algorithm B.1 Image Acquisition and Registration

```
initialize camera
while camera output required do
  read left and right camera images
  binarize both images:
  for (each pixel in image) do
    if pixel > threshold then
      pixel = 1
    else
      pixel = 0
    end if
  end for
  segment both images into marker blobs
  convert matching blobs into 3D points
  sort 3D points into registration and control points
  if first image acquired then
    compute registration from 3 reg. points
  end if
  convert 3D points from camera coords to robot coords
  for first 20 images do
    average the control points,
    set as initial position
  end for
  compute control point displacement from initial positon
  send displacement over UDP to simulink
end while
```

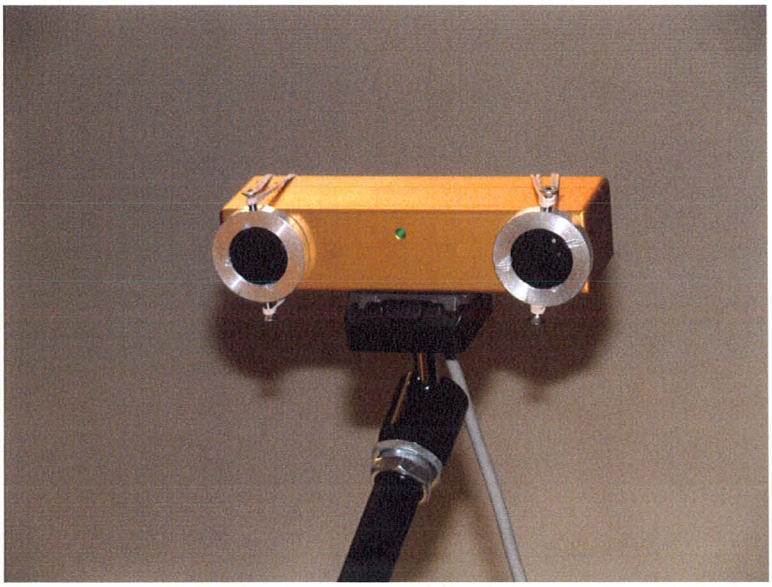


Figure B.1: Bumblebee®2 Stereo Vision camera with infrared filters.

done by manually adjusting the camera settings such as aperature time, exposure and gain. This reduces any chance of false positives during segmentation and recognizing an artifact as an LED marker.

The left and right camera images are acquired and stored as 800×600 arrays of 8-bit greyscale values. Both images then binarized so that only pixels above a certain value are retain and everything else filtered out. For our experiments, we use a threshold value of 225. Any pixel above this value is taken to be a part of our marker and set to `true`.

Next the images are segmented using a *blob* finding algorithm that creates a linked list of pixel groupings. There will be one blob for every registration LED and one for every control point in each image. These blob lists are then sorted so that the the blobs for the right image can be matched to the corresponding blobs of the left image.

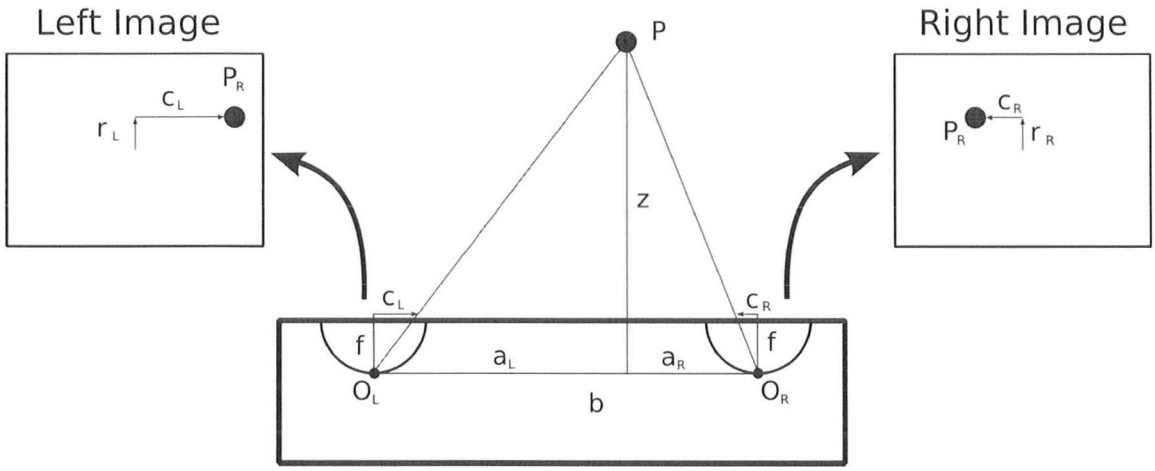


Figure B.2: Stereo vision geometry showing how a single point represented in the separate camera images.

Once two lists of blobs are computer in matching order, we use the centroids of a matching pair to compute a single 3D coordinate. We do this using the SDK function that returns an x,y,z-coordinate from the disparity (difference in column value) of the location in each image that should represent the same physical point, as illustrated in Figure B.2. Note, that because the camera lenses are aligned vertically, the row direction values should be the same in both images. The disparity can be converted to a 3-dimensional x,y,z-coordinate using what is called epipolar geometry. The tricplops SDK includes a function uses the camera baseline, b and focal-length, f to return a 3D point when given a disparity. This conversion is made for every pair of matching blobs in the images. In our experiments, we will get three distinct 3D points for the registration markers, and one point for every control point being tracked.

Once all points are given in 3D coordinates in camera the camera frame, the three point recognized as the registration LEDs are used in a to determine the rotation between the robot frame and the camera frame. This registration only needs to be performed once, and is completed upon receiving the first pair of images. Refer to

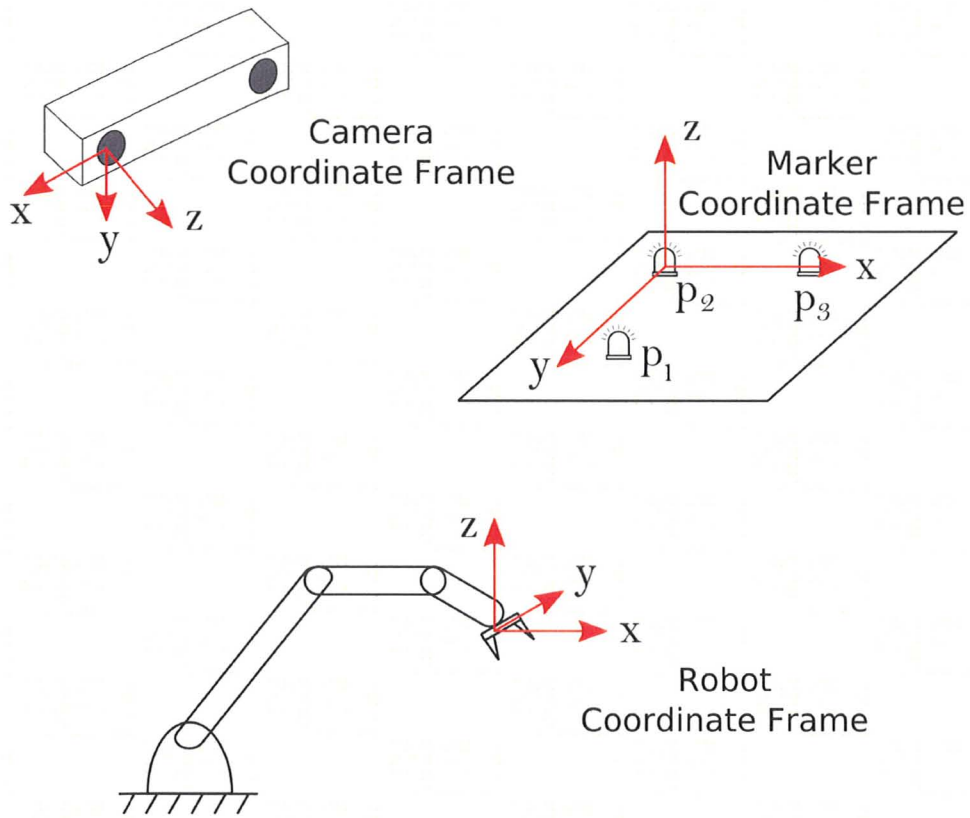


Figure B.3: Illustration differentiating the robot frame, the camera frame, and the marker frame.

Figure B.3 for an illustration of the different coordinate frames required for registration. Note, that since all our experiments focus on displacements, we only need to solve for the rotation that relates the orientation of the camera coordinate system to the robot coordinate system. Otherwise, we would have to acknowledge a translation between the coordinate frames, as well.

First, from the three points captured in the camera frame, we can create an intermediate orthogonal coordinate frame labeled the *marker frame*. Let the three registration points captured in the camera frame be $\boldsymbol{p}_1^C$, $\boldsymbol{p}_2^C$ and $\boldsymbol{p}_3^C$. Create a vector that will represent the marker space x-axis as seen by the camera; in this case $\boldsymbol{x}^C =$

$\boldsymbol{p}_3^C - \boldsymbol{p}_2^C$. Create a second vector that we assume is in the x,y-plane of the marker space: $\boldsymbol{v}^C = \boldsymbol{p}_1^C - \boldsymbol{p}_2^C$. Note that we cannot assume this to be exactly the y-axis because it may not be orthogonal to the x-axis we chose, as shown in the figure. We can however, create a third vector, orthogonal to this plane to use for the z-axis: $\boldsymbol{z}^C = \boldsymbol{x}^C \times \boldsymbol{v}^C$. Now, find the orthogonal y-axis as $\boldsymbol{y}^C = \boldsymbol{z}^C \times \boldsymbol{x}^C$. The three axis of our intermediate marker space are now given in camera frame coordinates as

$$\boldsymbol{x}^C = \frac{\boldsymbol{x}^C}{||\boldsymbol{x}^C||_2}, \qquad \boldsymbol{y}^C = \frac{\boldsymbol{y}^C}{||\boldsymbol{y}^C||_2}, \qquad \boldsymbol{z}^C = \frac{\boldsymbol{z}^C}{||\boldsymbol{z}^C||_2}.$$

The rotation that relates any point in the marker space to the camera space is given by

$$R_M^C = \begin{bmatrix} \boldsymbol{x}^C & \boldsymbol{y}^C & \boldsymbol{z}^C \end{bmatrix}.$$

In a very similar manner, we construct a rotation that relates the orientation of the robot coordinate frame to the marker coordinate frame. The registration LED points, need to be known in the robot coordinate system. This is commonly done by touching the robot end-effector to the three points of interest and recording their locations. In our case, we use the a grid in the workspace to place the LEDs a known distance apart. That is, $\boldsymbol{p}_2^R$ is placed in the workspace and considered to be the origin, $\boldsymbol{p}_1^R$ is placed an inch in the robot's negative x-direction, and $\boldsymbol{p}_3^R$ an inch in the negative y-direction. The only difference between this and touching the points is excluding the translation that relates the robot end-effector frame origin to $\boldsymbol{p}_2^R$. However, as mentioned above, we are working with displacements from original positions and thus the translational aspect is not needed.

We create an orthogonal frame of the registration markers exactly as before, this

time as seen in the robot coordinate frame perspective: $\boldsymbol{x}^R$, $\boldsymbol{y}^R$, $\boldsymbol{z}^R$. The rotation relating the marker frame to the robot frame is given by

$$R = \begin{bmatrix} \boldsymbol{x}^R & \boldsymbol{y}^R & \boldsymbol{z}^R \end{bmatrix}.$$

Now we can solve for the relation between the robot and camera frames from $R_C^R = R_M^C {R_M^R}^T$.

The remaining 3D point displacements representing the control points are sent via UDP to our Simulink robot controller. Before sending any data or moving the manipulators, the control point location in the camera coordinate system is averaged over the first 20 images. This location is used as the original, undeformed location for the ith control point, ${\boldsymbol{p}_c^i}^C(0)$. After establishing this initial position, the following captured control point locations in the camera frame, $\boldsymbol{p}^C$, are converted to displacements simply from the difference between the current location and the initial position, ${\boldsymbol{p}_c^i}^C(t) = \boldsymbol{p}^C - {\boldsymbol{p}_c^i}^C(0)$. This displacement is still in the camera coordinate system and therefore is converted to the robot coordinate system with $\boldsymbol{p}_c^i(t) = R_C^R \, {\boldsymbol{p}_c^i}^C(t)$. This displacement is sent over UDP to the Simulink simulation as the control point displacement. Since all experiments occur in the robot's x,y-plane, the z-displacement never changes, and the x and y coordinates of each control point are extracted and used as the plant output $\boldsymbol{y}$ for our output regulation controller.

Appendix C

Experimental Simulink Implementation

The camera-based position feedback, hybrid position/force control, and output regulation control schemes are all integrated using MATLAB's Simulink® and uses QUARC® by Quansar to control the manipulators with the MATLAB Real Time Workshop®. The entire Simulink system is shown in Figure C.1. The main subsystems of this setup include the camera output feedback subsystem (shown in light blue), the robot hybrid control subsystem (shown in orange), and the output regulation controller (shown in teal).

The robot hybrid control subsystem is shown in Figure C.2. The contents of this subsystem are described in Section 5.2.2. The Cartesian position error is projected into the subspace that includes the z, pitch and roll components using a selector matrix and converted to a joint level error. A simple PD controller converts this error to the first half of the control signal for the robot joints. Simultaneously, the force feedback from the end-effector is projected onto the x,y-plane subspace, converted

to a joint level torque error and a simple PI controller converts this to the second half of the control signal for the manipulators. The positional commands keep the manipulator in the x,y-plane of the deformable object. The force commands are determined from the output regulation controller subsystem.

The camera feedback subsystem is shown in Figure C.3. The actual control point position is calculated from the an algorithm running simultaneously and implemented in C++. It uses a stereo vision camera and the image registration described in Appendix B. This control point position calculated in this algorithm is received in Simulink® via the stream server block and UDP communication. The x,y-coordinate(s) of the control point(s) are passed to the output regulation controller subsystem as the output vector $\boldsymbol{y}$.

The output regulation subsystem is shown in Figure C.4. The feedback from the camera and the x,y-displacement of the robot end-effector are both calibrated to be zero when the object is in the undeformed state. The deformation of control points and manipulation points from the original position are concatenated and passed to the output regulation controller as the output measurement $\boldsymbol{y_m}$. The controller and estimator determine the appropriate control force the robots need to apply to the manipulation point to bring the control points to there desired location, as described in 5.1.

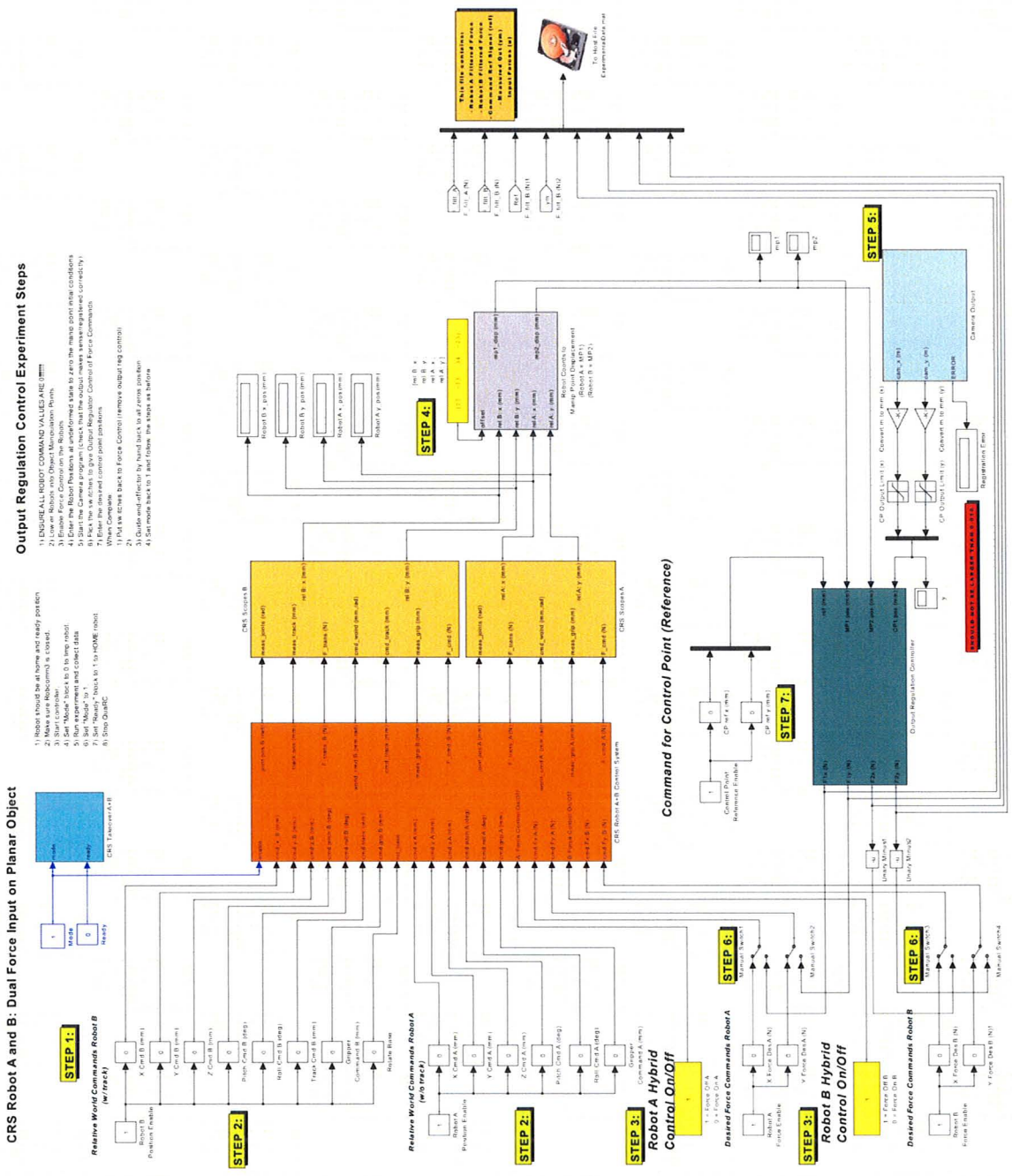


Figure C.1: Simulink block diagram used for experiments.

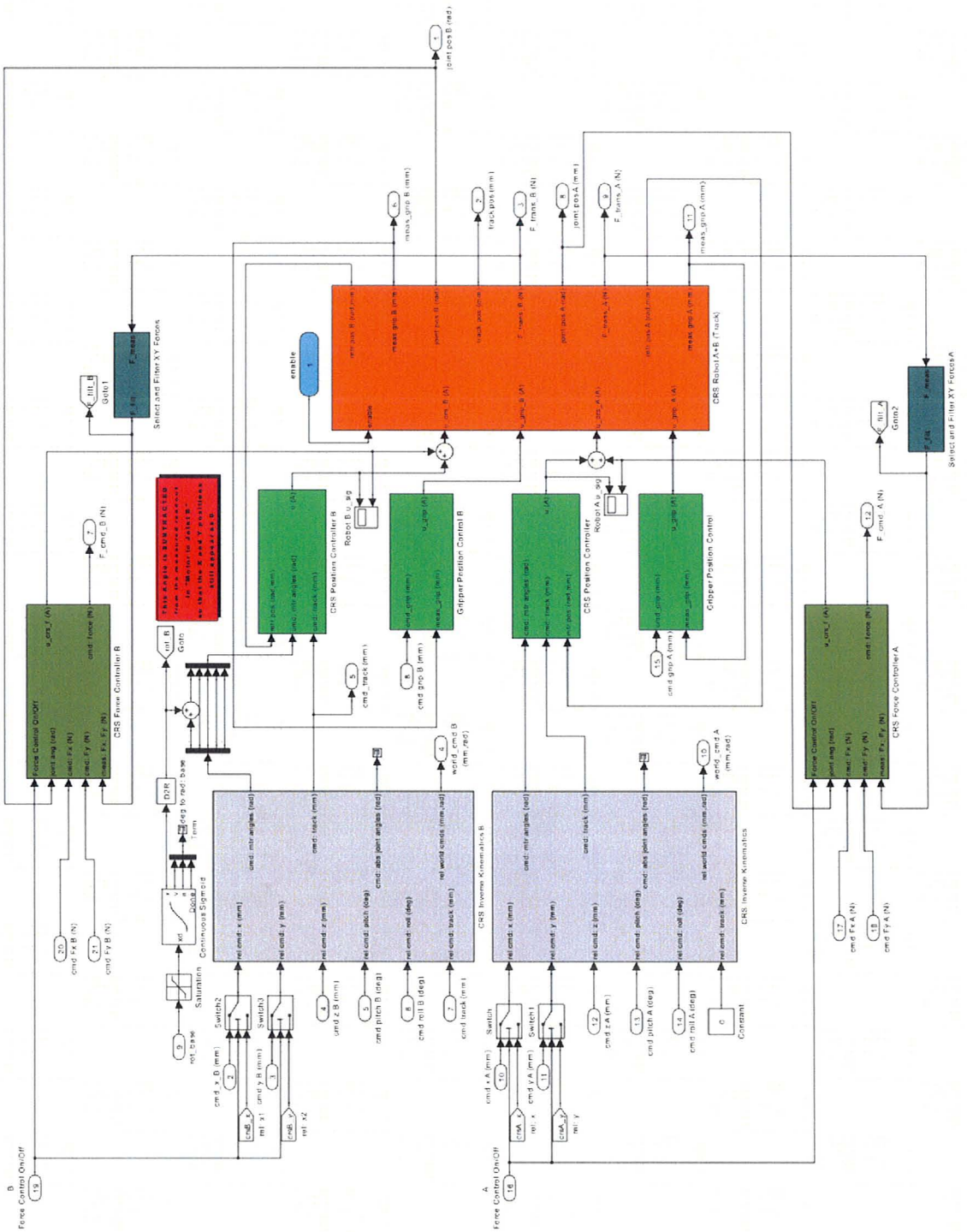


Figure C.2: Robot Hybrid Control Subsystem.

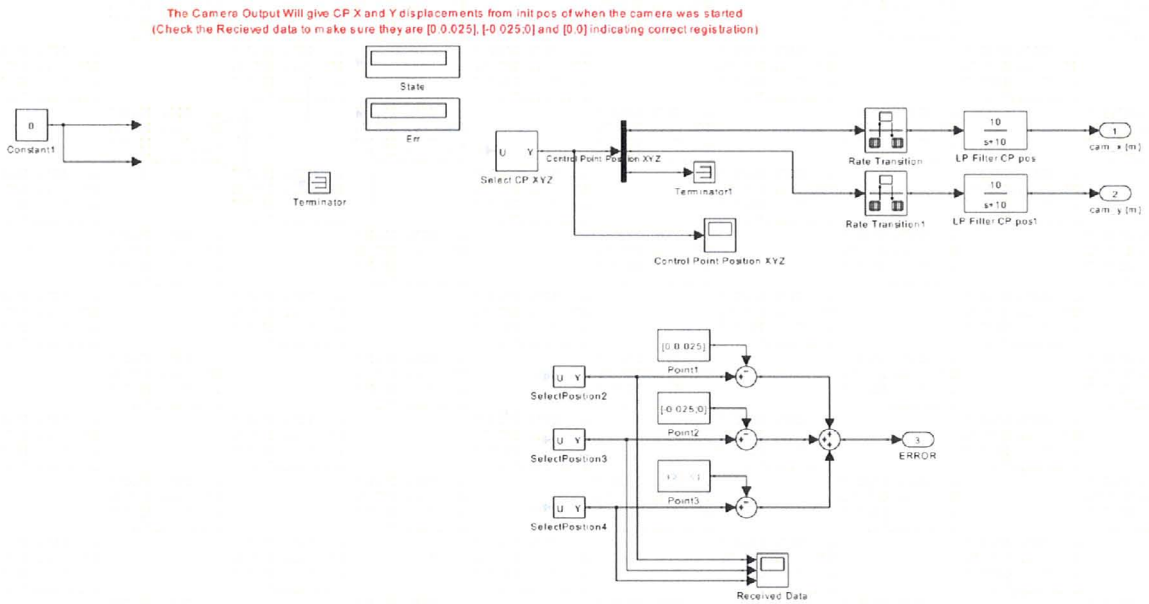


Figure C.3: Camera control point position feedback subsystem.

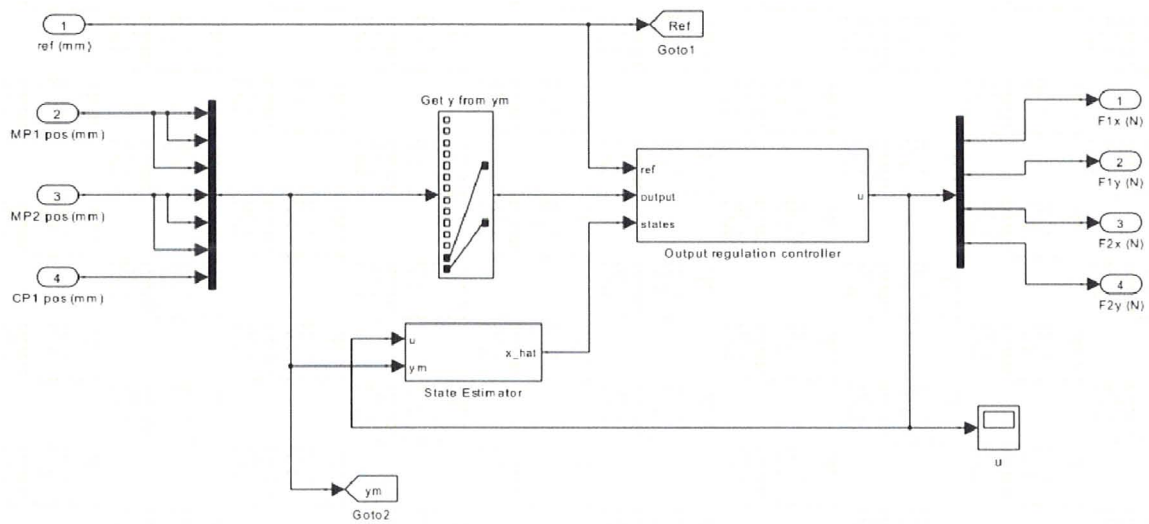


Figure C.4: Output regulation control subsystem.

Bibliography

Antoulas, A. C., Sorensen, D. C., and Gugercin, S. (2001). A survey of model reduction methods for large-scale systems. *Contemporary Mathematics*, **280**, 193–219.

Babuska, I., Banerjee, U., and Osborn, J. E. (2002). Meshless and generalized finite element methods: A survey of some major results. Technical Report 02-03, Institute for Computational Engineering and Sciences; University of Texas at Austin.

Bai, Z. and Su, Y. (2005). Dimension reduction of large-scale second-order dynamical systems via a second-order arnoldi method. *SIAM Journal on Scientific Computing*, **26**(5), 1692–1709.

Belytschko, T., Krongauz, Y., Organ, D., Fleming, M., and Krysl, P. (1996). Meshless methods: An overview and recent developments. *Computer Methods In Applied Mechanics And Engineering*, **139**(1-4), 3–47.

Bro-nielsen, M. and Cotin, S. (1996). Real-time volumetric deformable models for surgery simulation using finite elements and condensation. In *Computer Graphics Forum*, pages 57–66.

Chahlaoui, Y., Gallivan, K. A., Vandendorpe, A., and Dooren, P. V. (2005). *Model*

Reduction of Second-Order Systems, volume 45 of *Lecture Notes in Computational Science and Engineering*, pages 149–172. Springer Berlin Heidelberg.

Chen, C. and Zheng, Y. (1991). Deformation identification and estimation of one-dimensional objects by using vision sensors. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2306–2311.

Chen, J.-S., Pan, C., Wu, C.-T., and Liu, W. K. (1996). Reproducing kernel particle methods for large deformation analysis of non-linear structures. *Computer Methods in Applied Mechanics and Engineering*, **139**, 195–227.

Chen, J.-S., Pan, C., and Wu, C.-T. (1997). Large deformation analysis of rubber based on a reproducing kernel particle method. *Computational Mechanics*, **19**(3), 211–227.

DiMaio, S. P. and Salcudean, S. E. (2003a). Needle insertion modeling and simulation. *IEEE Transactions On Robotics And Automation*, **19**(5), 864–875.

DiMaio, S. P. and Salcudean, S. E. (2003b). Needle steering and model-based trajectory planning. In *Medical Image Computing And Computer-Assisted Intervention (MICCAI)*, volume 2878, pages 33–40.

DiMaio, S. P. and Salcudean, S. E. (2005a). Interactive simulation of needle insertion models. *IEEE Transactions On Biomedical Engineering*, **52**(7), 1167–1179.

DiMaio, S. P. and Salcudean, S. E. (2005b). Needle steering and motion planning in soft tissues. *IEEE Transactions On Biomedical Engineering*, **52**(6), 965–974.

Fisher, W. D. and Mujtaba, M. S. (1992). Hybrid position/force control: A correct formulation. *The International Journal of Robotics Research*, **11**(4), 299–311.

Francis, B. A. (1977). The linear multivariable regulator problem. *SIAM Journal on Control and Optimization*, **15**(3), 486–505.

Gibson, S. F. F. and Mirtich, B. (1997). A survey of deformable modeling in computer graphics. Technical Report TR-97-19, Mitsubishi Electric Research Laboratory(MERL).

Gourret, J.-P., Thalmann, N. M., and Thalmann, D. (1989). Simulation of object and human skin formations in a grasping task. *SIGGRAPH Comput. Graph.*, **23**(3), 21–30.

Henrich, D. and Worn, H. (2000). *Robot Manipulation of Deformable Objects*. Springer-Verlag.

Huang, J. (2004). *Nonlinear Output Regulation: Theory and Applications*. SIAM, Philadelphia, PA.

Isidori, A., Marconi, L., and Serrani, A. (2003). *Robust Autonomous Guidance: An Internal Model Approach*. Springer.

Li, R.-C. and Bai, Z. (2005). Structure-preserving model reduction using a krylov subspace projection formulation. *Commun. Math. Sci.*, **3**(2), 179–199.

Li, R.-C. and Bai, Z. (2006). Structure-preserving model reduction. In *Applied Parallel Computing*, volume 3732 of *Lecture Notes in Computer Science*, pages 323–332. Springer Berlin / Heidelberg.

Lin, Y., Bao, L., and Wei, Y. (2007). Model-order reduction of large-scale second-order mimo dynamical systems via a block second-order arnoldi method. *Int. J. Comput. Math.*, **84**(7), 1003–1019.

Mallapragada, V., Sarkar, N., and Podder, T. K. (2007). A robotic system for real-time tumor manipulation during image guided breast biopsy. In *Proc. of the IEEE International Conference on Bioinformatics and Bioengineering*, pages 204–210.

Nakagaki, H., Kitagaki, K., Ogasawara, T., and Tsukune, H. (1997). Study of deformation and insertion tasks of a flexible wire. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2397 –2402 vol.3.

Platt, S. M. and Badler, N. I. (1981). Animating facial expressions. *Computer Graphics*, **15**(3), 245–252.

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

Saadat, M. and Nan, P. (2002). Industrial application of automatic manipulation of flexible materials. *Industrial Robot: An International Journal*, **29**(5), 434–442.

Shibata, M. and Hirai, S. (2006). Soft object manipulation by simultaneous control of motion and deformation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2460– 2465.

Smolen, J. and Patriciu, A. (2009a). Deformation planning for robotic soft tissue manipulation. In *Proc. Int. Conf. on Advances in Computer-Human Interactions (ACHI)*, pages 199–204.

Smolen, J. and Patriciu, A. (2009b). Model based stabilization of soft tissue targets in needle insertion procedures. In *Proc. of the International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*.

Su, T.-J. and Craig, R. R. (1991). Model reduction and control of flexible structures using krylov vectors. *Journal of Guidance, Control, and Dynamics*, **14**(2), 260–267.

Terzopoulos, D. and Waters, K. (1990). Physically-based facial modeling, analysis, and animation. *Journal of Visualization and Computer Animation*, **1**(2), 73–80.

Torabi, M., Hauser, K., Alterovitz, R., Duindam, V., and Goldberg, K. (2009). Guiding medical needles using single-point tissue manipulation. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2705–2710.

Wada, T., Hirai, S., and Kawamura, S. (1998). Indirect simultaneous positioning operations of extensionally deformable objects. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1333–1338.

Wada, T., Hirai, S., and Kawamura, S. (1999). Planning and control of indirect simultaneous positioning operation for deformable objects. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 2572–2577 vol.4.

Wada, T., Hirai, S., Kawamura, S., and Kamiji, N. (2001). Robust manipulation of deformable objects by a simple pid feedback. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 85– 90.

Wakamatsu, H. and Hirai, S. (2004). Static modeling of linear object deformation based on differential geometry. *International Journal Of Robotics Research*, **23**(3), 293–311.

Wakamatsu, H., Hirai, S., and Iwata, K. (1995). Modeling of linear objects considering bend, twist, and extensional deformations. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 433 –438 vol.1.

Wakamatsu, H., Hirai, S., and Iwata, K. (1996). Static analysis of deformable object grasping based on bounded force closure. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3324 –3329 vol.4.

Wakamatsu, H., Tanaka, Y., Tsumaya, A., Shirase, K., and Arai, E. (2002). Representation and planning of deformable linear object manipulation including knotting. In *Industrial Technology, 2002. IEEE ICIT '02. 2002 IEEE International Conference on*, volume 2, pages 1321–1326 vol.2.

Zheng, Y., Pei, R., and Chen, C. (1991). Strategies for automatic assembly of deformable objects. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 2598–2603.